The Shape Space of Discrete Orthogonal Geodesic Nets

SIGGRAPH ASIA 2018

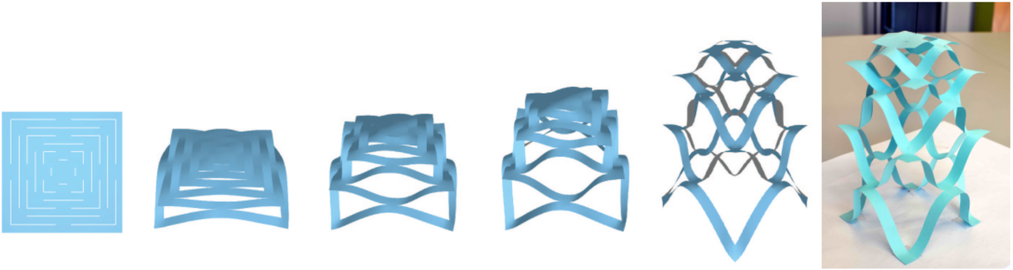
Michael Rabinovich, ETH Zurich, Switzerland
Tim Hoffmann, TU Munich
Olga Sorkine-Hornung, ETH Zurich, Switzerland

Xianzhong Fang

October 15, 2018

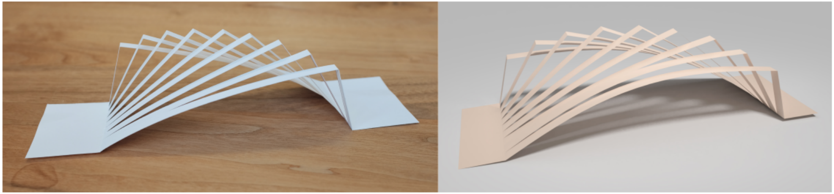
Contents

Developable surfaces



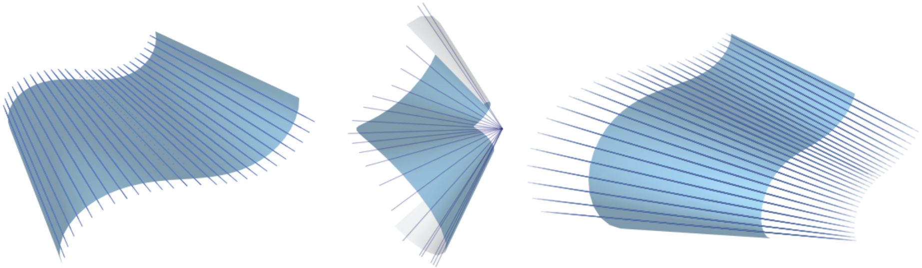
Isometric deformation: no ***stretch*** *and* ***tear***.

Developable surfaces



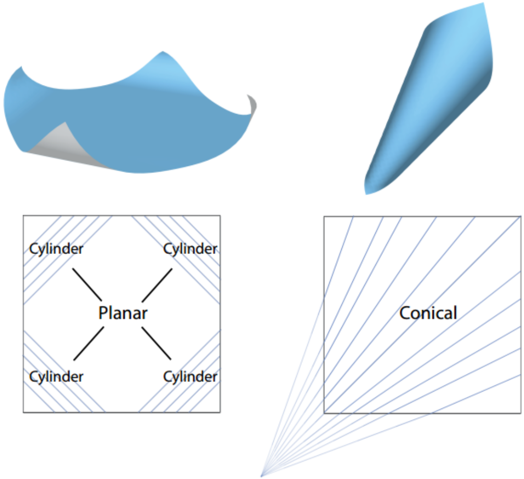
Isometric deformation: no ***stretch*** *and* ***tear****.*

Explicit Representation: Ruling

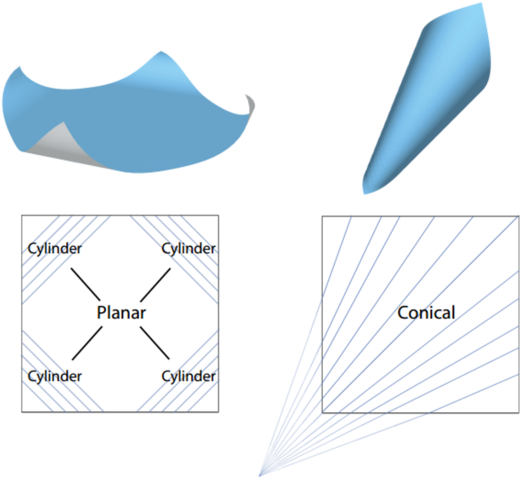


cylindrical surface   conical surface   tangent surface

Explicit Representation: Ruling

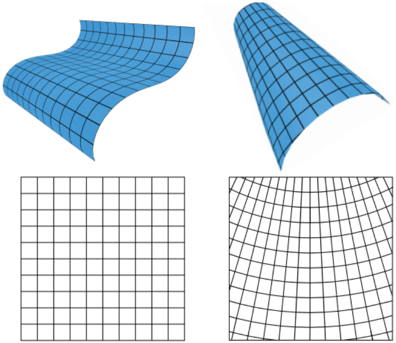


Explicit Representation: Ruling

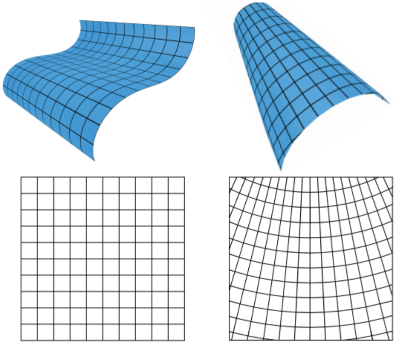


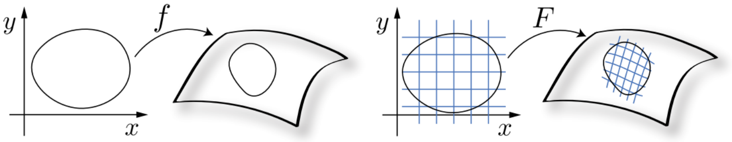
Structure is very different: not easy for editing and modeling

Curvature Line Parameterization



Curvature Line Parameterization

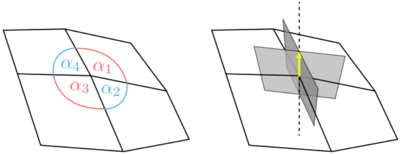




Discrete Orthogonal Geodesic Nets (DOG)

- **Corollary**: *A smooth surface is **developable** if and only if it can be locally parameterized by **orthogonal geodesics**.*

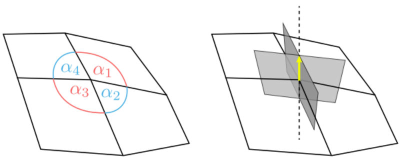
Discrete Orthogonal Geodesic Nets (DOG)



Geodesic: as straight as possible on surface

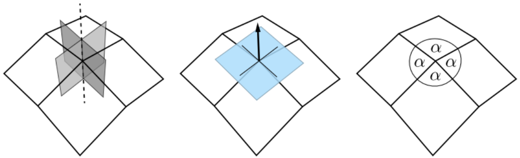
$$\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4,\ \alpha_1 + \alpha_4 = \alpha_2 + \alpha_3$$

Discrete Orthogonal Geodesic Nets (DOG)



Geodesic: as straight as possible on surface

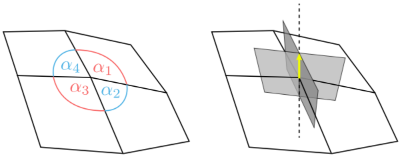
$$\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4, \; \alpha_1 + \alpha_4 = \alpha_2 + \alpha_3$$



Orthogonal Geodesic

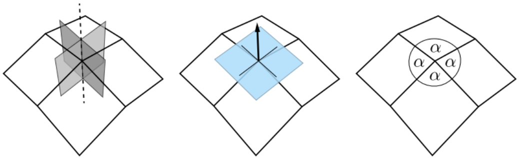
$$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$$

Discrete Orthogonal Geodesic Nets (DOG)



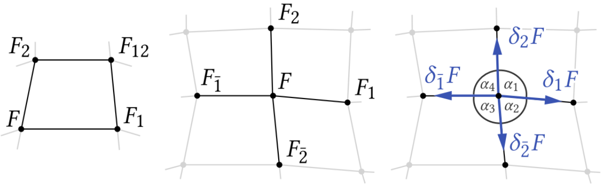
Geodesic: as straight as possible on surface

$$\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4, \; \alpha_1 + \alpha_4 = \alpha_2 + \alpha_3$$

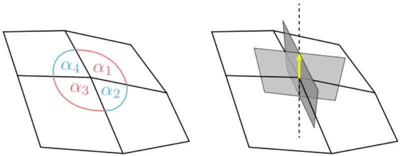


Orthogonal Geodesic

$$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$$

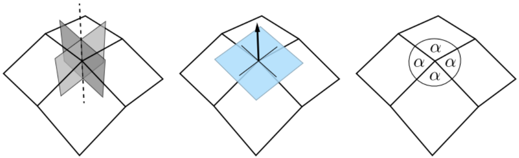


Discrete Orthogonal Geodesic Nets (DOG)



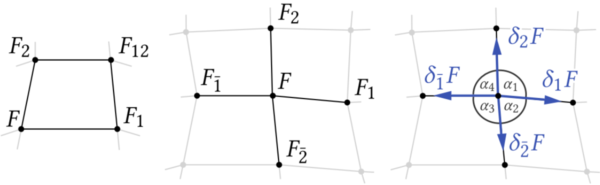
Geodesic: as straight as possible on surface

$$\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4, \ \alpha_1 + \alpha_4 = \alpha_2 + \alpha_3$$



Orthogonal Geodesic

$$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$$



$$\phi_1(\mathbf{F}) := \cos(\alpha_1) - \cos(\alpha_2) = 0,$$
$$\phi_2(\mathbf{F}) := \cos(\alpha_2) - \cos(\alpha_3) = 0,$$
$$\phi_3(\mathbf{F}) := \cos(\alpha_3) - \cos(\alpha_4) = 0.$$

Shape Space: DOG flow

- The Shape Space: $\mathcal{M} = \{\mathbf{F} \in C \,|\, \varphi_i(\mathbf{F}) = 0, \, \forall i = 1, 2, 3..., m\}$.
- DOG Flow: $\mathcal{F}(t) \in \mathcal{M}, \mathcal{F}(0) = F^0$.
- Tangent space of $\mathcal{M}$ ($T\mathcal{M}$): $\frac{\partial \varphi}{\partial \mathbf{F}} = J_{F^0}$, $J_{F^0}x = 0$.

Optimization

- The net: $\mathbf{F}$.
- Objective energy: $E(\mathbf{F})$.

Optimization

- The net: $\mathbf{F}$.
- Objective energy: $E(\mathbf{F})$.
- Gradient: $\nabla_M E(\mathbf{F}) = M^{-1}\nabla\mathbf{F}$.

Optimization

- The net: $\mathbf{F}$.
- Objective energy: $E(\mathbf{F})$.
- Gradient: $\nabla_M E(\mathbf{F}) = M^{-1}\nabla\mathbf{F}$.
- Projection: $\overline{\nabla}_M E(\mathbf{F}) := \text{argmin}_{\mathbf{t}} \|\nabla_M E(\mathbf{F}) - \mathbf{t}\|_M, \ \mathbf{t} \in T\mathcal{M}$

Optimization

- The net: $\mathbf{F}$.
- Objective energy: $E(\mathbf{F})$.
- Gradient: $\nabla_M E(\mathbf{F}) = M^{-1}\nabla\mathbf{F}$.
- Projection: $\overline{\nabla}_M E(\mathbf{F}) := \operatorname{argmin}_{\mathbf{t}} \|\nabla_M E(\mathbf{F}) - \mathbf{t}\|_M,\ \mathbf{t} \in T\mathcal{M}$
- KKT system:

$$K \begin{pmatrix} \overline{\nabla}_M E(\mathbf{F}) \\ \lambda \end{pmatrix} = \mathbf{b}$$

$$K = \begin{pmatrix} M & J^\top \\ J & 0 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} \nabla E(\mathbf{F}) \\ \mathbf{0} \end{pmatrix}$$

Optimization

- Bending minimization:

$$E_H(\mathbf{F}) = \sum_{v_i \in \mathbf{F}} A_i H_i^2$$

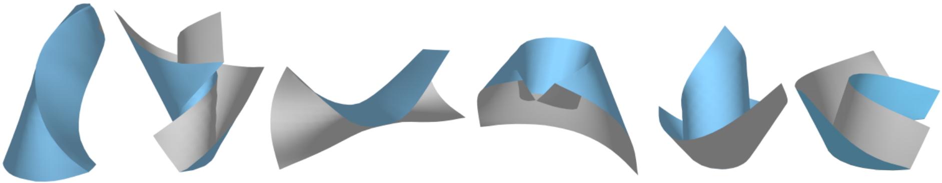
- Isometry term:

$$E_{iso}(\mathbf{F}) = \sum_{e \in \mathbf{F}} (\ell_e - \ell_e^0)^2$$

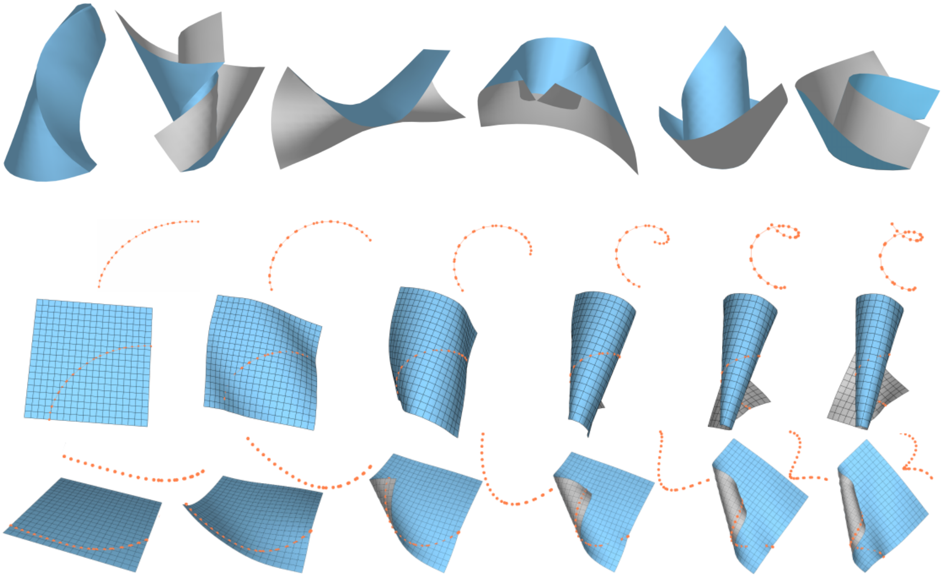
- Positional constraints:

$$\sum_{i \in C} \|F(i) - C(i)\|^2$$

Results



Results



Limitations

- Timecost is high: restricted to rather coarse models for interactive editing.
- No support for intricate crease patterns: here each crease curve is simple and starts and ends at a mesh boundary.

Thanks!