## A Algorithmic Details

### A.1 FiLM Conditioning

Feature-wise Linear Modulation (FiLM) [46] is a technique used for conditioning neural networks that allows the network to modulate its behavior based on an external conditioning signal, such as text instructions or observations. In the context of text conditioning for policy learning, the text instructions are first encoded into a conditioning vector. This conditioning vector is then used to modulate the activations of the neural network through FiLM layers. FiLM applies a feature-wise affine transformation (scaling and shifting) to the activations of the network, conditioned on the text embedding. In other word, assuming **x** is a FiLM layer's input, **z** is a conditioning input, and $\gamma$ and $\beta$ are **z**-dependent scaling and shifting vectors,

$$FiLM(\mathbf{x}) = \gamma(\mathbf{z}) \odot \mathbf{x} + \beta(\mathbf{z}) \tag{3}$$

This allows the network to adapt its computation and output based on the given text instructions, enabling tasks like instruction following or conditioning the policy on language descriptions.

### A.2 Action Heads

Having a separate action prediction module allows BAKU to leverage state-of-the-art techniques for action generation. In this work, we evaluate five different action head variants. Below we briefly describe each variant. For more details on these methods, please refer to the original publications.

**Multilayer Perceptron (MLP)** This is a simple neural network comprising multiple dense layers. We use a two-layer MLP for our experiments.

**Gaussian Mixture Model (GMM) [36]** A Gaussian mixture model (GMM) action head models the policy as a mixture of Gaussians, enabling multi-modal action sampling for continuous control problems. The GMM parameters are part of the learned policy network. For our experiments, we employ a two-layer GMM action head with five action modes and a Softplus activation function.

**Behavior Transformer (BeT) [60]** The Behavior Transformer (BeT) models continuous action prediction as a two-part problem. Actions in the training data are first clustered into $k$ bins using k-means clustering. A discrete action head classifies the cluster an action belongs to, while an offset action head predicts an offset value added to the corresponding cluster center. The discrete head uses a focal loss, while the offset head uses L2 loss. For our experiments, we use BeT with 64 action clusters.

**Vector-Quantized Behavior Transformer (VQ-BeT) [31]** The Vector-Quantized Behavior Transformer (VQ-BeT) extends BeT by replacing k-means clustering with residual VQVAE-based tokenization, significantly improving performance over BeT. For our experiments, we employ VQ-BeT with two residual VQ layers of codebook size and latent dimension 16 and 256, respectively.

**Diffusion [45, 10, 55]** A diffusion action head models action prediction as a diffusion process that generates actions over time by iteratively denoising samples from a Gaussian distribution. While highly effective for multi-modal distributions, the iterative denoising during inference slows deployment speed. In this work, we use a transformer-based diffusion head introduced by prior work [45, 10]. We use a two-layer diffusion head for our experiments.

### A.3 Temporal smoothing over action chunking

A naïve implementation of action chunking, where a new environment observation in incorporated every $k$ steps can be suboptimal and can result in jerky robot motion. To improve the smoothness in robot motion, we incorporate an exponential temporal ensembling technique, following prior work [77, 5]. Instead of querying the policy every $k$ steps, we query it at every timestep. This results in an overlap in predicted action chunks and at any given timestep, there will be more than one predicted actions. Instead of using only the current action prediction, we use a temporal ensemble to combine all the past predictions. This temporal ensemble performs a weighted average over these predictions with an exponential weighing scheme $w_i = exp(-m * i)$, where $w_0$ is the weight for the oldest action. The speed for incorporating a new observation is governed by $m$, where a smaller $m$ means faster incorporation. It must be noted that this ensembling incurs no additional training cost, only extra inference-time computation. In our experiments, similar to prior work [77, 5], we find both action chunking and temporal ensembling to be important for producing precise and smooth motion.

### A.4 Hyperparameters

The complete list of hyperparameters is provided in Table 4. For RT-1 [6], we use our implementation with an RT-1 action head that discretizes the continuous action into discrete bins uniformly. For MT-ACT [5], we use the open-source implementation with the default hyperparameters. We vary the action chunk length for MT-ACT for different benchmarks, the values for which have been provided in Table 4.

**Training time** Below we provide details about the time required to train BAKU on a single NVIDIA RTX A4000 GPU.

1. **LIBERO:** Training for $600k$ steps with a batch size of 64 and 2 camera views and robot proprioception as input requires around 10.5 hours.
2. **Meta-World:** Training for $600k$ steps with a batch size of 64 and 1 camera view as input requires around 8 hours.
3. **DM Control:** Training for $2M$ steps with a batch size of 128 and robot state as input requires around 26 hours.
4. **xArm Robot:** Training for $200k$ steps with a batch size of 64 and 4 camera views and robot proprioception as input requires around 6 hours.

## B Simulation Tasks

We evaluate BAKU on three simulated benchmarks: LIBERO-90 [34], MetaWorld [76], and DM Control [67]. For LIBERO-90, we directly use the dataset provided, which includes demonstrations for all 90 tasks. For details on the specific LIBERO-90 tasks, please refer to the original paper [34]. For MetaWorld and DM Control, we collected demonstrations from expert agents trained with reinforcement learning (RL). We include only the tasks for which we were able to obtain expert demonstration data. Table 5 lists the 30 MetaWorld tasks and 9 DM Control tasks used in our experiments.

## C Robot Tasks

We evaluate BAKU on 30 tasks in our real-world multi-task kitchen environment. We provide the task description along with policy deployment rollouts with BAKU for each task in Figures 5, 6, 7, 8, and 9. The long-horizon task rollouts have been shown in Figure 10.

**Robot control** We deploy our learned policies at 10Hz using a high-level controller. To facilitate smooth motion on the robot, we deploy a low-level Minimum-Jerk Controller at 100Hz.

## D Additional Results and Analysis

### D.1 Real-World Task-wise Results

Table 6 provides the task-wise performance for all 30 tasks in our real-world multi-task kitchen environment. We collect an average of 17 demonstrations per task, with a total of 520 demonstrations across all tasks. Task-wise performance for the real-world long-horizon tasks has been included in Table 7.

Table 4: List of hyperparameters.

| Method | Parameter | Value |
|---|---|---|
| Common | Learning rate | $1e^{-4}$ |
| | Image size | $128 \times 128$ (LIBERO-90, xArm) |
| | | $84 \times 84$ (Meta-World) |
| | Mini-batch size | 64 (LIBERO-90, Meta-World, xArm) |
| | | 128 (DM Control) |
| | Optimizer | Adam |
| | Number of training steps | 600000 (LIBERO-90, Meta-World) |
| | | 2000000 (DM Control) |
| | | 200000 (xArm) |
| | Number of demonstrations | 50 (LIBERO-90) |
| | | 35 (Meta-World) |
| | | 500 (DM Control) |
| | | 15 (xArm) |
| | Transformer architecture | minGPT [29] (with 8 layers and 4 heads) |
| | Action chunk length | 10 (LIBERO-90, Meta-World) |
| | | 3 (DMC) |
| | | 20 (xArm) |
| BAKU | Observation trunk | Transformer |
| | Action head | MLP (base) |
| | | GMM, BeT, VQ-BeT, Diffusion (variants) |
| | Hidden dim | 256 |
| | Observation history | False |
| | Action chunking | True |
| | Intermediate goal steps ($k$) | 50 (LIBERO-90) |
| | | 30 (Meta-World) |
| RT-1 | Observation trunk | Transformer |
| | Action head | MLP (base) |
| | Hidden dim | 512 |
| | Observation history | True |
| | History length | 6 |
| | Action chunking | False |
| MT-ACT | Observation history | False |
| | Action chunking | True |

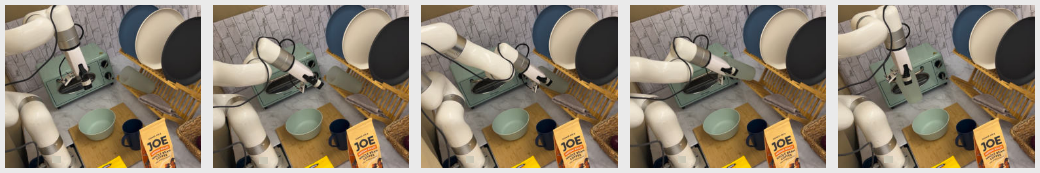
Table 5: List of tasks in Meta-World and DM Control.

| Meta-World | DM Control |
|---|---|
| basketball-v2 | cartpole swingup |
| bin-picking-v2 | cheetah run |
| button-press-v2 | hopper stand |
| button-press-topdown-v2 | quadruped run |
| button-press-topdown-wall-v2 | quadruped walk |
| button-press-wall-v2 | teacher easy |
| coffee-button-v2 | walker stand |
| coffee-pull-v2 | walker walk |
| coffee-push-v2 | walker run |
| dial-turn-v2 | |
| disassemble-v2 | |
| door-lock-v2 | |
| door-open-v2 | |
| door-unlock-v2 | |
| drawer-close-v2 | |
| drawer-open-v2 | |
| faucet-close-v2 | |
| faucet-open-v2 | |
| hammer-v2 | |
| handle-press-v2 | |
| handle-press-side-v2 | |
| handle-pull-v2 | |
| handle-pull-side-v2 | |
| peg-insert-side-v2 | |
| peg-unplug-side-v2 | |
| plate-slide-v2 | |
| plate-slide-back-v2 | |
| plate-slide-back-side-v2 | |
| plate-slide-side-v2 | |
| shelf-place-v2 | |
| soccer-v2 | |
| stick-push-v2 | |
| sweep-v2 | |
| sweep-into-v2 | |
| window-close-v2 | |
| window-open-v2 | |

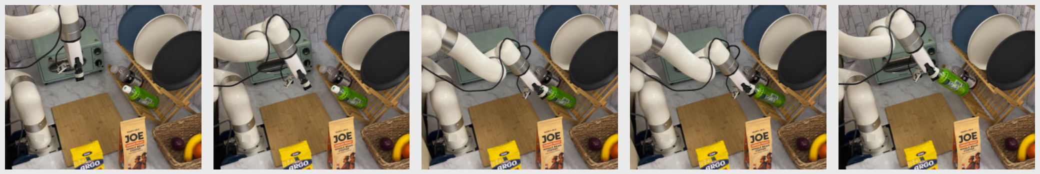
## D.2 Additional Analysis

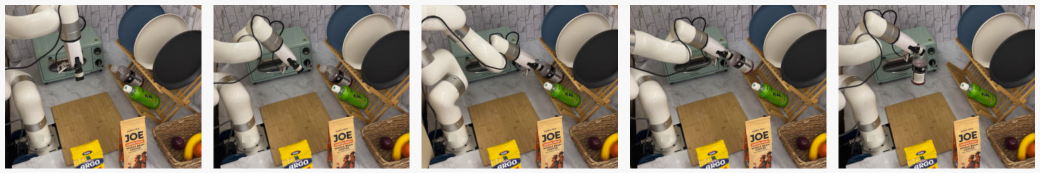
In addition to the analysis in Section 4.4, we provide further comparisons here to better justify our design choices.

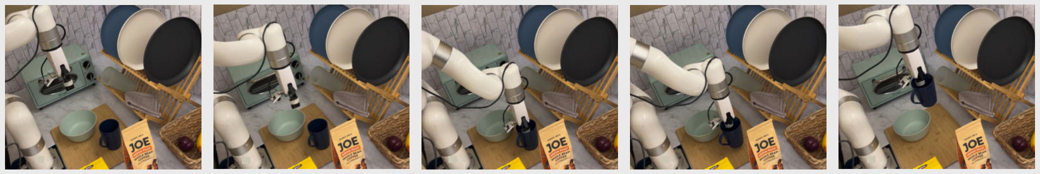
**Separate vs. Shared Vision Encoders** On the LIBERO-90 benchmark, environment observations include images from two camera views. Table 12 compares multi-task performance using either a common encoder for both views or separate view-specific encoders. While separate encoders provide a 2% boost in performance, this minor gain comes at the cost of a 15% increase in parameter count per camera view added (since the visual encoders comprise 1.5M parameters in our 10M parameter model). For our real-world experiments involving 4 camera views, this parameter increase would be even more significant. Therefore, in BAKU, we use a shared encoder for all views to keep the model compact, assisting with faster inference speeds.

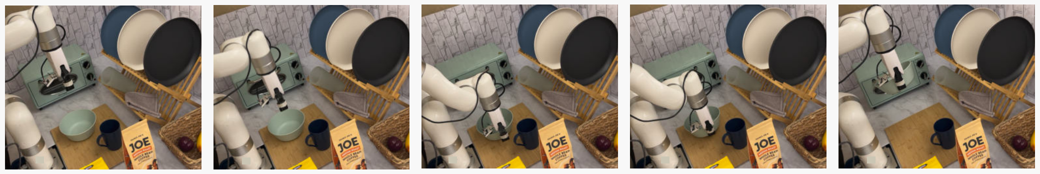

**Fetch glass from rack:** Fetch the glass from the lower rack.

**Fetch towel from rack:** Fetch the towel from the lower rack.


**Fetch tea bottle from rack:** Fetch the bottle of green tea from the lower rack.


**Fetch water bottle from rack:** Fetch the bottle of vitamin water from the lower rack.


**Pick blue mug:** Pick up the blue mug from the kitchen counter.


**Pick light blue bowl:** Pick up the light blue bowl from the kitchen counter.

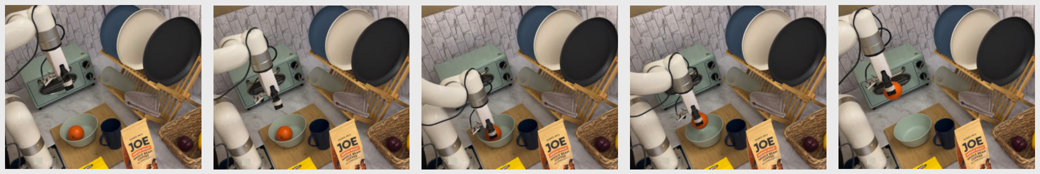
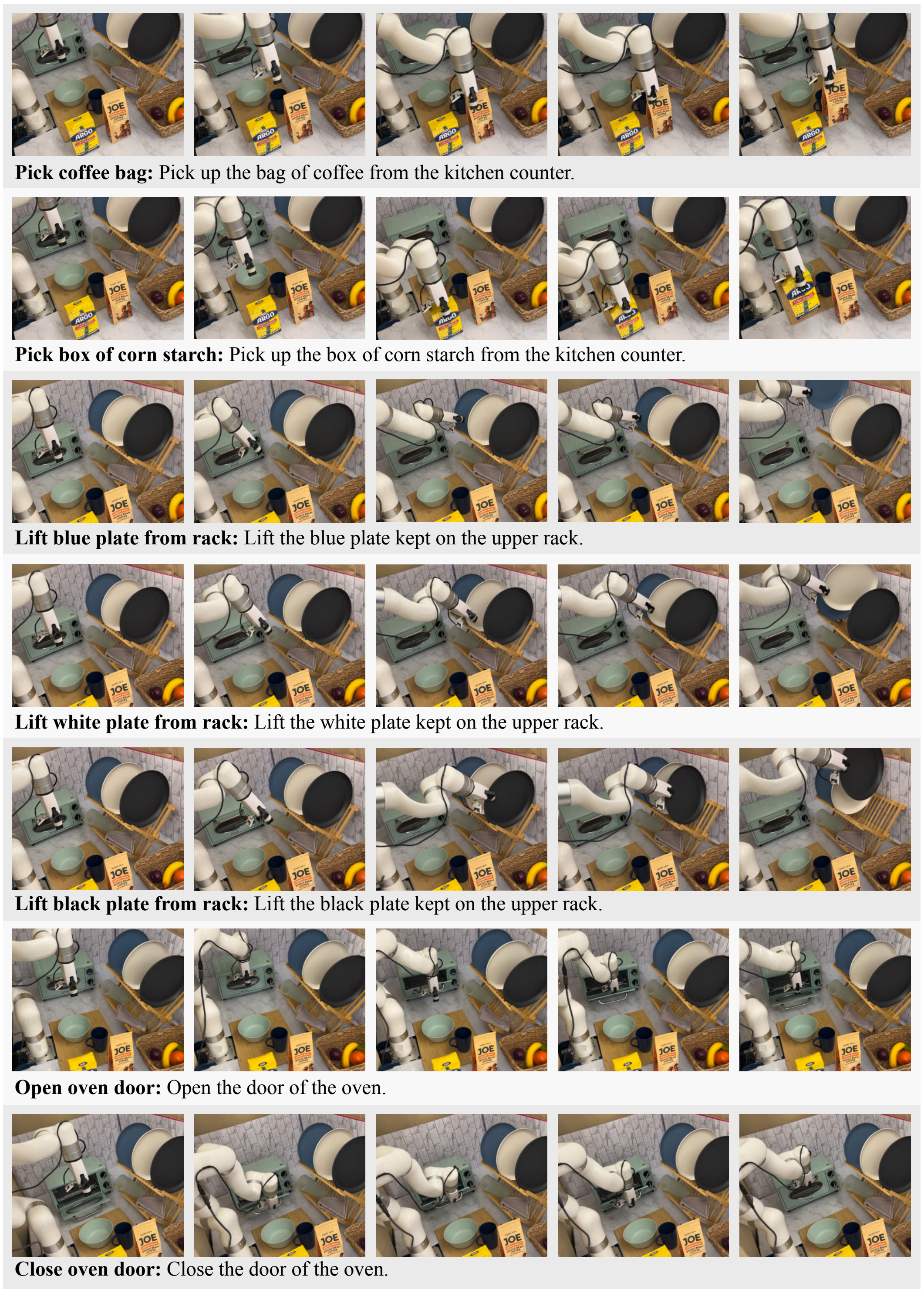

**Pick orange from bowl:** Pick up the orange from inside the light blue bowl kept on the kitchen counter.

Figure 5: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

**Data Efficiency Analysis** We analyze the performance of BAKU with varying number of demonstrations in Table 8 and Table 9. We observe that at each level of data availability, BAKU shows a significantly higher success rate than MT-ACT and RT-1.



**Pick coffee bag:** Pick up the bag of coffee from the kitchen counter.

**Pick box of corn starch:** Pick up the box of corn starch from the kitchen counter.

**Lift blue plate from rack:** Lift the blue plate kept on the upper rack.

**Lift white plate from rack:** Lift the white plate kept on the upper rack.

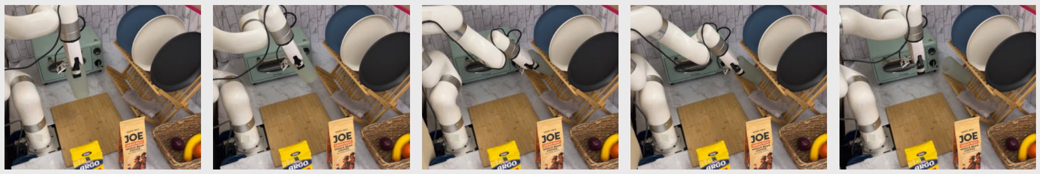
**Lift black plate from rack:** Lift the black plate kept on the upper rack.

**Open oven door:** Open the door of the oven.

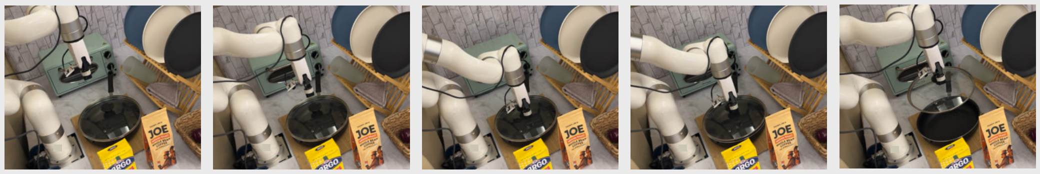
**Close oven door:** Close the door of the oven.

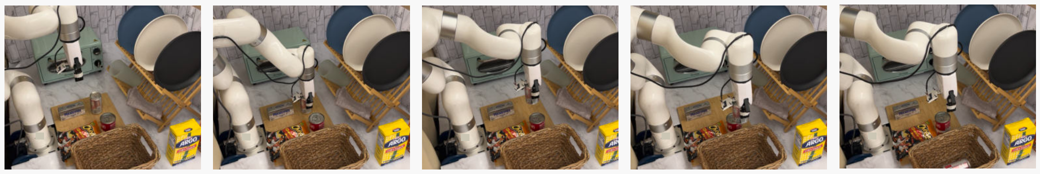
Figure 6: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

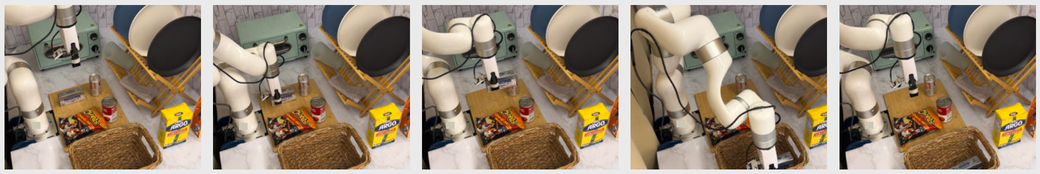
**Robustness to training seeds** We provide results on BAKU, RT-1, and MT-ACT across 3 seeds in Table 10. We observe that all three methods are robust to different seed values. Further, probabilistic approaches like GMM and diffusion might be sensitive to favorable seed values, and evaluating on a single seed might make the result unreliable. Thus, Table 11 includes results across 3 seeds on BAKU

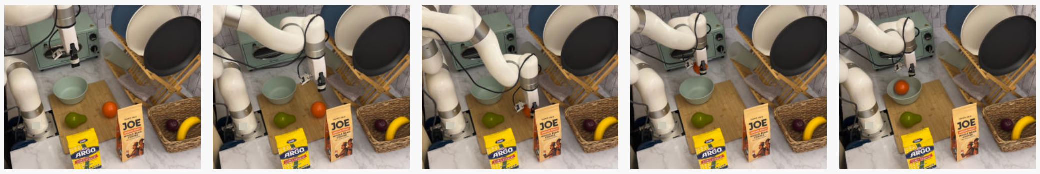

**Place glass on rack:** Place the glass on the lower rack.

**Wipe towel:** Wipe the cutting board with a towel.


**Lift pan lid:** Lift the lid of the pan kept on the kitchen counter.


**Put coke can in basket:** Pick up the can of coke and put it in the basket.


**Put cream cheese in basket:** Pick up the block of cream cheese and put it in the basket.


**Put orange in bowl:** Pick up the orange and put it in the bowl.

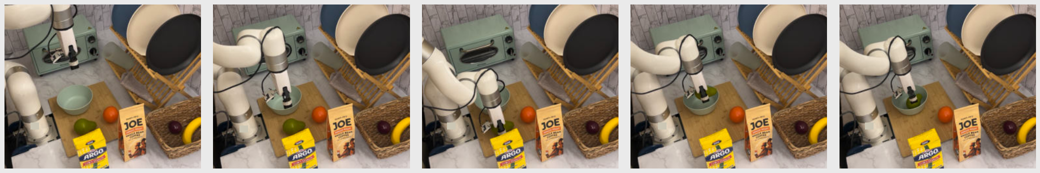
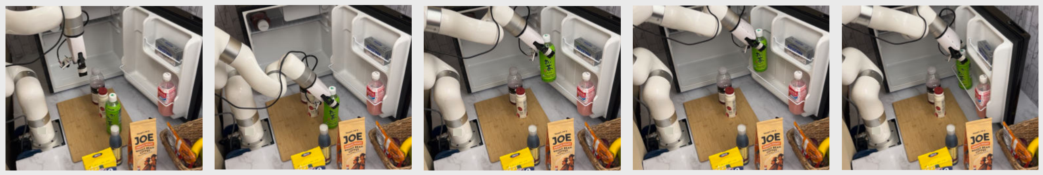
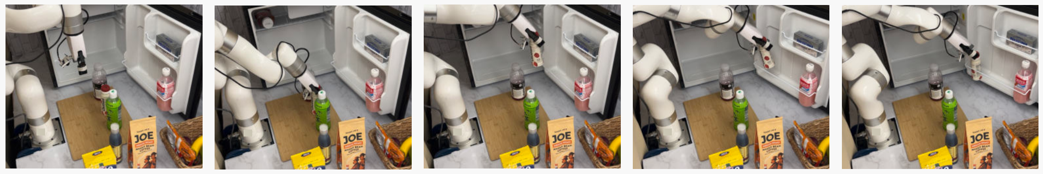
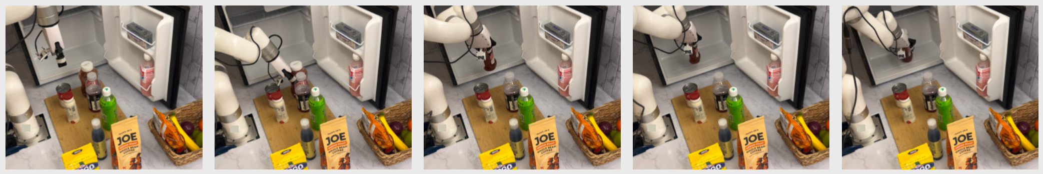

**Put pear in bowl:** Pick up the pear and put it in the bowl.

Figure 7: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

with different multimodal heads. We observe that BAKU with different action heads is robust to the value of the random seed. Due to limited compute and the large number of multi-task experiments, we provide these results on the LIBERO-90 and Metaworld benchmarks.

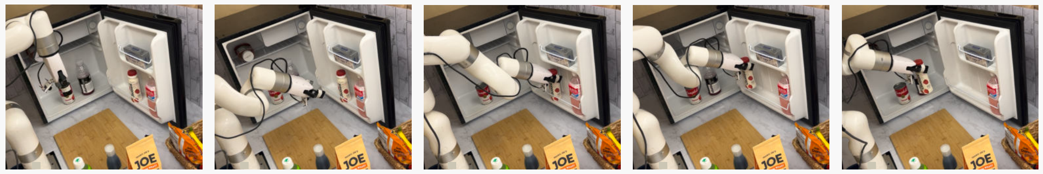

**Put tea bottle in fridge door:** Pick up the bottle of green tea and place it in the door of the fridge.


**Put yoghurt bottle in fridge door:** Pick up the bottle of yoghurt and place it in the door of the fridge.


**Put ketchup bottle inside fridge:** Pick up the bottle of tomato ketchup and put it inside the fridge.

**Put tomato can inside fridge:** Pick up the can of tomato soup and put it inside the fridge.

**Fetch tea bottle from fridge door:** Take the bottle of green tea out from the door of the fridge.


**Fetch yoghurt bottle from fridge door:** Take the bottle of yoghurt out from the door of the fridge.

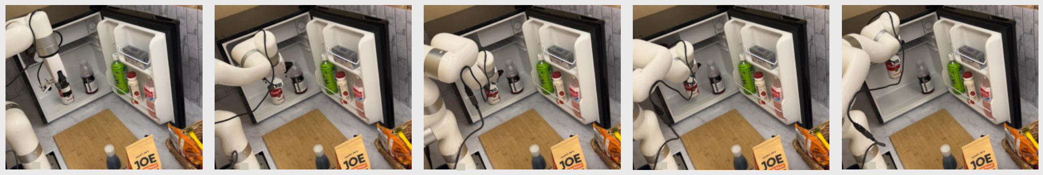
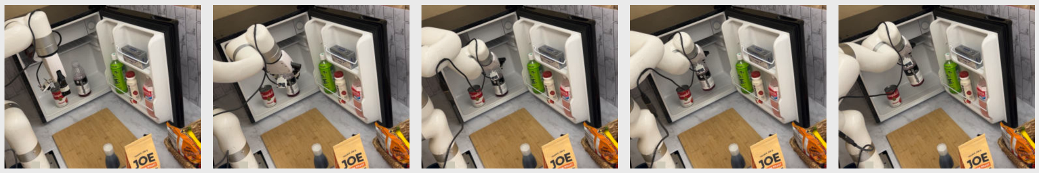

**Fetch tomato can from fridge:** Take the can of tomato soup out of the fridge.

Figure 8: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

**Observation trunk input** In our proposed architecture (see Section 3.4), the encoded observations from different modalities are passed individually as tokens into the observation trunk along with the action token to output the action feature representation. An alternative approach is to concatenate all the encoded inputs into a single vector and pass it through the observation trunk. As shown in Table 12, for Meta-World and DMC, which each have only a single input source, there is no


**Fetch water bottle from fridge:** Take the bottle of vitamin water out of the fridge.

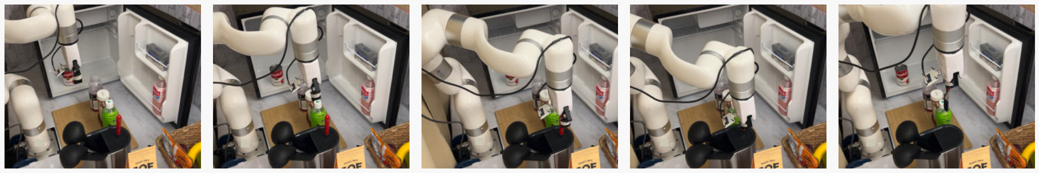
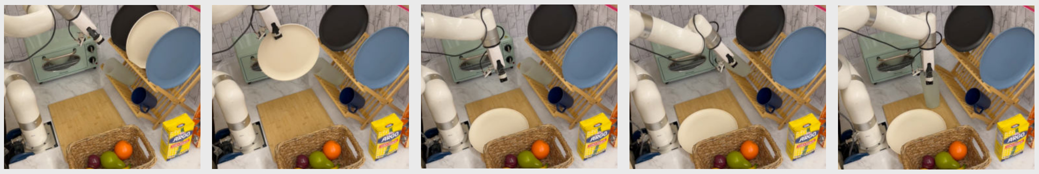
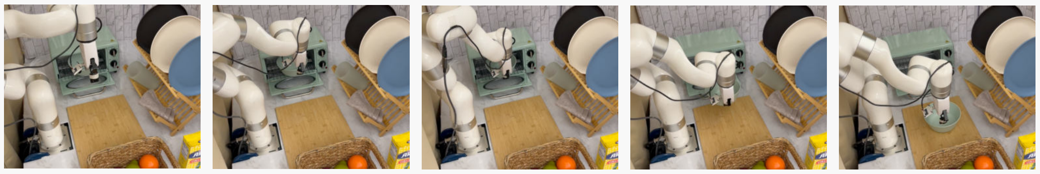

**Fetch knife from organizer:** Fetch the knife from the organizer placed on the kitchen counter.

Figure 9: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.


**Set up table:** Place the white plate and the glass on the kitchen counter.

**Pick broom and sweep:** Pick up the broom and sweep the cutting board.

**Pick towel and wipe:** Pick up the towel from the lower rack and wipe the cutting board.


**Take bowl out of the oven:** Take the bowl out of the oven and place it on the kitchen counter.

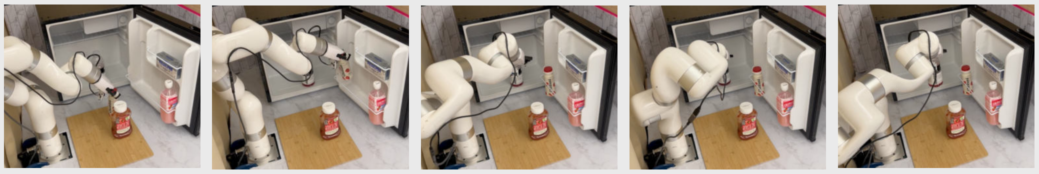

**Put yoghurt inside and take water bottle out of fridge:** Put the yoghurt in the door of the fridge and take out the bottle of water from inside the fridge.

Figure 10: Real-world policy rollouts showing BAKU's capability on long-horizon manipulation tasks.

Table 6: Real task-wise performance

| Task | Number of Demonstrations | Successes (out of 5) | | | |
|---|---|---|---|---|---|
| | | **RT-1** | **MTACT** | **Baku** | **Baku w/ VQ-BeT** |
| Fetch glass from rack | 20 | 5 | 5 | 5 | 5 |
| Fetch towel from rack | 28 | 5 | 2 | 5 | 5 |
| Fetch tea bottle from rack | 16 | 0 | 3 | 5 | 5 |
| Fetch water bottle from rack | 16 | 0 | 0 | 5 | 5 |
| Pick blue mug | 16 | 5 | 5 | 5 | 5 |
| Pick light blue bowl | 25 | 5 | 5 | 5 | 5 |
| Pick orange from bowl | 27 | 0 | 0 | 3 | 4 |
| Pick coffee bag | 19 | 3 | 5 | 5 | 5 |
| Pick box of corn starch | 14 | 0 | 3 | 5 | 5 |
| Lift blue plate from the rack | 18 | 0 | 4 | 5 | 5 |
| Lift white plate from the rack | 18 | 5 | 5 | 5 | 5 |
| Lift black plate from the rack | 12 | 2 | 3 | 5 | 5 |
| Open oven door | 17 | 0 | 0 | 0 | 3 |
| Close oven door | 27 | 0 | 3 | 3 | 4 |
| Place glass on rack | 19 | 5 | 5 | 5 | 5 |
| Wipe towel | 17 | 4 | 5 | 5 | 5 |
| Lift pan lid | 18 | 1 | 2 | 4 | 4 |
| Put coke can in basket | 19 | 0 | 0 | 3 | 3 |
| Put cream cheese in basket | 19 | 0 | 3 | 5 | 5 |
| Put orange into bowl | 14 | 0 | 0 | 4 | 5 |
| Put pear into bowl | 17 | 0 | 0 | 3 | 5 |
| Put tea bottle in fridge door | 18 | 0 | 0 | 1 | 0 |
| Put yoghurt bottle in fridge door | 17 | 3 | 5 | 3 | 5 |
| Put ketchup bottle inside fridge | 15 | 5 | 4 | 5 | 5 |
| Put tomato can inside fridge | 11 | 0 | 0 | 5 | 4 |
| Fetch tea bottle from fridge door | 11 | 5 | 5 | 5 | 5 |
| Fetch tomato can from fridge door | 11 | 0 | 1 | 5 | 5 |
| Fetch yoghurt bottle from fridge door | 10 | 0 | 3 | 5 | 4 |
| Fetch water bottle from fridge | 11 | 2 | 3 | 5 | 5 |
| Fetch knife from organizer | 20 | 0 | 5 | 5 | 5 |
| Mean | 17 | 1.83 | 2.8 | 4.3 | **4.53** |
| Mean success rate (out of 1) | – | 0.37 | 0.56 | 0.86 | **0.91** |

difference in performance, as expected. However, for LIBERO-90, which uses two camera views and the robot's proprioceptive state as inputs, there is a 3% absolute improvement in performance when using separate observation tokens as compare to a single concatenated vector.

## E Broader Impacts

In this work, we present BAKU, a simple and efficient transformer architecture for multi-task policy learning. This work takes an important step toward enabling more efficient training of generalist robotic agents capable of performing diverse tasks, reducing the need for large datasets of expert demonstrations which are costly and time-consuming to collect. Further, BAKU focuses on improving data efficiency by maximally leveraging available training data, which is particularly valuable in robotics where data collection is expensive.

Table 7: Real task-wise performance for long-horizon tasks

| Task | Number of Demonstrations | Successes (out of 5) | |
|---|---|---|---|
| | | MTACT | Baku |
| Set up table | 34 | 3 | 3 |
| Pick broom and sweep | 13 | 4 | 5 |
| Pick towel and wipe | 14 | 2 | 4 |
| Take bowl out of the oven | 18 | 5 | 5 |
| Put yoghurt inside and take water bottle out of fridge | 17 | 2 | 4 |
| Mean | 19 | 3.2 | **4.2** |
| Mean success rate (out of 1) | – | 0.64 | **0.84** |

Table 8: Data efficiency analysis on the LIBERO-90 benchmark.

| # Demos | RT-1 | MT-ACT | BAKU |
|---|---|---|---|
| 5 | 0 | 0.31 | **0.58** |
| 10 | 0.01 | 0.48 | **0.71** |
| 25 | 0.04 | 0.49 | **0.83** |
| 50 | 0.16 | 0.54 | **0.9** |

Table 9: Data efficiency analysis on the Meta-World benchmark.

| # Demos | RT-1 | MT-ACT | BAKU |
|---|---|---|---|
| 5 | 0.40 | 0.07 | **0.59** |
| 10 | 0.49 | 0.10 | **0.67** |
| 25 | 0.62 | 0.11 | **0.76** |
| 35 | 0.65 | 0.13 | **0.79** |

Table 10: Performance of multi-task policies learned using BAKU on LIBERO-90 and Meta-World. We report the mean and standard deviation for each variant across 3 seeds.

| Method | LIBERO-90 (90 tasks) | Meta-World (30 tasks) |
|---|---|---|
| RT-1 | $0.14 \pm 0.02$ | $0.64 \pm 0.01$ |
| MTACT | $0.55 \pm 0.01$ | $0.12 \pm 0.01$ |
| **BAKU (Ours)** | $\mathbf{0.89 \pm 0.01}$ | $\mathbf{0.81 \pm 0.02}$ |

Table 11: Performance of BAKU with different action heads on LIBERO-90 and Meta-World. We report the mean and standard deviation for each variant across 3 seeds.

| Action Head | LIBERO-90 | Meta-World |
|---|---|---|
| MLP | $0.89 \pm 0.01$ | $\mathbf{0.81 \pm 0.02}$ |
| GMM | $0.83 \pm 0.02$ | $0.64 \pm 0.02$ |
| BeT | $0.88 \pm 0.01$ | $0.77 \pm 0.01$ |
| VQ-BeT | $\mathbf{0.9 \pm 0.01}$ | $0.78 \pm 0.005$ |
| Diffusion | $0.88 \pm 0.01$ | $0.64 \pm 0.01$ |

Table 12: Study of design decisions for the model architecture that affects multi-task performance.

| Category | Variant | LIBERO-90 | Meta-World | DMC |
|---|---|---|---|---|
| Separate vs. Shared Vision Encoders | Common | 0.90 | – | – |
| | Separate | **0.92** | – | – |
| Observation Trunk Input | Separate | **0.90** | **0.79** | **0.70** |
| | Concatenated | 0.87 | **0.79** | **0.70** |