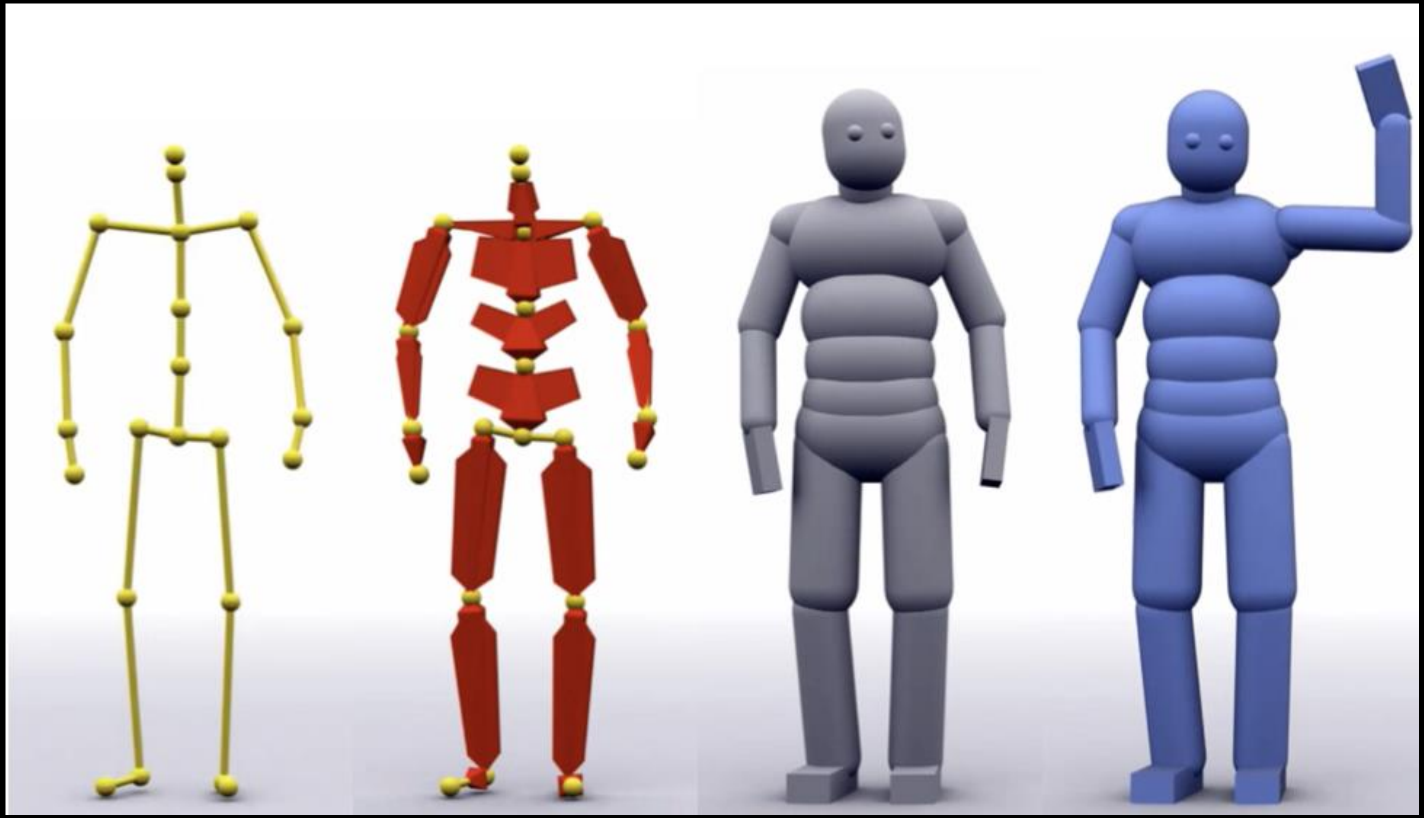
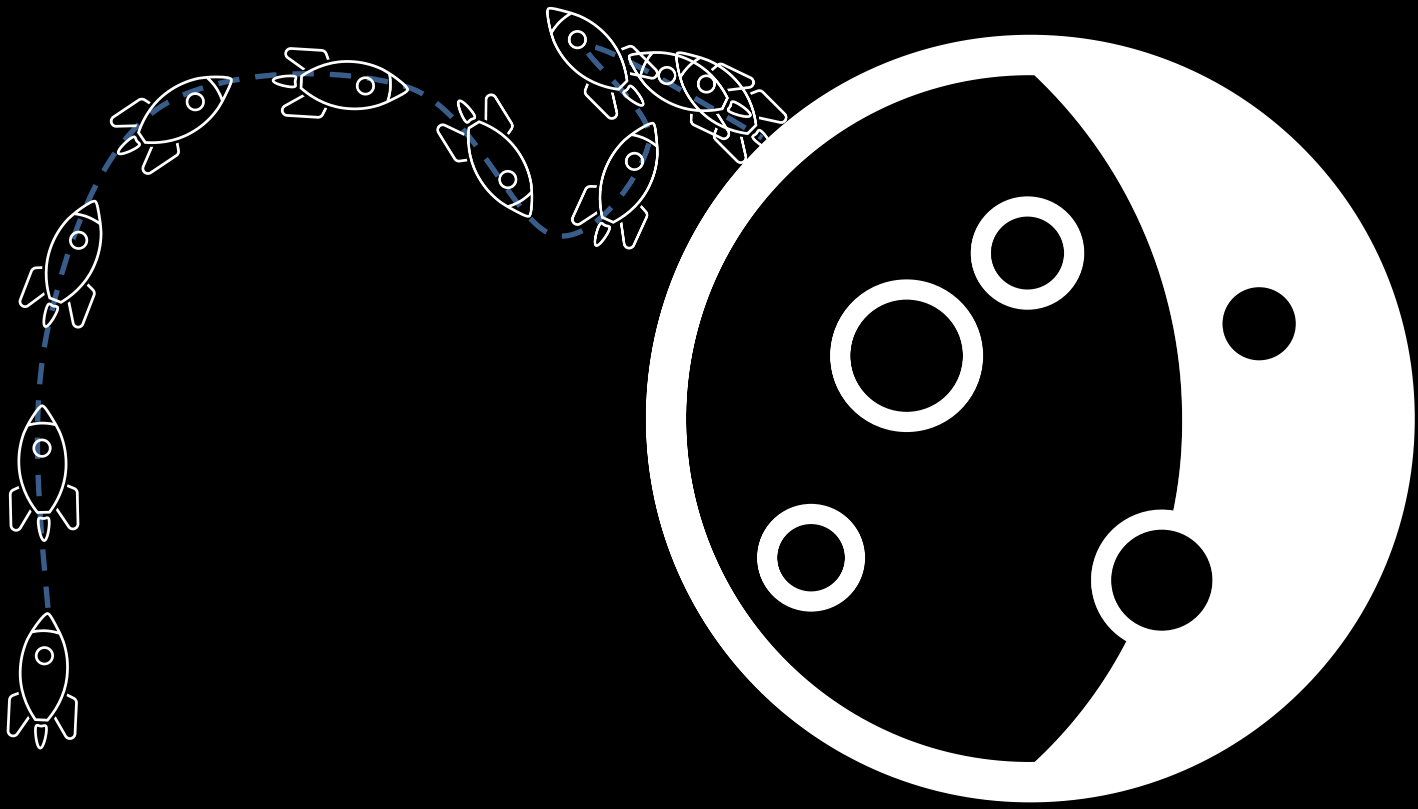
Feedback Control and Locomotion

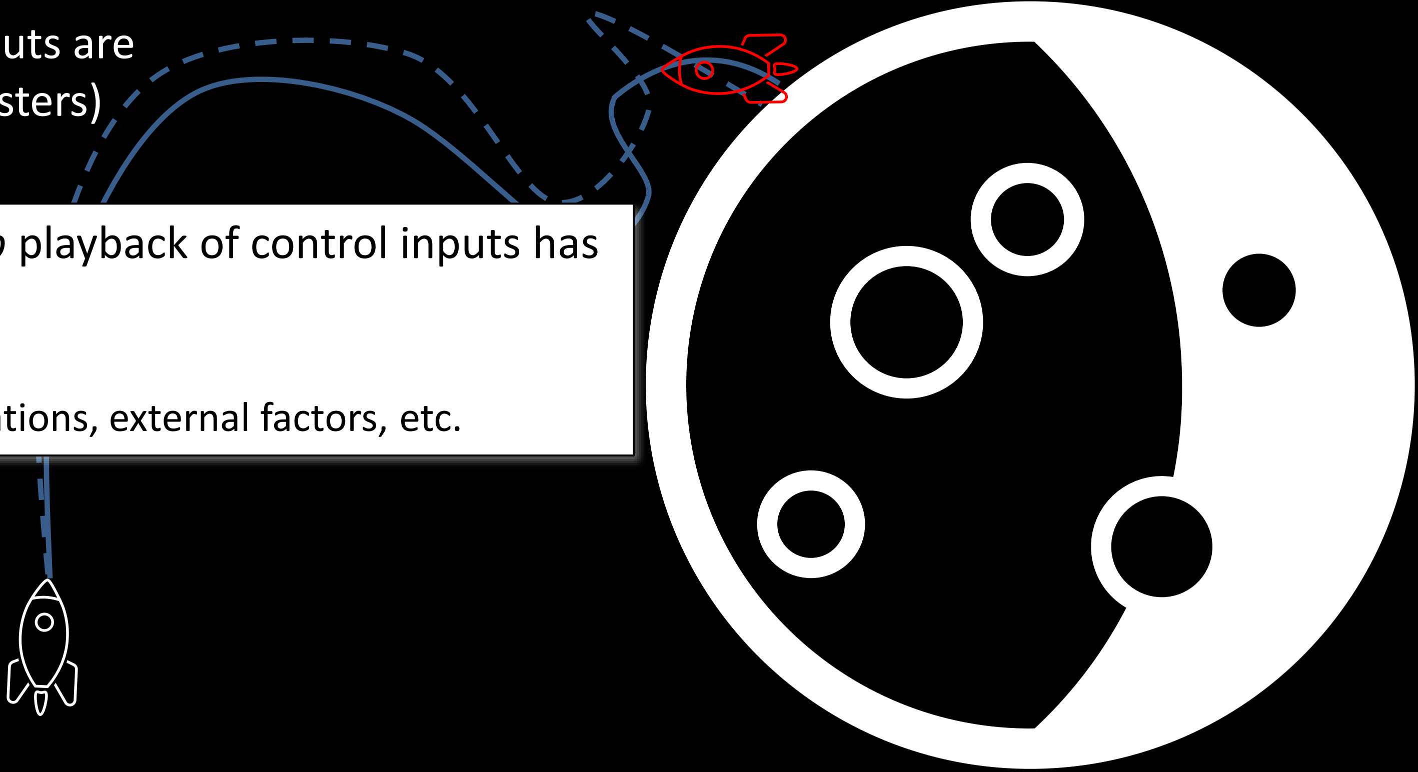


What we saw last class: trajectory optimization



What we saw last class: trajectory optimization

TO: used to *plan* trajectories.
Now: let's *execute* them!
(Let's assume control inputs are forces generated by thrusters)

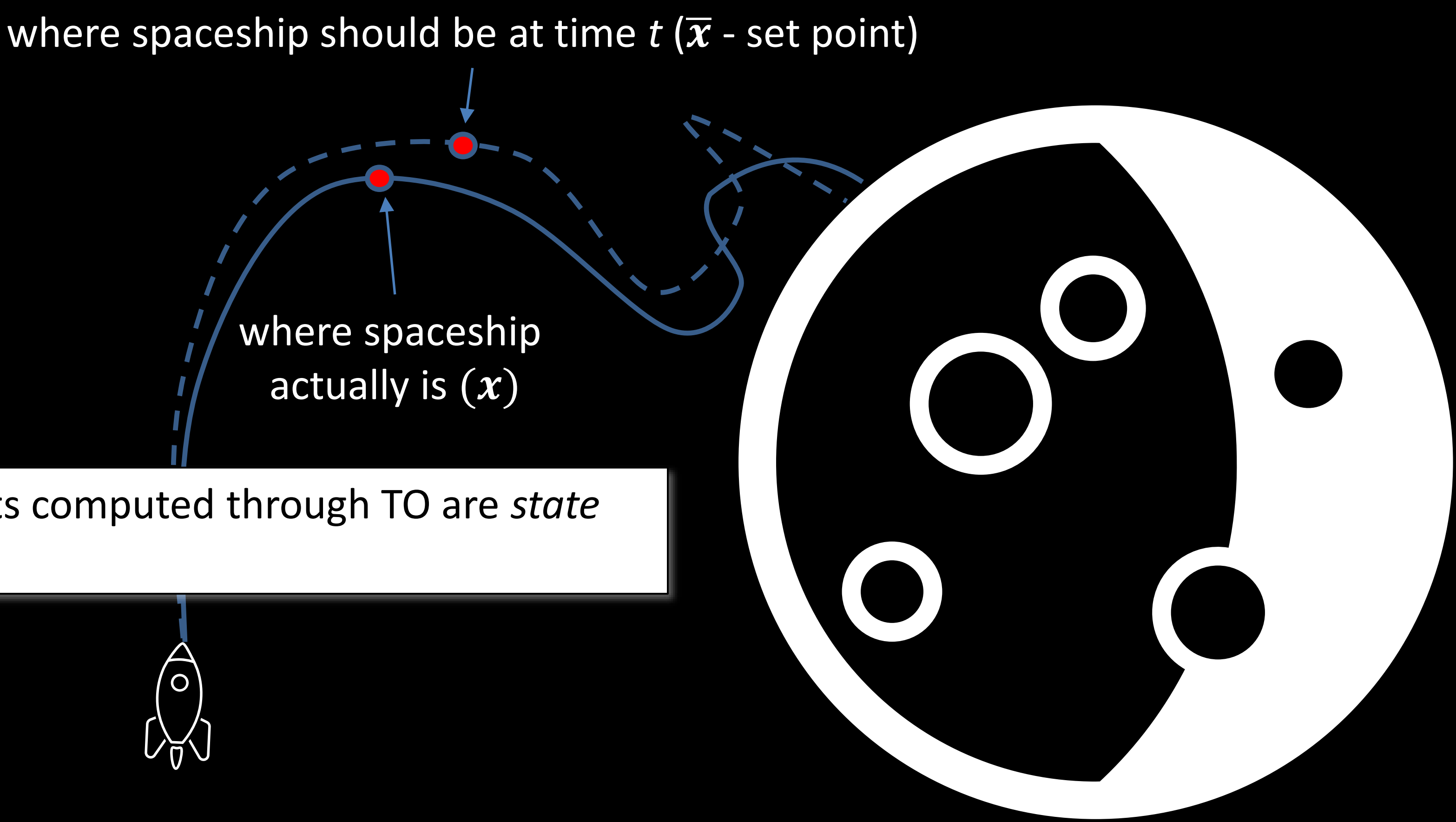


Feedforward/open loop playback of control inputs has limitations!

- Drift is inevitable
    - modeling approximations, external factors, etc.

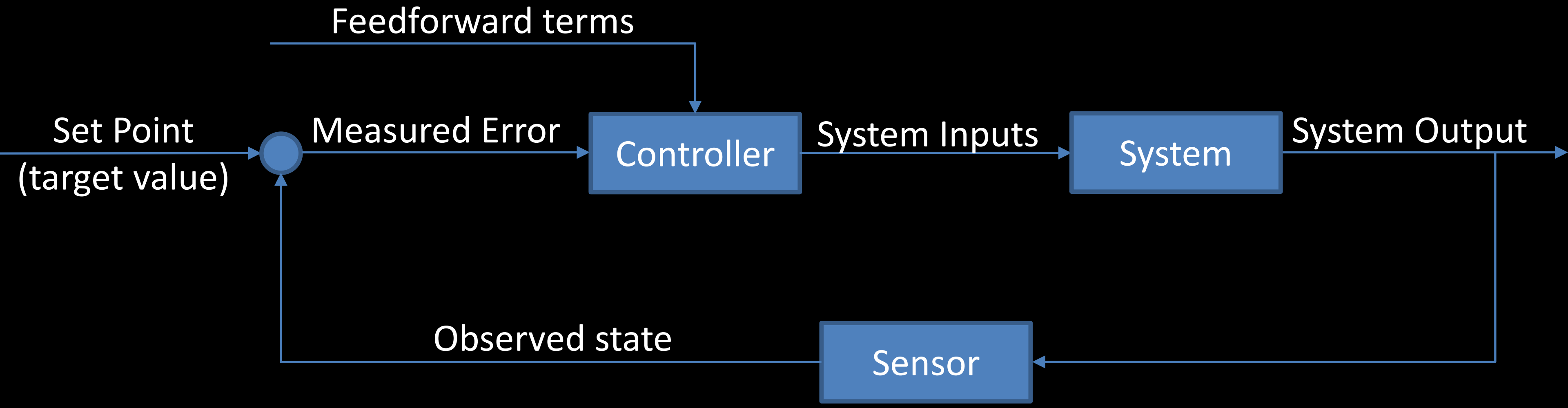
Open loop control

Different states need different actions

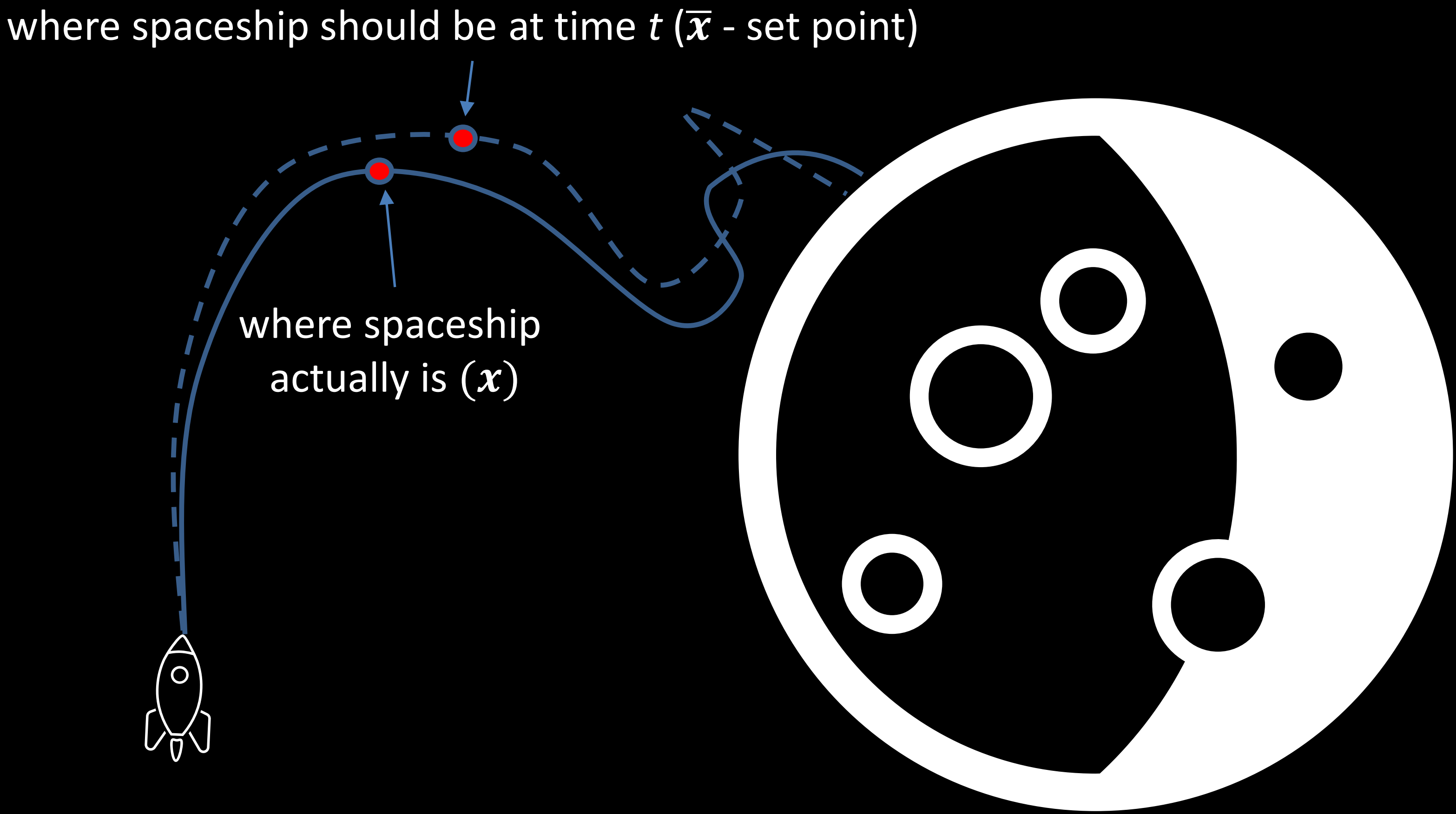


Control inputs computed through TO are *state dependent!*

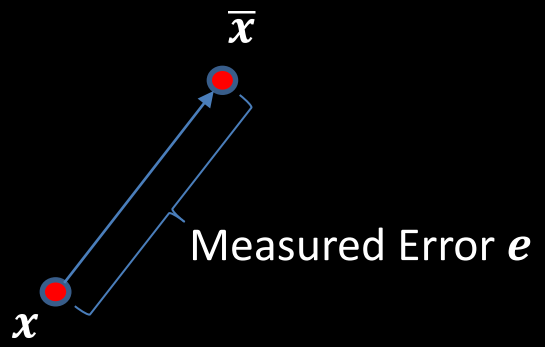
Feedback (closed loop) control



Different states need different actions



Simple strategies for feedback control

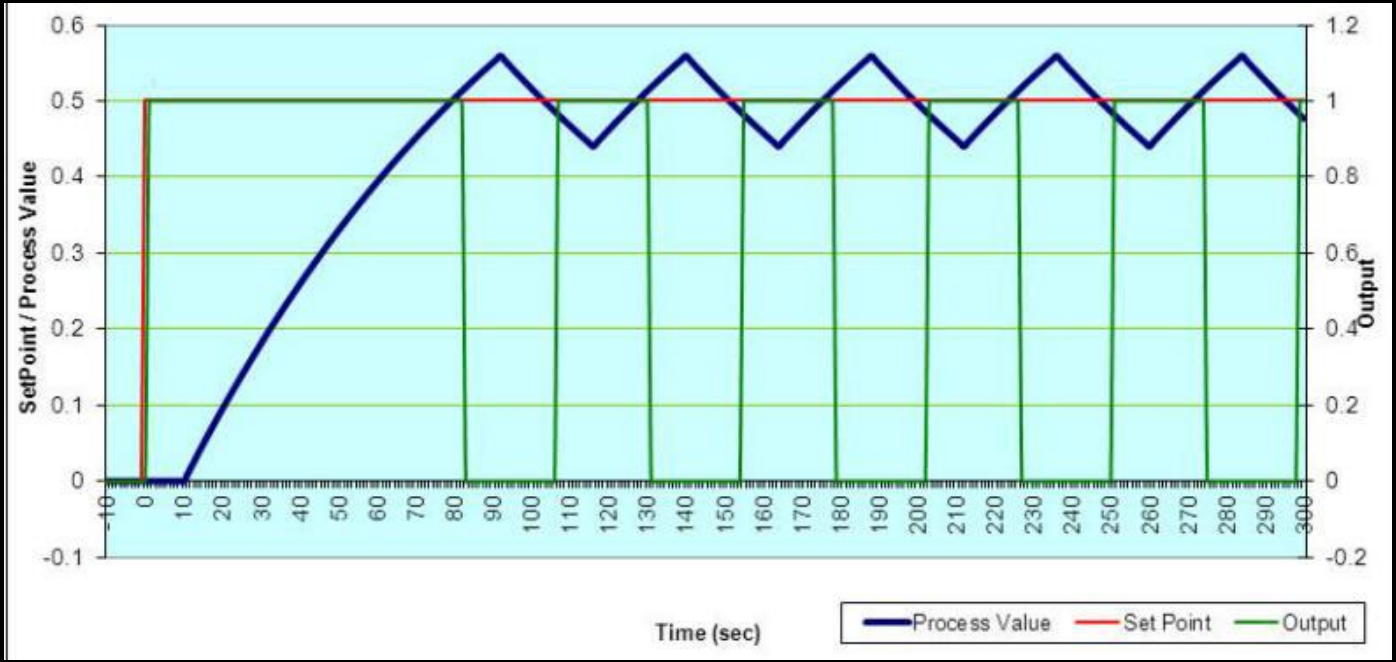


Feedback controller aims to eliminate $\boldsymbol{e}$.

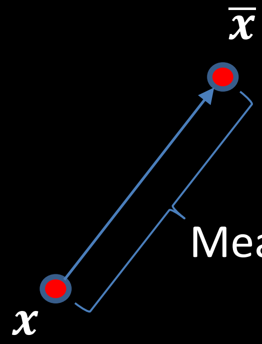
First attempt for our spaceship example:

$$\boldsymbol{F_f} = F_{MAX} \frac{\boldsymbol{e}}{|\boldsymbol{e}|}$$

Bang-bang control



Simple strategies for feedback control



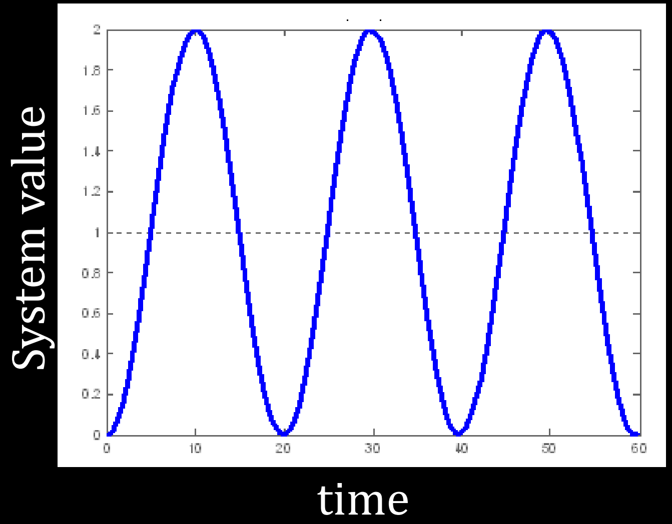
Feedback controller aims to eliminate $\boldsymbol{e}$.

Second attempt:

$$\boldsymbol{F}_{\boldsymbol{f}} = k_p \boldsymbol{e}$$

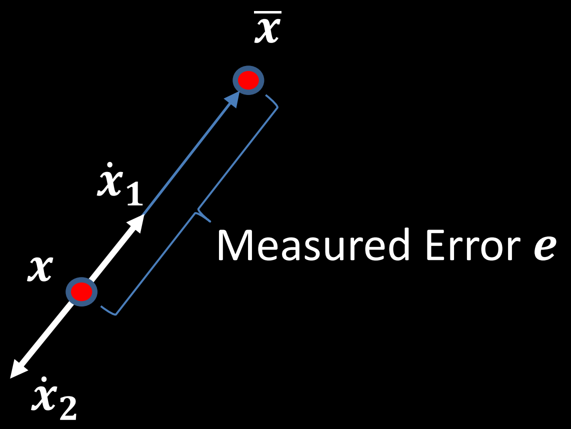
Proportional *gain*

Proportional Control



Note: changing k_p only affects the frequency of undamped oscillation, not the amplitude – think of it as an ideal spring!

Simple strategies for feedback control



Feedback controller aims to eliminate $\boldsymbol{e}$.

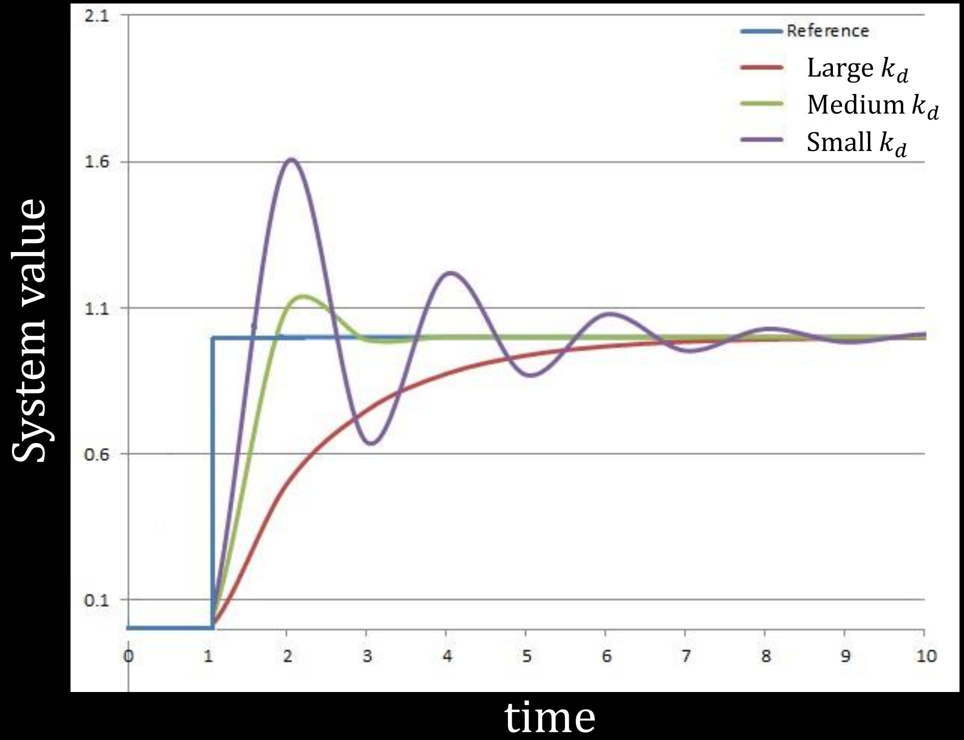
Should $\boldsymbol{F}_f$ be the same if the spaceship had velocity $\dot{\boldsymbol{x}}_1$ as if it has velocity $\dot{\boldsymbol{x}}_2$?

Third attempt:

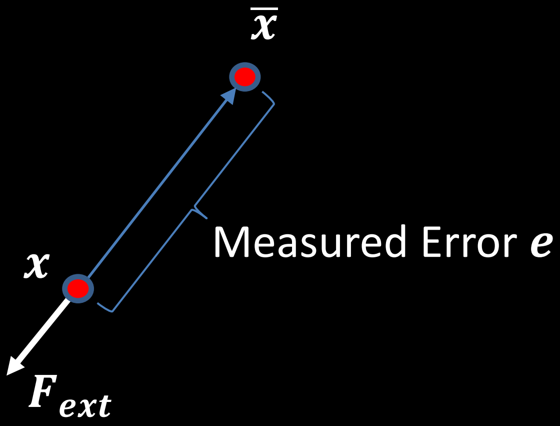
$$\boldsymbol{F}_f = k_p\boldsymbol{e} + k_d\dot{\boldsymbol{e}}$$

Derivative gain

Proportional-derivative (PD) Control



Simple strategies for feedback control



Feedback controller aims to eliminate $\boldsymbol{e}$.

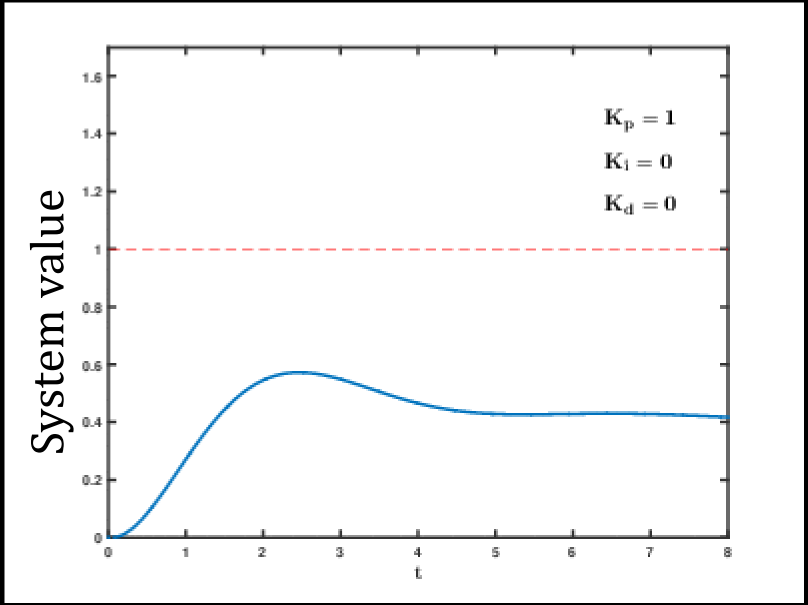
Assume there is a constant, unknown external force acting on the system. This leads to a *steady state error.*

Last attempt:

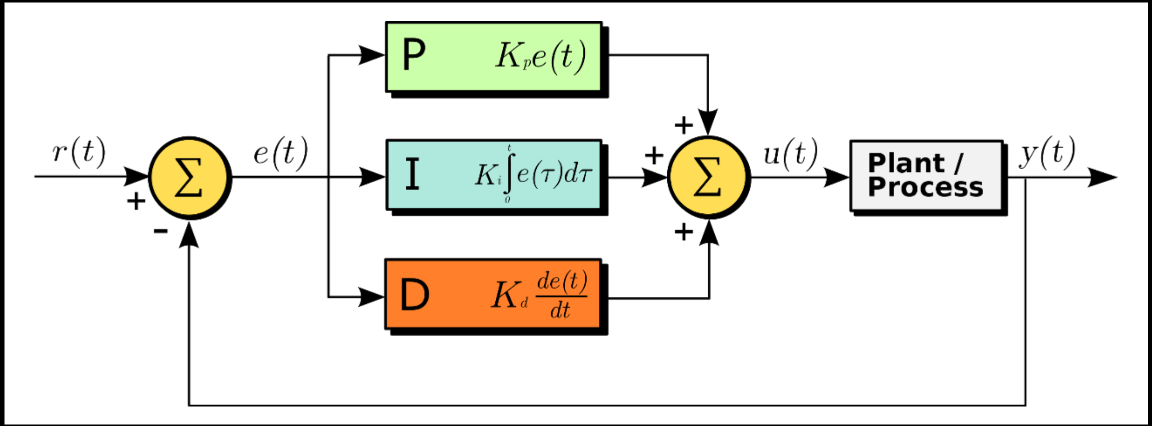
$$\boldsymbol{F}_{\boldsymbol{f}} = k_p \boldsymbol{e} + k_i \int_0^t \boldsymbol{e}(\tau) d\tau + k_d \dot{\boldsymbol{e}}$$

Integral gain

Proportional-integral-derivative (PID) Control

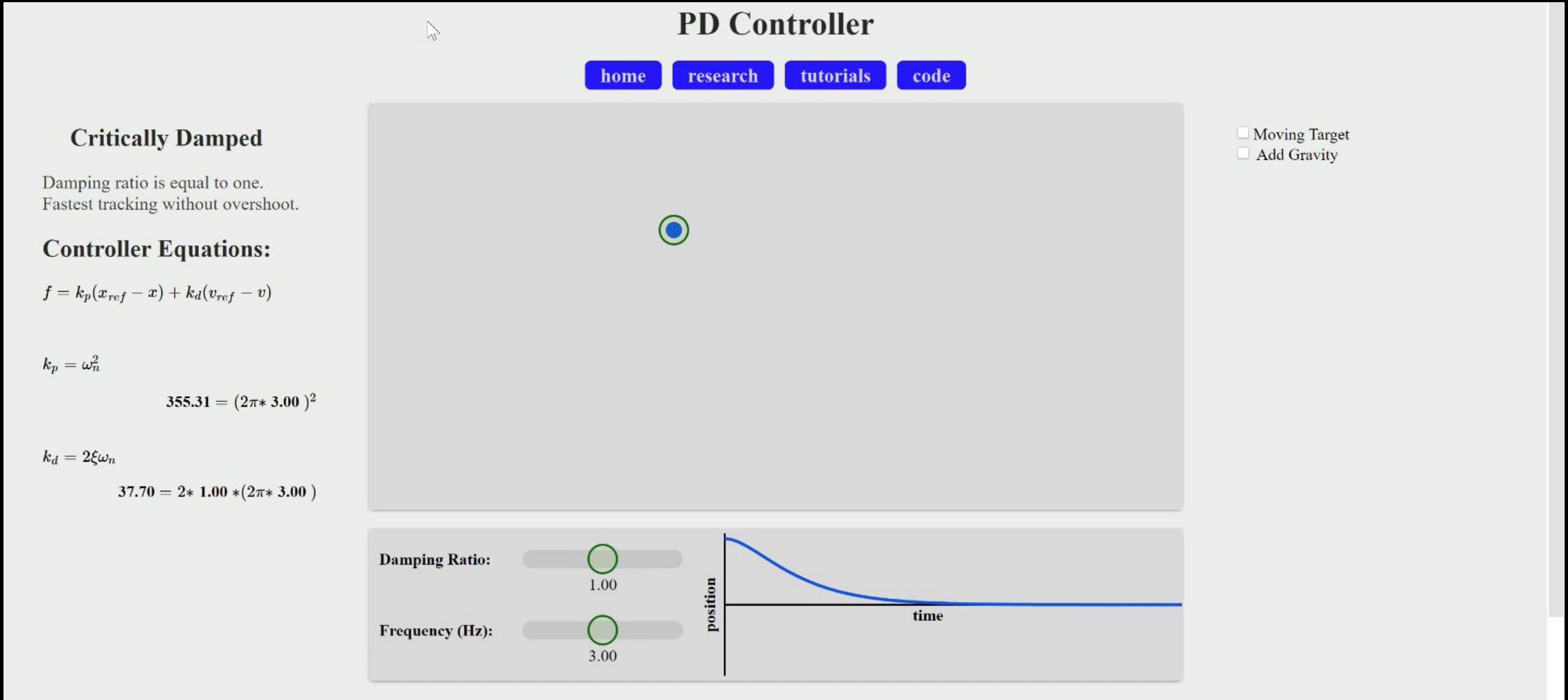


Proportional-integral-derivative (PID) Control



Note 1: The integral term is often neglected, as it can cause unstable/unsafe responses.
Note 2: Various heuristics (Ziegler–Nichols, Åström–Hägglund method) exist for setting PID gains – generally based on observations of the system response.
Note 3: Simple systems are well-understood, so gains are easy to set according to analytic models: e.g. for critically damped (no oscillations) behavior, $k_d = 2\sqrt{k_p}$

PD Control in action



http://www.matthewpeterkelly.com/tutorials/pdControl/index.html

What is it that a PD controller should output?

- What we've seen so far: (thruster) force
  - For complex systems, it is difficult to obtain desired behavior (e.g. critically damped response)
  - PD gains and overall behavior are dependent on system properties
    - a heavier spaceship needs different PD gains than a light one. If additional cargo is loaded on, or as fuel burns out, PD gains need to change. Not very convenient.
- An alternative: PD controller outputs target accelerations, use ***model-based*** methods (e.g. based on inverse dynamics) to generate forces:

$$\bar{\boldsymbol{a}} = k_p \boldsymbol{e} + k_i \int_0^t \boldsymbol{e}(\tau) d\tau + k_d \dot{\boldsymbol{e}} \quad \left\{ \begin{array}{l} \boldsymbol{F_f} = m\bar{\boldsymbol{a}} \\ \qquad\qquad \text{or even better} \\ \min\limits_{\boldsymbol{a},\boldsymbol{F_f}} \frac{\mathbf{1}}{\mathbf{2}} (\boldsymbol{a} - \bar{\boldsymbol{a}})^{\boldsymbol{T}} (\boldsymbol{a} - \bar{\boldsymbol{a}}) + \boldsymbol{O}(\boldsymbol{a}, \boldsymbol{F_f}) \\ \textbf{subject to } \boldsymbol{F_f} = m\boldsymbol{a}, \mathrm{C}(\boldsymbol{a}, \boldsymbol{F_f}) \geq \mathbf{0}, \end{array} \right.$$

We'll come back to this idea soon!

Another note on PD controllers

- PD controllers look a lot like *virtual* springs

$$\overline{\boldsymbol{a}} = k_p \boldsymbol{e} + k_d \dot{\boldsymbol{e}} = k_p(\overline{\boldsymbol{x}} - \boldsymbol{x_t}) + k_d(\dot{\overline{\boldsymbol{x}}} - \dot{\boldsymbol{x}}_{\boldsymbol{t}})$$

- Need small time steps (e.g. evaluate and apply new control signals with high frequency) for stability
- Can also formulate PD controllers implicitly:

$$\overline{\boldsymbol{a}} = k_p(\overline{\boldsymbol{x}} - \boldsymbol{x_{t+1}}) + k_d(\dot{\overline{\boldsymbol{x}}} - \dot{\boldsymbol{x}}_{\boldsymbol{t+1}})$$

Implicit PD controller

$$\overline{\boldsymbol{a}} = k_p(\overline{\boldsymbol{x}} - \boldsymbol{x}_{t+1}) + k_d(\dot{\overline{\boldsymbol{x}}} - \dot{\boldsymbol{x}}_{t+1})$$

Let's work it out (noting that $\dot{\boldsymbol{x}}_{t+1} = \dot{\boldsymbol{x}}_t + h\overline{\boldsymbol{a}}$; $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + h\dot{\boldsymbol{x}}_{t+1}$)

$$\dot{x}_{t+1} = \dot{x}_t + h\bar{a}$$

$$x_{t+1} = x_t + h\dot{x}_t + h^2\bar{a}$$

$$\bar{a} = -k_p\,(x_{t+1} - \bar{x}) - k_d(\dot{x}_{t+1} - \dot{\bar{x}})$$

$$= -k_p(x_t + h\dot{x}_t + h^2\bar{a} - \bar{x}) - k_d(\dot{x}_t + h\bar{a} - \dot{\bar{x}})$$

$$= -k_p\,(x_t + h\dot{x}_t - \bar{x}) + h^2\bar{a}\cdot k_p - k_d\,(\dot{x}_t - \dot{\bar{x}}) + h\bar{a}\cdot k_d$$

$$\therefore\ \bar{a} + h^2\bar{a}\cdot k_p + h\,\bar{a}\,k_d = -k_p\,(x_t + h\dot{x}_t - \bar{x}) - k_d(\dot{x}_t - \dot{\bar{x}})$$

$$\therefore\ \bar{a} = \frac{-k_p\,(x_t + h\dot{x}_t - \bar{x}) - k_d(\dot{x}_t - \dot{\bar{x}})}{1 + h^2 k_p + h\,k_d}$$

Please do double check the math!

Implicit PD controller

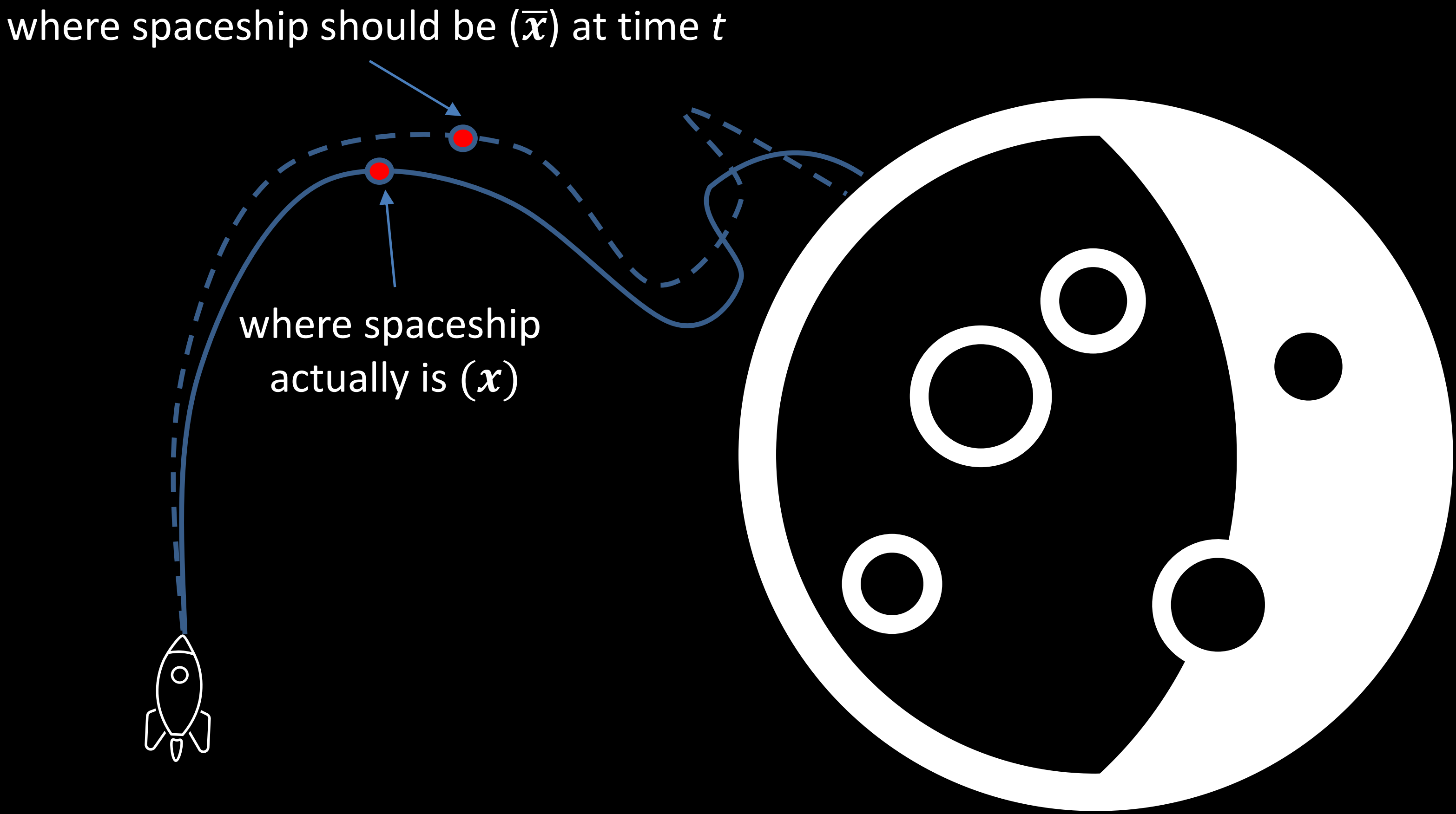
$$\bar{\boldsymbol{a}} = k_p(\bar{\boldsymbol{x}} - \boldsymbol{x}_{t+1}) + k_d(\dot{\bar{\boldsymbol{x}}} - \dot{\boldsymbol{x}}_{t+1})$$

Let's work it out (noting that $\dot{\boldsymbol{x}}_{t+1} = \dot{\boldsymbol{x}}_t + h\bar{\boldsymbol{a}}$; $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + h\dot{\boldsymbol{x}}_{t+1}$)

$$\bar{\boldsymbol{a}} = \frac{k_p(\bar{\boldsymbol{x}} - \boldsymbol{x}_t) - hk_p\dot{\boldsymbol{x}}_t + k_d\left(\dot{\bar{\boldsymbol{x}}} - \dot{\boldsymbol{x}}_t\right)}{1 + h^2k_p + hk_d}$$

Give it a try yourself!

We've seen some of the basics of feedback control



Locomotion

Dictionary

locomotion

locomotion

/ləʊkəˈməʊʃ(ə)n/

noun

movement or the ability to move from one place to another.
"the muscles that are concerned with locomotion"

Similar: movement | motion | moving | shifting | stirring | action | travel

Translations, word origin, and more definitions

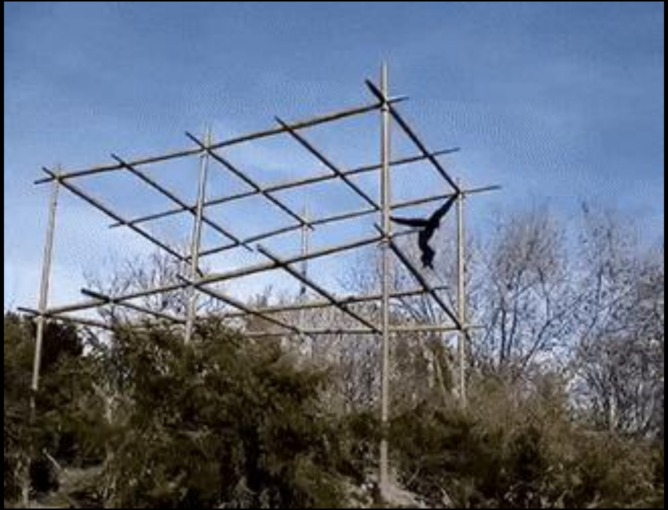
From Oxford | *Feedback*


DMM.com
STARBUCKS







luckyHercul



gifak-net

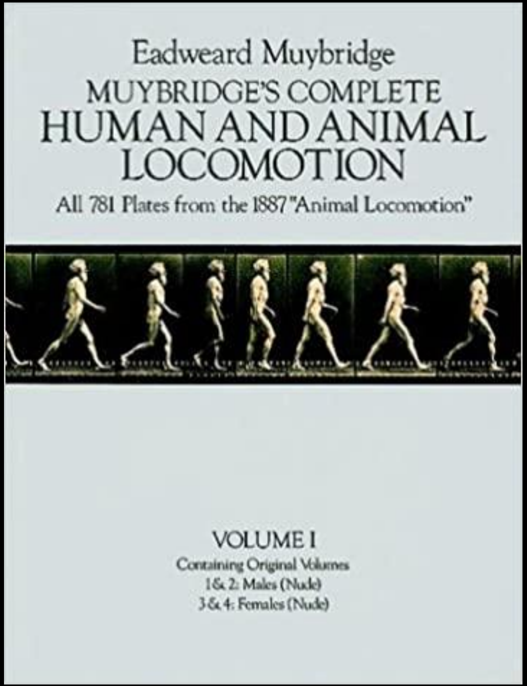

gifak-net

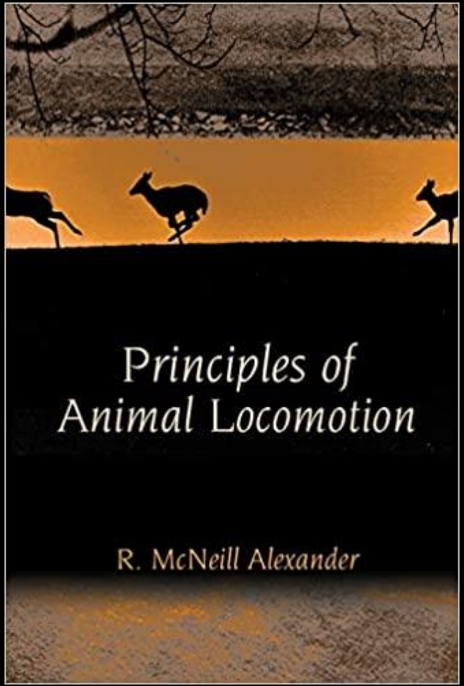

4GIFs.com

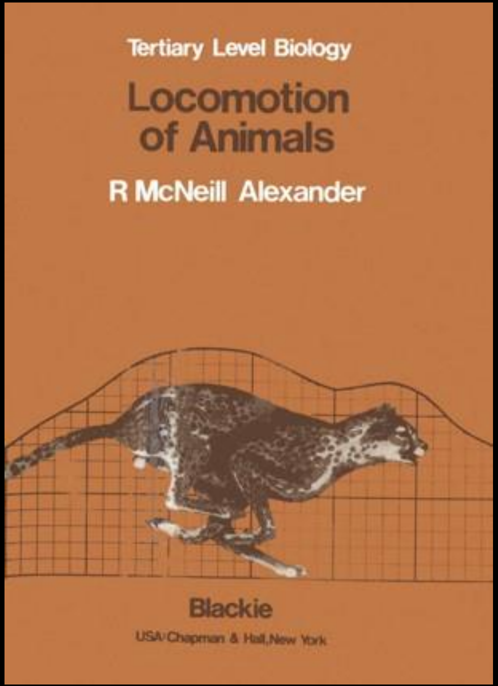

GIFAK-NET

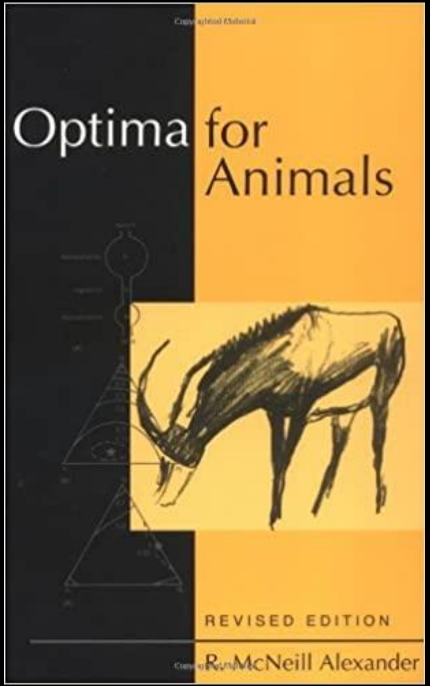

MakeAGIF.com

Locomotion: biomechanical foundations







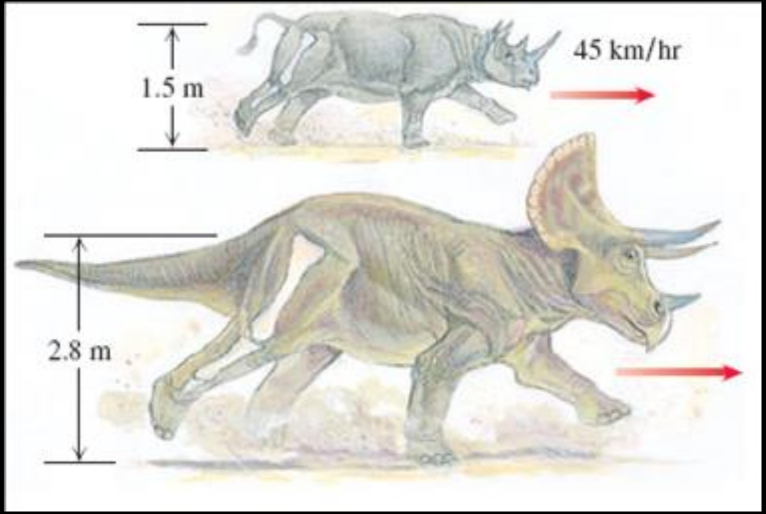


Locomotion: biomechanical foundations

Eadweard Muybridge, *"Sallie Gardner"* (1878)

Biomechanical studies shed light on the principles of animal locomotion

- **Lots** of very important lessons to be learned



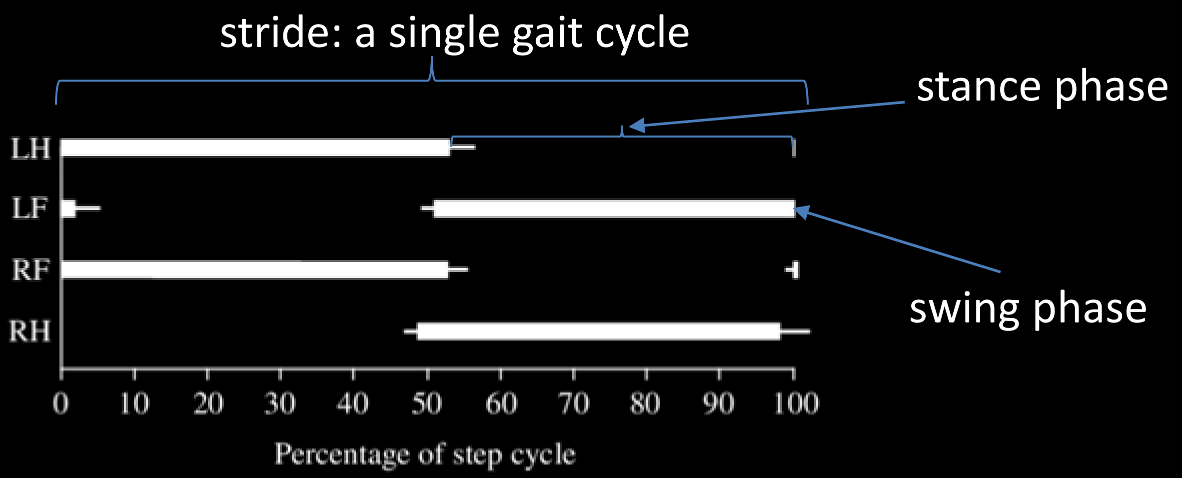
See, for example, *scaling laws*

Biomechanical studies shed light on the principles of animal locomotion

- **Gait**: the pattern of movement of the limbs of animals (including humans) during locomotion

walk

Hildebrand Gait diagrams

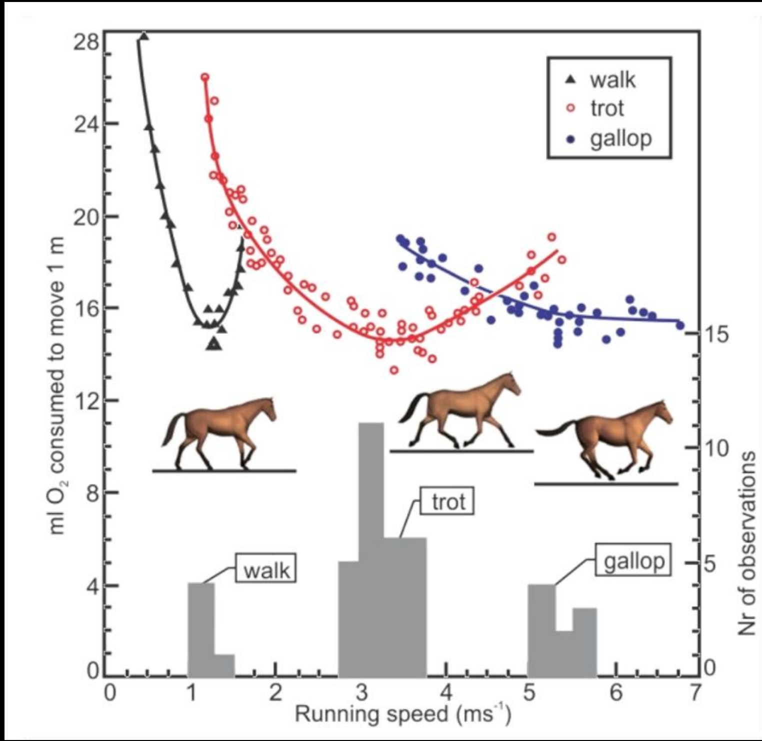


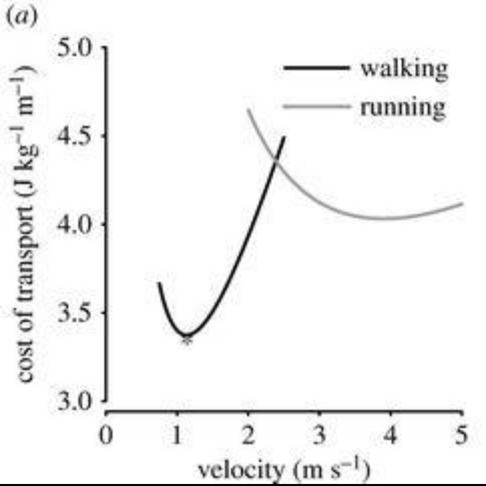
Duty cycle: percentage of a stride a limb spends in stance phase. Walking gaits have duty cycles > 50%. Running gaits have duty cycles < 50%.

Why so many types of gaits?

- Crawl, Walk, Trot, Running Trot, Pace, Bound, Gallop, etc
- First and foremost, speed and energetics
  - each gait has a particular speed at which the minimum calories per meter are consumed

Why so many types of gaits?





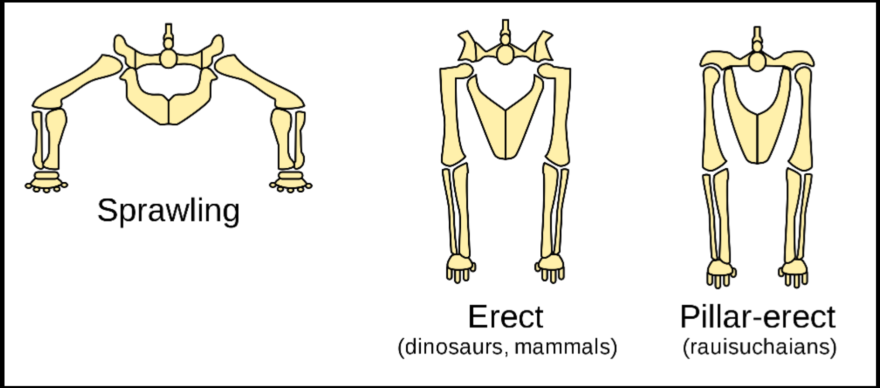
Why so many types of gaits?

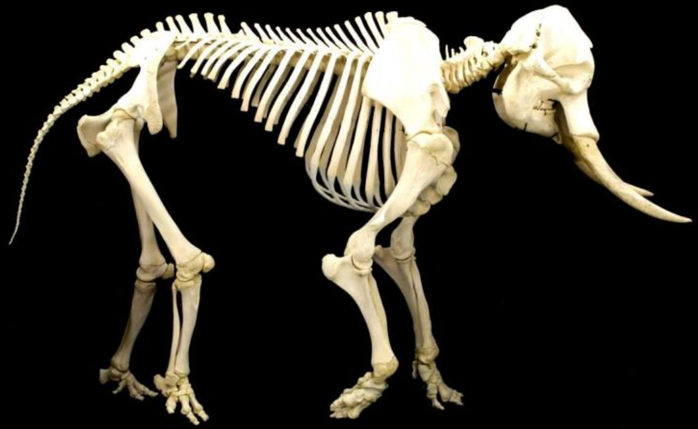
- Crawl, Walk, Trot, Running Trot, Pace, Bound, Gallop, etc
- First and foremost, speed and energetics
  - each gait has a particular speed at which the minimum calories per meter are consumed
- But other reasons as well
  - stability/robustness (e.g. dogs with long legs & short backs pace to prevent feet from stepping on each other)
  - comfort/accommodating injuries (some gaits need more spine movement than others)
  - Showing off?

Why so many types of gaits?

Much, much more to learn from the field of biomechanics

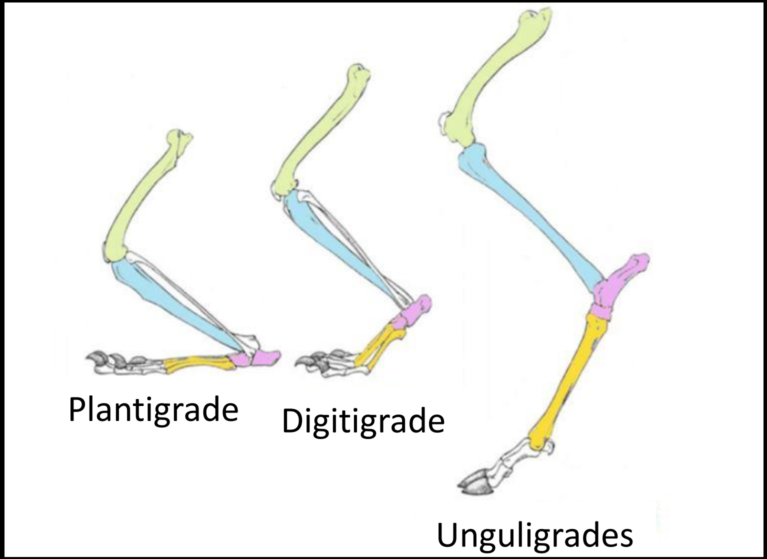
- Classification of postures

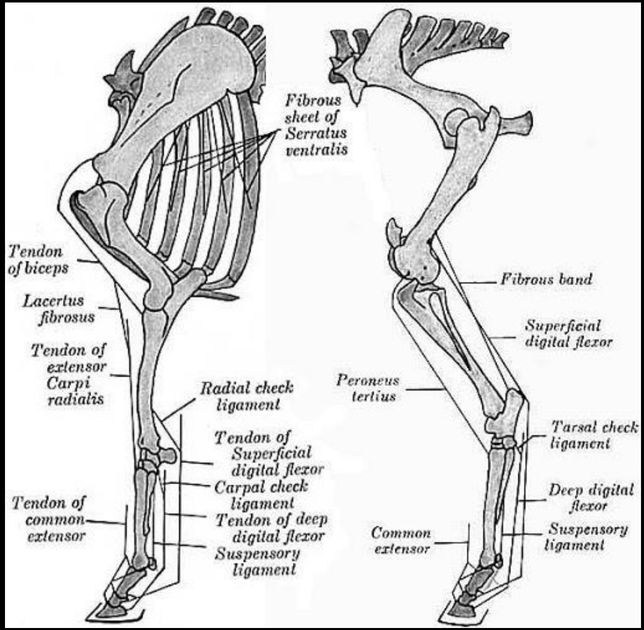




Much, much more to learn from the field of biomechanics

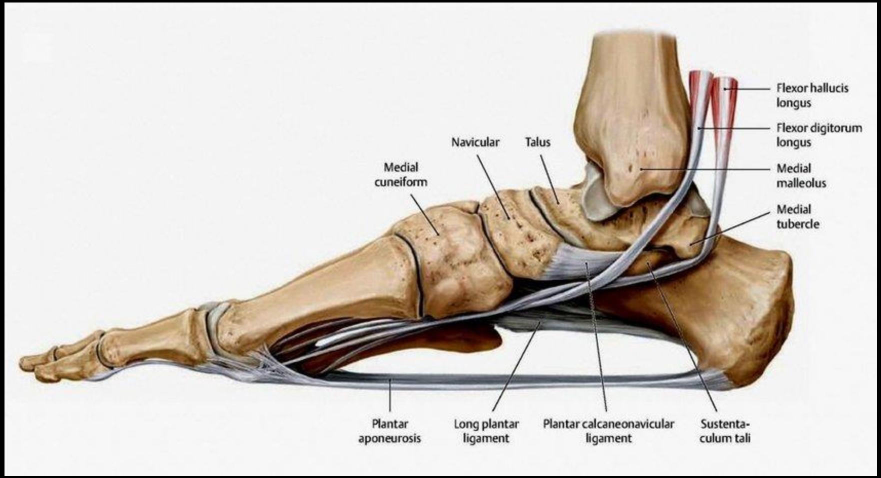
- Classification of postures, limb types and specializations

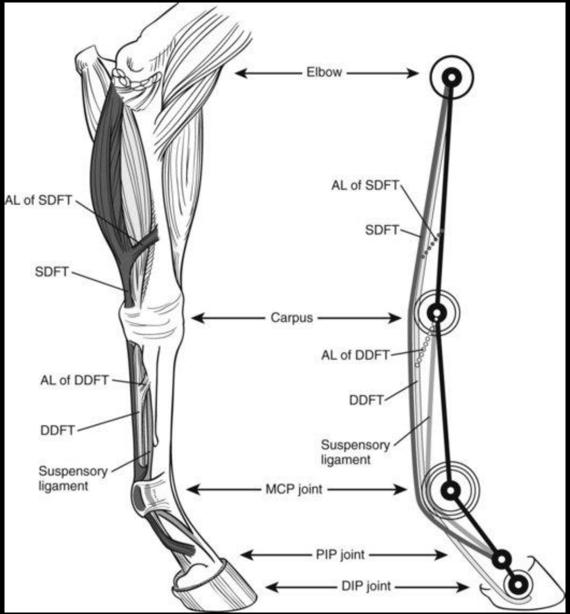




Much, much more to learn from the field of biomechanics

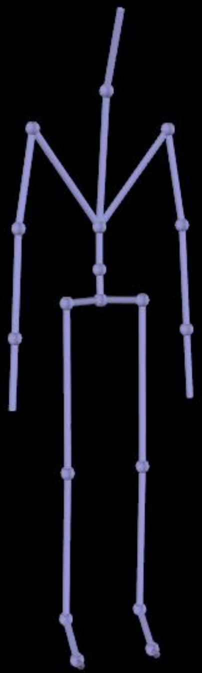
- Classification of postures, limb types and specializations, structural design of skeletons and muscles, the functional role of soft tissues, etc…





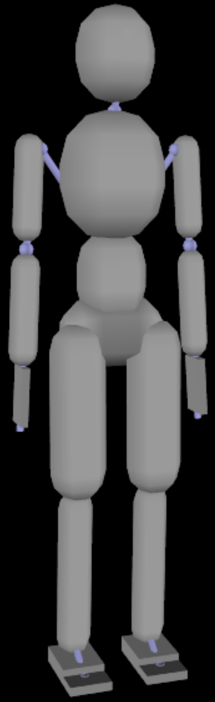
Basics of Locomotion Control

Simulation Model



Joint Hierarchy

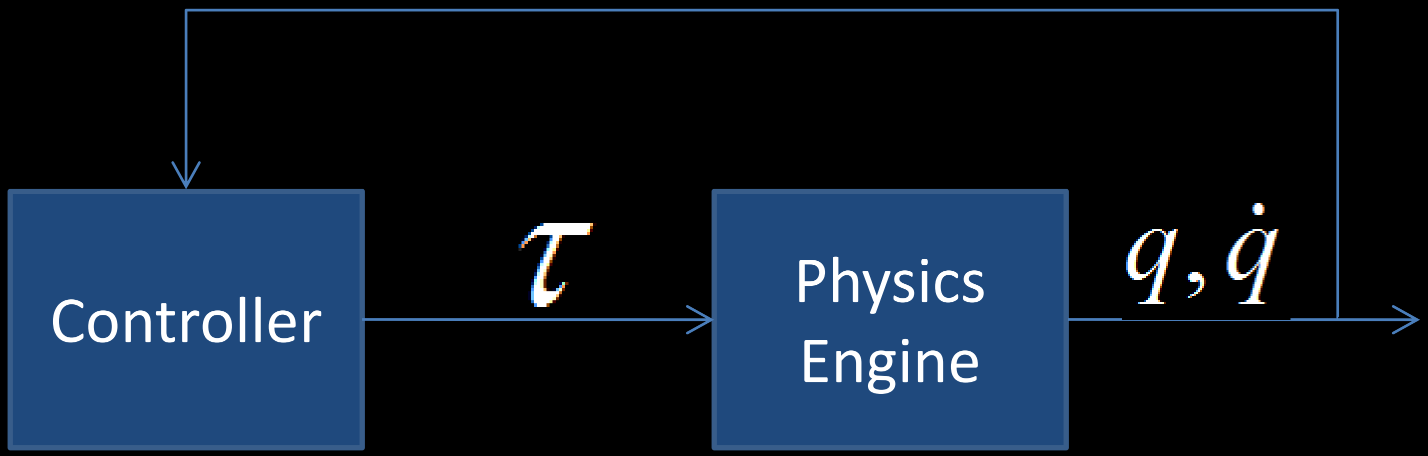
Actuators

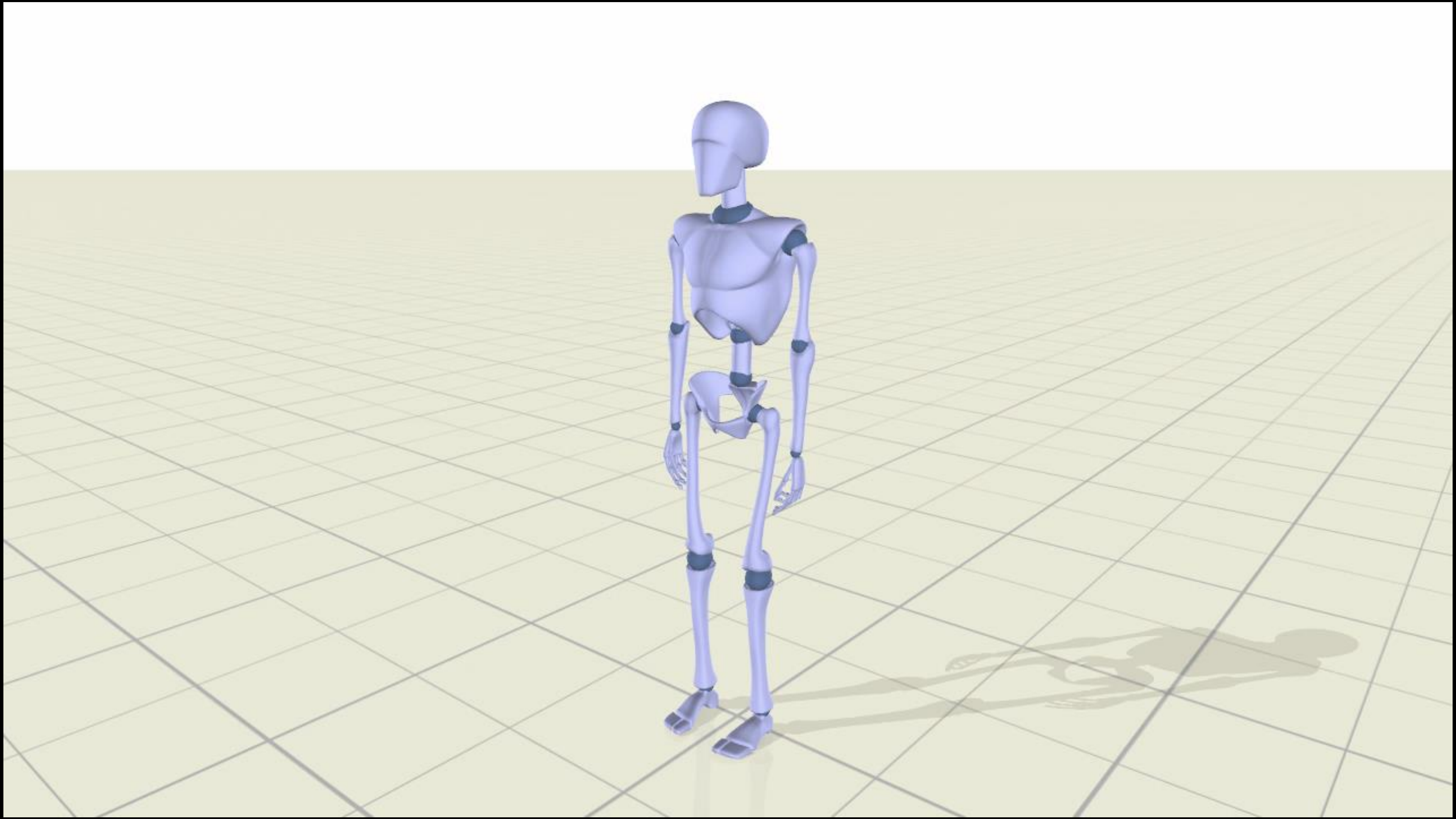


Proxy Geometry

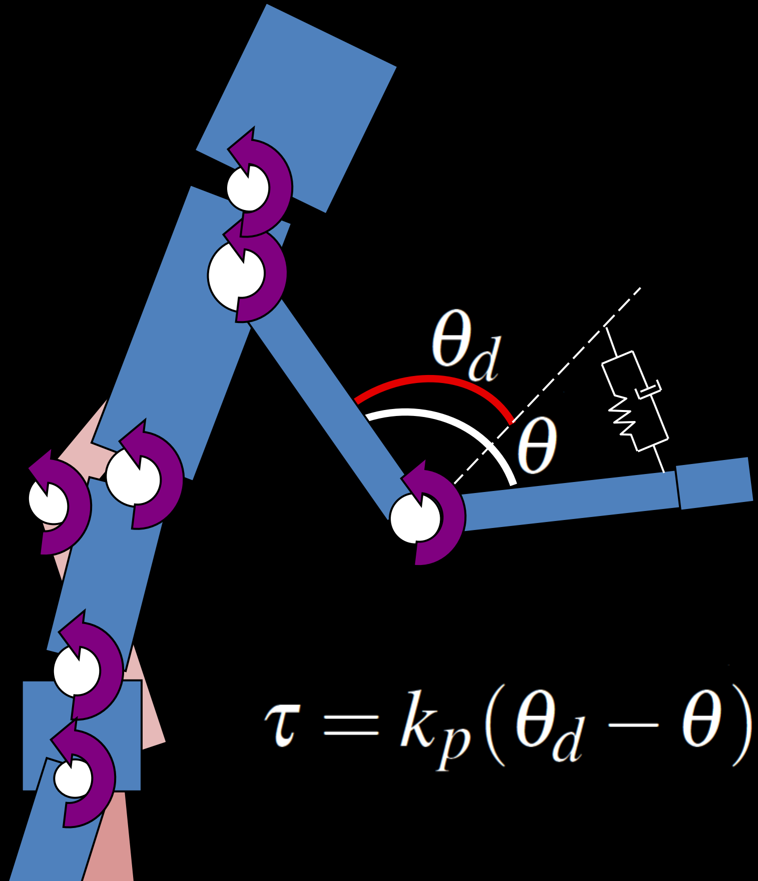
Visualization Mesh

Physics-based Animation

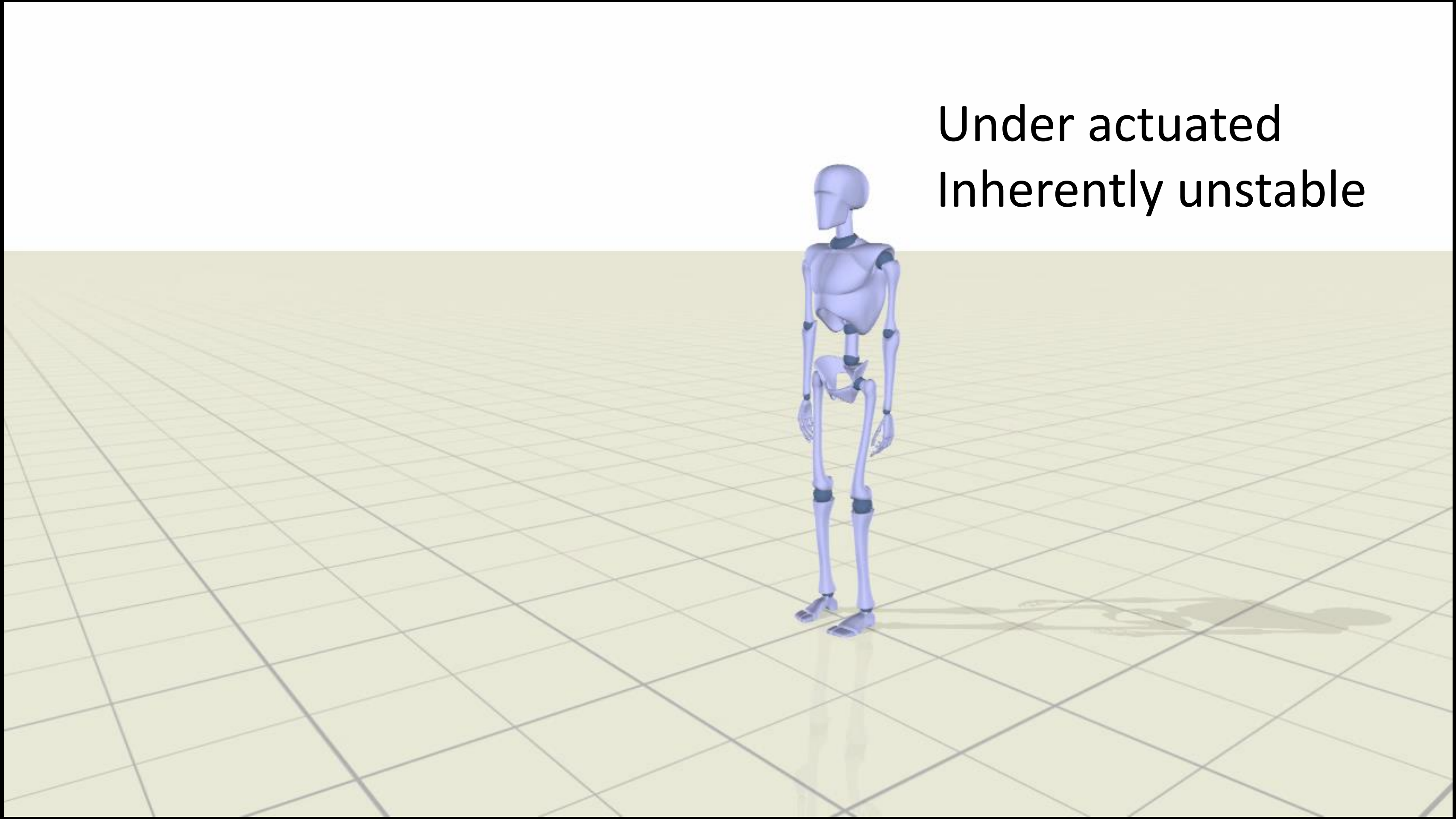




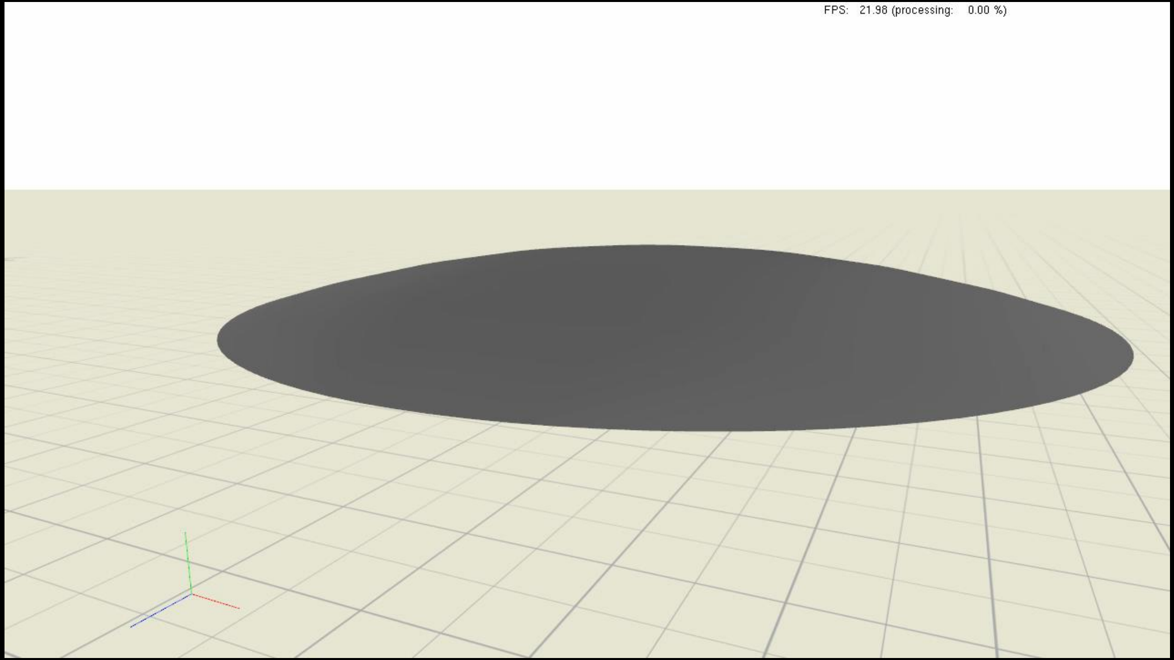
Posture Control – PD control applied to individual joints



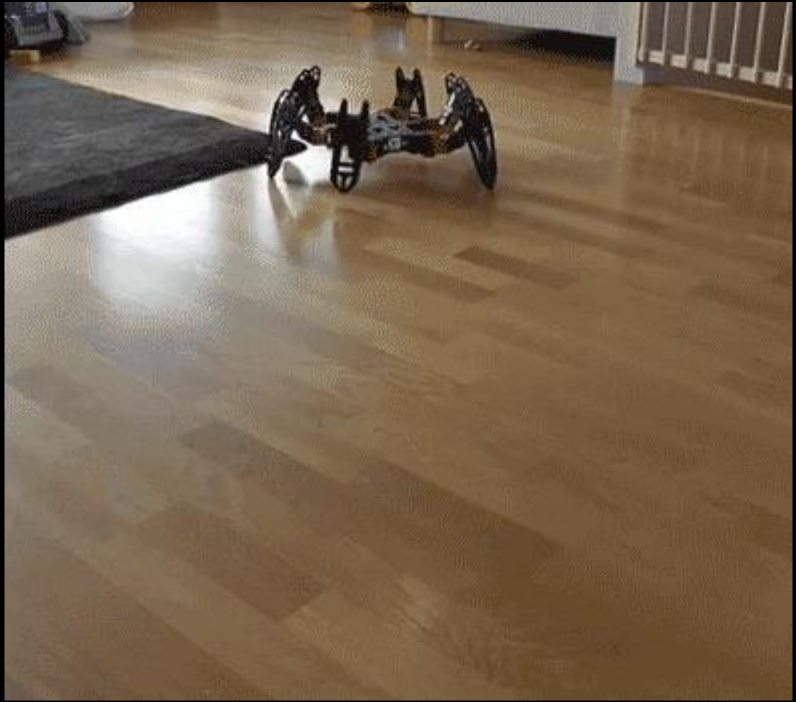
$$\tau = k_p(\theta_d - \theta) - k_d(\dot{\theta} - \dot{\theta}_d)$$


Under actuated
Inherently unstable

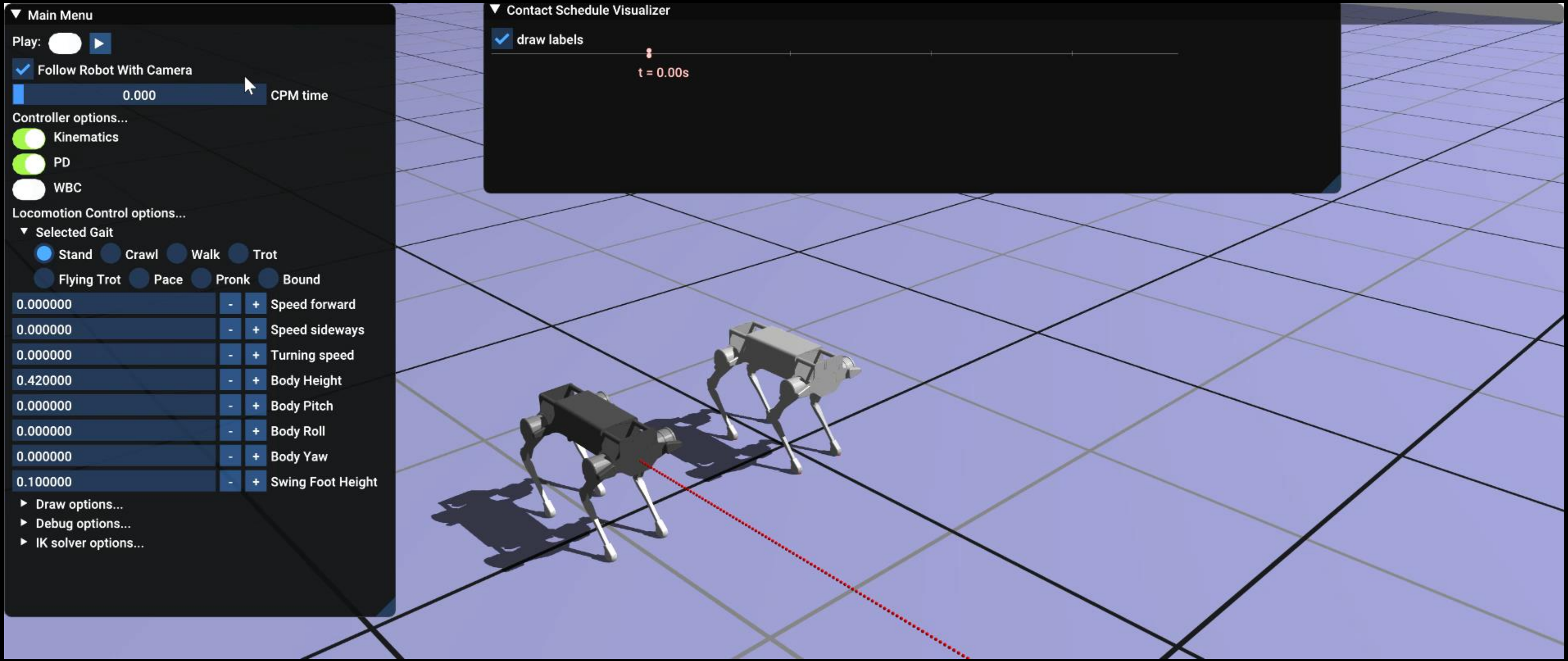
Now, that isn't to say that PD control isn't useful in some settings…



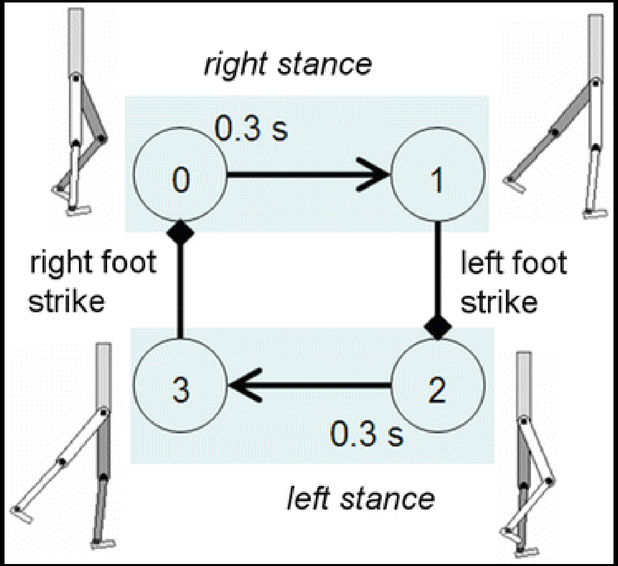
Now, that isn't to say that PD control isn't useful…



Now, that isn't to say that PD control isn't useful in some settings…

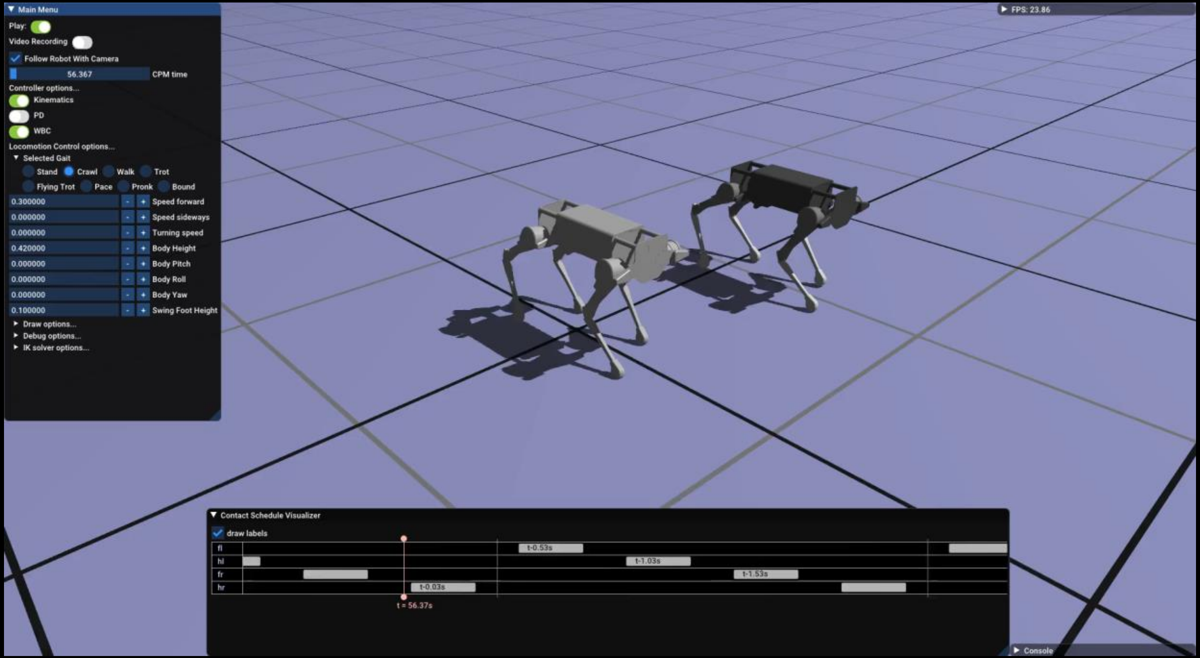


Now, that isn't to say that PD control isn't useful in some settings…

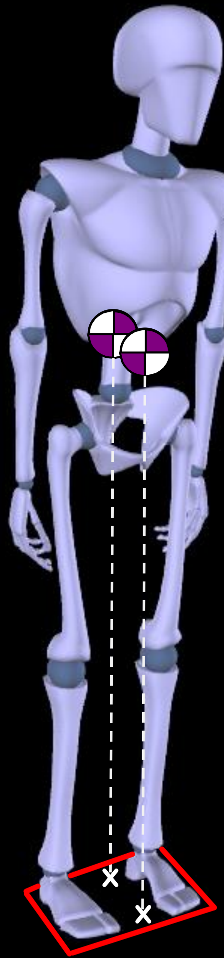


SIMBICON: Simple Biped Locomotion Control, Yin et al., Siggraph 2007

But we can do better than just posture control via PD servos



Whole-Body Control



$$\bar{\boldsymbol{a}}_{\boldsymbol{com}} = k_p \boldsymbol{e} + k_d \dot{\boldsymbol{e}}$$

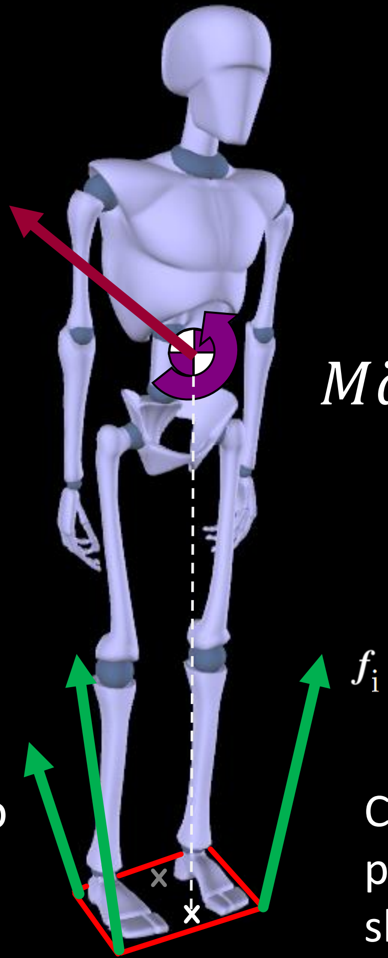
Target linear/angular acceleration for the center of mass

$\ddot{q}$: acceleration in generalized coordinates

f: cartesian-space forces applied at points of contact

u: joint torques (control forces, in generalized coordinates)

K: friction cone (normal component > 0, tangential component subject to Coulomb's law of friction)



$$|\bar{\boldsymbol{a}}_{\boldsymbol{com}} - \boldsymbol{a}_{\boldsymbol{com}}(\ddot{q})|_2^2$$

"keep head upright"

"track end effectors"

$$\min_{\ddot{q},u,f} \{g_1, g_2, \dots, g_n\}$$

subject to

$$M\ddot{q} + C(q,\dot{q}) + J^T f = \begin{bmatrix} 0 \\ u \end{bmatrix}$$

"no control forces on root DOFs"

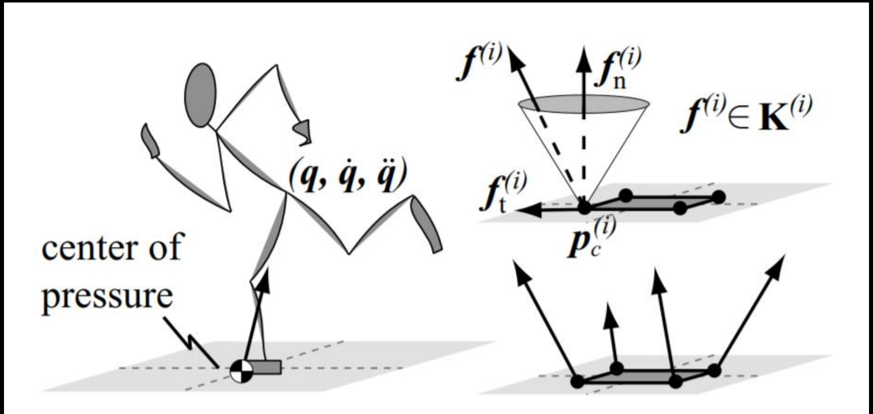
$$f \in K, u \in L$$

$$\underbrace{J\ddot{q} + \dot{J}\dot{q} = 0}$$

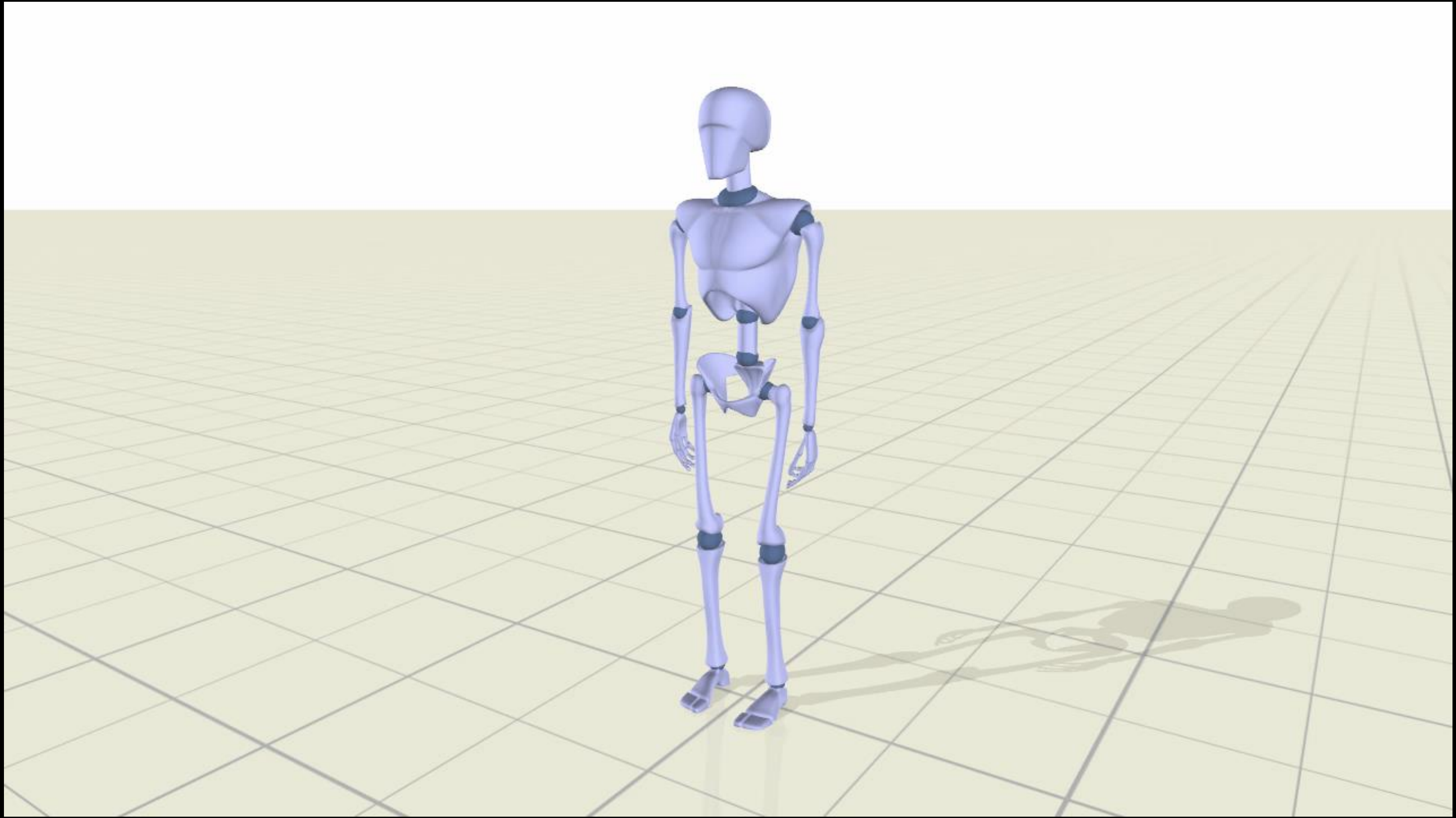
Cartesian-space acceleration of points in contact with the ground should be 0 – no sliding!

Whole Body Control

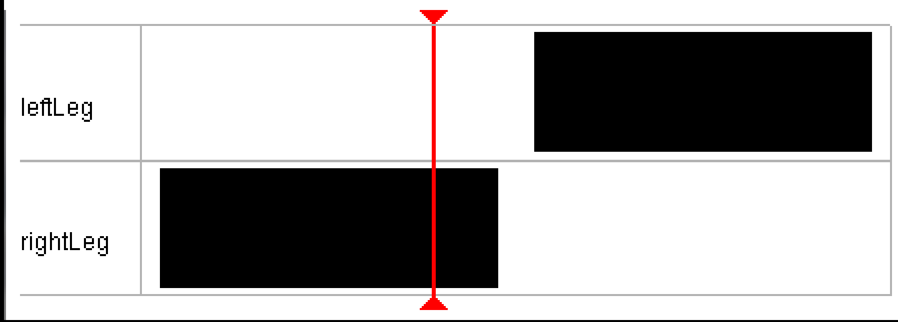
- Also known as Operational Space Control



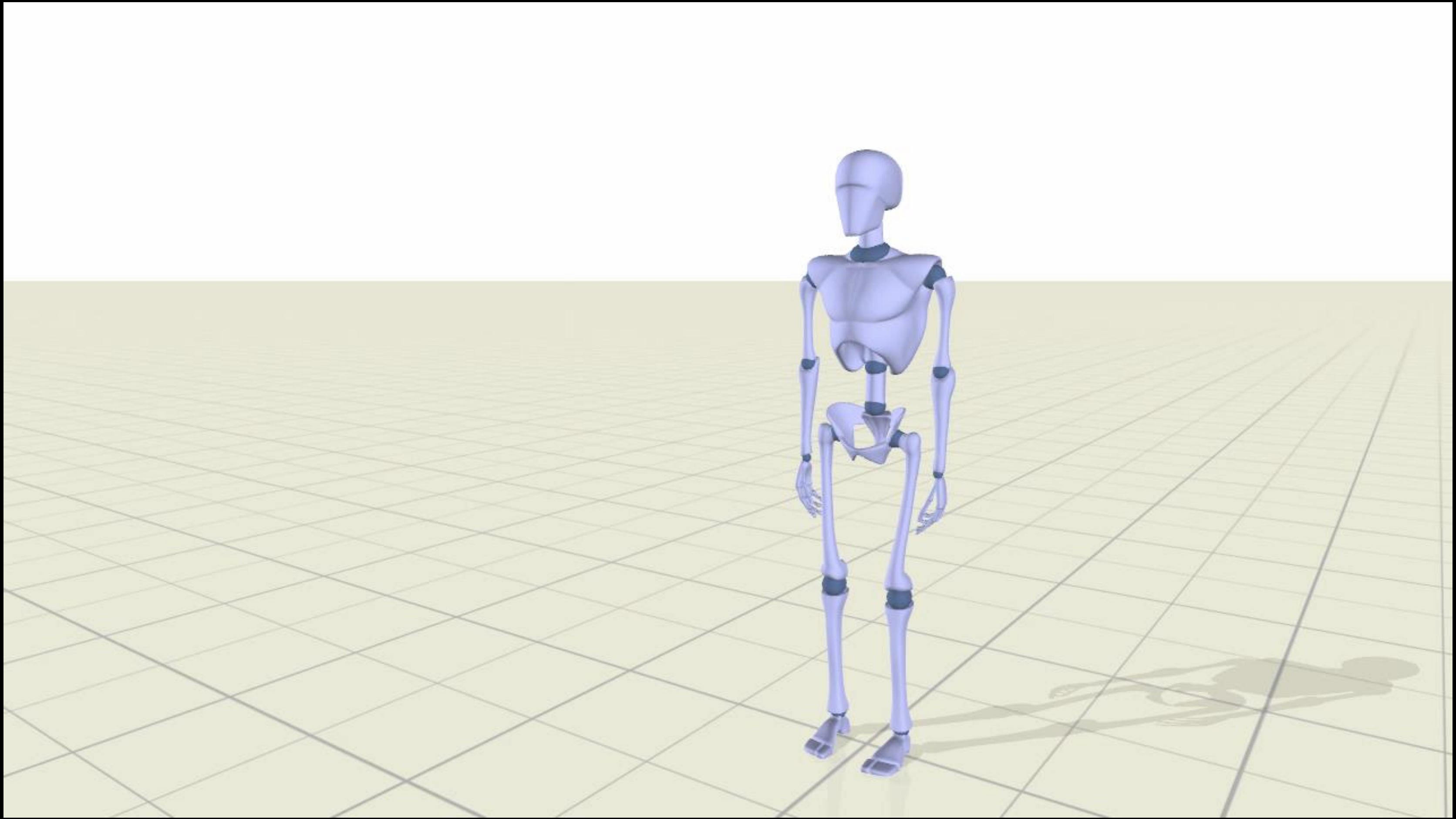
<u>Multiobjective Control with Frictional Contacts</u>, **Yeuhi Abe**, Marco da Silva and Popovic', J. ACM SIGGRAPH / Eurographics Symposium on Computer Animation 2007



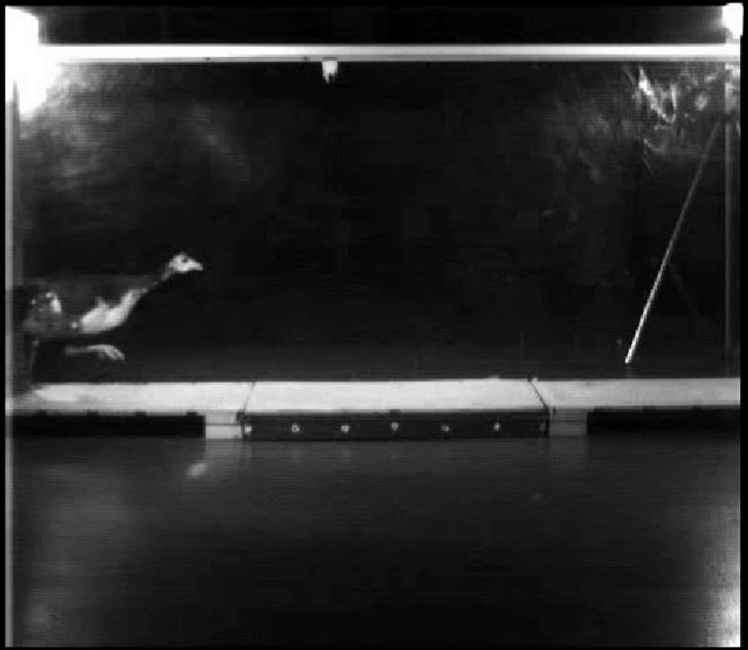
Onwards to walking motions…

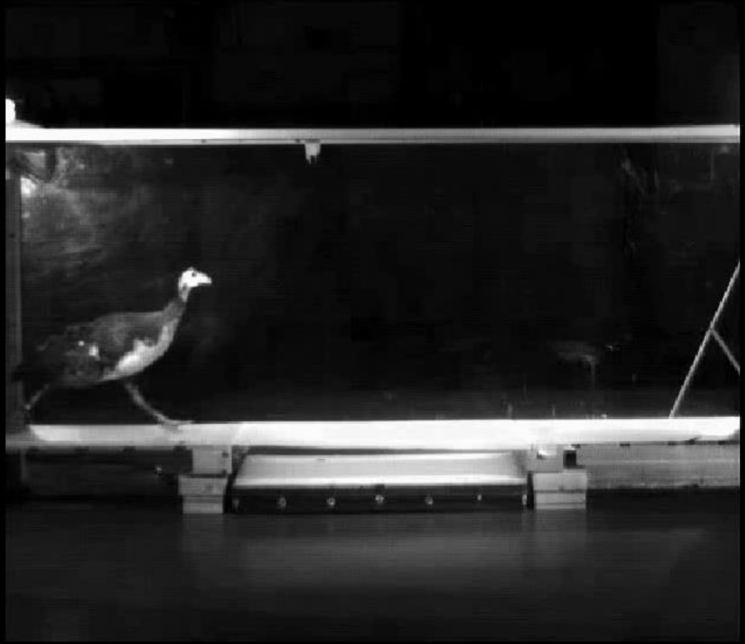


A footfall pattern manages the role of the limbs and therefore the decision variables, objectives and constraints added to the quadratic program that is solved by the locomotion controller.

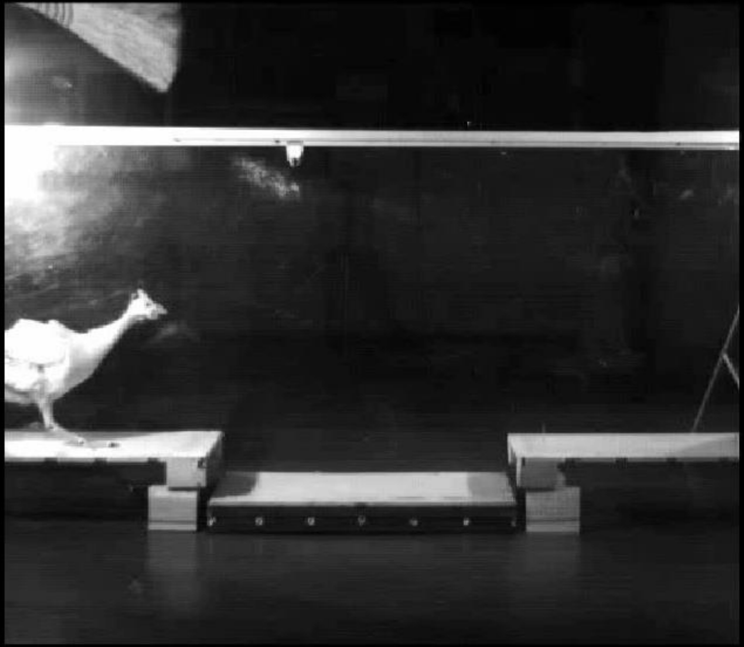


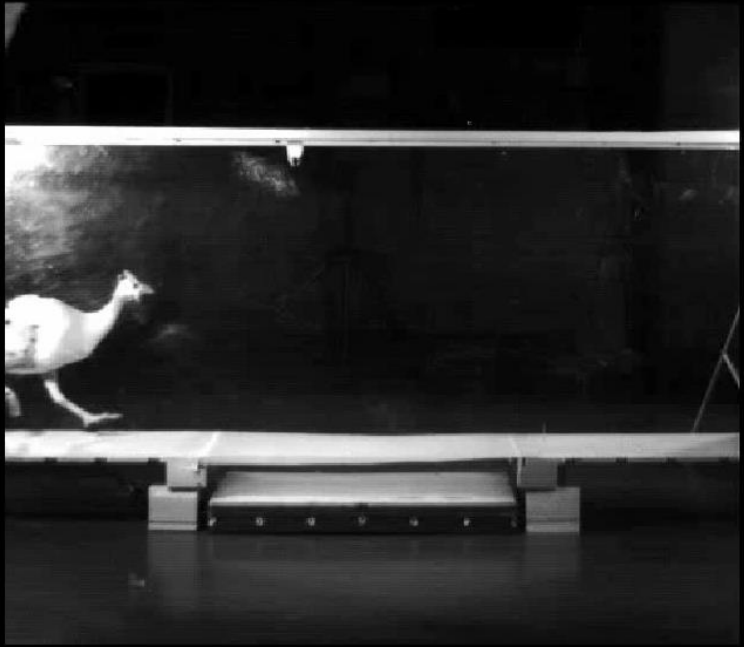
Reacting to unanticipated perturbations



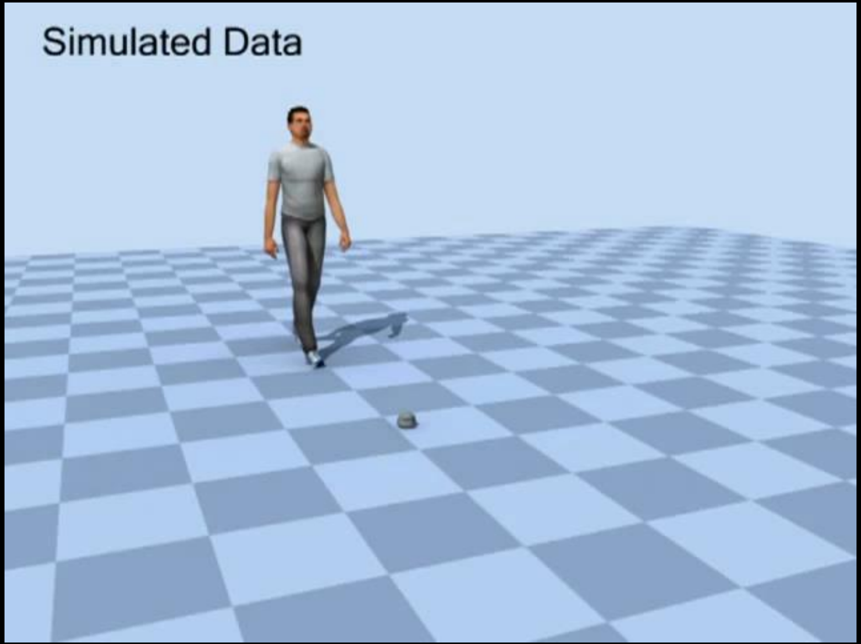


Planning vs reactive behaviors





Reacting to unanticipated perturbations



Simulating Balance Recovery Responses to Trips Based on Biomechanical Principles

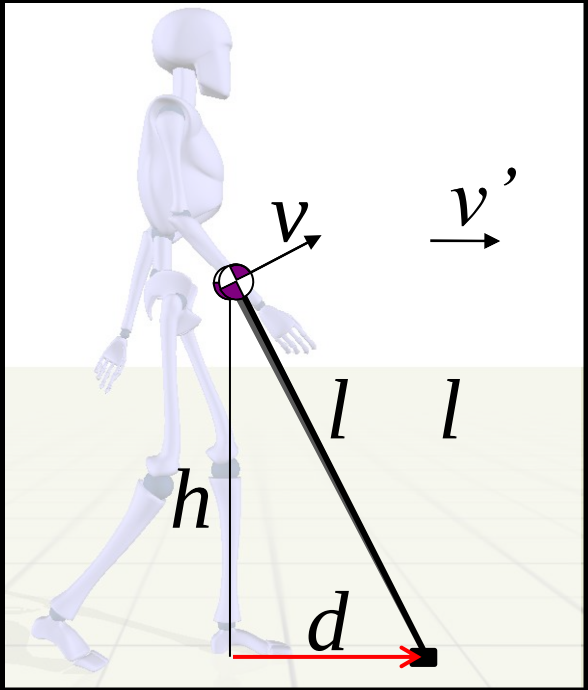
Takaaki Shiratori, Brooke Coley, Rakié Cham, Jessica K. Hodgins

Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation

Reacting to unanticipated perturbations

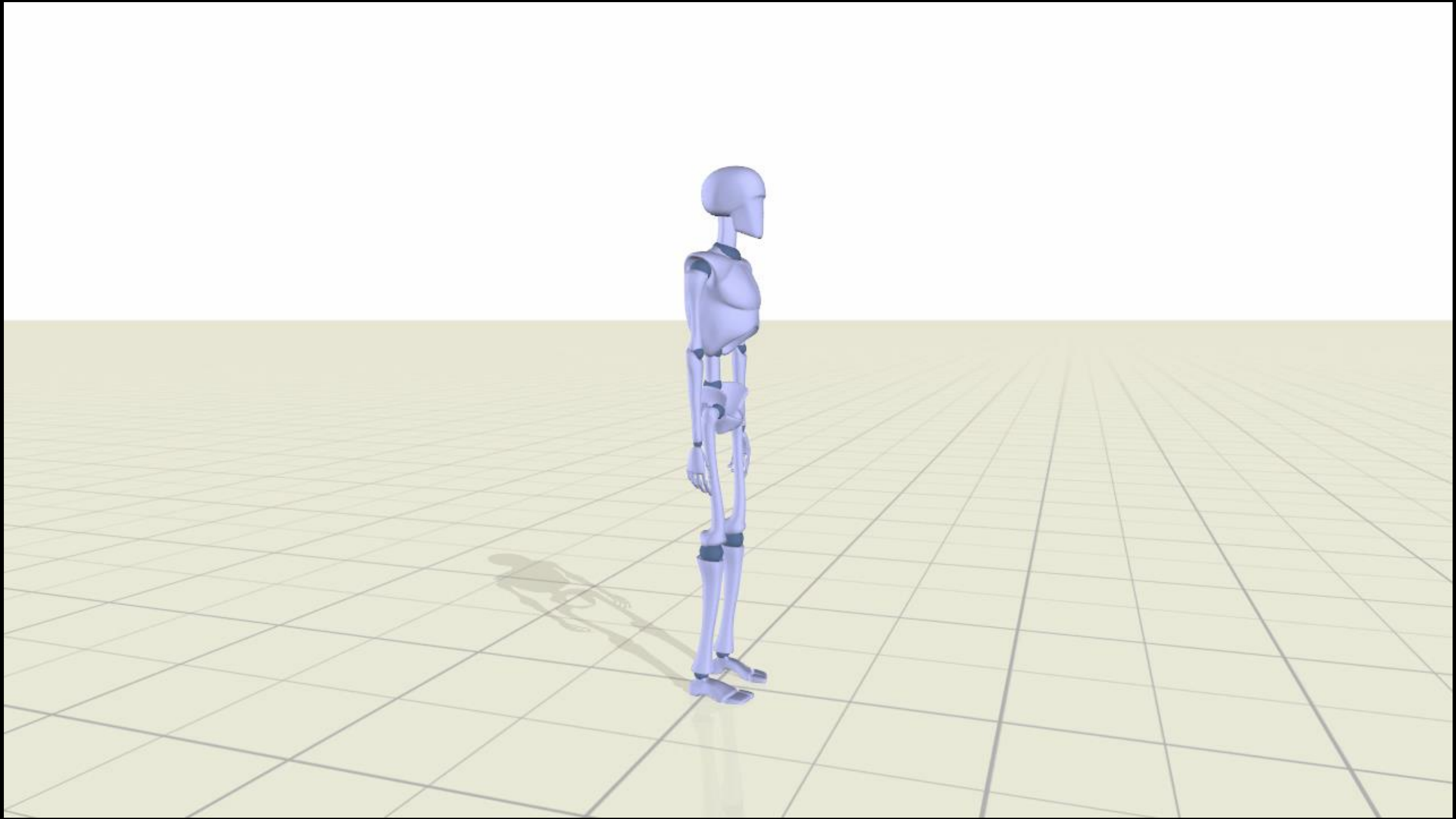
- Ideally we'd solve another Trajectory Optimization problem in real time
  - this strategy is also called Model Predictive Control (MPC), but generally too slow or too limited due to aggressive approximations
- Simple and very effective models for balance recovery (e.g. capture point methods, Raibert controllers) do exist

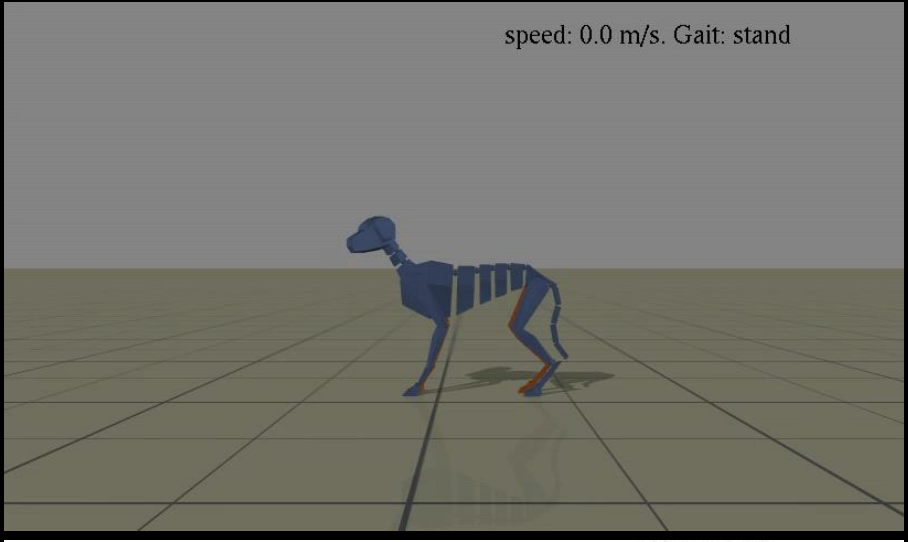
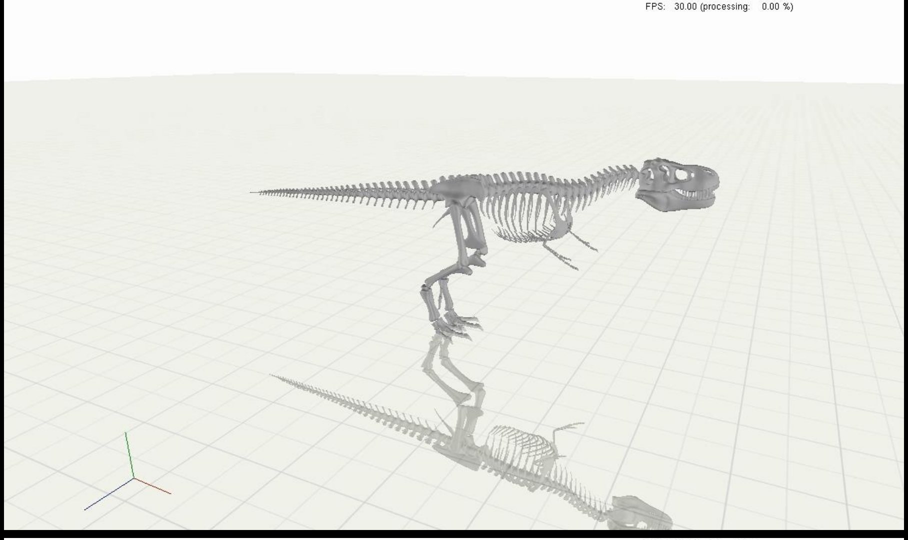
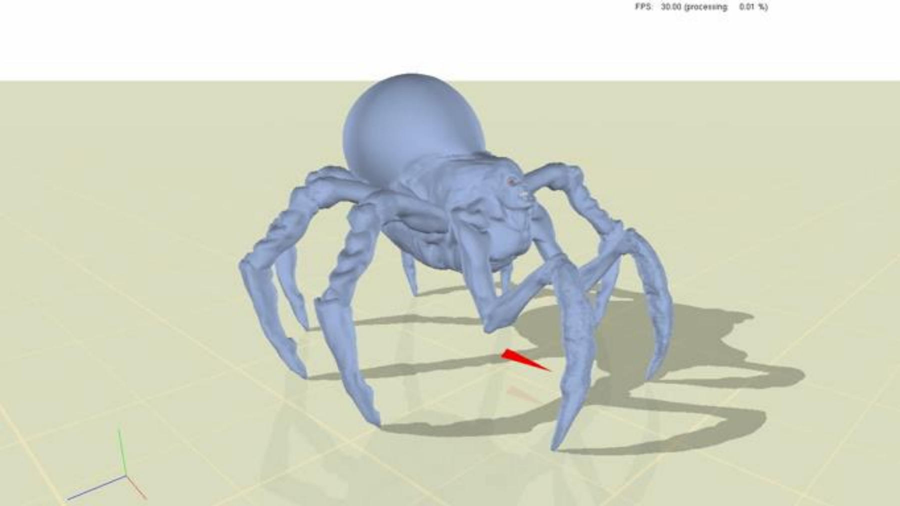
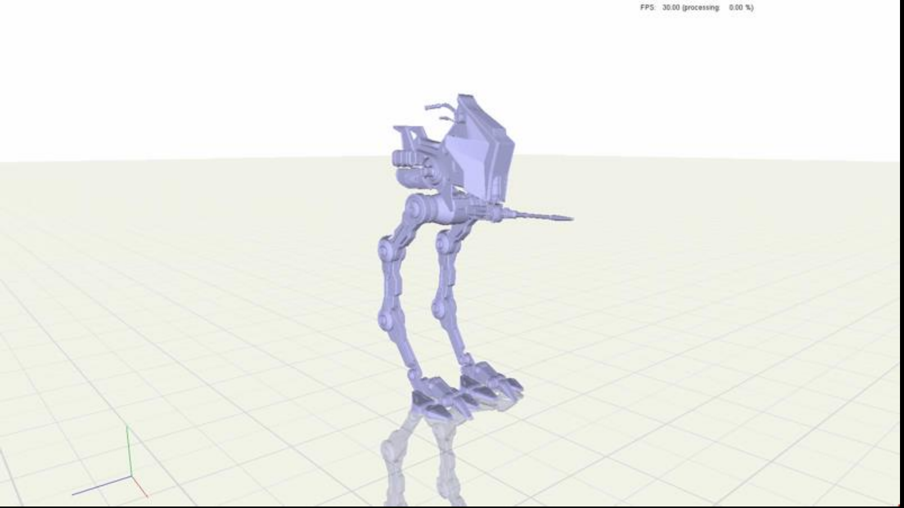
Simple model for foot placement adaptations



$$E = \frac{1}{2}mv^2 + mgh = \frac{1}{2}mv'^2 + mg\,l$$

$$d = d_f(v_d) + (v - v_d)\sqrt{\frac{h}{g}}$$

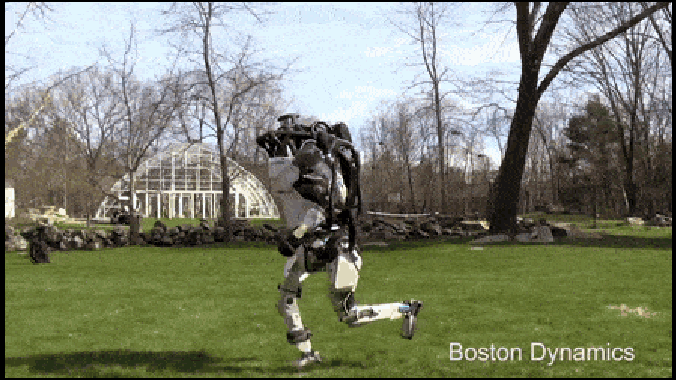


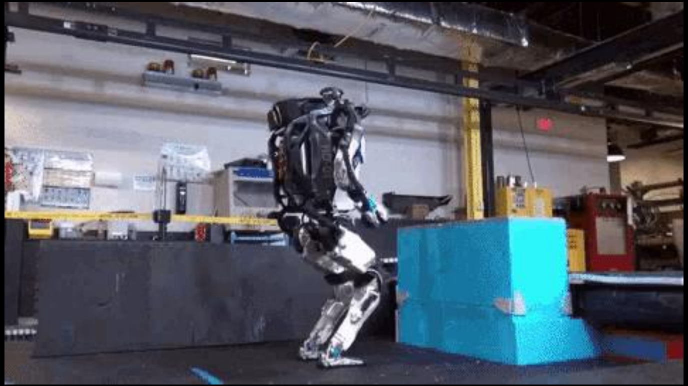

speed: 0.0 m/s. Gait: stand

FPS: 30.00 (processing: 0.00 %)

FPS: 30.00 (processing: 0.01 %)

FPS: 30.00 (processing: 0.00 %)

1m/s walk


clideo.com

Trajectory optimization and whole-body control: the basic techniques used to control the world's most advanced robots





So, what's next?

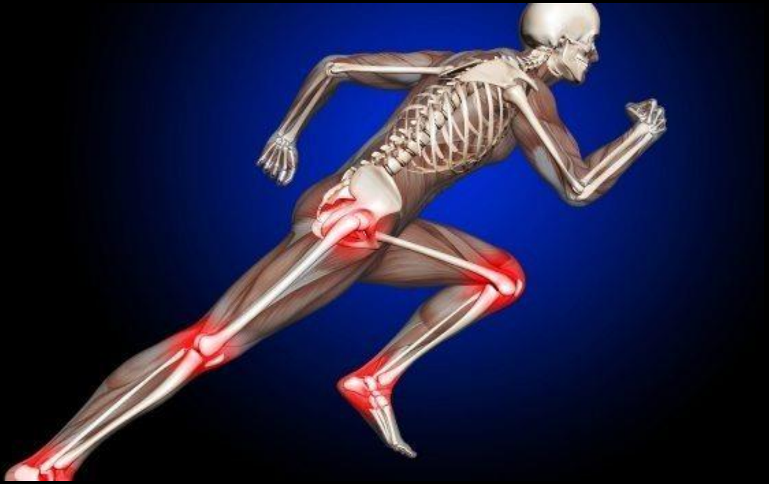
Locomotion in complex environments

Compliance and morphological computation

Boston Dynamics

So, what's next?

Complex maneuvers and rich physical interactions



Increasing the accuracy of our simulation models

That’s all for today