Collision Detection and Resolution

Tom Kelliher, CS 320

Mar. 25, 2009

1 Administrivia

Announcements

Assignment

Read Section 4.1, Appendices B and C.

From Last Time

Vectors.

Outline

1. Continued discussion of `collision.c`: ball placement, collision detection, collision resolution.

Coming Up

Animation.

2 `collision.c`

1. `placeBalls()`:

    (a) Placement of the first ball along unit circle.

    Velocity computation. Target: origin. Scaling velocity.

    Translating position to circle of radius 40.

    (b) Avoiding initial collision: Constrained placement of second ball; $\pi/4$ or more away.

    (c) Options: Aiming second ball at a point other than the origin. Computing and normalizing the velocity vector.

    Making the second ball stationary, at an arbitrary position.

2. `idle()`:

    (a) Updating ball position — this is animation.

    (b) Collision detection and response, $O(n^2)$ checks.

    (c) Beginning the next simulation when either ball leaves the "arena." No square roots.

    (d) Post a re-display event to render the new scene.

3. `collision()`

    (a) Simply, if the distance between the centers points is less than or equal to the sum of the radii, we've had a collision.

    (b) No square roots.

    (c) Discrete time step simulation: penetration problems.

4. `collisionResponse()`

(a) Dealing with penetration:

   i. Binary search over time step interval to find exact point of impact and take it from there. Computationally expensive.

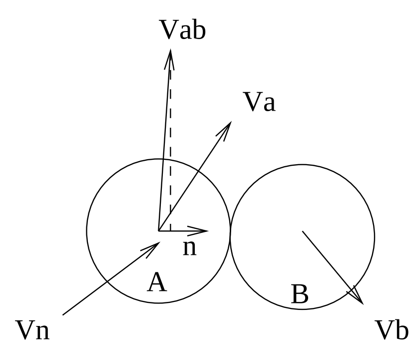
   ii. Ignore. Approximate collision point and normal. Allow collision response to push objects apart. May not look realistic, if collision response doesn't separate objects quickly enough.

   iii. Approximate collision normal and move each object 1/2 of penetration distance apart along normal. This may look abrupt. Could cause a cascade of penetrations. Assumes equal forces involved.

   This is what we use.

(b) We apply equal and opposite impulses along the collision normal to the two objects to bring them apart.

(c) Collision normal $(B - A)$, relative velocity vector $(V_a - V_b)$, and the projection back onto the normal (V_n):



$V_n = (V_{ab} \cdot \hat{n})\hat{n}$

(d) Coefficient of restitution: $v_n' = -\epsilon v_n$, or

$$(v_a' - v_b') = -\epsilon(v_a - v_b)$$

(e) Conservation of momentum: $m_a v_a + j\hat{n} = m_a v_a'$ or:

$$v_a' = v_a + \frac{j}{m_a}\hat{n}$$

Similarly for v_b'.

(f) Substituting and solving for j:

$$j = \frac{-(1+\epsilon)v_{ab} \cdot \hat{n}}{\frac{1}{m_a} + \frac{1}{m_b}}$$