Quadtree Collision Detection

Alden Page and Willem Yarbrough

Department of Computer Science
Allegheny College

April 28, 2015

What?

- Given 2D particles p_0 to p_n, which are colliding?

What?

- Given 2D particles p_0 to p_n, which are colliding?

Why?

- Simulations
- Games

What?

- Given 2D particles p_0 to p_n, which are colliding?

Why?

- Simulations
- Games

How?

- `colliding(p1, p2)` primitive, $O(1)$
- Brute-force
- Quadtree

Brute-force

- Call `colliding(`p_i`,` p_j`)` for each p_i, p_j.
- Complexity??

Brute-force

- Call `colliding(`p_i`,` p_j`)` for each p_i, p_j.
- Complexity??

$$O(n^2)$$

Quadtree

- Call `colliding(`p_i`,` p_j`)` for *nearby* p_i, p_j.
- Subdivide space into quadrants!!
- Complexity??

Quadtree

- Call `colliding`(p_i, p_j) for *nearby* p_i, p_j.
- Subdivide space into quadrants!!
- Complexity??

$$O(n \log n)$$

Python Implementation

- World of particles (location, velocity)
- Brute-force and quadtree “detectors” for comparison
- Drawing with Pygame

Other Stuff

- What about 3D? **Octree**
- What about any D? **K-D Tree**
- What about polygons? **BSP Trees**

Demo!