Collision Detection

Wayne Rippin (UoD)
Dr Panagiotis Perakis (MC)

Overview

- In the last session, you saw how we could detect collision with the terrain.
- This week, we will look at how we can detect collisions between objects

Bounding Volumes

- If we are testing for collisions between two objects, we would need to test to see if any triangle on one object intersects with any triangle on the other object.
- Given the number of triangles that make up many objects, this is clearly impractical due to the amount of time it would take.

Bounding Volumes

- If you need very fine collision detection, you might need to test every triangle, but a rough approximation can be made by associating a bounding volume with an object
- Collision detection can be done by testing to see if the bounding volumes intersect. Then, if you need to, you can do more fine-tuned testing on triangles.

Types of Bounding Volume

- Usually mathematical shapes
- The two most common are:
  - Sphere
  - Axis Aligned Bounding Box (AABB)
- Can also include:
  - Capsule
  - Lozenge
  - Cylinder
  - Simplified meshes

Bounding Volume Types

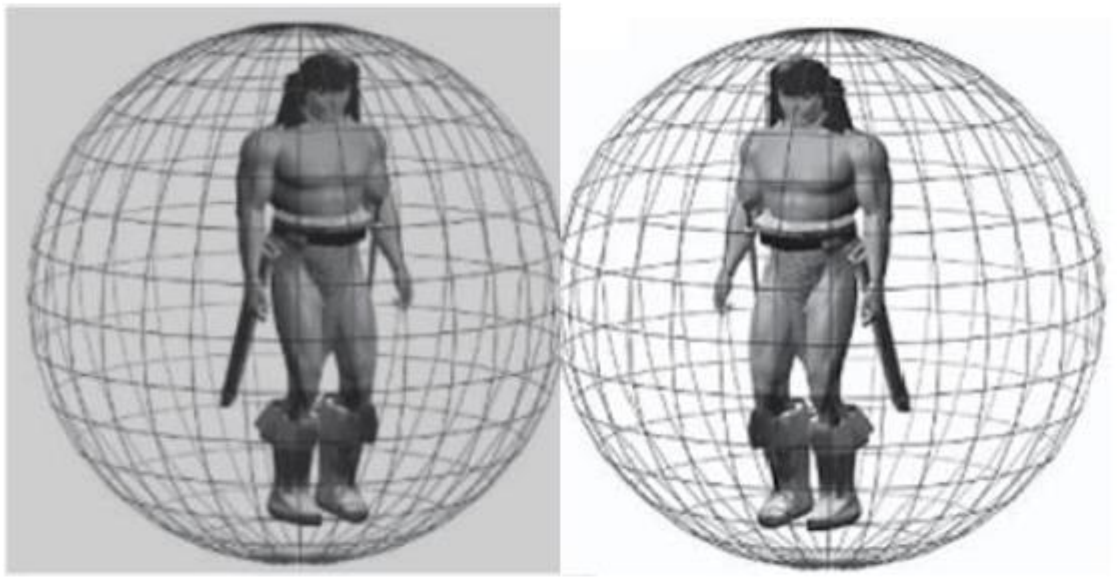
- The aim is to have an approximation to the geometry of the object that is being “bounded”
- Must completely enclose the object
- Not a visible object

Bounding Sphere

- Simple object
- All we need to keep track of is the centre (x, y and z) and the radius
- Does not change as the object inside is rotated
- Quick to create

Testing for Collisions

- We can simply look to see if two bounding spheres intersect (see later)



Creating a Bounding Sphere

- Many methods exist
- One algorithm is by Jack Ritter, “An Efficient Bounding Sphere” published in Graphics Gems.

Ritter Algorithm

1. Find min and max vertex along each axis
2. Find pair with largest distance between min and max
3. Use pair to form sphere with centre at mid-point and radius equal to distance between
4. Go through other vertices
   - Check distance, d, to sphere centre
   - If outside sphere’s radius, r
     - Move sphere centre toward vertex by (d-r)/2, set radius to (d+r)/2
     - Has effect of enclosing the vertex and existing sphere in new sphere

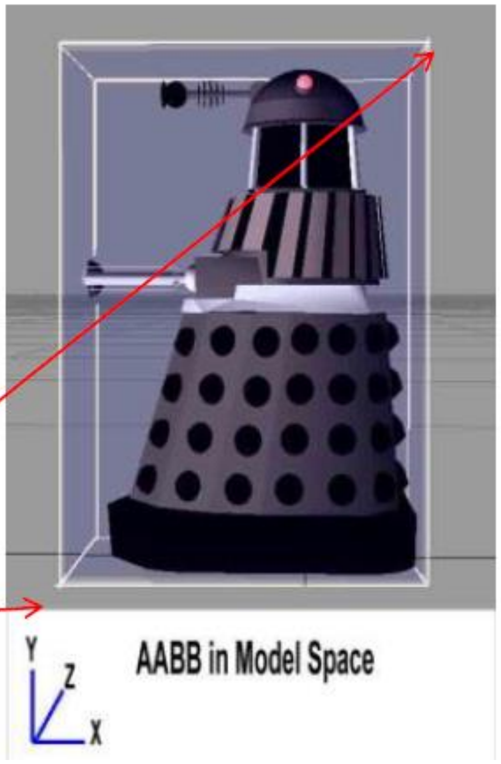
Computing Bounding Sphere

- However, we do not need to worry about this in this module
- In DirectXMath, a BoundingSphere class is provided. This includes a method (CreateFromPoints) that will calculate a bounding sphere for you from a series of points:
  - See https://msdn.microsoft.com/en-us/library/windows/desktop/microsoft.directx_sdk.directxcollision.boundingsphere.createfrompoints(v=vs.85).aspx
  - Note that these methods use the base DirectXMath types. Because we are using the SimpleMath library, instead of XMFLOAT3, use Vector3. Also, use Vector4 instead of XMVECTOR and Matrix instead of XMMATRIX.

Axis-Aligned Bounding Box

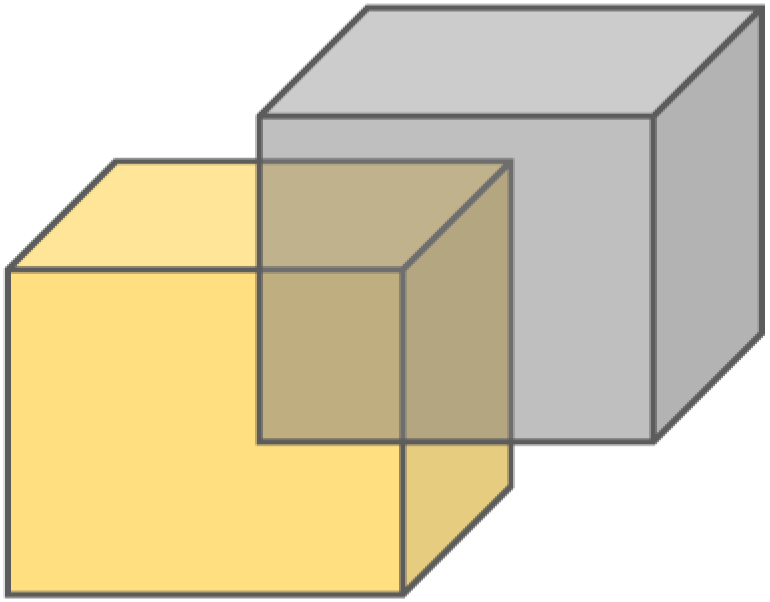
- A box that is always aligned with the standard x, y and z axes.

Need to calculate the position of the bottom left corner and top right corner



Testing for Intersections

- Since the AABBs are axis aligned, the test to see if two intersect is quite simple (see later)



Computing Bounding Sphere

- In DirectXMath, there is a BoundingBox class which contains a CreateFromPoints method.
  - See https://msdn.microsoft.com/en-us/library/windows/desktop/microsoft.directx_sdk.directxcollision.boundingbox.createfrompoints(v=vs.85).aspx

Transforming Bounding Volumes

- The previous slides show how an initial bounding sphere or AABB can be calculated
- However, as the object it surrounds is moved in world space, we need to also move the bounding volume to ensure it is still enclosing the object.

Transforming Bounding Spheres

- Transforming a bounding sphere is simple
- All we need to do is apply the same world transform to the centre position of the bounding sphere that we apply to the vertices of the object the sphere encloses.

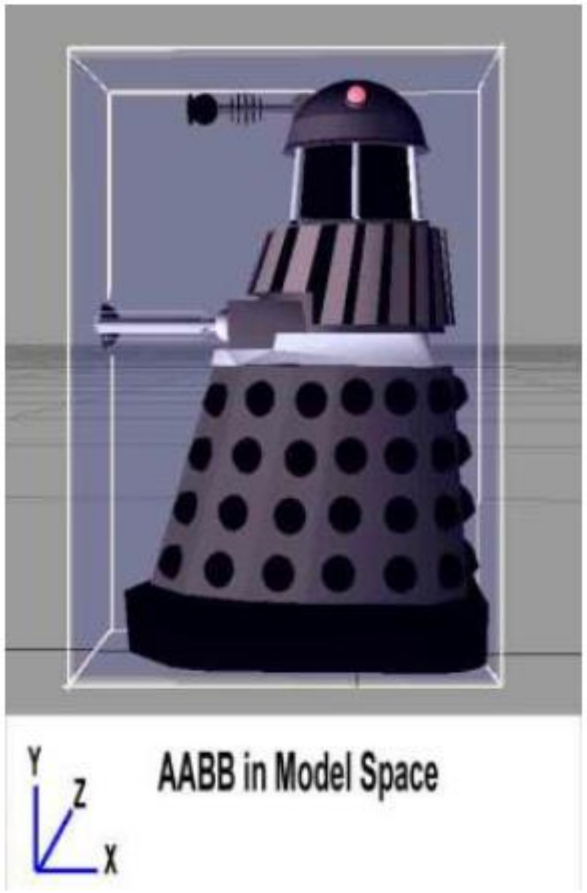
Apply world transform to object

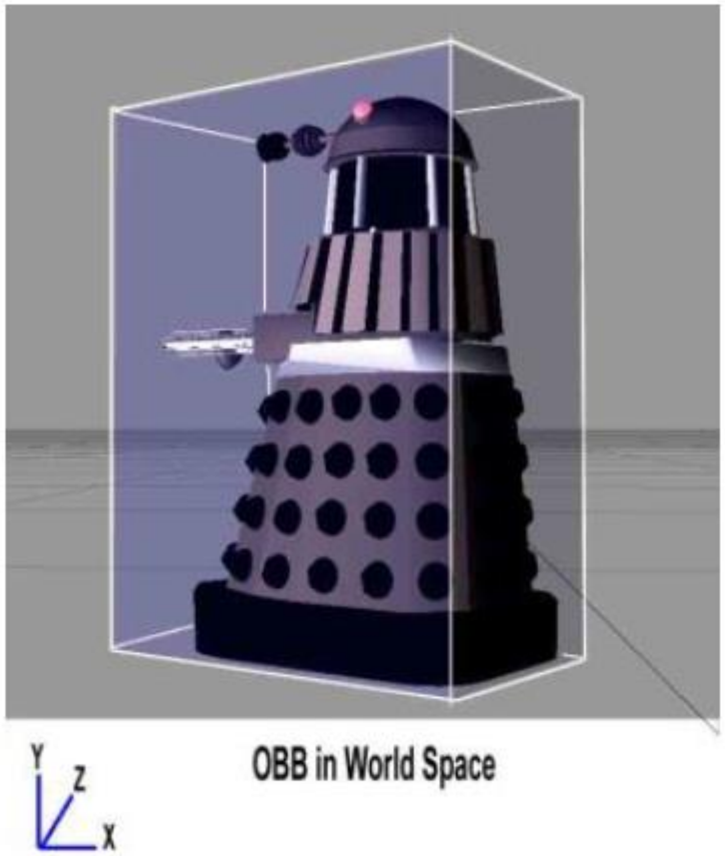
Apply world transform to centre of bounding sphere

Transforming a AABB

- Transforming an axis-aligned bounding box is more involved.
- If we apply any rotational transformations, the bounding box will no longer be axis aligned – it will become an oriented bounding box (OBB).

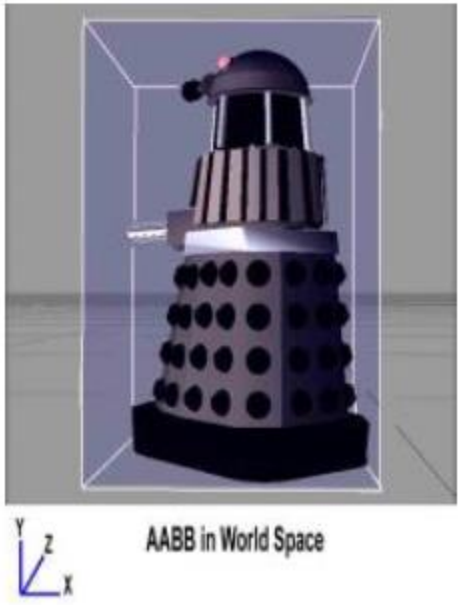
Transforming a AABB





Transforming a AABB

- Doing intersection testing on OBBs is considerably more complicated.
- What we want is a new AABB that surrounds the object now that it has been transformed



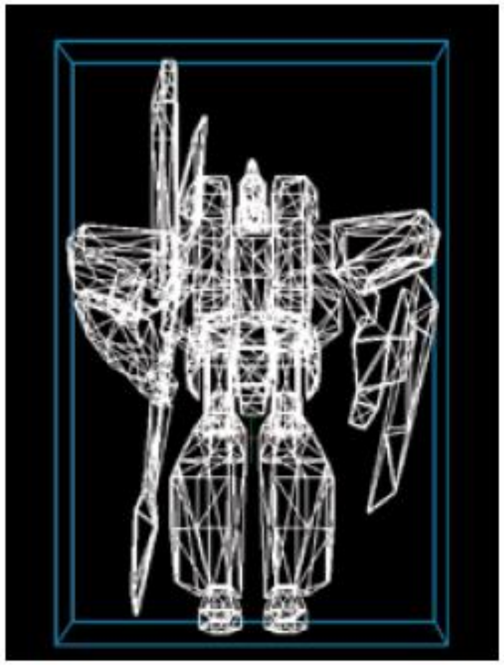
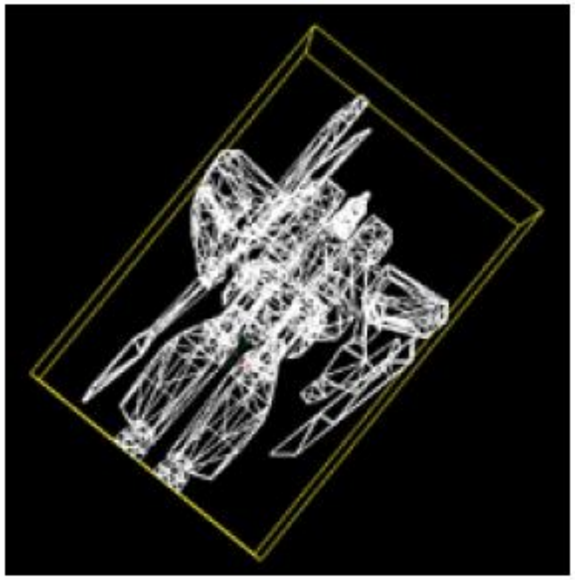
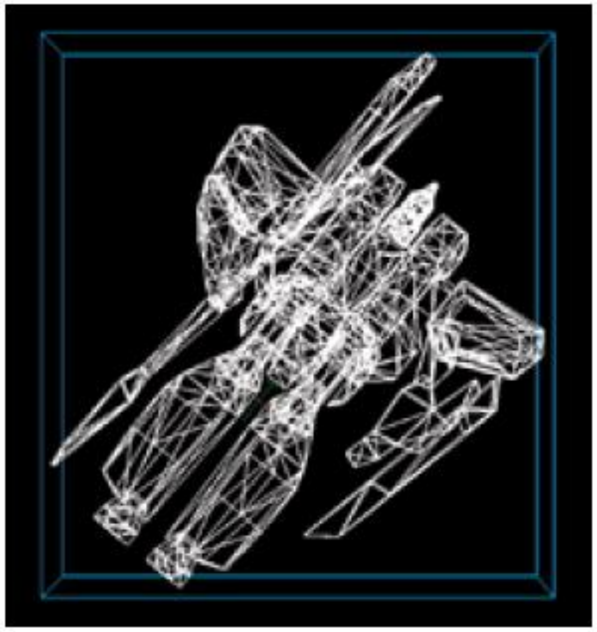
Transforming a AABB

- We don’t want to do a complete new calculation based on the vertices of the object since that will take too long.
- What we can do is change the information we stored about the original AABB so that we store the vertices of the eight corners.
- We can then transform these at the same time as we transform the object and use these to calculate the new AABB.

Transforming a AABB

- After the bounding box has been created:
  - Store the vertices of the box in an array
- After the object has been transformed:
  - Transform the vertices of the bounding box using the same world matrix as used to transform the object
  - Recreate the AABB using the transformed vertices.

Transforming a AABB

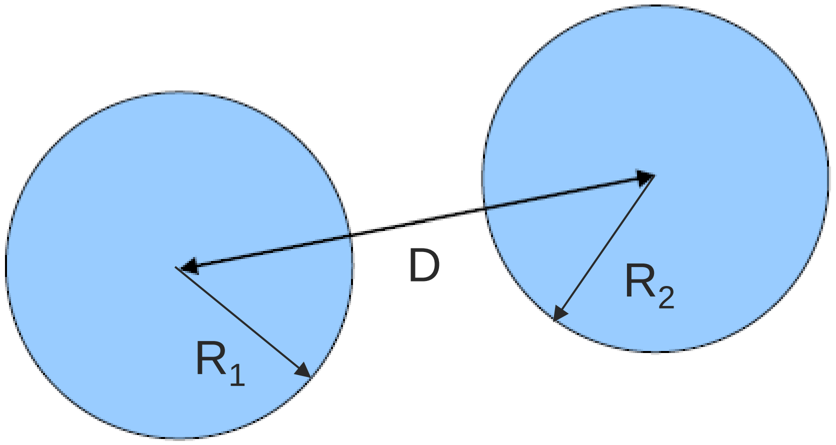
Gamasutra - When Two Hearts Collide: Axis-Aligned Bounding Boxes

Testing for Collisions

- Need to test if bounding volumes intersect
- Need to have routines to test different types of bounding volume
  - Test sphere against sphere
  - Test AABB against AABB
  - Test sphere against AABB
  - etc

Sphere-Sphere Intersection

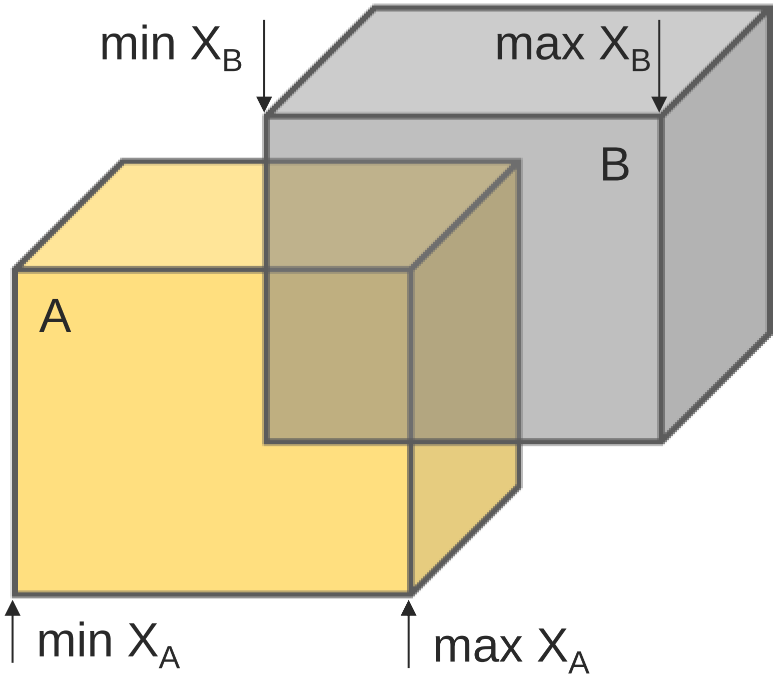
- Compute distance between centres of spheres
- Collision occurred if the distance between centres is less than the sum of radii



$$\boxed{D < R_1+R_2}$$

AABB-AABB Testing

- Compare box extents to determine boxes are NOT intersecting
  - If the max X position of A is less than the min X position of B they do not collide (A is at left of B)
  - If the min X position of A is greater than the max X position of B they do not collide (A is at right of B)
  - Repeat test for Y
  - Repeat test for Z
  - If none of these fail then the boxes MUST be intersecting



Other Testing

- Left to you to find...

DirectXMath Makes It Easier

- Note that DirectXMath makes this all a lot easier since the BoundingSphere and BoundingBox classes provide methods to test for intersection with other bounding volumes and to transform bounding shapes
- But it is worthwhile knowing how to do it manually anyway in case you ever have to write your own collision detection code.

Adding Bounding Volumes to Your Code

- Implement a base BoundingVolume class

```
class BoundingVolume
{
public:
    virtual bool      IsIntersecting(shared_ptr<BoundingVolume>
                      otherVolume)
      { return false; }
    virtual bool      IsIntersecting(Vector4& point)
      { return false; };
    virtual void      Update(Matrix& worldTransformation)
      {}
};
```

Adding Bounding Volumes to Your Code

- Add additional classes that inherit from BoundingVolume for each type of bounding volume type you want to implement (sphere, AABB, etc)
- Add a reference to a BoundingVolume to the base SceneNode.
  - Set the default value to be nullptr.
  - Edit the Update method to also update the BoundingVolume if it is not nullptr.

- Add the following methods to SceneNode:

```
shared_ptr<BoundingVolume> GetBoundingVolume()
      { return _boundingVolume; }
virtual bool IsColliding(shared_ptr<SceneNode> otherNode)
      { return false; }
```

Adding Bounding Volumes to Your Code

- Update the ResourceManager so that it adds the relevant bounding volumes to the mesh if requested.
- To do this:
  - Once a sub-mesh has been loaded (in LoadModelFromFile()), create a bounding volume for the sub-mesh
  - Once all of the sub-meshes have been created, create a bounding volume for the complete mesh by merging the sub-mesh bounding volumes together.

Intersection Testing

- Think about where in your framework you should add code to perform intersection testing
- How do you determine which objects in your scene graph to test against?

This Week

- Add support for bounding volumes to your code
- Start with one type of intersection testing (either sphere or AABB)

Next Week

- Working on your coursework assignment!!!

WITH YOUR ASSIGNMENT