Physics-Based Models for People Tracking: 3D Control and Optimization

Marcus Brubaker[1] Leonid Sigal[1,2] David J Fleet[1]

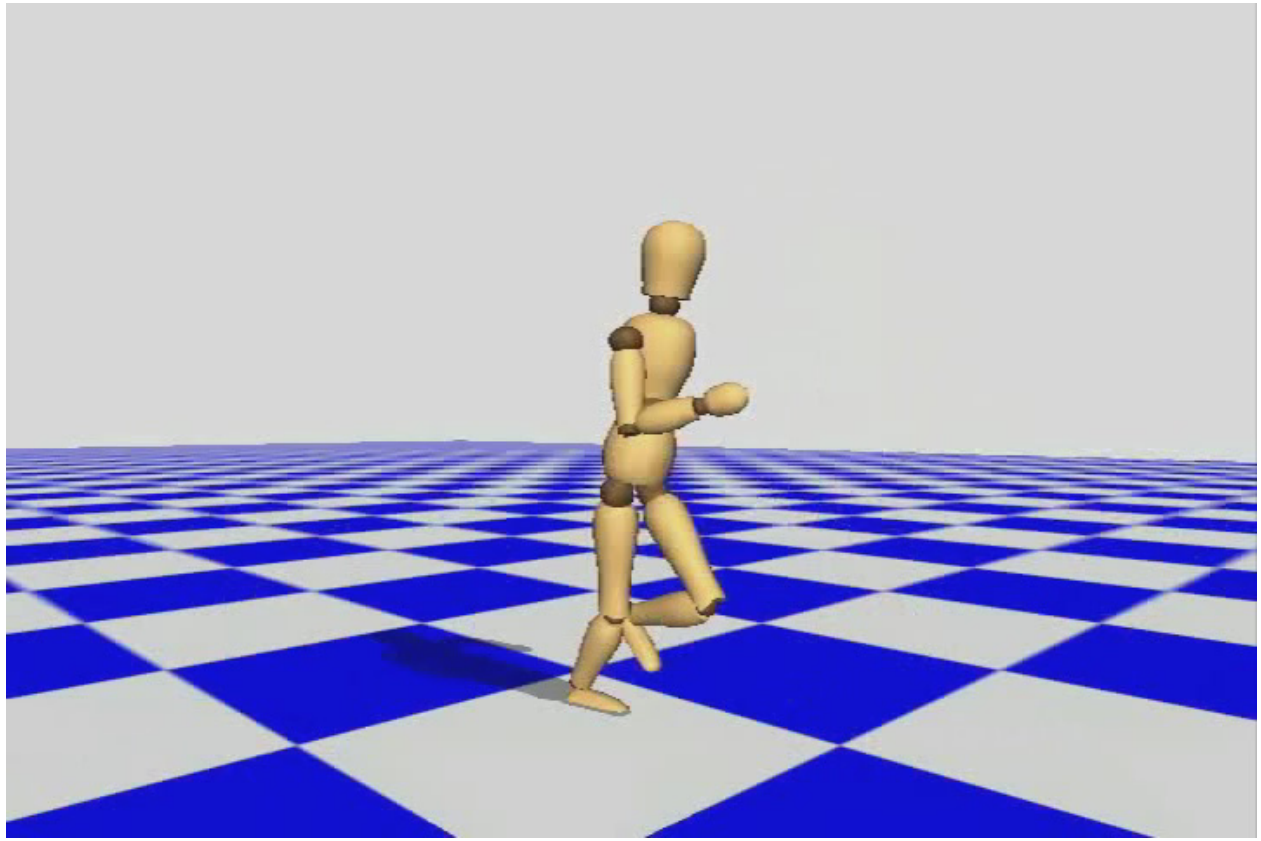
[1]University of Toronto

[2]Disney Research, Pittsburgh

Why do we need a controller?

- Our definitions of controller: given the current state of the body produce the forces (joint torques) that advance the state so that a particular motion is performed

Why do we need a controller?



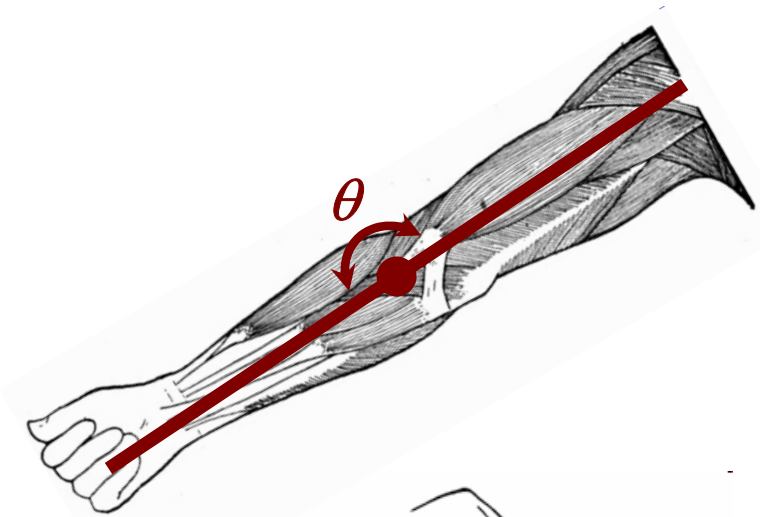
[*Yin, Loken, van de Panne, Siggraph 2007*]

- Our definitions of controller: given the current state of the body produce the forces (joint torques) that advance the state so that a particular motion is performed

Proportional Control

- Given current orientation (θ) and the desired orientation (θ_d) the torque can be specified simply as a function of the difference between the two:

$$\tau = k_p(\theta_d - \theta)$$



Torsional linear spring

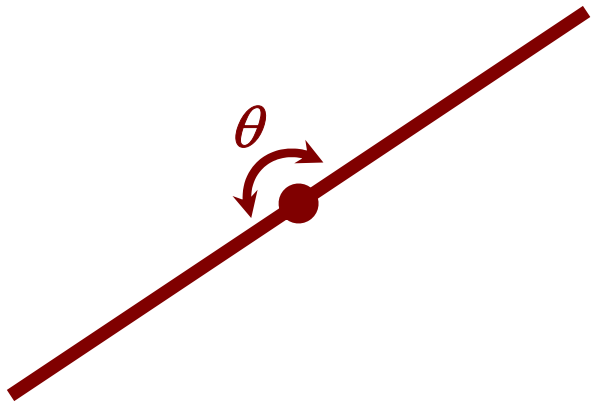
- k_p – *spring constant (gain)*

Problem: spring stores energy, which results in sinusoidal oscillations

Proportional Derivative (PD) Control

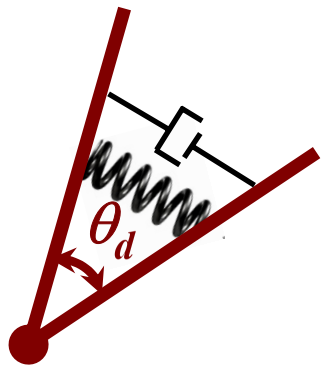
- Given current orientation (θ) and angular velocity ($\dot{\theta}$) and desired orientation (θ_d) the torques can be specified using the following strategy:

$$\tau = k_p(\theta_d \text{ ¡ } \theta) \text{ ¡ } k_d\dot{\theta}$$



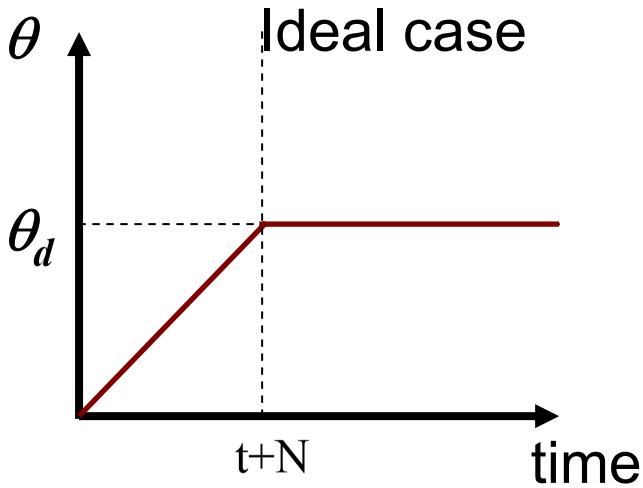
Damped torsional linear spring

- k_p – *spring constant (gain)*
- k_d – damping *gain*

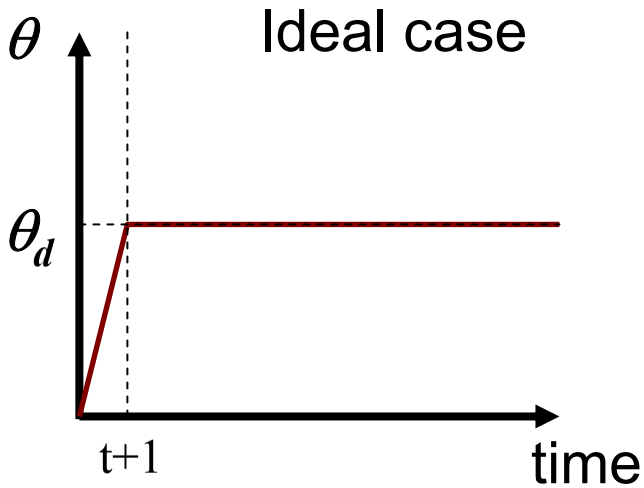


Key Insight: proper values for these constants are important for performance

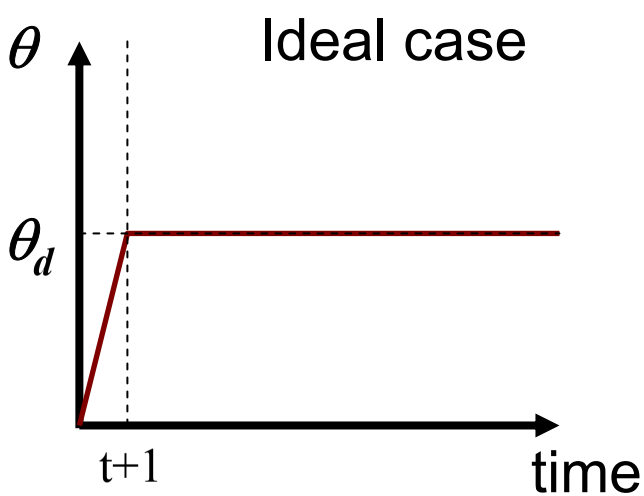
Proportional Derivative (PD) Control

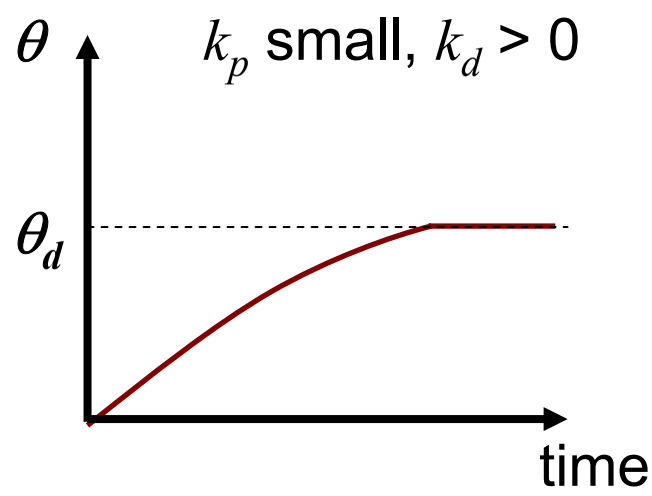


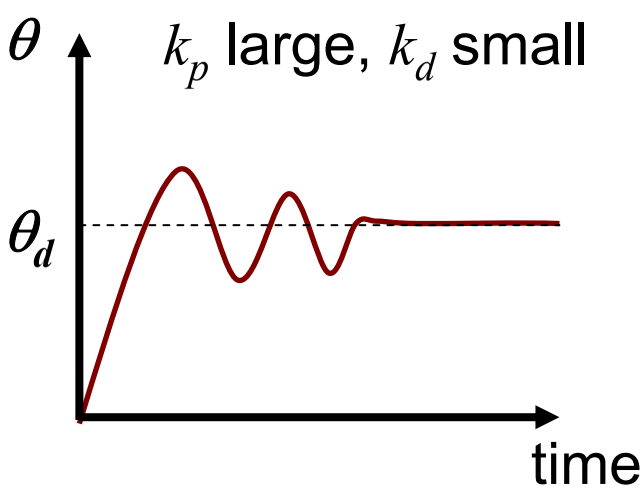
Proportional Derivative (PD) Control

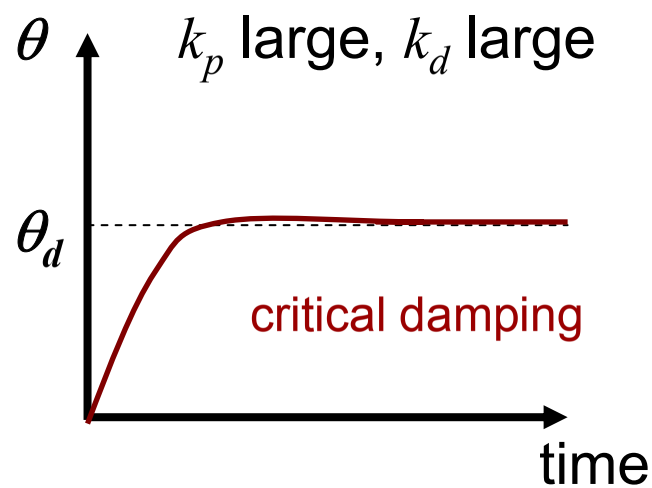


Proportional Derivative (PD) Control



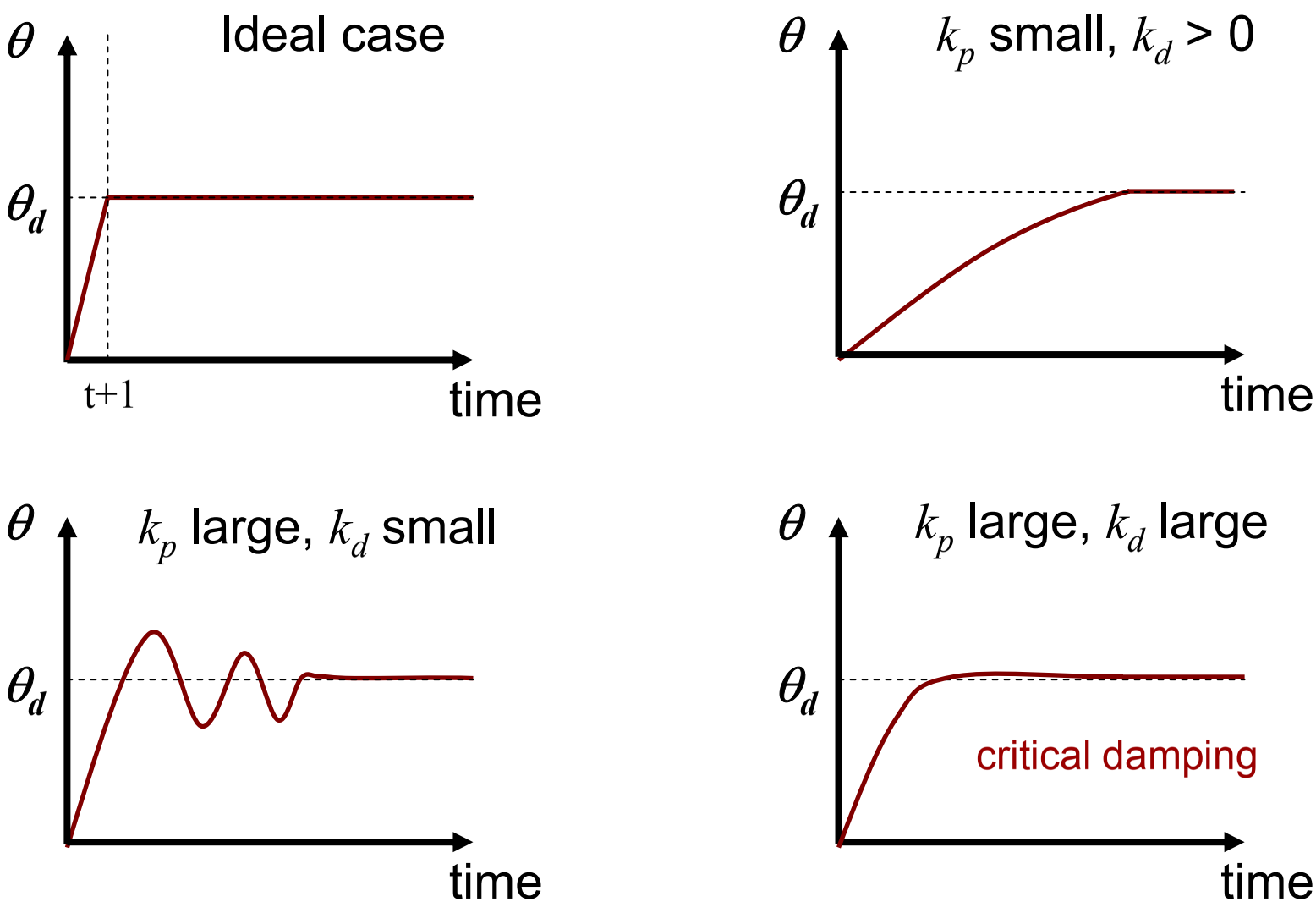






Very large values for constants can make simulation unstable

Proportional Derivative (PD) Control



Empirical rule of thumb: $k_d = 0.1 k_p$

PD Control – Generalization to many DOFs

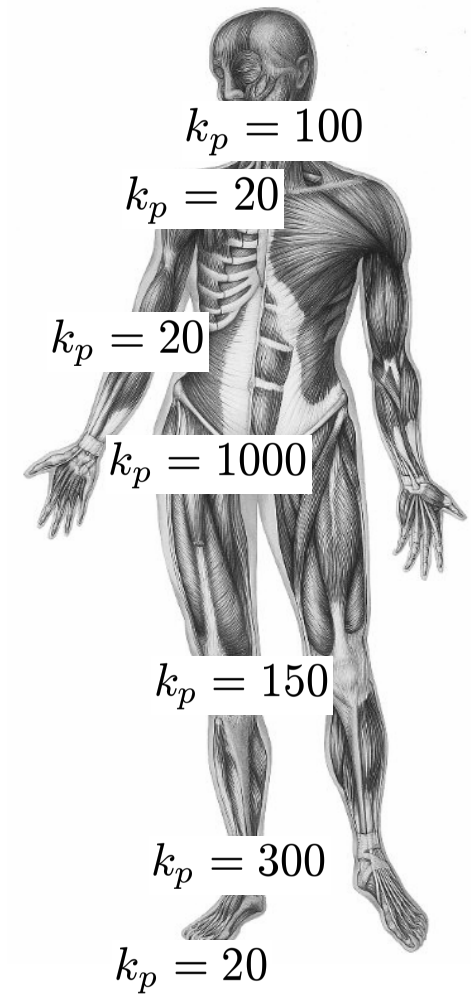
- This trivially generalizes to multiple degrees of freedom (including linear degrees of freedom):

$$\tau = k_p(\theta_d \text{ ¡ } \theta) \text{ ¡ } k_d\dot{\theta}$$

- Do we want the same gains for all joints?

- Answer: Typically not, these should be representative of muscle strength and response that is different

- Problem: PD control only works for simple atomic motions



[*Wang et. al., 2009*]

PD Control – Generalization to many DOFs

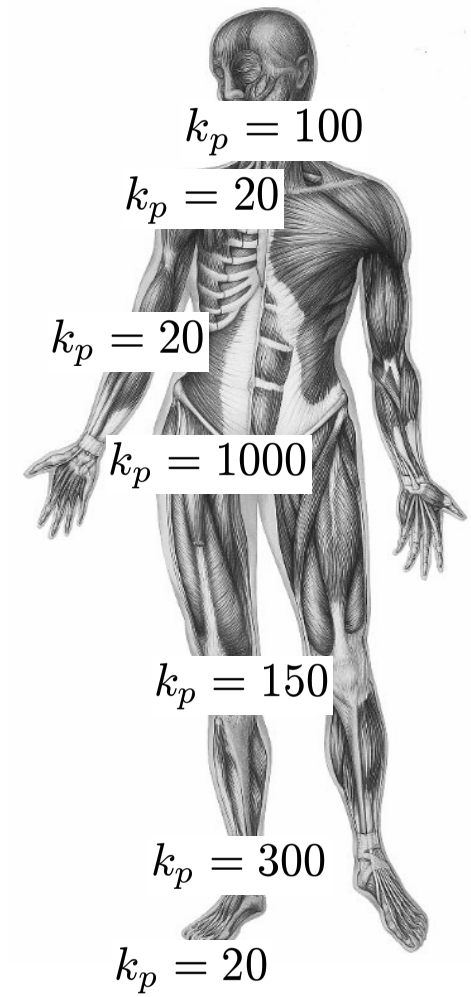
- This trivially generalizes to multiple degrees of freedom (including linear degrees of freedom):

$$\tau = k_p(\mathbf{u}_d \text{ ¡ } \mathbf{u}) \text{ ¡ } k_d\dot{\mathbf{u}}$$

- Do we want the same gains for all joints?

- Answer: Typically not, these should be representative of muscle strength and response that is different

- Problem: PD control only works for simple atomic motions



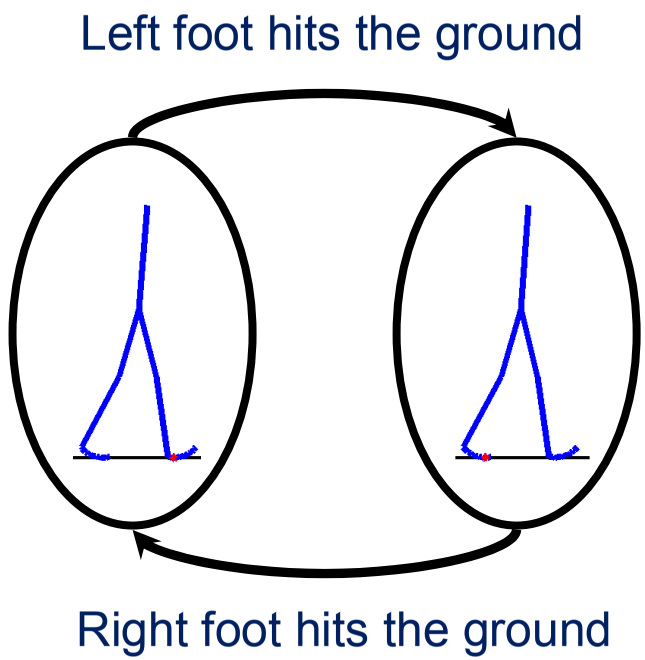
[*Wang et. al., 2009*]

State-space controllers

- Break up a complex behavior into simple sections for which (possibly different) but relatively simple control laws are executed

- Actually you have seen 2-state controller already (remember *kneed walker*)

- We typically need to switch control laws at contact



- That was relatively simple model in 2D, what happens in more complex high-DOF 3D models?

Control in 3D

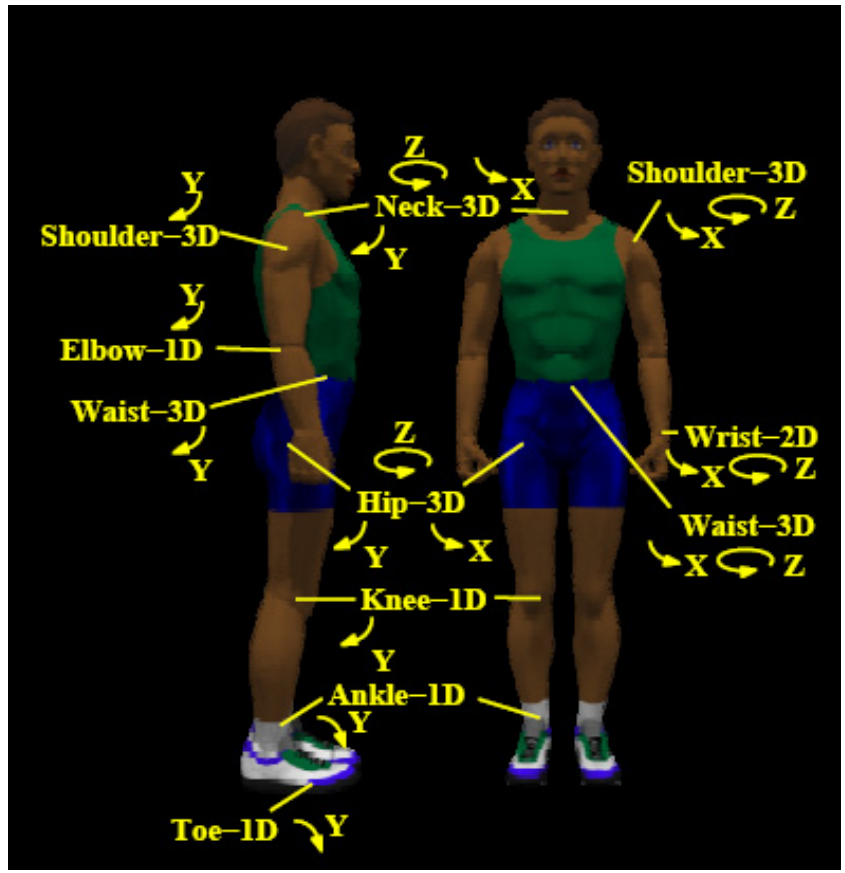
- Control in 3D is challenging
  - There are more degrees of freedom
  - There are move ways a body can fall
  - There is simply more to control

- Therefore typically
  - We need more complex control laws within states
  - Parameters of controllers are harder to tune by hand

- With respect to lower-dimensional models we may also have multiple contact states

3D Human Models

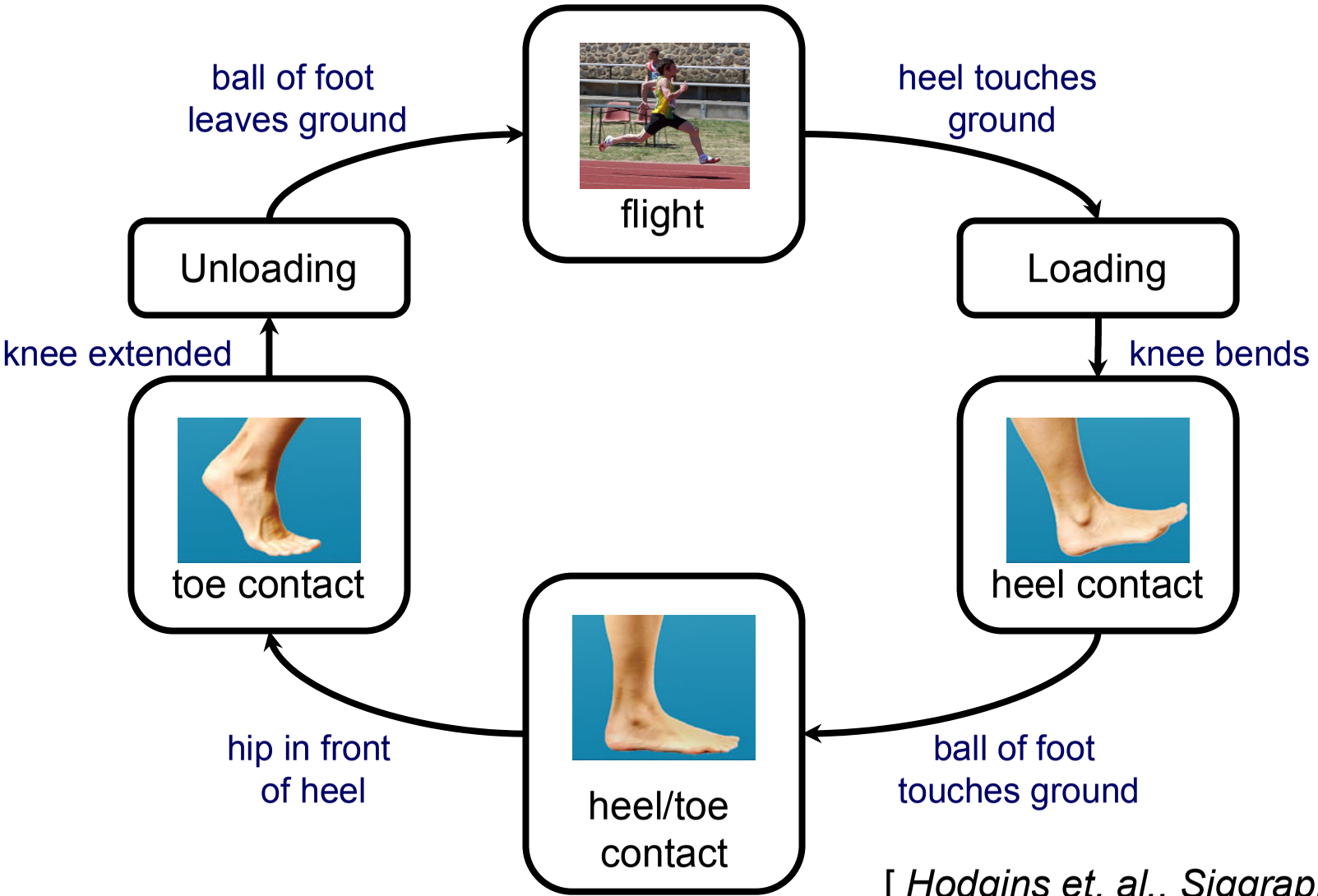
- Between 15 and 17 rigid bodies connected with rotary joints, allowing 22 to 32 controlled DOF
- Body segment densities obtained from biomechanical data
- Mass and moments of inertia calculated from the polygonal model



[*Hodgins et. al., Siggraph 1995*]

Human Running

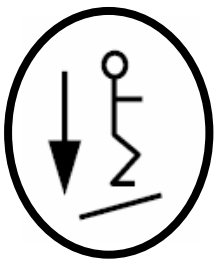
Proper foot contact requires more states



[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting

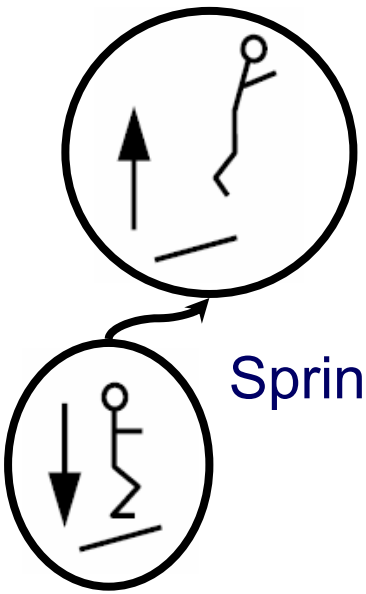
Complex Non-cyclic Motions: Vaulting



Compression

[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



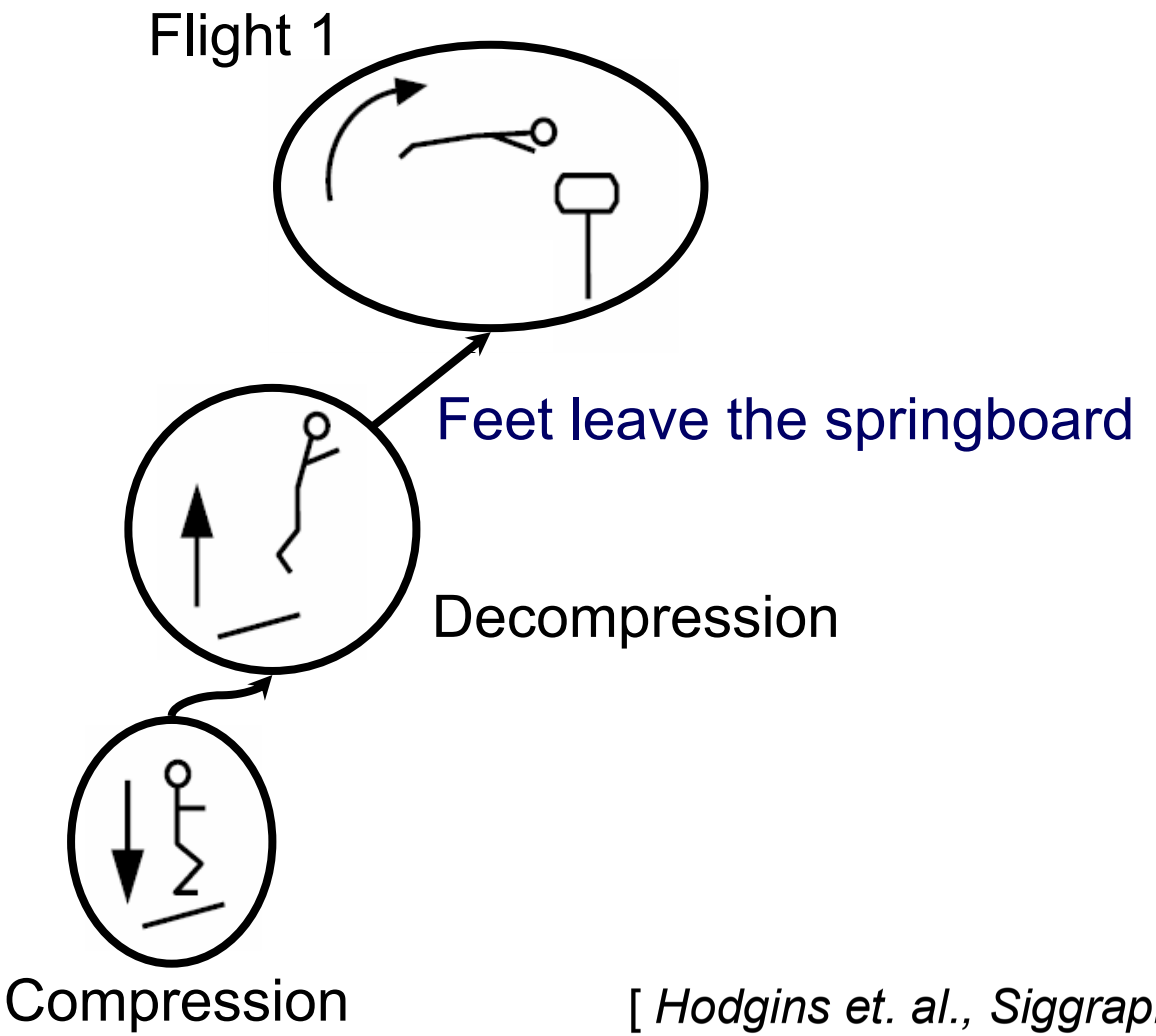
Decompression

Springboard reaches maximum compression

Compression

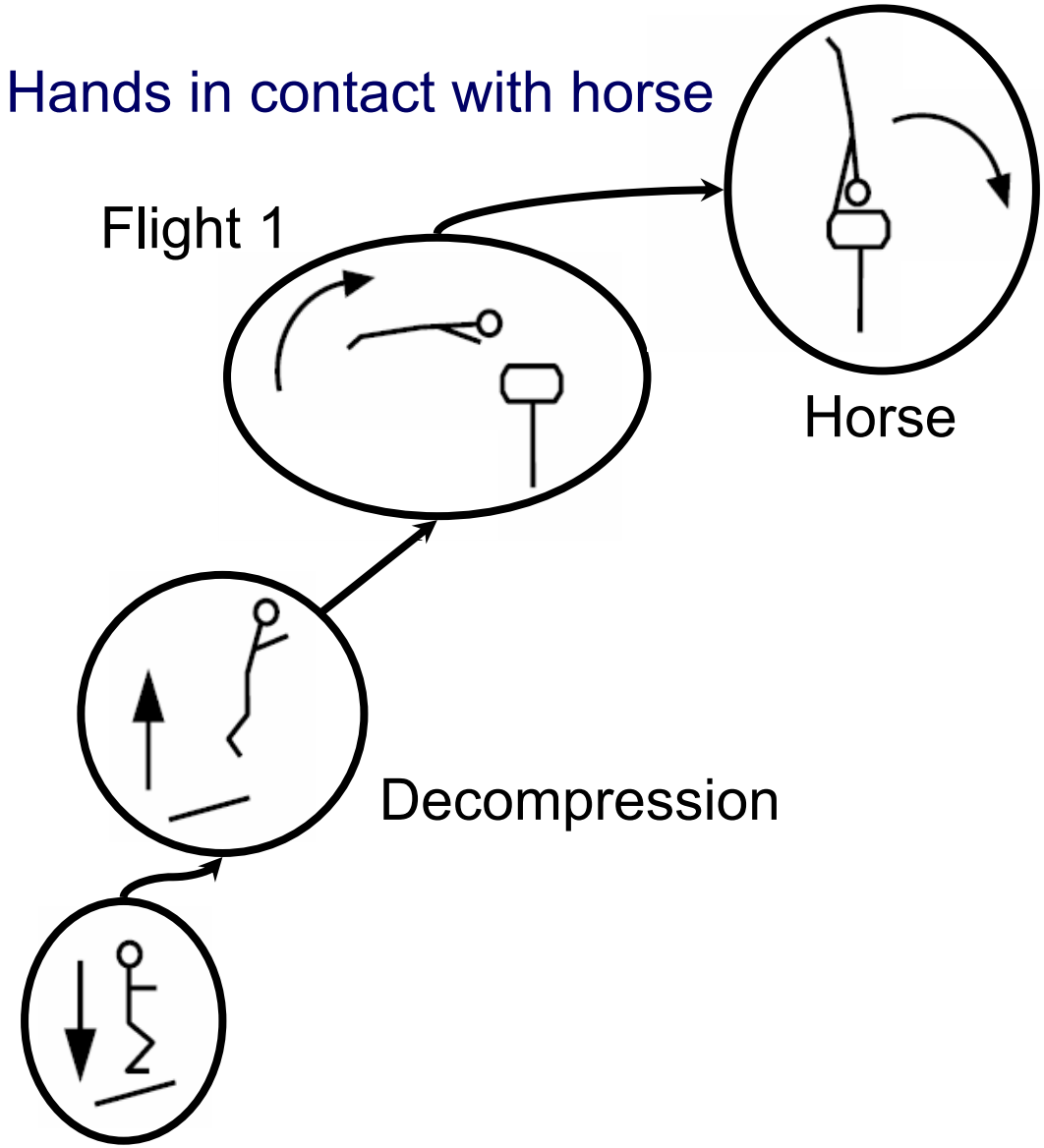
[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



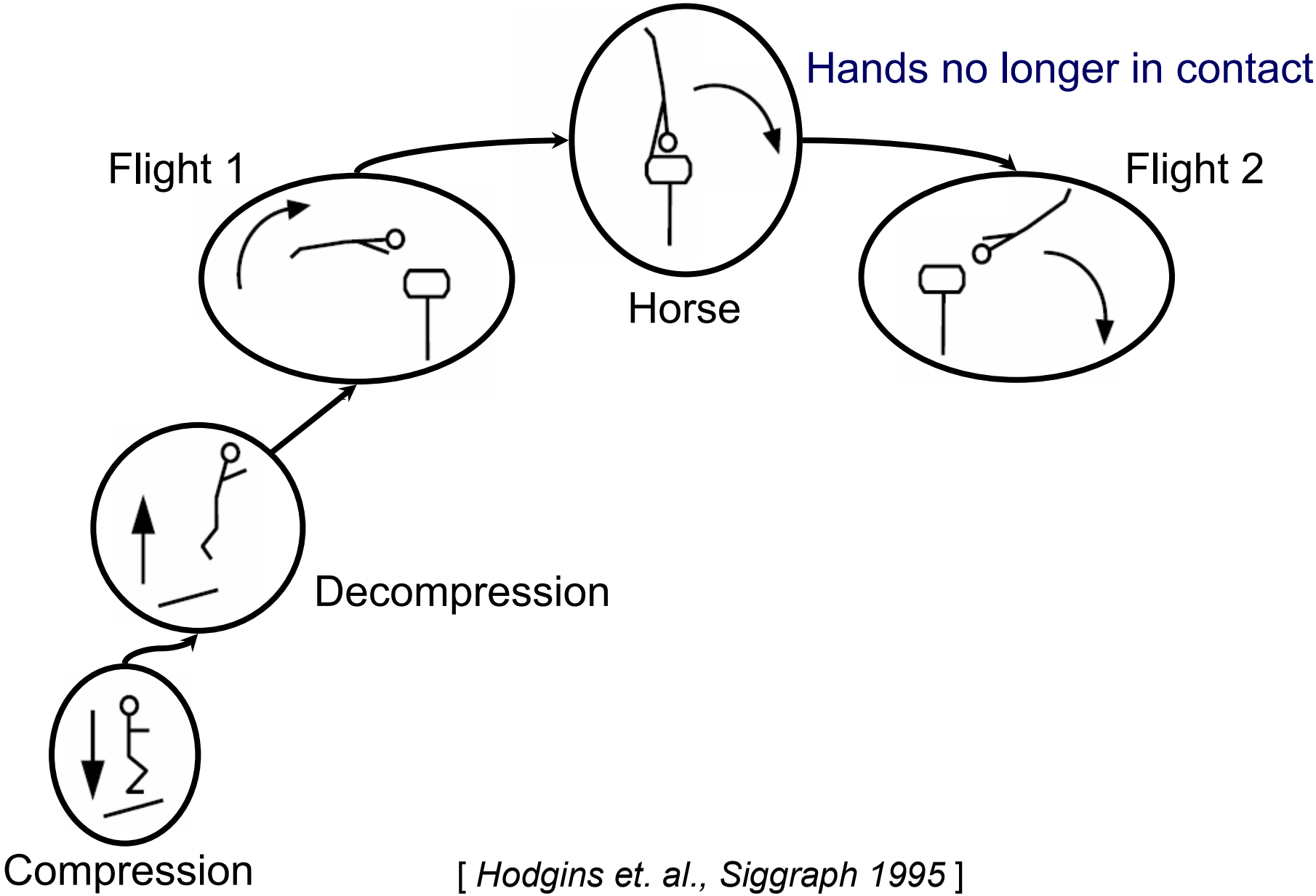
[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



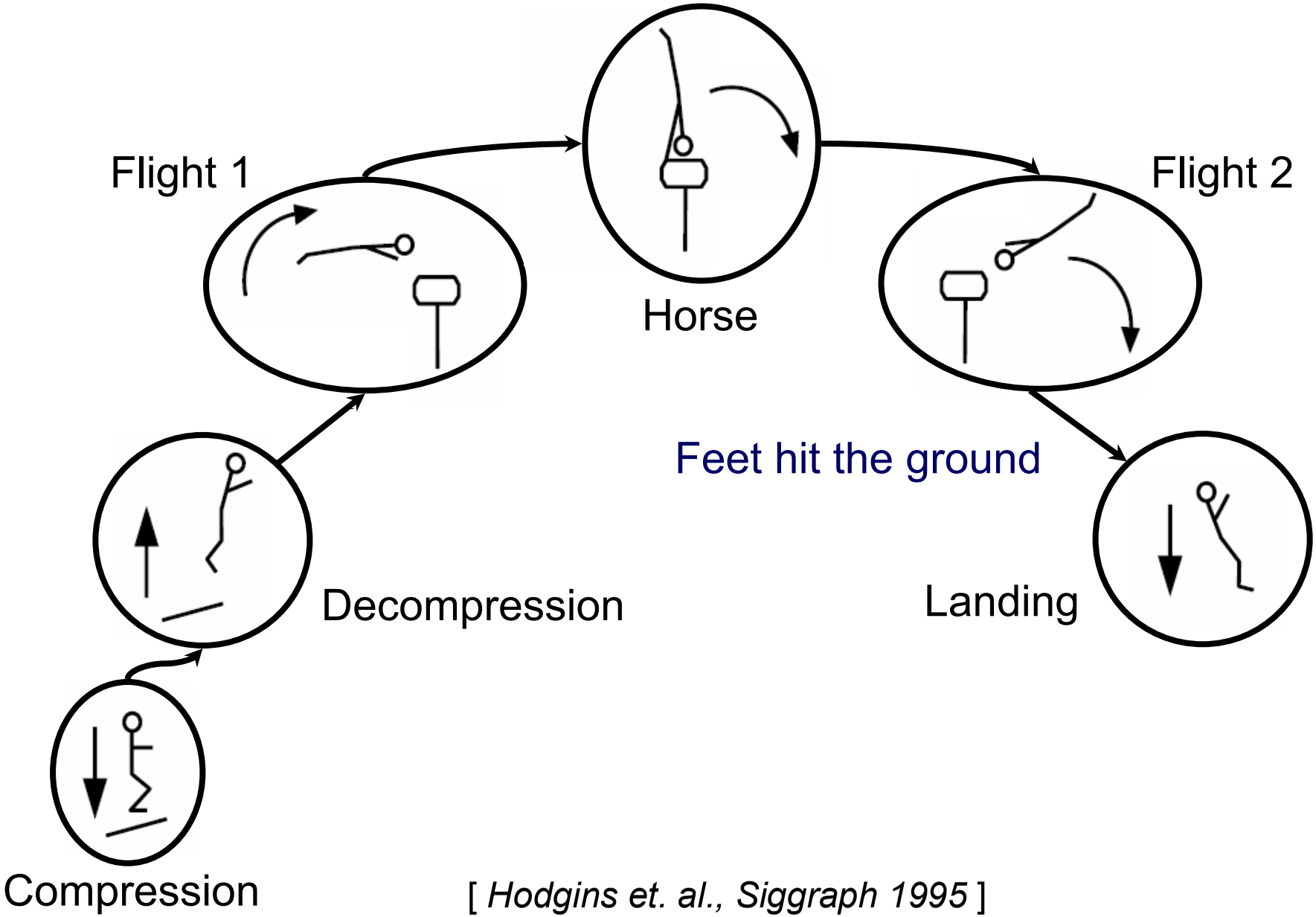
[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



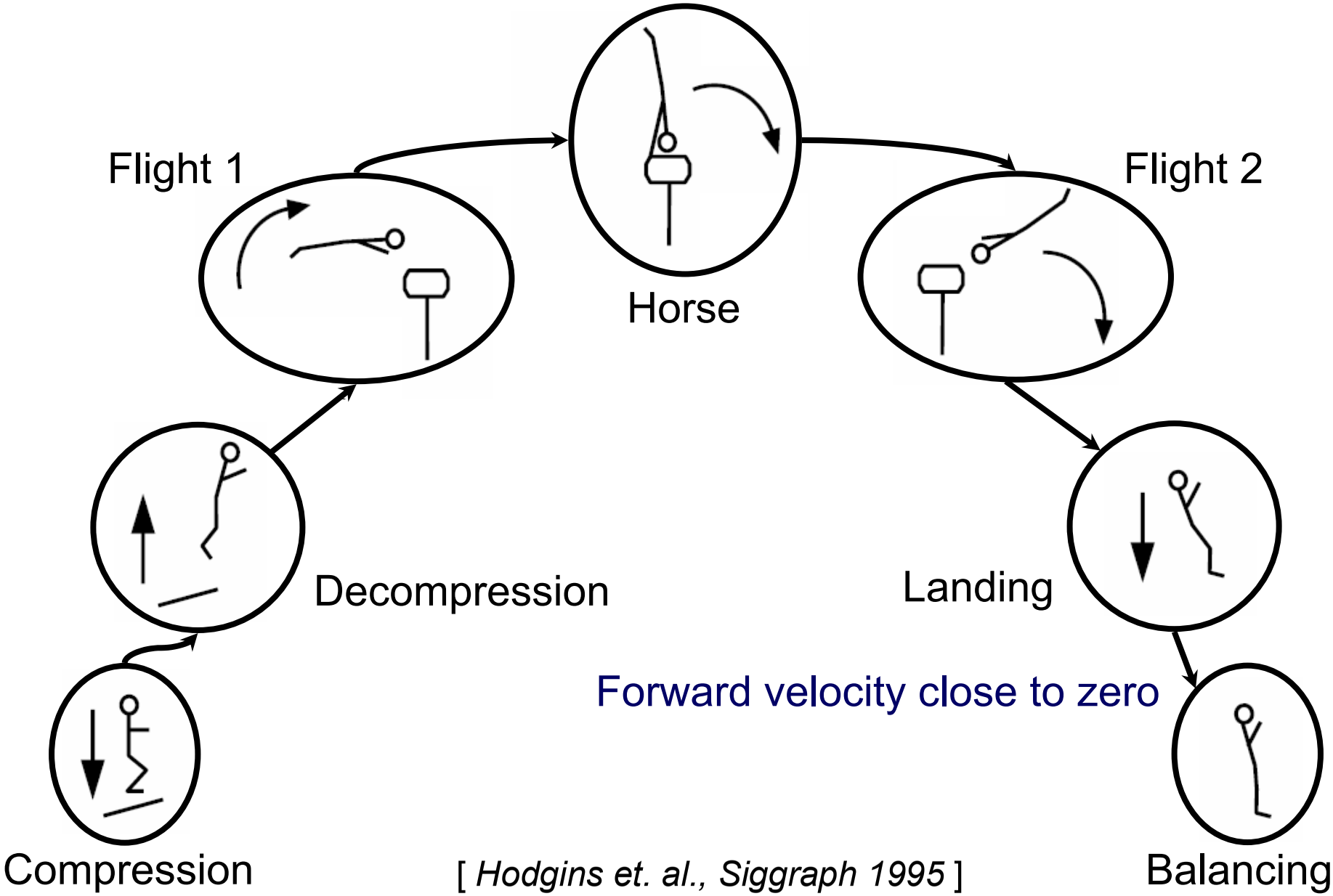
[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



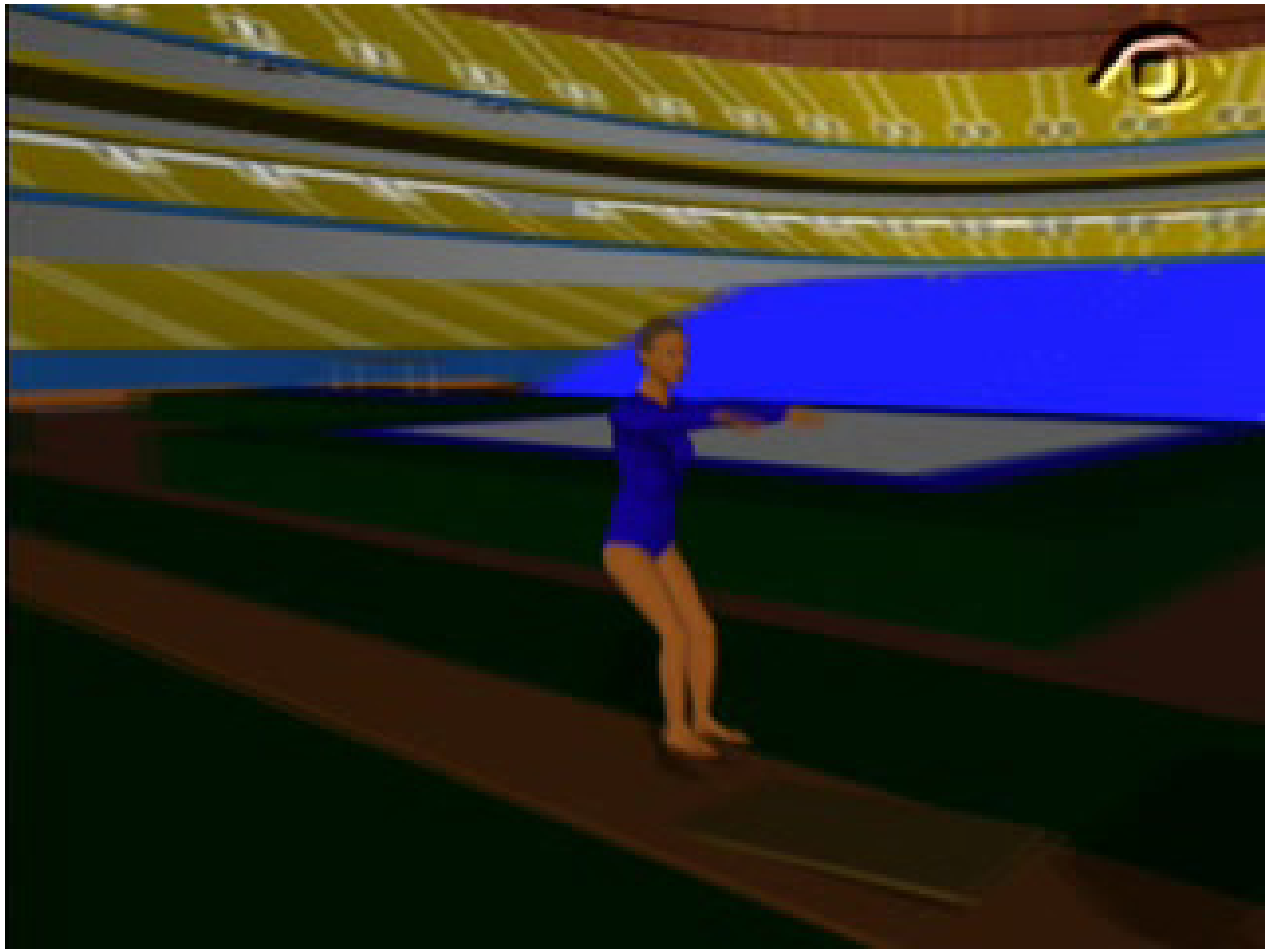
[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



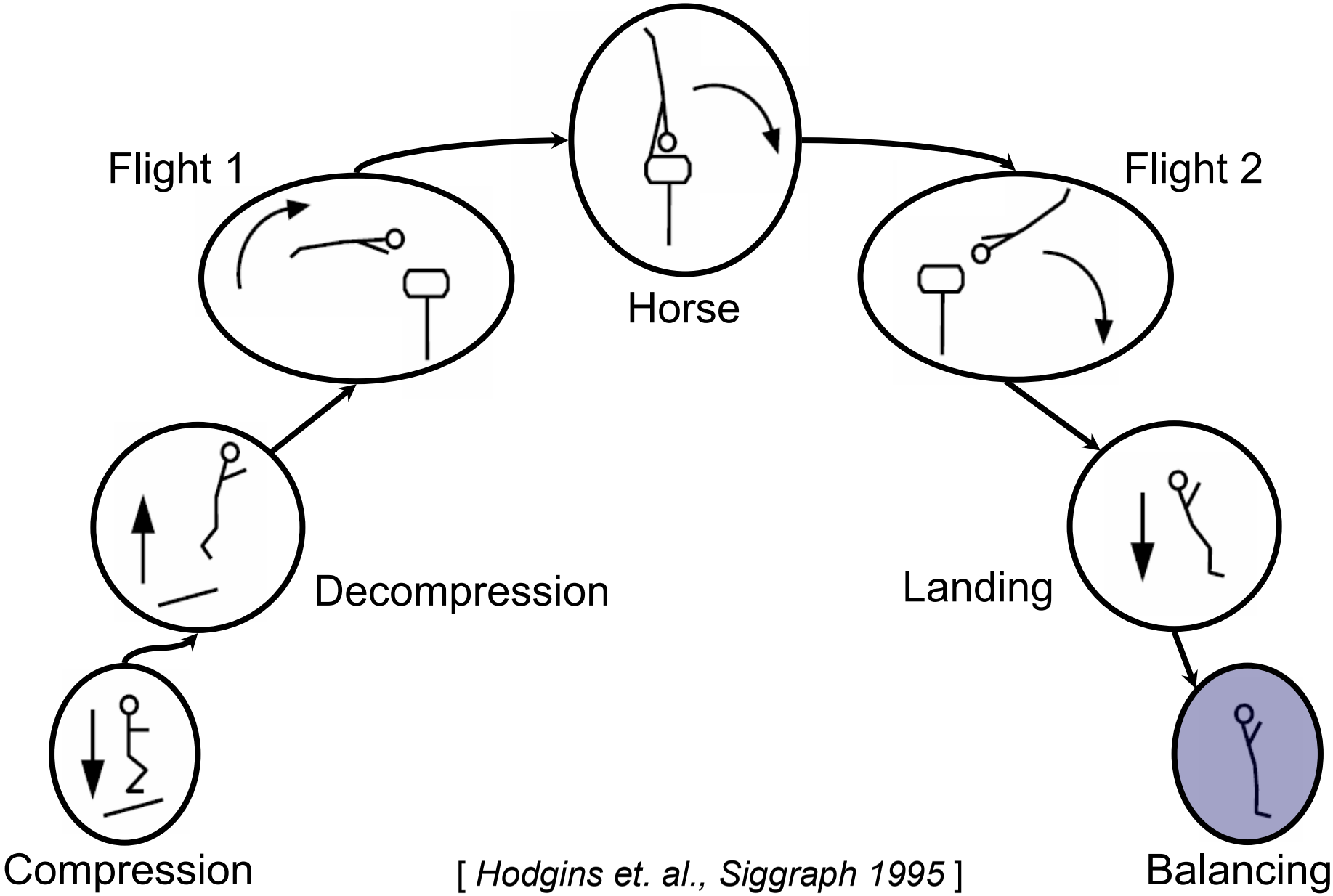
[*Hodgins et. al., Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



[*Hodgins, Wooten, Brogan, O'Brien, Siggraph 1995*]

Complex Non-cyclic Motions: Vaulting



[*Hodgins et. al., Siggraph 1995*]

Stability and Balance

- Balance is important to ensure that motion cannot be easily corrupted by the environmental perturbations and/or dynamic interactions (critical for computer graphics … may be important for computer vision)

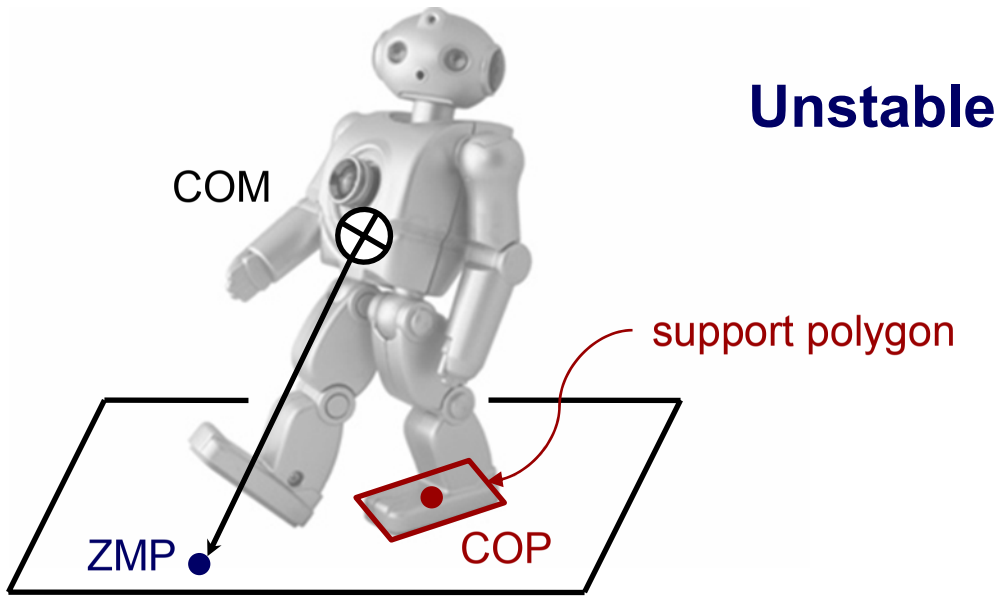
- One can think of controller as: (1) generating a nominal set of forces that produce the ideal motion, (2) generating a set of forces that ensure recovery from imminent failure

- Static balance – the body is in balance at all times (if the controller stops generating forces the robot will not fall)

- Dynamic balance – generate forces that push the body toward more balanced states

Dynamic Balance: ZMP

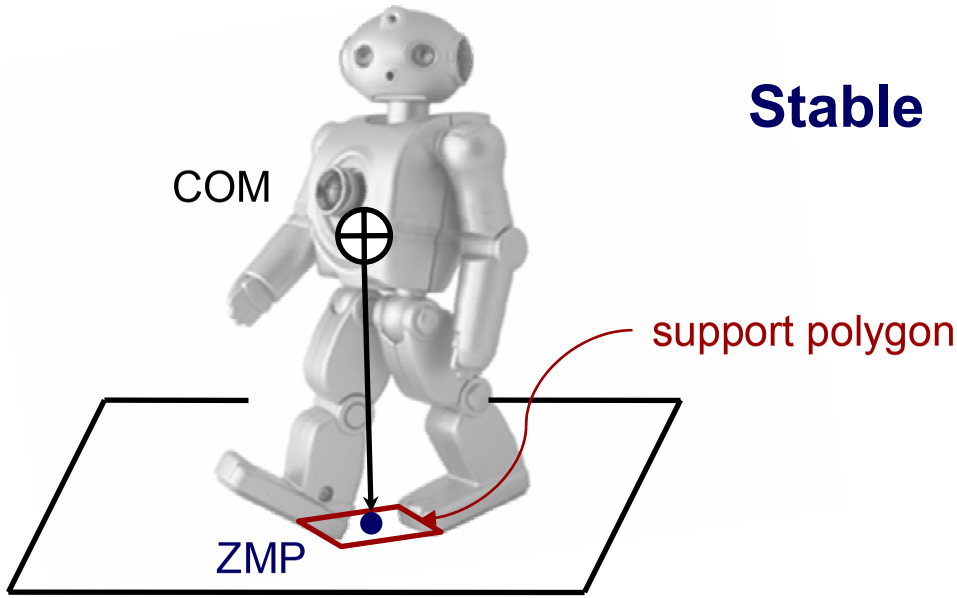
- Zero Moment Point (ZMP): point in space where dynamic force at the contact of the foot would not produce any moment about the center of mass (COM)



- If ZMP is within the support polygon (convex polygon encompassing all contact points), then motion is stable
- To force the ZMP to stay within this polygon: control the COM such that ZMP stays close to Center of Pressure (COP)

Dynamic Balance: ZMP

- Zero Moment Point (ZMP): point in space where dynamic force at the contact of the foot would not produce any moment about the center of mass (COM)



- If ZMP is within the support polygon (convex polygon encompassing all contact points), then motion is stable
- To force the ZMP to stay within this polygon: control the COM such that ZMP stays close to Center of Pressure (COP)

Dynamic Balance: ZMP

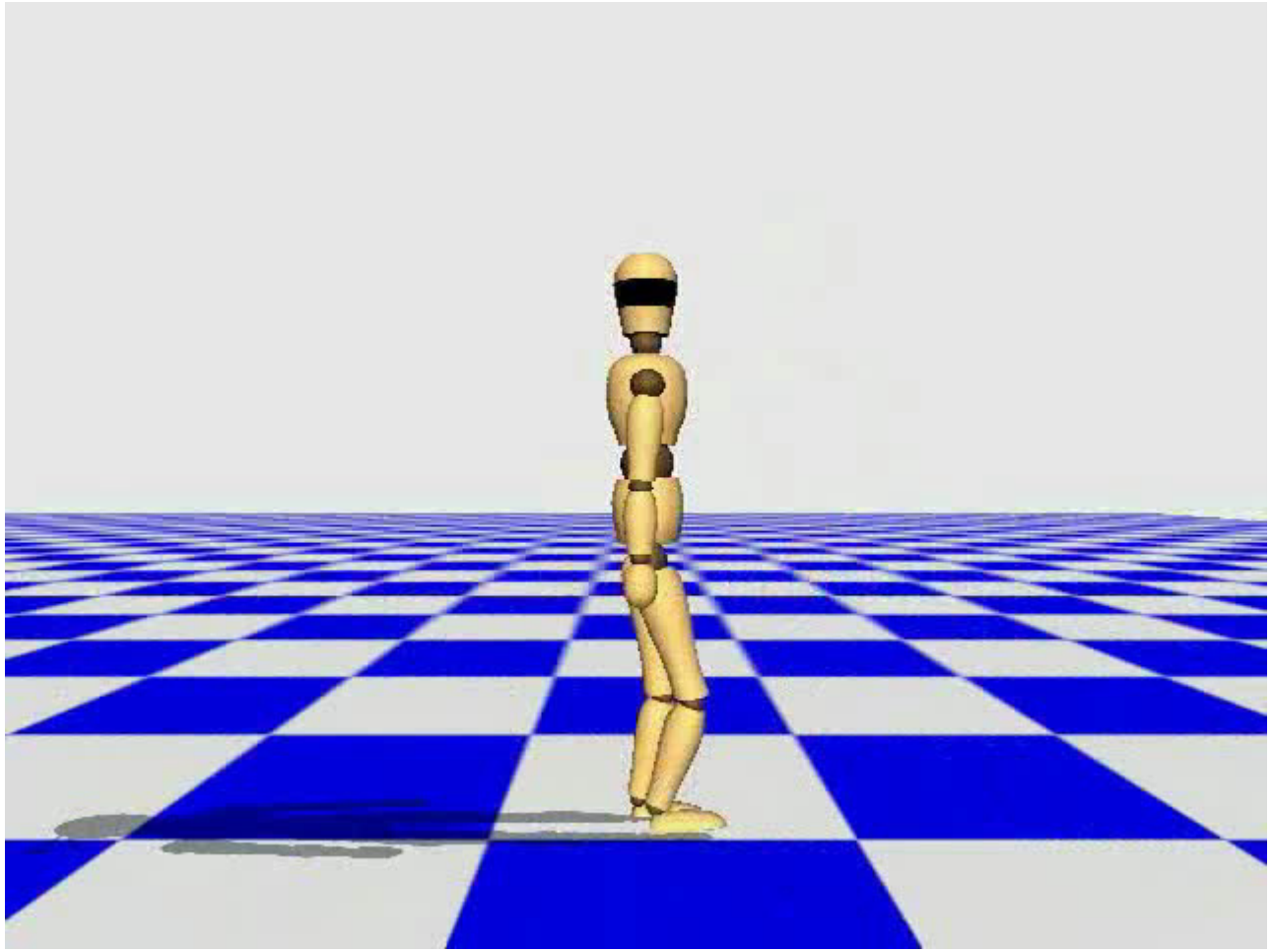
- ZMP balance control is used extensively in robotics
  - Ensures that expensive robots (almost) never fall

- Makes a number of often unrealistic assumptions
  - Surface is planar
  - Surface has sufficient friction to ensure that foot never slips

- Humans sometimes violate ZMP stability, why?
  - Because it is may be efficient to do so
    (and biological organisms like to save energy if they can)
  - We can utilize gravity to our advantage
    (remember walking is nearly passive)

SIMBICON Demo

- 4-state controller with a feedback balance term



[*Yin, Loken, van de Panne, Siggraph 2007*]

Estimating Parameters of Controllers

- Controllers have many parameters
  - For each DOF in each state of the state-space controller we need to specify: k_p, k_d, θ_d
  - We also need to start the simulation from the good initial state: $\mathbf{q}_0, \dot{\mathbf{q}}_0$

Typical use case:

$$30 \times 3 \times 6 + 60 = \boxed{600 \text{ parameters}}$$

DOFs states initial state

Estimating Parameters of Controllers

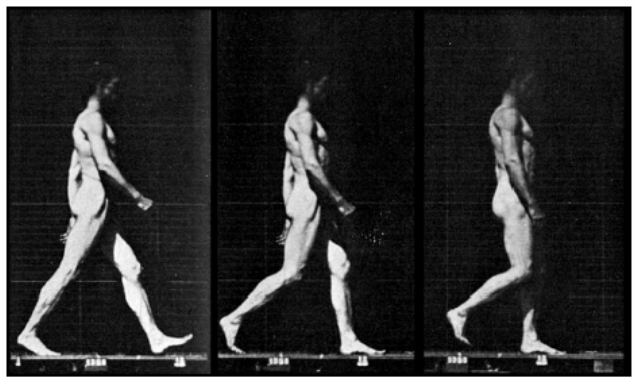
- Controllers have many parameters
  - For each DOF in each state of the state-space controller we need to specify: k_p, k_d, θ_d
  - We also need to start the simulation from the good initial state: $\mathbf{q}_0, \dot{\mathbf{q}}_0$

- Alternatives
  - Specify everything by hand: tedious and often hard to produce realistic looking motions
  - Optimization with MoCap or biomechanical properties as objectives: computationally challenging but more realistic

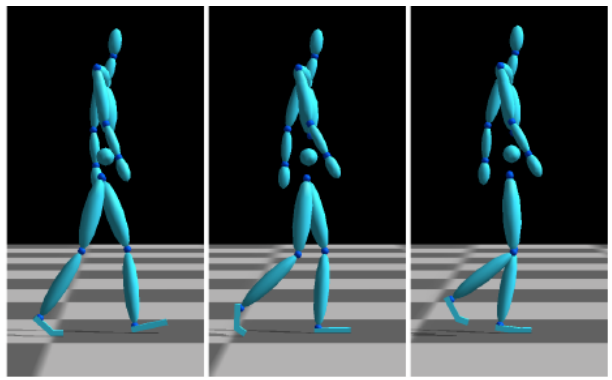
[*Yin, Carlos, Beaudoin, van de Panne, Siggraph 2008*]
[*Wang, Fleet, Hertzmann, Siggraph Asia 2009*]

Learning controller based on biomechanical properties

- Biomechanical properties can be used to optimize a walking 4-state PD controller with a weighted objective function
- User specified objectives (encourage desired motion)
- Practical objectives (encourage sustainable gait)
- Biomechanics objectives (encourage natural motion)



[*Muybridge, 1887*]



[*Weng et. al., 2009*]

Learning controller based on biomechanical properties

- Biomechanical properties can be used to optimize a walking 4-state PD controller with a weighted objective function

- User specified objectives (encourage desired motion)
  - Average speed
  - Average step length

- Practical objectives (encourage sustainable gait)

- Biomechanics objectives (encourage natural motion)

Learning controller based on biomechanical properties

- Biomechanical properties can be used to optimize a walking 4-state PD controller with a weighted objective function

- User specified objectives (encourage desired motion)

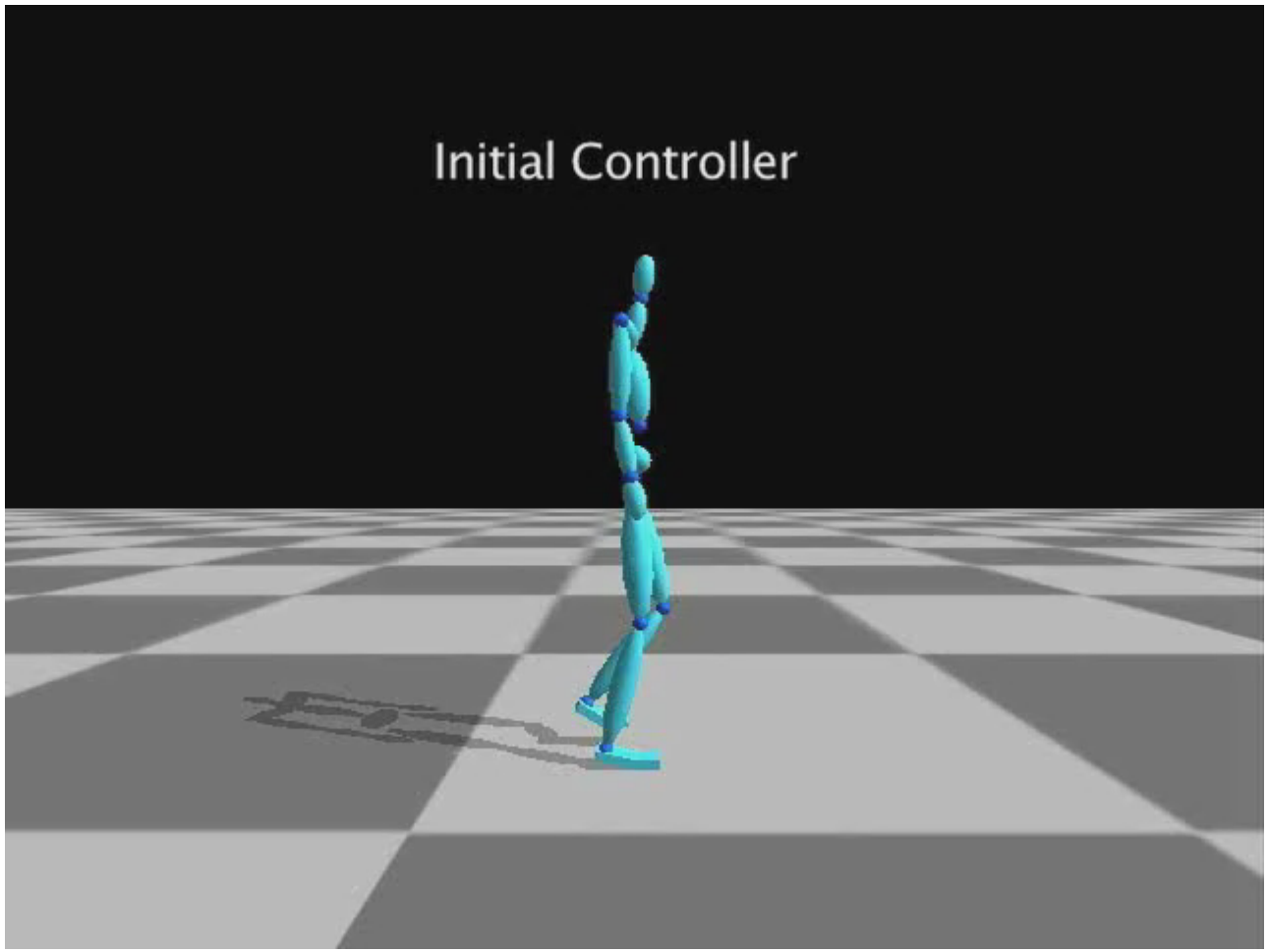
- Practical objectives (encourage sustainable gait)
  - Minimize lateral and vertical sway of center of mass
  - Foot clearance criterion
  - Left-right symmetric timing of the gait

- Biomechanics objectives (encourage natural motion)

Learning controller based on biomechanical properties

- Biomechanical properties can be used to optimize a walking 4-state PD controller with a weighted objective function
- User specified objectives (encourage desired motion)
- Practical objectives (encourage sustainable gait)
- Biomechanics objectives (encourage natural motion)
  - Minimize angular momentum of the body about COM
  - Minimize lateral and vertical head sway for stabilization of visual and vestibular systems during walking
  - Maximize efficiency (by minimizing torque output)
  - Panelize deviations from empirical power ratio between ankle-knee-hip for walking

Learning Walking Controllers

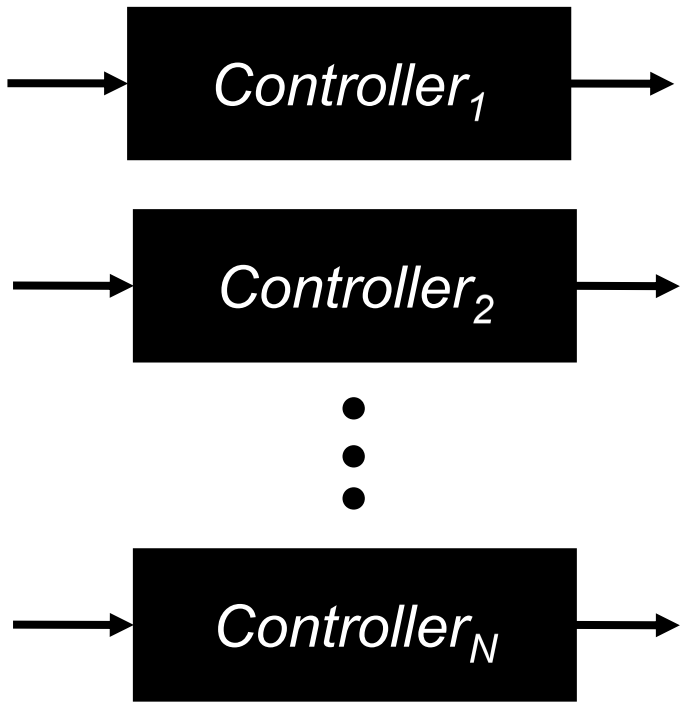


[*Weng, Fleet, Hertzmann, Siggraph Asia 2009*]

Composing Controllers

- So far we only looked at building controllers for relatively simple motions what about more complex activities?

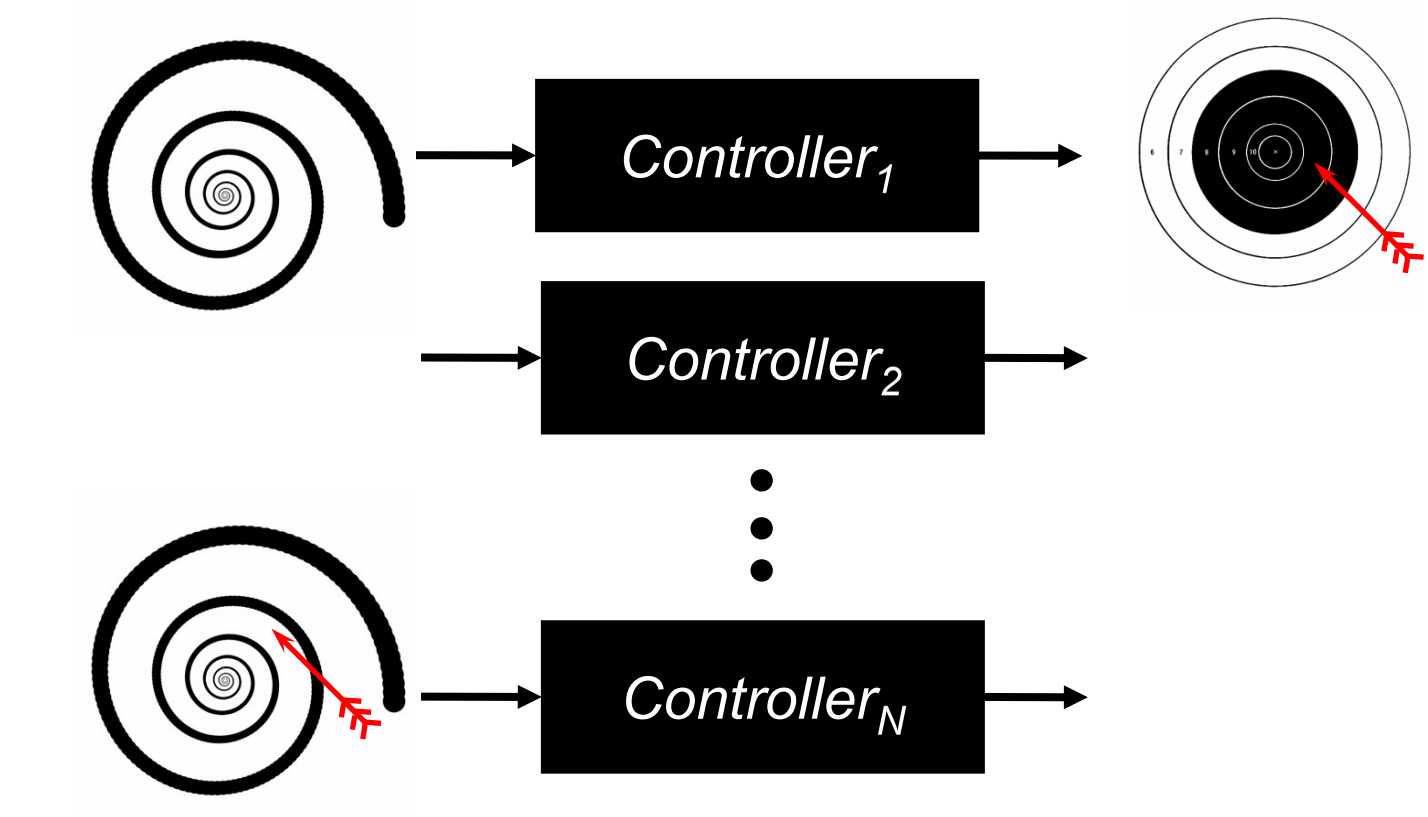
- We need a way to compose controllers



Composing Controllers

- Manually – again tedious
- More general approach - controller abstraction:
  - Pre-conditioning: pose, environment, balance
  - Post-conditioning: pose, environment, balance

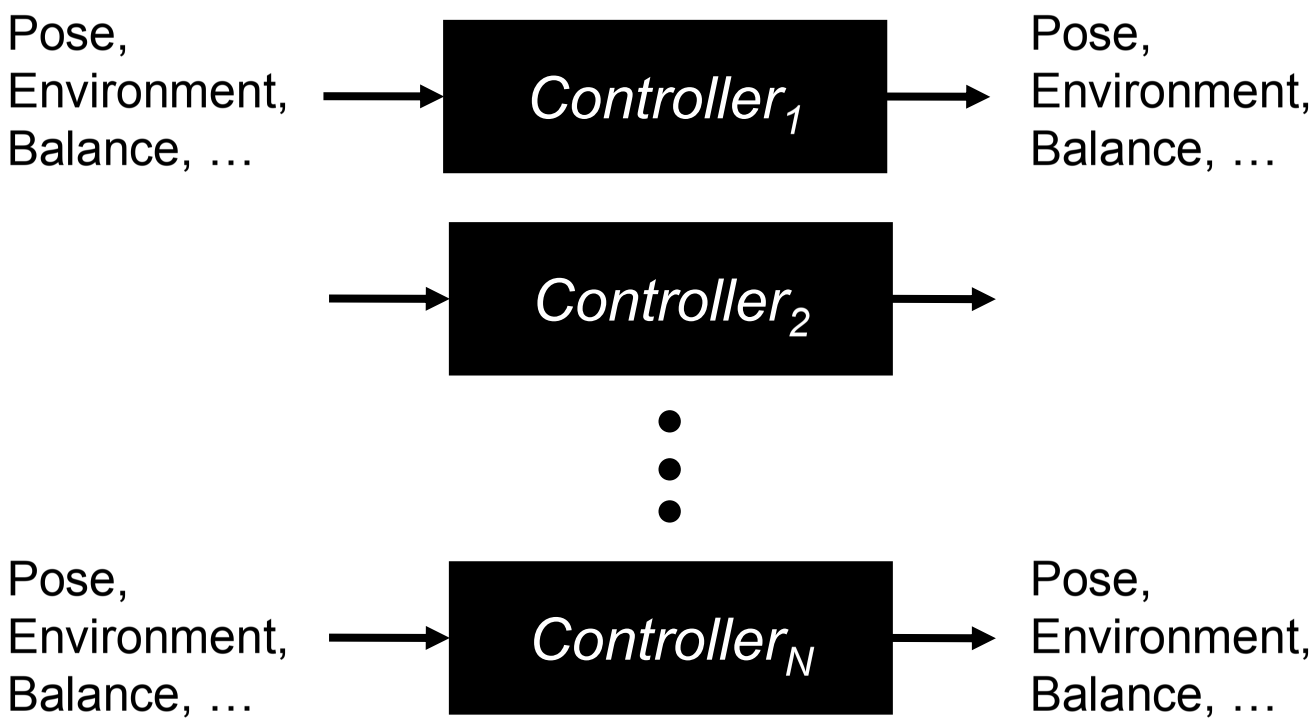
[*Faloutsos, van de Panne, Terzopoulos, Siggraph 2001*]



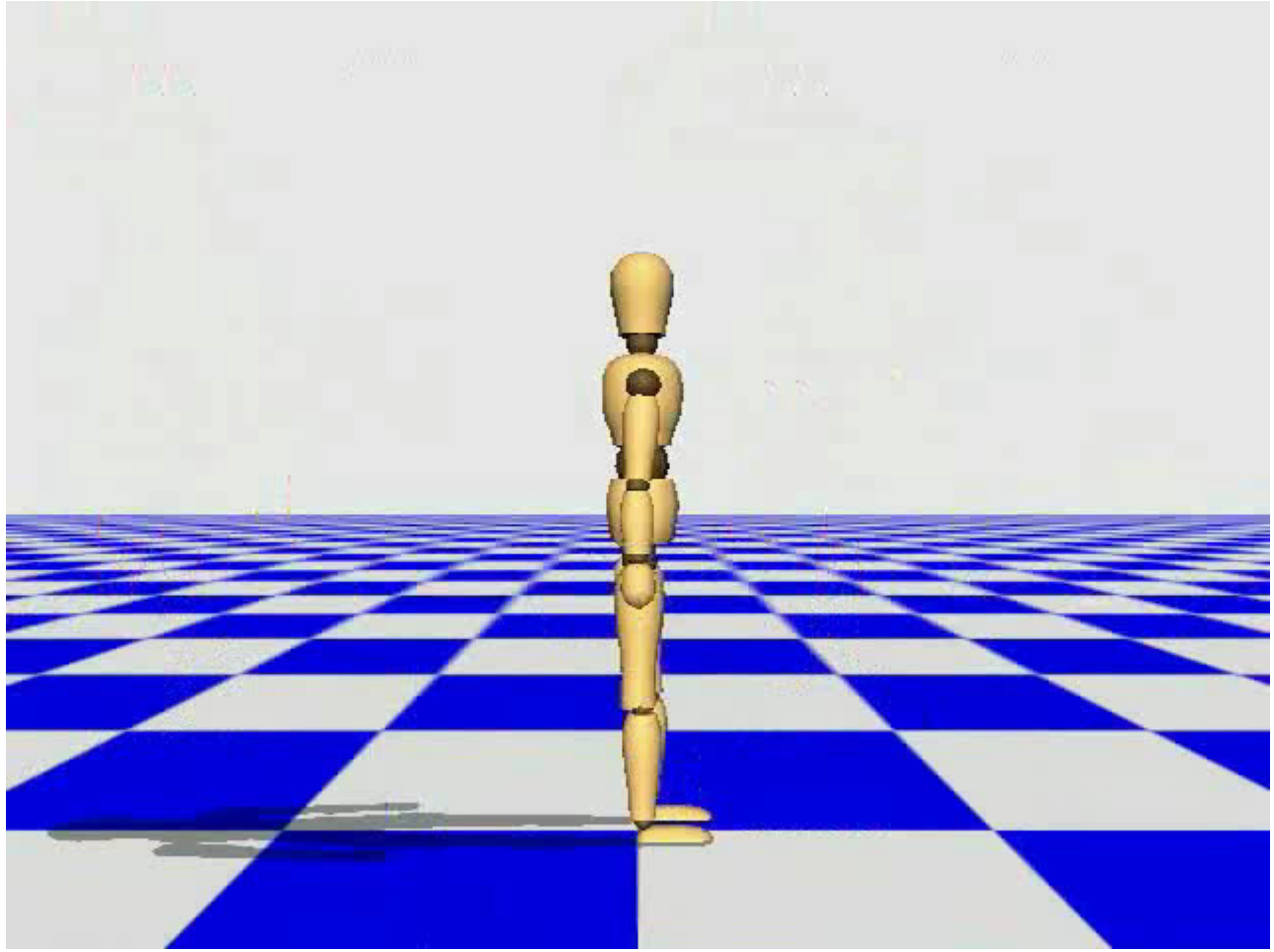
Composing Controllers

- Manually – again tedious
- More general approach - controller abstraction:
  - Pre-conditioning: pose, environment, balance
  - Post-conditioning: pose, environment, balance

[*Faloutsos, van de Panne, Terzopoulos, Siggraph 2001*]



Composing Controllers



[*Yin, Loken, van de Panne, Siggraph 2007*]

Motion-Capture Trajectory Control

- State-space controllers require intimate knowledge of the generated motion
- Even if we optimize parameters, we need to hand specify how many states a controller has and when to make transitions
- It is also often hard to embed stylistic variations into the produced motion

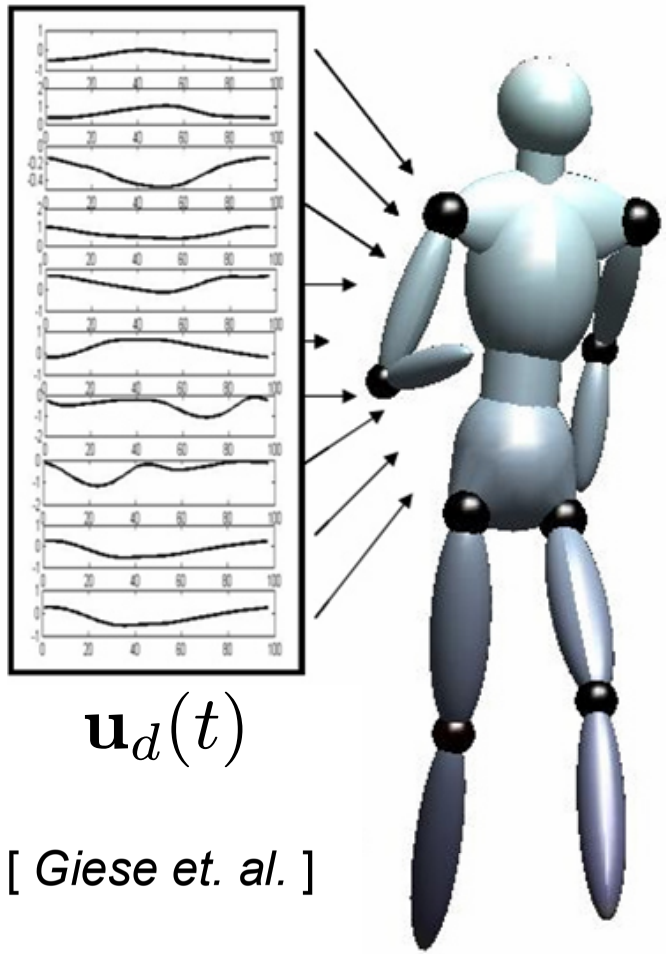
What else can we do?

- We can rely on captured kinematic data instead to drive the dynamic simulation: trajectory control or tracking

Motion-Capture Trajectory Control

- We can treat motion capture data as set of dense desired poses for a controller, now instead of having a θ_d for every DOF in every state of the state-space controller, we have $\theta_d(t)$

- We can again use the PD controller strategy, but now with a different set point at every time instant



$\mathbf{u}_d(t)$

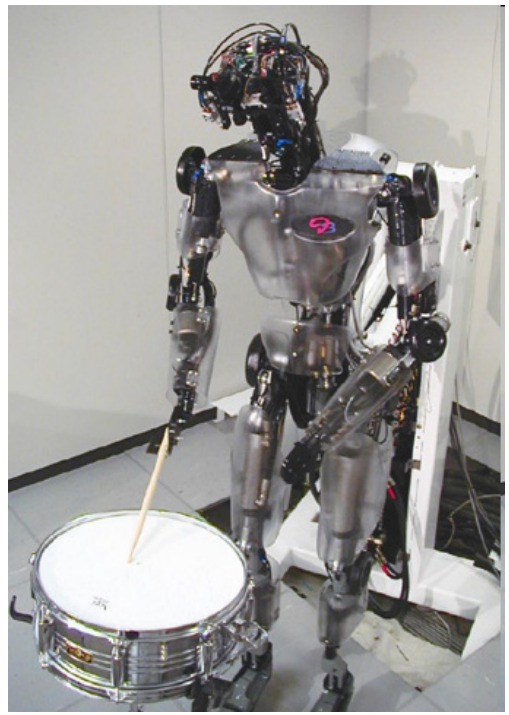
[*Giese et. al.*]

Motion-Capture Trajectory Control

- We can treat motion capture data as set of dense desired poses for a controller, now instead of having a θ_d for every DOF in every state of the state-space controller, we have $\theta_d(t)$

- We can again use the PD controller strategy, but now with a different set point at every time instant

- Benefit: captured kinematic motion is typically noisy (which results in small violations of physical realism), this smoothes it out and enhances the realism

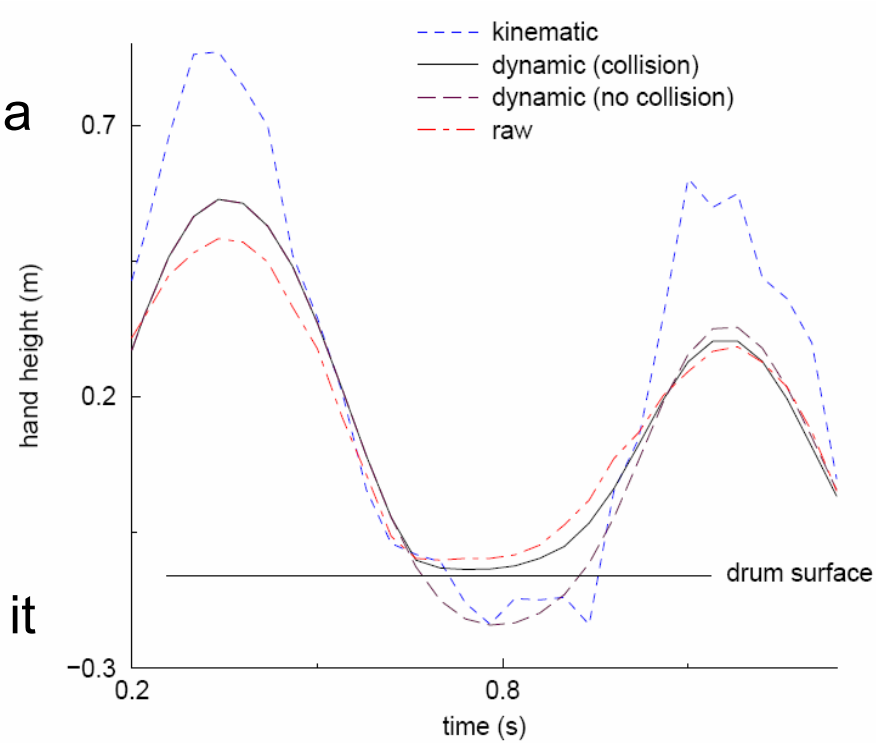


Motion-Capture Trajectory Control

- We can treat motion capture data as set of dense desired poses for a controller, now instead of having a θ_d for every DOF in every state of the state-space controller, we have $\theta_d(t)$

- We can again use the PD controller strategy, but now with a different set point at every time instant

- Benefit: captured kinematic motion is typically noisy (which results in small violations of physical realism), this smoothes it out and enhances the realism



[*Zordan and Hodgins, 1999*]

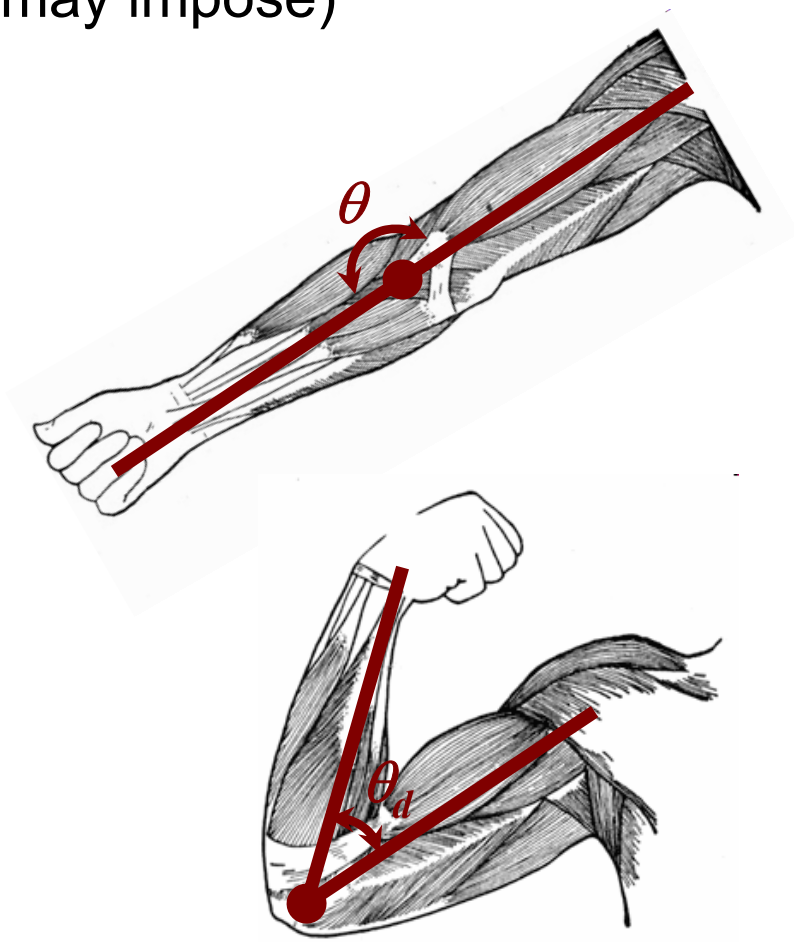
Constraint-Based Trajectory Control

- PD controllers do not take environment into account (produce joint forces that are independent of the constraints that environment may impose)

Can we do better?

- We can treat the desired poses as constraints and use constraint stabilization within the constraint solver to solve for required forces

$$e(\theta) = \theta_d \text{ ¡ } \theta = 0$$



[*Kokkevis, 2004*]

Constraint-Based Trajectory Control

- Constraint-based controller is a form of inverse dynamics

Input: desired values for the degree of freedom
Output: forces needed to reach these desired values

(use inverse kinematics to solve for desired DOFs)

Constraint-Based Trajectory Control

- Constraint-based controller is a form of inverse dynamics

Input: position of *points* rigidly attached to body segments
Output: forces needed to match these positions

(use inverse kinematics to solve for desired DOFs)

Constraint-Based Trajectory Control

- Constraint-based controller is a form of inverse dynamics

Input: position of *points* rigidly attached to body segments
Output: forces needed to match these positions

(use inverse kinematics to solve for desired DOFs)

- More faithful to the data and environment then PD control
- Biomechanical limits on the strength of the muscles can also be taken into account (as constraints)

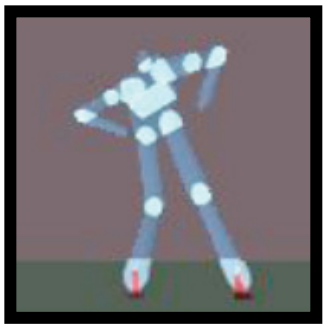
$$0 \cdot \tau \cdot \tau_{max}$$

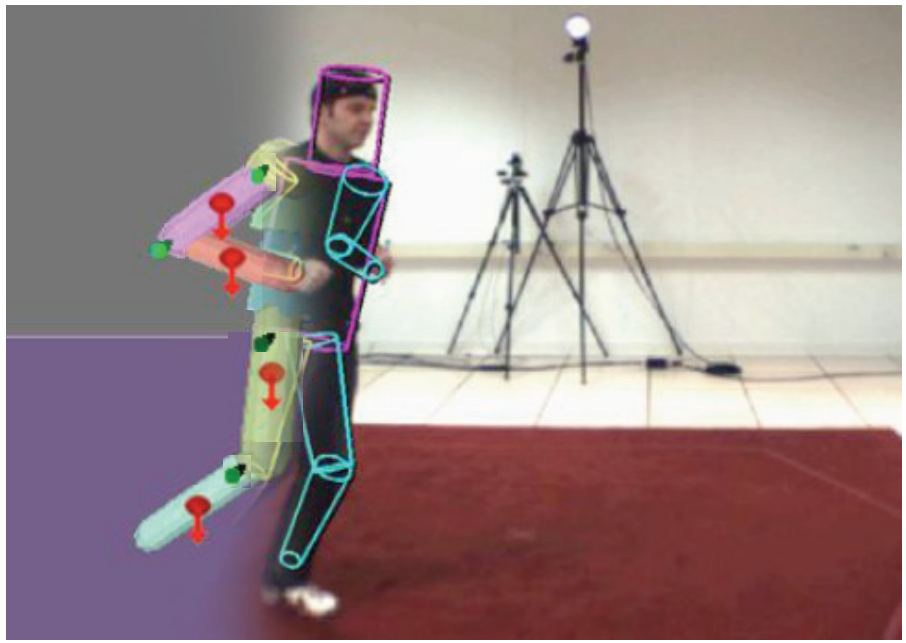
- Only applicable on relatively small time scales (one frame)

Video-Based Human Motion Tracking

[*Vondrak, Sigal, Jenkins, CVPR'08*]

- Trajectory controller as the means of building a full-body dynamical simulation prior for human Bayesian filtering





- Benefits we hope to gain (with respect to kinematic priors)
  - Physical realism in the recovered motion
  - Ability to generalize to new environments

Video-Based Human Motion Tracking

[*Vondrak, Sigal, Jenkins, CVPR'08*]

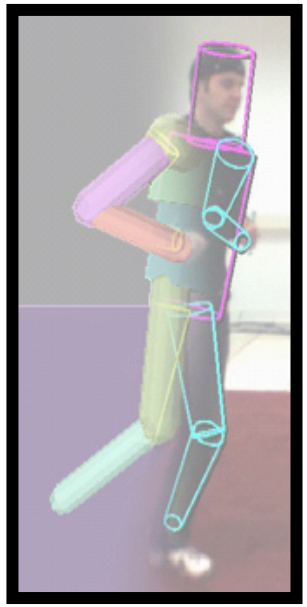
- Assumptions:
  - Known environment
  - Known physical properties

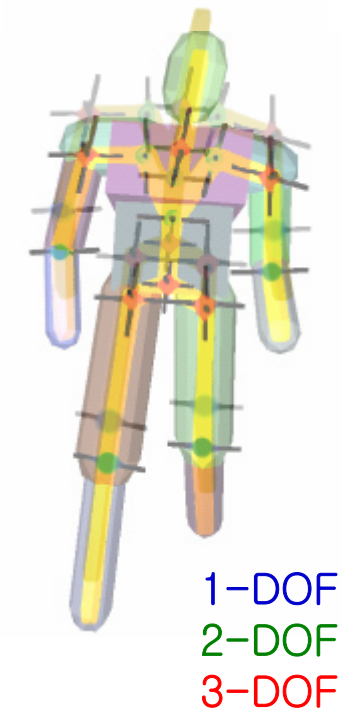
- Kinematic structure:

  13 segments, 31 DOFs

- State:

$$\mathbf{s}_t = [\mathbf{u}_t, \dot{\mathbf{u}}_t, \pi_t] \qquad \mathbf{u}_t \; 2 < ^{31}$$

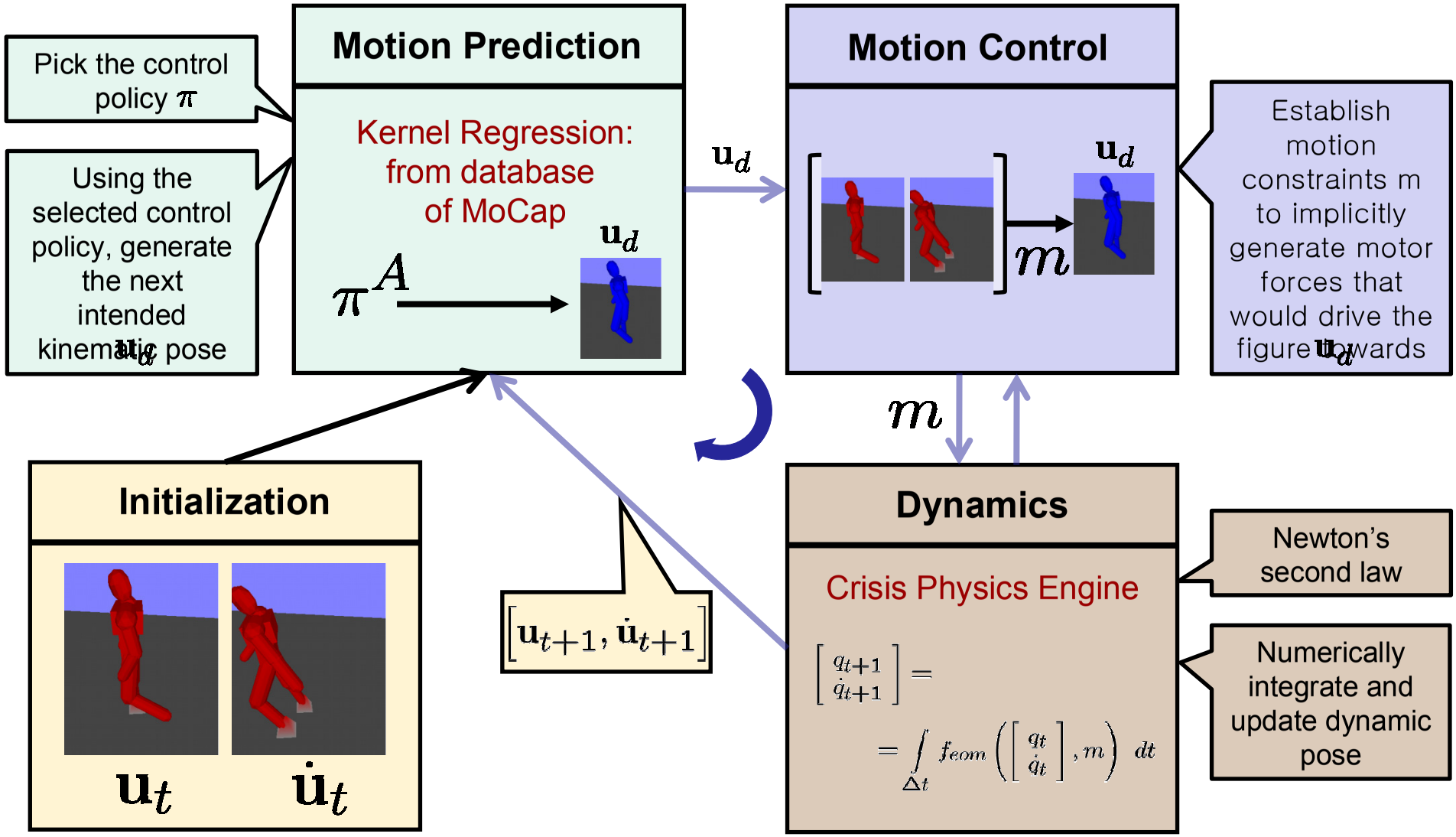




How do we formulate a prior?

Motion Prior: Control Loop

Executed for every hypothesis in our multi-hypothesis PF framework

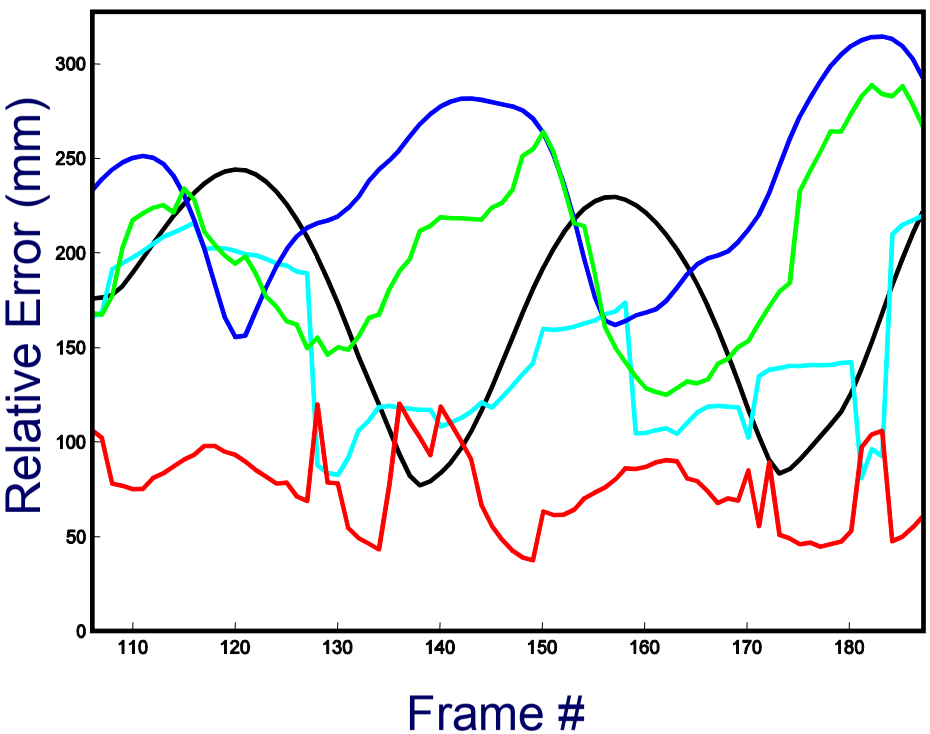


How good is physics-based motion prior?

[*Vondrak, Sigal, Jenkins, CVPR'08*]

Given: a pose of the body from walking MoCap data

Goal: predict the state Δt = 0.5 seconds into the future



No prior:

$$\mathbf{u}_{t+1} = \mathbf{u}_t$$

Constant velocity prior:

$$\mathbf{u}_{t+1} = \mathbf{u}_t + \Delta t \, \dot{\mathbf{u}}_t$$

MoCap-based kinematic prior:

$$\mathbf{u}_{t+1} = KernReg\left(\mathbf{u}_t, \dot{\mathbf{u}}_t\right)$$

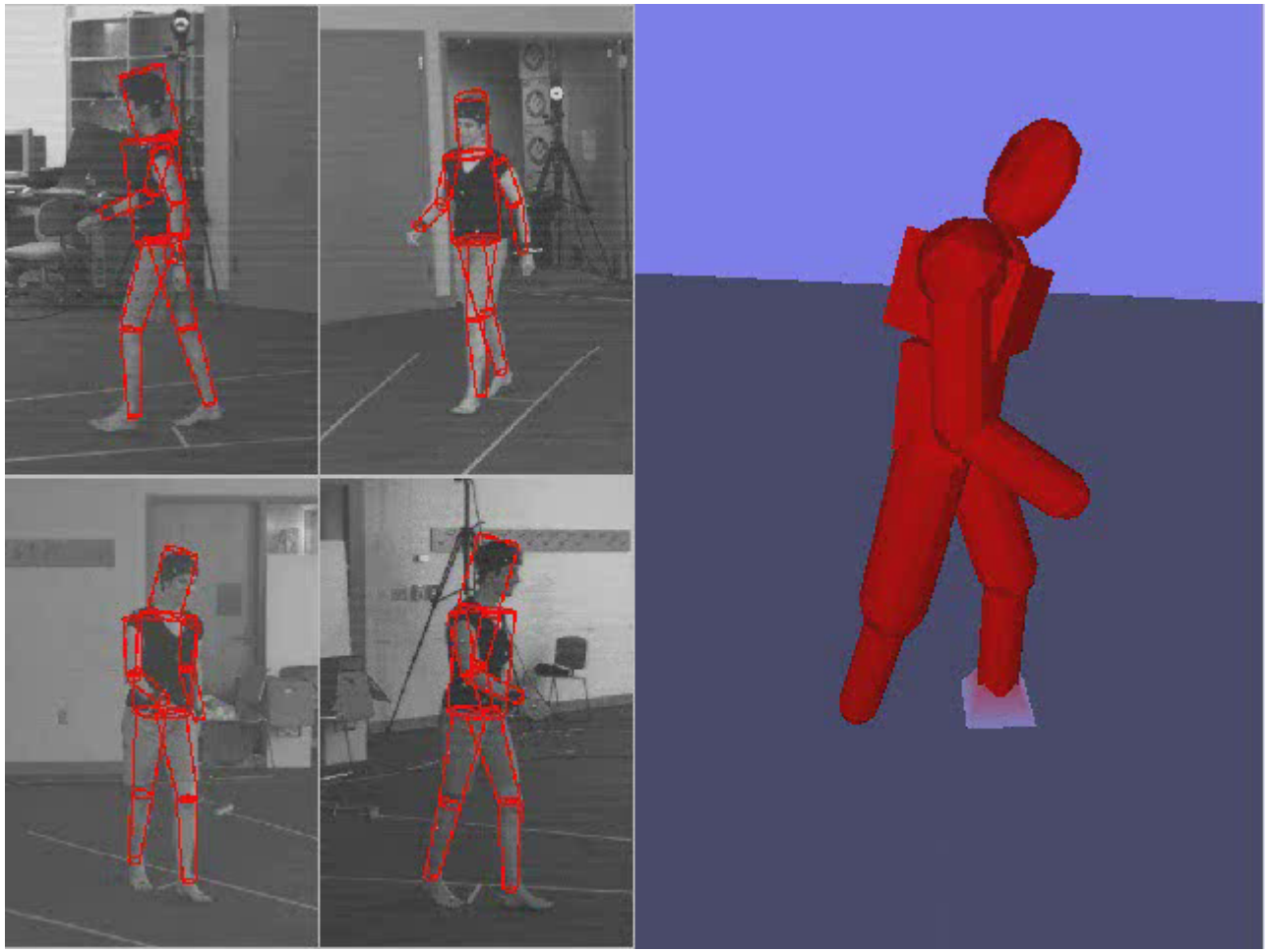
Physics (passive rag-doll):

$$\mathbf{u}_{t+1} = \int_{\Delta t} F_{eom}\left(\mathbf{u}_t, \dot{\mathbf{u}}_t, ;\right)$$

Physics (active mocap control):

$$\mathbf{u}_{t+1} = \int_{\Delta t} F_{eom}\left(\mathbf{u}_t, \dot{\mathbf{u}}_t, KernReg\left(\mathbf{u}_t, \dot{\mathbf{u}}_t\right)\right)$$

Multi-view Tracking

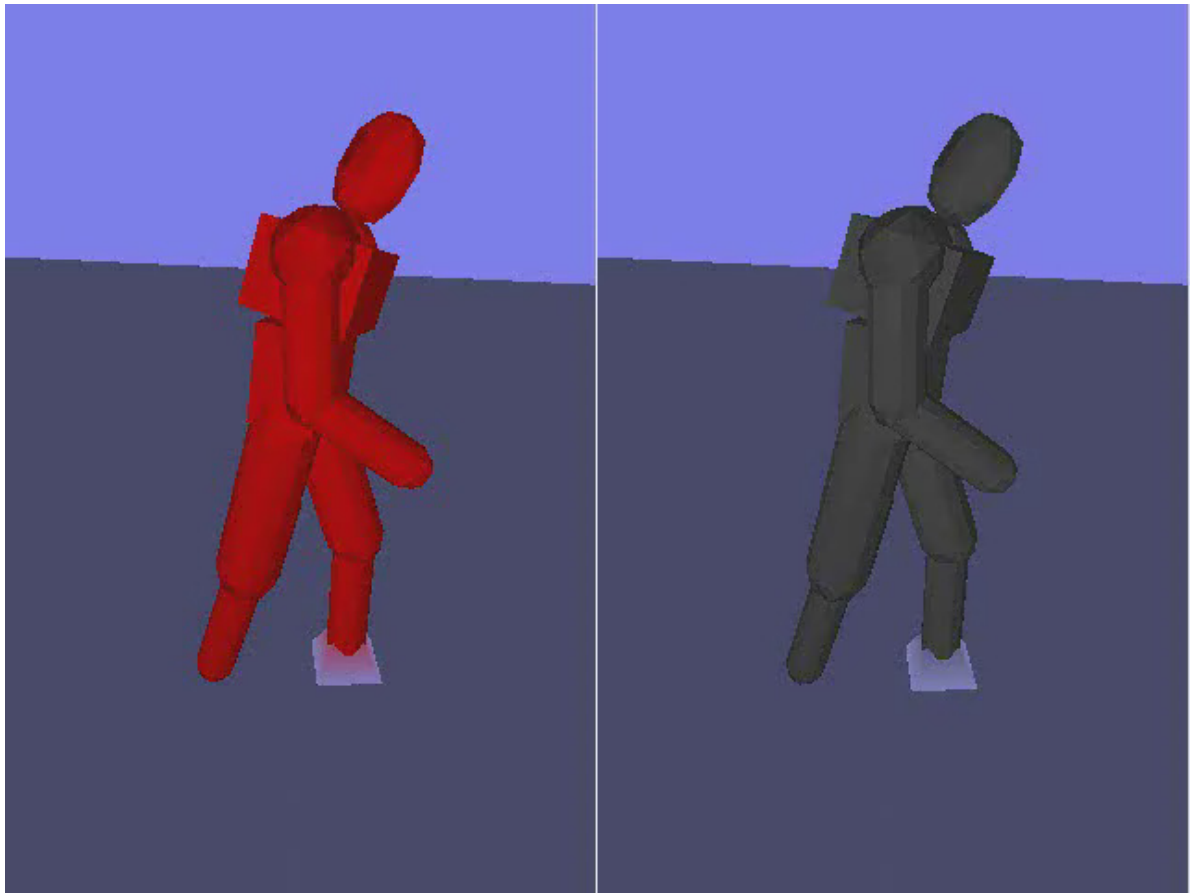


Purple square:
Ground contact

Inference: Particle Filtering
Motion Prior: Physics-based

Multi-view Tracking Comparison

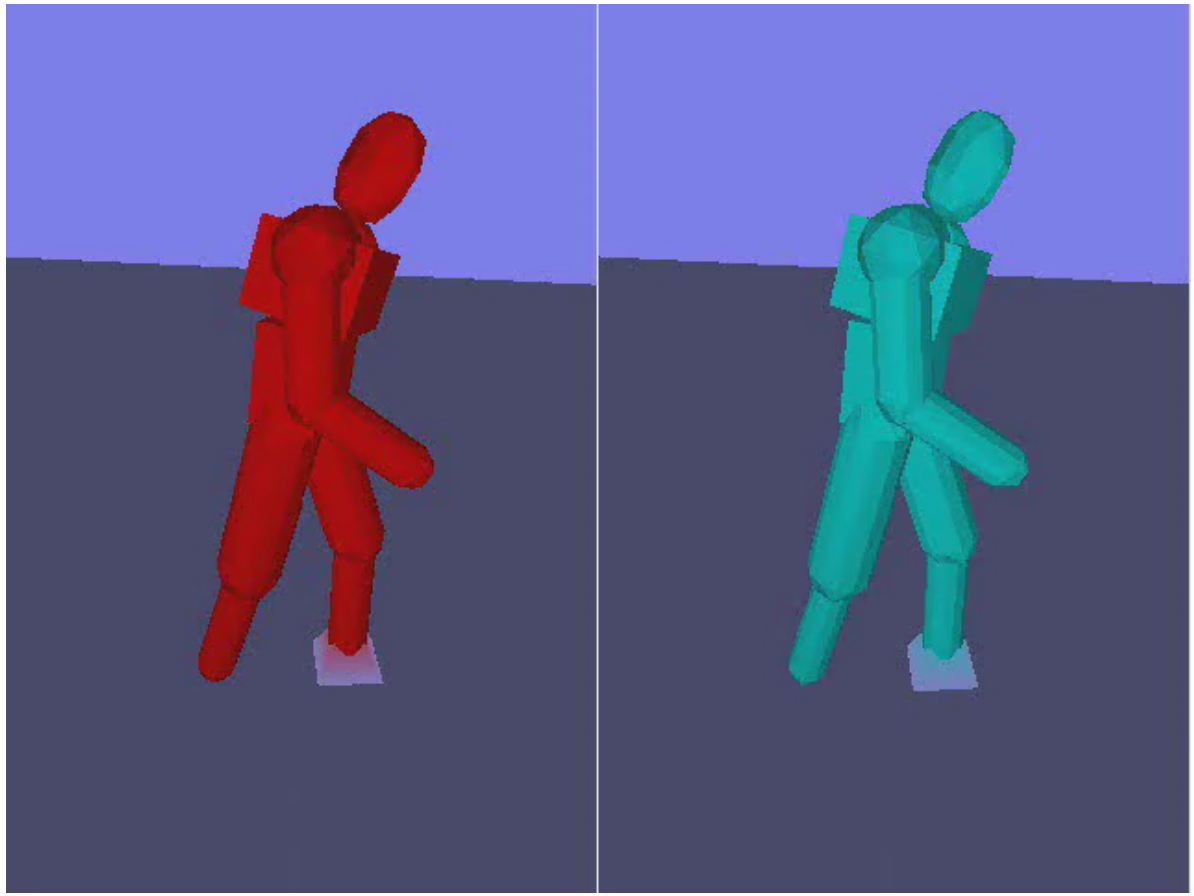
All likelihoods and parameters are comparable



Inference:	Particle Filtering	Particle Filtering
Motion Prior:	Physics-based	Smooth prior

Multi-view Tracking Comparison

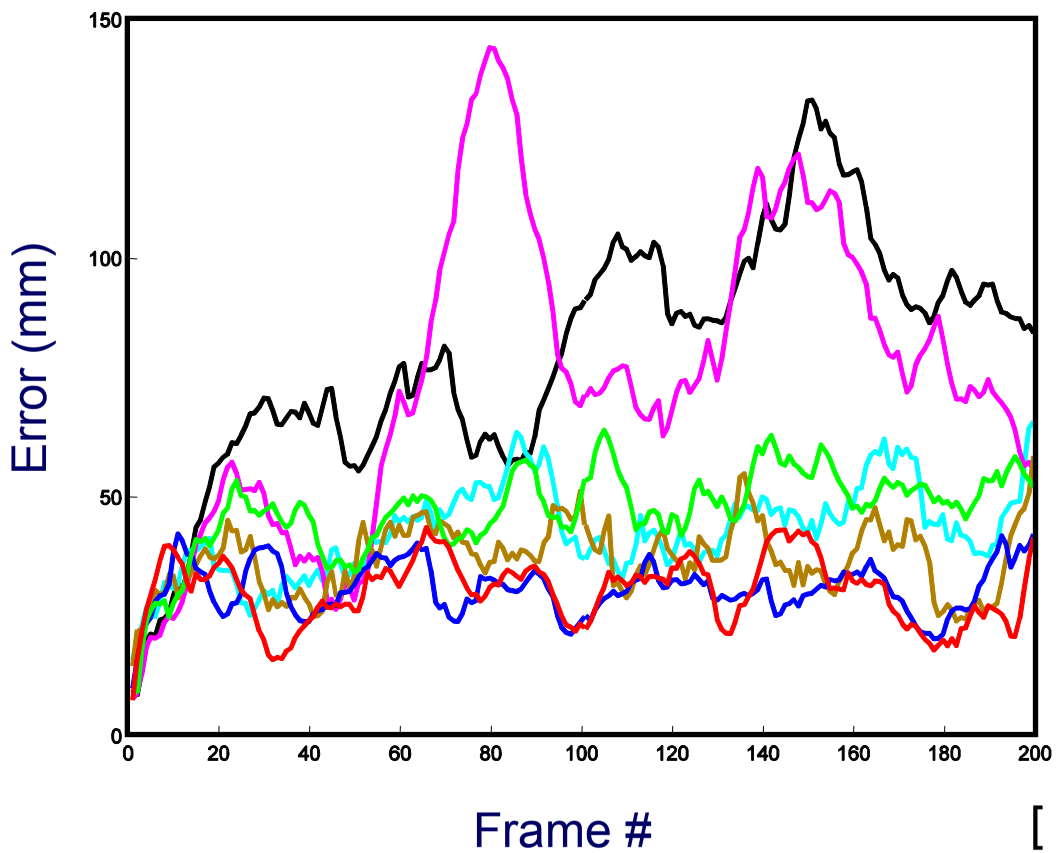
All likelihoods and parameters are comparable



Inference:	Particle Filtering	Annealed Particle Filtering
Motion Prior:	Physics-based	Smooth prior

Multi-view Tracking Quantitative Performance

PF (smooth)	APF 5 (smooth)	PF (MoCap)	APF 5 (MoCap)	PF Physics
80.0 **± 24.8**	42.8 **± 9.2**	**74.9 ± 31.2**	37.0 **± 7.1**	30.3 **± 6.8**

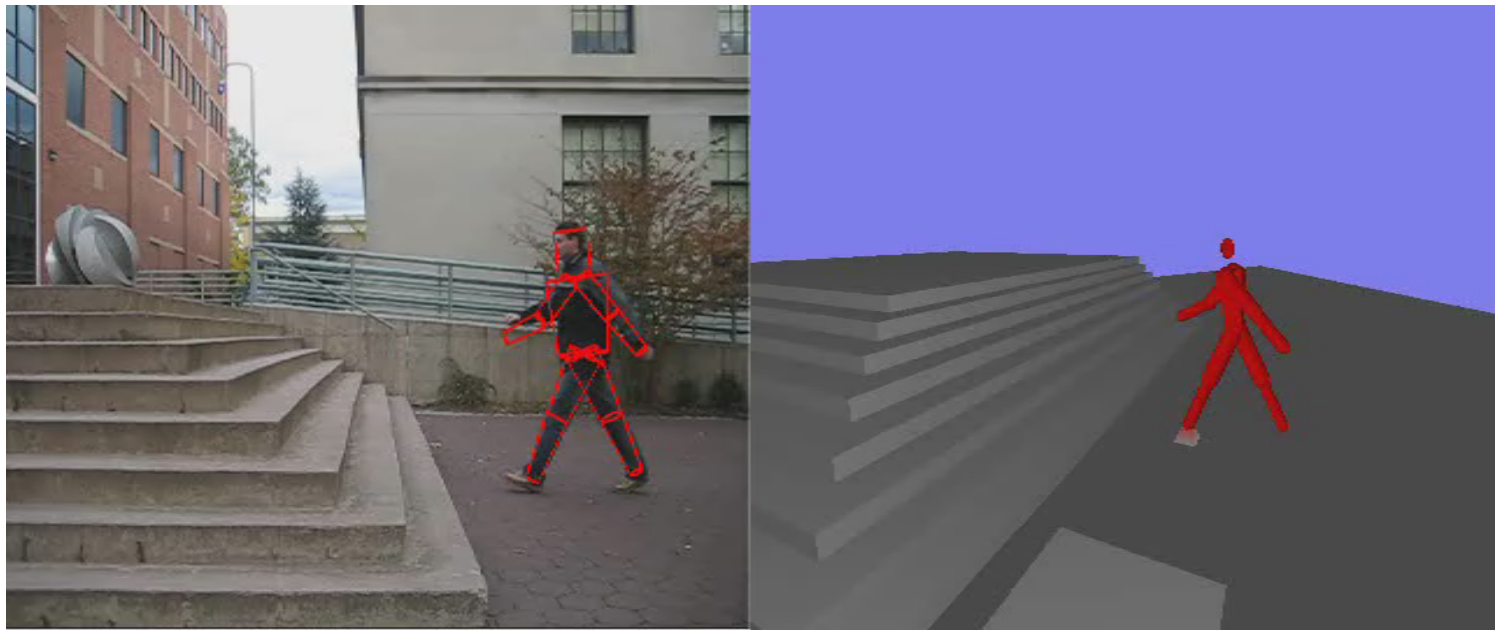


More accurate performance with lower variance

[*Vondrak, Sigal, Jenkins, CVPR'08*]

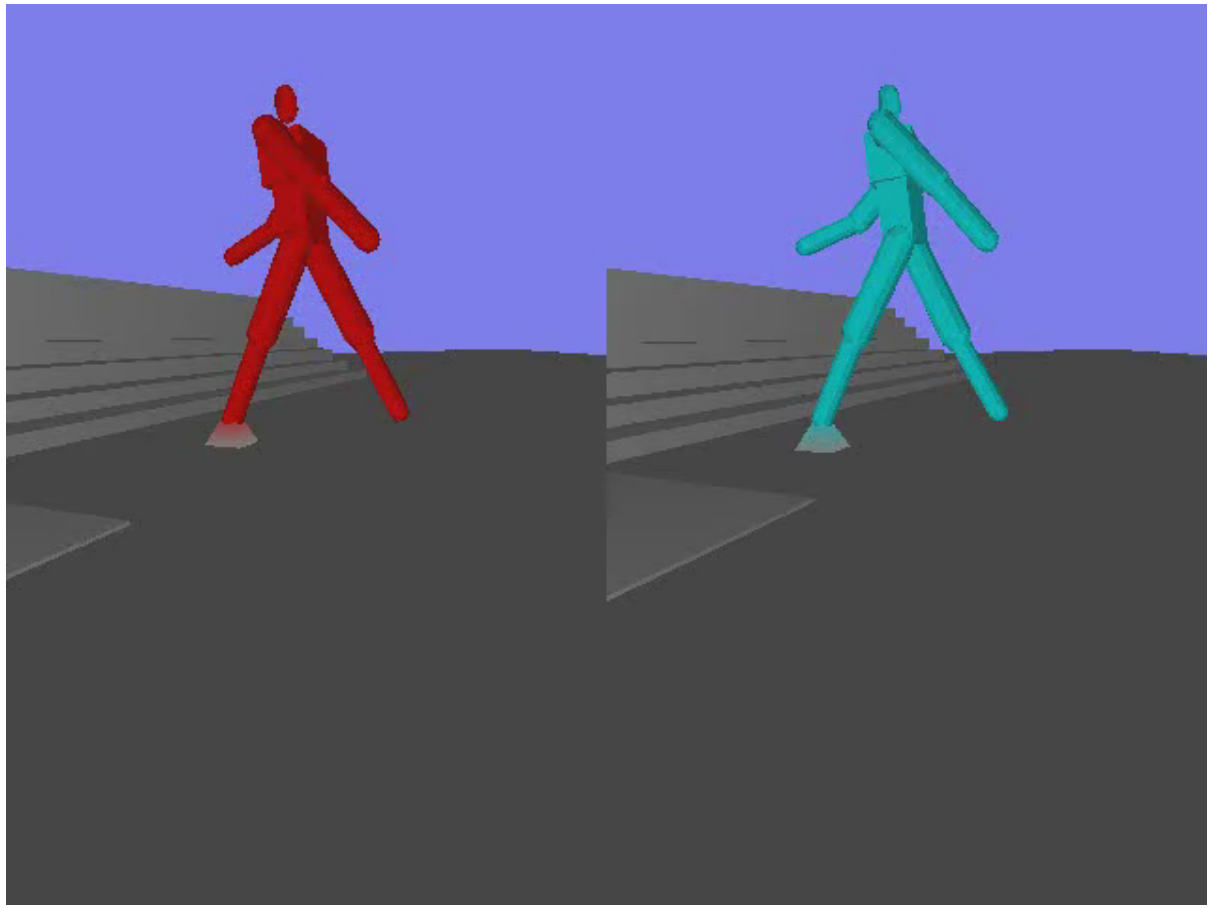
Monocular Tracking Generalization

Learn from walking on level-ground, track walking up the stairs



Monocular Tracking Generalization

All likelihoods and parameters are comparable



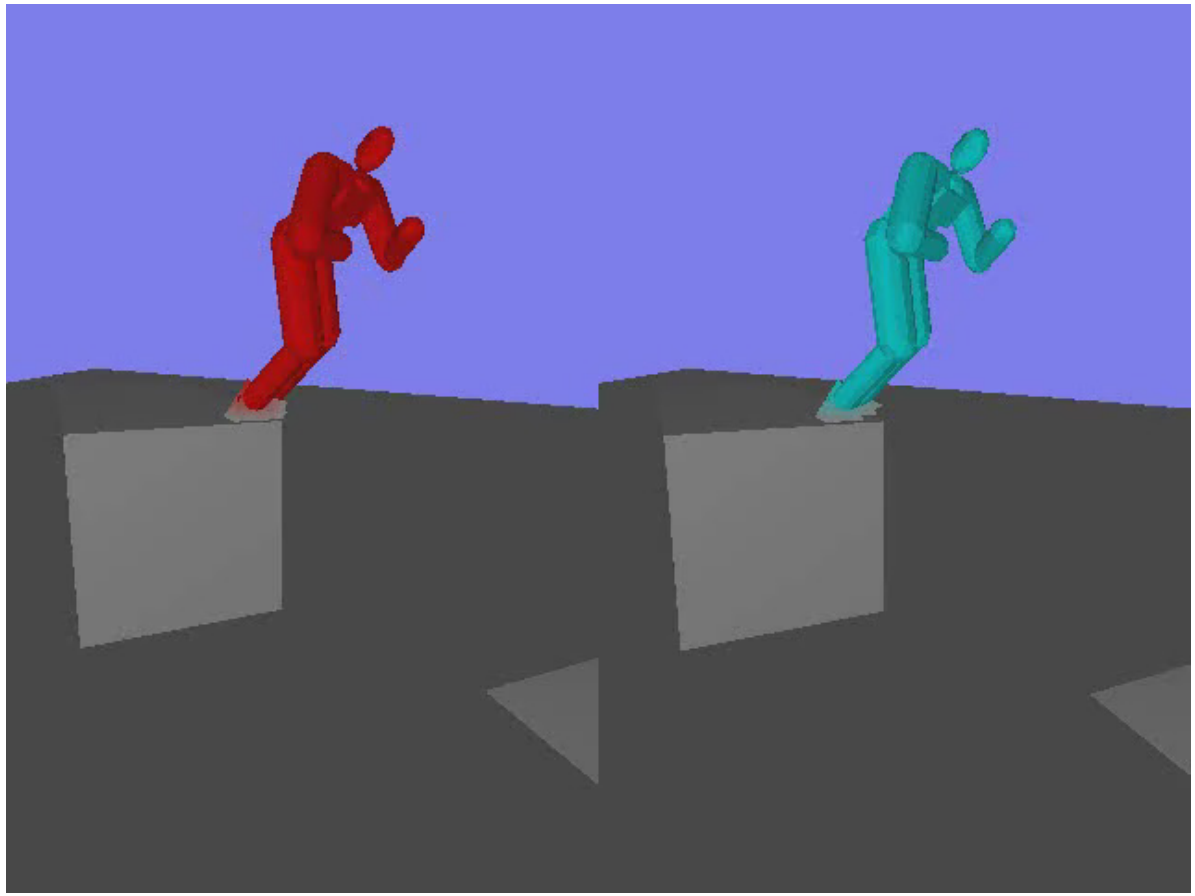
Inference:	Particle Filtering	Annealed Particle Filtering
Motion Prior:	Physics-based	Smooth prior

Monocular Tracking Effect of Physics

[*Vondrak, Sigal, Jenkins, CVPR'08*]

Monocular Tracking Effect of Physics

All likelihoods and parameters are comparable



Inference:	Particle Filtering	Annealed Particle Filtering
Motion Prior:	Physics-based	Smooth prior

Lessons Learned

- Trajectory control prior along with dynamical simulation and knowledge of environment is indeed an effective paradigm
- This type of prior model is appealing because it serves as a black box that adds physical realism to any traditional kinematic prior model (that can generate desired joint angle trajectories)
- In the form illustrated, relies heavily on motion capture data and has limited potential to generalize beyond training motions
- Search in the high-dimensional space is challenging, requires only approximate maintenance of physical realism (some noise added to the kinematic poses)

On-line Inference vs. Batch

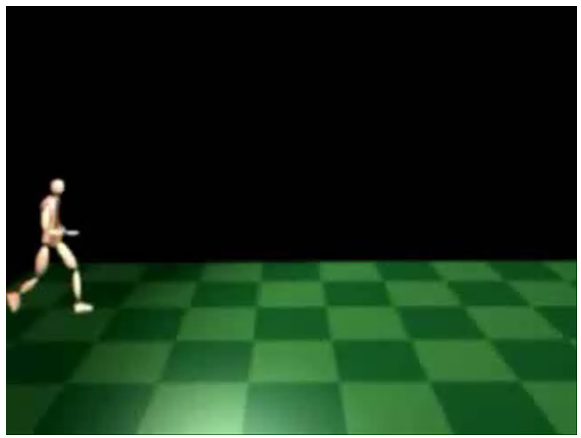
- So far we looked at controllers that are able to generate required forces to actuate the body in an on-line fashion
- This was intentional since for most tracking applications in vision this is what we need
- However, there are other ways to *synthesize* physically realistic motions

Trajectory or Space-time Optimization

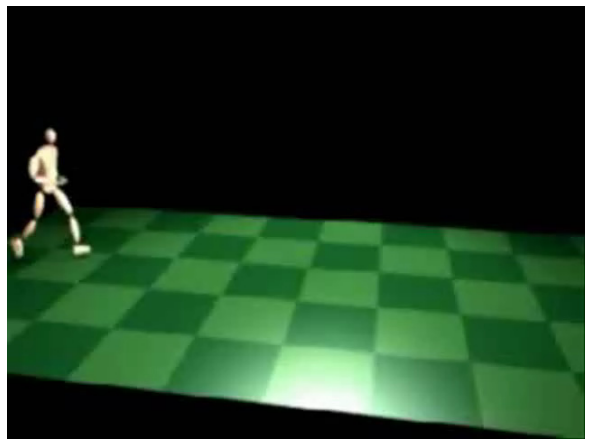
[*Witkin and Kass, Siggraph 1988*]

- An alternative to designing a controller that generates actuation of joints is optimization (space-time optimization)

Running in the lab



Running on Neptune
(10x gravity)

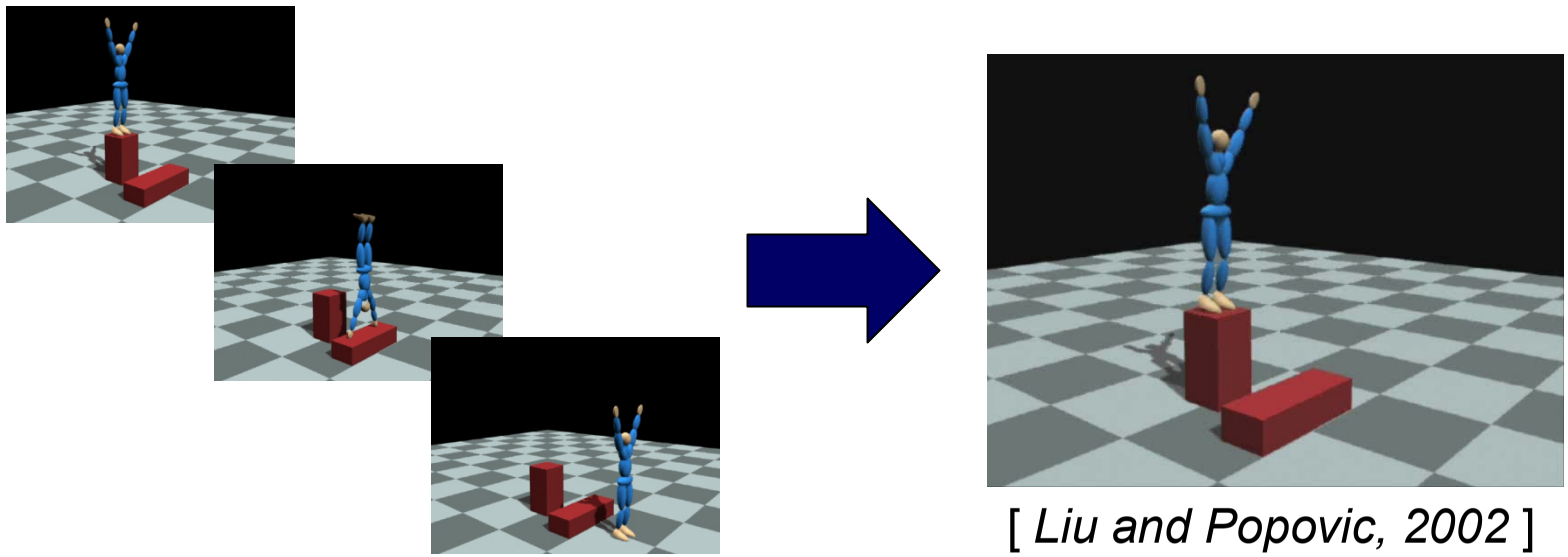


[*Popovic and Witken, 1999*]

- **Motion Adaptation**: Given a captured kinematic motion optimize how it would look if it was captured in different environment or under different physical conditions

Trajectory or Space-time Optimization

- An alternative to designing a controller that generates actuation of joints is optimization (space-time optimization)



[*Liu and Popovic, 2002*]

- **Animation**: Given sparse set of constraints (full or partial key poses) optimize the kinematic motion such that it realistic (as physically plausible as possible)

Space-Time Optimization

- Constrained optimization over the forces and the resulting motion

- Constraints encode:
  - Dynamics and physical realism
  - Kinematic objectives on motion

- Objective:
  - Prior on the motion (*e.g.*, minimize energy, or force)

$$\arg \min_{\mathbf{u}(t),\mathbf{f}(t)} E(\mathbf{u}(t),\mathbf{f}(t),t) \quad \textit{subject to} \begin{cases} C_d(\mathbf{u}(t),\mathbf{f}(t),t) = 0 \\ C_p(\mathbf{u}(t),\mathbf{f}(t),t) = 0 \\ C_m(\mathbf{u}(t),\mathbf{f}(t),t) = 0 \end{cases}$$

Space-Time Optimization

- Specify equations of motion for the system

$$C_d(\mathbf{u}_t, \mathbf{f}_t, t) = F_{eom}\left(\mathbf{u}_t, \frac{\mathbf{u}_{t+1} - \mathbf{u}_{t-1}}{2\Delta t}, \mathbf{f}_t\right) \text{¡} \quad \frac{\mathbf{u}_{t+1} - 2\mathbf{u}_t + \mathbf{u}_{t-1}}{(\Delta t)^2} = 0$$

- Specify spatio-temporal constraints (key frames)

$$C_p(\mathbf{u}_t, \mathbf{f}_t, t) = 0$$

e.g. boundary conditions

$$(\mathbf{q}_0 \text{¡} \quad \mathbf{q}_d(0))^2 = 0$$
$$(\mathbf{q}_T \text{¡} \quad \mathbf{q}_d(T))^2 = 0$$

- Specify environment (mechanical) constraints

$$C_m(\mathbf{u}_t, \mathbf{f}_t, t) = 0$$

e.g. gravity, contact

- Specify objective function (typically minimize energy)

$$E(\mathbf{u}_t, \mathbf{f}_t, t) = \sum_{t=0}^{T} \text{jj}\mathbf{f}_t\text{jj}_2$$

$$\arg \min_{\mathbf{u}(t), \mathbf{f}(t)} E(\mathbf{u}(t), \mathbf{f}(t), t) \quad \textit{subject to} \begin{cases} C_d(\mathbf{u}(t), \mathbf{f}(t), t) = 0 \\ C_p(\mathbf{u}(t), \mathbf{f}(t), t) = 0 \\ C_m(\mathbf{u}(t), \mathbf{f}(t), t) = 0 \end{cases}$$