A Z-Buffer CSG Rendering Algorithm for Convex Objects

Nigel Stewart, Geoff Leach and Sabu John
RMIT University, Australia

WSCG 2000, Plzen, Czech Republic

crc IMST

Z-Buffer CSG Rendering

Use z-buffer hardware to solve visible surface problem for CSG trees.

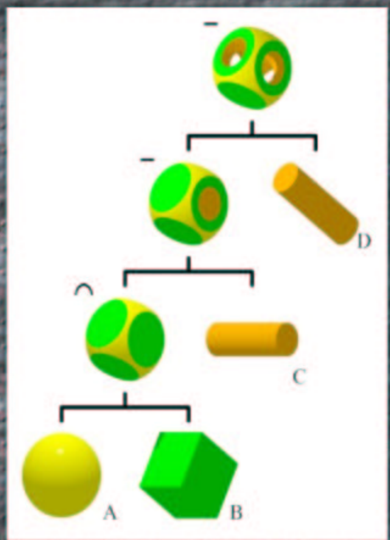
Trickle Algorithm
Subtract volumes from front to back using four z-buffers.

Goldfeather Algorithm
Clip one surface at a time in the z-buffer.
Merge the result into second z-buffer.

Constructive Solid Geometry (CSG)

Leaf Nodes - objects
Parent Nodes - operations

∪ Boolean Union
∩ Boolean Intersection
- Boolean Difference

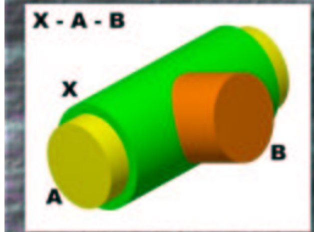


A new approach to CSG Rendering

Sequenced Convex Subtraction (SCS)

- Use convex objects.
- Subtract volume progressively.
- Minimise the number of z-buffers.
- Minimise Z-buffer copying.
- Avoid sorting surfaces in the z-buffer.
- Use subtraction sequence that works for any viewing direction.

SCS in Operation



CSG Tree: X - A - B

Sequence Abstraction

For n objects, use a sequence embedding all n! permutations of n volumes along a line.

```
For n=2,  aba
          ab*
          *ba

For n=3,  abcbabc
          abc****    a*cb***
          *b**a*c    *bc*a**
          **c*ab*    **cba**
```

SCS Subtraction Sequence

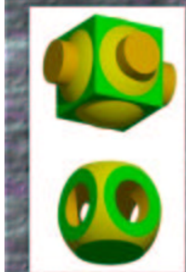
- Z-buffer can store one surface only
- Z-buffer can not store holes behind the current surface.
- Volumes may need to be subtracted more than once.
- Sequence depends on the way volumes interact.

If A is in front of B, subtract A first
If B is in front of A, subtract B first

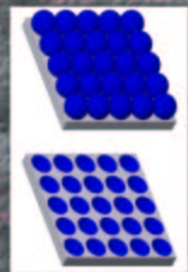
To avoid sorting, subtract in sequence ABA

SCS Subtraction Sequences

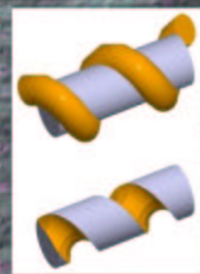
Subtraction Sequences are O(n^2)
With depth-complexity, O(kn)



Widget
n=4, k=4
abcdcbabcdcba



Grid
n=20
k=2, SCS length 39



Machine Tool
n=20
k=5, SCS length 96

Conclusion

Sequenced Convex Subtraction Algorithm (SCS)

- No sorting.
- No z-buffer copying for CSG product.
- Two z-buffers required for CSG tree.
- Less z-buffer copying than other algorithms.
- Requires convex volumetric representation.
- Uses a sequence based on depth complexity.

Graphics Hardware Implications
- Higher rasterisation load, lower z-buffer load.
- Particularly advantagous on bandwidth limited hardware. (Nvidia TNT2, for example)

Further Work

Other algorithms make use of seperability information.

How to generate sequences based on this information?

Optimal subtraction sequences.
(Better than O(n^2)?)

Fairer benchmark between SCS and Goldfeather.

Industrial Application.