REINFORCEMENT LEARNING FOR ROBOTICS

Target-driven Visual Navigation in Indoor Scenes Using Deep Reinforcement Learning [Zhu et al. 2017]

A. James E. Cagalawan

james.cagalawan@gmail.com

University of Waterloo

June 27, 2018

(Note: Videos in the slide deck used in presentation have been replaced with YouTube links).

Yuke Zhu[1]
Roozbeh Mottaghi[2]
Eric Kolve[2]
Joseph J. Lim[1]
Abhinav Gupta[2,3]
Li Fei-Fei[1]
Ali Farhadi[2,4]

[1]Stanford University
[2]Allen Institute for AI
[3]Carnegie Mellon University
[4]University of Washington

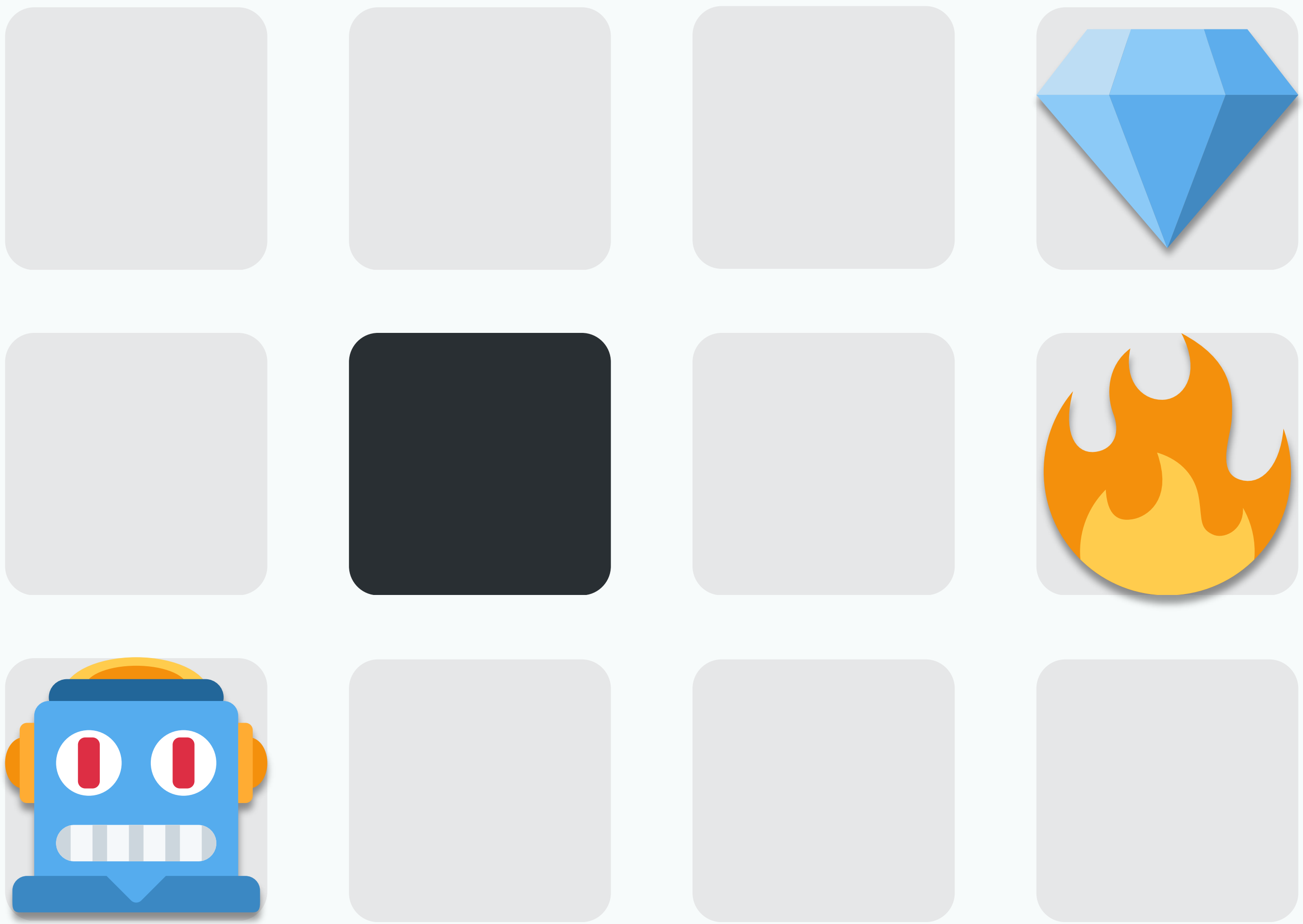
Free Space:

Occupied Space:

Agent:

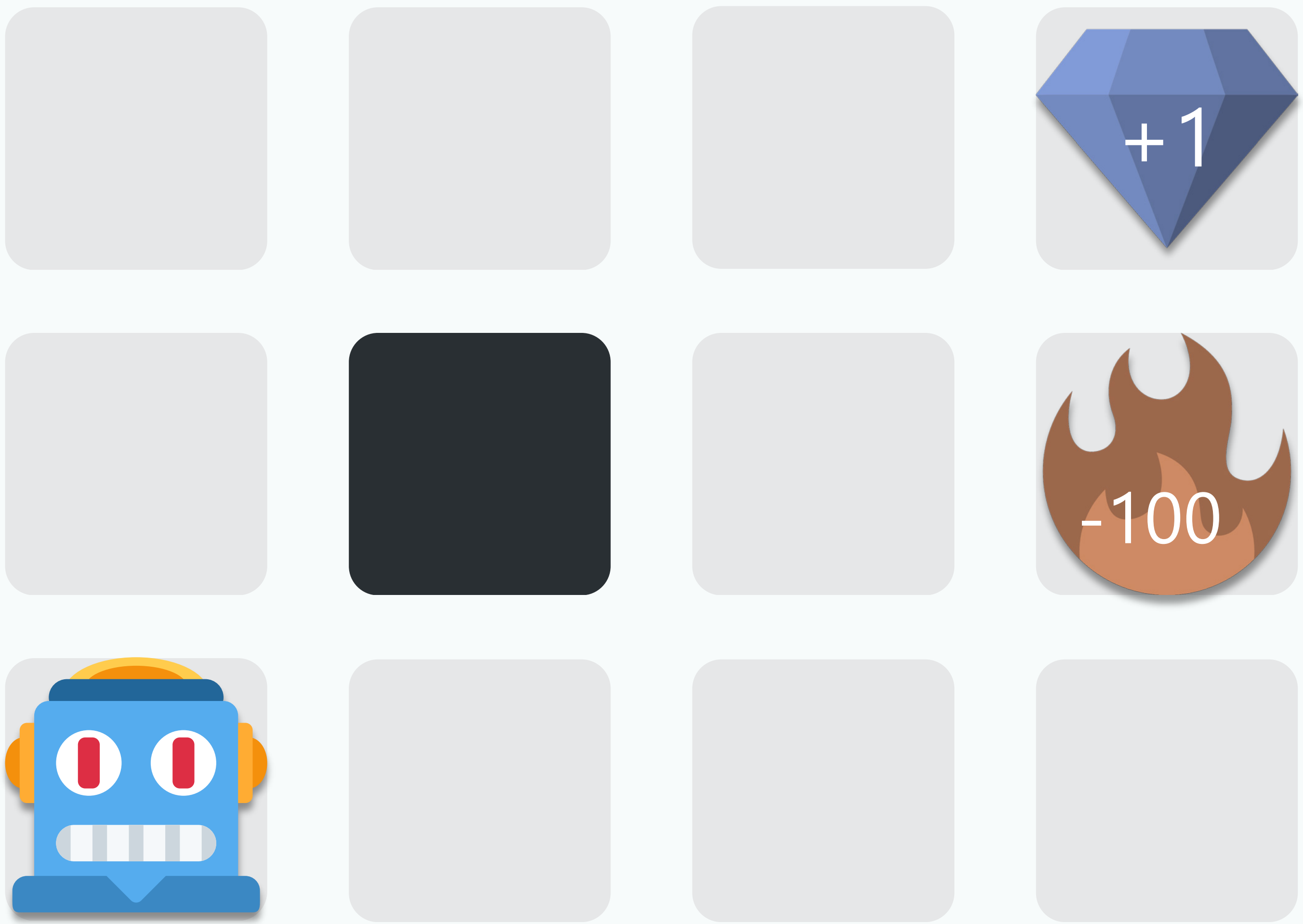
Goal:

Danger:



Free Space:
Occupied Space:
Agent:
Goal:
Danger:

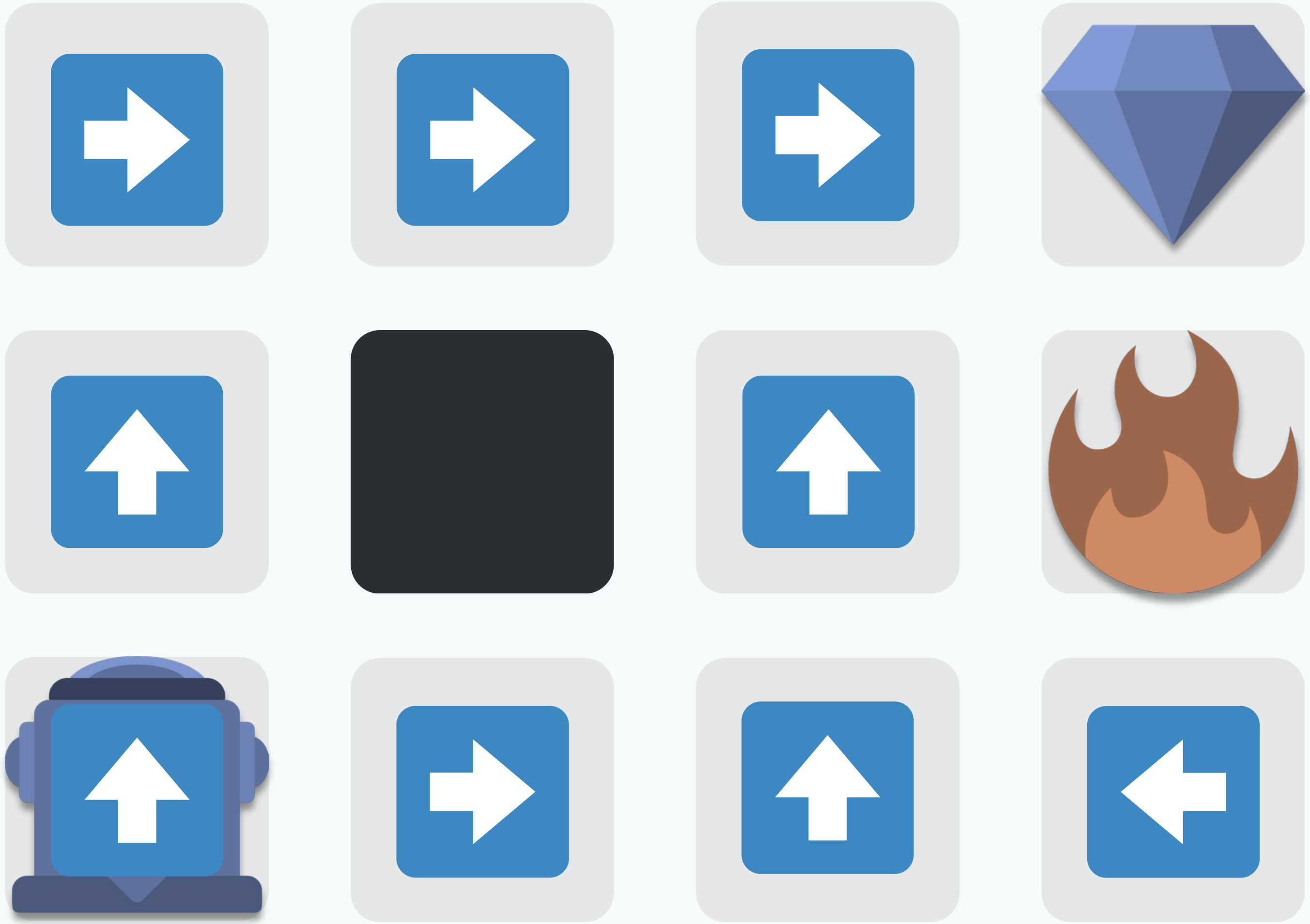
Can assign numerical rewards to the goal and the dangerous grid cells.



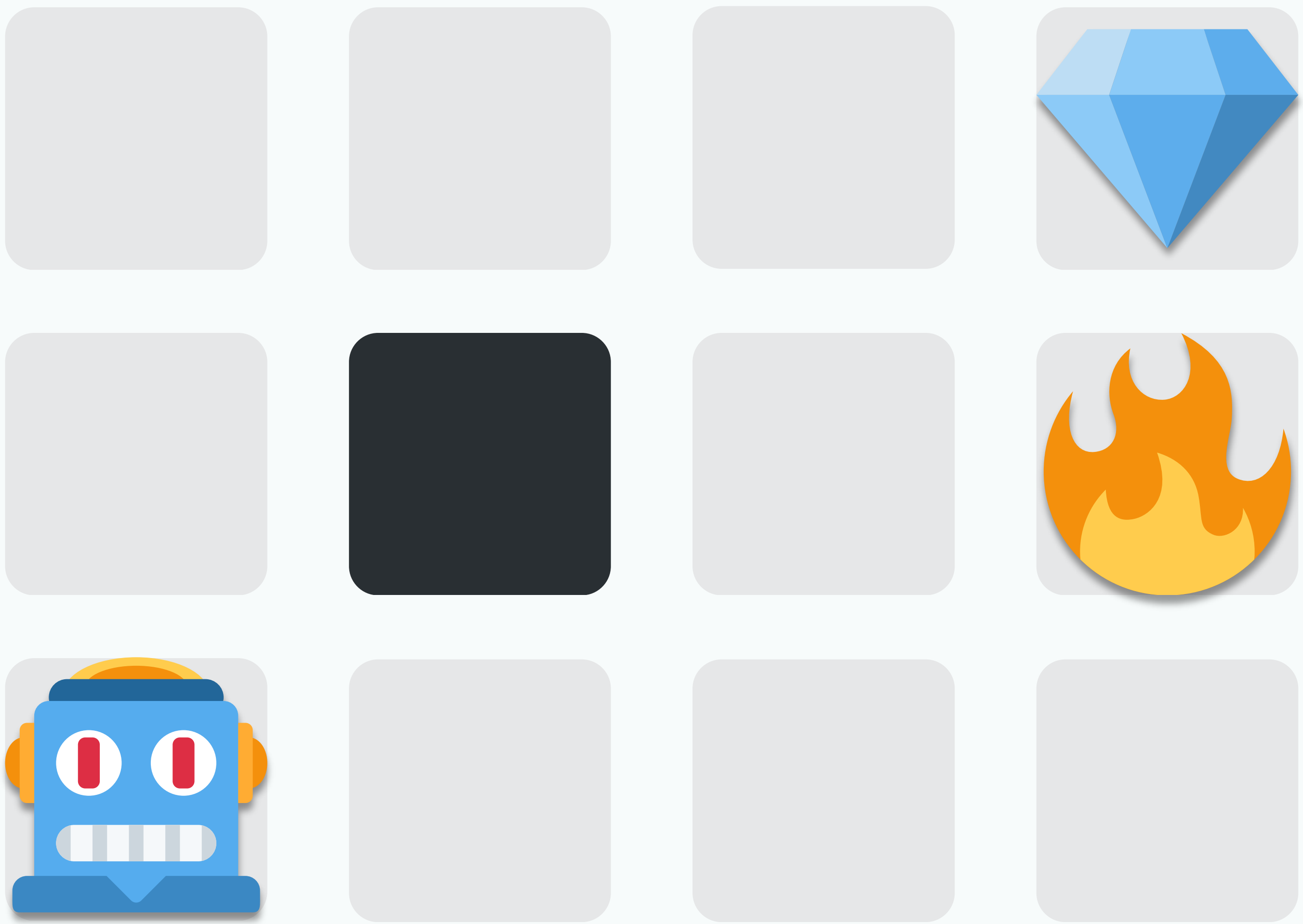
Free Space:
Occupied Space:
Agent:
Goal:
Danger:

Can assign numerical rewards to the goal and the dangerous grid cells.

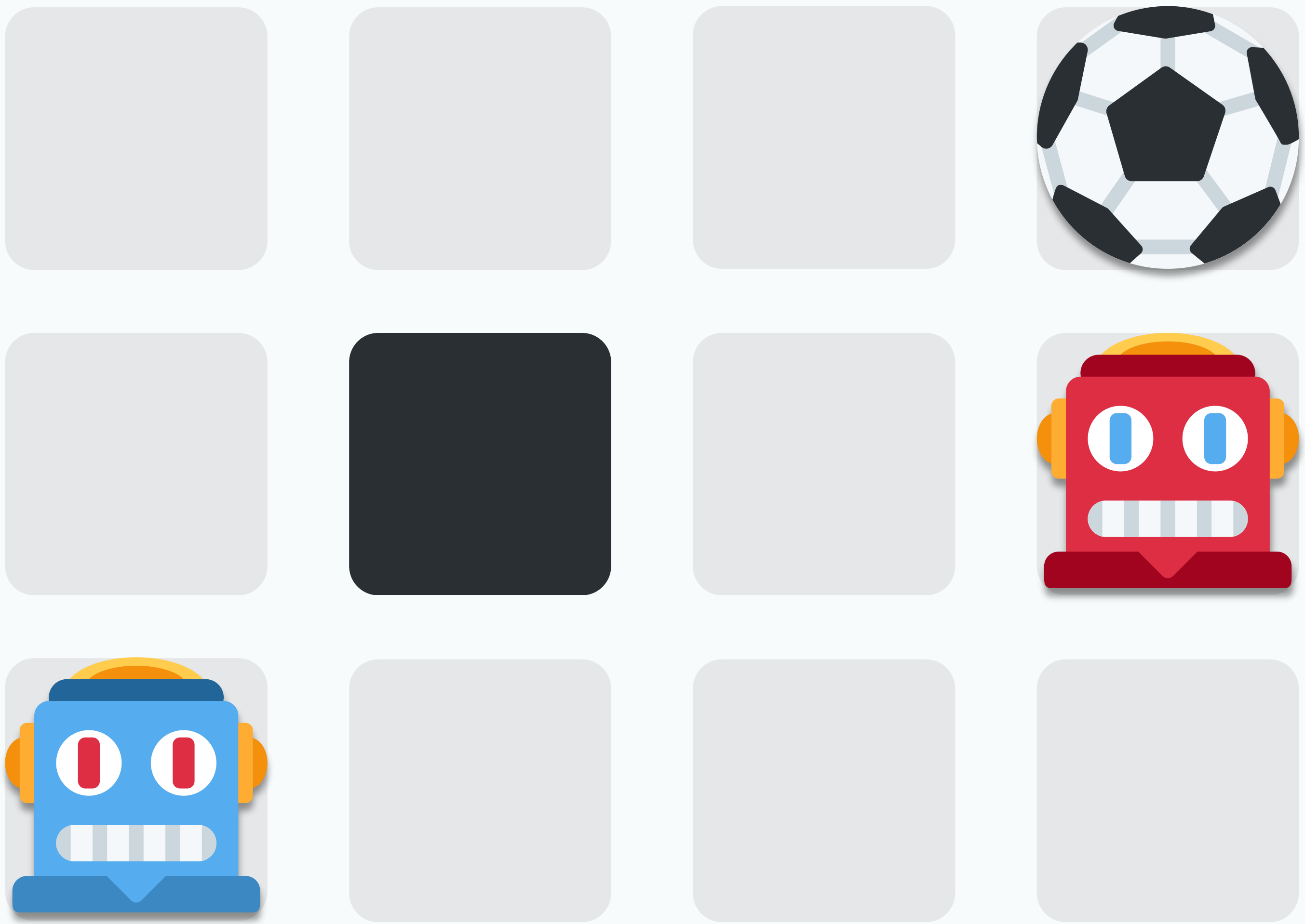
Then learn a policy.



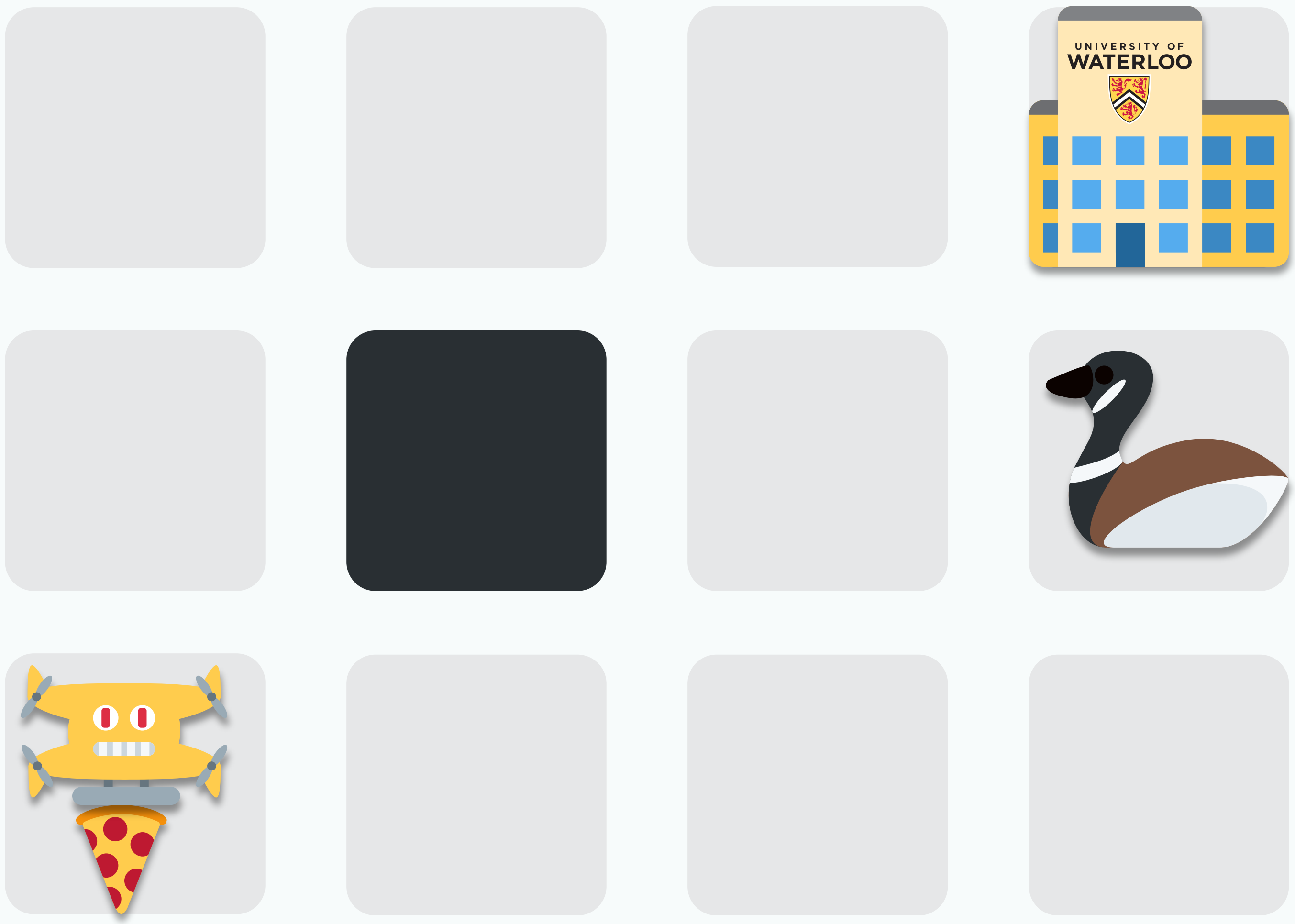
- Robotic visual navigation using robots has many applications.
- Treasure hunting.



- Robotic visual navigation using robots has many applications.
- Treasure hunting.
- Soccer robots getting to the ball first.

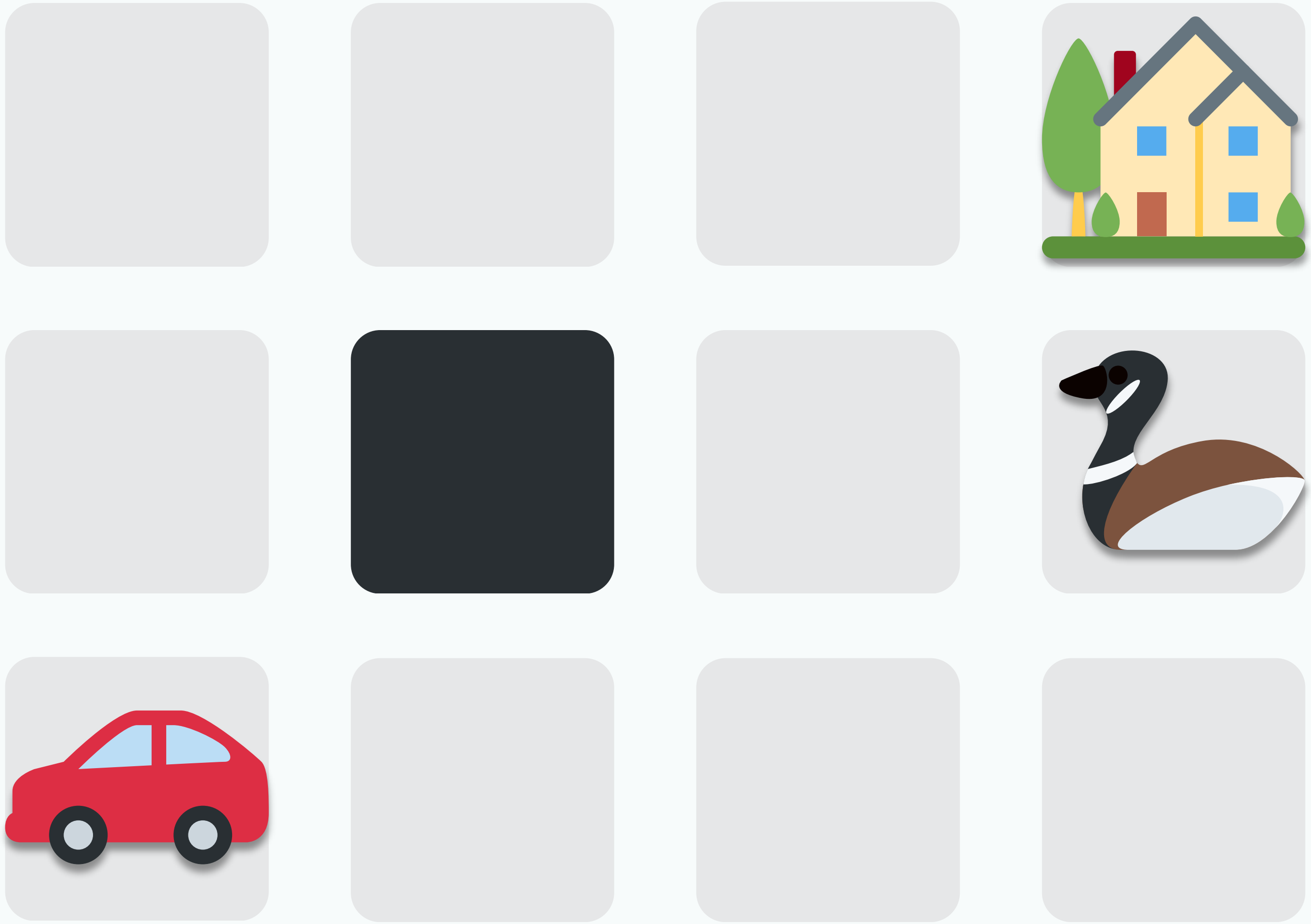


- Robotic visual navigation using robots has many applications.
- Treasure hunting.
- Soccer robots getting to the ball first.
- Drones delivering pizzas to your lecture hall.

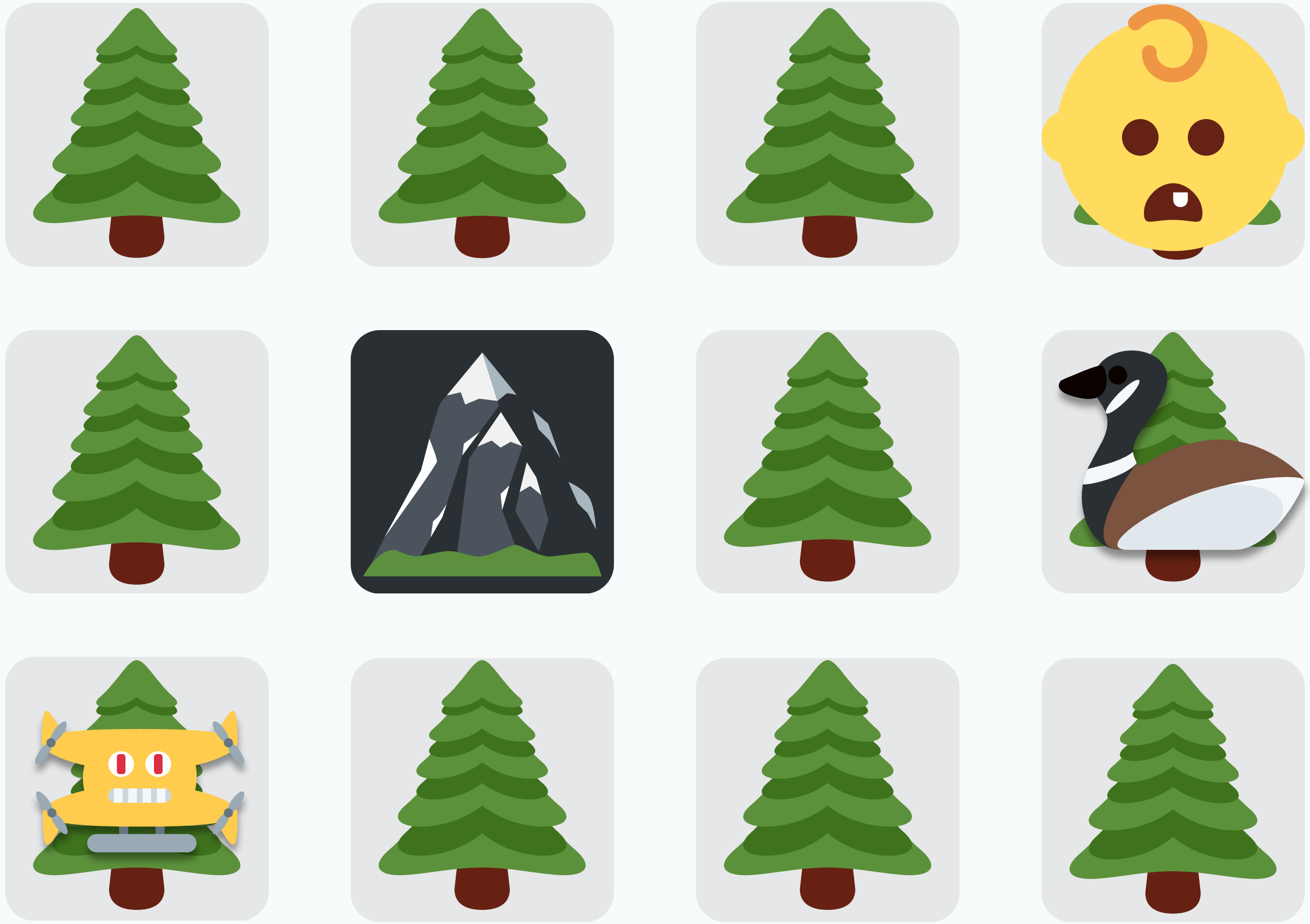


MOTIVATION – **Visual Navigation Applications: Self-Driving Cars**

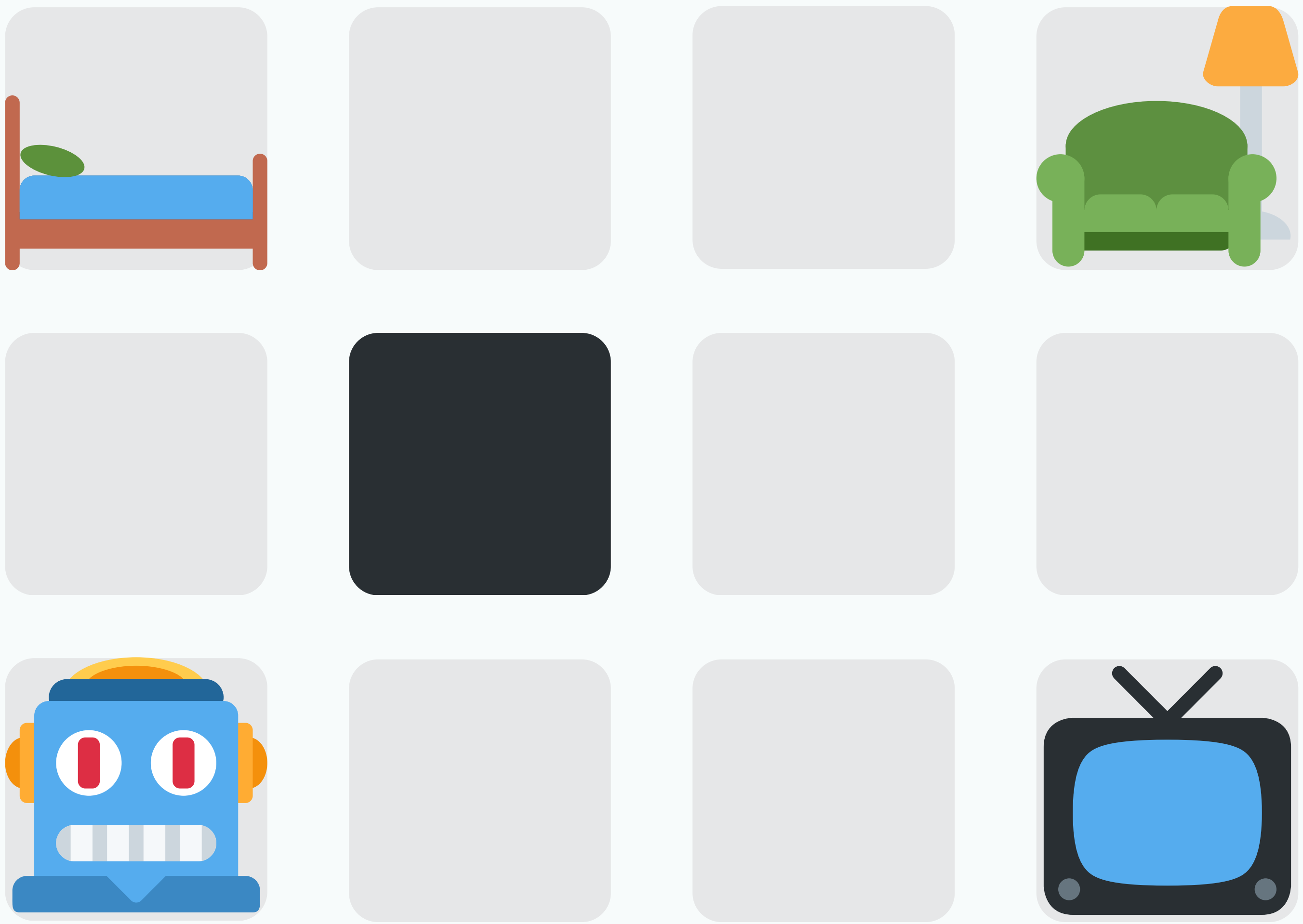
- Robotic visual navigation using robots has many applications.
- Treasure hunting.
- Soccer robots getting to the ball first.
- Drones delivering pizzas to your lecture hall.
- Autonomous cars driving people to their homes.



- Robotic visual navigation using robots has many applications.
- Treasure hunting.
- Soccer robots getting to the ball first.
- Drones delivering pizzas to your lecture hall.
- Autonomous cars driving people to their homes.
- Search and rescue robots finding missing people.

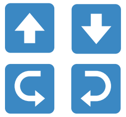


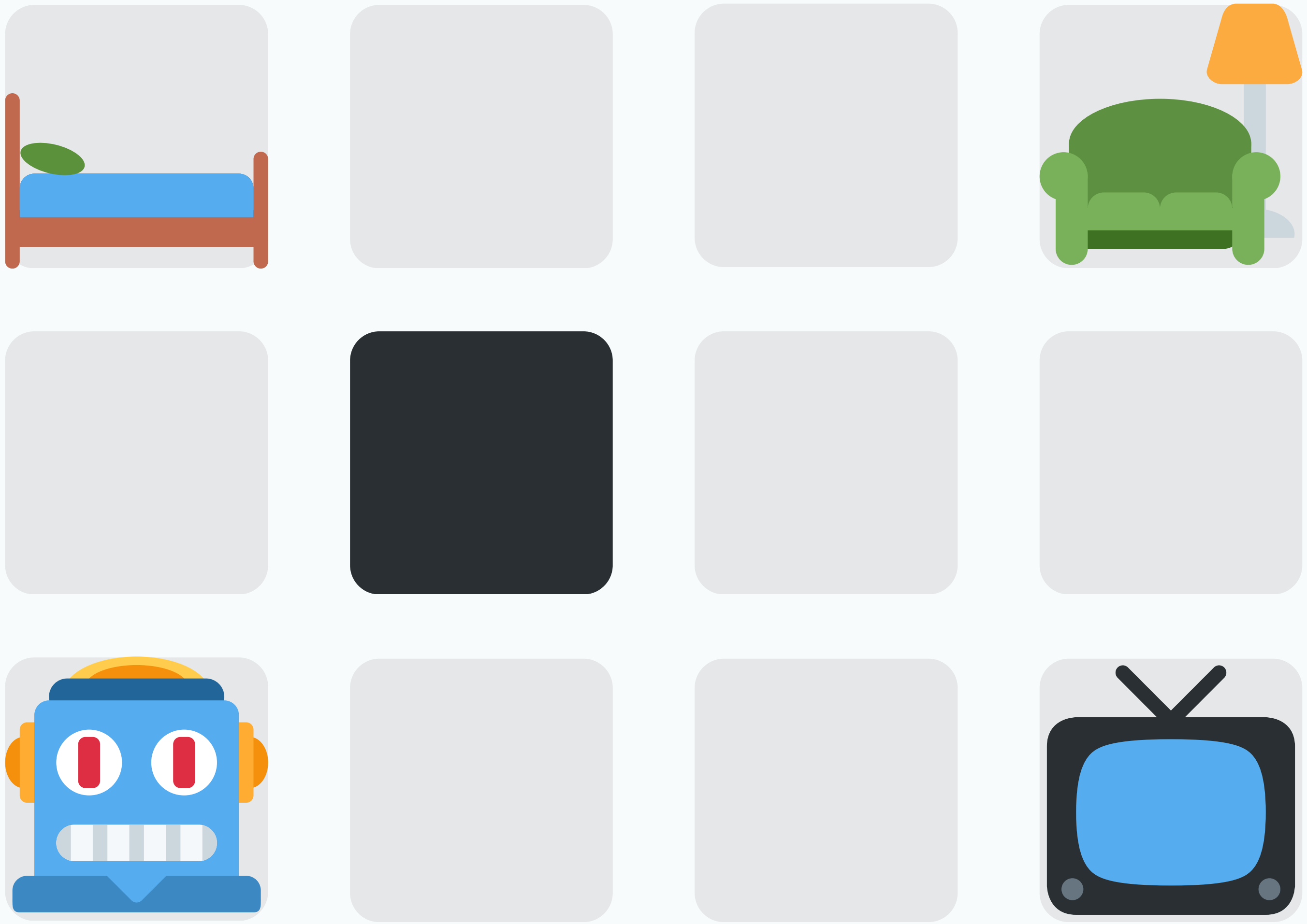
- Robotic visual navigation using robots has many applications.
- Treasure hunting.
- Soccer robots getting to the ball first.
- Drones delivering pizzas to your lecture hall.
- Autonomous cars driving people to their homes.
- Search and rescue robots finding missing people.
- Domestic robots navigating their way around houses to help with chores.



- Multiple locations in an indoor scene that our robot must navigate to.

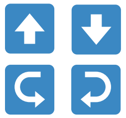
- Actions consist of moving forwards and backwards and turning left and right.



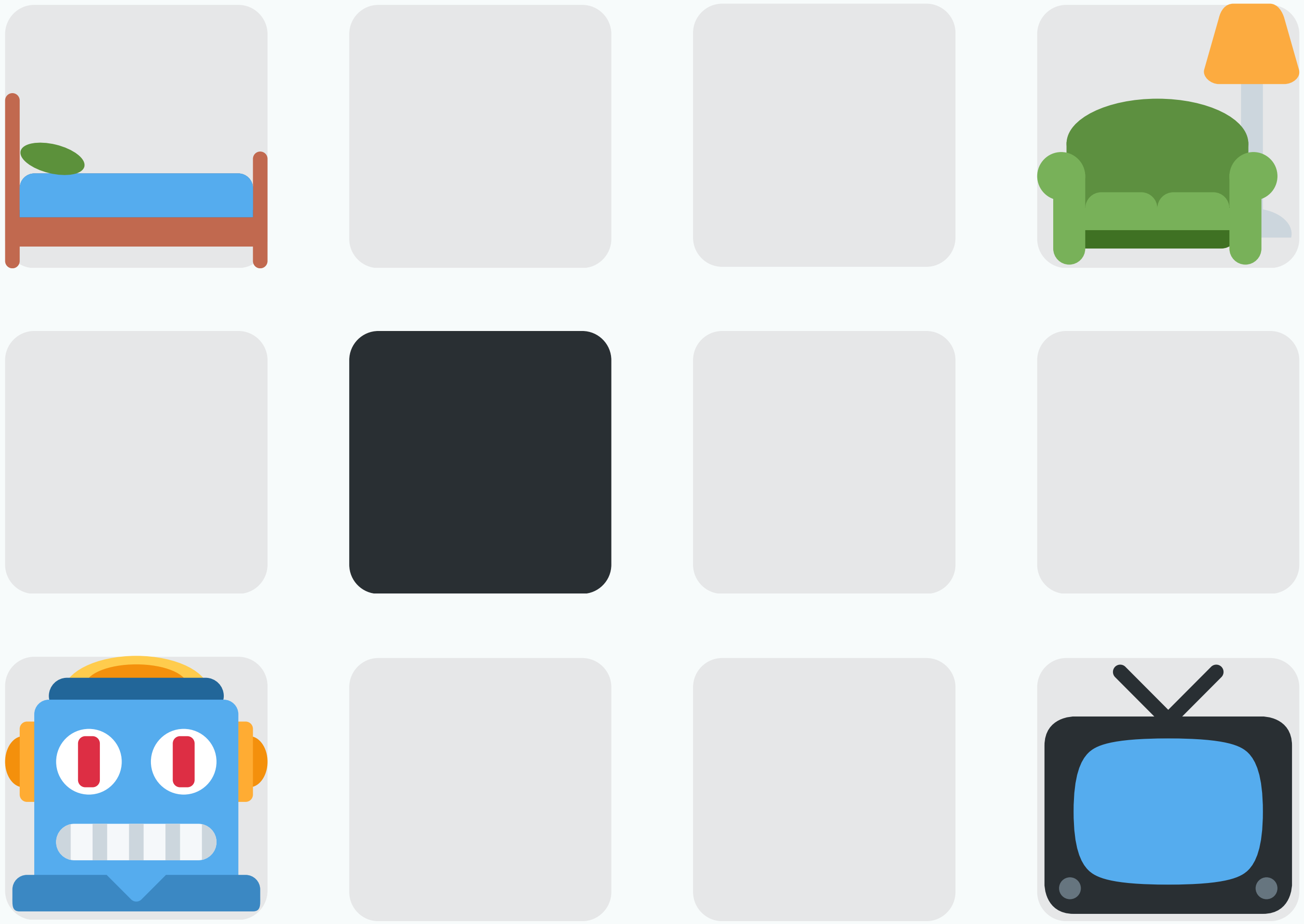


- Multiple locations in an indoor scene that our robot must navigate to.

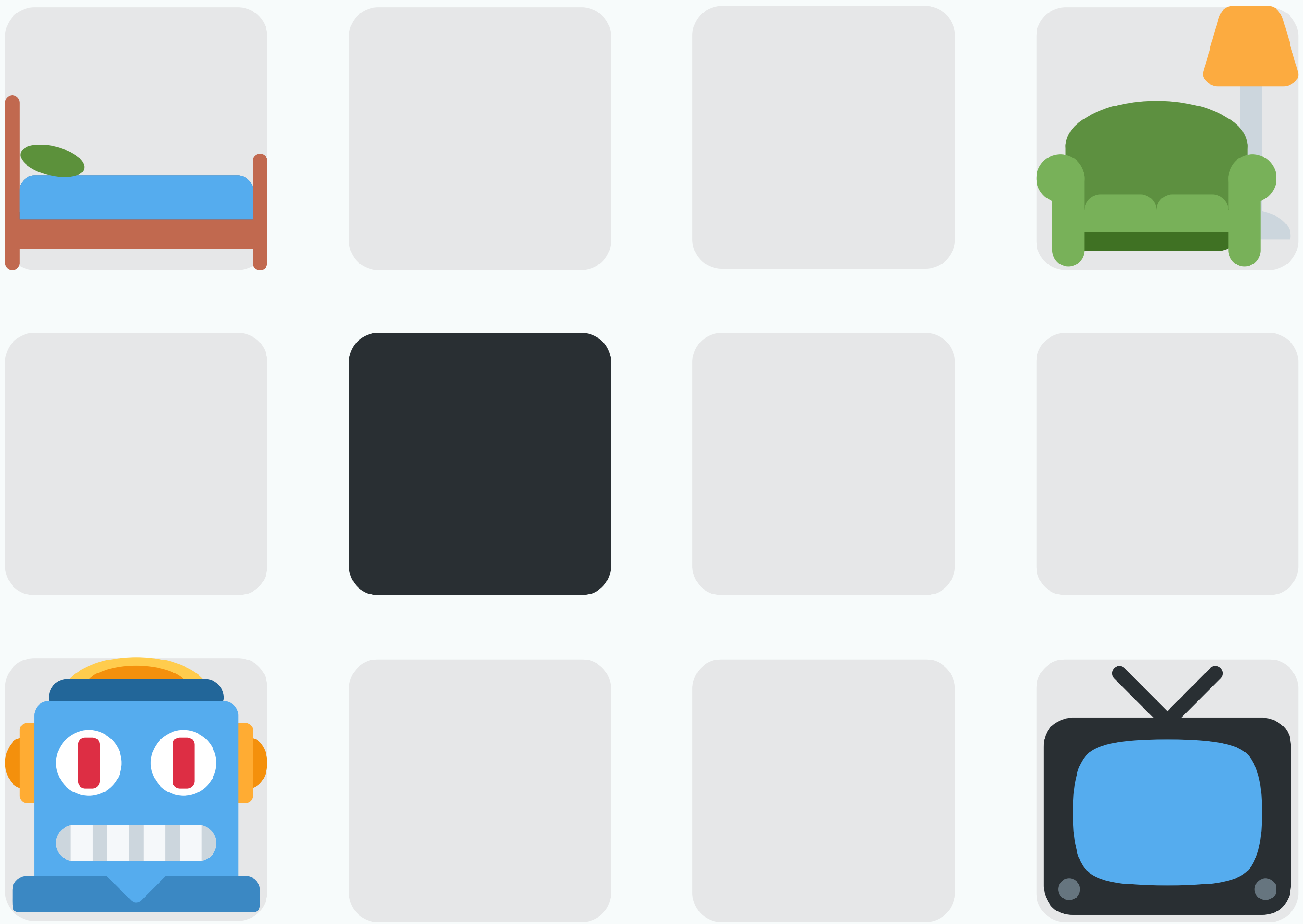
- Actions consist of moving forwards and backwards and turning left and right.



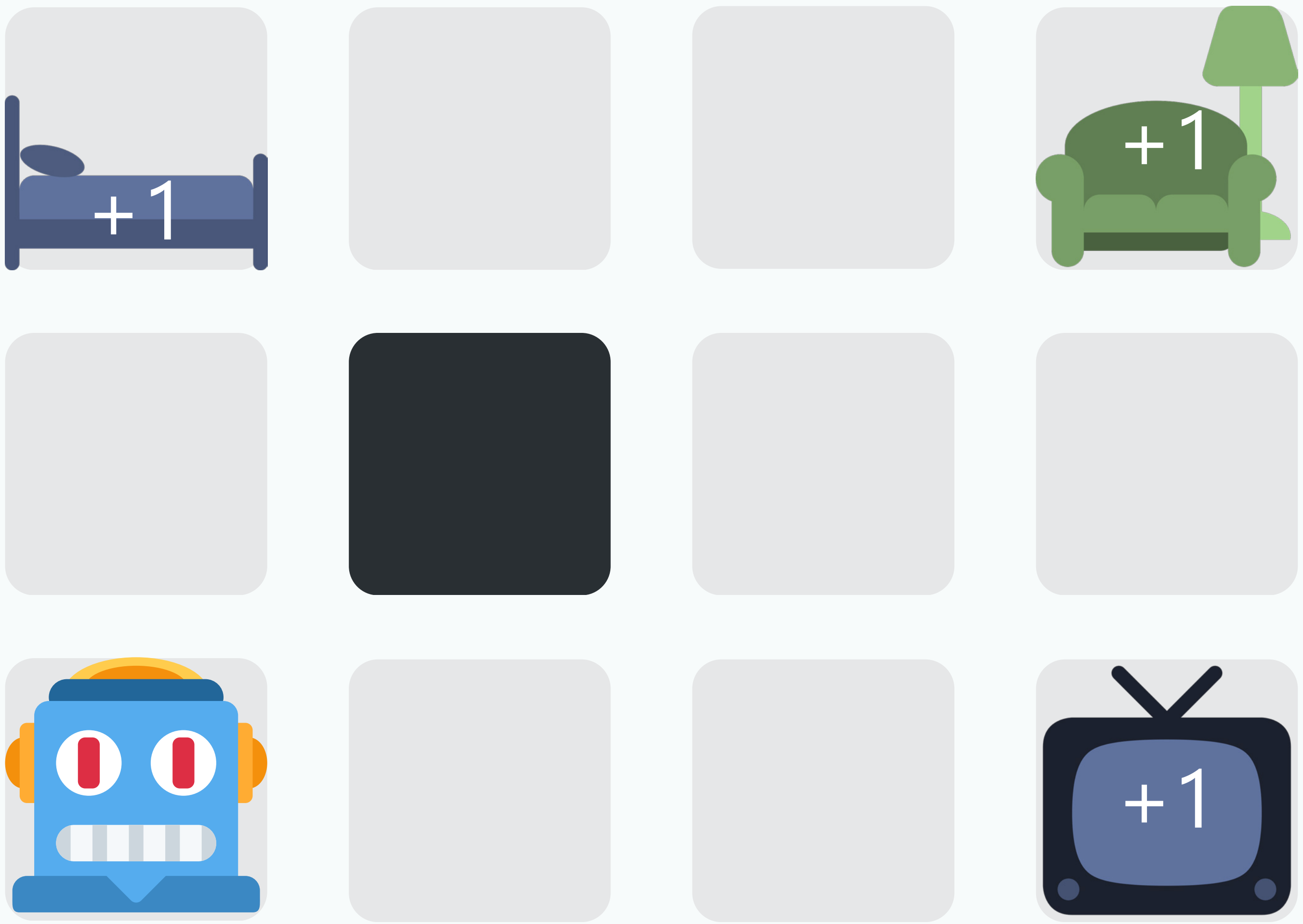
- **Problem 1:** Navigating to multiple targets.
- **Problem 2:** Using high-dimensional visual inputs is challenging and time consuming to train.
- **Problem 3**: Training on a real robot is expensive.



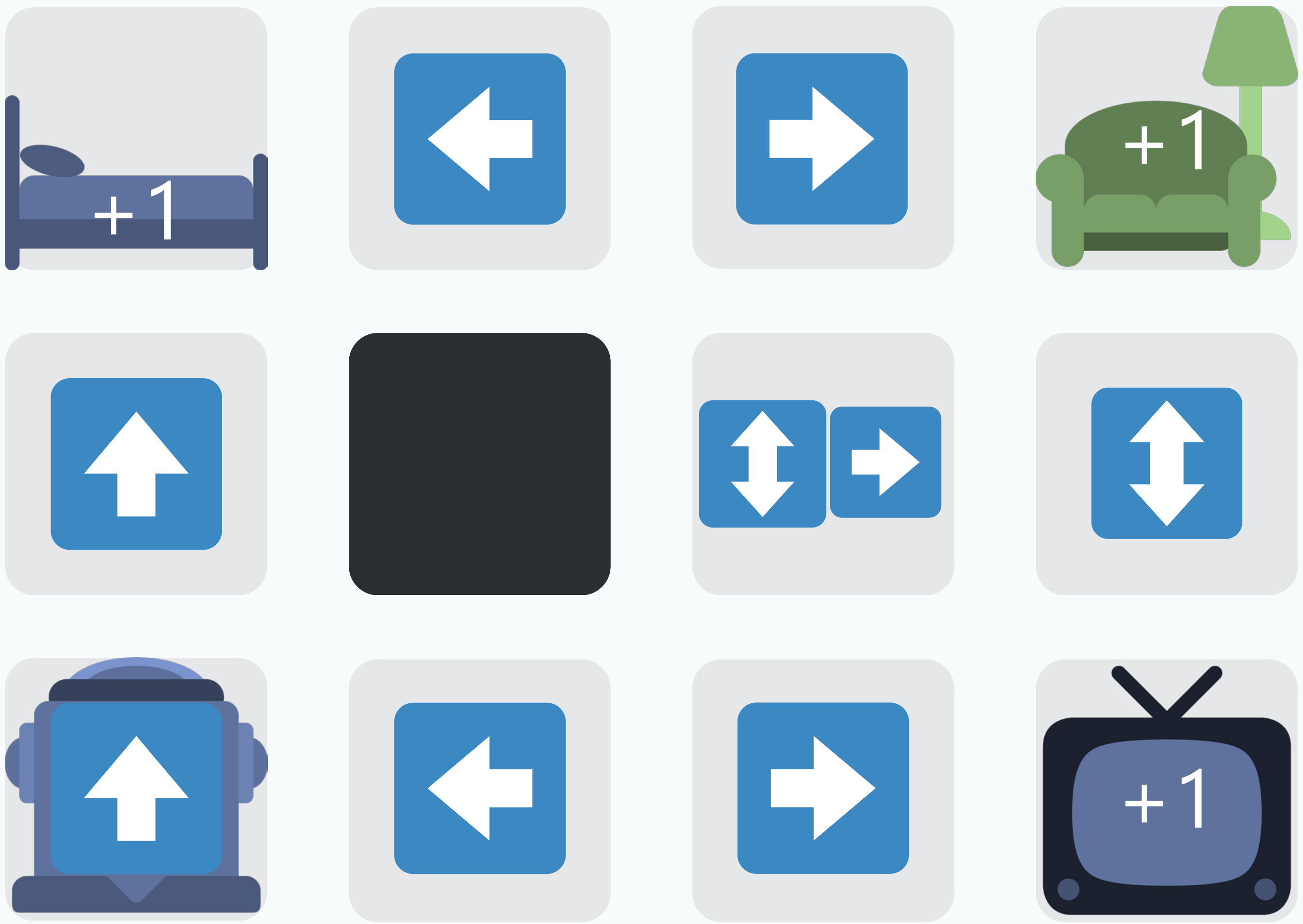
- We can already navigate grid mazes using Q-learning by assigning rewards for finding a target.
- Assigning rewards to multiple locations on the grid does not allow specification of different targets.



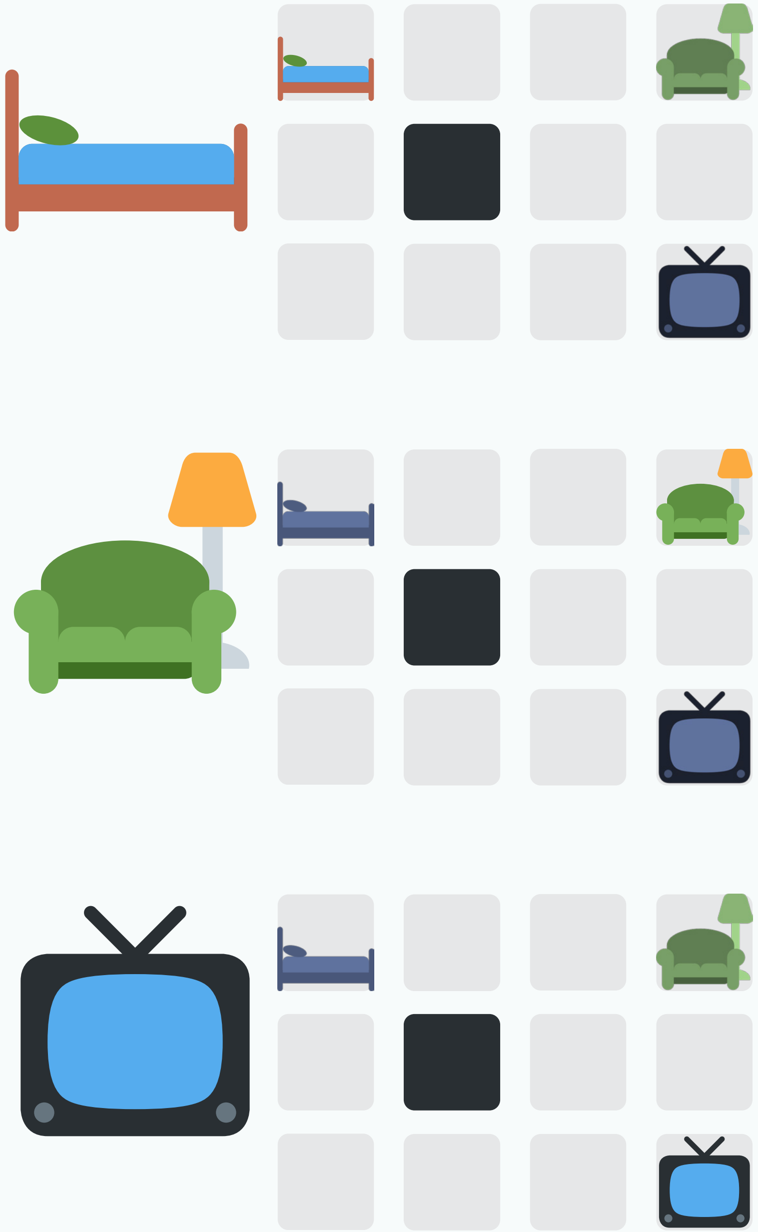
- We can already navigate grid mazes using Q-learning by assigning rewards for finding a target.
- Assigning rewards to multiple locations on the grid does not allow specification of different targets.



- We can already navigate grid mazes using Q-learning by assigning rewards for finding a target.
- Assigning rewards to multiple locations on the grid does not allow specification of different targets.
- Would end up at a target, but not any specific target.

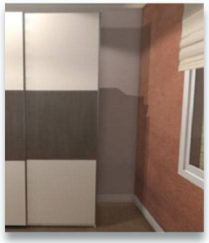


- We can already navigate grid mazes using Q-learning by assigning rewards for finding a target.
- Assigning rewards to multiple locations on the grid does not allow specification of different targets.
- Would end up at a target, but not any specific target.
- Could train multiple policies, but that wouldn't scale with the number of targets.



Sensor Image

Target Image



Step and Turn

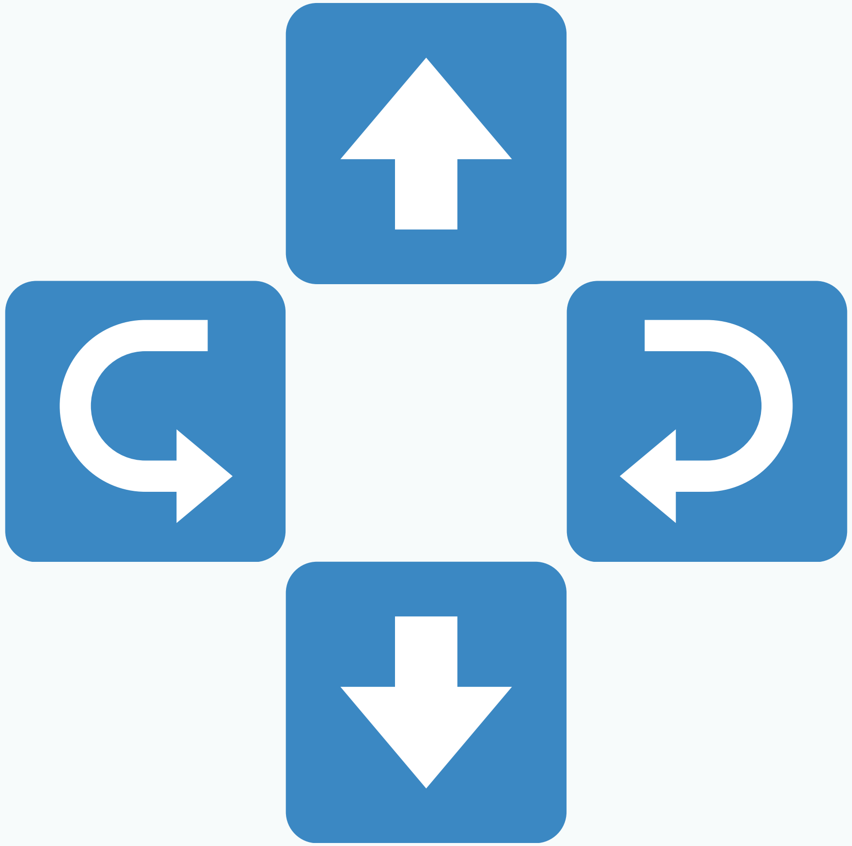


Image → **Action**

PROBLEM – **Visual Navigation Decomposition: Overview**

The visual navigation problem can be broken up into pieces with specialized algorithms solving each piece.

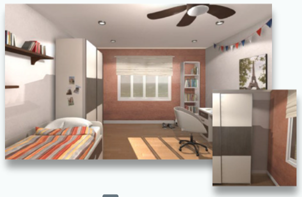
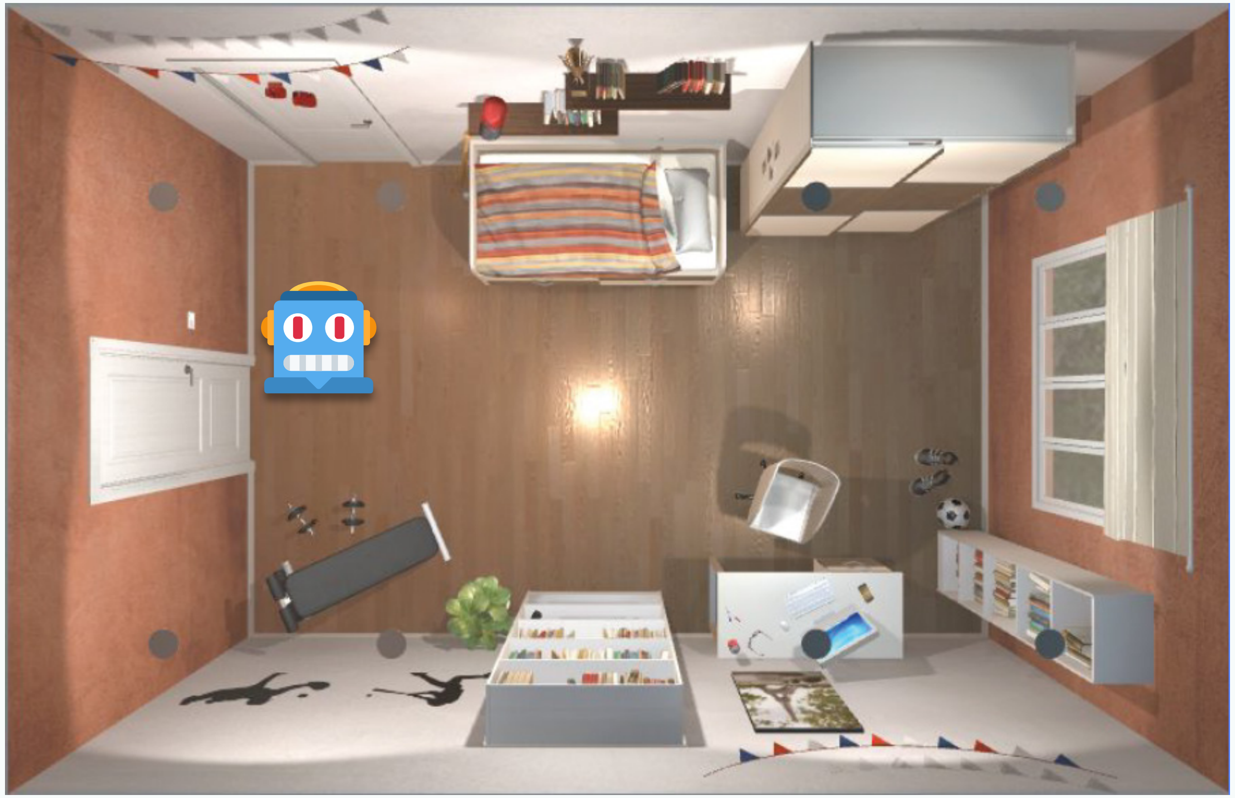


Image → **Perception** → **World Modelling** → **Planning** → **Action**

“Autonomous Mobile Robots” by Roland Siegwart and Illah R. Nourbakhsh (2011)

Image → Perception → World Modelling → Planning → Action

PROBLEM – **Visual Navigation Decomposition: Perception – Localization and Mapping**



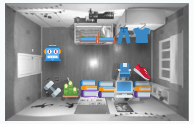
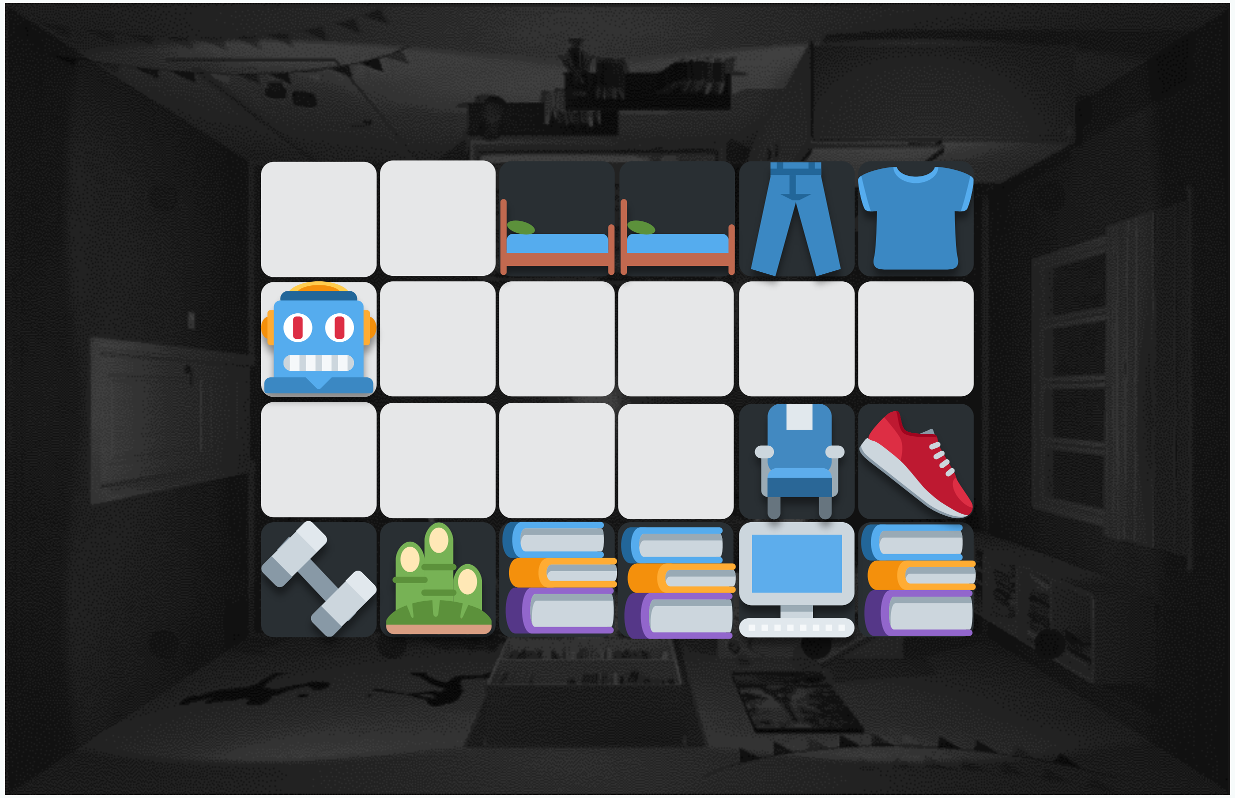


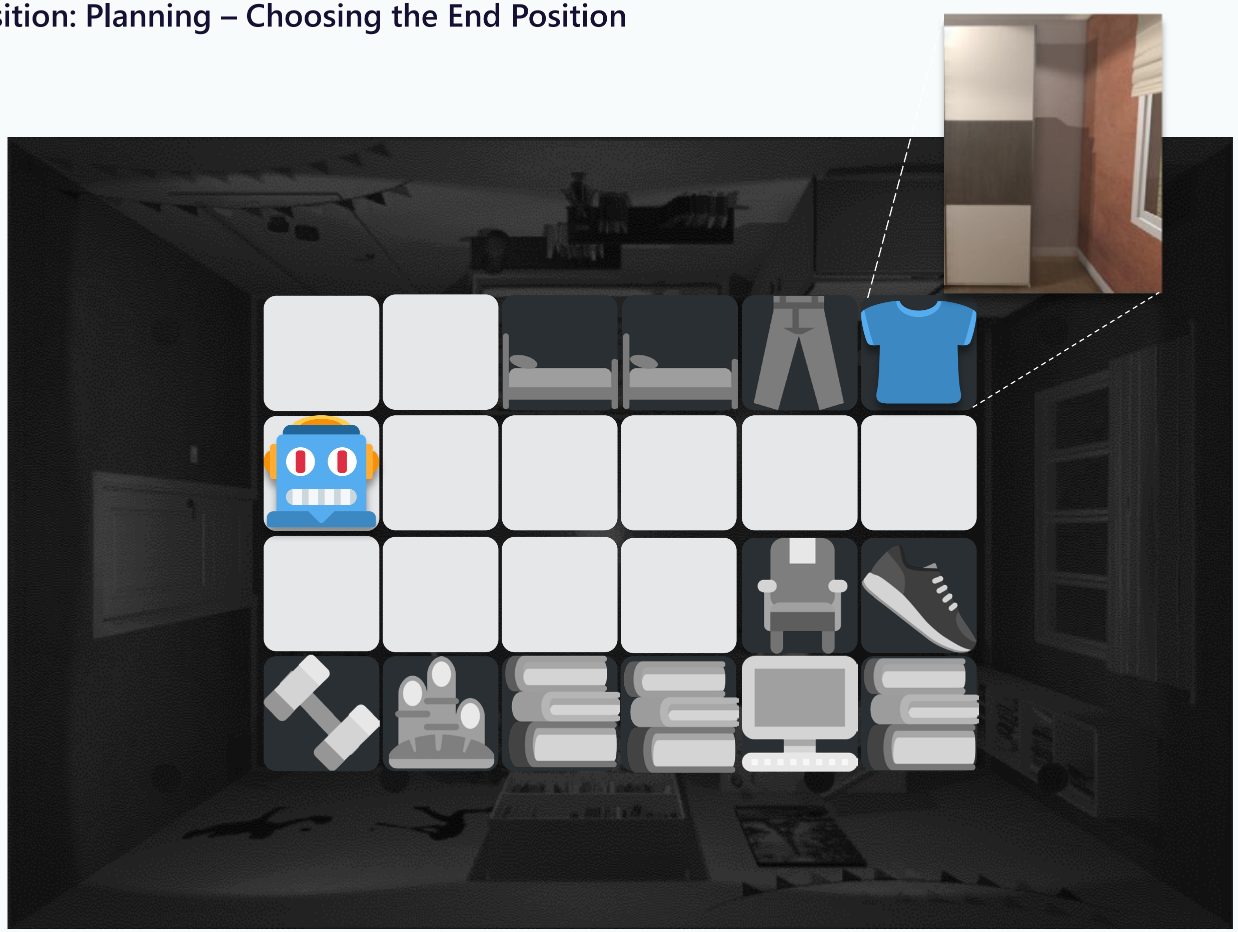
Image → **Perception** → World Modelling → Planning → Action

Image → **Perception** → World Modelling → Planning → Action

PROBLEM – **Visual Navigation Decomposition: World Modelling**



PROBLEM – **Visual Navigation Decomposition: Planning – Choosing the End Position**



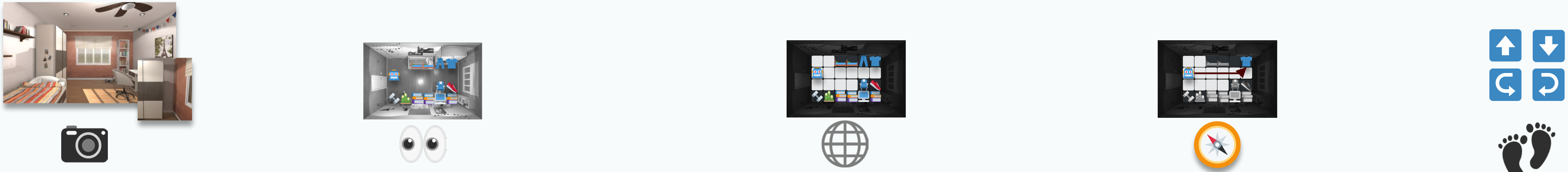
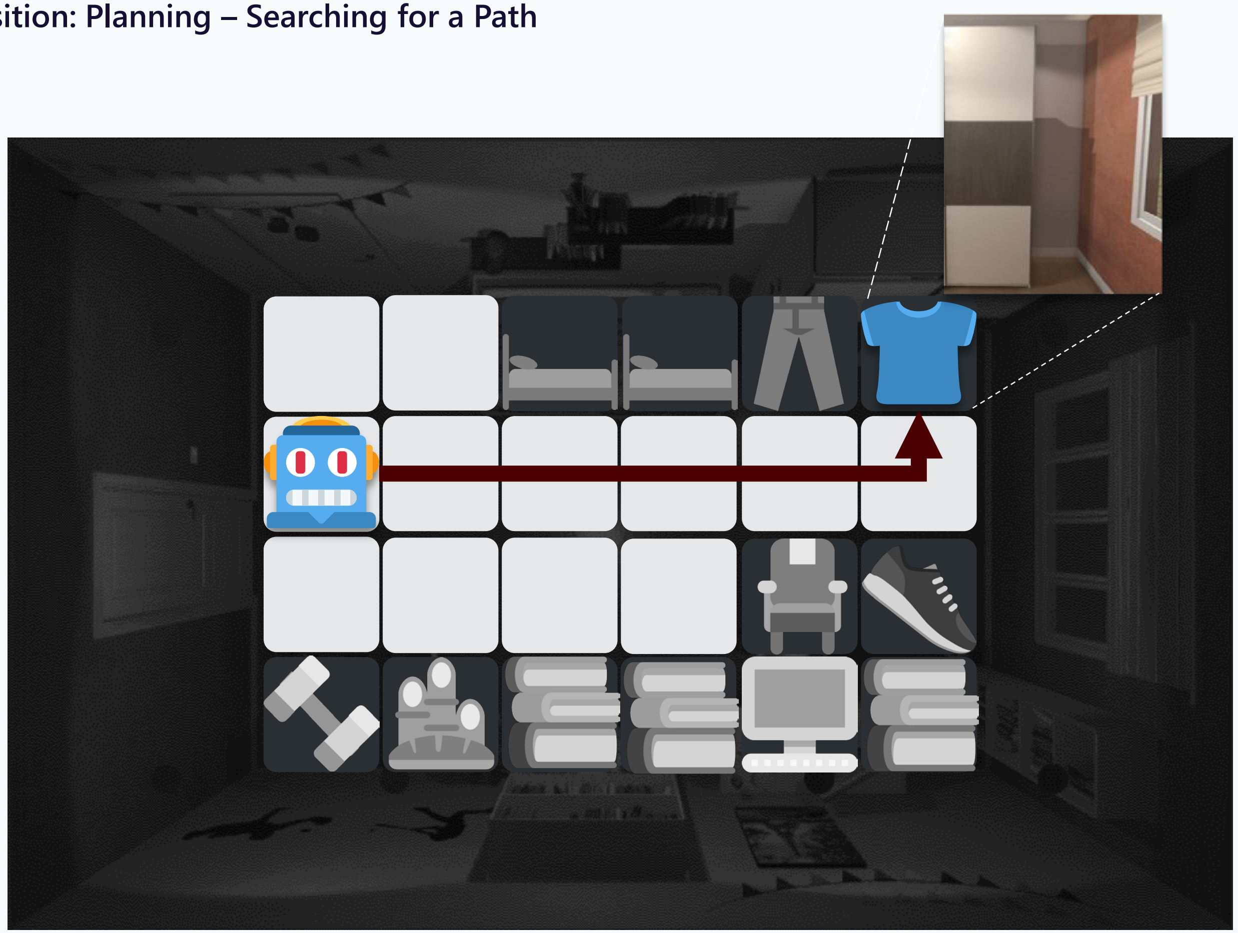


Image → **Perception** → **World Modelling** → **Planning** → **Action**

PROBLEM – **Visual Navigation Decomposition: Planning – Searching for a Path**



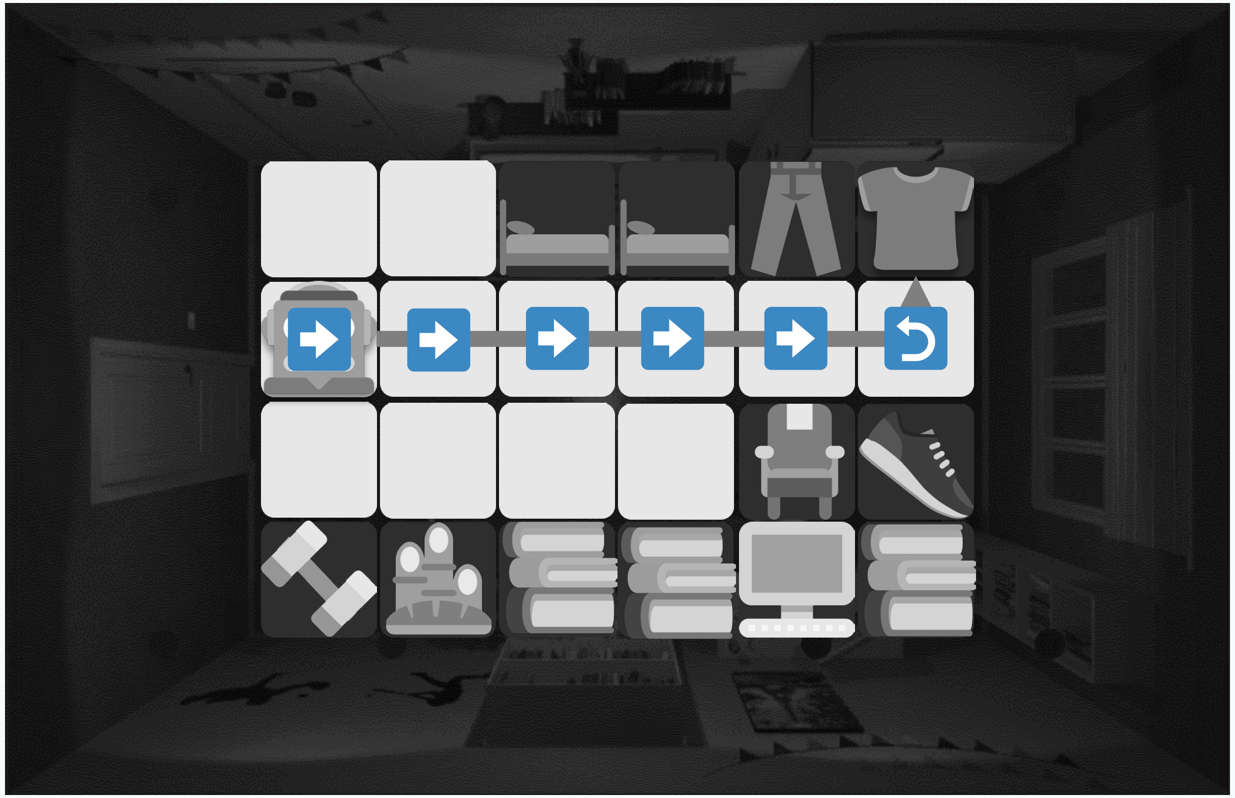
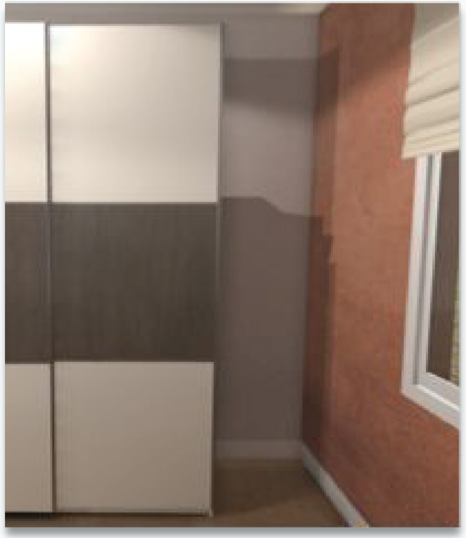



Image
Perception
World Modelling
Planning
Action

This decomposition is effective but each step requires a different algorithm.



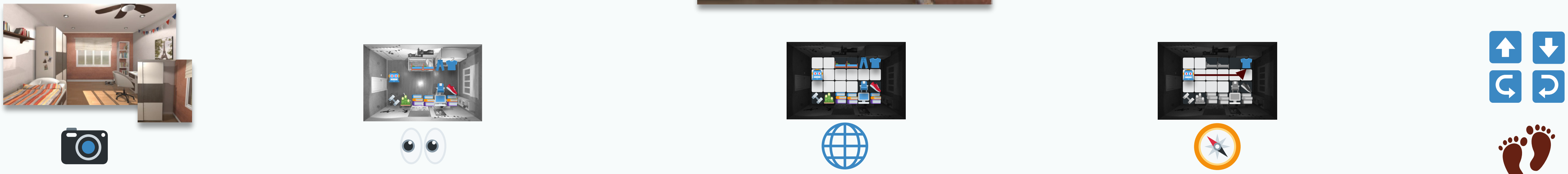


Image → **Perception** → **World Modelling** → **Planning** → **Action**

Design a deep reinforcement learning architecture to handle visual navigation from raw pixels.

Image → **Reinforcement Learning** → **Action**

PROBLEM – **Robot Learning: Reinforcement Learning with a Robot**

SCITOS

Test environment

Image → **Reinforcement Learning** → **Action**

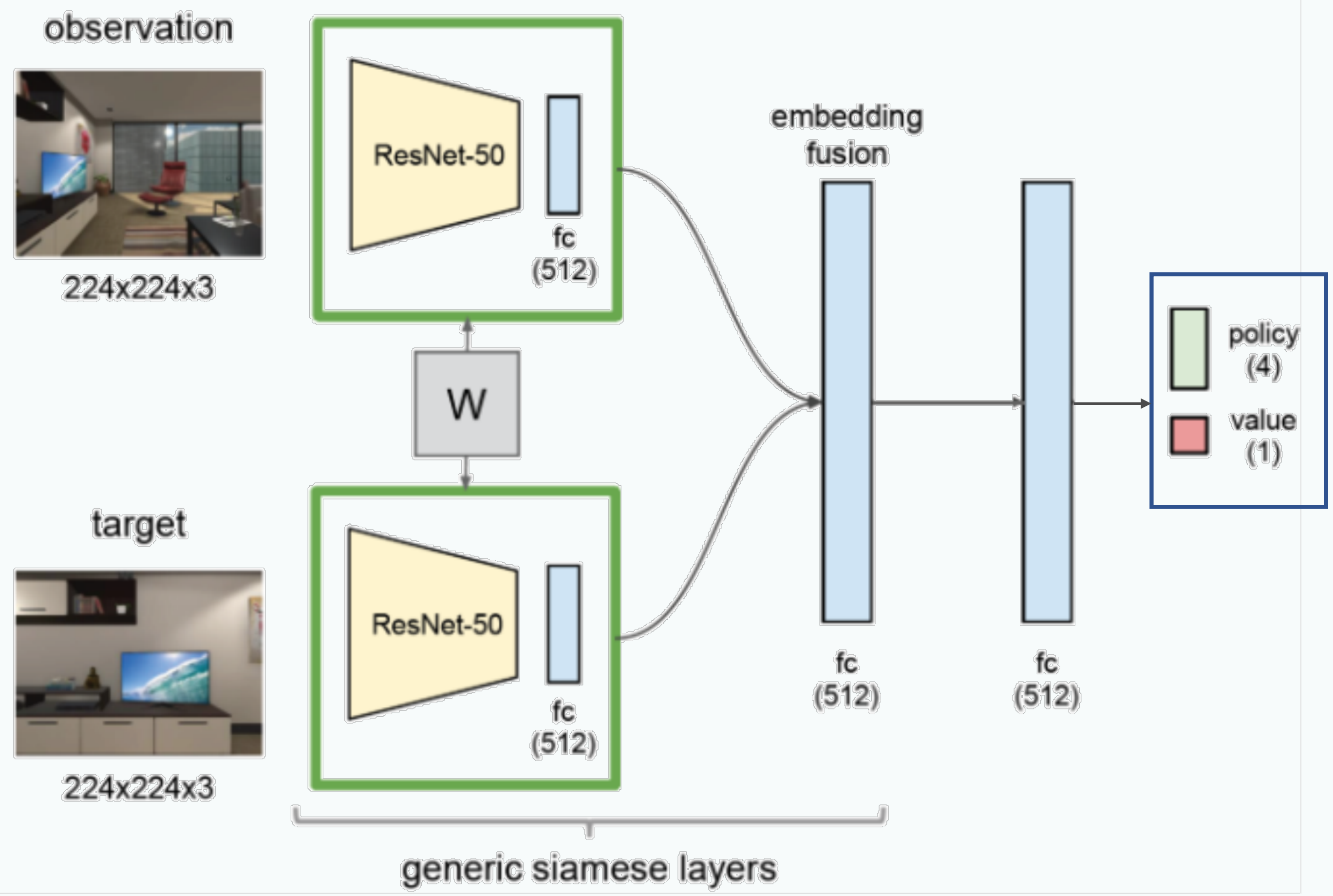
Idea: Train in simulation first, then fine tune learned policy on real robot.

Image → **Reinforcement Learning** → **Action**

Design a deep reinforcement learning architecture to handle visual navigation from raw pixels **with high data-efficiency.**

Image → **Reinforcement Learning** → **Action**

- Similar to actor-critic A3C method which outputs a policy and value running multiple threads.
- Train a different target on each thread rather than copies of same target.
- Use fixed ResNet-50 pretrained on ImageNet to generate embedding for observation and target.
- Fuse the embeddings into a feature vector to get an action and a value.



- **T**he **H**ouse **O**f inte**R**actions (THOR)
- 32 scenes of household environments.
- Can freely move around a 3D environment.
- High fidelity compared to other simulators.

Video: https://youtu.be/SmBxMDiOrvs?t=18

- Single layer for all scenes might not generalize as well.
- Train a different set of weights for every different scene in a vine-like model.
- Discretized the scenes with constant step length of 0.5 m and turning angle of 90°, effectively a grid when training and running for simplicity.

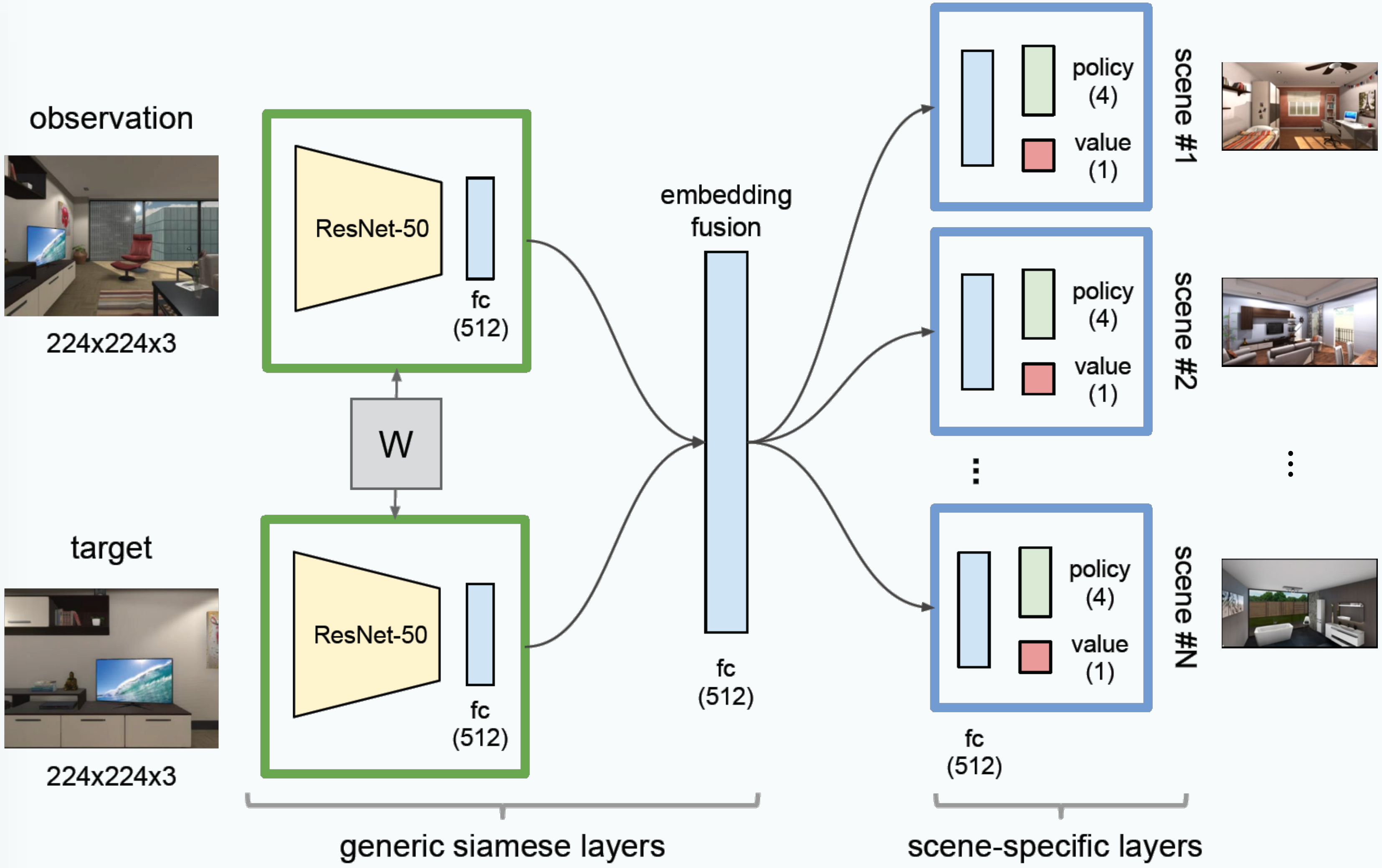
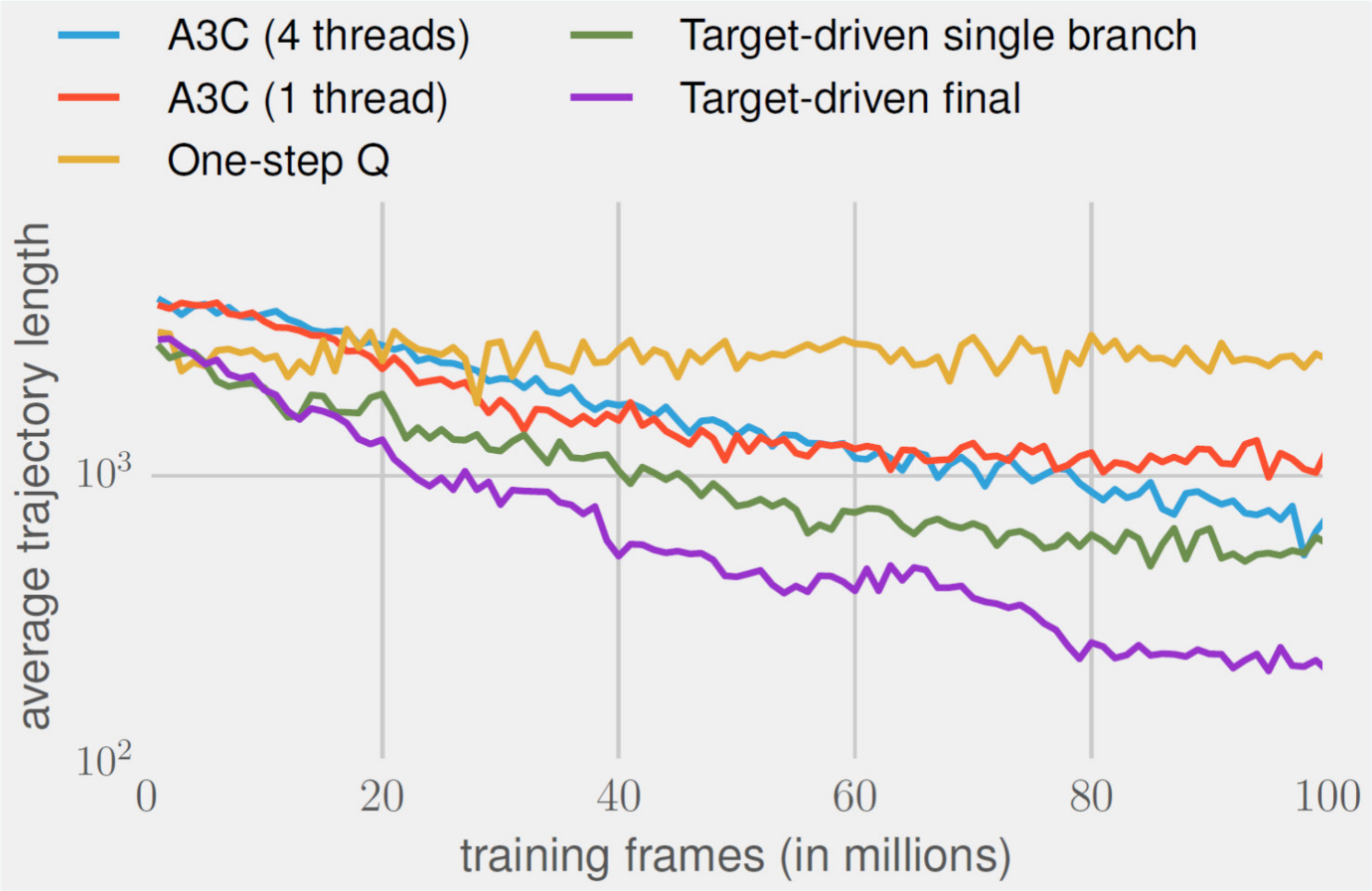


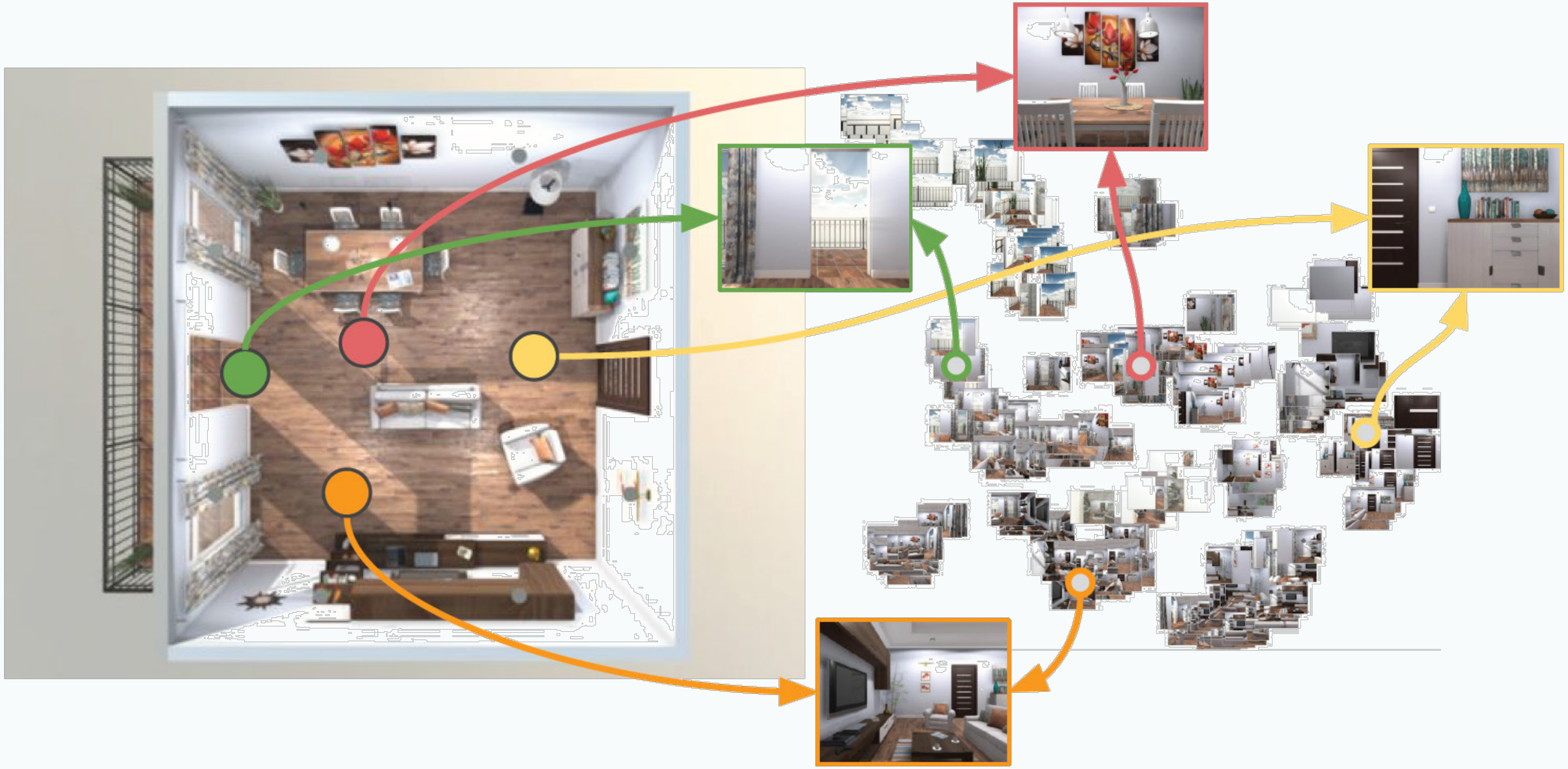
TABLE I

PERFORMANCE OF TARGET-DRIVEN METHODS AND BASELINES

Type	Method	Avg. Trajectory Length
Heuristic	Random walk	2744.3
	Shortest path	17.6
Purpose-built RL	One-step Q	2539.2
	A3C (1 thread)	1241.3
	A3C (4 threads)	723.5
Target-driven RL (Ours)	Single branch	581.6
	Final	**210.7**

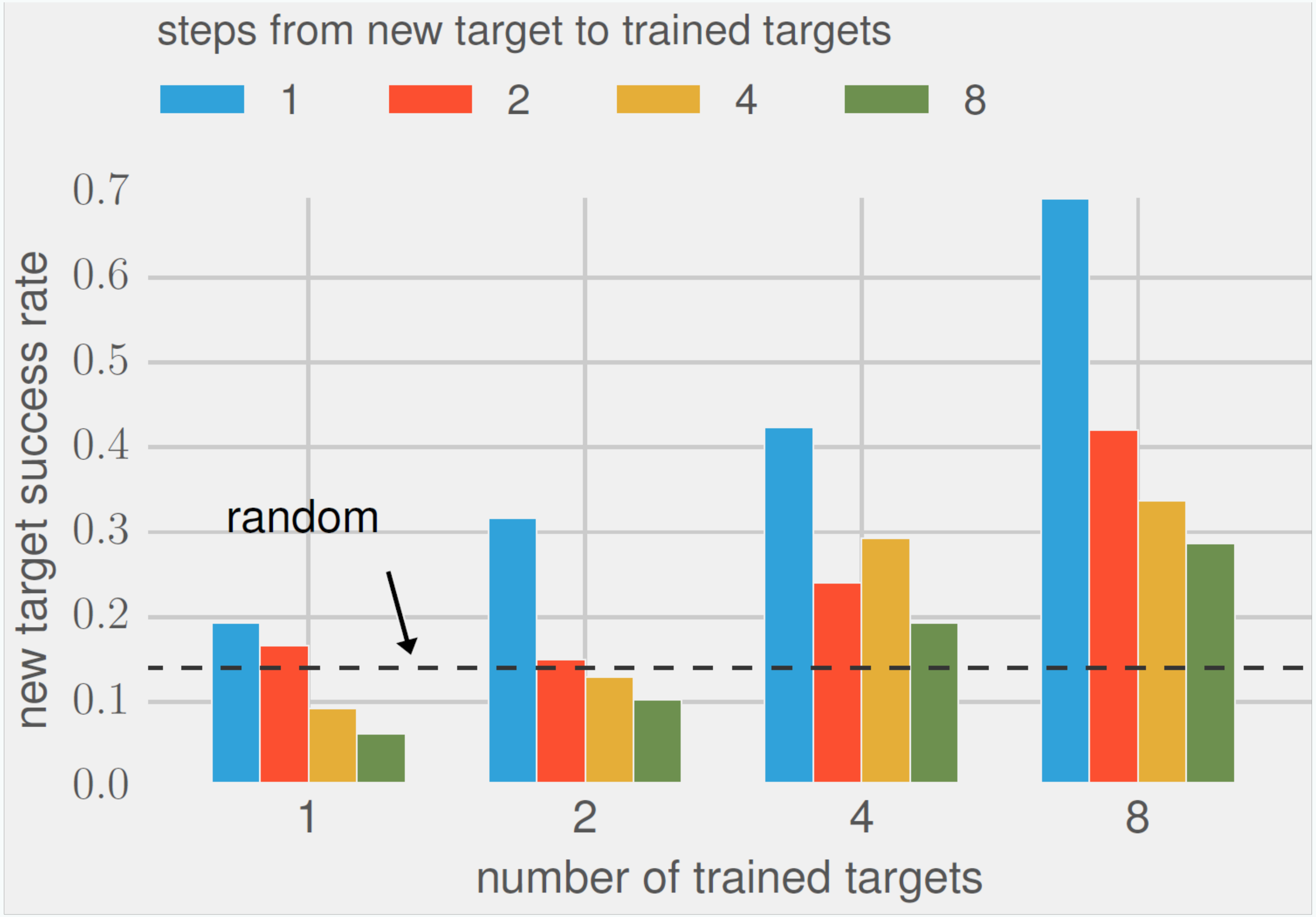
EXPERIMENTS – **Navigation Results: Data Efficiency – Target-driven Method is More Data Efficient**

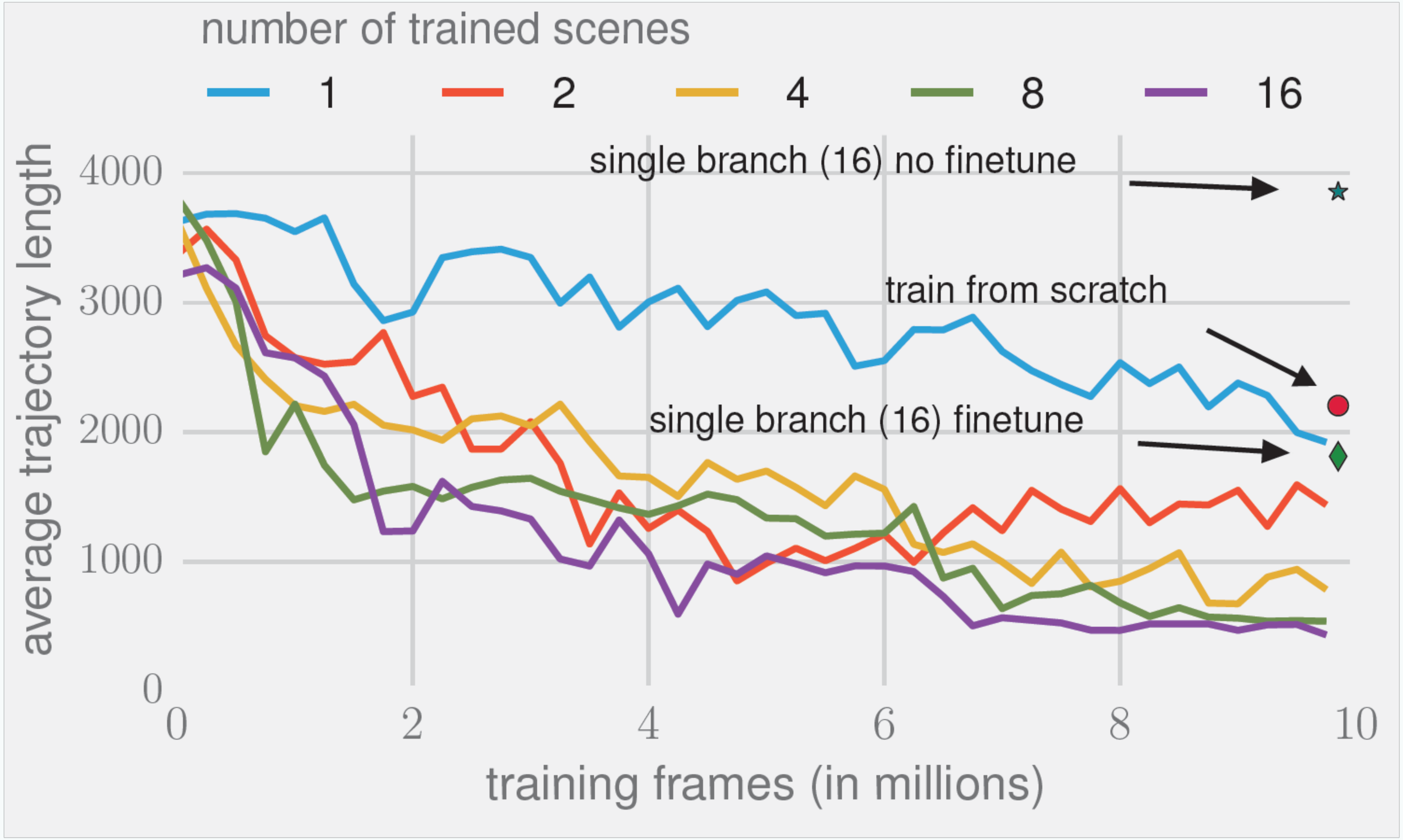




Bird's eye view

t-SNE Visualization of Embedding Vector


steps from new target to trained targets
1
2
4
8
new target success rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
random
1
2
4
8
number of trained targets


number of trained scenes
1
2
4
8
16
single branch (16) no finetune
train from scratch
single branch (16) finetune
average trajectory length
0
1000
2000
3000
4000
0
2
4
6
8
10
training frames (in millions)

- The method was evaluated with continuous action spaces.
- Robot could now collide with things and its movement had noise no longer always aligning with a grid.

- **Result:** Required 50M more training frames to train on a single target.
  - Could reach door in average of 15 steps.
  - Random agent took 719 steps.

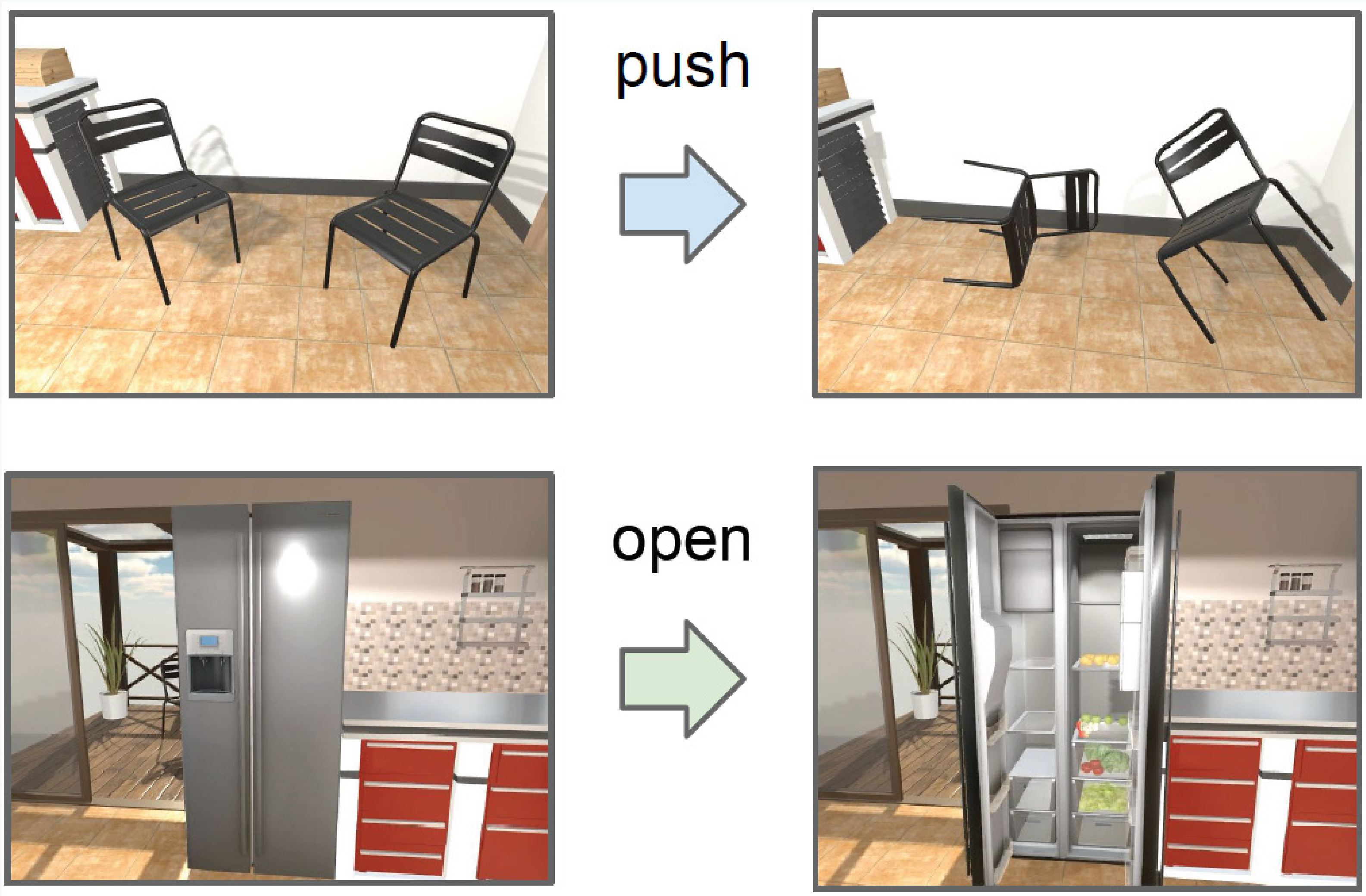
Video: https://youtu.be/SmBxMDiOrvs?t=169

Video: https://youtu.be/SmBxMDiOrvs?t=180

- Designed new deep reinforcement learning architecture using siamese network with scene-specific layers.
- Generalizes to multiple scenes and targets.
- Works with continuous actions.
- The trained policy in simulation can be run in the real world.

Video: https://youtu.be/SmBxMDiOrvs?t=98

- Physical interaction and object manipulation.
- **Situation:** Moving obstacles out of path.
- **Situation:** Opening containers for objects.
- **Situation:** Turning on the lights when the robot enters a dark room and can't see.



Thank You

Thanks to "Twemoji" from Twitter used under license CC-BY 4.0.

- "School" image changed to carry University of Waterloo logo.
- "Quadcopter" created from "Helicopter" and "Robot Face" images.
- "Goose" created from "Duck" image.
- "Red Robot" created from "Robot" image.
- "Dumbbell" created from "Nuts and Bolt" image.

RECAP 🙋‍♂️🙋‍♀️

- **Motivation**
  - Navigating the Grid World, Assign Numerical Rewards, Extract a Policy
  - Applications: Treasure Hunting, Robot Soccer, Pizza Delivery, Self-Driving Cars, Search and Rescue, Domestic Robots
- **Problem**
  - Domestic Robot
  - Multi Target: Can't We Just Use Q-Learning?
  - From Navigation to Visual Navigation
  - Visual Navigation Decomposition: Image, Perception, World Modelling, Planning, Action, Results
  - Goal of Visual Navigation
  - Robot Learning Considerations
- **Neural Network Architecture**
  - Embedding of Scene and Target using ResNet
  - Siamese Network
  - Scene-Specific Layers
- **AI2-THOR Simulator**
  - 32 household scenes with interactions, Trained on discretized representation with no interaction required
- **Experiments**
  - Shorter Average Trajectory Length, More Data Efficient, Training on More Targets Improves Success, Training on More Scenes Reduces Trajectory Length, Works in Continuous World, Transfer Learning Works
- **Future Work**
  - More Sophisticated Interaction