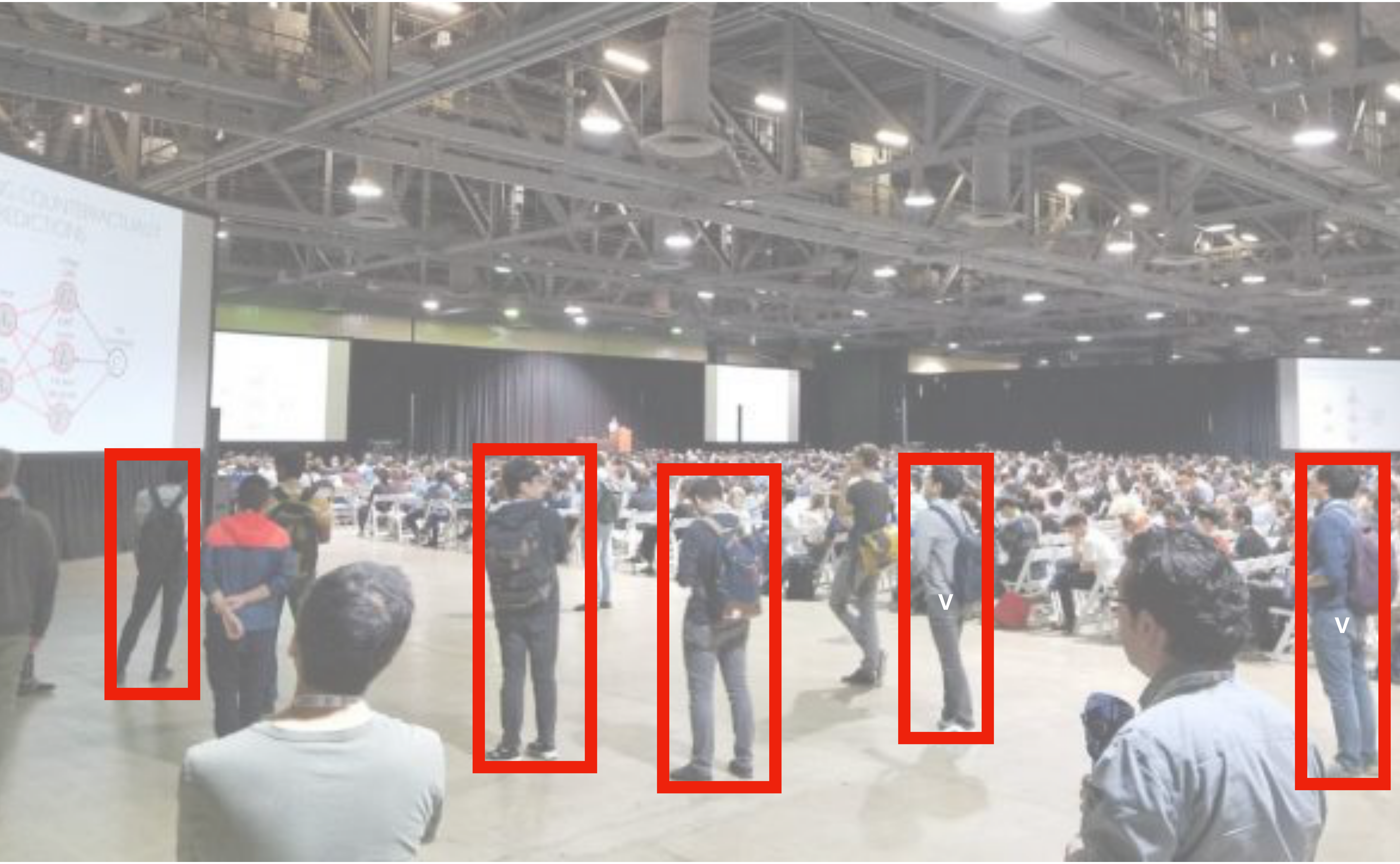
Neural scene representation and rendering

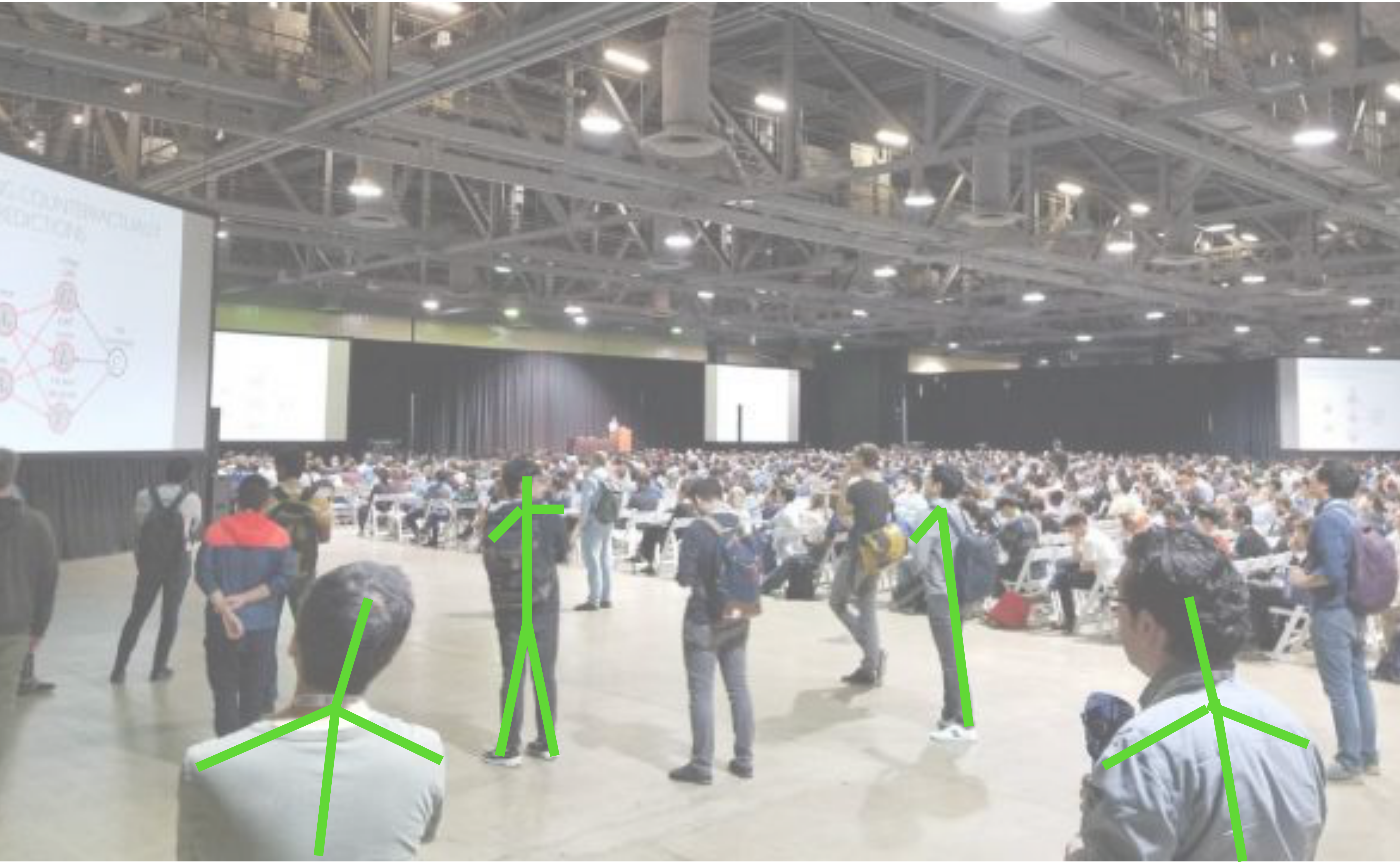
S. M. Ali Eslami*†, Danilo J. Rezende†, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, Andrei A. Rusu, Ivo Danihelka, Karol Gregor, David P. Reichert, Lars Buesing, Theophane Weber, Oriol Vinyals, Dan Rosenbaum, Neil Rabinowitz, Helen King, Chloe Hillier, Matt Botvinick, Daan Wierstra, Koray Kavukcuoglu, Demis Hassabis

DeepMind, 5 New Street Square, London EC4A 3TW, UK
*Corresponding author. Email: aeslami@google.com
†These authors contributed equally to this work

Presenter: Mei Chen
March 25, 2019

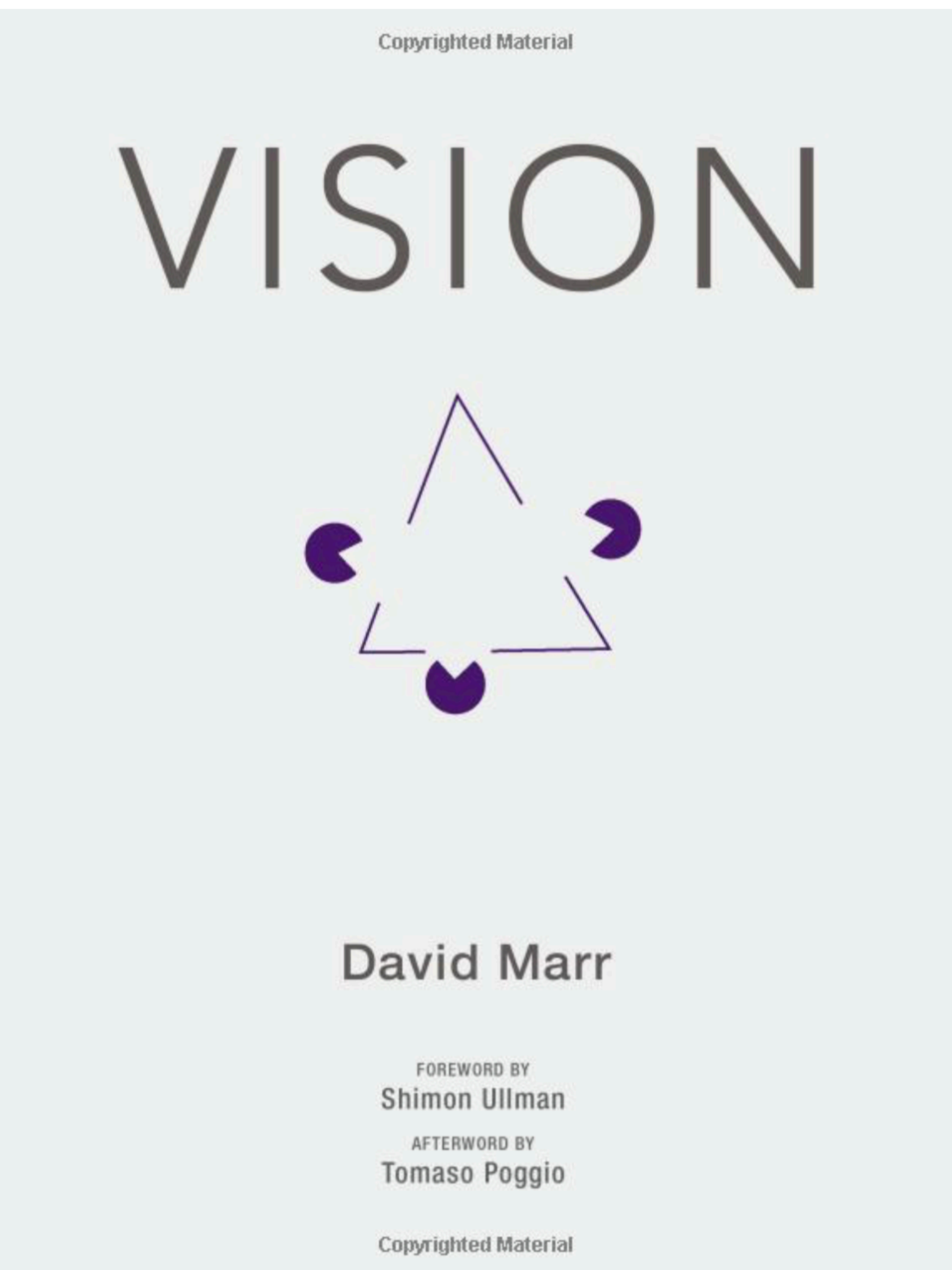
https://deepmind.com/blog/neural-scene-representation-and-rendering/

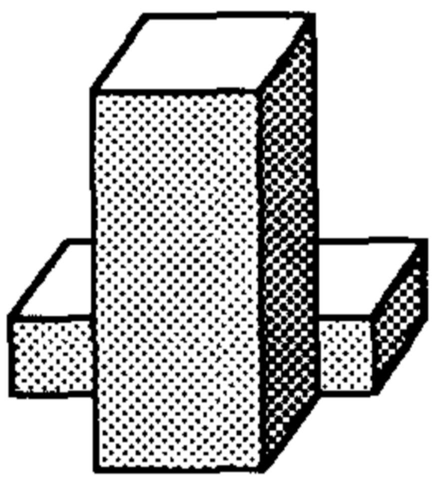

V
V



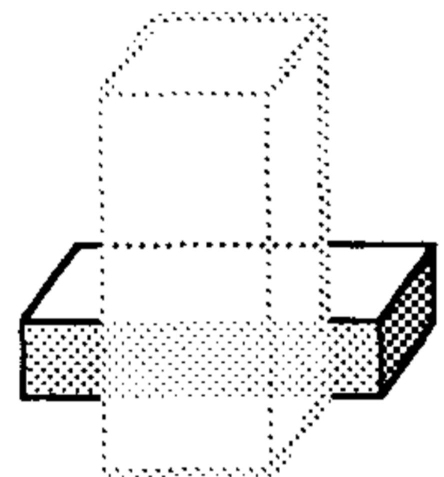
"What does it mean, to see? The plain man's answer would be, to know what is where by looking."

–David Marr

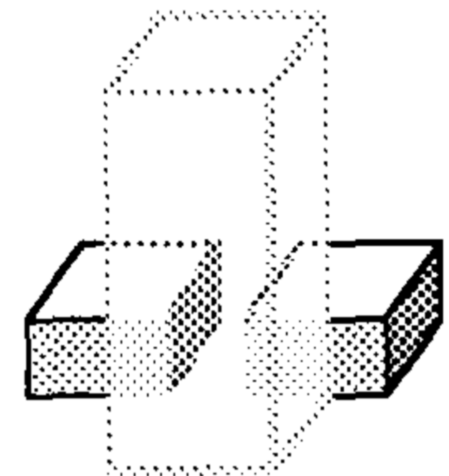




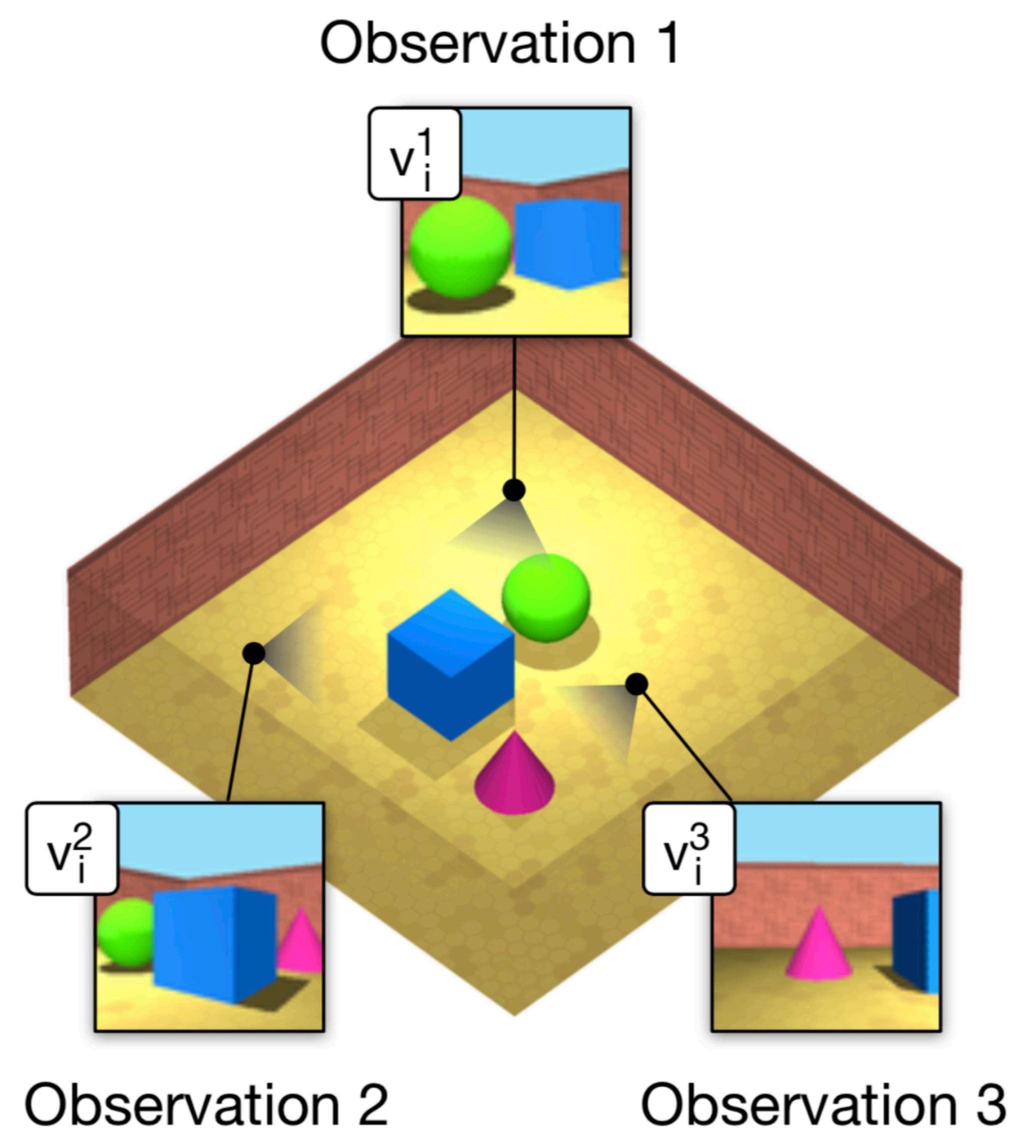
What you see.



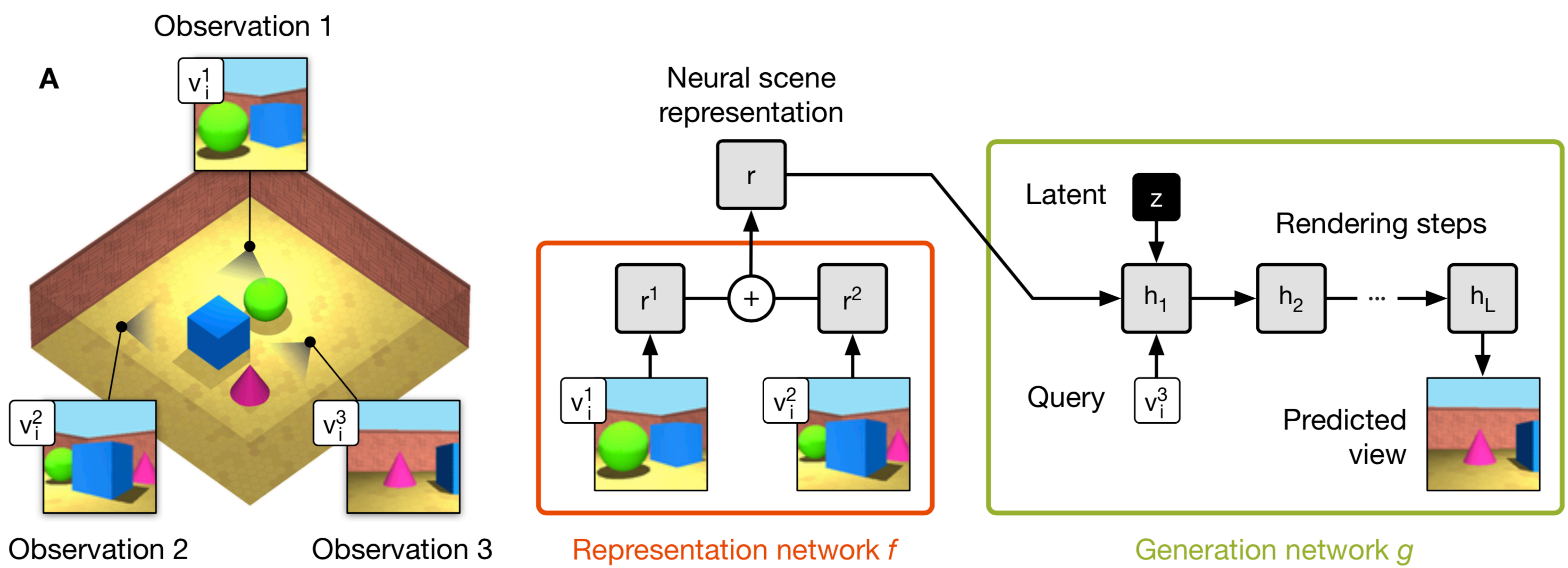
Is it this?



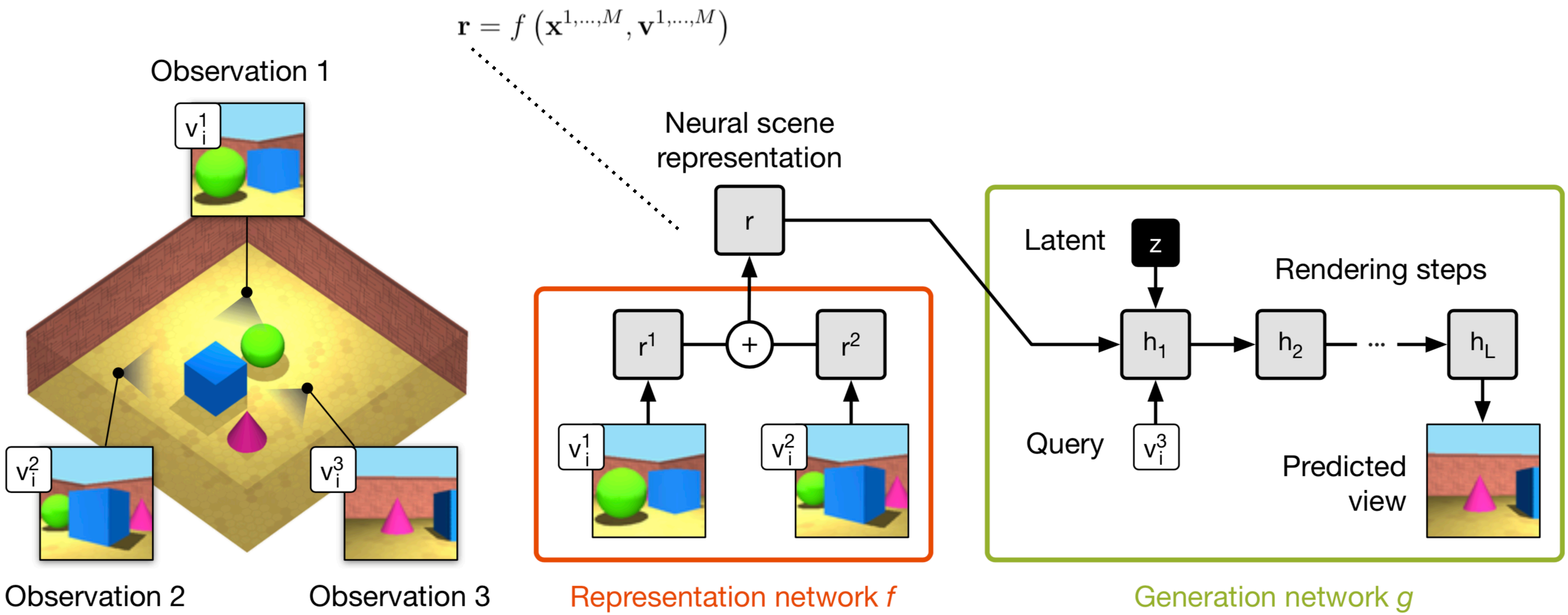
Or this?


Observation 1
v_i^1
v_i^2
v_i^3
Observation 2
Observation 3

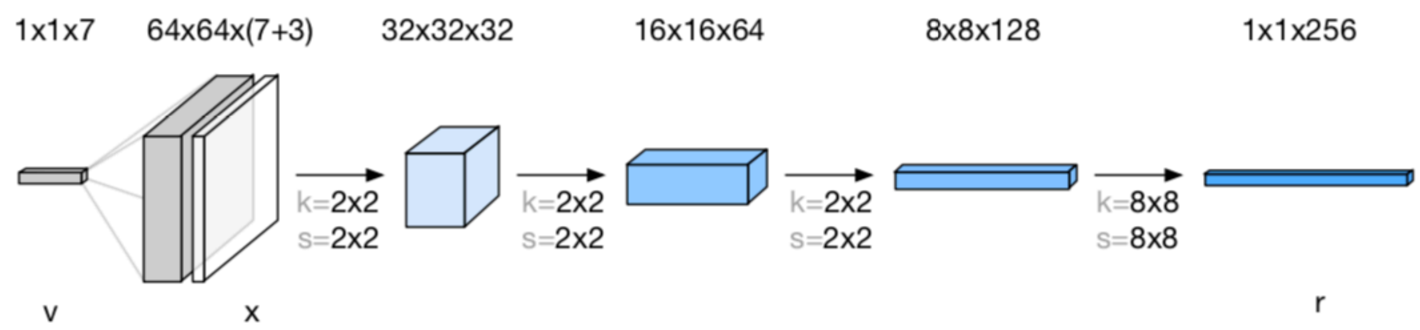
Generative Query Network



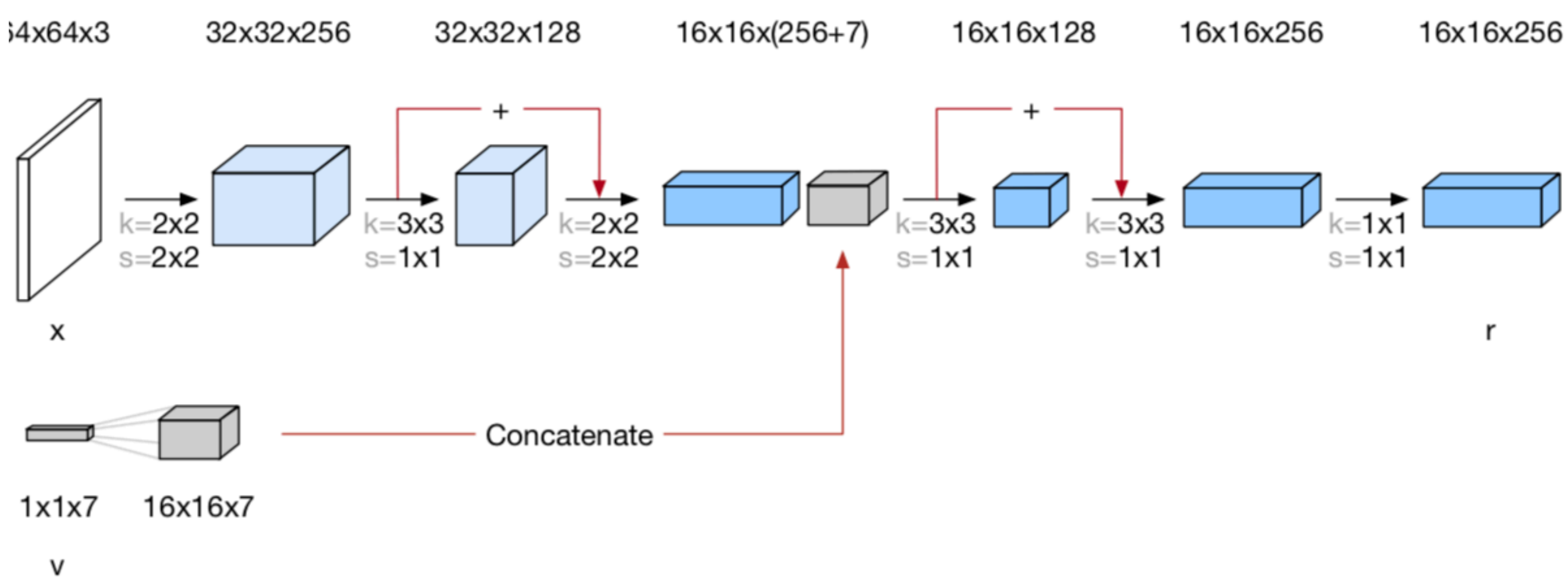
Generative Query Network



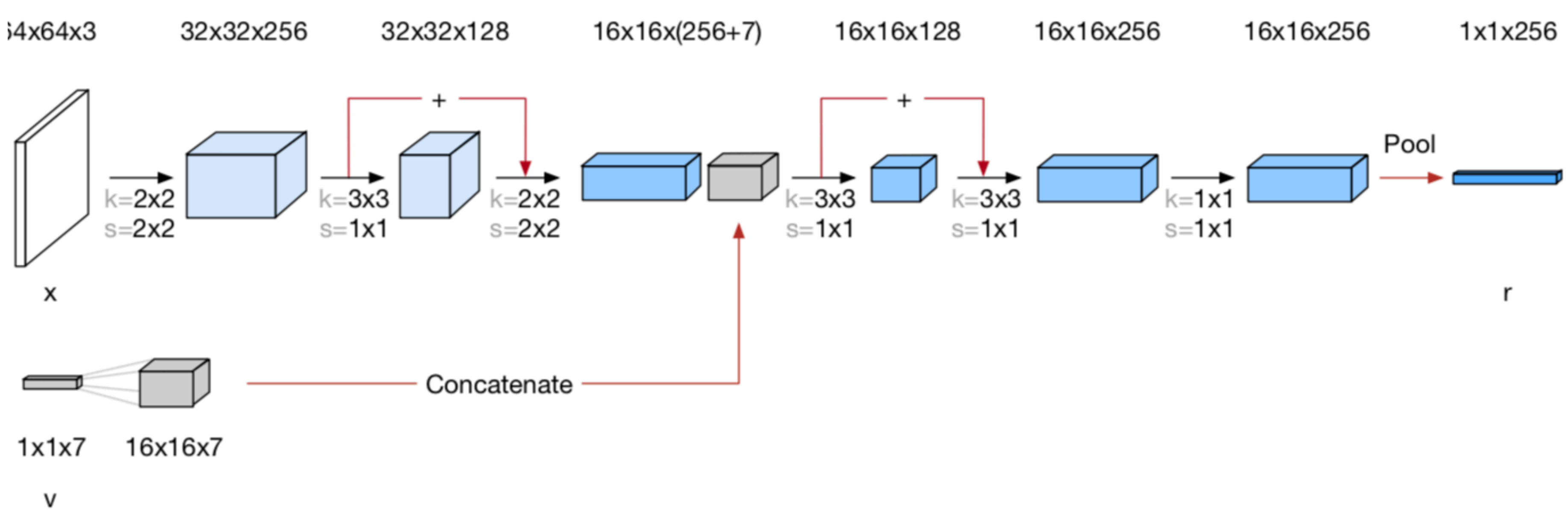
Representation Network Architecture



Pyramid

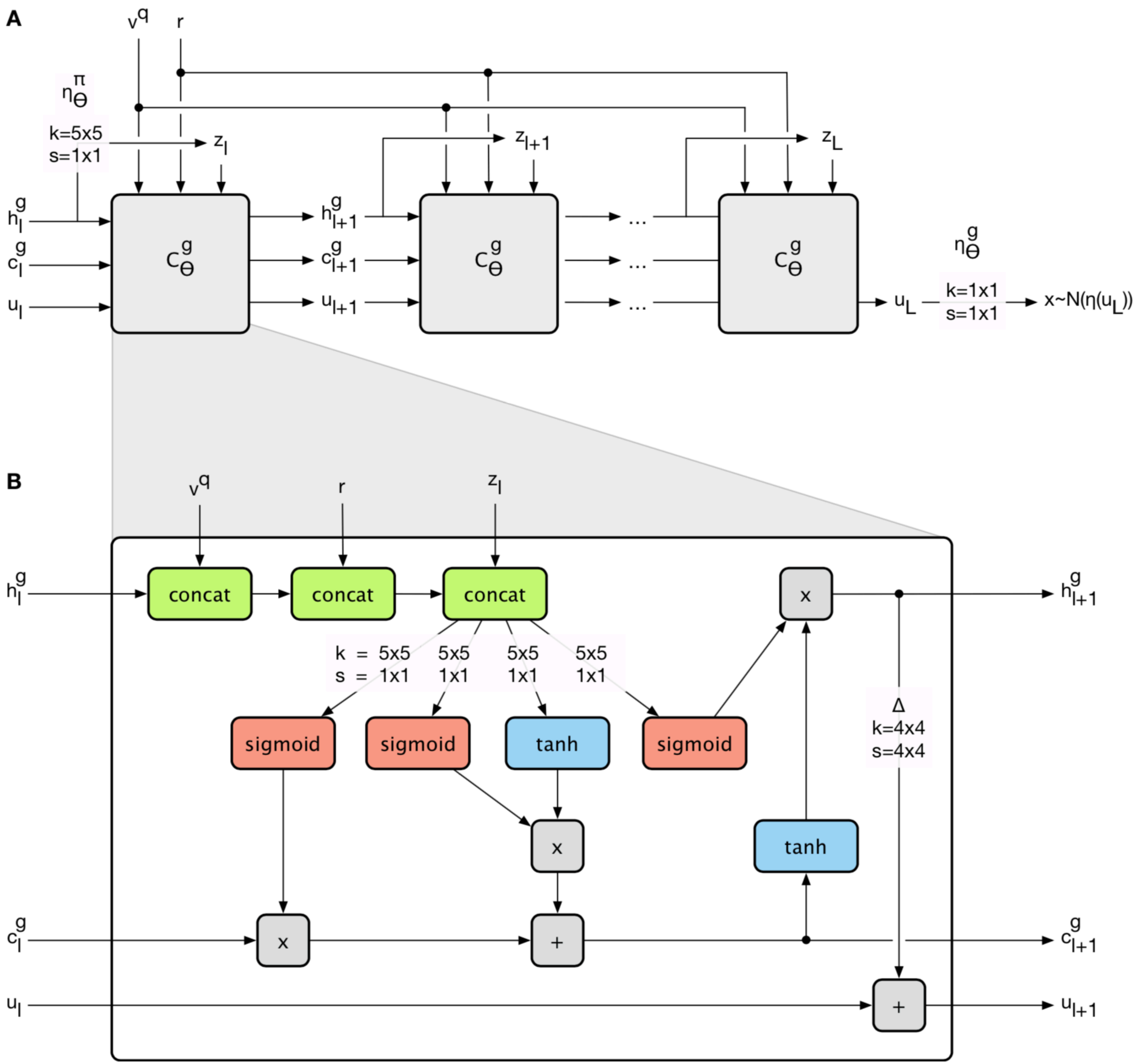


Tower

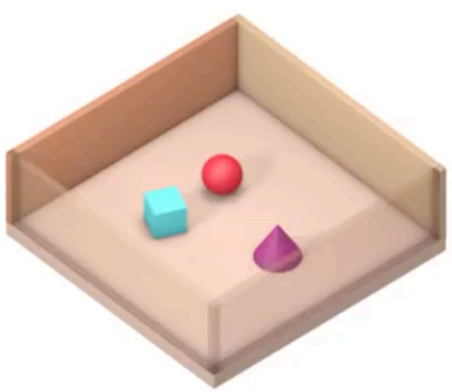


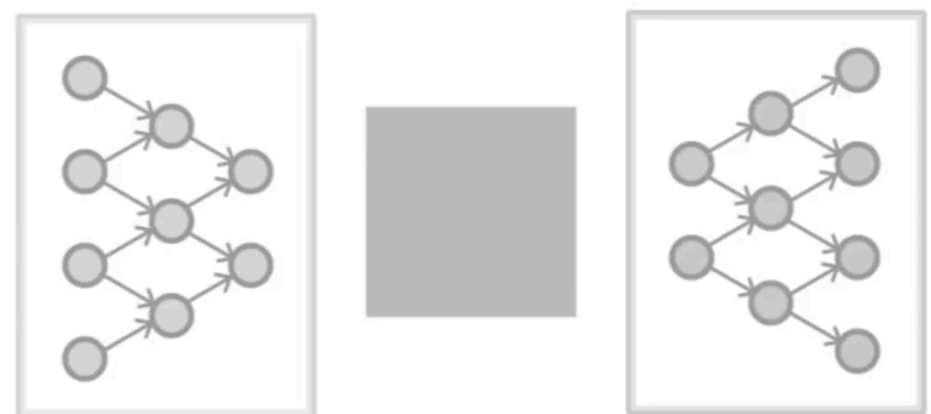
Pool

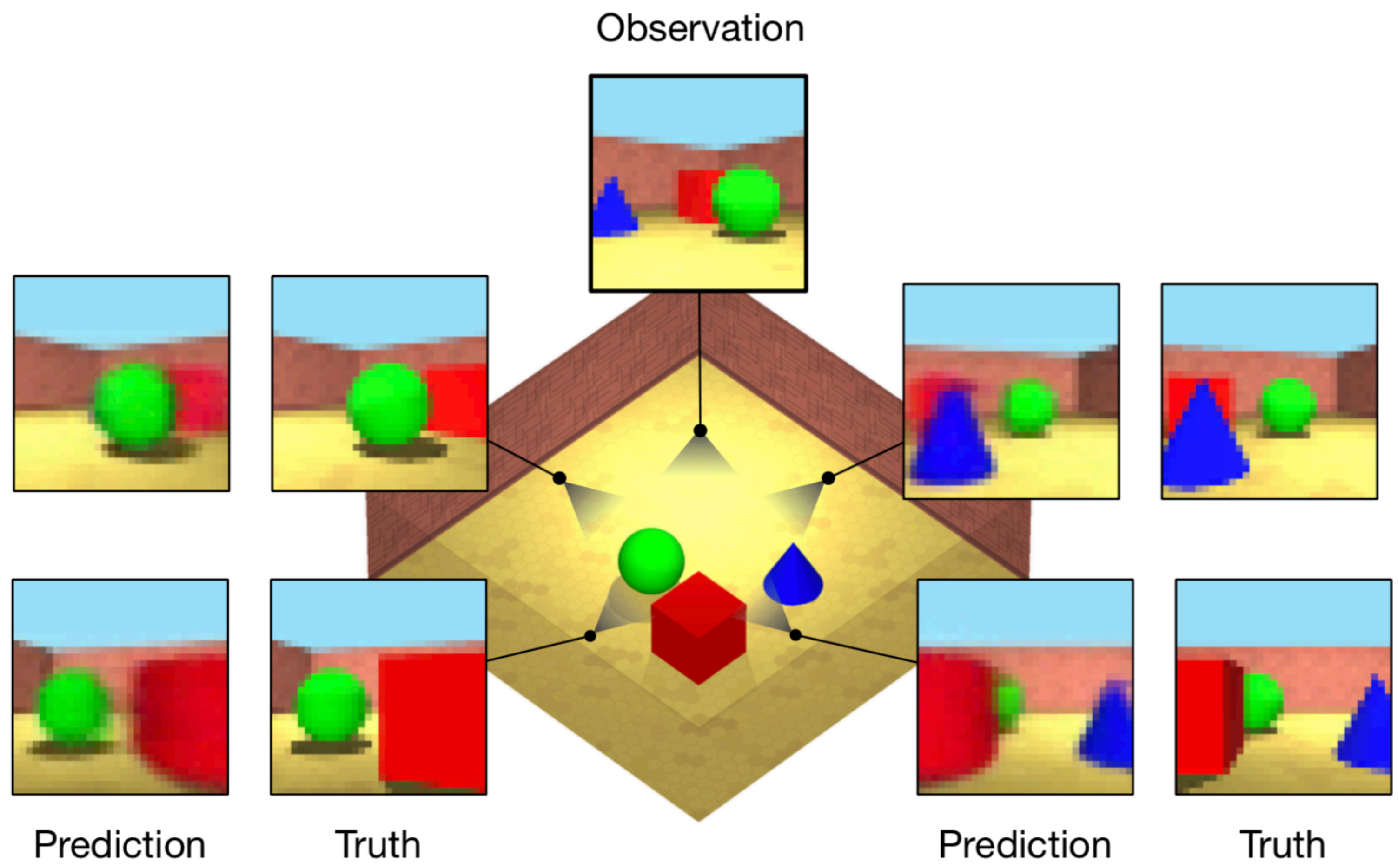
Generation Network Architecture

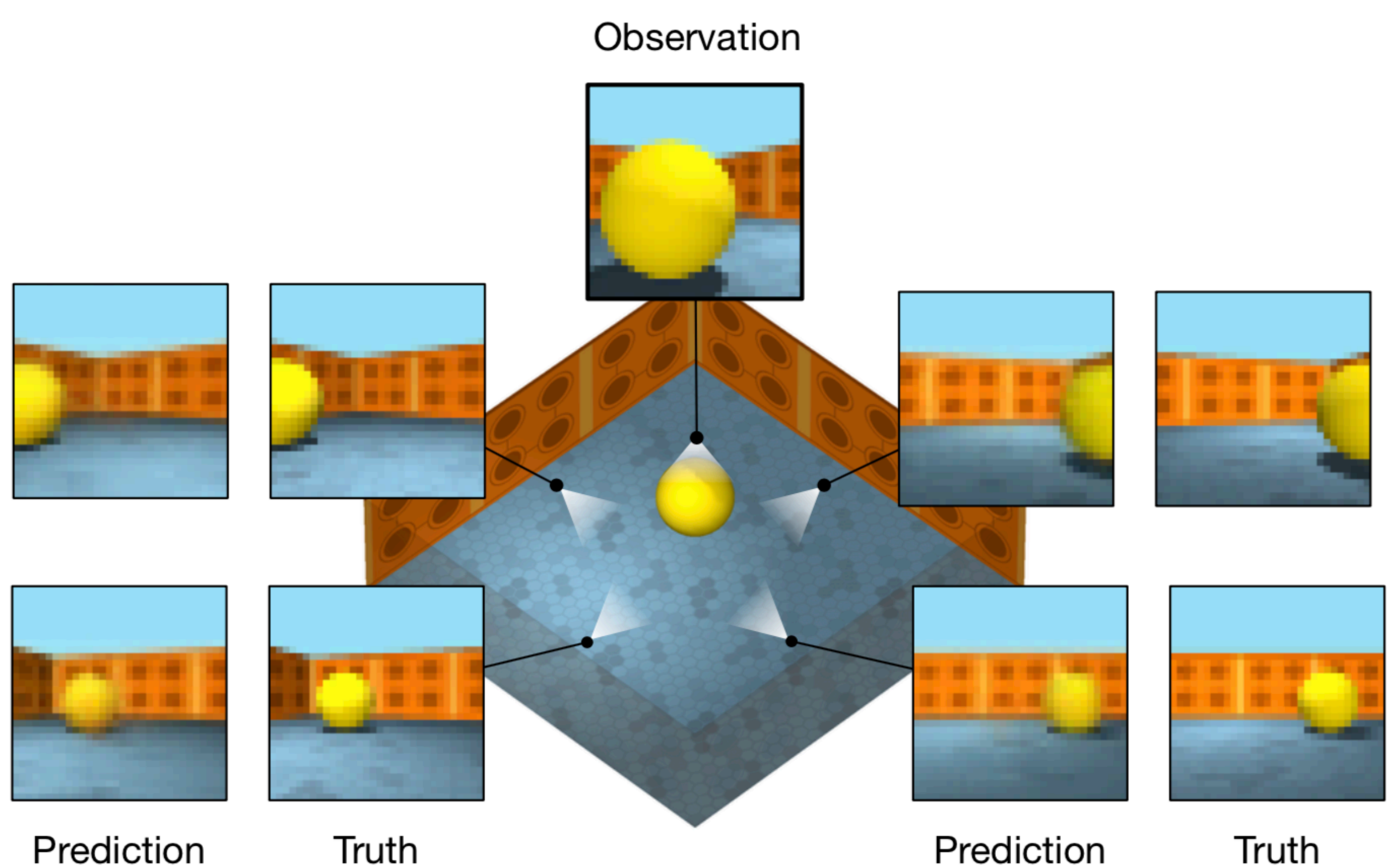


Generative Query Network



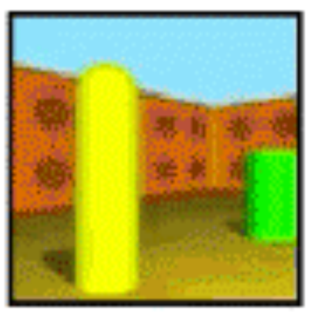



Observation
Prediction
Truth
Prediction
Truth

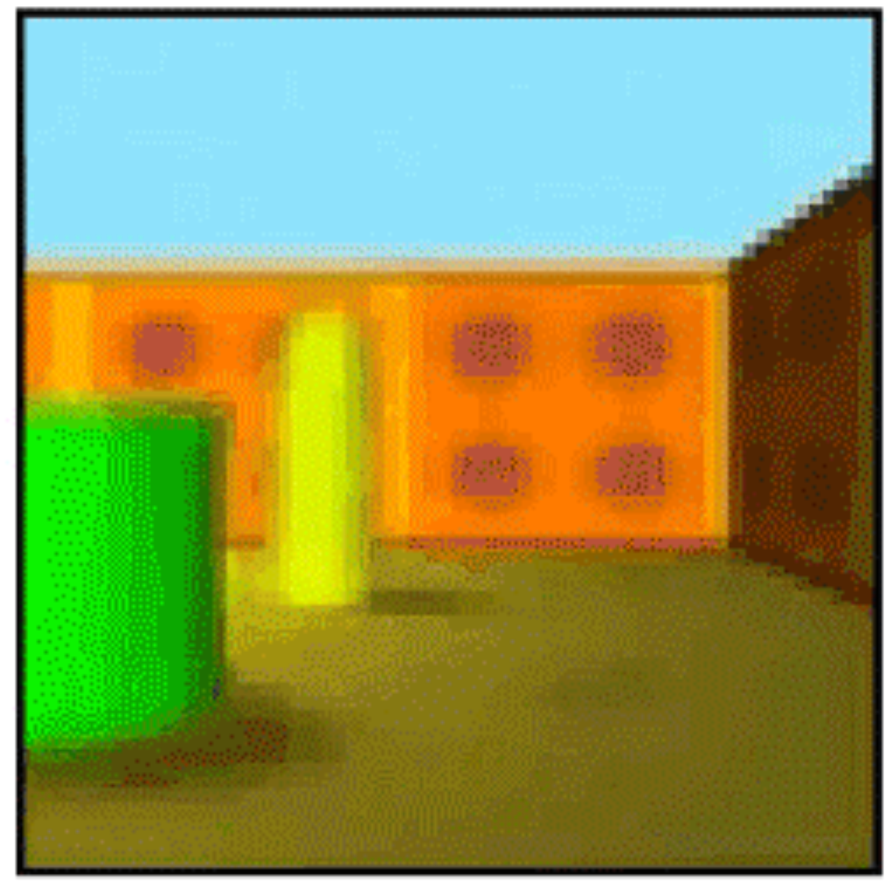

Observation
Prediction
Truth
Prediction
Truth

Scene Rendering

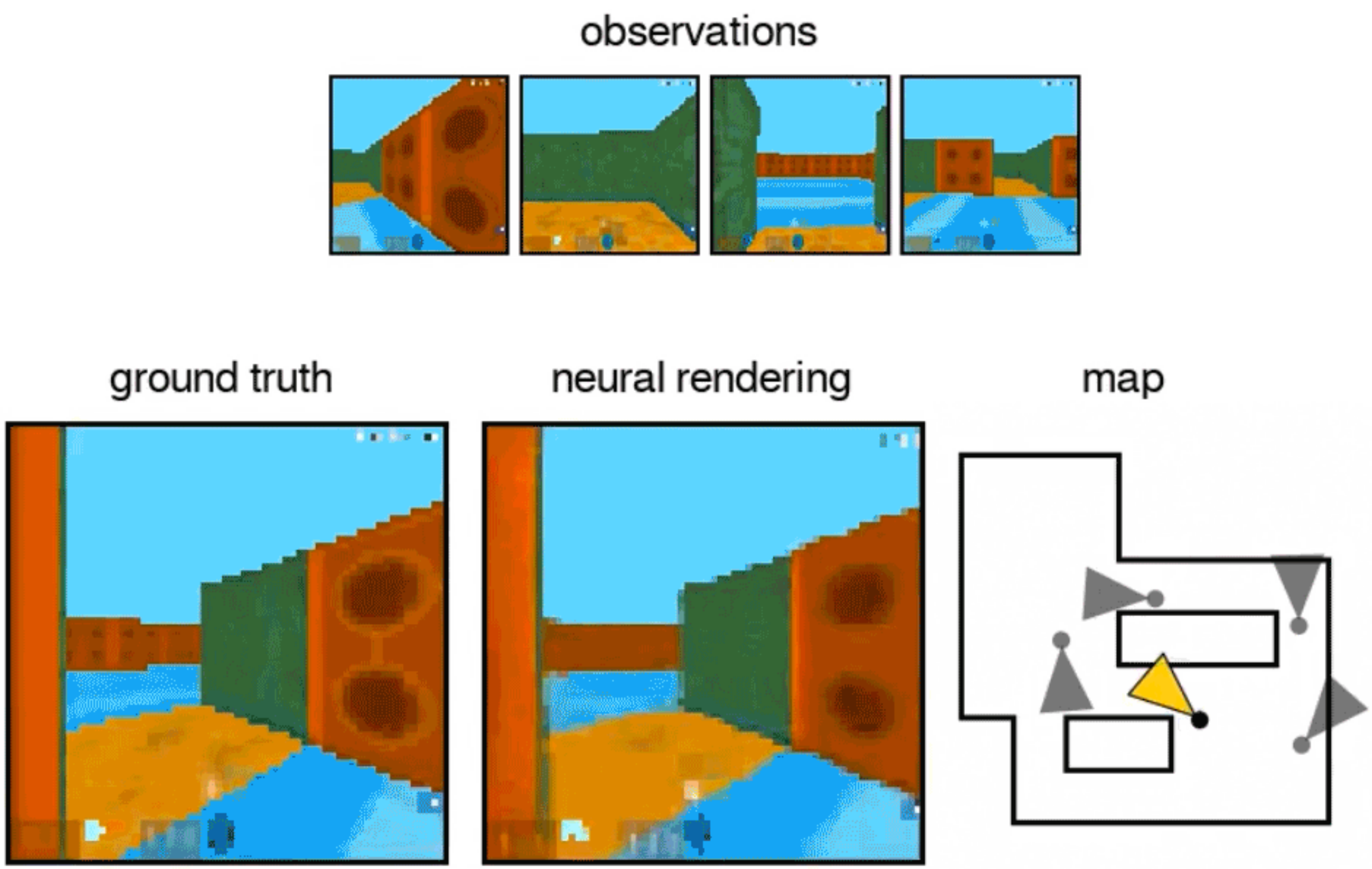
observation



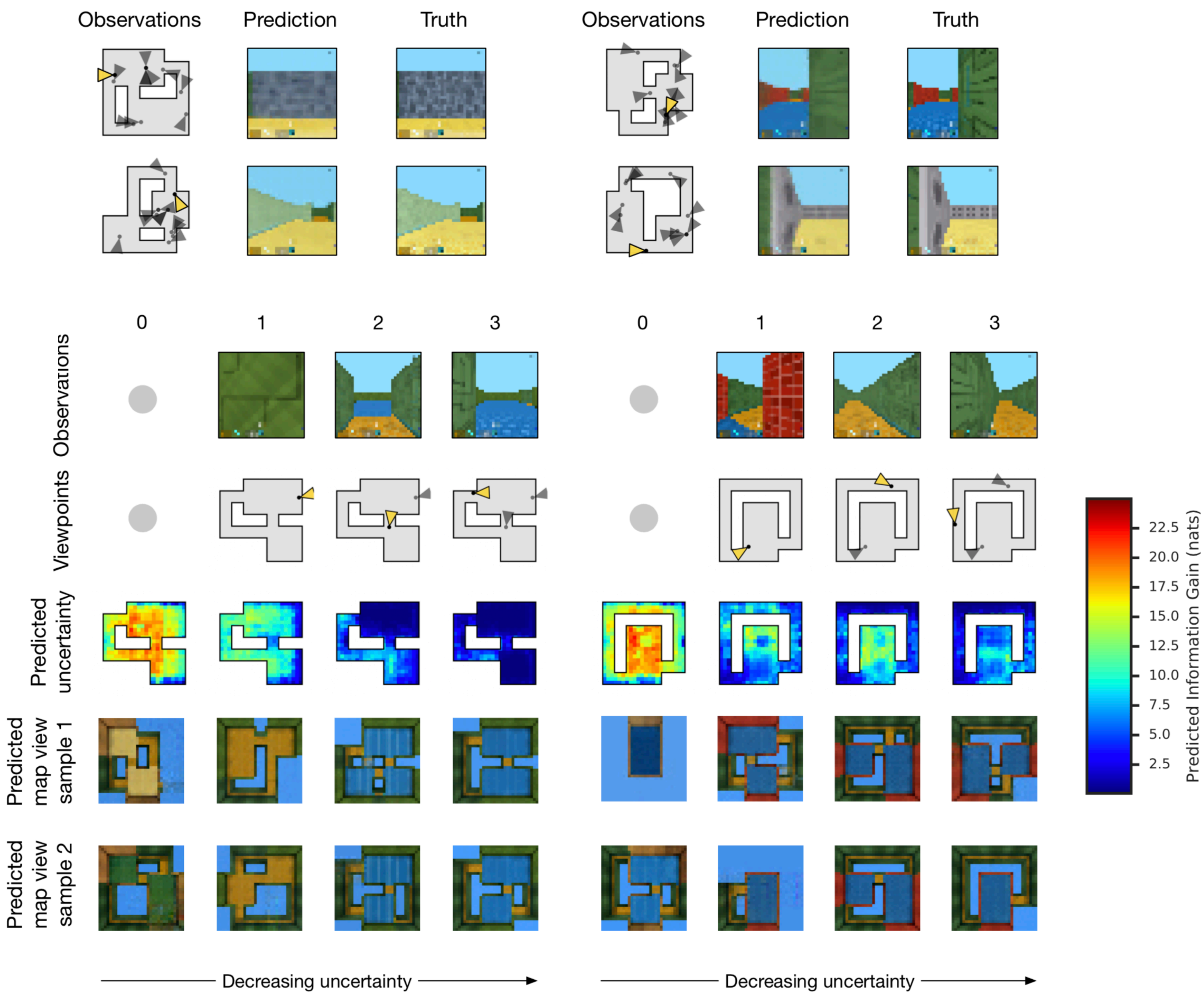
neural rendering



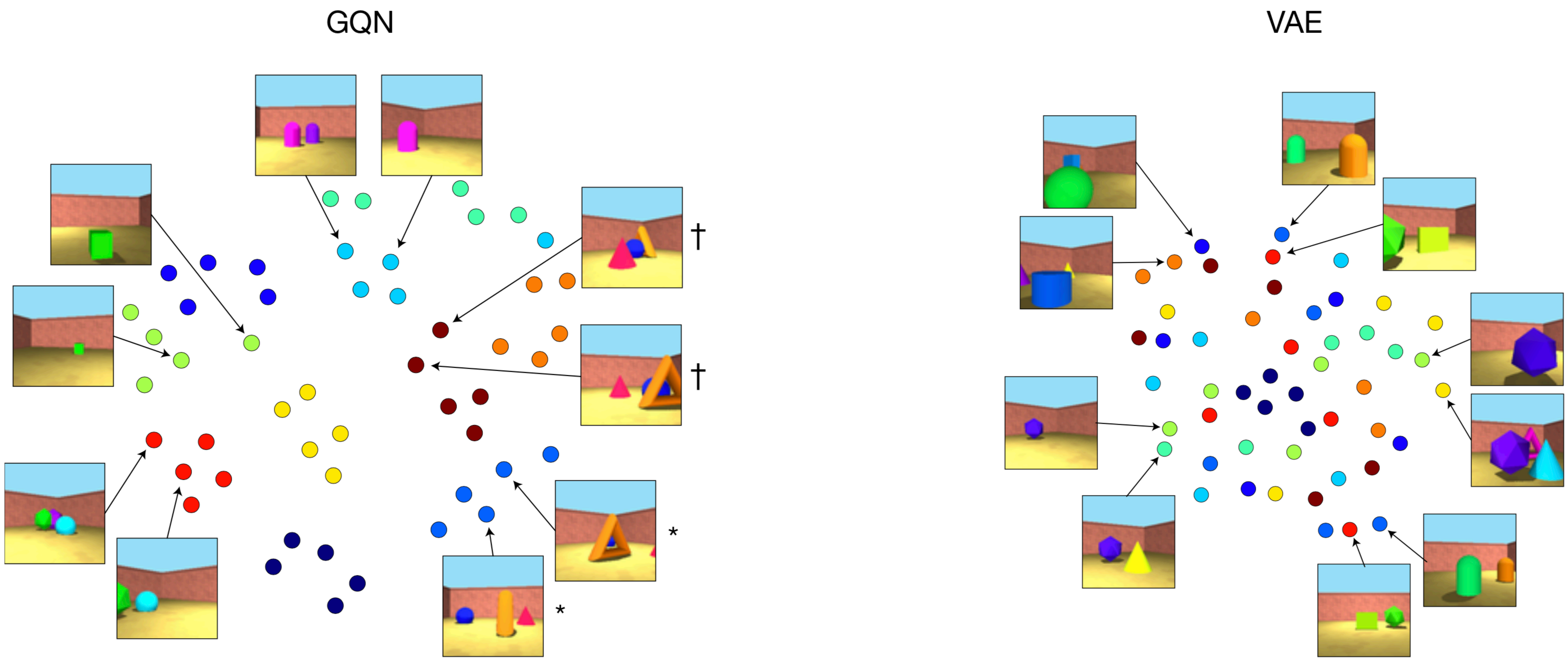
Labyrinth Navigation



Uncertainty in GQN

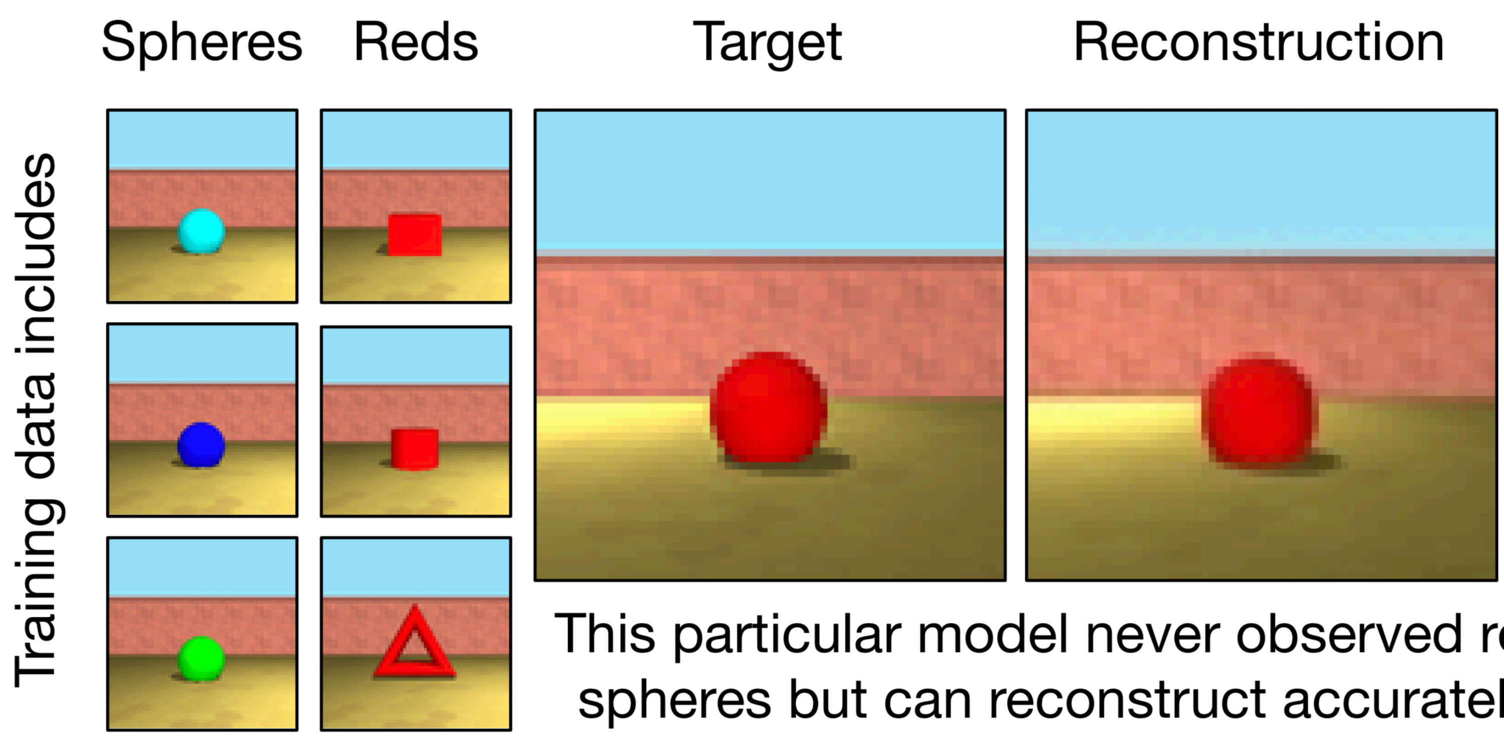


GQN Representations



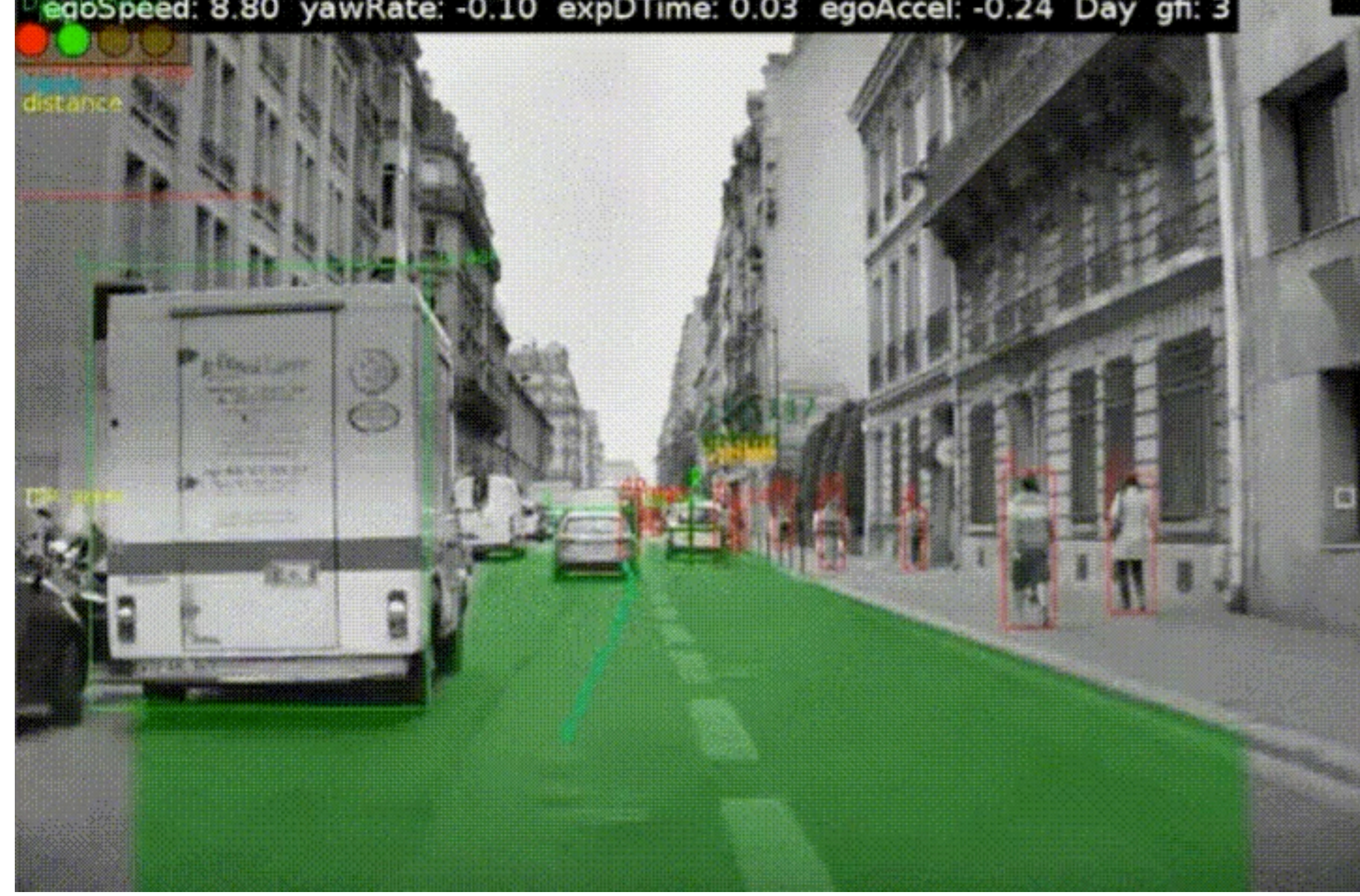
* & † -- Same objects, different positions, clearly separated

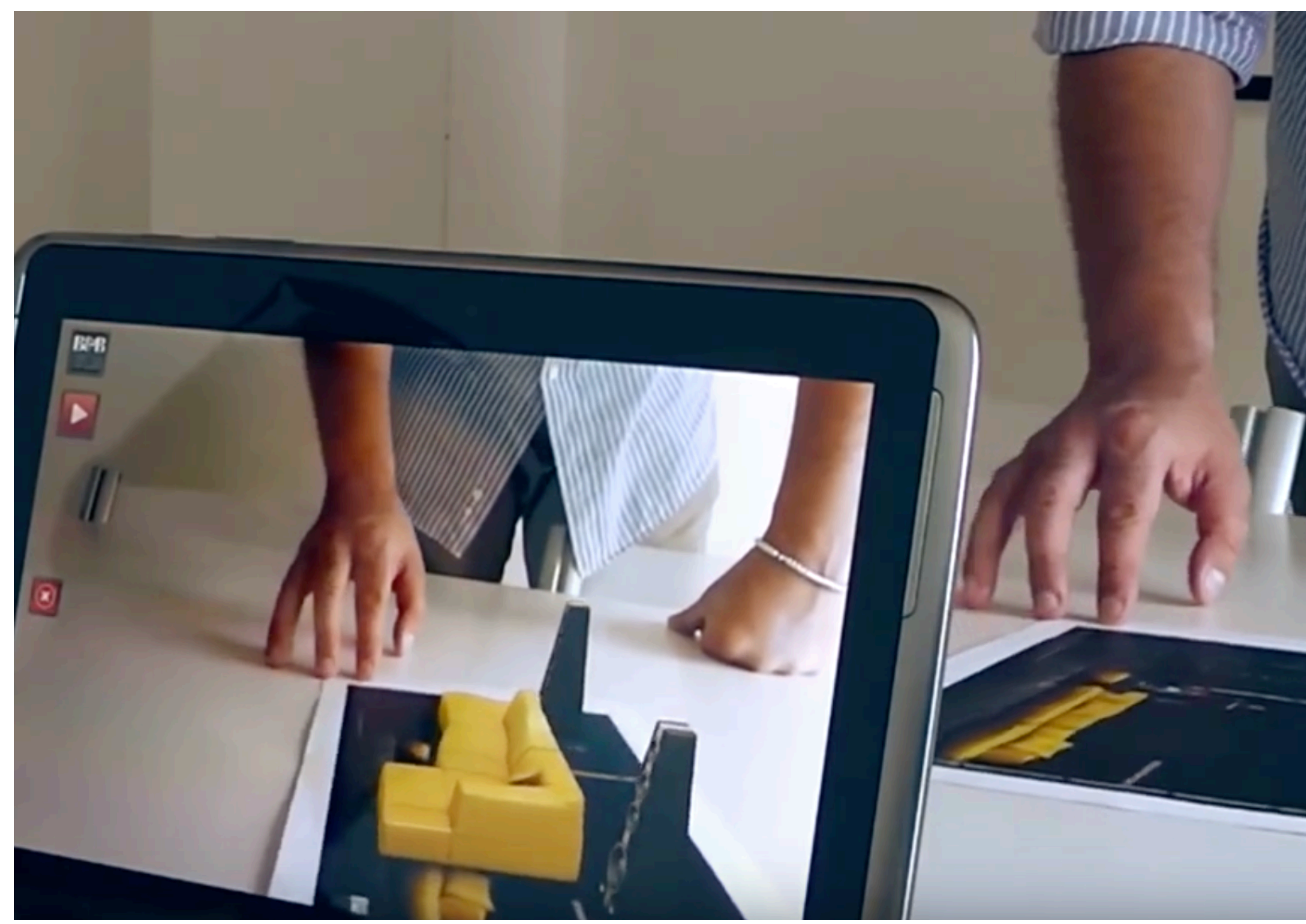
Generalization



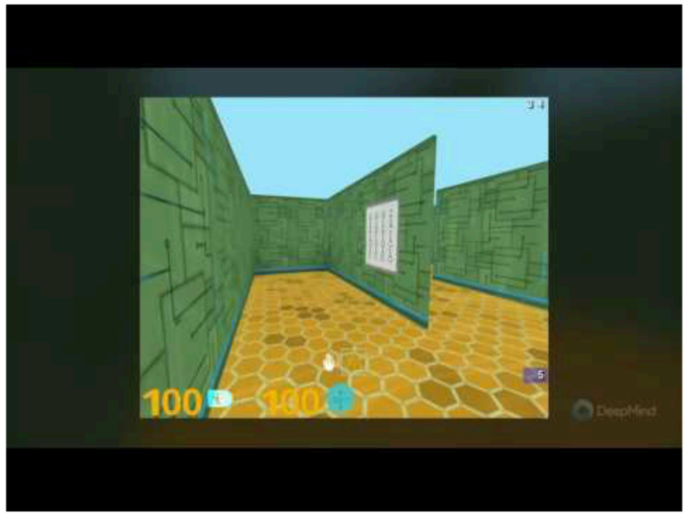
This particular model never observed red spheres but can reconstruct accurately

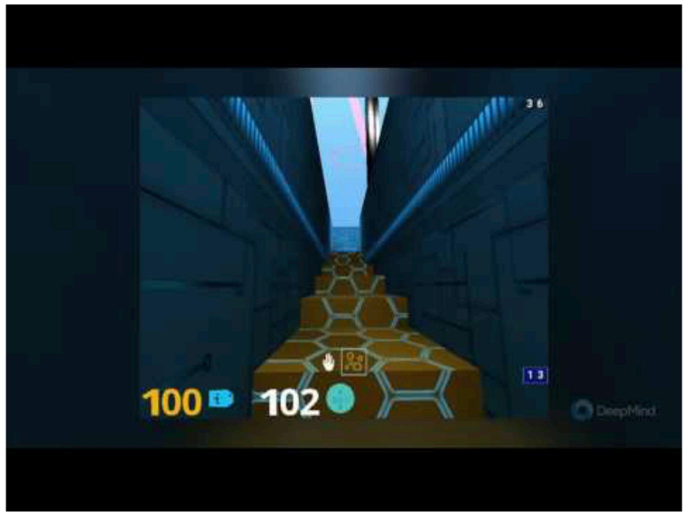
Applications



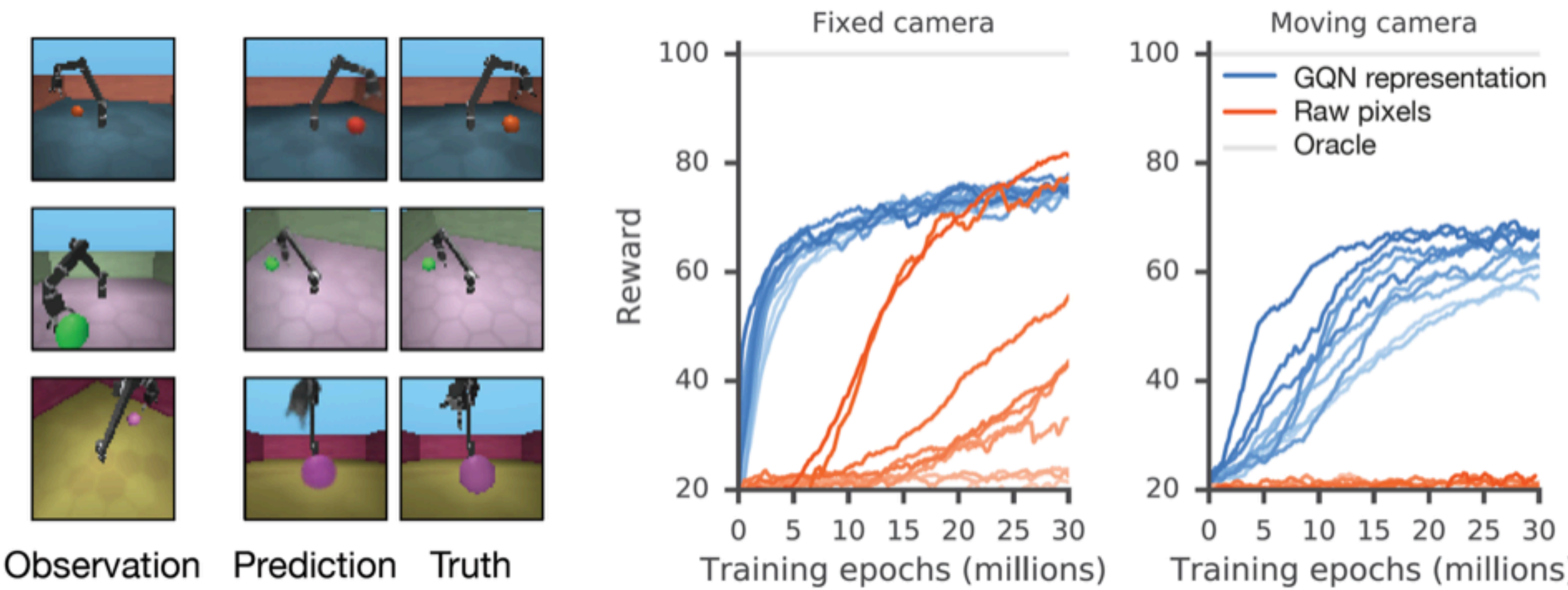


DeepMind Lab





https://github.com/deepmind/lab



Using GQN we observe substantially more data-efficient policy learning, obtaining convergence-level performance with approximately 4 times fewer interactions than a standard method using raw pixels.

https://www.youtube.com/watch?v=RBJFngN33Qo

Discussion

Building up an internal representation using an LSTM is an unusual approach in the image domain. The authors argue that it allows for the representation of much more complex domains. What are the limitations of this approach? Should all current representational models be swapped to the GQN framework?

Discussion

The results are incredibly impressive from a cognitive perspective. Running mazes and mental rotation tasks are canonical experiments in the literature. However, the model is so complex, it is not clear what is actually contributing to what in the model. Consequently, does this model help us understand the problem?

Discussion

What does this mean for game and simulation design? One could imagine a context where a 'game' is just a simulated world learned by a complex model, and 'playing the game' just means simulating different triggers in the model in different contexts. Is this still a game, or does the medium matter?

Discussion

Think of generating text pertaining to the same topic with different "viewpoints"

i.e. from the topic of taxation from a capitalist or socialist viewpoint, what sort of signal representation should we be using?

Could we summarize the representations r like the image signals, simply by adding all the other representations into one single representation?

Appendix

Dataset

https://github.com/deepmind/gqn-datasets observations (2D images & viewpoints)

$$r = f_\theta(\boldsymbol{o}_i)$$

learnable parameters

scene representation

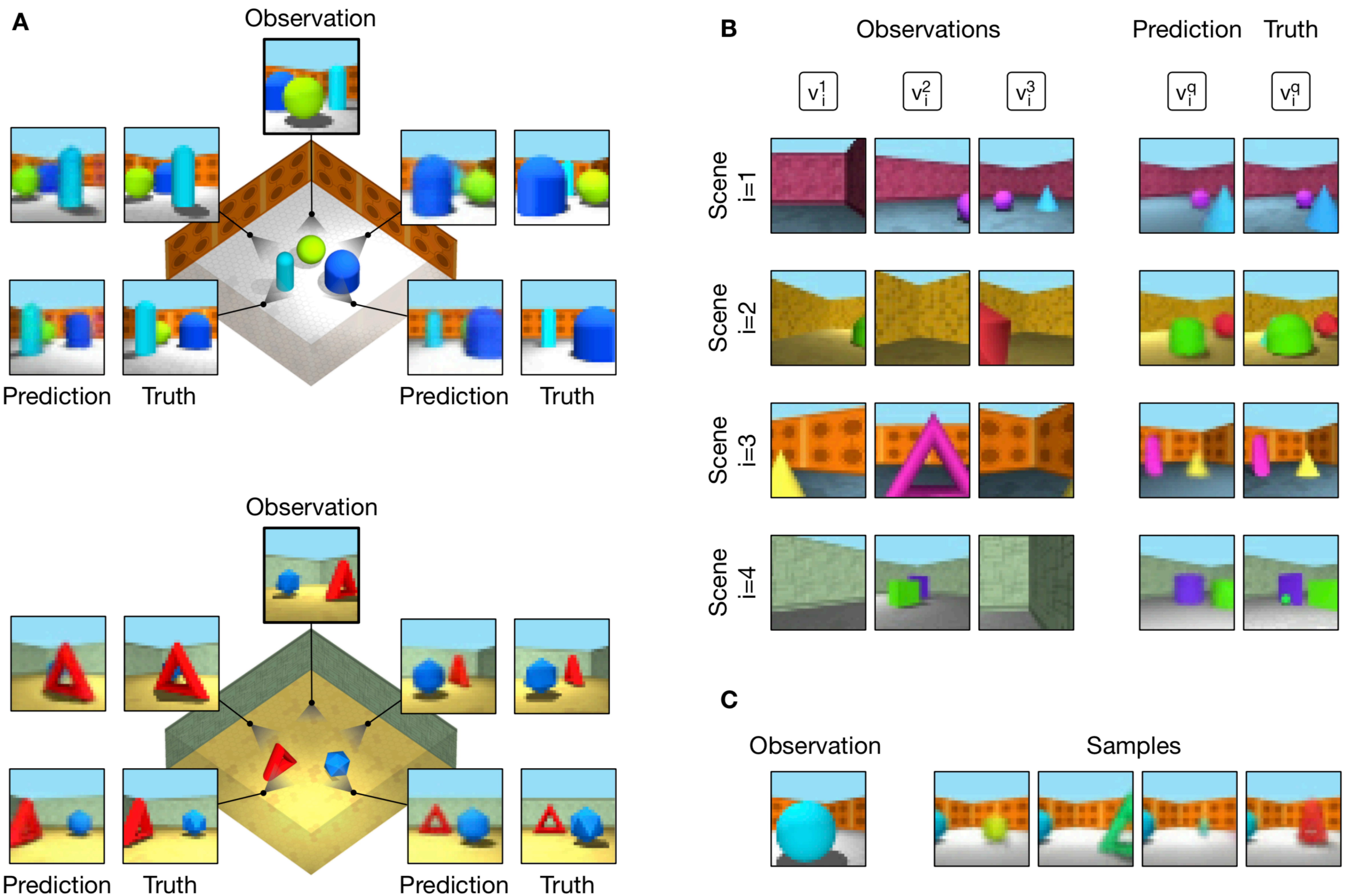
image

$$g_{\theta}(\boldsymbol{x}|\boldsymbol{v}^{q},\boldsymbol{r}) = \int g_{\theta}(\boldsymbol{x},\boldsymbol{z}|\boldsymbol{v}^{q},\boldsymbol{r})\,d\boldsymbol{z}$$

viewpoint

latent variables

learnable parameters


A
Observation
Prediction
Truth
Prediction
Truth
Observation
Prediction
Truth
Prediction
Truth
B
Observations
Prediction
Truth
v_i^1
v_i^2
v_i^3
v_i^q
v_i^q
Scene i=1
Scene i=2
Scene i=3
Scene i=4
C
Observation
Samples

Algorithm S1: GQN training loop.

Data: Choose dataset D from Room, Jaco, Labyrinth or Shepard-Metzler
Input: Initial parameters θ and ϕ. Optimizer parameters μ_i, μ_f, n, S_{max}, σ_i, σ_f, β_1 and β_2.
Output: Learned parameters θ and ϕ

def *SampleBatch*(B, M, K):
    /* Sample number of views */
    $M \sim \text{Uniform}(0, K)$
    /* Initialize data batch */
    $D = \{\}$
    for $b \leftarrow 0$ **in** $(B-1)$:
        /* Sample scene index */
        $i \sim \text{Uniform}(0, N-1)$
        for $k \leftarrow 0$ **in** $(M-1)$:
            /* Sample view */
            $(\mathbf{x}_i^k, \mathbf{v}_i^k) \sim \text{scene } i$
            $D \leftarrow D + \{(\mathbf{x}_i^k, \mathbf{v}_i^k)\}$
        /* Sample query view */
        $(\mathbf{x}_i^q, \mathbf{v}_i^q) \sim \text{scene } i$
        $D \leftarrow D + \{(\mathbf{x}_i^q, \mathbf{v}_i^q)\}$

/* Training Iterations */
for $t \leftarrow 0$ **in** $(S_{max} - 1)$:
    $D \leftarrow \text{SampleBatch}(B, M, K)$
    $\text{ELBO} \leftarrow \text{EstimateELBO}(D, \sigma_t)$ (Algorithm S2)
    /* Compute empirical ELBO gradients */
    $\nabla_\theta\text{ELBO}, \nabla_\phi\text{ELBO} \leftarrow \text{Backprop(ELBO)}$.
    /* Update parameters */
    $\theta, \phi \leftarrow \text{Optimizer}(\nabla_\theta\text{ELBO}, \nabla_\phi\text{ELBO}, \mu_t)$
    /* Update optimizer state */
    $\mu_t \leftarrow \max\left(\mu_f + (\mu_i - \mu_f)\left(1 - \frac{t}{n}\right), \mu_f\right)$
    /* Pixel-variance annealing */
    $\sigma_t \leftarrow \max\left(\sigma_f + (\sigma_i - \sigma_f)\left(1 - \frac{t}{n}\right), \sigma_f\right)$

Algorithm S2: Generating a sample from the approximate variational GQN posterior and estimating the ELBO.

Input: Observed views $\{(\mathbf{x}^k, \mathbf{v}^k)\}$, query camera: $\mathbf{v}^q$, target image: $\mathbf{x}^q$, pixel-variance: σ_t

Output: Sample from the posterior $\mathbf{z} \sim q_\phi\left(\mathbf{z}|\mathbf{x}^q, \mathbf{v}^q, \mathbf{r}\right)$, empirical estimate of the ELBO

def *EstimateELBO*$(\{(\mathbf{x}^k, \mathbf{v}^k)\}, (\mathbf{v}^q, \mathbf{x}^q), \sigma_t)$**:**

  Output: Empirical estimate of the ELBO

  `/* Scene encoder */`

  $\mathbf{r} \leftarrow 0$

  for $k \leftarrow 0$ **in** $(M-1)$**:**

    $\hat{\mathbf{v}}^k \leftarrow (\mathbf{w}^k, \cos(\mathbf{y}^k), \sin(\mathbf{y}^k), \cos(\mathbf{p}^k), \sin(\mathbf{p}^k))$

    $\mathbf{r}^k \leftarrow \psi\left(\mathbf{x}^k, \hat{\mathbf{v}}^k\right)$

    $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{r}^k$

  `/* Generator initial state */`

  $(\mathbf{c}_0^g, \mathbf{h}_0^g, \mathbf{u}_0) \leftarrow (\mathbf{0}, \mathbf{0}, \mathbf{0})$

  `/* Inference initial state */`

  $(\mathbf{c}_0^e, \mathbf{h}_0^e) \leftarrow (\mathbf{0}, \mathbf{0})$

  $\text{ELBO} \leftarrow 0$

  for $l \leftarrow 0$ **in** $(L-1)$**:**

    `/* Prior factor */`

    $\pi_{\theta_l}\left(\cdot|\mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right) \leftarrow \mathcal{N}\left(\cdot|\eta_\theta^\pi\left(\mathbf{h}_l^g\right)\right)$

    `/* Inference state update */`

    $\left(\mathbf{c}_{l+1}^e, \mathbf{h}_{l+1}^e\right) \leftarrow C_\phi^e\left(\mathbf{x}^q, \mathbf{v}^q, \mathbf{r}, \mathbf{c}_l^e, \mathbf{h}_l^e, \mathbf{h}_l^g, \mathbf{u}_l\right)$

    `/* Posterior factor */`

    $q_{\phi_l}\left(\cdot|\mathbf{x}^q, \mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right) \leftarrow \mathcal{N}\left(\cdot|\eta_\theta^e\left(\mathbf{h}_l^e\right)\right)$

    `/* Posterior sample */`

    $\mathbf{z}_l \sim q_{\phi_l}\left(\cdot|\mathbf{x}^q, \mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right)$

    `/* Generator state update */`

    $\left(\mathbf{c}_{l+1}^g, \mathbf{h}_{l+1}^g, \mathbf{u}_{l+1}\right) \leftarrow C_\theta^g\left(\mathbf{v}^q, \mathbf{r}, \mathbf{c}_l^g, \mathbf{h}_l^g, \mathbf{u}_l\right)$

    `/* ELBO KL contribution update */`

    $\text{ELBO} \leftarrow \text{ELBO} - \text{KL}\left[q_{\phi_l}\left(\cdot|\mathbf{x}^q, \mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right) || \pi_{\theta_l}\left(\cdot|\mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right)\right]$

  `/* ELBO likelihood contribution update */`

  $\text{ELBO} \leftarrow \text{ELBO} + \log \mathcal{N}\left(\mathbf{x}^q \middle| \mu = \eta_\theta^g(\mathbf{u}_L), \sigma = \sigma_t\right)$

Algorithm S3: Generating a prediction from GQN.

def *Generate*$(\{(\mathbf{x}^k, \mathbf{v}^k)\}, \mathbf{v}^q)$**:**

    Output: Generated image sample $\hat{\mathbf{x}}^q$

    /* Scene encoder */
    $\mathbf{r} \leftarrow 0$
    for $k \leftarrow 0$ **in** $(M-1)$**:**
        $\hat{\mathbf{v}}^k \leftarrow (\mathbf{w}^k, \cos(\mathbf{y}^k), \sin(\mathbf{y}^k), \cos(\mathbf{p}^k), \sin(\mathbf{p}^k))$
        $\mathbf{r}^k \leftarrow \psi\left(\mathbf{x}^k, \hat{\mathbf{v}}^k\right)$
        $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{r}^k$
    /* Initial state */
    $(\mathbf{c}_0^g, \mathbf{h}_0^g, \mathbf{u}_0) \leftarrow (\mathbf{0}, \mathbf{0}, \mathbf{0})$
    for $l \leftarrow 0$ **in** $(L-1)$**:**
        /* Prior factor */
        $\pi_{\theta_l}\left(\cdot \middle| \mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right) \leftarrow \mathcal{N}\left(\cdot \middle| \eta_\theta^\pi\left(\mathbf{h}_l^g\right)\right)$
        /* Prior sample */
        $\mathbf{z}_l \sim \pi_{\theta_l}\left(\cdot \middle| \mathbf{v}^q, \mathbf{r}, \mathbf{z}_{<l}\right)$
        /* State update */
        $\left(\mathbf{c}_{l+1}^g, \mathbf{h}_{l+1}^g, \mathbf{u}_{l+1}\right) \leftarrow C_\theta^g\left(\mathbf{v}^q, \mathbf{r}, \mathbf{c}_l^g, \mathbf{h}_l^g, \mathbf{u}_l, \mathbf{z}_l\right)$
    /* Image sample */
    $\hat{\mathbf{x}}^q \sim \mathcal{N}\left(\mathbf{x}^q \middle| \mu = \eta_\theta^g(\mathbf{u}_L), \sigma = \sigma_t\right)$

Name	Description	Values
μ_s	Learning rate at training step s with annealing $\mu_s = \max\left(\mu_f + (\mu_i - \mu_f)\left(1 - \frac{s}{n}\right), \mu_f\right)$	$\mu_i = 5 \times 10^{-4}$ $\mu_f = 5 \times 10^{-5}$ $n = 1.6 \times 10^{6}$
γ_s	Learning rate as used by the Adam algorithm $\gamma_s = \mu_s \frac{\sqrt{1-\beta_2^s}}{1-\beta_1^s}$	$\beta_1 = 0.9$ $\beta_2 = 0.999$
ϵ	Adam regularisation parameter	$\epsilon = 10^{-8}$
σ_s	Pixel standard-deviation with annealing $\sigma_s = \max\left(\sigma_f + (\sigma_i - \sigma_f)\left(1 - \frac{s}{n}\right), \sigma_f\right)$	$\sigma_i = 2.0$ $\sigma_f = 0.7$ $n = 2 \times 10^{5}$
L	Number of generative layers	12
B	Number of scenes over which each weight update is computed	36
S_{max}	Maximum number of training steps	2×10^{6}

Table S1: **List of hyper-parameters.** The values of all hyper-parameters were selected by performing informal search. We did not perform a systematic grid search owing to the high computational cost.