Summary of Discrete Shells Origami

November 3, 2024

1 Introduction

This paper uses the discrete shell model as a foundation to enable the creation of origami.

The paper mesh can be moved, rotated and folded using a UI. The creases aren't scripted, but executed upon request as the application runs and the user puts in the necessary information.

2 Background

2.1 Cloth Simulation

Cloth is represented as a triangle mesh, and each vertex has a mass associated with it, as well as springs that connect the vertex to nearby vertices. The underlying spring system gives the mesh cloth-like properties.

Classic cloth simulations use Euler integration, but this can cause instability.

Baraff and Witkin use the simple integration scheme

$$\begin{pmatrix} \Delta x \\ \Delta v \end{pmatrix} = h \begin{pmatrix} v_0 + \Delta v \\ M^{-1} f(x_0 + \Delta x, v_0 + \Delta v) \end{pmatrix} \tag{1}$$

but this paper uses the Newmark integration scheme like the DS paper.

This section is mainly a summary of the previous paper so nothing new here.

3 Implementation

3.1 Discrete Shells

The paper implements Discrete Shells as the underlying engine.

3.2 Edge Making

This is rather involved, so I suggest we just keep our folds to predetermined subsets of vertices.

3.3 Reference Angle Modification

The simulation stores a variable phisRef for each face that holds a function of the rest-angle to each neighboring face.

A primitive way to create folds would be to immediately change phisRef to the new desired angle, but this does not work well. It causes unnatural oscillations, which may cause the simulation to blow up.

In order to eliminate the problem, the paper is folded gradually. When a fold is made, the face records the current rest-angle, the desired angle, and the current simulation time.

During each step of the simulation, the phisRef values for each edge on each face are changed linearly to the new rest angle w.r.t. time.

Importantly, the dihedral angle is not directly used for the bending energy. To compensate for the lack of collision detection, the dihedral angle term in the flexural energy is replaced with a function of this angle:

$$W_B(\mathbf{x}) = \sum_e \left(2 \cdot \tan\left(\frac{1}{2}\theta_e\right) - \bar{\theta}_e\right)^2 \frac{\|\bar{e}\|}{\bar{h}_e} \tag{2}$$

3.4 Vertex Constraint

The UI necessitates the constraining of certain vertices while folding is in progress. I'm not sure we need this, but in case we do, this is achieved by assigning the constrained vertices very large mass values. This effectively sets velocity for these vertices to 0 and locks them in place.

3.5 Notes

This summary skips over large parts of the paper, such as edge-making and UI, since we do not want the same interactive functionality.