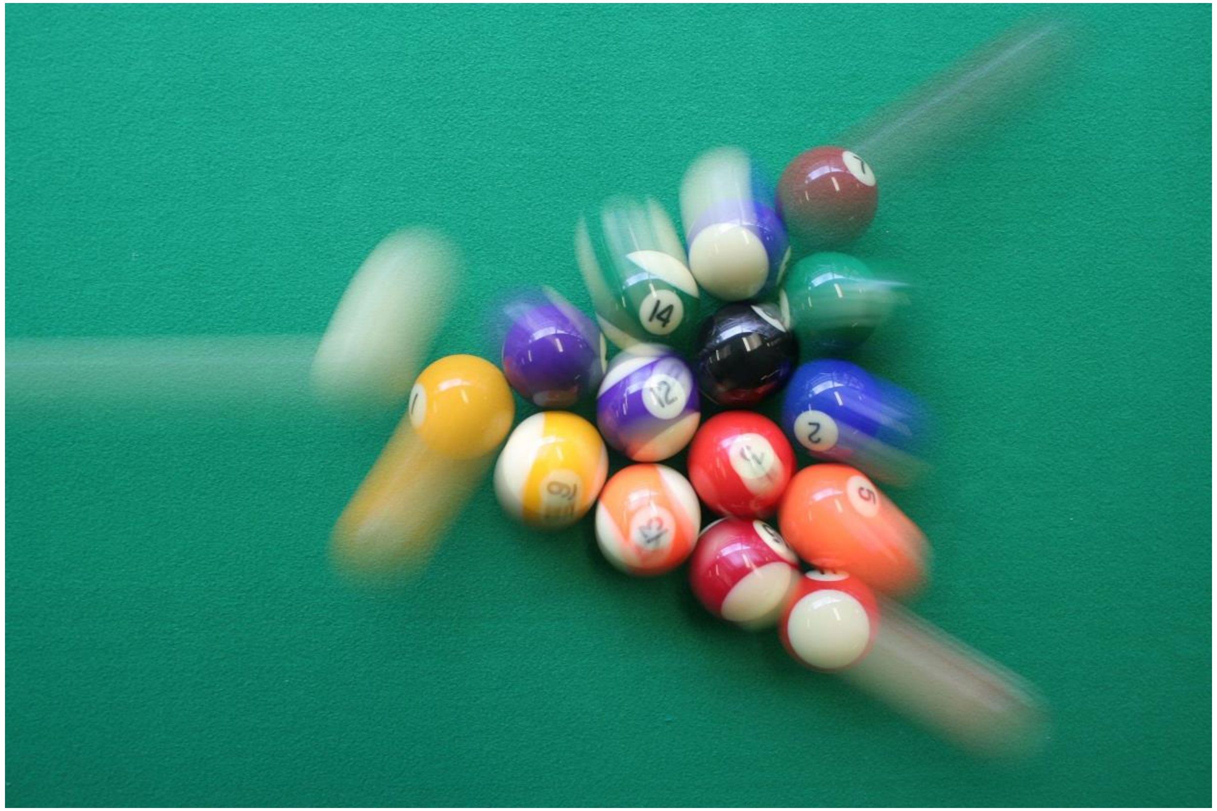
CPSC 427
Video Game Programming

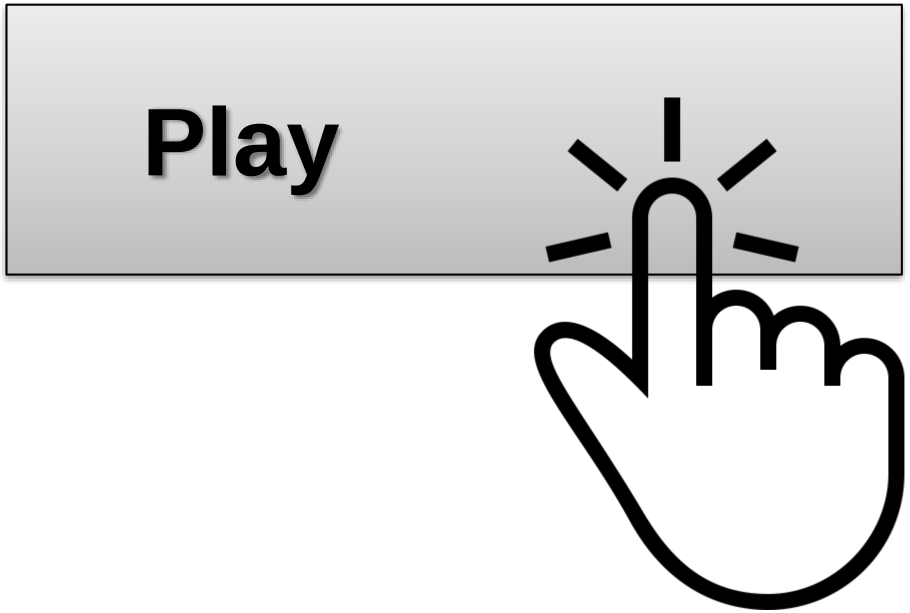
Collisions (and a bit of physics)



Helge Rhodin

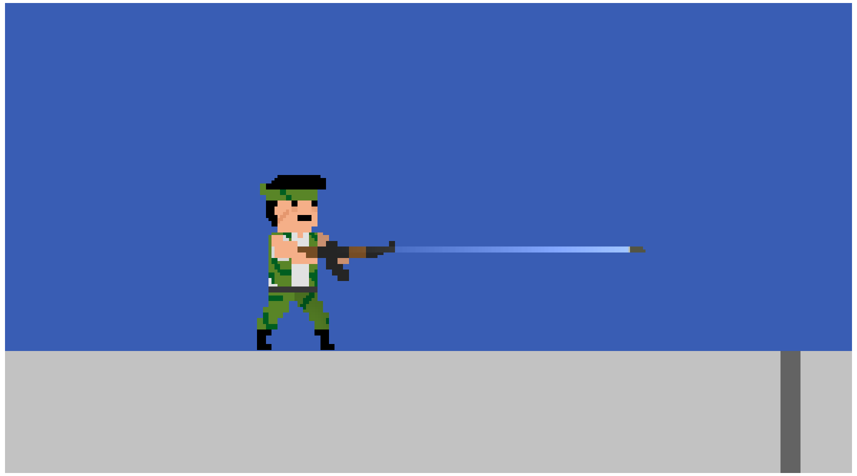
Motivation: Object selection

- ***Point inside object boundary?***



Motivation: Bullet trajectories

- ***Line-object or point-object intersection?***

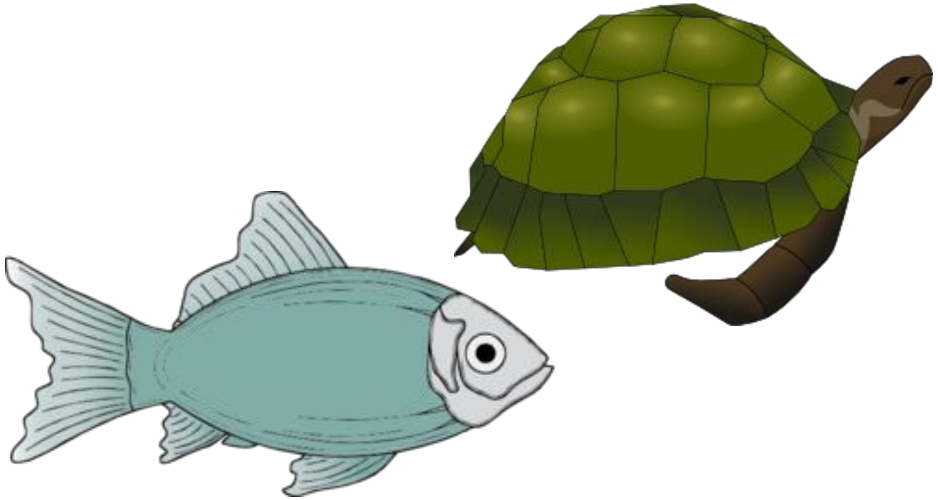


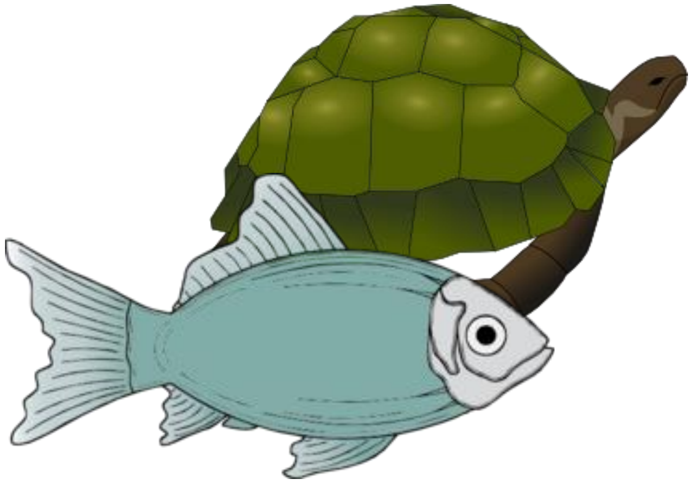
https://forum.unity.com/threads/2d-platformer-shooting.365971/

Motivation: Collision

- ***Prevent object penetration***

- ***How?***





Collision Configurations?

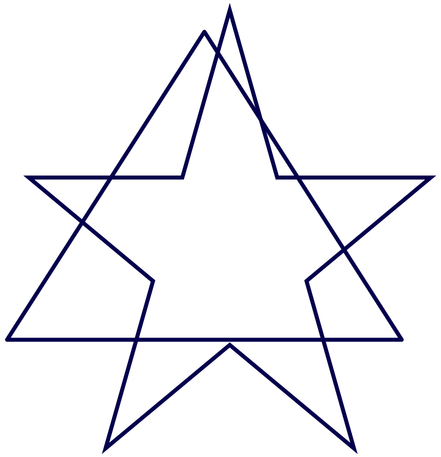
To detect collisions between polygons it is enough to test if their edges intersect

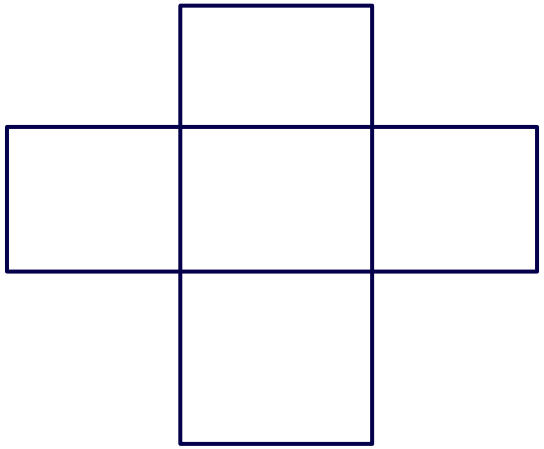
A. True
B. False

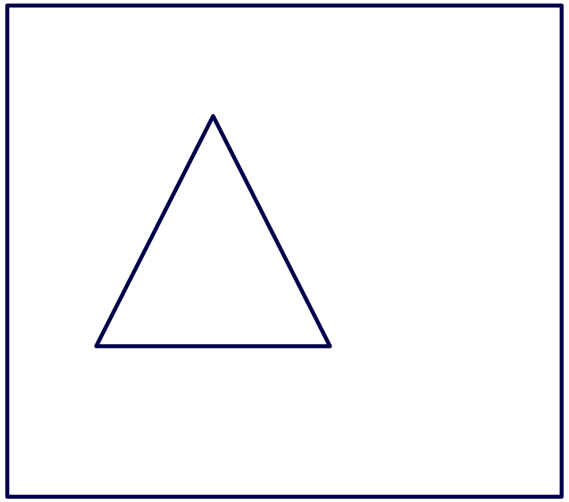
Collision Configurations?

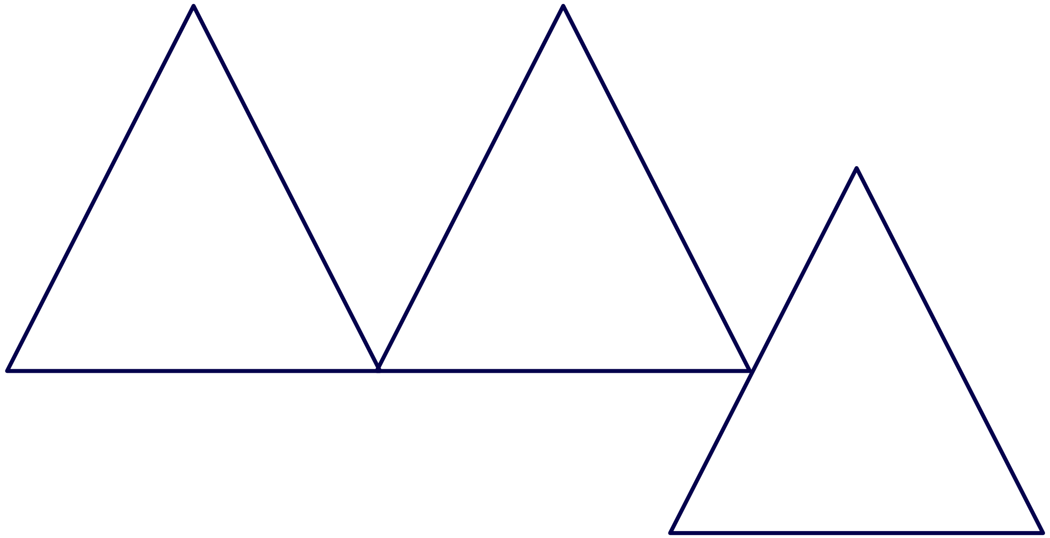
- Segment/Segment Intersection
  - *Point **on** Segment*

- Polygon inside polygon









Inside Test?

- How to test if one poly is inside another?

- Use inside test for point(s)

- How?
  - *Convex Polygon*
    - Same side WRT to line (all sides)
  - *Non-Convex*
    - Subdivide= triangulate
    - How?
    - Shoot rays (beware of corners and special cases)

Resources

http://www.realtimerendering.com/intersections.html

Curves

Mathematical representations:

- Explicit functions:
- Parametric functions
- Implicit functions

Explicit functions

- $y = f(x)$
- E.g. $y = ax + b$
- Single y value for each x
- Useful for?
  - *Terrain*
  - *“height field” geometry*

Parametric Functions

- 2D: x and y are functions of a parameter value t
- 3D: x, y, and z are functions of a parameter value t

$$C(t) \coloneqq \begin{pmatrix} p_y \\ p_x \end{pmatrix} t + \begin{pmatrix} q_x \\ q_y \end{pmatrix} (1 - t)$$

Line (segment)

$$C(t) \coloneqq \begin{pmatrix} cos(t) \\ sin(t) \end{pmatrix}$$

Circle (arc)

- Depends on parameter range $t_1 < t < t_2$

Implicit Function

- Curve (2D) or Surface (3D) defined by zero set (roots) of function

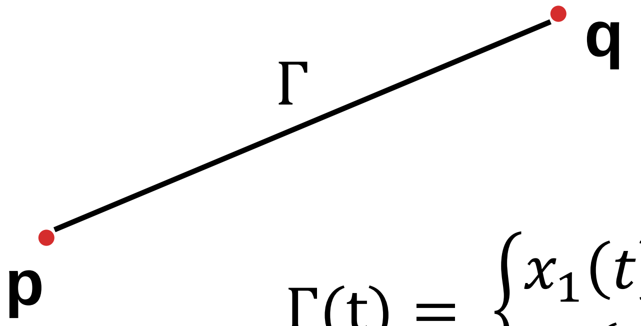
E.g:

$$S(x,y): x^2 + y^2 - 1 = 0$$

$$S(x,y,z): x^2 + y^2 + z^2 - 1 = 0$$

Lines & Segments

Segment Γ from $\mathbf{p} = (x_0, y_0)$ ***to*** $\mathbf{q} = (x_1, y_1)$



$$\Gamma(t) = \begin{cases} x_1(t) = x_0 + (x_1 - x_0)t \\ y_1(t) = y_0 + (y_1 - y_0)t \end{cases} \quad t \in [0, 1]$$

Find the line through $\boldsymbol{p} = (\boldsymbol{x_0}, \boldsymbol{y_0})$ ***and*** $\boldsymbol{q} = (\boldsymbol{x_1}, \boldsymbol{y_1})$

- Parametric: $\Gamma(t), t \in (-\infty, \infty)$
- Implicit: $Ax + By + C = 0$
  - *Solve 2 equations in 2 unknowns (set $A^2+B^2=1$)*

Implicit Line

Explicit: $y = mx + b$ **Implicit:** $Ax + By + C = 0$

$$y = \frac{dy}{dx}x + b$$

$$dx \cdot y = dy \cdot x + dx \cdot b$$

$$0 = dy \cdot x - dx \cdot y + dx \cdot b$$

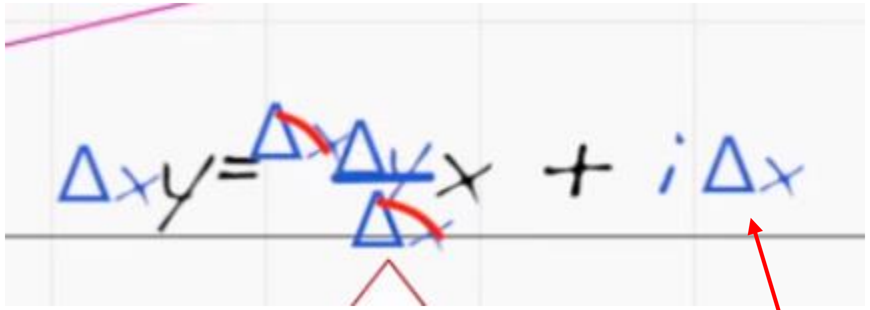
$$\Rightarrow A = dy, B = -dx, C = dx \cdot b$$

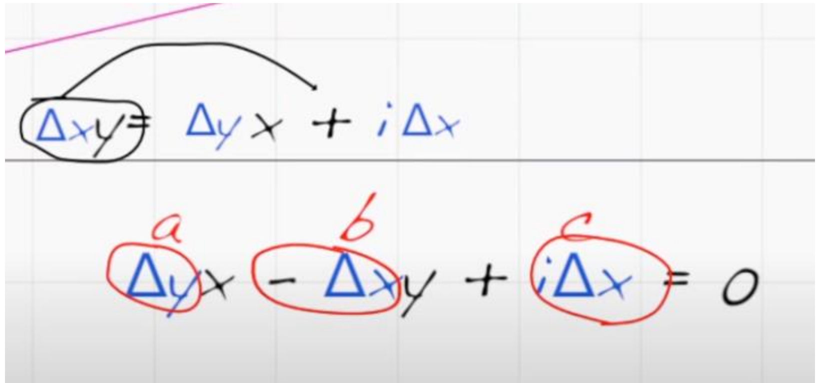
Example

$$y = \frac{-1}{3}x + 0$$

$$dx = -3, dy = 1, A = 1, B = 3, C = 0$$

$$\Leftrightarrow 1x + 3y = 0$$



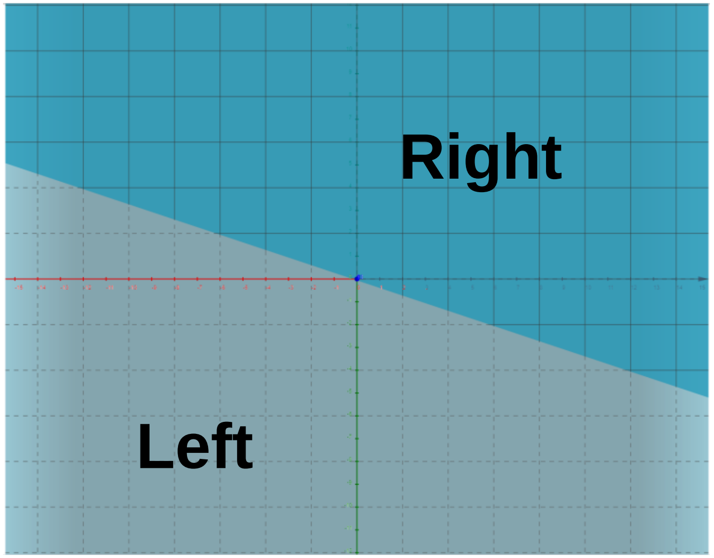


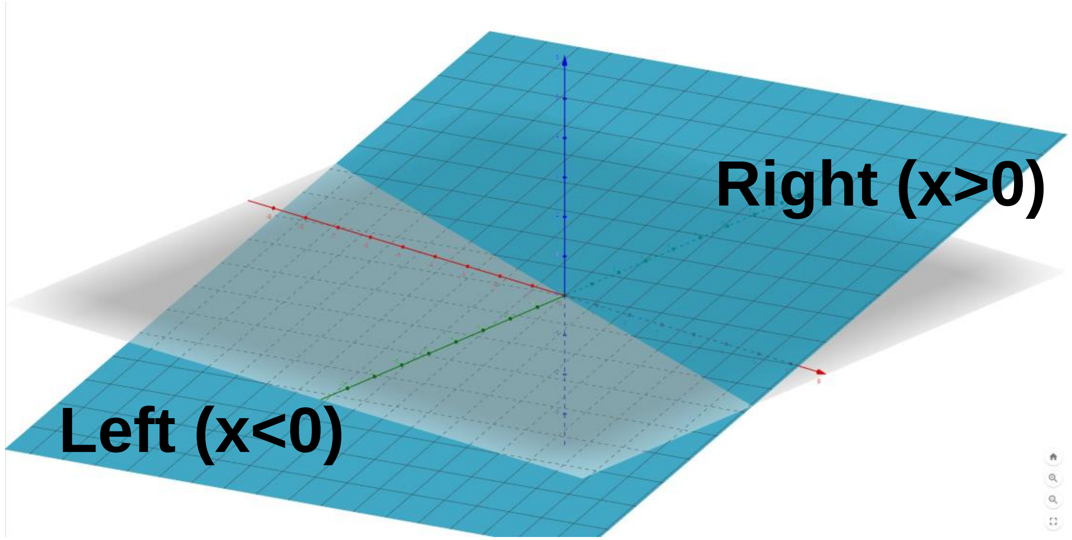
difference in x, (unrelated to the Laplace operator ∇^2, sometimes referred to by Δ)

Implicit Line – left or right?

Implicit line in 2D $\leftrightarrow$ ***Explicit plane in 3D***

$0.1x + 0.3y = 0 \quad \leftrightarrow \quad f(x, y) = 0.1x + 0.3y$

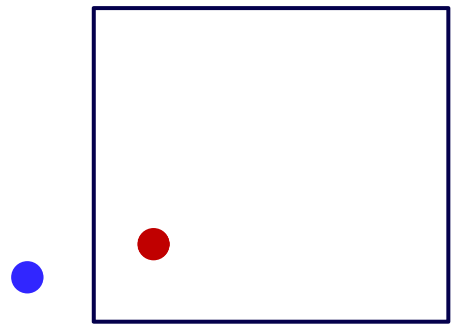




https://www.geogebra.org/3d

Point vs Line (Ray)

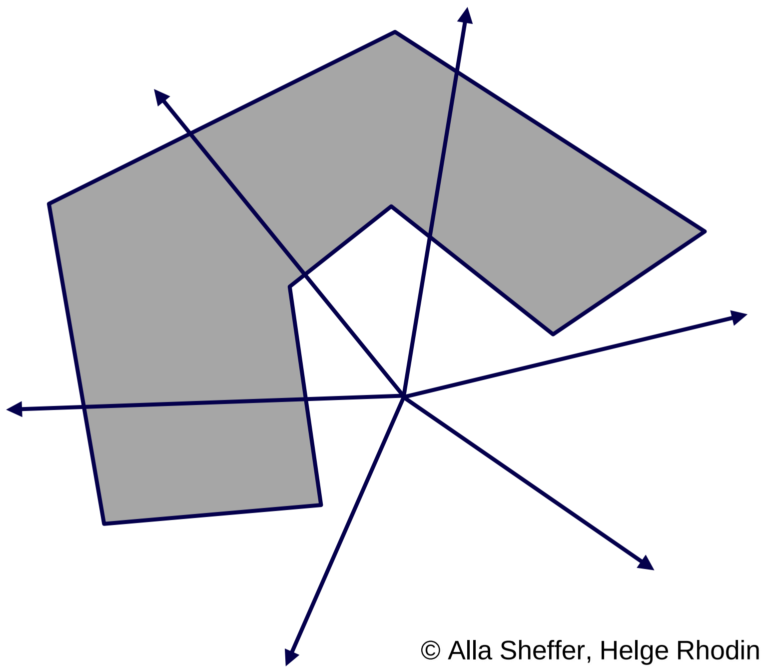
- Point $\mathbf{p} = (p_x, p_y)$
- Use implicit equation to determine coincidence & side
  - *Implicit* $Ax + By + C = 0$
  - *Solve 2 equations in 2 unknowns (third equation: set* $A^2 + B^2 = 1$*)*
  - *On:* $A \cdot p_x + B \cdot p_y + C = 0$
  - *Use same orientation to get consistent left/right orientation for inside test for lines defining CONVEX polygon*
    - Same sign implies inside
    - *Eg.* ***ALL*** $A \cdot p_x + B \cdot p_y + C < 0$



Recap: Inside Test?

- How to test if one poly is inside another?
- Use inside test for point(s)

- How?
  - *Convex Polygon*
    - Same side WRT to line equation (all sides)
  - *Non-Convex*
    - Subdivide, e.g., triangulate
      How?
    - Shoot rays in all directions
      (beware of corners and special cases)
    - Other ways?



Self-study:

Winding number algorithm

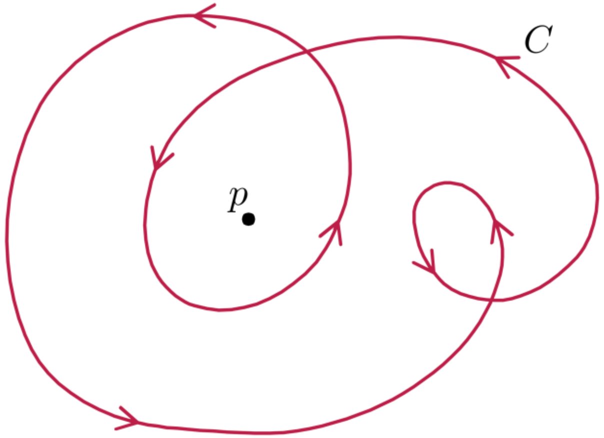
Point in polygon?

- If the winding number is non-zero

- How to compute the winding number?
  - http://geomalgorithms.com/a03-_inclusion.html

Winding number:

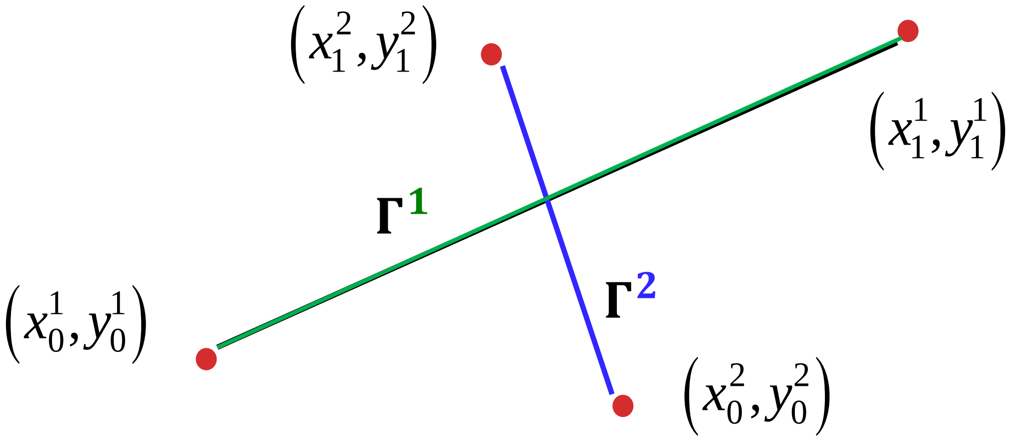
- the number of times that curve travels counterclockwise around the point
- negative if clockwise



Line-Line Intersection

$$\Gamma^1 = \begin{cases} x^1(t) = x_0^1 + (x_1^1 - x_0^1)t \\ y^1(t) = y_0^1 + (y_1^1 - y_0^1)t \end{cases} \quad t \in [0,1]$$

$$\Gamma^2 = \begin{cases} x^2(r) = x_0^2 + (x_1^2 - x_0^2)r \\ y^2(r) = y_0^2 + (y_1^2 - y_0^2)r \end{cases} \quad r \in [0,1]$$



Intersection: *x* & *y* values equal in both representations - two linear equations in two unknowns (*r*,*t*)

$$x_0^1 + (x_1^1 - x_0^1)t = x_0^2 + (x_1^2 - x_0^2)r$$
$$y_0^1 + (y_1^1 - y_0^1)t = y_0^2 + (y_1^2 - y_0^2)r$$

Question: What is the meaning if the solution gives *r*,*t* < 0 or *r*,*t* >1 ?

Question: What is the meaning of $r,t < 0$ or $r,t > 1$?

A. They still collide
B. They do not collide
C. They may or may not collide – need more testing

Efficiency

- Naïve implementation
  - *Test each moving object against ALL other objects at each step*
  - *Horribly expensive*

- How to speed up?

Efficiency

- Naïve implementation
  - *Test each moving object against ALL other objects at each step*
  - *Horribly expensive*

- Speed up
  - *Bounding Volumes*
  - *Hierarchies*

Bounding volumes

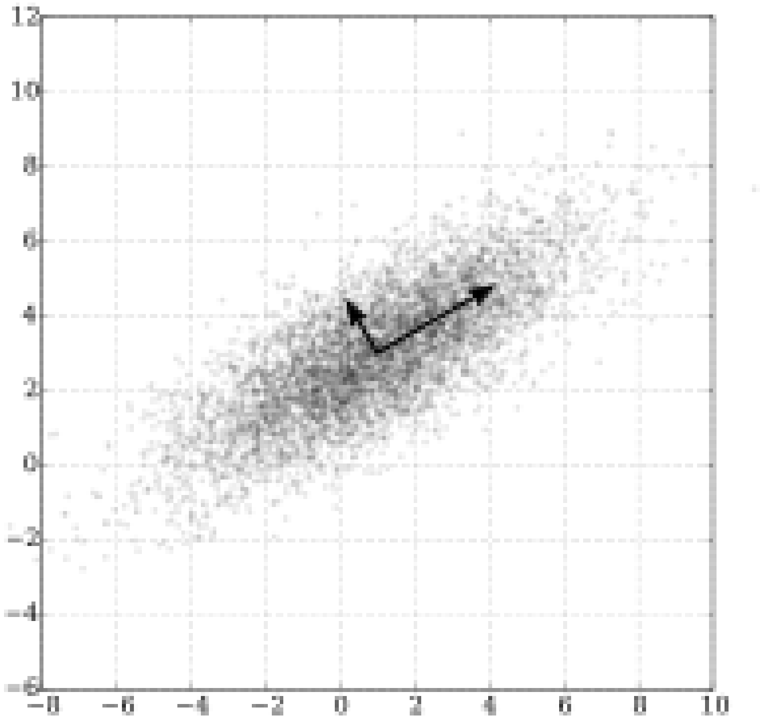
- Axis aligned bounding box (AABB)
  - *+ Trivial to compute*
  - *+ Quick to evaluate*
  - *- May be too big…*

- Tight bounding box
  - *- Harder to compute: Principal Component Analysis (PCA)*
  - *- Slightly slower to evaluate*
  - *- Compact*

Principle Component Analysis (PCA)

Derive the directions of maximum variance

$$\mathbf{w}_{(1)} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \sum_i \left(\mathbf{x}_{(i)} \cdot \mathbf{w}\right)^2 \right\}$$



Wikipedia

Bounding volumes

- Bounding circle
  - *A range of efficient (non-trivial) methods*

- Convex hull
  - *Gift wrapping & other methods…*

Bounding Volume Intersection

- Axis aligned bounding box (AABB)
  - *A.LO<=B.HI && A.HI>=B.LO (for both X and Y)*

    lower   *higher*

- Circles
  - $||A.C - B.C|| < A.R + B.R$

    Center   *Radius*

Moving objects

- Sweep – test intersections against before/after segment
  - *Avoid "jumping through" objects*
  - *How to do efficiently?*

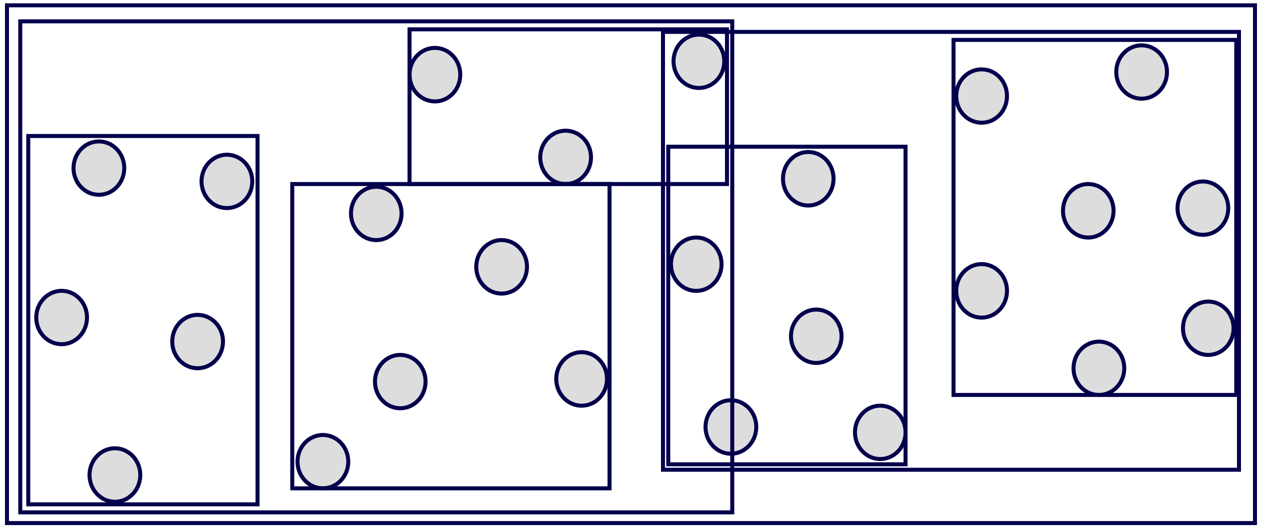
- Boxes?

- Spheres?

Hierarchical Bounding Volumes

Bound Bounding Volumes:

- Use (hierarchical) bounding volumes for groups of objects

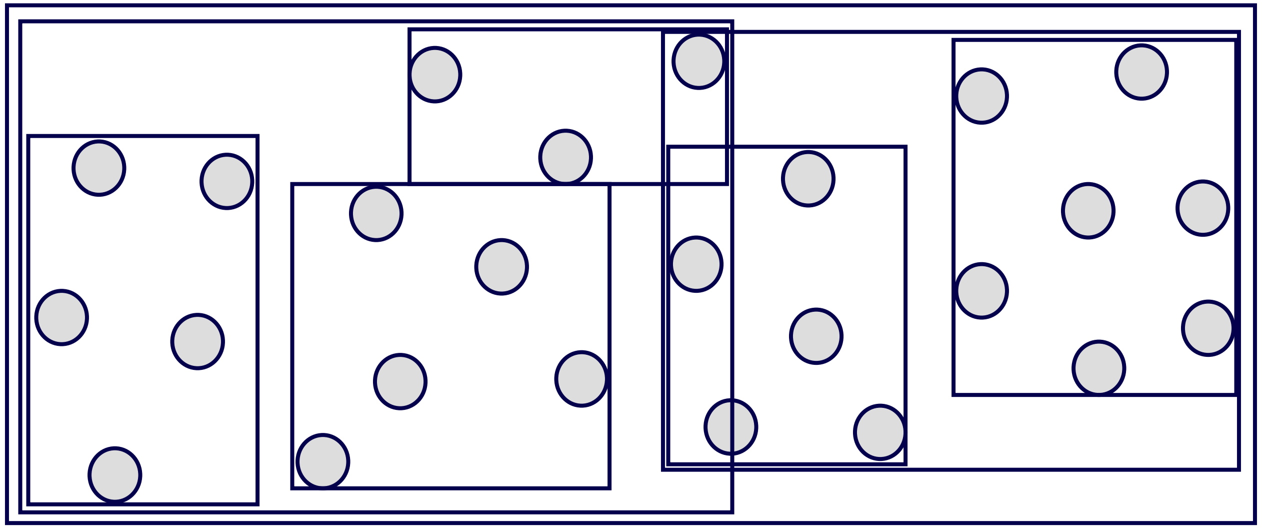


- How to group boxes?
  - *Closest*
  - *Most jointly compact (how?)*

Hierarchical Bounding Volumes

Bound Bounding Volumes:

- Use (hierarchical) bounding volumes for groups of objects



- Challenge: dynamic data…
  - *Need to update hierarchy efficiently*

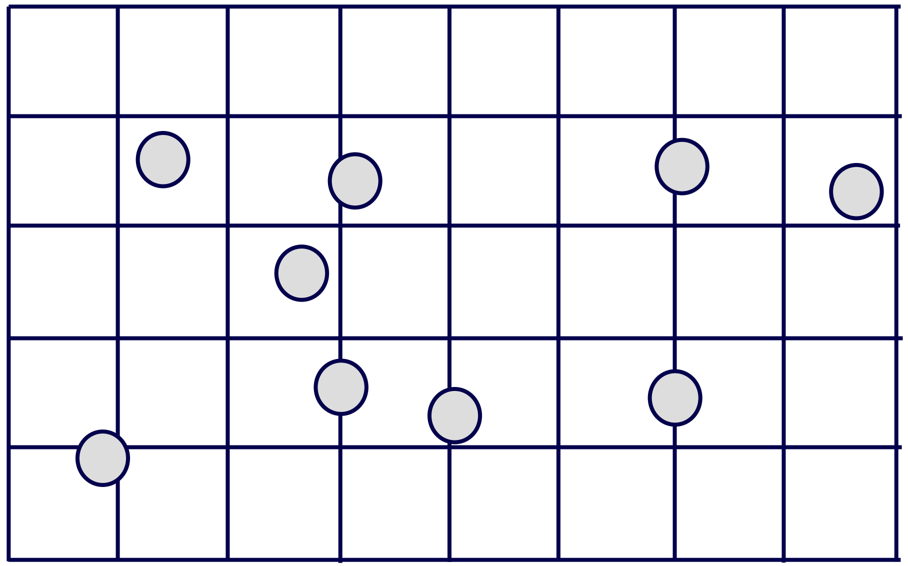
Spatial Subdivision DATA STRUCTURES

- Subdivide space (bounding box of the “world”)
- Hierarchical
  - *Subdivide each sub-space (or only non-empty sub-spaces)*
- Lots of methods
  - *Grid, Octree, k-D tree, (BSP tree)*

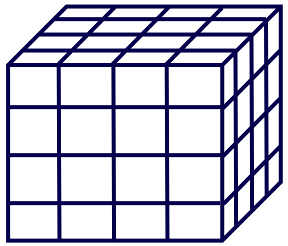
Regular Grid

Subdivide space into rectangular grid:

- Associate every object with the cell(s) that it overlaps with
- Test collisions only if cells overlap



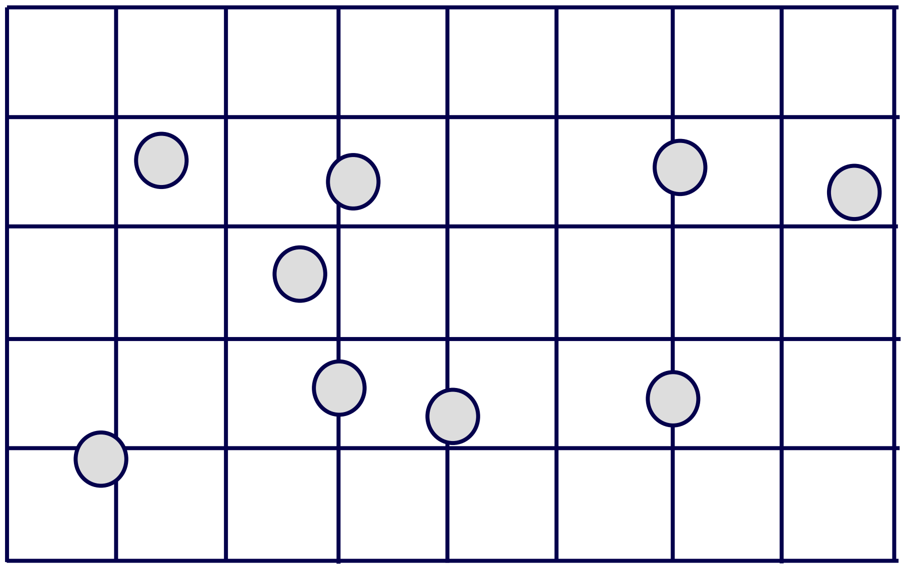
In 3D: regular grid of cubes (voxels):



Creating a Regular Grid

Steps:

- Find bounding box of scene
- Choose grid resolution in x, y, z
- Insert objects
- Objects that overlap multiple cells get referenced by all cells they overlap



Regular Grid Discussion

Advantages?

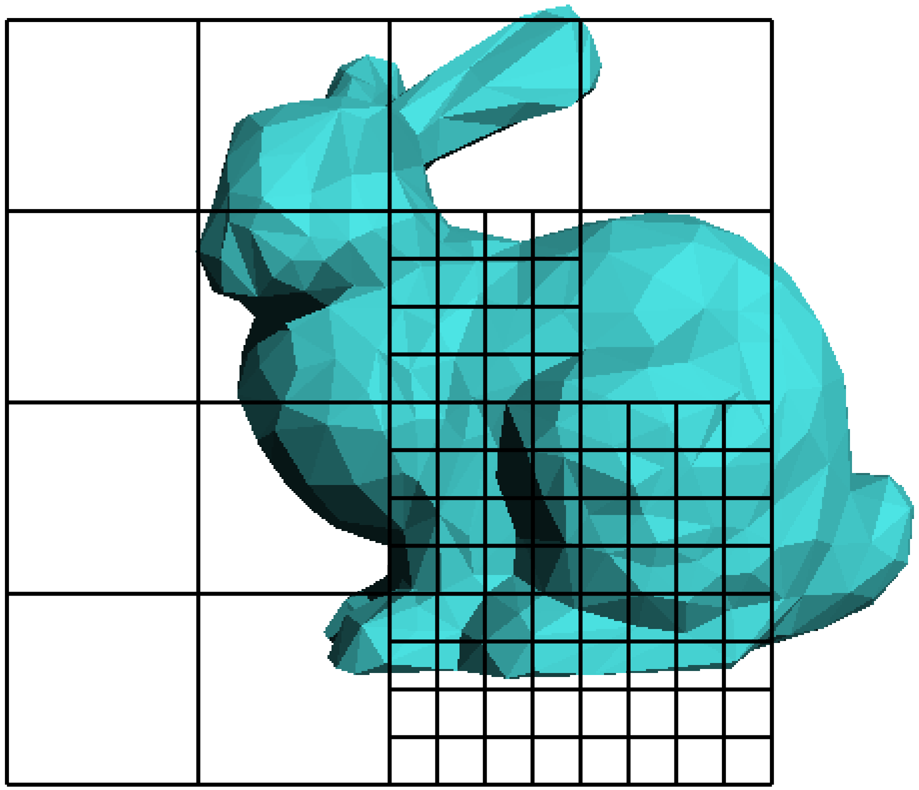
- Easy to construct
- Easy to traverse

Disadvantages?

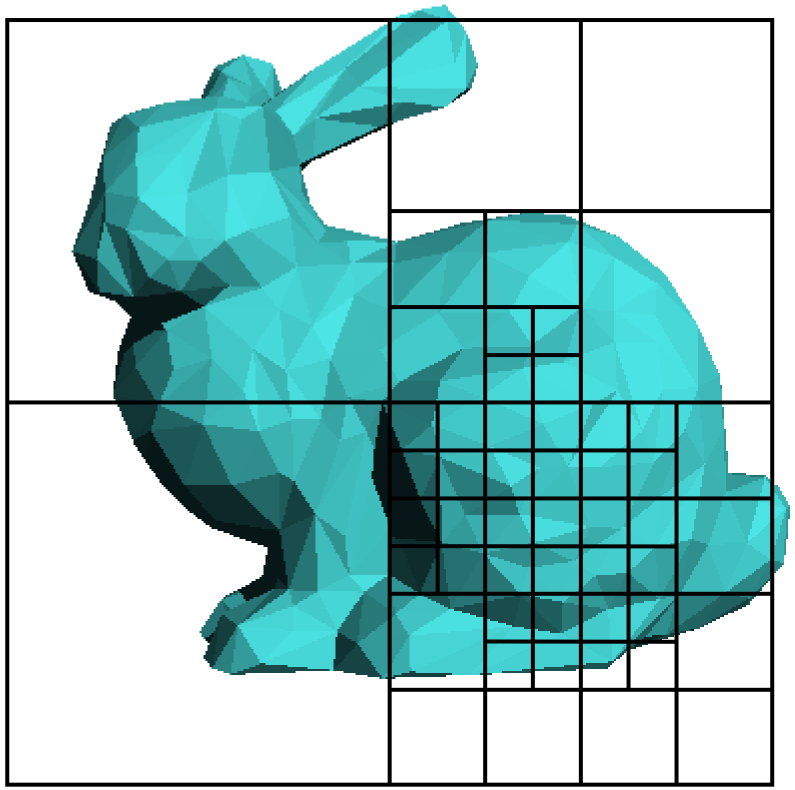
- May be only sparsely filled
- Geometry may still be clumped

Adaptive Grids

- **Subdivide until each cell contains no more than *n* elements, or maximum depth *d* is reached**



Nested Grids



Octree/(Quadtree)

- This slide is curtsey of Fredo Durand at MIT

Collision Resolution

Today: simplified example

Upcoming lecture:

Physics-based simulation

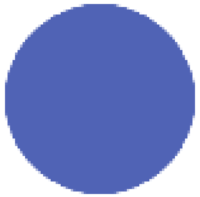
Basic Particle Simulation (first try)

How to compute the change in velocity?

$$d_t = t_{i+1} - t_i$$

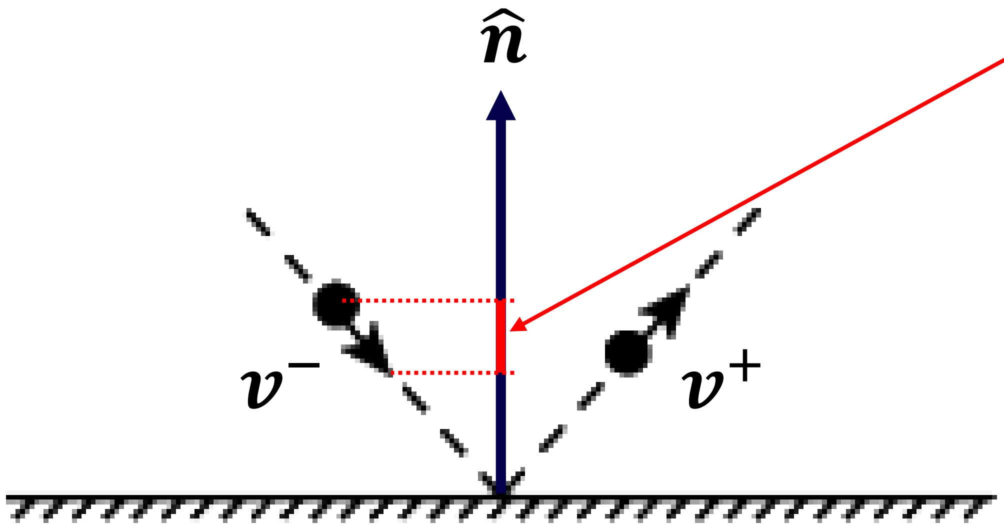
$$\vec{\boldsymbol{v}}_{i+1} = \vec{\boldsymbol{v}}_i + \Delta v$$

$$\vec{\boldsymbol{p}}_{i+1} = \vec{\boldsymbol{p}}(t_i) + \vec{\boldsymbol{v}}_i d_t$$



Particle-Plane Collisions

- ***Change in direction of normal***



Velocity along normal
(v projected on normal
by the dot product)

Frictionless

Apply change
along normal
(magnitude
times direction)

$$\Delta v = 2(v^- \circ \hat{n})\hat{n}$$

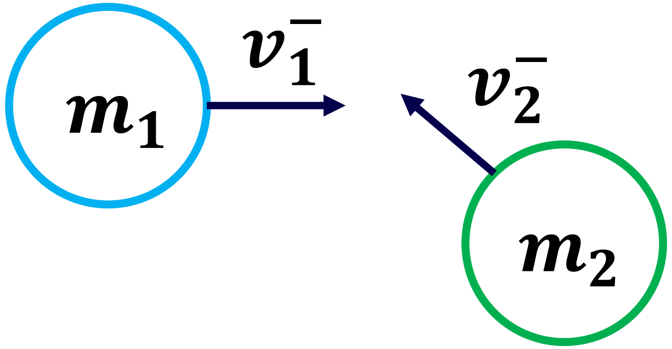
$$v^+ = v^- + \Delta v$$

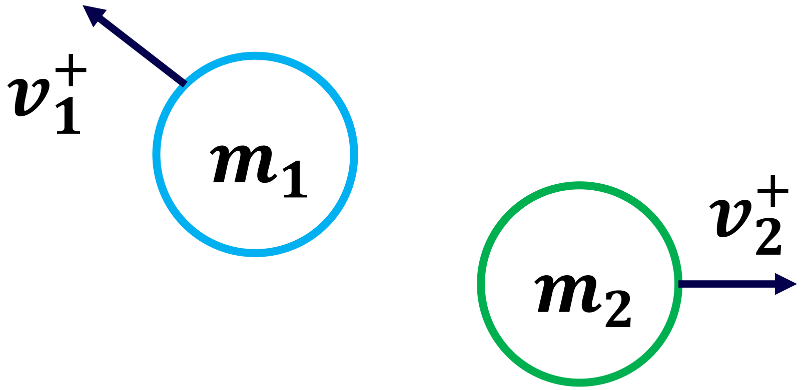
Loss of energy

$$\Delta v = (1 + \epsilon)(v^- \circ \hat{n})\hat{n}$$

Particle-Particle Collisions (spherical objects)

Before collision **After**





Response:

$$v_1^+ = v_1^- - \frac{2m_2}{m_1 + m_2} \frac{\langle v_1^- - v_2^- \rangle \cdot \langle p_1 - p_2 \rangle}{\|p_1 - p_2\|^2} \langle p_1 - p_2 \rangle$$

$$v_2^+ = v_2^- - \frac{2m_1}{m_1 + m_2} \frac{\langle v_2^- - v_1^- \rangle \cdot \langle p_2 - p_1 \rangle}{\|p_2 - p_1\|^2} \langle p_2 - p_1 \rangle$$

- This is in terms of velocity
- Upcoming lectures: derivation via impulse and forces