# LEARNING TO ACT

Session 3: Multi-Objective Learning

Dana Kulić
Monash University
dana.kulic@monash.edu

RVSS 2026

# Today's Session – Multi-Objective Learning

- Generative models for learning policy
- Foundation models for learning policy

# Behavioural Cloning – problems revisited

- How to collect the expert data?
- What is the right state representation? Does the robot see the same things as the expert does?
- Expert demonstrations may cover only a very small region of the state-space
  - For large state/action spaces, may require a huge data collection effort
- **What should the robot do when it encounters a situation that wasn't seen in the dataset?**
- **How to handle variations in strategy?**

# Generative Models for Action

# Behavioural Cloning - reminder

- Collect data from demonstration episodes $\mathcal{D}(e_{1:N})$
- Each episode is a sequence of states and actions $e_i = (s_0, a_1, s_1, a_2, \ldots s_T)$
- Learn a policy $\phi(s)$ using supervised learning:

$$L = (a_{\mathcal{D}}(s) - \phi(s))^2$$

  - The state $s$ corresponds to the input data
  - The action $a$ corresponds to the label
  - Behavioural cloning learns the policy function $\phi(s)$ to minimise the difference between the estimated action and the observed expert action from each state

# Behavioural Cloning - reminder

- Collect data from demonstration episodes $\mathcal{D}(e_{1:N})$
- Each episode is a sequence of states and actions $e_i = (s_0, a_1, s_1, a_2, \ldots s_T)$
- Learn a policy $\phi(s)$ using supervised learning:

$$L = (a_{\mathcal{D}}(s) - \phi(s))^2$$

  - The state $s$ corresponds to the input data
  - The action $a$ corresponds to the label
  - Behavioural cloning learns the policy function $\phi(s)$ to minimise the difference between the estimated action and the observed expert action from each state

# New Problem Formulation

Learn a conditional probability density model (generative model)

$$\rho_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{c})$$

That accurately captures the underlying probability distribution of the demonstration data $\rho_{\mathcal{D}}(\boldsymbol{a}|\boldsymbol{c})$

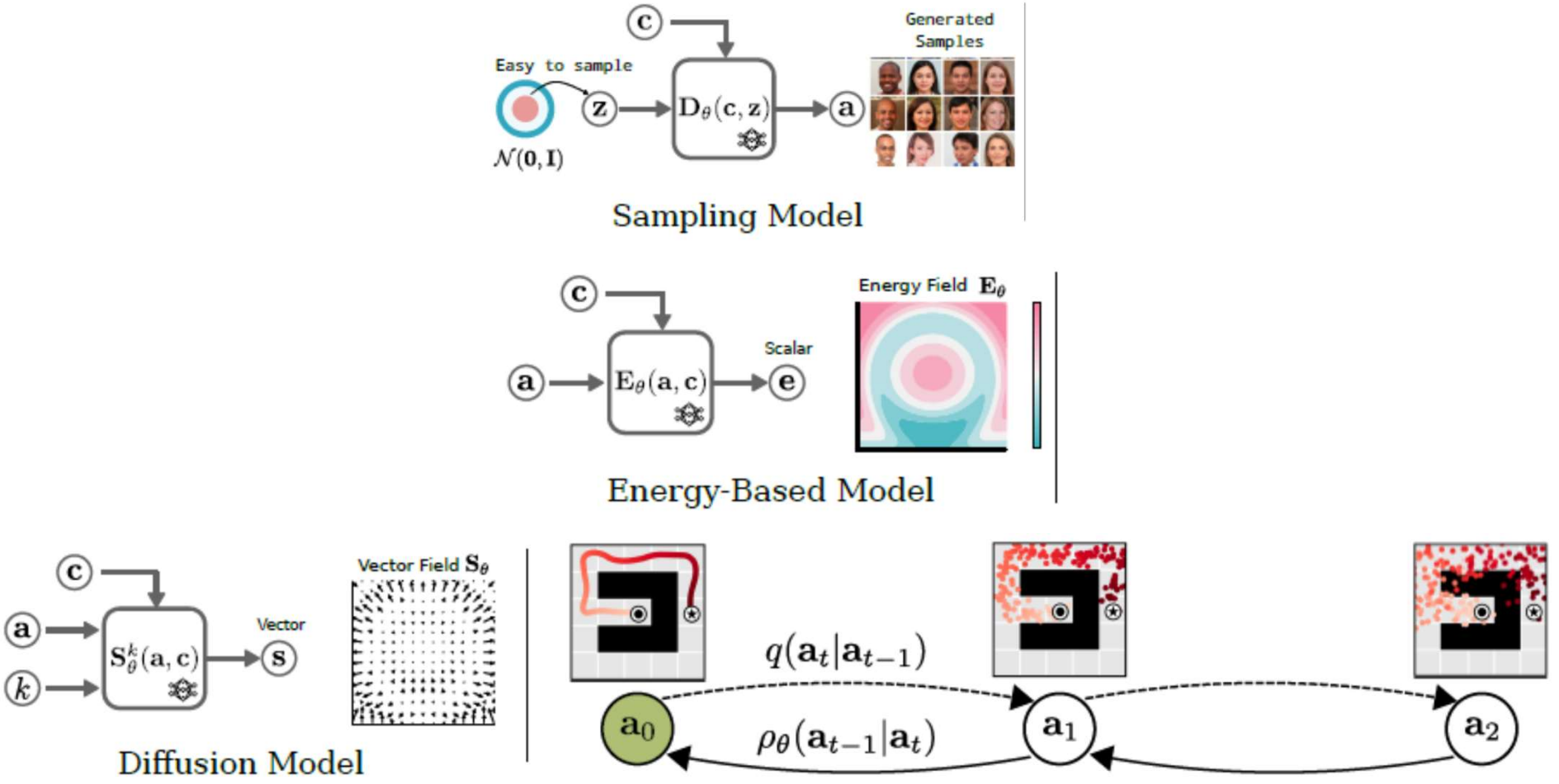
Where ***a*** is the action and ***c = (o,g)*** are the conditions (observations and goal)

Find the distribution that minimises the divergence between the model and the data distributions

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{a},\boldsymbol{c}\sim\mathcal{D}} \left[\mathbb{D}(\rho_{\mathcal{D}}(\boldsymbol{a}|\boldsymbol{c}), \rho_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{c}))\right]$$

J. Urain, A. Mandlekar, Y. Du, N. M. M. Shafiullah, D. Xu, K. Fragkiadaki, G. Chalvatzaki, J. Peters, Deep Generative Models in Robotics: A Survey of Learning from Multimodal Demonstrations, 2024

# Density Estimation Models




J. Urain, A. Mandlekar, Y. Du, N. M. M. Shafiullah, D. Xu, K. Fragkiadaki, G. Chalvatzaki, J. Peters, Deep Generative Models in Robotics: A Survey of Learning from Multimodal Demonstrations, 2024

# Diffusion Policy: Visuomotor Policy Learning via Action Diffusion

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, Shuran Song

RSS2023 paper

IJRR2024 paper

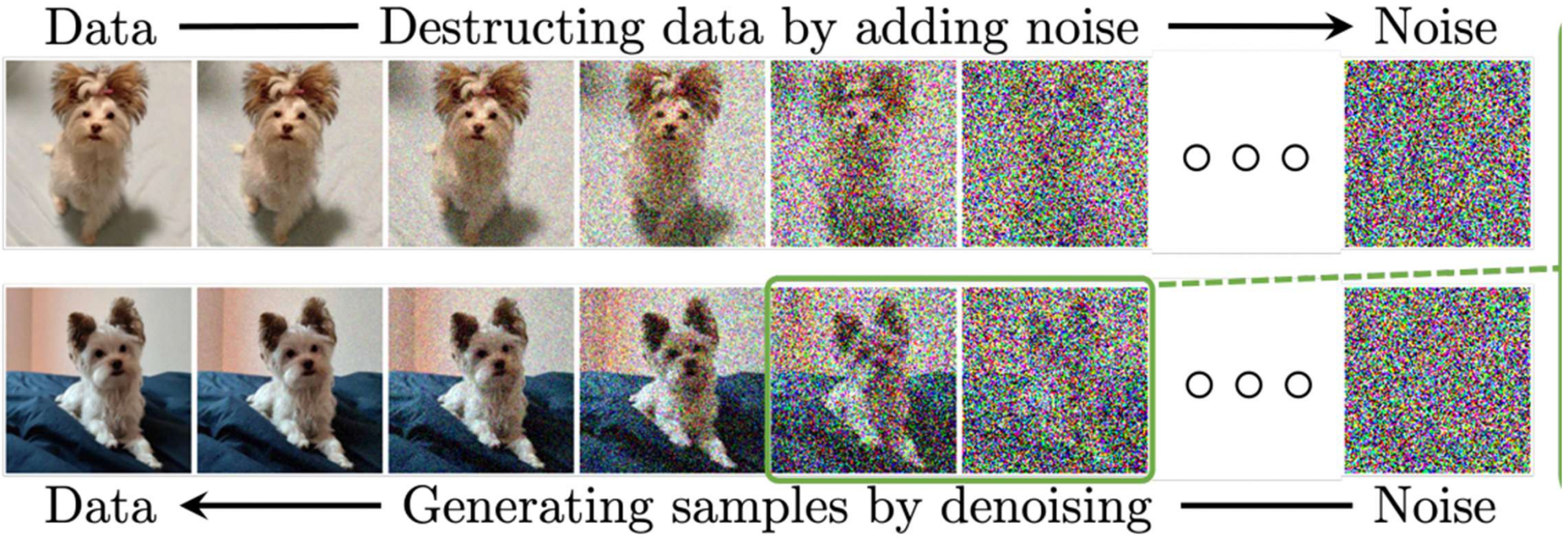
Project website

# Motivation

- How to best represent the action policy when learning from demonstrations?
  - Want to represent multimodal action distributions
  - Want to handle high-dimensional output space
  - Training stability

- Proposed approach:
  - represent the action as a conditional denoising diffusion process
  - Embed the action selection within a receding-horizon control framework
  - Policy conditioned on visual input
  - Investigate alternatives for the policy network architecture
  - Extensive validation

# Diffusion in Images




https://arshren.medium.com/noise-to-masterpiece-navigating-ais-diffusion-model-2b6de747a610

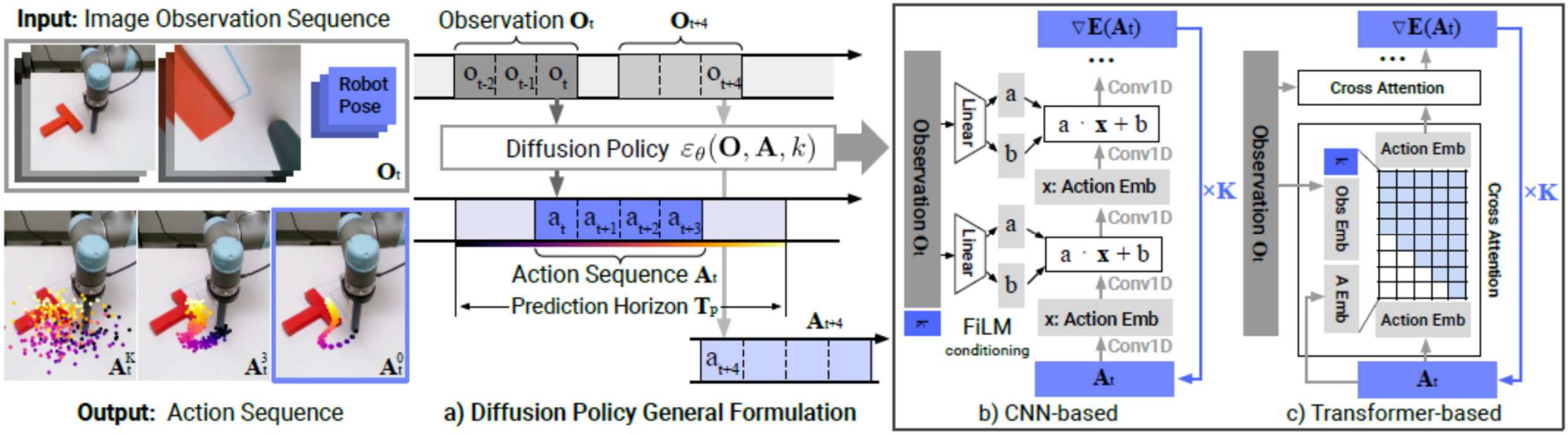
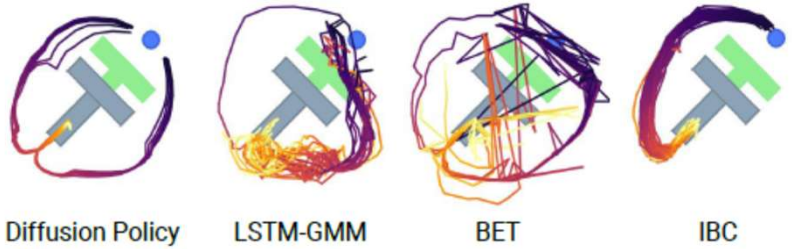



**Figure 2. Diffusion Policy Overview** a) General formulation. At time step $t$, the policy takes the latest $T_o$ steps of observation data $O_t$ as input and outputs $T_a$ steps of actions $A_t$. b) In the CNN-based Diffusion Policy, FiLM (Feature-wise Linear Modulation) Perez et al. (2018) conditioning of the observation feature $O_t$ is applied to every convolution layer, channel-wise. Starting from $\mathbf{A}_t^K$ drawn from Gaussian noise, the output of noise-prediction network $\varepsilon_\theta$ is subtracted, repeating $K$ times to get $\mathbf{A}_t^0$, the denoised action sequence. c) In the Transformer-based Vaswani et al. (2017) Diffusion Policy, the embedding of observation $\mathbf{O}_t$ is passed into a multi-head cross-attention layer of each transformer decoder block. Each action embedding is constrained to only attend to itself and previous action embeddings (causal attention) using the attention mask illustrated.

# Key Desirable Properties of Diffusion Policies

- Models multi-modal action distributions




- Works better with position control (vs. velocity control)
- Action-Sequence prediction is a good idea
  - Temporal action consistency
  - Robustness to idle actions
- Training Stability
- Connection to Control Theory
  - If the model is linear and cost quadratic, DP generates proportional controller with optimised gain!

# Evaluation

- Simulation Environments and datasets:
  - Robomimic: robotic manipulation benchmark (both proficient and non-proficient demonstrators)
  - Push-T: pushing a T-shaped block
  - Multimodal Block Pushing (tests multimodal policies)
  - Franka Kitchen: long-horizon tasks
- Real world evaluation:
  - Uni-manual tasks: Push-T, Mug flipping, Sauce Pouring and spreading
  - Bi-manual tasks: Egg Beater, Mat Unrolling, Shirt Folding

- Videos: https://diffusion-policy.cs.columbia.edu/

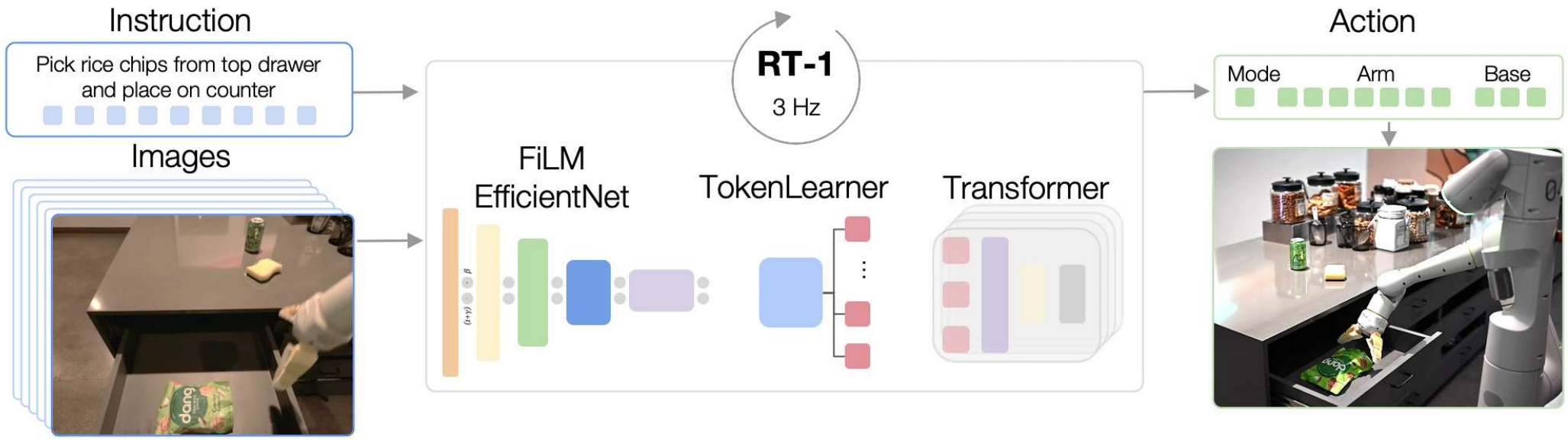
# Foundation Models for Action

# Moving beyond single-task learning

- Imitation learning problem formulation:  we want to learn the best policy for a specific (usually single) task

- Foundation models:  train in a self-supervised way (not for a specific task), then apply to many different tasks (with prompting or possibly some model fine-tuning)
  - Could we apply this idea to robots?
  - What kind of data would we need?
  - Anything else we need?

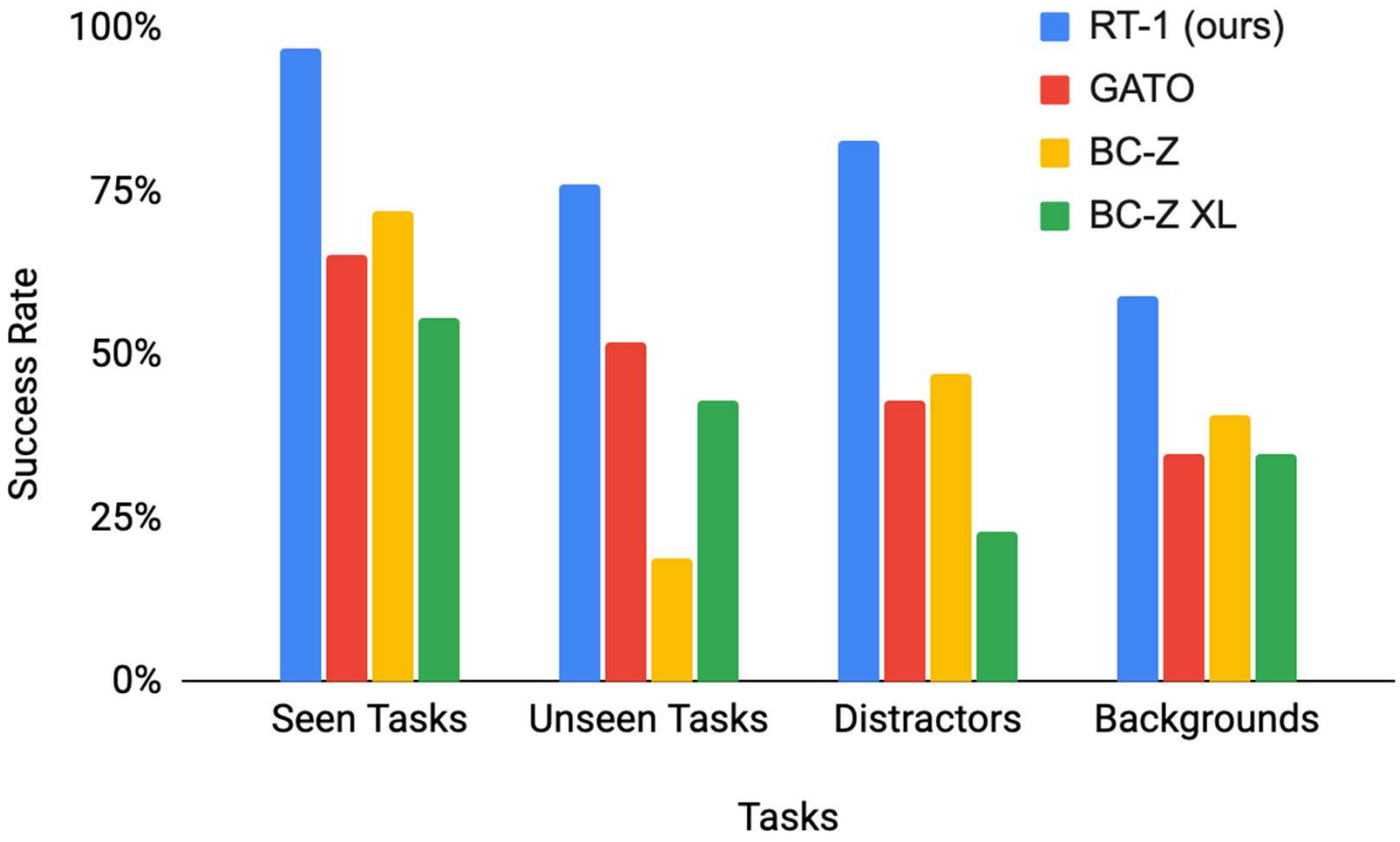
# RT-1



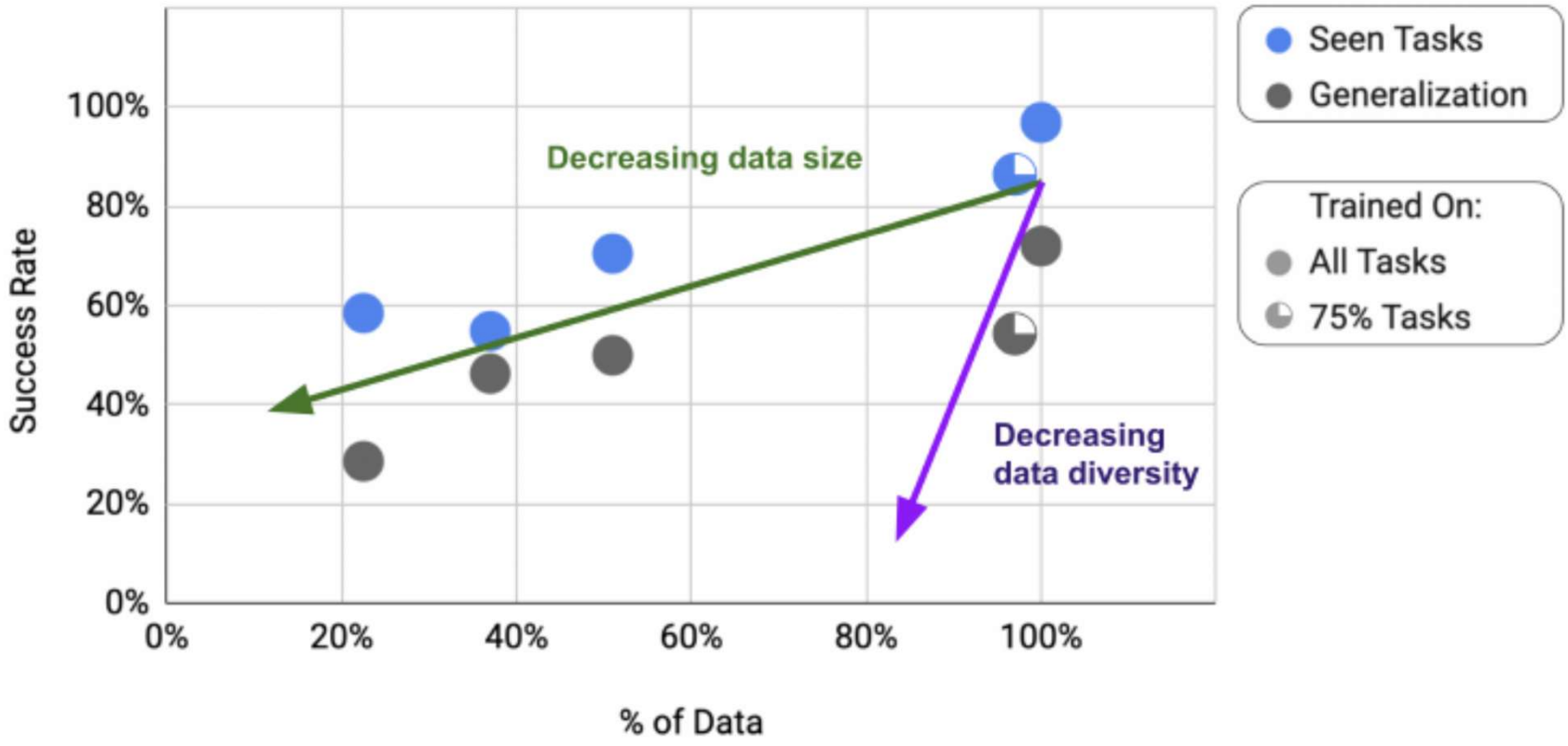

- Train using classical behaviour cloning, minimise the difference between action sequence observed in the training data and action sequence predicted by model
  - Assumes all training data are successful examples of action execution

https://robotics-transformer1.github.io/

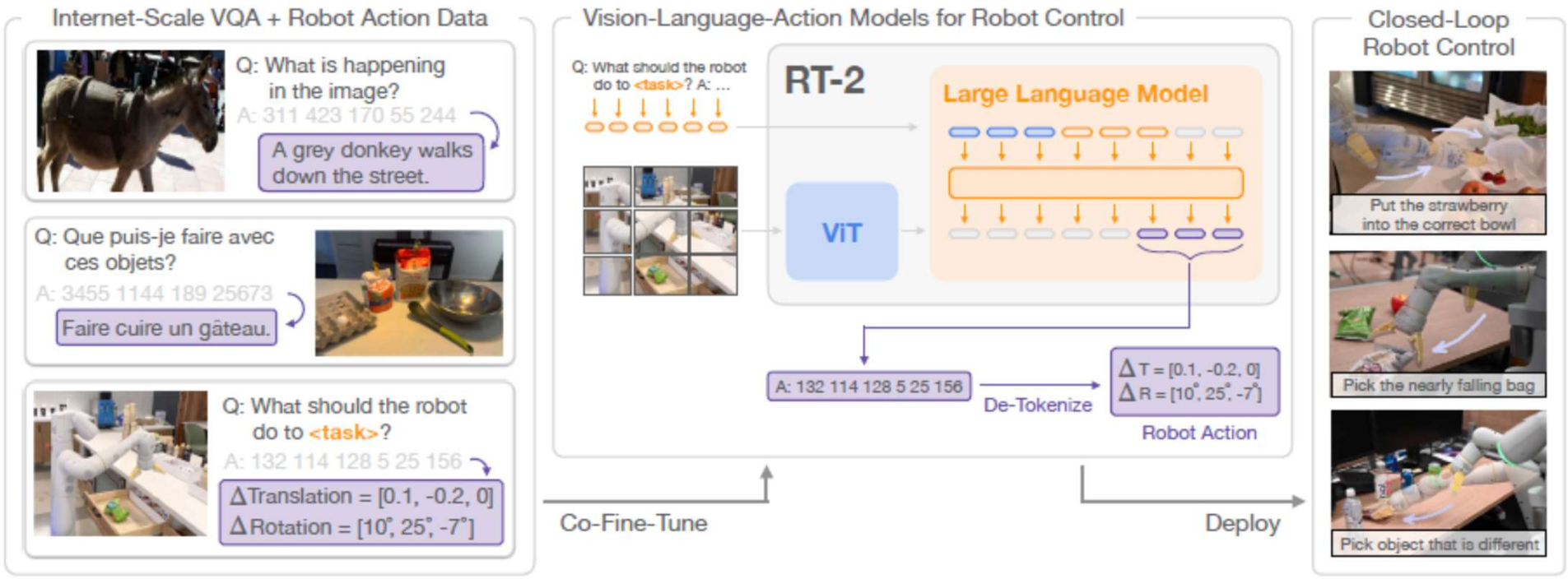
# Results

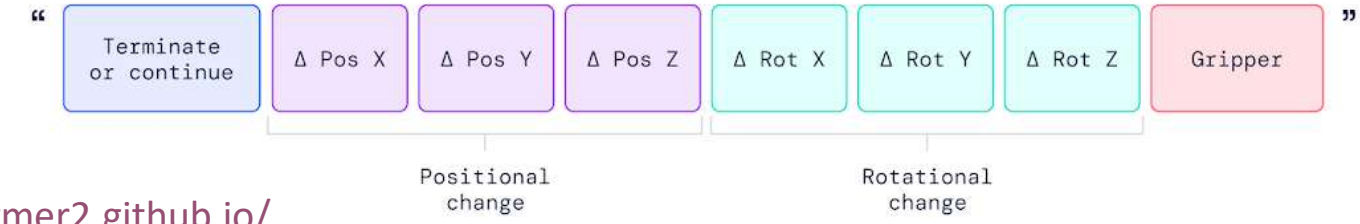

100%
75%
50%
25%
0%
Success Rate
RT-1 (ours)
GATO
BC-Z
BC-Z XL
Seen Tasks
Unseen Tasks
Distractors
Backgrounds
Tasks

# How does performance vary with data?



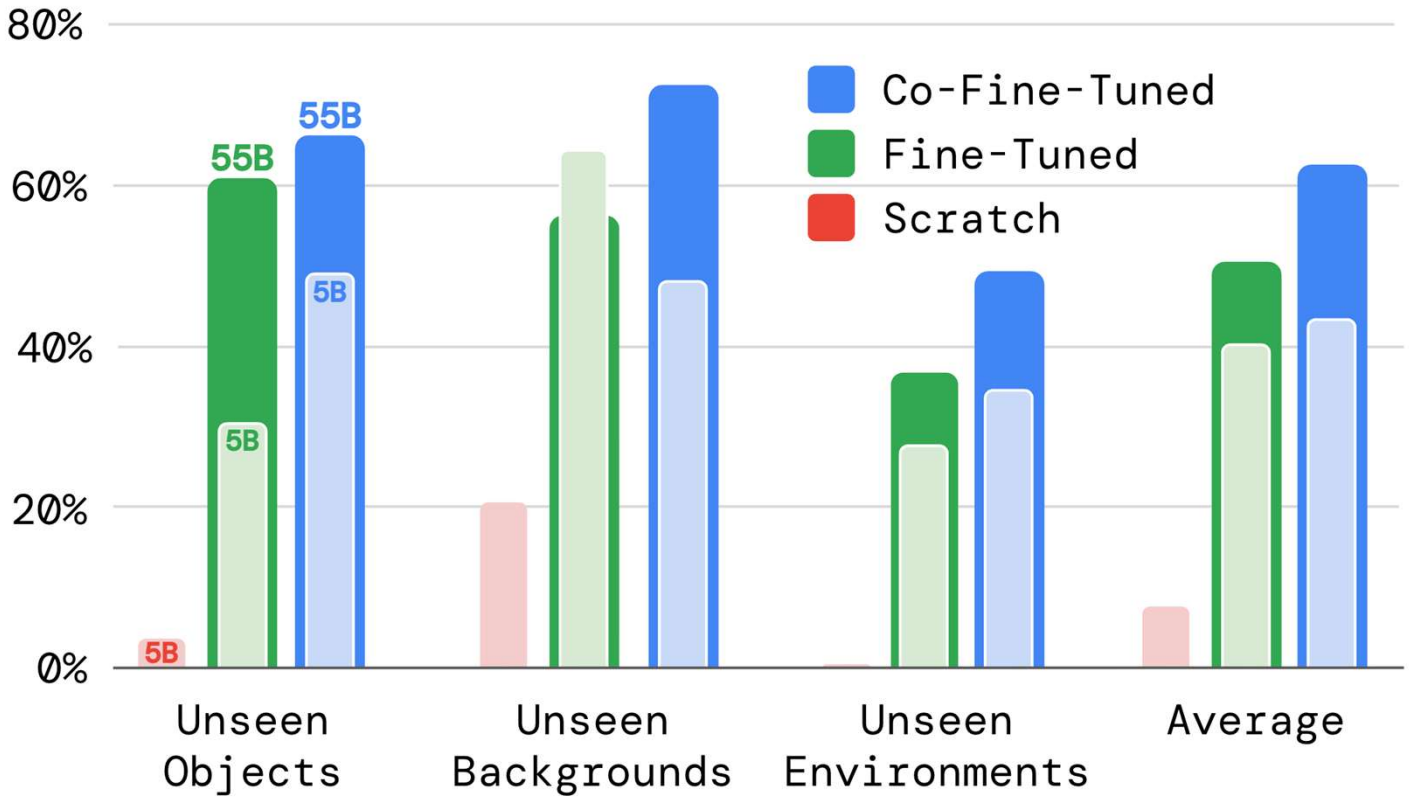
# RT-2



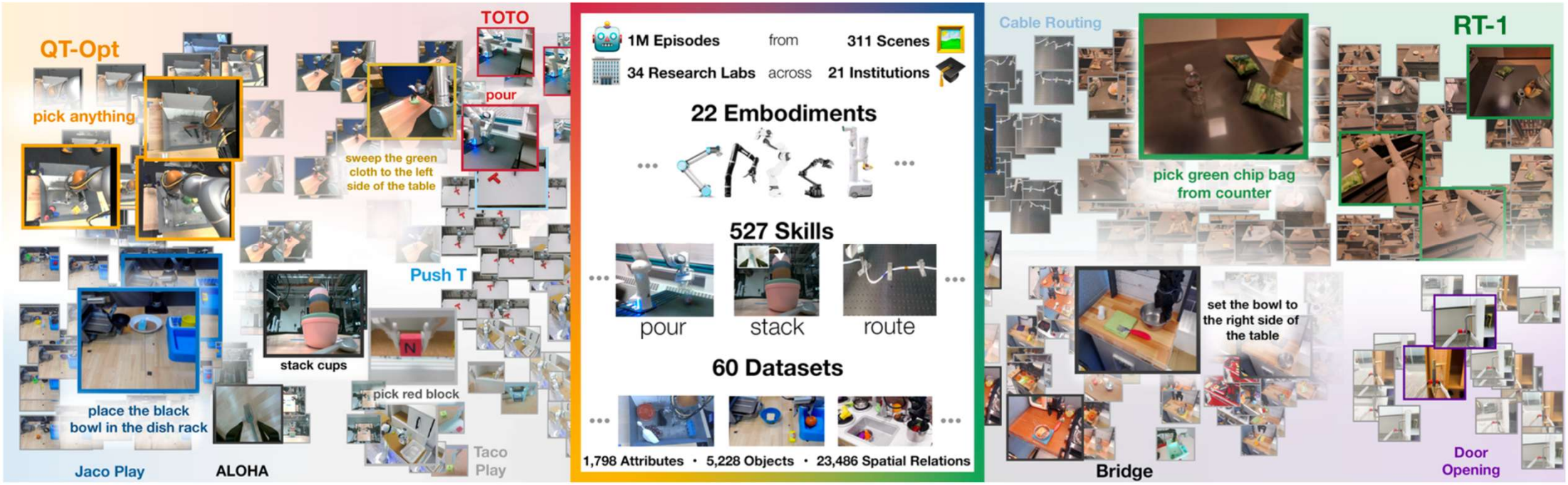




https://robotics-transformer2.github.io/

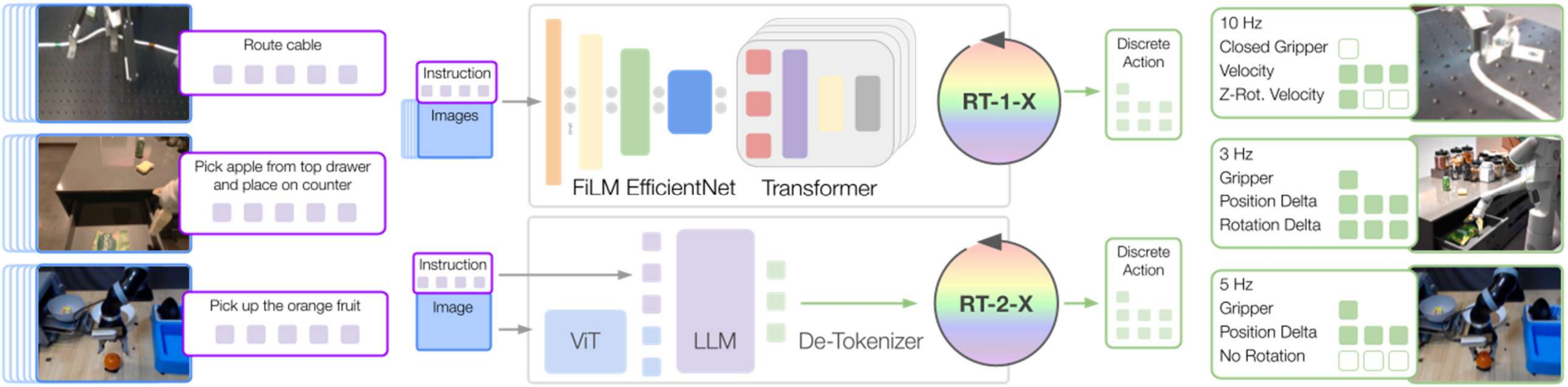
# Results


80%
60%
40%
20%
0%
55B
55B
5B
5B
5B
Co-Fine-Tuned
Fine-Tuned
Scratch
Unseen Objects
Unseen Backgrounds
Unseen Environments
Average

# RT-X

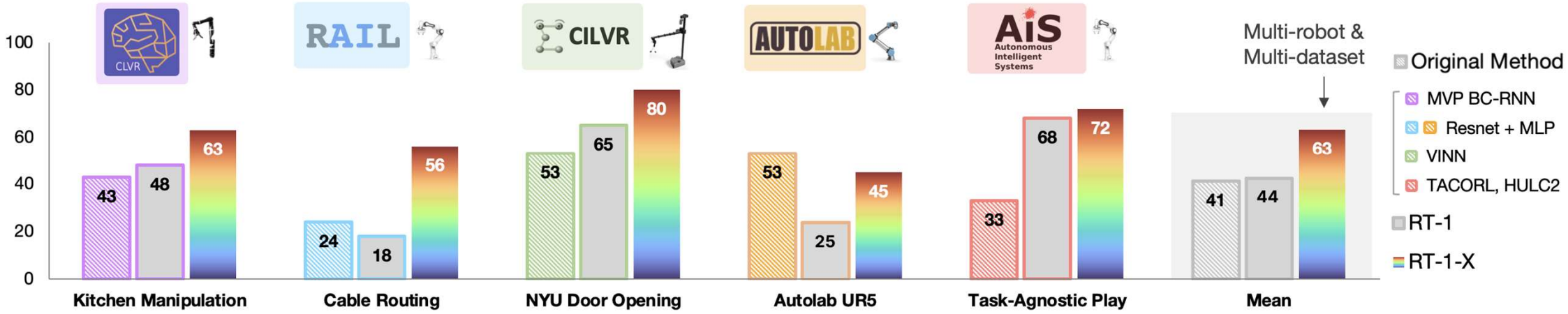

QT-Opt
pick anything
TOTO
pour
sweep the green cloth to the left side of the table
Push T
stack cups
pick red block
place the black bowl in the dish rack
Jaco Play
ALOHA
Taco Play
1M Episodes from 311 Scenes
34 Research Labs across 21 Institutions
22 Embodiments
527 Skills
pour
stack
route
60 Datasets
1,798 Attributes • 5,228 Objects • 23,486 Spatial Relations
Cable Routing
RT-1
pick green chip bag from counter
set the bowl to the right side of the table
Bridge
Door Opening

# RT-X

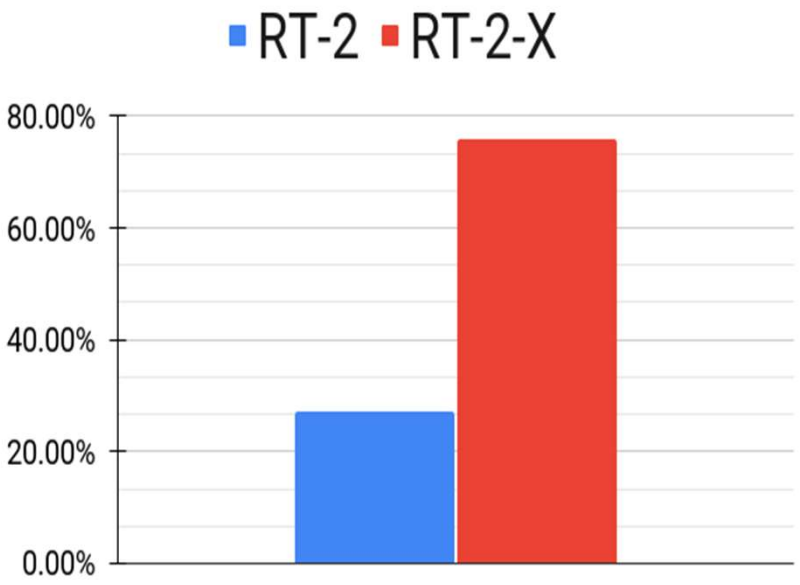



https://robotics-transformer-x.github.io/

# Results

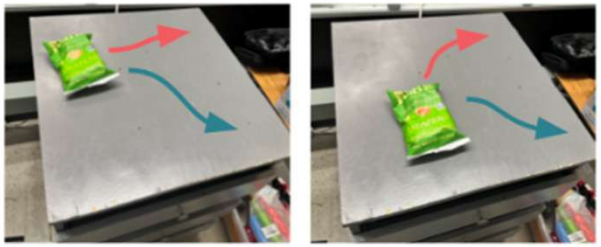

100
80
60
40
20
0
CLVR
RAIL
CILVR
AUTOLAB
AiS
Autonomous Intelligent Systems
43
48
63
24
18
56
53
65
80
53
25
45
33
68
72
41
44
63
Multi-robot & Multi-dataset
Kitchen Manipulation
Cable Routing
NYU Door Opening
Autolab UR5
Task-Agnostic Play
Mean
Original Method
MVP BC-RNN
Resnet + MLP
VINN
TACORL, HULC2
RT-1
RT-1-X

# Results

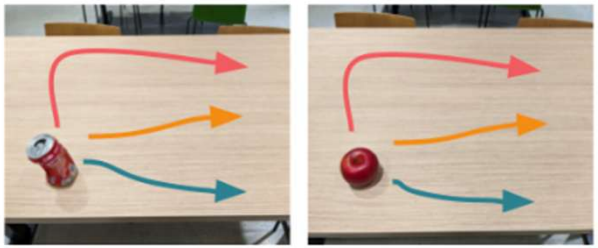



**(a) Absolute Motion**

*move the chip bag to the top / bottom right of the counter*

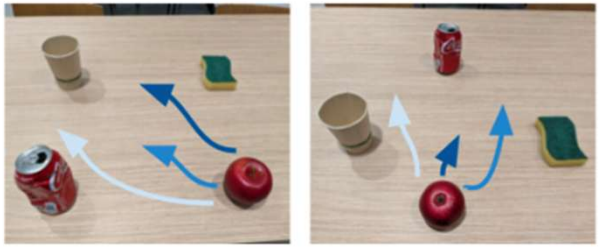


*move to top right / right / bottom right*



**(b) Object-Relative Motion**

*move apple between coke and cup / coke and sponge / cup and sponge*

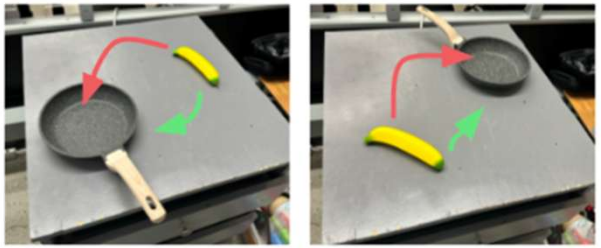


**(c) Preposition Alters Behavior**

*put apple on cloth / move apple near cloth*

*put orange into the pot / move orange near pot*

*put banana on top of the pan / move banana near pan*



# π0: A Vision-Language-Action Flow Model for General Robot Control

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, Ury Zhilinsky

Physical Intelligence

https://www.physicalintelligence.company/blog/pi0

# Motivation

- Large-scale models have been very effective at becoming generalists, but they are not grounded in the physical world
- It would be great if robot policy could also be trained in a generalist way:
  - Train on highly diverse, large scale data
  - Fine-tune or prompt for specific task
- Challenges:
  - There needs to be enough diverse data for pre-training
  - What is the right model architecture?
  - What is the right training procedure?

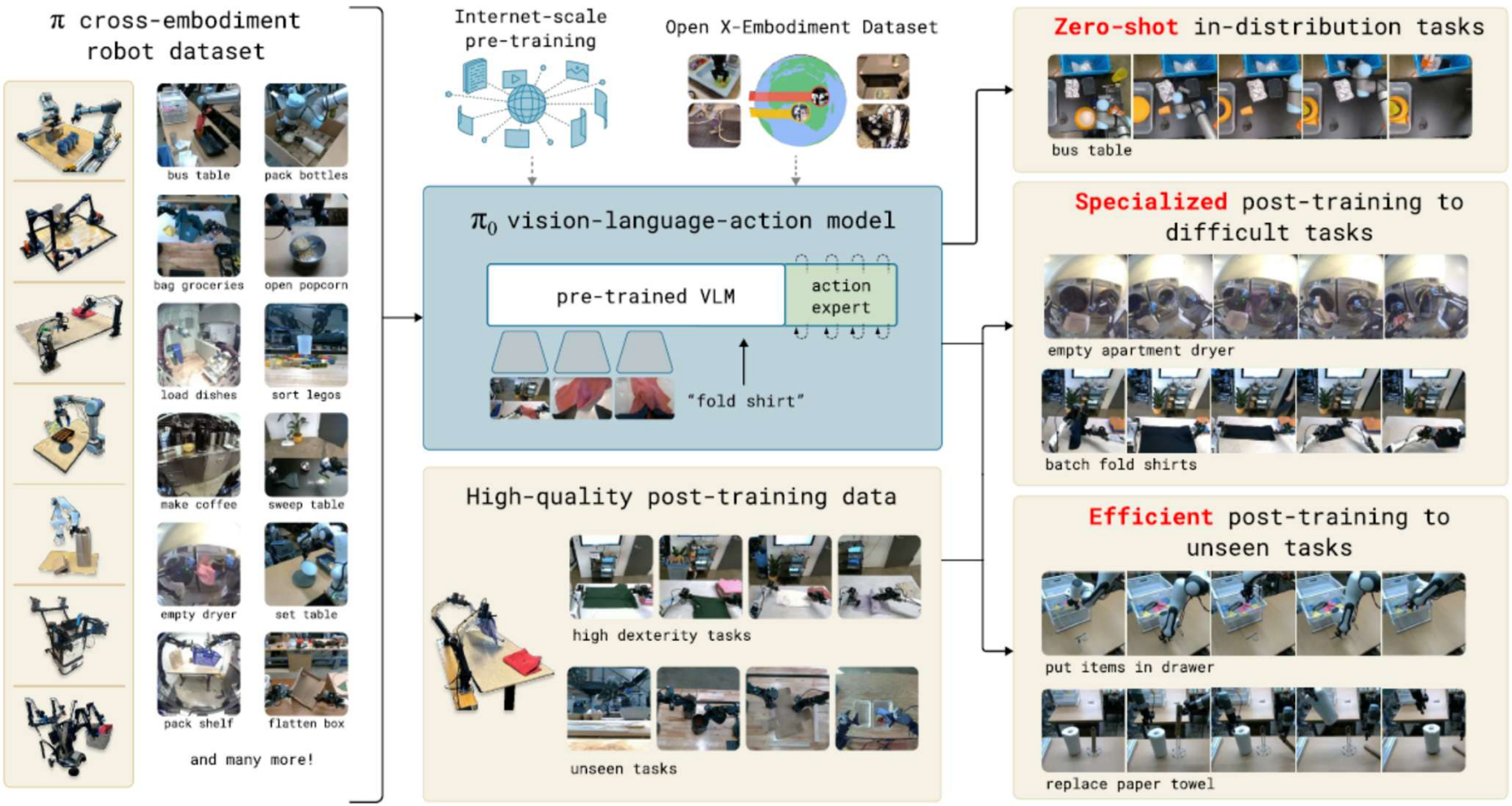



Fig. 1: Our generalist robot policy uses a pre-trained vision-language model (VLM) backbone, as well as a diverse cross-embodiment dataset with a variety of dexterous manipulation tasks. The model is adapted to robot control by adding a separate *action expert* that produces continuous actions via flow matching, enabling precise and fluent manipulation skills. The model can then be prompted for zero-shot control or fine-tuned on high-quality data to enable complex multi-stage tasks, such as folding multiple articles of laundry or assembling a box.

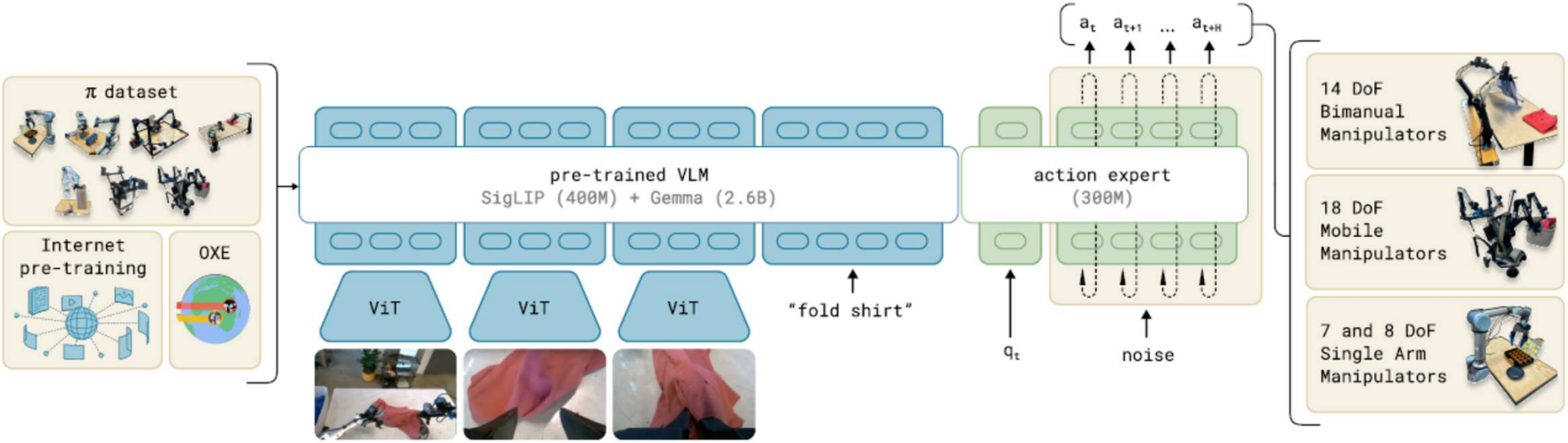



Fig. 3: **Overview of our framework.** We start with a pre-training mixture, which consists of both our own dexterous manipulation datasets and open-source data. We use this mixture to train our flow matching VLA model, which consists of a larger VLM backbone and a smaller *action expert* for processing robot states and actions. The VLM backbone weights are initialized from PaliGemma [5], providing representations learned from large-scale Internet pre-training. The resulting $\pi_0$ model can be used to control multiple robot embodiments with differing action spaces to accomplish a wide variety of tasks.

# Data Collection and Training

- Pre-training: combination of open-source, private datasets with a large variety of robot embodiments and tasks
- Post-training: fine-tuning with smaller task-specific dataset

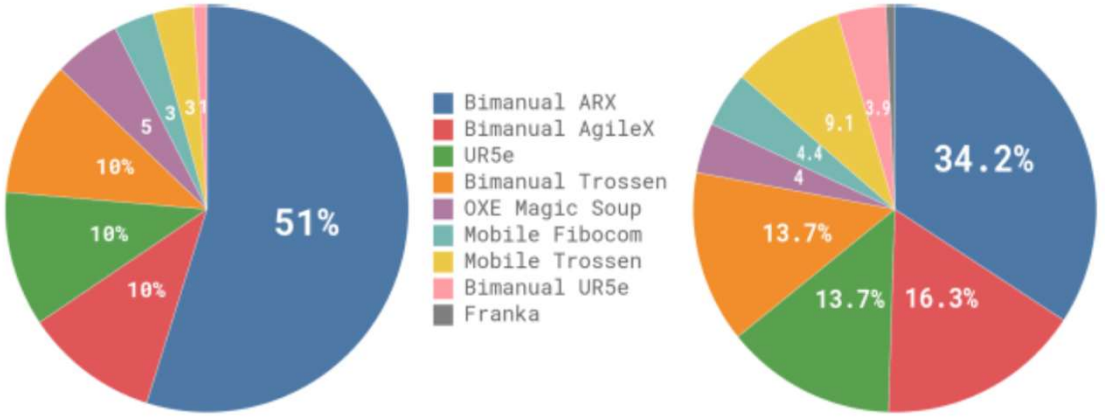



Fig. 4: **Overview of our dataset**: The pre-training mixture consists of a subset of OXE [10] and the $\pi$ dataset. We use a subset of OXE, which we refer to as OXE Magic Soup [24]. The right figure illustrates the weight of the different datasets in the pre-training mixture. The left figure illustrates their relative sizes as measured by the number of steps.

# Experiments

- How well does pi0 perform after pre-training on seen tasks?
- How well does pi0 follow language commands?
- How does pi0 compare to task-specific policies?
- Can pi0 be used for complex, multi-stage tasks?

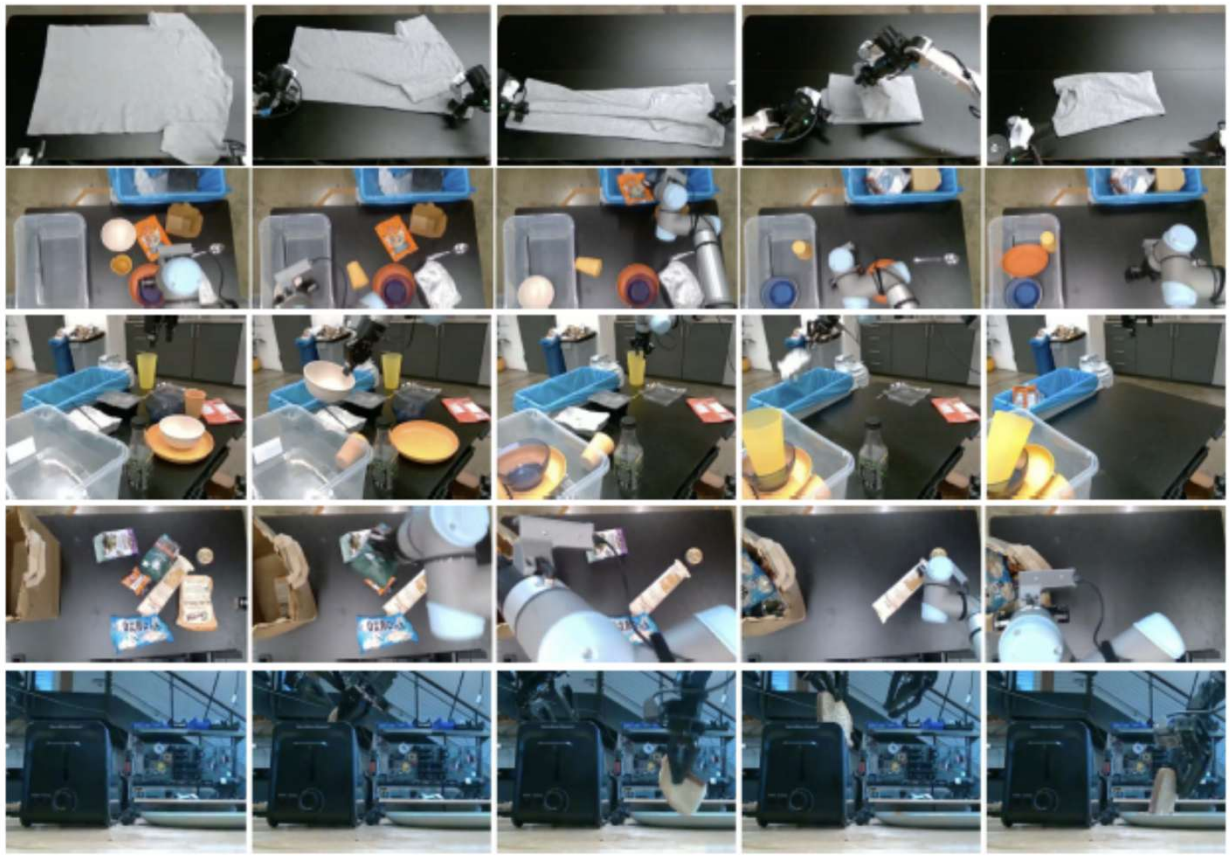


Fig. 6: **Zero-shot evaluation tasks**: To evaluate our base model, we run it after pre-training on five tasks: **shirt folding**, **bussing easy**, **bussing hard**, **grocery bagging**, and **toast out of toaster**. The tasks require a combination of dexterous manipulation, multi-stage behaviors, and semantic recognition.

https://www.physicalintelligence.company/blog/pi0

# All-in on additional data collection

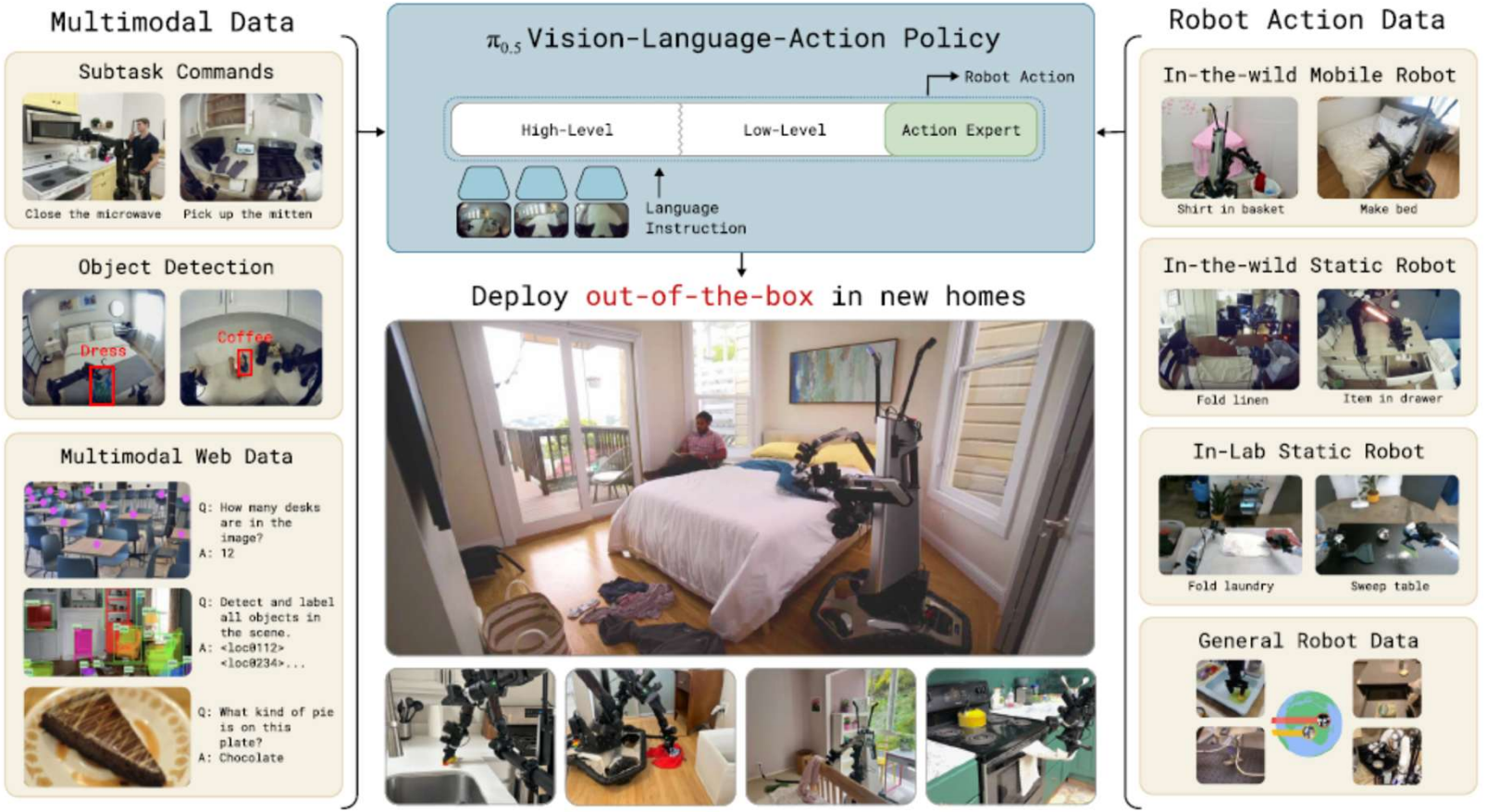



Fig. 1: The $\pi_{0.5}$ model transfers knowledge from a heterogeneous range of data sources, including other robots, high-level subtask prediction, verbal instructions, and data from the web, in order to enable broad generalization across environments and objects. $\pi_{0.5}$ can control a mobile manipulator to clean kitchens and bedrooms in new homes that were not present in the training data, performing complex multi-stage behaviors with durations of 10 to 15 minutes.

- https://www.pi.website/blog/pi05

# Combining IL with RL

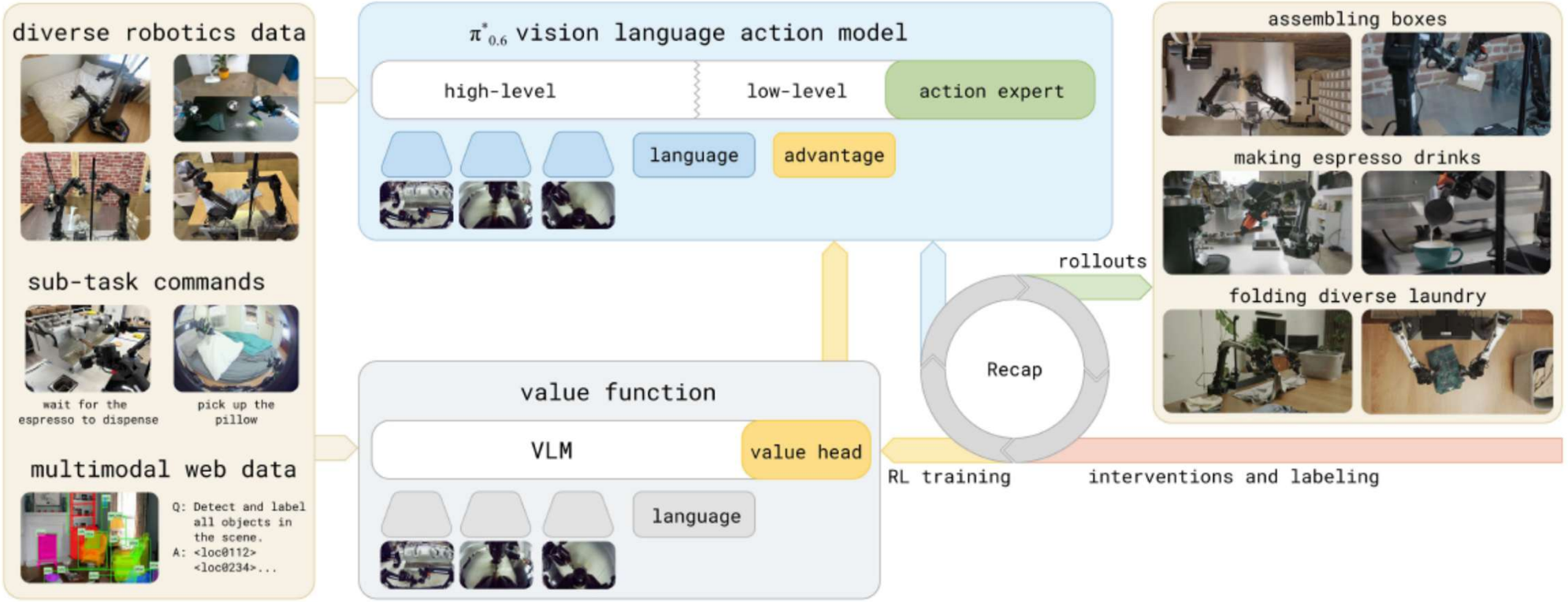



Fig. 1: **RECAP enables training VLAs with reward feedback and interventions.** Our system starts with a pre-trained VLA that incorporates *advantage conditioning*, allowing the model to learn effectively from real-world experience. For each task, we deploy the model and collect both autonomous rollouts and online human corrections. We then fine-tune the value function on this online data, improving its estimates of how actions influence performance. Fine-tuning and conditioning the VLA on these updated advantage estimates in turn improves policy behavior.

- https://www.pi.website/blog/pistar06

# Summary of today's session

- Improved models and network architecture for behavioural cloning improve performance:
  - More dexterous action
  - Better generalisation in perception
  - Combining visual and language prompts
  - Handling variations in strategy
- (Even more) reliant on large-scale, expensive data collection
- Still open challenges with generalisation, robustness, safety

# Summary and Outlook

Open Questions:

- Better and less effortful ways of collecting demonstration/expert data
  - Assessing quality before learning (Sakr et al., THRI 2025)
  - Guiding users to become better demonstrators (Sakr et al., THRI 2025)
- More efficient use of the data we have
  - Representation for action, e.g., FAST, BEAST
- More efficient use of prior knowledge?
  - Physics
  - Simulations
  - But are these just sporks?
- On-line, continuous learning?
  - From collaborative execution?

# Thank you!

Thank you for listening!

Thanks to all my students and colleagues in ECE6178/6188 and the robotics reading group!