

## A Data Mixture Details

We list our used data mixture in Table 3. The mixture mostly follows [5], with a few additional datasets.

OpenVLA Training Dataset Mixture	
Fractal [83]	12.7%
Kuka [45]	12.7%
Bridge[6, 47]	13.3%
Taco Play [84, 85]	3.0%
Jaco Play [86]	0.4%
Berkeley Cable Routing [87]	0.2%
Roboturk [88]	2.3%
Viola [89]	0.9%
Berkeley Autolab UR5 [90]	1.2%
Toto [91]	2.0%
Language Table [92]	4.4%
Stanford Hydra Dataset [93]	4.4%
Austin Buds Dataset [94]	0.2%
NYU Franka Play Dataset [95]	0.8%
Furniture Bench Dataset [96]	2.4%
UCSD Kitchen Dataset [97]	<0.1%
Austin Sailor Dataset [98]	2.2%
Austin Sirius Dataset [99]	1.7%
DLR EDAN Shared Control [100]	<0.1%
IAMLab CMU Pickup Insert [101]	0.9%
UTAustin Mutex [102]	2.2%
Berkeley Fanuc Manipulation [103]	0.7%
CMU Stretch [104]	0.2%
BC-Z [55]	7.5%
FMB Dataset [? ]	7.1%
DobbE [105]	1.4%
DROID [11]	10.0% <sup>3</sup>

Table 3: OpenVLA training data mixture using datasets from the Open X-Embodiment dataset [1], following [5] with a few additions.

## B Evaluation Tasks and Detailed Results

In this section, we provide more details on the BridgeData V2 WidowX and RT-1 robot evaluations discussed in Section 4.1, as well as the Franka-Tabletop and Franka-DROID fine-tuning evaluations discussed in Section 4.2.

### B.1 Bridge V2 WidowX Evaluation Details

Here we focus specifically on BridgeData V2 evaluations discussed in Section 4.1.

#### B.1.1 Bridge V2 Evaluation Tasks

As described in Section 4.1, we evaluate each generalist robot manipulation policy on 17 tasks with 10 trials each. In this section, we provide details on the task categories and individual tasks.

In total, we evaluate on 5 visual generalization tasks, 2 motion generalization tasks, 3 physical generalization tasks, 4 semantic generalization tasks, and 3 language grounding tasks. Note that all tasks we evaluate on introduce some form of distribution shift since we are unable to procure the exact objects used in the original dataset (other distribution shifts naturally arise as we reproduce a real-world test environment originally constructed at a different location; see Appendix B.1.2 for a detailed discussion on such distribution shifts). All 17 tasks are depicted in Fig. 6. Each rollout is

<sup>3</sup>We remove DROID for the last third of training due to slow learning progress (see Section 3.3) and re-distribute its mixture weights across all other datasets.

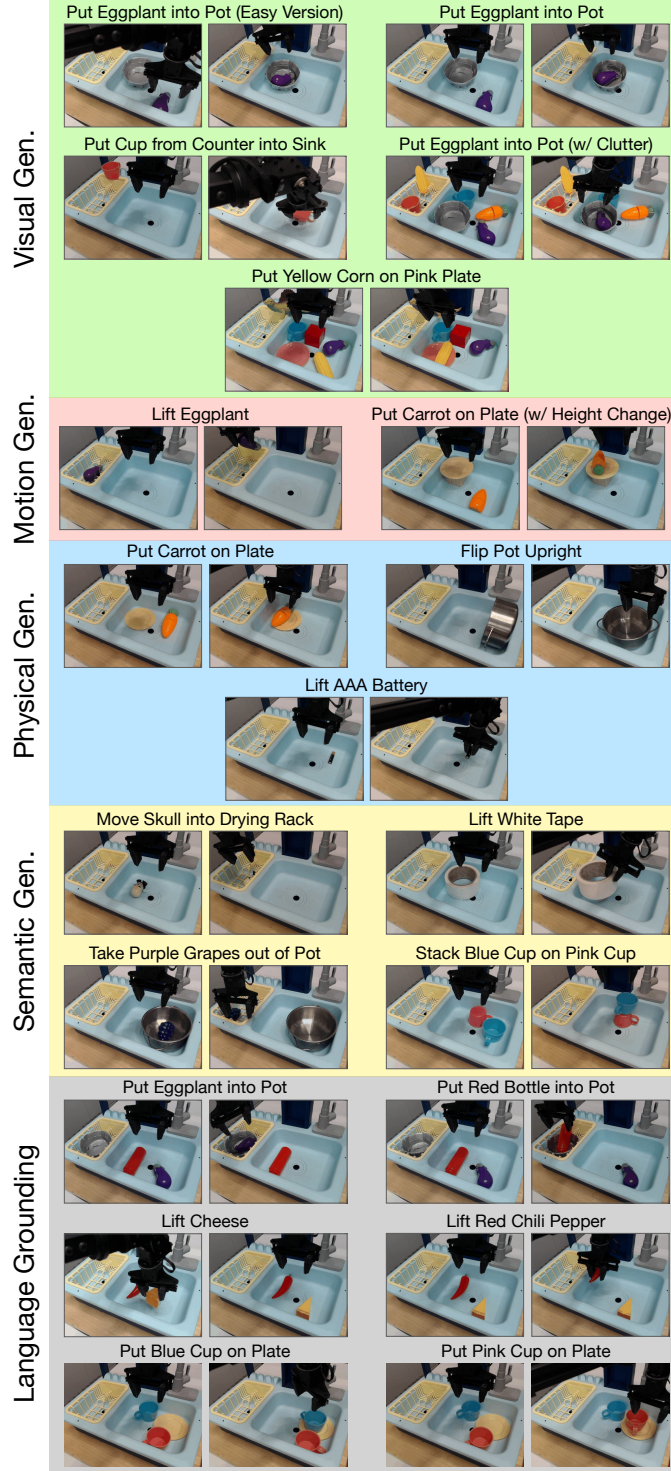


Figure 6: **BridgeData V2 WidowX robot evaluation tasks.** We evaluate every generalist robot policy on 4 types out-of-distribution (OOD) generalization tasks: **visual**, **motion**, **physical**, and **semantic** (as defined in [Section 4.1](#)). Every pair of images shows the start state and an example end state after the robot completes the task. We also rigorously assess **language grounding** in the 3 tasks shown in the bottom 3 rows, by changing the prompt while fixing the initial state and testing whether the policy can approach the correct target object.

marked as a failure (0) or success (1). In some more difficult tasks, we record partial successes (0.5); we describe the conditions for partial credit in the task descriptions below.

Below we describe each of the 17 tasks, in the order shown in Fig. 6:

1. **Put Eggplant into Pot (Easy Version):** The robot’s goal is to pick up the eggplant and drop it into the pot. This is a **visual generalization** task because we use a handcrafted paper pot that has a different appearance than the pot used in the original BridgeData V2 training dataset (since we are unable to procure the original pot). Unlike all 16 other tasks, for this particular task we initialize the robot’s end-effector directly above the eggplant before rolling out the policy; hence, we call this the “Easy Version” of the “Put Eggplant into Pot” task.
2. **Put Eggplant into Pot:** This is the same task as described above, except that the robot’s end-effector is not initialized directly above the eggplant. Instead, we initialize it in a position that is fixed across all rollouts, which means that the robot must horizontally reach for the eggplant first before manipulating it. (Note: The same applies to all other tasks described below.) This is a **visual generalization** task for the same reason as above.
3. **Put Cup from Counter into Sink:** The robot’s goal is to pick up the pink cup from either the kitchen countertop or drying rack and place it into the sink on the right. This is a **visual generalization** task because we use a pink cup rather than a blue cup (a blue cup is used in the original BridgeData V2 dataset, but we find that none of the methods we evaluate is able to manipulate it reliably – most likely because the color of the cup blends in with the color of the sink).
4. **Put Eggplant into Pot (w/ Clutter):** This is the same task as the “Put Eggplant into Pot” task, except that it is more difficult due to the presence of several distractor objects. It is a **visual generalization** task for the same reason discussed in the normal “Put Eggplant into Pot” task, and even more so given unseen distractors in the scene. *Partial credit (0.5 out of 1) is rewarded when the robot moves towards the correct target object.*
5. **Put Yellow Corn on Pink Plate:** The robot’s goal is to pick up the yellow corn and place it on the pink plate. This is a **visual generalization** task due to the presence of unseen distractor objects in the scene, such as a green dinosaur on the countertop in the back section of the sink. *Partial credit (0.5 out of 1) is rewarded when the robot moves towards the correct target object.*
6. **Lift Eggplant:** The robot’s goal is to grasp and lift the eggplant into the air. This is a **motion generalization** task because the eggplant is initialized in unseen positions and/or orientations, and the robot is forced to move beyond its training distribution of positions and/or orientations and often perform long-range reaching in order to complete the task. (Note: Long-range reaching is not demonstrated in this environment in the original BridgeData V2 demonstrations; see Appendix B.1.2 for details.) We find that this task, though seemingly simple, is deceptively challenging for many policies. *Partial credit (0.5 out of 1) is rewarded when the robot makes contact with the eggplant.*
7. **Put Carrot on Plate (w/ Height Change):** The robot’s goal is to pick up the carrot and place it on the yellow plate. This is a **motion generalization** task because the plate is elevated from its usual position at the bottom of the sink, and the robot must adjust its trajectory to correctly place the carrot on the elevated platform (without knocking down the plate in the process). *Partial credit (0.5 out of 1) is rewarded when the robot grasps the carrot and touches the plate with it.*
8. **Put Carrot on Plate:** This is the same task as above, except that the plate is at its normal position (at the bottom of the sink or drying rack). We consider this a **physical generalization** task because the carrot has a different size and shape than the one used in the original BridgeData V2 dataset, which is shorter and narrower. (Note that the previous version of this task listed above would also technically be a physical generalization task since it involves the same carrot, but we list it under the “motion generalization” category since that is the focus there.)

9. **Flip Pot Upright:** The robot’s goal is to manipulate the pot such that it is oriented upright in the sink at the end of the episode. This is a **physical generalization** task because this pot has a different size and shape than the one used in the original BridgeData V2 training demonstrations (the pot we use is wider and shorter).
10. **Lift AAA Battery:** The robot’s goal is simply to grasp the AAA battery and lift it up into the air. This is considered a **physical generalization** task because the battery is much smaller and thinner than target objects seen in the BridgeData V2 training demonstrations in this environment; see [Appendix B.1.2](#) for details. (Note that this target object does not exist in the original BridgeData V2 demonstrations in this environment, so this is also an instance of “semantic generalization”, but we classify it solely as “physical generalization” since that is the main focus here).
11. **Move Skull into Drying Rack:** The robot’s goal is to grasp the skull windup toy and drop it into the yellow drying rack in the left part of the sink. This is a **semantic generalization** task since the skull is an unseen target object (does not appear in the BridgeData V2 training demonstrations).
12. **Lift White Tape:** The robot’s goal is to grasp and lift the white roll of tape into the air. This is a **semantic generalization** task since the white tape roll is an unseen target object (does not appear in the BridgeData V2 training demonstrations). (Note that this task may also be considered as “physical generalization” because of its shape being different than the objects seen in the training demonstrations in this environment; most policies struggle to grasp objects with this ring structure, and they often move the robot’s end-effector directly into the center region.)
13. **Take Purple Grapes out of Pot:** The robot’s goal is to grasp the purple grapes lying inside the steel pot and remove it from the pot (by lifting it out and/or dropping it anywhere outside the pot). This is a **semantic generalization** task because it is an unseen language instruction; the robot has never seen this task in the original BridgeData V2 training dataset.
14. **Stack Blue Cup on Pink Cup:** The robot’s goal is to grasp the blue cup and place it securely on top of the pink cup. This is a **semantic generalization** task because it is an unseen language instruction; the robot has never seen this task in this environment in the original BridgeData V2 training dataset. *Partial credit (0.5 out of 1) is rewarded when the robot grasps the blue cup and touches the pink cup with the blue cup.*
15. **Put {Eggplant, Red Bottle} into Pot:** This is a **language grounding** task. The robot’s goal is to put the specified target object into the pot. Both the eggplant and red bottle are present in the scene. We conduct paired evaluations: for the same initial state, we prompt the policy to target the eggplant in one episode, and then the red bottle in the next episode. We test each method 5 times with the eggplant and 5 times with the red bottle, using the same set of 5 initial states for both target objects. *Partial credit (0.5 out of 1) is rewarded when the robot moves towards the correct target object.*
16. **Lift {Cheese, Red Chili Pepper}:** This is a **language grounding** task. The robot’s goal is to grasp and lift the specified target object. We conduct paired evaluations as described in the task above. *Partial credit (0.5 out of 1) is rewarded when the robot moves towards the correct target object.*
17. **Put {Blue Cup, Pink Cup} on Plate:** This is a **language grounding** task. The robot’s goal is to grasp the specified target object and place it onto the plate. We conduct paired evaluations as described in other language grounding tasks. *Partial credit (0.5 out of 1) is rewarded when the robot moves towards the correct target object.*

## B.1.2 Comparing Evaluation Tasks to Original Bridge V2 Training Data

We conduct our evaluations in a sink environment used in the original BridgeData V2 dataset [6]. We reproduce the environment to match the original environment in the BridgeData V2 dataset with rough approximations for the robot’s location relative to the sink, as well as the camera’s placement



relative to the scene. Given the lack of precise measurements of these positions in the original dataset, we are unable to reproduce the *exact* environment setup, and natural distribution shifts arise due to slightly different robot, sink, and camera placements. In addition, since we evaluate robot policies in a different location than where the training demonstrations were collected from, other natural distribution shifts arise. For example, the lighting conditions and background (e.g., visible areas behind the sink) are inevitably different than what was seen in the training dataset. Furthermore, we are unable to procure the exact set of objects used in the original BridgeData V2 dataset, so there are distribution shifts between the objects used at train time and those used at test time.

Despite all these challenges, we find that certain generalist policies, such as OpenVLA and RT-2-X, can still generalize and perform various tasks fairly reliably “out-of-the-box”. Other generalist policies, such as RT-1-X and Octo, can also complete some tasks, though they struggle when tested with more difficult generalization tasks in our BridgeData V2 evaluation suite.

The original BridgeData V2 dataset includes demonstrations of the following seven tasks in this specific sink environment: “Flip Pot Upright”, “Put Carrot on Plate”, “Put Cup from Counter (or Drying Rack) into Sink”, “Put Eggplant into Pot”, “Put Knife on Cutting Board”, “Put Spoon in Pot”, and “Turn Lever Vertical to Front”. See Fig. 7 for samples images of all these tasks from the original dataset. Note that all training demonstrations collected in this environment are initialized such that the robot’s end-effector is positioned directly above the target object in the beginning of the episode. (However, this is not the case across all environments in the BridgeData V2 dataset; in some other environments, the robot is initialized farther away from the target object, so it must horizontally reach for the object first before manipulating it.)



Figure 7: **Original BridgeData V2 sink environment tasks.** Images from sample demonstrations in the sink environment from the original BridgeData V2 dataset reveal that all demonstrations in this environment were initialized such that the robot’s end-effector was positioned immediately above the target object. Note that these initial states are different from the initial states we use in our BridgeData V2 evaluation tasks shown in Fig. 6. In our evaluations, we always initialize the robot’s end-effector to a fixed location above the sink, rather than positioning it directly above the target object (except for one task: “Put Eggplant into Pot (Easy Version)”).

In our BridgeData V2 evaluation suite, only one task – “Put Eggplant into Pot (Easy Version)” – is initialized with the robot’s end-effector hovering directly over the target object; in all 16 other tasks, the end-effector is initialized at a fixed location above the sink such that the robot must horizontally reach towards the object. This initial condition, in combination with the distribution shifts we introduce in the various types of OOD generalization in our evaluation suite, challenges the generalist policies and requires a high degree of robustness in order to complete the tasks successfully. Hence, the success rates for policies like RT-1-X and Octo are lower than what is reported in prior works. However, we find that other policies such as RT-2-X and OpenVLA still achieve relatively strong performance despite all these distribution shifts and challenges.

### B.1.3 Detailed Bridge V2 Evaluation Results

See Table 4 for the full BridgeData V2 WidowX evaluation results. The number of successes for each method, out of 10 trials, is listed for each of 17 tasks. OpenVLA achieves strongest performance in the majority of the tasks and has the highest aggregate success rate among the generalist policies. RT-2-X also shows good performance, outperforming RT-1-X and Octo, though it does not perform as well as OpenVLA. RT-1-X and Octo generally experience difficulty in these generalization tasks.

Table 4: **Detailed BridgeData V2 WidowX evaluation results.** We report performance on the full evaluation suite of 17 tasks (discussed in Section 4.1), including visual/motion/physical/semantic generalization tasks and language grounding tasks. Note that *partial success* (score of 0.5) is possible for some tasks; see Appendix B.1.1 for details. We find that OpenVLA performs best in most tasks and achieves highest performance overall, followed by RT-2-X. On the other hand, RT-1-X and Octo struggle in the evaluations, only getting 0–2 successes in several tasks. See Fig. 6 for illustrations of all tasks.

Category	Task	# Trials	RT-1-X # Successes	Octo # Successes	RT-2-X # Successes	OpenVLA (ours) # Successes
Visual gen	Put Eggplant into Pot (Easy Version)	10	1	5	7	10
Visual gen	Put Eggplant into Pot	10	0	1	5	10
Visual gen	Put Cup from Counter into Sink	10	1	1	0	7
Visual gen	Put Eggplant into Pot (w/ Clutter)	10	1	3.5	6	7.5
Visual gen	Put Yellow Corn on Pink Plate	10	1	4	8	9
Motion gen	Lift Eggplant	10	3	0.5	6.5	7.5
Motion gen	Put Carrot on Plate (w/ Height Change)	10	2	1	4.5	4.5
Physical gen	Put Carrot on Plate	10	1	0	1	8
Physical gen	Flip Pot Upright	10	2	6	5	8
Physical gen	Lift AAA Battery	10	0	0	2	7
Semantic gen	Move Skull into Drying Rack	10	1	0	5	5
Semantic gen	Lift White Tape	10	3	0	0	1
Semantic gen	Take Purple Grapes out of Pot	10	6	0	5	4
Semantic gen	Stack Blue Cup on Pink Cup	10	0.5	0	5.5	4.5
Language grounding	Put {Eggplant, Red Bottle} into Pot	10	2.5	4	8.5	7.5
Language grounding	Lift {Cheese, Red Chili Pepper}	10	1.5	2.5	8.5	10
Language grounding	Put {Blue Cup, Pink Cup} on Plate	10	5	5.5	8.5	9.5
Mean Success Rate			18.5±2.7%	20.0±2.6%	50.6±3.5%	70.6±3.2%

## B.2 RT-1 Robot Evaluation Details

In this section, we provide more details on the RT-1 robot evaluations introduced in Section 4.1.

### B.2.1 RT-1 Robot Evaluation Tasks

On the RT-1 robot, we evaluate each generalist robot policy on 12 tasks with 5 rollouts each, for a total of 60 rollouts. The first five tasks test on in-distribution conditions, and the last seven tasks test on more difficult out-of-distribution (OOD) conditions. All tasks are depicted in Fig. 8. Each rollout is marked as a failure (0) or success (1).

We describe the 12 tasks below:

1. **Pick Coke Can** (in-distribution): The robot is positioned in front of a platform with a can of Coke on top of it. The robot’s goal is to grasp and lift the Coke can.
2. **Move Apple near Green Can** (in-distribution): The robot is positioned in front of a platform with an apple and a green soda can on top of it. The robot’s goal is to grasp the apple and move it next to the green can.
3. **Move Blue Chip Bag near Apple** (in-distribution): The robot is positioned in front of a platform with a blue bag of chips and an apple on top of it. The robot’s goal is to grasp the blue bag of chips and move it close to the apple.
4. **Place Coke Can Upright** (in-distribution): The robot is positioned in front of a platform with a can of Coke on top of it, and the can is oriented horizontally on its side. The robot’s goal is to grasp the Coke can and orient it to be in a vertical position.
5. **Open Middle Drawer** (in-distribution): The robot is positioned in front of a set of three drawers. The robot’s goal is to grasp the middle drawer handle and pull the drawer open.
6. **Move Orange near Brown Chip Bag** (OOD): The robot is positioned in front of a platform with a brown bag of chips and an orange on top of it. A tablecloth with blue sky and white cloud patterns covers the platform underneath the objects. The robot’s goal is to grasp the

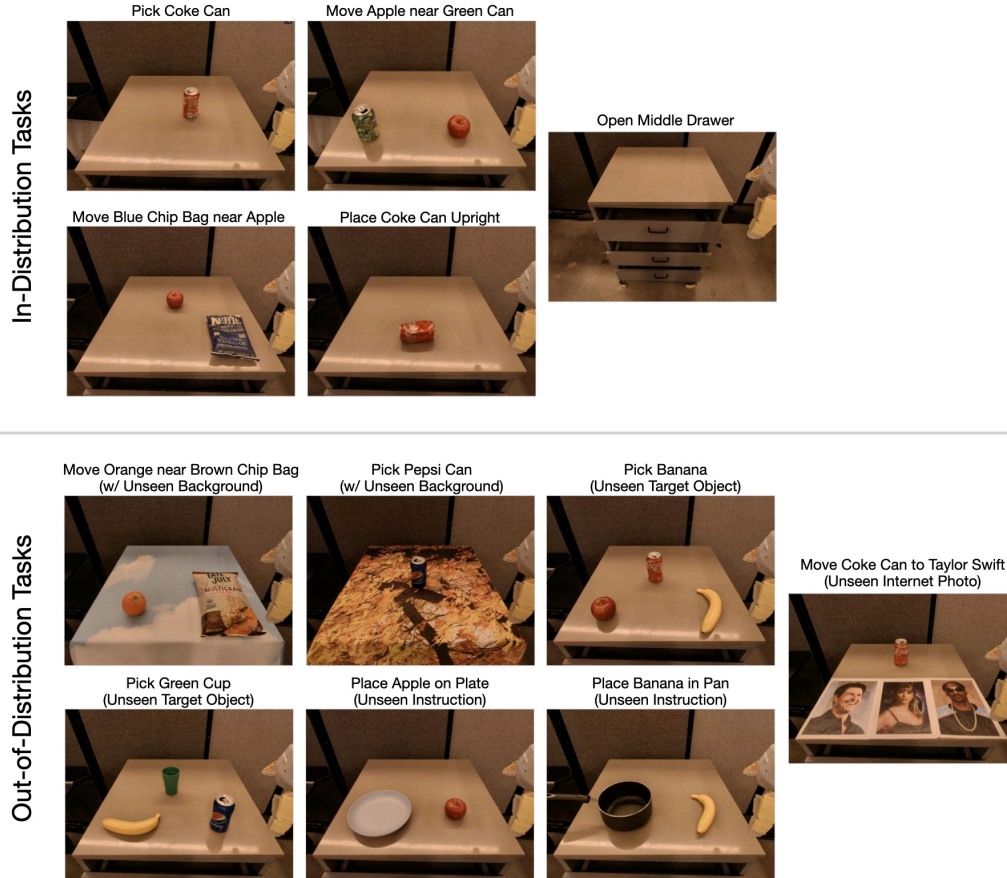


Figure 8: **RT-1 robot evaluation tasks.** We evaluate every generalist robot policy on in-distribution tasks and out-of-distribution (OOD) generalization tasks. OOD tasks involve unseen backgrounds, target objects, instructions/object relations, and semantic concepts (e.g., photos from the Internet that do not appear in robot action data).

- 850 orange and bring it next to the bag of chips. This task is OOD because the orange is an  
851 unseen object relative to the training dataset, and the tablecloth is an unseen background.<sup>4</sup>
- 852 7. **Pick Pepsi Can** (OOD): The robot is positioned in front of a platform with a can of Pepsi  
853 on top of it. A tablecloth with bright yellow/brown patterns covers the platform underneath  
854 the can. The robot’s goal is to grasp and lift the can. This task is OOD because the Pepsi  
855 can is an unseen object, and the tablecloth is an unseen background.
- 856 8. **Pick Banana** (OOD): The robot is positioned in front of a platform with an apple, a can  
857 of Coke, and a banana. The robot’s goal is to grasp and lift the banana. This task is OOD  
858 because the banana is an unseen target object.
- 859 9. **Pick Green Cup** (OOD): The robot is positioned in front of a platform with a banana, a can  
860 of Pepsi, and a green cup. The robot’s goal is to grasp and lift the green cup. This task is  
861 OOD because all objects in the scene are unseen in the training data.
- 862 10. **Place Apple on Plate** (OOD): The robot is positioned in front of a platform with a plate and  
863 an apple. The robot’s goal is to grasp the apple and move it onto the plate. This task is OOD  
864 because it is a novel instruction describing an unseen object relation: training demonstrations  
865 only cover moving the apple *near* the plate, rather than placing it *on top of* the plate.

<sup>4</sup>See Appendix of Brohan et al. [7] for a detailed list of OOD conditions in RT-1 robot evaluations.

11. **Place Banana in Pan (OOD)**: The robot is positioned in front of a platform with a pan and a banana. The robot’s goal is to grasp the banana and move it into the pan. This task is OOD because the banana is an unseen target object, and it is a novel instruction describing an unseen object relation, as explained in the previous task.

## B.2.2 Detailed RT-1 Robot Evaluation Results

Table 5: **Detailed RT-1 robot evaluation results.** We report full evaluation results for RT-1 robot evaluations discussed in Section 4.1. Each generalist policy is evaluated with 60 rollouts across 12 tasks, covering both in-distribution and out-of-distribution (OOD) testing conditions. In the bottom row, we report mean success rate  $\pm$  StdErr for each policy. OpenVLA and RT-2-X both significantly outperform RT-1-X and Octo overall (we bold the mean success rate for both due to overlapping error bars). See Fig. 8 for illustrations of all tasks.

Category	Task	# Trials	RT-1-X # Successes	Octo # Successes	RT-2-X # Successes	OpenVLA (ours) # Successes
In-distribution	Pick Coke Can	5	<b>5</b>	1	<b>5</b>	<b>5</b>
In-distribution	Move Apple near Green Can	5	3	3	3	<b>5</b>
In-distribution	Move Blue Chip Bag near Apple	5	0	3	4	<b>5</b>
In-distribution	Place Coke Can Upright	5	0	0	<b>4</b>	<b>4</b>
In-distribution	Open Middle Drawer	5	0	<b>4</b>	2	3
OOD	Move Orange near Brown Chip Bag	5	1	2	<b>5</b>	<b>5</b>
OOD	Pick Pepsi Can	5	3	0	<b>5</b>	4
OOD	Pick Banana	5	<b>5</b>	3	<b>5</b>	<b>5</b>
OOD	Pick Green Cup	5	1	0	<b>5</b>	<b>5</b>
OOD	Place Apple on Plate	5	0	0	<b>4</b>	<b>4</b>
OOD	Place Banana in Pan	5	0	0	2	<b>4</b>
OOD	Move Coke Can near Taylor Swift	5	2	0	<b>3</b>	2
Mean Success Rate			33.3 $\pm$ 6.1%	26.7 $\pm$ 5.8%	<b>78.3<math>\pm</math>5.4%</b>	<b>85.0<math>\pm</math>4.6%</b>

Full results for the RT-1 robot evaluations are shown in Table 5. Overall, we find that RT-1-X and Octo experience difficulty on the evaluation tasks; they are often unable to achieve a single success out of five trials in several tasks. On the other hand, RT-2-X and OpenVLA demonstrate strong performance, completing every task at least two times out of five trials; these two VLA policies perform comparably with each other on this particular evaluation suite.

## B.3 Data-Efficient Adaptation Experiment Details

In this section, we provide more details on the data-efficient adaptation experiments discussed in Section 4.2, where we investigate the effectiveness of fine-tuned OpenVLA policies on new robot setups such as Franka-Tabletop and Franka-DROID.

### B.3.1 Franka-Tabletop and Franka-DROID Tasks

We collect 10–150 demonstrations of each of seven tasks. The first six tasks correspond to a robot setup which we denote as “Franka-Tabletop” (Franka Emika Panda robot mounted on top of a table), and the final task corresponds to a robot setup which we call “Franka-DROID”.

In the Franka-Tabletop setup, the first three of six tasks correspond to single-instruction tasks and are narrow, while the last three tasks correspond to multi-instruction tasks in which multiple objects are present in the scene and the robot must manipulate the correct one depending on the language instruction.

Below we describe each of the six Franka-Tabletop tasks shown in Fig. 9:

1. **Put Carrot in Bowl** (single-instruction): The robot’s goal is to grasp the carrot and place it into the bowl. We collect 50 demonstrations of this task for the training dataset, randomly placing the carrot and the bowl at different locations on the table in every episode. The carrot is always initialized on the left side of the bowl. During evaluation, each trial is recorded as a success (1) or failure (0); there is no partial credit.
2. **Pour Corn into Pot** (single-instruction): The robot’s goal is to grasp the red bowl, move towards the steel pot, and pour the contents (a yellow corn) into the pot. We collect 50 demonstrations of this task for the training dataset, randomly placing the bowl and the pot at different locations on the table in every episode. The bowl is always initialized on the right side of the pot. During evaluation, each trial is recorded as a success (1) or failure (0); there is no partial credit.



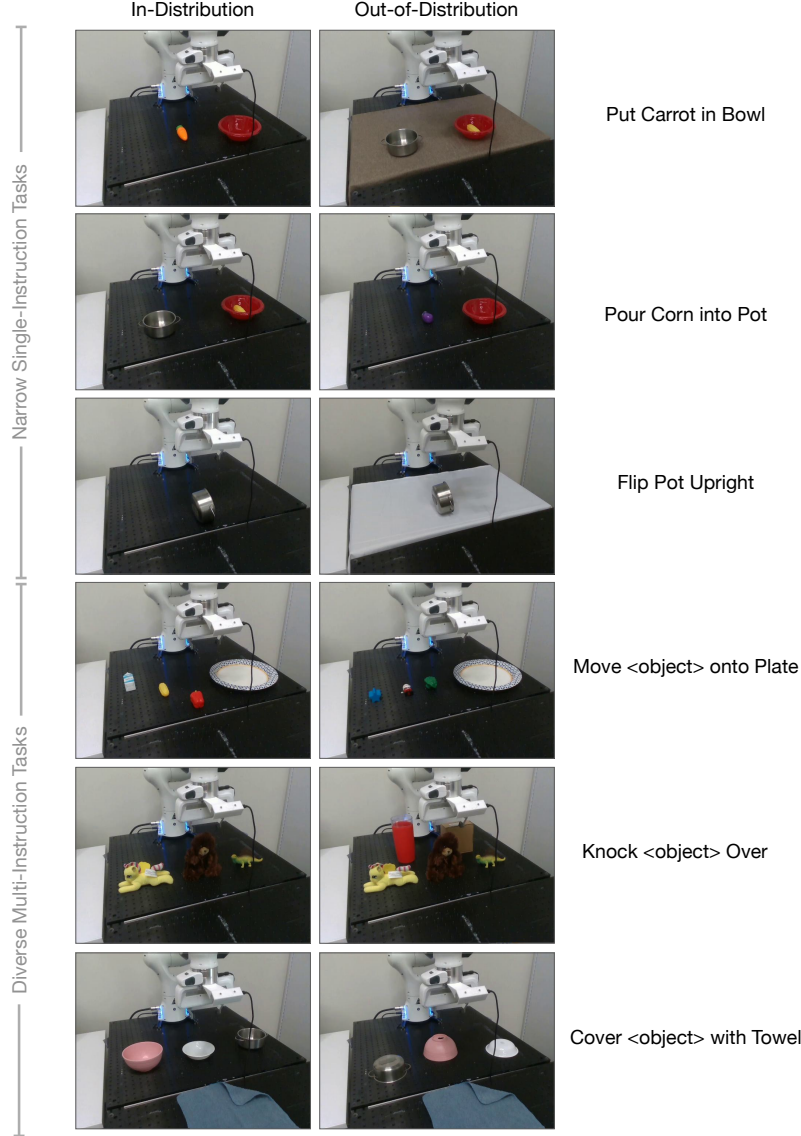


Figure 9: **Franka-Tabletop fine-tuning tasks.** Franka-Tabletop tasks used in the data-efficient adaptation experiments in Section 4.2 and described in detail in Fig. 9 are depicted above. The first three of six tasks, shown in the top three rows, only involve a single instruction, while the last three tasks in the bottom three rows involve multiple objects and instructions (the instructions specify the target object or target location). The first column shows sample initial states matching the training data distribution, while the second column shows out-of-distribution (OOD) initial states (e.g., unseen backgrounds, target objects, distractors, and object positions/orientations). Every policy in Section 4.2 is evaluated with 10–12 rollouts on in-distribution tasks and 5–6 rollouts on OOD tasks.

3. **Flip Pot Upright** (single-instruction): The robot’s goal is to grasp the steel pot (which is initially oriented vertically), rotate it to be in the upright position, and place it back onto the table. We collect only 10 demonstrations of this task for the training dataset, randomly placing the steel pot at various locations within a small section of the table. During evaluation, each trial is recorded as a success (1), failure (0), or partial success (0.5). Partial successes include grasping the pot but not orienting it upright, or knocking it over to the upright position but not carefully guiding it. The robot must release the pot at the end of the episode for full credit.

- 908 4. **Move <object> onto Plate** (multi-instruction): The robot’s goal is to grasp one out of  
 909 three objects (depending on the target specified in the language instruction) and place it  
 910 on the plate on the right side of the table. We collect 150 demonstrations of this task for  
 911 the training dataset, randomly placing different combinations of three objects on the table  
 912 and selecting one as the target. The plate is always initialized on the right side of the table.  
 913 During evaluation, each trial is recorded as a success (1), failure (0), or partial success (0.5).  
 914 Partial success is recorded when the first object that the robot makes contact with is the  
 915 correct target object (i.e., the object specified in the language instruction), but the robot does  
 916 not complete the task.
- 917 5. **Knock <object> Over** (multi-instruction): The robot’s goal is to approach one out of three  
 918 objects (depending on the target specified in the language instruction) and push it until  
 919 it falls over. We collect 70 demonstrations of this task for the training dataset, randomly  
 920 placing different combinations of three objects on the table and selecting one as the target.  
 921 During evaluation, each trial is recorded as a success (1), failure (0), or partial success (0.5).  
 922 Partial success is recorded when the first object that the robot makes contact with is the  
 923 correct target object (i.e., the object specified in the language instruction), but the robot does  
 924 not complete the task.
- 925 6. **Cover <object> with Towel** (multi-instruction): The robot’s goal is to grasp the blue towel  
 926 and place it on one out of three objects (depending on the target specified in the language  
 927 instruction). We collect 45 demonstrations of this task for the training dataset, randomly  
 928 placing different combinations of three objects on the table. During evaluation, each trial  
 929 is recorded as a success (1), failure (0), or partial success (0.5). Partial success is recorded  
 930 when the first object that the robot touches with the towel is the correct target object (i.e.,  
 931 the object specified in the language instruction), but the robot does not complete the task  
 932 (e.g., it drops the towel onto the table instead of on top of the target object). Full credit is  
 933 given when any part of the towel is resting over the top surface of the target object, i.e., the  
 934 object does not need to be fully covered.

935 For every Franka-Tabletop task, we evaluate each method with 10–12 in-distribution trials and 5–6  
 936 OOD generalization trials. The in-distribution and OOD test conditions are depicted in Fig. 9 (second  
 937 column).

938 We describe the OOD test conditions for each of the six tasks below:

- 939 1. Put Carrot in Bowl (OOD): An eggplant (unseen object) replaces the carrot.  
 940 2. Pour Corn into Pot (OOD): An unseen brown tablecloth covers the tabletop.  
 941 3. Flip Pot Upright (OOD): An unseen white tablecloth covers the tabletop  
 942 4. Move <object> onto Plate (OOD): A set of three unseen objects are placed on the table.  
 943 5. Knock <object> Over (OOD): Two unseen distractor objects (red plastic cup and brown  
 944 box) are positioned behind the set of three seen objects.  
 945 6. Cover <object> with Towel (OOD): The three objects on the table are placed upside-down  
 946 and at unseen positions.

947 Finally, in the Franka-DROID environment, we experiment with one task and variants of it: **Wipe**  
 948 **Table** (see Fig. 10). In this task, the robot’s goal is to grab the brush and sweep all three small brown  
 949 objects into the dustpan. We collect 70 demonstrations for this task for the training dataset, varying  
 950 the positions of all the objects.

951 At test time, we evaluate on in-distribution conditions matching the training data (Fig. 10, left), as  
 952 well as out-of-distribution (OOD) conditions in which distractor objects are also present in the scene  
 953 on the table (Fig. 10, right). Since there are various possible outcomes for each trial, we define a  
 954 scoring rubric as follows: The maximum score for each trial is 2 points. The policy receives the full  
 955 2 points if the robot sweeps all three objects into the dustpan. It receives 1 point for successfully  
 956 sweeping one or two objects into the dustpan. Otherwise, it receives 0 points. We evaluate each  
 957 policy with 18 in-distribution trials and 12 OOD trials, so each policy receives an aggregate score out  
 958 of 60 points.

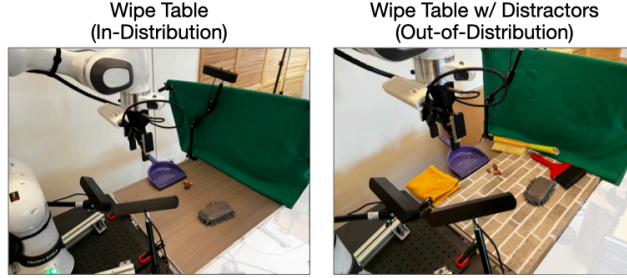


Figure 10: **Franka-DROID fine-tuning task.** The “Wipe Table” task shown here is the final task used in the data-efficient adaptation experiments in Section 4.2. The left image shows the initial conditions for an in-distribution trial. The right image shows an out-of-distribution trial in which unseen distractor objects are present on the table. To fully complete the task, the robot must grab the brush and sweep all three objects into the dustpan.

### B.3.2 Detailed Franka-Tabletop and Franka-DROID Evaluation Results

Full evaluation results for both Franka-Tabletop and Franka-DROID evaluations are shown in Table 6. We evaluate the methods discussed in Section 4.2. We find that Diffusion Policy demonstrates strong performance on the single-instruction Franka-Tabletop tasks (“Put Carrot in Bowl”, “Pour Corn in Pot”, and “Flip Pot Upright”), outperforming other methods. However, OpenVLA (Octo Mix) and Octo achieve higher performance in the more diverse multi-instruction tasks (“Move <object> onto Plate”, “Knock <object> Over”, and “Cover <object> with Towel”). In the Franka-DROID environment, OpenVLA (Octo Mix) obtains strong results. Overall, we find that OpenVLA (Octo Mix) achieves the highest average performance across both tasks.

Table 6: **Detailed data-efficient adaptation experiment results.** We report the performance of Diffusion Policy trained from scratch on new robot tasks, as well as generalist policies fine-tuned on the same data. Each policy is tested against both in-distribution and out-of-distribution (OOD) generalization conditions. We find that no single policy performs best on all tasks: Diffusion Policy achieves high success rates on single-instruction tasks, while OpenVLA (Octo Mix) and Octo performs well on diverse multi-instruction tasks. In terms of aggregate performance, however, OpenVLA (Octo Mix) obtains the highest average success rate across both environments.

	# trials	Diffusion Policy	Diffusion Policy (matched)	Octo	OpenVLA (scratch)	OpenVLA (Octo Mix) (ours)
Franka-Tabletop (5Hz)	“Put Carrot in Bowl” (in-distribution)	10	<b>90.0</b>	80.0	40.0	<b>90.0</b>
	“Put Carrot in Bowl” (OOD)	5	20.0	0.0	20.0	<b>40.0</b>
	“Pour Corn into Pot” (in-distribution)	10	<b>100.0</b>	90.0	0.0	10.0
	“Pour Corn into Pot” (OOD)	5	<b>80.0</b>	60.0	0.0	60.0
	“Flip Pot Upright” (in-distribution)	10	<b>100.0</b>	85.0	40.0	85.0
	“Flip Pot Upright” (OOD)	5	50.0	20.0	0.0	<b>60.0</b>
	“Move <object> onto Plate” (in-distribution)	12	25.0	25.0	41.7	<b>79.2</b>
	“Move <object> onto Plate” (OOD)	6	8.3	33.3	8.3	<b>58.3</b>
	“Knock <object> Over” (in-distribution)	12	33.3	25.0	83.3	<b>62.5</b>
	“Knock <object> Over” (OOD)	6	16.7	16.7	33.3	<b>83.3</b>
	“Cover <object> with Towel” (in-distribution)	12	16.7	20.8	<b>91.7</b>	20.8
	“Cover <object> with Towel” (OOD)	6	16.7	33.3	<b>91.7</b>	0.0
Average			48.5±4.9%	43.4±4.7%	43.4±4.4%	38.9±4.5%
Franka-DROID (15Hz)						
“Wipe Table” (in-distribution)		18	50.0%	27.8%	52.8%	30.6%
“Wipe Table” + Distractors (OOD)		12	12.5%	25.0%	16.7%	20.8%
Average			35.0±8.0%	26.7±7.5%	38.3±8.5%	26.7±7.5%
						<b>55.0±7.7%</b>

## C Infrastructure for Training and Inference

The final OpenVLA model is trained on a cluster of 64 A100 GPUs for 14 days, or a total of 21,500 A100-hours, using a batch size of 2048. During inference, OpenVLA requires 15GB of GPU memory when loaded in bfloat16 precision, i.e., without quantization, and runs at approximately 6Hz on one NVIDIA RTX 4090 GPU (without compilation, speculative decoding, or other inference speed-up tricks). We can further reduce the memory footprint of OpenVLA during inference via quantization, and we demonstrate minimal performance regression with quantized VLA execution in Section 4.3. We report inference speed on various consumer- and server-grade GPUs in Fig. 11. For convenience, we implement a remote VLA inference server to allow realtime remote streaming of action predictions to the robot – removing the need for the robot to have a powerful GPU. We release this remote inference solution as part of our open-source code release (Appendix D).

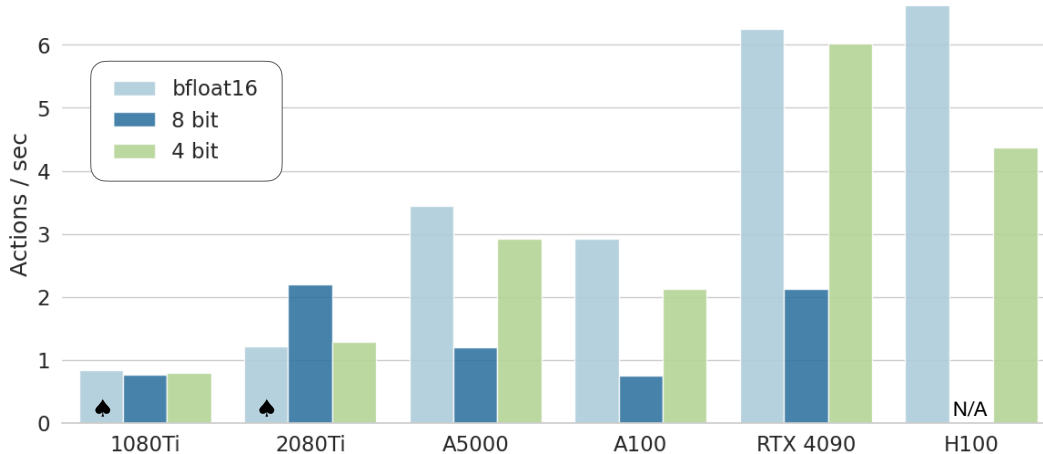


Figure 11: **OpenVLA inference speed for various GPUs.** Both bfloat16 and int4 quantization achieve high throughput, especially on GPUs with Ada Lovelace architecture (RTX 4090, H100). Further speed-ups are possible with modern LLM inference frameworks like TensorRT-LLM [106]. ♠: Model sharded across two GPUs to fit.

## D The OpenVLA Codebase

Along with our model, we will release the OpenVLA codebase, a modular PyTorch codebase for training VLA models (see <https://anonymous-openvla.github.io>). It scales from fine-tuning VLAs on individual GPUs to training billion-parameter VLAs on multi-node GPU clusters, and supports modern techniques for large transformer model training such as automatic mixed precision (AMP, PyTorch [107]), FlashAttention [108] and fully sharded data parallelism (FSDP, Zhao et al. [80]). Out of the box, the OpenVLA codebase has full support for training on the Open X-Embodiment dataset, integrates with HuggingFace’s [21] AutoModel class, supports LoRA fine-tuning [25] and quantized model inference [26, 81]. Everything will be released under a permissive MIT license.

## E RT-2-X vs. OpenVLA in Bridge V2 Evaluations

In this section, we provide additional details on RT-2-X vs. OpenVLA comparisons in BridgeData V2 evaluations discussed in Section 4.1. As discussed previously, OpenVLA is pretrained on a larger subset of OpenX data than RT-2-X and uses a fused SigLIP-DinoV2 vision backbone rather than a single visual encoder. However, in addition to these factors, we believe that OpenVLA’s significant improvement upon RT-2-X specifically in BridgeData V2 evaluations (as shown in Fig. 3) also stems from more careful preprocessing of the Bridge dataset.

During the development of the OpenVLA model, we discovered that the original version of the BridgeData V2 dataset contained many transitions with all-zero (no-op) actions. For instance, in every demonstration, an all-zero action was recorded as the ground-truth action in the first timestep. Consequently, training a highly expressive VLA model on the original dataset without any data preprocessing led to a policy that frequently predicted all-zero actions and froze during evaluations. Therefore, we simply filtered out the first transition in every demonstration when training the OpenVLA model, and this was sufficient for mitigating the freezing behavior in most cases.

However, the RT-2-X model was trained without such data preprocessing, so it often suffers the aforementioned freezing behavior if deployed out of the box without modifying the model querying procedure – which severely deteriorates rollout performance. Since this is a proprietary model that is infeasible for us to re-train (e.g., with our preprocessed version of the BridgeData V2 dataset), we mitigated this issue by simply querying the *second-most-likely* action from the model, since the first-most-likely action was often all zeros while the second-most-likely action was not. (Note that this is the same workaround that was applied by the developers of the RT-2-X model for BridgeData V2 evaluations reported in the Open X-Embodiment experiments [1].) This workaround led to much stronger RT-2-X performance on BridgeData V2 evaluations – though we believe that it is still suboptimal compared to re-training the model on the preprocessed version of the dataset.



1012 We also tried to *dynamically* query RT-2-X, i.e., by first sampling the first-most-likely action and  
1013 then sampling the second-most-likely action if the first one was all zeros. However, we empirically  
1014 found that dynamic querying led to worse performance than simply querying the second-most-likely  
1015 action at all times. We hypothesize that this is due to a change in the robot’s dynamics that arises  
1016 from dynamic querying: pausing in the middle of a trajectory to re-query the model leads to slight  
1017 interruptions in the robot’s movement due to non-negligible latency in the querying pipeline, and this  
1018 leads to subtle performance degradation. Therefore, we report the performance of RT-2-X when  
1019 always querying the second-most-likely action, as done in the Open X-Embodiment project [1].