Animation Transplantation

Jochen Süßmuth and Michael Zollhöfer and Günther Greiner

Computer Graphics Group, University Erlangen-Nuremberg, Germany

This is a pre-peer reviewed version of the following article
Animation Transplantation in Computer Animation and Virtual Worlds
which has been published in final form at http://dx.doi.org/10.1002/cav.364

Abstract

We present a novel method to animate a static geometry by cloning a captured animation sequence. More precisely, given a sequence of range scans of a deforming object which has been captured by a real-time 3D scanner, we describe a novel algorithm to clone the animation of the recorded geometry onto another triangle mesh. To achieve this, we reconstruct a coherent animated mesh of the input sequence using a template deformation approach. Then we employ a new algorithm for robust marker-less non-rigid registration to deform one frame of the generated animation such that it matches a different 3D model.The resulting registration is further used to find correspondences between the animation and the target object which are in turn used to transfer the animation of the recorded sequence onto the target shape.

Transferring the entire geometry of the animations results in very convincing facial expressions since even the smallest expression wrinkles are preserved. We evaluate the robustness and the performance of the proposed algorithms using a variety of data sets, including facial animations and whole body animations.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Animation—Computer Graphics [I.3.5]: Curve, surface, solid, and object representations—

1. Introduction

An essential step in the production of digital content for feature films and computer games is the generation of realistic and lifelike animations of human characters. For full body animations, skeletal shape deformation has established as de-facto standard as it offers numerous advantages: Performances can be recorded relatively easily using marker based motion capturing techniques. Furthermore, skeletal animations can be efficiently transferred onto different characters as it involves only an additional skinning step. However, skeletal animation has a major drawback: Traditional methods for skeletal deformation are not able to reproduce effects like muscle bulge or skin wrinkles. Since humans are very susceptible to subtle changes of facial expressions, facial performances cannot be handled adequately using skeletal deformations.

To capture facial animations, usually several hundred markers are applied to an actor's face and tracked by high-resolution cameras. The captured marker positions are then used to drive the animation of a digitized model of the actor's face. To determine how the model deforms between the recorded markers, various heuristics can be used. However, setting up the markers and registering them with the digital face model is very cumbersome. Furthermore, transferring the recorded animations to a different face model is a tedious

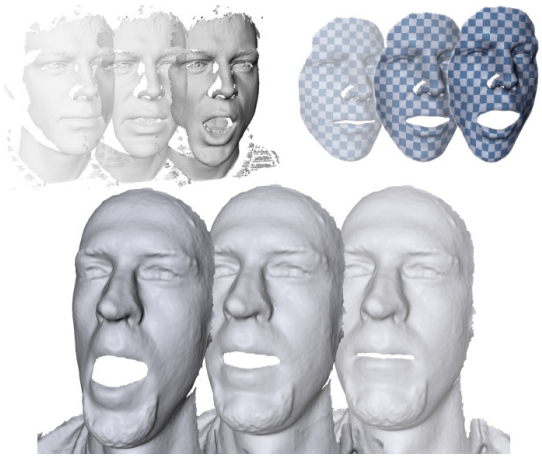

Figure 1: *Input range scans, reconstructed animation and the animation transplanted onto a different model.*

and time consuming task and requires longtime experience and artistic skills.

We provide a simple solution for capturing and reusing facial expressions and other soft body animations: Given as input a sequence of range scans recorded by a low-cost real-time 3D scanner (e.g. [WLG07], [ZH06]), we reconstruct an animated mesh which reproduces the recorded animation. By non-rigidly registering one frame of the obtained animation to a target triangle mesh, we obtain a dense set of correspondences between the animation and the target mesh. These correspondences are further used to adapt and transfer the entire animation onto the target mesh. To our knowledge, our processing pipeline is the first fully automatic makerless system for animation recording and re-targeting which is based solely on geometric matching.

The main technical contributions of this paper are:

Animation Reconstruction: We describe a fast and simple algorithm for computing a coherent animated mesh from real-time 3D scanner data. Our algorithm employs an energy minimization approach to fit a template mesh to the input data and is thereby able to accurately reconstruct the motion of small-scale features such as wrinkles and lip movement.

Robust Deformable Registration: We present a new method for robust deformable surface registration based on energy minimization. Using a reduced deformable model, our algorithm can non-rigidly align models which are in significantly different poses. Unlike previous approaches, our algorithm is not restricted to any specific input data or class of deformations.

Animation Transfer: We show that the previously presented methods for animation reconstruction and non-rigid model alignment are the key ingredients to animation retargeting and derive a simple algorithm for transferring animations onto different target meshes.

1.1. Related Work

Capturing, analyzing and synthesizing the motion of real-life objects has long been discussed in the computer vision and computer graphics communities and a huge body of work has been published over the last decades. Most of the proposed methods build on marker-based or feature-based motion tracking in combination with a linear face model. For a recent overview of these methods, see [ZSCS04, ZH06, BBA*07, BLB*08, MJC*08] and references therein.

Only recently, marker-less methods for facial motion capturing and expression transfer have been introduced: Wang and colleagues [WGQ08] combine a feature tracking approach with least-squares conformal maps to compute dense inter-frame correspondences for facial animations. Given a sparse set of user specified correspondences, they compute dense inter-face correspondences and transfer facial expressions using the method of Noh and Neumann [NN01]. Their method is very susceptible to the placement of the user defined correspondences and according to our experience, the expression cloning technique works only well for moderate deformations (cf. Figure 7). More similar to our method is the real time system for facial expression transfer by Weise et al. [WLGP09]: They track facial expressions using a combination of geometric and texture features. Facial expressions are then mapped onto another character using a variant of the technique proposed by Sumner et al. [SP04]. Due to the real-time requirements, they are forced to work with reduced deformation models which cannot reproduce small-scale detail like wrinkles properly.

Pairwise non-Rigid Registration
Based on the *iterative closest points* (ICP) algorithm [CM92, BM92], a number of techniques has been developed which aim at the non-rigid pairwise alignment of surfaces. These algorithms mainly differ in the way they compute correspondences and in the deformation model that they are based on. Li and co-workers [LSP08] formulate the non-rigid registration as a single non-linear optimization problem which solves for correspondences and deformation simultaneously. However, their method is limited to the special case of registering depth images moreover, it is rather complicated to implement. Huang et al. [HAWG08] use feature vectors of a sub-set of the input vertices to compute dense initial correspondences. A set of correspondences which are consistent w.r.t the geodesic distances on the surfaces is then extracted and used as position constraints to deform the source using a deformation graph approach [SSP07]. Chang and Zwicker [CZ08] present a method for the registration of articulated shapes where they compute a set of candidate rigid motions for each vertex. A graph cut algorithm is then used to compute a consistent deformation field. In a follow-up paper [CZ09], they model the motion of the underlying object using a volumetric reduced deformable model, yielding a significant performance boost and more robust registrations.

Animation Reconstruction
Animation reconstruction can be considered as a special case of non-rigid registration: instead of aligning two surfaces, the goal of animation reconstruction is to compute a coherent deforming model which aligns to all input frames of an animation. A method for reconstructing a coherent deforming mesh model from a sequence of point clouds was first presented by Shinya [Shi04], where a template mesh is tracked over time using a gradient descent optimization. Wand and co-workers [WJH*07] use a statistical framework to fit surfel trajectories to the sequence of point clouds. The huge computation costs of this approach are addressed in a follow-up publication [WAO*09], where a new reduced deformation model is used. Following Mitra et al. [MFO*07], Süßmuth and colleagues [SWG08] compute an implicit four-dimensional space-time surface from the input data and slide a template mesh along that surface while enforcing local rigidity. The time-space setting restricts their method to sequences with rather small inter-frame motion. More recently,

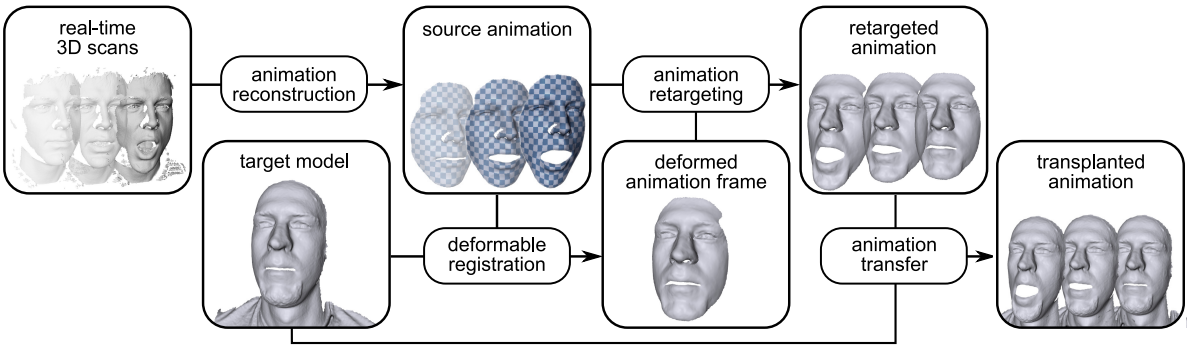

Figure 2: *Animation transplantation pipeline: An animated mesh is computed from a set of range scans. One frame of the animated mesh is then fitted to the target mesh and the animation is deformed accordingly. Finally, the animation is cloned onto the target mesh.*

Li et al. [LAGP09] propose a two step process for animation reconstruction. First they fit a smooth template mesh to the input frames using a variant of their previous work [LSP08]. In a second run, high frequency details are recovered by computing displacement vectors for all template vertices on a per frame-basis. The displacement vectors are then diffused to neighboring frames, thus resulting in a smooth animation. The disadvantage of this approach is that small details such as expression wrinkles are only superimposed rather than tracked.

1.2. Overview

An overview of our processing pipeline is depicted in Figure 2. Our new algorithm for reconstructing a deformable model from a sequence of real-time 3D scans is presented in Section 2. In Section 3, we introduce a simple and robust method for deformable registration. We then use the deformable registration to clone animation sequences onto other triangle meshes (Section 4). We present results in Section 5 and conclude our work in Section 6.

2. Animation Reconstruction

Recent advances in 3D scanning technology, e.g. fast structured light and space-time stereo scanners [WLG07, ZSCS04], facilitate capturing high quality 3D geometry and animation simultaneously. The output of real-time 3D scanners is typically a set of densely sampled 3D point clouds without any correspondence information across frames. In the first step of our processing pipeline, we establish a dense set of inter-frame correspondences by computing a temporal-coherent geometric model from the uncorrelated input range-scans.

To recover the animation of the scanned object, we successively deform an initial template mesh such that it fits the individual frames of the animation. For all examples in this paper, we computed the template mesh directly from the first input frame using standard surface reconstruction techniques. Marching cubes artifacts are eliminated by uniformly re-meshing the template mesh.

Assume, we are given a template mesh $\mathcal{M}$ and a set of sample points $\mathcal{P}$, furtheron called the *pose*. Both, the mesh and the point cloud, are discrete approximations of the same object $\mathcal{O}$ which have been recorded at different instants in time. Our goal is to recover the motion of the object $\mathcal{O}$ between the recording of $\mathcal{M}$ and $\mathcal{P}$. Therefor, we try to compute a natural deformation $\mathcal{D}$ which transforms the template mesh $\mathcal{M}$ such that it matches the pose described by the point cloud $\mathcal{P}$. The pose is matched if the distance between the vertices of $\mathcal{M}$ and the points of $\mathcal{P}$ is minimized and if the deformation changes $\mathcal{M}$ as little as possible while preserving small-scale surface detail.

To measure the distance between the mesh and the point cloud, we use the hierarchical RBF fitting described in [SMG] to compute a signed distance function $f_{\mathcal{P}} : \mathbb{R}^3 \to \mathbb{R}$ for the point cloud. Using this distance function, we can define the total distance $E_{\text{dist}}(\mathcal{M}, \mathcal{P})$ between the mesh $\mathcal{M}$ and the point cloud $\mathcal{P}$ as

$$E_{\text{dist}}(\mathcal{M}, \mathcal{P}) = \sum_{i=1}^{n} f_{\mathcal{P}}(\mathbf{v}_i)^2. \tag{1}$$

We use the *as-rigid-as-possible* deformation paradigm [SA07] to measure the magnitude of a deformation. Let $\mathcal{M}$ be the template mesh and let $\hat{\mathcal{M}}$ be a deformed instance of $\mathcal{M}$ with n vertices $\mathbf{v}_i$ and $\hat{\mathbf{v}}_i$, respectively. Sorkine et al. define a deformation error $E_{\text{def}}(\hat{\mathcal{M}}, \mathcal{M})$ between two meshes as the sum over the individual vertex errors. The vertex error is thereby defined as the (squared) deviation of the vertex's fans deformation between $\mathbf{v}_i$ and $\hat{\mathbf{v}}_i$ from being rigid:

$$E_{\text{def}}^{\text{vert}}(\hat{\mathbf{v}}_i, \mathbf{v}_i) = \sum_{\mathbf{v}_j \in \mathcal{N}(\mathbf{v}_i)} \omega_{ij} \left\| (\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j) - \mathbf{R}_i(\mathbf{v}_i - \mathbf{v}_j) \right\|_2^2,$$

where $\mathbf{R}_i$ is the rigid transformation which aligns the fan

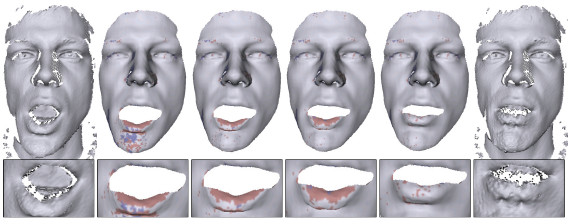

Figure 3: *A single step of our animation reconstruction algorithm: the left and right images show two succeeding input frames. Intermediate registration results after 1,3,5 and 7 iteration steps are shown in between.*

of $\mathbf{v}_i$ best with the fan of $\hat{\mathbf{v}}_i$. $\mathcal{N}(\mathbf{v}_i)$ is the 1-neighborhood of vertex $\mathbf{v}_i$ and ω_{ij} is the weight of the edge connecting the vertices $\mathbf{v}_i$ and $\mathbf{v}_j$. Thus, the deformation error for two meshes is then given as:

$$E_{\text{def}}(\hat{\mathcal{M}},\mathcal{M}) = \sum_{i=1}^{n} E_{\text{def}}^{\text{vert}}(\hat{\mathbf{v}}_i,\mathbf{v}_i). \tag{2}$$

Given an initial template mesh $\mathcal{M}$, a target pose $\mathcal{P}$ and a deformed mesh $\hat{\mathcal{M}}$, we can define a measure for the quality of a non-rigid registration using the energy terms defined in Equations (1) and (2):

$$\begin{aligned} E(\hat{\mathcal{M}},\mathcal{M},\mathcal{P}) &= \alpha \cdot E_{\text{dist}}(\hat{\mathcal{M}},\mathcal{P}) + E_{\text{def}}(\hat{\mathcal{M}},\mathcal{M}) \\ &= \sum_{i=1}^{n} \left(\alpha \cdot f_{\mathcal{P}}(\hat{\mathbf{v}}_i)^2 + E_{\text{def}}^{\text{vert}}(\hat{\mathbf{v}}_i,\mathbf{v}_i) \right). \end{aligned} \tag{3}$$

The parameter α can be used as a stiffness parameter to balance between a tight fitting and a small deformation error.

A suitable deformation which maps the mesh $\hat{\mathcal{M}}$ onto the point cloud $\mathcal{P}$ can now be found by minimizing the registration error in Equation (4). To solve the resulting non-linear minimization problem, we use iterative flip-flop optimization [SA07, SWG08] which solves for the unknown rigid transformations $\mathbf{R}_i$ first (cf. [SA07]) and uses them to compute improved vertex positions. To find the vertex positions $\hat{\mathbf{v}}_i$ which minimize the registration error E for given $\mathbf{R}_i$'s, we compute the derivative of E w.r.t. $\hat{\mathbf{v}}_i$. The derivative of E_{def} can be found in [SA07]:

$$\frac{\partial E_{\text{def}}}{\partial \hat{\mathbf{v}}_i} = \sum_{\mathbf{v}_j \in \mathcal{N}(\mathbf{v}_i)} 4\omega_{ij} \left((\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j) - \frac{\mathbf{R}_i + \mathbf{R}_j}{2} (\mathbf{v}_i - \mathbf{v}_j) \right).$$

To derive E_{dist} for $\hat{\mathbf{v}}_i$, we approximate the distance function $f_{\mathcal{P}}(\mathbf{x})$ by a Taylor series of first order which yields

$$\hat{f}_{\mathcal{P}}(\hat{\mathbf{v}}_i) = \left\langle (\hat{\mathbf{v}}_i^{\text{old}} - \hat{\mathbf{v}}_i), \mathbf{n}_i^{\text{old}} \right\rangle - f_i^{\text{old}}.$$

f_i^{old} is the function value and $\mathbf{n}_i^{\text{old}}$ the normalized gradient at the current vertex position $\hat{\mathbf{v}}_i^{\text{old}}$. Thus E_{dist} becomes

$$\begin{aligned} E_{\text{dist}} &= \sum_{i=1}^{n} \Bigg(\underbrace{\left\langle \hat{\mathbf{v}}_i^{\text{old}}, \mathbf{n}_i^{\text{old}} \right\rangle - f_i^{\text{old}}}_{c_i} - \left\langle \hat{\mathbf{v}}_i, \mathbf{n}_i^{\text{old}} \right\rangle \Bigg)^2 \\ &= \sum_{i=1}^{n} \left(c_i - \left\langle \hat{\mathbf{v}}_i, \mathbf{n}_i^{\text{old}} \right\rangle \right)^2 \end{aligned}$$

and the derivative of E_{dist} w.r.t. $\hat{\mathbf{v}}_i$ is given by

$$\frac{\partial E_{\text{dist}}}{\partial \hat{\mathbf{v}}_i} = -2c_i\mathbf{n}_i^{\text{old}} + 2\mathbf{n}_i^{\text{old}}(\mathbf{n}_i^{\text{old}})^T\hat{\mathbf{v}}_i. \tag{4}$$

Setting the partial derivatives $\frac{\partial E}{\partial \hat{\mathbf{v}}_i} = \frac{\partial E_{\text{def}}}{\partial \hat{\mathbf{v}}_i} + \alpha \frac{\partial E_{\text{dist}}}{\partial \hat{\mathbf{v}}_i}$ w.r.t. each $\hat{\mathbf{v}}_i$ to zero, we arrive at the following $(3n \times 3n)$ system of linear equations:

$$\begin{aligned} &\alpha\mathbf{n}_i^{\text{old}}(\mathbf{n}_i^{\text{old}})^T\hat{\mathbf{v}}_i + \sum_{\mathbf{v}_j \in \mathcal{N}(\mathbf{v}_i)} 4\omega_{ij}(\hat{\mathbf{v}}_i - \hat{\mathbf{v}}_j) \\ &= \\ &\alpha c_i\mathbf{n}_i^{\text{old}} + \sum_{\mathbf{v}_j \in \mathcal{N}(\mathbf{v}_i)} 2\omega_{ij}(\mathbf{R}_i - \mathbf{R}_j)(\mathbf{v}_i - \mathbf{v}_j). \end{aligned} \tag{5}$$

The system matrix in (5) is sparse and positive definite and can be efficiently solved using a sparse Cholesky factorization [Tol03]. Solving (5) for $\hat{\mathbf{v}}_i$ yields new vertex positions of the warped template mesh $\hat{\mathcal{M}}$ which reduce the registration error in Equation (4). We iterate the computation of the rigid transformations $\mathbf{R}_i$ and the determination of new vertex positions until the total vertex movement between two iteration steps falls under a certain threshold. In practice, the iteration usually terminates after 3-5 steps.

Minimizing the distance between the current mesh and the point cloud is similar to a non-rigid ICP algorithm which uses a point to plane matching for correspondence finding. To improve the robustness of our method, we extend the previously presented algorithm by pruning correspondences which are probably bad matches. During the computation of the Taylor-expansion of the distance function, we compute a pseudo correspondence vertex $\mathbf{v}_i^{\text{pc}} = \mathbf{v}_i^{\text{old}} - f_i^{\text{old}}\mathbf{n}_i^{\text{old}}$ for each vertex $\hat{\mathbf{v}}_i$. If the distance of $\mathbf{v}_i^{\text{pc}}$ to the closest point of the target point cloud $\mathcal{P}$ is larger than three times the average spacing between two neighboring points in $\mathcal{P}$, it is likely that the pseudo correspondence lies inside a hole and therefor the correspondence is pruned. We further prune correspondences if the angle between the mesh vertex and the gradient at $\mathbf{v}_i^{\text{pc}}$ is larger then 45 degrees. To prune a correspondence, it is sufficient to set the gradient $\mathbf{n}_i^{\text{old}}$ of the respective vertex to zero as this eliminates the influence of the data fitting term from the linear system in Equation (5). In Figure 3, we show some intermediate steps of a single template propagation iteration. Vertices for which correspondences have been pruned due to the distance criterion are shown in blue, those that have been pruned because of the angle criterion are shown in red.

Results of the proposed animation reconstruction are

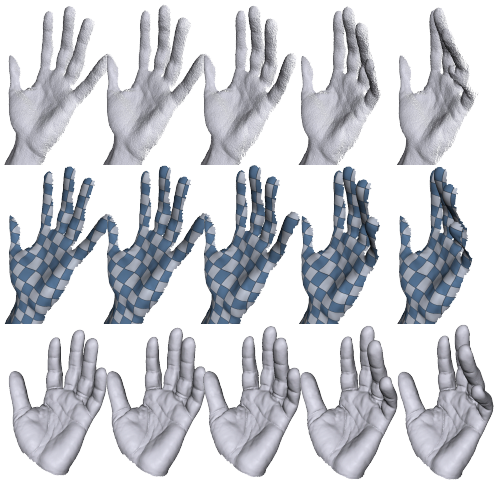

Figure 4: *Reconstructing and transplanting a hand animation: the first row shows the input scans, the reconstructed animated mesh is shown in the middle. The bottom row shows the animation transplanted onto another hand model.*

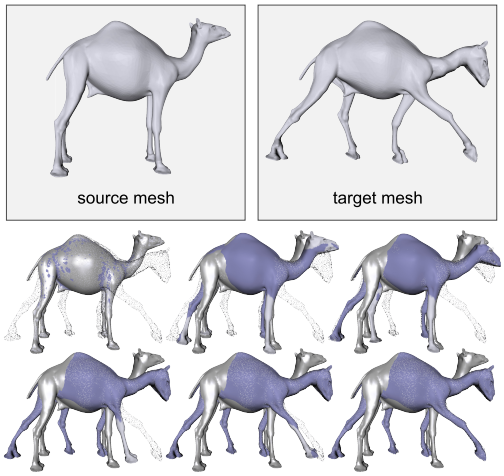

Figure 5: *Convergence behavior of the non-rigid registration algorithm. Active correspondences are highlighted blue.*

shown in Figures 4 and 9 and in the accompanying video. Our animation reconstruction technique is very efficient in terms of memory requirements and computation times. The memory consumption is low as we only need to keep the target point cloud of a single time step in memory. The template warping step took on average 5.2 seconds per frame for the hand data set and 12.3 seconds for the face data set. Experiments have shown that setting stiffness parameter α in Equation (4) to 1.0 worked best for our examples. A limitation of our template based approach is that we can only track geometry which is visible in the first frame. However, this problem could be solved by simply using a static multi-view reconstruction of the scanned object as initial template mesh as done in [LAGP09].

3. Robust Deformable Registration

To be able to transfer the reconstructed animation onto another target surface, we need to estimate dense inter-surface correspondences between the source animation and the target surface. We compute these correspondences by means of deformable registration. The algorithm that we used for warping the template mesh in the animation reconstruction step involves a non-linear optimization over all vertex positions. As this procedure is too unstable to align significantly different shapes, we present a more robust non-rigid registration technique which uses a reduced deformation model in this section.

Sumner and co-workers [SSP07] introduce *deformation graphs* for representing space deformations. A deformation graph $\mathcal{DG}$ consists of a set of nodes $\mathcal{N}$, which are placed on a subset of the vertices of the input mesh. Nearby nodes are connected by edges $e_{ij} \in \mathcal{E}$ in the graph. With every node n_i at position $\mathbf{p}_i$ a local affine transformation $A_i(\mathbf{x}) = \mathbf{M}_i(\mathbf{x} - \mathbf{p}_i) + \mathbf{p}_i + \mathbf{t_i}$, which describes how the surrounding space deforms, is associated. A global deformation at an arbitrary position $\mathbf{x}$ is computed by blending the affine transformations of the k nearest nodes $n_j \in \mathcal{I}_\mathbf{x}$ of $\mathbf{x}$:

$$\mathrm{Trans}_{\mathcal{DG}}(\mathbf{x}) = \sum_{n_j \in \mathcal{I}_\mathbf{x}} \omega(n_j, \mathbf{x}) \cdot A_j(\mathbf{x}),$$

and $\omega(n_j, \mathbf{x})$ is a positive weight.

The deformation graph allows us to measure the error introduced by a deformation. To preserve small-scale detail of the surface, it is important that the surface deforms locally rigid. This is exactly the case if $\mathbf{M}_i$ describes a rotation, i.e. $\mathbf{M}_i$ is orthogonal. Thus the deviation of a node n_i's transformation from being locally rigid can be defined as:

$$\begin{aligned}\mathrm{Rot}(n_i) = {} & \langle \mathbf{q}, \mathbf{r} \rangle^2 + \langle \mathbf{r}, \mathbf{s} \rangle^2 + \langle \mathbf{s}, \mathbf{q} \rangle^2 \\ & + (\langle \mathbf{q}, \mathbf{q} \rangle - 1)^2 + (\langle \mathbf{r}, \mathbf{r} \rangle - 1)^2 + (\langle \mathbf{s}, \mathbf{s} \rangle - 1)^2,\end{aligned}$$

where $\mathbf{q}, \mathbf{r}$ and $\mathbf{s}$ are the columns of n_i's transformation matrix $\mathbf{M}_i$. As neighboring nodes influence similar regions of the model, it is important that the computed transformations are consistent w.r.t. one another. To ensure this consistency, Sumner et al. define an energy function which punishes derivations between the transformations of two neighboring nodes n_i and n_j connected by the edge e_{ij} as

$$\mathrm{Con}(e_{ij}) = \left\| A_i(\mathbf{p_j}) - A_j(\mathbf{p_j}) \right\|_2^2 + \left\| A_j(\mathbf{p_i}) - A_i(\mathbf{p_i}) \right\|_2^2.$$

The energy term Con is defined for each edge and Rot is defined for each node of the deformation graph. By combining the two, we obtain a global indicator E_{def2} for the magnitude of the deformation induced by a deformation graph:

$$E_{\mathrm{def2}}(\mathcal{DG}) = \omega_{rot} \sum_{n_i \in \mathcal{N}} \mathrm{Rot}(n_i) + \omega_{con} \sum_{e_{ij} \in \mathcal{E}} \mathrm{Con}(e_{ij}).$$

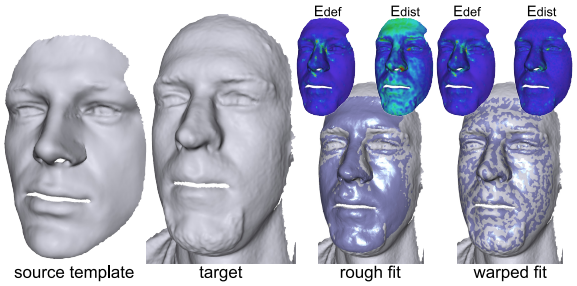

Figure 6: *Coarse registration and final warp: The resulting meshes of the respective registration steps are overlayed as blue meshes on top of the target surface.*

In order to utilize deformation graphs for non-rigid registration, we need to introduce an energy term which pulls the source mesh $\mathcal{M}$ towards the target mesh. By converting the target mesh into a point cloud $\mathcal{P}$, we are able to reuse the energy term introduced in Equation (1) with minor modifications:

$$E_{\text{dist2}}(\mathcal{M}, \mathcal{DG}, \mathcal{P}) = \sum_{\mathbf{v}_i \in \mathcal{M}} f_{\mathcal{P}}(\text{Trans}_{\mathcal{DG}}(\mathbf{v}_i))^2.$$

We use the same tests that were already used in Section 2 to discard poor correspondences that are either too far apart or have incompatible normals.

To perform the registration, we compute the unknown affine transformations of the deformation graph such that the overall registration error

$$E_{\text{reg2}}(\mathcal{M}, \mathcal{DG}, \mathcal{P}) = \omega_{\text{dist}} E_{\text{dist2}}(\mathcal{M}, \mathcal{DG}, \mathcal{P}) + E_{\text{def2}}(\mathcal{DG})$$

is minimized. The minimization of E_{reg2} leads to a non-linear optimization problem which can be solved using the Gauss-Newton method. In our implementation, we set $\omega_{\text{rot}} = 1$ and $\omega_{\text{con}} = 10$. The fitting weight ω_{dist} is initially set to $1/128$ and then doubled during each step of the Gauss-Newton algorithm until it reaches an upper bound of 32. This parameter adjustment, which was also proposed in [LSP08], can help to avoid local minima as the model deforms almost rigidly in the beginning. By increasing ω_{dist}, the fitting energy function becomes more dominant and the model deforms more elastic. An example registration which illustrates the convergence of our iterative non-rigid registration method is shown in Figure 5 and in the accompanying video.

The proposed non-rigid registration technique computes a coarse non-rigid alignment but will fail to register small detail. To improve the small-scale alignment, we use the algorithm from Section 2 to perform a final registration step. The effect of this final warp is shown in Figure 6. The error terms E_{dist} and E_{reg} have been computed according to Section 2.

4. Animation Transplantation

Given the reconstructed animation and a target object, the goal of animation transplantation is to transfer the source animation onto the target mesh. Thereby it is important to adapt the animation to the characteristics of the target model, e.g. when transferring the gallop animation of a horse model onto a dachshund, the steps must be scaled such that they match the short legs of the dog.

Animation transplantation is realized as a two steps procedure: In an *animation retargeting* step, we first adapt the animation to the target mesh, and then *transfer* the retargeted animation to the target mesh to obtain our final animation.

4.1. Animation Retargeting

By deforming a single frame of the animation to match the target surface using the algorithm for robust deformable registration presented in Section 3, we obtain a deformed reference mesh $\mathcal{M}_R$ which resembles the target geometry while having the topology of the source animation. Different proportions of the source and the target model are thereby compensated by stretching or squeezing the triangles in the respective regions.

We then apply the animation of the source mesh to the deformed reference mesh $\mathcal{M}_R$ by successively transferring the change in the pose between two frames in the source animation to target mesh. A comparison of various approaches for deformation transfer is shown in Figure 7: The deformation exhibited by the source mesh shown in the two leftmost images is transferred onto the target mesh shown in the third image. The next image shows the result of naive deformation transfer where the vertex displacements between two frames of the animation are directly applied to the target mesh. Noh and Neumann [NN01] extend the naive approach by scaling and rotating the displacement vectors according to local surface deformation (fifth image in Figure 7). For all results in this paper, we used the deformation gradient based approach of Sumner and Popovic [SP04] which performs best as it preserves local features best. However, as deformation transfer does only transfer the changes in pose, it does not determine a global position for the model. We find the global positioning for the target mesh by applying the extracted rigid part of the source deformation to the target mesh.

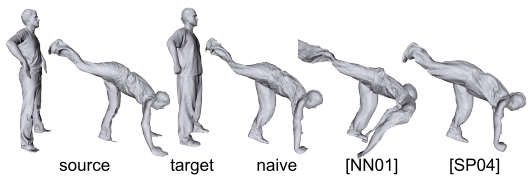

Figure 7: *Various approaches for deformation transfer.*

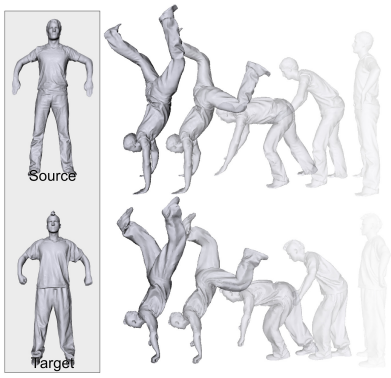

Figure 8: *Transferring a whole body animation onto a second character.*

4.2. Animation Transfer

Animation retargeting has deformed and adapted the input animation such that it suits the target mesh (cf. Figure 2). In this step, we transfer the thereby generated animation onto the final target model. As the static target mesh $\mathcal{M}_T$ is usually more detailed and contains more vertices then the source animation, we pick a subset of the target models vertices to drive the animation.

By marking every vertex of the target shape $\mathcal{M}_T$ which is the closest point to a vertex of the deformed reference mesh $\mathcal{M}_R$, we obtain a set of vertices $\mathcal{V}_C$ which are representative for the vertices of $\mathcal{M}_R$. For each of these vertices, we locate the closest point on a triangle of $\mathcal{M}_R$ and store the triangle index as well as the barycentric coordinates of the respective point. For each frame of the animation, we displace the vertices in $\mathcal{V}_C$ according to the movement of their corresponding points. As both the target shape and the deformed reference mesh are properly aligned and represent the same geometry, it is not necessary to adapt the length or the direction of the displacement vectors. The remaining vertices are positioned such that the mesh deforms locally rigid and preserves small-scale detail using the optimizations of [SSP07] and [SA07].

5. Results

We used the proposed framework to reconstruct and transplant various animations which have been acquired using different devices. Table 1 summarizes the statistics of the data sets shown in this paper. The first row contains the number of frames in the animation. The next two rows contain the total number of points in the input scans and the time it took to reconstruct the animation where applicable. $\text{vert}_{\text{source}}$ and $\text{vert}_{\text{target}}$ are the number of points of the input animated mesh and the target mesh respectively. The last three rows contain the computation times for registering the animation and the target mesh, animation retargeting and transferring the animations onto the target mesh. All computations were performed on a 2.93 GHz Core i7 machine with 6 GB of RAM and the timings are given in minutes.

data set	Hand	Face_a	Face_b	Handstand
# frames	100	201	384	398
input pts	3.8M	11.3M	n/a	n/a
t_{anirec}	8.7 min	41 min	n/a	n/a
# $\text{vert}_{\text{source}}$	11K	15K	74K	25K
# $\text{vert}_{\text{target}}$	53K	90K	50K	10K
t_{reg}	2.1 min	2.4 min	4.1 min	2.9 min
t_{retarget}	0.2 min	0.6 min	1 min	1.8 min
$t_{\text{transplant}}$	6 min	14 min	235 min	45 min

Table 1: *Statistics for the data sets shown in this paper.*

The hand data set shown in Figure 4 and the face_a data set in Figure 9 were both recorded using an active stereo scanner [WLG07]. We reconstructed an animated mesh from the captured range scans using the algorithm presented in Section 2. The hand animation was transplanted onto a static hand data set and the face animation was used to animate two different face scans. The second column of Table 1 contains data set statistics and timings for transplanting the facial animation onto the face shown in the third row of Figure 9.

The first row of Figure 10 shows a facial animation which was captured and reconstructed by Li Zhang using the algorithms of [ZSCS04]. Thus, no timings for the animation reconstruction step are available. The remaining rows show the results of mapping the animation onto various face scans.

To demostrate the ability of the proposed methods to transplant more complex animations, we transferred a whole body animation reconstructed by Gall et al. [GSdA*09] onto a second human character (cf. Figure 8 and the accompanying video).

Limitations: We can currently only transfer animations to models which are in a pose which is included in the animation. If we were to transfer the handstand animation to a sitting person we would have to bring the person into an upright position first. When transferring facial animations, it is crucial that the expressions of the reference animation frame and the target model are almost identical as can be seen in Figure 9: The face in the third row reproduces the recorded expressions very well. However, while still producing a reasonable animation, the face in the fourth row does not exactly reflect the emotions of the input sequence as the expression which was used for registration did not match the expression of the input data. Similar problems can be observed in Figure 10 and in the video, where a source animation with open eyes was transplanted onto target meshes which had their eyes closed.

6. Conclusion and Future Work

We have presented a novel marker-less framework for reconstructing and reusing facial animations which have been

Figure 9: *Results of our animation reconstruction and transplantation pipeline: The top row shows some input frames from a sequence of 201 scans, the second row shows the deforming mesh that was computed using our animation reconstruction method. The last 2 rows show the animation mapped onto two different face data sets.*

recorded by real-time 3D scanners. The proposed algorithm for animation reconstruction produces high quality animations that include small-scale details like expression wrinkles. Transferring the entire geometry of the animation instead of transferring only the motion of several key-points results in more convincing animations.

One of the main benefits of our method for animation transplantation is that it requires minimal user interaction as only a coarse rigid alignment of the source animation and the target model has to be specified. All our algorithms are purely based on the recorded geometry and no texture infomation is required at any stage of our pipeline. Hence, our methods are applicable for scanners which produce 3D information only, e.g. time-of-flight sensors.

In future work, we plan to address the requirement that the target pose must be contained in the input animation to achieve convincing results. For whole body animations, the problem could be solved by using an algorithm for registering articulated shapes to bring the target model into a common reference pose. We think that for facial animations, learning a neutral expression from the input animation and recording the target meshes in a neutral pose could be a first step to overcome this restriction.

Acknowledgements

We would like to express our thanks to Thibaut Weise, Li Zhang, Carsten Stoll, Daniel Vlasic and Pierre Alliez and Bob Sumner for providing the data sets used in this paper.

References

[BBA*07] BICKEL B., BOTSCH M., ANGST R., MATUSIK W., OTADUY M., PFISTER H., GROSS M.: Multi-scale capture of facial geometry and motion. *ACM Trans. Graph. 26*, 3 (2007), Paper 33. 2

[BLB*08] BICKEL B., LANG M., BOTSCH M., OTADUY M. A., GROSS M.: Pose-space animation and transfer of facial details. In *Proc. SCA '09* (2008), pp. 57–66. 2

[BM92] BESL P. J., MCKAY N. D.: A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2 (1992), 239–256. 2

[CM92] CHEN Y., MEDIONI G.: Object modelling by registration of multiple range images. *Image and Vision Computing 10*, 3 (1992), 145–155. 2

[CZ08] CHANG W., ZWICKER M.: Automatic registration for articulated shapes. *Computer Graphics Forum (Proc. SGP '08) 27*, 5 (2008), 1459–1468. 2

[CZ09] CHANG W., ZWICKER M.: Range scan registration using reduced deformable models. *Computer Graphics Forum (Proc. EG '09) 28*, 2 (2009), 447–456. 2

[GSdA*09] GALL J., STOLL C., DE AGUIAR E., THEOBALT C., ROSENHAHN B., SEIDEL H.-P.: Motion capture using joint skeleton tracking and surface estimation. In *Proc. CVPR '09* (2009), pp. 1–8. 7

[HAWG08] HUANG Q.-X., ADAMS B., WICKE M., GUIBAS L. J.: Non-rigid registration under isometric deformations. *Computer Graphics Forum (Proc. SGP '08) 27*, 5 (2008), 1449–1457. 2

[LAGP09] LI H., ADAMS B., GUIBAS L. J., PAULY M.: Robust single-view geometry and motion reconstruction. *ACM Trans. Graph. 28*, 5 (2009), to appear. 2, 5

Figure 10: *Results of our animation transplantation technique: The first row shows the input animation consisting of 384 frames which has been cloned onto the face models in the lower 3 rows.*

[LSP08] LI H., SUMNER R. W., PAULY M.: Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum (Proc. SGP '08) 27*, 5 (2008), 1421–1430. 2, 3, 6

[MFO*07] MITRA N. J., FLORY S., OVSJANIKOV M., GELFAND N., GUIBAS L., POTTMANN H.: Dynamic geometry registration. In *Proc. SGP '07* (2007), Eurographics Association, pp. 173–182. 2

[MJC*08] MA W.-C., JONES A., CHIANG J.-Y., HAWKINS T., FREDERIKSEN S., PEERS P., VUKOVIC M., OUHYOUNG M., DEBEVEC P.: Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph. 27*, 5 (2008), 1–10. 2

[NN01] NOH J.-Y., NEUMANN U.: Expression cloning. In *Proc. SIGGRAPH '01* (2001), pp. 277–288. 2, 6

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Proc. SGP '07* (2007), pp. 109–116. 3, 4, 7

[Shi04] SHINYA M.: Unifying measured point sequences of deforming objects. In *Proc. 3DPVT '04* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 904–911. 2

[SMG] SÜSSMUTH J., MEYER Q., GREINER G.: Surface reconstruction based on hierarchical floating radial basis functions. *Computer Graphics Forum 29*, X. DOI=10.1111/j.1467-8659.2010.01653.x 3

[SP04] SUMNER R. W., POPOVIĆ J.: Deformation transfer for triangle meshes. In *Proc. SIGGRAPH '04* (2004), pp. 399–405. 2, 6

[SSP07] SUMNER R. W., SCHMID J., PAULY M.: Embedded deformation for shape manipulation. *ACM Trans. Graph. 26*, 3 (2007), Paper 80. 2, 5, 7

[SWG08] SÜSSMUTH J., WINTER M., GREINER G.: Reconstructing animated meshes from time-varying point clouds. *Computer Graphics Forum (Proc. SGP '08) 27*, 5 (2008), 1469–1476. 2, 4

[Tol03] TOLEDO S.: *Taucs: A library of sparse linear solvers, version 2.2*. http://www.tau.ac.il/stoledo/taucs, 2003. 4

[WAO*09] WAND M., ADAMS B., OVSJANIKOV M., BERNER A., BOKELOH M., JENKE P., GUIBAS L., SEIDEL H.-P., SCHILLING A.: Efficient reconstruction of nonrigid shape and motion from real-time 3d scanner data. *ACM Trans. Graph. 28*, 2 (2009), 1–15. 2

[WGQ08] WANG S., GU X., QIN H.: Automatic non-rigid registration of 3d dynamic data for facial expression synthesis and transfer. In *Proc. CVPR'08* (June 2008), pp. 1–8. 2

[WJH*07] WAND M., JENKE P., HUANG Q., BOKELOH M., GUIBAS L., SCHILLING A.: Reconstruction of deforming geometry from time-varying point clouds. In *Proc. SGP '07* (2007), pp. 49–58. 2

[WLG07] WEISE T., LEIBE B., GOOL L. V.: Fast 3d scanning with automatic motion compensation. In *Proc. CVPR '07* (2007), pp. 1–8. 2, 3, 7

[WLGP09] WEISE T., LI H., GOOL L. V., PAULY M.: Face/off: Live facial puppetry. In *Proc. SCA '09* (2009), pp. XX–XX. 2

[ZH06] ZHANG S., HUANG P.: High-resolution, real-time three-dimensional shape measurement. *Optical Engineering 45*, 12 (2006), 123601. 2

[ZSCS04] ZHANG L., SNAVELY N., CURLESS B., SEITZ S. M.: Spacetime faces: high resolution capture for modeling and animation. In *Proc. SIGGRAPH '04* (2004), pp. 548–558. 2, 3, 7