

# Game Developers Conference®

February 28 - March 4, 2011  
Moscone Center, San Francisco

[www.GDCConf.com](http://www.GDCConf.com)

The logo for the Game Developers Conference (GDC) 2011. The letters 'GDC' are rendered in a large, bold, 3D font with a yellow-to-white gradient and a multi-colored outline (cyan, magenta, yellow). To the right of the 'C' is a small, tilted square icon containing the number '25', representing the 25th anniversary of the conference.

GDC

# *Physics for Game Programmers*

## Contacts

or

## “How (Not) To Make It Stick”

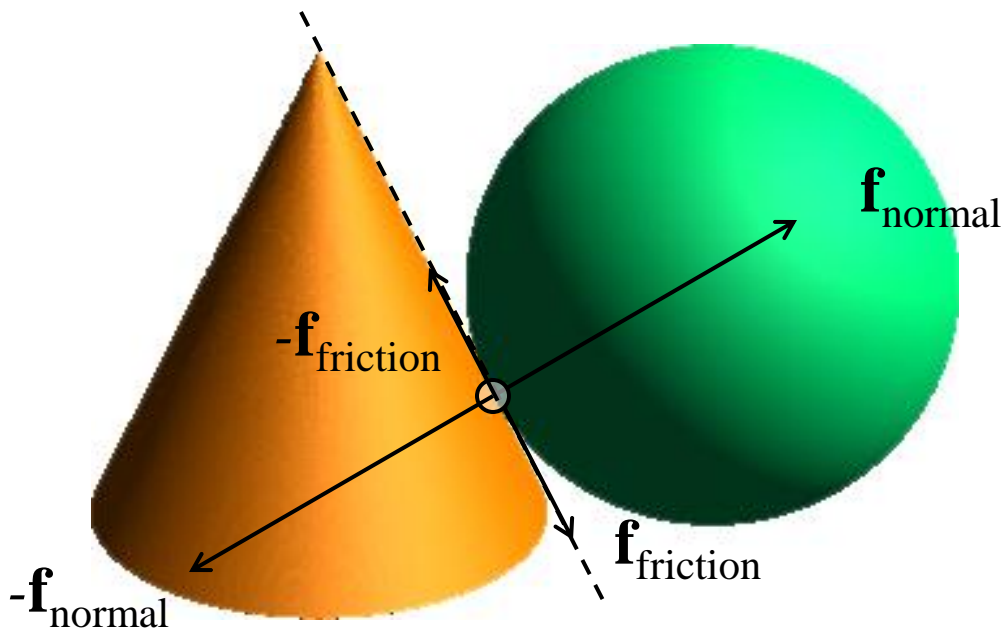
Gino van den Bergen

[gino@dtecta.com](mailto:gino@dtecta.com)

# Contacts Are Tricky

- ⊗ Act only in one direction (do not pull).
- ⊗ Commonly exist for a short period only.
- ⊗ Occur anywhere on an object's surface.
- ⊗ Involve sliding and rolling.
- ⊗ Multiple solutions due to overconstraining

# Contact Plane



# Contact Plane (Cont'd)

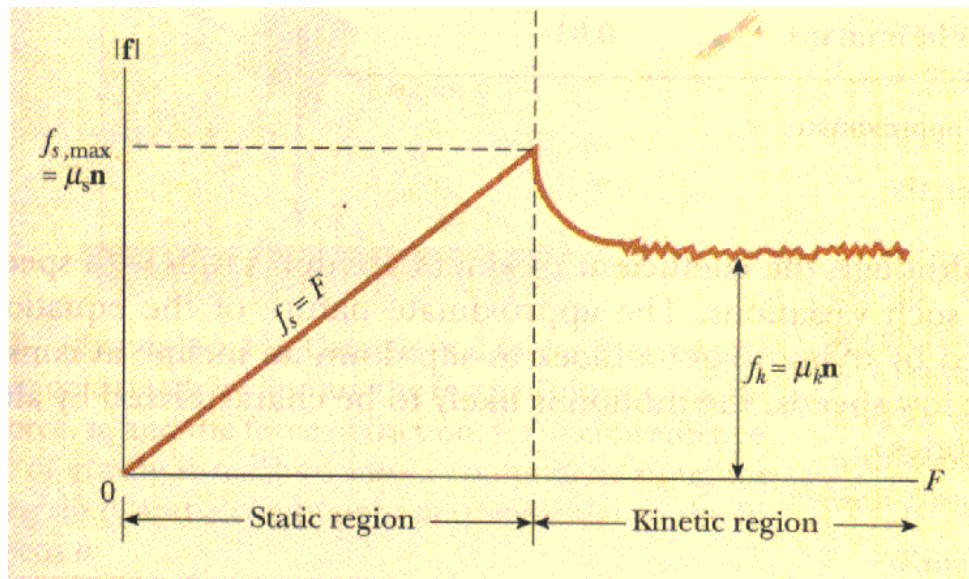
- ⊕ Normal forces/impulses act along the plane normal.
- ⊕ Friction forces/impulses act in the contact plane.
- ⊕ Forces/impulses apply to the contact point(s).

# Coulomb or “Dry” Friction

- ⊕ The maximum tangential force to the contact plane exerted by friction is equal to the normal force times the coefficient of friction

$$\mathbf{f}_{\text{friction}} \leq \mu \mathbf{f}_{\text{normal}}$$

# Static vs Kinetic Friction

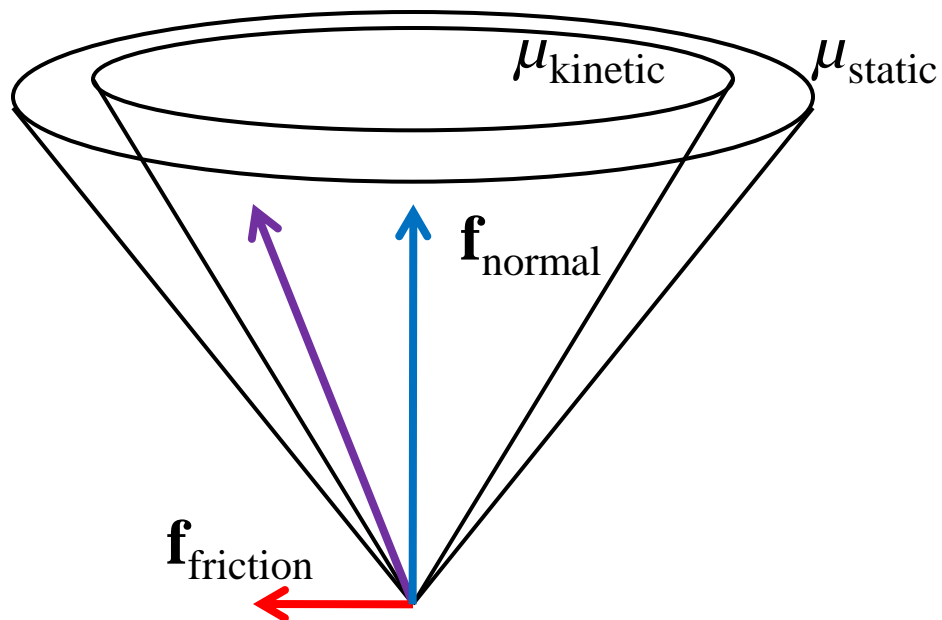


# Friction Cone

- ⊕ The total force exerted by a contact is normal force plus friction.
- ⊕ Coulomb's law constrains the total force to a cone.
- ⊕ Friction forces can reduce tangential velocity but cannot revert it.



# Friction Cone (Cont'd)



# Solving Contacts

- ⊕ Determine contact impulses that filter out inadmissible relative velocities.
- ⊕ Impulses do not increase momentum.
- ⊕ Restitution of velocity (rebound) only at impact and only in normal direction.
- ⊕ Restitution is zero for resting contacts.

# Iterative Methods

- ⊕ Are generally cheaper than direct methods:  $O(n)$  as opposed to  $O(n^3)$
- ⊕ Always return a solution (even when there isn't one).
- ⊕ Exploit frame coherence (warm starting).
- ⊕ Converge somewhat unpredictably.

# Gauss-Seidel Method

- ④ Solve each variable in a linear system in isolation using the best approximation so far for the other variables.
- ④ Converges for positive-definite systems. Any physics-based system (involving inertia) is bound to be positive definite.

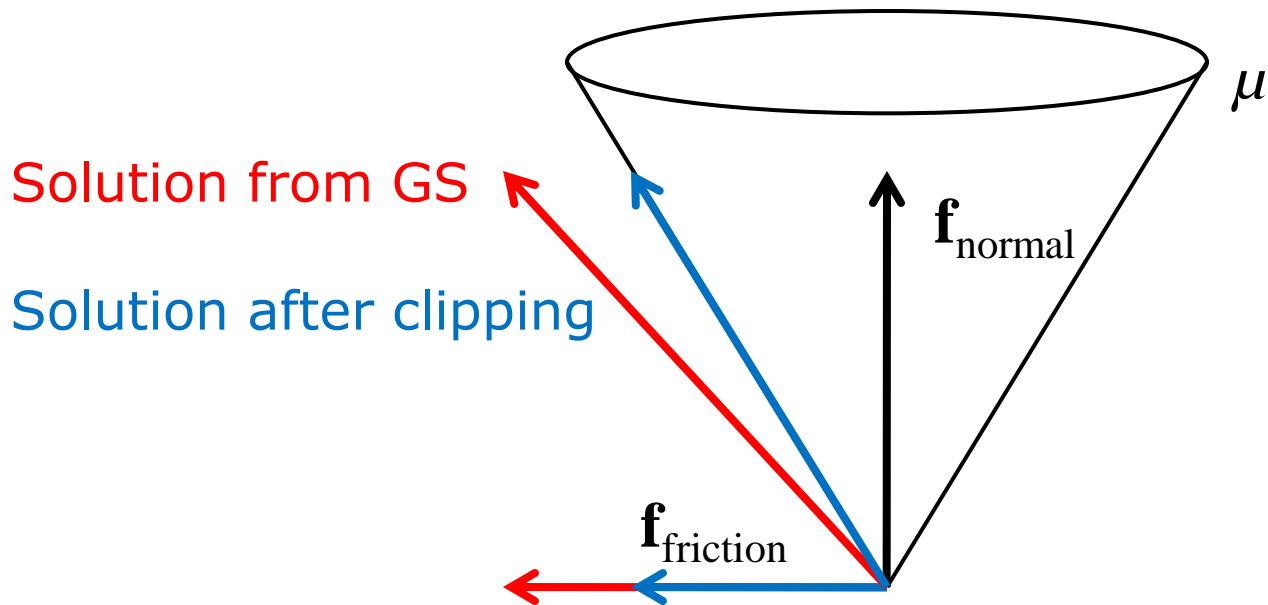
# Solving Contacts: Step #1

- ⌚ Solve contacts as ball joints, i.e. compute impulses that keep contacts glued.
- ⌚ Solve simultaneous contacts using a Gauss-Seidel (block) solver.
- ⌚ Contacts are solved one at a time.
- ⌚ Momentum changes with each step.

# Solving Contacts: Step #2

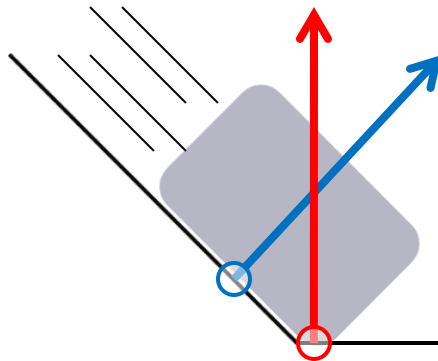
- ⌚ The solution returned by the GS solver is “clamped” to the applicable friction cone:
- ⌚ Negative (pulling) impulses are removed.
- ⌚ Friction impulses that exceed the friction cone are clipped.
- ⌚ We call this *Projected* Gauss-Seidel.

# Clip Against The Friction Cone



# Beware of Sliding Contacts!

- ⊕ Prioritize contacts on relative velocities in the normal direction.



Red contact should be solved before blue one in GS solver. Prefer normal impulses over friction. The latter may get clipped.



# Solving Drift

- ⊕ Solve drift either by:
- ⊕ Adding a tiny target velocity that moves the bodies out (Baumgarte stabilization).
- ⊕ Solving penetrations independent of velocities (post-stabilization).

# Baumgarte Stabilization

- ⊕ Additional impulses that correct constraints also add momentum.
- ⊕ Effectively, a tiny spring is added that may cause oscillations.
- ⊕ Needs some tweaking to get right.

# Post-Stabilization

- ④ Compute impulses that correct drift in one time step but add them to a pseudo-momentum that is zeroed each frame.
- ④ Since we fix relative position (not velocity) we may pre-integrate these pseudo-impulses by the time step.

# Post-Stabilization (Cont'd)

- ⊕ These pre-integrated impulses (*nudges*) result in linear and angular displacement.
- ⊕ Iteratively find simultaneous nudges that restore the constraints.
- ⊕ Cline & Pai arrive at the same solution coming from the other end.

# Solving Penetration

- ⌚ Don't try to solve penetration completely!
- ⌚ In-and-out contacts are a source of jitter.
- ⌚ Often penetration is inevitable. Solving it completely may result in a blowup.
- ⌚ Successive under-relaxation ( $\omega < 1$ ) helps.

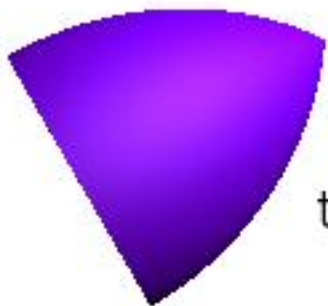
# Collision Detection

- ④ Find all pairs of objects that are colliding now, or will collide over the next frame.
- ④ Compute data for response: normal and contact points.
- ④ Penetration depth is distance between points.

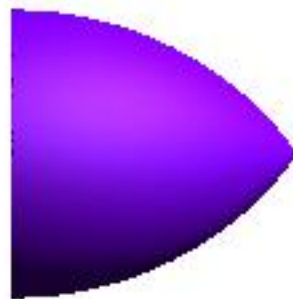
# Collision Objects

- ④ Static environment (buildings, terrain) is typically modeled using polygon meshes.
- ④ Moving objects (player, NPCs, vehicles, projectiles) are typically convex shapes.
- ④ We need to detect convex-convex and convex-mesh collisions.

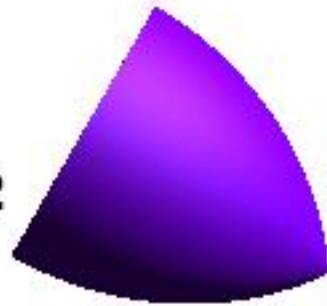
# Continuous Collision Detection



t=0



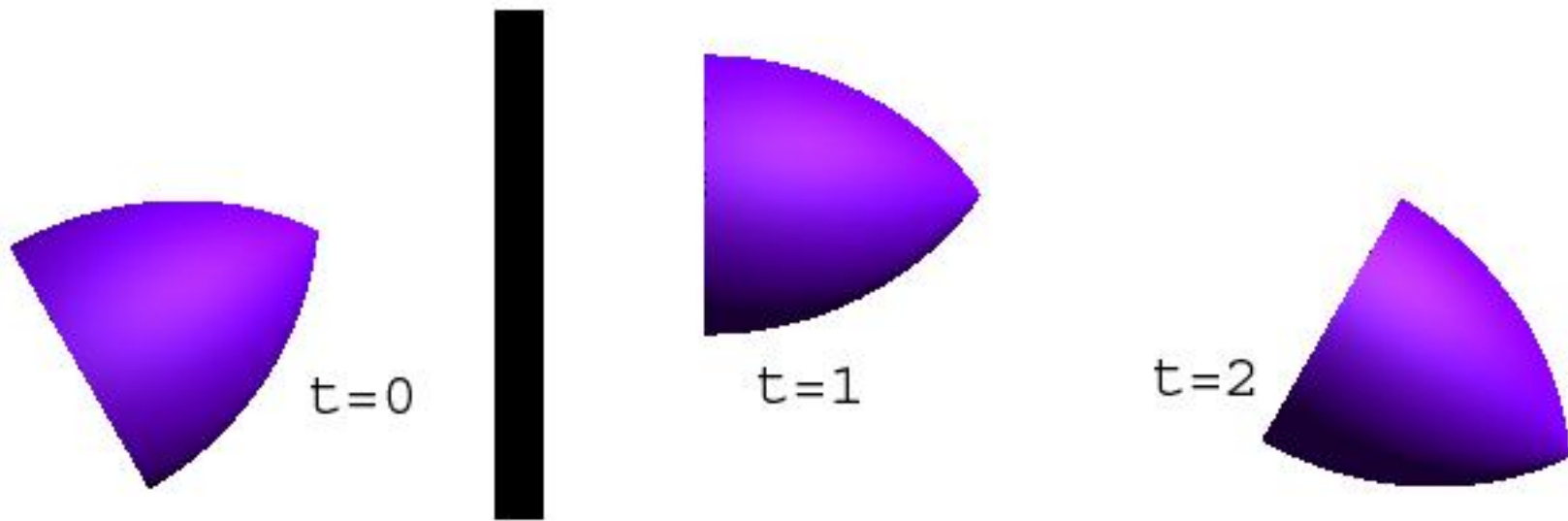
t=1



t=2



# Continuous Collision Detection



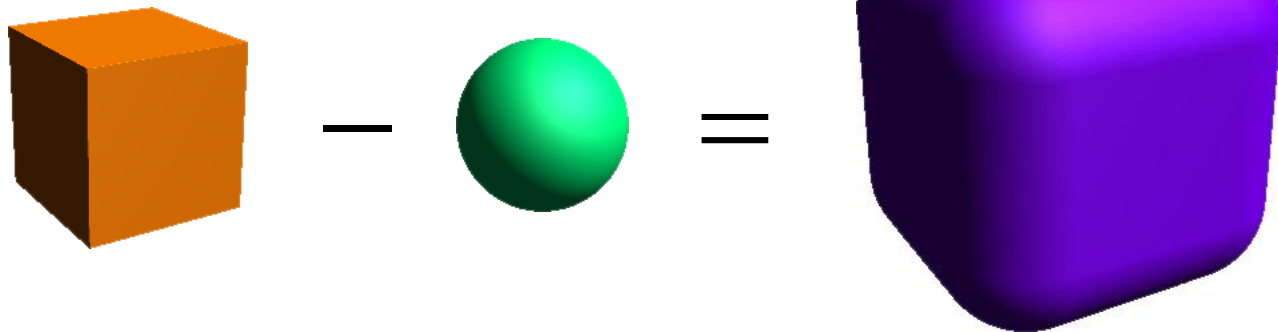
# Configuration Space

- ⊕ The *configuration space obstacle* of objects  $A$  and  $B$  is the set of all vectors from a point of  $B$  to a point of  $A$ .

$$A - B = \{\mathbf{a} - \mathbf{b} : \mathbf{a} \in A, \mathbf{b} \in B\}$$

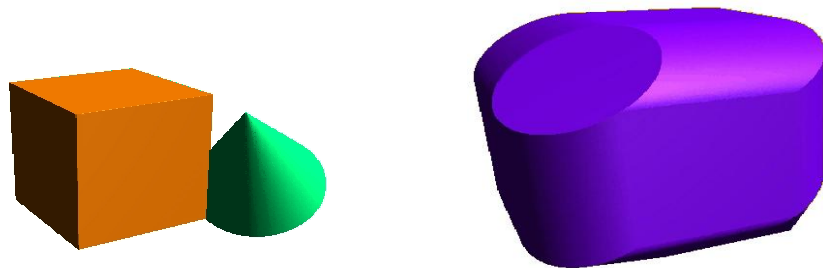
# Configuration Space (Cont'd)

- CSO is basically one object dilated by the other:



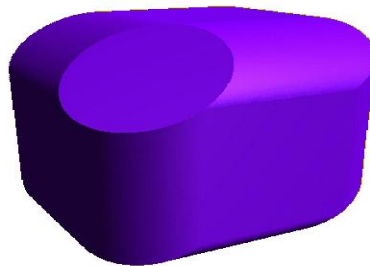
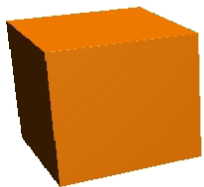
# Translation

- ⊕ Translation of  $A$  and/or  $B$  results in a translation of  $A - B$ .



# Rotation

- ⊕ Rotation of  $A$  and/or  $B$  changes the shape of  $A - B$ .



# Configuration Space?

- ⊕ Collision queries on a pair of convexes are reduced to queries on the position of the origin with respect to the CSO.
- ⊕ Point queries are easier than queries on pairs of shapes.

# Queries

- ⊕ The distance between two objects is the distance from the origin to the CSO.

$$d(A, B) = \min \{ \|\mathbf{x}\| : \mathbf{x} \in A - B \}$$

- ⊕ The objects intersect if the origin is contained by the CSO.

$$A \cap B \neq \emptyset \Leftrightarrow \mathbf{0} \in A - B$$

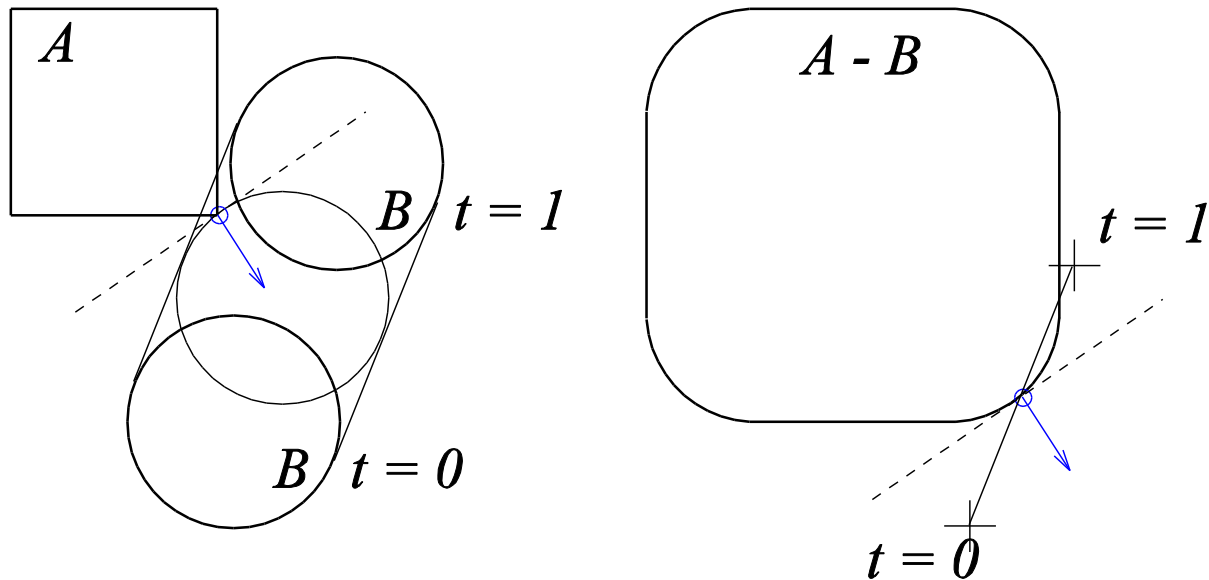
# Queries (Cont'd)

- ⊕ Whether and when two translated objects ever come in contact boils down to a ray query from the origin onto the CSO.
- ⊕ For  $A$  translated over  $s$  and  $B$  over  $t$ , the ray is cast from the origin along  $\mathbf{r} = \mathbf{t} - \mathbf{s}$ .

$$\min\{\lambda : \lambda\mathbf{r} \in A - B, 0 \leq \lambda \leq 1\}$$



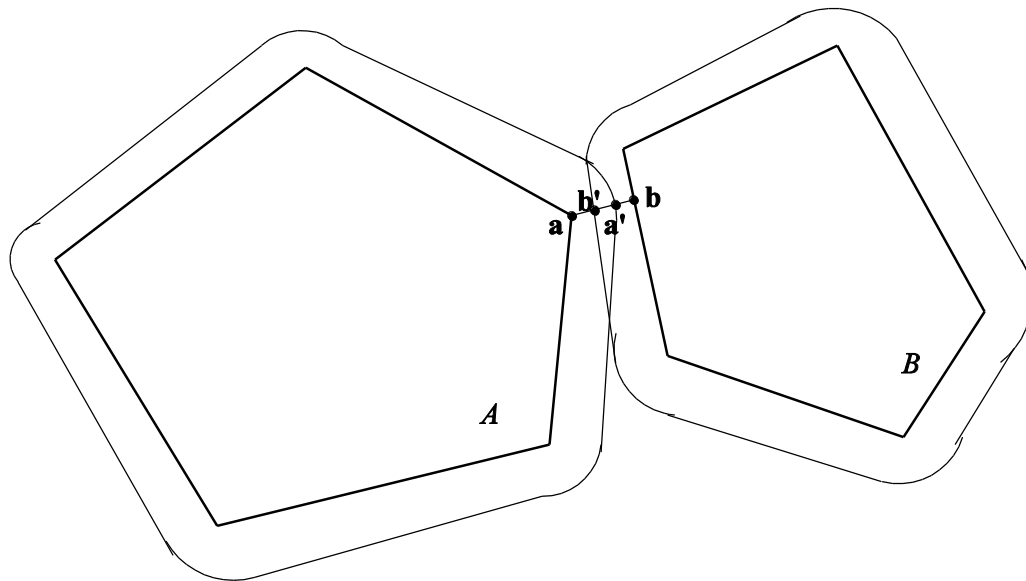
# Ray Query on the CSO



# Resting Contacts

- ⊕ Contact data for resting contacts are obtained through a hybrid approach.
- ⊕ Objects are dilated slightly to add a skin.
- ⊕ For interpenetrations that are only skin-deep the closest points of the “bones” give us the contact data.

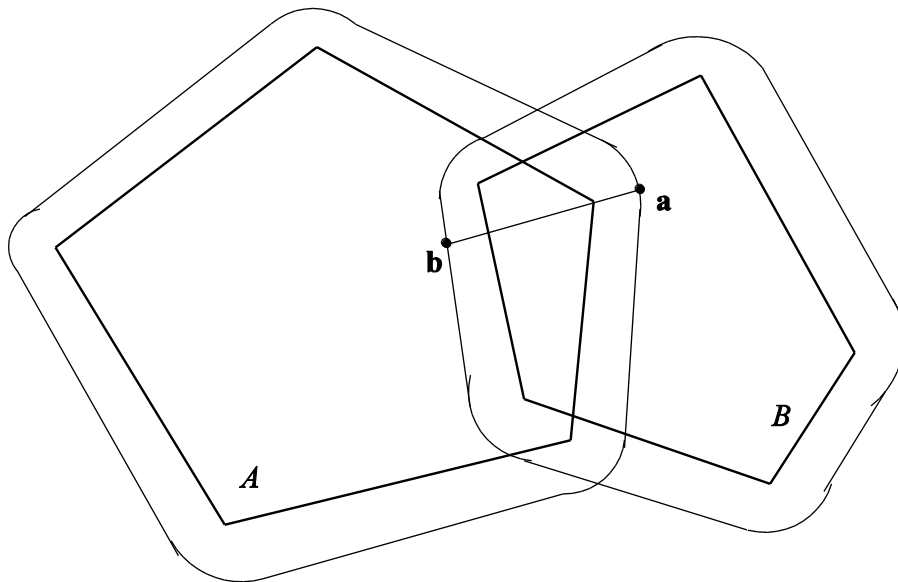
# Shallow Interpenetrations



# Resting Contacts (Cont'd)

- ⊕ For deeper interpenetrations contact data are obtained from the penetration-depth vector.
- ⊕ This should only be necessary in emergencies.

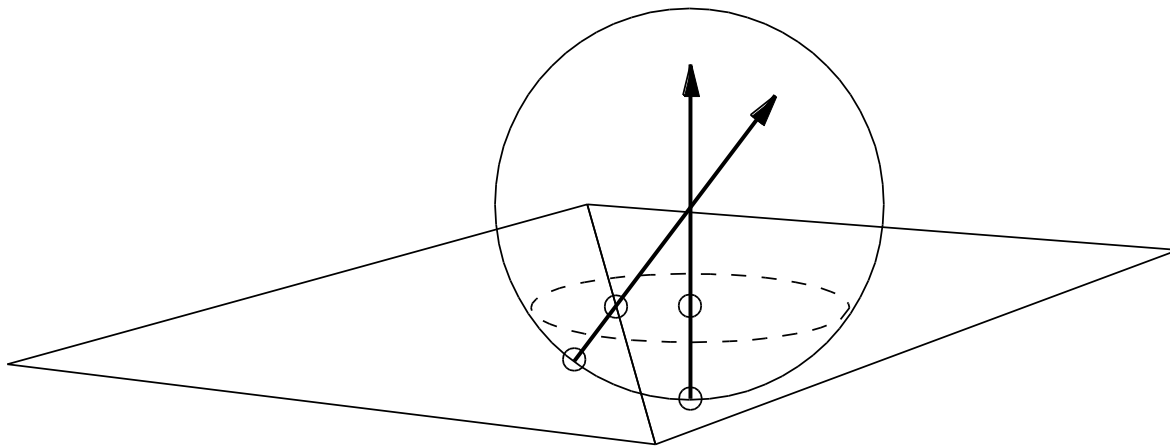
# Deep Interpenetrations



# GJK Saves the Day

- ④ GJK is an iterative method that computes closest points.
- ④ The GJK ray cast can perform continuous collision detection.
- ④ The *expanding polytope algorithm* (EPA) returns the penetration depth vector.

# Meshes Have Bumpy Edges

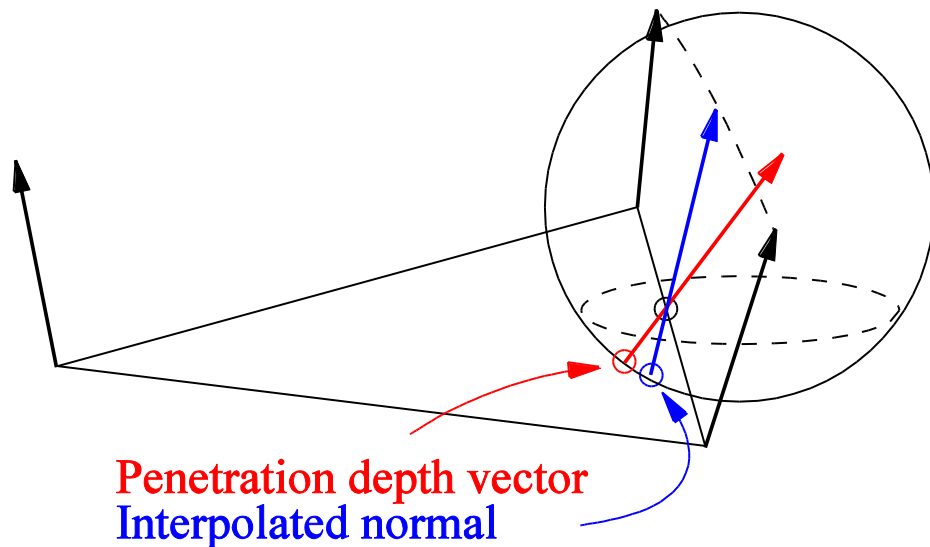


# Solving Bumpy Edges

- ④ GJK returns barycentric coordinates of the closest point.
- ④ Use these coordinates to interpolate the vertex normals.
- ④ Similar to Phong shading: Use a normalized lerp.



# Smooth Interpolated Normals



# References

- ④ Erin Catto. *Iterative Dynamics with Temporal Coherence*. GDC 2005.  
[box2d.org](http://box2d.org)
- ④ M.B. Cline and D.K. Pai, *Post-Stabilization for Rigid Body Simulation with Contact and Constraints*. In Proc. ICRA, 2003

# References (Cont'd)

- ④ Eran Guendelman, Robert Bridson, and Ronald Fedkiw. *Nonconvex rigid bodies with stacking*. In Proc. SIGGRAPH, 2003.
- ④ Chris Hecker. *How to Simulate a Ponytail*. Game Developer Magazine, 2000.  
`chrishecker.com`

# References (Cont'd)

- ④ Gino van den Bergen. *Collision Detection in Interactive 3D Environments*. Morgan Kaufmann Publishers, 2004.
- ④ Gino van den Bergen. *Smooth Mesh Contacts with GJK*. In *Game Physics Pearls*, A K Peters, 2010.

# Thank You!

» For papers and other information, check:

`www.dtectata.com`

`www.gamephysicspearls.com`