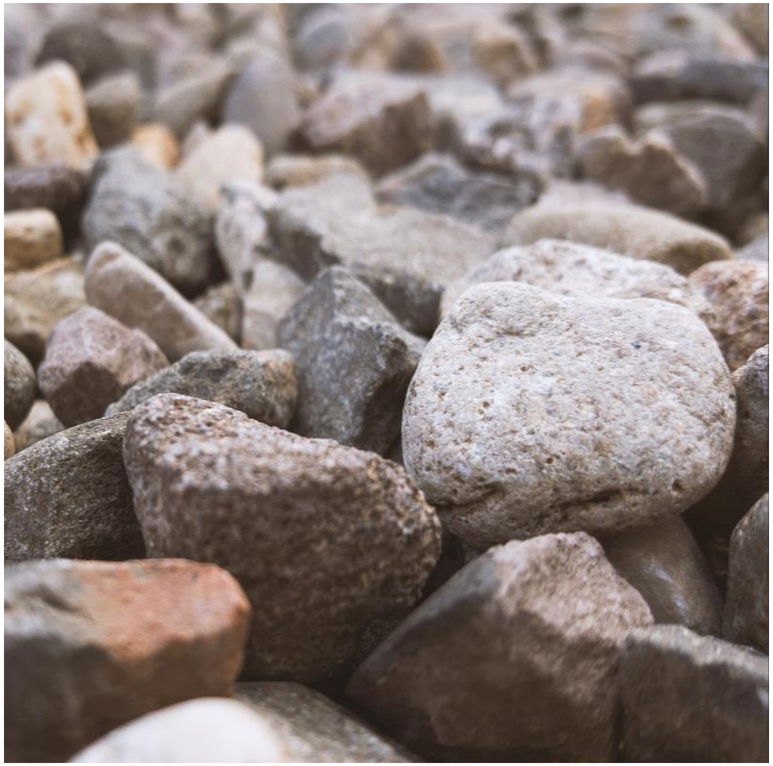
Rigid Body Dynamics

Libin Liu

CFCS, Peking University

Rigid Bodies

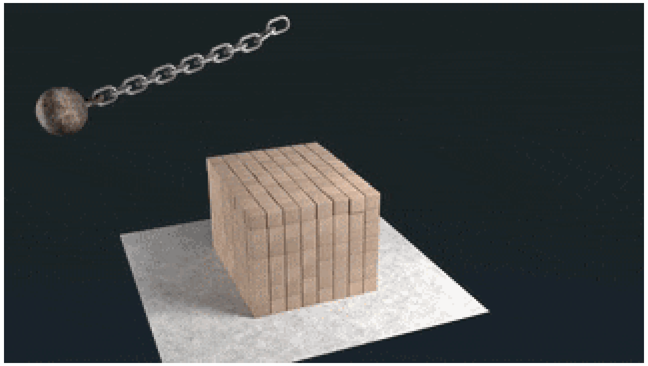
- They are rigid....

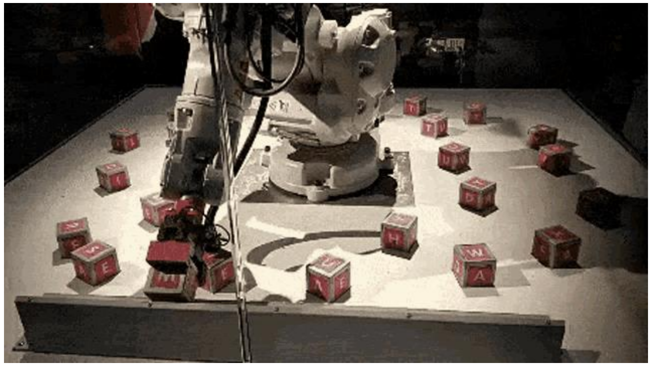


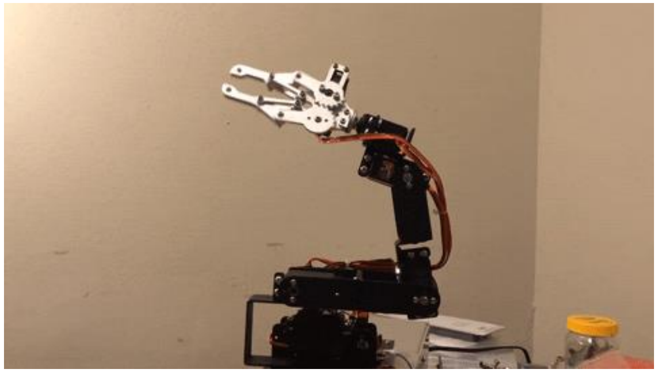
Outline

- Equations of Rigid Bodies
  - Rigid Body Kinematics
  - Newton-Euler equations
- Articulated Rigid Bodies
  - Joints and constraints
- Contact Models
  - Penalty-based contact
  - Constraint-based contact
- Control of rigid-body characters?

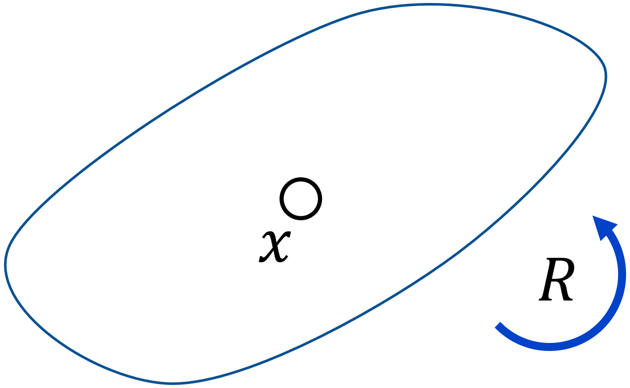
https://www.cs.cmu.edu/~baraff/sigcourse/



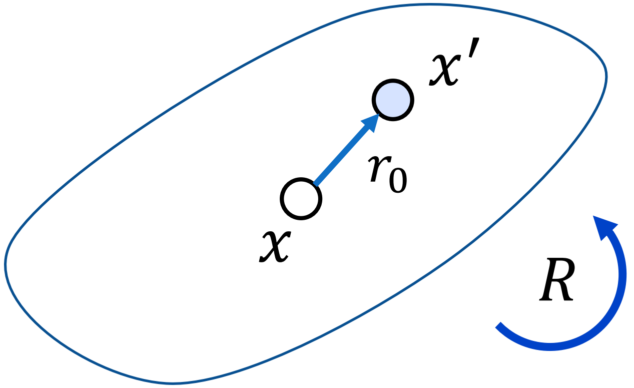




Position and Orientation

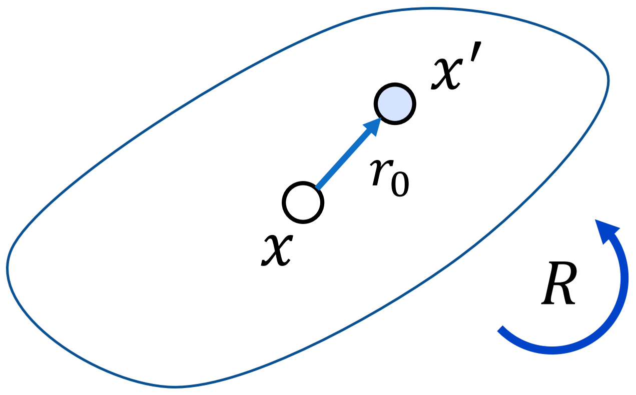


Position and Orientation



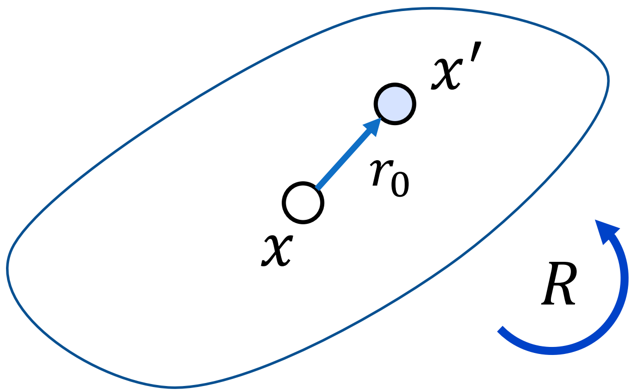
$$x' = x + Rr_0$$

Position and Orientation



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

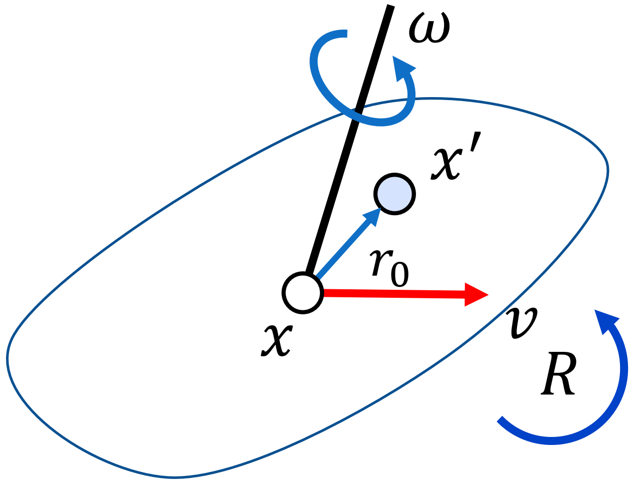
Linear and Angular Velocity



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

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

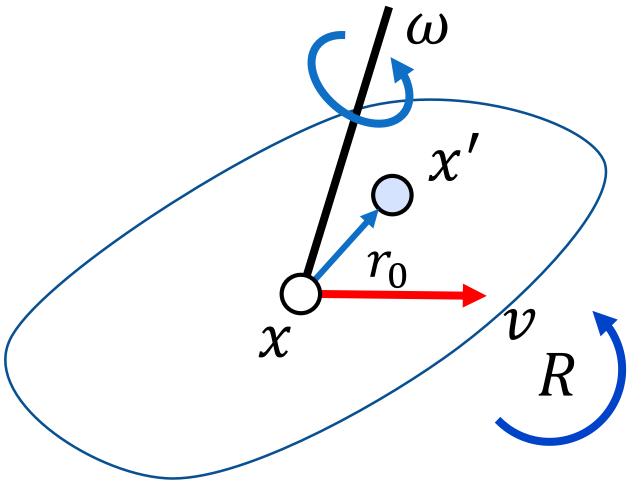
Linear and Angular Velocity



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

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

Linear and Angular Velocity

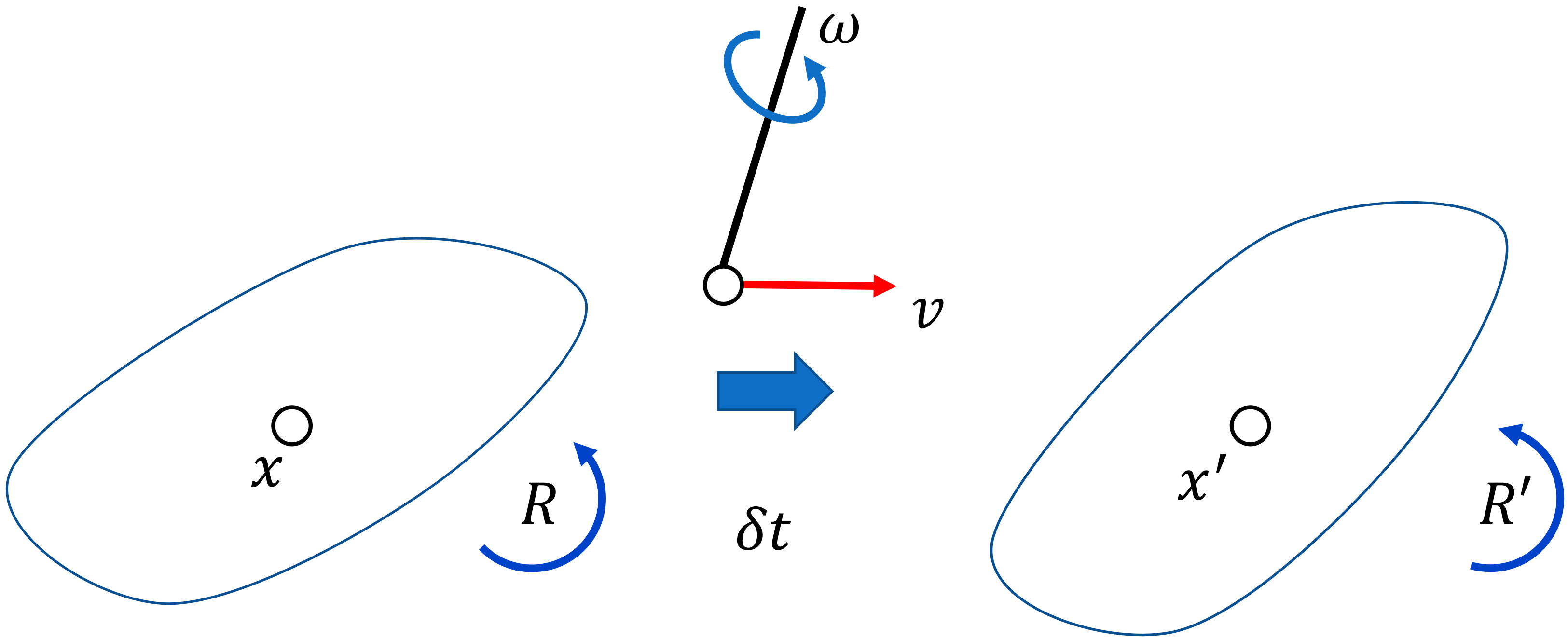


$$\omega = \dot{R}?$$

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

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

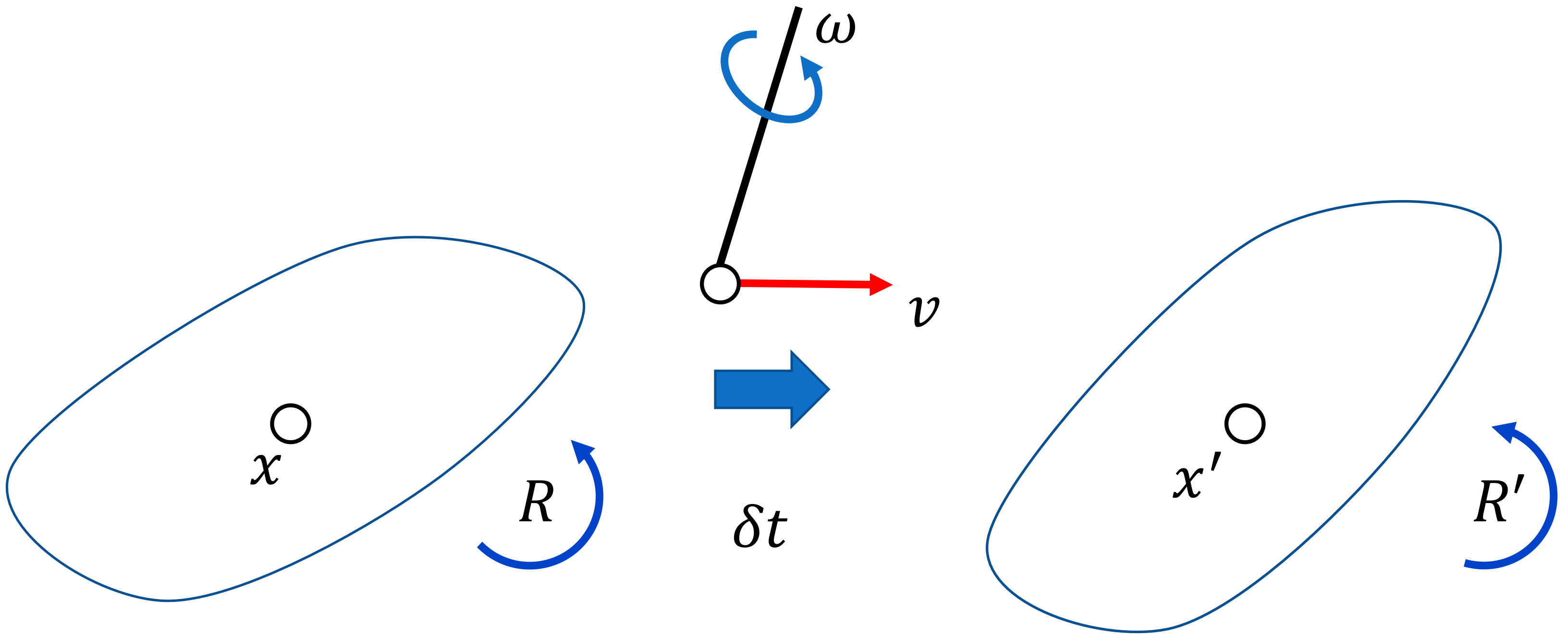
Numerical Integration



$$x' = ?$$

$$R' = ?$$

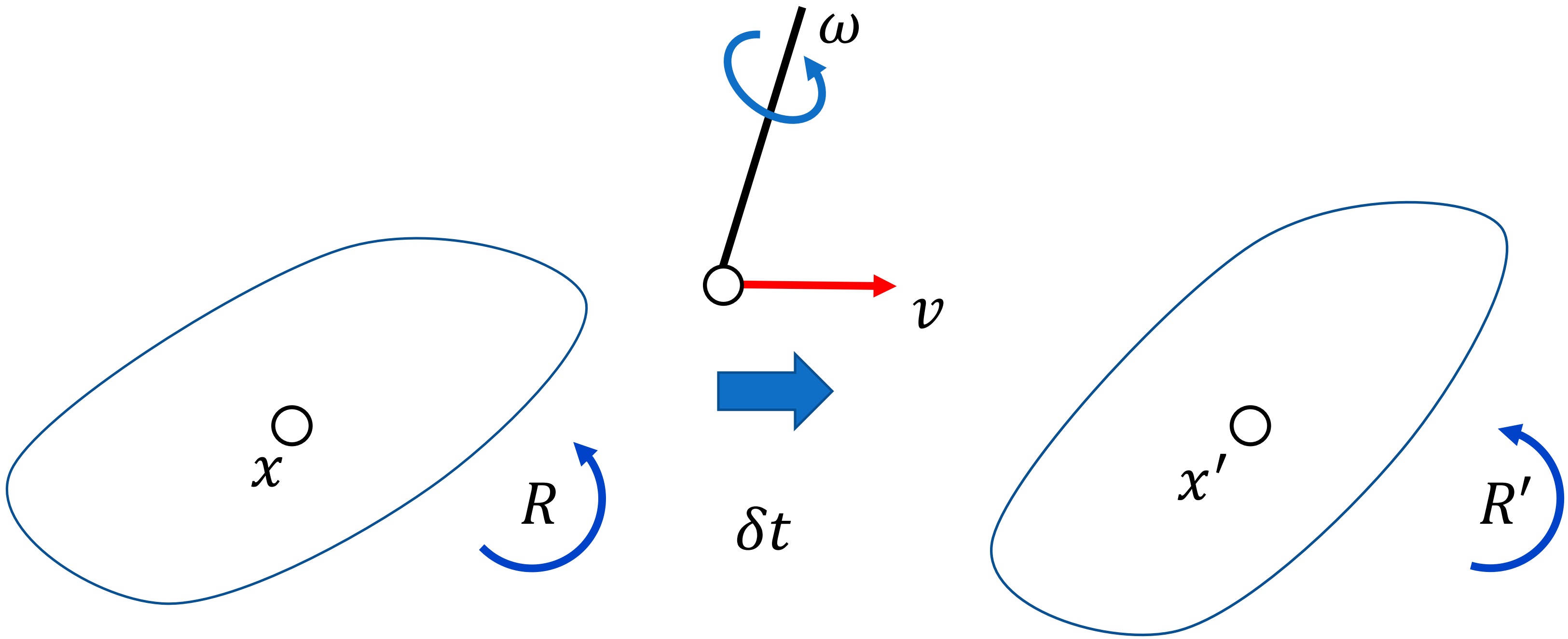
Numerical Integration



$$\dot{x} = v$$

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

Numerical Integration

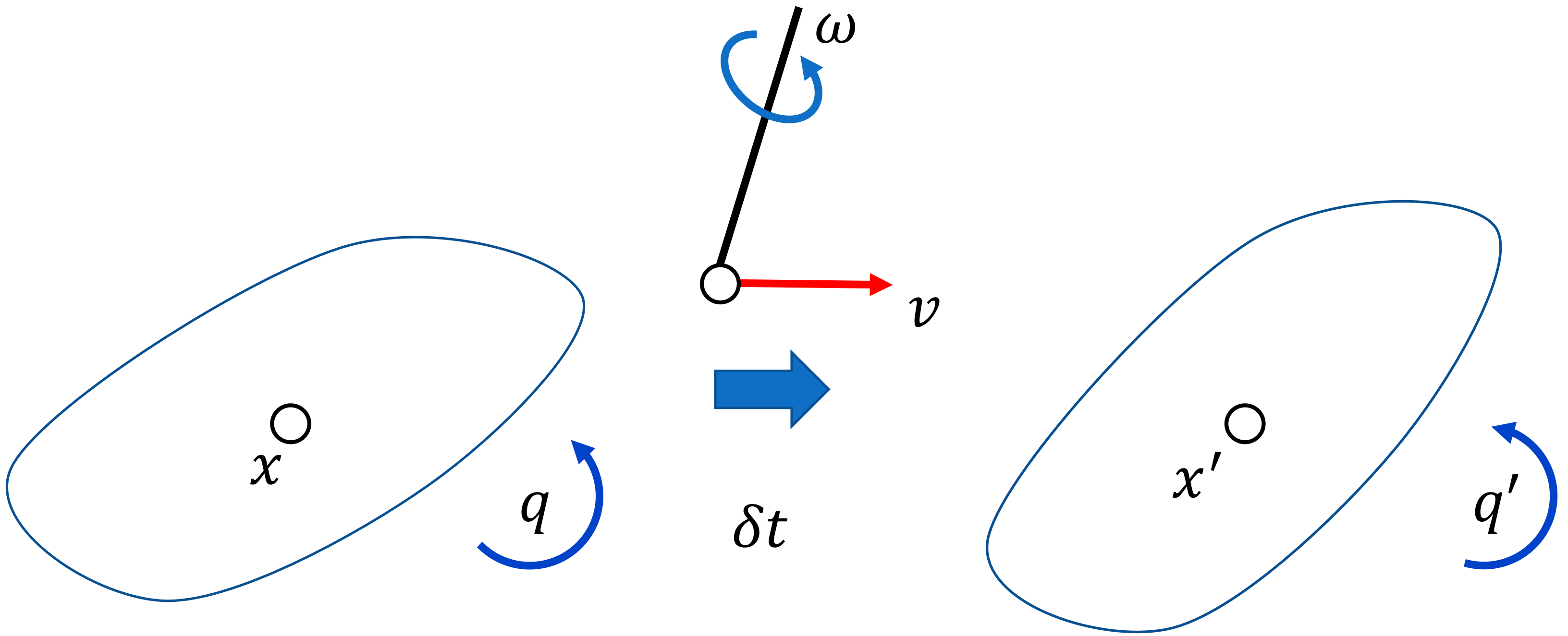


$$\dot{x} = v \qquad \Rightarrow \qquad x' = x + \delta t \cdot v$$

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

Need orthogonalization!

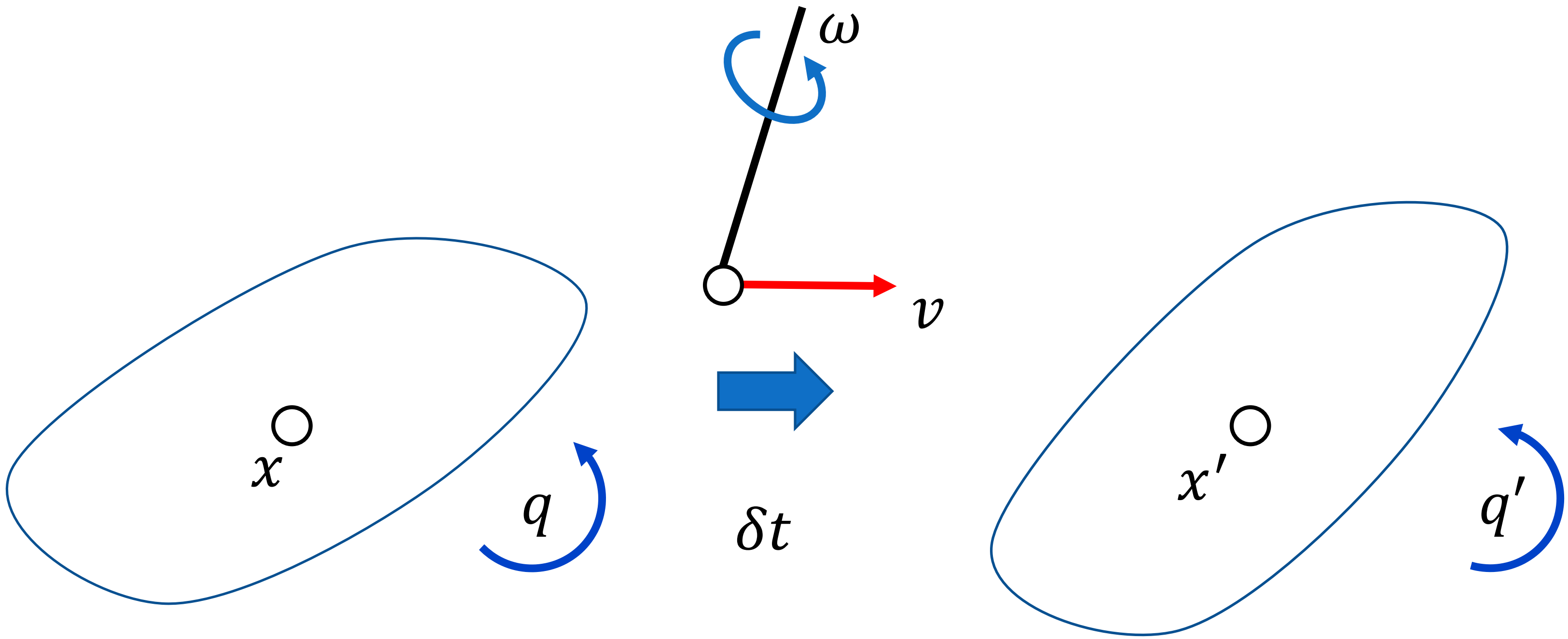
Numerical Integration: Quaternion



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

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

Numerical Integration: Quaternion



$$\dot{x} = v$$

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

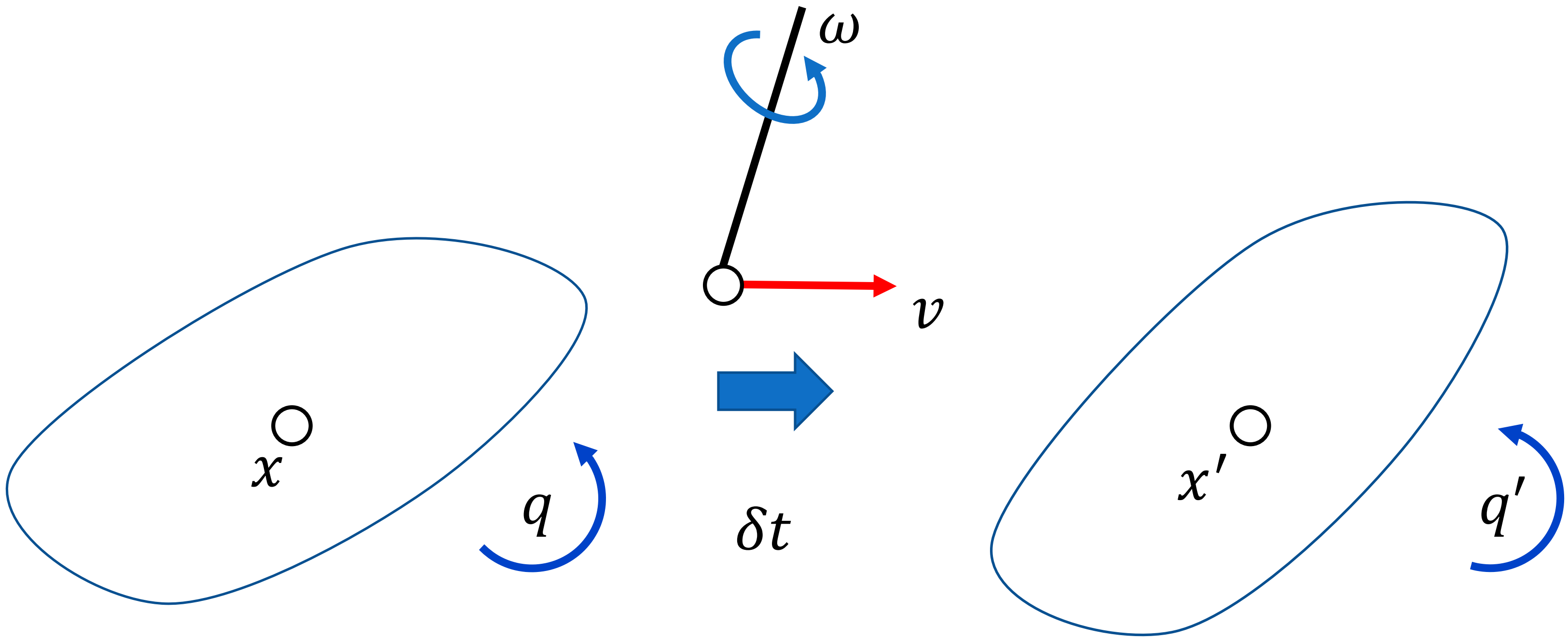
$\overline{\omega} = (0, \omega)$

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

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

Need Normalization!

Numerical Integration: Quaternion



$$\dot{x} = v$$

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

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

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

https://arxiv.org/abs/0811.2889

Need Normalization!

Kinematics vs. Dynamics

Kinematics

x, R

v, ω

a, α

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

...

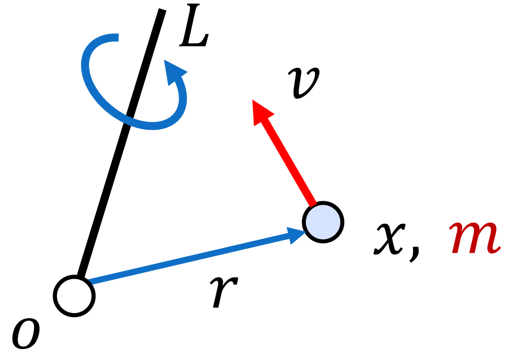
m, I

Dynamics

p, L

F, τ

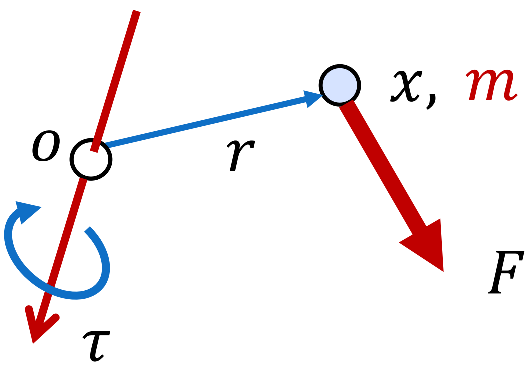
Linear and Angular Momentum of a Particle



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

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

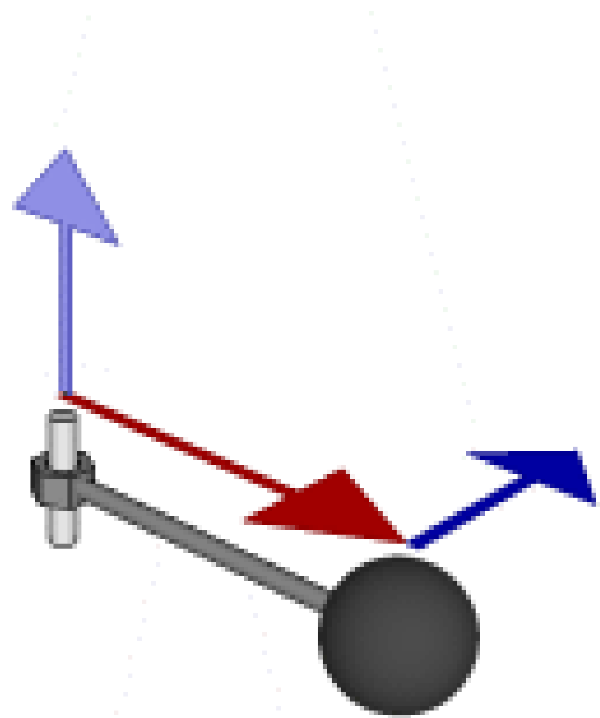
Force and Torque



$$\tau = r \times F$$

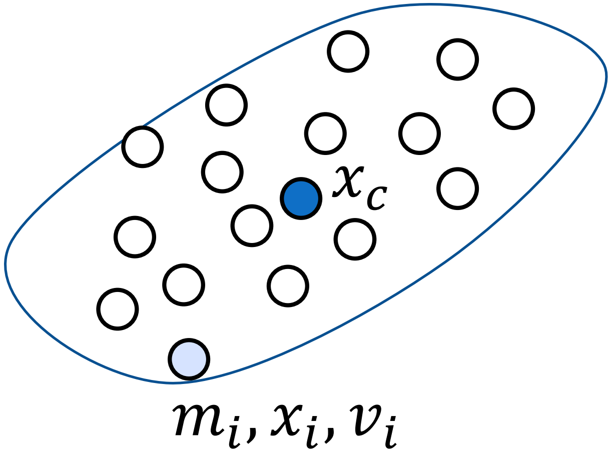
Torque and Angular Momentum

$$\boldsymbol{\tau} = \mathbf{r} \times \mathbf{F}$$
$$\mathbf{L} = \mathbf{r} \times \mathbf{p}$$



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

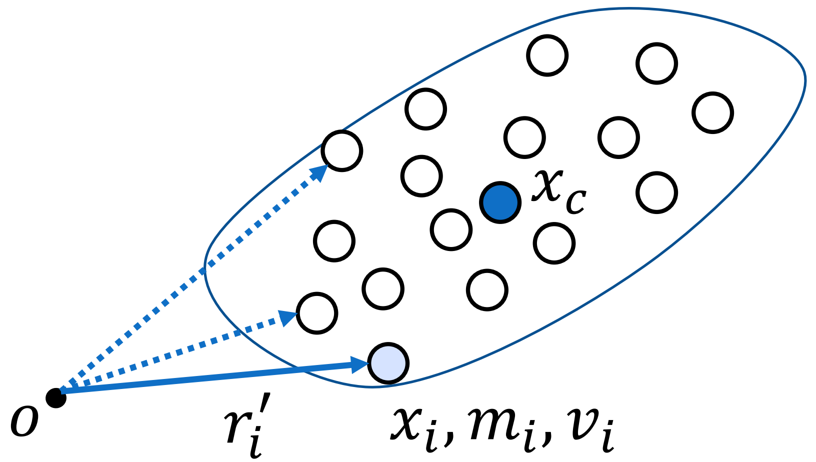
Rigid Body as a Collection of Particles



$$M = \sum_i m_i$$

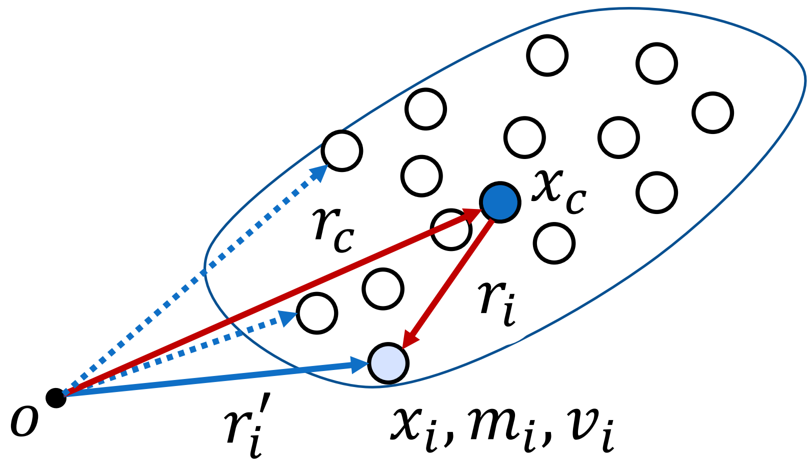
$$x_c = \frac{\sum_i m_i x_i}{\sum_i m_i} \qquad v_c = \frac{\sum_i m_i v_i}{\sum_i m_i}$$

Moments of a Rigid Body



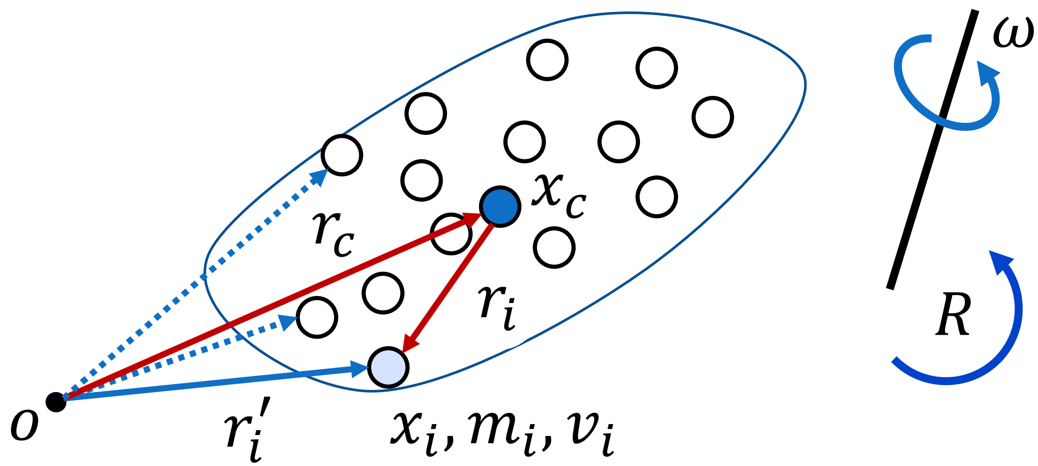
$$p = \sum_i m_i v_i \qquad L = \sum_i m_i r_i' \times v_i$$

Angular Moment of a Rigid Body



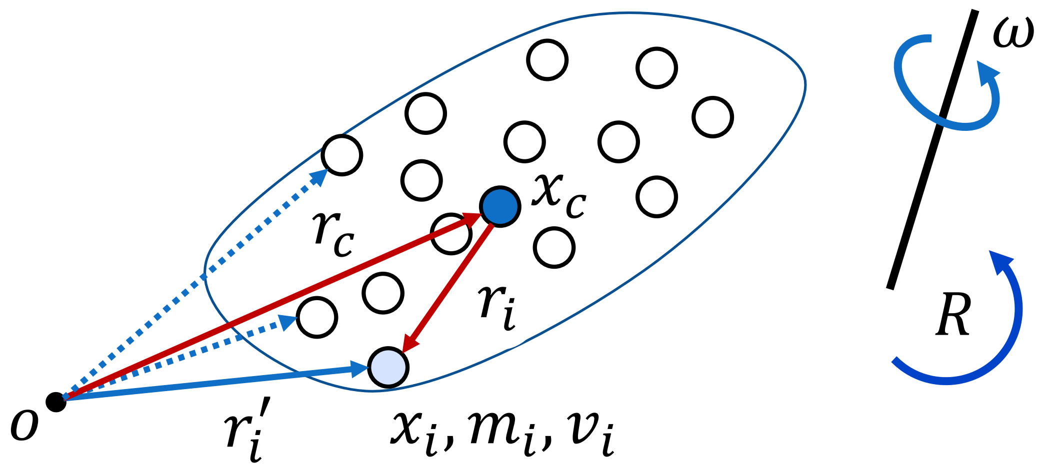
$$L = \sum_i m_i r_i' \times v_i = M r_c \times v_c + \sum_i m_i r_i \times v_i$$

Angular Moment of a Rigid Body



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

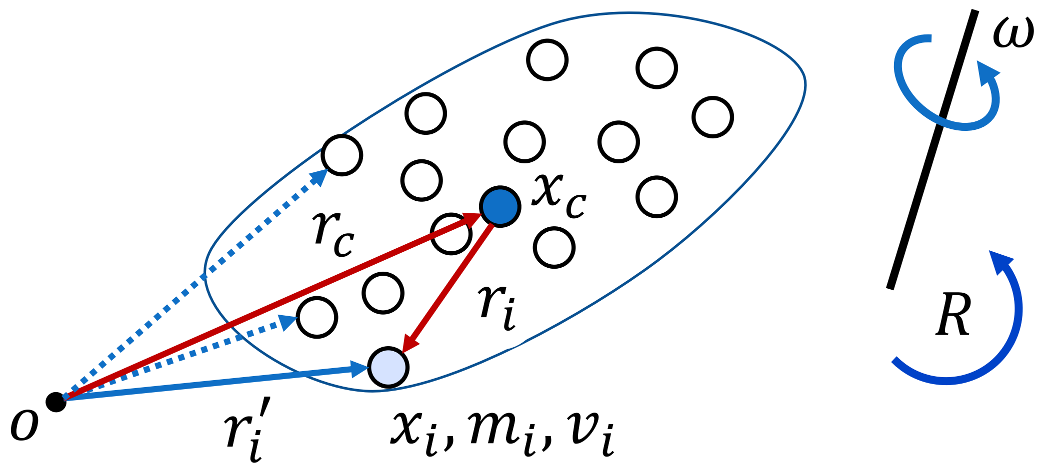
Angular Moment of a Rigid Body



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

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

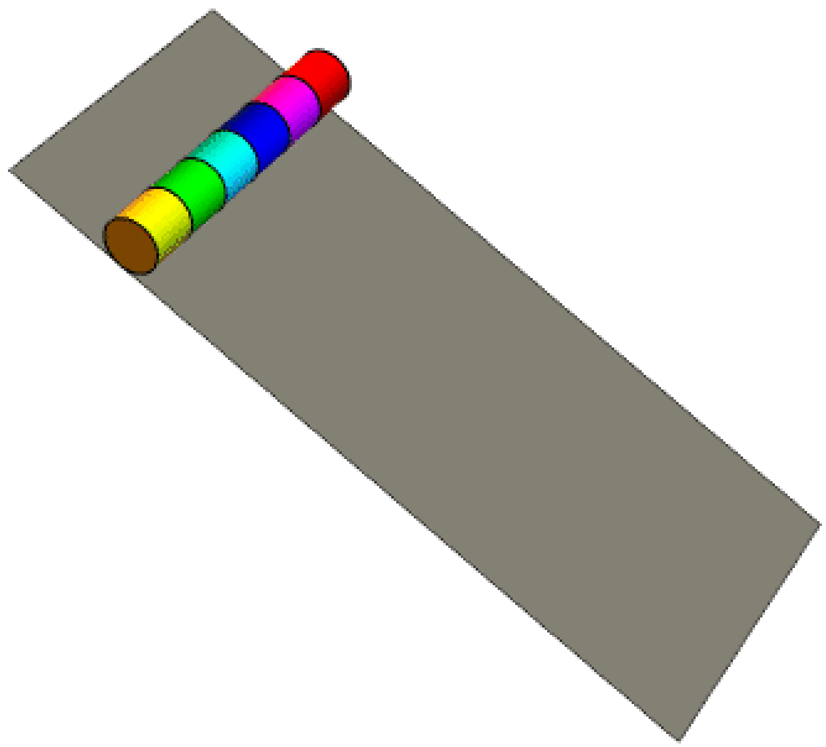
Moment of Inertia



$$L = Mr_c \times v_c + I\omega$$

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

Moment of Inertia



$m = m_0 \quad I = 1\, I_0$

$m = m_0 \quad I = 2\, I_0$

$m = m_0 \quad I = 3\, I_0$

$m = m_0 \quad I = 4\, I_0$

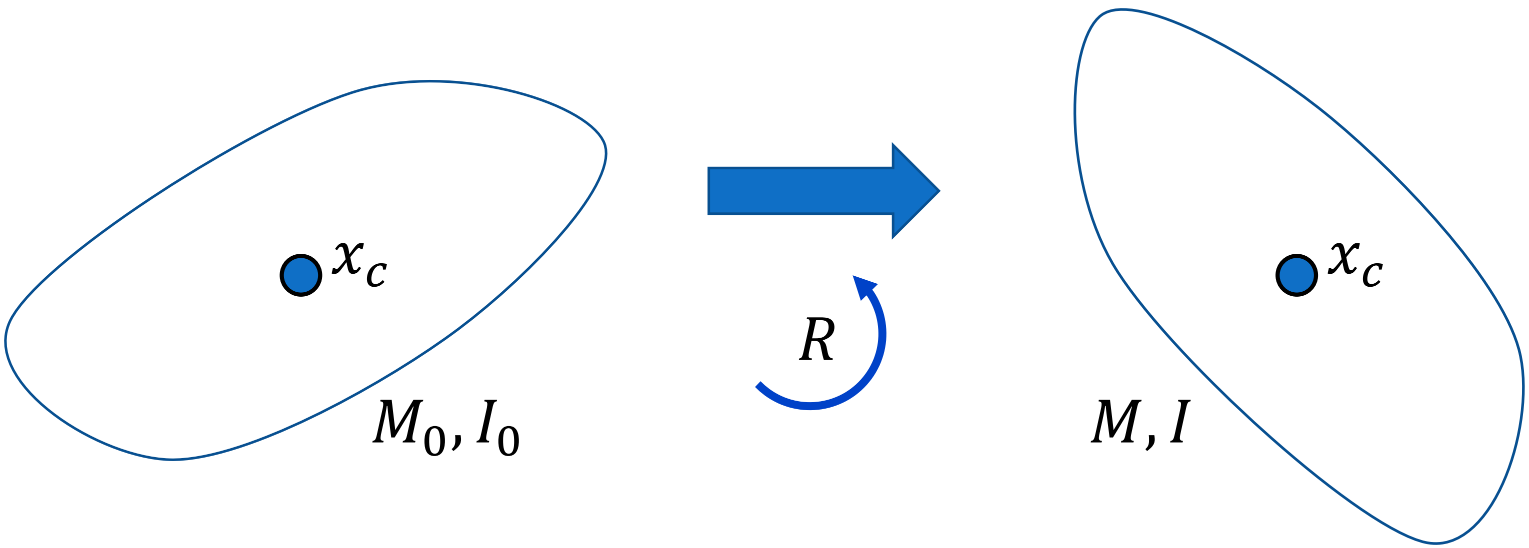
$m = m_0 \quad I = 5\, I_0$

$m = m_0 \quad I = 6\, I_0$

Moment of Inertia

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

Rotation of Moment of Inertia



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

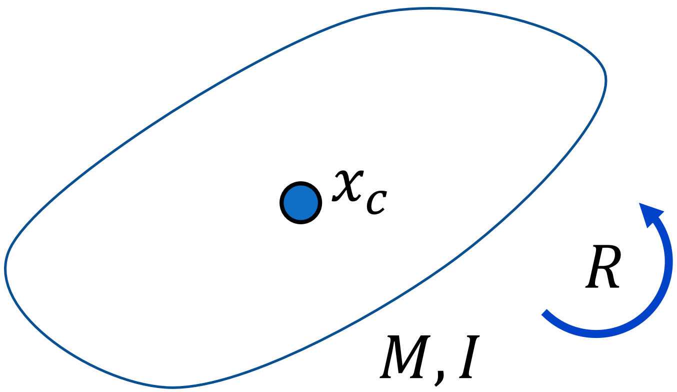
$$[Rr]_\times = R[r]_\times R^T$$

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

$$M = M_0$$

$$I = R I_0 R^T$$

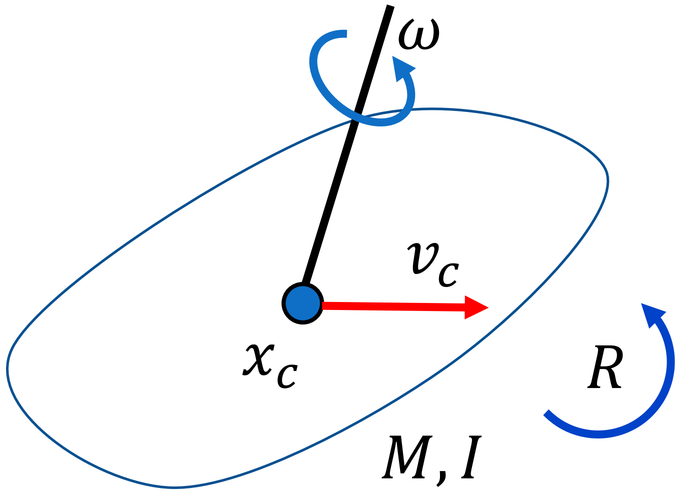
Principal Axes of Moment of Inertia



Eigendecomposition $\Rightarrow$ $I = R I_0 R^T$

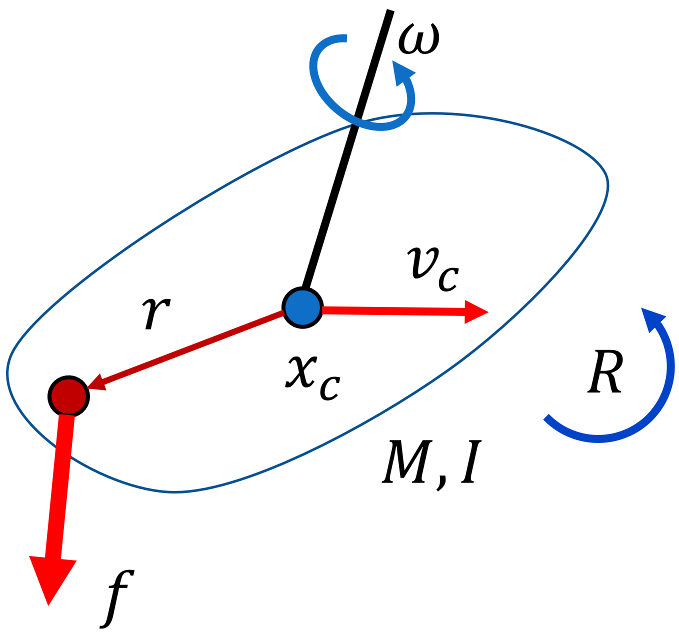
$$I_0 = \begin{bmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{bmatrix} = \mathrm{diag}(I_1, I_2, I_3)$$

Center of Momentum (CoM) Frame

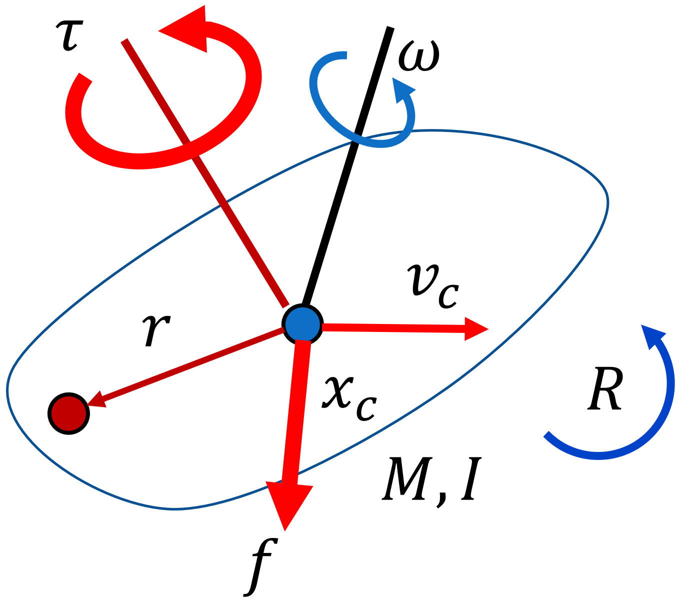


$p = Mv_c$ $\quad\quad\quad L = I\omega$

Force on a Rigid Body

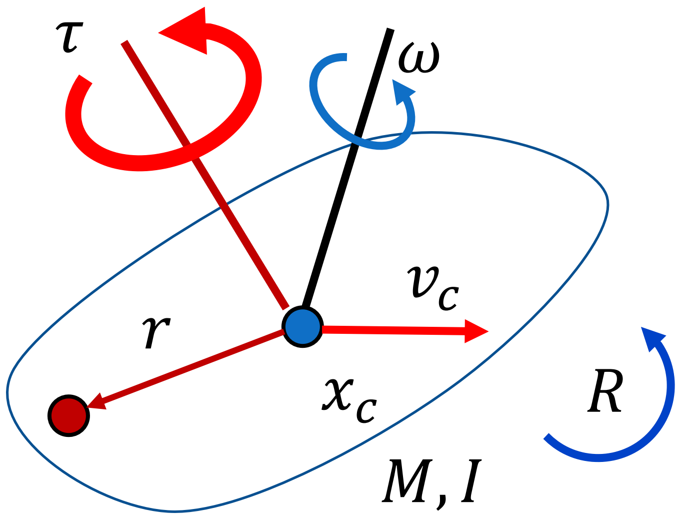


Force on a Rigid Body



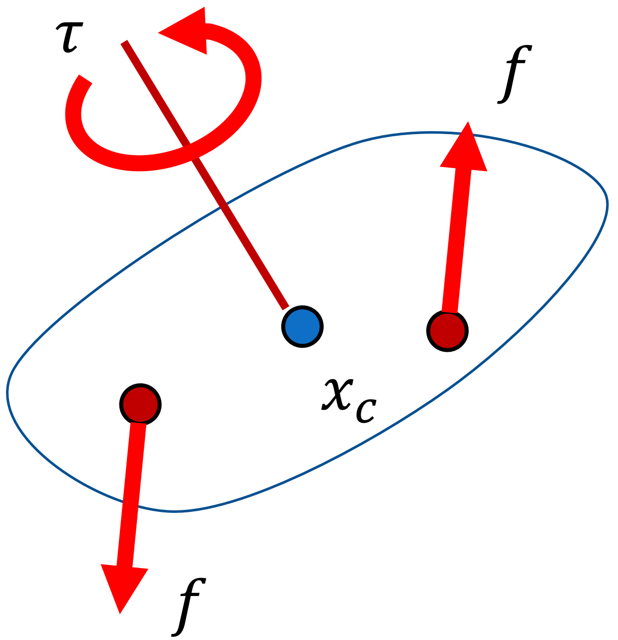
$$\tau = r \times f$$

Torque on a Rigid Body



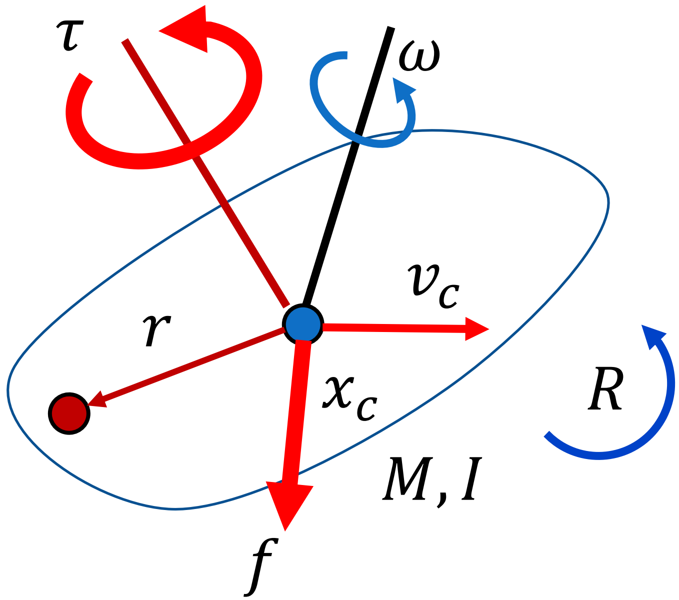
$$\tau = ???$$

Torque on a Rigid Body



$$\tau = ???$$

Equation of Motion of Rigid Body



Kinematics

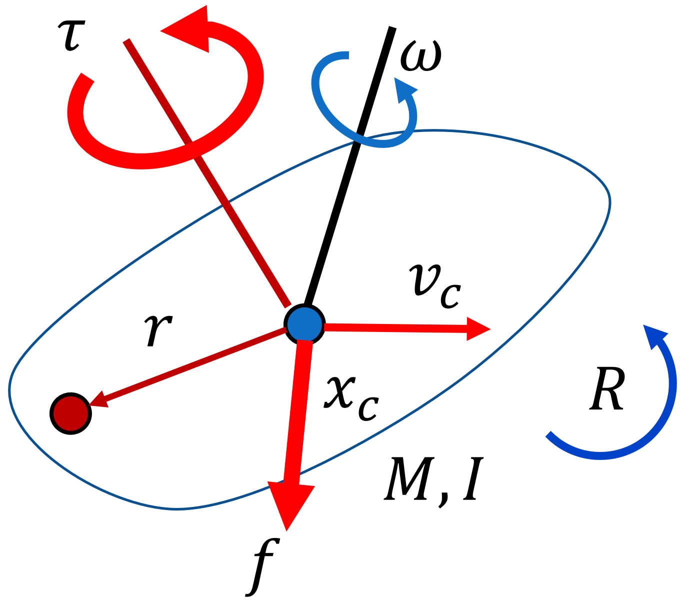
m, I

Dynamics

x, R
v, ω

p, L
f, τ

Equation of Motion of Rigid Body



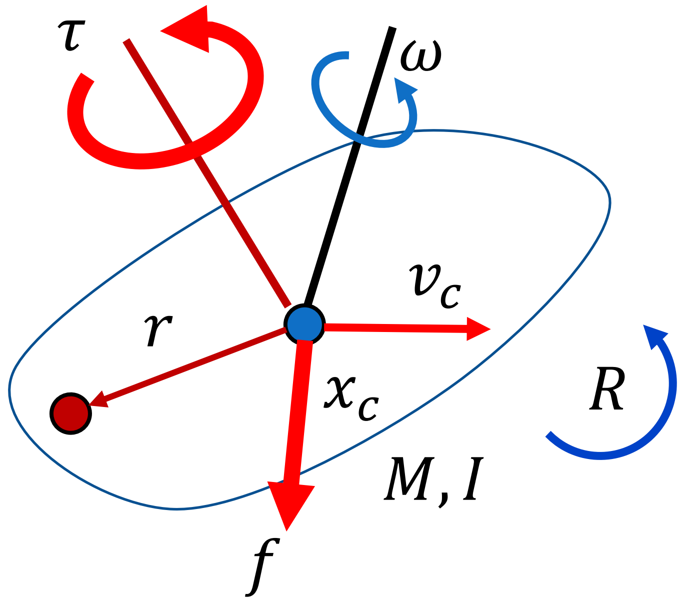
x_c, R, v_c, ω

$p = M v_c$

$L = I\omega$

Newton's Second Law: $f = Ma$

Equation of Motion of Rigid Body



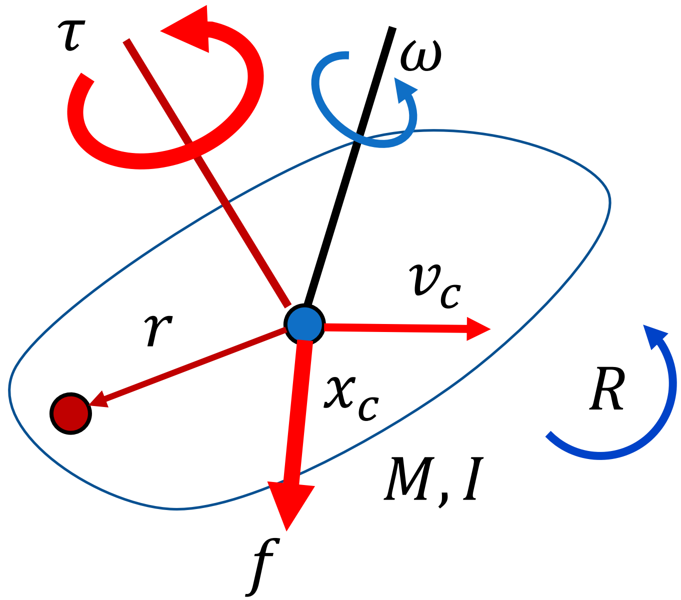
$$x_c, R, v_c, \omega$$

$$p = M v_c$$

$$L = I\omega$$

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

Equation of Motion of Rigid Body



x_c, R, v_c, ω

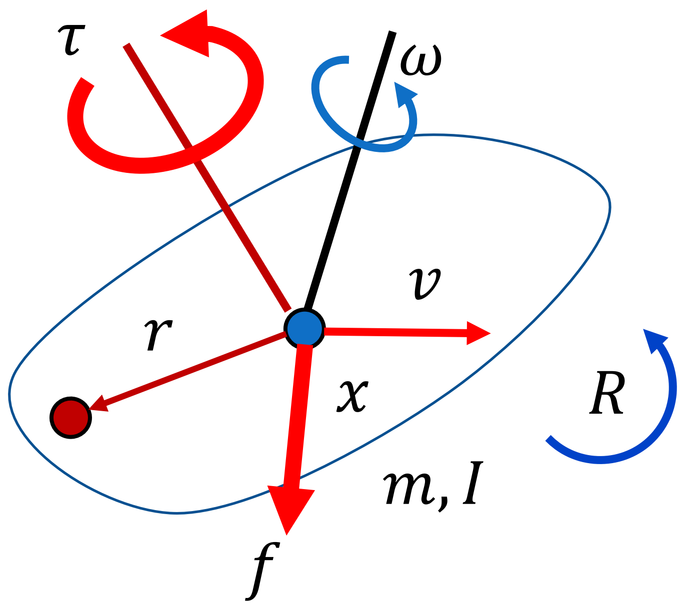
$p = Mv_c$

$L = I\omega$

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

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

Newton–Euler Equations



$$x, R, v, \omega$$

$$p = mv_c$$

$$L = I\omega$$

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

Numerical Integration

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

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

Rigid Body Simulation

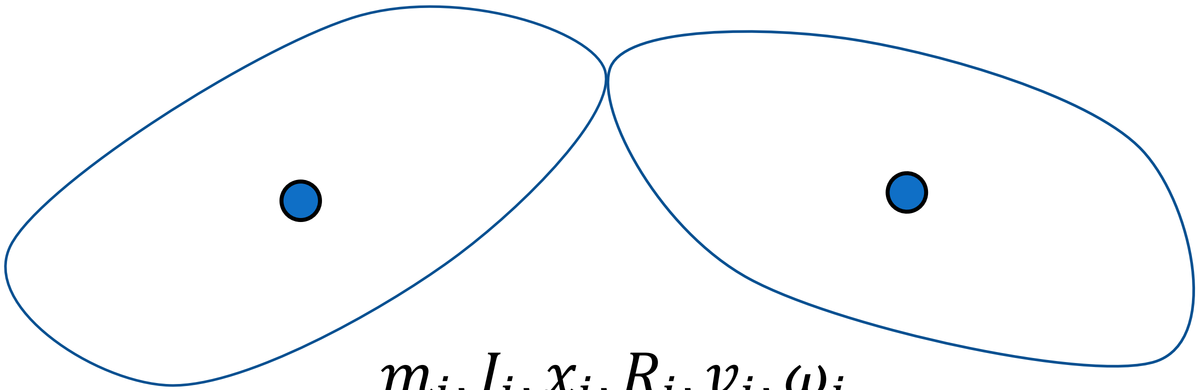
$$I_n = R_n I_0 R_n^T$$

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

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

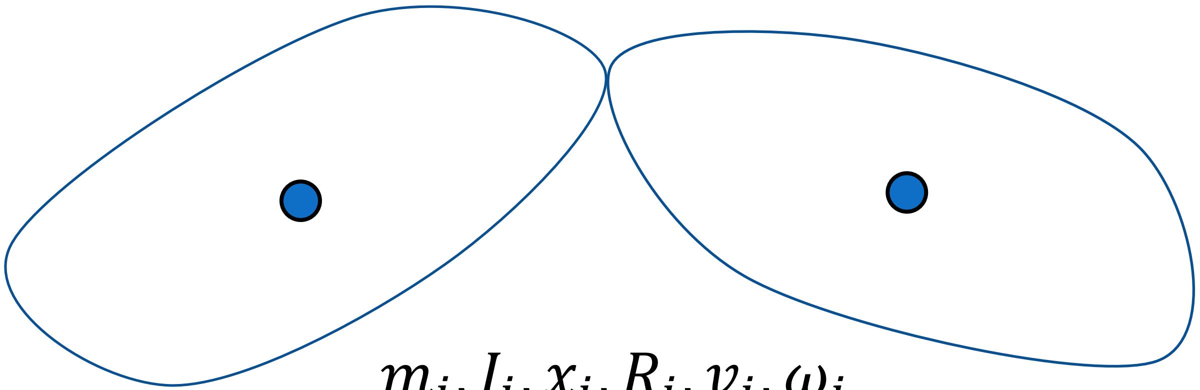
$$q_{n+1} = q_n + \frac{1}{2} h \overline{\omega}_{n+1} q$$

A System with Two Links



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

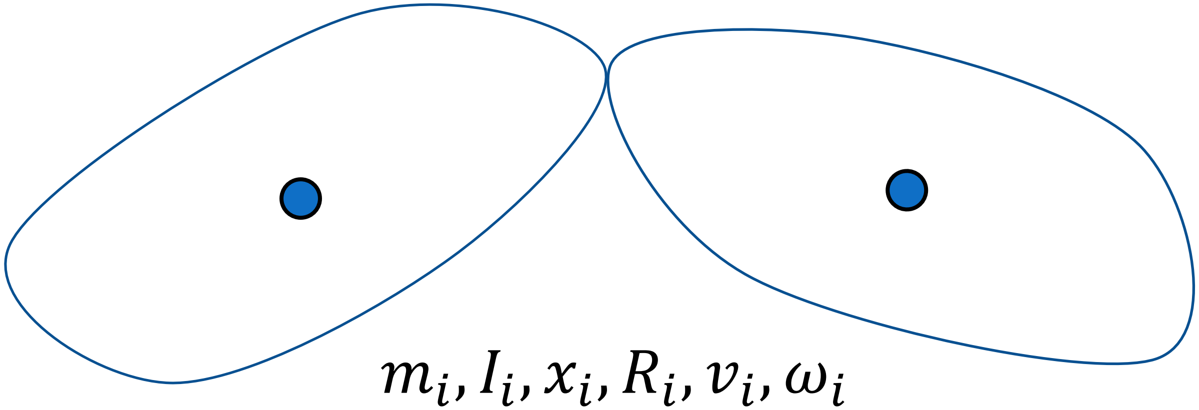
A System with Two Links



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

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

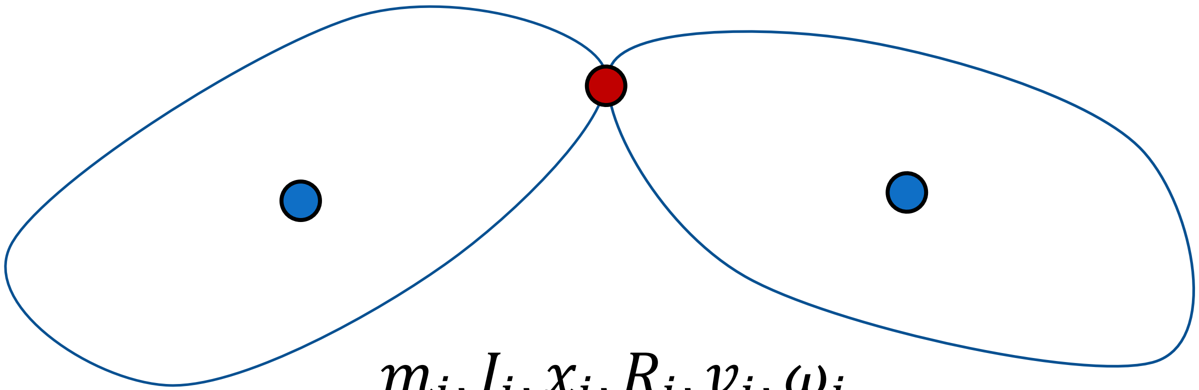
A System with Two Links



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

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

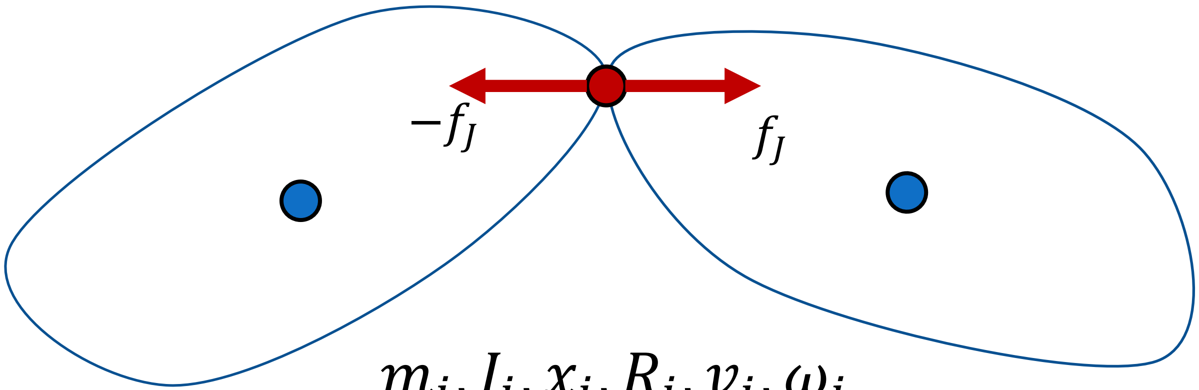
A System with Two Links and a Joint



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

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

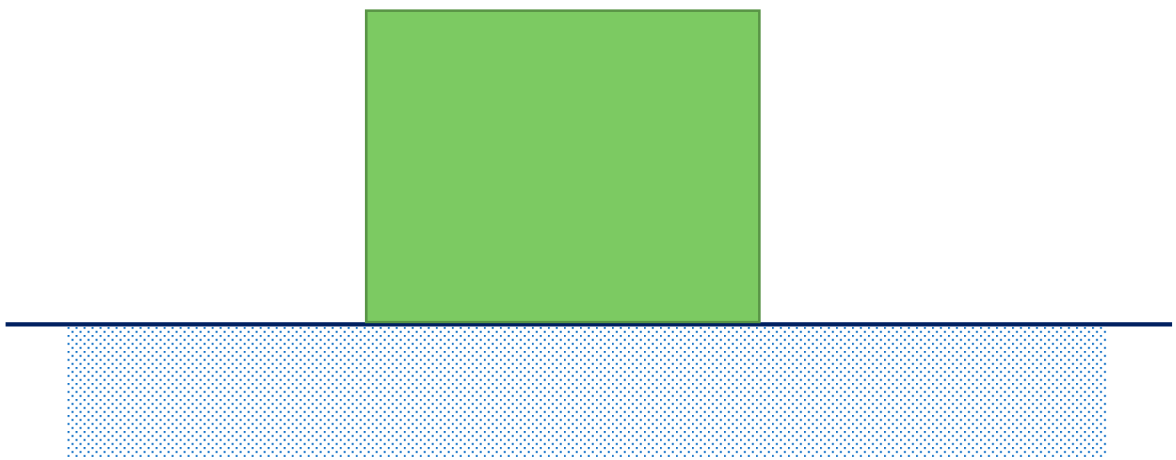
A System with Two Links and a Joint



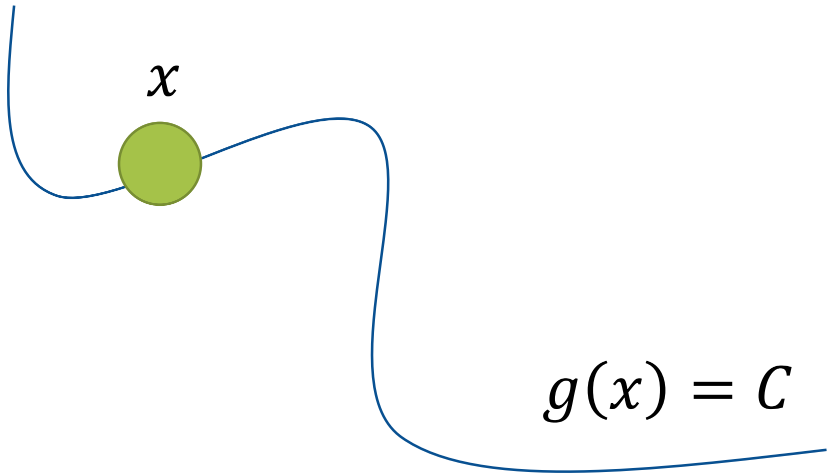
$m_i, I_i, x_i, R_i, v_i, \omega_i$

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

Constraints



Constraints



$$g(x) = C$$

$$\Downarrow$$

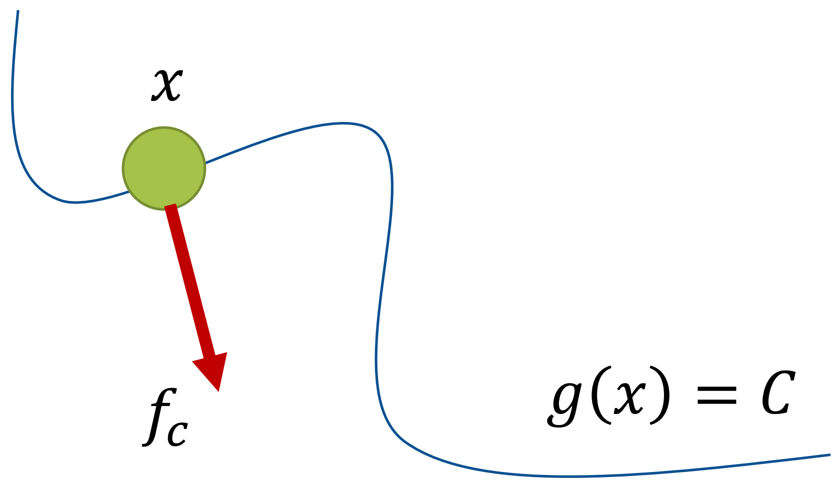
$$\frac{d}{dt} g(x) = 0$$

$$\Downarrow$$

$$Jv = 0$$

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

Constraint Force

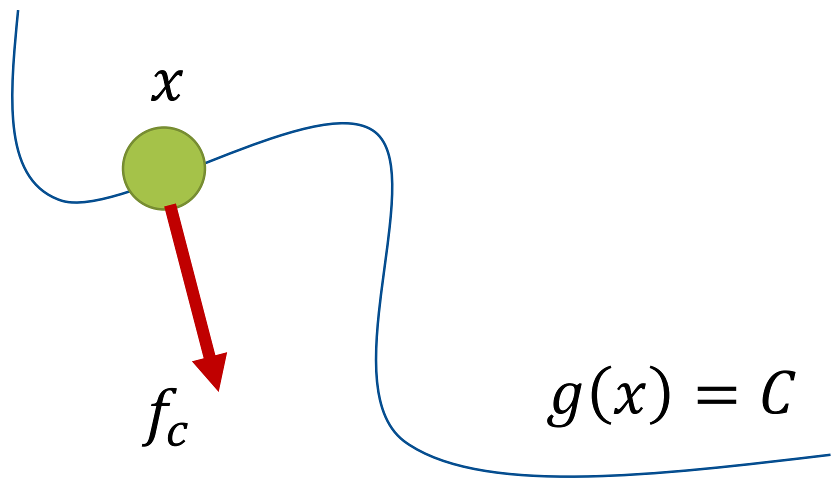


$$Jv = 0$$

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

$$f_c = J^T \lambda$$

Constraint Force



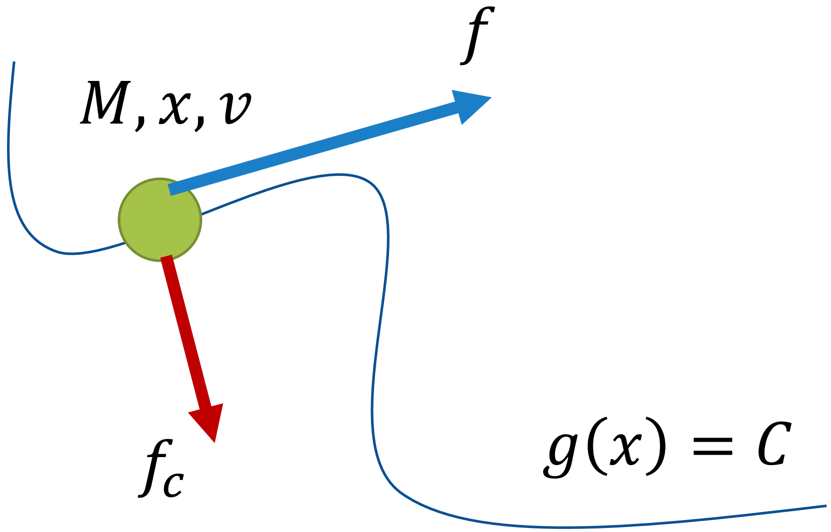
$$Jv = 0$$

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

$$f_c = J^T \lambda$$

unknown

Equation of Motion with Constraints



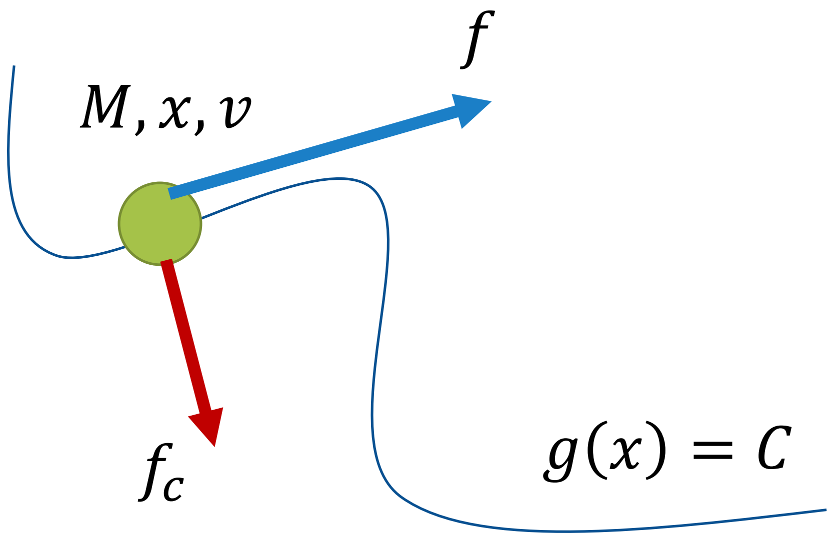
$$M\dot{v} = f + J^T\lambda$$

$$Jv = 0$$

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

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

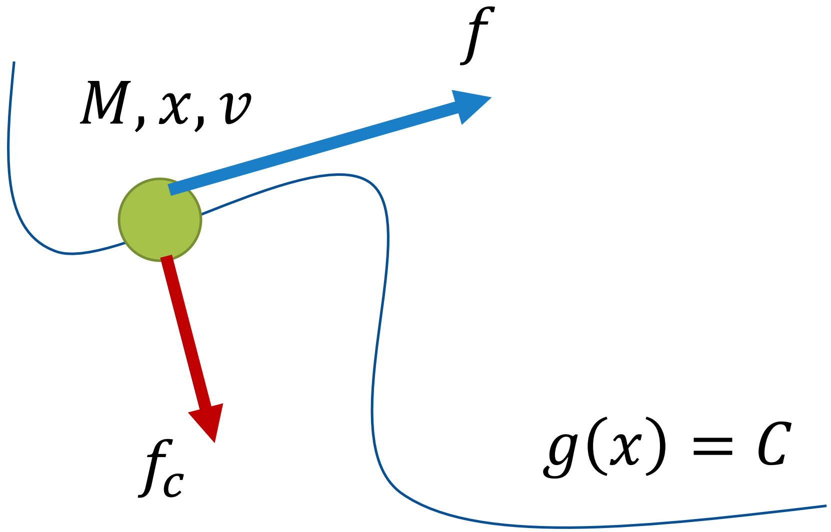
Numerical Solution



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

$$J v_{n+1} = 0$$

Numerical Solution



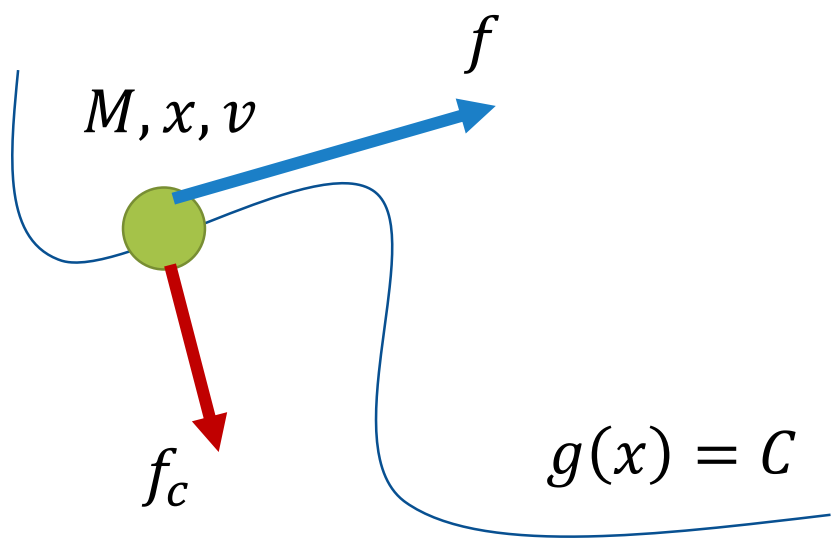
$$M \frac{v_{n+1} - v_n}{h} = f + J^T \lambda$$

$$J v_{n+1} = \mathbf{0}$$

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

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

Numerical Solution



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

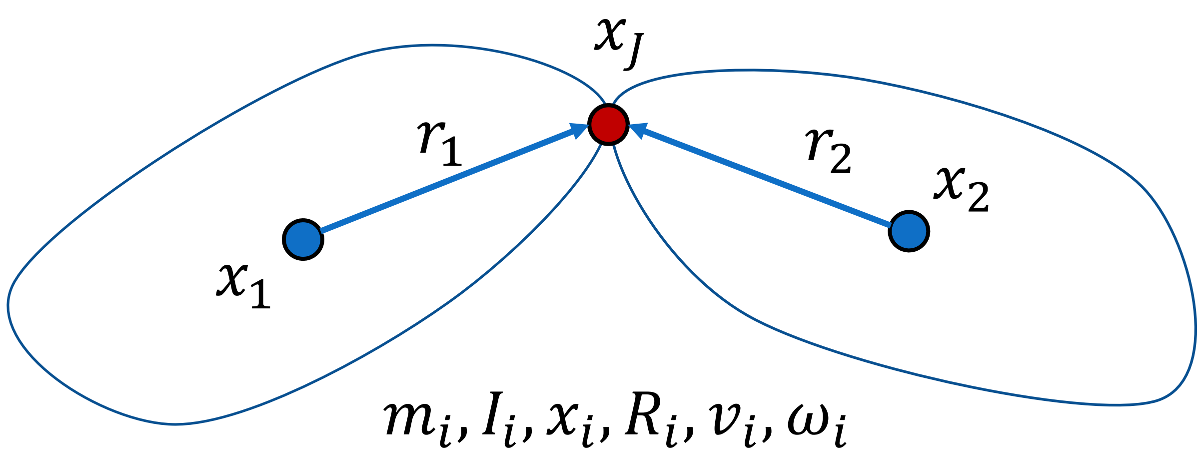
$$Jv_{n+1} = b_n$$

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

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

β: constraint force mixing (CFM)

Joint Constraint

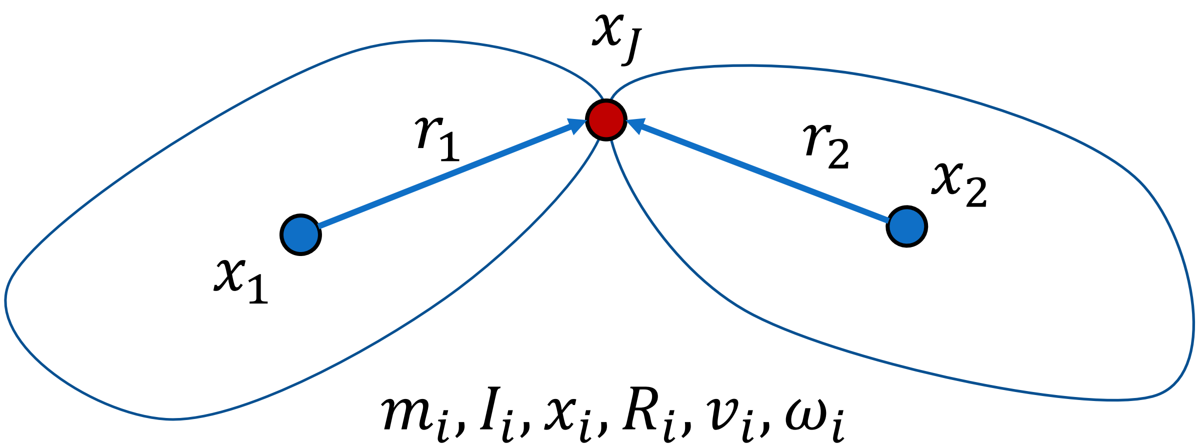


$$x_1 + R_1 r_1 = x_J = x_2 + R_2 r_2$$

$^{d}/_{dt}$

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

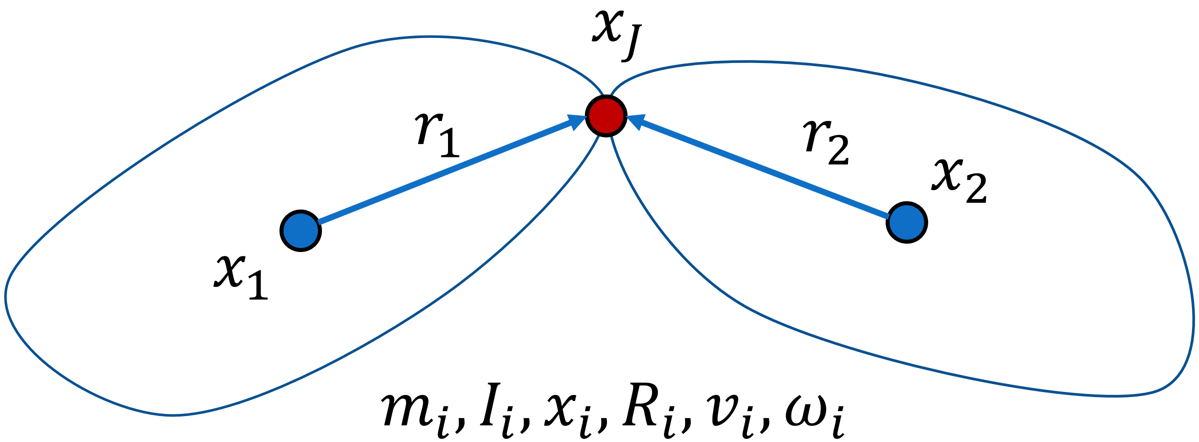
Joint Constraint



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

$$Jv = 0$$

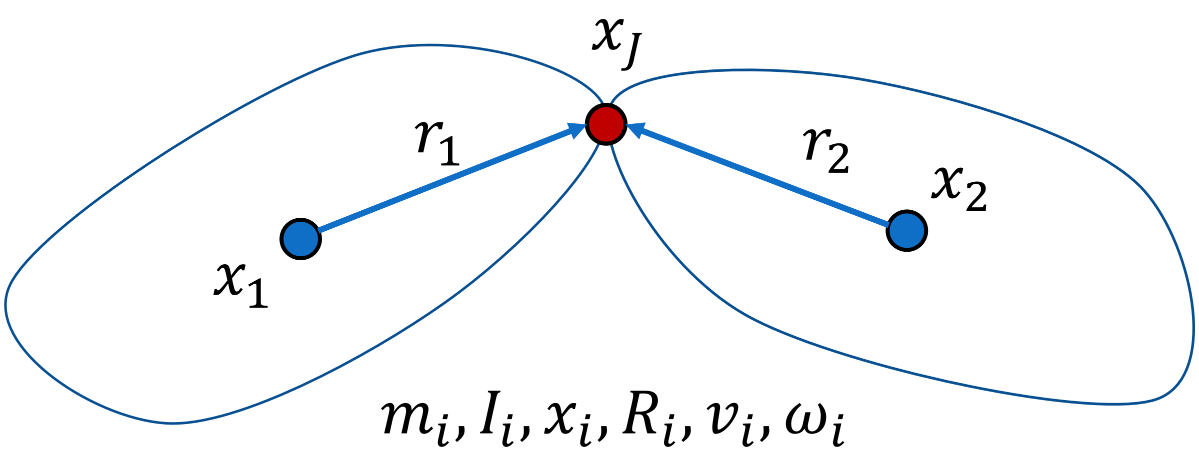
A System with Two Links and a Joint



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

$$Jv = 0$$

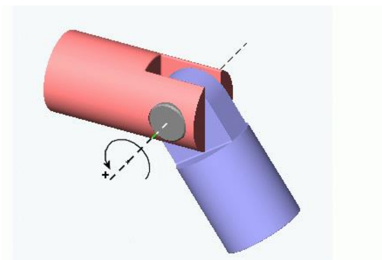
A System with Two Links and a Joint



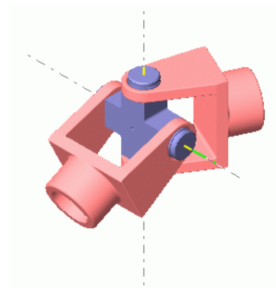
$$\begin{bmatrix} m_1\mathrm{I}_3 & & & \\ & I_1 & & \\ & & m_2\mathrm{I}_3 & \\ & & & I_2 \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{\omega}_1 \\ \dot{v}_2 \\ \dot{\omega}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_1 \times I_1\omega_1 \\ 0 \\ \omega_2 \times I_2\omega_2 \end{bmatrix} = \begin{bmatrix}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

$$Jv = 0$$

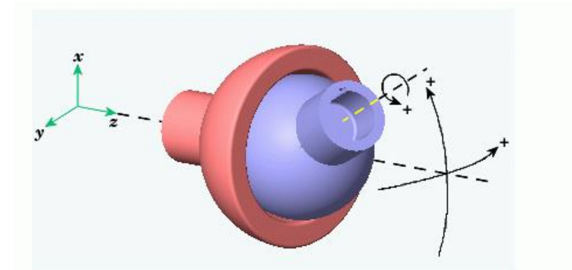
Different Types of Joints



Hinge joint
Revolute joint



Universal joint

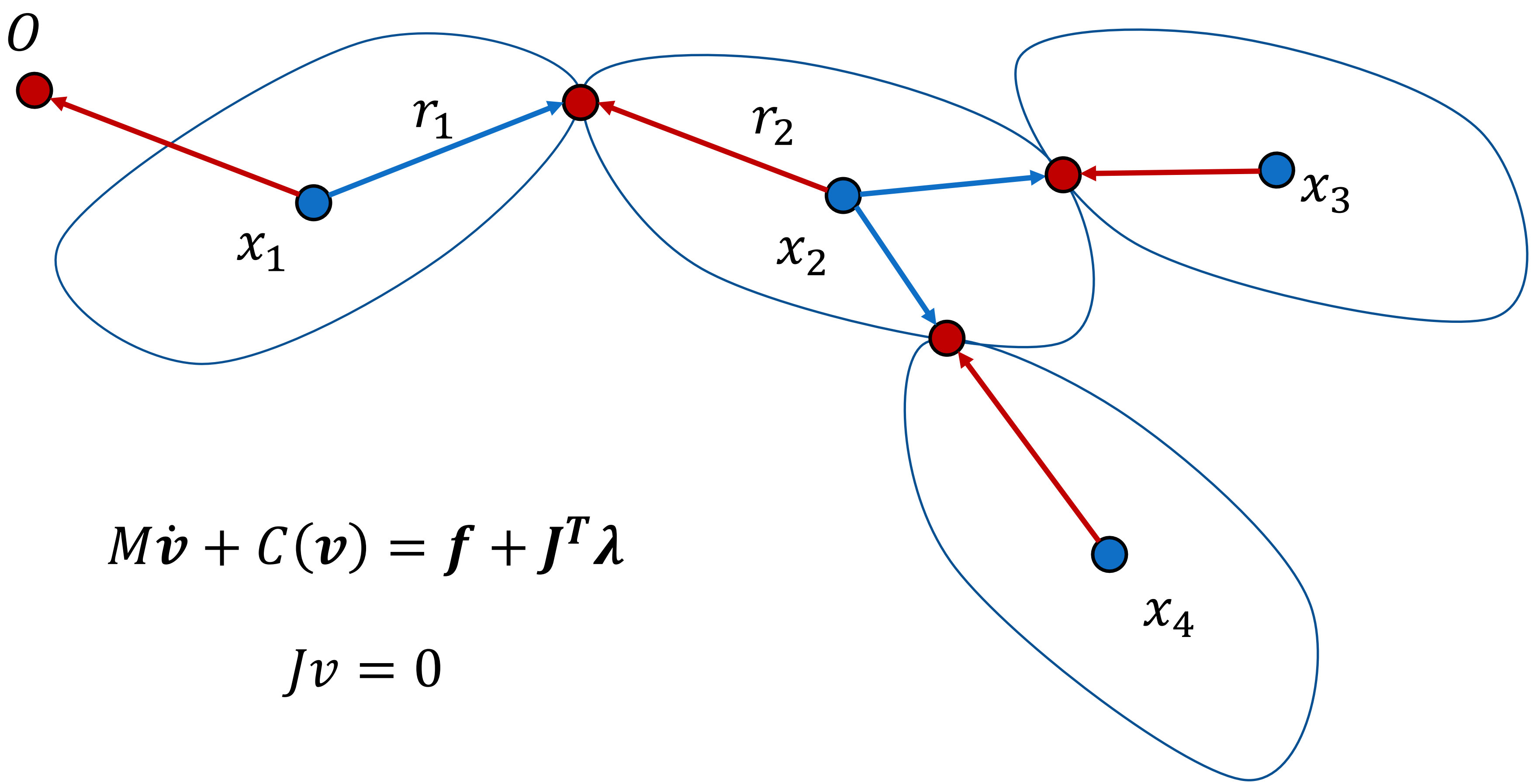


Ball-and-socket

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

A System with Many Links Joints

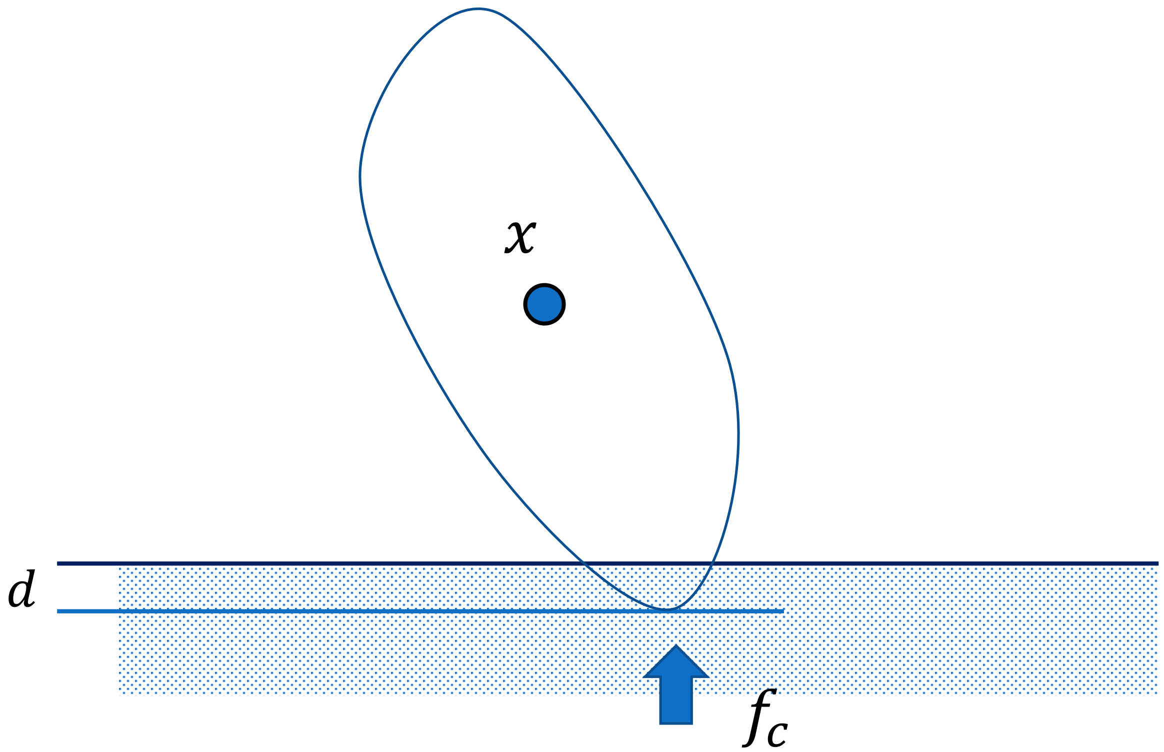
$m_i, I_i, x_i, R_i, v_i, \omega_i$



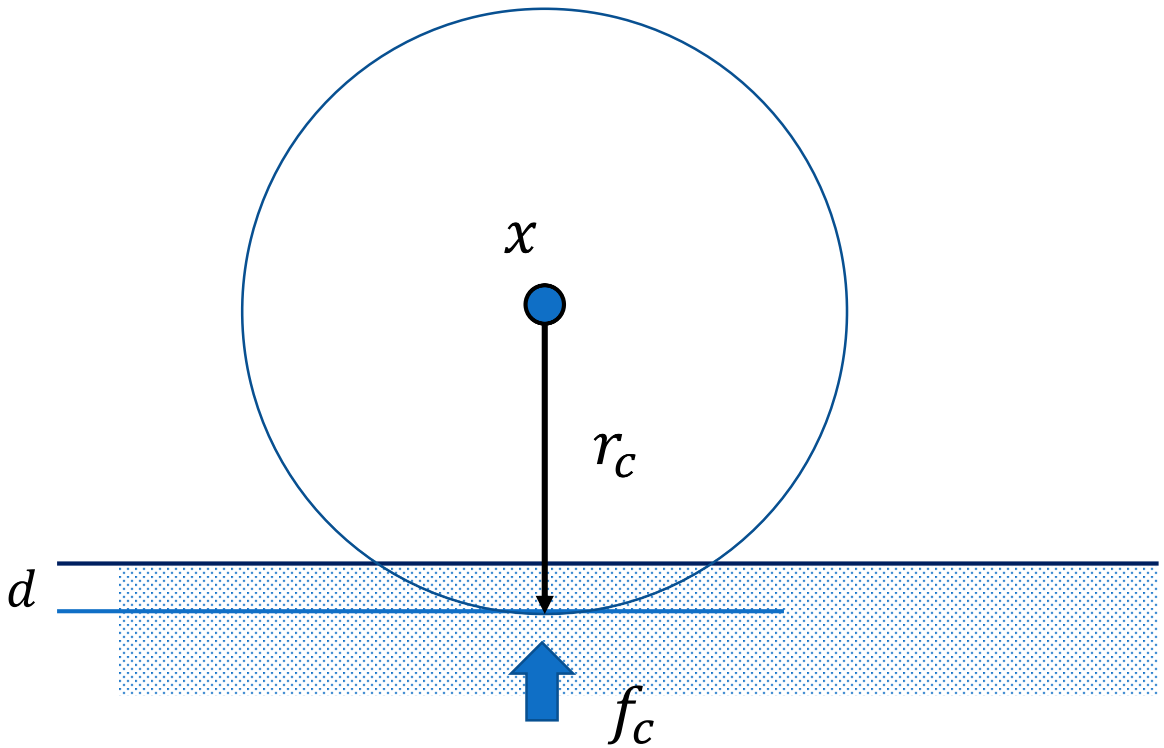
$$M\dot{\boldsymbol{v}} + C(\boldsymbol{v}) = \boldsymbol{f} + \boldsymbol{J}^T\boldsymbol{\lambda}$$

$$Jv = 0$$

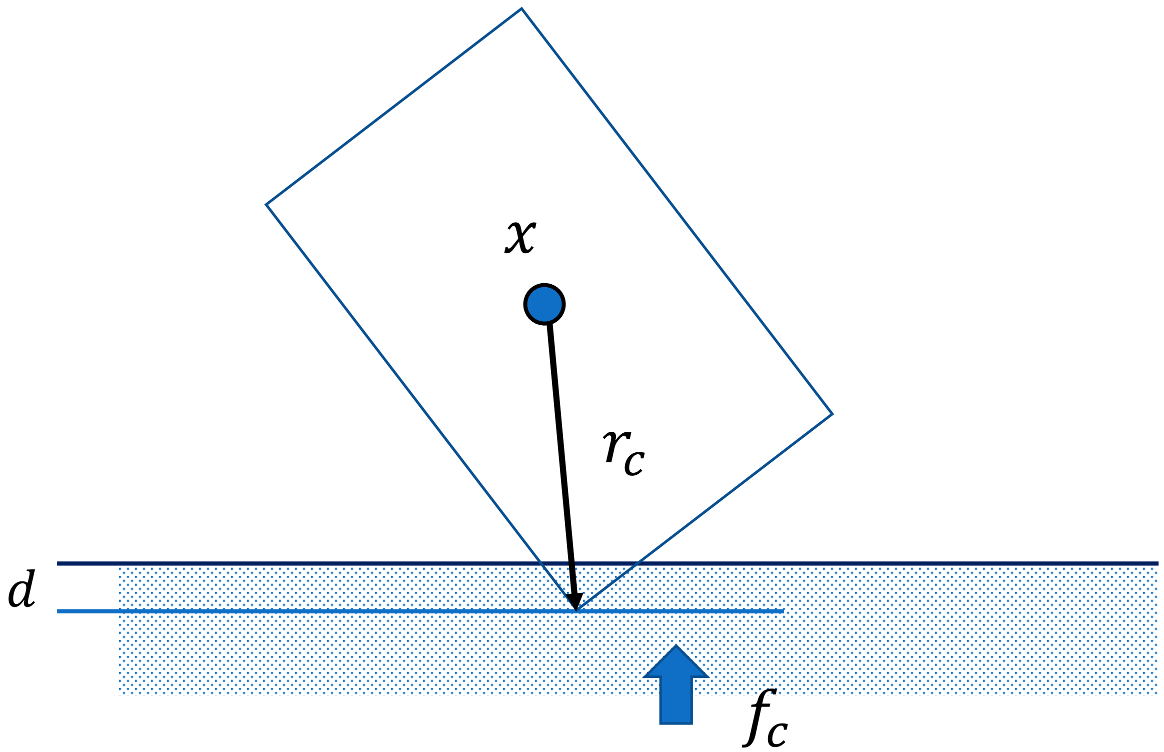
Contacts



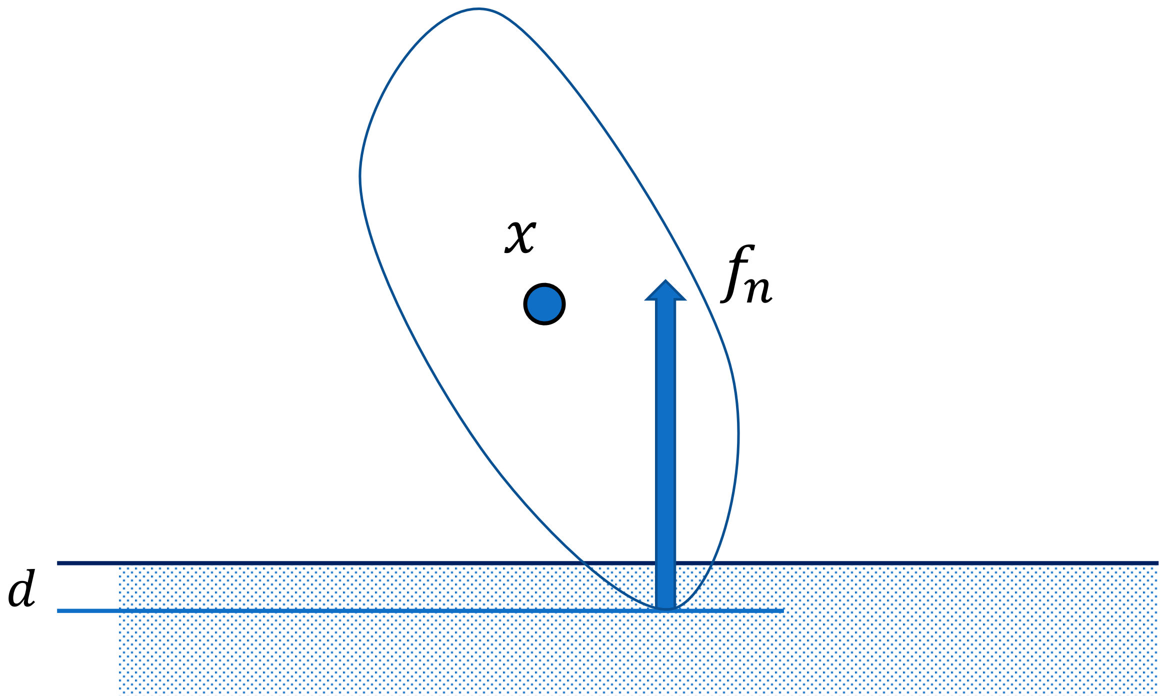
Contact Detection



Contact Detection

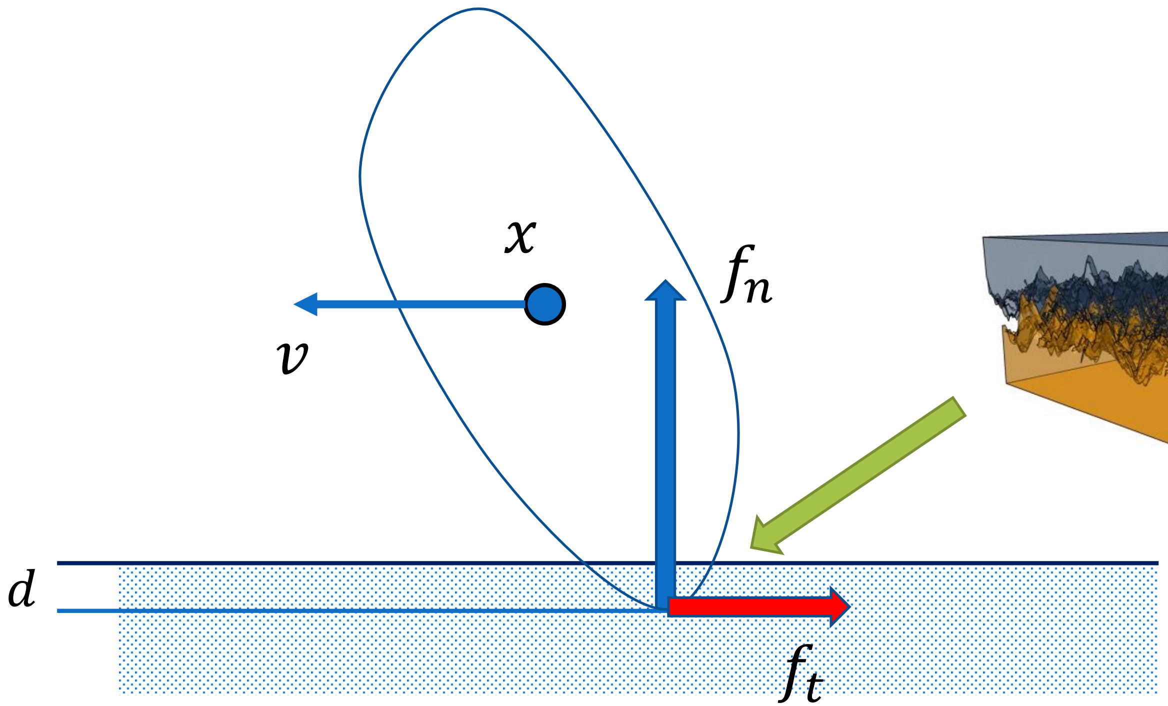


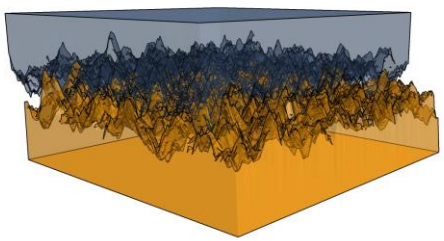
Penalty-based Contact Model



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

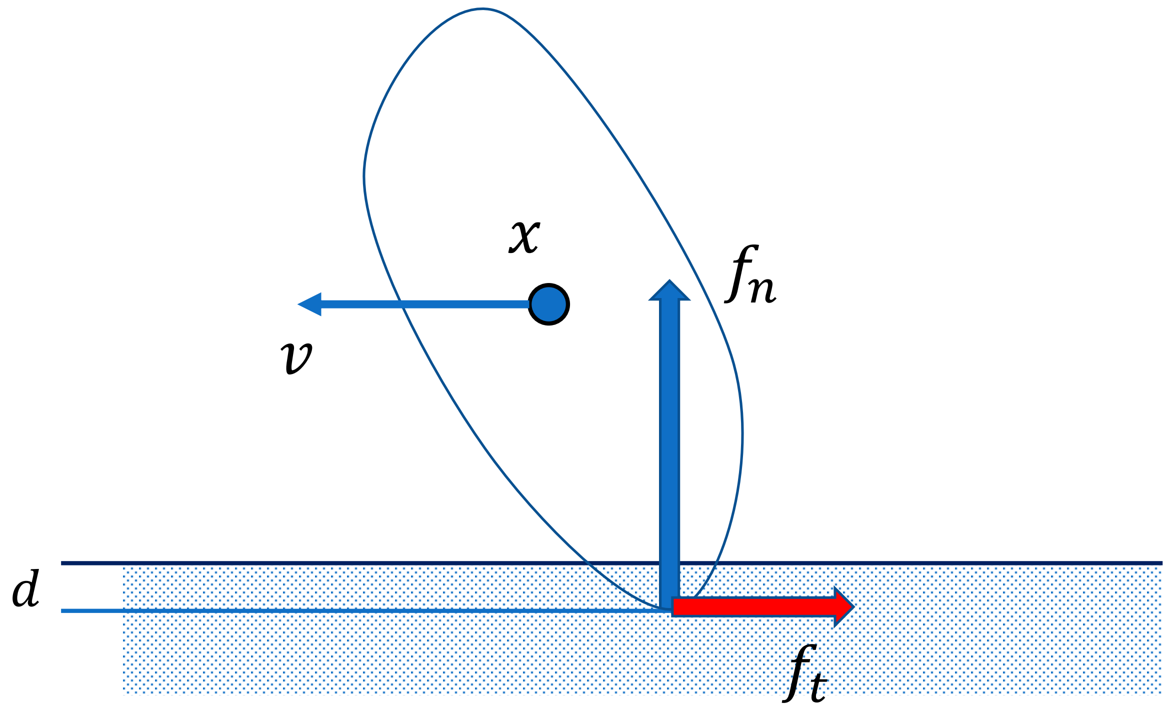
Frictional Contact





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

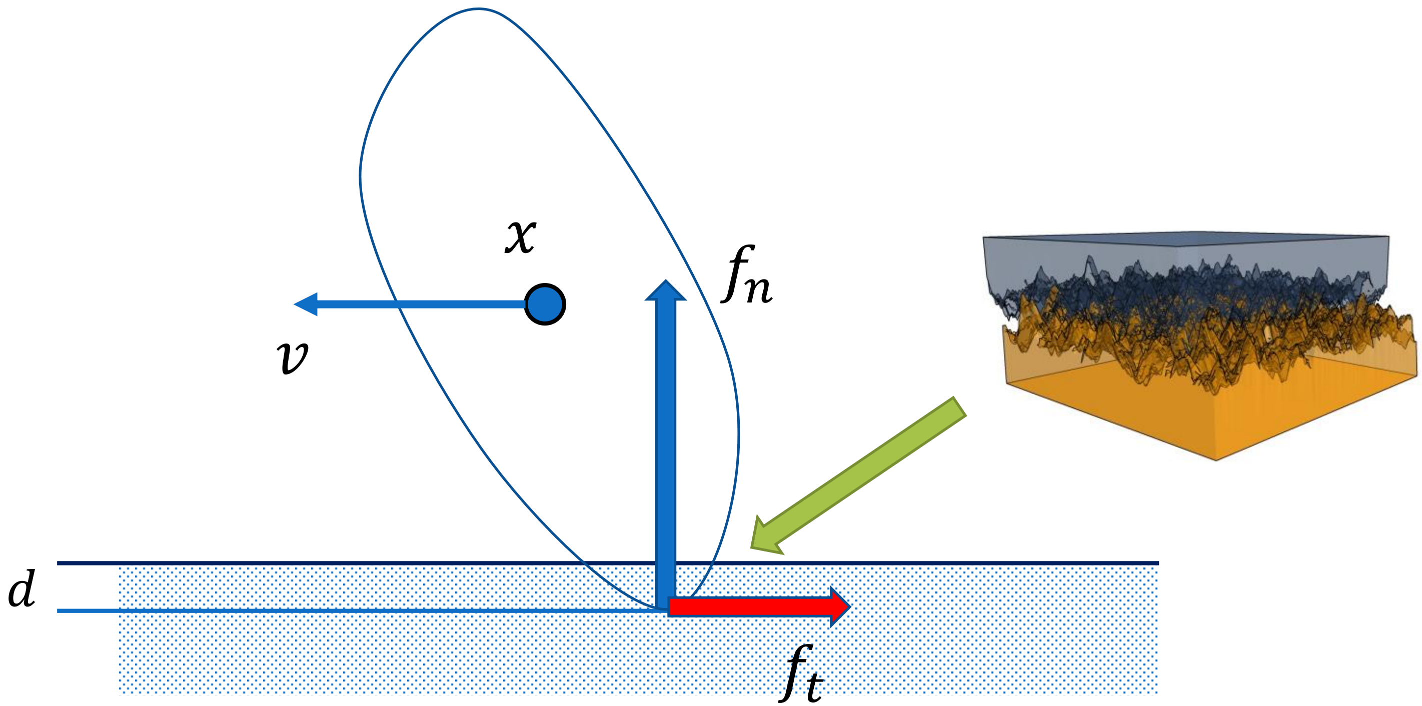
Frictional Contact



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

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

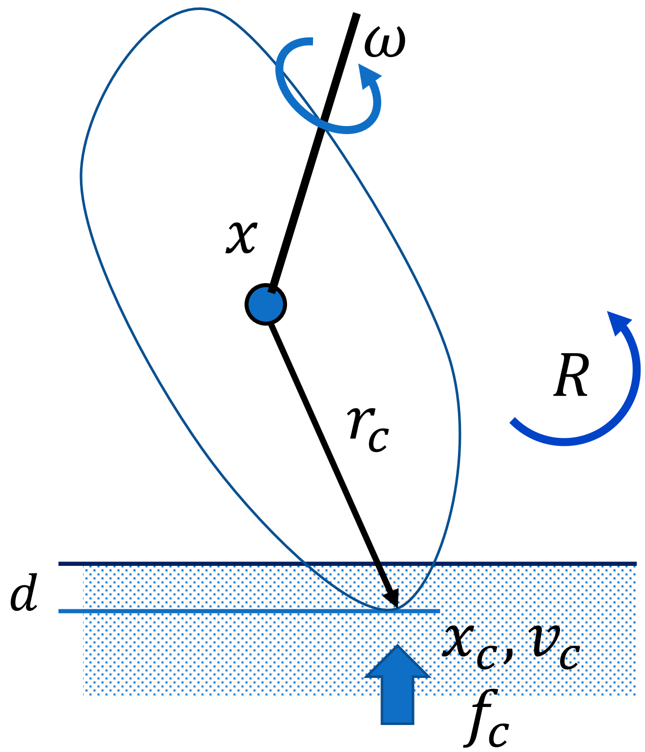
Frictional Contact



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

How to model static friction???

Contact as a Constraint

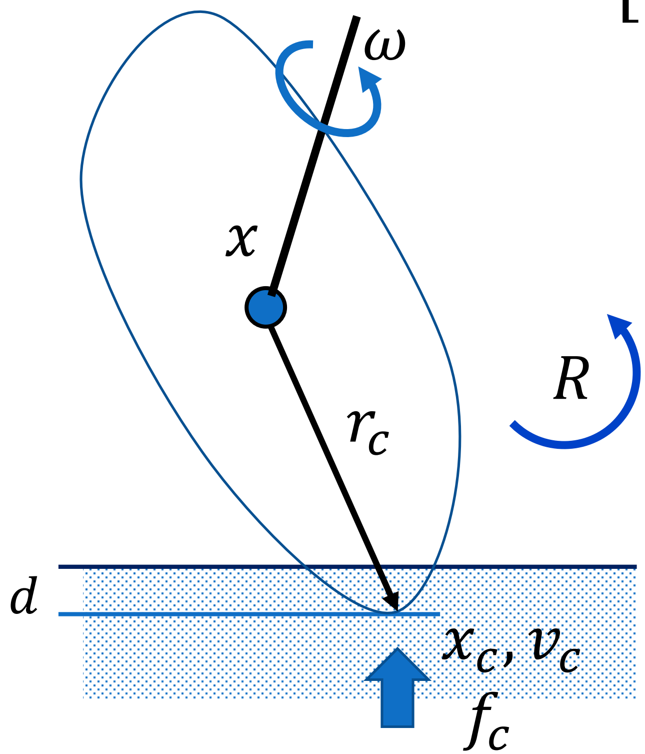


$$x_c = x + r_c$$

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

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

Contact as a Constraint

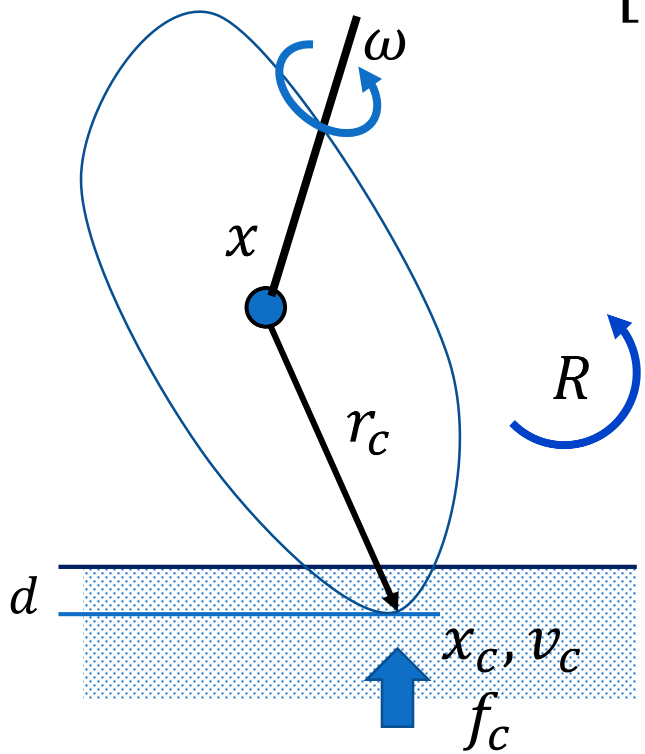


$$\begin{bmatrix} m\mathrm{I}_3 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} f \\ \tau \end{bmatrix} + J_c^T \lambda$$

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

$$\lambda \geq 0$$

Contact as a Constraint



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

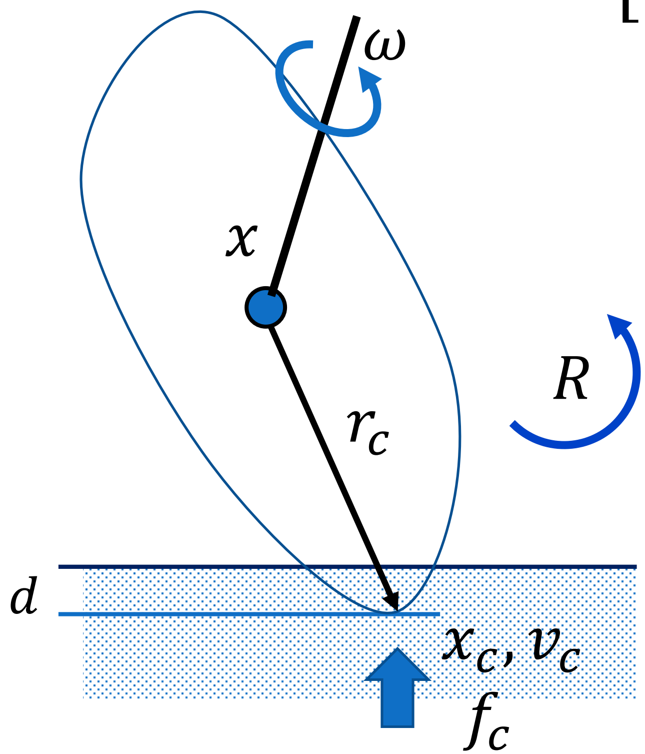
$$v_c = J_c \begin{bmatrix} v \\ \omega \end{bmatrix} \geq 0$$

$$\lambda \geq 0$$

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

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

Contact as a Linear Complementary Problem



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

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

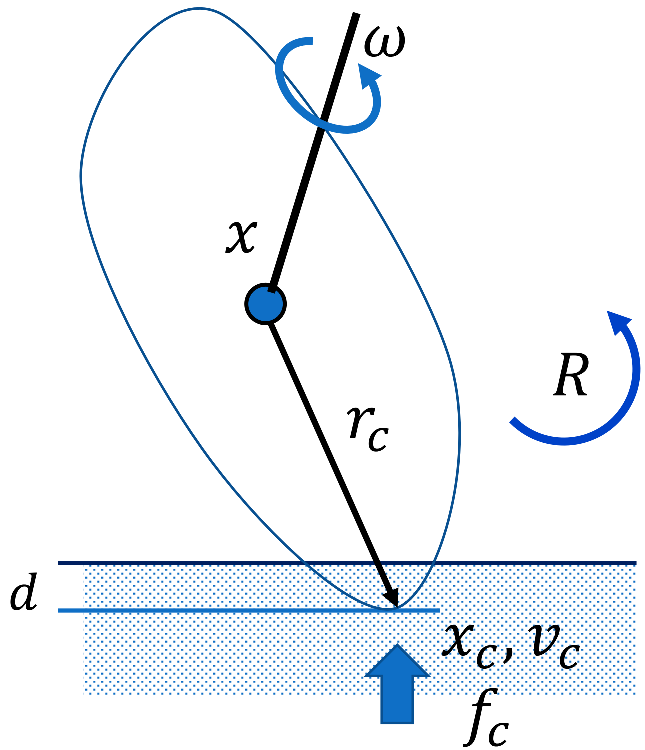
$$\lambda \geq 0$$

$$v_c \perp \lambda = 0$$

(Mixed) Linear Complementary Problem (LCP)

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

Contact as a Linear Complementary Problem



How to deal the friction?

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

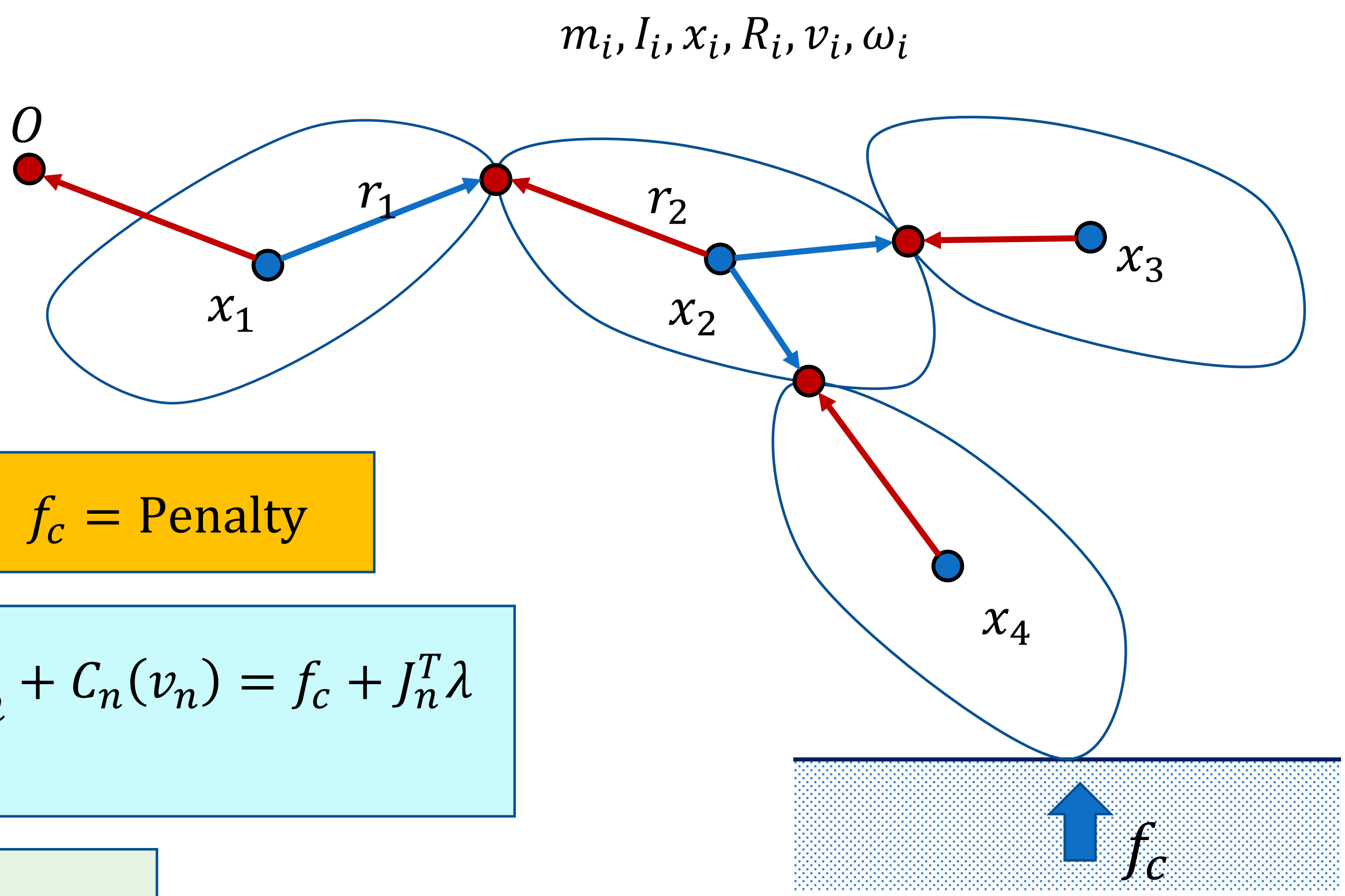
Lötstedt[10] represents the first attempt to compute friction forces in an analytical setting, by using quadratic programming to compute friction forces based on a simplification of the Coulomb friction model. Baraff[3] also proposed analytical methods for dealing with friction forces and presents algorithms 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

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

Simulation of a Rigid Body System



$I_n = R_n I_0 R_n^T$

$f_c = \text{Penalty}$

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

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

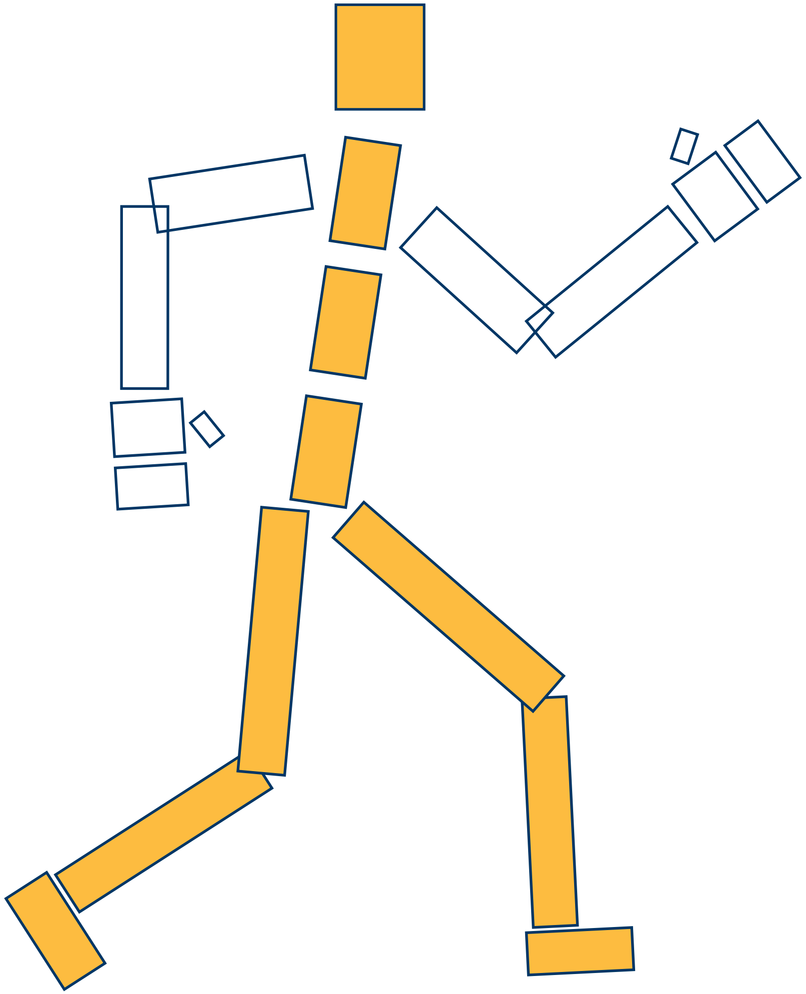
$$x_{n+1} = x_n + h v_{n+1}$$

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

Any Questions?

Physics-based Characters

An Articulated Character



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

$$Jv \geq 0$$

Ragdoll Simulation

Spider-Man: No Way Home - ragdoll simulation
https://www.youtube.com/watch?v=Yi56zagzDHY

Ragdoll Simulation

- Demo
  - https://schteppe.github.io/p2.js/demos/ragdoll.html
- Stiff vs. loose ragdoll
- Perturbed/controlled Ragdoll
  - Many-Worlds Browsing for Control of Multibody Dynamics (2:36)

Behavior Control

- hard for live and intelligent creatures
- needs full-body coordination consistent with physics constraints and achieves tasks

NaturalMotion Demos: Euphoria; Clumsy Ninja

- how about something as “simple” as walking?

walking is hard too!

- Underactuated
- Inherently unstable
- High dimensional state and action space

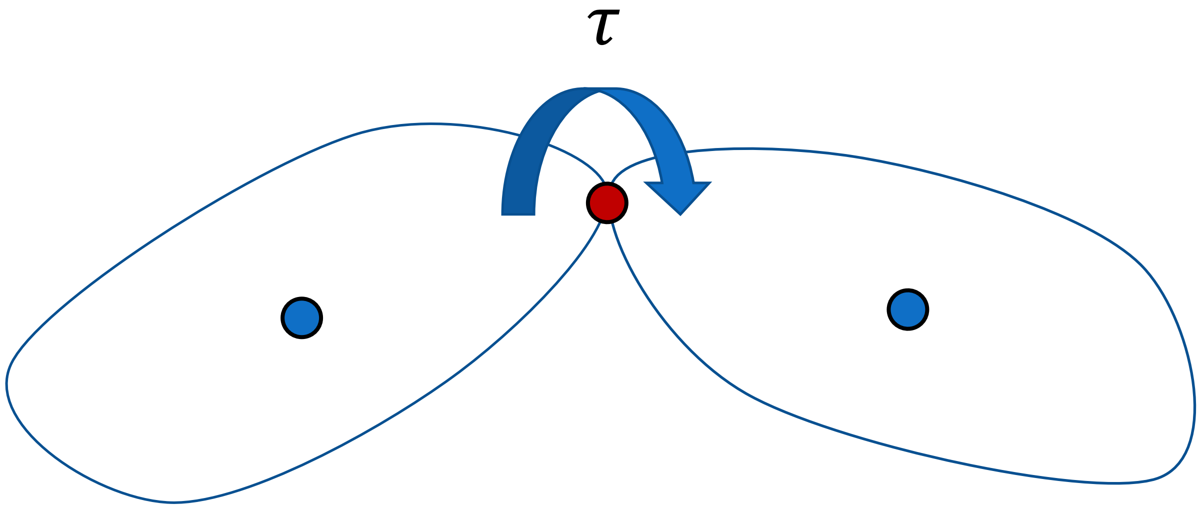
Controller → (Muscle activations / Joint torques) → Musculoskeletal system → Motion

200-400 muscles

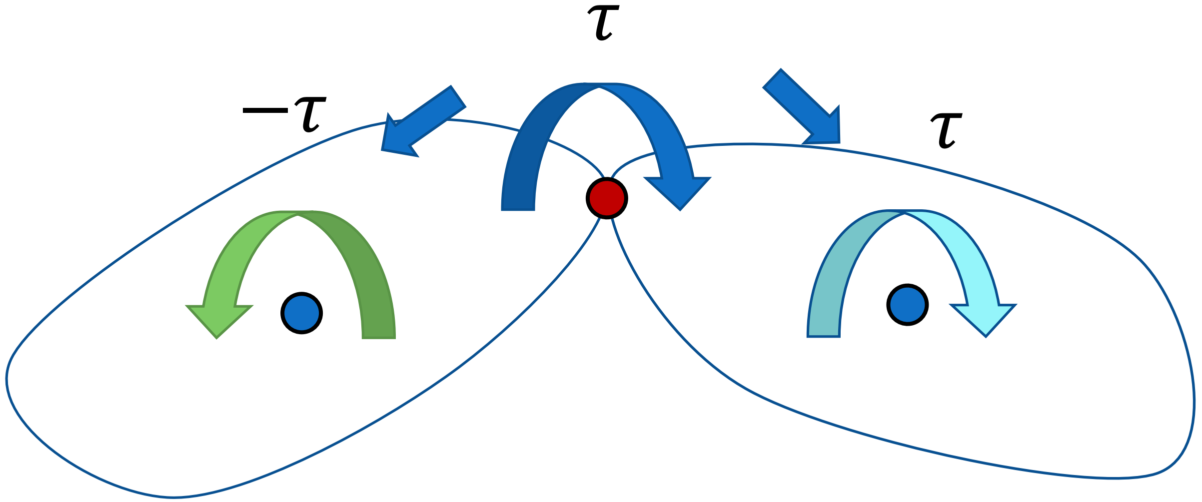
>20 joint actuators

QWOP - http://www.foddy.net/Athletics.html

Actuating a Joint

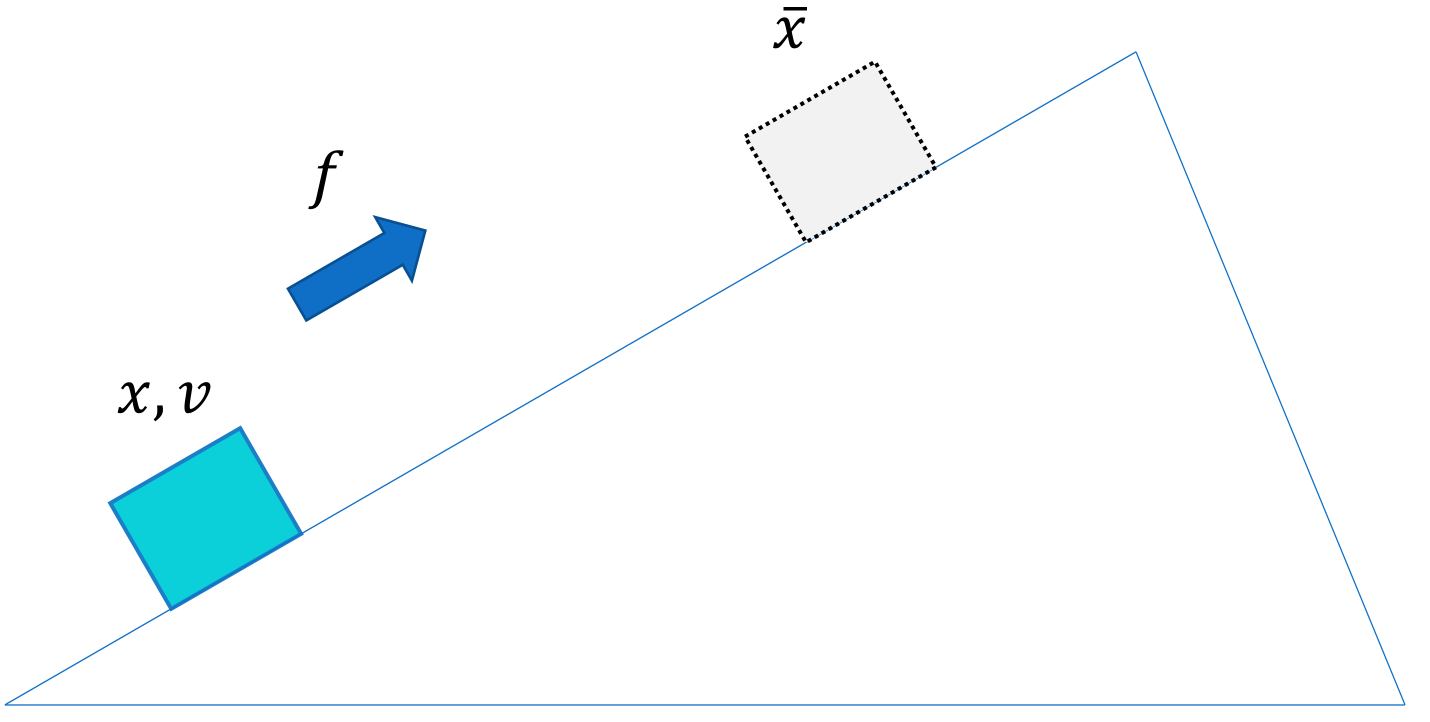


Actuating a Joint

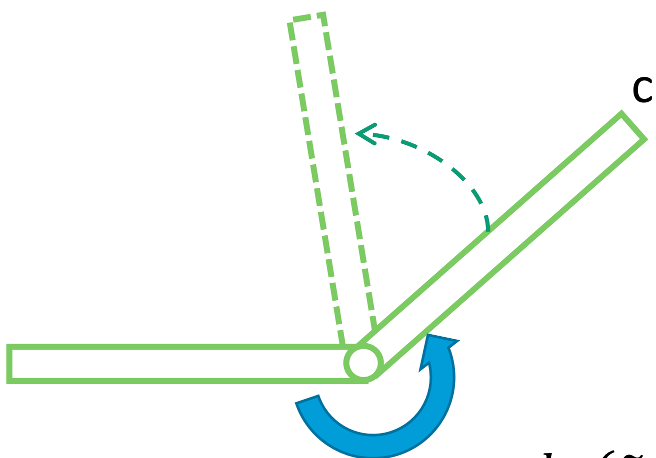


Proportional Derivative Controllers

$$f = k_p(\bar{x} - x) - k_d v$$



Proportional-Derivative (PD) Control

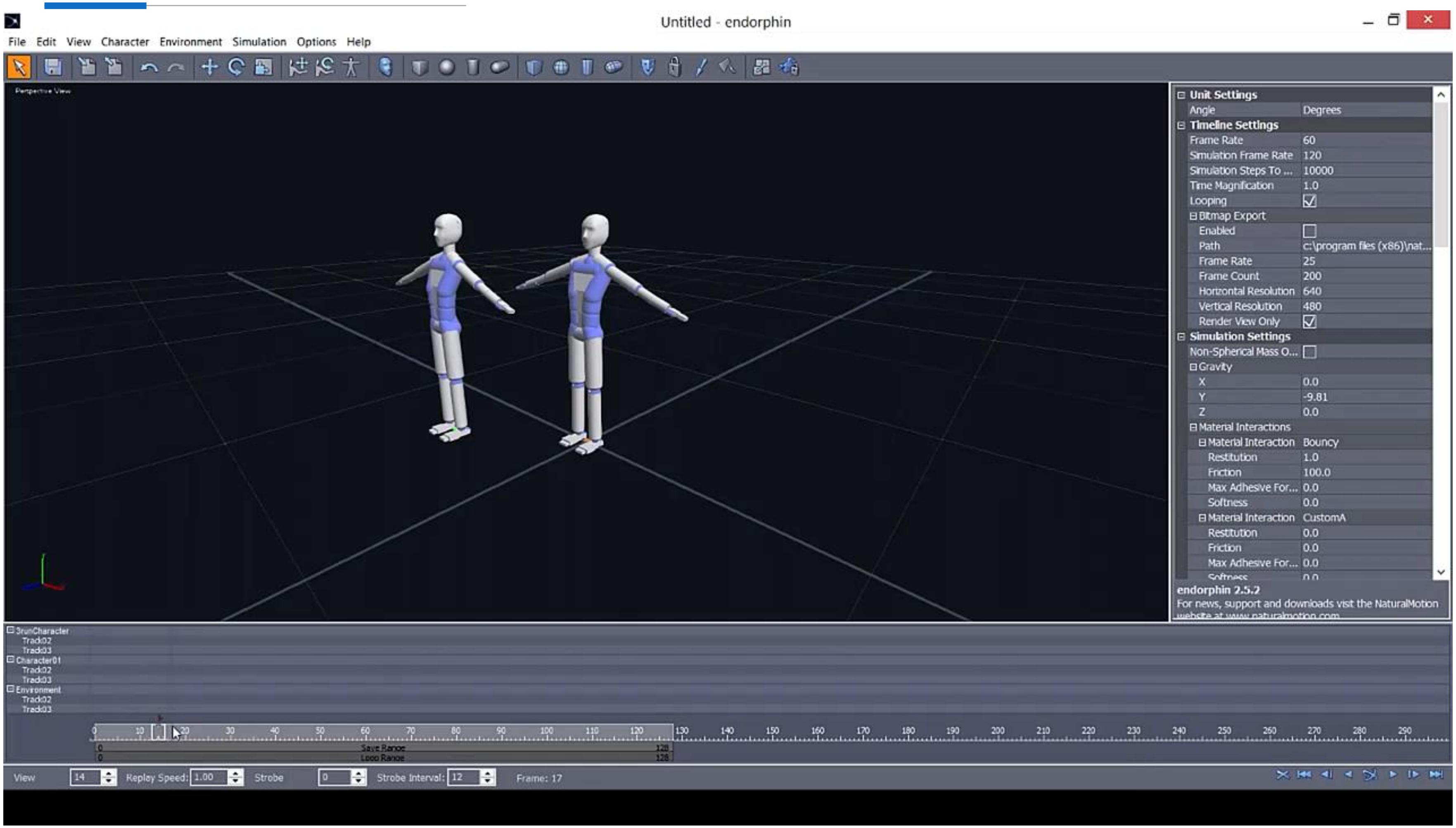


$$\tau = k_p(\tilde{q} - q) + k_d(\dot{\tilde{q}} - \dot{q})$$

derivative

proportional

Handcrafted Motion Control



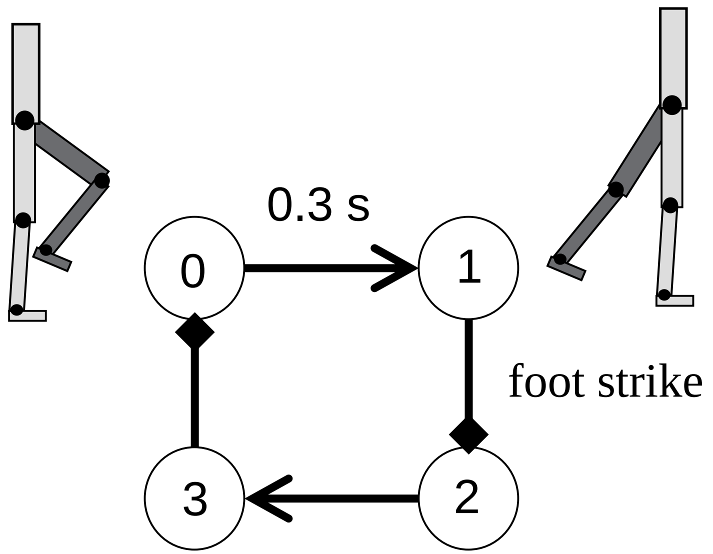
NaturalMotion - Endorphin

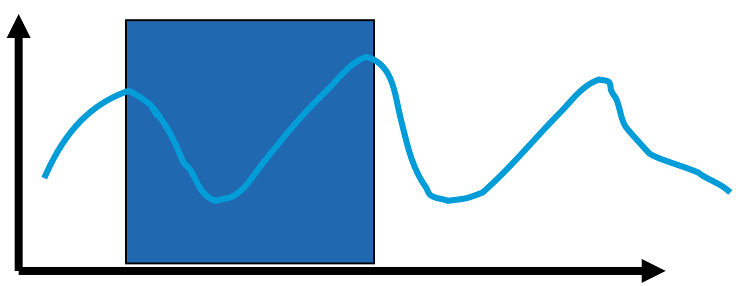
Pre-DeepRL era Walking Balance Control

- walking: series of controlled falls
- critical component: foot placement strategy
  - Simbicon (SIGGRAPH07): Linear Feedbacks
  - Generalized walking control (SIGGRAPH 10): Inverted Pendulum Models
  - Contact-aware Nonlinear Control of Dynamic Characters (SIGGRAPH 2009): Nonlinear Optimal Control

SIMBICON (SIMple BIped Locomotion CONtrol)

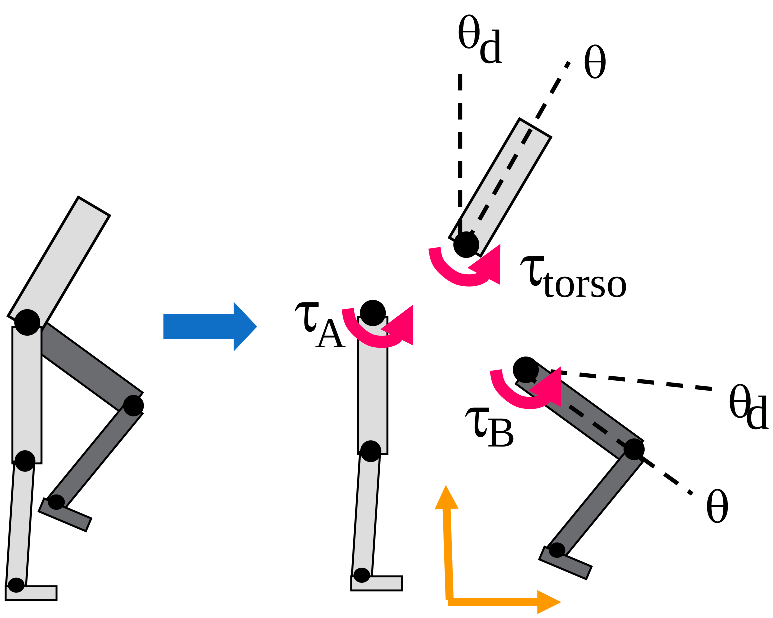
- Step 1: develop a cyclical base motion
  - PD controllers track target angles
  - FSM (Finite State Machine) or mocap





SIMBICON

- Step 2
  - control torso and swing-hip wrt world frame

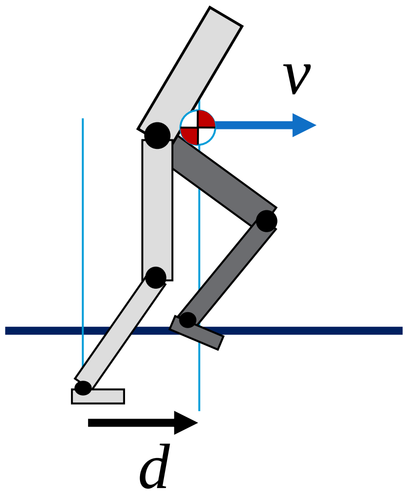


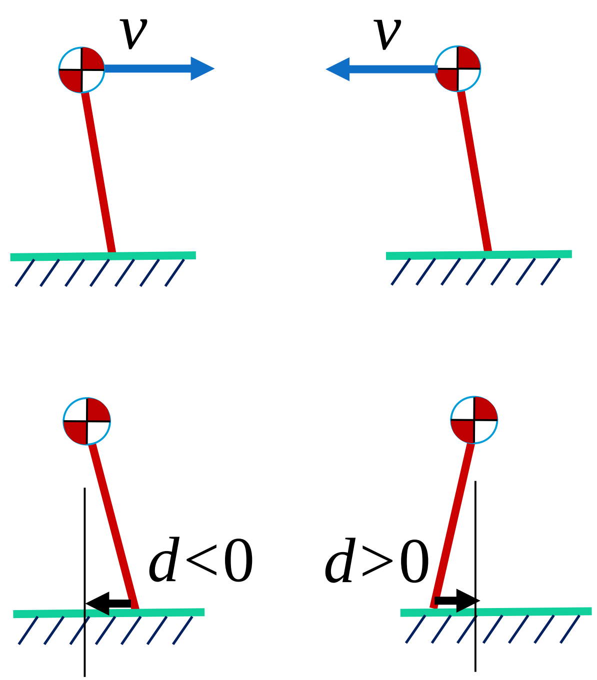
$$\tau_A = -\tau_{torso} - \tau_B$$

Newton’s third law

SIMBICON

- Step 3: COM feedback



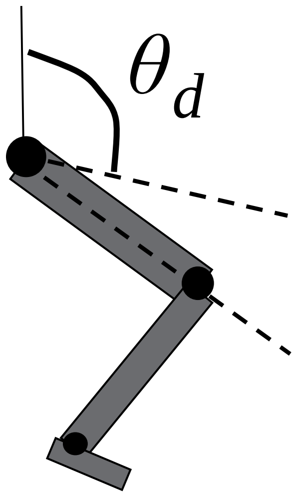


COM velocity matters

COM position matters

SIMBICON

- Step 3: COM feedback

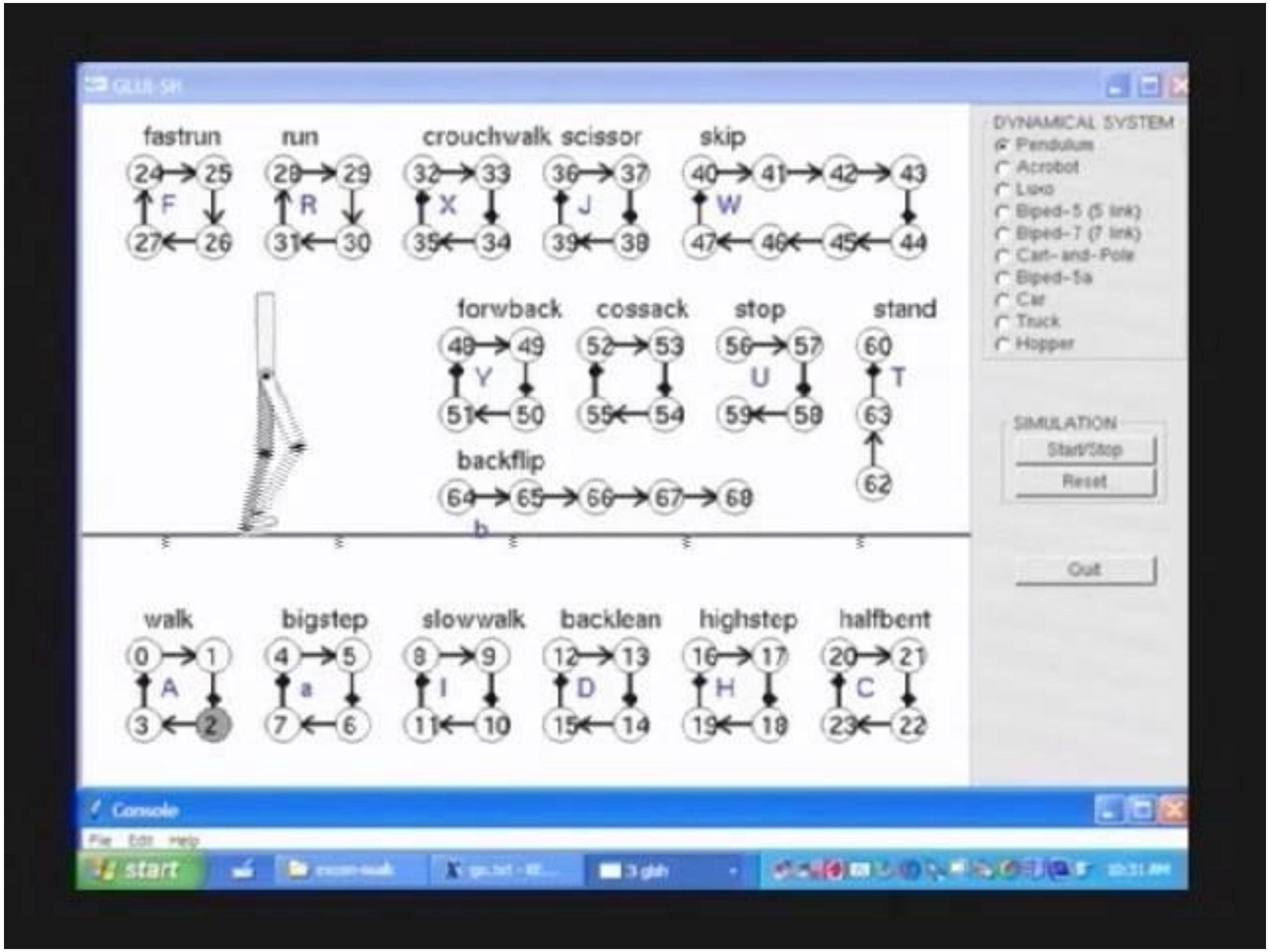


base controller | continuous feedback

$$\theta_d = \boxed{\theta_{d0}} + \boxed{c_d d + c_v v}$$

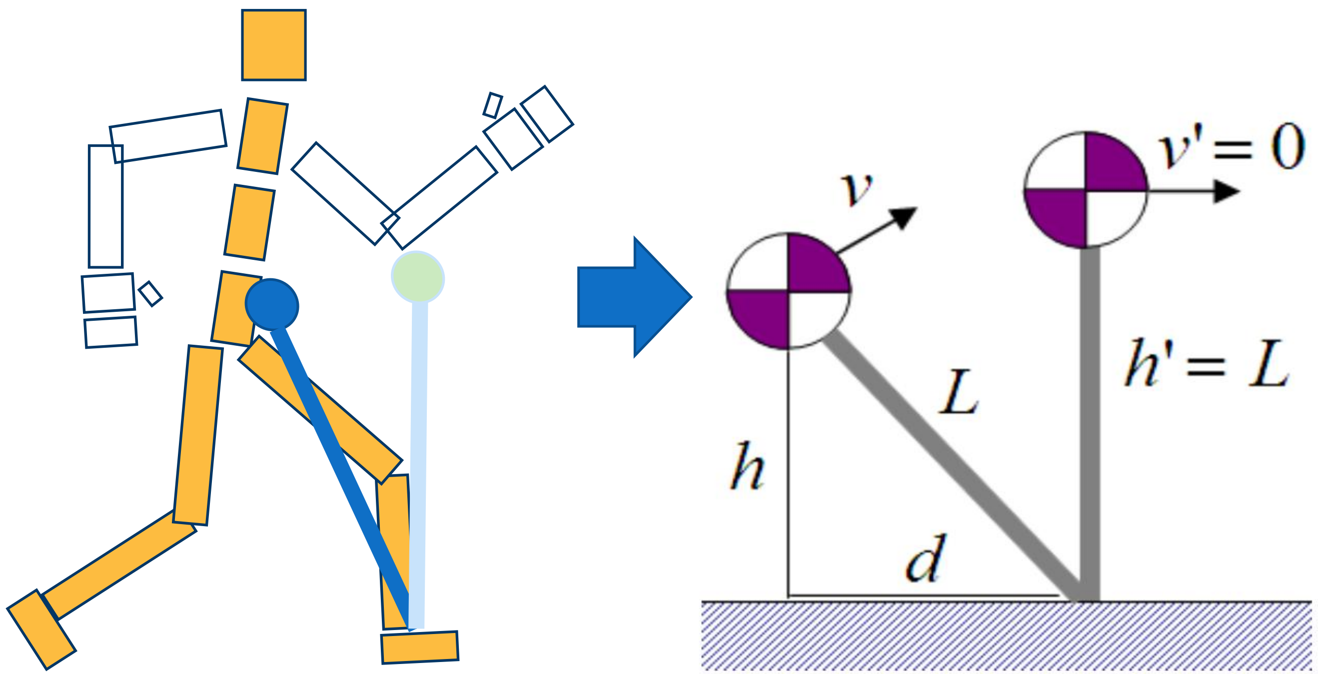
COM position | COM velocity

2D skills



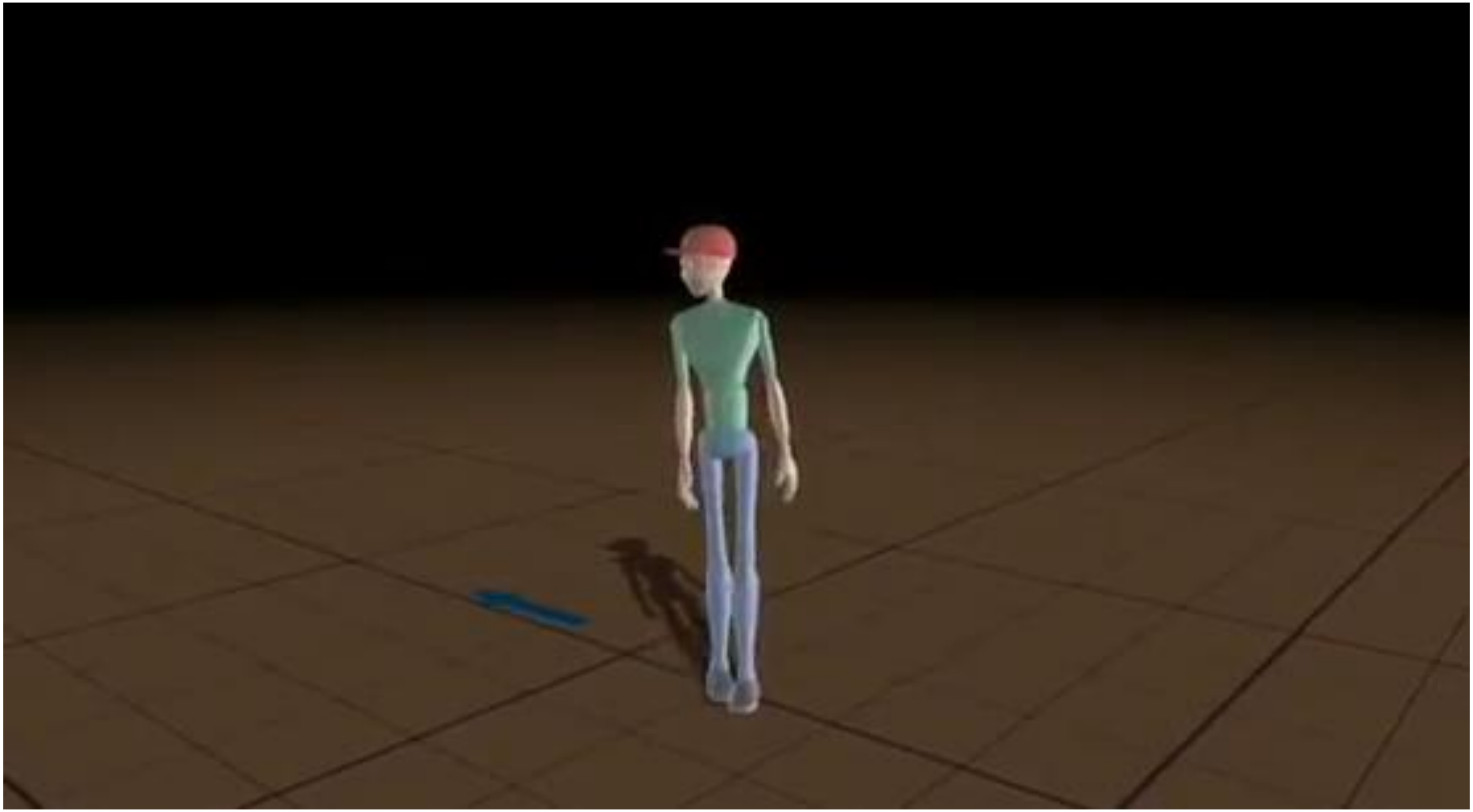
fastrun
run
crouchwalk
scissor
skip
forwback
cossack
stop
stand
backflip
walk
bigstep
slowwalk
backlean
highstep
halfbent
DYNAMICAL SYSTEM
SIMULATION
Console
start

Generalized walking control



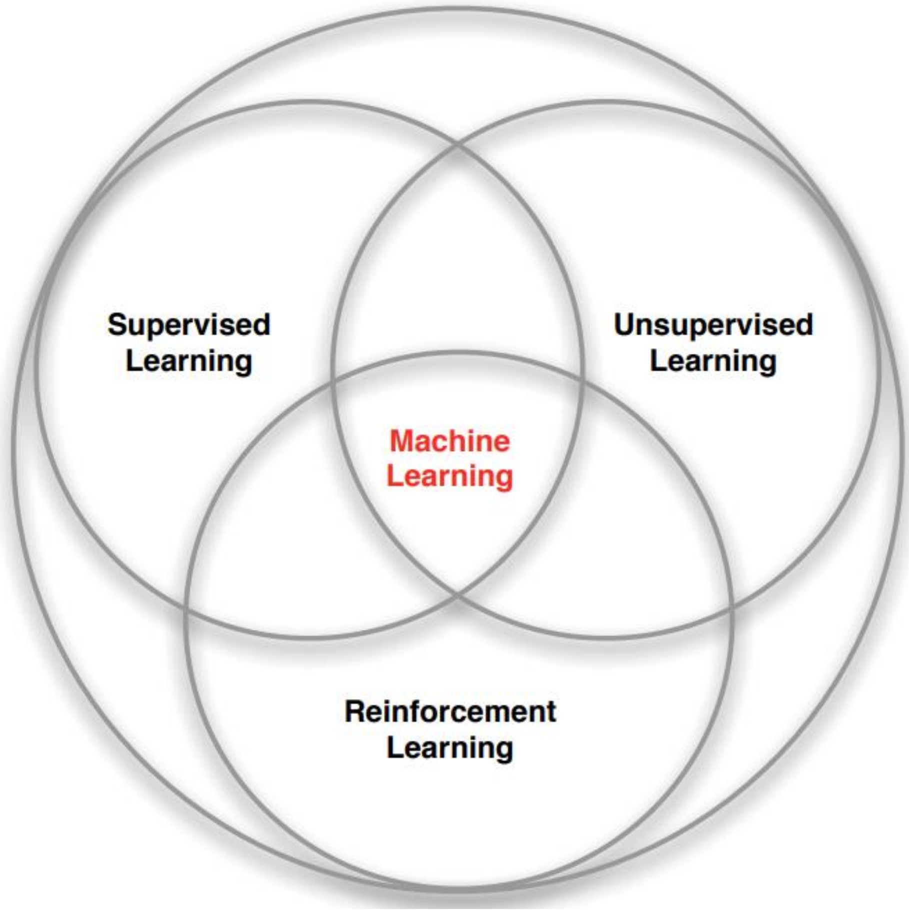
Inverted Pendulum Models

Generalized walking control

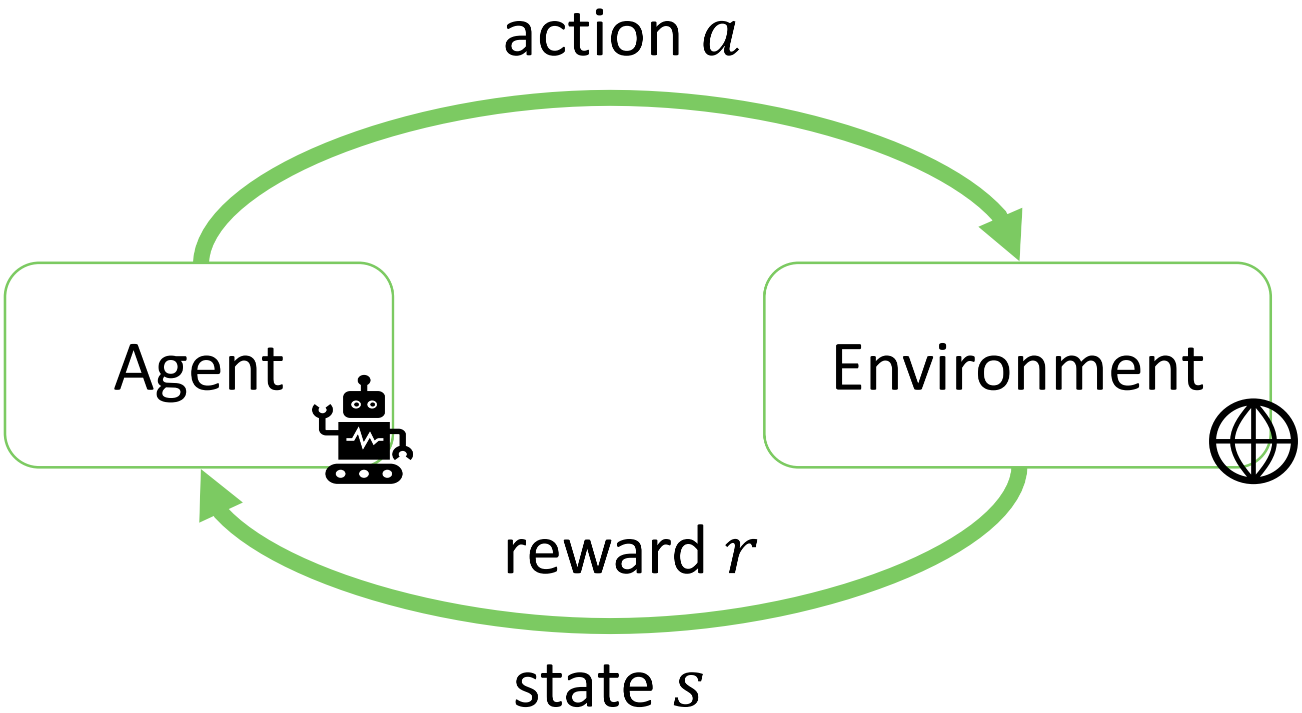


[Coros et al. 2010]

Branches of Machine Learning



Reinforcement Learning



Deep Reinforcement Learning

nature

Explore content | About the journal | Publish with us | Subscribe

nature > letters > article

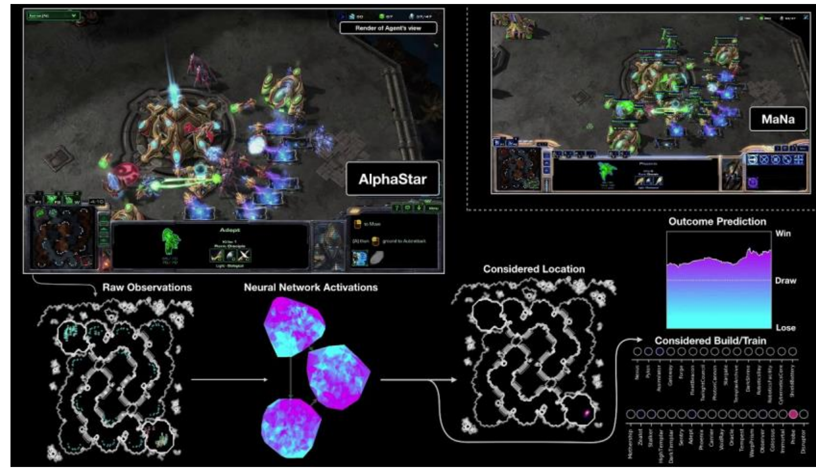
Published: 25 February 2015

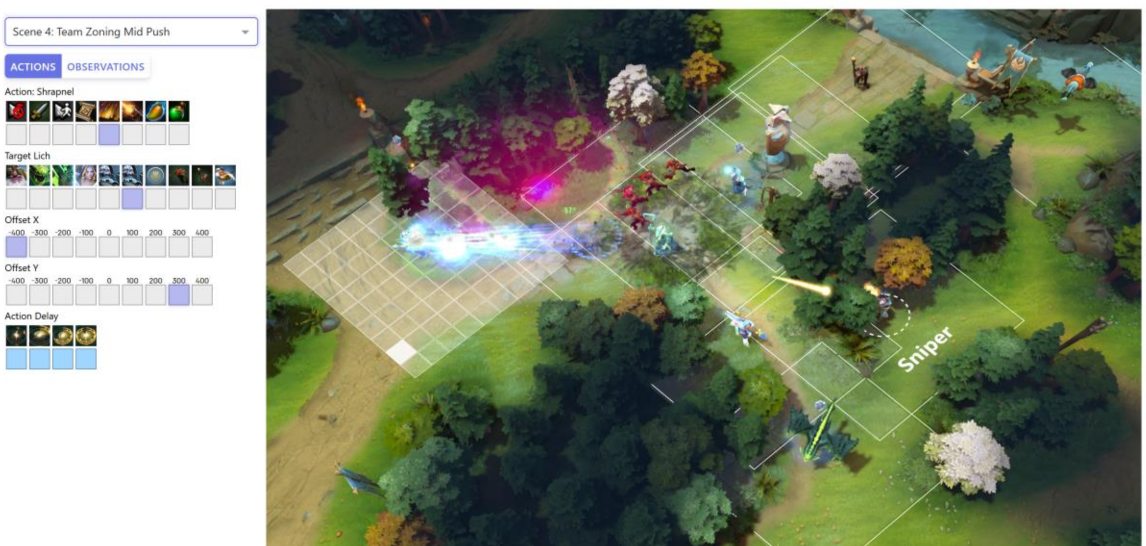
Human-level control through deep reinforcement learning

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg & Demis Hassabis

Nature **518**, 529–533 (2015) | Cite this article

Deep Reinforcement Learning

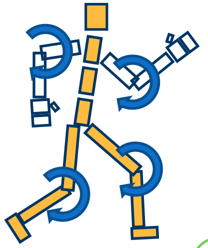


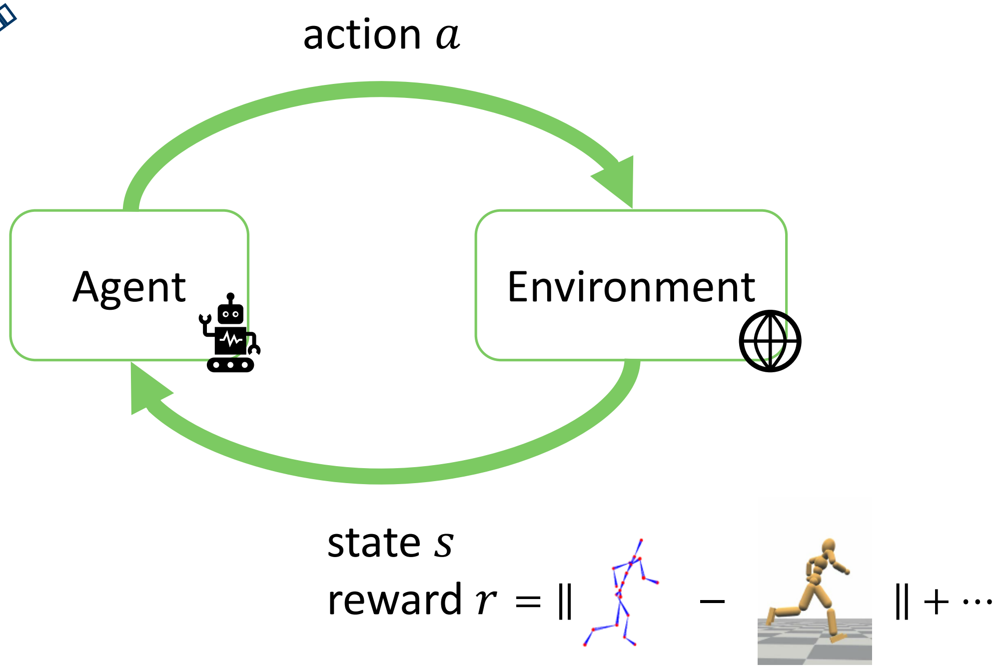


Example DRL-based Animation Research

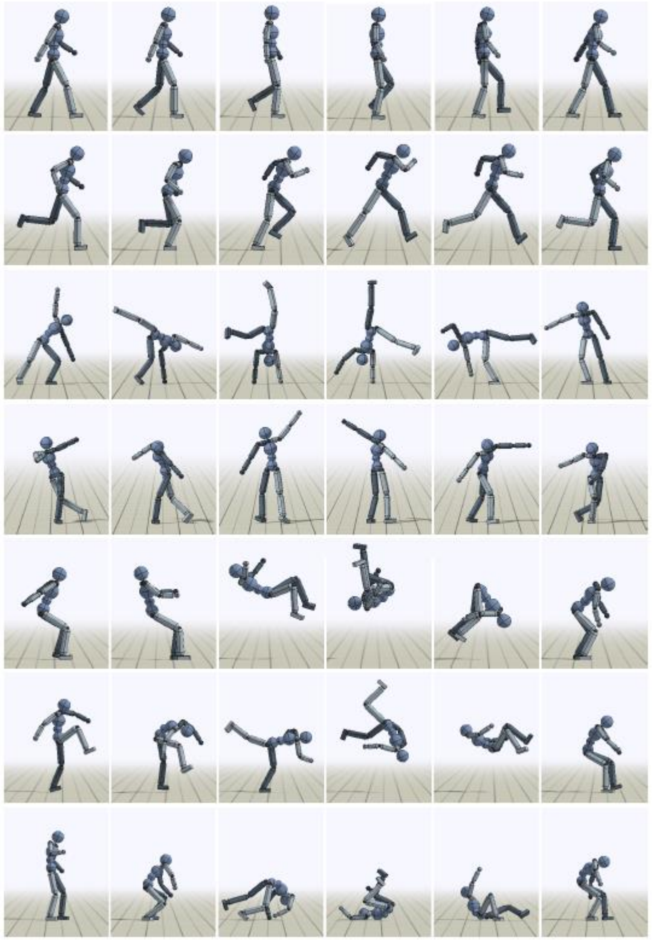
- DeepLoco (Siggraph 2017)
- DeepMimic (Siggraph 2018)
- SFV: Reinforcement Learning of Physical Skills from Videos (Siggraph Asia 2018)
- Symmetric and Low-Energy Locomotion (Siggraph 2018)
- Learning Basketball Dribbling Skills Using Trajectory Optimization and Deep Reinforcement Learning (Siggraph 2018)
- AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control (Siggraph 2021)
- Discovering Diverse Athletic Jumping Strategies (Siggraph 2021)

DeepMimic

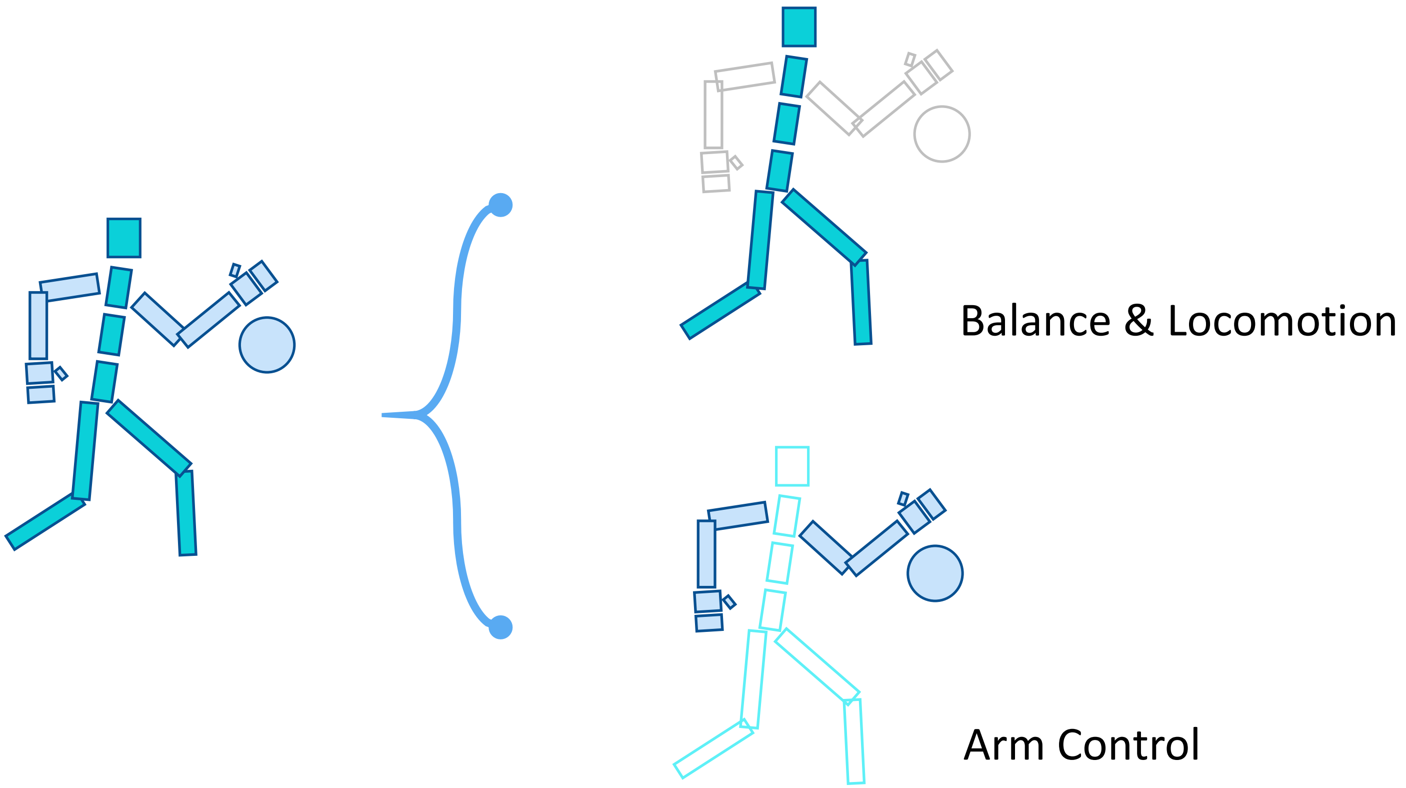




DeepMimic



Combined Control Policy

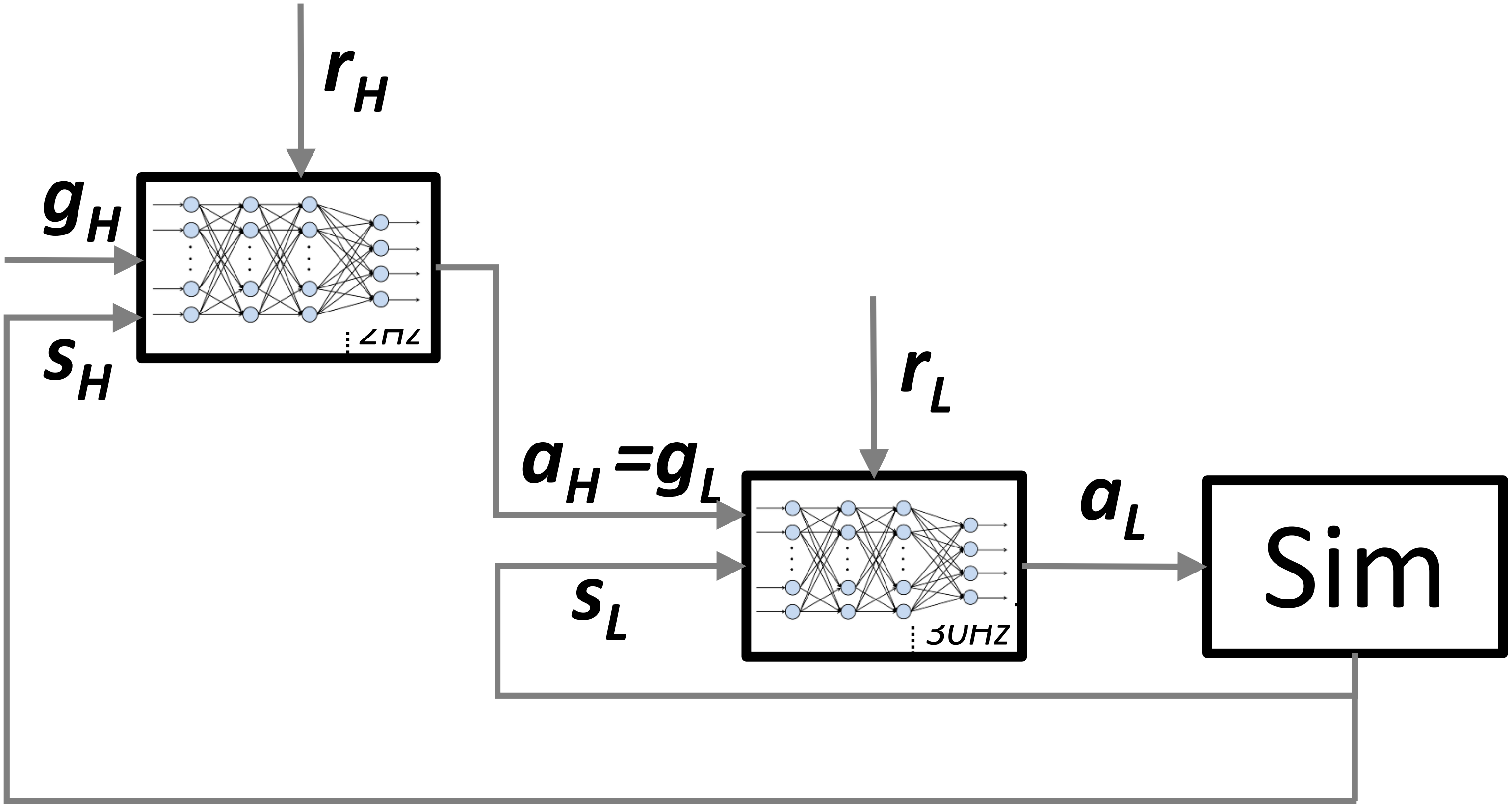


Basketball Dribbling Controllers

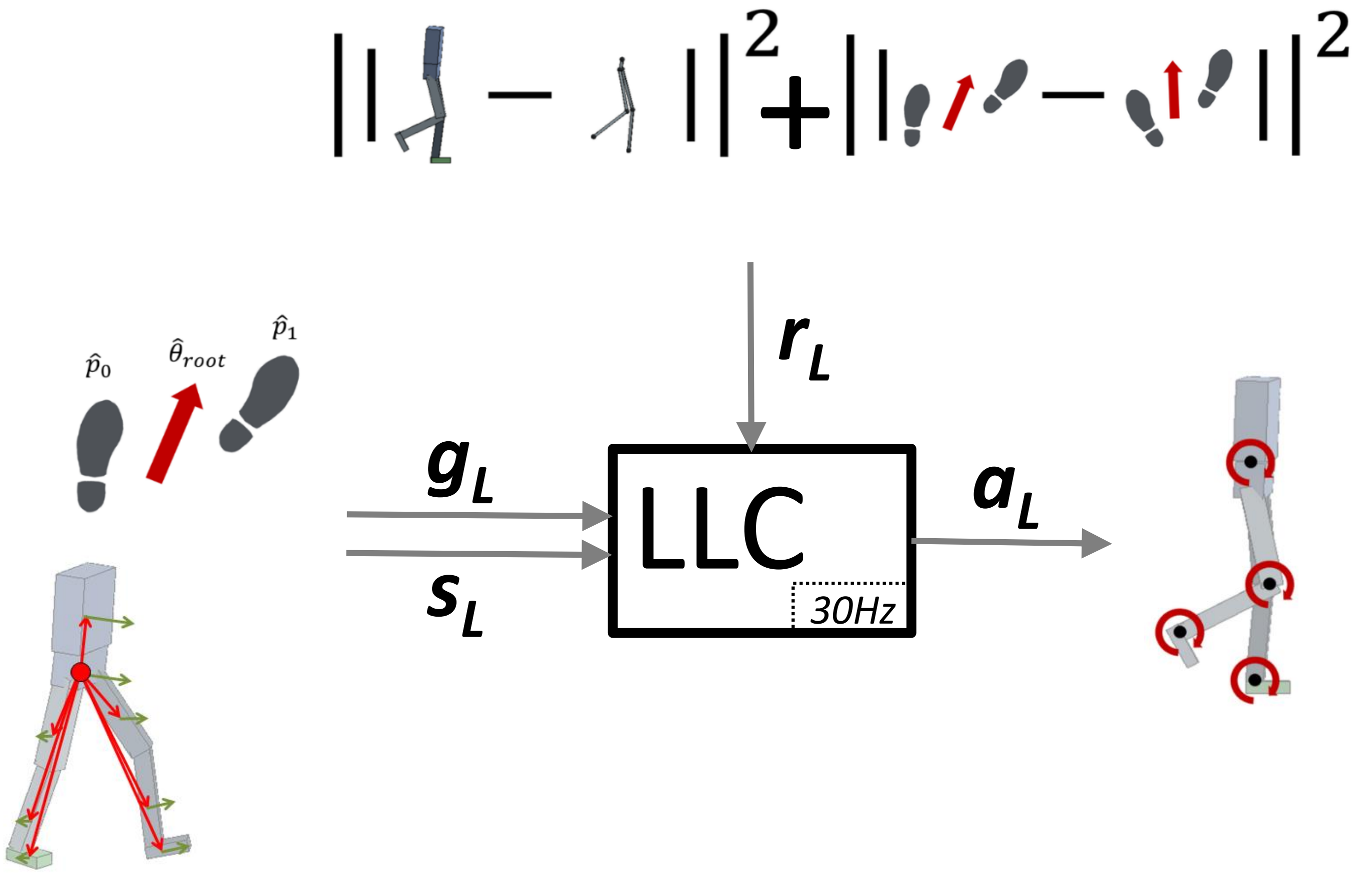
[Liu et al. 2018 (SIGGRAPH 2018)]

DeepLoco: Overview

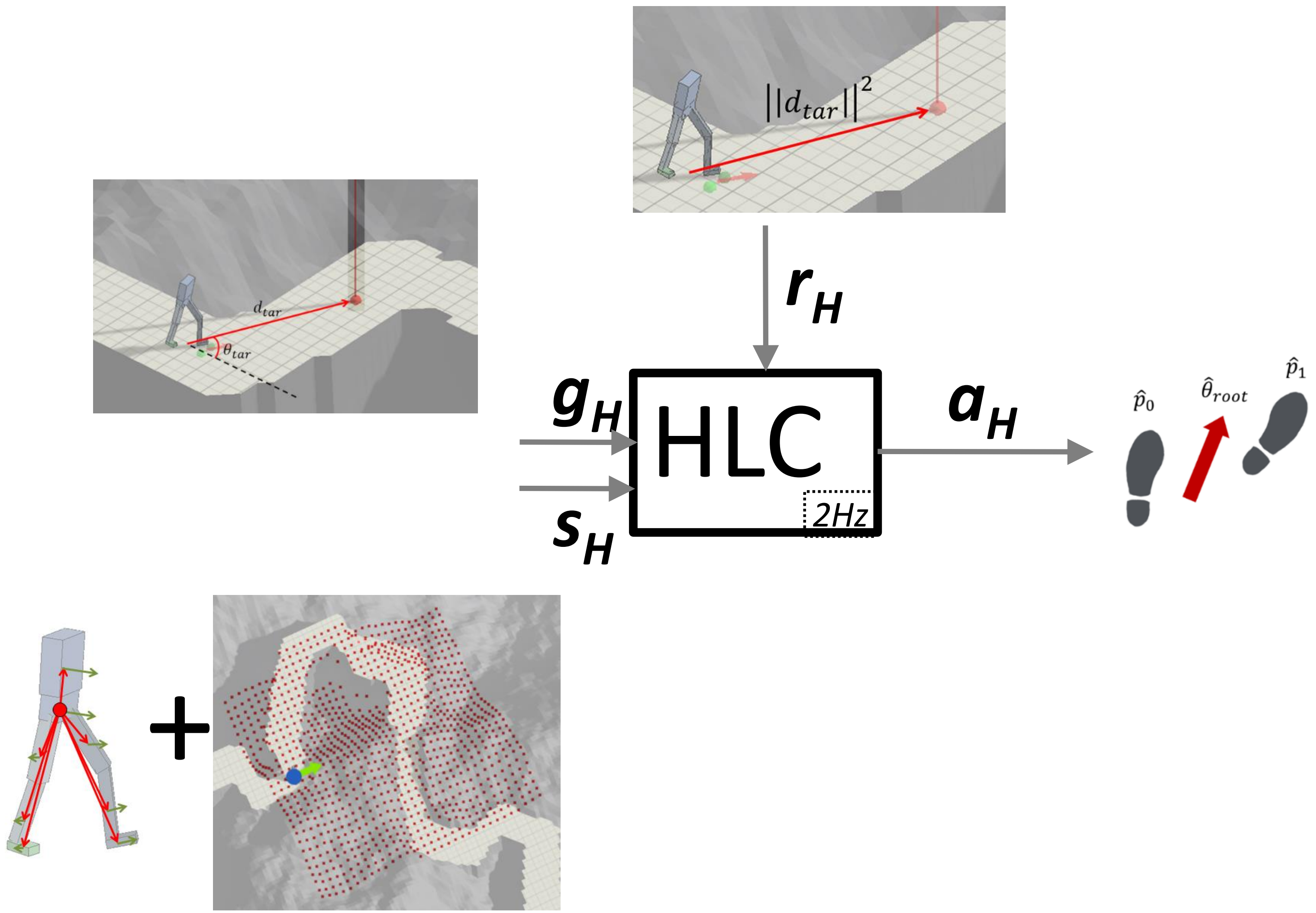
- dynamic locomotion skills using hierarchical DRL (Actor-Critic)



DeepLoco: LLC

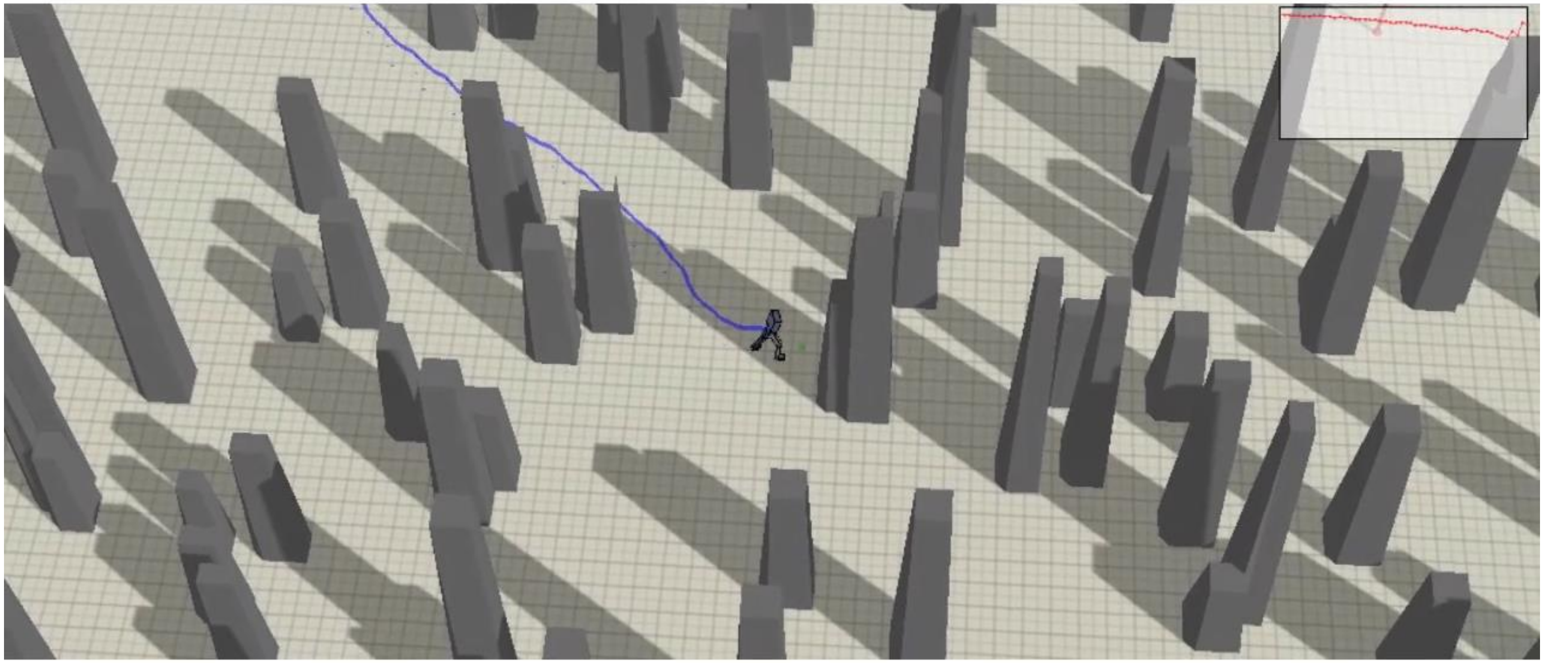


DeepLoco: HLC



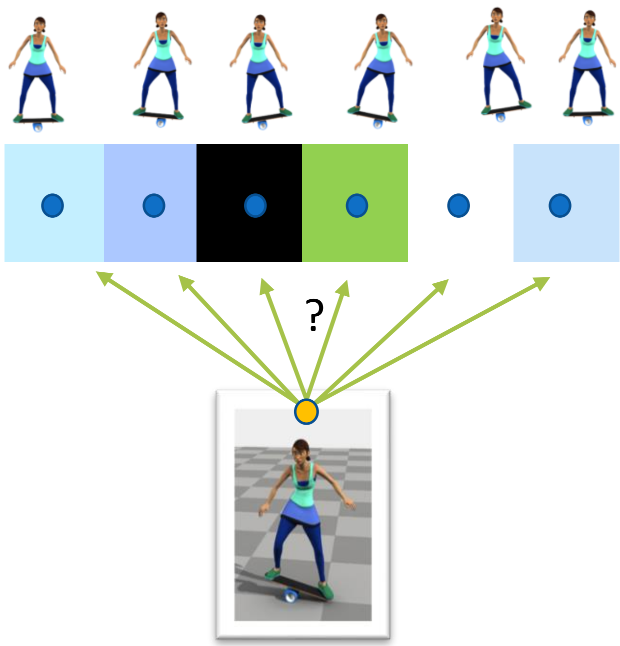
DeepLoco: Results

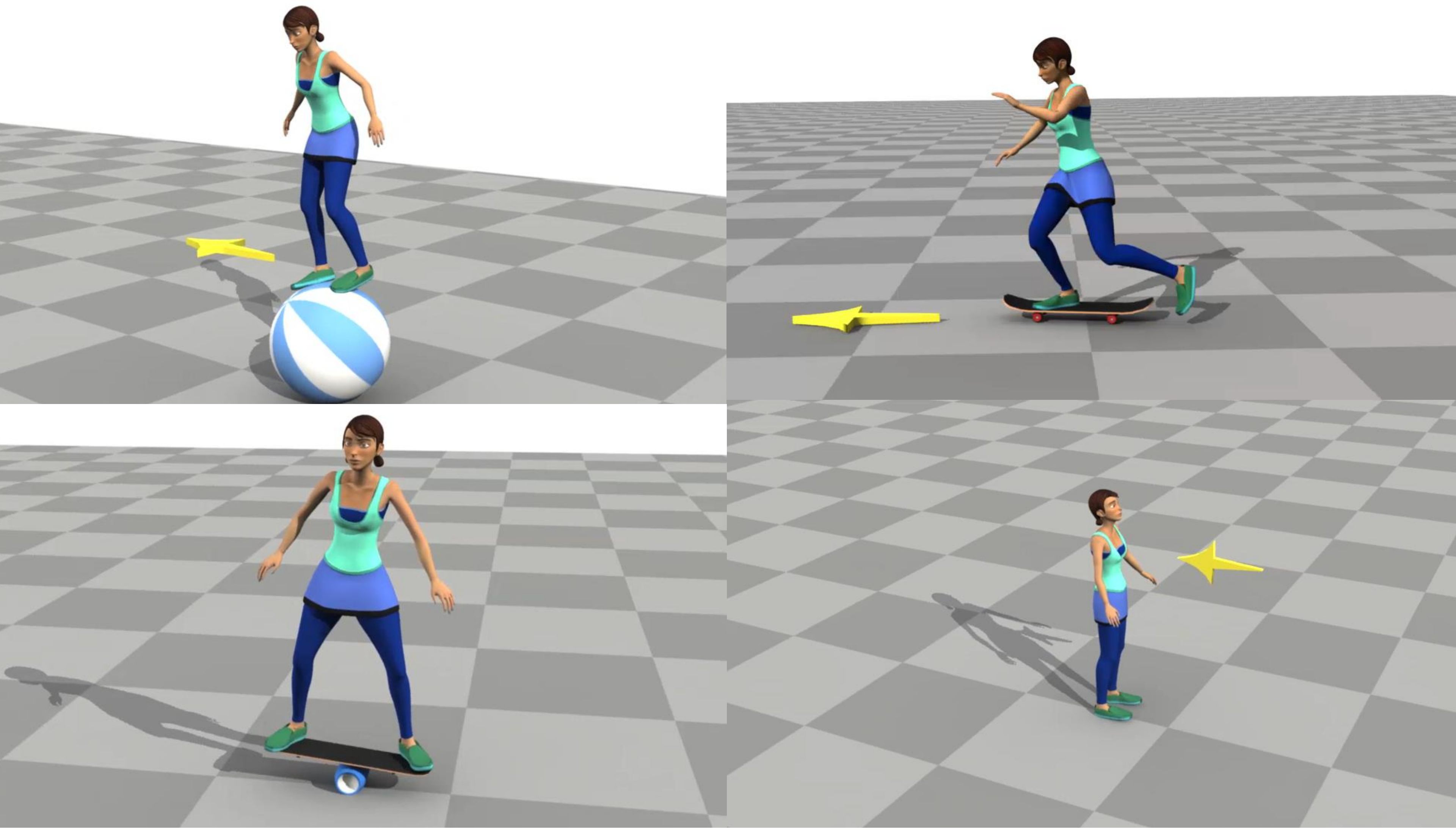
Pillars



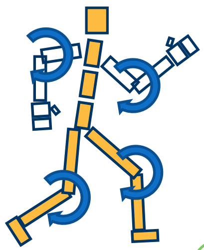
Scheduling of Control Fragments using Deep RL

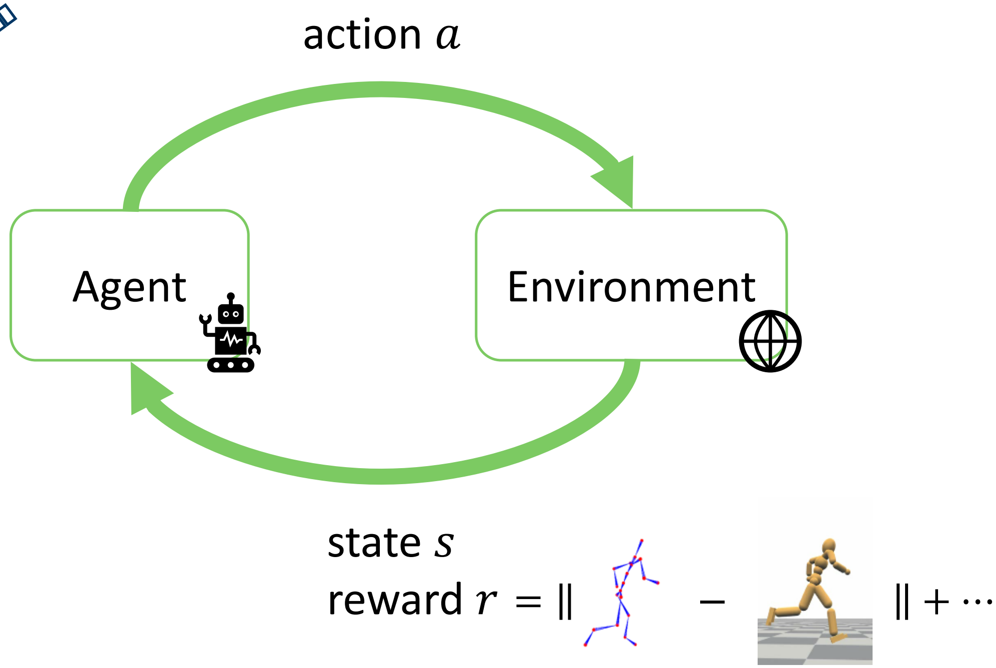
[Liu et al. 2017 (SIGGRAPH 2017)]





Generative Adversarial Imitation Learning (GAIL)





$$r(s_t, a_t) = \mathcal{D}(\tau_{sim} | \tau_{real})$$

Generative Adversarial Imitation Learning (GAIL)

[Peng et al. 2021]

Questions?