Linear-time CSG Rendering of Intersected Convex Objects

Nigel Stewart, Geoff Leach and Sabu John

RMIT School of Computer Science and Information Technology

WSCG 2002 - Pilzen, Czech Republic
February 2002

RMIT UNIVERSITY

crc IMST

Graphics Hardware Trends

- Faster polygon rasterisation
- Higher bandwidth
- Larger amounts of memory
- Pipeline Parallelisation & Graphics Clustering

Objective: Hardware-based algorithms for CSG rendering.
Context: Dynamic and interactive applications.

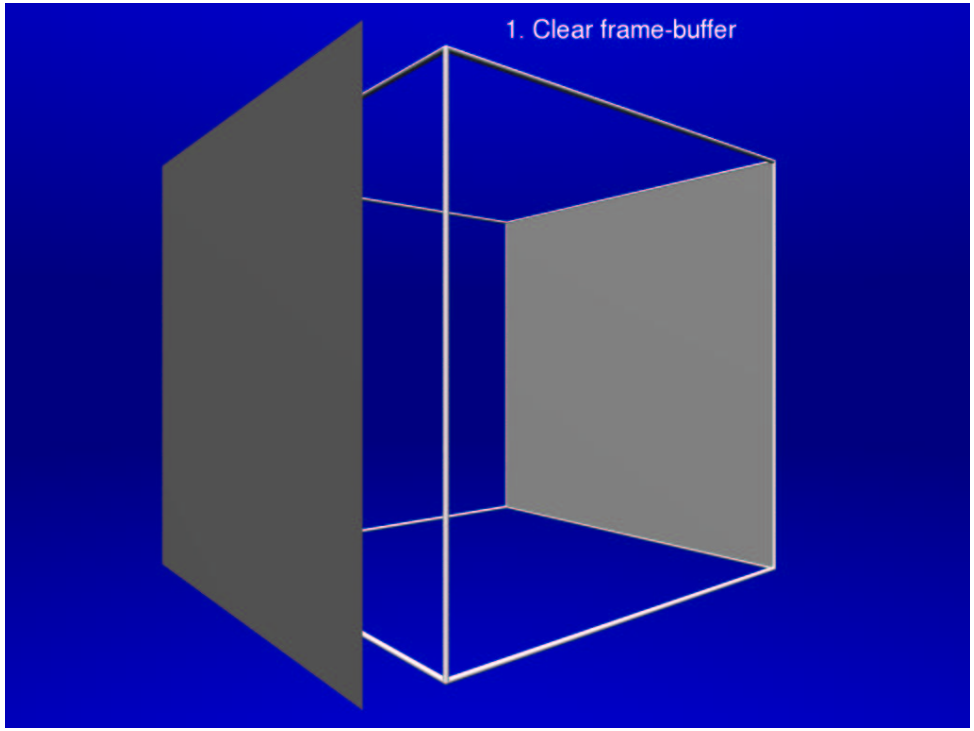
Overview

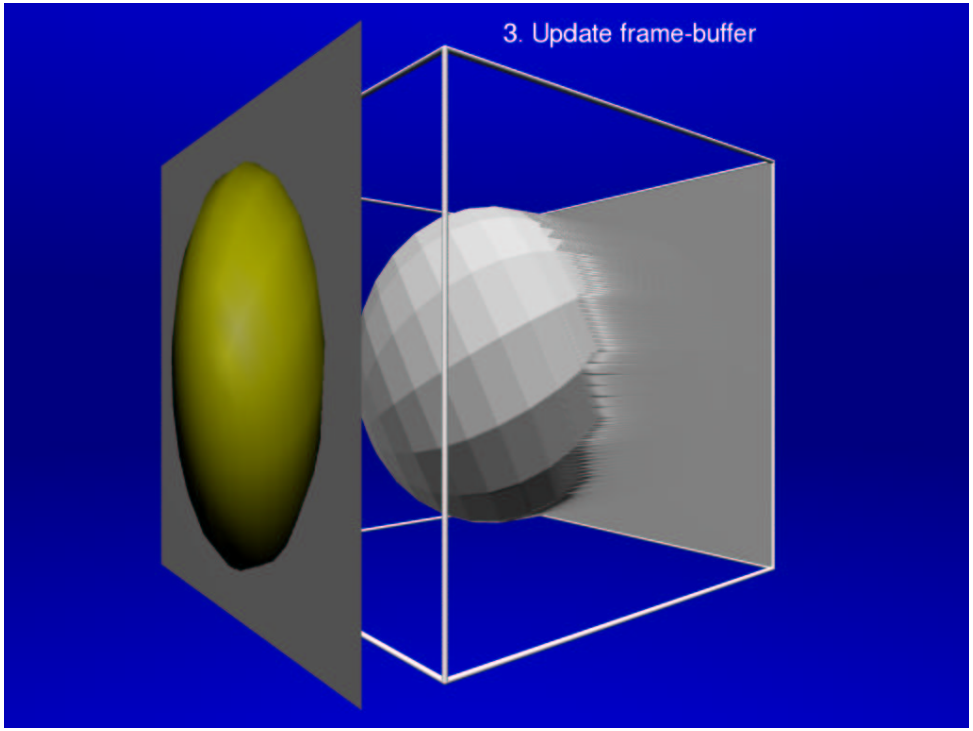
- Graphics Hardware Trends
- Z-Buffer
- Z-Buffer Intersection
- SCS CSG Algorithm
- Performance
- Conclusion & Future Work
- Live Demo
- MPEG Movies

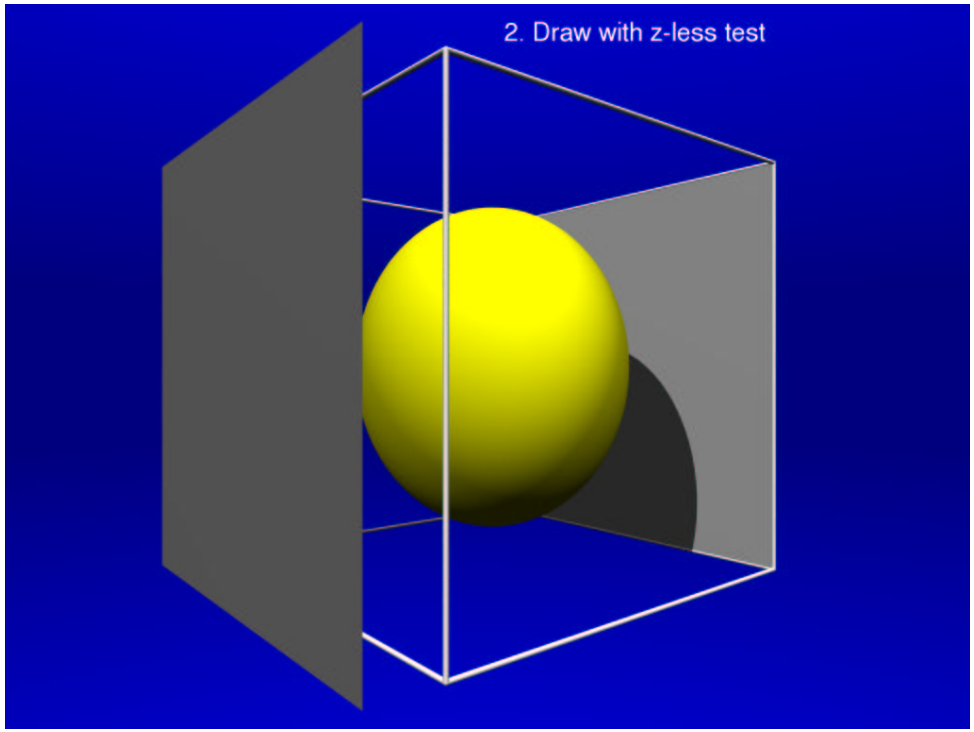
Z-Buffer

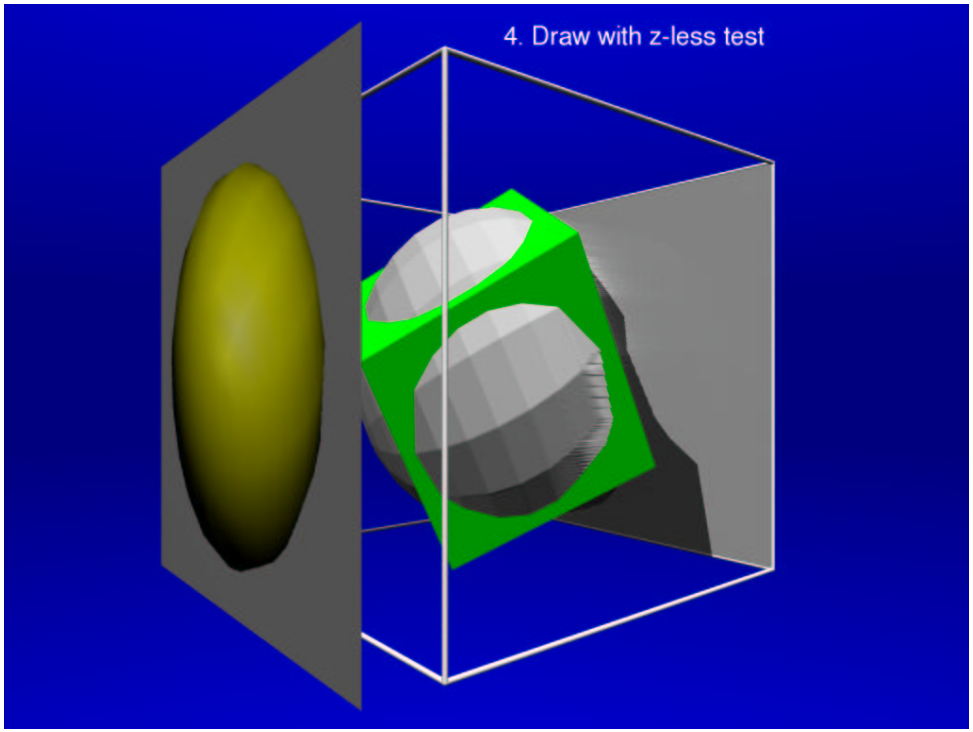
- Hardware-based solution to visible-surface problem.
- Implied volumetric union.
- Store *z* for each pixel in graphics hardware.
- Configurable z-test as part of pipeline.

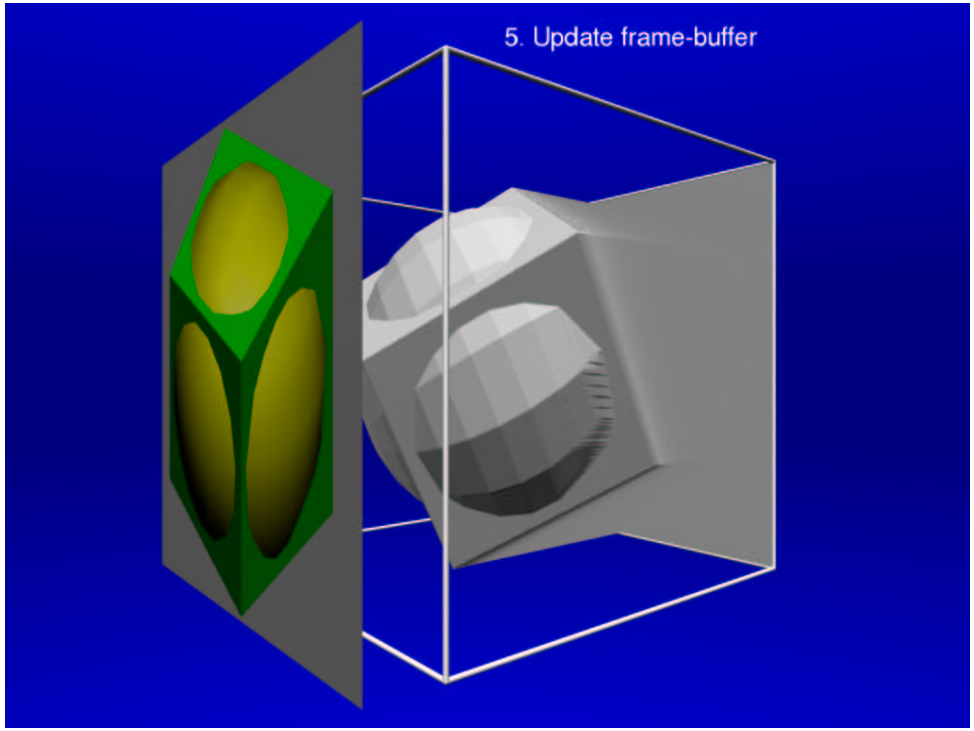
1. Initialise the z-buffer to Z_{far}
2. Only draw pixels closer than z-buffer.
3. Update z-buffer when z-test passes.

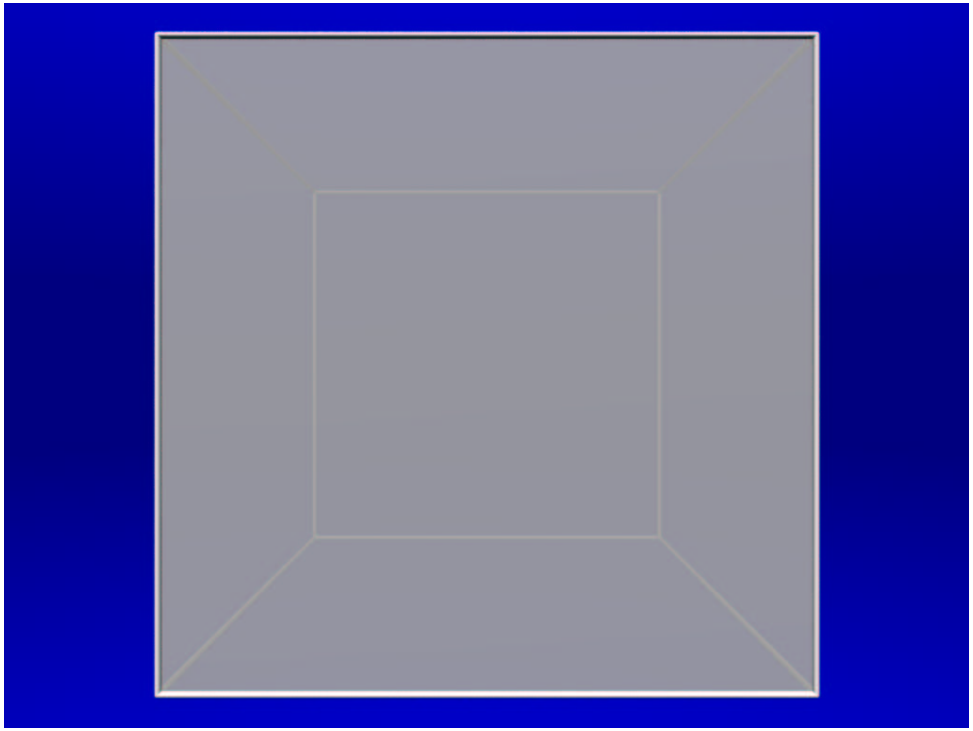

1. Clear frame-buffer

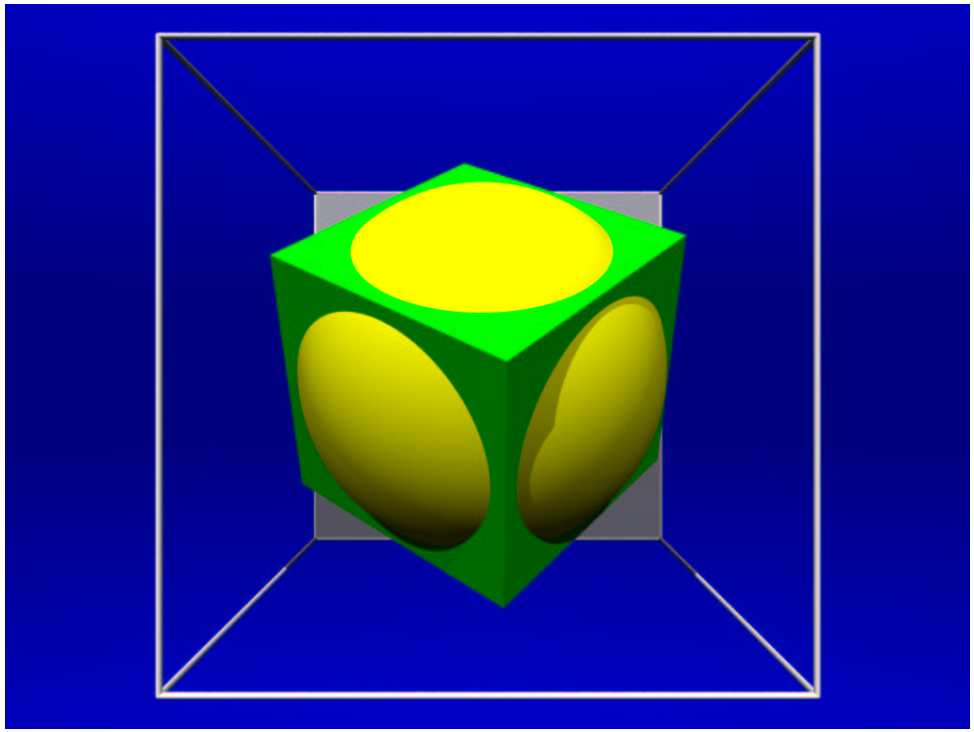

3. Update frame-buffer

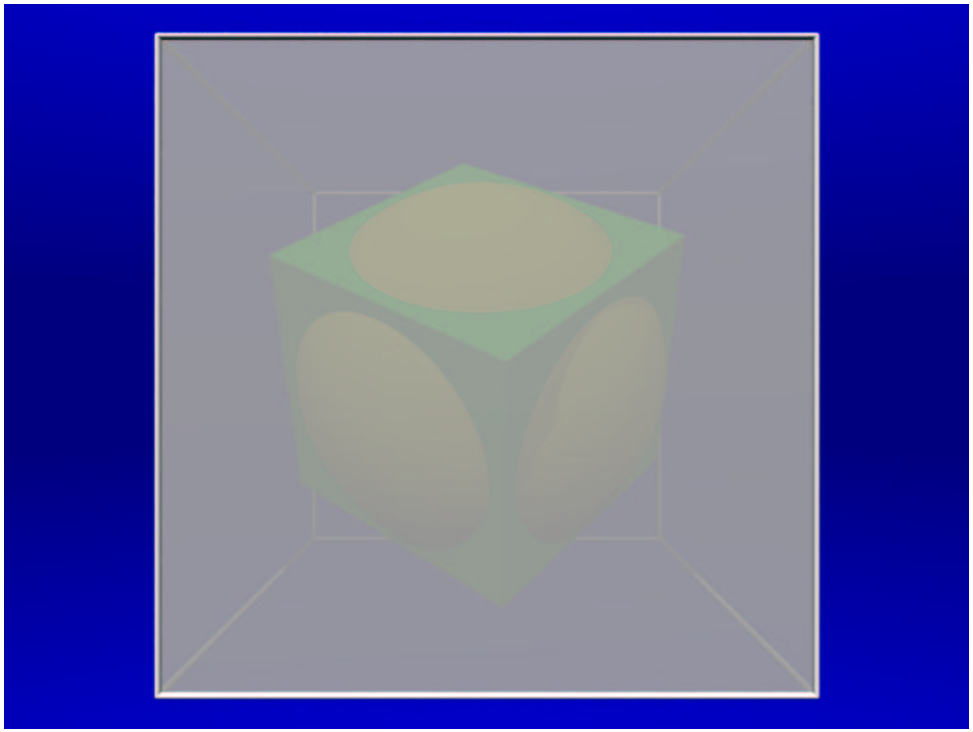

2. Draw with z-less test

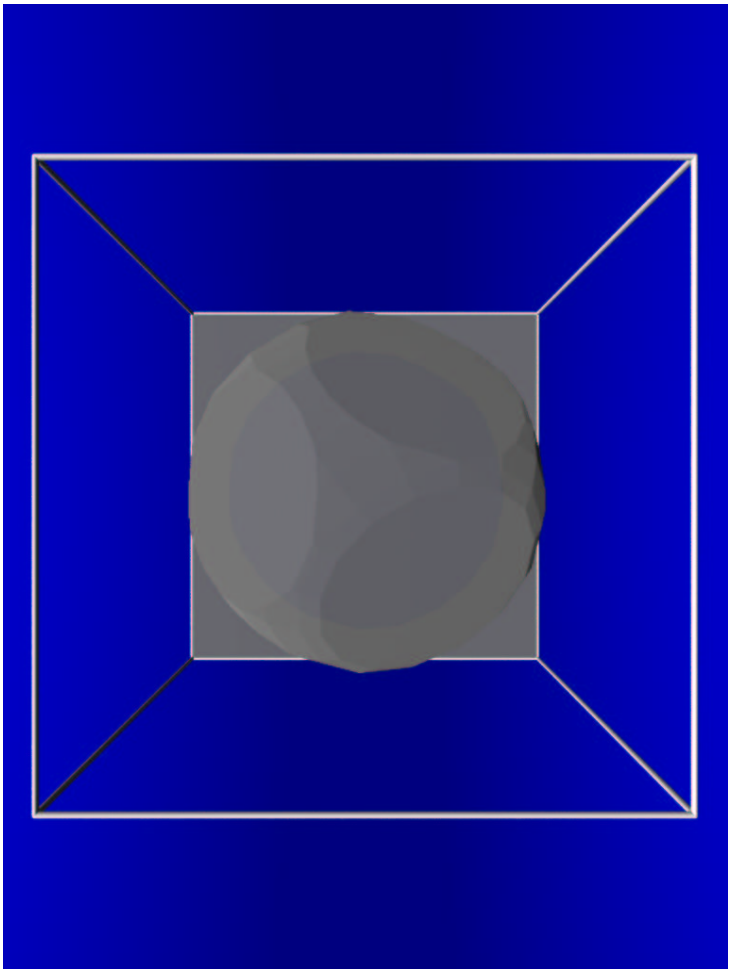

4. Draw with z-less test

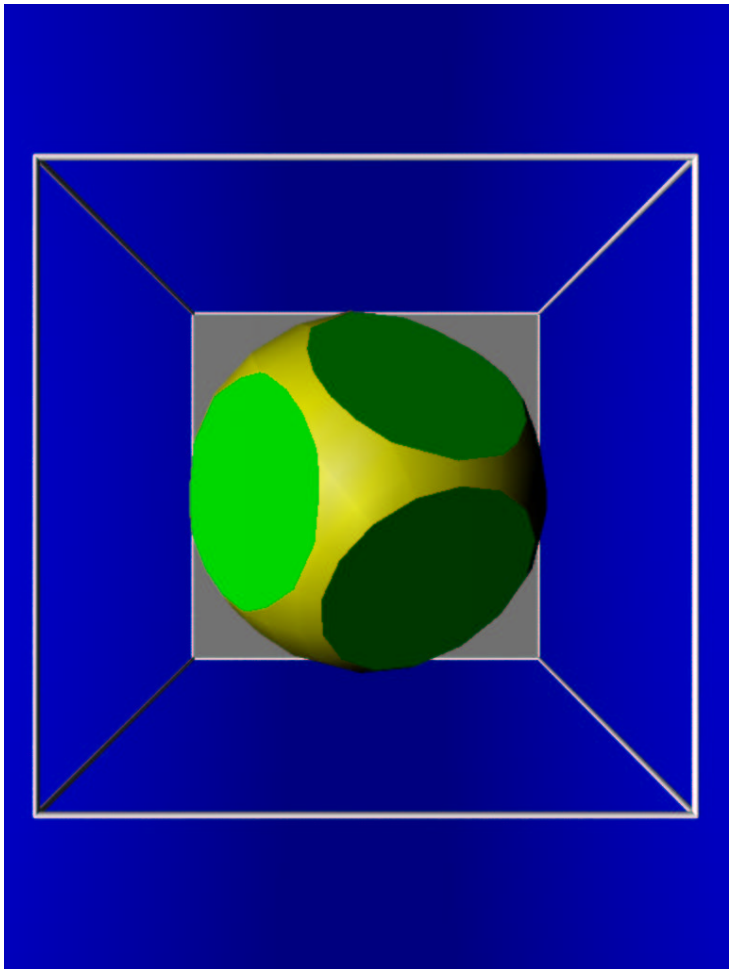

5. Update frame-buffer

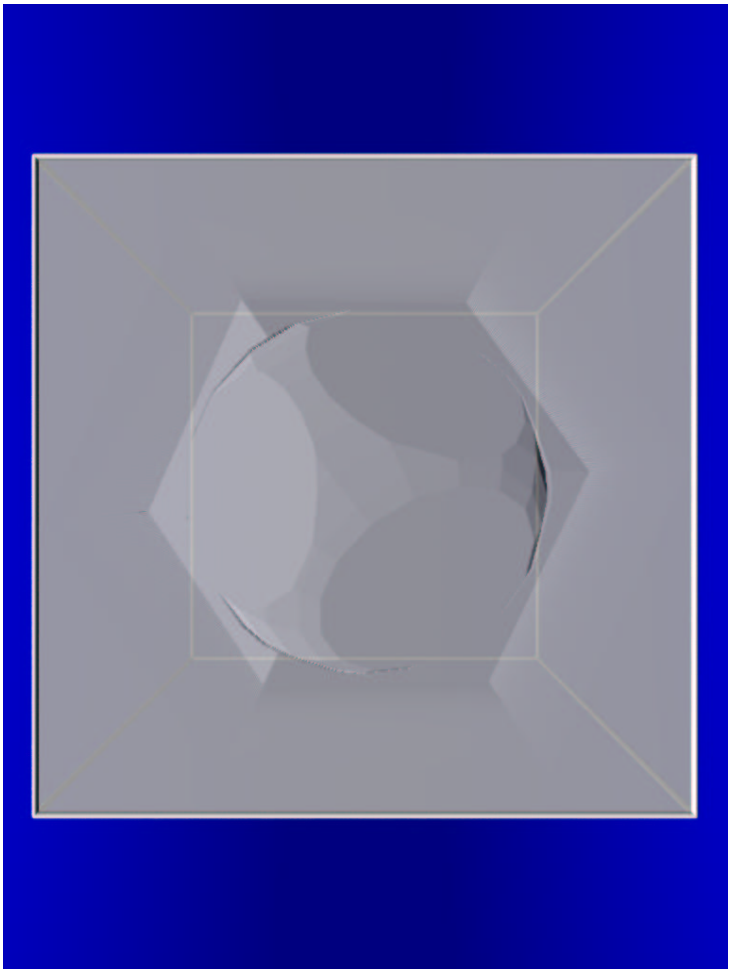


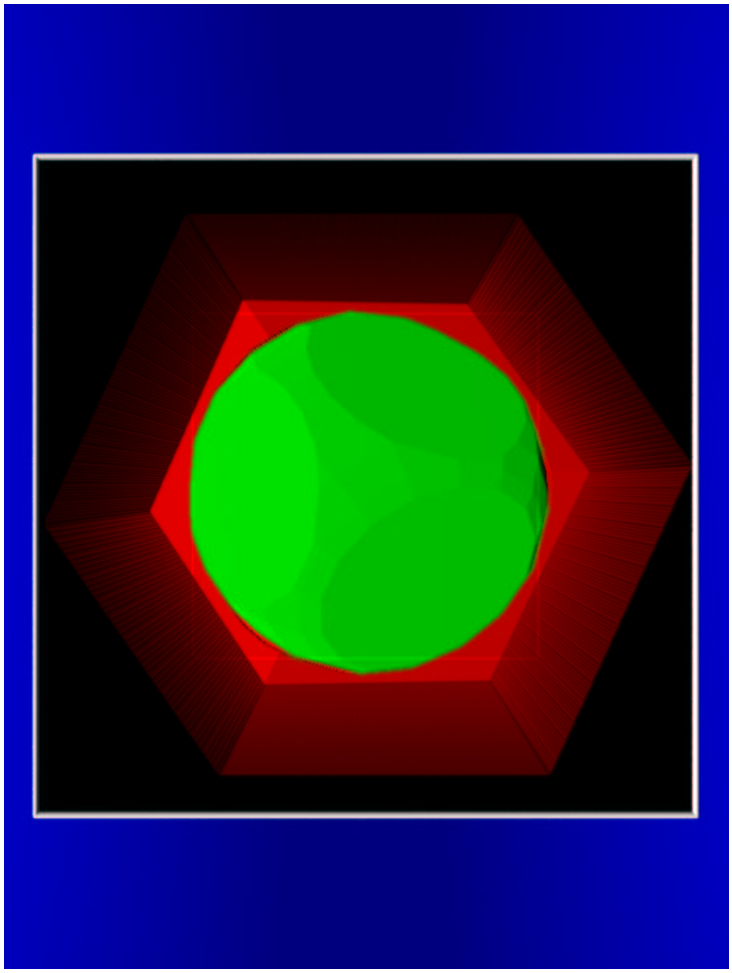












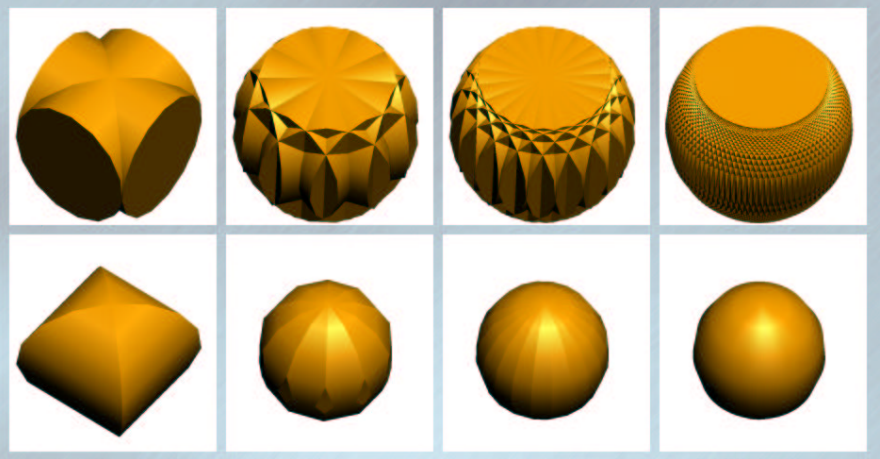
Z-Buffer Intersection

·Algorithm for rendering the intersection of convex objects.

·Use the z-buffer and stencil buffer.

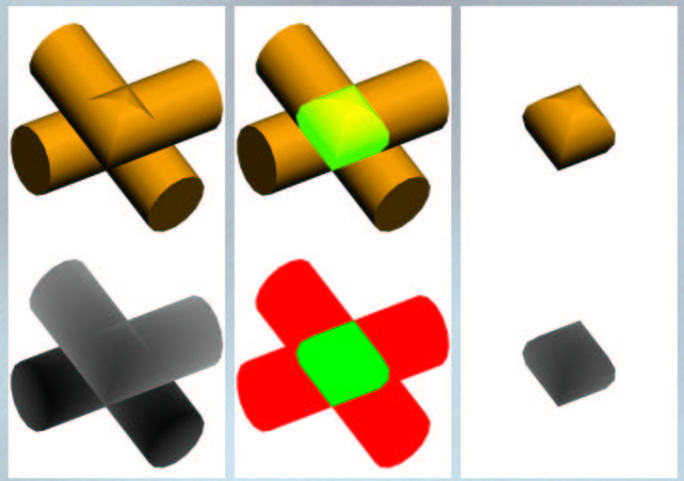
1. Initialise the z-buffer to Z_{near}
2. Draw font-facing surfaces with *z-greater* test
3. Count back-facing surfaces behind z-buffer
4. Reset pixels where *stencil != n*
5. Draw front-facing surfaces with z-equal test

SCS-Intersection Results



2 Cylinders | 5 Cylinders | 10 Cylinders | 50 Cylinders

SCS-Intersection Illustration



Furthest front-face | Count back-facing behind | Reset pixels

SCS CSG Algorithm

Algorithm for rendering CSG trees of convex objects.

1. Convert CSG tree to sum-of-products
2. For each product:
   - 2.1 Resolve intersections in z-buffer
   - 2.2 Resolve subtractions in z-buffer
   - 2.3 Merge z-buffer with accumulated result
3. Draw visible surfaces with *z-equal* test

Performance

·Convex intersection resolved in *O(n)* time.

·Convex subtraction still requires O(n^2) time.

·Speedup depends on relative number of intersections and subtractions.

·Z-buffer copying still a major bottleneck for multiple CSG products.

Future Work

·Improve performance for convex subtraction.

·Real-world applications: 5-axis NC verification.

·Models for deeper frame buffers in OpenGL 2.0

Conclusion

·Convex Intersection rendered in linear time on standard OpenGL graphics hardware.

·SCS-Intersect utilised by SCS algorithm for general CSG rendering.

·Interactive frame rates on consumer-grade graphics hardware.