**Implementation**

9. Object Parser Modifications

Overview diagram

Modifications

Object parser (Andrew’s)

PARSED OBJECT

Face

Triangle

Raytracer

COMPLEX OBJECT

OCT TREE

OBJ FILE

Face

Triangle

**1) Conversion from object parser types to native types:**

Basic object parser used: Andrew Goodney

The basic object parser was called for each object, the object parser types

Facelist, Normal list, Texture list were converted to the type Triangle for compatibillty and direct use with the raytracer code.

Face (vertices ,normal, textures) converted to Triangle(vertices, normal, textures)

**2) Triangle Intersection method: LEE**

Algorithm:

d = - N . v0

t = - ( (N . Ray\_origin) + d) / N . Ray\_dir

P = Ray\_origin + (t . Ray\_dir)

N=> normal

P => intersction point,

t=> intersection distance

**Inside outside test LEE**

For each edge

test = (N . (E X (P – v0) )

E => Edge vector

If test negative point is outside edge E0

**3) Optimization BVH: Bounding volume hierarchy using OCT trees**

*a) Construction of the Oct tree*

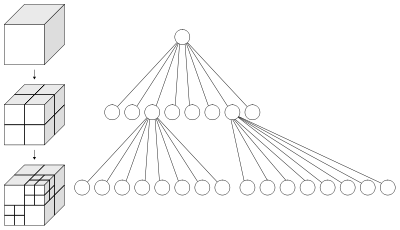
*i) Division of the Bounding box at each stage into sub oct*

*ii) Triangle inside oct test*

*b) Traversal of oct tree*

*i) Ray intersecting oct test*

**a) Construction of oct tree**



**i) Division of the Bounding box at each stage into sub octs**

A tree is constructed recursively each with 8 child nodes since each bounding box is divided into 8 smaller octs,

BoundingBox sub\_box[8];

sub\_box[0].SetBounds(x\_low, x\_mid, y\_mid, y\_high, z\_low, z\_mid); sub\_box[1].SetBounds(x\_mid, x\_high, y\_mid, y\_high, z\_low, z\_mid);

sub\_box[2].SetBounds(x\_low, x\_mid, y\_low, y\_mid, z\_low, z\_mid); sub\_box[3].SetBounds(x\_mid, x\_high, y\_low, y\_mid, z\_low, z\_mid);

sub\_box[4].SetBounds(x\_low, x\_mid, y\_mid, y\_high, z\_mid, z\_high); sub\_box[5].SetBounds(x\_mid, x\_high, y\_mid, y\_high, z\_mid, z\_high);

sub\_box[6].SetBounds(x\_low, x\_mid, y\_low, y\_mid, z\_mid, z\_high); sub\_box[7].SetBounds(x\_mid, x\_high, y\_low, y\_mid, z\_mid, z\_high);

**ii) Triangle inside oct test**

Bounding box of triangle is compared with bounding box of oct and if any of the part is inside then the triangle is added to that oct.

if( (SmallBox.left\_bound <= right\_bound) &&

(SmallBox.right\_bound >= left\_bound) &&

(SmallBox.bottom\_bound <= top\_bound) &&

(SmallBox.top\_bound >= bottom\_bound) &&

(SmallBox.front\_bound <= back\_bound) &&

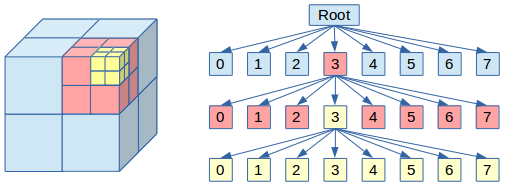
(SmallBox.back\_bound >= front\_bound) )

return true;

else

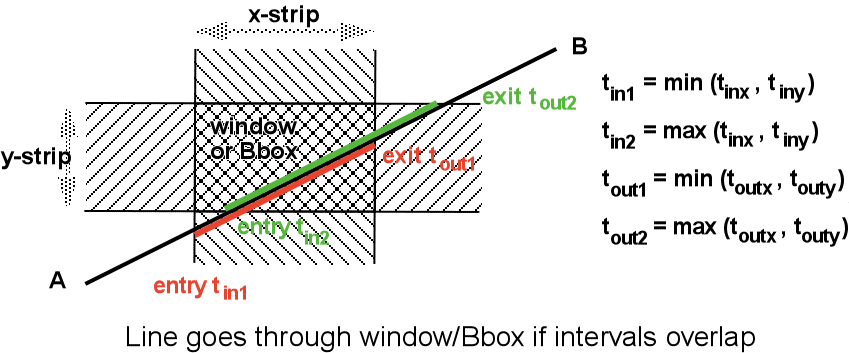
return false;

**b) Traversal of oct tree**



Only the bounding box that is intersected by the ray is further explored.

**i) Ray intersection test with bounding box**



**Code :**

For each pair of planes of X , Y and Z

t1 = (left\_bound - r.origin.x) / r.direction.x;

t2 = (right\_bound - r.origin.x) / r.direction.x;

if(t1 <= t2) {t\_near = t1; t\_far = t2;}

else {t\_near = t2; t\_far = t1;}

if(t\_near > t\_near\_furtherest) {t\_near\_furtherest = t\_near;}

if(t\_far < t\_far\_nearest) {t\_far\_nearest = t\_far;}

(above code shows for the x planes)

//------------------------------------------

if(t\_near\_furtherest > t\_far\_nearest)

Ray intersects

else

Ray does not intersect

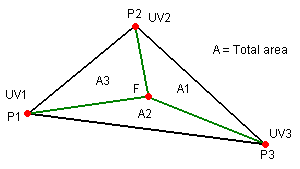
Reference:

<http://tavianator.com/2011/05/fast-branchless-raybounding-box-intersections/>

<http://www.siggraph.org/education/materials/HyperGraph/raytrace/rtinter3.htm>

**II > UV Interpolation and texture interpolation: using Barycentric coefficients**

Barycentric coefficients were found using the triangle are ratio, and use for UV and texture interpolation inside the triangle.



Code:

areaABC = dot( face\_normal, cross( sub(b,a), sub(c,a)) );

areaPBC = dot( face\_normal, cross( sub(b,P), sub(c,P)) ) ;

areaPCA = dot( face\_normal, cross( sub(c,P), sub(a,P)) ) ;

bary\_a = areaPBC / areaABC ; // alpha

bary\_b = areaPCA / areaABC ; // beta

bary\_c = 1.0f - bary\_a - bary\_b ; // gamma

Vector inter\_normal = add (normal\_a.scalarMult(bary\_a) ,normal\_b.scalarMult(bary\_b) , normal\_c.scalarMult(bary\_c));

All Reference:

http://www.scratchapixel.com/old/lessons/3d-basic-lessons/lesson-12-introduction-to-acceleration-structures/bounding-volume-hierarchy-bvh-part-2/

<http://tavianator.com/2011/05/fast-branchless-raybounding-box-intersections/>

<http://www.siggraph.org/education/materials/HyperGraph/raytrace/rtinter3.htm>