Accelerating Ray Tracing

Ray Tracing Acceleration Techniques

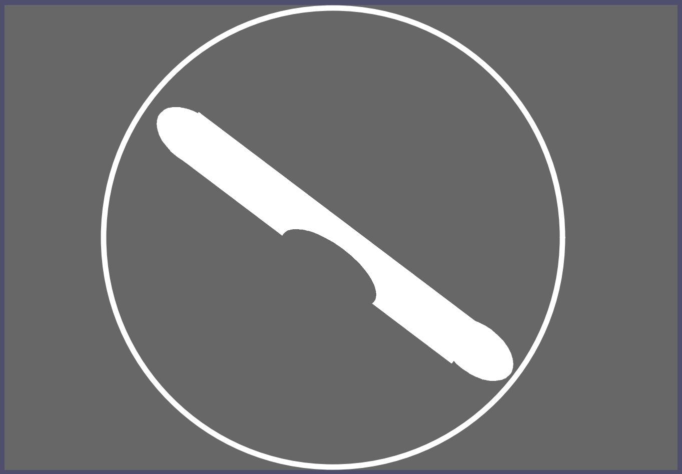
- Faster Intersections
  - Faster Ray-Object Intersections
    - • Object bounding volumes
    - • Efficient intersection routines
  - Fewer Ray-Object Intersections
    - • Bounding volume hierarchies
    - • Frustum culling
    - • Space subdivision
    - • Directional techniques
- Fewer Rays
  - • Adaptive tree depth
  - • Statistical optimizations
- Generalized Rays
  - • Beam tracing
  - • Cone tracing
  - • Pencil tracing

from Arvo and Kirk, “A Survey of Ray-tracing Acceleration Techniques”. In Glassner, ed., *An Introduction to Ray Tracing*

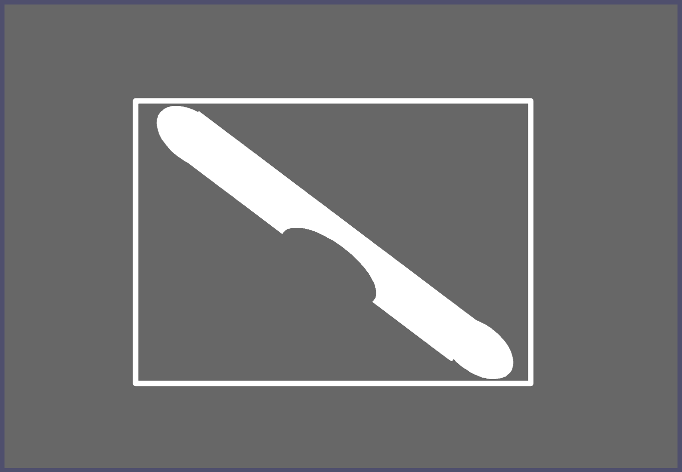
Bounding Volumes

- Simple volume description guaranteed to contain a more complex volume description
- Test ray against more complex primitive only if it intersects bounding volume
- Increases time for hits, but reduces time when ray misses bounding volume
- Provide bounds on interval of intersection

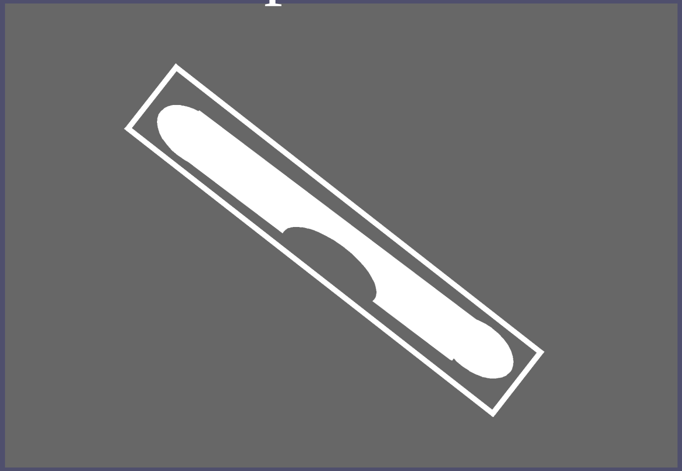
Bounding Volume Examples



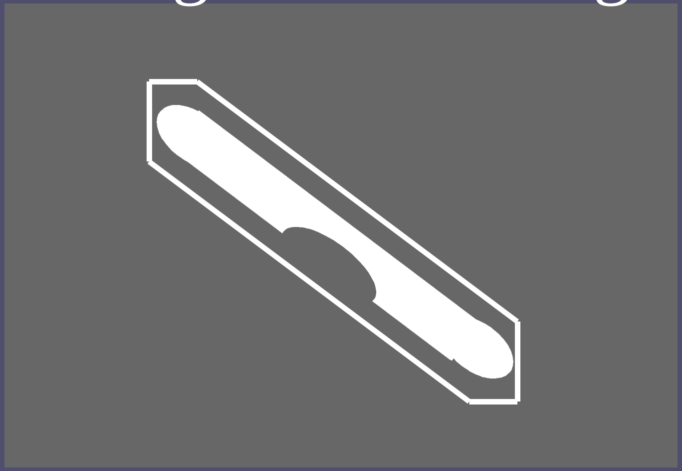
Sphere



Axis-Aligned Bounding Box



Oriented Bounding Box

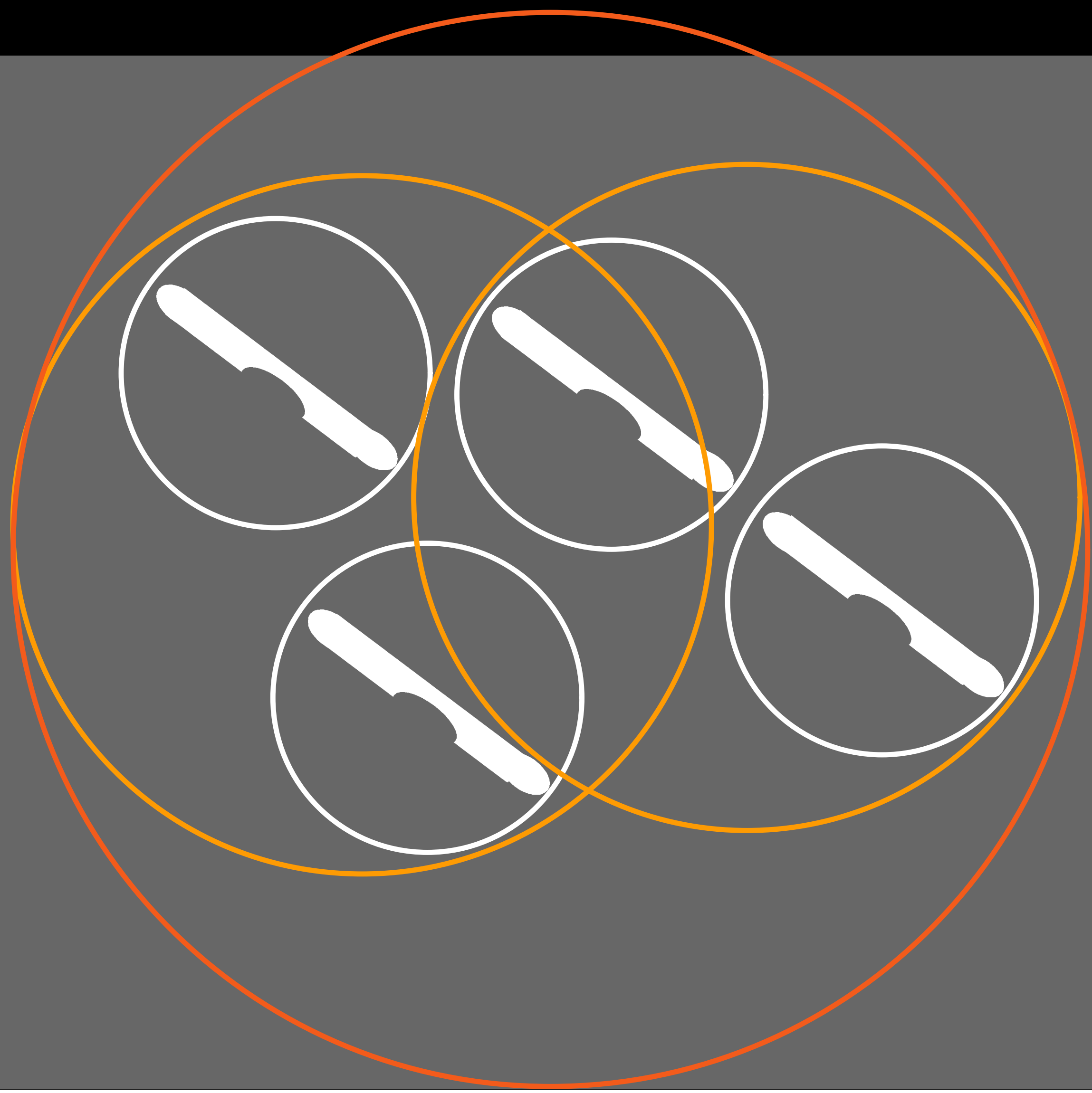


General Slab Intersection

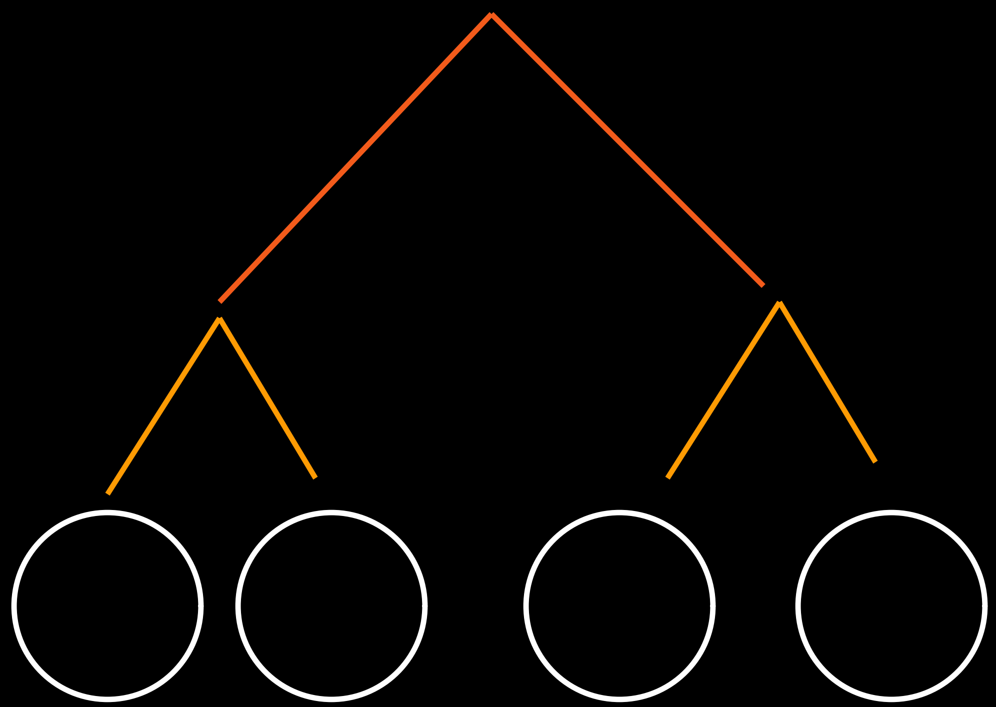
Bounding Volume Hierarchies

- Cluster bounding volumes hierarchically
- Only intersect ray with child volume if it intersects parent
- Reduces number of ray-volume and ray-object intersection tests

Bounding Volume Hierarchy Example



Bounding Volume Tree



Computing Hierarchies

Top-down

Bottom-up

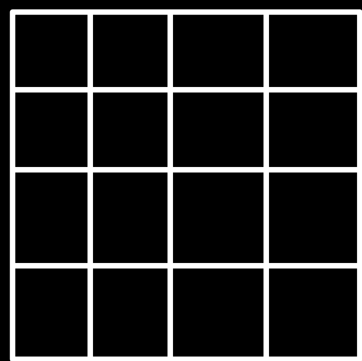
Minimize volume/surface area

Computing “good” hierarchies is difficult

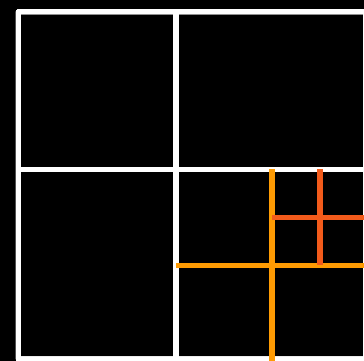
Space Partitioning

- Break model space into chunks
- Pre-compute which objects overlap each chunk
- Trace rays through chunks
- Only intersect rays with objects stored in current chunk
- Typically only allow each ray-object intersection once (not in multiple chunks)

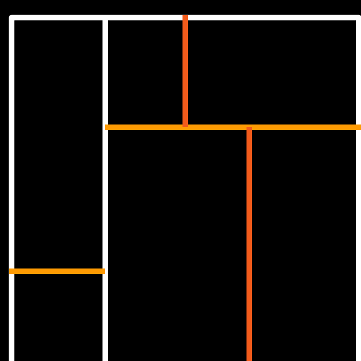
Types of Space Partitions



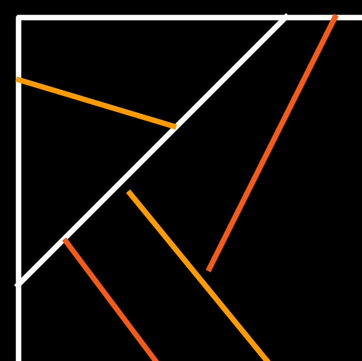
Uniform



Octree



K-d tree



BSP Tree

Tracing through a Space Partition

Incrementally

- trace along ray, walk from partition to partition
- difficult except for uniform partition
- other partitions may require augmented data structures

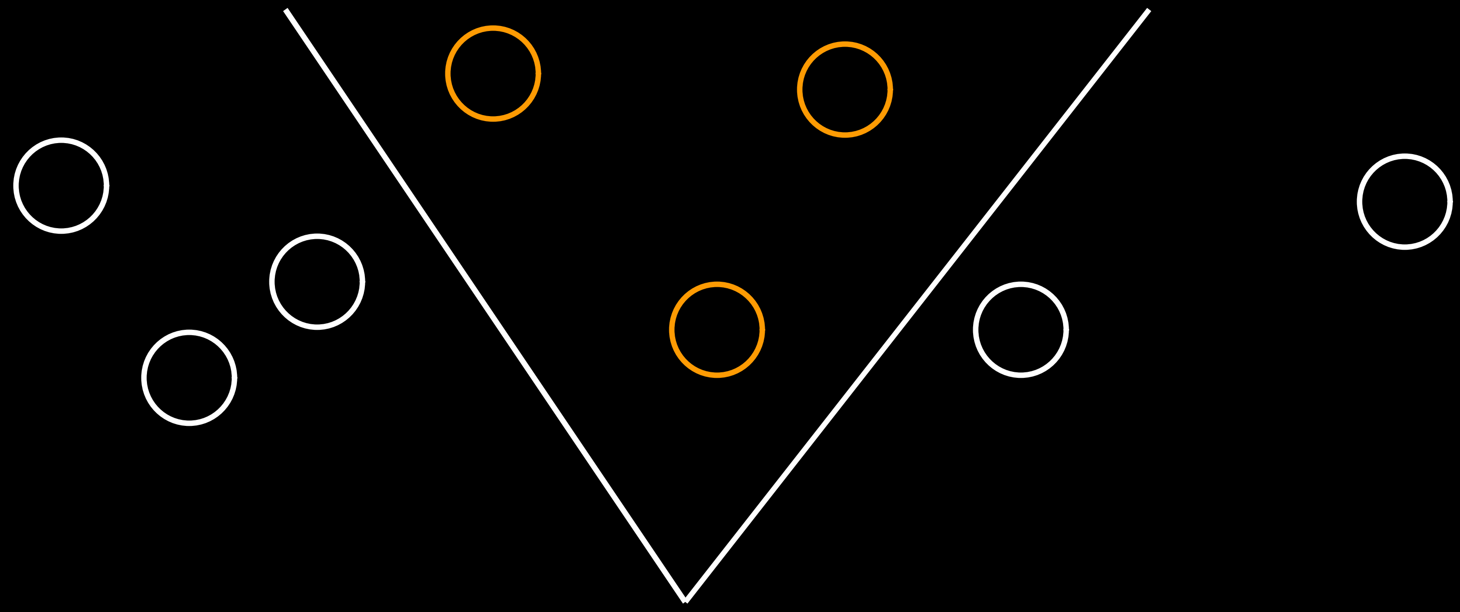
Top-down

- intersect ray with each of node’s children
- traverse children it intersects

Frustum Culling

Cull all bounding volume nodes that lies outside current viewing frustum

Probably not necessary if using space partition



Frameless Rendering

Double buffering - wait for all pixels to display image

- Slowly displays old images

Frameless rendering - display all pixels immediately

- Quickly displays partially-updated images

Applicable to interactive applications requiring low latency and high performance

Frameless Rendering Example

All pixels updated at 5 Hz

33% pixels updated at 15 Hz

From Bishop et al., “Frameless Rendering: Double Buffering Considered Harmful,” *Proceedings of SIGGRAPH 94*, page 176.

Interactive Ray Tracing

Experiments run on high-end parallel machine

- Scales to hundreds of processors on SGI Origin 2000
- Scalability limited only by load balancing and synchronization

Can employ frameless rendering

Interactive Ray Tracing Video

From Parker et al., "Interactive Ray Tracing," ***Proceedings of the 1999 Symposium on Interactive 3D Graphics*****".**

Questions about Assignment?

Input colors [0.0-1.0]

- **Output colors [0-255]**

Eye space

- **Looking down -Z axis**

Viewing plane

- **Distance is irrelevant**

File format

- **Assume lines not broken up**