2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

# ARCADE: Scalable Demonstration Collection and Generation via Augmented Reality for Imitation Learning

Yue Yang[1], Bryce Ikeda[2], Gedas Bertasius[3], Daniel Szafir[4]

***Abstract*— Robot Imitation Learning (IL) is a crucial technique in robot learning, where agents learn by mimicking human demonstrations. However, IL encounters scalability challenges stemming from both non-user-friendly demonstration collection methods and the extensive time required to amass a sufficient number of demonstrations for effective training. In response, we introduce the *A*ugmented *R*eality for *C*ollection and gener*A*tion of *DE*monstrations (ARCADE) framework, designed to scale up demonstration collection for robot manipulation tasks. Our framework combines two key capabilities: 1) it leverages AR to make demonstration collection as simple as users performing daily tasks using their hands, and 2) it enables the automatic generation of additional synthetic demonstrations from a single human-derived demonstration, significantly reducing user effort and time. We assess ARCADE's performance on a real Fetch robot across three robotics tasks: *3-Waypoints-Reach*, *Push*, and *Pick-And-Place*. Using our framework, we were able to rapidly train a policy using vanilla Behavioral Cloning (BC), a classic IL algorithm, which excelled across these three tasks. We also deploy ARCADE on a real household task, *Pouring-Water*, achieving an 80% success rate.**

## I. Introduction

Imitation Learning (IL) aims to empower end-users to teach robots skills and behaviors through demonstrations and has shown promising results in controlled laboratory environments [1], [2], [3], [4]. Behavioral Cloning (BC) [5], a common form of IL, mimics human actions from demonstrations using supervised learning, showing effectiveness in complex scenarios [6], [7]. Compared to alternate approaches, such as adversarial imitation learning (AIL) [8], [9], BC stands out for its simplicity in implementation and optimization. Moreover, it functions offline, eliminating the need for potentially risky environmental interactions [10], [11]. These advantageous features make BC a promising choice for deploying robots in real household environments. However, to realize this vision, we must overcome two significant BC challenges, the *complex process of demonstration collection* and the *data-hungry* nature of BC algorithms [12].

Regarding the first challenge, current methods for gathering demonstrations often require familiarity with teleoperation using specific controllers (e.g., joystick, 3D mouse) [13], [14], [15] or contact-based kinesthetic teaching with robots [4], [10]. Such approaches may not be feasible or desirable for non-expert users. Recently, Virtual Reality (VR) has been explored as a potential method to simplify demonstration collection process [16], [17], but such approaches involve additional efforts such as creating realistic VR environments, leading to further complications. We are inspired by an alternative approach, suggested by [18], to use Augmented Reality (AR) to enable more natural collection of demonstrations.

Regardless of the demonstration collection method, BC introduces a second challenge in requiring a substantial volume of expert demonstrations, often in the hundreds, for effective training. Amassing such a quantity of demonstrations may be excessively burdensome for end users. This challenge is primarily due to the covariate shift issue [19], [20], where minor discrepancies in action prediction accumulate over time, leading the agent to encounter states not covered by the demonstrations.

We introduce ARCADE, a novel AR-based framework tailored to effectively address both challenges. ARCADE provides a three-step process for generating demonstrations in a user-friendly and scalable manner. First, a user provides a single demonstration using their hands as they would in daily life, addressing the *complex process of demonstration collection* challenge. During this process, the user wears an AR headset that tracks their hand motion and visualizes a robot digital twin overlaying the user. Second, ARCADE automatically generates a small set (10–15) of candidate demonstrations by following waypoints that are randomly sampled from the single user-collected demonstration, during which we apply a *Key-Poses Detector* to preserve the core elements of the user's demonstration. These candidates are visualized in AR to the user who can rapidly filter out any that contain errors (e.g., violating user preferences or implicit constraints) to form a user-accepted set of demonstrations. Third, ARCADE generates a large set of additional high-quality demonstrations, all based on the initial user demonstration, and uses an *Automatic Validation* approach to compare each candidate against the user's accepted set without further need for user input, thus rapidly scaling the demonstration set and addressing the *data-hungry* challenge to enable effective BC training. We summarize our contributions as:

1) We introduce a novel framework for generating demonstrations at scale from a single AR-captured demonstration.
2) Within this framework, we have developed two innovative techniques: a *Key-Poses Detector* and *Automatic Validation*, both designed to facilitate the generation of

[1]Yue Yang, Department of Computer Science, the University of North Carolina at Chapel Hill, 232 S Columbia St, Chapel Hill, NC, USA yygx@cs.unc.edu
[2]Bryce Ikeda, Department of Computer Science, the University of North Carolina at Chapel Hill, 232 S Columbia St, Chapel Hill, NC, USA bikeda@cs.unc.edu
[3]Gedas Bertasius, Department of Computer Science, the University of North Carolina at Chapel Hill, 232 S Columbia St, Chapel Hill, NC, USA gedas@cs.unc.edu
[4]Daniel Szafir, Department of Computer Science, the University of North Carolina at Chapel Hill, 232 S Columbia St, Chapel Hill, NC, USA daniel.szafir@cs.unc.edu

high-quality demonstrations from one user-provided AR demonstrations.

3) We evaluate ARCADE on a physical Fetch robot for three manipulation tasks. The BC-trained policy with ARCADE-generated demonstrations demonstrates excellent performance across all tasks. Further validation of ARCADE in a more complex *Pouring Water* task showed the robot achieving an 80% success rate, highlighting ARCADE's potential for realistic robot assistance in homes.

## II. Related Work

### A. Demonstration Collection Methods

Methods for collecting demonstrations in IL have evolved alongside the field itself. The earliest and also the most popular method involves users kinesthetically guiding the robot in a tactile manner to perform tasks [1]. However, the effectiveness of kinesthetic teaching depends on the user's dexterity and their ability to perform smooth demonstrations. This makes it challenging to gather large-scale datasets, as maintaining the stamina needed for repeated physical demonstrations is difficult. Teleoperation offers an alternative [21], [22], [23], where users control the robot using various devices such as a keyboard and mouse [24], [25], 3D-mouse [26], [14], [27], joysticks [28], or mobile phones [29] to perform tasks. Unfortunately, Jiang et al., 2024 shows that such systems typically require a longer training process for users to effectively operate these systems and they may achieve the worst performance compared to alternative methods [30]. Recent research has explored the use of VR to streamline the process of demonstration collection, offering users a potentially more intuitive way to control robots [31], [17], [32], [33], [16], [12]. However, such VR methods necessitate the development of a simulation environment and remove the demonstration process from the contexts of real-world applications. To eliminate the need for a real robot during demonstration collection, Duan et al., 2023 suggests using AR for this purpose, which inspires our framework.

### B. Behavioral Cloning

Behavioral Cloning (BC), a key technique in Imitation Learning (IL), effectively completes tasks by replicating demonstrations provided by users [5]. For example, Ratliff et al., 2007 apply BC to manipulation and locomotion tasks by converting them into multiclass classification problems, subsequently solved via supervised learning [34]. However, one major challenge for BC methods is addressing covariate shift (also known as compounding errors) [19], [20], which results in action predictions for out-of-distribution states. To address covariate shift, Dataset Aggregation (DAgger) [19] can improve robot policies during training, but may be taxing for users as it requires continuous feedback throughout the training process. An alternative solution is to supply a large number of high-quality demonstrations that encompass a broader state space, thereby enhancing the performance of the BC task [6], [35], [36], yet this too can be burdensome for users due to the stamina required to create high quality large scale datasets. Recently, George et al., 2023 proposed generating multiple demonstrations from a single VR-collected demonstration [12]. However, this approach does not guarantee alignment with user preferences, potentially affecting the quality of the generated demonstrations. Instead, our framework ensures all generated demonstrations match user preferences by deriving them from a user-approved set after reviewing AR-visualized options.

## III. Methodology

### A. Framework Overview

We introduce the ARCADE as a framework for generating demonstrations in a user-friendly and scalable manner, as illustrated in Figure 1. §III-B details the initial AR-based user demonstration (Figure 1A). Next, §III-C describes the method for generating demonstrations (Figure 1B). Then, §III-D details the method for users to validate generated demonstrations (Figure 1C). Finally, §III-E introduces an automated validation approach for these demonstrations to rapidly scale up the size of the demonstration set (Figure 1D).

Our framework integrates three underlying techniques: Markov Decision Processes (MDPs), Dynamic Time Warping (DTW), and Behavioral Cloning (BC). We model the environment using an MDP, denoted as $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, R, T, \rho_0 \rangle$. Here, $\mathcal{S}$ represents the state space, for which we use the robot arm's joint values, and $\mathcal{A}$ denotes the action space, defined by changes in the arm's joint. The reward function is given by $R$, $T$ is the deterministic transition function, and $\rho_0$ represents the initial state probability distribution. We use DTW [37], an algorithm for quantifying the similarity between temporal sequences that may differ in timing or speed, as a crucial tool for *Automatic Validation*. We employ BC as our IL algorithm, training a policy, $\pi_\theta$, to mimic demonstrations, $\Xi = \{\tau_i\}_{i=1}^{M}$, where each demonstration is a list of state-action pairs, $\tau = \{(s_i, a_i)\}_{i=1}^{N}$.

### B. AR-assisted Demonstration Collection

To collect demonstrations of robot arm trajectories from users, we utilized the Microsoft HoloLens 2, an augmented reality head-mounted display (ARHMD). During the demonstration collection process (Figure 2), users wear the ARHMD, which overlays a digital twin of the robot on the user and provides an egocentric view of the robot's perspective. This setup facilitates real-time visual feedback of the robot's movements to the user. For our tasks and learning algorithm, the robot's end-effector must align with and track the user's hand. We accomplish this by using current inverse kinematics (IK) algorithms [38], which calculate the robot's joint angles based on the demonstrator's hand position. The distance between the user's pointer finger and thumb is used to inform gripper movements (open/close) for picking or placing objects. As the joint angles, end-effector positions, and pick or place actions are calculated during the demonstrations, this information is recorded on a separate machine, transmitted from the HoloLens via Unity Robotics Hub's ROS-TCP-Endpoint and ROS-TCP-Connector [39].

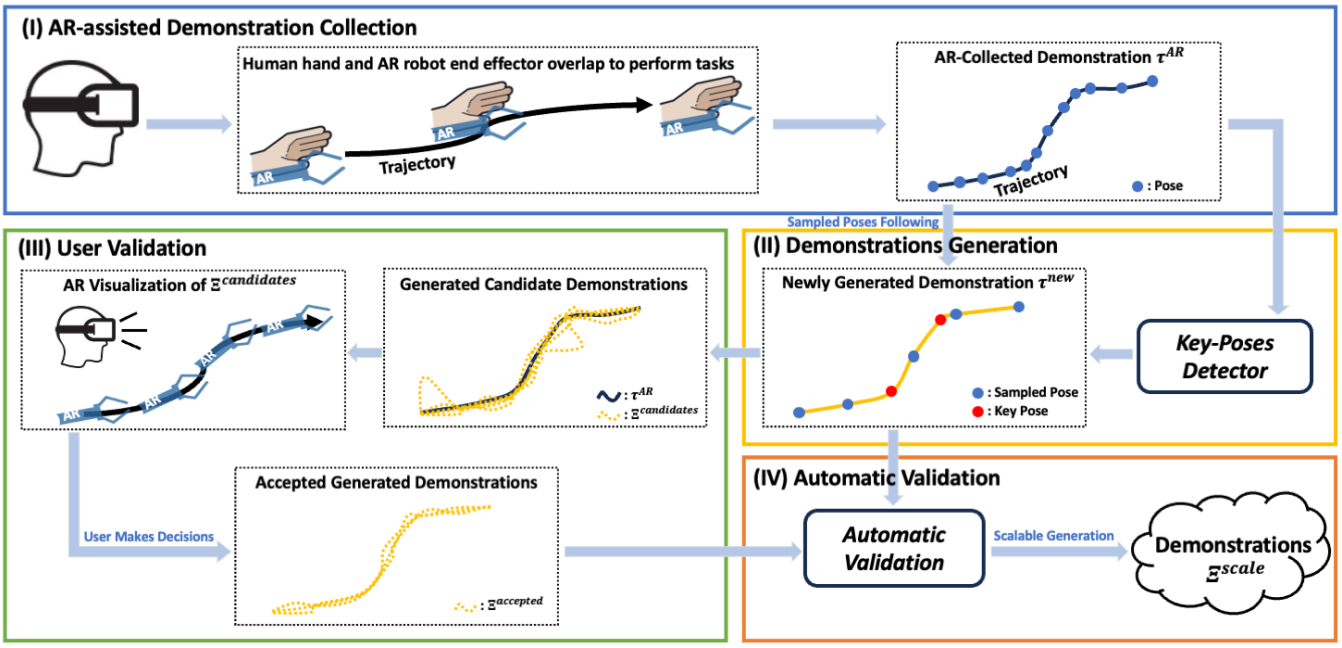


Fig. 1: This figure shows the architecture of ARCADE. (I) First, a user provides a single demonstration, $\tau^{AR}$, through AR. (II) We generate a new demonstration, $\tau^{new}$, by following sampled poses, extracted from $\tau^{AR}$, and key poses, obtained via *Key-Poses Detector*. (III) Additional candidate demonstrations are then generated, which are visualized in AR for user validation. Users filter the candidate demonstrations to form an accepted set of generated demonstrations, $\Xi^{accepted}$. (IV) Finally, we continue generating additional new demonstrations, automatically determining whether to keep or discard each demonstration based on comparing it to $\Xi^{accepted}$ via *Automatic Validation*.

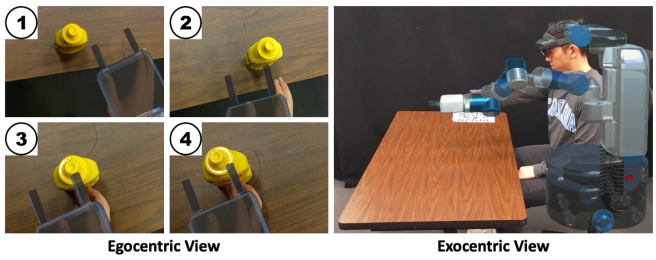


Fig. 2: Left: Egocentric view showing the robot's end effector overlapping with and following the human hand's movements to perform the Push task. Right: Exocentric view showing how the human performs the task manually, with the digital twin robot end effector mirroring the hand movements.

With this setup, we record a single demonstration $\tau^{AR}$ from the user with a form as shown in Equation 1. Each demonstration includes $N$ data points, with the form of end-effector pose, $p_i$, corresponding robot arm joints, $j_i$, and binary gripper state, $g_i$ for each timestep $i = 1, ..., N$.

$$\tau^{AR} := \{(p_i, j_i, g_i)\}_{i=1}^{N} \tag{1}$$

### C. Demonstrations Generation

Successfully collecting a single high-quality demonstration, $\tau^{AR}$, through user-friendly AR methods addresses one challenge, *complex process of demonstration collection*, mentioned in §I. However, gathering a sufficient number of demonstrations with minimal user effort remains a hurdle. We automatically generate additional demonstrations based on the initial user-provided $\tau^{AR}$. Such generated demonstrations must meet two criteria for effective BC: 1) they should encompass a broader state space than the states in $\tau^{AR}$ and ensure task completion simultaneously, and 2) they must maintain similarity to $\tau^{AR}$, as IL algorithms are less effective with heterogeneous demonstrations [3].

To satisfy the first criterion, we use a waypoint-following approach. For waypoints sampling, we utilize a random interval length method to select poses from $\tau^{AR}$. Considering the poses, $\mathcal{P}^{AR} = \{p_i\}_{i=1}^{N}$, extracted from $\tau^{AR}$, we choose one pose at every $t$ timesteps, where $t$ is randomly determined within the range $[j, k]$ for each selection. By navigating through these randomly determined waypoints, the robot arm can explore a broader range of joints or end-effector states, irrespective of the specific definition of the state space.

However, to ensure critical waypoints necessary for task completion are not inadvertently filtered out during the sampling process, we introduce an automatic *Key-Poses Detector*. Our method identifies key poses assuming that they occur either during grasping and releasing actions or at moments of significant angle changes in the user's hand trajectory coupled with slow movement (i.e., approaching zero velocity). Algorithm 1 details the pseudocode for the *Key-Poses Detector*. For grasp and release actions (Line 1), the collected demonstration, $\tau^{AR}$, inherently provides the required information. In situations involving angle changes, the function $ComputeAngle(\cdot)$ calculates the angle at the

**Algorithm 1:** *Key-Poses Detector*

**Input :** *points*: positions extracted from collected poses; *window_length*: the duration over which we compute pose angles and density; *sharp_turn_threshold*; *dense_region_threshold*.

1 $grasp\_release\_indices \leftarrow \{g_i^{grasp}\}_{i=1}^{G} \cup \{g_i^{release}\}_{i=1}^{R}$
2 $sharp\_turn\_indices, dense\_region\_indices \leftarrow$ Empty list
3 **for** *idx, point* ***in*** *enumerate(points)* **do**
4 $\quad angle \leftarrow$ `ComputeAngle`$(point, window\_length)$
5 $\quad$ **if** $angle > sharp\_turn\_threshold$ **then**
6 $\quad\quad sharp\_turn\_indices.append(idx)$
7 $\quad density\_score \leftarrow$ `ComputeDensity`$(point, window\_length)$
8 $\quad$ **if** $density\_score > dense\_region\_threshold$ **then**
9 $\quad\quad dense\_region\_indices.append(idx)$
10 $key\_poses\_indices \leftarrow grasp\_release\_indices \cup (sharp\_turn\_indices \cap dense\_region\_indices)$

**Output :** *key_poses_indices*

current position, using the start and end positions of the window to help identify sharp turns (Lines 5-6). Although the collected demonstration records only position-based data without velocity, we employ $ComputeDensity(\cdot)$ to gauge the density of neighboring poses within a window by calculating their average pairwise distances (Line 7), serving as a proxy to detect slow movements (Lines 8-9).

The combined set of sampled waypoints and detected key waypoints constitutes the waypoint set, denoted as $\mathcal{W}$. To reach these waypoints in $\mathcal{W}$, we use MoveIt's [40] built-in motion planner [41] and IK [42], which tends to yield diverse trajectories, thereby encompassing a larger state space.

### D. User Validation

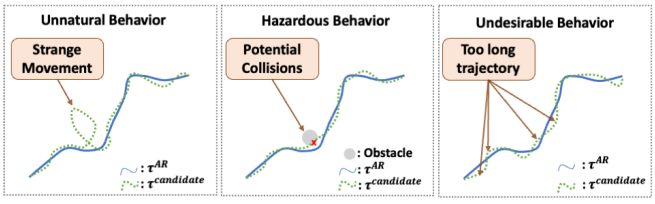


Fig. 3: Visualized candidate demonstrations may exhibit behaviors that could lead to rejection by the user: (left) unnatural motions due to poor IK solutions, (middle) potentially hazardous motions, (right) misalignment with the user's preferences.

Retaining key poses, as detailed in § III-C, aids in meeting the second criterion: maintaining similarity between $\tau^{AR}$ and the newly generated demonstrations, $\tau^{new}$. However, this strategy alone is inadequate to tackle the problem of heterogeneous demonstrations fully because of three potential scenarios (illustrated in Figure 3) that might occur in $\tau^{new}$:

**Algorithm 2:** *Automatic Validation*

**Input :** $\tau^{new}$: new candidate generated from $\tau^{AR}$; $\Xi^{accepted}$: accepted set of generated demonstrations; $\beta$: the acceptable level.

1 $\Xi^{scale} \leftarrow$ Empty list
2 $\mathcal{S} := \{\sum_{j=1, j\neq i}^{H} \texttt{DTW}(\tau_i^{accepted}, \tau_j^{accepted})\}_{i=1}^{H}$
3 $l \sim \text{Uniform}(1, H)$
4 $\delta = \sum_{i=1, i\neq l}^{H} \texttt{DTW}(\tau^{new}, \tau_i^{accepted})$
5 **if** $\delta \leq \beta \min(\mathcal{S})$ **then**
6 $\quad \Xi^{scale}.append(\tau^{new})$

**Output :** $\Xi^{scale}$

1) unnatural movements resulting from IK instability [38]; 2) potentially hazardous behaviors (e.g., the robot arm being too close to a table surface); and 3) any behaviors that the user may find undesirable (e.g., some users may favor shorter trajectories while some others might prioritize more human-like movements). Therefore, we must validate the behaviors in the generated demonstrations. To accomplish this, we initially create a set of generated candidate demonstrations, denoted as $\Xi^{candidates} := \{\tau_i^{new}\}_{i=1}^{H}$. Subsequently, we present each candidate in the set $\Xi^{candidates}$ to the user for validation, via AR. Viewing demonstrations in AR enables users to identify any problems in $\tau^{new}$ and decide whether to retain or remove the demonstration. The result of this process is a set of user-accepted demonstrations, $\Xi^{accepted} := \{\tau_i^{accepted}\}_{i=1}^{H}$. We believe this interactive approach, where users observe and filter a small set of automatically generated demonstrations, may be substantially less demanding and more efficient for users compared to traditional methods in which users must manually generate their own additional demonstrations.

### E. Automatic Validation

Effective training of BC often requires a large number of demonstrations, typically in the hundreds. Even with user's role shifted to validating generated demonstrations, the volume of necessary demonstrations for effective BC might still be daunting. Thus, we designed an automated method for scaling up the validation of generated demonstrations, where the user only needs to observe and approve a small set (e.g., 10–15 candidate demonstrations), after which the system can autonomously generate and self-validate candidates based on the characteristics of the user-approved set.

As shown in Algorithm 2, we leverage $\Xi^{accepted}$ to construct a similarity array, $\mathcal{S}$, utilizing dynamic time warping, $DTW(\cdot)$ (§III-A). Each element in $\mathcal{S}$ quantifies the similarity for every pair of user-accepted demonstrations, $\Xi^{accepted}$. Subsequently, we assess each newly generated candidate demonstration, $\tau^{new}$, against $\Xi^{accepted}$, excluding one randomly for fair comparison, utilizing $DTW(\cdot)$ (line 3-4). Acceptance of the newly generated demonstration, $\tau^{new}$, only occurs when $\delta \leq \beta \min(\mathcal{S})$ (line 5-6). The parameter $\beta$, a scalar (e.g., 0.95), determines the acceptable level: a higher acceptable level (i.e., smaller $\beta$) results in more similar generated demonstrations but smaller coverage of the state

space, and vice versa. This automatic filtering mechanism ensures that we only retain demonstrations aligning with user preferences. This process eliminates the need for constant user supervision (i.e., after the initial interaction where the user provides a single demonstration of their own and then filters generated candidates to the approved set of 10–15 demonstrations, no further user input is needed) and thus enhances the scalability of BC demonstration generation. Following the scalable, automated generation of demonstrations, BC models can be trained as usual, using the extensive set of demonstrations, $\Xi^{scale}$.

## IV. System Validation

We evaluate our framework using a real Fetch robot. We first examined performance on three archetypal tasks, *3-Waypoints-Reach*, *Push*, and *Pick-And-Place* (Figure 4), chosen because they exemplify fundamental manipulation behaviors that, when combined, can accomplish a variety of complex household activities. We provide an example of this in a fourth, more complex *Pouring Water* task (§IV-C).

In the *3-Waypoints-Reach* task, the robot arm aims to hit three predefined waypoints: $W_1$, $W_2$, and $W_3$. For the *Push* task, the robot arm must push an object on the desk from its starting position to a predefined goal point. The *Pick-And-Place* task involves the robot arm grasping an object, moving it to another location, and then releasing it. For *3-Waypoints-Reach* and *Push*, the state space consists of 7 arm joints, and the action space includes 7 delta arm joints, corresponding to each joint's movement. For *Pick-And-Place*, we expand the state space to eight dimensions to include the gripper angle, and the action space also increases to eight dimensions, encompassing the delta changes in the gripper.

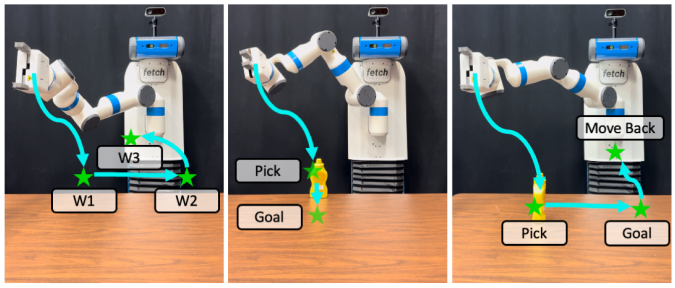


Fig. 4: We evaluate on three tasks: Left: 3-Waypoints-Reach, Middle: Push, Right: Pick-And-Place.

We introduce a *Task Completion Error (TCE)* metric, measured in meters, to evaluate BC performance across the three archetypal tasks. For the *3-Waypoints-Reach* task, we calculate this metric by averaging the minimum distances to the three waypoints during the evaluation. For the *Push* and *Pick-And-Place* tasks, it measures the distance from the object's final position to its target goal.

To benchmark ARCADE's effectiveness, we compared four BC policies across 3 archetypal tasks:

- ARCADE ($\tau^{AR}$): a policy trained using just the initial user AR demonstration from ARCADE.
- ARCADE ($\Xi^{scale}$): a policy trained with the full ARCADE system, consisting of 100 generated demonstrations.
- BL ($\tau^{ki}$): a baseline (BL) policy trained using a single demonstration collected via kinesthetic teaching, $\tau^{ki}$.
- BL ($\Xi^{scale\_ki}$): a baseline policy trained using 100 demonstrations, $\Xi^{scale\_ki}$, generated identically to $\Xi^{scale}$ but based on $\tau^{ki}$ instead of $\tau^{AR}$.

To assess each task, we execute each of the four BC-learned policies ten times, reporting the mean and standard deviation of the *TCE* for each policy as our results.

### A. Implementation Details

*1) Hardware:* We utilized the Microsoft HoloLens 2 as our augmented reality head-mounted display (ARHMD). The HoloLens 2 tracks both the position and orientation of the wearer's hands, allowing us to map the user's hand movements to the robot during the initial demonstration. It also enables the visualization of virtual imagery, which we use to align a digital twin of the robot with the user's movements during the demonstration and to display candidate demonstrations for the user to review and select for the accepted demonstration set. Our current implementation utilizes the Fetch robot, a mobile manipulation platform with a 7 degree-of-freedom arm, although our work can be generalized to any robot manipulator.

*2) Policy Architecture:* To represent the BC policy $\pi_\theta$, we use a GaussianMLP model [43]. This model is composed of two multilayer perceptrons, one for producing the mean $\mu$ and the other for the standard deviation $\sigma$, together forming a Gaussian distribution. The robot arm's action is then sampled from this distribution, denoted as $a \sim \pi_\theta$.

### B. Archetypal Task Results

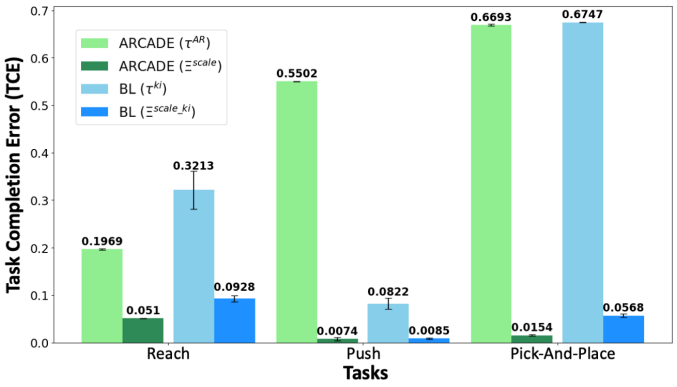


Fig. 5: The results of BC policies trained using ARCADE or a kinesthetic teaching baseline (BL) with either 1 or 100 demonstrations across three tasks. The full (100 demonstration set: $\Xi^{scale}$) ARCADE system offers the best performance in all tasks.

Figure 5 illustrates the results of each of the four BC policies across the three archetypal tasks. We conducted a two-way analysis of variance (ANOVA) to test whether the type of

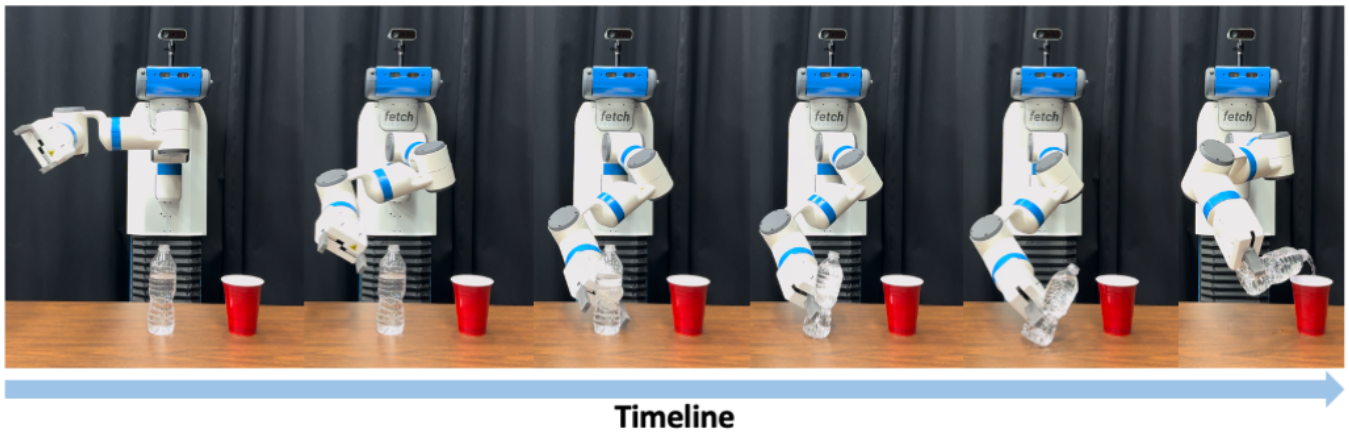


Fig. 6: The robot successfully pours water from a single-user demonstration, showcasing ARCADE's effectiveness in equipping robots for practical household chores.

demonstration collection (kinesthetic teaching baseline vs our ARCADE framework) and the size of the demonstration set ($|\Xi| = 1$ or $|\Xi| = 100$) affected BC policy TCE. We found significant main effects of both factors and their interaction on TCE at $p < .001$ for all three tasks. Using Tukey's Honestly Significant Difference (HSD) test to compare the performance of all four BC policies, we found that all four policies performed significantly differently ($p < .0001$ for each comparison), with performance ordered from best to worst as the ARCADE ($\Xi^{scale}$) (best), BL ($\Xi^{scale\_ki}$), ARCADE ($\tau^{AR}$), and BL ($\tau^{ki}$) (worst) for the 3-Waypoint-Reach and Pick-and-Place tasks. For the push task, Tukey's HSD did not reveal a significant difference between the performance of the ARCADE ($\Xi^{scale}$) and BL ($\Xi^{SCLAE\_ki}$) ($p = .976$), with both of them significantly better than the BL ($\tau^{ki}$) ($p < .0001$), which itself outperformed the ARCADE ($\tau^{AR}$) ($p < .0001$). These findings indicate that the ARCADE framework can generate demonstrations that match or surpass those from traditional kinesthetic teaching in terms of BC policy performance. Furthermore, the results show that both sets of demonstrations generated by our method, $\Xi^{scale}$ and $\Xi^{scale\_ki}$, facilitate effective BC training.

### C. Real Household Task - Pouring Water

To demonstrate ARCADE's capability in handling more complex household tasks and its potential for widespread home robot deployment, we introduce an additional task: *Pouring-Water*. Here, the goal is for the robot to learn to grasp a bottle and pour water into a cup from just a single demonstration given by the user. This task utilizes the same state and action spaces as the *Pick-And-Place* task and is deemed successful when water is poured into the cup. Testing the BC policy trained with $\Xi^{scale}$ from ARCADE, we achieved an 80% success rate (8 of 10 trials), with failures attributed to the plastic bottle's shape alteration. Figure 6 captures a successful instance of the robot performing the pouring action.

## V. Conclusion

We introduce ARCADE, a scalable framework that allows the collection of numerous high-quality demonstrations from a single user-collected demonstration via AR. This approach offers a user-friendly and time-efficient method for demonstration collection. Empirical evaluations across three archetypal robot tasks demonstrate ARCADE's effectiveness in generating high-quality demonstrations suitable for effectively training IL algorithms. Applying ARCADE to the real household task of *Pouring-Water* illustrates the framework's potential to facilitate the widespread integration of robots into daily home life.

## VI. Acknowledgements

This work was sponsored by the National Science Foundation (NSF) under award #2222953. This work was also supported by the Sony Faculty Innovation Award, Laboratory for Analytic Sciences via NC State University, ONR Award N00014-23-1-2356.

## References

[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[2] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. A. Theodorou, and B. Boots, "Imitation learning for agile autonomous driving," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 286–302, 2020.

[3] L. Chen, R. Paleja, and M. Gombolay, "Learning from suboptimal demonstration via self-supervised reward regression," in *Conference on robot learning*. PMLR, 2021, pp. 1262–1277.

[4] L. Chen, R. Paleja, M. Ghuy, and M. Gombolay, "Joint goal and strategy inference across heterogeneous demonstrators via reward network distillation," in *Proceedings of the 2020 ACM/IEEE international conference on human-robot interaction*, 2020, pp. 659–668.

[5] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[6] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion," *Robotics and autonomous systems*, vol. 47, no. 2-3, pp. 79–91, 2004.

[7] Y. Liu, D. Romeres, D. K. Jha, and D. Nikovski, "Understanding multimodal perception using behavioral cloning for peg-in-a-hole insertion tasks," *arXiv preprint arXiv:2007.11646*, 2020.

[8] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adverserial inverse reinforcement learning," in *International Conference on Learning Representations*, 2018.

[9] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016.

[10] Y. Yang, L. Chen, Z. Zaidi, S. Waveren, A. Krishna, and M. Gombolay, "Enhancing safety in learning from demonstration algorithms via control barrier function shielding," in *Proceedings of International Conference on Human-Robot Interaction (HRI)*, 2024.

[11] Y. Yang, L. Chen, and M. Gombolay, "Safe inverse reinforcement learning via control barrier function," *arXiv preprint arXiv:2212.02753*, 2022.

[12] A. George and A. B. Farimani, "One act play: Single demonstration behavior cloning with action chunking transformers," *arXiv preprint arXiv:2309.10175*, 2023.

[13] N. Freymuth, N. Schreiber, A. Taranovic, P. Becker, and G. Neumann, "Inferring versatile behavior from demonstrations by matching geometric descriptors," in *Conference on Robot Learning*. PMLR, 2023, pp. 1379–1389.

[14] S. Stepputtis, M. Bandari, S. Schaal, and H. B. Amor, "A system for imitation learning of contact-rich bimanual manipulation policies," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 810–11 817.

[15] M. Sakr, M. Freeman, H. M. Van der Loos, and E. Croft, "Training human teacher to improve robot learning from demonstration: A pilot study on kinesthetic teaching," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2020, pp. 800–806.

[16] A. George, A. Bartsch, and A. B. Farimani, "Openvr: Teleoperation for manipulation," *arXiv preprint arXiv:2305.09765*, 2023.

[17] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5628–5635.

[18] J. Duan, Y. R. Wang, M. Shridhar, D. Fox, and R. Krishna, "Ar2-d2: Training a robot without a robot," in *Conference on Robot Learning*. PMLR, 2023, pp. 2838–2848.

[19] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[20] J. Chang, M. Uehara, D. Sreenivas, R. Kidambi, and W. Sun, "Mitigating covariate shift in imitation learning via offline data with partial coverage," *Advances in Neural Information Processing Systems*, vol. 34, pp. 965–979, 2021.

[21] K. Goldberg, M. Mascha, S. Gentner, J. Tossman, N. Rothenberg, C. Sutter, and J. Wiegley, "Beyond the web: Excavating the real world via mosaic," in *Second International WWW Conference*, 1994, pp. 1–12.

[22] V. J. Lumelsky and E. Cheung, "Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 1, pp. 194–203, 1993.

[23] P. F. Hokayem and M. W. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.

[24] A. E. Leeper, K. Hsiao, M. Ciocarlie, L. Takayama, and D. Gossow, "Strategies for human-in-the-loop robotic grasping," in *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, 2012, pp. 1–8.

[25] D. Kent, C. Saldanha, and S. Chernova, "A comparison of remote robot teleoperation interfaces for general object manipulation," in *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 2017, pp. 371–379.

[26] A. D. Dragan and S. S. Srinivasa, "Online customization of teleoperation interfaces," in *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2012, pp. 919–924.

[27] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 785–799.

[28] M. Laskey, C. Chuck, J. Lee, J. Mahler, S. Krishnan, K. Jamieson, A. Dragan, and K. Goldberg, "Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 358–365.

[29] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay *et al.*, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," in *Conference on Robot Learning*. PMLR, 2018, pp. 879–893.

[30] X. Jiang, P. Mattes, X. Jia, N. Schreiber, G. Neumann, and R. Lioutikov, "A comprehensive user study on augmented reality-based data collection interfaces for robot learning," in *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, 2024, pp. 333–342.

[31] D. Whitney, E. Rosen, E. Phillips, G. Konidaris, and S. Tellex, "Comparing robot grasping teleoperation across desktop and virtual reality with ros reality," in *Robotics Research: The 18th International Symposium ISRR*. Springer, 2019, pp. 335–350.

[32] J. I. Lipton, A. J. Fay, and D. Rus, "Baxter's homunculus: Virtual reality spaces for teleoperation in manufacturing," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 179–186, 2017.

[33] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira, "Perceiver: General perception with iterative attention," in *International conference on machine learning*. PMLR, 2021, pp. 4651–4664.

[34] N. Ratliff, J. A. Bagnell, and S. S. Srinivasa, "Imitation learning for locomotion and manipulation," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2007, pp. 392–397.

[35] F. Sasaki, T. Yohira, and A. Kawaguchi, "Sample efficient imitation learning for continuous control," in *International conference on learning representations*, 2018.

[36] A. Reichlin, G. L. Marchetti, H. Yin, A. Ghadirzadeh, and D. Kragic, "Back to the manifold: Recovering from out-of-distribution states," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 8660–8666.

[37] M. Müller, "Dynamic time warping," *Information retrieval for music and motion*, pp. 69–84, 2007.

[38] D. Rakita, B. Mutlu, and M. Gleicher, "Relaxedik: Real-time synthesis of accurate and feasible robot arm motion." in *Robotics: Science and Systems*, vol. 14. Pittsburgh, PA, 2018, pp. 26–30.

[39] Unity-Technologies. (2020) Unity-robotics-hub. [Online]. Available: https://github.com/Unity-Technologies/Unity-Robotics-Hub.git

[40] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: a moveit! case study," *arXiv preprint arXiv:1404.3785*, 2014.

[41] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[42] R. Smits, "KDL: Kinematics and Dynamics Library," http://www.orocos.org/kdl.

[43] T. garage contributors, "Garage: A toolkit for reproducible reinforcement learning research," https://github.com/rlworkgroup/garage, 2019.