ENSTA ParisTech université PARIS-SACLAY

INRIA INVENTEURS DU MONDE NUMÉRIQUE

DREAM

ARMINES

Autonomous Systems and Robotics

Computer Science and System Engineering Laboratory

State Representation Learning for control: an overview

T. Lesort, N. Díaz-Rodríguez, J-F. Goudou, D. Filliat

Natalia Díaz Rodríguez, PhD

Paris, 4th April 2018

INRIA Flowers Deep Reinforcement Learning workshop

ABOUT ME

Universidad de Granada

Åbo Akademi

CERN

UNIVERSITY OF CALIFORNIA SANTA CRUZ

TUCS

Nokia Foundation

Women Techmakers

eit Digital

ENSTA ParisTech université PARIS-SACLAY

Inria INVENTEURS DU MONDE NUMÉRIQUE

Autonomous Systems and Robotics
Computer Science and System Engineering Laboratory

Google

PHILIPS

SILO.AI

NVIDIA

HEIDELBERG LAUREATE FORUM

STITCH FIX™

asteria

Kavli Summer Institute in Cognitive Neuroscience

SGX Analytics

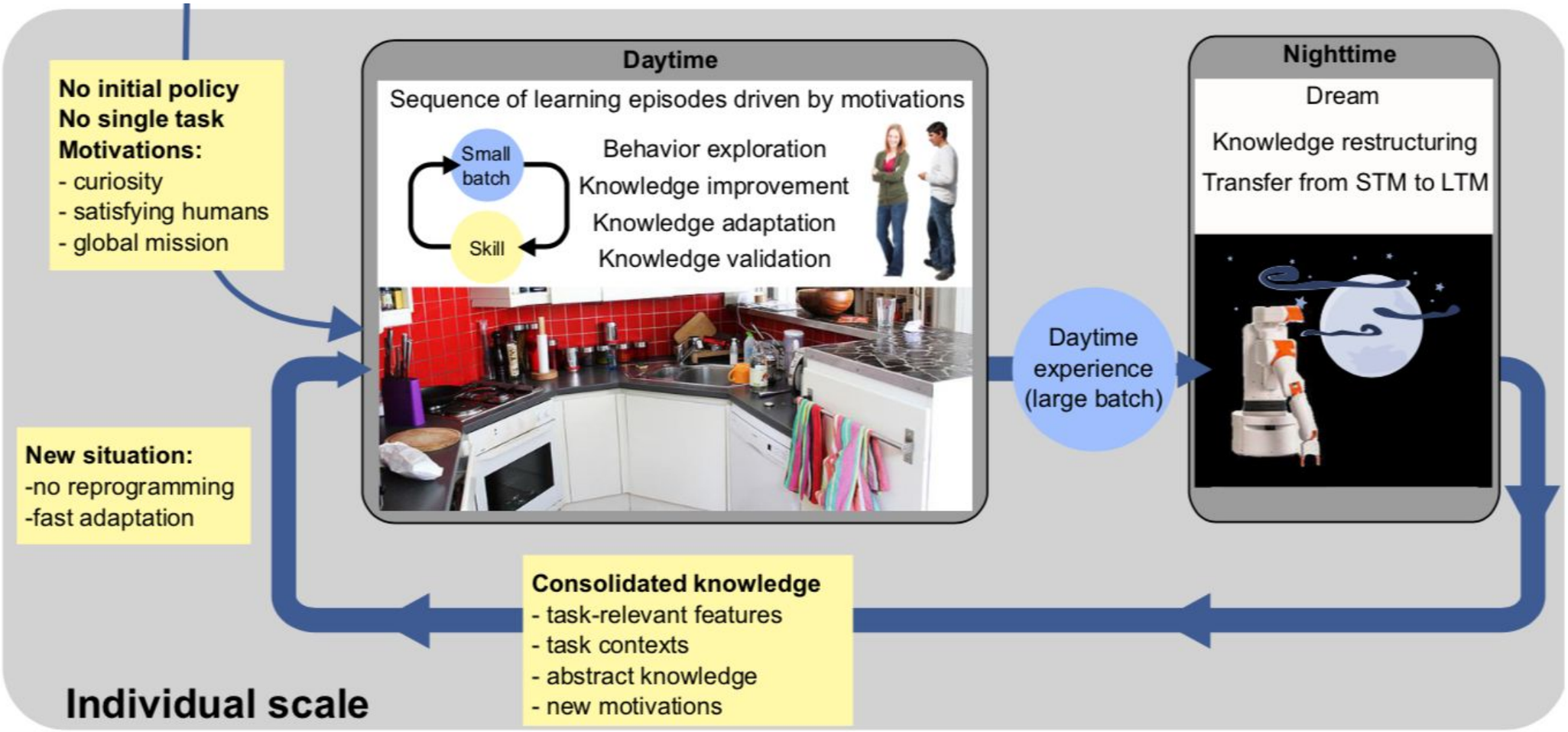
DREAM Project:

Deferred Restructuring of Experience in Autonomous Machines
A European Union Horizon 2020 project

www.robotsthatdream.eu

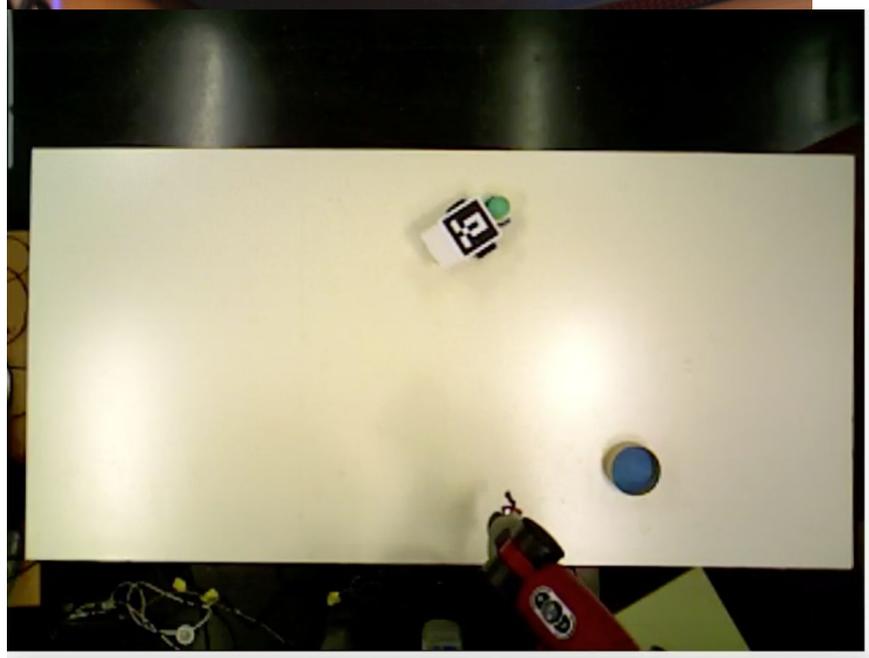
- Learning and adapting in open-ended domains
    - Knowledge compression, consolidation, and transfer
    - Artificial curiosity and representation redescription
- **Day**: getting data and learning tasks. **Night**: Finding better representations

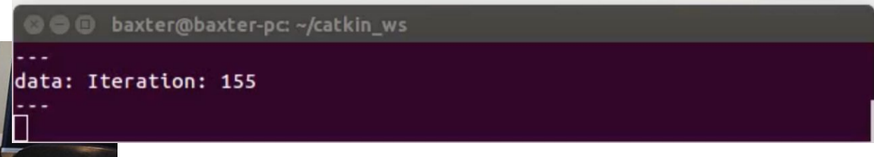
Rethink robotics' Baxter




ENCYCLOPÆDIA CORSICAE
ENCYCLOPÆDIA CORSICAE
ENCYCLOPÆDIA CORSICAE


SAMSUNG
DELL




baxter@baxter-pc: ~/catkin_ws
---
data: Iteration: 155


JACQUELINE
HEINERMAN

This presentation: fresh work

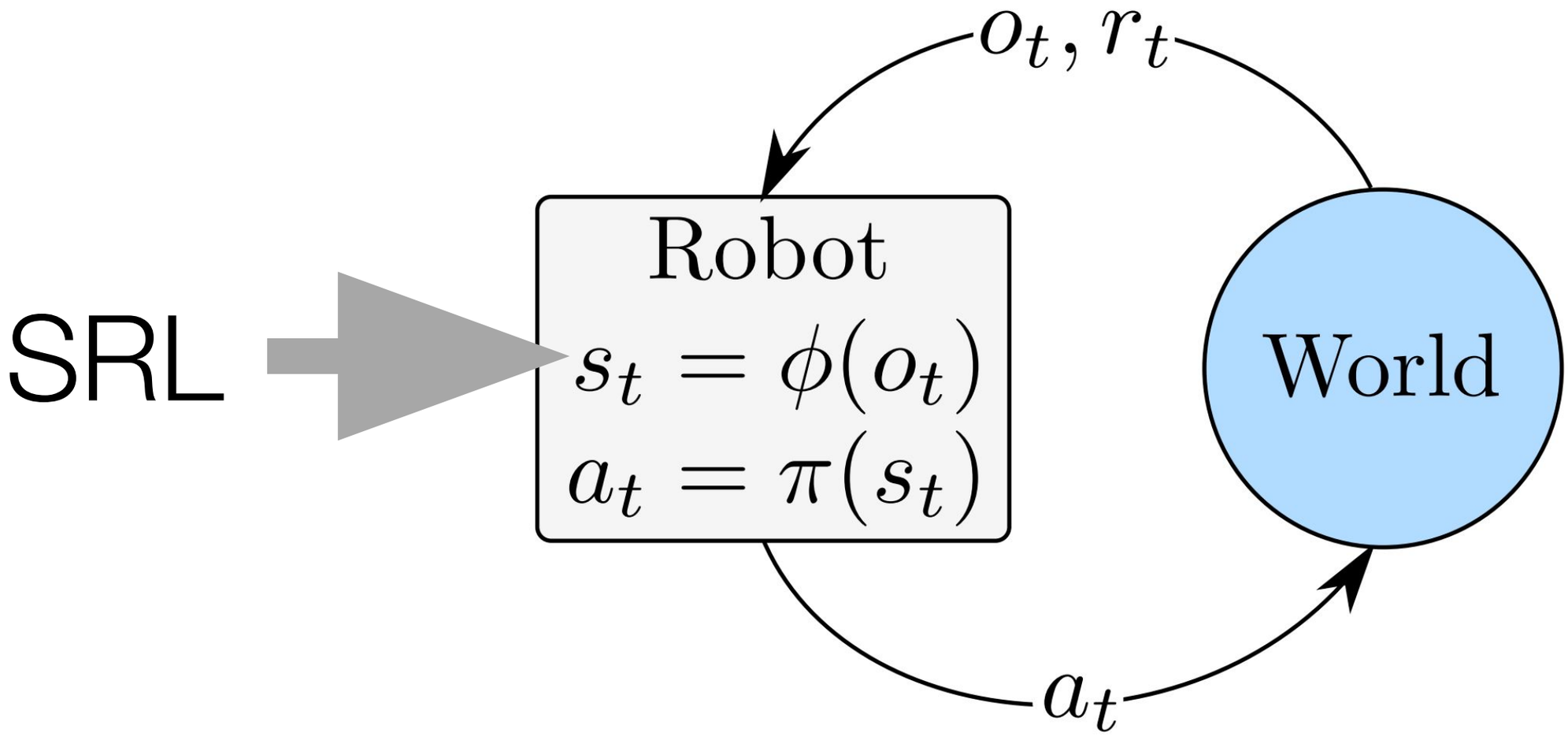
State Representation Learning for Control: An Overview

Timothée Lesort[1, 2], Natalia Díaz-Rodríguez[1], Jean-François Goudou[2], and David Filliat[1]

[1]U2IS, ENSTA ParisTech, Inria FLOWERS team, Université Paris Saclay, Palaiseau, France.,
{timothee.lesort, natalia.diaz, david.filliat}@ensta-paristech.fr
[2]Thales, Theresis Laboratory, Palaiseau, France.,
{jean-francois.goudou@thalesgroup.com

Contents

State Representation Learning (SRL) in RL context



Why learning states ?

- Often, robot controllers require simple, ‘high-level’ inputs (the ‘state’ of the world)
  - E.g., *grasping*: object position; *driving*: road direction

- High dimensional visual input perception requires filtering to get this relevant information (often hand-crafted)
  - @DREAM: Learning state = Discarding irrelevant knowledge (e.g., during ‘off’ or ‘sleep time’).

- Controllers are easier to train in such lower dimension
  - This ‘high-level’ information can be less dependent on the environment and help transfer learning

Learning states vs learning representations

- Learning a state is a particular case of representation learning
- Learning states entails a control/robotics context where actions are possible
- Often we look for interpretable info./ info. with a *physical* meaning
- Learning a state can be an objective by itself, but is often present in more general approaches
  - E.g., as an auxiliary task in RL

Learning states vs learning representations

- Focus in this survey:
  - Learning objectives that are relevant for SRL
  - Put in perspective common factors between very different use cases

First, what is a good state?

The Markovian view: [Böhmer'15]

A good state representation is a representation that is:

1. Markovian, i.e. it summarizes all the necessary information to be able to choose an action within the policy, by looking only at the current state.
2. Able to:
   a. Represent the true value of the current state well enough for policy improvement.
   b. Generalize the learned value-function to unseen states with similar futures.
3. Low dimensional for efficient estimation.

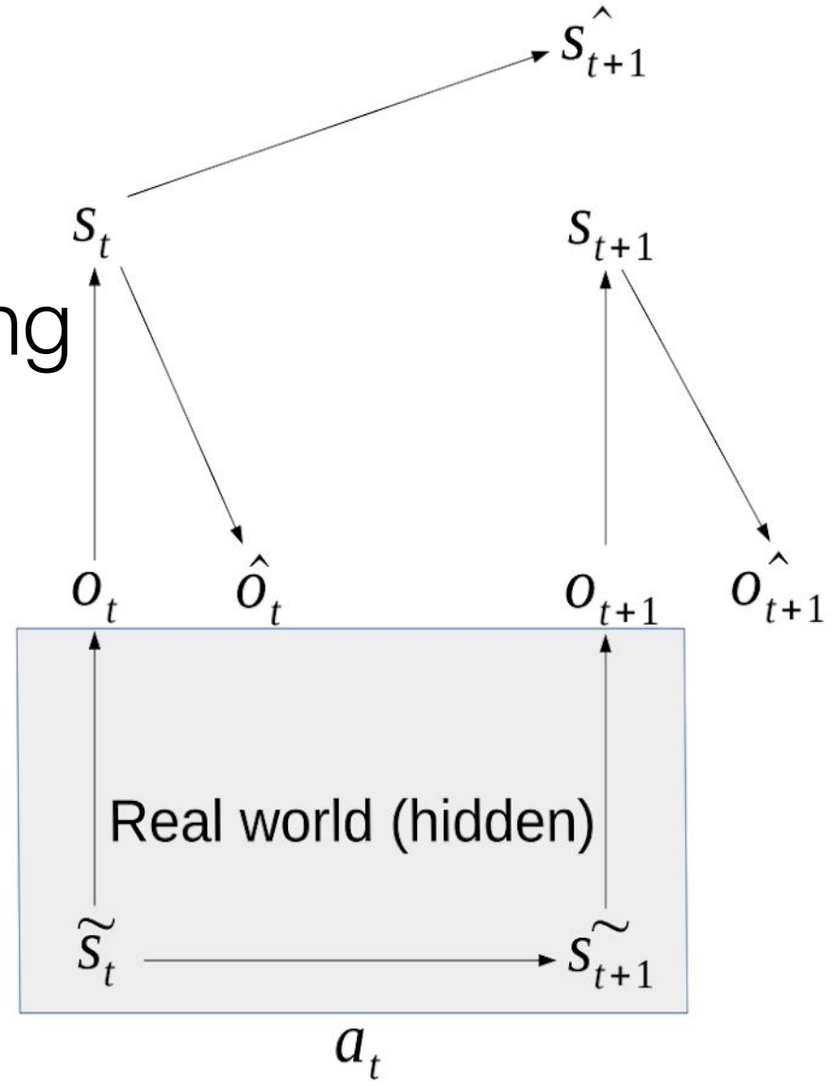
Further, a good state representation should:

- Sufficient
- As efficient as possible
  (i.e., easy to work with, e.g., factorizing the data-generating factors)
- Minimal (from all possible representations, take the most efficient one).
- In summary: exclude irrelevant information to encourage a lower dimensionality

[Achille and Soatto, 2017]

Learning states: challenges

- Learning without direct supervision
  - True *state* never available
- Solution: Use ‘self-supervised’ learning
  - RL framework
  - Exploit observations, actions, rewards
- Various complementary approaches
  - Reconstruct observation
  - Learn forward/inverse model
  - ….



What can SRL be used for?
e.g., in computer vision for robotics control

Our goal is to learn a relevant state from images, actions and rewards. We train a neural network in unsupervised manner using prior knowledge about the physical world in form of robotic priors.

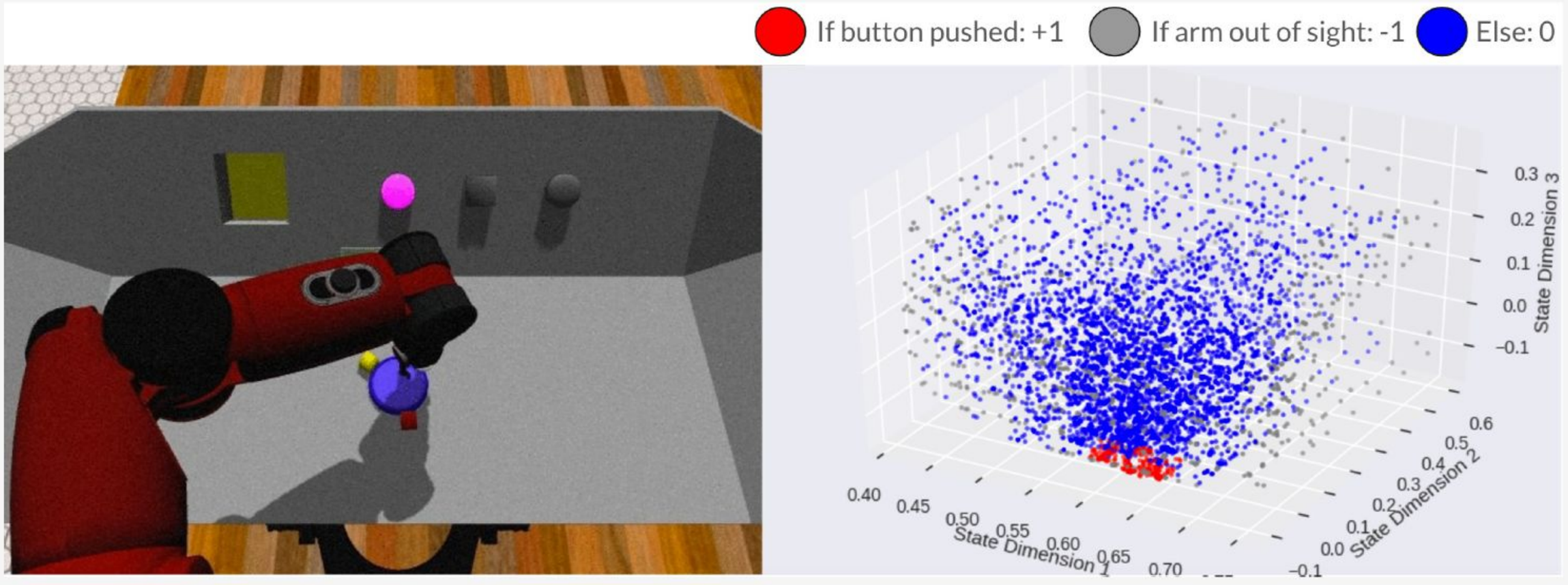
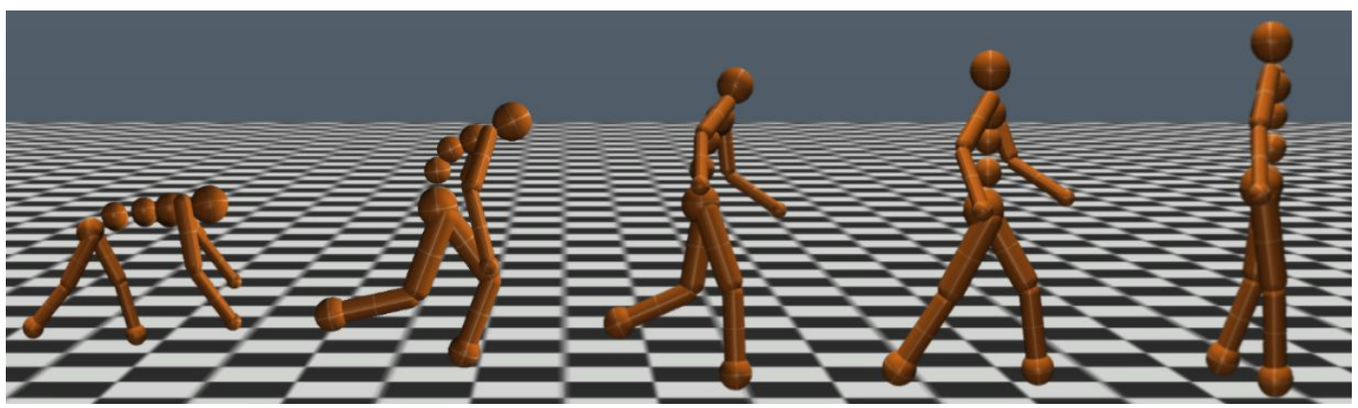


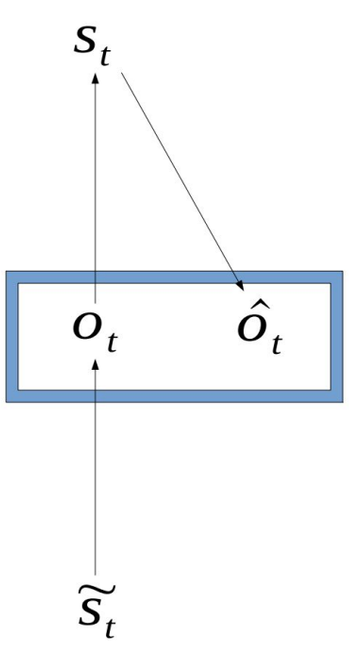
Figure 1: Left: Baxter's camera view for Static-Button-Distractors dataset 2. Right: Baxter's left hand ground truth position and its coded reward

Applications of SRL

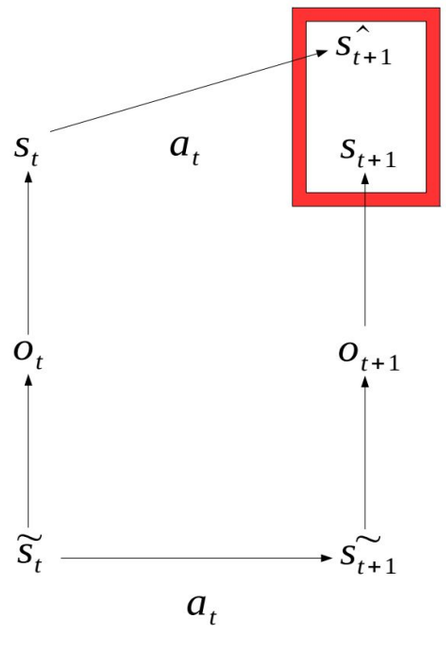
- RL and self-supervision
- Transfer learning
- Multimodal observation learning (e.g.,autonomous vehicles)
- Multi-objective optimization
- Evolutionary strategies



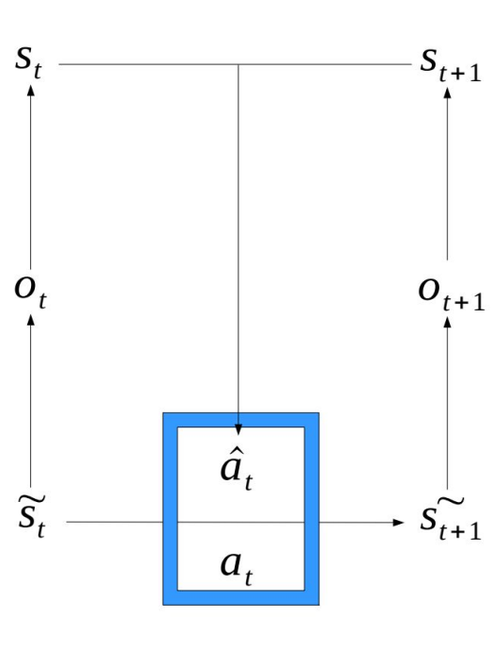
Overview of existing approaches



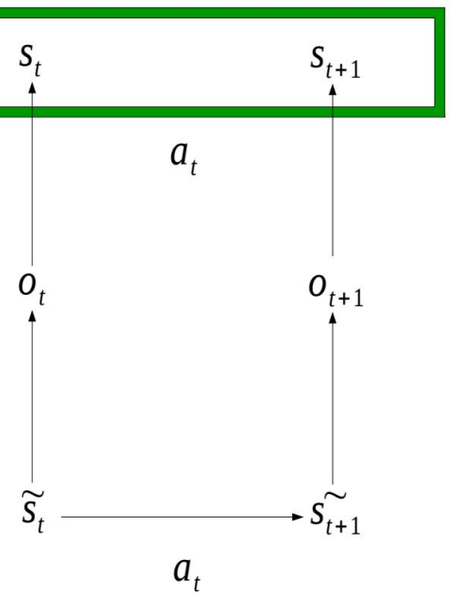
(a) Auto-Encoder



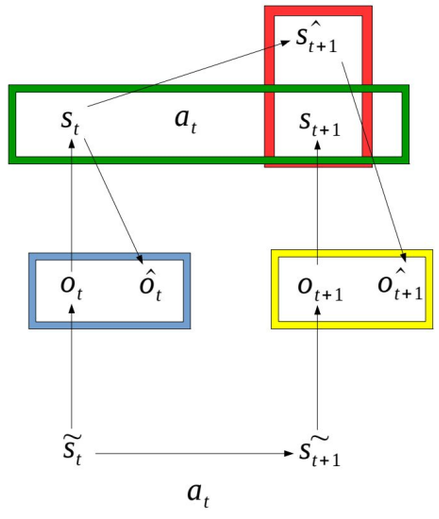
(b) Forward Model



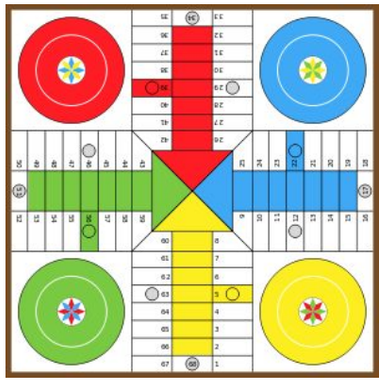
(c) Inverse Model



(d) Model with prior

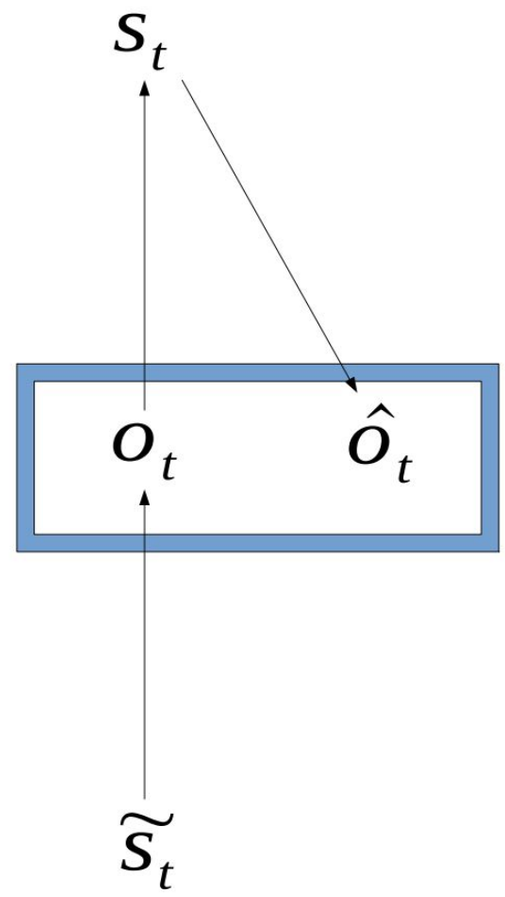


(e) Example of hybrid model



Models that reconstruct the observation

- Train a state that is sufficient for reconstructing the input observation
  - AE, DAE, VAE
  - GANs
- Downside: sensitive to irrelevant variations



(a) Auto-Encoder

$$s_t = \phi(o_t; \theta_\phi) \tag{1}$$

$$\hat{o}_t = \phi^{-1}(s_t; \theta_{\phi^{-1}}) \tag{2}$$

AE example: Deep Spatial Autoencoders

- Learns a representation that corresponds to objects coords. (image space):

$$\text{feature encoder } h_{\text{enc}}(I_t) = \mathbf{f}_t$$

- Torque control & 2nd-order dynamical system requires features and their velocities $\dot{\tilde{\mathbf{f}}}_t$

$$\text{full state } \mathbf{x}_t = [\tilde{\mathbf{x}}_t; \mathbf{f}_t; \dot{\tilde{\mathbf{f}}}_t]$$

Deep spatial autoencoders for visuomotor learning [Finn'15]

AE example: Deep Spatial Autoencoders

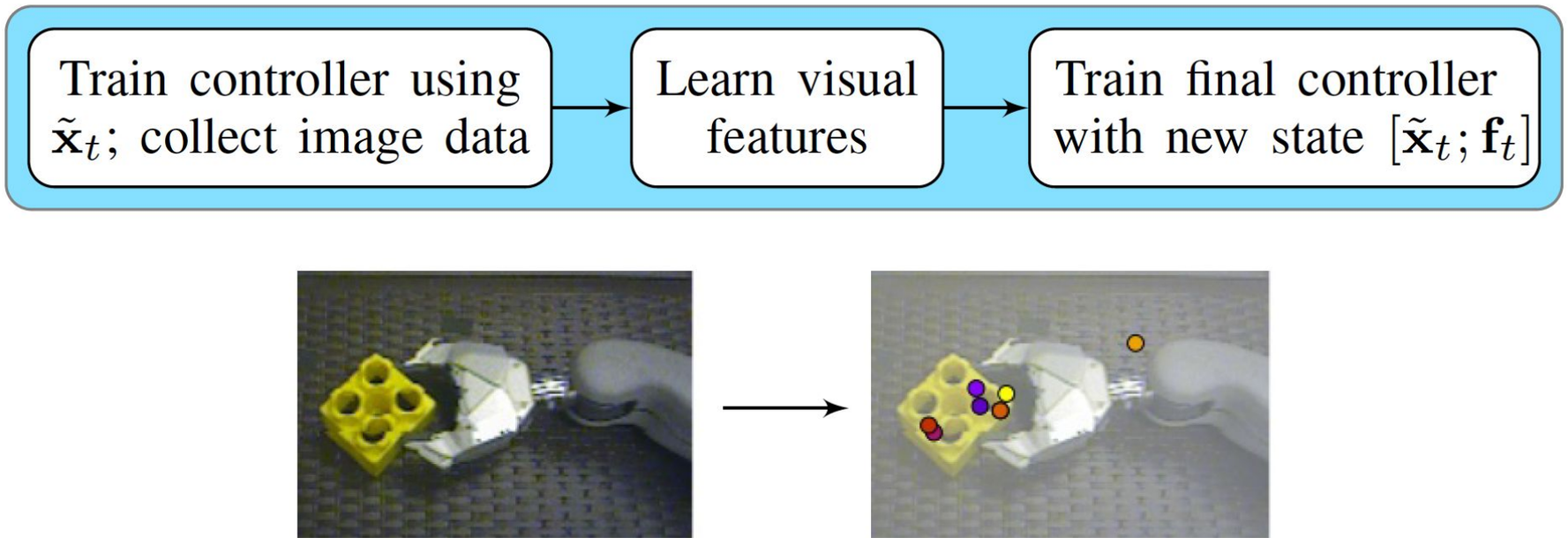
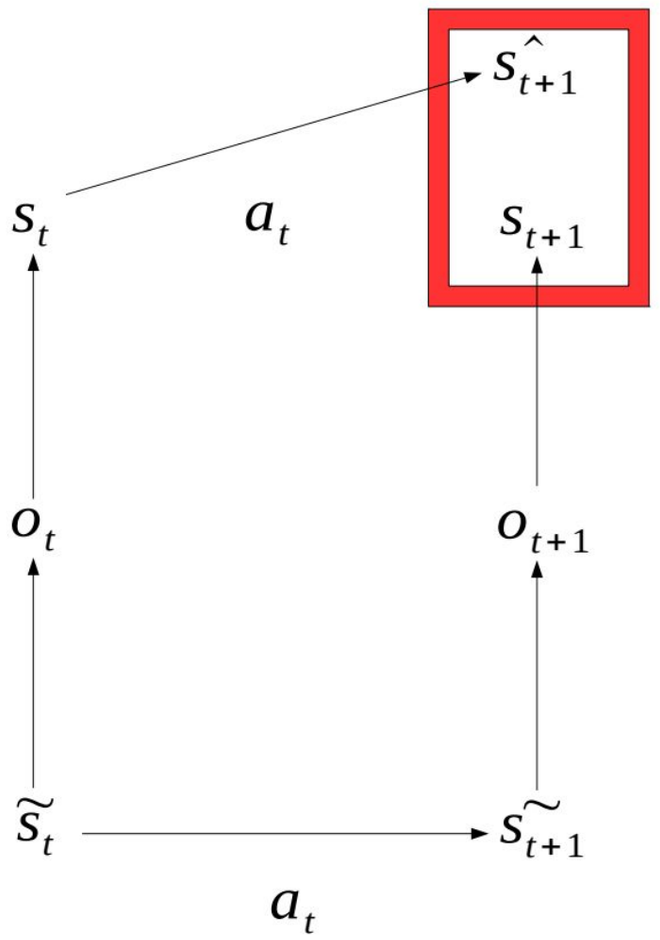


Fig. 5: Illustrations of the tasks in our experiments, as viewed by the robot's camera, showing examples of successful completion by our controllers and a typical failure by the controller without vision.

Deep spatial autoencoders for visuomotor learning [Finn'15]

Forward models:

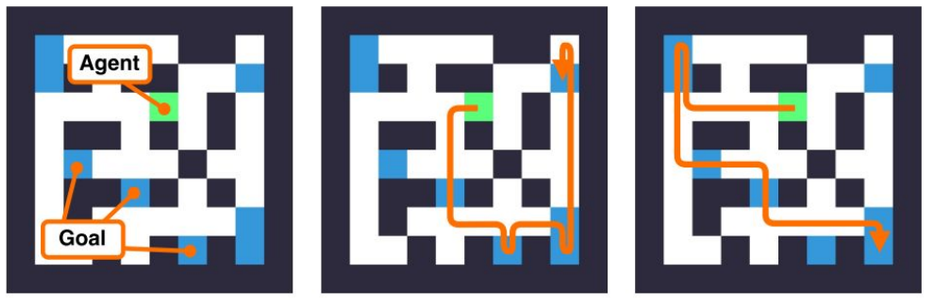
- Find state from which it is easy to predict next state
  - Impose constraints (e.g., simple linear model)
  - Naturally discard irrelevant features
- May be useful in, e.g. model based RL



(b) Forward Model

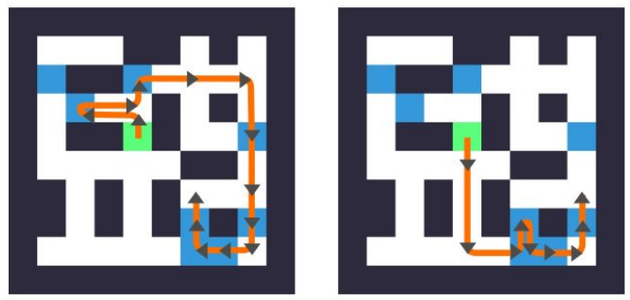
$$\hat{s}_{t+1} = f(s_t, a_t; \theta_{fwd})$$

Forward model example: Value Prediction Network [Oh'17]



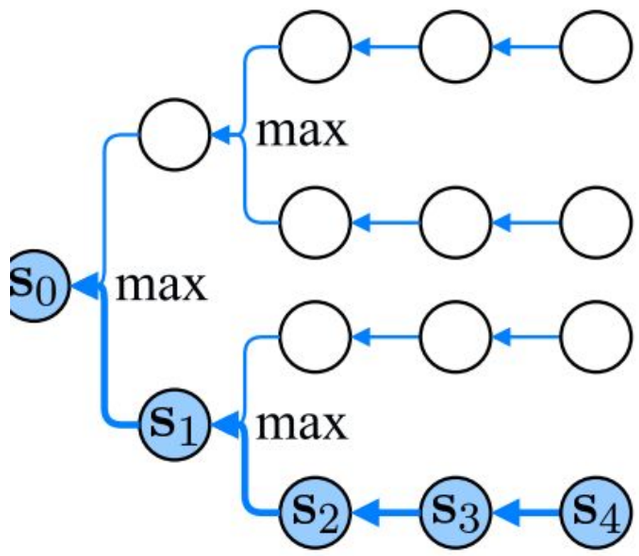
(a) Observation (b) DQN's trajectory (c) VPN's trajectory

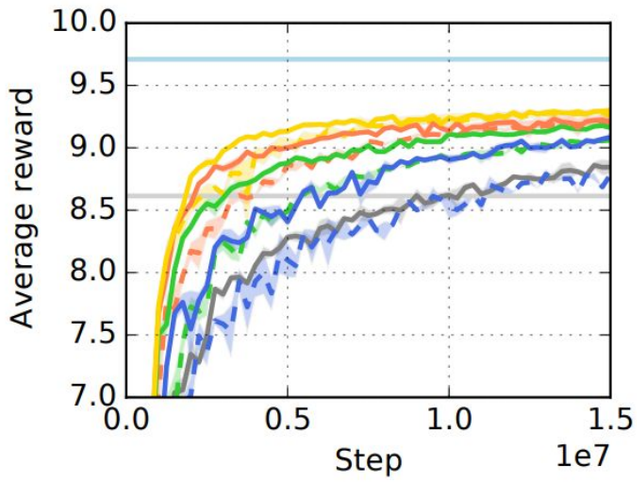
Figure 4: Collect domain. (a) The agent should collect as many goals as possible within a time limit which is given as additional input. (b-c) DQN collects 5 goals given 20 steps, while VPN(5) found the optimal trajectory via planning which collects 6 goals.



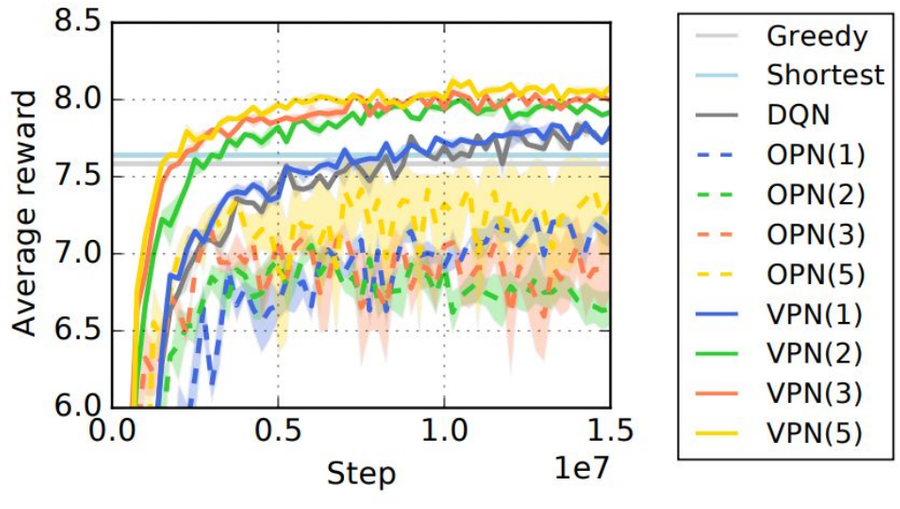
(a) Plan with 20 steps (b) Plan with 12 steps

Figure 5: Example of VPN's plan. VPN can plan the best future options just from the current state. The figures show VPN's different plans depending on the time limit.





(a) Deterministic



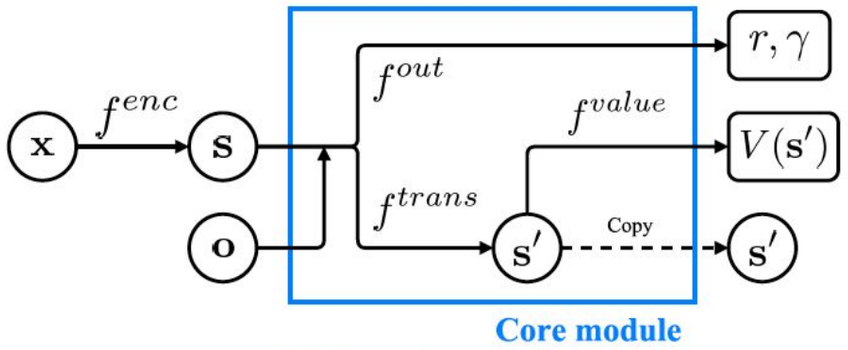
(b) Stochastic

Algorithm 1 Q-value from d-step planning

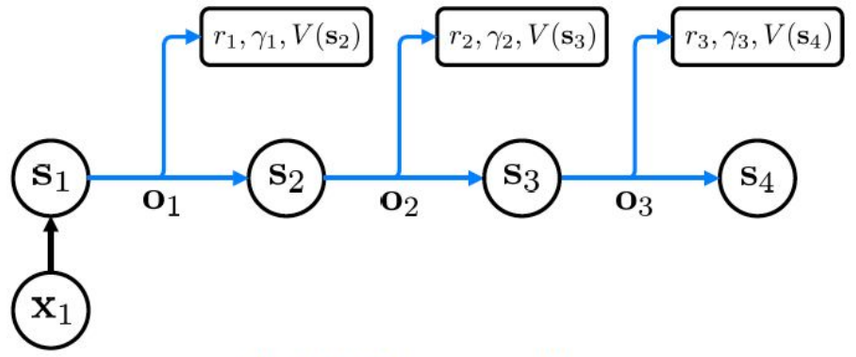
function Q-PLAN($\mathbf{s}, \mathbf{o}, d$)
  $r, \gamma, V(\mathbf{s}'), \mathbf{s}' \leftarrow f_\theta^{core}(\mathbf{s}, \mathbf{o})$
  if $d = 1$ **then**
    return $r + \gamma V(\mathbf{s}')$
  end if
  $\mathcal{A} \leftarrow b$-best options based on $Q^1(\mathbf{s}', \mathbf{o}')$
  for $\mathbf{o}' \in \mathcal{A}$ **do**
    $q_{\mathbf{o}'} \leftarrow$ Q-PLAN($\mathbf{s}', \mathbf{o}', d-1$)
  end for
  return $r + \gamma \left[\frac{1}{d} V(\mathbf{s}') + \frac{d-1}{d} \max_{\mathbf{o}' \in \mathcal{A}} q_{\mathbf{o}'}\right]$
end function

Figure 6: Learning curves on Collect domain. 'VPN(d)' represents VPN with d-step planning, while 'DQN' and 'OPN(d)' are the baselines.

Forward model example: Value Prediction Network [Oh'17]



(a) One-step rollout



(b) Multi-step rollout

The VPN consists of the following modules parameterized by $\theta = \{\theta^{enc}, \theta^{value}, \theta^{out}, \theta^{trans}\}$:

Encoding $f_\theta^{enc} : \mathbf{x} \mapsto \mathbf{s}$ **Value** $f_\theta^{value} : \mathbf{s} \mapsto V_\theta(\mathbf{s})$

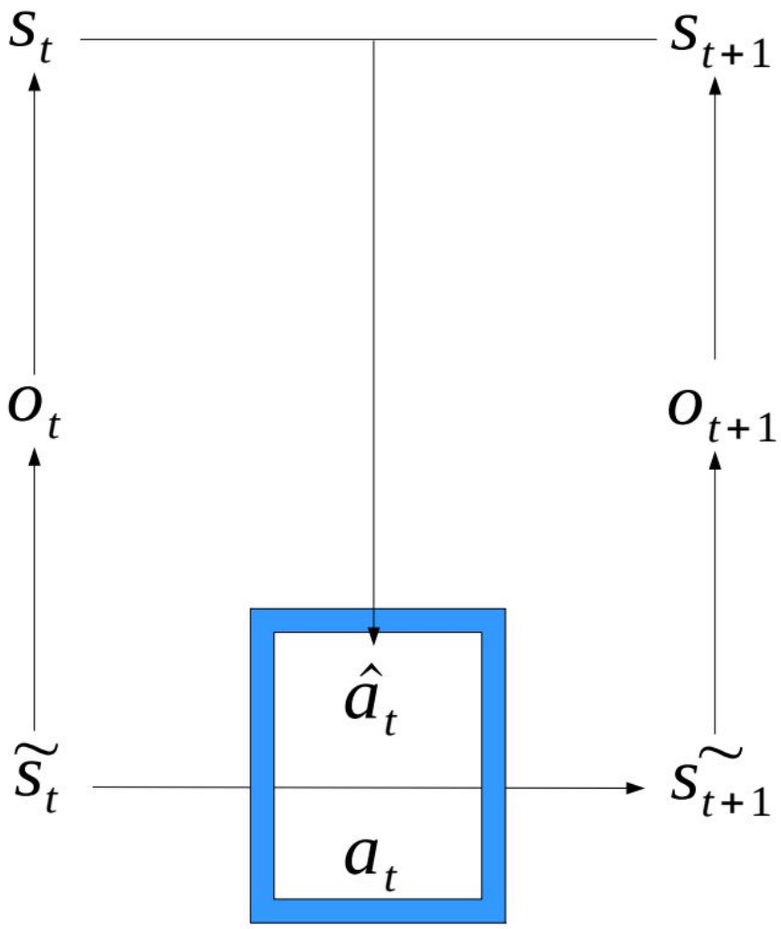
Outcome $f_\theta^{out} : \mathbf{s}, \mathbf{o} \mapsto r, \gamma$ **Transition** $f_\theta^{trans} : \mathbf{s}, \mathbf{o} \mapsto \mathbf{s}'$

- SRL happens in the **Transition** (fwd) model
- Learns the dynamics of the reward values to perform planning (based on **options**)

Inverse models

- Train a state:
  - Sufficient to find actions from 2 observations
  - Useful for a direct control model

- Impose constraints on the model, e.g.:
  - Simple linear model
  - Focus on states that can be controlled



(c) Inverse Model

$$\hat{a}_t = g(s_t, s_{t+1}; \theta_{inv})$$

Inverse model example: Intrinsic Curiosity Module (ICM)

- *Curiosity*: error in an agent's prediction of the consequence of its own actions in a visual feature space (learned by a self-supervised inv. dynamics model).
  - Allows reward to be optional
  - Bypasses difficulties of predicting pixels
- Cons: Not tested on robotics 3D images

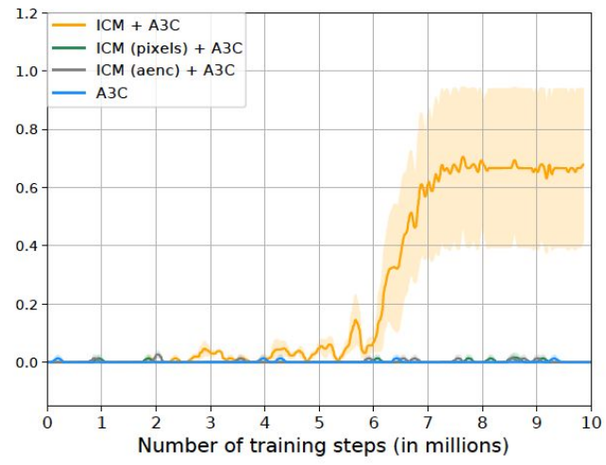
Curiosity-driven Exploration by Self-supervised Prediction

Deepak Pathak Pulkit Agrawal Alexei A. Efros Trevor Darrell
University of California, Berkeley

ICML 2017

[Download Paper] [Github Code]

(a) learn to explore on Level-1 (b) explore faster on Level-2



(c) "very sparse reward" setting

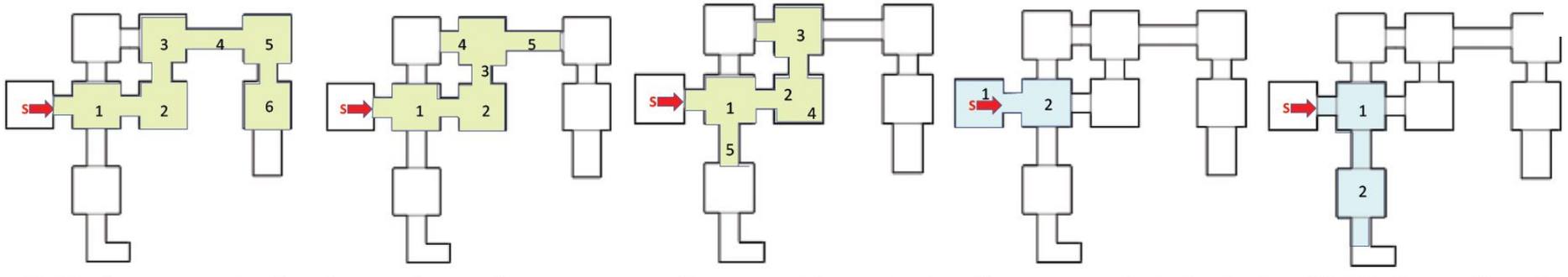
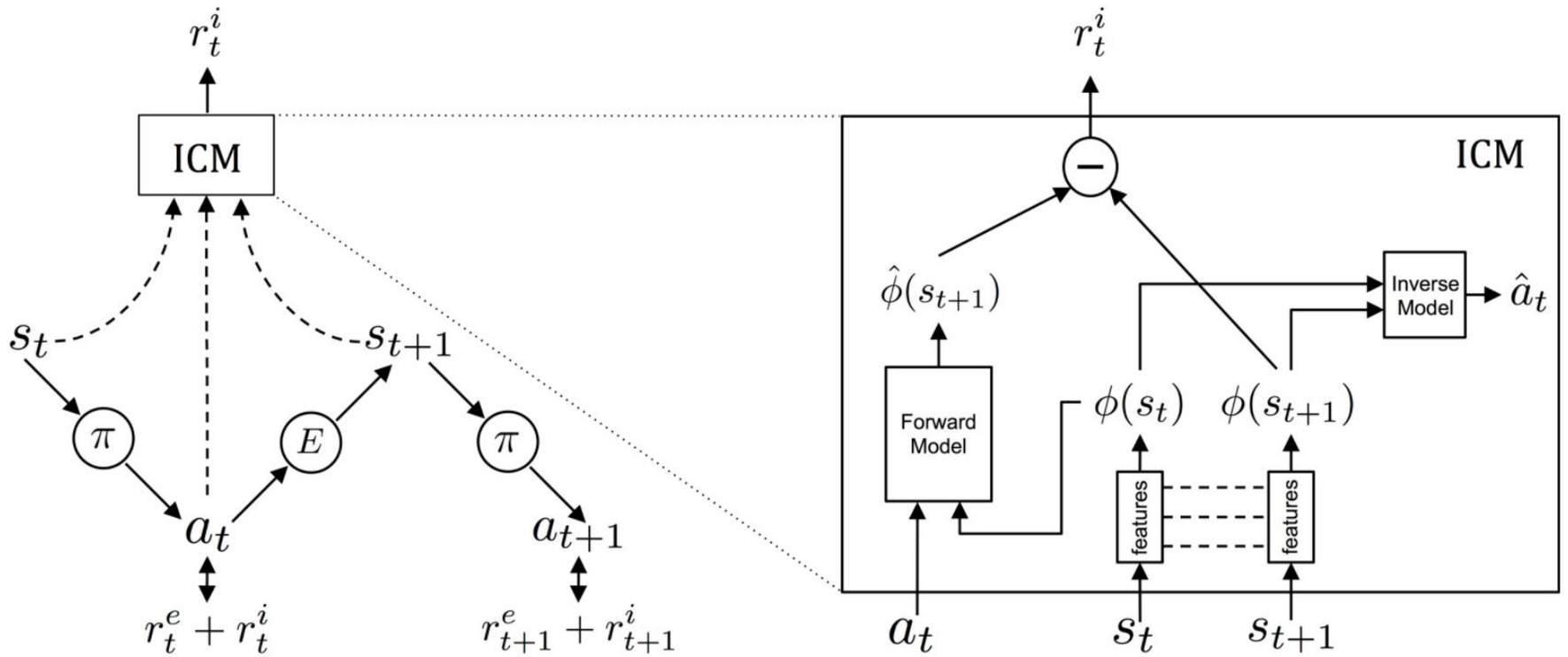


Figure 7. Each column in the figure shows the coverage of an agent by coloring the rooms it visits during 2100 steps of exploration. The red arrow shows the initial location and orientation of the agent at the start of the episode. The first three (in green) and the last two columns (in blue) show visitation of curious (ICM) and randomly exploring agents respectively. The results clearly show that the curious agent trained with intrinsic rewards explores a significantly larger number of rooms as compared to a randomly exploring agent.

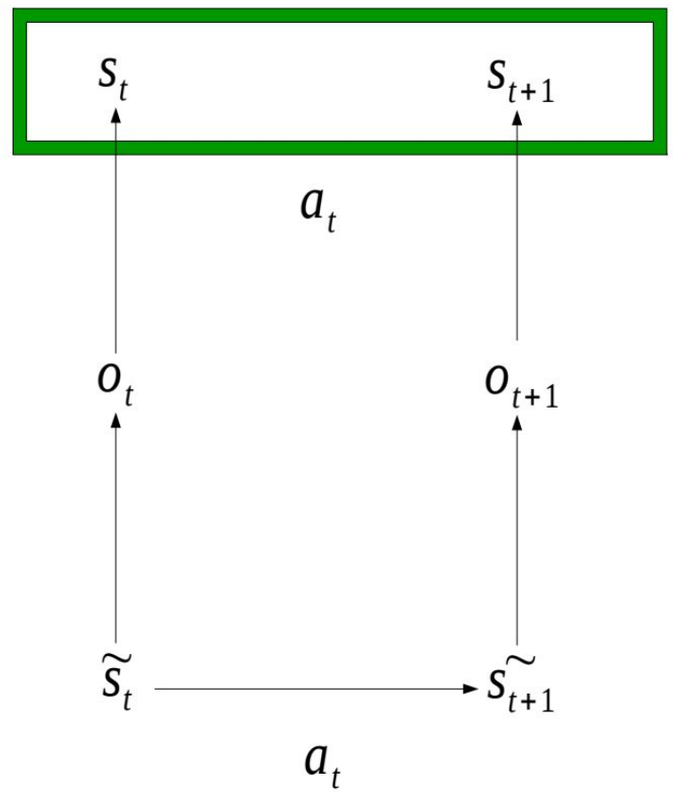
The role of SRL in the intrinsic curiosity model

- The fwd model encodes constraints and learns the state space
- Its prediction error is the intrinsic (i.e., curiosity) reward that encourages exploration of the space



Priors models

- Encode high-level constraints on the states:
  - Temporal continuity
  - Controllability
  - Inertia, etc.
- May exploit rewards

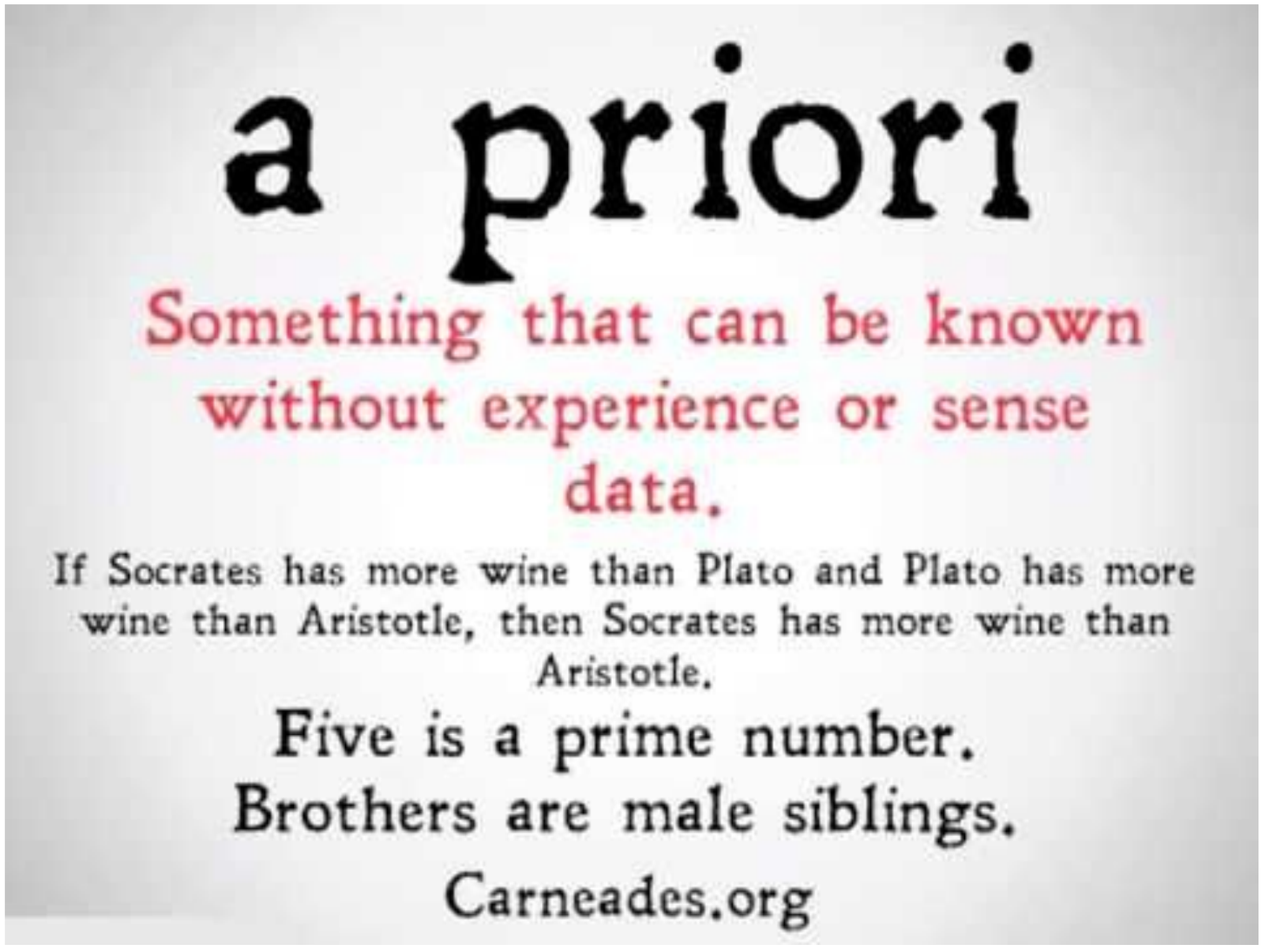


(d) Model with prior

$$Loss = \mathcal{L}_{prior}(s_1, ..., s_n; \theta_\phi | c) \tag{5}$$

Priors models

In case you wonder…
we do not mean PRIORS in the *bayesian* prior probability sense...rather:



Robotic Priors

[Jonschkowski et. al. 2015]

Use *a priori* knowledge to learn representations more relevant to the task

Robotic Priors [1] provide the model with basic knowledge about the environment dynamical features. Each prior is encoded as a loss function:

- **Temporal coherence Prior:** *Two states close to each other in time are also close to each other in the state representation space.*

$$L_{Temp}(D, \hat{\phi}) = \mathrm{E}[\| \Delta \hat{s}_t \|^2] \,, \tag{1}$$

- **Proportionality Prior:** *Two identical actions should result in two proportional magnitude state variations.*

$$L_{Prop}(D, \hat{\phi}) = \mathrm{E}[(\| \Delta \hat{s}_{t_2} \| - \| \Delta \hat{s}_{t_1} \|)^2 | a_{t_1} = a_{t_2}] \,, \tag{2}$$

- **Repeatability Prior:** *Two identical actions applied at similar states should provide similar state variations, not only in magnitude but also in direction.*

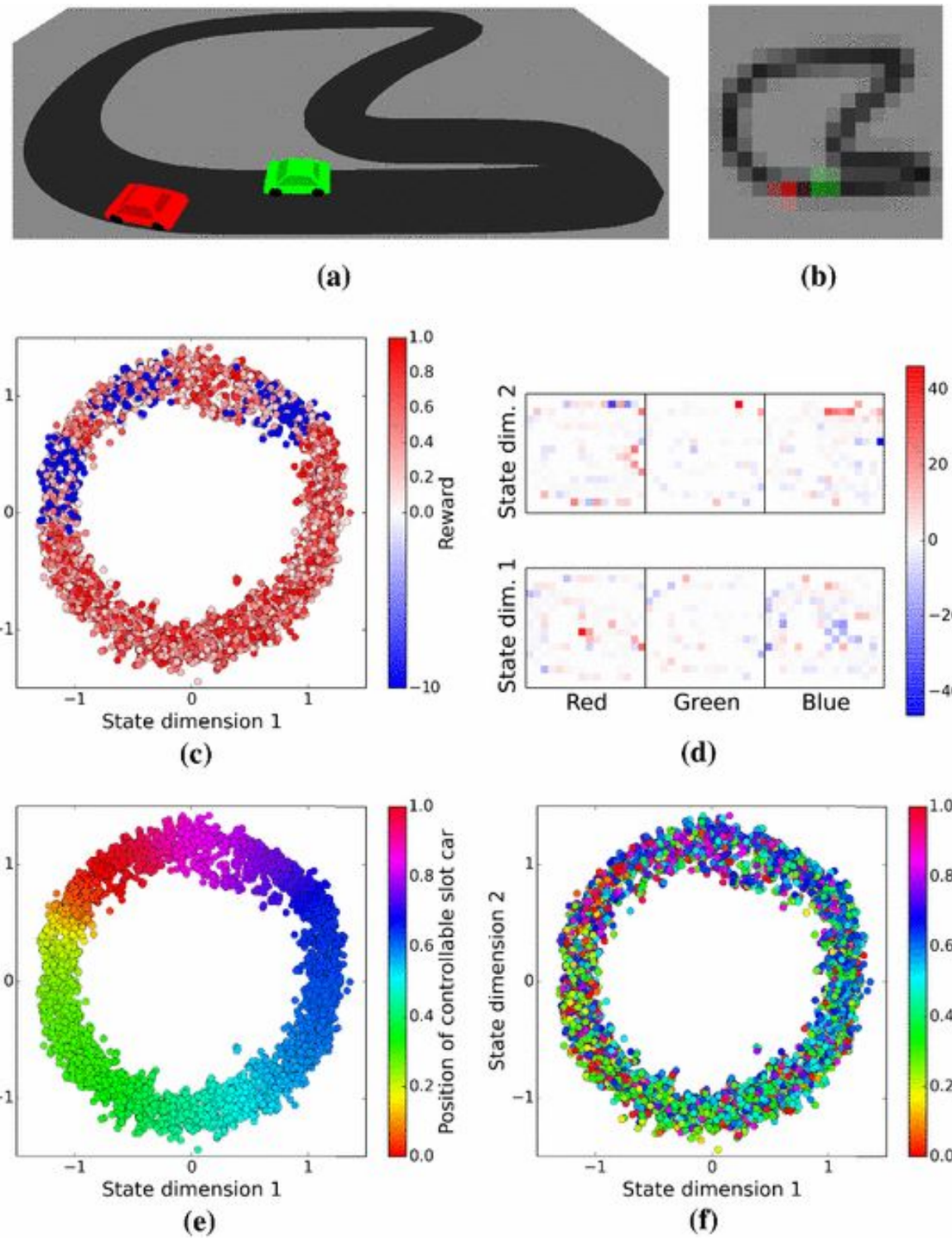
$$L_{Rep}(D, \hat{\phi}) = \mathrm{E}[e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|^2} \| \Delta \hat{s}_{t_2} - \Delta \hat{s}_{t_1} \|^2 | \, a_{t_1} = a_{t_2}] \,, \tag{3}$$

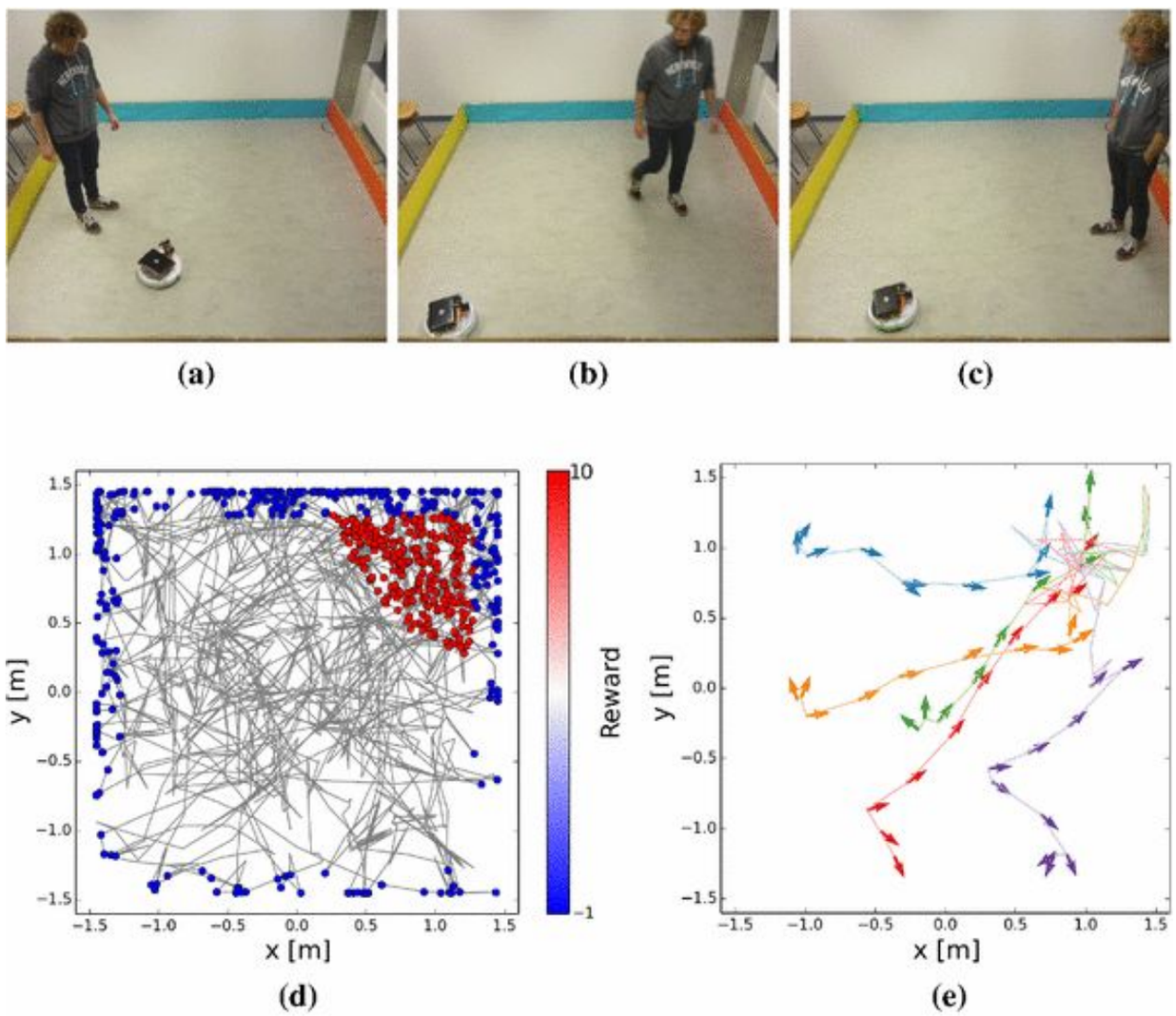
- **Causality Prior:** *If two states on which the same action is applied give two different rewards, they should not be close to each other in the state representation space.*

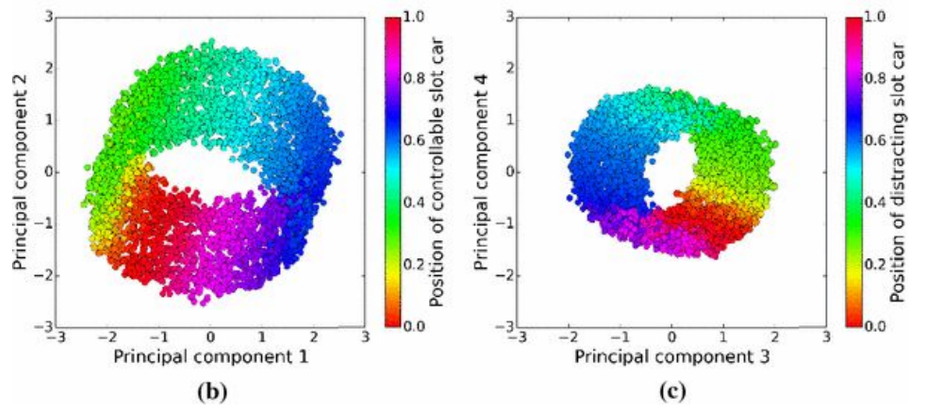
$$L_{Caus}(D, \hat{\phi}) = \mathrm{E}[e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|^2} \mid a_{t_1} = a_{t_2}, r_{t_1+1} \neq r_{t_2+1}] \,, \tag{4}$$

Robotic Priors

[Jonschkowski et. al. 2015]







Robotic Priors

[Jonschkowski et. al. 2015]

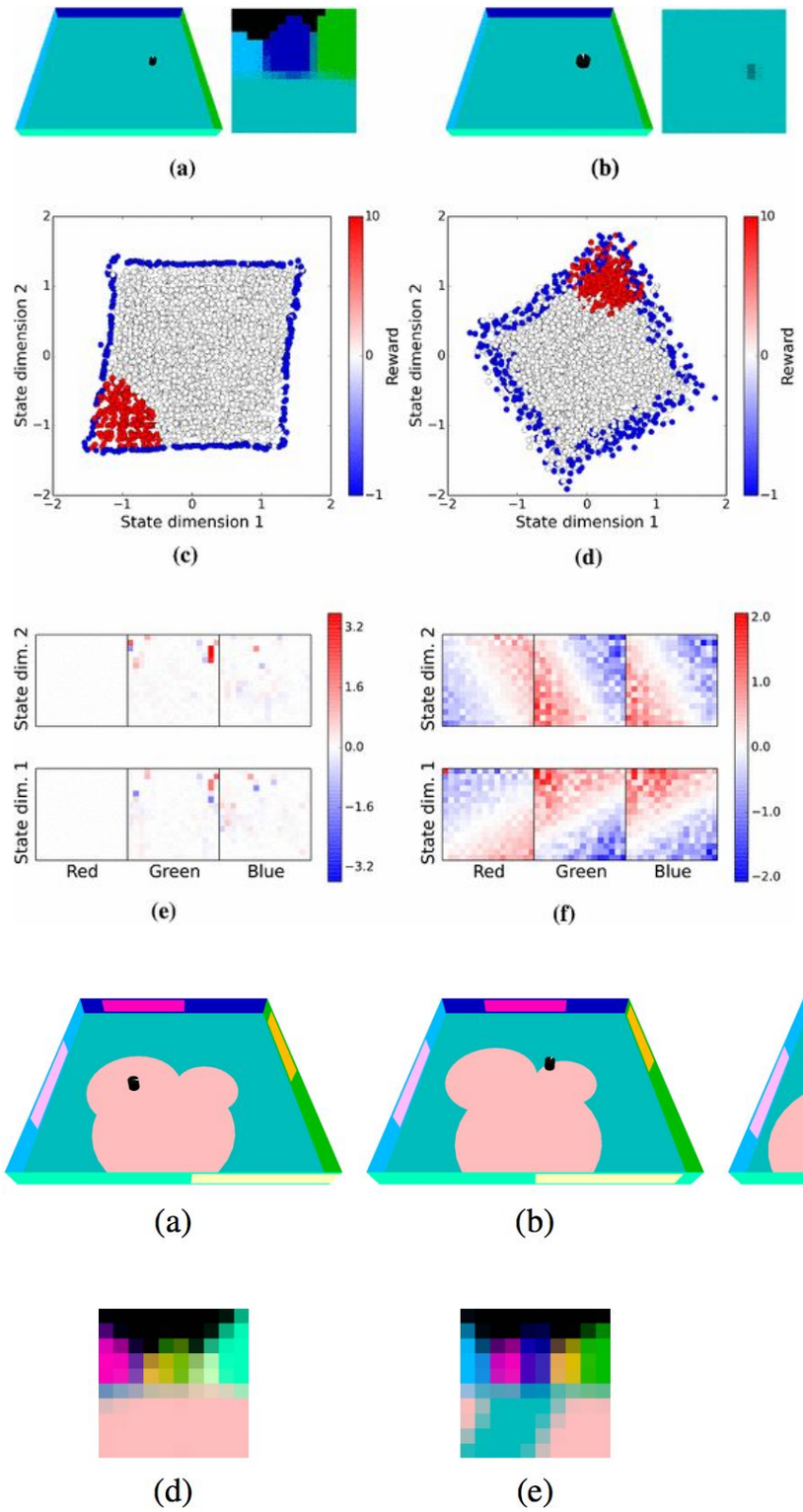


Fig. 7. Extended navigation task. (a-c) show the robot moving to the upper right corner while the distractors move randomly. (d-f) show the respective observations (note how they are influenced by the distractors).

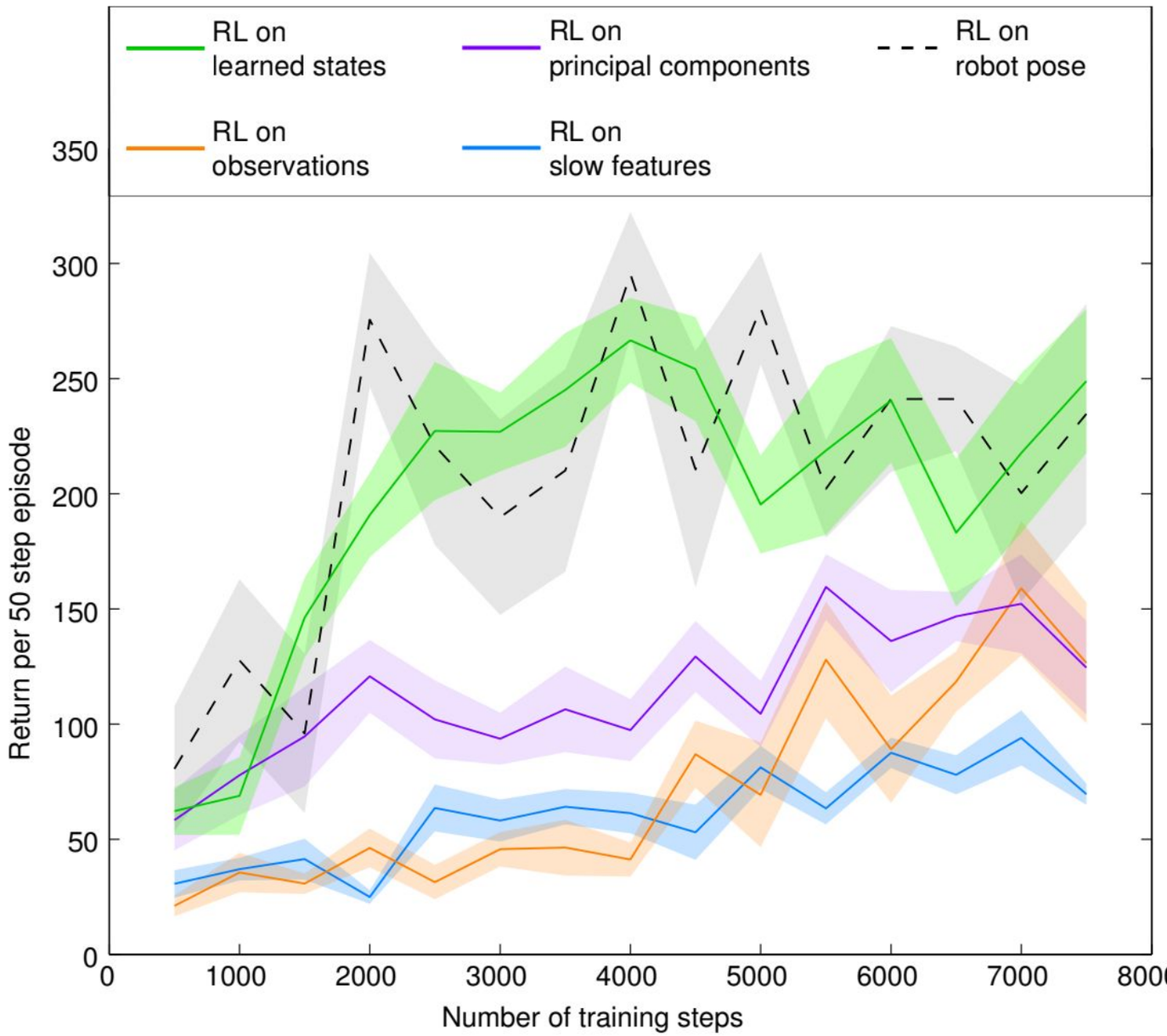


Fig. 8. Reinforcement learning performance for different state representations. Lines show means, surfaces display their standard errors.

Robotic Priors: Position Velocity Encoders

(PVE) [Jonschkowski et. al. 2017]

PVE encode an observation into a position state and velocity estimation. $\mathbf{s}_t^{(v)} = \alpha(\mathbf{s}_t^{(p)} - \mathbf{s}_{t-1}^{(p)}), \quad \mathbf{s}_t^{(a)} = \mathbf{s}_t^{(v)} - \mathbf{s}_{t-1}^{(v)}. \quad \mathbf{s}_t = \begin{bmatrix} \mathbf{s}_t^{(p)} \\ \mathbf{s}_t^{(v)} \end{bmatrix}.$

- ***Variation***: Positions of relevant things vary (representation of such positions should do as well).
- ***Slowness***: Positions change slowly (velocities are slow)
- ***Inertia***: Velocities change slowly (slowness applied to velocities).
- ***Conservation***: Velocity magnitudes change slowly (law of conservation of energy: similar amount of energy (”movement”) in consecutive time steps)
- ***Controllability***: Controllable things (those whose accelerations correlate with the actions of the robot) are relevant.

Robotic Priors: Position Velocity Encoders

(PVE) [Jonschkowski et. al. 2017]

$$L_{\text{variation}} = \mathbf{E}\left[e^{-\|\mathbf{s}_a^{(p)} - \mathbf{s}_b^{(p)}\|}\right]$$

$$L_{\text{slowness}} = \mathbf{E}\left[\|\mathbf{s}_t^{(p)} - \mathbf{s}_{t-1}^{(p)}\|^2\right]$$

$$L_{\text{inertia}} = \mathbf{E}\left[\|\mathbf{s}_t^{(v)} - \mathbf{s}_{t-1}^{(v)}\|^2\right] = \mathbf{E}\left[\|\mathbf{s}_t^{(a)}\|^2\right]$$

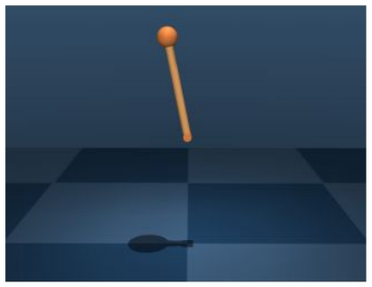
$$L_{\text{conservation}} = \mathbf{E}\left[\left(\|\mathbf{s}_t^{(v)}\| - \|\mathbf{s}_{t-1}^{(v)}\|\right)^2\right]$$

$$L_{\text{controlability (i)}} = e^{-\text{Cov}(\mathbf{a}_{t,i}, \mathbf{s}_{t+1,i}^{(a)})}$$

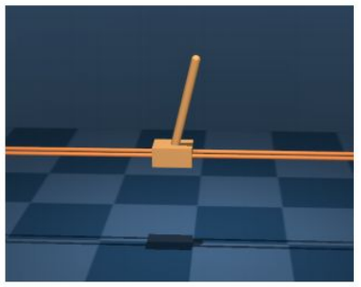
$$= e^{-\mathbf{E}\left[\left(a_{t,i} - \mathbf{E}[a_{t,i}]\right)\left(s_{t+1,i}^{(a)} - \mathbf{E}[s_{t+1,i}^{(a)}]\right)\right]}$$

Robotic Priors: Position Velocity Encoders

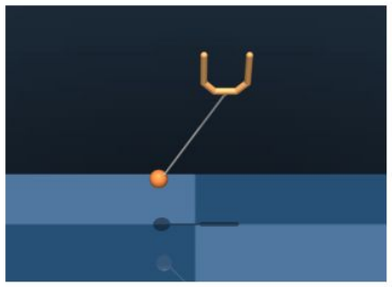
(PVE) [Jonschkowski et. al. 2017]



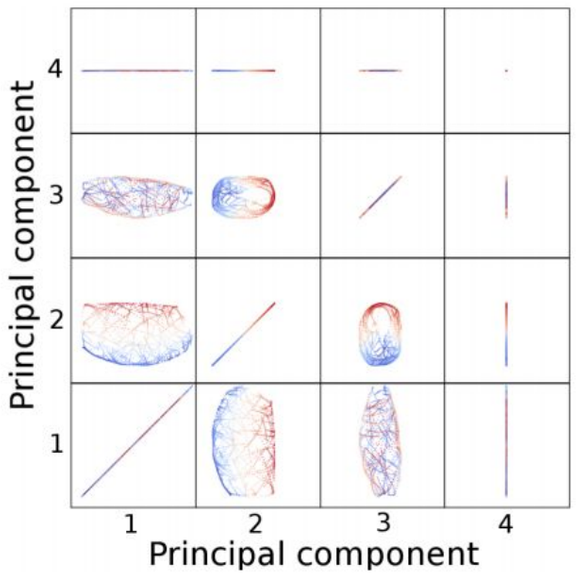
(a) Inverted pendulum



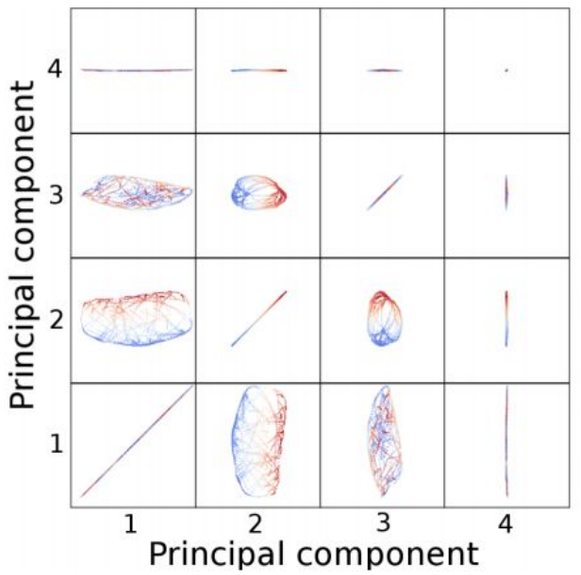
(b) Cart-pole



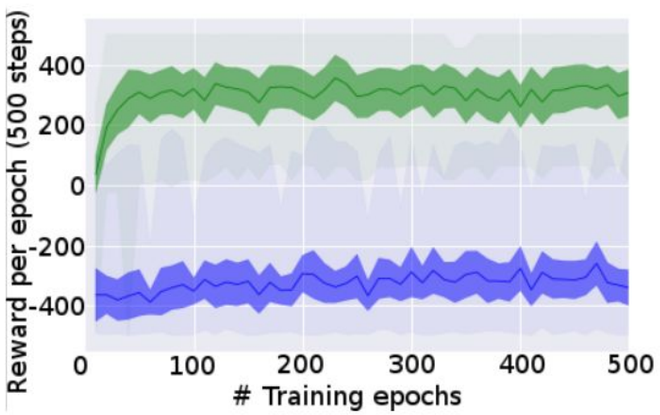
(c) Ball in cup



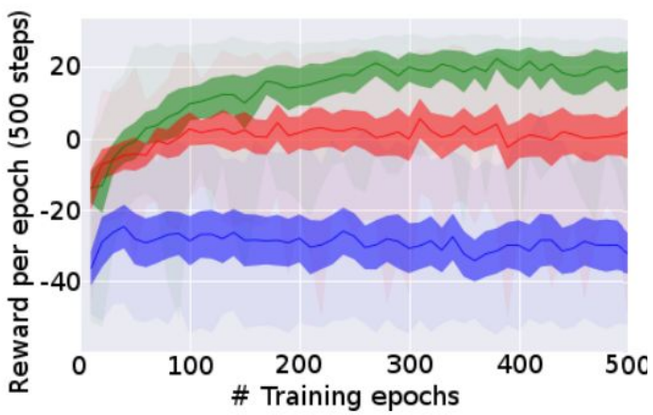
(a) Learned representation (moving camera)



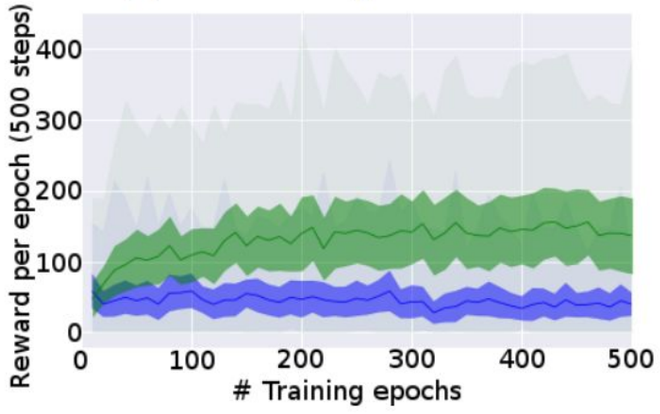
(b) Learned representation (static camera)



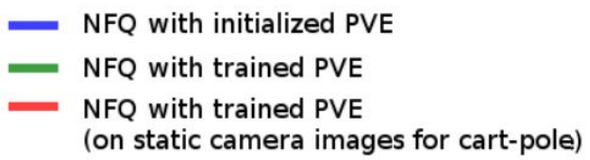
(a) Inverted pendulum

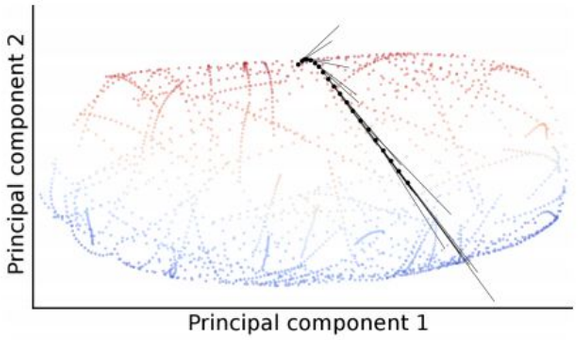


(b) Cart-pole

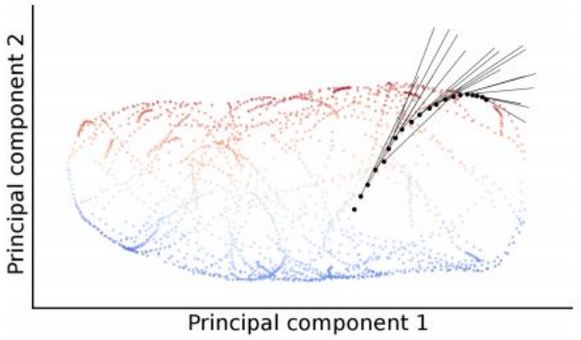


(c) Ball in cup

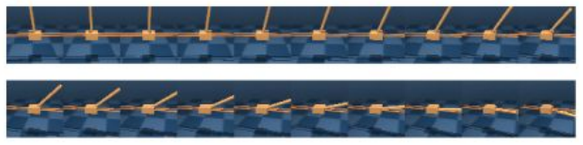




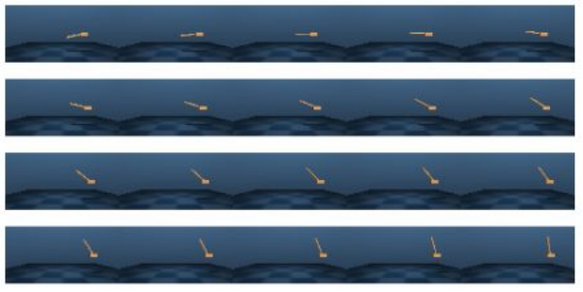
(c) Encoded sequence (moving camera)



(d) Encoded sequence (static camera)



(e) Observation sequence (moving camera)



(f) Observation sequence (static camera)

Other objective functions

Selectivity: [Thomas'17]

- *A good representation should be a disentangled representation of variation factors*.
- If actions are known to be independent, it measures the independence among variations of the representation under each action.

$$\mathcal{L}_{sel}(D, \phi, k) = \mathbb{E}\left[\frac{\| s_{t+1}^{(k)} - s_t^{(k)} \|}{\sum_{k'} \| s_{t+1}^{(k')} - s_t^{(k')} \|} | s_{t+1} \sim P^a_{s_t, s_{t+1}}\right] \quad (16)$$

Multi-criteria objective functions: Embed to Control (E2C)

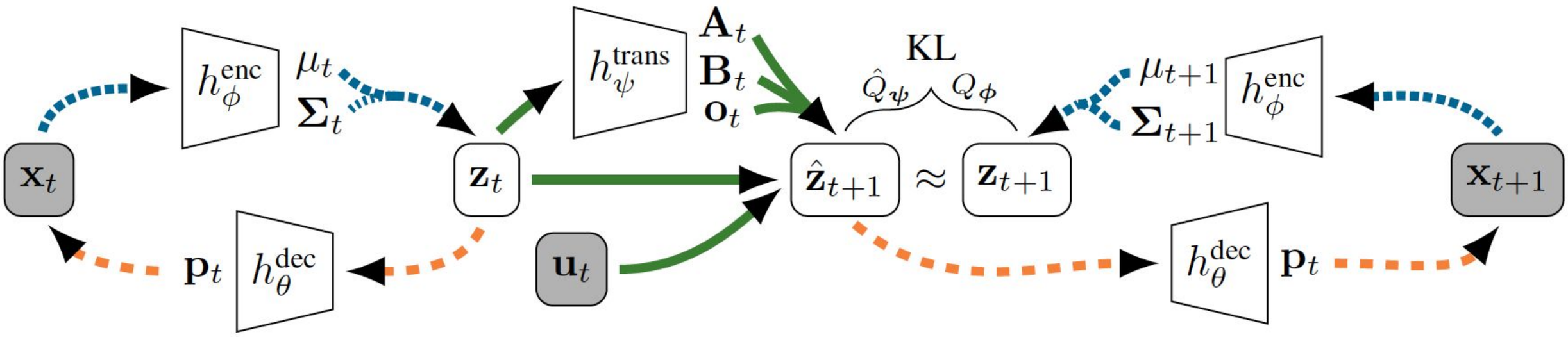
[Watter'18]

.

- A model to learn the mean and supplementary params for the variance of the distrib. (6)
  - W, U, V fixed or learned
  - Allows using KL-divergence to train a fwd model.

$$\hat{s}_{t+1} = W * \hat{s}_t + U * a_t + V \tag{6}$$

$$\hat{s}_{t+1} \sim \mathcal{N}(\mu = W * \hat{s}_t + U * a_t + V, \sigma) \tag{7}$$



SRL models - summary

Model	Actions/Next state constraints	Forward model	Inverse model	Reconstruct observation	Predicts next observation	Uses rewards
AE [Mattner et al., 2012]	no	no	no	yes	no	no
Priors [Jonschkowski and Brock, 2015]	yes	no	no	no	no	yes
PVE [Jonschkowski et al., 2017]	yes	no	no	no	no	no
E2C [Watter et al., 2015]	yes	yes	no	yes	yes	no
ML-DDPG [Munk et al., 2016]	yes	yes	no	no	no	yes
VAE/DAE [van Hoof et al., 2016]	yes	yes	no	yes	yes	no
AE [Finn et al., 2015]	yes	no	no	yes	no	no
DVBF [Karl et al., 2016]	yes	yes	no	yes	yes	no
[Goroshin et al., 2015]	yes	yes	no	yes	yes	no
ICM [Pathak et al., 2017]	yes	yes	yes	no	no	no

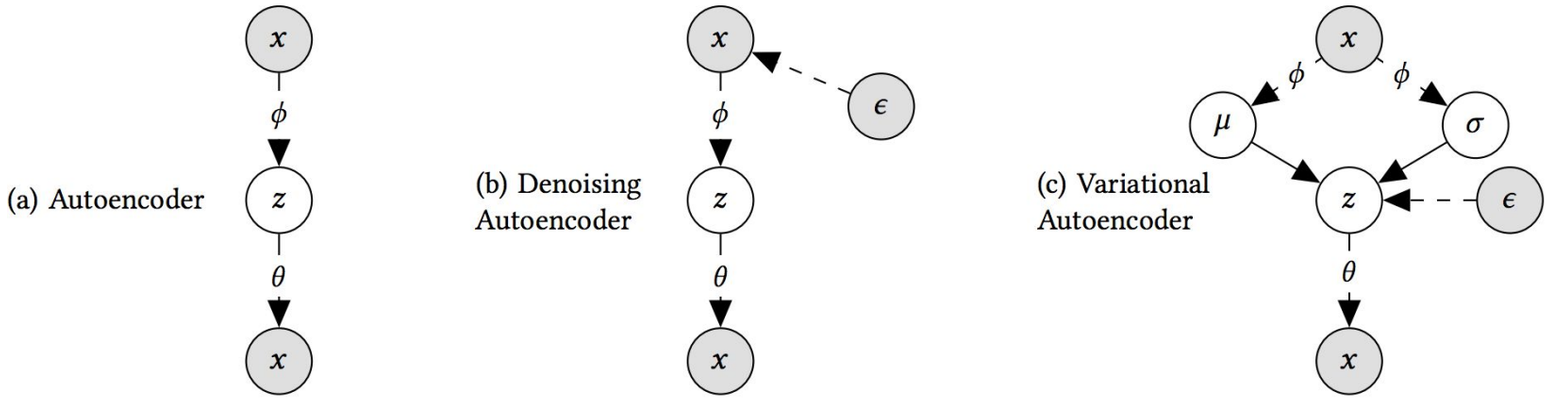
SRL models - summary

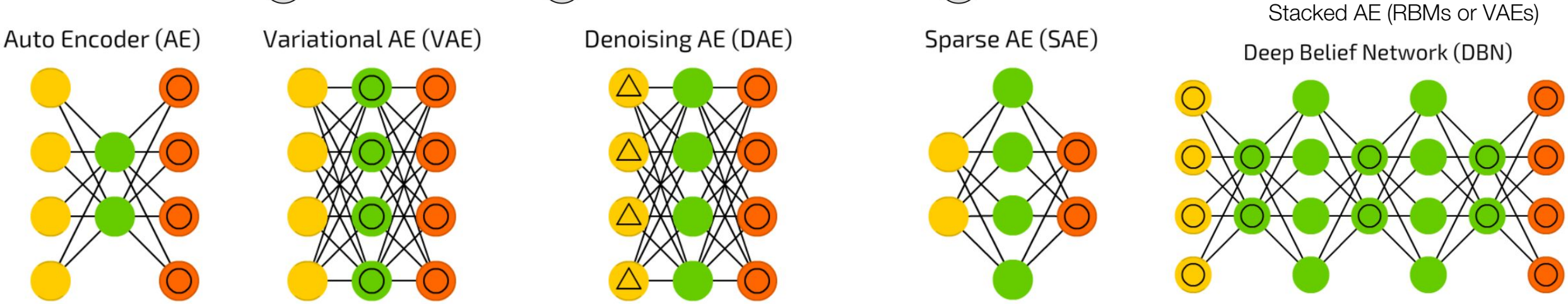
Model	Actions/Next state constraints	Forward model	Inverse model	Reconstruct observation	Predicts next observation	Uses rewards
[Shelhamer et al., 2017]	yes	no	yes	no	no	no
VPN [Oh et al., 2017]	no	yes	no	no	no	yes
DDM [Assael et al., 2015]	yes	yes	no	yes	yes	no
AE [Wahlström et al., 2015]	yes	yes	no	yes	yes	no
[Thomas et al., 2017]	yes	no	no	yes	no	no
PCA [Curran et al., 2015]	no	no	no	yes	no	no
PCA [Curran et al., 2016]	no	no	no	yes	no	no
rwPCA [Parisi, 2017]	no	no	no	yes	no	yes
[Magrans de Abril and Kanai, 2018]	yes	yes	no	no	no	yes
InfoGAN [Chen et al., 2016]	no	no	no	yes	no	no
BiGAN [Donahue et al., 2016]	no	no	no	yes	no	no
[Duan, 2017]	yes	yes	yes	yes	yes	no

Learning tools

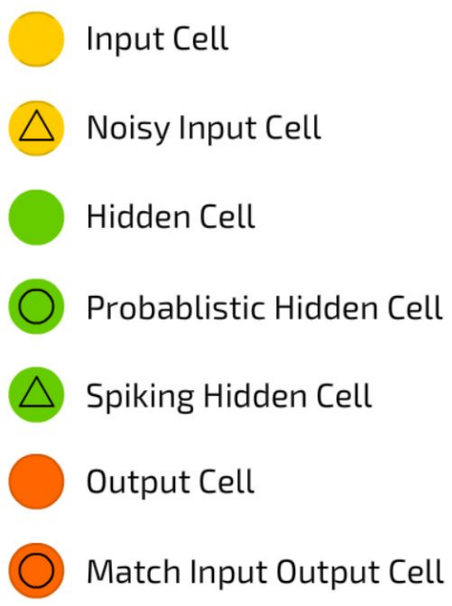
- Denoising (DAE), variational (VAE) and *vanilla* auto-encoders
- Siamese networks
- GANs
- Other objectives

Autoencoders: A taxonomy [Kramer'91, Salakhutdinov'06, Bengio'09]

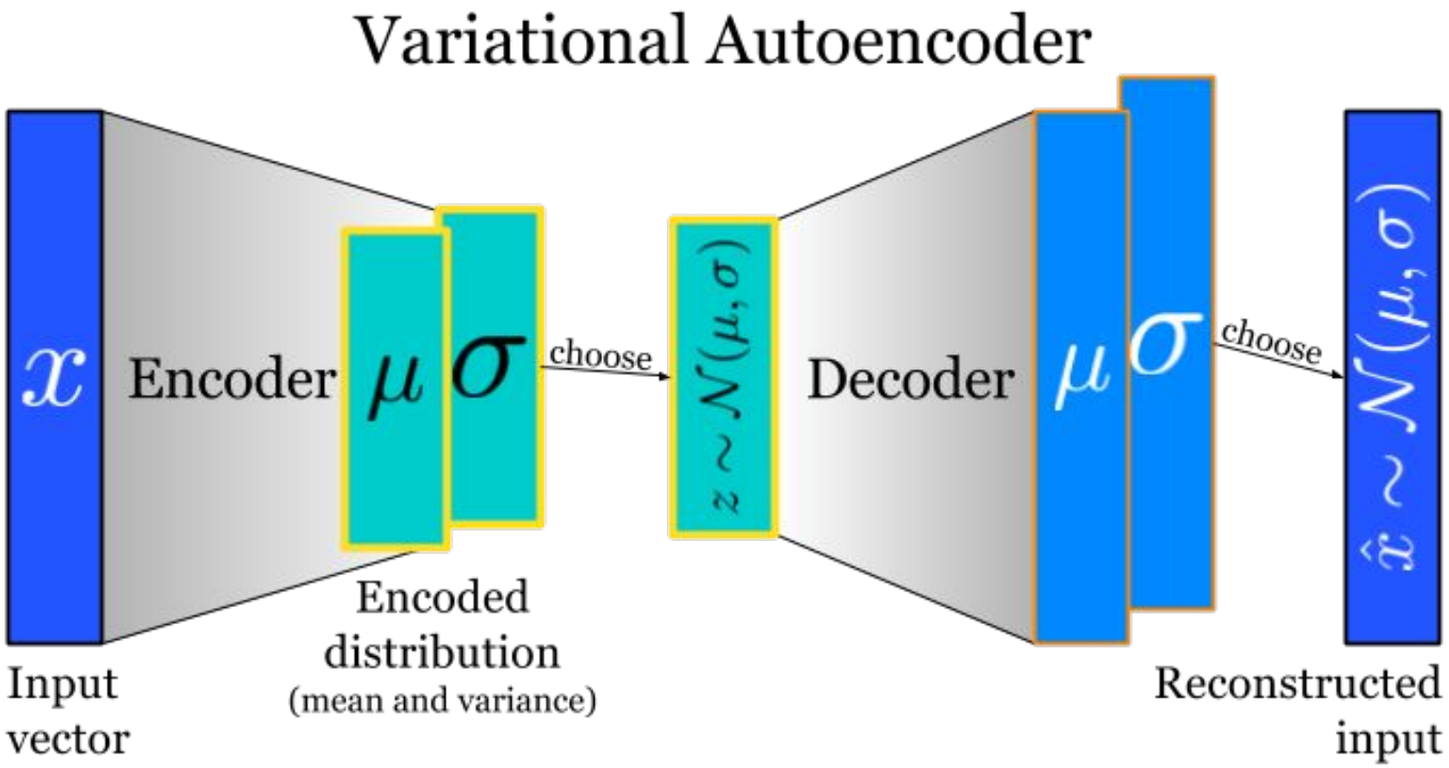


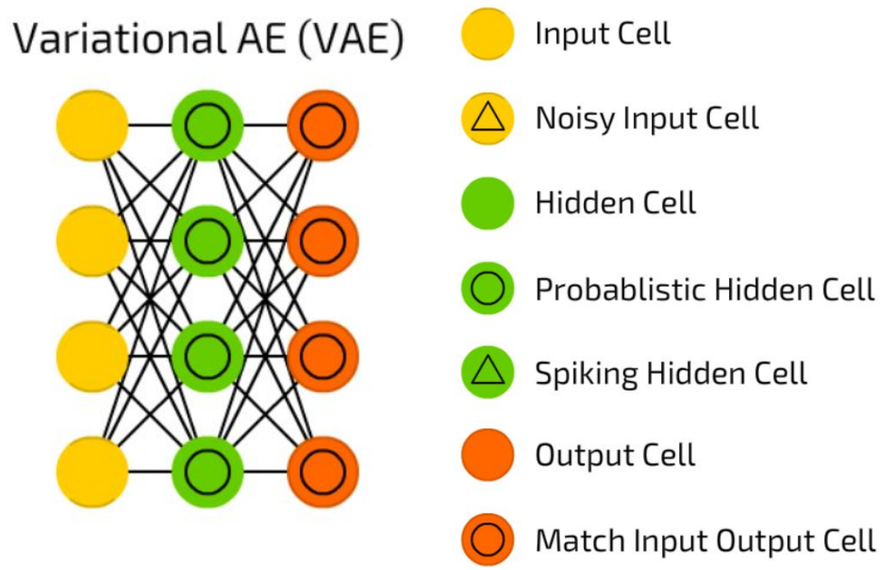


- Learn compact and relevant (undercomplete -bottleneck- and overcomplete [Zazakci'16]) representations minimizing the reconstruction error:
  - DAE inject noise in input, VAE in stochastic hidden layer
- Cons: Unable to ignore task-irrelevant features (domain randomization or saliency)



Autoencoders: VAE -makes an AE generative





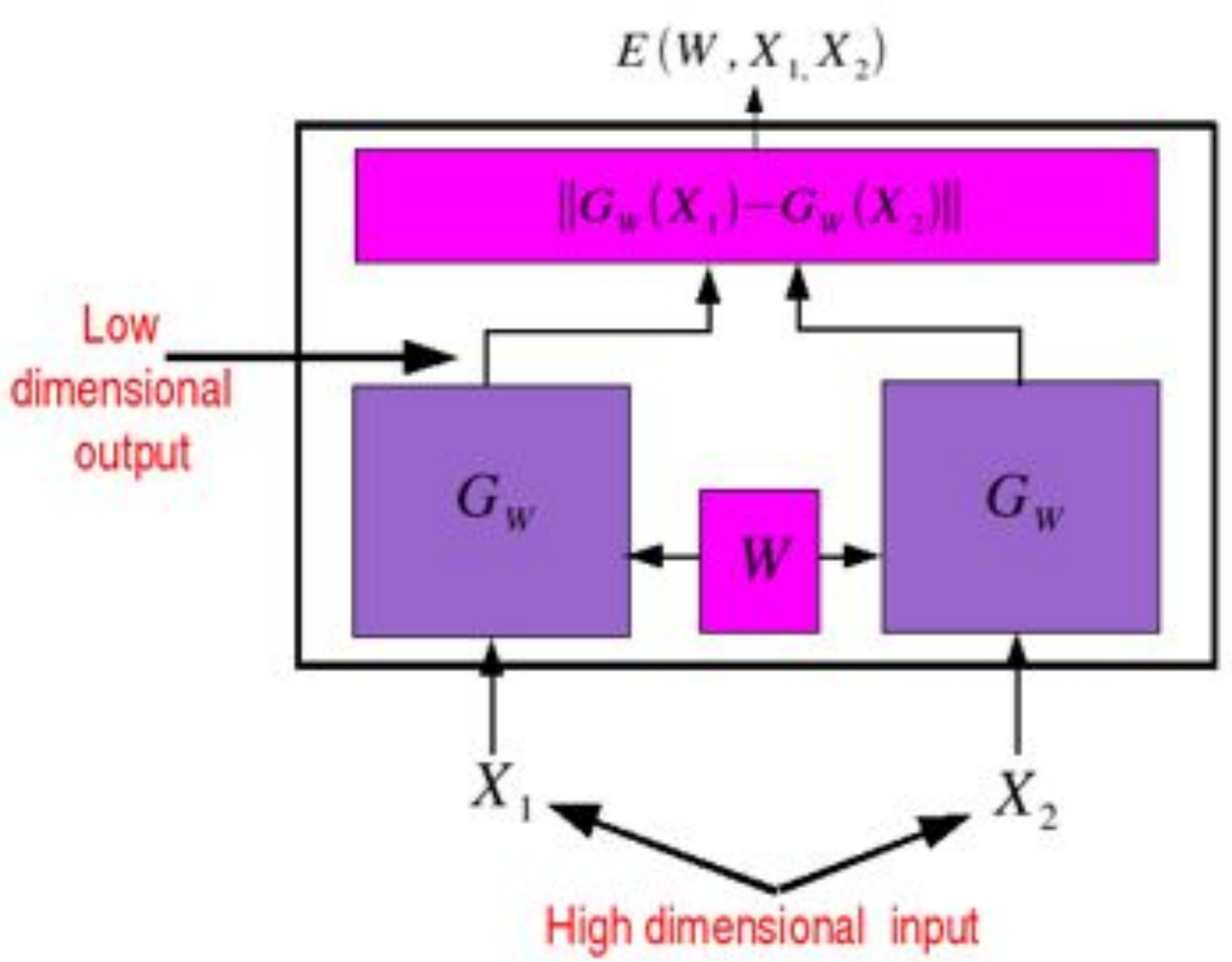
The objective of VAE is to maximize the following variational lower bound with respect to the parameters θ and ϕ.

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \quad (2)$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \mathbb{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})). \quad (3)$$

Siamese Networks

[Chopra'05]



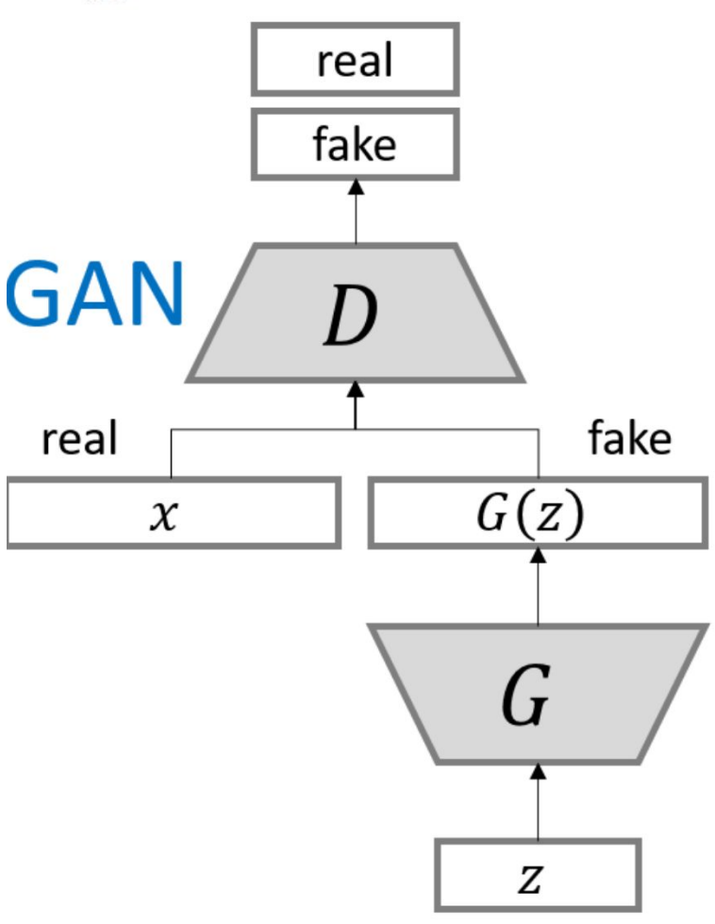
- 2+ identical copies of parameters and loss function
- Discriminatively learns a similarity metric from data
- Suitable:
  - In verification/recognition where #categories is large and unknown
  - Implementing priors (allows multiple passes in parallel)
- Cons: network size doubles

Generative Adversarial Networks

[Goodfellow'14]

- A two-player adversarial (minimax) game with value function V(G, D) (*1: fake, 0: real input*):

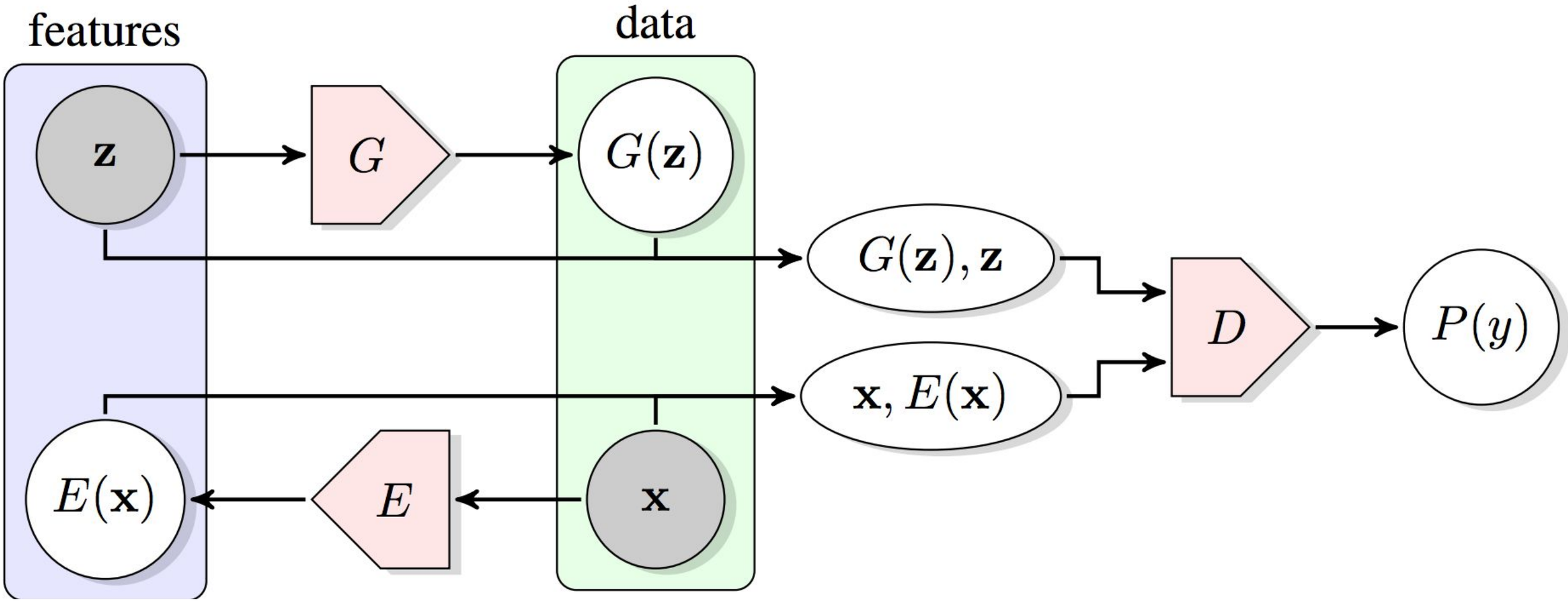
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$



$$L_D^{GAN} = E\left[\log\left(D(x)\right)\right] + E\left[\log\left(1 - D(G(z))\right)\right]$$

$$L_G^{GAN} = E\left[\log\left(D(G(z))\right)\right]$$

BiGAN: Bidirectional Generative Adversarial Network (= ALI)



$$\min_{G,E} \max_{D} V(D, E, G) \tag{2}$$

where

$$V(D, E, G) := \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{X}}} \Big[\underbrace{\mathbb{E}_{\mathbf{z} \sim p_E(\cdot|\mathbf{x})} \left[\log D(\mathbf{x}, \mathbf{z}) \right]}_{\log D(\mathbf{x}, E(\mathbf{x}))} \Big] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{Z}}} \Big[\underbrace{\mathbb{E}_{\mathbf{x} \sim p_G(\cdot|\mathbf{z})} \left[\log \left(1 - D(\mathbf{x}, \mathbf{z}) \right) \right]}_{\log(1 - D(G(\mathbf{z}), \mathbf{z}))} \Big]. \tag{3}$$

BiGAN: Bidirectional Generative Adversarial Network

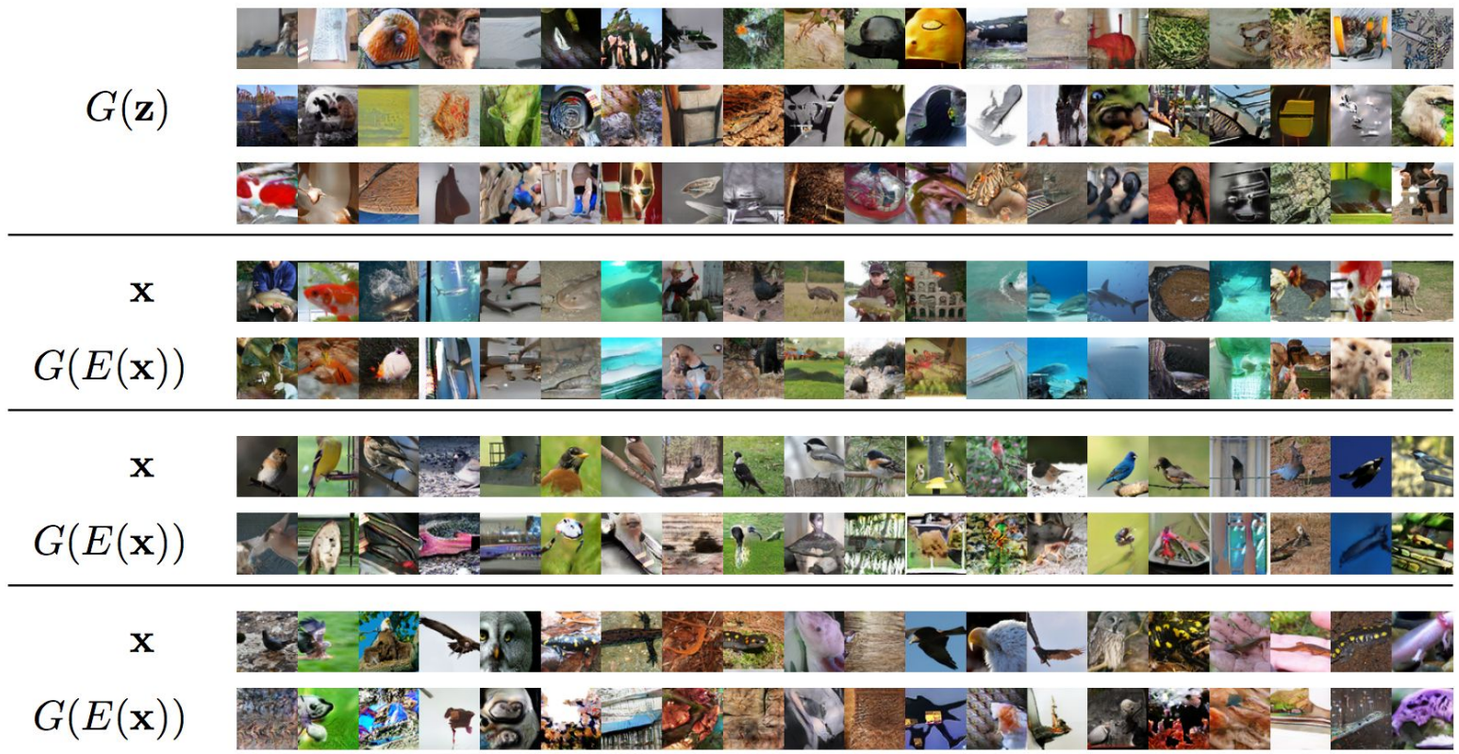
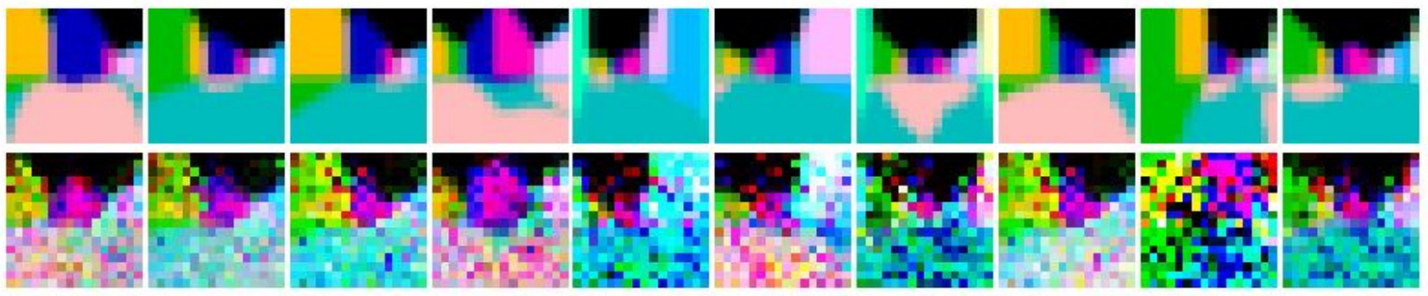


Figure 4: Qualitative results for ImageNet BiGAN training, including generator samples $G(\mathbf{z})$, real data $\mathbf{x}$, and corresponding reconstructions $G(E(\mathbf{x}))$.

BiGAN for SRL: observation reconstruction:



Evaluating state representations: the effect of dimensionality

- Few approaches use a *minimal* state dimension

- Larger dimensions often:
  - Improve performance
  - But limit interpretability

Reference	Observation Dimension	State Dimension	Action Dimension	Environment	Data
Priors [Jonschkowski and Brock, 2015]	16*16*3	2	25 discrete	Slot cars, mobile robot localization	Raw images
PVE[Jonschkow et al., 2017]	Unavailable	5	Discrete	Inverted pendulum, ball in cup, cart-pole	Raw images
E2C [Watter et al., 2015]	40*40*3	8	Discrete	Agent with obstacle	Raw images
E2C [Watter et al., 2015]	48*48*3	8	Discrete	Inverted pendulum	Raw images
E2C [Watter et al., 2015]	80*80*3	8	Discrete	Cart-pole	Raw images
E2C [Watter et al., 2015]	128*128*3	8	Discrete	3 link arm	Raw images
[van Hoof et al., 2016]	20*20*3	3	Continuous	Pendulum swing-up	Raw images
[van Hoof et al., 2016]	228	3	Continuous	Real-robot manipulation task	Tactile data
ML-DDPG [Munk et al., 2016]	18 or 24	6	2 discrete	2 link arm	Joint position
ML-DDPG [Munk et al., 2016]	192 or 308	96	36 discrete	Octopus	Joint position
[Finn et al., 2015]	240*240*3	32	Continuous	Robotics manipulation tasks	Raw images
DVBF [Karl et al., 2016]	16*16*3	3	Unavailable	Pendulum	Raw images
DVBF [Karl et al., 2016]	16*16*3	2	Unavailable	Bouncing ball	Raw images
DVBF [Karl et al., 2016]	16*16*3	12	Unavailable	2 bouncing balls	Raw images
[Goroshin et al., 2015]	3 frames of 32*32	2	2 discrete	NORB dataset	Raw images
ICM [Pathak et al., 2017]	42*42*3	3	4 discrete	3D VizDoom navigation game	Raw images
ICM [Pathak et al., 2017]	42*42*3	2	14 discrete	Mario Bros	Raw images
[Shelhamer et al., 2017]	Unavailable	Un.	Unavailable	Atari	Raw images
VPN [Oh et al., 2017]	3*10*10	Un.	4 discrete	2D navigation	Raw images
VPN [Oh et al., 2017]	4*84*84	Un.	4 discrete	Atari	Raw images
[Curran et al., 2016]	Unavailable	4	5 discrete	Mountain car 3D	Unavailable
[Curran et al., 2016]	Unavailable	12	243 discrete	6 link swimmer	Unavailable
rwPCA [Parisi, 2017]	21*21*3	Auto.	2 continuous	Picking a coin and putting it on a goal	Raw images
[Parisi, 2017]	20	Auto.	2 continuous	Hit a ball with a ping-pong paddle	Position of objects
[Magrans de Abril and Kanai, 2018]	40	2	Continuous	Explore a 3 room simulated 2D map	Position of agent
[Duan, 2017]	240*240*4	512	4 continuous	Poking cube	Raw images + depth

Evaluating learned representations: Environments

- (Inverted) pendulum, cart-pole
- Atari games (mostly 2D), VizDoom (3D)
- Octopus arms, driving/mountain cars, bouncing ball
- Mobile robots, labyrinths, navigation grids
- **Robotics** manipulation skills:
  - Grasping, balancing a pendulum, poking, pushing a button

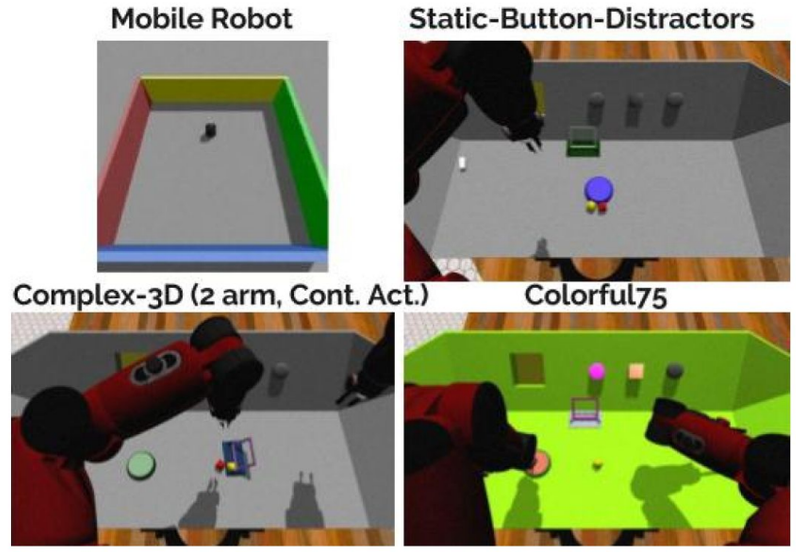
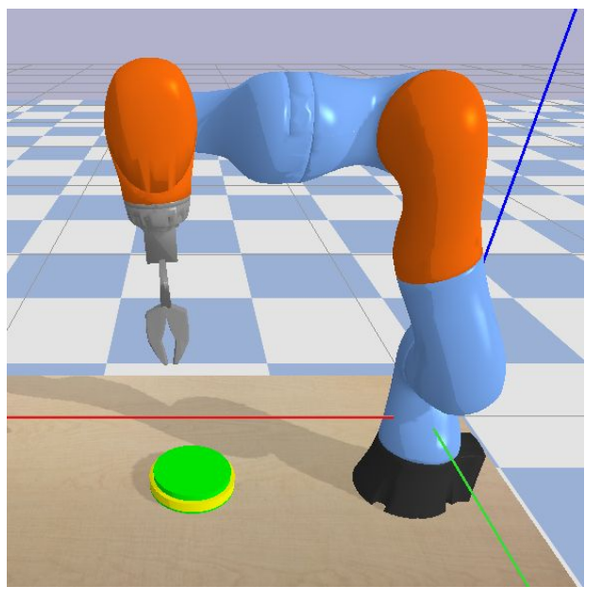


Fig. 3. A sample of each dataset (1-4), created for our benchmark

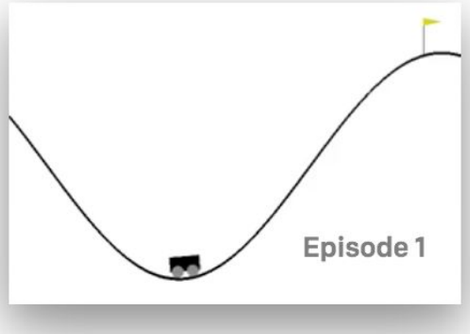


Evaluation environments: Control theory problems from classic RL literature

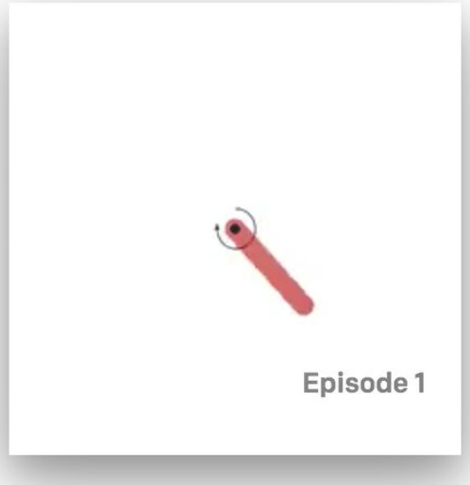
Acrobot-v1
Swing up a two-link robot.

CartPole-v1
Balance a pole on a cart.

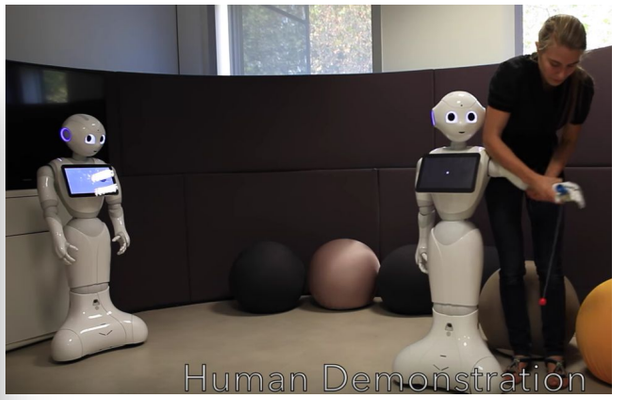
MountainCar-v0
Drive up a big hill.



MountainCarContinuous-v0
Drive up a big hill with continuous control.



Pendulum-v0
Swing up a pendulum.



Ball in cup

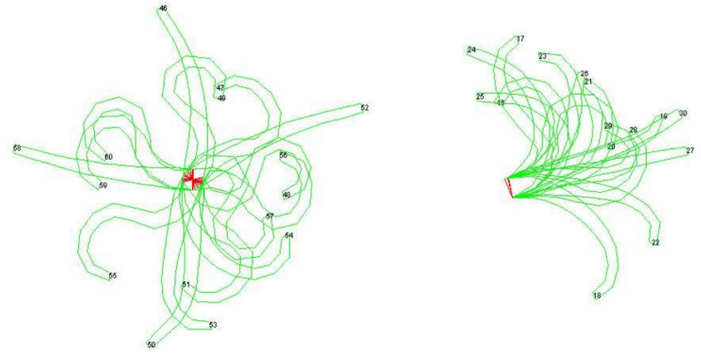


Figure 3: Examples of initial states for the rotating-base experiments (left) and the fixed-base experiments (right). Starting states also include velocities, which are not shown.

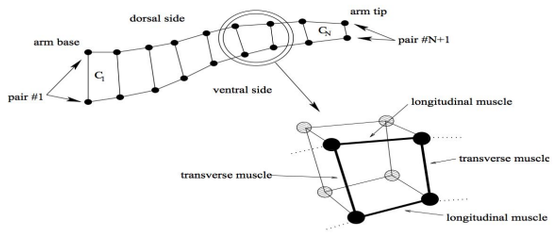


Figure 1: An N compartment simulated Octopus arm. Each constant area compartment C_i is defined by its surrounding 2 longitudinal muscles (ventral and dorsal) and 2 transverse muscles. Circles mark the $2N + 2$ point masses in which the arm's mass is distributed. In the bottom right one compartment is magnified with additional detail.

Figure : Ball-in-cup skill, which the Meka acquired through reinforcement learning

https://gym.openai.com/envs/#classic_control
Octopus arm [Engel'06, Munk'16]

Evaluating state representations: Metrics

Metric	Evaluation target	Context/Observations
Task performance [Jonschkowski and Brock, 2015, Jonschkowski et al., 2017, Munk et al., 2016, van Hoof et al., 2016, Finn et al., 2015, Pathak et al., 2017, Shelhamer et al., 2017, Oh et al., 2017, Parisi, 2017, Assael et al., 2015]	Quality of the state space for a given task	Reinforcement learning
Disentanglement metric score [Higgins et al., 2016]	Data-generating latent factor disentanglement	Transfer learning, object recognition. Assumes generative factors are known and interpretable
Distortion [Indyk, 2001]	Preservation of local and global geometry coherence	Unsupervised representation learning
NIEQA (Normalization Independent Embedding Quality Assessment) [Zhang et al., 2012]	Local & global neighborhood embedding quality assessment	Manifold learning, not limited to isometric embeddings
KNN-MSE [Lesort et al., 2017]	Task related representation learning	Unsupervised SLR for robotics

Metrics: KNN-MSE:

States that are neighbours in the ground truth should be neighbours in the learned state space

For an image I, this criterion is computed as follows:

$$\textbf{KNN-MSE}(I) = \frac{1}{k} \sum_{I' \in KNN(I,k)} ||\phi(I) - \phi(I')||^2 \qquad (6)$$

where $\text{KNN}(I, k)$ returns the k nearest neighbors of I in the learned state space and $\phi(I)$ gives the ground truth (s) associated to I.

Criterion	GT	Superv	4Priors	5Priors	AE	5Priors 0f
KNN-MSE	0.024	0.03	0.079	0.053	0.099	0.047
NIEQA local	0	0.239	0.66	0.50	0.599	0.52
NIEQA global	0	0.048	0.41	0.20	0.465	0.21

TABLE II

Static-Button-Distractor 3D (DATASET 2) RESULTS. *x*F MEANS X RESNET FROZEN LAYERS, GT: (HAND POSITION) GROUND TRUTH

Unsupervised state representation learning with robotic priors: a robustness benchmark [Lesort'17]

Future challenges/directions

- SRL for autonomous agents
  - Assessment, reproducibility, interpretability
- Automatically decide state dimension
- Efficient data gathering for SRL by blending in:
    - Learning by demonstration-> with human in the loop!
    - Few-shot learning
    - GANs
    - Intrinsic motivations via GEP
- -> LET'S NIPS/CORL/ICML!

(MAIN) REFERENCES

- GAN Zoo and GANs presentation
- Deep learning Zoo
- Lesort et. al. *Unsupervised state representation learning with robotic priors: a robustness benchmark*, 2017, DEMO video
- Lesort et al. *State Representation learning for control: an Overview* (submitted to Neural Networks, Elsevier)
- Doncieux et al. *Representational Redescription in Open-ended Learning: a Conceptual Framework* (in prep.)

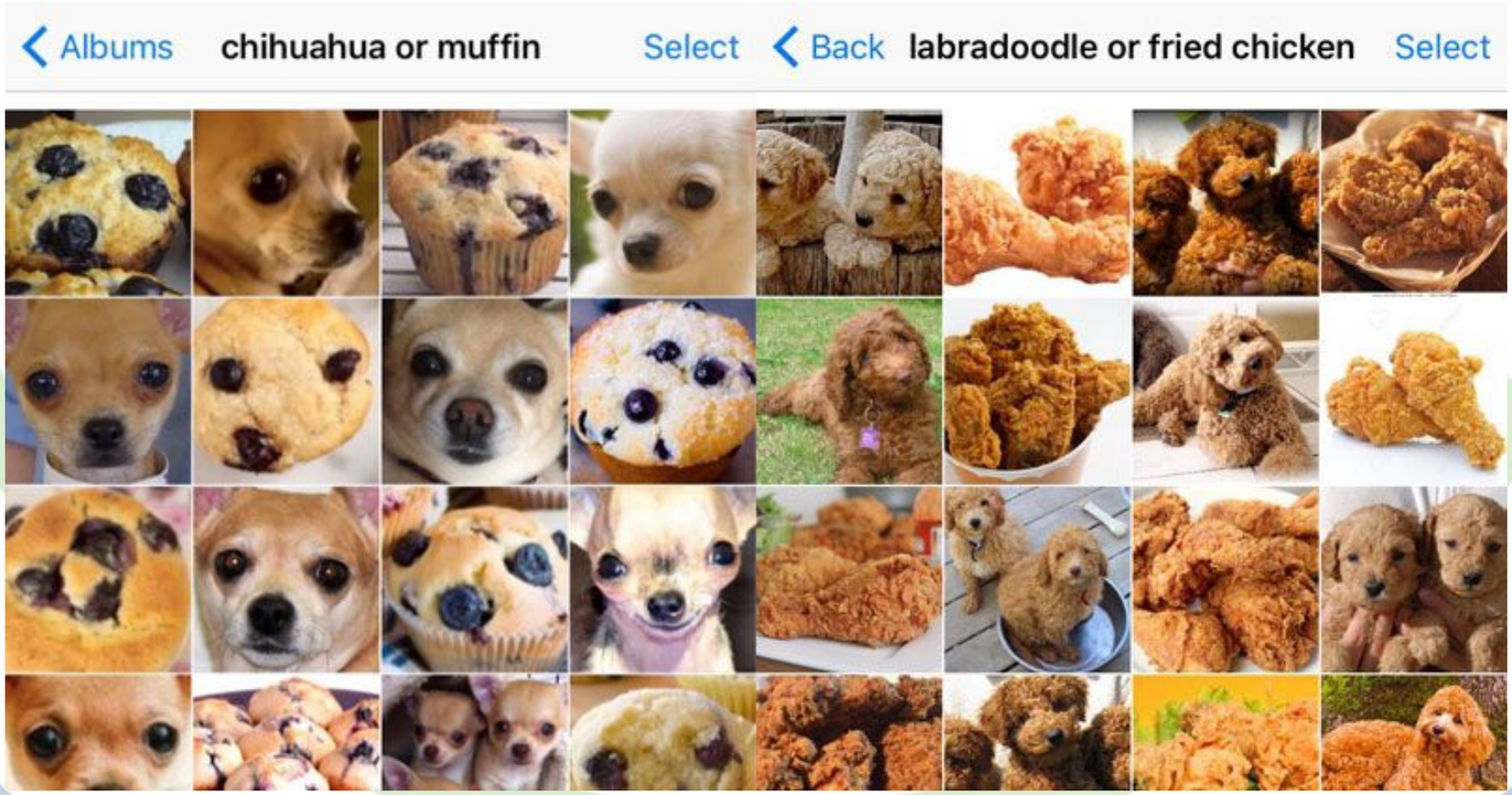
Thank you!

Ideas?

natalia.diaz@ensta-paristech.fr

@NataliaDiazRodr

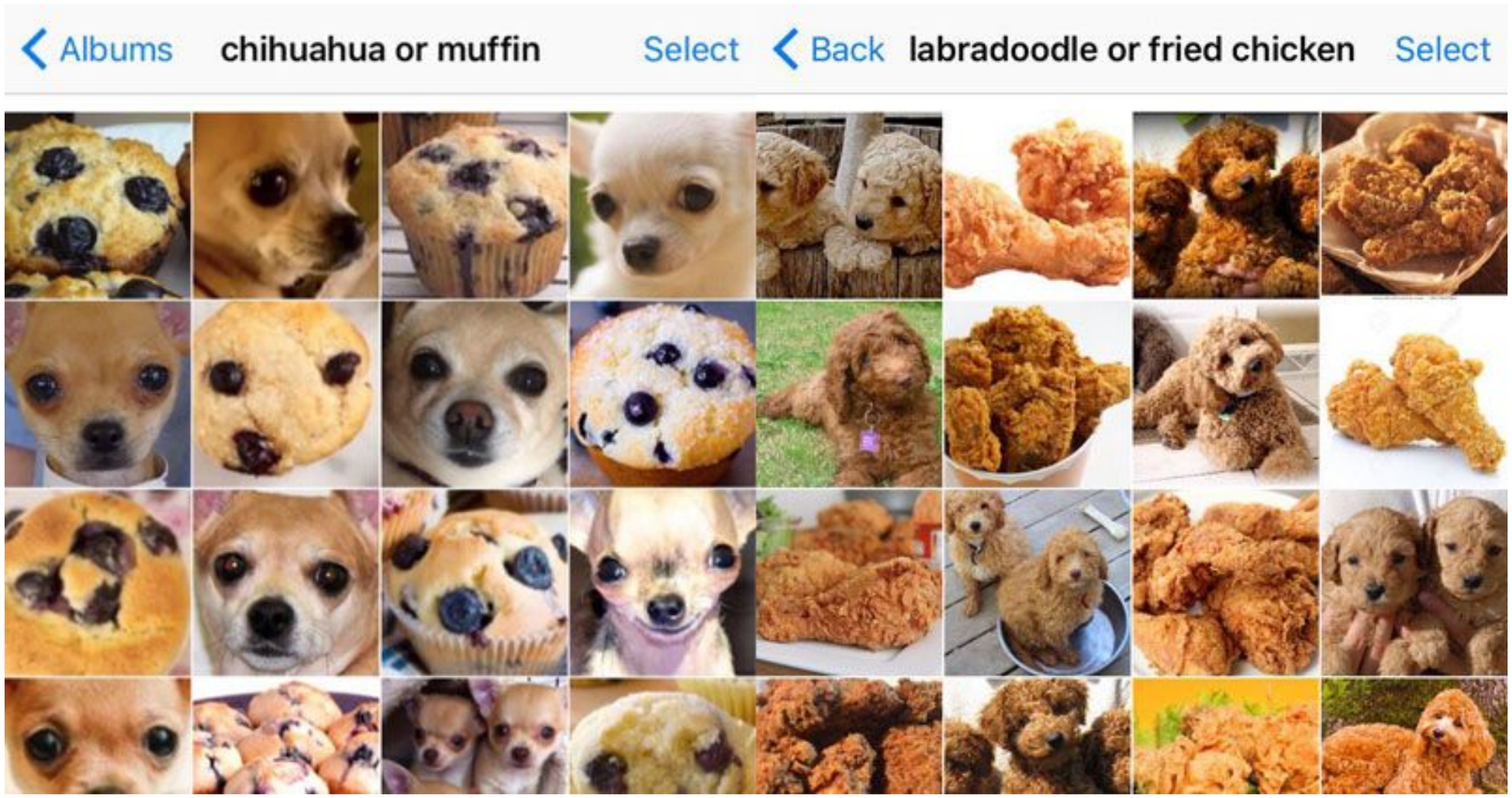
eit Digital ALUMNI



Appendix

Future directions

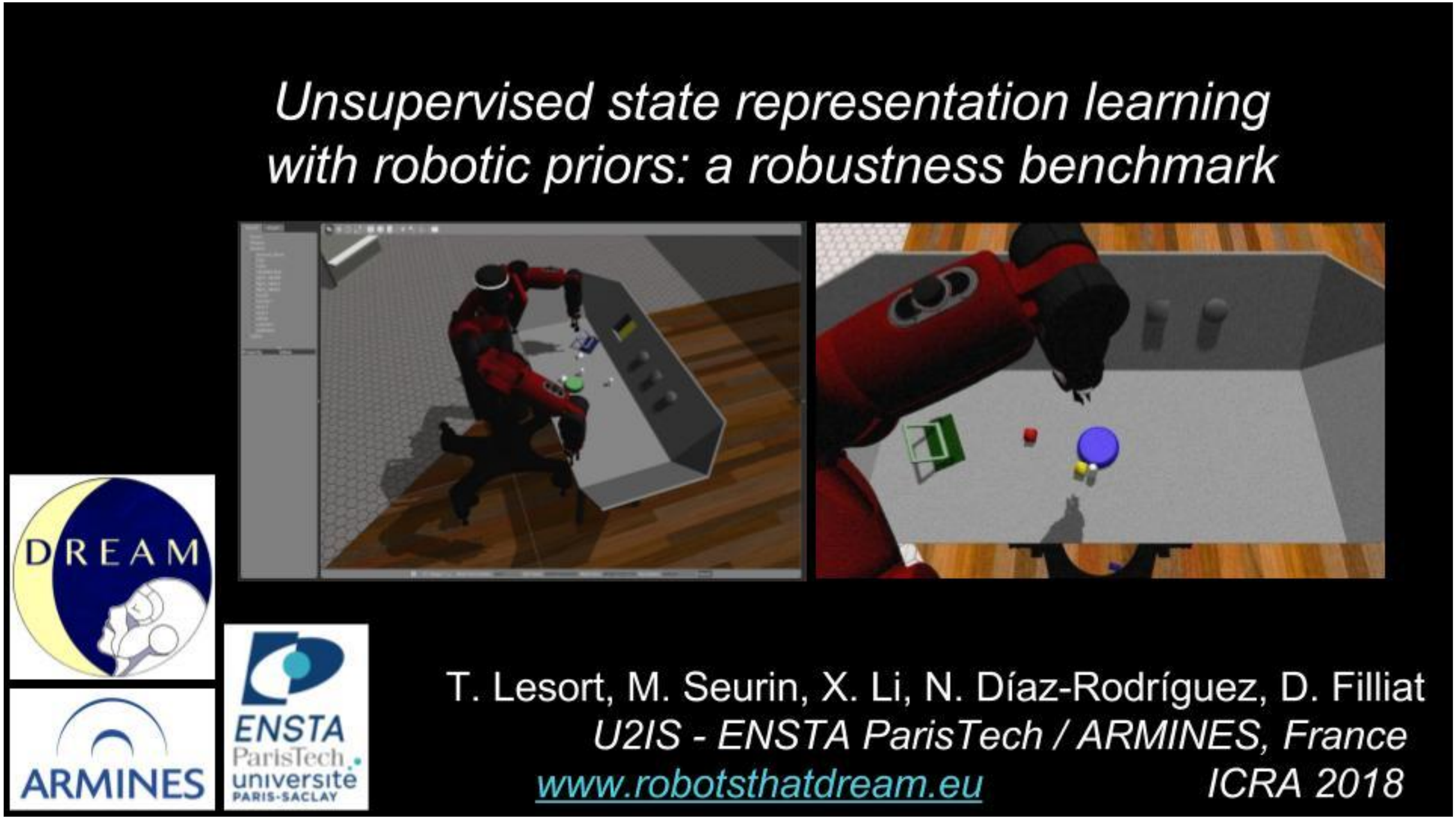
(or deep learning beyond classifying chihuahuas and blueberry muffins)



Appendix

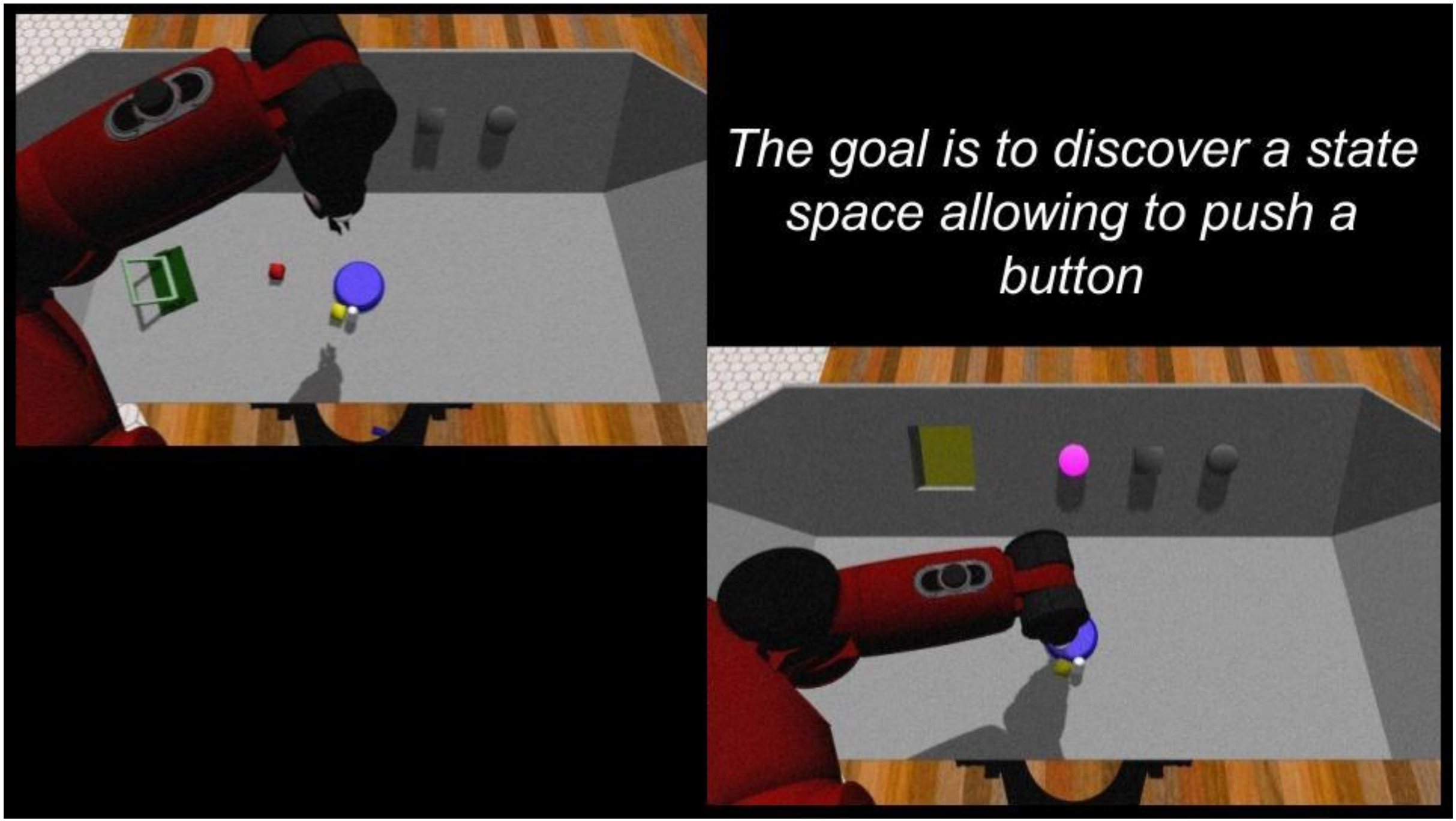
Unsupervised state representation learning with robotic priors: a robustness benchmark

State Representation Learning with robotics priors



T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4

State Representation Learning with robotics priors



T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4

State Representation Learning with robotics priors

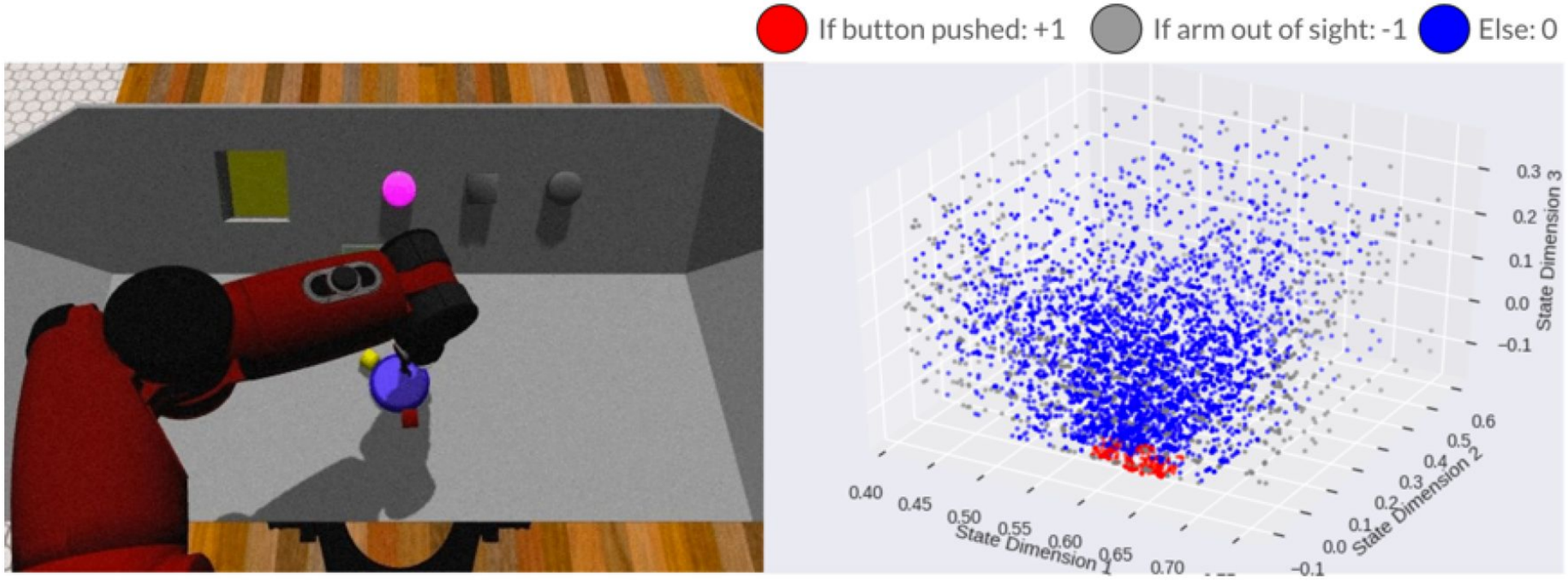
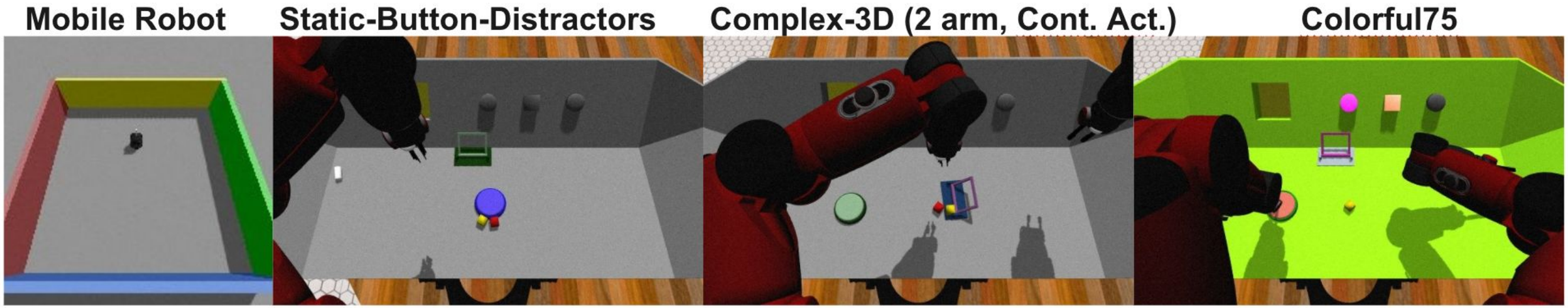


Fig. 2. Left: Baxter's camera view for Static-Button-Distractors dataset 2. Right: Baxter's left hand position ground truth position and its coded reward



T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4

State Representation Learning with robotics priors

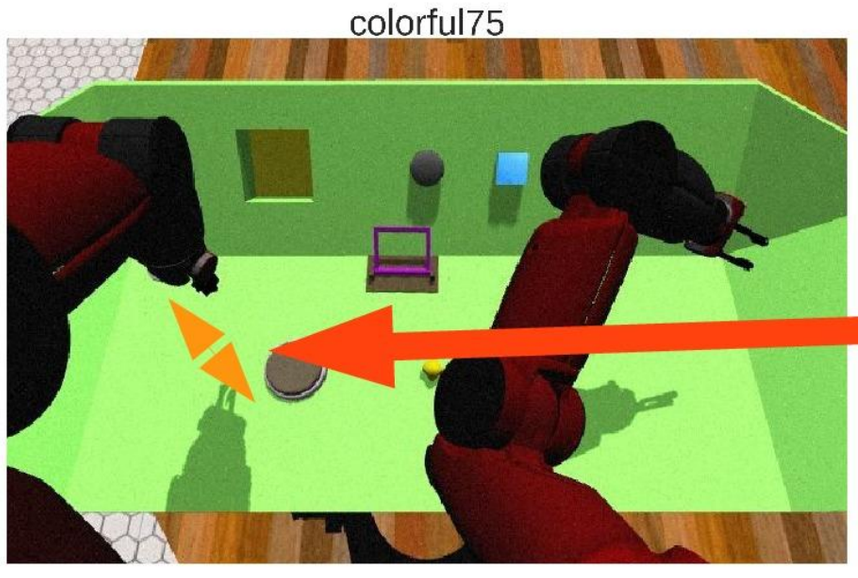
Figure 4: Nearest neighbor evaluation of robotic priors against supervised learning and Autoencoders on (the domain randomization) *Colorful75* dataset using the fixed ref. point prior. The auto-encoder-learnt nearest state's observation is far from the ground truth position.

Criterion	GT	Superv	4 Priors	5 Priors	AE	5 Priors 0f
KNN-MSE	0.024	0.03	0.079	0.053	0.099	0.047
NIEQA local	0	0.239	0.66	0.50	0.599	0.52
NIEQA global	0	0.048	0.41	0.20	0.465	0.21

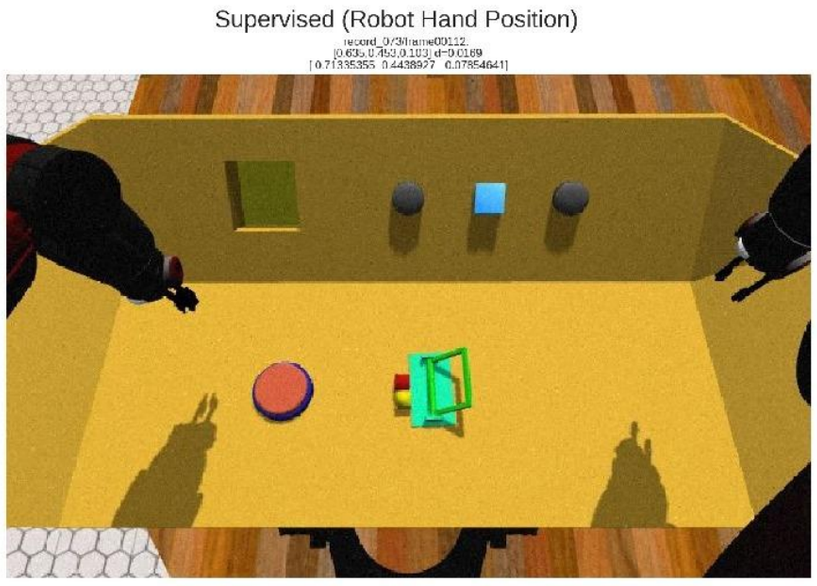
Table 1: *Static-Button-Distractor 3D* (dataset 2) Results. *x*f means x ResNet frozen layers, GT: (hand position) Ground Truth

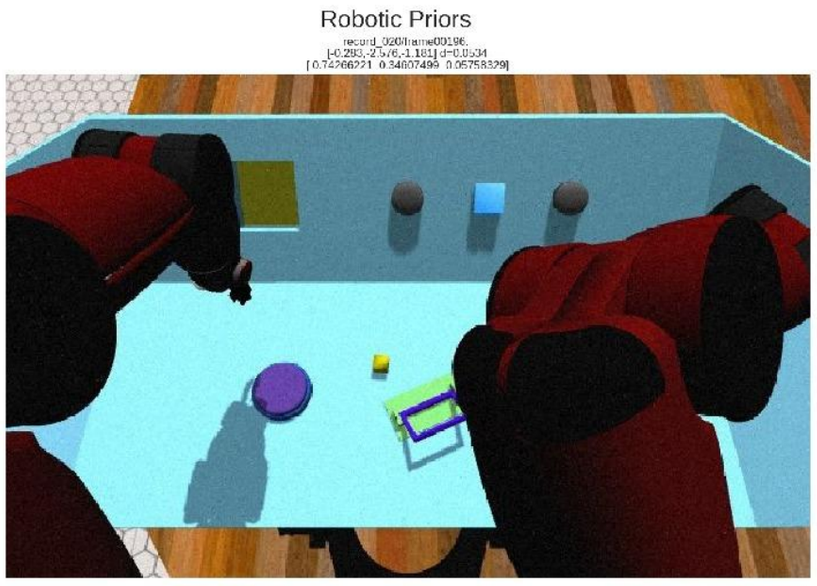
T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4

State Representation Learning with robotics priors



while autoencoders fail completely





T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4

State Representation Learning with robotics priors

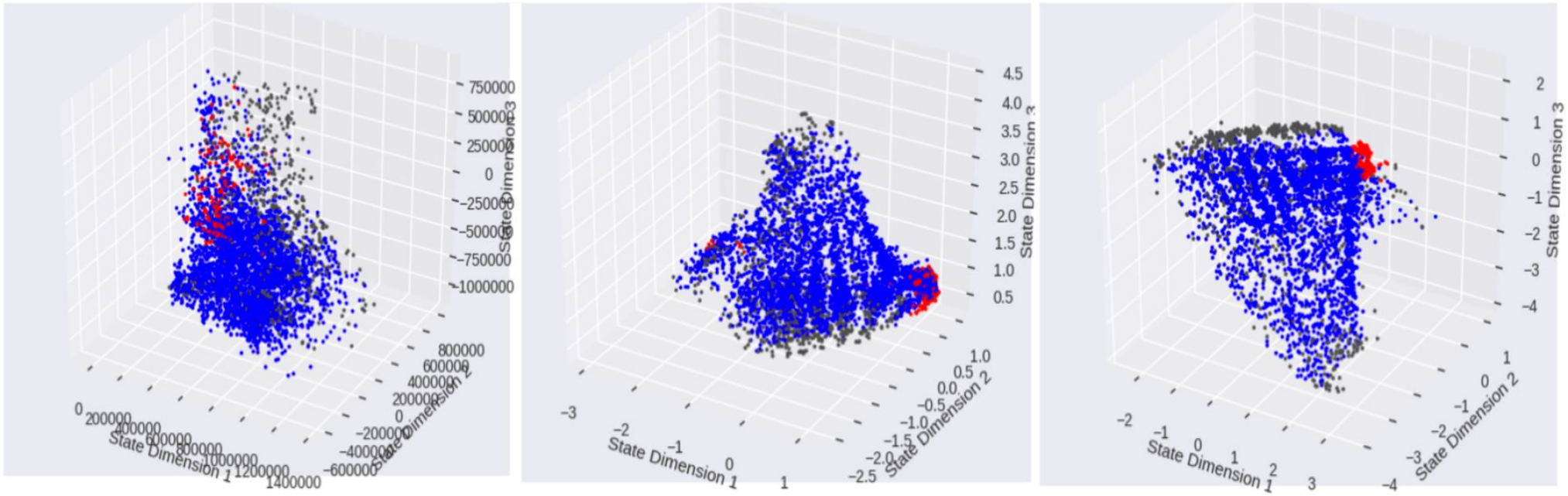


Fig. 4. Learned state space on Static-Button-Distractors (dataset 2): Left: Denoising Autoencoder. Middle: 4 Priors. Right: 5 Priors. A red reward (value +1) state means the button is being pushed, gray (-1) if the hand is out of sight, and blue (reward 0) if hand is elsewhere.

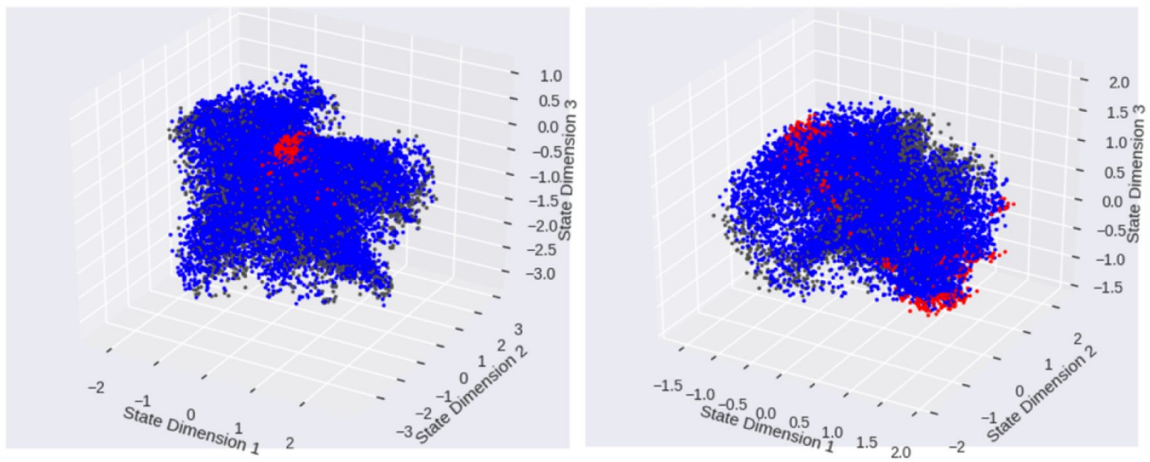


Fig. 6. Results of the 5 priors on *colorful75* (dataset 4). Left: 5 priors using button position as ref. point. Right: 5 priors using starting hand position as ref. point.

T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4

Robotic Priors

[Jonschkowski et. al. 2015]

Effect of distractors and domain randomization without Ref. point prior

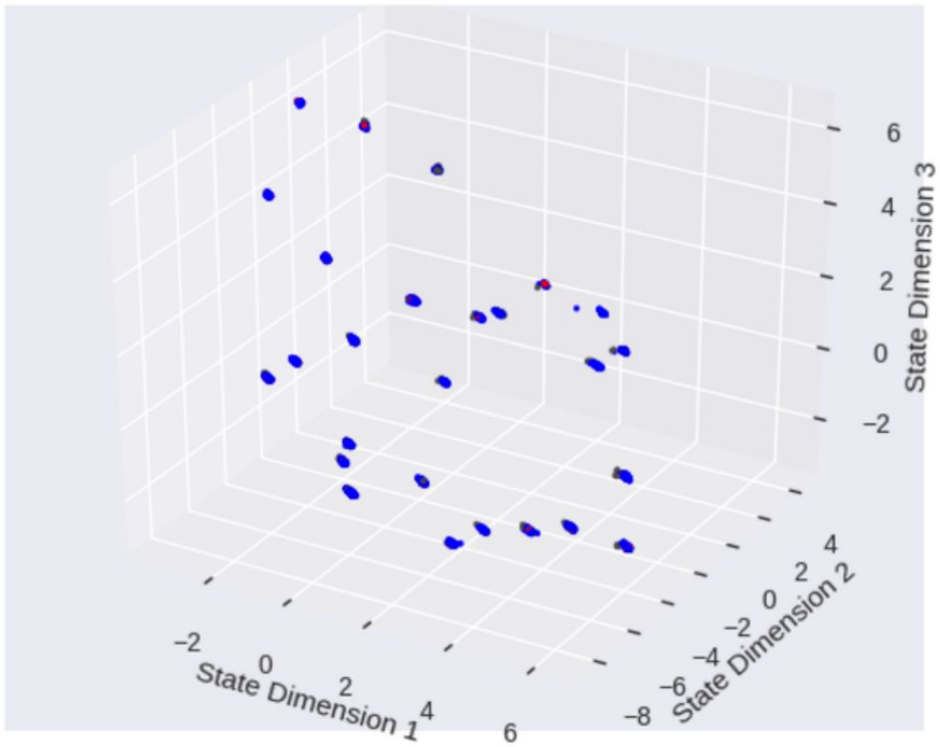
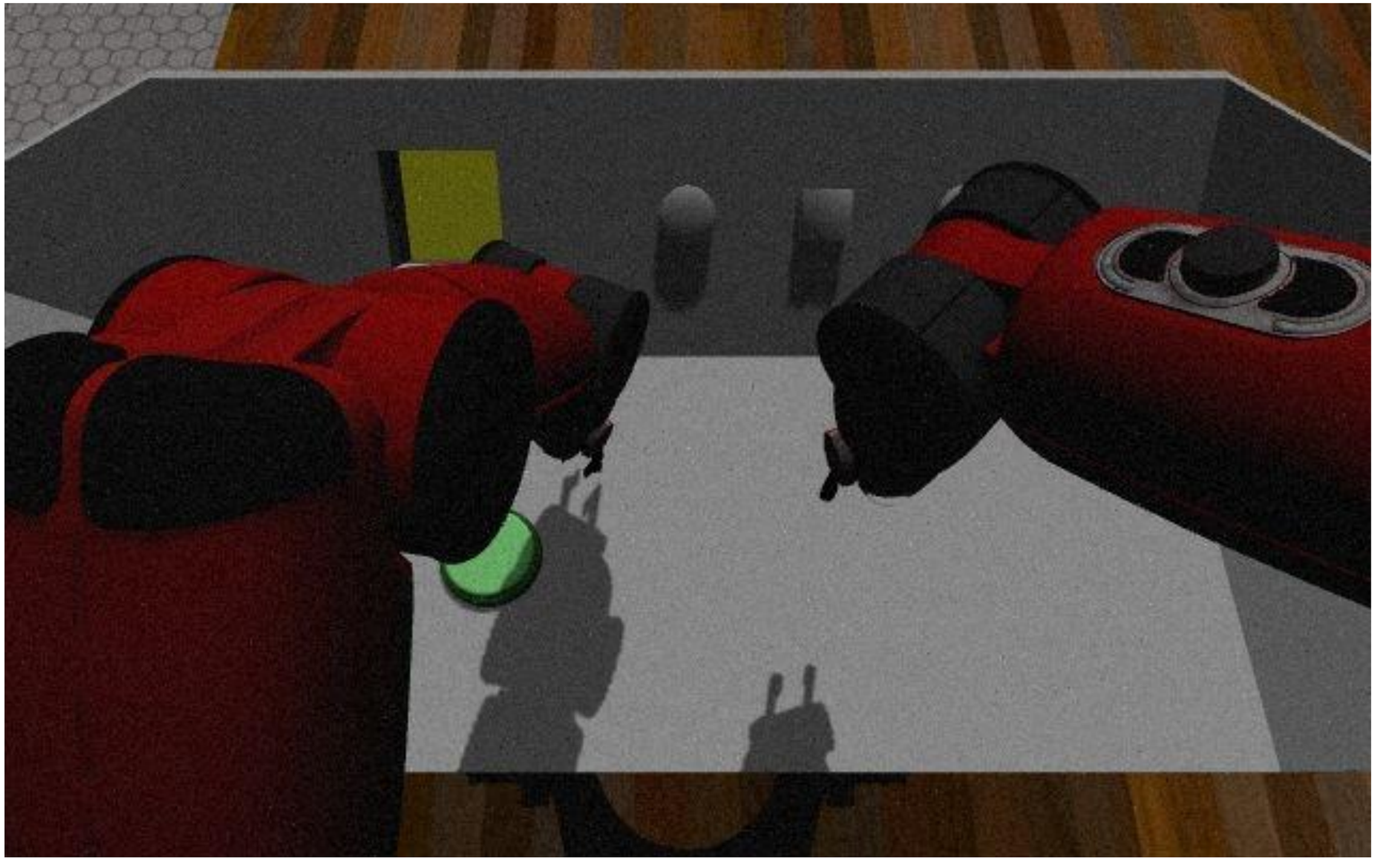


Fig. 5. Effect of static distractors in dataset 3 on the 4 priors approach learned state space.

T. Lesort et. al. Unsupervised state representation learning with robotic priors: a robustness benchmark, 2017, https://arxiv.org/abs/1709.05185 DEMO: https://www.youtube.com/watch?v=wFd0yJuJlQ4



CONTRIBUTIONS:

- *Robotic Priors ignore distractors and learn representations more relevant to the task than autoencoders*
- *Performance is close to the one learnt with supervision*
- *A quantitative and qualitative evaluation metric based on KNN-MSE*
- *Vulnerabilities of 4 original robotic priors are overcome with a contributed 5th fixed reference point prior*