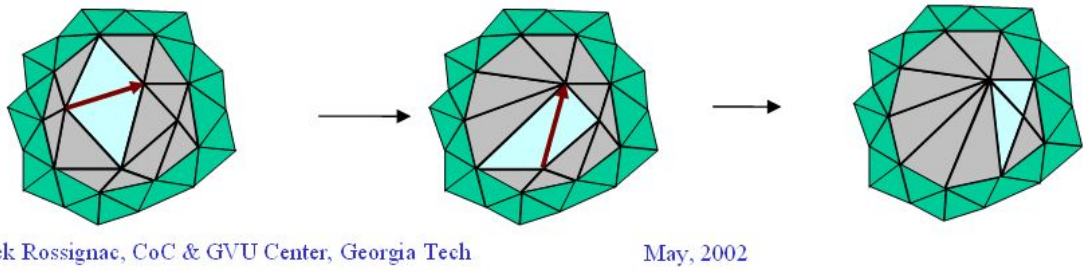
Professor Jarek Rossignac
cs7491

Class Notes 2006-3-30

Overview of Covered Material

Simplification

- Vertex clustering
- Edge collapse
- Resampling
- Multiresolution (LOD, mipmapping, etc)
    - view dependent (closer objects have more detail than distant objects)



An edge collapse example

Compression

- Quantization ("snapping" vertexes to a grid)
- Prediction (linear and high-order, parallelogram prediction)
- Entropy codes
- Connectivity encoding (edgebreaker)

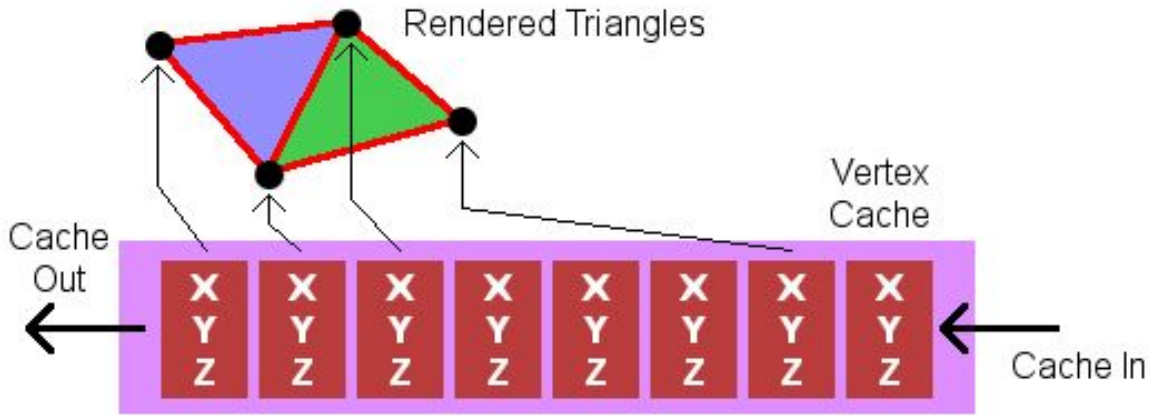
Refinement

- Refinement is the inverse of simplification, adding detail
    - hopefully, the detail we want
- Compressed refinements
- Dependency (allows for "cascading")
- Batches (economy of scale)
    - view dependent

Streaming

- Consider the problem of transmitting a triangle mesh to a client with limited vertex cache space of *n* vertexes
- When sending a mesh, we'd like to send each vertex to the client only once. The client can render a triangle only if all three vertexes are in the cache. Is this possible given *n*

and the number of vertexes?



A conceptual drawing of the vertex buffer problem

- Using triangle strips, each vertex will need to be sent twice if the strips are long
- Still an open question

Linear Points Compression Problem

Input: A one-dimensional line of length $2^{20} - 1$, and 2^{10} points on that line not regularly spaced. The distance between the points can vary between 0 and 2^{11} inclusive.

Objective: Losslessly encode this input.

Solution 1
- No compression (repeatedly send the distance from the previous point to the next)
- 11 bits per point (1st point may be special case, however)

Solution 2
- Linear prediction (given two points in a row, A and B, guess that dist(B, C) == dist (A,B) and then send the correction distance to get C)
- guaranteed 12 bits per point (11 + one sign bit)
- expected 6 bits per point

Solution 3
- Use a Binary Space Partition (BSP)
- BSPs are among the most pervasive storage structures in all of computer graphics, both in 2D and 3D
- Recursively slit the interval in two. At each iteration, record the number of points that are present in each interval.

A BSP of the input after four recursive steps. The lighter numbers are redundant and do not need to be sent

- After 20 steps, each interval will be of size one
- If at any time an interval is discovered to have zero points, that branch of the tree is not continued (no need to)
- Best case is all points at same location, worst case is an even distribution
- Distribution model is not trivial