Lecture 08

Physics-based Simulation and Articulated Rigid Bodies

Libin Liu

School of Intelligence Science and Technology
Peking University

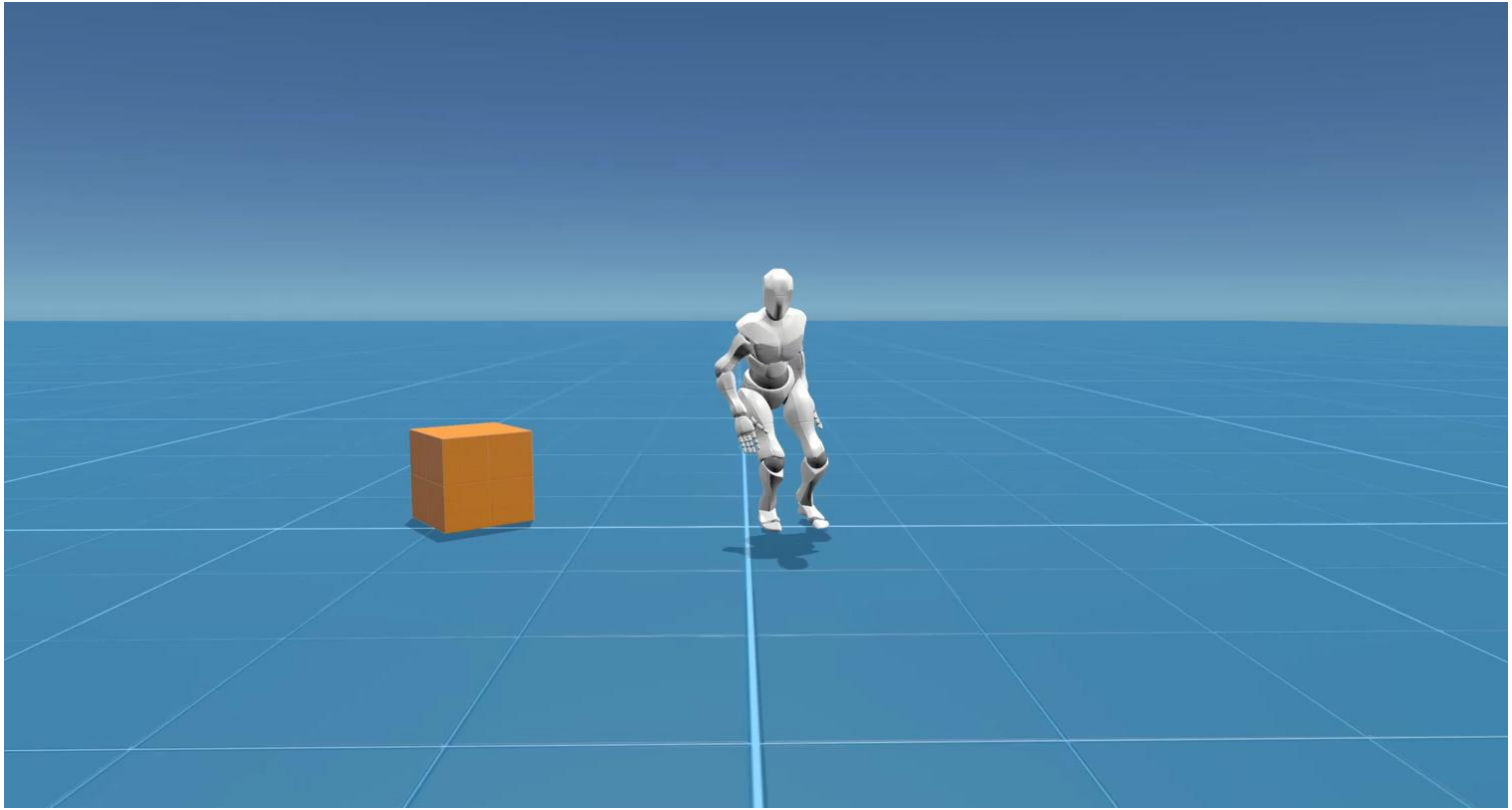
GAMES105 课程交流

VCL @ PKU

Problems of Kinematic Methods

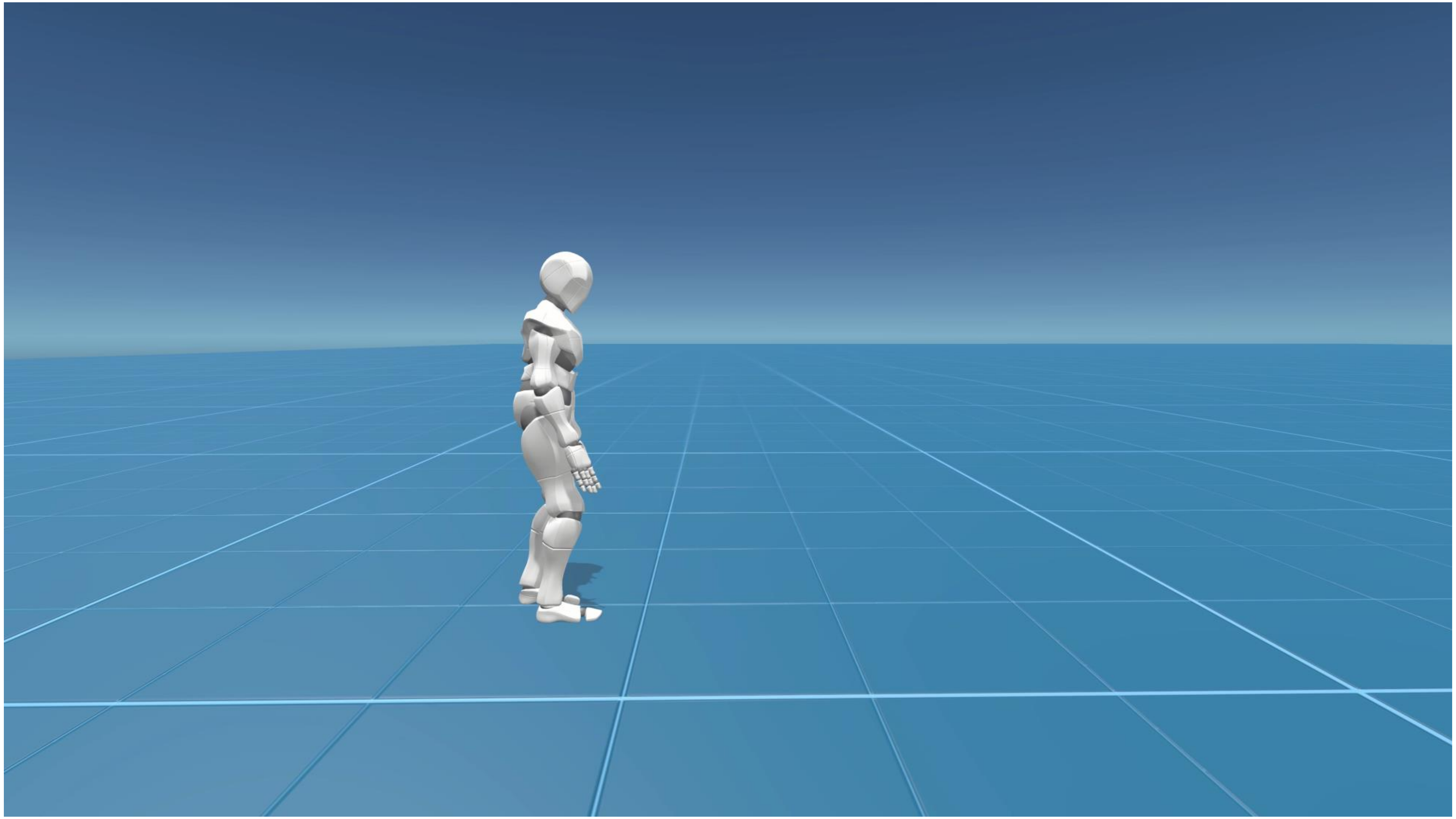
- Interaction with the environment

Physics-based Character Animation



[ControlVAE – Yao et al. 2022]

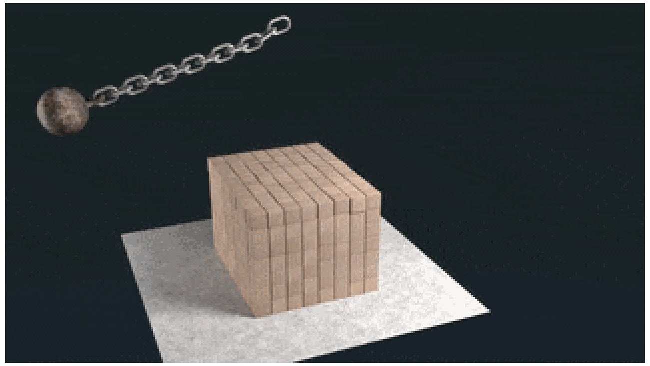
Physics-based Character Animation

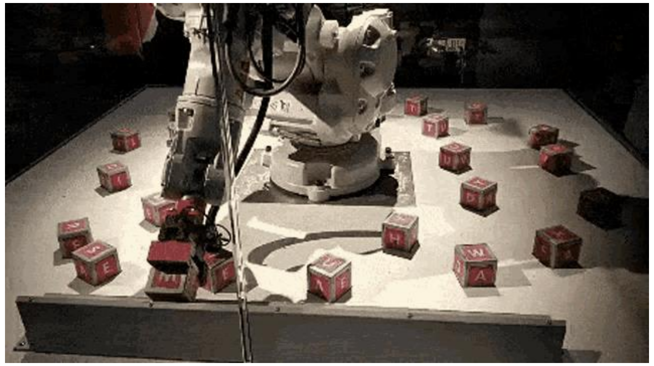


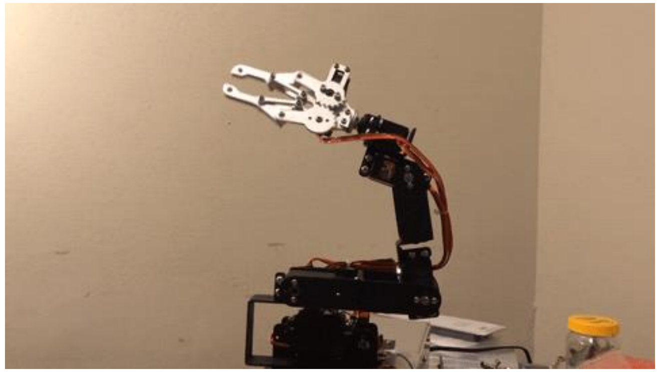
[ControlVAE – Yao et al. 2022]

Outline

- Simulation Basis
  - Numerical Integration: Euler methods
- Equations of Rigid Bodies
  - Rigid Body Kinematics
  - Newton-Euler equations
- Articulated Rigid Bodies
  - Joints and constraints
- Contact Models
  - Penalty-based contact
  - Constraint-based contact

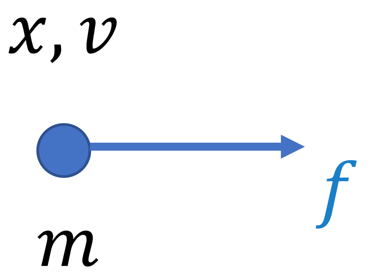






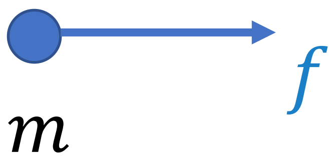
https://www.cs.cmu.edu/~baraff/sigcourse/

Dynamics of a Particle



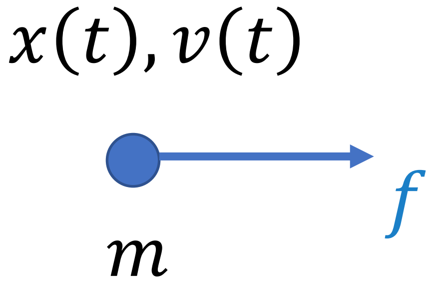
Dynamics of a Particle

$$x(t=0)$$
$$v(t=0)$$



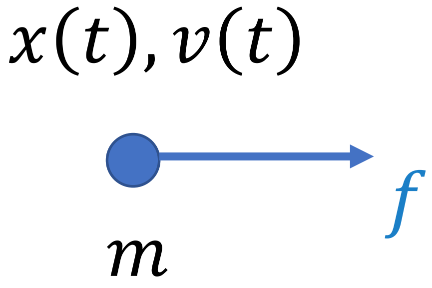
$$x(t=10)=?$$

Dynamics of a Particle



$$f = ma$$

Dynamics of a Particle

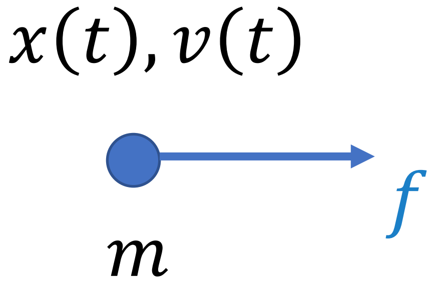


$$f = ma$$

$$a = \dot{v}$$

$$v = \dot{x}$$

Dynamics of a Particle



$$f = ma$$

$$a = \dot{v}$$

$$v = \dot{x}$$

$$a = f/m$$

$$v = v_0 + \int_{t_0}^{t} a dt$$

$$x = x_0 + \int_{t_0}^{t} v dt$$

Dynamics of a Particle

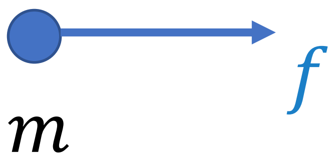
$$f = ma \qquad\qquad a = f/m$$

$$a = \dot{v} \qquad\qquad v = v_0 + at$$

$$v = \dot{x} \qquad\qquad x = x_0 + v_0 t + \frac{1}{2}at^2$$

Dynamics of a Particle

$$x(t), v(t)$$

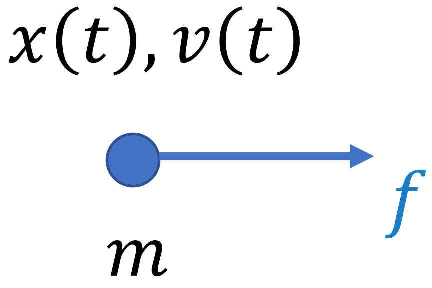


$$x(t = 10) = x_0 + 10v_0 + 50\frac{f}{m}$$

$$f = ma$$
$$a = \dot{v}$$
$$v = \dot{x}$$

$$a = f/m$$
$$v = v_0 + at$$
$$x = x_0 + v_0 t + \frac{1}{2}at^2$$

Dynamics of a Particle



$x(t = 10) = ?$

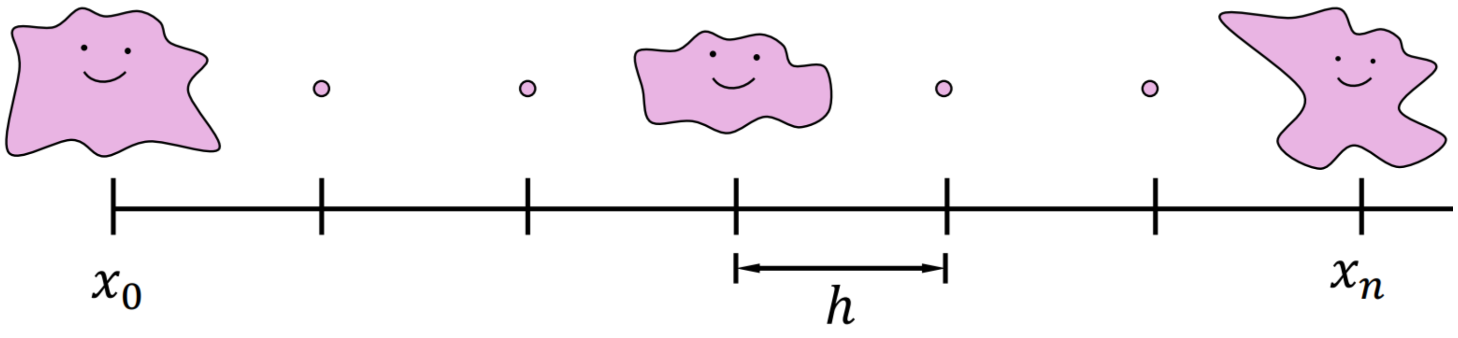
$$f = ma$$
$$a = \dot{v}$$
$$v = \dot{x}$$

$$a = f(x, v, t)/m$$
$$v = v_0 + \int_{t_0}^{t} a dt$$
$$x = x_0 + \int_{t_0}^{t} v dt$$

Temporal Discretization

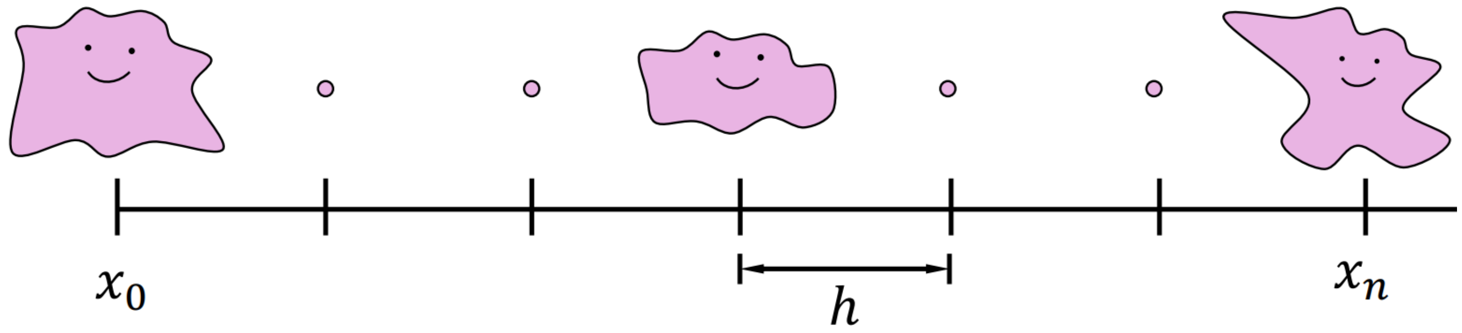
$$x = x(t)$$

$$x_n = x(t_n) \qquad t_n = nh$$



Simulation time step

Temporal Discretization



$$a = f(x, v, t)/m$$

$$v = v_0 + \int_{t_0}^{t} a dt$$

$$x = x_0 + \int_{t_0}^{t} v dt$$

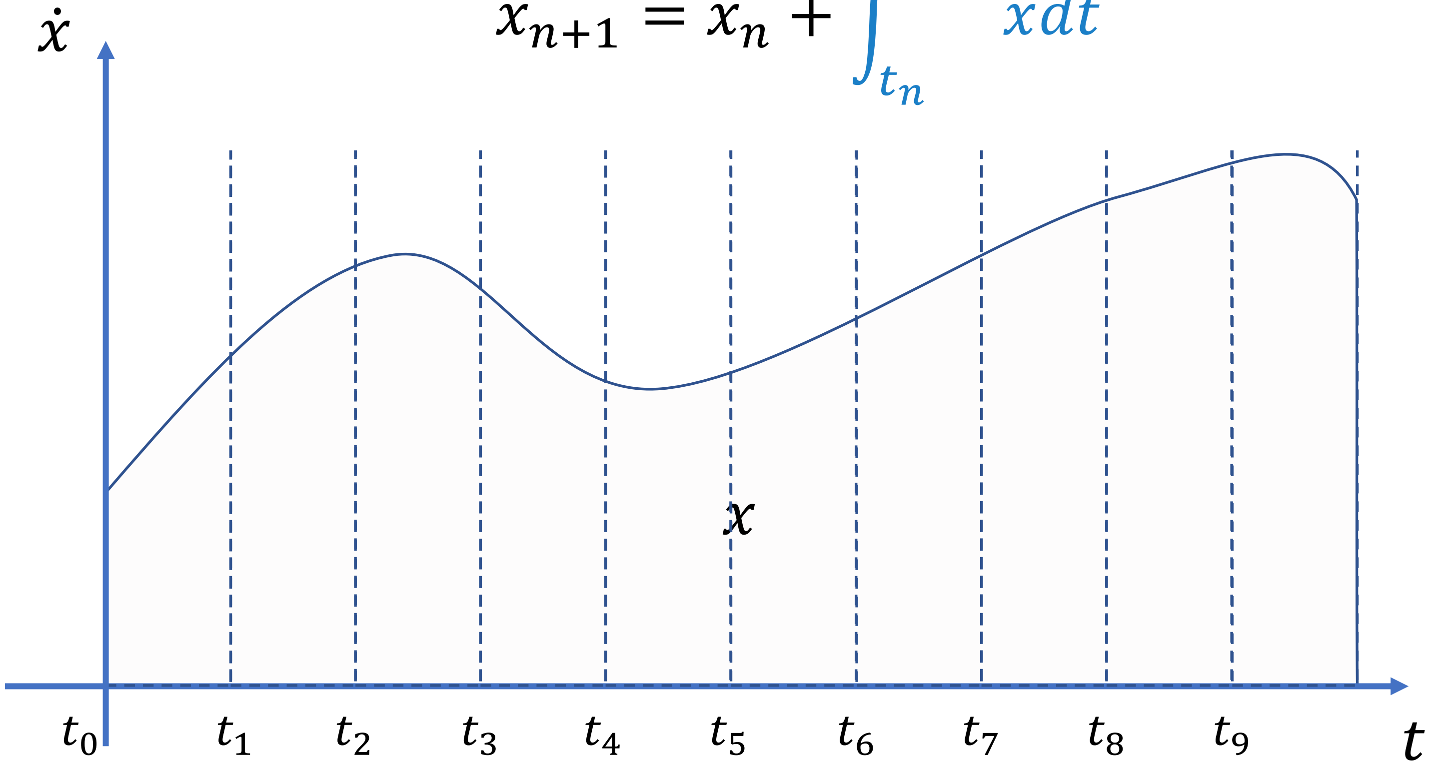
$$a = f(x, v, t)/m$$

$$v_{n+1} = v_n + \int_{t_n}^{t_{n+1}} a dt$$

$$x_{n+1} = x_n + \int_{t_n}^{t_{n+1}} \mathrm{v} dt$$

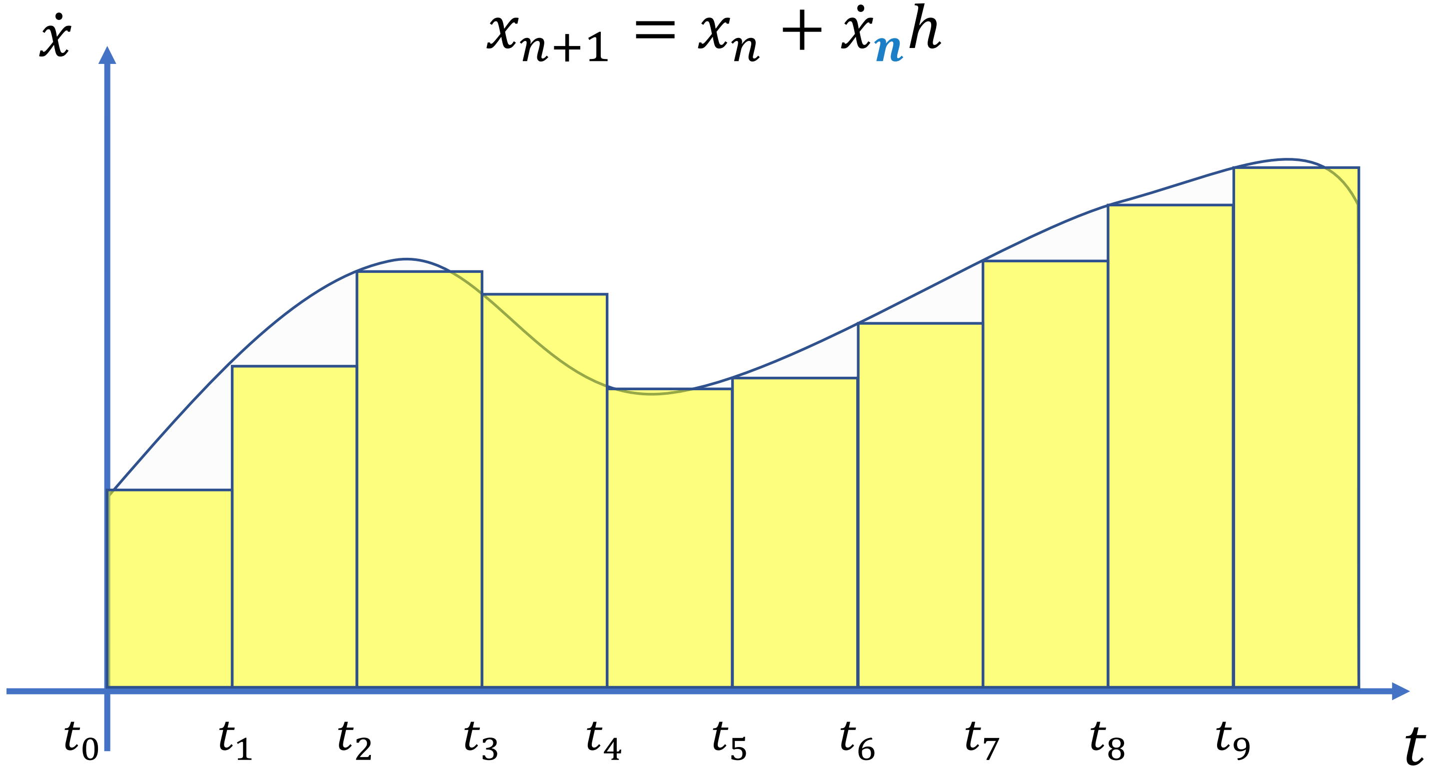
Numerical Integration

$$x_{n+1} = x_n + \int_{t_n}^{t_{n+1}} \dot{x} dt$$

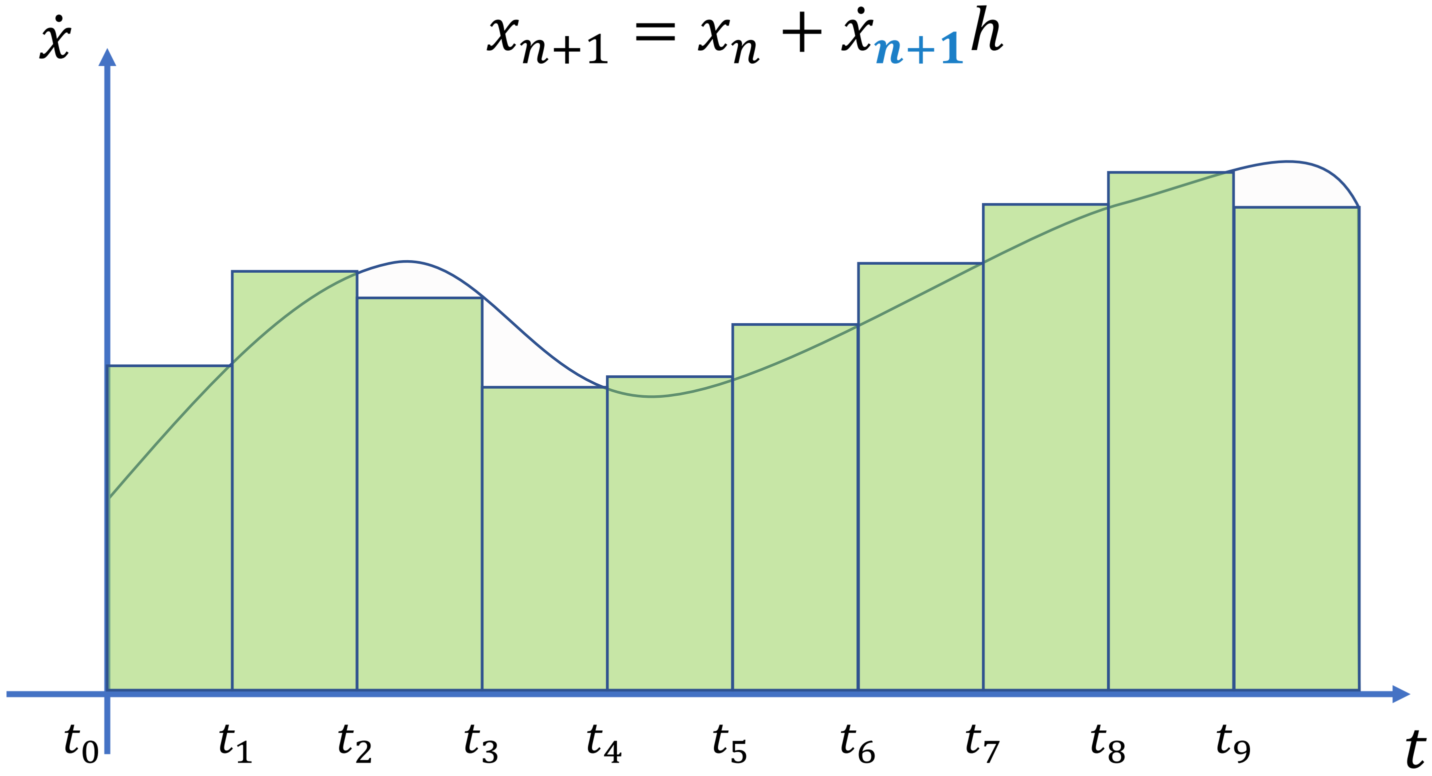


Numerical Integration

$$x_{n+1} = x_n + \dot{x}_n h$$



Numerical Integration



Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n+1}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + \color{red}{a_{\boldsymbol{n+1}}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Requires *"future"* information

Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + f(x_{n+1}, v_{n+1})h \Leftarrow$$ Requires *"future"* information
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}}h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}}h$$

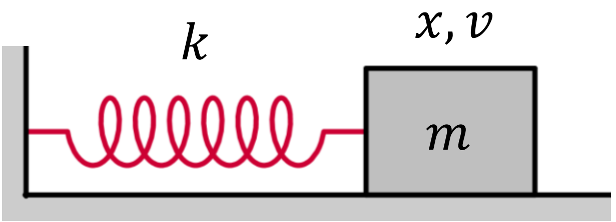
- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n+1}}h$$ Requires *"future"* information
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}}h$$

- Symplectic / Semi-implicit Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}}h$$ All information is current
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}}h$$

Mass on a Spring



$$f = -kx$$

Explicit Euler Integration

$$v_{n+1} = v_n - \frac{kx_{\boldsymbol{n}}}{m}h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}}h$$

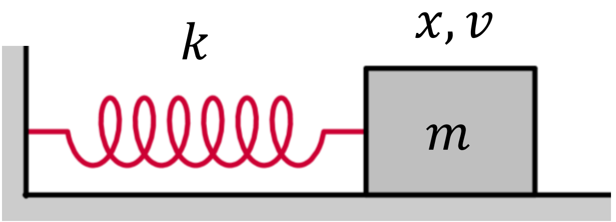
Semi-implicit Euler Integration

$$v_{n+1} = v_n - \frac{kx_{\boldsymbol{n}}}{m}h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}}h$$

Implicit Euler Integration

$$v_{n+1} = v_n - \frac{kx_{\boldsymbol{n+1}}}{m}h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}}h$$

Mass on a Spring



$$f = -kx$$
$$\hat{k} = k/m$$

Explicit Euler Integration

$$\begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & -\hat{k}h \\ h & 1 \end{bmatrix} \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

Semi-implicit Euler Integration

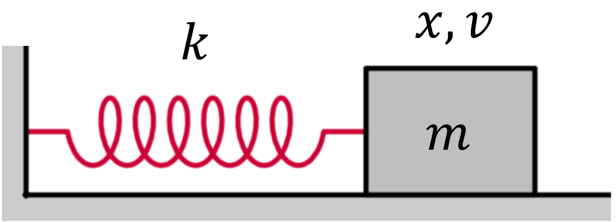
$$\begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & -\hat{k}h \\ h & 1 - \hat{k}h^2 \end{bmatrix} \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

Implicit Euler Integration

$$\begin{bmatrix} 1 & \hat{k}h \\ -h & 1 \end{bmatrix} \begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

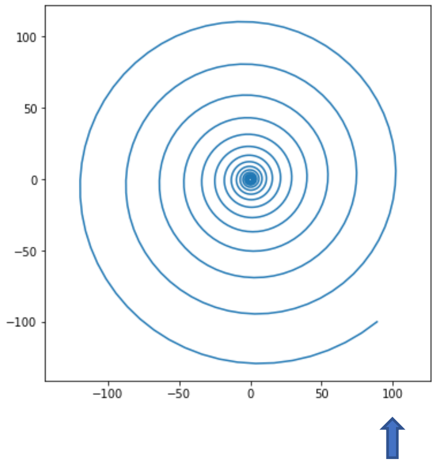
$$\begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \frac{1}{1 + \hat{k}h^2} \begin{bmatrix} 1 & -\hat{k}h \\ h & 1 \end{bmatrix} \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

Mass on a Spring

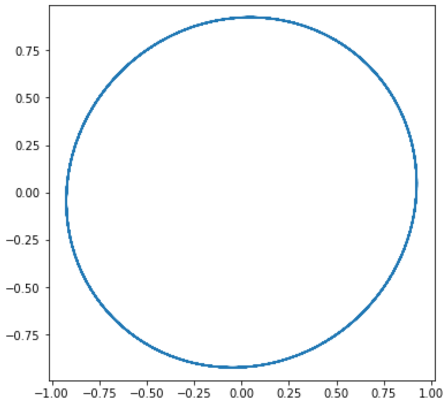


$$f = -kx$$

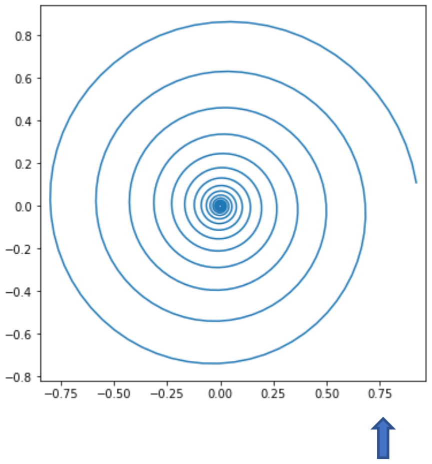
Explicit Euler Integration



Semi-implicit Euler Integration



Implicit Euler Integration



Numerical Integration

- Explicit/Forward Euler
  Symplectic/Semi-implicit Euler
  - Fast, no need to solve equations
  - Can be unstable under large time step

$$v_{n+1} = v_n + f(x_n, v_n)h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}}h$$

$$v_{n+1} = v_n + f(x_n, v_n)h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}}h$$

- Implicit/Backward Euler
  - Rock stable (unconditionally)
  - Slow, need to solve a large problem

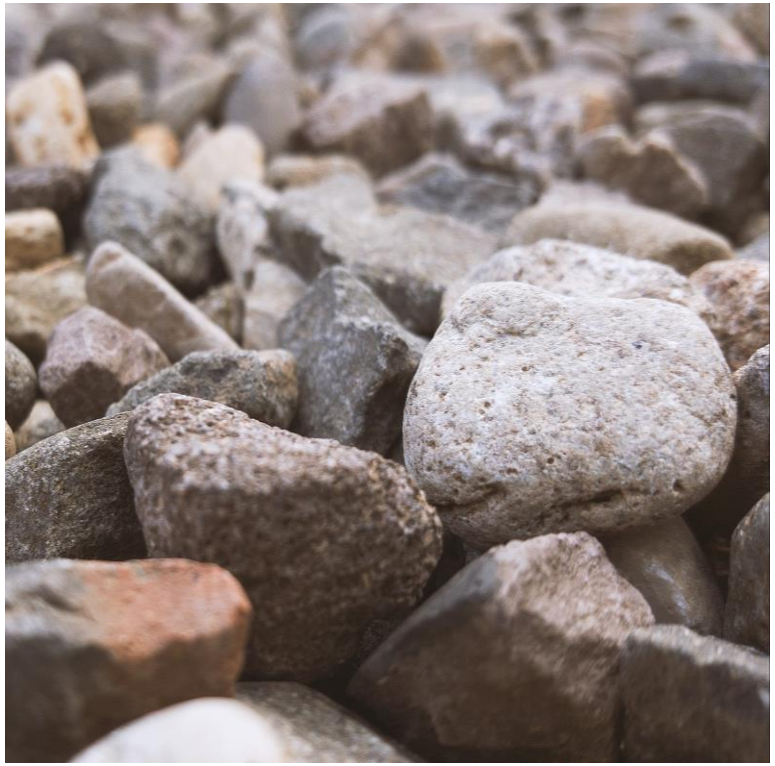
$$v_{n+1} = v_n + f(x_{n+1}, v_{n+1})h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}}h$$

More Advanced Integration

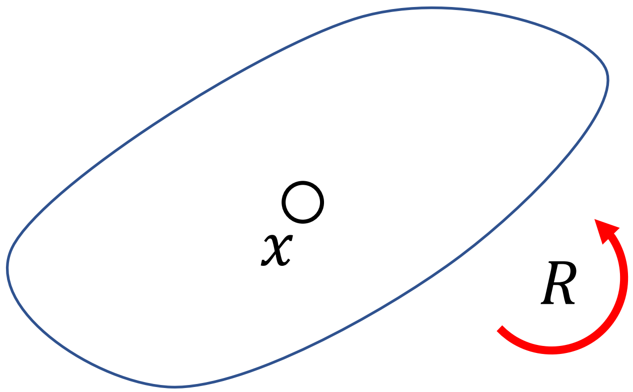
- Runge–Kutta methods
- Variational integration
- Position-based dynamics (PBD)
- ……

Rigid Bodies

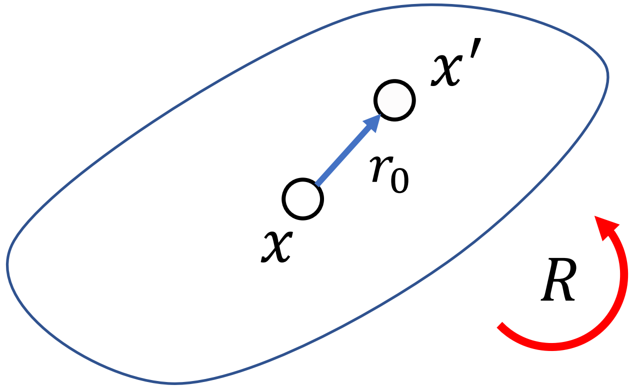
- They are rigid….



Position and Orientation

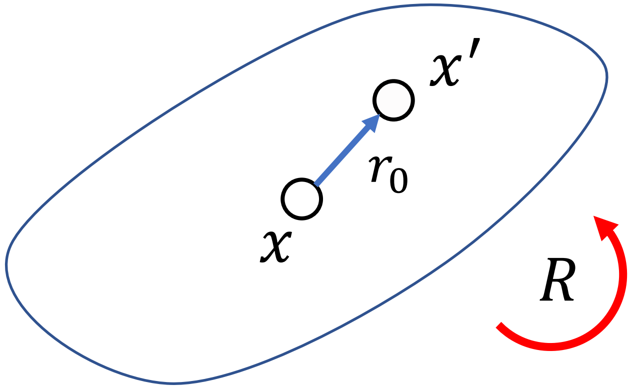


Position and Orientation



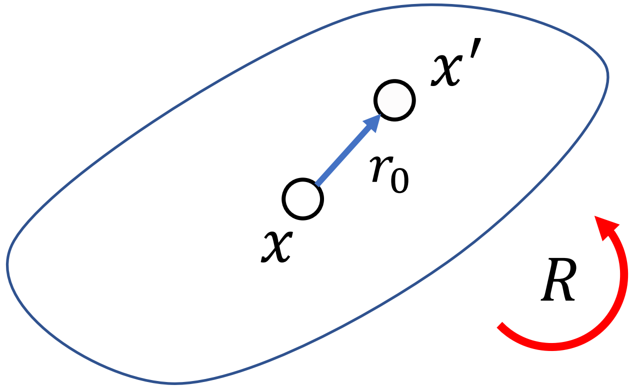
$$x' = x + Rr_0$$

Position and Orientation



$$x' = x + Rr_0 = x + r$$

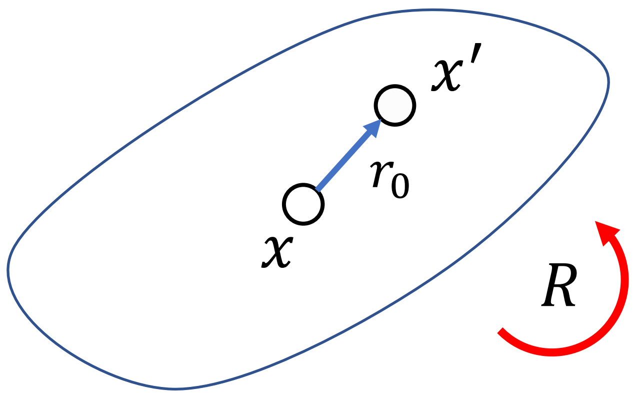
Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

Linear and Angular Velocity

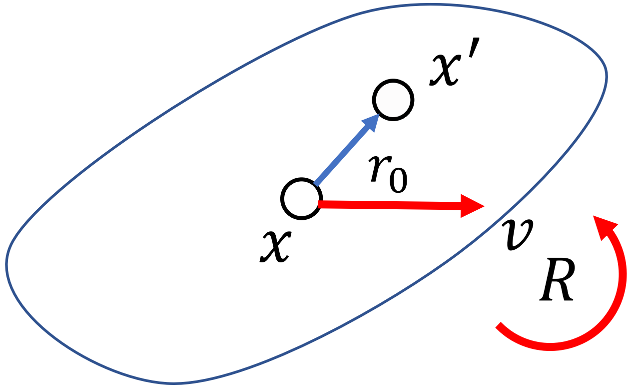


$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \dot{R}r_0$$

Linear and Angular Velocity

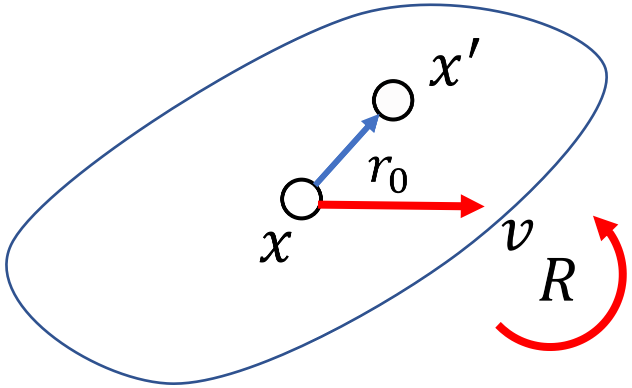


$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \underbrace{\dot{x}}_{v} + \dot{R}r_0$$

$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

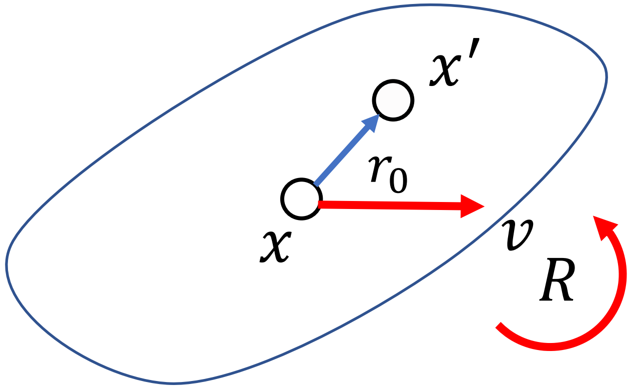
$$\frac{dx'}{dt} = ?$$

$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \dot{R}r_0$$

???

v

Linear and Angular Velocity



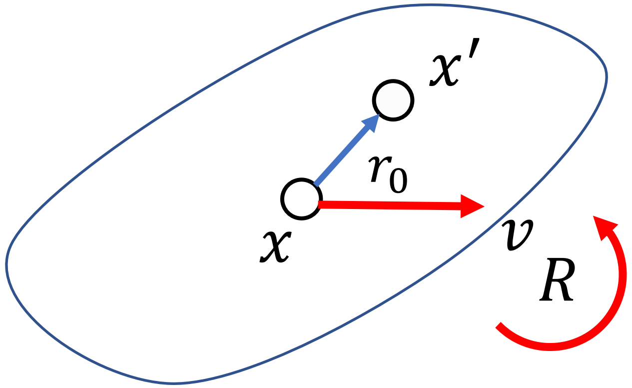
$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \underbrace{\dot{x}}_{v} + \overset{???}{\dot{R}} r_0$$

$$RR^T = I$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

???

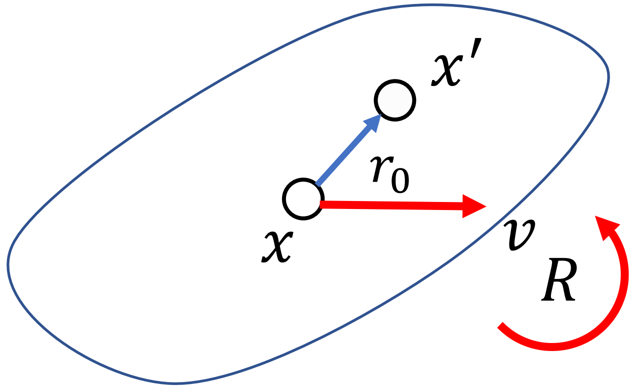
$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \dot{R}r_0$$

v

$$RR^T = I$$

$$\frac{d\left(RR^T\right)}{dt} = 0$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

???

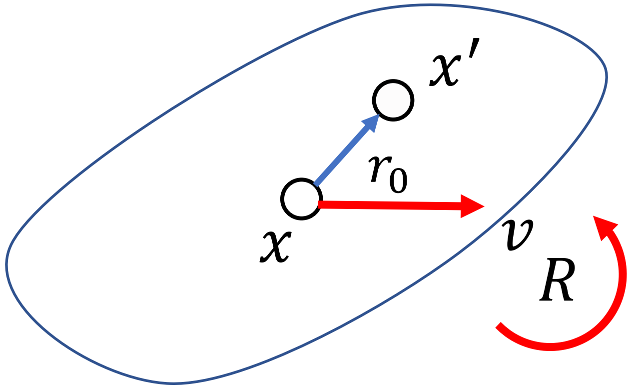
$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \dot{R}r_0$$

v

$$RR^T = I$$

$$\dot{R}R^T + R\dot{R}^T = 0$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

???

$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \dot{R}r_0$$

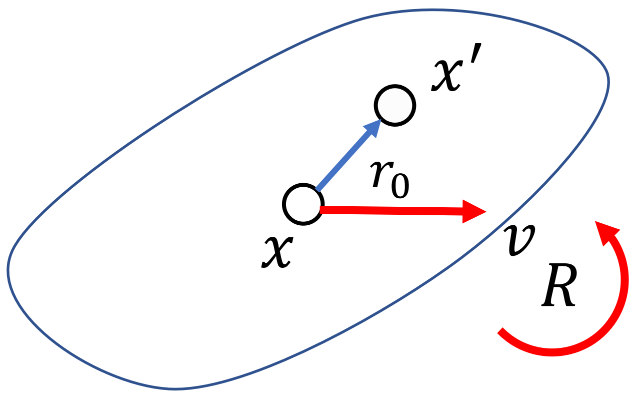
v

$$RR^T = I$$

$$\dot{R}R^T + \left(\dot{R}R^T\right)^T = 0$$

$\dot{R}R^T$ is a Skew-Symmetric Matrix

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

???

$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \dot{R}r_0$$

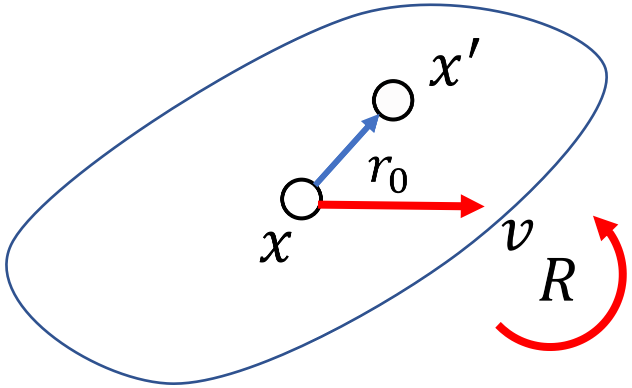
v

$$RR^T = I$$

$$\dot{R}R^T + \left(\dot{R}R^T\right)^T = 0$$

$$\dot{R}R^T = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} = [\omega]_\times$$

Linear and Angular Velocity



$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + [\omega]_\times R r_0$$

v

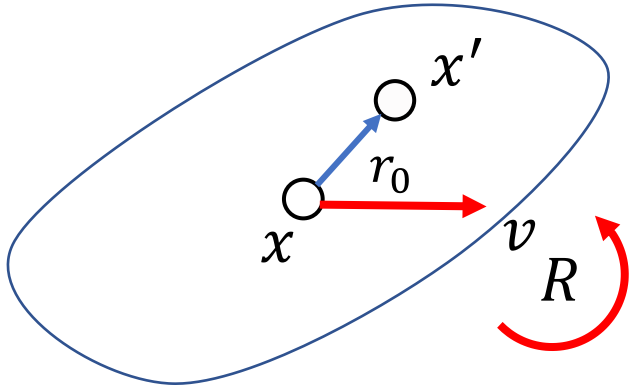
$$x' = x + R r_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

$$\dot{R} = [\omega]_\times R$$

$$\dot{R}R^T = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} = [\omega]_\times$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$\frac{dx'}{dt} = ?$$

$$\frac{dx'}{dt} \Leftrightarrow \dot{x}' = \dot{x} + \omega \times (Rr_0)$$

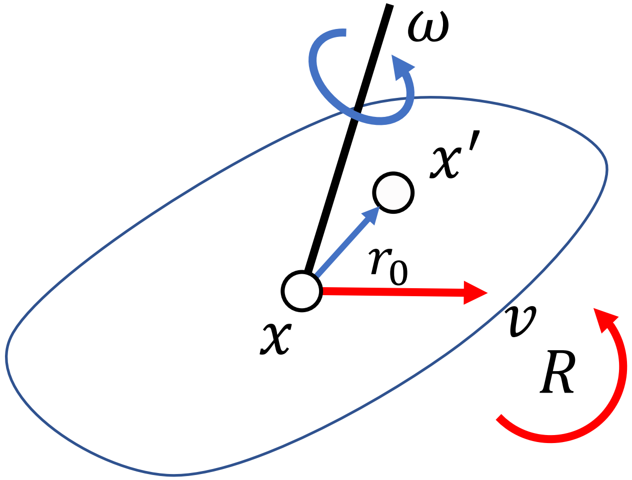
$$\dot{x} \Downarrow v$$

$$\dot{R} = [\omega]_\times R$$

$$\Uparrow$$

$$\dot{R}R^T = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} = [\omega]_\times$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$v' = v + \omega \times r$$

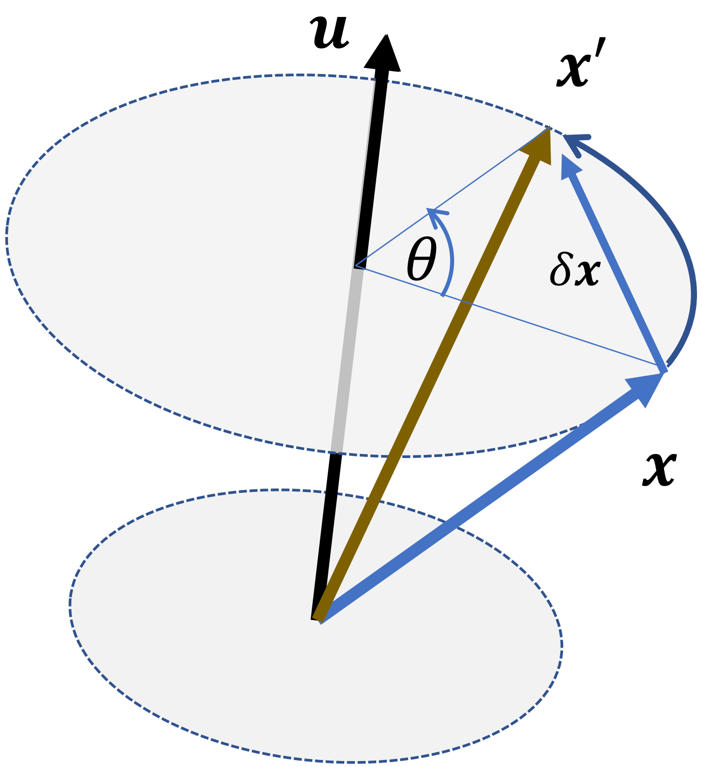
$$\dot{x} = v$$

$$\dot{R} = [\omega]_\times R$$

v: linear velocity

ω: angular velocity

Angular Velocity and Rotation Matrix



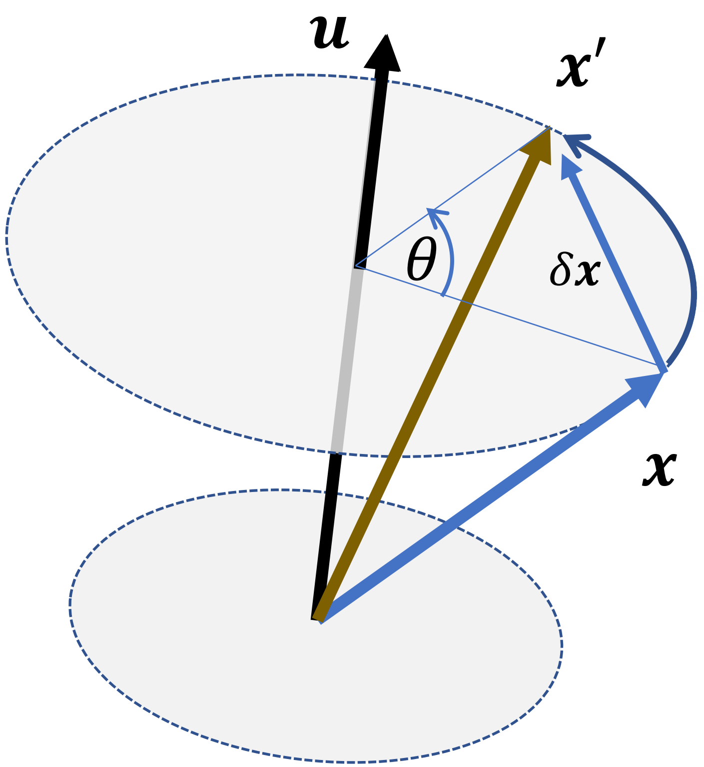
$\|\boldsymbol{u}\| = \mathbf{1}$

Rodrigues' rotation formula

$$\delta \boldsymbol{x} = \boldsymbol{x}' - \boldsymbol{x}$$

$$= (\sin\theta)\, \boldsymbol{u} \times \boldsymbol{x} + (1 - \cos\theta)\, \boldsymbol{u} \times (\boldsymbol{u} \times \boldsymbol{x})$$

Angular Velocity and Rotation Matrix



$\|\boldsymbol{u}\| = 1$

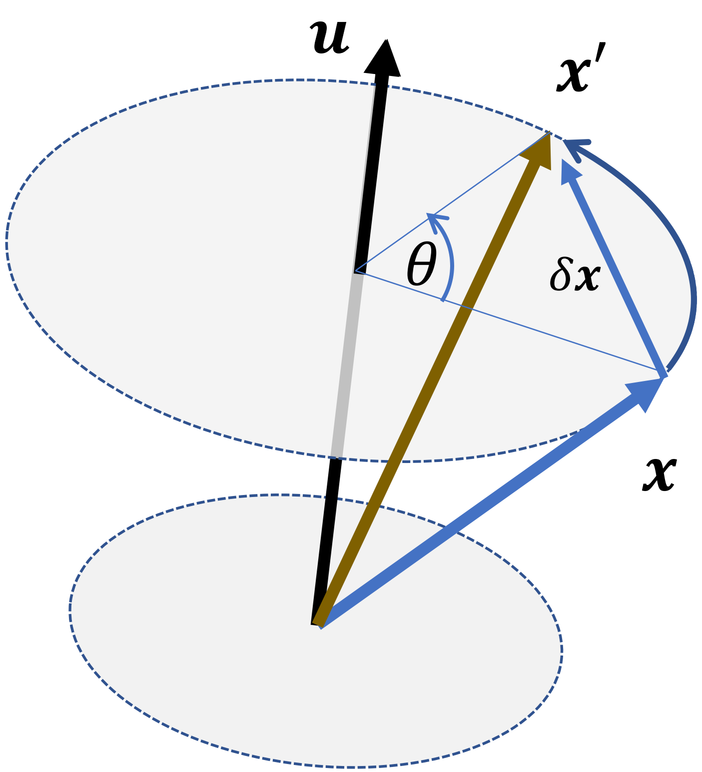
Rodrigues' rotation formula

$$\delta \boldsymbol{x} = \boldsymbol{x}' - \boldsymbol{x}$$

$$= (\sin\theta)\, \boldsymbol{u} \times \boldsymbol{x} + (1 - \cos\theta)\, \boldsymbol{u} \times (\boldsymbol{u} \times \boldsymbol{x})$$

$$\dot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{dt} = \frac{d\boldsymbol{x}}{d\theta} \cdot \frac{d\theta}{dt} = \dot{\theta} \boldsymbol{u} \times \boldsymbol{x}$$

Angular Velocity and Rotation Matrix



$$\|\boldsymbol{u}\| = \mathbf{1}$$

Rodrigues' rotation formula

$$\delta \boldsymbol{x} = \boldsymbol{x}' - \boldsymbol{x}$$

$$= (\sin\theta)\, \boldsymbol{u} \times \boldsymbol{x} + (1 - \cos\theta)\, \boldsymbol{u} \times (\boldsymbol{u} \times \boldsymbol{x})$$

$$\Downarrow$$

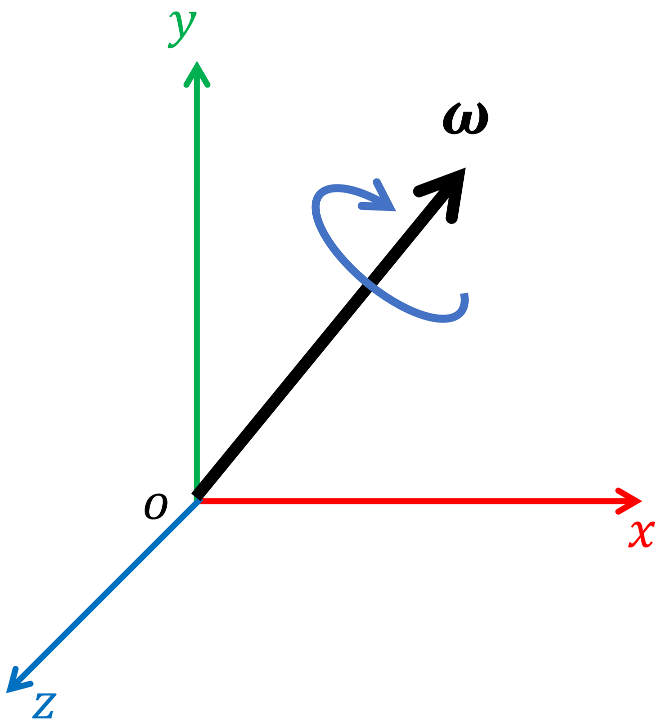
$$\dot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{dt} = \frac{d\boldsymbol{x}}{d\theta} \cdot \frac{d\theta}{dt} = \dot{\theta}\boldsymbol{u} \times \boldsymbol{x}$$

$$\Downarrow$$

$$\dot{\boldsymbol{x}} = \boldsymbol{\omega} \times \boldsymbol{x}$$

Angular Velocity and Rotation Matrix

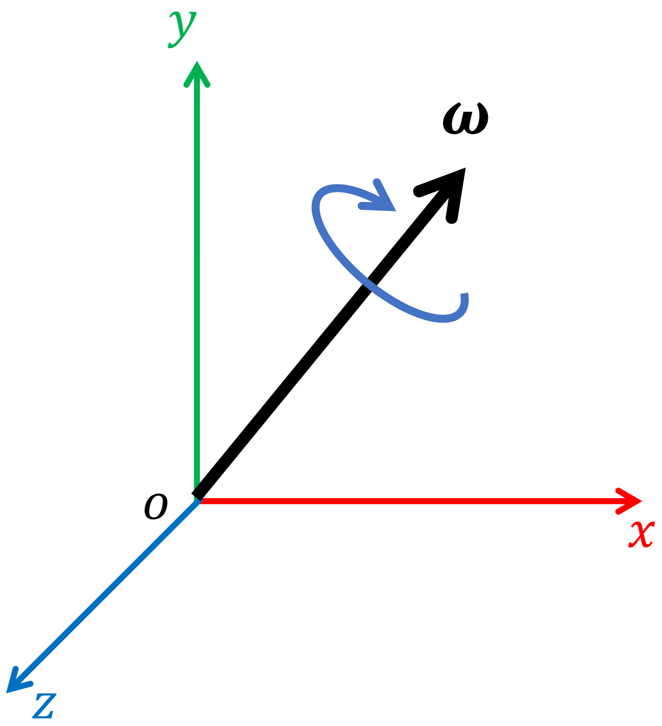
ω: angular velocity $\Longleftrightarrow$ $\dot{R} = [\omega]_\times R$



$$R = \begin{bmatrix} | & | & | \\ \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \\ | & | & | \end{bmatrix}$$

Angular Velocity and Rotation Matrix

ω: angular velocity $\Longleftrightarrow$ $\dot{R} = [\omega]_\times R$



$$R = \begin{bmatrix} | & | & | \\ \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \\ | & | & | \end{bmatrix}$$

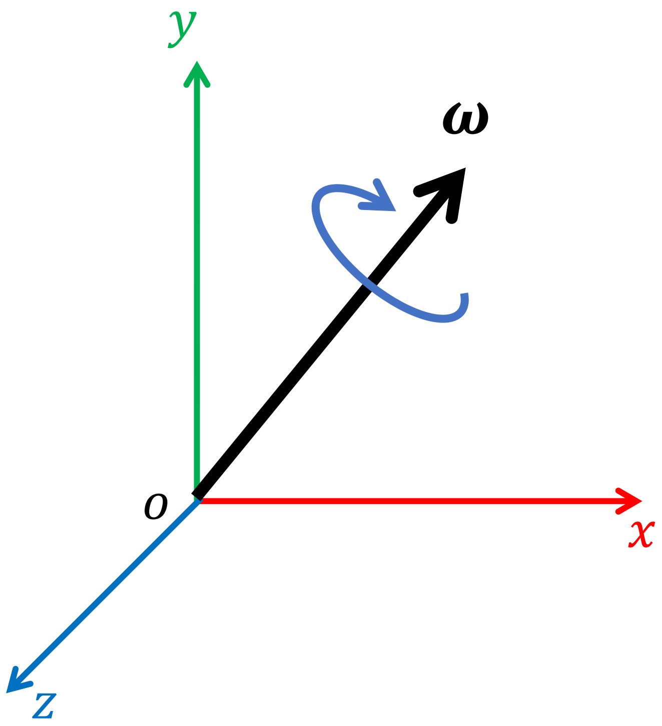
$$\dot{\boldsymbol{e}}_x = \boldsymbol{\omega} \times \boldsymbol{e}_x$$

$$\dot{\boldsymbol{e}}_y = \boldsymbol{\omega} \times \boldsymbol{e}_y$$

$$\dot{\boldsymbol{e}}_z = \boldsymbol{\omega} \times \boldsymbol{e}_z$$

Angular Velocity and Rotation Matrix

ω: angular velocity $\quad\Longleftrightarrow\quad$ $\dot{R} = [\omega]_\times R$



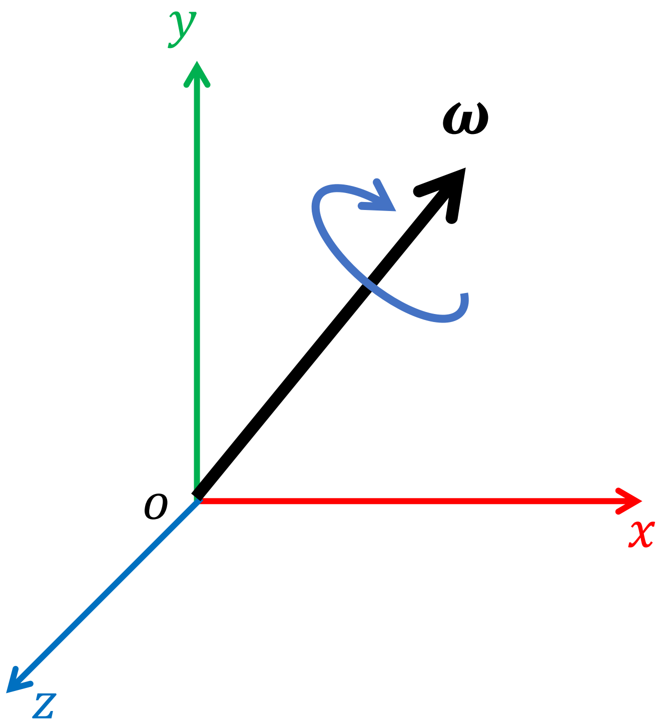
$$R = \begin{bmatrix} | & | & | \\ \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \\ | & | & | \end{bmatrix}$$

$$\Downarrow$$

$$\dot{R} = \begin{bmatrix} | & | & | \\ \dot{\boldsymbol{e}}_x & \dot{\boldsymbol{e}}_y & \dot{\boldsymbol{e}}_z \\ | & | & | \end{bmatrix} = \begin{bmatrix} | & | & | \\ \boldsymbol{\omega} \times \dot{\boldsymbol{e}}_x & \boldsymbol{\omega} \times \dot{\boldsymbol{e}}_y & \boldsymbol{\omega} \times \dot{\boldsymbol{e}}_z \\ | & | & | \end{bmatrix}$$

Angular Velocity and Rotation Matrix

ω: angular velocity $\Longleftrightarrow$ $\dot{R} = [\omega]_\times R$

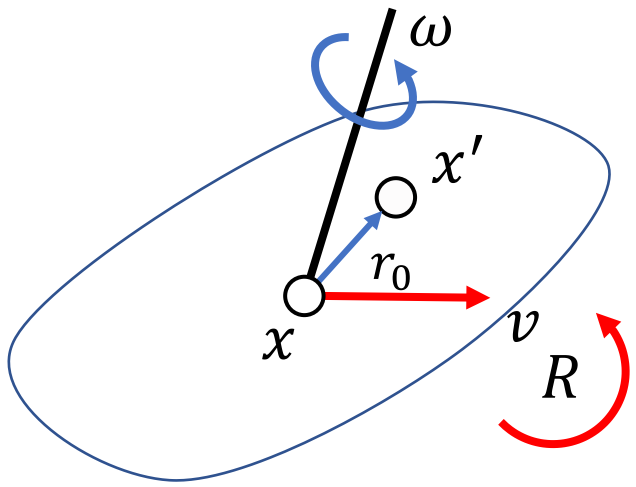


$$R = \begin{bmatrix} | & | & | \\ \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \\ | & | & | \end{bmatrix}$$

$$\Downarrow$$

$$\dot{R} = \begin{bmatrix} | & | & | \\ \dot{\boldsymbol{e}}_x & \dot{\boldsymbol{e}}_y & \dot{\boldsymbol{e}}_z \\ | & | & | \end{bmatrix} = [\boldsymbol{\omega}]_\times \begin{bmatrix} | & | & | \\ \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \\ | & | & | \end{bmatrix}$$

Linear and Angular Velocity



$$x' = x + Rr_0 = x + r$$

$$v' = v + \omega \times r$$

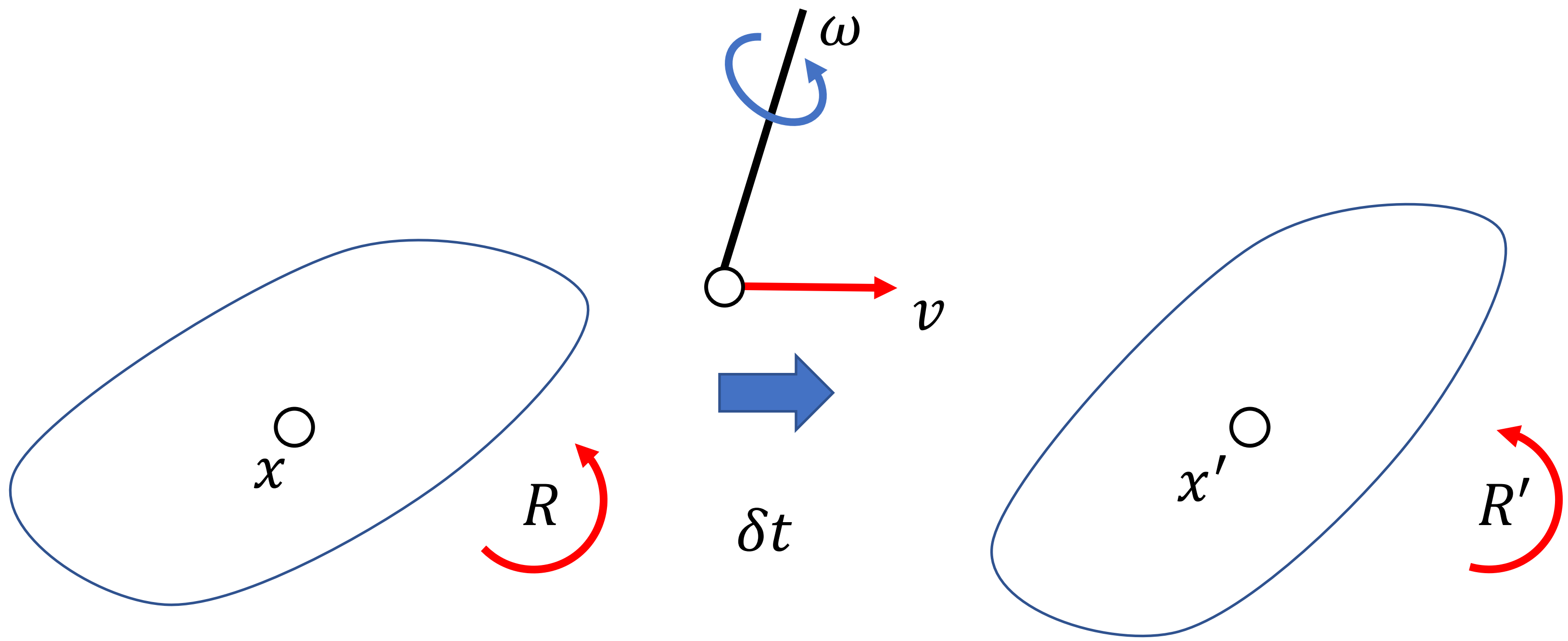
$$\dot{x} = v$$

$$\dot{R} = [\omega]_\times R$$

v: linear velocity

ω: angular velocity

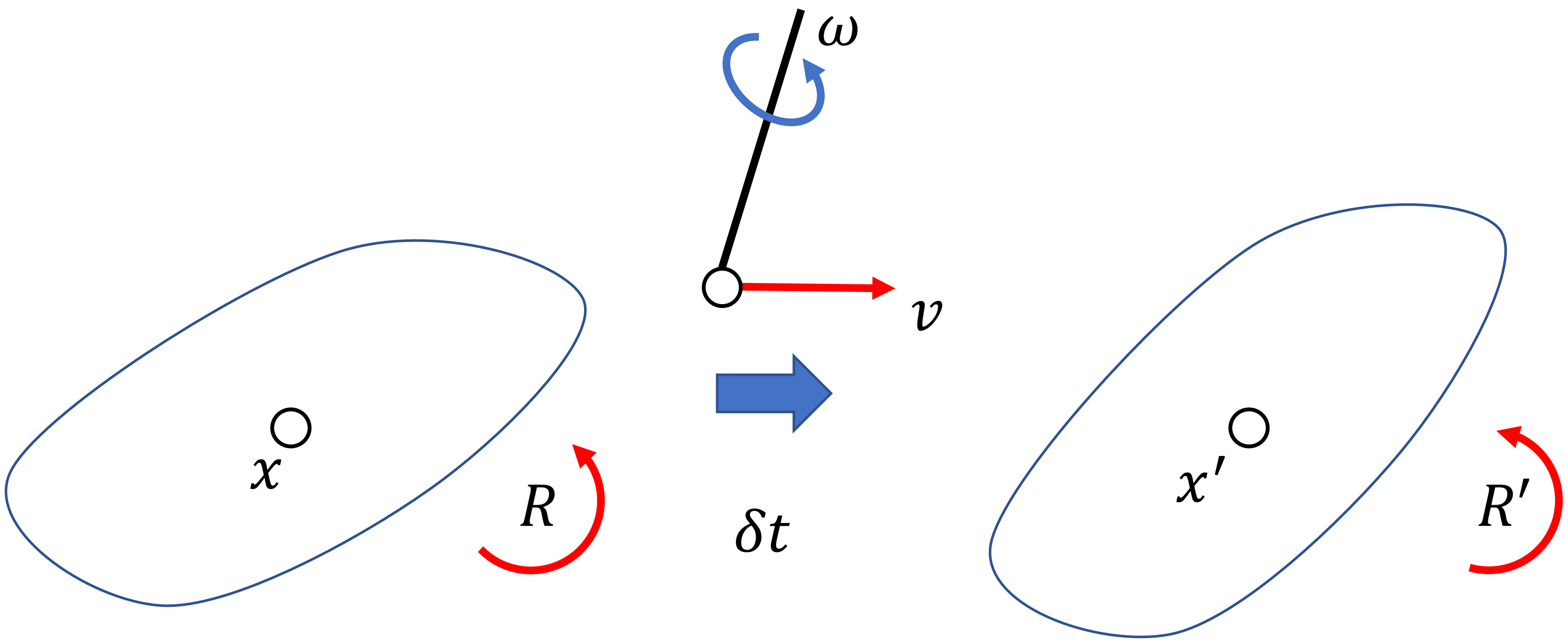
Numerical Integration



$$x' = ?$$

$$R' = ?$$

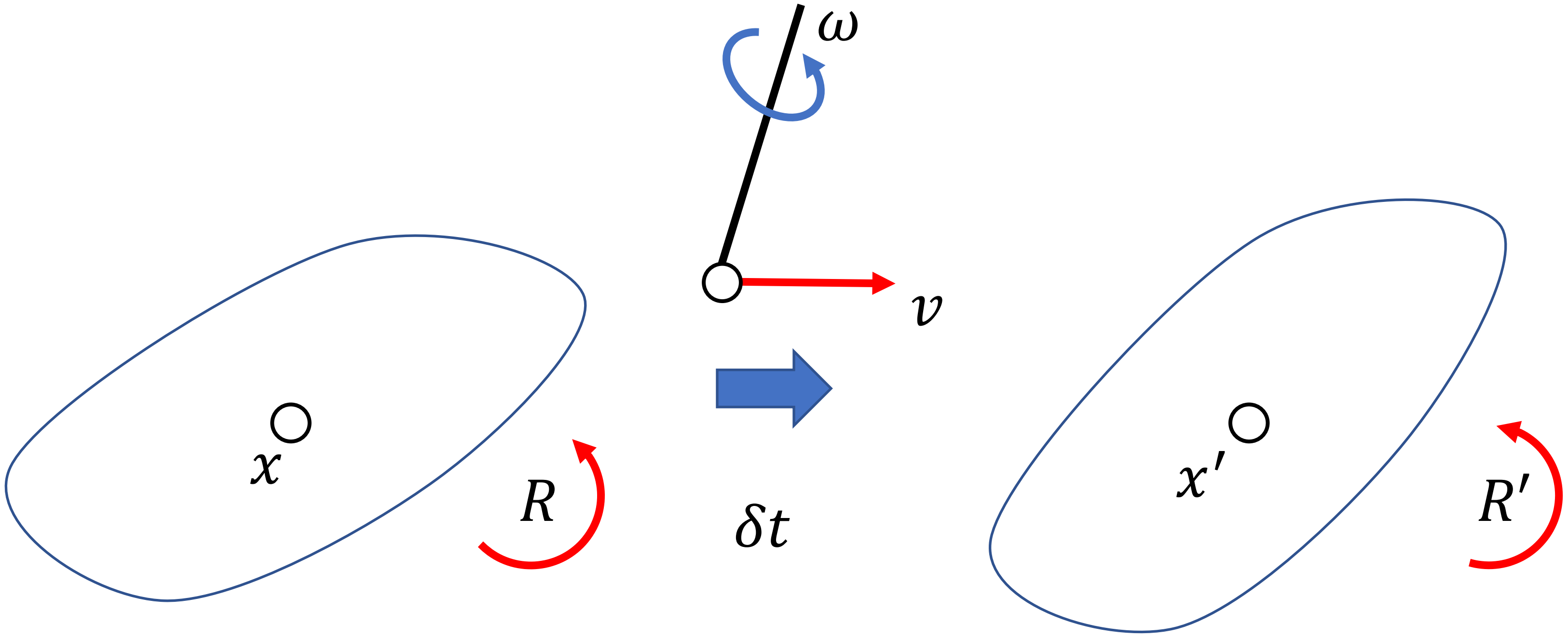
Numerical Integration



$$\dot{x} = v$$

$$\dot{R} = [\omega]_\times R$$

Numerical Integration

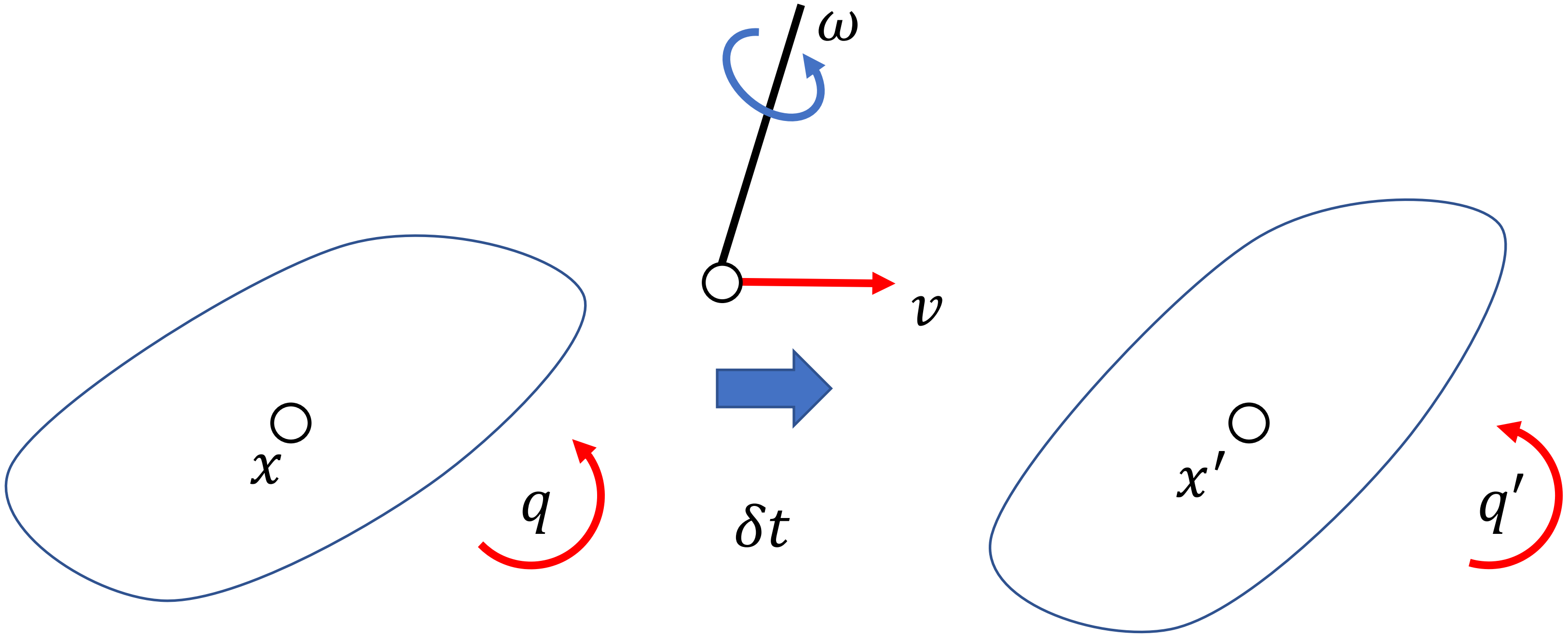


$$\dot{x} = v \quad \Rightarrow \quad x' = x + \delta t \cdot v$$

$$\dot{R} = [\omega]_\times R \quad \Rightarrow \quad R' = R + \delta t \cdot [\omega]_\times R$$

Need orthogonalization!

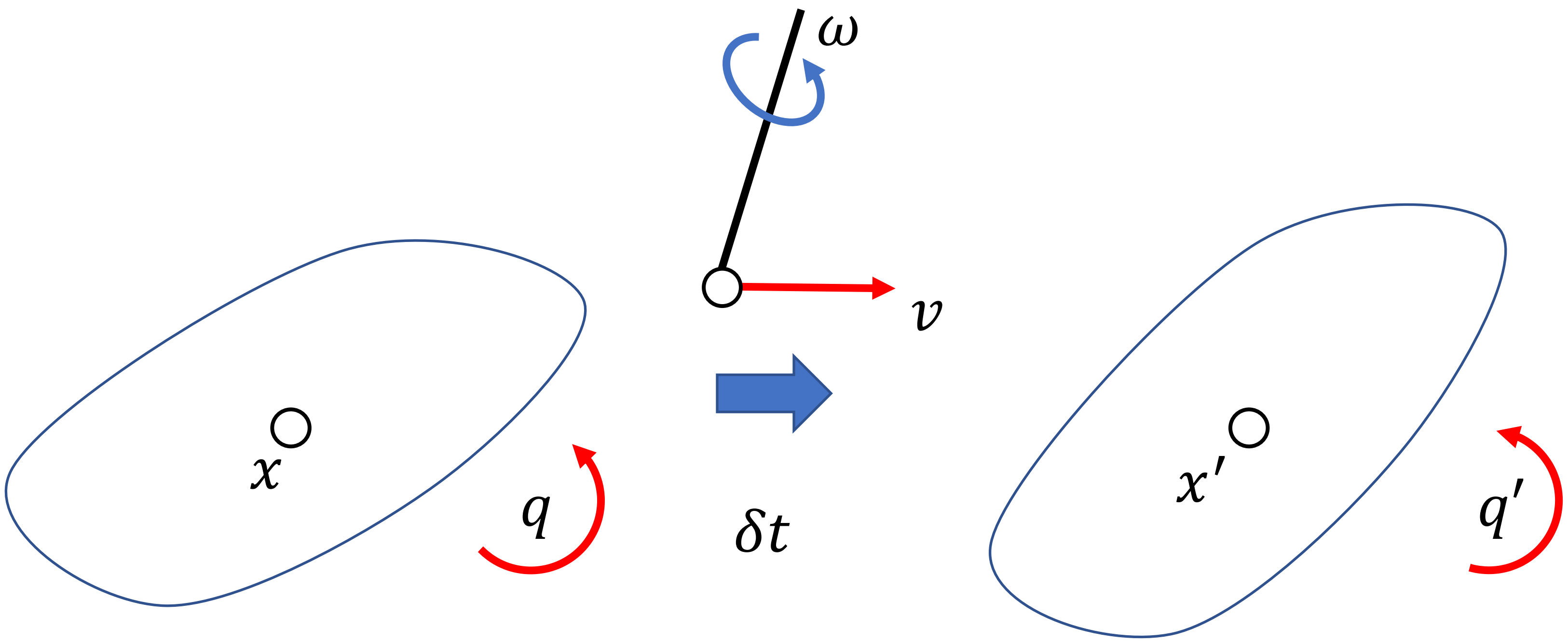
Numerical Integration: Quaternion



$$\dot{x} = v \qquad \Rightarrow \qquad x' = x + \delta t \cdot v$$

$$\dot{q} = ? \qquad \qquad q' = q + \delta t \cdot \dot{q}$$

Numerical Integration: Quaternion



$$\dot{x} = v$$

$$\dot{q} = \frac{1}{2}\bar{\omega}q$$

$$\bar{\omega} = (0, \omega)$$

$$x' = x + \delta t \cdot v$$

$$q' = q + \delta t \cdot \dot{q}$$

Need Normalization!

Kinematics vs. Dynamics

Kinematics

x, R

v, ω

a, α

$\dddot{x}, \ddot{\omega}$

...

m, I

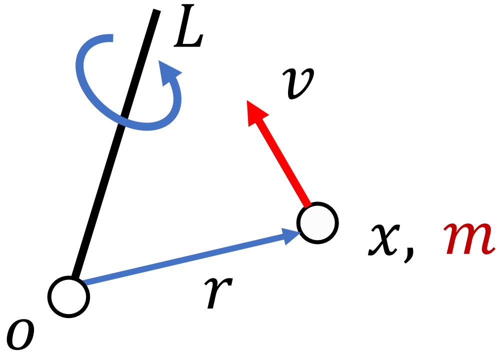
$\Longleftrightarrow$

Dynamics

p, L

F, τ

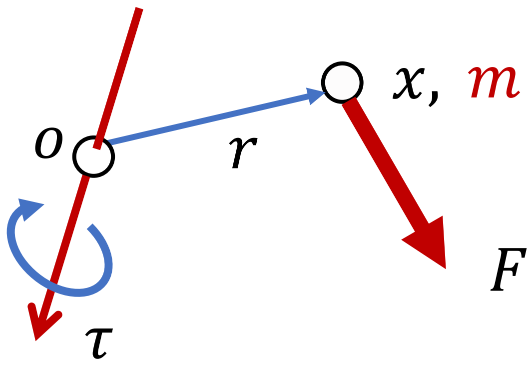
Linear and Angular Momentum of a Particle



$p = m\,v$ Linear momentum of x

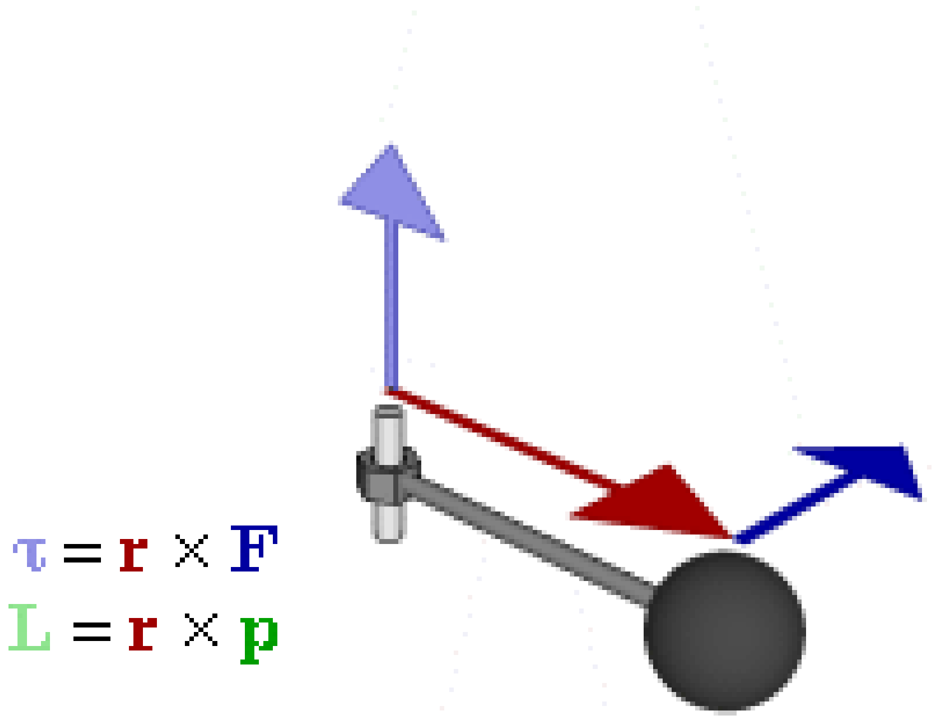
$L = m\,r \times v$ Angular momentum of x w.r.t. o

Force and Torque



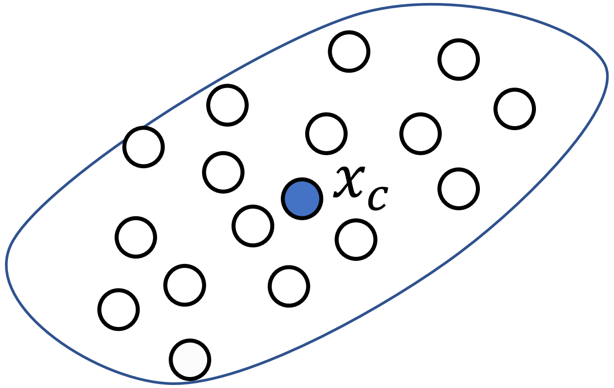
$$\tau = r \times F$$

Torque and Angular Momentum



https://en.wikipedia.org/wiki/Torque

Rigid Body as a Collection of Particles



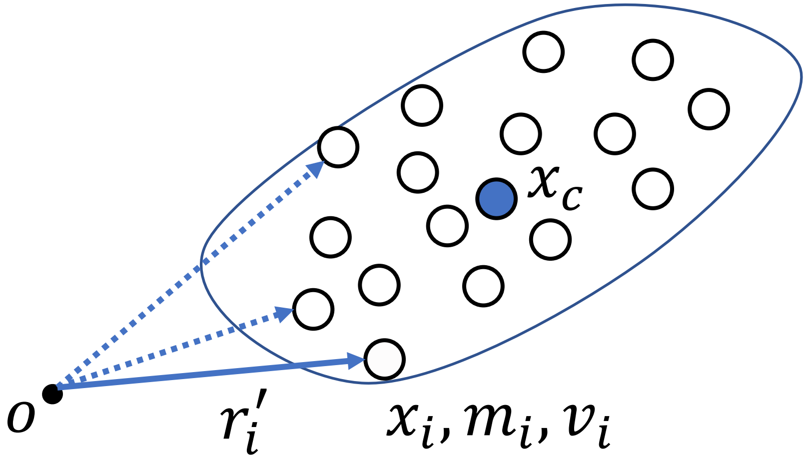
m_i, x_i, v_i

$$M = \sum_i m_i$$

$$x_c = \frac{\sum_i m_i x_i}{\sum_i m_i}$$

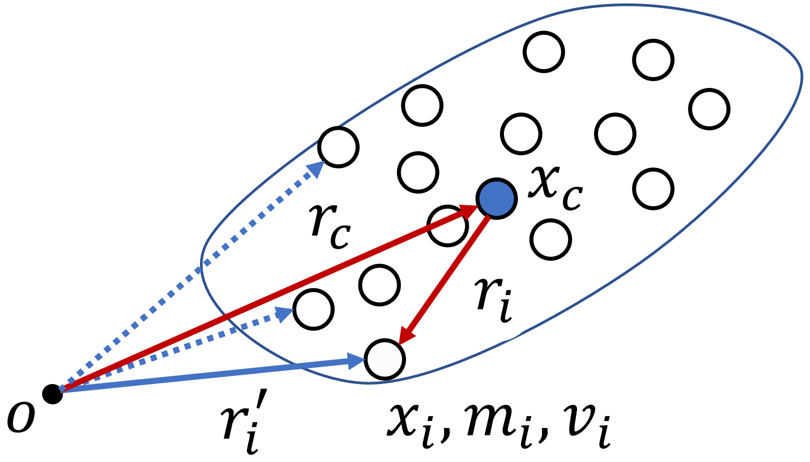
$$v_c = \frac{\sum_i m_i v_i}{\sum_i m_i}$$

Moments of a Rigid Body



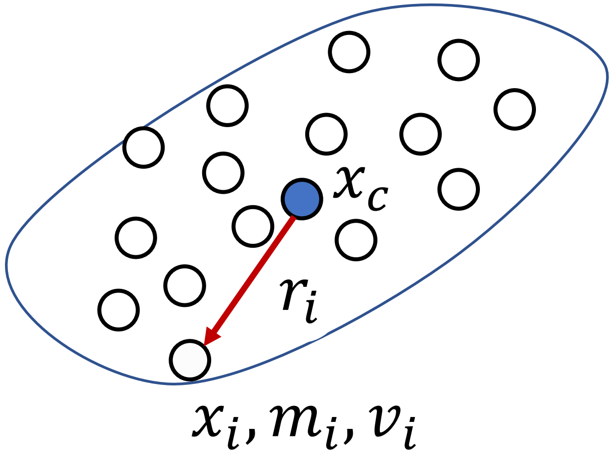
$$p = \sum_i m_i v_i \qquad L_o = \sum_i m_i r_i' \times v_i$$

Angular Moment of a Rigid Body



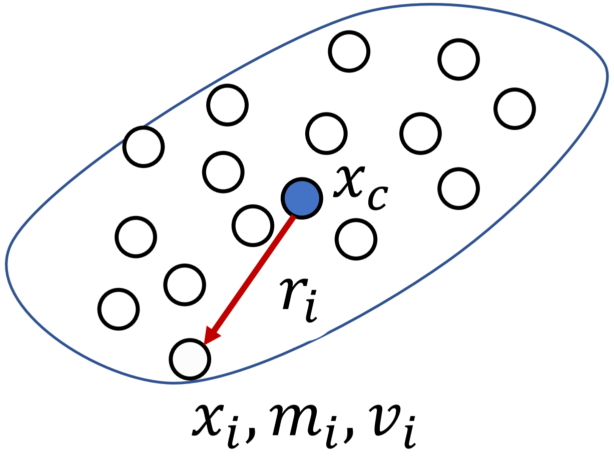
$$L_o = \sum_i m_i r_i' \times v_i = M r_c \times v_c + \sum_i m_i r_i \times v_i$$

Angular Moment of a Rigid Body



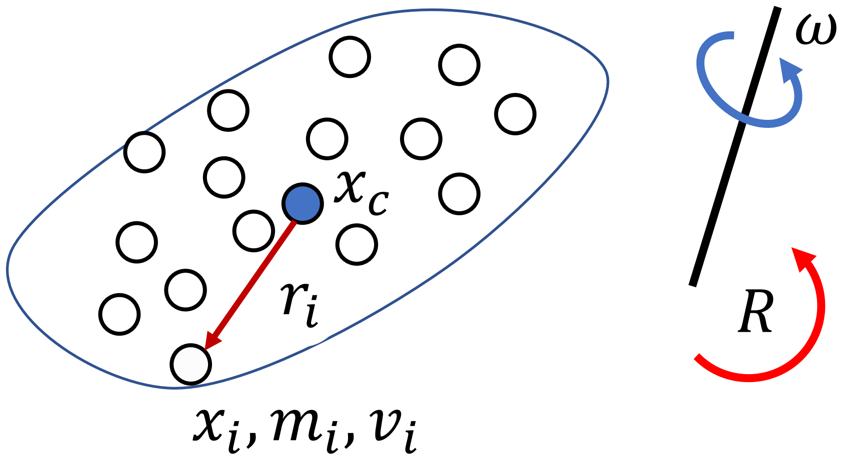
$$L_{x_c} = \sum_i m_i r_i \times v_i$$

Angular Moment of a Rigid Body



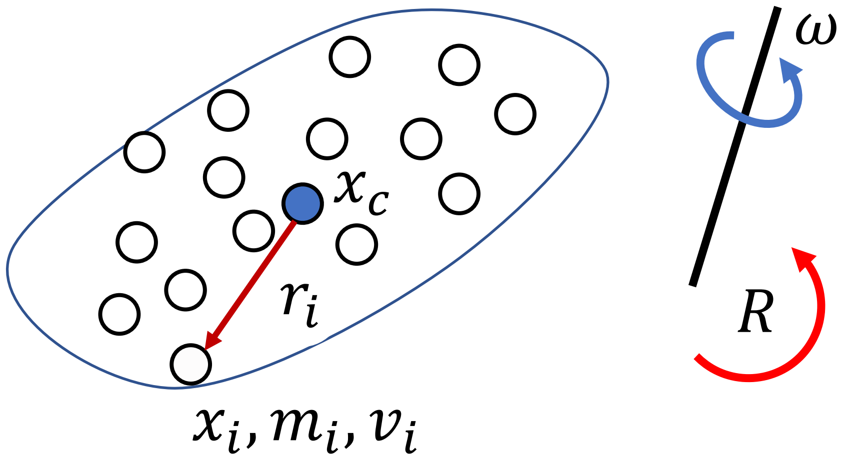
$$L = \sum_i m_i r_i \times v_i$$

Angular Moment of a Rigid Body



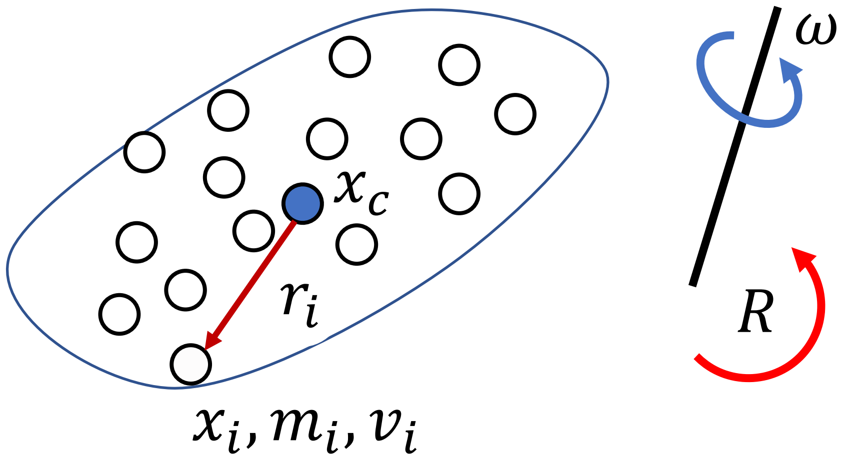
$$L = \sum_i m_i r_i \times v_i$$

Angular Moment of a Rigid Body



$$L = \sum_i m_i r_i \times (\omega \times r_i)$$

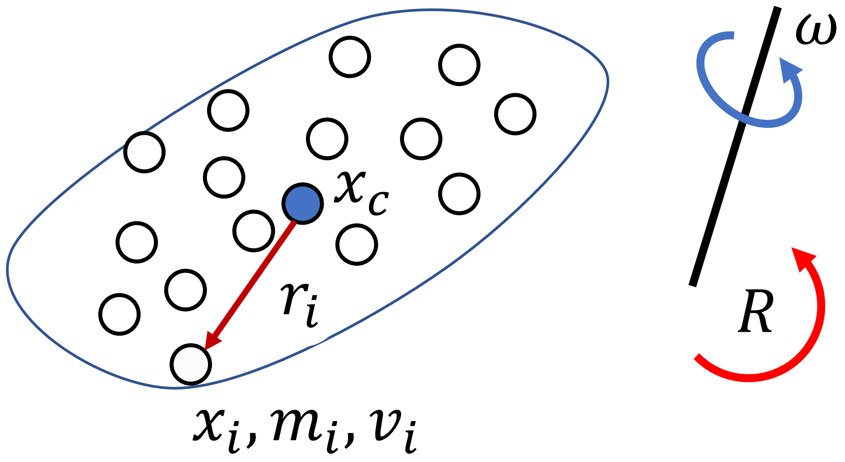
Angular Moment of a Rigid Body



$$L = \sum_i -m_i [r_i]_\times^2 \, \omega$$

$$[a]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

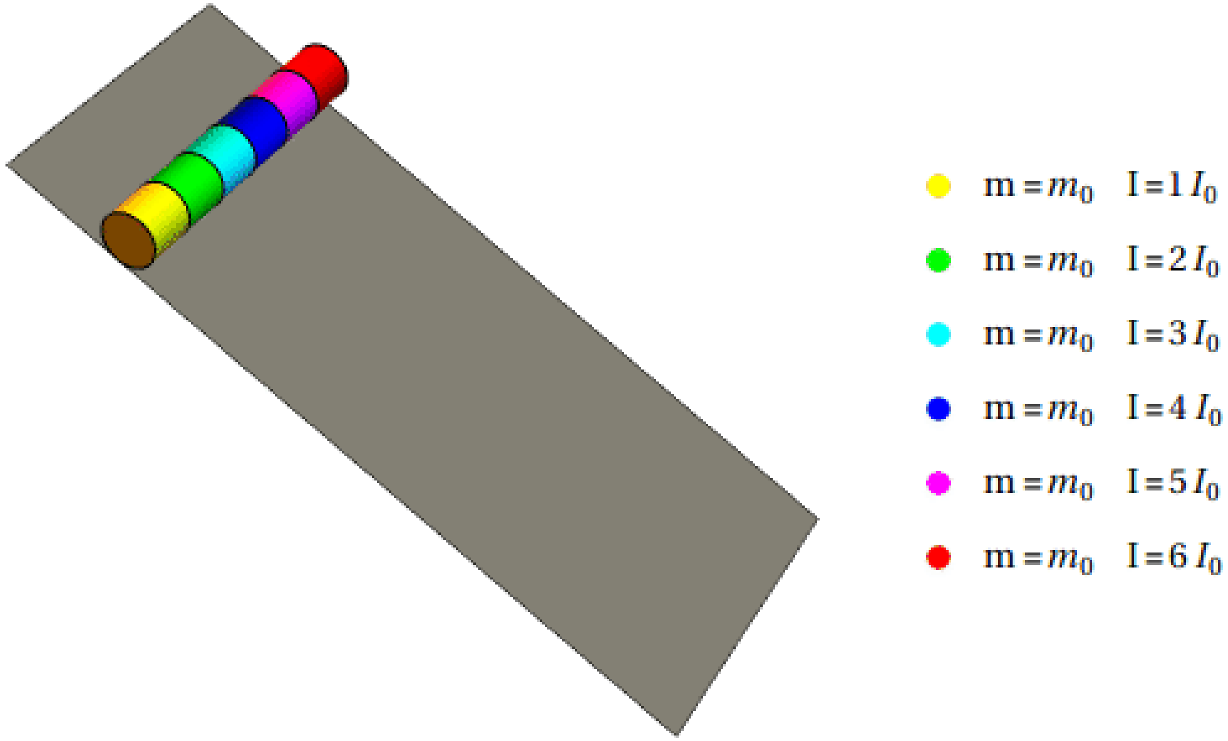
Angular Moment of a Rigid Body



$$L = I\omega$$

Moment of Inertia: $$I = \sum_i -m_i [r_i]_\times^2$$

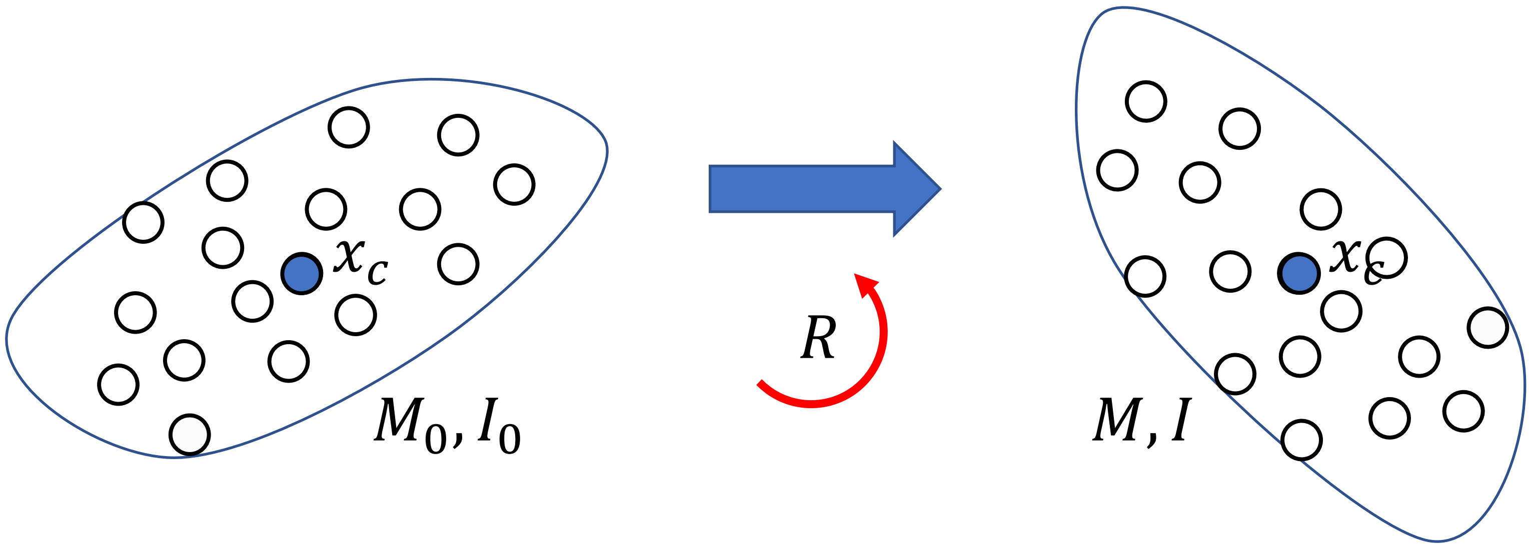
Moment of Inertia



Moment of Inertia

https://en.wikipedia.org/wiki/Moment_of_inertia

Rotation of Moment of Inertia



$$M = M_0$$

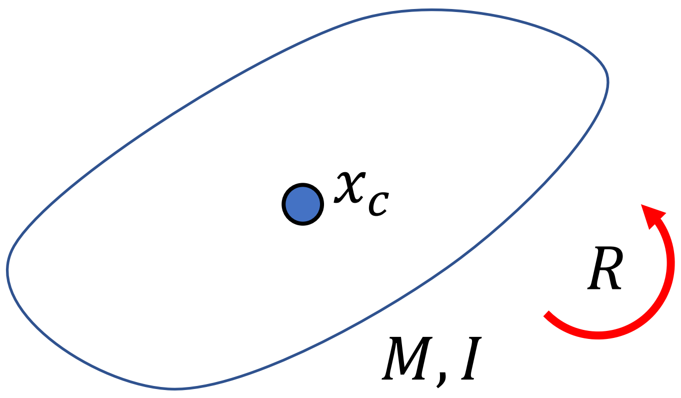
$$I = R I_0 R^T$$

$$(Rr) \times x = R\left(r \times \left(R^T x\right)\right)$$

$$[Rr]_\times = R[r]_\times R^T$$

$$[Rr]_\times^2 = R[r]_\times^2 R^T$$

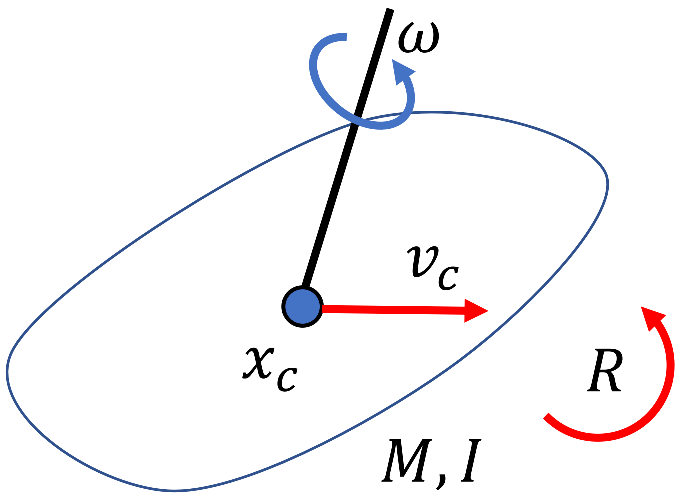
Principal Axes of Moment of Inertia



Eigendecomposition ⇒ $I = R I_0 R^T$

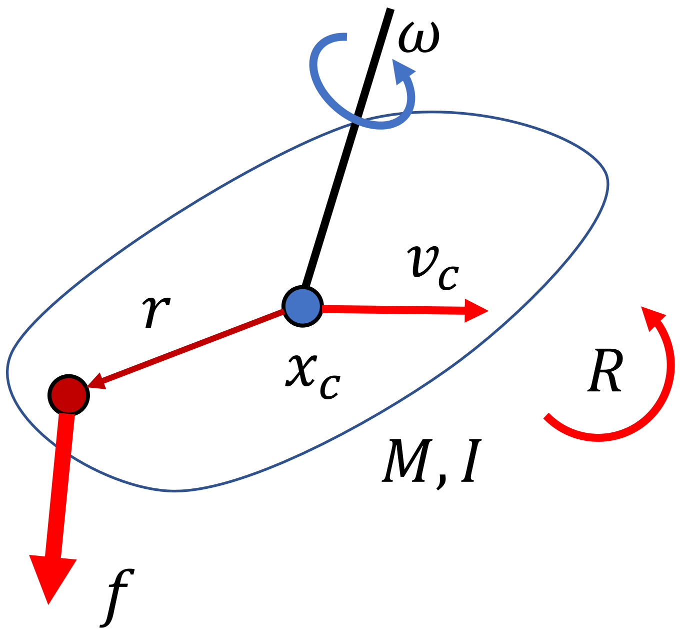
$$I_0 = \begin{bmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{bmatrix} = \mathrm{diag}(I_1, I_2, I_3)$$

Center of Momentum (CoM) Frame

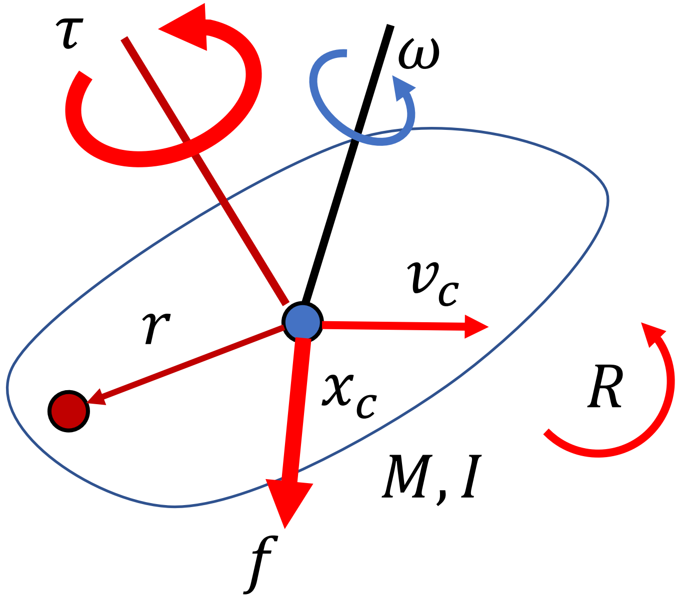


$$p = Mv_c \qquad\qquad L = I\omega$$

Force on a Rigid Body

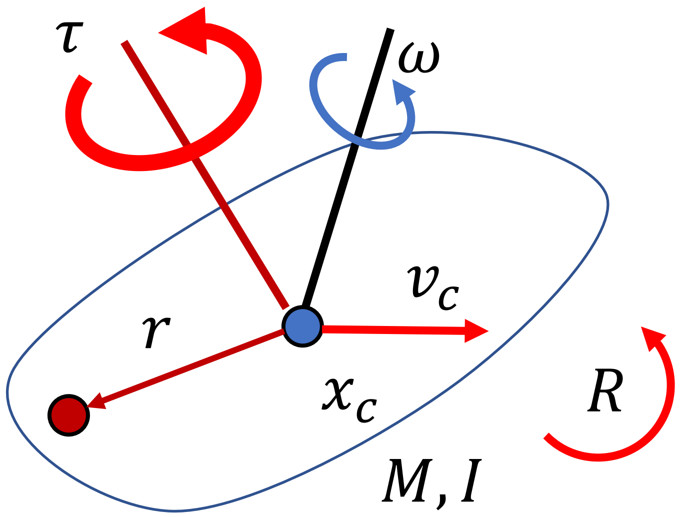


Force on a Rigid Body



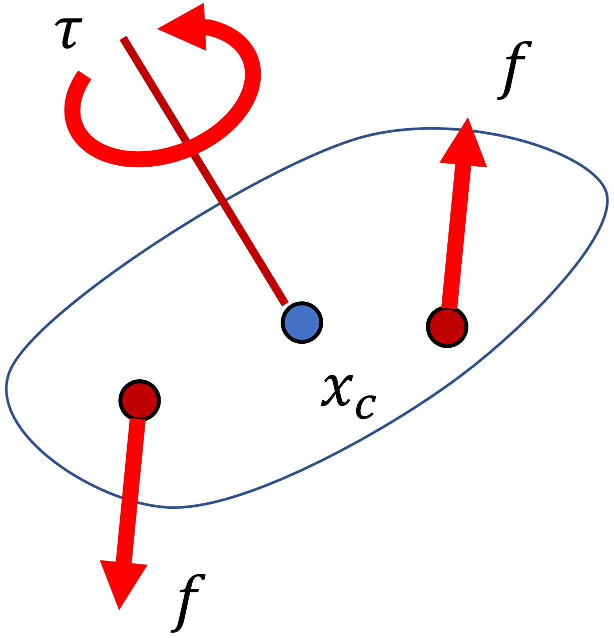
$$\tau = r \times f$$

Torque on a Rigid Body



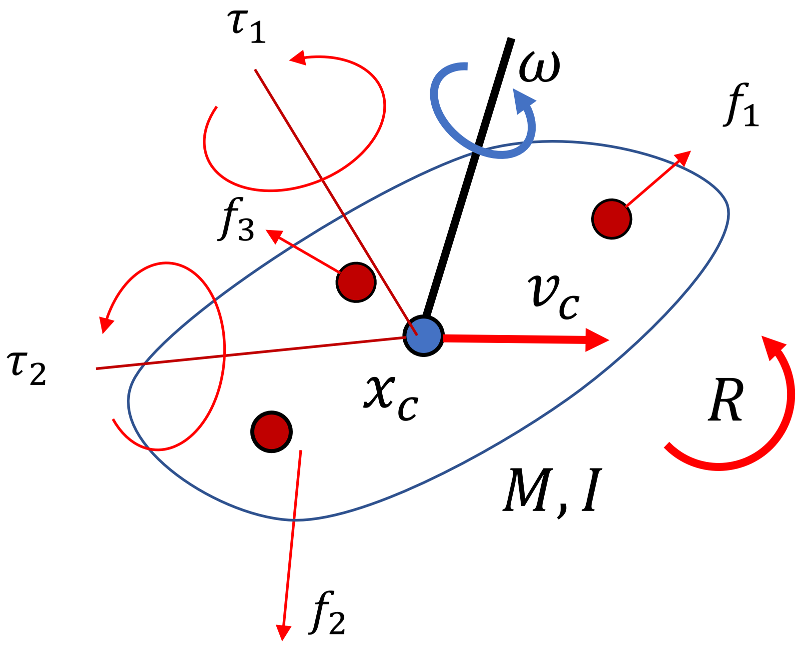
$$\tau = ???$$

Parallel Forces and Torques



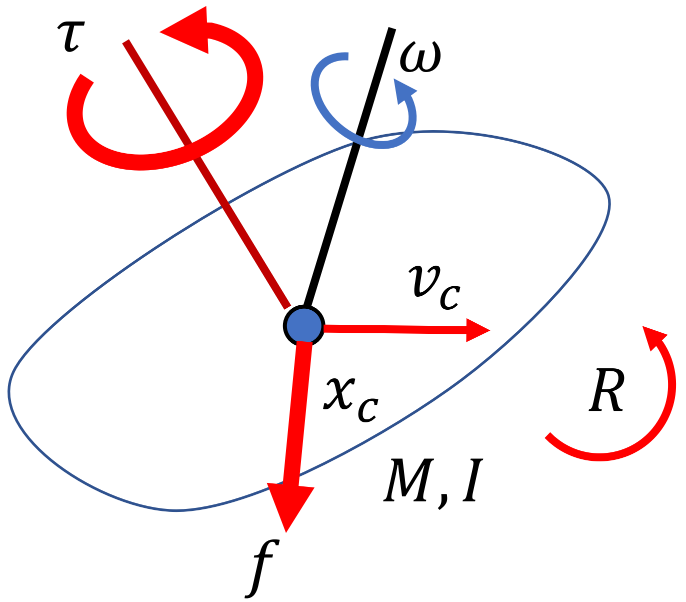
$$\tau = ???$$

Center of Momentum (CoM) Frame



$p = Mv_c$ $\qquad$ $L = I\omega$

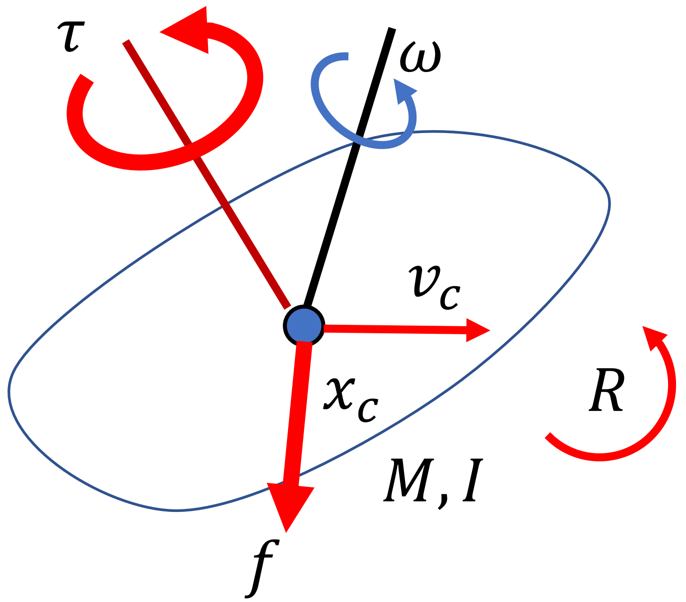
Center of Momentum (CoM) Frame



$$p = M v_c \qquad L = I\omega$$

$$f = \sum_i f_i \qquad \tau = \sum_i \tau_i$$

Equation of Motion of Rigid Body



Kinematics

m, I

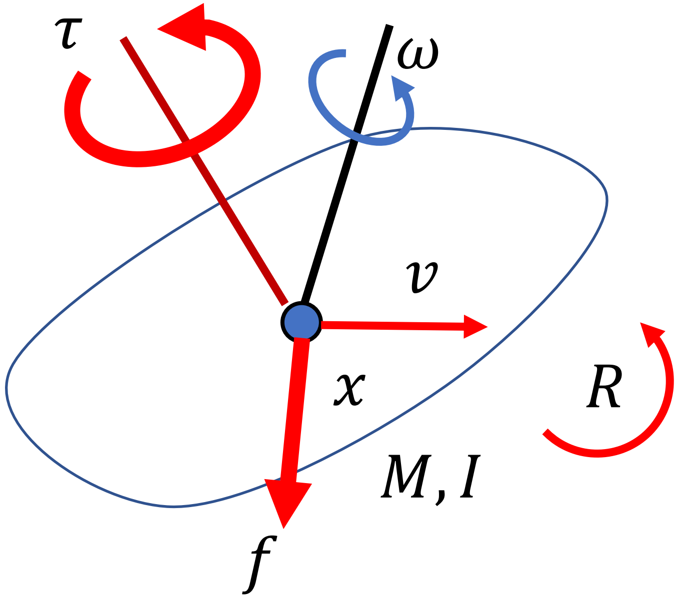
Dynamics

x, R
v, ω

$\Leftrightarrow$

p, L
f, τ

Equation of Motion of Rigid Body



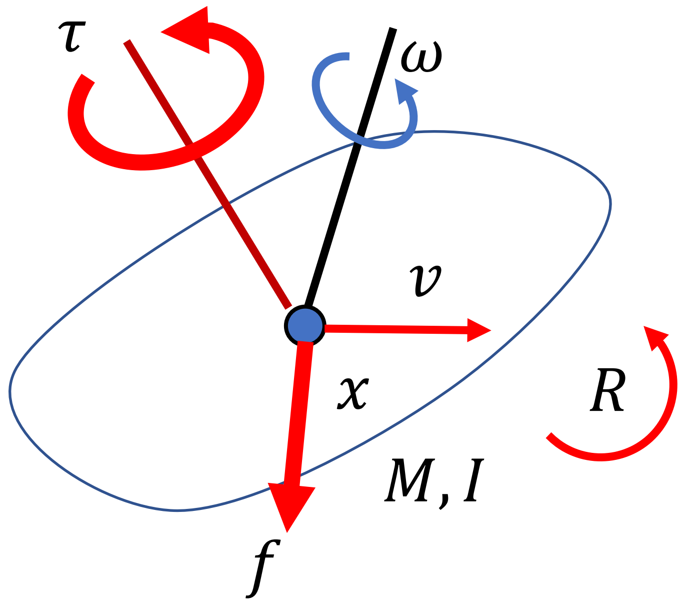
x, R, v, ω

$p = Mv$

$L = I\omega$

Newton's Second Law: $f = Ma$

Equation of Motion of Rigid Body



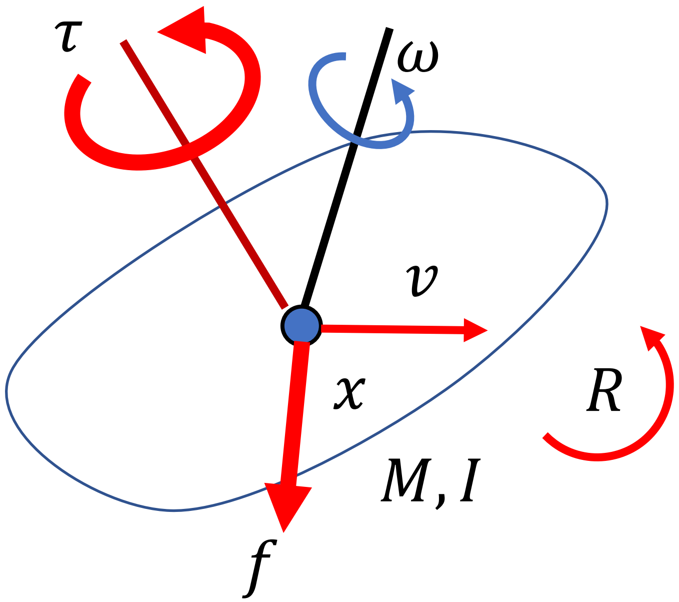
x, R, v, ω

$p = Mv$

$L = I\omega$

Newton's Second Law: $\dfrac{dp}{dt} = f$

Equation of Motion of Rigid Body



x, R, v, ω

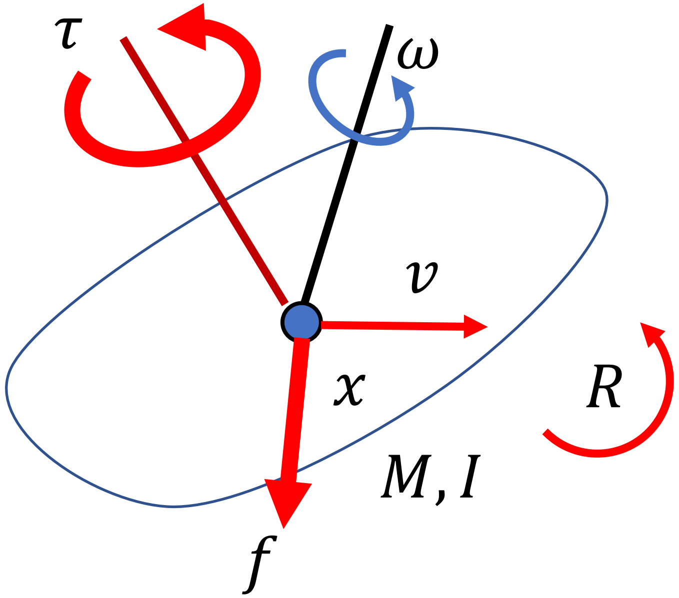
$p = Mv$

$L = I\omega$

Newton's Second Law: $\dfrac{dp}{dt} = f$

Euler's laws of motion: $\dfrac{dL}{dt} = \tau$

Equation of Motion of Rigid Body



x, R, v, ω

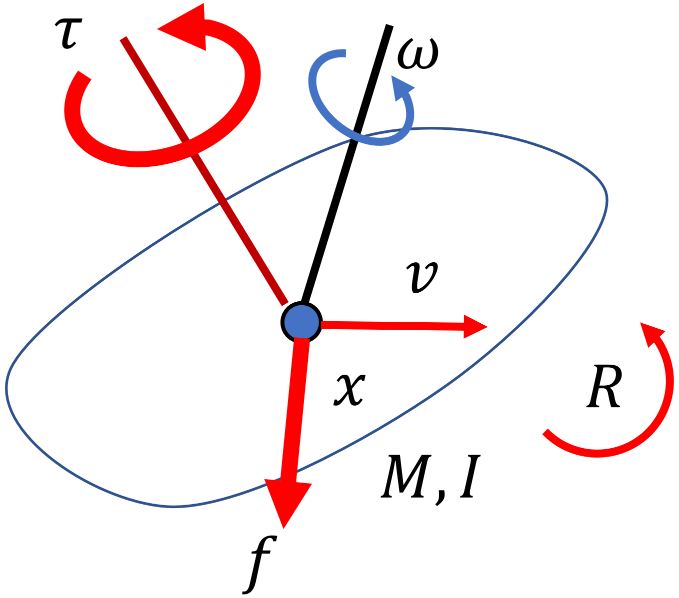
$p = Mv$

$L = I\omega$

Newton's Second Law: $\frac{dp}{dt} = f$ ➡ $M\dot{v} = f$

Euler's laws of motion: $\frac{dL}{dt} = \tau$ ➡ $I\dot{\omega} + \dot{I}\omega = \tau$

Equation of Motion of Rigid Body



x, R, v, ω

$p = Mv$

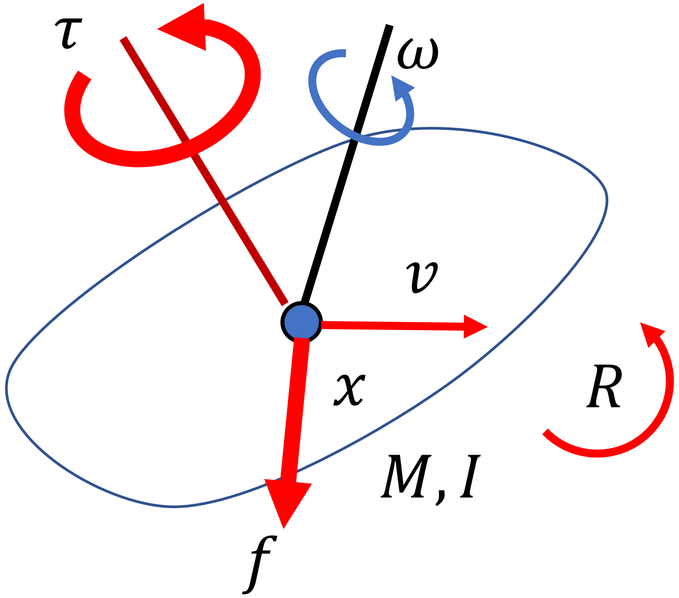
$L = I\omega$

$$\dot{I} = \frac{d}{dt}\left(RI_0R^T\right)$$
$$= \dot{R}I_0R^T + RI_0\dot{R}^T$$
$$= [\omega]_\times RI_0R^T + RI_0R^T[\omega]_\times^T$$
$$\Downarrow$$
$$\dot{I}\omega = \omega \times I\omega + I(-\omega \times \omega)$$

Newton's Second Law: $\frac{dp}{dt} = f \Rightarrow M\dot{v} = f$

Euler's laws of motion: $\frac{dL}{dt} = \tau \Rightarrow I\dot{\omega} + \omega \times I\omega = \tau$

Newton–Euler Equations



x, R, v, ω

$p = mv$

$L = I\omega$

$$\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix}$$

Numerical Integration

$$\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix}$$

$$\Downarrow$$

$$\frac{1}{h} \begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I_n \end{bmatrix} \begin{bmatrix} v_{n+1} - v_n \\ \omega_{n+1} - \omega_n \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n \times I_n\omega_n \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix}$$

Rigid Body Simulation

$$\frac{1}{h}\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I_n \end{bmatrix}\begin{bmatrix} v_{n+1} - v_n \\ \omega_{n+1} - \omega_n \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_n \times I_n \omega_n \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix}$$

$$I_n = R_n I_0 R_n^T$$

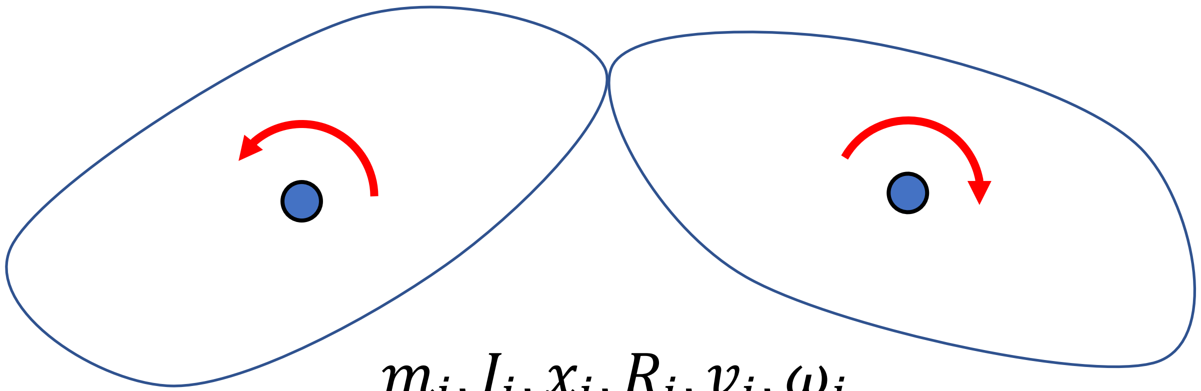
$$v_{n+1} = \cdots$$

$$\omega_{n+1} = \cdots$$

$$x_{n+1} = x_n + h v_{n+1}$$

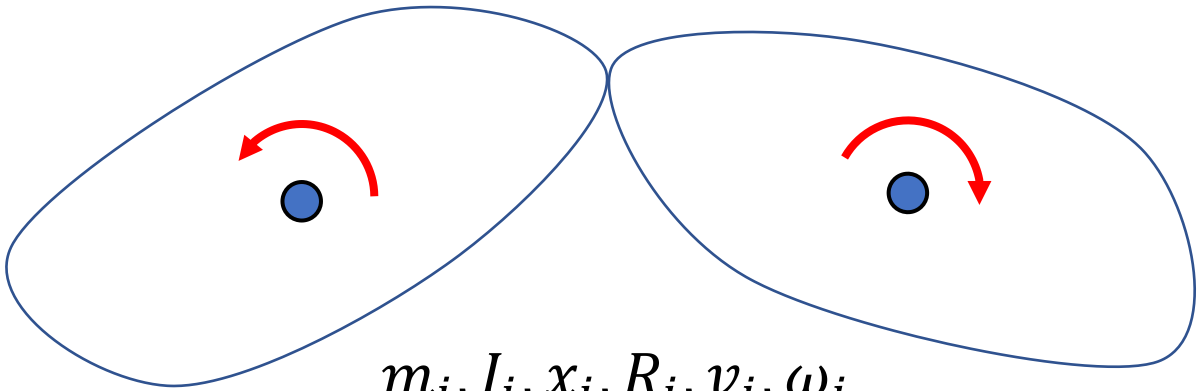
$$q_{n+1} = q_n + \frac{1}{2} h \bar{\omega}_{n+1} q$$

A System with Two Links



$m_i, I_i, x_i, R_i, v_i, \omega_i$

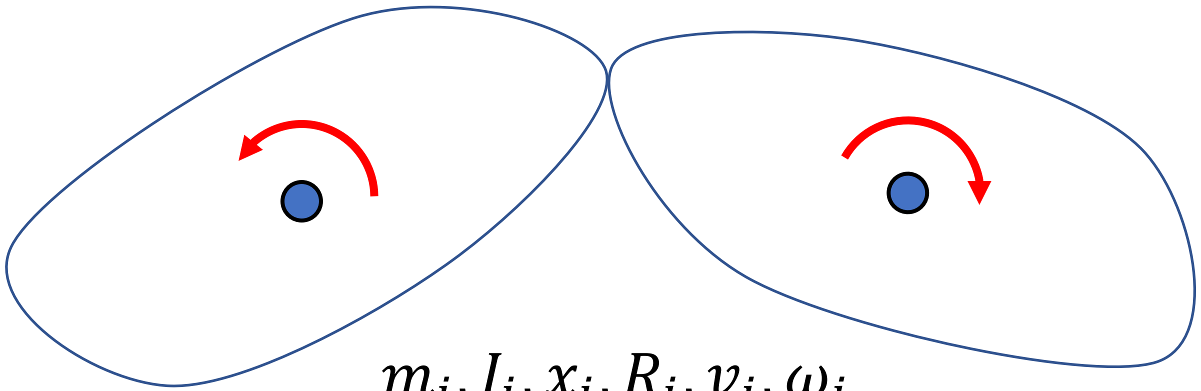
A System with Two Links



$$m_i, I_i, x_i, R_i, v_i, \omega_i$$

$$\begin{bmatrix} m_1\mathbf{I}_3 & & & \\ & I_1 & & \\ & & m_2\mathbf{I}_3 & \\ & & & I_2 \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{\omega}_1 \\ \dot{v}_2 \\ \dot{\omega}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_1 \times I_1\omega_1 \\ 0 \\ \omega_2 \times I_2\omega_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ \tau_1 \\ f_2 \\ \tau_2 \end{bmatrix}$$

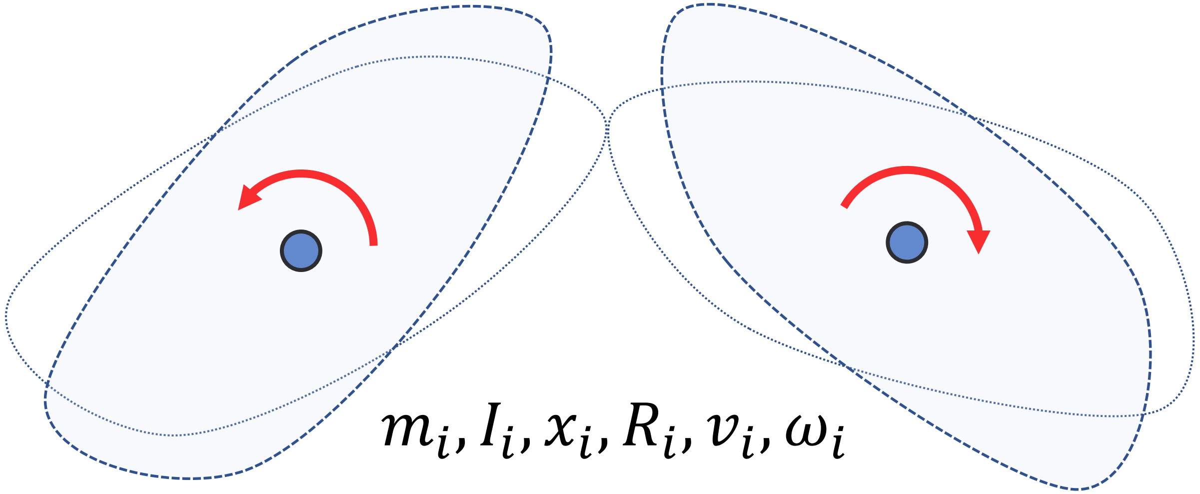
A System with Two Links



$$m_i, I_i, x_i, R_i, v_i, \omega_i$$

$$M\dot{\boldsymbol{v}} + C(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{f}$$

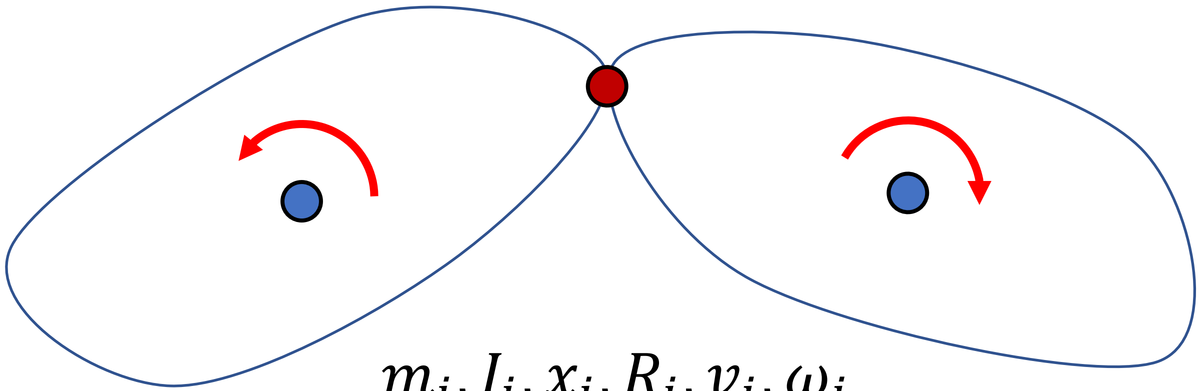
A System with Two Links



$$m_i, I_i, x_i, R_i, v_i, \omega_i$$

$$M\dot{\boldsymbol{v}} + C(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{f}$$

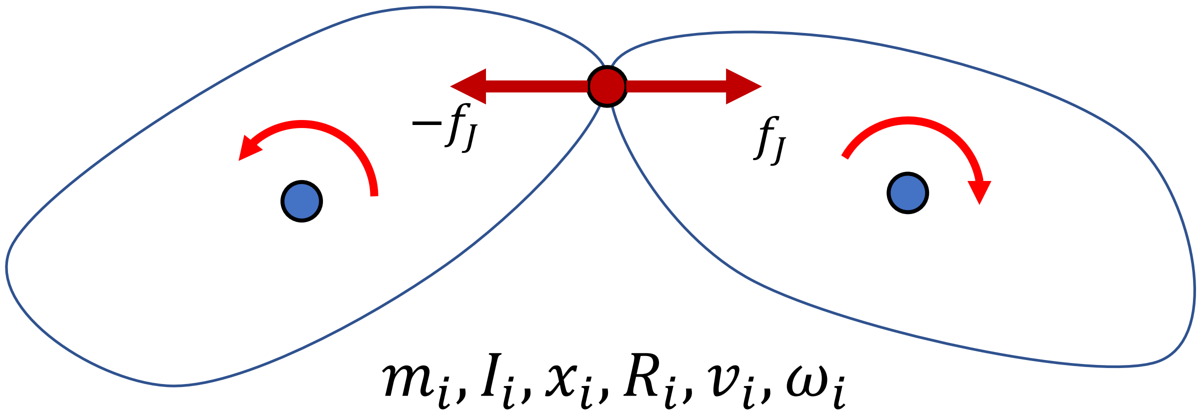
A System with Two Links and a Joint



$m_i, I_i, x_i, R_i, v_i, \omega_i$

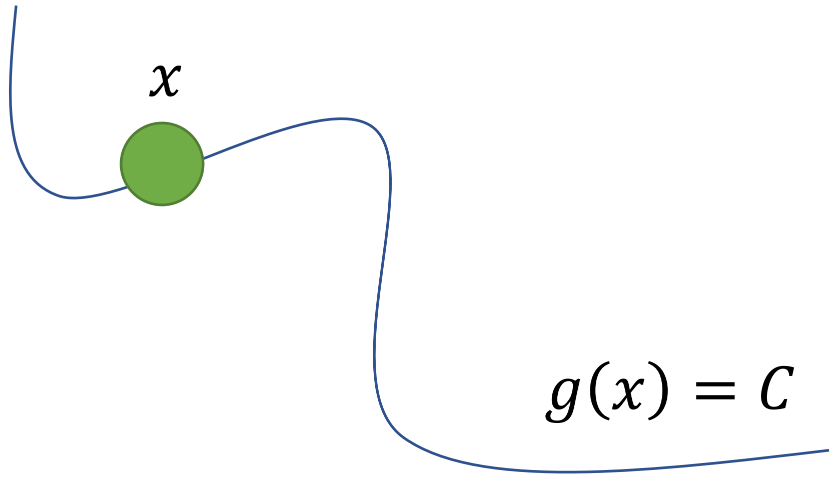
$$M\dot{\boldsymbol{v}} + C(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{f}$$

A System with Two Links and a Joint



$$M\dot{\boldsymbol{v}} + C(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{f} + \boldsymbol{f}_J$$

Constraints

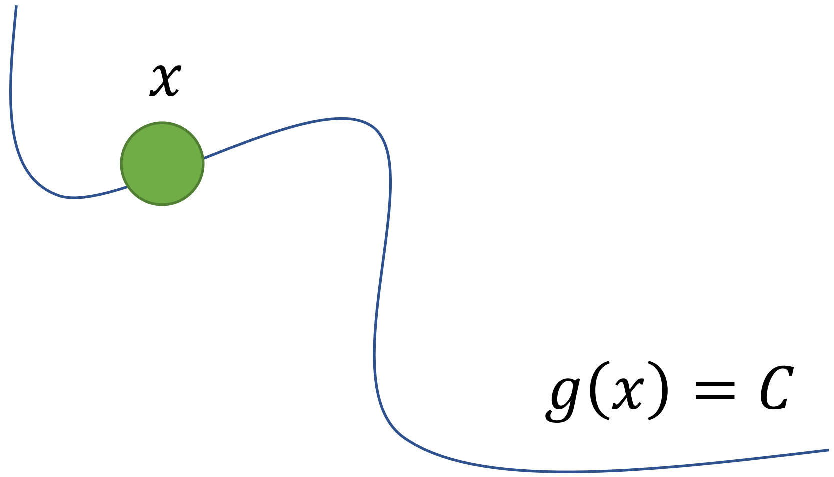


$$g(\boldsymbol{x}) = C$$

$$\Downarrow$$

$$\frac{d}{dt} g(\boldsymbol{x}) = 0$$

Constraints



$$g(\boldsymbol{x}) = C$$

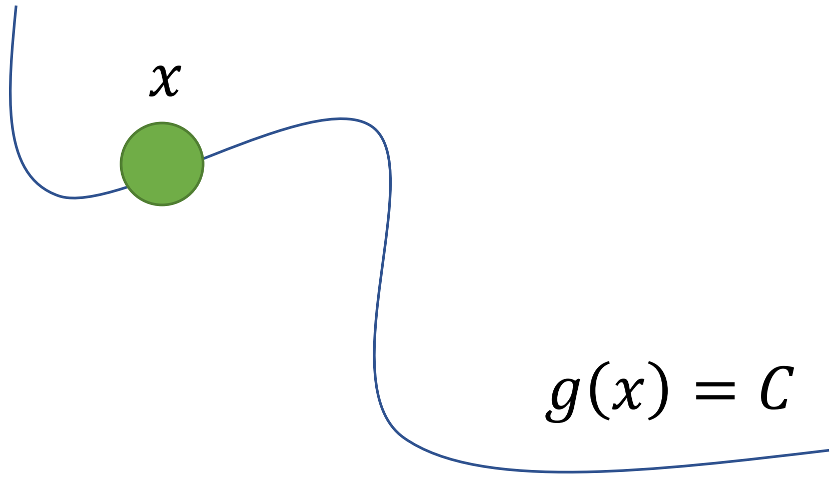
$$\Downarrow$$

$$\frac{d}{dt} g(\boldsymbol{x}) = 0$$

$$\Downarrow$$

$$\frac{\partial g}{\partial \boldsymbol{x}} \cdot \frac{d\boldsymbol{x}}{dt} = 0$$

Constraints



$$g(\boldsymbol{x}) = C$$

$$\Downarrow$$

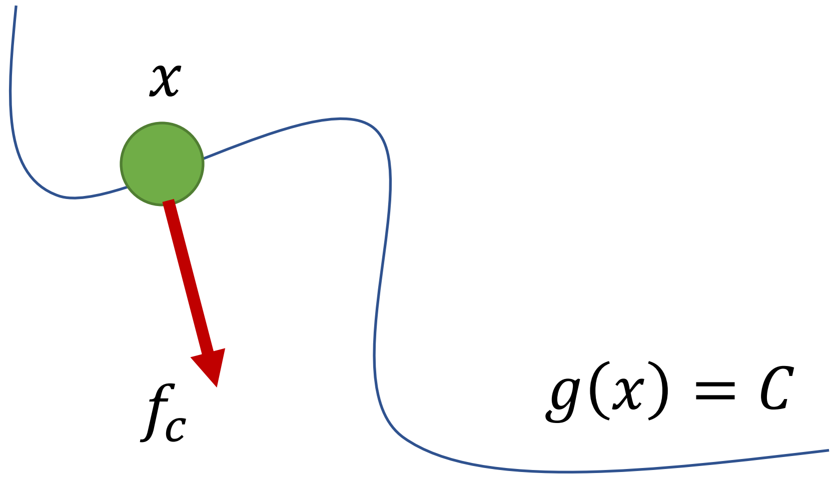
$$\frac{d}{dt} g(\boldsymbol{x}) = 0$$

$$\Downarrow$$

$$J\boldsymbol{v} = 0$$

$$J = [\nabla g]^T$$

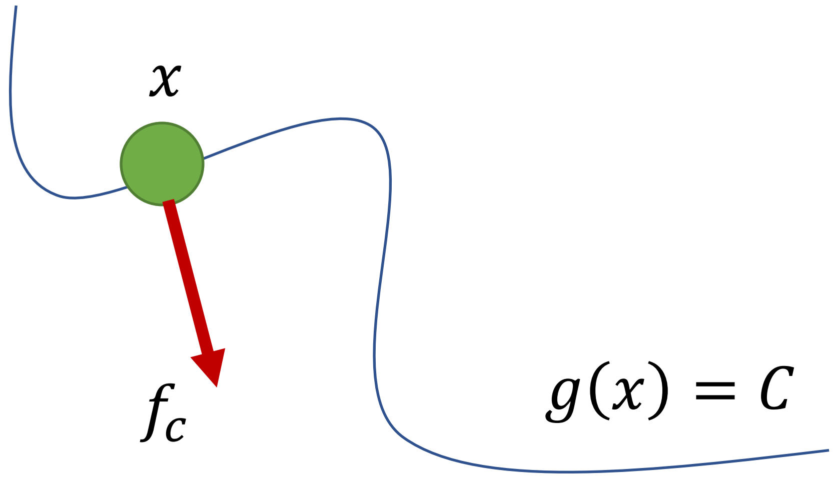
Constraint Force



* Constraint is passive
No energy gain or loss!!!

$$f_c \cdot v = 0$$

Constraint Force



* Constraint is passive
No energy gain or loss!!!

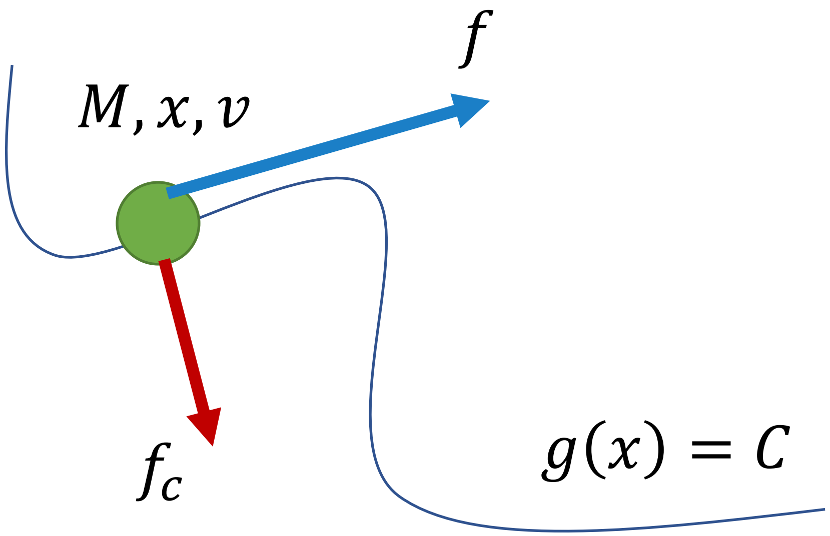
$$f_c \cdot v = 0 \Leftrightarrow f_c^T v = 0$$

$$Jv = 0$$

$$f_c = J^T \lambda$$

unknown

Equation of Motion with Constraints



$$M\dot{v} = f + J^T\lambda$$

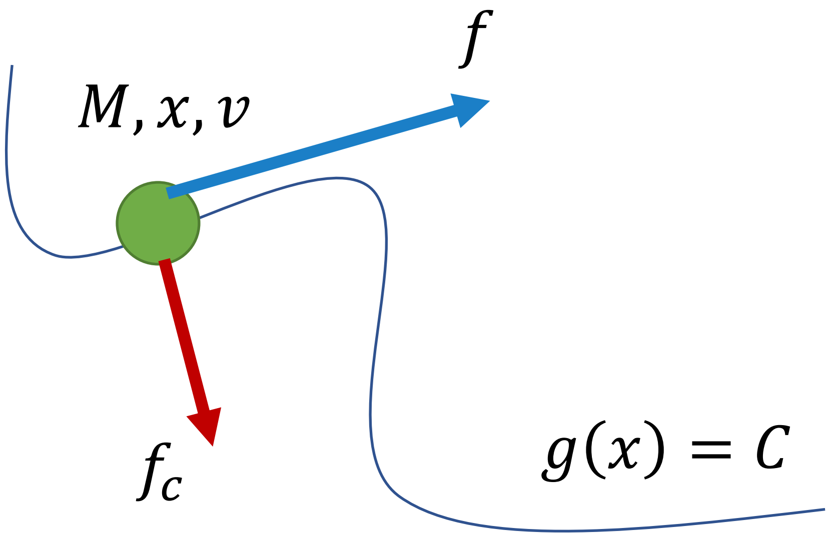
$$Jv = 0$$

$$\Downarrow$$

$$M\frac{v_{n+1} - v_n}{h} = f + J^T\lambda$$

$$Jv_{n+1} = 0$$

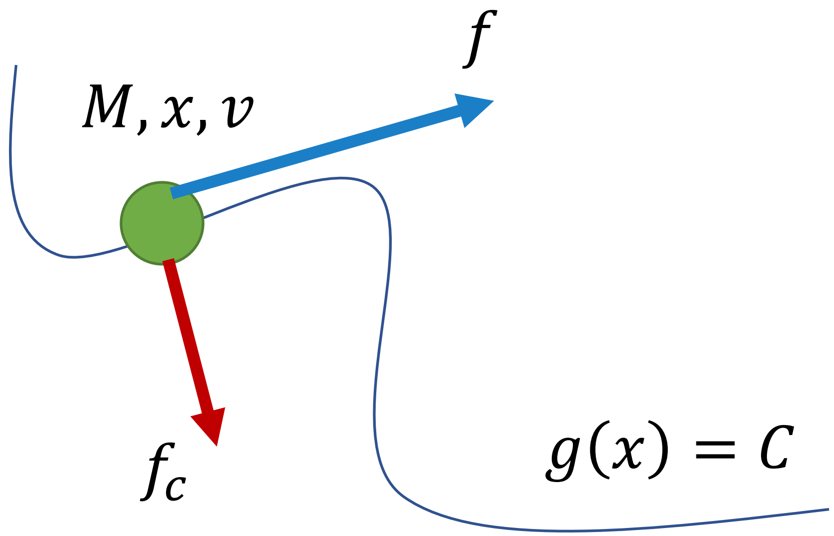
Numerical Solution



$$M\frac{v_{n+1} - v_n}{h} = f + J^T\lambda$$

$$Jv_{n+1} = 0$$

Numerical Solution



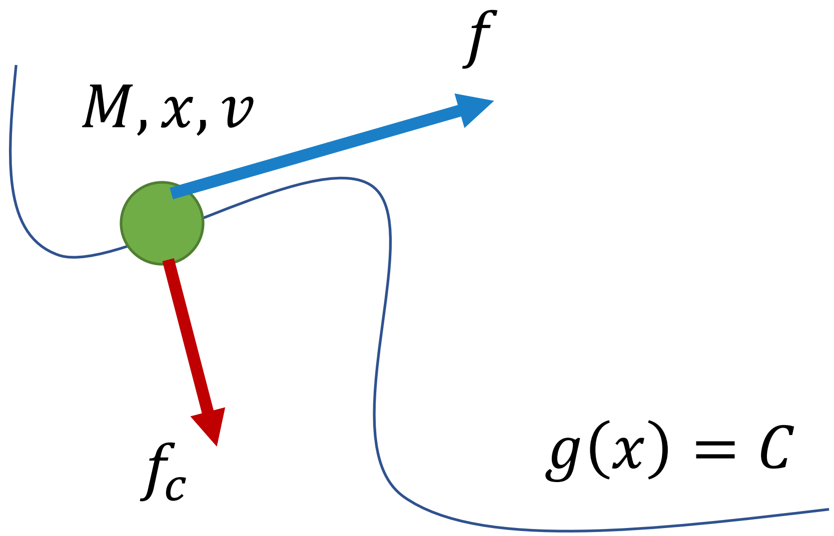
$$M\frac{v_{n+1} - v_n}{h} = f + J^T\lambda$$

$$Jv_{n+1} = \mathbf{0}$$

$$Jv_{n+1} = \alpha\frac{C - g(x_n)}{h}$$

Correction of numerical errors
α: error reduction parameter (ERP)

Numerical Solution



$$M\frac{v_{n+1} - v_n}{h} = f + J^T\lambda$$

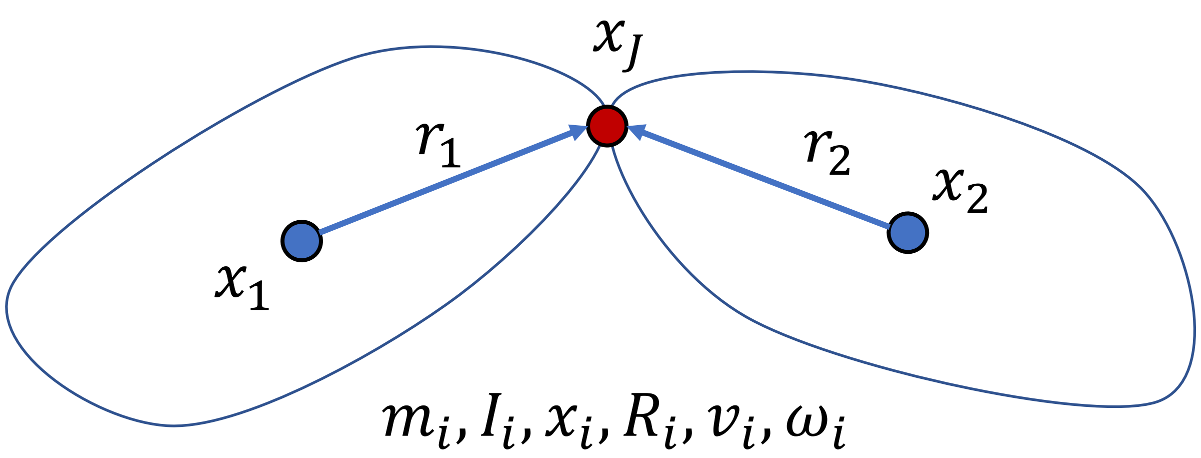
$$Jv_{n+1} = b_n$$

$$JM^{-1}J^T\lambda = c_n$$

$$(JM^{-1}J^T + \beta\mathbf{I})\lambda = c_n$$

β: constraint force mixing (CFM)

Joint Constraint

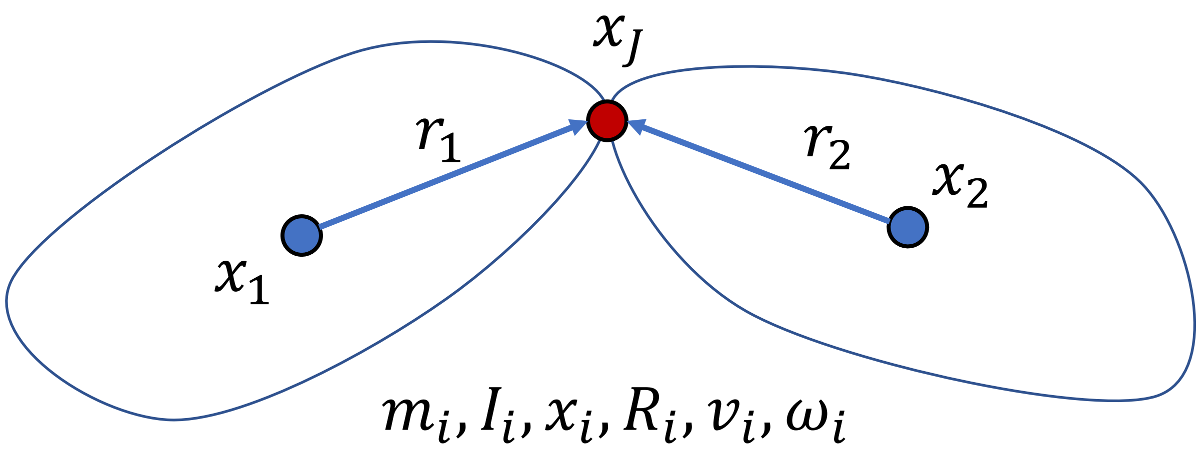


$$x_1 + R_1 r_1 = x_J = x_2 + R_2 r_2$$

d/dt

$$v_1 + \omega_1 \times r_1 = v_2 + \omega_2 \times r_2$$

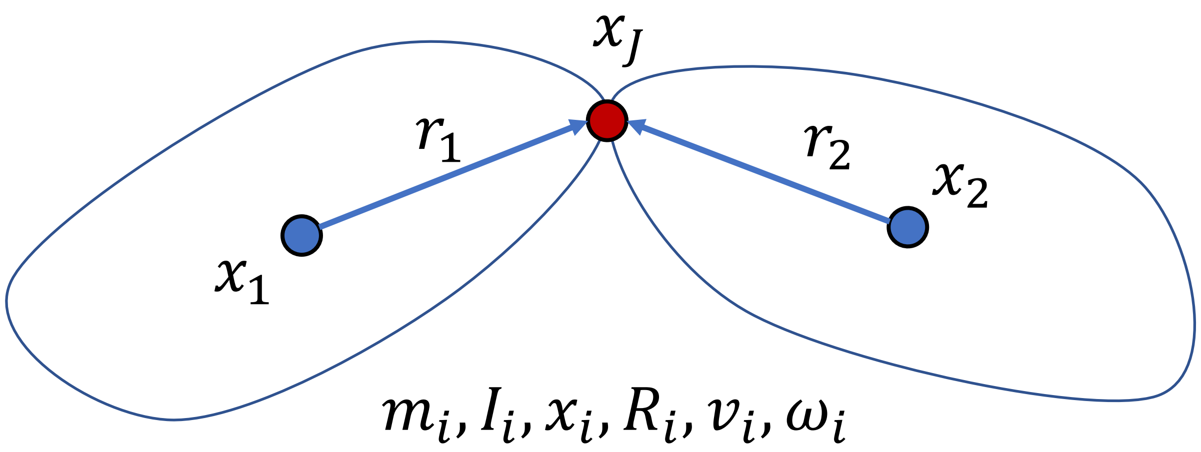
Joint Constraint



$$\begin{bmatrix} I_3 & -[r_1]_\times & -I_3 & [r_2]_\times \end{bmatrix} \begin{bmatrix} v_1 \\ w_1 \\ v_2 \\ w_2 \end{bmatrix} = 0$$

$$Jv = 0$$

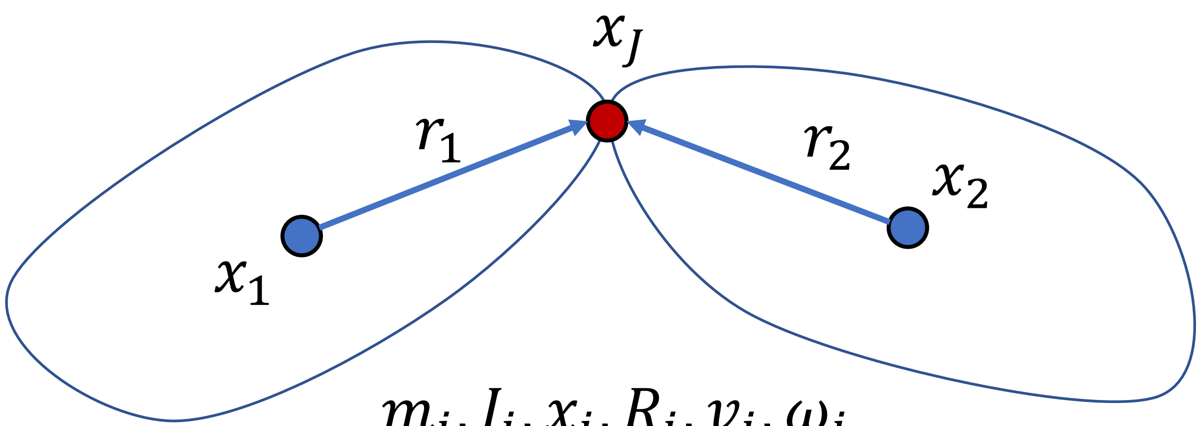
A System with Two Links and a Joint



$$M\dot{\boldsymbol{v}} + C(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{f} + \boldsymbol{J}^T\boldsymbol{\lambda}$$

$$Jv = 0$$

A System with Two Links and a Joint

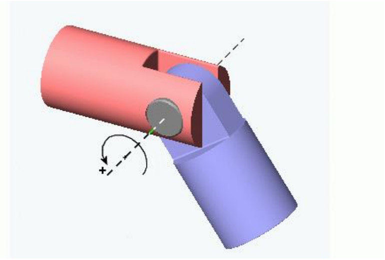


$m_i, I_i, x_i, R_i, v_i, \omega_i$

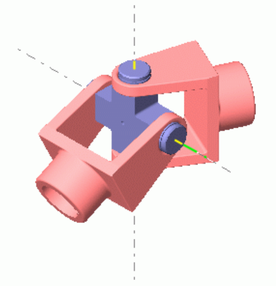
$$\begin{bmatrix} m_1\mathrm{I}_3 & & & \\ & I_1 & & \\ & & m_2\mathrm{I}_3 & \\ & & & I_2 \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{\omega}_1 \\ \dot{v}_2 \\ \dot{\omega}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_1 \times I_1\omega_1 \\ 0 \\ \omega_2 \times I_2\omega_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ \tau_1 \\ f_2 \\ \tau_2 \end{bmatrix} + \begin{bmatrix} I_3 \\ [r_1]_\times \\ -I_3 \\ -[r_2]_\times \end{bmatrix} \lambda$$

$$Jv = 0$$

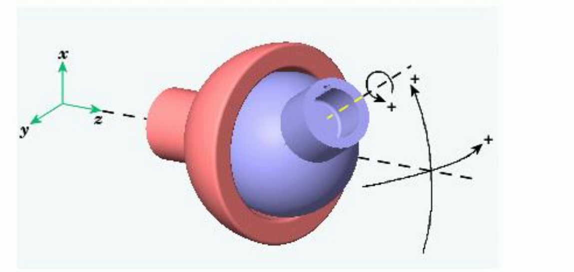
Different Types of Joints



Hinge joint
Revolute joint



Universal joint

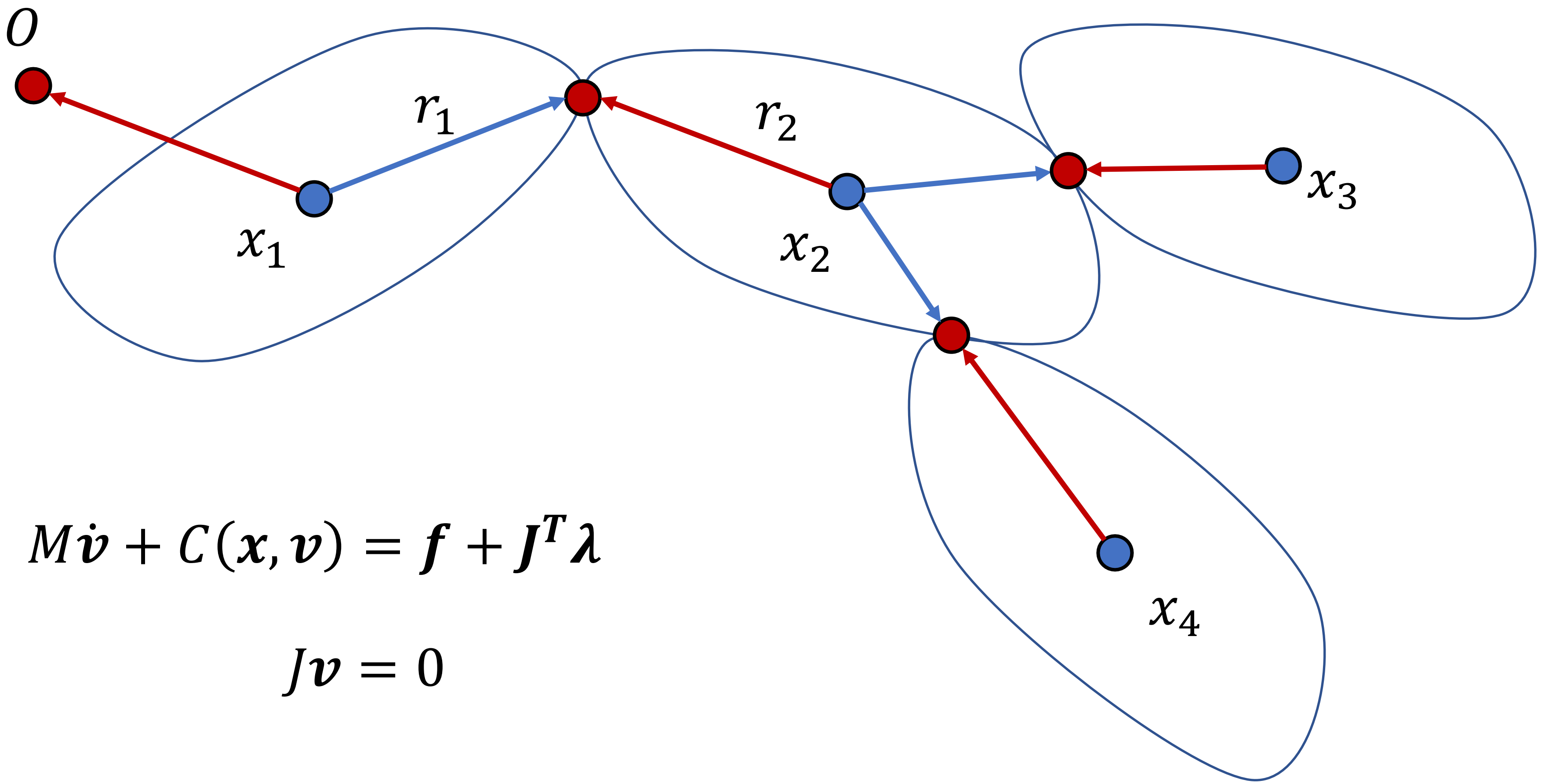


Ball-and-socket

$$\begin{bmatrix} I_3 & -[r_1]_\times & -I_3 & [r_2]_\times \\ ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} v_1 \\ w_1 \\ v_2 \\ w_2 \end{bmatrix} = 0$$

A System with Many Links Joints

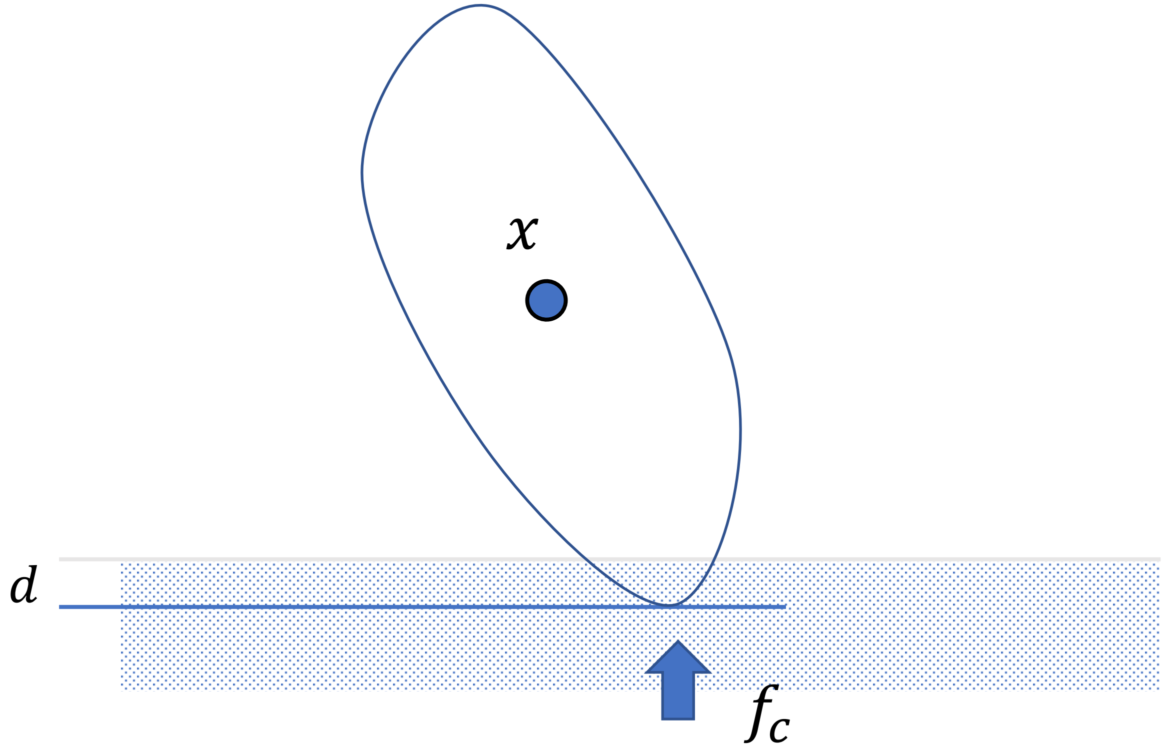
$$m_i, I_i, x_i, R_i, v_i, \omega_i$$



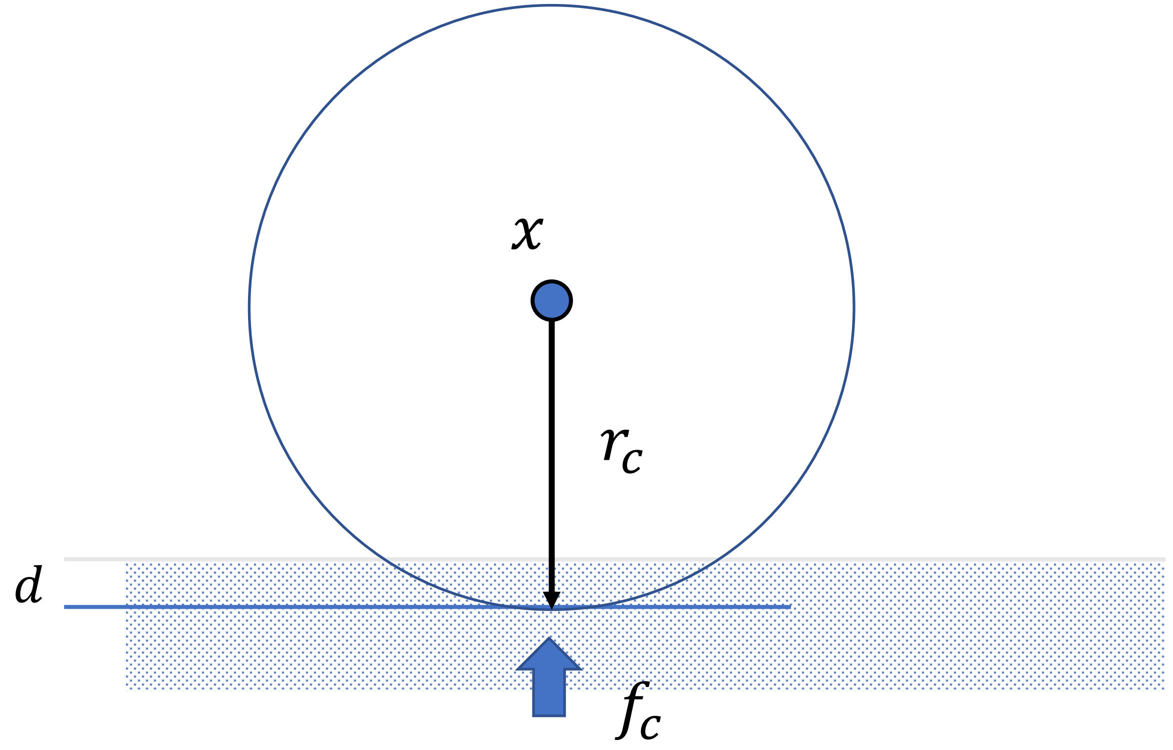
$$M\dot{\boldsymbol{v}} + C(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{f} + \boldsymbol{J}^T\boldsymbol{\lambda}$$

$$J\boldsymbol{v} = 0$$

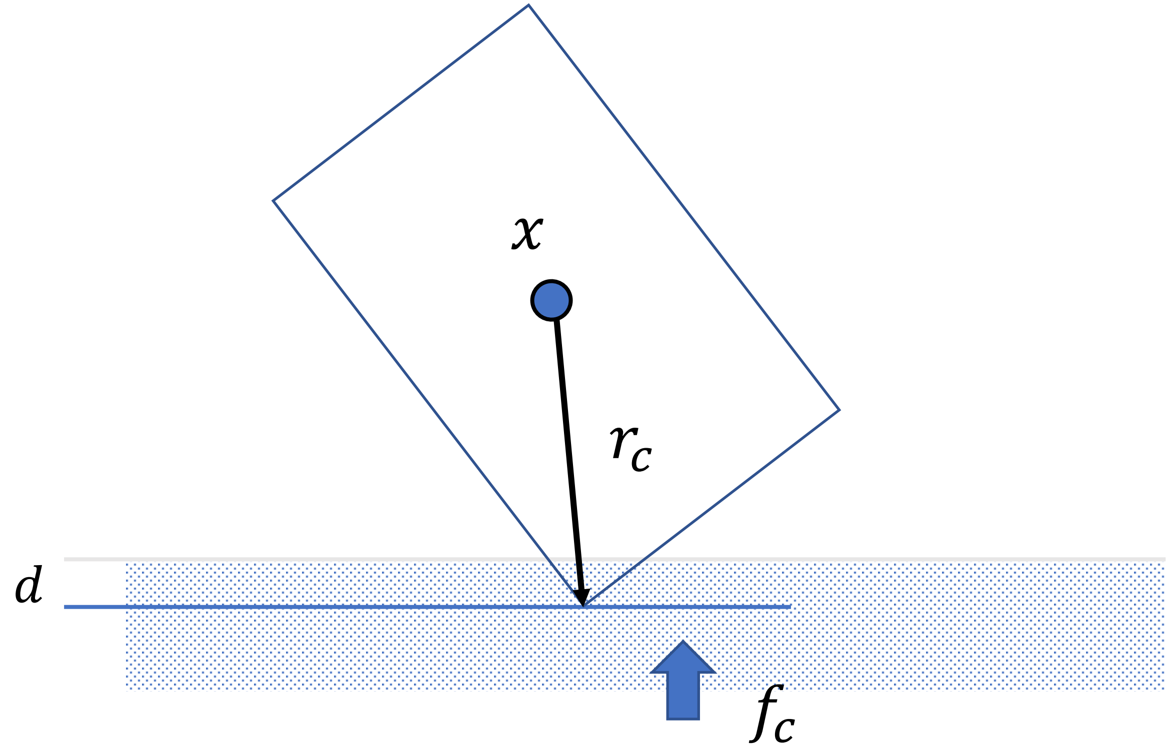
Contacts



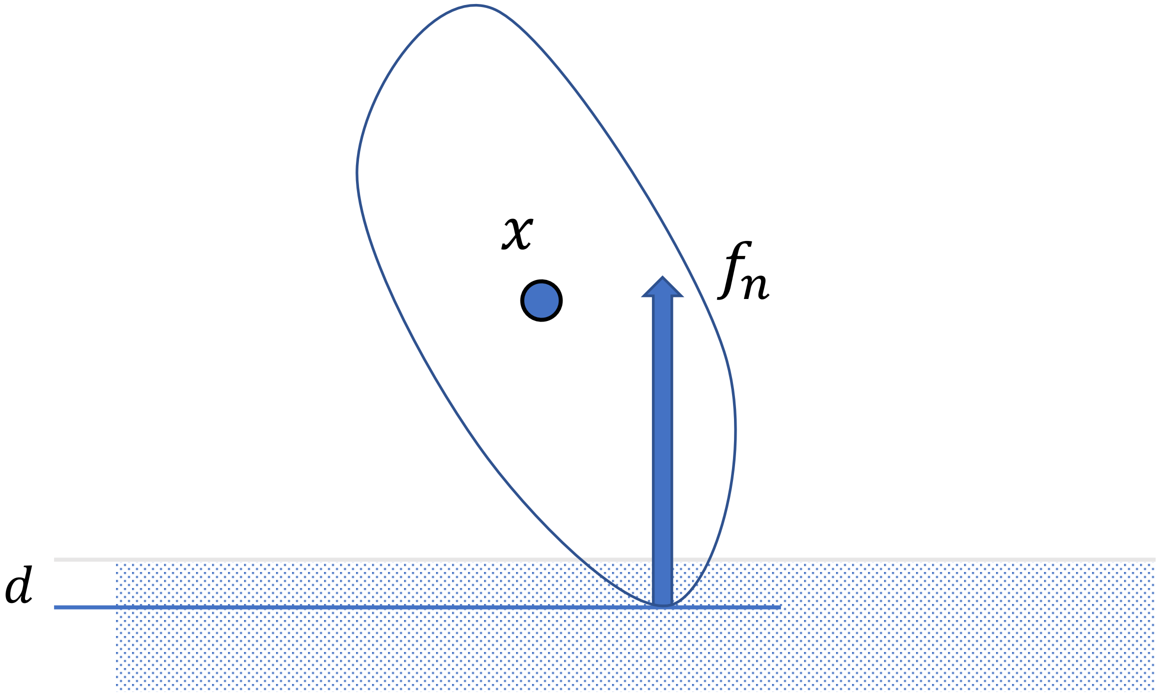
Contact Detection



Contact Detection

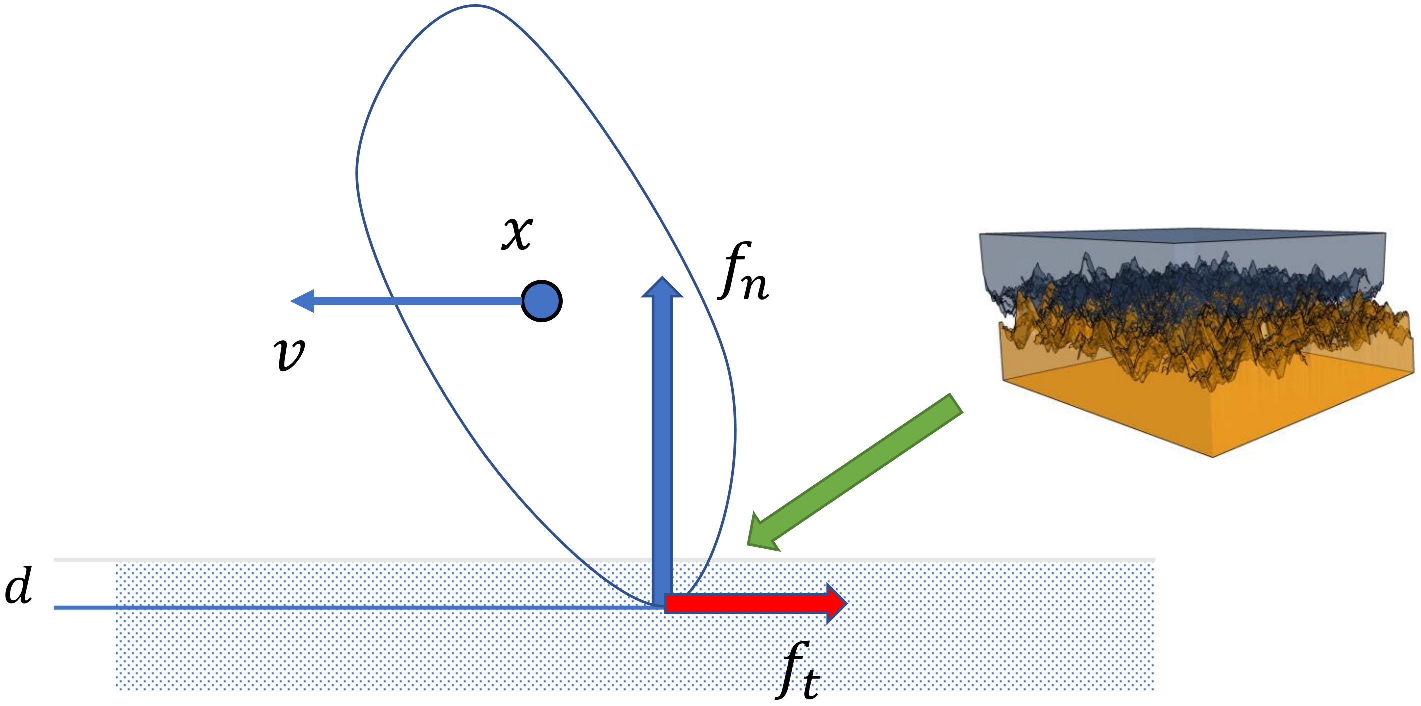


Penalty-based Contact Model



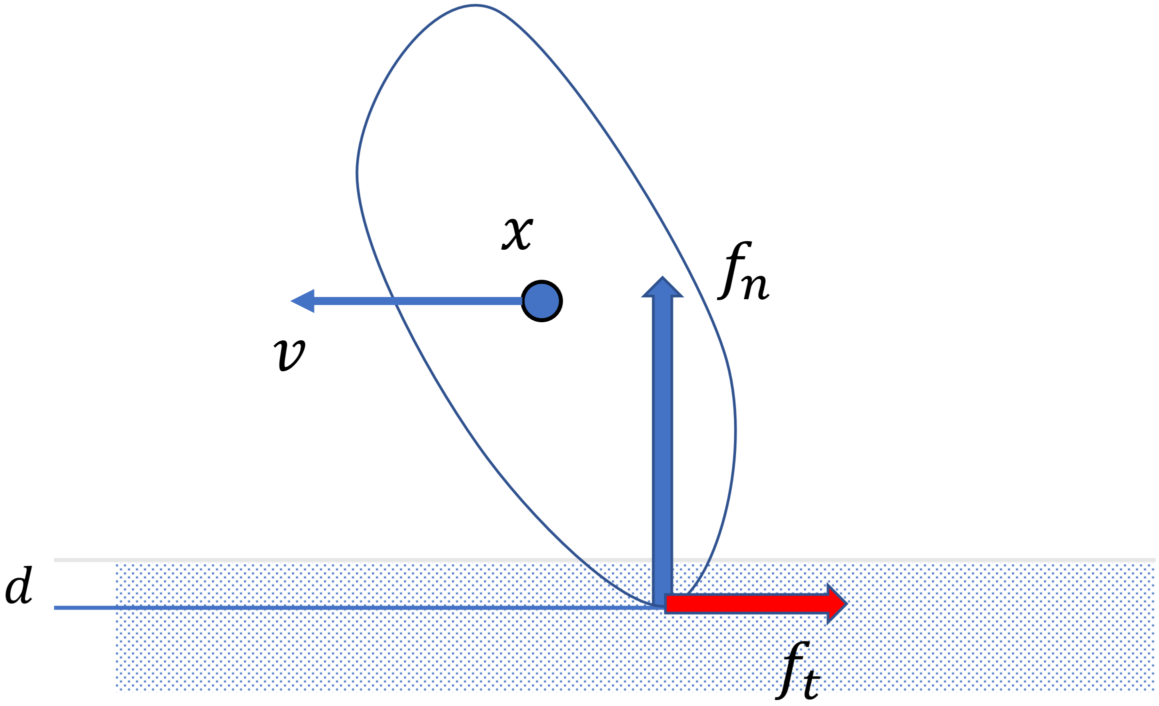
$$f_n = -k_p d - k_d v_{c,\perp}$$

Frictional Contact



Coulomb's law of friction: $|f_t| = \mu f_n$

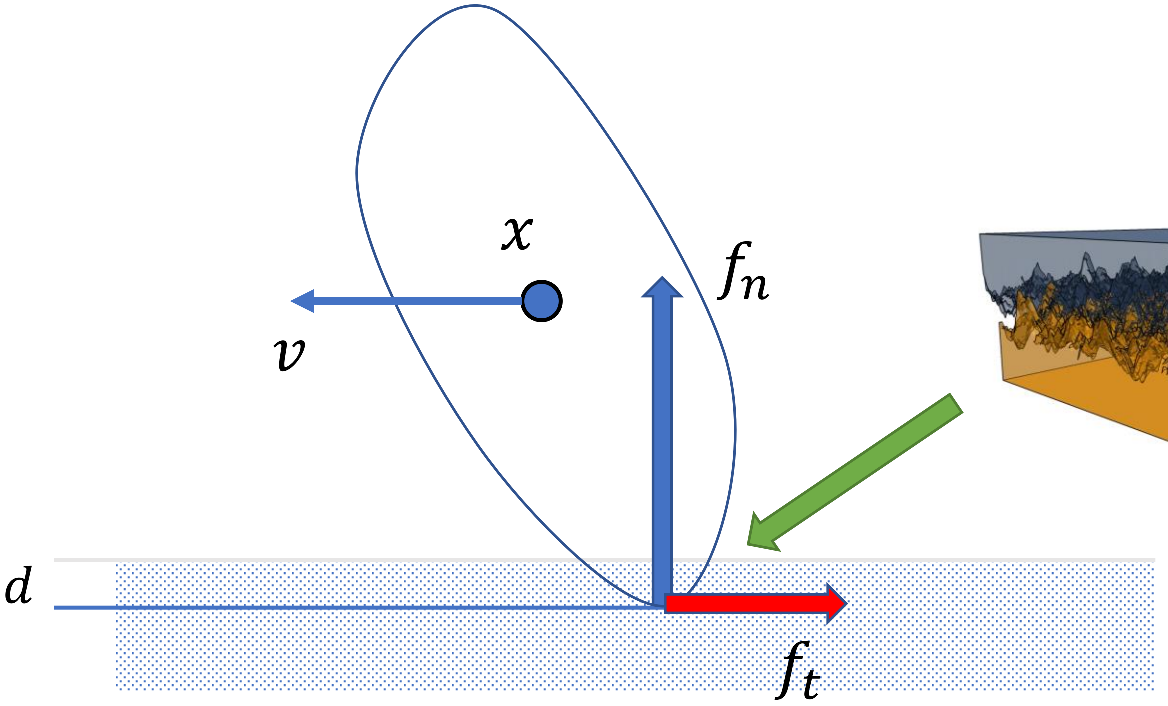
Frictional Contact

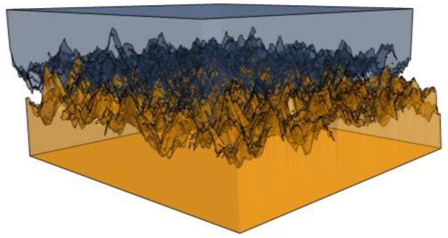


$$f_n = -k_p d - k_d v_{c,\perp}$$

$$f_t = -\mu f_n \frac{v_{c,\parallel}}{\left\| v_{c,\parallel} \right\|}$$

Frictional Contact

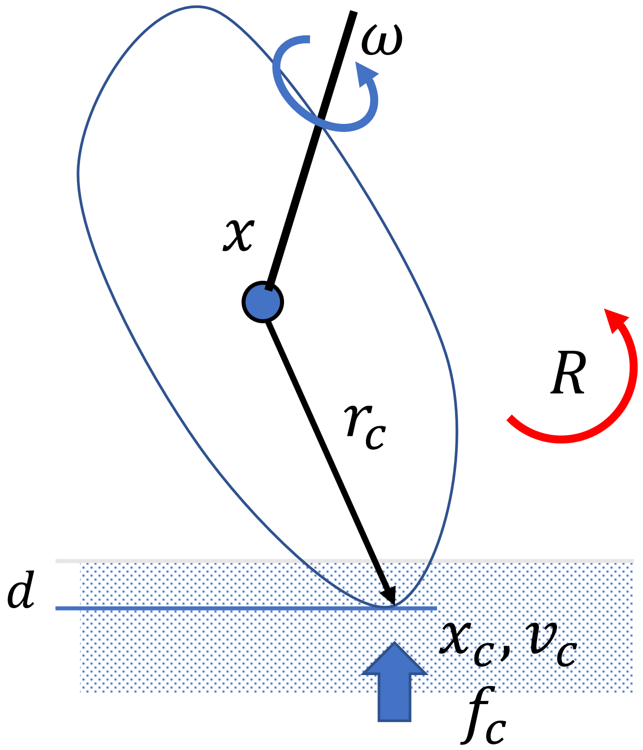




Coulomb's law of friction: $|f_t| \leq \mu f_n$

How to model static friction???

Contact as a Constraint

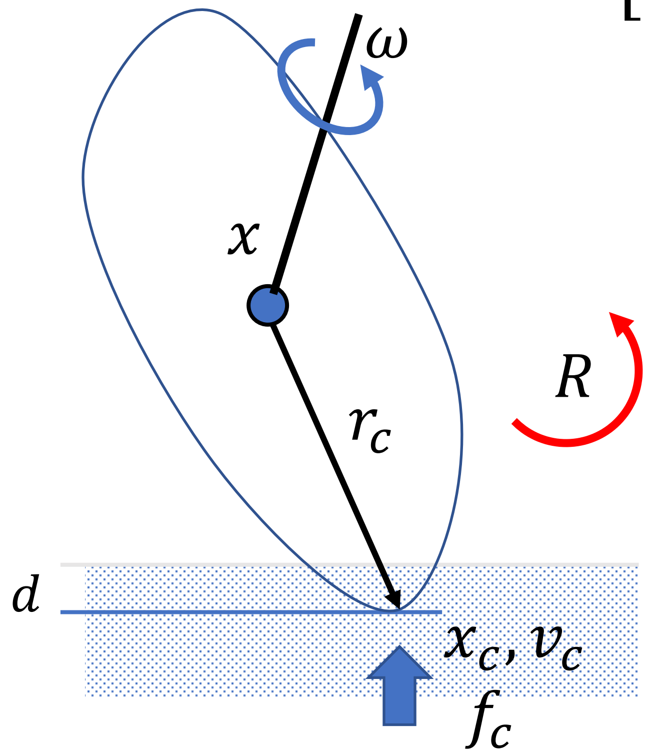


$$x_c = x + r_c$$

$$v_c = v + \omega \times r_c = J_c \begin{bmatrix} v \\ \omega \end{bmatrix}$$

$$v_{c,\perp} = v + \omega \times r_c = J_{c,\perp} \begin{bmatrix} v \\ \omega \end{bmatrix}$$

Contact as a Constraint

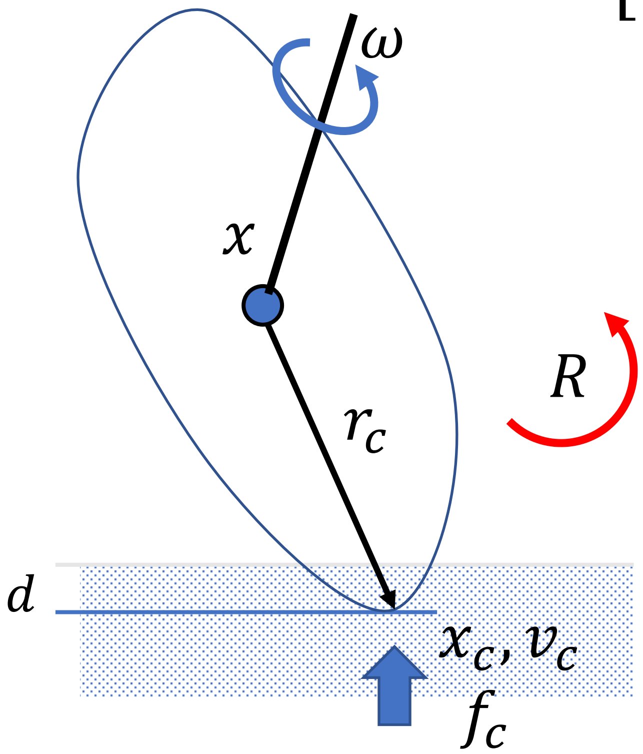


$$\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix} + J_c^T \lambda$$

$$v_c = J_c \begin{bmatrix} v \\ \omega \end{bmatrix} \geq 0$$

$$\lambda \geq 0$$

Contact as a Constraint



$$\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix} + J_c^T \lambda$$

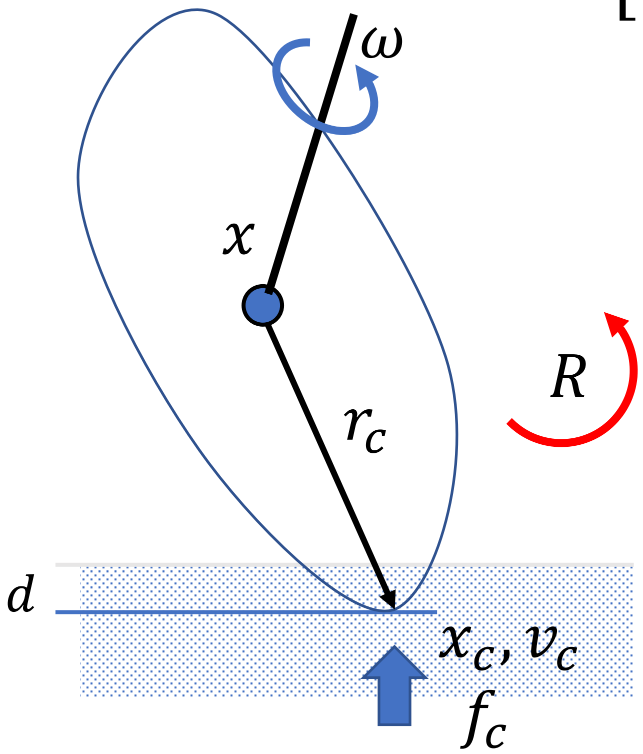
$$v_c = J_c \begin{bmatrix} v \\ \omega \end{bmatrix} \geq 0$$

$$\lambda \geq 0$$

$$v_c > 0 \Rightarrow \lambda = 0$$

$$\lambda > 0 \Rightarrow v_c = 0$$

Contact as a Linear Complementary Problem



$$\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix} + J_c^T \lambda$$

$$v_c = J_c \begin{bmatrix} v \\ \omega \end{bmatrix} \geq 0$$

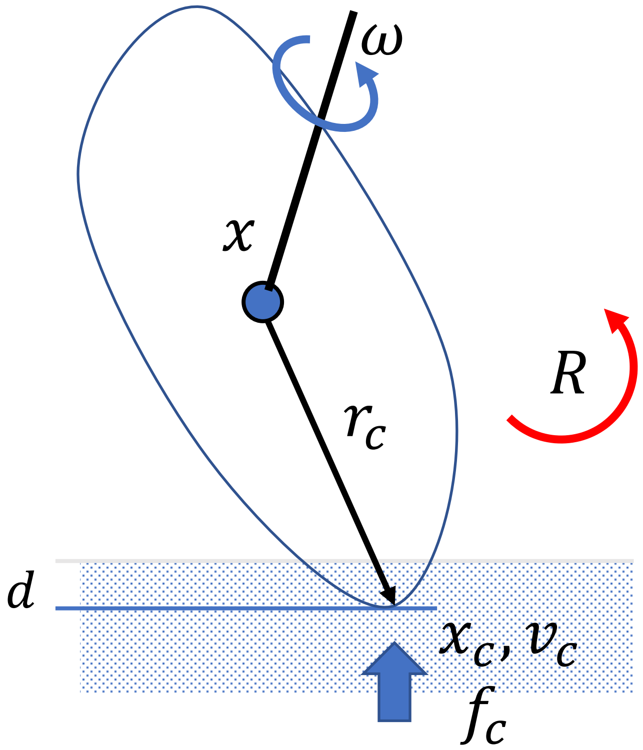
$$\lambda \geq 0$$

$$v_c \perp \lambda = 0$$

(Mixed) Linear Complementary Problem (LCP)

To solve an LCP:
e.g. Lemke's algorithm – a simplex algorithm

Contact as a Linear Complementary Problem



How to deal the friction?

Computer Graphics Proceedings, Annual Conference Series, 1994

Fast Contact Force Computation for Nonpenetrating Rigid Bodies

David Baraff

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

A new algorithm for computing contact forces between solid objects with friction is presented. The algorithm allows a mix of contact points with static and dynamic friction. In contrast to previous approaches, the problem of computing contact forces is not transformed into an optimization problem. Because of this, the need for sophisticated optimization software packages is eliminated. For both systems with and without friction, the algorithm has proven to be considerably faster, simpler, and more reliable than previous approaches to the problem. In particular, implementation of the algorithm by nonspecialists in numerical programming is quite feasible.

1. Introduction

In recent work, we have established the viability of using analytical methods to simulate rigid body motion with contact[1,2,3]. In situations involving only bilateral constraints (commonly referred to as "equality constraints"), analytical methods require solving systems of simultaneous linear equations. Bilateral constraints typically arise in representing idealized geometric connections such as universal joints, point-to-surface constraints etc. For systems with contact, unilateral (or "inequality") constraints are required to prevent adjoining bodies from interpenetrating. In turn, the simultaenous linear equations arising from a system of only bilateral constraints must be augmented to reflect the unilateral constraints; the result is in general an inequality-constrained nonlinear minimization problem.

However, analytical techniques for systems with contact have yet to really catch on in the graphics/simulation community. We believe that this is because of the perceived practical and theoretical complexities of using analytical techniques in systems with contact. This paper has two goals, one of which is to address these concerns: in particular, we present analytical methods for systems with contact that can be practically implemented by those of us (such as the author) who *are not specialists in numerical analysis or optimization*. These methods are simpler, reliable, and faster than previous methods used for either systems with friction, or systems without friction.

Our other goal is to extend and improve previous algorithms for computing contact forces with friction[3]. We present a simple, fast algorithm for computing contact forces with friction. The restriction of our algorithm to the frictionless case is equivalent to an algorithm described in Cottle and Dantzig[4] (but attributed to Dantzig) for solving linear complementarity problems. It is not our intention to reinvent the wheel; however, it is necessary to first understand Dantzig's algorithm and why it works for our frictionless sytems before going on to consider the more general solution algorithm we propose to deal with friction. We give a physical motivation for Dantzig's algorithm and discuss its properties and implementation in section 4. For frictionless systems, our implementation of Dantzig's algorithm compares very favorably with the use of large-scale, sophisticated numerical optimization packages cited by previous systems[11,7,8,6]. In particular, for a system with n unilateral constraints, our implementation tends to require approximately three times the work required to solve a square linear system of size n using Gaussian elimination. Most importantly, Dantzig's algorithm, and our extensions to it for systems with friction, are sufficiently simple that nonspecialists in numerical programming can implement them on their own; this is most assuredly *not* true of the previously cited large-scale optimization packages.

Interactive systems with bilateral constraints are common, and there is no reason that moderately complicated interactive simulation with collision and contact cannot be achieved as well. We strongly believe that using our algorithms, interactive simulations with contact and friction are practical. We support this claim by demonstrating the first known system for interactive simulations involving contact and a correct model of Coulomb friction.

2. Background and Motivation

Lötstedt[10] represents the first attempt to compute friction forces in an analytical setting, by using quadratic programming to compute friction forces based on a simplification of the Coulomb friction model. Baraff[3] also proposed analytical methods for dealing with friction forces and presents algorithms that deal with dynamic friction (also known as sliding friction) and static friction (also known as dry friction). The results for dynamic friction were the more comprehensive of the two, and the paper readily acknowledges that the method presented for computing contact forces with static friction (a Gauss-Seidel-like iterative procedure) was not very reliable. The method also required an approximation for three-dimensional systems (but not for planar systems) that resulted in anisotropic friction. Finally, the results presented did not fully exploit earlier discoveries concerning systems with only dynamic friction, and no static friction.

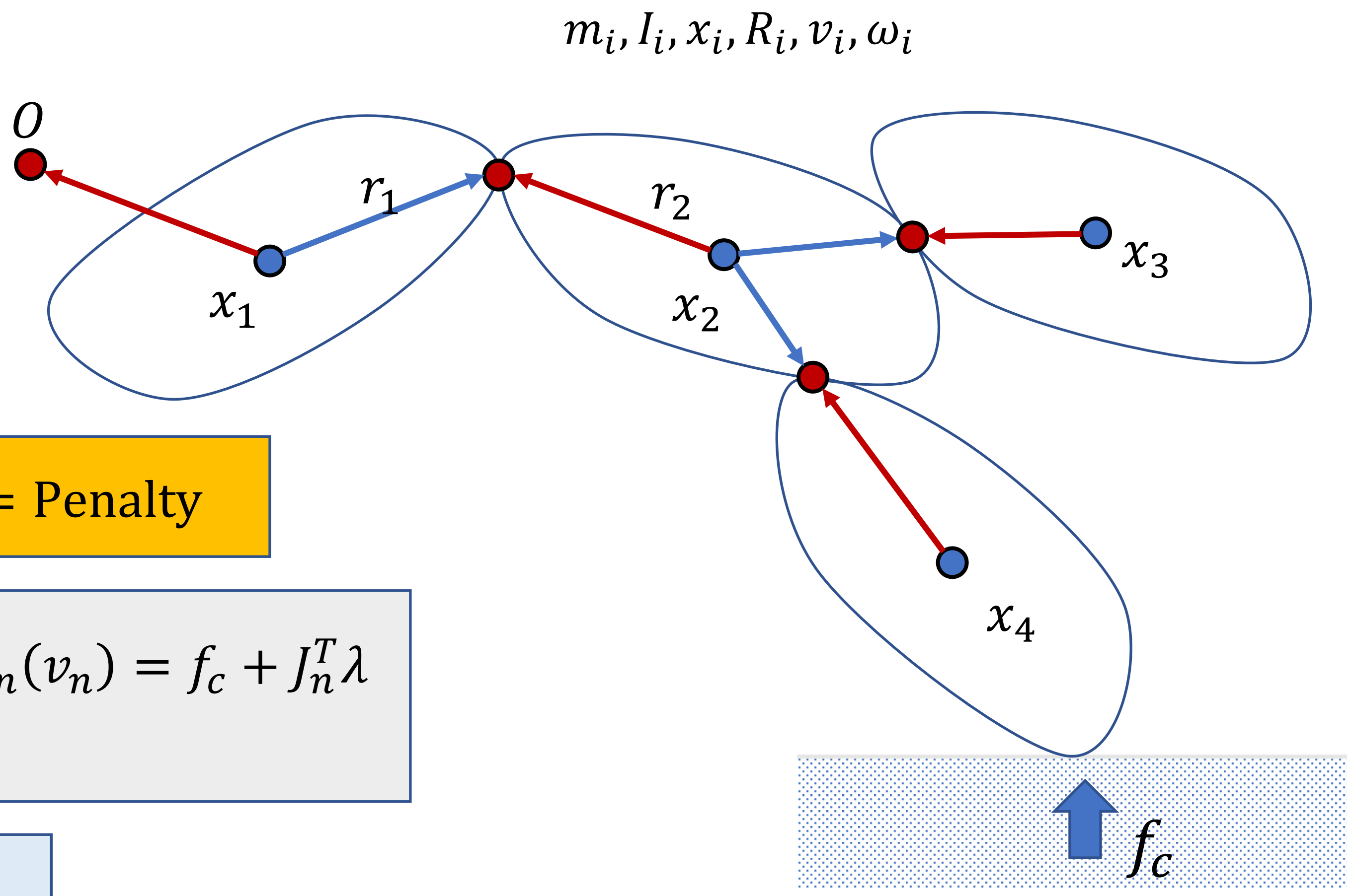
In this paper, we present a method for computing contact forces with both dynamic and static friction that is considerably more robust than previous methods. Our method requires no approximations for three-dimensional systems, and is much simpler and faster than previous methods. We were extremely surprised to find that our implementation of the method, applied to frictionless systems, was a large improvement compared with the use of large-scale optimization software packages, both in terms of speed and, especially,

SIGGRAPH '94, July 24-29, Orlando, Florida
© ACM 1994 ISBN: 0-89791-667-0 ...$5.00

23

David Baraff. SIGGRAPH '94
Fast contact force computation for nonpenetrating rigid bodies.

Simulation of a Rigid Body System



$I_n = R_n I_0 R_n^T$

$f_c = \text{Penalty}$

$$M_n(v_{n+1} - v_n)/h + C_n(v_n) = f_c + J_n^T \lambda$$

$$J_n v_{n+1} = c_n$$

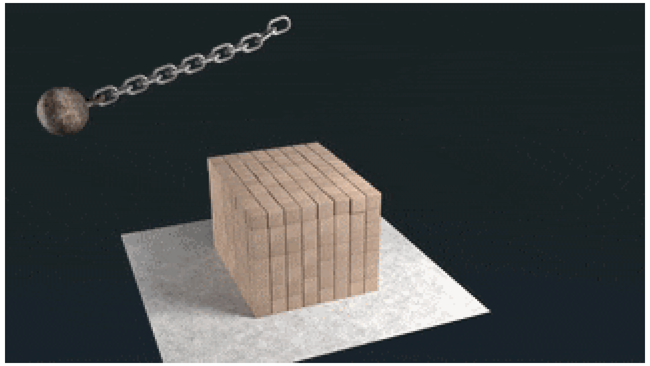
$$x_{n+1} = x_n + h v_{n+1}$$

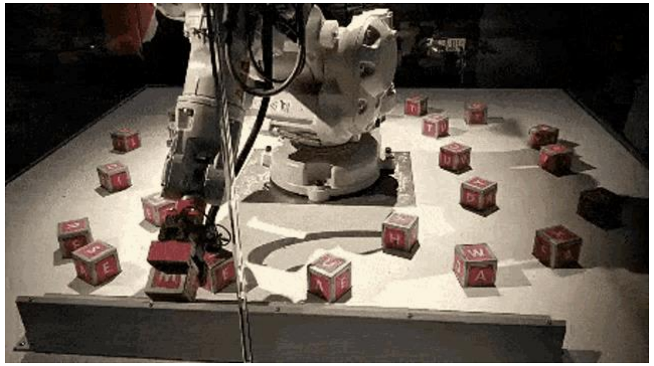
$$q_{n+1} = q_n + \frac{h}{2} \bar{\omega}_{n+1} q$$

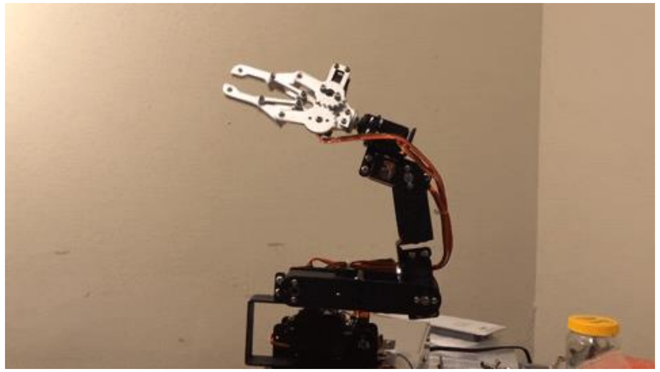
Outline

- Simulation Basis
  - Numerical Integration: Euler methods
- Equations of Rigid Bodies
  - Rigid Body Kinematics
  - Newton-Euler equations
- Articulated Rigid Bodies
  - Joints and constraints
- Contact Models
  - Penalty-based contact
  - Constraint-based contact

https://www.cs.cmu.edu/~baraff/sigcourse/







Questions?