AVID: Learning Multi-Stage Tasks via Pixel-Level Translation of Human Videos

Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, Sergey Levine

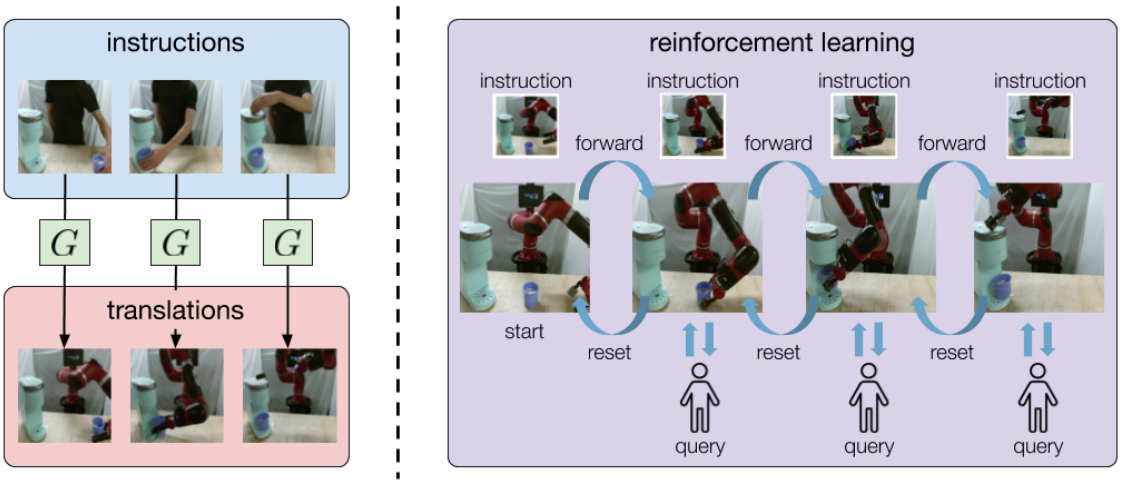

Fig. 1: Schematic of the overall method. Left: Human instructions for each stage (top) are translated at the pixel level into robot instructions (bottom) via the CycleGAN. Note the artifacts in the generated translations, e.g., the displaced robot gripper in the bottom right image. Right: The robot attempts the task stage-wise, automatically resetting and retrying until the instruction classifier signals success, which prompts the human to confirm via a key press.

***Abstract*— Robotic reinforcement learning (RL) holds the promise of enabling robots to learn complex behaviors through experience. However, realizing this promise requires not only effective and scalable RL algorithms, but also mechanisms to reduce human burden in terms of defining the task and resetting the environment. In this paper, we study how these challenges can be alleviated with an automated robotic learning framework, in which multi-stage tasks are defined simply by providing videos of a human demonstrator and then learned autonomously by the robot from raw image observations. A central challenge in imitating human videos is the difference in morphology between the human and robot, which typically requires manual correspondence. We instead take an automated approach and perform pixel-level image translation via CycleGAN to convert the human demonstration into a video of a robot, which can then be used to construct a reward function for a model-based RL algorithm. The robot then learns the task one stage at a time, automatically learning how to reset each stage to retry it multiple times without human-provided resets. This makes the learning process largely automatic, from intuitive task specification via a video to automated training with minimal human intervention. We demonstrate that our approach is capable of learning complex tasks, such as operating a coffee machine, directly from raw image observations, requiring only 20 minutes to provide human demonstrations and about 180 minutes of robot interaction with the environment. A supplementary video depicting the experimental setup, learning process, and our method's final performance is available from `https://sites.google.com/view/icra20avid`**

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, Berkeley, CA 94720

I. Introduction

Humans and animals can learn complex tasks not just from their own experience, but also by watching others. In robotics, this has been studied in the context of imitation learning [1]. However, robotic imitation learning typically utilizes demonstrations provided via the robot itself, either using teleoperation or kinesthetic teaching [2]–[5]. In contrast, humans can learn from *watching* other people, imagining how they would perform the task themselves, and then practicing that task on their own. The question that we study is: can we endow robots with the same ability?

We devise a robotic learning method that adopts a similar strategy of imagination and practice. Starting from a human demonstration, our method *translates* this demonstration, at the pixel level, into images of the robot performing the task, by means of unpaired image-to-image translation via CycleGAN [6] (see Figure 1). In order to handle multi-stage tasks, such as operating a coffee machine, we break up the human demonstration into a discrete set of *instruction images*, denoting the stages of the task. These instruction images are translated into synthesized robot images, which are then used to provide a reward for model-based reinforcement learning (RL), enabling the robot to practice the skill to learn its physical execution. This phase is largely automated as the robot learns to reset each stage on its own to practice it multiple times; thus, human supervision is only needed in the form of key presses and a few manual resets. We demonstrate that this approach is capable of

solving complex, long-horizon tasks with minimal human involvement, removing most of the human burden associated with instrumenting the task setup, manually resetting the environment, and supervising the learning process.

We name our method automated visual instruction-following with demonstrations (AVID), and this method is the main contribution of our work. We test AVID on two tasks on a Sawyer robot arm: operating a coffee machine and retrieving a cup from a drawer. AVID outperforms ablations and prior methods in terms of data efficiency and task success, learning coffee making with only 30 human demonstrations, amounting to 20 minutes of human demonstration time, and 180 minutes of robot interaction time.

II. Related Work

Prior methods for robotic imitation learning typically have assumed access to observations and actions that are given directly on the robot, rather than learning from human videos [7]–[10]. To handle human videos, some prior methods have used explicit pose and object detection [11]–[13], predictive modeling [14], [15], context translation [16], [17], and learning reward representations [18], [19]. In contrast to these works, we explicitly handle the change in embodiment through learned pixel-level translation, and we evaluate on long-horizon multi-stage tasks. We evaluate the single-view version of time-contrastive networks (TCN) [19] on our tasks in Section V as this prior method also handles embodiment changes and evaluates on visual robotic tasks. TCN typically requires multiple views of human demonstrations, but we do not assume access to this in our problem setting.

Providing reward through human videos alleviates the costs and challenges associated with real world task instrumentation, and to further reduce human burden, we enable the robot to reset each stage of the task. This is similar to prior work on learning reset controllers [20], [21]. However, in contrast to this prior work, our method learns from raw pixels and can achieve complex multi-stage tasks in the real world without manual reward design.

III. Preliminaries

To learn from human demonstrations, our method relies on several key steps: translating human videos, modeling robot images and actions, and extracting instruction images for model-based RL. In this section, we review the first two steps for which we borrow techniques from prior work.

A. Unsupervised Image-to-Image Translation

We approach human to robot translation as an unsupervised image-to-image translation problem, where the goal is to map images from a source domain X (human images, in our case) to a target domain Y (robot images) in the absence of paired training data [6], [22]. The approach we use is CycleGAN [6], which learns two mappings: $G : X \to Y$ translates source to target, and $F : Y \to X$ translates target to source. The translations are learned in order to fool discriminators D_Y and D_X which are trained to distinguish real images from translations in the target and source domains, respectively. This leads to a loss of the form

$$\mathcal{L}_{\text{GAN}}(G, D_Y) = \mathbb{E}[\log D_Y(y) + \log(1 - D_Y(G(x)))] \,,$$

And similarly for F and D_X, where x and y are source and target images drawn from the data distribution. An additional loss promotes cycle consistency between G and F, i.e.,

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}[\|x - F(G(x))\|_1 + \|y - G(F(y))\|_1] \,.$$

The overall training objective for CycleGAN is given by

$$\mathcal{L}_{\text{CG}}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y) + \mathcal{L}_{\text{GAN}}(F, D_X) + \lambda \mathcal{L}_{\text{cyc}}(G, F) \,,$$

Where λ is a hyperparameter. Though this approach does not exploit the temporal information that is implicit in our demonstration videos, prior work has shown that CycleGAN can successfully translate videos frame by frame, such as translating videos of horses into videos of zebras [6].

B. Structured Representation Learning

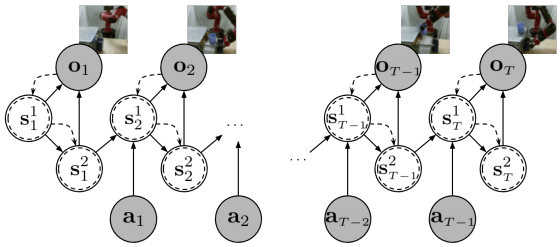

Fig. 2: The latent variable model we use to represent robot images and actions, with the generative model depicted in solid lines and the variational family and encoder network in dashed lines.

Prior work has shown that, for control learning in image-based domains, representation learning is an effective tool for improving data efficiency [23]–[26]. The approach we take is to utilize a probabilistic temporally-structured latent variable model that we learn with amortized variational inference, similar to [26]. In particular, we assume the underlying state of the system $\mathbf{s}_t$ is unobserved but evolves as a function of the previous state and action, and we treat the robot images as observations $\mathbf{o}_t$ of this state. Following [26], we also decompose $\mathbf{s}_t$ into two parts, i.e., $\mathbf{s}_t = [\mathbf{s}_t^1 \ \mathbf{s}_t^2]^\top$. The generative model for $\mathbf{s}_t$ can be summarized as an initial state distribution and a learned dynamics model, i.e.,

$$p(\mathbf{s}_1) = \mathcal{N}\left(\mathbf{s}_1^1; 0, \mathbf{I}\right) p(\mathbf{s}_1^2|\mathbf{s}_1^1) \,,$$
$$p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) = p(\mathbf{s}_{t+1}^1|\mathbf{s}_t^2, \mathbf{a}_t) p(\mathbf{s}_{t+1}^2|\mathbf{s}_{t+1}^1) \,,$$

And to complete the generative model, we learn a decoder $p(\mathbf{o}_t|\mathbf{s}_t)$ which we represent as a convolutional neural network. In order to learn this model, we introduce a variational distribution $q(\mathbf{s}_{1:T}; \mathbf{o}_{1:T})$ which approximates the posterior $p(\mathbf{s}_{1:T}|\mathbf{o}_{1:T}, \mathbf{a}_{1:T})$, where the $1:T$ notation denotes entire trajectories. We use a mean field variational approximation

$$q(\mathbf{s}_{1:T}; \mathbf{o}_{1:T}) = \prod_t q(\mathbf{s}_t; \mathbf{o}_t) \,,$$

Where the encoder $q(\mathbf{s}_t; \mathbf{o}_t)$ is also a convolutional network. Similar to [26], the generative model parameters are shared with the encoder according to

$$q(\mathbf{s}_t; \mathbf{o}_t) = q(\mathbf{s}_t^1 | \mathbf{o}_t) p(\mathbf{s}_t^2 | \mathbf{s}_t^1) \,.$$

We jointly learn p and q via maximization of the variational lower bound (ELBO), given by

$$\begin{aligned} \text{ELBO}[p, q] = \mathbb{E}_q[p(\mathbf{o}_t|\mathbf{s}_t)] - D_{\text{KL}}(q_{\mathbf{s}_1}(\cdot; \mathbf{o}_1) \| p_{\mathbf{s}_1}) \\ - \mathbb{E}_q \left[\sum_{t=1}^{T-1} D_{\text{KL}}(q_{\mathbf{s}_{t+1}}(\cdot; \mathbf{o}_{t+1}) \| p_{\mathbf{s}_{t+1}}(\cdot | \mathbf{s}_t, \mathbf{a}_t)) \right] . \end{aligned}$$

The entire structured latent variable model is visualized in Figure 2. As we explain in the next section, this model provides us with an encoder and dynamics model that we use for encoding instruction images and for model-based planning in the learned latent space.

IV. Automated Visual Instruction-Following with Demonstrations

In our problem setting, we assume that the task we want the robot to learn is specified by a set of human demonstration videos, each of which is a trajectory of image observations depicting the human performing the task. In contrast to most prior work in robotic imitation learning, we do not assume access to robot demonstrations given through teleoperation or kinesthetic teaching. We also do not assume access to rewards provided through motion capture or other instrumented setups. Our goal is to reduce human cost in task specification by having the robot learn directly from human videos, and in this section we detail several design choices that further reduce human burden during learning.

A. Translation for Goal Concept Acquisition

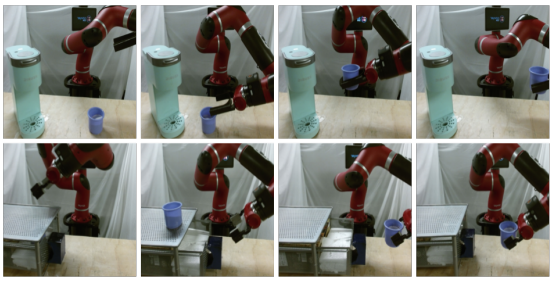

Fig. 3: Examples of robot data collected to train the CycleGAN for coffee making (top) and cup retrieval (bottom). Though the robot moves randomly, we cover different settings such as the robot holding the cup and the drawer being open by manually changing the environment.

CycleGAN requires training data from both source and the target domains, and we found that using diverse data was important for capturing a wide range of possible scenes and generating realistic translations. To that end, the training images we use from the human domain consist of both human demonstrations as well as a small amount of "random" data, in which the human moves around in the scene but does not specifically attempt the task. The training images from the robot domain consist entirely of random data, as we again do not assume access to robot demonstrations. However, we collect random robot data in a few different settings (see Figure 3), and in between settings we manually change the environment to provide more diverse data.

Once we have trained the CycleGAN model, the mapping G provides a mechanism for automatically translating human demonstration videos to robot demonstration videos. However, the translated videos are generally imperfect and exhibit artifacts, as shown in Figure 1. As we demonstrate in Section V, learning directly from the translated videos is susceptible to inaccuracies due to these artifacts, and the final learned performance is poor. We take a different approach in which the user manually selects a set of time steps $\{t_1, \ldots, t_S\}$ that correspond to the completion of stages, and we use the images from each translated video at time step t_i to specify the i-th *instruction image*. This works well because the human demonstrations are roughly time-aligned, but we could generalize this by manually selecting corresponding images from each video, which is feasible for the modest number of demonstrations we use.

To specify a reward function for RL, we train success classifiers for each stage $\{\mathcal{C}_1, \ldots, \mathcal{C}_S\}$, where $\mathcal{C}_i$ is provided the instruction images at time step t_i as positive examples and all other robot images as negative examples. The log probabilities from the classifiers can then be used as a reward signal, similar to [27], [28]. As mentioned earlier, providing rewards in this way requires only the translated demonstrations and does not rely on any manually designed reward shaping or instrumentation. In order to achieve greater data efficiency, we first encode the images using the encoder $q(\mathbf{s}_t; \mathbf{o}_t)$ and use the learned latent state as input to the classifier. As we discuss next, this setup lends itself naturally to an automated stage-wise model-based planning procedure.

B. Model-Based RL with Instruction Images

Though we could use any RL algorithm with our classifier-based reward, model-based RL has typically achieved greater data efficiency than model-free methods [29], [30], and the data efficiency afforded by the combination of model-based RL and representation learning further reduces the human supervision required during learning. Thus, the RL procedure that we use is the model-predictive control (MPC) variant of [24], which samples actions and generates rollouts in the latent space using the learned model, uses the cross-entropy method (CEM) [31] to optimize the sampled actions, and executes the first action corresponding to the rollout with the highest reward, which is given by a classifier in our case. This planning is repeated at every time step to produce a trajectory of images and actions, and we encapsulate this procedure in the `MPC-CEM` subroutine in Algorithm 1.

When the robot is in stage s, the planner uses the log probability of $\mathcal{C}_s$ as the reward function and aims to surpass a classifier threshold $\mathcal{T} \in [0, 1]$, which is a hyperparameter. If this threshold is not met, the planner automatically switches to $\mathcal{C}_{s-1}$, i.e., it attempts to reset to the beginning of the stage. This forward-reset behavior allows the robot to robustify its performance with very few human-provided resets, as the

human only intervenes to fix problems, such as the cup falling over, rather than manually resetting every episode. The robot runs this forward-reset loop for a maximum of $\mathcal{A}$ iterations, where $\mathcal{A}$ is also a hyperparameter.

Should the threshold be met during planning, the robot will query the human user, at which point the user signals either success or failure through a key press. On failure, the robot switches to the reset behavior, and the loop continues. On success, the robot moves on to the next stage and repeats the same process, with $\mathcal{C}_{s+1}$ specifying the goal and $\mathcal{C}_s$ specifying the reset. This stage-wise learning avoids the compounding errors of trying to learn the entire task all at once. The full procedure is summarized in Algorithm 1 and concludes when the user signals success for stage S.

Algorithm 1 AVID RL Forward Pass

Require: Pre-trained model $\mathcal{M}$ and $\{\mathcal{C}_i\}_{i=0}^{S}$, initial robot data $\mathcal{D}$
Require: Max attempts per stage $\mathcal{A}$, classifier threshold $\mathcal{T}$
1: `stage` $\leftarrow 1$
2: **while** `stage` $\leq S$ **do**
3: `attempts` $\leftarrow 0$
4: `stage-completed` $\leftarrow$ *false*
5: **while** `attempts` $< \mathcal{A}$ **and not** `stage-completed` **do**
6: $\tau \leftarrow$ `MPC-CEM`$(\mathcal{M}, \mathcal{C}_{\texttt{stage}})$; $\mathcal{D} \leftarrow \mathcal{D} \cup \tau$
7: Increment `attempts`
8: **if** $\mathcal{C}_{\texttt{stage}}$(last observation of τ) $> \mathcal{T}$ **then**
9: **if** human signals success **then**
10: Add last observation of τ to goal images for `stage`
11: `stage-completed` $\leftarrow$ *true*
12: **else**
13: $\mathcal{D} \leftarrow \mathcal{D} \cup$ `EXPLORE`()
14: Train $\mathcal{M}$ and $\{\mathcal{C}_i\}_{i=0}^{S}$ on $\mathcal{D}$ and goal images
15: **end if**
16: `attempts` $\leftarrow 0$
17: **else**
18: $\tau \leftarrow$ `MPC-CEM`$(\mathcal{M}, \mathcal{C}_{\texttt{stage}-1})$; $\mathcal{D} \leftarrow \mathcal{D} \cup \tau$
19: **end if**
20: **end while**
21: **if** `stage-completed` **then**
22: Increment `stage`
23: **else**
24: Train $\mathcal{M}$ and $\{\mathcal{C}_i\}_{i=0}^{S}$ on $\mathcal{D}$ and goal images
25: `attempts` $\leftarrow 0$
26: **end if**
27: **end while**

We include several important improvements on top of this procedure in order to achieve good performance. In line 13 of Algorithm 1, if the human signals failure, the `EXPLORE` subroutine is invoked, in which the robot moves randomly from the state it ended in and queries the human several times. This provides additional data within the critical area of what $\mathcal{C}_s$ believes is a success. In addition, in line 14, the current image is labeled as a negative for $\mathcal{C}_s$ and added to the dataset along with the images from random exploration. Similar to [27], [28], further training $\mathcal{C}_s$ on this data improves its accuracy by combating the false positive problem and offsetting the artifacts in the translated images that $\mathcal{C}_s$ was initially trained with. Finally, to further automate the entire process and avoid manual human resets, after the robot completes the last stage, we run Algorithm 1 in reverse in order to get back to the initial state. We repeat this entire process until the robot can reliably complete the entire task.

V. Experiments

We aim to answer the following through our experiments:

(1) Is AVID capable of solving temporally extended visual tasks directly from human demonstrations?
(2) What benefits, if any, do we gain from using instruction images and latent space planning?
(3) What costs, if any, do we incur from not directly having access to robot demonstrations?

We evaluate question (1) on two complex and temporally extended tasks (see Figure 4 and Figure 5). For (2), we compare AVID to learning from full human demonstrations with TCN [19] and an ablation of AVID based on behavioral cloning from observations (BCO) [32]. To understand the effects of latent space planning, we compare to an ablation of AVID based on deep visual foresight (DVF) [33]. Finally, to answer (3), we evaluate BCO and behavioral cloning on the same tasks but with access to robot demonstrations, and we also analyze the human supervision burden of AVID. A supplementary video depicting the learning process of AVID, as well as final performances for AVID and all of the comparisons, is available from the project website.[1]

A. Comparisons

We compare to the following methods:

Time-contrastive networks [19]. We evaluate the single-view version of TCN, which learns an embedding via a temporal consistency loss in order to generalize from human demonstrations to robot execution. TCN originally used PILQR for their RL algorithm [34], though any RL algorithm can be used. We use the same RL procedure as AVID, where the only difference is that instead of learning classifiers for the reward function, we use the Euclidean distance between TCN embeddings of the human demonstration and model-generated trajectories, decoded using $p(\mathbf{o}_t|\mathbf{s}_t)$.

Imitation ablation. An alternative approach to our stage-wise RL procedure is to learn directly from the full translated videos, and a natural way to do this is to use an imitation learning method that does not require expert actions [16], [32], [35]–[38]. We consider an ablation of AVID that learns via imitation of full translated demonstrations, using behavioral cloning from observations (BCO) [32], instead of RL from stage-wise instructions. We also test BCO using real robot demonstrations obtained via teleoperation.

Pixel-space ablation. DVF is a visual model-based RL method that plans directly in pixel space rather than learning a latent space. We consider an ablation of AVID in which we replace our latent space planning procedure with DVF to understand if AVID achieves better performance or data efficiency as a result of this design choice.

Behavioral cloning. Behavioral cloning is an imitation learning approach that requires real demonstrations consisting of both observations and actions. Although this "oracle" approach has access to real demonstrations, it does not utilize the interactive training procedure with learned resets and

[1] `https://sites.google.com/view/icra20avid`

human feedback, and so this comparison helps us determine what benefits AVID derives from the training procedure.

B. Experimental Setup

Fig. 4: Sample sequence of instructions for coffee making segmented from a human demonstration (top), translated into the robot's domain (bottom). The stages from left to right are: initial state, pick up the cup, place the cup in the coffee machine, and press the button on top of the machine.

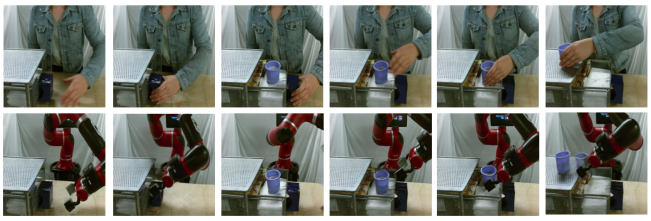

Fig. 5: Sample sequence of instructions for cup retrieval segmented from a human demonstration (top), translated into the robot's domain (bottom). The stages from left to right are: initial state, grasp the handle, open the drawer, lift the arm, pick up the cup, and place the cup on top.

We evaluate our method on two temporally-extended tasks from vision: operating a personal coffee machine and retrieving a cup from a closed drawer. These tasks illustrate that our method can learn to sequentially compose several skills by following a set of instructions extracted from a human demonstration. For all our experiments, we use end-effector velocity control on a 7 DoF Sawyer robotic manipulator, and our observations consist only of 64-by-64-by-3 RGB images.

Coffee making. The coffee making task has three stages as depicted in Figure 4. Starting with the cup on the table, the instructions are to pick up the cup, place the cup in the machine, and press the button on the top of the machine. For this task, we used 30 human demonstrations, which corresponds to 900 images and 20 minutes of human demonstration time, along with 500 images of random human data. The CycleGAN robot data consisted of 8100 images of random robot movements with the human changing the setting 6 times by moving the cup and placing the cup in the robot's gripper. We set the classifier threshold $\mathcal{T}$ to 0.8 and the maximum number of attempts per stage $\mathcal{A}$ to 3.

The same 8100 robot images for CycleGAN training were also used to learn the models for each method, and the BCO and DVF models received 7200 additional training images of random robot data.[2] During RL, AVID and the BCO ablation received 3015 and 1350 training images, respectively, for online model training. TCN was given 1395 human images and 14400 robot images to learn the embedding and 3060 additional images during online training. In the setting with real robot demonstrations, BCO and behavioral cloning used 1800 and 900 total images, respectively.

Cup retrieval. Cup retrieval has five stages, as shown in Figure 5: grasping the drawer handle, opening the drawer, moving the arm up and out of the way, picking up the cup, and placing the cup on top of the drawer. We found that the intermediate stage of moving the arm was important for good performance, as the model-based planning would otherwise bump into the drawer door. Providing this instruction was trivial with our setup, as we simply specified an additional instruction time step. For this task, we used 20 human demonstrations, amounting to 600 images and 20 minutes of human time, and 300 images of random human data. We collected 7200 images of random robot data, where the human changed the setting 11 times by moving the cup, placing the cup in the robot's gripper, and opening the drawer. $\mathcal{T}$ and $\mathcal{A}$ were again set to 0.8 and 3.

The BCO and DVF models were given 7200 additional training images of random robot data, and during RL, AVID and the BCO ablation used 1740 and 1400 additional training images, respectively. TCN was given 900 human images and 14400 robot images to learn the embedding and 1800 additional images during online training. In the setting with real robot demonstrations, BCO used 1200 total robot images and behavioral cloning used 900 total robot images.

C. Experimental Results

We report success rates for both coffee making and cup retrieval in Table I. Success rates are evaluated using 10 trials per method per task, with no human feedback or manual intervention. Success metrics are defined consistently across all methods and are strict, e.g., placing the cup on top of the drawer but toppling the cup over is a failure. The supplementary video shows representative evaluation trials, labeled with success and failure, for each method. For the final stage of coffee making, as it is difficult to visually discern whether the robot has noticeably depressed the coffee machine button, success is instead defined as being within $5\,\mathrm{cm}$ above the button, such that a pre-programmed downward motion always successfully presses the button.

For both tasks, AVID achieves the best performance among the methods that use human demonstrations, consistently learning the earlier stages perfectly and suffering less from accumulating errors in the later stages. As seen in the supplementary video, the failures of AVID for both tasks correspond to generally good behavior with small, but significant, errors, such as knocking over or narrowly missing the cup. AVID makes constant use of automated resetting and retrying during both RL and the final evaluation, allowing for fewer manual resets and more robust behavior compared to the methods that cannot exploit the stage-wise decomposition of the tasks. Though the pixel-space ablation also incorporates resetting, this method is less successful than AVID,

[2]We found this additional data to be important as these methods use pixel-space models, which are generally less data efficient.

Supervision	Method	Coffee making			Cup retrieval				
		Stage 1	Stage 2	Stage 3	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
Human Demos	AVID (ours)	**100%**	**80%**	**80%**	**100%**	**100%**	**100%**	**80%**	**70%**
	Imitation ablation	70%	10%	0%	0%	0%	0%	0%	0%
	Pixel-space ablation	60%	20%	0%	50%	50%	30%	10%	0%
	TCN [19]	10%	10%	0%	60%	20%	0%	0%	0%
Robot Demos	BCO [32]	80%	30%	0%	30%	10%	0%	0%	0%
	Behavioral Cloning	90%	90%	90%	100%	100%	60%	60%	40%

Table I: We report success rates up to and including each stage for both tasks, over 10 trials. The top rows are methods that learn from human demonstrations, and we bold the best performance in this category. The bottom two rows are methods trained with direct access to real robot demonstrations. AVID outperforms all other methods from human demonstrations, succeeding 8 times out of 10 on coffee making and 7 times out of 10 on cup retrieval, and even outperforms behavioral cloning from real robot demonstrations on the later stages of cup retrieval.

and this is due to the fact that the pixel-space model is not trained online or used for replanning as this is prohibitively expensive. This indicates the benefits that we derive from our latent variable model and planning procedure.

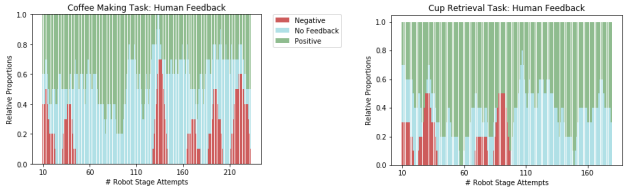

Fig. 6: Visualizing human feedback during the learning process for coffee making (left) and cup retrieval (right). The x-axis is the total number of robot stage attempts, and the y-axis indicates the proportions of feedback types, smoothed over the ten most recent attempts. "No feedback" means that the classifier did not signal success and the robot automatically switched to resetting. Coffee making and cup retrieval use a total of 131 and 126 human key presses, respectively, for the learning process.

The imitation ablation can learn to pick up the cup 70% of the time for the coffee making task, but it fails on the later stages and also fails entirely for cup retrieval. This indicates the difficulty of learning from full translated demonstrations due to the artifacts in the translations and the absence of stage-wise training, and the supplementary video shows that, although the behavior of this method visually looks almost correct in many instances, it generally still fails in subtle ways. Finally, despite using publicly available code from the authors, we were unable to achieve good performance with TCN on coffee making, though TCN was moderately successful on the early stages of cup retrieval. We note that we used the single-view version of this method, and the authors also found that this version did not perform well for their tasks [19]. The multi-view version of TCN is not applicable in our case as we do not assume multiple views of human demonstrations at training time.

In order to understand how using translated human demonstrations compares to using real robot demonstrations, we evaluate BCO and behavioral cloning from robot demonstrations obtained through teleoperation. As expected, BCO performs better in this case than from translated demonstrations, however the performance is still significantly worse than AVID from translated demonstrations. This indicates that learning from full demonstrations, even directly obtained from the robot, suffers from accumulating errors for long-horizon tasks. We note that, to our knowledge, BCO has never been tested on tasks exhibiting multiple stages or operating directly from image observations [32].

Finally, behavioral cloning outperforms our method for coffee making but surprisingly performs worse on cup retrieval. We believe that this is because the cup retrieval task has more stages and is harder to learn without explicit stage-wise training. Thus, compared to AVID, behavioral cloning has more stringent assumptions and may perform worse for more complex tasks, but the drawback of AVID is that it uses an RL training phase that requires human feedback. To understand the human burden associated with this training procedure, we plot the human feedback we provide during training in Figure 6. The training phase took about an hour for each task, and in both cases AVID used less than 150 key presses to learn the task, which was easy for a human supervisor to provide. We view this as a practical way to enable robots to learn complex tasks.

VI. Discussion and Future Work

We presented AVID, a method for learning visual robotic multi-stage tasks directly from videos of human demonstrations. AVID uses image-to-image translation, via CycleGAN, to generate robot instruction images that enable a stage-wise model-based RL algorithm to learn from raw image observations. The RL algorithm is based on latent space model-based planning and incorporates learned resets and human feedback to learn temporally extended tasks robustly and efficiently. We demonstrated that AVID is capable of learning multi-stage coffee making and cup retrieval tasks, attains substantially better results than prior methods and ablations using human demonstrations, and even outperforms behavioral cloning using real robot demonstrations (rather than videos of human demonstrations) on one of the tasks.

While AVID aims to minimize the per-task setup burden on the user, some human setup is still necessary for each task. Learning coffee making and cup retrieval required training separate CycleGAN models, which was computationally expensive and involved collecting separate datasets involving random human movements and setting changes for robot data collection. This cost can be amortized across multiple tasks in future work. For example, the CycleGAN trained

for cup retrieval can also be used to translate demonstrations for placing the cup back in the drawer, meaning that this new task can be learned from human demonstrations with no additional upfront cost of data collection and CycleGAN training. We can generalize this by training a CycleGAN on an initial large dataset, e.g., many different human and robot behaviors in a kitchen that has a coffee machine, multiple drawers, and numerous other objects. This should enable any new task in the kitchen to be learned with just a few human demonstrations, and this is a promising direction toward truly allowing robots to learn by watching humans.

REFERENCES

[1] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, 2009.

[2] S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, and A. Kheddar, "Learning collaborative manipulation tasks by demonstration using a haptic interface," in *ICAR*, 2009.

[3] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, "Online movement adaptation based on previous sensor experiences," in *IROS*, 2011.

[4] B. Akgun, M. Cakmak, J. Yoo, and A. Thomaz, "Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective," in *HRI*, 2012.

[5] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *ICRA*, 2018.

[6] J. Zhu, T. Park, P. Isola, and A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017.

[7] A. Billard and M. Mataric, "Learning human arm movements by imitation: Evaluation of a biologically inspired connectionist architecture," *Robotics and Autonomous Systems*, 2001.

[8] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transaction of the Royal Society of London, Series B*, 2003.

[9] J. Kober and J. Peters, "Learning motor primitives for robotics," in *ICRA*, 2009.

[10] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, "Learning objective functions for manipulation," in *ICRA*, 2013.

[11] K. Lee, Y. Su, T. Kim, and Y. Demiris, "A syntactic approach to robot imitation learning using probabilistic activity grammars," *Robotics and Autonomous Systems*, 2013.

[12] Y. Yang, Y. Li, C. Fermuller, and Y. Aloimonos, "Robot learning manipulation action plans by "watching" unconstrained videos from the world wide web," in *AAAI*, 2015.

[13] K. Ramirez-Amaro, M. Beetz, and G. Cheng, "Transferring skills to humanoid robots by extracting semantic representations from observations of human activities," *Artificial Intelligence*, 2017.

[14] N. Rhinehart and K. Kitani, "First-person activity forecasting with online inverse reinforcement learning," in *ICCV*, 2017.

[15] A. Tow, N. Sünderhauf, S. Shirazi, M. Milford, and J. Leitner, "What would you do? Acting by learning to predict," in *IROS*, 2017.

[16] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, "Imitation from observation: Learning to imitate behaviors from raw video via context translation," in *ICRA*, 2018.

[17] P. Sharma, D. Pathak, and A. Gupta, "Third-person visual imitation learning via decoupled hierarchical control," in *NeurIPS*, 2019.

[18] P. Sermanet, K. Xu, and S. Levine, "Unsupervised perceptual rewards for imitation learning," in *RSS*, 2017.

[19] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine, "Time-contrastive networks: Self-supervised learning from video," in *ICRA*, 2018.

[20] W. Han, S. Levine, and P. Abbeel, "Learning compound multi-step controllers under unknown dynamics," in *IROS*, 2015.

[21] B. Eysenbach, S. Gu, J. Ibarz, and S. Levine, "Leave no trace: Learning to reset for safe and autonomous reinforcement learning," in *ICLR*, 2018.

[22] M. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *NIPS*, 2017.

[23] T. Lesort, N. Díaz-Rodríguez, J. Goudou, and D. Filliat, "State representation learning for control: An overview," *Neural Networks*, 2018.

[24] M. Zhang, S. Vikram, L. Smith, P. Abbeel, M. Johnson, and S. Levine, "SOLAR: Deep structured representations for model-based reinforcement learning," in *ICML*, 2019.

[25] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *ICML*, 2018.

[26] A. Lee, A. Nagabandi, P. Abbeel, and S. Levine, "Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model," *arXiv preprint arXiv:1907.00953*, 2019.

[27] J. Fu, A. Singh, D. Ghosh, L. Yang, and S. Levine, "Variational inverse control with events: A general framework for data-driven reward definition," in *NIPS*, 2018.

[28] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," in *RSS*, 2019.

[29] M. Deisenroth, D. Fox, and C. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *PAMI*, 2014.

[30] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *NIPS*, 2018.

[31] R. Rubinstein and D. Kroese, *The Cross Entropy Method: A Unified Approach To Combinatorial Optimization, Monte-Carlo Simulation (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2004.

[32] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," in *IJCAI*, 2018.

[33] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," *arXiv preprint arXiv:1812.00568*, 2018.

[34] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, and S. Levine, "Combining model-based and model-free updates for trajectory-centric reinforcement learning," in *ICML*, 2017.

[35] W. Sun, A. Vemula, B. Boots, and J. Bagnell, "Provably efficient imitation learning from observation alone," in *ICML*, 2019.

[36] A. Edwards, H. Sahni, Y. Schroecker, and C. Isbell, "Imitating latent policies from observation," in *ICML*, 2019.

[37] T. Yu, C. Finn, A. Xie, S. Dasari, P. Abbeel, and S. Levine, "One-shot imitation from observing humans via domain-adaptive meta-learning," in *RSS*, 2018.

[38] D. Pathak, P. Mahmoudieh, G. Luo, P. Agrawal, D. Chen, Y. Shentu, E. Shelhamer, J. Malik, A. Efros, and T. Darrell, "Zero-shot visual imitation," in *ICLR*, 2018.