PSE Game Physics

Session (4)
Coarse/Fine Collision tests,
Collision: Sphere-Box, Box-Plane

Oliver Meister, Roland Wittmann

15.05.2015

Outline

Outline

Springs extended

Collision detection

- Overview
- Bounding Volume Hierarchy (BVH)
- Coarse collision detection
- Fine collision detection

Collision: Sphere-Box

Collision: Box-Plane

Undamped springs

We used the formula

$$F(t) = -k\, d(t)$$

to compute the force acting on a spring, where
$d(t)$ = length of spring $l(t)$ - equilibrium length l_0.
Applying this to two objects with mass centers $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$, distance $l = |\mathbf{x}_2 - \mathbf{x}_1|$ and masses $m_1, m_2 \in \mathbb{R}^+$ gives:

$$\mathbf{F}_1 = -\mathbf{F}_2 = k\, \frac{l - l_0}{l}(\mathbf{x}_2 - \mathbf{x}_1)$$

This means

$$\mathbf{a}_1 = \frac{\mathbf{F}_1}{m_1} = k\, \frac{l - l_0}{m_1\, l}(\mathbf{x}_2 - \mathbf{x}_1)$$

$$\mathbf{a}_2 = \frac{\mathbf{F}_2}{m_2} = -k\, \frac{l - l_0}{m_2\, l}(\mathbf{x}_2 - \mathbf{x}_1)$$

Damped springs

For damping, we could modify the formula by

$$F(t) = -k\, d(t) - c\, \dot{d}(t)$$

using a *damping coefficient* $c \in \mathbb{R}^+$.
Applying this to two objects with mass centers $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$, distance $l = |\mathbf{x}_2 - \mathbf{x}_1|$, velocities $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^3$ and masses $m_1, m_2 \in \mathbb{R}^+$ gives:

$$\mathbf{F}_1 = -\mathbf{F}_2 = \left(\frac{k\,(l - l_0)}{l} + \frac{c(\mathbf{v}_2 - \mathbf{v}_1) \cdot (\mathbf{x}_2 - \mathbf{x}_1)}{l^2}\right)(\mathbf{x}_2 - \mathbf{x}_1)$$

Question: Does this method suppress all motion eventually?

Outline

Springs extended

Collision detection

- Overview
- Bounding Volume Hierarchy (BVH)
- Coarse collision detection
- Fine collision detection

Collision: Sphere-Box

Collision: Box-Plane

Overview

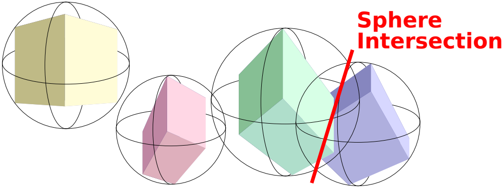
Collision detection is separated into several hierarchical passes:

- Bounding Volume Hierarchy (BVH) trees or similar structures
- Coarse collision detection
- Fine collision detection

Bounding Volume Hierarchy (BVH)

- When testing n objects pairwise for collisions, $\mathcal{O}(n^2)$ comparisons are required $\rightarrow$ Problem?
- BVHs reduce the complexity to at most $\mathcal{O}(n \log n)$:
  - Set up a spatial hierarchy of objects ($\mathcal{O}(n \log n)$).
  - Most objects are static $\Rightarrow$ **Reuse** the hierarchy for successive time steps.
  - During movement of objects and thus **deletion/insertion of nodes**, try to keep the tree **balanced** ($\mathcal{O}(n \log n)$).
  - Testing for intersections is **reduced to objects in direct neighborhood** ($\mathcal{O}(n \log n)$).
  - Further optimizations: Skip static subtrees, subtree-subtree tests instead of object-subtree tests, ...
- Not implemented so far in engine $\rightarrow$ Final project?

Coarse collision detection



- Idea: Use **simple & fast** collision tests to avoid running time-consuming fine collision tests.
- Simple collision tests: Spheres, Axis-aligned bounding boxes (AABB), Oriented bounding boxes (OBB)...
- When creating a new factory, its **bounding sphere radius** is computed.
- The object's radius and position is used for such an **early-reject** to avoid complex collision tests.
- Already implemented in the skeleton of our physics engine.

Fine collision detection

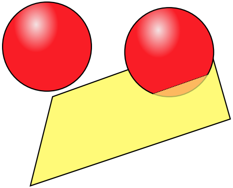


Figure: Example: Sphere-Plane collision

- **Accurate and time consuming** collision tests.
- In case of a detected collision, generates the collision data.
- Sphere-Sphere, Sphere-Box, Box-Box, ...
- ... the tests are your task to implement :-)
- Further tests - e.g. for generic polyeders - could be implemented in the final project.

Outline

Sphere-Box

- Can be handled similar to the sphere-plane collision test
- We **project** the sphere to the model space of the box
- Remember that boxes and spheres are symmetric!
  $\Rightarrow$ You can avoid redundant checks by using the absolute values of the sphere's center coordinates.
- After testing sphere-plane intersections, also check for sphere-edge and sphere-vertex intersections.

Outline

Box-Plane

- Our **finite plane** is still an **artificial construct** used for limiting box-planes. Therefore we assume that the size of a plane is larger than other physics primitives.
- Projecting the box to the model space of the plane simplifies many assumptions.
- Then **vertex-plane** testing can be done as follows:
  - Test whether all box **vertices** lie on one side, then no collision can be possible.
  - Otherwise, take the vertex with the **largest interpenetration depth** to get the collision data.
- Take also **edge-edge** contacts into account! All other cases can be excluded for now.
- Advanced collision tests in the next session (*separating axes*)