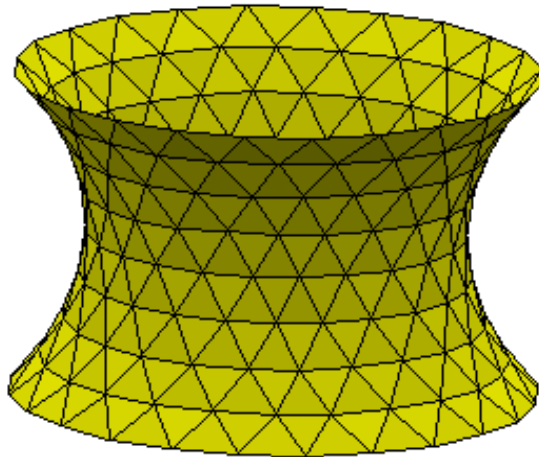


**Module III:**  
**Interfacial Films in three dimensions**

- application notes and description of datafiles

- Problems



## The Surface Evolver 2.10, July 6, 1998

<http://www.susqu.edu/facstaff/b/brakke/evolver/default.htm>

by Kenneth A. Brakke

Mathematics Department, Susquehanna University, Selinsgrove, PA, 17870, [brakke@susqu.edu](mailto:brakke@susqu.edu)

with support from The Geometry Center, University of Minnesota

website documentation index for the curious

Overview

Acquisition and installation

Command line options

Tutorial and examples

Geometric elements and attributes

Surface models

Energies

Constraints

Named quantities and methods

Syntax

Datafile

Surface initialization

Graphics

Command language

Graphics mode commands

Hessian, eigenvalues, and eigenvectors

Error handling

Interrupting execution

Parallel computation

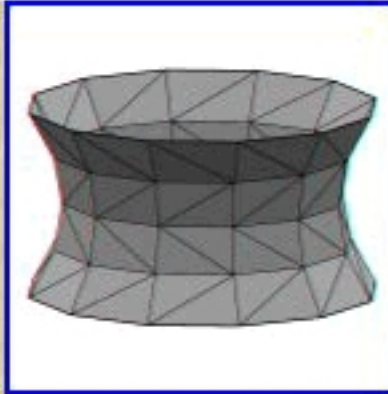
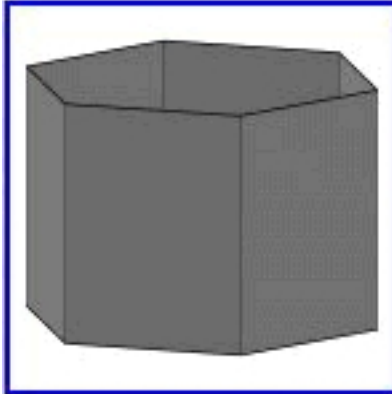
Bugs

Bibliography

Index

Newsletters

### EXAMPLE OF EVOLVER EVOLUTION



#### **cat.fe**

A catenoid, which is the soap film formed between two parallel rings. Illustrates parameterized boundaries, energy saddle points, tiny edge deletion, and vertex popping. For more catenoids, see the [catenoid soap film page](#).

### Surface Evolver Overview

The Surface Evolver is an interactive program for the study of surfaces shaped by surface tension, gravitational energy, squared mean curvature energies, and subject to various constraints. A surface is implemented as a union of triangles.

The user defines an initial surface in a textfile accessible with Microsoft Word. The Evolver evolves the surface toward minimal energy by a gradient descent method. The aim can be to find a minimal energy surface, or to model the process of evolution by mean curvature. The Evolver can handle arbitrary topology (as seen in a real soap bubble or a bubble cluster), volume constraints, boundary constraints, boundary contact angles, prescribed mean curvature, crystalline integrands, gravity, and constraints expressed as surface integrals. The user can interactively modify the surface to change its properties or to keep the evolution well-behaved. The Evolver was written for one and two dimensional surfaces, but it can do higher dimensional surfaces with some restrictions on the features available. Graphical output is available as screen graphics.

The Surface Evolver program is freely available and is in use by a number of researchers. Some of the applications of the Evolver so far include modelling the shape of fuel in rocket tanks in low gravity, computing capillary surfaces in cubes and in exotic containers, simulating grain growth, studying grain boundaries pinned by inclusions, finding partitions of space more efficient than Kelvin's tetrakaidecahedra, modelling the shape of molten solder on microcircuits, studying polymer chain packing, modelling cell membranes, and classifying minimal surface singularities.

The strength of the Surface Evolver program is in the breadth of problems it handles, rather than optimal treatment of some specific problem. It is under continuing development.

The Evolver is written in portable C and has been run on several systems: Sun, Silicon Graphics, NeXT, Dec, MS-DOS, Macintosh, and various MS-Windows versions.

The Surface Evolver parameterizes a surface in terms of vertex coordinates. Restricting attention to one fixed triangulation of a surface, let  $X$  be the vector of all vertex coordinates, containing  $N$  entries, where  $N$  could be thousands. Call  $X$  the "configuration vector", and call the  $N$ -dimensional vector space it inhabits "configuration space". The total energy  $E(X)$  is a scalar function on configuration space. Visualize the graph of the energy function as a rolling landscape with mountains, valleys, and mountain passes. The gradient of the energy is the steepest uphill direction. The ordinary 'g' iteration step of Evolver minimizes energy by moving in the downhill direction, the negative of the gradient direction, until it reaches a local minimum.

## Surface Evolver - Basic Concepts

The basic geometric elements used to represent a surface are vertices, edges, facets, and bodies. Vertices are points in space. Edges are straight line segments joining pairs of vertices. Facets are flat triangles bounded by three edges. A surface is a union of facets. (Actually, there is no separate surface entity in the program; all facets belong to one logical surface.) A body is defined by giving it bounding facets.

The term "surface", when used to refer to the entity upon which the Evolver operates, refers to all the geometric elements plus auxiliary data such as constraints, boundaries, and forces. There are no limitations on how many edges may share a vertex nor on how many facets may share an edge. A surface is deemed to have a total energy, arising from surface tension, gravitational energy, and possibly other sources. It is this energy which the Evolver minimizes. No particular units of measurement are used. The program only deals with numerical values. If you wish to relate the program values to the real world, then all values should be within one consistent system, such as cgs or mks.

The initial surface is specified in a text file (hereafter referred to as the datafile) that may be created with any standard text editor such as Microsoft Word. (The .fe extension for datafiles stands for facet-edge, which refers to the internal data structure used to represent the surface. You may use any name you wish for a datafile.)

The basic operation of the Evolver is to read in a datafile and take commands from the user. The main command prompt is

*Enter command:*

The most common commands are one letter (case is significant for these), sometimes with a numerical parameter. The most frequently used commands are:

<b>g n</b>	<b>do n iterations ('g' is for 'go')</b>
<b>r</b>	<b>refine the triangulation of surface</b>
<b>h</b>	<b>list commands</b>
<b>s</b>	<b>Graphics commands (type h again and quit to return to run command)</b>
<b>q</b>	<b>quit</b>

There is a more elaborate command language which is listed upon typing help; a graphics command language is also accessible. All commands must be followed with the ENTER key; Evolver only reads complete lines.

An iteration is one evolution step. The motion for the step is calculated as follows: First, the force on each vertex is calculated from the gradient of the total energy of the surface as a function of the position of that vertex. The force gives the direction of motion. Second, the force is made to conform to whatever constraints are applicable. Third, the actual motion is found by multiplying the force by a global scale factor.

## Surface Evolver Documentation - Energies

The Evolver usually works by minimizing the total energy of the surface, subject to constraints. This energy can have several components: Surface tension energy, Gravitational potential energy, and Constraint energy integrals.

### Surface tension energy

Soap films and interfaces between different fluids have an energy content proportional to their area. Hence they shrink to minimize energy. The energy per unit area can also be regarded as a surface tension, or force per unit length.

In a Surface Evolver surface, each facet has a surface tension which is 1 unless the datafile specifies otherwise (see TENSION attribute for faces). Different facets may have different surface tensions. Facet tensions may be changed interactively with the set facet tension ... command. The contribution to the total energy is the sum of all the facet areas times their respective surface tensions. The surface tension of a facet may also be specified as depending on the phases of the bodies it separates. In the string model, the tension resides on edges instead of facets.

### Gravitational potential energy

If a body has a density, then that body contributes its gravitational energy to the total. The acceleration of gravity  $G$  is under user control with the  $G$  command. Letting  $\rho$  be the body density, the energy is defined as

$$E = \int \int \int_{\text{body}} G \rho z \, dV$$

but is calculated using the Divergence Theorem as

$$E = \int \int_{\text{body surface}} G \rho \left\{ \frac{z^2}{2} \right\} \vec{k} \cdot \vec{dS}.$$

This integral is done over each facet that bounds a body. If a facet bounds two bodies of different density, then the appropriate difference in density is used. Facets lying in the  $z = 0$  plane make no contribution, and may be omitted if they are otherwise unneeded. Facets lying in constraints may be omitted if their contributions to the gravitational energy are contained in constraint energy integrals. In the string model, all this happens in one lower dimension.

Example datafile: mound.fe

### Constraint energy integrals

An edge on a level-set constraint may have an energy given by integrating a vectorfield  $F$  over the oriented edge:

$$E = \int_{\text{edge}} F \cdot dl.$$

The integrand is defined in the constraint declaration in the datafile. The integral uses the innate orientation of the edge, but if the orientation attribute of the edge is negative, the value is negated. This is useful for prescribed contact angles on walls (in place of wall facets with equivalent tension) and for gravitational potential energy that would otherwise require facets in the constraint. The mound example illustrates this.

## PROBLEM 1 - Catenoid Stability

The catenoid is the soap film surface that forms between two rings which are not separated too far apart. Among the places in the body where such film mechanics are important is the lungs which are filled with surfactants that reduce the tissue(water)-air interface. In cylindrical coordinates, the catenoid's equation is  $r(z) = A \cosh(z/A)$ . In **cat.fe**, both the upper and lower rings are given as one-parameter boundary wires. The separation height, ZMAX, and ring radius, RMAX, are adjustable constant PARAMETERS; you can change them during a run by just entering **A**. The initial radius in the code below is the minimum for which a catenoid can exist for the unit separation of the rings. To get a stable catenoid, you will find that you must increase this value. However, if you do run with the original value, you can watch the neck pinch out. The initial surface consists of six rectangles forming a cylinder between the two circles. The vertices on the boundaries are fixed, or else they would slide along the boundary to short-cut the curvature; two diameters is shorter than one circumference. The boundary edges are fixed so that vertices arising from subdividing the edges are likewise fixed. *Below is the catenoid datafile in two short columns:*

```
// cat.fe
// Evolver data for catenoid.

PARAMETER RMAX = 1.5088795 // min radius for
height
PARAMETER ZMAX = 1.0

boundary 1 parameters 1 // upper ring
x1: RMAX * cos(p1)
x2: RMAX * sin(p1)
x3: ZMAX

boundary 2 parameters 1 // lower ring
x1: RMAX * cos(p1)
x2: RMAX * sin(p1)
x3: -ZMAX

vertices // given in terms of boundary parameter
1 0.00 boundary 1 fixed
2 pi/3 boundary 1 fixed
3 2*pi/3 boundary 1 fixed
4 pi boundary 1 fixed
5 4*pi/3 boundary 1 fixed
6 5*pi/3 boundary 1 fixed
7 0.00 boundary 2 fixed
8 pi/3 boundary 2 fixed
9 2*pi/3 boundary 2 fixed
10 pi boundary 2 fixed
11 4*pi/3 boundary 2 fixed
12 5*pi/3 boundary 2 fixed

edges
1 1 2 boundary 1 fixed
2 2 3 boundary 1 fixed
3 3 4 boundary 1 fixed
4 4 5 boundary 1 fixed
5 5 6 boundary 1 fixed
6 6 1 boundary 1 fixed
7 7 8 boundary 2 fixed
8 8 9 boundary 2 fixed
9 9 10 boundary 2 fixed
10 10 11 boundary 2 fixed
11 11 12 boundary 2 fixed
12 12 7 boundary 2 fixed
13 1 7
14 2 8
15 3 9
16 4 10
17 5 11
18 6 12

faces
1 1 14 -7 -13
2 2 15 -8 -14
3 3 16 -9 -15
4 4 17 -10 -16
5 5 18 -11 -17
6 6 13 -12 -18
```

The parameter in a boundary definition, such as the sin or cos argument above, is always p1 (and p2 in a two-parameter boundary). The Evolver can handle periodic parameterizations, as is done in this example. Enter the following sequence of commands (while you display the surface):

```
cat.fe (reads datafile)
r (refine triangulation to get a crude, but workable, mesh)
u (equiangulation makes much better triangulation)
g 120 (takes this many iterations for neck to collapse)
t 0.05 (collapse neck to single vertex by eliminating all edges shorter than 0.05)

o (split neck vertex to separate top and bottom surfaces)
g 20 (spikes collapse)
```

QUESTION: Complete the table below using Surface Evolver with ZMAX = 1 and Surface Tension = 1.

<b>R<sub>max</sub></b>	<b>iteration scheme</b>	<b>Area</b>	<b>Energy</b>
1.5088795	120 then <b>o</b> (pop)		
	+20 more, <b>rr</b> , +20		
1.40	120 then <b>o</b> (pop)		
	+20 more, <b>rr</b> , +20		
3.0	120		
	<b>rr</b> , +30		

QUESTION: Is the convergence good? Is there a strong mesh-size effect?

QUESTION: Complete the table below using your own calculations.

<b>R<sub>max</sub></b>	<b>surface type</b>	<b>Area</b>	<b>Energy</b>
1.5088795	two disks	$2 \pi (R_{\max})^2 =$	
	catenoid*	calculate at home	
1.40	two disks		
	catenoid*	calculate at home	
3.0	two disks		
	catenoid*	calculate in class	

\* For each R<sub>max</sub>, first determine the waist radius A by iteratively solving<sup>(see next pg)</sup> the equation:

$$R_{\max} / A = \cosh(Z_{\max} / A) \quad \text{where } Z_{\max} = 1.$$

Then evaluate the expression:

$$\text{Catenoid Area} = 2 * 2 \pi \int_0^{Z_{\max}} A \cosh(z/A) dz = 4\pi A^2 \sinh(Z_{\max} / A).$$

QUESTION: How do the two tables compare?

**Iterative solution for catenoid waist radius A with  $R_{max} = 3$  and  $Z_{max} = 1$ .**

$$3 / A - \cosh(1 / A) = \text{Residual}$$

Based on the final Surface Evolver configuration, we expect that A is just less than 3. We therefore start with this as a first guess.

<i>guessed A</i>	$3 / A$	$\cosh(1 / A)$	Residual
3	1	1.056	-0.056
2	1.5	1.128	0.372
2.87			-0.016

*Now you should estimate the next best interpolated guess that further reduces the Residual.*

-----  
**Lastly, we note that the Surface Evolver evolution can be speeded up by turning on the conjugate gradient method with the U command.** For conjugate gradient cognoscenti, the saddle point demonstrates the difference between the Fletcher-Reeves and Polak-Ribiere versions of conjugate gradient. The saddle point seems to confuse the Fletcher-Reeves version (which used to be the default). However, the Polak-Ribiere version (the current default) has little problem. The U toggles conjugate gradient on and off, and ribiere toggles the Polak-Ribiere version. With Fletcher-Reeves conjugate gradient in effect, the saddle point is reached at iteration 17 and area starts decreasing again until iteration 30, when it reaches 6.4486. But then iteration stalls out, and the conjugate gradient mode has to be turned off and on to erase the history vector. Once restarted, another 20 iterations will get the area down to 6.4334. In Polak-Ribiere mode, no restart is necessary.

The catenoid shows some of the subtleties of evolution. Suppose the initial radius is set to RMAX = 1.0 and the initial height to ZMAX = 0.55. Fifty iterations with optimizing scale factor result in an area of 6.458483. At this point, each iteration is reducing the area by only .0000001, the triangles are all nearly equilateral, everything looks nice, and the innocent user might conclude the surface is very near its minimum. But this is really a saddle point of energy. Further iteration shows that the area change per iteration bottoms out about iteration 70, and by iteration 300 the area is down to 6.4336. The triangulation really wants to twist around so that there are edges following the lines of curvature, which are vertical meridians and horizontal circles. Hence the optimum triangulation appears to be rectangles with diagonals.