1996-11-29 Bo Nordin (bosse&cb.uu.se)

Lecture notes, computer graphics (F14) (Foley 9.3, 9.5, 10, FR)

Surfaces

In general, a (simple) extension of curves.
Recall:
Cubic curves (four unknowns i.e. control points)
$Q(t) = G * M * T$ where G, the geometry matrix, is a constant
Coefficients: $G * M$ (G "are" the control points, M the basis matrix)

replace t with s and allow the points in G to vary in 3D along some path parameter on t
$Q(s,t) = [G_1(t) G_2(t) G_3(t) G_4(t)] * M * S$
For a fixed t_1, $Q(s,t_1)$ is a curve because $G(t_1)$ is constant
By repeating this process for many other t_x values between 0 and 1 we get a family of curves
If G_i are cubics, the surface becomes a *parametric bicubic surface*.

The derivation of Hermite surfaces (x only) on page 352.:
G_{Hx} is a function of t, $P_{1x}(t)$ $P_{2x}(t)$ define x at start and end in *s*.
$x(s,t) = T^T M_H{}^T G_{Hx} M_H S$ (the same for y & z -> three geometry matrices)
Special cases if Q(0,t), Q(1,t), Q(s,0), S(s,1) are linear or coplanar

Normals to surfaces:
Calculate the cross-product of the two tangent vectors for s and t.
Becomes fifth-degree polynomial in 2 variables (s,t):

Displaying curves:
iterative evaluation of the bicubic polynomials
the surface equations must be evaluated about $2/d^2$ times
forward-differencing

Solid modelling (10)

Representation allows to describe curves and surfaces in 2D and 3D. Just as 2D primitives does not have to represent a closed curve, 3D primitives does not have to represent a closed volume
Must be able to encode things that look like a solid (not enough, though)
0/ large domain (many objects)
1/ unambiguous (complete) (lines to present a polyhedron - filled/unfilled)
2/ unique (a solid should be represented in exactly one way, easy equality tests)
3/ accurate (no approximation)
4/ validity, impossible to create an invalid representation and easy to create a valid representation
5/ closure (rotation, scaling, translation should produce only valid solids)
6/ compact (space, time in communication) and efficient (for measurements, display)
unevaluated vs. evaluated

Primitive instancing

Complex primitives may be defined
-Limited domain

Sweep representations (10.4)

Sweeping an object along a trajectory through space defines a new object, a *sweep*
Useful for creation of 3D objects with translational, rotational, or other symmetry.
translational sweeps
rotational sweeps
-Limited domain

Boundary representation

b-reps (vertices, edges, and faces)
-Approximation
+large domain
+many algorithms for display exist

Spatial-partitioning representations

Decompose into a collection of adjoining, non-intersecting solids (more primitive)
Spatial-occupancy enumeration (voxel, cuberille)
Failing parallel those of representing 2D shape by bitmaps. (space!!)
Octrees (divide & conquer by binary subdivision) partly filled -> recursively subdivided

-Approximation
+Can represent "everything" (large domain)
+Unique

CSG (Constructive Solid Geometry)

Uses Boolean operators (on, often simple, primitives) to create hierarchical trees. with operators in internal nodes and primitives in the leaves.
-Not unique
-Limited domain (but systems may allow creation of primitives (curves, sweeps)
+Easy to validate

User interfaces for solid modelling

Manipulation of 3D objects with 2D interaction devices
Can use multiple display windows.
VR environments
"Visual sculptor"

Creating Volume representations

Sectioning (often $\Delta z > \Delta x, \Delta y$ => approximation in z less accurate)
Image analysis in 2D (several views)

Fractal curves

Methods described so far adequate for manufactured objects
natural objects clouds, forests, mountains have irregular and fragmented features.

fractal-geometry methods where procedures rather than equations are used to model objects.
A fractal object has two basic characteristics:
infinite detail at every point and self-similarity between object parts and the overall features of the object.

DRAW

in principle: natural objects are represented with procedures repeated infinitely.
if we zoom in on a fractal object we continue to see as much detail as we did from the start.

self-similar fractals have parts that are scaled down versions of the entire object
if we apply random variations to the scaled-down subparts the fractal is said to be *statistically* self-similar.
self-affine fractals have parts that are formed with different scaling parameters in the different coordinate directions.
invariant fractal sets are formed with nonlinear transformations. includes *self-squaring* fractals formed with squaring functions in the complex space, and *self-inverse* formed with inversion procedures

Produced by functions: $P_0 = (x_0,y_0,z_0)$ $P_1 = F(P_0)$ $P_n = F(P_{n-1})$
The transformation function can be applied to a point set, or lines, curves, surfaces, solids
Applied a finite # o times (deterministic and random)

Generation of deterministic self-similar fractals
initiator - start with a given geometric shape
generators - subparts of the initiator are then replaced with a pattern (the generator)
ex: the Koch curve (length increases by 4/3 at each step -> tends to infinity.

Fractal dimension: D measure of roughness, or fragmentation

D = ln(n) / ln(1/s) s is the scale, n is numbe of segments

a line divided into two sublines 1/N, each looks like the original scaled by a factor of $N=N^1$
a square divided into N (4,9) subparts, each part looks like orig scaled by a factor of $N^{1/2}$
a cube divided into N (8) subparts, each part looks like orig scaled by a factor if $N^{1/3}$
1: ln(2)/ln(20 2. ln(9)/ln(3) 3: ln(8)/ln(2).

what about the Koch snowflake: when divided down by four, each part looks like
the original scaled down by three. $4^{1/d} = 3$ d = log(4)/log(3) = 1.26..
The length of the curve tends to infinity as more detail is added.

pg. 368 log(3) / log(sqrt(7)) log(8)/log(4), log(18)/log(6)

multiple disjoint components (for islands, for instance):
log(2) / log(3) log(16)/log(8) log(16)/log(8)

Fractal in the plane: 1 < d < 2 (typically, if > 2, the curve self-intersects) d==2 Peano, fills
Fractal in the space: 1 < d < 3
Fractal sutfaces: 2 < d < 3 (if > 3 overlapping), d == 3, fills the space
Fractal solids: usually in the range 3 < d <= 4. (if d > 4, self-overlapping)

example of self-similar construction for a surface:
a regular tetraeder (scale by 2
each face is converted to six smaller faces (face area is increased by 3/2 log(6)/log(2)=2.58

random: midpoint
twist
Brownian motion
starting from a position we generate a straight-line segment in a random direction and with a random length. Then move to the endpoint and repeat the process
Fractional Brownian motion is obtained by adding another parameter (fractal dimension)
Calculating fractional Brownian motion is time-consuming (Fourier FFT)

random midpoint-displacement: fig page 374

ymid = (y(a) + y(b))/2 + r where r is $N(0,k|b-a|^{2H})$ where H = 2 - D and D > 1 is the fractal dimension

or one can take $r = sr_g|b-a|$ where s i a roughness factor and r_g is N(0,1)
repeat until line sections are short

terrain features by applying the random midpoint-displacement technique to a rectangular ground plane
triangulation

*self-squaring fractal*s

complex number that is squared: -> infinity, -> attractor (0), or remain on boundary of an object
for some functions, the boundary between the points is a fractal (*Julia set*)
we can find the bounday either by iteratively search or by inverting the function

a very famous shape is obtained from the Mandelbrot set, the set of z which do not diverge under $z_o = z$, $z_k = z^2_{k-1} + z_0$