

CS7491 Class Notes

for date 1/22/2004

Topic: Simplification of Curves

Algorithm 1:

- Repeat $n/2$ times
 - Remove vertex v_i that is closest to the line (v_{i-1}, v_{i+1})

Problems:

- Error is memory-less: the error accumulates and is unbounded
- Result can self intersect
 - Fix: detect new intersection and undo the vertex removal

Algorithm 2:

- Repeat $n/2$ times
 - Remove vertex v_i for which the area of triangle (v_{i-1}, v_i, v_{i+1}) is minimal

Problems: Same as in algorithm 1

Algorithm 2':

- Repeat $n/2$ times
 - Remove vertex v_i for which the difference of area between original curve and simplified curve is minimal

Is this good or bad? First, let's consider the computation of the area.

Side Topic: Polygon Area Computation

Assume a polygon $V = v_0v_1 \dots v_nv_0$.

There are two tools to consider that are at our disposal.

Tool 1 (old): Cross Product

Consider the quantity $(\sum_i (o v_i \times o v_{i+1})) / 2$ for some arbitrary point o .

The magnitude of that vector is the area. The vector is useful to keep around because the direction provides a concept of positive and negative areas. If we want to change the curve, a change to the vector is much easier. Otherwise, this summation must be completely recalculated instead of adjusted.

Tool 2 (new): Trapezoid Area

The area under each segment (a,b) of the curve can be calculated with the equation:

$$(y_a + y_b)(x_b - x_a) / 2$$

The overall area of the curve thus becomes:

$$(\sum_i (y_i + y_{i+1})(x_{i+1} - x_i)) / 2$$

The area is the absolute value of that quantity. However, the sign is useful. How?

Side Note: Area of a polygon in 3-D Space

Let A_x, A_y, A_z be the vectors calculated by the cross product method of the polygons projected onto the yz, xz and xy planes respectively.

Let $N = (A_x, A_y, A_z)$

$\|N\|$ = area of the 3-D polygon

Back to Algorithm 2':

Problem(?):

Could algorithm 2' "displace" the area, simply shifting the curve's pointset?

A counter example is harder to make. Consider a curve with convex side and a concave side. As area is removed from the convex side, it is added to the concave side. This might be avoided by only performing 2' on convex polygons?

Algorithm 3:

- Repeat $n/2$ times
 - Remove a vertex such that the smooth curve obtained by the subdivision of the resulting simplified curve is the closest to (in terms of the Hausdorff distance) to the original curve

Side Topic: Subdivision of Closed Polygons

Subdivision Algorithm

- For each v_i compute $D_i = ((v_{i+1} + v_{i-1}) / 2 - v_i) / 2$
- Split each edge $v_i v_{i+1}$ at its middle and insert vertex $m_{i,i+1}$

If v_i moves to $v_i + D_i$ with repetition, then the whole process will result in a cubic B-Spline curve at the limit.

Shrinkage occurs though...

Subdivision Algorithm (tweaked)

- For each v_i compute $D_i = ((v_{i+1} - v_{i-1}) / 2 - v_i) / 2$
- Split each edge $v_i v_{i+1}$ at its middle and insert vertex $m_{i,i+1}$
- For each new vertex $m_{i,i+1}$, move it by $-s((D_i + D_{i+1})/2)$
- For each old vertex v_i , move it by $(1-s)D_i$

Consider the possible values of s from 0 to 1:

$s=0 \Rightarrow$ cubic B-Spline

$s=1 \Rightarrow$ 4-point subdivision

$s = 1/2 \Rightarrow$ Jarek's Curve

What is nice about Jarek's curve?

It bulges edges and brings vertices in. This is a nice blend between B-Spline and 4-point.

...back to Algorithm 3

It's an expensive greedy algorithm. For each single vertex to be removed, you must test each vertex on the curve by performing removal, subdivision, and a Hausdorff distance calculation.

NOTE: Look at the paper on the website!

Another approach: Consider polygon vertices to be circular balls of some determined radius. Stab the maximum number of circular balls in order with a circular arc. This is a form of extrapolation instead of interpolation.

Algorithm:

- Start with 3 balls (always possible to stab three balls with an arc)
- Expand the arc to stab the balls in order until a ball along the way is not stabbed

Why is this good? The area enclosed by the circle and the chord between arc end-points bounds the error.

Hint: We don't have to use arcs. More complex equations such as cubics, quadrics, etc. could be used instead.

New Topic: Wavelets

Algorithm

- Remove every other vertex on the curve. Do we lose this information? No.
- Encode the residues between the position and the guess (try using 4-point subdivision at the first level)
- Repeat until all vertices are gone or only one vertex left.

Last layer of points is typically twice better at having small errors than other layers. This can be seen as including elements of loss-less compression into simplification.

Corrections:

The formula for the i th displacement in subdivision was corrected James Vanderhyde.