Non-supervised Learning & Decision Making

Danilo J. Rezende

DeepMind

Hammers & Nails 2019, Weizmann Institute of Science, Israel

"The revolution will not be supervised" (Yann Lecun, 2017)

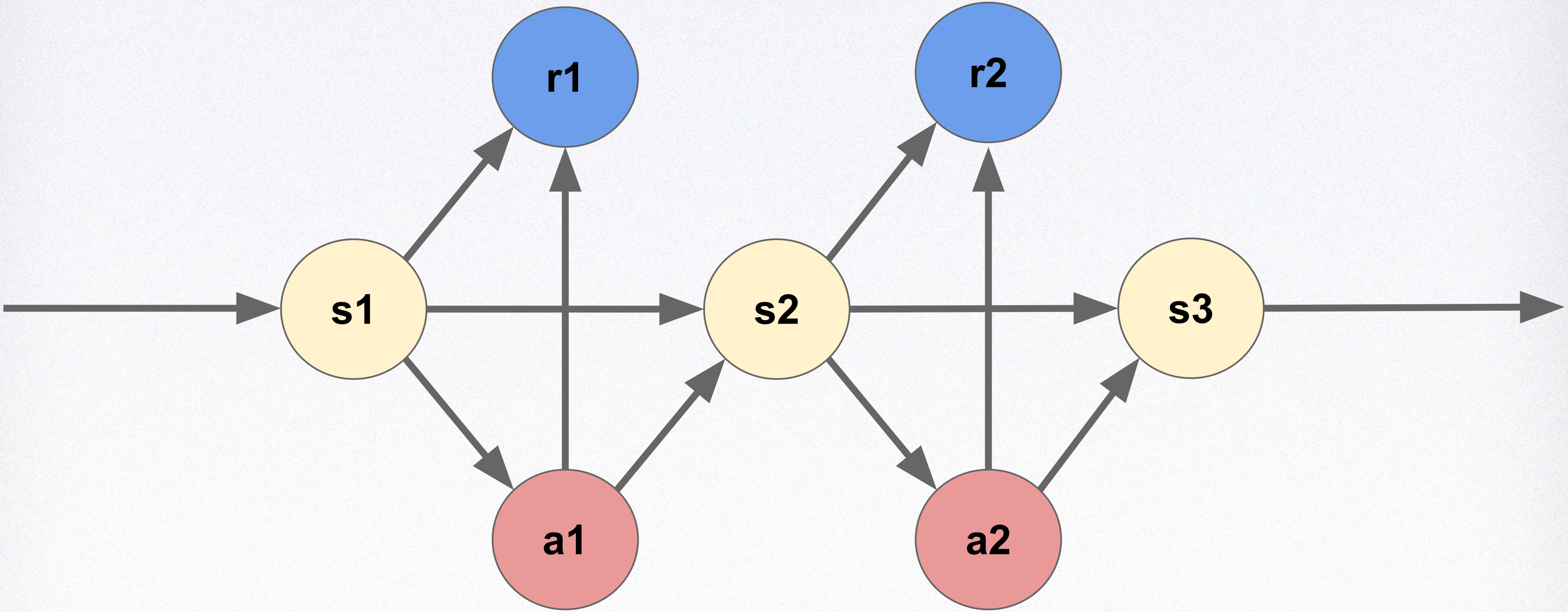
Topics

- General discussion about Reinforcement Learning
- Why build models?
- Using generative models for representations learning:
  - GQN
  - SimCore
- Some thoughts on applications of group theory in ML

Reinforcement Learning

DeepMind

Let's consider a discrete Markov Decision Process (MDP)



RL in a discrete world

Set of States	$\mathcal{S} = \{1, \ldots, N_s\}$
Set of Rewards	$\mathcal{R} = \{1, \ldots, N_r\}$
Set of Actions	$\mathcal{A} = \{1, \ldots, N_a\}$
Reward function	$r : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$
Model/Environment	$m : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$

Let's consider a discrete world MDP

$$Q(s_0, a_0) = \max_{\text{all possible paths}} \sum_{t=1}^{H} \gamma^t r(s_t, a_t)$$

Optimal decision $$a^\star(s_0) = \mathrm{argmax}_a Q(s_0, a)$$

Let's consider a discrete world MDP

$$Q(s_0, a_0) = \max_{\text{all possible paths}} \sum_{t=1}^{H} \gamma^t r(s_t, a_t)$$

Set of States	$\mathcal{S} = \{1, \ldots, N_s\}$
Set of Rewards	$\mathcal{R} = \{1, \ldots, N_r\}$
Set of Actions	$\mathcal{A} = \{1, \ldots, N_a\}$
Reward function	$r : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$
Model/Environment	$m : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$
Q-function	$Q : \mathcal{S} \times \mathcal{A} \to \mathcal{R}^H$

How much data do we need to learn r, m and Q?

$$\text{Amount of Data to learn } f \approx e^{\text{Description Length of f}}$$

$$\begin{aligned}
\mathrm{DL}(r) &= N_s N_a \log_2(N_r) \\
\mathrm{DL}(m) &= N_s N_a \log_2(N_s) \\
\mathrm{DL}(Q) &= N_s N_a H \log_2(N_r)
\end{aligned}$$

When is it worth learning a model?

$$\mathrm{DL}(r+m) = N_s N_a \log_2(N_r) + N_s N_a \log_2(N_s)$$

$$\mathrm{DL}(Q) = N_s N_a H \log_2(N_r)$$

Hypothesis: It will be favorable to learn a model when

When is it worth learning a model?

$$\mathrm{DL}(r+m) = N_s N_a \log_2(N_r) + N_s N_a \log_2(N_s)$$

$$\mathrm{DL}(Q) = N_s N_a H \log_2(N_r)$$

Hypothesis: It will be favorable to learn a model when

$$\mathrm{DL}(Q) > \mathrm{DL}(r+m)$$

$$(H-1)\log_2(N_r) > \log_2(N_s)$$

When is it worth learning a model?

$$(H-1)\log_2(N_r) > \log_2(N_s)$$

In general, we operate in a regime where Ns >> Nr, H. So it seems that we would always prefer to learn Q from scratch rather than learning a model first.

When is it worth learning a model?
Many tasks, a single world

$$L(H-1)\log_2(N_r) > \log_2(N_s)$$

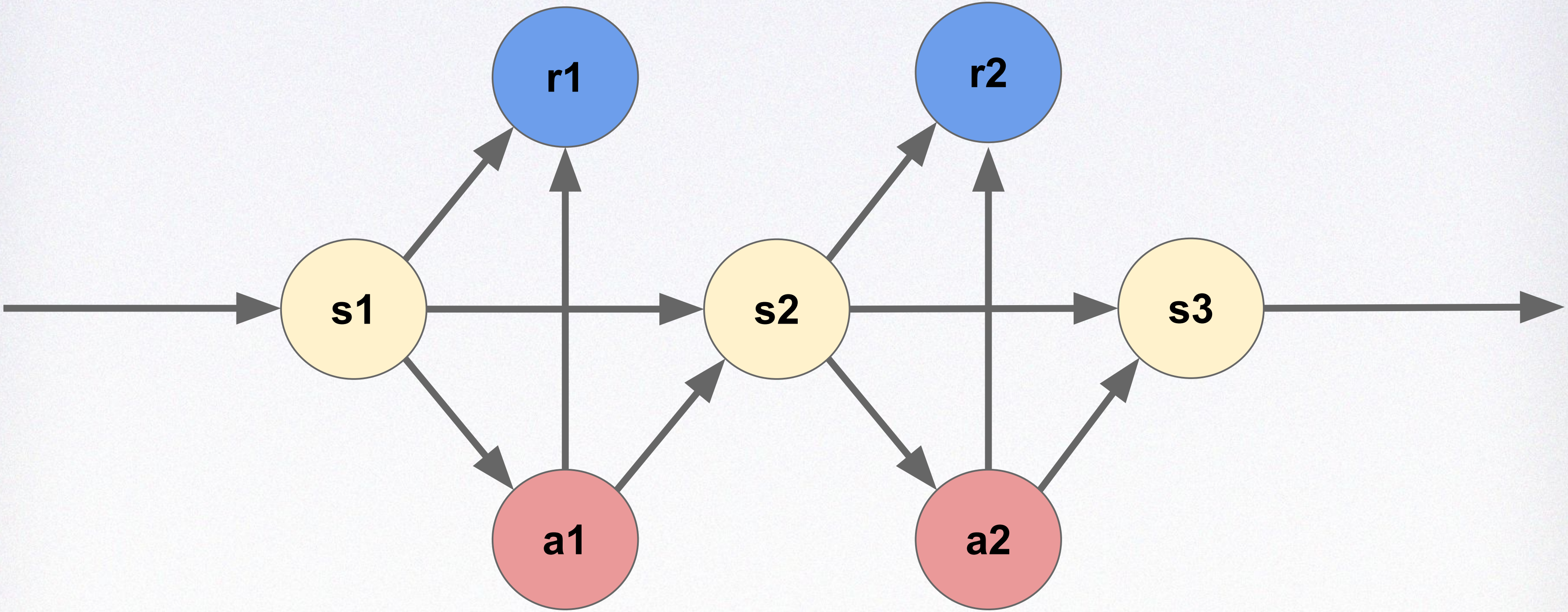
For a sufficiently large number of different tasks L, it will require less data if we learn a model first.

Summary

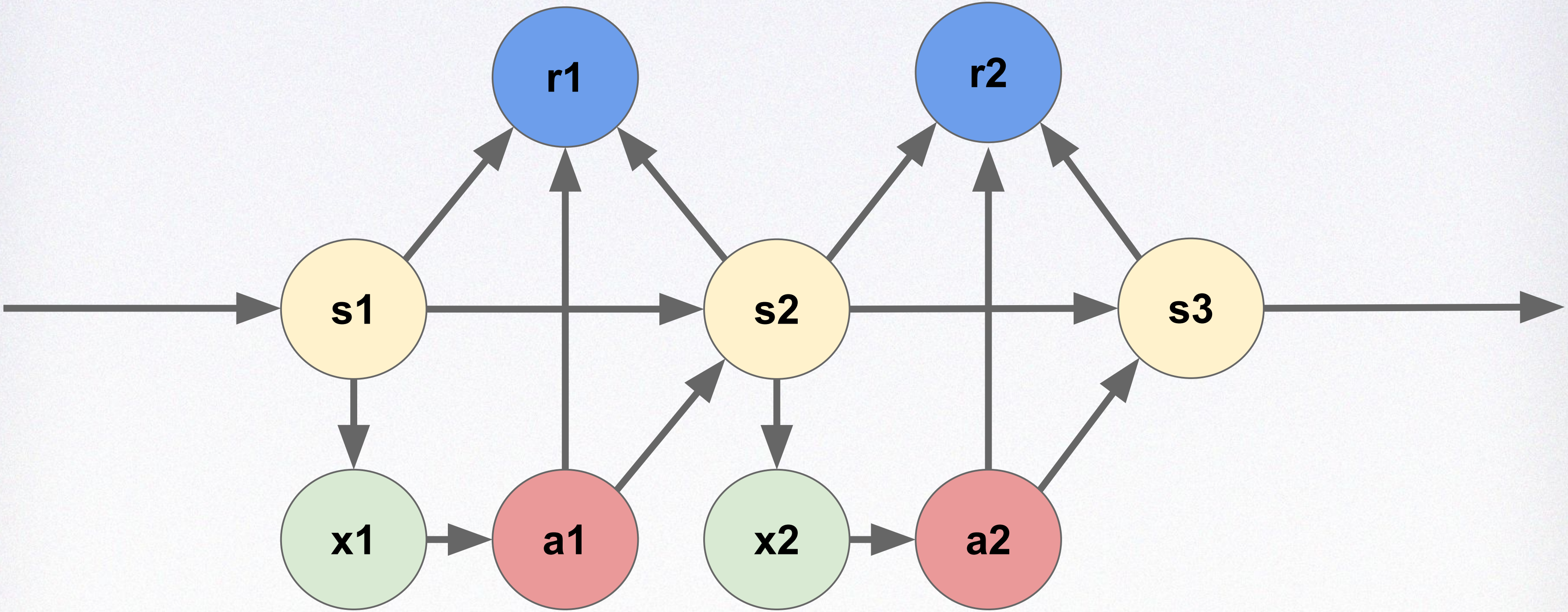
- Learning a model first and using that to compute Q-functions via Monte-Carlo can be more data-efficient in some cases, specially in multi-task problems

- Models offer other advantages in addition to planning:
  - Notion of uncertainty or novelty
  - Unsupervised learning of useful features
  - Unsupervised learning to use complex memory architectures

RL in a stochastic and partially observed world



RL in a stochastic and partially observed world



RL in a stochastic world

Policy $\pi(a_t | x_{0,\ldots,t}, a_{0,\ldots,(t-1)})$

Model/Simulator $m(x_t | x_{0,\ldots,(t-1)}, a_{0,\ldots,(t)})$

Path/trajectory $\tau = \{(x_1, a_1), \ldots, (x_t, a_t), \ldots (x_H, a_H)\}$

$$Q^\pi(s_0, a_0) = \mathbb{E}_{p(\tau)}\left[\sum_t \gamma^t r(s_t, a_t) \middle| s_0, \mathrm{do}(a_0)\right]$$

RL in a stochastic world

Policy $\pi(a_t | x_{0,\ldots,t}, a_{0,\ldots,(t-1)})$

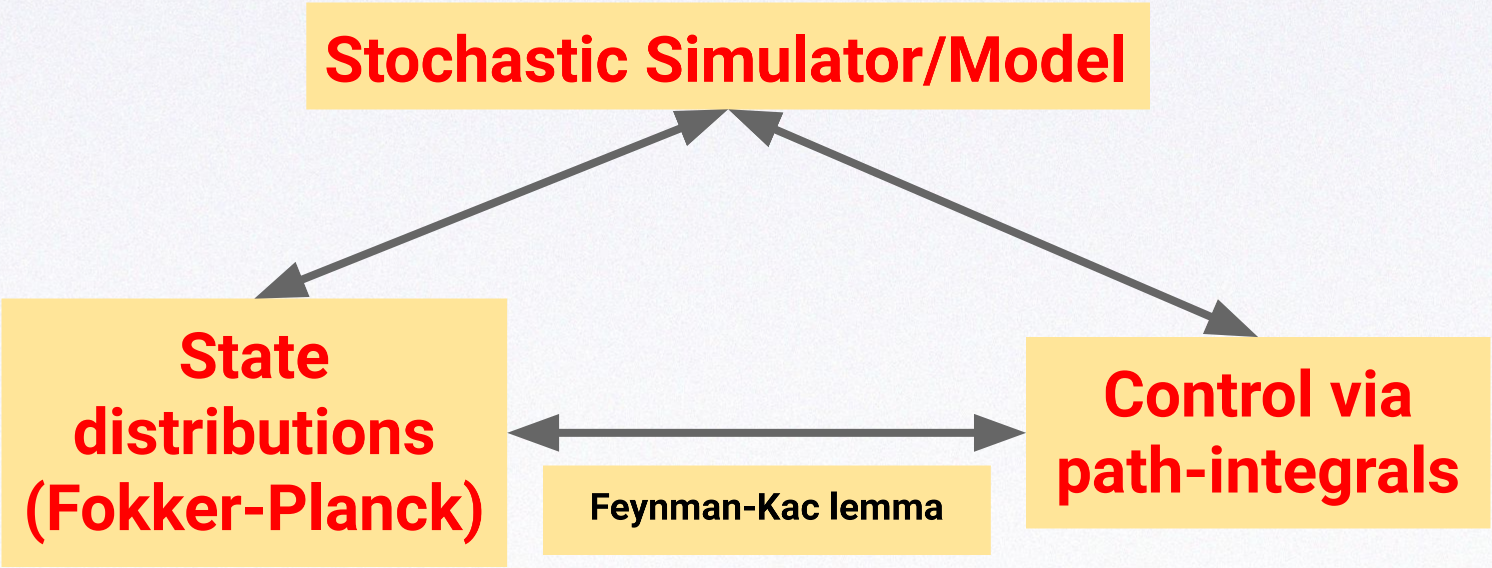
Model/Simulator $m(x_t | x_{0,\ldots,(t-1)}, a_{0,\ldots,(t)})$

Path/trajectory $\tau = \{(x_1, a_1), \ldots, (x_t, a_t), \ldots (x_H, a_H)\}$

$$Q^\pi(s_0, a_0) = \mathbb{E}_{p(\tau)}\left[\sum_t \gamma^t r(s_t, a_t) \middle| s_0, \mathrm{do}(a_0) \right]$$

Path Integral

RL in a stochastic world



Path Integral Formulation of Stochastic Optimal Control with Generalized Costs Yang et al
A Generalized Path Integral Control Approach to Reinforcement Learning Theodorou et al

RL in a stochastic world

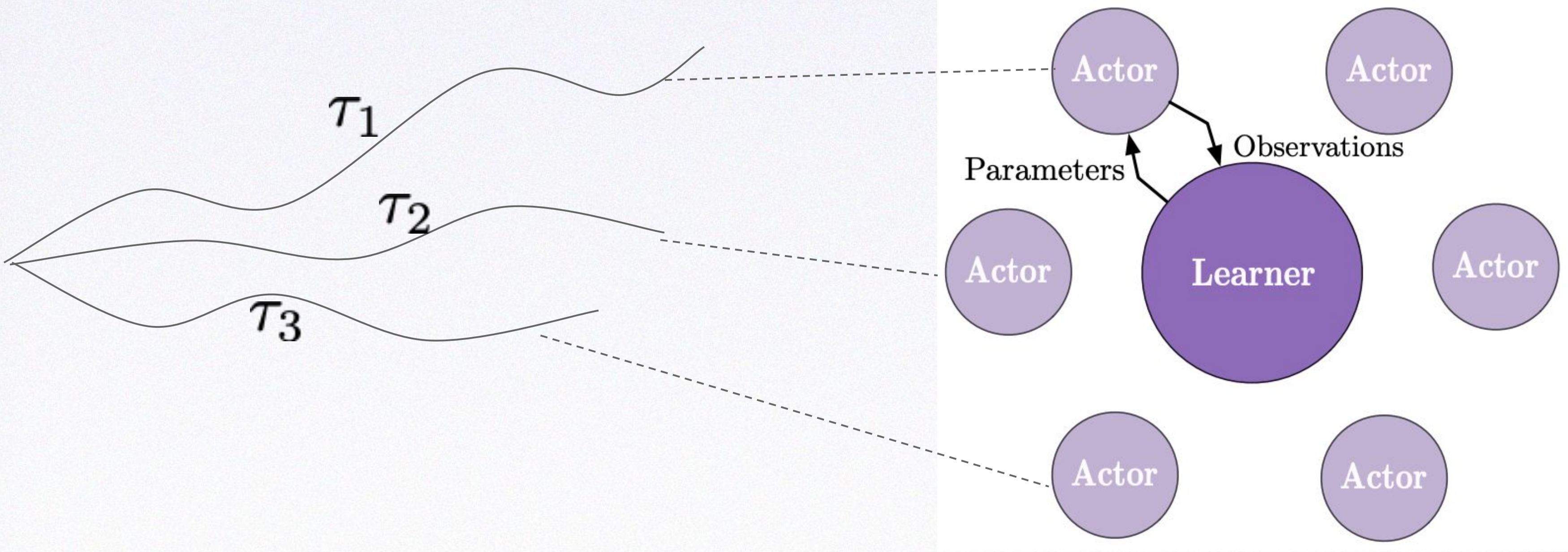
$$Q^\pi(s_0, a_0) = \mathbb{E}_{p(\tau)}\left[\sum_t \gamma^t r(s_t, a_t) \middle| s_0, \mathrm{do}(a_0)\right]$$

RL in a stochastic world

$$Q^\pi(s_0, a_0) = \mathbb{E}_{p(\tau)}\left[\sum_t \gamma^t r(s_t, a_t) \middle| s_0, \mathrm{do}(a_0)\right]$$

$$Q^\pi(s_0, a_0) \approx \frac{1}{N}\sum_k \sum_t \gamma^t r(s_t^k, a_t^k)$$

RL in a complex world



[IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures]
Lasse Espeholt, Hubert Soyer

RL in a stochastic world

$$Q^\pi(s_0, a_0) = \mathbb{E}_{p(\tau)}\left[\sum_t \gamma^t r(s_t, a_t) \middle| s_0, \mathrm{do}(a_0)\right]$$

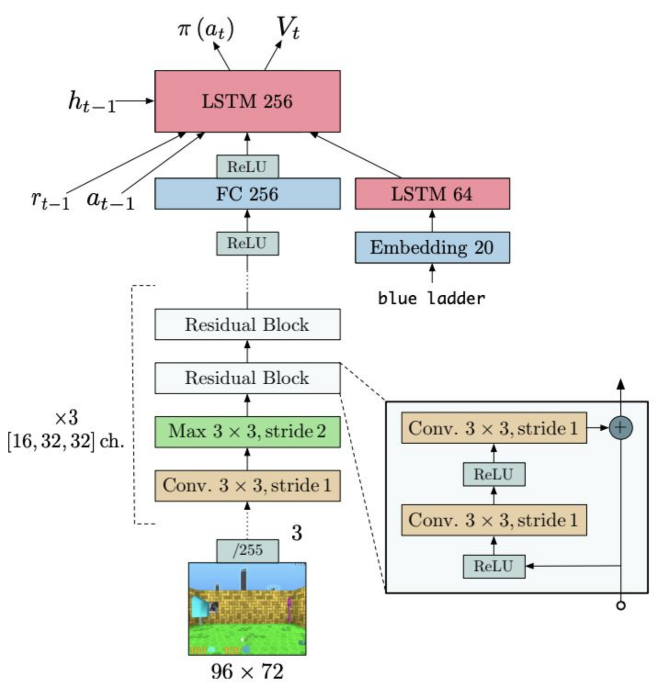
$$Q^\pi(s_0, a_0) \approx \frac{1}{N}\sum_k \sum_t \gamma^t r(s_t^k, a_t^k)$$

$$\boxed{Q^\pi(s_0, a_0) \approx \frac{1}{N}\sum_k \sum_t \gamma^t r(s_t^k, a_t^k)\frac{\pi_{\text{current}}(a_t^k)}{\pi_{\text{old}}(a_t^k)}}$$

[IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures]

Lasse Espeholt, Hubert Soyer

"Canonical" Agent



[IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures]()

Lasse Espeholt, Hubert Soyer

Models and Reinforcement Learning

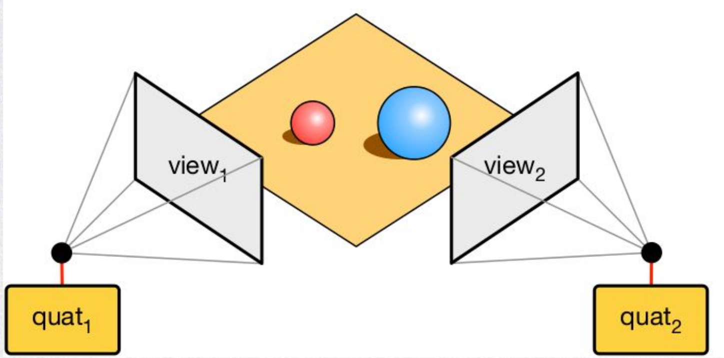
DeepMind

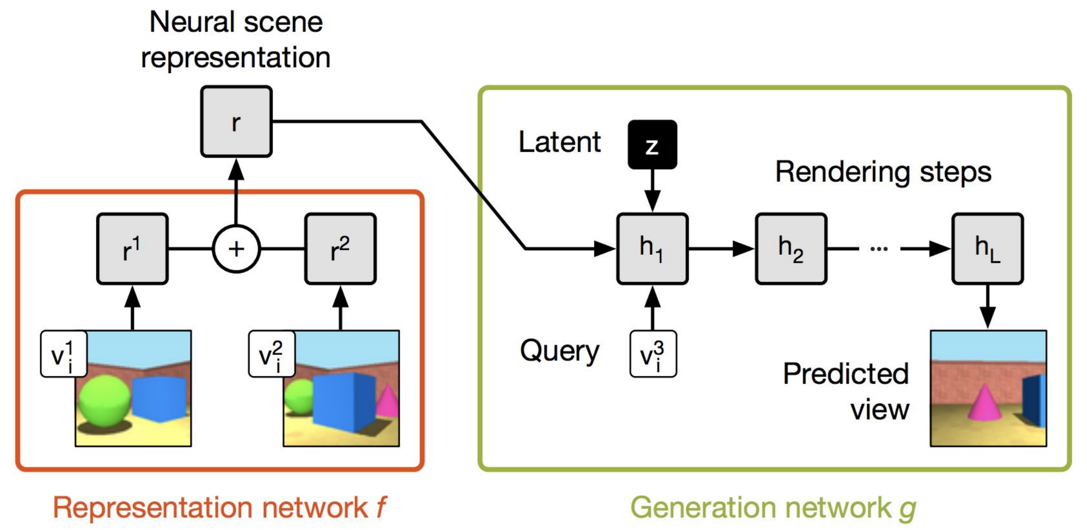
Neural Scene Representation and Rendering

S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018

DeepMind

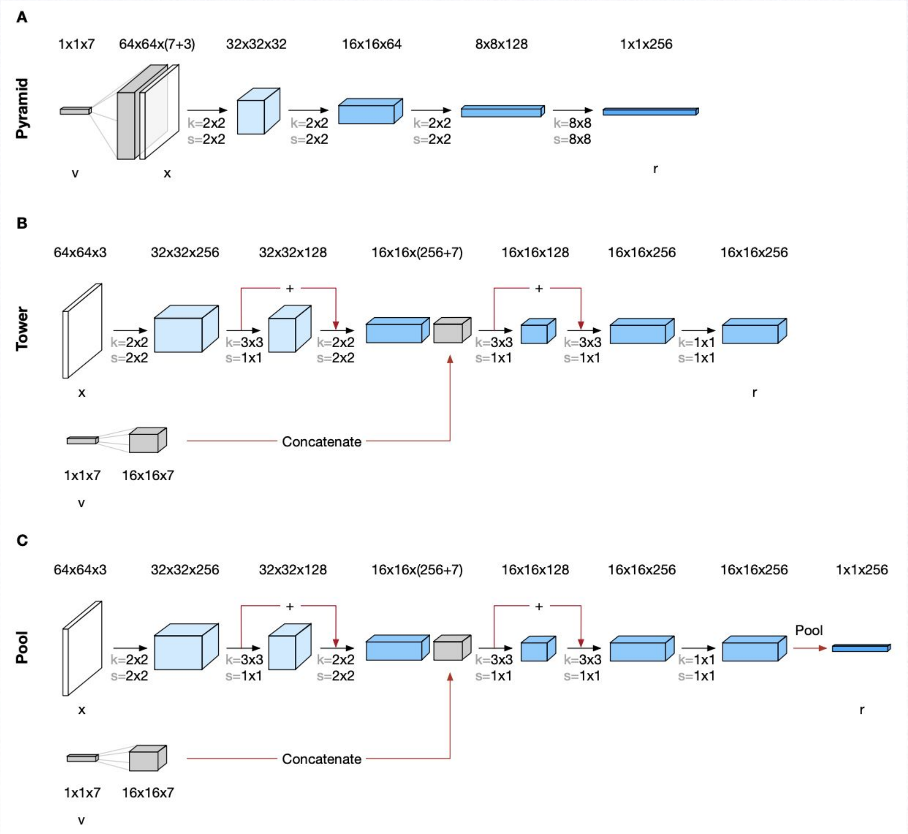
The Model

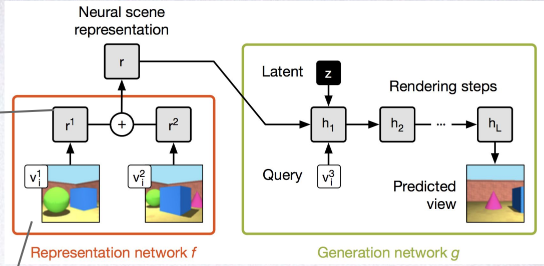




[“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.]

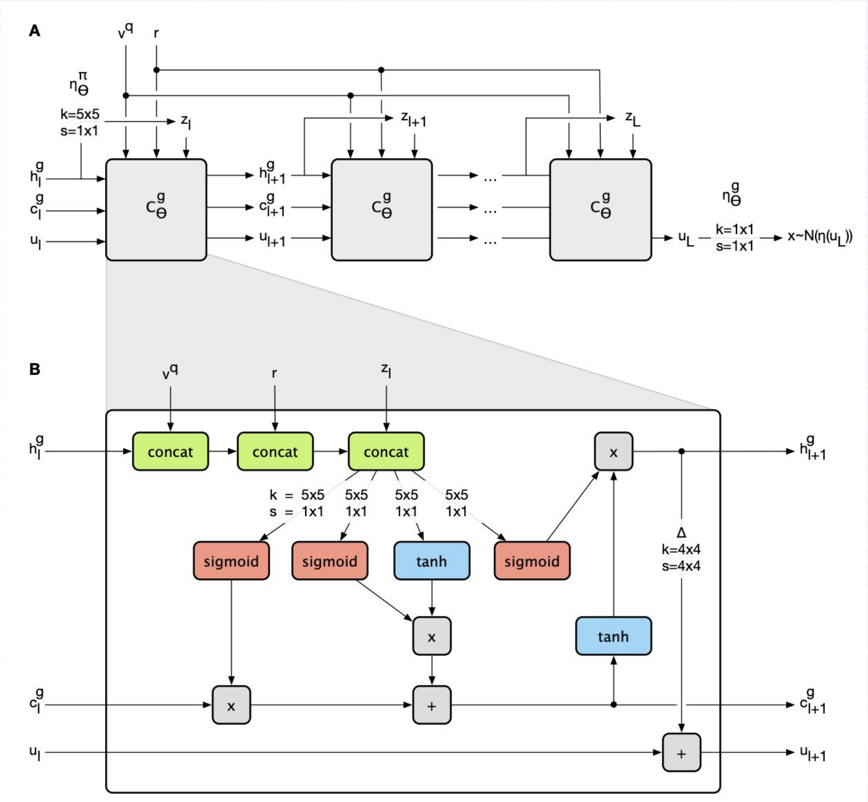
The Model

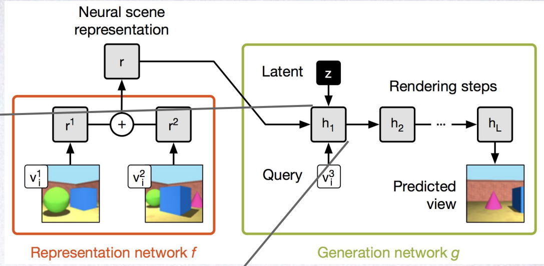




"Neural Scene Representation and Rendering". S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

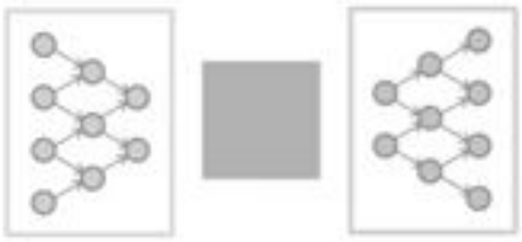
The Model





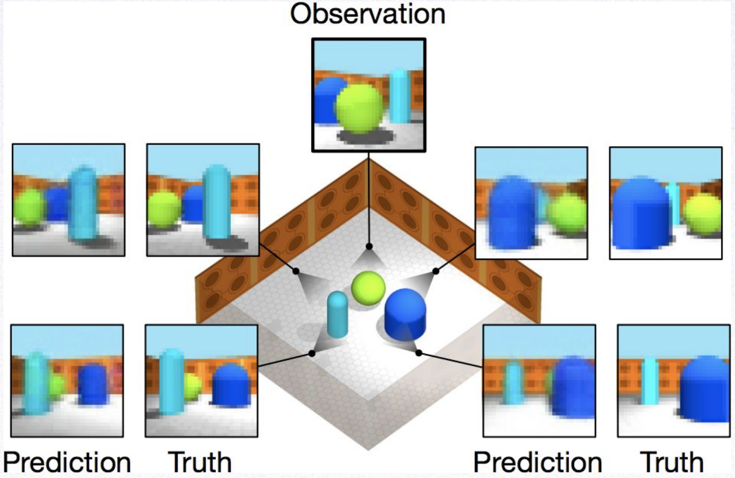
[“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.]

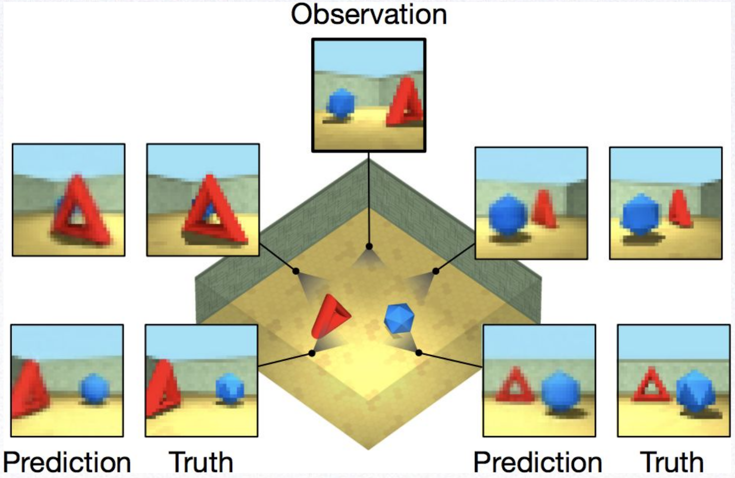
The Model



“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

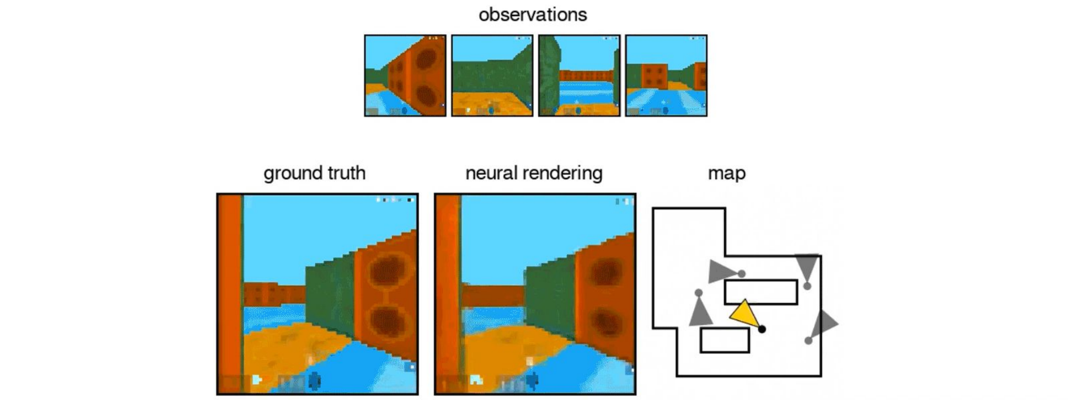
Near-Deterministic Predictions

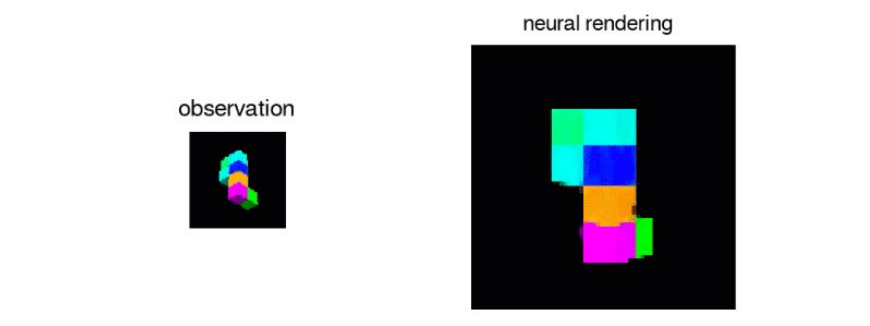


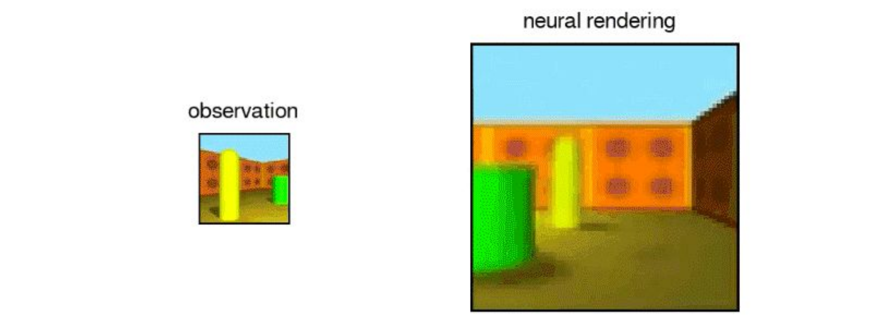


“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Data fusion and predicting with uncertainty

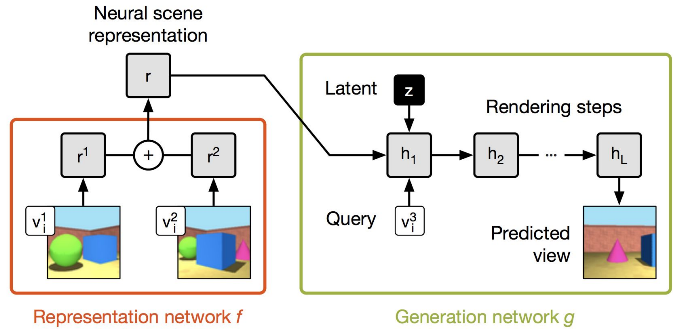


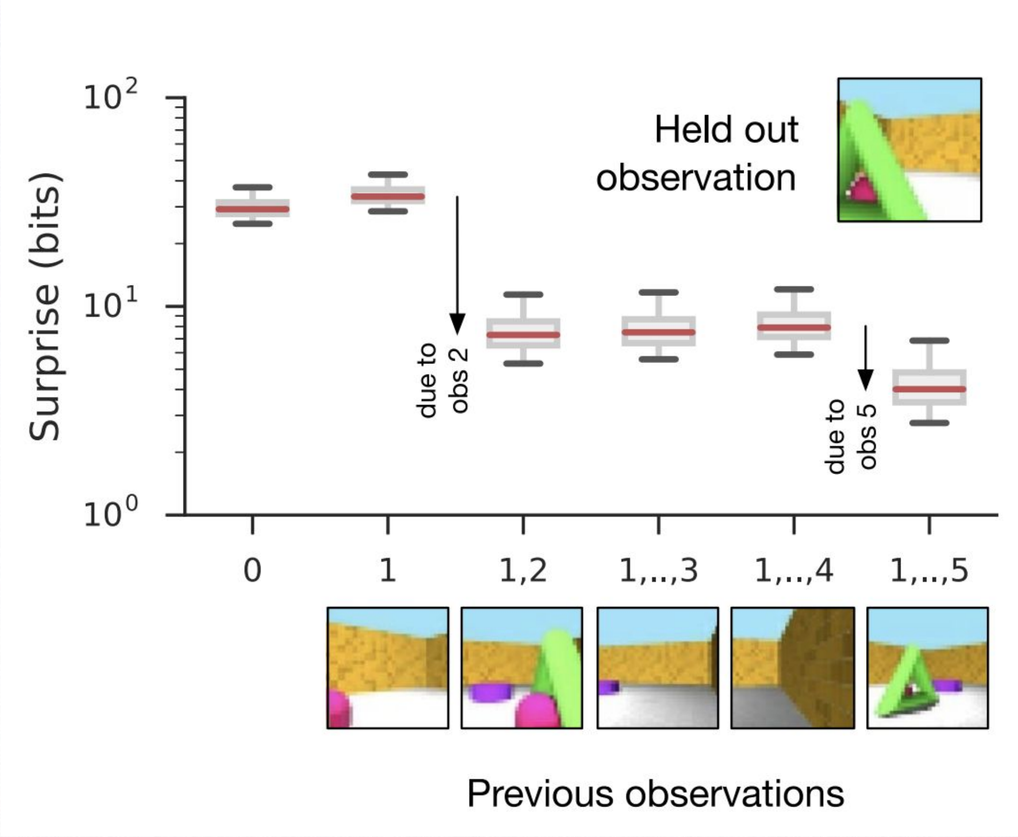




“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

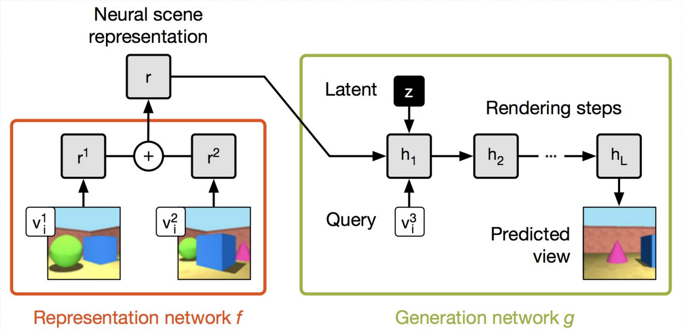
Quantifying uncertainty

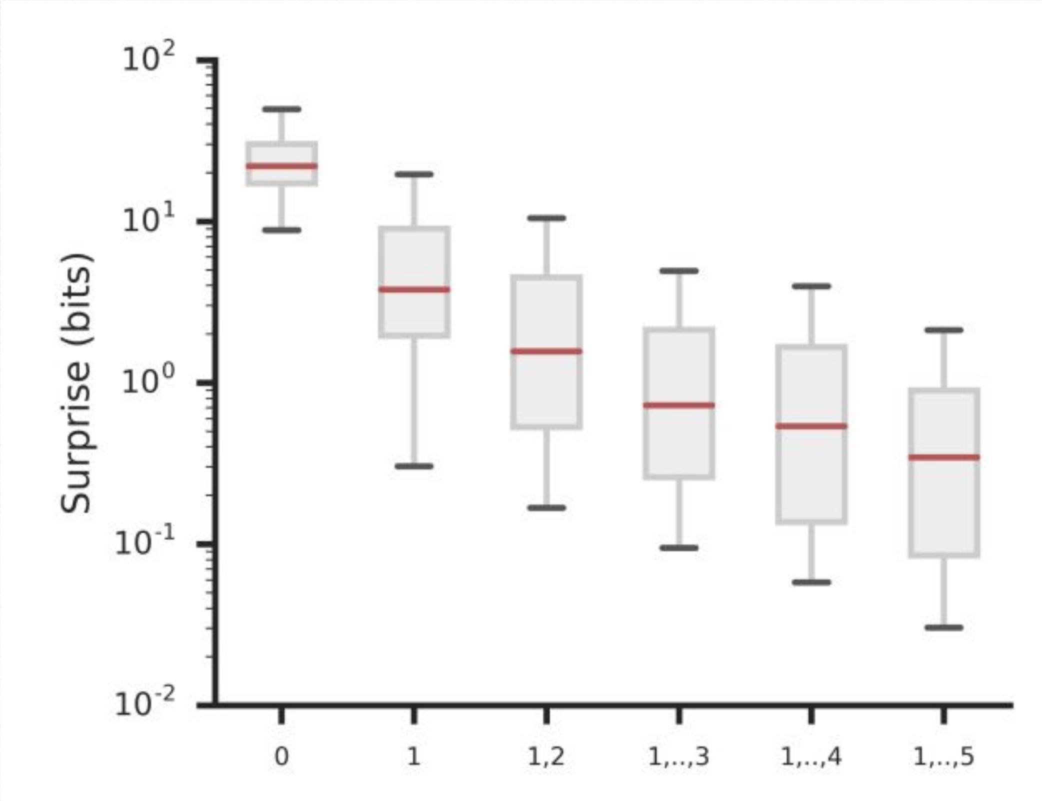




“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

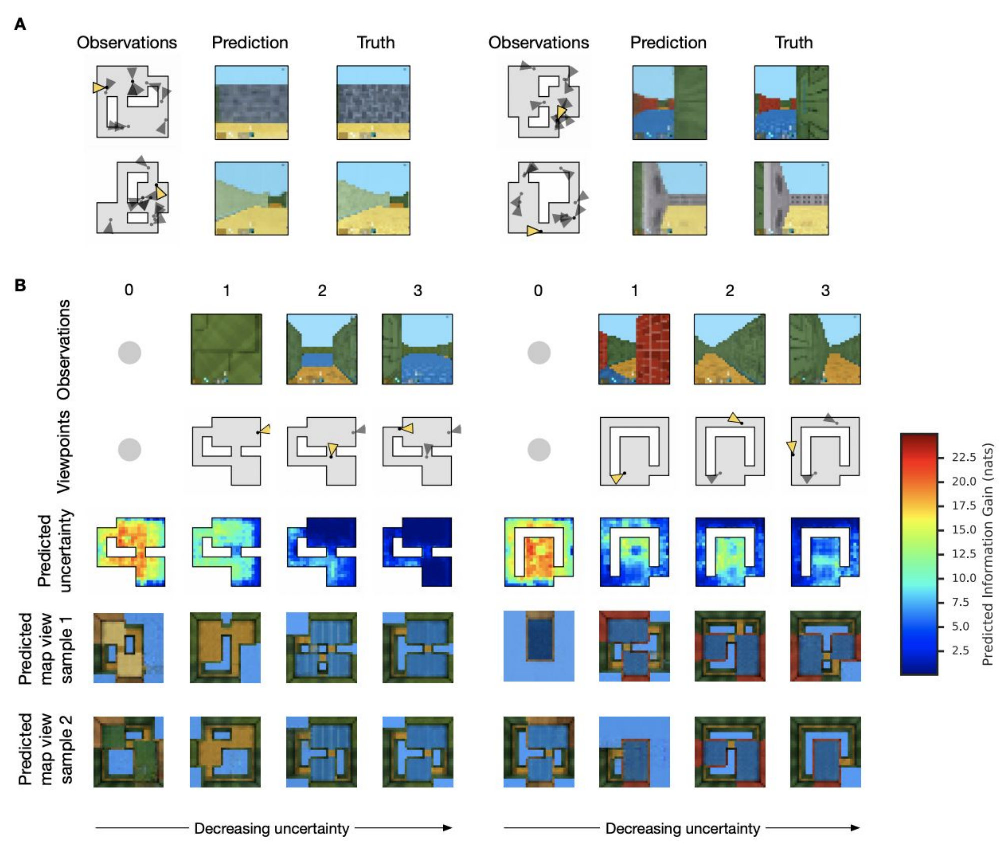
Quantifying uncertainty





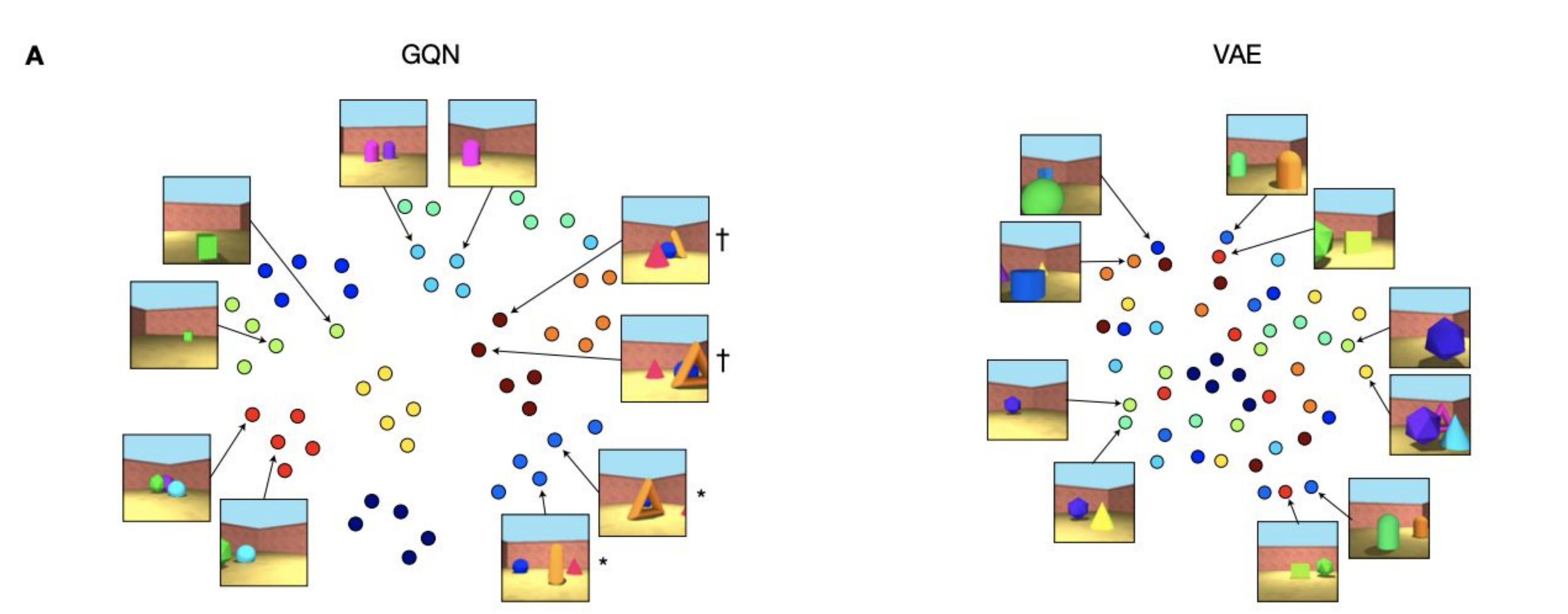
"Neural Scene Representation and Rendering". S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Quantifying uncertainty



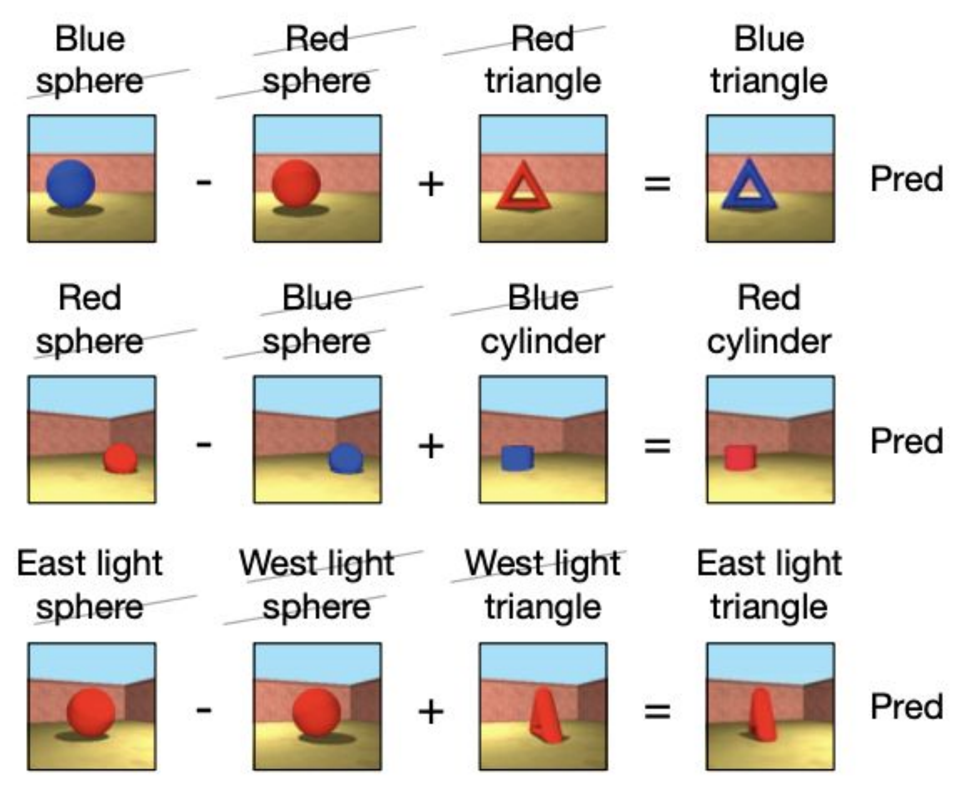
"Neural Scene Representation and Rendering". S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Viewpoint invariance of the learned scene representations



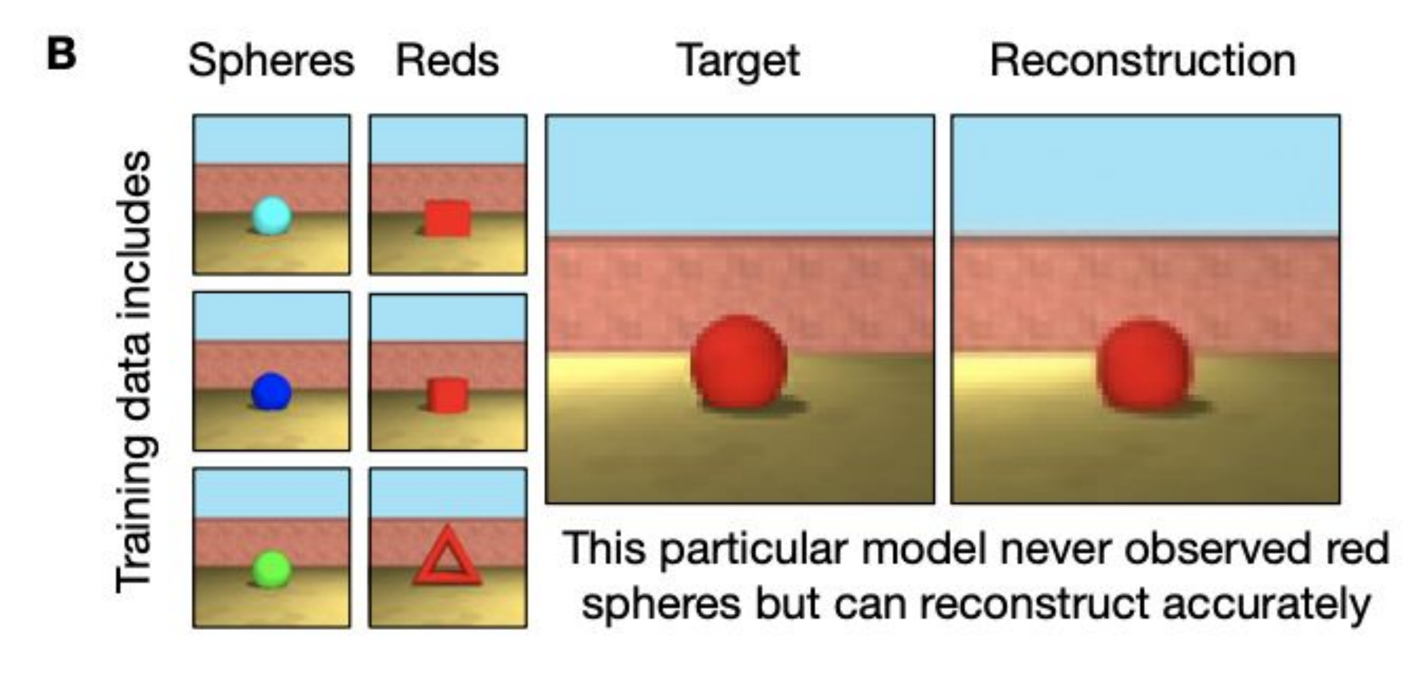
“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Factorization of the learned scene representations



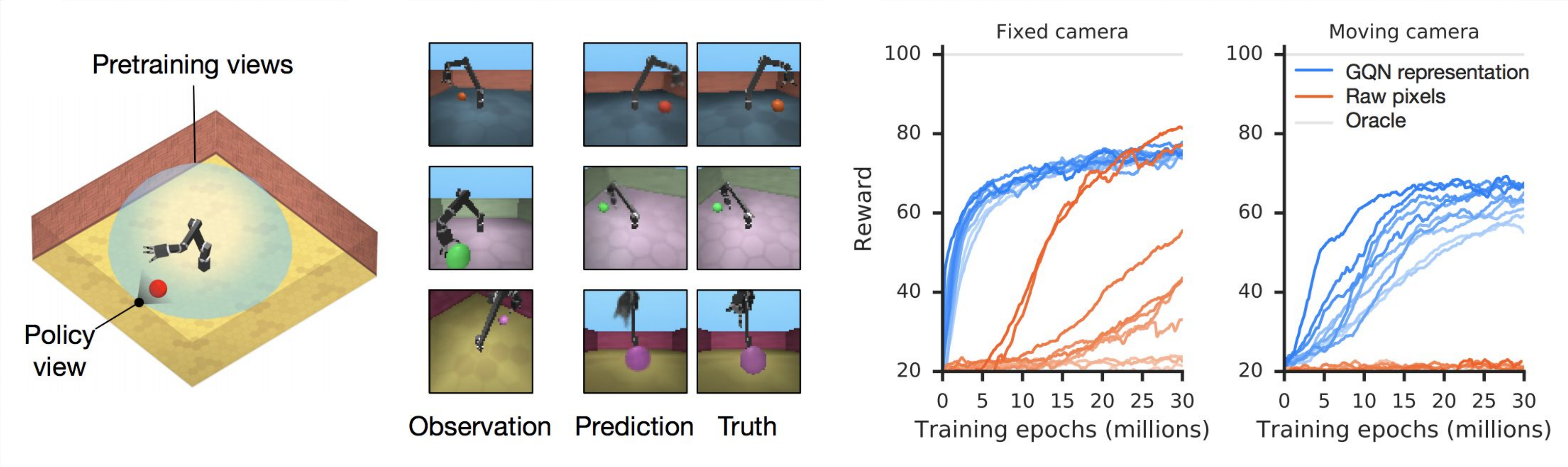
"Neural Scene Representation and Rendering". S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Factorization of the learned scene representations



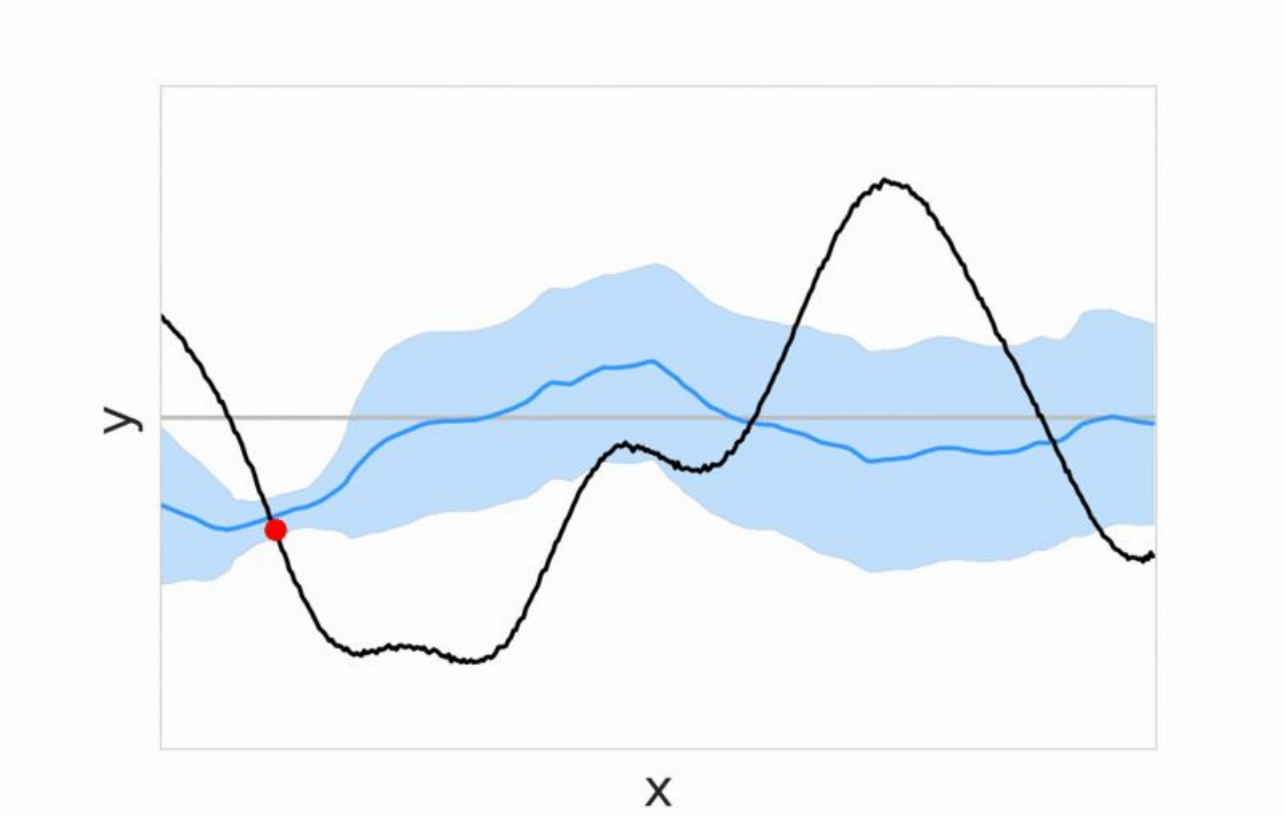
“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Learning useful representations for control



“Neural Scene Representation and Rendering”. S. M. Ali Eslami, Danilo J. Rezende, et al. Science, 2018.

Deep-Learning with Uncertainty



Slide credit: Marta Garnelo

“Conditional Neural Processes”. Marta Garnelo, Dan Rosenbaum, et al. ICML, 2018.

TL;DL

- GQN can learn factored scene representations
- It is also a kind of meta-learning model (learning to do one-shot scene inference)
- It doesn't have a notion of time
- It requires knowledge about "camera locations"

Shaping Belief States with Generative Environment Models for RL

Karol Gregor, Danilo Rezende, Frederic Besse,
Yan Wu, Hamza Merzic, Aaron van den Oord,
https://arxiv.org/abs/1906.09237

DeepMind

Example (actual result)

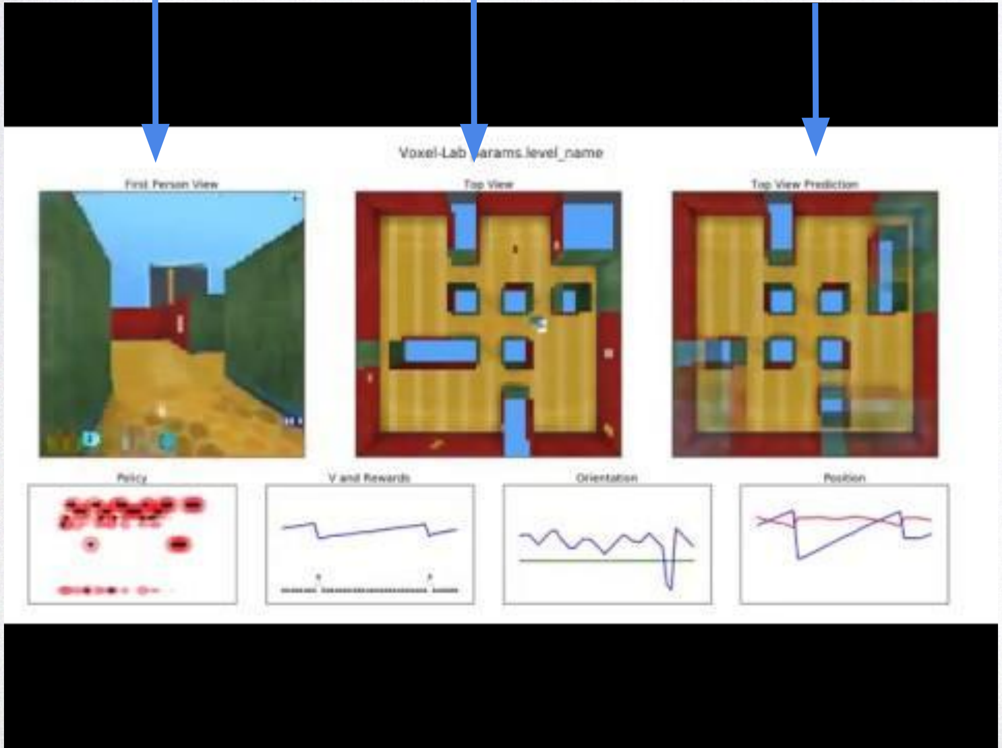
What agent sees

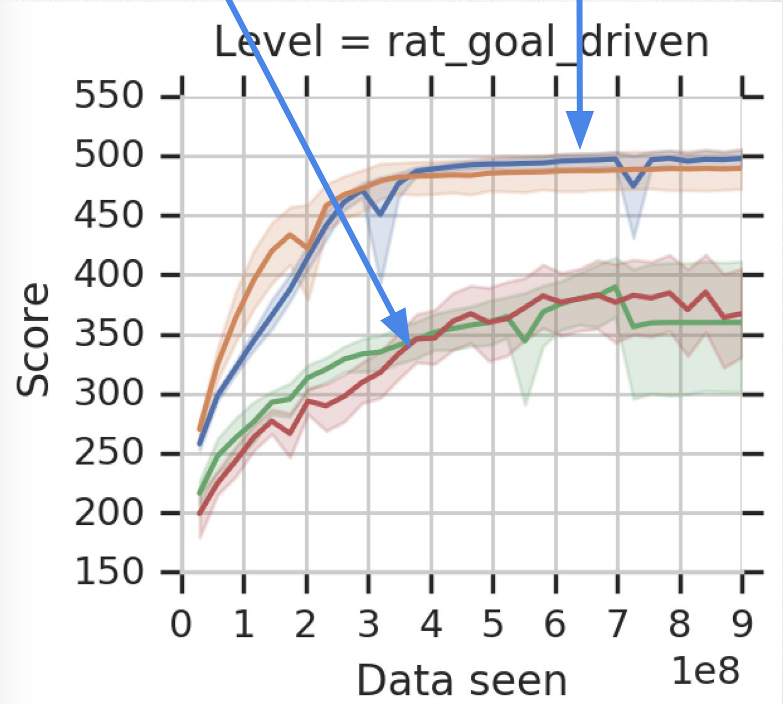
Actual top down view

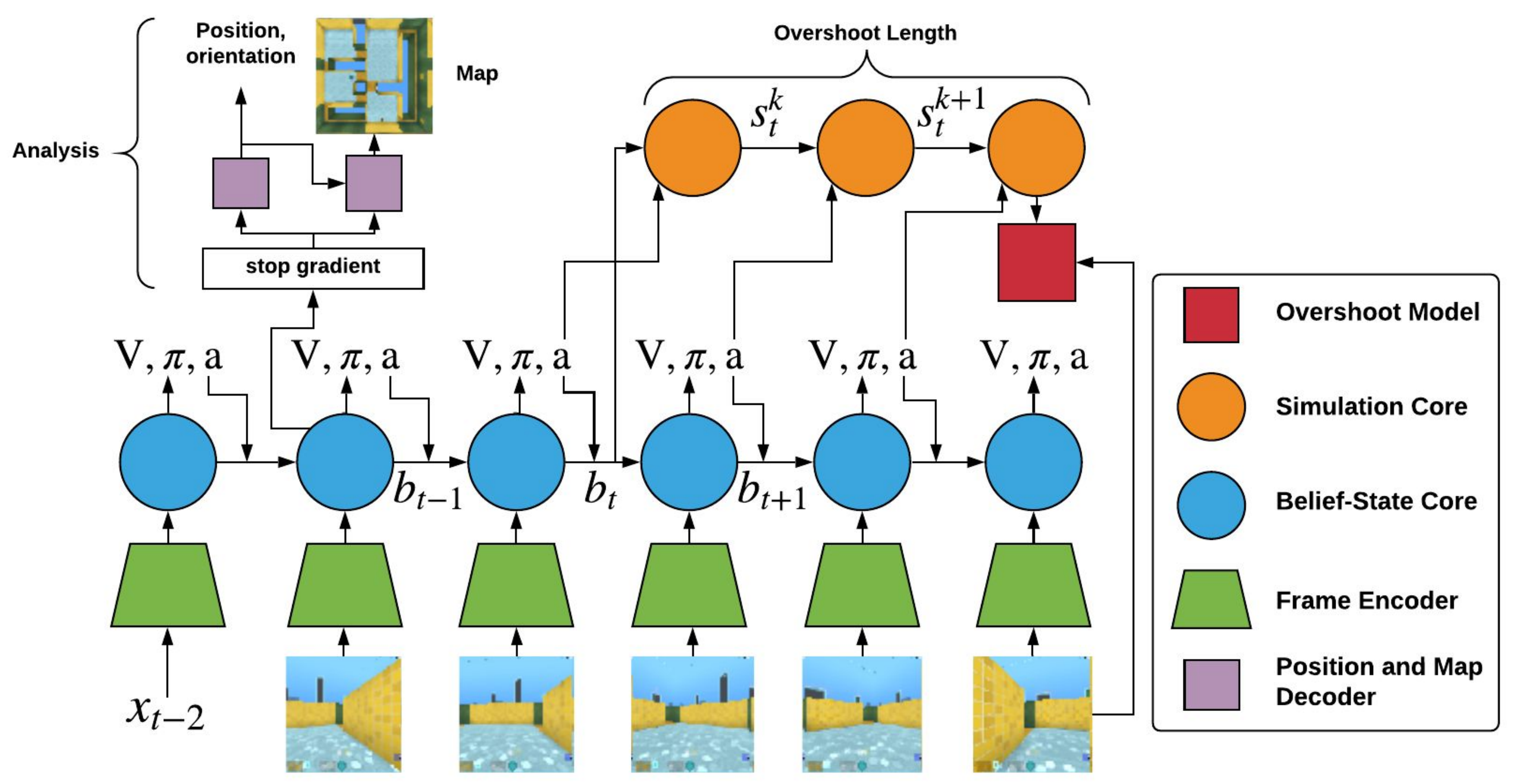
Top down view reconstructed from agent rep.

Without belief state training

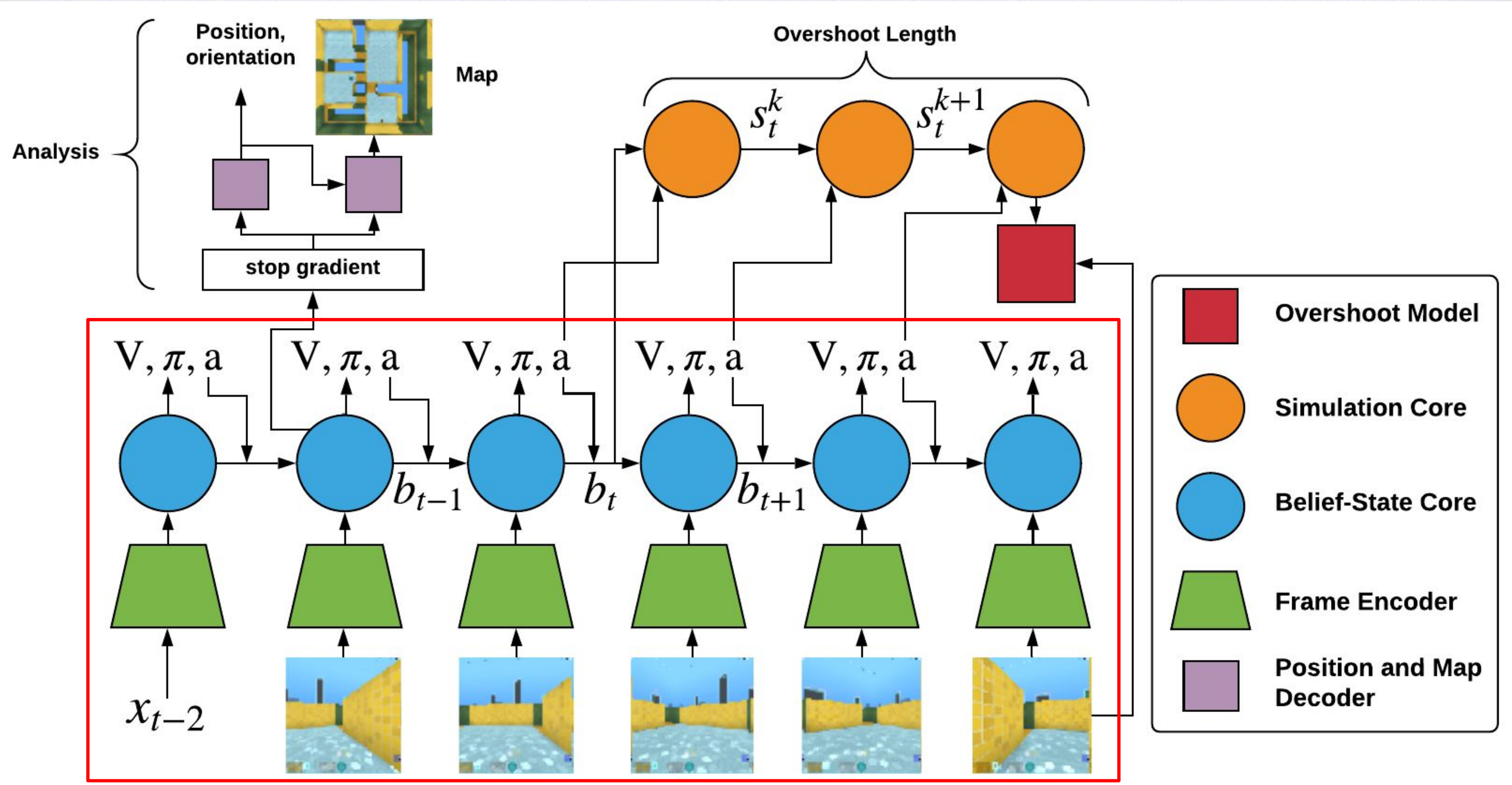
With belief state training



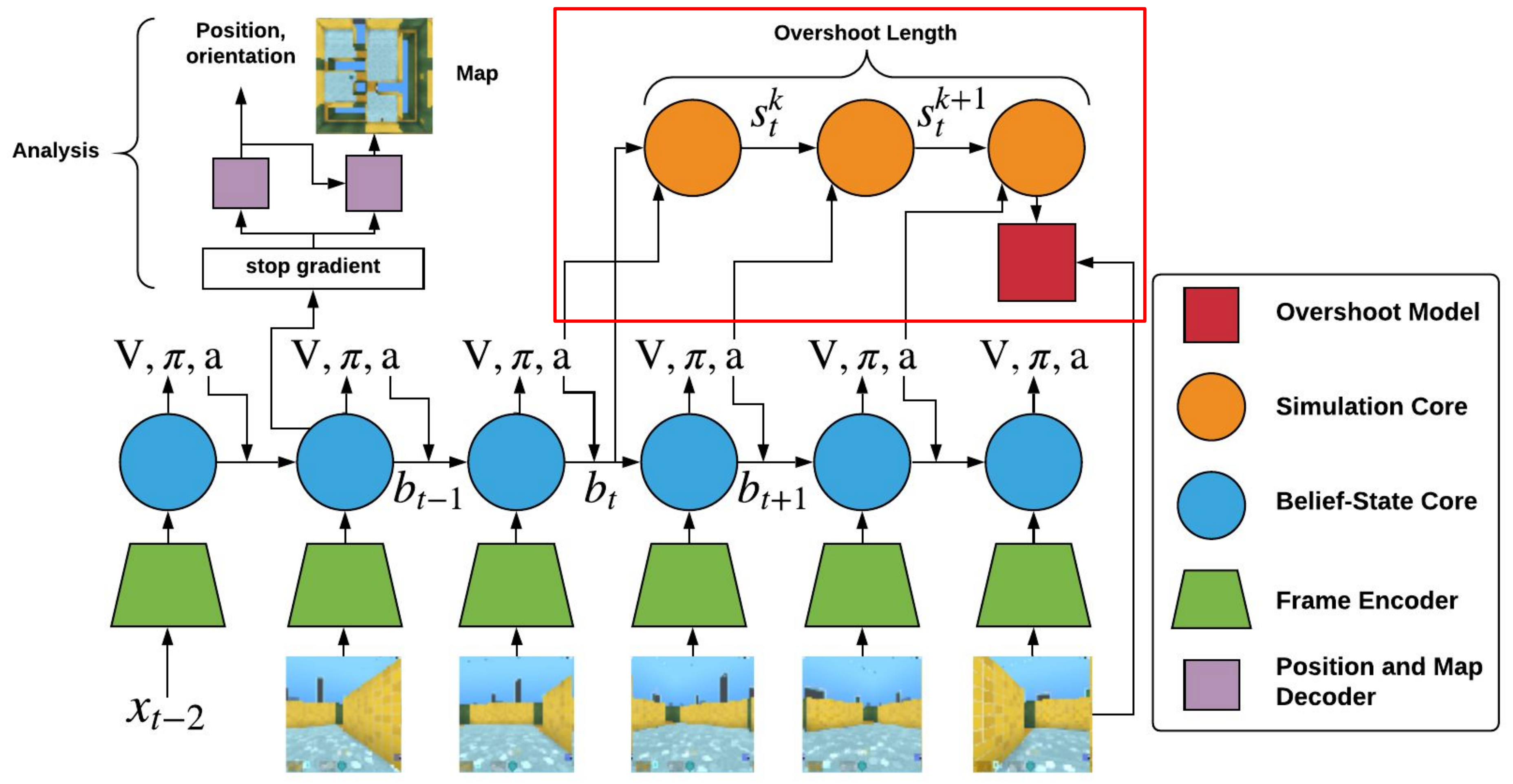




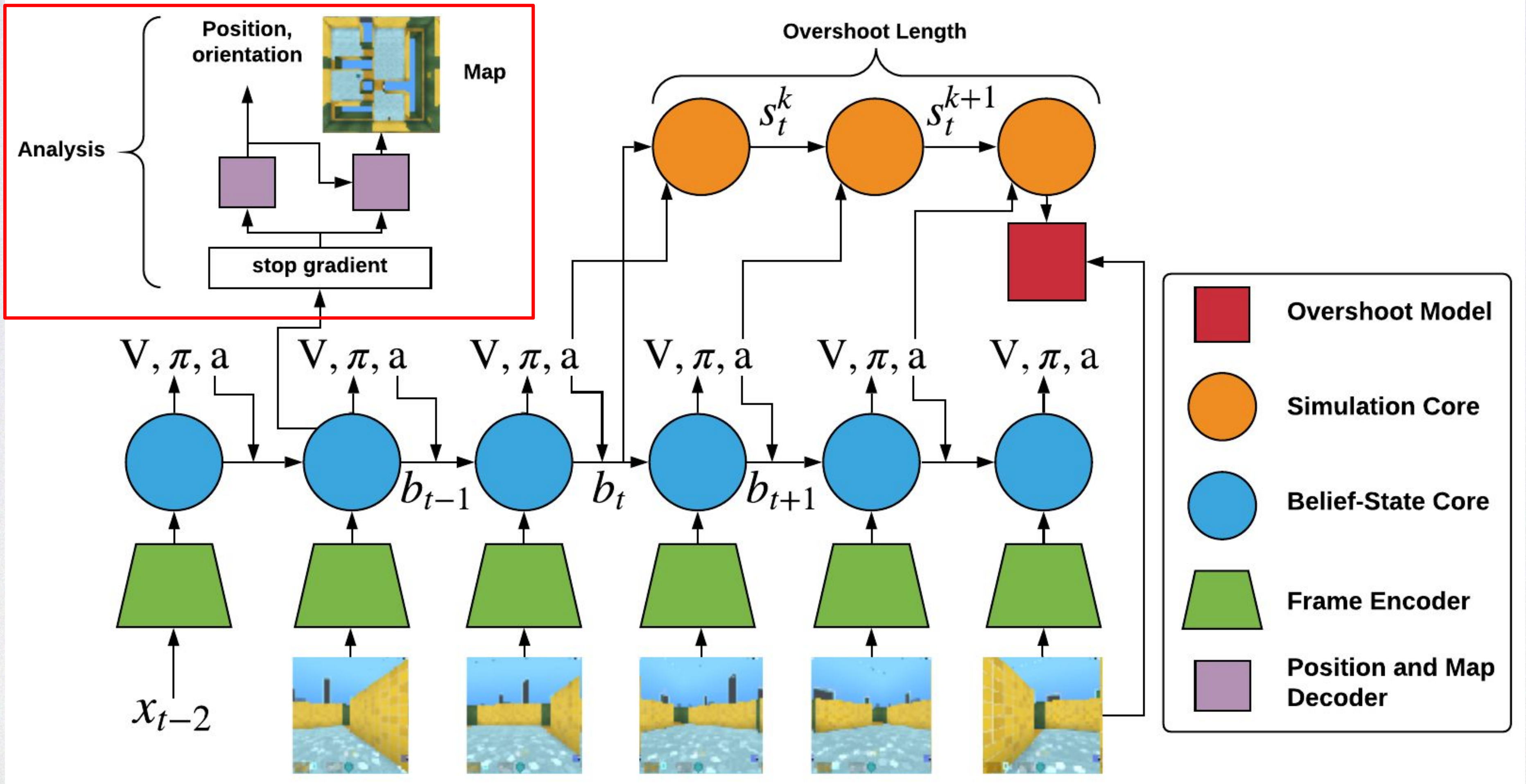
Analysis
Position, orientation
Map
stop gradient
Overshoot Length
s_t^k
s_t^{k+1}
V, π, a
b_{t-1}
b_t
b_{t+1}
x_{t-2}
Overshoot Model
Simulation Core
Belief-State Core
Frame Encoder
Position and Map Decoder



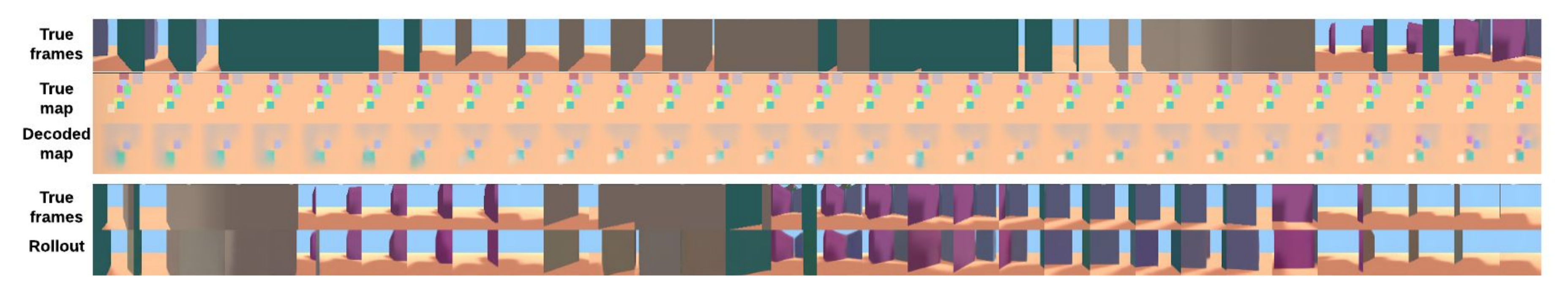
Analysis
Position, orientation
Map
stop gradient
Overshoot Length
s_t^k
s_t^{k+1}
V, π, a
V, π, a
V, π, a
V, π, a
V, π, a
V, π, a
b_{t-1}
b_t
b_{t+1}
x_{t-2}
Overshoot Model
Simulation Core
Belief-State Core
Frame Encoder
Position and Map Decoder

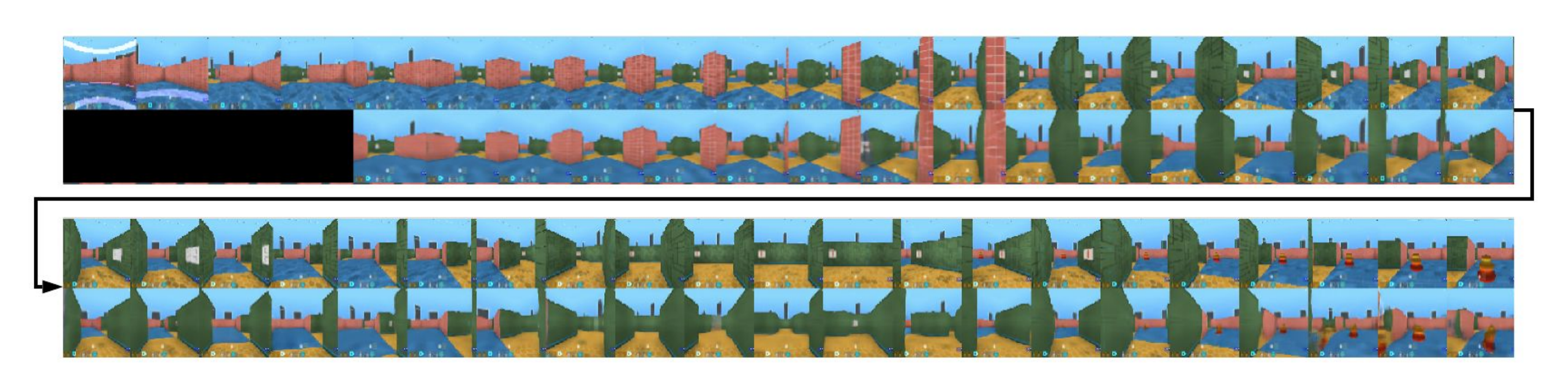


Analysis
Position, orientation
Map
stop gradient
Overshoot Length
s_t^k
s_t^{k+1}
V, π, a
b_{t-1}
b_t
b_{t+1}
x_{t-2}
Overshoot Model
Simulation Core
Belief-State Core
Frame Encoder
Position and Map Decoder

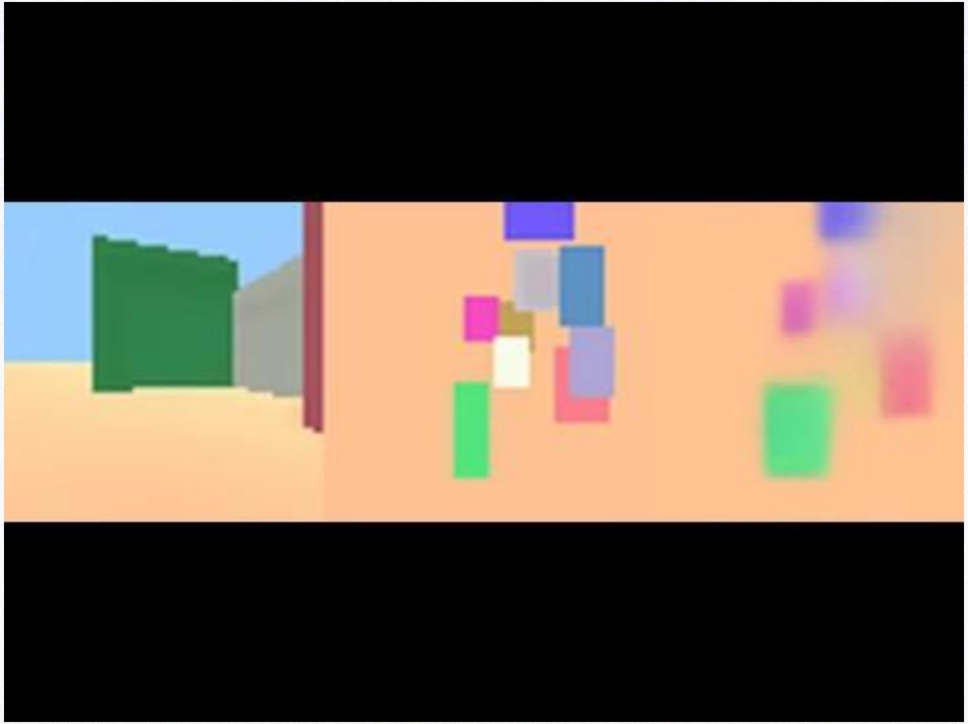

Analysis
Position, orientation
Map
stop gradient
Overshoot Length
s_t^k
s_t^{k+1}
V, π, a
b_{t-1}
b_t
b_{t+1}
x_{t-2}
Overshoot Model
Simulation Core
Belief-State Core
Frame Encoder
Position and Map Decoder

Model Rollouts



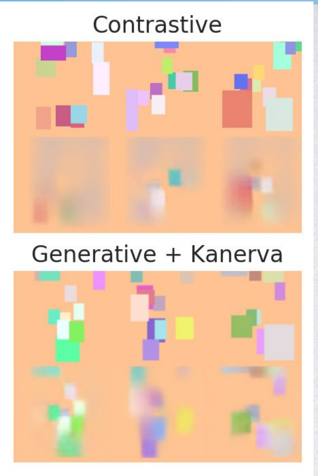


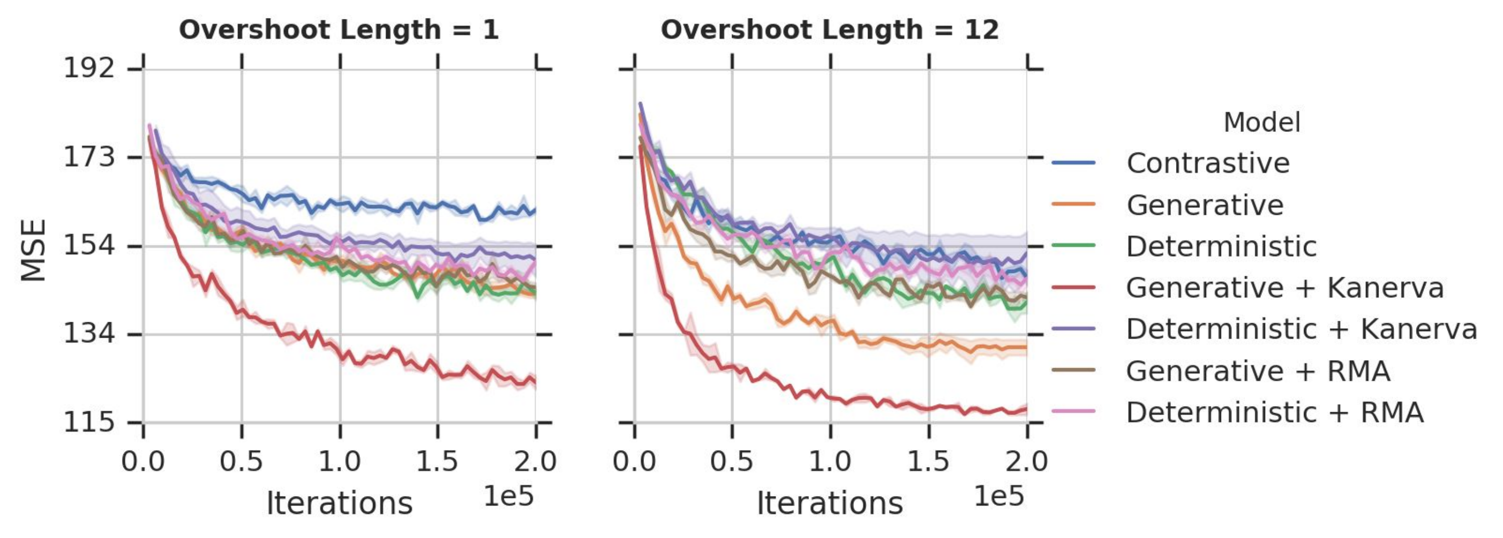
Formation of belief states



Top down view reconstruction error (mapping)

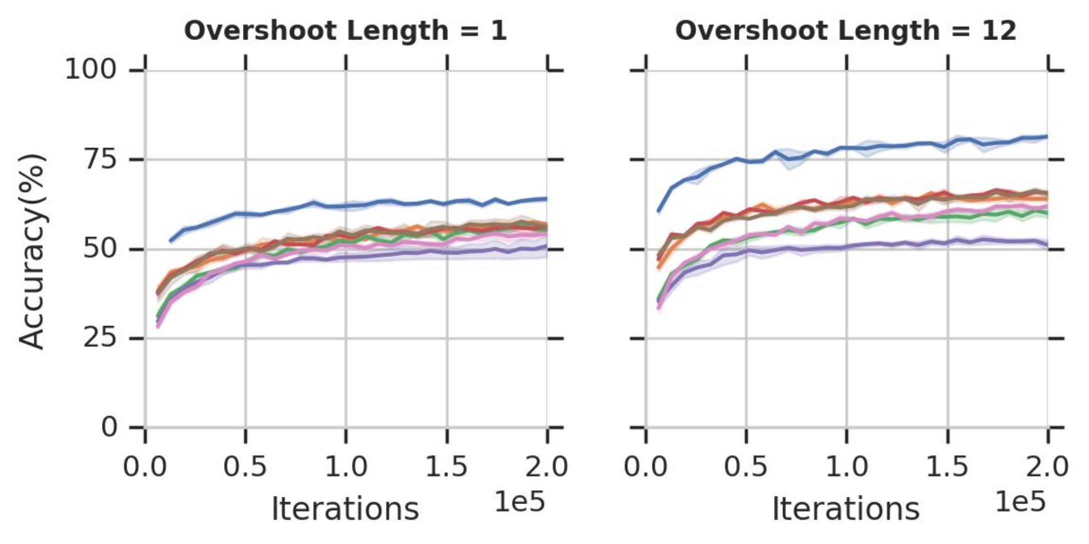
1) Longer simulations improve map decoding performance (if proper generative model is used)
2) Generative model works better than deterministic decoder and CPC
3) Kanerva Machine episodic memory works the best



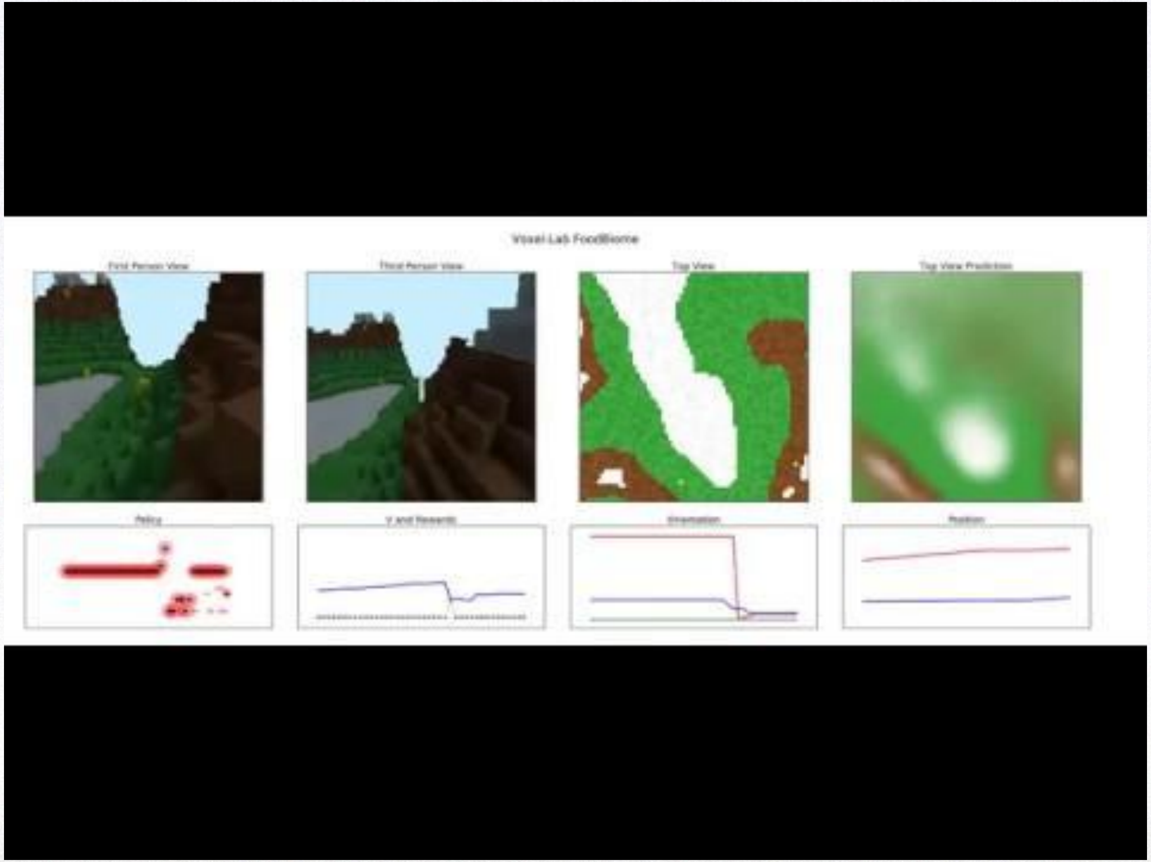


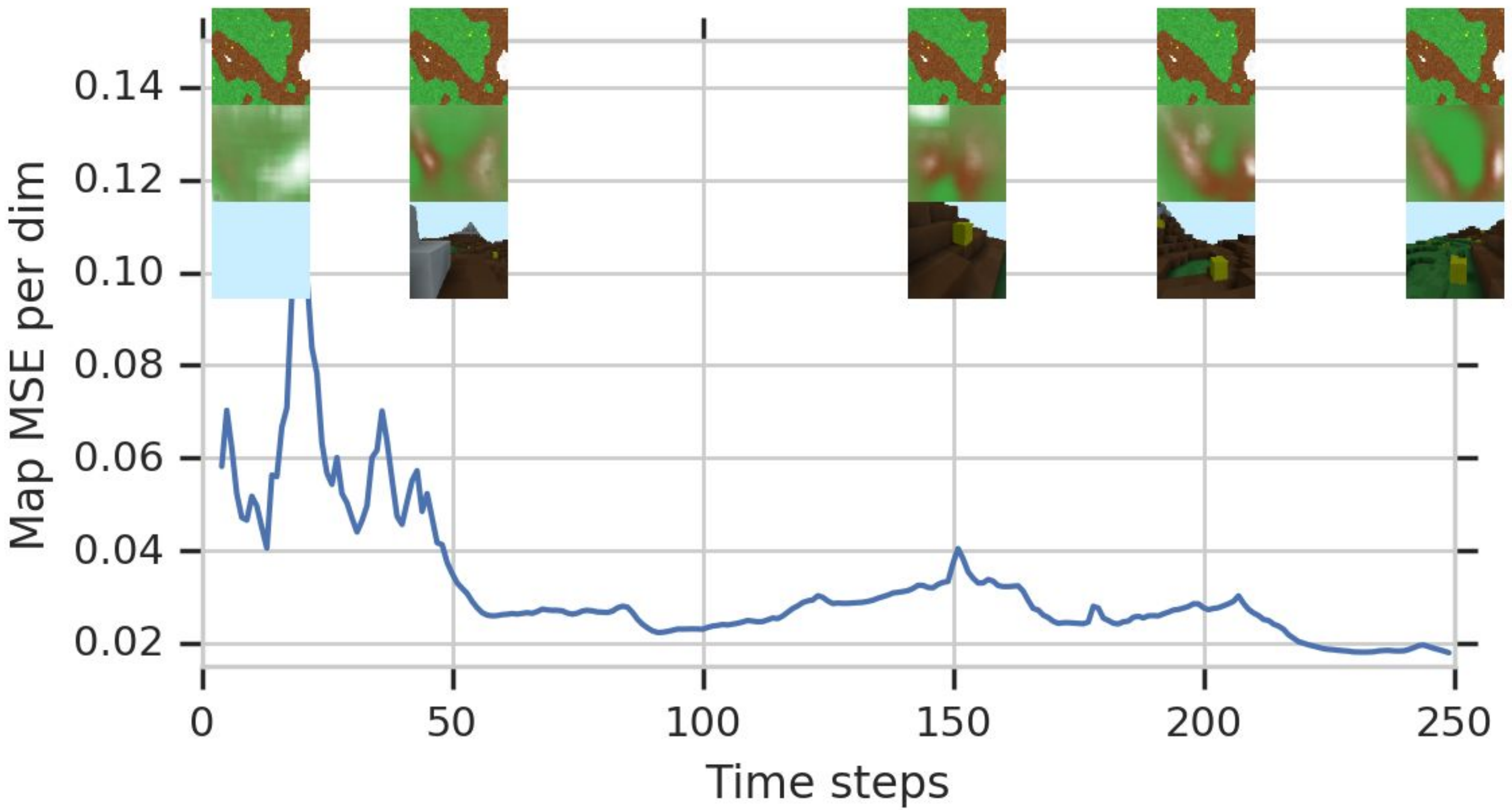
Position and orientation prediction (localization)

1) Longer simulations improve localization
2) CPC works the best



More complex naturalistic environment

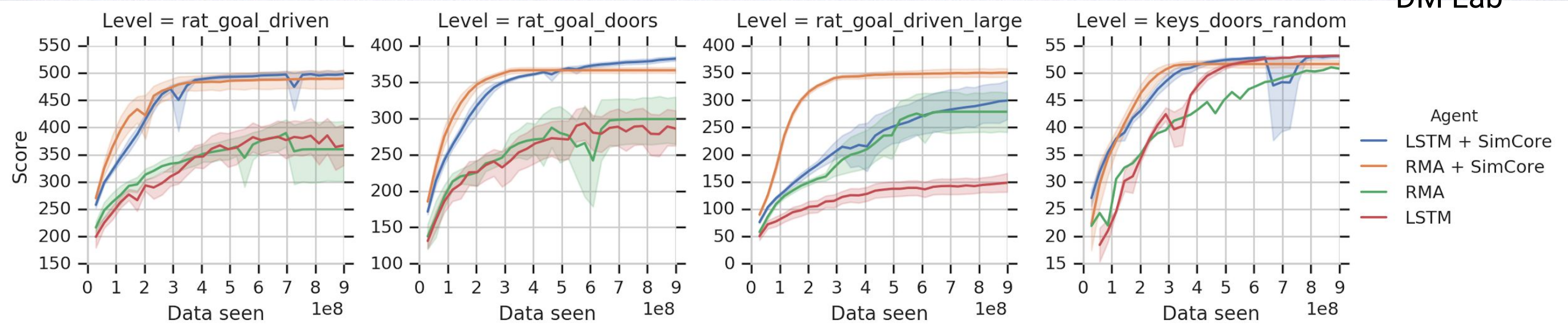




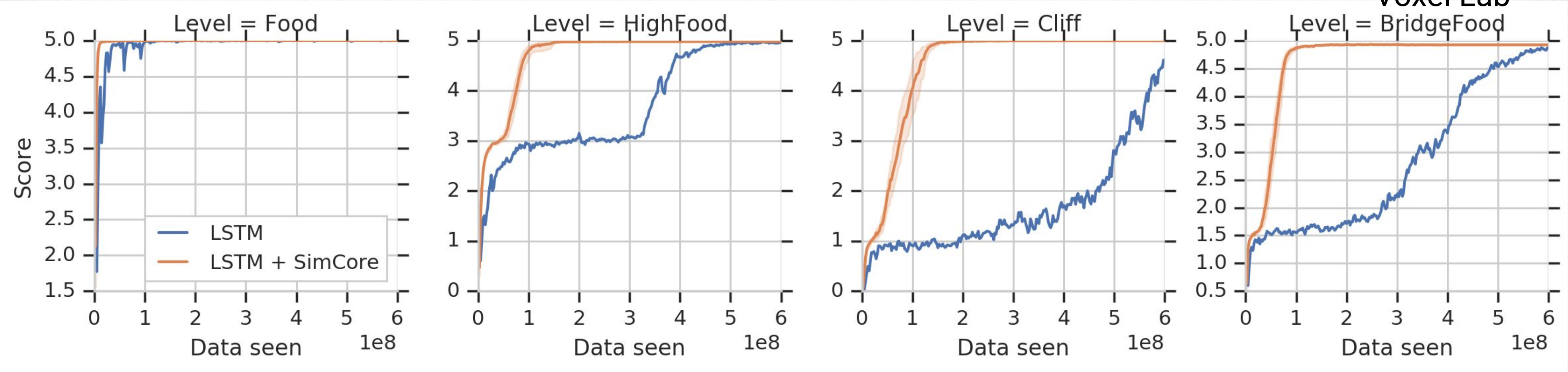
Map MSE per dim
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0
50
100
150
200
250
Time steps

RL Performance

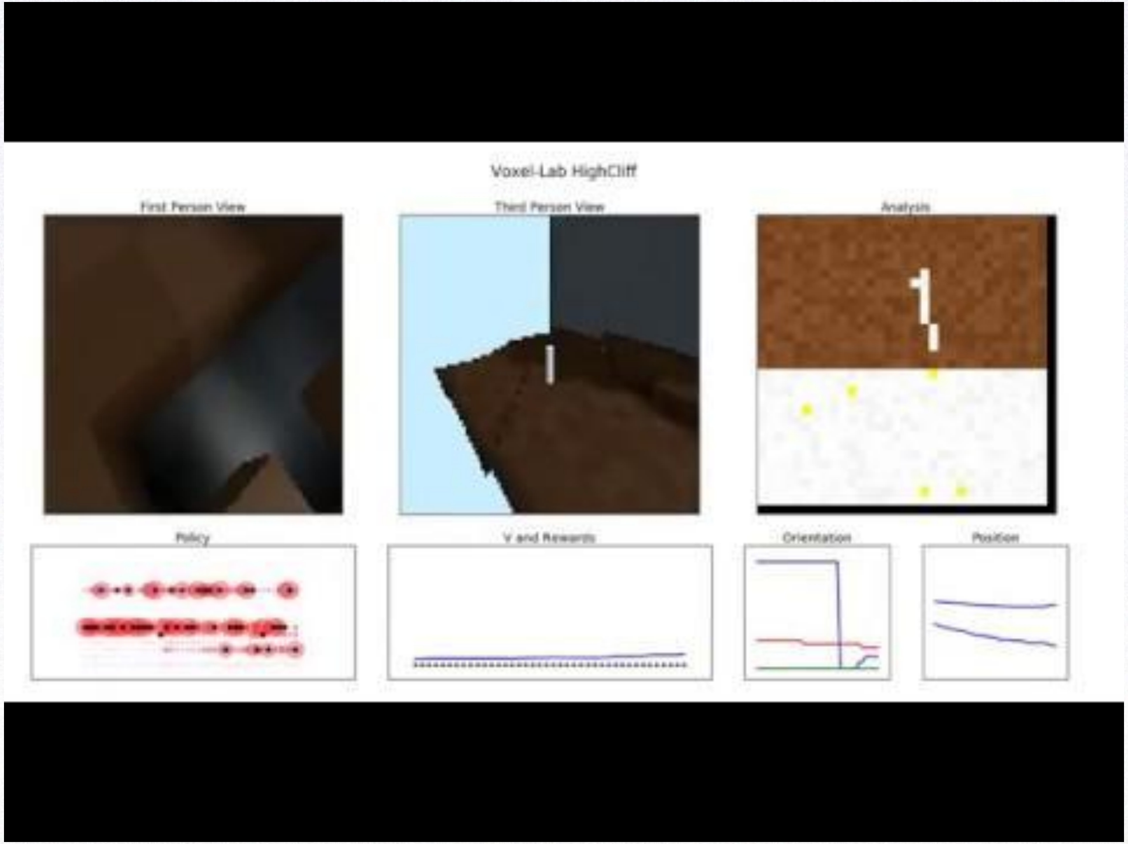
DM Lab



Voxel Lab



Learns to build stairs and towers



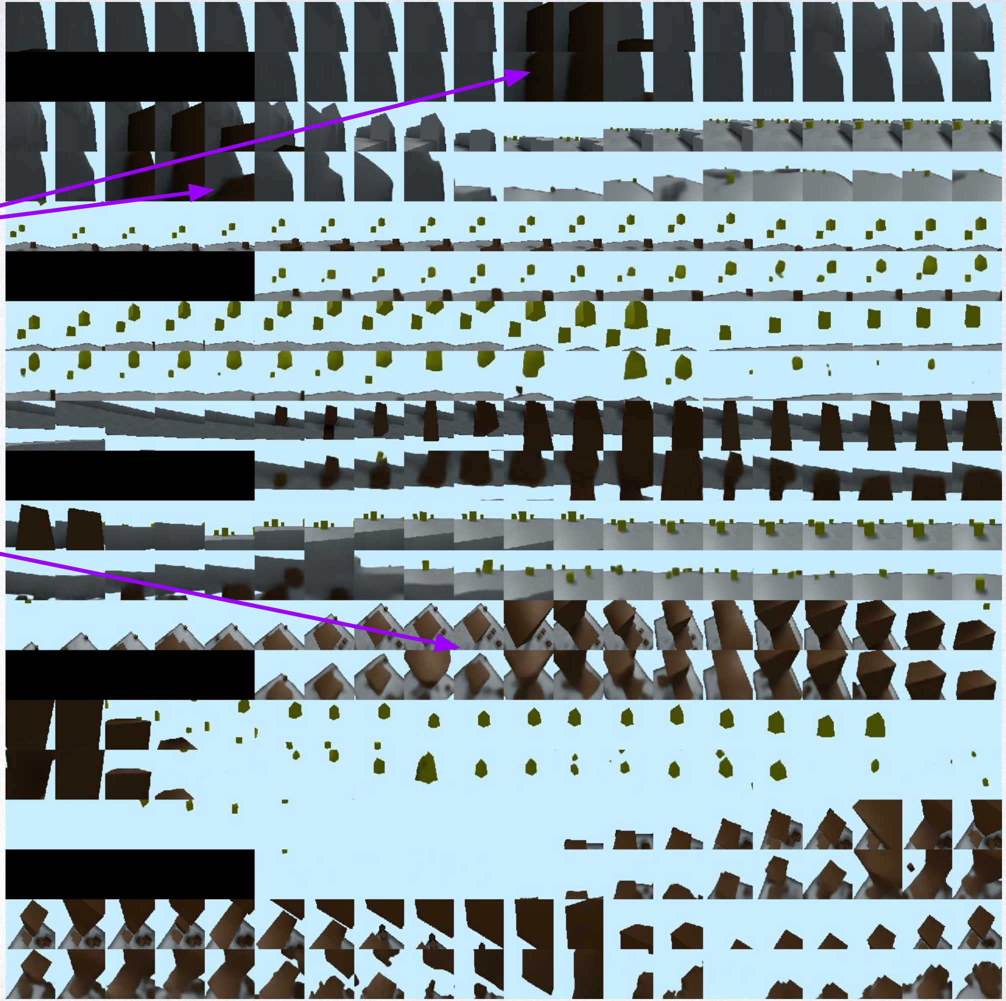
Learns to imagine building them

Building stairs

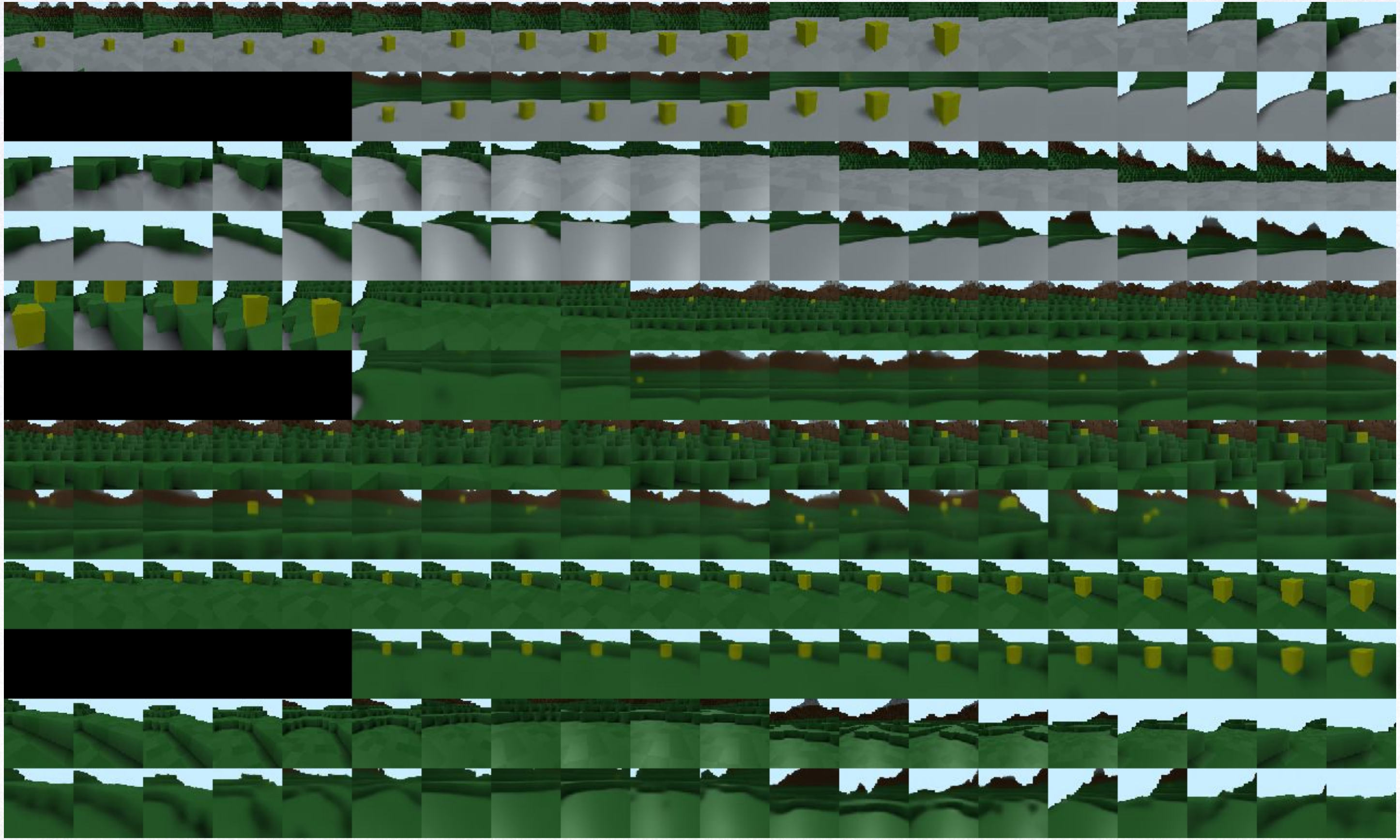
Building towers

input

rollout



Naturalistic landscape - imagines eating the yellow block



Towards a Definition of Disentangled Representations

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, Alexander Lerchner

DeepMind

Motivations

- What is the role Group theory in "feature/representation learning"?
- invariance vs covariance vs equivariance
- When is "Disentanglement" useful?
- Our intuition should not rely on a specific coordinate system
- Lack of universality => lack of generalisation to domains where we have no intuition

Group theory 1:1
What is a Group?

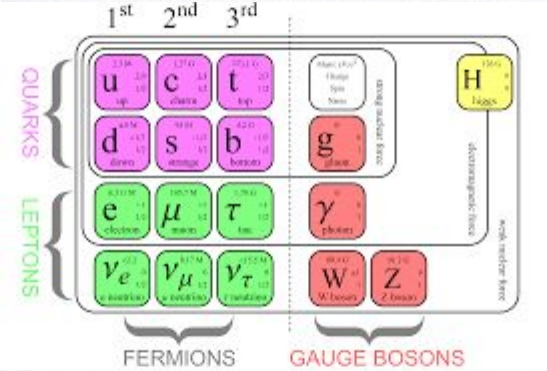
- A group G is a set endowed with a binary operator o. The operator must satisfy the following properties:
  - Closure: x1, x2 in G, then x1 o x2 is in G
  - Identity: there is an e in G such that e o x = x (for all x in G)
  - Inverse: for any x in G there is a y such that y o x = e
  - Associativity: x1 o (x2 o x3) = (x1 o x2) o x3

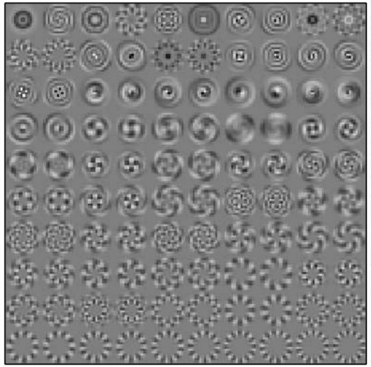
Example:
G = (Reals, o=+, e = 0)

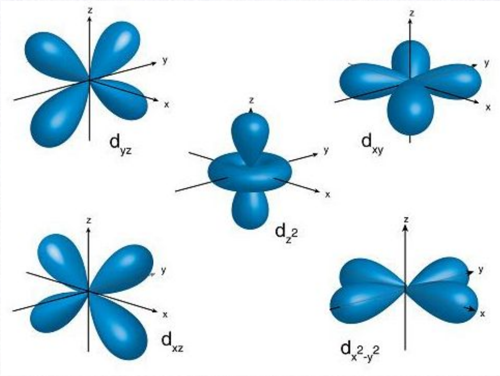
Example:
G = (Rotation matrices, o=matrix product, e = I)

Group theory 1:1
What is a representation of a group?

A representation of a group is a mapping from group element g to an operator rho(g) acting on a different space H with operator x. Such that \rho (g1 o g2) = \rho(g1) x \rho(g2)







Group theory 1:1

What is an irreducible representation of a group?

Representations that leave invariant some subset of H that cannot be broken down into smaller invariant subsets.

Group theory 1:1
What is a Lie Group?

A Lie Group is a group that is also a manifold (e.g. Translations and Rotations)

Group theory 1:1
Invariance and Equivariance

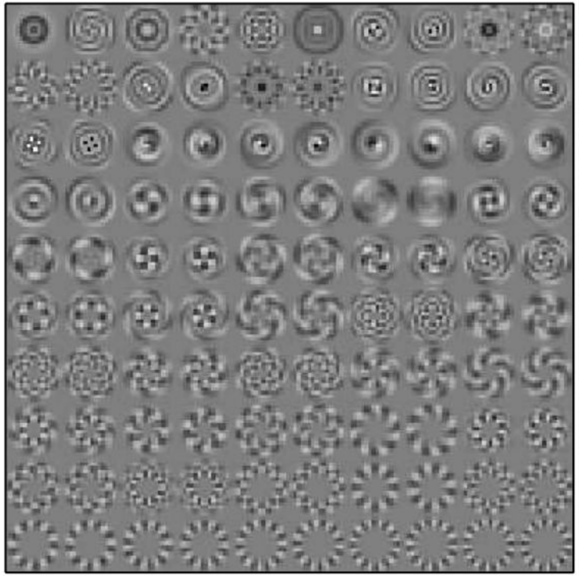
We say that our features f are *invariant* under a symmetry group G iff f(g o data) = f(data) for any g in G. Example: output of pooling layers

We say that we have *equivariant features* f under a symmetry group G if f(g o data) = J x f(data) Example: output of convolution layers, vector fields

Too abstract?

$$p(\mathbf{y}|\mathbf{x},\varphi) = \mathcal{N}(\mathbf{y}|\mathbf{W}\mathbf{R}(\varphi)\mathbf{W}^T\mathbf{x},\sigma^2)$$

$$p(\varphi_j) = \frac{1}{2\pi I_0(\|\eta_j\|)}\exp\left(\eta_j^T T(\varphi_j)\right).$$



[Learning the Irreducible Representations of Commutative Lie Groups]

[2D/3D Rotation-Invariant Detection using Equivariant Filters and Kernel Weighted Mapping]

Invariance, Equivariance, Classifiers and Generative Models

When building classifiers it is desirable that its output to be invariant under some groups (e.g. translation, rotation).

That is, we want to build ***both features and losses*** such that Loss(g o data) = Loss(data), where o = 2D Translations x 2D Rotations

However, when building generative models, we should seek ***invariant losses***, but the ***representations should be equivariant***. This is because invariant representations loose information, destroying the ability to reconstruct the data.

General Recipe to build invariant networks

First, build equivariant features $\phi(g \circ x) = T_g \circ \phi(x)$

Second, apply a 'pooling' operator $\psi(x) = \int d\mu(g) T_g \circ \phi(x)$

Example: convnets

Weyl's Principle

The elementary components of a system are the irreducible representations of the symmetry group of the system. Example: entire physics.

Why does it matter to AI?

3D OBJECTS are irreducible representations of 3D Translation x Rotation x Scale groups (Galileo Group)

Group theory 1:1

Weyl principle => Irreducible representations ⇔ Disentanglement (since we have broken down our representations into the smallest possible invariant sets)

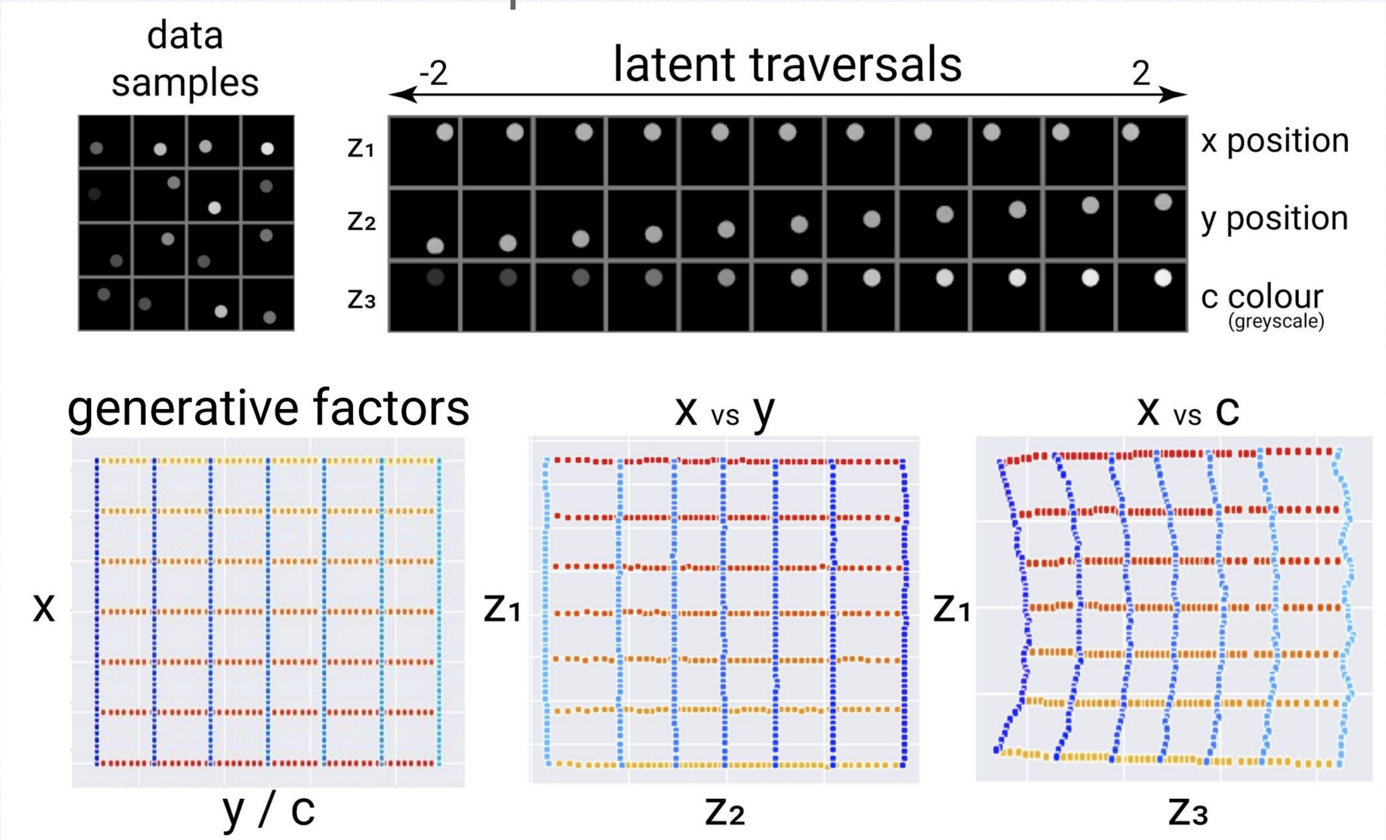
Towards a Definition of Disentangled Representations

A space Z is **disentangled** with respect to a group decomposition $G = G_1 \times G_2 \times \ldots \times G_N$ if:

1) There is an action $\cdot : G \times Z \to Z$
2) The map $f : W \to Z$ is equivariant between the actions on W and Z
3) There is a decomposition $Z = Z_1 \times Z_2 \times \ldots \times Z_N$ such that each Z_i is fixed by the action of all G_j $j \neq i$ and affected only by G_i

The actions are assumed to preserve any structure of Z (e.g. be linear or continuous).

Towards a Definition of Disentangled Representations



Summary

- The language of symmetry groups and differential geometry allows for generalisation of useful tools such as conv operators beyond classification of images.

- Such ideas have only been superficially explored in ML

- Do you want objects to emerge from neural nets + data? Let's build "Weyl machines" (or automated physicists).

Thanks to great collaborators

Karol Gregor
Marta Garnelo
Ali Eslami
Frederic Besse
Fabio Viola
Hamza Merzic
Yan Wu
Theophane Weber

Daniel Zoran
Alex Mott
Irina Higgins
Sebastian Racaniere
David Pfau
Mihaela Rosca
Marco Fraccaro
Aaron Van der Oord
Daan Wierstra