Lecture 06:
Learning-based Character Animation

Libin Liu

School of Intelligence Science and Technology
Peking University

GAMES105 课程交流

VCL @ PKU

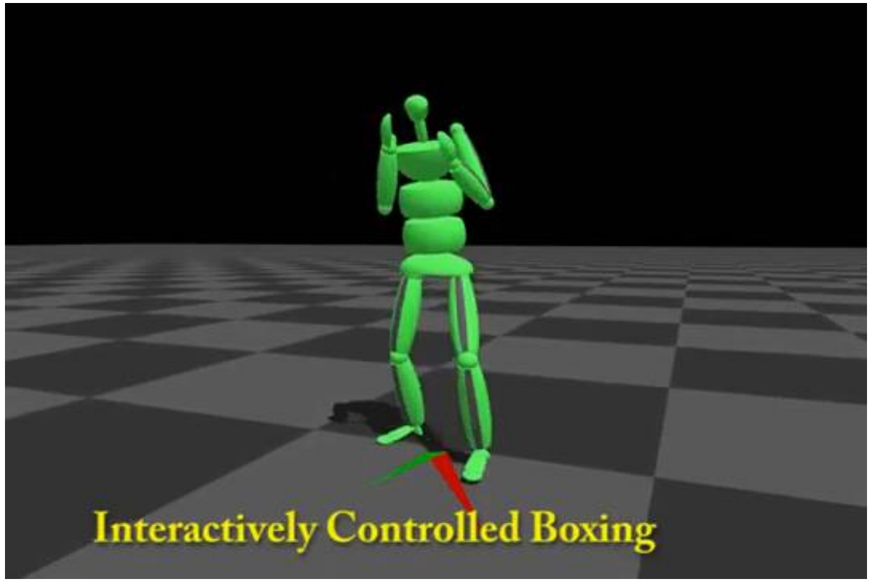
Outline

- Recap: interactive character animation
  - Motion Graphs
  - Motion Matching
- Statistical Models of Human Motion
  - Principal Component Analysis
  - Gaussian Models
- Learning-based Models
  - ……

Recap: Interactive Animation

How to make a character respond to user command?

How to create interactive animation?



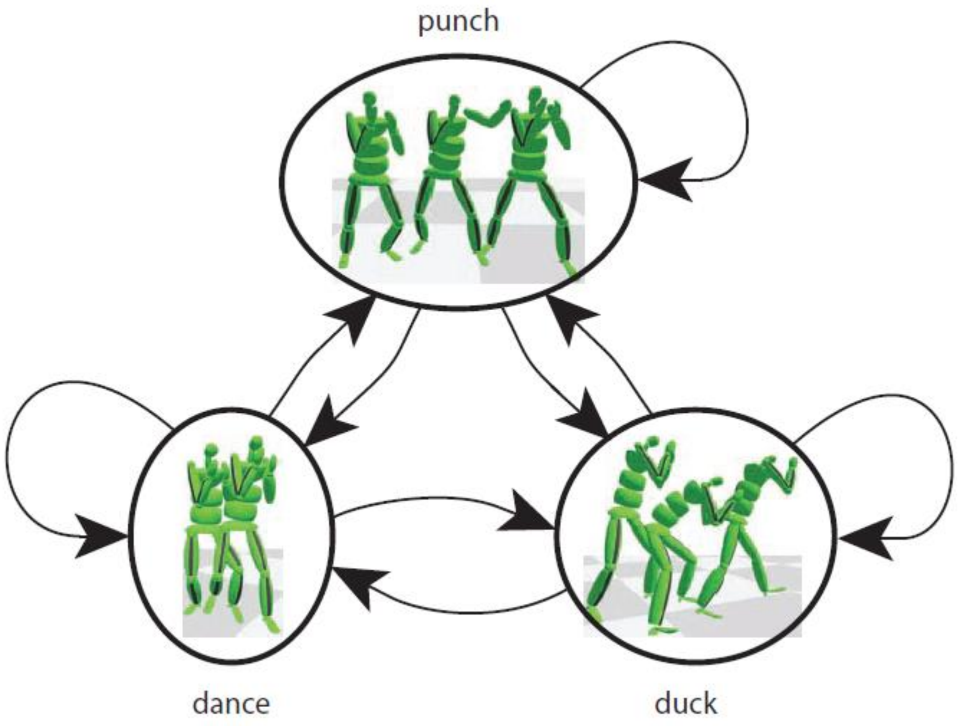
[Heck and Gleicher 2007, Parametric Motion Graphs]

check user input

find a nice animation clip

play it

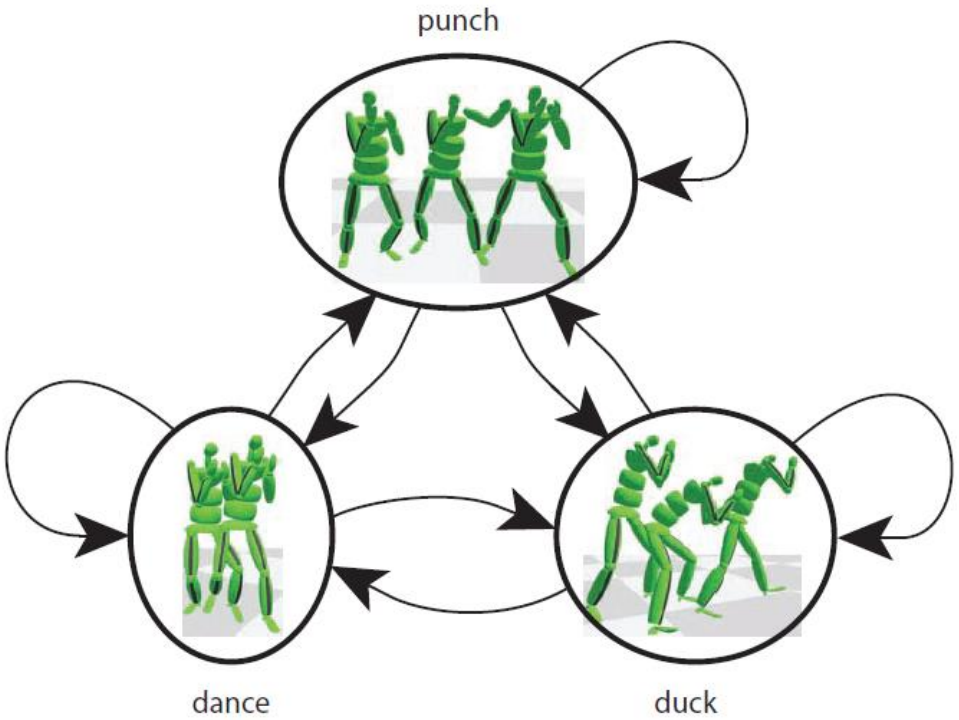
Motion Graphs



[Heck and Gleicher 2007, Parametric Motion Graphs]

- check user input
- find a nice animation clip
- play it

Motion Graphs



[Heck and Gleicher 2007, Parametric Motion Graphs]

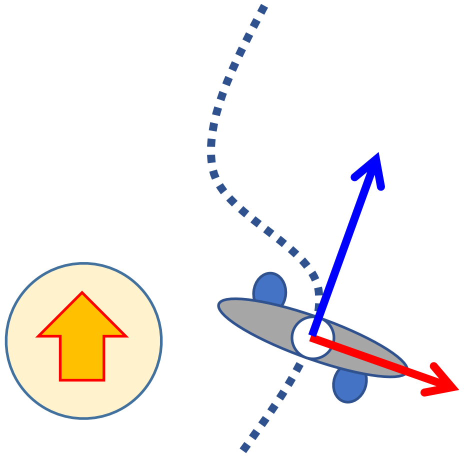
at the end of the current clip:

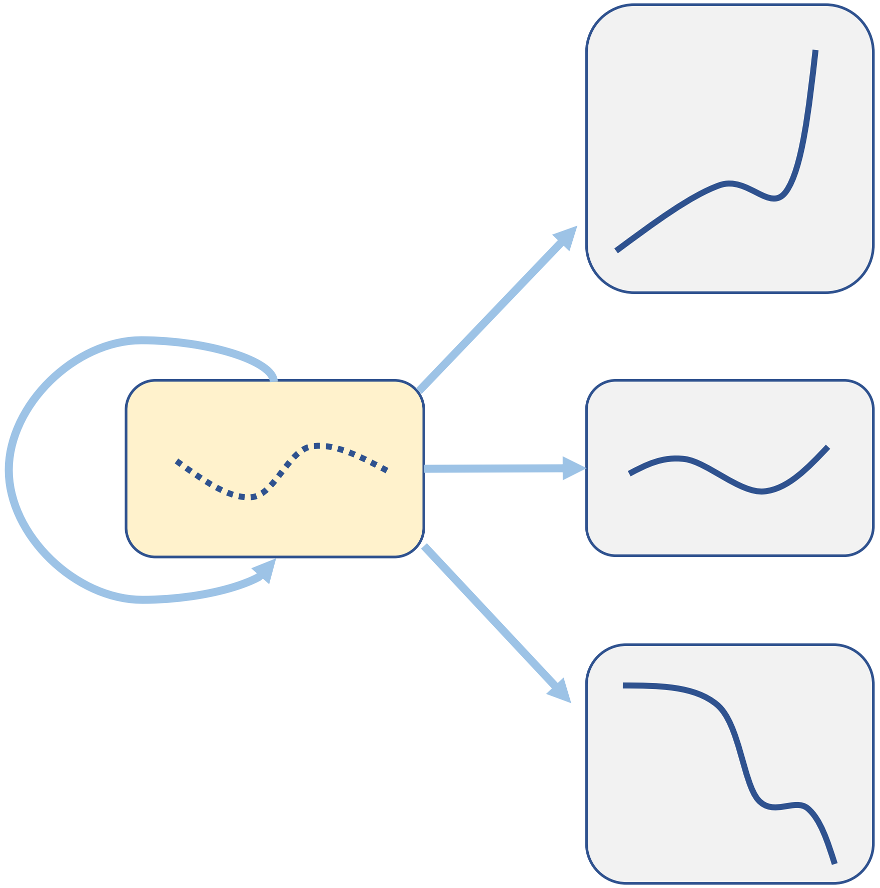
check user input

find a nice animation clip

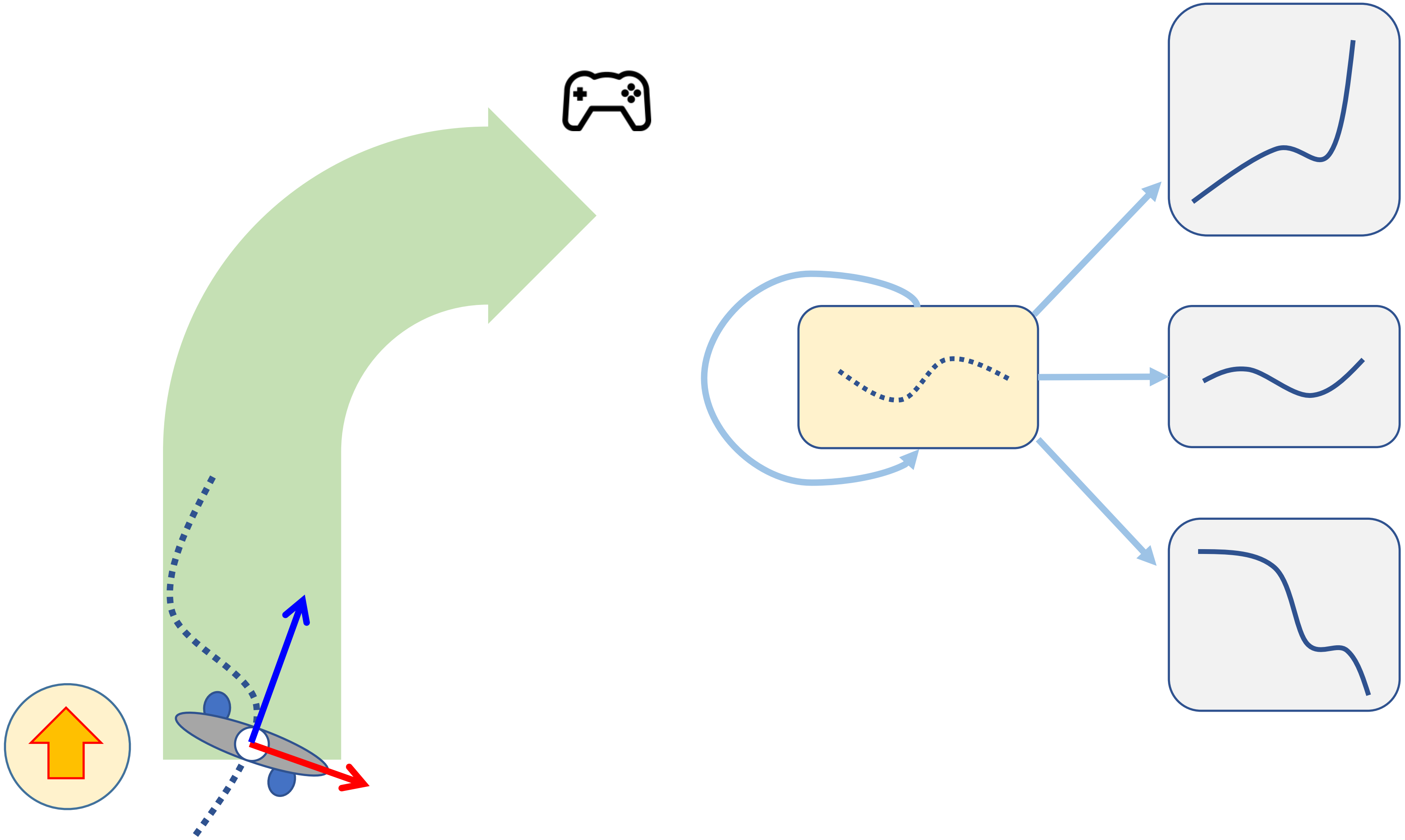
play it

Motion Graphs

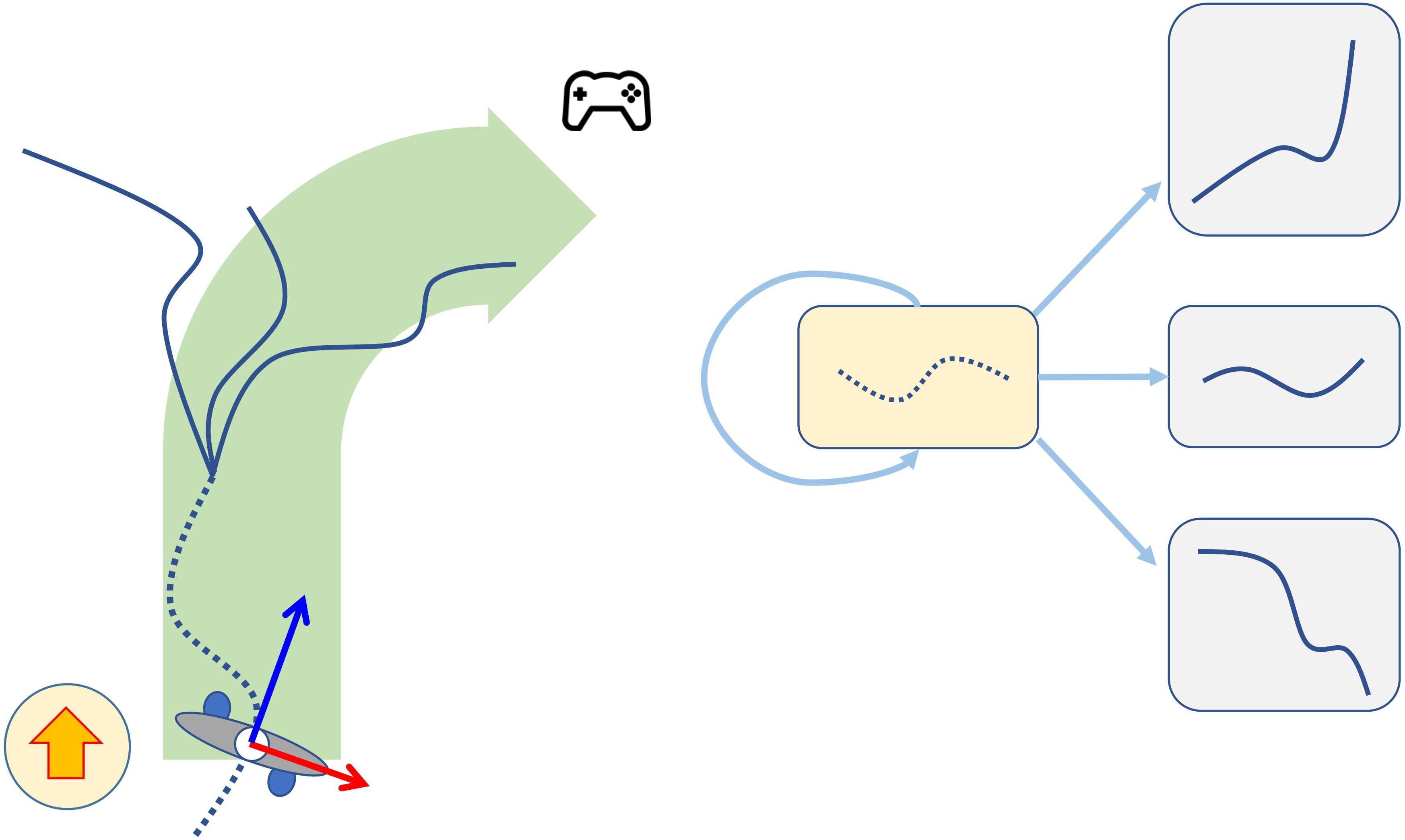




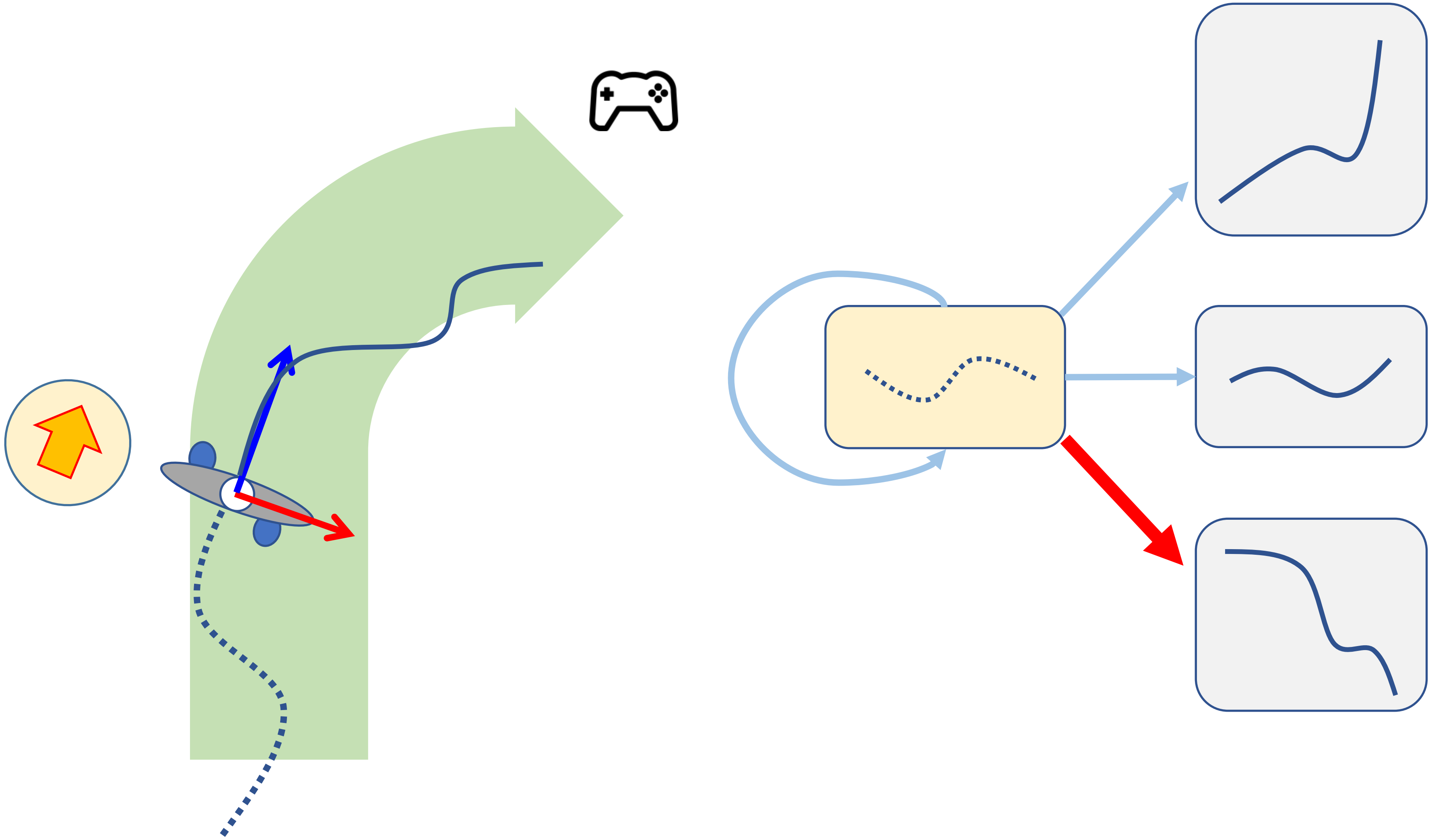
Motion Graphs



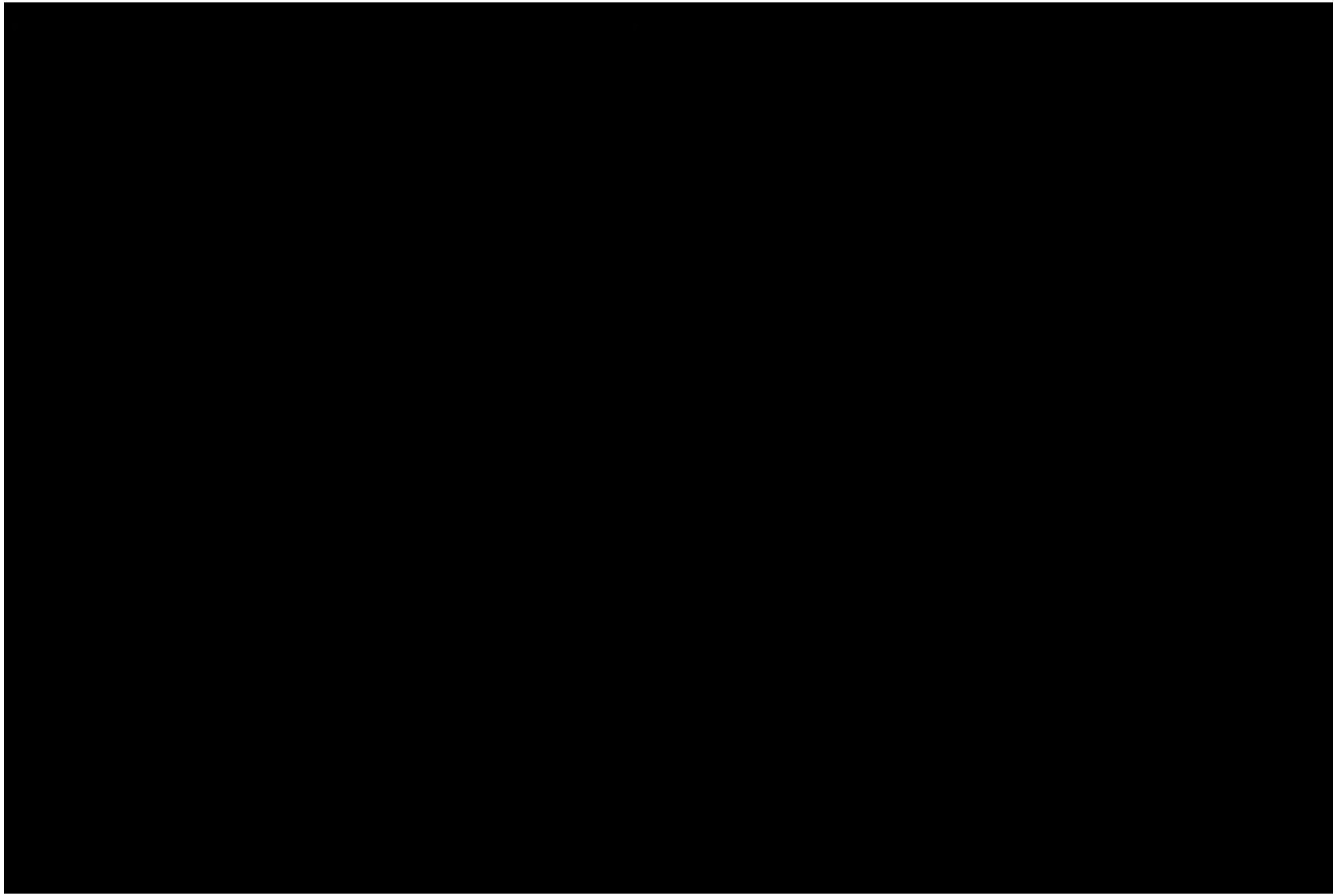
Motion Graphs



Motion Graphs

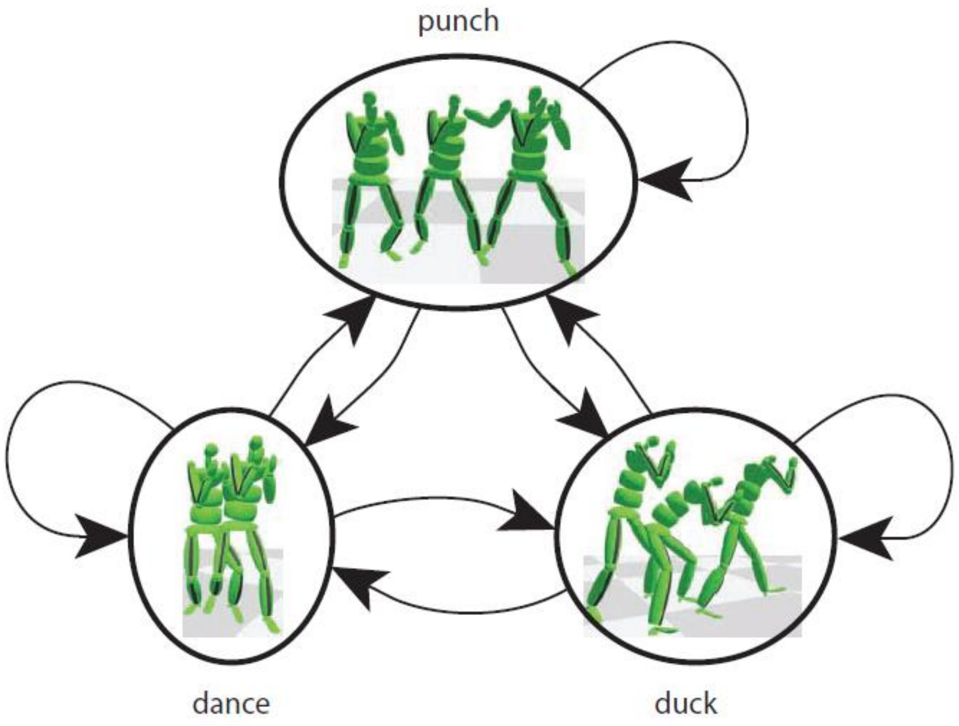


Motion Graphs



Motion Planning with Motion Graph and A*
https://www.youtube.com/watch?v=ekx0bXz25Pw

Motion Graphs



[Heck and Gleicher 2007, Parametric Motion Graphs]

at the end of the current clip:

check user input

find a nice animation clip

play it

Need a Faster Response?

Motion Graphs / State Machines

at the end of the current clip:

check user input

find a nice animation clip

play it

at every frame:

check user input

find a nice next pose

update the character

Need a Faster Response?

Motion Graphs / State Machines

at the end of the current clip:

check user input

find a nice animation clip

play it

Motion Fields / Motion Matching

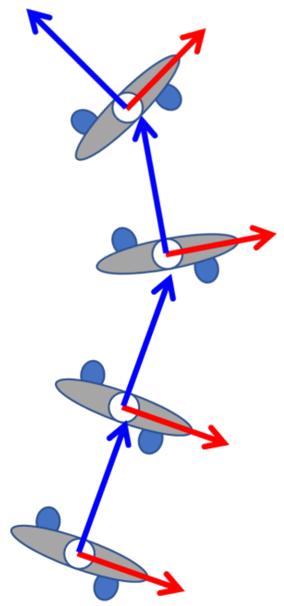
at every frame:

check user input

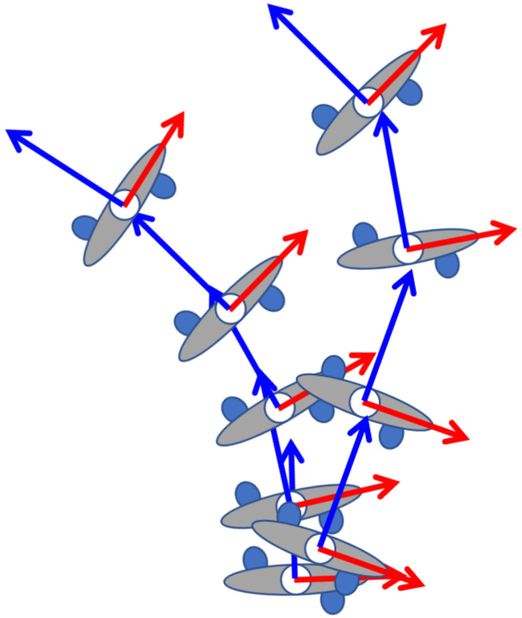
find a nice next pose

update the character

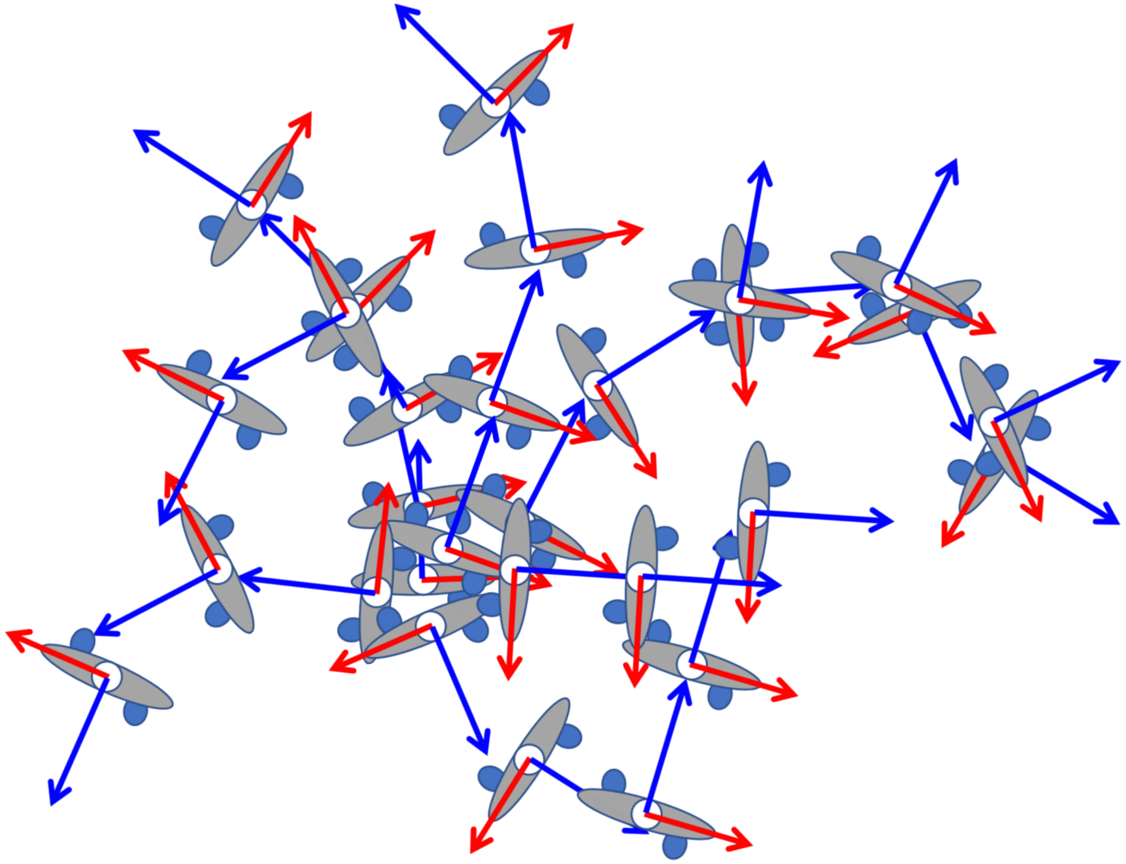
Motion Fields



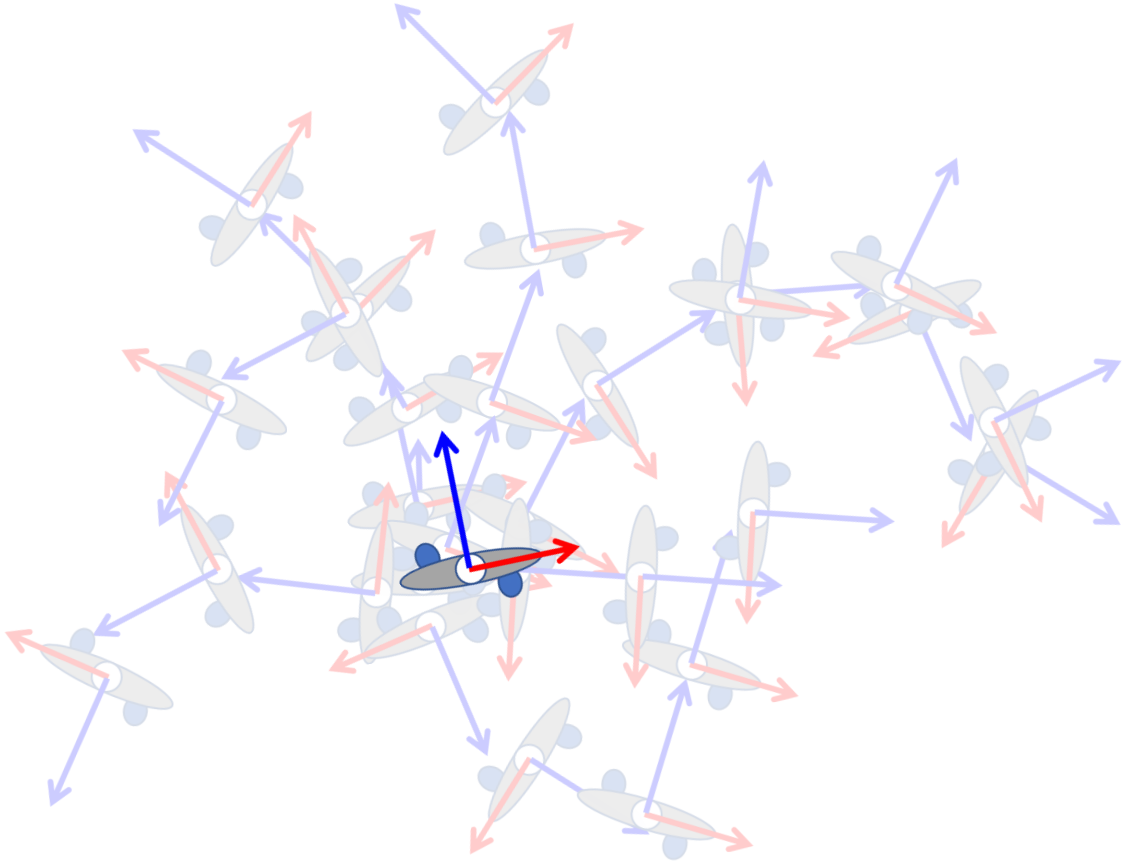
Motion Fields



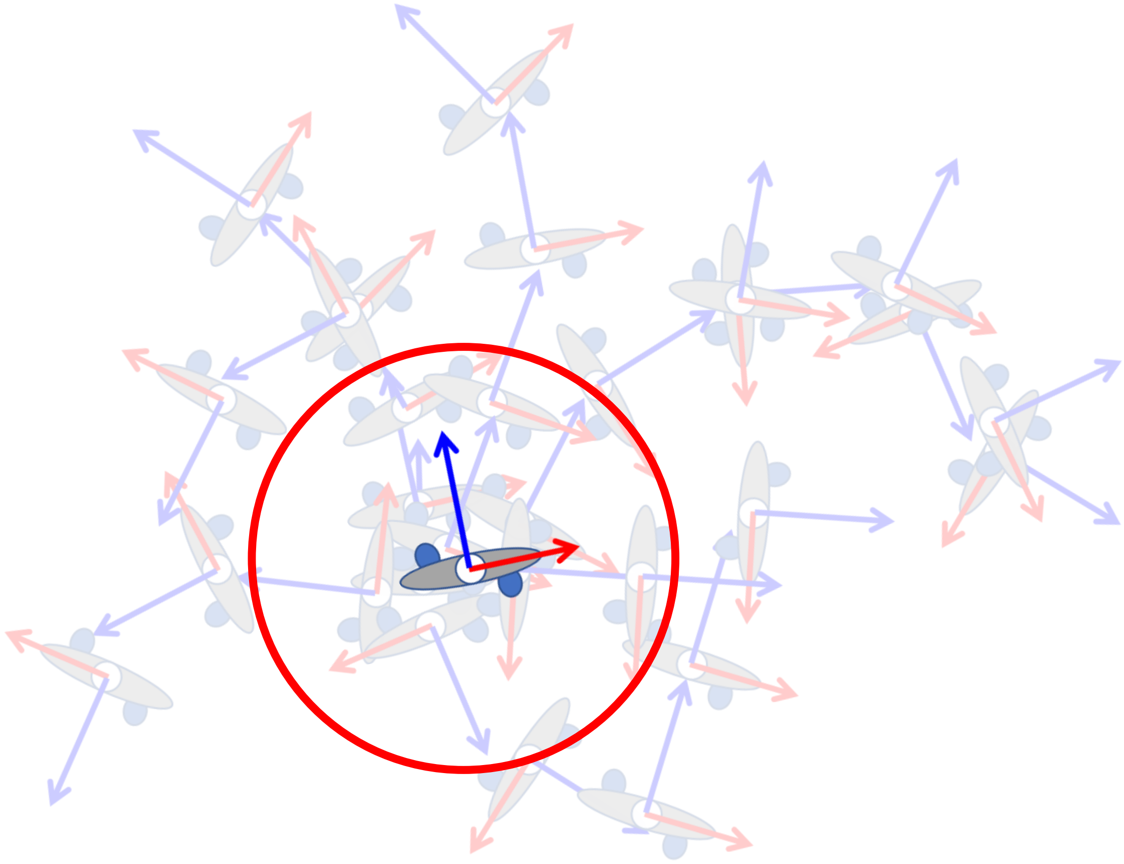
Motion Fields



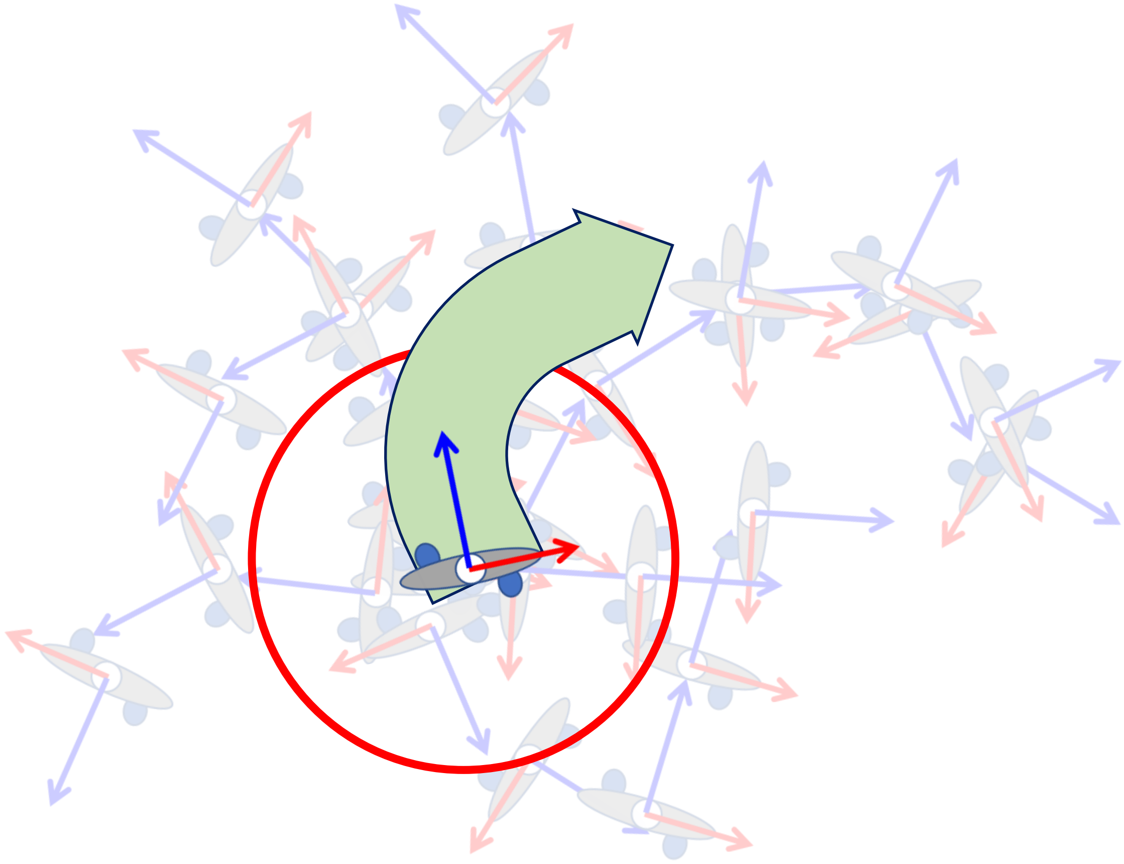
Motion Fields



Motion Fields



Motion Fields

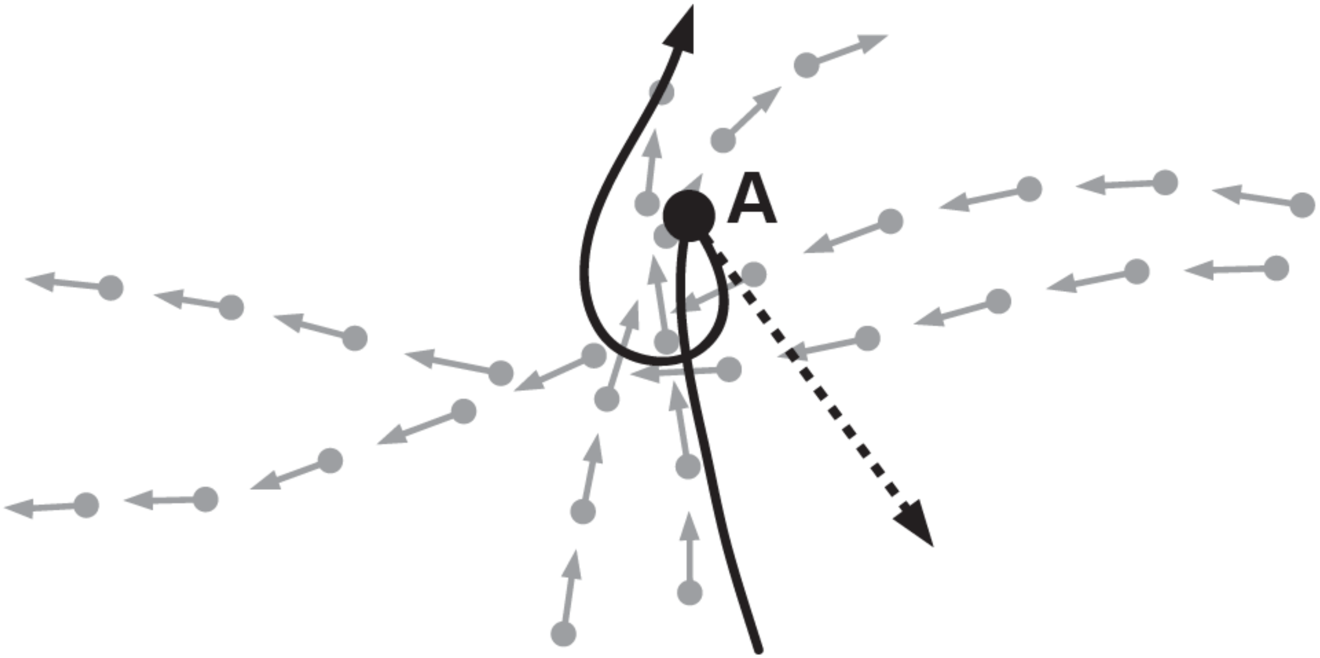


Motion Fields

Motion Fields for Interactive Character Locomotion

Yongjoon Lee[1,2*] Kevin Wampler[1†] Gilbert Bernstein[1] Jovan Popović[1,3] Zoran Popović[1]
[1]University of Washington [2]Bungie [3]Adobe Systems

* SIGGRAPH 2010

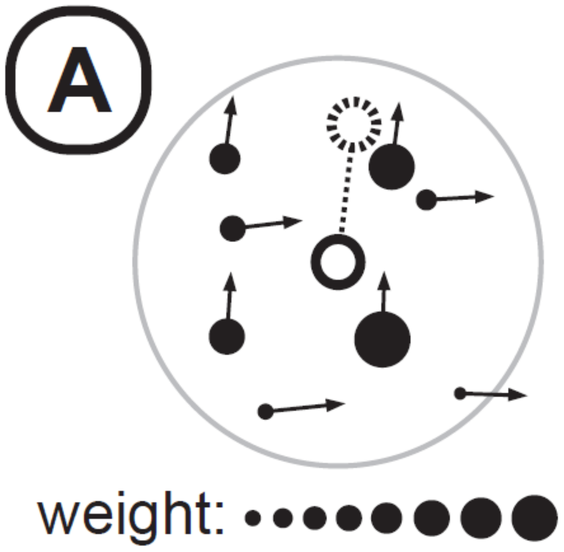


Motion Fields

Motion Fields for Interactive Character Locomotion

Yongjoon Lee[1,2*] Kevin Wampler[1†] Gilbert Bernstein[1] Jovan Popović[1,3] Zoran Popović[1]
[1]University of Washington [2]Bungie [3]Adobe Systems

* SIGGRAPH 2010

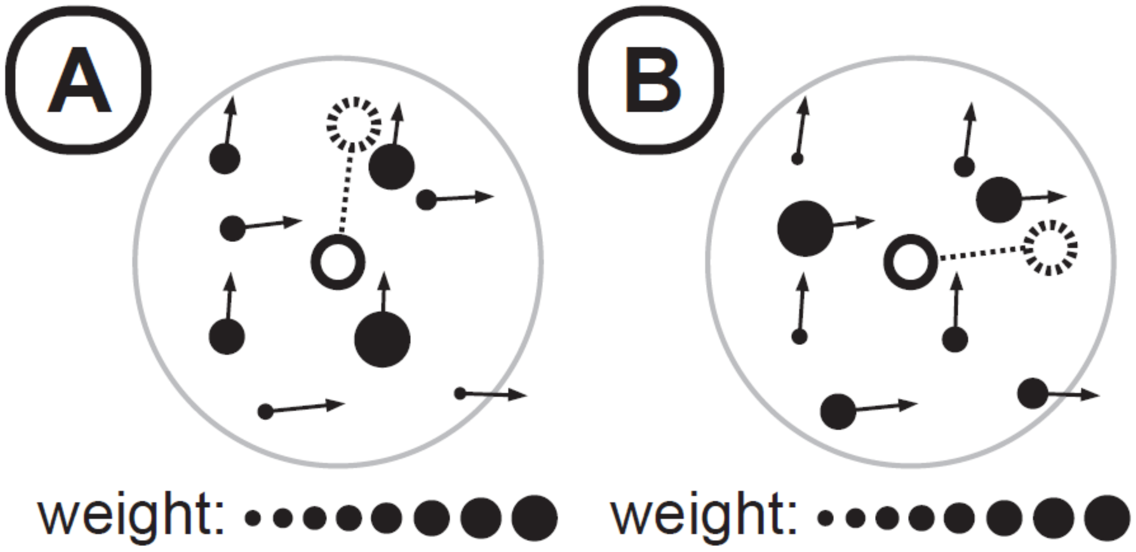


Motion Fields

Motion Fields for Interactive Character Locomotion

Yongjoon Lee[1,2*] Kevin Wampler[1†] Gilbert Bernstein[1] Jovan Popović[1,3] Zoran Popović[1]
[1]University of Washington [2]Bungie [3]Adobe Systems

* SIGGRAPH 2010

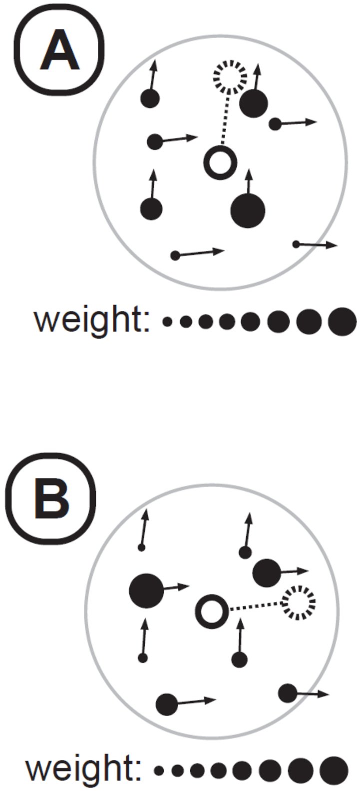


Motion Fields

Motion Fields for Interactive Character Locomotion

Yongjoon Lee[1,2*] Kevin Wampler[1†] Gilbert Bernstein[1] Jovan Popović[1,3] Zoran Popović[1]
[1]University of Washington [2]Bungie [3]Adobe Systems

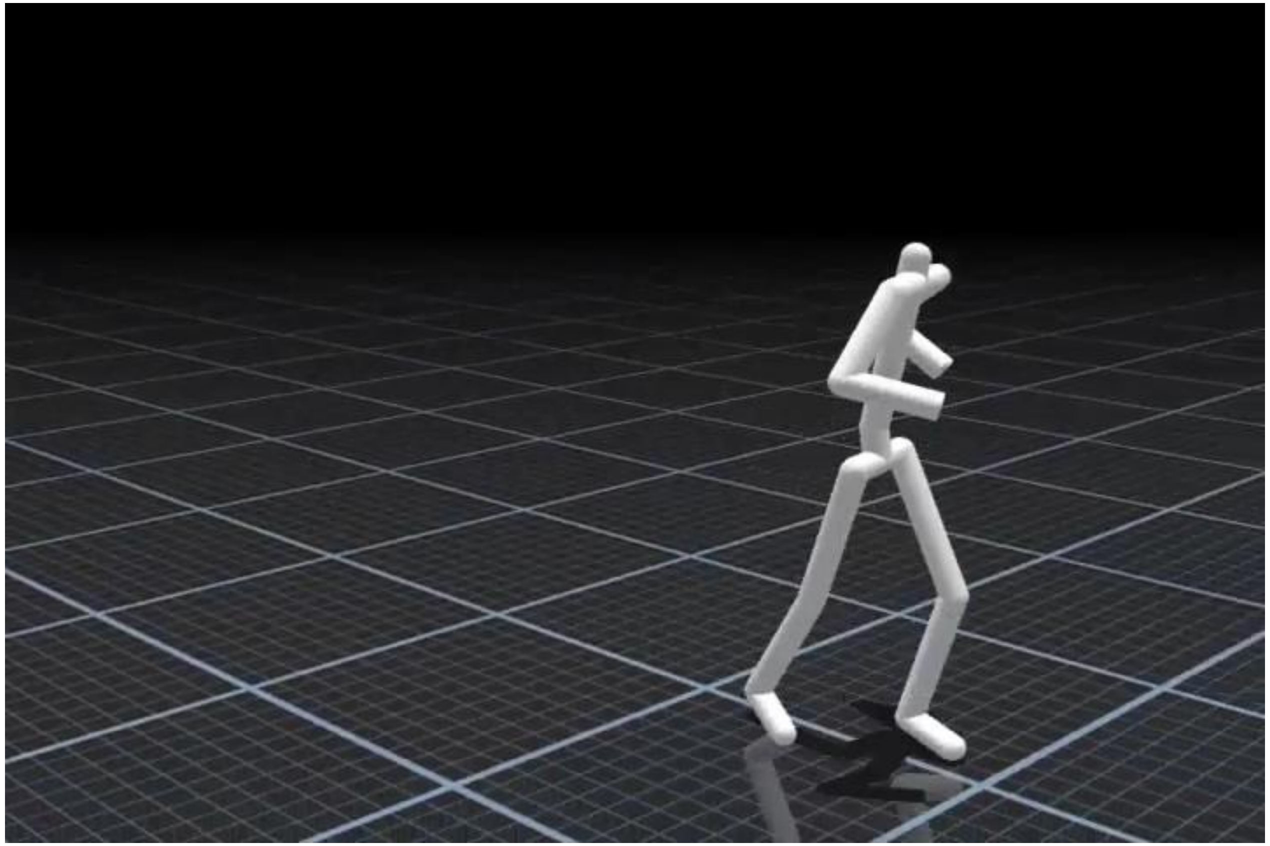
* SIGGRAPH 2010



at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

Motion Fields



Lee et al. 2010. Motion Fields

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

Motion Fields

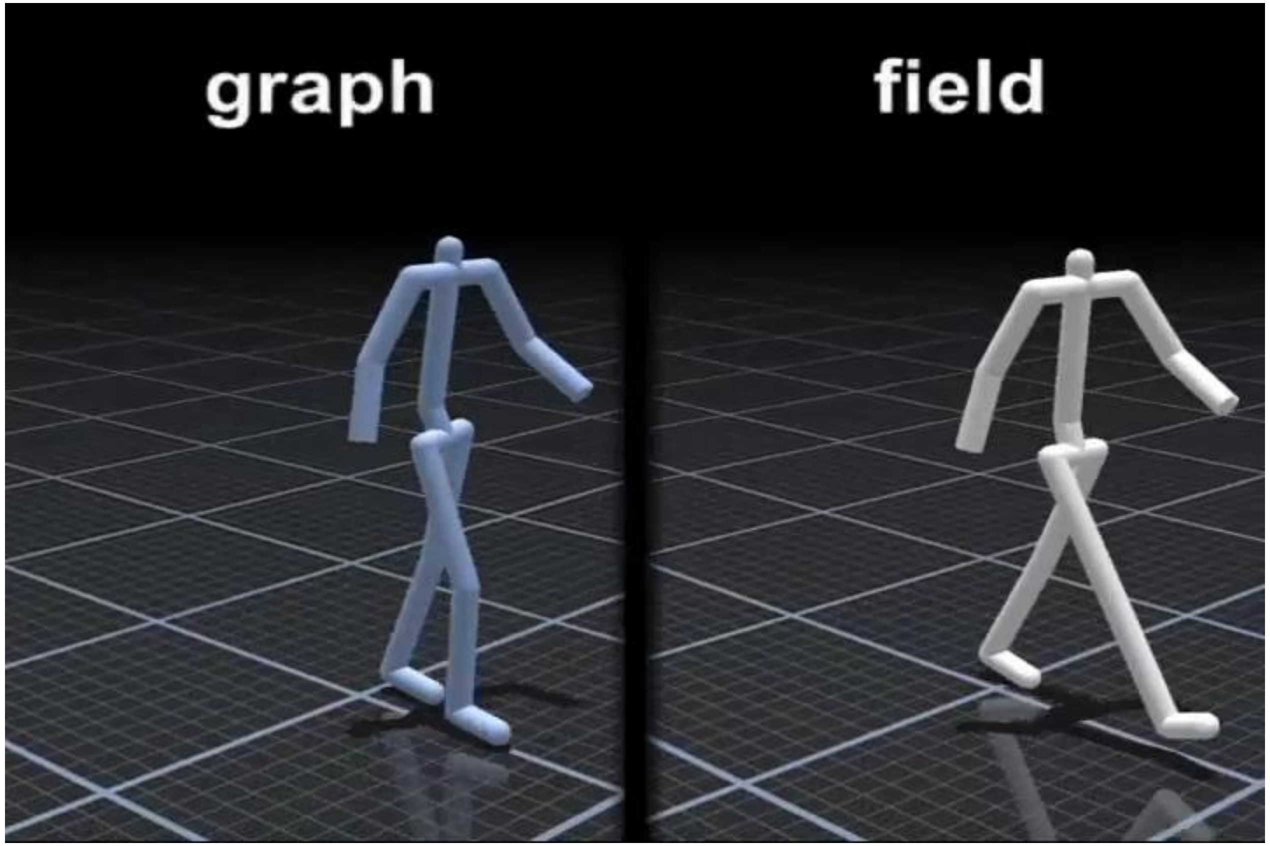
Lee et al. 2010. Motion Fields

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

Motion Fields



Lee et al. 2010. Motion Fields

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

Motion Fields

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

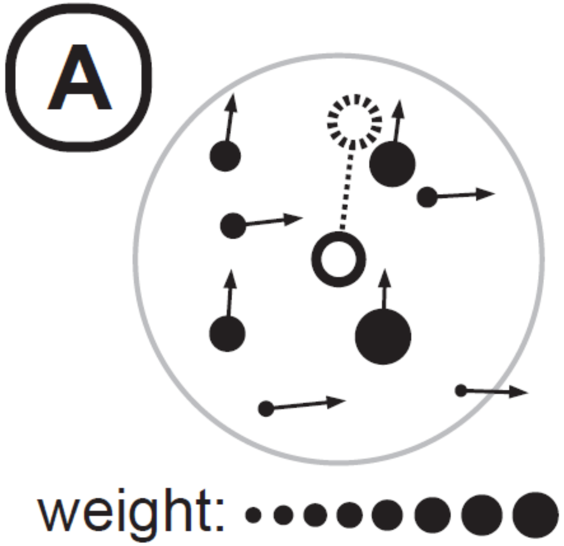
Motion Fields

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

How?



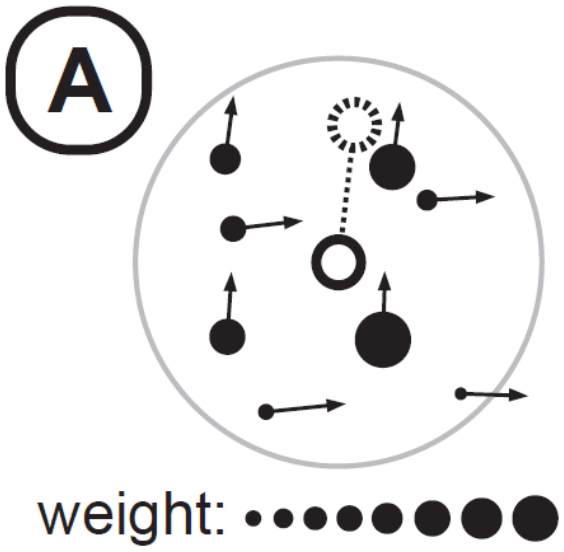
Motion Fields

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

How? Reinforcement learning...



Motion Matching

Motion Fields

at every frame:

- check user input
- find N nearest neighbors of the current state
- blend these neighbors according to user input
- update the character

Motion Matching

at every frame:

- check user input
- find **the** nearest neighbors of the current state **according to** user input
- smoothly blend current pose to the nearest neighbor pose

Motion Matching

- We need a distance function / metric to define the **nearest** neighbor

Motion Matching

- We need a distance function / metric to define the **nearest** neighbor

$$\text{next\_pose} = \min_{i \in \text{Dataset}} \|x_{\text{curr}} - x_i\|$$

x: feature vector

Motion Matching

- We need a distance function / metric to define the **nearest** neighbor

$$\text{next\_pose} = \min_{i \in \text{Dataset}} \|x_{\text{curr}} - x_i\|$$

x: feature vector

A possible set of feature vectors:

- root linear/angular velocity
- position of end effectors w.r.t. root joint
- linear/angular velocity of end effectors w.r.t. root joint
- future heading position/orientation (e.g. in 0.5s, 1.0s, 1.5s, etc.)
- foot contacts
- ……

Motion Matching

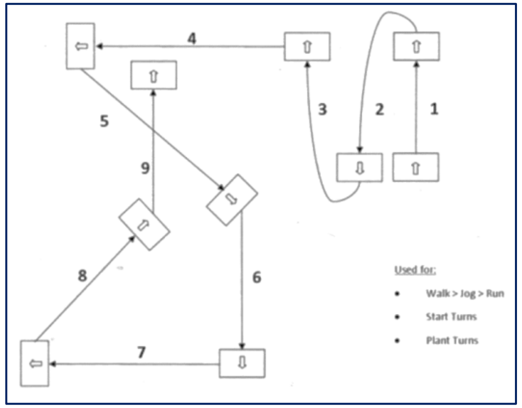
- We need a smooth motion
  - Only do the search every few frames
  - Smoothly blend current pose to the target pose
    - Inertialized blending (ref. https://www.theorangeduck.com/page/spring-roll-call by Daniel Holden)

Motion Matching

- We need a smooth motion
  - Only do the search every few frames
  - Smoothly blend current pose to the target pose
    - Inertialized blending (ref. https://www.theorangeduck.com/page/spring-roll-call by Daniel Holden)

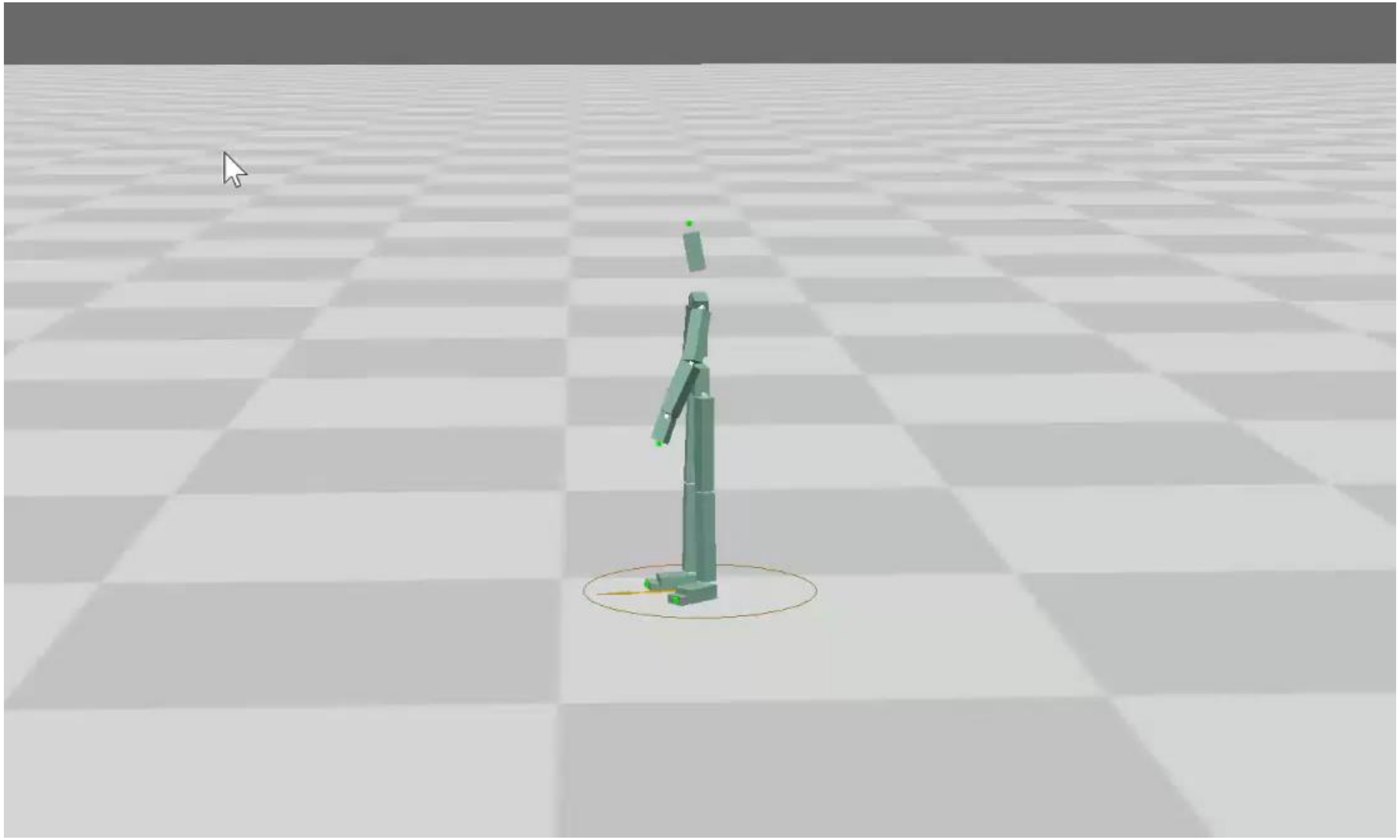
- We need a good performance
  - An efficient data structure for searching
    - e.g. KD-tree

  - A efficient dataset
    - "Dance card"



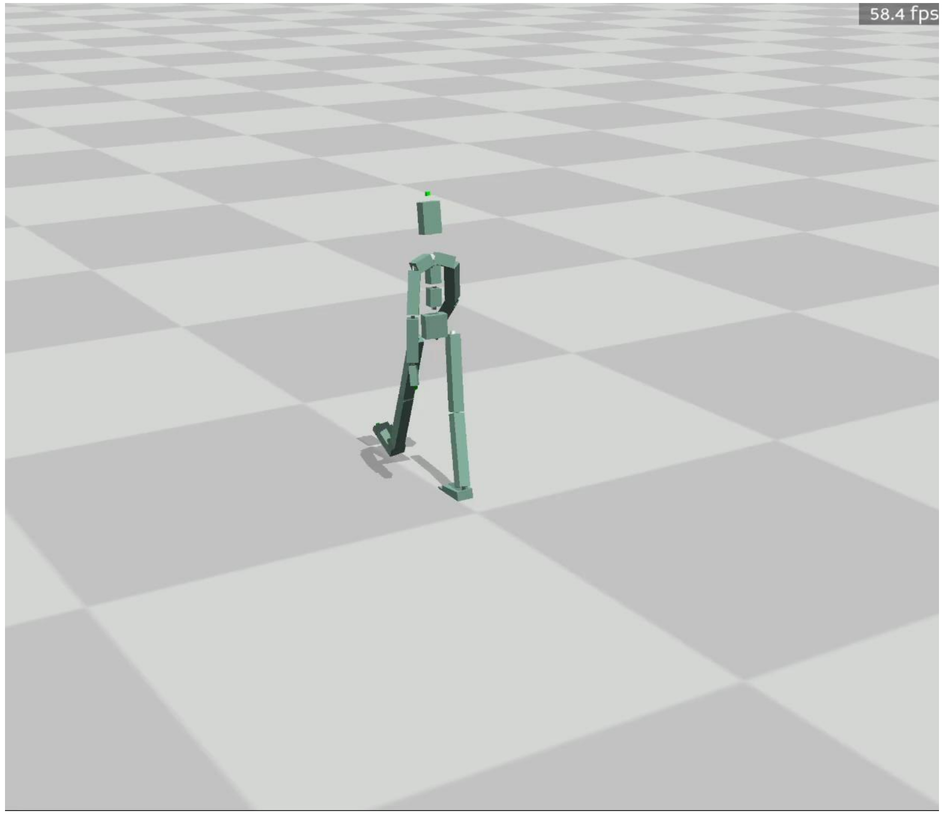
Motion Matching

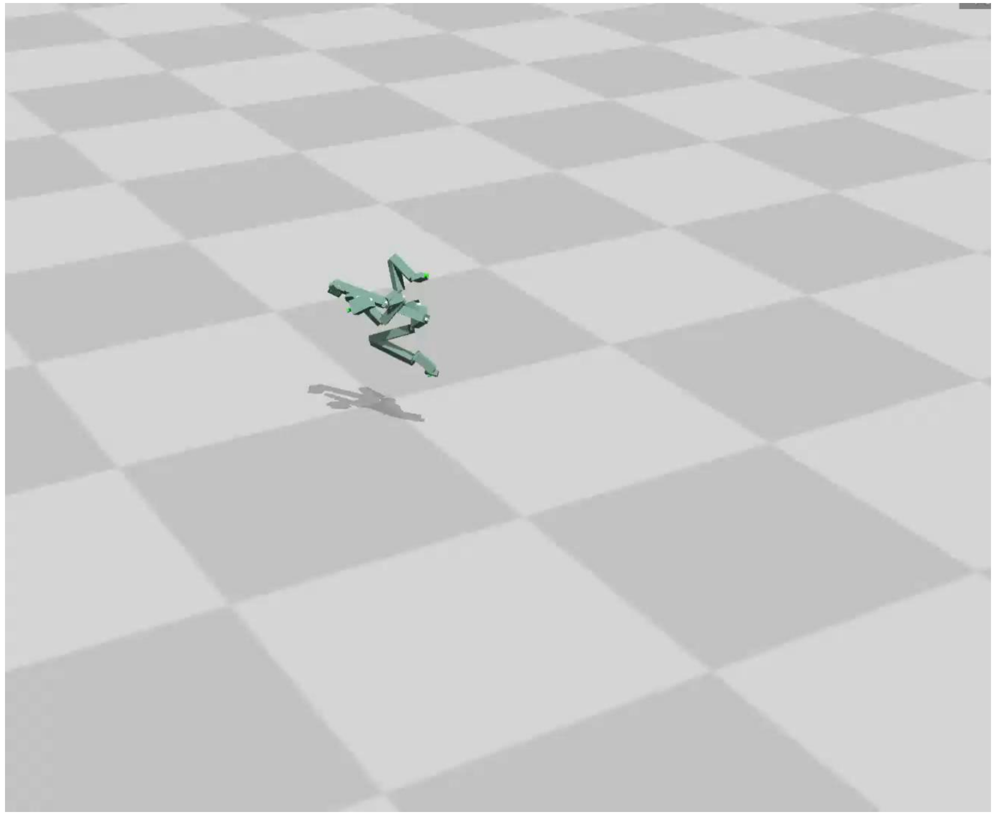
Motion Matching



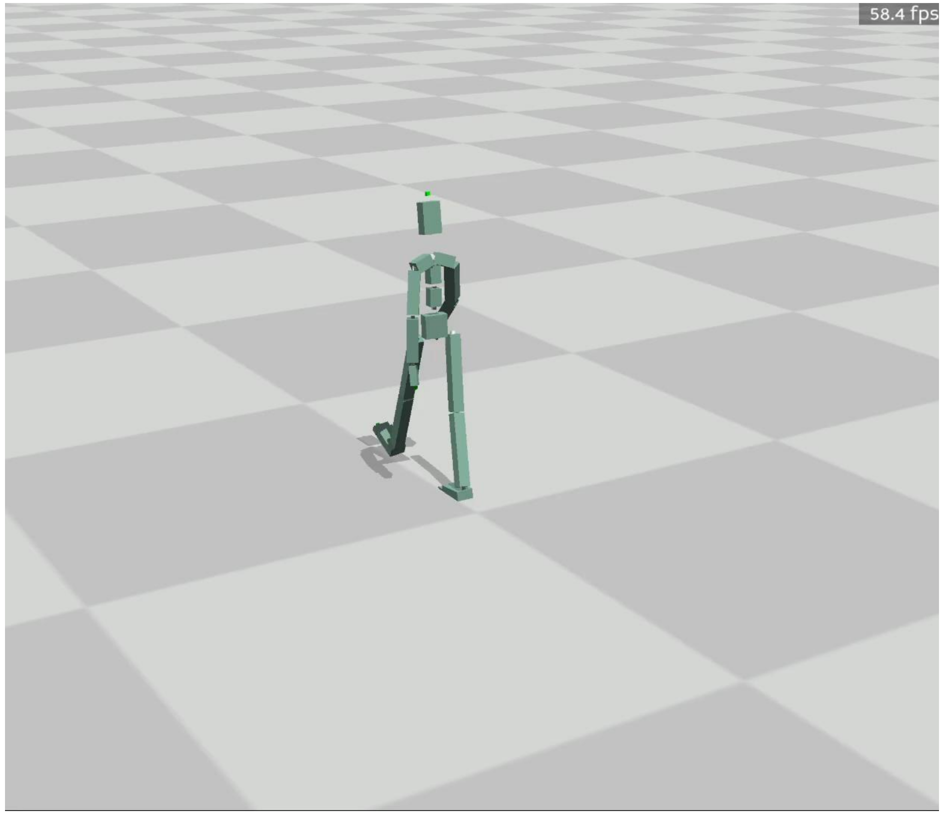
Statistical Models of Human Motion

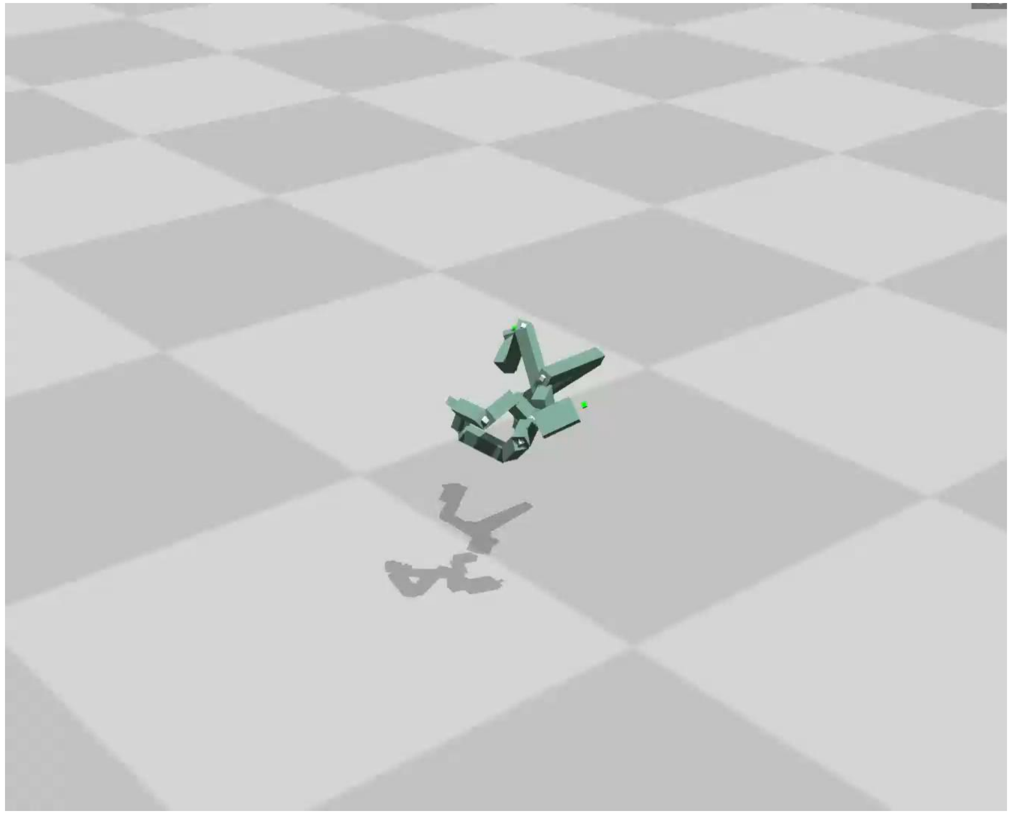
What is a natural-looking motion?



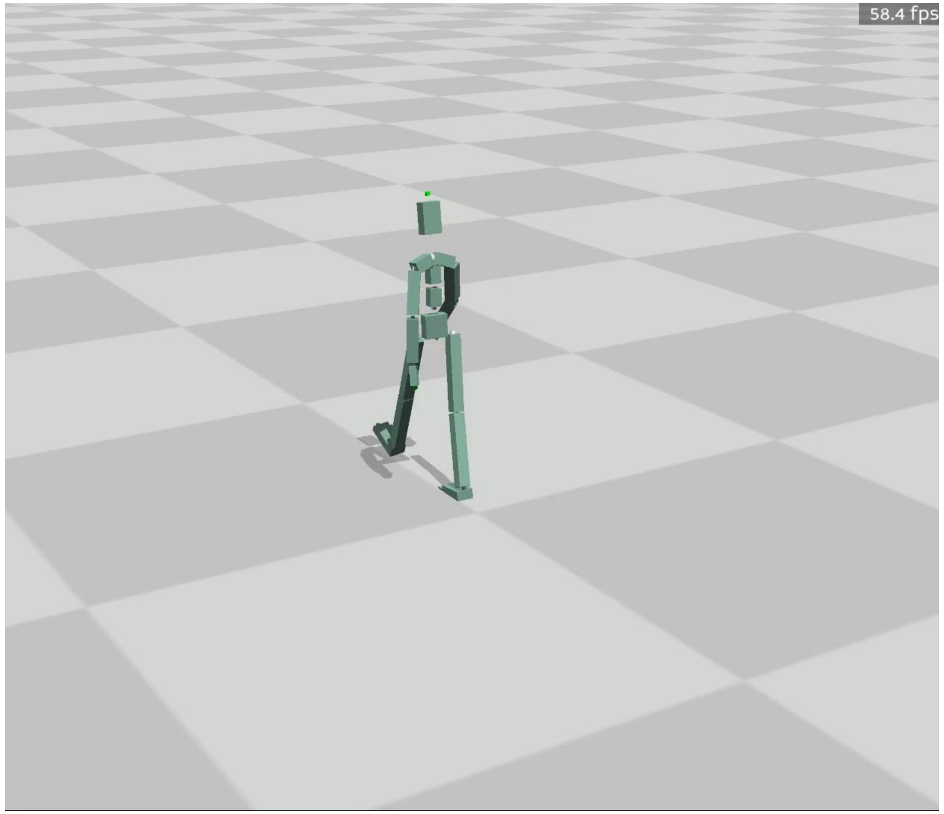


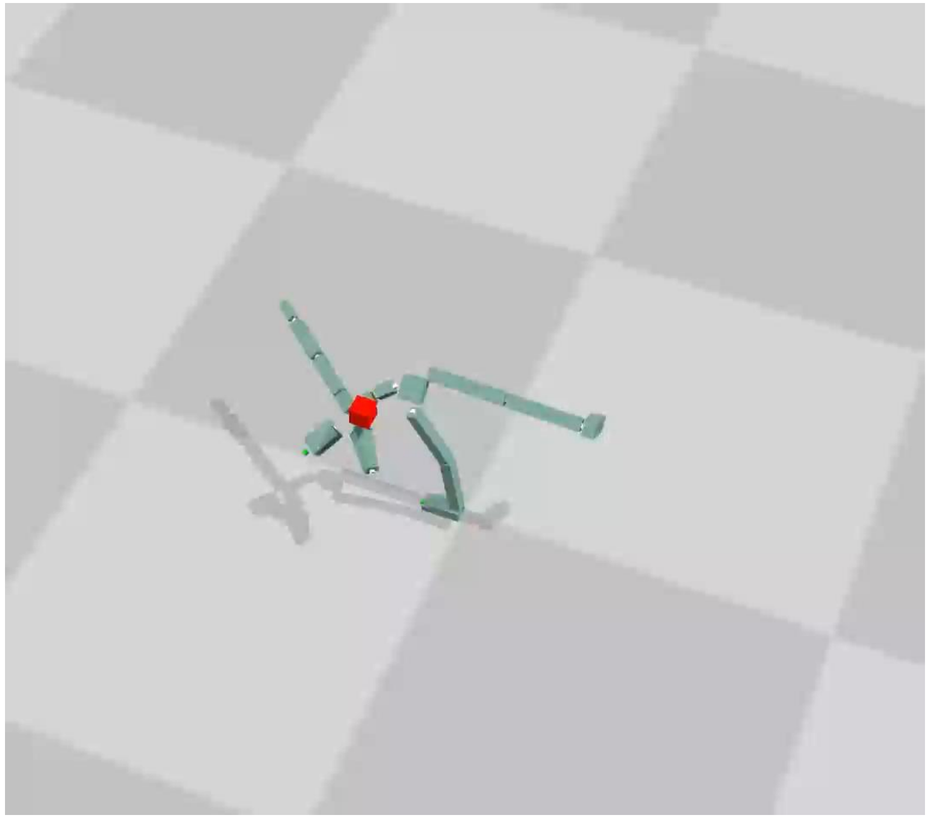
What is a natural-looking motion?



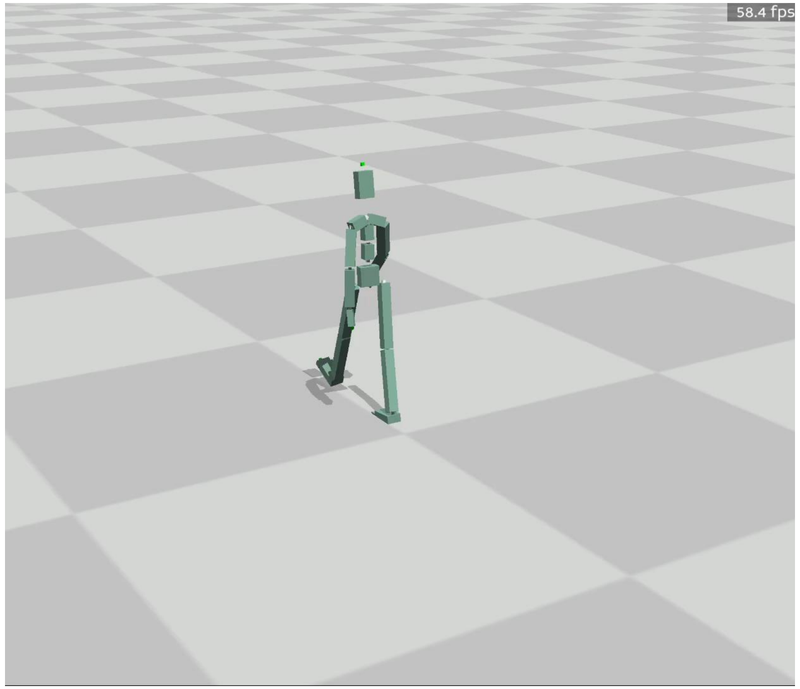


What is a natural-looking motion?



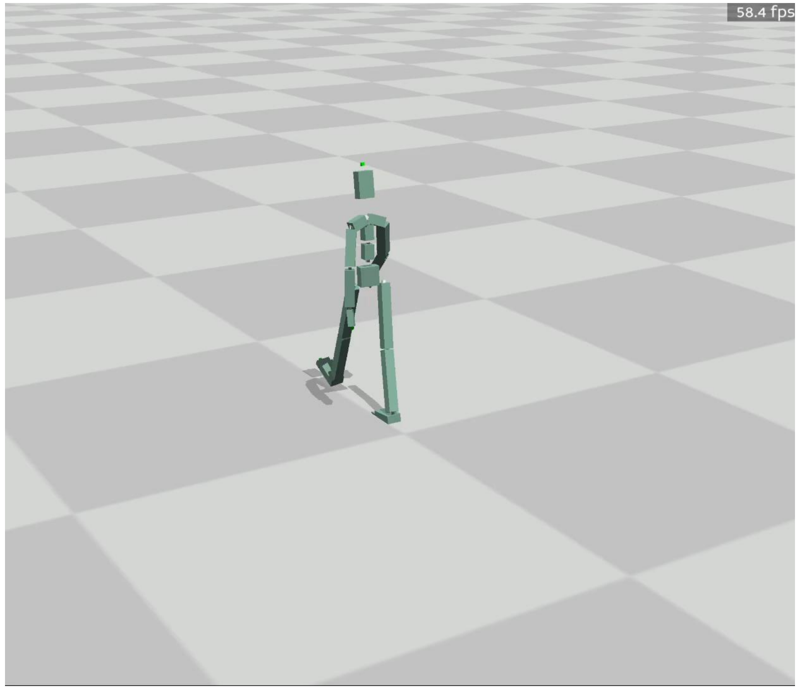


The Low-dimensionality of Human Motions

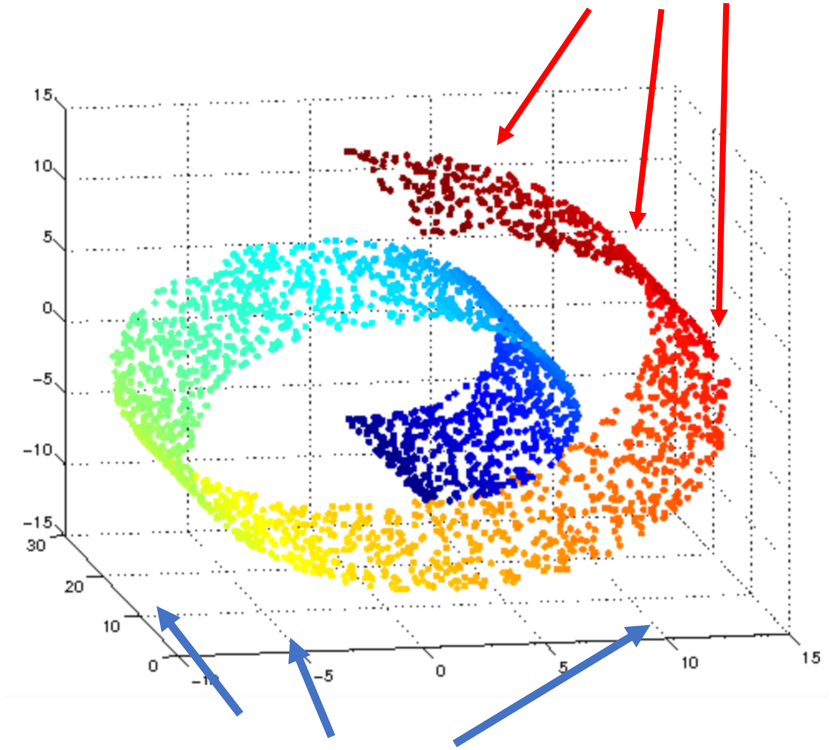


- Coordinated arm/leg movement
- Musculoskeletal structure
- Laws of physics
- ……

The Low-dimensionality of Human Motions



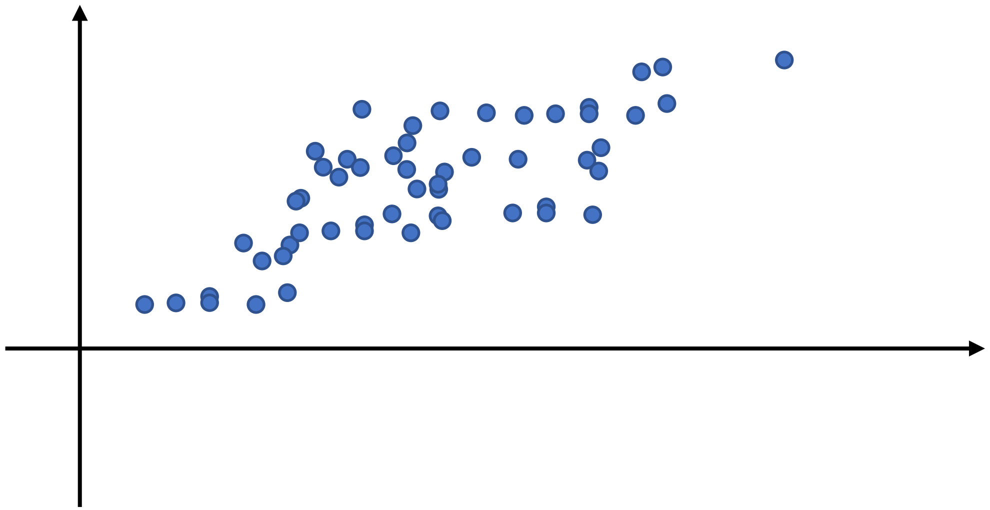
Where a natural motion locates

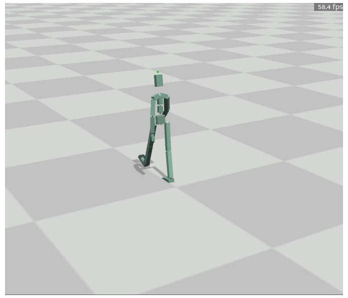


The entire “pose” space

Principal Component Analysis (PCA)

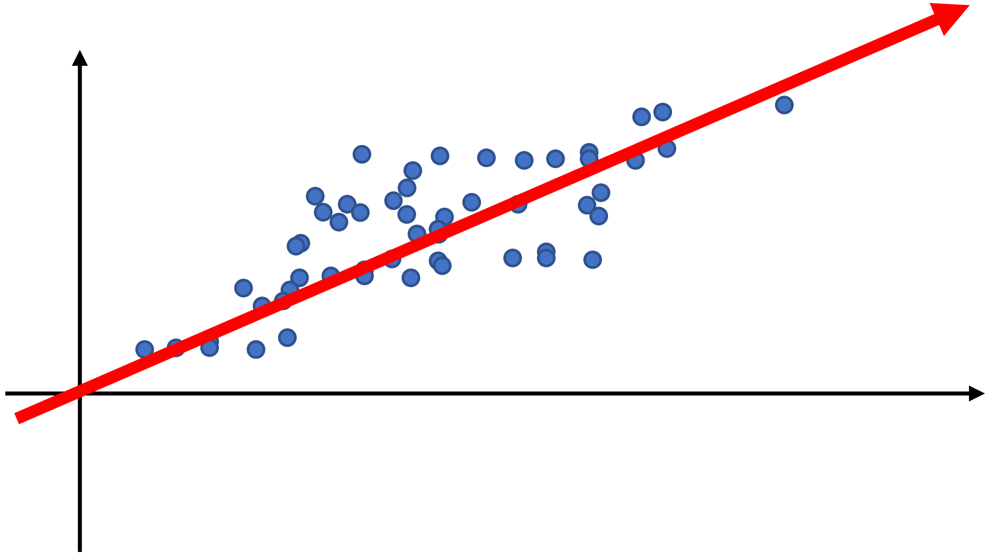
- A technique for
  - finding out the correlations among dimensions
  - dimensionality reduction



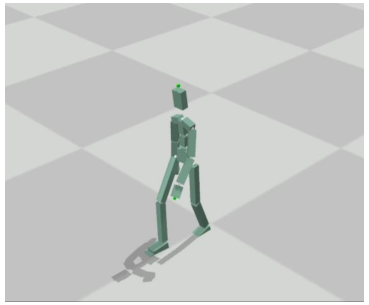


Principal Component Analysis (PCA)

- A technique for
  - finding out the correlations among dimensions
  - dimensionality reduction

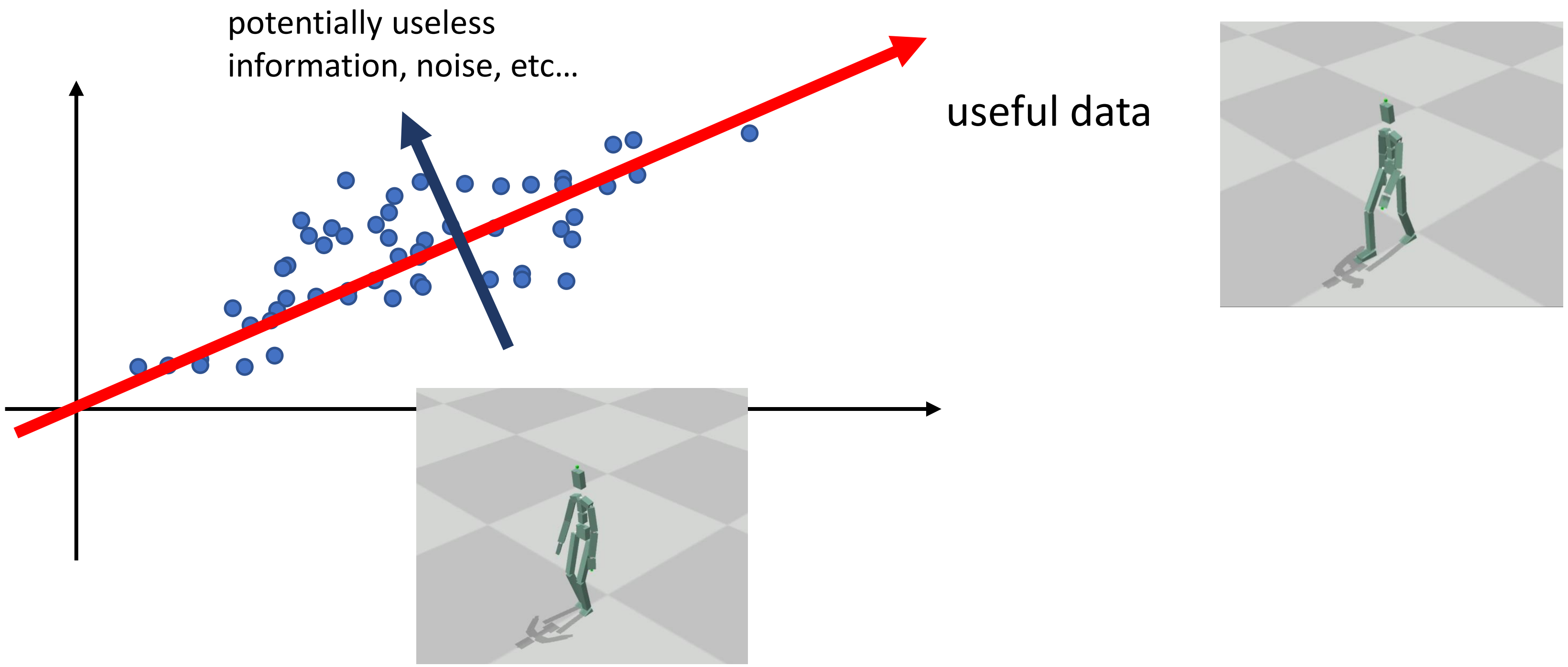


useful data

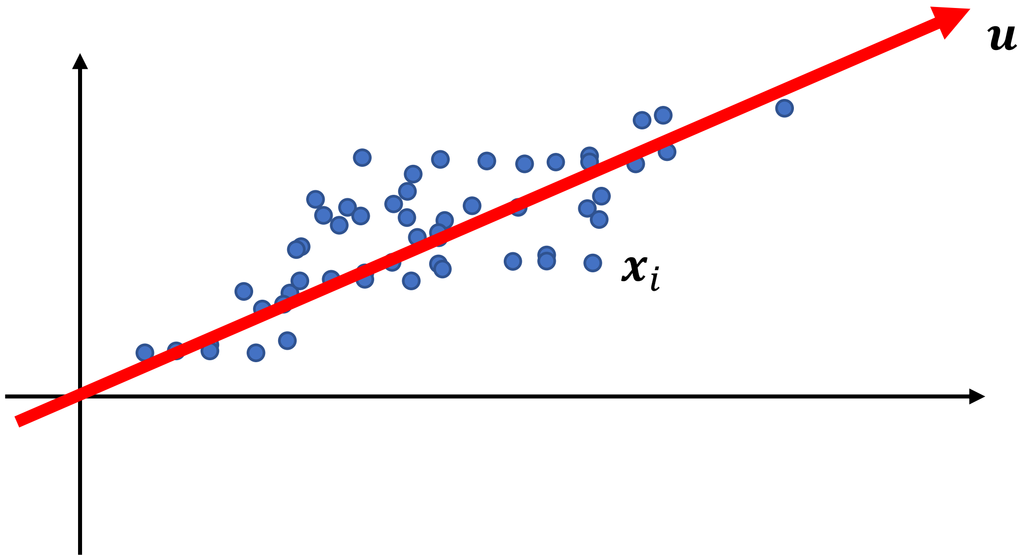


Principal Component Analysis (PCA)

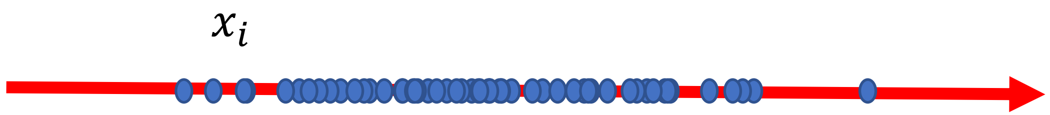
- A technique for
  - finding out the correlations among dimensions
  - dimensionality reduction



Principal Component Analysis (PCA)

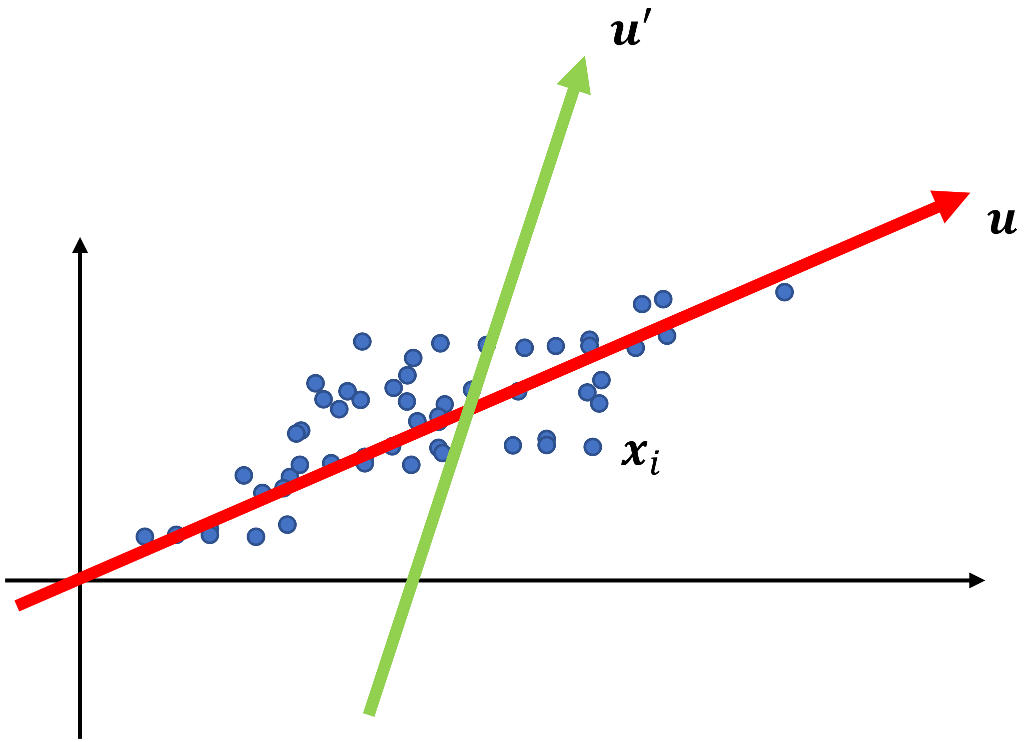


Projection of $\boldsymbol{x}_i$ on $\boldsymbol{u}$: $w_i = \boldsymbol{x}_i \cdot \boldsymbol{u}$

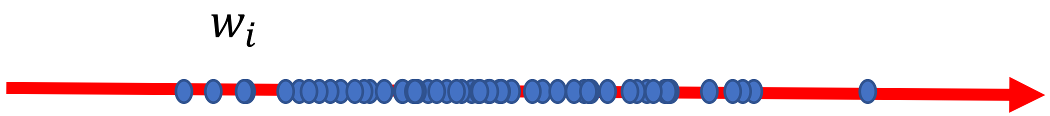


We can approximate $\boldsymbol{x}_i \approx w_i \boldsymbol{u}$

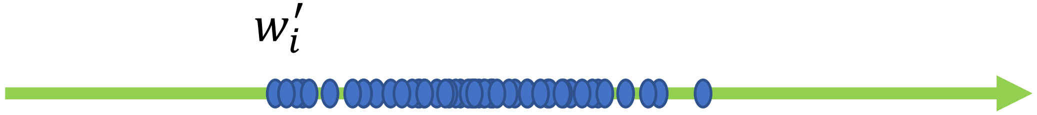
Principal Component Analysis (PCA)



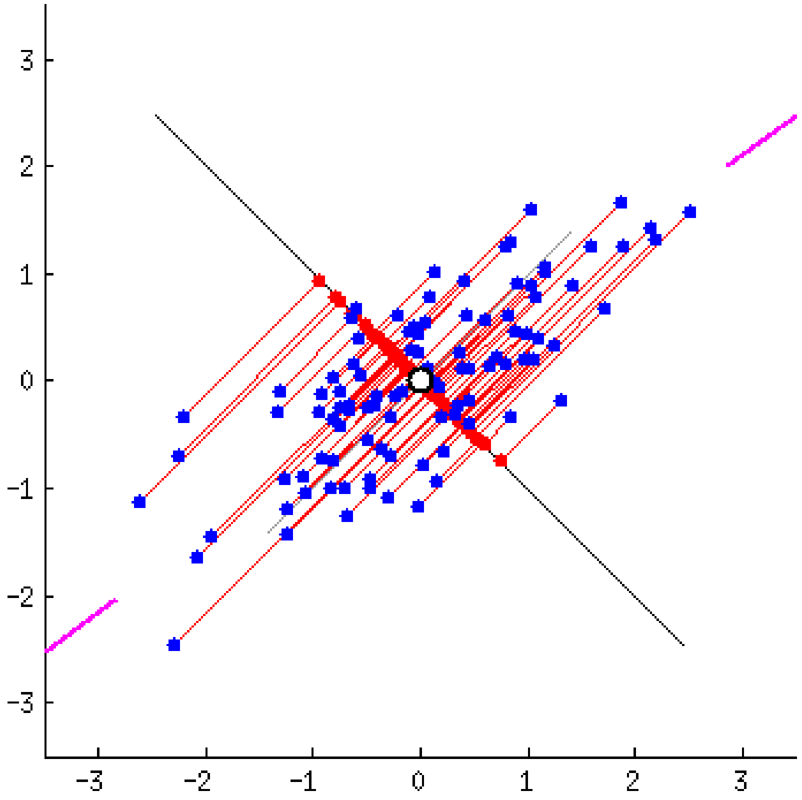
Projection of $\boldsymbol{x}_i$ on $\boldsymbol{u}$: $w_i = \boldsymbol{x}_i \cdot \boldsymbol{u}$



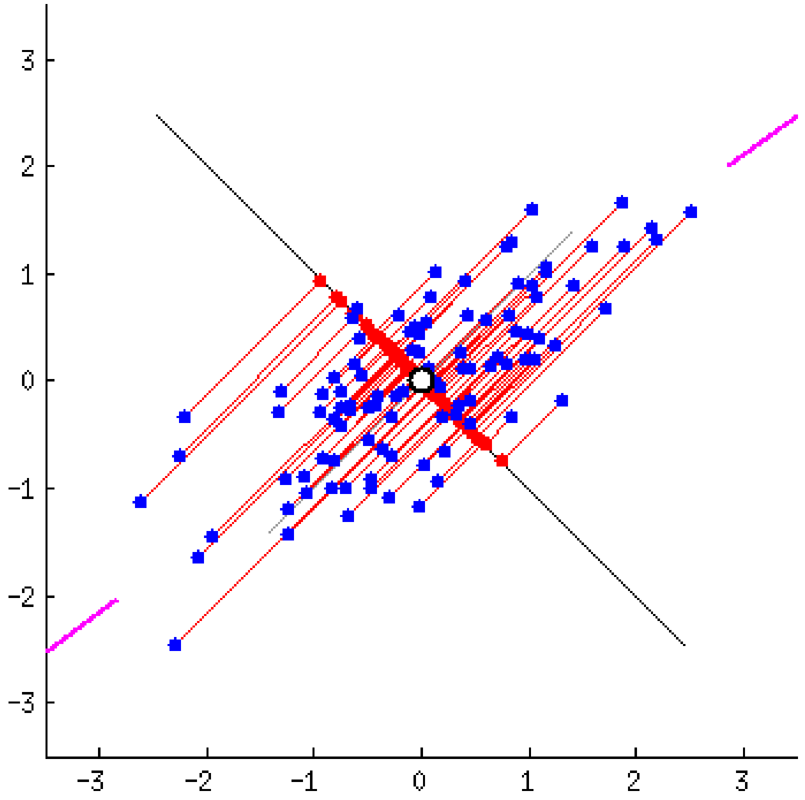
Projection of $\boldsymbol{x}_i$ on $\boldsymbol{u}'$: $w_i' = \boldsymbol{x}_i \cdot \boldsymbol{u}'$



Principal Component Analysis (PCA)



Principal Component Analysis (PCA)



Find a direction $\boldsymbol{u}$ such that $\|\boldsymbol{u}\| = 1$, and the projections of $\{\boldsymbol{x}_i\}$ on $\boldsymbol{u}$: $w_i = \boldsymbol{x}_i \cdot \boldsymbol{u}$ have the maximal variance:

$$\frac{1}{N}\sum_i (w_i - \bar{w})^2$$

Principal Component Analysis (PCA)

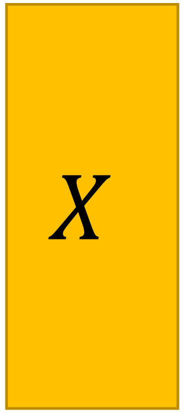
Find a direction $\boldsymbol{u}$ such that $||\boldsymbol{u}|| = 1$, and

the projections of $\{\boldsymbol{x}_i\}$ on $\boldsymbol{u}$: $w_i = \boldsymbol{x}_i \cdot \boldsymbol{u}$

have the maximal variance:

$$\frac{1}{N}\sum_i (w_i - \overline{w})^2$$

let $X = \begin{bmatrix} (\boldsymbol{x}_0 - \overline{\boldsymbol{x}})^T \\ (\boldsymbol{x}_1 - \overline{\boldsymbol{x}})^T \\ \cdots \\ (\boldsymbol{x}_N - \overline{\boldsymbol{x}})^T \end{bmatrix}$

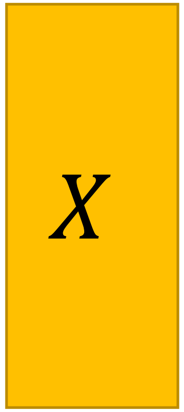


Principal Component Analysis (PCA)

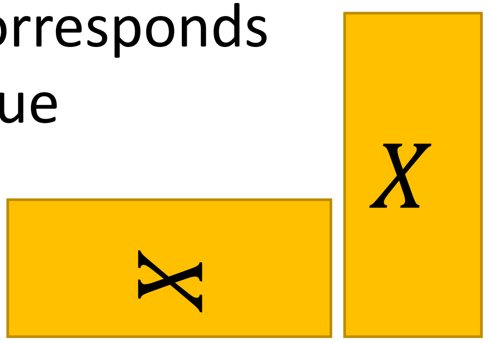
Find a direction $\boldsymbol{u}$ such that $\|\boldsymbol{u}\| = 1$, and the projections of $\{\boldsymbol{x}_i\}$ on $\boldsymbol{u}$: $w_i = \boldsymbol{x}_i \cdot \boldsymbol{u}$ have the maximal variance:

$$\frac{1}{N}\sum_i (w_i - \overline{w})^2$$

let $X = \begin{bmatrix} (\boldsymbol{x}_0 - \overline{\boldsymbol{x}})^T \\ (\boldsymbol{x}_1 - \overline{\boldsymbol{x}})^T \\ \cdots \\ (\boldsymbol{x}_N - \overline{\boldsymbol{x}})^T \end{bmatrix}$



It can be proved that $\boldsymbol{u}$ is an **eigenvector** of $X^T X$ corresponds to the largest eigenvalue



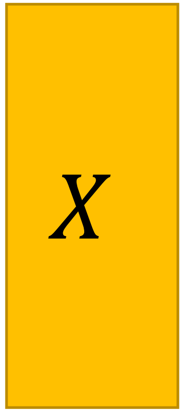
Principal Component Analysis (PCA)

Find a direction $\boldsymbol{u}$ such that $||\boldsymbol{u}|| = 1$, and the projections of $\{\boldsymbol{x}_i\}$ on $\boldsymbol{u}$: $w_i = \boldsymbol{x}_i \cdot \boldsymbol{u}$ have the maximal variance:

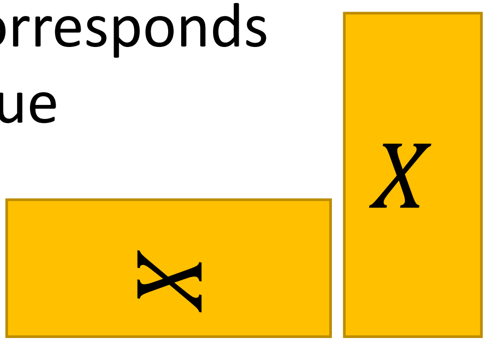
$$\frac{1}{N}\sum_i (w_i - \overline{w})^2$$

Note: we can approximate $\boldsymbol{x}_i \approx \overline{\boldsymbol{x}} + w_i \boldsymbol{u}$

let $X = \begin{bmatrix} (\boldsymbol{x}_0 - \overline{\boldsymbol{x}})^T \\ (\boldsymbol{x}_1 - \overline{\boldsymbol{x}})^T \\ \cdots \\ (\boldsymbol{x}_N - \overline{\boldsymbol{x}})^T \end{bmatrix}$



It can be proved that $\boldsymbol{u}$ is an **eigenvector** of $X^T X$ corresponds to the largest eigenvalue



PCA in a Nutshell

- Given a dataset $\{\boldsymbol{x}_i\}, \boldsymbol{x}_i \in \mathbb{R}^N$, then PCA gives

$$\boldsymbol{x}_i = \bar{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k} \boldsymbol{u}_{\boldsymbol{k}}$$

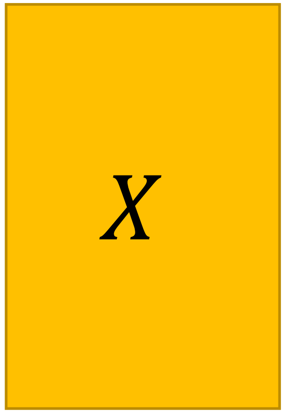
- $\boldsymbol{u}_{\boldsymbol{k}}$ is the k-th principal component
  - A direction in $\mathbb{R}^N$ along which the projection of $\{\boldsymbol{x}_i\}$ has the k-th maximal **variance**
- $w_{i,k} = (\boldsymbol{x}_i - \bar{\boldsymbol{x}}) \cdot \boldsymbol{u}_k$ is the score of $\boldsymbol{x}_i$ on $\boldsymbol{u}_k$

PCA in a Nutshell

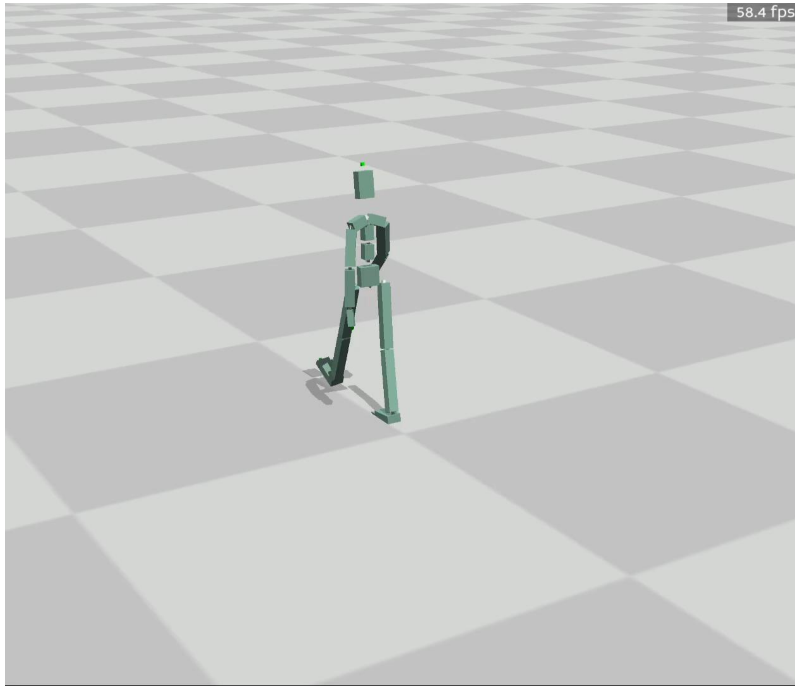
- Given a dataset $\{\boldsymbol{x}_i\}, \boldsymbol{x}_i \in \mathbb{R}^N$, the PCA can be computed by apply **eigen decomposition** on the covariance matrix

$$\Sigma = X^T X = U \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_N^2 \end{bmatrix} U^T$$

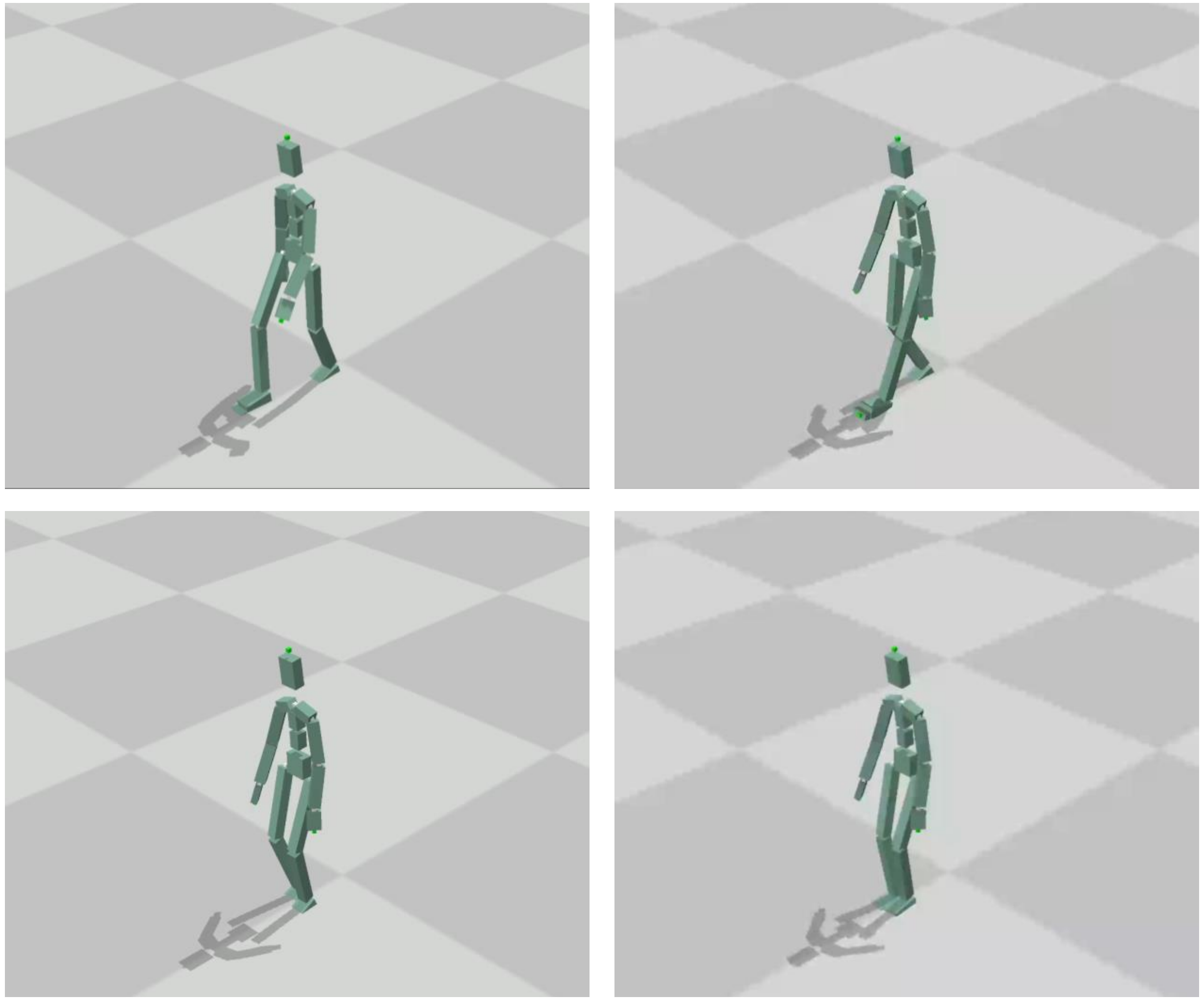
- $X = [\boldsymbol{x}_0 - \overline{\boldsymbol{x}}, \boldsymbol{x}_1 - \overline{\boldsymbol{x}}, \dots, \boldsymbol{x}_N - \overline{\boldsymbol{x}}]^T$
- $\sigma_i \geq \sigma_j \geq 0$ when $i < j$, corresponds to the Explained Variance
- $U = [\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_N]$



PCA of Walking



$\boldsymbol{x}_i$: joint rotations



PCA in a Nutshell

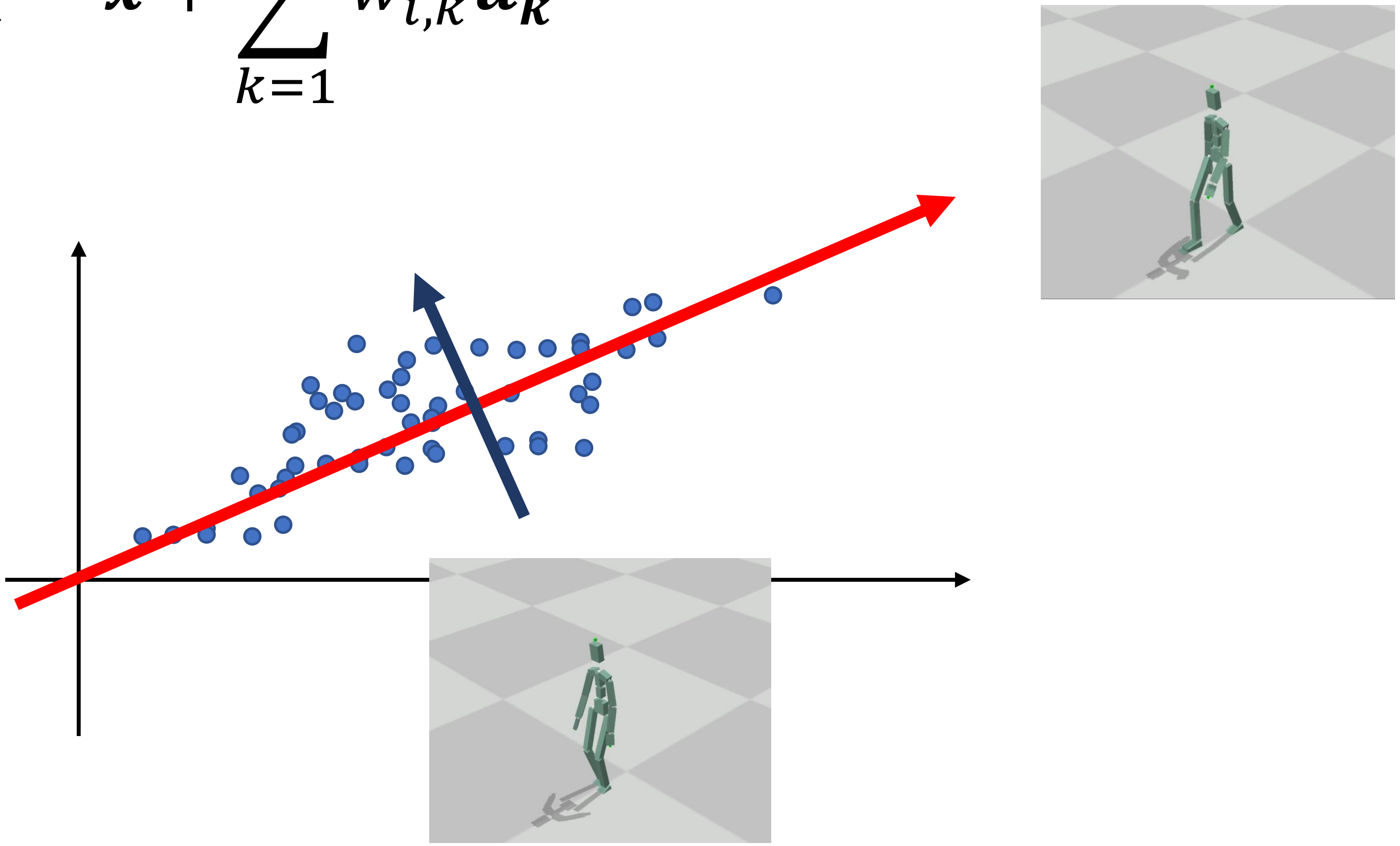
- Given a dataset $\{\boldsymbol{x}_i\}, \boldsymbol{x}_i \in \mathbb{R}^N$, then PCA gives

$$\boldsymbol{x}_i = \overline{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k} \boldsymbol{u}_{\boldsymbol{k}}$$

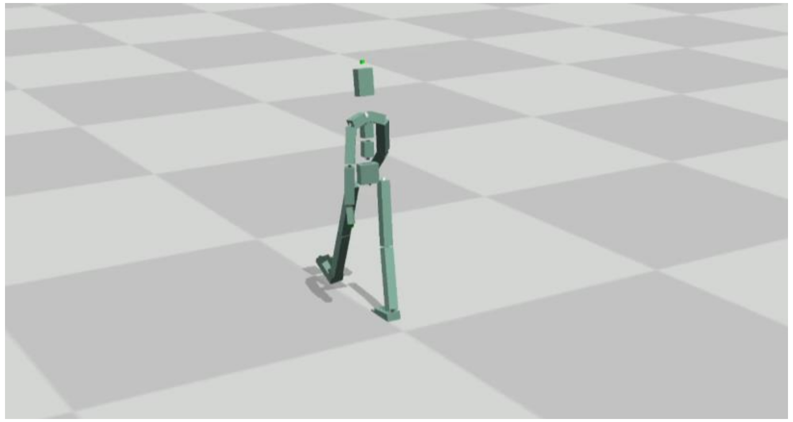
- $\boldsymbol{u}_{\boldsymbol{k}}$ is the k-th principal component
  - A direction in $\mathbb{R}^N$ along which the projection of $\{\boldsymbol{x}_i\}$ has the k-th maximal **variance**
- $w_{i,k} = (\boldsymbol{x}_i - \overline{\boldsymbol{x}}) \cdot \boldsymbol{u}_k$ is the score of $\boldsymbol{x}_i$ on $\boldsymbol{u}_k$

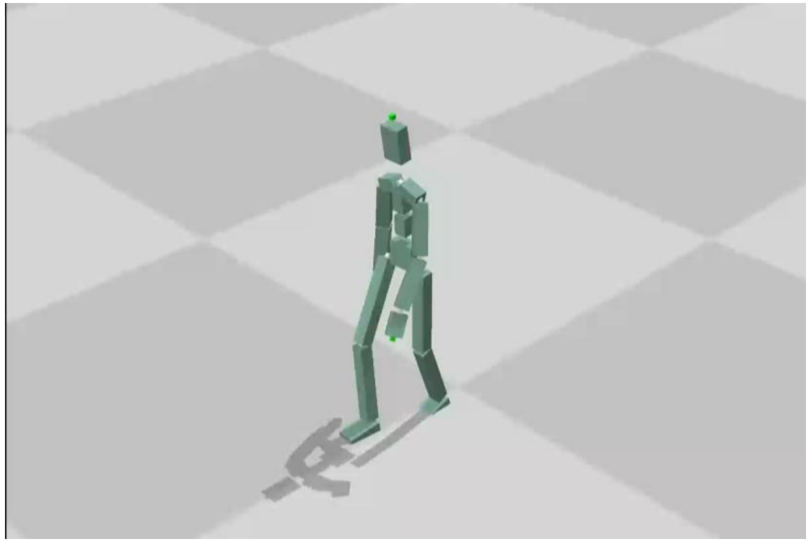
PCA in a Nutshell

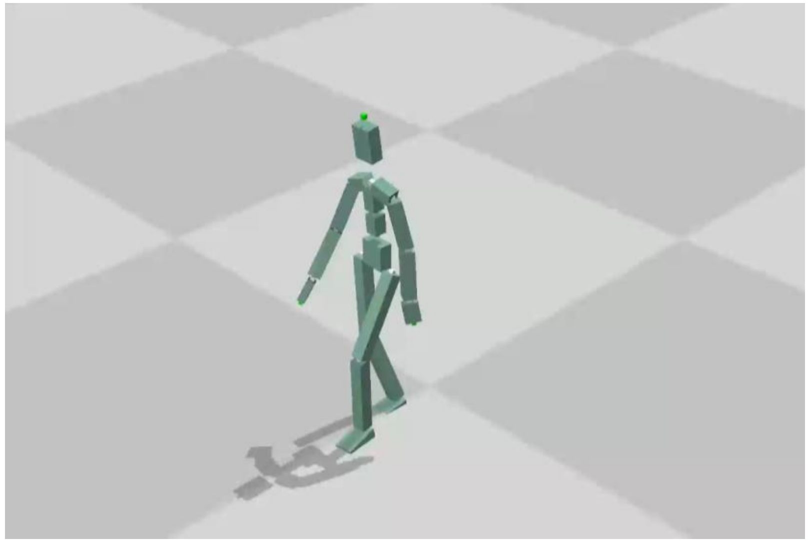
$$\boldsymbol{x}_i = \overline{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k}\boldsymbol{u}_{\boldsymbol{k}}$$

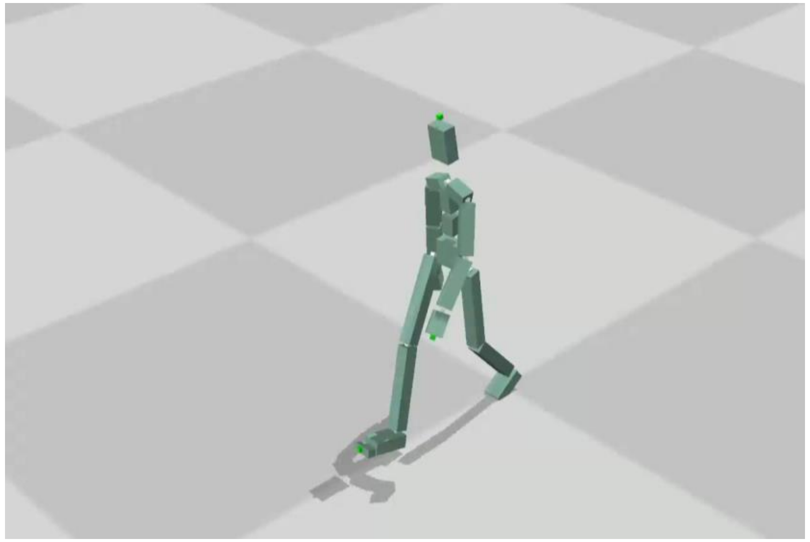


PCA of Walking

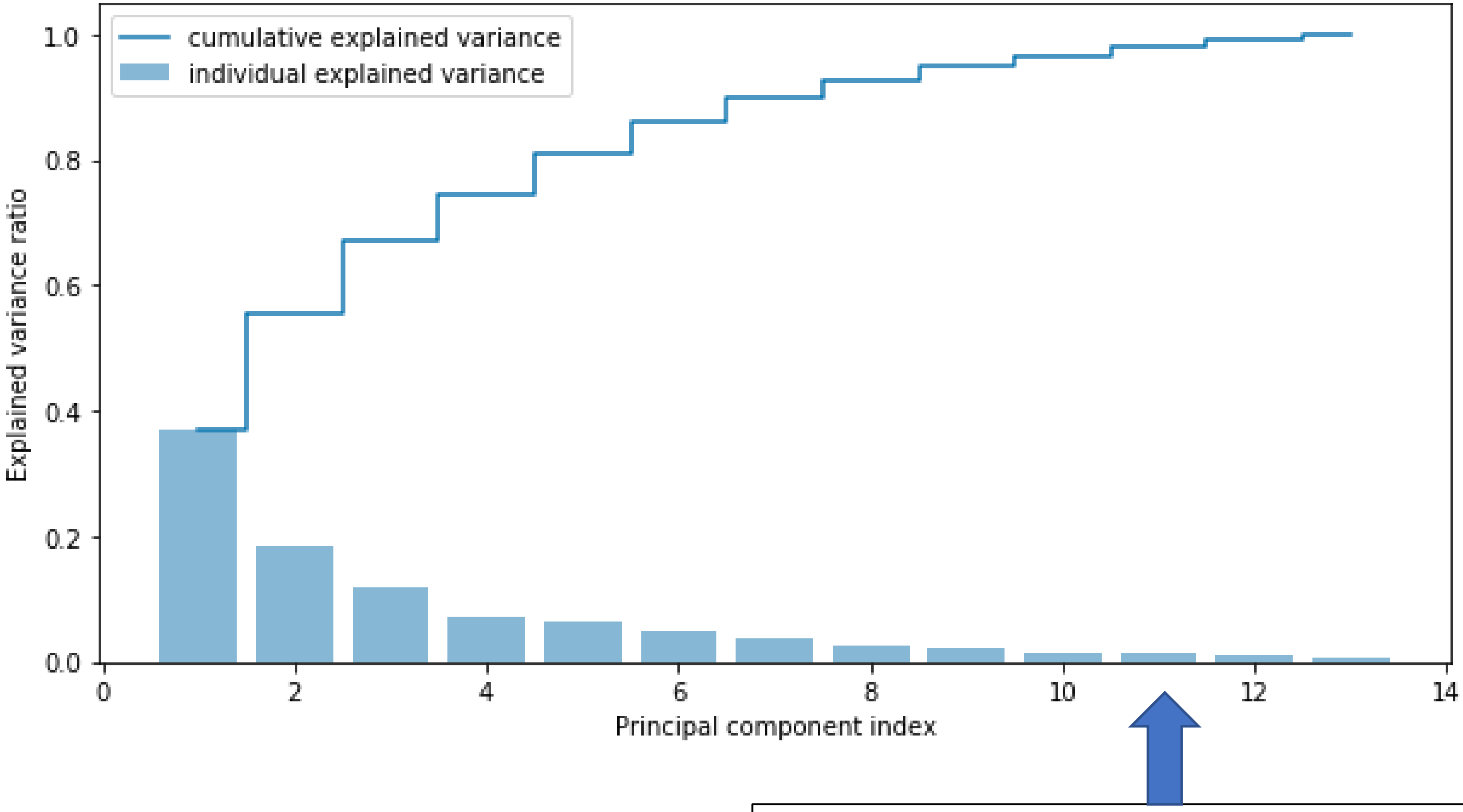








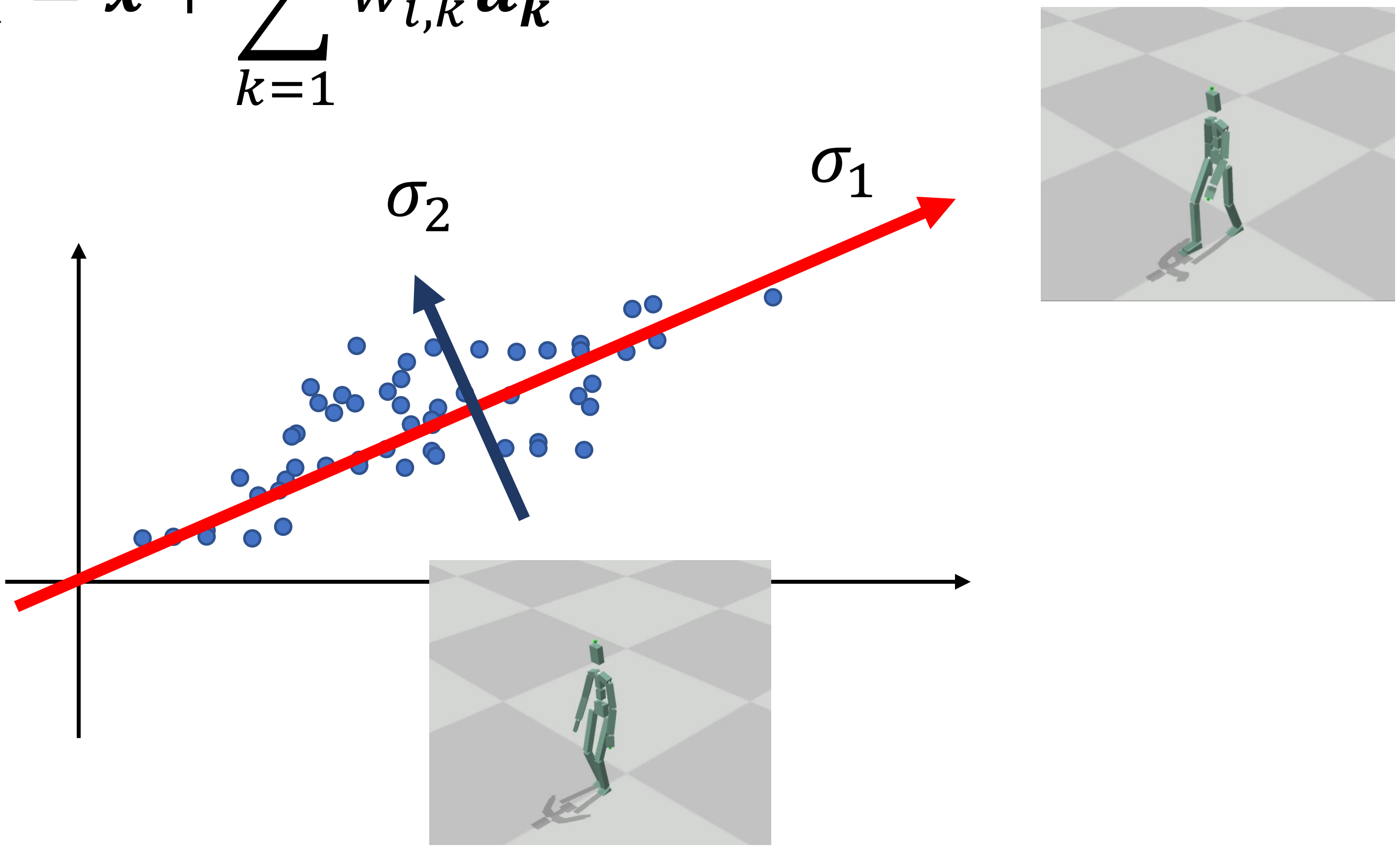
PCA in a Nutshell



Choose n principal components that explains enough (e.g. 95%) of the variance

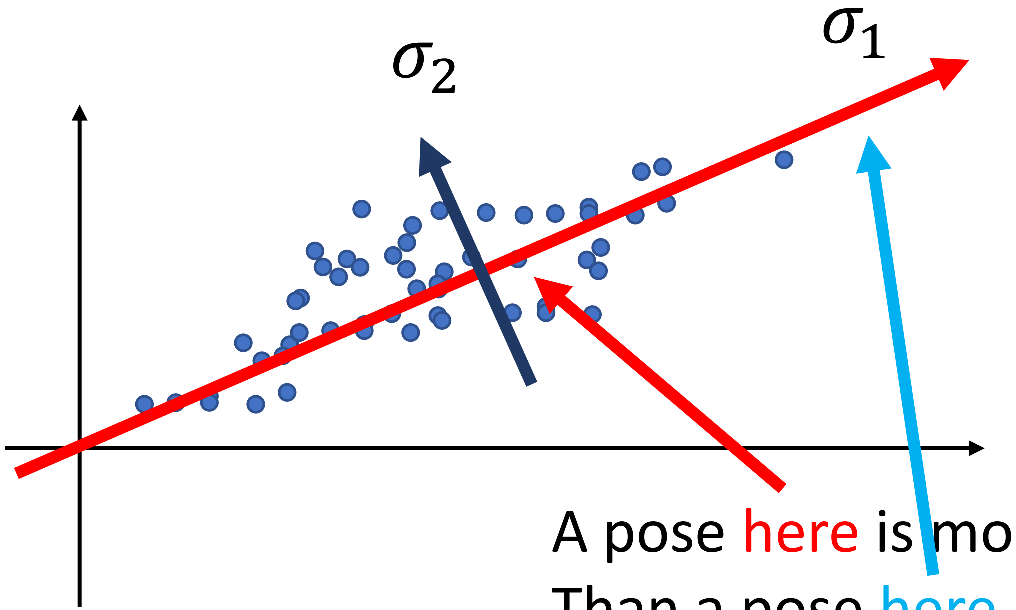
PCA in a Nutshell

$$\boldsymbol{x}_i = \overline{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k}\boldsymbol{u}_{\boldsymbol{k}}$$



PCA in a Nutshell

$$\boldsymbol{x}_i = \overline{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k} \boldsymbol{u}_{\boldsymbol{k}}$$



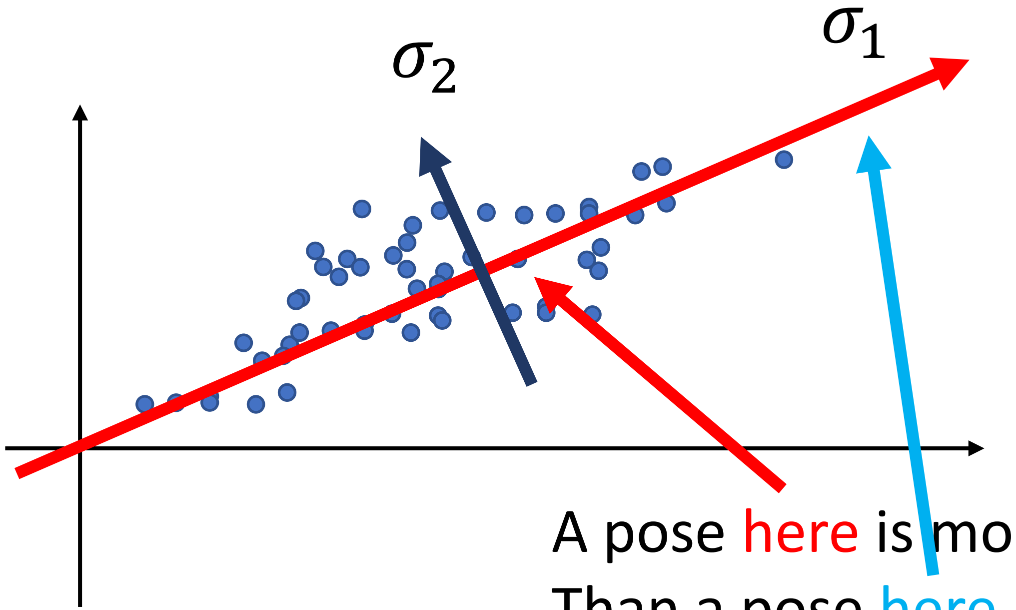
A pose here is more likely to be natural
Than a pose here

PCA in a Nutshell

$$\boldsymbol{x}_i = \bar{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k}\boldsymbol{u}_{\boldsymbol{k}}$$

a pose $\boldsymbol{x}_i$ with smaller $\sum_k \left(\frac{w_{i,k}}{\sigma_k}\right)^2$

is more likely to be a good pose



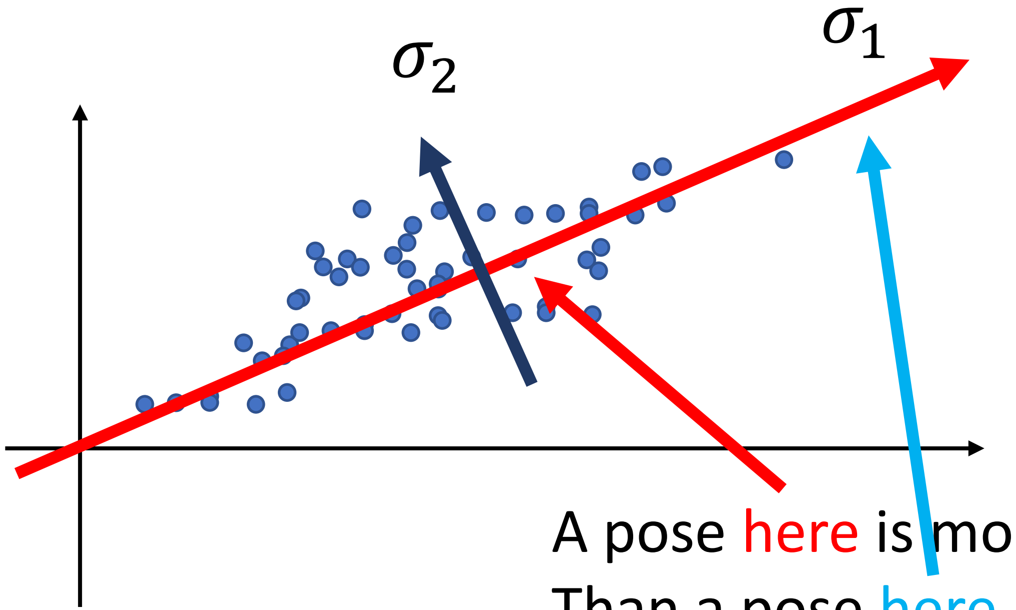
A pose here is more likely to be natural

Than a pose here

PCA in a Nutshell

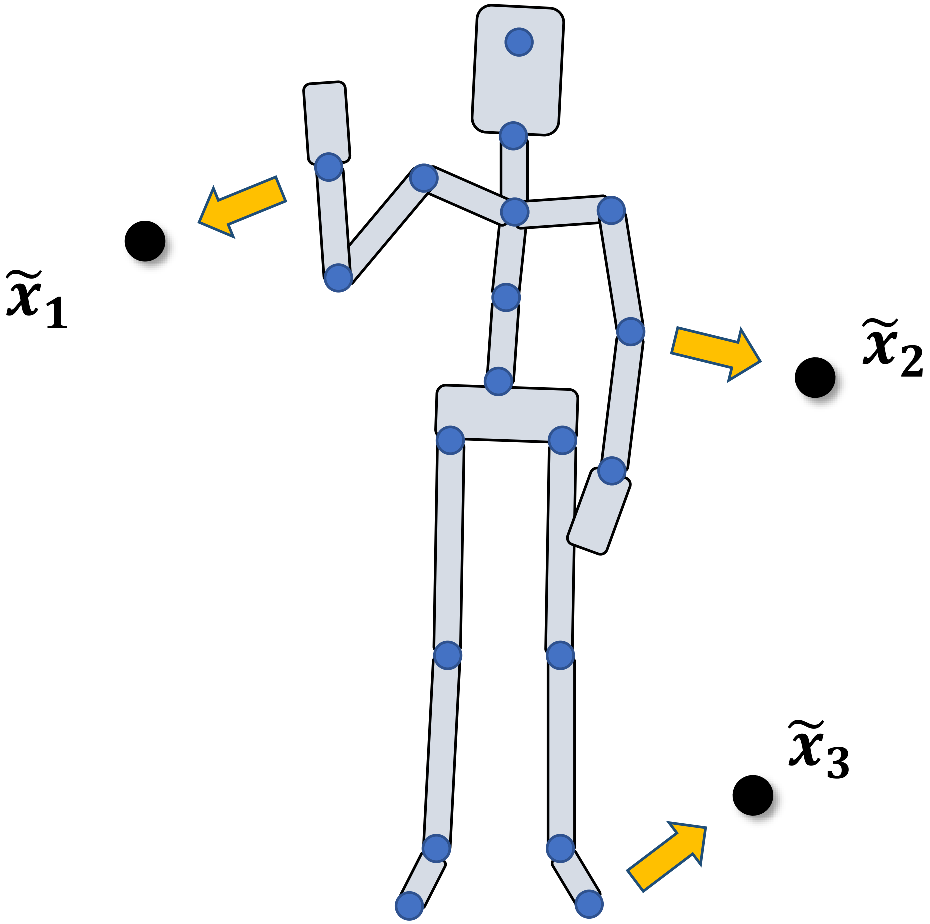
$$\boldsymbol{x} = \overline{\boldsymbol{x}} + \sum_{k=1}^{n} w_k \boldsymbol{u_k}$$

a pose $\boldsymbol{x}$ with smaller $\sum_k \frac{\left((\boldsymbol{x}-\overline{\boldsymbol{x}})\cdot\boldsymbol{u}_k\right)^2}{\sigma_k^2}$ is more likely to be a good pose



A pose here is more likely to be natural
Than a pose here

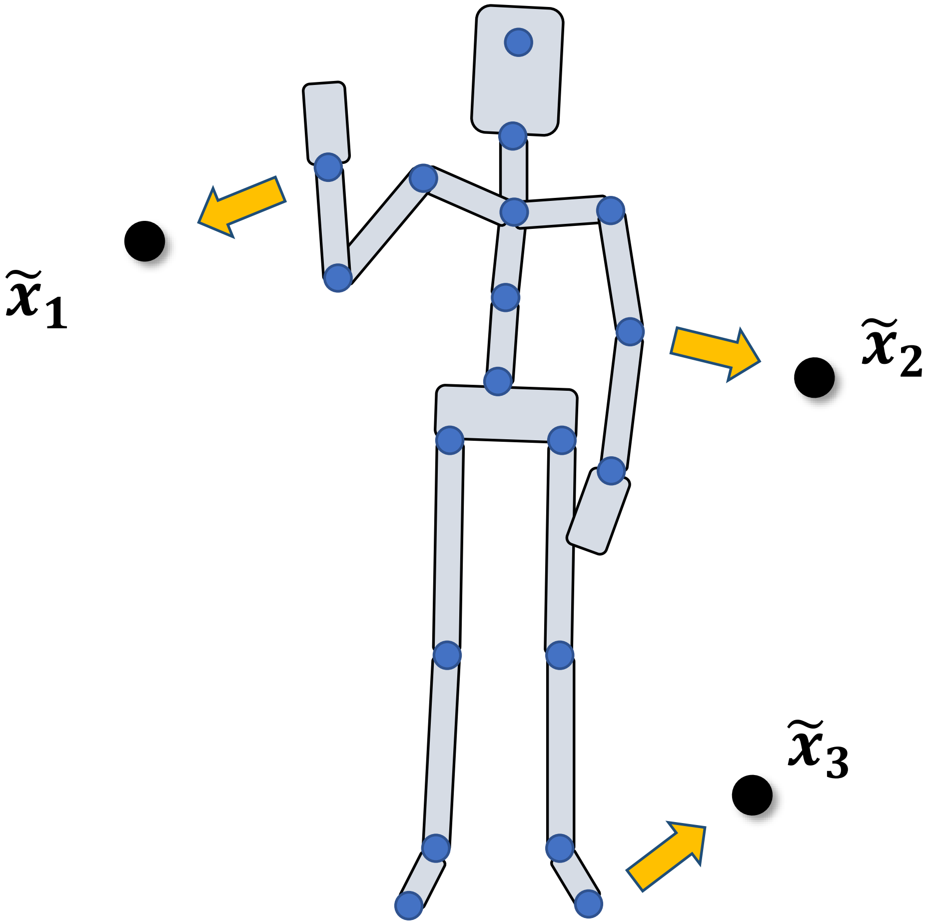
Character IK



$$F(\theta) = \frac{1}{2}\sum_i \|f_i(\boldsymbol{\theta}) - \tilde{\boldsymbol{x}}_{\boldsymbol{i}}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2$$

$$\boldsymbol{\theta} = (\boldsymbol{t}_0, R_0, R_1, R_2, \dots \dots)$$

Character IK with a Reference Pose

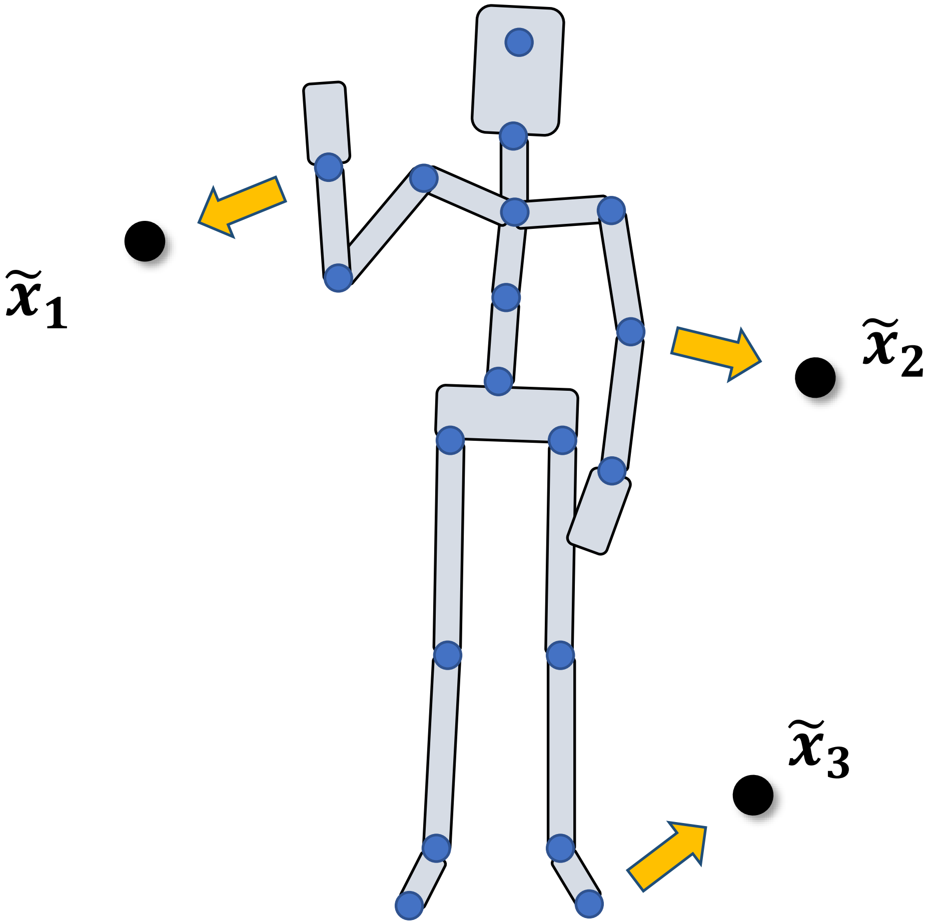


$$F(\theta) = \frac{1}{2}\sum_i \|f_i(\boldsymbol{\theta}) - \tilde{\boldsymbol{x}}_i\|_2^2$$

$$+\frac{\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2$$

$$\boldsymbol{\theta} = (\boldsymbol{t}_0, R_0, R_1, R_2, \dots \dots)$$

Character IK with a Motion Prior



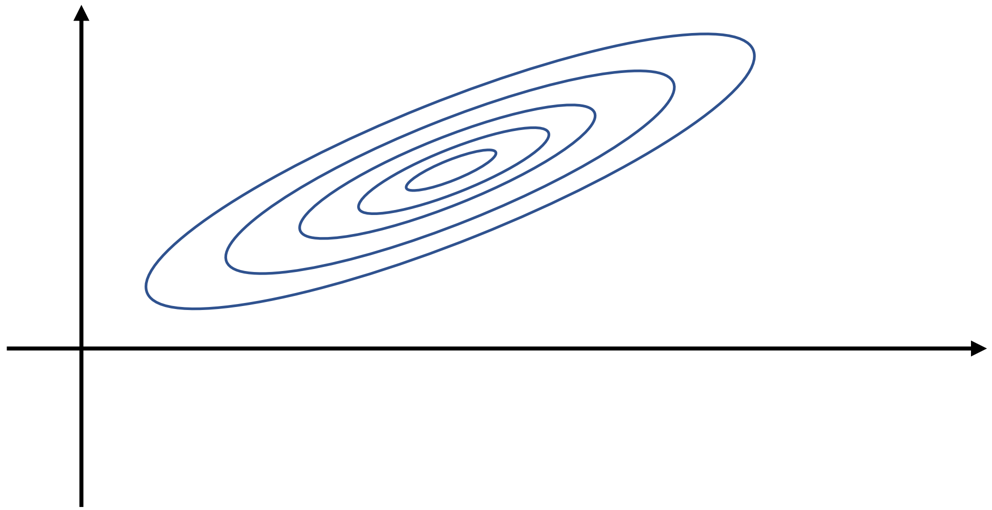
$$F(\theta) = \frac{1}{2}\sum_i \|f_i(\boldsymbol{\theta}) - \tilde{\boldsymbol{x}}_i\|_2^2$$

$$+\frac{w}{2}\sum_k \left(\frac{(\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}) \cdot \boldsymbol{u}_k}{\sigma_k}\right)^2$$

$$\boldsymbol{\theta} = (\boldsymbol{t}_0, R_0, R_1, R_2, \dots \dots)$$

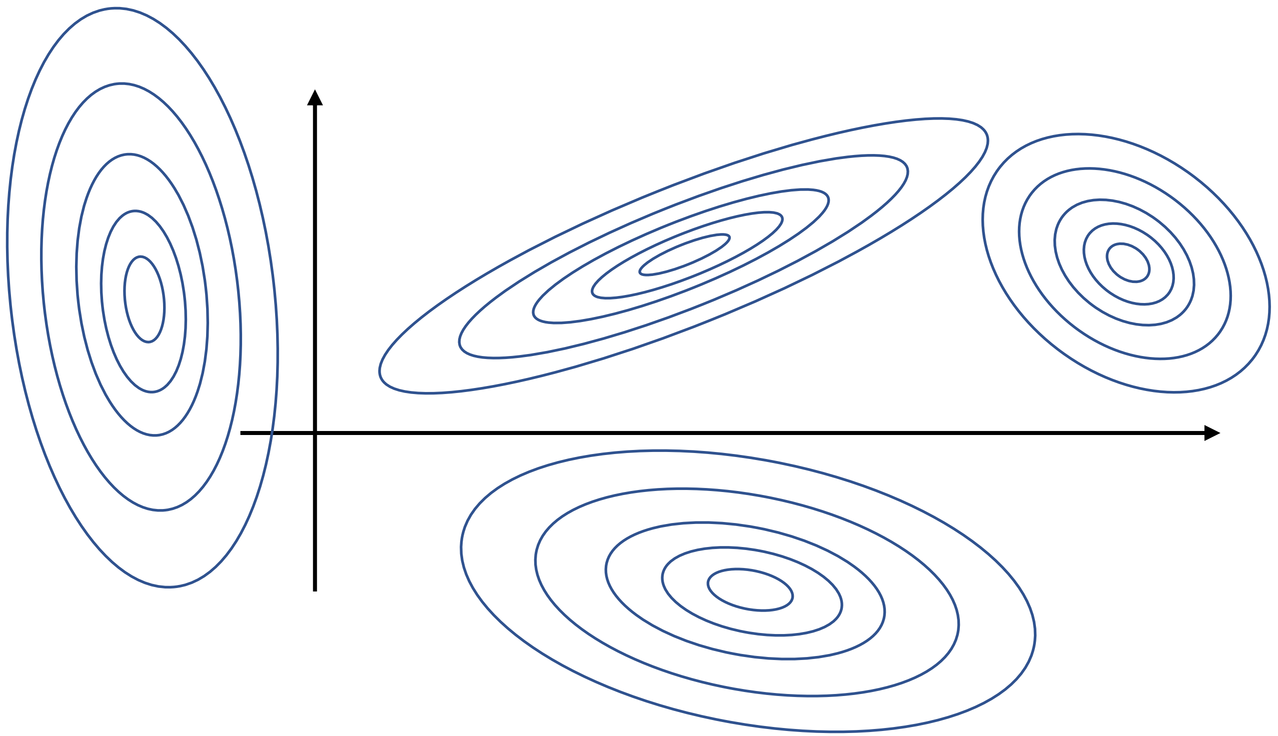
Data Distribution

$p(\boldsymbol{x})$: probability that $\boldsymbol{x}$ is a natural pose



Data Distribution

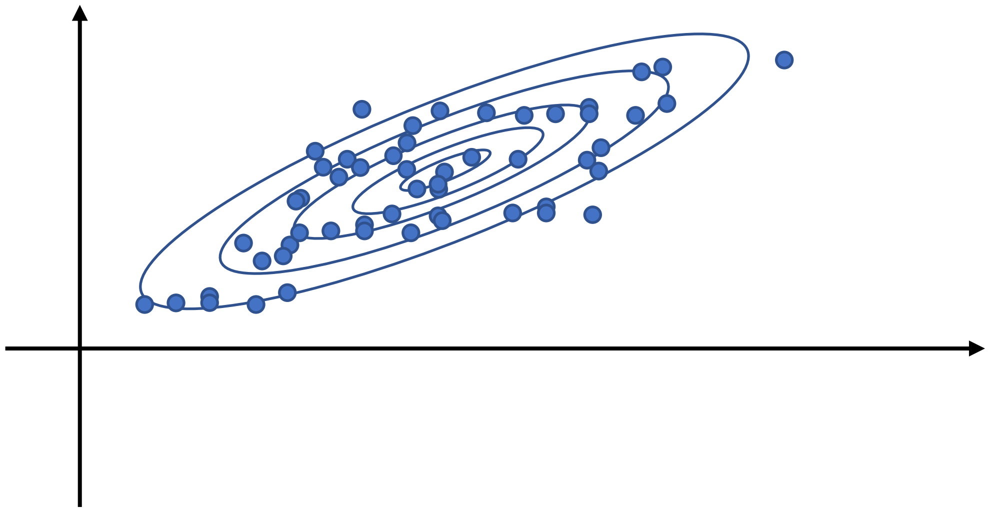
$p(\boldsymbol{x})$: probability that $\boldsymbol{x}$ is a natural pose



Data Distribution

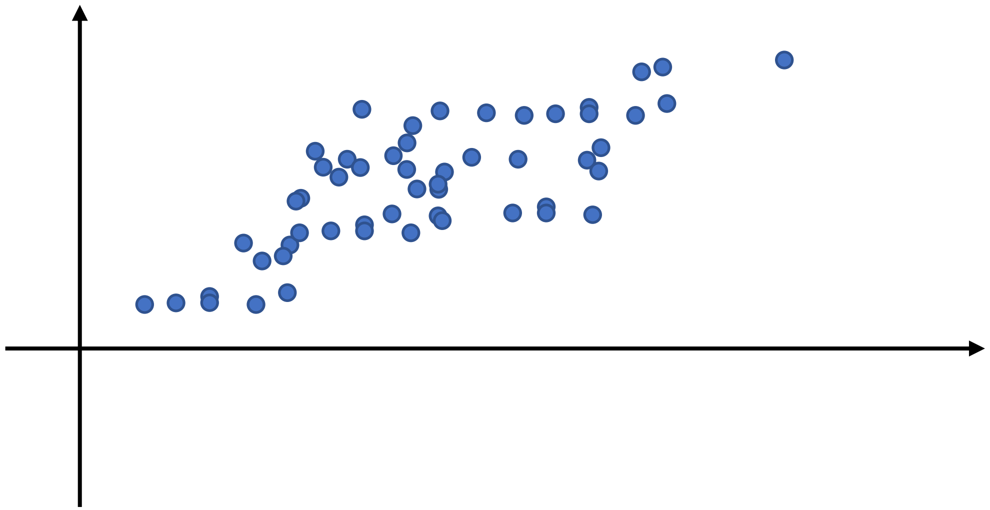
$p(\boldsymbol{x})$: probability that $\boldsymbol{x}$ is a natural pose

a set of data points $\{\boldsymbol{x}_i\} \sim p(\boldsymbol{x})$



Data Distribution

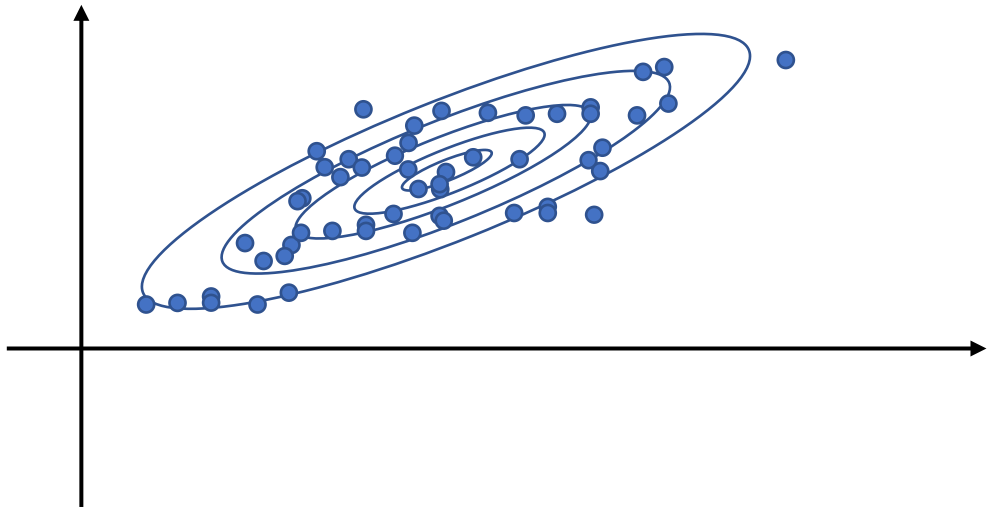
Given a dataset of mocap poses $\{\boldsymbol{x}_i\}$



Data Distribution

Given a dataset of mocap poses $\{\boldsymbol{x}_i\}$

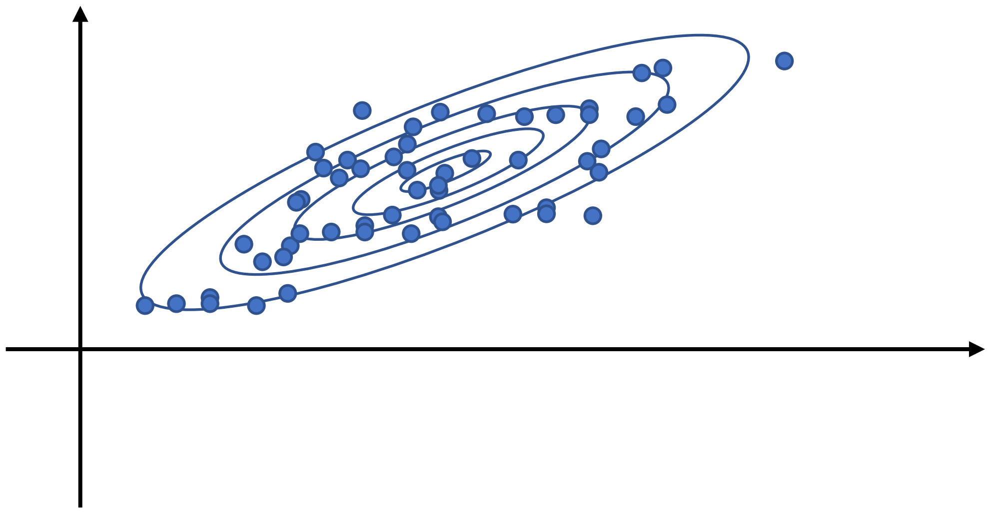
How to find $p(\boldsymbol{x})$?



Gaussian Distribution

Dataset $\{\boldsymbol{x}_i\}$

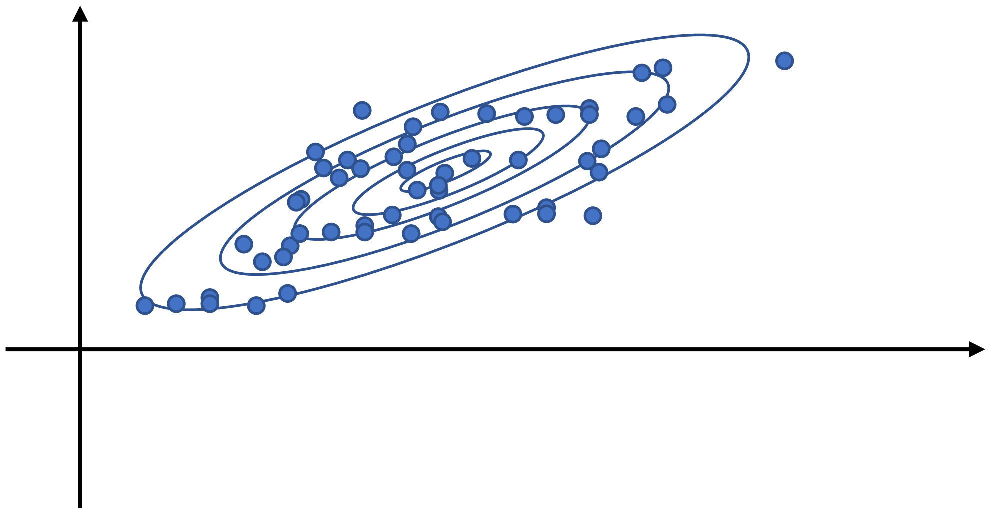
$$p(\boldsymbol{x}) = \mathcal{N}(\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^k|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\overline{\boldsymbol{x}})^T\Sigma^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})}$$



Gaussian Distribution

Dataset $\{\boldsymbol{x}_i\}$

$$p(\boldsymbol{x}) = \mathcal{N}(\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^k|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\bar{\boldsymbol{x}})^T\Sigma^{-1}(\boldsymbol{x}-\bar{\boldsymbol{x}})}$$



Maximum Likelihood Estimators (MLE):

$$\bar{\boldsymbol{x}} = \frac{1}{N}\sum_i \boldsymbol{x}_i$$

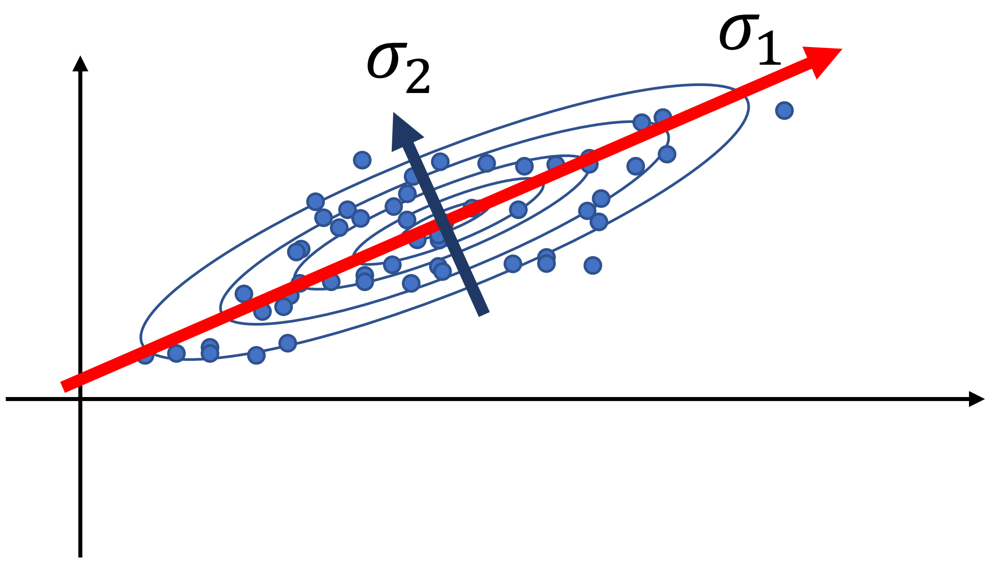
$$\Sigma = \frac{1}{N} X^T X$$

X

PCA and Gaussian Distribution

$$p(\boldsymbol{x}) = \mathcal{N}(\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^k|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\overline{\boldsymbol{x}})^T\Sigma^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})}$$

Dataset $\{\boldsymbol{x}_i\}$



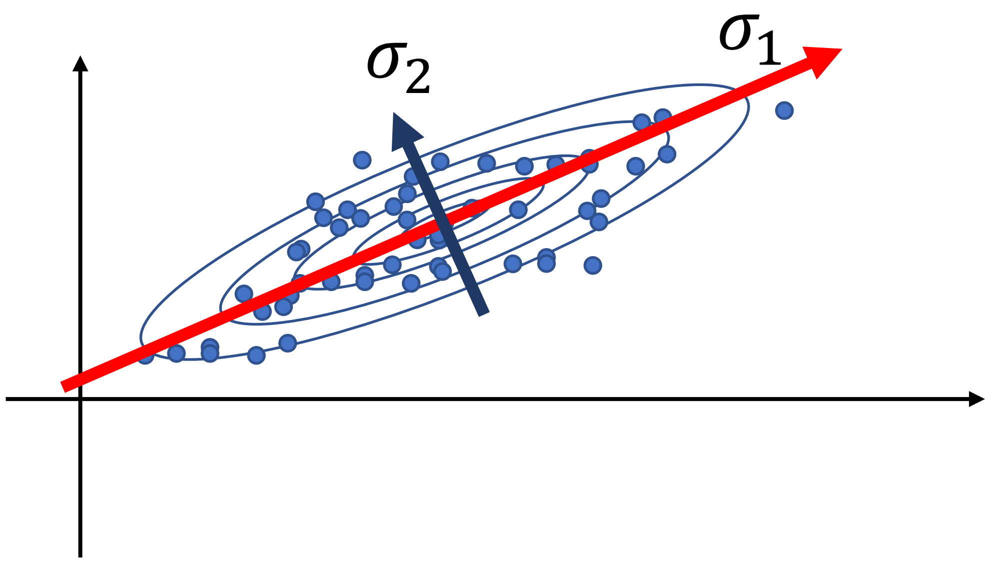
$$\Sigma = X^T X = U \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_N^2 \end{bmatrix} U^T$$

$$\boldsymbol{x} - \overline{\boldsymbol{x}} = \sum_{k=1}^{n} w_k \boldsymbol{u_k}$$

PCA and Gaussian Distribution

Dataset $\{\boldsymbol{x}_i\}$

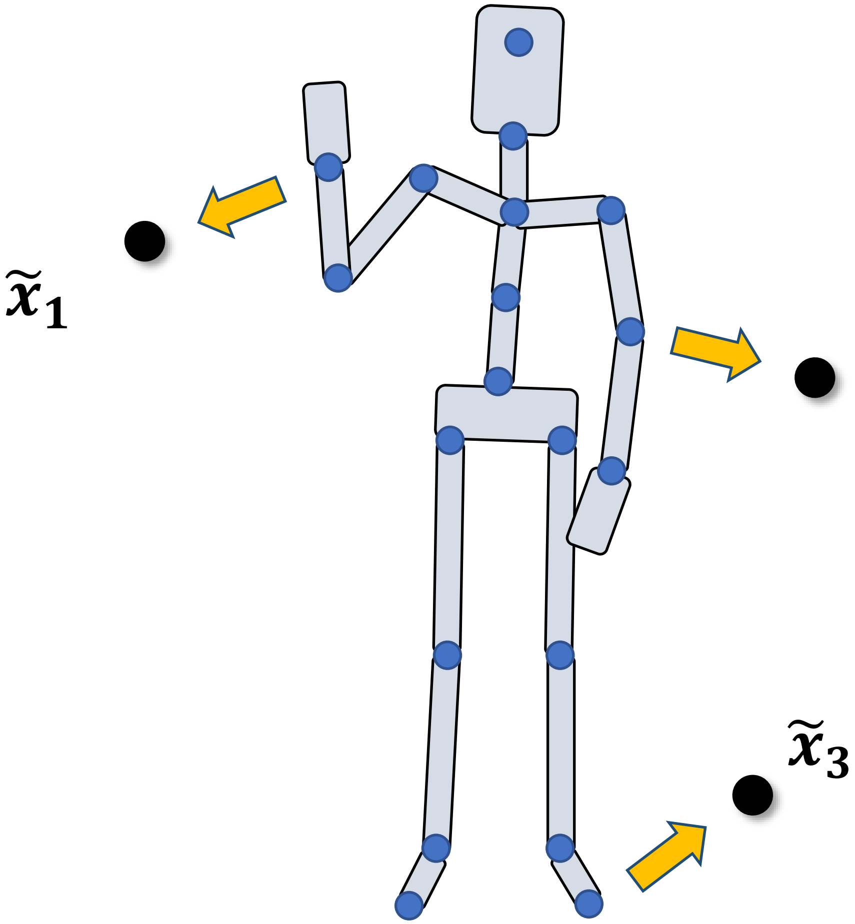
$$p(\boldsymbol{x}) = \mathcal{N}(\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\overline{\boldsymbol{x}})^T \Sigma^{-1} (\boldsymbol{x}-\overline{\boldsymbol{x}})}$$



$$p(\boldsymbol{x}) = \prod_k \frac{1}{\sigma_k \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{w_k}{\sigma_k}\right)^2}$$

$$w_k = (\boldsymbol{x} - \overline{\boldsymbol{x}}) \cdot \boldsymbol{u}_k$$

Character IK with a Motion Prior

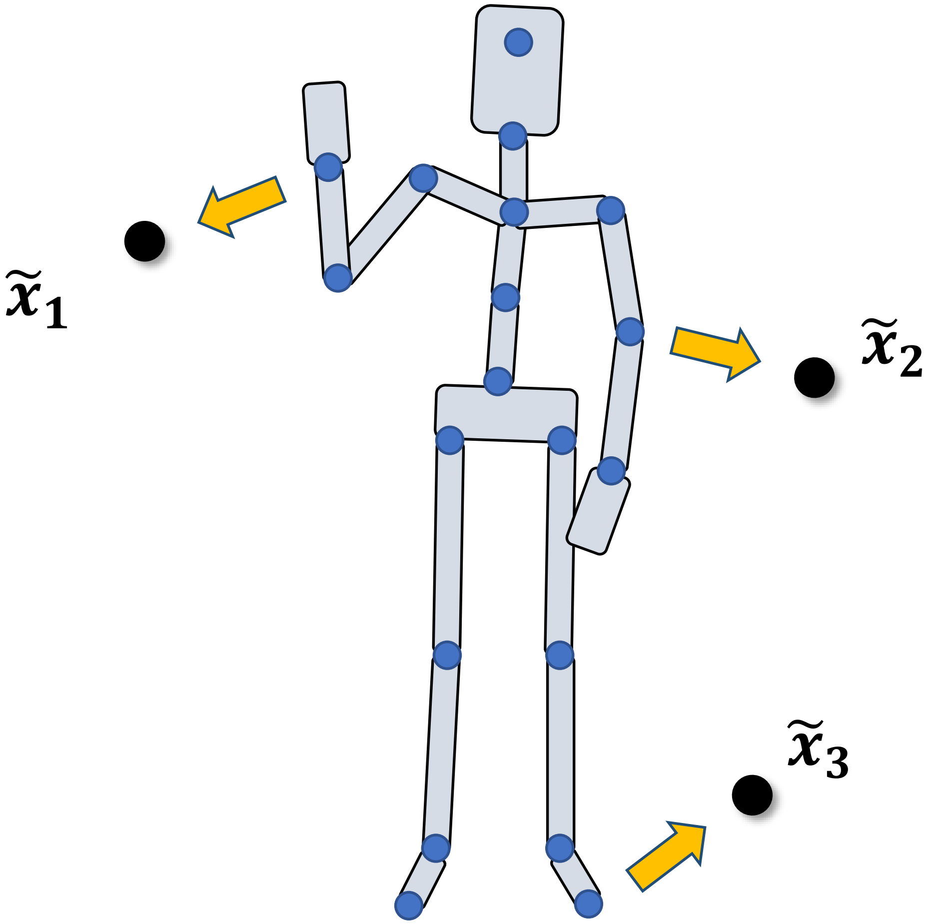


$$F(\theta) = \frac{1}{2}\sum_i \|f_i(\boldsymbol{\theta}) - \tilde{\boldsymbol{x}}_i\|_2^2$$

$$+\frac{w}{2}\sum_k \left(\frac{(\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}) \cdot \boldsymbol{u}_k}{\sigma_k}\right)^2$$

$$\boldsymbol{\theta} = (\boldsymbol{t}_0, R_0, R_1, R_2, \dots \dots)$$

Character IK with a Motion Prior

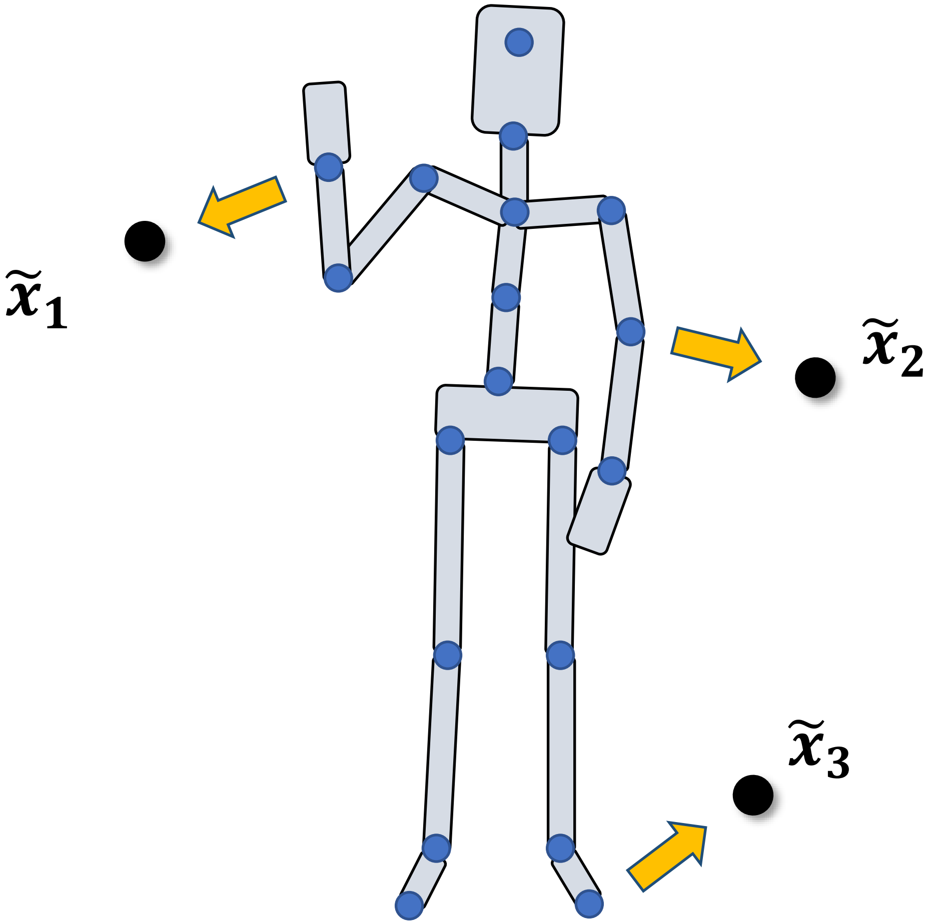


$$F(\theta) = \frac{1}{2}\sum_i \|f_i(\boldsymbol{\theta}) - \tilde{\boldsymbol{x}}_i\|_2^2$$

$$-w \log \prod_k e^{-\frac{1}{2}\left(\frac{(\boldsymbol{\theta}-\bar{\boldsymbol{\theta}})\cdot \boldsymbol{u}_k}{\sigma_k}\right)^2}$$

$$\boldsymbol{\theta} = (\boldsymbol{t}_0, R_0, R_1, R_2, \dots\dots)$$

Character IK with a Motion Prior



$$F(\theta) = \frac{1}{2}\sum_i \|f_i(\boldsymbol{\theta}) - \tilde{\boldsymbol{x}}_i\|_2^2$$

$$-w \log p(\theta)$$

$$\prod_k e^{-\frac{1}{2}\left(\frac{(\boldsymbol{\theta}-\bar{\boldsymbol{\theta}})\cdot \boldsymbol{u}_k}{\sigma_k}\right)^2}$$

$$\boldsymbol{\theta} = (\boldsymbol{t}_0, R_0, R_1, R_2, \dots\dots)$$

Motion Synthesis with a Motion Prior

Given a motion prior $p(\boldsymbol{x})$ learned from a set of data points $\mathrm{D} = \{\boldsymbol{x}_i\}$,

Synthesize a motion $\boldsymbol{x}$ that minimize the objective

$$F(x) = f(\boldsymbol{x}) - w \log p(\boldsymbol{x})$$

Note: x can represent a pose $\boldsymbol{\theta}$

or a motion clip → a sequence of poses $\{\boldsymbol{\theta_t}\}$

or any features of a motion → e.g. w_k in PCA

Motion Synthesis with a Motion Prior

Given a motion prior $p(\boldsymbol{x})$ learned from a set of data points $\mathrm{D} = \{\boldsymbol{x}_i\}$,

Synthesize a motion $\boldsymbol{x}$ that minimize the objective

$$F(x) = f(\boldsymbol{x}) - w \log p(\boldsymbol{x})$$

$f(\boldsymbol{x})$ ⇨

IK

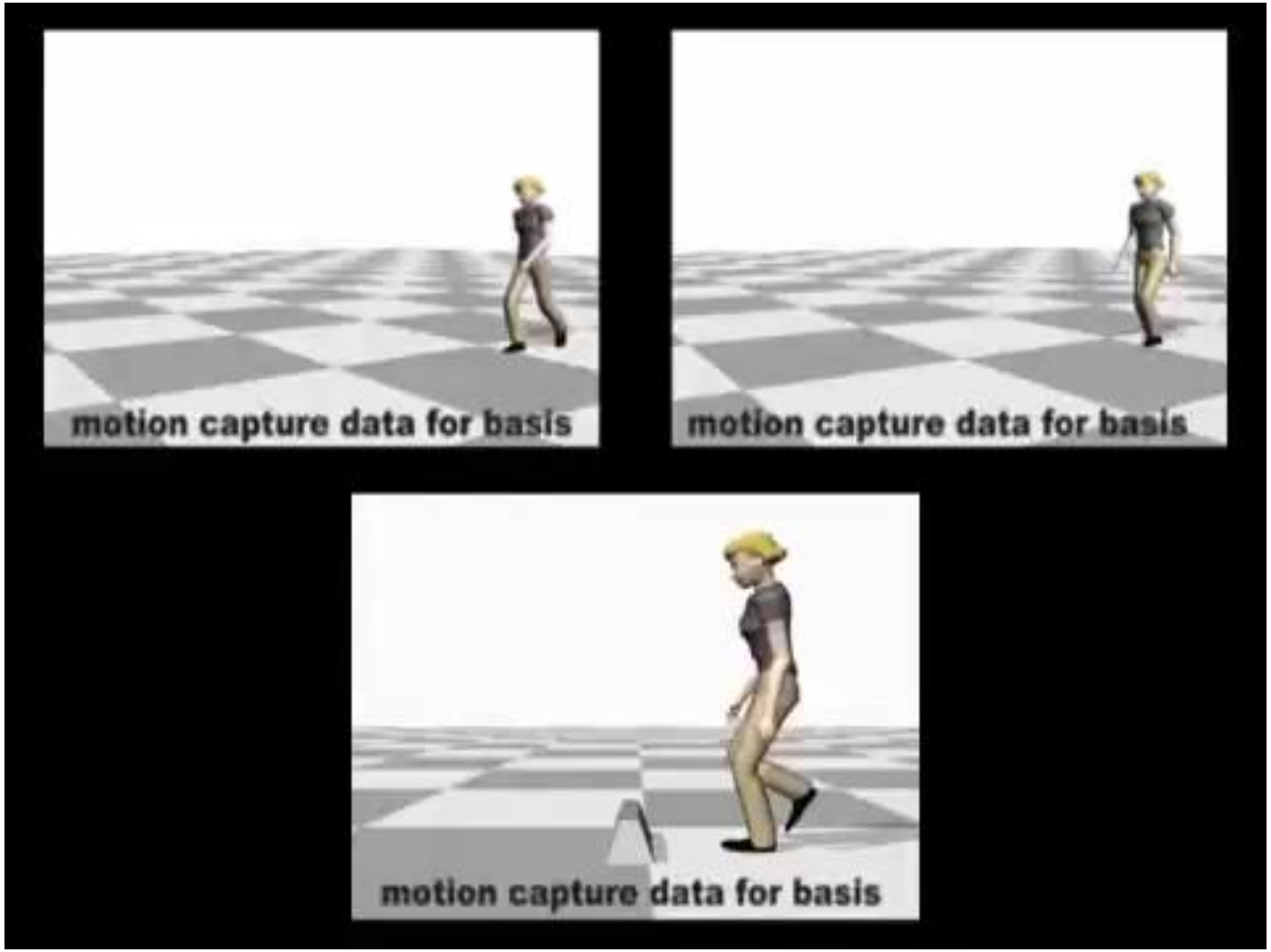
Keyframes

User control

Environment constraints

……

Motion Synthesis with a Motion Prior



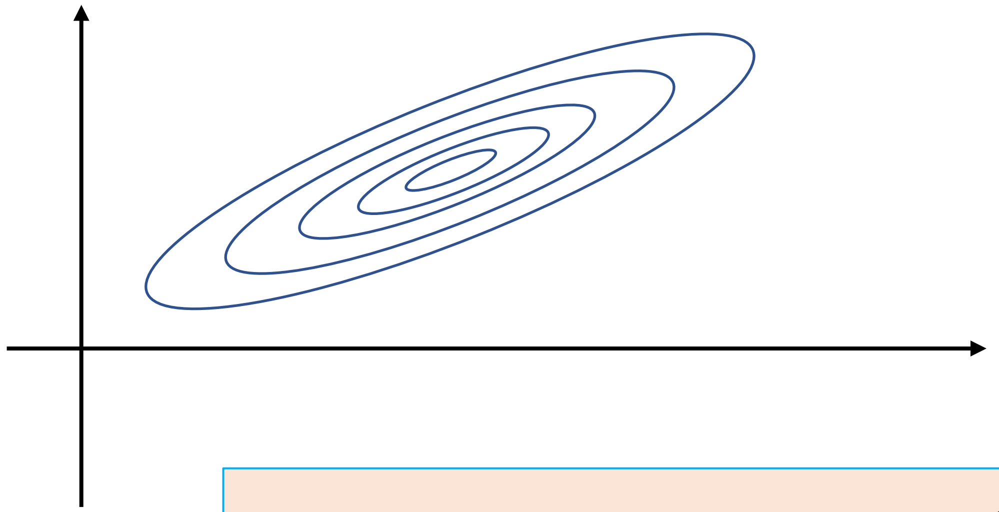
Synthesizing Physically Realistic Human Motion in Low-Dimensional, Behavior-Specific Spaces

Alla Safonova Jessica K. Hodgins Nancy S. Pollard
School of Computer Science
Carnegie Mellon University *

*SIGGRAPH 2004

Gaussian Distribution is not Enough!

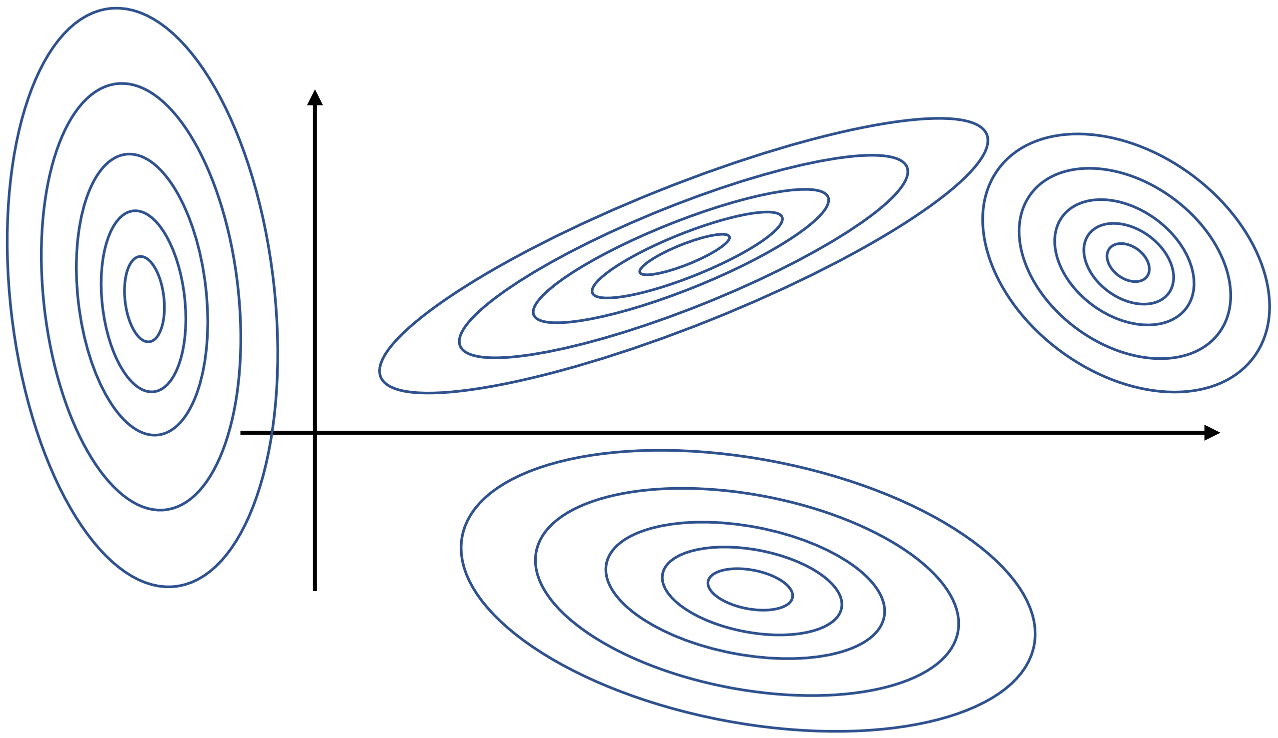
$p(\boldsymbol{x})$: motion prior



$$p(\boldsymbol{x}) = \mathcal{N}(\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^k|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\overline{\boldsymbol{x}})^T\Sigma^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})}$$

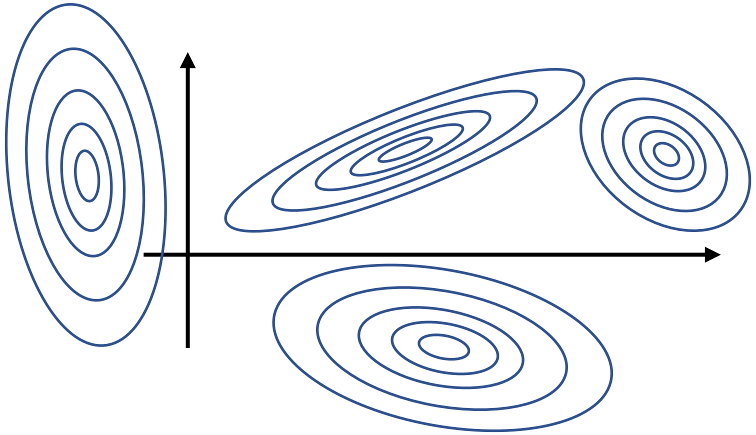
Gaussian Distribution is not Enough!

$p(\boldsymbol{x})$: motion prior



Gaussian Distribution is not Enough!

$p(\boldsymbol{x})$: motion prior



Interactive Generation of Human Animation with Deformable Motion Models

Jianyuan Min
Texas A&M University

Yen-Lin Chen
Texas A&M University

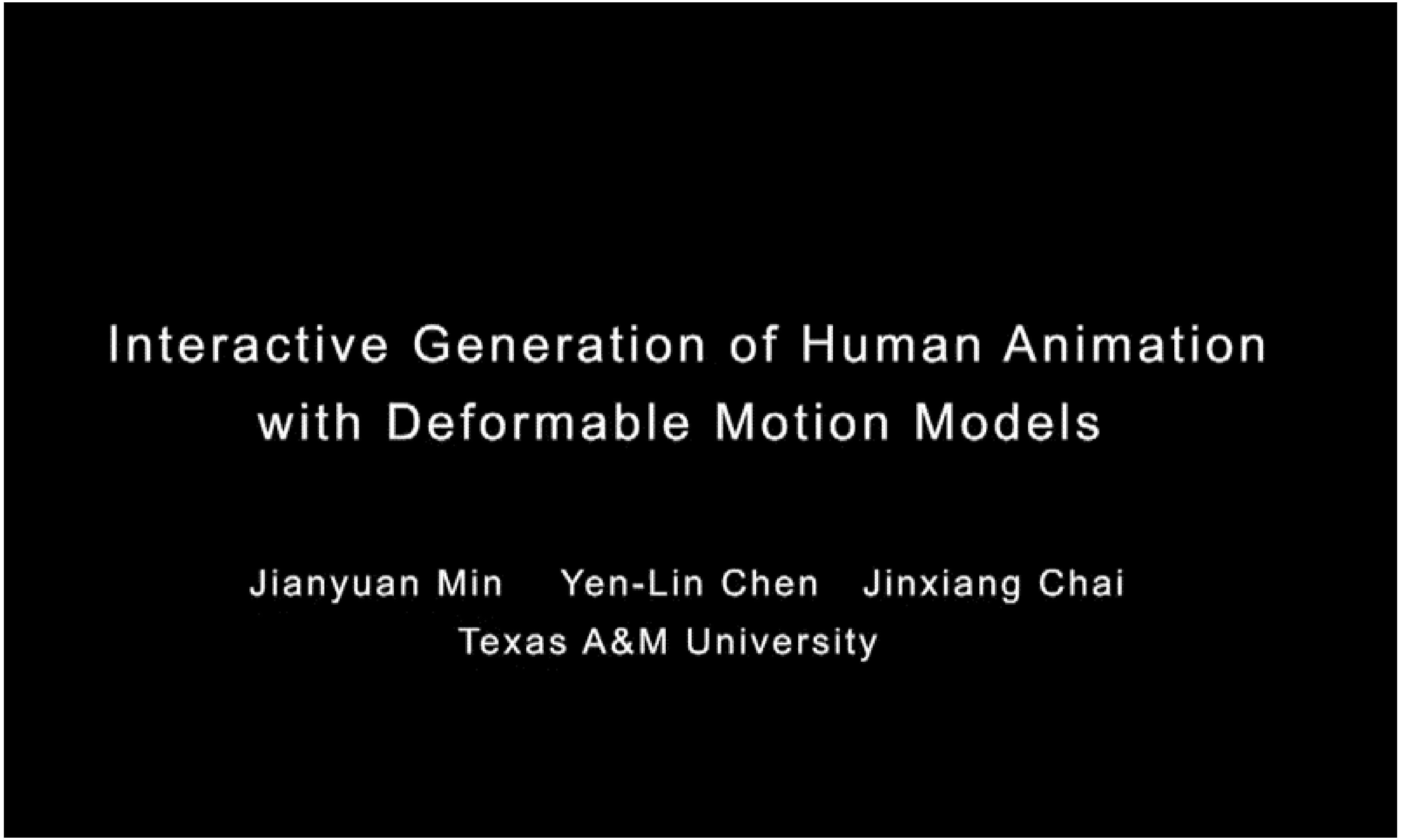
Jinxiang Chai
Texas A&M University

* SIGGRAPH 2009

Gaussian Mixture Models (GMM)

$$p(\boldsymbol{x}) = \sum_i \phi_i \mathcal{N}(\mu_i, \Sigma_i)$$

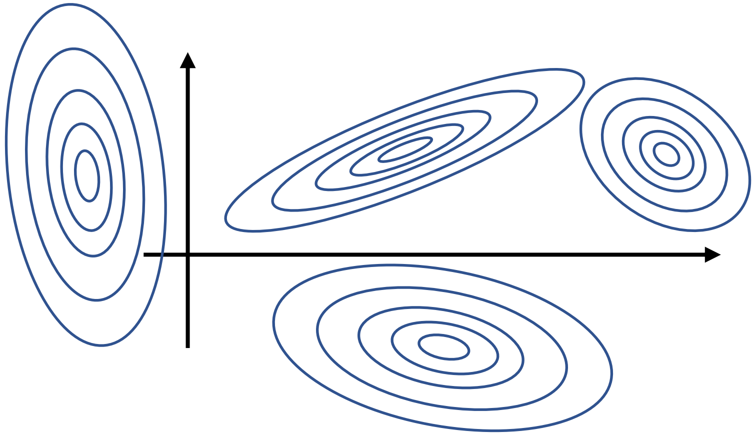
Motion Synthesis with a Motion Prior

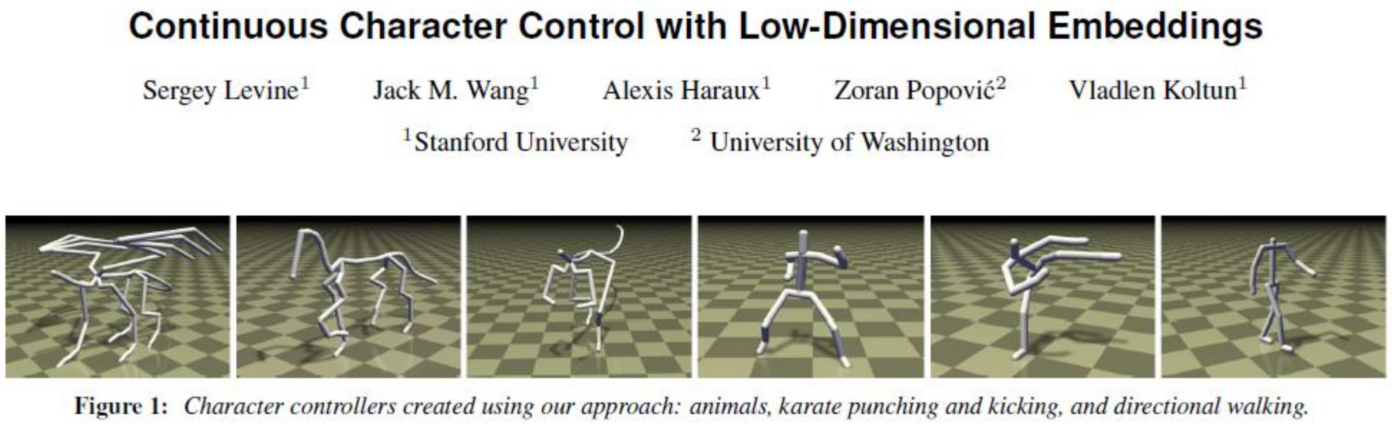


Min et al. 2009

Gaussian Distribution is not Enough!

$p(\boldsymbol{x})$: motion prior




Continuous Character Control with Low-Dimensional Embeddings

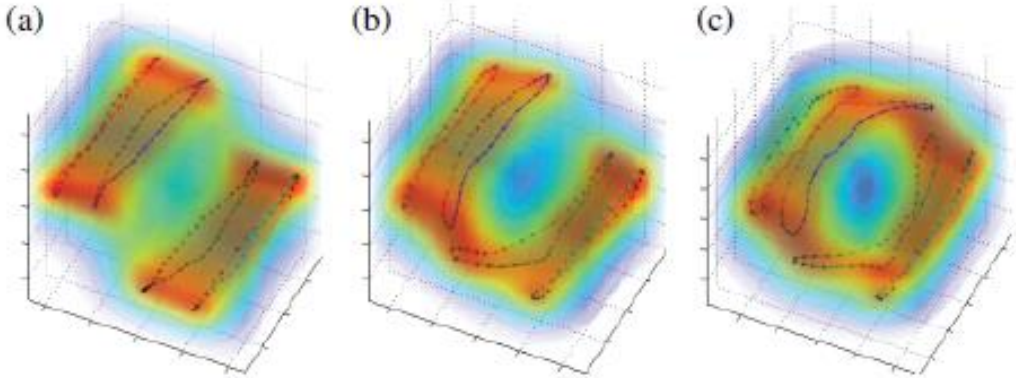
Sergey Levine[1] Jack M. Wang[1] Alexis Haraux[1] Zoran Popović[2] Vladlen Koltun[1]

[1]Stanford University [2] University of Washington

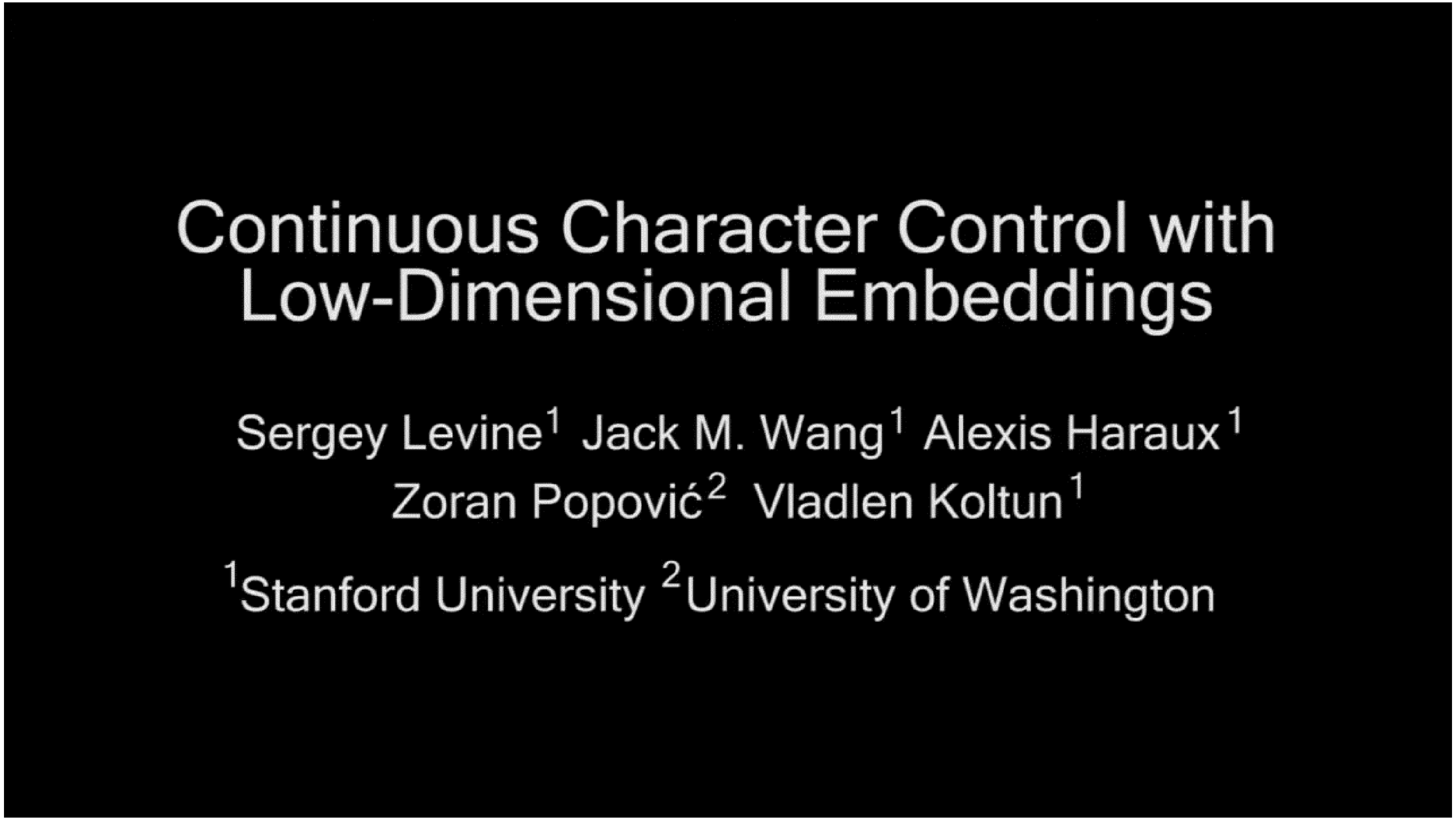
Figure 1: *Character controllers created using our approach: animals, karate punching and kicking, and directional walking.*

* SIGGRAPH 2012

Gaussian Process Latent Variable Model (GPLVM)



Motion Synthesis with a Motion Prior

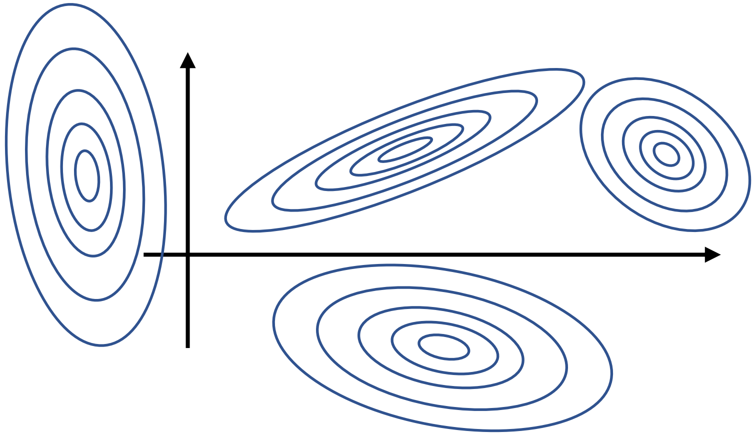


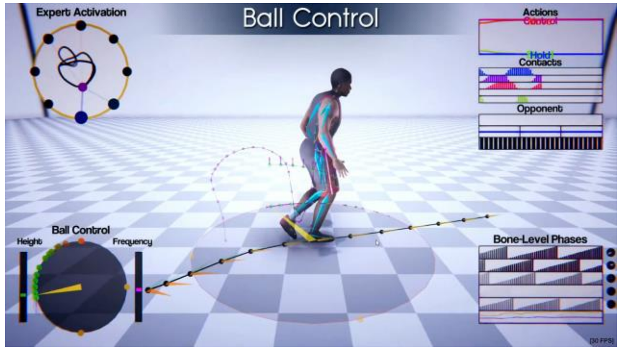
Levine et al. 2012

Gaussian Distribution is not Enough!

Neural networks…

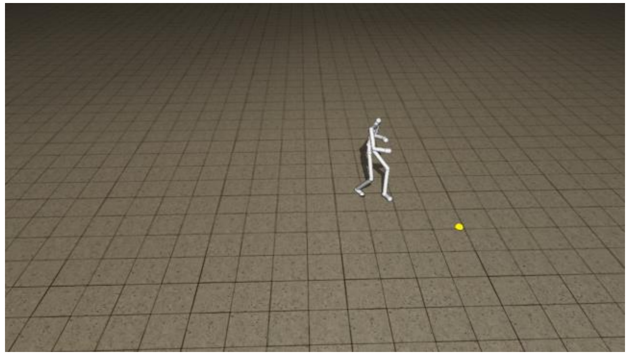
$p(\boldsymbol{x})$: motion prior





[Starke et al 2020, Local Motion Phases for Learning Multi-Contact Character Movements]

[Henter et al. 2020, MoGlow: Probabilistic and Controllable Motion Synthesis Using Normalising Flows]



[Lee et al 2019, Interactive Character Animation by Learning Multi-Objective Control]

[Holden et al 2020, Learned Motion Matching]

Questions?