# Reconstruction of Deforming Geometry from Time-Varying Point Clouds

Michael Wand[1,3] Philipp Jenke[2] Qixing Huang[1,3]
Martin Bokeloh[2] Leonidas Guibas[3] Andreas Schilling[2]

[1]Max Planck Center for Visual Computing and Communication
[2]WSI/GRIS, University of Tübingen
[3]Stanford University

## Abstract

In this paper, we describe a system for the reconstruction of deforming geometry from a time sequence of unstructured, noisy point clouds, as produced by recent real-time range scanning devices. Our technique reconstructs both the geometry and dense correspondences over time. Using the correspondences, holes due to occlusion are filled in from other frames. Our reconstruction technique is based on a statistical framework: The reconstruction should both match the measured data points and maximize prior probability densities that prefer smoothness, rigid deformation and smooth movements over time. The optimization procedure consists of an inner loop that optimizes the 4D shape using continuous numerical optimization and an outer loop that infers the discrete 4D topology of the data set using an iterative model assembly algorithm. We apply the technique to a variety of data sets, demonstrating that the new approach is capable of robustly retrieving animated models with correspondences from data sets suffering from significant noise, outliers and acquisition holes.

**Categories and Subject Descriptors (according to ACM CCS):** I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism – Animation, I.3.3 Picture/Image Generation – Digitizing and scanning

## 1. Introduction

Modeling of realistic three dimensional objects is one of the fundamental problems in computer graphics. Despite many advances in interactive modeling, the creation of high quality geometric models is still a very time consuming task that requires substantial artistic and technical skills. Consequently, a lot of research in recent years has focused on 3D shape acquisition by measuring real-world objects, allowing the creation of models of those objects with drastically reduced manual effort. Recently, several approaches have been proposed to extend such techniques to capture animated scenes in real-time. Common approaches are based on active stereo or

structured light techniques [RHL02, ZSC*04, DNR*05, FB05], passive multi-view stereo [CTM*03, ZKU*04, WLS*03], or time-of-flight measurement systems (such as [PMD, CSEM]). Animation scanning devices open up a large variety of interesting new applications, such as creating special effects for movies or content creation for interactive applications and games. However, currently available technology imposes some significant restrictions: Due to the real-time capturing requirements, dynamic acquisition techniques suffer especially badly from noise problems. In addition to that, any optical acquisition technique is limited by occlusions so that it is usually not possible to capture hole-free models. Typical data sets have large unknown regions and these holes move over the acquired object over time. Currently, there is no obvious way to solve these problems by hardware improvements in scanner technology. Consequently, the reconstruction of noise-free and hole-free animated models is a necessary prerequisite for being able to make use of these exciting new acquisition techniques. A second, very important problem is the lack of correspondences: The scanning device only outputs a series of 3D measurements without keeping track of the movement of the physical object. However, many processing and editing techniques (such as changing geometry or texture in multiple frames simultaneously) demand correspondences over time. Therefore, an automated mechanism for establishing dense and stable correspondences is also required. As an additional benefit, the correspondence estimates will also provide additional information to improve the reconstruction quality: Geometry from adjacent frames can be used to fill in holes and to remove noise more reliably, being based on more data.

In this paper, we consider the problem of reconstructing an animated model with dense correspondences from real-time range scanner data. We assume that the surface measurement is given by a sequence of point clouds sampled at a series of discrete time steps as this is the output of almost any range scanning device. We also assume that the point cloud is distorted by measurement noise and outliers. In addition, we expect the measurement to be incomplete, with large holes moving over the surface. Our reconstruction technique is based on a Bayesian statistical model, extending ideas from [JWB*06, DTB06]. We represent surfaces as a graph of oriented particles or "*surfels*" [STT93] that move over time, thus implicitly defining correspondences. Position and orientation of the surfels are controlled by statistical potentials that trade off fitting the data and prior assumptions on surface smoothness. To reconstruct the latent (not directly measured) correspondences between surfels over time, we add priors that penalize non-rigid movements and acceleration. As an additional prior, we assume the existence of a latent rest state of the shape that can be deformed optimally in this sense into all other frames and compute such a shape based on the rigidity priors. The resulting "urshape" serves as a template model and as such improves the reconstruction quality substantially. It is computed automatically from the data. In contrast to previous work, no prior knowledge of the class of objects being considered is needed; the "urshape" is not an input to but an output of our algorithm. We show how to divide the statistical maximum a posteriori reconstruction of the deforming geometry into two subproblems: The first is the discrete problem of computing the 4D connectivity structure of the data set, which describes the spatial proximity of surface points and their correspondence over time. The second subproblem is the continuous problem of computing the most likely geometry given the 4D connectivity. We derive a non-linear least-squares formulation for the second, continuous problem that can be solved efficiently using a Gauss-Newton-based optimization

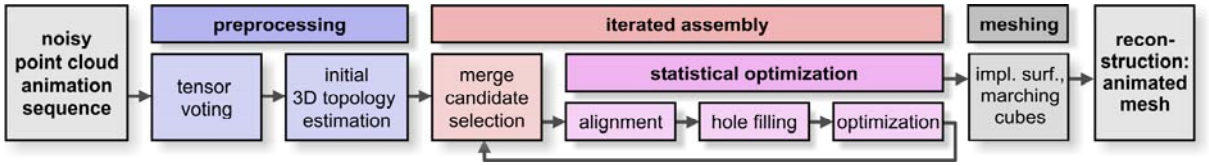



**Figure 1:** An overview of the animation reconstruction pipeline: A preprocessing step extracts 3D pieces of geometry in each frame. Adjacent frames are then iteratively merged using a statistical model to align pieces and optimize their shape as well as fill-in holes. Finally an animated triangle mesh is created by a marching cubes based surface extraction algorithm.

procedure. For the first problem of finding the discrete connectivity structure, we use a heuristic iterative model assembly algorithm that repeatedly aligns pieces of the animation sequence in order to form larger chunks of reconstructed geometry with dense correspondences. This is done using the continuous optimization as a subroutine. We evaluate the proposed method by applying it to a range of synthetic and real-world data sets, acquired with different acquisition devices, showing that the reconstruction algorithm is able to robustly reconstruct animated models from real-world data sources.

The key contribution of this paper is a complete and practical animation reconstruction pipeline that reconstructs topology, shape and correspondences as well as a deformable template model from unstructured point clouds. To our knowledge, this has not yet been done in similarly general form.

## 2. Related Work

Our technique combines previous work in 3D surface reconstruction from point clouds and deformation modeling. We also discuss the relation to existing techniques that reconstruct animation sequences from certain types of data.

**3D Reconstruction:** A large number of techniques have been proposed to reconstruct surfaces from point clouds; a full survey is beyond the scope of this paper. Our technique is an extension of the statistical point-based reconstruction method proposed by [JWB*06], which in turn is based on [STT93]. In contrast to these previous techniques, we incrementally construct a graph of surface points that explicitly describes the topology of the point set based on observations in several frames rather than using simple distance criteria, which do not yield stable results under strong noise artifacts. A similar statistical model has been proposed by Diebel et al. [DTB06], requiring however knowledge of a mesh topology as part of the input to the algorithm. For the final meshing stage in our algorithm, we use a variant of the moving least squares (MLS) technique of Shen et al. [SOS04]. We also adopt the approach of Lu et al. [LZJ*05] of preprocessing point clouds with tensor voting [MLT00] to identify well-behaved regions and remove outliers.

**Deformation Modeling:** As part of the statistical priors, we need to quantify deformation induced by reconstructed correspondences. From a differential geometry perspective, deformations of corresponding surfaces can be characterized by deviations in their first and second fundamental forms [TPB*87]. Alexa et al. [ACL00] introduce a deformation gradient model to describe least-

deforming shape interpolation. Allen et al. [ACP02] compute correspondences between different range scans of a person by fitting a skeleton controlling a displaced template surface to the data. In follow-up work [ACP03], the authors use smoothness of local affine deformations to fit a template surface to range scans of different subjects. Sumner and Popovi [SP04] use a similar approach to transfer deformations between meshes. All these techniques need a topologically equivalent and geometrically similar template mesh as input. Related deformation models are also used by Hähnel et al. [HTB03] in a non-rigid variant of ICP and Pauly et al. [PMG*05] to complete scans from pieces of similar objects. None of the aforementioned techniques considers the problem of reconstructing a multi-frame animation sequence. Our deformation model is a variant of Sumner and Popovi's technique [SP04]. We add additional orthonormality constraints as we are dealing with one and the same object deforming over time. Recently, several multi-grid methods have been proposed to improve the performance of deformation models [HSL06, SYB*06]. Our method currently does not use a multi-resolution representation but we would expect performance improvements from such an approach in future work.

**Animation Reconstruction:** Up to now, only few approaches to reconstructing animations with correspondences in general settings have been published. A common strategy is to fit template meshes to the data [MGR00, CTM03, ZSC*04]. Anuar and Guskov [AG04] use hierarchical optical flow on adaptively sampled distance fields to propagate a template model over multiple frames. Using templates typically yields good results but requires the user to provide a suitable model. Our approach assembles a suitable template model as part of the reconstruction process, thus allowing for general input data sets, making only low low-level prior assumptions (spatio-temporal smoothness rather than knowledge of the class of shapes). Sand et al. [SMP03] reconstruct animations of humans using a combination of motion capture and silhouette-based reconstruction. Their method yields impressive results but is restricted to this specific acquisition setup as well as prior knowledge of the model skeleton. Park and Hodgins [PH06] reconstruct high quality animated meshes from dense motion capture data with several hundred markers. The main difference to our approach is that their sampling (given by the markers) needs to be fixed over time. In addition, the processed marker point clouds are of low complexity, low noise and high temporal resolution, which is not the case for general scanner data. Anguelov et al. [ASP*04] consider the problem of automatically matching geometry to (complete) template models. They model the problem as a Markov random field that tries to match local geometry descriptors while preserving geodesic distances. In follow up work [ASK*05], the algorithm is employed to learn deformation and shape models of humans in different poses. Shinya [Shi04] reconstruct animations using an energy function based on deformation and data matching. By triangulating the first frame, an initial mesh is obtained that is subsequently tracked over time to perform the reconstruction. This restricts the method to inputs where complete geometry is obvious in the first frame. The influence of noise is not modeled. The optimization is based on gradient descent, which causes numerical issues in terms of performance and stability [BW98].

## 3. The Reconstruction Pipeline

Our reconstruction pipeline consists of four major components (Figure 1): Preprocessing, iterative model assembly, (continuous) statistical optimization, and triangle mesh construction. For clarity,

we will describe the statistical model and the corresponding continuous optimization procedure first (Section 3.1). The continuous optimization assumes that the 4D topology of the data set (spatial connectivity and correspondences) are already known. After that, we discuss the outer optimization pipeline: the iterative assembly procedure that actually determines the full 4D topology (Section 3.2). The assembly process makes use of different variants of the continuous optimization procedure to align and globally optimize shape and correspondence, as well as to fill in holes. Finally, Section 3.3 explains how a globally consistent mesh is obtained from the point-based representation that will be employed within the rest of the pipeline.

## 3.1. Statistical Model and Continuous Optimization

We start the discussion of the statistical model by introducing some notation. The input to our system is a set of unstructured point clouds $d^{(t)}$ sampled at discrete time intervals $t = t_0, \ldots, t_N$. We refer to the individual data points as $\mathbf{d}_i^{(t)}$, $i = 1 \ldots \#d^{(t)}$. The complete data set is referred to as $D$, its reconstruction is denoted by $S$. We employ a Bayesian approach to surface reconstruction [DTB06, JWB*06]: Given a data set $D$, we compute the posterior probability $\Pr(S \mid D)$ for a candidate reconstruction $S$ as:

$$\Pr(S \mid D) \sim \Pr(D \mid S) \Pr(S) \tag{1}$$

The likelihood term $\Pr(D \mid S)$ models how well the data is explained by the candidate reconstruction and the prior $\Pr(S)$ quantifies how likely the reconstruction itself is a priori, not considering the data. This last term is crucial; it describes additional domain knowledge about the object being reconstructed. Without any such prior, no reconstruction (other than removing measurement bias) is possible. The goal of the reconstruction is to find the original animated scene $S$ that has the largest posterior probability. To simplify computations, the optimization is done in log space, leading to an objective function

$$-\log \Pr(S \mid D) \sim -\log \Pr(D \mid S) - \log \Pr(S). \tag{2}$$

For 3D surface reconstruction, the priors typically encode some form of smoothness assumption about the original surface. For animation reconstruction, we add additional priors that couple shape and correspondences over time: First, we assume spatio-temporal smoothness, i.e. the reconstruction is more likely if the surface deformation over time is small. Second, we assume temporal smoothness, which means that surface pieces should form smooth trajectories over time. Overall, we obtain an objective function (negative log-posterior) of the following form:

$$E(S) = \underbrace{E_{match}(D, S)}_{likelihood} + \underbrace{E_{smooth}(S) + E_{rigid}(S) + E_{accel}(S)}_{priors} \tag{3}$$

In the following, we will define all terms in the objective function in the logarithmic domain. They can be converted to probability densities by taking the exponential of the negative value.

**Discretization:** To simplify the formulation, we define the probabilistic model directly in the discretized domain. Following [STT93, JWB*06], we choose a set of oriented particles (surfels) $\mathbf{s}_i^{(t)}$, $i = 1 \ldots n_s$, $t = 1 \ldots n_t$ as discretization. Particles with the same index $i$ are always in correspondence,

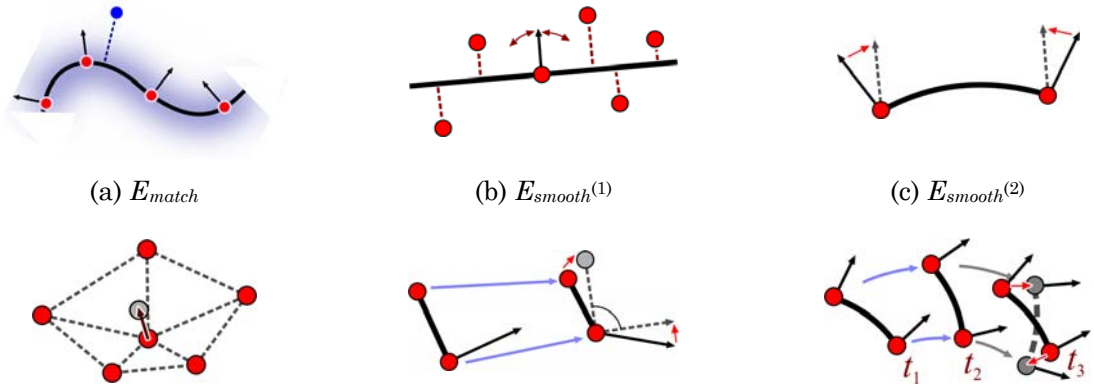
i.e. they form a trajectory over time $t$ that stays on the same physical piece of surface (i.e., the number of particles is the same for every frame). In case of holes with no data, particles become latent parameters that are estimated from the priors. For theoretical soundness, we think of truncating all prior densities to zero outside a large bounding box containing the scene to make them integrable [JWB*06] (the truncation is just a theoretical requirement, not affecting the actual implementation). A resampling procedure in the outer optimization loop (Section 3.2) will assure that the surfels (roughly) retain a sampling distance of $\varepsilon_{sampl}$. This distance is a user chosen constant that determines the maximum resolution of the reconstruction. The trajectories are connected by a graph $T_S$ that describes their topology: An edge $e_{i,j}$ is contained in $T_S$ if the geodesic distance between the trajectories is smaller than a user defined constant $\varepsilon_{top}$. We typically set $\varepsilon_{top} = c \cdot \varepsilon_{sampl}$ with $c = 2$. The topology is global over time, which means that every reconstructed frame has the same topology. We refer to a set of surfel trajectories with connectivity graph $T_S$ as 4D topology. Please note that, as we use it in this paper, the term topology refers to the local connectivity structure over time and space, not to global properties such as the genus. We denote the set of surfels that are topological neighbors of a surfel $s$ by $N_T(s)$. Using this representation, we are able to handle topological pseudo changes, such as opening the mouth (where the topology actually does not change, but it seems like it does). However, the model cannot handle inputs like the surface of a splashing liquid. An alternative representation of topology would be a triangle mesh. We do not use this type of discretization because the outer model assembly loop will perform frequent changes to the topology. Maintaining a consistent triangle mesh in these operations is much more complex both in terms of computation time and implementation complexity than just maintaining a graph of surfels (in contrast to a full mesh, the surfel graph does not need to be conforming and may contain intersecting edges). We therefore postpone the creation of a consistent mesh to the last stage of the pipeline. Given the discretization, we can now define the different terms in the negative log-likelihood function:

**Likelihood (Data Attraction):** The likelihood term $E_{match}(D,S)$ models the negative log-probability that data $D$ has originated from the reconstructed surfaces $S$. We assume that all data points have been created independently from each other, according to a noise probability density $noise^{(t)}(\mathbf{x}, \mathbf{y})$, $\mathbf{x} \in S$, $\mathbf{y} \in \mathbb{R}^3$. Assuming uniform sampling probability (i.e., having no prior knowledge about how densely different portions have been sampled) and denoting the reconstructed surface at time $t$ by $S^{(t)}$, the probability density $p(D \mid S)$ is then given by the product of all

$$p(\mathbf{d}_i \mid S) = \prod_{t=1}^{n_t} \frac{1}{\left\| S^{(t)} \right\|} \int_{S^{(t)}} noise^{(t)}(\mathbf{s}, \mathbf{d}_i^{(t)}) d\mathbf{s}. \tag{4}$$

This means that the surface is blurred by the noise distribution. In this paper, we assume an unbiased Gaussian distribution and employ the standard approximation [BM92] of just using the closest distance to the surface instead of calculating the integral expression (4). The negative log-likelihood is then given by the squared distance function [PH03], scaled according to the variance of the Gaussian noise model in normal direction

$$E_{match}(D,S) = \sum_{t=1}^{n_t} \sum_{i=1}^{\#d^{(t)}} \omega_i^{(t)} dist(S^{(t)}, \mathbf{d}_i^{(t)})^2, \tag{5}$$




(a) $E_{match}$ (b) $E_{smooth}^{(1)}$ (c) $E_{smooth}^{(2)}$

(d) $E_{Laplace}$ (e) $E_{rigid}^{(1,2)}$ (f) $E_{accel}^{(1,2)}$

**Figure 2:** Surfel potentials – red: surfels, blue: data points, grey: optimal position, light blue: correspondences

where $\omega_i^{(t)}$ is proportional to the inverse noise variance in normal direction. An issue in Gaussian noise models is outliers: Isolated points far away from the surface significantly alter the solution. In order to deal with this problem, we truncate the gradients to zero (and thus the Hessian as well) for points that are too far away. This approximates a mixture of a Gaussian and a uniform distribution [BFS04], the latter accounting for outliers. We use different strategies for different stages of the optimization: For the global statistical optimization of already aligned surfaces, we truncate at a small distance of $\varepsilon_{top}$. However, for the heuristic assembly algorithm that initially aligns surfaces, we need to tolerate much larger distances in order to handle situations with substantial motion. In this case, we determine the cutoff distance automatically as the 85% percentile of point distances and additionally reject data points with an angle of more than 45° towards the current surface normal estimate. Additionally, we check for a doublet constraint [PMG*05]: if the nearest data point of the nearest surface point a data point is not close to the data point itself, we prune this point as well.

**Spatial Smoothness (Noise Removal):** In order to evaluate the smoothness of the surface, we assign each surfel $\mathbf{s}_i^{(t)}$ a normal vector $\mathbf{n}(\mathbf{s}_i^{(t)})$. The normals are latent variables; they are not measured directly but only inferred due to the priors. The objective function (negative log likelihood) prefers normals and point positions so that neighboring points are located close to the plane (Eq. 6). In addition, neighboring normals should be similar (Eq. 7):

$$E_{smooth}^{(1)}(S) = w_{smooth}^{(1)} \sum_{t=1}^{n_t} \sum_{i=1}^{n_s} \sum_{j \in N(\mathbf{s}_i^{(t)})} \left\langle \mathbf{s}_i^{(t)} - \mathbf{s}_j^{(t)}, \mathbf{n}(\mathbf{s}_i^{(t)}) \right\rangle^2 \tag{6}$$

$$E_{smooth}^{(2)}(S) = w_{smooth}^{(2)} \sum_{t=1}^{n_t} \sum_{i=1}^{n_s} \sum_{j \in N(\mathbf{s}_i^{(t)})} [\mathbf{n}(\mathbf{s}_i^{(t)}) - \mathbf{n}(\mathbf{s}_j^{(t)})]^2 \tag{7}$$

The overall objective function is the sum of these two functions. In addition to the smoothness terms, we also employ a Laplacian potential

$$E_{Laplace}(S) = w_{Laplace} \sum_{t=1}^{n_t} \sum_{i=1}^{n_s} \left( \mathbf{s}_i^{(t)} - \frac{1}{\#N(\mathbf{s}_i^{(t)})} \sum_{j \in N(\mathbf{s}_i^{(t)})} \mathbf{s}_j^{(t)} \right)^2 \quad (8)$$

which attracts surfels to the centroid of their neighbors. This leads to a uniform distribution of surfels on the sampled surface. As this is only an additional regularization, we use only a small weight, typically 10% of the position weight. Our smoothness model is mostly identical to that of [STT93]. We currently do not use the feature preserving heavy tail potentials of Diebel et al. [DTB06]. This is not that critical for our application because we typically need to perform only very little smoothing as much of the noise is removed due to the averaging effect of the common urshape. So far, we have only employed priors on the 3D shape, neglecting the behavior of the surface over time. In the following, we will formulate 4D priors that take into account the special properties of a deforming surface over time. For these priors, we consider the correspondences between frames and treat those as latent, non-observed variables. The special correspondence-based structure of these priors is the main difference to just applying a multi-dimensional reconstruction technique to the four dimensional set of points over time and space. We use two different priors: A prior on the deformation and a prior prescribing temporal smoothness, which are described subsequently.

**Spatio-Temporal Smoothness (Rigidity):** For a deforming surface, we expect it to deform as little as possible, unless evidence (measured data) shows otherwise. Consequently, we formulate priors that try to keep the object as rigid as possible. The key component of our model is to assign a local transformation to each particle [SP04]. This local transformation is a latent variable, being reconstructed indirectly using priors. The transformation is modeled as a local rigid transformation. We use $\mathbf{A}_i^{(t)}$ to denote the corresponding, orthonormal $3 \times 3$ rotation matrix. We then employ the following objective function:

$$E_{rigid}^{(1)} = w_{rigid}^{(1)} \sum_{t=1}^{n_t} \sum_{i=1}^{n_s} \sum_{j \in N(\mathbf{s}_i^{(t)})}^{n_t} \left[ \mathbf{A}_i^t \left( \mathbf{s}_i^{(t)} - \mathbf{s}_j^{(t)} \right) - \left( \mathbf{s}_i^{(t+1)} - \mathbf{s}_j^{(t+1)} \right) \right]^2 \quad (9)$$

This potential tries to make the deformation of points and normals in a local spatial neighborhood agree with a local rigid transformation. We apply the same potential function to the normals as well, which is a valid condition because the $\mathbf{A}_i^{(t)}$ are orthonormal. In principle, the point condition alone is sufficient; having an additional constrain on the normals just improves coupling of the latent variables. Please note that the affine component of the transformation is modeled implicitly by the position of $\mathbf{s}_i^{(t+1)}$ [SP04]. This keeps the origin of the rotation in the center of each surfel and thus avoids coordinate system dependencies that arise when specifying affine mappings [ACP03]. The matrices $\mathbf{A}_i^{(t)}$ can also be interpreted as gradients of the deformation function that forcibly have been made rigid. Nevertheless, non-rigid mappings are possible as the constraints are only preserved in a least squares sense. Unlike [SP04], we do not explicitly penalize local deviations of the transformations. This is expressed implicitly in our model as the neighborhoods overlap spatially: In order to yield the same prediction for surfels joining an edge in the graph, their local transformations need to be similar.

**Temporal Smoothness (Acceleration Prior):** The last prior we employ is conservation of momentum: as any different behavior needs additional force, we expect a priori that points keep their trajectory over time.

$$E_{accel} = \sum_{t=2}^{n_t-1} \sum_{i=1}^{n_s} w_{accel} \left[ \mathbf{s}_i^{t-1} - 2\mathbf{s}_i^t + \mathbf{s}_i^{t+1} \right]^2 \tag{10}$$

Again, we apply the same penalty function (scaled accordingly) to the normals as well. The acceleration priors enforce spatial coherence and turn out to be highly effective in avoiding high frequency tangential deformation noise in the final reconstruction.

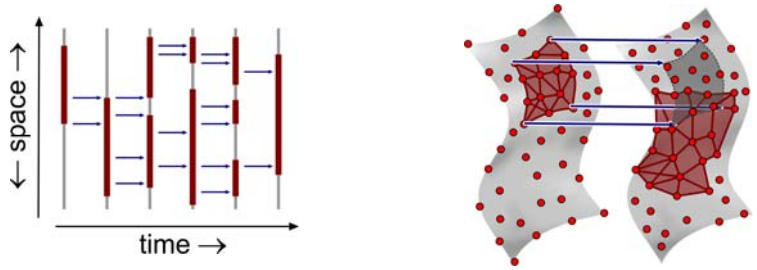
**Numerical Optimization:** The sum of the log probabilities of the described probability densities yields a non-linear least-squares problem, which we solve using numerical optimization. Several problems have to be dealt with: First, we need a reasonable starting value as the log likelihood might have multiple local extrema. This problem is addressed in the next subsection, which describes the iterative assembly algorithm. A second issue is parameterization: While the surfel position is unconstraint, the optimization of normals and rotation matrices is a constraint optimization problem. In order to avoid the difficulties of dealing with constraint non-linear optimization, we employ the parameterization approach of [HP04]: We describe changes to the normal by offset vectors in the tangent plane spanned by vectors $\mathbf{t_u}$, $\mathbf{t_v}$,

$$\mathbf{n}(u,v) = \mathbf{n} + u\mathbf{t_u} + v\mathbf{t_v}\,, \tag{11}$$

and solve for the parameters $u$, $v$ during the optimization. Then we recompute the normals and the local parameterization for the next step of the non-linear optimization. Using this parameterization, the normals might only grow, which enlarges the objective function, thus avoiding degenerate solutions (as for unconstrained normals). Initial normals are computed using PCA and region growing to unify their orientation [HDD*92]. Similarly, we need an unconstraint representation for rotation matrices: We parameterize the $\mathbf{A}^{(t)}_i$ by 3 dimensional rotation vectors $\mathbf{c}_i^{(t)}$: Their orientation defines the rotation axis and their length the angle of rotation. We can compute the rotation matrix as matrix exponent of the skew symmetric matrix $\mathbf{C}_{\mathbf{x}i}^{(t)}$ which describes the linear operation of taking the cross product with the vector $\mathbf{c}_i^{(t)}$:

$$\mathbf{A}_i^{(t)} = \exp(\mathbf{C}_{\mathbf{x}}{}_i^{(t)}) \doteq I + \mathbf{C}_{\mathbf{x}}{}_i^{(t)}. \tag{12}$$

During the optimization, we use the first order Taylor expansion (12) of the matrix exponential to parameterize the transformations in the local neighborhood of the current estimate. As in the case of tangential normal parameterizations, a valid orthonormal rotation matrix and a new local parameterization are computed after each step. Initial estimates are computed using the technique of [Hor87]. A last but important issue is the actual numerical optimization that computes the local minimum: At this point, the rigidity priors cause trouble: Defining a stiff object by making local pieces rigid yields a problem that becomes increasingly ill posed with increasing resolution of the discretization. A gradient descent strategy fails for models containing more than just a few hundred surfels: Tracking the gradient of the deformation energy yields a stiff differential equation that needs to be solved with very small timesteps to avoid explosive divergence [BW98]. For large models, often no convergence is obtained at all. We therefore employ




*(a) Merging partially known topology (schematic)*

*(b) Pieces of known 3D topology (red) are merged*

**Figure 3:** Iterative assembly to compute the 4D topology

a Newton optimizer that computes a global equilibrium of all forces in a local quadratic approximation to the non-quadratic objective function. As the objective function is of a non-linear least square type, we generally use the Gauss-Newton approximation to compute analytic approximate Hessians, which is computationally less involved. The Hessian of the squared distance function is approximated using the technique of [PH03]. We solve the linear systems in the Newton iteration using a diagonally preconditioned conjugate gradient solver. This preconditioning is important to balance the different scale of normals, rotation and positions. Using this approach, large surfel graphs can be handled without stability issues.

## 3.2. Iterative Model Assembly

The previously described optimization method needs to know the 4D topology of the data set (neighborhood graph $T_S$ and trajectory relationship) as well as a rough initialization of the variables in order to perform the optimization. In theory, we could think of trying all possible (well connected) topologies, finding all local minima of the objective function and determining the best one. However, a naïve algorithm for such an approach is exponential and it is not obvious if there exists a more efficient algorithm. We use a heuristic approximation that reconstructs the 4D topology by iteratively assembling pieces of the moving surface (**Figure 3**). This works well for real-time scans of animated scenes, which are usually densely sampled in the temporal domain. A more rigorous statistical reconstruction of 4D topology from noisy data is a very interesting, non-trivial problem that we leave for future work.

The iterative model assembly pipeline works in two steps: First, we extract *seed regions* in 3D. These are pieces of geometry for which the 3D topology can be safely determined by a local proximity heuristic. The selection of these regions is done using tensor voting [MLT00]. After that, in the second step, we iteratively merge geometry from adjacent frames. Each merging operation yields a set of trajectories over two frames with a common topology. The process is then iterated subsequently to create longer trajectories until each merged trajectory spans the complete animation and one single neighborhood graph $T_S$ describes the 3D topology of these trajectories in all frames simultaneously, which is the final reconstruction. Each merging operation itself is also performed in multiple steps: The first step is to align the first frame of the next and the last frame of the previous range of frames geometrically (**Figure 3**b, Figure 4a). Then, the topology of these

two frames is stitched together. As the topology graph $T_S$ of each set of frames is global over time, connecting the frames connects the topology in all frames. The new set of trajectories covers the entire range of frames (Figure 4b). As a result, all trajectories from the left part are uninitialized in the right part and vice versa (Figure 4b, grey points). Thus, the next task is to fill in the interpolated estimates of the shape in these regions. We initialize the new points with a simple prediction and then locally perform statistical optimization. Finally, the resulting surfel graph (Figure 4c) is resampled to a fixed sampling density (in order to keep the complexity of the representation constant) and the optimization is applied once more to all surfels simultaneously to distribute errors globally, which finalizes the merging of two ranges of frames. Filling in missing values with estimates and optimizing these estimates also happens in regions where no data is present, i.e. in acquisition holes. In this way, holes are automatically filled with estimates from neighboring frames Thus, in the acquisition holes, the fill-in will resemble the neighboring geometry with as little deformation as possible. In the following, we will first describe the preprocessing procedure that estimates the initially recoverable seed regions in 3D. Then we describe the steps of the assembly algorithm one after the other.

**Initial Estimation of Recoverable 3D Topology:** The first step is a preprocessing step that extracts areas where the 3D topology $T_S$ of the data is obvious from a single frame. Later, these potentially disconnected pieces will be stitched together to assemble the global topology, assuming that it will become obvious over several frames. This step might still make mistakes: In some situations, such as an opening mouth, the topology cannot be estimated reliably from some of the frames. These problems will be dealt with in later stages of the algorithm. The main idea for the initial stage is that locally smooth areas are likely to represent a simple, topologically disc-like piece of the original model. In order to extract smooth, possibly curved pieces of surfaces, we employ tensor voting, which is a powerful feature extraction technique [MLT00]. It represents surface points as ellipsoids for which the main axis refers to the normal direction and non-zero values for the other two axes represent uncertainty. Our implementation closely follows the surface voting procedure described in [MLT00]: First, "cross-product" voting tensors are accumulated at every data point; then the resulting tensor field is convolved with a surface voting tensor to enhance the contrast. As a result, every data point (we accumulate votes at data points only) obtains a normal direction tensor. We then threshold their aspect ratio (typically: second smallest / largest eigenvalue ≤ 0.8) to identify points in locally smooth regions. These points get connected by a neighborhood graph that forms edges between all points within distance $\varepsilon_{top}$. Points with uncertain normal directions are excluded from the topology computation. As a useful byproduct, this procedure also removes isolated outliers [LZJ*05]. For the resulting surface patches, we apply the 3D part (data matching and smoothness) of the statistical optimization to remove noise from the initial estimate. Given these initial pieces of surface, we can now start merging frames in the process of which the individual topological pieces will be stitched together. Each merging step involves geometric alignment, topology stitching, hole filling, and global optimization, which we will discuss subsequently:

**Geometric Alignment:** The first step in the merging process is to align two frames at adjacent times such that corresponding pieces of the surface come to rest at the same spatial positions. This alignment will then be used subsequently to form topological connections. In the following, we will call the two ranges of frames the "*left*" and the "*right*" piece, the left ending at

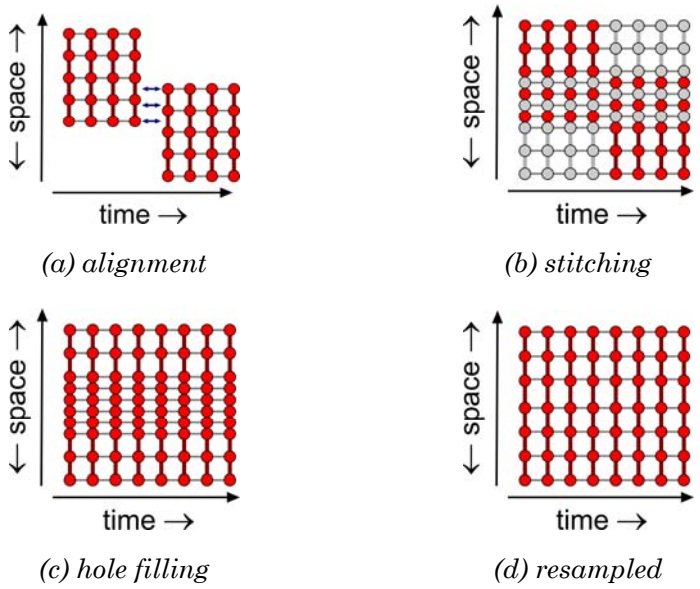



**Figure 4:** Steps of the topological merging pipeline. (a) Two ranges of frames are aligned. (b) The topology is stitched together. (c) Unknown surfels (grey) are filled in. (d) The surfel graph is resampled and optimized.

time $t_{left}$ and the right starting at $t_{right} = t_{left} + 1$ (Figure 4a). We employ the statistical optimization procedure to obtain a variant of non-rigid ICP [HTB03]: We form an auxiliary scene with two frames: The first frame is constant and set to $S^{(t_{left})}$. The second frame contains the unknown alignment to be optimized; the variables are also initialized with $S^{(t_{left})}$ as starting position. The data points in the second frame are set to $S^{(t_{right})}$ in order to attract the aligned frame to the configuration in the following frame. We then perform continuous optimization on the second frame using data matching and rigidity potentials only, until convergence. If the frames of the animation are densely sampled over time, the likelihood term will attract the predicted points to the actual data points, while the priors retain a smooth deformation. In order to increase the robustness of the pairwise alignment process, we perform the alignment twice, once aligning the left with the right frame and once the other way round. We compute the resulting average potential in both alignments and choose the variant with the smaller stretch. After stitching the topology (see below), we align the other part using the established correspondences; this third alignment step converges quickly, typically within 2-3 iterations.

**Topological Stitching:** Having aligned the two frames, we have to stitch together their topology. In order to do so, we start from scratch and just connect every surfel to all of its neighbors within distance $\varepsilon_{top}$ in the aligned configuration. Doing this naïvely could create a wrong topology: Surfels that have been connected previously will still be connected after the merge as they keep their distance due to the rigidity potential. However, surfels that come close to each other for only a few frames might be falsely connected. To detect such situations, we employ an additional filter: Whenever the distance between surfels changes over the newly created trajectory by more than a certain factor (typically 1.5), the connection is rejected. In this way, the topology combines proximity information in all corresponding frames.

**Hole Filling:** After topological stitching, some parts of the trajectories are still unknown. These are the right parts from the left frame's trajectories and left parts of right frame's trajectories (Figure 4b). The frames directly involved in the alignment have already been filled in with aligned data but the surfels in all other frames are uninitialized so far. This step fills acquisition holes as well as extrapolated trajectories supported by data points; the procedure is the same: We first initialize surfels in frames directly adjacent to already initialized frames with a copy of that data. Then, we use again the statistical optimization procedure to compute in the most likely values at these points. We restrict the optimization to run only at the newly estimated surfels, treating all other surfels, including spatial and temporal neighbors, as constant boundary conditions. This ensures both quick convergence (due to many known neighbors) and limited computational effort. The procedure is iterated by a region growing in time until all frames have been filled up. Please note that we do not fill surface area unless it is visible in at least one frame. For geometry that has been unobserved so far, hole filling is done when merging with the first frame that contains data in this area.

**Resampling:** The next step is to resample the surfel graph in order to keep the discretization density (and thus the computational costs) constant. This is done by greedily deleting all surfels that are "unnecessary", i.e. which are within distance $\varepsilon_{sampl}/2$ of another point over all frames and are not the last surfel to support another previously deleted point. This strategy creates a provably good approximation of an optimal surfel distribution on the surface [Wan04]. To our experience, upsampling is not necessary as the iterated merging constantly adds new surfels anyway. The resampled topology is created by simply again recomputing the topology using the previously described strategy.

**Global Optimization and the Urshape:** Next, we perform a global optimization of the complete merged range of frames on all surfels, again minimizing the previously derived statistical energy function. This avoids error accumulation by balancing all terms in the statistical model over time and space. Global optimization is done in two steps: first, only the trajectories are optimized. In the second step, we compute the urshape: We form an empty frame, not containing any data points and initialize it with any of the reconstructed frames (in our implementation just the first frame). Then we connect this frame to all other frames of the animation. This means, the rigidity potential is evaluated for any pair of a surfel in a frame and a surfel in the urshape. Then we run the optimization procedure (experiments show that a joint optimization of both geometry and urshape at the same time shows a good convergence behavior; the presence of data points in all frames avoids artifacts due to the initialization of the urshape). The rigidity potentials between frames are not used at this stage. Empirically, splitting up the optimization into one trajectory and a subsequent urshape fitting step led to the best results on our test data sets. The addition of the urshape fitting step proved to be especially helpful to avoid correspondence noise and drift in the final reconstruction. In addition, it yields an "average" template model that can be deformed most easily into all other frames, which is a useful output of its own. Global optimization finalizes the merging of two frame ranges.

**Iteration:** The whole merging procedure (alignment, hole filling, stitching, resampling and global optimization) is iterated in a binary scheme until all frames of the animation are merged: We iteratively merge adjacent pairs frame ranges, leading to a logarithmic number of merging steps per frame involved. Urshapes are currently recomputed at each level from scratch. After all

frames have been merged, we run an additional global optimization step. In this stage, we use increased rigidity weights to finally smooth the reconstruction. The reason for this extra step is that strong rigidity penalties during merging make the problem of detecting topological changes harder to solve. To avoid oscillations during merging, we also postpone the usage of strong acceleration penalties to this last optimization step.

### 3.3. Meshing

The result of the reconstruction is a graph of trajectories sampled with oriented surfels. From this representation, we create the final animated mesh. To convert a single frame into a triangle mesh, we construct an implicit function for each surfel according to the local linear model defined by its surface normal. Then we blend between these local implicit functions [SOS04] using a Gaussian weighting kernel with standard deviation proportional to $\varepsilon_{sampl}$. From this representation, we extract a triangle mesh using a standard marching cubes algorithm [HDD*92], augmented with a border detection step that clips regions not supported by surfels [JWB*06]. Each triangle vertex is associated with the nearby surfels according to their Gaussian window weight, which we renormalize to form a partition of unity. Therefore, we can compute a meshing of the whole animation by just copying the mesh topology for all frames and recompute the vertex positions, normals and possible additional attributes according to the Gaussian weights.

## 4. Implementation and Results

We have implemented the proposed system and applied it to a number of synthetic and real-world data sets. The input data sets and the reconstructions are show in detail in the video accompanying this paper. Figure 5 and Table 1 give a rough overview of the results. We render the final meshes textured, with texture coordinates of the first frame to show the correspondences. We also add a specular environment map to visualize surface smoothness.

**Synthetic tests:** Our first synthetic test data set is the well known Venus torso model, rotating about 120° over 20 frames. We subsequently add uniform Gaussian noise and 10% random outliers within the bounding box of the scene (see video for a comparison). The reconstructed correspondences do not show any visible drift. Correspondences are stable under noise and outliers, only some high frequency details in the geometry are lost as more smoothing becomes necessary to control the noise level. A second synthetic data set shows a model of an elephant with bending legs and proboscis that has been modeled using a commercial 3D modeling package. Again, we add noise and outliers and additionally cut out several large holes over the course of the animation. Again, we obtain globally stable correspondences over the entire sequence. Holes are seamlessly filled in with geometry from other frames. Some artifacts are visible in the final mesh at the elephant's ears as the sampling of the surfel graph was chosen too coarse for the final marching cubes step.

| | venus | elephant | face [DNR*05] | face [GK07] | hand gesture | popcorn tin |
|---|---|---|---|---|---|---|
| *frames / surfels / data pts.* | 20 / 25,905 / 399,920 | 20 / 49,500 / 963,671 | 21 / 63,651 / 1,333,000 | 20 / 32,740 / 400,000 | 26 / 21,294 / 520,000 | 15 / 56,985 / 896,301 |
| *preprocessing* | 272 sec | 412 sec | 1,203 sec | 419 sec(*) | 708 sec(*) | 434 sec |
| *merging* | 5,816 sec | 14,755 sec | 16,721 sec | 19,674 sec | 4,002 sec | 19,621 sec |
| *final global opt.* | – | 1,196 sec | 4,910 sec | 7,367 sec | 1,398 sec | 1,320 sec |

**Table 1:** Computation time on a single core Pentium-4 3.4GHz with 2GB of main memory ((*) Pentium-4 3.0Ghz)

**Real-world data sets:** Next, we apply the method to a set of real-world data sets obtained with different types of animation scanning devices. The first data set is a human face with opening mouth acquired using space-time stereo [DNR*05]. The data set is incomplete and even over all frames some amount of hole area remains unobserved. A special challenge is the topological pseudo change of the opening mouth. After adjusting rigidity weights and maximum edge length change tolerance manually, our algorithm is able to compute the correct topology. Correspondences are tracked reliably, without global drift. A small amount of tangential noise remains, which is mostly damped out by increasing acceleration penalty weights. In the final meshing, some marching cubes artifacts remain at the boundaries. In addition, some small patches created from structured outliers remain in the reconstruction; we currently do not remove small non- or loosely connected pieces automatically. We obtain comparable results for a facial animation captured with a prototype structured light scanner of Gumhold and König [GK07]. Again, the topology of the data set is correctly reconstructed as far as this is apparent from the incomplete input data. Area that is visible in some of the scans is filled in reliably with roughly believable dynamics (due to the deformation and acceleration priors). As a simple application example, we use the computed correspondences to propagate color painted on the first frame to all other frames automatically (see video) using the same technique that extrapolates the position and normal attributes from the single frame marching cube reconstruction. The third real-world example is a gesturing hand, acquired using a real-time structured light scanner [ABW]. The main problem is once more to reconstruct the topological pseudo-change when the fingers meet. This is achieved correctly up to sampling resolution. Correspondences are again tracked reliably with no global drift and little tangential noise. The video shows how the computed correspondences can be used to paint simultaneously on all frames. A last test data set shows a person shaking a popcorn bowl acquired with a color coded real-time structured light scanner [FB05]. Here, the algorithm merges the tin with parts of the hand because the input sequence is ambiguous: it does not show examples with sufficient separation of the two. Otherwise, we again obtain smooth surfaces and visually correct correspondences and some artifacts in boundary regions or due to surface-like structured outliers.

## 5. Discussion and Future Work

We have presented a system for performing animation reconstruction from time series of noisy and incomplete point clouds. The method uses a statistically motivated optimization procedure that removes noise, establishes correspondences over multiple frames and fills in holes from other frames. Using this approach, we were able to reconstruct animated meshes with dense correspondences from data corrupted by strong noise and outlier artifacts as well as large portions

of missing data. For synthetic data sets that meet our model assumptions, we obtain almost artifact free results and globally and locally stable correspondences. Stable correspondences are also obtained for real-world data sets; however, some geometric artifacts remain due to structured outliers (which are not modeled) and in some boundary regions. The latter is a shortcoming of our simple meshing technique that could be improved in future work by adding better handling of boundary curves [JWB*06]. As far as data is available, holes are filled in reliably and the algorithm is able to distinguish between acquisition holes and topological changes based on the deformation of adjacent geometry. Our approach has several limitations that could be addressed in future work: The employed iterative assembly heuristic does not provide guarantees for finding a good solution. For data sets with bad spatial or temporal sampling, the assembly algorithm cannot determine the 4D topology correctly. We have conducted some successful preliminary experiments of using geometric feature matching to improve correspondence estimation for large temporal spacing. However, a general solution to determine a strong statistical estimate of the 4D topology still remains an open problem. A further, important practical limitation is the large computational costs of the current method. A scene representation that separates low resolution, adaptively sampled, time variant deformation and high resolution, static geometry could improve upon this. We would also like to integrate the reconstruction more tightly into a computer vision system, using more accurate noise distributions and visibility constraints. Using the urshape as common reference frame, it would also be interesting to try to learn the weights of the prior potentials from data by trading off fitting residuals and additional regularization constraints that avoid degenerate zero weights. Having a practical and stable technique to reconstruct shape and correspondences from dynamic 3D scanner data could serve as a basis for more general geometry processing algorithms on animated data. The long term goal of such efforts could be an application such as a scanner based 3D movie editing pipeline.

## Acknowledgements

The authors acknowledge the support of NSF grants ITR 0205671 and CARGO 0310661, NIH grant GM-072970, DARPA grant 32905, and the Max-Planck Center for Visual Computing and Communication. We would also like to thank James Davis, Stefan Gumhold, Christian Theobald, Simon Pabst, Phil Phong and Oliver Schall for providing data sets and Natasha Gelfand and Steve Oudot for fruitful discussions.

## References

[ABW] ABW GMBH: *http://www.abw-3d.de*

[ACL00] ALEXA, M., COHEN-OR, D., LEVIN, D.: As-rigid-as-possible shape interpolation. In: *Proc. SIGGRAPH 2000*.

[ACP02] ALLEN, B., CURLESS, B., POPOVIĆ, Z.: Articulated Body Deformation from Range Scan Data. In: *ACM Trans. on Graphics 21(3),* 2002.

[ACP03] ALLEN, B., CURLESS, B., POPOVIĆ, Z.. The space of all body shapes: reconstruction and parameterization from range scans. In: *ACM Trans. on Graphics 22(3)*, 587–594, 2003.

[AG04] ANUAR, N., GUSKOV, I.: Extracting Animated Meshes with Adaptive Motion Estimation. In: Proc. *Vision, Modeling, and Visualization 2004*.

[ASK*05] ANGUELOV, D., SRINIVASAN, KOLLER, D., THRUN, S., RODGERS, J., DAVIS, J.: SCAPE: Shape Completion and Animation of People. In: *ACM Trans. on Graphics 24(3)*, 408–416, 2005.

[ASP*04] ANGUELOV, D., SRINIVASAN, P., PANG, H.C., KOLLER, D., THRUN, S., DAVIS, J.: The Correlated Correspondence Algorithm for Unsupervised Registration of Nonrigid Surfaces. In: *NIPS* 2004.

[BFS04] BIBER, P., FLECK, S. STRASSER, W.: A Probabilistic Framework for Robust and Accurate Matching of Point Clouds. In: *Proc. 26th Pattern Recognition Symposium,* 2004.

[BM92] BESL, P., MCKAY, N.: A method for registration of 3d shapes. In: *IEEE PAMI 14(2),* 1992.

[BW98] BARAFF, D., WITKIN, A.: Large steps in cloth simulation. In: *Proc. SIGGRAPH 98,* pp. 43–54, 1998.

[CTM*03] CARRANZA, J., THEOBALT, C., MAGNOR, M., SEIDEL, H.P.: Free-Viewpoint Video of Human Actors. In: *ACM Trans. on Graphics 22(3),* 569–577, 2003

[CSEM] CSEM SA: *http://www.swissranger.ch*.

[DNR*05] DAVIS, J., NEHAB, D., RAMAMOOTHI, R., RUSINKIEWICZ, S.: Spacetime Stereo: A Unifying Framework for Depth from Triangulation. In: *IEEE PAMI, 27(2)*, 2005.

[DTB06] DIEBEL, J.R., THRUN, S., BRÜNIG, M.: A Bayesian method for probable surface reconstruction and decimation. In: *ACM Trans. on Graphics 25(1)*, 39–59, 2006.

[FB05] FONG, P., BURON., F.: High-Resolution Three-Dimensional Sensing of Fast Deforming Objects. In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.

[GK07] GUMHOLD, S., KÖNIG, S.: *personal communication*, 2007.

[HDD*92] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLE W.: Surface Reconstruction from Unorganized Points. In: *Computer Graphics, 26(2)*, 71–78, 1992.

[Hor87] HORN, B.. Closed-form solution of absolute orientation using unit quaternions. In: *J. Opt. Society of America A 4*, 1987.

[HP04] HOFER, M., POTTMANN, H.: Energy-minimizing splines in manifolds. In: *ACM Trans. on Graphics 23(3)*, 284–293, 2004.

[HSL*06] HUANG, J., SHI, X., LIU, X., ZHOU, K., WEI, L.-Y., TENG, S.-H., BAO, H., GUO, B., SHUM, H.-Y.: Subspace gradient domain mesh deformation. In: *ACM Trans. on Graphics 25(3)*, 2006.

[HTB03] HÄHNEL, D., THRUN, S., BURGARD, W.: An Extension of the ICP Algorithm for Modeling Nonrigid Objects with Mobile Robots. In: *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI) 2003*.

[JWB*06] JENKE, P., WAND, M., BOKELOH, M., SCHILLING, A., STRASSER, W.: Bayesian Point Cloud Reconstruction. *In: Computer Graphics Forum 25(3)*, 379–388, 2006.

[LZJ*05] LU, D., ZHAO, H., JIANG, M., ZHOU, S., ZHOU, T.: A Surface Reconstruction Method for Highly Noisy Point Clouds. In: *Proc. Variational, Geometric, and Level Set Methods in Computer Vision, LNCS, Springer,* 2005.

[MGR00] MARSCHNER, S.R., GUENTER, B., RAGHUPATHY, S.: Modeling and Rendering for Realistic Facial Animation. In: *Proc. Eurographics Workshop on Rendering 2000*.

[MLT00] MEDIONI, G. LEE, M.S., TANG C.K.: *A Computational Framework for Segmentation and Grouping*, Elsevier, 2000.

[PH03] POTTMANN, H., HOFER, M.: Geometry of the squared distance function to curves and surfaces. In: *Visualization and Mathematics III*, Springer, pp. 223–244, 2003.

[PH06] PARK, S.I., HODGINS, J.K: Capturing and Animating Skin Deformation in Human Motion. In: *ACM Transaction on Graphics 25(3)*, 881–889, 2006.

[PMD] PMD technologies: *http://www.pmdtec.com*

[PMG*05] PAULY M., MITRA N. J., GIESEN, J., GROSS, M., GUIBAS L. J.: Example-Based 3D Scan Completion. In: Proc. Eurographics Symp. on Geometry Processing, pp. 23–32, 2005.

[RHL02] RUSINKIEWICZ, S., HALL-HOLT, O., LEVOY, M.: Real-Time 3D Model Acquisition. In: *ACM Trans. on Graphics, 21(3),* 2002.

[SCL*04] SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., SEIDEL, H.-P.: Laplacian surface editing. In: *Proc. Symp. on Geometry Processing 2004*, pp. 175 – 184.

[Shi04] SHINYA, M.: Unifying measured point sequences of deforming objects. In: *Proc. 3D Data Processing, Visualization and Transmission 2004*.

[SMP03] SAND, P., MCMILLAN, L., POPOVIĆ, J.: Continuous Capture of Skin Deformation. In: *ACM Trans. on Graphics 22(3)*, 2003.

[SOS04] SHEN, C., O'BRIEN, J. F., AND SHEWCHUK, J. R.: Interpolating and approximating implicit surfaces from polygon soup. In: *ACM Trans. Graph. 23(3)*, 896–904, 2004.

[SP04] SUMNER, R.W., POPOVIĆ, J.: Deformation transfer for triangle meshes. In: *ACM Trans. on Graphics 23(3)*, 399–405, 2004.

[STT93] SZELISKI, R., TONNESEN, D., TERZOPOULOS, D.,: Modeling surfaces of arbitrary topology with dynamic particles. In: *Proc. IEEE CVPR,* pp. 82–87, 1993.

[SYB*06] SHI, L., YU, Y., BELL, N., FENG, W.-W.: A fast multigrid algorithm for mesh deformation. In: *ACM Trans. on Graphics, 25(3)*, 1108–1117, 2006.

[TPB*87] TERZOPOULOS, D., PLATT, J., BARR, A., FLEISCHER, K.: Elastically deformable models. In: *Computer Graphics 21(4)*, 1987.

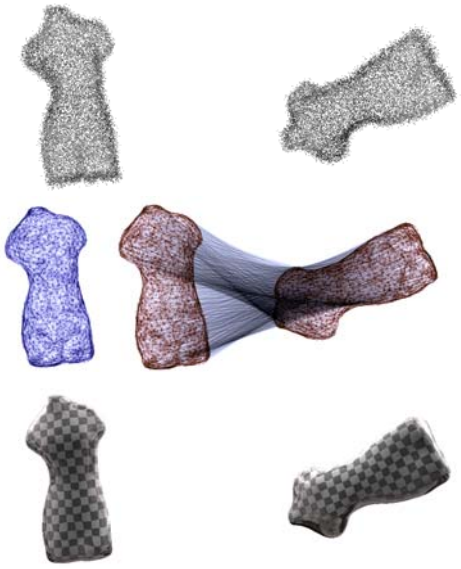
[Wan04] WAND, M.: *Point-based Multi-Resolution Rendering*. PhD thesis, University of Tübingen, 2004.

[WLS*03] WÜRMLIN, S., LAMBORAY, E., STAADT, O., GROSS, M.: 3D Video Recorder: A System for Recording and Playing Free-Viewpoint Video. In: *Computer Graphics Forum, 22(2)*, 2003.

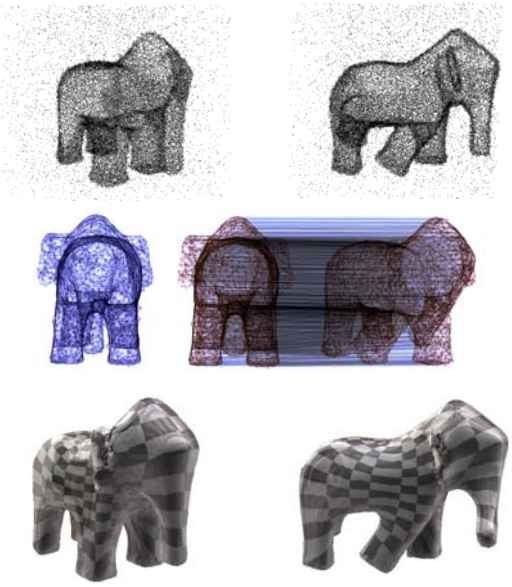
[ZSC*04] ZHANG, L., SNAVELY, N., CURLESS, B., SEITZ., S.M.: Spacetime Faces: High-resolution capture for modeling and animation. In: *ACM Trans. on Graphics23(3)*, 548 – 558, 2004.

[ZHS*05] ZHOU, K., HUANG, J., SNYDER, J., LIU, X., BAO, H., GUO, B., SHUM, H.-Y.: Large mesh deformation using the volumetric graph Laplacian. In: *ACM Trans. on Graphics 24(3)*, 2005.

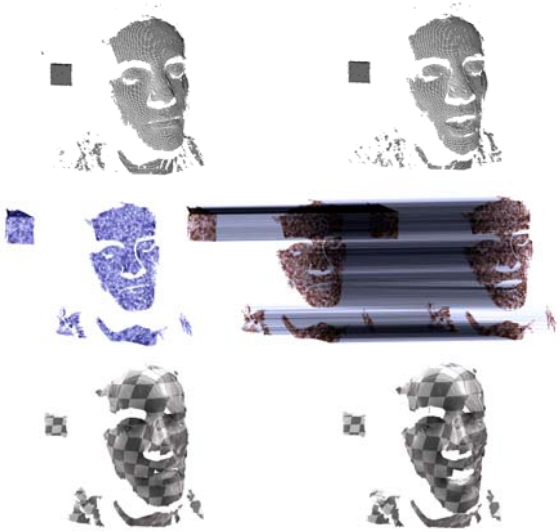
[ZKU*04] ZITNICK, C.L., KANG, S.B., UYTTENDAELE, M., WINDER, S., SZELISKI, R.: High-quality video view interpolation using a layered representation. In: *ACM Trans. on Graphics, 23(3),* 2004.



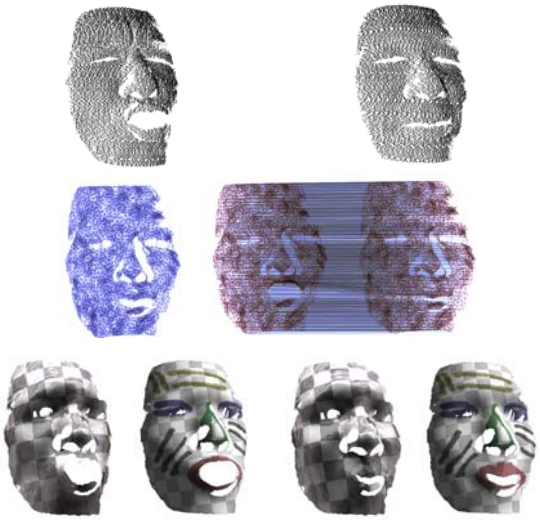
(a) venus torso (synthetic)



(b) elephant (synthetic)



(c) face I [DNR*05], courtesy of James Davis



(d) face II [GK07 ], courtesy of Stefan Gumhold

(e) hand gesture, courtesy of Oliver Schall (f) popcorn tin, courtesy of Phil Fong

**Figure 5:** Example scenes – top: input data, center: reconstructed surfel graphs (blue: urshapes), bottom: final meshes. The images show two different frames with large temporal spacing; please refer to the video for the full animations.