

Large-Scale Robot Learning I

Dhruv Shah

MLSS 2025

Google DeepMind



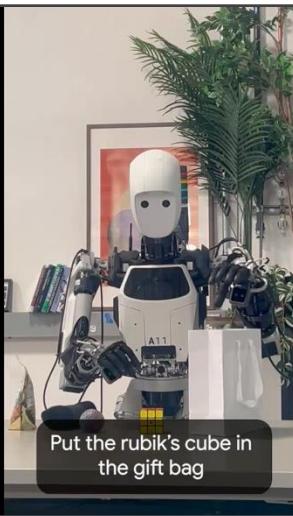
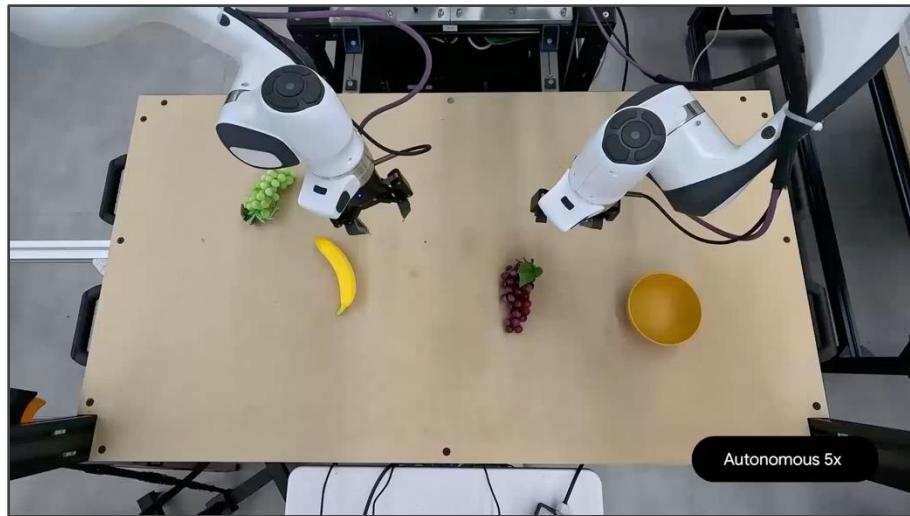
About Me



- Undergraduate in India (IIT Bombay)
 - Control theory, swarms / multi-agent interaction dynamics
- PhD at UC Berkeley
 - Reinforcement learning, long-range navigation, foundation models
- Senior Research Scientist at Google DeepMind
 - Foundation models, scaling up data collection, dexterous manipulation
- Assistant Professor at Princeton University
 - All of the above and more ☺



What do I work on?



Do this over a large number of tasks
and lots of data → general-purpose

Large-Scale Robot Learning

Program systems to autonomously
perform tasks/behaviors from data

Solve problems in the physical world

Data-driven robotics today

Avigal et al. 2021



3X

Gupta et al. 2017



Agarwal et al. 2022



But...

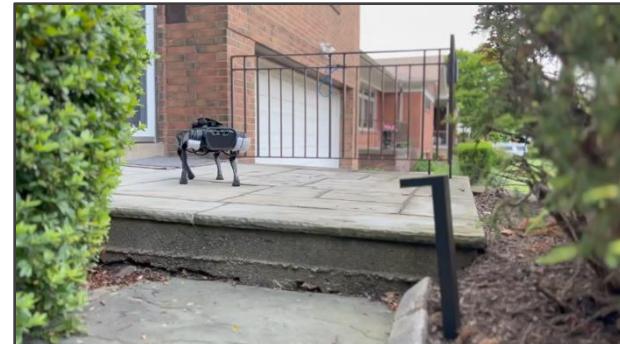
Avigal et al. 2021



Gupta et al. 2017



Agarwal et al. 2022



Works in a very **structured** setting

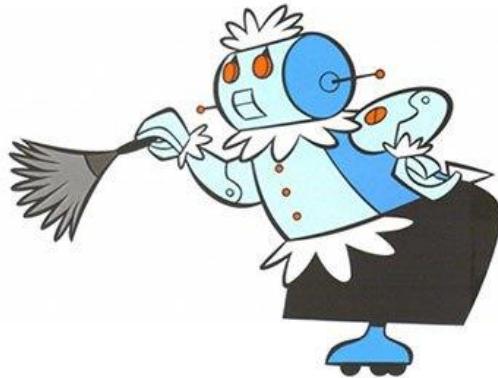
Requires **expensive** trial-and-error

Lacks **semantic** understanding

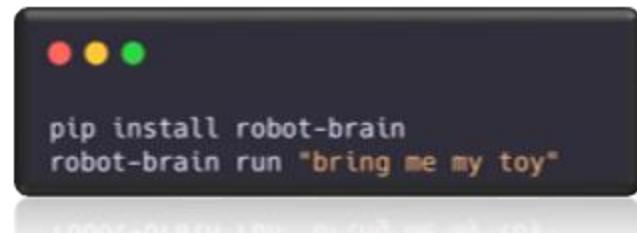
Need to be trained for each separate robot, environment, task.

Don't understand visual affordances, human instructions etc.

What we *really* want



Any robot



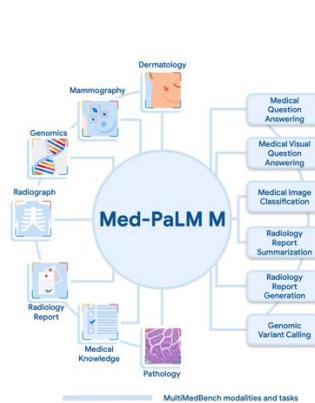
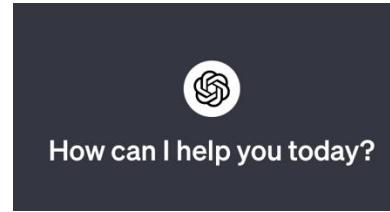
Download intelligent stack



Plug-and-play

Elsewhere...

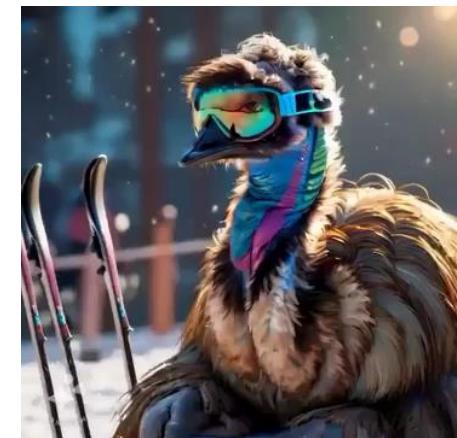
How did these models get here?



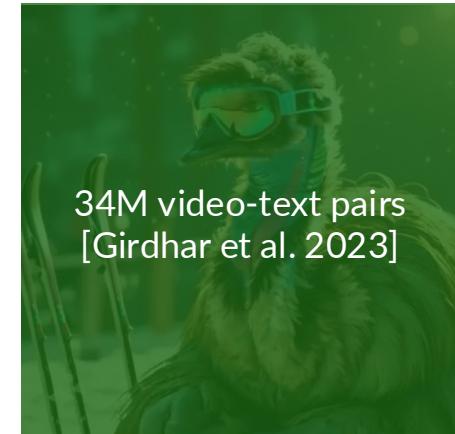
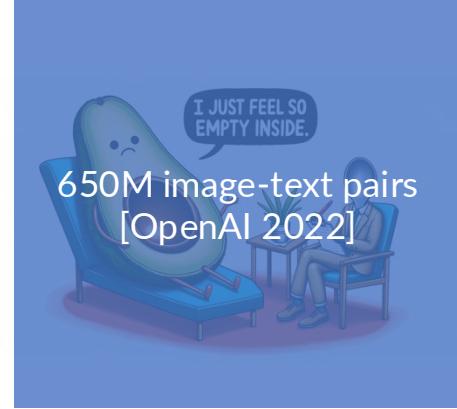
LLaVA Chatbot

What is unusual about this image?

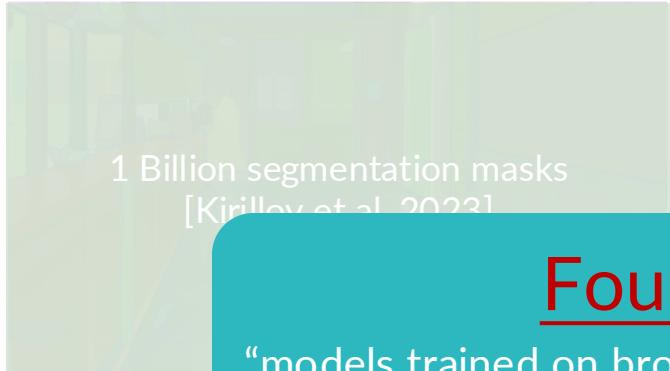
The image shows a person ironing clothes on an ironing board that is placed on the back of a moving vehicle, specifically a yellow taxi. This is an unusual and unsafe situation because ironing clothes while standing on the back of a moving vehicle is not only impractical but also poses a significant risk of injury to the person doing the ironing. The person is exposed to the elements, the risk of falling, and the potential for accidents due to the vehicle's movement. Additionally, the ironing board and the person's position on it could obstruct the driver's view or create a hazard for other road users.



The Secret Sauce



The Secret Sauce



Foundation Models

“models trained on broad data (generally using self-supervision at scale) that can be adapted to a wide range of downstream tasks”



The Secret Sauce



1 Billion segmentation masks
[Kirillov et al. 2023]

500B+ “Tokens”
[OpenAI 2022]

650M image-text pairs
[OpenAI 2022]

Internet Foundation Models

“models trained on broad data (generally using self-supervision at scale) that can be adapted to a wide range of downstream tasks”

780B+ “Tokens”
[Med-PaLM M]
[Singhal et al. 2022]

150K language instructions
[Liu et al. 2022]

The image shows a person ironing clothes on an ironing board that is placed on the back of a moving vehicle, specifically a yellow taxi. This is an unusual and unsafe situation because ironing clothes while standing on the back of a moving vehicle is not only impractical but also poses a significant risk of injury to the person doing the ironing. The person is exposed to the elements, the risk of falling, and the potential for accidents due to the vehicle's movement. Additionally, the ironing board and the person's position on it could obstruct the driver's view or create a hazard for other road users.

34M video-text pairs
[Girdhar et al. 2023]

The Secret Sauce



Internet Foundation Models

[K] “models trained on broad data (generally using self-supervision at scale) that can be adapted to a wide range of downstream tasks”

Scale and consolidation can play a crucial role in data-driven robotics

Course Overview

1. Setup a simple robotics problem: *visual navigation*
2. Design a machine learning method to solve it
3. Scaling this up to a “robot foundation model”
4. How can LLMs and VLMs (LMMs) help robotics

Part 1: Visual Navigation

Problem setup and prior work



Late 2020



2022

Mobile Robots are Coming





Shah et al. "ViNG: Learning Open-World Navigation with Visual Goals" (2020)

Courtesy: Serve Robotics, Amazon Scout, and Starship Technologies

High-Level “Task”



Robot with Ego Camera



Dropped in an Environment



“go to the bus”

Goal

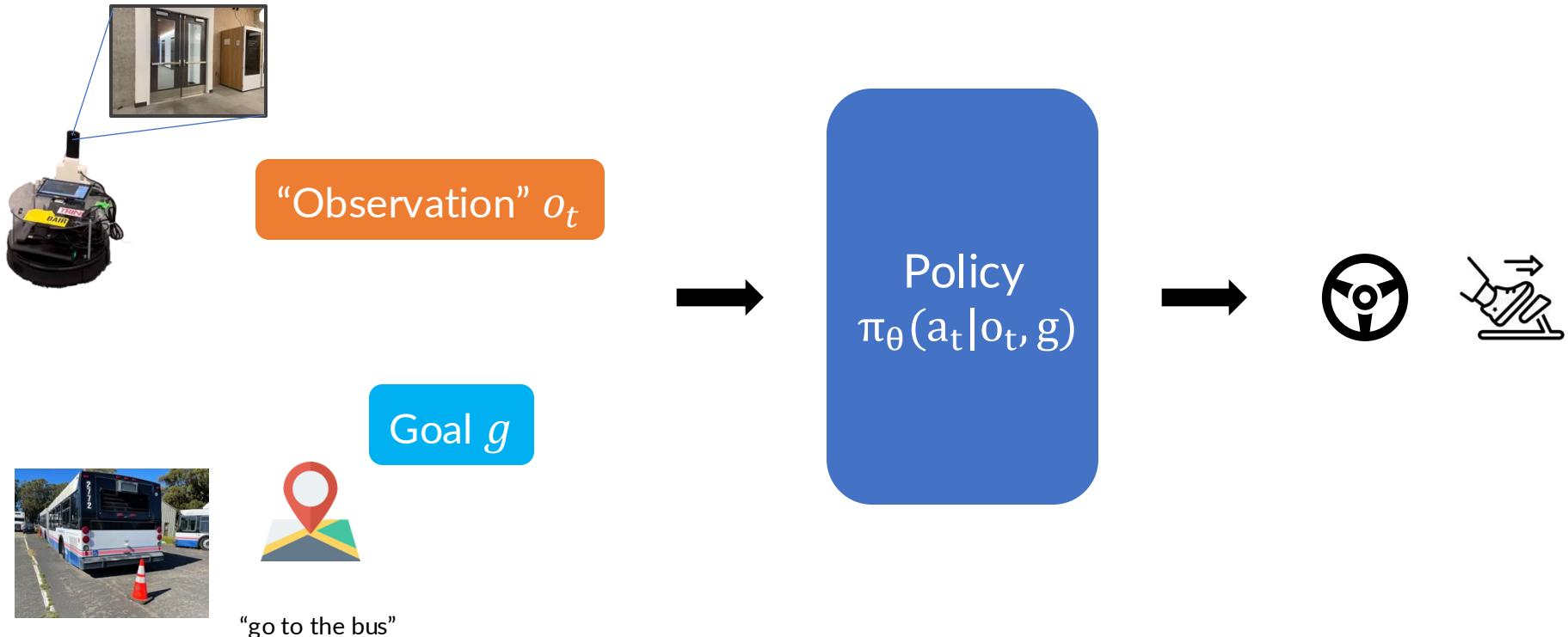
Take a right next to the stop sign. Look for a glass building, after passing by a white car.

Instructions

“drive on the right side”
“avoid freshly mowed lawn”
“avoid driveways”

Behavior

Mathematically



The “Classical” Solution



Robot Observ

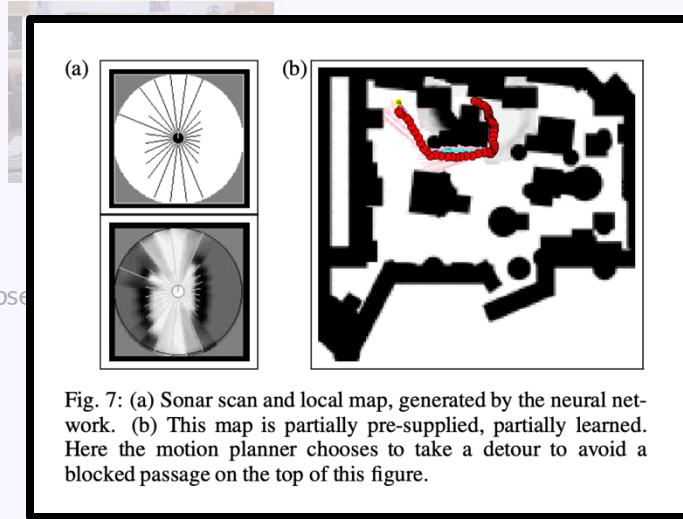


Fig. 7: (a) Sonar scan and local map, generated by the neural network. (b) This map is partially pre-supplied, partially learned. Here the motion planner chooses to take a detour to avoid a blocked passage on the top of this figure.

Burgard et al. 1998

Thrun, Burgard, Fox. "Probabilistic Robotics" (2005)

Davison et al. "MonoSLAM: Real-time Single Camera SLAM" (2007)

Canny. "The Complexity of Robot Motion Planning" (1988)

LaValle and Kuffner. "Rapidly-exploring Random Trees" (2001)

Karaman and Frazzoli. "Sampling-based Algorithms for Optimal Motion Planning" (2011)

Videos Courtesy of Davide Scaramuzza and Sertac Karaman

RGBD SLAM on the Ground Robot
using the Kinect Sensor



DARPA Grand Challenge 2007





JPL

Geometric Representations

Too Much

Too Little

Intricate 3D Information Unnecessary

- Humans can do without 3D reconstruction
 - Landmarks and cognitive maps

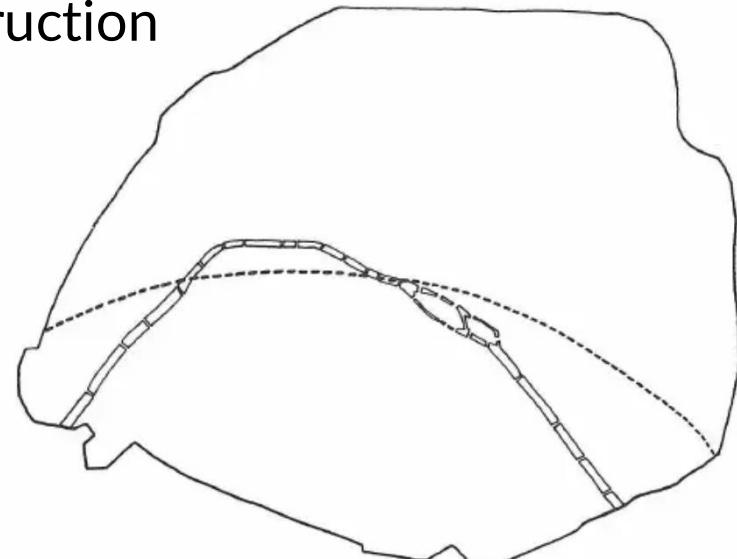
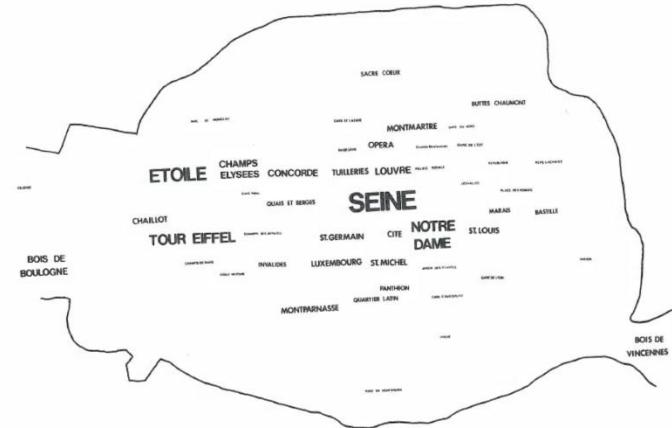


FIGURE 8.5
Perceived curvature of the Seine. The dotted line represents the median curvature imparted to the Seine in the subject's handdrawn maps. It is superimposed on the actual course of the river.

Tolman. "Cognitive Maps in Rat and Men" (1948)

Jodelet and Milgram. "Psychological Maps of Paris" (1976)

Gupta et al. "Unifying Map and Landmark Based Representations for Visual Navigation" (2017)

Geometric Representations

Too Much

- Full 3D reconstruction of the scene may not be needed

Too Little

Lacks Traversability Information

- Binary notion of traversability/occupancy not sufficient



Doesn't Benefit from Experience



Geometric Representations

Too Much

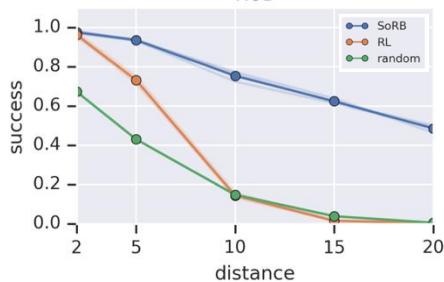
- Full 3D reconstruction of the scene may not be needed

Too Little

- Lacks traversability information
- Doesn't benefit from experience

The “Modern” Solution

- Learn goal-reaching policies from data
 - Imitating an expert
 - Improving using past experience
- “Flat” policies tend to not be sufficient
 - Credit assignment and model errors

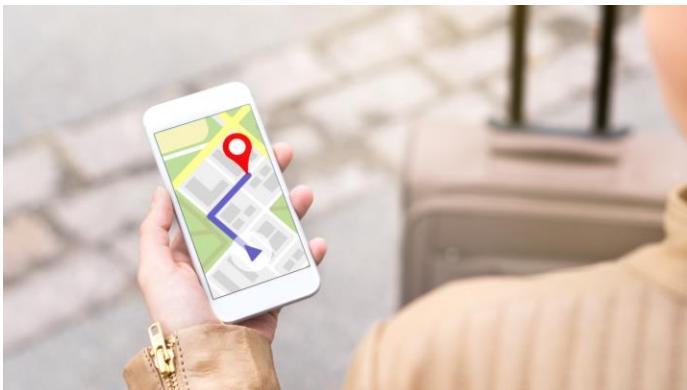


Part 2: Learning Navigation

Let's design a long-range navigation system using machine learning

Problem Statement

- Vision-based navigation from egocentric observations



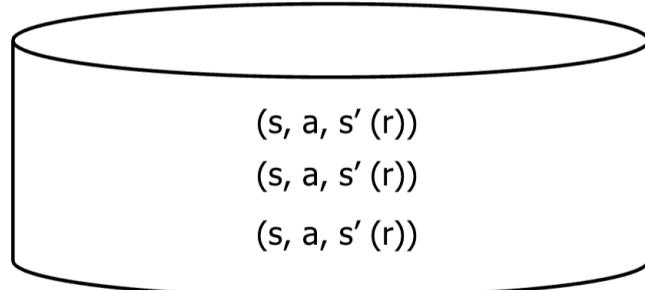
- Explore previously unseen environments
- Use “side information” like a human would

Recall: Data-driven decision making

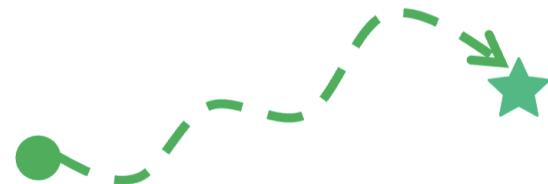
Formalism for data-driven decision making

We will study how to develop algorithms that go from logged data to a sequential decision making policy

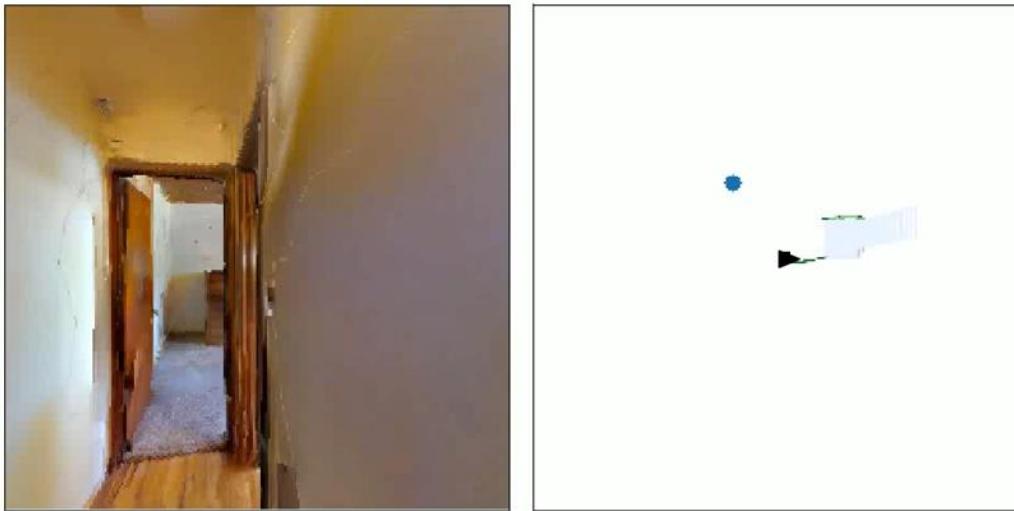
Given: Dataset of prior interactions



Goal: Find performant policy π_θ



Learning from Simulated Data

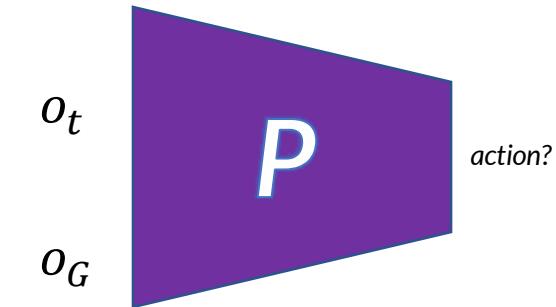
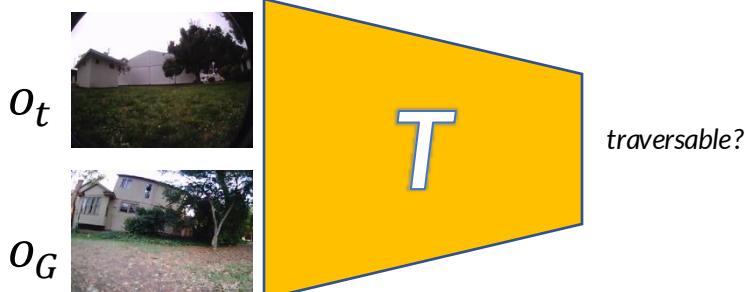


Learning from large, diverse data

- ▼ Needs photorealistic simulator
- ▼ sim2real transfer is hard
- ▼ Needs lots of online experience

Learning from Real-World Data

- Learn policy and distance function from scripted offline data



Key Component: Topological Graph

- Observations are nodes
- Edges are estimated traversability

Topological Graphs in Robotics

- Topological graphs were proposed in the 1990s
 - Simplifying assumptions, not practical
- The deep learning rebirth: SPTM
 - “Seen” environment, simple 2D navigation

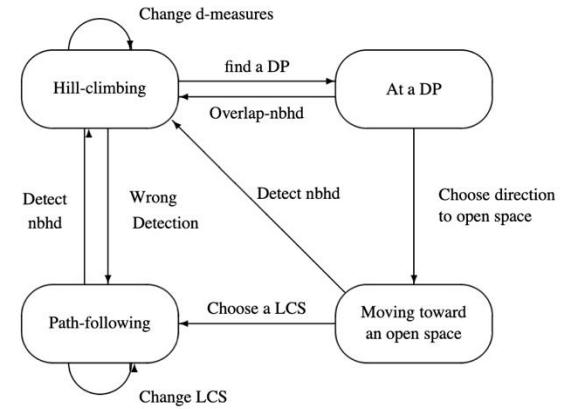
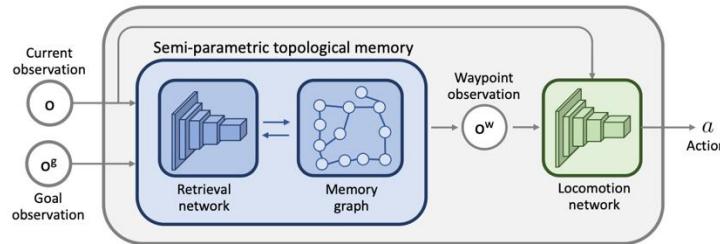


Figure 4: A state-event diagram for the exploration strategy

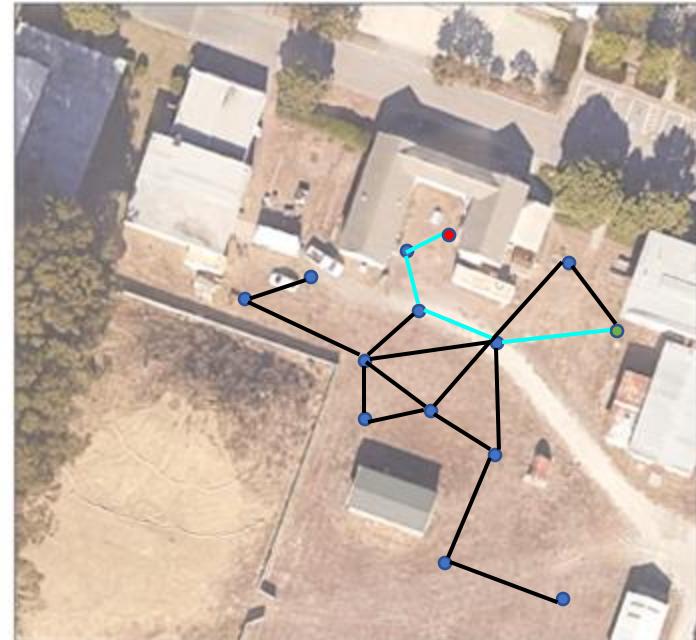
Kuipers and Byun. “A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations” (1991)

Kuipers. “The Spatial Semantic Hierarchy” (2000)

Savinov et al. “Semi-parametric Topological Memory for Navigation” (2018)

Key Component: Topological Graph

- Observations are nodes
- Edges are estimated traversability
- Features
 - Non-metric
 - Represent task-relevant connectivity
 - Plan through this representation



The Simple Case

pre: train GC distance function and policy

use **T** to infer graph edges

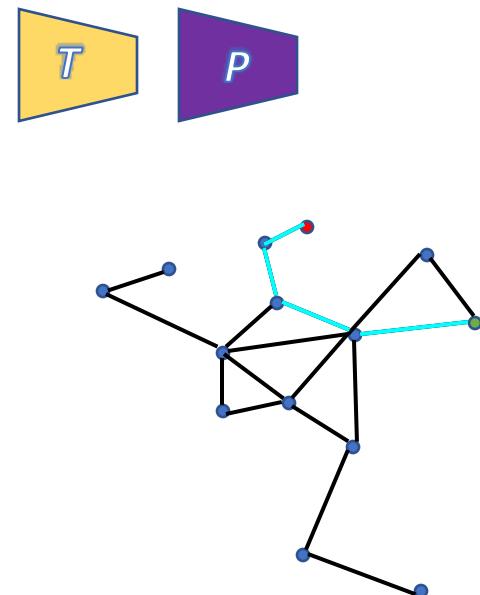
localize **observation** and **goal** in graph

while (*not at goal*):

 get **plan to goal** from topological graph

 execute policy **P** to reach first subgoal

 localize observation in graph



Satellite view for visualization purposes only



Goal



Goal Location

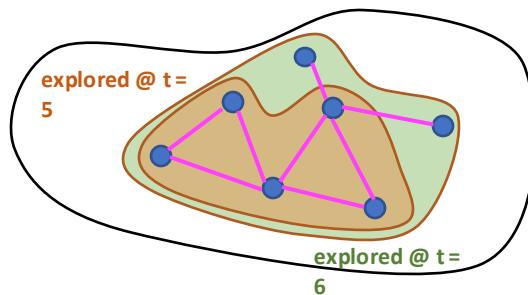


ViNG

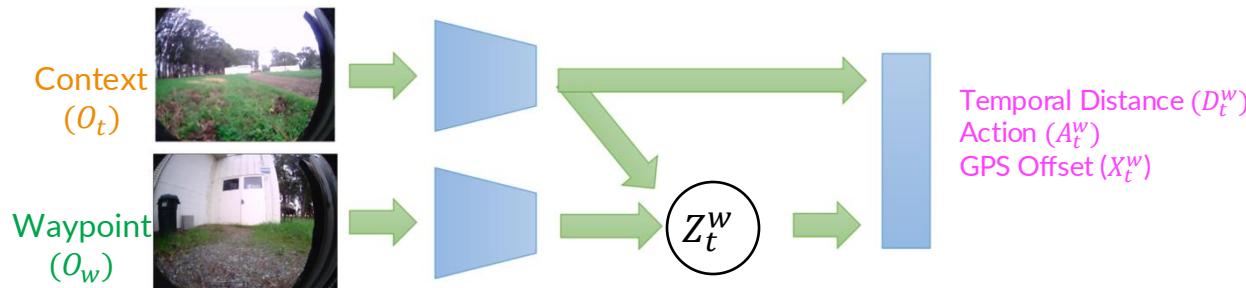
What about unseen environments?

How do we visit *new* states?

1. Reachability prior
2. Frontier-based exploration algorithm



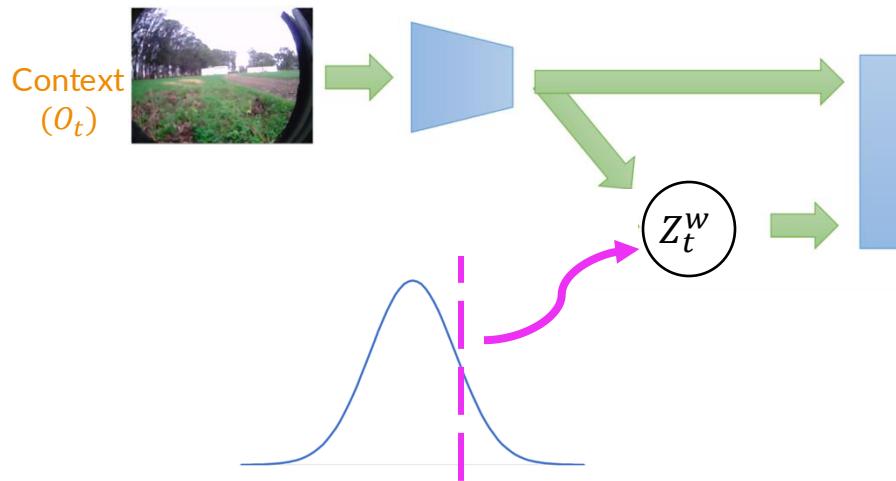
Key Component: Latent Goal Model



- context-conditioned
- implemented as a variational information bottleneck (VIB)

$$I((A_t^w, D_t^w, X_t^w); Z_t^w | o_t) - \beta I(Z_t^w; o_w | o_t)$$

Reachability Prior





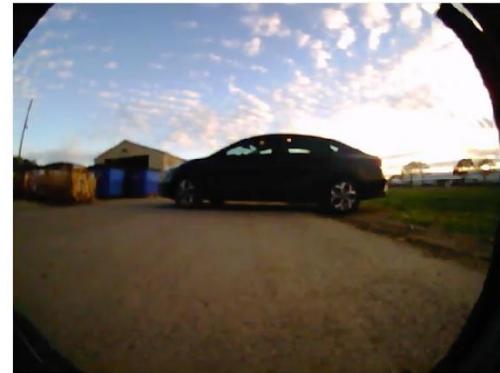
Perturbing the environment with unseen obstacles

Target Image

RECON is able to avoid the obstacles and reliably navigate to the dumpster in each case.



Traffic Cones



Car



Trashcans



Target Image

Bonus: Robustness to Distractors

RECON is able to reliably navigate to the dumpster in each case, suggesting that it learns an invariance to such visual distractors.



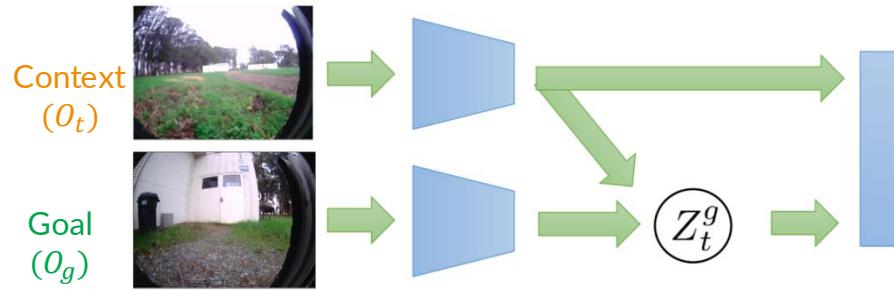
Rainy



Sunny

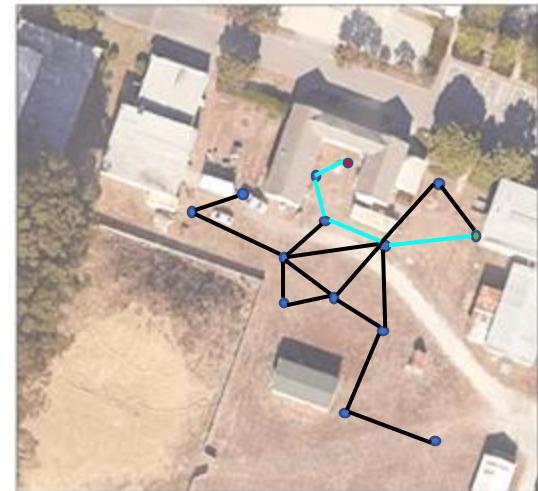


Twilight



Distance (D_t^g)
Action (A_t^g)

Topological Graph



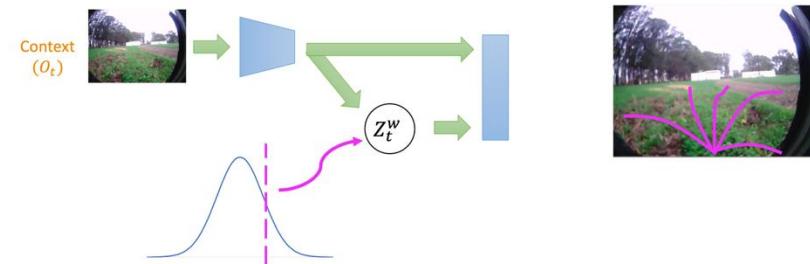
The Challenges with Physical Search

- Not purely *computational* (BFS, Dijkstra, A*, ...)
- Search algorithms assume:
 - ~~the ability to visit an arbitrary node~~ Robots cannot teleport!
 - ~~access to a set of neighbors and edges~~ Real-world doesn't specify neighbors

Changes for Physical Search

Two changes from A*:

- Use a latent variable model to represent feasible subgoals
 - *Sample* feasible neighbors
 - *Predict* estimated cost-to-drive
- Use estimated cost-to-drive when computing the g -score



Exploration with RECON

```
while (not at goal):  
    if (goal is feasible):  
        rollout directly to goal  
  
    else if (robot at the frontier):  
        explore via rollout to imagined goals  
  
    else  
        rollout to frontier of the graph  
  
fine-tune models with environment data  
update graph nodes and edges  
(distances)
```



Strategy
-based)

Exploration with RECON

```
while (not at goal):
    if (goal is feasible):
        rollout directly to goal
    else if (robot at the frontier):
        explore via rollout to imagined goals
    else
        rollout to frontier of the graph
update models and graph
```

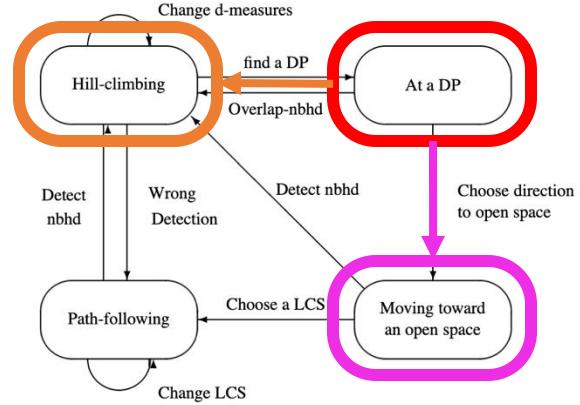


Figure 4: A state-event diagram for the exploration strategy

When deployed in a *previously unseen* environment, RECON explores the environment using a latent goal model.

Run 1: Exploration



Target Image



Goal Location



RECON can build a topological map of the environment from this traversal and use it for future navigation.



- ~ Exploration Path
- ~ Discovered Path
- Start
- Map Node
- Goal Image

Run 2



Target Image



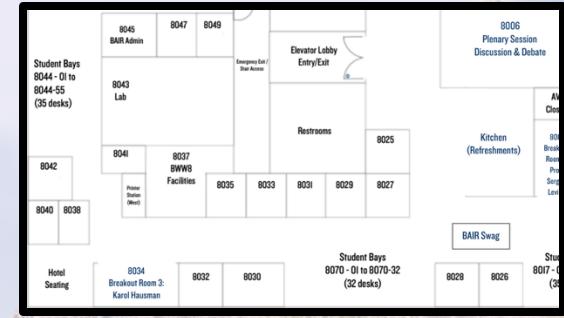
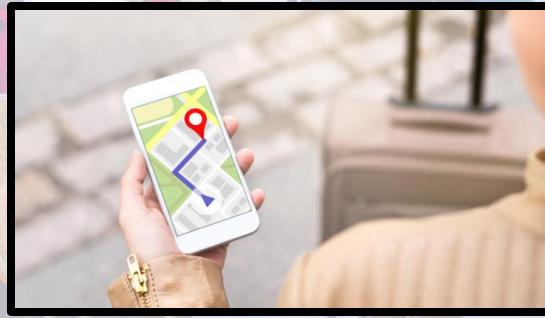
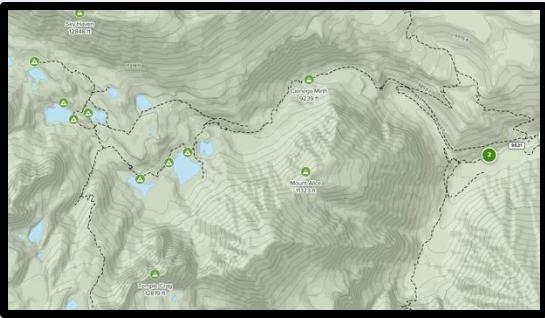
Goal Location



Works, but clumsy!

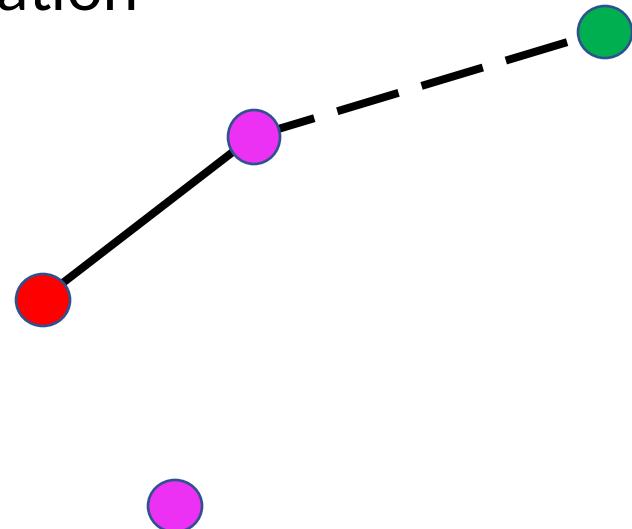


~~~~ Exploration Path      ● Start      ○ Goal Image  
~~~ Discovered Path      ○ Map Node



Biasing Search with a Heuristic

- Informed search using goal information
- Naively use GPS
 - $h(w) = |x_g - x_w|$



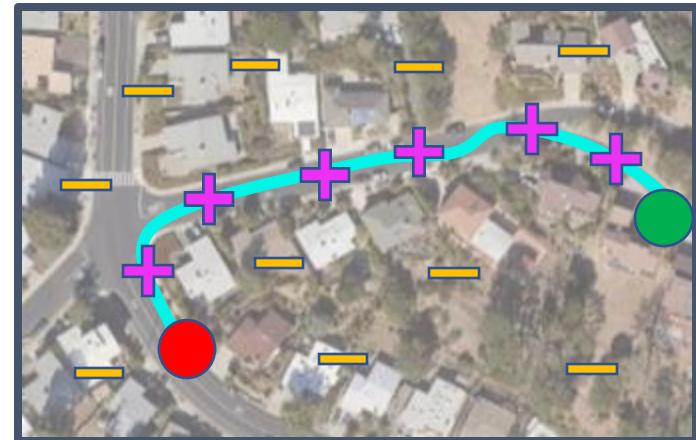
Biasing Search with a Heuristic

- Using geographic “hints”



Learning a Goal-Directed Heuristic

- Train an estimator $p(w \rightarrow G | x_w, x_G, x_t, c_t)$
 - Contrastive learning on training data (InfoNCE)
- $h := \lambda(1 - p)$



Algorithm Summary

Algorithm 1 *ViKiNG-A* for Physical Search*

- 1: **function** ViKiNG-A*(start S , goal info o_G , x_G)
- 2: $\Omega \leftarrow \{S\}$
- 3: **while** Ω not empty **do**

Algorithm Summary

Algorithm 1 *ViKiNG-A* for Physical Search*

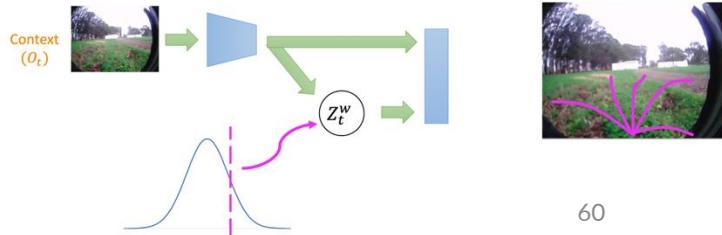
```
1: function ViKiNG-A*(start  $S$ , goal info  $o_G, x_G$ )
2:    $\Omega \leftarrow \{S\}$ 
3:   while  $\Omega$  not empty do
4:      $w_t \leftarrow \min(\Omega, f)$ 
5:     DriveTo( $w_t$ )            $\triangleright$  update visitations  $v$  on the way
6:     observe image  $o_t$ 
```



Algorithm Summary

Algorithm 1 *ViKiNG-A* for Physical Search*

```
1: function VIKING-A*(start  $S$ , goal info  $o_G, x_G$ )
2:    $\Omega \leftarrow \{S\}$ 
3:   while  $\Omega$  not empty do
4:      $w_t \leftarrow \min(\Omega, f)$ 
5:     DriveTo( $w_t$ )            $\triangleright$  update visitations  $v$  on the way
6:     observe image  $o_t$ 
7:
8:     if close( $o_t, o_G$ ) finish   $\triangleright$  use  $q_{\theta, \phi}(\{a, d, x\}_t^w | o_t, o_G)$ 
9:     remove  $w_t$  from  $\Omega$ 
10:    sample waypoints  $w$  near  $w_t$  (Section III-A)
```



Algorithm Summary

Algorithm 1 *ViKiNG-A* for Physical Search*

```
1: function ViKiNG-A*(start  $S$ , goal info  $o_G, x_G$ )
2:    $\Omega \leftarrow \{S\}$ 
3:   while  $\Omega$  not empty do
4:      $w_t \leftarrow \min(\Omega, f)$ 
5:     DriveTo( $w_t$ )            $\triangleright$  update visitations  $v$  on the way
6:     observe image  $o_t$ 

8:     if close( $o_t, o_G$ ) finish   $\triangleright$  use  $q_{\theta, \phi}(\{a, d, x\}_t^w | o_t, o_G)$ 
9:     remove  $w_t$  from  $\Omega$ 
10:    sample waypoints  $w$  near  $w_t$  (Section III-A)

13:    for each waypoint  $w \in \Omega$  do
14:       $f(w) = g(t, w) + d_{\text{Pr}[w]}^w + h(w)$ 
15:    return failure
```



Algorithm Summary

Algorithm 1 *ViKiNG-A* for Physical Search*

```
1: function VIKING-A*(start  $S$ , goal info  $o_G, x_G$ )
2:    $\Omega \leftarrow \{S\}$ 
3:   while  $\Omega$  not empty do
4:      $w_t \leftarrow \min(\Omega, f)$ 
5:     DriveTo( $w_t$ )            $\triangleright$  update visitations  $v$  on the way
6:     observe image  $o_t$ 
7:     add  $w_t$  to graph  $\mathcal{T}$      $\triangleright$  use  $q_{\theta, \phi}$  on  $o_t$  to get distances
8:     if close( $o_t, o_G$ ) finish   $\triangleright$  use  $q_{\theta, \phi}(\{a, d, x\}_t^w | o_t, o_G)$ 
9:     remove  $w_t$  from  $\Omega$ 
10:    sample waypoints  $w$  near  $w_t$  (Section III-A)
11:    for each  $w$  sampled near  $w_t$  do
12:      if not contains( $\Omega, w$ ) then add  $w$ 
13:      for each waypoint  $w \in \Omega$  do
14:         $f(w) = g(t, w) + d_{\text{Pr}[w]}^w + h(w)$ 
15:    return failure
```

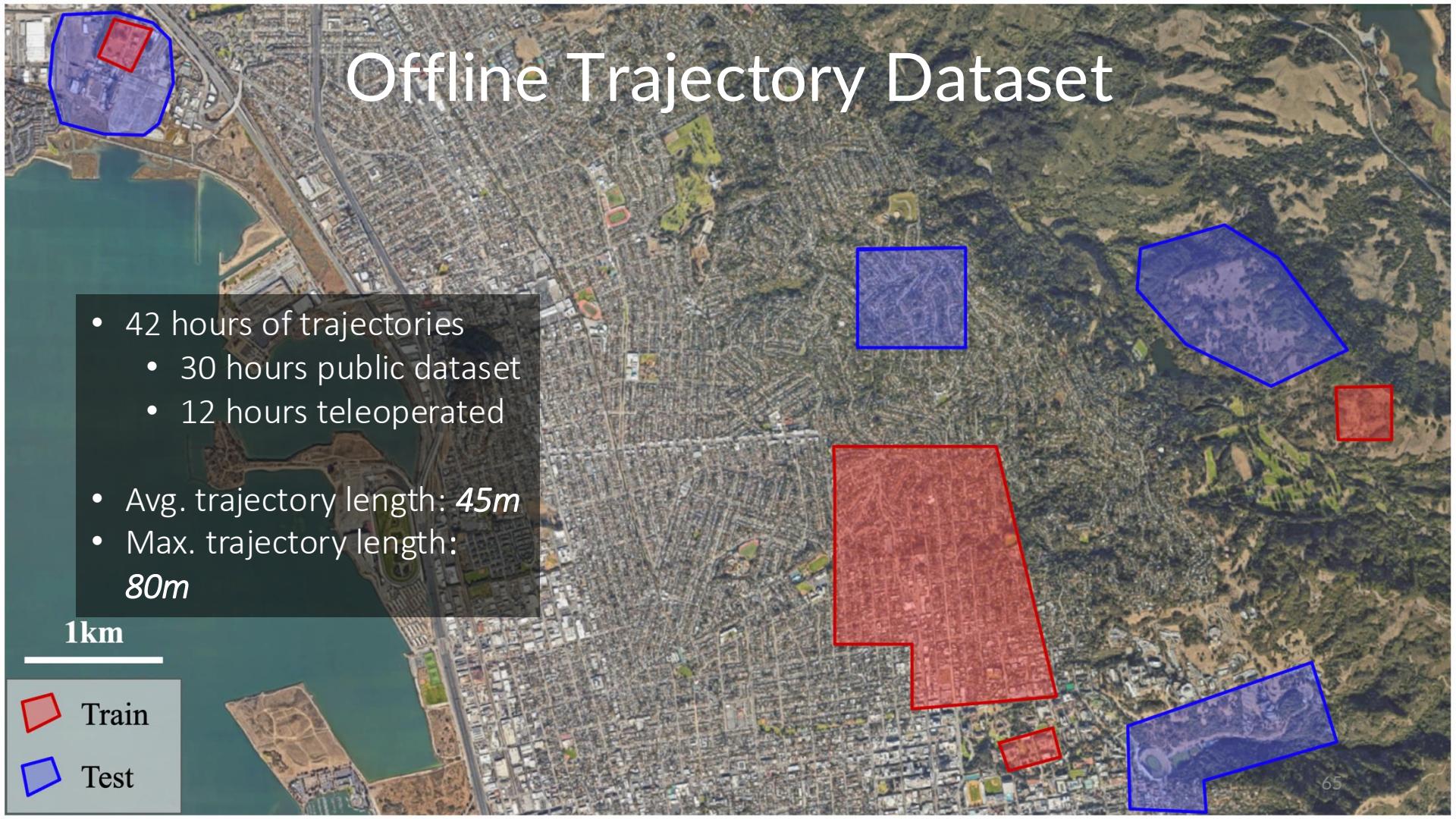
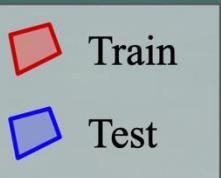
Where do we get the data?



Offline Trajectory Dataset

- 42 hours of trajectories
 - 30 hours public dataset
 - 12 hours teleoperated
- Avg. trajectory length: **45m**
- Max. trajectory length:
80m

1km

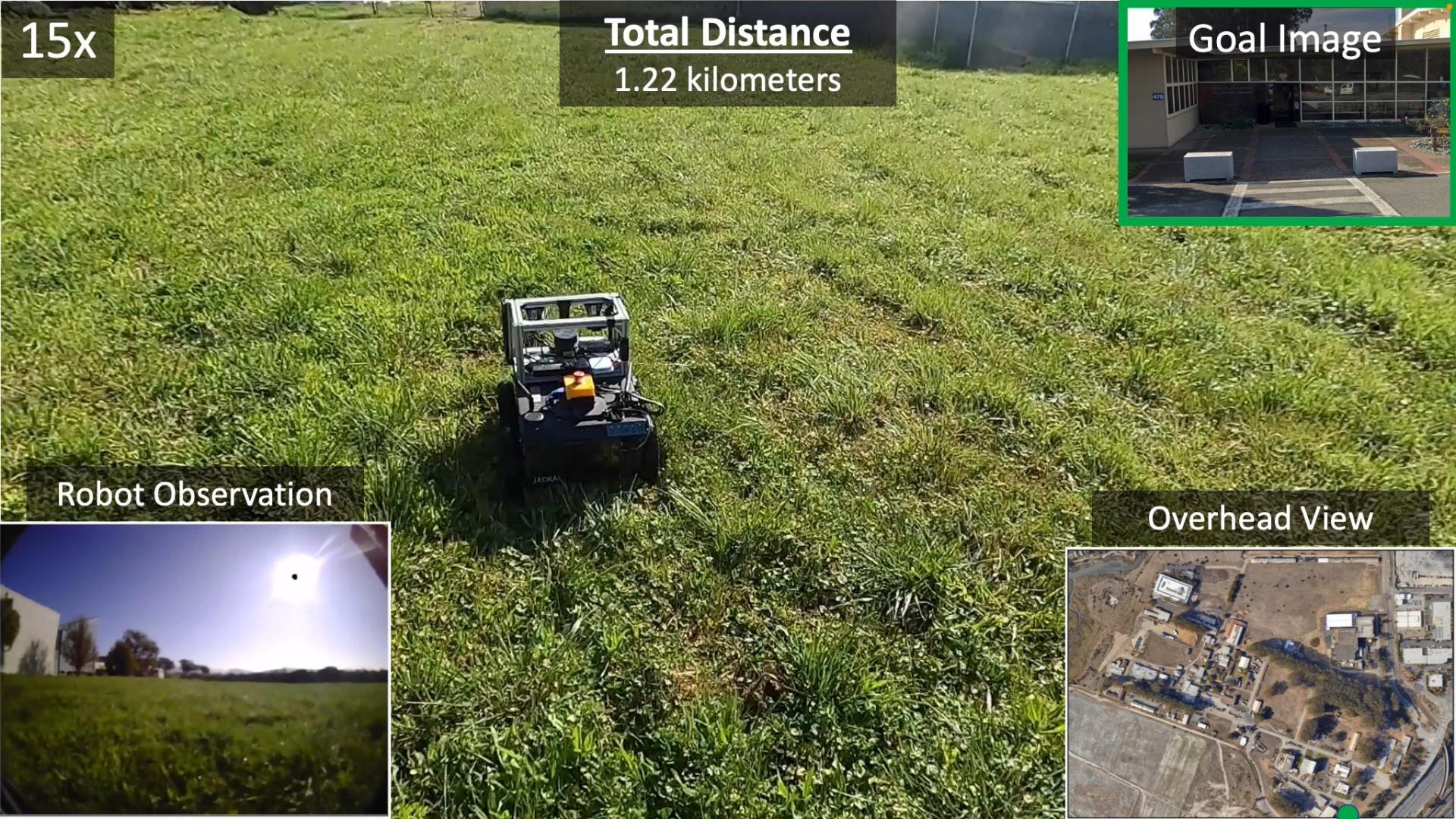


Deploying ViKiNG in the Real World

15x

Total Distance
1.22 kilometers

Goal Image



Robot Observation

Overhead View



15x

Total Distance
782 meters

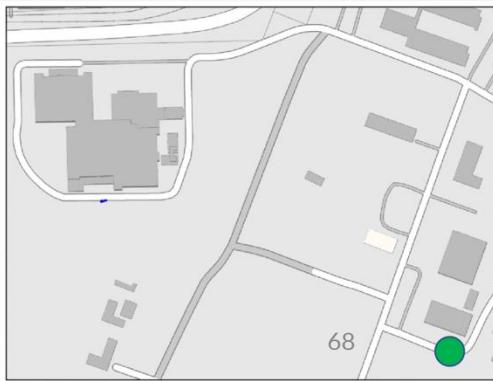
Goal Image



Robot Observation



Overhead View



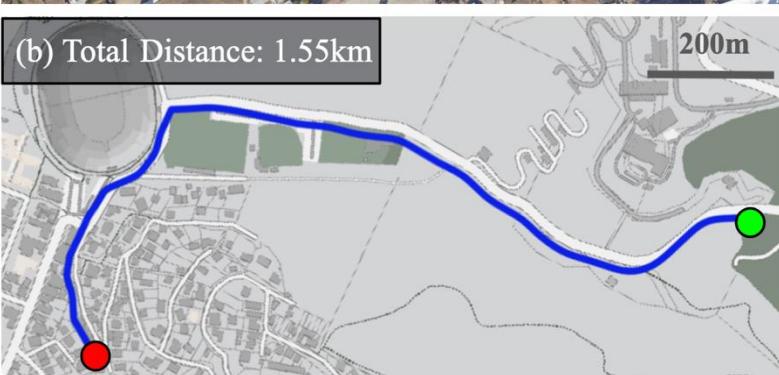
What Happens if we Disable Hints?



● Start

● Goal

~ Robot Path



● Start

~ Robot Path

Checkpoints



Summary

1. Goal-conditioned policies combined with a topological graph
2. Latent goal model to represent a reachability prior
3. Hints incorporated as a planning heuristic
 - a) Emergent preferences
 - b) Robustness to imperfect hints

Questions?

Detour: Beyond Goal-Reaching with RL

Incorporating preferences and rewards for customizing behaviors

Beyond Goal-Reaching



Stay on Sidewalk



Drive in the Sun

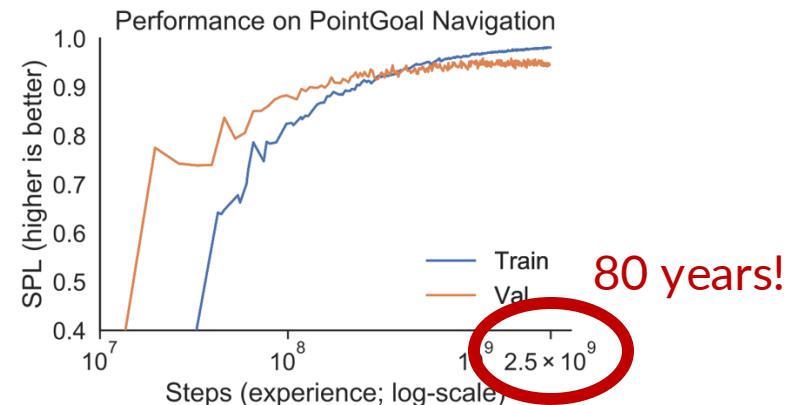
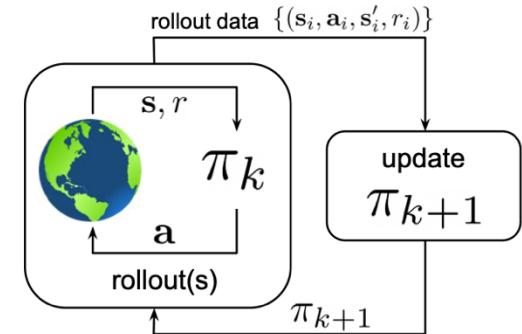


Stay off Grass

Reinforcement Learning

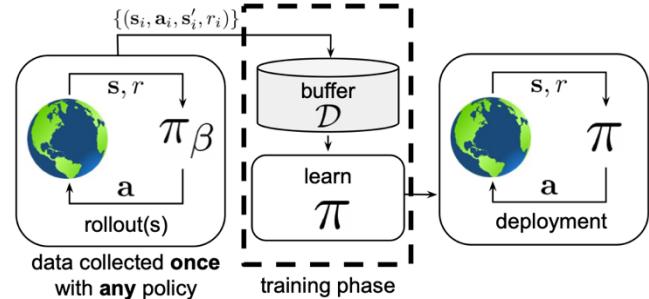
- Specify preferences as a reward function
- Collect rollouts in the environment, update policy
- But...
 - Online RL can be really expensive!

How can we do RL without being “online”?



Offline RL

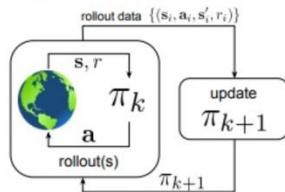
- Optimize policy with an *offline* dataset



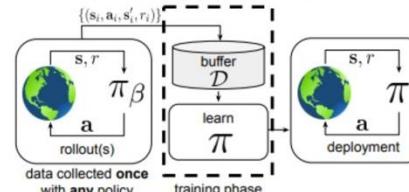
Why is this called offline **RL**?

What we typically think of as RL?
→ Trial and error online

What is offline RL?
→ **No trial and error online**
→ Sequential decision making
→ Similar solution tools to RL



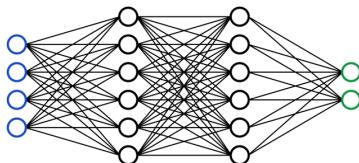
On-policy RL



Data-driven Offline RL

How can we learn decision making from non-optimal offline data?

(Offline) Values are Distances



$$d(\mathbf{s}, \mathbf{g})$$

$$\pi(\mathbf{a}|\mathbf{s}, \mathbf{g})$$

$$\gamma = 1$$

No Discounting

$$r(\mathbf{s}, \mathbf{a}, \mathbf{g}) = -1$$

Survival Penalty

$$V_{\pi}^*(\mathbf{s}, \mathbf{g}) = d^*(\mathbf{s}, \mathbf{g})$$

Theorem

For the optimal value function V^* for nearby goals and full coverage, finding the minimum-cost path between s, g in a graph with edge-weights $-V^*(s_i, s_j)$ recovers the optimal policy π^* that maximizes $V^*(s, g)$.

(Offline) Values are Distances

- Simple case: reduction to temporal distance

- $-V(s_t, g) = D(s_t, g)$

$$R_{\text{dist}}(s_t, a_t, g) = \begin{cases} -1 & \forall s_t \neq g \\ 0 & \text{otherwise.} \end{cases}$$

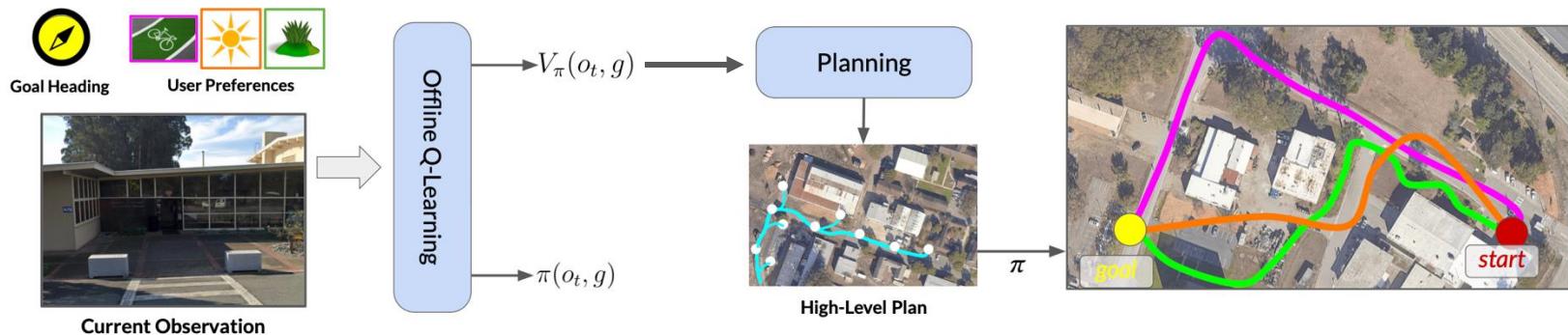
- “Solar robot” reward

- 4s in the sun \approx 1s in shade
- Detours to stay in the sun

$$R_{\text{sun}}(s_t, a_t, g) = \begin{cases} -1 + 0.75 * \mathbb{1}_{\text{sun}}\{s_t\} & \forall s_t \neq g \\ 0 & \text{otherwise.} \end{cases}$$

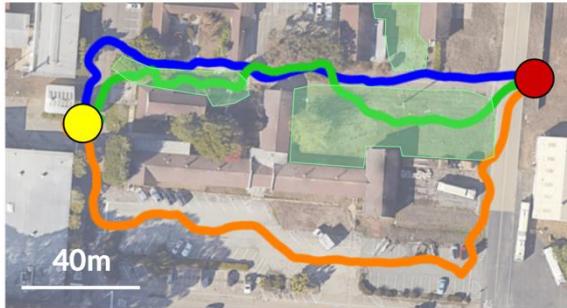
Offline Value Graphs

- Topological graph with (negative) values as edge weights



- Lemma: Under certain assumptions, min-cost search through the graph recovers the optimal policy π^*

 Start
 Goal
Agent Paths
 Min-Dist
 Grassy
 Sunny



- Distance Traveled: 182m
- $\mathbb{E}[\mathbb{I}_{grass}]$: 0.16
- $\mathbb{E}[\mathbb{I}_{sun}]$: 0.51

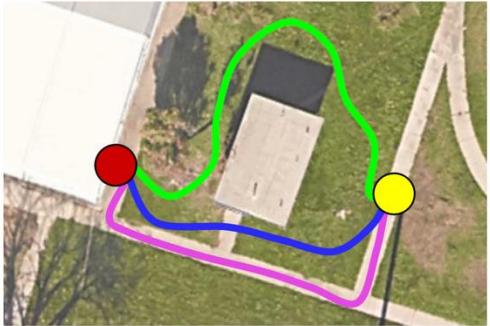


- Distance Traveled: 194m
- $\mathbb{E}[\mathbb{I}_{grass}]$: 0.87
- $\mathbb{E}[\mathbb{I}_{sun}]$: 0.37

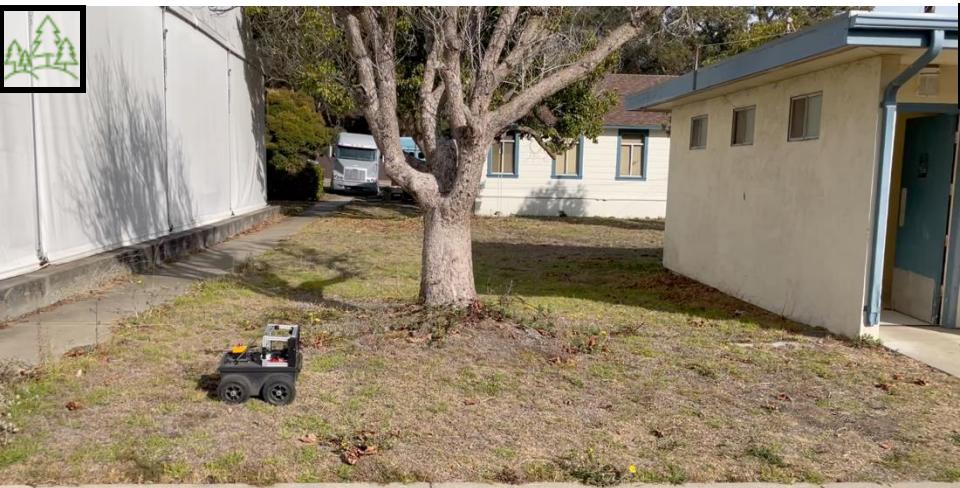


- Distance Traveled: 237m
- $\mathbb{E}[\mathbb{I}_{grass}]$: 0.05
- $\mathbb{E}[\mathbb{I}_{sun}]$: 0.78





Start
Goal
Agent Paths
Min-Dist
Grassy
Pavement



Questions?

Large-Scale Robot Learning II

Dhruv Shah

MLSS 2025

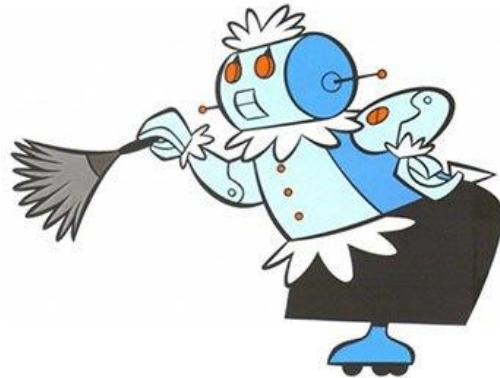
Google DeepMind



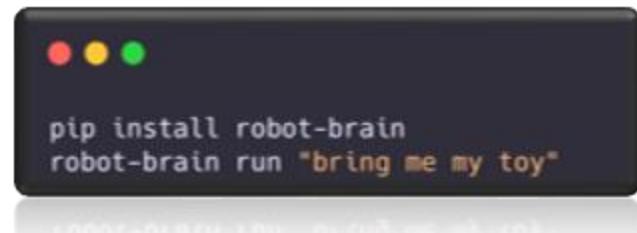
Course Overview

1. Setup a simple robotics problem: visual navigation
2. Design a machine learning method to solve it
3. Scaling this up to a “robot foundation model”
4. How can LLMs and VLMs (LMMs) help robotics

What we *really* want



Any robot

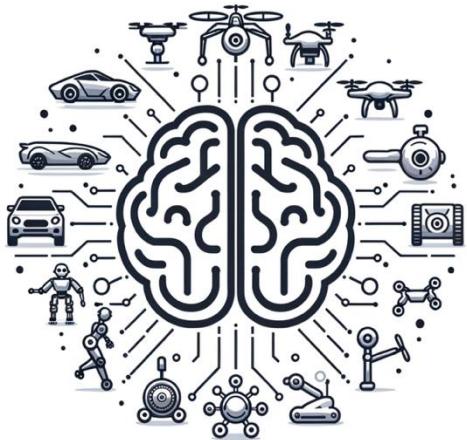


Download intelligent stack

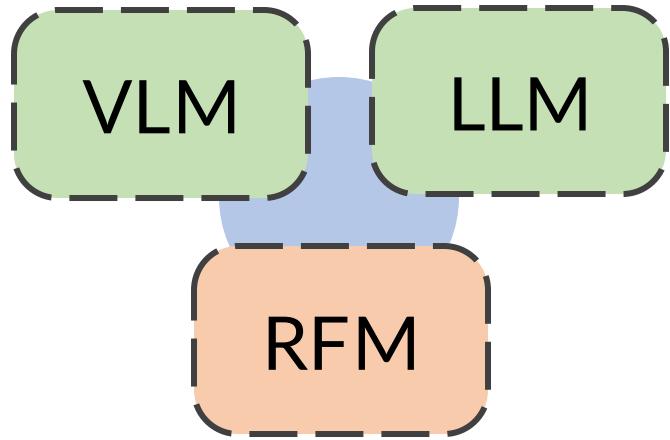


Plug-and-play

Foundation Models Of & For Robotics



Training a Cross-Embodiment
“Robot Foundation Model”



Empowering RFMs with
Internet Foundation Models

Part 3: Robot Foundation Models

What is a robot foundation model?

The Secret Sauce



Internet Foundation Models

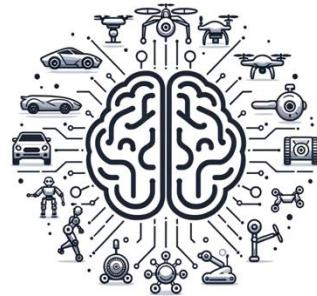
[K] “models trained on broad data (generally using self-supervision at scale) that can be adapted to a wide range of downstream tasks”

780B+ “To
[Singhal et al.]

How would you define a *robot foundation model*?

text pairs
al. 2023]

“Robot Foundation Model”?



A model trained with minimal supervision that can be

- (i) **deployed zero-shot** in completely different (useful) settings
 - *different sensors, robots, environments etc.*
- (ii) **adapted** to a downstream task of choice
 - *different objectives, goal specification, behavior etc.*



Open-Source

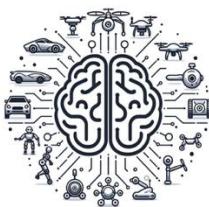


Any Robot



Lecture I Overview

1. Goal-conditioned policies combined with a topological graph
2. Latent goal model to represent a reachability prior
3. Hints incorporated as a planning heuristic
 - a) Emergent preferences
 - b) Robustness to imperfect hints



Section Overview

Pre-Training Task

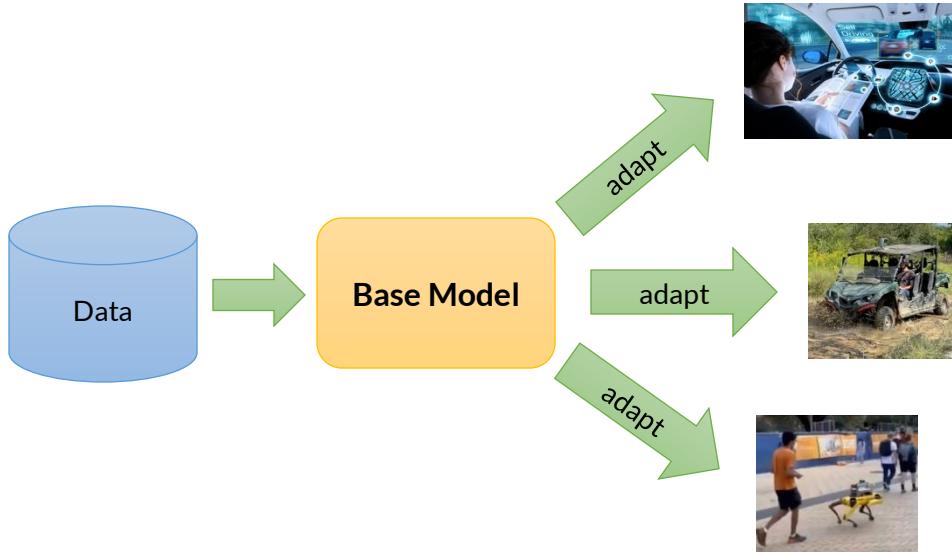
Model Architecture

Downstream Adaptation

What do we *really* want to learn?

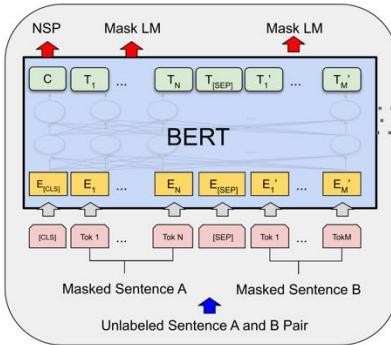
What objective can we use to learn general navigation skills from diverse data?

Without using additional supervision or labels

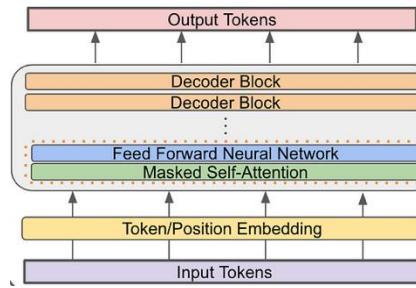


What works for Vision and Language

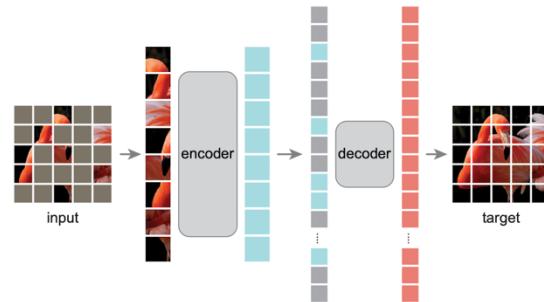
BERT: predict missing token



GPT: predict next token



MAE: predict missing image patches



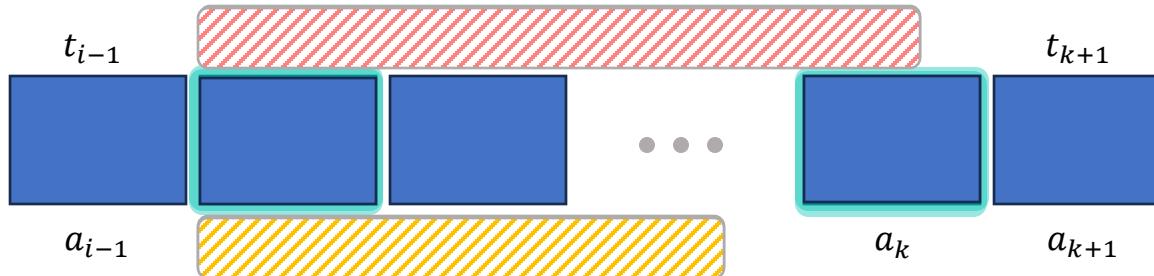
Fill in the Blanks: transferable representations without supervision

What does this objective look like for robot data?

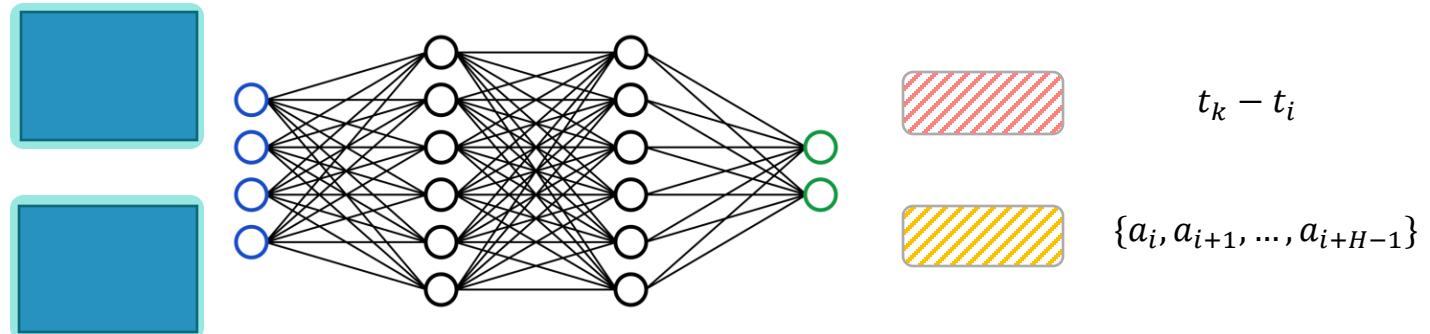
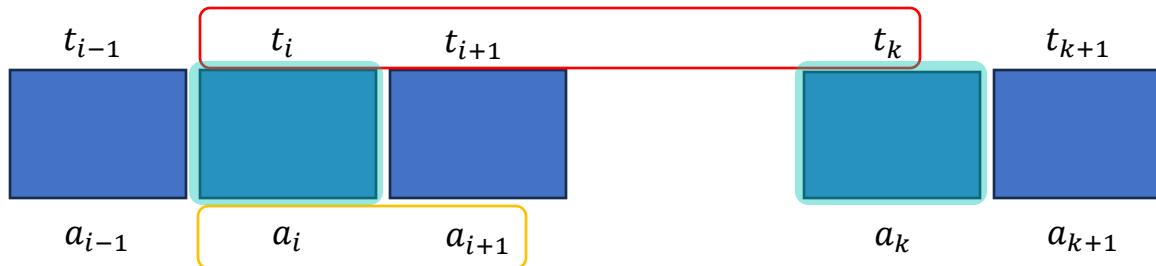


Key Idea: Masked Pre-training Objective

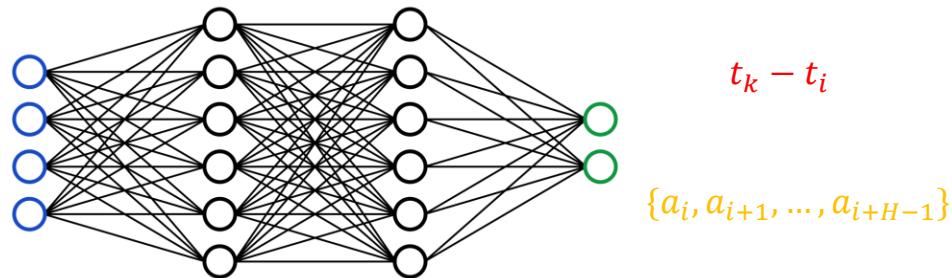
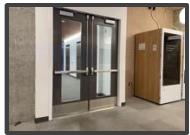
τ :



Key Idea: Masked Pre-training Objective

 $\tau:$ 

Learning to navigate from prior data

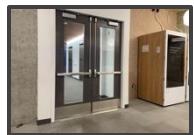
 $t_k - t_i$ $\{a_i, a_{i+1}, \dots, a_{i+H-1}\}$

Temporal distance to the goal

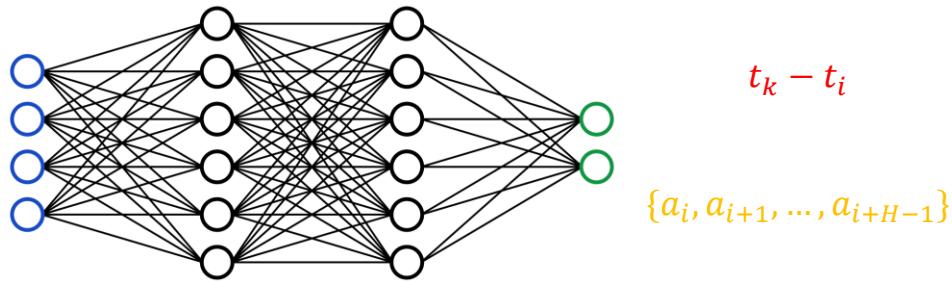
Best actions to take

Connection to LLMsMasking-based
Pre-Training Objective

Learning to navigate from prior data



Short-range Policy



Temporal distance to the goal

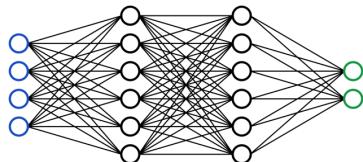
Best actions to take

Train with Goal-Conditioned Behavior Cloning

Suboptimal?

:)

RL Connections: Optimal Goal-Reaching



$$d(\mathbf{s}, \mathbf{g})$$

$$\pi(\mathbf{a}|\mathbf{s}, \mathbf{g})$$

$$\gamma = 1$$

No Discounting

$$r(\mathbf{s}, \mathbf{a}, \mathbf{g}) = -1$$

Survival Penalty

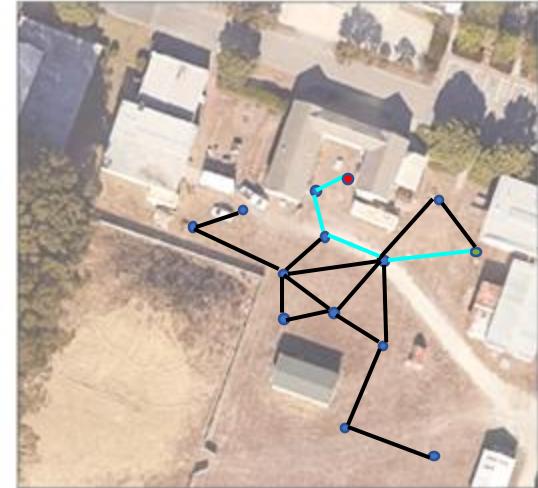
$$V_{\pi}^*(\mathbf{s}, \mathbf{g}) = d^*(\mathbf{s}, \mathbf{g})$$

Theorem

For the optimal value function V^* for nearby goals and full coverage, finding the minimum-cost path between s, g in a graph with edge-weights $-V^*(s_i, s_j)$ recovers the optimal policy π^* that maximizes $V^*(s, g)$.

Example Instantiation: Topological Graph

- Nodes are observations o_i
- Edges are estimated distances $d(o_i, o_j)$
- Goal-reaching ~ graph search



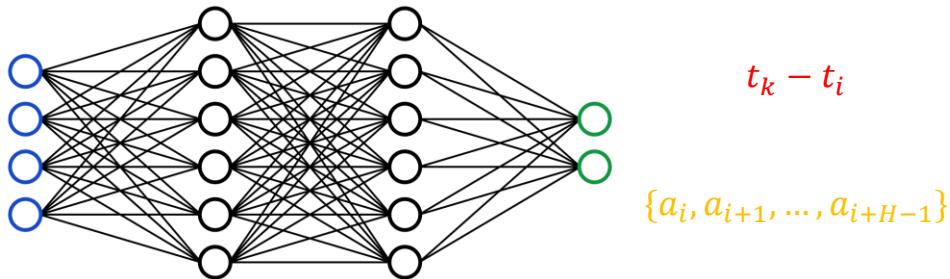
Theorem

For the optimal value function V^* for nearby goals and full coverage, finding the minimum-cost path between s, g in a graph with edge-weights $-V^*(s_i, s_j)$ recovers the optimal policy π^* that maximizes $V^*(s, g)$.

Learning to navigate from prior data



Short-range Policy



Temporal distance to the goal

Best actions to take

Train with Goal-Conditioned Behavior Cloning

Pair with memory/planner for long-horizon tasks

Cross-Embodiment Learning

Go1 Indoor Policy



Jackal Outdoor Policy



⋮

⋮

RACER Off-road Policy



Robot-specific End-to-End Policies

Trajectory Datasets

Cross-Embodiment Learning

Go1 Indoor Policy

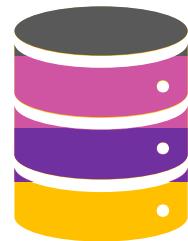
Jackal Outdoor Policy

⋮

RACER Off-road Policy



⋮

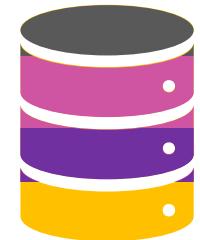
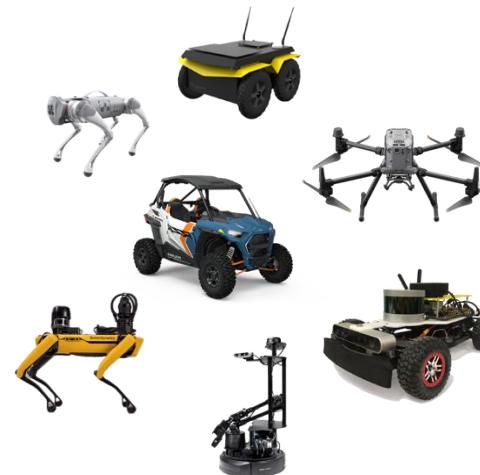


Curate Heterogeneous
Dataset

Cross-Embodiment Learning

RFM

Robot Foundation Model



Heterogeneous
Dataset

Shared pre-training task across all robots

Where will the data come from?



RC-Car
[Kahn et al. 2018]



TurtleBot
[Hirose et al. 2019]



Jackal
[Shah et al. 2021, 2022]



Spot
[Karnan et al. 2022]



Warthog
[Shaban et al. 2021]



ATV
[Triest et al. 2022]



Stanford
University



Berkeley
UNIVERSITY OF CALIFORNIA



Carnegie
Mellon
University



Georgia
Tech.



UNIVERSITY of WASHINGTON

Where will the data come from?



RC-Car
[Kahn et al. 2018]



Spot
[Karnan et al. 2022]



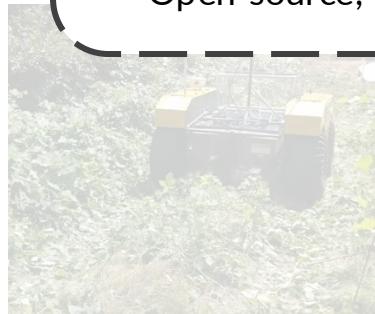
Warthog
[Shaban et al. 2021]



ATV
[Triest et al. 2022]

Artifact

- 150+ hours of visual navigation datasets
- 10 different robots, 8 research labs
- Open-source, MIT license



What does cross-embodiment training get us?

Robustness to robot variability and degradation

Zero-shot generalization capabilities

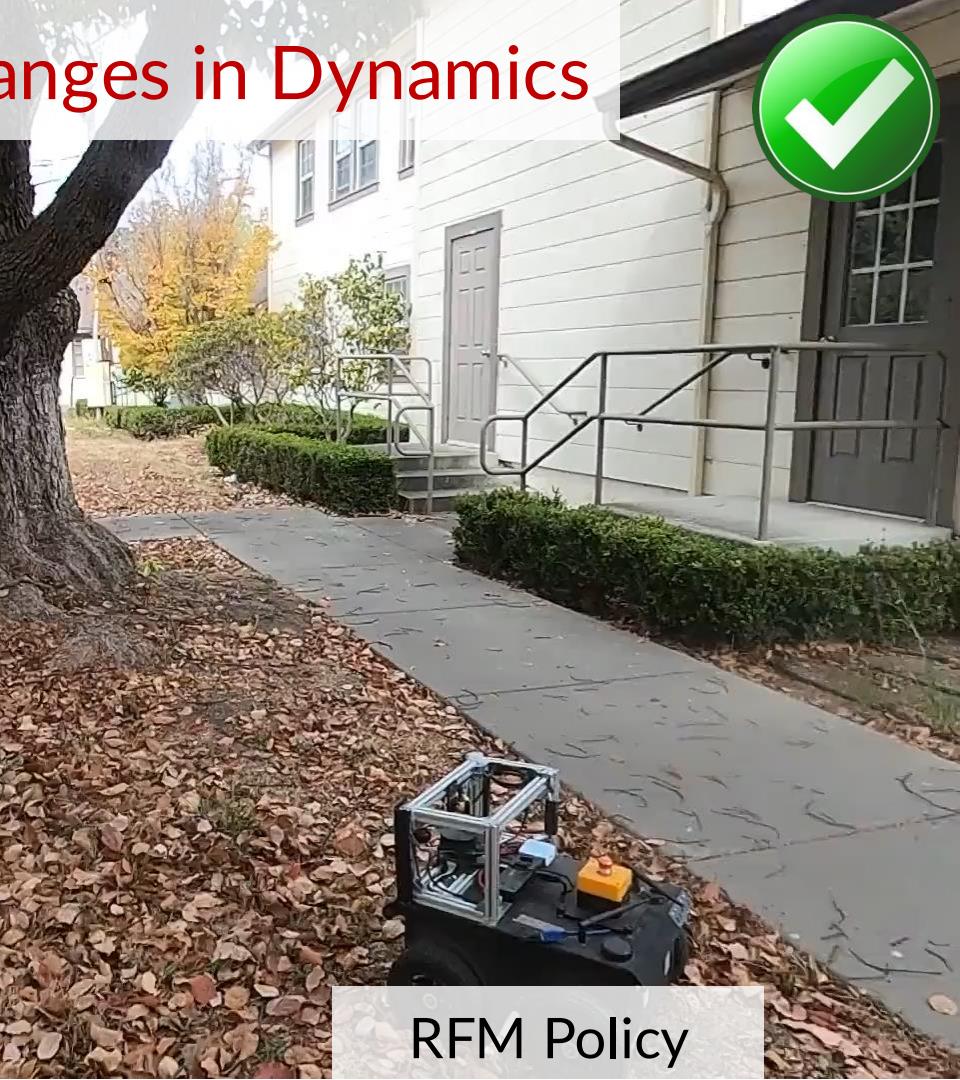
Robustness to robot variability



Robustness to Changes in Dynamics



Best Baseline



RFM Policy

Perturbing Viewpoints



Low

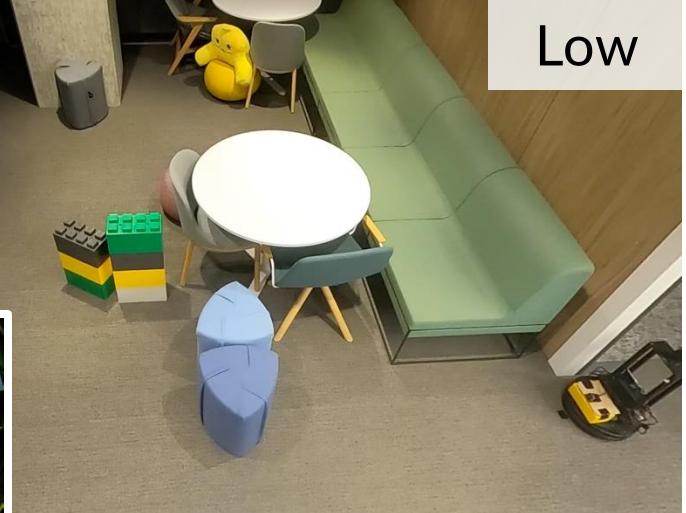


Mid

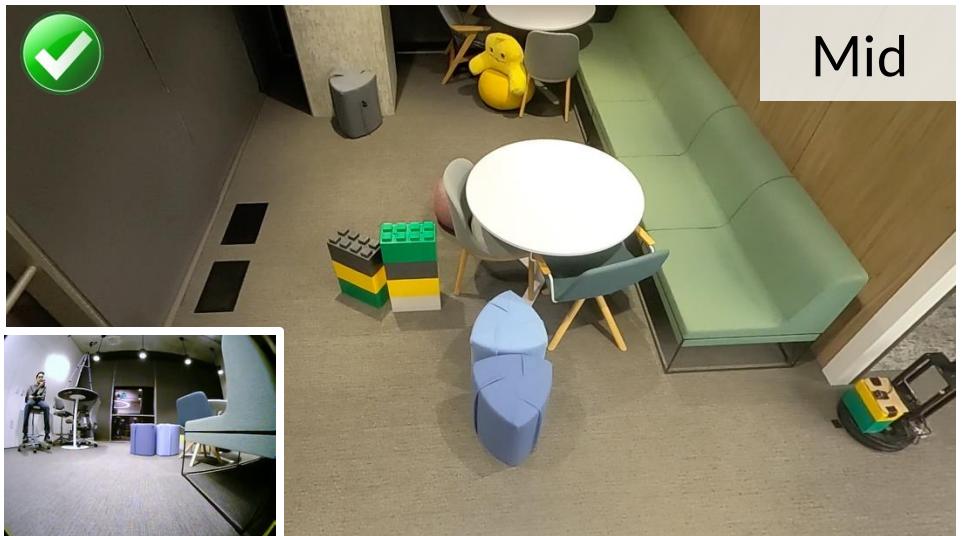


High

Goal Image



Low



Mid



High

Train on Ground Robots → Deploy on Drone

4x

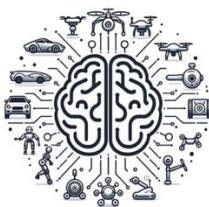


Current Observation



Goal Image





Section Overview

Pre-Training Task

Model Architecture

Downstream Adaptation

Model Desiderata



- Support robot heterogeneity
 - Condition on the robot's capabilities
 - Different action spaces
- High capacity and scalable
 - Real-time inference on edge hardware

RFM

Cross-Embodiment Learning

Go1 Indoor Policy



Jackal Outdoor Policy



RACER Off-road Policy



Robot-specific End-to-End Policies

Trajectory Datasets

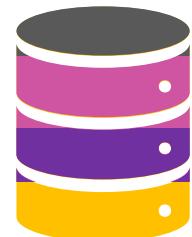
Cross-Embodiment Learning

Go1 Indoor Policy

Jackal Outdoor Policy

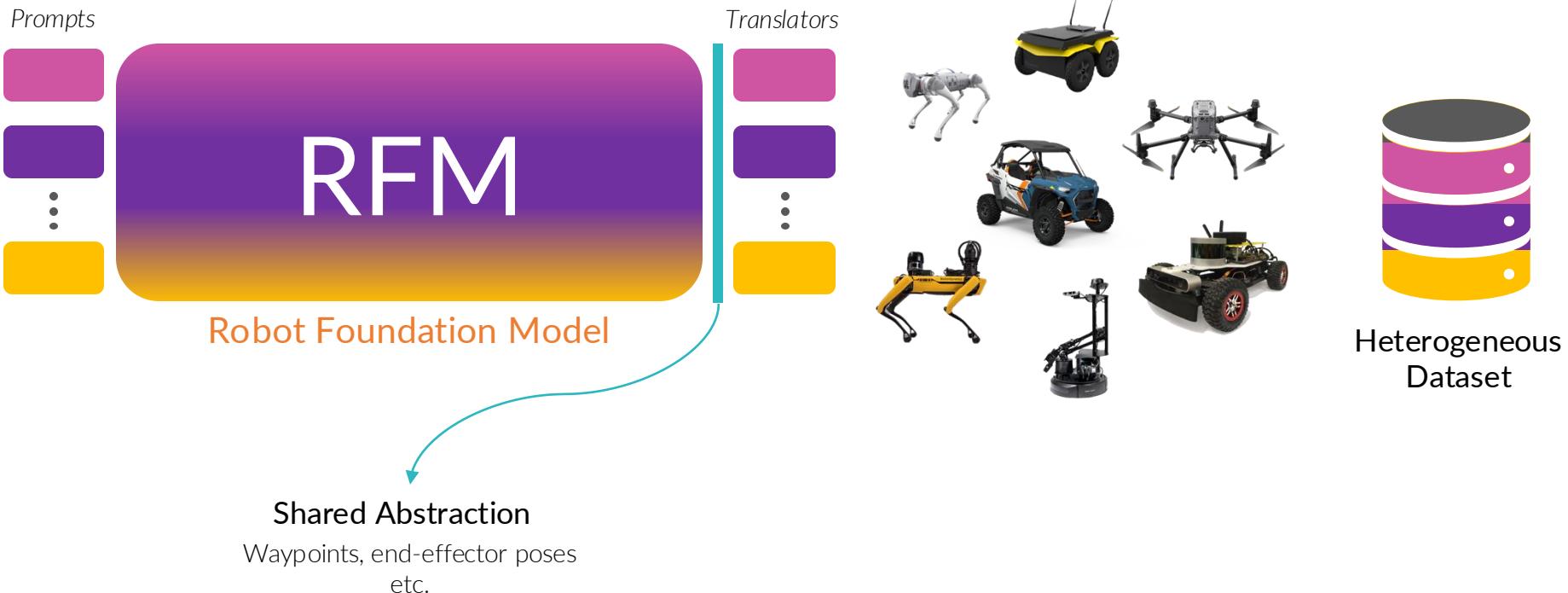
⋮

RACER Off-road Policy



Curate Heterogeneous
Dataset

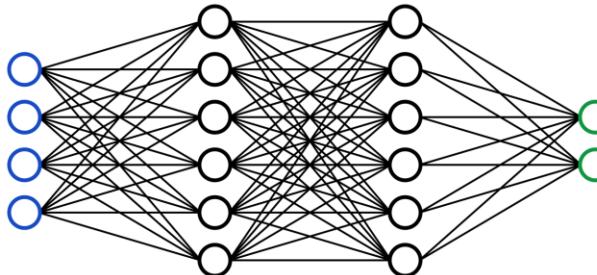
Cross-Embodiment Learning



Key Idea: Shared Action Abstraction



Short-range Policy



$t_k - t_i$

Temporal distance is scale invariant

$\{a_i, a_{i+1}, \dots, a_{i+H-1}\}$

Actions are robot-dependent

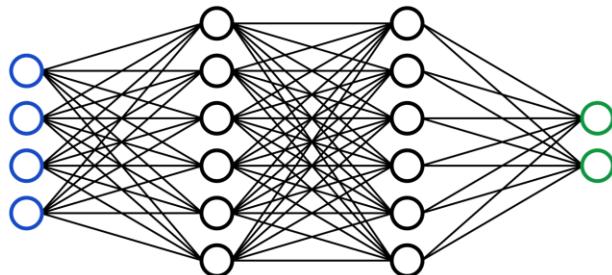
How do we change this so that the model can be shared across all robots?

Key Idea: Shared Action Abstraction

Change 1: Create a unified action abstraction across all robots



Short-range Policy



$t_k - t_i$

Temporal distance is scale invariant

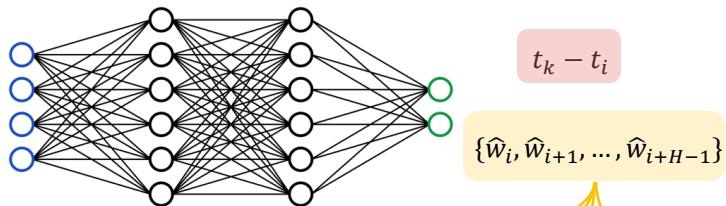
$\{\hat{w}_i, \hat{w}_{i+1}, \dots, \hat{w}_{i+H-1}\}$

Action space: normalized waypoints

Normalized units, scale invariant

Shared Action Abstraction: Translators

Change 2: Use robot-specific translators to *decode* native actions



Temporal distance is scale invariant

Action space: normalized waypoints



$\{a_2, \dots, a_{H+1}\}$



$\{a_2, \dots, a_{H+1}\}$



$\{a_2, \dots, a_{H+1}\}$



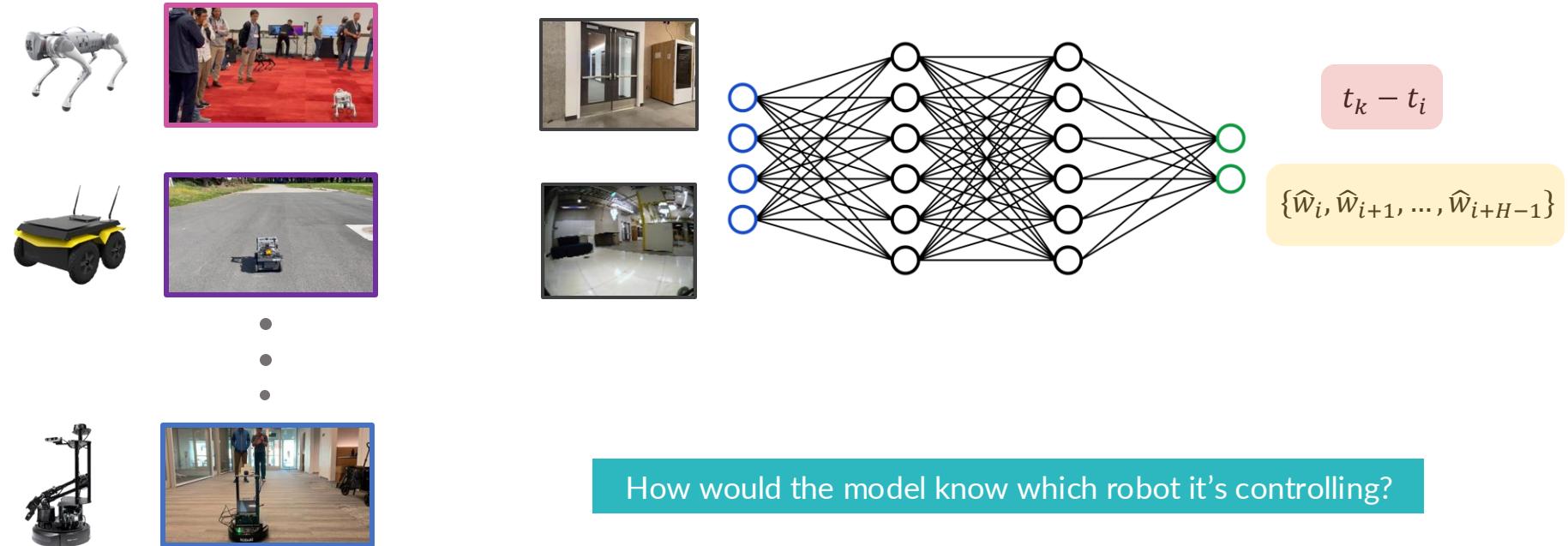
$\{a_2, \dots, a_{H+1}\}$



Robot-specific translators

classical/model-based or learned

Key Idea: Providing Embodiment Context



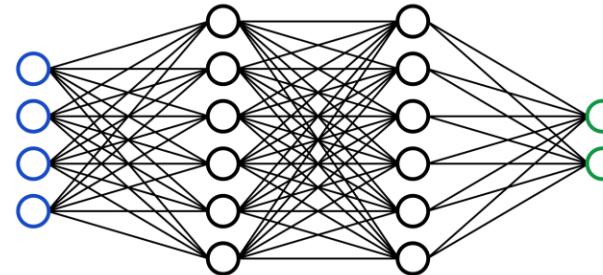
Providing Embodiment Context: Prompts

Option 1: Robot ID

Will this generalize to new robots?



Sys ID
A small color bar consisting of three colored squares: purple, yellow, and green.



$t_k - t_i$

Option 2: Sys ID

How do we parametrize robots?

Learning Many Related Tasks at the
Same Time With Backpropagation

Can we implicitly condition the model on the robot's identity/capabilities?

For known robots: same as Option 1

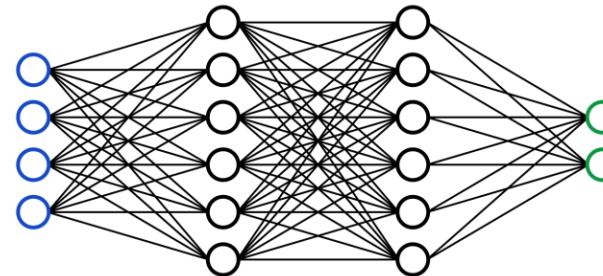
For new robots: same as Option 2, but learned similarity

Providing Embodiment Context: Prompts

Connection to LLMs

Equivalent to *prompting*
autoregressive LLMs

Embodiment Prompt



$t_k - t_i$

$\{\hat{w}_i, \hat{w}_{i+1}, \dots, \hat{w}_{i+H-1}\}$

Robot's observations may be indicative of its identity

Providing Embodiment Context: Prompts

Emergence of Implicit System Identification via Embodiment Randomization

Pranav Putta^{*1} Gunjan Aggarwal^{*1} Roozbeh Mottaghi² Dhruv Batra^{1,2}

Naoki Yokoyama¹ Joanne Truong¹ Arjun Majumdar¹

¹Georgia Institute of Technology ²FAIR, Meta AI *equal contribution

Embodiment P

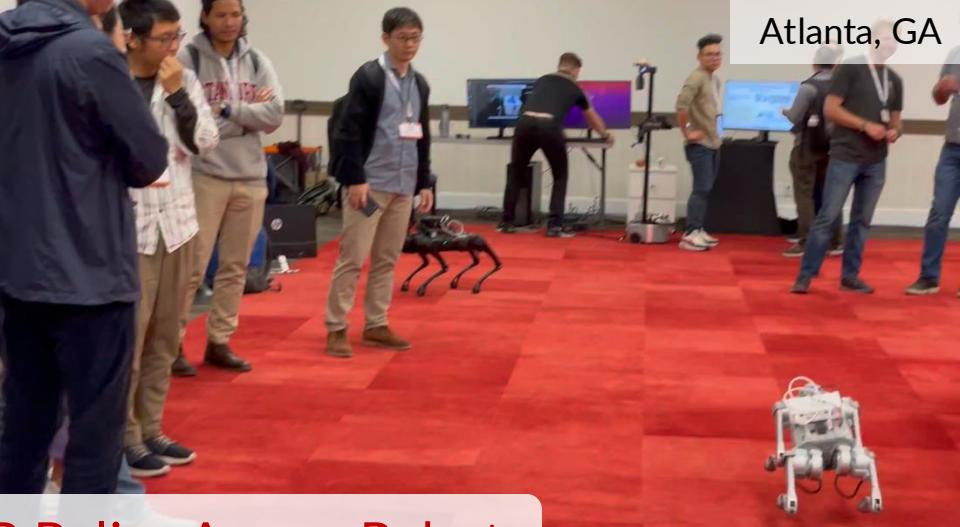
Evidence

Policies trained on multiple embodiments can *implicitly* learn to perform system identification.

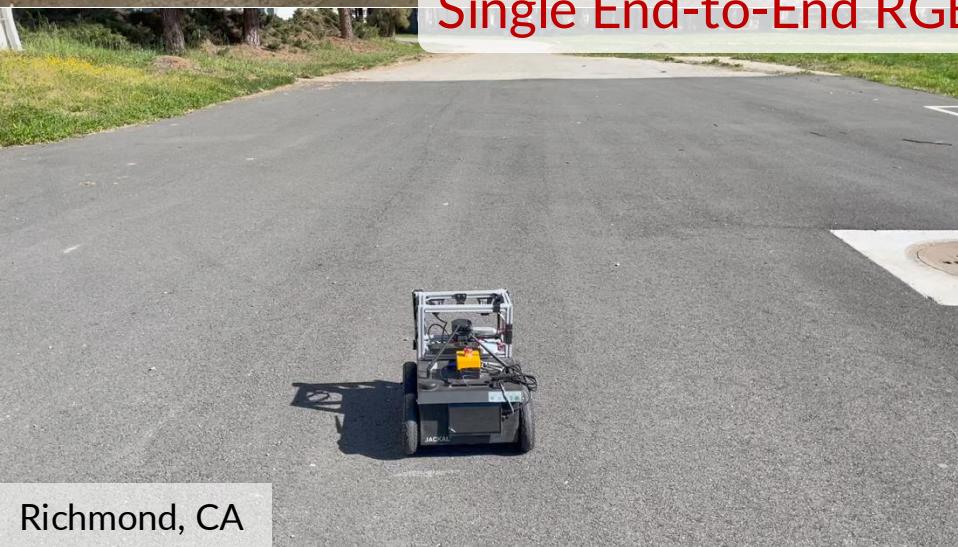
Berkeley, CA



Atlanta, GA



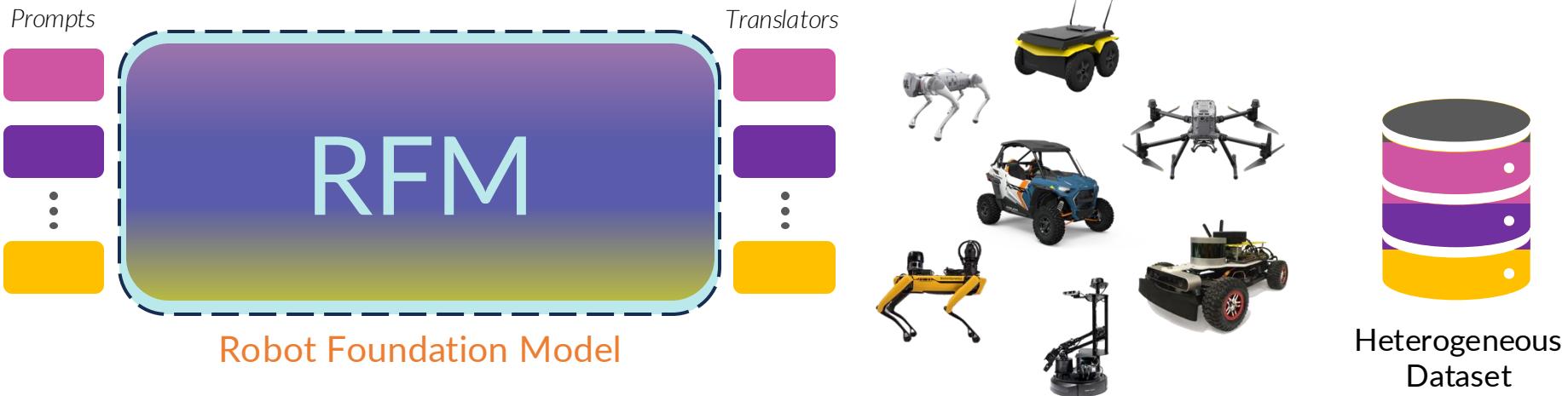
Single End-to-End RGB Policy Across Robots



Richmond, CA

Stanford, CA

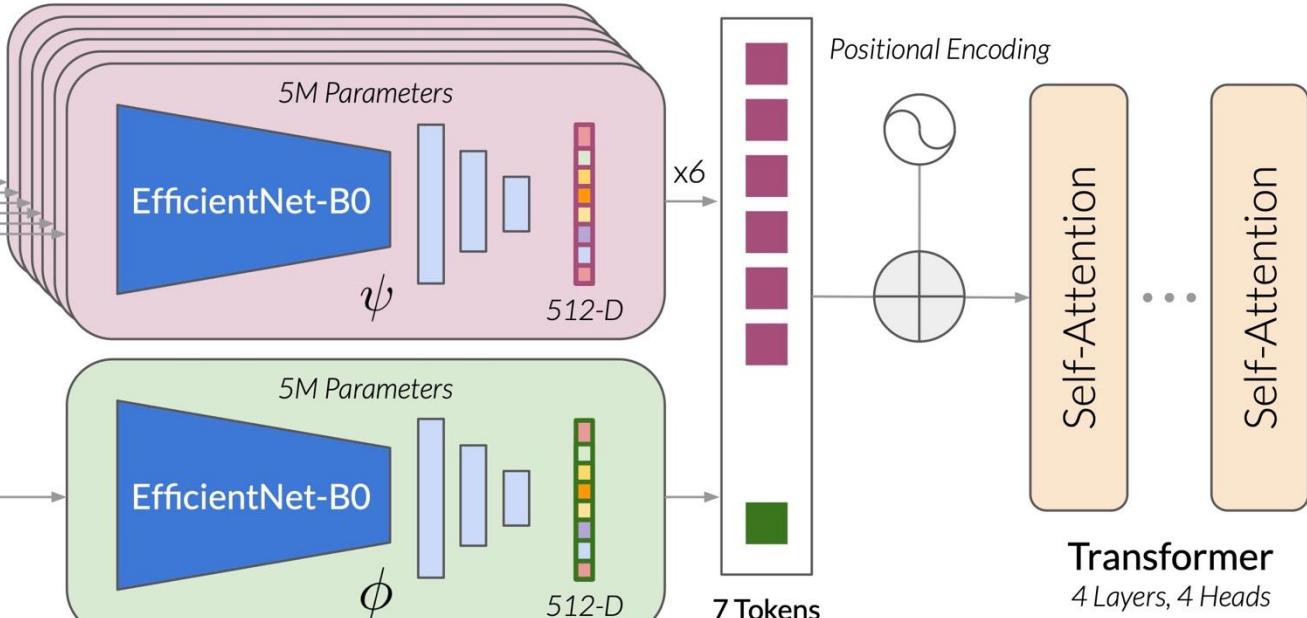
Roadmap: Cross-Embodiment Learning



(Includes Embodiment Prompt)

Observations

Past 5 timesteps
85x64x3 RGB

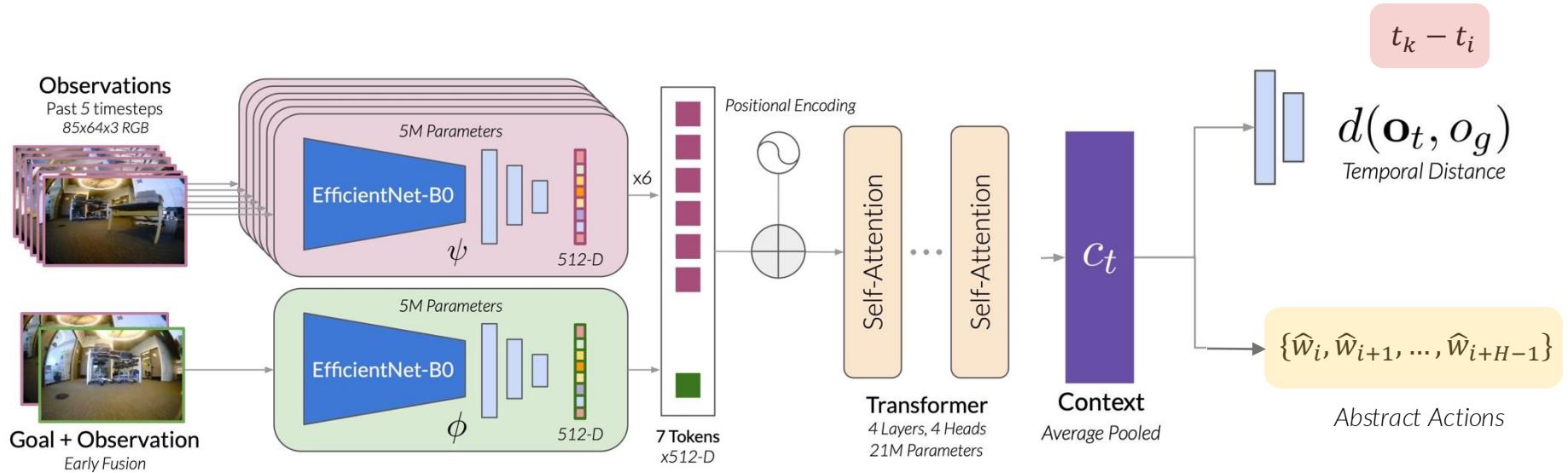
Goal + Observation
Early Fusion

CNN Encoders

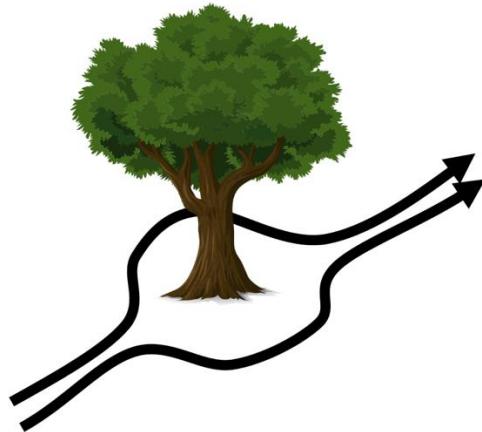
Easy to scale up to 60M each

Attention Block

Easy to scale beyond 1B



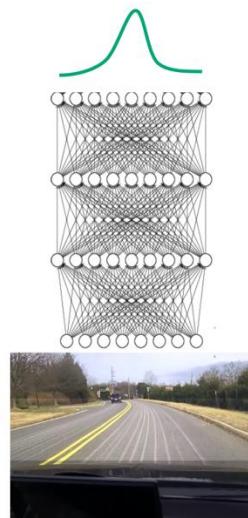
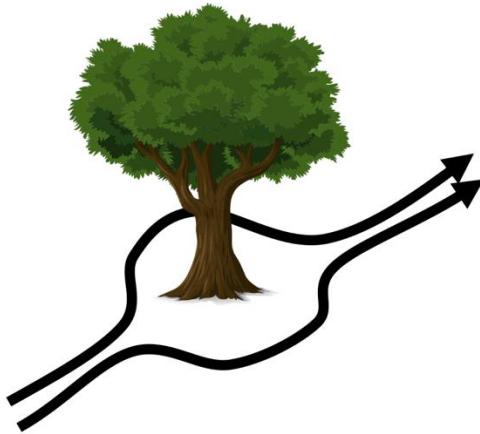
Possible flaw?



How do we represent multimodal action distributions?

Using Richer Policy Distribution Classes

Multimodal behavior → use more expressive probability distributions, no mode averaging issues



1. Output mixture of Gaussians
2. Latent variable models
3. Autoregressive discretization
4. Diffusion models
5. ...



How do diffusion models work?

Denoising diffusion models

● Forward / noising process



● Reverse / denoising process

- Sample noise $p_T(x_T) \rightarrow$ turn into data

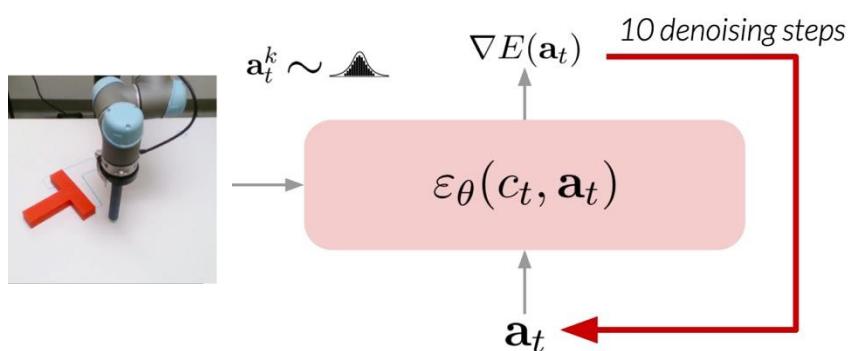
Algorithm 1 Training

```
1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
         $\nabla_{\theta} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, t)\|^2$ 
6: until converged
```

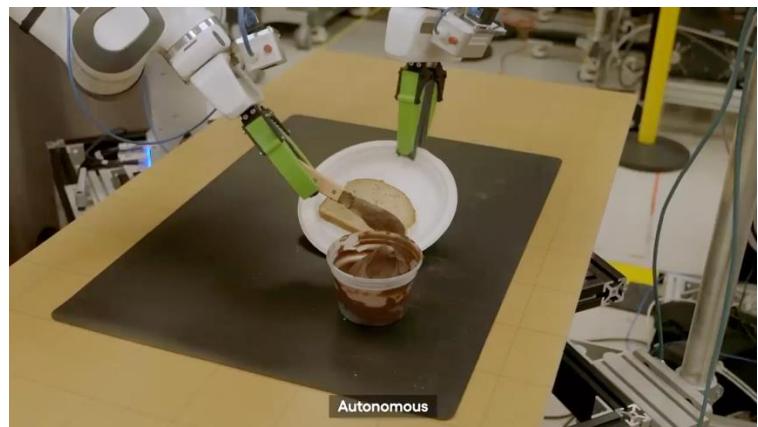
Algorithm 2 Sampling

```
1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  if  $t > 1$ , else  $\mathbf{z} = \mathbf{0}$ 
4:    $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$ 
5: end for
6: return  $\mathbf{x}_0$ 
```

State-conditional diffusion models → actions



Diffusion Policy



Single-task, data-efficient policies



“General” multi-task, multi-robot policies?

Chi et al., Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. RSS 2023

Reuss et al., Goal-Conditioned Imitation Learning using Score-based Diffusion Policies. RSS 2023

Sridhar, Shah et al., NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration. ICRA 2024 (Best Paper Award)

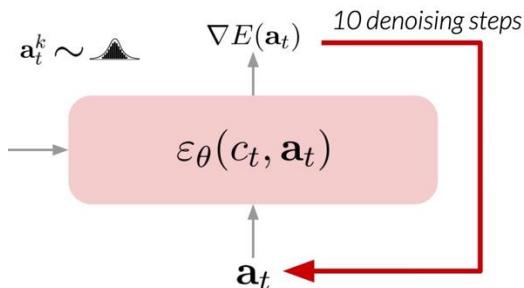
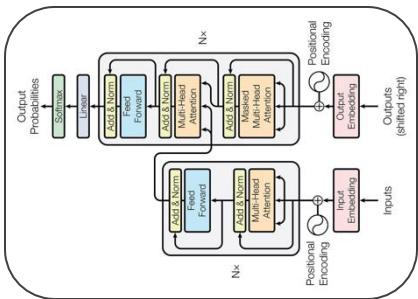
Scaling up Diffusion Policy

Key Insights

1. DP over-optimized for data efficiency
2. Goal-conditioning helps generalization and positive transfer
3. Richer visual representations help OOD robustness



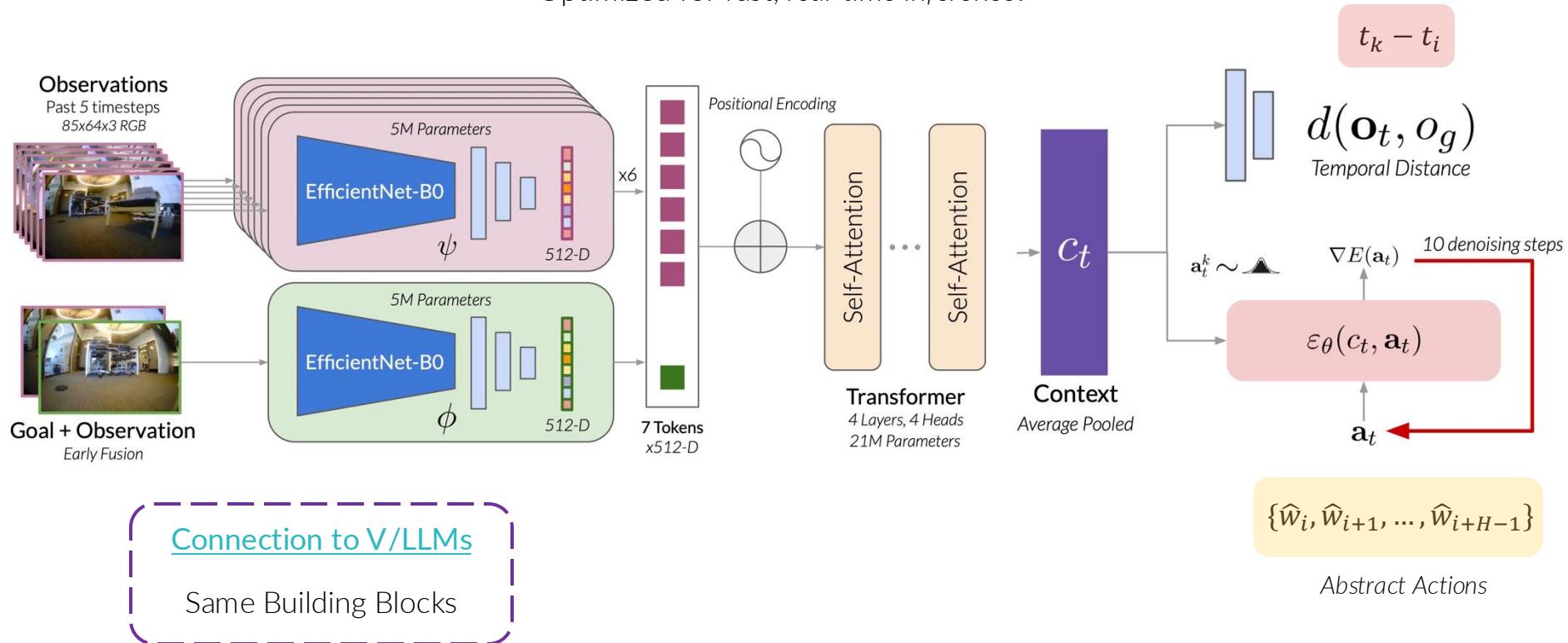
Goal



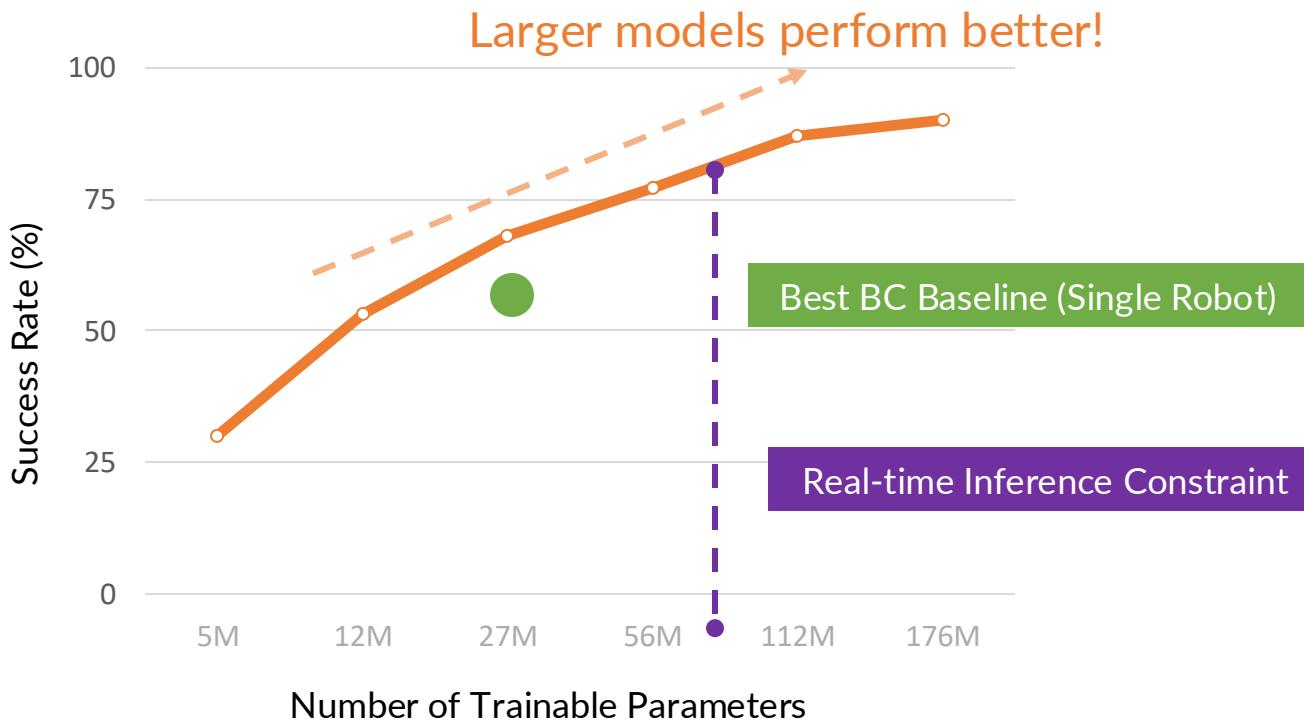
Diffusion Policy scales to cross-embodiment training!

Scalable Architecture: 5M to 250M+ Parameters

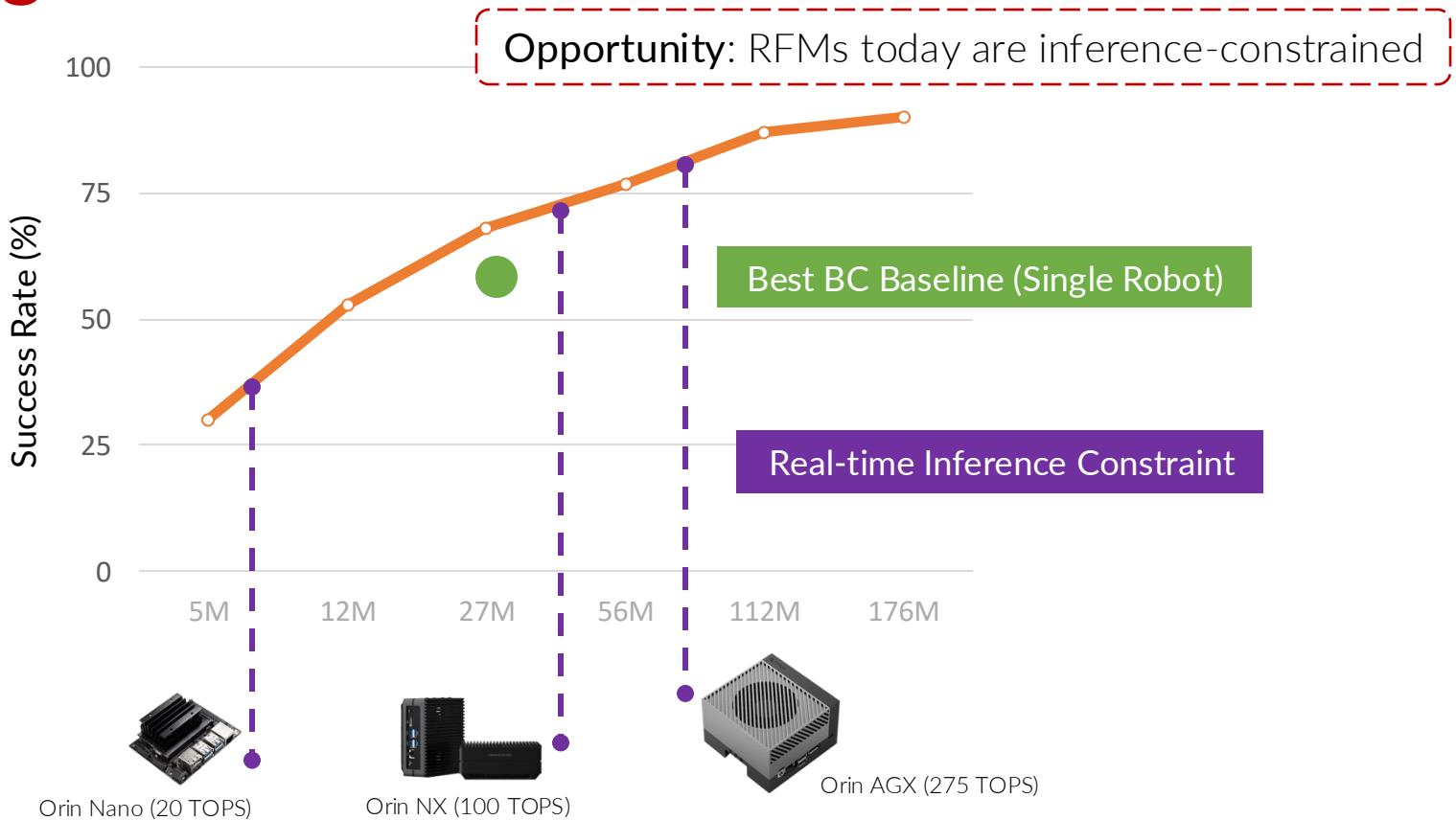
Optimized for fast, *real-time inference!*



Scaling Trade-offs

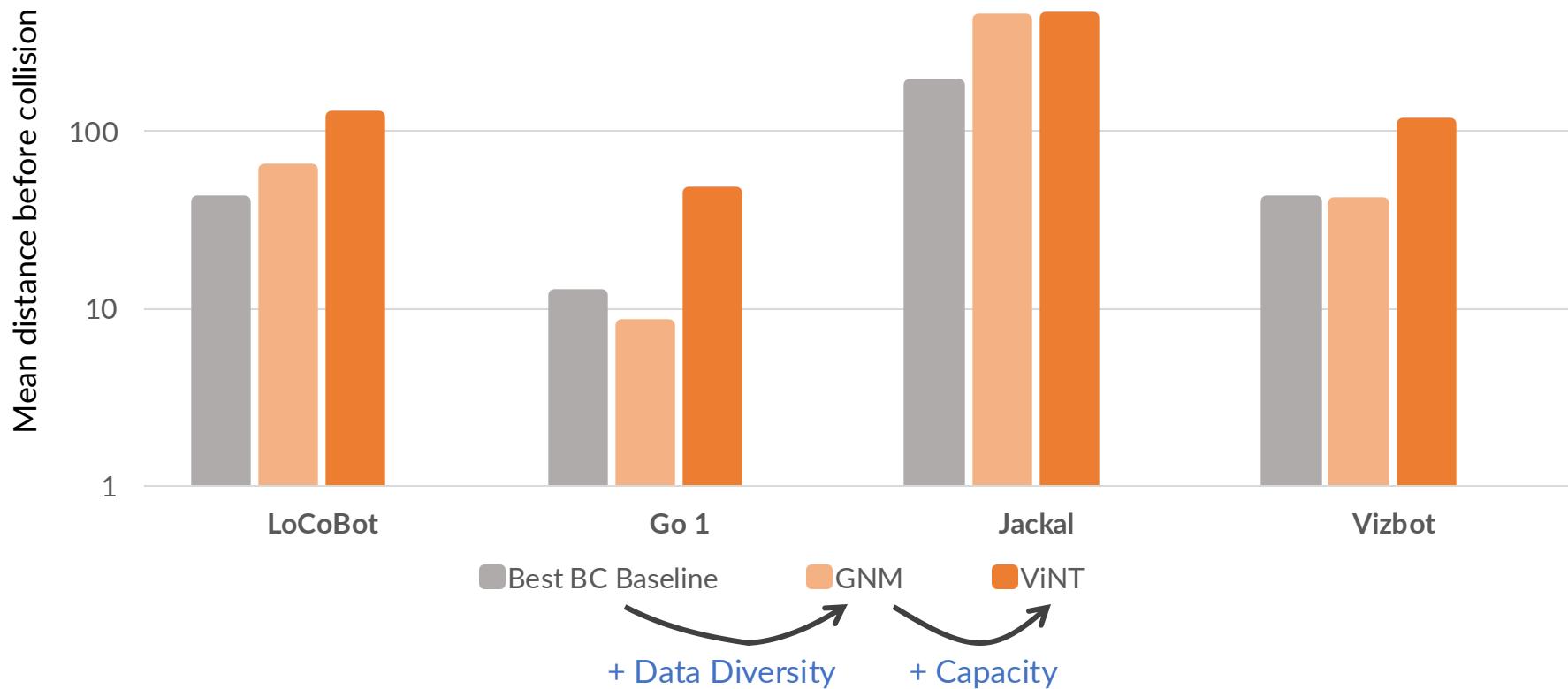


Scaling Trade-offs



Co-training helps positive transfer

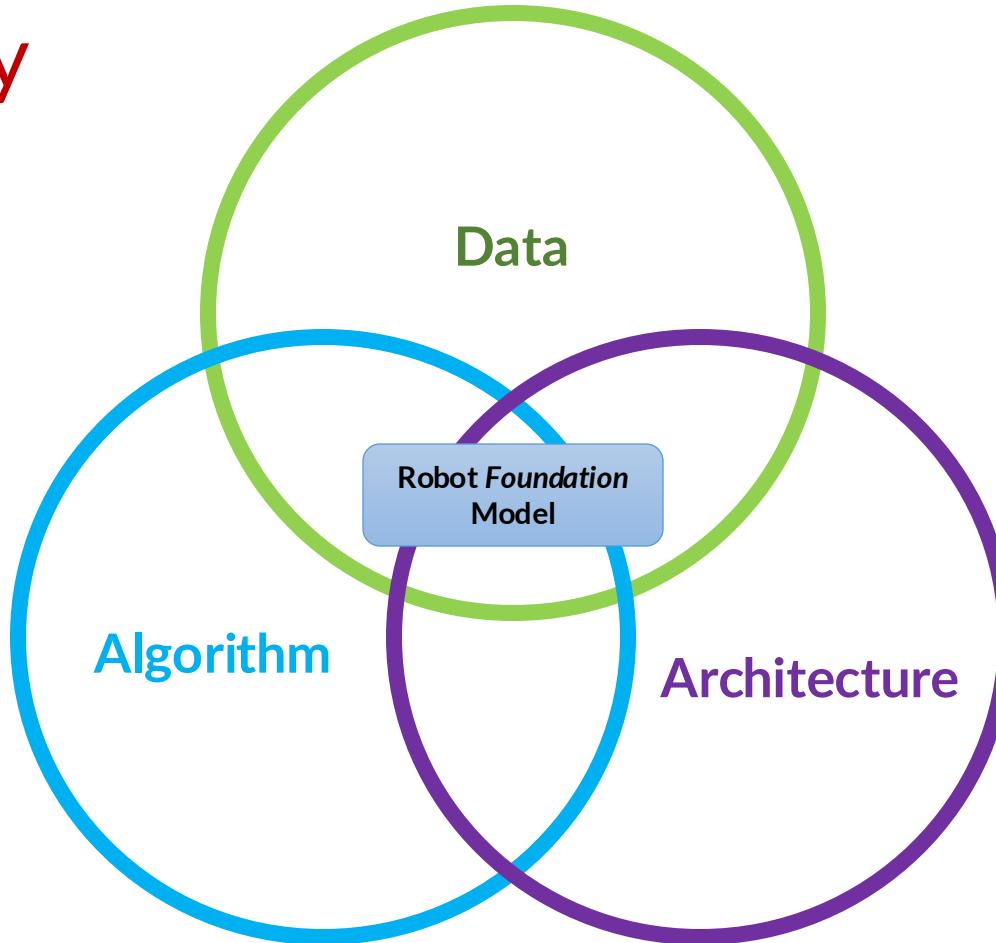
Confirms the cross-embodiment hypothesis!

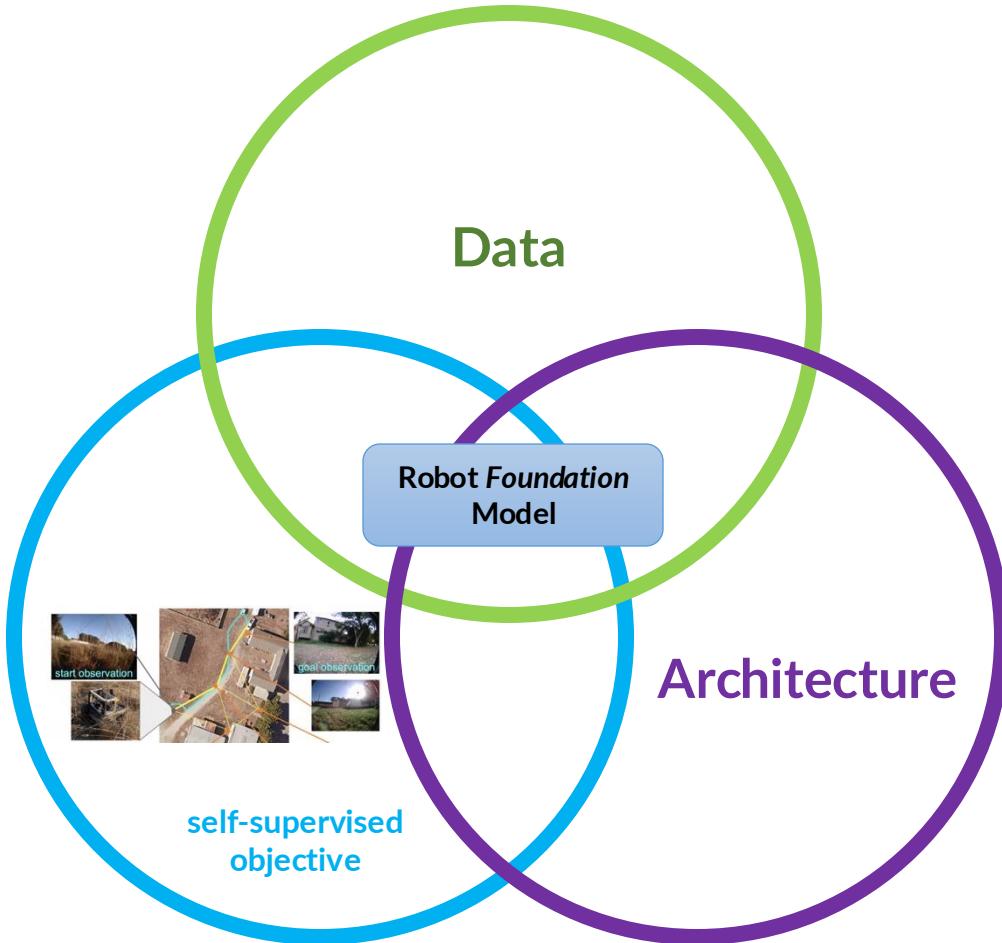


Zero-shot Locomotion from RGB Observations (no depth)
Live Demo @ CoRL 2023 (Atlanta, GA)



Summary





Prediction targets:

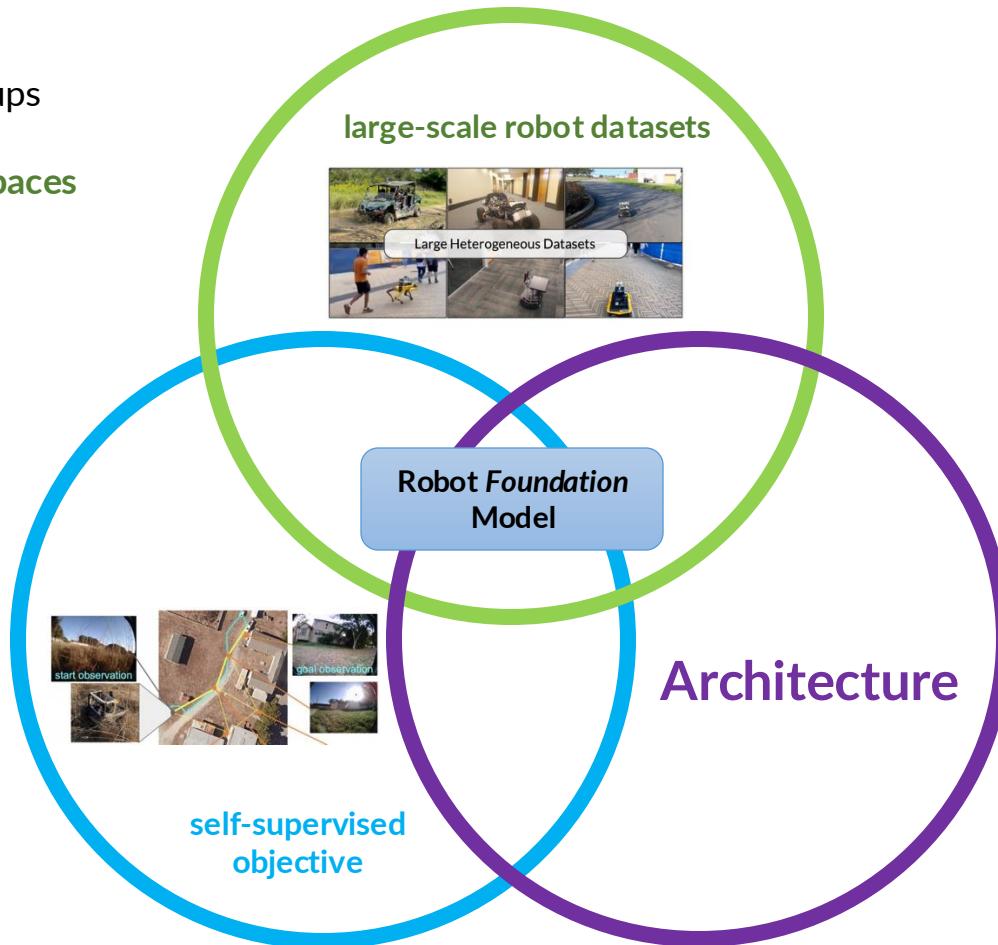
- Temporal distance between 2 frames
- Actions taken in the dataset (*offline*)

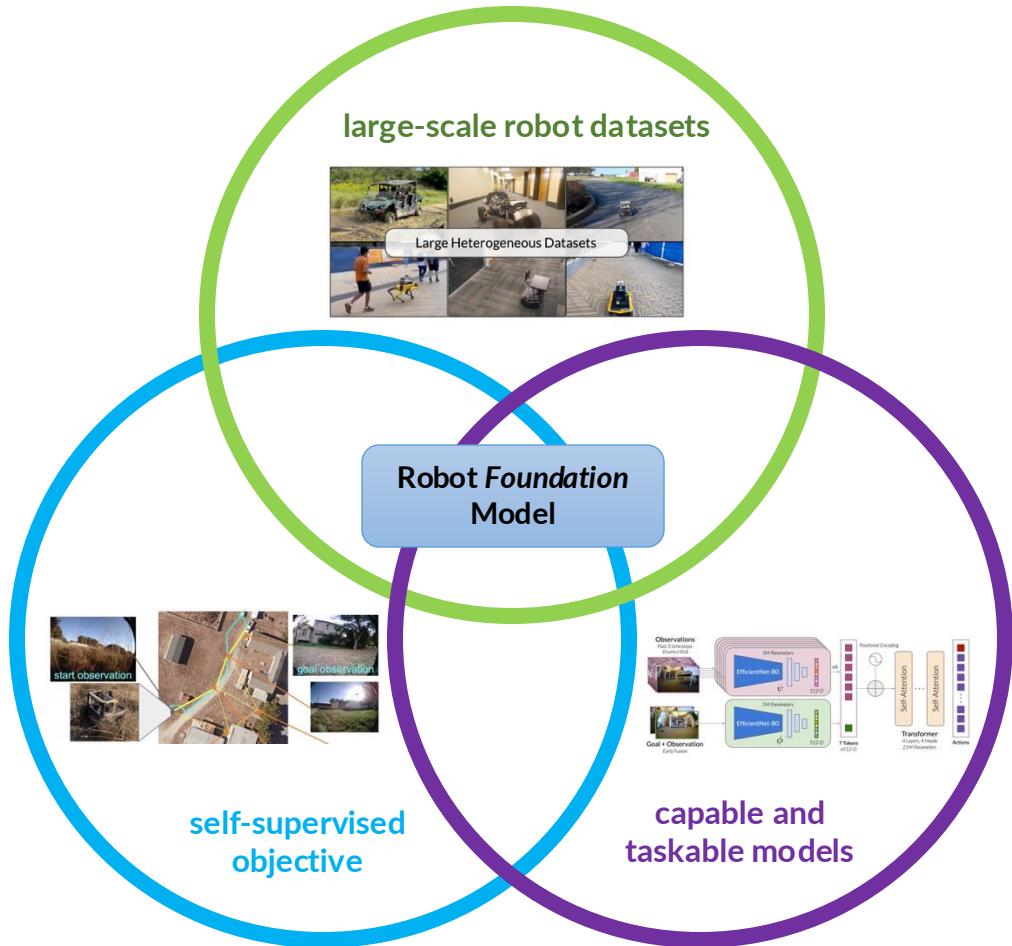
Objective: Goal-conditioned BC

Cross-embodiment data:

Sourced across 10+ robots and research groups

Different sizes, speeds, capabilities, action spaces



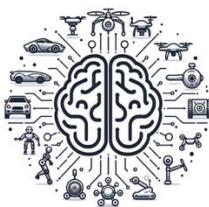


Transformer backbone:

- Observations and goals encoded as *tokens*, temporal history

Diffusion over actions:

- Expressing **multimodal** actions



Section Overview

Pre-Training Task

Model Architecture

Downstream Adaptation

Adapting Robot Foundation Models

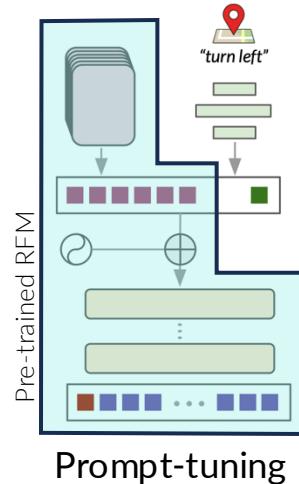
New Objectives



Reward Fine-tuning
Socially-Compliant Navigation

Hirose, Shah et al., SACSoN. RA-L 2023

New Modalities



Shah et al., ViNT. CoRL 2023
Sridhar, Shah et al., NoMaD. ICRA 2024

New Robots & Tasks



Visual Representation Learning
Autonomous Off-Road Racing

Shah et al., FastRLAP. CoRL 2023

Socially Compliant Navigation via Reward Fine-Tuning



Key Idea

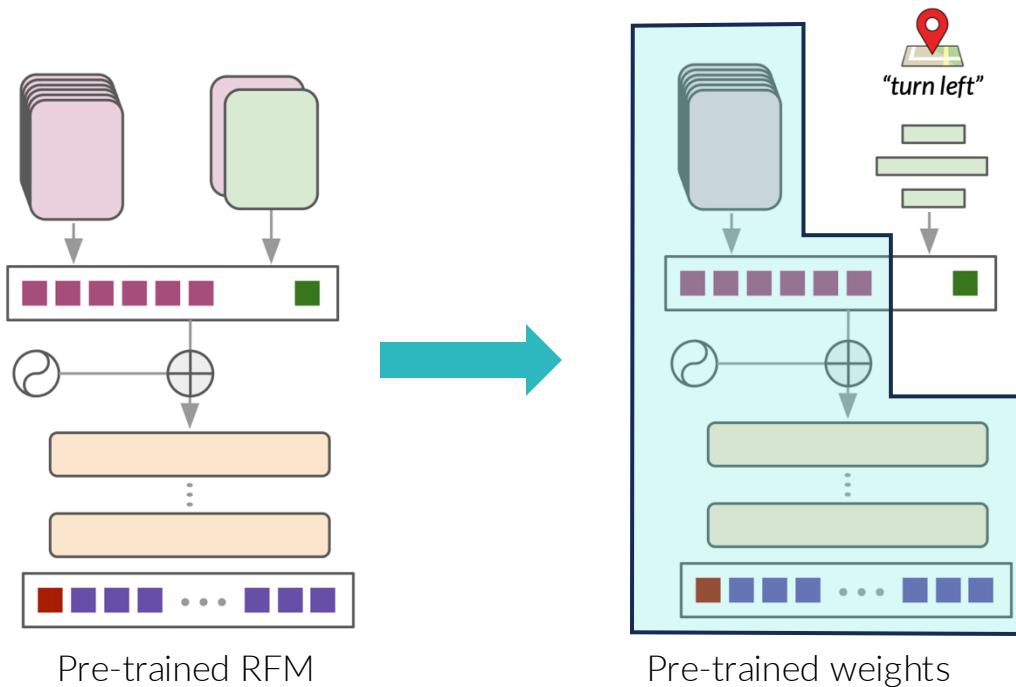
Fine-tune the RFM with a reward model to improve social compliance.

Train a separate pedestrian trajectory forecasting model and use it as a reward model to iteratively optimize the RFM to reach goals while minimizing human disturbance.

Connection to LLMs

Resembles SFT/RLHF procedures for alignment of LLMs.

New modalities by prompt-tuning RFMs



Key Idea

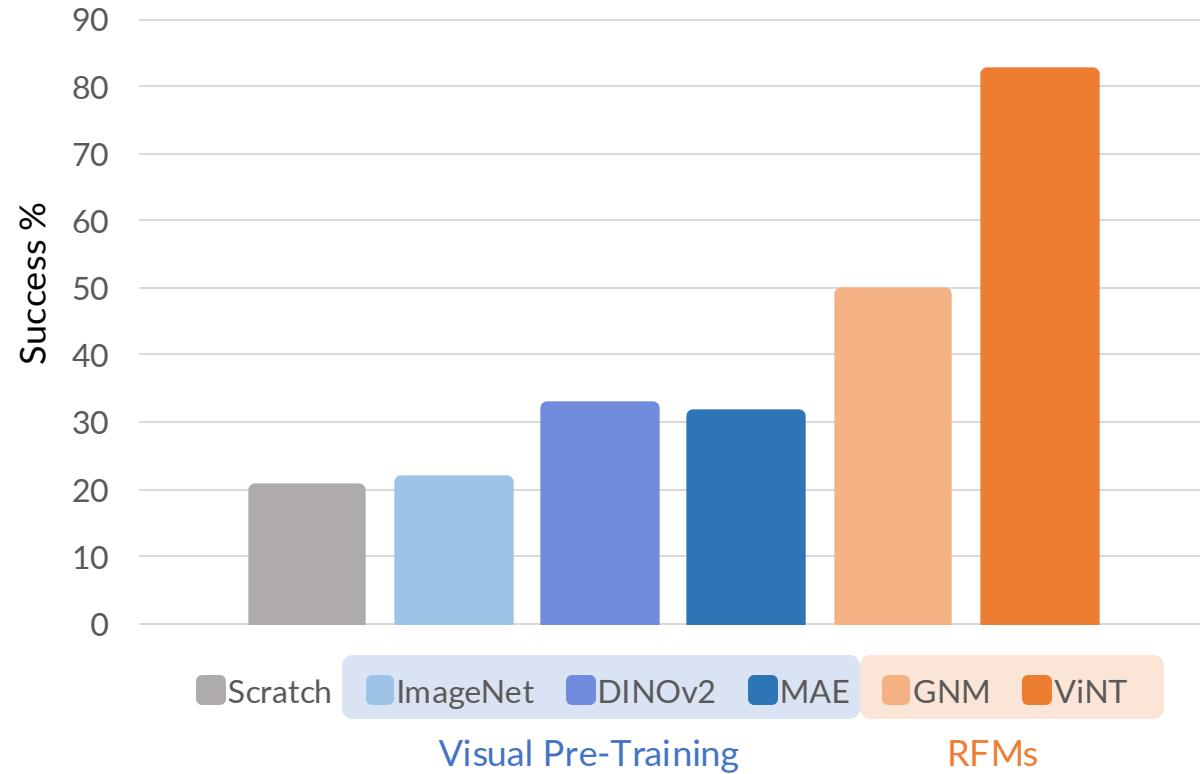
RFMs can be efficiently adapted to reach different types of goals

Swap the goal encoder to project new modalities (e.g., text commands, GPS) into the token space, while retaining collision avoidance and other pre-trained behaviors.

Connection to LLMs

Analogous to prompt-tuning / soft-prompting in LLMs

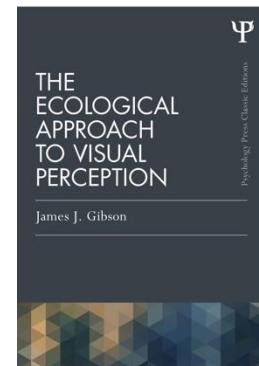
RFMs learn better visual features



Key Idea

RFMs learn better representations for downstream transfer.

We quantify the amount of downstream transfer of representations (CNN encoder) trained using the best self-supervised **vision objectives** and the **RFM objective**.



Adapting Robot Foundation Models

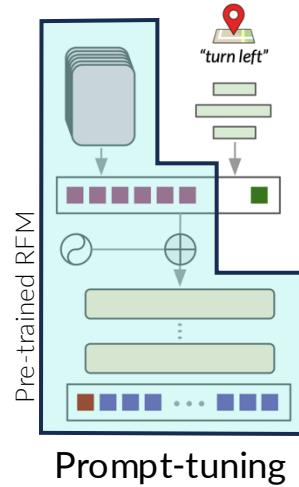
New Objectives



Reward Fine-tuning

Hirose, Shah et al., SACSoN. RA-L 2023

New Modalities



Shah et al., ViNT. CoRL 2023
Sridhar, Shah et al., NoMaD. ICRA 2024

New Robots & Tasks



Visual Representation Learning

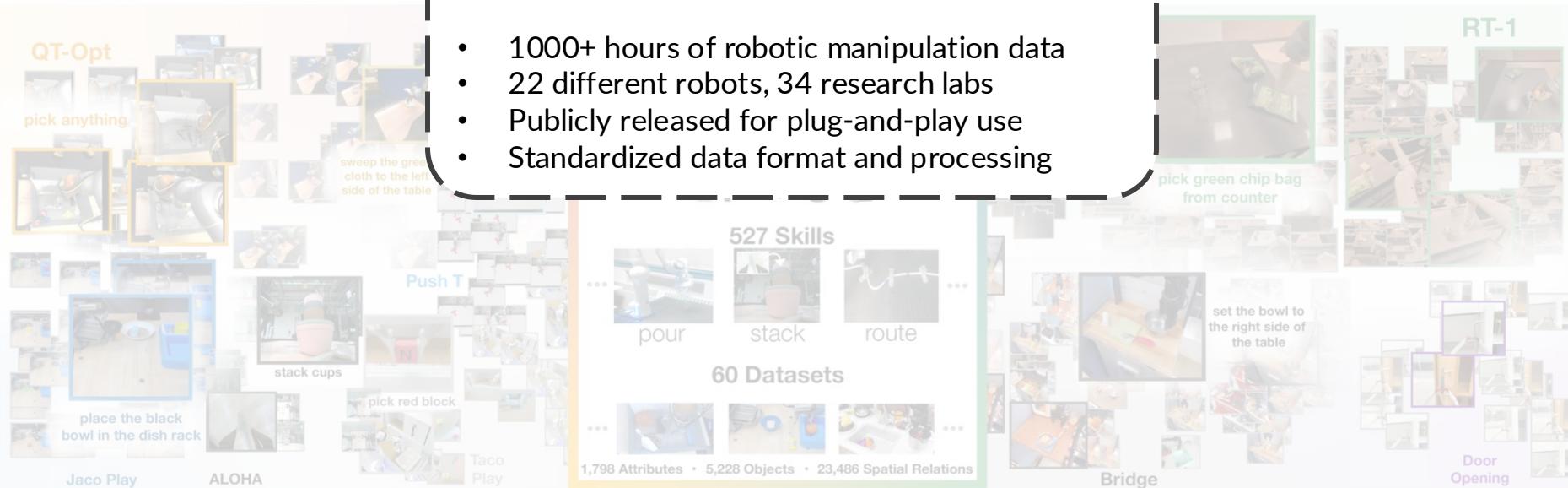
Shah et al., FastRLAP. CoRL 2023



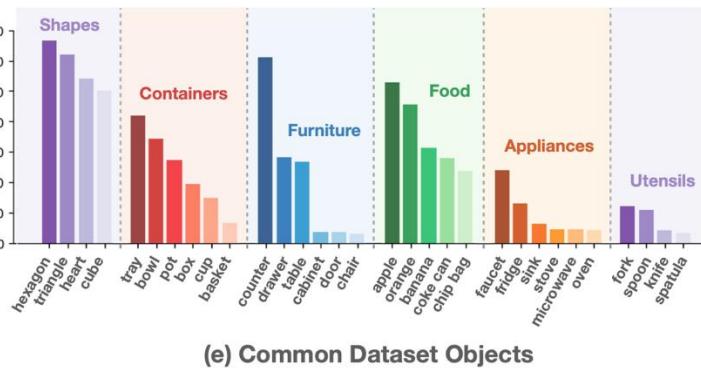
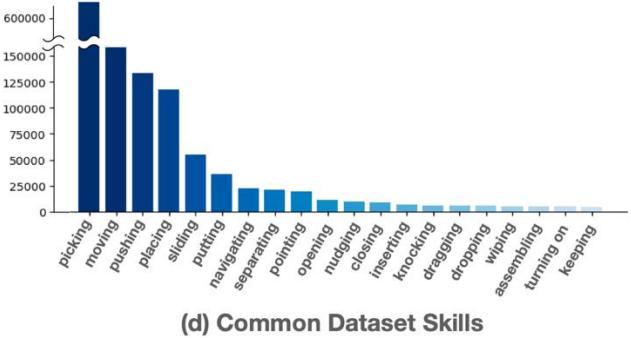
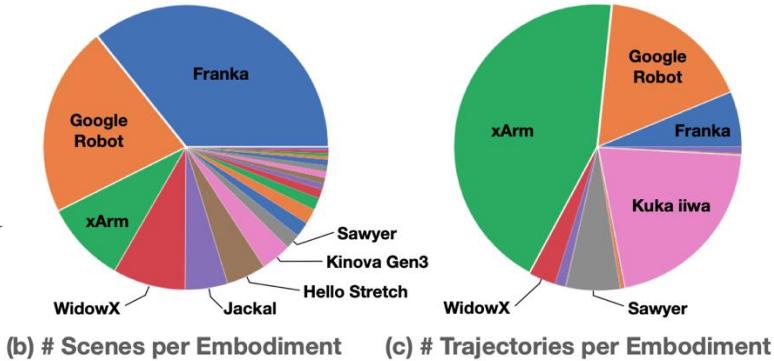
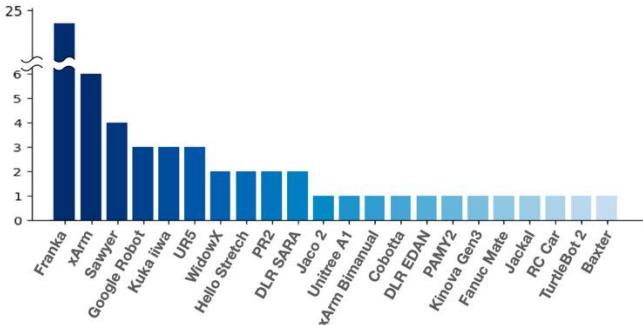
Open X-Embodiment: Robotic Learning Datasets and RT-X Models

Artifact

- 1000+ hours of robotic manipulation data
- 22 different robots, 34 research labs
- Publicly released for plug-and-play use
- Standardized data format and processing



RFMs Beyond Navigation



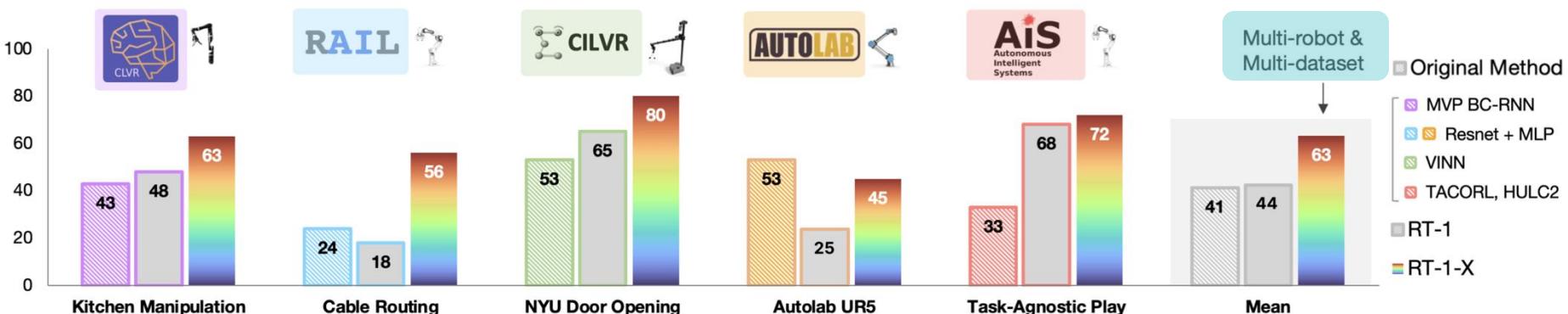


RFMs Beyond Navigation

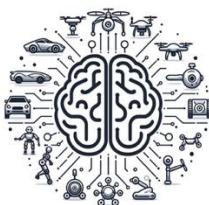
Task: Tabletop manipulation, different variants

Shared Abstraction: 7-DoF end-effector pose, normalized

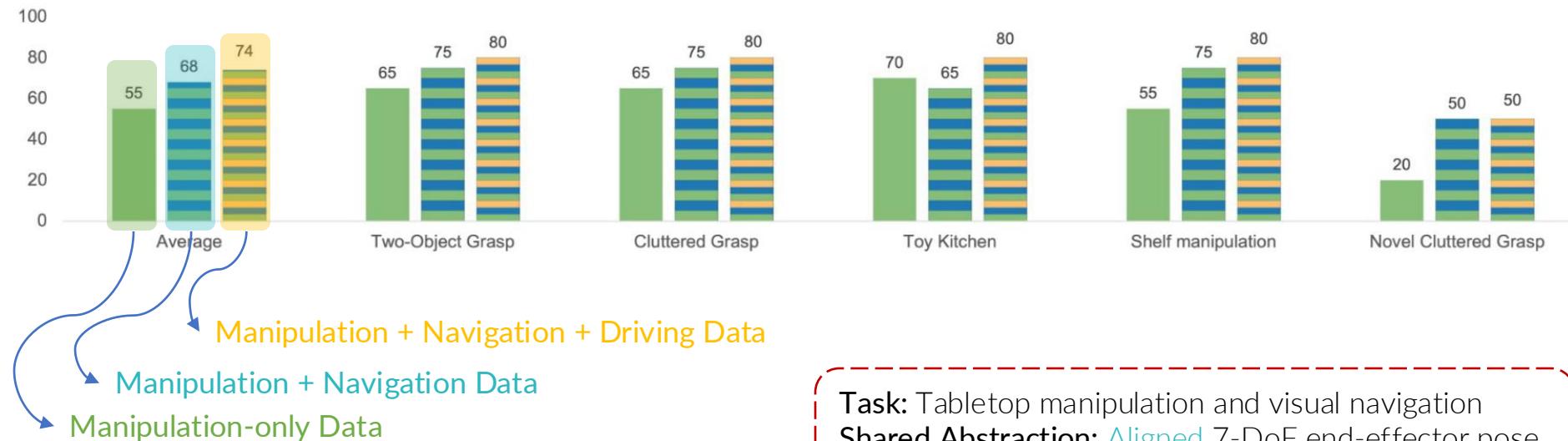
Observations: RGB images, prompt-conditioning



Cross-embodiment RFMs in tabletop manipulation show similar trends



RFMs Beyond Navigation



Further confirms the cross-embodiment hypothesis!

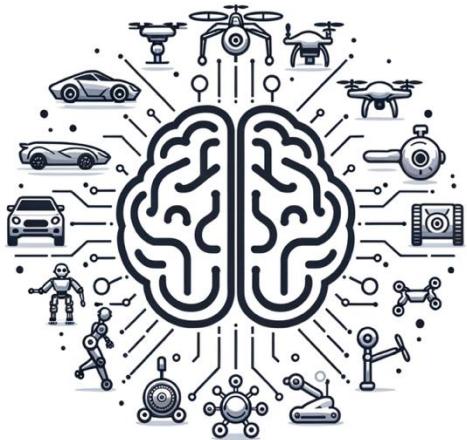
RFMs Beyond Navigation



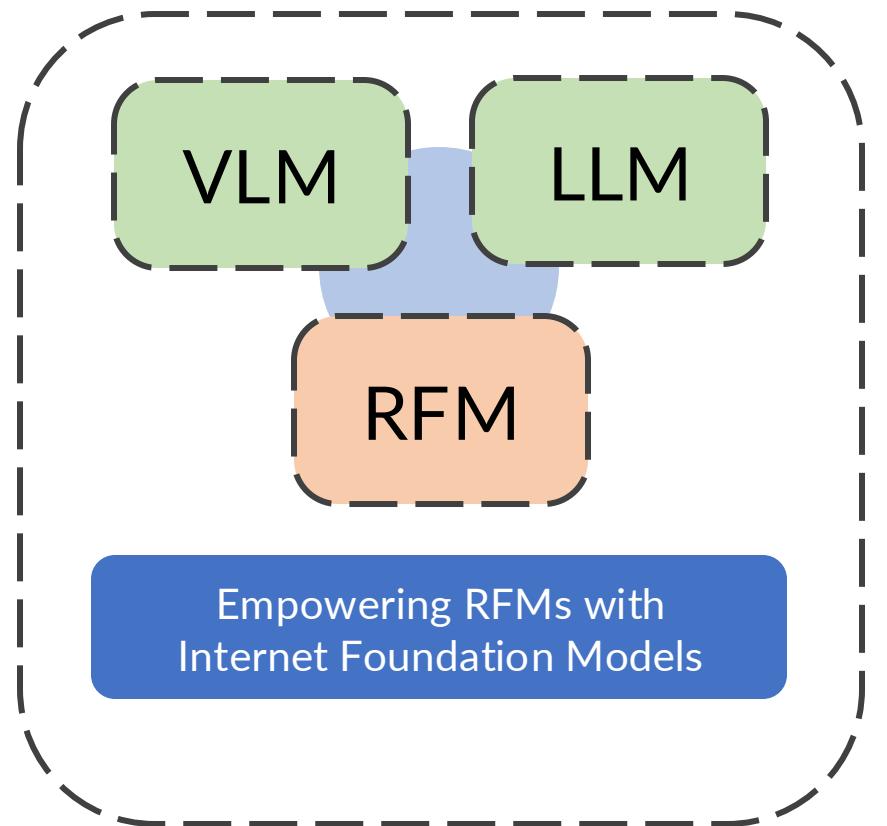
The same policy can control both the base and the arm!

Part 4: How can Internet Foundation Models help?

Foundation Models Of & For Robotics

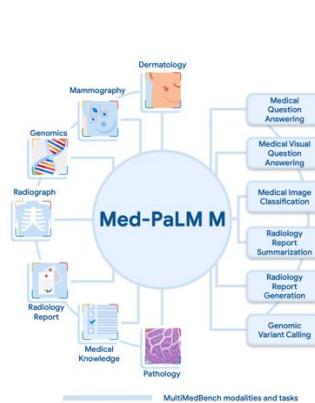
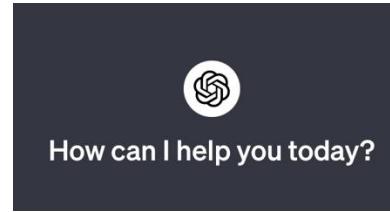


Training a Cross-Embodiment
“Robot Foundation Model”



Elsewhere...

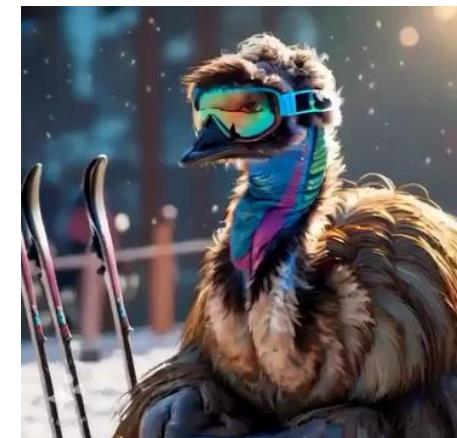
How did these models get here?



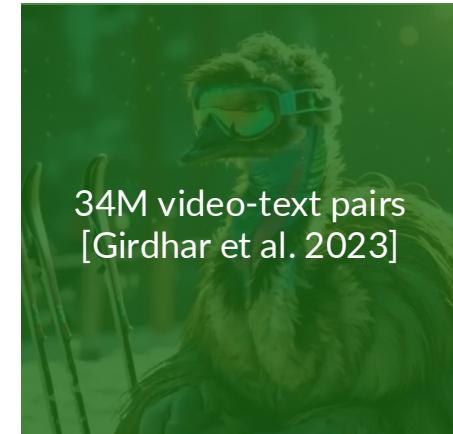
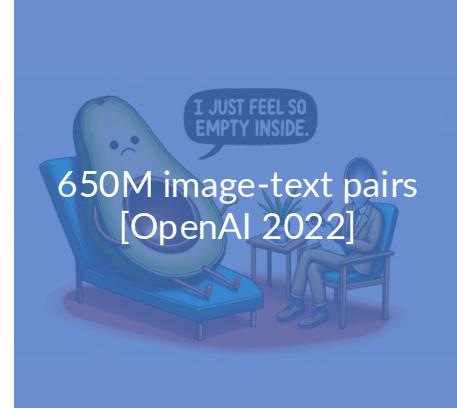
LLaVA Chatbot

What is unusual about this image?

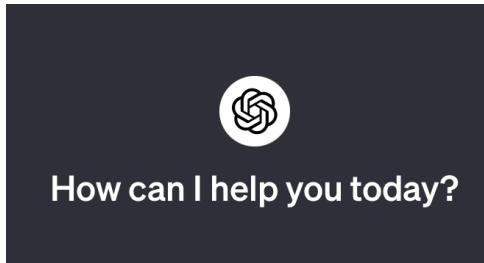
The image shows a person ironing clothes on an ironing board that is placed on the back of a moving vehicle, specifically a yellow taxi. This is an unusual and unsafe situation because ironing clothes while standing on the back of a moving vehicle is not only impractical but also poses a significant risk of injury to the person doing the ironing. The person is exposed to the elements, the risk of falling, and the potential for accidents due to the vehicle's movement. Additionally, the ironing board and the person's position on it could obstruct the driver's view or create a hazard for other road users.



The Secret Sauce



How much data will “solving robotics” need?



500B text tokens for pre-training

RedPajama-Data-v2: An open dataset with 30 trillion tokens for training large language models



11T multimodal tokens for pre-training



1+ Million hours of videos/audio

We will never have enough robot data

Can we reduce this number somehow?

What can LLMs do?



I am an indoor home robot and my job is to clean up the apartment. There are many dirty things all over the apartment. How should I do this, give me step by step instructions

Show drafts



Unfortunately, even though I can guide you, I don't have access to your specific capabilities and limitations as an indoor home robot. However, I can offer general cleaning steps that you can adapt based on your functions:

Step 1: Assess the Situation

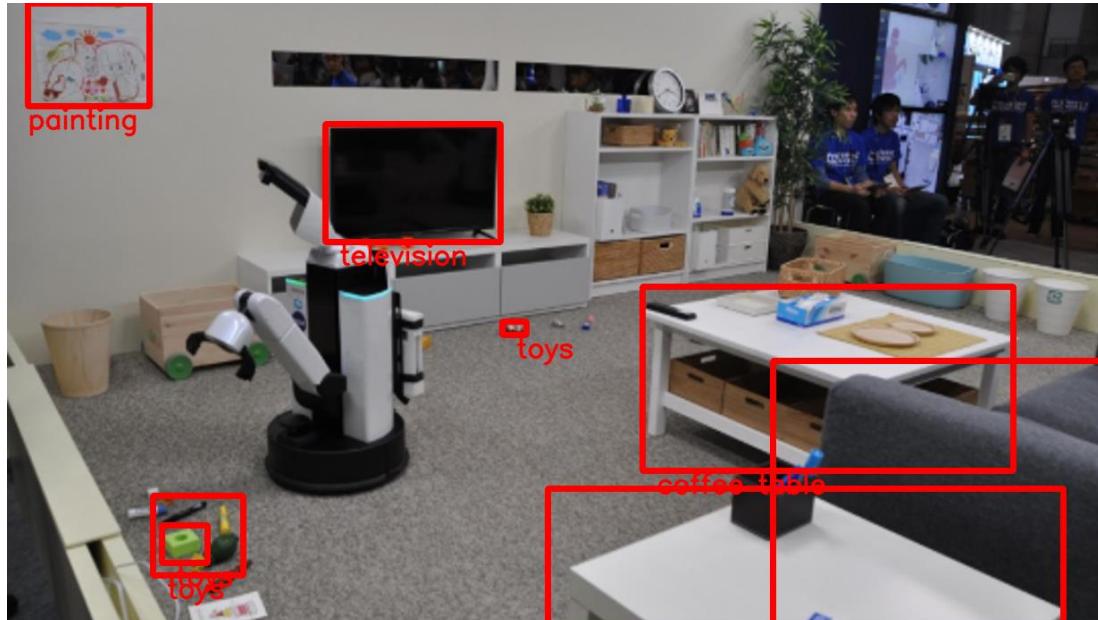
- **Start with a map:** If you have a map of the apartment, use it to plan your cleaning route. If not, create a mental map or explore systematically.
- **Identify the messes:** Look for areas with clutter, dust, spills, and debris. Prioritize based on urgency and difficulty (e.g., spills before dust bunnies).
- **Consider your tools:** What cleaning tools do you have (e.g., vacuum, mop, dusters)? Can you use them on all surfaces?

Step 2: Declutter and Tidy

- **Pick up objects:** Remove clutter like clothes, toys, or misplaced items from floors and surfaces. Place them in designated areas or bins.
- **Empty trash:** Find and empty trash cans and recycling bins.

LLMs are great at parsing text, simple reasoning, and constructing narratives

What can VLMs do?



Minderer et al., OWL-ViT, 2022

VLMs are great at scene understanding, grounding images in text

How can LLMs help robotics?

- LLMs for *parsing* instructions

How can LLMs help robotics?

- LLMs for planning
 - Writing PDDL/STL descriptions
 - Writing code

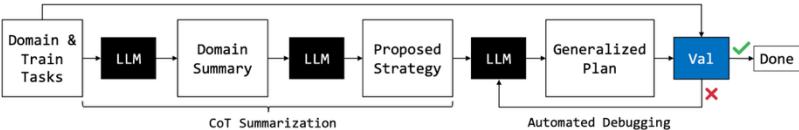
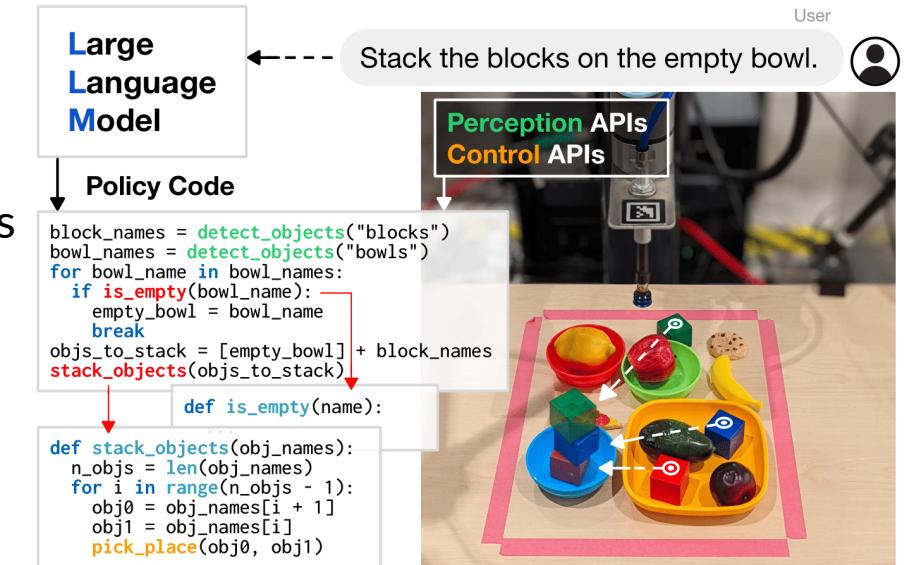


Figure 1: Overview of pipeline for generalized planning with pretrained LLMs. See text for details.

Generalized Planning in PDDL Domains with Pretrained Large Language Models

Tom Silver¹, Soham Dan², Kavitha Srinivas²,
Joshua Tenenbaum¹, Leslie Kaelbling¹, Michael Katz²

¹MIT Computer Science and Artificial Intelligence Laboratory, ²IBM Research
Correspondence: ts1vr@mit.edu, Michael.Katz1@ibm.com



Code as Policies: Language Model Programs for Embodied Control

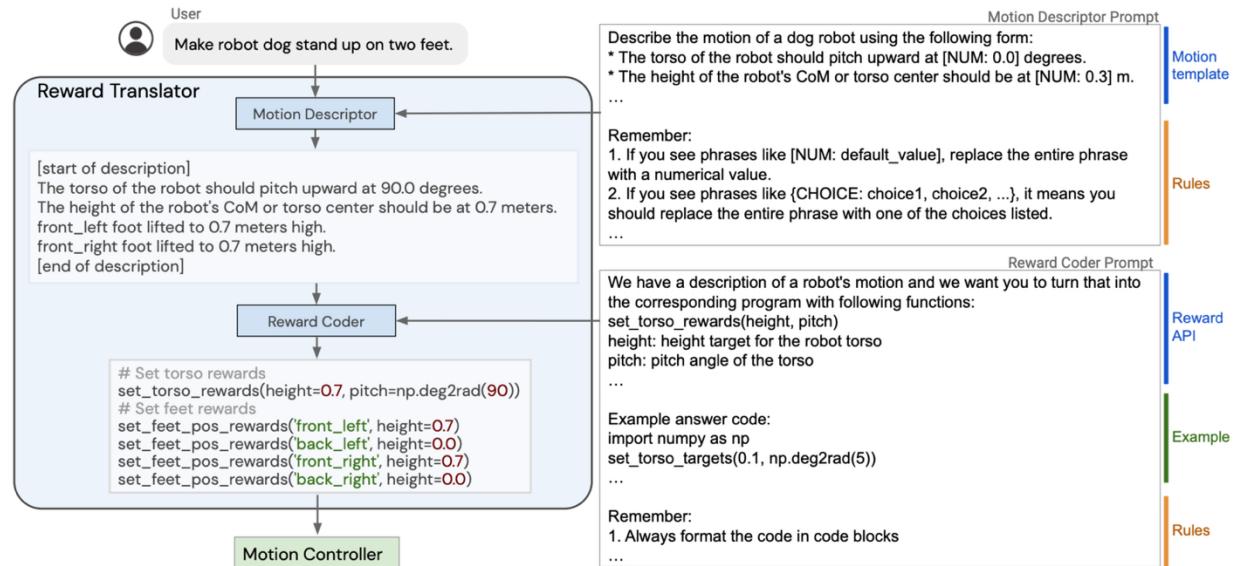
Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, Andy Zeng

PROGPROMPT: Generating Situated Robot Task Plans using Large Language Models

Ishika Singh¹, Valts Blukis², Arsalan Mousavian², Ankit Goyal², Danfei Xu²,
Jonathan Tremblay², Dieter Fox², Jesse Thomason¹, Animesh Garg²

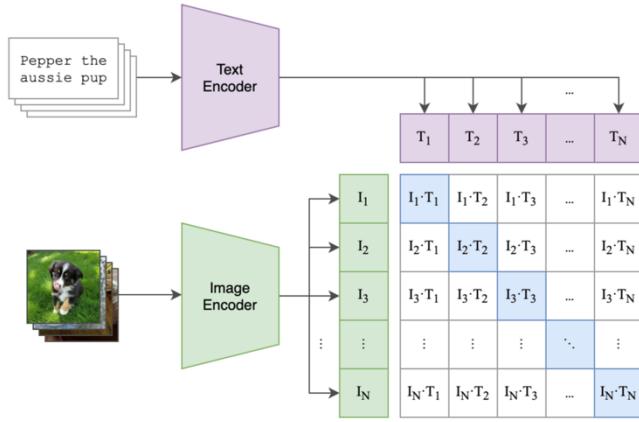
How can LLMs help robotics?

- LLMs for writing reward functions



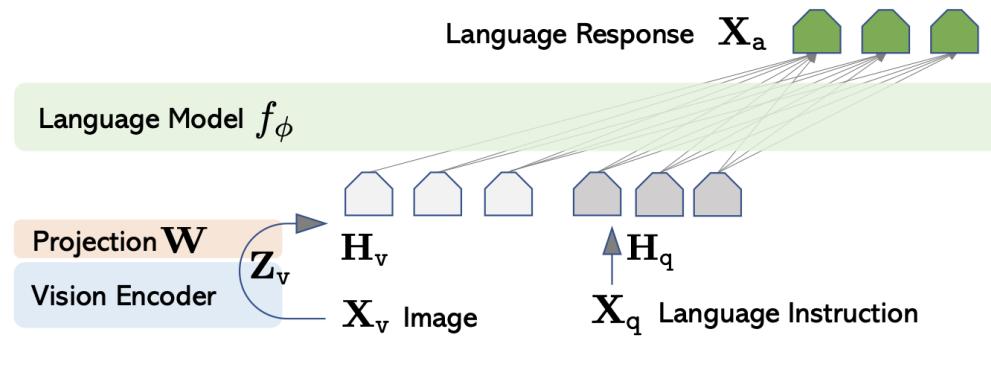
How can VLMs help robotics?

What is a VLM?



Learning Transferable Visual Models From Natural Language Supervision

Alec Radford^{*1} Jong Wook Kim^{*1} Chris Hallacy¹ Aditya Ramesh¹ Gabriel Goh¹ Sandhini Agarwal¹
Girish Sastry¹ Amanda Askell¹ Pamela Mishkin¹ Jack Clark¹ Gretchen Krueger¹ Ilya Sutskever¹

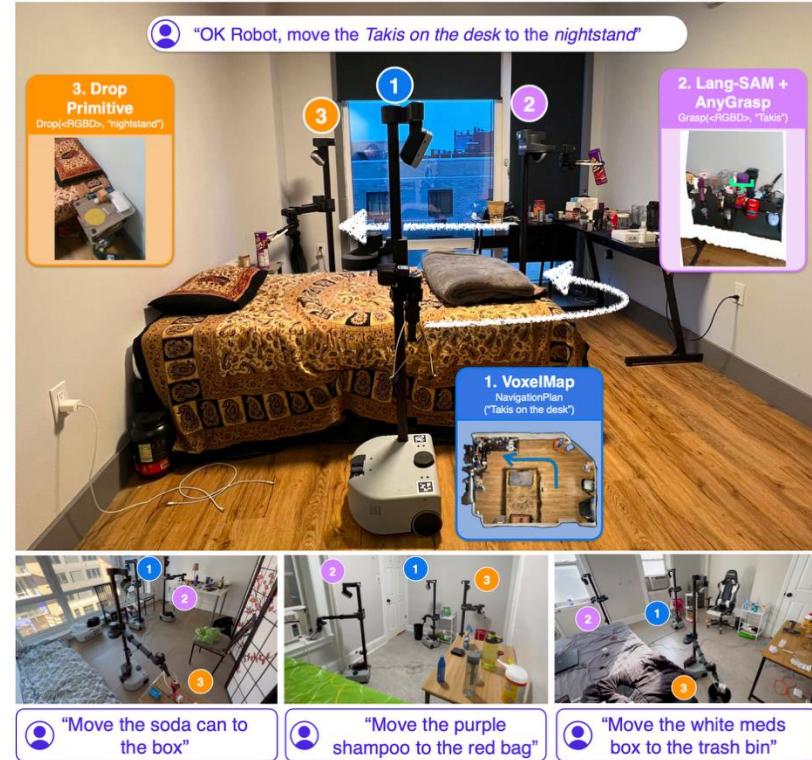


Visual Instruction Tuning

Haotian Liu^{1*}, Chunyuan Li^{2*}, Qingyang Wu³, Yong Jae Lee¹
¹University of Wisconsin-Madison ²Microsoft Research ³Columbia University
<https://llava-vl.github.io>

How can VLMs help robotics?

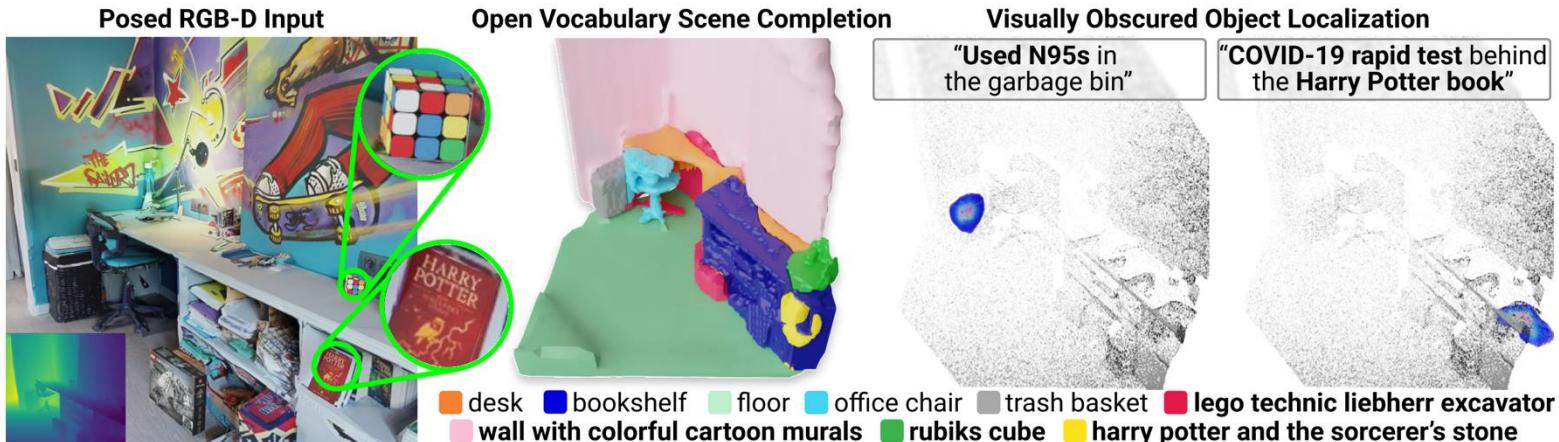
- VLMs for grounding images in text



OK-Robot:
What Really Matters in Integrating Open-Knowledge
Models for Robotics

How can VLMs help robotics?

- VLMs for detection and scene understanding



**Semantic Abstraction: Open-World 3D
Scene Understanding from 2D Vision-Language Models**

How can VLMs help robotics?

- VLMs for explainability

LINGO-1: Exploring Natural Language for Autonomous Driving



Ask a new question



How can VLMs help robotics?

- Long-context VLMs as *memory*



Towards Vision-Language-Action Models

RT-2: New model translates vision and language into action

LINGO-2: Driving with Natural Language

π_0 : A Vision-Language-Action Flow Model for General Robot Control

Physical Intelligence

OpenVLA:
An Open-Source Vision-Language-Action Model

PaLM-E: An Embodied Multimodal Language Model

Danny Driess^{1,2} Fei Xia¹ Mehdi S. M. Sajjadi³ Corey Lynch¹ Aakanksha Chowdhery³
Brian Ichter¹ Ayzaan Wahid¹ Jonathan Tompson¹ Quan Vuong¹ Tianhe Yu¹ Wenlong Huang¹
Yevgen Chebotar¹ Pierre Sermanet¹ Daniel Duckworth³ Sergey Levine¹ Vincent Vanhoucke¹
Karol Hausman¹ Marc Toussaint² Klaus Greff³ Andy Zeng¹ Igor Mordatch³ Pete Florence¹

Gemini Robotics: Bringing AI into the Physical World

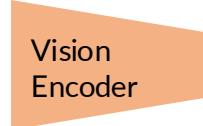
Gemini Robotics Team, Google DeepMind¹

Towards Vision-Language-Action Models

Autoregressive LLM

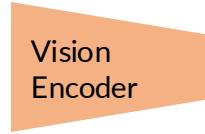


Autoregressive VLM



Towards Vision-Language-Action Models

Simple Autoregressive VLA



Transformer

General, Flexible VLA

- Interleaved text, images
- Output actions and text

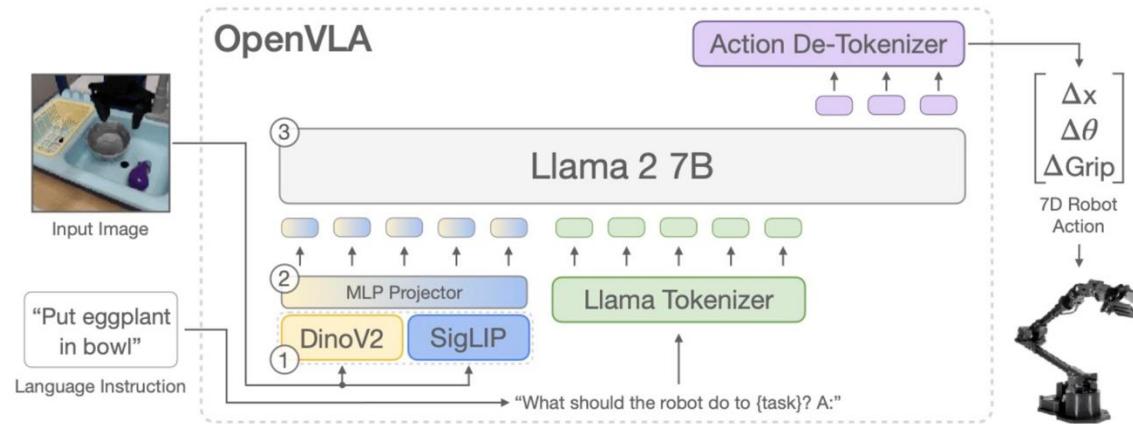


Towards Vision-Language-Action Models



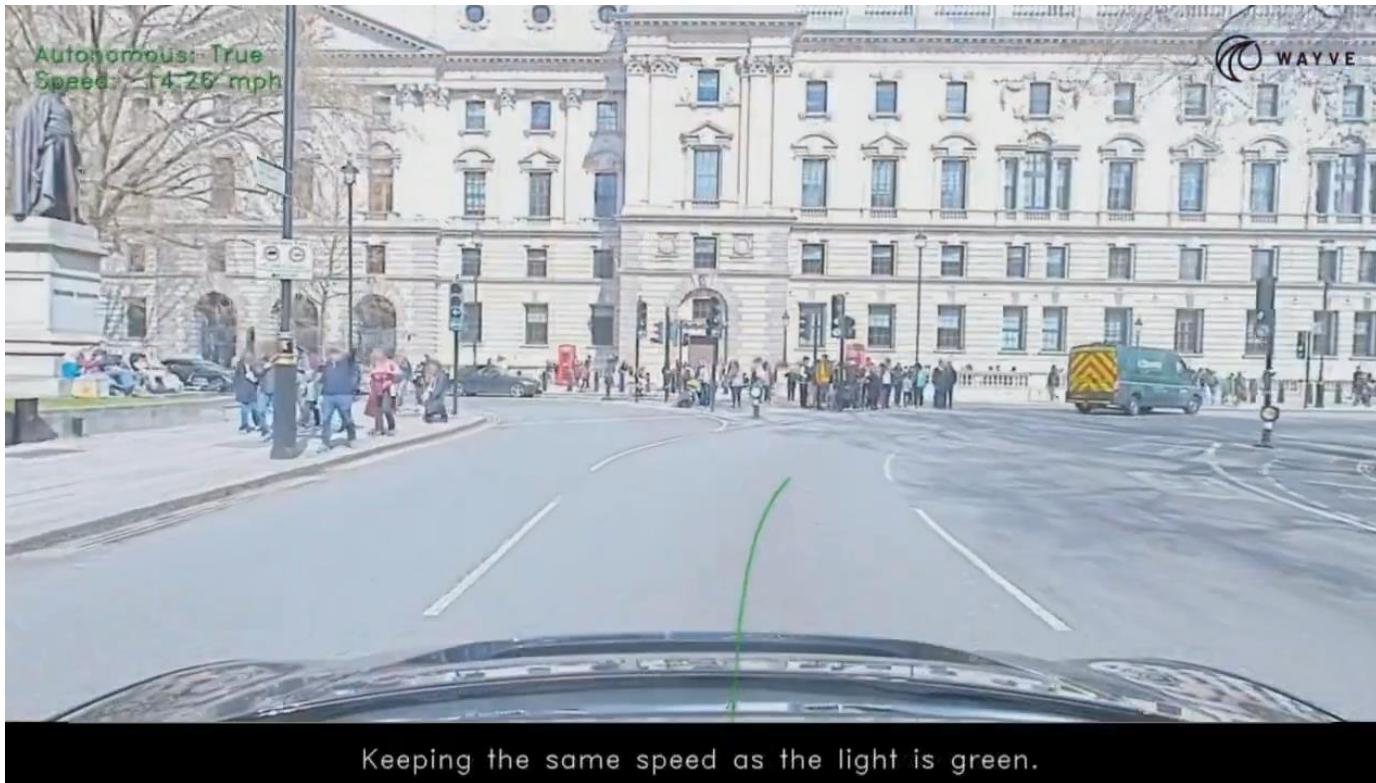
Vision Encoder

Transformer



OpenVLA:
An Open-Source Vision-Language-Action Model

Towards Vision-Language-Action Models



Takes vision input
Predicts real-world actions
Can output text explanations

Combining IFM and RFM with *Planning*

Not grounded in the robot's experience and capabilities

LLM

Parsing instructions, simple reasoning

VLM

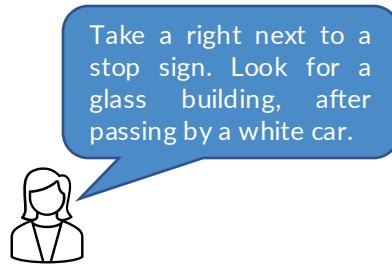
Ground visual observations in text

Planning

RFM

Robust control policies, grounded in experience & capabilities

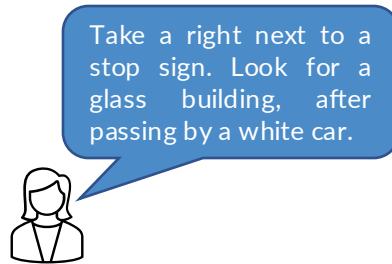
Instruction Following as Probabilistic Inference

 $\{v_1, v_2, \dots, v_k\}$

$$P(\text{success} | \bar{v}, \bar{\ell})$$

Simplifying Assumption: Instructions can be decomposed into landmark $\{l_1, l_2, \dots, l_n\}$

Instruction Following as Probabilistic Inference

 $\{v_1, v_2, \dots, v_k\}$

Task Success

$$P(\text{success}|\bar{v}, \bar{\ell}) \propto P(c_{\bar{v}} = 1|\bar{v})P(\bar{v}|\bar{\ell}) = \prod_{1 \leq j < k} \gamma^{d(v_j, v_{j+1})} \max_{1 \leq t_1 \leq \dots \leq t_n \leq k} \prod_{1 \leq i \leq n} P(v_{t_i}|\ell_i)$$

Navigation Success

Grounding Accuracy

Instruction Following as Probabilistic Inference

Take a right next to a stop sign. Look for a glass building, after passing by a white car.



$$P(\text{success} | \bar{v}, \bar{l}) \propto \prod_{1 \leq j < k} \gamma^{d(v_j, v_{j+1})} \max_{1 \leq t_1 \leq \dots \leq t_n \leq k} \prod_{1 \leq i \leq n} P(v_{t_i} | l_i)$$



Sum of distances along a traversal through waypoints $\{v_1, v_2, \dots, v_k\}$

Navigation Success

Probability that image v_{t_i} contains the textual landmark l_i

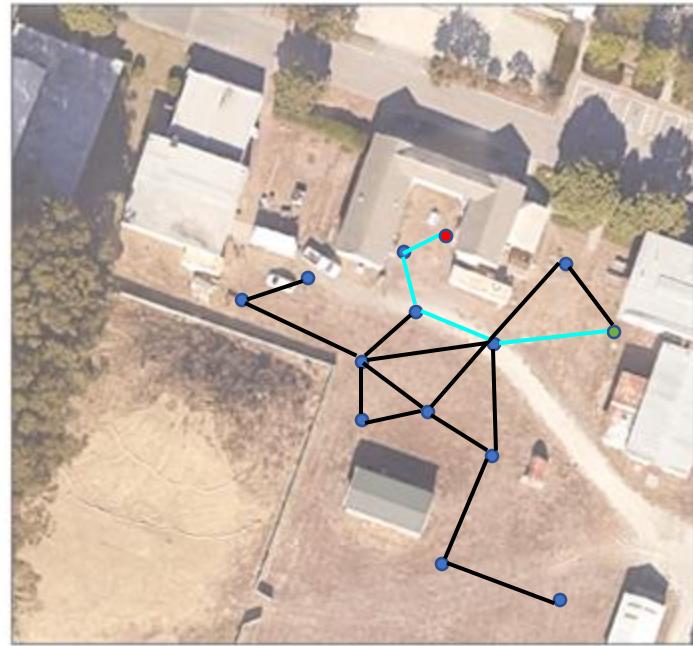
Image-text grounding VLM

How do we represent environment memory for a planner?

Memory: Topological Graph

Sparse and flexible episodic memory

- Nodes are observations o_i
- Edges are estimated distances $d(o_i, o_j)$
- Goal-reaching ~ graph search

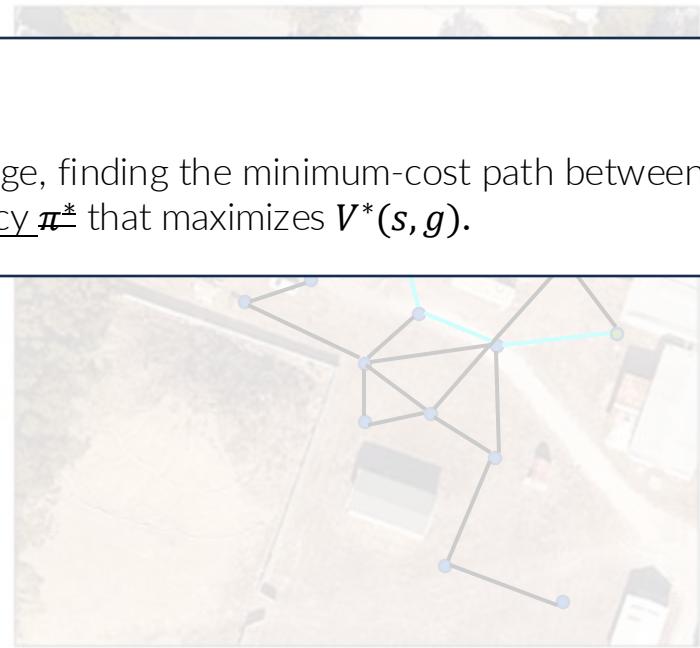


Memory: Topological Graph

Theorem

For the optimal value function V^* for nearby goals and full coverage, finding the minimum-cost path between s, g in a graph with edge-weights $-V^*(s_i, s_j)$ recovers the optimal policy π^* that maximizes $V^*(s, g)$.

- Edges are estimated distances $d(o_i, o_j)$
- Goal-reaching \sim graph search



Instruction Following as Probabilistic Inference

$$P(\text{success} | \bar{v}, \bar{l}) \propto \prod_{1 \leq j < k} \gamma^{d(v_j, v_{j+1})} \max_{1 \leq t_1 \leq \dots \leq t_n \leq k} \prod_{1 \leq i \leq n} P(v_{t_i} | l_i)$$

Sum of distances along a traversal through waypoints $\{v_1, v_2, \dots, v_k\}$

Navigation Success

Probability that image v_{t_i} contains the textual landmark l_i

Image-text grounding VLM

Instruction following ~ modified graph search: *solve with dynamic programming!*

Landmarks To Visit

blue semi-truck

traffic cone

blue dumpster

picnic bench

Follow the road and take the right, you should see a blue semi-truck. Behind the truck, take a right next to an orange traffic cone. Go towards a blue dumpster and take left. Look for a picnic bench.



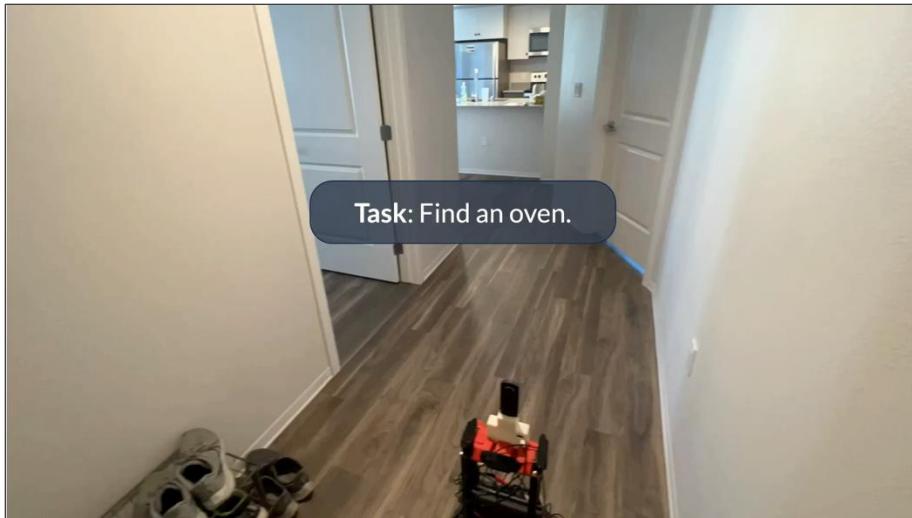
Overhead Visualization



Egocentric Observations



More Ways to Plan with IFMs



LLMs as a Planning Heuristic

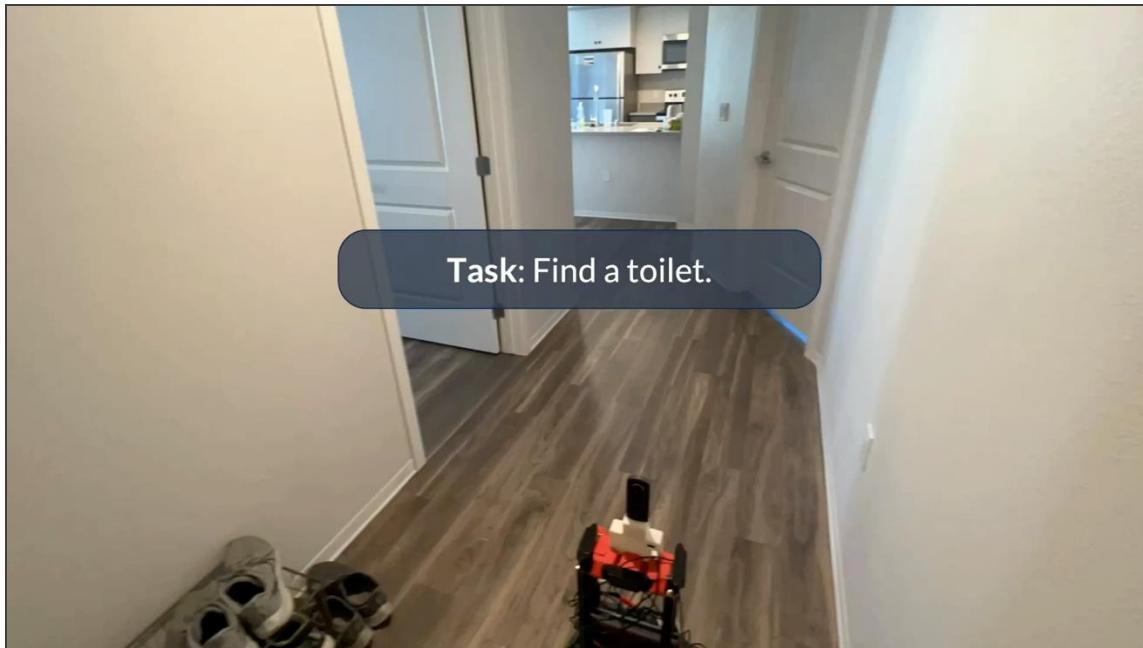
Achieves SOTA on Habitat ObjectNav Task!



Semantic Memory with VLMs in the Wild

Extensive real-world evaluation in 10 AirBnBs!

Exploration with LLMs as a Search Heuristic



Key Idea

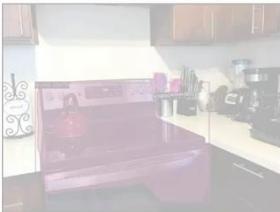
A* search with an LLM heuristic!

In a new environment, use a VLM to suggest proposals for future states to explore; then ask an LLM to “think through” the futures and pick the best proposal.

Achieves SOTA on Habitat ObjectNav Task!

Semantic Instance Memory with VLMs

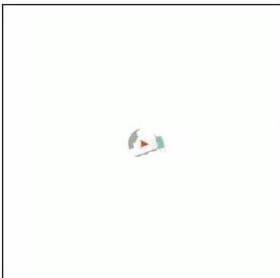
Observation



Third-person view



Semantic Map



Key Idea

Home robot with a semantic instance memory to find any object!

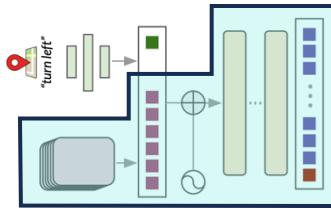
Build a dense semantic map augmented with VLMs to detect arbitrary objects specified by language, image, or categories. Can be dropped in a new house and builds a consistent semantic map on-the-fly.

Tested in 10 AirBnBs!

Adapt to new objectives



Adapt to new modalities



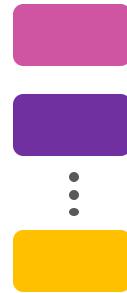
Pre-trained RFM

Takeaways

Prompts



Translators



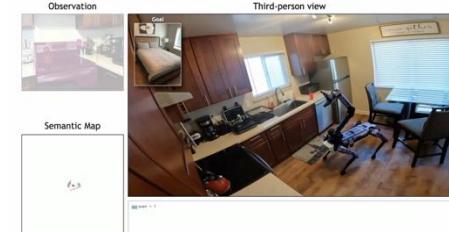
RFM



Cross-embodiment RFM

- Single policy to control many robots (zero-shot)
- Robustness to visual and dynamical perturbations
- Applies to navigation, manipulation, and more

Combine with V/LLMs via planning



Discussion & Open Challenges

- Evaluation of closed-loop models is *hard*
 - time-series data does not follow IID assumptions
 - offline metrics (e. g., MSE, ADE) don't align with real performance
- Safety and regulation for real-world deployment
- Robot policies need to run inference onboard and fast
- V/LLM hallucination can be a real problem
- Robustness, interpretability, faulty analysis
- ...

Thank You!

shahd@princeton.edu