MILES: Making Imitation Learning Easy with Self-Supervision

Georgios Papagiannis and Edward Johns
The Robot Learning Lab
Imperial College London, UK
g.papagiannis21@imperial.ac.uk

Abstract: Data collection in imitation learning often requires significant, laborious human supervision, such as numerous demonstrations and/or frequent environment resets for methods that incorporate reinforcement learning. In this work, we propose an alternative approach, MILES: a fully autonomous, self-supervised data collection paradigm and show that this enables efficient policy learning from just a single demonstration and a single environment reset. Our method, MILES, autonomously learns a policy for returning to and then following the single demonstration, whilst being self-guided during data collection, eliminating the need for additional human interventions. We evaluate MILES across several real-world tasks, including tasks that require precise contact-rich manipulation, and find that, under the constraints of a single demonstration and no repeated environment resetting, MILES significantly outperforms state-of-the-art alternatives like reinforcement learning and inverse reinforcement learning. Videos of our experiments, code, and supplementary material can be found on our website: www.robot-learning.uk/miles.

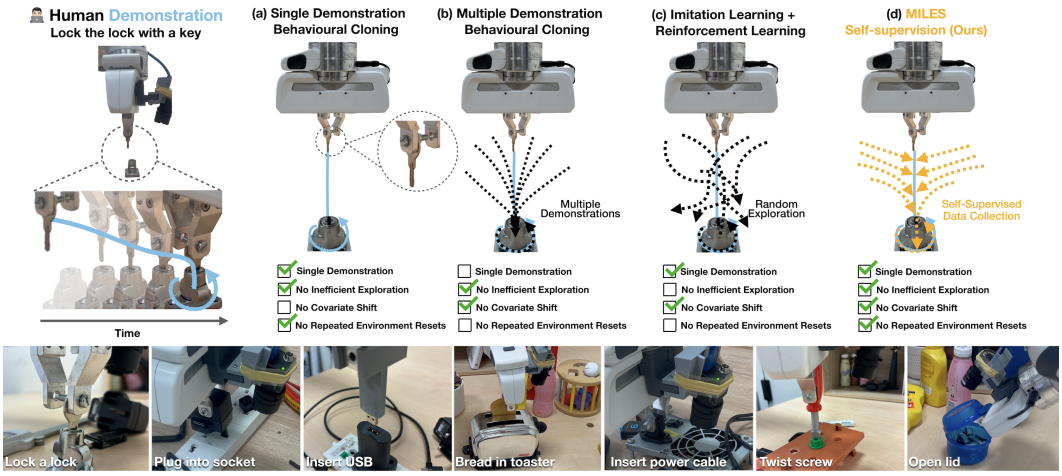

Figure 1: (a) Behavioural cloning from a single demonstration fails to generalize to states outside the demonstration, due to covariate shift. (b) Providing multiple demonstrations addresses this, but requires significant human effort. (c) While incorporating reinforcement learning addresses the issue of covariate shift and the need for multiple demonstrations, it requires frequent environment resetting and is highly inefficient due to random exploration. (d) In MILES, we propose a new self-supervised paradigm that overcomes these issues and can learn a range of complex tasks from a single demonstration and no additional human effort, by collecting augmentation trajectories that guide the robot back to the demonstration.

1 Introduction

Imitation learning is frequently described as a convenient way to teach robots new skills. But is this true in practice? Behavioral cloning (BC) methods leverage supervised learning to train robust policies, but doing so typically requires tens or hundreds of demonstrations per task [1, 2] to collect a sufficiently diverse training dataset. Inverse reinforcement learning (IRL) methods offer a solution to this as policies can be learned autonomously through random exploration, and reward functions can be inferred from a few demonstrations [3, 4]. However, unlike supervised learning methods, policy learning with IRL can be unstable [5], and random exploration makes data collection inefficient. Besides, IRL typically requires repeated environment resetting, and so in practice, it is often

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

equally or even more laborious than simply providing numerous demonstrations. Instead, learning from a single demonstration appears to be the most convenient form of imitation learning due to its effortlessness, but policies learned this way suffer from covariate shift [6]. As such, imitation learning today is not as easy as we would like it to be: significant human supervision is still required for data collection either via demonstrations, environment resetting, or both.

Motivated by this, we propose MILES, a framework that makes imitation learning easy by enabling humans to teach robots tasks effortlessly with just a *single* demonstration while requiring no prior task knowledge, and only a single environment reset. MILES learns robot skills by collecting *trajectories* in a self-supervised manner that demonstrate to the robot how to return to, and then follow the single human demonstration as shown in Figure 1. MILES leverages these trajectories to train a policy using BC, consequently inheriting the power and ease of supervised learning. However, unlike common BC methods, the self-supervised data collection replaces the need to collect multiple demonstrations and compared to BC from a single demonstration, MILES does not suffer from covariate shift as its data densely covers the space around the demonstration. Similarly to inverse RL methods, MILES collects data autonomously, but instead of being guided by random exploration, MILES' data collection is highly efficient as it benefits from self-labeled data that directly guides the robot back to the demonstration. Additionally, exploration can cause disturbances to the environment and thus requires repeated environment resetting. However, as we later explain, MILES' self-supervised data collection procedure strategically shapes policy learning to be independent of such laborious environment resets. Consequently, MILES aims to incorporate the benefits of training BC policies with supervised learning on large demonstration datasets with the benefits of autonomous data collection, similarly to RL methods, in the setting where only a single demonstration is available.

Through multiple real-world experiments, we find that when only a single human demonstration is available, without additional human effort such as repeated environment resetting, self-supervised data collection outperforms recent alternatives in reinforcement learning, inverse reinforcement learning, and replay-based imitation learning. MILES can learn a suite of versatile skills, shown in Figure 1, ranging from those involving interactions with movable, articulated objects such as opening the lid of a box, to those involving precise, contact-rich interactions with objects rigidly mounted to a surface, such as inserting and twisting a key in a lock, with each learned from a single demonstration, no further human input, and around 30 minutes of self-supervised data collection.

2 Related Work

As follows, we ground our work relative to methods that can learn manipulation skills from a single demonstration, unlike most approaches that require large demonstration datasets [1, 7, 8].

Imitation learning from prior knowledge. An effective way to compensate for the lack of large demonstration datasets is to leverage prior task knowledge such as access to ground truth object poses [9, 10] or by meta-learning policies by first pretraining on large demonstration datasets [11, 12]. However, precise knowledge of the objects' poses is hard to obtain in practice and meta-learning methods are often limited to tasks similar to the ones seen in the demonstrations. Instead, MILES can learn a new task from just a single demonstration without any prior object or task knowledge.

Imitation learning via Reinforcement learning (RL). Inverse RL methods from a single demonstration learn to follow that demonstration by minimizing a similarity metric between the trajectories of the learned policy and the demonstration [3, 13, 4, 14]. Other RL methods that learn from demonstrations infer rewards through alternative means, like goal images [15]. Though effective, these methods are often inefficient as they rely on random exploration and repeated environment resets which require significant human effort. Instead, our self-supervised data collection makes MILES highly efficient and eliminates the need for repeated environment resetting.

Imitation learning via pose estimation and demonstration replay. Replay-based imitation learning methods first estimate and move the robot to a similar pose relative to the objects of interest as in the demonstration and then replay the demonstrated robot actions [16, 17, 18, 19, 20]. While these methods are the most efficient in terms of human time, small errors in pose estimation cause errors to compound during demonstration replay, leading to task failures [2]. And even under the assumption of perfect pose estimation, potential environment collisions may prevent the robot from reaching the desired pose or may perturb the objects such that replaying the demonstration fails to complete the task. Instead, MILES' self-supervised data collection procedure retains the human-time effi-

ciency of pose estimation methods, while learning to avoid unnecessary collisions, and minimizing or completely eliminating open-loop replay errors depending on the task.

Imitation learning by demonstration augmentation. Demonstration augmentation approaches like DAgger [6] and DART [21] mitigate covariate shift by relying on laborious interactive expert queries to expand the known state distribution of a policy. And methods that do not require an interactive expert still rely on multiple demonstrations or task-specific optimizations [22, 23, 24, 25, 26] which limit their practical application. Instead, MILES is a fully autonomous method that uses self-supervision to augment a single demonstration and can learn a wide range of diverse tasks.

3 MILES: Making Imitation Learning Easy with Self-Supervision

As follows we describe MILES, a framework that makes imitation learning easy by leveraging a single human demonstration as guidance to collect self-supervised data demonstrating to the robot how to return to, and then follow the demonstration. By training a policy with behavioral cloning on that data, MILES learns to perform a task from a range of initial states and object poses.

3.1 Preliminaries

Assumptions. Our setup assumes access to a wrist camera that is rigidly mounted to the robot's end-effector (EE) and (optionally) a sensor that measures external forces and torques. We follow prior work [18, 2, 16] and assume that each task is object-centric, such that only the task-relevant object is in camera view during data collection and the demonstration can be expressed relative to a single object, where combining several such tasks results in a multi-stage task. Additionally, as we are interested in dealing with all types of tasks, including those that require contact-rich manipulation, we control our robot using an impedance controller.

Single Demonstration. For each task, a human provides a *single* demonstration $\zeta := \{(w_n^\zeta, o_n^\zeta, a_n^\zeta)\}_{n=1}^N$ comprising a sequence of N waypoints w_n^ζ, observations o_n^ζ, and actions a_n^ζ, as shown in Figure 2 (1). A waypoint w_n^ζ corresponds to the EE's 6-DoF pose at timestep n captured via proprioception. An observation o_n^ζ consists of an RGB image captured from the wrist camera and a force-torque measurement. We refer to (w_n^ζ, o_n^ζ) as the state of the environment at timestep n. An action a_n^ζ contains the gripper's state and the 6-DoF pose tracked by the impedance controller at timestep n, expressed relative to the EE's pose at timestep $n-1$. After providing ζ the human resets the environment *only once*, such that if the actions in ζ are executed, the robot would successfully perform the task; a trivial process that requires a few seconds of human time.

3.2 Self-Supervised Data Collection

Augmentation Trajectories. Given a single demonstration, MILES collects a dataset of augmentation trajectories $\mathcal{D} := \{\tau_k\}$, where $1 \leq k \leq N$ and each $\tau_k := \{(w_m^{\tau_k}, o_m^{\tau_k}, a_m^{\tau_k})\}_{m=1}^M$ is a robot trajectory whose final, M_{th} state corresponds to a k_{th} state in the demonstration, i.e., $(w_M^{\tau_k}, o_M^{\tau_k}) = (w_k^\zeta, o_k^\zeta)$. That is, each augmentation trajectory guides the robot to some k_{th} state in the demonstration from any state $(w_m^{\tau_k}, o_m^{\tau_k}) \in \tau_k$. We can *fuse* each augmentation trajectory with the demonstration segment following each k_{th} state, $\{(w_n^\zeta, o_n^\zeta, a_n^\zeta)\}_{n=k}^N \subseteq \zeta$, to create a *new* demonstration that demonstrates to the robot how to **return to** and then **follow** the human demonstration as shown in Figure 3. By collecting augmentation trajectories that densely cover the state space near the demonstration we can create a dataset of new demonstrations which we can leverage to train a policy using standard BC methods. But how do we create such a dataset of augmentation trajectories automatically?

Data Collection. We achieve this by collecting data in the simplest possible way. An overview of our data collection procedure is shown in Figure 2. To generate a τ_k, from a demonstration waypoint w_k^ζ, we first move the robot to some random pose near the demonstration. The robot then attempts to return back to w_k^ζ in a straight line, while recording RGB and force-feedback observations $\{o_m^{\tau_k}\}_{m=1}^M$, and actions $\{a_m^{\tau_k}\}_{m=1}^M$ automatically generated by computing the EE's relative movement between consecutive timesteps, as shown in Figure 2 (3) (gripper actions are copied directly from the demonstrated action). In practice, the actual trajectory τ_k may not be a straight line, as collisions with external objects can yield a more complex, *curved path* due to the robot's compliance; an example of this happening for a locking a lock task

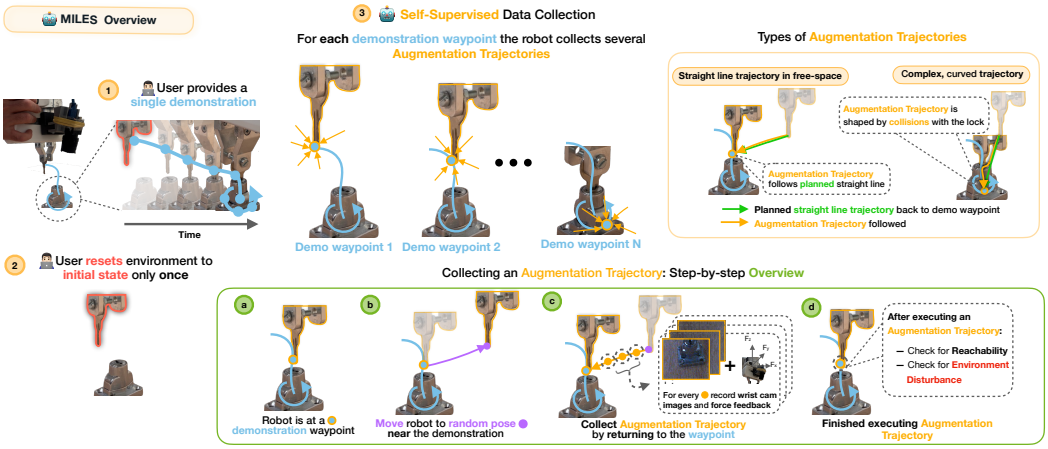

Figure 2: **MILES Overview:** (1) First, the user provides a single demonstration and (2) resets the environment only once. (3) Then, the robot (autonomously) collects self-supervised data. Several augmentation trajectories are collected for each demonstration waypoint until an environment disturbance is detected or sufficient data is collected for all waypoints. Each augmentation trajectory is either a straight line, if the motion occurs in free space, or a more complex, curved path as the augmentation trajectory can be reshaped by collisions with the environment (e.g., with the lock as shown above). (3) (a-b) To collect an augmentation trajectory, the robot first moves from a demonstration waypoint to a random pose. (c) Then, it attempts to return back to the waypoint while recording RGB images and force-torque feedback . (d) After completing the trajectory, we check whether the achieved state meets the conditions of *reachability* and *environment disturbance*.

is shown in Figure 2 (3). This is particularly useful for contact-rich tasks, as these trajectories contain information on overcoming potential collisions or large friction areas by regulating force when in contact with an object. After executing the potential augmentation trajectory we check that it is valid and that it can be fused with the demonstration by evaluating two key conditions, that of **reachability** and **environment disturbance** which we describe in section 3.3. If both conditions are met we store that augmentation trajectory, otherwise, it is discarded. We repeat this process several times for each demonstration state, starting from the first waypoint in the demonstration w_1^ζ and gradually progress through the demonstration, as shown in Figure 2 (3), until: (1) an environment disturbance is detected, in which case we stop the data collection and store the demonstration timestep where the disturbance occurred, denoted R, and the actions $\zeta_{remaining} := \{a_n^\zeta\}_{n=R}^N$ for the remaining demonstration states for which no data is collected; or (2) we have collected a prespecified number of Z augmentation trajectories for each of the N demonstration states, in which case $R = N$.

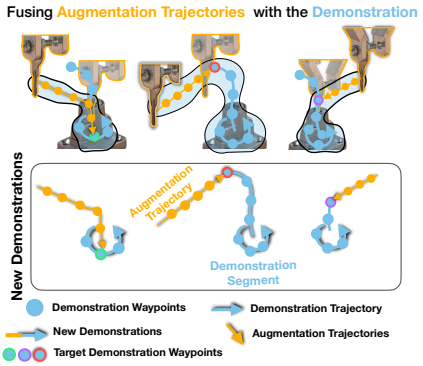

Figure 3: After finishing the data collection, each augmentation trajectory is fused with the demonstration segment following the demonstration waypoint it returns to, to create a dataset of new demonstration trajectories.

At the end of the data collection process, every augmentation trajectory is fused with ζ to form a new demonstration $\zeta_k = \{(o_m^{\tau_k}, a_m^{\tau_k})\}_{m=1}^M \cup \{(o_n^\zeta, a_n^\zeta)\}_{n=k}^R$ as shown in Figure 3. And as a result, we obtain a dataset $\mathcal{D}_{new} = \{\zeta, \zeta_1^1, \zeta_1^2, .., \zeta_1^Z, ..., \zeta_R^{Z-1}, \zeta_R^Z\}$. Every trajectory in $\mathcal{D}_{new}$ corresponds to a *new* demonstration, automatically created, that solves the task up to the R_{th} state in the demonstration. Further details concerning a practical implementation of our data collection procedure and pseudocode can be found in our supplementary material section A.1.

3.3 Validity Conditions for Augmentation Trajectories

As follows, we introduce two conditions that determine whether an augmentation trajectory can be fused with the human demonstration. Consider an augmentation trajectory τ_k aimed at returning the robot to the k_{th} demonstration state, (w_k^ζ, o_k^ζ):

(1) **Condition 1, Reachability:** After executing the augmentation trajectory, the EE's pose must equal the pose of the demonstration waypoint w_k^ζ. This equality can be verified trivially using proprioception. In many scenarios, the environment's dynamics (e.g. collisions) or inevitable sys-

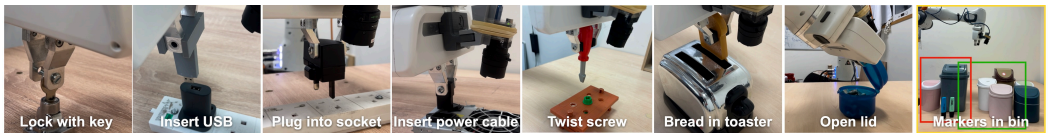

Figure 4: The tasks used in our experiments. The "Markers in Bin" is used to evaluate MILES' ability to generalize (the bins marked green denote the training set, while the red denote the test set).

tematic inaccuracies in a robot's controller may prevent it from reaching its target waypoint w_k^ζ. Thus, if $w_M^{\tau_k} \neq w_k^\zeta$, the augmentation trajectory cannot return to demonstration state k, rendering the augmentation trajectory invalid.

(2) **Condition 2, Environment Disturbance:** While collecting τ_k, the robot may disturb the environment, resulting in a final observation $o_M^{\tau_k}$ that no longer matches that of the demonstration (even if w_k^ζ is reached). For instance, during data collection if the robot's gripper pushes an object to a different pose than it had at timestep k of the demonstration, the final observation in the augmentation trajectory will differ from the demonstration's k_{th} observation. Therefore, if $o_M^{\tau_k} \neq o_k^\zeta$, the augmentation trajectory cannot be combined with the human demonstration to create a new, valid demonstration. To detect such disturbances, we compare the cosine similarity of the DINO features [27, 28] of the RGB image I_k^ζ from the demonstration's observation o_k^ζ and the image $I_M^{\tau_k}$ after executing the augmentation trajectory. If the similarity falls below a threshold θ, we assume the environment has been disturbed and stop data collection. In the supplementary material sections A.2, A.3, we provide visual explanations of our validity conditions and figures demonstrating the robustness of the DINO features in detecting varying environment disturbances for an object.

3.4 Policy

Training. We train a separate policy π for each task as an LSTM network with behavioral cloning that receives as input the RGB and force-torque observations in the dataset $\mathcal{D}_{new}$ and regresses the corresponding actions. Note that $\mathcal{D}_{new}$ does not contain proprioception data, allowing our policies to generalize to different object poses naturally due to the use of our wrist camera.

Inference. We deploy our policy π to solve a task up to the R_{th} demonstrated state. If no environment disturbance occurred during data collection for that task, then the R_{th} state is the final state in the demonstration and π solves the task completely in a closed-loop manner. Otherwise, after π completes the task up to the R_{th} state, the remaining demonstrated action segment $\zeta_{remaining}$ *is replayed*. More details regarding how we deploy our policy, the network architecture, and how we detect that π has reached the R_{th} state can be found in the supplementary material section A.4.

4 Experiments

We evaluate MILES through several real-world experiments. Through these experiments, we aim to answer the following questions: 1) Can MILES solve a range of everyday tasks and how does it perform against baselines that learn from a single demonstration? 2) How does MILES perform under different component ablations? 3) How important are vision and force modalities to the performance of MILES? and 4) How does MILES perform under different sizes of self-supervised data? Videos of our experiments can be found on our webpage: www.robot-learning.uk/miles

Implementation Details. For our experiments, we use a FLIR camera mounted to the wrist of a Franka Emika Robot. We sample $Z = 10$ augmentation trajectories for each demonstration waypoint ($\approx$ approximately 1 minute of data collection per waypoint). This number is set arbitrarily, but as we show later in our ablations, some tasks may require less data. We collect augmentation trajectories with initial poses near the demonstration in the range of 4cm and 4 degrees around each demonstration waypoint. As commonly done in the literature [3, 15, 16, 19], we provide our demonstrations starting near each object. At deployment, to reach the object from far away we first estimate the object's pose using pose estimation and approach it before switching to MILES. Finally, we set the environment disturbance threshold θ to 0.94 for all our tasks. Additional details on the pose estimation method we use and how to set each one of MILES' parameters can be found in our supplementary material sections B.1 and B.2.

Methods	Lock with key	Insert USB	Plug into socket	Insert power cable	Twist screw	Bread in toaster	Open lid	Mean
Demo Replay	0	0	0	0	0	15	25	6
Reset Free Residual RL	0	15	35	0	0	0	0	7
Reset Free FISH (Inverse Residual RL)	0	30	25	15	0	0	0	10
Pose Estimation + Demo Replay	50	10	85	80	70	**100**	**100**	71
MILES	**90**	**70**	85	**85**	**85**	95	**100**	**87**

Table 1: Task success rates (%) for 20 trials reported for each method.

4.1 Can MILES solve a range of everyday tasks and how does it perform against baselines that learn from a single demonstration?

In this experiment, we assess the performance of MILES across a diverse set of everyday tasks and compare it to various baseline methods capable of learning from a single demonstration. We select seven distinct tasks, shown in Figure 4, spanning a range of complexities, each learned from a single demonstration. The tasks are: 1) Lock with key; 2) Insert USB; 3) Plug into socket; 4) Insert power cable; 5) Twist screw; 6) Bread in toaster; 7) Open lid. Tasks 1-4 are *contact-rich* and and our setup follows prior work on contact-rich manipulation [15, 29, 30, 31, 32] and the NIST benchmark [33]. Similar to prior work [34, 35], we focus our evaluation on single-task performance, but also report results on generalization, robustness to distractors, and multi-stage tasks in section 4.5. We provide a detailed description of each task in the supplementary material section B.3 including data collection times, information on which tasks involve demonstration replay, which tasks stopped data collection due to an environment disturbance and information on the length of each human demonstration.

Baseline Methods. We chose 4 baselines that can learn from a single demonstration without prior task knowledge, similar to MILES. (1) **Demo Replay** which involves replaying the demonstrated actions. (2) **Pose Estimation + Demo Replay** follows [16] and leverages MILES' data to perform pose estimation followed by demonstration replay. (3) **Reset Free Residual RL** replays the demonstration's actions at each timestep and learns corrective actions on top using DDPG [36]. Like MILES, no human intervenes to reset the environment during training, hence we call it "Reset Free". Finally, (4) **Reset Free FISH (Inverse Residual RL)** uses the state-of-the-art inverse RL method FISH [3] but no human intervenes to reset the environment during training. Further, implementation details on the baselines can be found in our supplementary material section B.4.

Evaluation. For a fair evaluation, we carefully tuned each method's hyperparameters. Additionally, each learning-based baseline collected the same number of observations as MILES during data collection for each task. We evaluated each method's success rate across 20 trials. For each trial we randomized the relative starting pose of the robot and the task-relevant object equivalently across all methods within a sphere of 20cm around the object as long as the object was visible to the camera. Finally, we emphasize that for all evaluations both MILES and the baselines predict *6-DoF actions*.

Results. Table 1 presents the success rates of MILES and the 4 baselines across 7 tasks. As shown **MILES** obtains a high success rate across all tasks. Specifically, MILES obtains an average of 87% success rate across all tasks with the "Open lid" task having the highest success rate of 100%. From the contact-rich tasks, inserting the key and locking the lock achieved the best performance despite the task's low tolerance and complex interaction, with an impressive 90% success rate. The lowest success rate is observed in the USB insertion task, where MILES obtains 70%, a performance dip we attribute to the task's low tolerance of less than 1mm. Despite the USB task, for the remaining tasks that required high precision MILES was able to complete them consistently well.

The next best-performing method, **Pose Estimation + Demo Replay**, obtained an average success rate of 71%. Our experiments showed that small errors in object pose estimation led to failures due to the compounding errors of demonstration replay, as observed in prior work [2, 19]. This is particularly evident in tasks requiring precise manipulation including "Lock with key", "Insert USB" and "Twist screw". Instead, while MILES also replays part of the demonstration for some tasks, the fact that it does so for a much shorter horizon allows it to obtain considerably higher success rates.

Reset Free FISH failed to solve the majority of tasks, yielding an average performance of 10% across all evaluated scenarios. During training, for several tasks, the policy caused significant disturbances to the environment that made policy learning very hard without manual resetting. We believe that this is the central reason behind Reset Free FISH's low performance Additionally, compared to the original FISH implementation, as we train policies that predict 6-DoF actions the learning efficiency of Reset Free FISH is negatively affected. Lastly, **Reset Free Residual RL** obtained a

Method Ablations	Lock with key	Insert USB	Plug into socket	Insert power cable	Twist screw	Bread in toaster	Open lid	Mean
No Sequence	60	20	20	10	0	85	95	43
No Environment Disturbance	90	70	85	85	0	0	0	47
No Reachability	75	40	95	20	85	95	**100**	73
No Memory	50	65	**100**	75	35	90	**100**	74
MILES	**90**	**70**	85	**85**	**85**	95	**100**	**87**

Table 2: Task success rates (%) for 20 trials reported for each method ablation.

lower success rate, averaging 7% due to similar challenges with Reset Free FISH. As anticipated, the **Demo Replay** baseline was the least effective among the baselines. Simply replaying the demonstration without pose estimation or online action correction leads to task failures.

While the RL methods would have benefited from manual environment resets during training, our results demonstrate that in our "easy" imitation learning setting, where only a single demonstration is available without additional human interventions, MILES significantly outperformed the baselines.

Simulation Evaluation. Finally, to support the reproduction of our results, we conducted additional experiments on our method and the baselines in simulation using 5 tasks from the RLBench benchmark [37] which can be found in the supplementary material section C.1.

4.2 How does MILES perform under different method ablations?

This section studies MILES' performance by ablating 4 different components of the method: (1) **No Environment Disturbance:** we ablate the environment disturbance condition by not checking for that condition when collecting augmentation trajectories. (2) **No reachability:** we ablate the reachability condition by relabeling each observation's action (of the existing MILES data), to move the robot to the nearest waypoint in the demonstration based on their Euclidean distance. If the constraint for reachability is not important, then simply moving from each pose to the nearest waypoint in the demonstration in a straight line would be sufficient to solve a task. (3) **No sequence:** we recollect MILES' data but instead of collecting Z augmentation trajectories for the first demonstration state, then progressing to the second state and so on, we collect data *without* following the demonstration's waypoint sequence and instead follow a random one. (4) **No Memory:** For this ablation we retrain a network on the existing MILES data that does not account for history.

Results. Table 2 shows MILES performance after ablating each component. Collecting augmentation trajectories for each demonstration state in a random order (**No Sequence**), with an average success rate of 43%. Additionally, not checking for the environment disturbance condition (**No Environment Disturbance**) appears to cause significant performance degradation for the tasks where an environment disturbance occurred during data collection, corresponding mostly to the non-contact rich tasks. On the other hand, not checking for the reachability condition (**No Reachability**) also lowers performance, particularly for the precise, contact-rich tasks, indicating that the reachability condition is the most important when learning tasks requiring precise manipulation. Finally, the lower performance obtained by removing the LSTM (**No Memory**) demonstrates the performance benefits of training memory-based networks on datasets collected using MILES.

4.3 How important are vision and force modalities to the performance of MILES?

In this section, we ablate the use of vision and force feedback as policy inputs for the four contact-rich tasks from our earlier experiments. We retrain and evaluate two policies: one using only vision and one using only force. The results, shown in Figure 5, indicate that the vision-based policy improves MILES' performance in the "Insert USB" and "Plug into socket" tasks but reduces performance in the other two tasks. This suggests that force feedback might not consistently benefit MILES, possibly due to its noisy signal which makes it hard to distinguish between different environment states. The force-based policy, however, fails almost completely. This is expected as force feedback is zero in free space and can be ambiguous due to

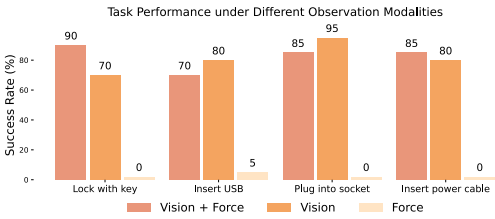

Figure 5: MILES' performance when trained only on either vision or force feedback or both.

symmetries in object surfaces. Overall, while force feedback aids performance in some tasks, it is not always necessary. Vision remains the most crucial modality to MILES' high performance.

4.4 How does MILES perform under different sizes of self-supervised data?

In this section, we ablate the dataset size used to learn four tasks by splitting their original datasets into chunks containing 75%, 50%, and 25% of the original data. We evaluated the best and worst performing contact-rich tasks ("Lock with key" and "Insert USB") and non-contact-rich tasks ("Open lid" and "Twist screw"). Data collection times for each task can be found in the supplementary material. Figure 6 shows that for high tolerance tasks like "Open lid," MILES achieves a 100% success rate even with 25% of the data, corresponding to *only* 8 minutes of data collection. However, for precise tasks, success rates decrease as dataset size is reduced. Notably, for "Lock with key" and "Twist screw," reducing the dataset to 50% results in a high failure rate. To summarize, we observe that high tolerance tasks are likely to require less data, and in practice only a few minutes of data collection time. Instead, for high-precision tasks, like inserting a USB, the dataset size appears to impact MILES' performance significantly.

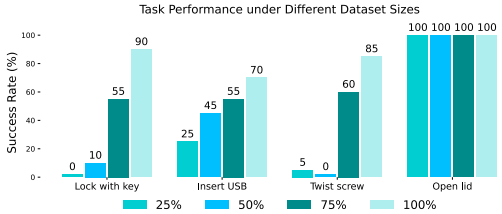

Figure 6: MILES' performance when trained on different dataset sizes. 100% corresponds to the original dataset. 75%, 50%, and 25% correspond to splits of the original dataset.

4.5 Further questions about MILES

Can MILES generalize? We conduct additional experiments on generalization for MILES in the supplementary material section D.1. **Can MILES perform multi-stage tasks?** We provide additional experimental results on how MILES performs on multi-stage tasks in the supplementary material section D.2. **Is MILES robust to distractor objects?** We conduct additional experiments studying MILES' performance in the presence of distractors in the supplementary material section D.3. **What if MILES stops data collection early due to a detected environment disturbance?** We provide a discussion and intuition on MILES' behavior in scenarios where data collection stops early in our supplementary material section D.4.

5 Discussion

Limitations. We now highlight some important limitations of our method. Firstly, MILES' reliance on a wrist camera enables MILES to obtain spatial generalization, however, simultaneously this limits its field of view and its applicability to larger task spaces. Future work could address this by incorporating an external camera to initially approach an object before switching to the wrist camera, similarly to [19]. Secondly, while MILES is robust to distractors at deployment before data collection begins it requires a human to set up the robot's workspace such that only the task-relevant object is in camera view for the policy to achieve spatial generalization. While this requires only a few seconds of human time, future work could address this by extending MILES to incorporate segmentation methods, similar to [19, 38], that segment the task-relevant object in the dataset. Similarly, to address any unwanted collisions that MILES could cause in the presence of multiple objects, future work could study incorporating an external camera during self-supervised data collection to plan and collect collision-free augmentation trajectories. Thirdly, our current implementation of MILES trains a separate policy for each task and hence it is unclear how well MILES would generalize to completely new tasks. In future work, we aim to study this by training a single monolithic policy on MILES' self-supervised data combined with replay-trajectory retrieval [39] similarly to our generalization task in section 4.5.

Conclusion. We introduced MILES, a framework that makes imitation learning easy. MILES requires only a single demonstration and collects self-supervised data that demonstrate to the robot how to return to and then follow that demonstration. Subsequently, this enabled us to obtain manipulation skills comprising either (1) a single end-to-end policy trained with behavioral cloning or (2) a combination of an end-to-end policy and demonstration replay. Our real-world experiments showed that when only a single demonstration is available self-supervised data enable the acquisition of skills that achieve considerably improved performance compared to several state-of-the-art baselines on many everyday tasks ranging from learning to open a lid to using a key to lock a lock or inserting a USB into a port, both of which require complex and precise contact-rich manipulation.

Acknowledgements

We would like to thank Norman Di Palo, Vitalis Vosylius, Pietro Vitiello, Kamil Dreczkowski, Yifei Ren and Ivan Kapelyukh for their insightful discussions on the method and feedback in drafting the first versions of this paper.

References

[1] A. Brohan et al. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

[2] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In *Conference on Robot Learning*, 2023.

[3] S. Haldar, J. Pari, A. Rai, and L. Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023.

[4] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Conference on Neural Information Processing Systems*, page 4572–4580, 2016.

[5] V. Mnih et al. Human-level control through deep reinforcement learning. *Nat.*, 518(7540):529–533, 2015.

[6] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

[7] O. X.-E. Collaboration et al. Open X-Embodiment: Robotic learning datasets and RT-X models, 2023.

[8] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. BC-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, 2021.

[9] Y. Hu, M. Cui, J. Duan, W. Liu, D. Huang, A. Knoll, and G. Chen. Model predictive optimization for imitation learning from demonstrations. *Robotics and Autonomous Systems*, 163, 2023.

[10] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell. Generalized task-parameterized skill learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5667–5474, 2018. doi: 10.1109/ICRA.2018.8461079.

[11] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *ArXiv*, abs/1709.04905, 2017.

[12] Z. Mandi, F. Liu, K. Lee, and P. Abbeel. Towards more generalizable one-shot visual imitation learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2434–2444, 2022. doi: 10.1109/ICRA46639.2022.9812450.

[13] G. Papagiannis and Y. Li. Imitation learning with sinkhorn distances. In *European Conference in Machine Learning and Knowledge Discovery in Databases*, 2022.

[14] W. Sun, A. Vemula, B. Boots, and D. Bagnell. Provably efficient imitation learning from observation alone. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6036–6045. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/sun19b.html.

[15] G. Schoettler, A. Nair, J. Luo, S. Bahl, J. Aparicio Ojea, E. Solowjow, and S. Levine. Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards. In *International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[16] E. Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[17] E. Valassakis et al. Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning. 2022.

[18] N. D. Palo and E. Johns. On the effectiveness of retrieval, alignment, and replay in manipulation. *RA-Letters*, 2024.

[19] P. Vitiello, K. Dreczkowski, and E. Johns. One-shot imitation learning: A pose estimation perspective. In *Conference on Robot Learning*, 2023.

[20] B. Wen, W. Lian, K. E. Bekris, and S. Schaal. You only demonstrate once: Category-level manipulation from single visual demonstration. *ArXiv*, abs/2201.12716, 2022.

[21] M. Laskey, J. Lee, R. Fox, A. D. Dragan, and K. Goldberg. Dart: Noise injection for robust imitation learning. In *Conference on Robot Learning*, 2017.

[22] L. Ke, J. Wang, T. Bhattacharjee, B. Boots, and S. Srinivasa. Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[23] A. Zhou, M. J. Kim, L. Wang, P. Florence, and C. Finn. Nerf in the palm of your hand: Corrective augmentation for robotics via novel-view synthesis, 2023.

[24] M. Jia, D. Wang, G. Su, D. Klee, X. Zhu, R. Walters, and R. Platt. Seil: Simulation-augmented equivariant imitation learning. In *International Conference on Robotics and Automation (ICRA)*, pages 1845–1851, 2023. doi:10.1109/ICRA48891.2023.10161252.

[25] L. Ke, Y. Zhang, A. Deshpande, S. Srinivasa, and A. Gupta. CCIL: Continuity-based data augmentation for corrective imitation learning. In *First Workshop on Out-of-Distribution Generalization in Robotics at CoRL 2023*, 2023.

[26] G. Cideron, B. Tabanpour, S. Curi, S. Girgin, L. Hussenot, G. Dulac-Arnold, M. Geist, O. Pietquin, and R. Dadashi. Get back here: Robust imitation by return-to-distribution planning, 2023.

[27] M. Caron, H. Touvron, I. Misra, H. J'egou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. *International Conference on Computer Vision (ICCV)*, 2021.

[28] S. Amir et al. Deep vit features as dense visual descriptors. *ECCVW What is Motion For?*, 2022.

[29] J. Luo, O. O. Sushkov, R. Pevceviciute, W. Lian, C. Su, M. Vecerík, N. Ye, S. Schaal, and J. Scholz. Robust multi-modal policies for industrial assembly via reinforcement learning and demonstrations: A large-scale study. *ArXiv*, abs/2103.11512, 2021.

[30] J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, and S. Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*, 2024.

[31] T. Z. Zhao, J. Luo, O. Sushkov, R. Pevceviciute, N. Heess, J. Scholz, S. Schaal, and S. Levine. Offline meta-reinforcement learning for industrial insertion. In *International Conference on Robotics and Automation (ICRA)*, pages 6386–6393, 2022.

[32] A. Nair, B. Zhu, G. Narayanan, E. Solowjow, and S. Levine. Learning on the job: Self-rewarding offline-to-online finetuning for industrial insertion of novel connectors from vision. In *International Conference on Robotics and Automation (ICRA)*, pages 7154–7161, 2023.

[33] K. Kimble, K. Van Wyk, J. Falco, E. Messina, Y. Sun, M. Shibata, W. Uemura, and Y. Yokokohji. Benchmarking protocols for evaluating small parts robotic assembly systems. *IEEE Robotics and Automation Letters*, 5(2):883–889, 2020.

[34] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[35] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023.

[36] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. M. O. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *ArXiv*, abs/1509.02971, 2015.

[37] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *CoRR*, abs/1909.12271, 2019. URL http://arxiv.org/abs/1909.12271.

[38] D. Seita, Y. Wang, S. J. Shetty, E. Y. Li, Z. Erickson, and D. Held. Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1038–1049. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/seita23a.html.

[39] N. Di Palo and E. Johns. Learning multi-stage tasks with one demonstration via self-replay. In *Conference on Robot Learning (CoRL)*, 2021.

[40] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '16, pages 770–778. IEEE, June 2016.

[41] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8), 1997.

[42] A. Mandlekar et al. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, 2021.

[43] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto. The surprising effectiveness of representation learning for visual imitation. *CoRR*, abs/2112.01511, 2021. URL https://arxiv.org/abs/2112.01511.