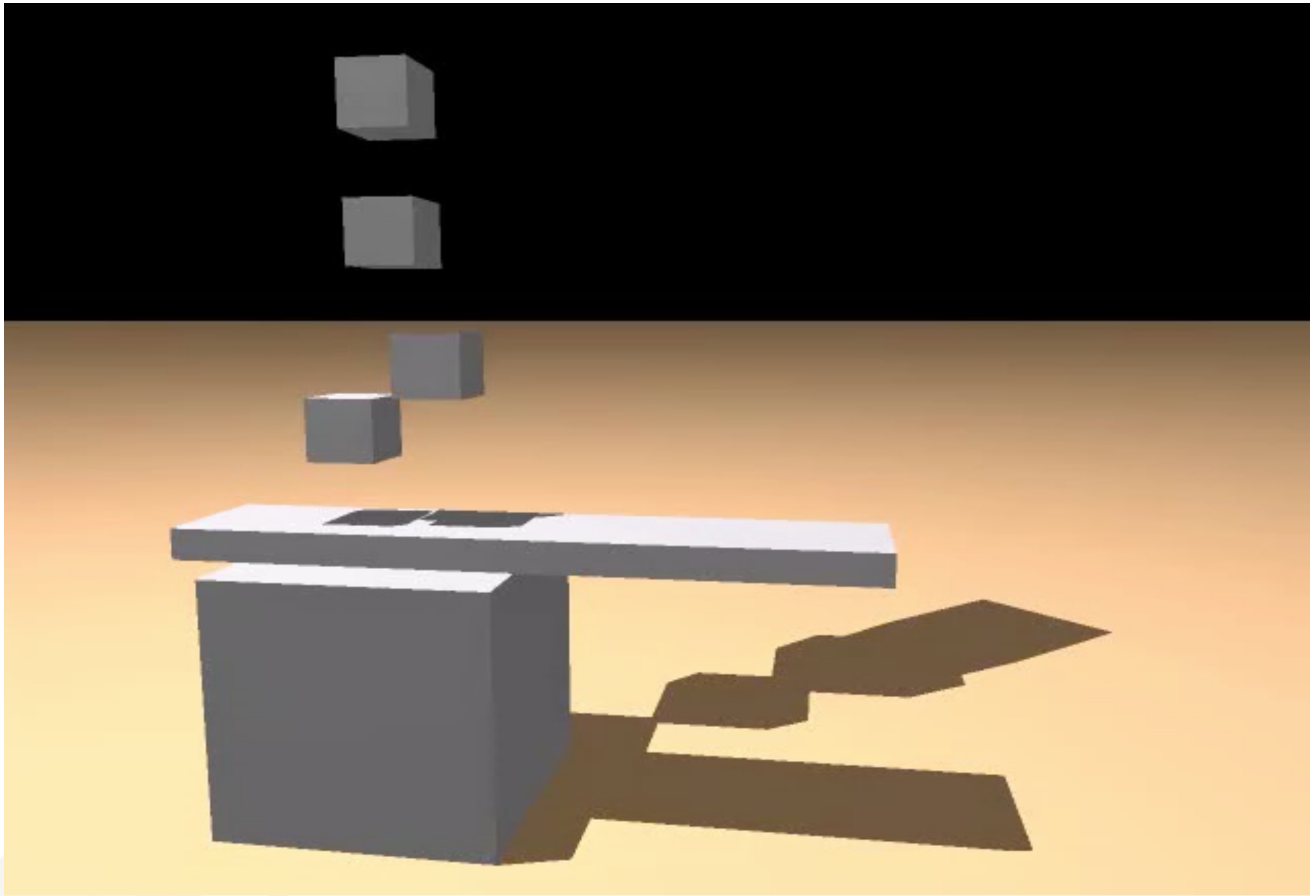
UNIVERSITÉ Concordia UNIVERSITY

Animation for Computer Games
COMP 477/6311

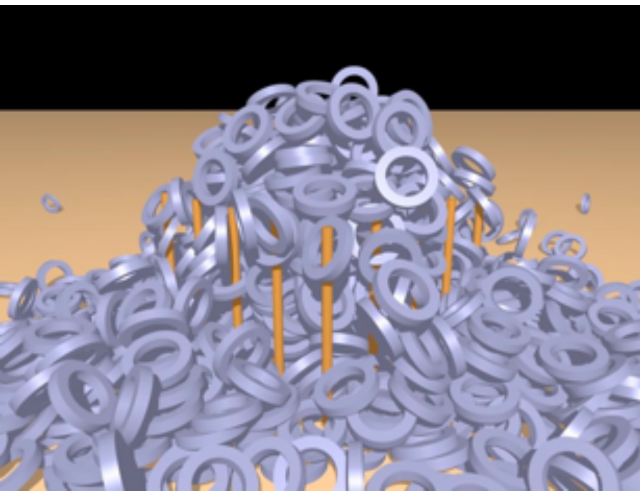
Prof. Tiberiu Popa

Rigid Body Simulation

Physics in Computer Graphics

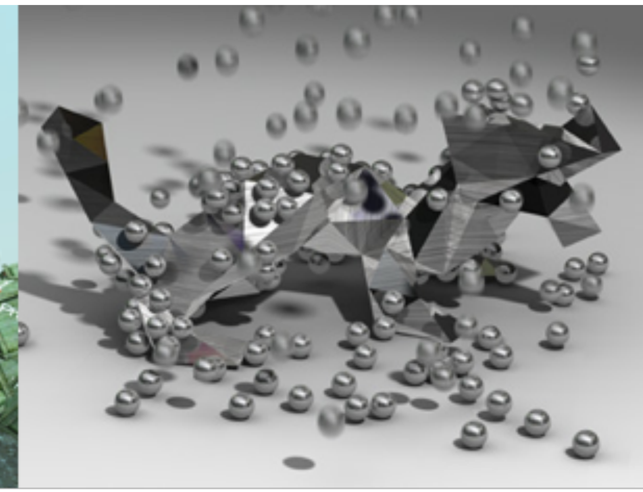


Rigid Body Simulation



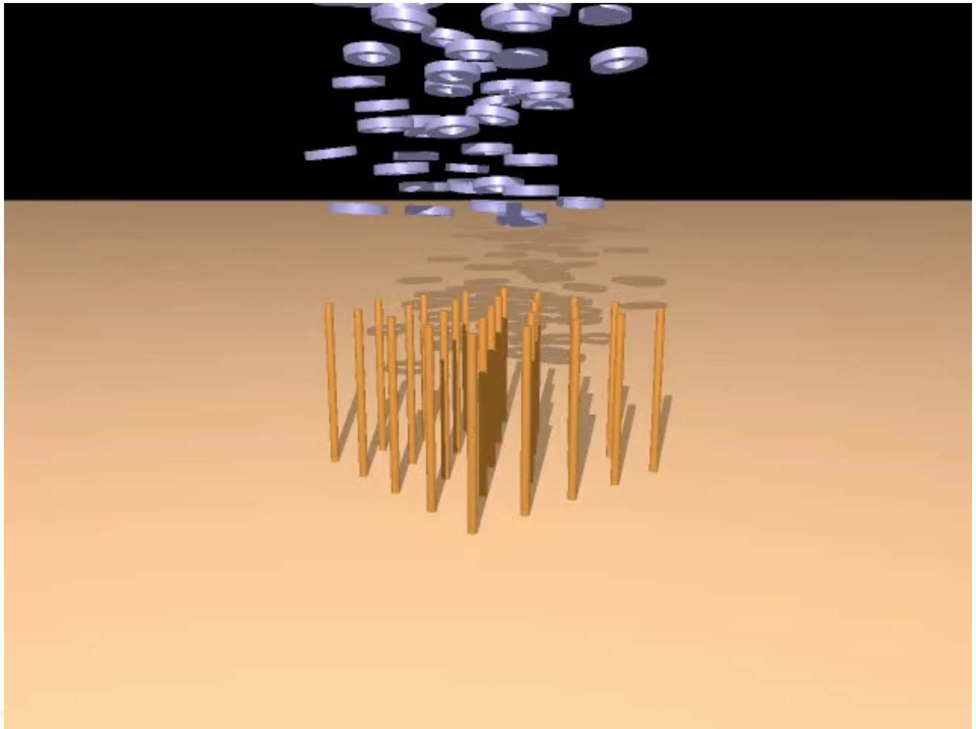
[Guendelman 03]

NVIDIA



[Thomaszewski 09]

Collision Handling



Larger Scale

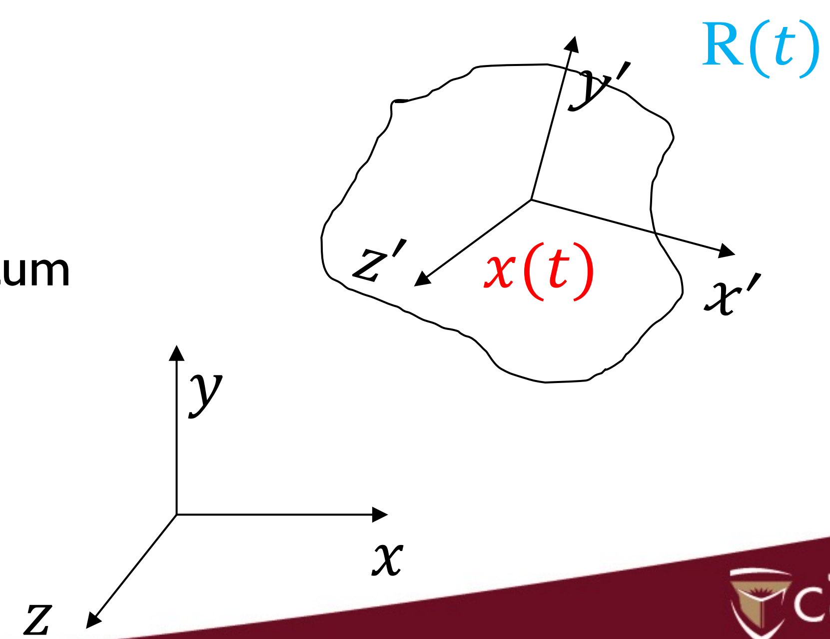
http://www.youtube.com/watch?v=J9HaT23b-xc

Real-Time

http://www.nvidia.com/

Outline

- Representation of a Rigid Body
  - Position and orientation
  - Center of mass
- Rigid Body Kinematics
  - Linear and angular velocity
- Rigid Body Dynamics
  - Force and torque
  - Linear and angular momentum
- Collision Handling



Tutorials

- Siggraph course notes
  http://www.cs.cmu.edu/~baraff/pbm/pbm.html

  David Baraff:
  - *An Introduction to Physically Based Modeling: Rigid Body Simulation I - Unconstrained Rigid Body Dynamics*
  - *An Introduction to Physically Based Modeling: Rigid Body Simulation II - Nonpenetration Constraints*

- The lecture slides follow these course notes and some illustrations are taken from there

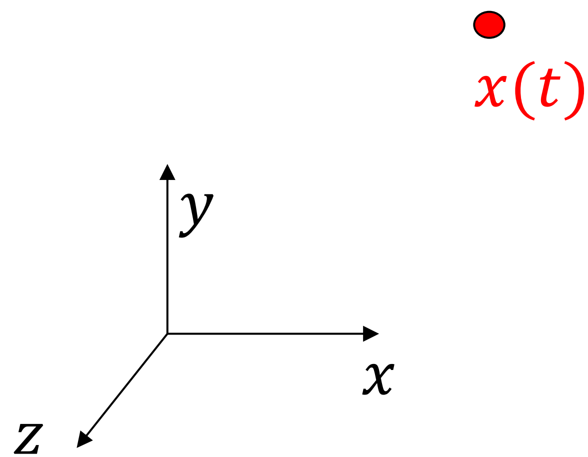
Linear Velocity

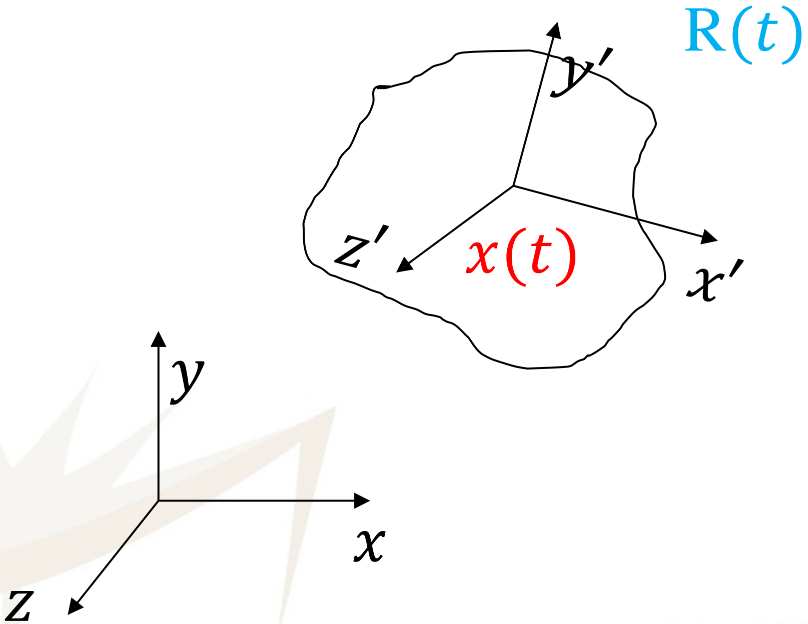
- Time integration: $x(t+\Delta t) = ?$

- How do position change over time?
  - Position: $x(t)$
  - Linear Velocity: $v(t) = \dot{x}(t) = \frac{d}{dt}x(t)$
  - Vector representation:
    - direction
    - magnitude

The general case

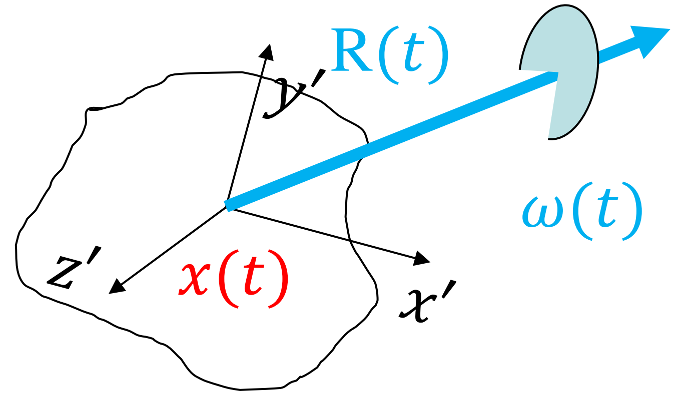
- Location of rigid body: translation and rotation
- Translation $x(t)$ (position) and rotation (orientation) $R(t)$





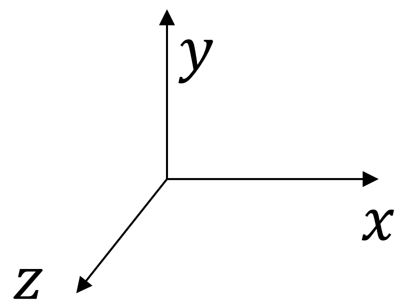
Spin

- Body can spin around axis
  - angular velocity $\omega(t)$
  - spin axis: direction of $\omega(t)$



- Linear velocity:

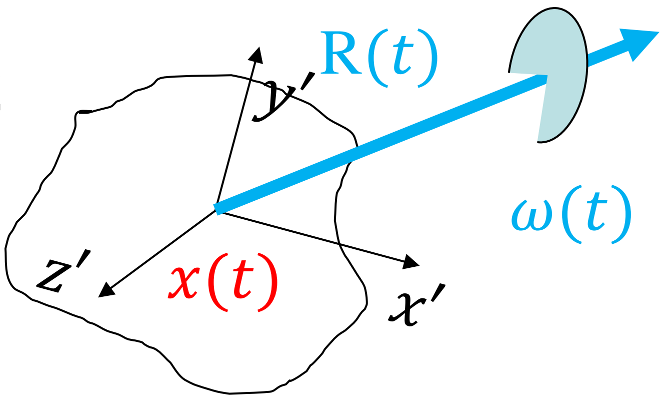
$$v(t) = \frac{d}{dt}x(t)$$

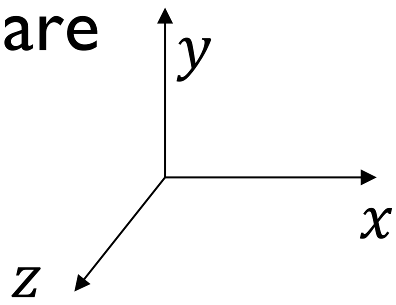


- Angular velocity:
  - $\omega(t)$ is a vector → corresponds to $v(t)$
  - $R(t)$ is a matrix → corresponds to $x(t)$
  - What is the relationship between $\dot{R}(t)$ and $\omega(t)$

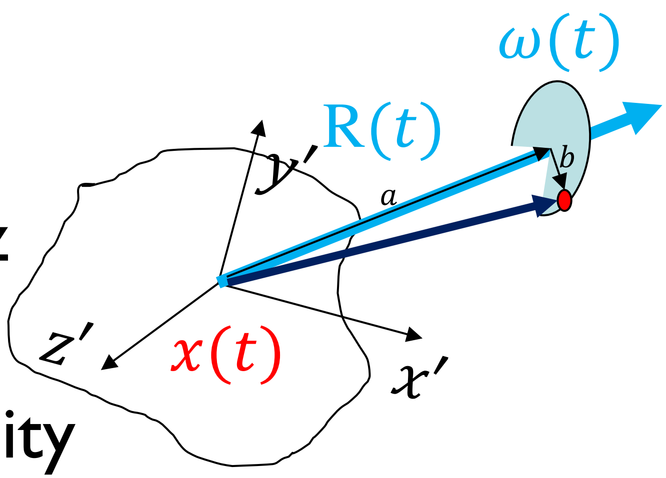
Rotation Matrix

- Rotation matrix $R(t) = [x', y', z']$
  - 3x3 matrix, each column describes direction of the transformed x, y, z axis
  - Columns of $\dot{R}(t)$ describe velocity with which axes are transformed





Rotation Matrix

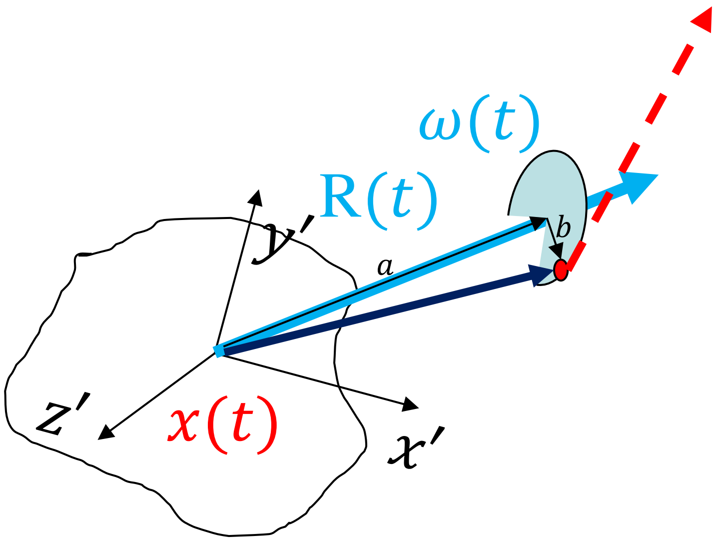


- Rotation matrix $R(t) = [x', y', z']$
  - 3x3 matrix, each column describes direction of the transformed x, y, z axis
  - Columns of $\dot{R}(t)$ describe velocity with which axes are transformed
  - Let $r(t)$ be a point on your object expressed as a vector from center of mass
  - Unaffected by linear velocity
  - $r(t)$ traces a circle around the object

Vector Rate of Change

$$\dot{r}(t) = \omega(t) \times b$$

$$r(t) = a + b$$



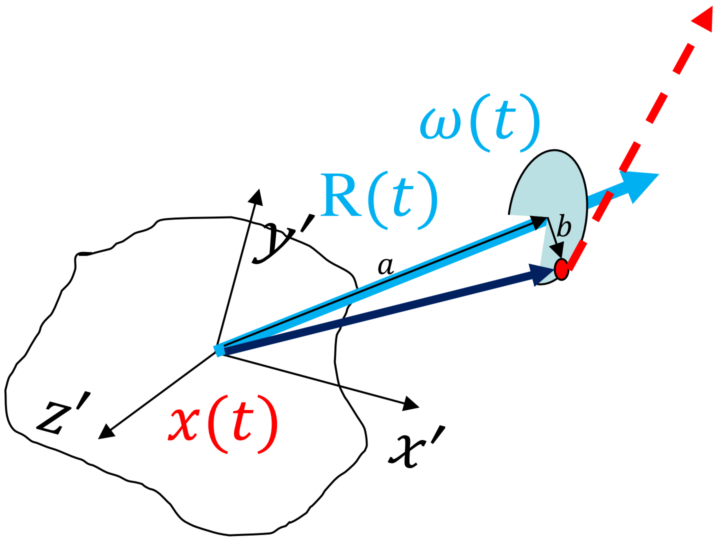
- Change of $r(t)$ perpendicular to b and $\omega(t)$

Vector Rate of Change

$$\dot{r}(t) = \omega(t) \times b$$

$$r(t) = a + b$$

Change of $r(t)$ perpendicular to b and $\omega(t)$



$$\dot{r}(t) = \omega(t) \times b$$

$$\dot{r}(t) = \omega(t) \times b + \omega(t) \times a$$

$$\dot{r}(t) = \omega(t) \times (a + b)$$

$$\dot{r}(t) = \omega(t) \times r(t)$$

Put it Together

- Given a point on the objects expressed as a vector from center of mass to the point $r(t)$

$$\dot{r}(t) = \omega(t) \times r(t)$$

- If $\begin{pmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{pmatrix}$ is the first column of $R(t)$, its rate of change is $\omega(t) \times \begin{pmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{pmatrix}$

- For all columns we can write:

$$\dot{R} = \left(\omega(t) \times \begin{pmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{pmatrix} \quad \omega(t) \times \begin{pmatrix} r_{yx} \\ r_{yy} \\ r_{yz} \end{pmatrix} \quad \omega(t) \times \begin{pmatrix} r_{zx} \\ r_{zy} \\ r_{zz} \end{pmatrix} \right)$$

Simplifications

- Cross product:

$$a \times b = \begin{pmatrix} a_y b_z - b_y a_z \\ -a_x b_z + b_x a_z \\ a_x b_y - b_x a_y \end{pmatrix}$$

$$= \boxed{\begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix}} \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = a^{\boxed{*}} b$$

- Define $a*$ operator $\begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix}$

- Why? It works between vectors as well as vector matrix operator

Angular Velocity

- Using $*$ notation:

$$\dot{R}(t) = \left(\omega(t)^* \begin{pmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{pmatrix} \quad \omega(t)^* \begin{pmatrix} r_{yx} \\ r_{yy} \\ r_{yz} \end{pmatrix} \quad \omega(t)^* \begin{pmatrix} r_{zx} \\ r_{zy} \\ r_{zz} \end{pmatrix} \right)$$

$$\dot{R}(t) = \omega(t)^* \left(\begin{pmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{pmatrix} \quad \begin{pmatrix} r_{yx} \\ r_{yy} \\ r_{yz} \end{pmatrix} \quad \begin{pmatrix} r_{zx} \\ r_{zy} \\ r_{zz} \end{pmatrix} \right)$$

$$\dot{R}(t) = \omega(t)^* R(t)$$

$$\dot{r}(t) = \omega(t) \times r(t)$$

Total Velocity

- (Constant) location of particle i in body space: r_{0_i}

$$r_i(t) = R(t)r_{0_i} + x(t)$$

$$\begin{aligned}
\dot{r}_i &= \omega(t) * R(t)r_{0_i} + v(t) \\
&= \omega(t) * (R(t)r_{0_i} + x(t) - x(t)) + v(t) \\
&= \omega(t) * (r_i(t) - x(t)) + v(t) \\
&= \omega(t) \times (r_i(t) - x(t)) + v(t)
\end{aligned}$$

Outline

- Representation of a Rigid Body
  - Position and orientation
  - Center of mass
- Rigid Body Kinematics
  - Linear and angular velocity
- Rigid Body Dynamics
  - Force and torque
  - Linear and angular momentum
- Collision Handling

Dynamics - May the force be with you!

- Relationship between force and velocity:
  - Physics tools:
  - 1) Newton second law of motion:

$$F = ma$$

  - Not sufficient here…
  - Why?
  - Acceleration changes velocity
  - When we have 2 types of velocities → which type is changed and in what amount?

Dynamics - May the force be with you!

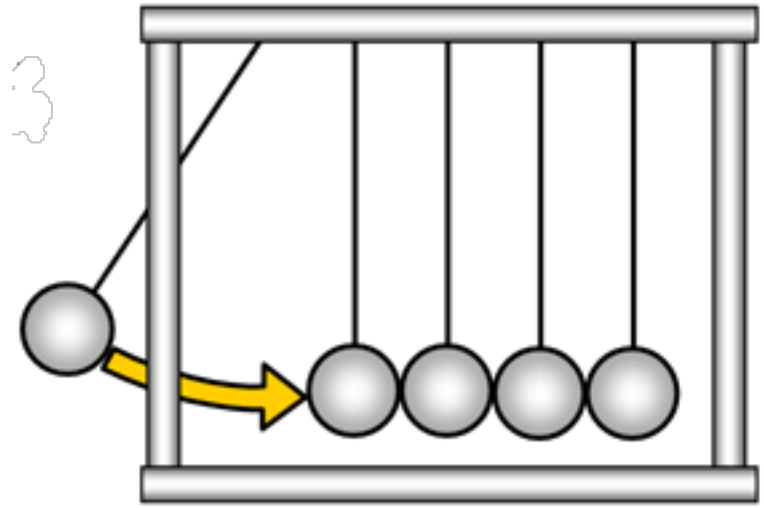
- Newton first law of motion
- Conservation of momentum
  - Linear momentum:

$$P(t) = \boxed{Mv(t)}$$

  - Angular momentum:

$$L(t) = \boxed{I(t)\omega(t)}$$

Conservation of momentum



Dynamics - May the force be with you!

- Newton first law of motion
- Conservation of momentum
  - Linear momentum:

$$P(t) = \boxed{Mv(t)}$$

  - Angular momentum:

$$L(t) = \boxed{I(t)\omega(t)}$$

Conservation of momentum

Conservation of momentum

https://www.youtube.com/watch?v=FmnkQ2ytlO8

Inertia Tensor

- 3x3 matrix describes how mass in a body is distributed relative to CM.
- Depends on orientation but not on translation of body

$$I(t) = \sum \begin{pmatrix} m_i(r'^2_{iy} + r'^2_{iz}) & -m_i r'_{ix} r'_{iy} & -m_i r'_{ix} r'_{iz} \\ -m_i r'_{ix} r'_{iy} & m_i(r'^2_{ix} + r'^2_{iz}) & -m_i r'_{iz} r'_{iy} \\ -m_i r'_{ix} r'_{iz} & -m_i r'_{iz} r'_{iy} & m_i(r'^2_{iy} + r'^2_{ix}) \end{pmatrix}$$

$${r_i}' = r_i(t) - x(t)$$

Body Space Inertia Tensor

- The inertia tensor (and inverse) in the original body space can be precomputed

$$I(t) = \sum m_i(({r_i}'^T {r_i}')1 - {r_i}'{r_i}'^T)$$

$$I_{body} = \sum m_i(({r_{0i}}^T r_{0i})1 - r_{0i}{r_{0i}}^T)$$

$$I(t) = R(t) I_{body} R(t)^T$$

Dynamics - May the force be with you!

- Linear momentum:

$$P(t) = Mv(t)$$

- Newton Second law:

$$\frac{dP}{dt} = F$$

Dynamics - May the force be with you!

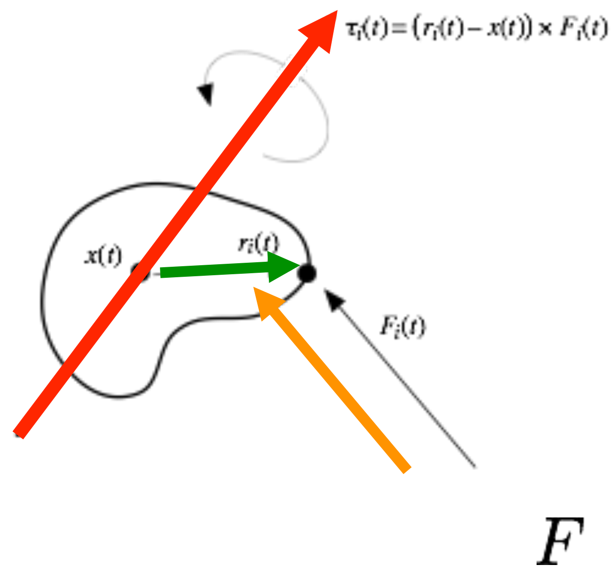
- Angular momentum:

$$L(t) = I(t)\omega(t)$$

- Newton second law:
  - $\frac{dL}{dt} = \tau$

$$\boxed{\tau_i(t)} = \boxed{(r_i(t) - x(t))} \times \boxed{F_i(t)}$$

$$\tau$$



$$\tau(t) = \sum \tau_i(t)$$

Putting all together

Angular momentum: $\boxed{L = I\omega}$

Torque: $\tau = \sum (r_i - x_i) \times F_i$

Angular version of Newton's 2nd law: $\boxed{\dot{L} = \tau}$

Explicit Newton:

$$L \leftarrow L + \boxed{\Delta t \cdot \tau}$$

$$\boxed{\omega \leftarrow I^{-1} L}$$

Algorithmic chain (Euler)

$$r_i(t) = R(t)r_{0_i} + x(t)$$

- Translation component:

$$x(t+\Delta t) \approx x(t) + \Delta t \cdot \dot{x}(t) \rightarrow \dot{x}(t) = v(t) \rightarrow v(t) \approx v(t-\Delta t) + \Delta t \cdot \dot{v}(t-\Delta t)$$

$$\dot{v}(t-\Delta t) = a(t-\Delta t) \rightarrow F(t-\Delta t) = m \cdot a(t-\Delta t)$$

- Rotation component

$$R(t+\Delta t) \approx R(t) + \Delta t \cdot \dot{R}(t) \rightarrow \dot{R}(t) = \omega(t) * R(t) \rightarrow \omega(t) = I(t)^{-1} L(t)$$

$$L(t) \approx L(t-\Delta t) + \Delta t \cdot \dot{L}(t-\Delta t) \rightarrow \dot{L} = \tau = \sum (r_i - x_i) \times F_i$$

$$I(t) = R(t) I_{body} R(t)^T$$

Euler Time Integration Algorithm for rigid bodies

Input: Initial state (pos. and vel.), external forces at every time frame
Output: position at each time frame

1. Compute object specific parameters that do not change over time

   Center of mass & inertia tensor

   $$x = \frac{1}{M}\sum_i m_i r_i \qquad I_{body}^{-1} \leftarrow \left(\sum m_i((r_{0_i}^T r_{0_i})1 - r_{0_i} r_{0_i}^T\right)^{-1}$$

1. Given current state of the system, compute the state at the next time interval
   1. Compute external forces: $F = \sum F_i$ (no internal F)
   2. Estimate linear velocity: $v(t+\Delta t) \approx v(t) + \Delta t \cdot \frac{F}{M}$
   3. Estimate translational motion:

      $$x(t+\Delta t) \approx x(t) + \Delta t \cdot v(t)$$

Euler Time Integration Algorithm for rigid bodies

4. Compute torque: $\tau = \sum (r_i - x_i) \times F_i$
5. Estimate angular momentum: $L(t+\Delta t) \approx L(t) + \Delta t \cdot \tau$
6. Compute the inverse of the inertial tensor: $I^{-1} \leftarrow R I_{body}^{-1} R^T$
7. Compute angular velocity: $\omega(t) = I(t)^{-1} L(t)$
8. Estimate new rotation: $R(t+\Delta t) \approx R(t) + \Delta t \cdot \omega(t) * R(t)$
9. Update position for each particle: $r_i = R r_{0i} + x$