Curve Folded Surfaces Based On Coupled Prisms

ZEJIAN WANG and QINGYOU ZHAO, University of Southern California

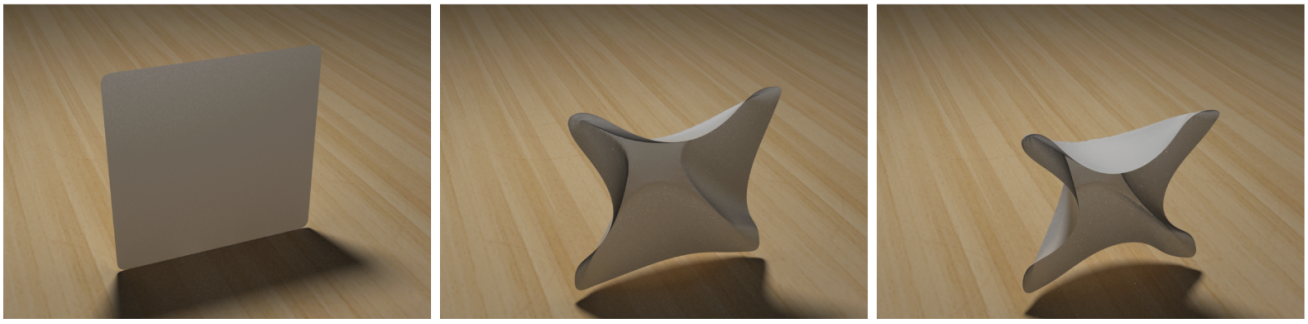


Fig. 1. Paper folding sequence with paperBSDF

Curve folded surfaces hold its special position in the field of computational origami, which is widely used in architecture and art design. We demonstrate a curve folding technique[Kilian et al. 2017] based on coupled prisms structure[Botsch et al. 2006]. By combining the original prism's face energy with energy between prism's edge, we penalize stretching more than bending and minimize this non-linear energy to simulate the curve folding sequence as shown in Figure 1. We show both the original prisms based deformation and the curve folding technique on several examples, and rendered the paper folding example with paper BSDF[Papas et al. 2014].

Additional Key Words and Phrases: Shape Deformation, Folding Motion, Realistic Paper Shading

1 INTRODUCTION

PriMo[Botsch et al. 2006] is a robust method to achieve surface deformation by emulating physically plausible surface behavior. Each polygonal face of the surface mesh is embedded in a volumetric prism, which are coupled through non-linear, elastic forces. Each polygonal face has the same transformation as its prism structure during deformation, and only rigid transformation(rotation and translation) is allowed for each face(prism) in this method. Given rigid transformation of parts of the prisms as hard constraints by a user, other prisms' rigid transformation are solved by minimize the non-linear energy among prisms. The rigidity of the prisms prevents extreme degenerations, making this method numerical stable and robust. We implement the PriMo method based on OpenMesh[Botsch et al. 2002], a generic and efficient polygon mesh data structure with half-edge implementation.

For curve folding, surface bending is more physically plausible than stretching. We treat the energy between prisms' faces as a regularization energy term, and add a new term between prisms' edges as the main energy term to penalize stretching more than bending. The hard constraint is the folding(dihedral) angle between the polygonal faces, which share a same crease edge segment. We firstly triangulate a 2D planar mesh based on constrained Delaunay triangulation[Shewchuk 1996][Jacobson et al. 2017] with given crease pattern, and increase folding angle 1° at each step to generate the paper folding examples. To render realistic paper appearance, we implement the paper shader based on the state-of-the-art analytical paper BSDF model[Papas et al. 2014], which converts a multi-layer subsurface scattering model(BSSRDF)[Donner and Jensen 2005] into a BSDF and retains physically-based absorption and scattering parameters obtained from the measurements.

2 METHODOLOGY

We demonstrate PriMo method with triangle mesh for simplicity, though this "prism" concept could be extended to any general polygonal mesh. After demonstrating the general deformation model, we show how to adopt PriMo method to do curve folding deformation by modifying the energy term. Specifically, the implementation details are given afterwards including mesh triangulation from crease pattern, and how to generate folding sequence. Finally, we demonstrate the rendering technique that we use to synthesize paper-like folding sequence as shown in Figure 1.

2.1 PriMo

In this section, we define the prism structure the non-linear energy on triangle mesh, followed by the numerical method and algorithm to solve the rigid transformation for each prism while minimizing this energy.

2.1.1 Prism structure. We denote the triangle mesh as the set of triangle primitives:

$$\{\Delta_i | i \in \{1, 2, ..., N_f\}\},$$

where i denotes the index of triangle faces starting from 1, and N_f is the number of triangle faces in the triangle mesh. Therefore, Δ_i denotes the ith triangle.

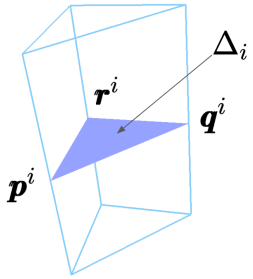


Fig. 2. Prism structure for single triangle face

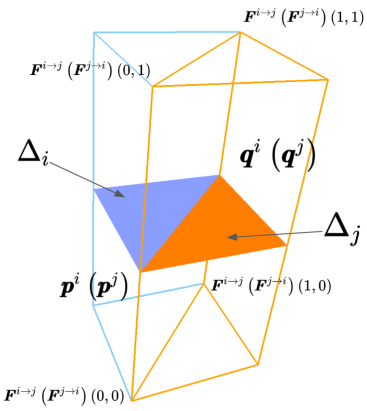


Fig. 3. Prism structure for neighboring triangle faces

In the following sections, any triangle related properties, including vertices, normals, edges, etc. that carry a superscript i refer to properties of the triangle Δ_i

For each triangle Δ_i, we define a prism structure P^i as shown in Figure 2, where $\boldsymbol{p}^i$,$\boldsymbol{q}^i$,$\boldsymbol{r}^i$ are the three vertices of Δ_i. The model is called PriMo just because the structure looks like prism, however the elastic joint energy only exists between any pair of prism P^i and P^j, thus there is no energy related to each top and bottom prism face.

In the undeformed state, a pair of prisms P^i and P^j share a common face, but after a rigid transformation the side faces might not coincide. The face of P^i neighboring P^j is a planar bi-linear patch $\boldsymbol{F}^{i\to j}(u,v)$, $(u,v)\in[0,1]^2$, which interpolates its four corner vertices $\{\boldsymbol{F}^{i\to j}(0,0), \boldsymbol{F}^{i\to j}(0,1), \boldsymbol{F}^{i\to j}(1,0), \boldsymbol{F}^{i\to j}(1,1)\}$. Similarly, we denote the opposite face by $\boldsymbol{F}^{j\to j}(u,v)\subset P_j$ as shown in Figure 3. We define the energy between P^i and P^j as

$$E^{ij} := \int_{[0,1]} \left\| \boldsymbol{F}^{i\to j}(u,v) - \boldsymbol{F}^{j\to i}(u,v) \right\|^2 dudv, \tag{1}$$

which is the integration of squared euclidean distance between all pairs of points on $\boldsymbol{F}^{i\to j}(u,v)$ and $\boldsymbol{F}^{j\to i}(u,v)$. The deformation energy of the whole mesh can now be defined as an weighted-sum of pairwise energies E^{ij}

$$E := \sum_{\{i,j\}} w^{ij} E^{ij}, \quad w^{ij} := \frac{\|\boldsymbol{e}^{ij}\|^2}{|\Delta_i| + |\Delta_j|}, \tag{2}$$

where $|\Delta_k|$ denotes the area of triangle face k before deformation, and $\|\boldsymbol{e}^{ij}\| := \|(\boldsymbol{p}^i, \boldsymbol{q}^i)\| = \|(\boldsymbol{p}^j, \boldsymbol{q}^j)\|$ is the length of the shared edge of Δ_i and Δ_j also before deformation. We initialize w^{ij} before mesh deformation and maintain it unchanged during deformation. Besides, we initialize the four corner of $\boldsymbol{F}^{i\to j}(u,v)$ as following:

$$\begin{aligned}
\boldsymbol{F}^{i\to j}(0,0) &= \boldsymbol{p}^i - h\boldsymbol{n}^{\boldsymbol{p}^i},\\
\boldsymbol{F}^{i\to j}(0,1) &= \boldsymbol{p}^i + h\boldsymbol{n}^{\boldsymbol{p}^i},\\
\boldsymbol{F}^{i\to j}(1,0) &= \boldsymbol{q}^i - h\boldsymbol{n}^{\boldsymbol{q}^i},\\
\boldsymbol{F}^{i\to j}(1,1) &= \boldsymbol{q}^i + h\boldsymbol{n}^{\boldsymbol{q}^i},
\end{aligned} \tag{3}$$

where $\boldsymbol{n}^{\boldsymbol{v}}$ is the vertex normal of vertex v, and h is defined as the prism height. Analogously, we initialize all $\boldsymbol{F}^{i\to j}(u,v)$ based on equation (3). Notice that due to $\boldsymbol{p}^i = \boldsymbol{p}^j$, $\boldsymbol{q}^i = \boldsymbol{q}^j$, $\boldsymbol{n}^{\boldsymbol{p}^i} = \boldsymbol{n}^{\boldsymbol{p}^j}$, and $\boldsymbol{n}^{\boldsymbol{q}^i} = \boldsymbol{n}^{\boldsymbol{q}^j}$, thus the initial(undeformed) state of prisms is the unique global minimum of the energy E. Any bending, shearing, twisting or stretching increases it.

2.1.2 Numerical Solution. The user controls the surface deformation by manually setting the position and/or orientation of certain prisms(faces), and then the optimization step finds the optimal rotation $\boldsymbol{R}^i$, translation $\boldsymbol{t}^i$ for each unconstrained prism(face), such that the E defined by equation(2) is minimized:

$$\underset{\{\boldsymbol{R}^i,\boldsymbol{t}^i\}}{\operatorname{argmin}} \sum_{\{i,j\}} w^{ij} \int_{[0,1]} \|\boldsymbol{R}^i \boldsymbol{F}^{i\to j}(u,v) + \boldsymbol{t}^i - \boldsymbol{R}^j \boldsymbol{F}^{j\to i}(u,v) - \boldsymbol{t}^j\|^2 dudv. \tag{4}$$

We solve minimization defined by equation(4) using a generalized global shape matching approach of [Pottmann et al. 2002].

Pottmann et al. propose an iterative Newton-type simultaneous registration method for points by solving sparse linear system. In our case, during each iteration, a sparse linear system is solved for a descent direction, which corresponds to an affine transformation $\boldsymbol{A}^i$ for each unconstrained prism. And then, a projection of $\boldsymbol{A}^i$ to the manifold of rigid transformation, containing a rotation transform $\boldsymbol{R}^i$ and translation $\boldsymbol{t}^i$, is achieved by solving a pairwise shape matching problem.

2.1.3 Global shape matching. The descent direction of the Newton-type iteration approximate the rigid motions $(\boldsymbol{R}^i, \boldsymbol{t}^i)$, which can be formulated by angular velocity $\boldsymbol{\omega}^i$ and linear velocity $\boldsymbol{v}^i$:

$$\boldsymbol{R}^i(\boldsymbol{p}) + \boldsymbol{t}^i \approx \boldsymbol{p} + \boldsymbol{\omega}^i \times \boldsymbol{p} + \boldsymbol{v}^i =: \boldsymbol{A}^i(\boldsymbol{p}), \tag{5}$$

where p is an arbitrary point. Using the first-order approximation defined in equation (5), we reformulate equation (4) as:

$$\begin{aligned}
\underset{\{\boldsymbol{\omega}^i,\boldsymbol{v}^i\}}{\operatorname{argmin}} \sum_{\{i,j\}} w^{ij} \int_{[0,1]} \|&\boldsymbol{F}^{i\to j}(u,v) + \boldsymbol{\omega}^i \times \boldsymbol{F}^{i\to j}(u,v) + \boldsymbol{v}^i\\
&- \boldsymbol{F}^{j\to i}(u,v) - \boldsymbol{\omega}^j \times \boldsymbol{F}^{j\to i}(u,v) - \boldsymbol{v}^j\|^2 dudv.
\end{aligned} \tag{6}$$

To solve the minimization problem defined by equation(6), we need to solve the point-alignment problem in [Pottmann et al. 2002]

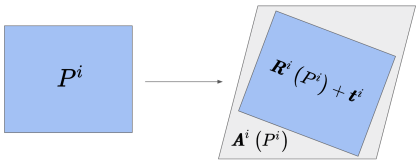


Fig. 4. Project affine transformation to rigid transformation

at first. Assume two corresponding points $\boldsymbol{p}^{i\to j}$ and $\boldsymbol{p}^{j\to i}$ sampled from neighboring faces $\boldsymbol{F}^{i\to j}(u,v)$ and $\boldsymbol{F}^{j\to i}(u,v)$. The global energy E could be represented as:

$$E=\sum_{\{i,j\}} w^{ij}\int_{[0,1]} \|\boldsymbol{p}^{i\to j}+\boldsymbol{\omega}^i\times\boldsymbol{p}^{i\to j}+\boldsymbol{v}^i - \boldsymbol{p}^{j\to i}-\boldsymbol{\omega}^j\times\boldsymbol{p}^{j\to i}-\boldsymbol{v}^j\|^2 dudv. \quad (7)$$

As demonstrated in [Pottmann et al. 2002], collecting all unknowns in the vector $\boldsymbol{c}=\{\boldsymbol{\omega}^1,\boldsymbol{v}^1,\boldsymbol{\omega}^2,\boldsymbol{v}^2,...,\boldsymbol{\omega}^{N_f},\boldsymbol{v}^{N_f}\}$, we may write equation(7) in matrix form:

$$E=\boldsymbol{c}^T\boldsymbol{B}\boldsymbol{c}+2\boldsymbol{A}\boldsymbol{c}+\sum_{\{i,j\}} w^{ij}\left(\boldsymbol{p}^{i\to j}-\boldsymbol{p}^{j\to i}\right), \quad (8)$$

where $\boldsymbol{B}$ is a $6N\times 6N$ symmetric positive definite matrix and A is a $6N\times 1$ vector. Hence, we let $\frac{\partial E}{\partial \boldsymbol{c}}=0$ to get the local minimum for each Newton-type iteration. This leads to:

$$\frac{\partial E}{\partial \boldsymbol{c}}=\boldsymbol{B}\boldsymbol{c}+\boldsymbol{A}^T=\boldsymbol{0}. \quad (9)$$

Notice that equation(9) formalize a sparse linear system, which could be solved efficiently by a linear solver using sparse Cholesky factorization[Guennebaud et al. 2010].

We replace all the component-wise products of the form

$$\left(\boldsymbol{p}^{i\to j}\right)_x\left(\boldsymbol{p}^{j\to i}\right)_y \quad (10)$$

by

$$\left\langle\left(\boldsymbol{F}^{i\to j}\right)_x,\left(\boldsymbol{F}^{j\to i}\right)_y\right\rangle_2, \quad (11)$$

which is defined in the following section.

2.1.4 Pairwise shape matching. We implemented the pair wise shape matching proposed by [Botsch et al. 2006] that solves for the best rigid transformation which registers one set of faces to the other set of faces. The implementation is a generalization of the pairwise point set registration proposed by [Horn 1987]. This shape matching is used in both local and global shape matching in the original Primo implementation. In our implementation, given the relatively small number of triangles in the crease pattern mesh, the global shape matching algorithm converges faster than local shape matching.

The results of the global energy minimization $\boldsymbol{A}^i$ need to be projected back on to the manifold of rigid motions $(\boldsymbol{R}^i,\boldsymbol{t}^i)$ before being applied on the prisms to keep the deformed mesh as rigid as possibleP^i [Botsch et al. 2006]. To retrieve this step, [Botsch et al. 2006] propose to project A^i by finding the closest rigid motion $(\boldsymbol{R}^i,\boldsymbol{t}^i)$ that matches the affine transformed prism P^i instead of [Pottmann et al. 2002]'s proposed method of choosing$(\boldsymbol{R}^i,\boldsymbol{t}^i)$ as the helical motion associated with $(\boldsymbol{\omega}^i,\boldsymbol{v}^i)$ for faster convergence on large deformations. This projection, as shown in Figure 4 yields a shape matching problem that minimizes the energy between the affine transformed face $\boldsymbol{A}^i\left(\boldsymbol{F}^{i\to j}(u,v)\right)$ and the rigidly transformed face $\boldsymbol{R}^i\boldsymbol{F}^{i\to j}(u,v)+\boldsymbol{t}^i$:

$$\underset{\{\boldsymbol{R}^i,\boldsymbol{t}^i\}}{\operatorname{argmin}}\int_{[0,1]}||\boldsymbol{R}^i\boldsymbol{F}^{i\to j}(u,v)+\boldsymbol{t}^i-\boldsymbol{A}^i\left(\boldsymbol{F}^{i\to j}(u,v)\right)||^2 dudv. \quad (12)$$

Details of the shape matching algorithm will be explained as follow. Suppose we have two functions defined with bi-linear interpolation $a(u,v)$ and $b(u,v)$, their L_2 inner product can be simplified to the weighted sum of 16 combinations of corner values:

$$\int_{[0,1]} a(u,v)\cdot b(u,v)dudv=\frac{1}{9}\sum_{i,j,k,l=0}^{1} a_{ij}\cdot b_{kl}2^{(-|i-k|-|j-l|)} =:\langle a,b\rangle_2. \quad (13)$$

With equation(13), the continuous energy of equation(12) evaluates to

$$\left\langle\left(\boldsymbol{F}^{i\to j}-\boldsymbol{A}^i\left(\boldsymbol{F}^{i\to j}\right)\right),\left(\boldsymbol{F}^{i\to j}\right)-\boldsymbol{A}^i\left(\boldsymbol{F}^{i\to j}\right)\right\rangle_2, \quad (14)$$

where $F^{i\to j}$ and $\boldsymbol{A}^i\left(\boldsymbol{F}^{i\to j}\right)$ are the bi-linear interpolation function of the two faces to be matched.

We firstly calculate the weighted centroids $\boldsymbol{c}^i$ and $\boldsymbol{c}^*$ of the two face sets to be matched.

$$c^i=\frac{1}{\sum_{j\in N_i}}\sum_{j\in N_i}\frac{w^{ij}}{4}\sum_{k,l=0}^{1}\boldsymbol{F}^{i\to j}(k,l), \quad (15)$$

$$c^*=\frac{1}{\sum_{j\in N_i}}\sum_{j\in N_i}\frac{w^{ij}}{4}\sum_{k,l=0}^{1}\boldsymbol{A}^i\left(\boldsymbol{F}^{i\to j}(k,l)\right), \quad (16)$$

We then derive the optimal rotation according to [Horn 1987] $\boldsymbol{R}_i$ by first building the matrix N =

$$\begin{pmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx}\\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz}\\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & -S_{xx}+S_{yy}-S_{zz} & S_{yz}+S_{zy}\\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & -S_{xx}-S_{yy}+S_{zz}\end{pmatrix}$$

from the component-wise L_2 inner products.

$$S_{xx}=\sum_{j\in N_i} w^{ij}\left\langle\left(\boldsymbol{F}^{i\to j}-c^i\right)_x,\left(\boldsymbol{F}^{j\to i}-c^*\right)_x\right\rangle_2, \quad (17)$$

$$S_{xy}=\sum_{j\in N_i} w^{ij}\left\langle\left(\boldsymbol{F}^{i\to j}-c^i\right)_x,\left(\boldsymbol{F}^{j\to i}-c^*\right)_y\right\rangle_2, \quad (18)$$

The eigenvector Q corresponding to the largest eigen value of N is the optimal rotation $\boldsymbol{R}^i$ in the form of a unit quaternion. With the rotation, the optimal translation is derived by $\boldsymbol{t}^i=\boldsymbol{c}^*-\boldsymbol{R}^i\boldsymbol{c}^i$.

2.1.5 Boundary condition. Suppose that the Prism P^j in equation(4) is constrained by user input, so P^j should be remain identical before and after optimization. Therefore, $\boldsymbol{\omega}^j=0$ and $\boldsymbol{v}^j=0$ in the first-order approximation equations.

Algorithm 1: Newton-type Global Shape Matching

$\lambda = 1$;
repeat
  Find optimal velocities $\left[\boldsymbol{\omega}^i, \boldsymbol{v}^i\right]$;
  for P^i *in all unconstrained prisms* **do**
    $\left[\boldsymbol{R}^i, \boldsymbol{t}^i\right] = project\left(\lambda\boldsymbol{\omega}^i, \lambda\boldsymbol{v}^i\right)$;
    $P^i = \boldsymbol{R}^i\left(P^i\right) + \boldsymbol{t}^i$;
  end
  $\lambda = \lambda * 0.5$
until *converged*;

2.1.6 Newton-type algorithm. Following [Botsch et al. 2006], we scale each Newton-like descent direction by a step size λ, and halve it after each step. The whole Newton-type non-linear optimization algorithm is demonstrated in Algorithm 1.

2.2 Curve Folding

In this section, we demonstrate how to generate curve folding sequence based on our coupled prisms model with modified energy term. Also, we show how to triangulate a 2D planar mesh given a crease pattern, and fold the triangle faces along the crease edge segments step by step as hard constraints for our optimization algorithm.

2.2.1 Deformation model. Following the deformation model as described in [Kilian et al. 2017], we penalize the stretching more than bending between prisms. Also, we fold all the pairs of faces along the crease curve locally, and then treat each pair of faces as a whole rigid unit during optimization. All of our folding results shown in the next section are generated in this way.

2.2.2 Computing a folding sequence. Let $e^i = \left(\boldsymbol{p}^i, \boldsymbol{q}^i\right) \in \Delta_i$ be oriented such that $\boldsymbol{x}^i$ be the unit vector parallel to $\boldsymbol{q}^i - \boldsymbol{p}^i$. $\boldsymbol{n}^i$ is the normal of Δ_i. Thus,

$$\boldsymbol{n}^i(\theta) = \cos(\theta)\,\boldsymbol{n}^i - \sin(\theta)\left(\boldsymbol{n}^i \times \boldsymbol{x}^i\right), \tag{19}$$

where θ is the folding angle. In all our folding example, we start from $\theta = 0$ and increase it by 1° per step.

2.2.3 Meshing. The underlying representation of the deforming creased sheet is a triangle mesh. The crease pattern is defined by a set of line segments and curves defined in 2D space. We first initialize a planar polygonal mesh with sampled segments on the crease curves and boundaries as polygon edges. Then according to [Kilian et al. 2017] we triangulate the planar mesh by computing a constrained Delaunay triangulation [Shewchuk 1996] on the crease pattern mesh, as shown in Figure 5. The level of triangulation is calculated to balance between crease curve fidelity and numerical error due to small triangles.

2.3 Rendering

To achieve the realistic appearance of paper, we implement the paper BSDF[Papas et al. 2014]. We implement this paper BSDF in our own renderer, and render our folding sequence using path tracing. Figure 6 shows a paper rendered with back-lit light source and textured depth.

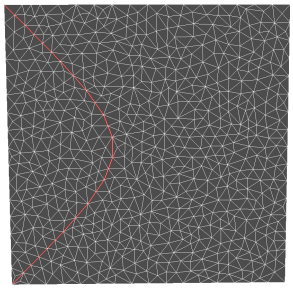


Fig. 5. Constrained Delaunay triangulation

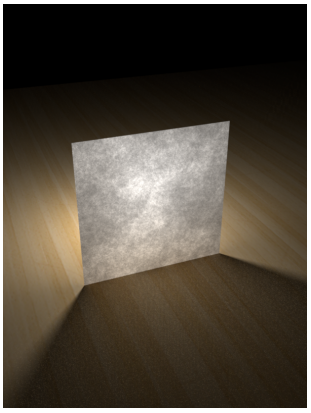


Fig. 6. Paper BSDF with textured depth

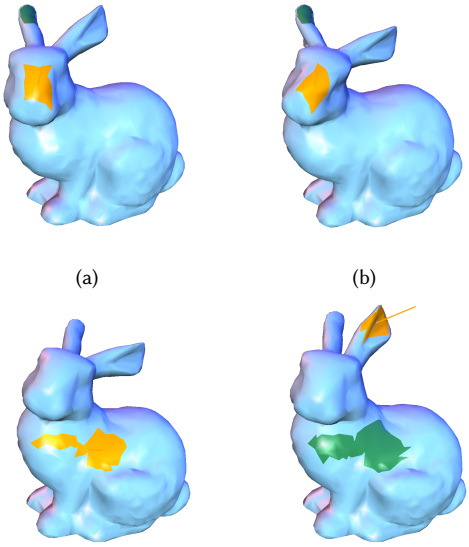
3 RESULT

We implement our PriMo model and paper folding model based on OpenMesh[Botsch et al. 2002]. In Figure 7, we show a typical interactive deformation editing demo. In order to show the robustness of PriMo model, we randomly rotate all the prisms and translate them to the same position of world space, and then try to recover the original shape of the mesh. As shown in Figure 8, the shape could be recovered successfully after 20 iterations.

For curve folding, we generate 4 folding sequences with 4 different crease pattern. Figure 9-12 shows the result when folding angle $\theta = 0^\circ, 30^\circ, 60^\circ, and 90^\circ$

REFERENCES

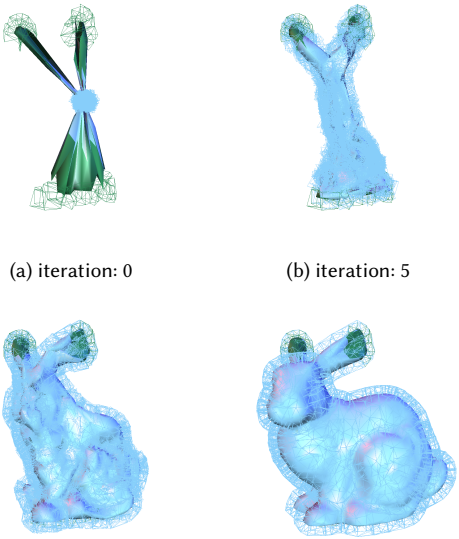
Mario Botsch, Mark Pauly, Markus H Gross, and Leif Kobbelt. 2006. PriMo: coupled prisms for intuitive surface modeling. In *Symposium on Geometry Processing*. 11–20.
M. Botsch, S. Steinberg, S. Bischoff, and L. Kobbelt. 2002. OpenMesh - a generic and efficient polygon mesh data structure.
Craig Donner and Henrik Wann Jensen. 2005. Light diffusion in multi-layered translucent materials. In *ACM Transactions on Graphics (ToG)*, Vol. 24. ACM, 1032–1039.
Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.
Berthold K. P. Horn. 1987. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A* 4, 4 (Apr 1987), 629–642. https://doi.org/10.1364/JOSAA.4.000629
Alec Jacobson, Daniele Panozzo, et al. 2017. libigl: A simple C++ geometry processing library. http://libigl.github.io/libigl/.
Martin Kilian, Aron Monszpart, and Niloy J. Mitra. 2017. String Actuated Curved Folded Surfaces. *ACM Transactions on Graphics* 36, 3, Article 25 (May 2017), 13 pages. https://doi.org/10.1145/3015460



(a) (b)

(c) (d)

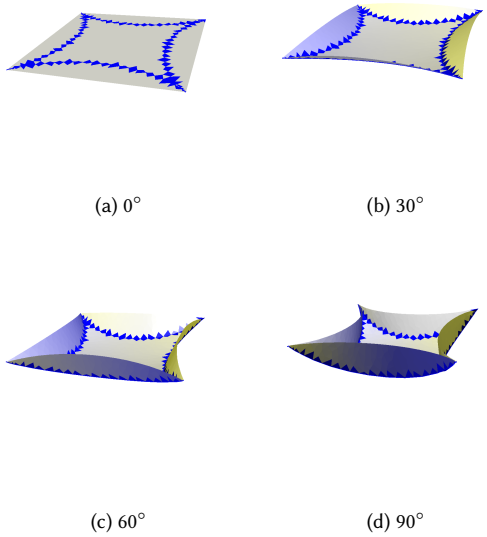
Fig. 7. Interactive deformation editing



(a) iteration: 0 (b) iteration: 5

(c) iteration: 10 (d) iteration: 20

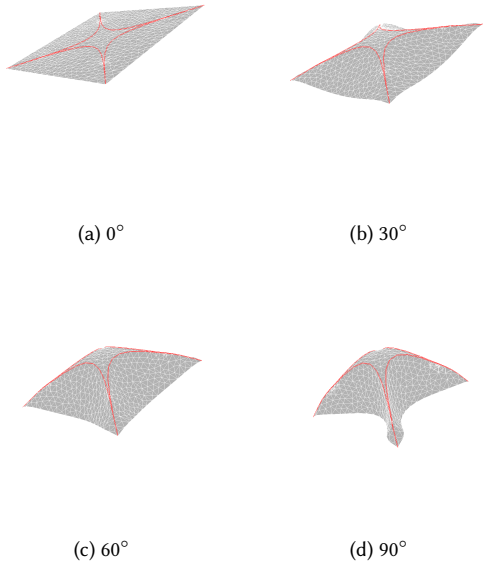
Fig. 8. PriMo Robustness Demo



(a) 0° (b) 30°

(c) 60° (d) 90°

Fig. 9. Curve folding demo 1



(a) 0° (b) 30°

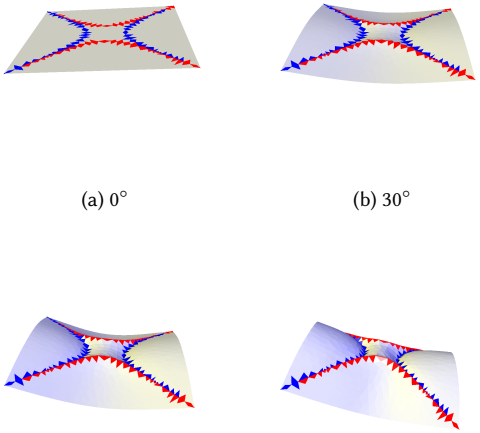
(c) 60° (d) 90°

Fig. 10. Curve folding demo 2

Marios Papas, Krystle de Mesa, and Henrik Wann Jensen. 2014. A Physically-Based BSDF for Modeling the Appearance of Paper. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 133–142.

Helmut Pottmann, Stefan Leopoldseder, and Michael Hofer. 2002. Simultaneous registration of multiple views of a 3D object. *INTERNATIONAL ARCHIVES OF PHOTOGRAMMETRY REMOTE SENSING AND SPATIAL INFORMATION SCIENCES* 34, 3/A (2002), 265–270.

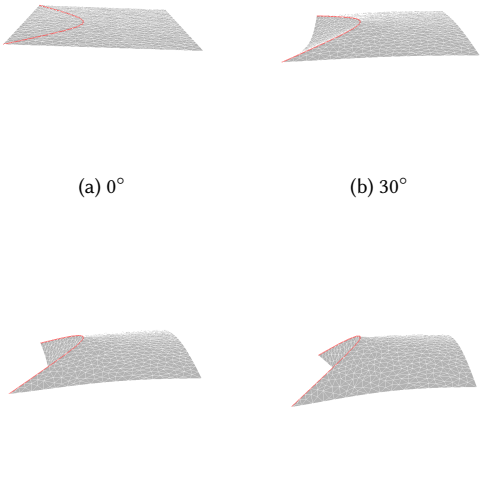
Jonathan Richard Shewchuk. 1996. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Applied computational geometry towards geometric engineering*. Springer, 203–222.



(a) 0° (b) 30°

(c) 60° (d) 90°

Fig. 11. Curve folding demo 3



(a) 0° (b) 30°

(c) 60° (d) 90°

Fig. 12. Curve folding demo 4