

CS529

Fundamentals of Game Development

Lecture 5b

Antoine Abi Chakra
Karim Fikani

Questions?

- Object Kinematics
- Game Loop Review
- Object Animation

Overview

- Static Collision
 - Point/Circle
 - Circle/Circle
 - Point/Rectangle
 - Rectangle/Rectangle
- Review of Vector Algebra and Geometry
- Normal Line Equation
- Animated Point to Line Classification

Static Collision - Point/Circle

- Can be used for small objects, or objects' corners
- Compare the “Point to Center” distance to the radius
- Broken down to 3 cases:
 - $|CP| < R$: The point is inside the circle -> Collision
 - $|CP| > R$: The point is outside the circle -> No collision
 - $|CP| = R$: The point lies on the perimeter of the circle. It can be considered as either colliding or non-colliding, but be **consistent**.

Static Collision - Circle/Circle (1 / 2)

- Multiple “mathematical” cases
- In games, it’s enough to compare the centers’ distance to the sum of the radius
 - Point centerA, centerB
float radiusA, radiusB

RadiusSum = radiusA + radiusB

CentersDistance = Length(centerA, centerB)

if(CentersDistance <= RadiusSum)

Collision

else

No Collision

Static Collision - Circle/Circle (2/2)

- “Length” can be avoided
 - Square root operations are expensive
- Solution: Square both sides
 - Point centerA, centerB
float radiusA, radiusB

$\text{RadiusSumSq} = (\text{radiusA} + \text{radiusB})^2$

$\text{CentersDistanceSq} = \text{LengthSquared}(\text{centerA}, \text{centerB})$

if($\text{CentersDistanceSq} \leq \text{RadiusSumSq}$)

Collision

else

No Collision

Static Collision - Point/Rectangle

- Again, enough for small objects
- Rectangle is defined as: Top, Bottom, Left & Right
- Algorithm:
 - Point P;
float left, right, top, bottom;

if($P.X < \text{left}$) then no collision

if($P.X > \text{right}$) then no collision

if($P.Y < \text{bottom}$) then no collision

if($P.Y > \text{top}$) then no collision

Collision!

Static Collision - Rectangle/Rectangle (1/2)

- Used in bounding rectangles

- Algorithm:

- float leftA, leftB
float rightA, rightB
float topA, topB
float bottomA, bottomB

if(leftA > rightB) then no collision

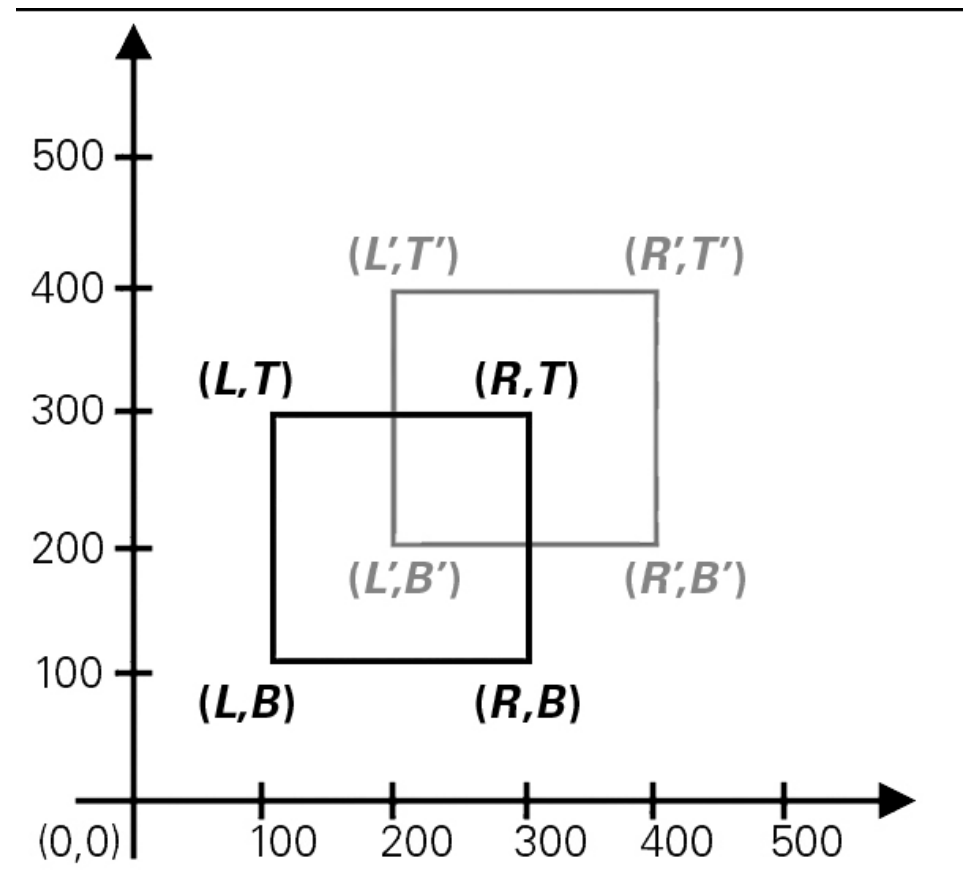
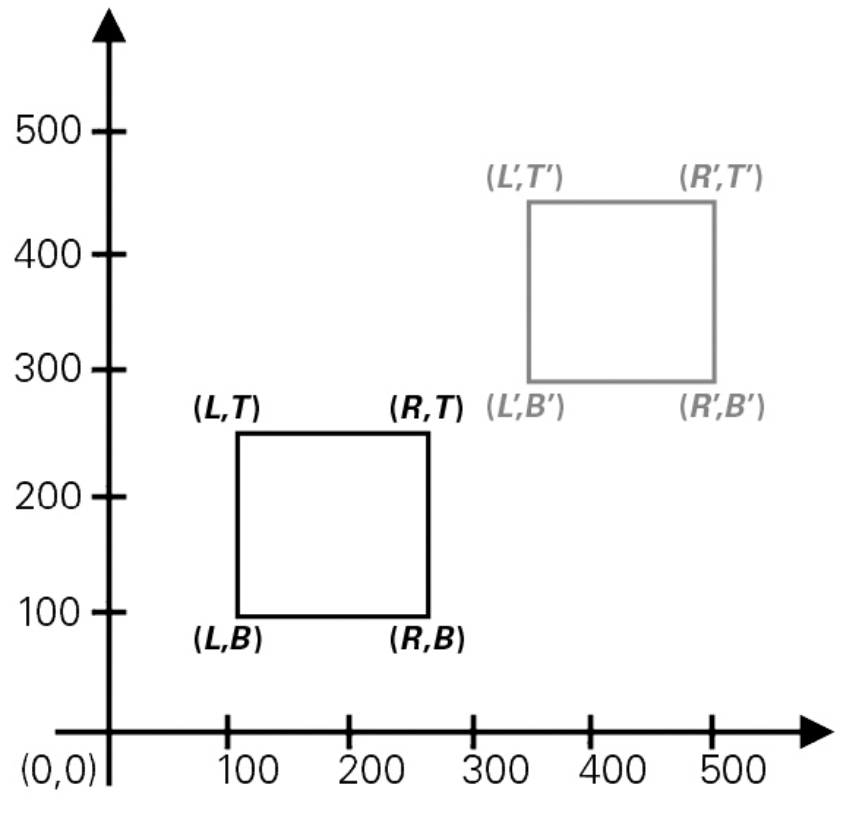
if(leftB > rightA) then no collision

if(topA < bottomB) then no collision

if(topB < bottomA) then no collision

Collision!

Static Collision - Rectangle/Rectangle (2/2)

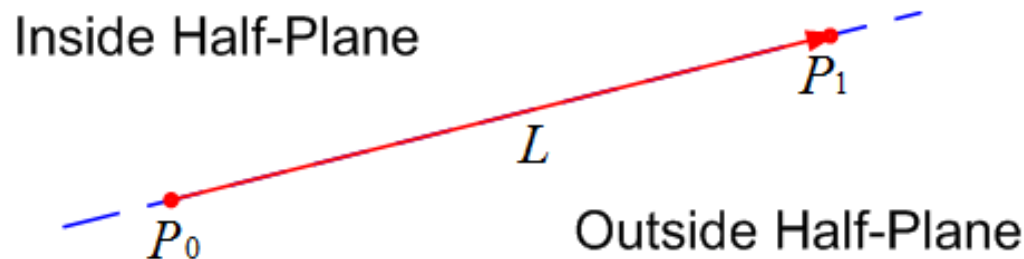


Overview

- Static Collision
 - Point/Circle
 - Circle/Circle
 - Point/Rectangle
 - Rectangle/Rectangle
- Review of Vector Algebra and Geometry
- Normal Line Equation
- Animated Point to Line Classification

Line Segment & Half Planes

- Consider directed line segment L from position P_0 to position P_1
- Infinite extension of L divides XY -plane into two half-planes
 - Half-plane on L 's right-hand side is by (our) convention referred to as *outside* (or, *positive*) half-plane
 - Half-plane on L 's left-hand side is *inside* (or, *negative*) half-plane

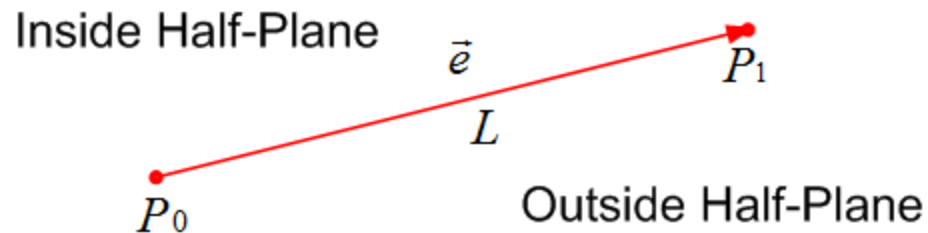


Line Segment: Edge Vector

- Compute L's edge vector

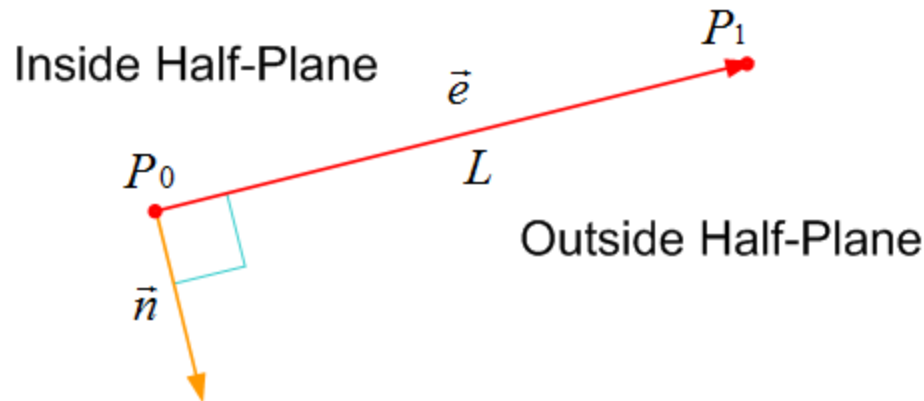
$$\vec{e} = P_1 - P_0 = (e_x, e_y)$$

$$\Rightarrow (e_x, e_y) = (x_1 - x_0, y_1 - y_0)$$



Outward Normal of Line Segment

- What is *outward normal* to line segment L ?
 - Vector n is orthogonal to L 's edge vector e such that n is oriented from L 's inside to outside half-plane



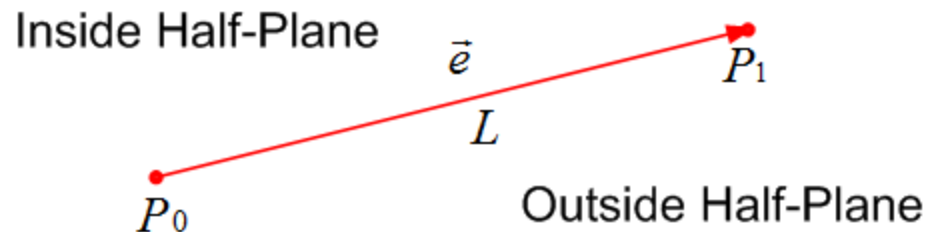
Computing Outward Normal (1 / 3)

- How is the outward normal n to line segment L computed?
 - Rotate edge vector e by -90° about Z-axis
 - That is, edge vector e is rotated about Z-axis in *clockwise direction* by 90°

Computing Outward Normal (2/3)

- First, compute directed line segment L 's edge vector e :

$$\vec{e} = (e_x, e_y) = P_1 - P_0 = (x_1 - x_0, y_1 - y_0)$$

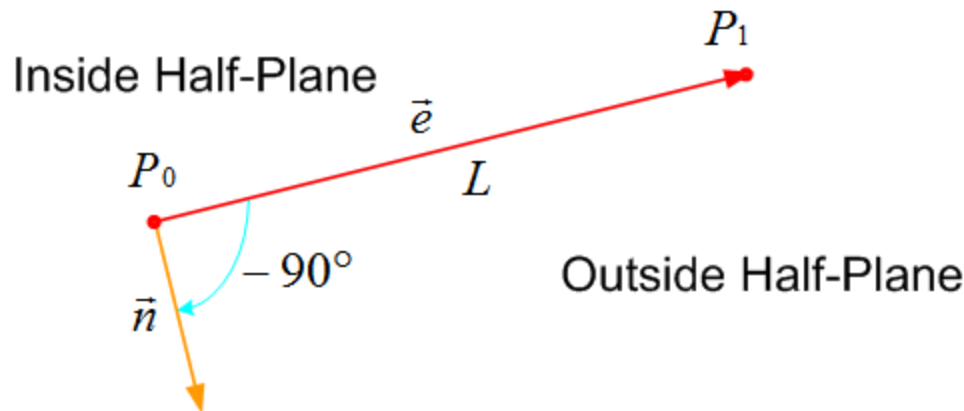


Computing Outward Normal (3/3)

- To compute outward normal n , rotate edge vector e about Z-axis by -90°

$$\begin{bmatrix} n_x \\ n_y \end{bmatrix} = \begin{bmatrix} \cos(-90^\circ) & -\sin(-90^\circ) \\ \sin(-90^\circ) & \cos(-90^\circ) \end{bmatrix} \begin{bmatrix} e_x \\ e_y \end{bmatrix}$$

$$\Rightarrow \vec{n} = (n_x, n_y) = (e_y, -e_x)$$



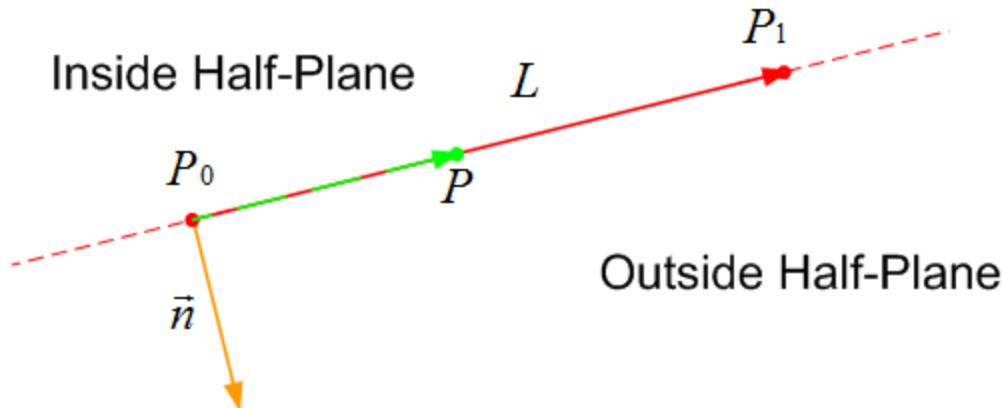
Overview

- Static Collision
 - Point/Circle
 - Circle/Circle
 - Point/Rectangle
 - Rectangle/Rectangle
- Review of Vector Algebra and Geometry
- Normal Line Equation
- Animated Point to Line Classification

Point-Normal Line Equation (1/2)

- Let $P(x, y)$ be an arbitrary point on L 's infinite extension
- Point-normal equation of L is:

$$\vec{n} \bullet (P - P_0) = 0 \Rightarrow \vec{n} \bullet P - \vec{n} \bullet P_0 = 0$$

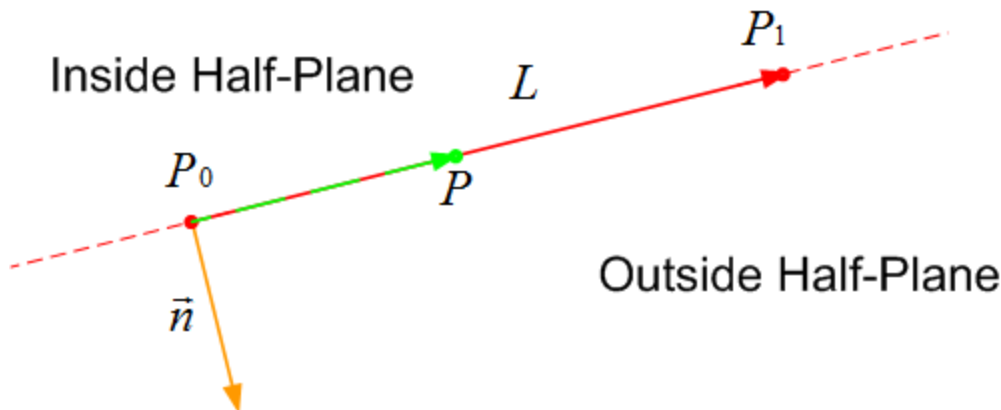


Point-Normal Line Equation (2/2)

- Better to use normalized outward normal
- Using normalized outward normal, point-normal equation of line segment L from point P_0 to point P_1 is written as:

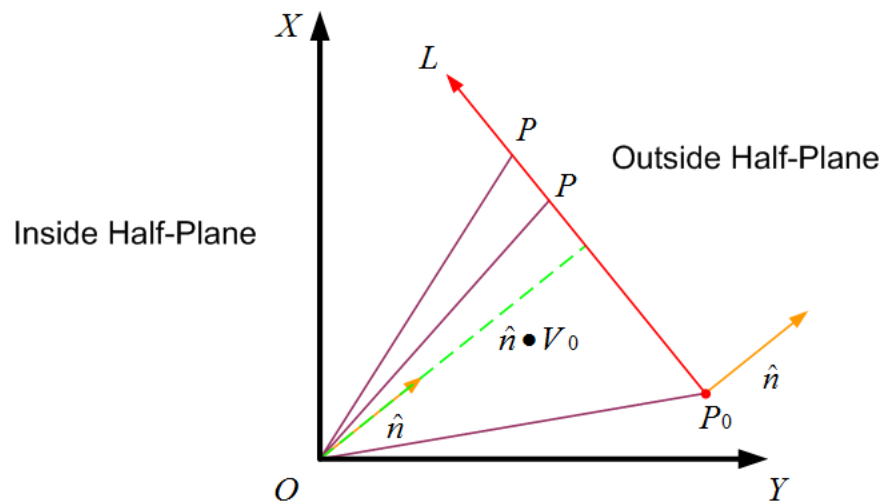
$$L : \hat{n} \bullet P - \hat{n} \bullet P_0 = 0$$

$$\hat{n} = \frac{\vec{n}}{\|\vec{n}\|}$$



Geometrical Interpretation

- Projections of position vectors of each of the infinite points P on L onto normalized outward normal \hat{n} will result in same value
 - Value is *orthogonal* (or, *shortest*) distance from coordinate system origin to L

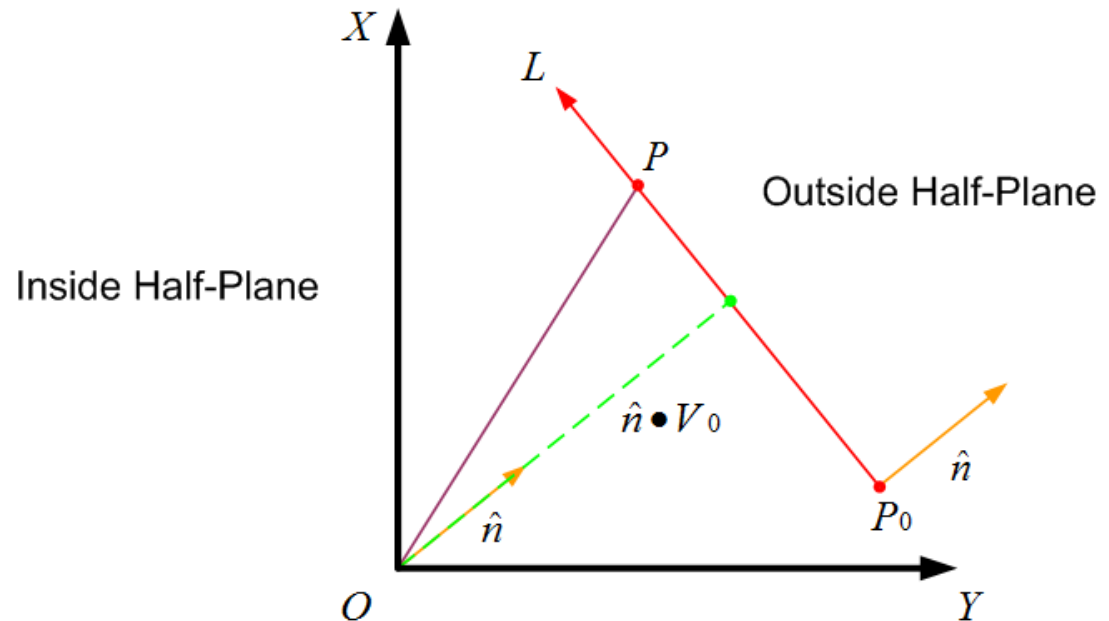


$$L : \hat{n} \bullet P - \hat{n} \bullet P_0 = 0$$

Point-Line Classification (1/3)

$$\hat{n} \bullet P - \hat{n} \bullet P_0 = 0 \Leftrightarrow$$

P is co-linear with L

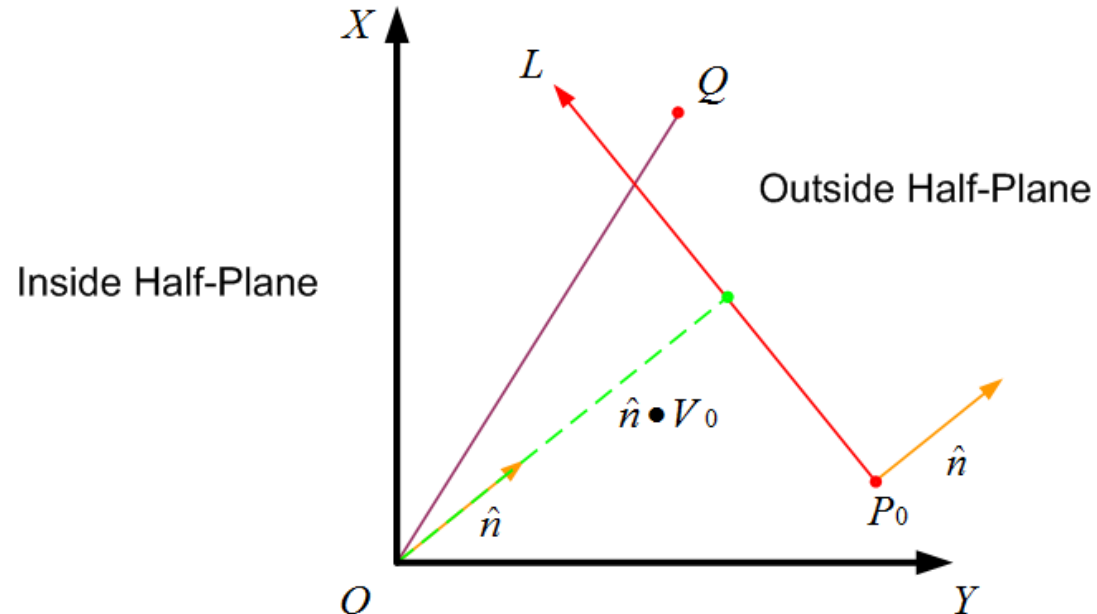


- Distance between origin and arbitrary point P (measured along normalized normal to L) is equal to shortest distance from origin to line segment L
- This implies that P must lie on infinite extension of L

Point-Line Classification (2/3)

$$\hat{n} \bullet Q - \hat{n} \bullet P_0 > 0 \Leftrightarrow$$

Q in outside half-plane of L



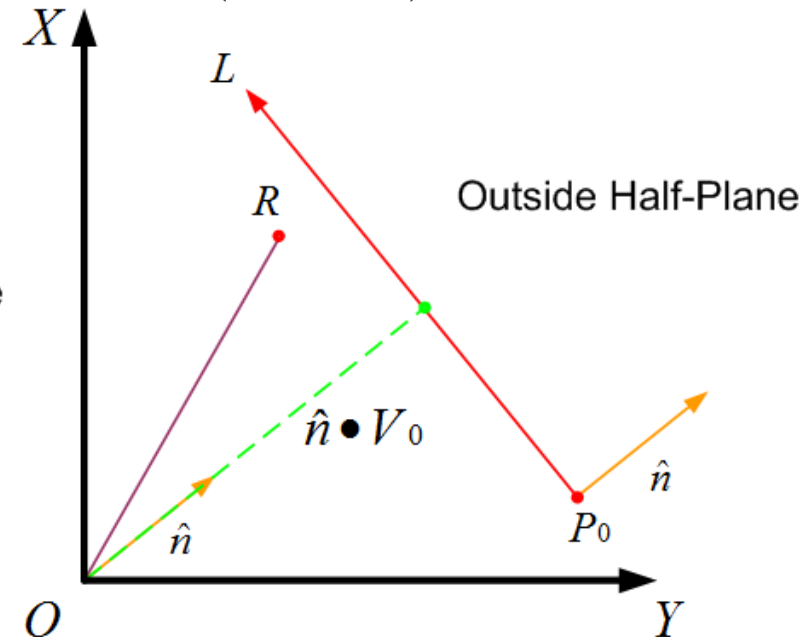
- Distance between origin and arbitrary point Q (measured along normalized normal to L) is greater than shortest distance from origin to line segment L
- This implies that Q must lie in outside half-plane of L

Point-Line Classification (3/3)

$$\hat{n} \bullet R - \hat{n} \bullet P_0 < 0 \Leftrightarrow$$

R in the inside half-plane of L

Inside Half-Plane



- Distance between origin and arbitrary point R (measured along normalized normal to L) is smaller than shortest distance from origin to line segment L
- This implies that R must lie in inside half-plane of L

Overview

- Static Collision
 - Point/Circle
 - Circle/Circle
 - Point/Rectangle
 - Rectangle/Rectangle
- Review of Vector Algebra and Geometry
- Normal Line Equation
- **Animated Point to Line Classification**

Collision Experiment

- Given:
 - Static wall of finite length and infinitesimal thickness
 - Animated pinball with an infinitesimal radius
- Problem:
 - Ensure animated pinball correctly collides and bounces off wall

Boundary Condition of Point

- Evaluation of an arbitrary point in line segment's point-normal equation is called *boundary condition of arbitrary point* with respect to the line segment
- Boundary condition of arbitrary point P with respect to line segment L is:

$$BC_L^P = \hat{n} \cdot (P - P_0)$$

- Boundary condition BC_L^P evaluates to three results:
 - Positive \Leftrightarrow Point P in outside half-plane of line segment L
 - Negative \Leftrightarrow Point P in inside half-plane of line segment L
 - Zero \Leftrightarrow Point P on line segment L

Geometrical and Mathematical Model of Wall

- Geometrical model of wall
 - Directed line segment from position P_o to position P_1
- Mathematical model of wall
 - Infinite extension of directed line segment from position P_o to position P_1
 - $L: n \bullet P - n \bullet P_o = 0$
 - n is the normalized outward normal of directed line segment from position P_o to position P_1
 - P is any arbitrary point on infinite extension of line segment
 - $n \bullet P_o$ is the orthogonal distance from origin to line segment

Modeling Pinball Animation

- Pinball modeled as an infinitesimal point
- Located at points B_s and B_e at times t_s (frame start time) and t_e (frame end time), respectively within the current frame
- Moving with speed k units along direction given by normalized vector v
- Pinball location during current frame is modeled as

$$B(t) = B_s + k\hat{v}t, t \in [0,1]$$

- Velocity per **frame** $\vec{v} = \overrightarrow{B_s B_e}$

$$\Rightarrow B(t) = B_s + \vec{v}t, t \in [0,1]$$

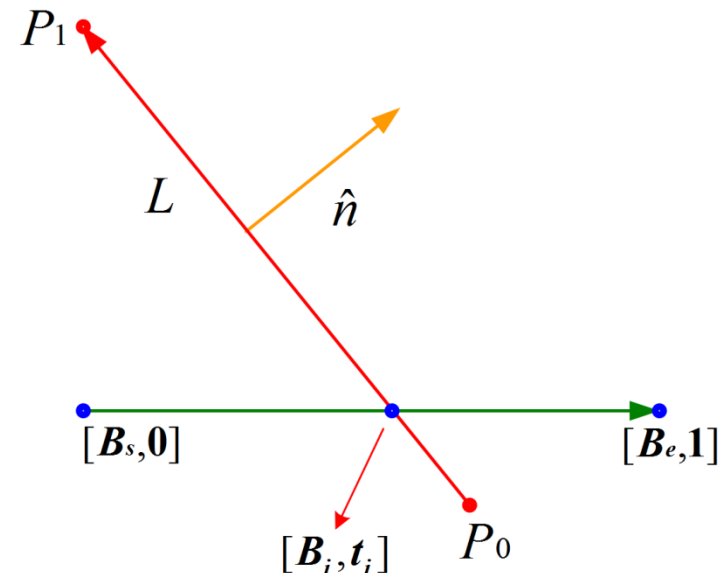
Intersection Between Wall and Animated Ball

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{\mathbf{v}}t$

Wall modeled as $L: \hat{\mathbf{n}} \bullet \mathbf{P} - \hat{\mathbf{n}} \bullet \mathbf{P}_0 = 0$

$$t_i = \frac{\hat{\mathbf{n}} \bullet \mathbf{P}_0 - \hat{\mathbf{n}} \bullet \mathbf{B}_s}{\hat{\mathbf{n}} \bullet \vec{\mathbf{v}}} \text{ and } t_i \in [0,1]$$

$$\mathbf{B}_i = \mathbf{B}_s + \vec{\mathbf{v}} \left(\frac{\hat{\mathbf{n}} \bullet \mathbf{P}_0 - \hat{\mathbf{n}} \bullet \mathbf{B}_s}{\hat{\mathbf{n}} \bullet \vec{\mathbf{v}}} \right)$$

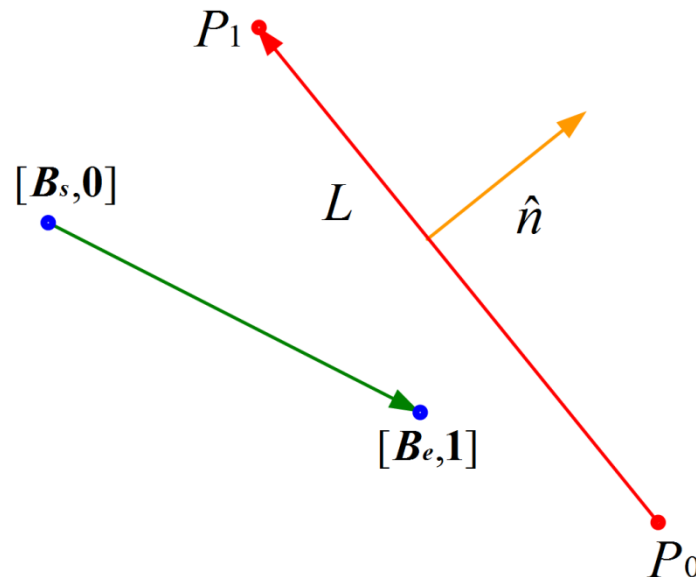


Test for Non-Collision (1/5)

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{v}t$

Wall modeled as $L: \hat{n} \bullet \mathbf{P} - \hat{n} \bullet \mathbf{P}_0 = 0$

$$(\hat{n} \bullet \mathbf{B}_s < \hat{n} \bullet \mathbf{P}_0) \& \& (\hat{n} \bullet \mathbf{B}_e < \hat{n} \bullet \mathbf{P}_0)$$

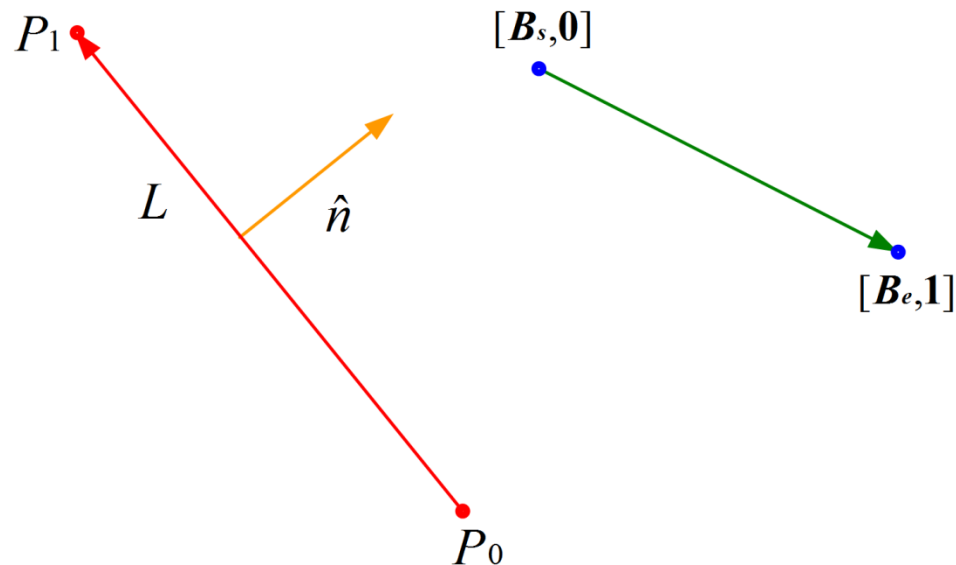


Test for Non-Collision (2/5)

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{v}t$

Wall modeled as $L: \hat{n} \bullet \mathbf{P} - \hat{n} \bullet \mathbf{P}_0 = 0$

$(\hat{n} \bullet \mathbf{B}_s > \hat{n} \bullet \mathbf{P}_0) \& \& (\hat{n} \bullet \mathbf{B}_e > \hat{n} \bullet \mathbf{P}_0)$

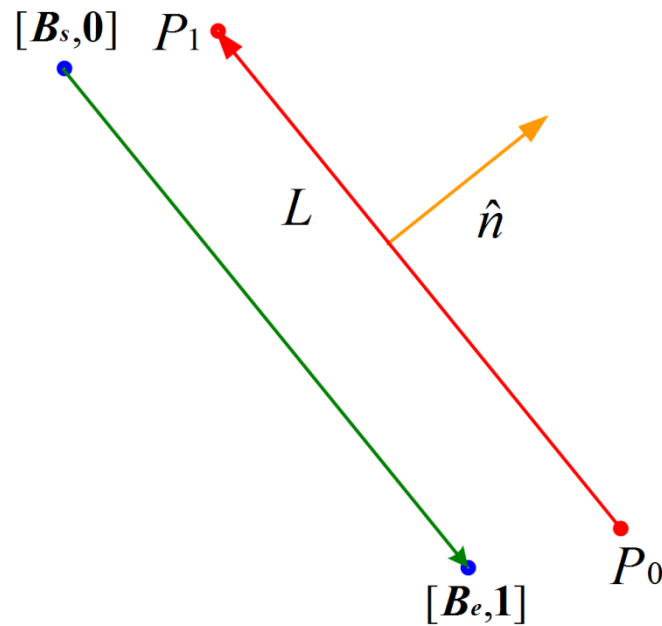


Test for Non-Collision (3/5)

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{v}t$

Wall modeled as $L: \hat{n} \bullet \mathbf{P} - \hat{n} \bullet \mathbf{P}_0 = 0$

$$\hat{n} \bullet \vec{v} = 0$$

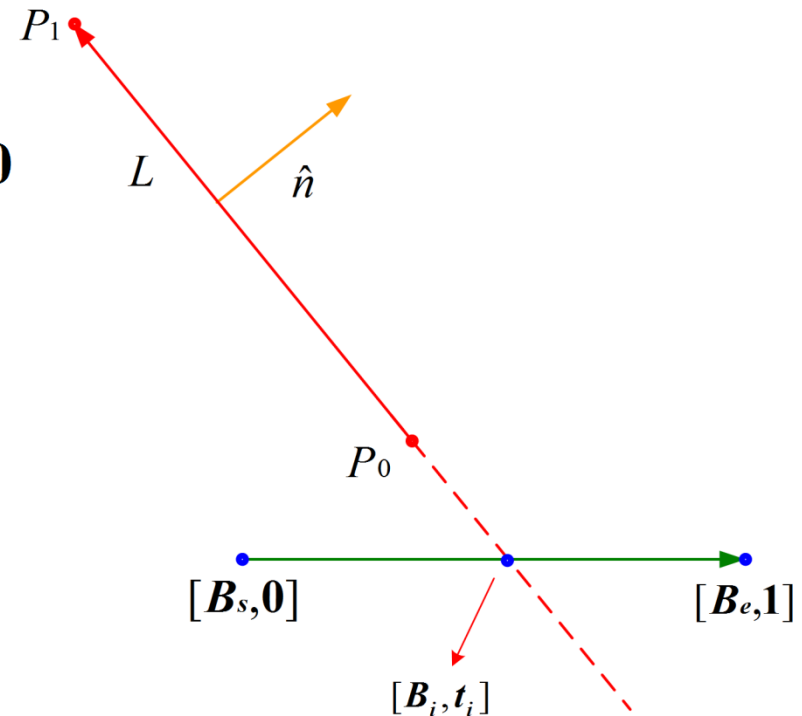


Test for Non-Collision (4/5)

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{v}t$

Wall modeled as $L: \hat{n} \bullet \mathbf{P} - \hat{n} \bullet \mathbf{P}_0 = 0$

$$(\mathbf{B}_i - \mathbf{P}_0) \bullet (\mathbf{P}_1 - \mathbf{P}_0) < 0$$



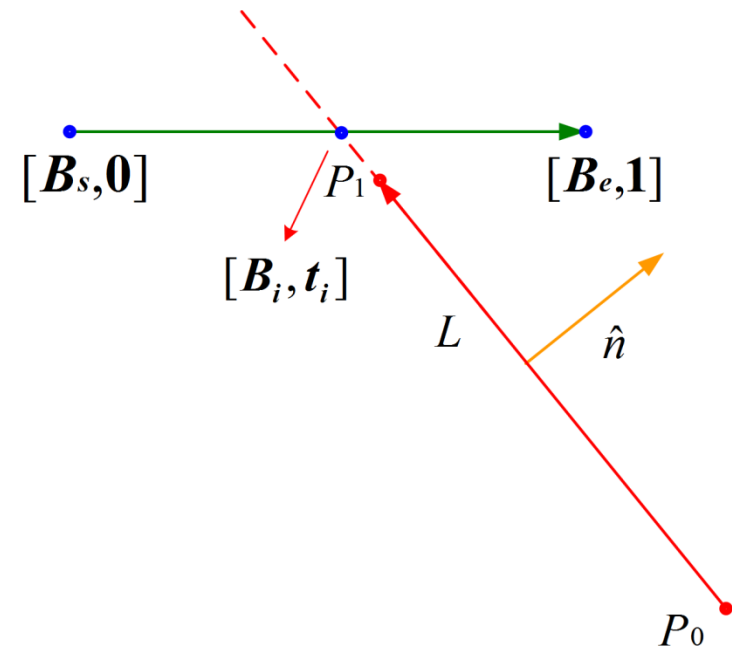
- Ball collides with infinite extension of wall ... not finite wall!

Test for Non-Collision (5/5)

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{v}t$

Wall modeled as $L: \hat{n} \bullet \mathbf{P} - \hat{n} \bullet \mathbf{P}_0 = 0$

$$(\mathbf{B}_i - \mathbf{P}_1) \bullet (\mathbf{P}_0 - \mathbf{P}_1) < 0$$



- Ball collides with infinite extension of wall ... not finite wall!

Collision of Animated Ball with Wall

Ball modeled as : $\mathbf{B}(t) = \mathbf{B}_s + \vec{\mathbf{v}}t$

Wall modeled as $L: \hat{\mathbf{n}} \bullet \mathbf{P} - \hat{\mathbf{n}} \bullet \mathbf{P}_0 = 0$

$$t_i = \frac{\hat{\mathbf{n}} \bullet \mathbf{P}_0 - \hat{\mathbf{n}} \bullet \mathbf{B}_s}{\hat{\mathbf{n}} \bullet \vec{\mathbf{v}}} \text{ and } t_i \in [0,1]$$

$$\mathbf{B}_i = \mathbf{B}_s + \vec{\mathbf{v}} \left(\frac{\hat{\mathbf{n}} \bullet \mathbf{P}_0 - \hat{\mathbf{n}} \bullet \mathbf{B}_s}{\hat{\mathbf{n}} \bullet \vec{\mathbf{v}}} \right)$$

