Eurographics Symposium on Point-Based Graphics (2006)
M. Botsch, B. Chen (Editors)

Template Deformation for Point Cloud Fitting

C. Stoll[1], Z. Karni[1], C. Rössl[2], H. Yamauchi[1], and H.-P. Seidel[1]

[1]Max-Planck Institut für Informatik, Saarbrücken, Germany
[2]INRIA Sophia-Antipolis, France

Abstract

The reconstruction of high-quality surface meshes from measured data is a vital stage in digital shape processing. We present a new approach to this problem that deforms a template surface to fit a given point cloud. Our method takes a template mesh and a point cloud as input, the latter typically shows missing parts and measurement noise. The deformation process is initially guided by user specified correspondences between template and data, then during iterative fitting new correspondences are established. This approach is based on a Laplacian setting for the template without need of any additional meshing of the data or cross-parameterization. The reconstructed surface fits to the point cloud while it inherits shape properties and topology of the template. We demonstrate the effectiveness of the approach for several point data sets from different sources.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Geometric algorithms

1. Introduction

In recent years we have observed increasing interest in spatial scanning devices and the development of new and improved technologies that enable real-time capturing of real-world objects. This proliferation is mostly due to the fact that geometry acquisition and reconstruction are essential to many fields of application: in the life-cycle of industrial product design, prototypes are digitized to serve as feedback to the designer; scanners are used along manufacture lines for quality and process control. In medicine, the shape of internal organs is captured to detect malfunctions and diseases using minimal invasive methods. In security and authentication, spatial scanning introduces an additional dimension upon the traditional image based methods. However, the most evident use of shape digitization is in the entertainment industry, where digital models in games produce realistic scenes and motion, while the movies show realistic special effects. Independent of scanning technology and application domain, most geometry acquisition results in unstructured point cloud data, where each point provides a sample of the acquired object, typically afflicted with measurement error. Precision of measurement depends on many factors, such as acquisition device and technology, environmental conditions, complexity of the scanned object and many more. Fig. 1 (b), 2 (b), 7 (b) show examples of point clouds acquired with different methods. It is clearly visible that some surface parts are missing as they could not be acquired because of technical reasons.

The process of transferring an unstructured point cloud model into a consistent discrete surface model such as a polygonal mesh is commonly referred to as surface reconstruction. Here, the main task consists of the generation of a manifold mesh that approximates the input data, i.e., that captures its global shape and topology together with its fine geometry details. However, typical data from acquisition shows missing surface regions even if multiple scans are spatially aligned and combined into a single model. Any reconstruction method must fail for this data in the sense that missing parts can be filled or extrapolated reasonably only if additional knowledge on the original shape is provided. This applies locally as well as globally: without knowledge, holes are patched smoothly (if at all), and global shape properties such as the genus cannot be detected from incomplete data.

There are various approaches how to apply additional knowledge to surface reconstruction. Our method uses a template shape for reconstruction of point clouds without any preprocessing such as noise and outliers removal. In particular no high-level tools such as parameterization or maps between surfaces are required. The only restriction is that in order to get reasonable results, the template should share the same global structure or "nature" with the data, e.g., a

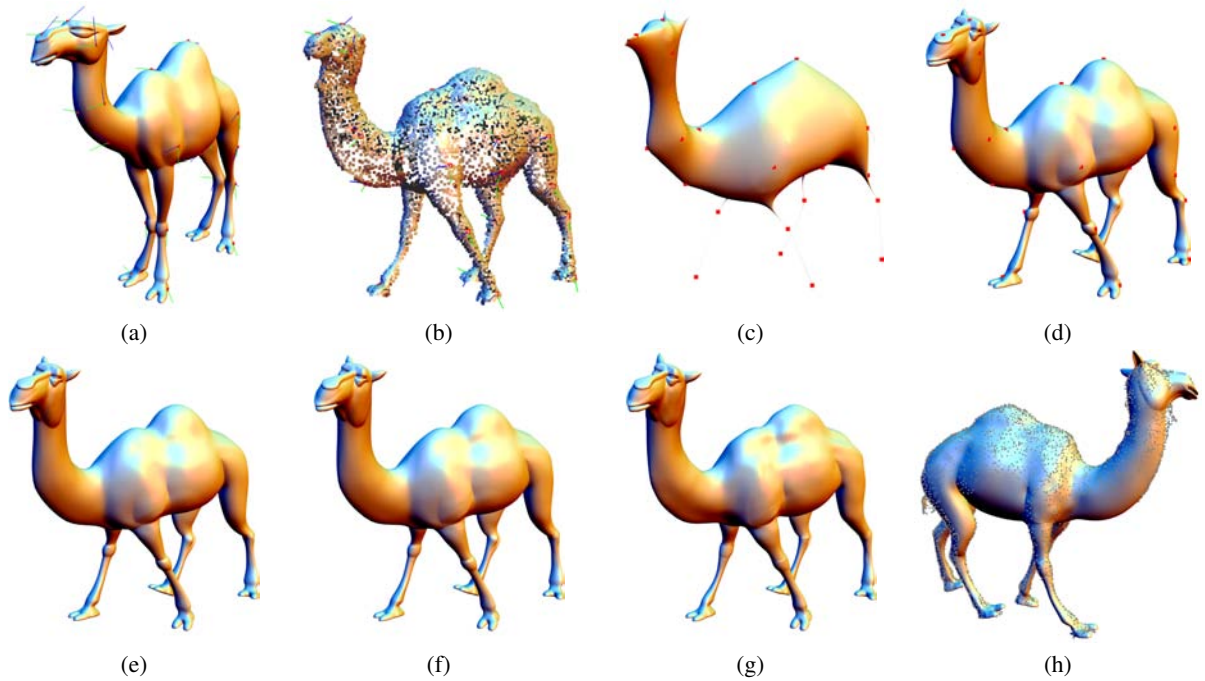


Figure 1: *Overview of the method for the camel: (a) Template with* 30 *oriented markers; (b) point cloud with oriented markers; (c) initial deformation (no prior adjustment applied on input data, which are extremely out of scale); (d) after scaling. (e)-(g) show iterations 1,2, and 4, respectively; (h) shows the final result from the backside with points overlaid. Front head, feet, and tail were masked out due to missing data.*

human model serves as template for reconstructing another human although shapes are in different poses.

Our method uses a small number of correspondence points marked interactively by the user to deform the template shape into the pose of the acquired point cloud model. Successive deformations improve approximation of the data. These deformations are guided by additional local correspondences which are established based on certain geometric criteria. Throughout the whole process, information is propagated from known to unknown parts of the shape. Our results show that with only minimal user interaction, the approach reconstructs a consistent and smooth surface that approximates the input data. If suitable templates exist, template-based methods like ours carry an advantage compared to those that work only on the plain data. We remark that in practice, templates exist for most common applications, many of them processing a certain class of shapes, e.g., humans. This leads to another advantage of our approach: If the same template is used for different data sets we trivially obtain a map between the deformed templates. Indeed, such maps are required by many applications, e.g., for texturing, and they are especially valuable for processing time-dependent data.

2. Related work

Surface reconstruction from unstructured point cloud data has been within interest of computer graphics since its early days. The further development of acquisition techniques with increasing amount of data and the quest for accuracy, correctness and robustness have rendered it a topic of active research. In the following, we cannot go into details of reconstruction methods in general, and for an overview refer to the recent work of Kazhdan [Kaz05] and the references therein. Following Kazhdan, reconstruction methods can be categorized as: Computational Geometry based methods, many of them with certain guarantees, implicit function based approaches which fit and extract iso-surfaces, and such methods which fit an explicit surface to the data. Our approach is within the latter category. Early work in this direction are active meshes [TV91], where a grid structure is deformed to fit sampled intensity and range data based on a mass-spring model. In [CM95], a dynamic physical model is applied to "inflate" a balloon-like mesh to fit scanned data from inside; the process is supported by local adaptation. An inverse shrink wrapping method [KVLS99] fits meshes to meshes. Fitting methods are particularly used in reverse engineering together with surface classification, segmentation, and feature detection; often local regions are processed separately as patches (see, e.g., [VMC97]). However, for most methods it is difficult or impossible to generate rea-

sonable results if parts of the input data are missing. On the other hand this situation is typical for most acquisition techniques. Nevertheless, consistency of models and efficiency of the reconstruction process are vital for many applications. Lévy [Lév03] fills missing parts smoothly working in the parametric domain. Sharf et al. [SAC04] generate local geometry based on similarity measures to existing parts. This leads to template-based approaches, where missing or contaminated input data is compensated by additional knowledge in form of a template shape. Template-based methods were frequently used in the previous years especially for reconstructing animated models. Kähler et al. [KHYS02] use a detailed model of face anatomy to generate facial animation; this is an example for a highly specialized model. In [ACP02, ACP03, ASK*05], a combination of templates together with learning techniques is used to reconstruct motion of a human model with a known skeleton. Most recently, Kraevoy and Sheffer [KS05] establish maps between triangulated data and template in order to transfer missing geometry. This work, together with the work by [ACP02, ACP03], is built on cross-parameterization between the template and the input model. In contrast to the above, our approach is applied to the original scanned point cloud model. It is based on deformations of the template, and it requires no additional tools such as parameterization. Recently, Pauly et al. [PMG*05] apply global deformation, similar to [ACP03], to a collection of templates. They determine the best fitting template parts by segmentation and rebuild the target model from these. Our method relies only on a single template, and uses local fitting in addition to global deformation to match the template to the target point cloud. Deformation for mesh editing has been exhaustively researched, and we refer to recent surveys by Alexa [ACS05] and Sorkine [Sor05], who focuses on the Laplacian settings used in this work. Furthermore, we remark that for non-rigid registration, warping techniques based on thin-plate splines have been used [BR04, CR03]. In contrast to our method, which deforms the surface, these approaches apply smooth deformations of the embedding space.

3. Overview of the algorithm

Our method processes general point data with normals that are either given from data acquisition or can be estimated (see, e.g., [JDZ04]). Such data is typically afflicted with noise and outliers, is incomplete with missing parts and hence not directly appropriate for meshing. Our goal is to deform a template mesh such that the input data is approximated in global pose and local surface features, i.e., missing parts are filled from the template.

In order to achieve this, the user specifies pairs of corresponding points on point cloud and template (cf. Fig. 1 (a), (b), and Sec. 5.1). In addition to positions, correspondence in local frames is established for each pair. This first step is accomplished quickly with an intuitive user interface. Corresponding pairs are typically placed near shape features. The correspondance points and local frames do not have to match perfectly on the template and point cloud. In practice a rough approximation of similar positions and directions (i.e. have the local frame point towards the foot on a constraint point on a knee) is sufficient.

The approach is based on a Laplacian setting w.r.t. the template mesh (Sec. 4). We compute Laplacian coordinates of the template and estimate local rotations from the given corresponding local frames. In order to obtain an initial deformation, local rotations are interpolated over the template and Laplacian coordinates are rotated accordingly. Then the mesh is reconstructed subject to positional constraints from the correspondences. The result mimics the global pose of the data, however, it suffers from improper scaling (cf. Fig. 1 (c) and Sec. 5.2).

In order to compensate for this, we recover a global scale from averaging ratios of discrete geodesic distances between the user specified points on the original and on the deformed template. The reconstruction from the scaled Laplacian coordinates now captures the global pose of the input data (Fig. 1 (d)). However, taking only into account the input correspondence so far, the deformed mesh still resembles the template and does not yet provide sufficient local approximation of the data points.

We address the latter issue with an iterative process of matching and reconstruction. For all vertices of the template, we search for matches on the data. This search is guided by a maximum radius and a maximum angular deviation of normals from template and data. False matches in erroneous or insufficiently sampled data regions are avoided by masking out such regions. For every match a weighted positional constraint is included in the linear system, and the deformed template mesh is reconstructed. This process is iterated based on the new deformation until the deformed template approximates the data sufficiently (cf. Fig. 1 (e)–(g) and Sec. 5.3).

In a final step, we estimate local scales for all local matches in the previous set based on the displacements from the initially deformed template and to last deformation. The lengths of displacements are interpolated over the template to generate new positional constraints for the final reconstruction. The rationale of this last step consists of propagation of scaling information into regions of the template with no matches or correspondences due to missing data. A final result is shown in Fig. 1 (h).

4. Laplacian setting and notation

Our method is based on Laplacian surface editing [Ale03, LSC*04, SLC*04]. Here, we summarize on the essential notation and differences to our approach. Let $\mathcal{T} = (\mathbf{p}, T)$ be the template mesh defined by vertex positions $\mathbf{p} = (\mathbf{p}_i)$

(a) (b) (c) (d) (e)

Figure 2: *Fitting a female model. (a) Template with* 46 *markers placed; (b) point cloud (with parts missing); (c) ignored regions stay rigid; (d) initial fit after scaling; (e) final result (see also closeup in Fig. 5 (a)).*

$(i = 1,\ldots,n)$ and a triangulation T of the vertices. The triangle mesh $\mathcal{T}$ represents a piecewise linear surface, and discretizations of the Laplacian w.r.t. $\mathcal{T}$ are well known [PP93, MDSB02], in the following we apply the discrete Laplace operator $\mathbf{L}$ based on the cotangent-weights.

Following [LSC*04], we compute Laplacian coordinates $\mathbf{d}$ of the template as $\mathbf{d} = \mathbf{L}\,\mathbf{p}$. (Here, and in the following, all coordinates are treated component-wise in x, y, z.) The reconstruction $\mathbf{x}$ of the surface from $\mathbf{d}$ subject to a certain number of constraints $\mathbf{x}_j \approx \mathbf{q}_j$, $j \in \{1,\ldots,n\}$, leads to minimizing

$$\operatorname*{argmin}_{\mathbf{x}} \left\{ \|\mathbf{L}\mathbf{x} - \mathbf{d}\|^2 + \|\mathbf{C}\mathbf{x} - \mathbf{q}\|^2 \right\},$$

i.e., to a linear least-squares problem and finally to solving

$$(\mathbf{L}^\top \mathbf{L} + \mathbf{C}^\top \mathbf{C})\mathbf{x} \;=\; \mathbf{L}^\top \mathbf{d} + \mathbf{C}^\top \mathbf{q}\,. \qquad (1)$$

Here, we assume $\mathbf{C}$ is a diagonal matrix with non-zero entries $\mathbf{C}_{j,j} = w_j$ (where w_j is the weight of the additional entry) only for constrained vertices j, and consequently only positions $\mathbf{q}_j$ are non-zero.

It is well-known that this setting as is cannot handle rotations adequately, and there are different strategies to cope with the rotation-invariance of Laplacian coordinates [SLC*04, LSLC05]. In the following, we take a different approach and use harmonic interpolation [ZRKS05] over the template shape to propagate appropriate rotations, which are then applied to the coordinates $\mathbf{d}$. For the harmonic interpolation of rotations $\mathbf{r}$ a Laplace system is solved, namely $\mathbf{L}\,\mathbf{r} = 0$ subject to Dirichlet boundary conditions (specifying rotations at the correspondence points).

We use this basic setting throughout all subsequent steps, and we can take advantage from precomputations such as matrix factorizations. We are aware that other approaches could be used in a similar way to achieve our goals like for instance [YZX*04, ZRKS05], see [Sor05] for a recent survey. We chose this setting because it shows excellent results, it is simple in concept and implementation, and it is efficient.

5. Deformation-guided matching

5.1. User interaction

Given the point data, the user chooses an appropriate template mesh that will be deformed to approximate the input. Our method is robust, and there are no restrictions on the template in general, however, no meaningful results can be expected from severe mismatches, i.e., in genus.

In the next step the user identifies and marks pairs of corresponding points on template and point data. Such correspondences are required on or near shape features, e.g., at feet and knees for human or animal models. In addition, the user defines a local frame for every marked point. With normals for point data and template given, this means that corresponding tangential directions on both surfaces are chosen. Interactive definition of correspondence is typically a matter of minutes for reasonably complex input. The quality of the final result depends on the number of correspondences and their locations. Fig. 1 (a) shows oriented markers on the camel template. Their corresponding positions are first identified by clicking on the point cloud, and within the same user action tangent vectors are rotated (using the mouse wheel) until rotation matches. Note that orientation is only required to be consistent between template and data, In particular, no explicit alignment to surface features or principal curvatures is necessary.

Correspondences will be used to globally bring the template into the "pose" of the point data. In the subsequent step, local correspondences between shapes will be searched based on heuristics. Here, it is important to remove the influence of data regions which are insufficiently sampled and

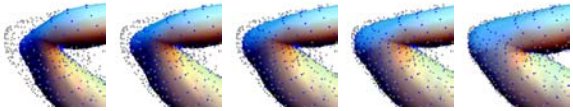


Figure 3: *Closeup of the elbow from Fig. 2 with data points overlaid. Iterative improvement, from left to right: Initial fit and iterations 1,2,4,8.*

cannot provide reasonable information. For this reason, the user selects a global region of interest either on the point data or on the template (as done in our examples). Points outside this region are ignored for any subsequent matching (Sec. 5.3), i.e., for these masked-out regions the template is deformed rigidly. The choice of this region and hence assessment of the data is straightforward even for a non-trained person. An example for this is the hand in Fig. 2, which is open in the template and closed in the point-cloud. The density of the point-cloud is not high enough to correctly fit single fingers like those present in the template.

5.2. Initial deformation and global scaling

From the selected correspondences, we compute the initial deformation of the template. Let pairs $(\mathbf{p},\mathbf{q})$ represent positions of corresponding points on the template and on the point cloud, respectively. Then we treat $\mathbf{q}$ as constraints in the Laplacian setting (Sec. 4) using uniform weights in $\mathbf{C}$. Prior to reconstruction, we estimate local rotations for every pair of corresponding points based on the two local frames. We represent rotations $\mathbf{r}_j$ as quaternions and propagate them (element-wise) over the surface by harmonic interpolation following the approach in [ZRKS05].

Interpolating rotations requires solving one Laplace system in the four components of quaternions. After normalization the result is a field of rotations, and for every vertex of the template we rotate associated Laplacian coordinates $\mathbf{d}$. The application of all rotations simultaneously can be written as linear operator $\mathbf{R}$ such that for the initial deformation of the template we solve

$$\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \begin{pmatrix} \mathbf{S}\,\mathbf{R}\,\mathbf{d} \\ \mathbf{q} \end{pmatrix}, \qquad (2)$$

where $\mathbf{A}^\top = (\mathbf{L}^\top, \mathbf{C}^\top)$ (from (1)) and the diagonal scaling matrix $\mathbf{S} = \mathbf{I}$ is the identity for now.

The result (see Fig. 1 (c)) already shows the desired pose, however, ignoring the fact that template and input data may be (and generally are) scaled differently may distort the resulting shape in an unacceptable way. In order to fix this issue, we estimate a global scaling factor λ for the template which is applied to the Laplacian coordinates $\mathbf{d}$. Therefore, we compute discrete geodesic paths (using Dijkstra's algorithm) between marked points on the original template and the initially deformed one. We obtain a global scale as average of the ratios of geodesic distances on both shapes. (Our experiments show that a global scale is sufficient, and local diversification does not yield significant improvements except for extreme configurations, which did not occur in any of our examples. On the other hand simple global measures such as ratio of bounding box diagonals are not reliable enough due to different poses.) Hence, we now solve the same system (2) again with updated right-hand-side applying $\mathbf{S} = \lambda\mathbf{I}$ for scaling. A result is shown in Fig. 1 (d). In the following we will re-apply the same system with updates in the constraints $\mathbf{C}$ and $\mathbf{q}$.

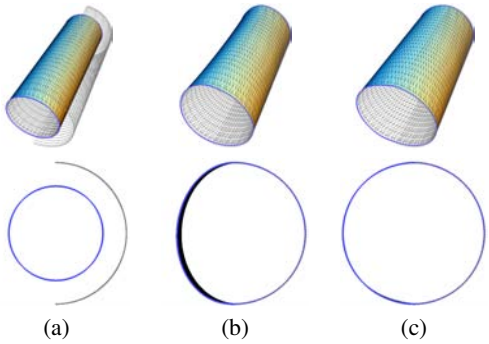
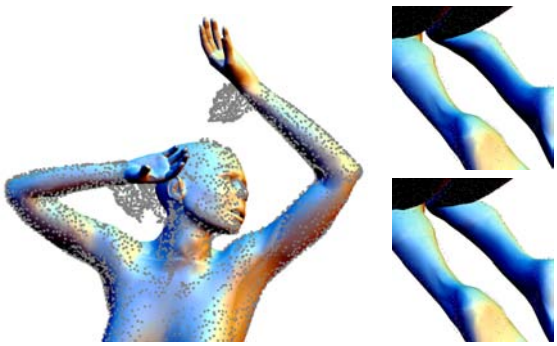


Figure 4: *The cylinder example illustrates the effect of displacement propagation for the final reconstruction. (a) Template and point cloud, there is no data for the left half; (b) result after iterations using only data points; (c) for the final result, displacements were propagated to the region with missing data.*

5.3. Iterative improvement of approximation

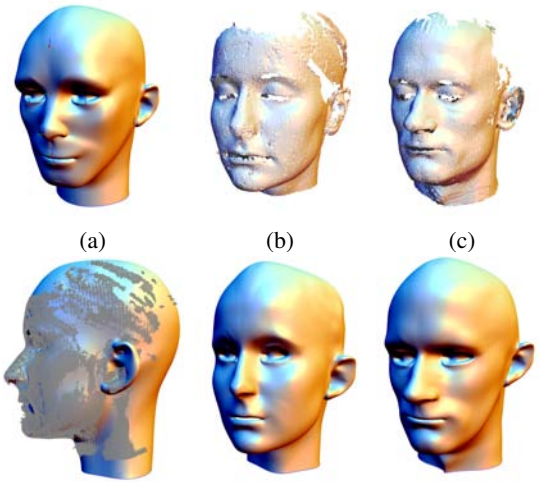
The initial deformation of the template mesh using proper scaling globally captures the pose of the point cloud. However, the deformed shape does not yet provide sufficient local approximation of the data. The following iterative process moves the template nearer towards the data points guided by local correspondences which are established from simple heuristics. This local matching is motivated by iterative closest point [BM92] (ICP) algorithms for finding (rigid) transformations for shape registration.

For every vertex of the template, we search for matches in the nearest data points within a maximum distance $r_{\max}$. Of all points found for a single vertex j those are rejected for which their normal deviates from the vertex normal by more than a maximum angle. From the remaining points we compute a positional constraint $\mathbf{q}_j$ as a weighted average of the point positions. We restrict these displacements to their contribution in direction of template normals to avoid tangential drift. The weighting is based on point-vertex distances mapped by a quadratic B-spline transfer function which maps distances 0 and $r_{\max}$ to 1 and 0, respectively, with C^1 continuity at the interval boundaries. This information is used to update $\mathbf{C}$ and $\mathbf{q}$ in (2), where we chose



(a) (b)

Figure 5: *(a) Closeup of the final result in Fig. 2 (e); point cloud overlaid for comparison. (b) Closeups to the legs to visualize the effect of displacement propagation (bottom).*



(a) (b) (c)

(d) (e) (f)

Figure 6: *Fitting of two heads. (a) Template with* 11 *markers; (b), (c) data sets; (d), (e) result for (b), the overlay of points illustrates coverage of the data; (f) result for (c).*

$\mathbf{A}_{n+j,j} = \mathbf{C}_{j,j} = w_j$ using the same weighting scheme as above but now based on distances to averaged target positions $\mathbf{q}_j$ instead of data points. The updated system is then solved again, yielding a new deformation of the template. Starting from this new configuration, we search again for local matches and iterate the process. Note that this formulation only involves changing $\mathbf{C}$ and $\mathbf{q}$, where prior constraints are either preserved or overwritten by updates. Fig. 3 illustrates the iteration.

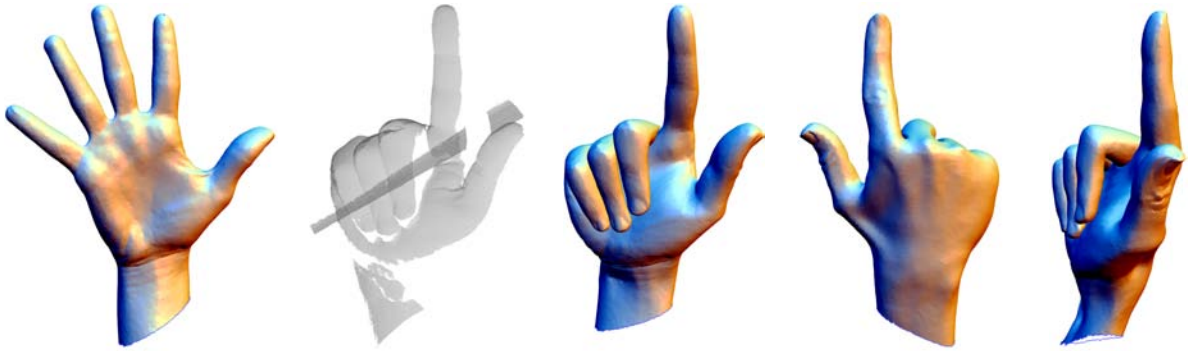
The local matching for finding new constraints must fail in regions where the point data is erroneous, e.g., due to measurement error, insufficient sampling, and missing data. This situation cannot be compensated by manually choosing correspondence and because of the lack of data no meaningful deformations can be extracted. Besides the fact that such regions may be subject to manual postprocessing, they must be excluded from local matching as they typically lead to false matches. This is done by restricting the search not only by local distance and angular thresholds but also to the globally defined region of interest within the point cloud.

After a sufficient number of iterations (4-15 for all our examples), the template mesh is deformed in a way that it approximates the point cloud in global pose and local shape (Fig. 3). Of course this refers only to shape regions where point data is available (and the region of interest, respectively). In a final step, we improve on the remaining regions of the deformed template for which no counterparts exist in the data. We identify regions with sufficient point data simply as all vertices j for which (local) constraints have been found before, i.e., $\mathbf{C}_{j,j} > 0$. For all such vertices, we measure the displacement into normal direction for the initially deformed template and the result of the last iteration, respectively. These displacements capture shape information of the point cloud, and their lengths are then propagated over the template mesh in the same way as the rotations. The interpolated values are added as additional positional constraints by displacing the points of the initial deformation in direction of their normal to the interpolated distance. These new positional constraints obtain low weights (we used $w_j = \frac{1}{2}$). We observe that this heuristic provides good results, Fig. 4 illustrates the effect for a simple example, the closeups on Fig. 5 (b) show the effect on the female model (Fig. 2).

6. Results and discussion

We have implemented the method described above in an interactive environment. Positioning a constraint point and adjusting its local frame is a single action that the user applies on template and point data, respectively. In rare cases it is necessary to place a few markers freely, away from the data; this situation occurred only for the hand example in Fig. 7, which consist of a single scan without backside. A typical fitting session takes between 5–20 minutes (in particular for all our examples). System response is immediate such that the overall timing is driven by user interaction. Computation times such as matrix factorization are negligible in this context; this fact is also known from Laplacian-based interactive shape editing systems (see, e.g., [Sor05, BBK05]). Computational cost is dominated by matrix operations and thus size of the template (rather than nearest neighbor searches on the point cloud), all automatic stages of iterations were performed in less than 30 seconds for all models.

Our method successfully addresses common problems in point cloud reconstruction: noise, outliers, fragmented scans, and missing parts. The weighted averaging of iterative deformation together with smoothness inherited from the Laplacian operator effectively acts as a low-pass filter suppress-



(a) (b) (c) (d) (e)

Figure 7: *A template hand model (a) is fitted to a single range scan (b). The final result is shown in (c)–(e) from different views.*

ing artifacts such as high-frequency noise and outliers on the point data. Our method has even proven to be robust towards poorly aligned fragments combined in a point cloud (Fig. 9).

In all our examples, we observed a significant amount of missing data for the scans, e.g., belly of the camel, back of the hand/head, and large parts of the human models (1, 2, 6, 7, 9). These parts are filled with the template shape in a natural way, transitions to such regions are hardly noticeable in the results.

We show that our method is capable of handling extreme deformations (Fig. 2, 8) to different poses and even shapes. However, depending on the quality of the input and the application it is sometimes not desirable to align each of the small details of the template to the points. For example the left hand of the original female model in Fig. 3 (a) is in a fist while the template displays an open hand. Although our method is capable of this deformation (see Fig. 7), it would require user interaction and careful placement of markers and cannot be achieved automatically. Therefore, we exclude the hand part from iterative matching and deformation. Masking out certain parts or focussing on a region of interest, enables us to preserve certain details and features of the template model, e.g., the head in Fig. 2 or most noticeable the legs in Fig. 8. For other applications, it is vital to keep parts of the template that do not match semantically to the point data such as the open and closed eyes in Fig. 6. Table 1 summarizes facts on all of our examples. The main limitation of our method is the incapability to reconstruct sharp features and small details of the point cloud. This is due to the smoothing required for noise suppression, and also a feature sensitive resampling of the point data would be required. A possible approach to this problem could consist in extending coating transfer [SLC*04] to point cloud settings.

model	figures	#marks	#verts	#pts	#err
camel	1	30	39024	6195	0.57%
female	2, 5	46	13784	11798	0.38%
head 1	6 (b)	11	16544	52954	0.12%
head 2	6 (c)	11	16544	65593	0.16%
hand	7	33	25735	114767	0.25%
woman	9	26	13784	11798	0.32%
bronto	8	41	39024	23982	0.55%
horse	8	33	48485	6195	0.33%

Table 1: *Summary on datasets used. Columns refer number of correspondence markers (# marks), vertices in the template (# verts), points in the point cloud (# pts), and relative approximation error as ratio of one-sided Hausdorff distance from points to results (ROI only) to length of bounding-box diagonal.*

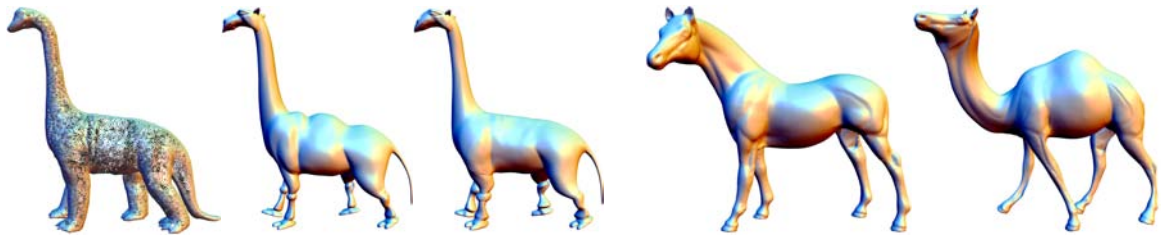
7. Summary

We present a method for surface reconstruction from point cloud data using a template mesh. Our method deforms the template using a simple Laplacian setting to capture the pose of the point cloud guided by user generated correspondences and a subsequent iterative adjustment. It is interactive and intuitive to use. Finally, in our examples we show the effectiveness of the method even for data with noise, outliers, holes, and missing parts, and for large scale deformation — all of which are common challenges in surface reconstruction from point clouds.

8. Acknowledgements

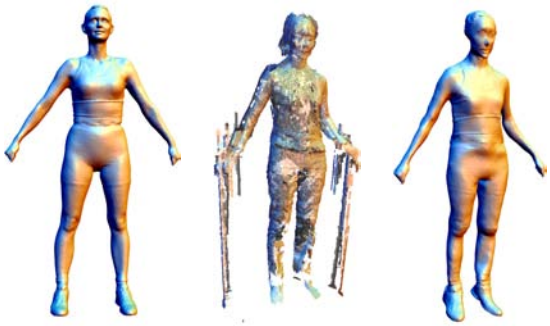
This work was supported in part by AIM@SHAPE, a Network of Excellence project (506766) within EU's Sixth Framework Programme.

Some of the models are courtesy of the AIM@SHAPE Shape Repository found at http://shapes.aim-at-shape.net/ .



(a) (b) (c) (d) (e)

Figure 8: *Left: The Brontocamelosaurus is an example of extreme deformation using the camel template (Fig. 1(a)). (a) Point cloud; (b) initial deformation; (c) final result. Lower legs, tail and ears were masked out. Right: Use of the horse (d) model as template for the camel point cloud from Fig. 1 (b), result shown in (e).*



(a) (b) (c)

Figure 9: *Reconstruction of a female model from a noisy dataset consisting of two fragments (front and back) which are poorly registered. Note missing parts and artifacts from shadowing in the input data. (a) Template; (b) point cloud, hands and supporting bars were masked out; (c) result.*

References

[ACP02] ALLEN B., CURLESS B., POPOVIĆ Z.: Articulated body deformation from range scans. *ACM Transactions on Graphics 21*, 3 (2002), 612–619. 3

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

[ACS05] ALEXA M., CANI M.-P., SINGH K.: Interactive shape modeling. In *Eurographics course notes* (2005). 3

[Ale03] ALEXA M.: Differential coordinates for local mesh morphing and deformation. *The Visual Computer 19*, 2 (2003), 105–114. 3

[ASK*05] ANGUELOV D., SRINIVASAN P., KOLLER D., THRUN S., RODGERS J., DAVIS J.: SCAPE: shape completion and animation of people. *ACM Transactions on Graphics 24*, 3 (2005), 408–416. 3

[BBK05] BOTSCH M., BOMMES D., KOBBELT L.: Efficient linear system solvers for geometry processing. In *11th IMA conference on the Mathematics of Surfaces* (2005). 6

[BM92] BESL P. J., MCKAY N. D.: A method for registration of 3-d shapes. *IEEE Trans. Pat. Anal. and Mach. Intel. 14*, 2 (1992), 239–256. 5

[BR04] BROWN B., RUSINKIEWICZ S.: Non-rigid range-scan alignment using thin-plate splines. In *Symposium on 3D Data Processing, Visualization, and Transmission* (2004). 3

[CM95] CHEN Y., MEDIONI G.: Description of complex objects from multiple range images using an inflating balloon model. *Computer Vision and Image Understanding 61*, 3 (1995), 325–334. 2

[CR03] CHUI H., RANGARAJAN A.: A new point matching algorithm for non-rigid registration. *Comput. Vis. Image Underst. 89*, 2-3 (2003), 114–141. 3

[JDZ04] JONES T. R., DURAND F., ZWICKER M.: Normal improvement for point rendering. *IEEE Computer Graphics and Applications 24*, 4 (2004), 53–56. 3

[Kaz05] KAZHDAN M. M.: Reconstruction of solid models from oriented point sets. In *Symposium on Geometry Processing* (2005), pp. 73–82. 2

[KHYS02] KÄHLER K., HABER J., YAMAUCHI H., SEIDEL H.-P.: Head shop: Generating animated head models with anatomical structure. In *ACM SIGGRAPH Symposium on Computer Animation* (2002), pp. 55–64. 3

[KS05] KRAEVOY V., SHEFFER A.: Template-based mesh completion. In *Symposium on Geometry Processing* (2005), pp. 13–22. 3

[KVLS99] KOBBELT L., VORSATZ J., LABSIK U., SEIDEL H.-P.: A shrink wrapping approach to remeshing polygonal surfaces. *Compututer Graphics Forum 18*, 3 (1999), 119–130. 2

[Lév03] LÉVY B.: Dual domain extrapolation. *ACM Transactions on Graphics 22*, 3 (2003), 364–369. 3

[LSC*04] LIPMAN Y., SORKINE O., COHEN-OR D.,

LEVIN D., RÖSSL C., SEIDEL H.-P.: Differential coordinates for interactive mesh editing. In *Shape Modeling International* (2004), pp. 181–190. 3, 4

[LSLC05] LIPMAN Y., SORKINE O., LEVIN D., COHEN-OR D.: Linear rotation-invariant coordinates for meshes. *ACM Transactions on Graphics 24*, 3 (2005), 479–487. 4

[MDSB02] MEYER M., DESBRUN M., SCHRÖDER P., BARR A.: Discrete differentialgeometry operators for triangulated 2-manifolds. In *Proc. VisMath* (2002), pp. 35–57. 4

[PMG*05] PAULY M., MITRA N. J., GIESEN J., GROSS M., GUIBAS L.: Example-based 3d scan completion. In *Symposium on Geometry Processing* (2005), pp. 23–32. 3

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experiment. Math. 2*, 1 (1993), 15–36. 4

[SAC04] SHARF A., ALEXA M., COHEN-OR D.: Context-based surface completion. *ACM Transactions on Graphics 23*, 3 (2004), 878–887. 3

[SLC*04] SORKINE O., LIPMAN Y., COHEN-OR D., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *Symposium on Geometry Processing* (2004), pp. 179–188. 3, 4, 7

[Sor05] SORKINE O.: Laplacian mesh processing. In *Eurographics STAR* (2005), pp. 53–70. 3, 4, 6

[TV91] TERZOPOULOS D., VASILESCU M.: Sampling and reconstruction with adaptive meshes. In *IEEE Computer Vision and Pattern Recognition* (1991), pp. 70–75. 2

[VMC97] VARADY T., MARTIN R. R., COX J.: Reverse engineering of geometric-models: An introduction. *Computer-Aided Design 29*, 4 (1997), 255–268. 2

[YZX*04] YU Y., ZHOU K., XU D., SHI X., BAO H., GUO B., SHUM H.-Y.: Mesh editing with Poisson-based gradient field manipulation. *ACM Transactions on Graphics 23*, 3 (2004), 644–651. 4

[ZRKS05] ZAYER R., RÖSSL C., KARNI Z., SEIDEL H.-P.: Harmonic guidance for surface deformation. In *Proc. Eurographics* (2005), pp. 601–609. 4, 5