CS354R

DR SARAH ABRAHAM

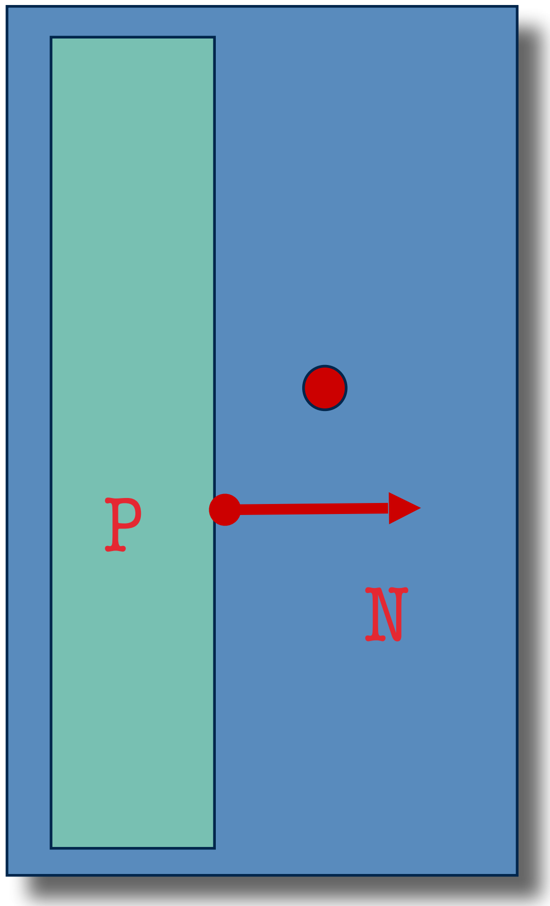
COLLISION DETECTION

COLLISION DETECTION AND RESPONSE

- Physics tells us how forces act on objects
- Collision *detection* tells us when two or more objects interact
- Collision *response* tells us what to do to resolve these interactions

WHEN DO THINGS COLLIDE?

- For now, let's just consider a point-plane collision

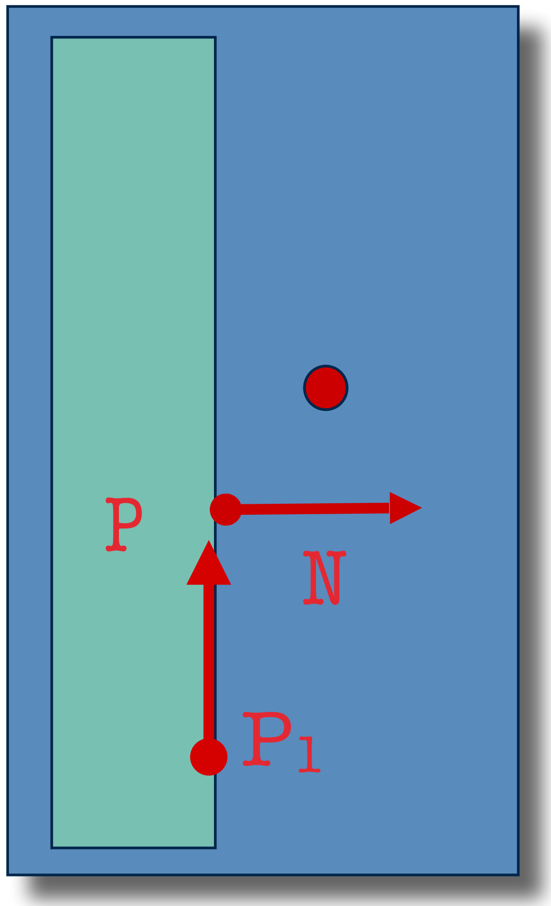


A plane is fully specified by any point **P** on the plane and its normal **N** (the cross product of two of its vectors).

THE PLANE EQUATION

The plane equation:

$$(N \cdot P) + D = 0$$



$$N \cdot (P - P_1) = 0$$

$$(N_x, N_y, N_z) \cdot (P_x - P_{1x}, P_y - P_{1y}, P_z - P_{1z}) = 0$$

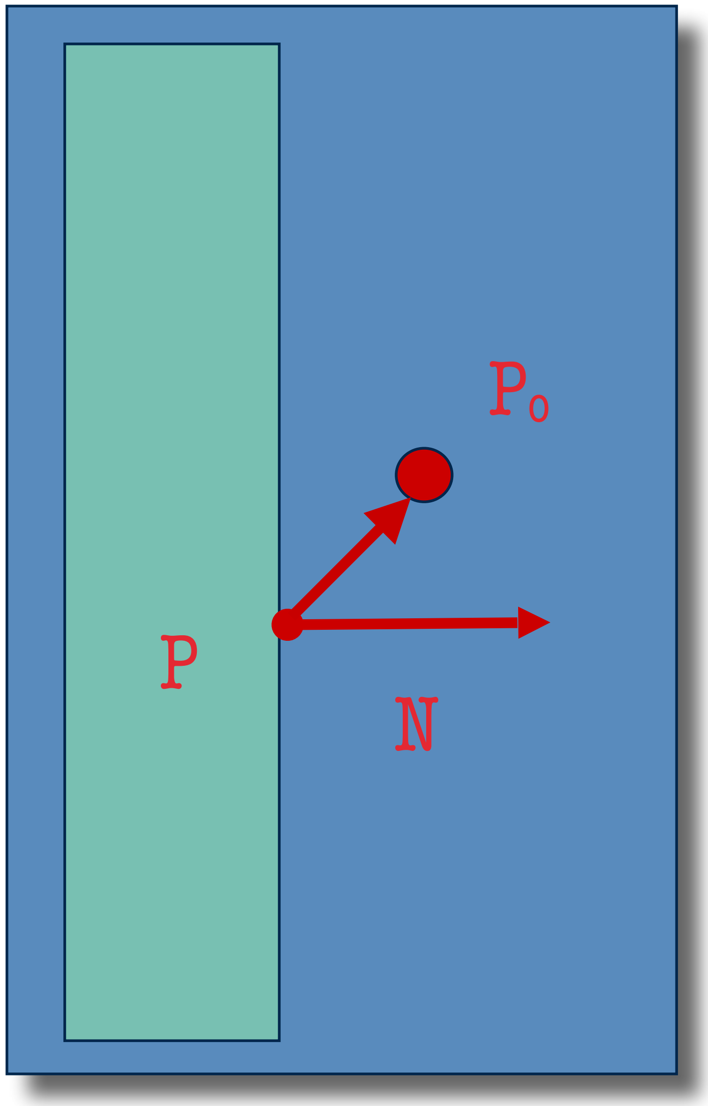
$$N_x(P_x - P_{1x}) + N_y(P_y - P_{1y}) + N_z(P_z - P_{1z}) = 0$$

the constant D

$$N_xP_x + N_yP_y + N_zP_z - (N_xP_{1x} + N_yP_{1y} + N_zP_{1z}) = 0$$

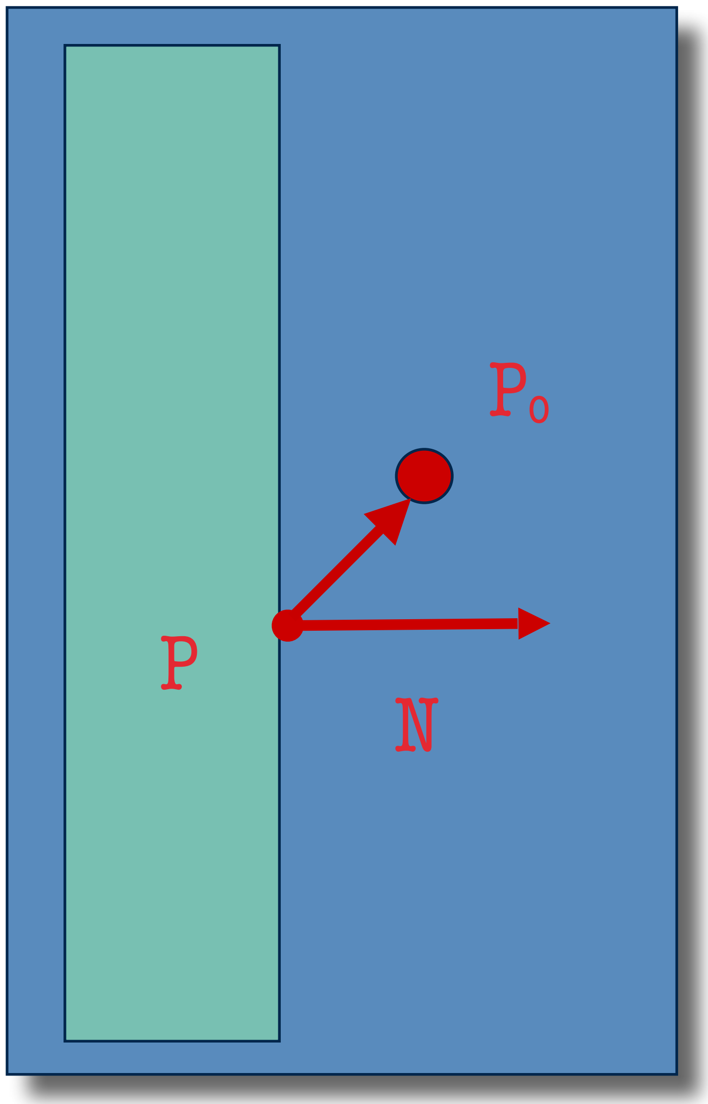
COLLISION DETECTION

How do you decide when you've crossed a plane?



POINT DISTANCE FROM PLANE

- Project vector (P_o - P) onto N to get length



$$d = (P_o - P) \cdot \frac{N}{\|N\|}$$

constant D

$$d = \frac{N_x P_{ox} + N_y P_{oy} + N_z P_{oz} - (N_x P_x + N_y P_y + N_z P_z)}{\sqrt{N_x^2 + N_y^2 + N_z^2}}$$

$$d = \frac{N \cdot P_o + D}{\sqrt{N_x^2 + N_y^2 + N_z^2}}$$

- Distance d is signed relative to N's direction

WHAT IS THIS IGNORING?

- …everything with volume…

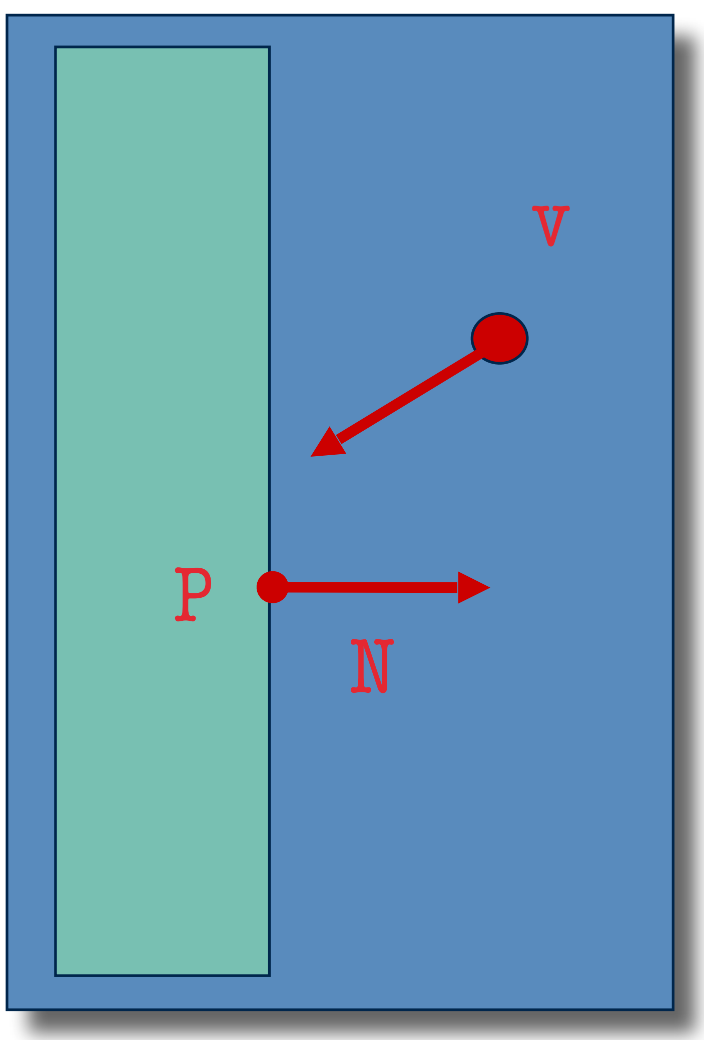
- But we'll come back to rigid bodies in a bit…

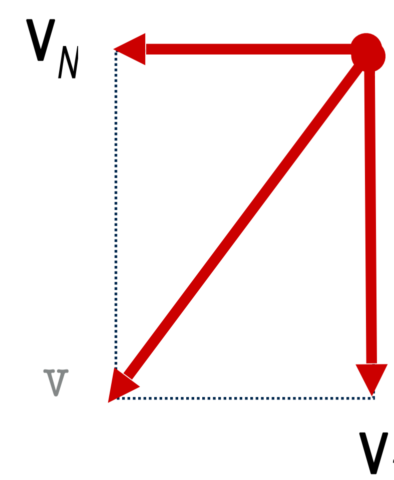
WHAT NOW?

- We can detect when a point has passed across a plane boundary
  - How do we respond?

COLLISION RESPONSE

To compute the collision response, we need to consider the normal and tangential components of a particle's velocity in the direction of N



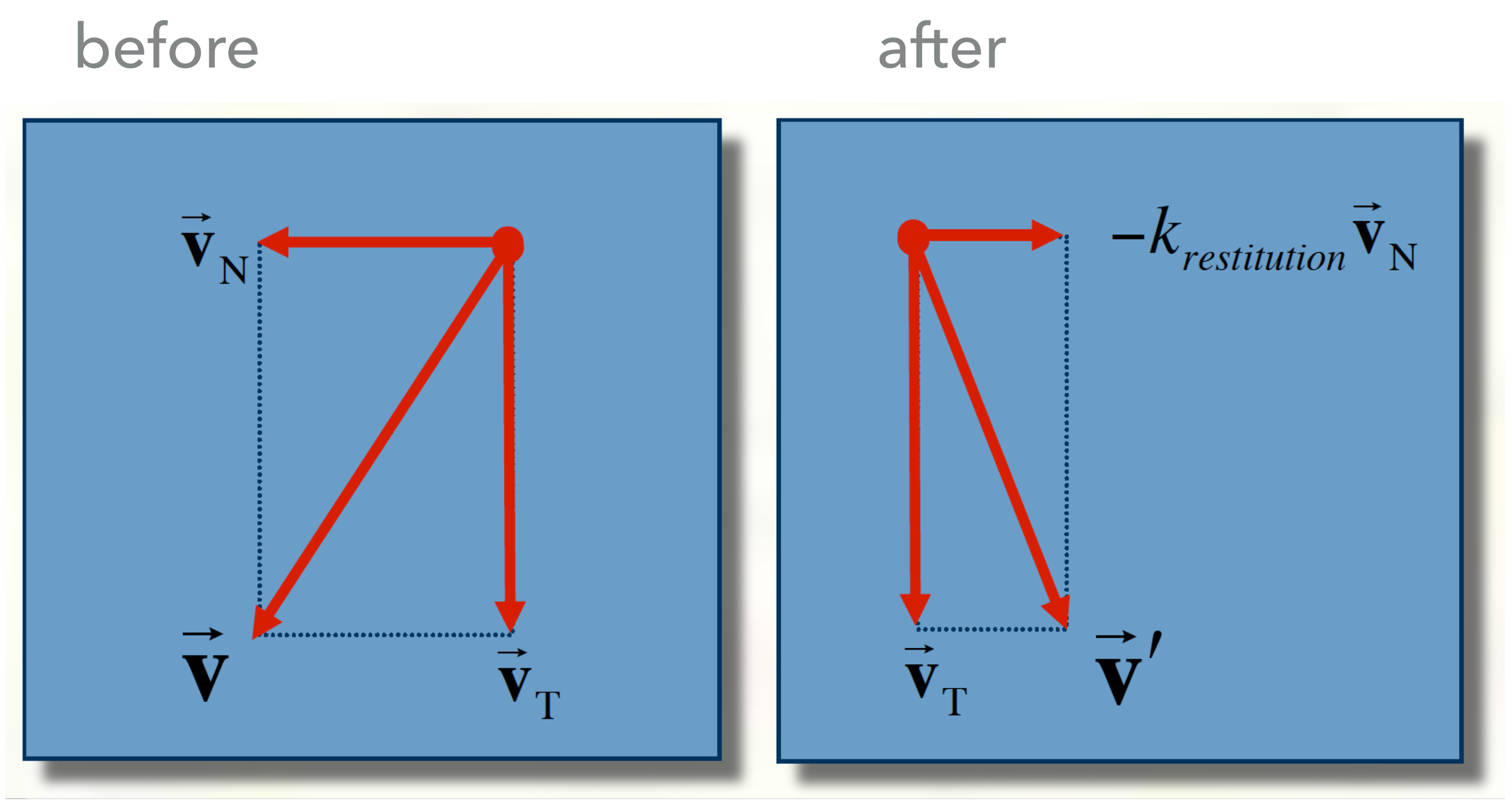


$$V_N = (N \cdot V)N$$

$$V_T = V - V_N$$

COLLISION RESPONSE

- Restitution represents amount of kinetic energy retained after collision between two material types
- Can also respond by modeling impulse and friction forces
  - More accurate but also more computation



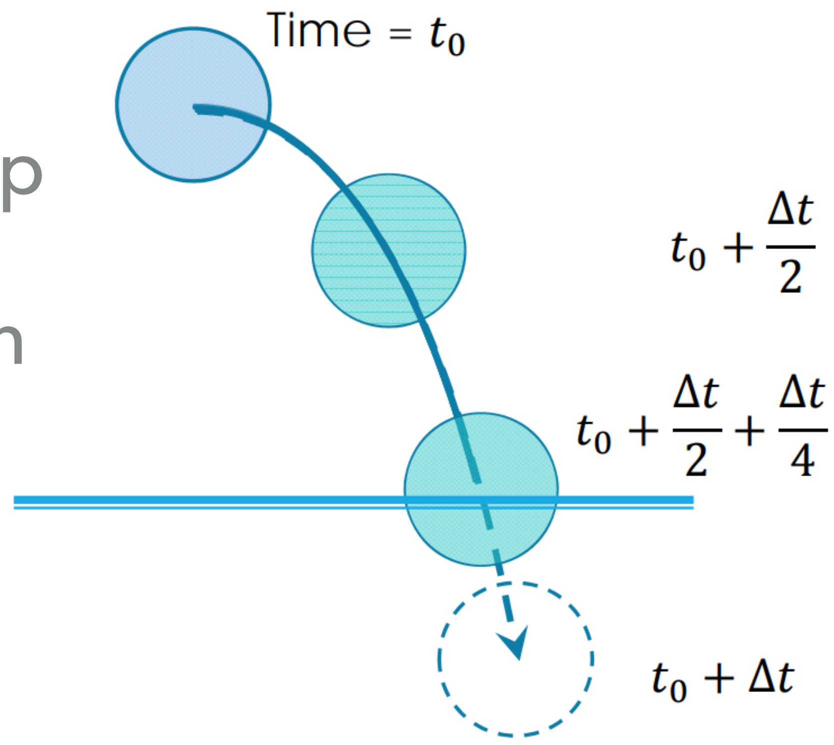
$$\vec{V}' = \vec{V}_T - k_{restitution}\vec{V}_N$$

WHEN TO DETECT?

- After Contact (a posteriori)
  - Run simulation
  - “Roll back” if intersection occurs
- Before Contact (a priori)
  - Predict time of collision
  - Update position accordingly
- Resting Contact
  - Two objects are in contact with each other
  - A surprisingly difficult special case

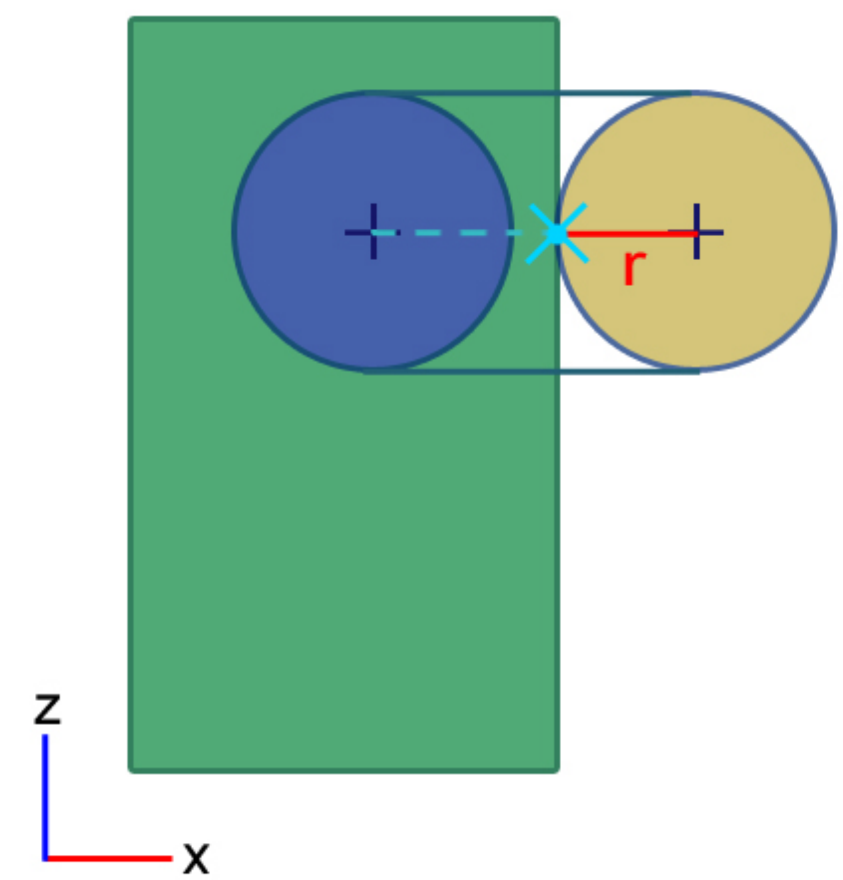
ROLLBACK RESPONSE

- If an object has a positive distance d_i to plane at start of timestep i, and a negative distance d_{i+1} to plane at start of timestep i+1, a collision has occurred during timestep i
  - Must determine *when* in timestep i collision occurred
- Backtracking waits for collision to be detected then steps back through timestep
  - Bisect timestep to approximate collision time
  - Must run backtracking for all objects to avoid incorrect response



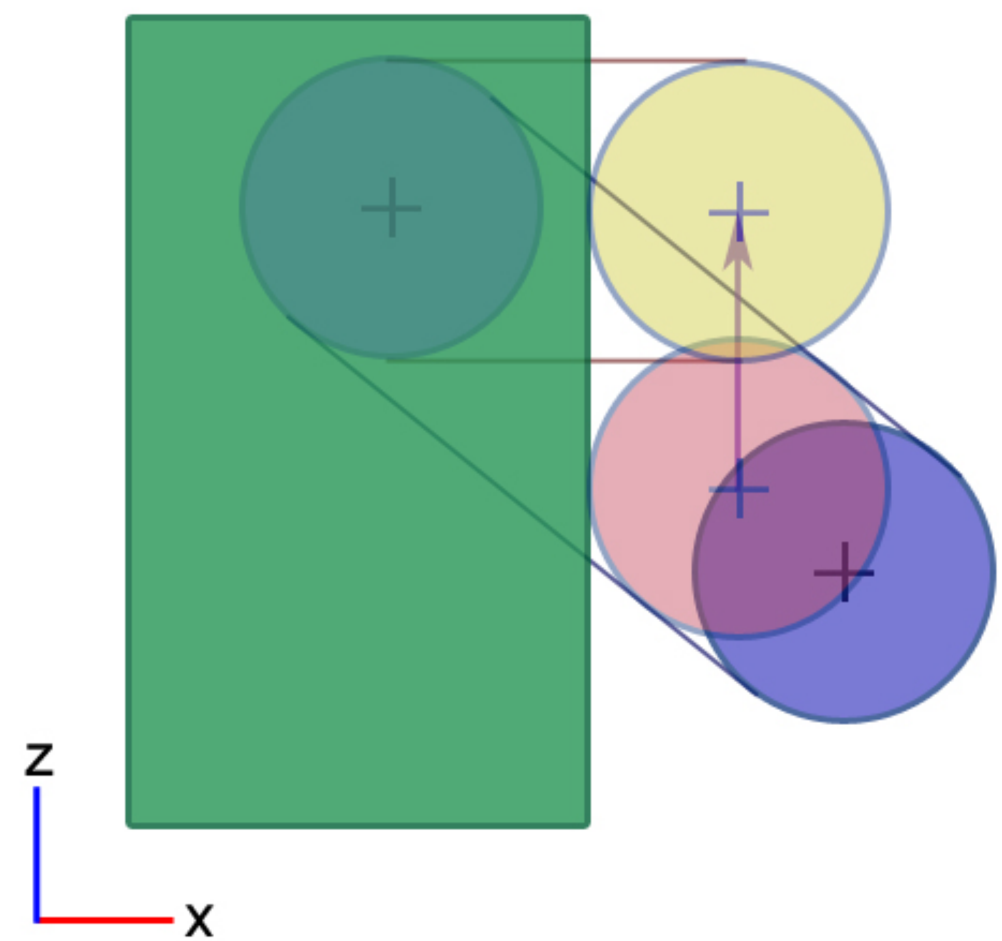
ROLLBACK RESPONSE CHALLENGES

- Response may not bring object to the other side of a wall
  - Include a velocity check to ensure object leaves wall
- Object can also be moved to just outside the wall
  - May have accuracy issues when applying velocity/force to move object



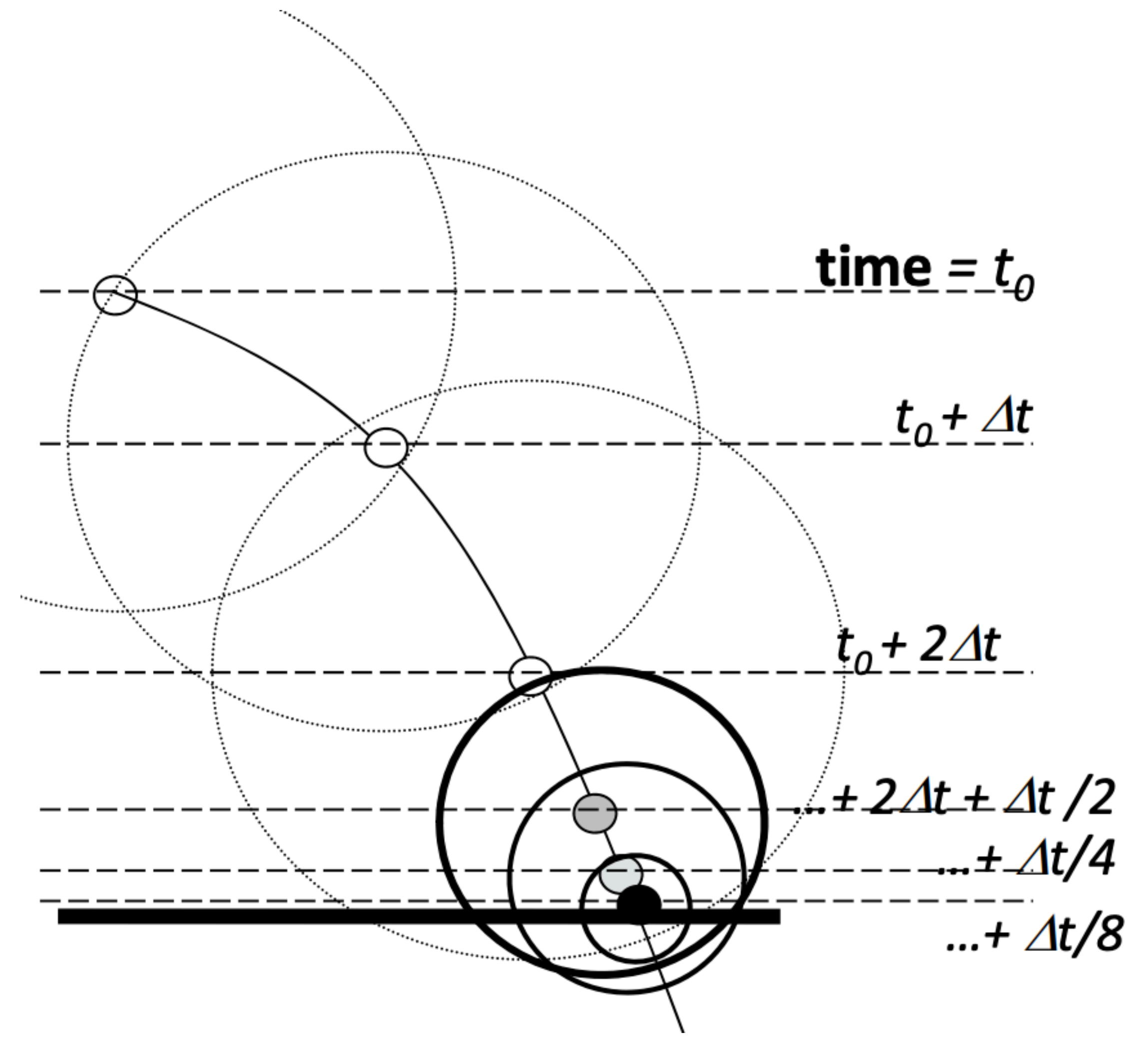
PREDICTION RESPONSE

- Sweep along object's trajectory to determine when collision will occur within a timestep
- Reduce time steps to determine when the collision happens and respond accordingly



PREDICTION RESPONSE CHALLENGES

- More initial bookkeeping but saves retroactive computation
- Must pick bounds of sweep with care
  - Possible to miss check if bounds do not match particle velocity
- Must perform iterative timesteps for all simulated objects in the scene



ISSUES WITH LARGE SYSTEMS

- Both responses have performance issues as number of simulated objects grows
- Must reduce time step for all objects when a collision is about to occur/has occurred for one object even if objects are unrelated
- Timewarp simulation allows for adaptive time steps only on colliding rigid body and its related bodies
  - Multibodies
  - Contact bodies
  - Active bodies

WHAT ABOUT RESTING CONTACT?

- Very difficult case!
- Issues with numeric precision and time step continuity
  - i.e. We can almost never zero out forces
  - Must check if forces are below some threshold and treat them as zero

RESTING CONTACT

https://www.reddit.com/r/GamePhysics/comments/d8o4yt/borderlands_3_i_honestly_shouldve_seen_this_coming

RIGID BODIES

- Particles are limited in terms of simulating real world interactions
  - Need volume to capture more behaviors
- Rigid bodies are an approximation of physically-based volumetric bodies
  - Assume (incorrectly) that their shape cannot be deformed
  - Have a center of mass
  - Have angular velocity

RIGID BODIES AND ROTATION

- Rigid bodies rotate around their center of mass
  - Calculated as the center of the geometry (centroid)
  - Assumes uniform density
- In 2D, rigid body rotations always occur within a plane
  - Can represent orientation as a radian (scalar)
  - Can also represent angular velocity, angular acceleration, moment of inertia, and torque as scalars
- In 3D, rigid body can rotate along any axis
  - Can represent orientation as a quaternion
  - Can represent moment of inertia as a 3x3 matrix but it varies based on orientation (which changes over time)

COLLISION DETECTION FOR RIGID BODIES

- Two stage process: broad and narrow phase
- Broad phase catches potential intersections across all rigid body pairs
- Narrow phase confirms if broad phase candidate pairs actually did collide

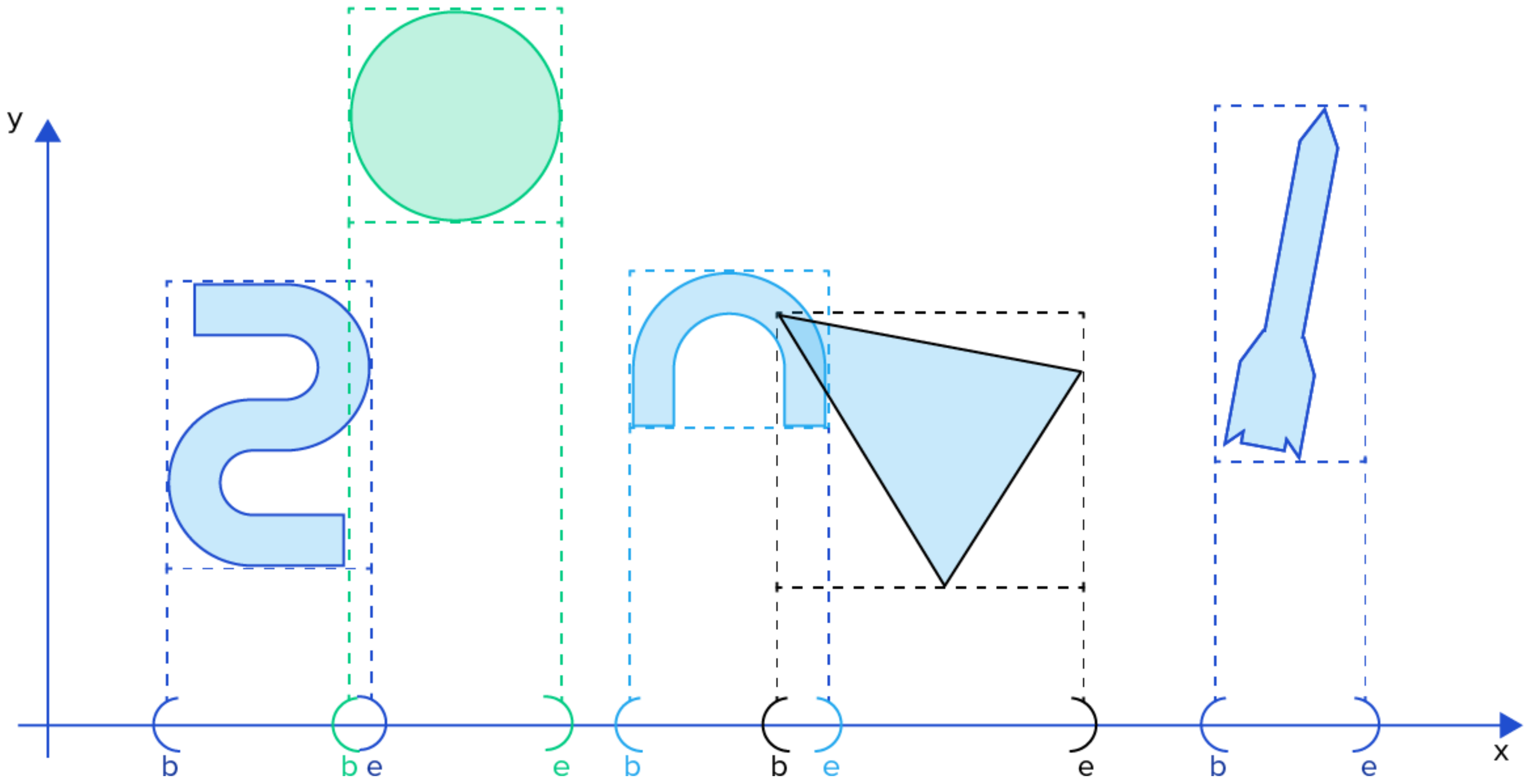
BOUNDING BOXES AND SPATIAL STRUCTURES

- Detecting for every polygon in a mesh is inefficient
  - Bounding boxes simplify detection
  - Use of course-scale versus fine-scale detection can also reduce calculations
  - Volumes usually axis-aligned (AABBs) for computation efficiency
- Detecting all potential collisions at every time step is also inefficient
  - Spatial structures reduce number of checks
  - Use of trees limit potential points of contact within a region

BROAD PHASE ALGORITHMS

- Sort and Sweep
  - Choose one axis to project all AABBs onto by beginning and end of volume
  - Check active intervals (where bounding volume has begun but not ended) and add pairs of volumes as candidates for narrow phase detection
  - Can re-sort easily between time steps
  - Can alternate axes to reduce false positives

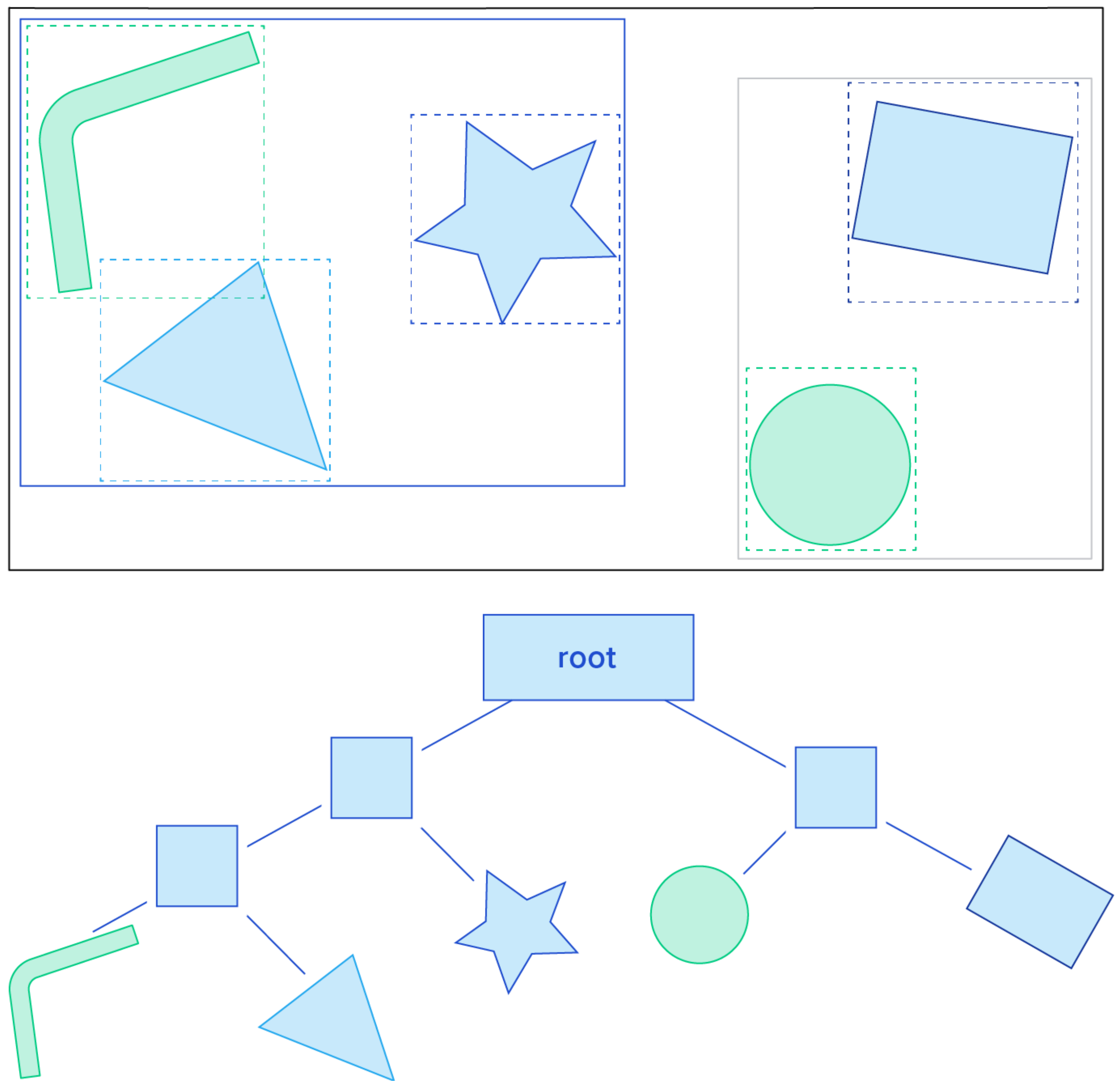
SORT AND SWEEP



BROAD PHASE ALGORITHMS

- Dynamic Bounding Volumes
  - Binary tree that stores hierarchy of AABBs
  - Object AABBs stored at leaves
  - Can traverse tree to determine candidate pairs for narrow phase
  - Can be rebalanced between timesteps

DYNAMIC BOUNDING VOLUMES

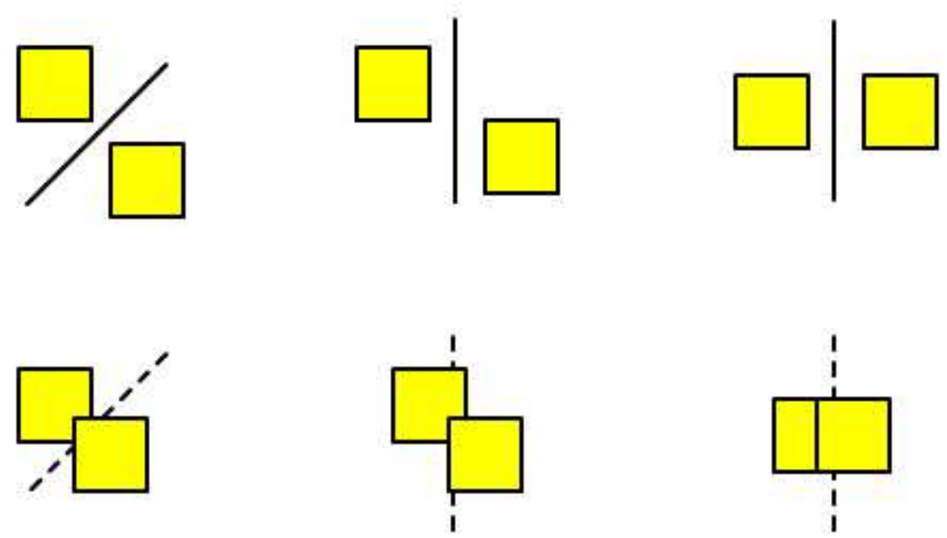


NARROW PHASE

- Determines which of the candidate pairs are intersecting and at what point
- Need a more precise bounding volume than the AABBs for accuracy
  - Enclose object in a **convex hull**, or smallest shape that completely encloses object with no concavity
- Many algorithms for testing if objects are intersecting, at what point of contact, at what depth etc

SEPARATING AXIS THEOREM

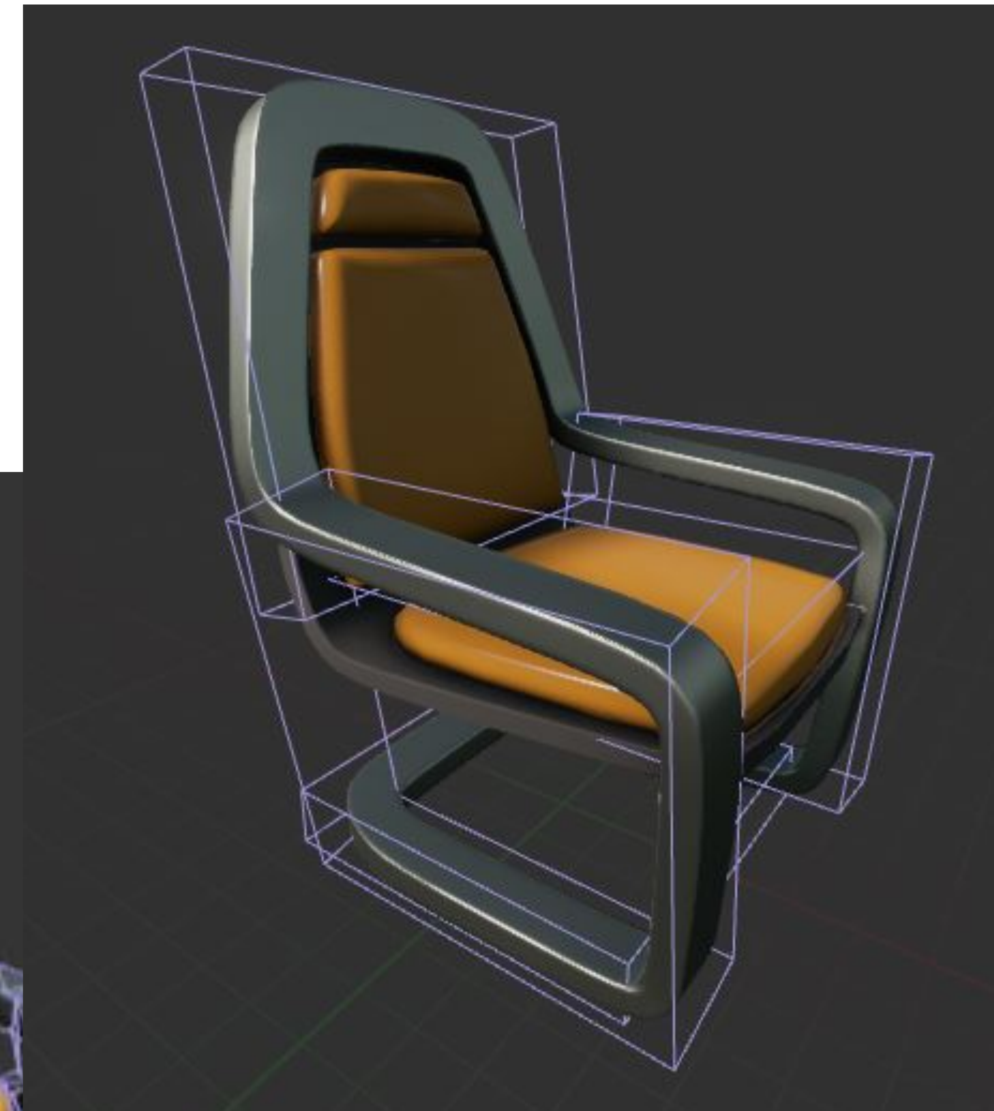
- There should exist at least one axis where orthogonal projections of objects onto axis do not overlap if convex shapes do not intersect
    - Basically if you can create a plane between two 3D shapes, they are not intersecting
- One of many techniques for determining separability in the narrow phase



AN ADDITIONAL NOTE ON CONVEX HULLS

- Sometimes objects need concavity
- Can create a concave object out of several convex hulls (convex decomposition)
  - Can automatically generate group of convex hulls
  - Can have artist manually create convex hulls

CONVEX DECOMPOSITION IN UNREAL



RESOLVING MULTIPLE COLLISIONS

- Can resolve all collisions simultaneously or handle them sequentially
- Simultaneous Collision Handling
  - Detect and calculate all collisions in a given time step
  - Apply necessary forces to all involved objects
  - Highly parallel
  - Incorrect behavior (does not handle transfer of force)
- Sequential Collision Handling
  - Detect and calculate collisions in a given time step serially
  - Resolve collision then resolve additional collisions caused by changes to the system
  - Done in a single time step

STILL MANY CHALLENGES…

- Both types of collision resolvers can be inaccurate in certain situations
  - Hybrid methods exist but also can have issues
- The point is, simulating physically-based things is inherently inaccurate so must decide how best to balance realism, speed, and tunability
- And we haven't even talked about the hard stuff like cloth and hair simulation…

TWEAKING PHYSICS TO MAKE THINGS FUN

- Example:
  - Elsa's hair flip is commonly considered an animation "error"
  - Probably a creative decision because the animation looked better ignoring collisions than respecting them

https://www.youtube.com/watch?v=SxSV9RxG7JM

WHAT ABOUT GAMES?

- Can't fake things as easily in games!
- Player has control of camera and change the simulation with their actions
  - …and arguably physics glitches are sometimes a feature…

https://www.youtube.com/watch?v=pqiZFuSeUCk

ADDITIONAL RESOURCES

- [https://www.gamasutra.com/view/feature/131834/collision_response_bouncy_.php]
- [https://www.scss.tcd.ie/~manzkem/CS7057/cs7057-1516-09-CollisionResponse-mm.pdf]
- [https://www.merl.com/publications/docs/TR2000-17.pdf]
- [https://roystanross.wordpress.com/2014/05/07/custom-character-controller-in-unity-part-1-collision-resolution/]
- [https://www.toptal.com/game/video-game-physics-part-i-an-introduction-to-rigid-body-dynamics]
- [https://gamedevelopment.tutsplus.com/tutorials/collision-detection-using-the-separating-axis-theorem--gamedev-169]
- [https://www.myphysicslab.com/engine2D/collision-methods-en.html]