Chapter 8

Special Models for Animation

Chapter 8

L-Systems

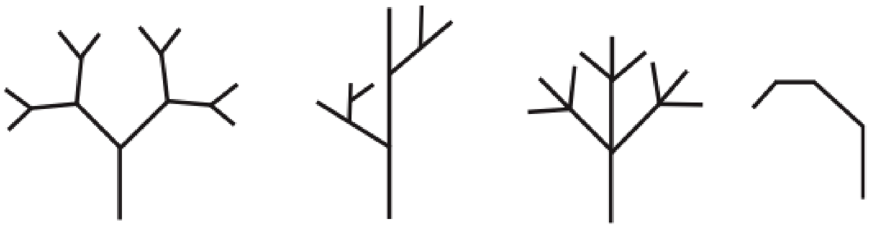
Chapter 8

L-Systems

Branching Structures
Botany
Display
- geometric substitution
- turtle graphics

Animating plants, animals

Basic branching schemes

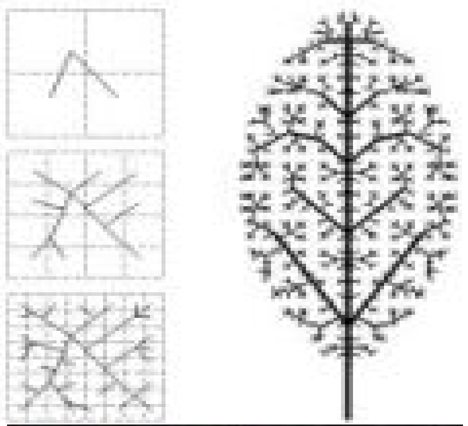


Structures resulting from repeated application of a single branching scheme

Chapter 8

Plant examples

http://algorithmicbotany.org/papers/#abop






Chapter 8

As a Formal Grammar

Related to fractals

- recursive branching structure
- often self-similar under scale

Grammar

- parallel rewriting system
- context-free (in basic version)

Chapter 8

Historical development

Aristid Lindenmayer
- botanist
- the ‘L’ in L-systems

Przemyslaw Prusinkiewicz
- U. of Calgary
- introduced L-systems to graphics
- *The Algorithmic Beauty of Plants*

Chapter 8

DOL-systems

Basic version

Deterministic

Context-free

rules	words	
S –> ABA	S	← axiom
A –> XX	ABA	
B –> TT	XXTTXX	

Chapter 8

Geometric interpretation

Geometric substitution

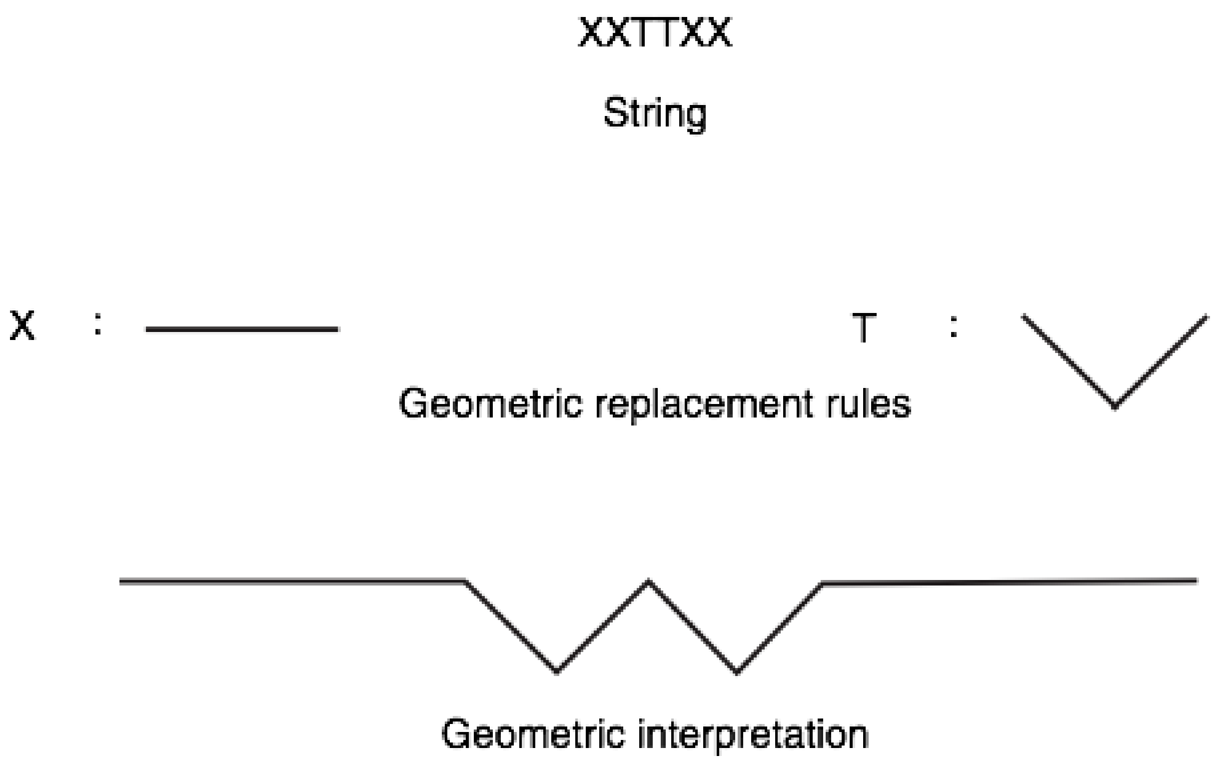
symbol -> geometric element

Turtle graphics

symbol -> drawing command

Chapter 8

Geometric substitution



Chapter 8

Turtle graphics

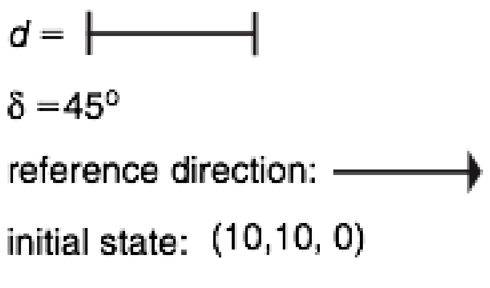
F	move forward w/ drawing
f	move forward w/o drawing
+	turn left
-	Turn right

rules	words
S -> ABA	**S**
A -> FF	**ABA**
B -> TT	**FFTTFF**
T->-F++F-	**FF-F++F--F++F-FF**

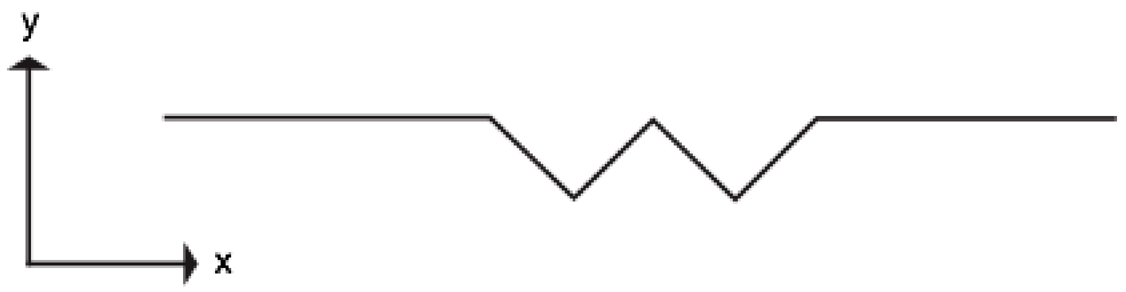
Chapter 8

Turtle graphics

FF-F++F--F++F-FF



Initial conditions



Geometric interpretation

Chapter 8

Botany: terms

Stems, roots, buds, leaves, flowers
Nodes, internodes
Herbaceous v. woody
Dichotomous, monopodial
Lateral bud
Leavs from buds: alternate, opposite whorled
Cell influence: lineage, tropisms, obstacles

Discrete components: apices, internodes, leaves, flowers
Finite number of components
Components represented by symbols

Chapter 8

Bracketed L-Systems

Also add non-determinism
database amplification
procedural models

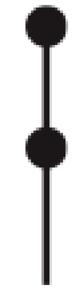
Brackets -> branch

S->FAF
A->[+FBF]
A->F
B->[-FBF]
B->F

Chapter 8

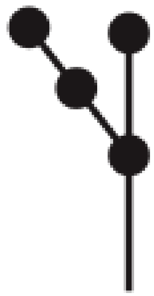
Example

FAF

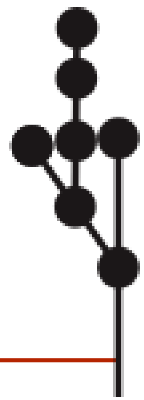


S->FAF
A->[+FBF]
A->F
B->[-FBF]
B->F

F(+FBFF)



F(+F(−FFF)F)F



Chapter 8

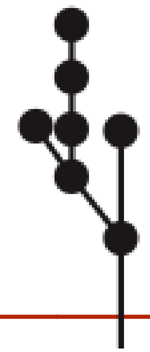
More examples

FFF

S->FAF
A->[+FBF]
A->F
B->[-FBF]
B->F

F(+FFF)F

F(+F(−FFF)F)F



Chapter 8

Stochastic L-System

Add probabilities to non-deterministic L-systems

These probabilities will control how likely a production will be to form a branch at each possible branching point:

Controls average termination level

$$S_{1.0} \Rightarrow FAF$$
$$A_{0.8} \Rightarrow (+FBF)$$
$$A_{0.2} \Rightarrow F$$
$$B_{0.4} \Rightarrow (-FBF)$$
$$B_{0.6} \Rightarrow F$$

Chapter 8

Context-sensitive

Better control of rule application

$$S \Rightarrow FAT$$
$$A > T \Rightarrow (+ FBF)$$
$$A > F \Rightarrow F$$
$$B \Rightarrow (- FAF)$$
$$T \Rightarrow F$$

S
FAT
F(+FBF)F
F(+F(−FAF)F)F

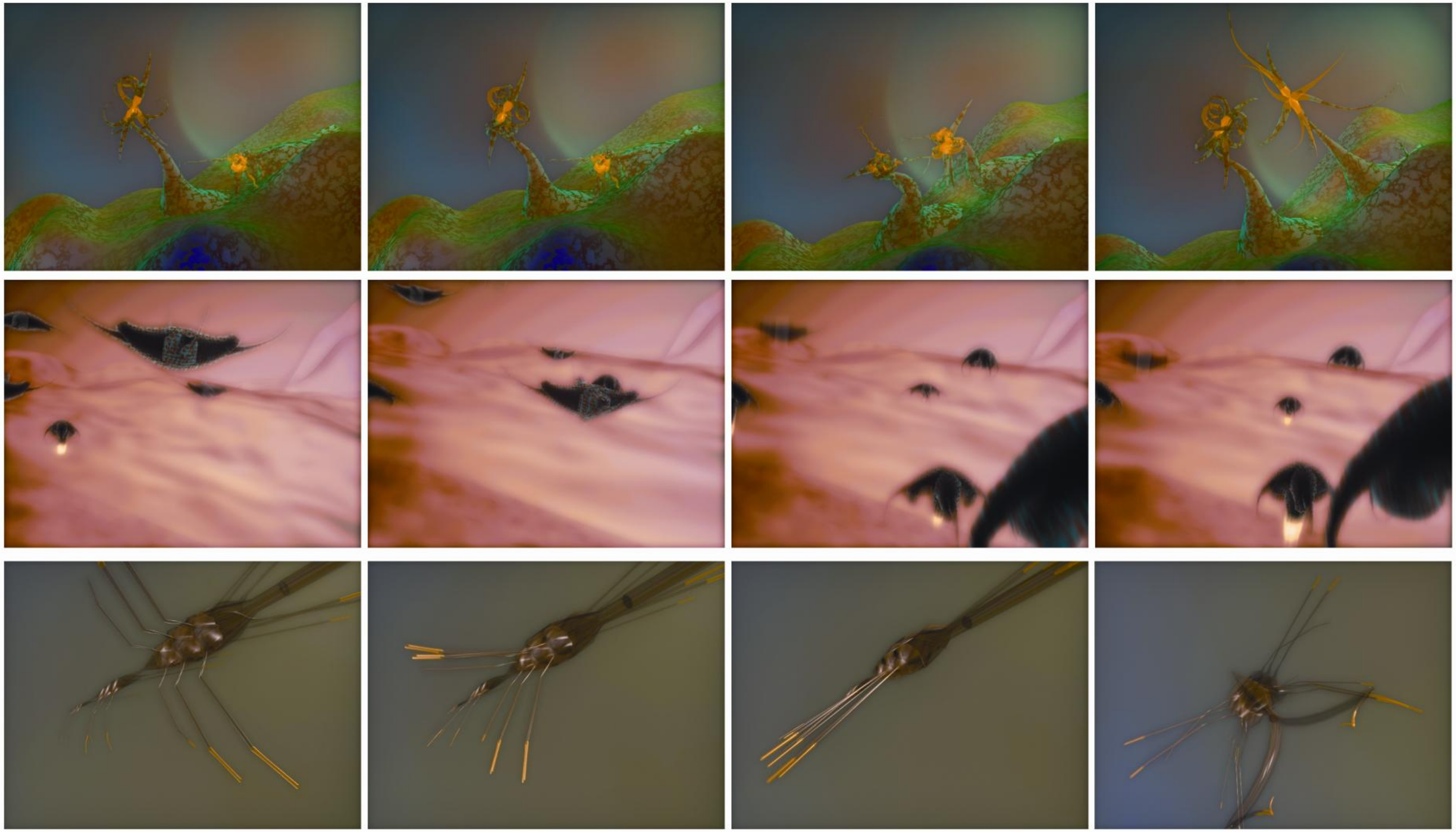
Chapter 8

Animating plant growth

Changes in topology
Elongation of existing structures
Changing angles, lengths

Chapter 8

Animating branches



Chapter 8

Parametric L-systems

A parameter can be associated with the symbols:

S	=> A(0)
A(t)	=> A(t + 0.01)
A(t):t>=1.0	=> F

Chapter 8

Context-sensitive, timed w/ conditions

$$A(t0) < A(t1) > A(t2): t2 > t1 \; \& \; t1 > t0 \Rightarrow A(t1+0.01)$$

Chapter 8

Open L-systems environmental interaction

Plant

Reception of information from environment
Transport and processing of info inside plant
Response in form of growth changes

Environment

Perception of plants actions
Simulate processes of environment (e.g. light propagation)
Present modified environment to plant

Chapter 8

Open L-systems environmental interaction

Add construct to L-systems

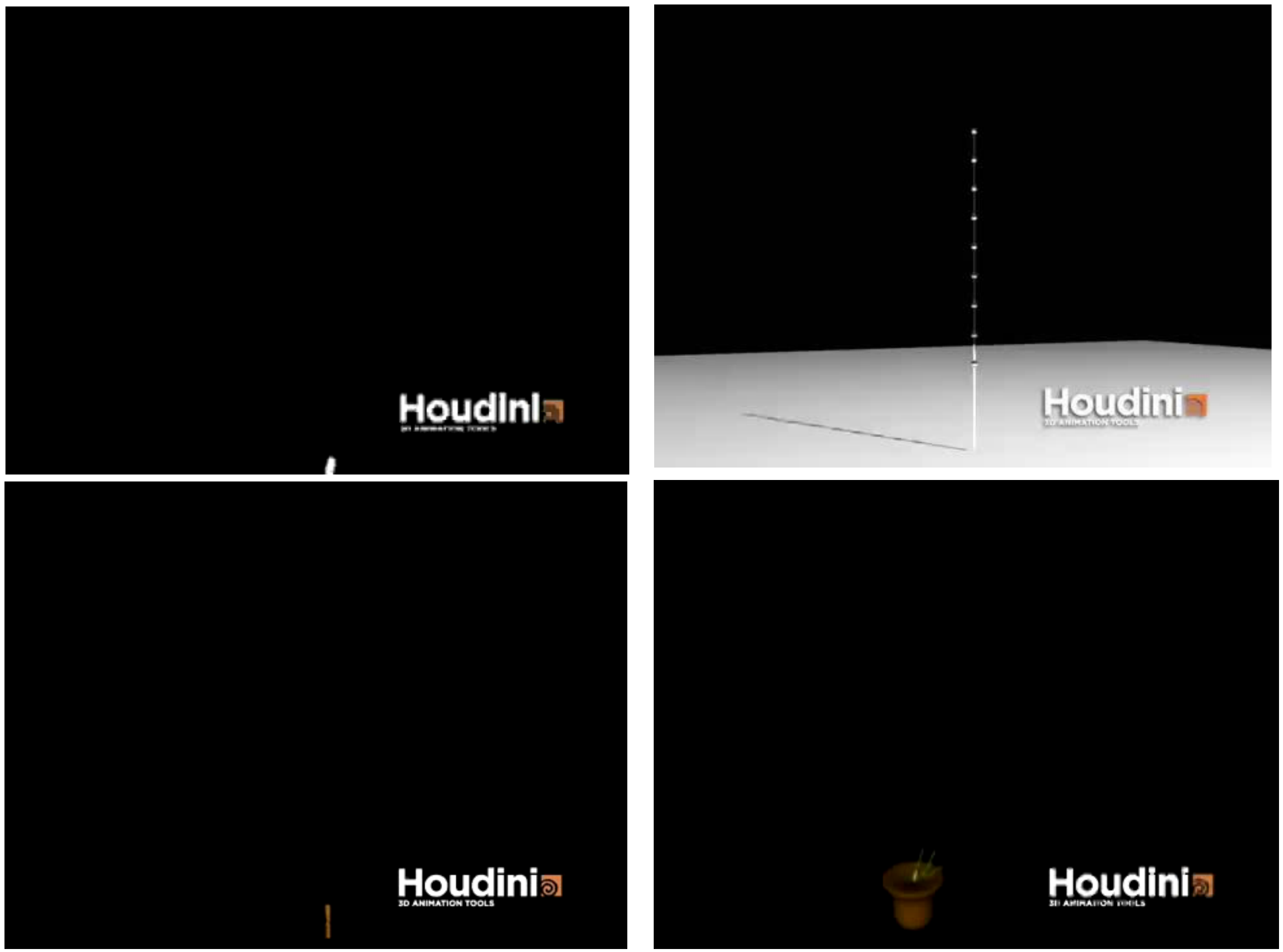
$\underline{?E(x_1,x_2,\ldots,x_m)}$

(a bit simply) Query appears in production - sends message to environment which then returns value to production

See paper by Mech and Prusinkiewicz for details

Chapter 8

Examples



Chapter 8

Implicit Surfaces

Chapter 8

Implicit Surfaces

Surface is only *implicitly* defined

$$f(P) = 0$$

explicit $\quad y = f(x)$

parametric $\quad \begin{aligned} x &= f(t) \\ y &= g(t) \end{aligned}$

Chapter 8

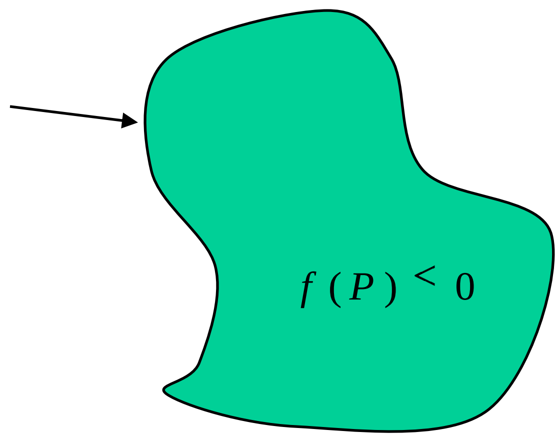
Implicit Surfaces

Basic formulation
Animation
Collision detection
Deforming implicits
Level set methods

Chapter 8

Implicit Surfaces

$f(P) = 0$



$f(P) > 0$

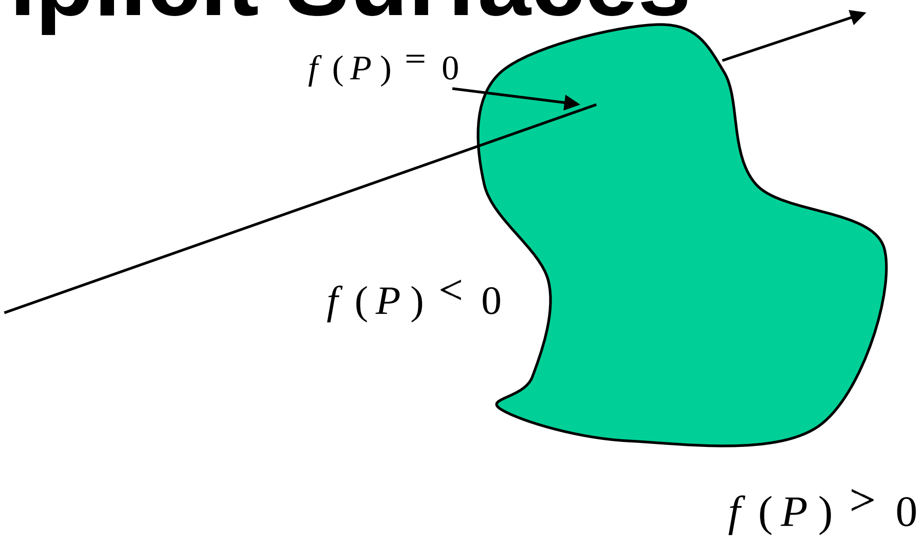
Usually define so:
surface = 0
inside <0
outside > 0

Chapter 8

Displaying Implicit Surfaces

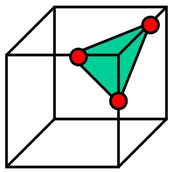
Ray Tracing

Search along ray
to find zero point



Marching cubes

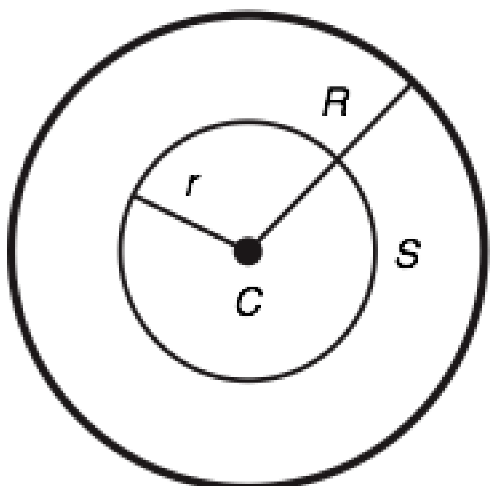
embed in 3D volume of cells
intersect cell edges with surface
define polygonal pieces from cell intercepts
see examples a few slides later

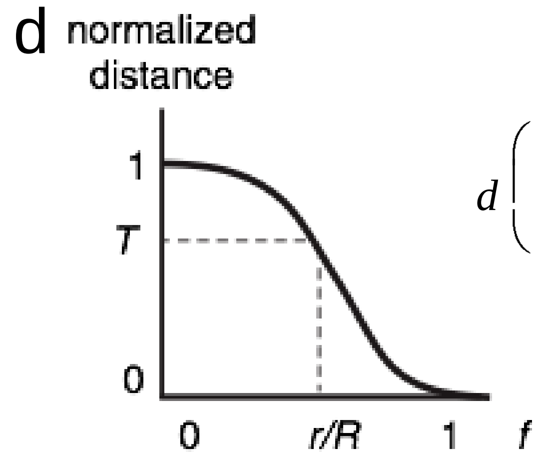


Chapter 8

Metaball - spherical, distance-based implicit

The best-known implicit primitive is often referred to as the metaball and is defined by a central point C, a radius of influence R, a density function f, and a threshold value T.





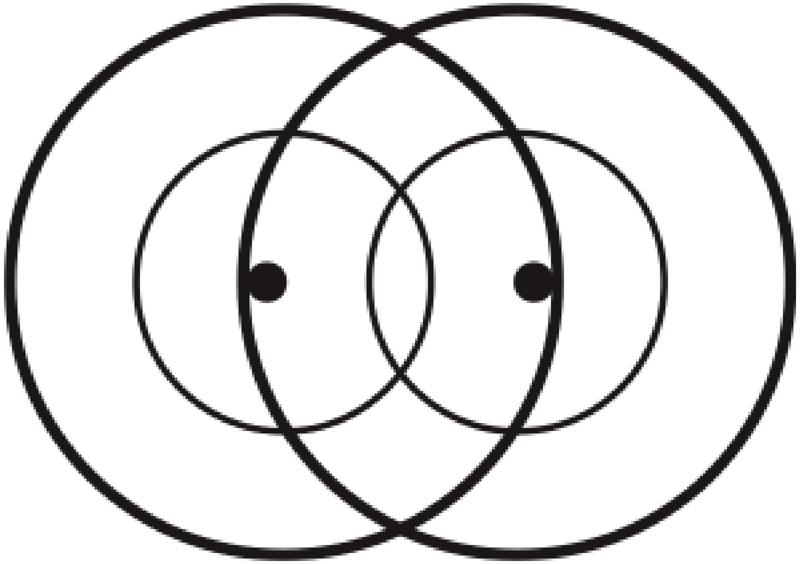
$$d\left(\frac{r}{R}\right) = T$$

$f(P) = d\left(\frac{dist(P,C)}{R}\right) - d\left(\frac{r}{R}\right) = 0$ describes the surface S

Chapter 8

Multiple Implicits

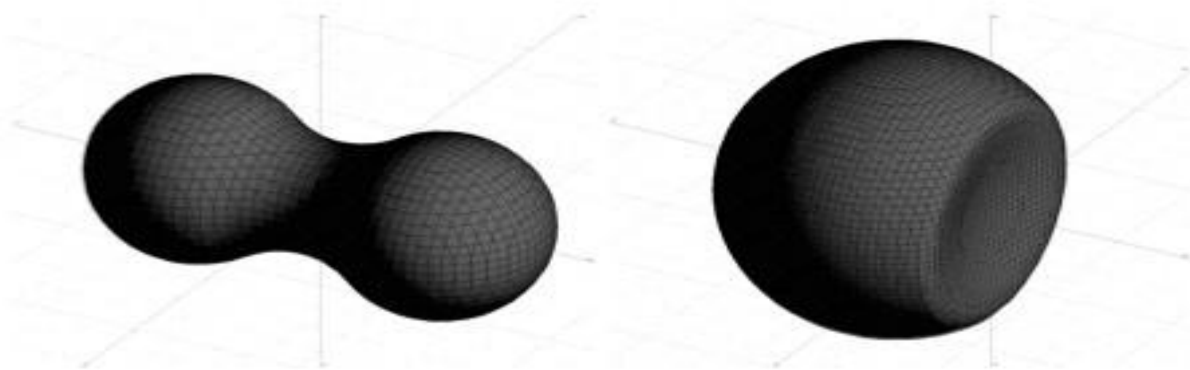
Sum overlapped implicits - with weights



$$F(P) = \sum w_i f_i(P) - T = 0$$

Chapter 8

Implicit Surfaces

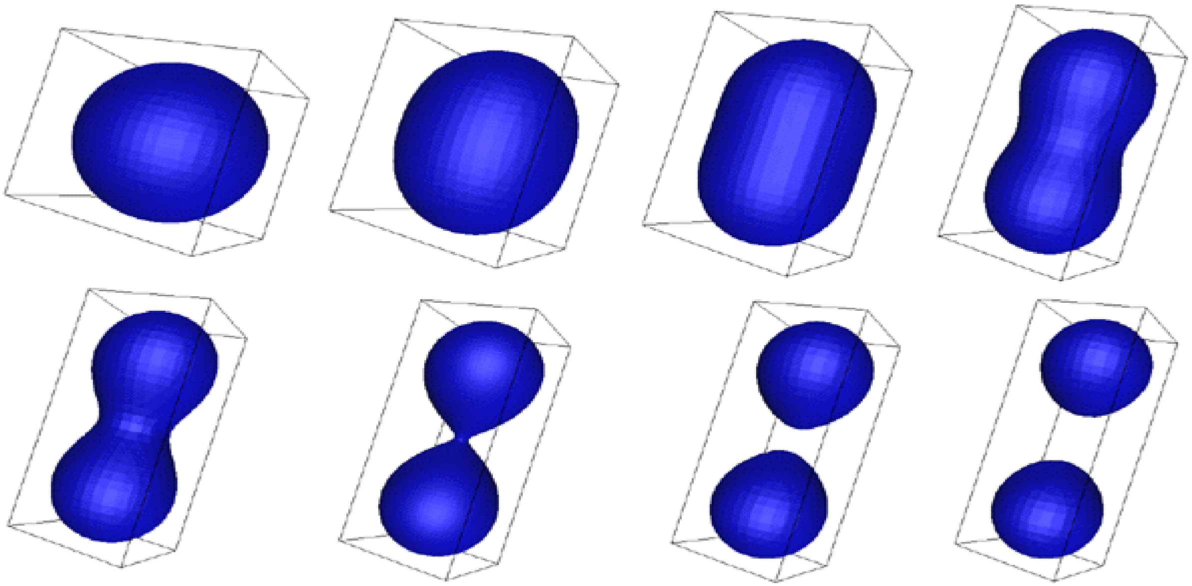


Surface constructed when positive weights are associated with density functions

Surface constructed when one positive weight and one negative weight are associated with density functions

Chapter 8

Topology smoothly changes



http://local.wasp.uwa.edu.au/~pbourke/modelling_rendering/implicitsurf/

Chapter 8

Signed-distance-based primitives

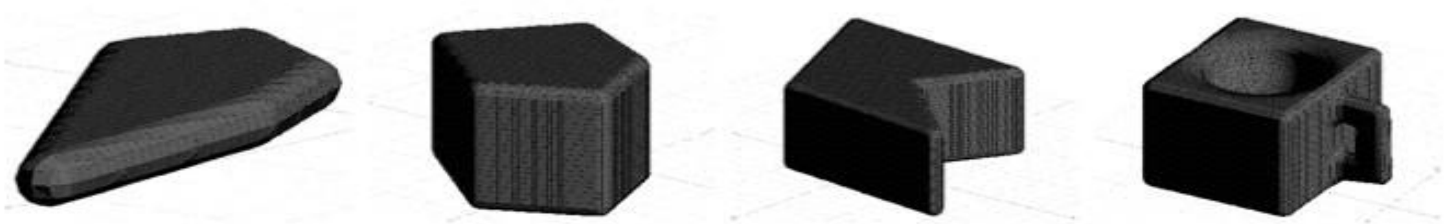
From
- Point
- Edge
- Face
- Polyhedron

$$f(P) = d\left(\frac{dist(P, central - element)}{R}\right) - T = 0$$

Hence, the density function describes the distance from the basic primitive instead of just a point.

Chapter 8

Implicit Surfaces



a) distance-based implicit primitive based on single polygon

b) distance-based implicit primitive based on a a single polygon

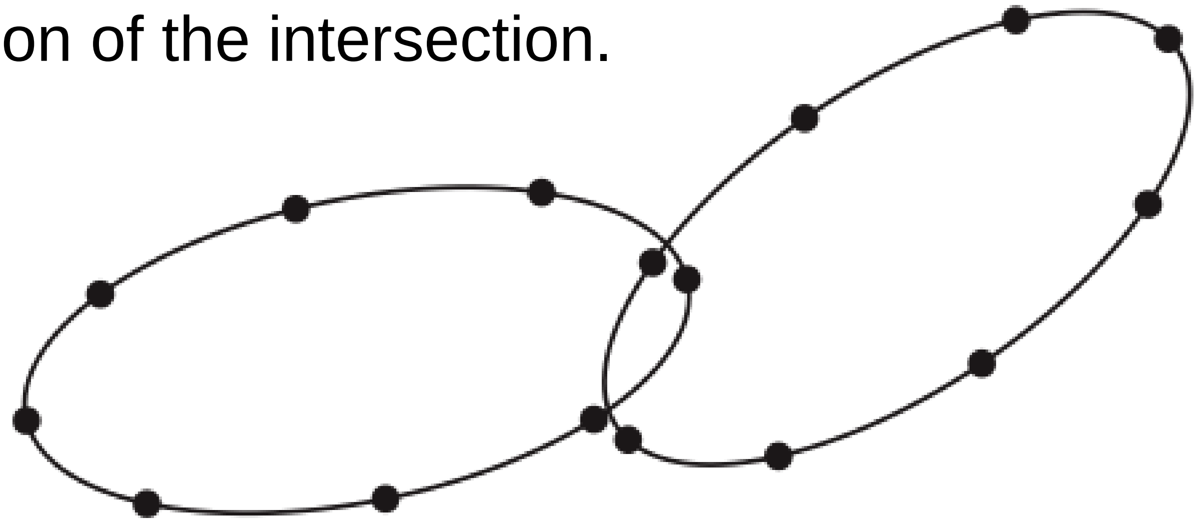
c) distance-based implicit primitive based on a single polygon

d) Compound implicitly defined object

Chapter 8

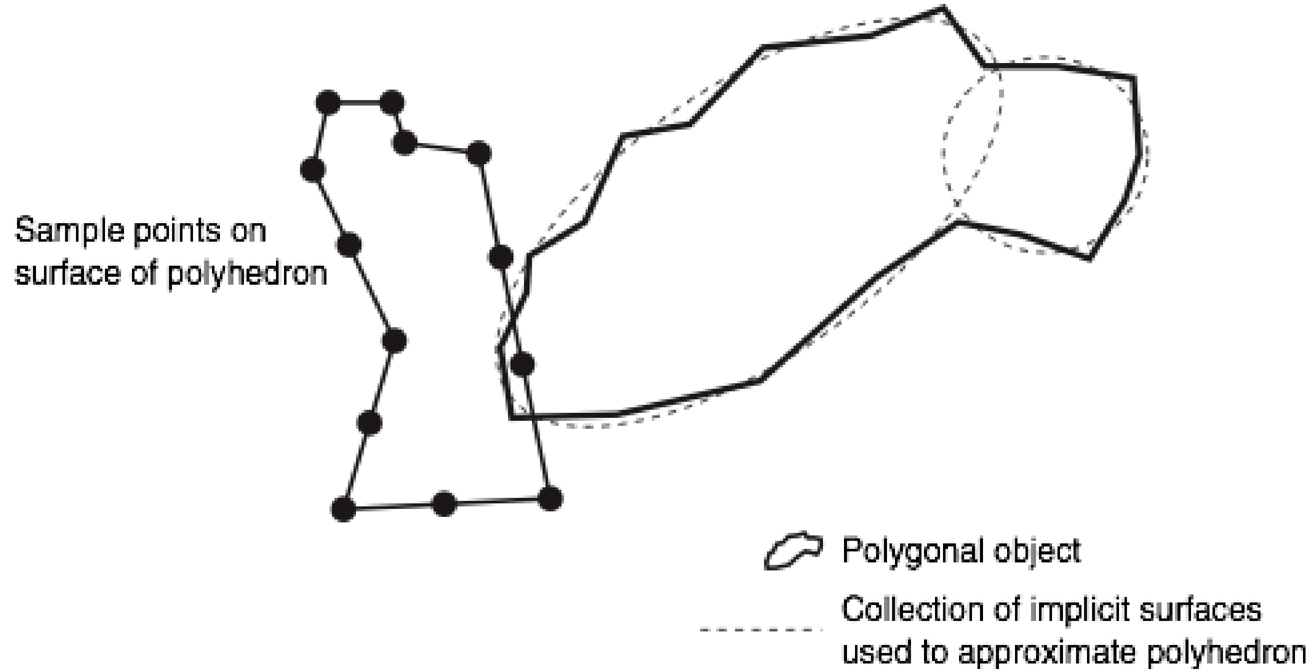
Testing - good for collision detection

Implicitly defined objects lend themselves to collision detection. Sample points on the surface of one object can be tested for penetration with an implicit object by merely evaluating the implicit function at those points. Numerical subdivision can yield a more accurate location of the intersection.



Chapter 8

Polyhedra embedded in implicits



Using implicit surfaces for detecting collision between polyhedral objects

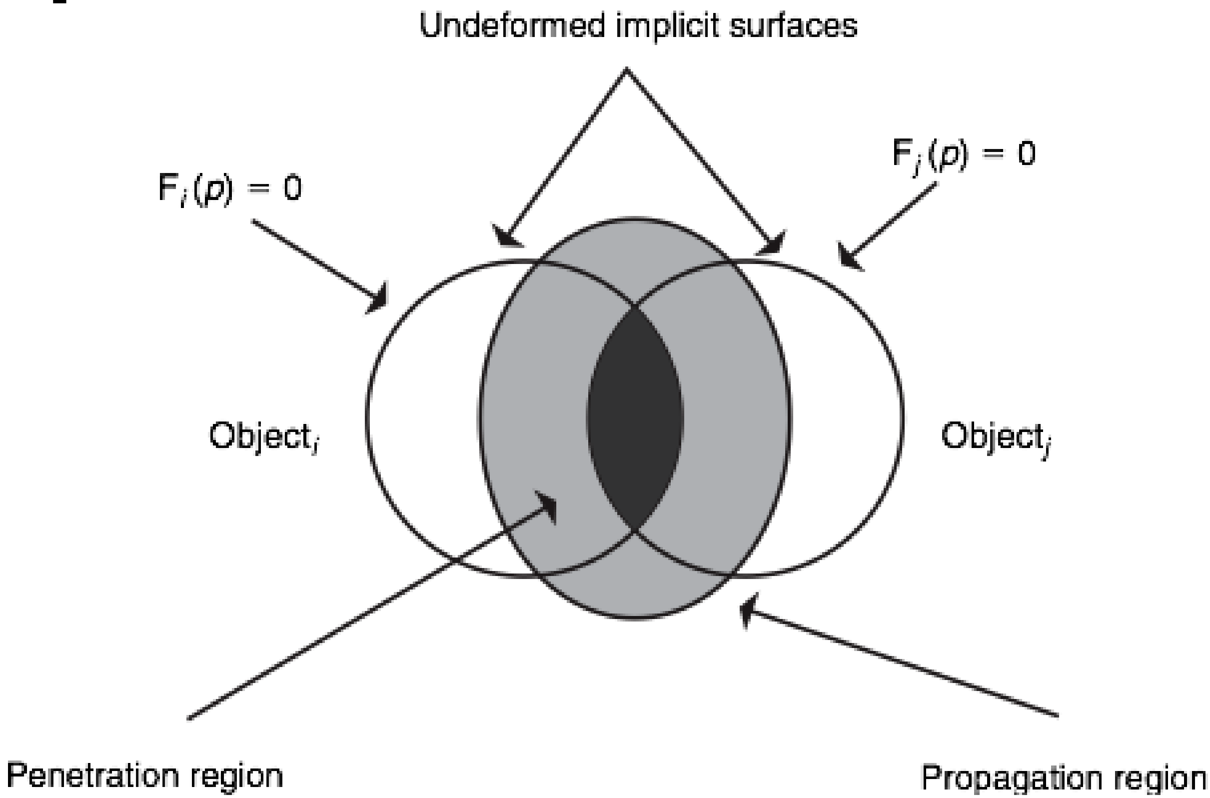
Chapter 8

Deforming as a Result of Collision

Marie-Paul Cani has developed a technique to compute the deformation of colliding implicit surfaces. This technique first detects the collision of two implicit surfaces by testing sample points on the surface of one object against the other. The overlap of the areas of influence of the two implicit objects is called the **penetration region**. An additional region just outside the penetration region is called the **propagation region**.

Chapter 8

Implicit Surfaces



Penetrating implicit surfaces

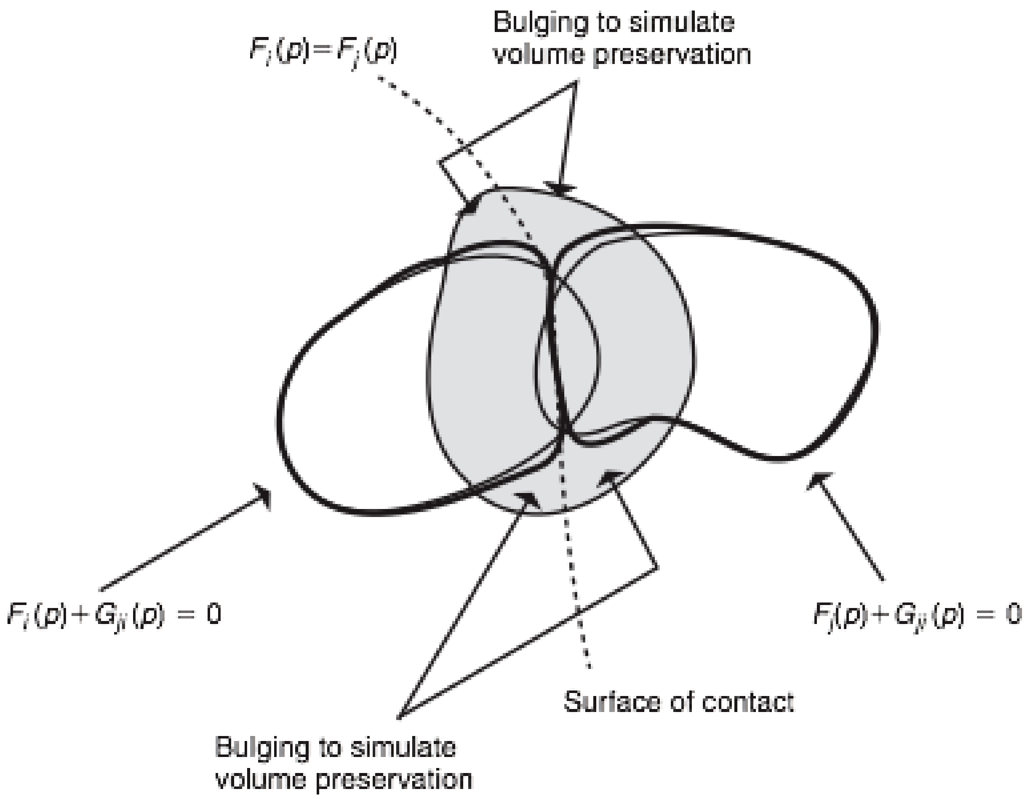
Chapter 8

Deforming as a Result of Collision (continued)

The density function of each object is modified by the overlapping density function of the other object so as to deform the implicitly defined surface of both objects so that they coincide in the region of overlap, thus creating a contact surface. A deformation term is added to F_i as a function of $Object_j$'s overlap with $Object_i$, G_{ij}, to form the contact surface. Similarly, a deformation term is added to F_j as a function of $Object_i$'s overlap with $Object_j$, G_{ji}. The deformation functions are defined so that the isosurface of the modified density functions, $F_i(p)+G_{ij}(p)=0$ and $F_j(p)+G_{ji}(p)=0$, coincide with the surface defined by $F_i(p)=F_j(p)$.

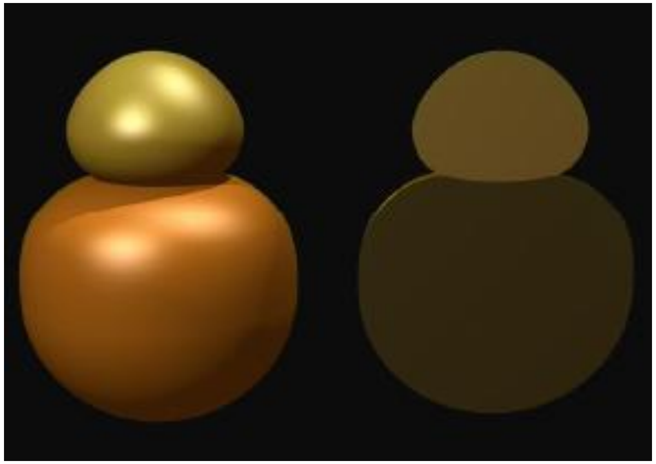
Chapter 8

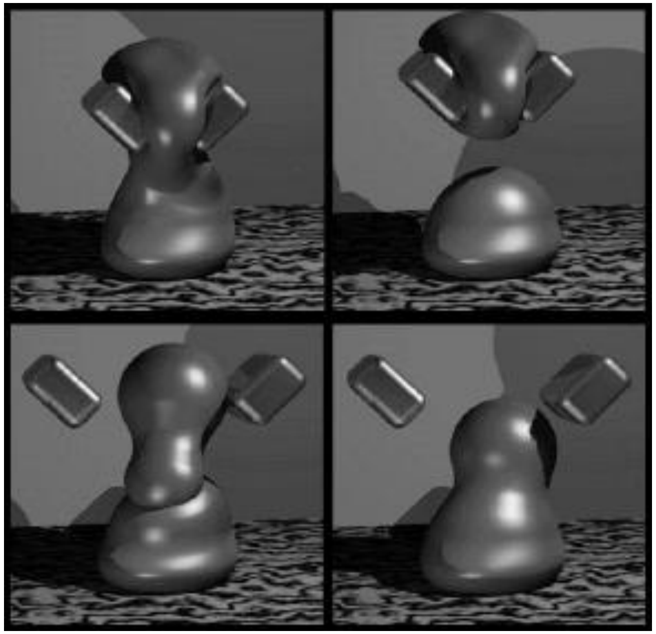
Colliding Implicit Surfaces



Chapter 8

Colliding Implicit Surfaces





Chapter 8

Level Sets

Chapter 8

Level Set Methods

Adds dynamics to implicit surfaces

Usually operate on signed distance function

Isosurface updated according to velocity field defined over interface

Tracing particles on curve is problematic

Chapter 8

Fundamental Idea

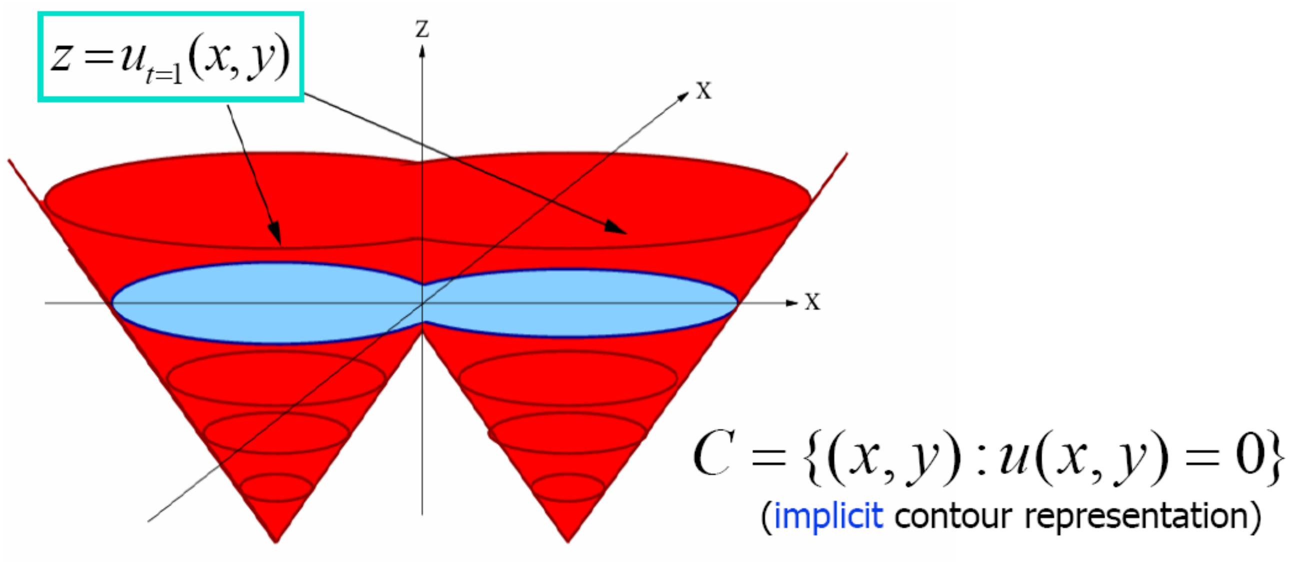
Instead of evolving curve $C(t) = 0$
Evolve surface, U, that curve is a level set of

Common surface used is signed distance function
$U(x,y)$ = distance to nearest point in C

If U evolves according to U_t, C will evolve by C_t

Chapter 8

Level Set - Fundamental idea



Chapter 8

Front Propogation

Isosurface advects

Normally in direction of gradient

$$n = \frac{\nabla \phi}{\left|\nabla \phi\right|}$$

Can have constant magnitude

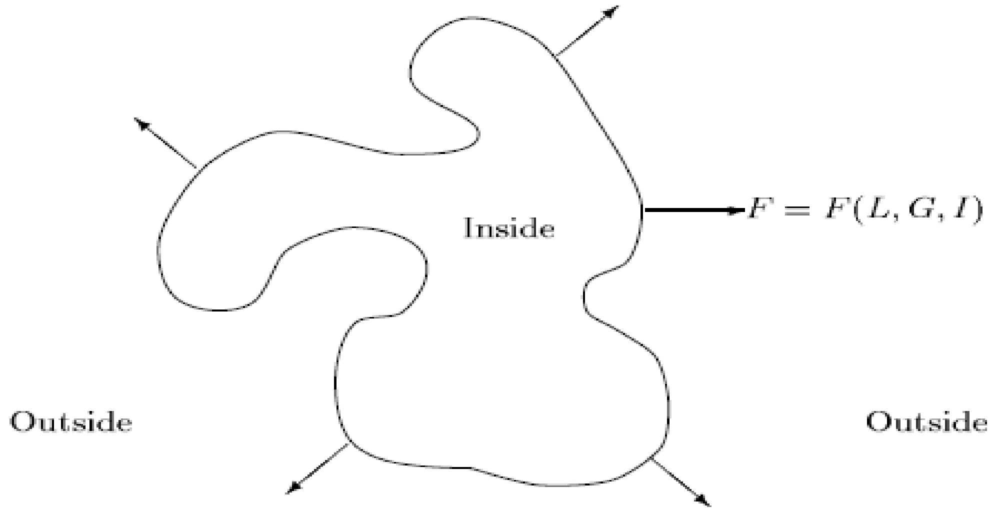
$$\nabla \phi = \left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right)$$

Or use magnitude of curvature

$$\frac{d^2 \phi}{dt^2}$$

Chapter 8

Level Set Methods



Chapter 8

Front Propagation

More generally $$\frac{d\phi}{dt} = F(L, G, I)$$

Velocity can depend on:

- Local properties (e.g. curvature)
- Global properties (e.g. position of front)
- Properties Independent of shape (e.g. transport function)

Chapter 8

Level Set Equation

V - velocity field

Convection equation $$\frac{\partial \phi}{\partial t} + V \cdot \nabla \phi = 0$$

$$V \cdot \nabla \phi = V \cdot \frac{\nabla \phi}{|\nabla \phi|} |\nabla \phi| = V \cdot n |\nabla \phi|$$

$$V \cdot n = F$$

$$\frac{\partial \phi}{\partial t} + F |\nabla \phi| = 0$$

Chapter 8

Level Set Equation

F can be:

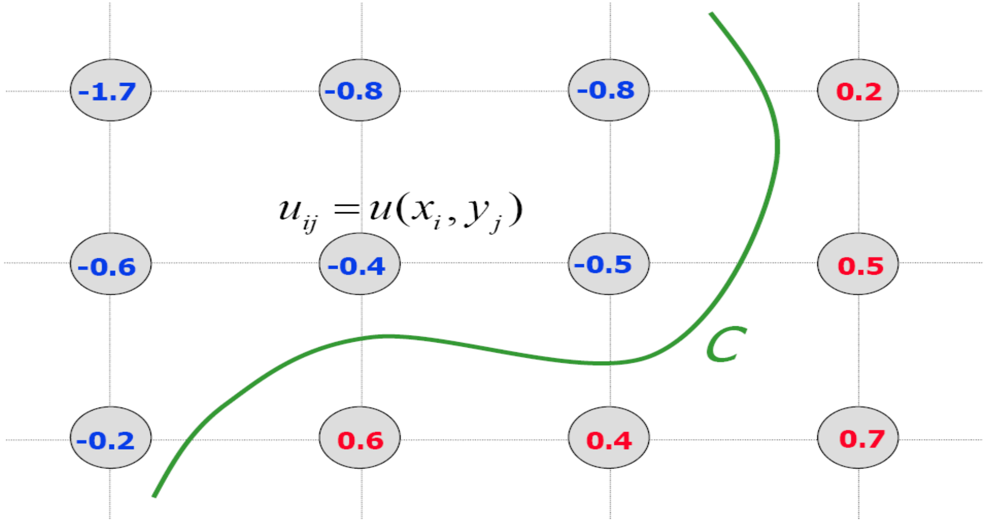
- Constant
- Function of gradient
- Function of curvature

$$\kappa = div\left(\frac{\nabla\phi}{\left|\nabla\phi\right|}\right) = F(\nabla\phi)$$

Chapter 8

Implementation

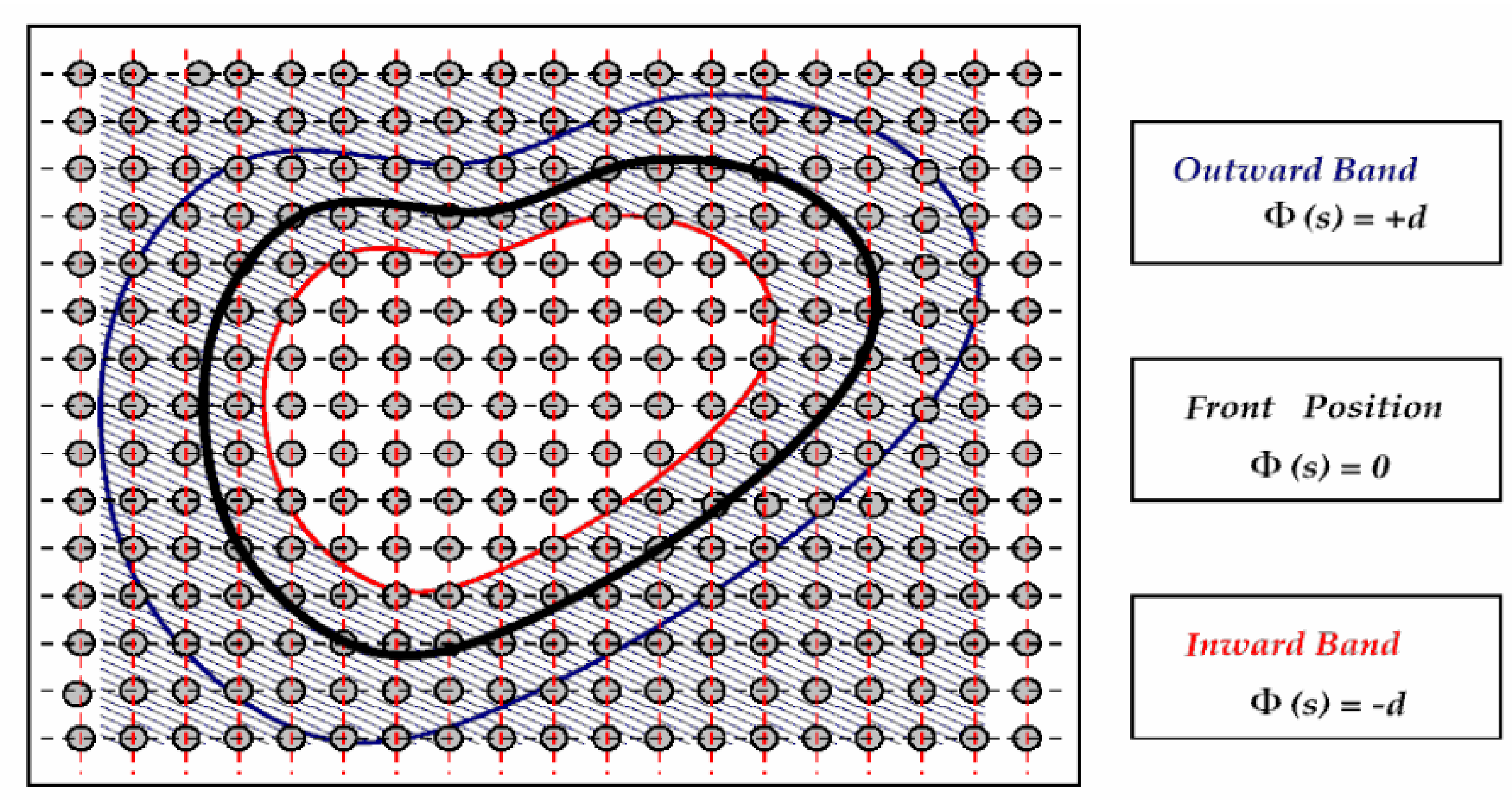
Use grid to hold distance function



http://www.cs.cornell.edu/Courses/cs664/2005fa/Lectures/lecture24.pdf

Chapter 8

Narrow band



Chapter 8

Subdivision surfaces

Chapter 8

Subdivision surfaces

Use coarse polyhedron as general shape
Refine to generate smooth surface

Gives high-level shape control

By its nature is a level-of-detail representation

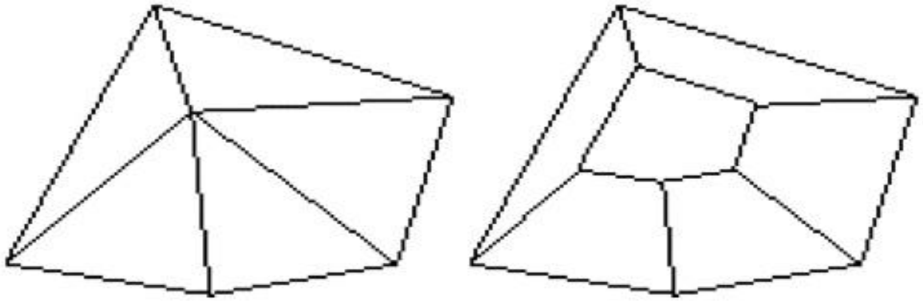
Does it shrink or expand the object?

Issue: what is the limit surface?

Chapter 8

Simple Subdivision

Cut off each corner of polyhedron and replace with face



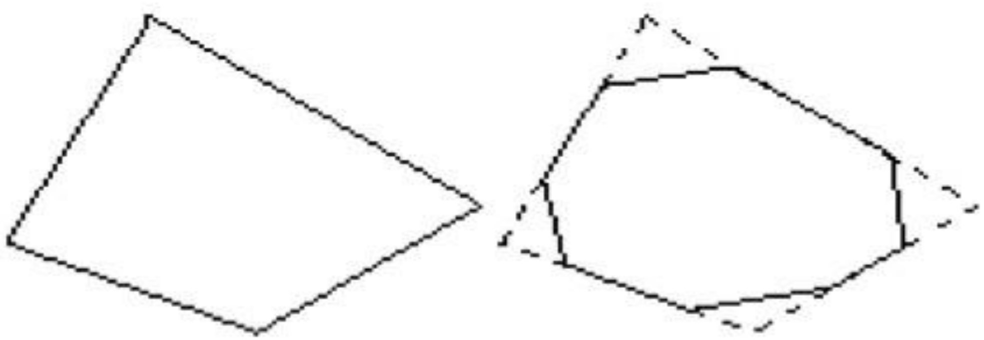
a) original vertex of object to be subdivided

b) A face replaces the vertex by using new vertices defined on connecting edges

Chapter 8

Subdivision surfaces

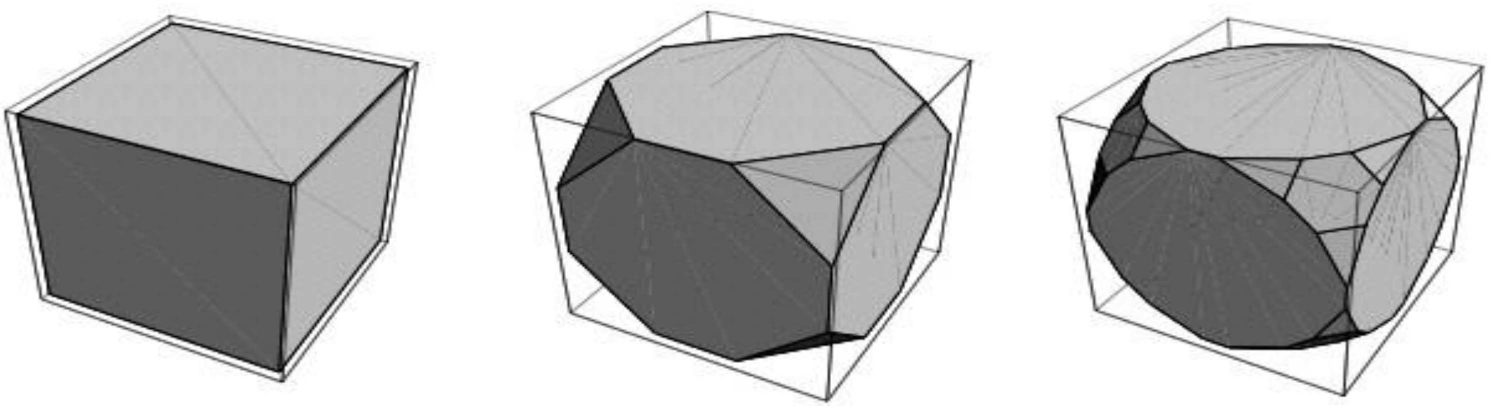
Redefine faces



a) Original face of object to be subdivided

Chapter 8

Subdivision surfaces



But doesn't smooth large flat areas

Chapter 8

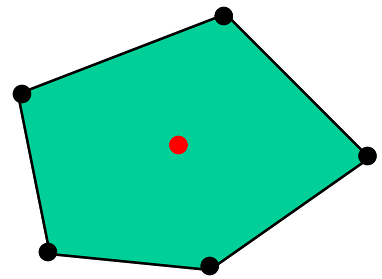
Various Subdivision schemes

Doo-Sabin
Catmull-Clark
Loop
Butterfly (not shown below)

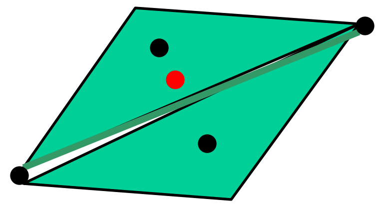
Following images are from:
http://www.holmes3d.net/graphics/subdivision/

Chapter 8

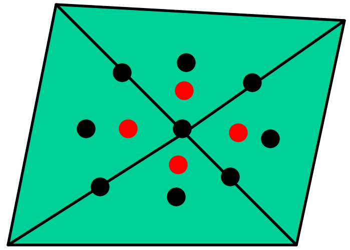
Doo-Sabin Subdivision



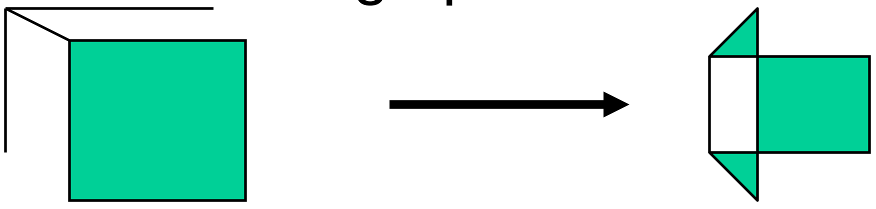
Face point - for each face, average of vertices



Edge point - average of 2 edge vertices and 2 new face points

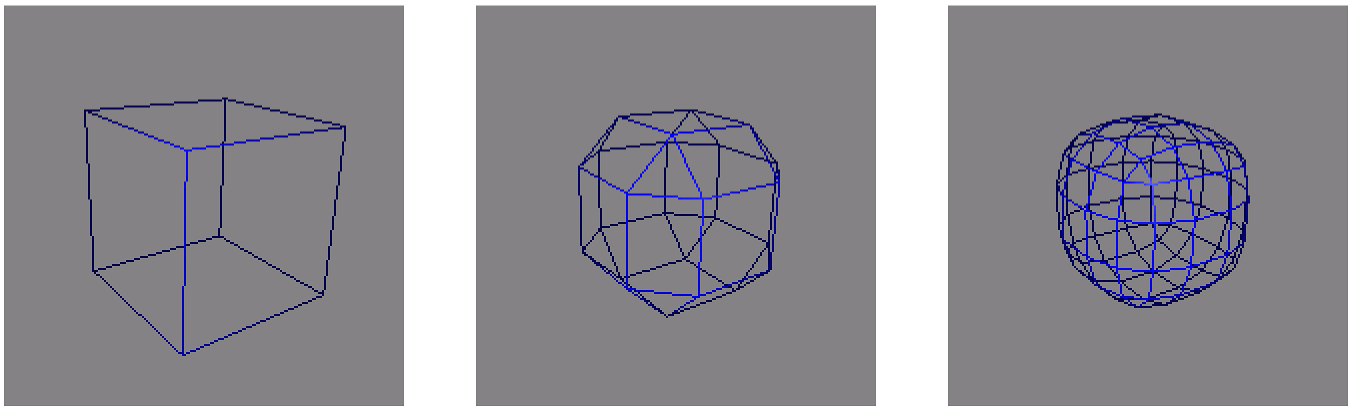


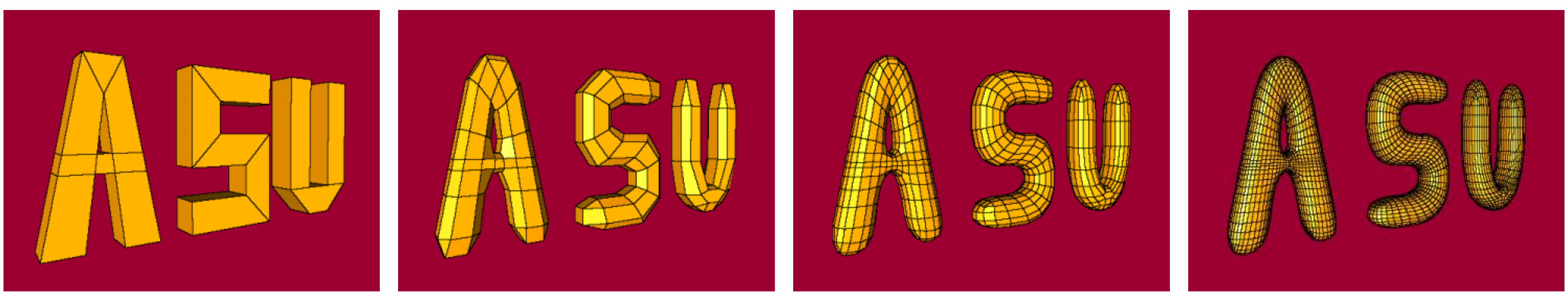
Vertex point - for each face, average the vertex, the face point and two edge points



Chapter 8

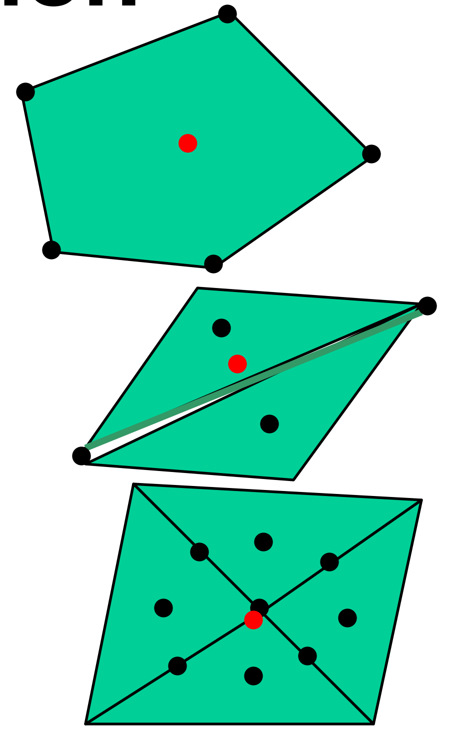
Doo-Sabin Subdivision





Chapter 8

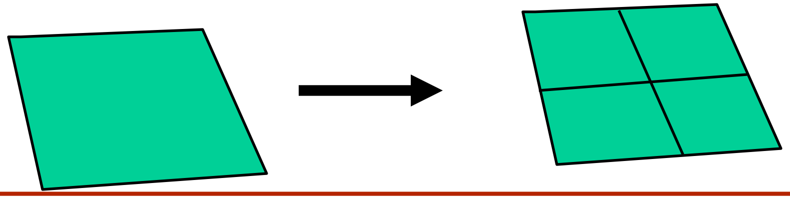
Catmull-Clark Subdivision



Face point - for each face, average of vertices

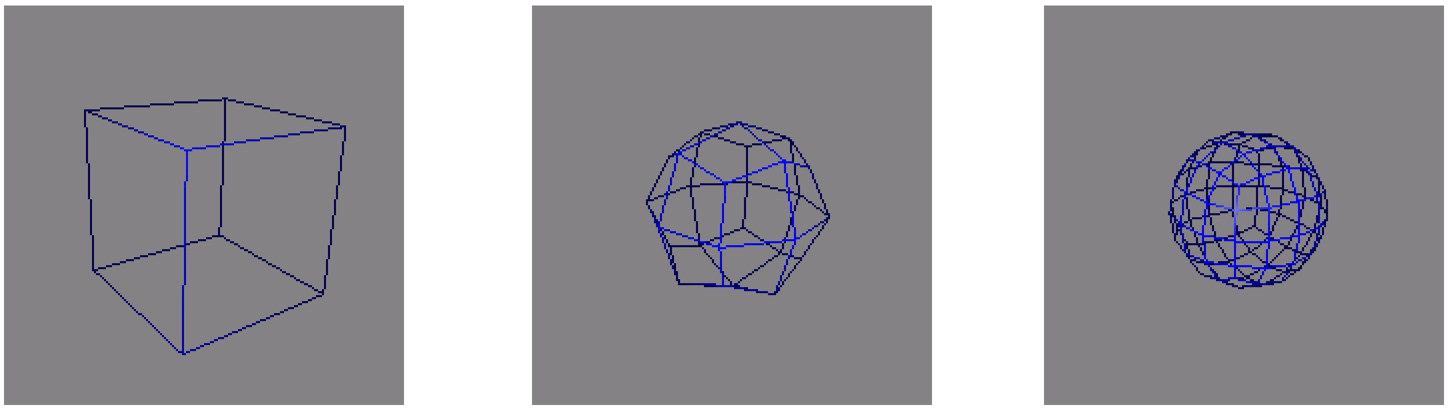
Edge point - average of 2 edge vertices and 2 new face points

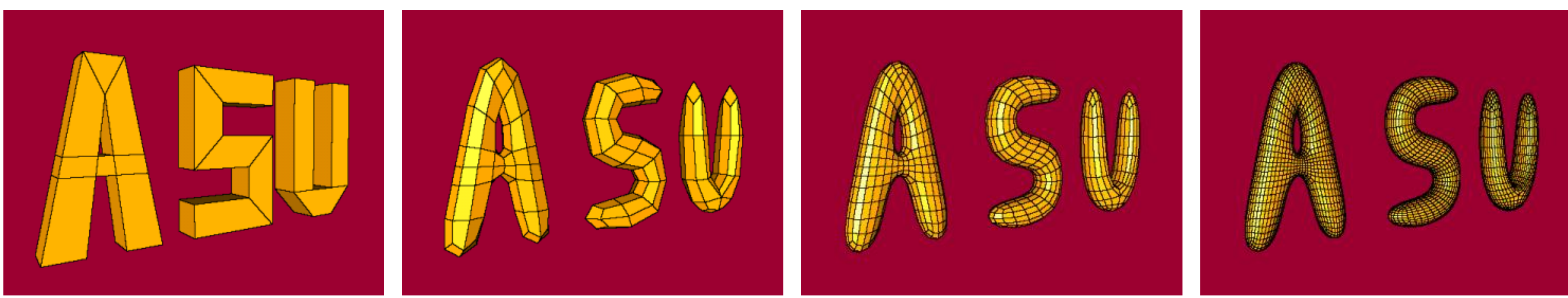
Vertex point - (n-3/n)*vertex
+ 1/n average of face points
+ 2/n midpoints of edges



Chapter 8

Catmull-Clark Subdivision

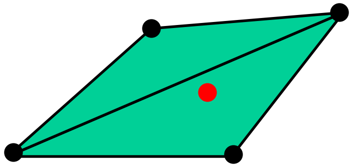




Chapter 8

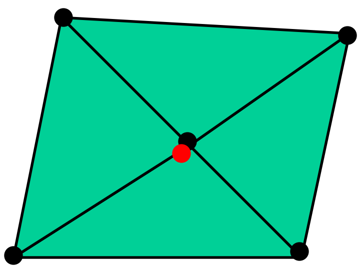
Loop Subdivision

Only works on triangles



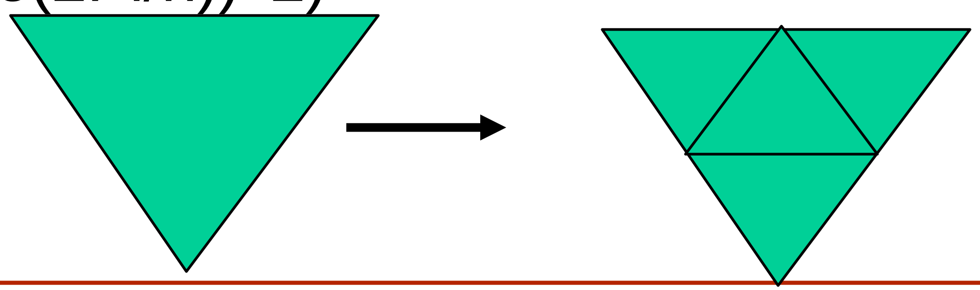
Edge point - = (3/8)*2 edge vertices + (1/8)*2 triangle vertices

Vertex point - (1-n)*s*vertex + s*(sum of neighboring vertices)



For n=3, s = 3/16

Else s = (1/n)(5/8-(3/8 + 1/4cose(2Pi/n))^2)



Chapter 8

Loop Subdivision