ROBOT INTELLIGENCE LAB

Visual Navigation

Sungjoon Choi, Korea University

kakaobrain

Contents

ROBOT INTELLIGENCE LAB

- Active Exploration
- Visual Navigation
- Memory Architecture
- Vision-Language Navigation
- Path Following

ROBOT INTELLIGENCE LAB

Active Exploration

"Modeling Observation Correlations for Active Exploration and Robust Object Detection," 2012

"Nonmyopic View Planning for Active Object Detection," 2013

"Learning Exploration Policies For Navigation," 2019

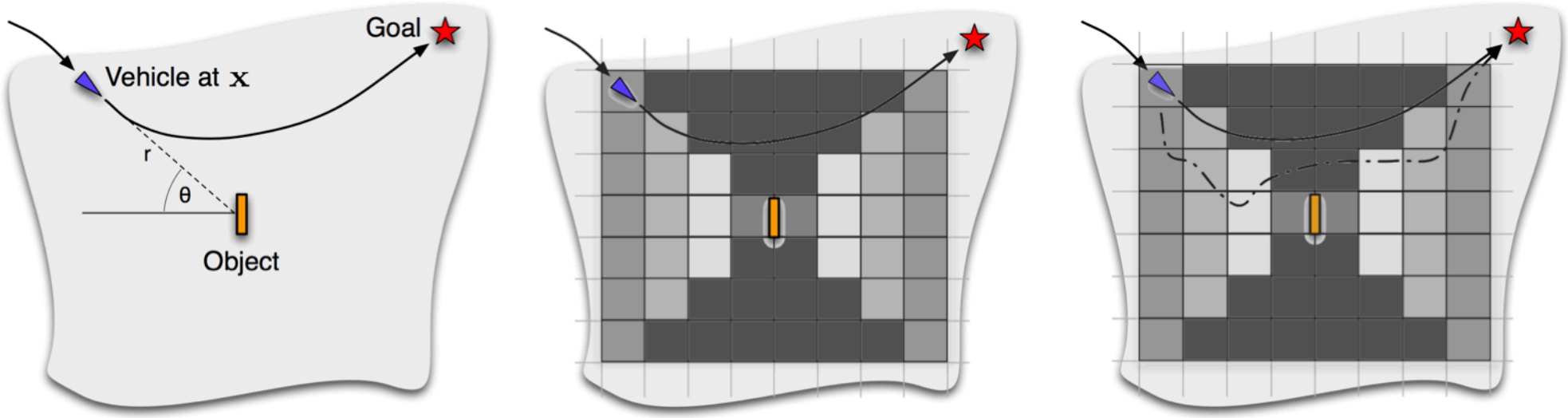
Active Exploration

ROBOT INTELLIGENCE LAB

- "Modeling Observation Correlations for Active Exploration and Robust Object Detection," 2012
  - This paper presents an **online planning** algorithm which learns an explicit model of the spatial dependence of object detection.
  - It generates the plan which maximizes the expected performance of the detection.
  - It learns the **spatial correlations** between measurements to estimate the mutual information between measurements taken at different locations.

Active Exploration

ROBOT INTELLIGENCE LAB

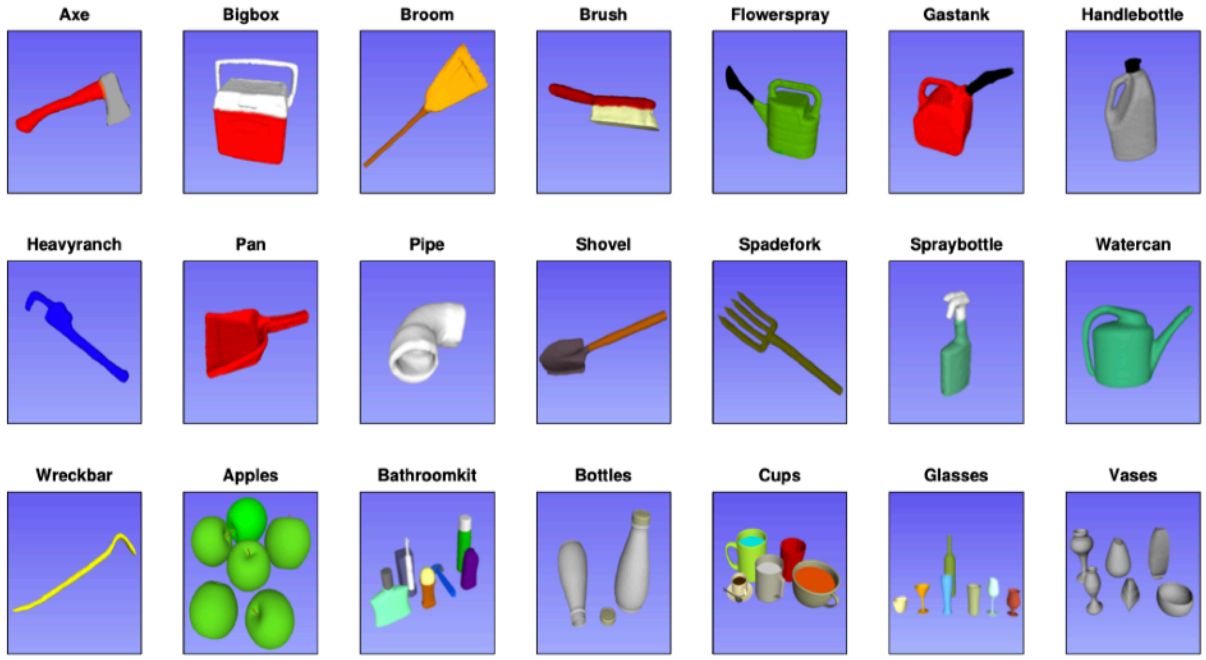


- The goal of this work is to find more informative route (dash-dotted line) compared to the original route (bold line).
- It placed an emphasis on spatial relations to model the correlations (mutual information) between sensor readings and combined it with path-planning.

Active Object Detection

ROBOT INTELLIGENCE LAB

- "Nonmyopic View Planning for Active Object Detection," 2013
  - This paper proposes an active approach to object detection by controlling the point of view of a mobile depth camera.
  - The mobile sensor plans its trajectories by balancing the amount of **energy used to move** with the chance of **identifying the correct hypothesis**.

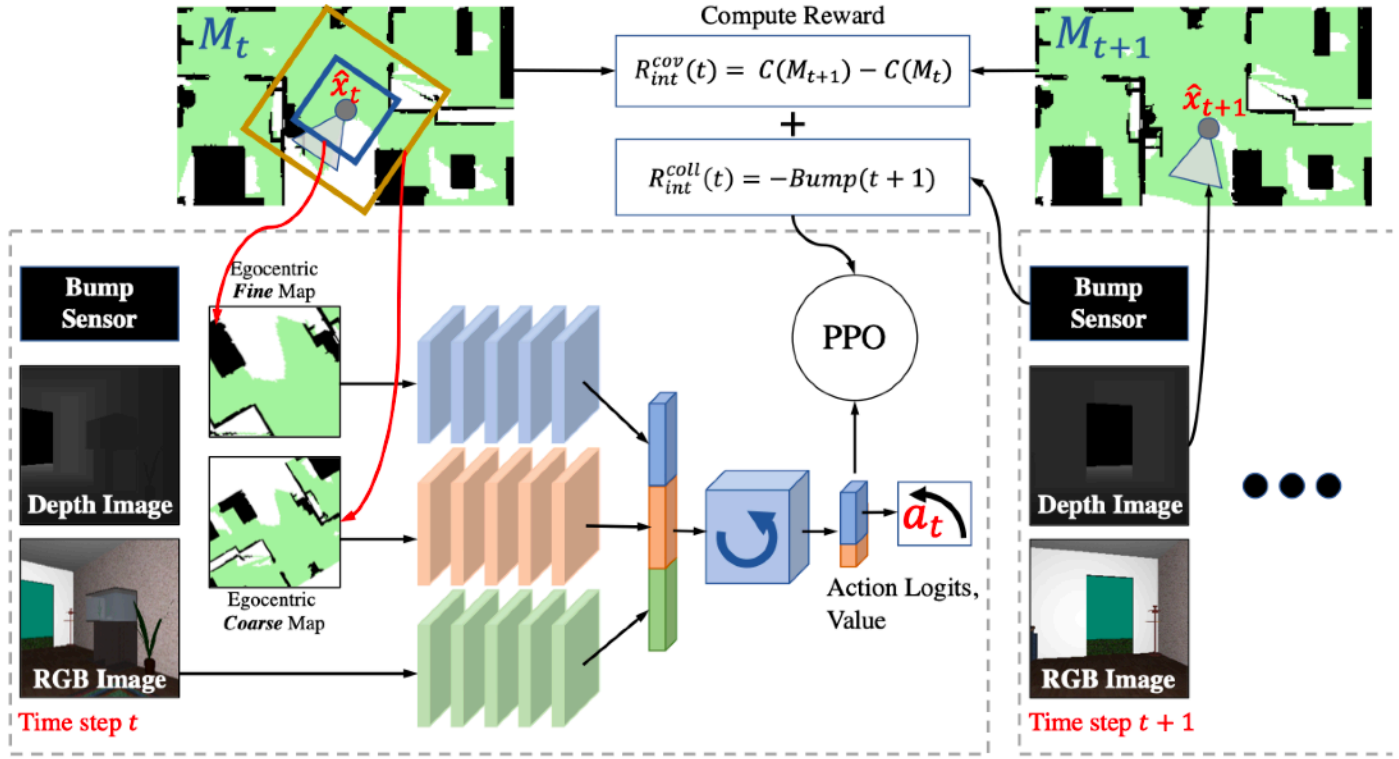


Exploration Policy

ROBOT INTELLIGENCE LAB

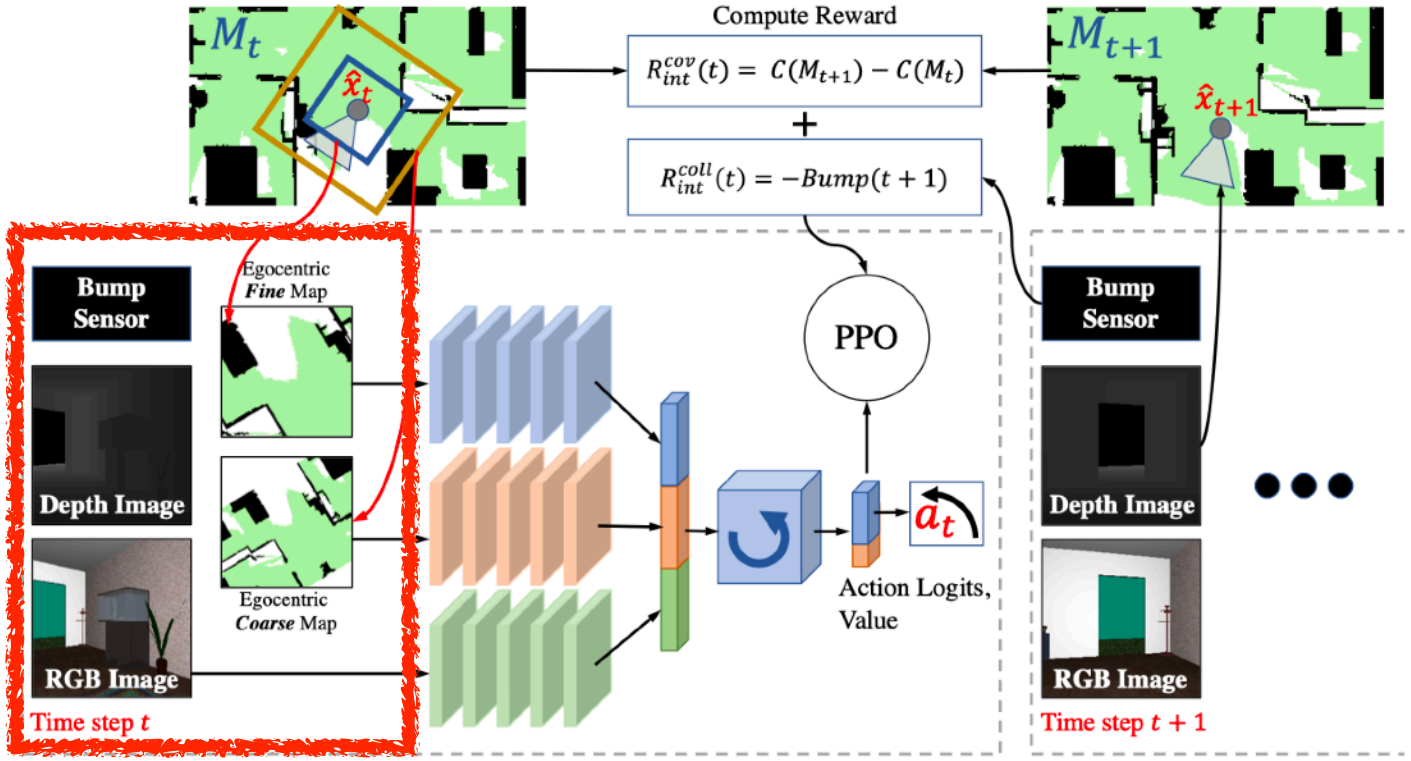
- "Learning Exploration Policies For Navigation," 2019 (CMU, FAIR)
  - It studies how agents can autonomously explore realistic and complex 3D environments without the context of **task-rewards**.
  - The usage of policies with **spatial memory** that are bootstrapped with imitation learning + fine-tuning with **coverage rewards** can be effective at exploring novel environments.

Exploration Policy



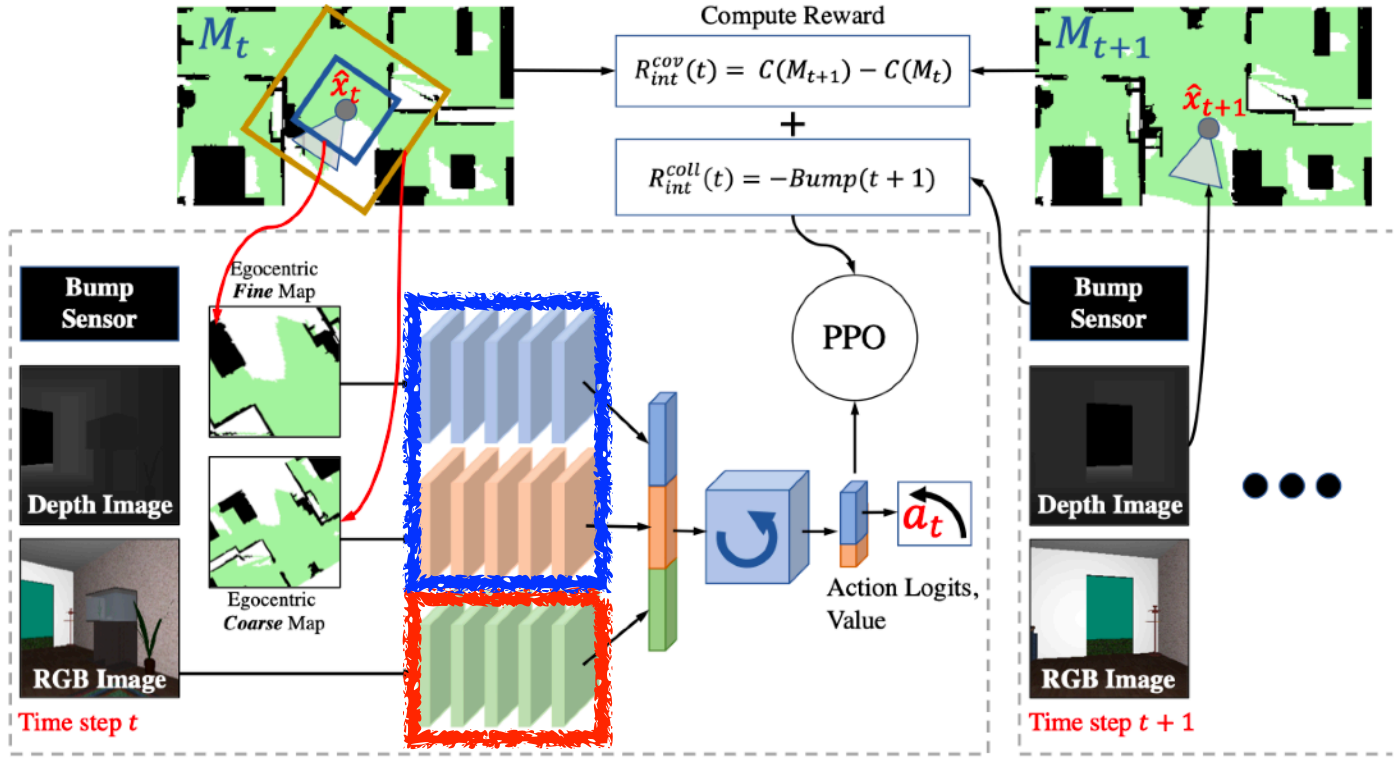
- It is assumed that a mobile robot is equipped with an **RGB-D camera** and a **bump sensor** and has been dropped into a **novel environment** that it has never been in before.

Exploration Policy



- It uses the depth camera to build and update **a map** by maintaining an estimate of its current location and projecting 3D points observed in the depth image. A **2D map** is also generated.

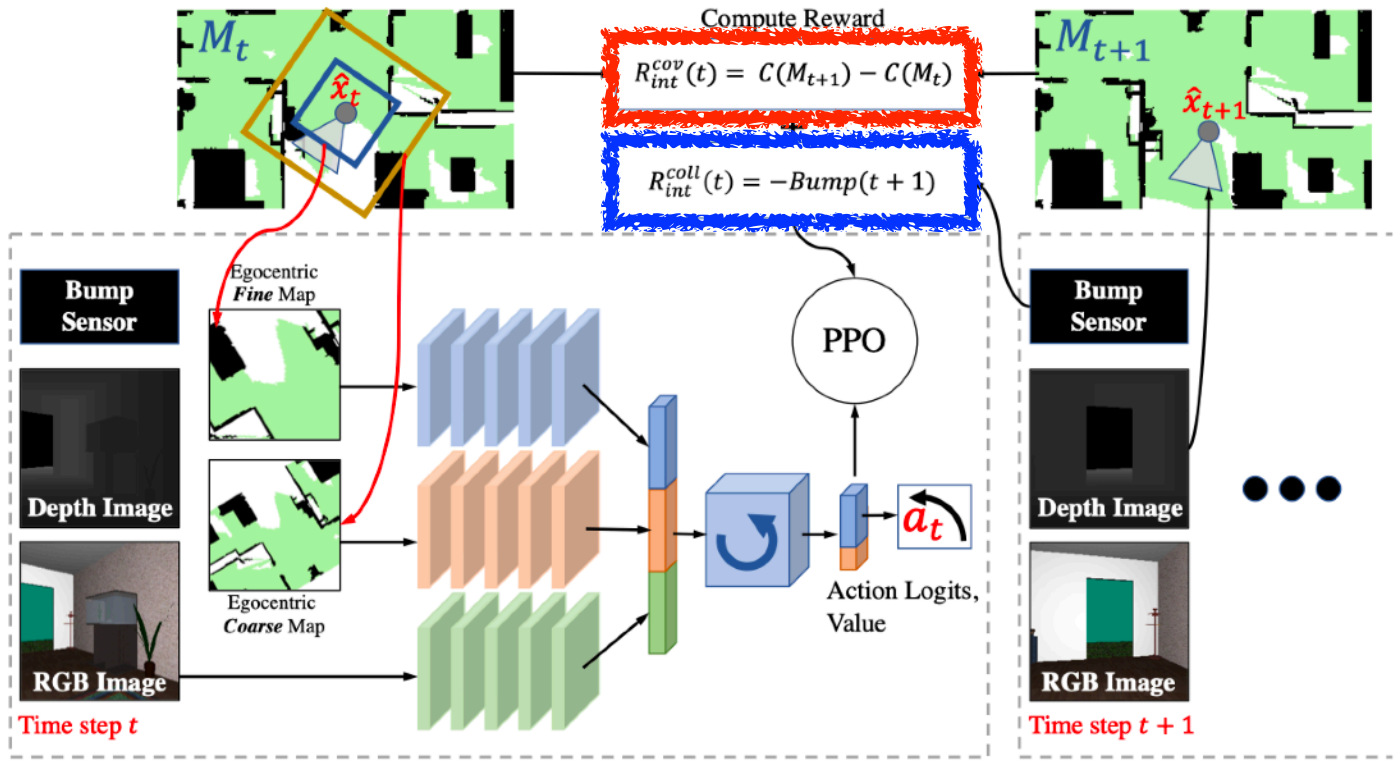
Exploration Policy



- The policy receives **RGB image** and **Occupancy Maps**.

Exploration Policy

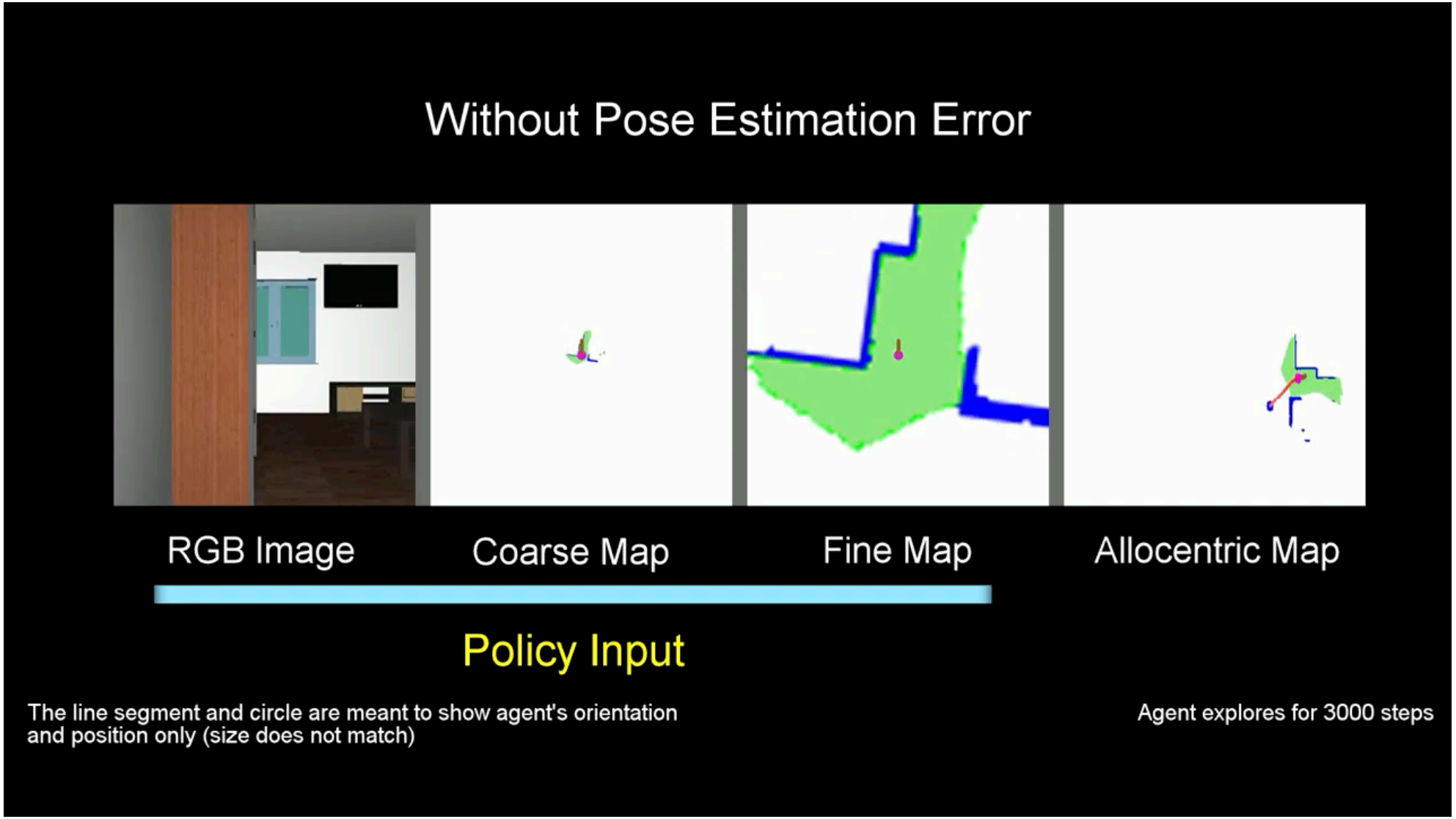
ROBOT INTELLIGENCE LAB



- The **coverage reward** used as an intrinsic reward function to train the exploration policy and obtained via gain in coverage in the map.
- It also penalize the **bump**.

Exploration Policy

ROBOT INTELLIGENCE LAB



ROBOT INTELLIGENCE LAB

Visual Navigation

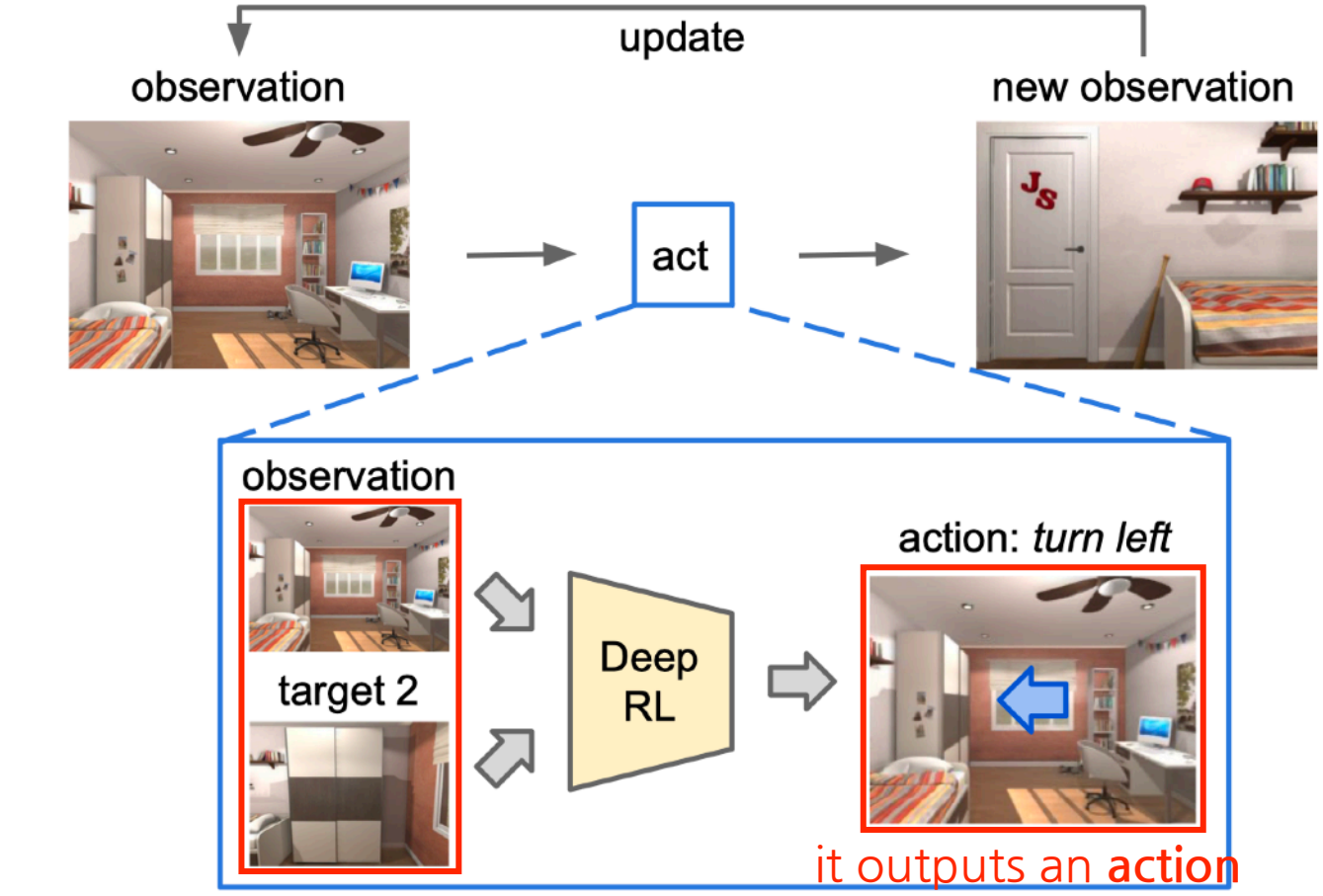
"Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning," 2016

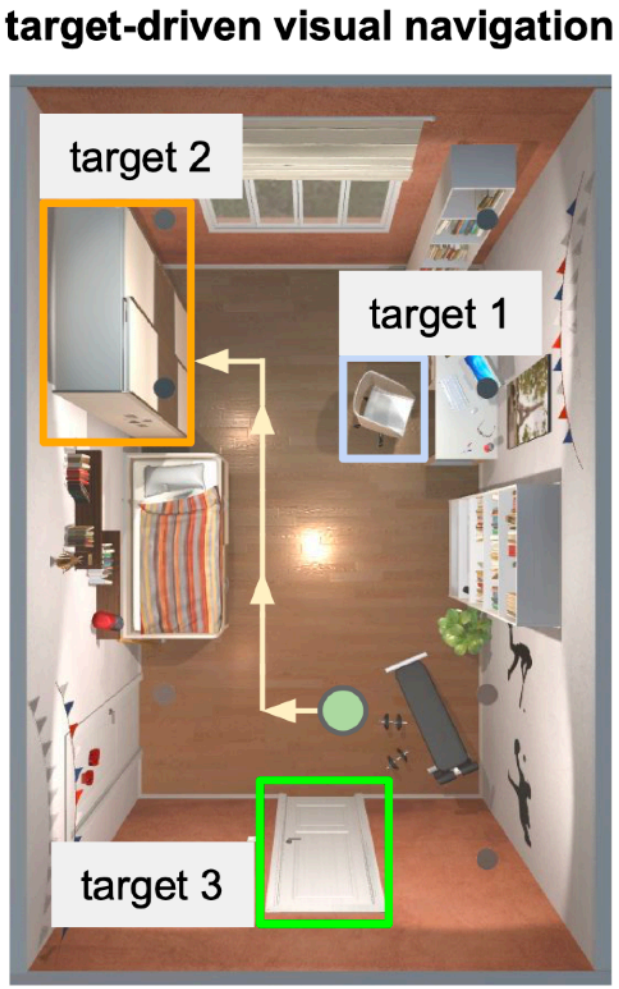
"Learning to Navigation in Complex Environments," 2017

Target-driven Visual Navigation

- "Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning," 2016
  - The goal of deep reinforcement learning is to navigate towards a visual target with a minimum number of steps.
  - The model takes the **current observation** and the **image of the target** as inputs and generates an action in the 3D environment as the output.
  - It learns to navigate to different targets in a scene without retraining.
  - This is a **seminal** paper in visual navigation!
    - from Stanford, Allen Institute for AI, CMU, and UW

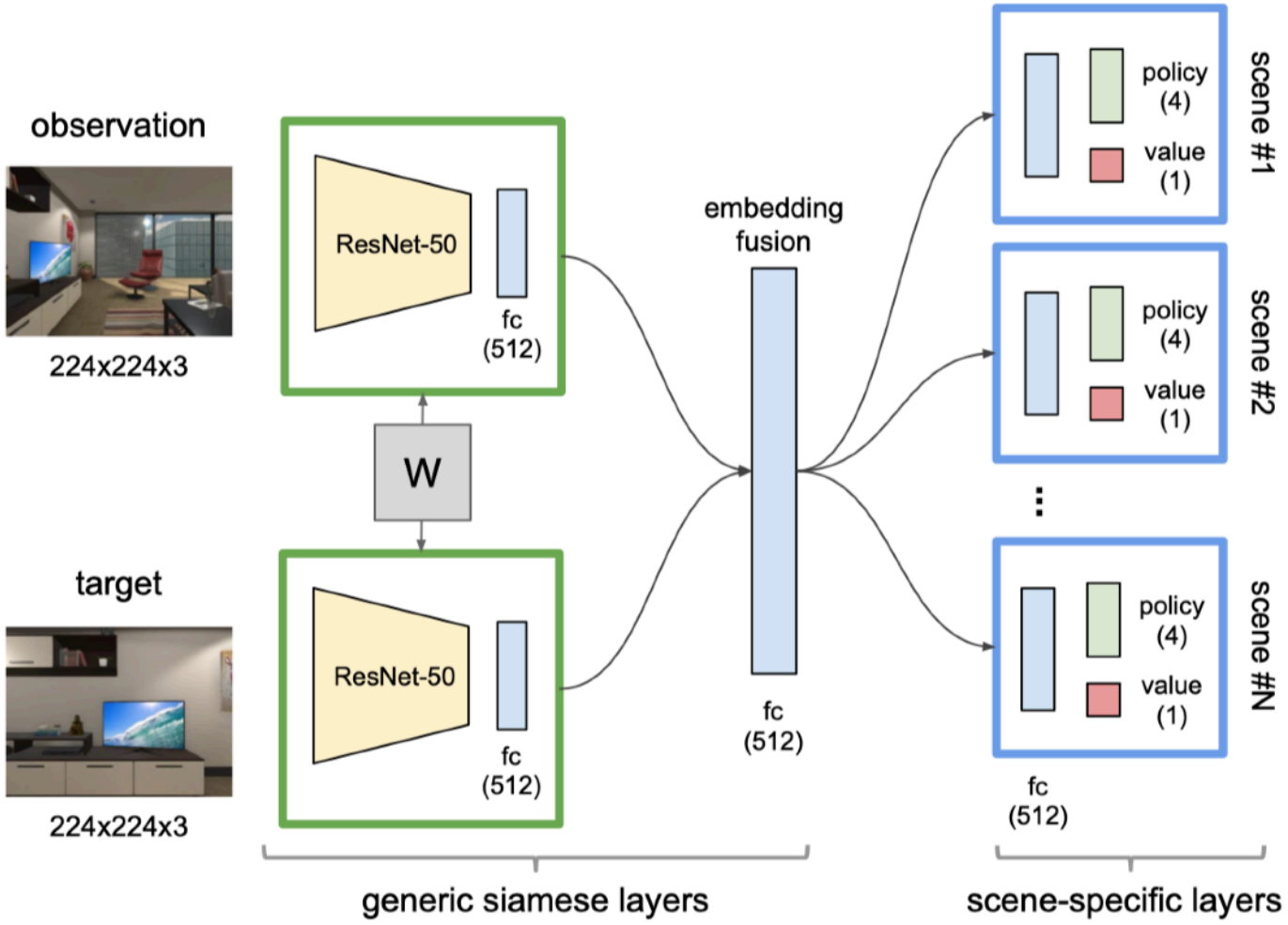
Target-driven Visual Navigation





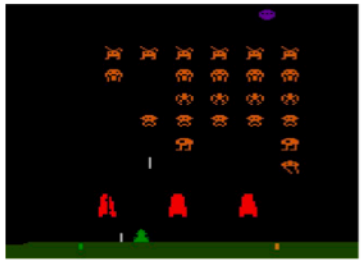
Target-driven Visual Navigation

ROBOT INTELLIGENCE LAB



Target-driven Visual Navigation

ROBOT INTELLIGENCE LAB

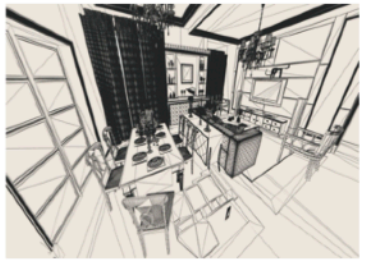


ALE

ViZDoom

UETorch

Project Malmo



SceneNet

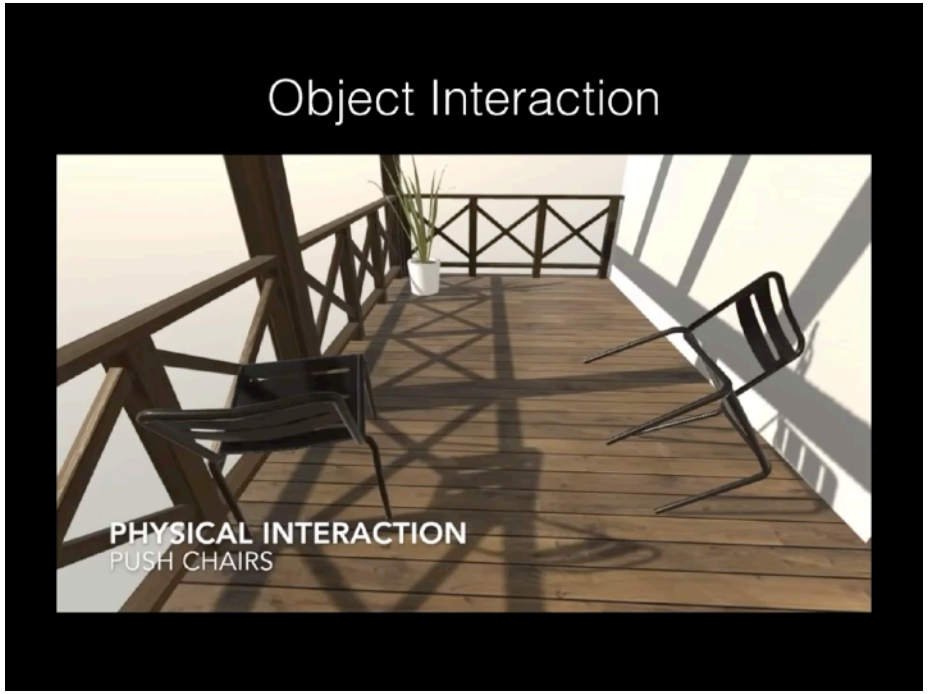
TORCS

SYTHNIA

Virtual KITTI

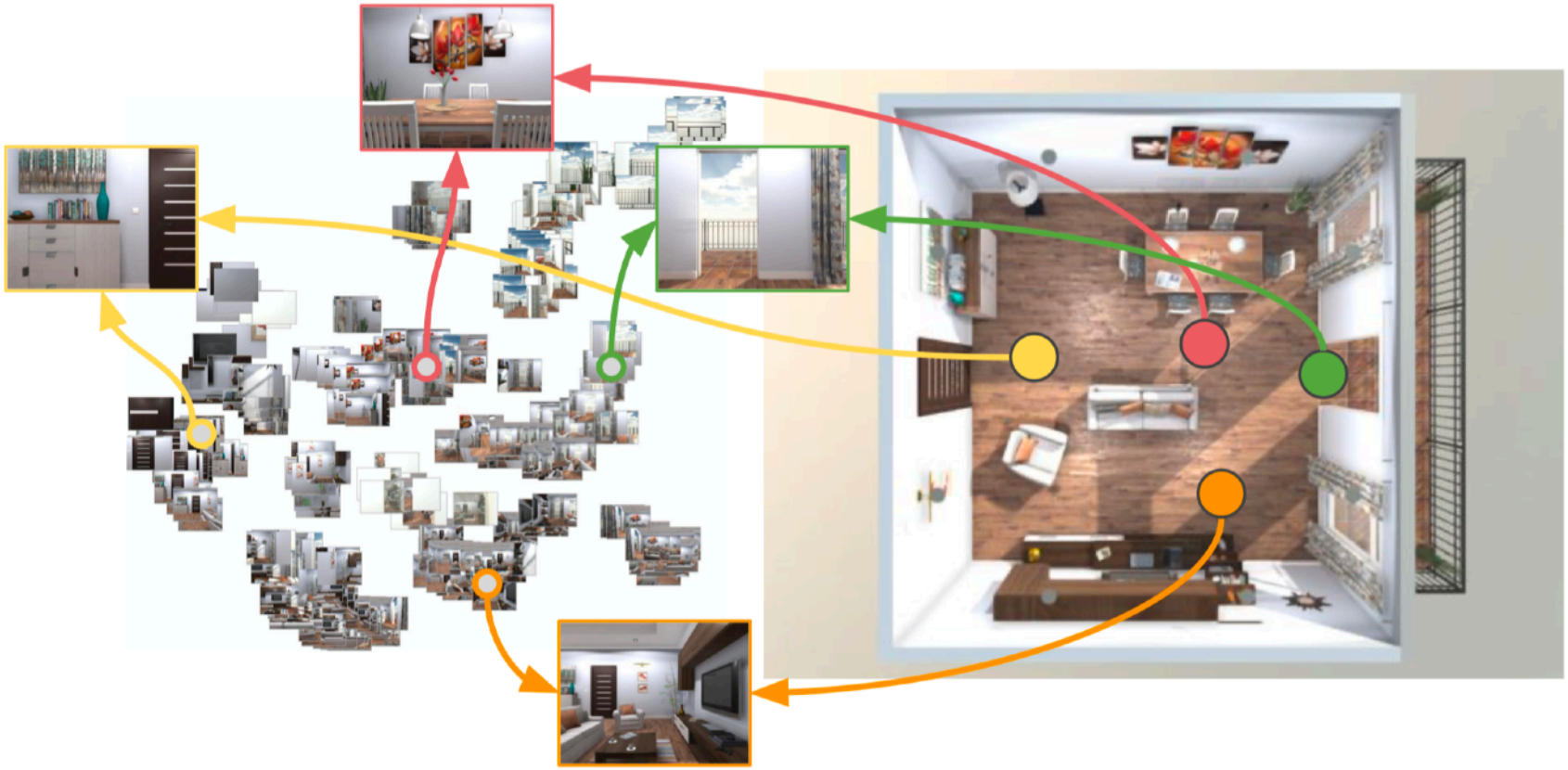
Ours

- The developed simulation framework (**AI2-THOR**), can freely navigate and interact with the environment utilizing **Unity 3D**.



Target-driven Visual Navigation

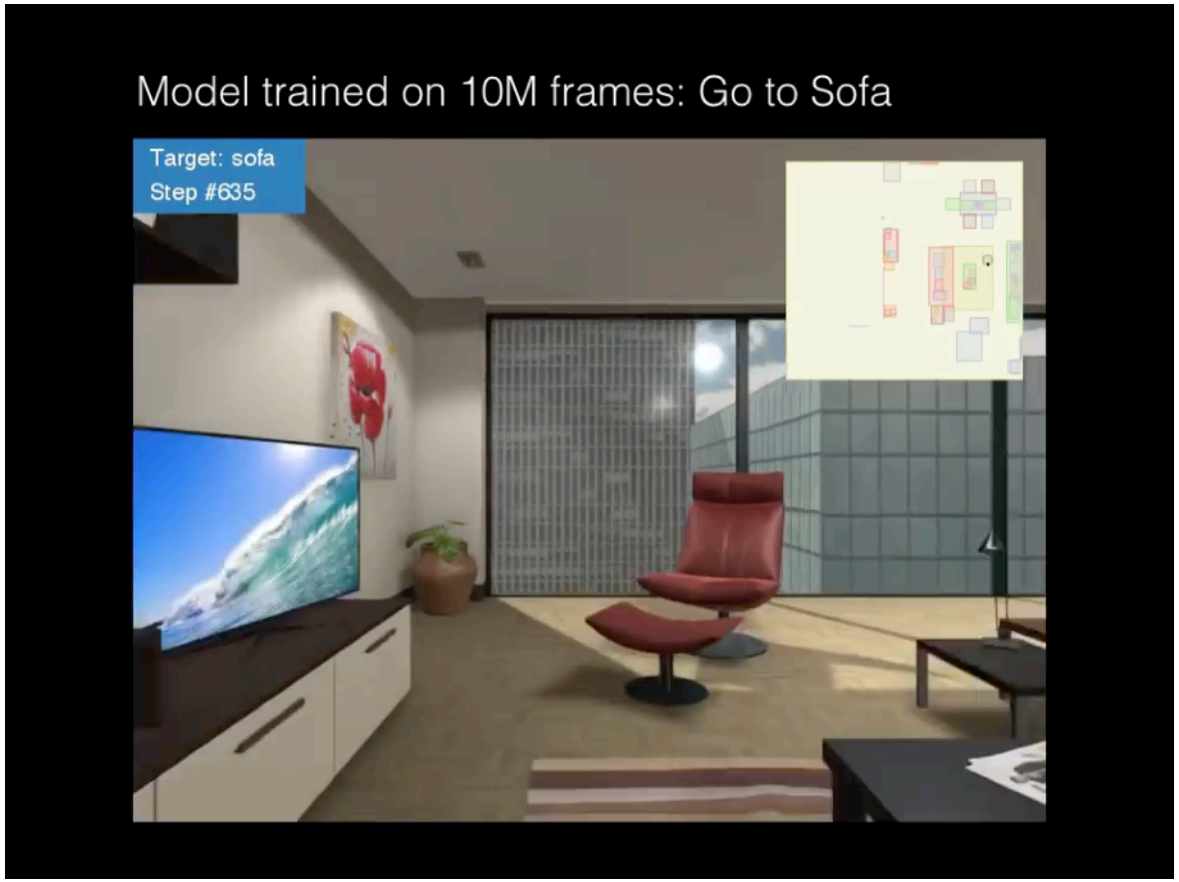
ROBOT INTELLIGENCE LAB



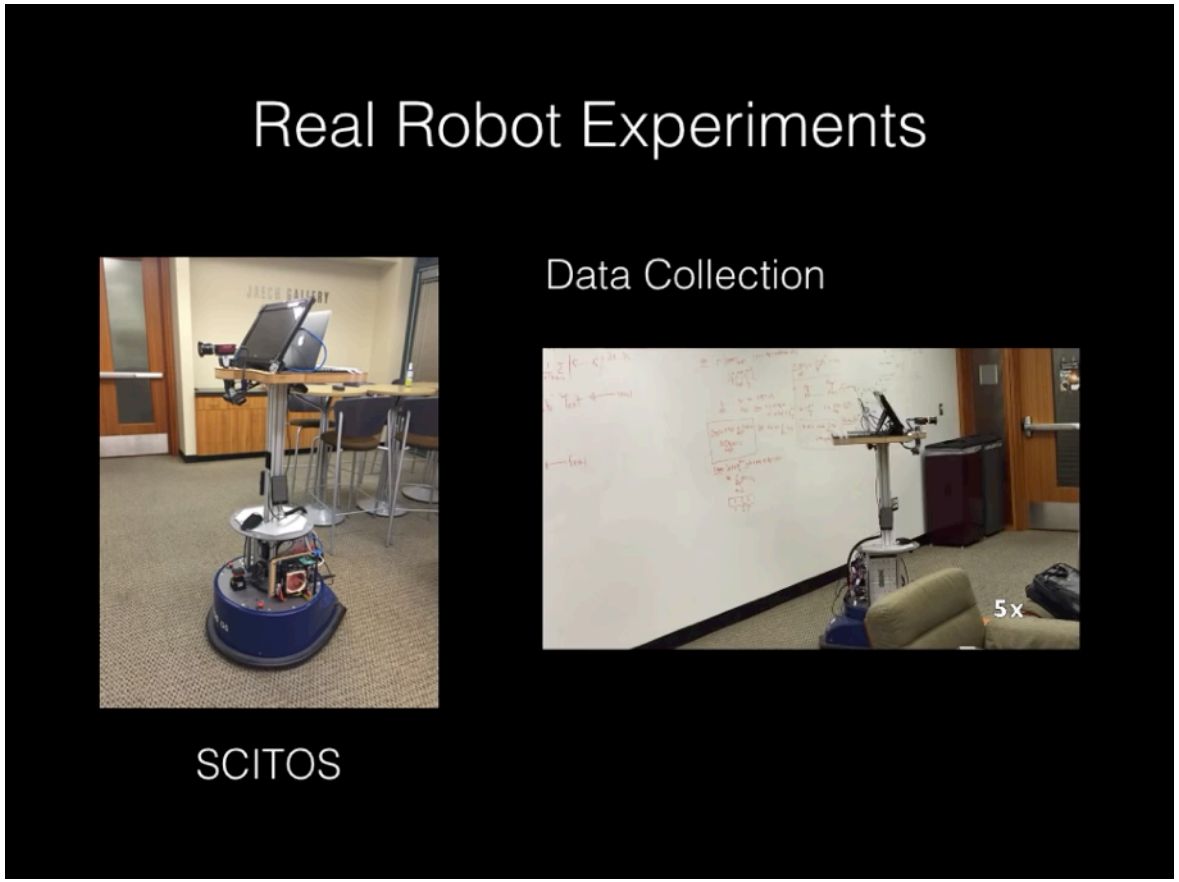
- The model has learned observation embeddings while preserving their spatial layout.

Target-driven Visual Navigation

ROBOT INTELLIGENCE LAB



Simulation

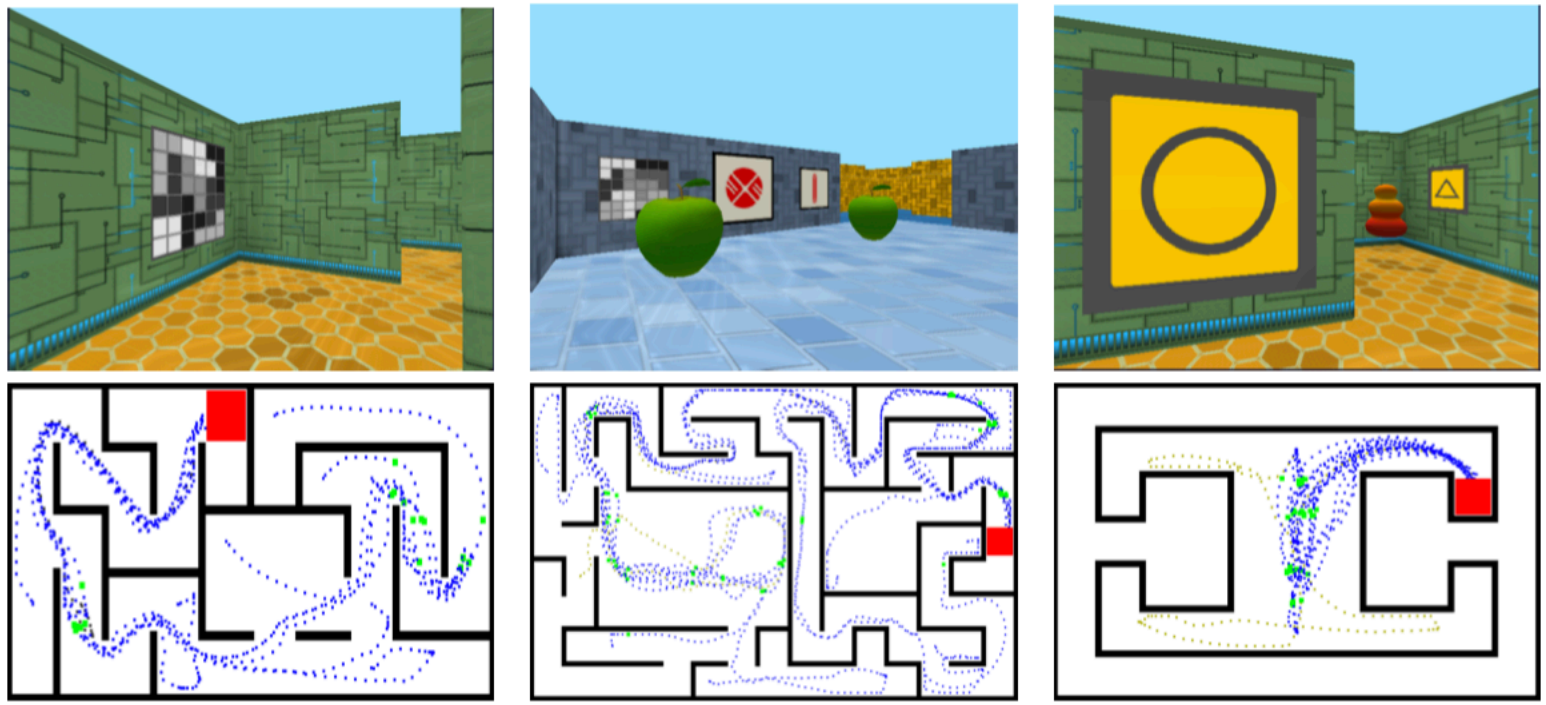


Real Robot Experiments

Learning to Navigate

ROBOT INTELLIGENCE LAB

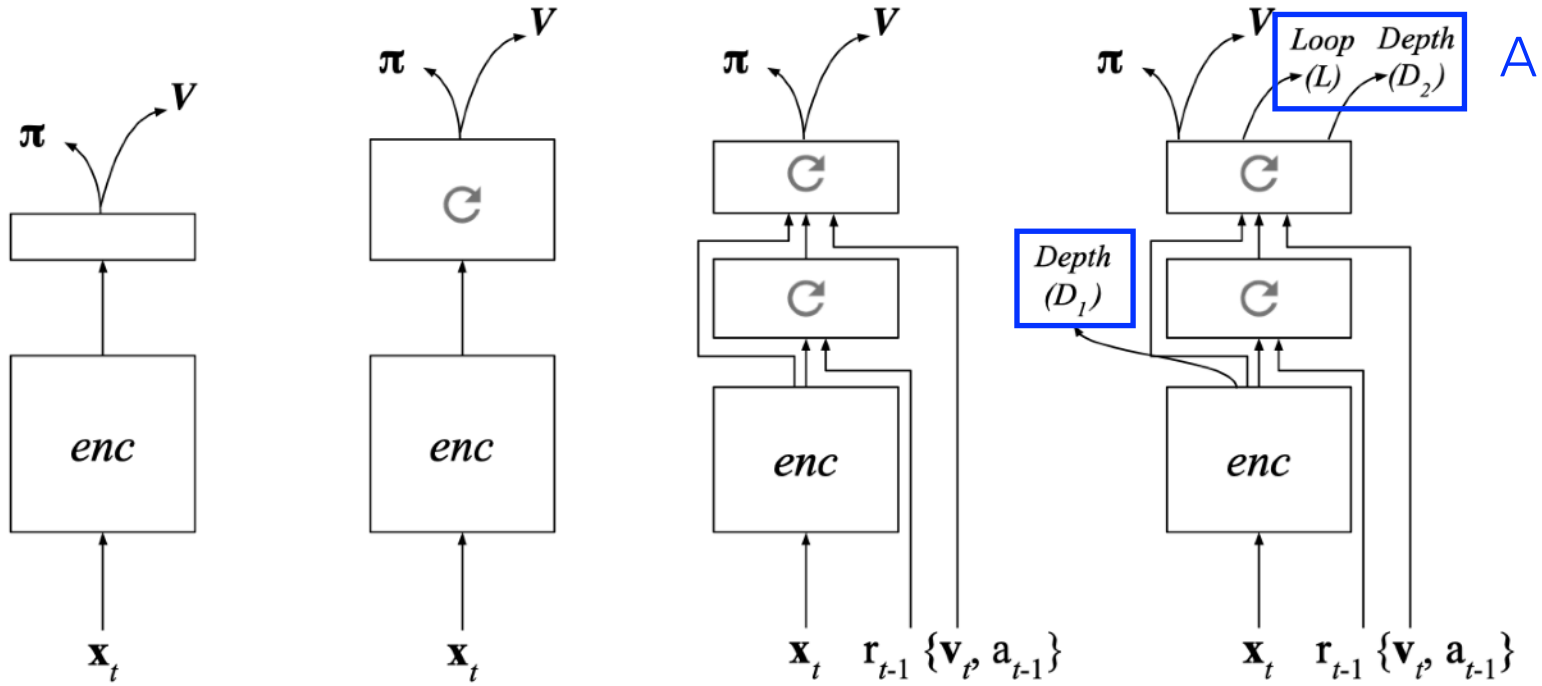
- "Learning to Navigation in Complex Environments," 2017 (DeepMind)
  - It utilizes goal-driven reinforcement learning problem for visual navigation with auxiliary depth prediction and loop closure classification tasks.
  - This is from DeepMind.



Learning to Navigate

ROBOT INTELLIGENCE LAB

- Auxiliary tasks of **depth prediction** and **loop closure classification** are utilized with a goal-driven reinforcement learning problem.

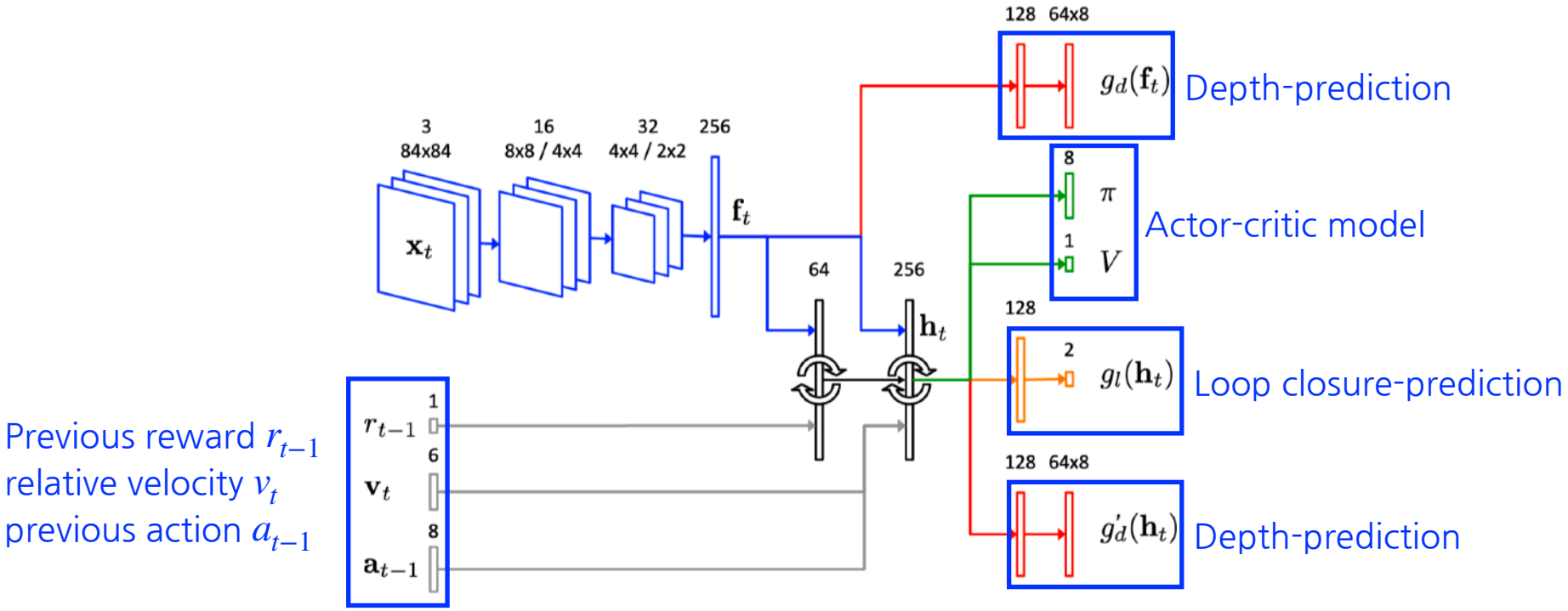


a. FF A3C b. LSTM A3C c. Nav A3C d. Nav A3C +D_1D_2L

Learning to Navigate

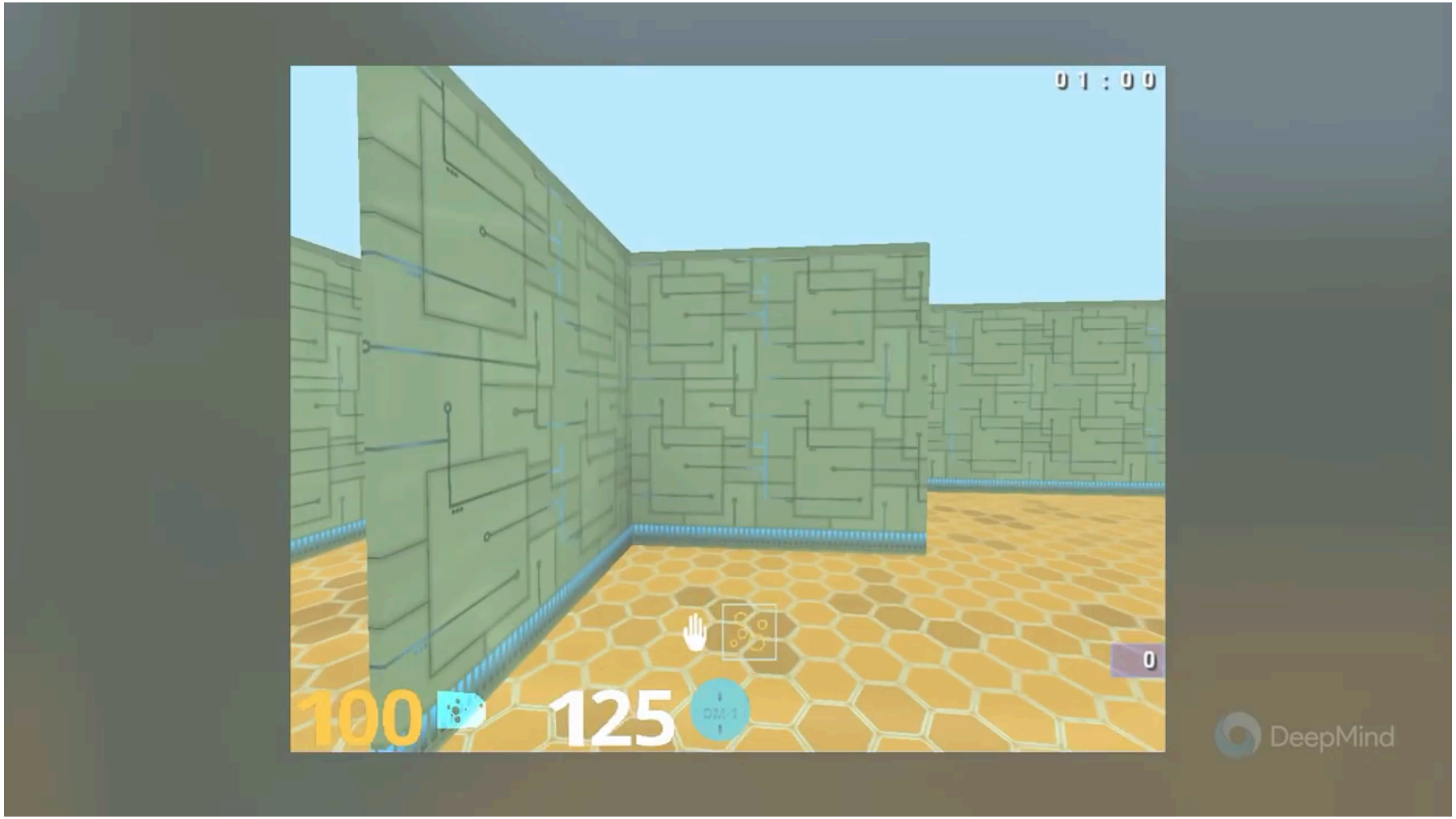
ROBOT INTELLIGENCE LAB

- It is not learning a goal-conditioned policy.
- Reward is achieved in the environments by reaching the goal from a random start location and orientation.



Learning to Navigate

ROBOT INTELLIGENCE LAB



https://youtu.be/M40rN7afngY

ROBOT INTELLIGENCE LAB

Memory Architecture

"Neural Map: Structured Memory For Deep Reinforcement Learning," 2017

"Semi-Parametric Topological Memory for Navigation," 2018

"Bayesian Relational Memory for Semantic Visual Navigation," 2019

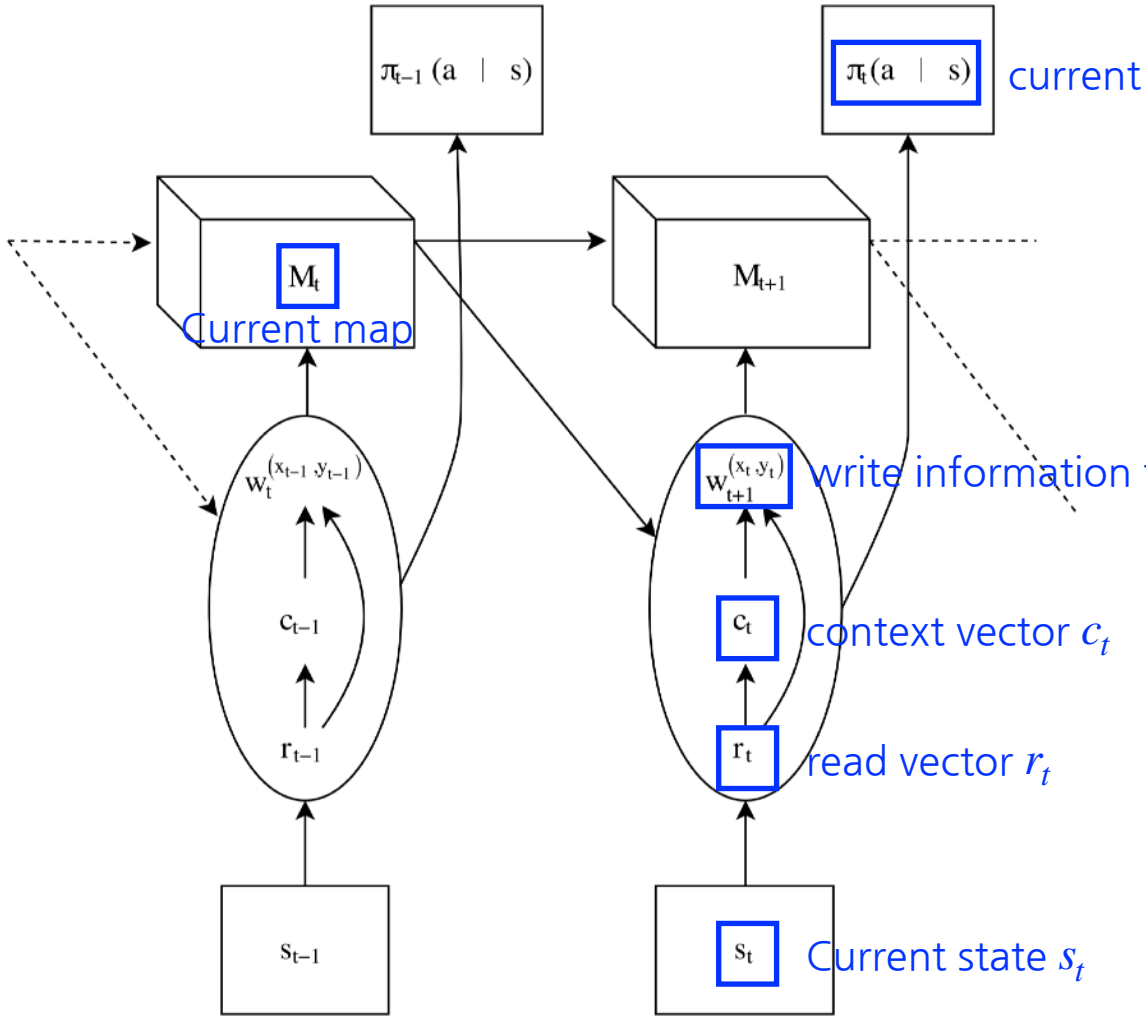
Neural Map

ROBOT INTELLIGENCE LAB

- "Neural Map: Structured Memory For Deep Reinforcement Learning," 2017
  - Deep reinforcement learning has been widely used for visual navigation tasks.
  - However, deep RL with an explicit **memory** has not been explored much.
  - It uses a spatially **structured 2D memory image** to learn to store arbitrary information about the environment over long time lags.

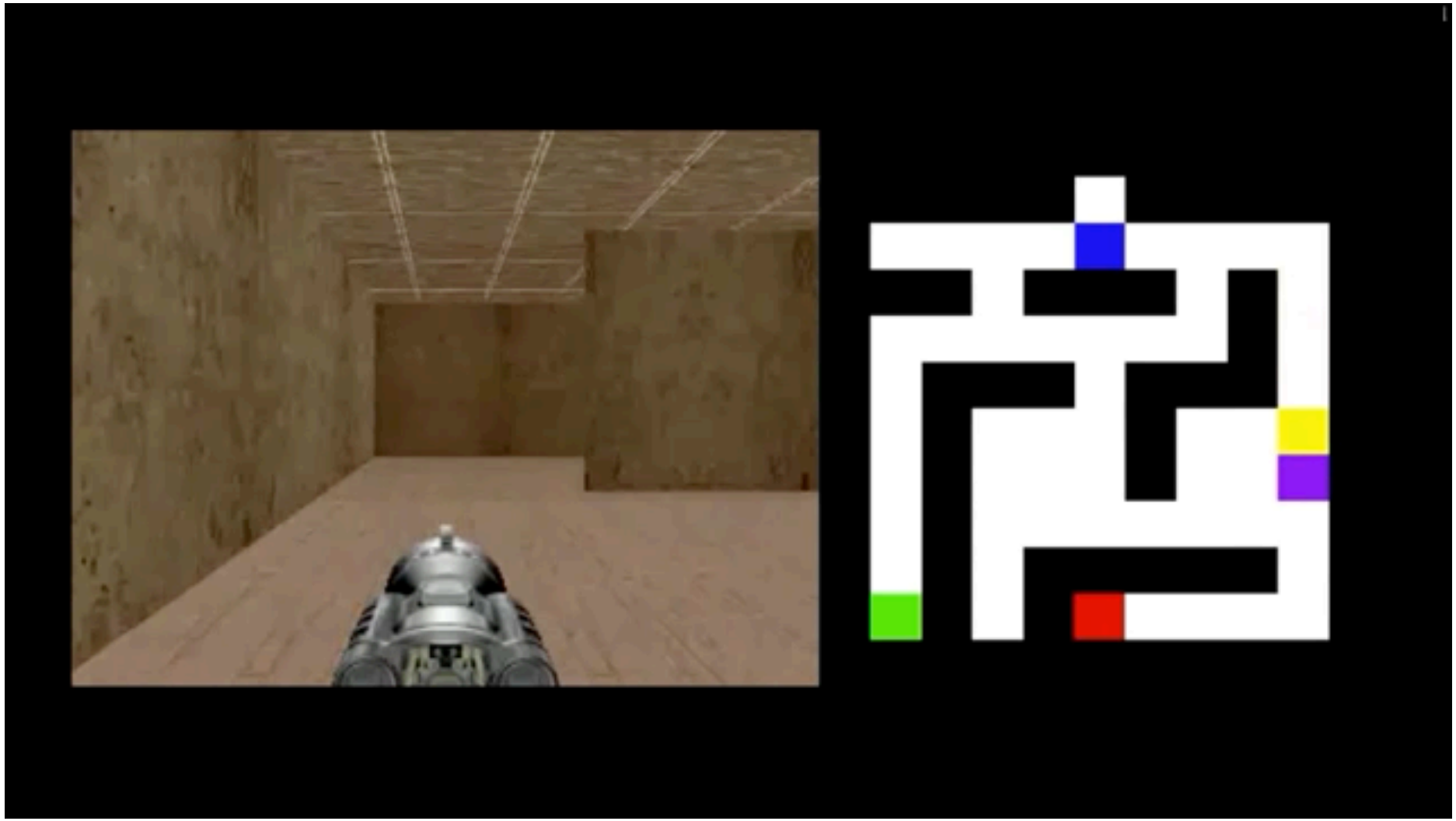
$$
\begin{aligned}
r_t &= read(M_t) && \text{Current neural map } M_t \\
c_t &= context(M_t, s_t, r_t) && \text{Current state } s_t \\
w_{t+1}^{(x_t,y_t)} &= write(s_t, r_t, c_t, M_t^{(x_t,y_t)}) && \text{Current position of the agent} \\
M_{t+1} &= update(M_t, w_{t+1}^{(x_t,y_t)}) && \text{Update neural map } M_t \\
o_t &= [r_t, c_t, w_{t+1}^{(x_t,y_t)}] \\
\pi_t(a|s) &= \mathrm{Softmax}(f(o_t)),
\end{aligned}
$$

Neural Map


$\pi_{t-1}(a \mid s)$
$\pi_t(a \mid s)$
current action a_t
M_t
Current map
M_{t+1}
$w_t^{(x_{t-1}, y_{t-1})}$
$w_{t+1}^{(x_t, y_t)}$
write information to the next map
c_{t-1}
c_t
context vector c_t
r_{t-1}
r_t
read vector r_t
s_{t-1}
s_t
Current state s_t

Neural Map

ROBOT INTELLIGENCE LAB



SPTM

ROBOT INTELLIGENCE LAB

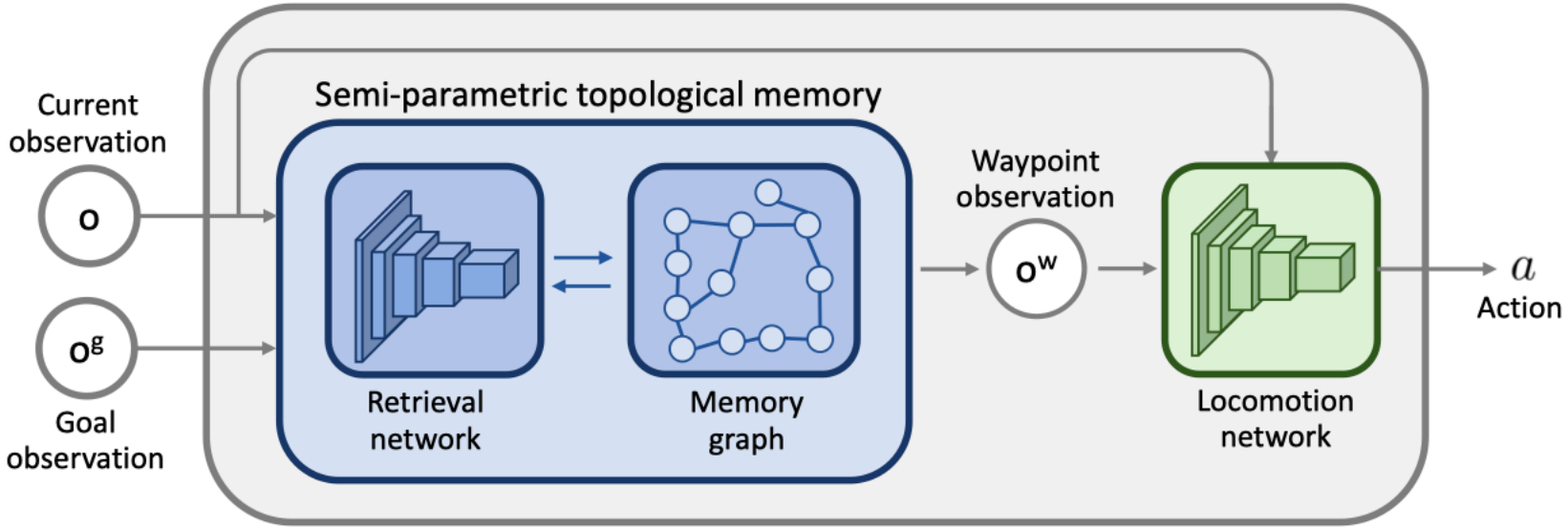
- "Semi-Parametric Topological Memory for Navigation," 2018
  - It introduces a **memory architecture** for navigation inspired by landmark-based navigation in animals.
  - Semi-parametric topological memory (SPTM) consists a **(non-parametric) graph** with nodes corresponding to locations in the environments and a **(parametric) deep network** capable of retrieving nodes from the graph based on observations.
  - The SPTM is used as a **planning module** in a navigation system.
  - It is a two-staged method:
    - **exploration**: records the traversal of the environment and build the internal representation.
    - **goal-directed navigation**: use the internal representation to reach the goal location.

SPTM

ROBOT INTELLIGENCE LAB

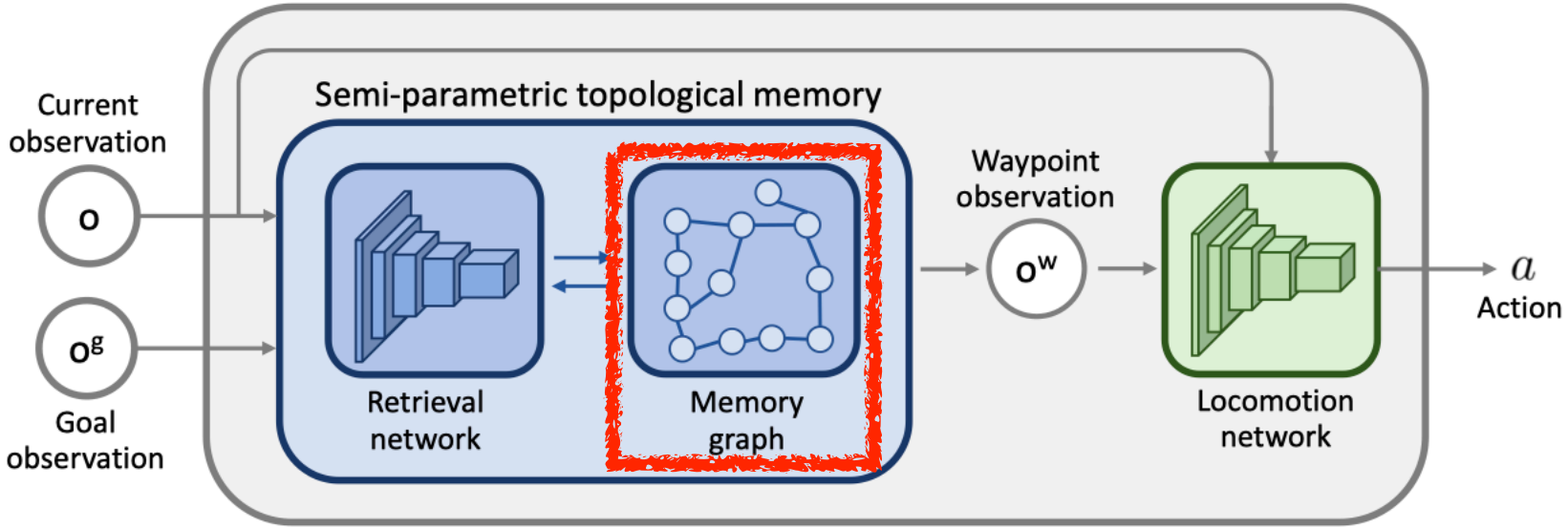
- **Landmark based navigation**
  - Animals do not rely strongly on metric representations, rather, animals employ a range of specialized navigation strategies such landmark navigation.
  - Landmark navigation is the ability to orient with respect to a known object (topological knowledge of the environment).
- **Semi-parametric topological memory (SPTM)**
  - A deep learning based memory architecture for navigation.
  - A non-parametric memory graph G
    - Each node corresponds to location in the environment.
    - The graph does not contain metric information.
  - A parametric deep network R
    - It retrieves nodes from the graph based on observations.

SPTM



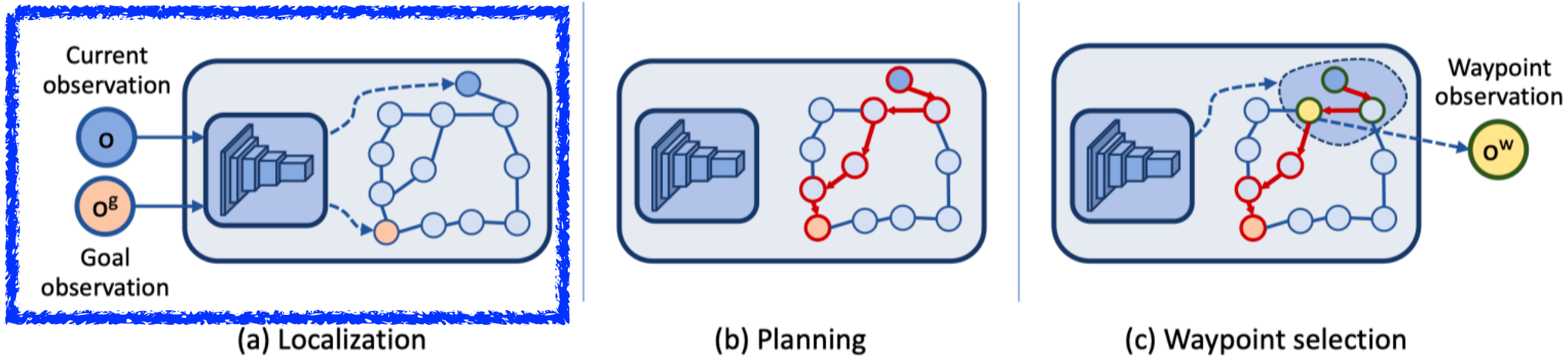
- At each time step t, the agent gets an observation o_t and takes an action a_t.
- The interaction is set up in two stages: **exploration** and **goal-directed navigation**.
- In the above figure, SPTM acts as a **planning module**: given the current and goal observations, it generates a waypoint and the corresponding action.

SPTM



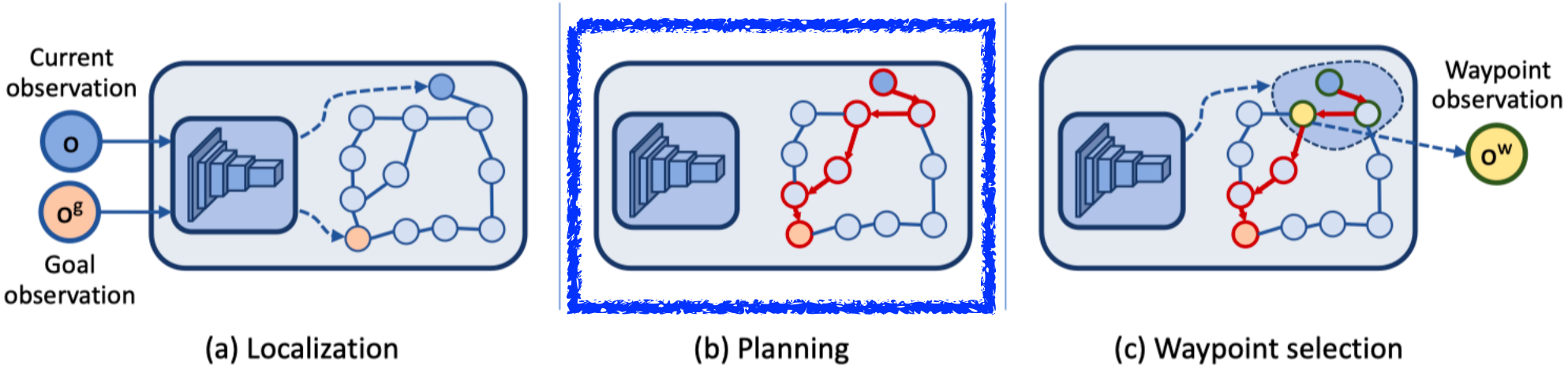
- **Memory graph**
  - The graph is populated based on an **exploration sequence** provided to the agent.
  - Two vertices are connected by an edge in one of two cases: if they correspond to consecutive time steps, or if the observations are very close, as judged by the retrieval network .

SPTM



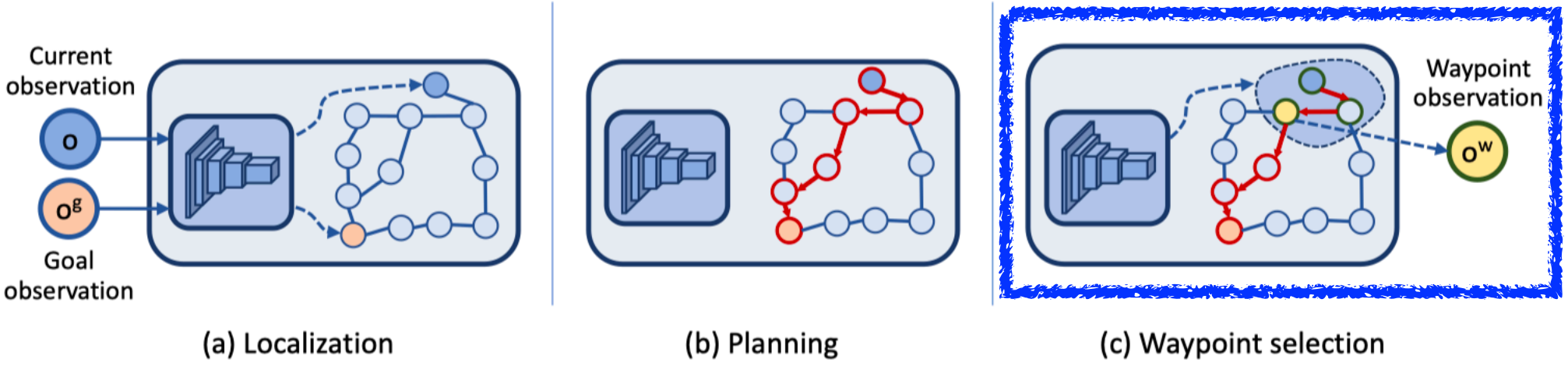
- The retrieval network R localizes in the graph the vertices v^a and v^g, corresponding to the current agent's observation o and the goal observation o^g, respectively.
- The network R estimates the similarity of two observations (o_1, o_2) trained on a set of environments in self-supervised manner.

SPTM



- The shortest path on the graph between these vertices is computed (red arrows).
- Dijkstra's algorithm is used in the experiments.

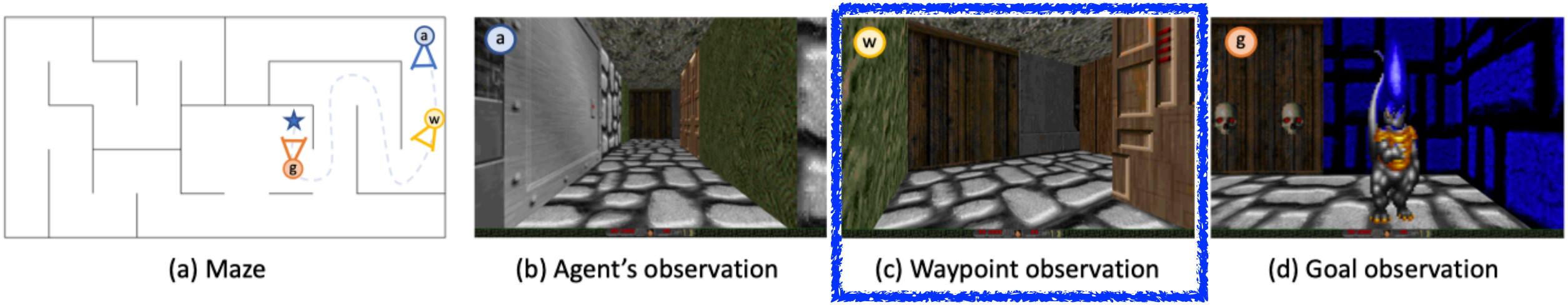
SPTM



- The waypoint vertex v^w (yellow) is selected as the vertex in the shortest path that is furthest fro the agent's vertex v^a but can still be confidently reached by the agent.
- The output of the SPTM is the corresponding waypoint observation $o^w = o_{v^w}$.

SPTM

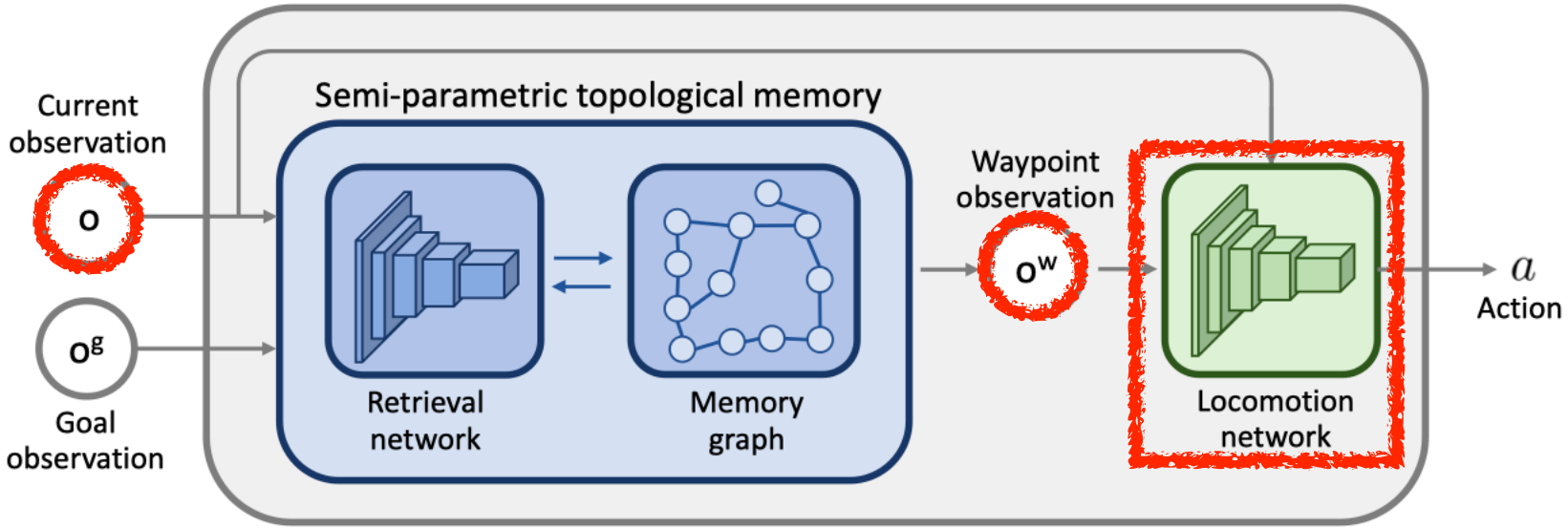
ROBOT INTELLIGENCE LAB



(a) Maze (b) Agent's observation (c) Waypoint observation (d) Goal observation

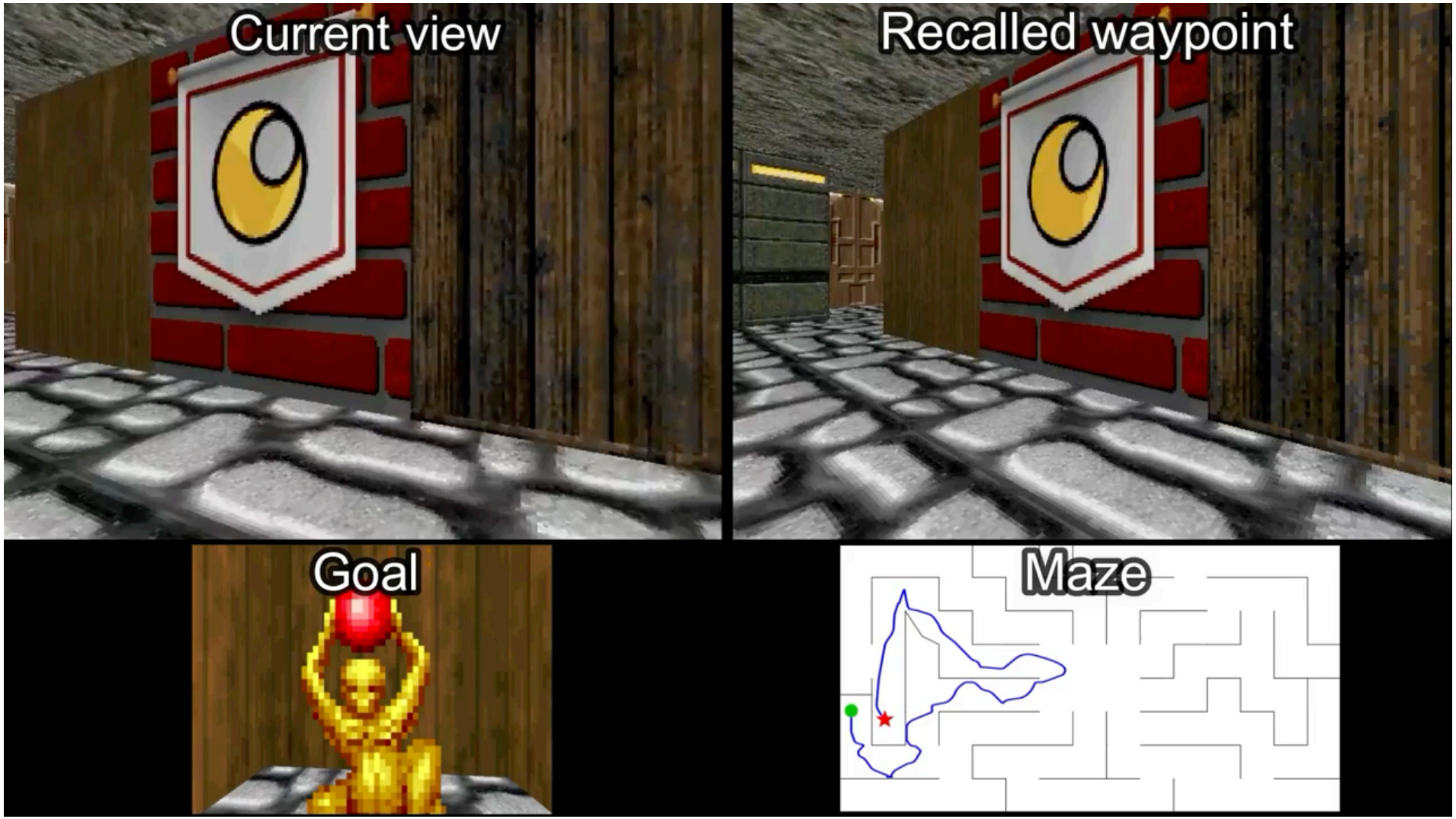
- A waypoint observation is produced by STPM given agent's observation (b) and goal observation (d).

SPTM



- **Locomotion network L:**
  - The network L is trained to navigate toward **target observations** near the agent.
  - The network maps a pair (o_1, o_2), which consists of a current and a goal observations, into action probabilities.

SPTM



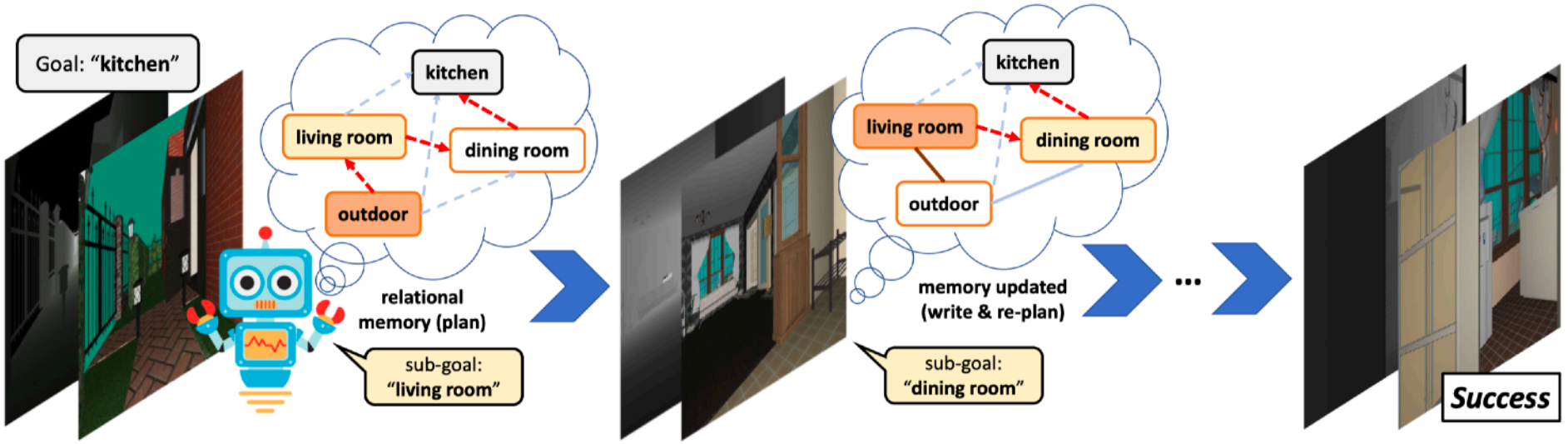
Bayesian Relational Memory

ROBOT INTELLIGENCE LAB

- "Bayesian Relational Memory for Semantic Visual Navigation," 2019 (UCB, FAIR, Technion, OpenAI)
  - **Bayesian Relational Memory** (BRM) is presented to (1) capture the **layout prior** from training environments, (2) estimate **posterior layout** at test time, and (3) **efficient planning** for navigation.
  - The BRM agent consists of (1) a BRM module that produces sub-goals and (2) a goal-conditioned module for control.
  - This paper adapts the **topological graph** approach (instead of spatial memory approaches) where the **vertices** are landmarks in the environments and **edges** denote short-term reachability between landmarks.

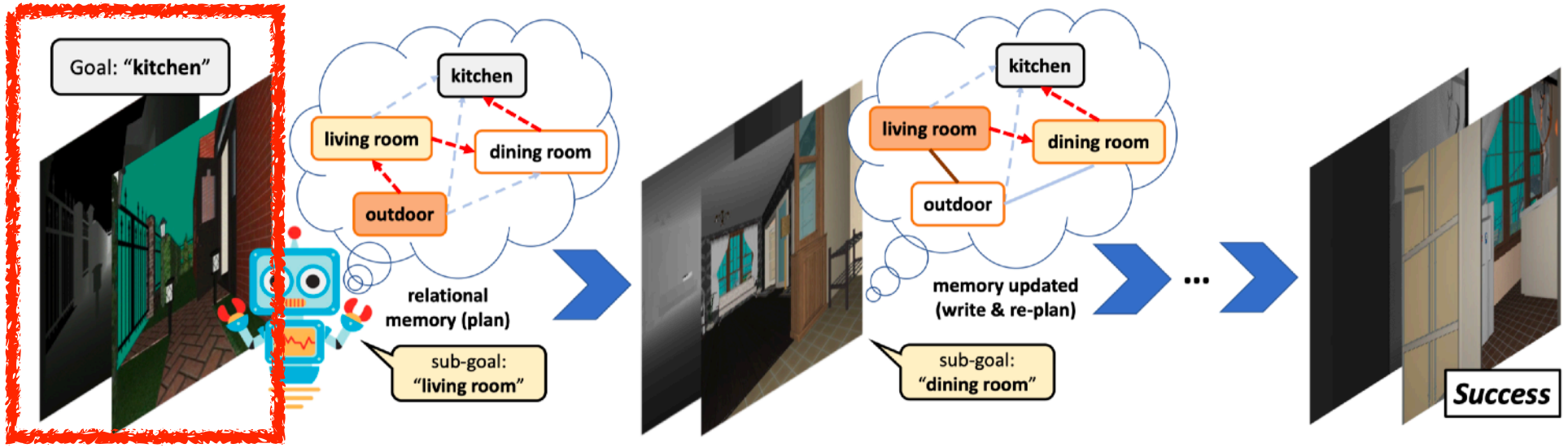
Bayesian Relational Memory

ROBOT INTELLIGENCE LAB



- BRM can be viewed as a probabilisitic version of a topological memory with semantic abstractions (each node in BRM denotes a semantic concept such as a room type detected via a neural detector.

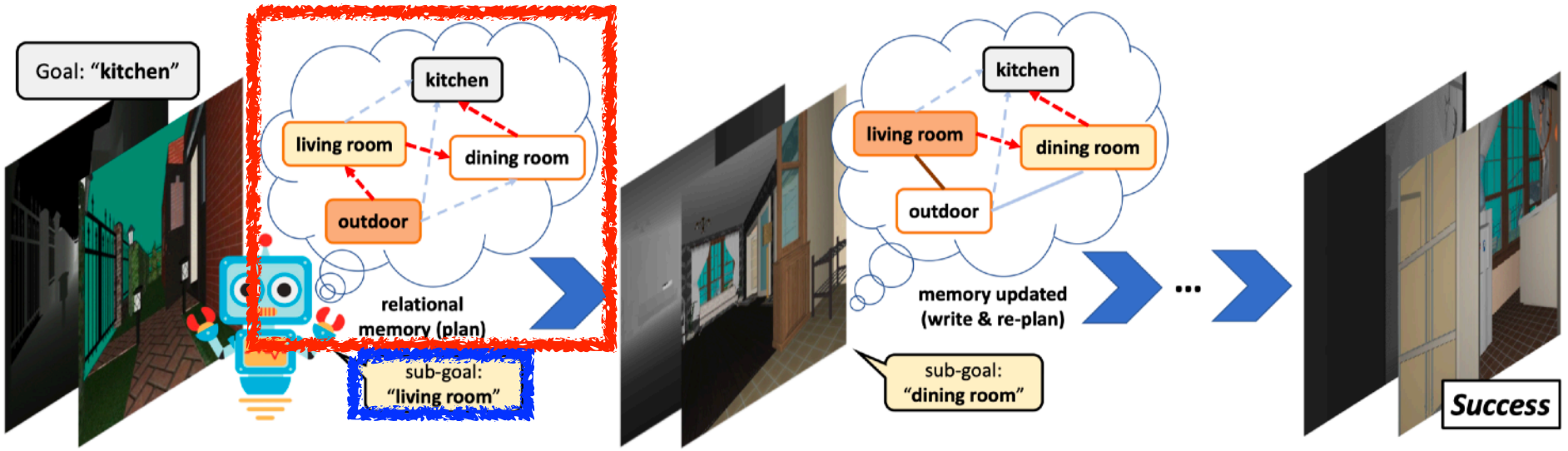
Bayesian Relational Memory



- The agent perceives visual signals and **needs to find the kitchen** which cannot be seen from outside.

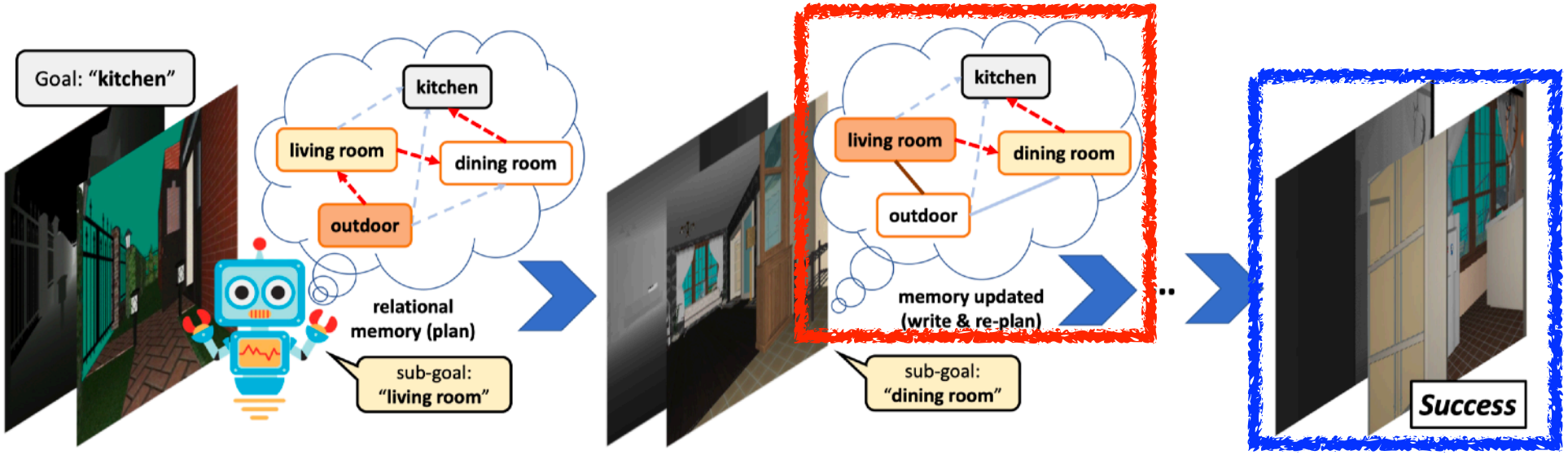
Bayesian Relational Memory

ROBOT INTELLIGENCE LAB



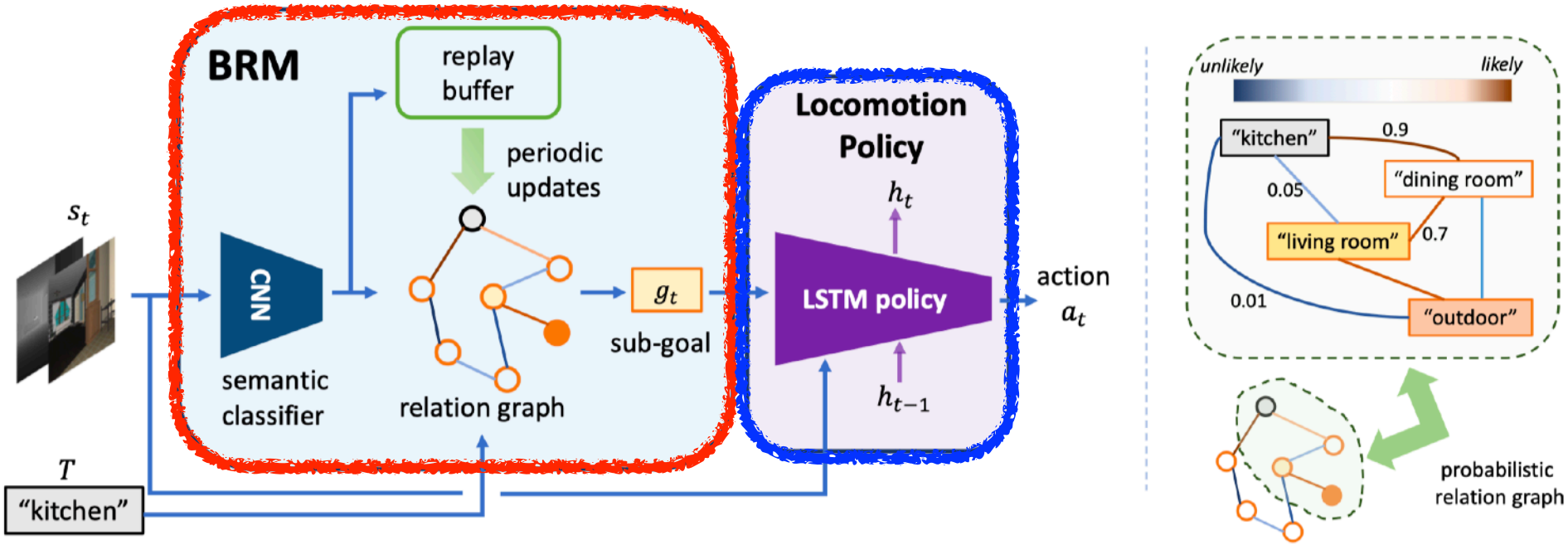
- The agent perceives visual signals and **needs to find the kitchen** which cannot be seen from outside.
- So the agent plans in its memory and conclude that it may first find a likely intermediate waypoint (i.e., living room).

Bayesian Relational Memory



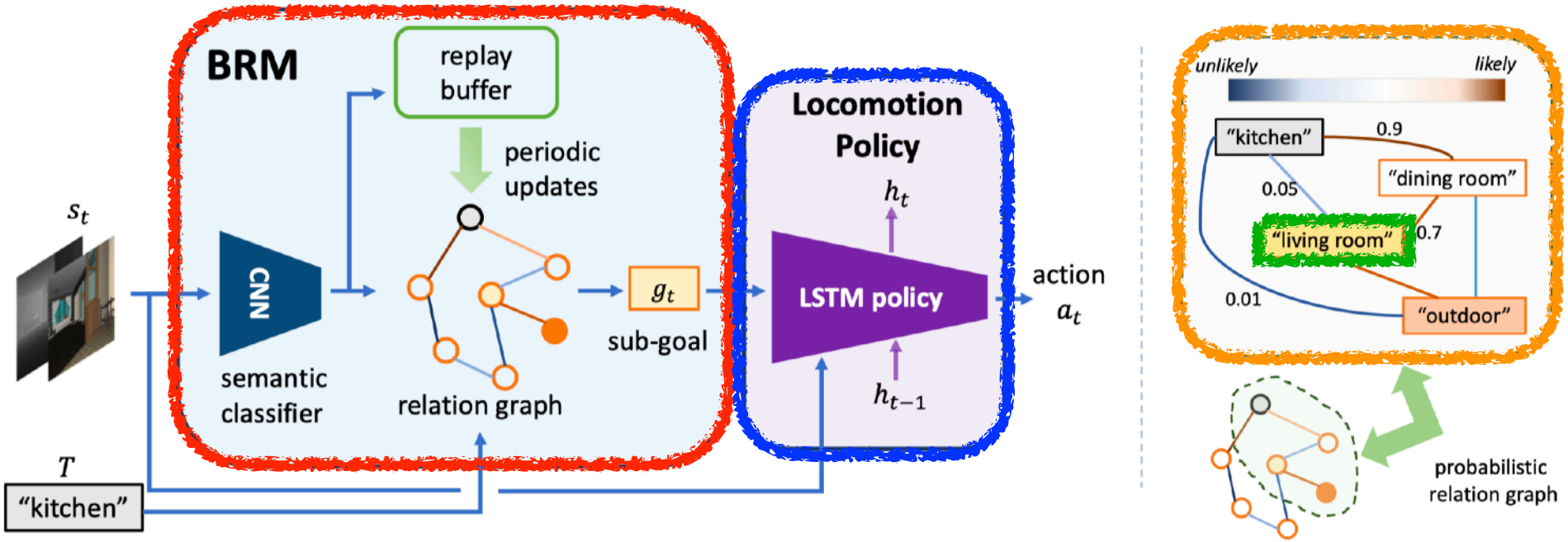
- The agent repeatedly **updates its belief** of the environment layout, re-plans accordingly and **reaches the kitchen** in the end.

Bayesian Relational Memory



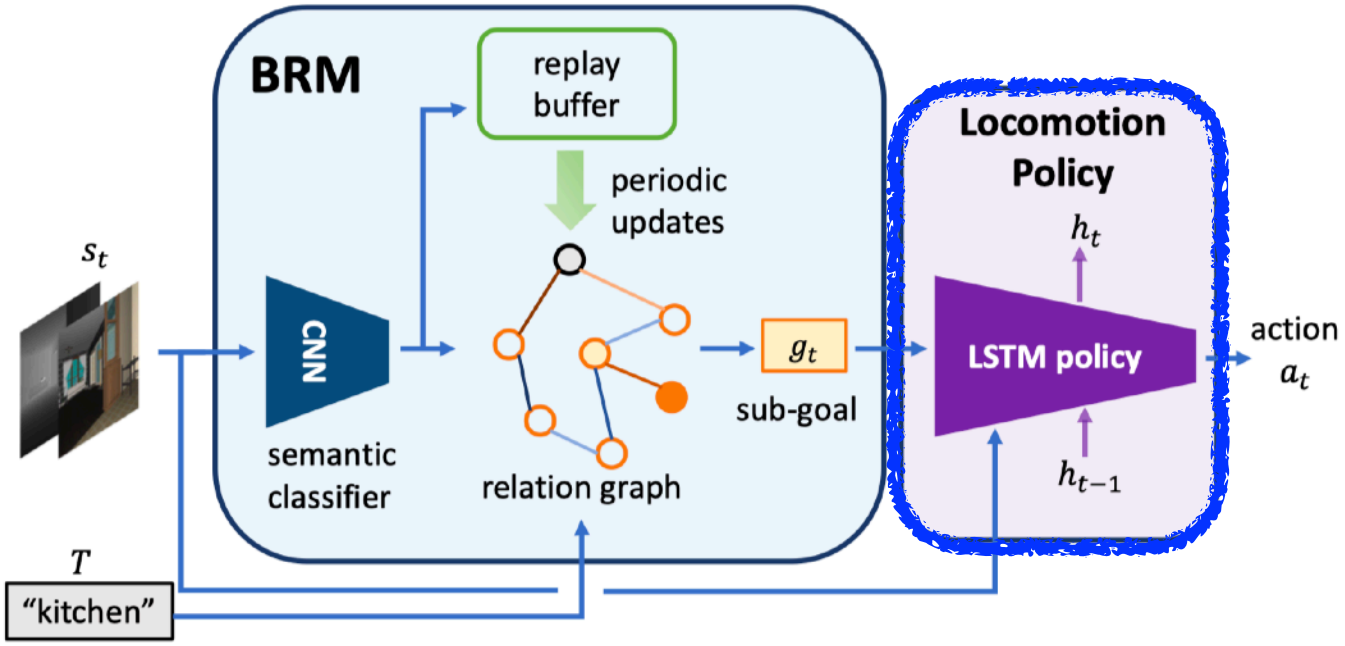
- IT has two modules, the **BRM** as well as an **LSTM locomotor policy**.

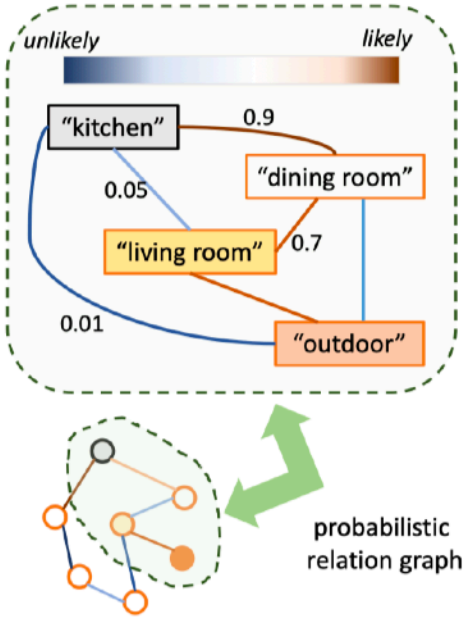
Bayesian Relational Memory



- IT has two modules, the BRM as well as an LSTM locomotor policy.
- The key component of a BRM is a probabilisitic relation graph where each node corresponds to a particular semantic target. and the edge denotes the "close-by" relation.

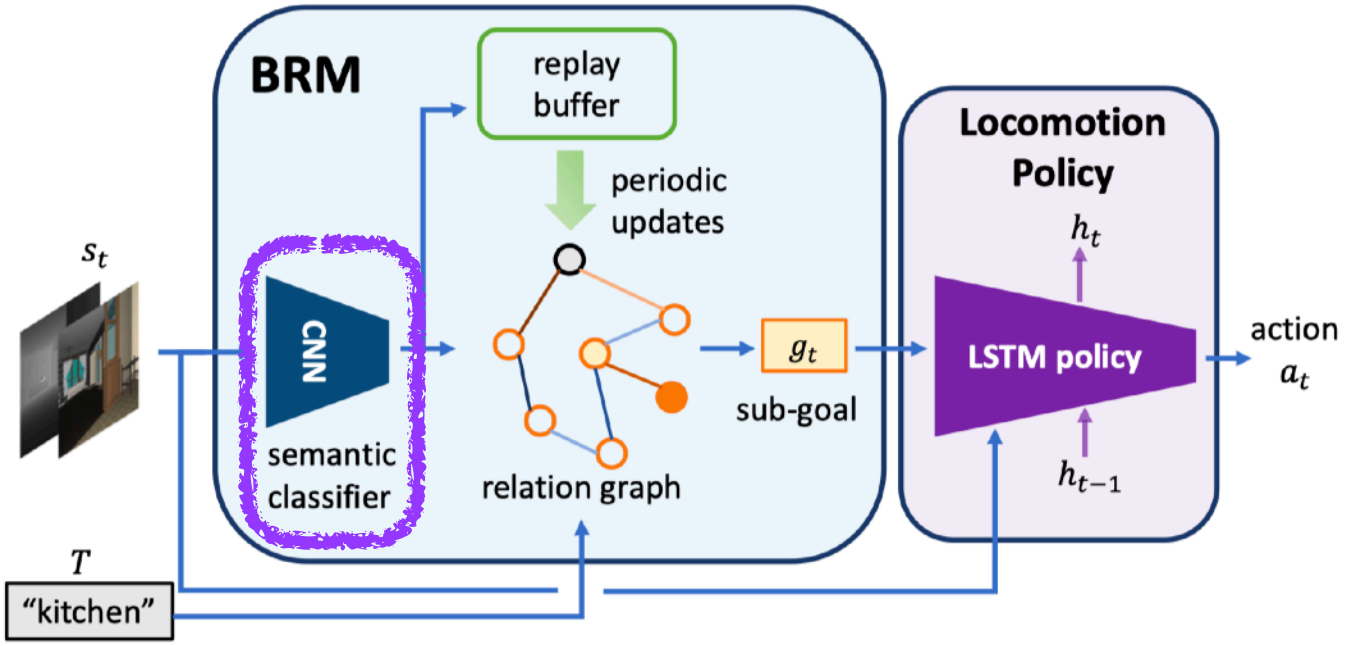
Bayesian Relational Memory

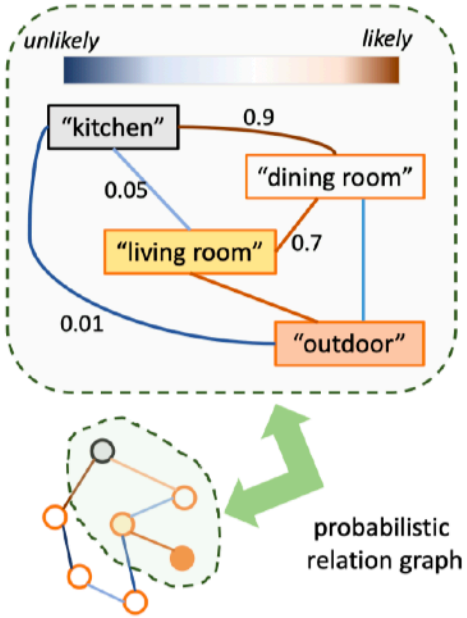




- The **locomotion policy** is conditioned on the current visual input s_t and the sub-goal (semantic target) $g \in \mathcal{T}$ and emits an action a_t.

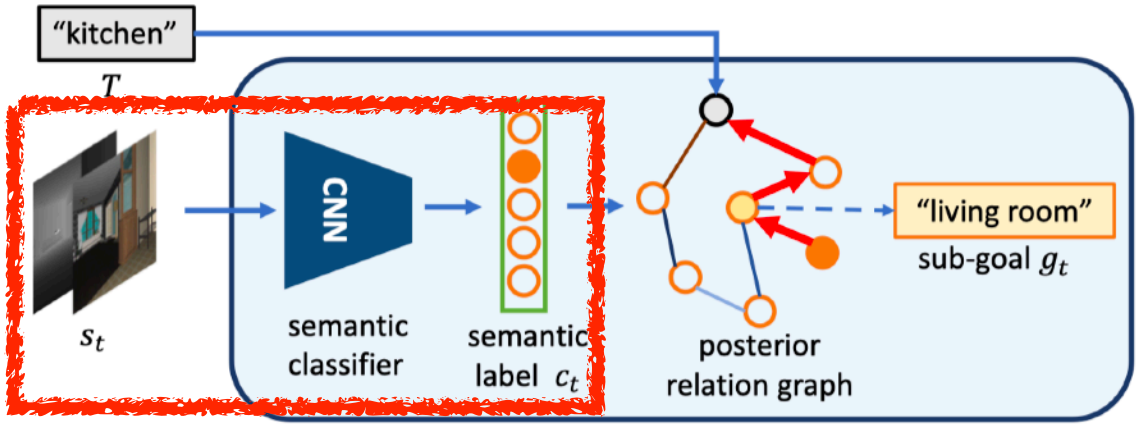
Bayesian Relational Memory

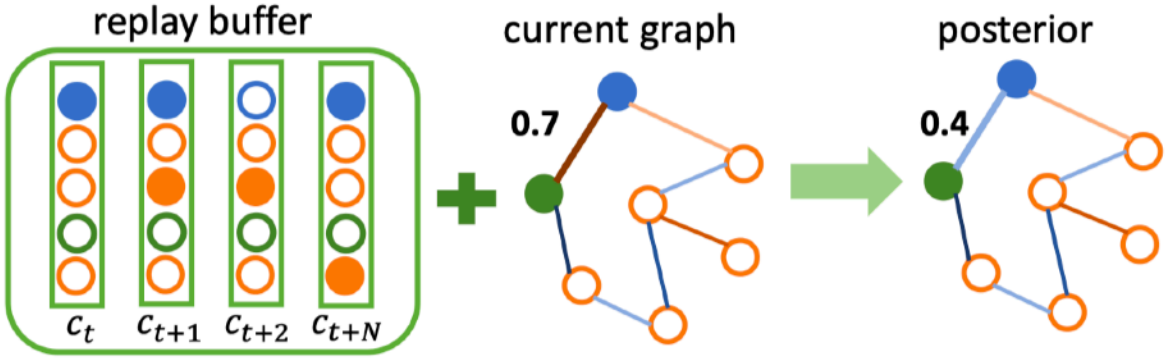




- A semantic classifier extracts the **semantic information** from the visual observation s_t at each time step.

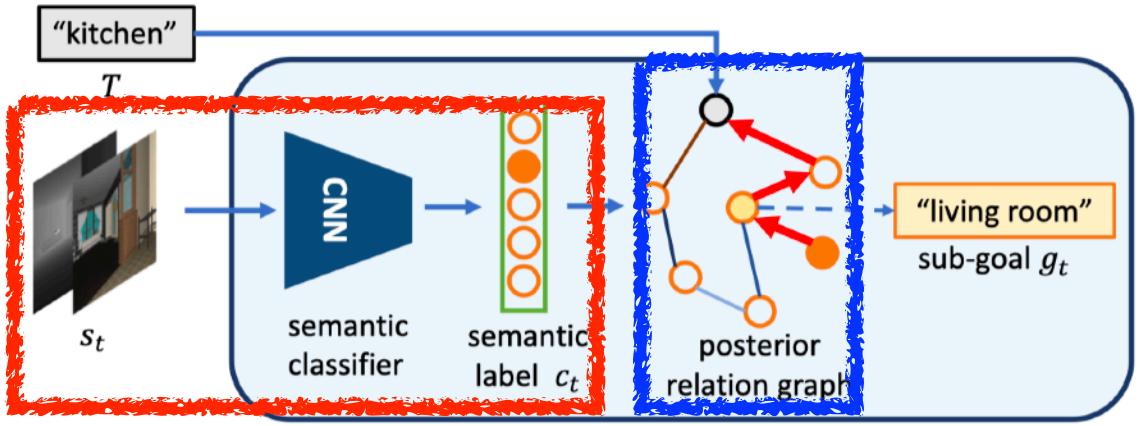
Bayesian Relational Memory

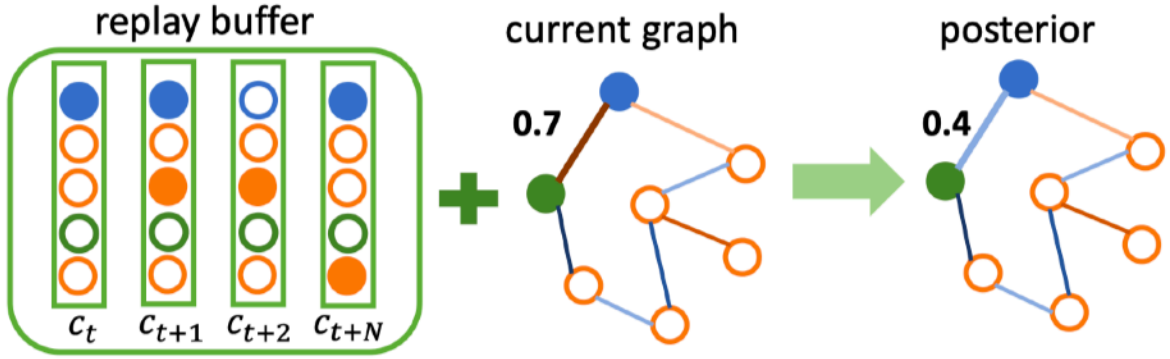




- At each time step, the agent detects the **surrounding room type**,

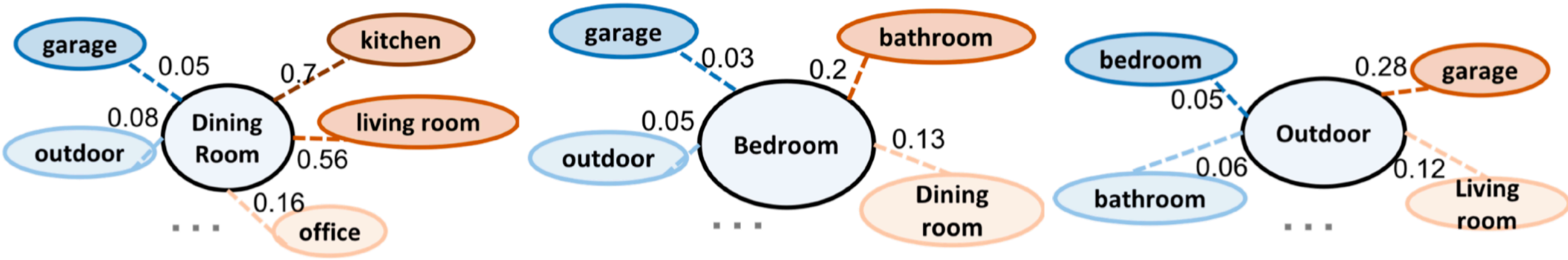
Bayesian Relational Memory





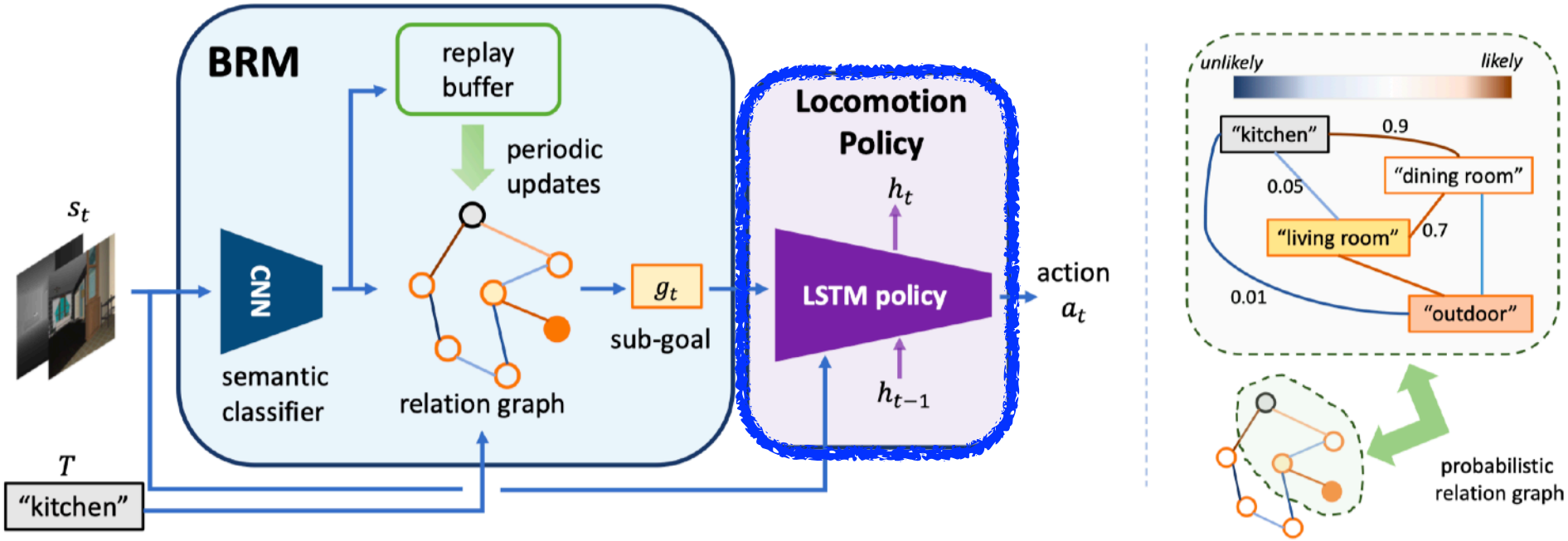
- At each time step, the agent detects the **surrounding room type**, and maintains the probability of whether two room types T_i and T_j are **nearby in the current environment**.

Bayesian Relational Memory



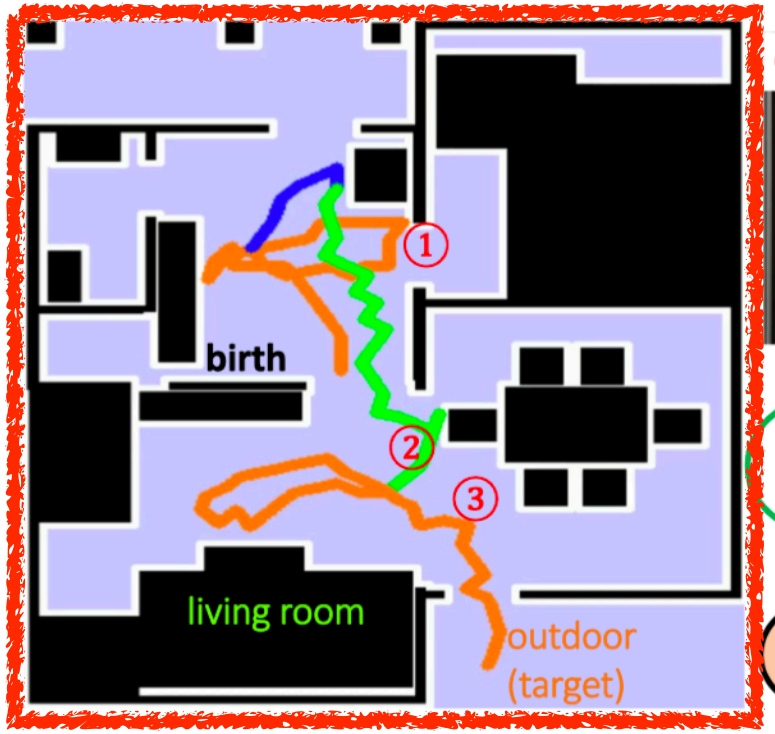
- At each time step, the agent detects the **surrounding room type**, and maintains the probability of whether two room types T_i and T_j are **nearby in the current environment**.
- The **semantic concepts** are predefined (K nodes and $K(K-1)/2$ edges) and the prior of the relations are pre-trained.

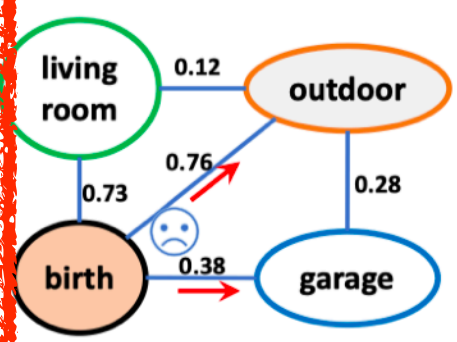
Bayesian Relational Memory

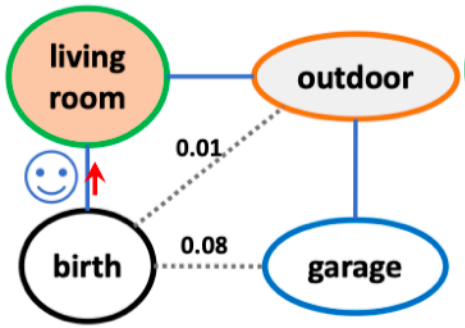


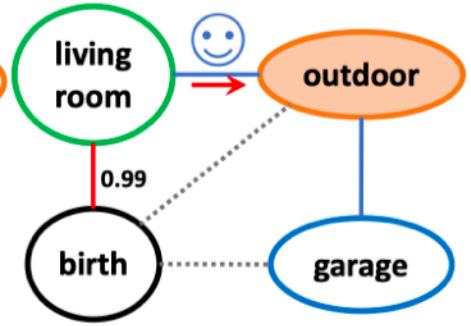
- The **goal-conditioned policy** that navigates towards the goal g is trained using a deep reinforcement learning.

Experiments



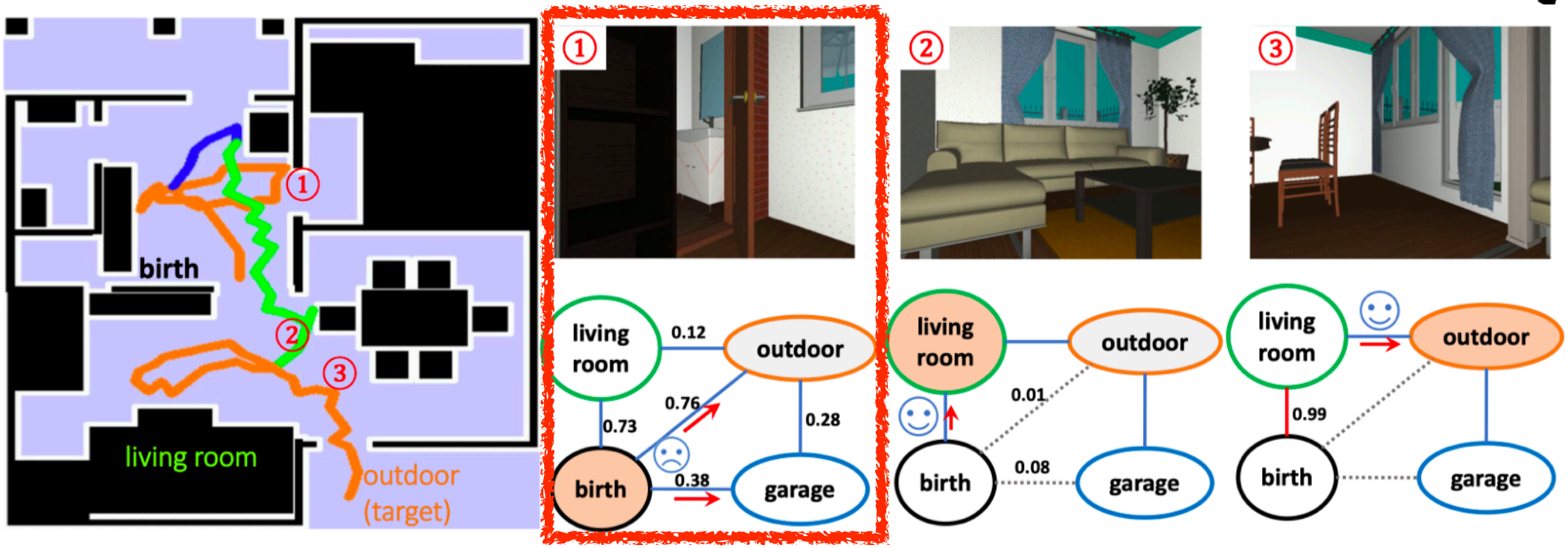






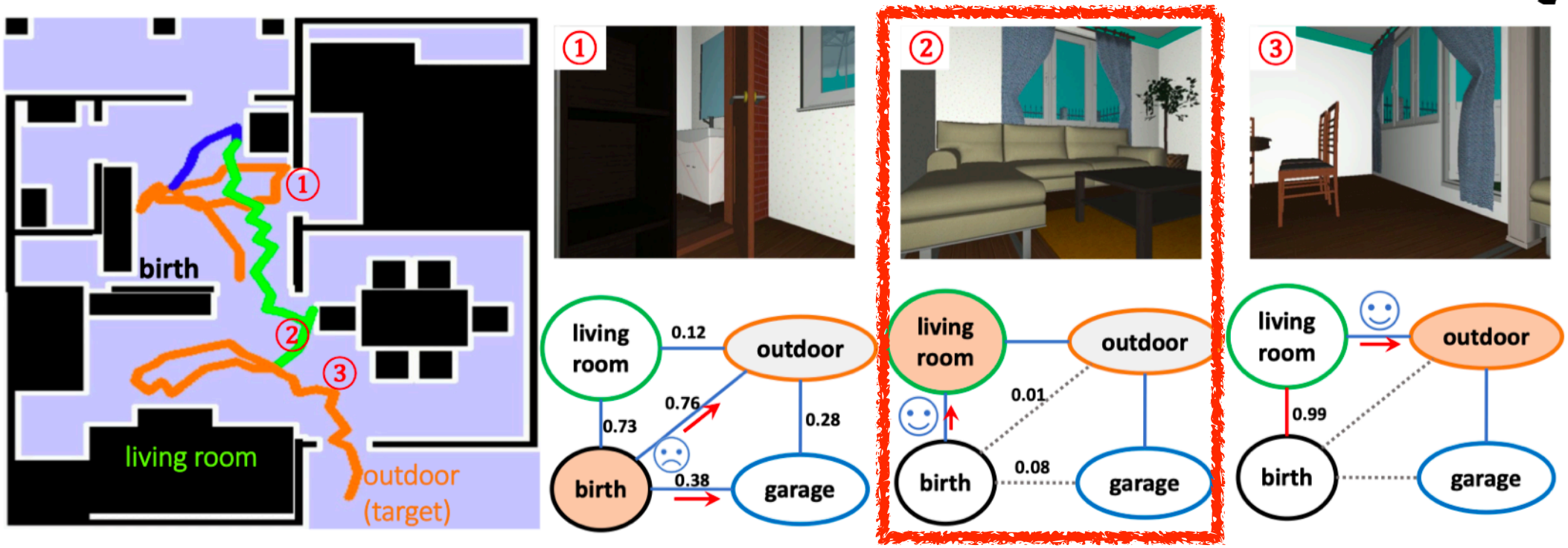
- The agent is spawned inside the house, targeting "**outdoor**".

Experiments



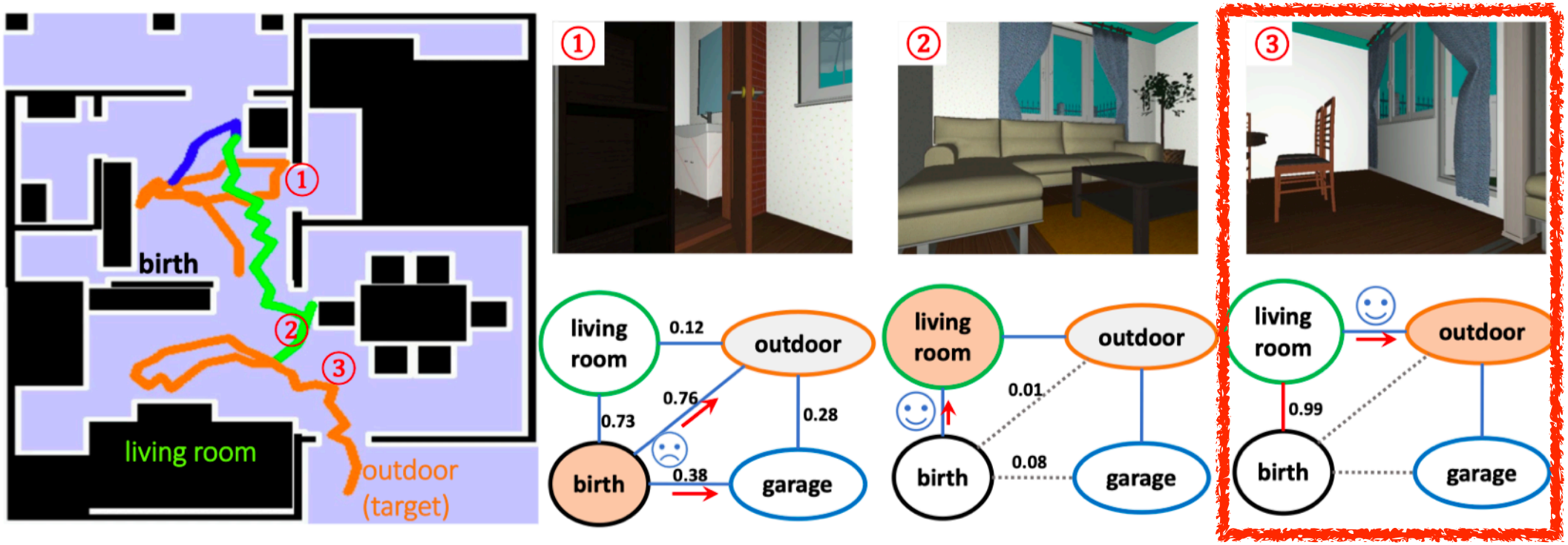
- Initially, the agent starts by executing the locomotion for "**outdoor**" and then "**garage**" according to **the prior knowledge**.

Experiments



- However, it fails and it updates its belief that "garage" and "outdoor" are not nearby.
- It then executes locomotion for "**living room**".

Experiments



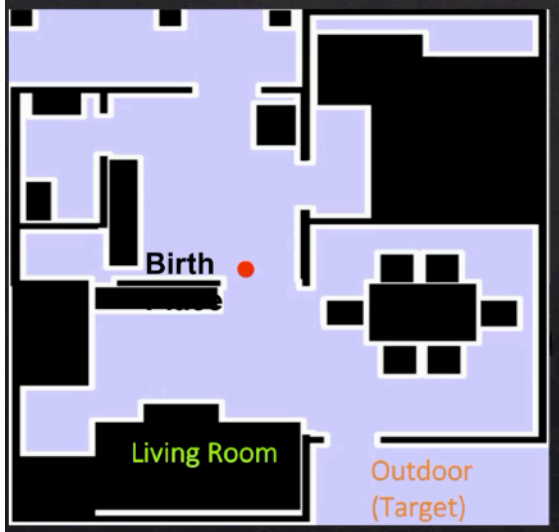
- Finally, it executes sub-policy for "**outdoor**" again, explores the living room and reaches the goal.

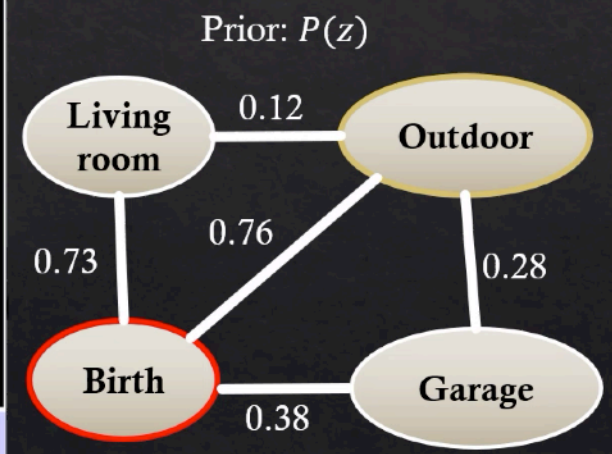
Experiments

ROBOT INTELLIGENCE LAB

Demo

- A case study
  - Go to "outdoor"





ROBOT INTELLIGENCE LAB

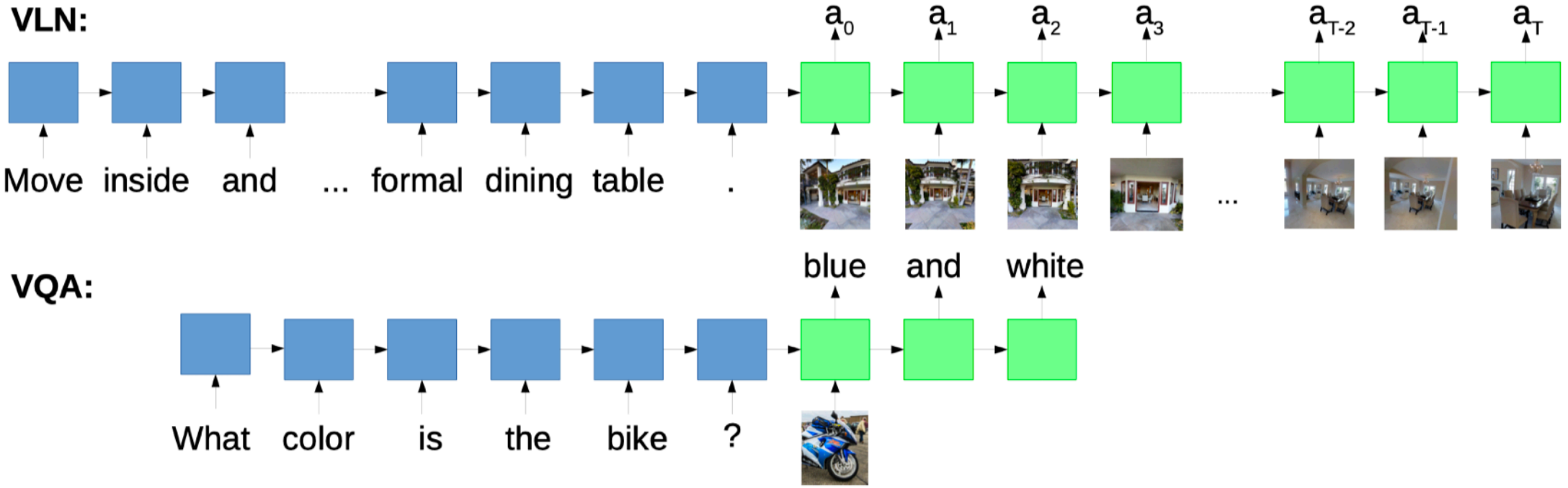
Vision-Language Navigation

"Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments," 2018

Vision-Language Navigation

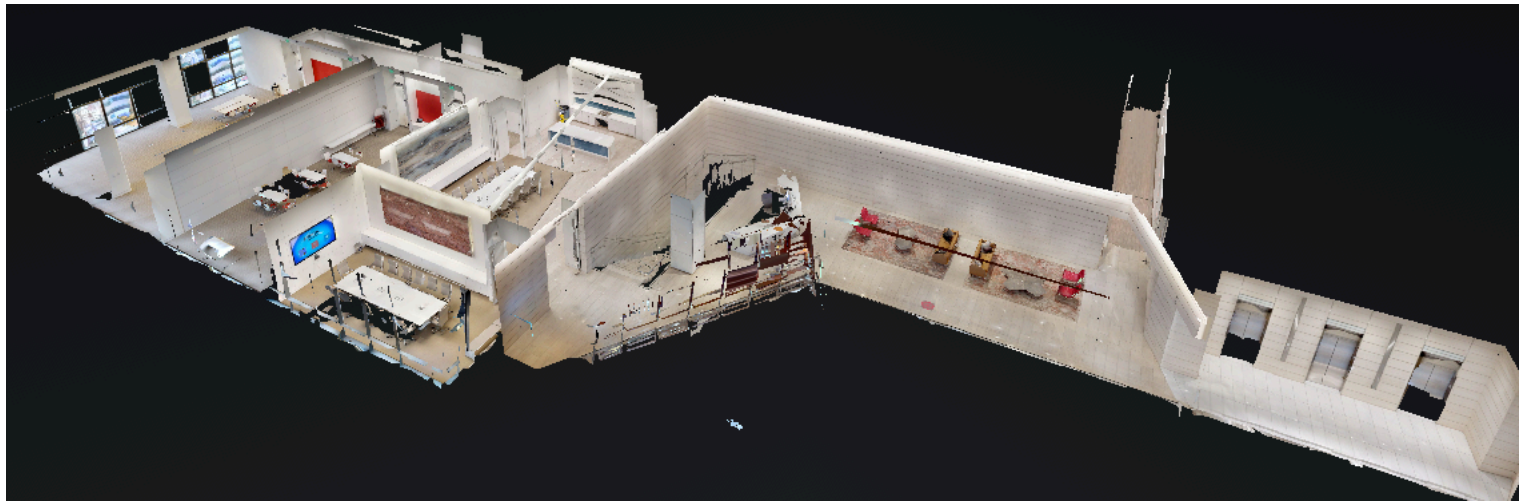
- "Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments," 2018
  - A robot that can carry out a **natural-language instruction** has been a long dream that remains stubbornly distant.
  - This problem of a robot interpreting a natural-language navigation instruction on the basis of what it sees is similar to **Visual Question Answering (VQA)**.
  - The **Matterport3D simulator**, a large-scale RL environment based on real imagery, is presented.

Vision-Language Navigation



- **VLN** vs. **VQA**:
  - the decoder of **VLN** generates actions conditioned on current observations.

Matterport3D

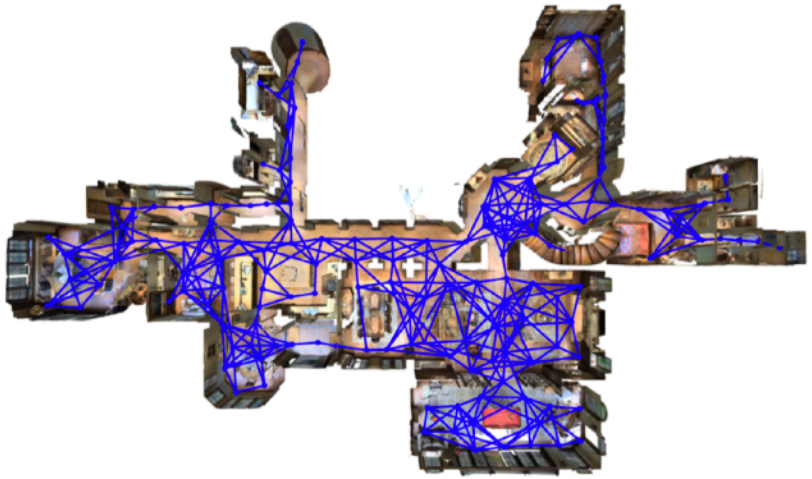


- **Matterport3D Dataset**
  - It consists of 10,800 panoramic views constructed from 194,400 RGB-D images of 90 building-scale scenes.
  - Panoramic viewpoints are distributed throughout the walkable floor plan at an average separation of 2.25m.

Matterport3D

ROBOT INTELLIGENCE LAB

- Simulator
  - **Observation**: agent poses are defined in terms of **3D position**, **heading**, and **camera elevation** angle (i.e., $\mathbb{R}^3 \times [0,2\pi] \times [-\frac{\pi}{2}, \frac{\pi}{2}]$) and at each time step, the simulator outputs an **RGB image observation**.
  - **Action**: action is selected among the reachable waypoints from the navigation graph.

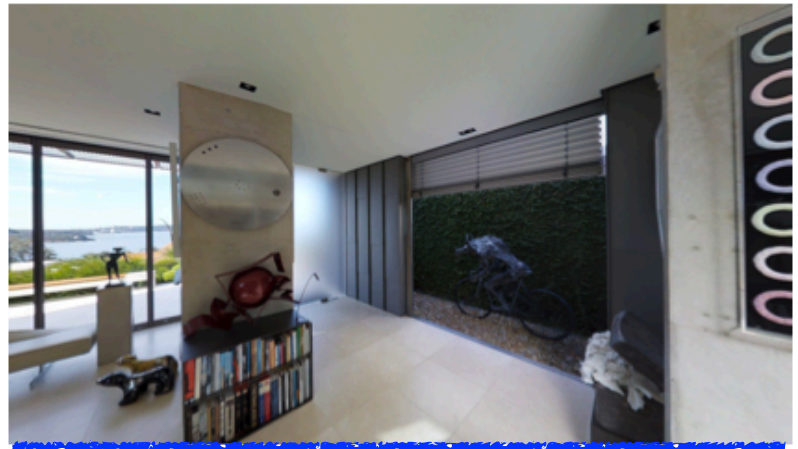


Navigation graph

https://github.com/peteanderson80/Matterport3DSimulator

Vision-Language Navigation

- **Room-to-Room (R2R) Navigation**
  - At the beginning, a natural language instruction is given.
  - The agent observes an RGB images, makes an action, leading to a new pose, and observes a new image.
  - The episode ends when the agent selects the special **stop** action.
  - The dataset is collected using AMT with an interactive 3D WebGL environment.



Go up the stairs and turn right. Go past the bathroom and stop next to the bed.

Walk all the way up the stairs, and immediately turn right. Pass the bathroom on the left, and enter the bedroom that is right there, and stop there.

Walk up the stairs turn right at the top and walk through the doorway continue straight and stop inside the bedroom.

Vision-Language Navigation

ROBOT INTELLIGENCE LAB

Leave the bedroom, and enter the kitchen. Walk forward, and take a left at the couch. Stop in front of the window.

ROBOT INTELLIGENCE LAB

Path Following

"Visual Memory for Robust Path Following," 2018

"Combining Optimal Control and Learning for Visual Navigation in Novel Environments," 2019

"Deep Visual MPC-Policy Learning for Navigation," 2019

Robust Path Following

ROBOT INTELLIGENCE LAB

- "Visual Memory for Robust Path Following," 2018 (UCB)
  - Given a demonstration of a path, a **first network** generates a **path abstraction**.
  - Equipped with this **abstraction**, a **second network** observes the worlds and decides how to **act to retrace the path** under noisy actuation and a changing environment.
  - Both **networks** are optimized end-to-end at training time.

Robust Path Following

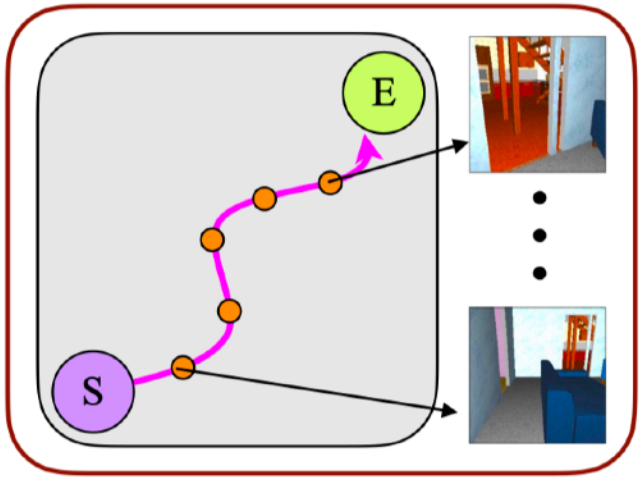
ROBOT INTELLIGENCE LAB

- **Human's ability to Retrace**
  - Consider the first morning of a conference in a city you have never been to. Rushing to the first talk, you might follow your phone's direction through a series of twists and turns to reach the venue.
  - When you return later in the day, you can **retrace** your steps to your hotel relatively robustly, remembering to take a left turn at the bistro and keep straight past the coffee shop.
  - The next day, you may probably only look at your phone to check your email.

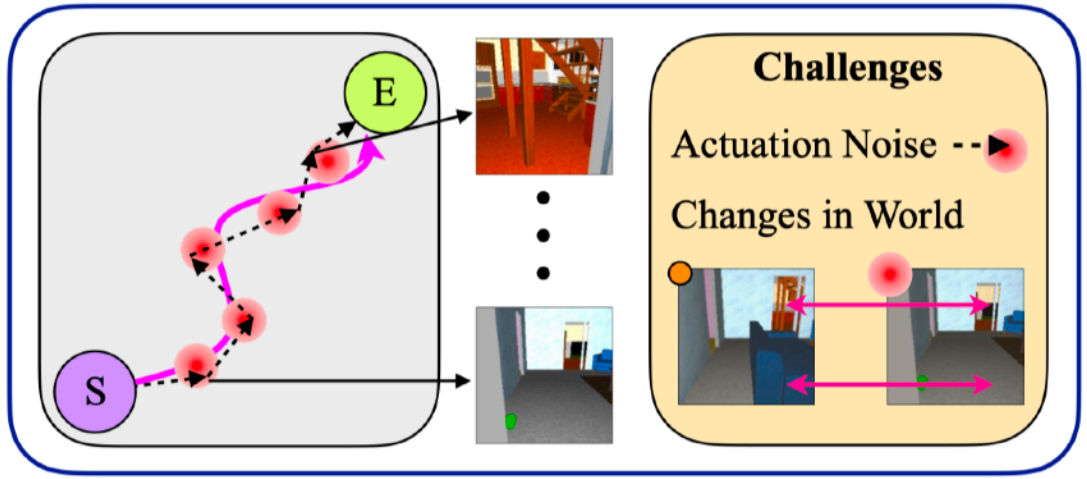
Robust Path Following

ROBOT INTELLIGENCE LAB

- **How to solve?**
  - One classical approach is to build a full 3D model of the world via SLAM.
  - However, for the task of navigation, this might be an overkill.
  - A large number of learning-based approaches have sprung up. The proposed approach only rely on a **single demonstration** in a new environment.

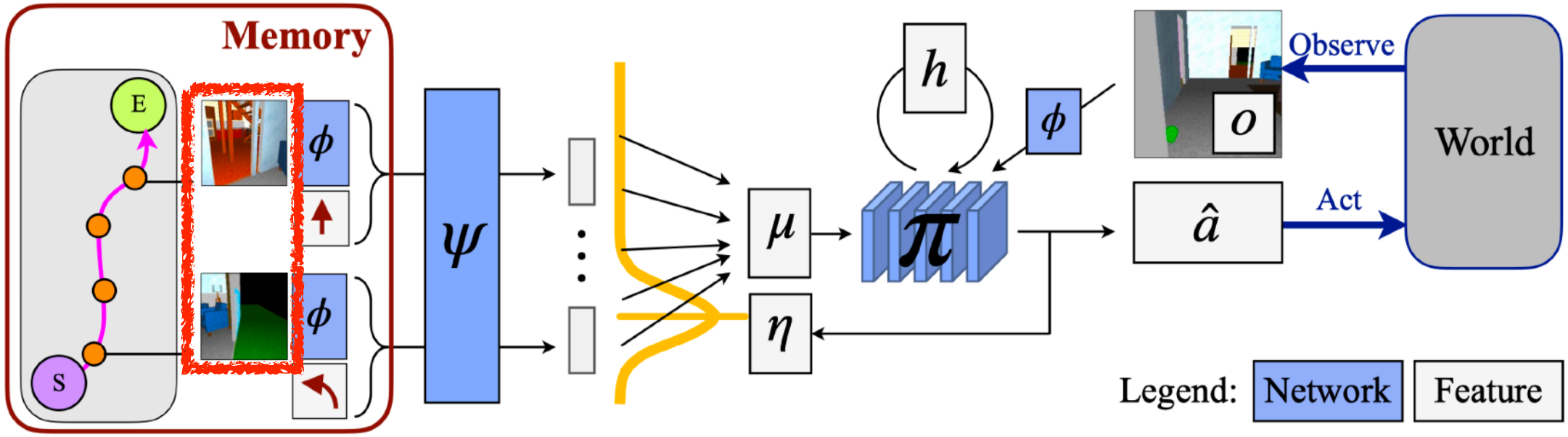


Demonstration



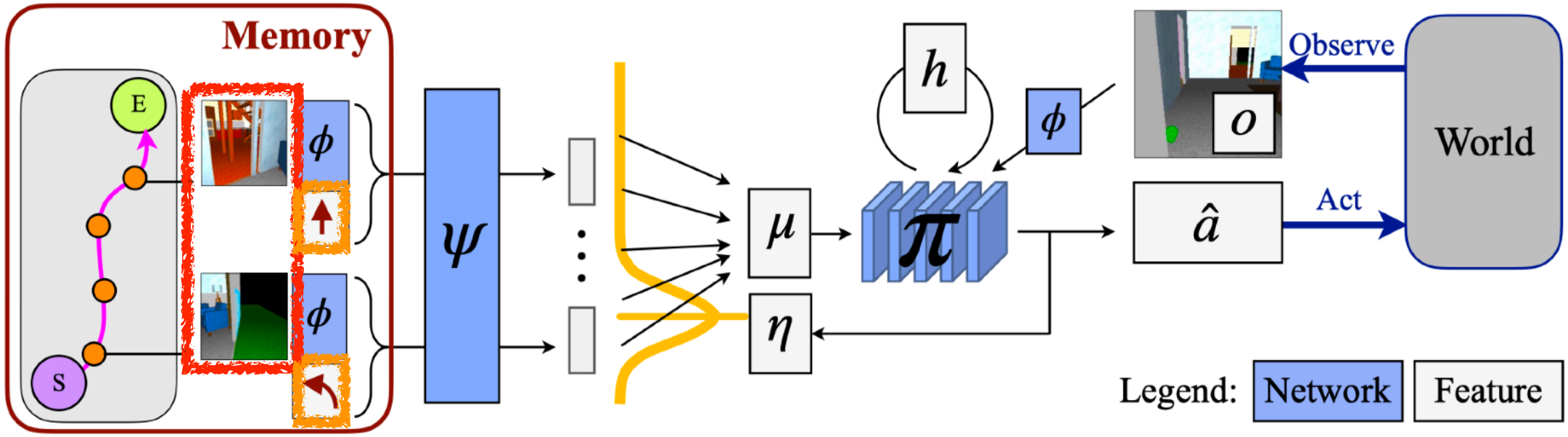
Execution under noisy actions and changes in the world.

Robust Path Following



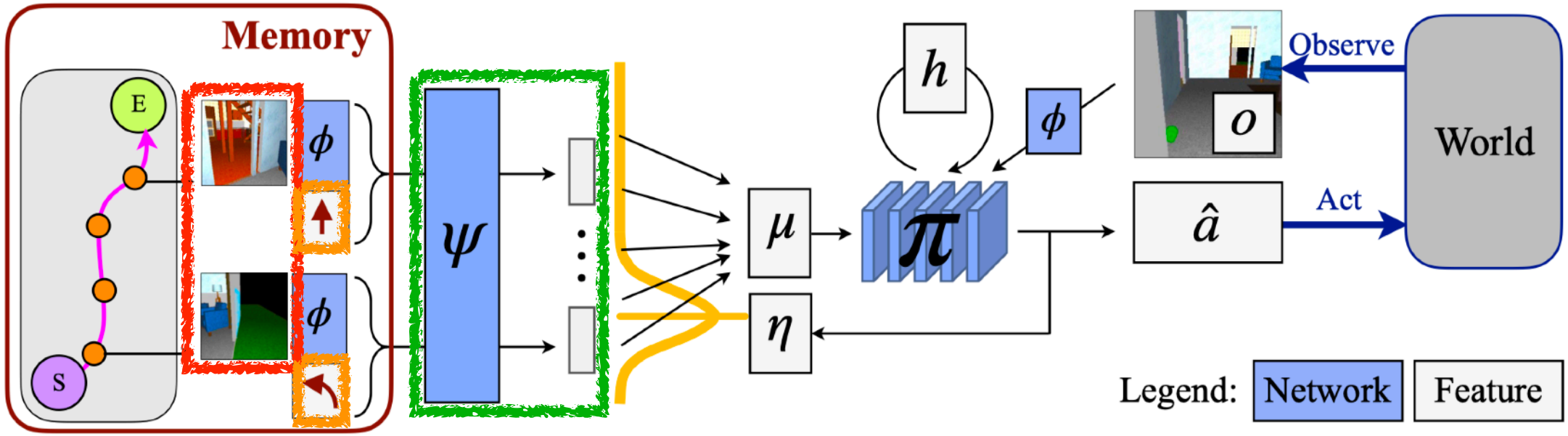
- Given a sequence of images

Robust Path Following



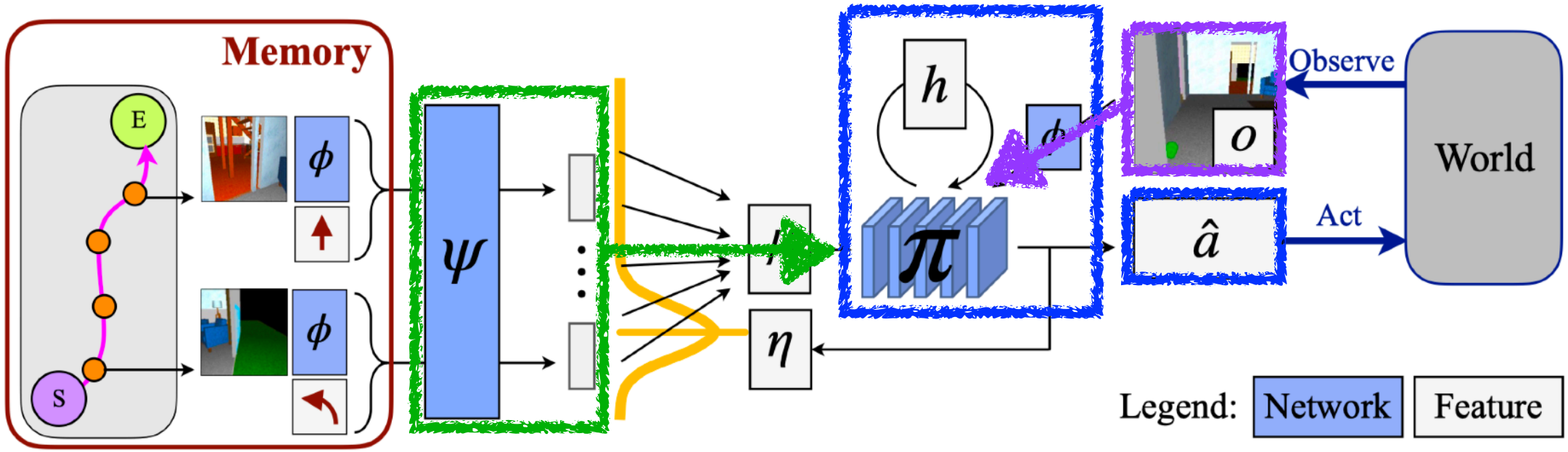
- Given a sequence of images and actions at these images

Robust Path Following



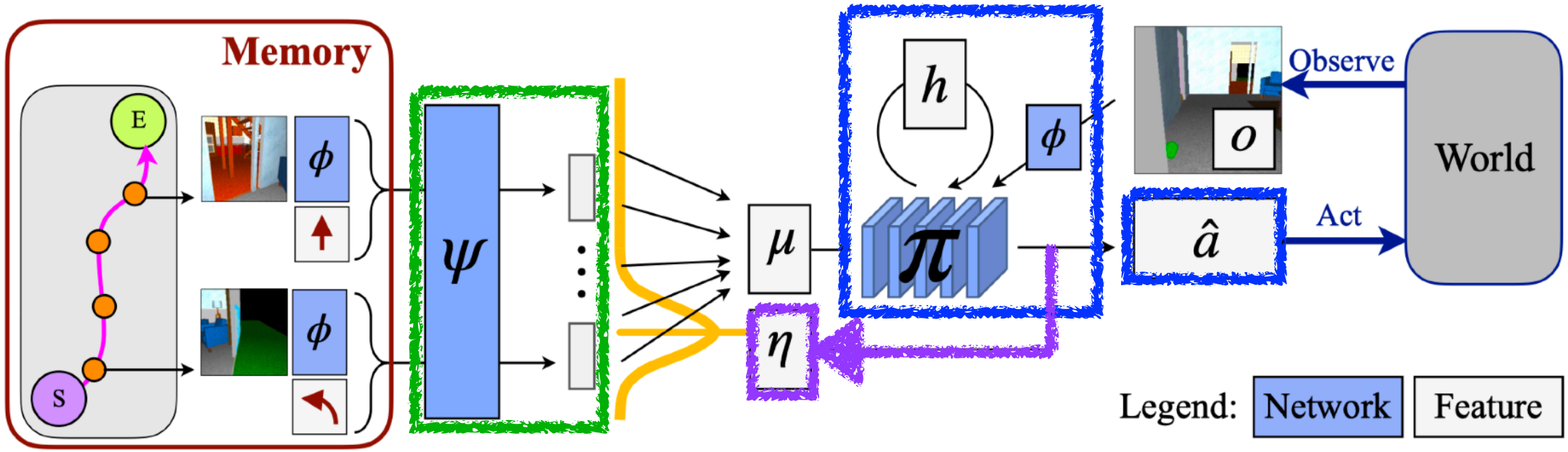
- Given a sequence of images and actions at these images, it abstracts the sequence into a sequence of memories.

Robust Path Following



- A second recurrent network π uses this sequence and the current observation to emit actions that **retrace the path**.

Robust Path Following



- A second recurrent network π uses this sequence to emit actions that **retrace the path**. It also updates the attention location η.

Robust Path Following

ROBOT INTELLIGENCE LAB

- **Problem setup**
  - Consider an agent that operates in a new environment that it has **never been in before**.
  - At each time step t, the agent is in s_t and has some action primitive set $\mathscr{A}$. It selects an action $a \in \mathscr{A}$ which leads to s_{t+1} stochastically (i.e., $s_{t+1} \sim f(s_t, a, E)$).
  - The agent is equipped with a first person RGB camera to obtain $I = \rho(E, s)$ where ρ is a rendering function.
  - Assume we have visual observations along the path $I = \{I_1, \ldots, I_J\}$, as well as the current visual observation O. The controller Π predicts actions from $\mathscr{A}$ that successfully convey the agent to the destination.

Robust Path Following

ROBOT INTELLIGENCE LAB

- **Learned Controller Π for Robust Path Following**
  - We first use the path $\mathbf{p}$ and images $\mathbf{I}$ to compute a **path description** $M(\mathbf{p}) = \{(\mathbf{a}(\mathbf{p})_j, \phi(I_j)) : j \in [1 \dots J]\}$
  - This **path description** is used with a learned controller that takes as input the current image to output actions to follow the path under noisy actuation.
  - A soft attention mechanism centered at η_t is used where η is computed from the controller.

Attention location

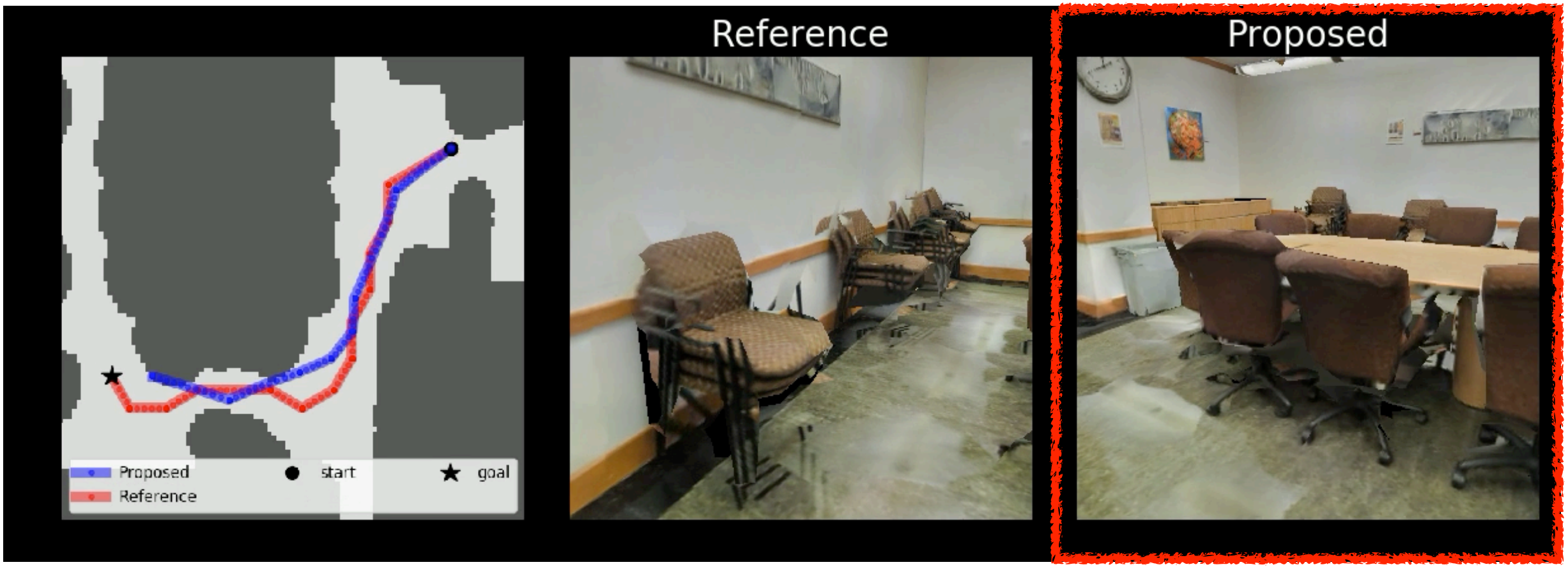
$$\mu_t = \sum_j \psi(M(\mathbf{p})_j) e^{-|\eta_t - j|}$$

Robust Path Following

ROBOT INTELLIGENCE LAB

Robust Path Following

ROBOT INTELLIGENCE LAB

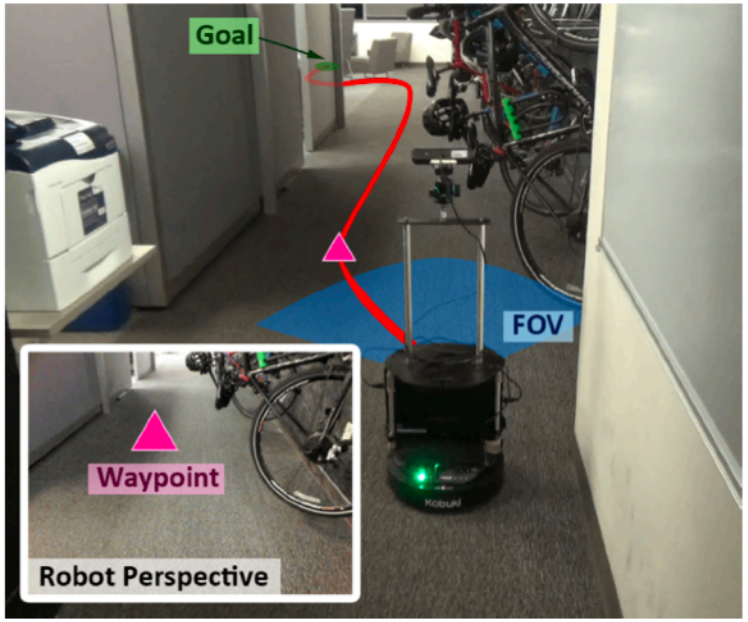


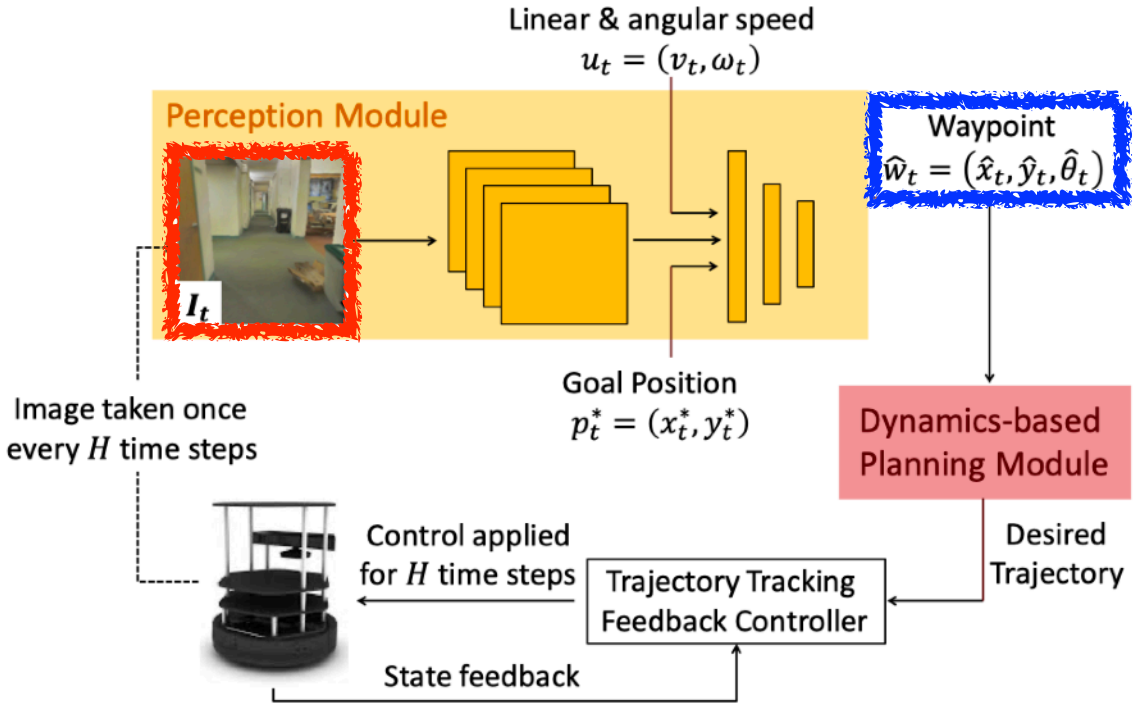
WayPoint Navigation

- "Combining Optimal Control and Learning for Visual Navigation in Novel Environments," 2019
  - It combines **model-based control** with **learning-based perception**.
  - The learning-based perception module produces a **series of waypoints** that guide the robot to the goal.
  - These **waypoints** are used by a model-based planner to generate a smooth and dynamically feasible trajectory that is executed on the physical system using feedback control.
  - It is assumed that odometry is perfect (i.e., the exact vehicle state is available).

WayPoint Navigation

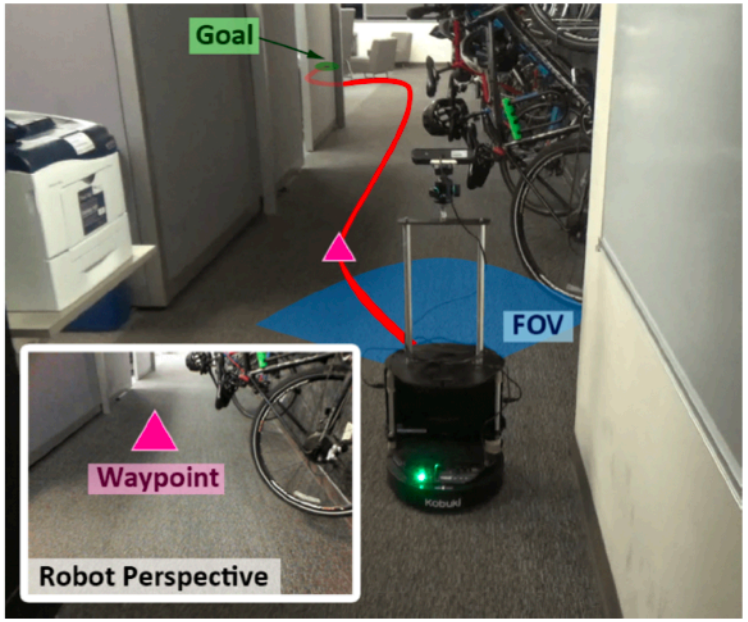
ROBOT INTELLIGENCE LAB

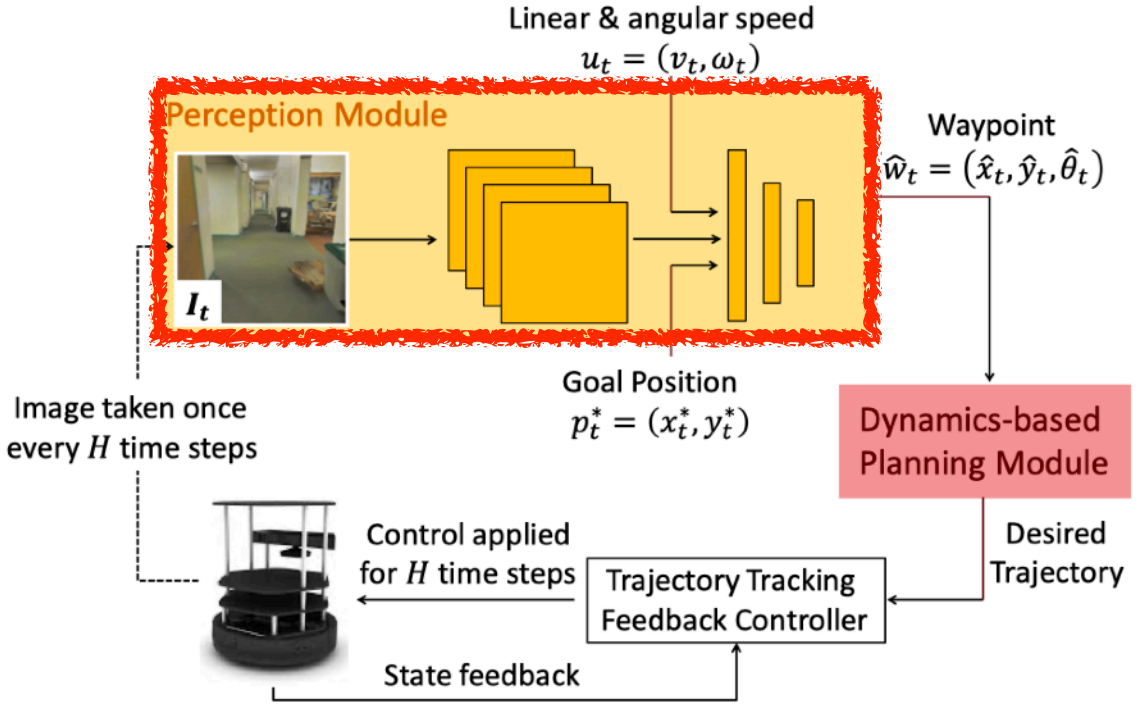




- The perception module takes an RGB image I_t, the relative target position p_t^*, and vehicles's current linear and angular speed u_t and outputs the desired waypoint $(\hat{x}_t, \hat{y}_t, \hat{\theta}_t) = \psi(I_t, u_t, p_t^*)$.

WayPoint Navigation

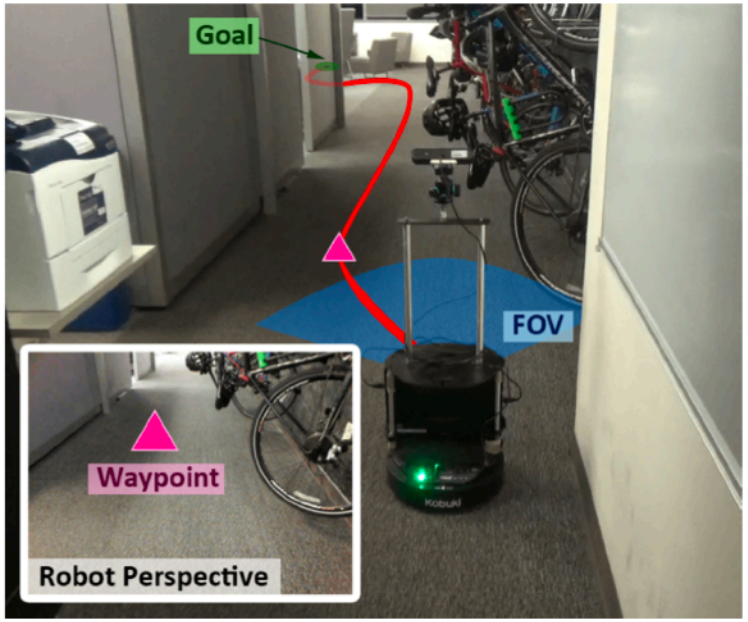


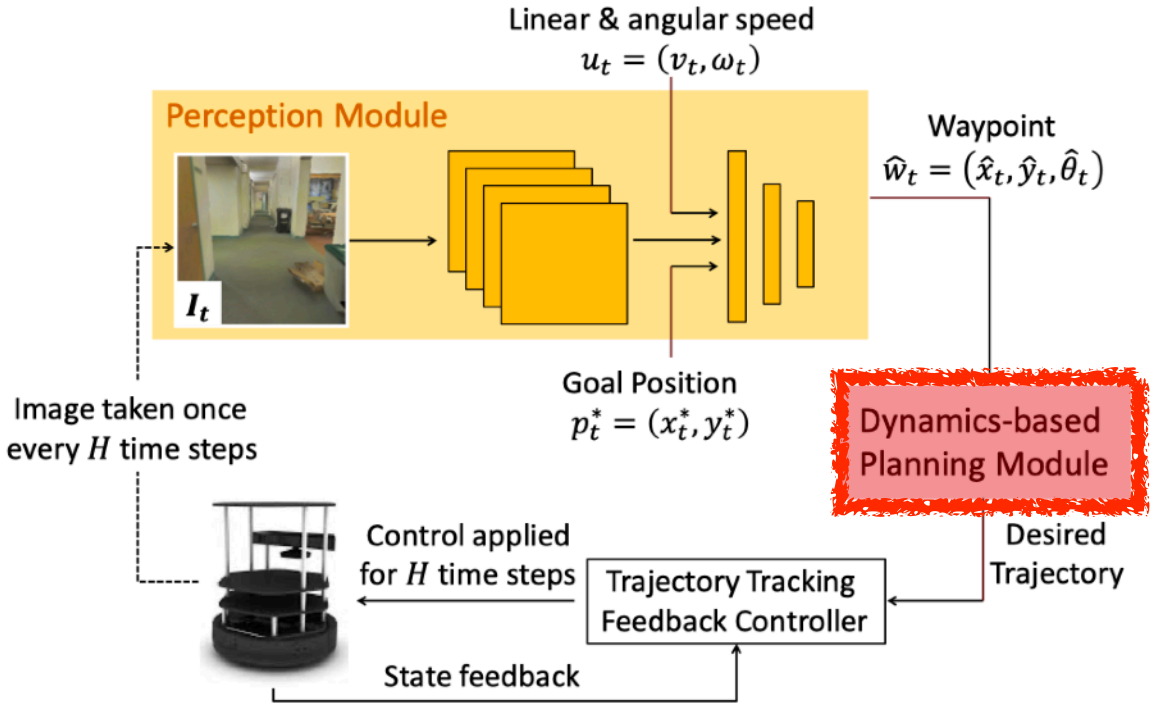


- The training of the **perception module** is done via supervised learning where the underlying map is known. The waypoints for the training phase are generated by solving MPC.

WayPoint Navigation

ROBOT INTELLIGENCE LAB



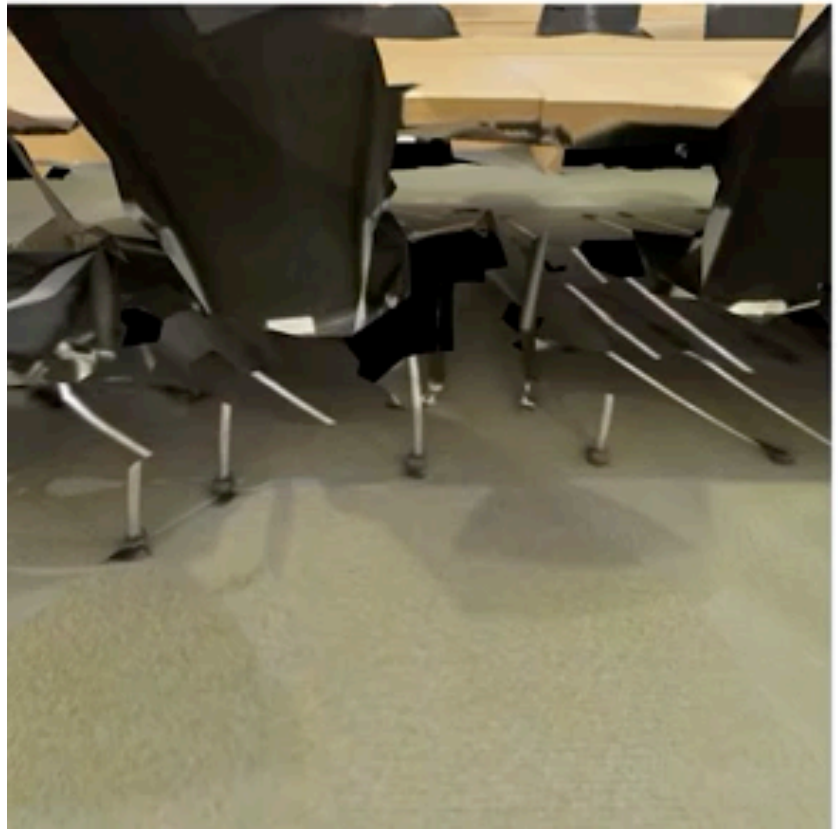


- The **controller** combines spline-based trajectory planner with LQR using a Dubins car system.

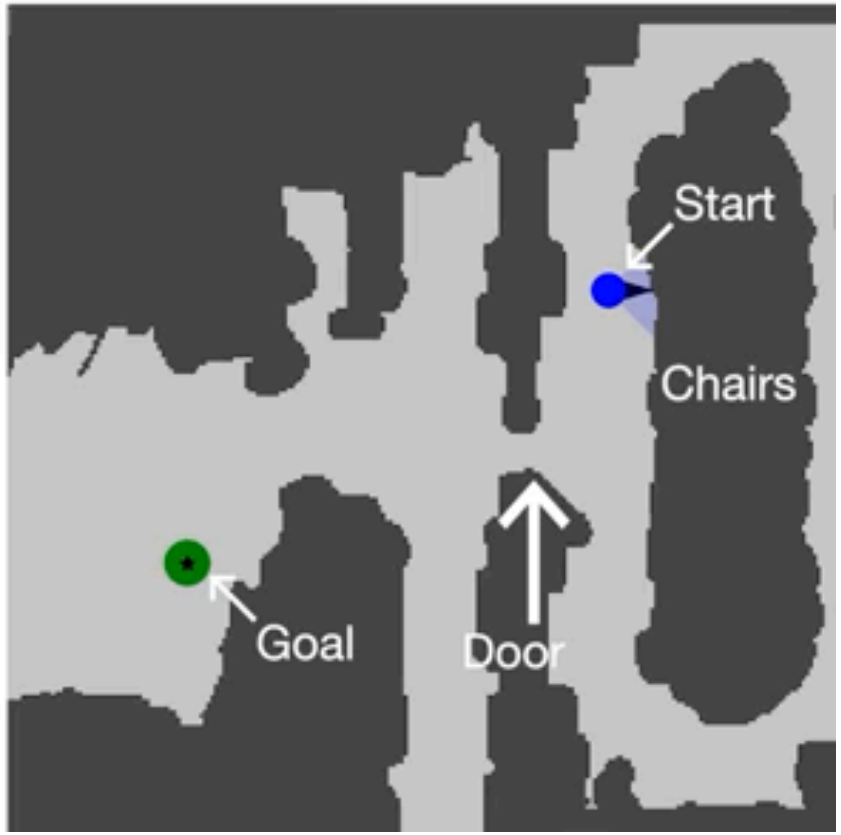
WayPoint Navigation

ROBOT INTELLIGENCE LAB

First Person View



Top View



WayPoint Navigation

ROBOT INTELLIGENCE LAB

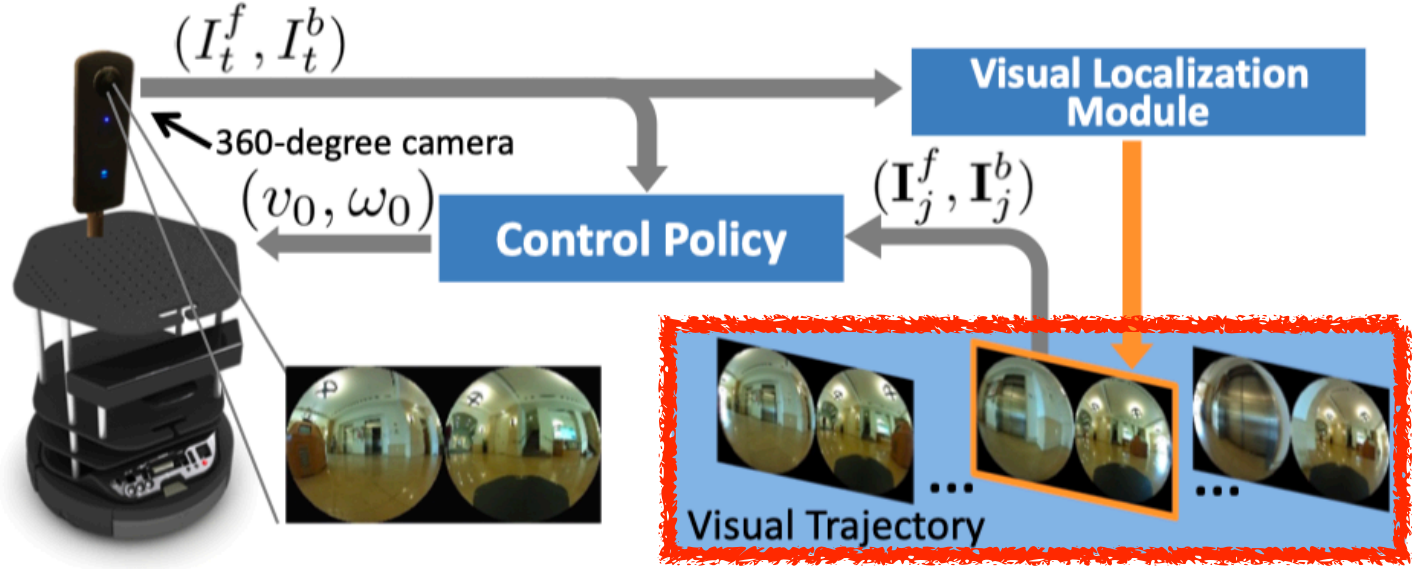
- This method only works in the trained environment with a known map and also assumes exact odometry.
- The only fun part is the it relies on RGB images on the execution phase (as well as exact locations..)
- Not practical.

Deep Visual MPC

ROBOT INTELLIGENCE LAB

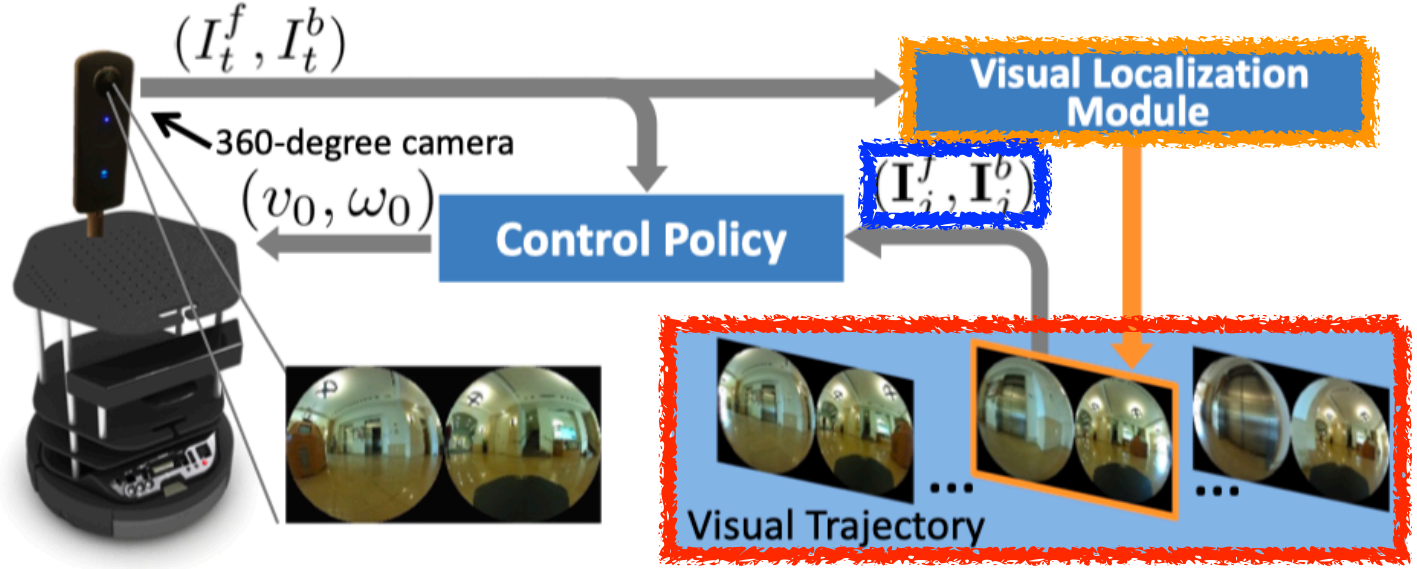
- "Deep Visual MPC-Policy Learning for Navigation," 2019
  - It present a navigation system based solely on visual information provided by 360° camera.
  - The main contribution is **PoliNet** that generates the velocity commands for a mobile robot to follow **a visual path** (a video sequence) while keeping the agent safe from collisions.
  - **PoliNet** is trained to minimize a model predictive control objective by back-propagating through a differentiable visual dynamics model, **VUNet360** and a differentiable traversability estimation network, **GONet**.
  - It used **Turtlebot 2** with a **Rocoh THETA S 360** camera.

Deep Visual MPC



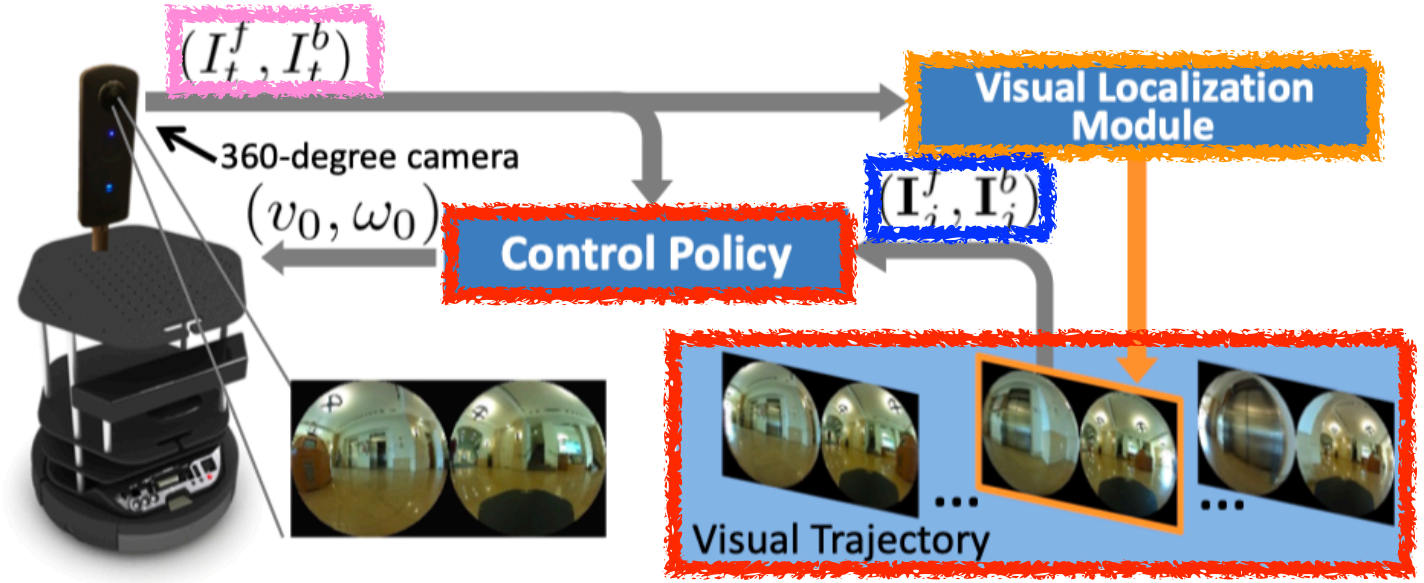
- Given a **visual trajectory** (consecutive images, i.e., **subgoals**), obtained from tele-operations.

Deep Visual MPC



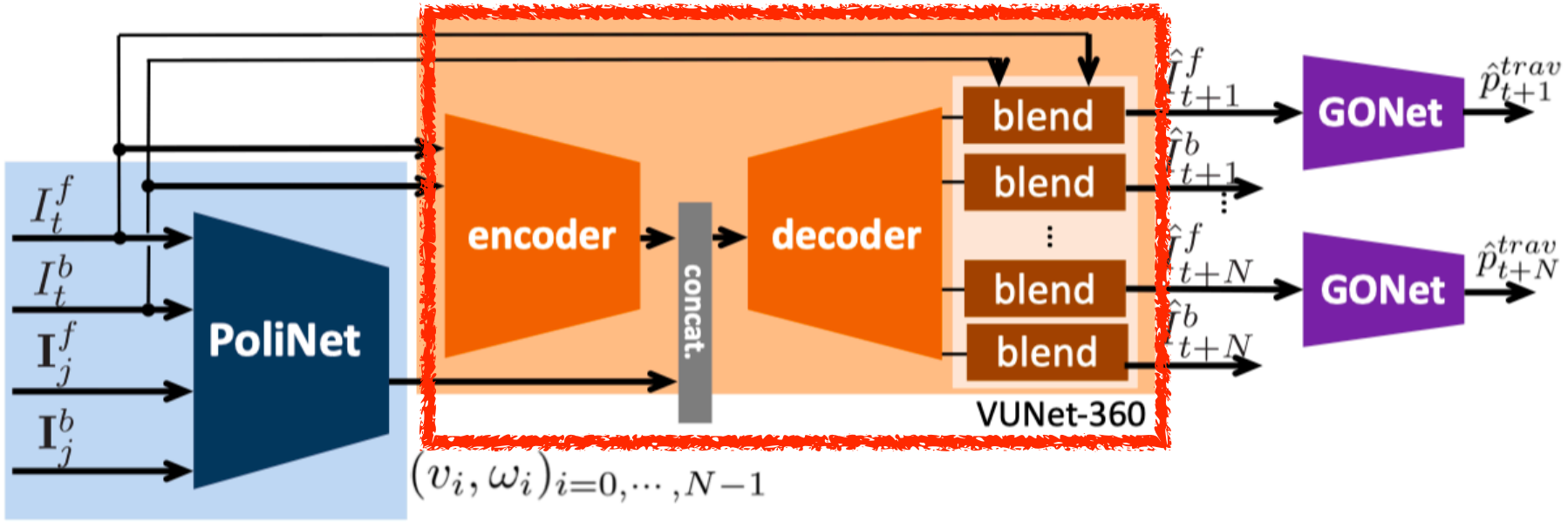
- Given a visual trajectory, the localization module selects a subgoal (or visual waypoint) from the trajectory.

Deep Visual MPC



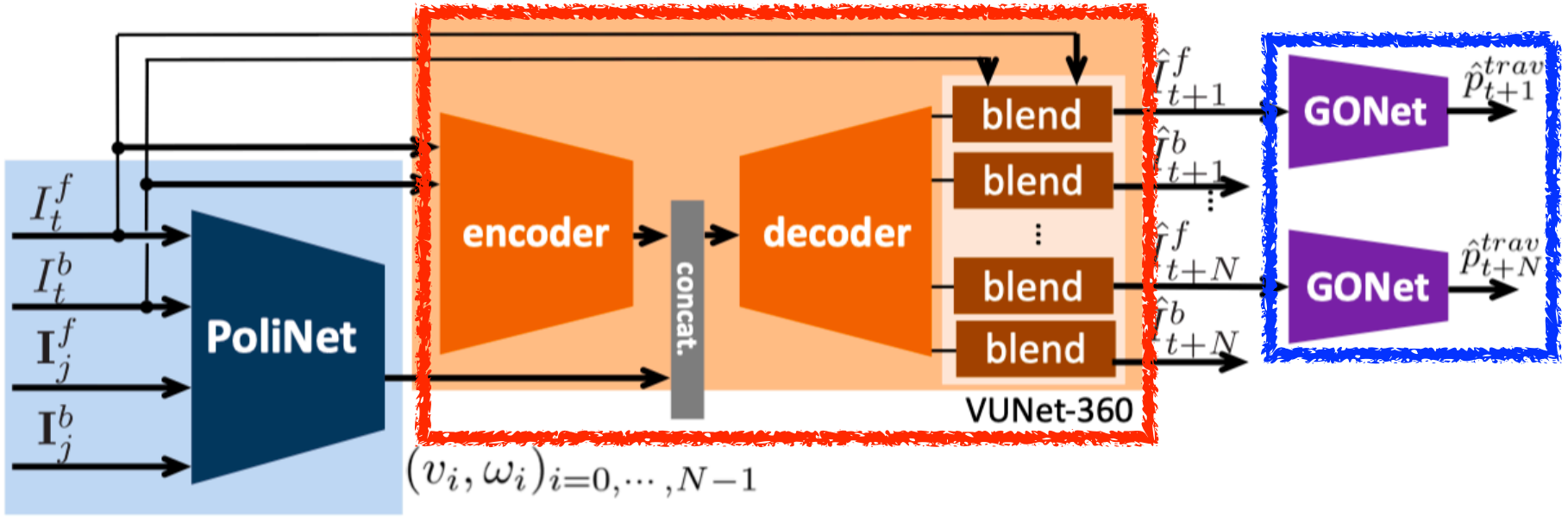
- Given a visual trajectory, the localization module selects a subgoal (or visual waypoint) from the trajectory.
- Then, the control policy outputs the robot command from the subgoal and current observation.

Deep Visual MPC



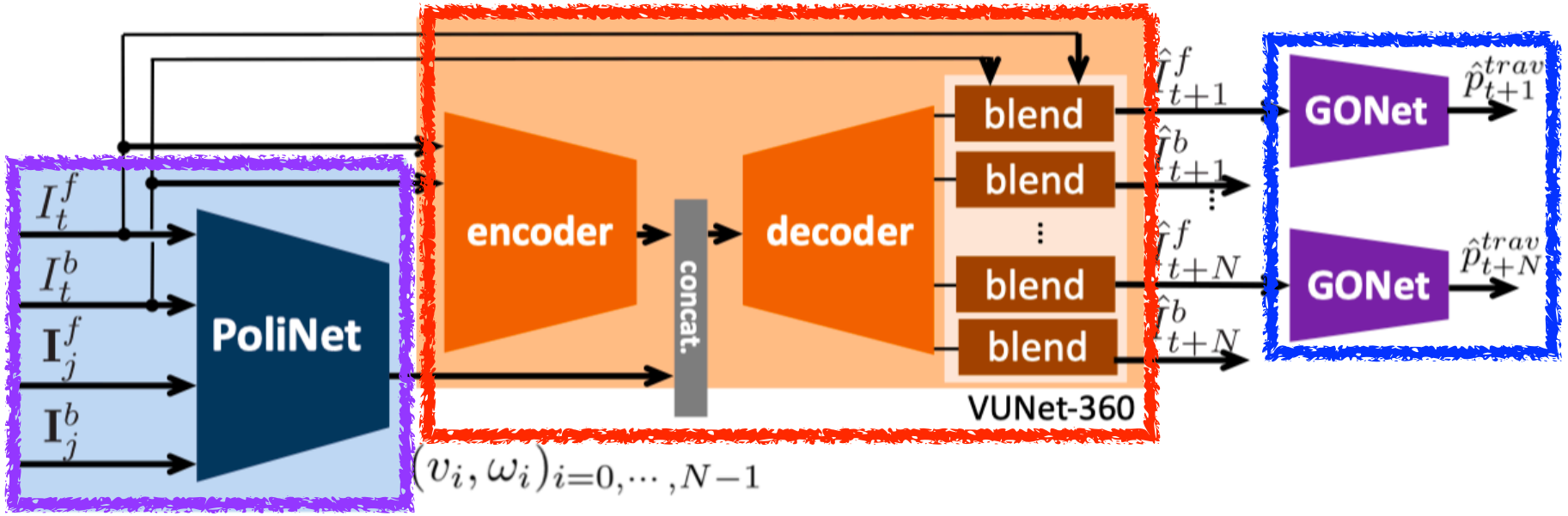
- On the **training** phase, VUNet-360 predicts future images given current image and action

Deep Visual MPC



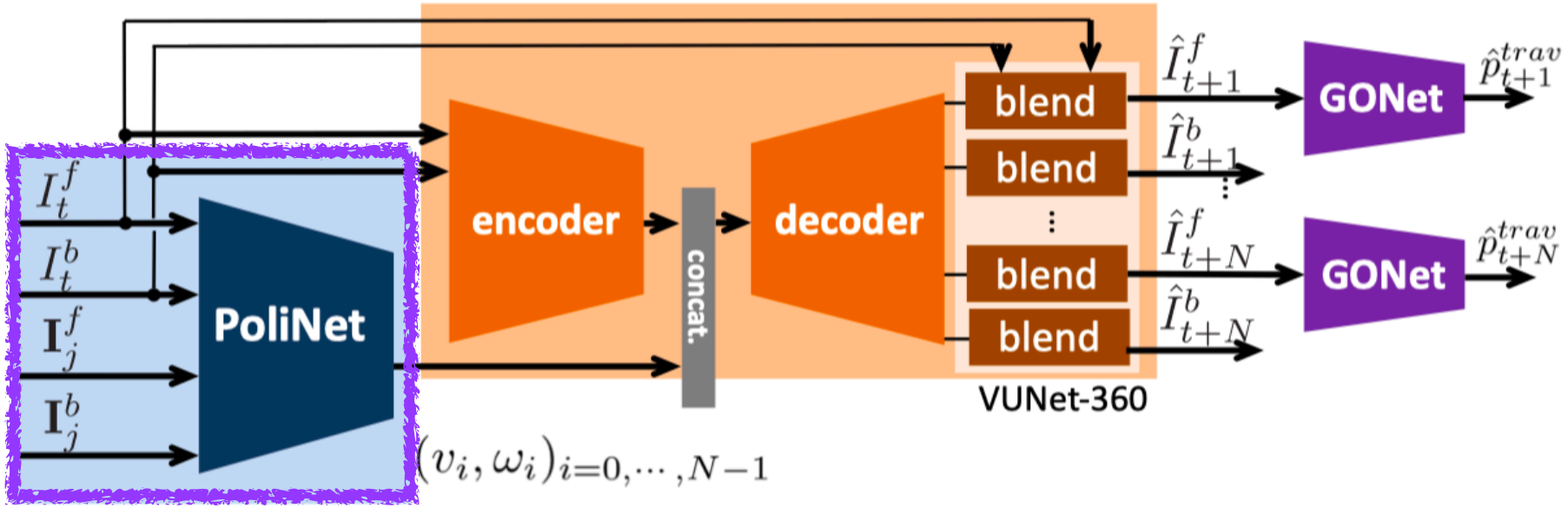
- On the **training** phase, VUNet-360 predicts future images given current image and action, where the generated images are then fed into GONet to predict the **traversability** of each image.

Deep Visual MPC



- The **PoliNet** is trained to maximize the traversability as well as minimize the visual difference between the current robot's image and subgoal images.

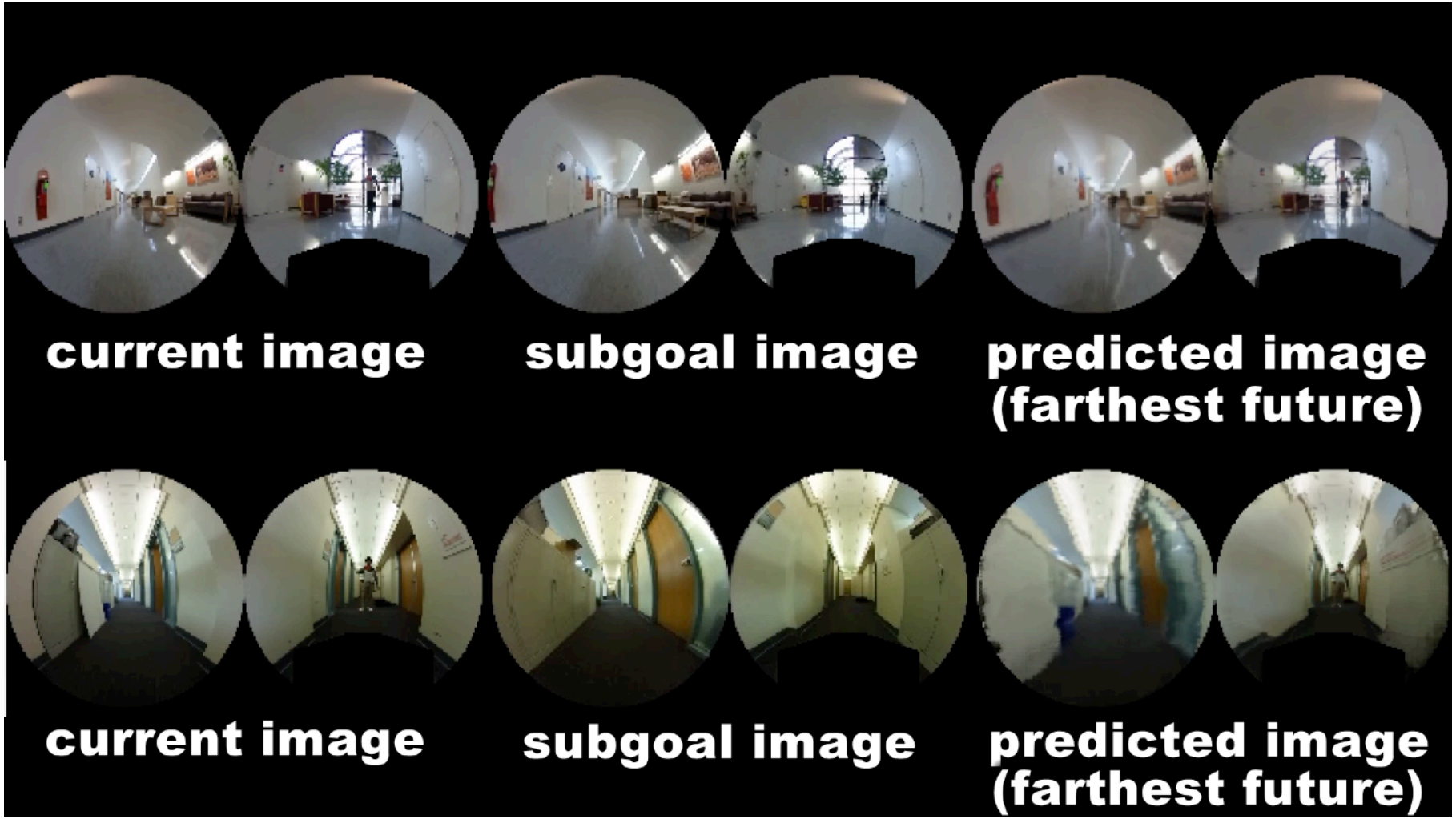
Deep Visual MPC



- On the **execution** phase, only the PoliNet is used.

Deep Visual MPC

ROBOT INTELLIGENCE LAB



Thank You