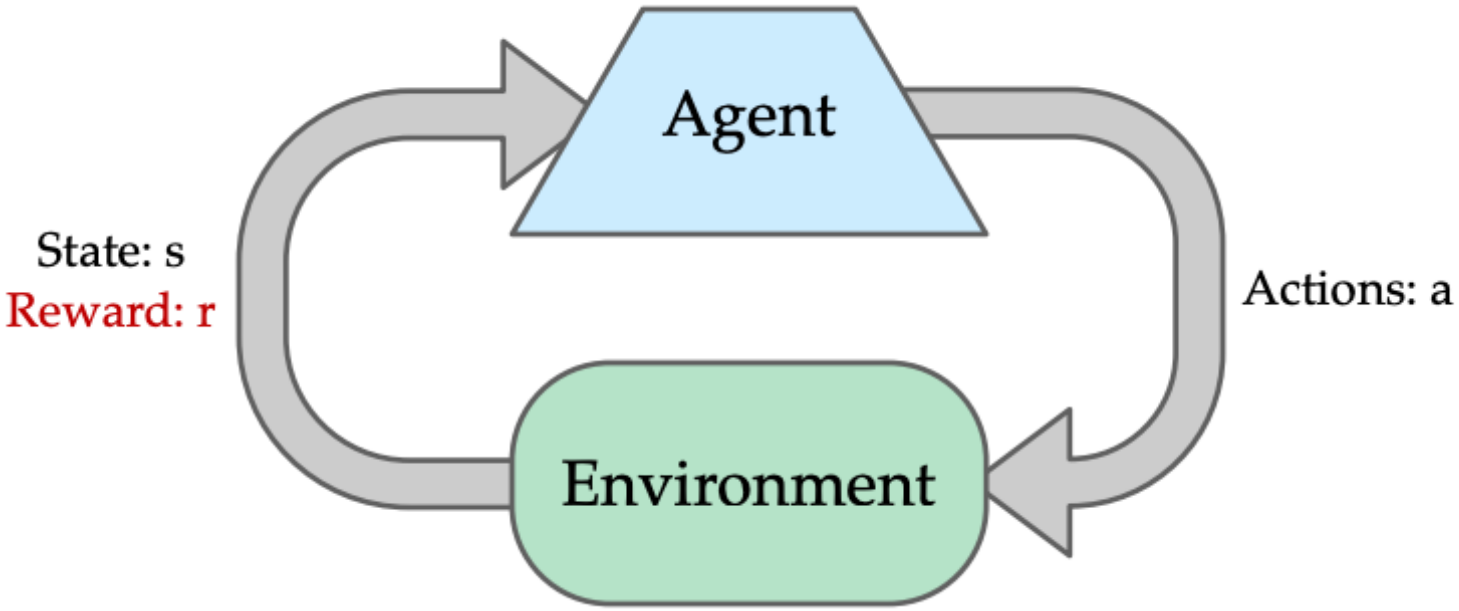
MAX-PLANCK-INSTITUT
FÜR INTELLIGENTE SYSTEME

Extending Intrinsically Motivated Reinforcement Learning to Real Robots

Shukrullo Nazirjonov

Advised by: Cansu Sancaktar, Marco Bagatella, Georg Martius

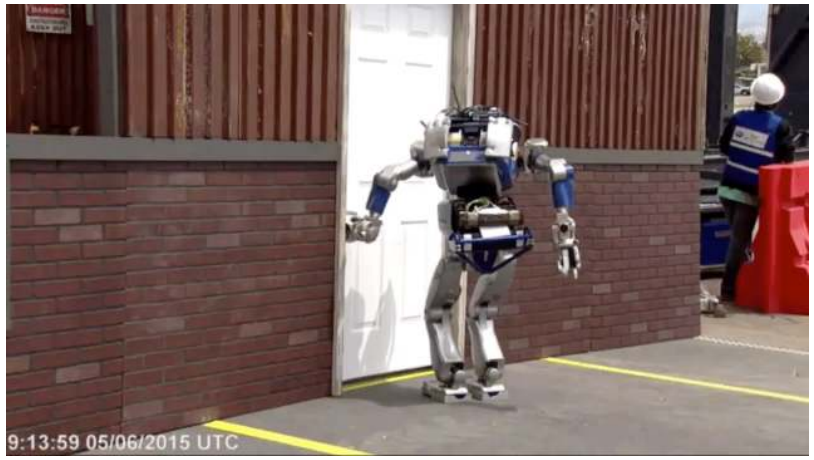
What is Reinforcement Learning (RL)?



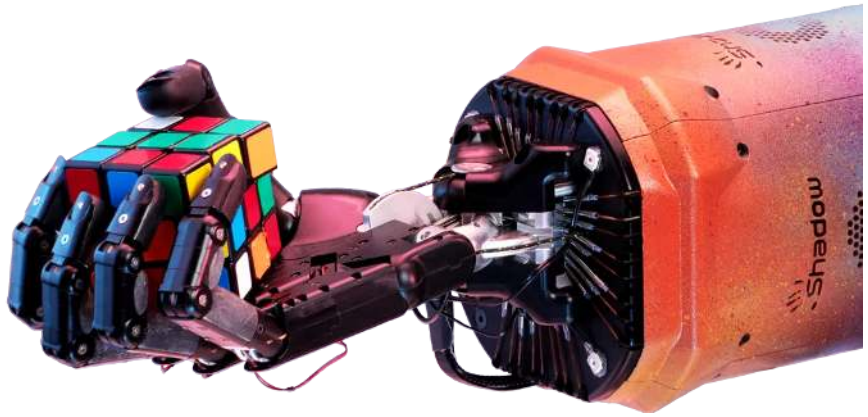
Why use RL for robotics?

Some famous examples from
DARPA robotics challenge (2015)

Recent RL applications in robotics



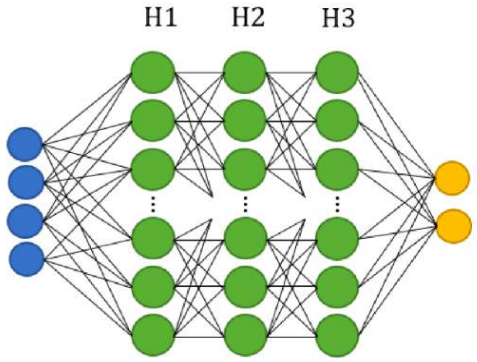
Some famous examples from
DARPA robotics challenge (2015)



Solving Rubik's cube
OpenAI, 2019

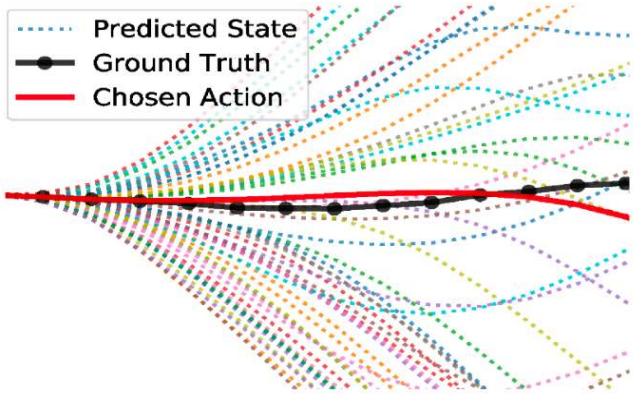
Anymal robot
ETH Zurich, 2020

Model-Based Reinforcement Learning



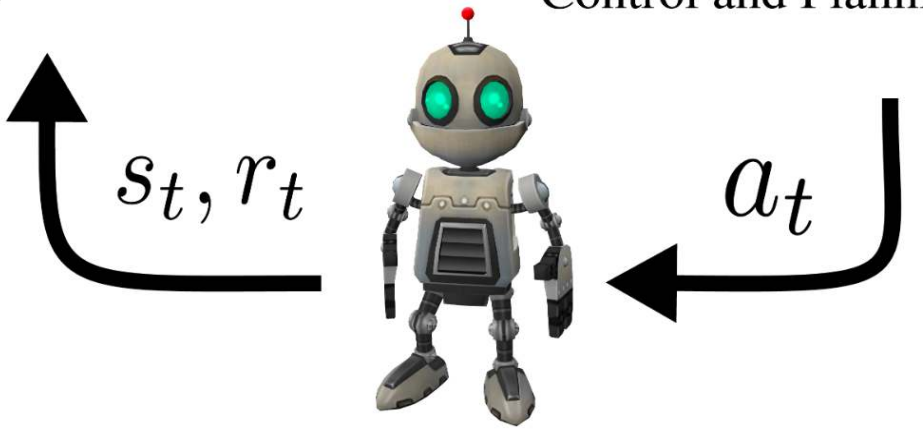
$$s_{t+1} = f_\theta(s_t, a_t)$$

Dynamics Model



$$\arg\max_{u(t)} \sum_{t=0}^{T} r(s_t, a_t)$$

Control and Planning

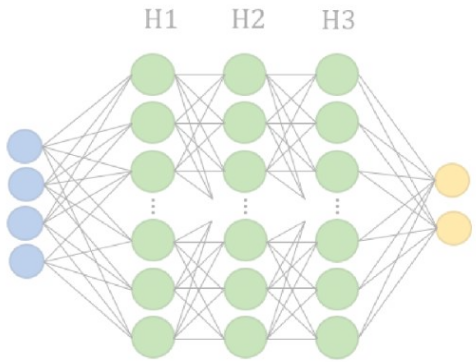


While improving:

- agent acts in environment
- learns model of dynamics
- plans actions to maximize reward

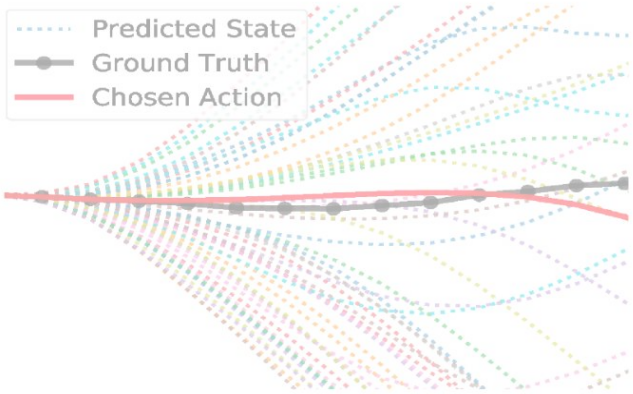
Credits: Nathan Lambert
https://www.natolambert.com/research

Model-Based Reinforcement Learning



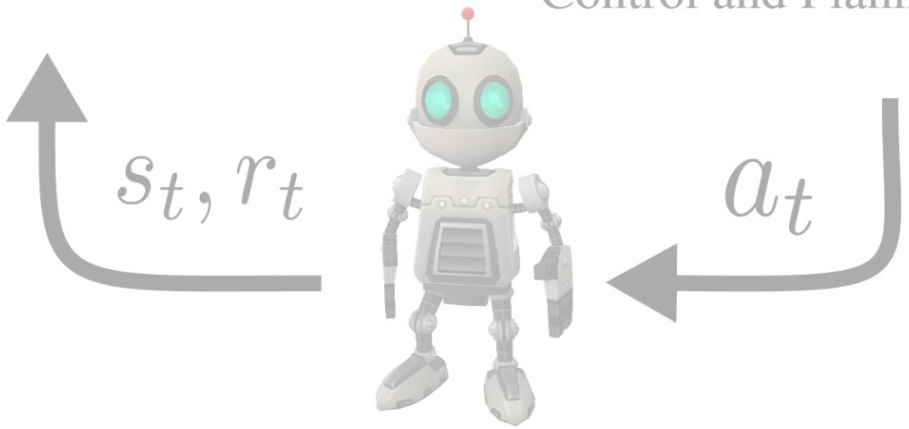
$$s_{t+1} = f_\theta(s_t, a_t)$$

Dynamics Model



$$\arg\max_{u(t)} \sum_{t=0}^{T} r(s_t, a_t)$$

Control and Planning



While improving:

- agent acts in environment
- learns model of dynamics
- plans actions to maximize reward

Extrinsic reward?

Intrinsic reward?

Credits: Nathan Lambert
https://www.natolambert.com/research

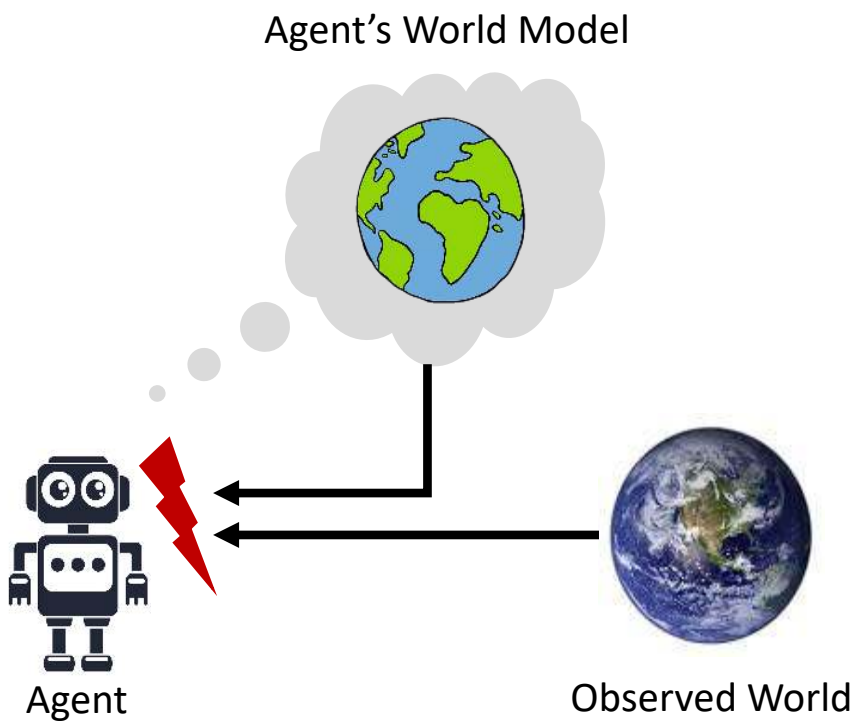
Curious Free Play

Curious Free Play

Can we emulate this free play behavior with robots?

Curious Exploration in Reinforcement Learning

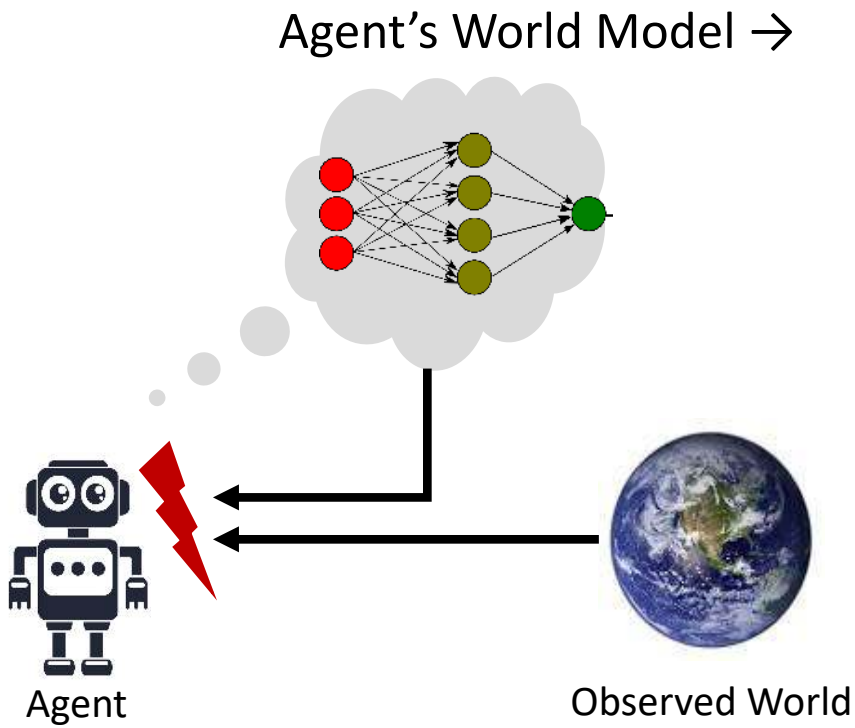
- Novelty as intrinsic reward



- visit novel *states* s
- observe novel state *transitions*

Curious Exploration in Reinforcement Learning

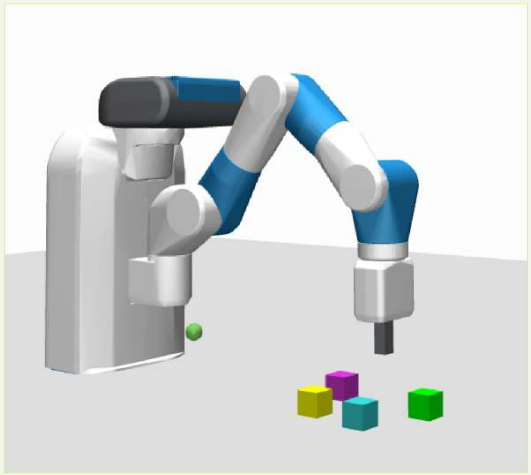
- Novelty as intrinsic reward



Ensemble of neural networks

Prior work: Intrinsic Free Play with CEE-US

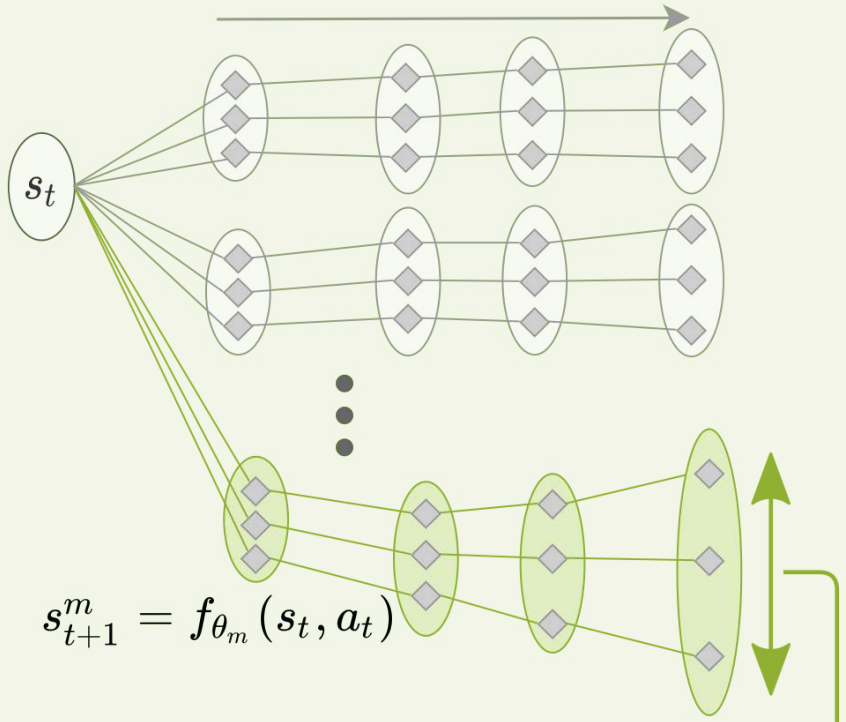
CEE-US: **C**urious **E**xploration using **E**pistemic **U**ncertainty via **S**tructured World Models



Observation

$$\begin{bmatrix} q_{robot} \\ \dot{q}_{robot} \\ q_{obj_1} \\ \dot{q}_{obj_1} \\ \vdots \\ q_{obj_N} \\ \dot{q}_{obj_N} \end{bmatrix} = s_t$$

Planning in imagination with intrinsic rewards

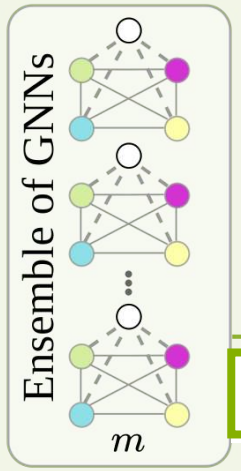


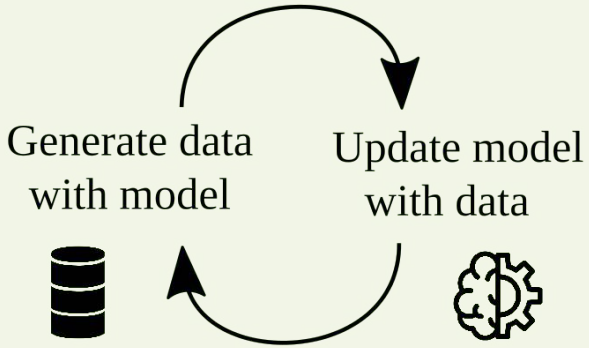
$$s^m_{t+1} = f_{\theta_m}(s_t, a_t)$$

Maximize ensemble disagreement

$$R_{\mathrm{I}}(s_t, a_t) = \mathrm{tr}\big(\mathrm{Cov}(\{\hat{s}^m_{t+1} = f_{\theta_m}(s_t, a_t) \mid m = 1, \ldots, M\})\big)$$

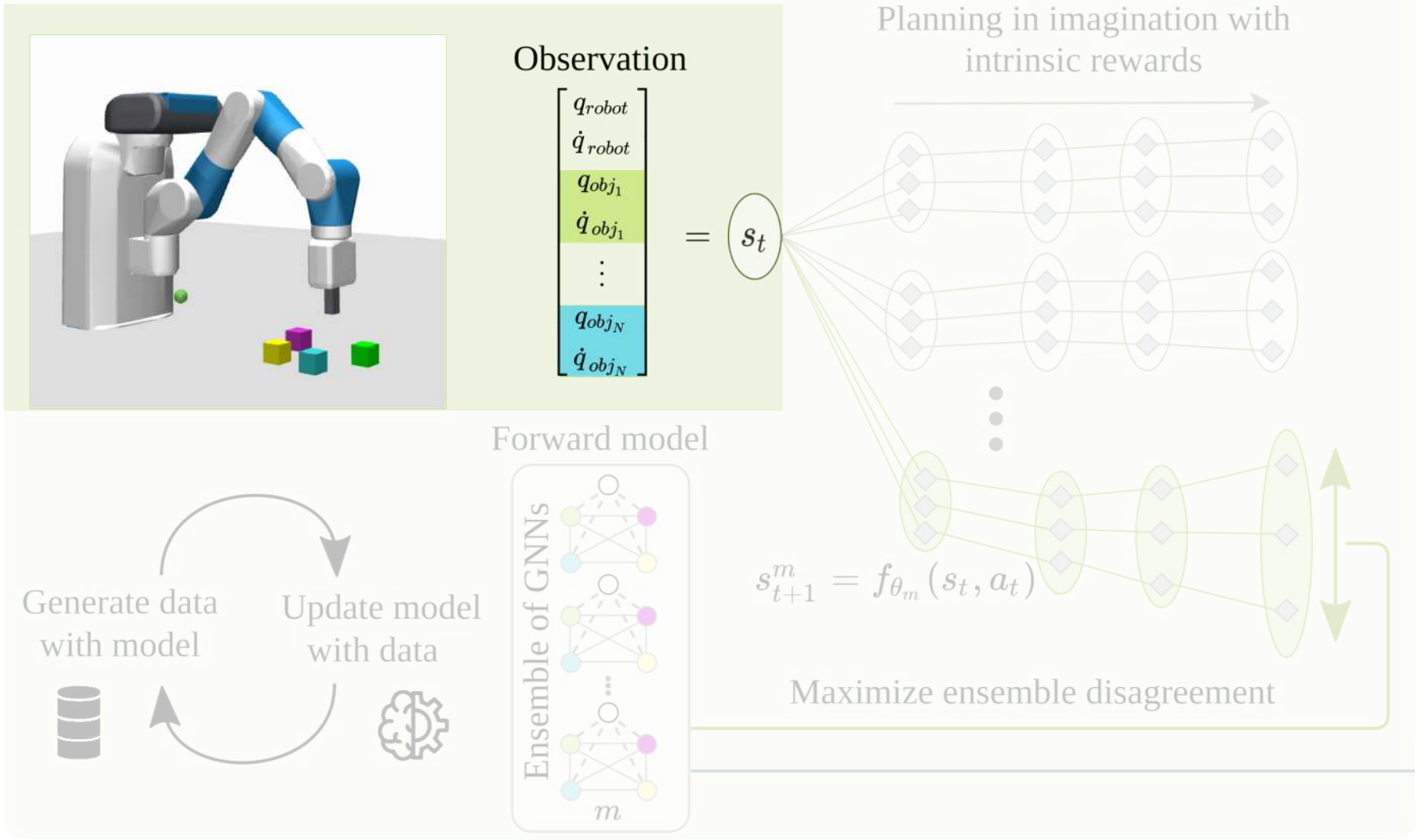
Forward model





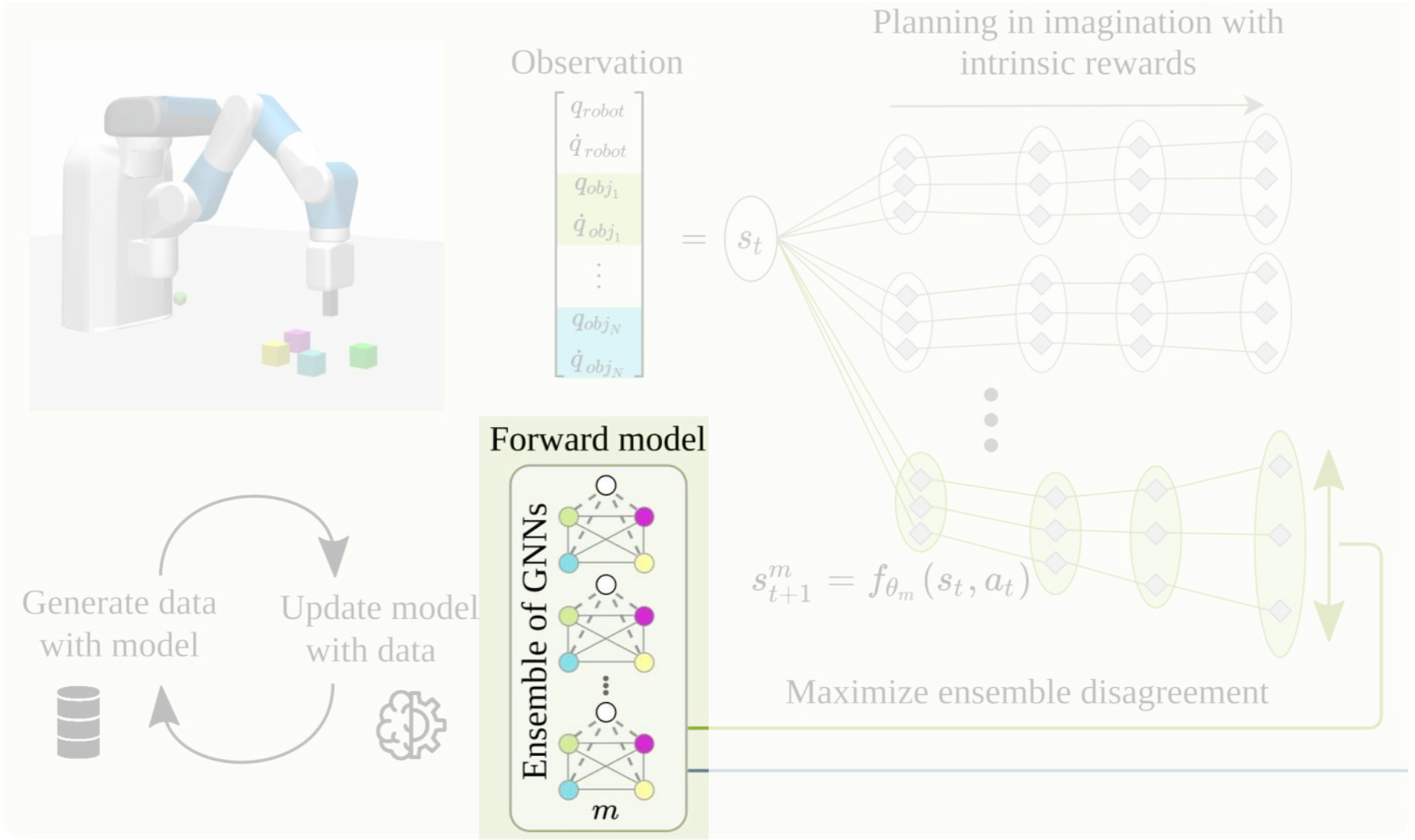
Prior work: Intrinsic Free Play with CEE-US

CEE-US: **C**urious **E**xploration using **E**pistemic **U**ncertainty via **S**tructured World Models



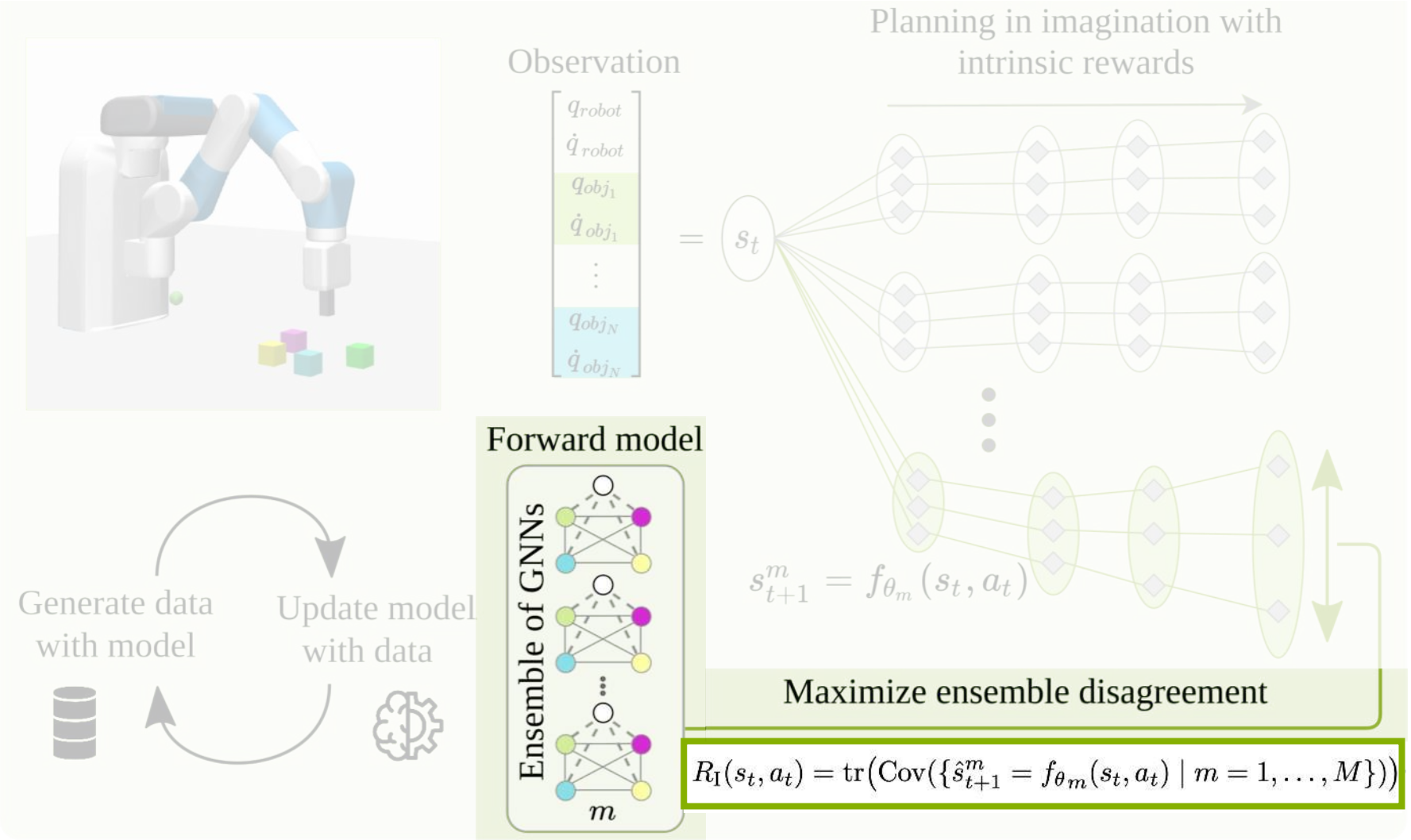
Prior work: Intrinsic Free Play with CEE-US

CEE-US: **C**urious **E**xploration using **E**pistemic **U**ncertainty via **S**tructured World Models



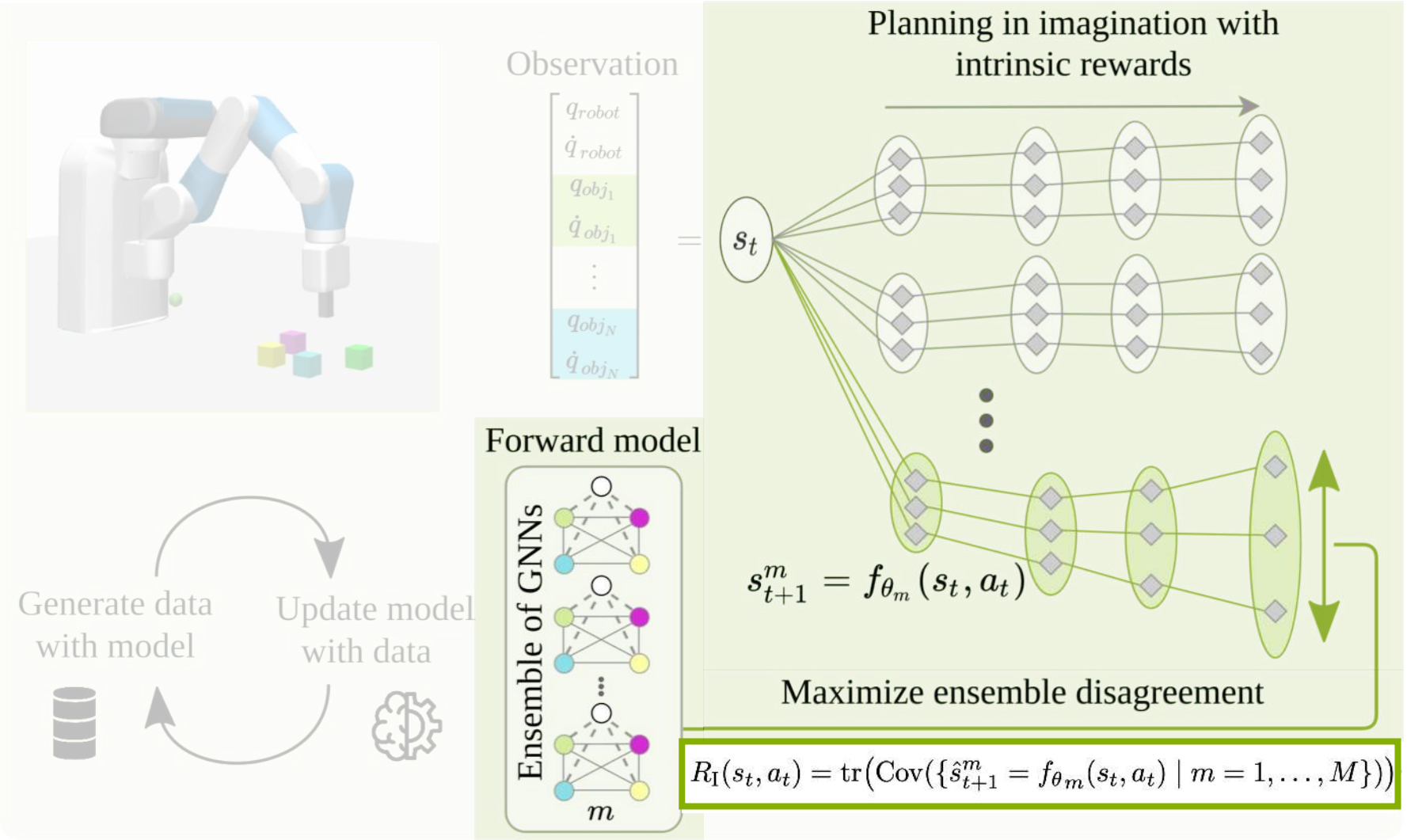
Prior work: Intrinsic Free Play with CEE-US

CEE-US: **C**urious **E**xploration using **E**pistemic **U**ncertainty via **S**tructured World Models



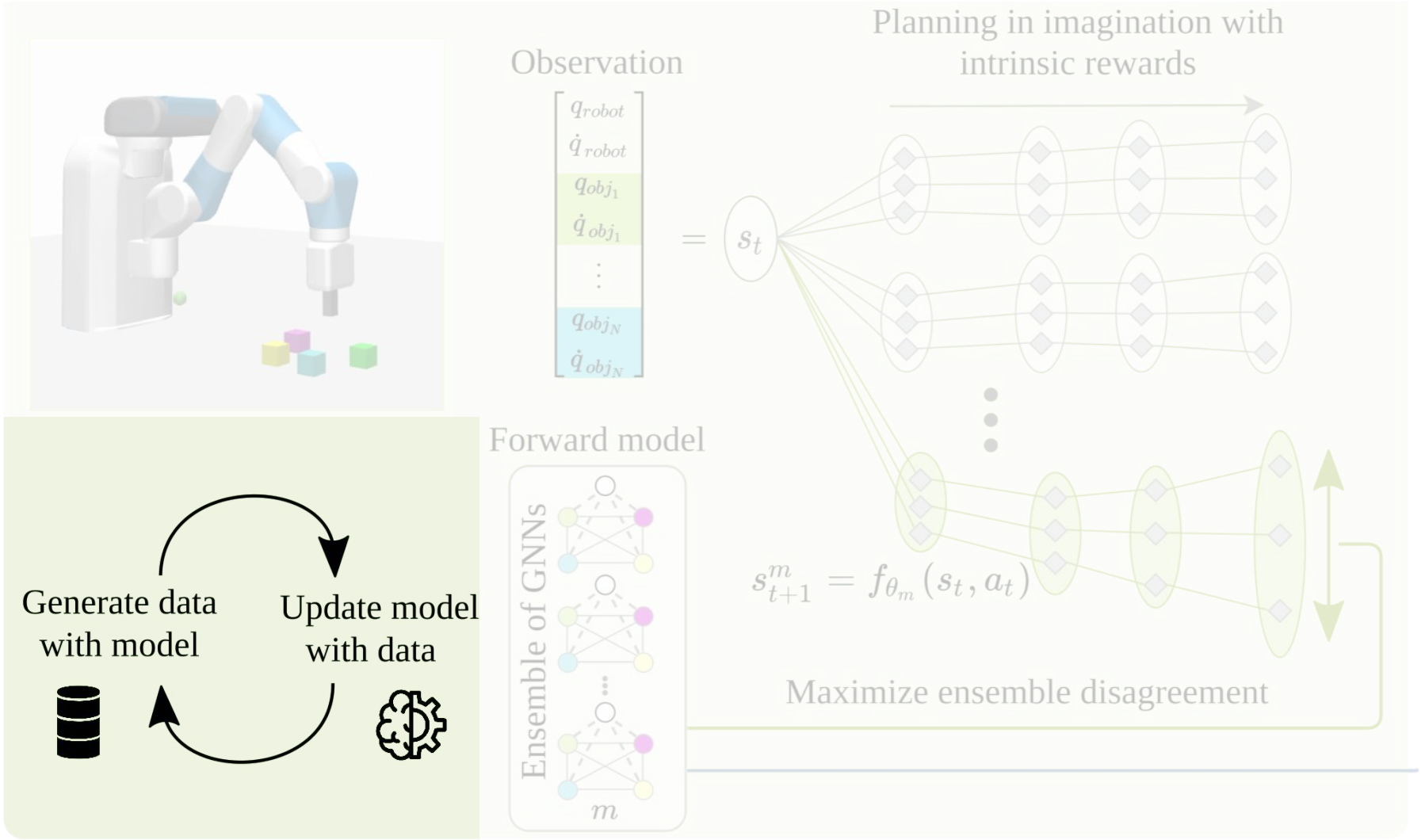
Prior work: Intrinsic Free Play with CEE-US

CEE-US: **C**urious **E**xploration using **E**pistemic **U**ncertainty via **S**tructured World Models

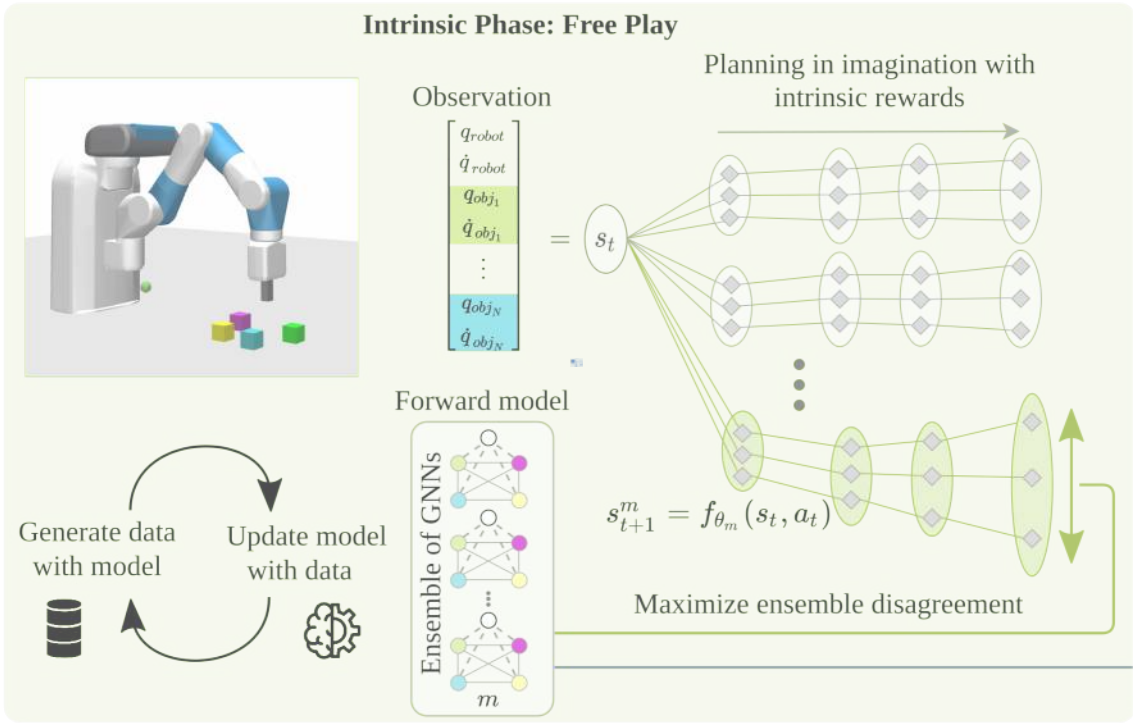


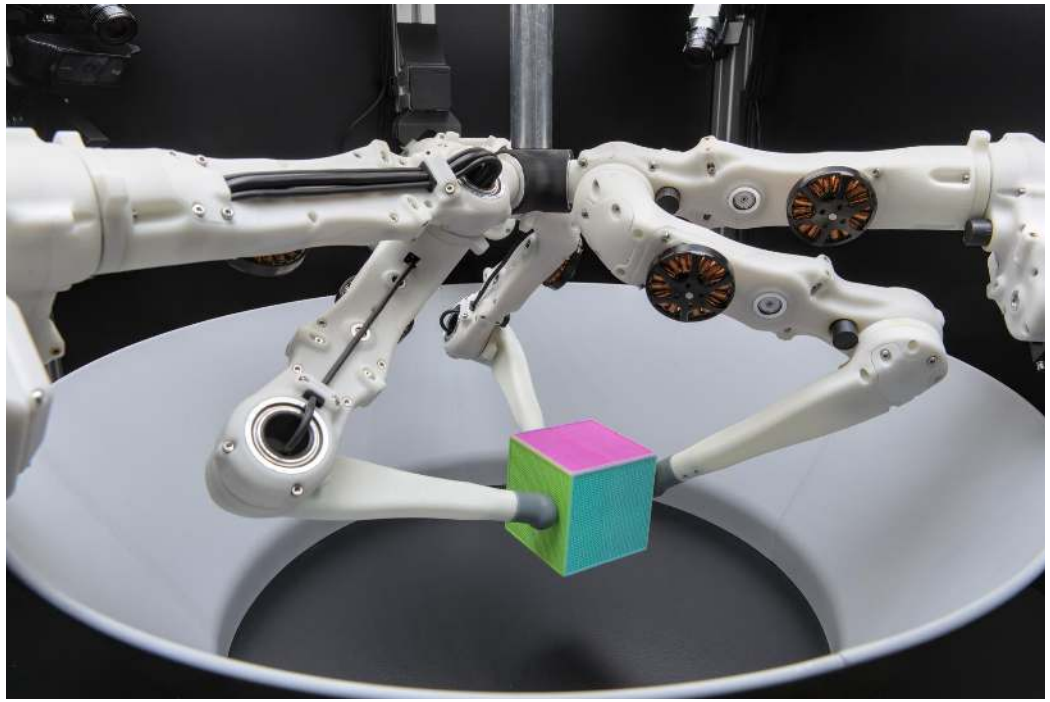
Prior work: Intrinsic Free Play with CEE-US

CEE-US: **C**urious **E**xploration using **E**pistemic **U**ncertainty via **S**tructured World Models



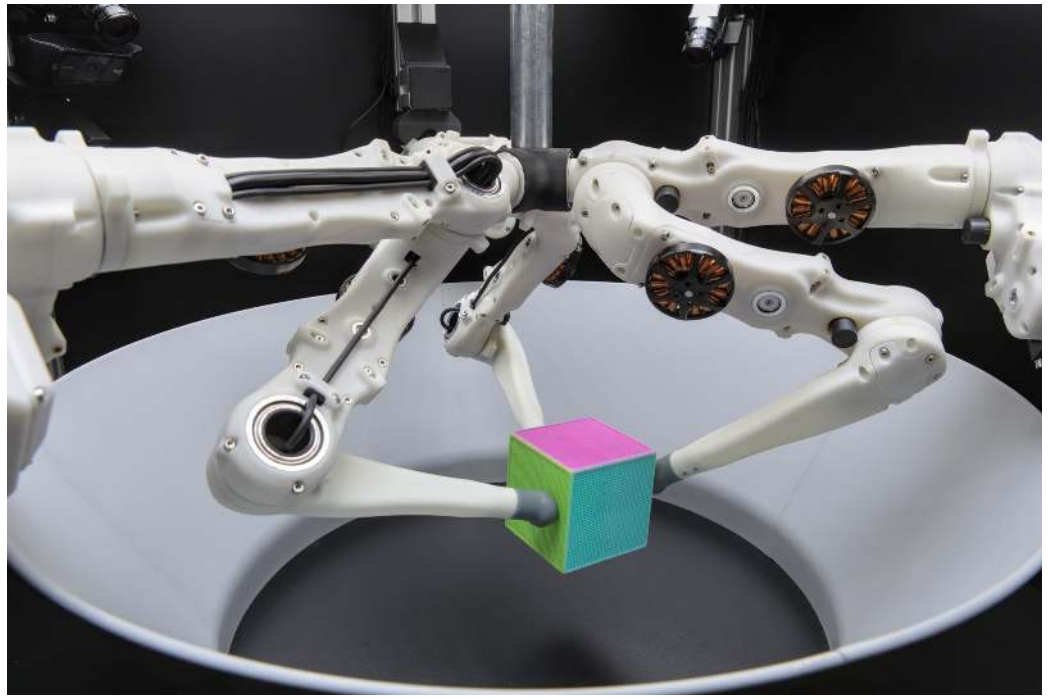
Integrating CEE-US with real robot systems

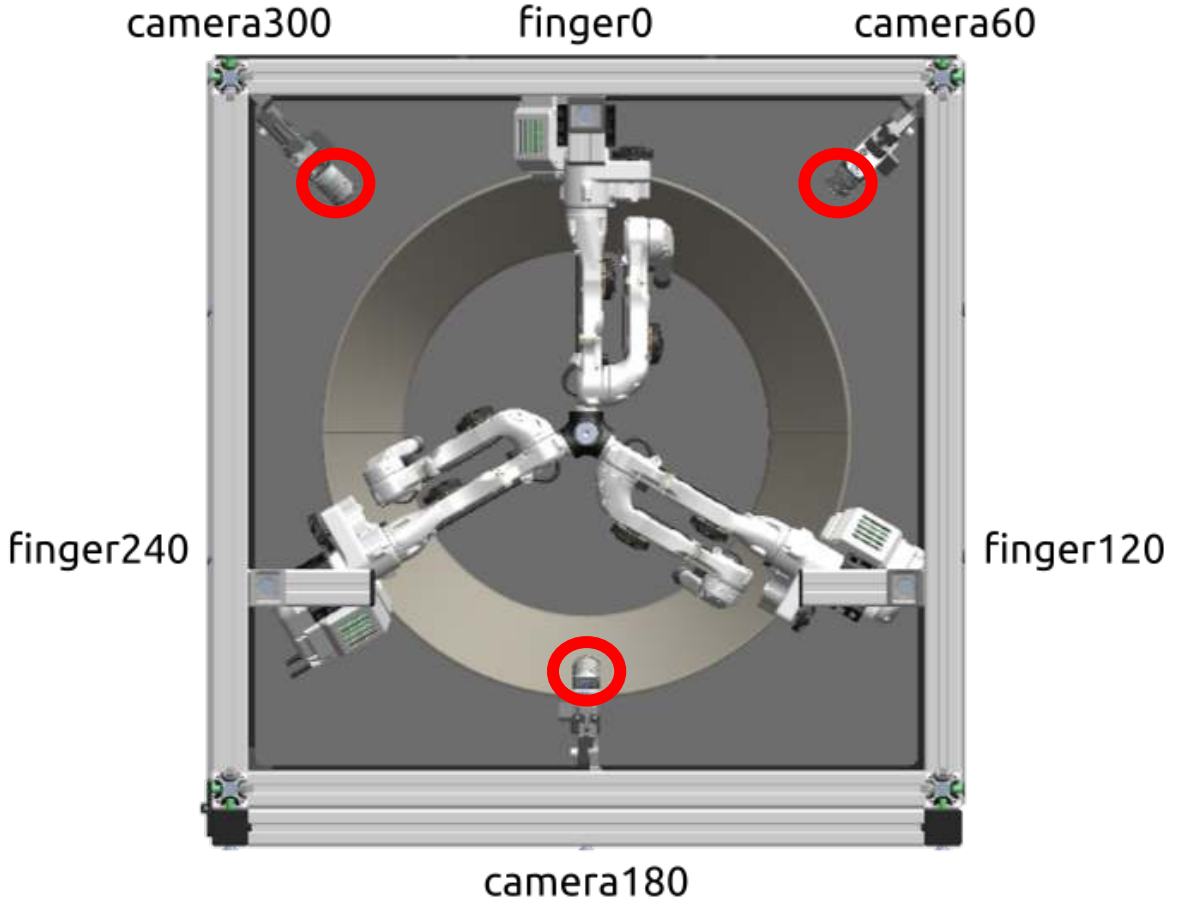




How can we integrate CEE-US in real world systems?

TriFinger Robot Platform

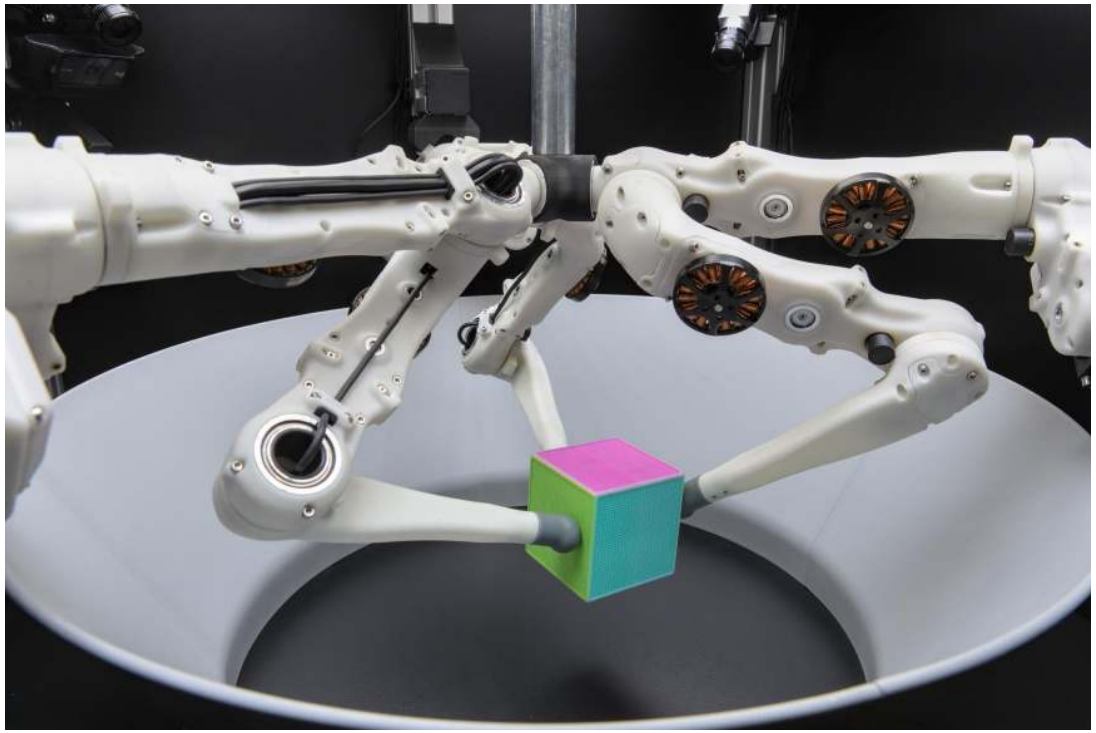




Learn how to effectively manipulate colored cubes using free-play

Wüthrich et al., 2021

Integrating CEE-US with TriFinger Robot



CEE-US currently needs:

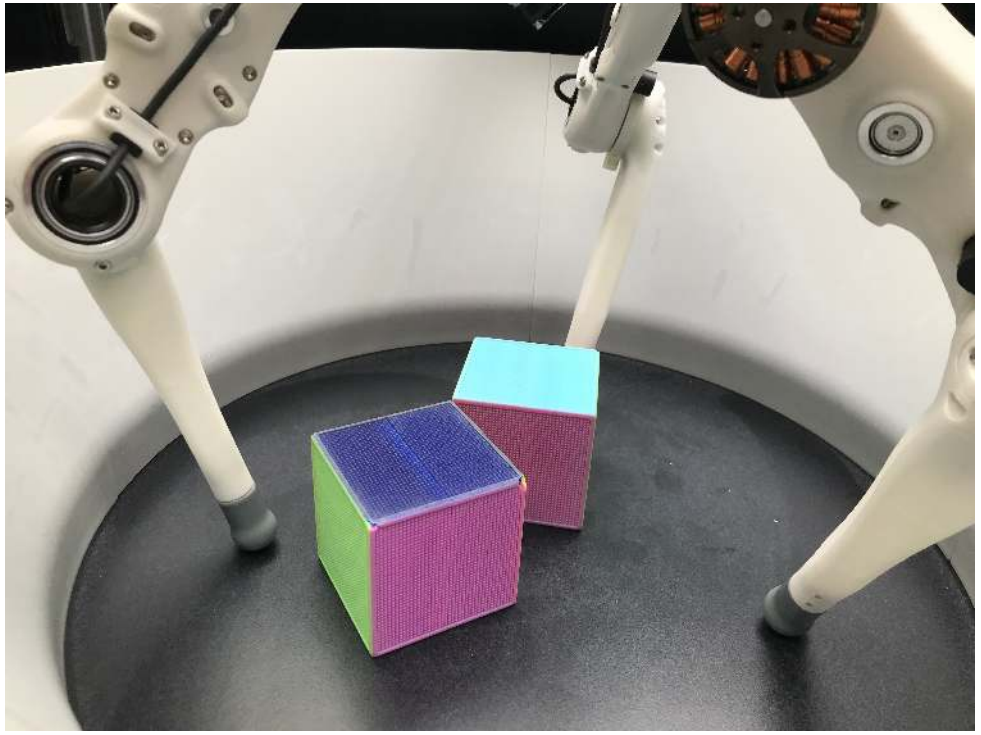
- object location
- object velocity
- object orientation

$$\text{Observation} = \begin{bmatrix} q_{robot} \\ \dot{q}_{robot} \\ q_{obj_1} \\ \dot{q}_{obj_1} \\ \vdots \\ q_{obj_N} \\ \dot{q}_{obj_N} \end{bmatrix}$$

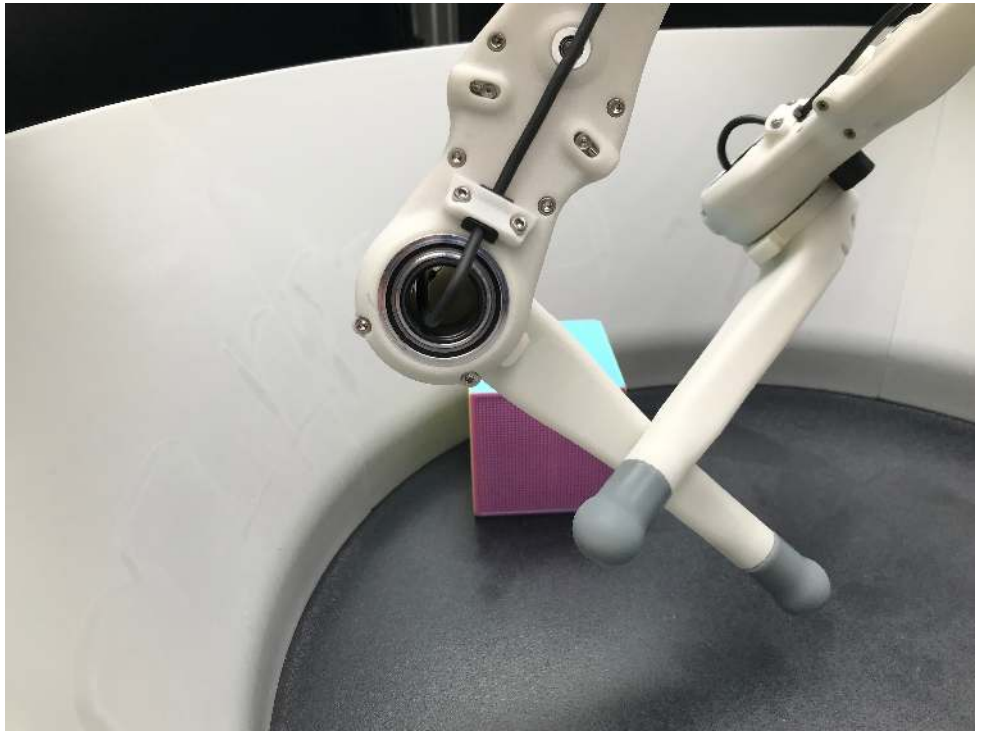
In simulation, these sensory information are easily available.

```
camera_observation = self.platform.get_camera_observation(t)
object_observation = camera_observation.object_pose
```

Challenges in real world perception



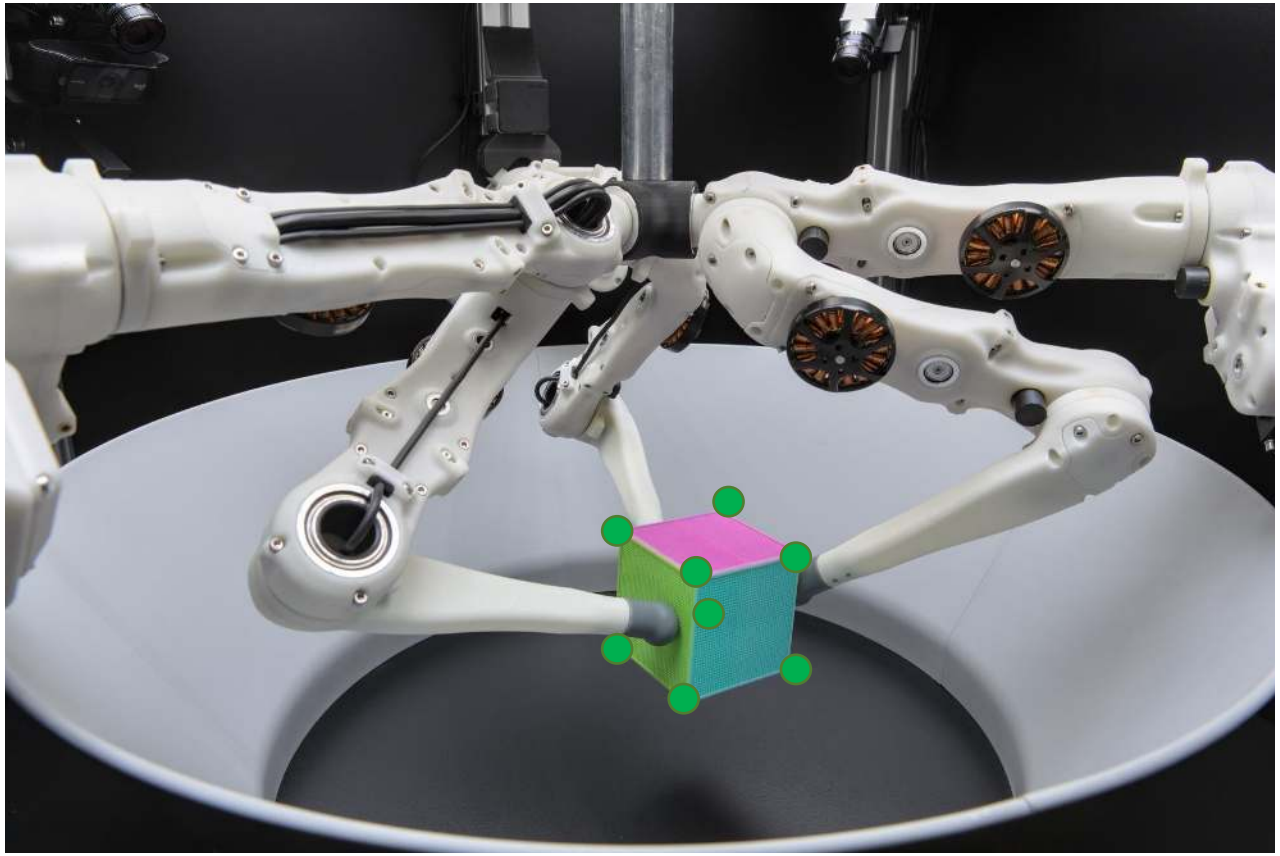
Occlusion between cubes



Occlusion between cube and arm

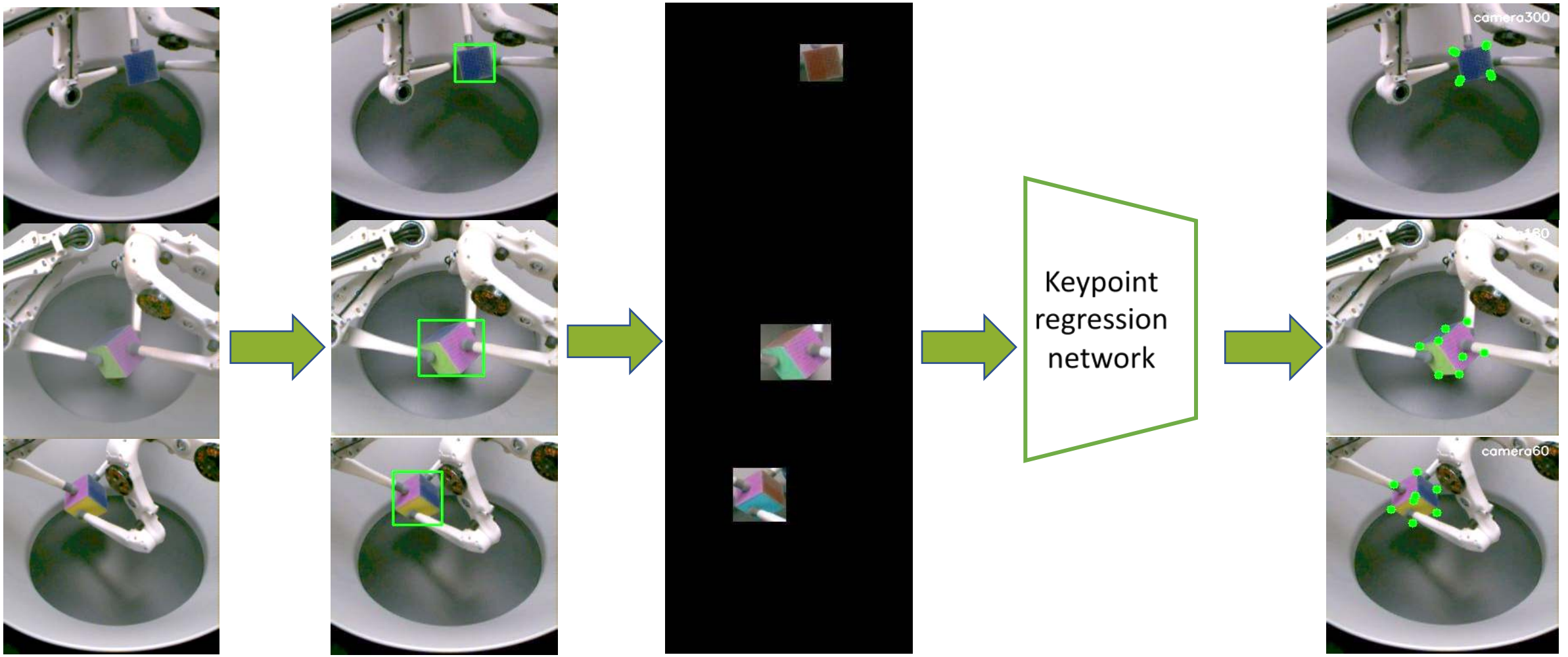
How about sensor noise?

Proposed solution: Cube Corner Predictor



Cube corners as 3D keypoints

Proposed solution: Cube Corner Predictor



3 RGB images

Finetuned object detector (YOLOv5)

Pre-processing

Keypoint results

Cube Corner Predictor

Using known cube geometry, we can extract:

- object location
- object orientation
- object velocity

Summary of perception system

Existing Perception system

only works with one object

processing speed is 10 FPS

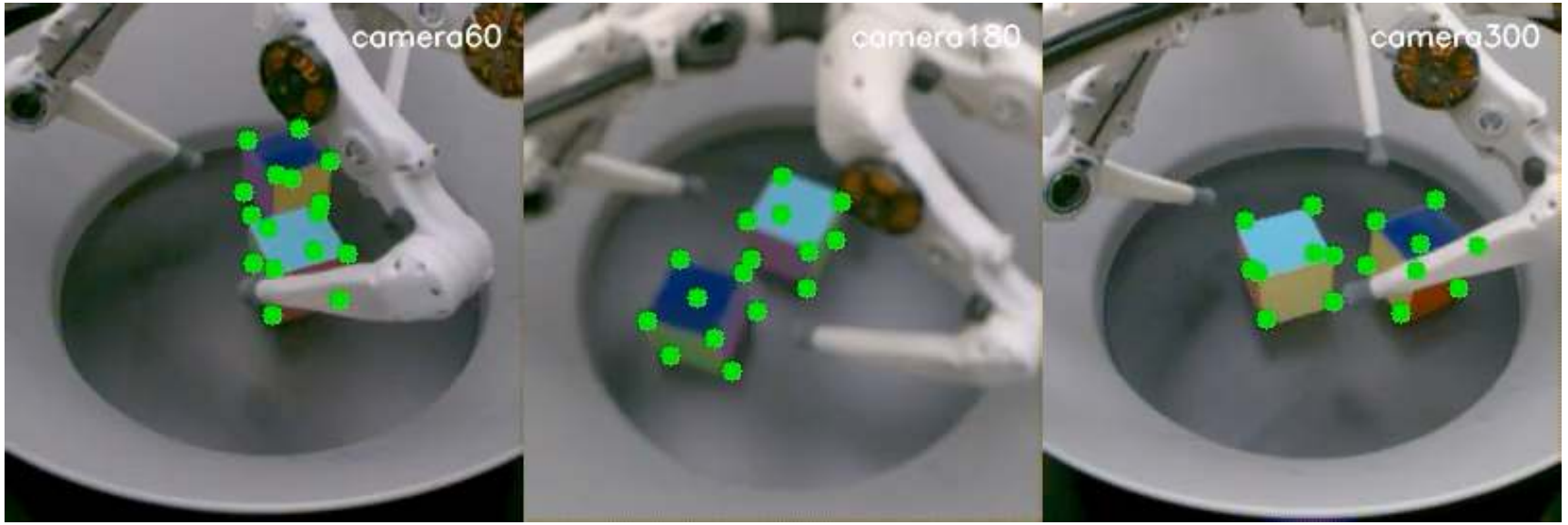
jittery

New Perception System

can handle multiple objects

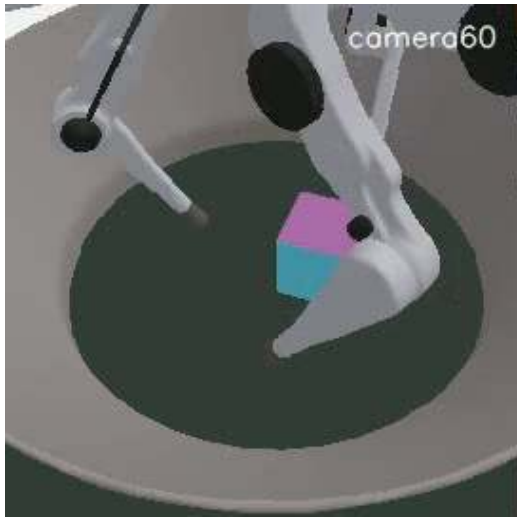
1.5 x faster (15 FPS) with multi-object tracking

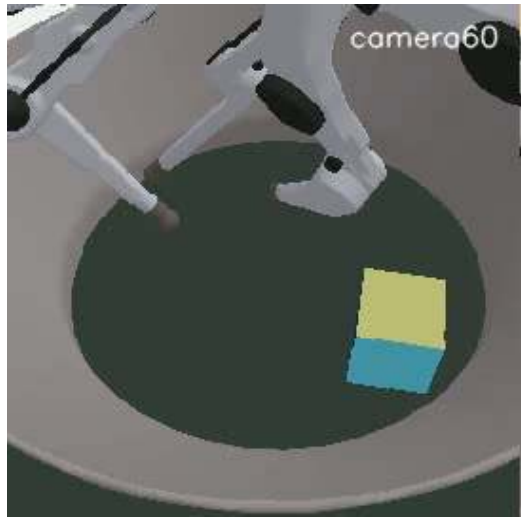
jittery (needs robust filtering)



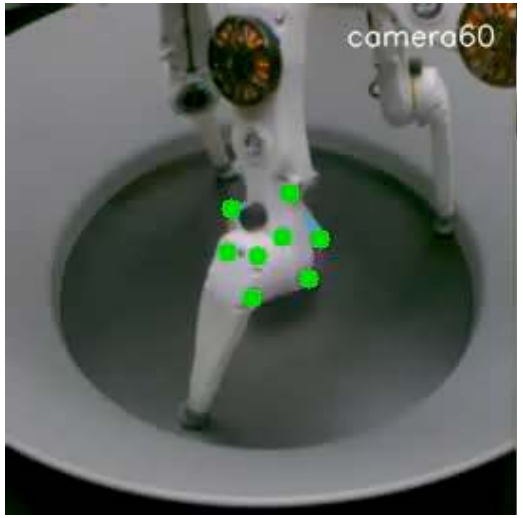
Free-play behavior in simulation and real world

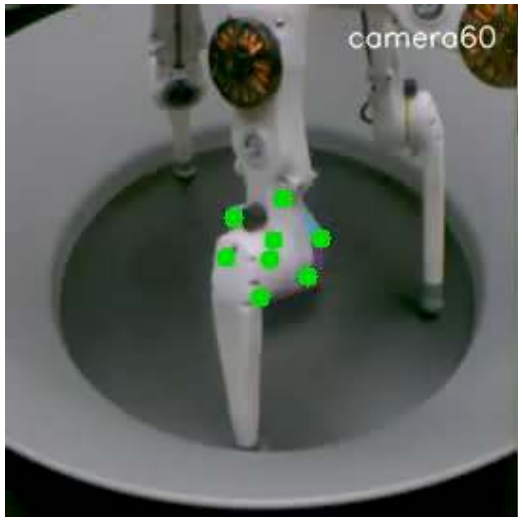
Simulation

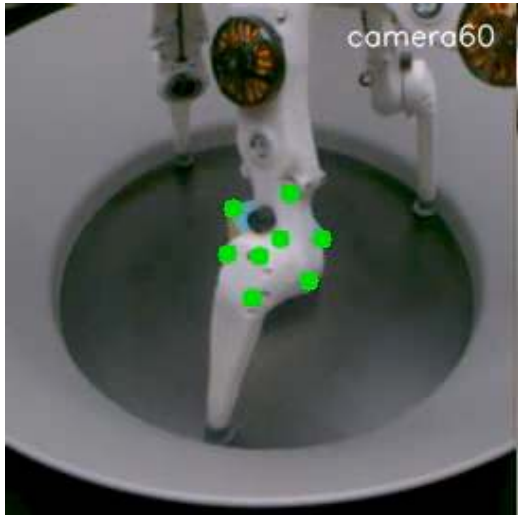




Real world







Thank you!

Shukrullo Nazirjonov
shukrullo.nazirjonov@fau.de