Geodesics Guided Constrained Texture Deformation

Carsten Stoll Zachi Karni Hans-Peter Seidel

Max-Plank-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken
Germany

[*stoll* | *karni* | *hpseidel*]*@mpi-inf.mpg.de*

Abstract

We present a method that deforms an image plane to visually meet the shape and pose of a manifold surface. The user provides constraints that couple a small number of surface points with their corresponding image pixels to initially deform the plane. Matching, based on geodesic distances, couples additional points, followed by a second deformation that brings the image plane into its final pose and shape. The method works on any type of surface that supports geodesic distances evaluation. This includes not-triangulated and high genus models with arbitrary topology. The result is a smooth, visually pleasing and realistic textured surface that can be superimposed onto or used instead of the original model and with some limitations can be considered as a parameterization or remeshing method for the area of interest.

1. Introduction

Texture mapping is a common technique in computer graphics that wraps a two-dimensional image around a polygonal mesh model to add details and enhance the visual appearance of the model. Constrained texture mapping is applied when alignments are required between the model and the image. A known application that uses constraints is facial texture mapping that requires the alignment of face-features such as eyes, nose, mouth, etc.

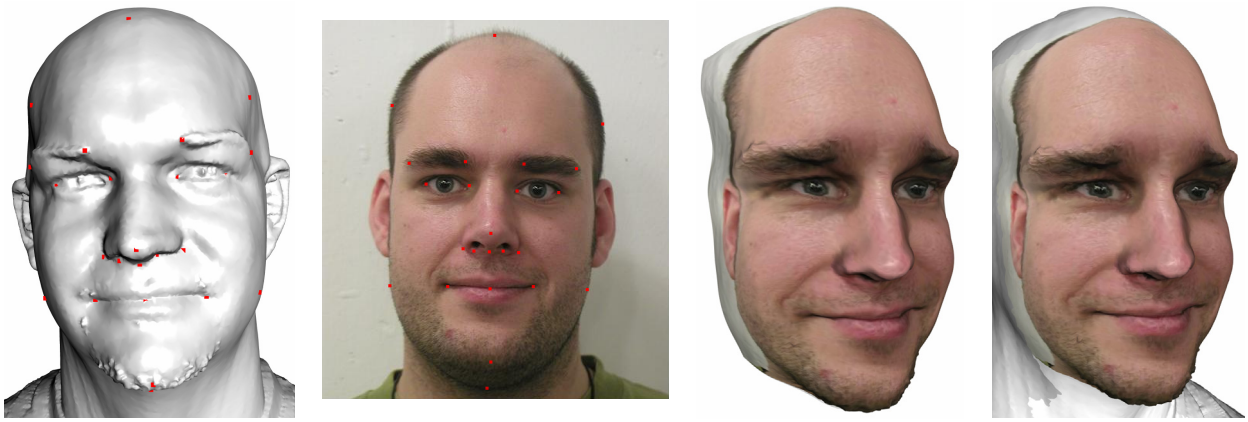
In computer graphics the term *mapping* or *parameterization* refers to the process of establishing a bijective (one-to-one and onto) correspondence between the 3D model and a 2D domain. Texture mapping is considered as a *parameterization* problem in which the 3D position of the model's vertices are defined as a bivariant piecewise-affine function and the two independent variables are used as *texture-coordinates*. In constrained texture mapping the texture-coordinates for part of the vertices are enforced and the result is a bivariant function that complies with the constraints and parameterizes the remaining vertices. This makes the parameterization problem much more difficult and might even render the problem unfeasible.

Texture mapping improves the visual appearance of a polygonal model. A highly detailed image gives the illusion of a detailed object, although the underlying geometric structure of polygons and vertices may be coarse. Hoverer, when closely examined, the rough geometric structure becomes noticeable and together with image distortion artifacts, caused by the parameterization process, the model may look less realistic.

2. Previous work

Mesh parameterization and texture mapping received significant attention from the scientific community for several years. Rather than list them all, we point the interested reader to the excellent survey by Floater and Hormann [3] and references therein for more details on this topic. We note that common to most methods is their goal of minimizing a distortion measure while guaranteeing a bijective mapping (a mapping that is one-to-one and onto).

Most parameterization methods focus on meshes with a topology homeomorphic to a disc and are limited to triangulated surfaces. To parameterize non-disc-like meshes, it is common to first partition (segment) the mesh into disc-like patches, parameterize each separately and pack them back together in the parameterization space. See [9] and references therein for more details on atlas-based parameterization methods. A different parameterization approach can be applied



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

Figure 1: Deformed texture and texture mapping on a genus-0 mesh model: (a) The mesh model, (b) Texture image, (c) Deformed texture, (d) Mesh model with texture mapping

to genus zero meshes with a topology equivalent to a sphere. Gotsman et al. [4] show how to generate a bijective mapping by generalizing barycentric coordinate methods for planar parameterization. Saba et al. [10] present an optimized numerical scheme that solves Gotsman's method efficiently.

In contrast to the proliferation of parameterization methods, only a few techniques satisfy positional constraints. Lévy [8] suggests adding the constraints to the linear parameterization system. He satisfies the constraints in a least-squares manner, yielding a "soft" constraints solution. For many applications such a solution is sufficient. However, when adding large number of constraints the method might result in a non bijective parameterization. Eckstein et al. [2] and Kraevoy et al. [7] suggest a method that guarantees the constraints' position together with the validity of the embedding. Both methods add new vertices, a.k.a. *Steiner* vertices, to the mesh when the constraints cause an over-determined system. Kraevoy's method also relies on a valid parameterization of the original mesh. Karni et al. [5] suggest using a free-boundary linear parameterization method that compels the positional constraints into their place. In case of an invalid solution, they suggest an iterative method that after 'several' iterations fixes the invalid areas. However, this method does not guarantee an upper bound on this number of iterations nor that a valid solution will be reached.

To the best of our knowledge, there are no methods for constrained parameterization of high genus models. Alexa [1] embeds genus zero meshes on a sphere with the presence of constraints. However, this suggestion does not guarantee a solution, even in the event that such a case exists.

3. Overview

Our method deals with the constrained texture mapping challenge differently. Based on the constrained vertices, the image plane is deformed to roughly capture the 3D model's pose and shape (Section 4.1). Similar to Lévy [8] the constraints are satisfied in a least-squares manner. A matching based on geodesic distances is then used to couple the unconstrained vertices and the image pixels (Section 4.2). Using the matching and the initial constraints, the image plane is deformed again into its final shape (Section 4.3). Figure 2 visually presents the different stages of our method. We show that our technique results in a smooth, visually pleasing and realistic textured model that can be imposed onto or used instead of the original model.

Aside from its visual advantages, our method has the virtue of being applicable to any type of a manifold surface, in particular high genus models and non-triangulated meshes, as long as it supports the evaluation of geodesic distances. Figure 1 shows a texture deformed to meet the shape of a human's head model, which is of genus zero (higher genus models can be used in the same way). Our method is not meant to be a parameterization technique and hence does not guarantee a bijective mapping. However, for many cases it achieves a valid parameterization and traditional texture mapping can be also applied as well.

4. Texture deformation

Let M be a 3D mesh (not necessarily triangulated) represented by the pair (V,T), where V represents the vertices' position in $\mathbb{R}^3$ and T represents the mesh connectivity. Let S be an image surface (embedded in 3D

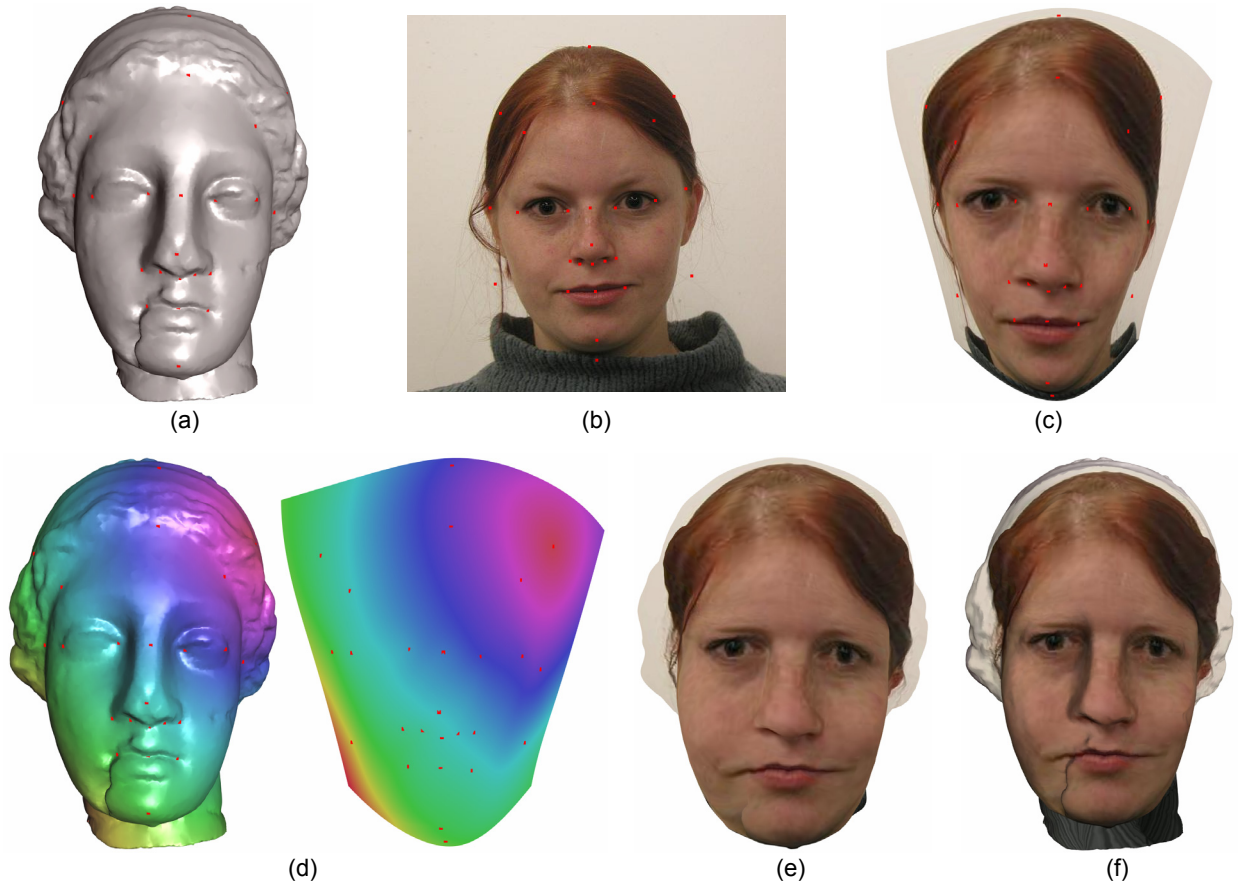


Figure 2: Processing steps: (a,b) A genus-0 mesh model and the texture image together with the constraint points (c) Initially deformed texture (d) Geodesic distances from one constraint point along the mesh and deformed texture surfaces (e) Final deformed texture (f) Texture mapping based on the geodesic distances parameterization

space) represented by the pair (P,C) where P represents the set of pixels' position in $\mathbb{R}^3$ and C the pixels' color. Also, *w.l.o.g.* let (Vi,Pi), $i \in \{1,...,k\}$ ($k << |V|$ and $k << |P|$) be pairs of constraints that couple k vertices with k pixels.

4.1. Initial deformation

Sorkine et al. [12] use the connectivity together with constraints to reconstruct a mesh. They use the uniform Laplacian operator to state the surface smoothness and together with the constraints establish an over determined system. The position of the mesh vertices is the least-squares solution of this system. In a similar way we use a Laplacian based mesh editing technique to deform the image plane into its initial pose. Mesh Laplacian was first introduced by Taubin [14] for mesh processing and Karni and Gotsman [5] for mesh compression.

Let A be an adjacency matrix and D a diagonal matrix such that $D_{ii} = d_i$, where d_i is the degree of vertex i. The matrix $L = I - D^{-1}A$, where I is the identity matrix, is the mesh Laplacian (with uniform weights).

The image plane does not have an explicit connectivity but instead a grid structure. Therefore, we use the Laplacian as it is defined for images. The adjacency matrix holds for each pixel, its four closest neighbors: one from its left, right, bottom and top. Exceptions are boundary pixels with three neighbors and corner pixels with two.

To deform the image surface S we solve the following equations for $\tilde{P}$, the new "pixel" positions, using least-squares:

$$\begin{bmatrix} L_{|P|\times|P|} \\ \hline I_{k\times k} \mid 0 \end{bmatrix} \begin{bmatrix} \tilde{p}_1 \\ \tilde{p}_2 \\ \vdots \\ \vdots \\ \tilde{p}_{|P|} \end{bmatrix} = \begin{bmatrix} 0 \\ \hline v_1 \\ \vdots \\ v_k \end{bmatrix} \qquad (1)$$

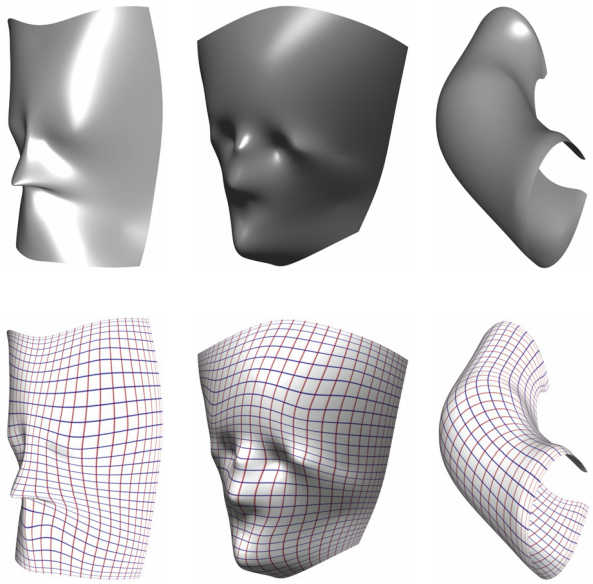


Figure 3: Deformed image plane based on the constraint points alone

The deformed surface $\tilde{S} = (\tilde{P},C)$ roughly captures the mesh pose as it is imposed by the constraints. It is important to note that $\tilde{S}$ does not exactly satisfy the positional constraints. However, it minimizes the following function:

$$\min_{\tilde{P}}\left(\left\|L\tilde{P}\right\|^2 + \sum_{i=1}^{k}\left\|\tilde{p}_i - v_i\right\|^2\right)$$

Besides mimicking the mesh pose, the initial deformation has an important role in scaling the image surface to approximate the area of the mesh surface. This stage is essential for the matching processes described in the next section. **Error! Reference source not found.** shows several examples of deformed image planes based on the constraints points alone. It is easy to notice that the intrinsic parameterization of the image plane is preserved during the deformation.

We would like to refer the interested reader to the excellent state-of-the-art report by Sorkine [11] and the references therein for more details on Laplacian mesh processing.

4.2. Matching

The matching stage couples the rest of the mesh vertices $v_i\ i\in\{k+1,\ldots,|V|\}$ with correspondence pixels $\{p_i\}\ i\in\{k+1,\ldots,|P|\}$. The matching is one-to-one but not onto, meaning that several pixels exist without a matching (it is common to assume that the number of pixels in the image is much larger than the number of vertices in the mesh).

The matching should capture the intrinsic properties of the two surfaces. For example: a vertex that lies between two constraint vertices should be matched with a pixel that lies between the corresponding constraint pixels. Zayer et al. [15] used vector fields generated by harmonic maps to match between two meshes for the application of transformation transfer. The principle of the matching is: Let F_M and F_S be the vector fields along the mesh and image surface, respectively. For each vertex $v_i\ i\in\{k+1,\ldots,|V|\}$, we match the pixels $p_j\ j\in\{k+1,\ldots,|P|\}$ such that $j = argmin(\|F_{M_i} - F_{S_j}\|)$ and $\|F_{M_i} - F_{S_j}\| \le TH$. The second term prevents the matching of points that are too far from any constraints and the difference between their vector fields is too high. In our implementation TH is defined to be one percent of the model's bounding box multiplied by k (the number of constraints).

Harmonic functions have a cyclic nature that can cause the mapping not to be bijective. Therefore, our matching is based on vector field of geodesic distances. We calculate the distances using the method introduced by Surazhsky et al. [12], starting from each constraint vertex to the entire mesh vertices and from the con-

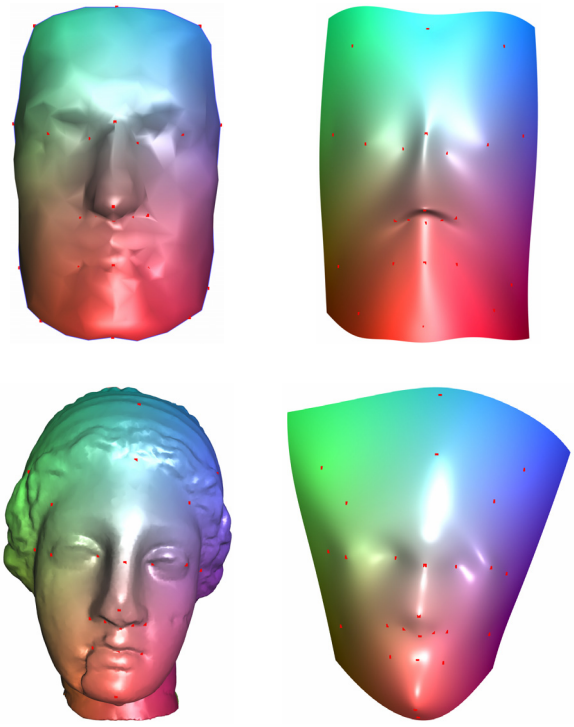


Figure 4: Geodesic distances along the surfaces of the Igea and face models together with their corresponding deformed image surface. Each color represents one entry in the distances vector field.

straint pixels to the rest of the image pixels on the deformed image surface. This generates a vector field of dimension k (each entry is a distance from one constraint vertex or pixel) along the surfaces of the mesh and the deformed image. Figure 4 shows a three-dimensional subset of the vector field as [R,G,B] colors on the Igea and face model surfaces together with their deformed image surface.

Deforming an image around a disc-like mesh surface using geodesic distances, results in a valid matching. However, when dealing with closed meshes (genus-0) or high-genus models, a trivial geodesic distance matching might lead to a mismatch. Figure 5 demonstrates the problem for a simple two-dimensional case. The bar, analog to an image plane, is to deform around the circle, analog to a genus-0 cylinder, using the marked constraints. Observe that the distance along the bar between points 5 and 6 is significantly different than the distance between the corresponding points on the circle. This might lead to matching between points in the surroundings of point 4 on the bar to points in the surroundings of point 6 on the circle and to a mismatch which will result in a distorted deformation and for sure will lead to an invalid parameterization.

To avoid this problem we recognize a potential mismatch by inspecting the distance vectors at the constraint vertices and pixels for significant variations. A

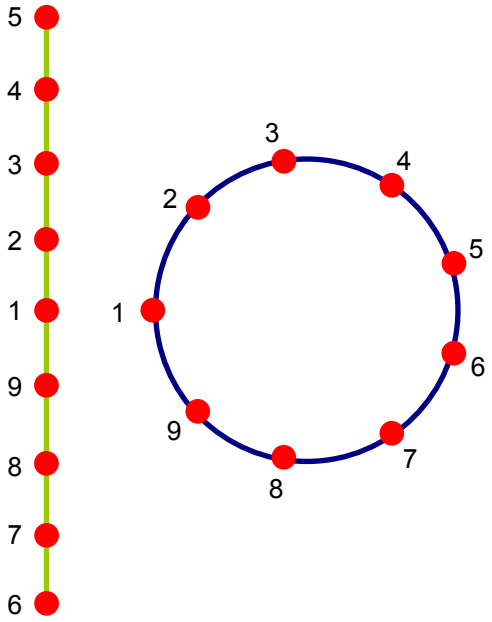


Figure 5: A two-dimensional analog to the deformation of a plane (represented as a bar) around a sphere (represented as a circle) and the constraints points between them.

Figure 6: Texture mapping of a tiger image onto the face model (left) and the parameterization domain (right).

mismatch is recognized when the ratio FM_i/FS_i at the constraint points $i \in \{1,...,k\}$ is below 0.8 or above 1.25. When such a mismatch is discerned, the suspected entries are eliminated from the distance vectors for pixels and vertices close to the suspected region.

4.3. Final deformation

The final deformation brings the image surface into its final pose. Similar to what was described in Section 4.1, a Laplacian based deformation is used, but instead of considering only the initial constrained points, the entire set of matched points is added. This results in a new, over-determined linear system that is a generalization of Eq. (1) and is solved using least-squares.

$$\begin{bmatrix} L_{|P|\times|P|} \\ \hline \omega I_{k\times k} \mid 0 \\ \hline 0 \mid \sigma I_{(|V|-k)\times(|V|-k)} \mid 0 \end{bmatrix} \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \vdots \\ \vdots \\ \hat{p}_{|P|} \end{bmatrix} = \begin{bmatrix} 0 \\ \hline \omega v_1 \\ \vdots \\ \omega v_k \\ \hline \sigma v_{k+1} \\ \vdots \\ \sigma v_{|V|} \end{bmatrix} \quad (2)$$

The least squares solution minimizes the following function:

$$\min_{\hat{P}} \left(\left\| L\hat{P} \right\|^2 + \sum_{i=1}^{k} \omega^2 \left\| \hat{p}_i - v_i \right\|^2 + \sum_{i=k+1}^{|V|} \sigma^2 \left\| \hat{p}_i - v_i \right\|^2 \right)$$

(a) (b) (c)

Figure 9: Constrained Texture Deformation: (a) Polygonal surface (b) Texture image (c) Deformed texture surface together with the texture image

Figure 7: Deformation based on geodesic distances measured along the deformed image surface (left) compared with deformation based on geodesic distances measured on the image plane (right).

By adjusting the value of ω, the user can control the penalty measure for any deviation from the original constraints position. High value of ω is considered as hard-constraints. However, forcing hard-constraints might result in internal surface intersections and other not visually pleasing results. By adjusting the value of σ the user can control the smoothness degree of the deformed surface. A small value of σ will reduce the influence of the matched points. For example, setting its value to 0 will result in the initial smooth deformed surface.

4.4. Texture coordinates

Matching the vertices of a mesh surface to image pixels establishes texture-coordinates. Therefore, we can use the matching results of Section 4.2 to render the image plane on the mesh surface using traditional texture mapping. Figure 6 shows the texture mapping of the tiger onto the face model together with its parameterization. However, the matching stage neither guarantees to cover the entire mesh vertices, nor to establish a valid mapping.

Zigelman et al. [16] used a multi-dimensional-scaling technique on the geodesic distances to embed the mesh vertices on a 2D plane by preserving their original distances along the mesh surface. The embedding on a plane by itself is sufficient to generate texture-coordinates. However, their method that aims on facial texture-mapping also does not guarantee the validity of the mapping, and also does not satisfy positional constraints.

Figure 8: Deformed texture along one handle of the Figure Eight, high-genus, model

An analog to Zigelman's method would be to match the mesh vertices to the image pixels based on the geodesic distances along the mesh surface and the image plane (instead of the deformed image surface). Calculating geodesic distances along a plane sums up to calculating Euclidian distances. Figure 7 shows the differences between the two approaches. It is easy to see that our approach (on the left) visually performs better. The distortions in the Euclidian distances method (right image) and the failure to meet the constraints are due to the differences in the distance measurements, especially around the nose area.

5. Results

We implemented the suggested method and it works as follows: The user loads a polygonal model and an image and interactively marks pairs of constraints points between the two. From this point on the system is completely automatic. It generates the initial deformed surface, calculates the geodesic distances and uses them for matching. Finally, it generates the final deformed surface. The user can then interactively add and remove constraints and change the weights of the final least-square system (see Section 4.3) to fine-tune the result and fit it to its needs.

We tested our method with several types of surfaces. Figure 9 shows a texture-deformation for a disc-like surface. In Figure 2e and Figure 1 our method is challenged with genus-0 models, and in Figure 8, a texture plane is deformed around one handle of the Figure Eight model (a genus-2 model). All the results show that the deformed texture-surface followed the shape of the mesh-surface while keeping the positional constraints in place, as much as they can be held, due to the least-squares nature of the solution.

In Figure 10 our method was tested with a noisy mesh. Gaussian noise was added to the Igea model and the same texture plane as in Figure 2 was deformed over it. It is evident that in spite of the noise over the mesh surface, the matching based on geodesic distances was successful and the deformed texture-surface inherited the noisy nature of the model surface. By fine-tuning the weights of the second deformation stage, we were able to generate a smooth version of the deformed texture while preserving all the facial details. The smoothed version looks visually better and natural.

In Figures 2, 8 and 10 we provide a texture-mapped model based on geodesic-distances parameterization. We did not check for the validity of the mapping although visually we could not find any evidence for it being invalid. In all examples the texture is well placed and the model looks visually good. However, the smooth version of the deformed texture-surface seems to us more natural, and in other words, better.

5.1. Statistics

All experiments were preformed using an AMD Opteron-250 system with 2GB of memory. Table 1 summarizes the performance of our method on several of the models. We would like to emphasize that the timings listed in Table 1 are for the initial constraints points. Adding or removing constraints requires only an update of the already factorized deformation systems and the evaluation of geodesic distances from the new points alone.

Table 1: Time performances for the different stages of the "The Mask" system (in sec)

Model	Face	Igea	Igea	Male	Eight
Vertices	1394	33K	33K	220K	1536
Pixels	40K	65K	262K	65K	65K
Constraints	24	26	26	24	26
Deform 1	2.25	4.45	35.03	4.32	4.42
Mesh Geod	0.12	14.38	14.38	31.2	0.312
Text Geod	15.36	27.56	123.24	25.44	27.56
Matching	0.01	0.49	3.36	1.17	1.17
Deform 2	2.23	4.52	36.51	4.48	4.48
Total	19.97	51.4	212.52	66.61	36.83

It is evident that the major factor in the method's time performance is the size of the texture image. It influences the performances of the deformation stages and the geodesic distance calculations. This is not surprising. In most cases the number of pixels in the texture image is significantly larger than the number of vertices in the mesh model. In our experiments we used small and medium sized images. However, we would like our method to handle images with 5 to 10 Mega-pixels, such as current digital cameras produce.

It is important to note that the texture image surface has the special connectivity structure of a grid. Our solver, which was used in the deformation stages, does not exploit this property. We believe that by using solvers and an algorithm for geodesic distance computation that are dedicated to work on grid structures (e.g., multi-grid methods), we will be able to significantly reduce the computation time of those stages and enable the use of large images.

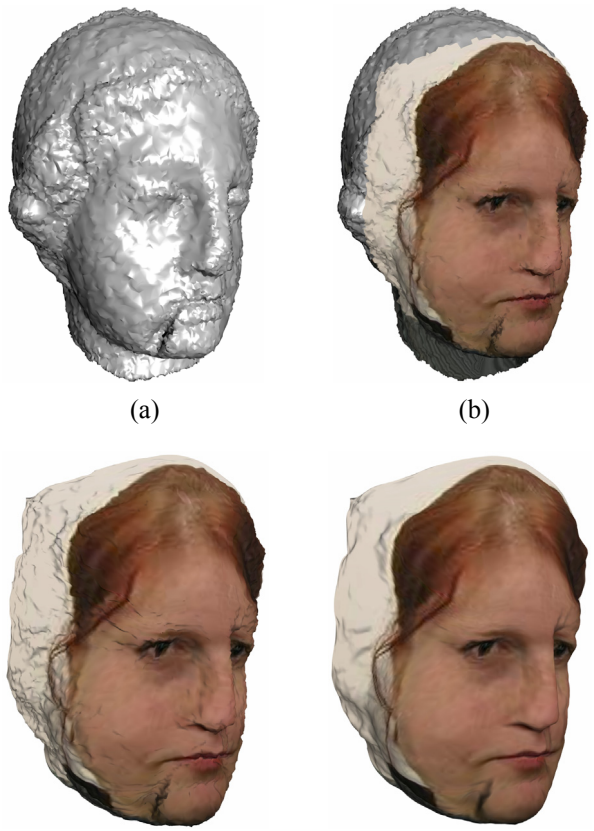


Figure 10: Deformed texture and texture mapping on noisy mesh model: (a) Noised Igea mesh model (b) Texture mapping on noisy Igea model (c) Deformed texture with smoothing σ=1.0 (d) Deformed texture with smoothing σ=0.1.

6. Summary and future work

We present a method that deforms an image plane to meet the geometry of a mesh surface. The deformation results in a smooth and visually pleasing surface that can be used instead of the original or as a parameterization of it. We show that the method performs on any type of mesh surface, even those with high genus.

There are several directions for future work: The least-squares optimization solver that is used for the image surface deformation should be replaced by one that will exploit the grid structure of an image. Such a solver will enable our method to be interactive even when dealing with today's image sizes.

In texture mapping, fine details in the texture are flattened onto the polygon surface. For example, the fine details of the tiger's whiskers are not influenced in the geometry. They are snapped onto the polygon surface and hence look flat. The fine geometry structure of the deformed image surface enables us to add the image's fine details to the geometry. This will lead to highly detailed and much more realistic models.

Finally, this method can serve as a fundamental basis for surface reconstruction out of points-clouds. Developing this is a real challenge, mostly because there is no easy or fast way to calculate geodesic distance along a points-cloud surface.

7. Acknowledgments

We thank Vitaly Surazhsky for sharing his method for geodesic distance calculation on meshes. This work was supported in part by AIM@SHAPE, a Network of Excellence project (506766) within EU's Sixth Framework Program.

8. References

[1] M. Alexa, "Merging Polyhedral Shapes with Scattered Features", The Visual Computer, 16, pp 26-37, 2000.

[2] I. Eckstein, V. Surazhsky and C. Gotsman, "Texture Mapping with Hard Constraints", Proceedings of Eurographics, 2001.

[3] M. S. Floater and K. Hormann, "Surface Parameterization: A Tutorial and Survey", Advances in Multiresolution for Geometric Modeling, Springer, pp 157-186, 2004.

[4] C. Gotsman, X. Gu and A. Sheffer, "Fundamentals of Spherical Parameterization for 3D Meshes", ACM Transactions on Graphics (SIGGRAPH 2003), 22(3), pp 358-363, 2003.

[5] Z. Karni and C. Gotsman, "Spectral Compression of Mesh Geometry", International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2000), pp 279-286, 2000.

[6] Z. Karni, C. Gotsman and S. J. Gortler, "Free-Boundary Linear Parameterization of 3D Meshes in the Presence of Constraints", Proceedings of Shape Modeling International, pp 266-275, 2005.

[7] V. Kraevoy, A. Sheffer and C. Gotsman, "Matchmaker: Constructing Constrained Texture Maps", ACM Transactions on Graphics (SIGGRAPH 2003), 22(3), pp 326-333, 2003.

[8] B. Lévy, "Constrained Texture Mapping for Polygonal Meshes" , International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2001), pp 417-424, 2001.

[9] B. Lévy, S. Petitjean, N. Ray and J. Maillot, “Least Squares Conformal Maps for Automatic Texture Atlas Generation”, ACM Transactions on Graphics (SIGGRAPH 2002), 21(3), pp 362-371, 2002.

[10] S. Saba, I. Yavneh, C. Gotsman and A. Sheffer, “Practical Spherical Embedding of Manifold Triangle Meshes”, Proceedings of Shape Modeling International, 2005.

[11] O. Sorkine, “State-of-The-Art Report: Laplacian Mesh Processing”, Eurographics 2005.

[12] O. Sorkine and D. Cohen-Or, “Least-Squares Meshes”, Proceedings of Shape Modeling International, pp 191-199, 2004.

[13] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler and H. Hoppe, “Fast Exact and Approximate Geodesics on Meshes”, ACM Transactions on Graphics (SIGGRAPH 2005), 24(3), pp 553-560, 2005.

[14] G. Taubin, “A Signal Processing Approach to Fair Surface Design”, International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1995), pp 351-358, 1995.

[15] R. Zayer, C. Rössl, Z. Karni and H. P. Seidel, “Harmonic Guidance for Surface Deformation”, Computer Graphics Forum (Proceedings of Eurographics), 24(3), pp 601-609, 2005.

[16] G. Zigelman, R. Kimmel and N. Kiryati, “Texture Mapping Using Surface Flattening via Multi-Dimensional Scaling”, IEEE Transactions on Visualization and Computer Graphics, 8(2), pp 198-207, 2002.