Supplementary Material: Inter-Surface Maps via Constant-Curvature Metrics

PATRICK SCHMIDT, RWTH Aachen University
MARCEL CAMPEN, Osnabrück University
JANIS BORN, RWTH Aachen University
LEIF KOBBELT, RWTH Aachen University

1 TANGENT VECTOR TRANSPORT

We here provide the details of the transport operator used (in Sec. 6.7) to (1) trace vertex updates on a target surface, (2) maintain a smoothed derivative history, and (3) implement a connection-Laplacian operator as pre-conditioner.

1.1 Transport Operator

Each face of a triangle mesh defines a local barycentric coordinate system for points and tangent vectors in $\mathbb{R}^2$. Given two adjacent faces t_i, t_j, we define the transport operator $\tau_{ij} \in \mathbb{R}^{2\times 2}$ which takes a vector from the barycentric coordinate system of t_i, transports it across the common edge, and represents it in the barycentric coordinate system of t_j.

This transport is via the Levi-Civita connection, i.e. can be computed as follows: Jointly flatten the two triangles t_i and t_j isometrically into $\mathbb{R}^2$. With vertex coordinates in $\mathbb{R}^2$ given by $\mathbf{a}_i$, $\mathbf{b}_i$, $\mathbf{c}_i$, and $\mathbf{a}_j$, $\mathbf{b}_j$, $\mathbf{c}_j$, we have

$$\tau_{ij} = \left[\mathbf{a}_j - \mathbf{c}_j, \mathbf{b}_j - \mathbf{c}_j\right] \left[\mathbf{a}_i - \mathbf{c}_i, \mathbf{b}_i - \mathbf{c}_i\right]^{-1}.$$

Given a dual path on a mesh represented by a sequence of faces $t_1, \dots, t_n$, we compute the transport $\tau_{1,\dots,n}$ along that path by composition $\tau_{1,\dots,n} = \tau_{n-1,n} \cdots \tau_{23} \cdot \tau_{12}$.

1.2 Vertex Update Tracing

For each vertex v_i of $\mathcal{A}$, we are given an update direction $\mathbf{d}_i = (\mathrm{d}\alpha_i, \mathrm{d}\beta_i)^\mathsf{T}$ in barycentric coordinates of some face t of $\mathcal{B}$. To trace this update direction over the surface of $\mathcal{B}$, we again use the Levi-Civita connection to transport the tracing direction across different tangent spaces of $\mathcal{B}$. In practice, we trace in local barycentric coordinates and apply a transformation τ_{jk} to the current tracing direction $(\mathrm{d}\alpha_i, \mathrm{d}\beta_i)^\mathsf{T}$ every time we cross an edge from a face t_j to t_k in $\mathcal{B}$.

This tracing yields for every vertex v_i of $\mathcal{A}$ a path on $\mathcal{B}$. Computing the transport operator $\tau_{\dots}$ along the corresponding dual edge path tells us how to transport a vector along this path. For each vertex v_i, denote this operator by τ_i.

We construct a $\mathbb{R}^{2|\mathcal{V}_\mathcal{A}|\times 2|\mathcal{V}_\mathcal{A}|}$ block diagonal matrix from the transport operators τ_i of all vertices:

$$\mathbf{T} = \begin{pmatrix} \tau_1 & & \\ & \ddots & \\ & & \tau_{|\mathcal{V}_\mathcal{A}|} \end{pmatrix}.$$

Consider $\mathbf{v} \in \mathbb{R}^{2|\mathcal{V}_\mathcal{A}|}$ representing a tangent vector per vertex of $\mathcal{A}$ on $\mathcal{B}$ before tracing. Transporting each tangent vector along its respective path yields a new set of tangent vectors $\mathbf{v}'$ represented in potentially different coordinate systems. $\mathbf{T}$ captures these changes of coordinates and we can compute $\mathbf{v}' = \mathbf{T}\mathbf{v}$.

1.3 Derivative Smoothing

We maintain smoothed versions $\tilde{\mathbf{g}}$ and $\tilde{\mathbf{H}}$ of the gradient and Hessian. At the beginning of each iteration, we compute the current gradient and Hessian and blend them into the smoothed versions. We then compute a Newton step based on the the smoothed gradient and Hessian. If the negative smoothed gradient is no descent direction or the smoothed Newton step yields no descent direction, we reset the smoothed derivatives and re-solve using $\mathbf{g}$ and $\mathbf{H}$.

When applying the computed update step, vertices of mesh $\mathcal{A}$ are moved to new positions on $\mathcal{B}$. Since they potentially move to new faces, their barycentric representation changes, rendering $\tilde{\mathbf{g}}$ and $\tilde{\mathbf{H}}$ incompatible with the current state. To restore compatibility, we update $\tilde{\mathbf{g}}$ and $\tilde{\mathbf{H}}$ with the same transport operator (including a change of coordinate systems). We compute transported versions $\tilde{\mathbf{g}}'$ and $\tilde{\mathbf{H}}'$ by applying $\mathbf{T}$ as defined above:

$$\tilde{\mathbf{g}}' = \mathbf{T}\tilde{\mathbf{g}}$$
$$\tilde{\mathbf{H}}' = \mathbf{T}^{-\mathsf{T}}\tilde{\mathbf{H}}\mathbf{T}^{-1}$$

The update rule for $\tilde{\mathbf{H}}$ is derived as follows: Consider an original vector field $\mathbf{v}$ and its transported result $\mathbf{v}' = \mathbf{T}\mathbf{v}$. We need the transported Hessian to have the same effect on the transported vector as the original Hessian on the original vector, i.e.

$$\mathbf{v}^\mathsf{T}\tilde{\mathbf{H}}\mathbf{v} = \mathbf{v}'^\mathsf{T}\tilde{\mathbf{H}}'\mathbf{v}'$$

which is fulfilled by choosing the update as described above.

After this transport, $\tilde{\mathbf{H}}'$ and $\tilde{\mathbf{g}}'$ are prepared to be blended with the gradient and Hessian at the new position, computed in the beginning of the next iteration. Note that the sparsity pattern of $\tilde{\mathbf{H}}$ never changes.

We keep separate smoothed versions of the gradient and Hessian for both meshes $\mathcal{A}$ and $\mathcal{B}$. The previous explanations describe how to transport the derivatives for mesh $\mathcal{A}$ along the surface of $\mathcal{B}$ while the map is represented in direction $\mathcal{A}$ to $\mathcal{B}$. However, each update of $\phi^\triangleright$ in this direction also implies a change of the induced vertex-to-surface map $\phi^\triangleleft$. Thus, we also carry out the symmetric construction, i.e. transport the derivatives with respect to $\mathcal{B}$ along the paths its vertices travelled on the surface of $\mathcal{A}$.

1.4 Laplacian Pre-Conditioning

Similar to [Schmidt et al. 2019], we make use of a squared Laplacian pre-conditioner, penalizing deviations in update direction for neighboring vertices.

In our case, the update direction is a tangent vector field $\mathbf{d} \in \mathbb{R}^{|2\mathcal{V}_\mathcal{A}|}$ defined at the mapped vertices of $\mathcal{A}$ on $\mathcal{B}$. Each pair of entries in $\mathbf{d}$ describes a tangent vector in a local barycentric coordinate system. We transport a vector from its own tangent space to the one of a neighboring vertex along the path defined by the image of their connecting edge on the target surface. There we switch from a barycentric coordinate system to local orthonormal coordinate system and compute the squared norm of the difference between both vectors. We can combine these operations into a quadratic form $\mathbf{d}^\mathsf{T}\mathbf{P}\mathbf{d}$ with

$$\mathbf{P} = \mathbf{L}^\mathsf{T}\mathbf{S}^\mathsf{T}\mathbf{M}\mathbf{S}\mathbf{L}$$

where

- $\mathbf{L}$ is a connection Laplacian using parallel transport to align the tangent spaces of neighboring vertices. It maps from and to local barycentric coordinates. Details of the construction are given below.
- $\mathbf{S}$ maps from the local barycentric coordinate system at each vertex to an isometric 2D parametrization of the triangle. This map is responsible for converting from barycentric coordinates to tangent vectors that have a comparable length.
- $\mathbf{M}$ is a diagonal "mass matrix" used to integrate the squared lengths of the computed tangent vector. We use area weights of $\mathcal{A}$ (constant throughout the optimization).

For each vertex v_i with a local update direction $\mathbf{d}_i \in \mathbb{R}^2$, the connection Laplacian is computed as

$$\mathbf{L}(\mathbf{d}_i) = \sum_{j \in \mathcal{N}(i)} \omega_{ij} \left(\tau_{ji}\mathbf{d}_j - \mathbf{d}_i\right)$$

where τ_{ji} is the parallel transport operator along the edge path from vertex v_j to vertex v_i on the target mesh. ω_{ij} are cotangent weights of the source mesh (constant throughout the optimization). Still, $\mathbf{L}$ changes in each iteration, i.e. needs to be re-computed, as the transport paths between vertices change.

A similar definition of a connection Laplacian is given in [Kyng et al. 2016]. Applications also appear in [Knöppel et al. 2013, 2015].

2 YAMABE FLOW

To compute an initial constant curvature metric, we make use of Euclidean and hyperbolic discrete Yamabe flow [Bobenko et al. 2015; Zhang et al. 2014]. Given an Euclidean / hyperbolic input metric ℓ, it produces a constant curvature metric ℓ' which is discrete conformal to ℓ. It iteratively updates the discrete conformal factor u_i per vertex, which defines ℓ' via a scaling of the initial metric ℓ. Starting with $\mathbf{u} = \mathbf{0}$ we perform the following steps:

(1) Compute scaled edge lengths from discrete conformal factor:

$$\ell'_{ij} = \begin{cases} e^{\frac{1}{2}(u_i+u_j)}\ell_{ij} & \mathbb{E}^2 \\ 2\sinh^{-1}\left(e^{\frac{1}{2}(u_i+u_j)}\sinh(\frac{\ell_{ij}}{2})\right) & \mathbb{H}^2 \end{cases}$$

(2) Compute angles from edge lengths via law of cosines:

$$\theta_i = \cos^{-1}\begin{cases} \frac{{\ell_j}^2+{\ell_k}^2-{\ell_i}^2}{2\ell_j\ell_k} & \mathbb{E}^2 \\ \frac{\cosh\ell_j\cosh\ell_k-\cosh\ell_i}{\sinh\ell_j\sinh\ell_k} & \mathbb{H}^2 \end{cases}$$

(3) Compute angle defects $\mathbf{K}$ per vertex:

$$K_i = 2\pi - \sum_{ijk \in \mathcal{N}(i)} \theta^i_{jk}$$

(4) Update conformal factor $\mathbf{u}$ (Newton step). The gradient $\mathbf{g}$ and Hessian $\mathbf{H}$ are given by

$$\mathbf{g} = -\mathbf{K}$$

$$\mathbf{H} = \sum_{ij \in \mathcal{E}} \frac{1}{2}\omega_{ij}(\mathrm{d}u_i - \mathrm{d}u_j)^2 + \begin{cases} 0 & \mathbb{E}^2 \\ \frac{1}{2}\tanh\frac{\ell_{ij}}{2}(\mathrm{d}u_i + \mathrm{d}u_j)^2 & \mathbb{H}^2 \end{cases}$$

with

$$\omega_{ij} = \begin{cases} \frac{1}{2}\cot\theta^k_{ij} + \frac{1}{2}\cot\theta^l_{ji} & \mathbb{E}^2 \\ \frac{1}{2}\cot\left(\frac{1}{2}\left(\pi - \theta^i_{jk} - \theta^j_{ki} + \theta^k_{ij}\right)\right) & \mathbb{H}^2 \\ +\frac{1}{2}\cot\left(\frac{1}{2}\left(\pi - \theta^j_{il} - \theta^i_{lj} + \theta^l_{ji}\right)\right) & \end{cases}$$

per edge ij with opposite vertices k.

In the Euclidean case, $\mathbf{H}$ is the discrete cotangent Laplace operator. We solve $\mathbf{H}\mathrm{d}\mathbf{u} = -\mathbf{g}$ and update $\mathbf{u} \rightarrow \mathbf{u} + s \cdot \mathrm{d}\mathbf{u}$ with step size s. In the flat case, $\mathbf{H}$ has rank $n-1$ and the solution is unique only up to a constant offset. Here, we shift $\mathbf{u}$ (i.e. uniformly scale the edge lengths) such that the total surface area remains constant. In the hyperbolic case, $\mathbf{H}$ has full rank.

When used for metric sanitization (Sec. 6.7), we apply a single iteration (with $s = 1$) of the flow. When used to compute an initial metric of the macro-mesh $\mathcal{L}$ (Sec. 7.1), in our experiments we iterate steps (with $s = 10^{-3}$), stopping when the maximum absolute angle defect (thus the deviation from the desired constant curvature state) falls below 10^{-12}. We remark that for success guarantees a strictly convergent optimization strategy (e.g. trust-region) [Springborn et al. 2008] and connectivity modification operators [Luo 2004], [Campen and Zorin 2017] would have to be employed.

REFERENCES

Alexander I Bobenko, Ulrich Pinkall, and Boris A Springborn. 2015. Discrete conformal maps and ideal hyperbolic polyhedra. *Geometry & Topology* 19, 4 (2015), 2155–2215.

Marcel Campen and Denis Zorin. 2017. Similarity Maps and Field-Guided T-Splines: a Perfect Couple. *ACM Trans. Graph.* 36, 4 (2017).

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally optimal direction fields. *ACM Trans. Graph.* 32, 4 (2013), 1–10.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2015. Stripe patterns on surfaces. *ACM Trans. Graph.* 34, 4 (2015), 1–11.

Rasmus Kyng, Yin Tat Lee, Richard Peng, Sushant Sachdeva, and Daniel A Spielman. 2016. Sparsified cholesky and multigrid solvers for connection laplacians. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. 842–850.

Feng Luo. 2004. Combinatorial Yamabe flow on surfaces. *Communications in Contemporary Mathematics* 6, 05 (2004), 765–780.

Patrick Schmidt, Janis Born, Marcel Campen, and Leif Kobbelt. 2019. Distortion-Minimizing Injective Maps Between Surfaces. *ACM Trans. Graph.* 38, 6 (2019).

Boris Springborn, Peter Schröder, and Ulrich Pinkall. 2008. Conformal equivalence of triangle meshes. *ACM Trans. Graph.* 27, 3 (2008).

Min Zhang, Ren Guo, Wei Zeng, Feng Luo, Shing-Tung Yau, and Xianfeng Gu. 2014. The unified discrete surface Ricci flow. *Graphical Models* 76, 5 (2014), 321–339.