A Implementation details

Here we lay down the details of the data collection, training, and testing process.

Collecting human play data and training details. The human play data is collected by letting a human operator directly interact with the scene with a single hand for 10 minutes for each scene. The entire trajectory $\boldsymbol{\tau}$ is recorded at the speed of 60 frames per second and is used without cutting or labeling. The 3D hand trajectory is detected with an off-the-shelf multi-view human hand tracker [49]. The total number of video frames within 10 minutes of human play video is around 36k. We train one latent planner for each environment with the collected human play data. For the multi-environment setup (for the experiments in Tab. 3), we merge the human play data from each scene to train a single latent planner. The latent planner contains two ResNet-18 [54] networks for image processing and MLP-based encoder-decoder networks together with a GMM model, which has $K=5$ distribution components. We train 100k iterations for the latent planner which takes a single GPU machine for 12 hours.

Collecting robot demonstrations and training details. The robot teleoperation data is collected with an IMU-based phone teleoperation system RoboTurk [55]. The control frequency of the robot arm is 17-20Hz and the gripper is controlled at 2Hz. For each task, we collect 20 demonstrations. In the experiments, we also have a 40 demonstration dataset for testing the sample efficiency of different approaches. The robot policy model is a GPT-style transformer [52], which consists of four multi-head layers with four heads. We train 100k iterations for the policy with a single GPU machine in 12 hours. For a fair comparison with our method, the baseline approaches trained without human play data have five more demonstrations during training the latent planner $\mathcal{P}$ and the low-level policy π.

Video prompting. In this work, we use a one-shot video $\mathcal{V}$ (either human video $\mathcal{V}^h$ or robot video $\mathcal{V}^r$) to prompt the pre-trained latent planner to generate corresponding plans $p_t=\mathcal{P}(o_t,g_t,l_t),g_t\in\mathcal{V}$. During training (Fig. 2(b)), we specify the goal image g_t^r ($g_t^r\in\mathcal{V}^r$) as the frame H steps after the input observation o_t^r in the robot demonstration. H is a uniformly sampled integer number within the range of [200,600], which equals 10-30 seconds in wall-clock time. l_t here is the 3D location of the robot's end-effector. During inference (Fig. 2(c)), we assume access to a task video (either human or robot video) which is used as a source of goal images. The goal image will start at the 200 frame of the task video and move to the next i frame after each step. We use $i=1$ in all our experiments. Based on the inputs, the latent planner generates a latent plan feature embedding p_t of shape $\mathbb{R}^{1\times d}$, which is used as guidance for the low-level robot policy.

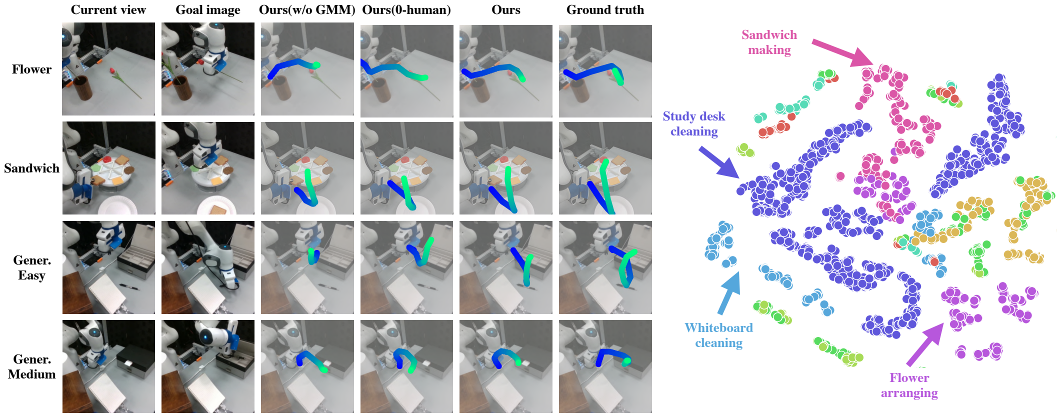
Data visualization. We visualize the collected human play data and robot demonstration data in Tab. 9. For the human play data, we use an off-the-shelf hand detector [49] to localize the hand's 2D location on the left and right image frame, which are visualized as red bounding boxes in Tab 9. For the robot demonstration data, we directly project the 3D location of the robot end-effector to the left and right image frames, which are visualized as blue bounding boxes in Tab 9.

Testing. We perform real-time inference on a Franka Emika robot arm with a control frequency of 17Hz—directly from raw image inputs to 6-DoF robot end-effector and gripper control commands with our trained models. The robot is controlled with the Operational Space Control (OSC) [56].

B Experiment setups

Environments. We design six environments with a total of 14 tasks for a Franka Emika robot arm, as illustrated in Fig. 3. These environments feature several manipulation challenges, such as contact-rich tool manipulation (cleaning the whiteboard), articulated-object manipulation (opening the oven and the box on the study desk), high-precision tasks (inserting flowers and turning on the lamp by pressing the button), and deformable object manipulation (folding cloth).

Tasks. We design three tasks in the Kitchen environment and four tasks in the Study desk environment. All these tasks have different goals. In this work, we focus on long-horizon tasks that require the robot to complete several subgoals. To better analyze the performance of each method, we define the *Subgoal* task category that only counts whether the first subgoal of the task has been achieved and the *Long horizon* task category which is the full task. In the Study desk environment, we design three tasks for testing the



(a) Trajectory prediction results decoded from the latent plans

(b) t-SNE visualization of the latent plans

Figure 6: Qualitative visualization of the learned latent plan. (**a**) Visualization of the trajectory prediction results decoded from the latent plans learned by different methods. The fading color of the trajectory from blue to green indicates the time step from 1 to 10. (**b**) t-SNE visualization of latent plans, the latent plans of the same task tend to cluster in the latent space.

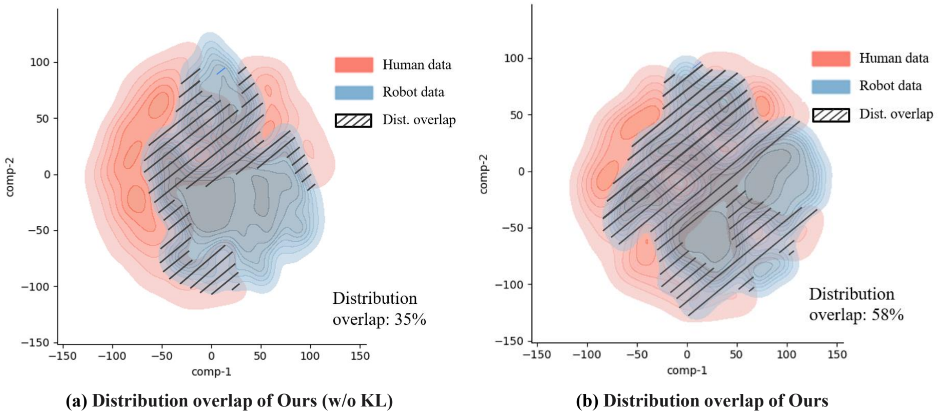
compositional generalization ability of the models to novel task goal sequences, which are not included in the training dataset. These three tasks are classified as *Easy*, *Medium*, and *Hard* depending on their difference compared to the training tasks. The *Easy* task is a simple concatenation of two trained tasks and their subgoals. The *Medium* task contains an unseen composition of a pair of subgoals that is not covered by any trained tasks, i.e., the transition from subgoal A to subgoal B is new. The model needs to generate novel motions to reach these subgoals. The *Hard* task contains two such unseen transitions. For the rest four environment, each scene has one task goal and features different types of challenges in manipulation, *e.g.*, generalization to new spatial configuration, extremely long horizon, and deformable object manipulation.

Baselines. We compare with five prior approaches: (1). GC-BC (BC-RNN) [20]: Goal-conditioned behavior cloning algorithm [5] implemented with recurrent neural networks (RNN) [57]. (2). GC-BC (BC-trans) [52]: Another goal-conditioned behavior cloning algorithm implemented with GPT-like transformer architecture. (3). C-BeT [6]: Goal-conditioned learning from teleoperated robot play data algorithm implemented with Behavior Transformer (BeT) [53]. (4). LMP [5]: A learning from teleoperated robot play data algorithm designed to handle variability in the play data by learning an embedding space. LMP (single) is a variant by training each task with a separate model. (5). R3M-BC [40]: A goal-conditioned imitation learning framework that leverages R3M visual representation pre-trained with internet-scale human video dataset Ego4D [42]. R3M-BC (single) is a variant by training each task with a separate model.

Ablations. We compare four variants of our model to showcase the effectiveness of our architecture design: (1). Ours: MIMICPLAY with full collection (10 min) of human play data. Ours (single) is a variant by training each task with a separate model. (2). Ours (0-human): variant of our model without using human play data. The pre-trained latent plan space is trained only with the teleoperated robot demonstrations. (3). Ours (50-human): variant of our model where the latent planner is trained with 50% of human play data (5 min). (4). Ours (w/o GMM): variant without using the GMM model for learning the latent plan space from human play data. (5). Ours (w/o KL): Our approach without using KL loss for addressing the visual gap between human and robot data when pre-training the latent planner.

C Supplementary Experiment Results

Visualization of the trajectory prediction results. We visualize the 3D trajectory decoded from the latent plan by projecting it onto the 2D image in Fig. 6. In the last two rows, we showcase the results of



(a) Distribution overlap of Ours (w/o KL)

(b) Distribution overlap of Ours

Figure 7: t-SNE visualization of the generated feature embeddings by taking human data and robot data as inputs. The slashes refer to the overlap region of two data distributions. (**a**) Feature visualization results of our method without using KL divergence loss. (**b**) Feature visualization results of our method with KL divergence loss. Our approach covers 23% more area than the baseline.

two unseen subgoal transitions. The trajectory generated by our model is most similar to the ground truth trajectory, while Ours (0-human) is overfitted to the subgoal transitions in the training set and generates the wrong latent plan. For instance, in the training data, the robot only learns to open the box after turning off the lamp, meanwhile in the *Easy* setting of generalization tasks, the robot is prompted to pick up the pen after turning off the lamp. Ours (0-human) variant still outputs a latent plan to open the box, which causes the task to fail since the box is already open.

Visualization of the learned latent plans. We use t-SNE [58] to visualize the generated latent plans conditioned on different tasks, as shown in Fig. 6(b). We find that the latent plans of the same task tend to cluster in the latent space, which shows the effectiveness of our approach in distinguishing different tasks.

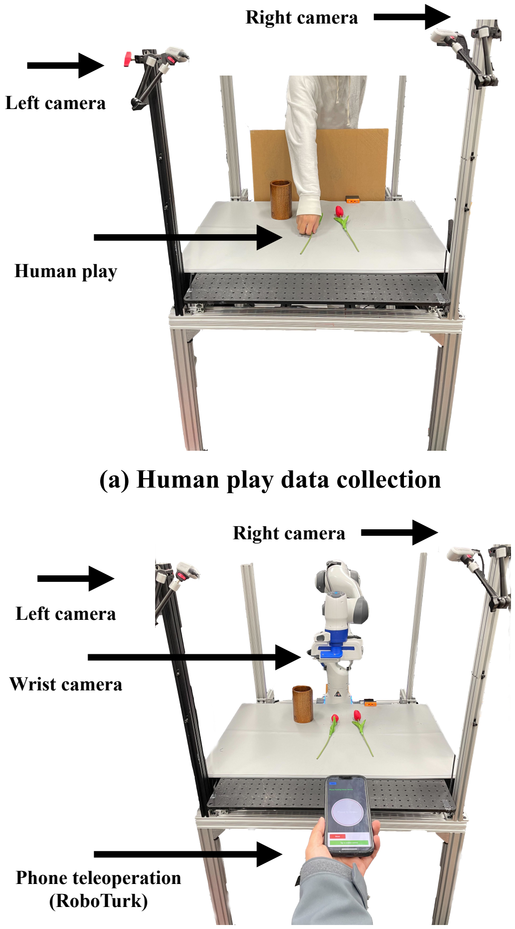
Transformer architecture helps multi-task learning. In Tab. 1, GC-BC (BC-trans) with the GPT transformer architecture outperforms GC-BC (BC-RNN) by more than 30% in a 40-demos Subgoal setting. However, the performance of GC-BC (BC-trans) quickly drops to the same level as GC-BC (BC-RNN) in 20-demos settings. The result showcases that training vision-based transformer policy end-to-end requires more data.

Analysis of the visual gap between human and robot data. As is introduced in the method Sec. 3.2, to minimize the visual gap between human play data and robot demonstration data, we use a KL divergence loss over the feature embeddings outputted by the visual encoders. In Fig. 7, we use t-SNE to process and visualize the learned feature embeddings generated by Ours and the model variant Ours (w/o KL) on the 2D distribution plots. To better visualize the distribution overlap, we use slashes to highlight the overlap area in both plots. We observe that our approach with KL loss has a 23% larger overlap between the human data and the robot data compared to Ours (w/o KL). This result showcases the effectiveness of our KL divergence loss and supports the result in Tab. 2 (Ours (w/o KL) is inferior to Ours in task success rate).

D Details of system setups

We illustrate the system designs for the data collection in Fig. 8. The human play data is collected by having a human operator directly interact with the environment with one of its hands (Fig. 8(a)). The left and right cameras record the video at the speed of 100 frames per second. During the collection process of human play data, no specific task goal is given and the human operator freely interacts with the scene for interesting behaviors based on its curiosity. For each scene in our experiments, we collect 10 minutes of human play data.

The robot teleportation demonstration is collected with a phone teleoperation system RoboTurk [55] (Fig. 8(b)). The left, right, and end-effector wrist cameras record the video at the speed of 20 frames per second, which is aligned with the control speed of the robot arm (20Hz). Each sequence of robot demonstration has a pre-defined task goal. During the data collection, the human demonstrator completes the assigned sub-goals one by one and finally solves the whole task. For each training task in our experiments, we collect 20 demonstrations. In the Kitchen environment, we collect 40 demonstrations for each task to figure out which approach is more sample inefficiency.



(a) Human play data collection

(b) Robot demonstration data collection

Figure 8: System setups for the data collection. (**a**) Human play data collection. A human operator directly interacts with the scene with one of its hand and perform interesting behaviors based on its curiosity without a specific task goal. (**b**) Robot demonstration data collection. A human demonstrator uses a phone teleoperation system to control the 6 DoF robot end-effector. The gripper of the robot is controlled by pressing a button on the phone interface.

E Details of the task designs

The definition of our long-horizon tasks is listed below. For each task, the initial state and subgoals are pre-defined. The whole task is completed if and only if all subgoals are completed in the correct order.

E.1 Kitchen

- *Task-1*
  - Initial state: A drawer is placed on the left side of the table. The drawer is not fully open and contains pumpkin and lettuce. A closed microwave oven is placed on the right side of the desktop. A bowl and a stove are placed on the lower edge of the tabletop. There is a carrot inside the bowl. A pan is placed on top of the stove.
  - Subgoals: a) Open the microwave oven door. b) Pull out the microwave oven tray. c) Pick up the bowl. d) Place the bowl on the microwave tray.
- *Task-2*
  - Initial state: same as Kitchen Task-1.
  - Subgoals: a) Open the drawer. b) Pick up the carrot. c) Put the carrots in the drawer.
- *Task-3*
  - Initial state: same as Kitchen Task-1.
  - Subgoals: a) Pick up the pan. b) Place the pan on the table. c) Pick up the bowl. d) Place the bowl on the stove.

E.2 Study desk

- *Task-1*
  - Initial state: The book is on the rack. The lamp is on. The box is opened and closed in a random state. The pen is located either in the center of the table or in the box.
  - Subgoals: a) Turn off the lamps. b) Pick up the book. c) Place the book on the shelf position.
- *Task-2*
  - Initial state: The location of the book is either on the shelf or on the rack. The lamp is off. The box is closed. The pen is in the center of the table.
  - Subgoals: a) Turn on the lamps. b) Open the box. c) Pick up the pen. d) Put it in the box.
- *Task-3*
  - Initial state: The book is on the rack. The state of the lamp is random. The box is closed. The pen is in the center of the table.
  - Subgoal a) Open the box. b) Pick up the pen. c) Place the pen in the box. d) Pick up the book. e) Place the book on the shelf.
- *Task-4*
  - Initial state: The location of the book is either on the shelf or on the rack. The lamp is on. The box is closed. The pen is located either in the center of the table or in the box.
  - Subgoals: a) Open the box. b) Turn off the lamp.
- *Easy*
  - Initial state: The location of the book is either on the shelf or on the rack. The lamp is off. The box is closed. The pen is located either in the center of the table or in the box.
  - Subgoals: a) Turn on the lamp. b) Open the box. c) Turn off the lamp.
- *Medium*
  - Initial state: The location of the book is either on the shelf or on the rack. The lamp is on. The box is closed. The pen is in the center of the table.
  - Subgoals: a) Open the box. b) Turn off the lamp. c) Pick up the pen. d) Place the pen in the box.
- *Hard*
  - Initial state: The book is on the shelf. The lamp is on. The box is closed. The pen is located either in the center of the table or in the box.
  - Subgoals: a) Turn off the lamp. b) Open the box. c) Pick up the book. d) Place the book on the shelf.

E.3 Flower

- Initial state: Two flowers and a vase are placed on the table. The vase will randomly be placed on the top left or top right corner of the table.
- Subgoals: a) Picking up a flower. b) Insert the flower into the vase. c) Pick up the other flower. d) Insert the flower into the vase.

E.4 Whiteboard

- Initial state: A whiteboard and board eraser are placed on the table. The board eraser is placed on the left side of the whiteboard.
- Subgoals: a) Pick up the board eraser. b) Moves over the curve line. c) Erase the curve line. d) Return the eraser to the original location.

E.5 Sandwich

- Initial state: A circular ingredient selector is placed in the upper right corner of the table. Half of the circle holds ingredients for a sandwich (bread, lettuce, sliced tomato) and half holds ingredients for a cheeseburger (bread, cheese, burger patty). A white plate is placed in the lower left corner of the table.
- Subgoals for a sandwich: a) Rotate the ingredient selector to the right position. Pick up a piece of bread from it and place it on the plate. b) Rotate the ingredient selector to the correct position. Pick up the lettuce and place it on top of the bread. c) Rotate the ingredient selector to the right position. Pick up the sliced tomato and place it on top of the lettuce. d) Rotate the ingredient selector to the right position. Pick up another piece of bread and place it on top of the tomato.

E.6 Cloth

- Initial state: An unfolded brown cloth is randomly placed on the table.
- Subgoals: a) The robot folds the cloth in half once to become 1/2 of its original size. b) The robot folds the cloth once more to become 1/4 of its original size.

F Training hyperparameters

We list the hyperparameters for training the models in Tab. 4 for the latent planner $\mathcal{P}$ and Tab. 5 for the robot policy π. The hyperparameters that are named starting with GMM are related to the MLP-based GMM model. The hyperparameters that are named starting with GPT are related to the transformer architecture. We also list the hyperparameters for the baseline GC-BC (BC-trans) in Tab. 6.

Hyperparameter	Default
Batch Size	16
Learning Rate (LR)	1e-4
Num Epoch	1000
LR Decay	None
KL Weights λ	1000
MLP Dims	[400, 400]
Image Encoder - Left View	ResNet-18
Image Encoder - Right View	ResNet-18
Image Feature Dim	64
GMM Num Modes	5
GMM Min Std	0.0001
GMM Std Activation	Softplus

Table 4: Hyperparameters - Ours (Latent Planner $\mathcal{P}$)

Hyperparameter	Default
Batch Size	16
Learning rate (LR)	1e-4
Num Epoch	1000
Train Seq Length	10
LR Decay Factor	0.1
LR Decay Epoch	[300, 600]
MLP Dims	[400, 400]
Image Encoder - Wrist View	ResNet-18
Image Feature Dim	64
GMM Num Modes	5
GMM Min Std	0.01
GMM Std Activation	Softplus
GPT Block Size	10
GPT Num Head	4
GPT Num Layer	4
GPT Embed Size	656
GPT Dropout Rate	0.1
GPT MLP Dims	[656, 128]

Table 5: Hyperparameters - Ours (Robot Policy π)

Hyperparameter	Default
Batch Size	16
Learning rate (LR)	1e-4
Num Epoch	1000
Train Seq Length	10
LR Decay Factor	0.1
LR Decay Epoch	[300, 600]
MLP Dims	[400, 400]
Image Encoder - Wrist View	ResNet-18
Image Encoder - Left View	ResNet-18
Image Encoder - Right View	ResNet-18
Image Feature Dim	64
GMM Num Modes	5
GMM Min Std	0.01
GMM Std Activation	Softplus
GPT Block Size	10
GPT Num Head	4
GPT Num Layer	4
GPT Embed Size	656
GPT Dropout Rate	0.1
GPT MLP Dims	[656, 128]

Table 6: Hyperparameters - GC-BC (BC-trans)

G Network Architecture

Transformer-based policy network. The embedding sequence of T time steps is represented as $s_{[t:t+T]} = [w_t, e_t, p_t, \cdots, w_{t+T}, e_{t+T}, p_{t+T}]$, which passes through a transformer architecture [51]. The transformer model f_{trans} processes the input embeddings using its N layers of self-attention and feed-forward neural networks. Given an embedding sequence of $T-1$ time steps, f_{trans} generates the embedding of trajectory prediction in an autoregressive way - $x_T = f_{\text{trans}}(w_{1:T-1}, e_{1:T-1}, p_{1:T-1})$, where x_T is the predicted action embedding at time step T. The transformer architecture uses the multi-head self-attention mechanism to gather context and dependencies from the entire history trajectory at each step. The final robot control commands a_t are computed by processing the action feature x_t with a two-layer fully-connected network. To handle the multimodal distribution of robot actions, we also use a MLP-based GMM model [50] for the action generation.

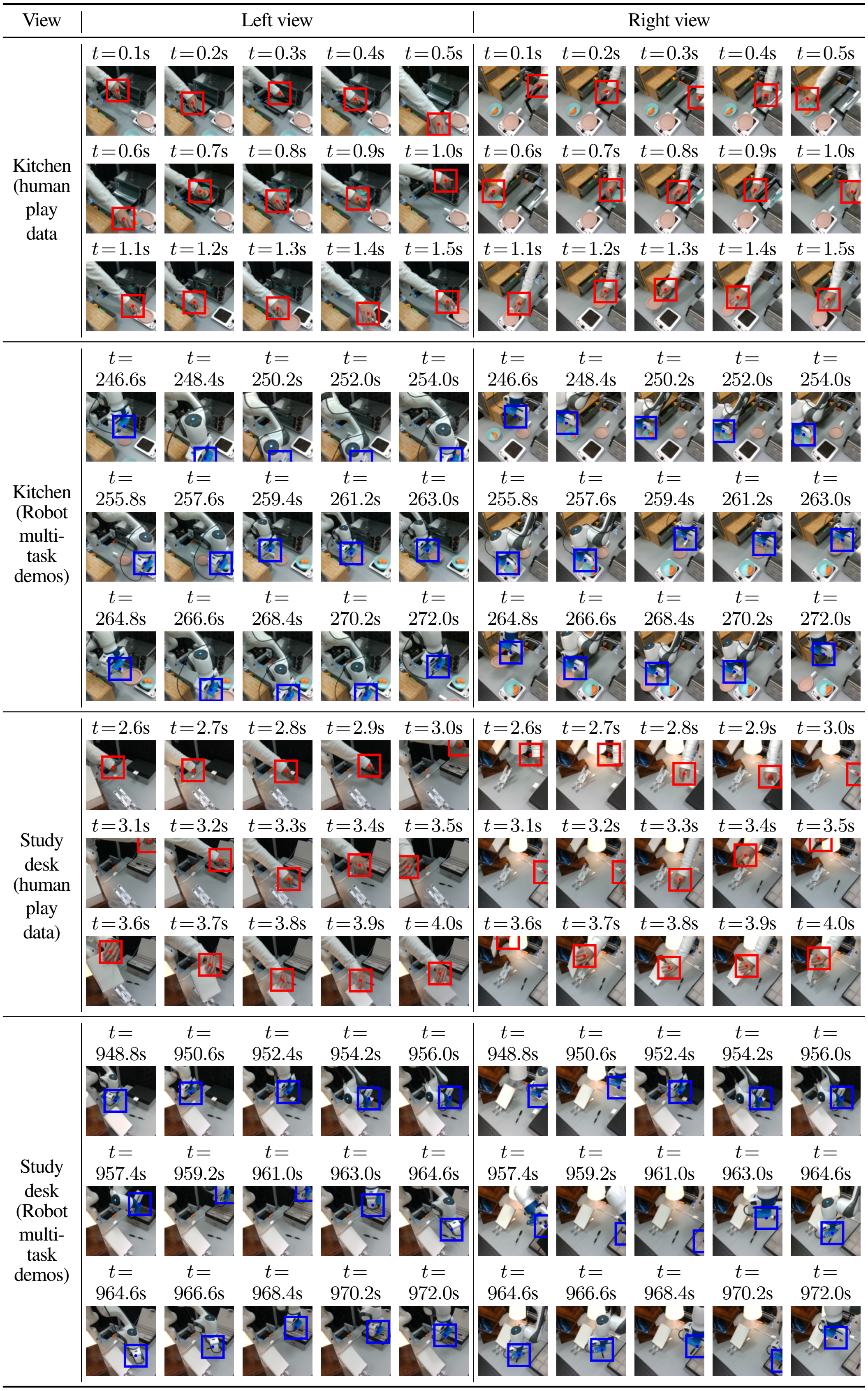

View
Left view
Right view
Kitchen (human play data
t=0.1s t=0.2s t=0.3s t=0.4s t=0.5s t=0.1s t=0.2s t=0.3s t=0.4s t=0.5s
t=0.6s t=0.7s t=0.8s t=0.9s t=1.0s t=0.6s t=0.7s t=0.8s t=0.9s t=1.0s
t=1.1s t=1.2s t=1.3s t=1.4s t=1.5s t=1.1s t=1.2s t=1.3s t=1.4s t=1.5s
Kitchen (Robot multi-task demos)
t= 246.6s t= 248.4s t= 250.2s t= 252.0s t= 254.0s t= 246.6s t= 248.4s t= 250.2s t= 252.0s t= 254.0s
t= 255.8s t= 257.6s t= 259.4s t= 261.2s t= 263.0s t= 255.8s t= 257.6s t= 259.4s t= 261.2s t= 263.0s
t= 264.8s t= 266.6s t= 268.4s t= 270.2s t= 272.0s t= 264.8s t= 266.6s t= 268.4s t= 270.2s t= 272.0s
Study desk (human play data)
t=2.6s t=2.7s t=2.8s t=2.9s t=3.0s t=2.6s t=2.7s t=2.8s t=2.9s t=3.0s
t=3.1s t=3.2s t=3.3s t=3.4s t=3.5s t=3.1s t=3.2s t=3.3s t=3.4s t=3.5s
t=3.6s t=3.7s t=3.8s t=3.9s t=4.0s t=3.6s t=3.7s t=3.8s t=3.9s t=4.0s
Study desk (Robot multi-task demos)
t= 948.8s t= 950.6s t= 952.4s t= 954.2s t= 956.0s t= 948.8s t= 950.6s t= 952.4s t= 954.2s t= 956.0s
t= 957.4s t= 959.2s t= 961.0s t= 963.0s t= 964.6s t= 957.4s t= 959.2s t= 961.0s t= 963.0s t= 964.6s
t= 964.6s t= 966.6s t= 968.4s t= 970.2s t= 972.0s t= 964.6s t= 966.6s t= 968.4s t= 970.2s t= 972.0s

Figure 9: Dataset visualization.