CS 563 Advanced Topics in Computer Graphics
Ray-Object Intersections

by Nik Deapen

Shooting Rays

- Bounding Boxes
- Simple Objects
- Generic Objects
- Part Objects
- Compound Objects
- More?

Bounding Boxes

- Saves Computational Time
- Idea
  - Put the object in a box
  - If the ray doesn’t hit the box, it cant hit the object
- The object must be entirely contiained within the box
- The box must be axis aligned for the speedup to be greatest

Bounding Box

- Box Representation
  - Bottom Left = Pl = (x1,y1,z1)
  - Bottom Right = Pr = (x2,y2,z2)
- Code to Compute Hit or Miss
  - Page 358 Ray Tracing From the Ground Up
- Software Deign Considerations

```
public class ConcreteObject extends GeometricObject {
        private BoundingBox box;
        public boolean hit(Ray ray){
                if (!box.hit(ray))
                        return;
                …
        }
        @Override public boolean getBoundingBox(){return box;}
}
```

Simple Objects

- Planes
- Spheres
- Triangles
- Disks

Plane

- Equation of a Plane
  - (p – a) * n = 0
- Drop in the Equation of A Ray
  - (o + td – a) * n = 0
  - t = (a – o) n / (d*n)
- Linear Equation (at + b = 0)
  - a = d * n
  - b = - n (a – o)
- Solve for t
  - t = -b/a   (a != 0)
- Normal
  - Given in Plane Definition

Sphere

- Equation of a Sphere
  - $(p - c)*(p - c) - r^2 = 0$
- Drop in the Equation of A Ray
  - $(o + td - c) * (o + td - c) - r^2$
- Gives Quadratic Equation ($at^2 + bt + c = 0$)
  - $a = d*d$
  - $b = 2\,(o - c) * d$
  - $c = (o - c) * (o - c) - r^2$
- Can have 0, 1 or 2 Solutions for t
- Normal
  - $(o-p)$

Triangle

- Defined as three points (q,r,s)
  - points must not be colinear
- Normal
  - (q – r) x (s – r)
- Barycentric Coordinates
  - p(a,b,y) = qa + rb + sy
    - a + b + y = 1
  - For inside the triangle
    - (a,b,y) □ $(0,1)^3$
  - Substituting (a = 1 – b – y) gives
    - p(a, b, y) = q + b (r – q) + y (s – q)
    - (b,y,b+y) □ $(0,1)^3$

Triangle

- Hitting The Triangle
  - $o + td = q + b(r - q)\ y(s - q)$
  - $b(r - q)\ y(s - q) - td = o - q$
- This can be written as a system of 3 equations (one for each dimension)
  - Solve by using linear algebra
- Rays that hit the triangle satisfy the first constraint on (b,a,y)

Disk

- Very Simple
  - Calculate the hitpoint on the plane
  - Measure the distance from the center of the disk to the hit point
  - Save time
    - Don’t calculate SQRT

Generic Objects

- What are Generic Objects?
  - Objects where
  - Example
    - Sphere
      - r=1
      - Center = (0,0)
- Why?
  - We can only ray trace objects we can
    1. Derive the Ray-Intersection Equation
    2. Solve the Ray-Intersection Equation
  - Generic Objects provide another technique for solving the ray-intersection equation
    - If we can solve the equation for a generic object, we can solve the equation for any linear transformation of that object

Higher Order Objects

- Cylinder
  - Euclidean
    - $x^2 + y^2 - r^2 = 0$
  - Leads to Quadratic Equation
- Torus
  - $(z - a)^2 + y^2 - b^2 = 0$
  - Leads to Quartic Equation
    - Solvable (code available)
- Any cylinder or torus can be solved by making it a generic object by transformations

Part Objects

- Cylinder
  - Limit the y values
  - Limit the angle
    - Must compute the hist point angle
- Sphere
  - Limit y values
  - Limit angle
- Tori
  - Limit Either of the angles

Compound Objects

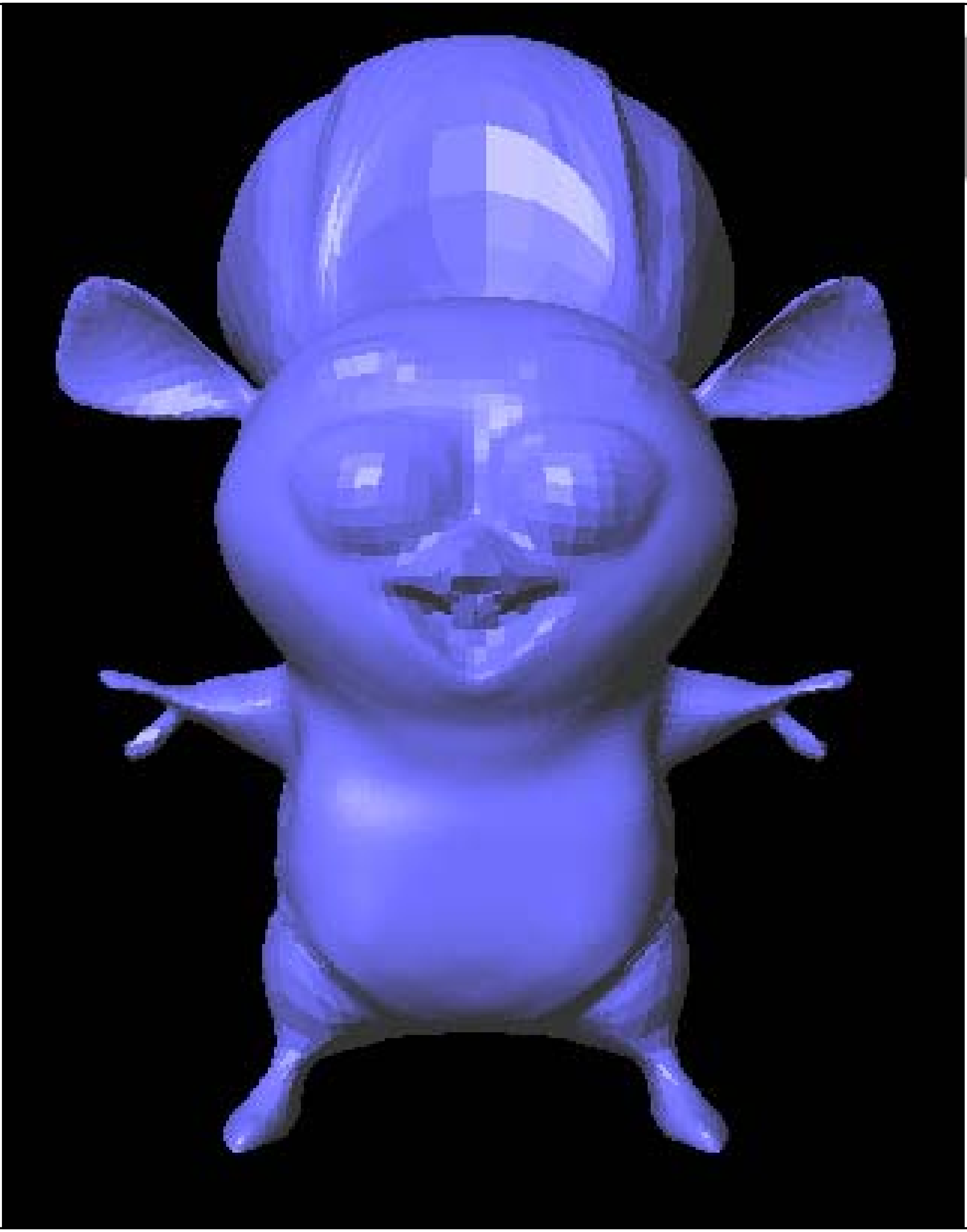
- Can create more interesting objects
  - Solid Cylinder
- Leads to Hierarchal Bounding Boxes

More…

- Transformations
  - Start with a generic object, transform it
  - Compute the hitpoint for the generic object
  - Transform it Back
- Regular Grids
  - Divide up the space in to a grid
  - Only computer hit function for objects the ray passes through
- Clever Modeling

Current Work

- How do we speed up Intersection Calculations?
  - Bounding Boxes
  - Regular Grids
    - Hierarchal Grids
  - Dividing objects into space boxes
  - Dividing rays into categories
    - See paper (Fast Ray Tracing by Ray Classification)



Meshes

- Low Rez
  - 10^3
- High Rez
  - 10^5
- Better Models
  - 10^7

Speeding Up Meshes

- Convert to a Function(???)
  - Stomach is mostly flat(ish)
    - Why use so many polygons?
  - How?
    - Subdivide
    - 3D Version Beizier Curve?
- Space Partition the Mesh
  - Head/Tail/Legs/Arms/Body
  - Further Subdivision leads to binary search
  - Can also be done with a heirarchal regular grid

Further Reading

- Calculus (3D)
- Linear Algebra

References

- Ray Classifications
  - http://portal.acm.org/citation.cfm?id=37401.37409&coll=Portal&dl=GUIDE&CFID=82389408&CFTOKEN=35547816
- Ray Tracing From The Ground Up