2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Panasonic

Deep Generative Models for Robot Control

Masashi OKADA, Ph.D
Senior staff engineer, Panasonic

Lecture @ IROS2020 Tutorial sponsored by RSJ

Tutorial on Deep Probabilistic Generative Models for Robotics

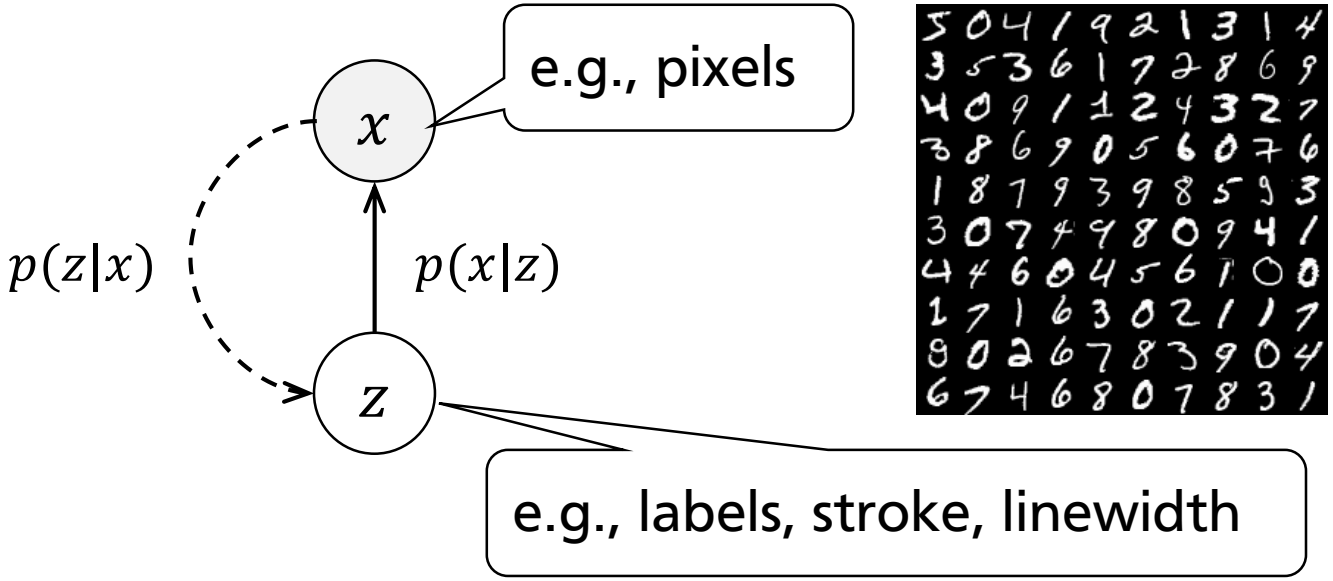
978-1-7281-6211-9/20/$31.00 ©2020 IEEE

Table of Contents

- A basic deep generative model: variational autoencoder (VAE)
  - Theory
  - Applications
    - World model, PlaNet, Dreamer, etc…
- Control as probabilistic inference
  - Theory
  - Applications
    - SAC, SLAC, VI-MPC, PaETS, PlaNet-Bayes

Problem formulation of VAE

- Variables
  - Observation : x
  - Latent : z



- What we want:
  - $p(x, z)$
  - $= p(z|x)p_{\mathcal{D}}(x)$
    - We can infer the latent z from an observation x
  - $= p(x|z)p(z)$
    - We can generate samples x from latent z drawn from a prior $p(z)$

748

How to derive the models?

- Define parameterized <u>deep generative models</u>
  - $p(x, z) = p_\theta(x|z)p(z)$
    - θ: model parameter (i.e., weight of neural networks)
    - $p(z)$: prior (e.g., standard normal dist.)
- Find θ^* that maximizes the data log-likelihood (MLE: Maximum Likelihood Estimation)
  - $\theta^* = \mathrm{argmax}_\theta \log p(x) = \mathrm{argmax}_\theta \log \int p_\theta(x|z)p(z)dz$
- However, this is intractable, because …
  - No analytic solution
  - No efficient estimator

How to make it tractable

- $\log p(x) = \log \int p_\theta(x|z)p(z)dz$
- $= \log \int p_\theta(x|z)p(z)\frac{q_\phi(z|x)}{q_\phi(z|x)}dz$
- $\geq \int q_\phi(z|x)\log\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}dz$
- $:= J_{\mathrm{ELBO}}$ (ELBO: Evidence Lower Bound)
  - J_{ELBO} can be efficiently optimized by using general deep learning techniques (discussed later)

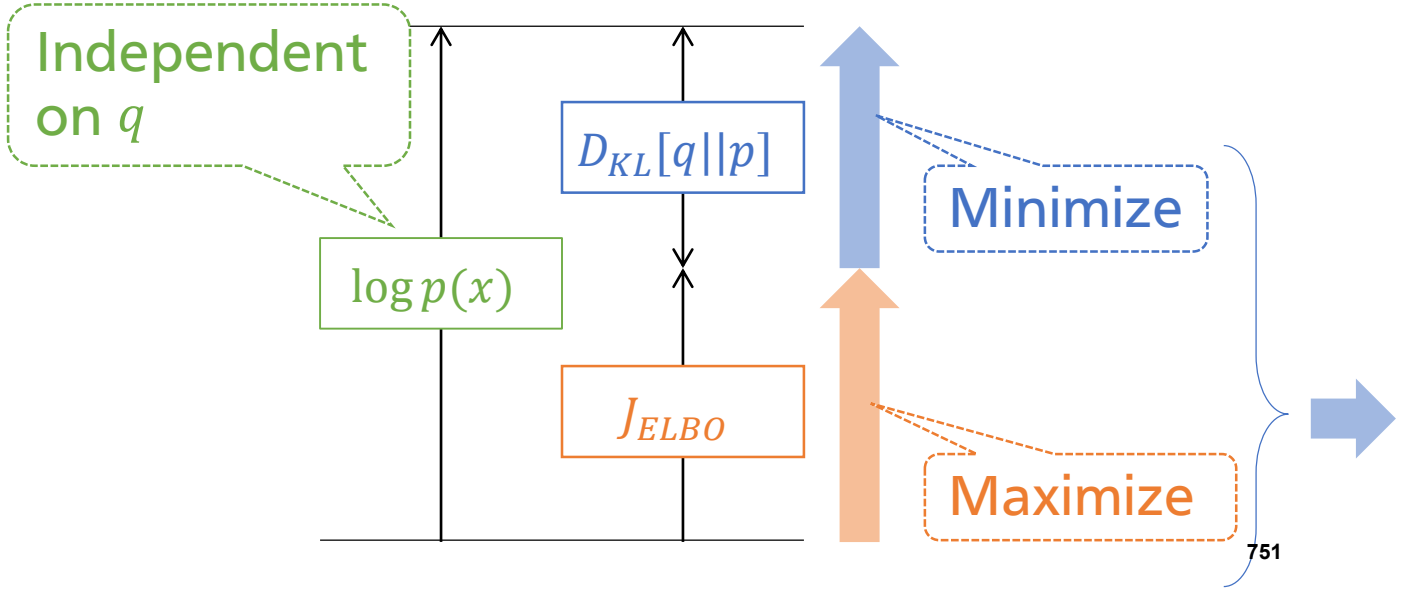
$q_\phi(z|x)$: *variational distribution* to approx. the posterior $p(z|x)$

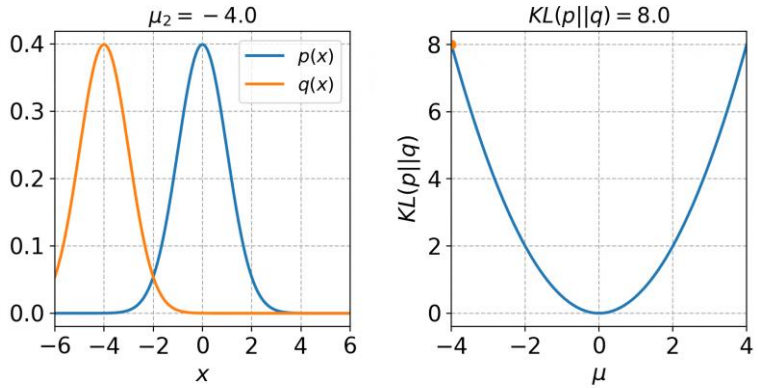
Jensen's inequality $\log \mathbb{E}[x] \geq \mathbb{E}[\log x]$

750

What ELBO also optimizes?

- $\log p(x) - J_{\mathrm{ELBO}} = D_{KL}[q_\phi(z|x)||p(z|x)]$
  - $D_{KL}[q||p] \left(:= \int q \log \frac{q}{p} dz\right)$: Kullback-Leibler divergence (KL-divergence)
    - A measure of how q, p are different





https://qiita.com/ceptree/items/9a473b5163d5655420e8

ELBO max. is equivalent to variational inference $D_{KL}[q||p]$

How to compute J_{ELBO}

$$J_{\mathrm{ELBO}} := \int q_\phi(z|x) \log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} dz$$

$$= \int q_\phi(z|x) \log p_\theta(x|z)\, dz - \int q_\phi(z|x) \log \frac{q_\phi(z|x)}{p(z)} dz$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\phi(z|x)||p(z)]$$

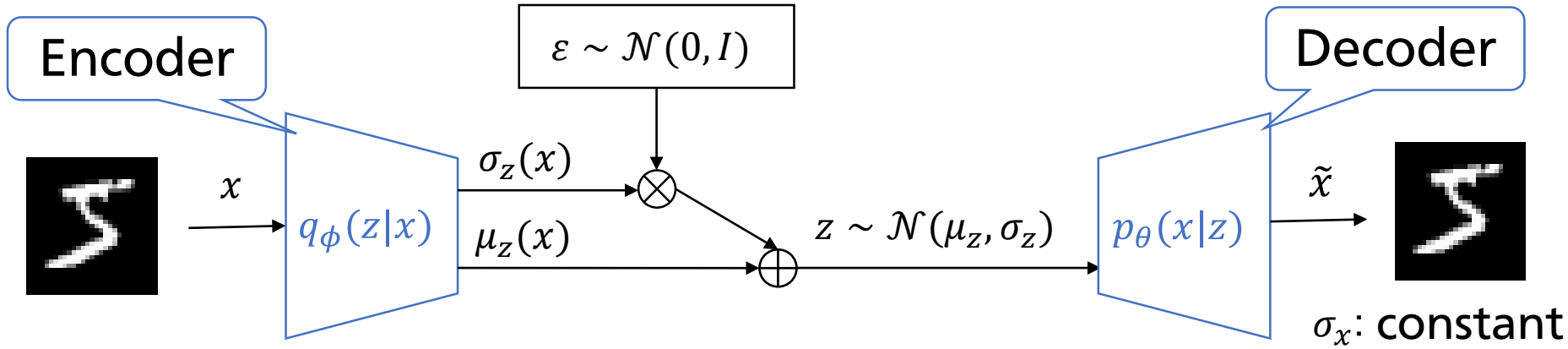
Log-likelihood of autoencoded x

Regularizer to encourage $q_\phi(z|x)$ to be close to the prior $p(z)$

752

How to compute J_{ELBO}

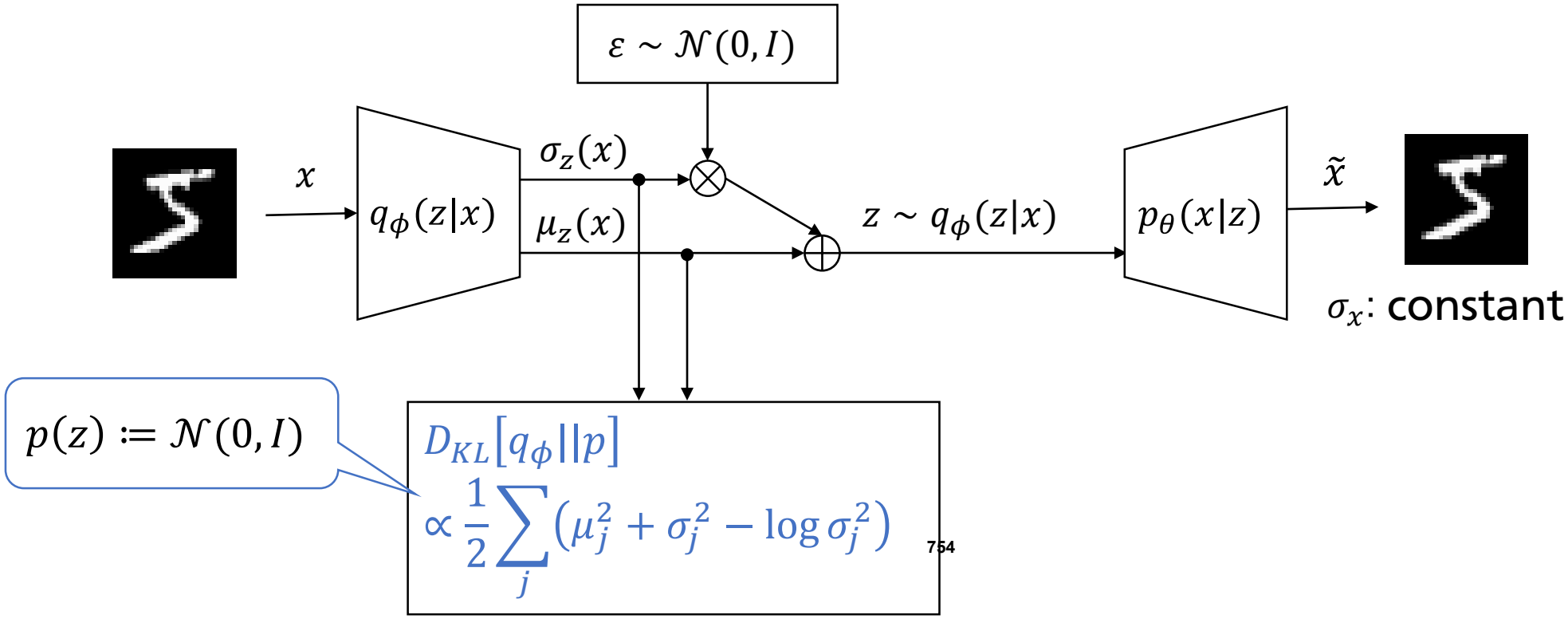
- Computational Graph with Deep Gaussian Modeling of p_θ, q_ϕ
  - $J_{\mathrm{ELBO}} = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\phi(z|x)||p(z)]$



753

How to compute J_{ELBO}

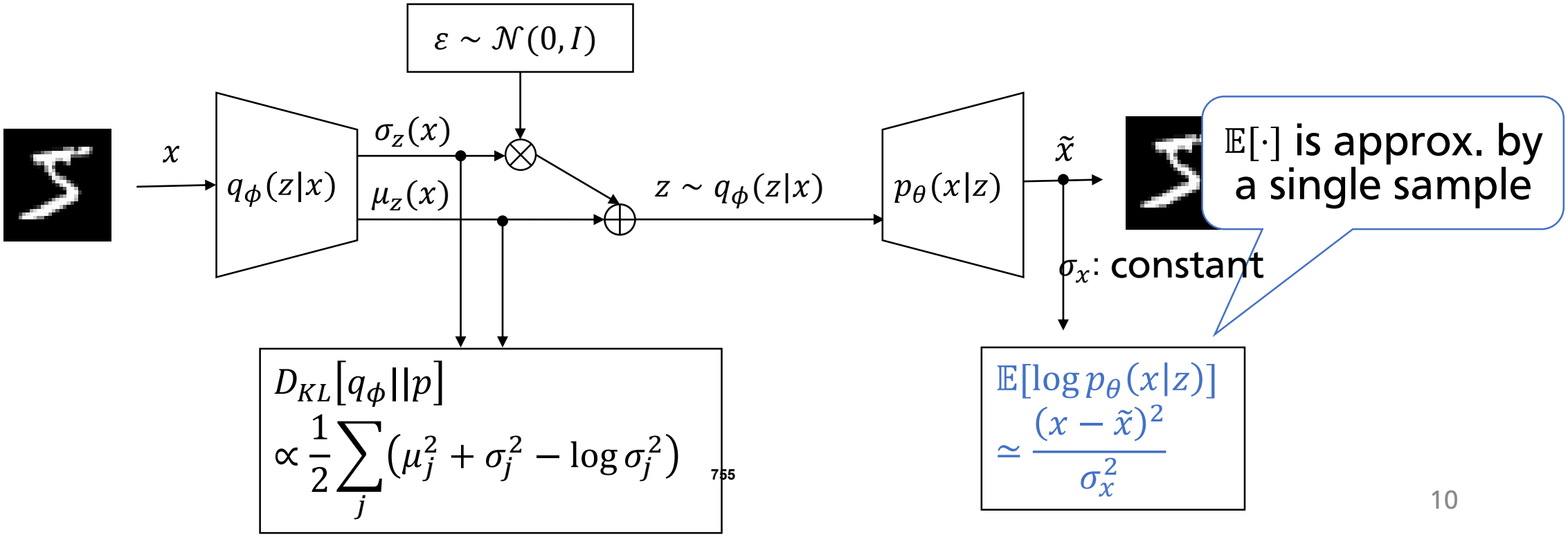
- Computational Graph with Deep Gaussian Modeling of p_θ, q_ϕ
  - $J_{\mathrm{ELBO}} = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left[q_\phi(z|x)||p(z)\right]$



754

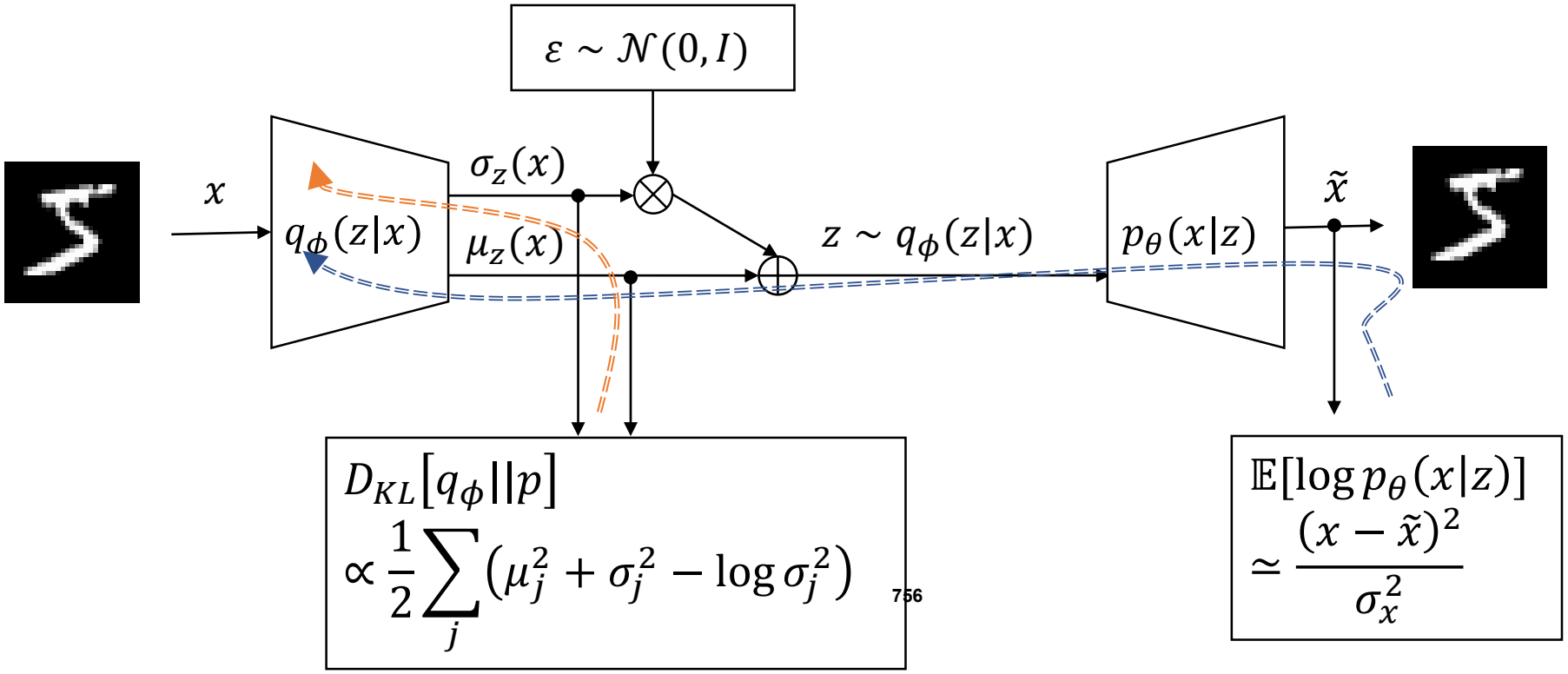
How to compute J_{ELBO}

- Computational Graph with Deep Gaussian Modeling of p_θ, q_ϕ
  - $J_{\mathrm{ELBO}} = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}[q_\phi(z|x)||p(z)]$

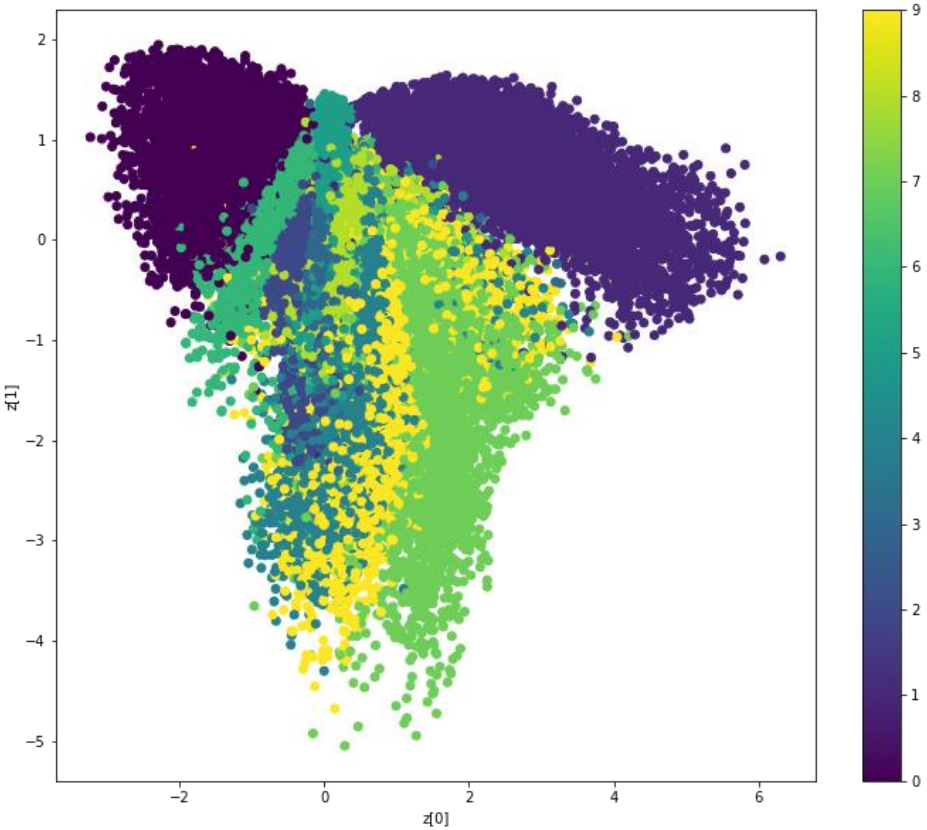


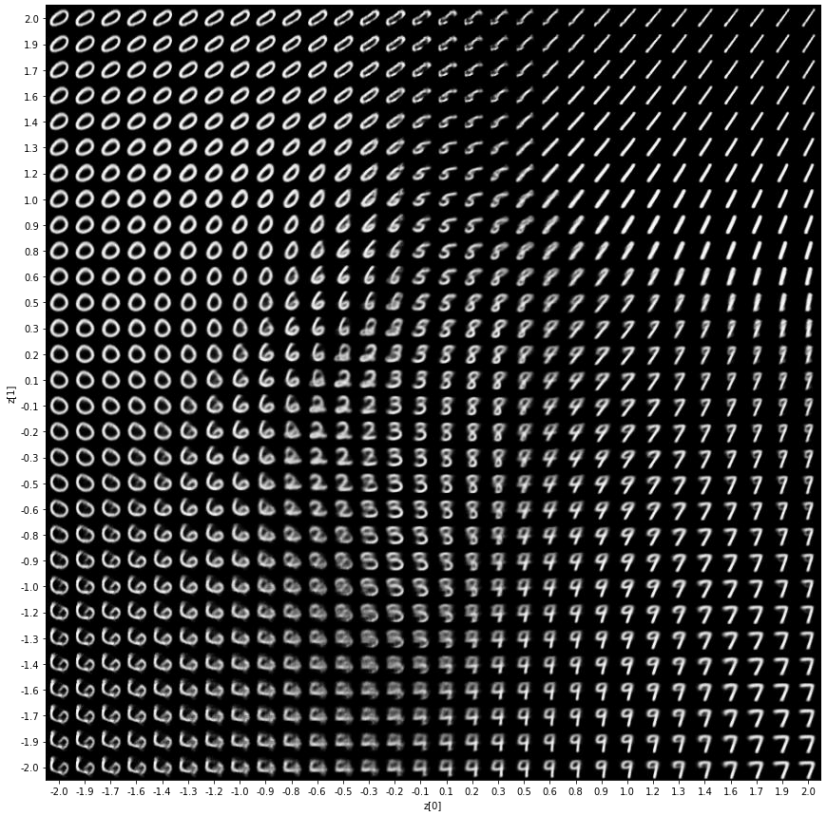
How to compute J_{ELBO}

All forward operations are differentiable
$\Rightarrow \nabla_{\theta,\phi} J_{\mathrm{ELBO}}$ can be efficiently computed by back-prop.



How MNIST data are encoded

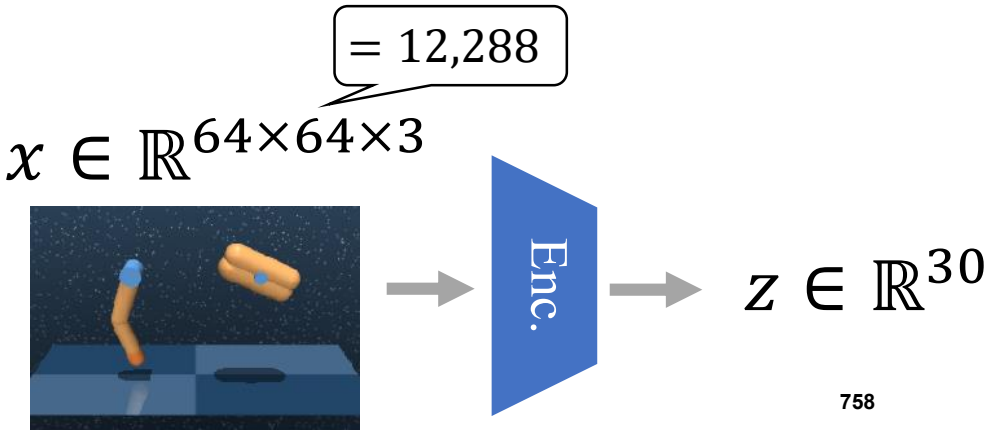




https://keras.io/examples/generative/vae/ [757]

Application of VAE to robotics

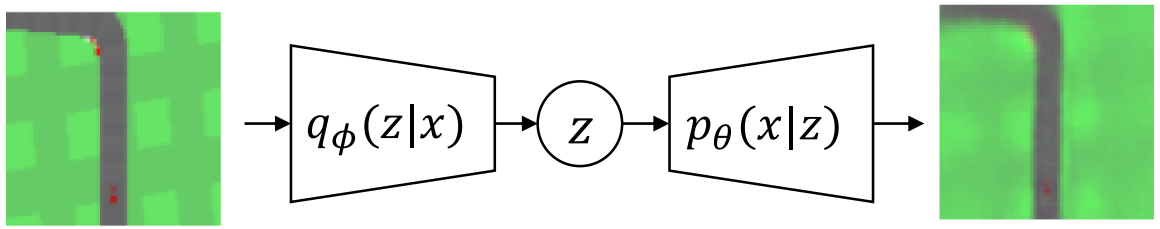
- Control from high-dimensional input (e.g., pixels)
- Compact latent representation makes planning and policy optimization much easier
  - Simple control schemes are also applicable; e.g.,
    - Linear programming [Water+, NeurIPS2015]
    - Linear controller [Ha+, NeurIPS2018]



758

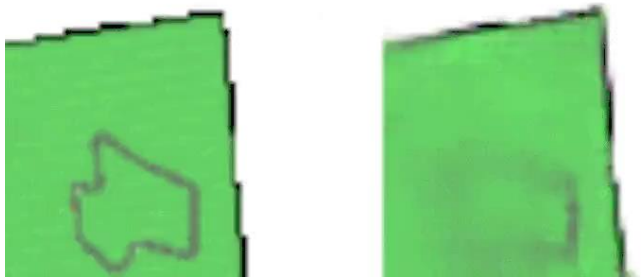
World Model [Ha+,NeurIPS2018]

- Model-based reinforcement learning utilizing VAE
- Procedure
  1. Train a VAE



  2. Train a latent dynamics model
     $p(z_{t+1}|z_t, a_t)$

  3. Train a policy utilizing *imagined* trajectories (learning inside of a dream)

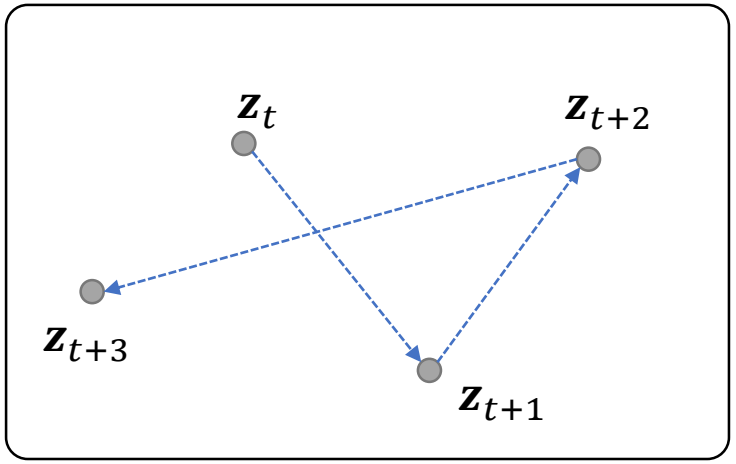


759

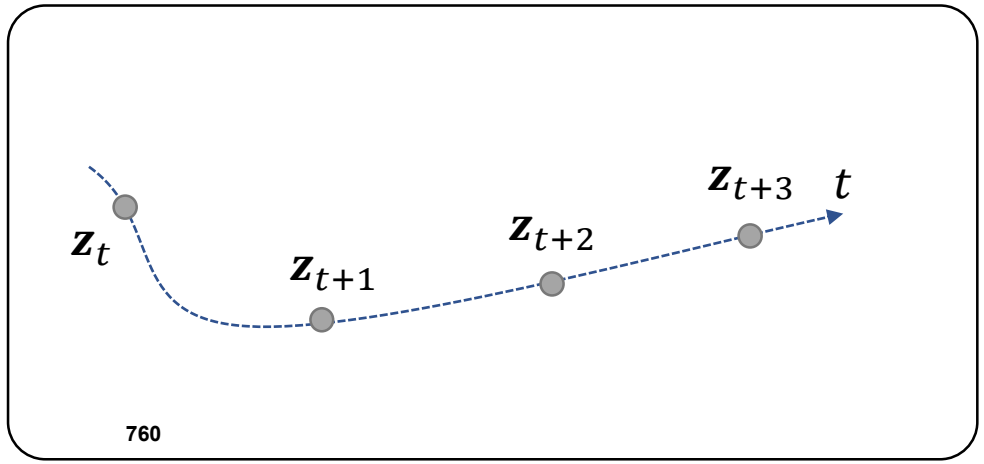
Is (vanilla-)VAE enough for control?

- World Model trains VAE and dynamics independently
- VAE does not consider time correlation
  - Consecutive latent might be distributed unsmoothly
  - How to incorporate time correlation for more smooth space?

Latent space

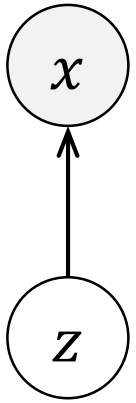


Latent space



Time-series VAE

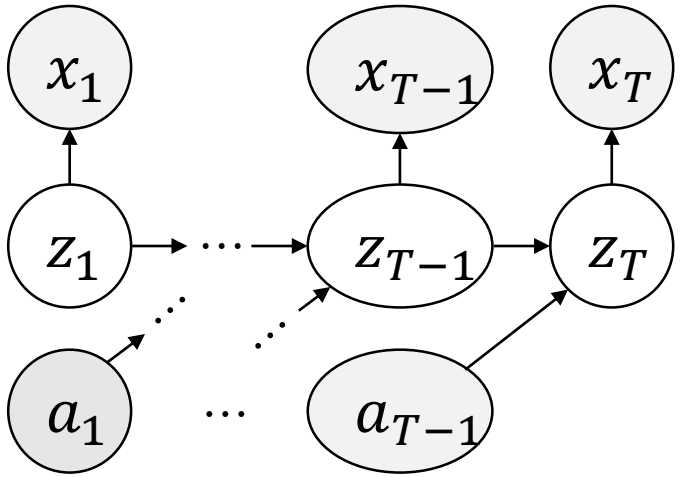
PGM of vanilla-VAE



$$J_{\text{ELBO}} := \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{KL}[q(z|x)||p(z)]$$

PGM of time-series VAE

a.k.a. POMDP
(Partially observable Markov Decision Process)



$$J_{\text{ELBO}} := \sum_t \mathbb{E}_{q(z_t|x_{\leq t}, a_{<t})}[\log p(x_t|z_t)] - \sum_t \mathbb{E}_{q(z_{t-1}|\cdot)}\left[D_{KL}[q(z_t|x_{\leq t}, a_{<t})||p(z_t|z_{t-1}, a_{t-1})]\right]$$

761

Time-series VAE

- $J_{\mathrm{ELBO}} := \sum_t \mathbb{E}_{q(z_t|x_{\leq t}, a_{<t})}[\log p(x_t|z_t)] - \sum_t \mathbb{E}_{q(z_{t-1}|\cdot)}\left[D_{KL}[q(z_t|x_{\leq t}, a_{<t})||p(z_t|z_{t-1}, a_{t-1})]\right]$
  - $q(z_t|x_{\leq t}, a_{<t})$: encoder
  - $p(x_t|z_t)$: decoder
  - $p(z_t|z_{t-1}, a_{t-1})$: dynamics model

  $x_{\leq t} := \{x_1, x_2, \cdots, x_t\}$
  $a_{<t} := \{a_1, a_2, \cdots, a_{t-1}\}$

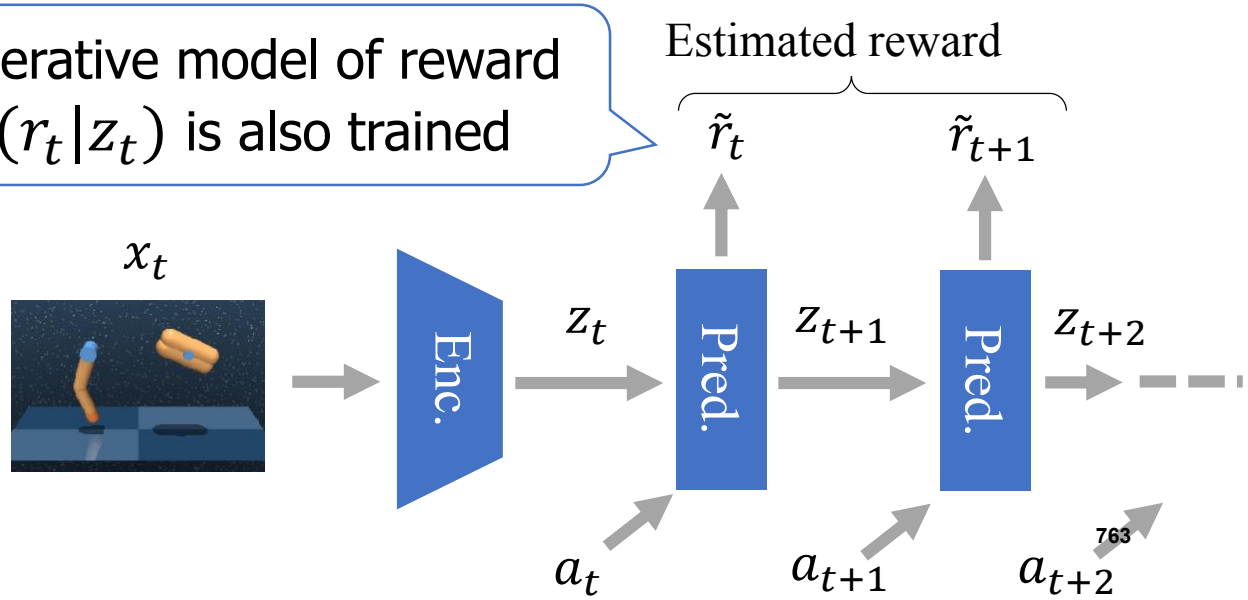
  ⇒ We can jointly train VAE and dynamics
  ⇒ Smooth latent space is constructed so that consecutive latent is easily predictable

762

Deep Planning Network (PlaNet) [Hafner+, ICML2018]

- Planning (trajectory optimization) in latent space utilizing a learned <u>time-series VAE</u>
- Optimize $a_{\geq t}$ to maximize the predicted return
  - Opt. method: Cross-entropy Method (Monte Carlo based)

Generative model of reward $p(r_t|z_t)$ is also trained



Dreamer [Hafner+, ICLR2019]

- Policy optimization in latent space utilizing learned <u>time-series VAE</u>
- Train parametrized policy π_θ (and value function) by backpropagating the estimated return

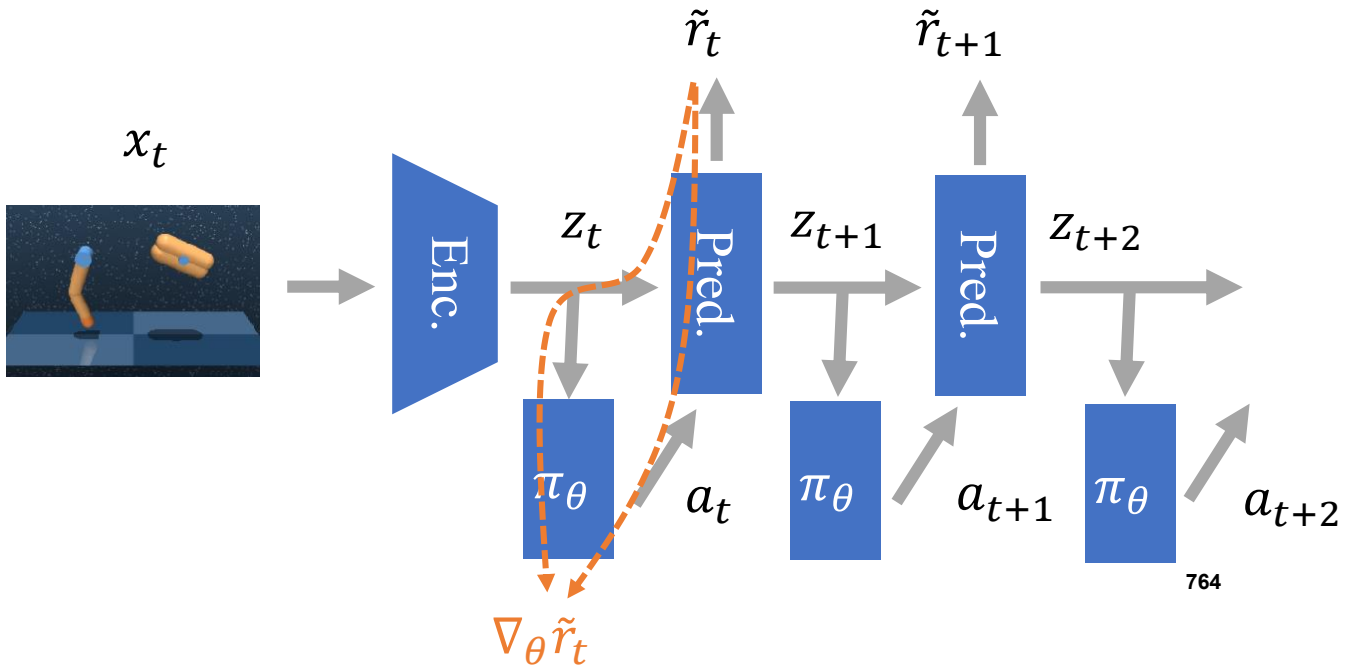
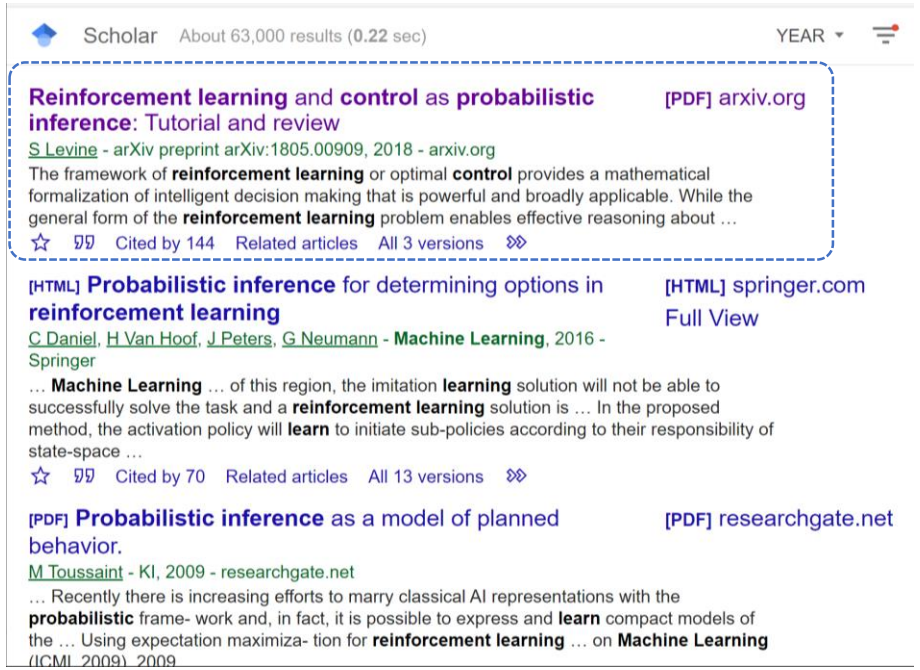


Table of Contents

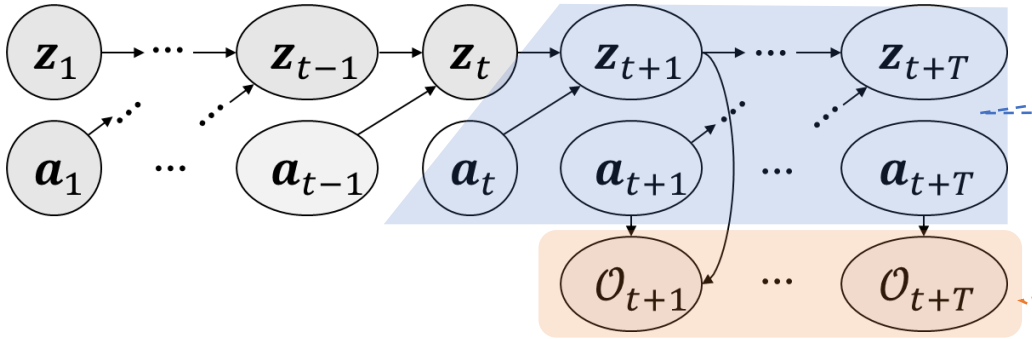
- A basic deep generative model: variational autoencoder (VAE)
  - Theory
  - Applications
    - World model, PlaNet, Dreamer, etc…
- Control as probabilistic inference
  - Theory
  - Applications
    - SAC, SLAC, VI-MPC, PaETS, PlaNet-Bayes



765

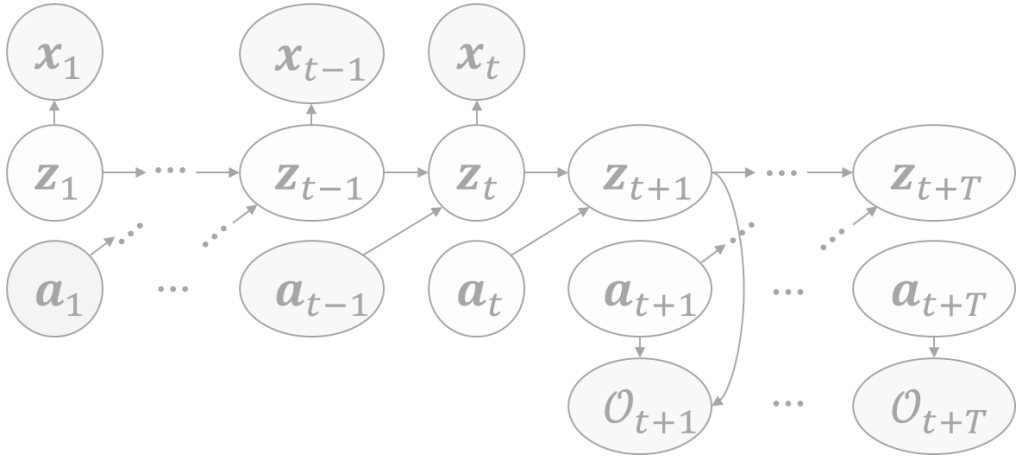
Problem Formulation

Fully Observable Markov Decision Process (MDP)



Trajectory: $\tau = (z_{>t}, a_{\geq t})$

Partially Observable MDP (discussed later)



Optimality

- Binary variable
  - $\mathcal{O}_t = 1$ if $r(z_t, a_t)$ is optimal
  - $r(z_t, a_t)$: reward function
- $p(\mathcal{O}_t = 1 | z_t, a_t) \coloneqq f(r(z_t, a_t))$
  - $f(\cdot)$: increasing function (discussed later)

What we want:

$p(\tau | \mathcal{O}_{>t} = \mathbf{1}, z_t)$

will be omitted for readability

How to infer $p(\tau|\mathcal{O}_{>t} = \mathbf{1})$

- Analytical inference $p(\tau|\mathcal{O}_{>t} = \mathbf{1})$ is intractable
  $\Rightarrow$ Variational inference $D_{KL}[q(\tau)||p(\tau|\mathcal{O}_{>t} = \mathbf{1})]$
  by ELBO maximization

- How to define a variational distribution $q(\tau)$
  - $q(\tau) \coloneqq q(a_{\geq t}) \prod_t p(z_{t+1}|z_t, a_t)$
    - To derive <u>trajectory optimization</u>
  - $q(\tau) \coloneqq \prod_t p(z_{t+1}|z_t, a_t)\pi(a_t|s_t)$
    - To derive <u>policy optimization</u>
  - Note: $p(z_{t+1}|z_t, a_t)$ governs the state transition

767

Trajectory optimization as inference

- Derivation of ELBO
  - $\log p(\mathcal{O}_{>t} = \mathbf{1}) = \log \int p(\tau, \mathcal{O}_{>t} = \mathbf{1})\, d\tau$
  - $= \log \int p(\tau, \mathcal{O}_{>t} = \mathbf{1}) \cdot \frac{q(\tau)}{q(\tau)}\, d\tau$
  - $= \log \mathbb{E}_{q(\tau)} \left[\frac{p(\tau, \mathcal{O}_{>t} = \mathbf{1})}{q(\tau)} \right]$
  - $= \log \mathbb{E}_{q(\tau)} \left[\frac{\prod p(z_{t+1} | z_t, a_t) \cdot \prod p(\mathcal{O}_t = 1 | z_t, a_t)}{q(a_{\geq t}) \prod p(z_{t+1} | z_t, a_t)} \right]$
  - $\geq \mathbb{E}_{q(\tau)} [\log \prod p(\mathcal{O}_t = 1 | z_t, a_t) - \log q(a_{\geq t})] := J_{\text{ELBO}}$

$q(\tau)$: variational distribution

$p(a_{\geq t})$: uninformative prior

Jensen's inequality

$\sum r(z_t, a_t)$
if $p(\mathcal{O}_t | z_t, a_t) := \exp(r(z_t, a_t))$

Entropy of $q(a_{\geq t})$

768

Max. J_{ELBO}
=
Max. Return + Entropy

Variational Inference MPC
[Okada+,CoRL2019]

- Max. $J_{ELBO} \coloneqq \mathbb{E}_{q(\tau)}[\log p(\mathcal{O}_{>t} = \mathbf{1}|\tau) - \log q(a_{\geq t})]$

$\coloneqq \prod p(\mathcal{O}_t|z_t, a_t)$

Mirror Descent

- VI-MPC (variational inference model predictive control)

$$q(a) \leftarrow \frac{q(a) \cdot \mathbb{E}_\tau[p(\mathcal{O}_{>t} = \mathbf{1}|\tau)] \cdot q(a)^{-\kappa}}{\mathbb{E}_{q(a)}[\mathbb{E}_\tau[p(\mathcal{O}_{>t} = \mathbf{1}|\tau)] \cdot q(a)^{-\kappa}]}$$

  - This generalizes various MPC methods

$p(\mathcal{O}_{>t}|\tau)$

	MPPI [18]	CEM [16]	Prop-CEM [20]	CMA-ES [19]
$f(r(\tau))$	$\propto e^{r(\tau)}$	$\mathbb{1}[r(\tau) > r_{thd}]$	$\frac{r(\tau) - r_{min}}{r_{max} - r_{min}}$	$\propto \log g(r(\tau)) \cdot \mathbb{1}[r(\tau) > r_{thd}]$

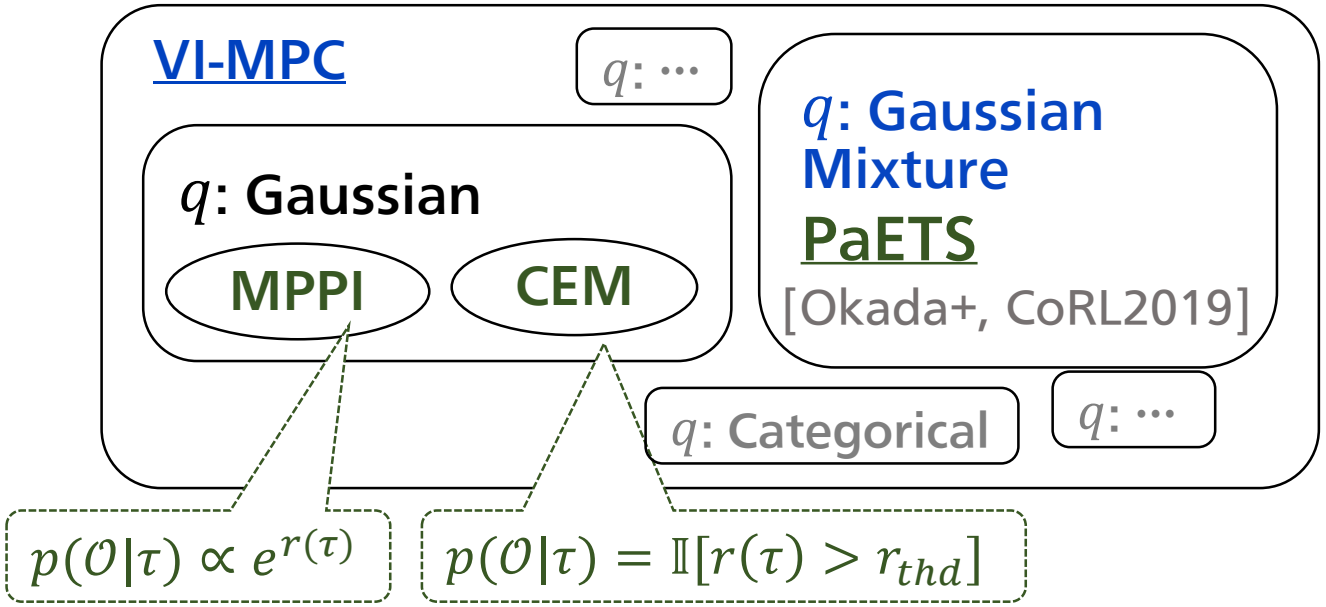
Model Predictive Path Integral

Cross Entropy Method

VI-MPC & PaETS [Okada+, CoRL2019]

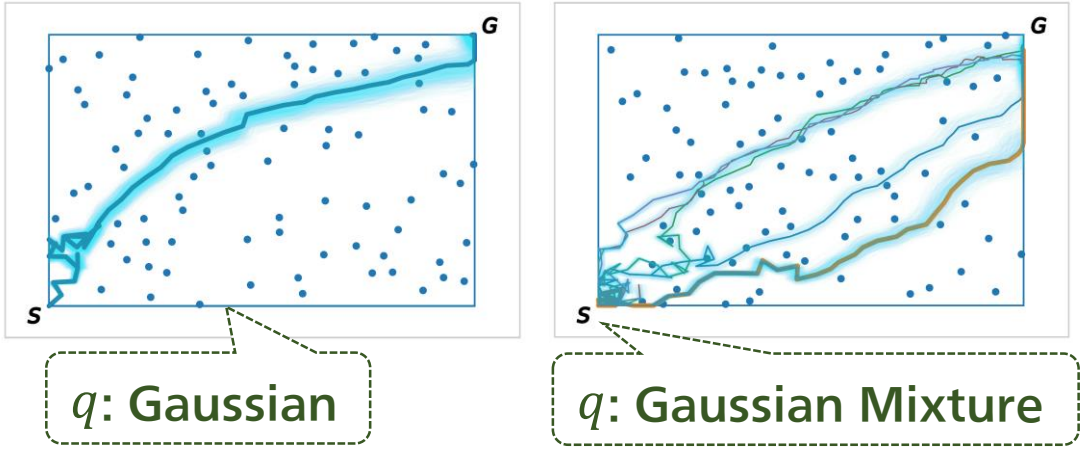
(Probabilistic Action Ensemble with Traj. Sampling)

- Different definitions of $p(\mathcal{O}|\tau)$ & $q(a)$ derive various MPC methods



770

e.g., Toy navigation task



Encourage active exploration in reinforcement learning

Policy optimization as inference

- Derivation of ELBO
  - $\log p(\mathcal{O}_{>t} = \mathbf{1}) \geq$
  - $\mathbb{E}_{\tau \sim q(\tau)}[\sum r(z_t, a_t) - \sum \log \pi(a_t|z_t)] \coloneqq J_{\text{ELBO}}$

$q(\tau) \coloneqq \prod p(z_{t+1}|z_t, a_t)\pi(a_t|z_t)$

$p(\mathcal{O}_t|z_t, a_t) \coloneqq \exp(r(z_t, a_t))$

Entropy of π

- Solve ELBO by dynamic programming
  - $J_{\text{ELBO}} = \mathbb{E}_{a_t \sim \pi(a_t|z_t)}[r(z_t, a_t) - \log \pi(a_t|z_t) + \mathbb{E}_{z_{t+1} \sim p(z_{t+1}|z_t, a_t)}[V(z_{t+1})]]$
  - $= \mathbb{E}_{a_t \sim \pi(a_t|z_t)}[Q(z_t, a_t) - \log \pi(a_t, z_t)]$
  - where
    - $V(z_{t+1}) \coloneqq \mathbb{E}_{\pi(a_{t+1}|z_{t+1})}[r(z_{t+1}, a_{t+1}) - \log \pi(a_{t+1}, z_{t+1}) + \mathbb{E}_{p(z_{t+2}|\cdot)}[V(z_{t+2})]]$
    - $Q(z_t, a_t) \coloneqq r(z_{t+1}, a_{t+1}) + \mathbb{E}_{p(z_{t+1}|z_t, a_t)}[V(z_{t+1})]$

Policy optimization as inference

- Solve ELBO by dynamic program
  - $J_{\text{ELBO}} = \mathbb{E}_{a_t \sim \pi(a_t|z_t)}[Q(z_t, a_t) - \log \pi(a_t, z_t)]$
  - $= \mathbb{E}_{a_t \sim \pi(a_t|z_t)}\left[\log e^{Q(z_t,a_t)} - \log \pi(a_t, z_t)\right]$
  - $= \mathbb{E}_{a_t \sim \pi(a_t|z_t)}\left[\log e^{Q(z_t,a_t)} - \log \pi(a_t, z_t) + \log Z - \log Z\right]$
    - $Z(= \int e^{Q} da)$: normalizer
  - $\propto -D_{KL}\left[\pi(a_t|z_t) || \frac{e^{Q(z_t,a_t)}}{Z}\right]$
- Optimal policy
  - $\pi^*(a_t|z_t) = \frac{e^{Q(z_t,a_t)}}{Z}$
    - Boltzmann distribution

772

Soft Actor Critic (SAC) [Haarnoja+, ICML2018]

- Soft bellman equation
  - $V^*(z_t) = \log \int \exp Q^*(z_t, a_t) da_t \; (= \log Z)$
  - $Q^*(z_t, a_t) = r(z_t, a_t) + \mathbb{E}_{p(z_{t+1}|z_t, a_t)}[V^*(z_{t+1})]$
  - $\pi^*(a_t|z_t) = \exp[Q^*(z_t, a_t) - V^*(z_t)] = \operatorname{argmin} D_{KL}[\pi || \exp[Q^* - V^*]]$
- Concept of SAC
  - Define deep parameterized models V_ψ, Q_θ, π_ϕ
  - Train V_ψ, Q_θ, π_ϕ to be V^*, Q^*, π^* by optimizing <u>bellman errors</u>
    - $J_V = \left(V_\psi - \log \int \exp Q_\theta \, da\right)^2 \geq \left(V_\psi - \mathbb{E}[Q_\theta - \log \pi_\phi]\right)^2$
    - $J_Q = \left(Q_\theta - (r + \mathbb{E}[V_\psi])\right)^2$
    - $J_\pi = -D_{KL}[\pi_\phi || \exp[Q_\theta - V_\psi]]$

Lower bound

773

Soft Actor Critic (SAC) [Haarnoja+, ICML2018]

Screenshot @ Oct. 5th, 2020

soft actor critic

Scholar About 88,000 results (0.05 sec) YEAR

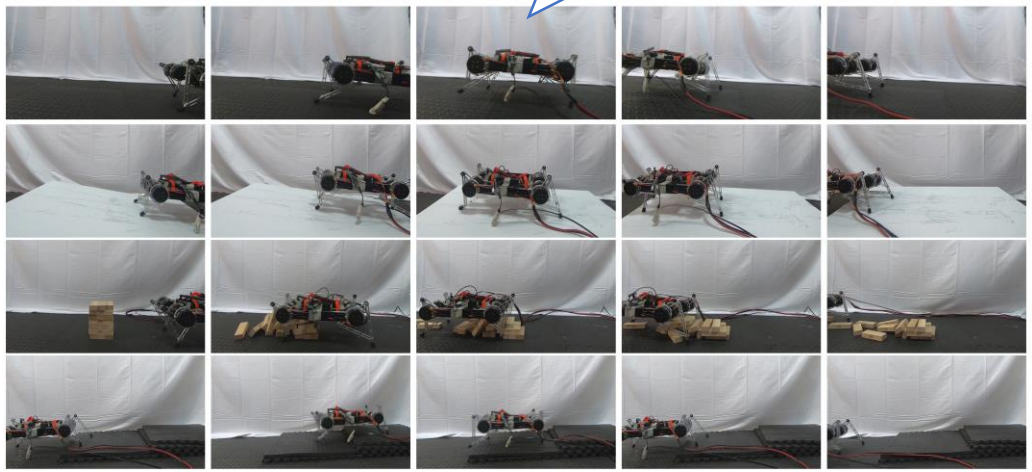
Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic **actor** [PDF] arxiv.org
T Haarnoja, A Zhou, P Abbeel, S Levine - arXiv preprint arXiv:1801.01290, 2018 - arxiv.org
Model-free deep reinforcement learning (RL) algorithms have been demonstrated on a range of challenging decision making and control tasks. However, these methods typically suffer from two major challenges: very high sample complexity and brittle convergence …
Cited by 867 Related articles All 6 versions

Soft actor-critic algorithms and applications [PDF] arxiv.org
T Haarnoja, A Zhou, K Hartikainen, G Tucker… - arXiv preprint arXiv …, 2018 - arxiv.org
Model-free deep reinforcement learning (RL) algorithms have been successfully applied to a range of challenging sequential decision making and control tasks. However, these methods typically suffer from two major challenges: high sample complexity and brittleness to …
Cited by 227 Related articles All 2 versions

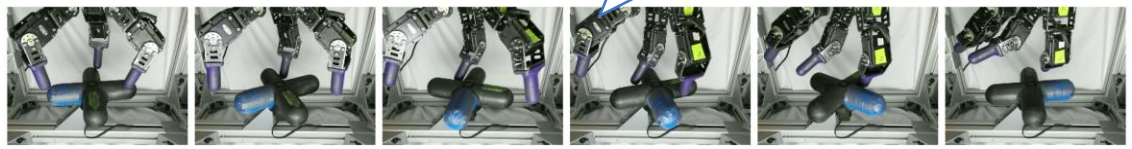
Improving exploration in **soft-actor-critic** with normalizing flows policies [PDF] arxiv.org
PN Ward, A Smofsky, AJ Bose - arXiv preprint arXiv:1906.02771, 2019 - arxiv.org
Deep Reinforcement Learning (DRL) algorithms for continuous action spaces are known to be brittle toward hyperparameters as well as\cut {being} sample inefficient. **Soft Actor Critic** (SAC) proposes an off-policy deep **actor critic** algorithm within the maximum entropy RL …
Cited by 9 Related articles All 2 versions

Soft actor-critic for discrete action settings [PDF] arxiv.org

2 hours



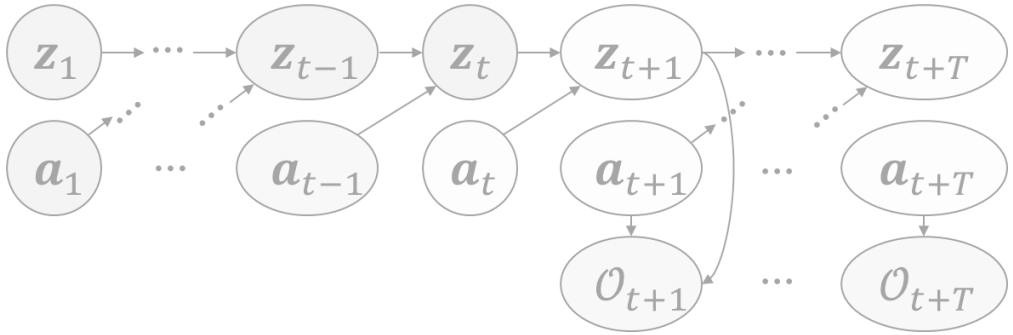
3 hours



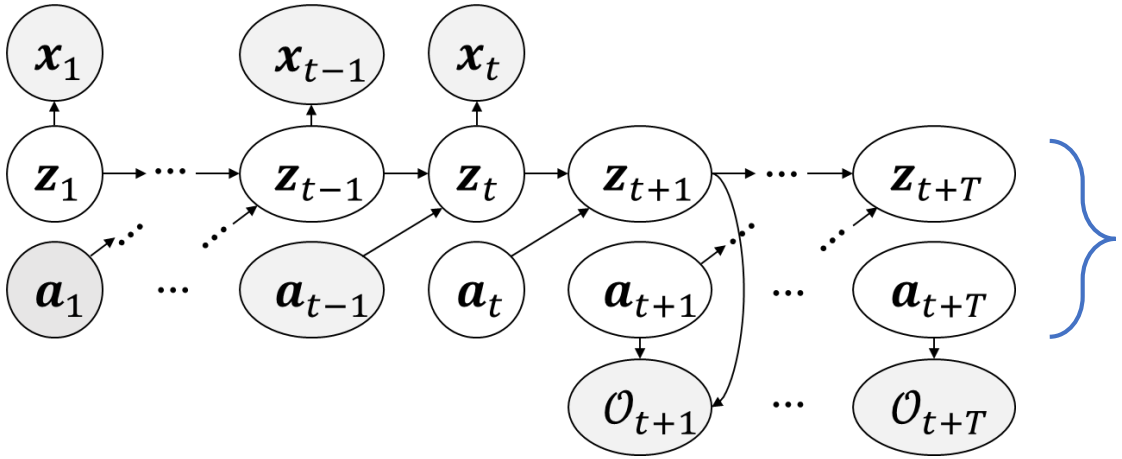
774

Problem Formulation

Fully Observable Markov Decision Process (MDP)



Partially Observable MDP



Trajectory: $\tau = (z_{\leq t+T}, a_{\geq t})$

What we want:

$$p(\tau | \mathcal{O}_{>t} = \mathbf{1}, x_{\leq t}, a_{<t})$$

775

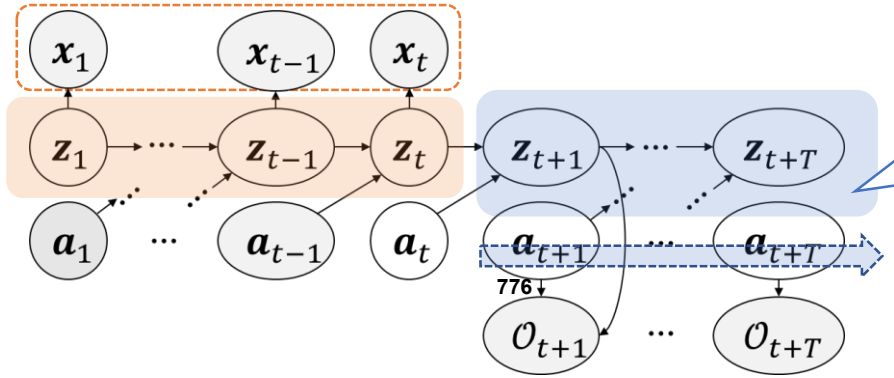
Trajectory optimization as inference

- Derivation of ELBO
  - $\log p(\mathcal{O}_{>t} = \mathbf{1}, x_{\leq t} | a_{<t})$
  - $\geq \mathbb{E}_{q(\tau)}[\log p(\mathcal{O}_{>t}|\tau) - \log q(a_{\geq t}) + \log p(x_{\leq t}|z_{\leq t})]$

Observation likelihood

- Optimizing ELBO by latent VI-MPC [Okada+,IROS2020]
  - $q(a_{>t}) \leftarrow \dfrac{q(a_{>t}) \cdot \mathbb{E}_{z_{\leq t+T} \sim \mathbb{P}}[p(\mathcal{O}_{>t}|\tau)] \cdot q^{-\kappa}}{\mathbb{E}_{q(a_{>t})}\left[\mathbb{E}_{z_{\leq t+T} \sim \mathbb{P}}[p(\mathcal{O}_{>t}|\tau)] \cdot q^{-\kappa}\right]}$
    - $\mathbb{P} := \left(\prod_{t'=1}^{t} q(z_{t'}|x_{\leq t'}, a_{<t'})\right)\left(\prod_{t'=t}^{t+T} p(z_{t'+1}|z_{t'}, a_{t'})\right)$

Inference by encoder $q(z_{t'}|\cdot)$



Prediction by latent dynamics $p(z_{t'+1}|\cdot)$

PlaNet of the Bayesians (PlaNet-Bayes) [Okada+,IROS2020]

- Extension of PlaNet based on (Bayesian) inference

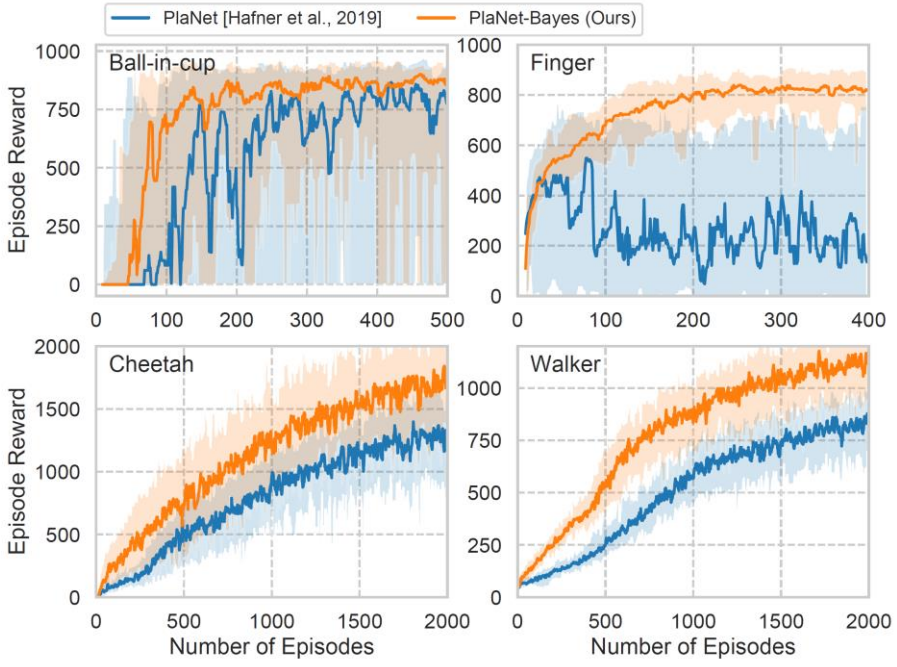
Heuristically employs CEM

Utilize Bayesian inference?

	1. Latent Model θ	2. MPC-Policy
PlaNet [Hafner et al., 2019]	✗: $\mathrm{argmax}_\theta\, p(\mathcal{D}\|\theta)$	✗: $\mathrm{argmax}_{\boldsymbol{a}_{\geq t}}\, r(\boldsymbol{z}_{\geq t}, \boldsymbol{a}_{\geq t})$
PlaNet-Bayes (Ours)	✓: Infer $p(\theta\|\mathcal{D})$	✓: Infer $p(\boldsymbol{z}_{\geq t}, \boldsymbol{a}_{\geq t}\|\mathcal{O}_{\geq t})$

Model posterior inference is also conducted

Employs latent VI-MPC & PaETS



777

Policy optimization as inference
SLAC (stochastic latent actor critic) [Lee+, arXiv2019]

- Derivation of ELBO
  - $\log p(\mathcal{O}_{>t} = \mathbf{1}, x_{\leq t} | a_{<t})$
  - $\geq \sum_{t' \leq t} \mathbb{E}_{q\left(z_{t'} | x_{\leq t'}, a_{<t'}\right)} [\log p(x_{t'} | z_{t'})] - \sum_{t' > t} \mathbb{E}_{q(z_{t'-1} | \cdot)} \left[D_{KL} [q(z_{t'} | x_{\leq t'}, a_{<t'}) || p(z_{t'} | z_{t'-1}, a_{t'-1})] \right] + \sum_{t' > t} \mathbb{E}_{\pi(a_{t'} | x_{\leq t'}, a_{<t'})} [r(z_{t'}, a_{t'}) - \log \pi(a_{t'} | \cdot)]$
  - $:= \mathcal{J}_{\mathrm{VAE}} + \mathcal{J}_{\mathrm{SAC}}$
- Concept of SLAC
  - Define parameterized models V_ψ, Q_θ, π_ϕ, q_ϕ, p
  - Jointly optimize $\mathcal{J}_{\mathrm{VAE}}$ and $\mathcal{J}_{\mathrm{SAC}}$ by similar training procedures of time-series VAE and SAC

Takeaways

- Various popular algorithms are based on variational inference (or ELBO maximization)

- Derive your novel algorithm by
  - Formulate your graphical model (or generative model)
  - Define variational distribution q
  - Derive J_{ELBO}
- And implement your algorithm by
  - TensorFlow, Keras, Pytorch,
  - Pixyz, SERKET