Appendix

A Website

Videos and code for our approach can be found at https://sites.google.com/view/goal-instructions.

B Environment Details

We provide more details on the real-world environment in this section.

B.1 Robot

We use a 6DOF WidowX 250 robot with a 1DOF parallel-jaw gripper. We install the robot on a tabletop where it can reach and manipulate objects within an environment set up in front of it. The robot receives inputs from a Logitech C920 RGB camera installed in an over-the-shoulder view. The images are passed into the policy at a 128 x 128, and the control frequency is 5Hz. Teleoperation data is collected with a Meta Quest 2 VR headset that controls the robot.

B.2 Dataset Details

The dataset consists of trajectories collected from 24 different environments, which includes kitchen-, sink-, and tabletop-themed manipulation environments. The dataset features around 100 objects, including containers, utensils, toy food items, towels, and other kitchen-themed objects. It includes demonstrations of 13 high-level skills (pick and place, sweep, etc.) applied to different objects. Out of the 54k total trajectories, 7k are annotated with language instructions. Around 44k of the trajectories are expert demonstrations and around 10k are collected by a scripted policy.

C Method Details

C.1 Policy Network

Our policy network $\pi_\theta(a|s, z)$ uses a ResNet-34 architecture. To condition on the task embedding z, it is first passed through 2 fully connected layers. Then, the policy network is conditioned on the embedding using FiLM layers, which are applied at the end of every block throughout the ResNet. The image encoding is then passed into a fully connected network to predict the action distribution. The policy network predicts the action mean, and we use a fixed standard deviation.

C.2 CLIP Model Surgery

Instead of separately encoding s_0 and g inside f_φ, we perform a "surgery" to the CLIP model to enable it to take (s_0, g) as inputs while keeping most of its pre-trained network weights as intact as possible. Specifically, we clone the weight matrix W_{in} of the first layer in the pre-trained CLIP and concatenate them along the channel dimension to be $[W_{\text{in}}; W_{\text{in}}]$, creating a model that can accept the stacked $[s_0, g]$ as inputs. We also halve the values of this new weight matrix to make it $W'_{\text{in}} = [W_{\text{in}}/2; W_{\text{in}}/2]$, ensuring its output $0.5(W_{\text{in}}s_0 + W_{\text{in}}g)$ will follow a distribution similar to the output by the original first layer $W_{\text{in}}s_0$. While this surgery alone cannot perfectly close the gap, the resultant modified encoder can serve as a capable initialization for the transition encoder h_ψ. We further fine-tune h_ψ on the labeled robot dataset $\mathcal{D}_L$ using the aforementioned method to adapt it for instruction-following tasks.

C.3 Negative Sampling

For training the contrastive objective on $\mathcal{D}_L$, our batch sampling strategy is non-standard. We use 2 dataloaders in parallel; the first samples from shuffled trajectories, while the second iterates through

trajectories in the order that they are stored in the dataset. Each samples batches of 128 trajectories and they are concatenated to produce a batch size of 256. The reason for this is that if we were to use a standard sampling strategy, most examples in a batch would be from different scenes. This is not useful for the contrastive loss because the representations would just learn to distinguish tasks based on the set of objects that appear. The robot benefits from being able to distinguish different tasks in the same scene, so we try to include many trajectories from the same scene in each batch. Using an unshuffled dataloder is a convenient way to achieve this since trajectories from the same scene are stored together. This can be considered a form of negative mining for the contrastive learning stage.

C.4 Instruction Augmentation

In order to increase the diversity of language annotations, we augment our natural language annotations using GPT-3.5. Through the API, we query the gpt-3.5-turbo model to generate paraphrases of instructions in our dataset. We generate 5 paraphrases per instruction and sample from them randomly during training. An example prompt and response are shown below. We found in preliminary experiments that using augmented instructions slightly improved language generalization, so we keep this augmentation for all models and baselines.

```
Prompt:
    Generate 5 variations of the following command:
    "put the mushroom in the metal pot"
    Number them like 1. 2. 3.
    Be concise and use synonyms.
Response:
    1. Place the fungus in the metallic container.
    2. Insert the mushroom into the steel vessel.
    3. Set the toadstool inside the iron cauldron.
    4. Position the champignon within the tin pot.
    5. Place the fungi in the metallic kettle.
```

C.5 Goal Relabeling

For unlabeled trajectories in $\mathcal{D}_U$, we use a simple goal relabeling strategy: with 50% probability, we use the final achieved state as the goal, and with 50 % probability we uniformly sample an intermediate state in the trajectory to use as the goal. We do not relabel the annotated trajectories in $\mathcal{D}_L$.

C.6 Hyperparameters

When training the task encoders using the contrastive learning objective, we use a batch size of 256. We reduce the batch size to 128 when we train the policy network. We use the Adam optimizer with a learning rate schedule that uses linear warmup and cosine decay. The peak learning rate is 3e-4 for all parameters except the CLIP ViT encoders, for which we use 3e-5. We use 2000 warmup steps and 2e6 decay steps for the learning rate schedule. When we jointly train the alignment and behavioral cloning losses, we use a weight of 1.0 on both terms. These hyperparameters were found through random search. We train our models for 150k steps, which takes around 13 hours on 2 Google Cloud TPU cores.

D Experimental Details

The scenes were constructed with the objects shown in Table 1 within a toy kitchen setup.

During evaluation, we roll out the policy given the instruction for 60 steps. Task success determined by a human according to the following criteria:

- Tasks that involve putting an object into or on top of a container (e.g. pot, pan, towel) are judged successes if any part of the object lies within or on top of the container.

- Tasks that involve moving an object toward a certain direction are judged successes if the object is moved sufficiently in the correct direction to be visually noticeable.

- Tasks that involve moving an object to a location relative to another object are judged successes if the object ends in the correct quadrant and are aligned with the reference object as instructed. For example, in "place the knife in front of the microwave", the knife should be placed in the top-left quadrant, and be overlapping with the microwave in the horizontal axis.

- If the robot attempts to grasp any object other than the one instructed, and this results in a movement of the object, then the episode is judged a failure.

Table 1: Evaluation Scenes

Scene	Objects
A	knife, pepper, towel, & pot
B	mushroom, towel, spoon, & pot
C	towel

E Experimental Results

We show per-task success rates for our approaches, the baselines, and the ablations in this section. The tasks in scenes A and B were evaluated for 10 trials each, while those in C were evaluated for 5 trials.

Table 2: Comparison of Approaches

		Success Rate				
Scene	**Task**	**GRIF**	**LCBC**	**LLfP**	**R3M**	**BC-Z**
A	put the yellow bell pepper on the cloth	**0.6**	0.0	0.0	0.0	**0.6**
	move the pan to the front	**1.0**	0.0	0.6	0.0	0.0
	put the pan on the towel	0.8	0.0	0.3	0.0	**0.9**
	move the bell pepper to the left of the table	0.7	0.0	0.0	0.0	**0.8**
	put the bell pepper in the pan	**0.8**	0.0	0.1	0.0	0.3
	put the knife on the purple cloth	**0.7**	0.0	0.2	0.0	0.0
	place the knife in front of the microwave	**0.7**	0.0	0.0	0.0	0.1
	move the pan in front of the cloth	**0.6**	0.0	0.3	0.0	0.0
B	put the mushroom in the metal pot	**0.9**	0.0	0.5	0.0	0.4
	put the spoon on the towel	**0.9**	0.0	0.3	0.0	0.4
	place the metal pot on top of the blue towel	**0.8**	0.0	0.0	0.0	0.2
C	move the towel to the left	**1.0**	0.0	**1.0**	0.0	**1.0**
	move the towel to the front	**1.0**	0.0	**1.0**	0.0	**1.0**
	move the towel next to the cans	**0.6**	0.0	0.0	0.0	0.2
	move the towel next to the microwave	**1.0**	0.0	0.2	0.0	0.8

Table 3: Comparison of Ablations

		Success Rate				
Scene	**Task**	**GRIF**	**Joint**	**Muse**	**Implicit**	**Static**
⋮	put the yellow bell pepper on the cloth	0.6	**0.8**	0.3	0.5	0.0
⋮	move the pan to the front	**1.0**	**1.0**	0.6	0.8	0.0
⋮	put the pan on the towel	0.8	**1.0**	0.6	0.6	0.0
⋮	move the bell pepper to the left of the table	**0.7**	0.4	0.4	0.6	0.2
A	put the bell pepper in the pan	**0.8**	0.6	0.7	0.6	0.1
⋮	put the knife on the purple cloth	**0.7**	0.4	0.2	0.2	0.0
⋮	place the knife in front of the microwave	**0.7**	0.6	0.1	0.0	0.0
⋮	move the pan in front of the cloth	0.6	**0.9**	0.4	0.0	0.3