1 Appendix

1.1 Canonical Task Setup

We considered four canonical tasks: `reach`, `slide`, `lift` and `PnP`. To apply ORL, each task can be formulated as an MDP. The state contains the joint position of the robot, the gripper open position ($\mathcal{R} \sim [0, 0.08]$), (optionally) the velocity of the joints, (optionally) the tracked tag position and a goal position. To facilitate RL training, we came up with a continuous reward function for each task r : *state* $\rightarrow \mathcal{R}$, as shown in Table 1, considering the position of the gripper x, the position of tracked AprilTag t (if exists), the position of goal g, the Euclidean distance function dis between two 3D coordinates, a convenient function $height$ to denote the height of a given coordinates. While the reward for `reach` and `slide` are naturally smaller than 1, we explicitly cap the maximum reward for `lift` to be 1 since we don't encourage agents to lift up the lid arbitrarily high. We don't cap the `PnP` reward since we encourage the pick-n-place policy to be distinguished from the policy with a height bonus $height(t)$.

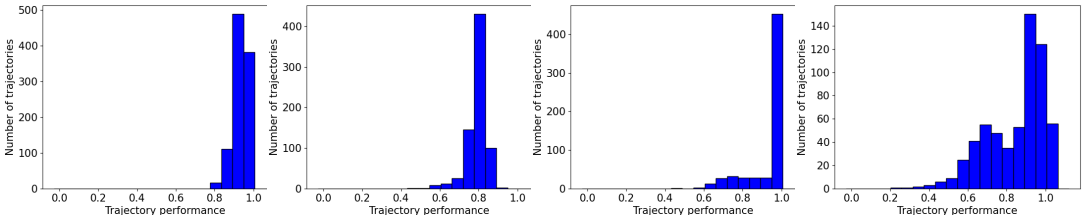
We used heuristic policies to collect the demonstration data, as described in Sec. **??**. Our policies have a reasonable success rate *accomplishing* the task but is not designed to be optimal in solving the MDP. To evaluate and compare between agents, we instead report the maximum reward over the trajectory as a proxy of the task completion ("score"). We report our heuristic policies' accumulated reward average over trajectories and the score.

Task	$r(s)$	$\sum r(s)$	Score
Reach	$1 - dis(g - x)$	173	0.99
Slide	$1 - (2 * dis(g - t) + dis(t - x))$	223	0.93
Lift	$min(1, 0.57 - dis(t - x) + height(t))$	167	1
Pick-n-place	$1 - (dis(g - t) + 2 * dis(t - x)) * 0.9 + height(t)$	281	1.09

Table 1: Characteristics of task and collected data.

1.2 Dataset

In addition to the reward functions and statistics of our dataset, we also attach the score distribution on each task to demonstrate our dataset's overall quality. From Figure 1, we can see that the score distribution for each task skew heavily to the left, which means the datasets are suitable for imitation learning as well.



(a) Reaching (b) Sliding (c) Lifting (d) Pick-n-place

Figure 1: Score distribution for each task of our dataset.

1.3 Open Source Code and Dataset

To remain anonymity we have only uploaded our collected dataset to here. Once accepted, we would share code and instructions on how to process and use our dataset.

1.4 Training Details

Our code base was built upon the author's implementation of MOREL [1] and the D3RLPY [2] library. We used the same fixed random seed for all our experiments, unless otherwise specified.

For hyperparameter tuning, we always started by training using the default hyperparameters. If the training loss reported by the agent did not converge, we adjusted the learning rate and retrain, up to 5 agents, till we find a model that converge or have been trained for $5,000,000$ steps using batch size 2048. For model whose training loss exploded (e.g., AWAC), we choose an checkpoint from earlier of the training when the loss were relatively stable for $100,000$ steps (frequently, this was an agent that finished about half a million to a million training steps). Surprisingly, when evaluated on real robot, models that reported convergence did not necessarily perform better than model that did not converge.

Practicality of Training and Tuning BC was the cheapest to train ($\sim$ 3min) and easiest to converge (no additional tuning required). MOREL was the second shortest to train ($\sim$ 4 hours); most MOREL agents were able to converge, judged by the reward of trajectories generated by the learned dynamics model. AWAC agents took longer to train ($\sim$ 12 hours) and had the most trouble converging (8 of the 16 agents in the ablation table could not converge in allocated trials). IQL agents took the longest to train (10 $\sim$ 24 hours) but had more success converging. Though loss convergence during training or a good reward estimated by the learned dynamics model or learned value function cannot indicate the agent's true performance, it is helpful for selecting an agent to test. Since some AWAC agents had trouble converging, we selected an earlier checkpoint before loss explosion and documented their performance, which, surprisingly, yielded higher reward than some agents that reported convergence. We leave it to future work to investigate this phenomenon.

1.5 Training Behavior Cloning with Top-K% Trajectories

To ensure that our dataset contains high quality trajectories that is sufficient to train behavior cloning, we launched new experiments training behavior cloning using only the Top-k % of the best trajectories. In Figure. 1, we plot the distribution of performance of our data for each task. For `reach`, `slide`, `lift`, 90% of trajectories complete the task with good scores (> 0.75). For (our most difficult task), 50% of our collected trajectories completed the task (scores > 0.8).

Thus we train BC for `reach`, `slide`, `lift` on Top-90% of data and train BC for `PnP` on Top-50%,70%,90% of data and observe that, BC in our experiments benefit from using the full dataset.

Task	Top-k%	#Trajs	Threshold for Demo	Score	Original Score (BC with full data)
`reach`	90	900	0.909	0.899 ± 0.037	0.924 ± 0.048
`slide`	90	657	0.774	0.659 ± 0.152	0.681 ± 0.147
`lift`	90	554	0.787	0.784 ± 0.157	0.823 ± 0.177
`PnP`	50	304	0.935	0.723 ± 0.217	0.818 ± 0.185
	70	426	0.792	0.789 ± 0.290	0.818 ± 0.185
	90	548	0.656	0.789 ± 0.204	0.818 ± 0.185

1.6 Sweeping of Random Seeds

We evaluated an addition of 28 agents for 340 trajectories for a total of 70 hours including training and testing to inspect how the scores for critical agents (i.e., the best agents for a category) would vary by random seeds. We now have 3 seeds for each of the following agents:

1. The Best Agents for each task in Table **??**
2. The Second Best Agents for each task in Table **??**
3. ORL agents with out-domain datasets in in Table **??**

The original agents are trained with seed 123, we trained the additional agents with seed 122 and seed 124. Each seed is evaluated on 12 trajectories. The results are listed and we observe that $\sim$60% of newly trained agents change score by less than 1%, $\sim$90% of agents change by less than 2%, and the maximum change was 6% from one agent (whose score change does not affect our conclusion).

Best Agents in Table **??**	**Seed 122**	**Seed 124**	**Seed 123** (original seed)	**Means w/ 3 seeds**	**Mean diff**
AWAC, DeltaVel,`reach`	0.920 ± 0.031	0.919 ± 0.066	0.935 ± 0.032	0.925 ± 0.047	0.01 (1.07%)
IQL, DeltaVel, `slide`	0.781 ± 0.038	0.763 ± 0.044	0.757 ± 0.095	0.767 ± 0.065	-0.01 (-1.32%)
IQL, DeltaVel, `lift`	0.877 ± 0.166	0.878 ± 0.158	0.884 ± 0.120	0.880 ± 0.149	0.004 (0.45%)
BC, AbsVel, `PnP`	0.819 ± 0.199	0.800 ± 0.195	0.836 ± 0.157	0.818 ± 0.185	0.018 (2.15%)

Second Best in Table **??**	**Seed 122**	**Seed 124**	**Seed 123** (original seed)	**Means w/ 3 seeds**	**Mean diff**
MOREL, DeltaVel, `reach`	0.919 ± 0.034	0.908 ± 0.042	0.925 ± 0.028	0.917 ± 0.036	0.008 (0.86%)
BC, DeltaVel, `reach`	0.921 ± 0.051	0.917 ± 0.055	0.934 ± 0.032	0.924 ± 0.048	0.01 (1.07%)
BC, AbsVel, `slide`	0.699 ± 0.125	0.698 ± 0.120	0.645 ± 0.18	0.681 ± 0.147	-0.036 (-5.58%)
MOREL, DeltaVel, `slide`	0.655 ± 0.157	0.602 ± 0.180	0.629 ± 0.136	0.629 ± 0.160	0 (0%)
AWAC, DeltaVel,`slide`	0.757 ± 0.068	0.703 ± 0.108	0.739 ± 0.144	0.732 ± 0.113	0.007 (0.95%)
BC, AbsVel, `lift`	0.821 ± 0.192	0.832 ± 0.177	0.818 ± 0.161	0.823 ± 0.177	-0.005 (0.61%)

1.7 Statistical Significance of Conclusions

In this section we verify the statistical significance of the conclusions we drew from our empirical study. To evaluate every trained agent for every task, we collected 12 trajectories and calculated their scores. One one hand, the estimated standard deviations of such scores were large, making the comparison between agents challenging (i.e. comparing 0.818 ± 0.161 with 0.884 ± 0.120). On the other hand, the distribution of scores is unknown. We cannot exclude the possibility of the distribution being skewed, as the agent could perform better in a certain task region because of the nature of the task. Therefore, we conducted both the dependent t-test (p) and the Wilcoxon signed T-test (p_w) for paired samples to calculate the p-value to reject or accept this null hypothesis: the two models' have identical scores.

We will reject the hypothesis with a small p-value (p or $p_w < 0.1$). Tasks and application-domains determine the confidence level requirements for any application. This often requires domain knowledge and might not transfer between different applications even for the same task. For openness and interpretability, we clearly outline our statistical tests and list our p-values, leaving it up to the readers to justify their statistical significance required for their applications. We found that:

1. On in-domain tasks, we initially observe that: on `reach`, BC and the best ORL agent (AWAC) achieved similar performance ($0.93 \sim 0.93, p = 0.953, p_w = 0.844$); on `slide`, IQL outperform BC ($0.76 > 0.64, p = 0.066, p_w = 0.110$); on `lift`, we observe that BC is identical to the best ORL ($0.82 \sim 0.88, p = 0.146, p_w = 0.110$); on `PnP`, we observed that BC outperformed the best ORL agent ($0.90 > 0.75, p = 0.012, p_w = 0.016$). After running the best and the second best agents with multiple seeds, we can confirm the statistical significance of IQL outperforming BC on `lift` and `slide` ($0.88 > 0.82, p = 0.084, p_w = 0.041$, $0.77 > 0.68, p = 0.001, p_w = 0.001$). With such observation, we recommend IQL and BC as a strong baseline for in-domain tasks.
2. Testing agent's ability to generalize to task space lacking data support, we verify that MOREL and AWAC achieved comparable performance or better to BC for regions lacking data support (MOREL: $0.80 \sim 0.77, p = 0.235, p_w = 0.500$, AWAC: $0.82 > 0.77, p = 0.006, p_w = 0.250$). It's worth noting that MOREL was having an initial disadvantage

ORL in Table **??**	**Seed 122**	**Seed 124**	**Seed 123** (original seed)	**Means w/ 3 seeds**	**Mean diff**
AWAC on `PnP` w/ slide+lift	(diverged)	0.811 ± 0.103	0.815 ± 0.134	0.813 ± 0.121	0.002 (0.25%)
AWAC on `PnP` w/ slide+lift+pnp	0.759 ± 0.180	0.773 ± 0.204	0.742 ± 0.175	0.758 ± 0.188	-0.016 (-2.16%)
IQL on `PnP` w/ slide+lift	0.838 ± 0.103	0.847 ± 0.117	0.843 ± 0.120	0.842 ± 0.114	0.001 (0.12%)
IQL on `PnP` w/ slide+lift+pnp	0.842 ± 0.170	0.826 ± 0.211	0.829 ± 0.163	0.833 ± 0.183	-0.004 (-0.48%)
MOREL on `lift` w/ slide+lift+pnp	0.879 ± 0.124	0.904 ± 0.119	0.906 ± 0.151	0.896 ± 0.133	-0.01 (-1.1%)

of having poorer performance on regions that have more data support ($0.67 < 0.78, p = 0.050, p_w = 0.062$).

3. In terms of leveraging task-agnostic data, MOREL has benefited from inclusion of more data. On Slide, the model achieved significantly higher performance when using combined data from three tasks $0.64 \sim 0.72, p = 0.113, p_w = 0.027$). On Lift, the model achieved significantly higher performance when using combined data from three tasks ($0.65 \rightarrow 0.91, p = 0.000, p_w = 0.003$). AWAC and IQL agents, however, had less success achieved higher scores. The only significant improvement is AWAC on Lifting ($0.82 \rightarrow 0.90, p = 0.082, p_w = 0.059$). Otherwise, AWAC agents performed the same irregardless of training data ($p > 0.1$). IQL had mostly similar or worse performance leveraging more data (e.g. worse slide performance: $0.70 \rightarrow 0.64, p = 0.069, p_w = 0.077$).

References

[1] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL : Model-Based Offline Reinforcement Learning. In *NeurIPS*, 2020.

[2] Michita Imai Takuma Seno. d3rlpy: An offline deep reinforcement library. In *NeurIPS 2021 Offline Reinforcement Learning Workshop*, December 2021.