

RICL: Adding In-Context Adaptability to Pre-Trained Vision-Language-Action Models

Anonymous Author(s)

Affiliation

Address

email

1 **Abstract:** Multi-task “vision-language-action” (VLA) models have recently
2 demonstrated increasing promise as generalist foundation models for robotics,
3 achieving non-trivial performance out of the box on new tasks in new environments.
4 However, for such models to be truly useful, an end user must have easy means to
5 teach them to improve. For language and vision models, the emergent ability to
6 perform in-context learning (ICL) has proven to be a versatile and highly useful
7 interface to easily teach new tasks with no parameter finetuning. Unfortunately,
8 VLAs pre-trained with imitation learning objectives do not naturally acquire ICL
9 abilities. In this paper, we demonstrate that, with the right finetuning recipe and a
10 small robot demonstration dataset, it is possible to inject in-context adaptability
11 *post hoc* into such a VLA. After retraining for in-context learning (RICL), our sys-
12 tem permits an end user to provide a small number (10-20) of demonstrations for
13 a new task. RICL then fetches the most relevant portions of those demonstrations
14 into the VLA context to exploit ICL, performing the new task and boosting task
15 performance. We apply RICL to inject ICL into the π_0 -FAST VLA, and show that it
16 permits large in-context improvements for a variety of new manipulation tasks with
17 only 20 demonstrations per task, without any parameter updates. When parameter
18 updates on the target task demonstrations is possible, RICL finetuning further boosts
19 performance. We release code and model weights for RICL- π_0 -FAST alongside
20 the paper to enable, for the first time, a simple in-context learning interface for new
21 manipulation tasks¹.

22 **Keywords:** Vision-Language-Action (VLA) models, In-Context Learning (ICL),
23 Retrieval-Augmented Generation (RAG)

24 1 Introduction

25 Robot learning is undergoing a transformative moment with the emergence of the first generation of
26 general-purpose Vision-Language-Action (VLA) models, capable of performing a wide spectrum of
27 robotic tasks — a development with profound practical implications. Such models [1, 2, 3, 4, 5, 6, 7, 8]
28 could address persistent challenges in robotics, including data scarcity, robustness, and generalization.

29 A natural point of comparison for these VLAs is large language models (LLMs). One important
30 factor in the widespread adoption of LLMs today is that they appear to be able to quickly learn new
31 tasks, simply through providing a few examples as “context” alongside the query, with no parameter
32 tuning. This capability, called in-context learning (ICL) [9], emerges naturally in LLMs pre-trained
33 for next-token prediction, due to the nature of web text data. Even better, one need not even manually
34 provide these few examples. Instead, a retrieval mechanism could automatically fetch the most
35 relevant data from a large corpus and place them into the LLM context. This retrieval-augmented
36 generation (RAG) mechanism is widely adopted as a versatile interface to improve a base LLM [10].

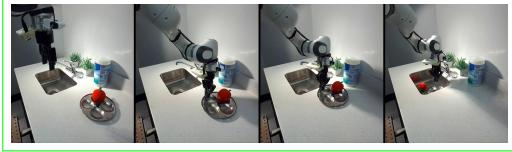
¹ Website: <https://ricl-vla.github.io>



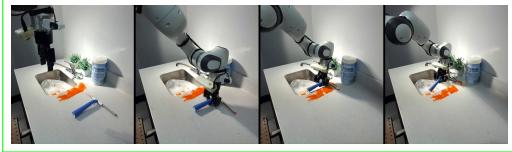
(a) Task: "pick up the poke ball and put it in the tray". π_0 -FAST-DROID [L] picks up the distractor (duck) instead (language grounding issue). RICL- π_0 -FAST-DROID [R] actually moves the unseen object (pokeball) with only RAG and ICL.



(b) Task: "pick up the bagel and put it in the toaster". π_0 -FAST-DROID [L] aimlessly wanders and cannot figure out the grasp or motion (adaptation issue). RICL- π_0 -FAST-DROID [R] almost completes the task (only with RAG and ICL) but drops the unseen object (bagel) at the end of the novel motion—a combination of an unfamiliar grasp at its rim, its unique initial vertical position, and the twist-and-lift motion.



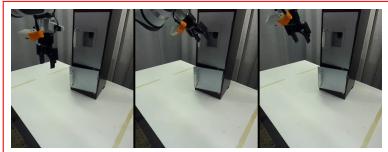
(c) Task: "move the idli plate to the sink". In π_0 -FAST-DROID's best test rollout shown here, it still struggles with the grasp and motion for this novel object (adaptation issue) or moves the apple (distractor) instead (language grounding issue). RICL- π_0 -FAST-DROID can perform the novel motion (gripper in depressions) on the unseen object (idli plate) in this new scene with new camera positions/orientations and with lighting changes (which is different from the table where the priming demonstrations were collected).



(d) Task: "use the squeegee to clean the counter". π_0 -FAST-DROID oscillates without success. It gets close, but it cannot figure out the grasp or the motion (adaptation issue). RICL- π_0 -FAST-DROID adapts to novel object (squeegee) and motion (part lifting, part dragging) in the new scene. Notice the pellets dropping into sink showing contact with the surface.



(e) Task: "push the lever on the toaster". π_0 -FAST-DROID [L] aimlessly wanders. It cannot figure out the precise location or the grasp (adaptation to a variant of training object issue). RICL- π_0 -FAST-DROID [R] completes the precise task, only with RAG and ICL, and with elicited latent actions not in the retrieval data (more information in Section 6). This long-tail task appears infrequently in the DROID dataset.



(f) Task: "open the door of the bottom shelf". π_0 -FAST-DROID [L] aimlessly wanders. It cannot figure out the motion to adjust or avoid the top door acting as obstacle (adaptation issue). RICL- π_0 -FAST-DROID [R] completes the task, with this variant of a seen object (this particular shelf) and novel motion (precise door opening adjusting for the obstructing top shelf), only with RAG+ICL. This is also a long-tail task.

Figure 1: Qualitative comparison between π_0 -FAST-DROID [L] and RICL- π_0 -FAST-DROID [R], with 20 task specific demonstrations for RAG and ICL, on new tasks, including novel objects, motions, and scenes. Additional comparisons can be found in Figure 8 in Appendix D.

37 Unfortunately, VLAs are trained with imitation learning objectives on relatively narrow demonstration
 38 datasets. As one would expect, this does not naturally produce any in-context learning abilities. This
 39 means that “improving” a pre-trained VLA today means tuning its parameters on a new demonstration
 40 dataset [2]. To make it possible for an end user to easily improve a VLA with no parameter tuning,
 41 we ask the following question:

42 *How can we inject in-context learning abilities into a pre-trained VLA?*

43 Once this is done, we should be able to painlessly boost the VLA’s performance on any task, including
44 handling unseen objects, novel motions, and new scenes that don’t exist in the VLA training data.
45 Our solution is to *retrain* for **in-context learning** (RICL, pronounced “rickle”). RICL borrows from
46 prior recipes [11, 12] to train generalist models for in-context learning and RAG. In particular,
47 while REGENT [12] trained generalist game-playing agents from scratch, RICL uses this approach
48 to instead post-train an off-the-shelf VLA priming it to use its context effectively. The resulting
49 RICL-VLA can improve the base VLA’s performance for any target tasks without a single gradient
50 update, instead adapting purely through retrieval-augmentation and in-context learning, with only
51 10-20 demonstrations in its retrieval buffer. We demonstrate this on various manipulation tasks
52 depicted in Figure 1 where a state-of-the-art (SOTA) VLA fails but our RICL-VLA adapts simply via
53 RAG and ICL. We further find that it is possible to get even better task performance by “finetuning
54 like you pretrain” [13]: we optimize the RICL objective on the same demonstrations as used above
55 for ICL, and get large performance boosts.

56 2 Related Work

57 **Training VLAs and multi-task generalist agents:** There has been a spate of work in recent years
58 on training multi-task agents in simulated settings like games [14, 12, 15, 16, 17, 18] and in recent
59 months on training VLAs for robotics [1, 2, 3, 4, 6, 7, 5, 8]. To our knowledge, there are only three
60 prior attempts to train general agents with in-context learning (ICL) capabilities [16, 12, 18], none
61 for general-purpose robotics. This is the focus of RICL: we show how to post-train a pre-trained VLA
62 to effectively learn in-context.

63 **In-context learning for robotics:** The in-context learning abilities of large language models (LLMs)
64 and vision-language models (VLMs) have already been found to be very useful in robotics: with
65 suitable representations (such as keypoints or code), these models can in-context learn imitation
66 policies [19, 20, 21, 22] or value functions [23, 24]. But, using LLMs and VLMs requires these
67 methods to run completely (or mostly) open-loop [19, 20, 21] and using only LLMs makes them lose
68 significant visual information [19, 21]. Both these drawbacks affect their ability to adapt.

69 3 Background on ICL, RAG, and π_0 VLAs

70 **In-Context Learning (ICL)** is the property of sequence to sequence models that allows them to
71 predict an output (such as an action \hat{a}_t) for a new input (such as state s_t) given a few examples of
72 input-output pairs in the context (such as state-action pairs $\{(s', a'), (s'', a''), \dots\}$). The input to the
73 model is the concatenation of the context and the new input.

74 **Retrieval-Augmented Generation (RAG)** refers to the strategy commonly used to help LLMs
75 predict an answer for a query. This is achieved by obtaining the information necessary for the answer
76 by searching through a dataset and then placing said information in the context of an LLM.

77 Of the three methods for embodied agents mentioned in Section 2 that learn to in-context learn,
78 MTT [16] and ICRT [18] place a few complete demonstrations in their context while REGENT
79 [12] retrieves select states and actions (from the same few complete demonstrations) to place in its
80 context. The RAG+ICL method employed by REGENT outperforms the former ICL method across
81 held-out games and held-out simulated robotics tasks. REGENT demonstrates that combining the
82 two i.e. retrieving specific examples to place into context from a demonstration dataset offers a
83 computationally less intensive, higher performing alternative to directly placing demonstrations in
84 context. We adopt this idea in RICL.

85 π_0 -**FAST** [1] is a state-of-the-art auto regressive VLA model that takes images, language instruction,
86 and proprioceptive state as input and predicts actions. It can be deployed (in a variety of scenes)
87 zero-shot on robot embodiments that are a part of its training data and few-shot after finetuning on
88 new robot embodiments. π_0 -**FAST-DROID** [1] is a VLA that was created by further finetuning
89 π_0 -FAST on the large DROID dataset [25]. The DROID dataset was collected with the Franka
90 DROID platform, shown in Figure 3, across many research labs (additional details in Appendix B).

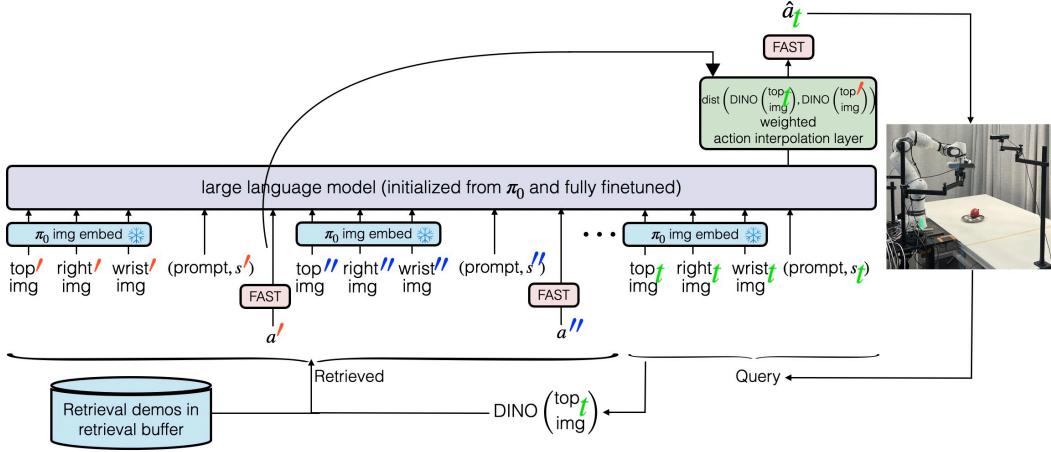


Figure 2: Architecture of RICL-VLAs, specifically that of RICL- π_0 -FAST.

91 4 RICL and creating in-context learning capable RICL-VLAs

92 This work aims to combine the best of both worlds from Section 3—*i.e.*, it aims to quickly convert a
 93 VLA that can be generally deployed like π_0 -FAST-DROID into one that also has in-context adaptation
 94 capabilities like REGENT. Once an in-context learning capable VLA has been created, “teaching” it
 95 to improve its performance on a new task is as simple as downloading the model, collecting a few
 96 demonstrations, and providing them as a retrieval dataset. Then, the in-context learning capable VLA
 97 should instantly have much better success rates on this new task than the baseline VLA.

98 **Re-training for In-Context Learning (RICL):** RICL enables the aforementioned conversion of a
 99 pre-trained VLA to a in-context learning capable-VLA (that we call a RICL-VLA). In RICL, a VLA
 100 is post-trained on sequences of query images/states and many images, states, actions, and action
 101 obtained from the retrieval demonstrations as depicted in Figure 2. The query information at time
 102 t consists of three images (top image $_t$, side image $_t$, wrist image $_t$), a language prompt describing
 103 the task, and proprioceptive state s_t . We use the term “query” following terminology from RAG for
 104 LLMs. The retrieved neighbors also consist of three images, the same text prompt, proprioceptive
 105 state, and action chunk (*i.e.* an array of actions over many time steps). The retrieved information is
 106 placed in the context with the closest neighbor (to the query) on the left and farther away neighbors
 107 towards the right. The closest neighbor’s images, states, and actions is represented with a single $/$,
 108 the second closest with a double $//$ and so on (see Figure 2). This finetuning utilizes a few “priming”
 109 demonstrations. These demonstrations are called “priming” demonstrations since their role is to prime
 110 the VLA to use its context effectively. Further, as depicted in Figure 2, only the LLM is finetuned
 111 during RICL while the image encoder is kept frozen. RICL-VLAs perform retrieval by embedding
 112 only the top query image with an off-the-shelf DINO-v2 [26] image encoder and comparing it with
 113 the embeddings of top images of the demos in the retrieval buffer with an ℓ_2 distance metric.

114 Like REGENT [12], the predicted action \hat{a}_t involves a distance-weighted interpolation between the
 115 action tokens of the closest retrieved action a' and the final output of the large language models. We
 116 refer to this as the action interpolation layer and depict it within the green box above the LLM in
 117 Figure 2. This distance corresponds to the distance between the DINO embeddings of the query
 118 top image and the closest retrieved top image. The action interpolation layer assumes a maximum
 119 number of action tokens numbering N_{act} and combines the one-hot encoding of each token of a' with
 120 the corresponding token output by the LLM π_θ (retrieved, query) as follows:

$$\pi_{RICL-VLA}^\theta(\text{retrieved}, \text{query}) = e^{-\lambda d} \text{one-hot}(a') + (1 - e^{-\lambda d})\sigma(\pi_\theta(\text{retrieved}, \text{query})) \quad (1)$$

121 where σ represents the Softmax function and d denotes the ℓ_2 distance between the DINO embeddings
 122 of top image $_t$ and its nearest neighbor top image'. The RICL-VLA performs the above interpolation
 123 for each of the N_{act} tokens. These tokens are then detokenized by the FAST tokenizer to obtain an
 124 action chunk that can be executed on the robot.

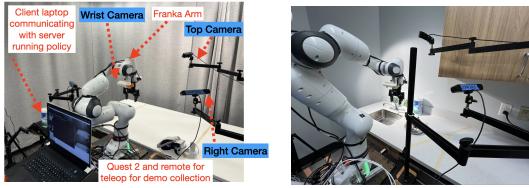
125 Unlike the process to train REGENT [12], RICL on the other hand, only predicts and minimizes the
126 cross-entropy loss over the query (prompt, s_t) tuple and predicted action chunk during training
127 whereas REGENT [12] predicted and minimized the loss over all retrieved and query actions.

128 The RICL-VLA, after having been primed to use its context in RICL, can now be deployed on a target
129 task, which can include unseen objects and novel motions, with just a few task-specific demonstrations
130 for RAG and ICL, and without any further training on those demonstrations.

131 **Further finetuning of a RICL-VLA:** If a RICL-VLA is further finetuned on those target task
132 demonstration—that it was only retrieving from and throwing into its context previously—it can
133 significantly improve its performance, outperforming a VLA directly vanilla finetuned on those
134 unseen task demonstrations. This finetuning process on the few task-specific demonstrations is
135 done exactly like RICL—*i.e.*, a retrieval-augmented finetune of the RICL-VLA (which has the action
136 interpolation layer) with the same objective of minimizing the cross-entropy loss over the query
137 (prompt, s_t) tuple and predicted action chunk. At deployment, the finetuned RICL-VLA still retrieves
138 from the same data that it is finetuned with, *i.e.* no extra data (hyperparameters in Appendix C).

139 5 Experimental Setup

140 **Training (*a.k.a.*, Priming) data for RICL:** We collect 20 demonstrations with the Franka DROID
141 platform (see Figure 3), randomizing the initial position of the primary object, in 20 pick and place
142 tasks (total 400 demonstrations). The exact list of tasks, an image of all the objects used, and
143 more details about the platform are in Appendix C. We perform RICL— starting from the weights
144 of π_0 -FAST-DROID, fully finetuning its LLM, and keeping its image encoder frozen—with the
145 hyperparameters detailed in Appendix C. We call the model obtained after three epochs of training as
146 RICL- π_0 -FAST-DROID.



153 Figure 3: [LEFT] Our Franka DROID setup, annotated.
154 [RIGHT] Franka DROID, including the top camera and
155 right camera, moved to a new scene (kitchen sink).

In each task, for each demonstration, and for each state in that demonstration, we use that state as the query and retrieve four neighbors from the other 19 demonstrations to create training input sequences. In this way, we end up training the model in the same way we would deploy the model. We collect data as detailed above since such a dataset is not available.

Evaluation tasks with unseen objects, novel motions, different scenes:

We evaluate all methods on the following tasks, which involve a task-relevant unseen object and/or a completely novel motion in two different scenes (tabletop and kitchen sink). “Unseen” here means that these objects and motions are not in either the RICL priming data or the DROID data [25]. We checked the latter by searching over the DROID dataset’s language annotations (and some recordings when language was not adequate). First, we start with a simple task that primarily tests language grounding.

- 162 • (pokeball) "pick up the pokeball and put it in the tray": has an unseen object (pokeball) with a couple of distractors on the table.

164 Next, we test on simple tasks that test both language grounding and adaptation to novel motions.

- 165 • (idliplate) "move the idli plate to the left": has an unseen object (idli plate) with a apple (distractor) sitting on the plate. It also requires the robot to do an unfamiliar grasp to move the uniquely shaped plate with depressions to the right.

- 168 • (squeegee) "move the squeegee to the right and try to drag it": has an unseen object (squeegee). It also requires the robot to do a novel motion of slightly lifting the handle of the squeegee while keeping its rubber on the table to drag it across the table.

171 We then test on versions of the above two simple tasks in a new scene (kitchen sink area) with new
172 camera positions/orientations (no calibration necessary), new lighting, and new distractors.

- 173 • (`sink-idliplate`) "move the idli plate to the sink": all of the previous challenges & a new scene.
 174 • (`sink-squeegee`) "use the squeegee to clean the counter": all of the challenges of the previous
 175 task with pellets to be cleaned up on the counter in the new scene.
- 176 We also test on the long-tail of the training task distribution. These tasks consist of variations of
 177 objects in the DROID dataset. These object classes appear infrequently in the dataset.
- 178 • (`toaster`) "push the lever on the toaster": has a different version of an object (a particular toaster
 179 brand) in the DROID dataset. This task tests the long tail of the training task distribution. It also
 180 requires a precise placement and movement to push the lever down.
- 181 • (`door`) "open the door of the bottom shelf": has a different version of an object (a particular shelf)
 182 in the DROID dataset. This task also is in the long tail. It requires a novel and precise motion that
 183 can handle the large top shelf acting as an obstacle when reaching the bottom door's handle.
- 184 Finally, we test on a longer horizon task that is a composition of many simple tasks.
- 185 • (`bagel1`) "pick up the bagel and put it in the toaster": has an unseen object (bagel). It also requires
 186 the robot to do a composite novel motion—an unfamiliar grasp on the edge of the bagel, the
 187 twist-and-lift motion, and placing in the slot.
- 188 **Retrieval data in evaluation tasks:** In all the evaluation tasks, we collect 20 demonstrations,
 189 randomizing the initial position of the primary object, for RICL- π_0 -FAST-DROID to retrieve from
 190 and throw in its context to adapt to the task.
- 191 **Evaluation metrics and comprehensive randomization:** We collect 10 test rollouts for all methods
 192 on all evaluation tasks with randomly chosen initial positions and orientations in each rollout. We
 193 set these intial positions and orientations all across the table. The distractors, if any, are kept
 194 approximately in the same region of the table but they are also not fixed in place. We calculate the
 195 success of the full tasks, in addition to tracking intermediate checkpoints for a better understanding
 196 of the progress of each method on each task.
- 197 **Baselines and ablations:** We compare RICL- π_0 -FAST-DROID with vanilla π_0 -FAST-DROID on
 198 all tasks. We also compare with 'Retrieve and play', a 1 nearest neighbor baseline from [12], which
 199 simply outputs the first retrieved action a' . We also compare with a trained-from-scratch Diffusion
 200 Policy baseline. We perform these two comparisons on the simpler evaluation tasks (`pokeball`,
 201 `idliplate`, `squeegee`). Upon observing their low success rates, we leave them out for the more
 202 complex tasks. In the tasks performed in a new scene (`sink-idliplate`, `sink-squeegee`), we aim
 203 to test RICL's ability to retain π_0 's helpful scene-generalization capabilities while adapting in-context
 204 to a new task and hence only test these two methods. We also ablate the number of demonstrations
 205 used by each method in the `idliplate` task.
- 206 **Further finetuning:** We further finetune RICL- π_0 -FAST-DROID on each evaluation task on the 20
 207 demonstrations collected for retrieval. For comparison, we also further (vanilla) finetune π_0 -FAST-
 208 DROID on these same 20 demonstrations in each task.

209 6 Experimental Evaluation

- 210 **Generalization to unseen objects and novel motions and new scenes:** We plot the quantitative
 211 results across tasks and methods in Figure 4. We observe that RICL- π_0 -FAST-DROID outperforms
 212 π_0 -FAST-DROID, especially in earlier checkpoints of the task, but also in overall task success. In
 213 aggregate, across all evaluated tasks, π_0 -FAST-DROID obtains a complete task success rate of 2.5%
 214 and a checkpoint completion rate of up to 21.25%. On the other hand, RICL- π_0 -FAST-DROID obtains
 215 a significantly improved complete task success rate of 31.25% and a checkpoint completion rate of
 216 up to 83.75%.
- 217 We particularly note that RICL- π_0 -FAST-DROID has significantly improved language grounding
 218 to move towards the unseen objects just based on contextual information. More importantly, RICL-

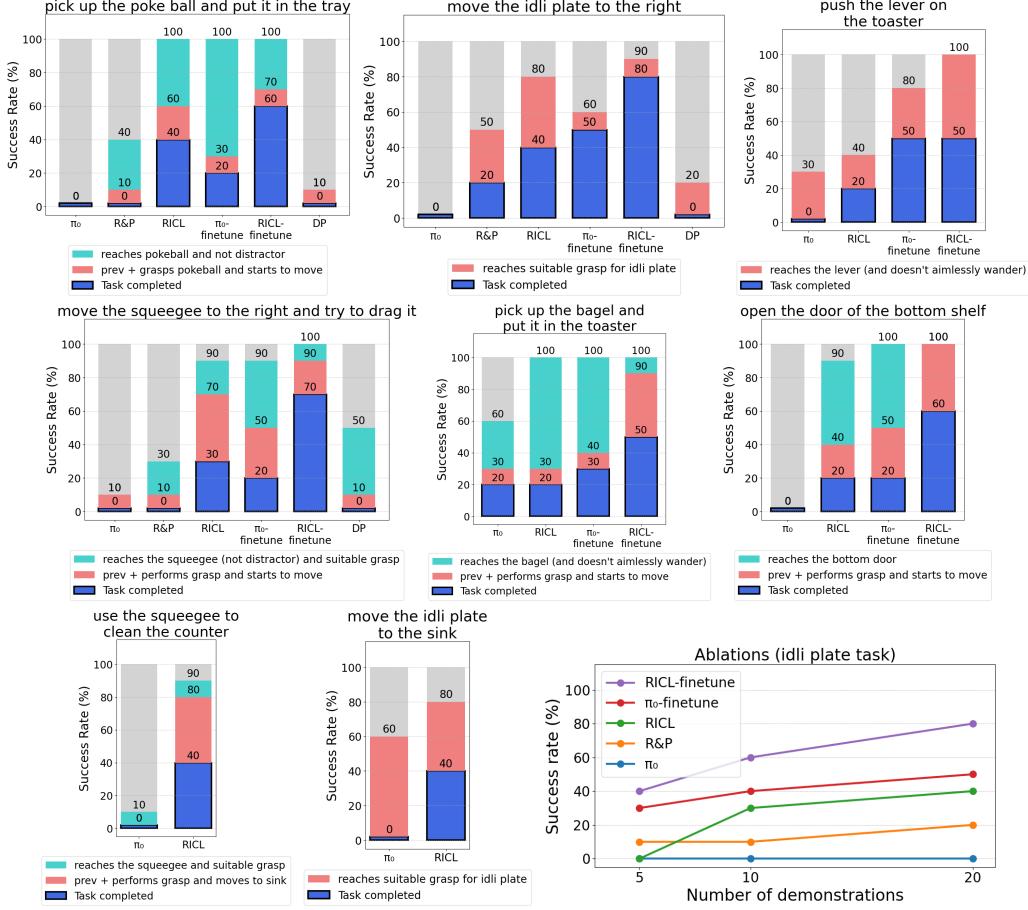


Figure 4: Success rates of 10 test rollouts from various methods across various tasks represented by stacked bar plots. The lowest bar (dark blue) in each stacked column represents full task success rate, and other bars are the success rates for reaching earlier waypoints. Gray regions represent the fraction of runs that did not even reach the first waypoint for the task. We note that π_0 refers to π_0 -FAST-DROID and RICL to RICL- π_0 -FAST-DROID in the plots. We also plot the performance of various methods vs the number of demonstration in the *idliplate* task on the bottom right.

219 π_0 -FAST-DROID also overcomes the adaptation issue faced by π_0 -FAST-DROID. Where π_0 -FAST-
220 DROID struggles with grasps and motions, RICL- π_0 -FAST-DROID demonstrates the ability to
221 infer novel grasps and motions from its context as evidenced in six tasks (both *squeegee* tasks,
222 *sink-idliplate*, *bagel*, *toaster*, and *door*). We plot the qualitative results depicting key test
223 rollouts and behaviors of π_0 -FAST-DROID and RICL- π_0 -FAST-DROID in Figure 1 and Figure 8. We
224 also provide side-by-side comparisons and detailed explanations of rollouts in the same Figures.

225 Unexpectedly, we observe in some tasks that RICL- π_0 -FAST-DROID seems to predict and execute
226 action sequences that are not like the motions in the retrieval dataset. For example, in *idliplate*,
227 RICL- π_0 -FAST-DROID moves to the left of the idli plate, closes its gripper and pushes the plate to
228 the right. But, all 20 demonstrations in the retrieval buffer were collected with the motion of dipping
229 the gripper into the depressions and moving the plate to the right. Hence, RICL- π_0 -FAST-DROID
230 has seemingly elicited latent actions or knowledge to accomplish this task (also seen in *toaster*).

231 **Significantly improved performance after further finetuning the RICL-VLA:** We observe a
232 significant improvement in performance after further finetuning RICL- π_0 -FAST-DROID on each
233 task’s 20 demonstrations. In aggregate, across all evaluated tasks, π_0 -FAST-DROID-finetuned
234 obtains a complete task success rate of 31.67%, while RICL- π_0 -FAST-DROID-finetuned obtains a
235 complete task success rate of 61.67%. In fact, we not only see that the RICL-VLA with finetuning is
236 significantly better than the base VLA with finetuning (at almost double the aggregate performance).

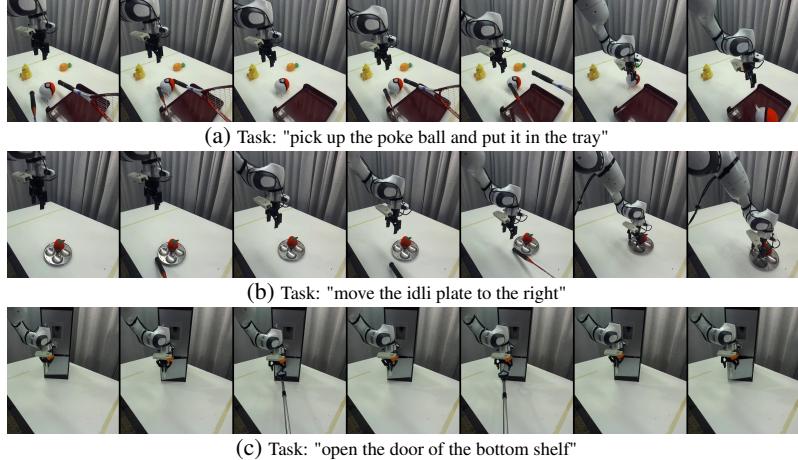


Figure 5: Qualitative visualization of the reactivity and robustness of RICL- π_0 -FAST-DROID-finetuned on 20 task-specific demonstrations in a dynamic test rollout. In the above, a human randomly perturbs and displaces the primary object during the test rollout. Additional results can be found in Figure 9 in Appendix D.

237 But we also observe comparable performance in complete task success rate, in aggregate, between
 238 the RICL-VLA (at 31.25%), which only uses RAG and ICL, and the base VLA with finetuning on the
 239 target task data (at 31.67%).

240 We hypothesize that further finetuning our VLA is significantly better than doing so with the base
 241 VLA simply because our VLA can use all of its capacity to focus on interpolating amongst the
 242 retrieved images, states, and actions to predict a new action and does not have to memorize any data.
 243 This is in line with the observed parameter-efficiency & performance advantages of RAG LLMs [27].

244 We qualitatively demonstrate the reactivity and robustness of RICL- π_0 -FAST-DROID by randomly
 245 perturbing and displacing objects during a test rollout as shown in Figure 5 and Figure 9.

246 **Ablating the number of retrieval/finetuning demonstrations:** We plot the ablation results, ablating
 247 the number of demonstrations used in the retrieval buffer or for finetuning, for *idliplate* in the
 248 bottom right of Figure 1. We found that too few demonstrations (such as 5) results in RICL- π_0 -
 249 FAST-DROID starting to behave like π_0 -FAST-DROID to the extent where in one test rollout, it
 250 too moves the apple instead of the idli plate. This demonstrates the requirement for atleast 10
 251 demonstrations. Also, from this figure, we conclude that more retrieval demonstrations help RICL-
 252 π_0 -FAST-DROID improve, towards catching up with π_0 -FAST-DROID-finetuned. We also see that
 253 RICL- π_0 -FAST-DROID-finetuned is significantly better than π_0 -FAST-DROID-finetuned at every
 254 number of demonstrations.

255 **No loss-of-capabilities results:** One might wonder: does RICL post-training come at the cost of
 256 losing the ability in the base VLA to perform without any retrieval data? To evaluate this, we test
 257 RICL- π_0 -FAST-DROID with randomly chosen priming demonstration in the retrieval buffer, rather
 258 than any meaningful task-specific demonstrations, on three tasks: "move the can to the tray", "pick
 259 up the marker and put it in the tray", and "place the apple next to the can". It obtains an 80% success
 260 rate, just like π_0 -FAST-DROID, demonstrating that RICL has not led to any loss of capabilities.

261 7 Conclusions and Future Work

262 We have demonstrated how RICL can be used to convert a VLA to a RICL-VLA that can use its
 263 context to adapt to completely new tasks, including unseen objects and novel motions, with just RAG
 264 and ICL. We found the RICL-VLA to even have comparable performance, in aggregate, with the base
 265 VLA finetuned on target task data. We have also demonstrated a significant boost in performance
 266 when a RICL-VLA is further finetuned on task-specific demonstrations. In future work, we believe
 267 that scaling up RICL in both the number of priming demonstrations and parameter size will further
 268 boost the performance of the in-context learning capable RICL-VLA.

269 **8 Limitations**

270 We find that RICL does not typically enable learning any tasks that are too far beyond those that the
271 base VLA can perform. This is why in this paper we primarily focus on pick and place tasks, the
272 primary strength of the π_0 VLAs used in this work. Further, while RICL-VLAs, unlike vanilla VLAs,
273 can adapt to new tasks in-context, they can still benefit from improved performance. We believe that
274 scaling up RICL in both the number of priming demonstrations/tasks and number of parameters can
275 help with this issue in future work. Another limitation of RICL-VLAs remains their need for a few
276 teleoperated demonstrations. Collecting these demonstrations for every new task across settings is not
277 scalable. We believe that videos of human demonstrations (such as in [20]) can help bridge this gap.
278 In our experiments, RICL- π_0 -FAST-DROID very easily removes the issue of language grounding for
279 new tasks and motions but struggles with generalizing to significantly novel motions such as playing
280 a forehand with a tennis racket. We believe future work including more diverse motions, rather than
281 just pick and place, in the RICL dataset can fix this issue.

282 **References**

- 283 [1] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and
284 S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint*
285 *arXiv:2501.09747*, 2025. (Cited on 1, 3, 13)
- 286 [2] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman,
287 B. Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv*
288 *preprint arXiv:2410.24164*, 2024. (Cited on 1, 2, 3, 13)
- 289 [3] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster,
290 G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv*
291 *preprint arXiv:2406.09246*, 2024. (Cited on 1, 3)
- 292 [4] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman,
293 C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*,
294 2024. (Cited on 1, 3)
- 295 [5] K. Bousmalis, G. Vezzani, D. Rao, C. Devin, A. X. Lee, M. Bauza, T. Davchev, Y. Zhou,
296 A. Gupta, A. Raju, et al. Robocat: A self-improving foundation agent for robotic manipulation.
297 *arXiv preprint arXiv:2306.11706*, 2023. (Cited on 1, 3)
- 298 [6] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al.
299 Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference*
300 *on Robot Learning*, pages 2165–2183. PMLR, 2023. (Cited on 1, 3)
- 301 [7] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh,
302 A. Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint*
303 *arXiv:2310.08864*, 2023. (Cited on 1, 3)
- 304 [8] G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakr-
305 ishna, R. Baruch, M. Bauza, M. Blokzijl, et al. Gemini robotics: Bringing ai into the physical
306 world. *arXiv preprint arXiv:2503.20020*, 2025. (Cited on 1, 3)
- 307 [9] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan,
308 P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan,
309 R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler,
310 M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever,
311 and D. Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL
312 <https://arxiv.org/abs/2005.14165>. (Cited on 1)
- 313 [10] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, and H. Wang. Retrieval-augmented
314 generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023. (Cited
315 on 1)
- 316 [11] Y. Gu, L. Dong, F. Wei, and M. Huang. Pre-training to learn in context. *arXiv preprint*
317 *arXiv:2305.09137*, 2023. (Cited on 3)
- 318 [12] K. Sridhar, S. Dutta, D. Jayaraman, and I. Lee. REGENT: A retrieval-augmented generalist
319 agent that can act in-context in new environments. In *The Thirteenth International Conference on*
320 *Learning Representations*, 2025. URL <https://openreview.net/forum?id=NxyfSW6mLK>.
321 (Cited on 3, 4, 5, 6, 13)
- 322 [13] S. Goyal, A. Kumar, S. Garg, Z. Kolter, and A. Raghunathan. Finetune like you pretrain:
323 Improved finetuning of zero-shot vision models. In *Proceedings of the IEEE/CVF Conference*
324 *on Computer Vision and Pattern Recognition*, pages 19338–19347, 2023. (Cited on 3)
- 325 [14] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez,
326 Y. Sulsky, J. Kay, J. T. Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*,
327 2022. (Cited on 3)

- 372 [31] M. Du, S. Nair, D. Sadigh, and C. Finn. Behavior retrieval: Few-shot imitation learning by
373 querying unlabeled datasets. *arXiv preprint arXiv:2304.08742*, 2023. (Cited on 13)
- 374 [32] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdul-
375 mohsin, M. Tschannen, E. Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv*
376 *preprint arXiv:2407.07726*, 2024. (Cited on 13)
- 377 [33] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-
378 training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages
379 11975–11986, 2023. (Cited on 13)
- 380 [34] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière,
381 M. S. Kale, J. Love, et al. Gemma: Open models based on gemini research and technology.
382 *arXiv preprint arXiv:2403.08295*, 2024. (Cited on 13)

383 **Appendix**

384 **A Additional Related Work**

385 We discuss some other relevant work below.

386 **Retrieval in robotics:** Recent work on retrieval in robot learning has created surprisingly adaptive
387 and reliable policies [28, 29, 30, 31]. Early work in VINN [29] and recent work in MCNN [28]
388 demonstrates the capabilities of retrieval based on image embeddings to improve behavior cloning
389 policies. Recent work in behavior [31] and flow retrieval [30] demonstrate the ability of retrieval to
390 augment task-specific data with a subset of offline data to further improve behavior cloning policies.
391 While RICL builds on the former use of retrieval, we believe future work that includes the latter use
392 of retrieval could further improve RICL-VLAs that are finetuned on task-specific demonstrations.

393 **B Additional Background Information**

394 We provide additional background information below.

395 **π_0 -FAST:** It was created by finetuning the PaliGemma Vision-Language model (VLM) [32] on a
396 (unknown but) large number of robot trajectories from different embodiments. The PaliGemma VLM
397 is a combination of a SigLIP image encoder [33] and the 3B parameters Gemma large language
398 model (LLM) [34]. In particular, it differentiates itself from the similar π_0 diffusion-based VLA [2]
399 with the use of the FAST tokenizer [1] to bring action tokens into the text token space for simple
400 autoregressive prediction.

401 **π_0 -FAST-DROID:** π_0 -FAST-DROID was created by finetuning π_0 – FAST on the DROID dataset
402 [25]. The DROID dataset was collected with the Franka DROID platform—which consists of a
403 Franka arm on a movable platform with cameras on the left, right, and wrist-in-the-wild, across
404 universities and scenes and with a large variety of objects. Each episode in the DROID dataset
405 also has proprioceptive states, actions, and language annotations. In this work, we use the DROID
406 platform (see Figures 3 and 2) for all results and hence, we will refer to this model frequently in the
407 rest of the paper.

408 **REGENT** [12], as discussed above, trains a transformer based multi-task policy on sequences of
409 query and retrieved state-action pairs, *from scratch*, on 145 simulated robotics tasks and games. Given
410 a few demonstrations in a held out simulated environment or game, REGENT retrieves state-action
411 pairs from these demonstrations, throws it into its context, and plays the held out game.

412 **C Additional Info on Franka DROID Setup, RICL, and**
413 **RICL- π_0 -FAST-DROID**

414 **Exact list of tasks for RICL:** We collect 20 demonstrations in each of the following 20 tasks (for a
415 total of about 400 demonstrations):

- 416 • 5x of "move <object> to the right" where object in [apple, orange, strawberry, the coffee
417 pod, the cup],
- 418 • 5x of "move <object> to the left" where object in [box, cup, duck, pan, pot],
- 419 • 6x of "pick up <object> and put it in the bowl" where object in [the block, the cloth, the
420 grapes on the waffle, the orange, the pineapple, the watermelon slice],
- 421 • 1x of "pour the contents of the mug into the bowl"
- 422 • 1x of "pick up the spoon in the bowl and put it in the other bowl"
- 423 • 1x of "put the cable in the bowl"
- 424 • 1x of "move the watermelon slice from one bowl to the other bowl"



Figure 6: [LEFT] Objects used during RICL, either as the primary object in the task or a distractor, to create RICL- π_0 -FAST-DROID. [RIGHT] Unseen objects and depictions of novel motions in red arrows. Unseen tasks also include distractors from the objects used during training.

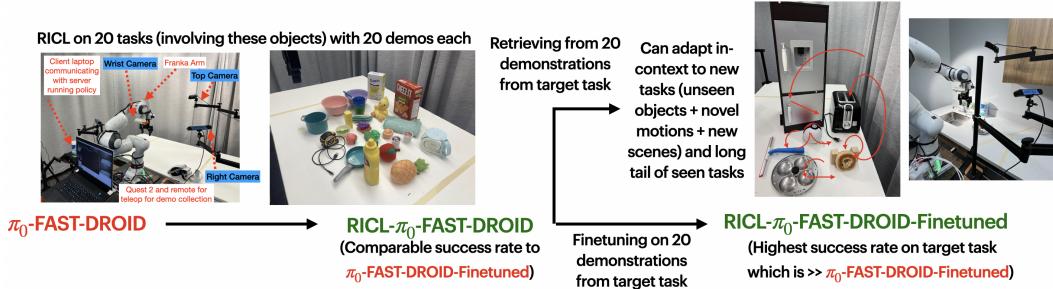


Figure 7: Overview of RICL.

the pictures of these objects and distractors used can be seen in Figure 6. We also depict the unseen objects and novel motions in the same Figure. We note that we collect data at an action frequency of 15 Hz.

Observation and action space: The proprioceptive observation consists of 7 joint angle and gripper position. The action chunks are of shape (15, 8) where 15 corresponds to the prediction horizon (which we empirically found to have better performance than 10). Each action has 7 joint velocities and 1 gripper position. All images are resized with padding to 224×224 .

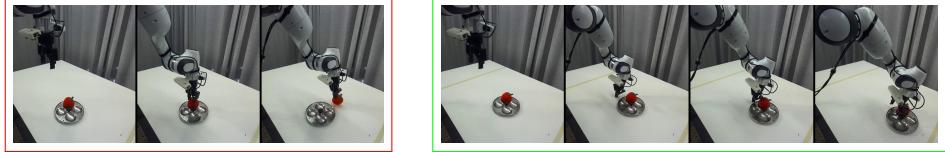
More details and key hyperparameters for RICL on priming demonstrations to create RICL- π_0 -FAST-DROID: We use four retrieved groups in our context and set $\lambda = 10$. For all other hyperparameters we build on top of the openpi repository² and make the following key changes: using action chunks of 15 steps, training for three epochs, with a CosineDecaySchedule with 300 warmup steps, 2.5e-5 peak learning rate, 3000 decay steps, and 2.5e-6 decay learning rate. We also use a batch size of 16 and two A100 GPUs. Our detailed codebase can be found in our website.

438 Key hyperparameters for further finetuning RICL- π_0 -FAST-DROID on each task's 20 demon-
439 strations: We use a recipe similar to RICL but only finetune for a 1000 steps, and with the same
440 learning rate scheduler but only for 1000 decay steps with 50 warmup steps.

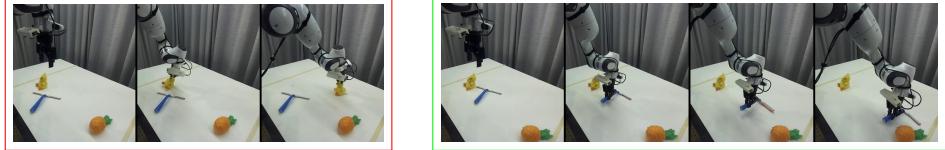
Key hyperparameters for further π_0 finetuning on each task's 20 demonstrations: We use the same number of train steps, decay steps and warmup steps in the optimizer as above but otherwise use the recommended pi0 finetuning recipe in the openpi repository.

Key hyperparameters at deployment: For RICL- π_0 -FAST-DROID, we execute 3 of the predicted 15 actions in the action chunk until a gripping action is predicted in the last action of an action

² <https://github.com/Physical-Intelligence/openpi>



(a) Task: "move the idli plate to the right". π_0 -FAST-DROID [L] moves the apple sitting on the plate (language grounding issue). RICL- π_0 -FAST-DROID [R] moves the unseen object (idli plate) with depressions that require an unfamiliar grasp, only with RAG and ICL. Further, it does so with elicited latent actions not in the retrieval data (see Section 6).



(b) Task: "move the squeegee to the right and try to drag it". π_0 -FAST-DROID [L] moves the distractor (duck) instead (language grounding issue). In other rollouts, it cannot figure out the grasp or the motion (adaptation issue). RICL- π_0 -FAST-DROID [R] moves the unseen object (squeegee) in a novel motion of partly lifting and partly dragging, only with RAG and ICL

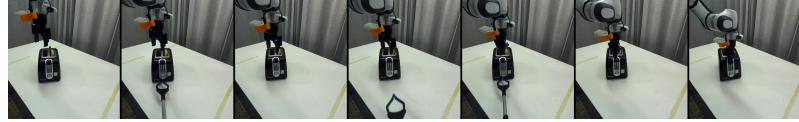
Figure 8: Additional qualitative comparisons between π_0 -FAST-DROID [L] and RICL- π_0 -FAST-DROID [R], with 20 task specific demonstrations for RAG and ICL, on new tasks, including novel objects, motions, and scenes.



(a) Task: "move the squeegee to the right and try to drag it"



(b) Task: "pick up the bagel and put it in the toaster"



(c) Task: "push the lever on the toaster"

Figure 9: Additional qualitative visualizations of the reactivity and robustness of RICL- π_0 -FAST-DROID-finetuned on 20 task-specific demonstrations in a dynamic test rollout. In the above, a human randomly perturbs and displaces the primary object during the test rollout.

446 chunk, at which point we switch to using 8 of the predicted 15 actions in an action chunk. Similarly,
447 for RICL- π_0 -FAST-DROID-finetuned, we execute 3 of the predicted 15 until a gripping action is
448 predicted, at which point we switch to using 5 of the predicted 15.

449 D Additional Results

450 We provide additional qualitative comparisons between π_0 -FAST-DROID and RICL- π_0 -FAST-
451 DROID in Figure 8. We also provide additional qualitative visualizations of the reactivity and
452 robustness of RICL- π_0 -FAST-DROID-Finetuned in Figure 9.

453 **Understanding the quality of retrieval (and hence the quality of the context):** To understand the
454 quality of retrieval, we depict retrieved top images and their corresponding side and wrist images
455 for key states in Figure 10 for the poke ball task. We find that our simple retrieval mechanism can
456 find similar states with close initial positions and RICL- π_0 -FAST-DROID interpolates amongst these
457 similar states and actions to predict a suitable action.

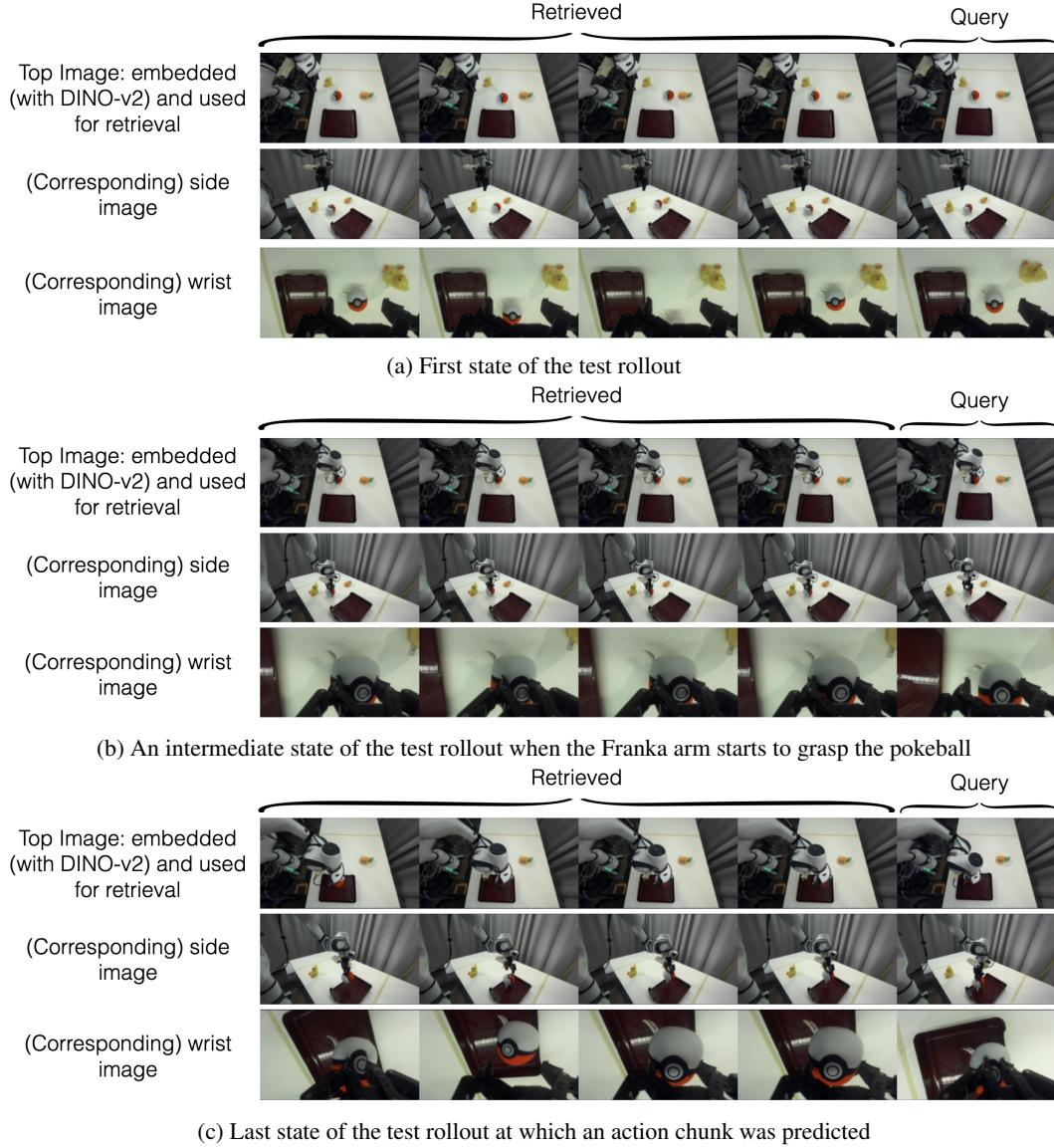


Figure 10: Visualization of Query and Retrieved images at three states during a RICL- π_0 -FAST-DROID test rollout on the pokeball task.