FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING

Computeranimation

Lesson 3 – Collision

COMPUTER GRAPHICS • ERLANGEN

Introduction

Primitive Test

Bounding Volumes

Collision Response

Application

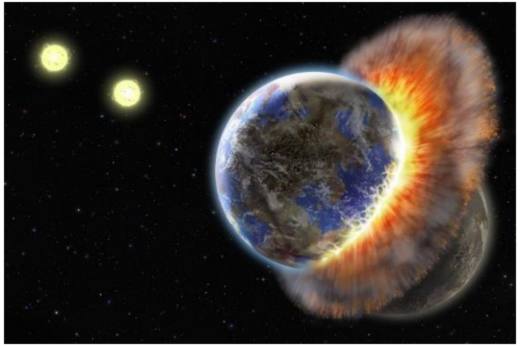
Motivation

Topics

- Rigid Transformation
- Animation
- **Collision**
- Dynamic
- Mass-Spring Simulation
- Rigging and Skeletal Animation

Introduction

Collision Basics



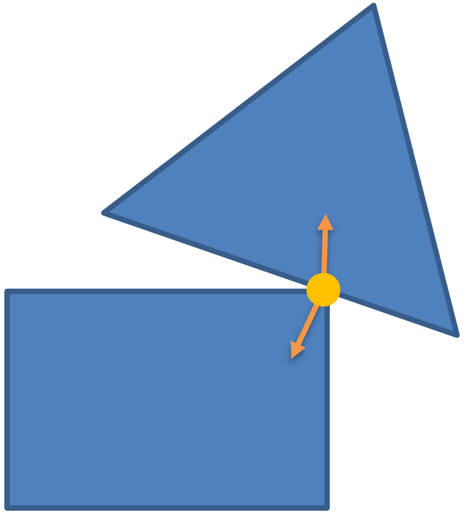
- Collision Handling in general is divided into two phases

1. **Collision-Detection**: Determine Locations where collisions occur
2. **Collision-Response**: Resolve colliding actors

Collision Detection

- In this phase all data for a proper collision handling is determined
- The collision state consists of:

- Contact Position(s)
- Contact Normal(s)



Introduction

Primitive Tests

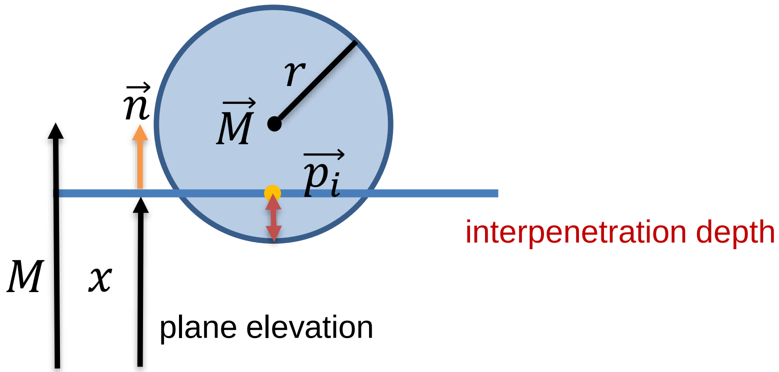
Bounding Volumes

Collision Response

Application

Collision Detection

Sphere-Plane Intersection



intersection if:

$$x > (M - r)$$

intersection point

$$\vec{p_i} = M - (r - d)\vec{n}$$

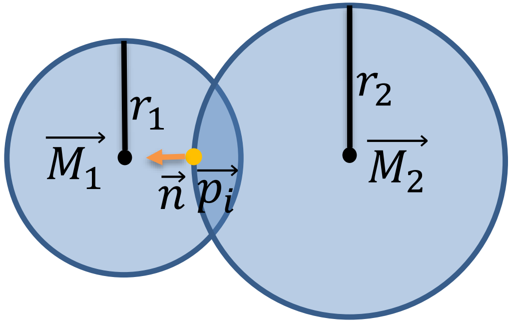
$$d = \vec{M} \cdot \vec{n} - x$$

intersection normal

$$\vec{n}$$

Collision Detection

Sphere-Sphere Intersection



intersection point

$$\overrightarrow{p_i} = M_2 + r_2 \cdot (\overrightarrow{M_1} - \overrightarrow{M_2})$$

intersection normal

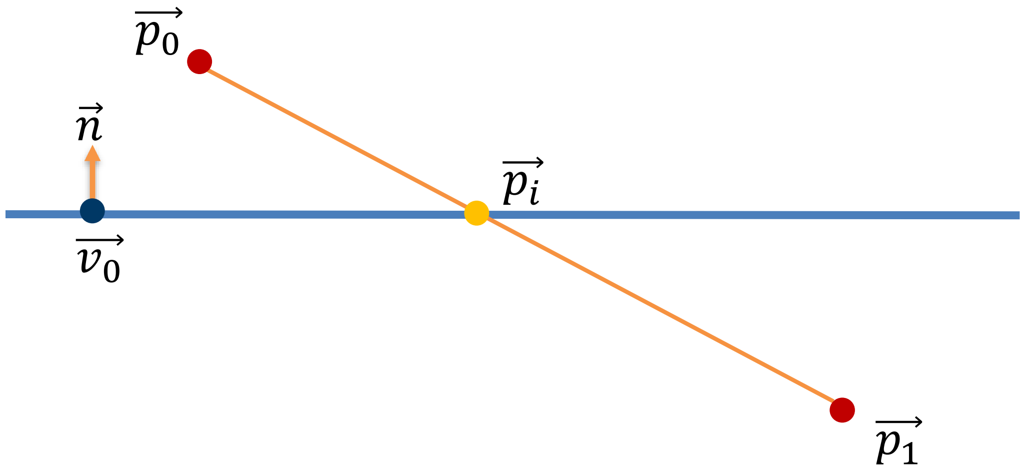
$$\vec{n} = \frac{\overrightarrow{M_1} - \overrightarrow{M_2}}{\left\|\overrightarrow{M_1} - \overrightarrow{M_2}\right\|}$$

LGDV
COMPUTER GRAPHICS • ERLANGEN

Collision Detection

Segment-Plane intersection

- important for many intersection tests including triangle meshes
- also important for raytracing



intersection point

$$\overrightarrow{p_i} = \overrightarrow{p_o} + \boldsymbol{r} \cdot (\overrightarrow{p_1} - \overrightarrow{p_0})$$

$$\boldsymbol{r} = \frac{\vec{\boldsymbol{n}} \cdot (\overrightarrow{v_0} - \overrightarrow{p_0})}{\vec{\boldsymbol{n}} \cdot (\overrightarrow{p_1} - \overrightarrow{p_0})}$$

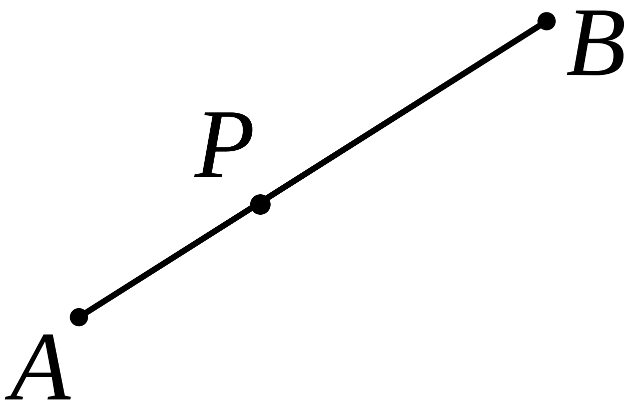
If denominator equals zero, segment is parallel to plane

Collision Detection

Barycentric Coordinates

- Used to define a point within a segment/triangle

1D Segment:



$$P = (1-t)A + tB \quad t \in [0,1]$$

$$= \alpha A + \beta B$$

condition: $\alpha + \beta = 1$

Collision Detection

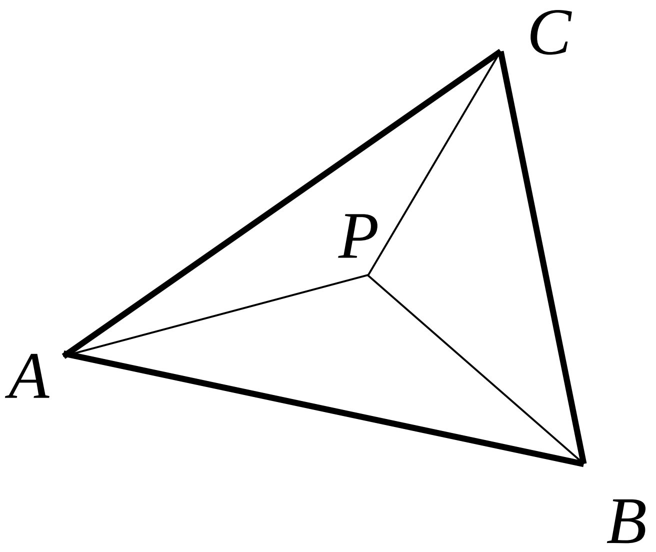
Barycentric Coordinates

- Used to define a point within a segment/triangle

2D Segment:

$$P = \alpha A + \beta B + \gamma C$$

$$\text{condition: } \alpha + \beta + \gamma = 1$$



Collision Detection

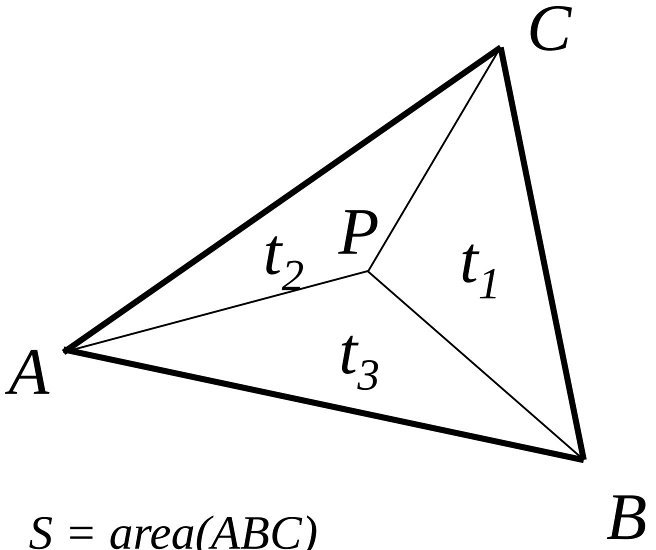
Barycentric Coordinates

- Used to define a point within a segment/triangle

2D Segment:

$$P = \alpha A + \beta B + \gamma C$$

$$\alpha = \frac{t_1}{S};\ \beta = \frac{t_2}{S};\ \gamma = \frac{t_3}{S}$$

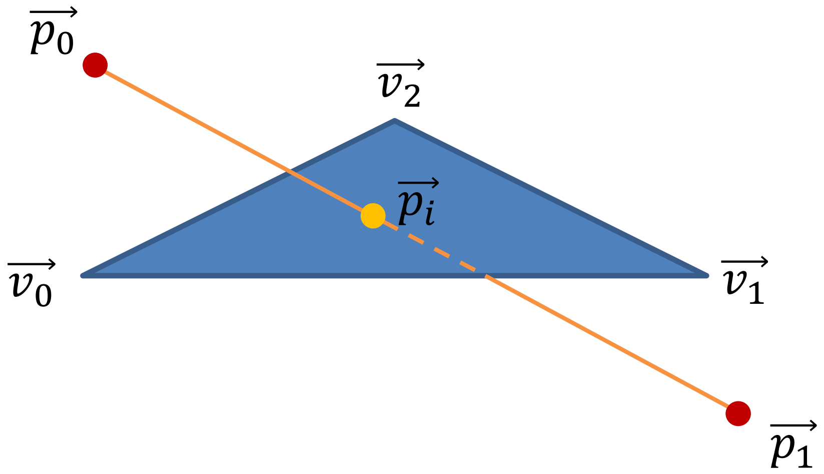


$$S = area(ABC)$$
$$t_1 = area(PBC)$$
$$t_2 = area(PCA)$$
$$t_3 = area(PAB)$$

Collision Detection

segment-triangle intersection

- important for many intersection tests including triangle meshes
- also important for raytracing



intersection point

$$\vec{u} = (\vec{v_1} - \vec{v_0})$$

$$\vec{v} = (\vec{v_2} - \vec{v_0})$$

$$\vec{p_i} = \vec{v_o} + \boldsymbol{s} \cdot \vec{u} + \boldsymbol{t} \cdot \vec{v}$$

$$\vec{w} = (\vec{p_i} - \vec{v_0})$$

$$\boldsymbol{s} = \frac{(\vec{u} \cdot \vec{v})(\vec{w} \cdot \vec{v}) - (\vec{v} \cdot \vec{v})(\vec{w} \cdot \vec{u})}{(\vec{u} \cdot \vec{v})^2 - (\vec{u} \cdot \vec{u})(\vec{v} \cdot \vec{v})}$$

$$\boldsymbol{t} = \frac{(\vec{u} \cdot \vec{v})(\vec{w} \cdot \vec{u}) - (u \cdot \vec{u})(\vec{w} \cdot \vec{v})}{(\vec{u} \cdot \vec{v})^2 - (\vec{u} \cdot \vec{u})(\vec{v} \cdot \vec{v})}$$

Introduction

Primitive Tests

Bounding Volumes

Collision Response

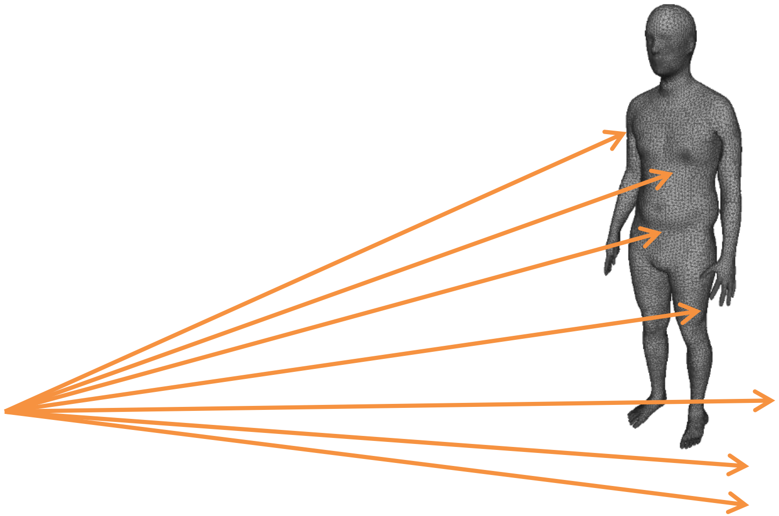
Application

COMPUTER GRAPHICS • ERLANGEN

Bounding Volumes

Motivation

- Intersection tests are expensive computations
- e.g. a ray has to be tested against **each** triangle
- even rays that miss the object !!!



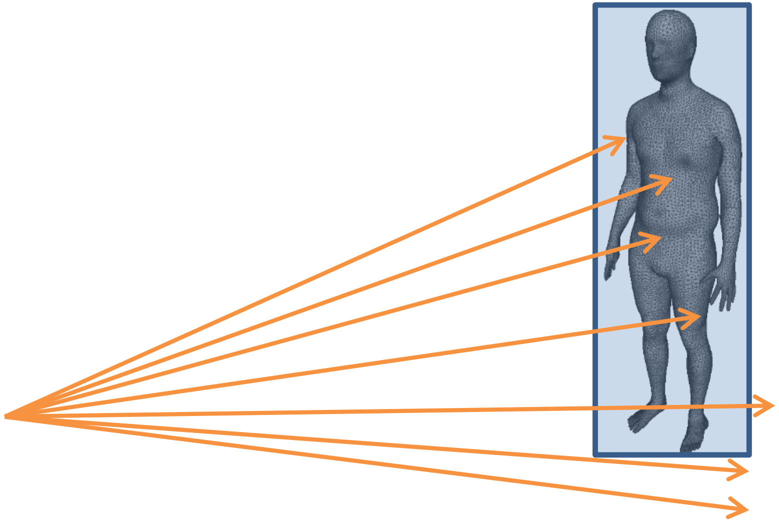
Bounding Volumes

Motivation

- Intersection tests are expensive computations
- e.g. a ray has to be tested against **each** triangle
- even rays that miss the object !!!

Idea

- Embed object in a host geometry (simple box)
- Test each ray with the box first



Bounding Volumes

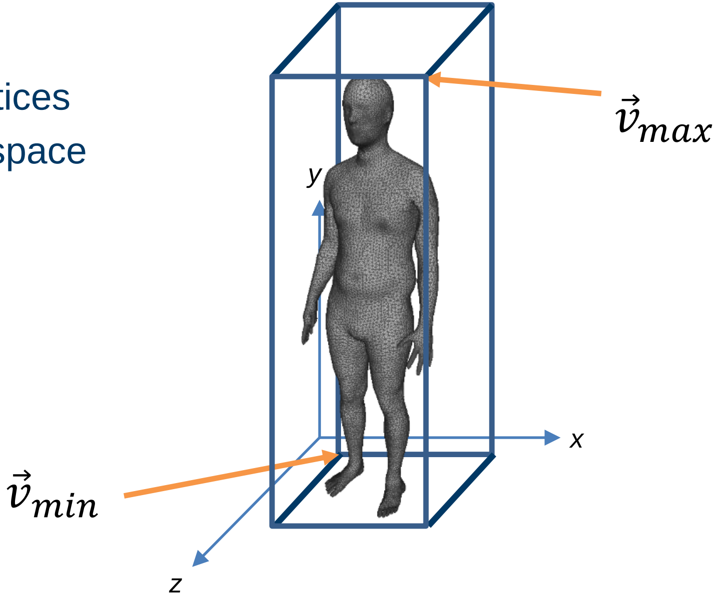
Axis-Aligned-Bounding-Box (AABB)

- A host geometry that contains all object‘s vertices
- and is aligned to the axis of the surrounding space
- AABB is uniquely defined by $\vec{v}_{min}$ and $\vec{v}_{max}$

AABB by expansion:

```
v_min = vertex[0];
v_max = vertex[0];

for i = 1 to numVertices-1
  if (vertex[i].x < v_min.x) then v_min.x =  vertex[i].x end
  if (vertex[i].y < v_min.y) then v_min.y =  vertex[i].y end
  if (vertex[i].z < v_min.z) then v_min.z =  vertex[i].z end
  if (vertex[i].x > v_max.x) then v_max.x =  vertex[i].x end
  if (vertex[i].y > v_max.y) then v_max.y =  vertex[i].y end
  if (vertex[i].z > v_max.z) then v_max.z =  vertex[i].z end
end
```

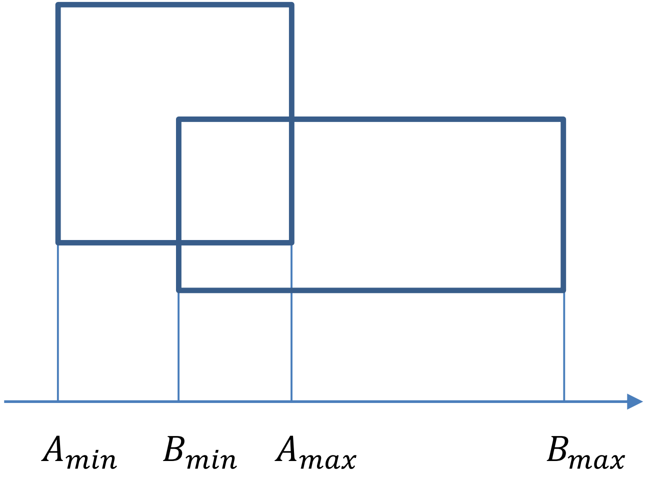


Bounding Volumes

Intersecting two AABBs

- There is **no** intersection **if** for all three coordinate axis holds:

$$A_{min} > \mathrm{B}_{\mathrm{max}} \,||\, B_{min} > \mathrm{A}_{\mathrm{max}}$$



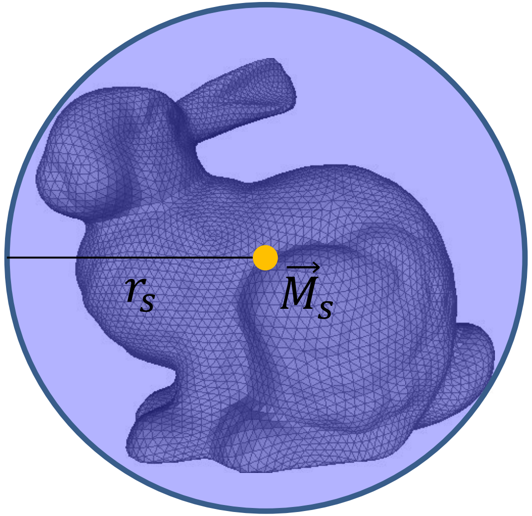
Discussion

- Advantage: Fast computation, easy to implement
- Disadvantage: Update for moving/rotating objects

Bounding Volumes

Bounding-Spheres

- Similar Idea as AABB, but with Spheres as enclosing Objects
- Advantage: Intersection tests are simple (simple distance test)
- Advantage: Rotation invariant
- Disadvantage: works well only for *compact* objects



$\vec{\boldsymbol{M}}_s = \frac{1}{N}\sum_{i=0}^{numVertices-1} \vec{v}_i$; // center of mass
r_s = 0.0; // radius

for i = 1 **to** numVertices-1
  if ($\|\vec{\boldsymbol{M}}_s - \vec{\boldsymbol{v}}_i\| > r_s$) **then** $r_s = \|\vec{\boldsymbol{M}}_s - \vec{\boldsymbol{v}}_i\|$ **end**
end

Introduction

Primitive Tests

Bounding Volumes

Collision Response

Application

Collision Response

Introduction

- Once a collision is detected, a proper reaction has to be performed
- We only consider ***rigid object collision***

Collision Response

Kinematic Reaction

Position Manipulation

Movement Reflection

Dynamic Reaction

Reflect Forces

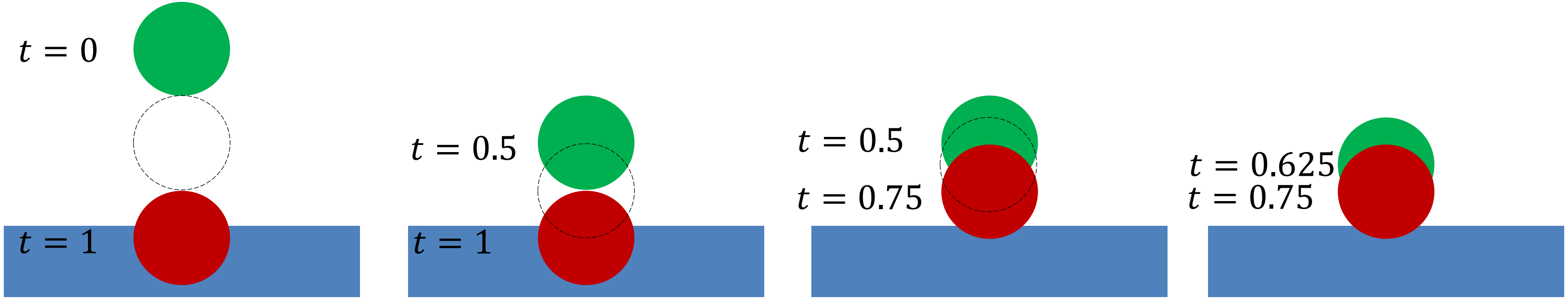
Apply Spring Forces

LGDV
COMPUTER GRAPHICS • ERLANGEN

Collision Response

Interval Halving

- Find collision in time and space accurately
- Halving the interval. If Sphere at midpoint does not intersect → new lower bound
- If sphere at midpoint intersects → new upper bound



Introduction

Primitive Tests

Bounding Volumes

Collision Response

Application