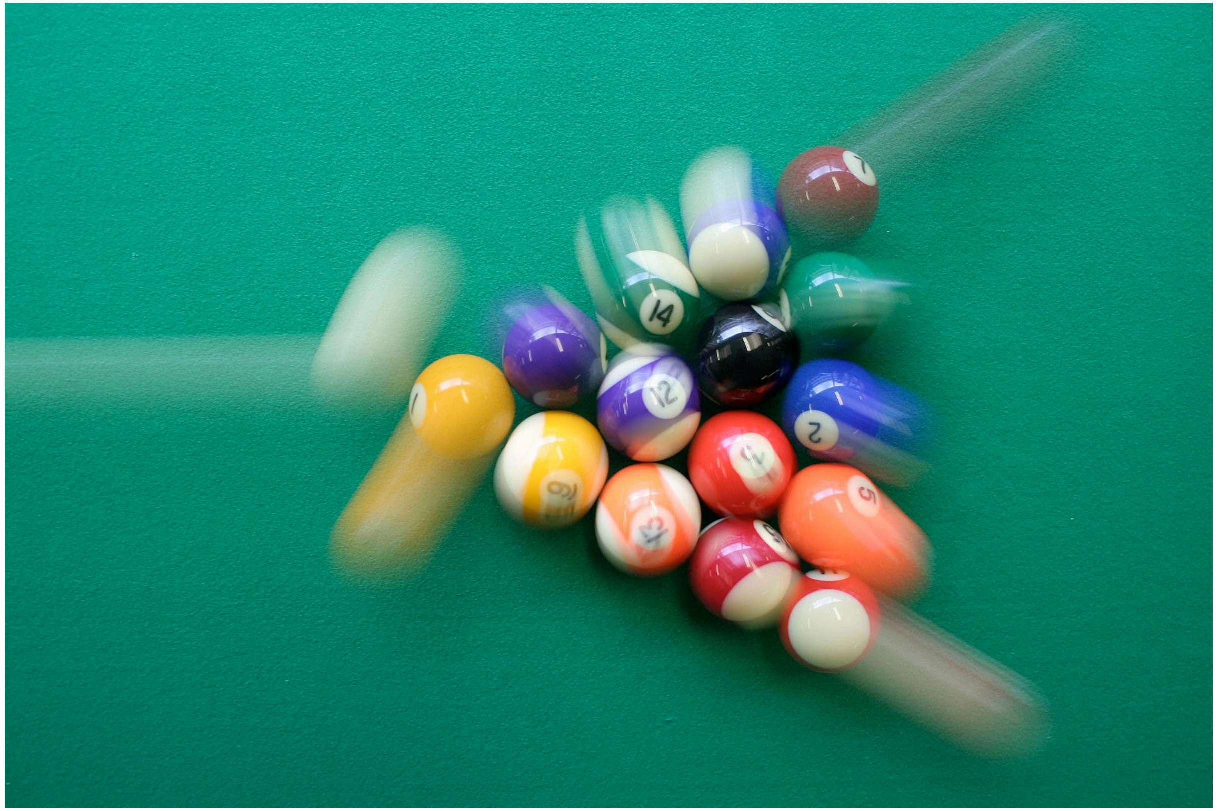
CPSC 427
Video Game Programming

Collisions (and a bit of physics)



Helge Rhodin

Objectives

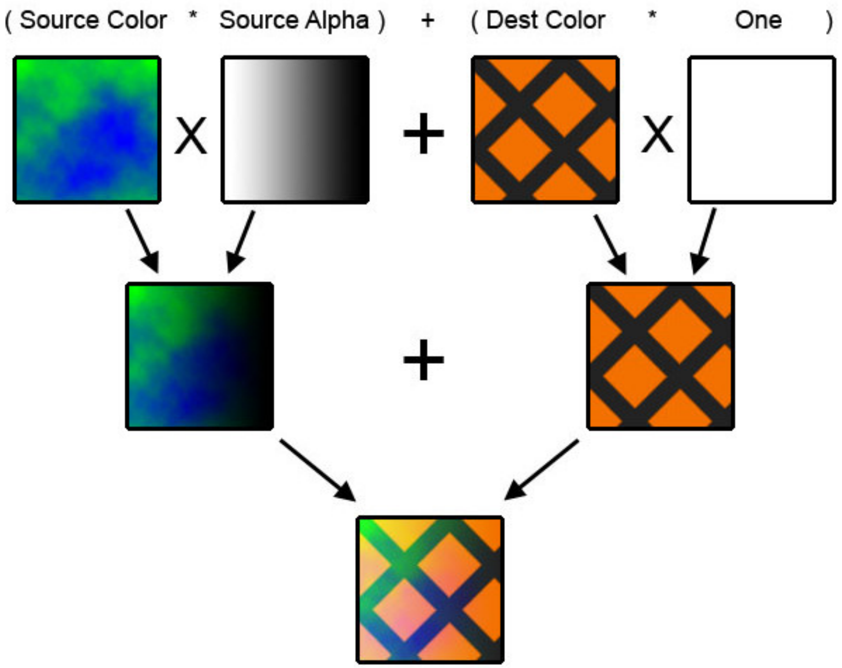
- *How to mathematically test that two objects intersect?*
- *How to implement intersection?*
- *Learn about points, lines, polygons*

Future lectures: How to resolve collisions

Recap: Blending

- Controls how pixel color is blended into the FBO's Color Attachment
- Control on factors and operation of the equation
- RGB and Alpha are controllabe separately

$$RGB_o = RGB_{src} * F_{src}\ [+\ \text{-}\ /\ *]\ RGB_{dst} * F_{dst}$$

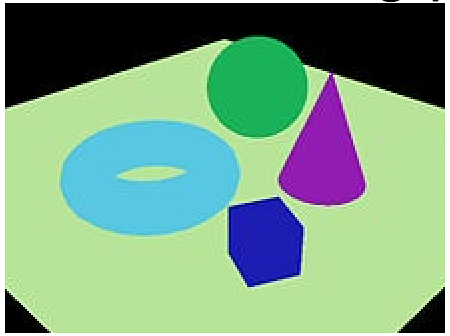


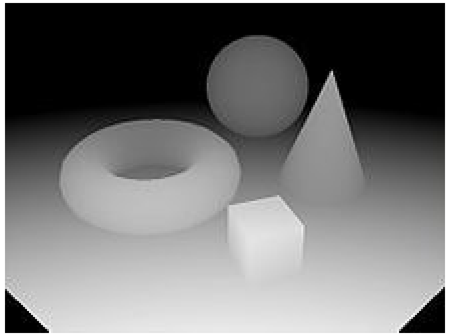
Cloud (source) on top of grid (dest)

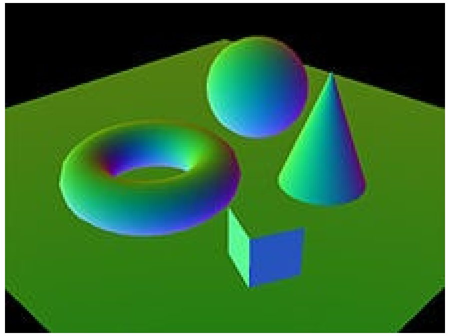
Recap: Two-pass rendering

- ***Deferred shading (a form of screen-space rendering)***

First rendering pass

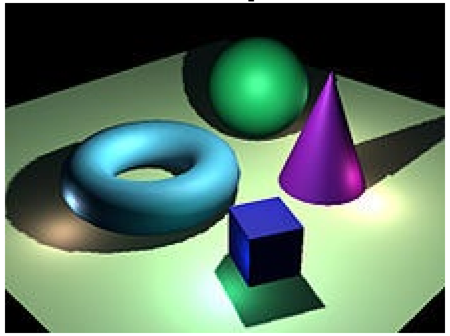






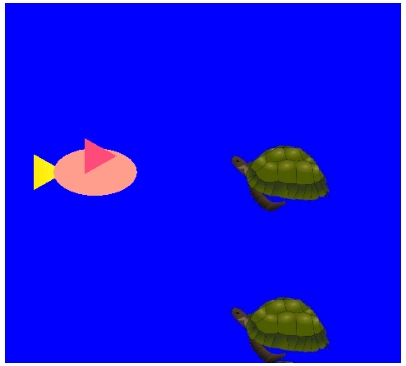
Input

Second pass

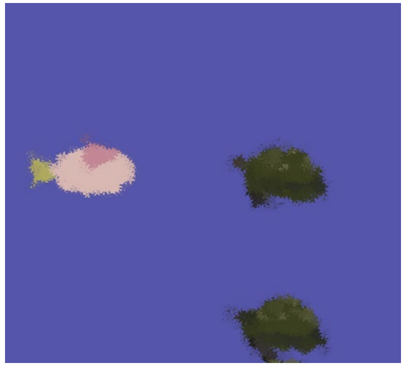


- ***or water effects***

First pass

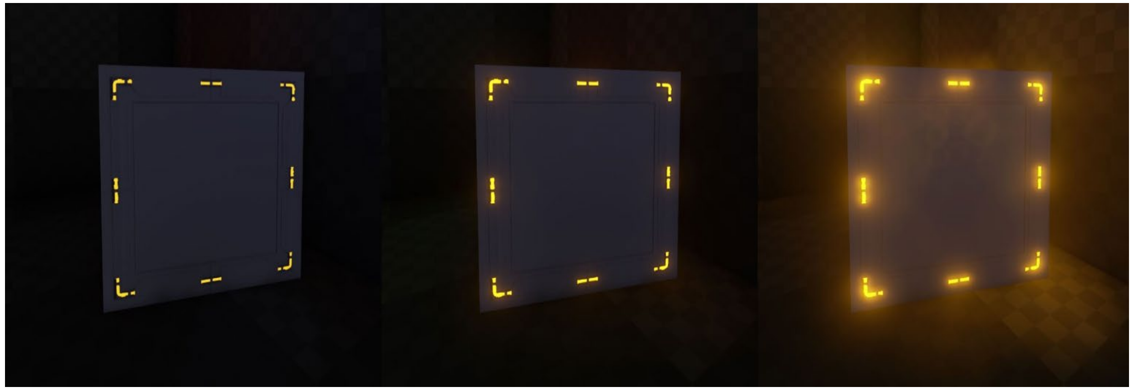


Second pass



Post-processing: Bloom

- Fullscreen Effect to highlight bright areas of the picture
- Post-processing: Operates on Images after the scene has been rendered

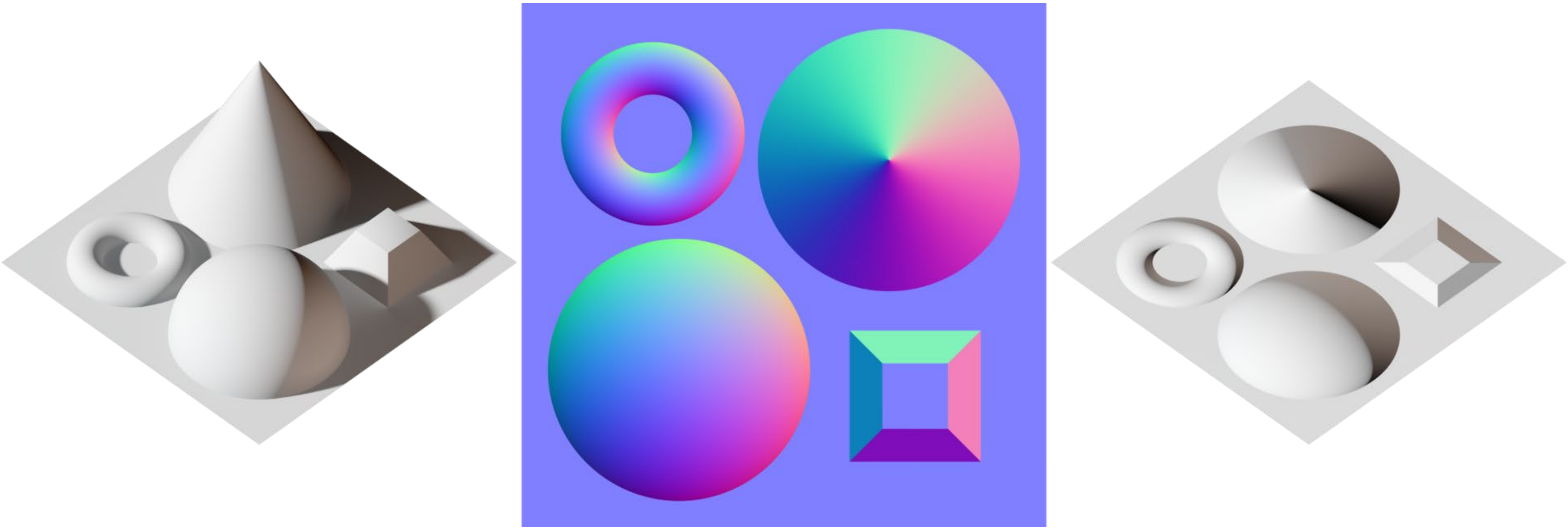


- High level overview:

1. Render scene to texture
2. Extract bright regions by thresholding
3. Gaussian blur pass on the bright regions
4. Combine original texture and highlights texture with additive blending

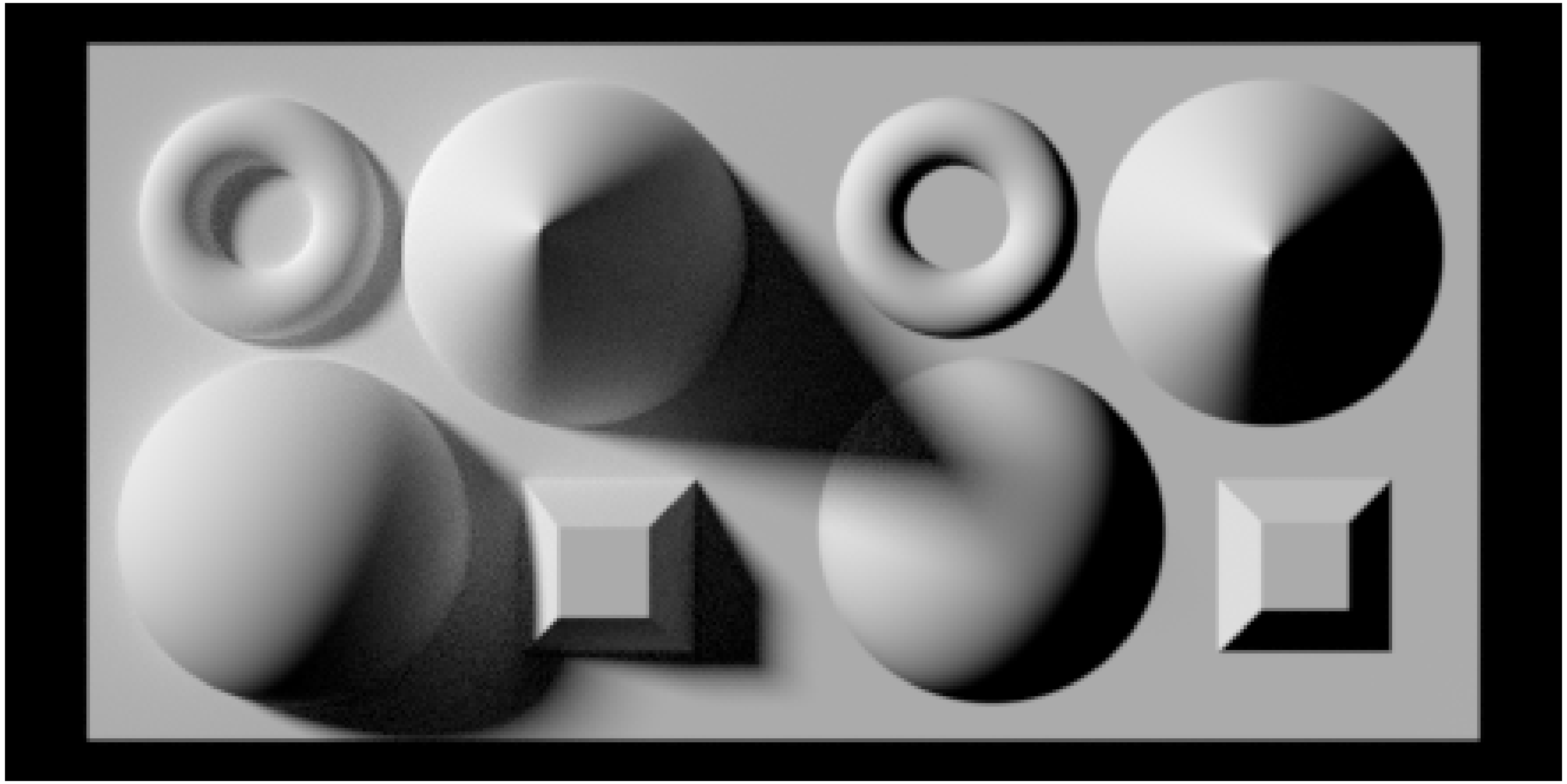
Normal maps

A way to fake 3D details



Perfect for illumination in 2D games

- *What do you observe?*



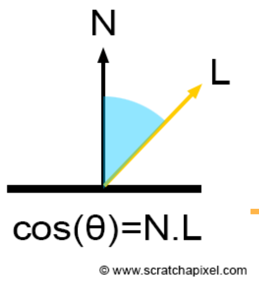
How to implement?

Either:

1. ***Include shading into your fragment shader***
   - Load and sample from RGB texture
   - Load and sample from normal map (the new aspect)
   - Compute shading

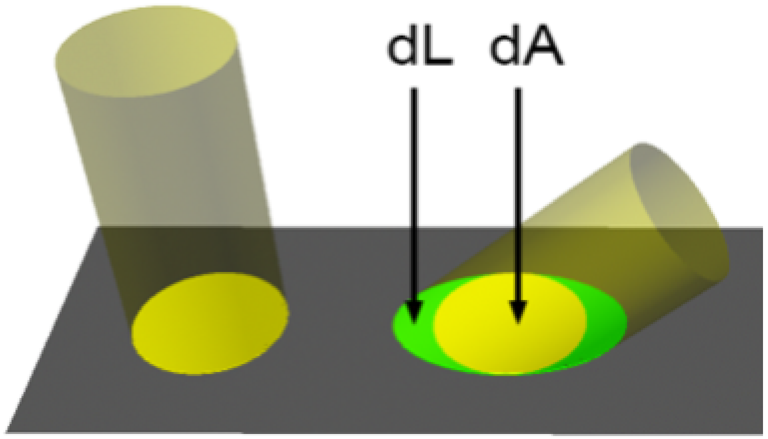
2. ***Two-pass rendering***
   - Render color in one pass
   - Render the normal in a second pass
   - Compute shading in a separate pass, as for the water shader

Shading equation?



- ***Single light source:***
  - *Dot product of normal and light direction*
    - Light direction: computed from light source (L) and pixel location (x)
    - Normal direction: load from normal map

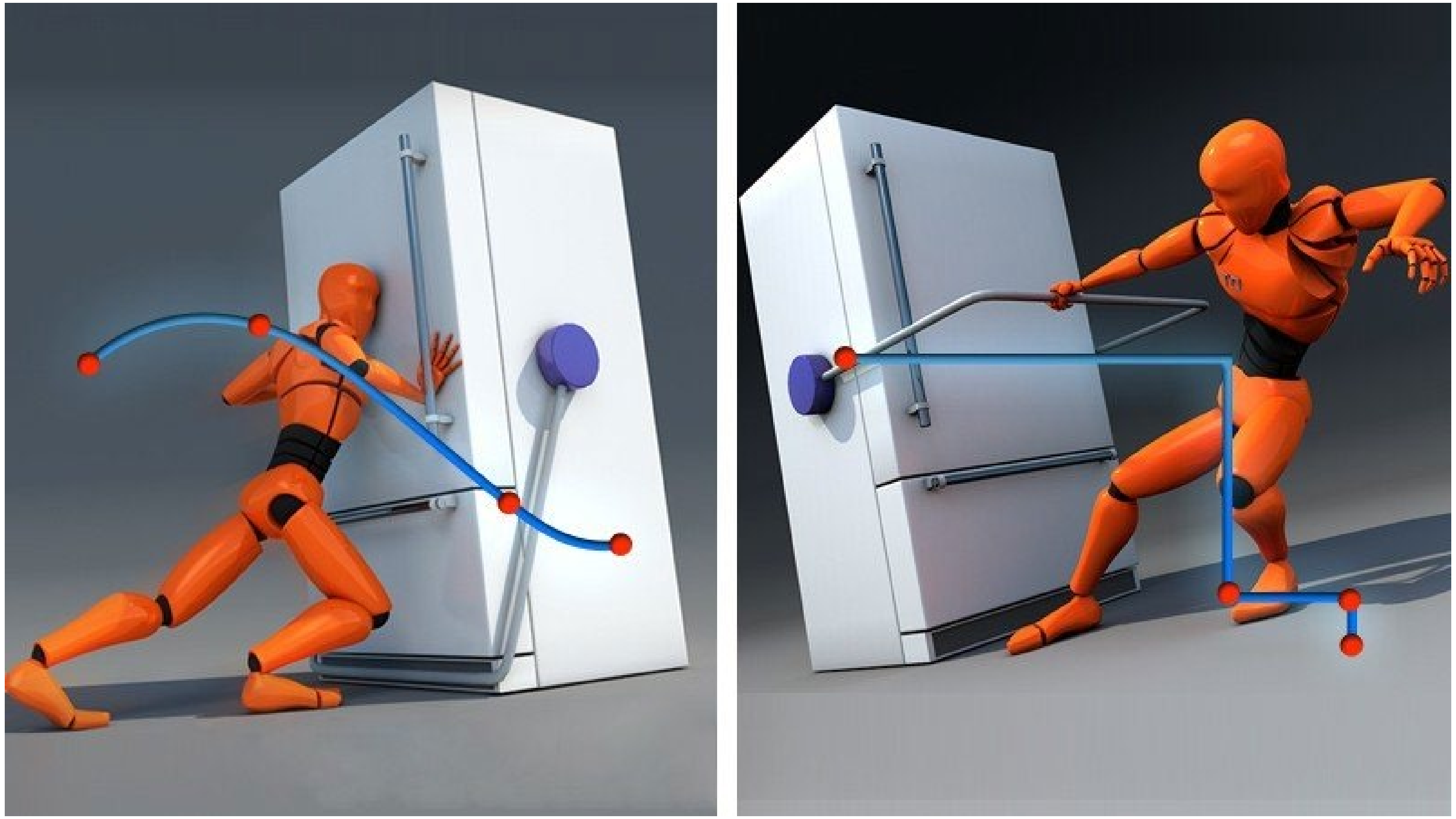
*color = texture(x) * dot(normal(x), normalized(x-L))*



- Multiple light sources? Specular highlights?

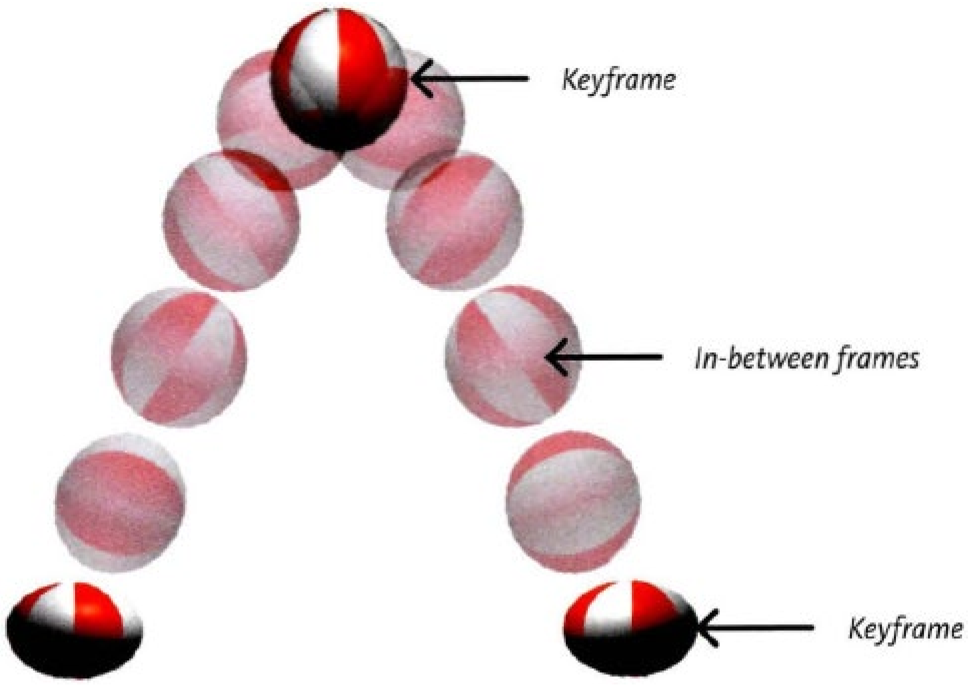
CPSC 427
Video Game Programming

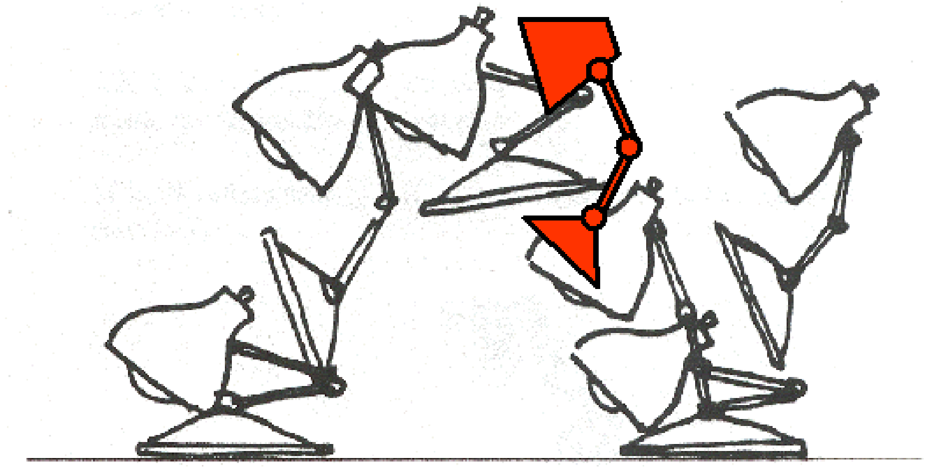
Curves and Animation



https://www.pluralsight.com/blog/film-games/stepped-vs-spline-curves-blocking-animation

Keyframe animation





Lasseter `87

Line equation

Parametric form

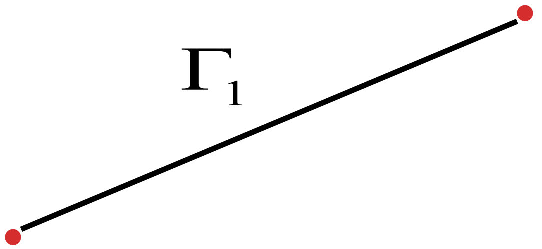
- 3D: x, y, and z are functions of a parameter value t

$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1-t)$$

What things can we interpolate?

Line segment

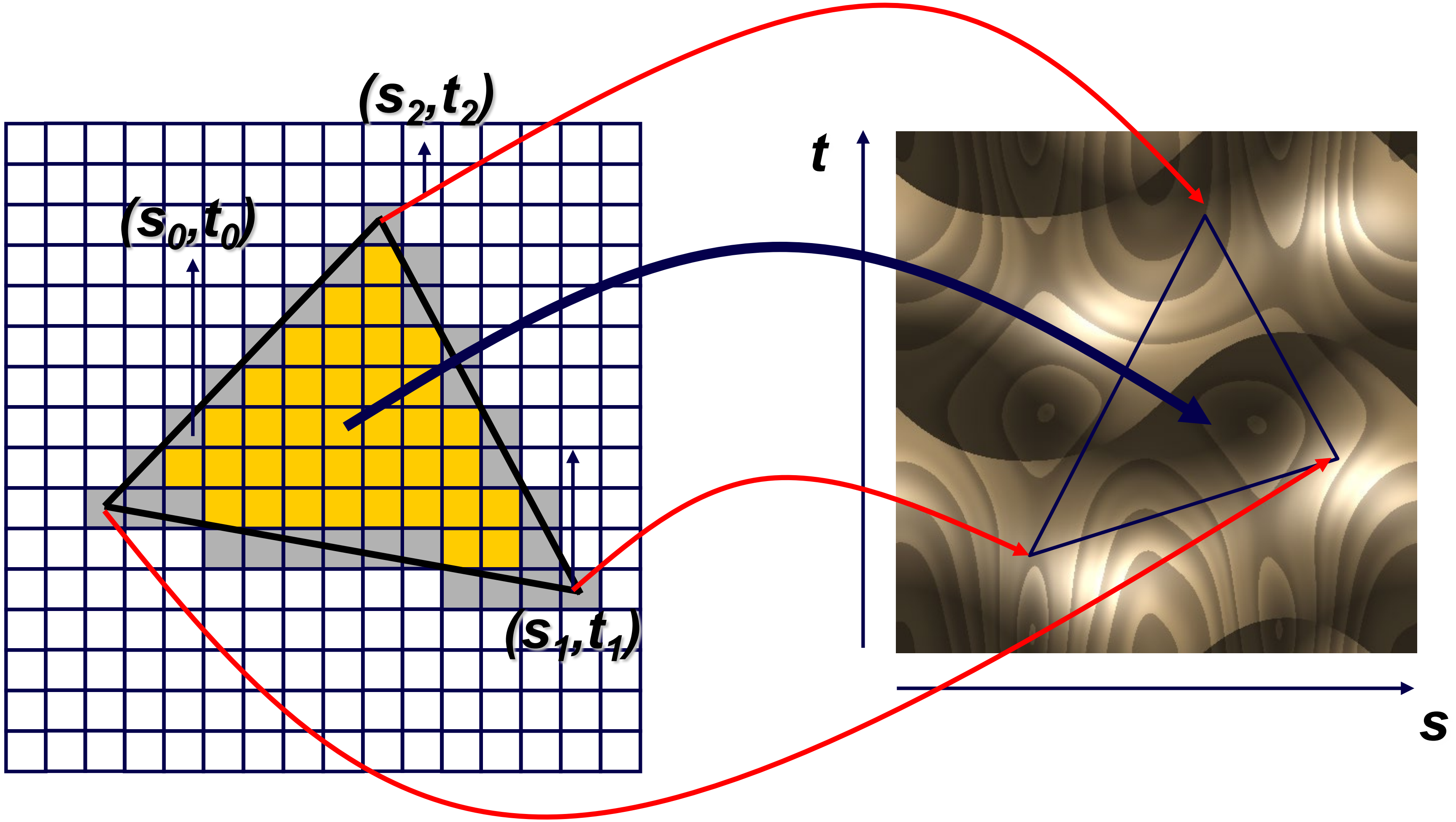
$$P_1 = \left(x_1^1, y_1^1\right)$$



$$P_0 = \left(x_0^1, y_0^1\right)$$

$$G_1 = \begin{cases} x^1(t) = x_0^1 + \left(x_1^1 - x_0^1\right)t \\ y^1(t) = y_0^1 + \left(y_1^1 - y_0^1\right)t \end{cases} t \in [0,1]$$

Recap: Texture mapping



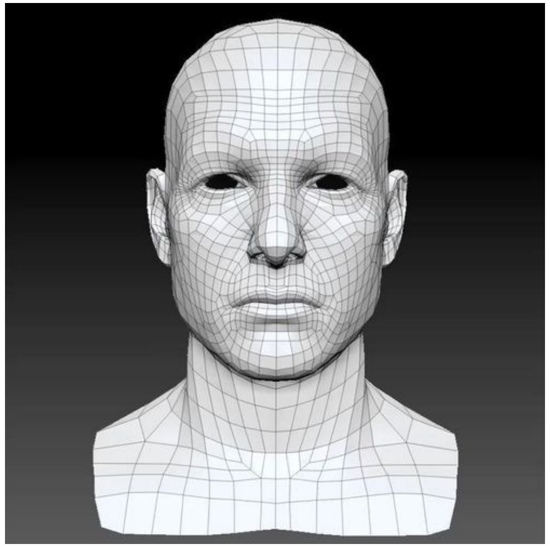
Interpolating general properties

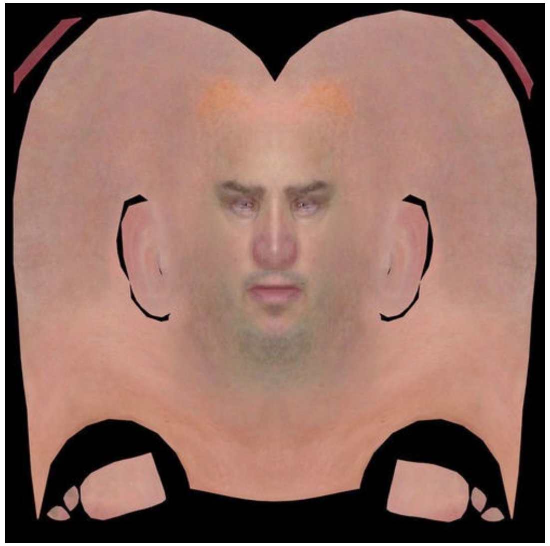
- *position*
- *aspect ratio?*
- *scale*
- *color*
- *What else?*

$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1-t)$$

$$s^0 \qquad s^1$$

$$c^0 \qquad c^1$$





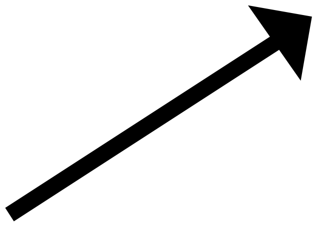
Barycentric coordinates / interpolation

Other Parametric Functions

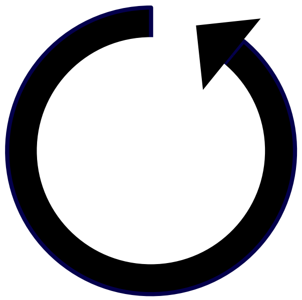
$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1-t)$$

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

Line segment



Circle (arc)



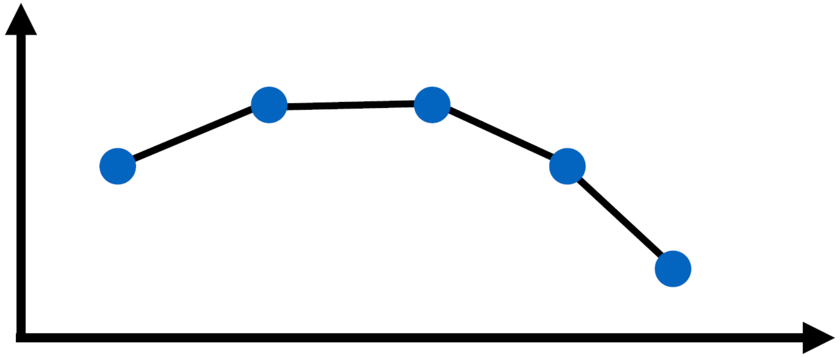
?

Future lecture: Splines

Segments of simple functions

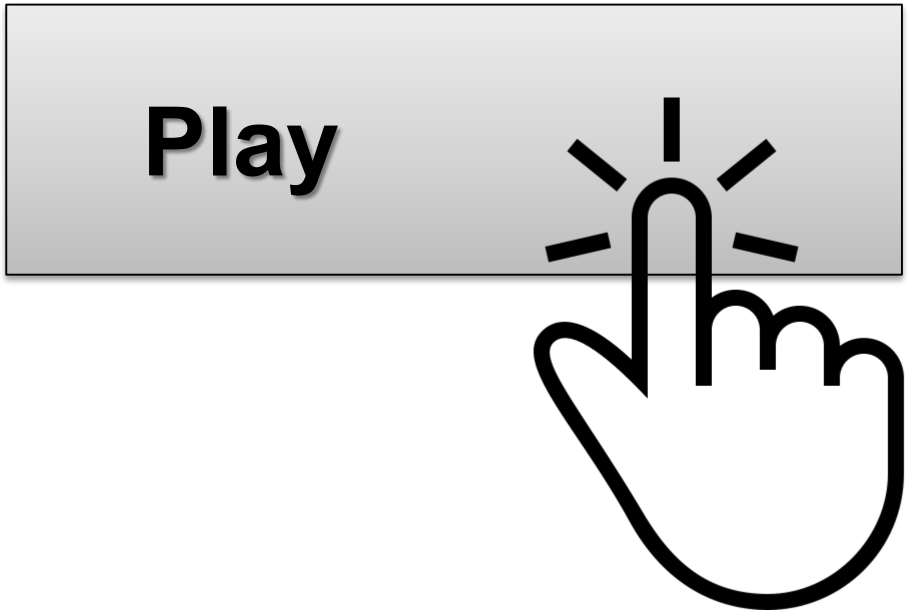
$$f(x) = \begin{cases} f_1(x), & \text{if } x_1 < x \le x_2 \\ f_2(x), & \text{if } x_2 < x \le x_3 \\ \vdots & \vdots \\ f_n(x), & \text{if } x_n < x \le x_{n+1} \end{cases}$$

E.g., linear functions



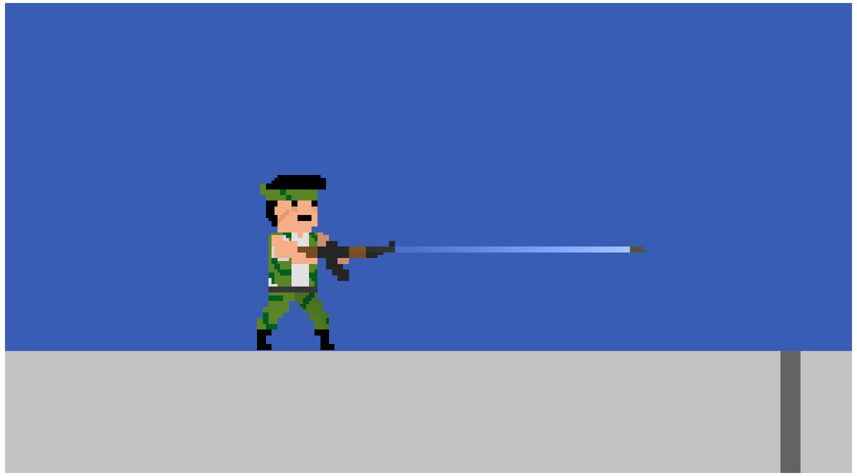
Collision Motivation: Object selection

- *Point inside object boundary?*



Motivation: Bullet trajectories

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

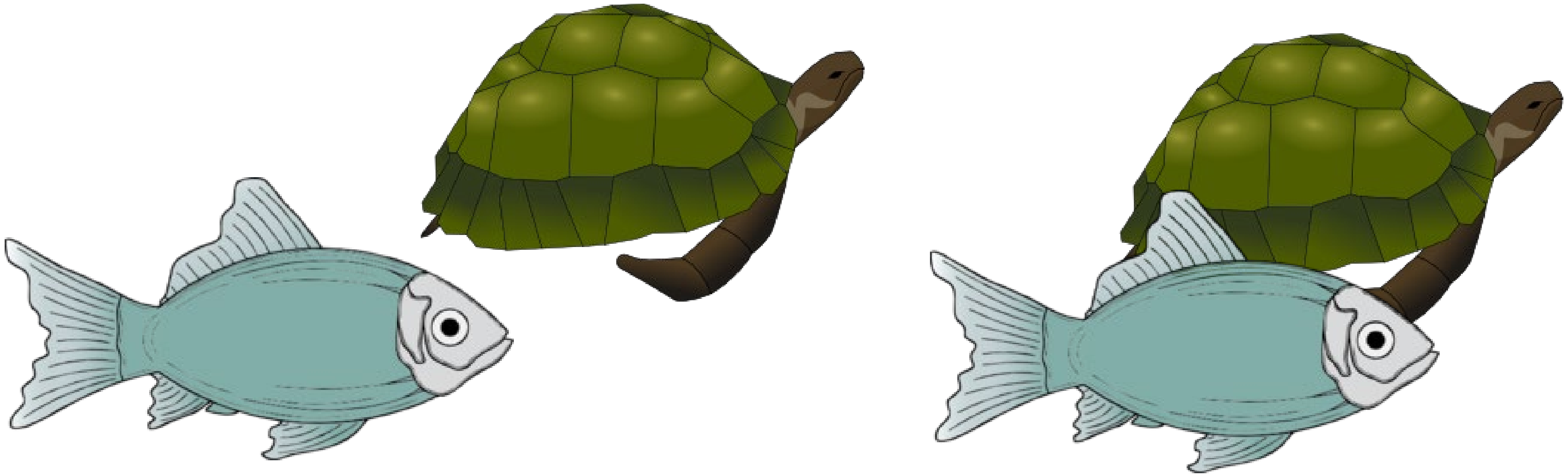


https://forum.unity.com/threads/2d-platformer-shooting.365971/

Motivation: Collision

- *Prevent object penetration*

- *How?*



Collision Configurations?

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

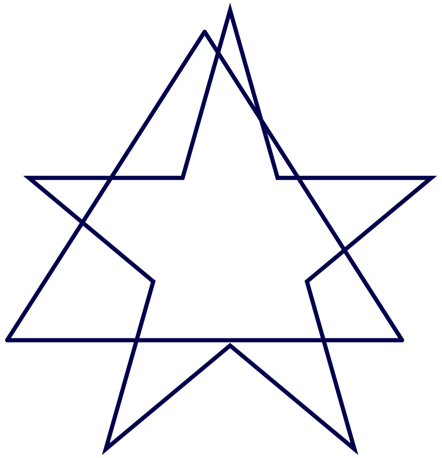
A. True

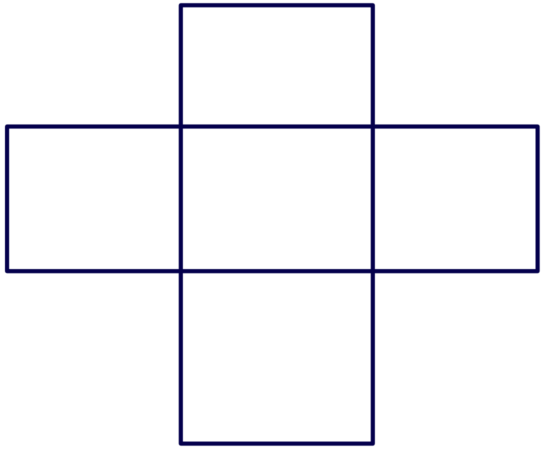
B. False

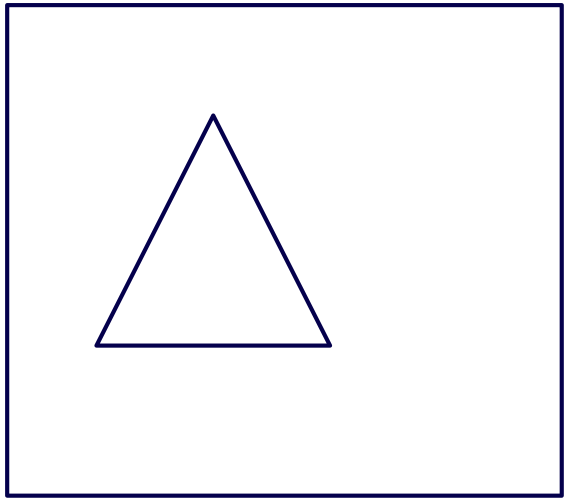
Collision Configurations?

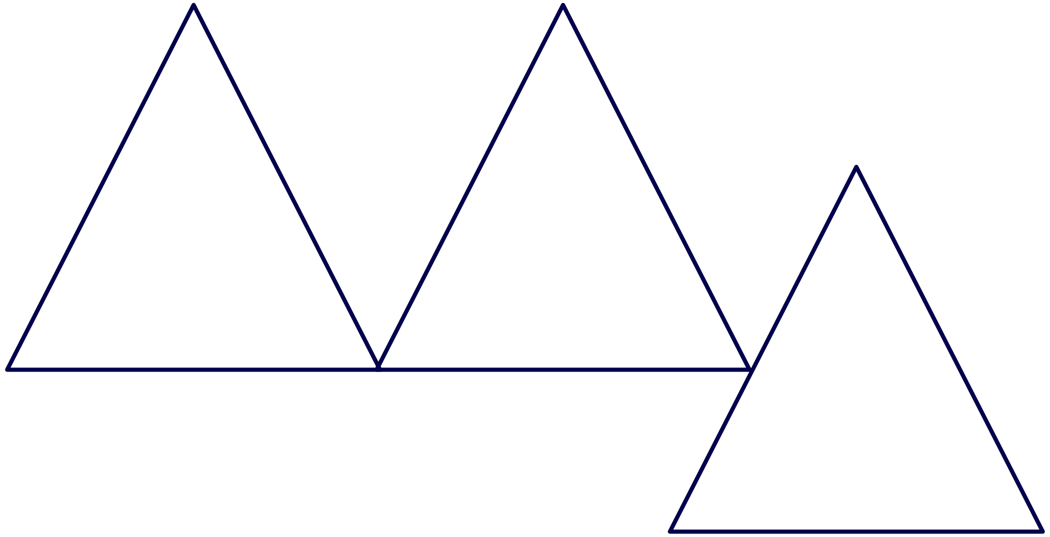
- Segment/Segment Intersection
  - *Point* ***on*** *Segment*

- Polygon inside polygon



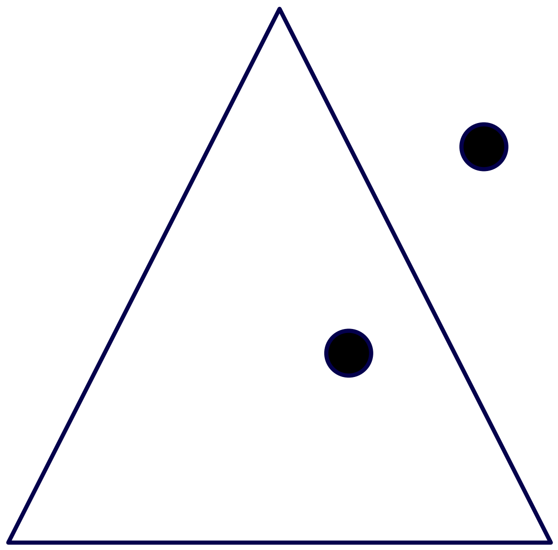






Inside Test?

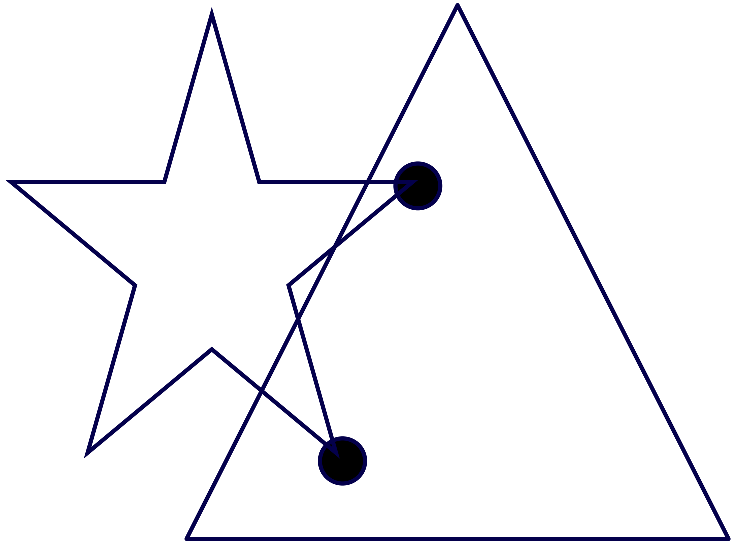
- How to test if one poly is inside another?
- Use inside test for point(s)
- How?
    - *Convex Polygon*
        - Same side WRT to line (all sides)
    - *Non-Convex*
        - Subdivide= triangulate (not that easy)
        - Shoot rays (beware of corners and special cases)



Collision Test?

- How to test if one poly collides with another?

- Use inside test for points on vertices



Resources

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

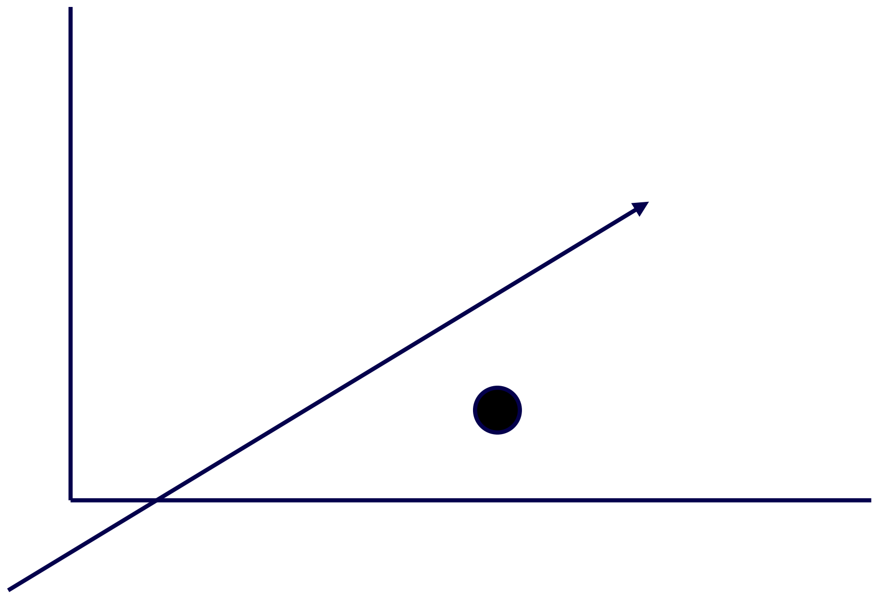
Curves

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*
- **Issues?**



Left or right?

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)

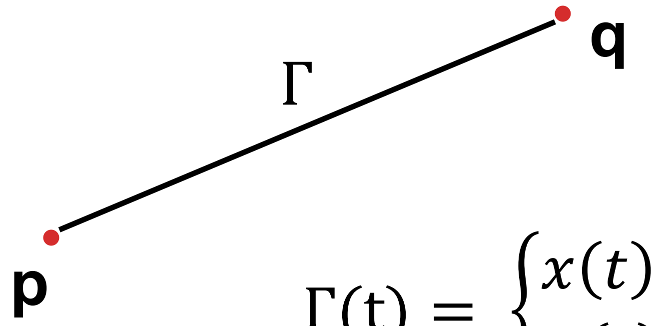
$$C(t) \coloneqq \begin{pmatrix} cos(t) \\ sin(t) \end{pmatrix}$$

Circle (arc)

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

Lines & Segments

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



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

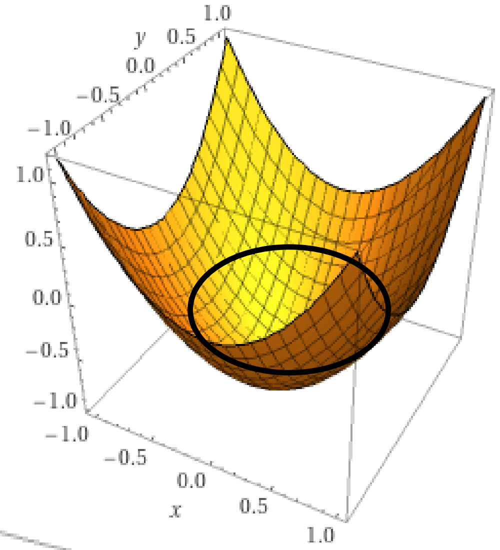
How to determine if left or right of the line?

Implicit Function

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

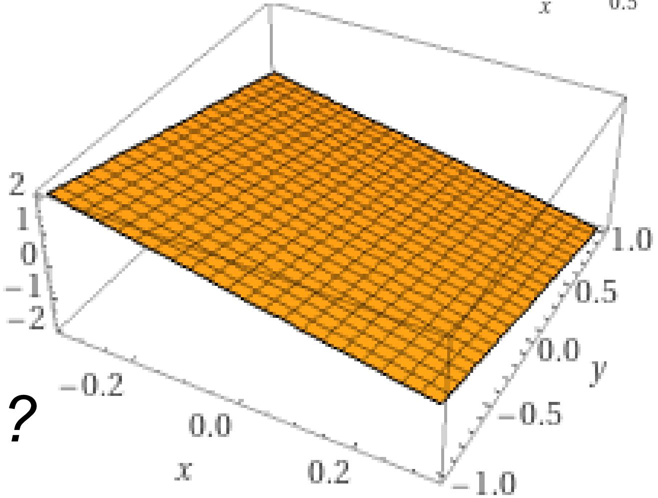
E.g:

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



How to determine if inside circle?

$$S(x,y):\quad -3x - y = 0$$

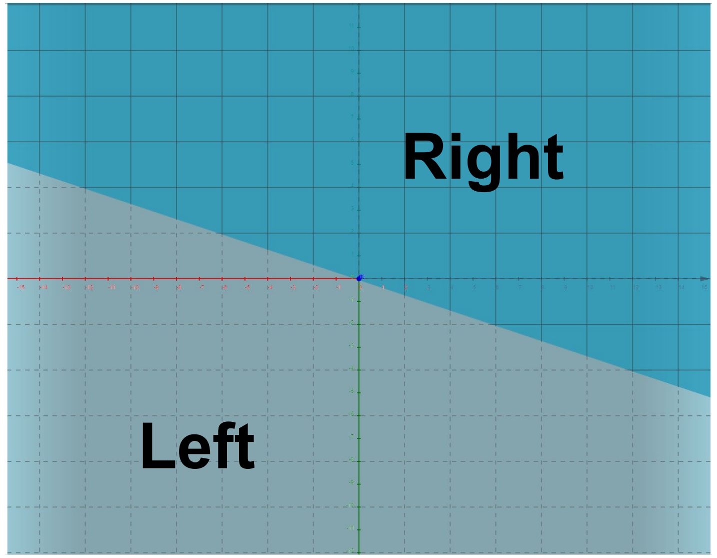


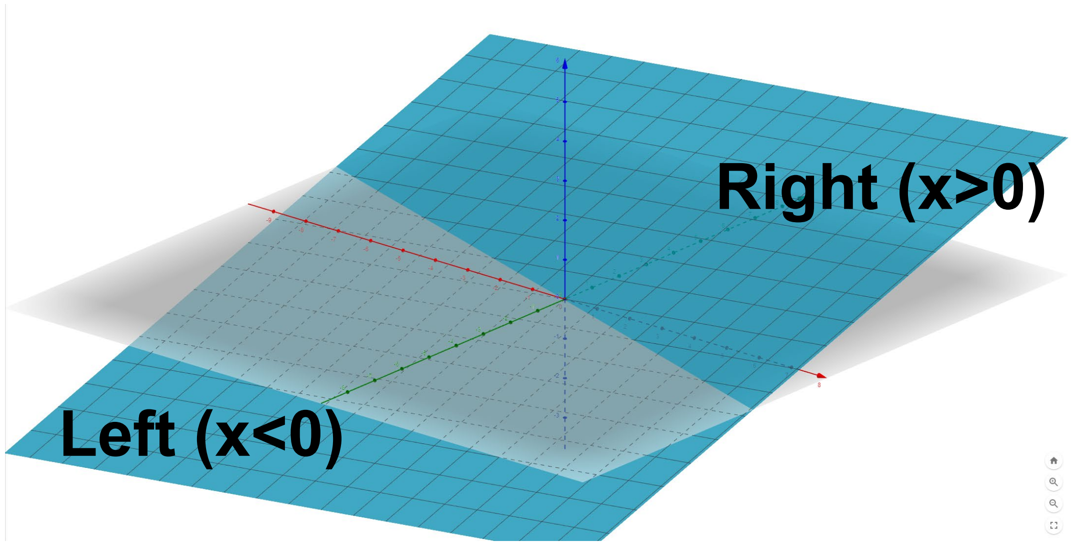
How to determine if left or right of the line?

Implicit Line – left or right?

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

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

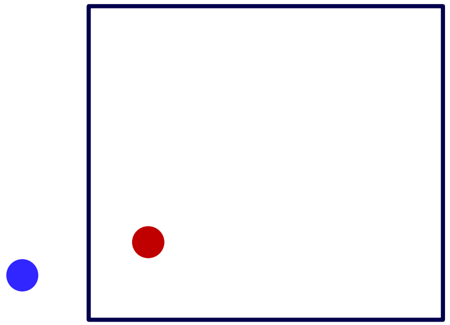




https://www.geogebra.org/3d

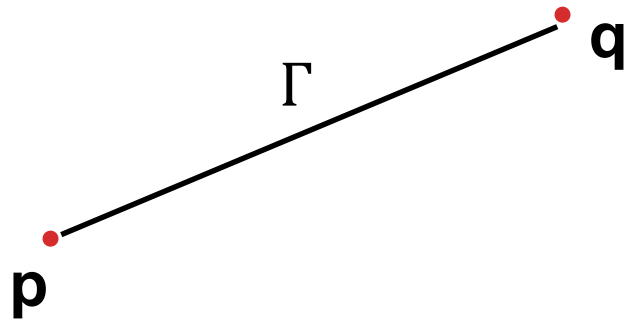
Point vs Line (-> inside test for convex poly)

- Point $\mathbf{p} = (p_x, p_y)$
- Use implicit equation to determine coincidence & side
  - *Implicit* $Ax + By + C = 0$
  - *Get there by solving 2 equations in 2 unknowns (unique with third equation:* $A^2 + B^2 = 1$*)*
  - *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*** $A \cdot p_x + B \cdot p_y + C < 0$



From parametric to implicit lines

Parametric: $\Gamma(t)$ Implicit: $Ax + By + C = 0$



$$\Gamma(\mathrm{t}) = \begin{cases} x(t) = x_0 + (x_1 - x_0)t \\ y(t) = y_0 + (y_1 - y_0)t \end{cases}$$

$$x = x_0 + (x_1 - x_0)t \qquad y = y_0 + (y_1 - y_0)t$$

Issues?

$$\leftrightarrow \frac{x - x_0}{(x_1 - x_0)} = t \qquad \leftrightarrow \frac{y - y_0}{(y_1 - y_0)} = t$$

$= C$

$$\frac{x - x_0}{(x_1 - x_0)} - \frac{y - y_0}{(y_1 - y_0)} = 0$$

$$= \frac{x}{(x_1 - x_0)} - \frac{y}{(y_1 - y_0)} - \frac{x_0}{(x_1 - x_0)} + \frac{y_0}{(y_1 - y_0)}$$

$A =$ $B =$

Without singularities?

Implicit: $Ax + By + C = 0$

$$\frac{x - x_0}{(x_1 - x_0)} - \frac{y - y_0}{(y_1 - y_0)} = 0$$

$$\frac{(x - x_0)(y_1 - y_0)}{1} - \frac{(y - y_0)(x_1 - x_0)}{1} = 0$$

$$x(y_1 - y_0) + x_0(y_1 - y_0) - y(x_1 - x_0) - y_0(x_1 - x_0) = 0$$

$$x\underbrace{(y_1 - y_0)}_{A} + y\underbrace{(x_0 - x_1)}_{B} + \underbrace{x_0(y_1 - y_0) - y_0(x_1 - x_0)}_{C} = 0$$

Self study: From explicit to implicit Line

Explicit: $y = mx + b$

$$0 = m\,x + b\ - y$$

$$\Rightarrow A = m,\ \ B = -1,\ \ C = b$$

Implicit: $Ax + By + C = 0$

Example

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

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

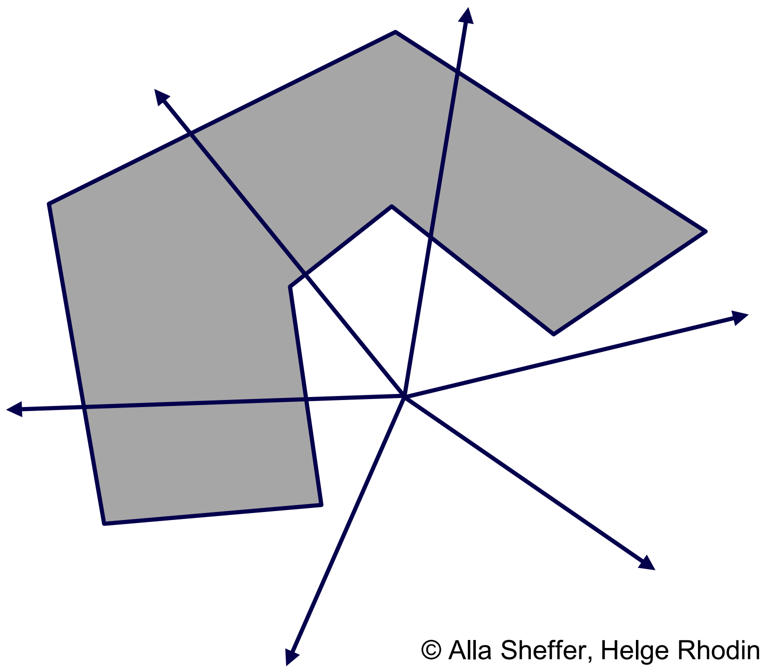
$$A = -\frac{1}{3}, \qquad B = -1, \qquad C = 0$$

Issues?

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

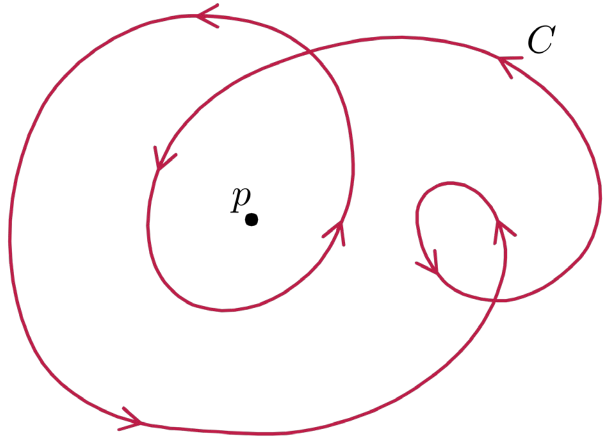
Point in polygon?

- If the winding number is non-zero

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

Winding number:

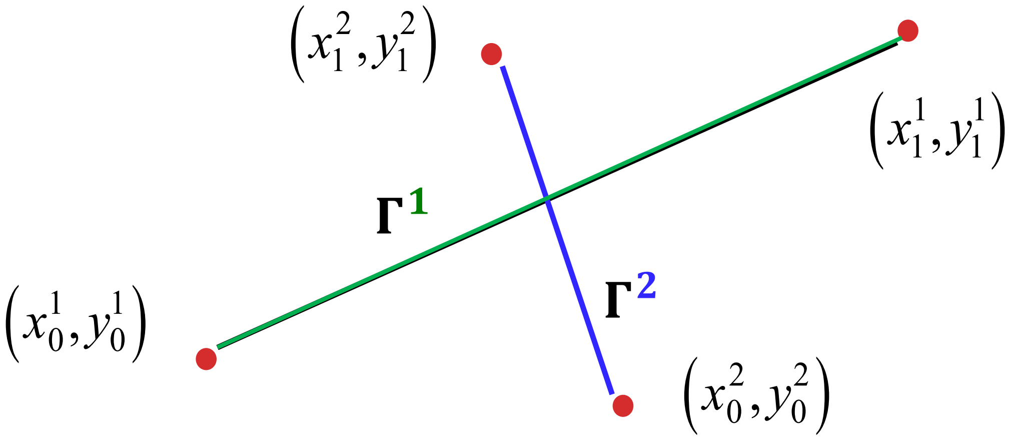
- the number of times that curve travels counterclockwise around the point
- negative if clockwise



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*)

$$\begin{aligned} x_0^1 + (x_1^1 - x_0^1)t &= x_0^2 + (x_1^2 - x_0^2)r \\ y_0^1 + (y_1^1 - y_0^1)t &= y_0^2 + (y_1^2 - y_0^2)r \end{aligned}$$

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

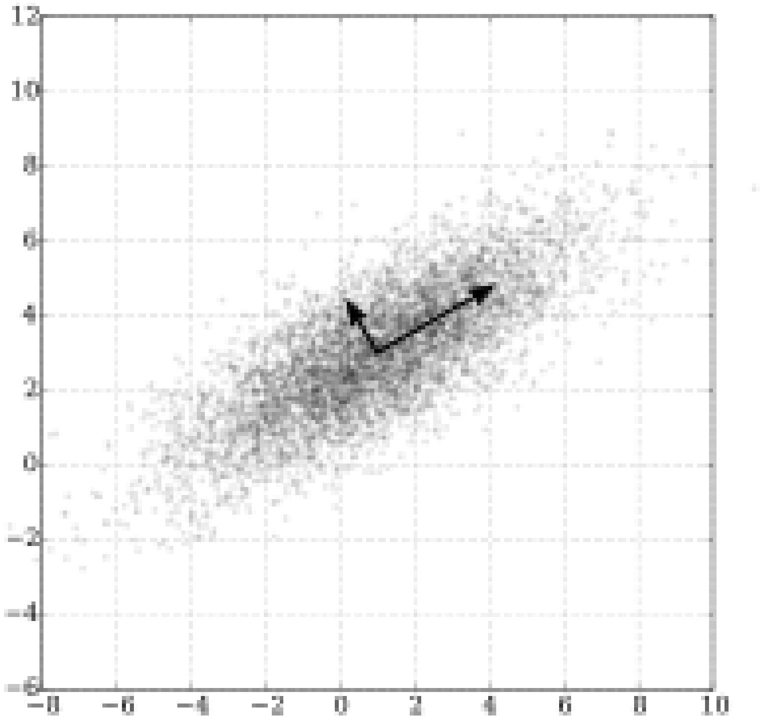
- Axis aligned bounding box (AABB)
  - *+ Trivial to compute*
  - *+ Quick to evaluate*
  - *- May be too big…*

- 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

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



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

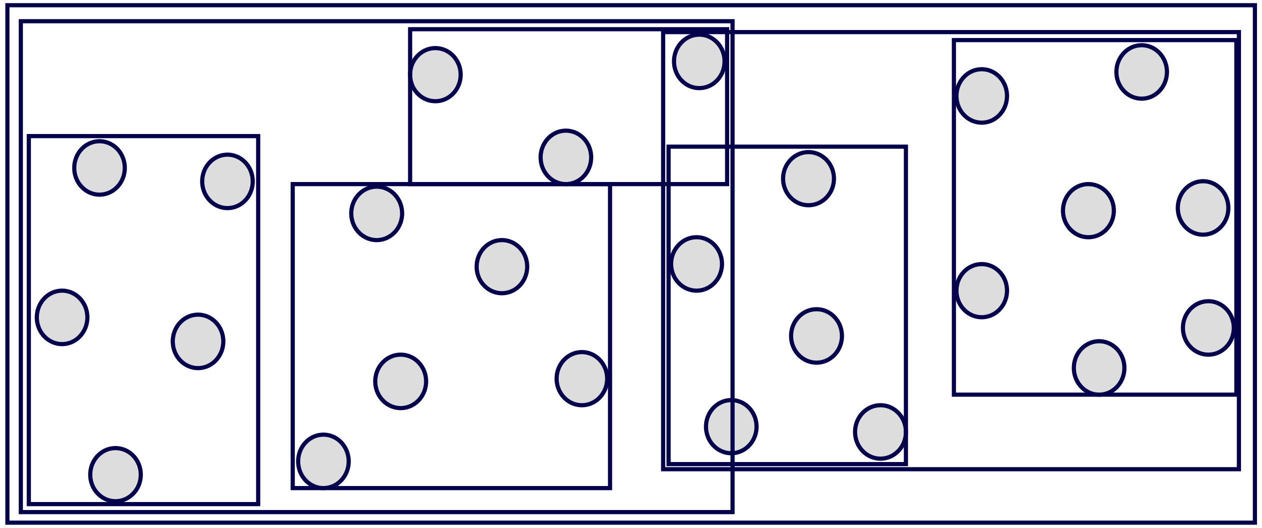
Moving objects

- 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

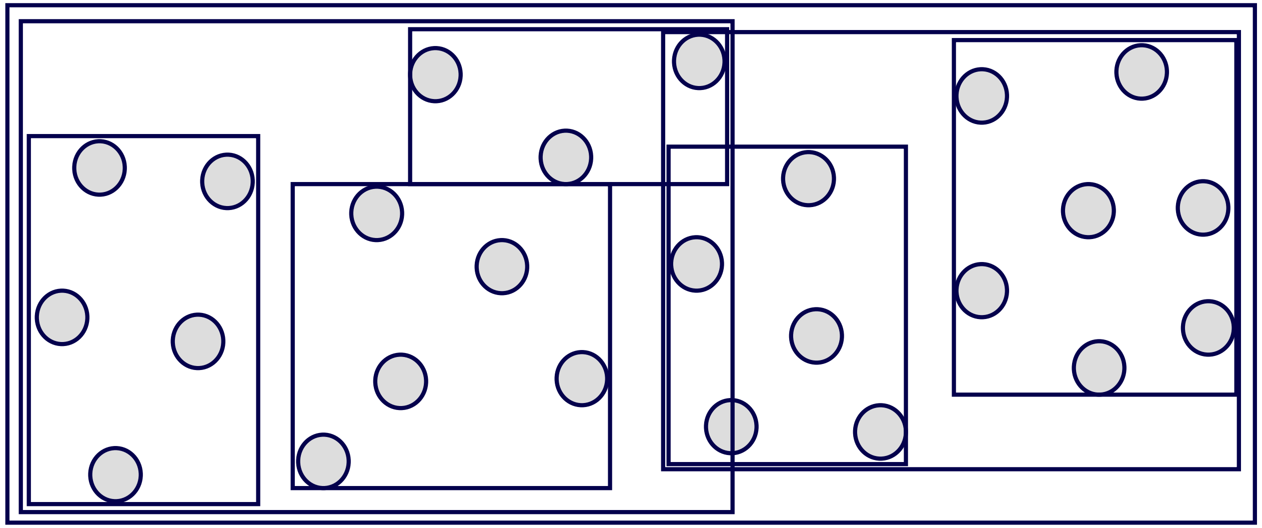


- How to group boxes?
  - *Closest*
  - *Most jointly compact (how?)*

Hierarchical Bounding Volumes

Bound Bounding Volumes:

- Use (hierarchical) bounding volumes for groups of objects



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

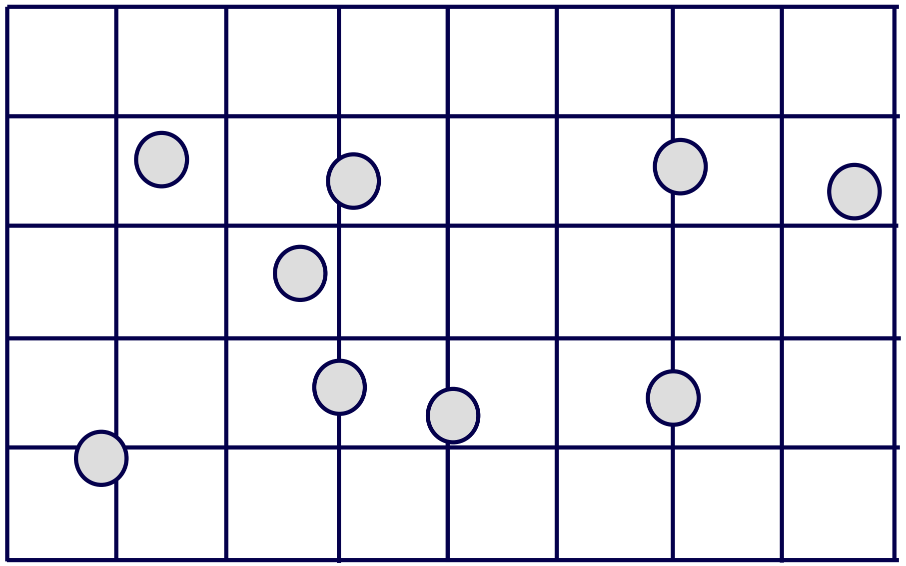
Spatial Subdivision DATA STRUCTURES

- 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)*

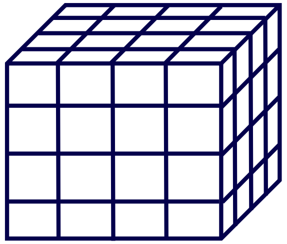
Regular Grid

Subdivide space into rectangular grid:

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



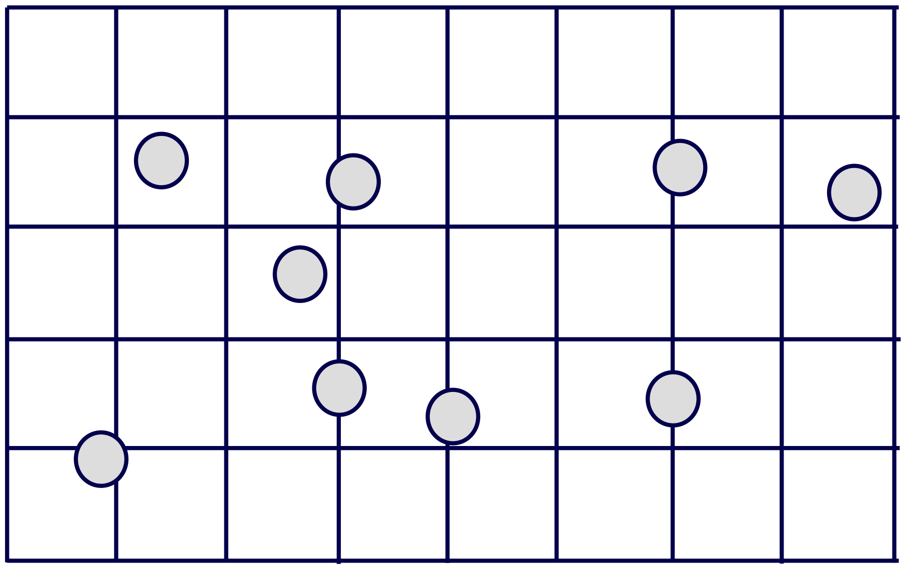
In 3D: regular grid of cubes (voxels):



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?

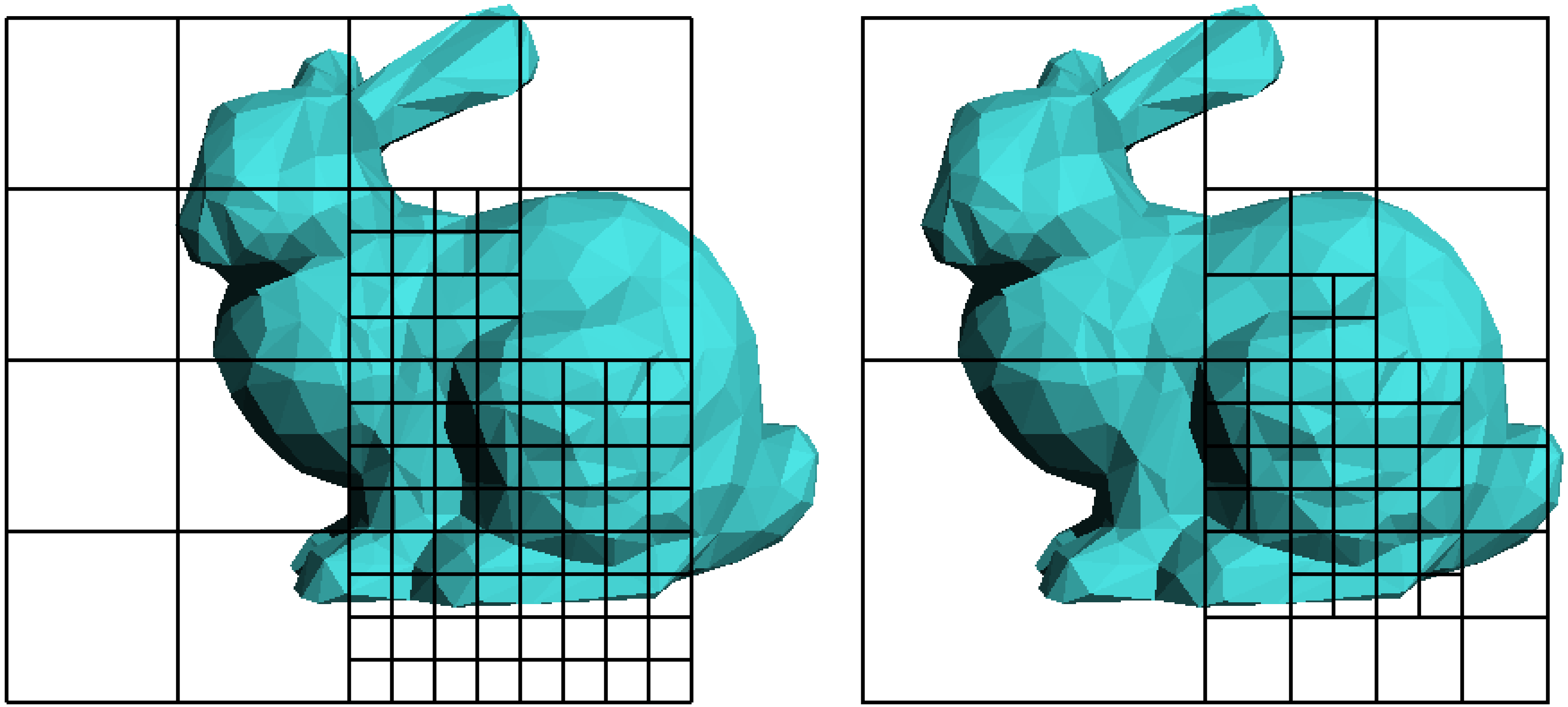
- Easy to construct
- Easy to traverse

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

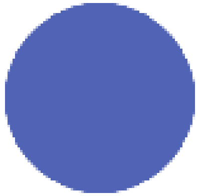
Upcoming lecture:

Physics-based simulation

Basic Particle Simulation (first try)

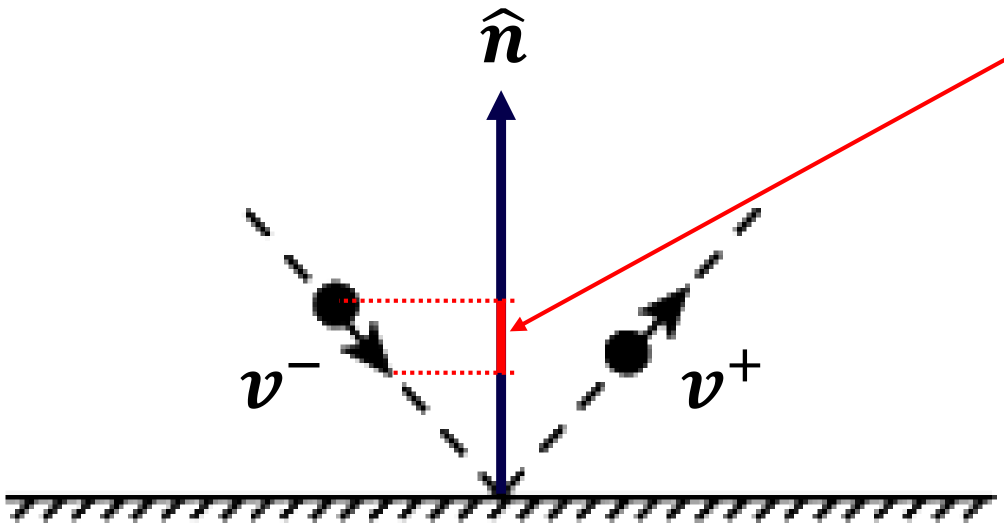
How to compute the change in velocity?

$$d_t = t_{i+1} - t_i$$
$$\vec{\boldsymbol{v}}_{i+1} = \vec{\boldsymbol{v}}_i + \color{red}{\Delta v}$$
$$\vec{\boldsymbol{p}}_{i+1} = \vec{\boldsymbol{p}}(t_i) + \vec{\boldsymbol{v}}_i d_t$$



Particle-Plane Collisions

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



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

Frictionless

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

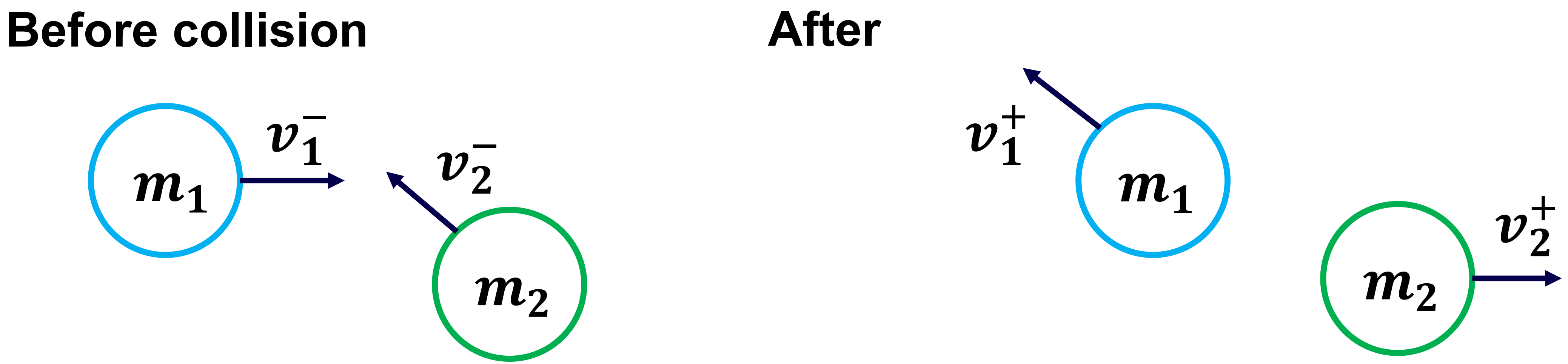
Apply change
along normal
(magnitude
times direction)

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

Loss of energy

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

Particle-Particle Collisions (spherical objects)



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