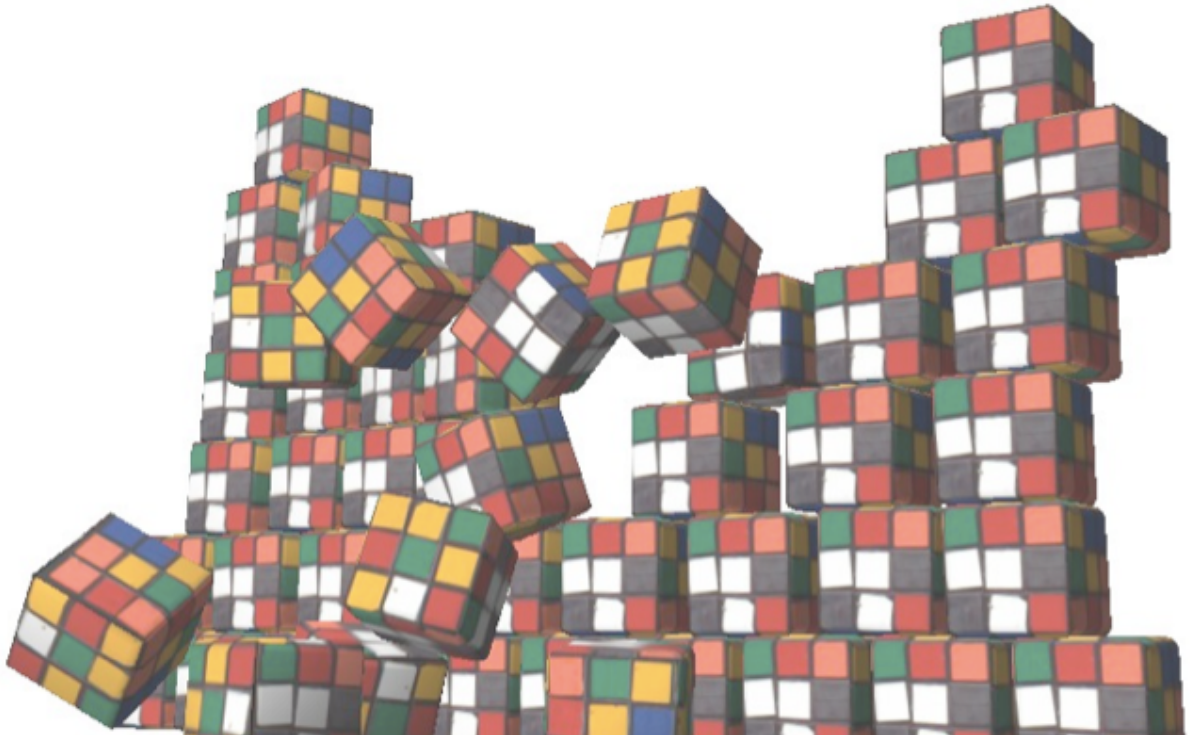
Rigid Body Simulations

Thuerey / Game Physics



Re-cap Types of Materials

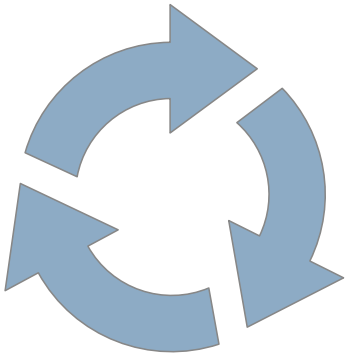
- Simple Particles
  - Great for … particles without interaction
  - Not really for solid materials, more for visuals
- Mass-spring Systems
  - Can model elastic ropes, sheets and bodies
  - Simple model, fast
  - Stiffness / discretization difficult to fine tune
  - Stability problems for stiff materials

Rigidity

- All materials are elastic to some extent in reality
- Example: try to model metal bar with high stiffness
  - Will enforce inter object distances
  - Propagate deformation, real behavior in the limit
- Problem: high stiffness means
  - large forces
  - tiny time step for stability

Observation

- High stiffness means: vertices should not move w.r.t. each other
  - Effectively removes degrees of freedom from the system
- Obvious: don't even simulate them in the first place
- New representation: center of mass and orientation
- Consequence: new equations of motion for rotational component



Overview

- Rigid bodies in 2D
  - Orientation
  - Integrating rotational motion
  - Angular momentum
  - Impulses
- Orientations in 3D
- Rigid bodies in 3D

Learning Outcomes

- Be able to explain the rigid body representation
- Know how to integrate linear and angular values in time (in 2D and 3D)
- Realize options and differences for orientations
- Be able to calculate and apply collision response
- Know how to implement…

Translations

- Center of mass : Schwerpunkt
- Momentum : Impuls
- Angular momentum : Drehimpuls
- Torque : Drehmoment
- Inertia tensor : Traegheitsmoment

  (also “moment of inertia”)
- Impulse : Impuls (!?)

Points vs. Rigid Bodies

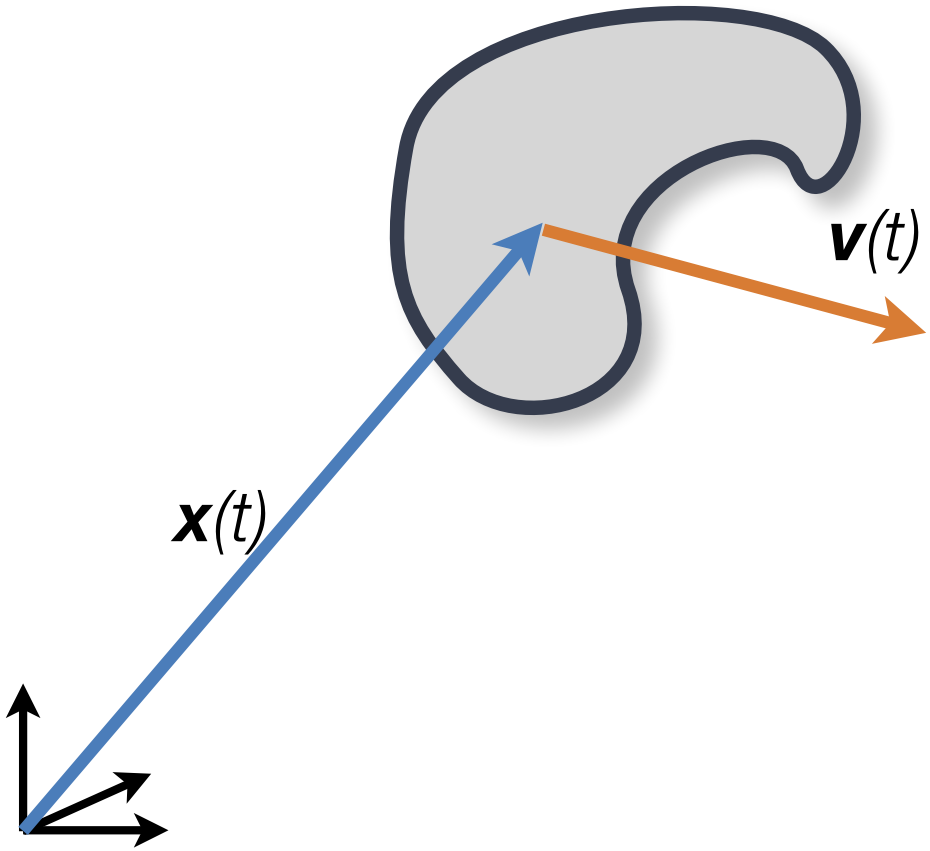
- For particles:
  - Position ***x***
    Velocity ***v***
    Mass *M*
- Dynamics:

$$\mathbf{v}(t) = \frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t}$$

$$\mathbf{a}(t) = \frac{\mathrm{d}\mathbf{v}(t)}{\mathrm{d}t}$$

- For a rigid body:
  - Position ***x***
  - ?
  - Velocity ***v***
  - ?
  - Mass *M*
  - **?**

Representation



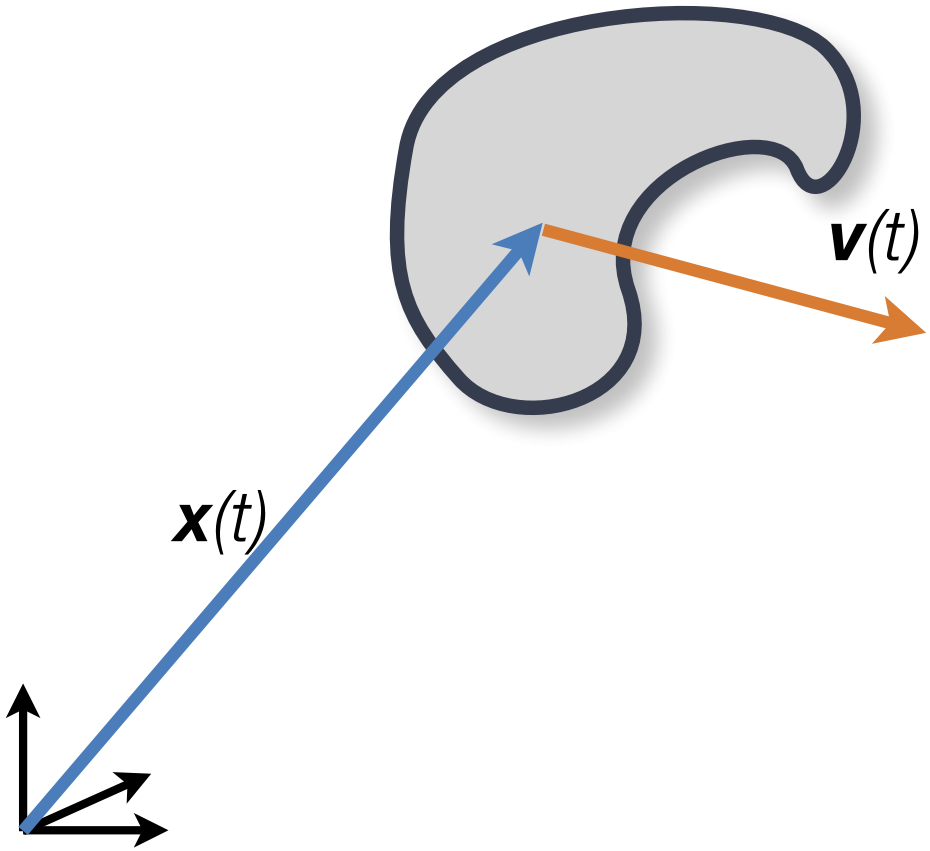
- Reference point on body: center of mass
- Continuous:

$$\mathbf{x}_{cm} = \frac{\int \mathbf{x}\rho(x)\mathrm{d}V}{\int \rho(x)\mathrm{d}V}$$

- Discrete:

$$\mathbf{x}_{cm} = \frac{\sum_i m_i \mathbf{x_i}}{\sum_i m_i}$$

Representation



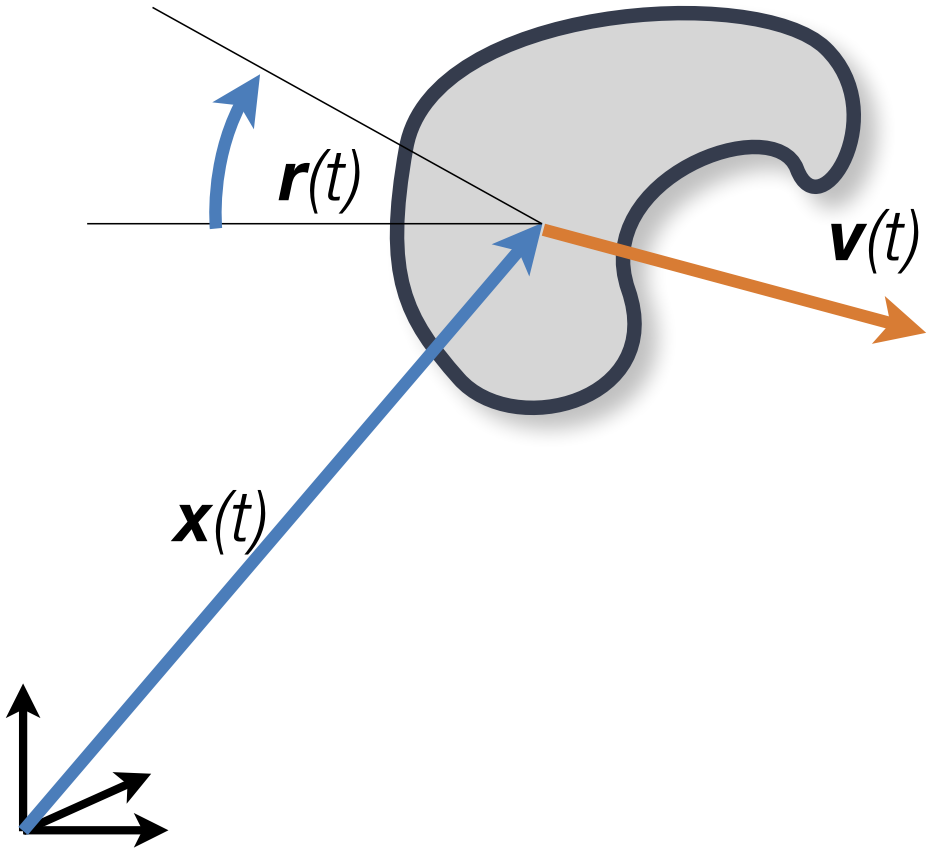
- Center of mass behaves like a mass point with total mass of the body $M = \sum_i m_i$

$$\mathbf{x}_{cm} = \frac{\sum_i m_i \mathbf{x_i}}{M}$$

$$\mathbf{v}_{cm} = \frac{\sum_i m_i \mathbf{v_i}}{M}$$

$$\mathbf{a}_{cm} = \frac{\sum_i m_i \mathbf{a_i}}{M}$$

Representation



- Orientation: rotation around center of mass
- Point coordinates relative to center of mass (body space)
- Absolute position (world space)

$$\mathbf{x}_i^{world} = \mathbf{x}_{cm}(t) + \mathrm{Rot}_{r(t)}(\mathbf{x}_i)$$

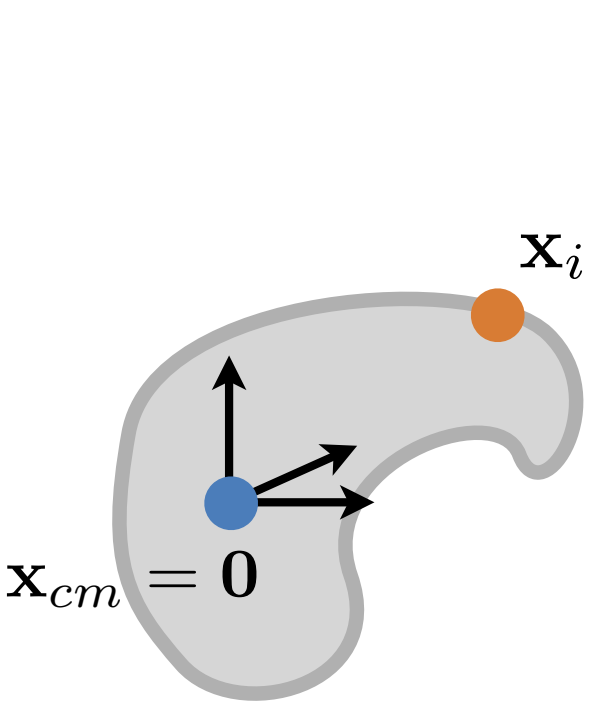
More on the details of orientations later...

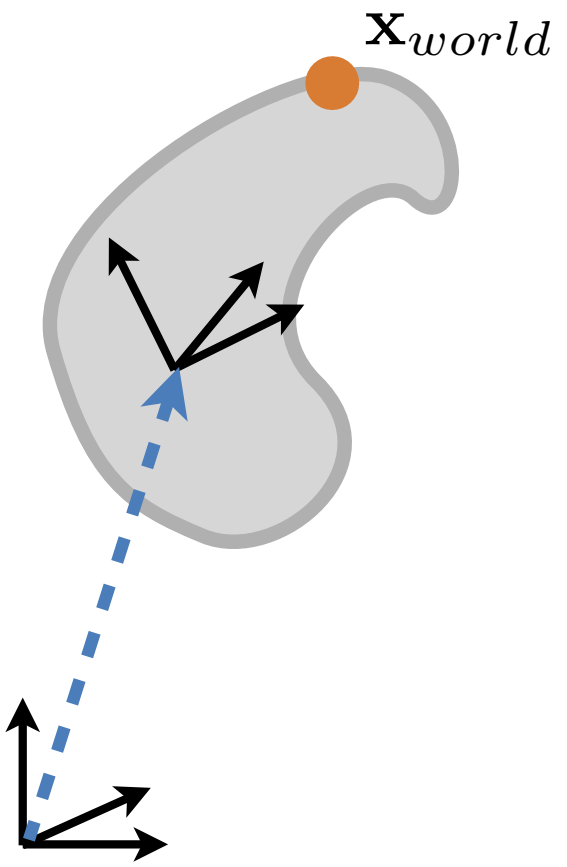
Representation

Body space

World space

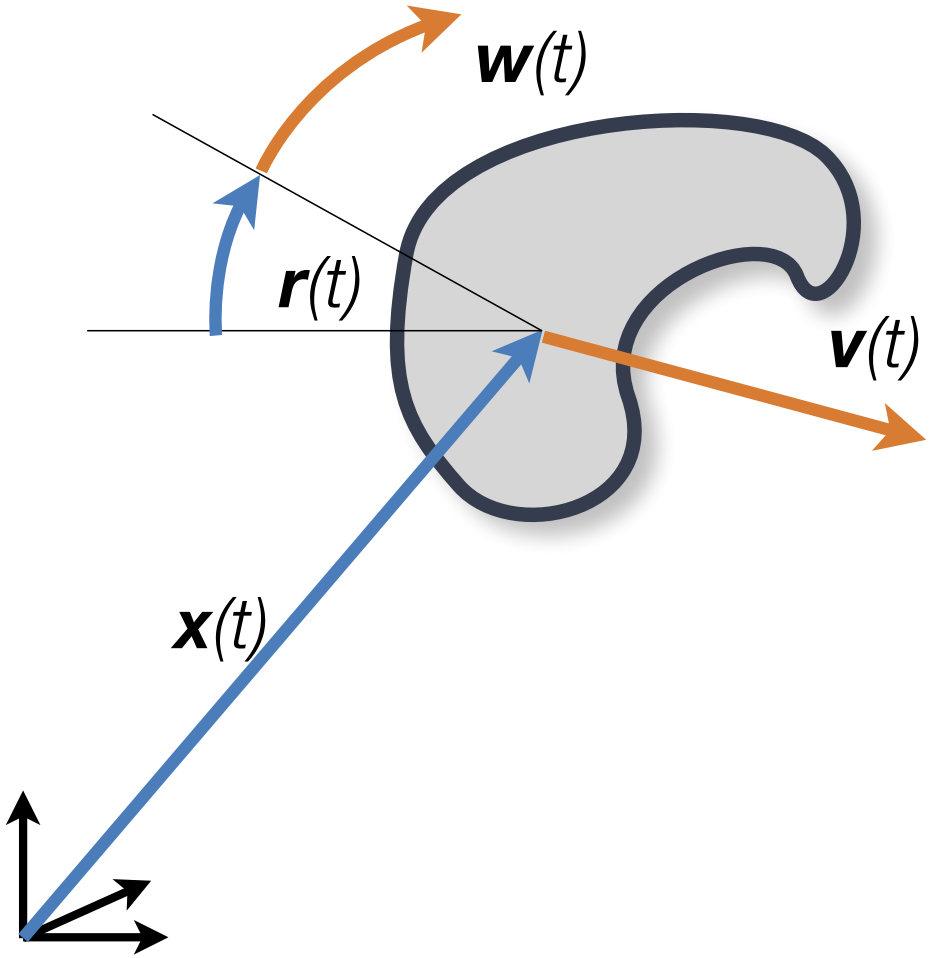
$$\mathbf{x}_i^{world} = \mathbf{x}_{cm}(t) + \mathrm{Rot}_{r(t)}(\mathbf{x}_i)$$





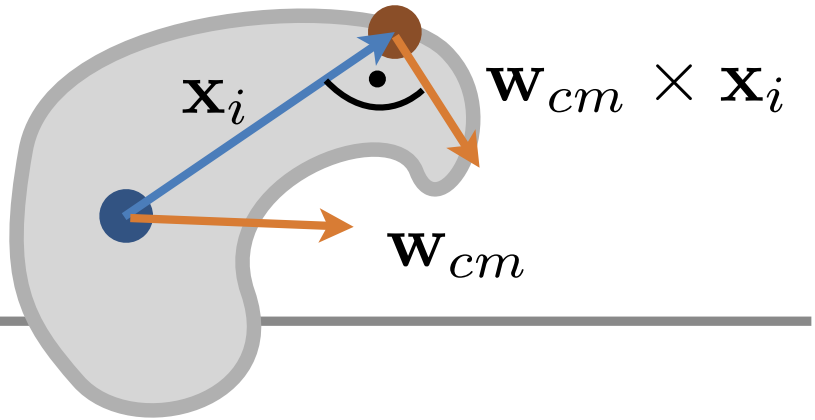
(For convenience, keep center of mass at zero.)

Representation



- Previously: linear velocity
- Now also: angular velocity
- 3 component vector encoding rate of angular change, and axis of rotation
- Total velocity of a point:

$$\mathbf{v}_i = \mathbf{v}_{cm} + \mathbf{w}_{cm} \times \mathbf{x}_i$$



Orientation & Angular Velocity in 2D

- Only rotation around z, so ***w*** is a single number
- Usually given in radians
- Sometimes denoted with ω
- Note: orientation ***r*** is limited to [0 .. 2*pi], while angular velocity ***w*** can take arbitrary values
- Velocity of a point is given by:

$$\mathbf{v}_i = \begin{pmatrix} -w\ x_{i,y} \\ w\ x_{i,x} \end{pmatrix}$$

- Apply orientation to point with matrix:

$$\mathrm{Rot}_\alpha = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}$$

2D Rotations Re-cap

$$\begin{bmatrix} \cos 90^\circ & \sin 90^\circ \\ -\sin 90^\circ & \cos 90^\circ \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} a_2 & -a_1 \end{bmatrix}$$

[xkcd]

Special Case for 2D

- Same as for point masses, e.g., Euler step:

$$\mathbf{x}_{cm} = \mathbf{x}_{cm} + h\mathbf{v}_{cm}$$

$$\mathbf{r}_{cm} = \mathbf{r}_{cm} + h\mathbf{w}_{cm}$$

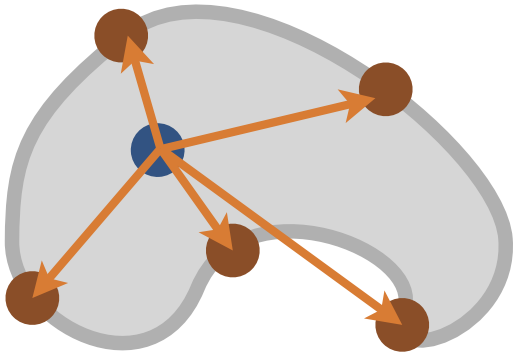
- Works only because we have 1 axis of rotation
- Later on: conserve angular momentum (not angular velocity)
- How to compute accelerations? What is the "mass" for angular motion?

Inertia Tensor in 2D

- “resistance to rotation” around a certain axis
- Always pre-compute for reference state
- In 2D, using body coordinates:

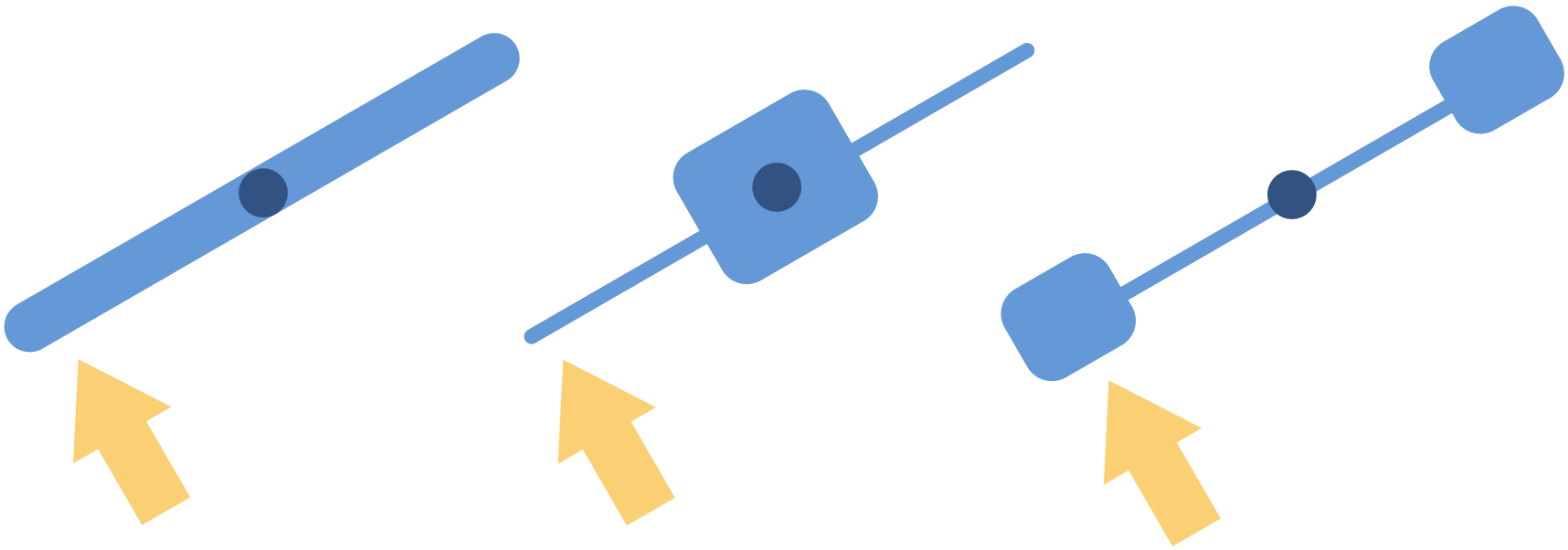
$$i = \sum_n m_n \mathbf{x}_n \cdot \mathbf{x}_n$$

- Example



Inertia Tensor in 2D

- What does *i* look like for these shapes? (Assume equal total mass, and same material density.)
- Or - which shape spins more easily when poked?



Another example

- Figure skating
  - Starts in normal pose
  - Rotation in plane
  - Moment of inertia is reduced by pulling in arms

Points vs. Rigid Bodies

- For particles:
  - Position ***x***
    Velocity ***v***
    Mass *M*
- Dynamics:

$$\mathbf{v}(t) = \frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t}$$

$$\mathbf{a}(t) = \frac{\mathrm{d}\mathbf{v}(t)}{\mathrm{d}t}$$

- For a rigid body:
  - Position ***x***
  - Orientation ***r***
  - Linear velocity ***v***
  - Angular velocity ***w***
  - Mass M
  - Inertia tensor ***I***
- Dynamics:
  - **?**

Rotational Dynamics

- Mass points are restricted to move perpendicular to their body space position
- Use cross product for projection
- Newton's 2nd law: $\frac{\mathrm{d}}{\mathrm{d}t}(m_i \mathbf{v}_i) = \mathbf{f}_i$
- Newton's 2nd law, restricted:

$$\mathbf{x}_i \times \frac{\mathrm{d}}{\mathrm{d}t}(m_i \mathbf{v}_i) = \mathbf{x}_i \times \mathbf{f}_i$$

Both sides are vectors parallel to actual axis of rotation

Rotational Dynamics

- From before

$$\mathbf{x}_i \times \frac{\mathrm{d}}{\mathrm{d}t}(m_i \mathbf{v}_i) = \mathbf{x}_i \times \mathbf{f}_i$$

- Move time derivative

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{x}_i \times m_i \mathbf{v}_i) = \mathbf{x}_i \times \mathbf{f}_i$$

- For whole body

$$\frac{\mathrm{d}}{\mathrm{d}t} \sum_i (\mathbf{x}_i \times m_i \mathbf{v}_i) = \sum_i \mathbf{x}_i \times \mathbf{f}_i$$

- Rename

angular momentum $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{L} = \mathbf{q}$ torque

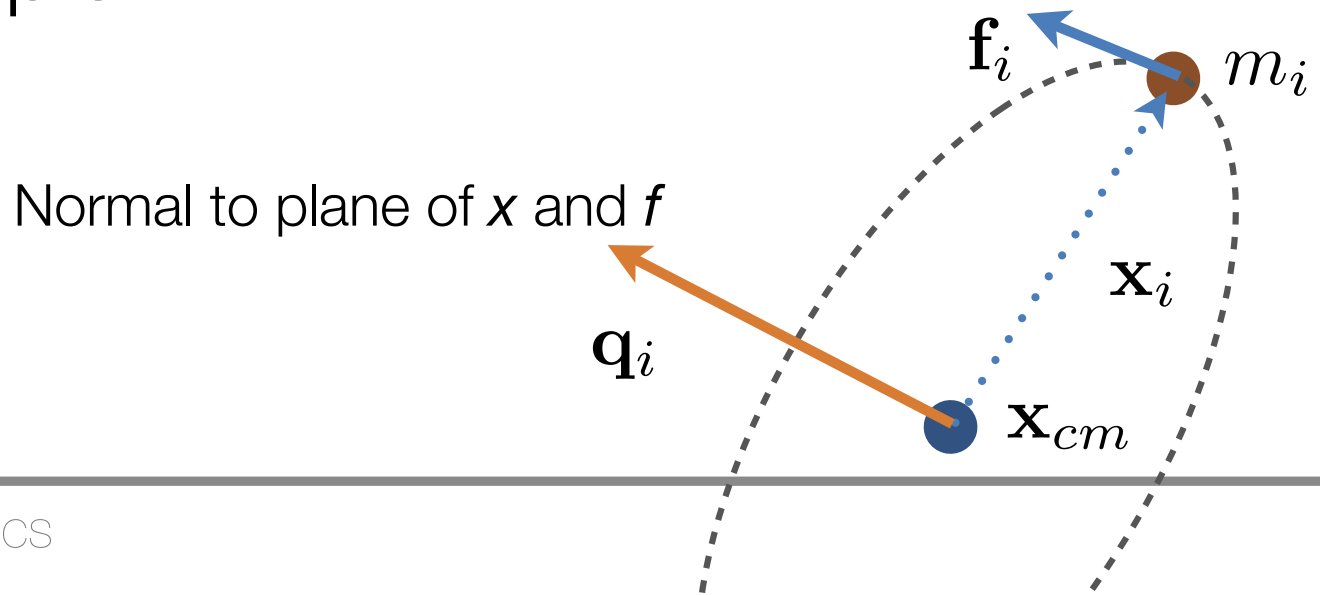
Both sides are still vectors
parallel to axis of rotation

Torque

- Equivalent of force for linear motion
- “Force producing rotation”

Torque

- Equivalent of force for linear motion
- *"Force producing rotation"*
- Torque of a mass point $\mathbf{q}_i = \mathbf{x}_i \times \mathbf{f}_i$
- Total torque of body $\mathbf{q} = \sum_i \mathbf{q}_i = \sum_i \mathbf{x}_i \times \mathbf{f}_i$
- Example:



Newton's 2nd Law for Rotations

- Angular momentum: $\mathbf{L} = \sum_i \mathbf{x}_i \times m_i \mathbf{v}_i = \mathbf{I}\mathbf{w}$
- Torque $\mathbf{q} = \sum_i \mathbf{x}_i \times \mathbf{f}_i$
- Angular version of Newton's 2nd law: $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{L} = \mathbf{q}$
- Compute the change of angular velocity over time, for 2D:

$$\mathbf{w}(t+h) = \mathbf{I}^{-1}\ \mathbf{L}(t+h)$$

Inertia tensor is constant over time, and just a single number, thus:

$$\mathbf{w}(t+h) = \mathbf{w}(t) + h\mathbf{q}/i$$

Points vs. Rigid Bodies

- For particles:
  - Position ***x***

    Velocity ***v***

    Mass *M*
- Dynamics:

$$\mathbf{v}(t) = \frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t}$$

$$\mathbf{a}(t) = \frac{\mathrm{d}\mathbf{v}(t)}{\mathrm{d}t}$$

- For a rigid body:
  - Position ***x***
  - Orientation ***r***
  - Linear velocity ***v***
  - Angular velocity ***w***
  - Mass M
  - Inertia tensor ***I***
  - Angular dynamics:

$$\mathbf{q}(t) = \sum_i \mathbf{x}_i \times \mathbf{f}_i$$

$$\mathbf{w}(t+h) = \mathbf{w}(h) + h\mathbf{q}/i$$

Simulation Algorithm in 2D

Pre-compute:

$$M \leftarrow \sum_i m_i$$

$$\mathbf{x}'_{cm} \leftarrow \sum_i \mathbf{x}'_i m_i / M$$

$$\mathbf{x}_i \leftarrow \mathbf{x}'_i - \mathbf{x}'_{cm}$$

$$i \leftarrow \sum_i m_i \mathbf{x}_i \cdot \mathbf{x}_i$$

Initialize:

$$\mathbf{x}_{cm}, \mathbf{v}_{cm}$$

$$\mathbf{r}, \mathbf{L}$$

$$\mathbf{w} \leftarrow \mathbf{L}/i$$

$$\mathbf{q} \leftarrow \sum_i \mathbf{x}_i \times \mathbf{f}_i$$

$$\mathbf{F} \leftarrow \sum_i \mathbf{f}_i$$

External forces

$$\mathbf{x}_{cm} \leftarrow \mathbf{x}_{cm} + h\mathbf{v}_{cm}$$

$$\mathbf{v}_{cm} \leftarrow \mathbf{v}_{cm} + h\mathbf{F}/M$$

$$\mathbf{r} \leftarrow \mathbf{r} + h\mathbf{w}$$

$$\mathbf{w} \leftarrow \mathbf{w} + h\mathbf{q}/i$$

Euler step

$$\mathbf{x}_i^{world} \leftarrow \mathbf{x}_{cm} + \mathrm{Rot}_r \mathbf{x}_i$$

$$\mathbf{v}_i^{world} \leftarrow \mathbf{v}_{cm} + \mathbf{w} \times \mathbf{x}_i$$

World position

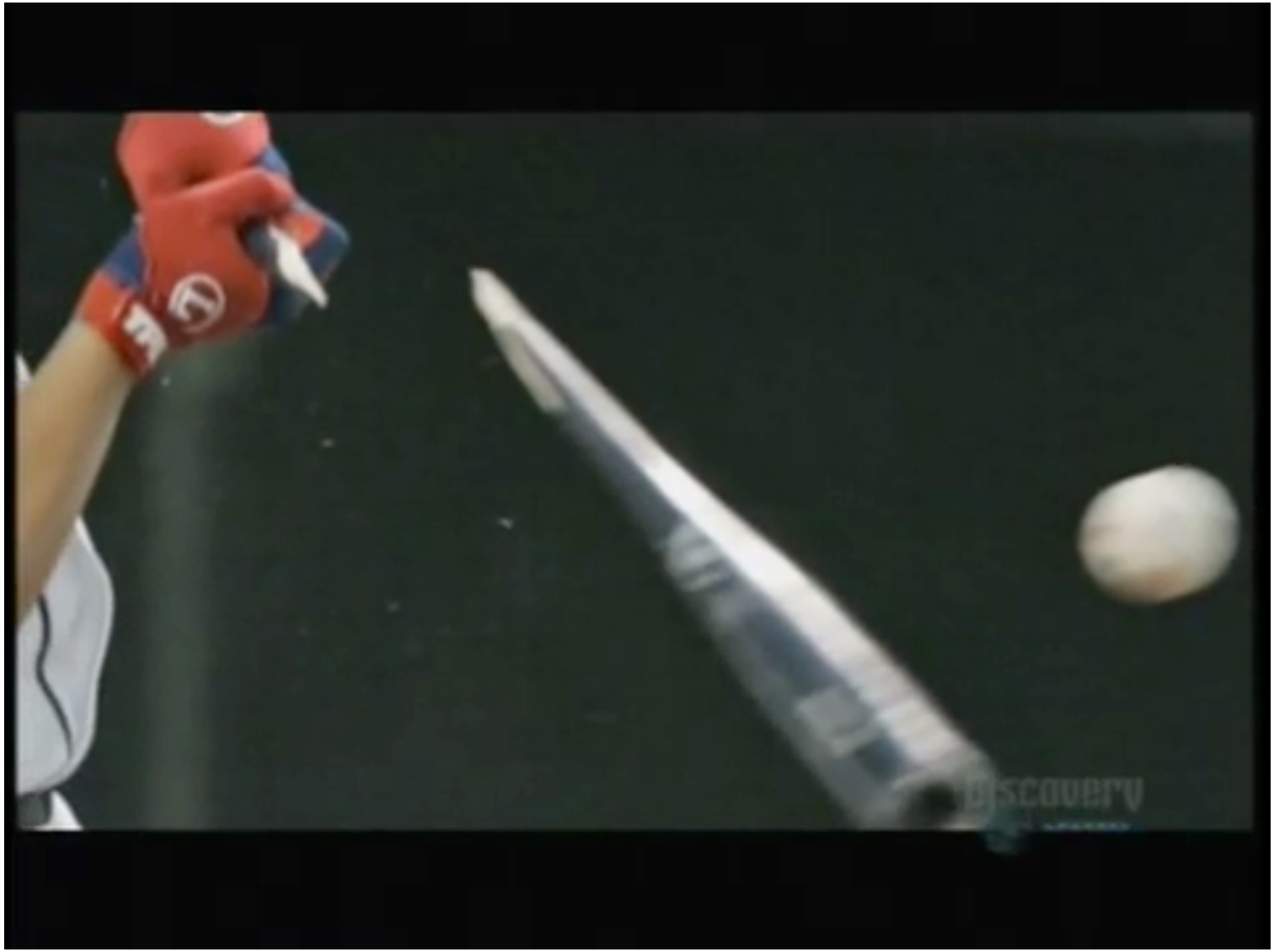
Additional Notation

t	time
h	time step size
$\mathbf{q}$	torque
$\mathbf{L}$	angular momentum
M	total mass
$\mathbf{I}$	inertia tensor
$\mathbf{v}$',$\mathbf{w}$'	post collision velocities
$\mathbf{x}$'	In simulation algorithm: original / un-centered vertices

Collisions

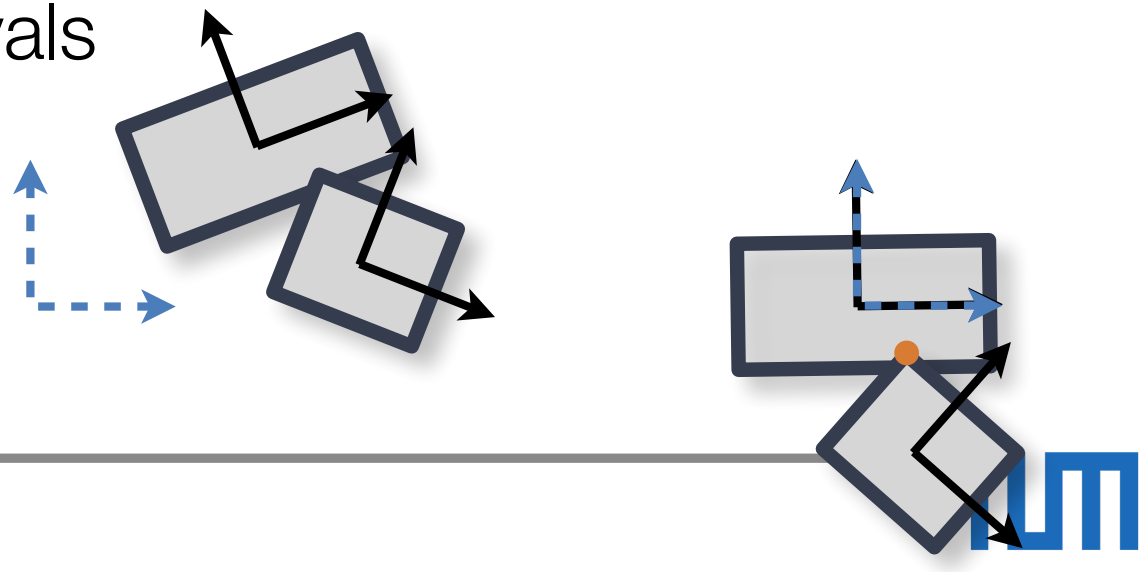
- What happens in reality?
  - Different materials have different elasticity and plasticity
  - That means:
    - Resistance to deformation
    - Tendency to return to original state
- Internal force resisting deformation causes a change of the body's motion
- For non-plastic materials, bodies will bounce back
- Usually happens in a fraction of a second...

Collisions



Collision Detection

- More accurate versions later!
- For now, very simple approach, e.g.:
  - Simulate boxes
  - Check corner points (or points on surface)
  - For target body, undo translation & rotation
  - Test points for intervals

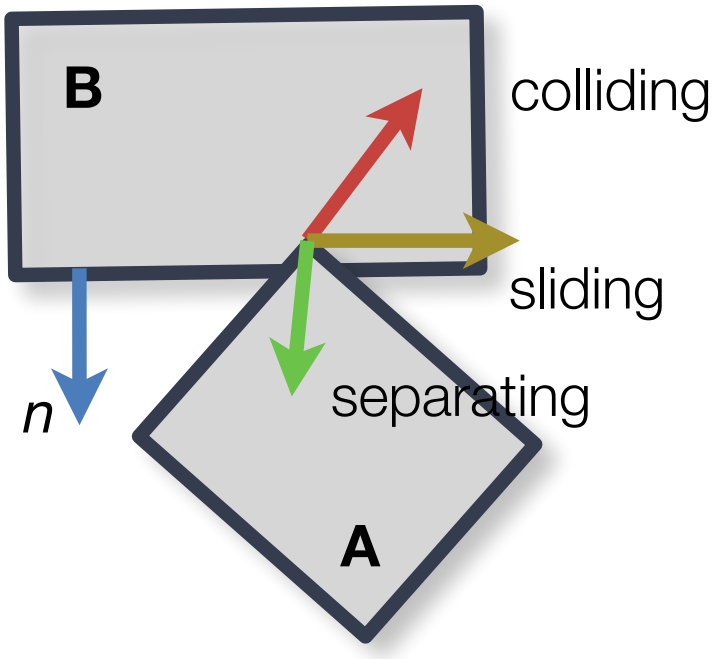


Classifying Contacts

- Velocity of x_i on rigid body: $\mathbf{v}_i = \mathbf{v}_{cm} + \mathbf{w} \times \mathbf{x}_i$
- Retrieve collision normal
- Compute relative velocity:

$$\mathbf{v}_{rel} = (\mathbf{v}_A - \mathbf{v}_B)$$

$$v_{rel} = \mathbf{n} \cdot (\mathbf{v}_A - \mathbf{v}_B)$$



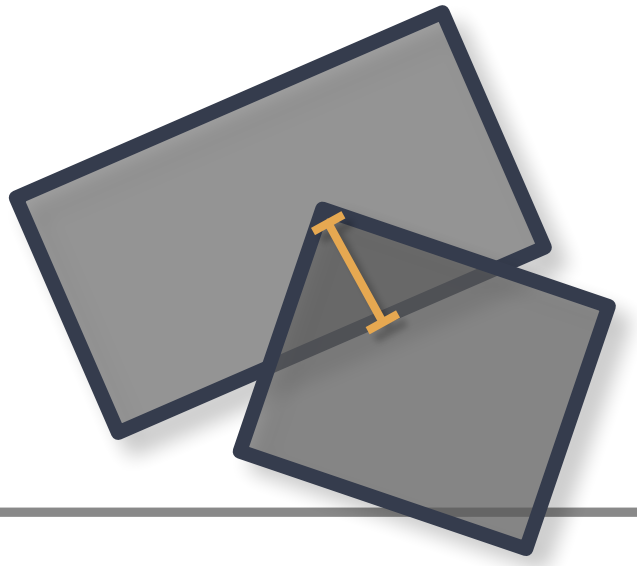
- 3 Cases:
  - Colliding contact $v_{rel} < 0$
  - Separating (easy!) $v_{rel} > 0$
  - Sliding contact $v_{rel} = 0$

Collision Response

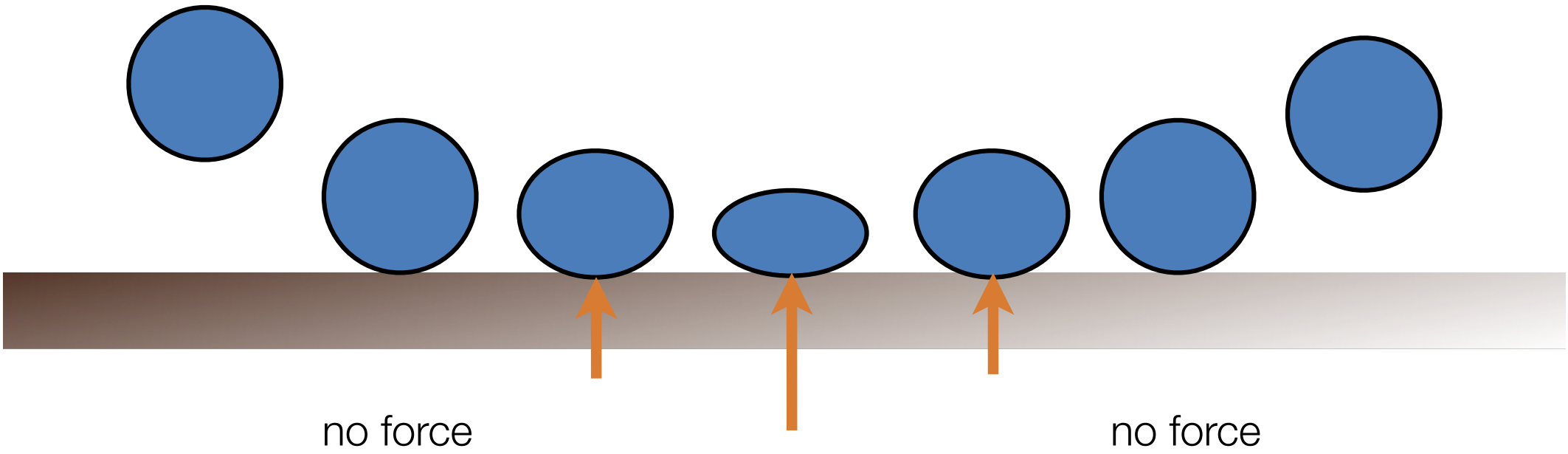
- Compute instantaneous effect of material deformation
- Separate handling of
  - linear motion
  - angular motion
- Make sure the object stop flying into each other...

Impulses

- We could try to model instantaneous deformation with forces, e.g.:
  - Measure penetration distance d
  - Apply force proportional to d
  - Hope that it keeps objects from moving into each other...
- Not a good idea:
  - No guarantees
  - Can cause large forces

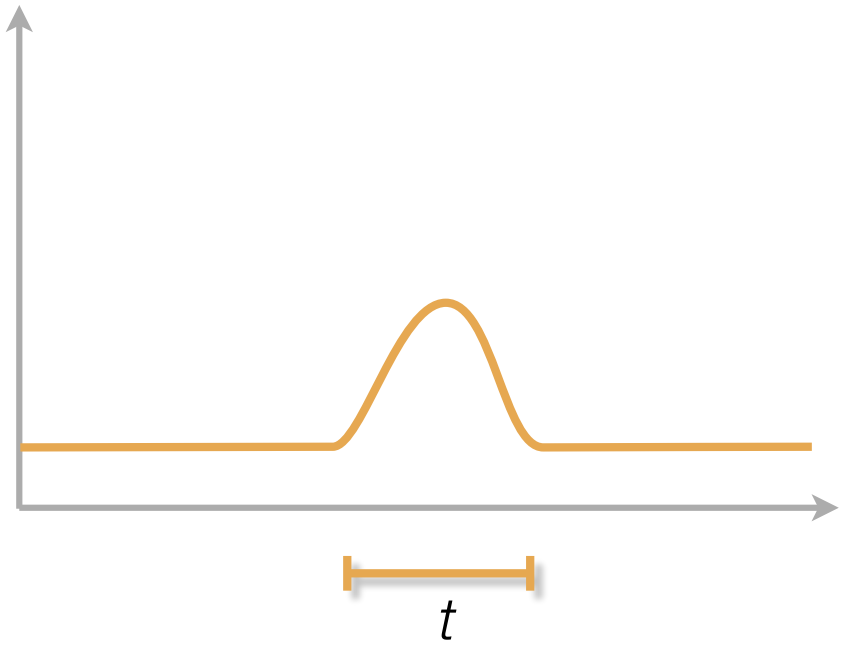


Impulses

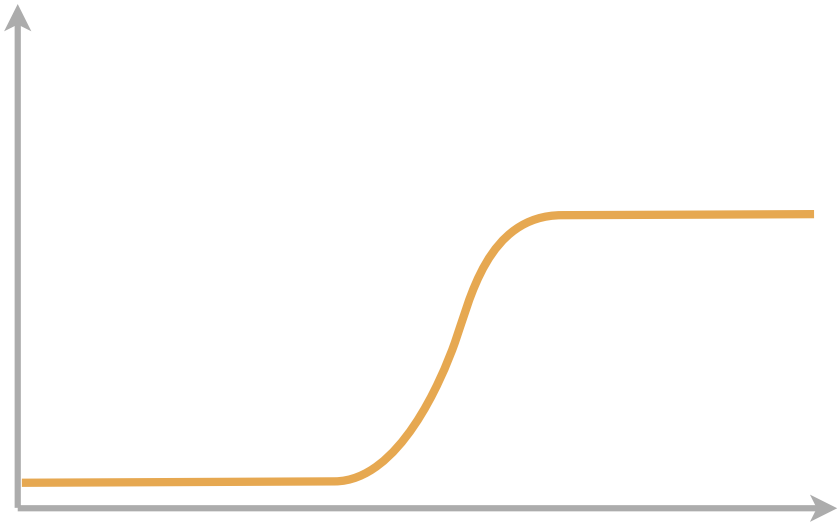


Soft Collision

- Force

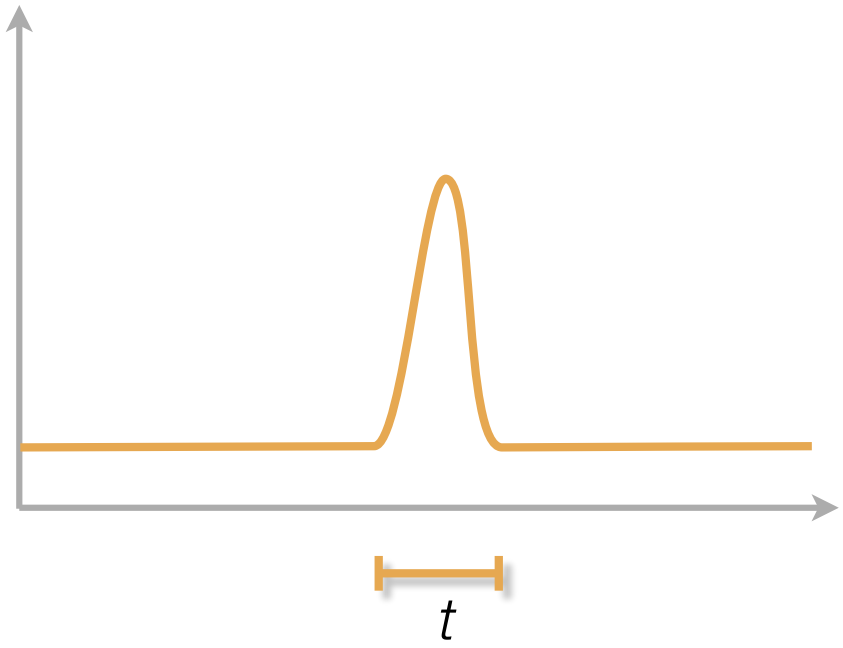


- Velocity

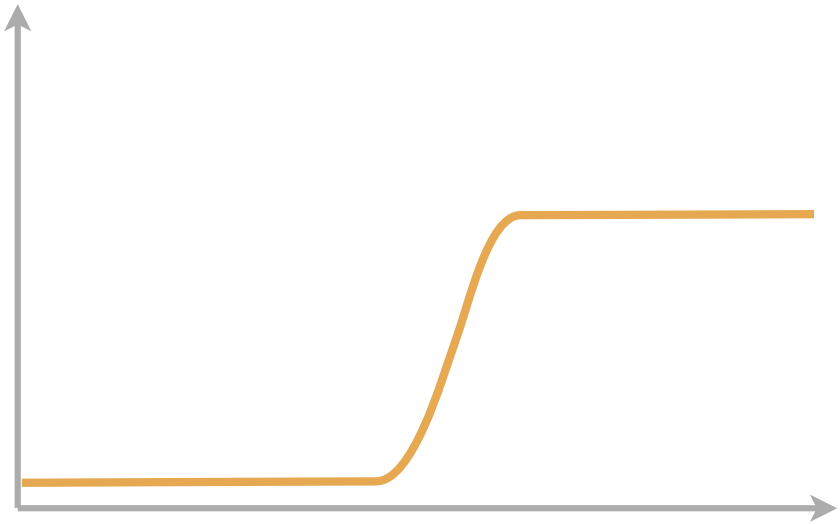


Harder Collision

- Force

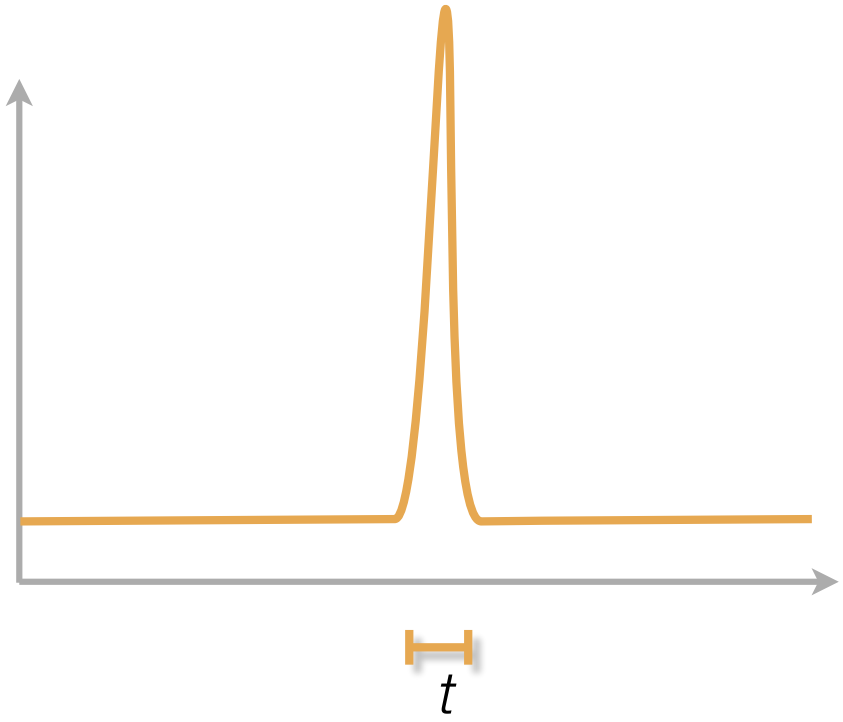


- Velocity

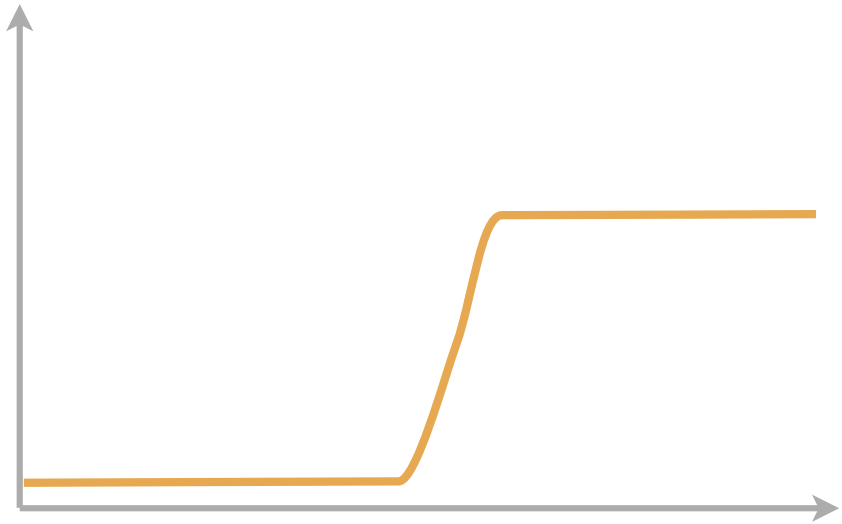


Very Hard Collision

- Force

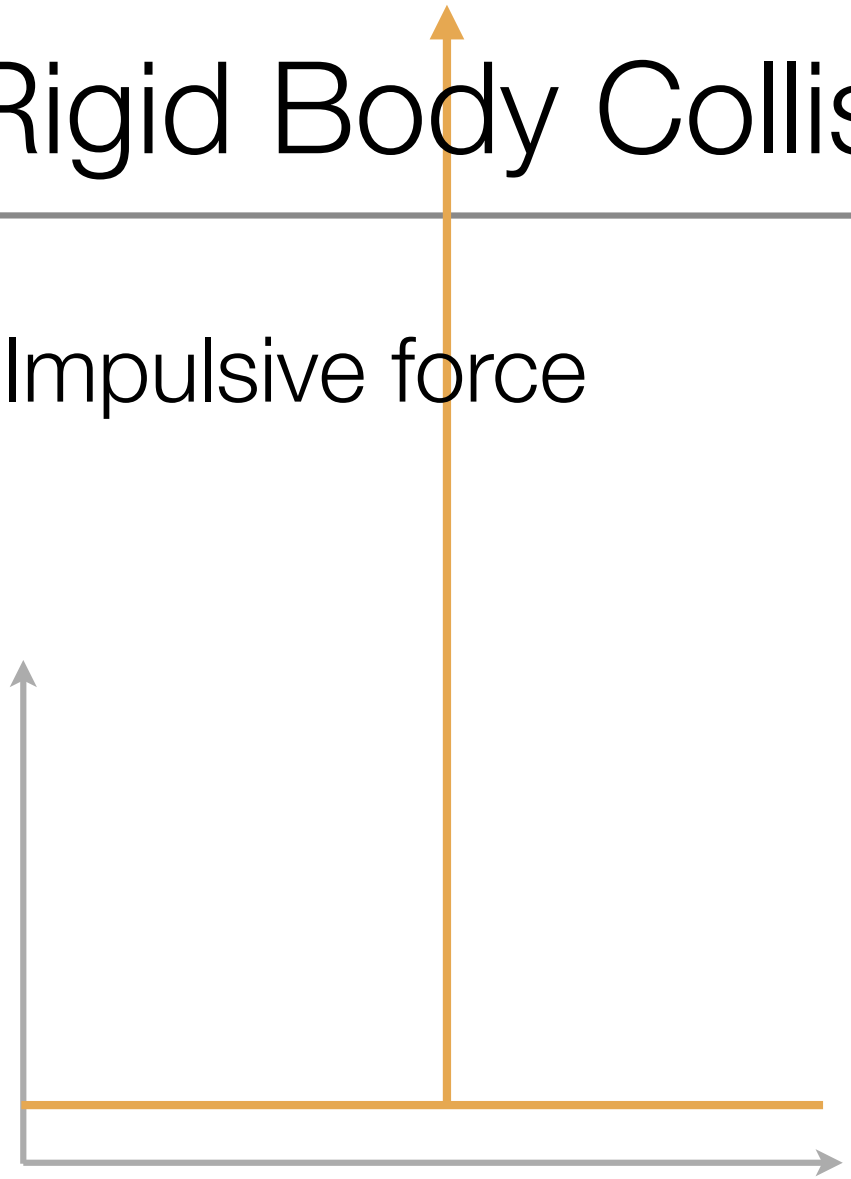


- Velocity



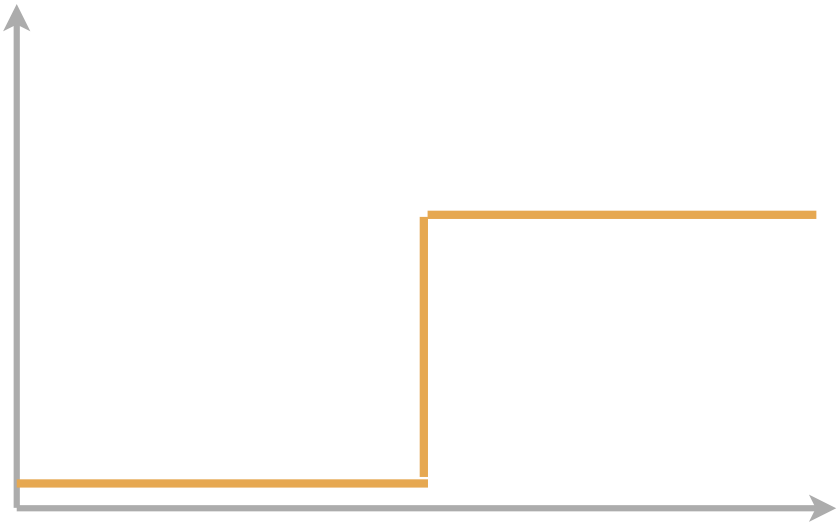
Rigid Body Collision

- Impulsive force



t=0 , infinite force

- Velocity



Impulses

- Fully rigid body would exert infinite elastic force over zero time interval
- Immediate velocity change; impulse has units like momentum (not a force!)
- To avoid singularity, apply impulses that change velocity directly
- Use: $\mathbf{J} = m\Delta\mathbf{v}$ (no time step!)
- Instead of: $\Delta\mathbf{v} = h\mathbf{F}/m$

“Bounciness”

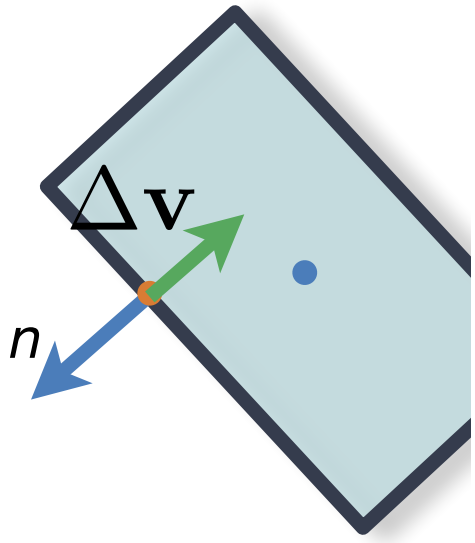
- We can’t resolve real deformation (i.e. plasticity effects)
- Instead simplified, empirical model: single coefficient *c* (coefficient of restitution)
- *c*=1 fully elastic, *c*=0 means plastic
- Loss of kinetic energy for c<1 (is transferred to permanently deformed state of the body)

Resolving Collisions for Linear Motion

- Let's look at point masses for a moment:
  - Assuming two points (really) collide...
  - Total momentum needs to be conserved

$$m_a \mathbf{v}_a + m_b \mathbf{v}_b = m_a \mathbf{v}'_a + m_b \mathbf{v}'_b$$

- Impulses act along contact normals
  - Velocity should reverse along collision surface
  - Tangential motion shouldn't be affected
  - Convention: impulse acts positively on body A
- Scalar equations: $\Delta \mathbf{v} = \Delta v \mathbf{n}$



Resolving Collisions for Linear Motion

- Impulse calculated with

$$J_{lin} = \frac{-(1+c)\mathbf{v}_{rel} \cdot \mathbf{n}}{\frac{1}{M_a} + \frac{1}{M_b}}$$

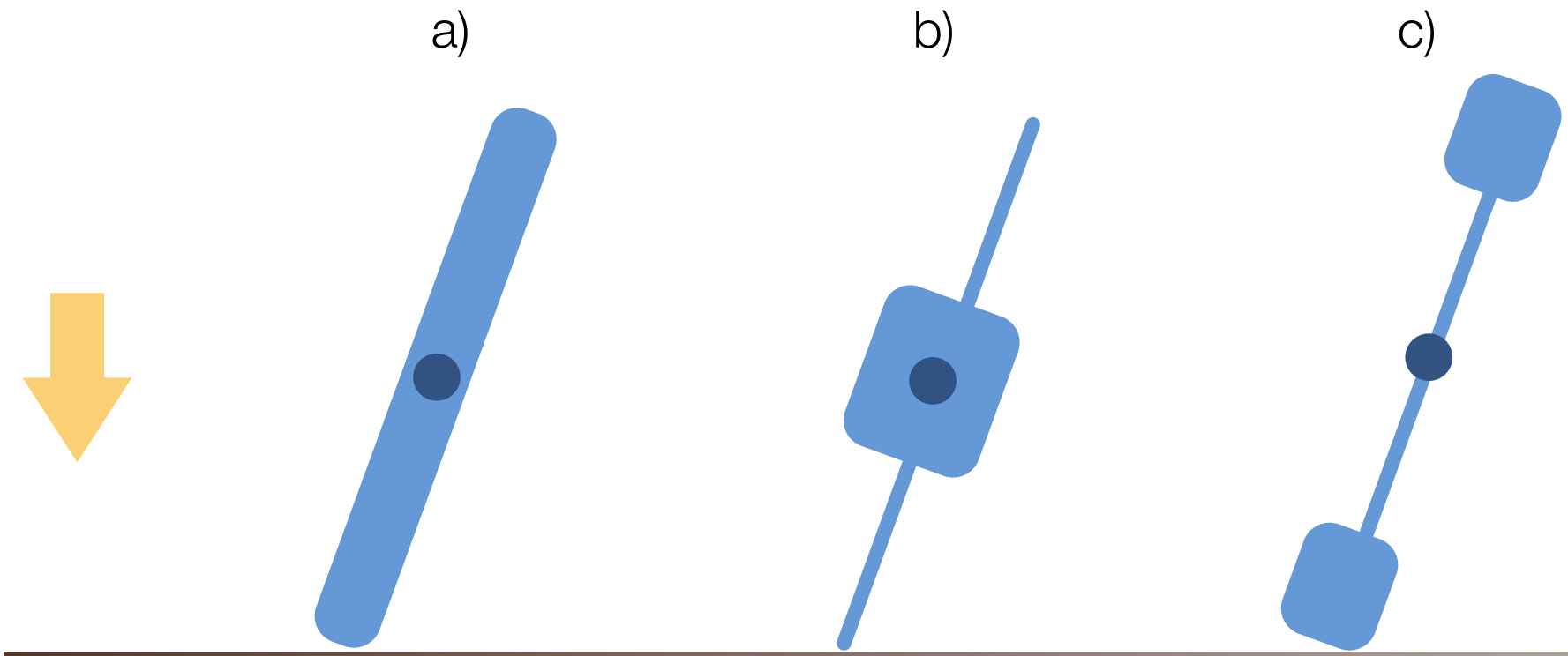
- Velocity update

$$\mathbf{v}'_a = \mathbf{v}_a + J_{lin}\mathbf{n}/M_a$$

$$\mathbf{v}'_b = \mathbf{v}_b - J_{lin}\mathbf{n}/M_b$$

Rotating Collisions

- Objects falling onto the ground
- What happens?



Include Angular Components

- Impulses still along contact normals, still scalar
- Impulse changes linear and angular velocity

$$J\mathbf{n} = M\Delta\mathbf{v}_{cm}$$
$$\mathbf{x}_{coll} \times J\mathbf{n} = \Delta\mathbf{L} = \mathbf{I}\Delta\mathbf{w}$$

- Angular velocity update

$$\mathbf{w}'_a = \mathbf{w}_a + (\mathbf{x}_a \times J\mathbf{n})/I_a$$
$$\mathbf{w}'_b = \mathbf{w}_b - (\mathbf{x}_b \times J\mathbf{n})/I_b$$

Resolving Collisions for Linear & Angular Motion

- Total impulse calculated with

$$J = \frac{-(1+c)\mathbf{v}_{rel} \cdot \mathbf{n}}{\frac{1}{M_a} + \frac{1}{M_b} + (\mathbf{x}_a \times \mathbf{n})^2 / I_a + (\mathbf{x}_b \times \mathbf{n})^2 / I_b}$$

- Velocity update

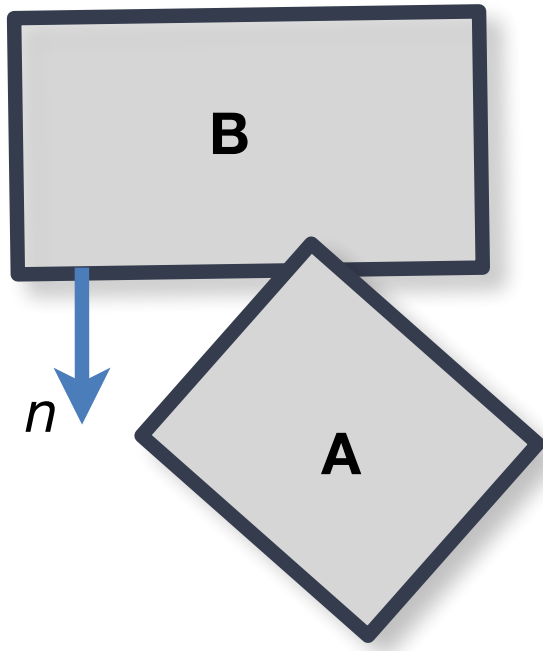
$$\mathbf{v}'_a = \mathbf{v}_a + J\mathbf{n}/M_a$$
$$\mathbf{v}'_b = \mathbf{v}_b - J\mathbf{n}/M_b$$

$$\mathbf{w}'_a = \mathbf{w}_a + (\mathbf{x}_a \times J\mathbf{n})/I_a$$
$$\mathbf{w}'_b = \mathbf{w}_b - (\mathbf{x}_b \times J\mathbf{n})/I_b$$

Conventions

- Collision normal points “into” body A
- Relative velocity is negative for collision…
- … thus J is always a positive scalar!

- Pay attention to these details for exercise!



So much for 2D...