Data-Driven Computer Animation

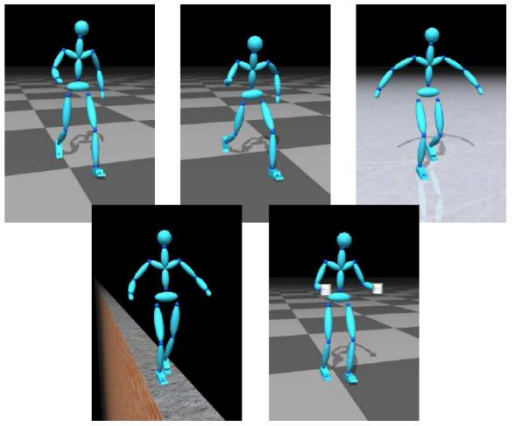
Lecture 5.

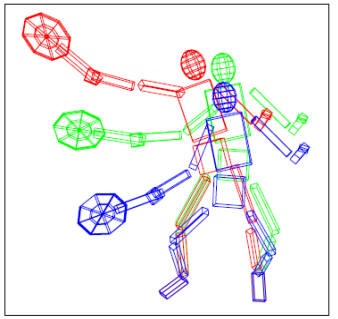
Motion Synthesis by Optimization / Motion Editing

Taku Komura

Overview

- **Motion synthesis by Optimization**
  - **Using forward / inverse dynamics**
- Motion Editing
  - Motion Warping
  - Motion Blending
  - Dynamic Time Warping
- Motion databases

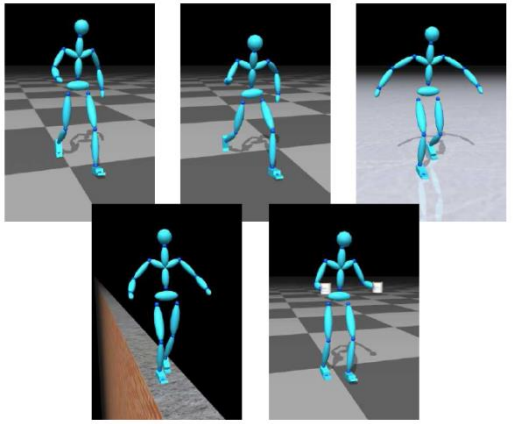




Motion Synthesis by Optimization

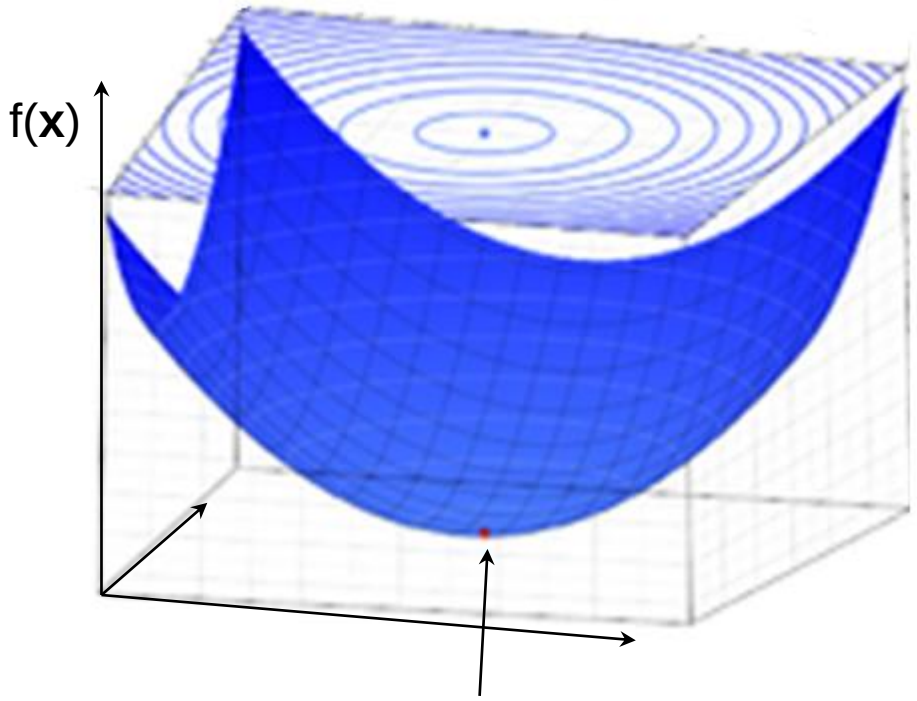
Why?

- We want the character to stably walk in a different condition (forward dynamics)
- We want characters to not spend too much energy (inverse dynamics)
  - Humans move efficiently
- We also want them to satisfy constraints
  - passing some specific postures, feet on the ground etc.



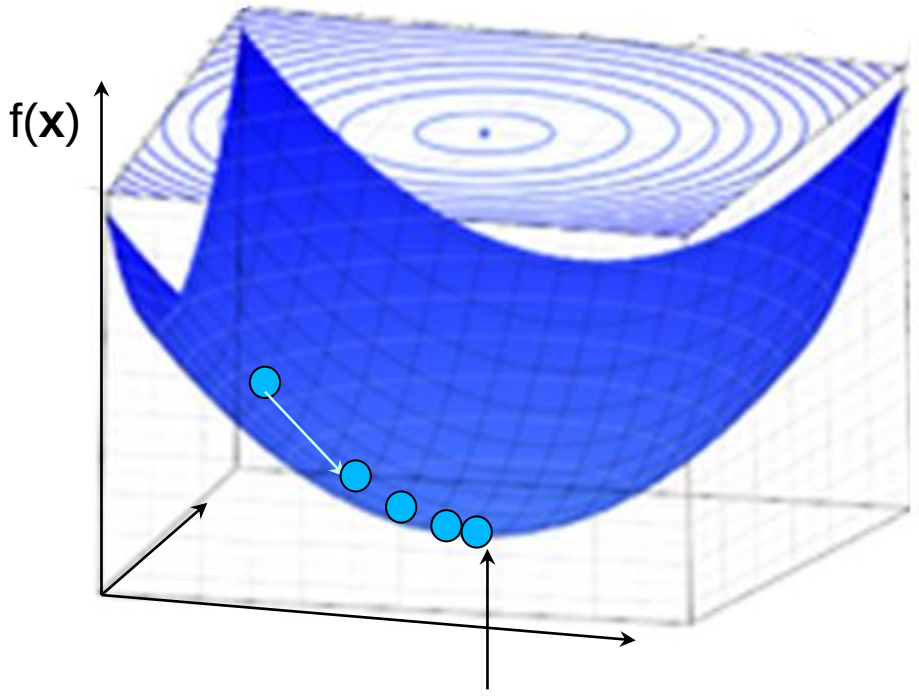
What is Optimization?

- Given an objective function f(**x**), find the **x** that minimizes f(**x**)



What is Optimization?

- Solution is found by iteratively computing the gradient at the current point, and moving in that direction



What are the parameters?

- Given some function f(**x**), find the **x** that minimizes f(**x**)
  - Subject to constraints g(**x**) = **0**

parameters

- Forward dynamics : Torque at the joints, or parameters of PD control (target postures, gains)
- Inverse dynamics : Trajectory of joint angles (or keyframes)

$$\boldsymbol{\tau} = \mathbf{H}\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_g$$

Torque at the joints

motion

$\boldsymbol{\tau}_1$ $\boldsymbol{\tau}_2$ …… $\boldsymbol{\tau}_n$

$$\boldsymbol{\tau} = \mathbf{H}\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_g$$

$\mathbf{q}_1$ $\mathbf{q}_2$ …… $\mathbf{q}_n$

$$\boldsymbol{\tau} = \mathbf{H}\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_g$$

Difficulty 1: High Dimensionality

- The body has many degrees of freedom
- There are many frames in one motion :
- Even we simplify the control (reducing the number of keyframes), the problem can still be very high dimensional

- i.e. motion

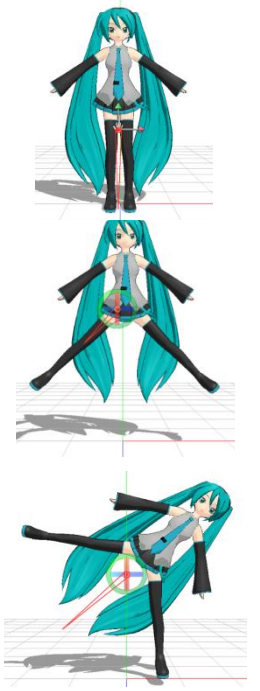
$$\mathbf{q}_1 = (q_1, q_2, q_3, q_4, q_5, q_6, q_7, \ldots, q_m)$$
$$\mathbf{q}_2 = (q_1, q_2, q_3, q_4, q_5, q_6, q_7, \ldots, q_m)$$
$$\vdots$$
$$\mathbf{q}_n = (q_1, q_2, q_3, q_4, q_5, q_6, q_7, \ldots, q_m)$$

- m x n parameters

torque

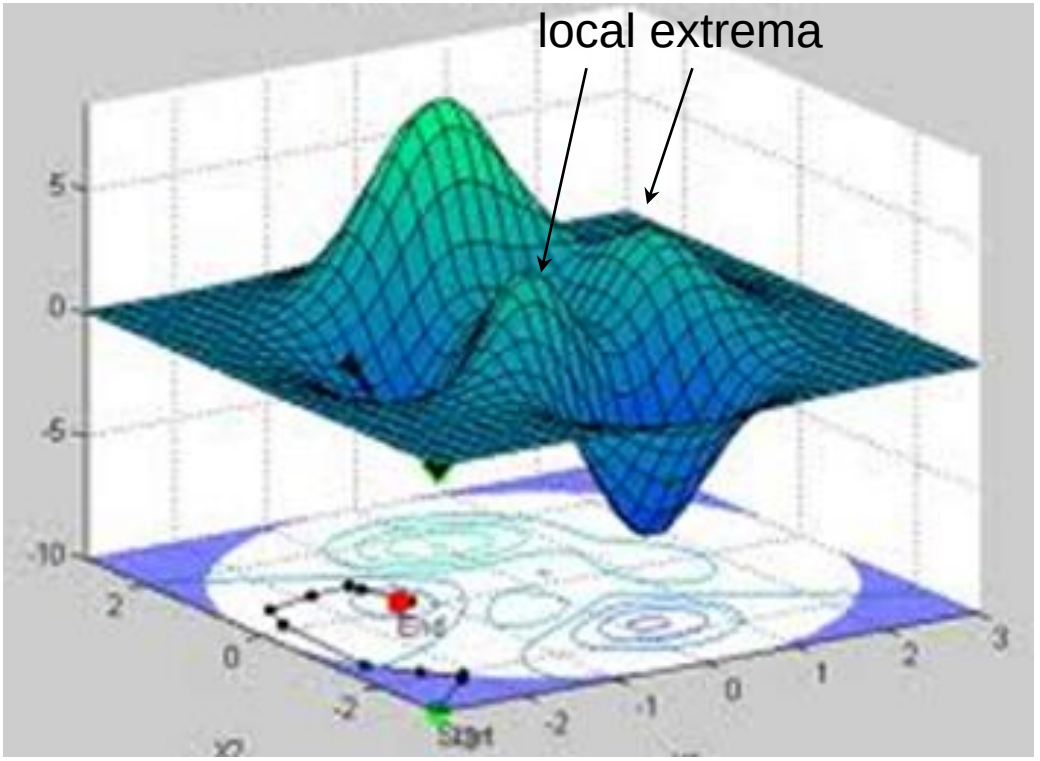
$$F_1 = (\tau_1, \tau_2, \tau_3, \ldots, \tau_{m-6})$$
$$F_2 = (\tau_1, \tau_2, \tau_3, \ldots, \tau_{m-6})$$
$$\vdots$$
$$F_n = (\tau_1, \tau_2, \tau_3, \ldots, \tau_{m-6})$$

(m-6) x n parameters



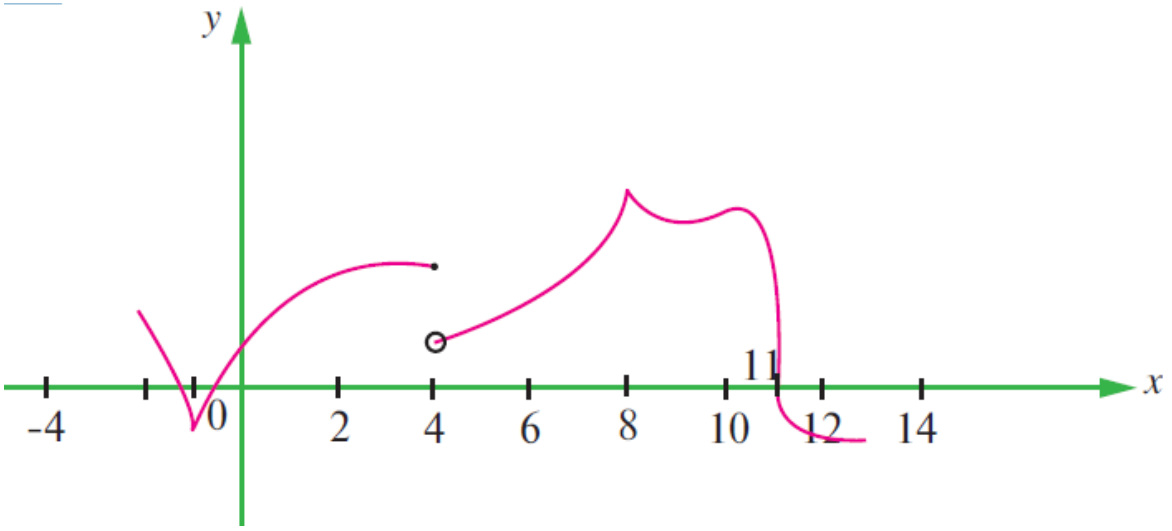
Difficulty 2: Local Extrema

- Functions with many parameters can have lots of local extrema
- An ordinary optimization method can get trapped in local extrema



Difficulty 3: Non-differentiability (forward simulation)

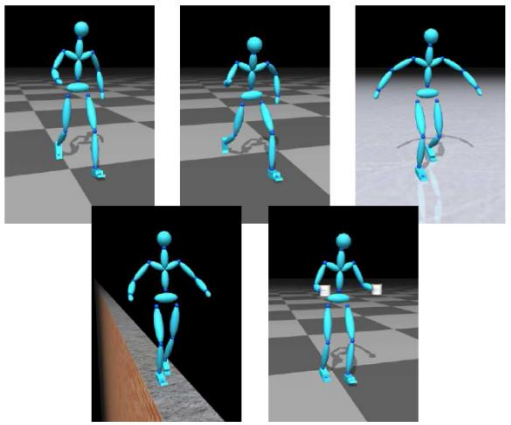
- The motion is nondifferentiable with respect to the parameters, due to the balance/collisions – collisions can suddenly happen and that makes the motion discontinuous with respect to the parameters



Optimization in Forward Dynamics

- Need to specify the objective function f(x) that evaluates the motion
  - Height of the jump
  - How stably the character can walk
  - How similar the motion is to the motion capture data

  - Parameters: PD targets, PD gains



Optimizing Walking Controllers for Uncertain Inputs and Environments

Jack M. Wang
David J. Fleet
Aaron Hertzmann

University of Toronto

What are the parameters?

- Given some function f(**x**), find the **x** that minimizes f(**x**)
  - Subject to constraints g(**x**) = **0**

parameters

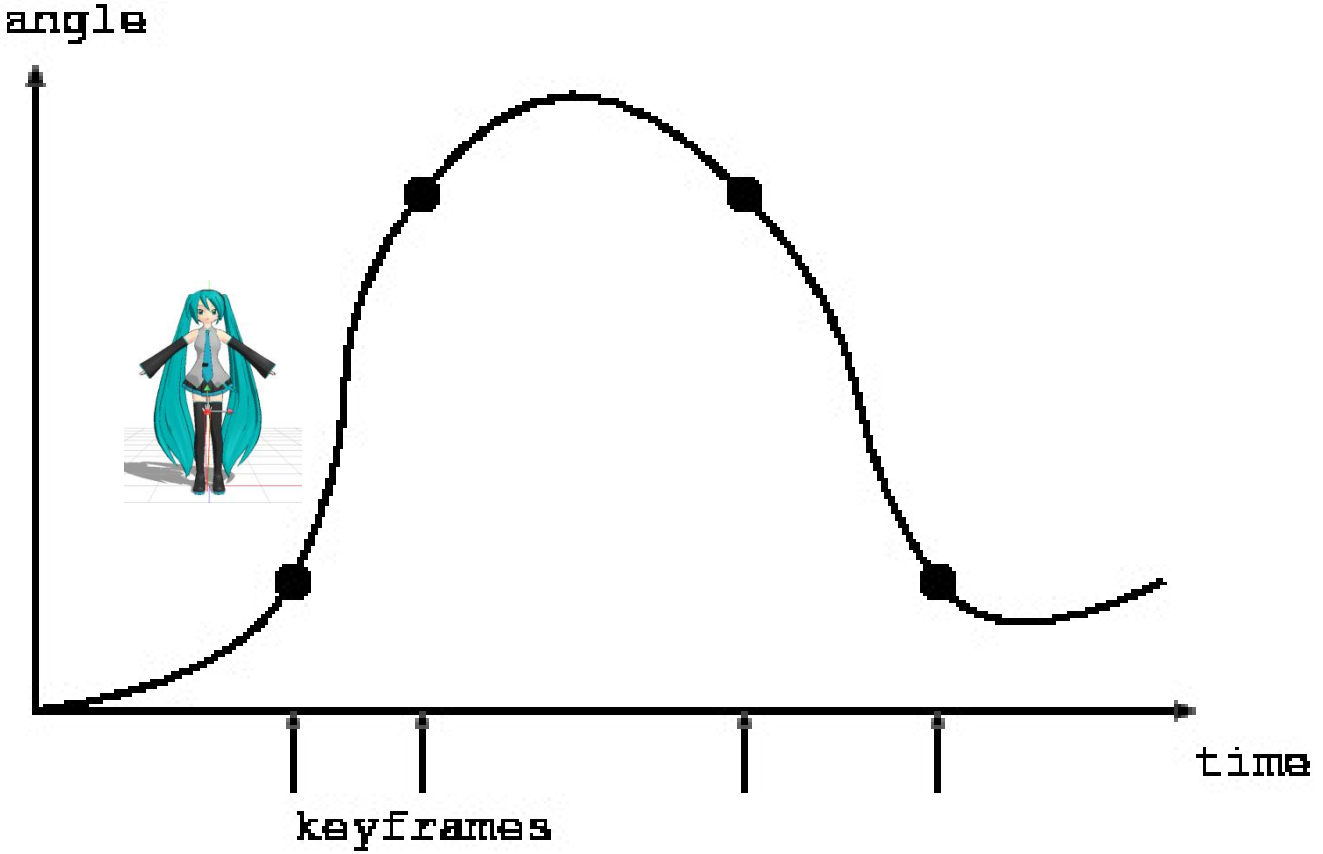
- Forward dynamics : Torque at the joints, or parameters of PD control (keyframes, coefficients)
- Inverse dynamics : Trajectory of joint angles (or keyframes)

$$\boldsymbol{\tau} = \mathbf{H}\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_g$$

Torque at the joints

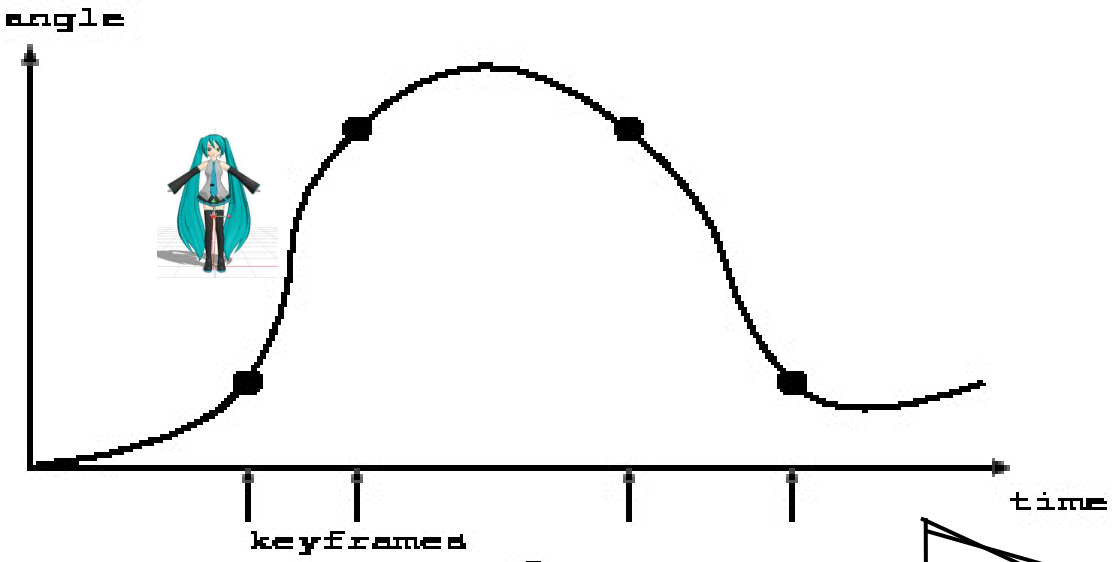
motion

Recap: Kinematic motion representation



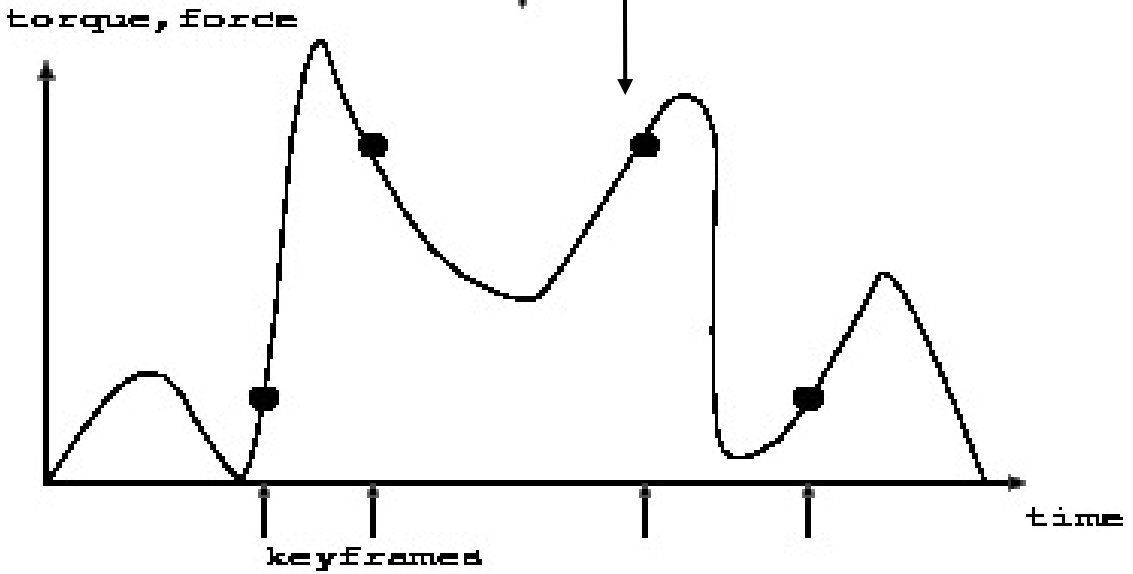
The user gives the keyframes and the motion is computed by interpolation (linear, b-splines, etc)

Keyframe Animation and Inverse Dynamics

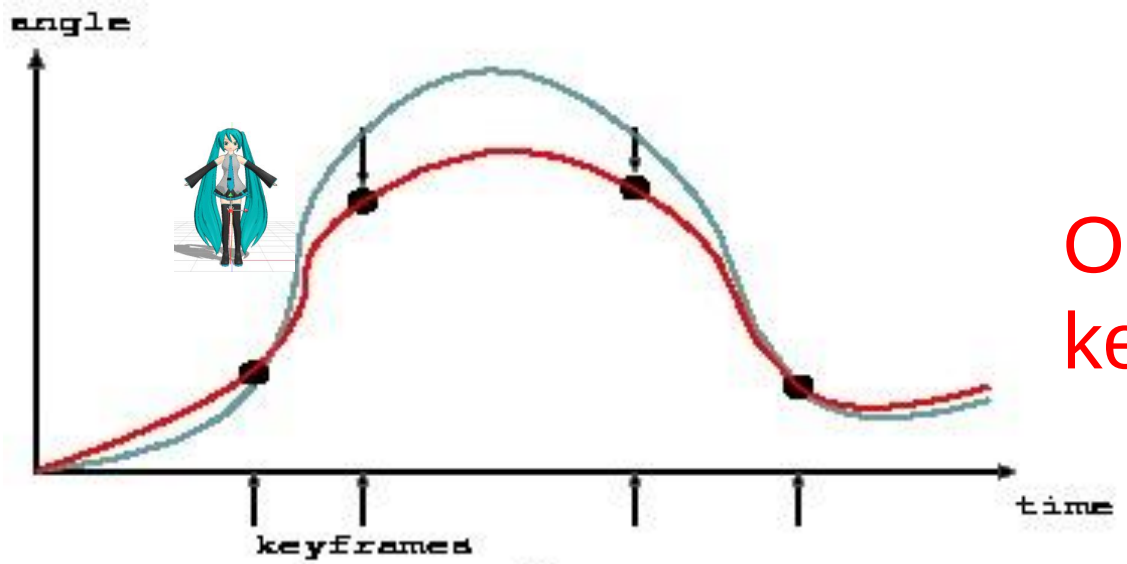


Inverse Dynamics

$$\boldsymbol{\tau} = \mathbf{H}\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_g$$

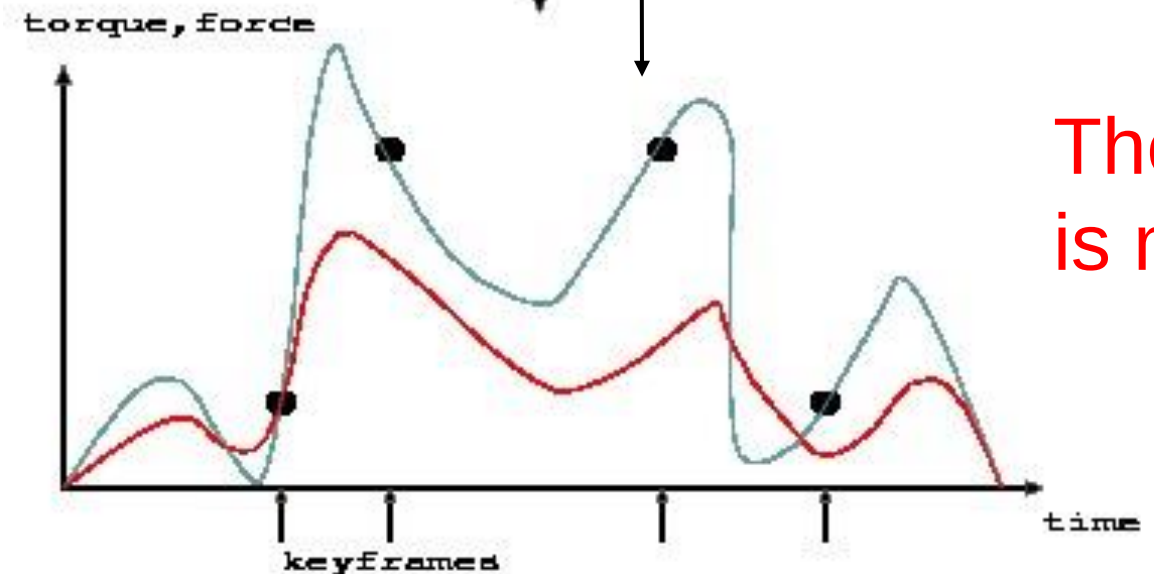


Keyframe Animation and Inverse Dynamics



Optimize the keyframes s.t.

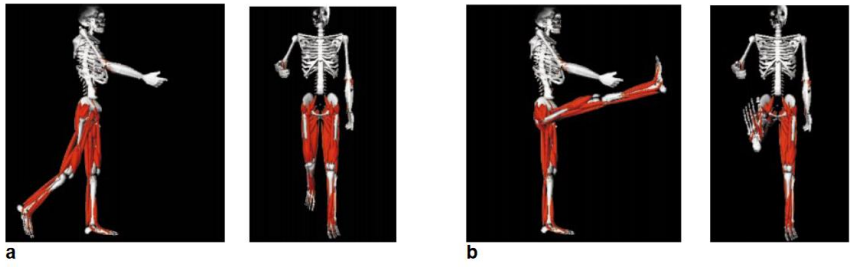
Inverse Dynamics

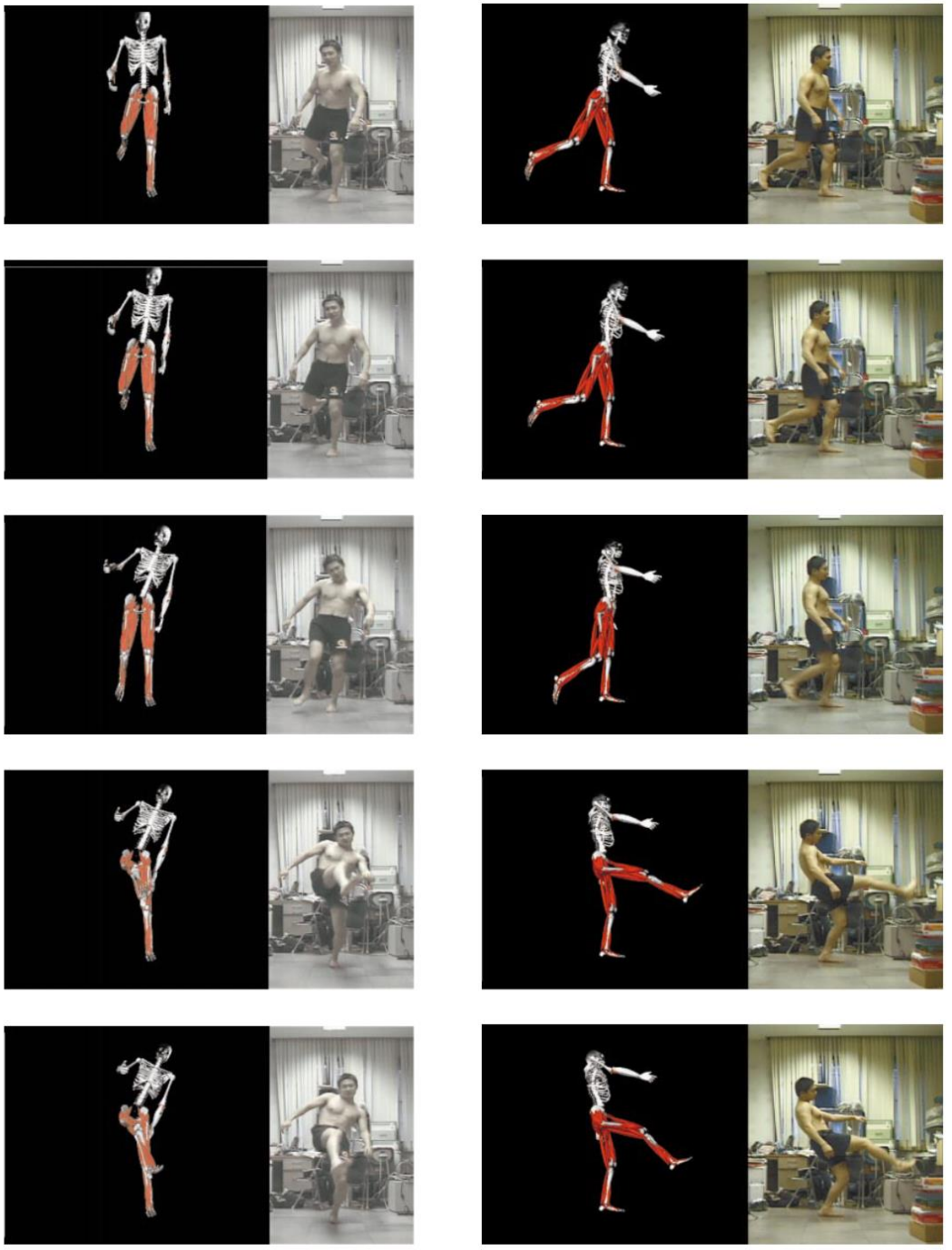


The required force is minimal

Optimization in Inverse Dynamics

- Given the first and second pose to specify the initial motion, optimize the motion such that
- balance is preserved, and
- muscles can afford the motion

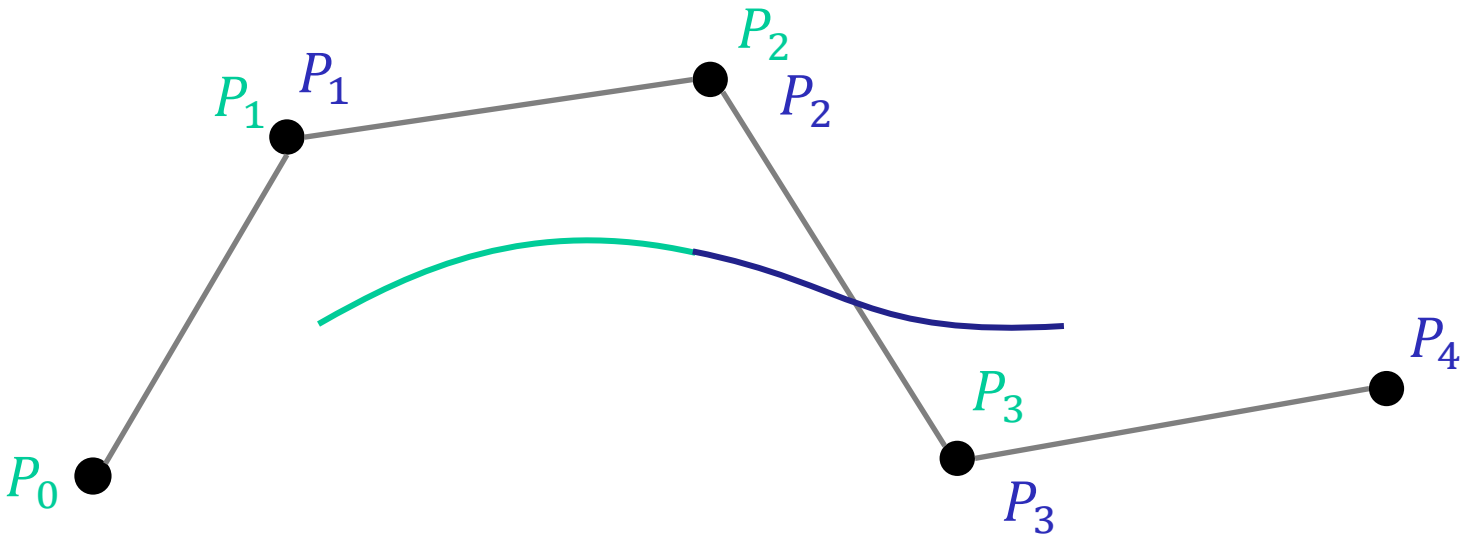




https://homepages.inf.ed.ac.uk/tkomura/VisComp.pdf

Optimization in Inverse Dynamics

- Motion represented by nonrational uniform cubic B-splines
- The knots are located uniformly
- The curve does not necessarily pass through the control points – need to either repeat the control points or add constraints



Uniform Cubic B-Splines

- The matrix form and the basis functions
- The knots specify the range of the curve

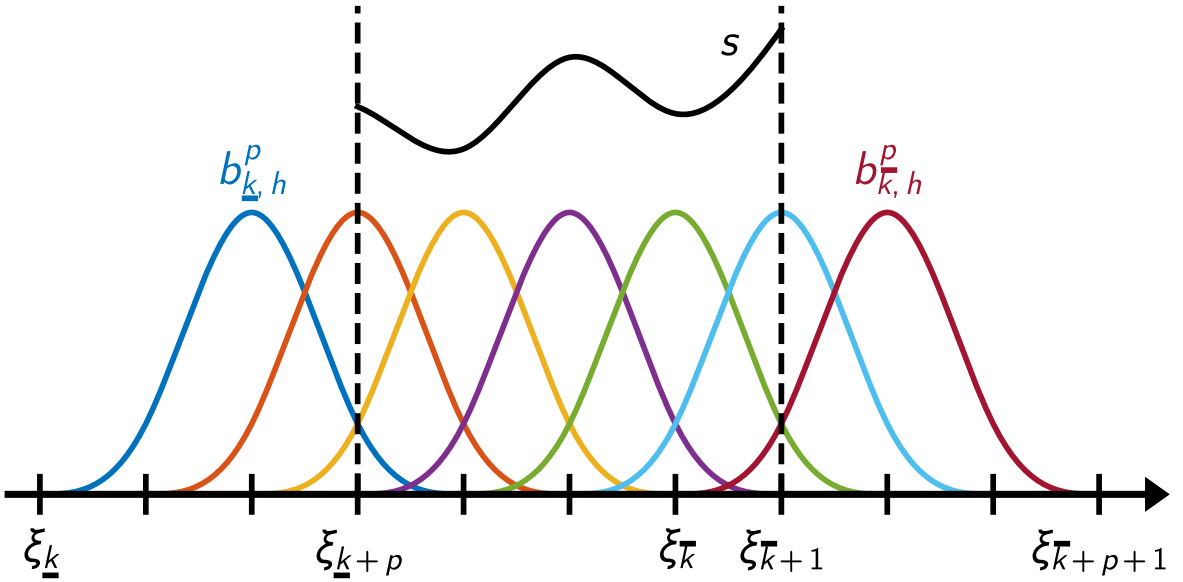
$$X(t)=\mathbf{t}^{\mathbf{T}}\mathbf{M}\mathbf{Q}^{(\mathbf{i})} \qquad for \;\; t_i \le t \le t_{i+1}$$

$$where \qquad \mathbf{Q}^{(\mathbf{i})}=(x_{i-3},\ldots,x_i)$$

$$\mathbf{M}=\frac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix},$$

$$\mathbf{t}^{\mathbf{T}}=\left((t-t_i)^3,(t-t_i)^2,t-t_i,1\right)$$

t_i : knots, $3 \le i$



Optimization in Inverse Dynamics

- Objective Function: $f(\mathbf{x}) = \|\tau\|^2 + \left\|\tau_{feet}\right\|^2$

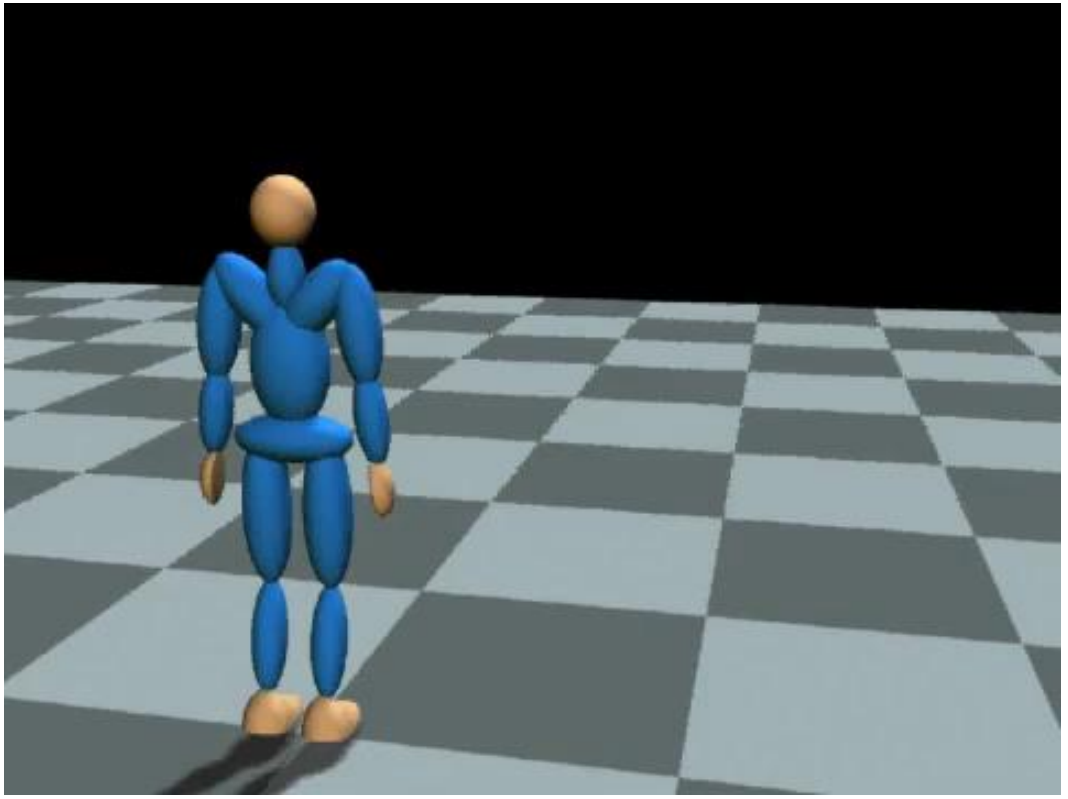
- τ : Torque produced at the joints
- **x :** The parameters of the motion – here the control points of the uniform cubic B-Splines
- τ_{feet}: torque needed to be added to the supporting foot to keep the balance

Solved by nonlinear optimization

Optimizing Motion Considering Physics (Liu and Popovic 2002)

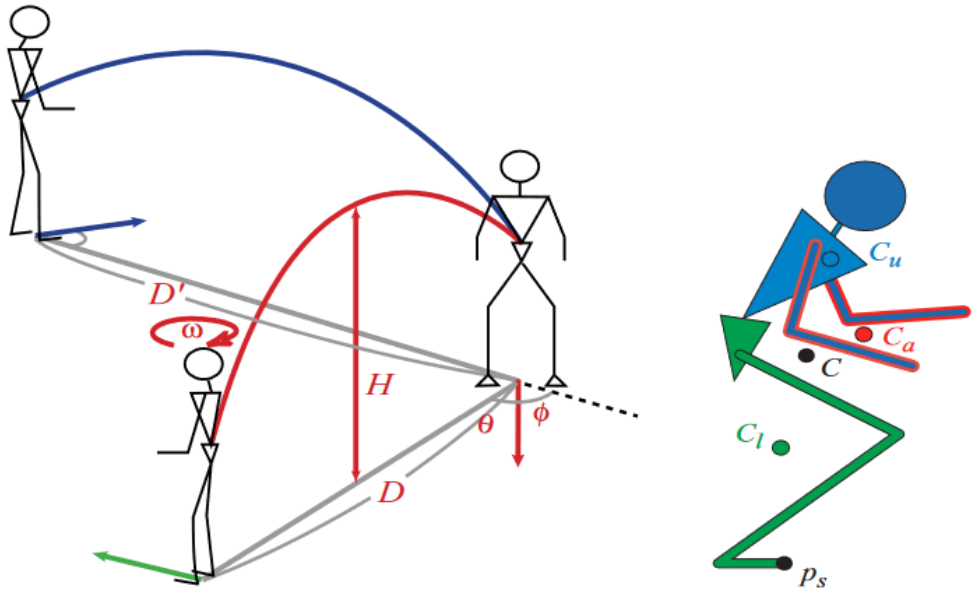
- Compute the trajectories of the body such that the body follows the user inputs and produce natural motion

  Liu and Popovic ‘02



Optimizing Motion Considering Physics (Liu and Popovic 2002)

- The motion is divided into the unconstrained stage (in the air) and constrained stage (on the ground)



Optimizing Motion Considering Physics (Liu and Popovic 2002)

Constraints

- Momentum constraint (when in the air)
- Feet position on the ground, specified by the user

Objective function

- Minimum mass displacement
- Minimal velocity of DOFs
- Static balance (when on the ground)

Optimizing Motion Considering Physics (Liu and Popovic 2002)

Momentum constraint

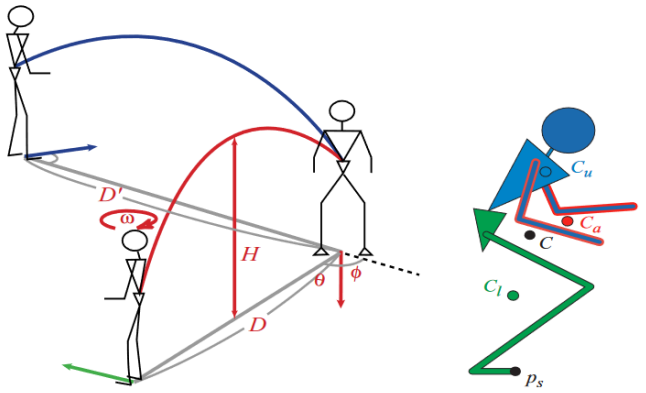
In the air :

- Linear momentum only affected by gravity

$$d\mathbf{P}(\mathbf{q})/dt = m\ddot{\mathbf{C}}(\mathbf{q}) = m\mathbf{g},$$

- Angular momentum : No changes

$$d\mathbf{L}(\mathbf{q})/dt = \mathbf{0}.$$



Optimizing Motion Considering Physics (Liu and Popovic 2002)

Objective Function: Minimum mass displacement

- We prefer to minimize the motion or energy

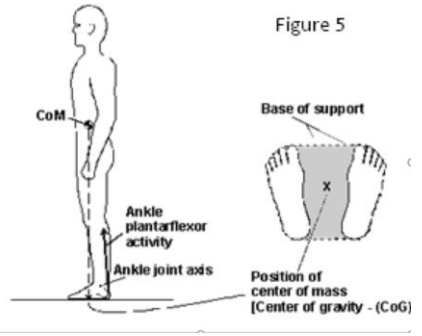
$$E_k = \iiint_i \mu_i (\mathbf{p}_i - \bar{\mathbf{p}}_i)^2 \, dx \, dy \, dz,$$

- Objective Function: Minimum joint angle motion

$$E = \int \|\dot{\mathbf{q}}\|^2 \, dt$$

- Static balance

  Distance to the base of support boundary



Discussions

- Optimization in Forward Dynamics
  - Finding a good initial motion is not easy
  - Need to keep the balance
- Optimization in Inverse Dynamics
  - Easier to start as the motion does not have to satisfy dynamic rules
  - Need a good objective function that well describes the motion
- In both cases, the cost of computation is rather high
  - Due to the large number of variables

Overview

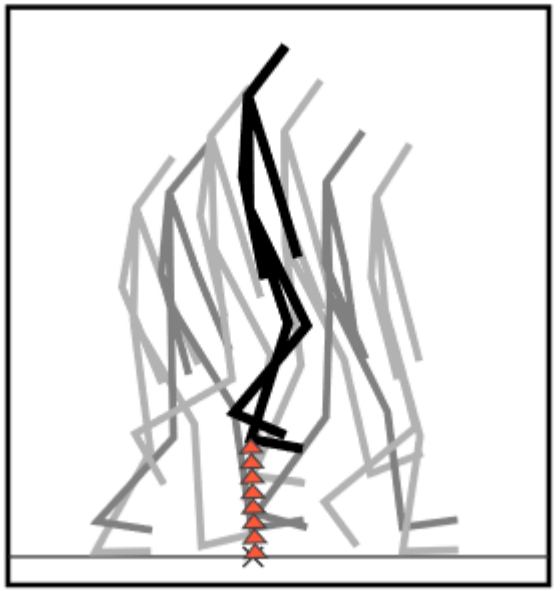
- Motion synthesis by Optimization
  - Using forward / inverse dynamics
- **Motion Editing**
  - Motion Warping
  - Motion Blending
  - Dynamic Time Warping
- Motion database

Motion Editing / Motion Blending : Background

- We learnt about motion capture
- It can provide realistic movements for characters
- However,…

Motion needs adjustments

- There might be obstacles in the scene – the character needs to avoid them

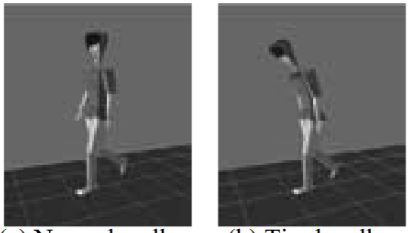


Motion needs adjustments

- There might be obstacles in the scene – the character needs to avoid them
- The character's body size is different from those of humans
  - The hand not reaching the right position
  - Need to "retarget" the motion to the character size while maintaining constraints

Motion needs adjustments

- There might be obstacles in the scene – the character needs to avoid them
- The character's body size is different from those of humans
  - The hand not reaching the right position
  - Need to "retarget" the motion to the character size while maintaining constraints
- The style of motion might not be good
  - Want to exaggerate some expressions



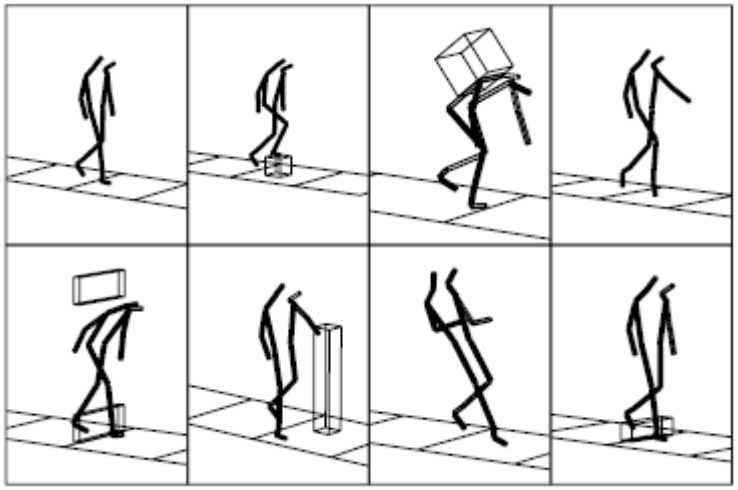
(a) Normal walk (b) Tired walk

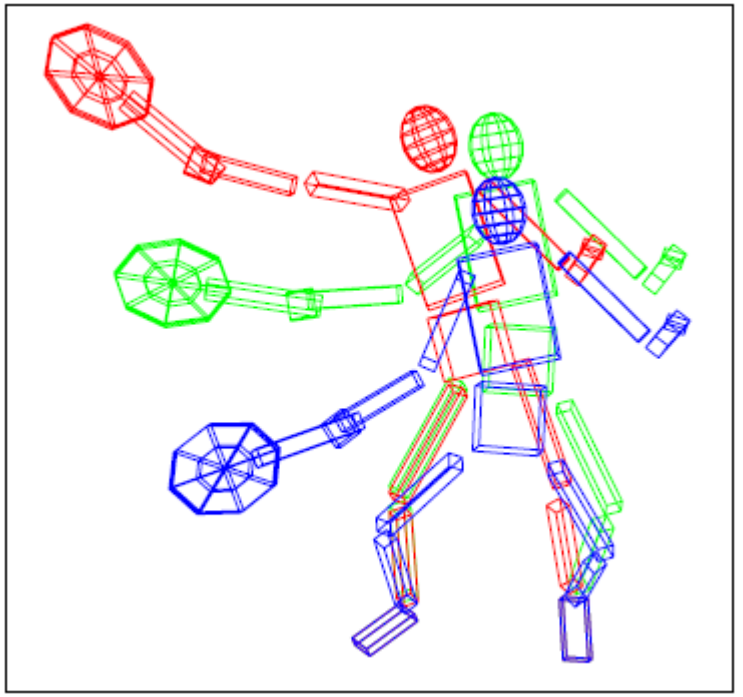
Editing motions

- Need an efficient method to edit the trajectories of the body
- Whilst satisfying constraints
- Must be interactive – the tools are used by animators

Motion Warping

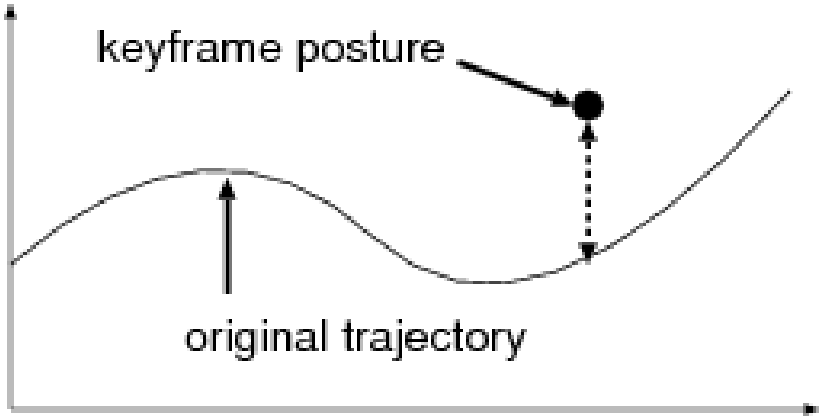
- Edit the captured motion a little bit so that it satisfies the requirements
  - Effective for changing the location the hand or the foot passes

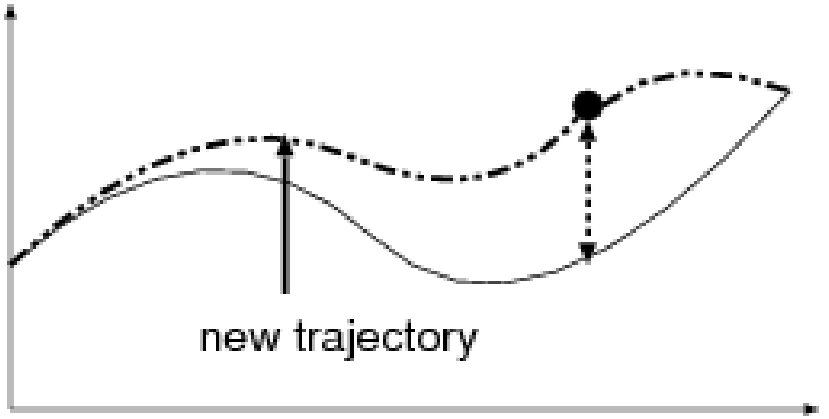




Basic idea

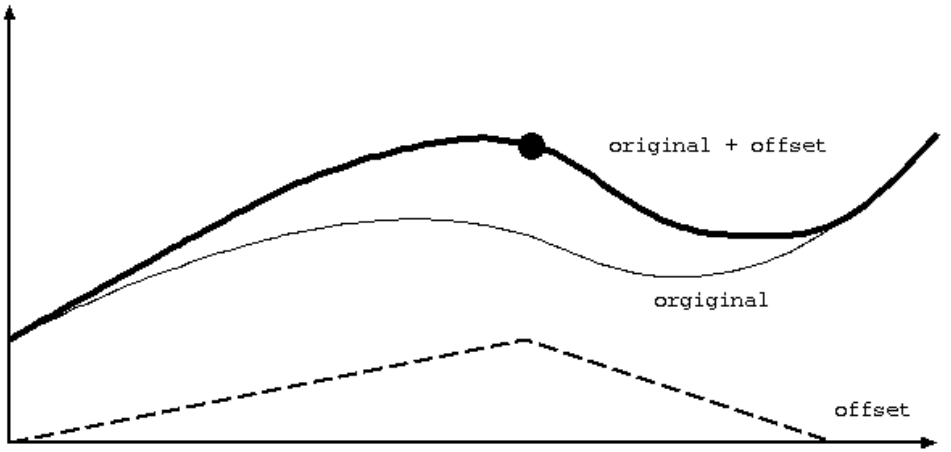
- Adding offset to the data so the constraint is satisfied





Warped Motion = original motion + offset

Offset can be a simple 1D motion

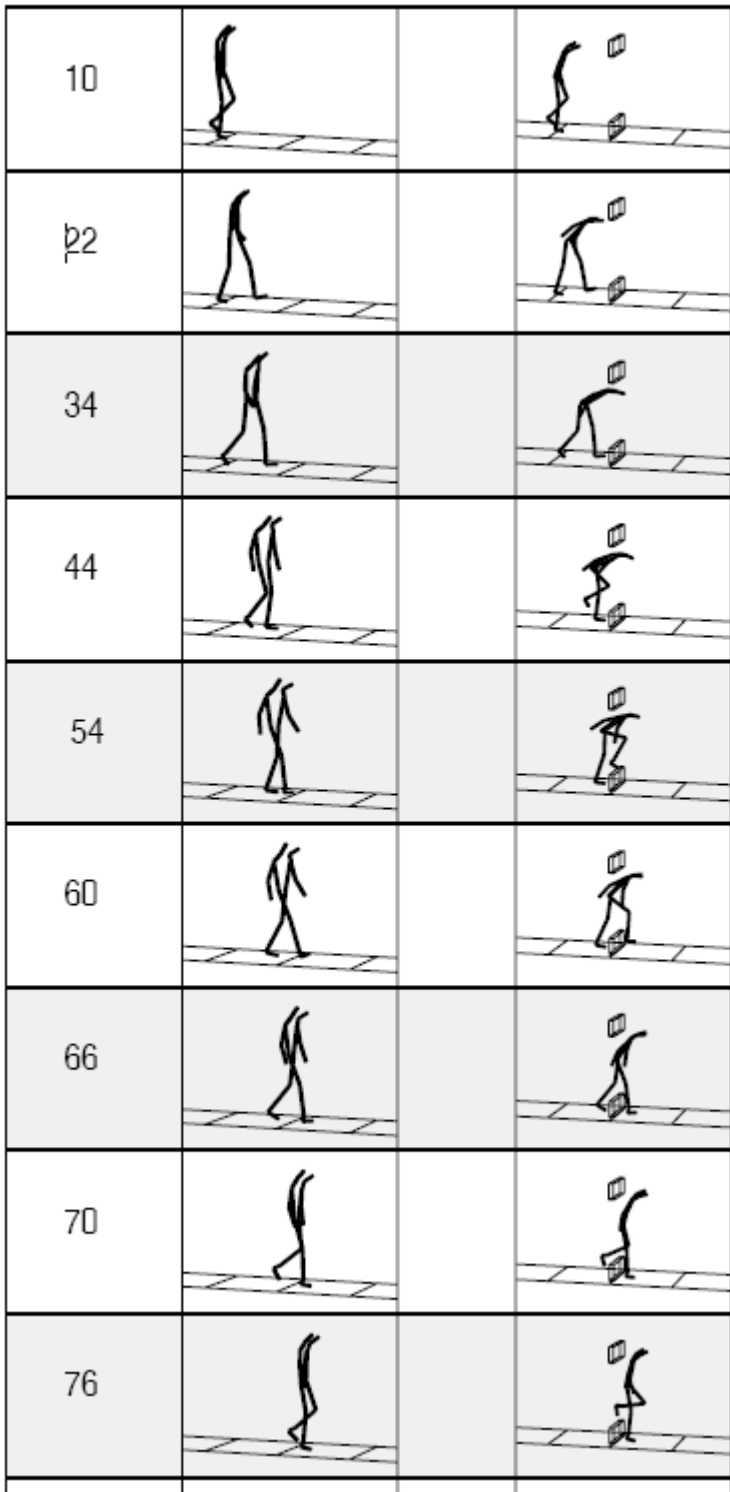


Motion Warping

- Very simple
- Good for avoiding obstacles

http://www.youtube.com/watch?v=BzfxGwO7swg

http://www.youtube.com/watch?v=Ro30uWsWl_s

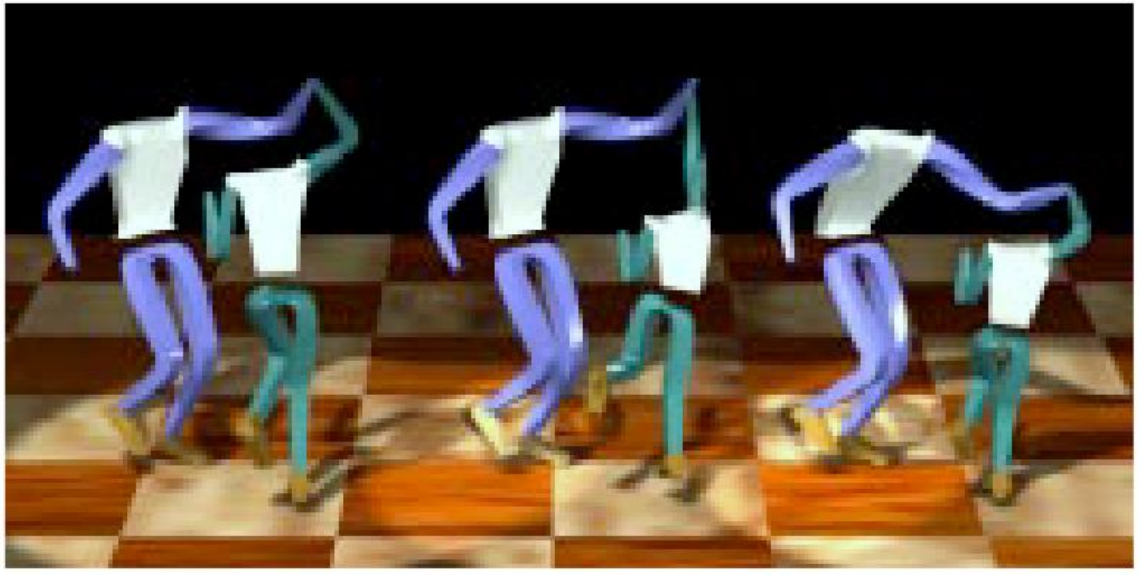


Tennis
Forehand

Obstacles

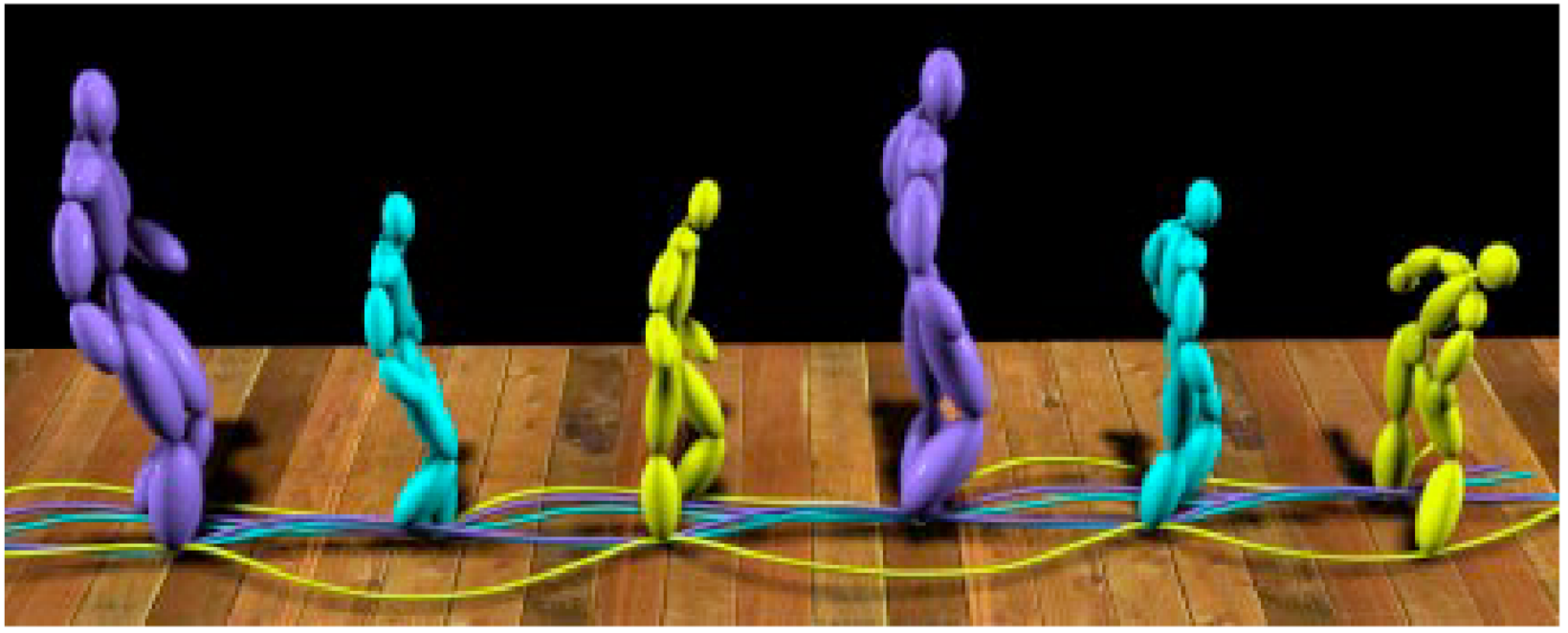
Motion Editing, Motion Retargeting

- Just adding offsets is not satisfactory in some cases
- For the dancing example, we need the hands to be connected during the entire motion
- Or the support foot must be on the ground all the time
- Then need to edit the motion by inverse kinematics instead of simply adding an offset to the motion



Motion Editing, Motion Retargeting

- Edit the end-effector trajectory
- Compute the pose of the character by inverse kinematics so that
  - It tracks the end effector positions
  - the motion is similar to the original motion



Motion Editing by Inverse Kinematics

Mapping from the joint angles $\mathbf{q}$ to the joint positions $\mathbf{x}$

$$\mathbf{x} = f(\mathbf{q})$$

We can linearize the equation and obtain the relation for the velocities:

$$\dot{\mathbf{x}} = J\dot{\mathbf{q}}$$

For computing the joint motion from end effector motion, we can use the pseudo inverse:

$$\dot{\mathbf{q}} = \mathbf{J}^{+}\dot{\mathbf{x}}$$

$$\mathbf{J}^{+} = \mathbf{J}^{T}\left(\mathbf{J}\mathbf{J}^{T} + \lambda^{2}\mathbf{I}\right)^{-1}$$

Tracking the end-effector movements

All solutions that satisfies the constraints:

$$\dot{\mathbf{q}} = \mathbf{J}^{+}\dot{\mathbf{x}} + \underbrace{(\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\mathbf{y}}_{\text{null space term}}$$

This null space term corresponds to redundant degrees of freedom and can be used to perform secondary tasks.

For example, we can set $\mathbf{y}$ to the desired motion $\dot{\mathbf{q}}_{\mathbf{des}}$ to make $\dot{\mathbf{q}}$ similar to $\dot{\mathbf{q}}_{\mathbf{des}}$

$$\dot{\mathbf{q}} = \mathbf{J}^{+}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\dot{\mathbf{q}}_{\mathbf{des}}$$

Tracking the end-effector movements

$$\dot{\mathbf{q}} = \mathbf{J}^{+}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\dot{\mathbf{q}}_{\mathbf{des}}$$

null space term

$(\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\dot{\mathbf{q}}_{\mathbf{des}}$ projects $\dot{\mathbf{q}}_{\mathbf{des}}$ to a motion that does not move the end effector.

Remember $\mathbf{J}^{+}$ produces the minimum motion for the body to produce the end effector motion $\dot{\mathbf{x}}$:

$$(\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\dot{\mathbf{q}}_{\mathbf{des}} = \dot{\mathbf{q}}_{\mathbf{des}} - \mathbf{J}^{+}\mathbf{J}\dot{\mathbf{q}}_{\mathbf{des}} = \dot{\mathbf{q}}_{\mathbf{des}} - \mathbf{J}^{+}\dot{\mathbf{x}}_{\mathbf{des}}$$
$$= \dot{\mathbf{q}}_{\mathbf{des}} - \hat{\dot{\mathbf{q}}}_{\mathbf{des}}$$

this produces $\dot{\mathbf{x}}_{\mathbf{des}}$ — Minimal motion for producing $\dot{\mathbf{x}}_{\mathbf{des}}$

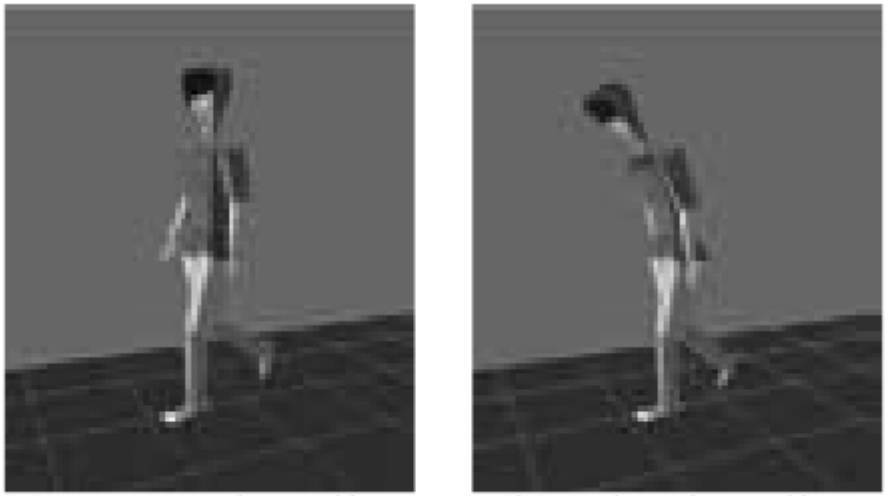
Subtracting the elements that produce the end effector motion


Swing
Dancing

Editing/Blending Motion Styles

Editing the motion styles is a difficult problem but simple methods exists for locomotion

Editing/Blending the motion in the frequency domain is effective for stylizing the motion



(a) Normal walk (b) Tired walk

Editing/Blending Motion Styles

We first do a Fourier analysis of each degrees of freedom of motion 1 (say normal walk)

$$q_m^A(t) = A_{m,0} + \sum_{n>0} A_{m,n} \sin(n\,t + \phi_{m,n})$$

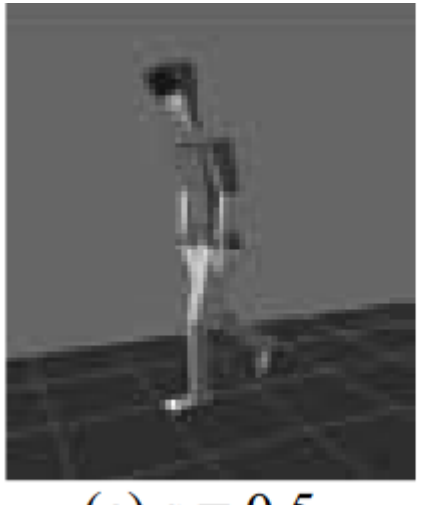
Let's do this for another motion (say tired walk)

$$q_m^B(t) = B_{m,0} + \sum_{n>0} B_{m,n} \sin(n\,t + \psi_{m,n})$$

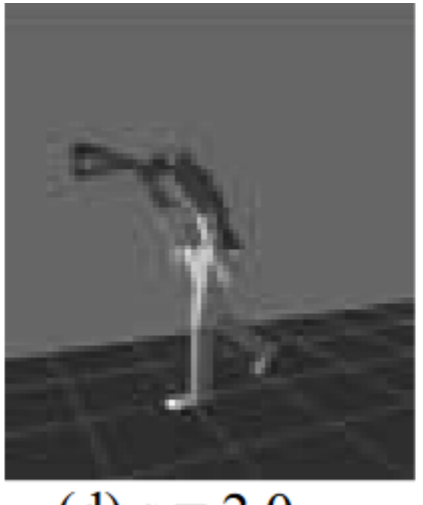
Editing/Blending Motion Styles

Then, we can interpolate/extrapolate the styles by interpolating the Fourier parameters

$$q_m^{A-B}(t) = (1-s)A_{m,0} + sB_{m,0} + \sum_{n>0}\Big((1-s)A_{m,n} + sB_{m,n}\Big)\sin(n\,t + \Big((1-s)\phi_{m,n} + s\psi_{m,n}\Big))$$



(c) s = 0.5



(d) s = 2.0

(e) s = - 0.5

Motion Editing and Blending

Motion editing is good but if you edit too much, the motion can appear unnatural.

Probably it is better to prepare different motions and gradually switch between them to keep the motion natural

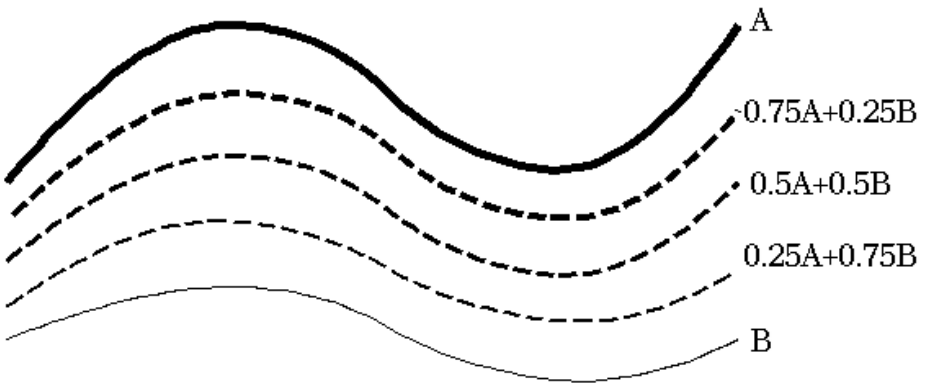
Motion Blending

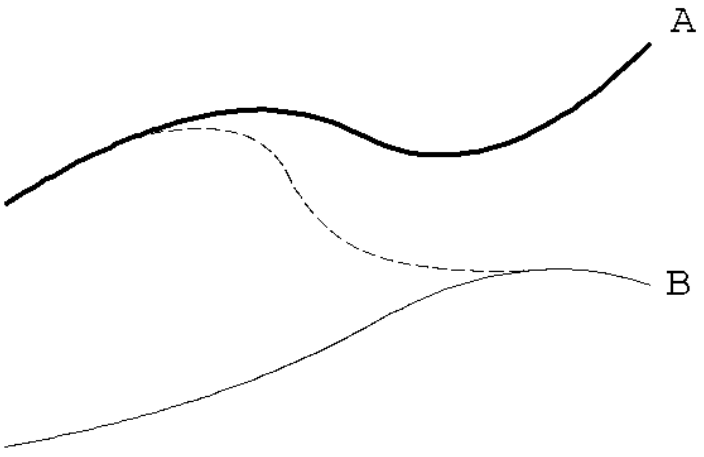
- Given two different motions A and B

  Blended Motion = $(1 - \mathrm{w})\boldsymbol{A} + w\boldsymbol{B}$ $(0 < w < 1)$

How can we use this?

  - Generate a motion inbetween
  - Gradually shift from motion A to motion B
  - Concatenating two motions – blending the two ends by gradually shifting s from 0 to 1





Motion Blending: Blending Multiple Motions

Given example motion $M_1,\ldots,M_n$

We can define parameters like

$(w_1, w_2, \ldots, w_n)$ s.t.

$w_1+w_2+ \ldots. +w_n = 1$

And then blend the motion by

$w_1M_1 +w_2M_2 \ldots. +w_nM_n$

Motion Blending: Blending Multiple Motions

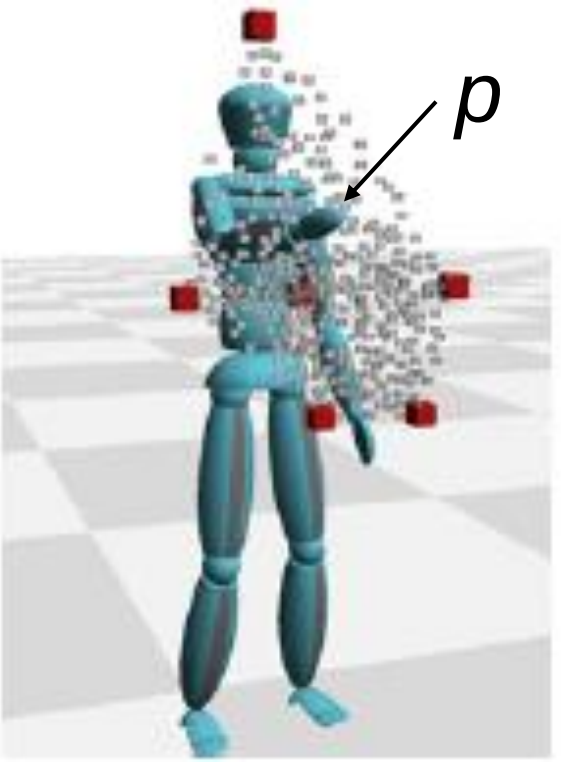
Computing the weights?

- Say we have a target hand position p:
- And the position for each motion is $(p_1, p_2, \ldots, p_n)$,

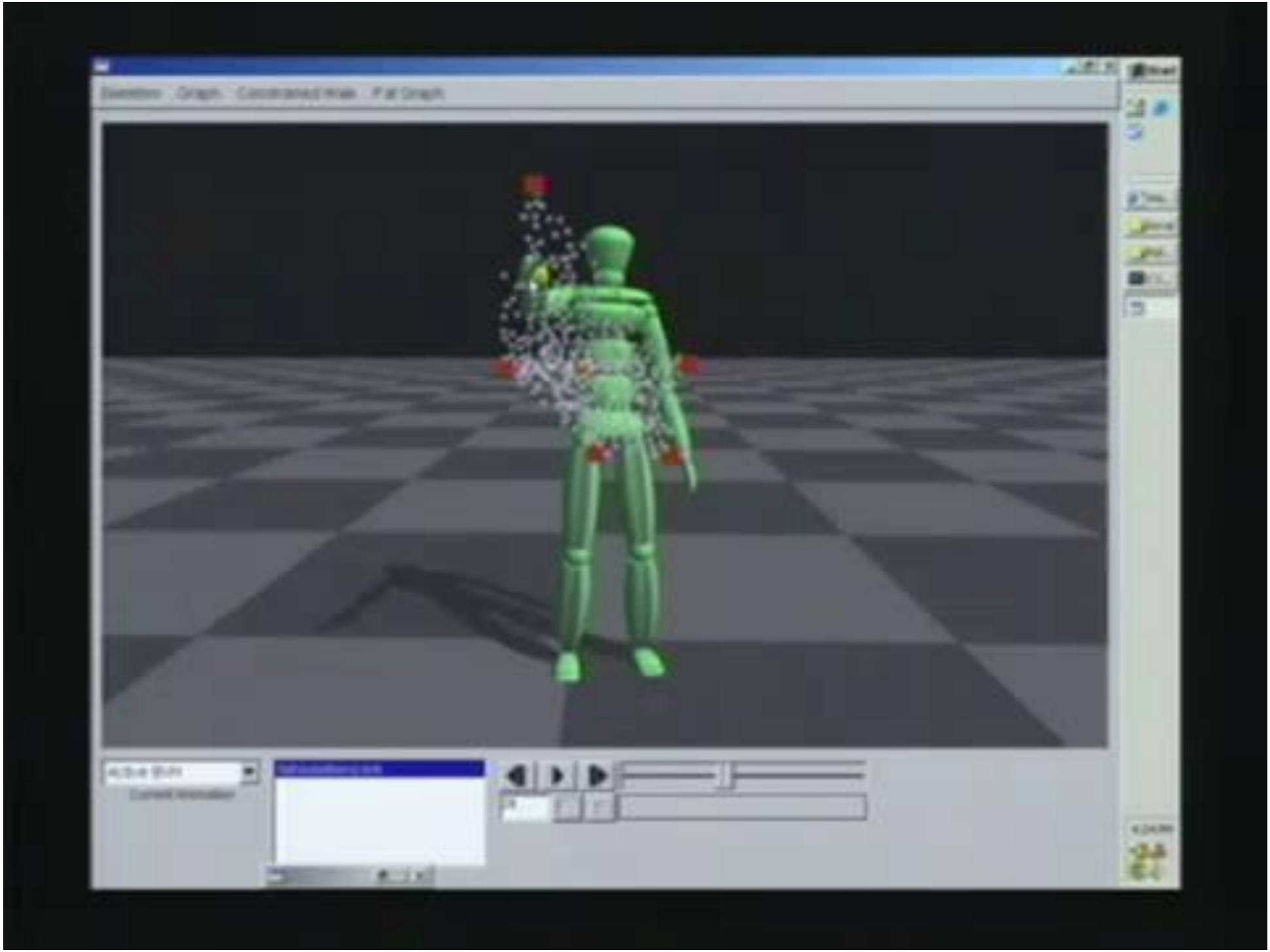
The weights can be computed by

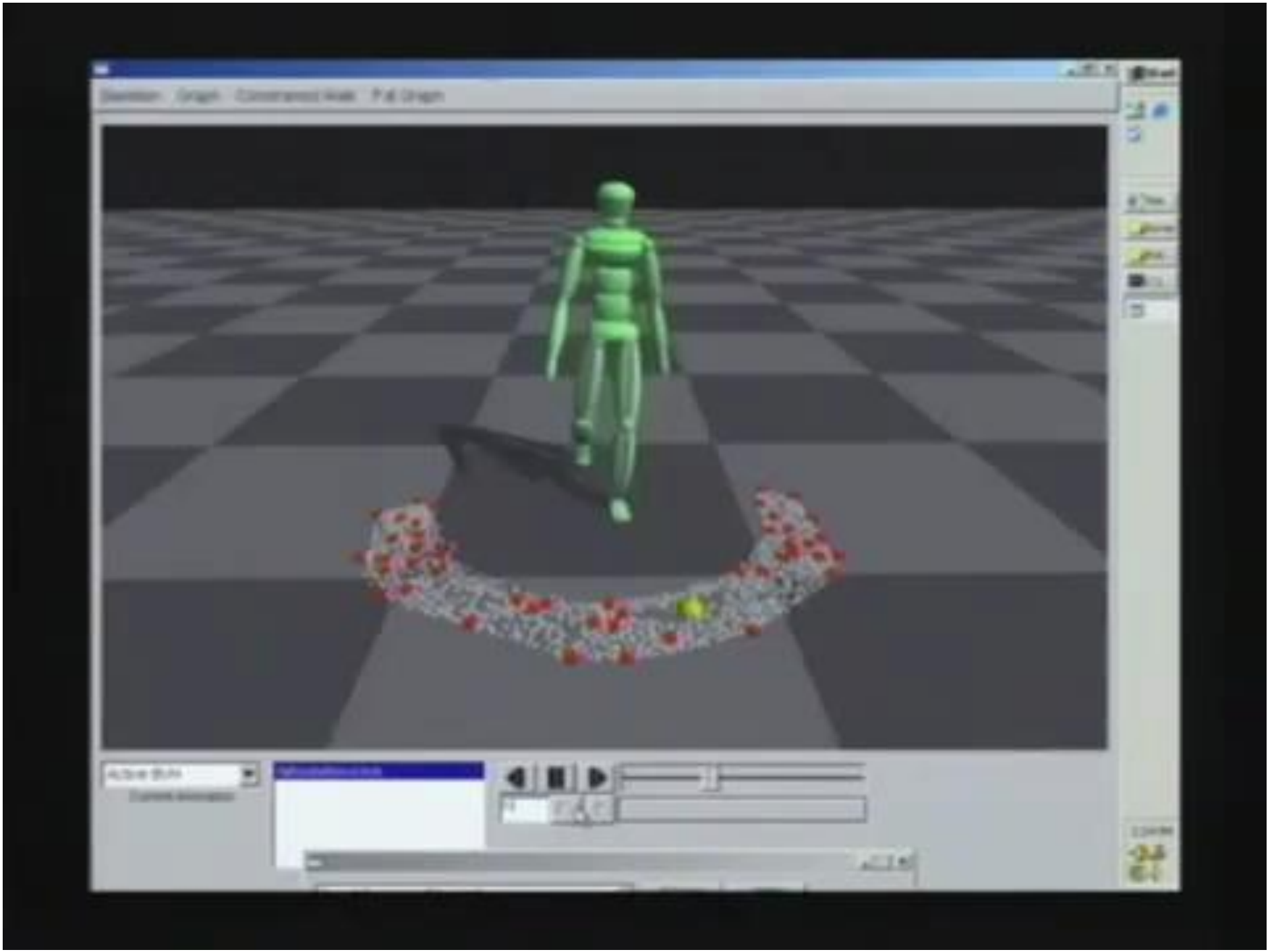
$$w_i = a_i / (a_1 + a_2 + \ldots + a_n)$$

where $a_i = 1 / |p_i - p|$



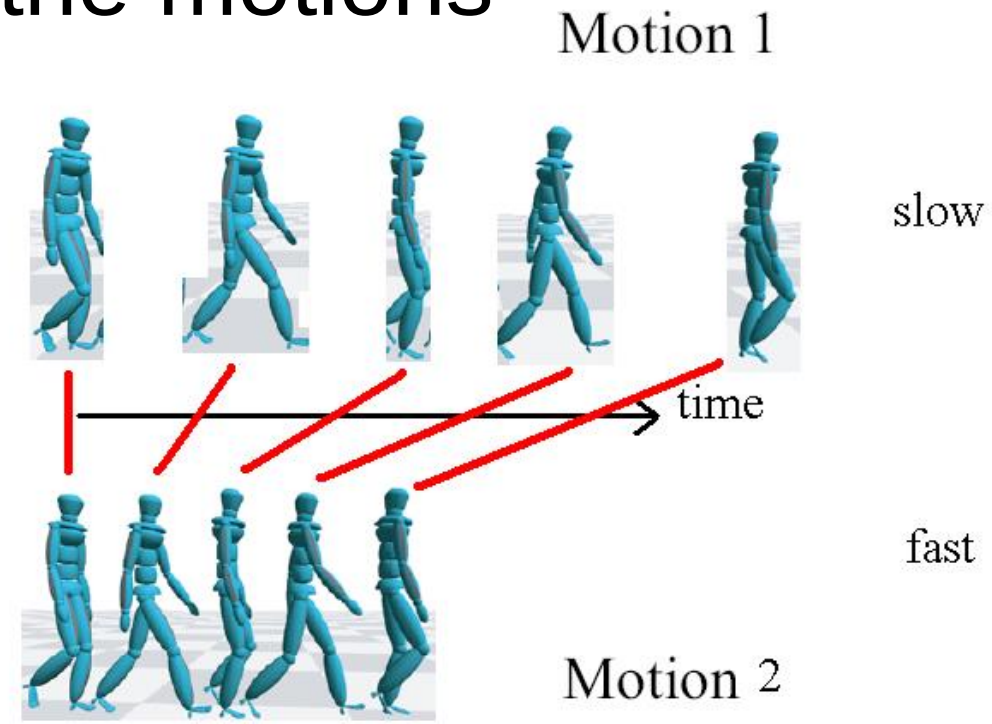
http://www.youtube.com/watch?v=qtNsF-nTp4A (around 0:50)





The problems when blending motions

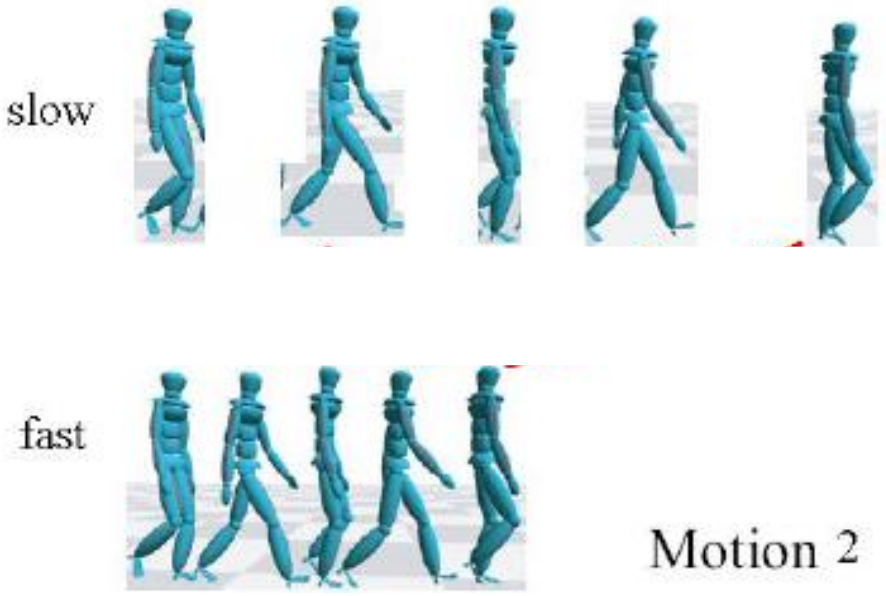
- The durations of motions are different
- The timing of some events are different
  - Suppose we blend two walking motions
  - The timing of steps are different
  - Need to synchronize the motions



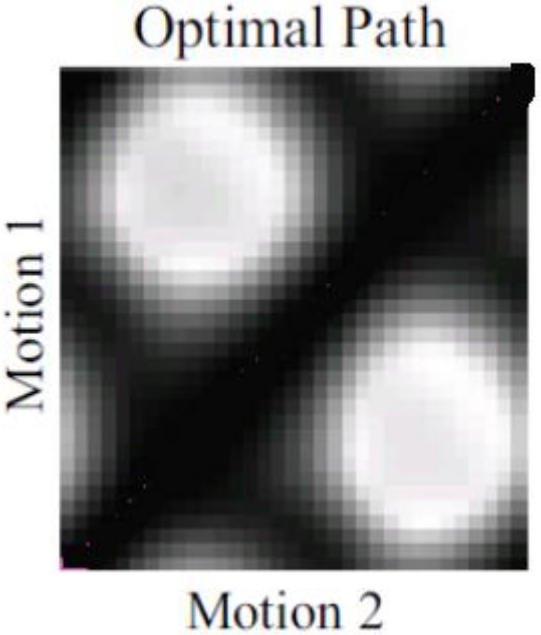
Dynamic Time Warping (DTW)

– Define the difference of postures at every frame in motion 1 and motion 2 ： $D(\mathbf{q}_1, \mathbf{q}_2) = d(\mathbf{q}_1, \mathbf{q}_2) + \alpha d(\dot{\mathbf{q}}_1, \dot{\mathbf{q}}_2)$

– Create a similarity matrix of all frames

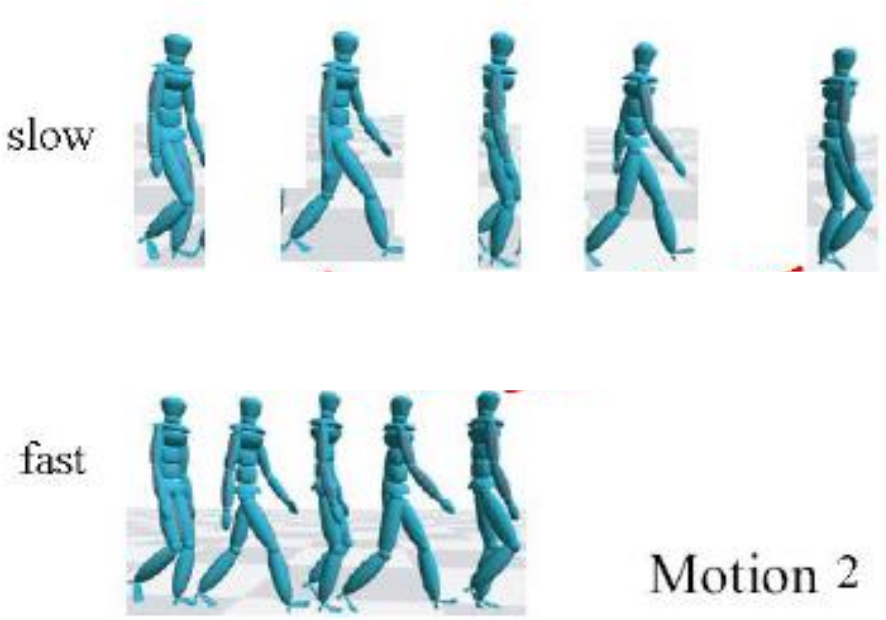


http://www.youtube.com/watch?v=EL7ARaH5jHU



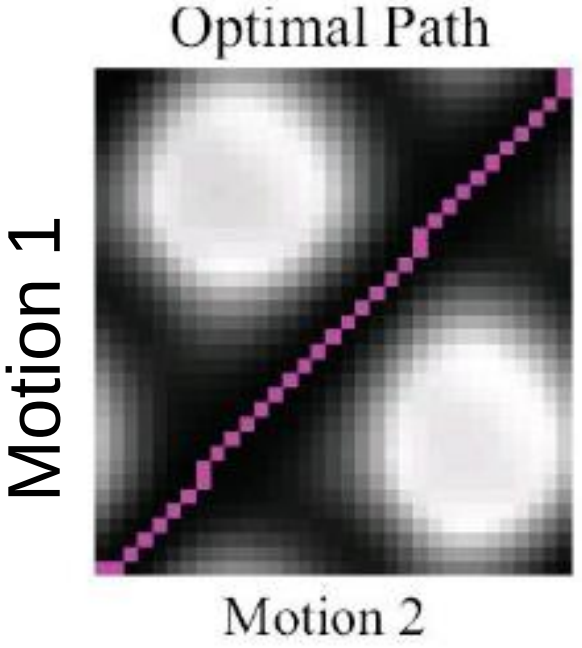
Dynamic Time Warping (DTW)

- Define the difference of postures at every frame in motion 1 and motion 2 :
- Create a similarity matrix of all frames
- Find the shortest path from the left bottom to the right top in this matrix (assume the matrix is a map) - this gives



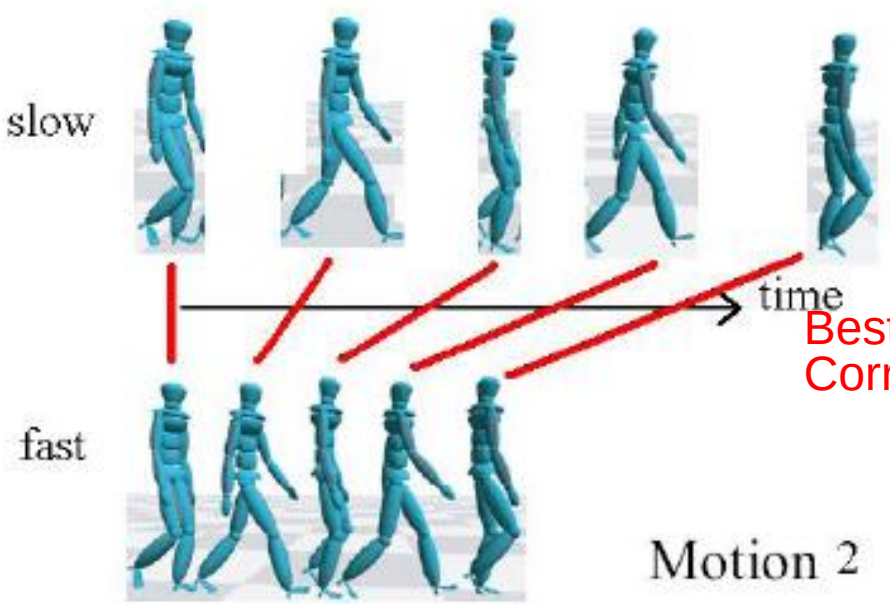
http://www.youtube.com/watch?v=EL7ARaH5jHU

$$min \sum_i d_i(q_1, q_2)$$



Dynamic Time Warping (DTW)

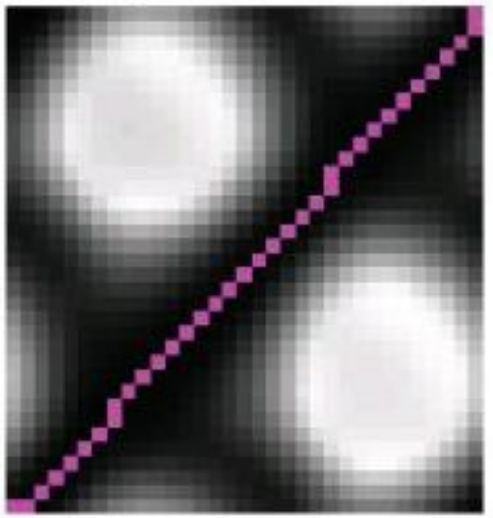
- Define the difference of postures at every frame in motion 1 and motion 2 :
- Create a similarity matrix of all frames
- Find the shortest path from the left bottom to the right top in this matrix (assume the matrix is a map) - this gives



Best Correspondence!!

$$min \sum_i d_i(q_1, q_2)$$

Optimal Path



Motion 2

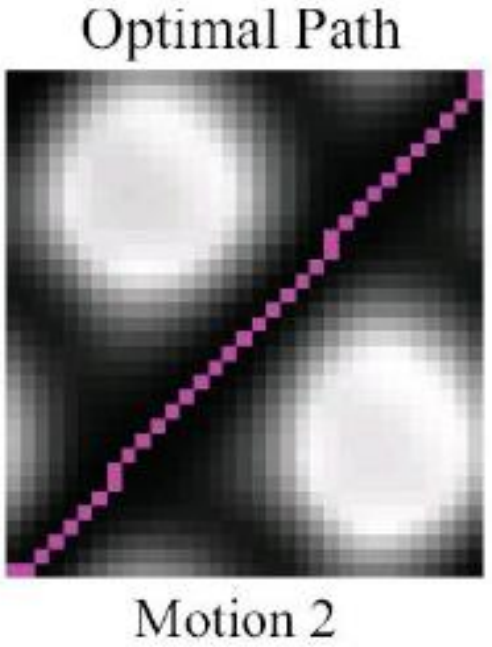
http://www.youtube.com/watch?v=EL7ARaH5jHU

Dynamic Time Warping (DTW)

– The computed distance also represents the similarity of the motion

– We can search for motion clips that are similar in the entire motion database

$$min \sum_{i} d_i(q_1, q_2)$$




Searching Motion
Data Sets

Discussions: Motion Editing/Blending

- Motion editing: Editing a motion capture data to satisfy constraints, retargeting and change the style

- Motion blending: synthesize novel motion by blending different movements.
    - Can produce more natural motion
    - Data-driven motion synthesis

Motion Databases

Mixamo: https://www.mixamo.com/#/
Not only the motion but also the skin data

Website to easily download it
https://github.com/ChrisWu1997/2D-Motion-Retargeting/blob/master/dataset/Guide%20For%20Downloading%20Mixamo%20Data.md

CMU : http://mocap.cs.cmu.edu/

– Various motion data in different topics
  – Locomotion
  – Physical activity and sports
  – Human interactions

Lafan1 : https://github.com/ubisoft/ubisoft-laforge-animation-dataset

Readings / References

- DART (Dynamic Animation and Robotics Toolkit) : https://dartsim.github.io/
- Witkin and Popovic “Motion Warping”, SIGGRAPH 95
- Choi, Kwang-Jin, and Hyeong-Seok Ko. "Online motion retargetting." *The Journal of Visualization and Computer Animation* 11.5 (2000): 223-235.
- Kovar and Gleicher, “Automated Extraction and Parameterization of Motions in Large Data Set”, SIGGRAPH 2004
- Liu, C. K. and Popović, Z. Synthesis of Complex Dynamic Character Motion from Simple Animation (ACM SIGGRAPH 2002)
- Wang et al., Optimizing Walking Controllers for Uncertain Inputs and Environments , SIGGRAPH 2010
- Komura et al. “Creating and retargeting motion by the musculoskeletal human body model”, Visual Computer 2000