Gradient-Domain Processing of Meshes

Ming Chuang
Microsoft

Szymon Rusinkiewicz
Princeton University

Misha Kazhdan
Johns Hopkins University

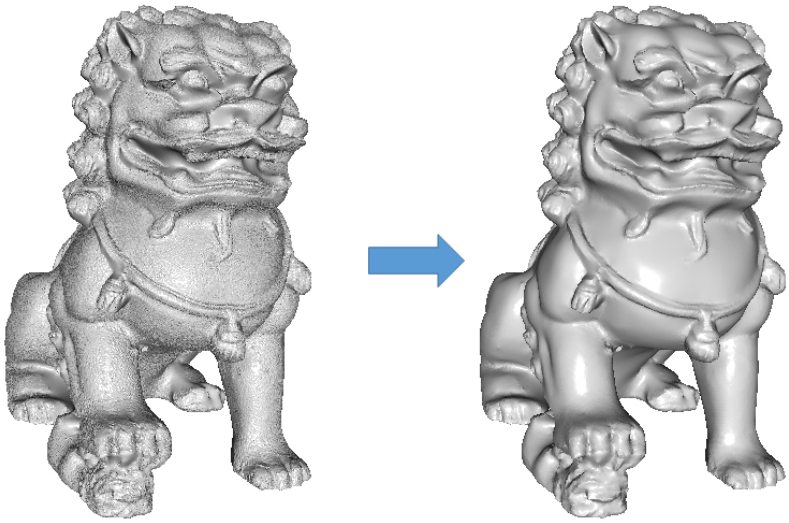

Figure 1. A noisy model before (left) and after (right) anisotropic gradient-domain smoothing.

Abstract

This paper describes an implementation of gradient-domain processing for editing the geometry of triangle meshes in 3D. We show applications to mesh smoothing and sharpening and describe anisotropic extensions that enable edge-aware processing.

1. Introduction

Gradient-domain processing is a well-established technique for editing the content of RGB images. Formulated as the minimization of an energy function that combines value- and derivative-fitting terms, it provides a simple method for adjusting the content of an image and has been used for operations like smoothing, sharpening, stitching, dynamic range compression, and color correction [Fattal et al. 2002; Pérez et al. 2003; Levin et al. 2004; Agarwala et al. 2004; Agarwala 2007; Bhat et al. 2008; Kazhdan et al. 2015].

 ISSN 2331-7418

In this work we describe a simple and unified extension of this approach to the signals defined over triangle meshes in 3D. In particular, when the signal is taken to be the x-, y-, and z-coordinates of the mesh vertices, our implementation supports the processing of the geometry itself. In doing so, we combine earlier work in both isotropic smoothing [Taubin 1995; Desbrun et al. 1999] and anisotropic smoothing [Clarenz et al. 2000; Bajaj and Xu 2003; Andreux et al. 2014], as well as work in gradient-domain stitching for fusing surfaces [Yu et al. 2004].

Unifying these approaches in a common setting, we provide a single formulation for solving a wide class of geometry-processing problems. As we show, the extension of the approach to the anisotropic setting requires a simple modification of the well-known formulas for computing the mass and stiffness matrices [Dziuk 1988; Pinkall and Polthier 1993], modifying the traditional per-element assembly by incorporating a constant metric-tensor with each triangle to define integration and differentiation.

1.1. Related Work

Extensions of 2D diffusion techniques to the smoothing of surface geometry began with the seminal work of Taubin [1995], which was later formulated as a semi-implicit time-step of a diffusion PDE, $\partial_t \rho = \Delta_g \rho$, in the work of Desbrun et al. [Desbrun et al. 1999]. The advantage of this formulation was that smoothing was formulated in terms of the metric-dependent Laplace-Beltrami operator, Δ_g. Modifying the Riemannian metric, g, this approach could be extended to anisotropic diffusion to support feature-aware smoothing [Clarenz et al. 2000; Bajaj and Xu 2003; Andreux et al. 2014].

In more recent work, Chuang and Kazhdan [2011] describe an equivalent formulation that expresses a smoothed surface as the minimizer of the sum of interpolation and smoothness energies, $E(\rho) = \|\rho - \rho_0\|_g^2 + \|\nabla\rho\|_g^2$. Using the observations of [Clarenz et al. 2000; Bajaj and Xu 2003], Chuang and Kazhdan show that the energy-minimization formulation can also be used to support anisotropic diffusion through the modification of the Riemannian metric. Following the approach for image-processing [Bhat et al. 2008], the authors further extend the approach by replacing the Dirichlet energy, $\|\nabla\rho\|_g^2$, with a more general gradient-domain energy term, $\|\nabla\rho - \vec{V}\|_g^2$, that allows explicit prescription of target gradient values.

In their work, Chuang and Kazhdan implemented the processing using a multigrid formulation defined by restricting regularly spaced 3D B-Splines to the surface. Here we discretize the system using the ubiquitous "hat" basis functions (e.g., [Dziuk 1988; Pinkall and Polthier 1993]) and derive an implementation that uses standard direct solvers (e.g., [Chen et al. 2008; Guennebaud et al. 2010]).

There are two advantages to formulating geometry-processing in terms of minimizing a gradient-domain energy rather than through a diffusion PDE:

1. The energy minimization formulation supports a wider class of surface edits, including smoothing (as in [Desbrun et al. 1999; Clarenz et al. 2000; Bajaj

and Xu 2003; Andreux et al. 2014]), gradient-domain stitching (as in [Yu et al. 2004]), and sharpening [Chuang and Kazhdan 2011].

2. Though both approaches give the same linear system, the derivation using energy minimization is simpler since it only requires computing first-order derivatives. In contrast, using the diffusion PDE requires first defining the second-order Laplace-Beltrami operator (including differentiation of the metric tensor) and then eliminating second-order terms using Stokes' Theorem.

2. Gradient-Domain Processing

Given a 2D manifold $\mathcal{M} \subset \mathbb{R}^3$, a scalar function $f : \mathcal{M} \to \mathbb{R}$, and a tangent vector-field $\vec{v} : \mathcal{M} \to T\mathcal{M}$, the goal of gradient domain-processing is to solve for the new function, $g : \mathcal{M} \to \mathbb{R}$ that minimizes the energy [Bhat et al. 2008]:

$$E(g) = \alpha \int_{\mathcal{M}} [f(p) - g(p)]^2 \, dp + \beta \int_{\mathcal{M}} \|\vec{v}(p) - \nabla g(p)\|^2 \, dp. \tag{1}$$

Using the Euler-Lagrange formulation, the minimizer is the solution to the system:

$$(\alpha - \beta\Delta)\, g = \alpha \cdot f - \beta \cdot \text{div}(\vec{v}) \tag{2}$$

where $\text{div}(\vec{v})$ is the divergence of $\vec{v}$ and Δ is the (negative semi-definite) Laplace-Beltrami operator, $\Delta f = \text{div}(\nabla f)$.

2.1. Applications

The above gradient-domain formulation has proven to be a powerful tool for editing signals and has been used in a number of applications. For example, when setting the constraint vector field to be a multiple of the gradient, $\vec{v} = \lambda \cdot \nabla f$, one gets:

- *Smoothing* ($\lambda < 1$): This encourages a solution that is close to the input but with less pronounced changes in value. In the extreme case that $\lambda = 0$, the solution g is equivalent to the result of performing an implicit time-step of diffusion on the signal. And, in the particular case that the signal consists of the x-, y-, and z-coordinates of the surface, this gives a (semi)-implicit time-step of mean-curvature flow (Figure 2 (left)).

- *Sharpening* ($\lambda > 1$): This encourages a solution that is close to the input but with more pronounced changes in value. This has the effect of amplifying changes in the signal value, effectively acting as an unsharp masking filter (Figure 2 (right)).

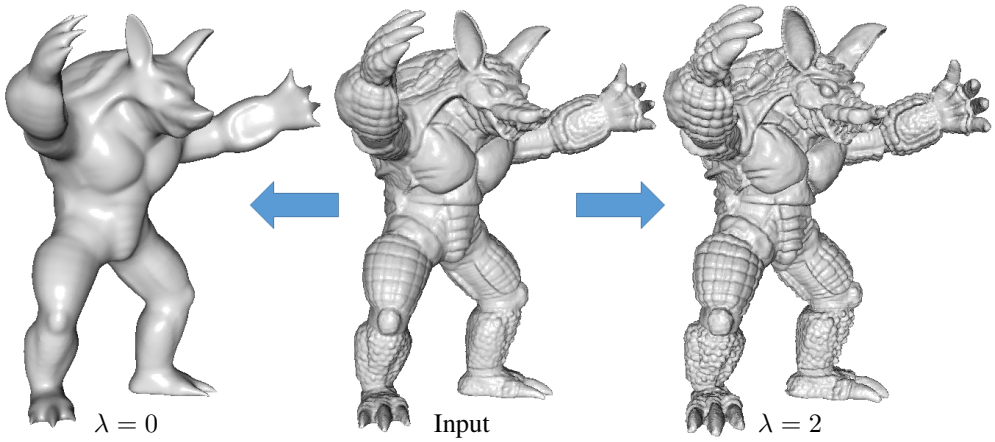

Figure 2. An input mesh (center) and the results obtained with gradient-domain smoothing (left) and gradient-domain sharpening (right).

2.2. Discretization

To discretize Equation (2) over a triangle mesh $\mathcal{M} = (\mathcal{V}, \mathcal{T})$, we must choose bases for the spaces of scalar fields and vector fields. In our implementation we follow the standard approach. We define the space of scalar fields to be the span of piecewise-linear "hat" basis functions, associating the space of scalar fields with $\mathbb{R}^{|\mathcal{V}|}$. And, we define the space of vector fields by first choosing an arbitrary frame for each triangle and then taking the span of the piecewise-constant vector fields, associating the space of vector fields with $\mathbb{R}^{2|\mathcal{T}|}$.

Letting $\{\boldsymbol{\phi}_\nu\}_{\nu\in\mathcal{V}}$ and $\{(\vec{\mathbf{t}}_{2\tau}, \vec{\mathbf{t}}_{2\tau+1})\}_{\tau\in\mathcal{T}}$ denote the bases for these two spaces, we can define discrete mass, stiffness, and divergence operators:

$$\mathbf{M}_{ij} = \int_{\mathcal{M}} \phi_i \cdot \phi_j, \qquad \mathbf{S}_{ij} = \int_{\mathcal{M}} \langle \nabla\phi_i, \nabla\phi_j \rangle, \qquad \mathbf{D}_{ij} = \int_{\mathcal{M}} \langle \nabla\phi_i, \vec{\omega}_j \rangle.$$

This gives a discretization of Equation (2) as

$$(\alpha\mathbf{M} + \beta\mathbf{S})\mathbf{g} = \alpha\mathbf{M}\mathbf{f} + \beta\mathbf{D}\mathbf{v},$$

where $\mathbf{f} \in \mathbb{R}^{|\mathcal{V}|}$ are the coefficients of the target scalar field, $\mathbf{v} \in \mathbb{R}^{2|\mathcal{T}|}$ are the coefficients of the target vector field, and $\mathbf{g} \in \mathbb{R}^{|\mathcal{V}|}$ are the coefficients of the solution.

As the mass and stiffness matrices are sparse symmetric and positive (semi-)definite, the system can be solved using a sparse Cholesky factorization (e.g., using either the Eigen [Guennebaud et al. 2010] or the CHOLMOD [Chen et al. 2008] package).

2.3. Edge-Aware Gradient-Domain Processing

One standard way to compute the coefficients of the system matrices is using finite-element assembly. Specifically, noting that the scalar (respectively, vector) basis

functions are strictly linear (respectively, constant) within each triangle, the integrals defining the matrix coefficients can be computed by computing the definite integrals over each triangle and summing the individual contributions. (For more details, we refer the reader to the appendix.)

The advantage of this decomposition is that it allows us to replace the metric induced by the embedding of a triangle in 3D with a user-specified metric. As an example, Figure 1 shows the results obtained when performing gradient-domain smoothing where we have scaled the metric on each triangle by $\left(1 + \varepsilon \cdot (\kappa_1^2 + \kappa_2^2)\right)$, with (κ_1, κ_2) the principal curvatures estimated at the triangle. This has the effect of increasing distances in regions of high curvature, making it harder for the diffusion to propagate across sharp features, thereby resulting in edge-aware smoothing.

More formally, when we (uniformly) scale a region by a factor σ, this impacts Equation (1) in two ways. First, the area over which the integral is computed is multiplied by a factor of σ^2, so that both the value- and gradient-integrals scale in the same way. Second, scaling a region by σ scales the gradient of a function by $1/\sigma$, so the gradient-integral is additionally scaled by $1/\sigma^2$. Taken in combination, scaling by a factor of σ scales the value-integral by σ^2 and leaves the gradient-integral unchanged. Thus, when the scale factor σ is greater than one, this gives more weight to value interpolation and when it is less than one, it gives more weight to gradient interpolation.

Anisotropic filtering

We can also support anisotropic geometry processing by separately scaling along the principal curvature directions. For example, Figure 3 shows an input model (center), and the results obtained by only allowing the smoothing to occur across regions of negative (left) and positive (right) curvature. To preserve the positively curved fea-

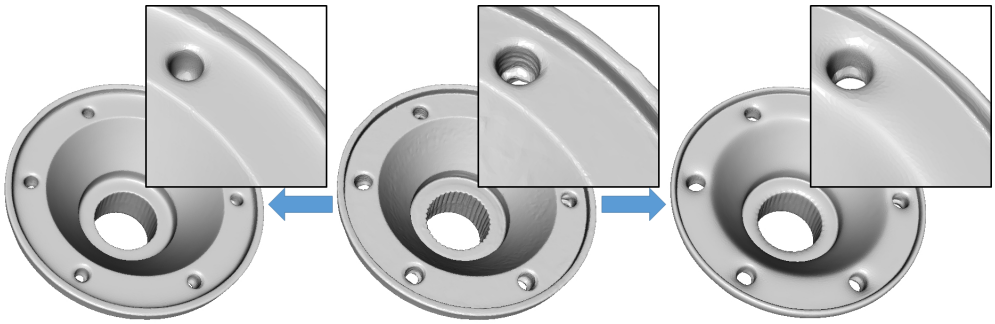

Figure 3. An input mesh (center), results obtained by smoothing across regions of negative curvature (left), and results obtained by smoothing across regions of positive curvature (right). At the insets show, only smoothing along regions of negative curvature preserves the convex features while only smoothing along regions of positive curvature preserves the concave.

tures we "stretch" the metric along directions of positive curvature, supressing the effects of diffusion along these directions. This is done by setting the scale factor associated to ith principal curvature direction to be

$$1 + \varepsilon \cdot \max(0, \kappa_i)^2.$$

Similarly, we preserve the negatively curved features by "stretching" the metric along directions of negative curvature, setting the scale factor to be

$$1 + \varepsilon \cdot \min(0, \kappa_i)^2.$$

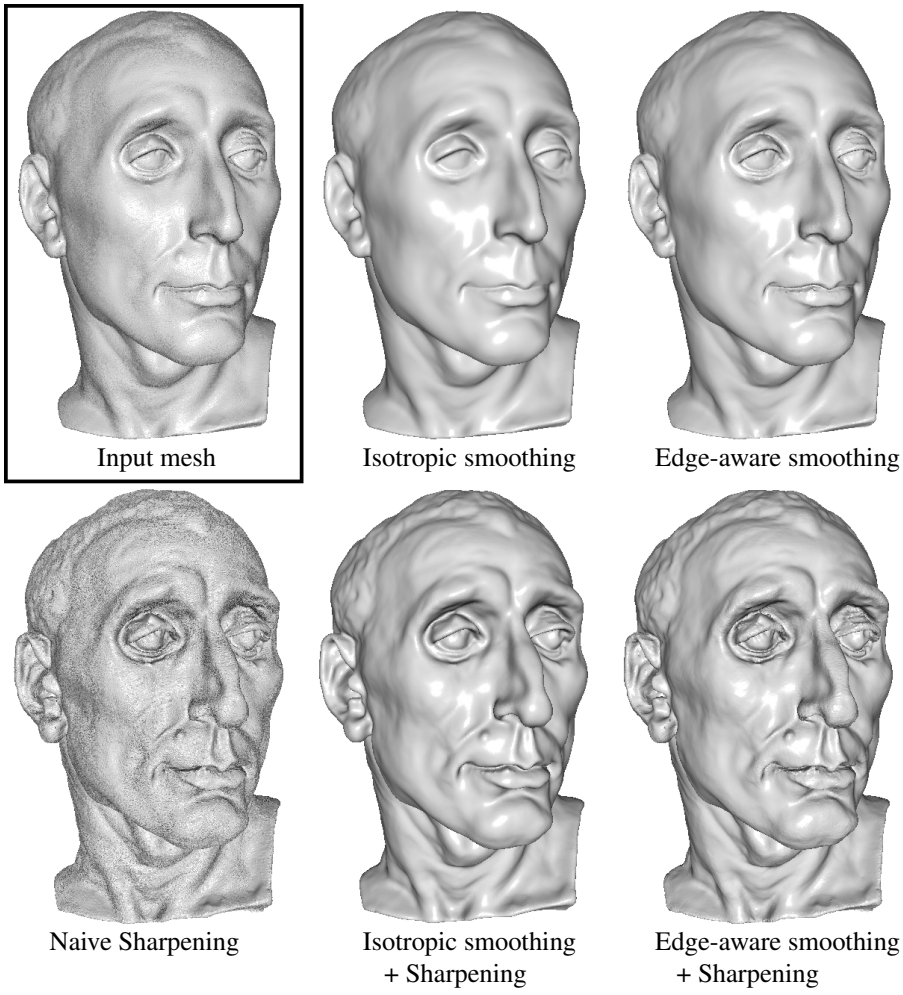

Figure 4. An input mesh (top left), results obtained by performing naive sharpening (bottom left), the results obtained by first performing isotropic smoothing (top center) and then sharpening (bottom center), and the results obtained by first performing edge-aware smoothing (top right) and then sharpening (bottom right).

2.4. Two-pass Filtering

As shown in the work of Taubin [1995], effects like band-pass filtering can be simulated by first running a smoothing pass to filter out the noise, followed by a sharpening pass (with a narrower filter) to re-introduce the detail within a particular band. We leverage this approach to support detail enhancement on noisy meshes. As shown in Figure 4, when the input (top left) is noisy, a naive sharpening pass accentuates the noise in the geometry as well as the features (bottom left). This problem can be resolved by first running an isotropic smoothing pass to remove the noise (top center) and then running a sharpening pass on the smoothed geometry (bottom center). Using our framework, we can further preserve the sharp detail by performing edge-aware smoothing in the first pass (top and bottom right).

3. Evaluation

We implemented our system using the publicly available Eigen and CHOLMOD solvers, using both the default and MKL-backed implementations of Eigen. For Eigen, we used the "LLT" factorization. For CHOLMOD, we used the default settings. (The implementation is publicly available at [Kazhdan 2016]). Table 1 summarizes the model sizes, running times, and memory usage for processing the models shown in Figures 1–4. For most evaluations, the CHOLMOD solver out-performs the two implementations of Eigen. It is also the case that the MKL-backed implementation of Eigen out-performs the default implementation, with more noticeable gains for large systems.

		Eigen		Eigen + MKL		CHOLMOD	
Figure	Vertices	Time	Memory	Time	Memory	Time	Memory
1	656K	33.4	971	14.6	1025	11.9	920
2	173K	4.3	228	4.3	263	3.2	224
3	81K	2.5	114	2.0	126	1.5	117
4	947K	91.9	1561	20.9	1524	22.7	1240

Table 1. Size, running time (seconds), and memory usage (megabytes) for processing the models shown in Figures 1–4 using different implementations the Cholesky factorization.

A more fine-grained analysis is shown in Table 2 which decomposes the linear solve time as the sum of symbolic factorization, numerical factorization, and back-substitution. It is interesting to note that although the MKL-based implementation is faster than the default Eigen implementation, its back-substitution time is noticeably slower. Thus, in scenarios where one needs to repeatedly solve the same system over and over again, it is not necessarily the case that the MKL-based implementation provides the better solution.

Figure	Eigen	Eigen + MKL	CHOLMOD
1	14.4 + 12.4 + 0.7	3.1 + 0.6 + 4.4	3.1 + 1.8 + 0.6
2	1.4 + 1.0 + 0.1	1.0 + 0.2 + 1.3	0.8 + 0.4 + 0.1
3	0.9 + 0.7 + 0.1	0.5 + 0.1 + 0.6	0.4 + 0.2 + 0.1
4	41.7 + 39.2 + 1.1	4.3 + 1.0 + 6.0	9.4 + 2.5 + 0.9

Table 2. Fine-grained running times (seconds) for solving the linear systems associated with processing the models shown in Figures 1–4. The running times are decomposed as the sums of the symbolic factorization, numerical factorization, and back-substiution times.

4. Conclusion and Future Work

In this work we have described an extension of the image-based gradient-domain-processing paradigm to the processing of surface geometries. We have shown that the associated linear systems are not difficult to formulate and can be easily adapted by modifying the underlying Riemannian metric. Leveraging existing direct solvers, we show that this provides a simple tool for removing noise and exaggerating detail in 3D meshes, while constraining the editing process to be edge-aware.

In future work, we would like to consider two different extensions of this approach. We would like to explore the effects of replacing the Laplace-Beltrami operator by the Hessians of other deformation energies, as proposed by Hildebrandt et al. [2012]. Such an approach could have the advantage of using an operator whose lower frequency eigenvectors better respect the extrinsic features of the geometry, providing an alternate means for performing edge-aware smoothing. And, more generally, we would like to explore the spectral properties of the anisotropic operators in the context of manifold harmonics [Vallet and Lévy 2008].

A. Appendix: Matrix Assembly

In this appendix we describe how to compute the coefficients of the system matrix. As we construct the system matrices using finite-element assembly, we focus on computing the per-triangle contributions. For a given triangle $\tau \in \mathcal{T}$, we assume that τ is parameterized over the unit right triangle $\mathbb{T}$ (the triangle in $\mathbb{R}^2$ with vertices $\{\mathbf{v}_0 = (0,0), \mathbf{v}_1 = (1,0), \mathbf{v}_2 = (0,1)\}$), and that we are given a symmetric positive definite matrix $g^\tau \in \mathbb{R}^{2\times 2}$ describing the constant metric over $\mathbb{T}$. For example, given a 3D triangle with vertices $\{v_0, v_1, v_2\} \subset \mathbb{R}^3$, the metric tensor defined by the 3D embedding is

$$g^\tau = \begin{pmatrix} \langle v_1 - v_0, v_1 - v_0 \rangle & \langle v_1 - v_0, v_2 - v_0 \rangle \\ \langle v_2 - v_0, v_1 - v_0 \rangle & \langle v_2 - v_0, v_2 - v_0 \rangle \end{pmatrix}.$$

A.1. Mass Matrix

Given the hat functions $\{\boldsymbol{\phi}_i : \mathbb{T} \to \mathbb{R}\}$, with $\boldsymbol{\phi}_i$ the linear function on $\mathbb{T}$ satisfying $\boldsymbol{\phi}_i(\mathbf{v}_j) = \delta_{ij}$, the coefficients of the mass matrix, $\mathbf{M}^\tau \in \mathbb{R}^{3\times 3}$, are defined by integrating the products of the basis functions over the triangle, relative to the metric:

$$\mathbf{M}^\tau_{ij} = \int_{\mathbb{T}} \boldsymbol{\phi}_i \cdot \boldsymbol{\phi}_j \, dg_\tau.$$

Using the fact that the area of the triangle is one half the square-root of the determinant of the metric tensor, this gives

$$\mathbf{M}^\tau = \sqrt{\det(g^\tau)} \begin{pmatrix} 1/6 & 1/12 & 1/12 \\ 1/12 & 1/6 & 1/12 \\ 1/12 & 1/12 & 1/6 \end{pmatrix}.$$

A.2. Gradient Matrix

Given a linear function on $\mathbb{T}$, we define the gradient as the *cotangent* vector-field (i.e., the differential). This allows us to define the gradient operator in a metric-independent fashion. In particular, we set $\{\vec{\mathbf{t}}_1 = \mathbf{v}_1 - \mathbf{v}_0, \vec{\mathbf{t}}_2 = \mathbf{v}_2 - \mathbf{v}_0\}$ to be vectors spanning the tangent space of $\mathbb{T}$. Then the cotangent space is spanned by the dual vectors $\{\vec{\mathbf{t}}^*_1, \vec{\mathbf{t}}^*_2\}$ and the gradient matrix, $\mathbf{G} \in \mathbb{R}^{3\times 2}$, is defined by taking finite-differences:

$$\mathbf{G} = \begin{pmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}.$$

A.3. Divergence Matrix

By Stokes' Theorem, the coefficients of the divergence matrix, $\mathbf{D}^\tau \in \mathbb{R}^{2\times 3}$, is defined by integrating the dot-product of a cotangent vector field, $\vec{\mathbf{t}}^*_i$, with the gradient of a hat basis functions, $\nabla\boldsymbol{\phi}_j$:

$$\mathbf{D}^\tau_{ij} = \int_{\mathbb{T}} \langle \vec{\mathbf{t}}^*_i, \nabla\boldsymbol{\phi}_j \rangle_{g^\tau} \, dg^\tau.$$

Using the fact that the metric tensor g^τ can be interpreted as a map from the tangent space to its dual, and that the inverse $(g^\tau)^{-1}$ is the map from the dual to the primal, this gives

$$\mathbf{D}^\tau = \sqrt{\det(g_\tau)} \cdot \mathbf{G}^t \circ (g^\tau)^{-1}.$$

A.4. Stiffness Matrix

Finally, the stiffness matrix, $\mathbf{S}^\tau \in \mathbb{R}^{3\times 3}$, is defined by taking the divergence of the gradient:

$$\mathbf{S}^\tau = \sqrt{\det(g_\tau)} \cdot \mathbf{G}^t \circ (g^\tau)^{-1} \circ \mathbf{G} = \frac{1}{\sqrt{\det(g_\tau)}} \cdot \mathbf{G}^t \circ \begin{pmatrix} g^\tau_{11} & -g^\tau_{10} \\ -g^\tau_{01} & g^\tau_{00} \end{pmatrix} \circ \mathbf{G}.$$

In the case that g^τ is defined by the 3D embedding, this gives the standard cotangent-Laplacian [Dziuk 1988; Pinkall and Polthier 1993].

References

AGARWALA, A., DONTCHEVA, M., AGRAWALA, M., DRUCKER, S., COLBURN, A., CURLESS, B., SALESIN, D., AND COHEN, M. 2004. Interactive digital photomontage. *ACM Transactions on Graphics (SIGGRAPH '04)*, 294–302. URL: http://doi.acm.org/10.1145/1015706.1015718. 44

AGARWALA, A. 2007. Efficient gradient-domain compositing using quadtrees. *ACM Transactions on Graphics (SIGGRAPH '07)*, 94:1–94:5. URL: http://doi.acm.org/10.1145/1275808.1276495. 44

ANDREUX, M., RODOLA, E., AUBRY, M., AND CREMERS, D. 2014. Anisotropic Laplace-Beltrami operators for shape analysis. In *Proceedings of the Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment*. URL: http://dx.doi.org/10.1007/978-3-319-16220-1_21. 45, 46

BAJAJ, C. L., AND XU, G. 2003. Anisotropic diffusion of surfaces and functions on surfaces. *ACM Transactions on Graphics 22*, 4–32. URL: http://doi.acm.org/10.1145/588272.588276. 45, 46

BHAT, P., CURLESS, B., COHEN, M., AND ZITNICK, L. 2008. Fourier analysis of the 2D screened Poisson equation for gradient domain problems. In *European Conference on Computer Vision*, Springer, Berlin, 114–128. URL: http://dx.doi.org/10.1007/978-3-540-88688-4_9. 44, 45, 46

CHEN, Y., DAVIS, T., HAGER, W., AND RAJAMANICKAM, S. 2008. Algorithm 887: Cholmod, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software 35*, 3, 22:1–22:14. URL: http://doi.acm.org/10.1145/1391989.1391995. 45, 47

CHUANG, M., AND KAZHDAN, M. 2011. Interactive and anisotropic geometry processing using the screened poisson equation. *ACM Transactions on Graphics 30*, 57:1–57:10. URL: http://doi.acm.org/10.1145/1964921.1964952. 45, 46

CLARENZ, U., DIEWALD, U., AND RUMPF, M. 2000. Anisotropic geometric diffusion in surface processing. In *Proceedings of the Conference on Visualization '00*, IEEE Computer Society Press, Los Alamitos, CA, 397–405. URL: http://dl.acm.org/citation.cfm?id=375213.375276. 45, 46

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, ACM Press/Addison-Wesley Publishing Co., New York, 317–324. URL: http://dx.doi.org/10.1145/311535.311576. 45, 46

DZIUK, G. 1988. Finite elements for the Beltrami operator on arbitrary surfaces. In *Partial Differential Equations and Calculus of Variations, Lecture Notes in Mathematics*, vol. 1357. Springer, Berlin, 142–155. URL: http://dx.doi.org/10.1007/BFb0082865. 45, 53

FATTAL, R., LISCHINKSI, D., AND WERMAN, M. 2002. Gradient domain high dynamic range compression. *ACM Transactions on Graphics (SIGGRAPH '02)*, 249–256. URL: http://doi.acm.org/10.1145/566654.566573. 44

GUENNEBAUD, G., JACOB, B., ET AL., 2010. Eigen v3. http://eigen.tuxfamily.org. 45, 47

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2012. Modal shape analysis beyond laplacian. *Computer Aided Geometric Design 29*, 204–218. URL: http://dx.doi.org/10.1016/j.cagd.2012.01.001. 51

KAZHDAN, M. M., LILLANEY, K., RONCAL, W. G., BOCK, D., VOGELSTEIN, J. T., AND BURNS, R. C. 2015. Gradient-domain fusion for color correction in large EM image stacks. *CoRR abs/1506.02079*. URL: http://arxiv.org/abs/1506.02079. 44

KAZHDAN, M., 2016. Shape gradient domain. https://github.com/mkazhdan/ShapeGradientDomain. 50

LEVIN, A., ZOMET, A., PELEG, S., AND WEISS, Y. 2004. Seamless image stitching in the gradient domain. In *European Conference on Computer Vision*, Springer, Berlin, 377–389. URL: http://dx.doi.org/10.1007/978-3-540-24673-2_31. 44

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Transactions on Graphics (SIGGRAPH '03)*, 313–318. URL: http://doi.acm.org/10.1145/882262.882269. 44

PINKALL, U., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 15–36. URL: http://dx.doi.org/10.1080/10586458.1993.10504266. 45, 53

TAUBIN, G. 1995. A signal processing approach to fair surface design. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, SIGGRAPH '95, 351–358. URL: http://doi.acm.org/10.1145/218380.218473. 45, 50

VALLET, B., AND LÉVY, B. 2008. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum 2*, 251–260. URL: http://dx.doi.org/10.1111/j.1467-8659.2008.01122.x. 51

YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics 23*, 644–651. URL: http://doi.acm.org/10.1145/1015706.1015774. 45, 46

Author Contact Information

Ming Chuang
PerceptIn Inc.
4633 Old Ironsides Drive
Santa Clara, CA 95054
ming.chuang@perceptin.io

Szymon Rusinkiewicz
Princeton University
35 Olden Street
Princeton, NJ 08540
smr@princeton.edu

Michael Kazhdan
Johns Hopkins University
Malone 229
3400 North Charles Street
Baltimore, MD 21218
misha@cs.jhu.edu

Ming Chuang, Szymon Rusinkiewicz, Misha Kazhdan, Gradient-Domain Processing of Triangle Meshes, *Journal of Computer Graphics Techniques (JCGT)*, vol. 5, no. 4, 44–55, 2016
http://jcgt.org/published/0005/04/04/

Received:	2016-09-10		
Recommended:	2016-11-14	Corresponding Editor:	Reynold Bailey
Published:	2016-12-22	Editor-in-Chief:	Marc Olano