

GAMES 105

Fundamentals of Character Animation

Lecture 06+

Learning-based Character Animation (cont.)

Libin Liu

School of Intelligence Science and Technology
Peking University



GAMES105 课程交流



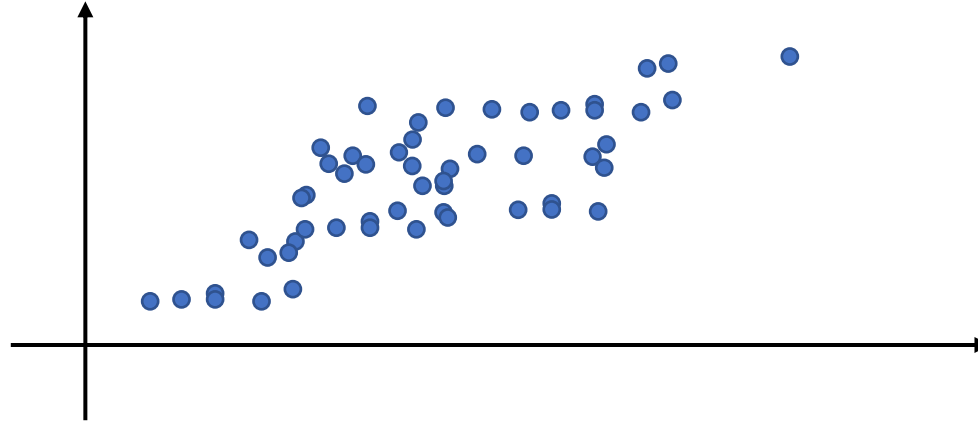
VCL @ PKU

Outline

- Learning-based Character Animation (cont.)
 - Motion Models
 - Autoregressive models: PFNN
 - Generative models

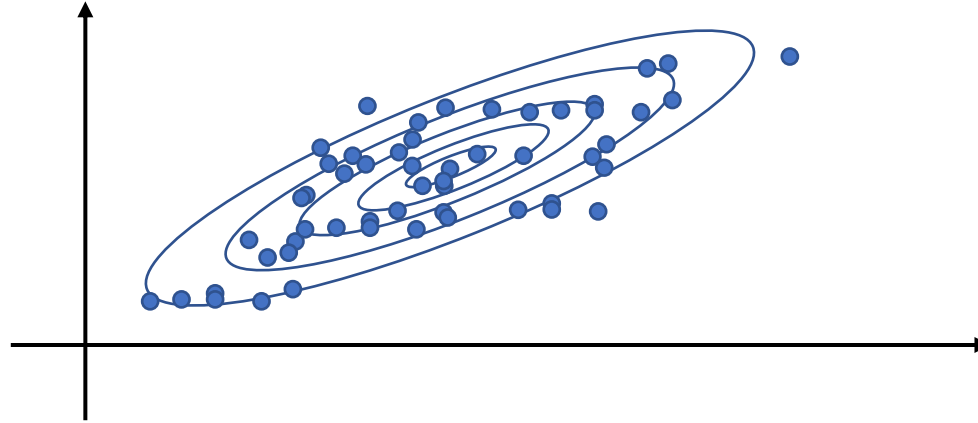
Learning Motion Models

Given a set of example motions $\{x_i\} \sim p(x)$



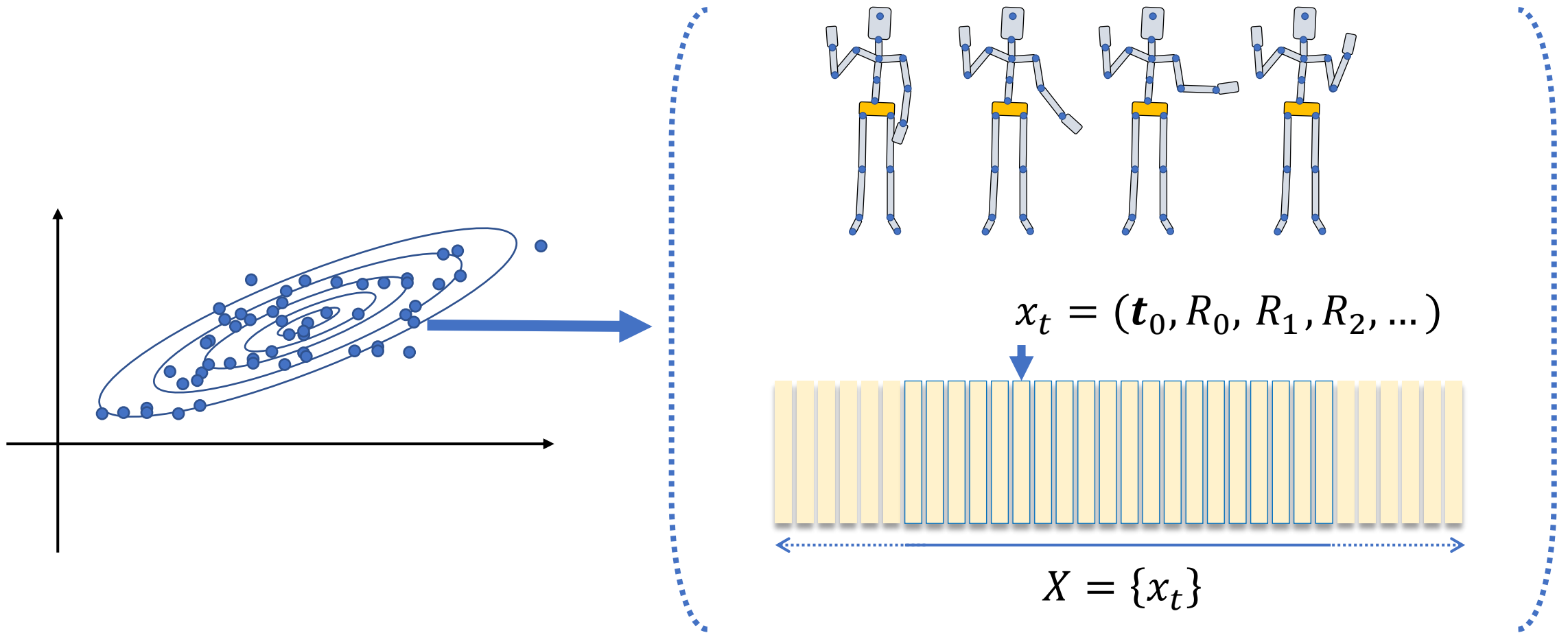
Learning Motion Models

Given a set of example motions $\{\mathbf{x}_i\} \sim p(\mathbf{x})$

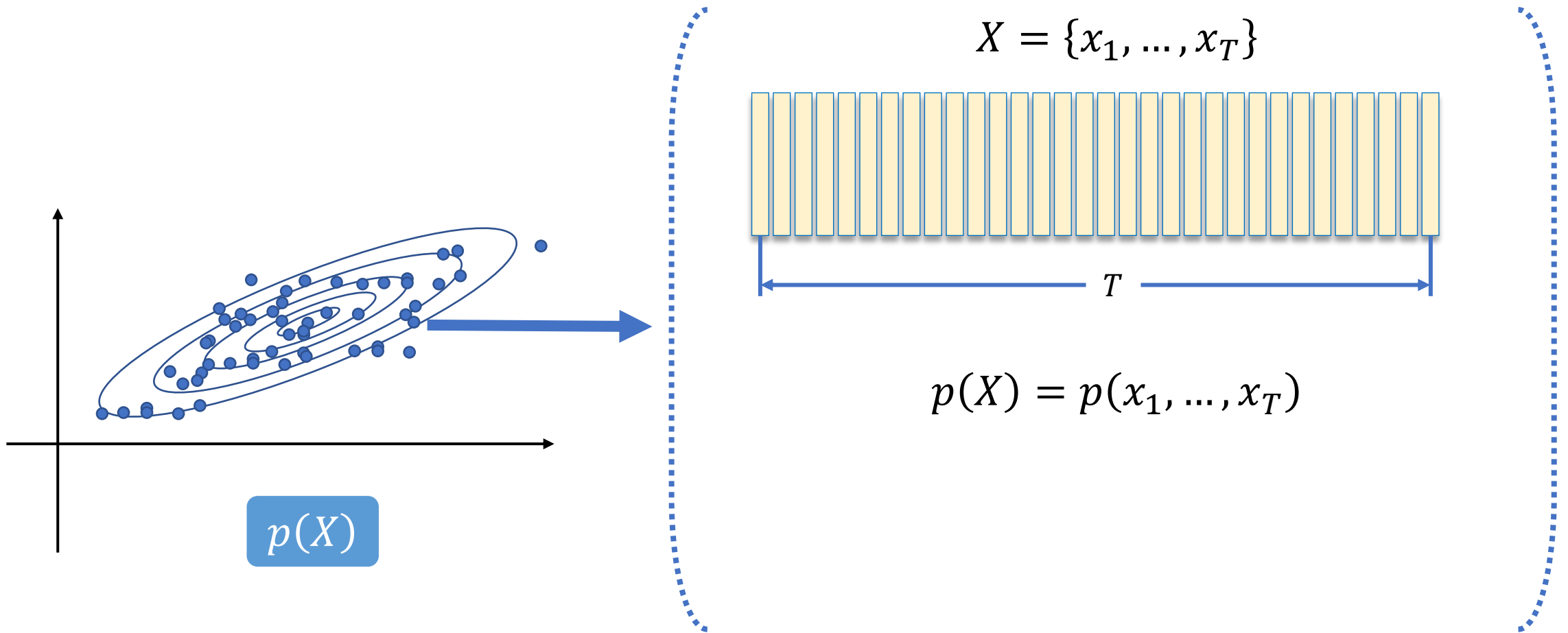


$p(\mathbf{x})$: probability that \mathbf{x} is a natural motion

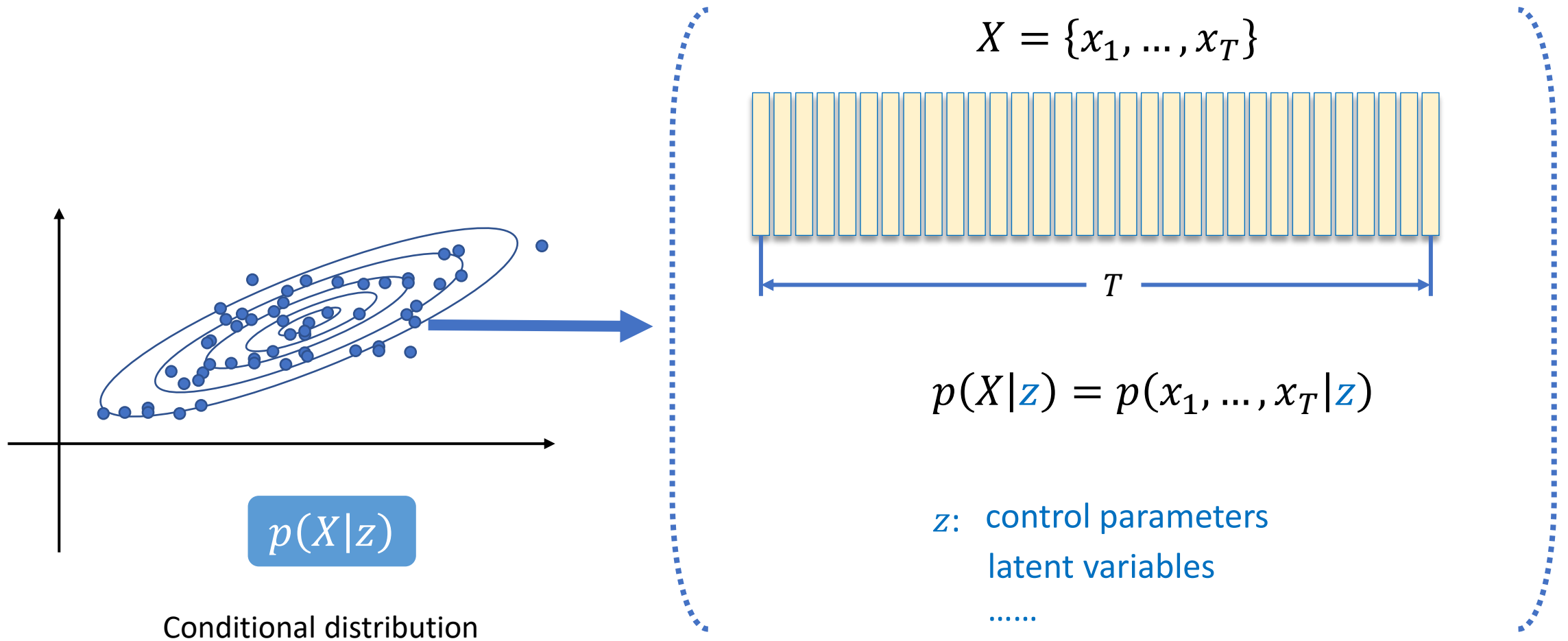
Learning Motion Models



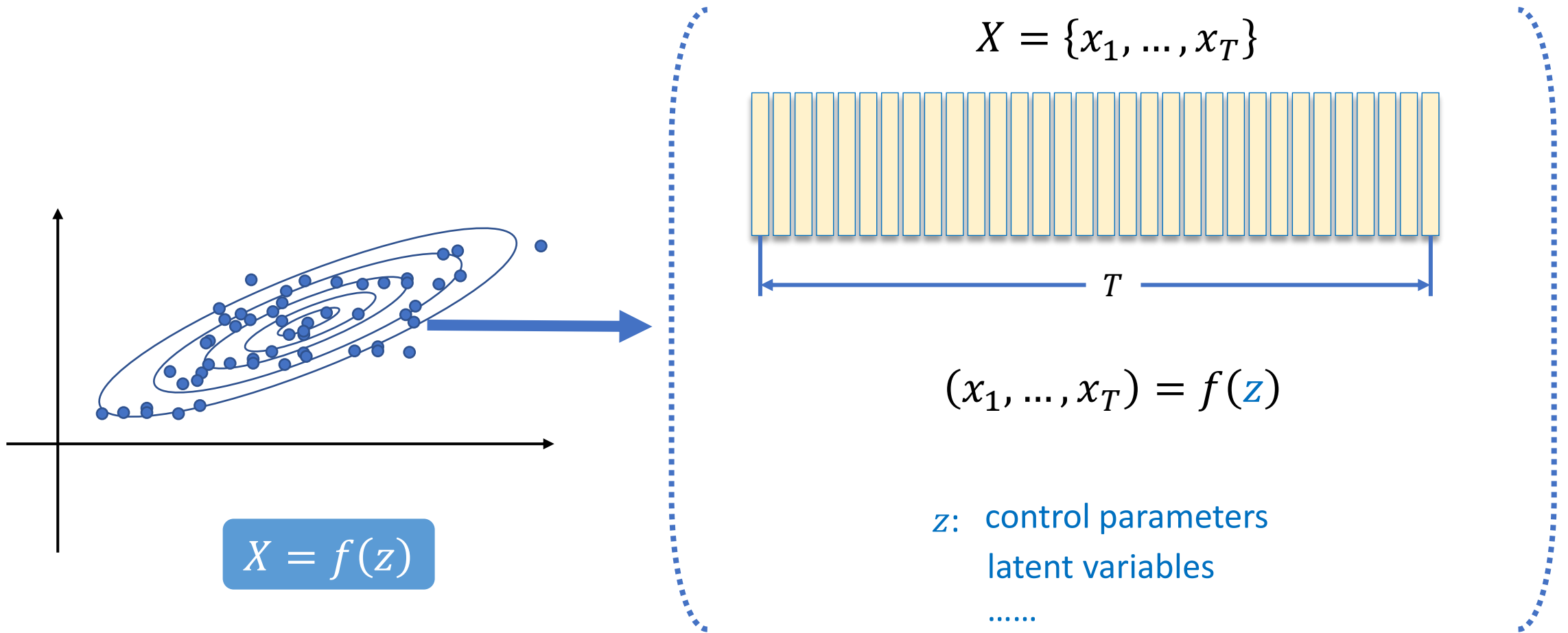
Learning Motion Models



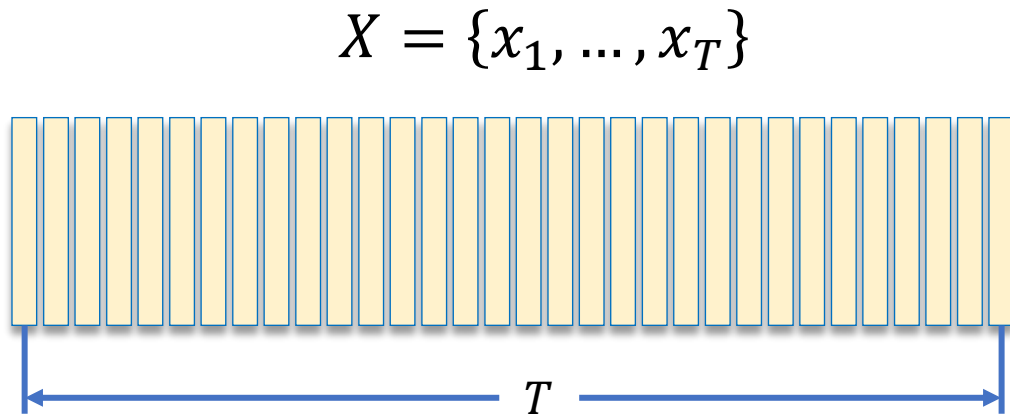
Learning Motion Models



Learning Motion Models



Two Perspectives on a Motion Sequence

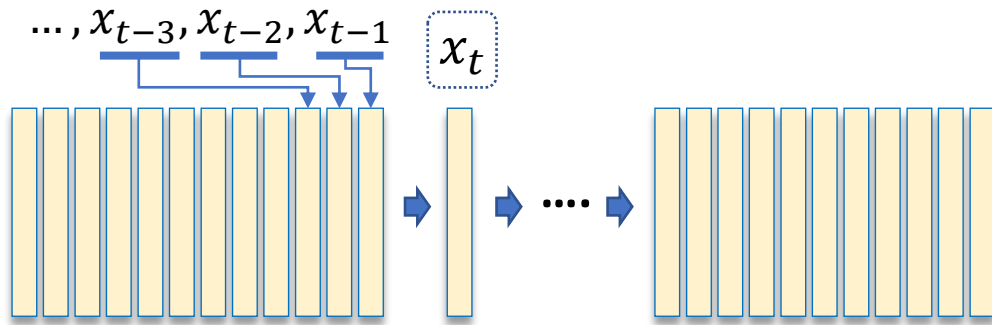


$$p(X|z)$$

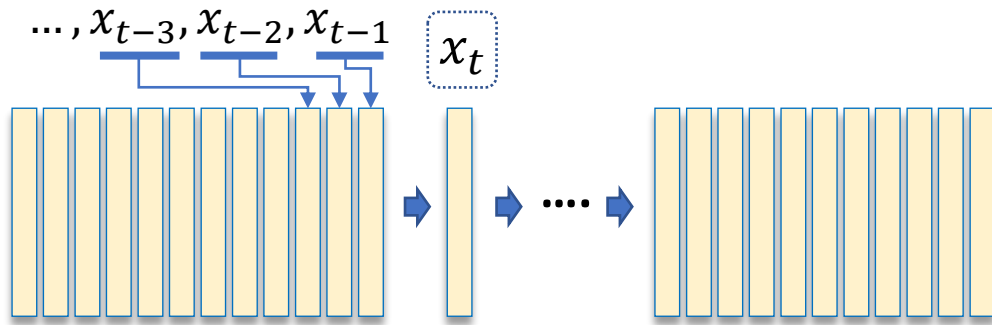


$$X = f(\textcolor{red}{z})$$

Two Perspectives on a Motion Sequence

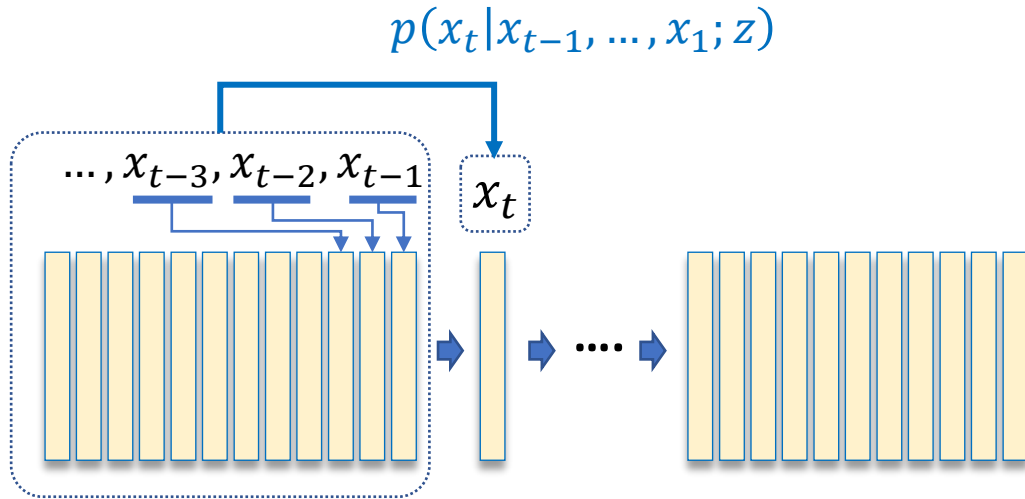


Two Perspectives on a Motion Sequence



$$p(X|z) = p(x_1, \dots, x_T|z)$$

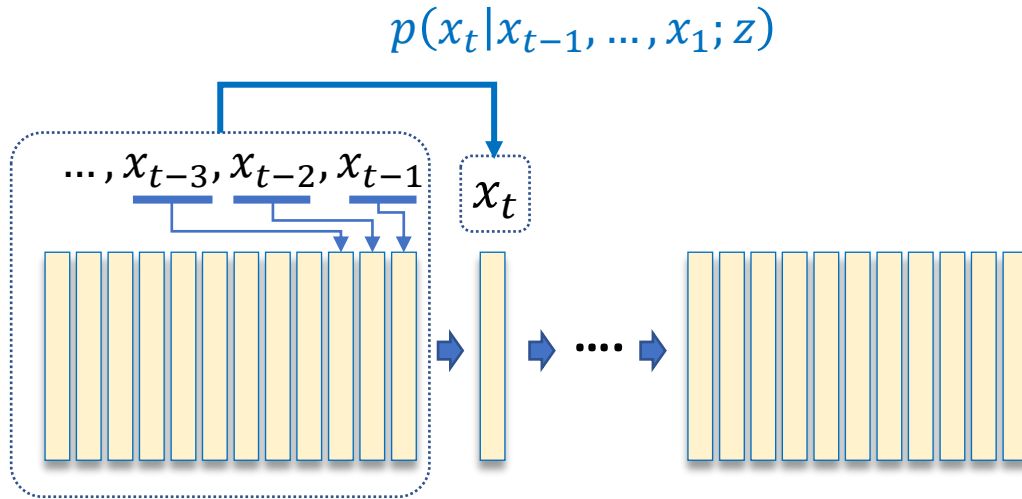
Two Perspectives on a Motion Sequence



$$p(X|z) = p(x_1, \dots, x_T|z)$$

$$= p(x_1) \prod_t p(x_t | x_{t-1}, \dots, x_1; z)$$

Two Perspectives on a Motion Sequence



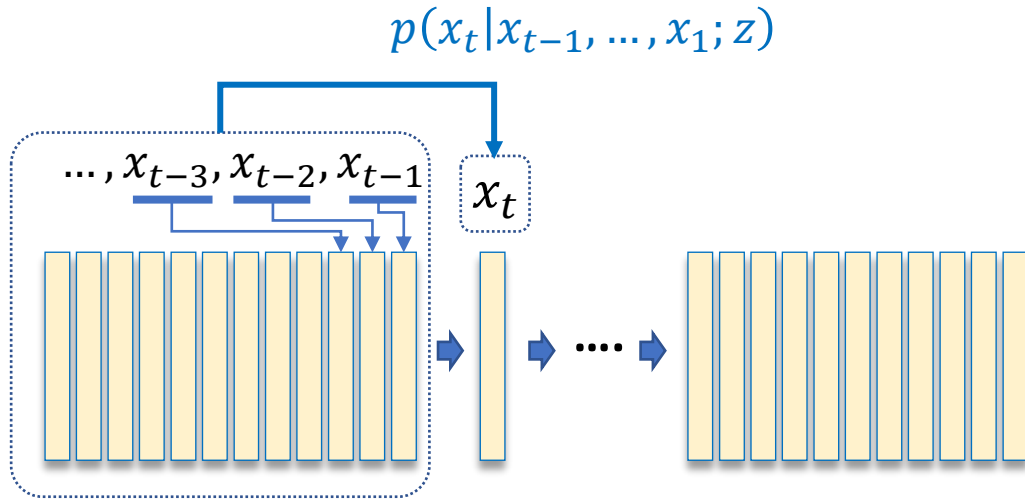
$$p(X|z) = p(x_1, \dots, x_T|z)$$

$$= p(x_1) \prod_t p(x_t | x_{t-1}, \dots, x_1; z)$$

*The chain rule of conditional probabilities:

$$p(x_1, x_2) = p(x_2 | x_1) p(x_1)$$

Two Perspectives on a Motion Sequence



$$p(X|z) = p(x_1, \dots, x_T|z)$$

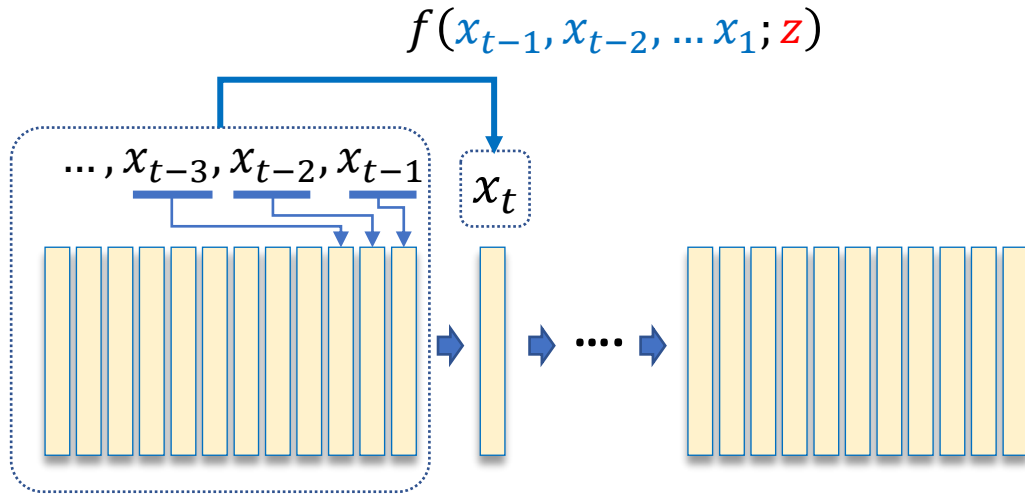
$$= p(x_1) \prod_t p(x_t | x_{t-1}, \dots, x_1; z)$$

*The chain rule of conditional probabilities:

$$p(x_1, x_2) = p(x_2 | x_1) p(x_1)$$

$$\begin{aligned} p(x_1, x_2, x_3) &= p(x_2, x_3 | x_1) p(x_1) \\ &= p(x_3 | x_2, x_1) p(x_2 | x_1) p(x_1) \end{aligned}$$

Two Perspectives on a Motion Sequence



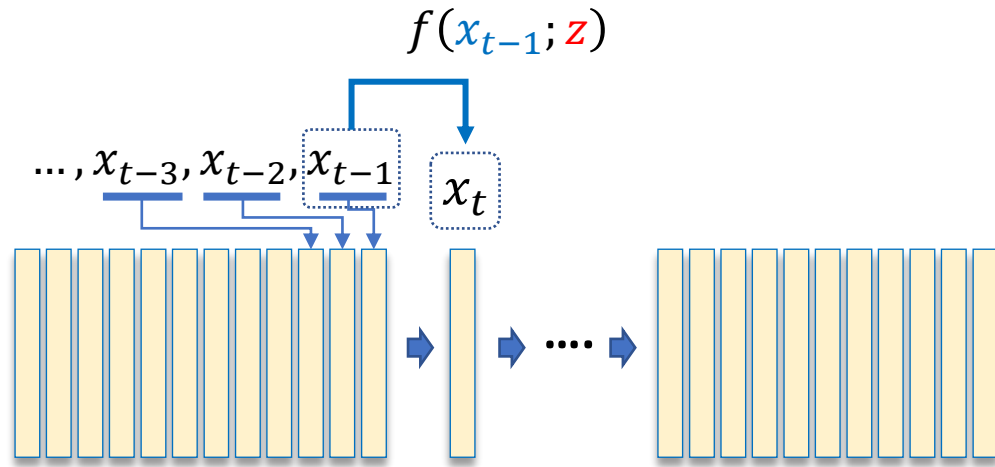
$$p(X|z) = p(x_1, \dots, x_T|z)$$

$$= p(x_1) \prod_t p(x_t | x_{t-1}, \dots, x_1; z)$$



$$x_t = f(x_{t-1}, x_{t-2}, \dots, x_1; z)$$

Two Perspectives on a Motion Sequence



Markov Property

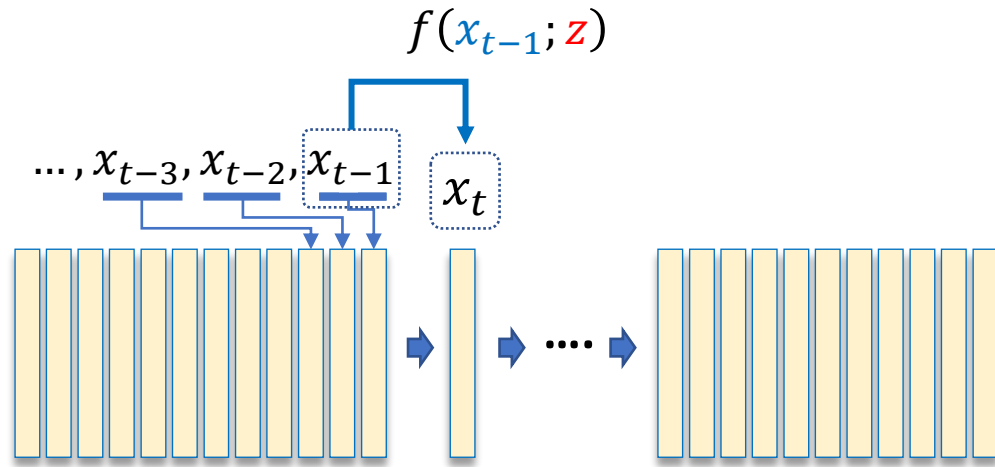
$$p(X|z) = p(x_1, \dots, x_T|z)$$

$$= p(x_1) \prod_t p(x_t | x_{t-1}, \dots, x_1; z)$$



$$x_t = f(x_{t-1}, \dots, x_1; z)$$

Two Perspectives on a Motion Sequence



Markov Property

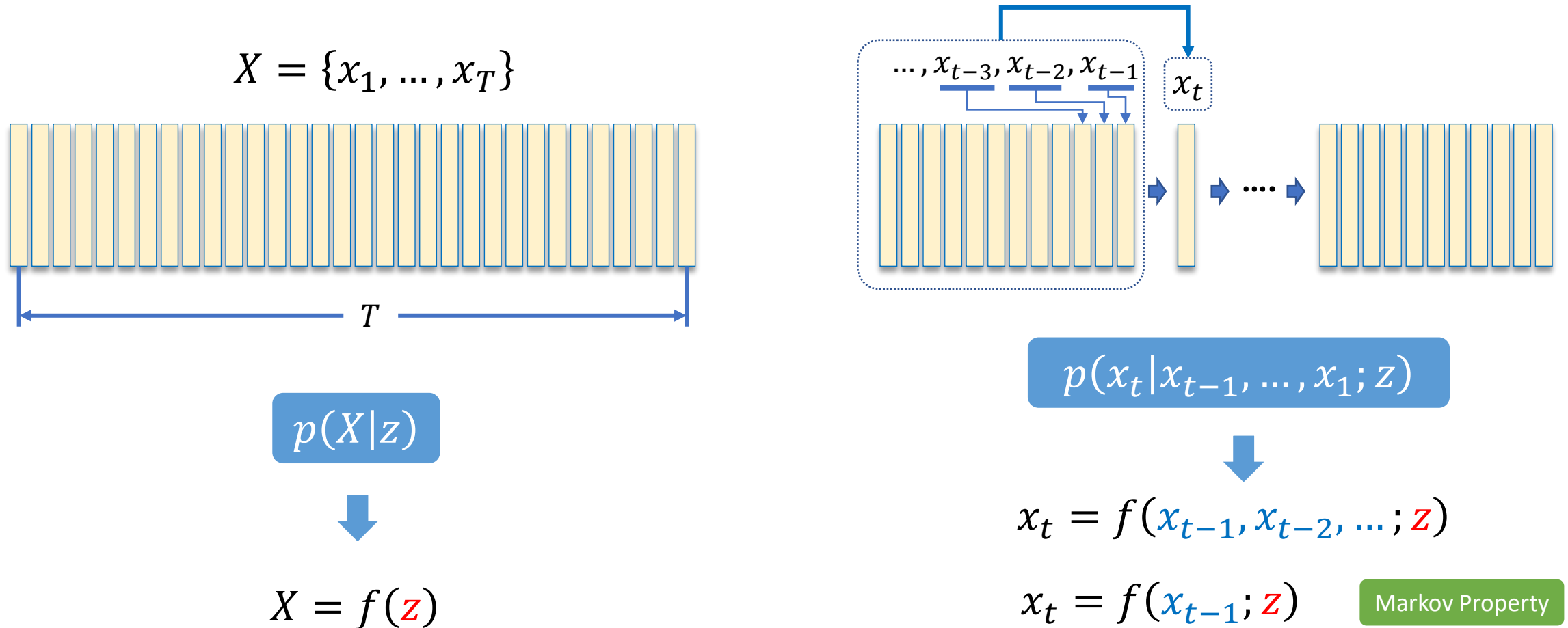
$$p(X|z) = p(x_1, \dots, x_T|z)$$

$$= p(x_1) \prod_t p(x_t|x_{t-1}; z)$$



$$x_t = f(x_{t-1}; z)$$

Two Perspectives on a Motion Sequence



Autoregressive Model

$$x_t = f(x_{t-1}, x_{t-2}, \dots; \mathbf{z})$$


$$x_t = f(x_{t-1}; \mathbf{z})$$

Autoregressive Model

$$x_t = f(x_{t-1}, x_{t-2}, \dots; \mathbf{z})$$

$$x_t = f(x_{t-1}; \mathbf{z})$$

Autoregressive Model

$$x_t = f(x_{t-1}; \mathbf{z})$$


\mathbf{z} : control parameters
latent variables

.....

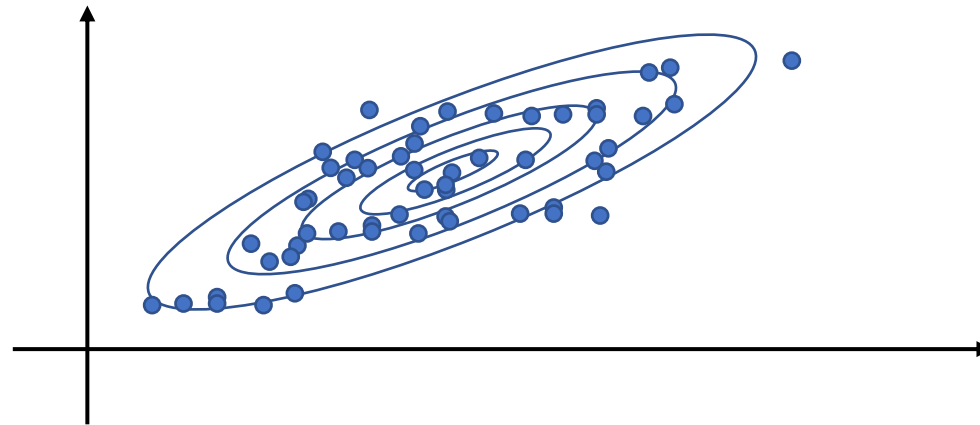
Autoregressive Model

$$x_t = f(x_{t-1})$$

Autoregressive Model

$$x_t = f(x_{t-1})$$

Given a set of example motions $\{X_i\} \sim p(X)$

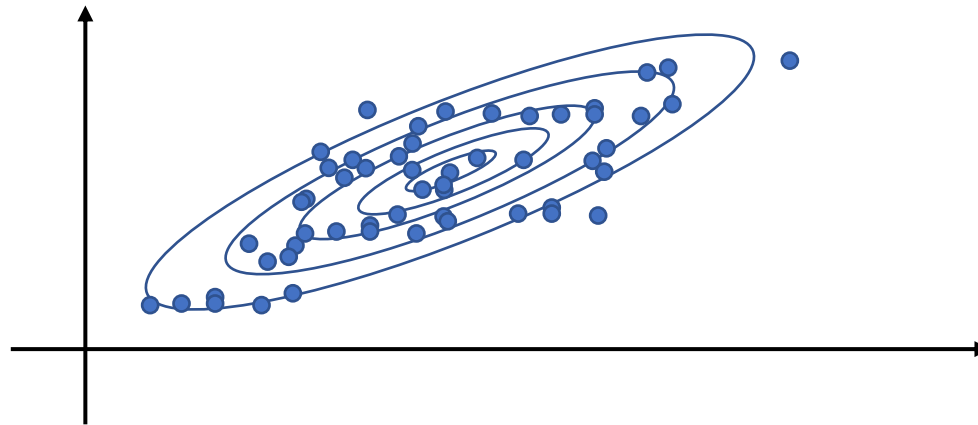


Autoregressive Model

$$x_t = f(x_{t-1})$$

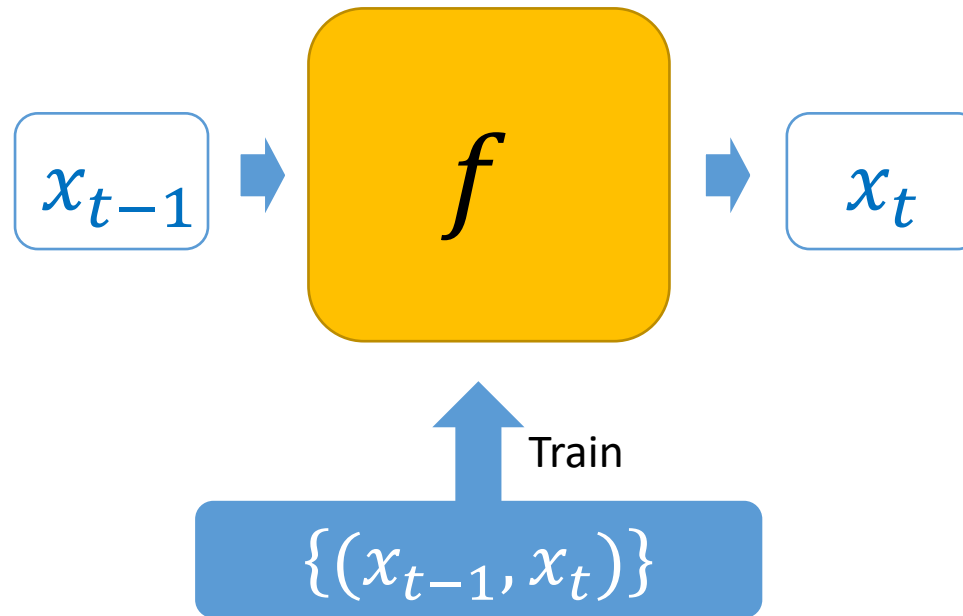
Given a set of example **transitions**

$$\{(x_{t-1}, x_t)\} \sim p(\mathbf{X})$$



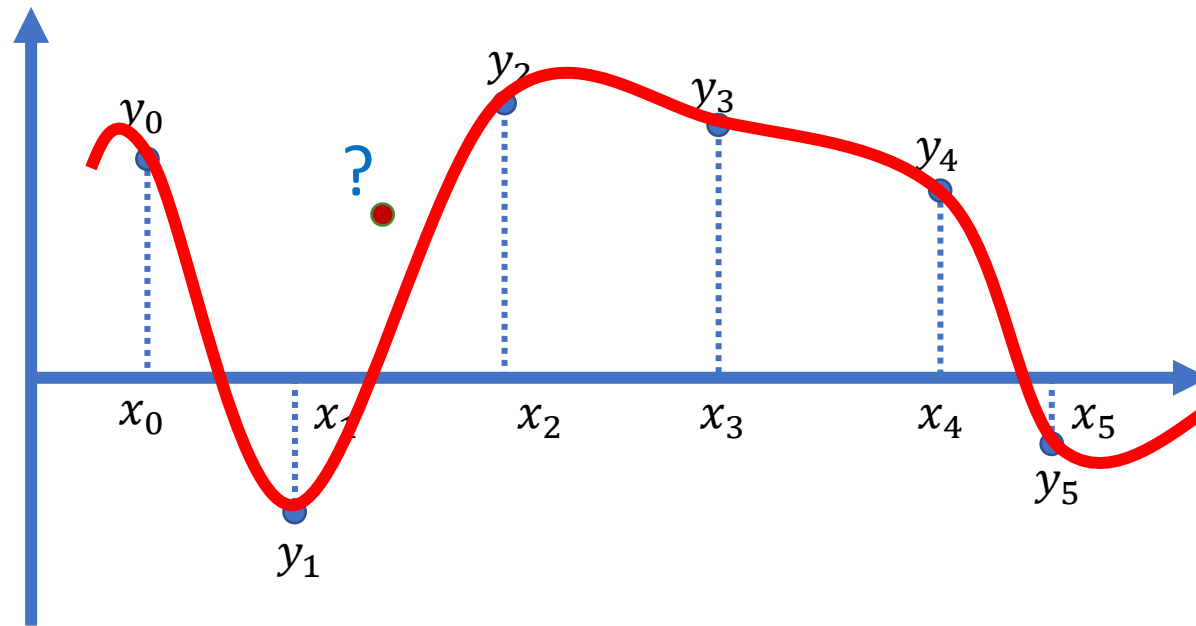
Learning Motion Models

$$x_t = f(x_{t-1})$$

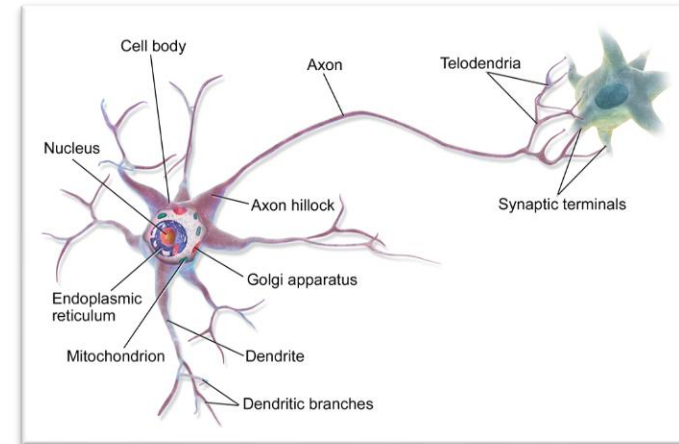


Interpolation

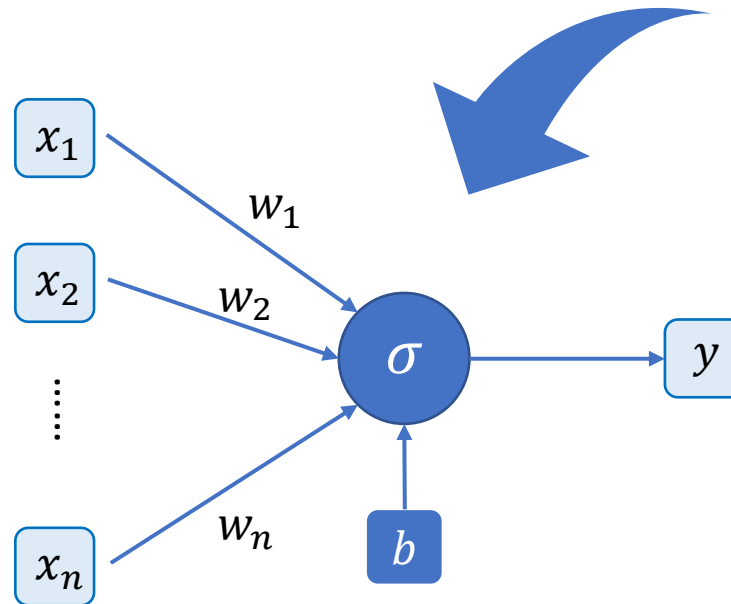
$$f(x_i) = y_i \quad \rightarrow \quad f(x) = ?$$



Neural Networks

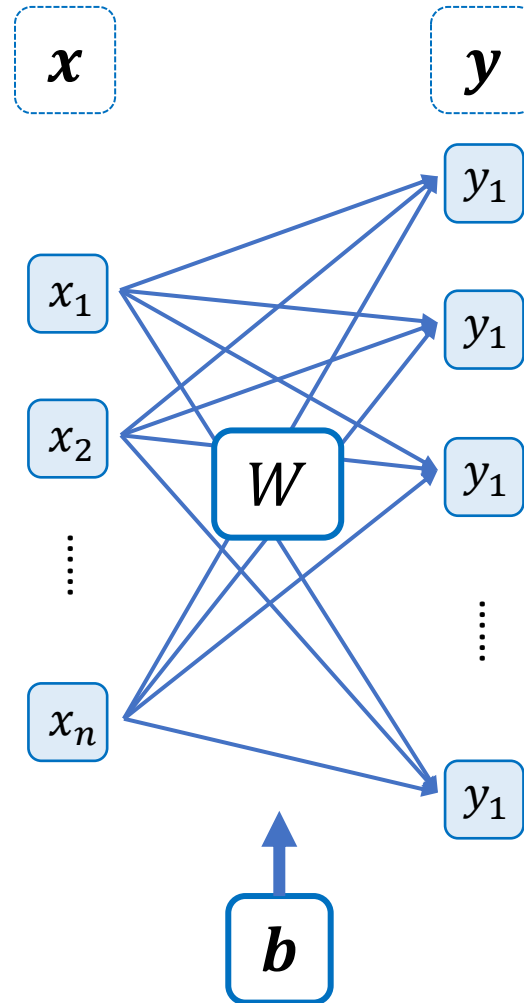


A Multipolar Neuron



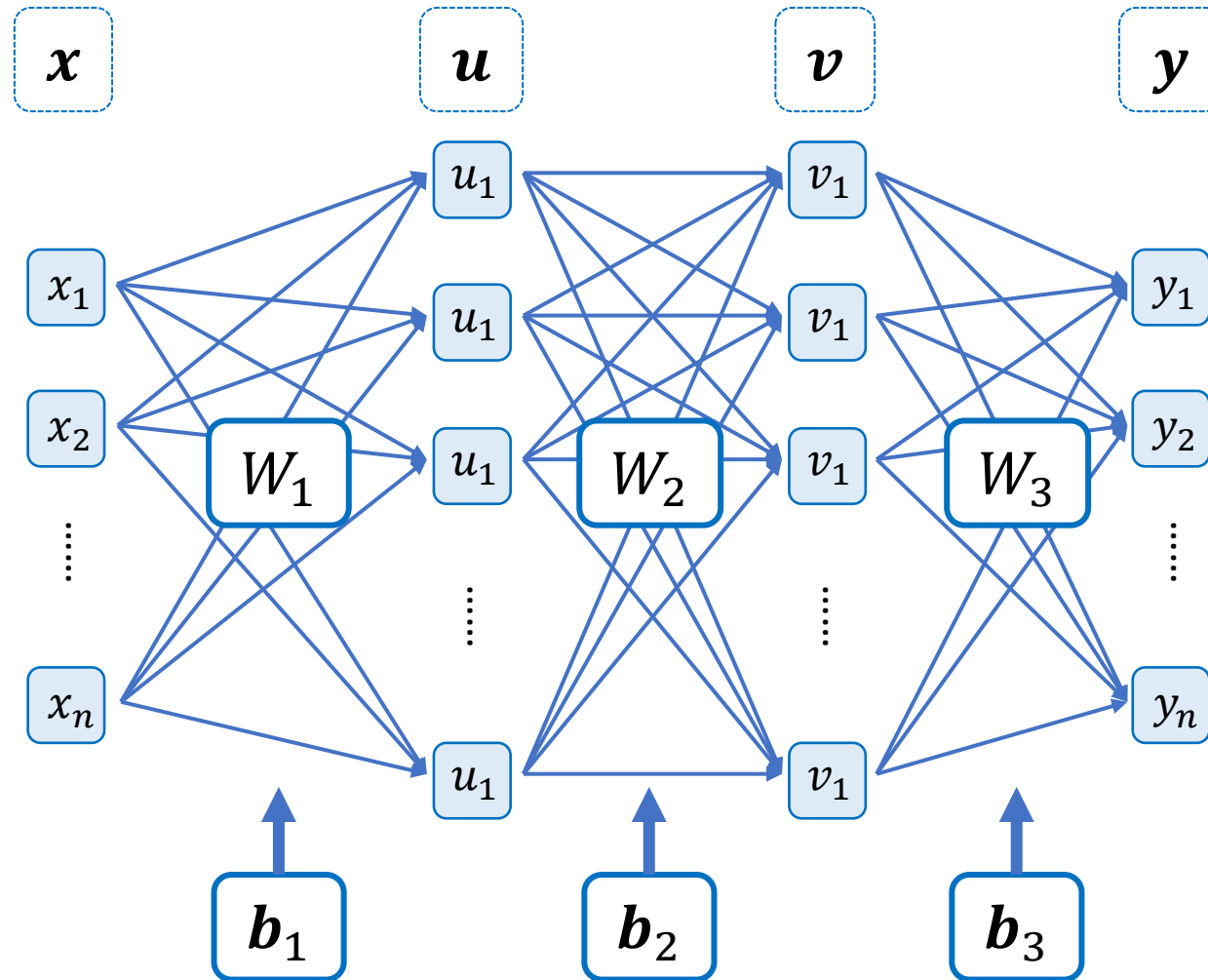
$$y = \sigma \left(\sum_i w_i x_i + b \right)$$

Neural Networks

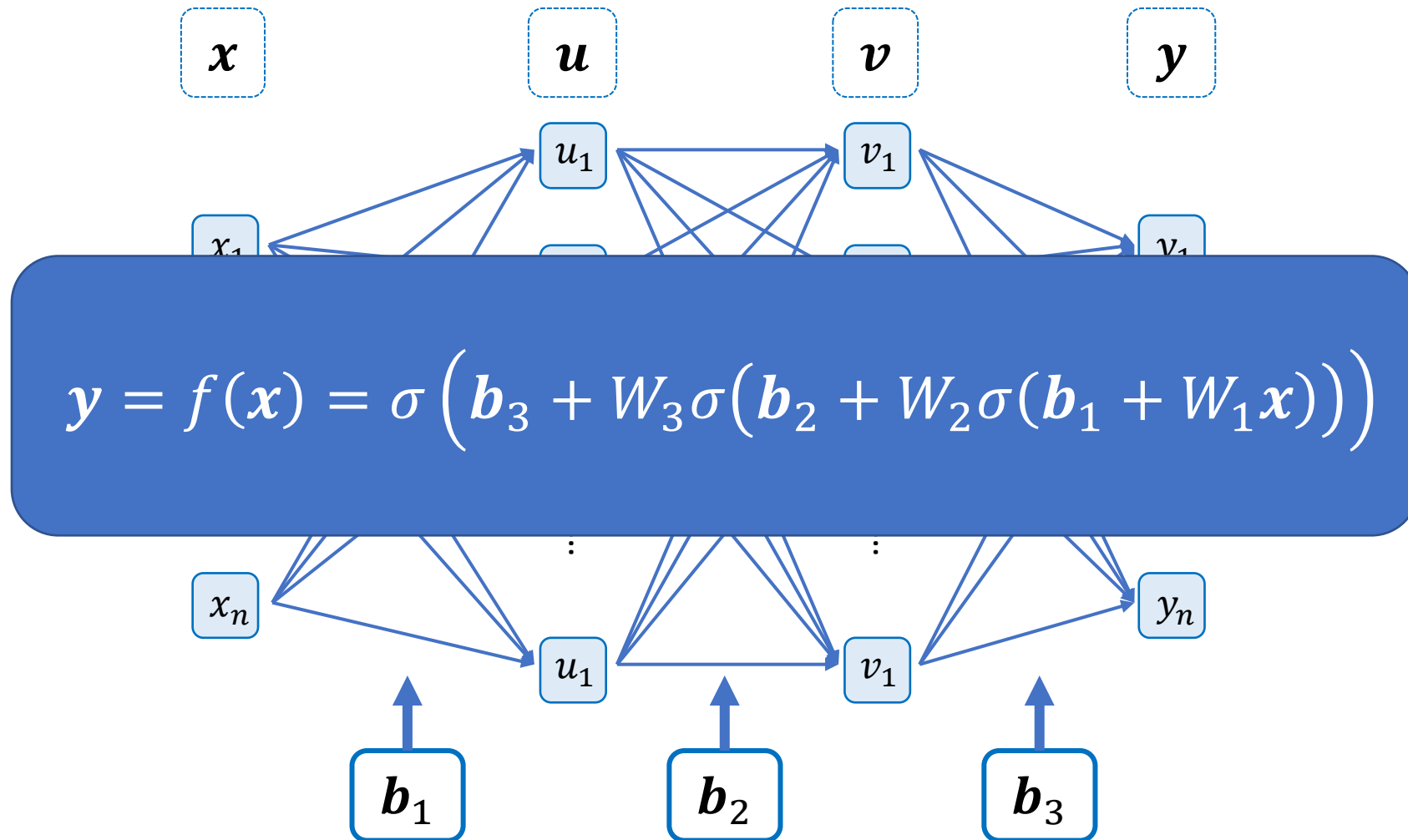


$$y = \sigma(Wx + b)$$

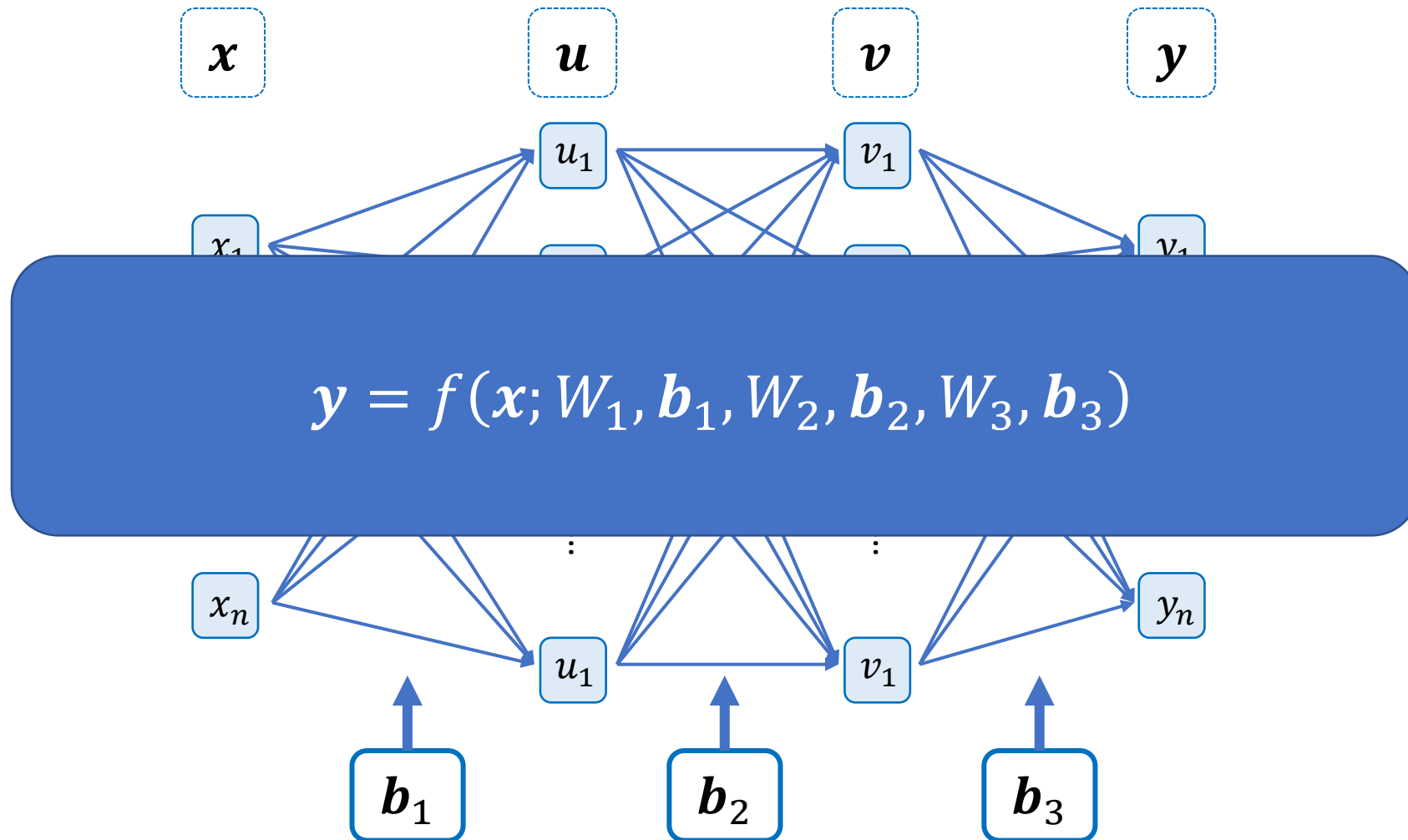
Neural Networks



Neural Networks



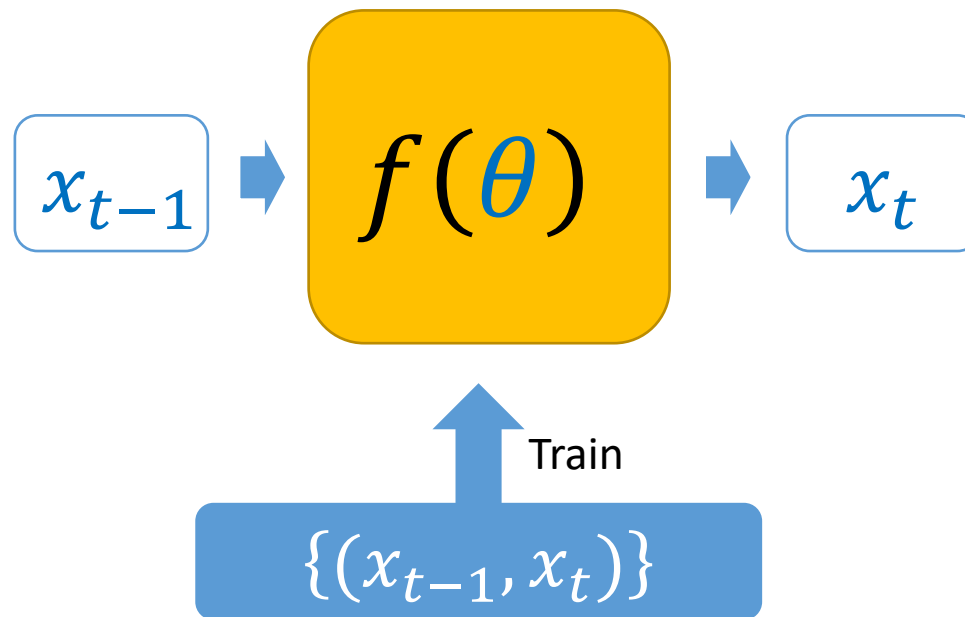
Neural Networks



Learning Motion Models

$$x_t = f(x_{t-1}; \theta)$$

$$\theta = (W_1, \mathbf{b}_1, W_2, \mathbf{b}_2, \dots)$$



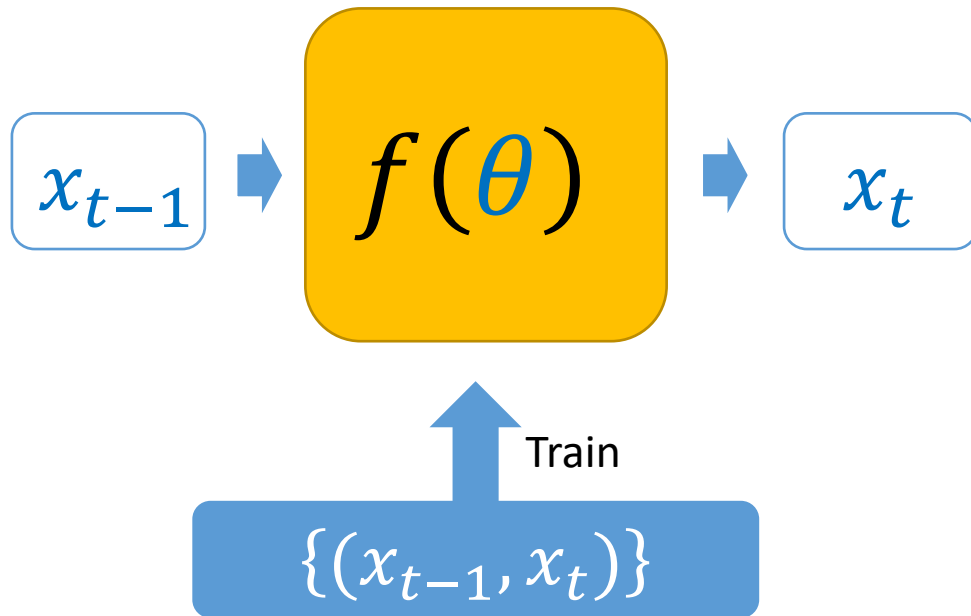
Learning Motion Models

$$x_t = f(x_{t-1}; \theta)$$

$$\theta = (W_1, \mathbf{b}_1, W_2, \mathbf{b}_2, \dots)$$

Given a set of example **transitions**

$$\{(x_{t-1}, x_t)\} \sim p(X)$$



Find $\theta = (W_1, \mathbf{b}_1, W_2, \mathbf{b}_2, \dots)$ that minimizes

$$F(\theta) = \sum_{(x_{t-1}, x_t)} \|f(x_{t-1}; \theta) - x_t\|$$

Learning Motion Models

Stochastic Gradient Descent:

For a batch of random sample $\{(x_{t-1}^{(i)}, x_t^{(i)})\} \sim \{(x_{t-1}, x_t)\}$

Compute the **approximate** gradient

$$\nabla_{\theta} F(\theta) \approx \sum_i \nabla_{\theta} \left(\left\| f \left(x_{t-1}^{(i)}; \theta \right) - x_t^{(i)} \right\| \right)$$

update $\theta = (W_1, \mathbf{b}_1, W_2, \mathbf{b}_2, \dots)$ as

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} F(\theta)$$

Learning Motion Models

Stochastic Gradient Descent:

For a batch of random sample $\{(x_{t-1}^{(i)}, x_t^{(i)})\} \sim \{(x_{t-1}, x_t)\}$

Compute the **approximate** gradient

$$\nabla_{\theta} F(\theta) \approx \sum_i \nabla_{\theta} \left(\left\| f \left(x_{t-1}^{(i)}; \theta \right) - x_t^{(i)} \right\| \right)$$

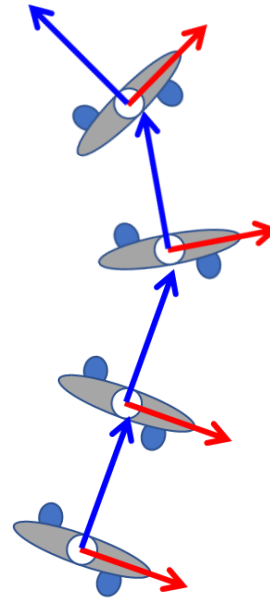
← using
backpropagation

update $\theta = (W_1, \mathbf{b}_1, W_2, \mathbf{b}_2, \dots)$ as

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} F(\theta)$$

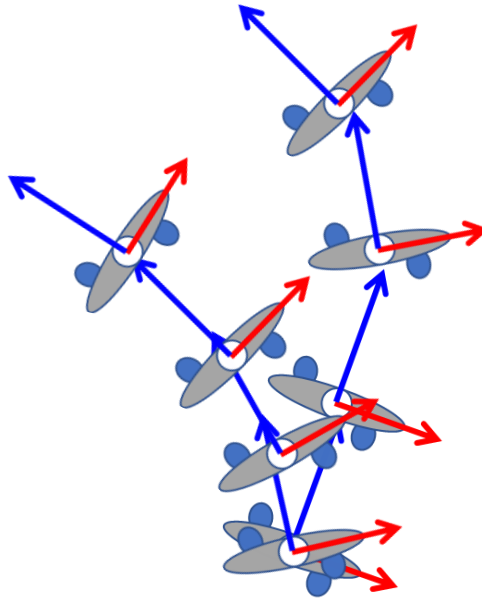
Ambiguity Issue

$$x_t = f(x_{t-1})$$



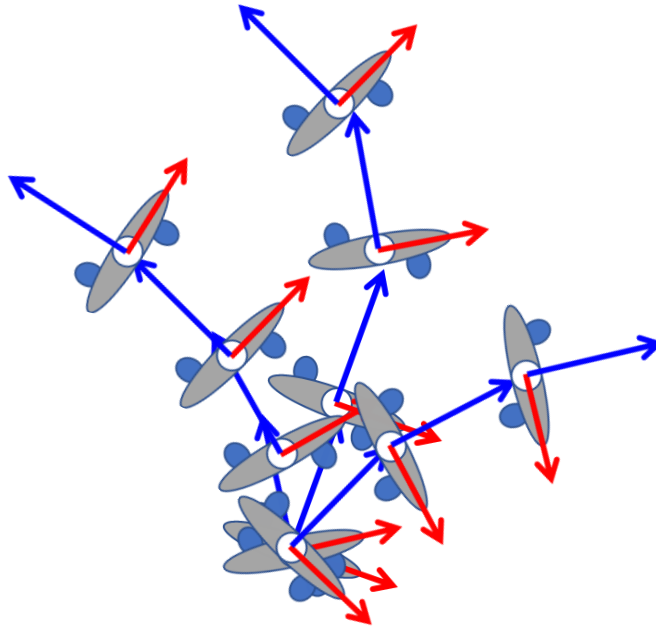
Ambiguity Issue

$$x_t = f(x_{t-1})$$



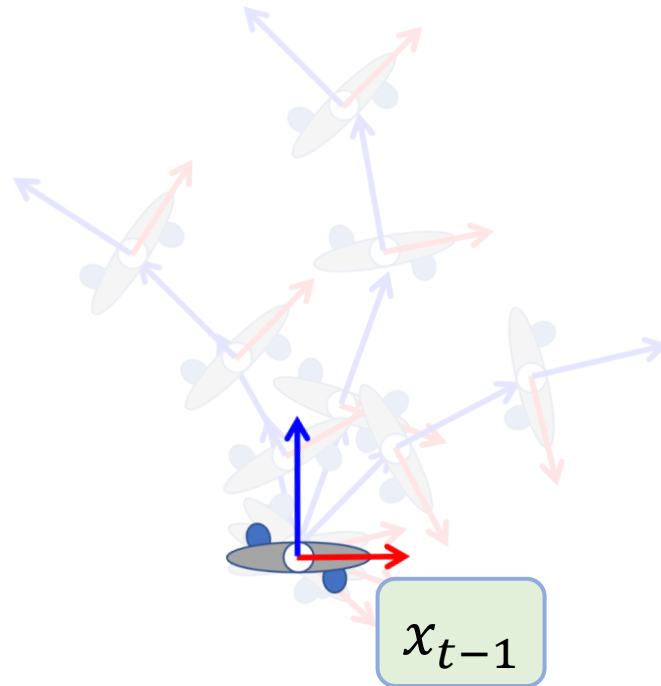
Ambiguity Issue

$$x_t = f(x_{t-1})$$



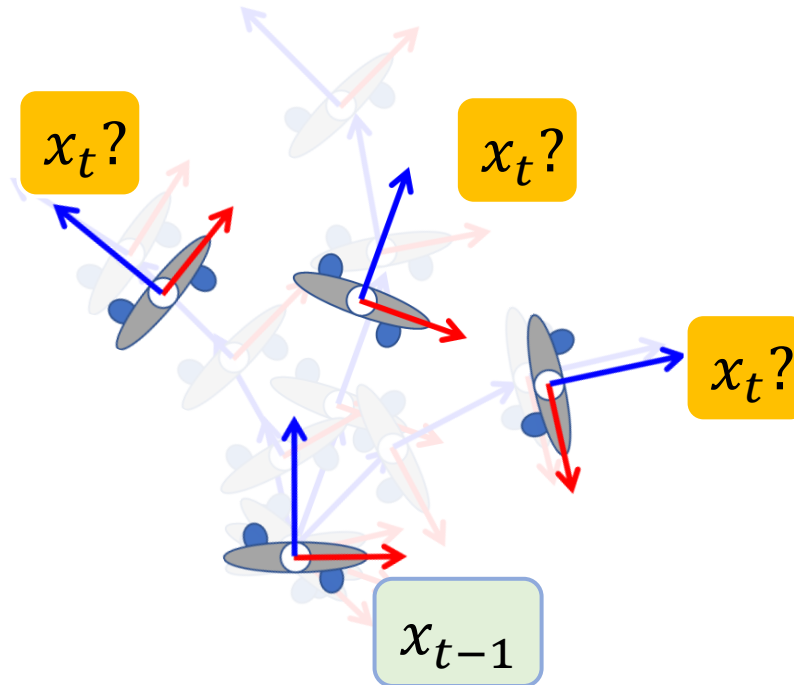
Ambiguity Issue

$$x_t = f(x_{t-1})$$



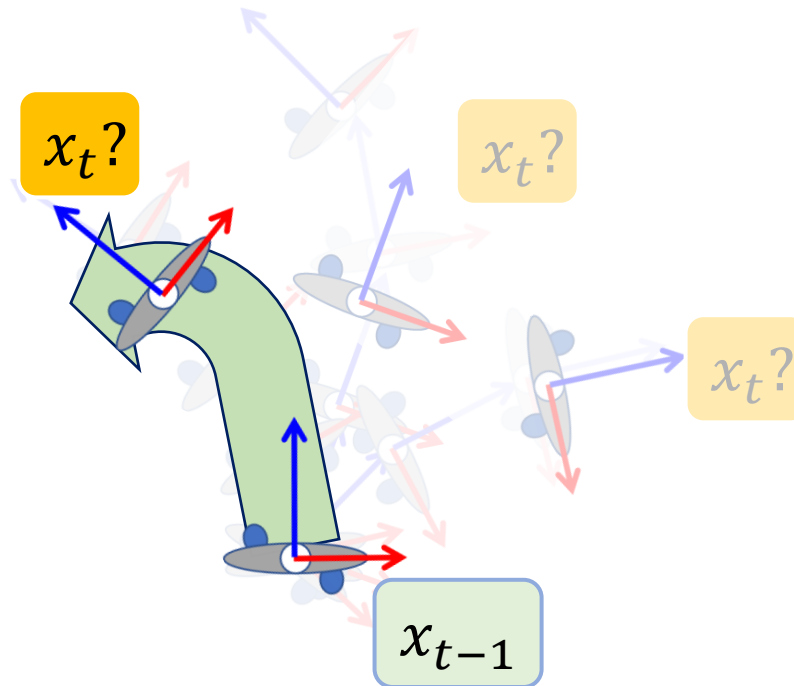
Ambiguity Issue

$$x_t = f(x_{t-1})$$

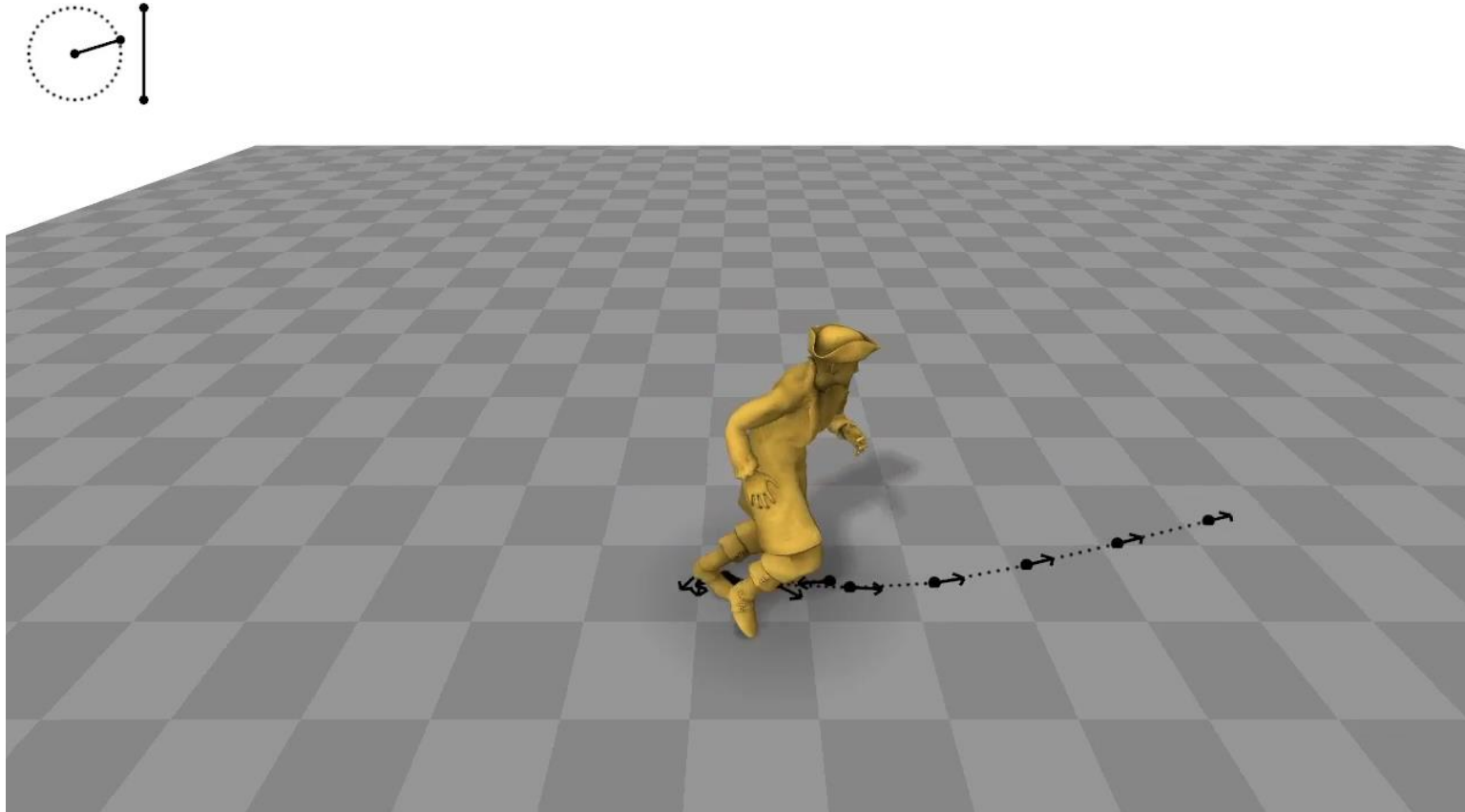


Hidden Variables

$$x_t = f(x_{t-1}; \mathbf{z})$$



PFNN: Phase-Functioned Neural Networks



Phase-Functioned Neural Networks for Character Control

DANIEL HOLDEN, University of Edinburgh

TAKU KOMURA, University of Edinburgh

JUN SAITO, Method Studios

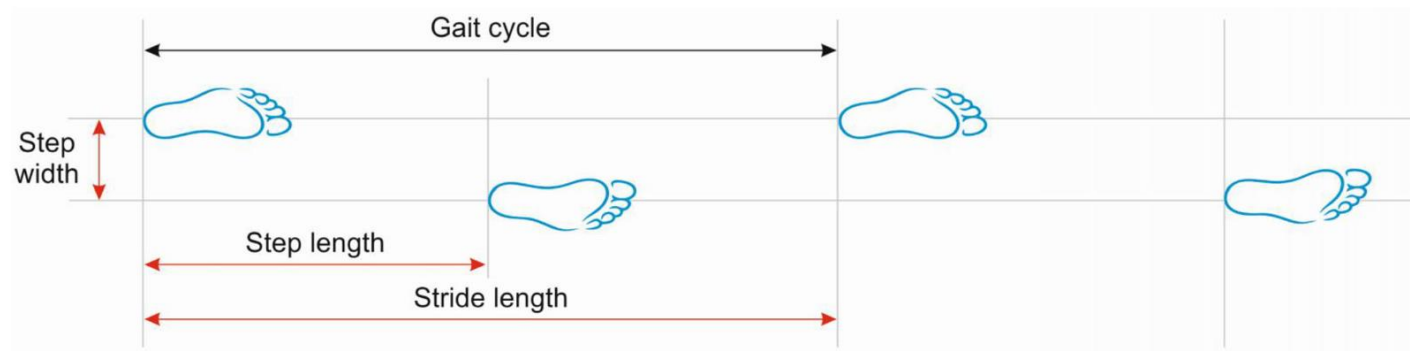
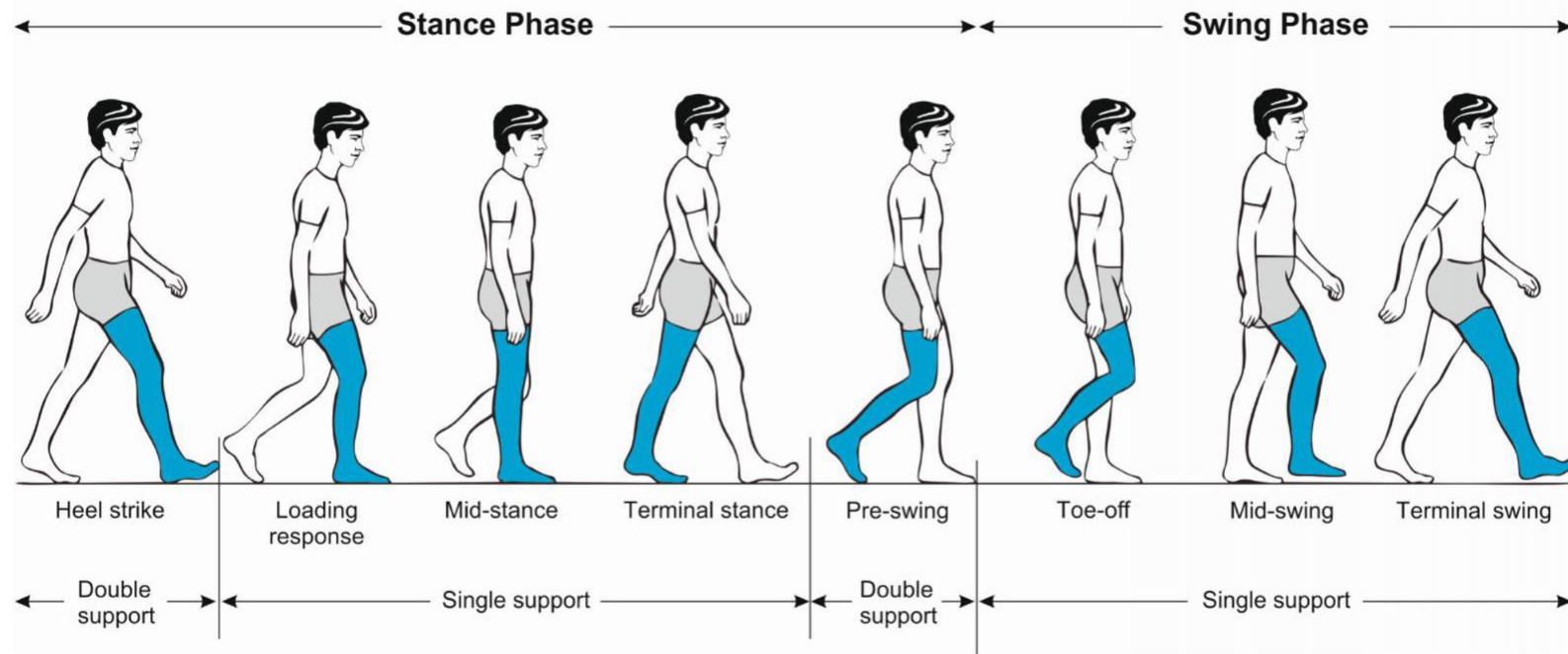
*SIGGRAPH 2017

PFNN: Phase-Functioned Neural Networks

$$x_t = f(x_{t-1}; z_t)$$

$z_t \rightarrow$ control parameters
phase parameter

PFNN: Phase-Functioned Neural Networks

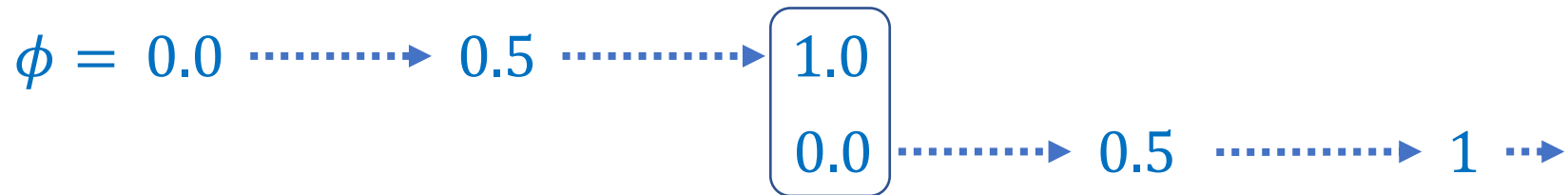
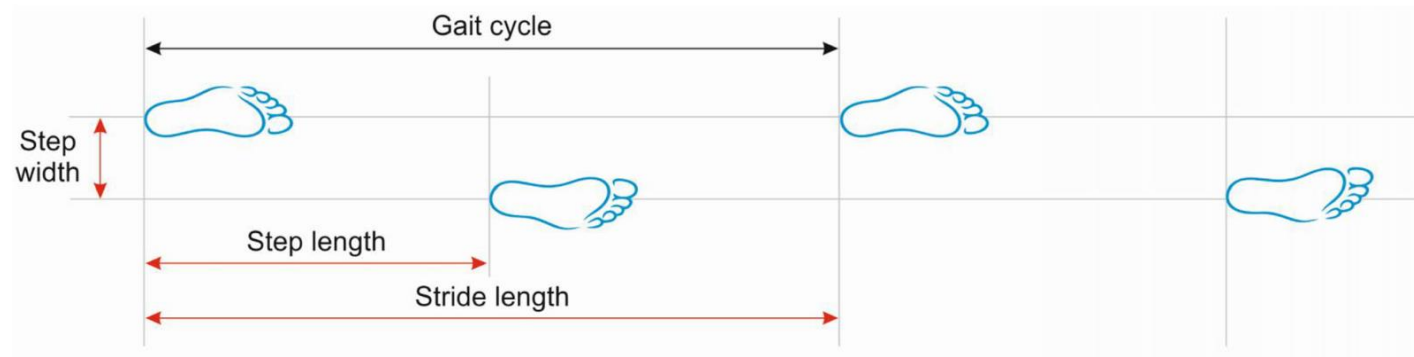


phases of a walking gait cycle

Pirker and Katzenschlager 2017.
Gait disorders in adults and the elderly.

PFNN: Phase-Functioned Neural Networks

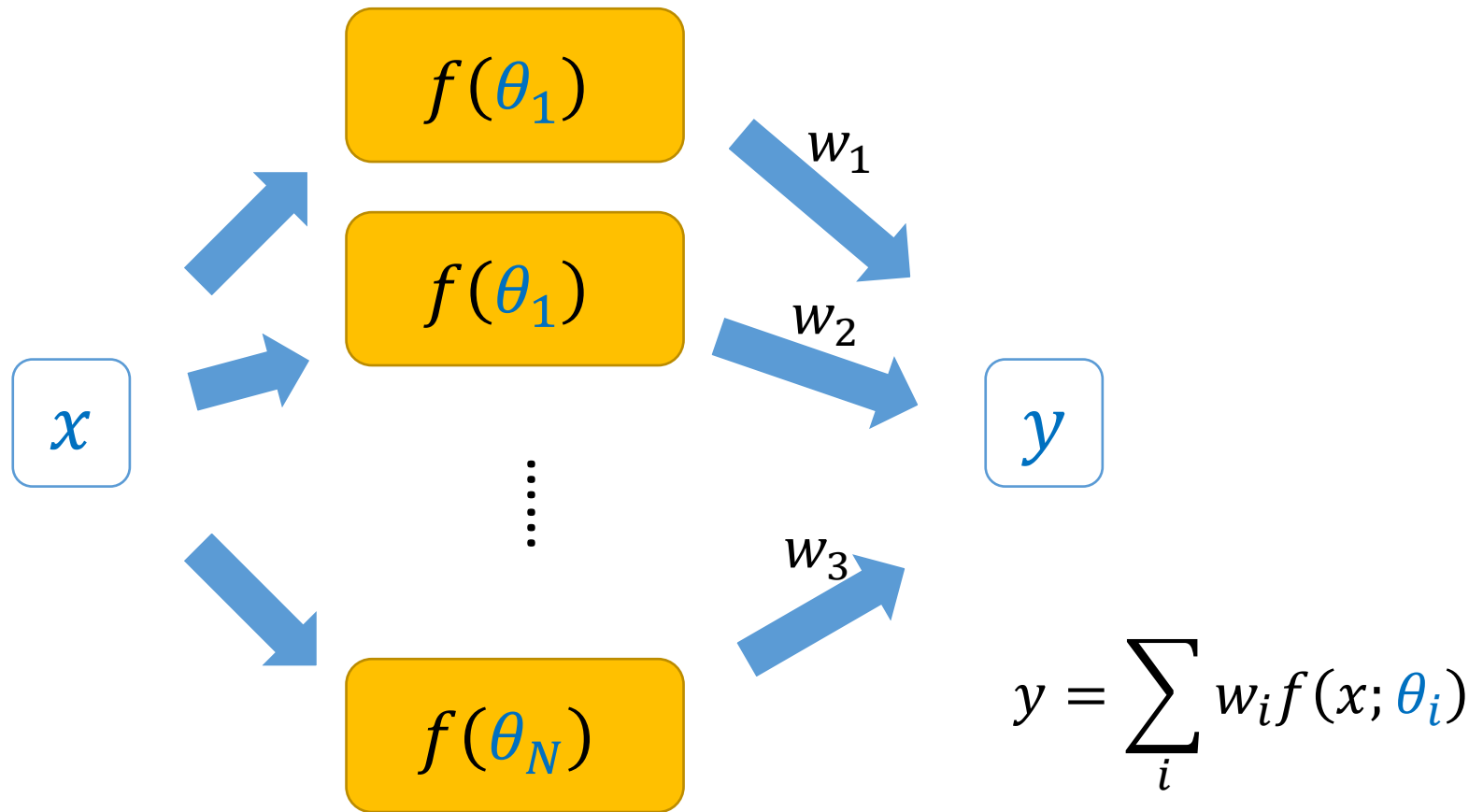
$$x_t = f(x_{t-1}; z_t) \quad z_t = (c_t, \phi_t)$$



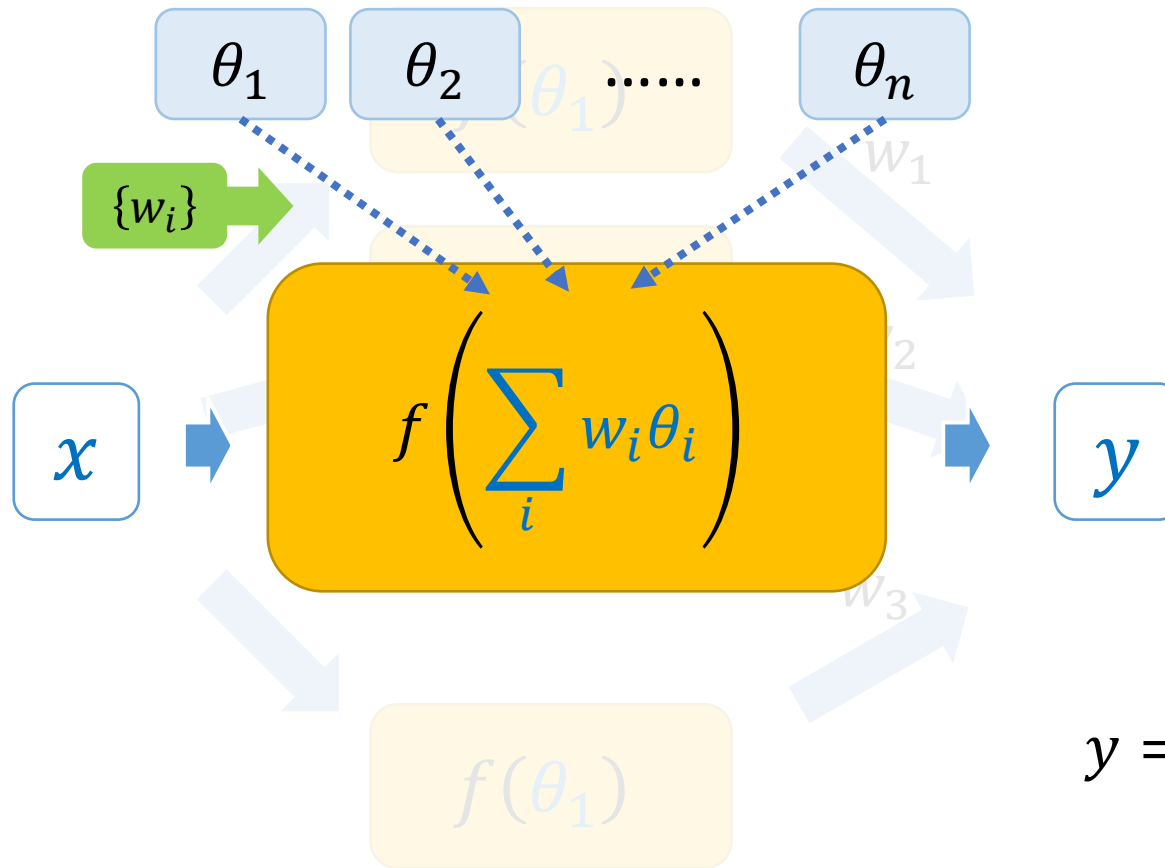
Mixture of Experts



Mixture of Experts



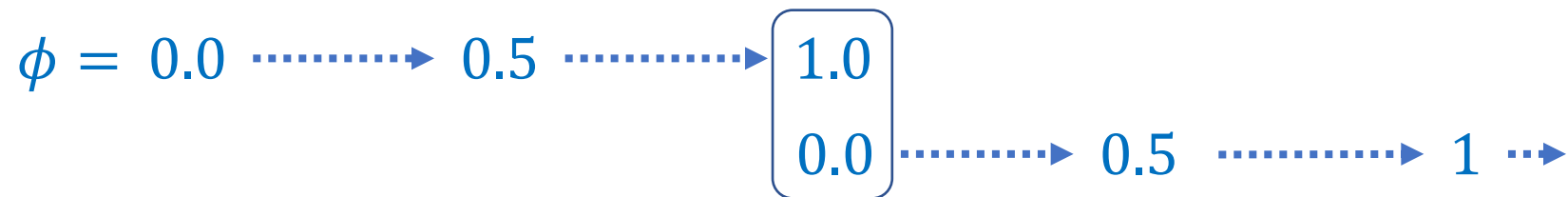
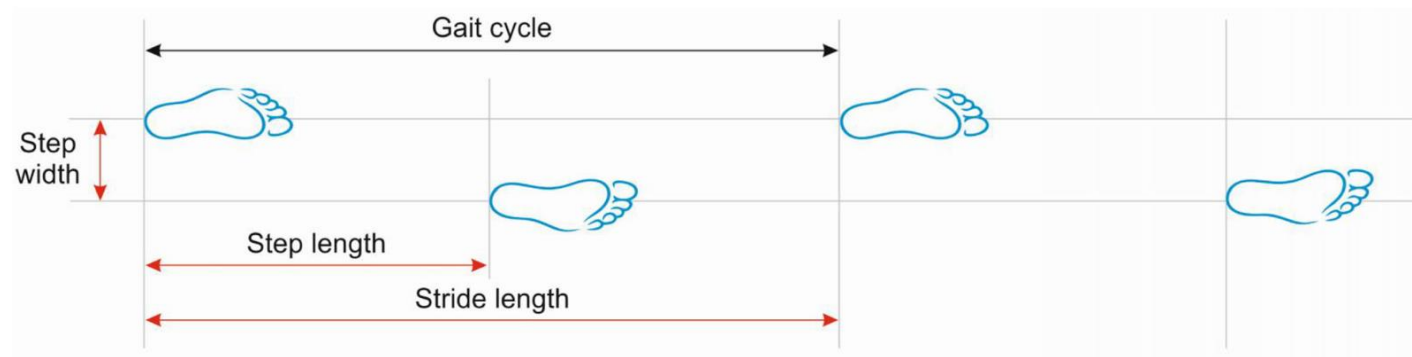
Weighted-Blended Mixture of Experts



$$y = f\left(x; \sum_i w_i \theta_i\right)$$

PFNN: Phase-Functioned Neural Networks

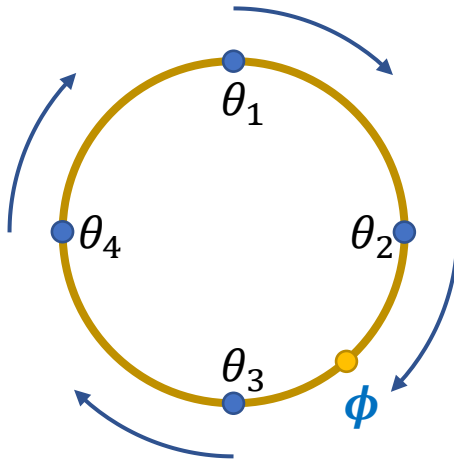
$$x_t = f\left(x_{t-1}; c_t, \theta_t = \sum_i w_i(\phi_t) \theta_i\right)$$



PFNN: Phase-Functioned Neural Networks

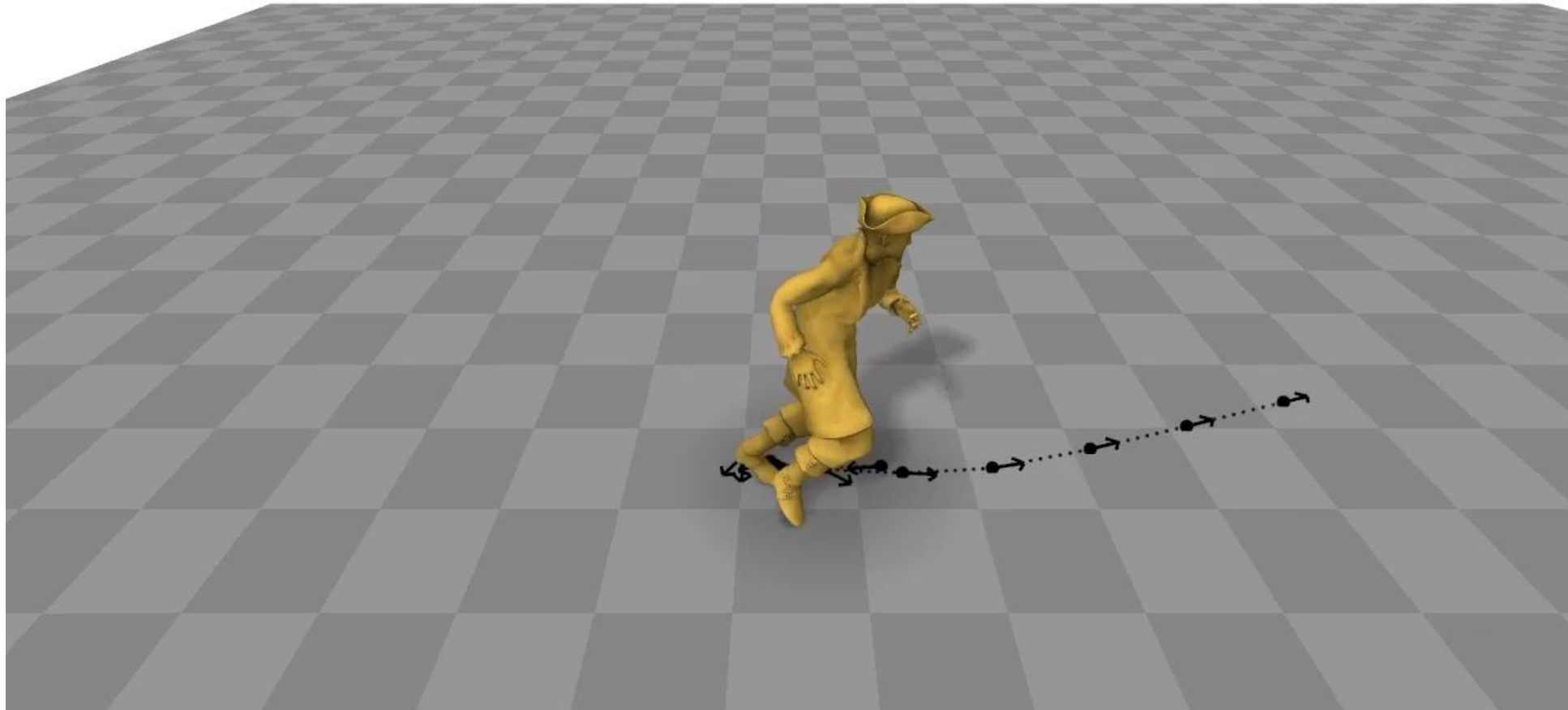
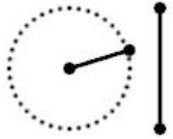
$$x_t = f\left(x_{t-1}; c_t, \theta_t = \sum_i w_i(\phi_t) \theta_i\right)$$

Cubic Catmull-Rom Spline:



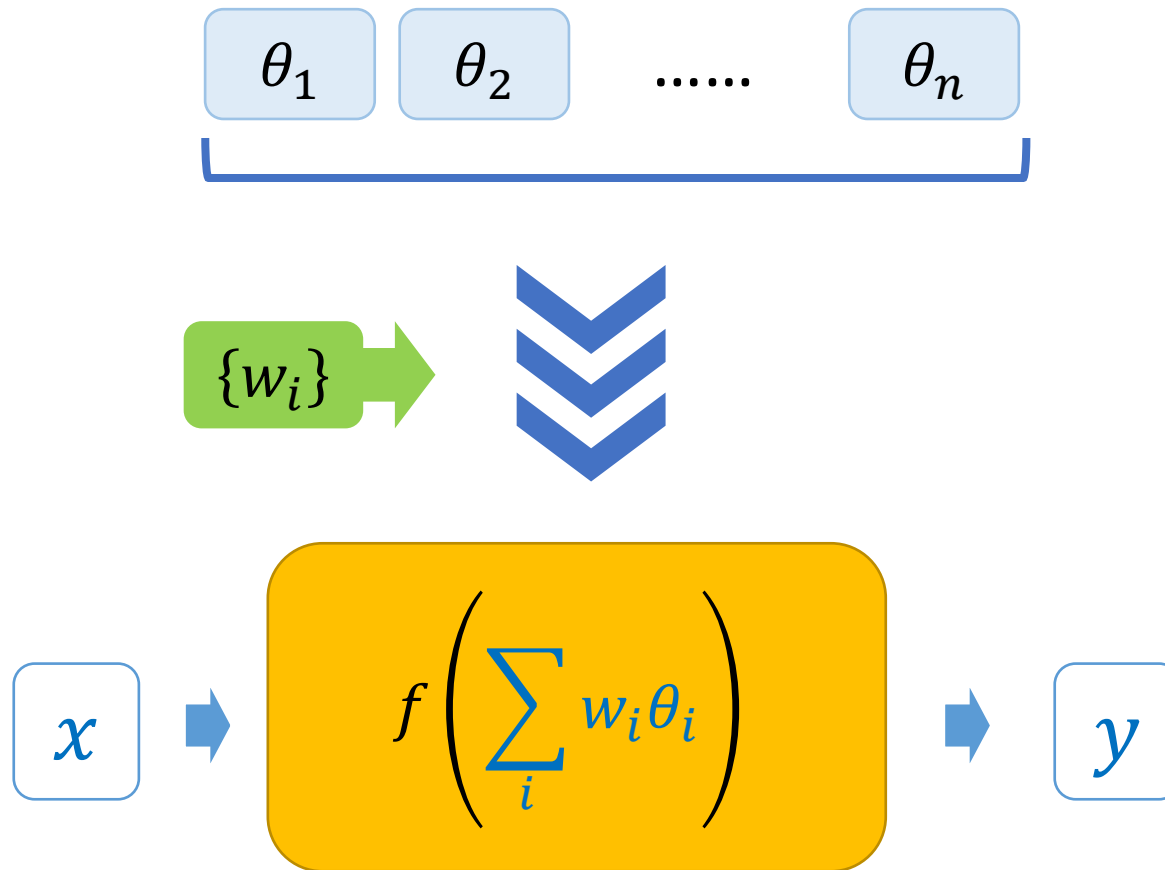
$$\begin{aligned} \theta_t = & \theta_2 \\ & + \phi \left(-\frac{1}{2} \theta_1 + \frac{1}{2} \theta_3 \right) \\ & + \phi^2 \left(\theta_1 - \frac{5}{2} \theta_2 + 2\theta_3 - \frac{1}{2} \theta_4 \right) \\ & + \phi^3 \left(-\frac{1}{2} \theta_1 + \frac{3}{2} \theta_2 - \frac{3}{2} \theta_3 + \frac{1}{2} \theta_4 \right) \end{aligned}$$

PFNN: Phase-Functioned Neural Networks

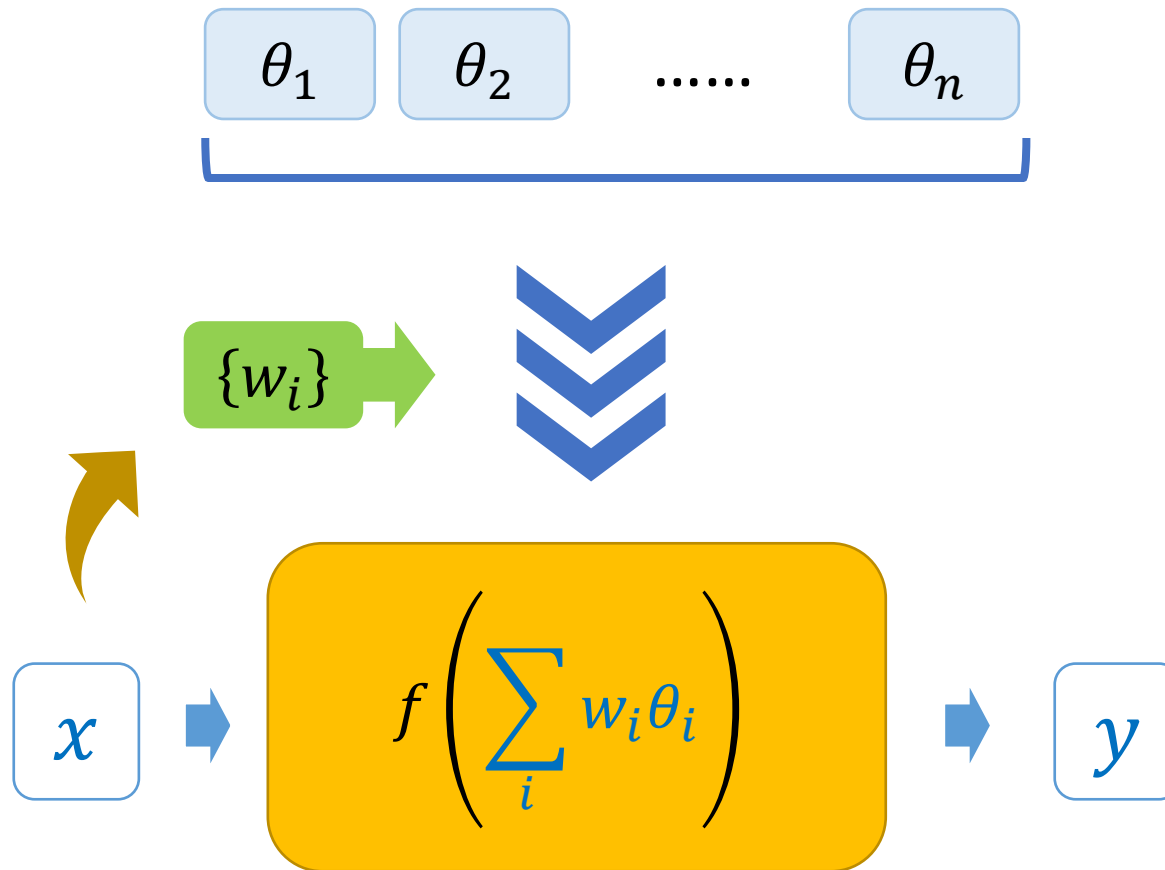


[Holden et al. 2017]

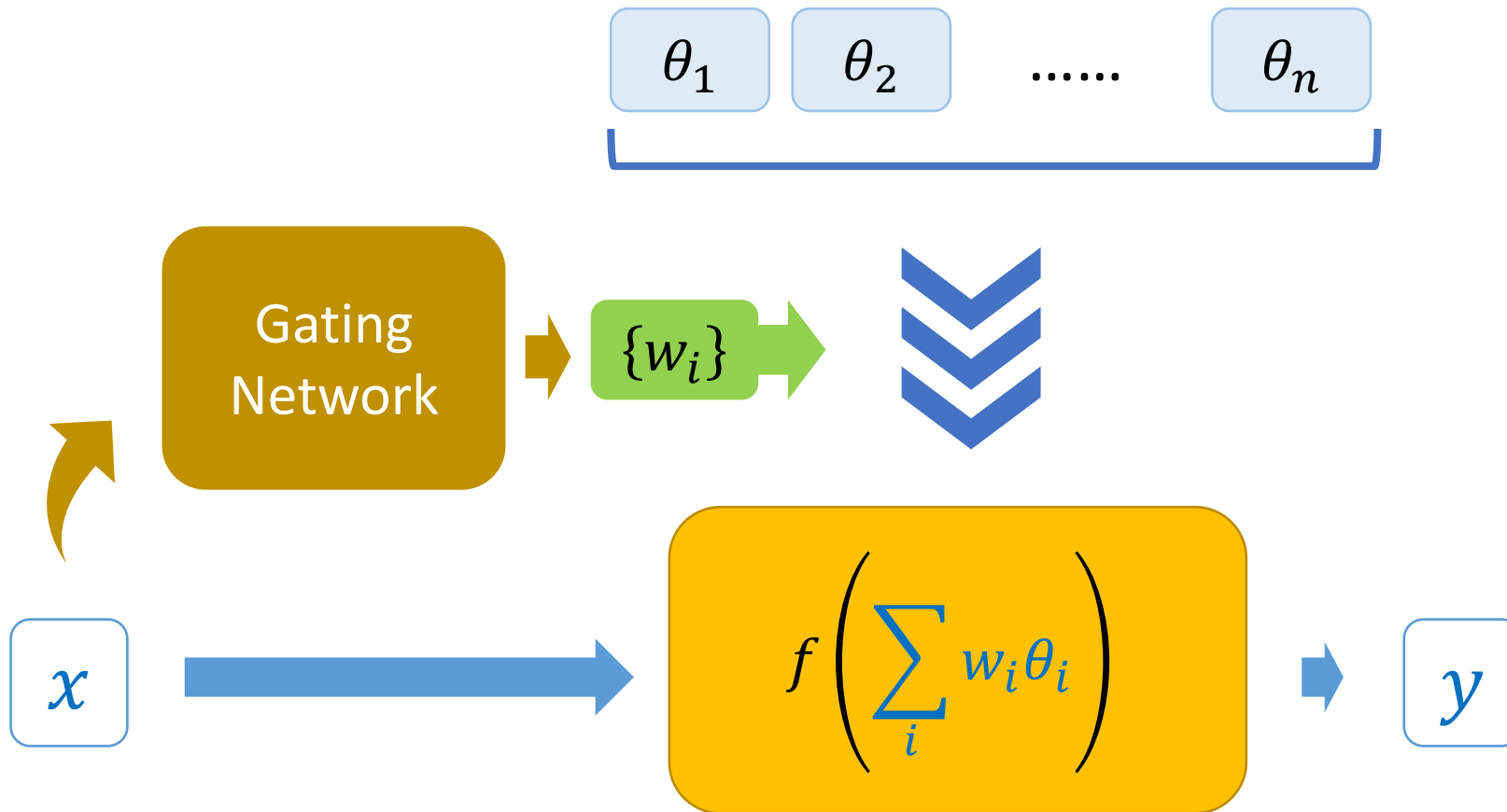
Advanced Phase Functions



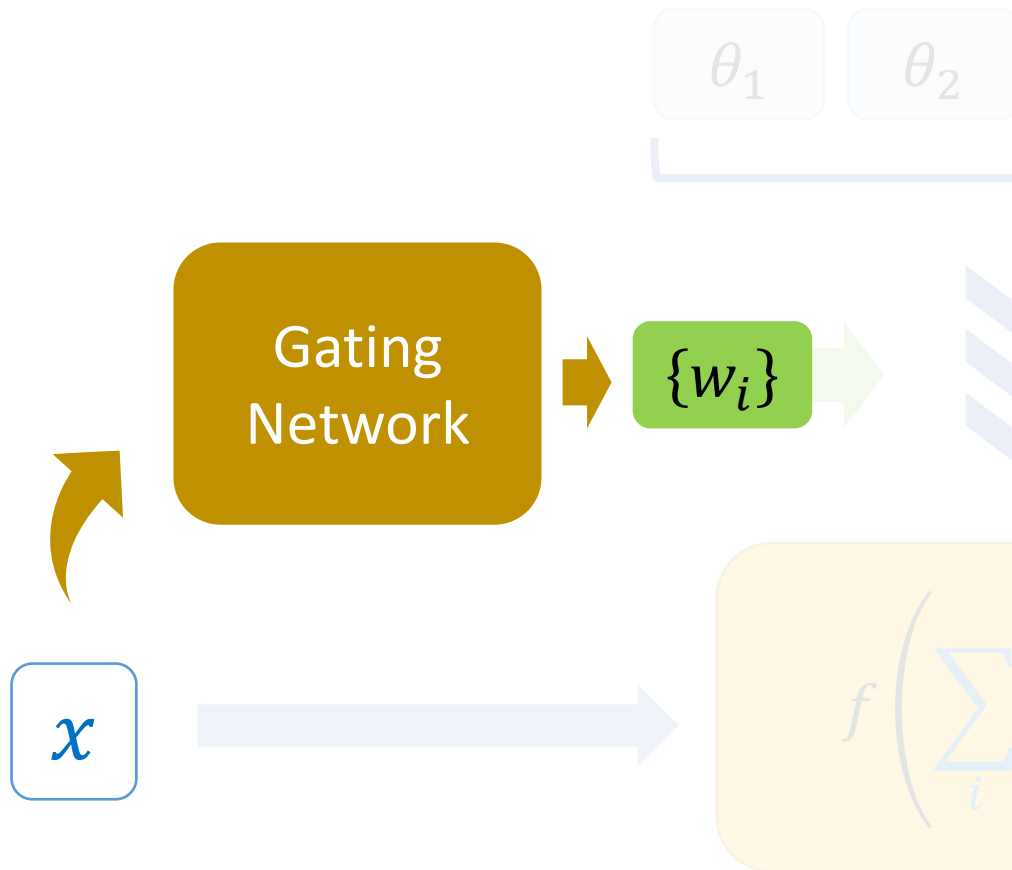
Advanced Phase Functions



Advanced Phase Functions



Advanced Phase Functions



Mode-Adaptive Neural Networks for Quadruped Motion Control

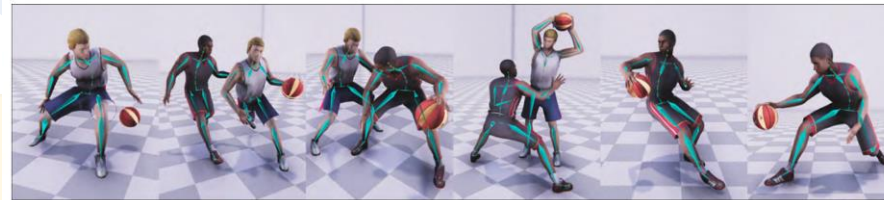
HE ZHANG[†], University of Edinburgh
 SEBASTIAN STARKE[‡], University of Edinburgh
 TAKU KOMURA, University of Edinburgh
 JUN SAITO, Adobe Research



*SIGGRAPH 2018

Local Motion Phases for Learning Multi-Contact Character Movements

SEBASTIAN STARKE, University of Edinburgh, UK and Electronic Arts, USA
 YIWEI ZHAO, Electronic Arts, USA
 TAKU KOMURA, University of Edinburgh, UK
 KAZI ZAMAN, Electronic Arts, USA



*SIGGRAPH 2020

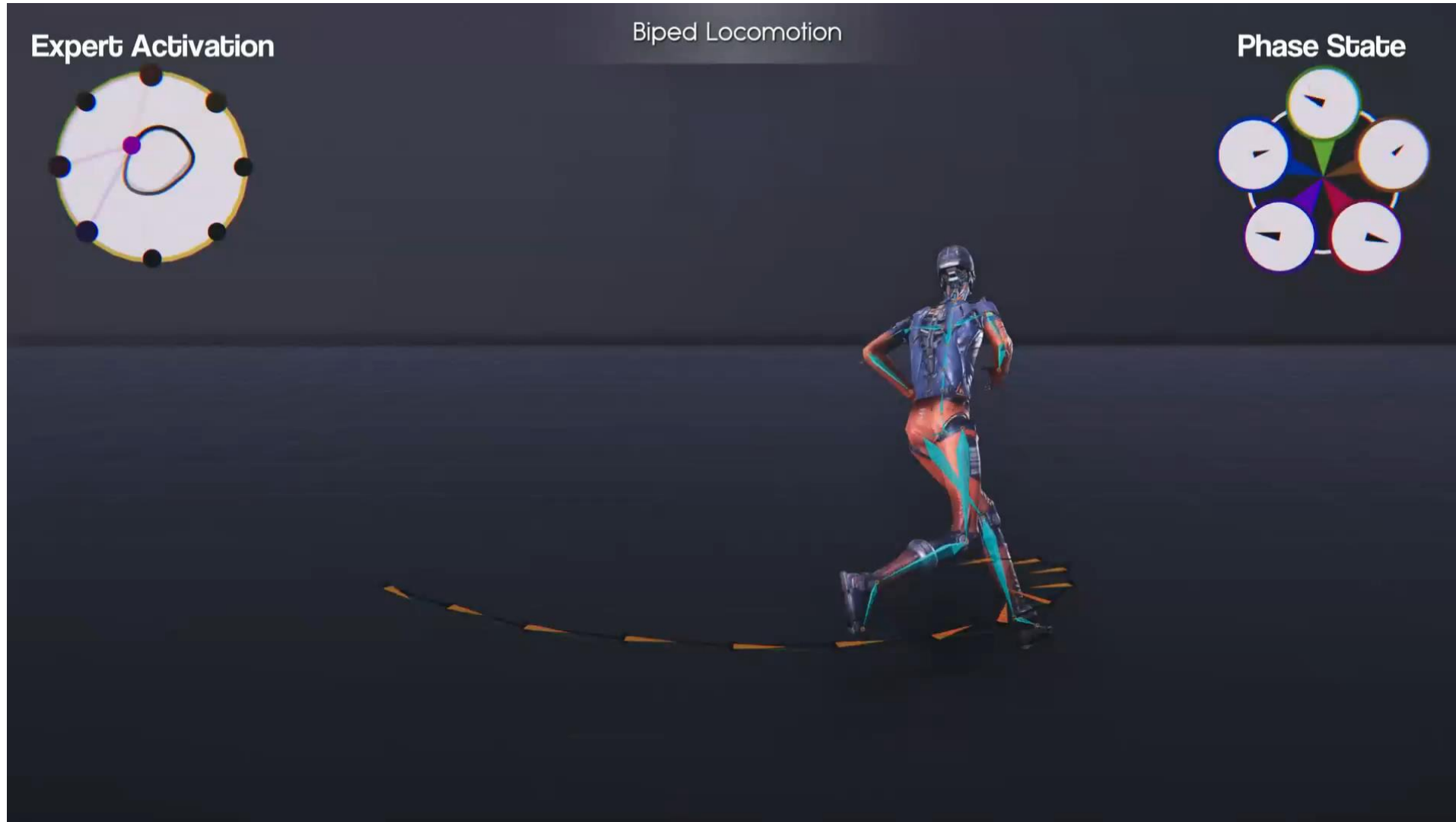
DeepPhase: Periodic Autoencoders for Learning Motion Phase Manifolds

SEBASTIAN STARKE, The University of Edinburgh, UK and Electronic Arts, USA
 IAN MASON, The University of Edinburgh, UK
 TAKU KOMURA, The University of Hong Kong, Hong Kong



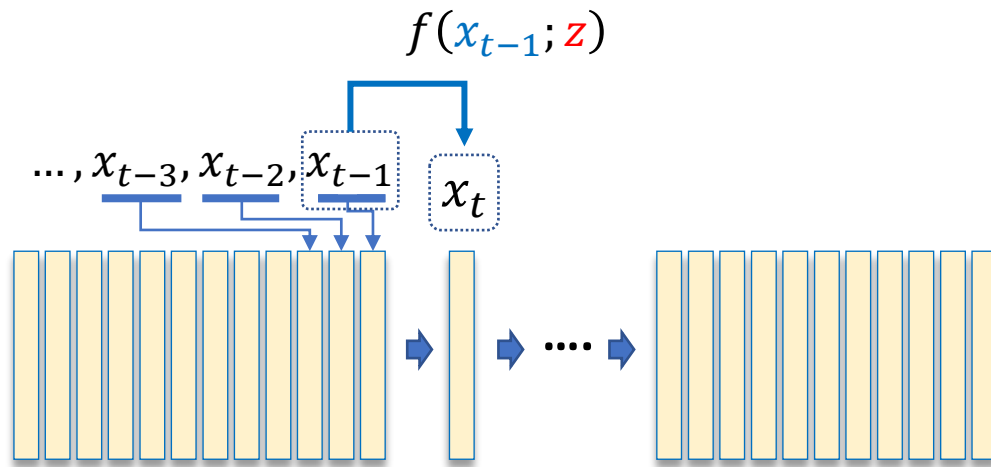
*SIGGRAPH 2022

Advanced Phase Functions



[Starke et al. 2022]

Generative Models



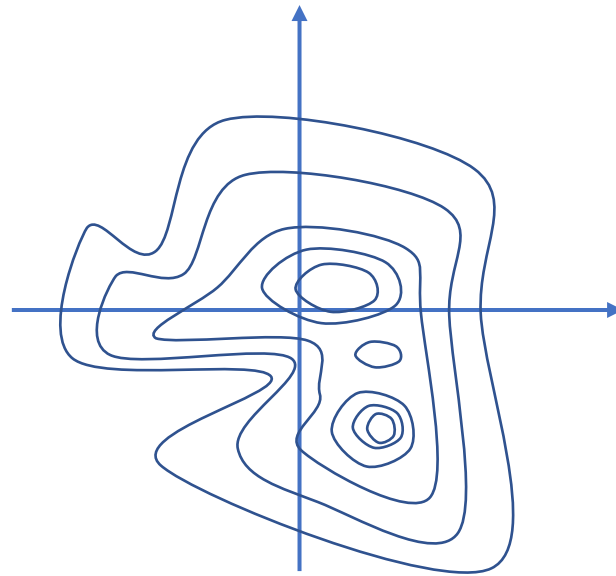
$$p(X|z) = p(x_1, \dots, x_T|z)$$

$$= p(x_1) \prod_t p(x_t|x_{t-1}; z)$$



Generative Models

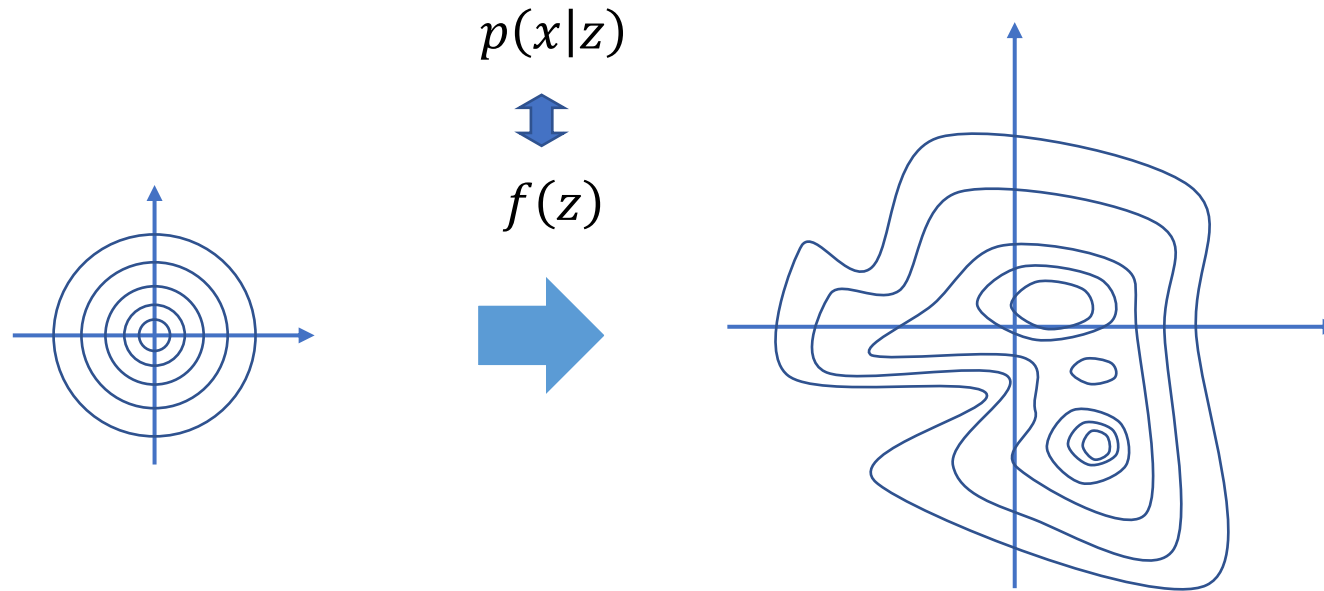
$p(x)$



Generative Models

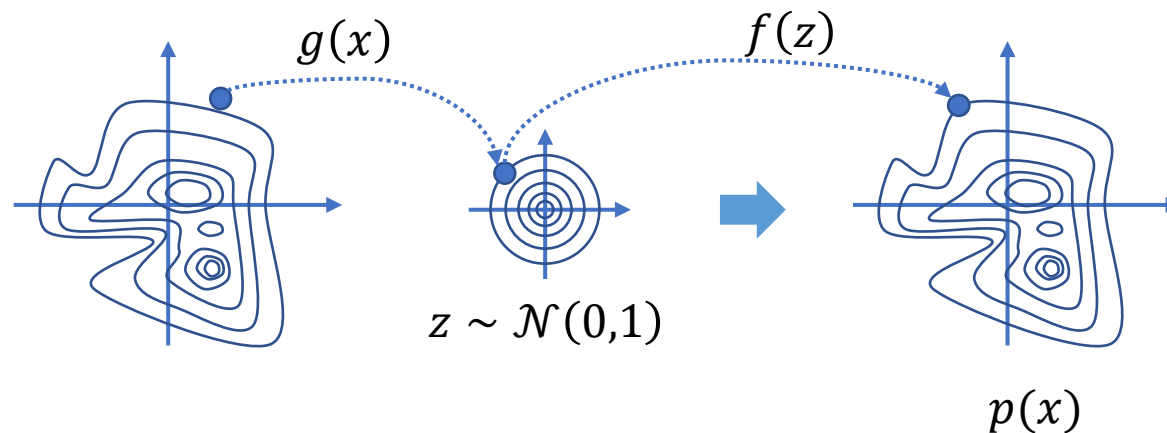
$$z \sim \mathcal{N}(0,1)$$

$$p(x)$$

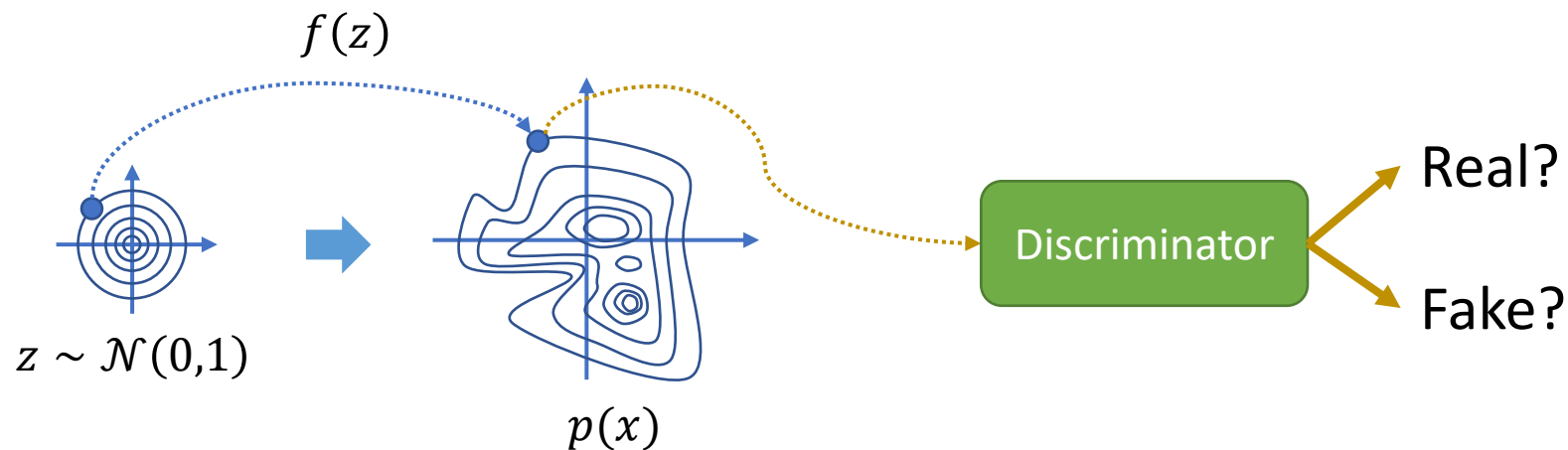


Generative Models

Variational
Autoencoders

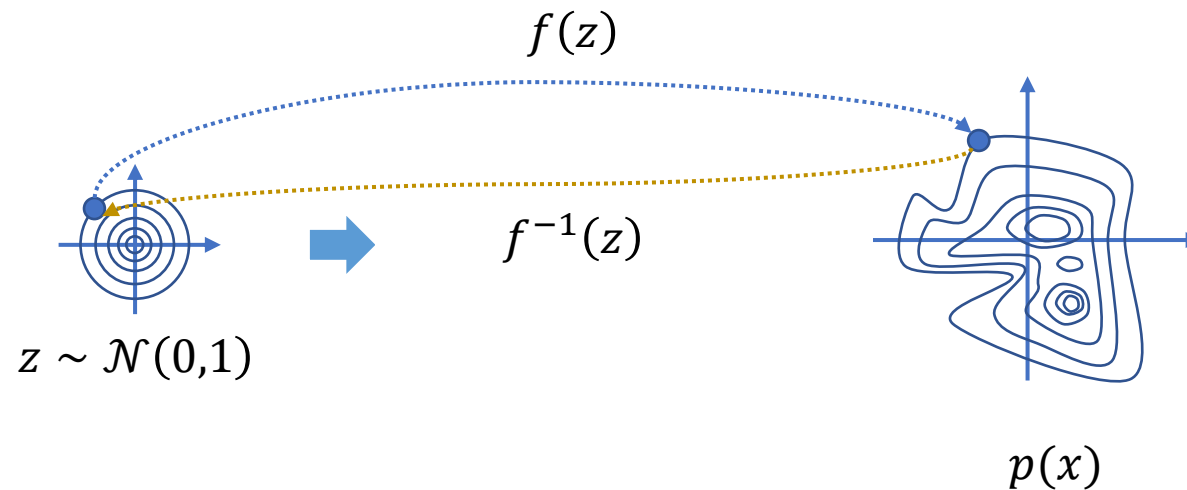


Generative
Adversarial
Network

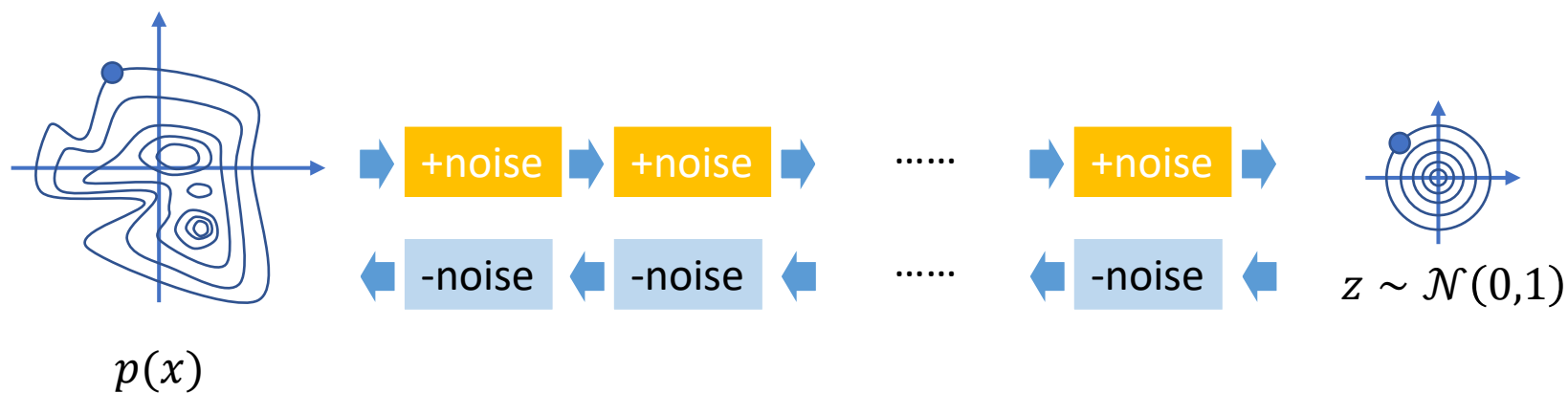


Generative Models

Normalizing Flows



Diffusion Models



Generative Models

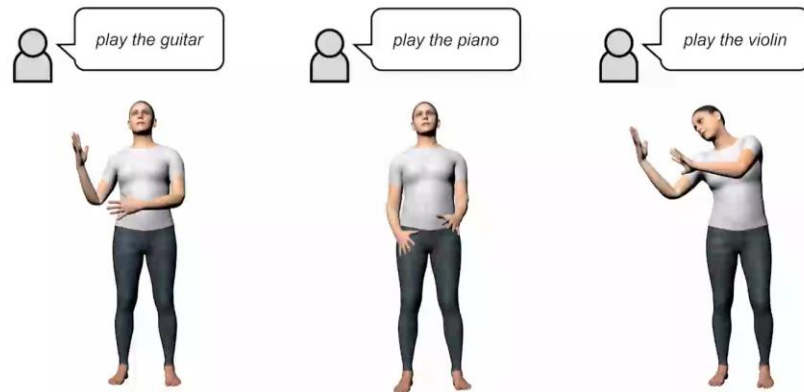


[Ling et al. 2021 Character Controllers Using Motion **VAEs**]



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

Generative Models



[Zhang et al. 2022, [arXiv](#), MotionDiffuse: Text-Driven Human Motion Generation with [Diffusion Model](#)]

MDM: A Human Motion Framework

Diverse Motion



[Tevet et al. 2022, [arXiv](#), MDM: Human Motion [Diffusion Model](#)]

Outline

- Learning-based Character Animation (cont.)
 - Motion Models
 - Autoregressive models: PFNN
 - Generative models

Questions?

