2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

COBALT: Crowdsourcing Robot Learning via Cloud-Based Teleoperation with Smartphones

Ayush Agarwal[1*], Ansh Gandhi[1,2*], Jeremy A. Collins[1], Omar Rayyan[3], Aryan Sarswat[1], Ranjani Koushik[1], Masoud Moghani[4], Ajay Mandlekar[5], Animesh Garg[1]

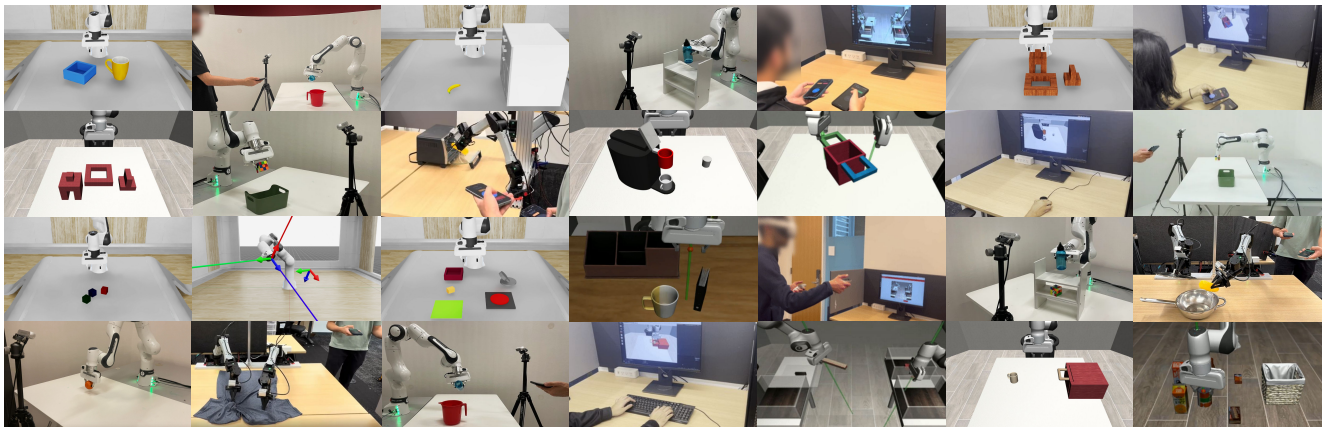

Fig. 1: COBALT can be used to collect data across a variety of both simulated and real-world environments, including bimanual tasks.

***Abstract*— The scarcity of large-scale, high-quality demonstration data remains a bottleneck in scaling imitation learning for robotic manipulation. We present COBALT, a teleoperation platform designed to democratize robot learning at scale both in simulation and in the real world. By leveraging vectorized environments, our scalable, load-balanced infrastructure supports concurrent teleoperation by multiple users on a single GPU, yielding a significant reduction in teleoperation cost. Operators can connect from nearly anywhere on Earth using commonly available devices, including single or dual smartphones, VR headsets, 3D mice, and keyboards. An in-memory data cache and efficient video streaming keep control and rendering synchronous, sustaining dozens of concurrent users at 20 Hz with sub-100 ms end-to-end latency. We demonstrate concurrent support for 256 clients across 8 GPUs, underscoring the system's ability to scale across hardware and within individual servers. We perform a comprehensive user study showing that phone-based teleoperation performs comparably to or better than specialized hardware, enabling faster, more ergonomic data collection. To ensure data quality, COBALT logs a suite of real-time metrics to automatically filter suboptimal demonstrations. We further demonstrate that a structured user training curriculum significantly improves data collection quality. Guided by insights from our user study, we crowdsource the collection of a large-scale, high-quality pilot dataset with 7500+ demonstrations (50+ hours) collected with smartphones across nine countries over five days. We validate the dataset's quality by training state-of-the-art imitation learning algorithms. Please visit cobalt-teleop.github.io for more details.**

I. INTRODUCTION

The long-term vision of robotics increasingly relies on data-driven methods like imitation learning [1], [2], [3], [4], [5], [6]. While these techniques efficiently teach robots skills using human demonstrations, their ability to generalize remains severely limited by the quantity, quality, and diversity of available training data. This constitutes a fundamental bottleneck: compared to the billions of images and trillions of text tokens fueling foundation models in computer vision and NLP [7], [8], robotics operates in a data desert, with even the largest datasets being orders of magnitude smaller ($\mathcal{O}(10^6)$ trajectories) [9], [10], [11], [12]. Bridging this vast data gap is arguably the most critical step toward realizing robots with broad capabilities for assisting humans across diverse tasks and environments.

Gathering demonstrations on physical hardware is notoriously time-consuming and cost-prohibitive, limiting dataset size and diversity. Collecting demonstrations in simulation serves as a complementary method of scaling robotics data. Modern simulation frameworks, such as MuJoCo [13] and Isaac Sim [14], accelerate dataset generation by enabling fast creation of diverse teleoperation environments [15], [16], [17], [18]. Regardless of the simulated or physical teleoperation environment, a critical obstacle in bridging the data gap is how effectively human operators can be brought into the loop to provide high-quality demonstrations. This is heavily influenced by factors such as cost, ease of onboarding, ergonomics, and ease of use.

We present COBALT, a scalable data collection platform that leverages cloud-based infrastructure to enable teleoperation from geographically distributed users using a variety of off-the-shelf devices. COBALT is the first to support concurrent, uninterrupted teleoperation across GPU-accelerated vectorized simulation environments (supporting Isaac Lab [19], robosuite [20], and LIBERO [21]), significantly improving the cost and efficiency of data collection. COBALT is globally deployable and supports remote operators connecting

*denotes equal contribution
[1]Georgia Institute of Technology [2]University of California, Berkeley
[3]University of California, Los Angeles [4]University of Toronto
[5]NVIDIA

TABLE I: Comparison of Existing Teleoperation Techniques in Literature. **S** = Smartphone, **VR** = Virtual Reality, **3DM** = 3D mouse, **K** = keyboard, **J** = Joystick. *TeleMoMa data collection infrastructure is not public. **Assumes RTX 3090, real value varies per GPU. Assumes zero marginal cost of a smartphone.

Method	RoboTurk [16]	MoMaRT [22]	TeleMoMa [15]	RoboTurk Real-World [23]	GELLO [24]	ALOHA [6]	COBALT
Device Cost	$0-$500	$0-$500	$0-$500	$0-$500	∼$300	∼ $20k	$0-$500
Input Devices	S, VR, 3DM, K	S, J	S, VR, 3DM, K	S	J	J	S, VR, 3DM, K
Coverage	iOS	iOS	iOS	iOS	–	–	iOS, Android
Bimanual	✗	✗	✓	✗	✓	✓	✓
Simulators	MuJoCo	MuJoCo	MuJoCo	MuJoCo	–	–	MuJoCo & Isaac Lab
Training Curriculum	✗	✗	✗	✗	✗	✗	✓
Sim/Real	Sim	Sim	Sim & Real	Real	Real	Real	Sim & Real
Remote	✓	✓	✓	✓	✗	✗	✓
Publicly Available	✗	✗	✓*	✗	✓	✓	✓
Cloud-Scaling	✓	✗	✗	✗	✗	✗	✓
Users Per Machine	1	1	1	1	–	–	8+**

via single smartphones (Android/iOS), dual smartphones for bimanual control, VR headsets, 3D mice, and keyboards. We achieve low-latency (sub-100 ms at 20 Hz) interaction through optimized networking, caching, and multiprocessing.

Recognizing that scale without quality is insufficient, COBALT incorporates a suite of real-time performance metrics to automatically filter suboptimal demonstrations and a structured training curriculum proven to improve user proficiency and data quality. Our contributions are summarized below:

1) **COBALT**: An open-source, cloud-based teleoperation platform designed for scalability and accessibility, supporting 190+ environments. COBALT accommodates several concurrent users on a *single GPU* and integrates several commonly available input devices, including smartphones, thus lowering the barrier to entry.
2) **Data Quality at Scale**: We introduce a structured training curriculum and an extensive suite of performance metrics that help refine data quality and lay the groundwork for future systems capable of autonomous user onboarding and data curation at scale.
3) **User Study and Analysis**: We perform a comprehensive user study comparing input devices for teleoperation, providing insights into device ergonomics and performance, alongside stress tests quantifying the platform's scalability.
4) **Pilot Dataset**: We crowdsource a pilot dataset (7500+ human-collected demos, 50+ hours) using COBALT from 50+ *inexperienced* teleoperators across *nine* countries over *five* days and evaluate its quality by training imitation learning policies.

II. RELATED WORK

A. Teleoperation Frameworks

Large-scale data collection requires robust infrastructure that supports low-latency streaming, offers accessible input modalities, and enables distributed deployment. RoboTurk [16] introduced a server-client architecture that shifted simulation computation to remote servers. This design allowed participants to control robots through smartphone and VR interfaces with minimal local hardware requirements, improving scalability and enabling crowdsourcing. Subsequent works, such as TeleMoMa [15] and MoMaRT [22], extended these contributions to mobile manipulators, enabling more complex tasks and control strategies. However, these platforms generally lack comprehensive user testing, quantitative evaluation of collected trajectories, and a demonstration of the ability to scale crowdsourced data collection.

COBALT differentiates itself from these works in its ability to arbitrarily scale geographically distributed data collection with increasing compute. Earlier efforts collected datasets from a handful of users and tasks [15], [16], enabling users to gain strong proficiency and provide near-expert demonstrations over time. We demonstrate true global crowdsourced teleoperation, yielding over 50 hours of successful demonstrations across 10+ environments, multiple simulators, and 50+ inexperienced teleoperators. In contrast to prior platforms [15], [16], [22], COBALT supports concurrent, uninterrupted teleoperation across vectorized environments on a single GPU, significantly improving scalability, efficiency, and cost-effectiveness. To lower the barrier for community adoption and extension, COBALT will be released as a fully open-source teleoperation platform supporting accessible, low-cost devices like smartphones, including both Android and iOS. With around 80% of the global market share [25], providing Android support is particularly crucial in enabling global crowdsourcing. COBALT overcomes significant limitations of previous systems that were either closed-source [16], [22], [23], incompletely released [15], or required costly specialized hardware [6], [24].

B. Input Devices

Scalable teleoperation depends on selecting input devices that strike a balance between cost, availability, user comfort, and precision. Specialized options, such as leader-follower setups [6] or VR headsets [26], [27], can provide highly accurate and intuitive control, but often require equipment that is neither widely accessible nor fatigue-free over extended sessions. Similarly, 3D mice have been refined for teleoperation using techniques like deadbands and low-pass filtering [28], but remain limited in their adoption due to cost and niche usage. By contrast, most modern smartphones include built-in augmented reality (AR) frameworks capable of tracking 6-DoF poses with competitive accuracy [29]. This has enabled platforms such as RoboTurk [16] and MoMaRT [22] to successfully gather large volumes of manipulation data. Furthermore, independent work on user interface improvements, such as explicit input assistance [30] and automated grasp planning [31], further simplify the data collection process by reducing fatigue and user error. Overall, the broad availability of smartphones presents

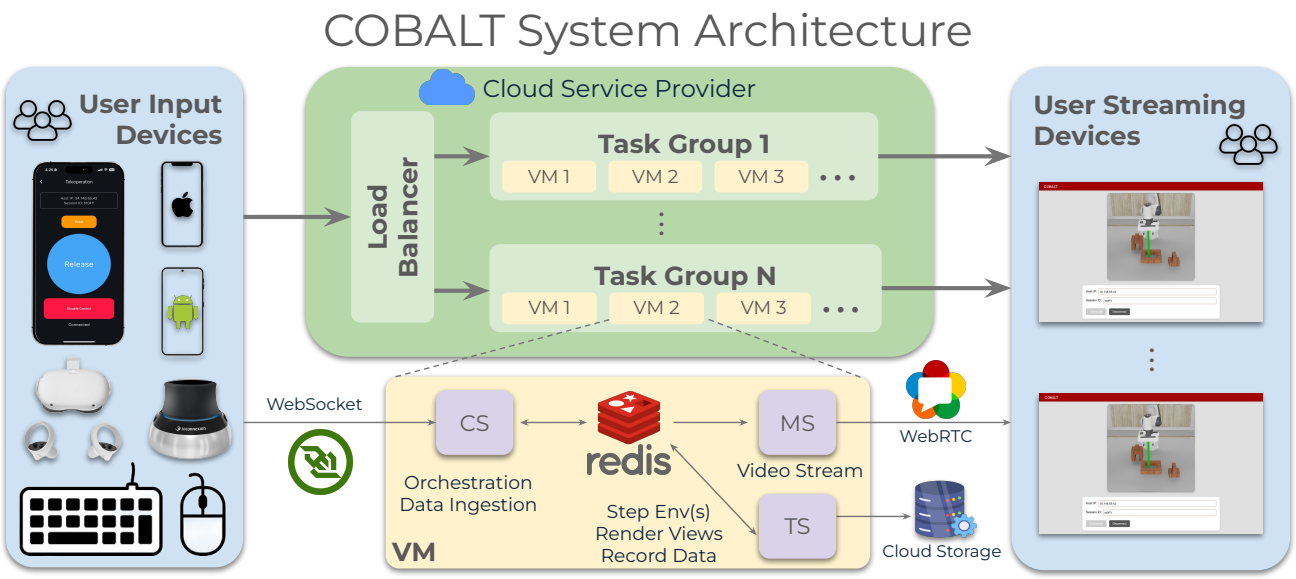

Fig. 2: COBALT System Architecture. a) Cloud provider hosts one group of virtual machines (VM) per task, with dynamic allocation of servers based on demand. b) A load balancer sits in front of the different groups of servers, functioning as a rate limiter and reverse proxy. c) Three main services are utilized: CS (Client Session Service) for client data ingestion, MS (Media Service) for video streaming, and TS (Teleoperation Service) for the vectorized simulation backend. d) Data pipeline handles data storage, cleaning scripts, and augmentation.

an ideal pathway to crowdsource demonstrations at scale, especially for tasks requiring full pose control, without specialized hardware.

III. COBALT: DESIGN AND ARCHITECTURE

COBALT is a scalable, cloud-based data collection platform that enables users worldwide to remotely teleoperate simulated and real robots. By leveraging low-latency networking, diverse input devices, multiple simulation frameworks, and real-world teleoperation capabilities, COBALT enables large-scale crowdsourcing and democratizes the creation of high-quality robotics datasets.

COBALT improves scalability and accessibility, as compared to prior work in RoboTurk [16], through the integration of robust cloud infrastructure and support for a variety of input devices. The platform accommodates smartphones (both Android and iOS), virtual reality (VR) headsets, keyboards, and 3D mice. For bimanual tasks, COBALT also supports VR headsets and dual smartphones. By offering an extensible control interface, COBALT allows developers to integrate arbitrary input devices. Additionally, the platform incorporates a structured training curriculum to onboard users effectively, ensuring the collection of high-quality demonstration data.

A. *System Architecture*

COBALT's architecture is designed to be intuitive and modular. Users connect to a server using their input device, and the server streams the simulated environment to a web browser or VR headset. Building such a system to scale, however, requires nuanced design choices. Namely: (1) users should be able to connect to and exit the system at any point in time, essentially as a *distributed on-demand* service, (2) the platform must support concurrent users on a single simulation instance to optimize compute-cost trade-offs with maximal resource utilization, and (3) the system must support teleoperation with low latency so that users have a smooth and comfortable experience.

COBALT meets these requirements with a modular architecture to facilitate ease of development, use, and extensibility. COBALT consists of three primary components: a Client Session (CS) Service to communicate with client input devices, a Teleoperation Service (TS) to run a simulation backend capable of handling multiple users, and a Media Service (MS) to stream visual feedback with low latency (Figure 2).

Centralized Communication via Redis – To handle communication among all three services within the platform, we utilize Redis, an in-memory database with low-latency read and write operations that can be deployed in a distributed fashion across several machines. This database decouples the main services, allowing them to exchange state information such as user commands and rendered video frames asynchronously and efficiently.

User Connection and Input Handling – Users connect to the platform via the Client Session Service using a WebSocket connection from their chosen input device. This service manages user authentication and session lifecycles and acts as the primary ingestion point for user control data. It receives raw pose information from the client device at 20 Hz and continuously publishes these 6-DoF pose commands to the Redis store, tagged by user session. This approach allows users to join and leave seamlessly on-demand. Redis enables the Client Session Service to asynchronously communicate this information to downstream services, maintaining performance and consistency.

Vectorized Simulation Core – The Teleoperation Service is our system's computational engine. COBALT leverages *vectorized simulation environments*, allowing the service to

manage and step multiple independent simulation instances *concurrently* on a *single* GPU. The service orchestrates the assignment of available simulation environments to clients upon new connections. For each active user session, the service subscribes to the session's pose commands in the Redis store, pulls the latest pose update, performs necessary coordinate transformations, and dispatches the action to the corresponding simulation environment. Internally, the Teleoperation Service applies the standardized 6-DoF commands to the specific robot model within the chosen simulator backend. After stepping the simulation, it renders the visual output for each environment, encodes it with H.264, and publishes the encoded frames to a fixed-size buffer stored in Redis. In parallel, this service also logs all pertinent demonstration data (states, actions, timestamps, metrics) for offline use.

Low-Latency Visual Feedback – To provide users with real-time visual feedback, the Media Service subscribes to the encoded video frames published to Redis by the Teleoperation Service. It utilizes WebRTC to establish a direct, low-latency peer-to-peer streaming connection with the user's display client (e.g., a web browser or VR headset). This minimizes the delay between a user's action and the visual result, which is crucial for real-time teleoperation.

Scalable Cloud Deployment – The entire architecture is designed for robust deployment on cloud platforms. We containerize and deploy the system across auto-scaling VM instance groups, segregated by task type and/or geographic region. A central load balancer distributes incoming user connections, thus enabling high availability and responsiveness. This infrastructure allows our platform to dynamically scale compute resources based on demand, supporting in principle an arbitrary number of concurrent users globally while keeping operational costs low.

B. *Training Curriculum*

To ensure users are prepared for teleoperating simulated robots using COBALT, we developed a training curriculum consisting of calibration and evaluation tasks. This curriculum is designed to onboard users to ensure they can collect high-quality demonstration data.

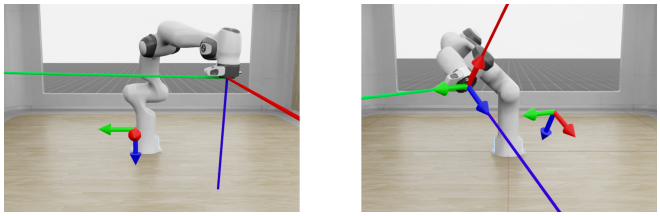

Fig. 3: Subset of Calibration Tasks. Left: Position Task (translational motion only). Right: Pose Task (translation and rotational motion).

Calibration – Calibration tasks are designed to familiarize users with basic controls. *Position calibration* asks users to place the gripper at randomly spawned targets; *rotation calibration* aligns an attached beam to a target circle; and *pose calibration* combines both position and orientation targets.

Evaluation – Evaluation tasks build on calibration tasks by introducing accuracy and precision measurements along with time constraints. *Position (accuracy)* tasks require reaching position targets under time limits; *rotation (accuracy)* tasks align the beam to disappearing targets; *pose (accuracy)* tasks demand full 6-DoF alignment under time pressure; and *beam (precision)* tasks trace line trajectories of decreasing thickness.

Performance metrics from these tasks inform user proficiency and data quality. Analysis can also reveal which aspects of each input modality are most responsible for errors.

C. *Performance Metrics*

We developed multiple metrics to quantify the quality of demonstrations, helping to evaluate the efficacy of input devices, training curricula, ergonomics, and data utility:

Task Completion Time reflects the amount of time taken to complete a task successfully. Shorter times generally indicate higher efficiency.

Network Latency conveys information that helps to understand the implications of network delays on teleoperation. It is measured by calculating the time delay from when a message is sent from the client to when it is received by the server, synchronized with a server clock.

Trajectory Path Length measures the total translational and rotational distance traveled during a demonstration. We define two path length metrics:

a) Total Translational Distance: Let $\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_T\}$ be a sequence of end-effector positions in $\mathbb{R}^3$ measured over the course of a teleoperated episode. The *total translational distance* is the sum of instantaneous translational velocities:

$$D_{\text{trans}} = \sum_{t=0}^{T-1} \|\mathbf{p}_{t+1} - \mathbf{p}_t\|_2.$$

A larger value indicates the end-effector traveled a greater distance during teleoperation, suggesting less efficient motion.

b) Total Rotational Distance: Let $\{\mathbf{R}_0, \mathbf{R}_1, \ldots, \mathbf{R}_T\}$ be a sequence of orientation matrices in $\mathrm{SO}(3)$. For each consecutive pair $(\mathbf{R}_t, \mathbf{R}_{t+1})$, define the relative rotation matrix and the angle of rotation:

$$\mathbf{R}_{\text{rel}} = \mathbf{R}_t^{\mathsf{T}} \mathbf{R}_{t+1}, \quad \theta_t = \arccos\left(\frac{\text{trace}(\mathbf{R}_{\text{rel}}) - 1}{2}\right).$$

The *total rotational distance* is the sum of these incremental angles: $D_{\text{rot}} = \sum_{t=0}^{T-1} \theta_t$. A higher total rotation implies more rotational movement throughout the task execution.

Motion Jitter captures the smoothness or abruptness of motion, as defined by local accelerations:

a) Mean Translational Jitter: Define a sequence of translational positions $\{\mathbf{p}_t\}$. Over a sliding window of size L, we compute maximal local translational accelerations and average them to obtain the *mean translational jitter*:

$$J_{\text{trans}} = \frac{1}{N_L} \sum_{w=1}^{N_L} \left(\max_{t \in [w, w+L-2]} a_t \right), \quad a_t = \frac{v_{t+1} - v_t}{\Delta t'_t},$$

where $v_t = (\mathbf{p}_{t+1} - \mathbf{p}_t)/\Delta t_t$ is the discrete velocity, Δt_t is the time interval between timestamp t and $t+1$, $\Delta t'_t$ is the time difference relevant for the velocity interval (e.g., $(\Delta t_t + \Delta t_{t+1})/2$), and N_L is the number of windows. Larger values indicate more abrupt changes in translational speed.

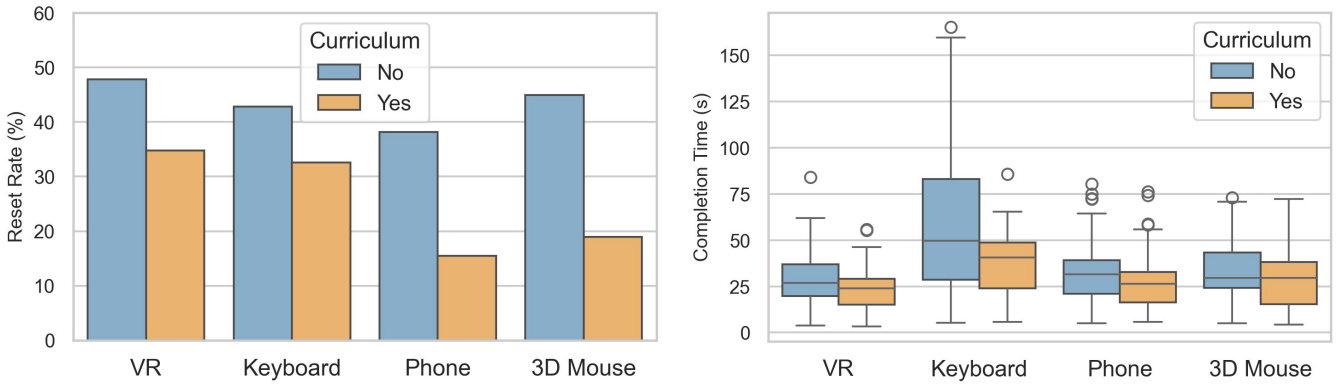

Fig. 4: (a) Reset Rate by Device and Curriculum. Across all devices, curriculum training yields a significant decrease in reset rate across tasks, leading to faster and more efficient data collection. (b) Execution Time by Device and Curriculum. Across all devices, curriculum training reduces the mean and standard deviation of execution time, leading to shorter and more consistent demonstrations.

b) Mean Rotational Jitter: Similarly, for orientations $\{\mathbf{R}_t\}$, let θ_t be the incremental rotation angle between $\mathbf{R}_t$ and $\mathbf{R}_{t+1}$. Over a sliding window of size L, we compute maximal local angular accelerations and average them:

$$J_{\text{rot}} = \frac{1}{N_L}\sum_{w=1}^{N_L}\left(\max_{t\in[w,w+L-2]}\alpha_t\right), \quad \alpha_t = \frac{\omega_{t+1}-\omega_t}{\Delta t'_t},$$

where $\omega_t = \theta_t/\Delta t_t$. Larger rotational jitter values indicate more sudden changes in orientation speed.

Communication Loop Jitter measures simulation and client device stability. We propose two metrics:

a) Server Loop Jitter: During teleoperation, the server processes incoming teleoperation commands at discrete timestamps, which we denote $\{t_0, t_1, \ldots, t_T\}$. The *server loop jitter* measures the variability in these intervals:

$$J_{\text{server}} = \text{Std}\left(\{t_{t+1}-t_t\}_{t=0}^{T-1}\right).$$

where Std denotes the standard deviation. Lower jitter indicates more consistent server-side loop timing.

b) Client Loop Jitter: On the client side (e.g., user device), we track similarly spaced timestamps $\{\tau_0, \tau_1, \ldots, \tau_T\}$ at which inputs are sent. The *client loop jitter* is:

$$J_{\text{client}} = \text{Std}\left(\{\tau_{t+1}-\tau_t\}_{t=0}^{T-1}\right).$$

Higher client loop jitter suggests variable intervals between sent commands, possibly reflecting inconsistent local processing or network delays affecting the sending rate.

IV. Experiments

We conducted a series of experiments to systematically evaluate COBALT across multiple dimensions, structured around the following research questions:

RQ1: How does the choice of input device (smartphones, VR, 3D mice, keyboards) affect teleoperation performance, user experience, and data quality?

RQ2: Does a structured training curriculum improve user proficiency and the quality of collected demonstrations?

RQ3: Can COBALT scale to support numerous concurrent users while maintaining low end-to-end latency and high simulation control frequency?

RQ4: What is the cost-efficiency of demonstration collection using COBALT?

RQ5: Is the data from COBALT useful for training performant behavior cloning policies in simulation?

RQ6: Is the data from COBALT useful for training performant behavior cloning policies in the real world?

A. *RQ1 & RQ2: Input Device Comparison and Curriculum Effectiveness*

To address RQ1 and RQ2, we recruited 12 participants for an initial user study. All participants provided written informed consent prior to participation. Six participants were randomly assigned to first complete the training curriculum described in Section III-B, while the other six served as a control group (no prior training). Each participant used two randomly assigned input devices (chosen from smartphone, VR headset, 3D mouse, keyboard) to perform a set of calibration and manipulation tasks. Participants were instructed to provide five successful demonstrations (per assigned input device) for each of the four manipulation tasks: Three-Piece Assembly, Lift, Mug Cleanup, and Coffee (see [17] for task details). We collected a set of metrics that include task completion time, total path length, translational and rotational jitter, and task reset rates (Section III-C). Subjective feedback was collected using NASA-TLX surveys and Likert scale questionnaires focusing on ease of use, comfort, and perceived accuracy. A separate study with six additional participants compared dual-smartphone versus VR control for bimanual tasks.

(RQ1: Device Comparison) Performance varied significantly across devices (Table II). Smartphones and VR headsets generally yielded better objective metrics, including shorter completion times and smoother trajectories (lower jitter) when compared with keyboards and 3D mice. Although keyboards and 3D mice yielded shorter absolute path lengths, this stems from the way their inputs are received. VR controllers and phones transmit continuous pose updates, where even small movements accumulate into longer trajectories, whereas 3D mice support direct velocity control and keyboards restrict control to discrete steps. Nonetheless, the pose evaluation

(a) (b)

Fig. 5: (a) Visualization of Isaac Lab tasks in the pilot dataset. Arrangement of tasks left-to-right, top-to-bottom: Assembly, Lift, Cleanup, Kitchen, Stack, Pour. (b) COBALT can be used to control physical (single-arm and bimanual) robots. A real-world recreation of the pour task and a corn cooking task are shown.

Metric		Smartphone	VR Headset	3D Mouse	Keyboard
Avg. Completion Time (s)	(↓)	30.00±16.97	**25.60**±13.91	31.14±17.12	46.49±32.74
Avg. Translational Path Length	(↓)	2.47±1.57	2.30±1.29	**1.95**±1.02	2.00±1.09
Avg. Rotational Path Length	(↓)	4.23±2.74	4.18±2.91	**1.63**±1.17	2.03±1.48
Avg. Translational Jitter	(↓)	**0.24**±0.09	0.43±0.18	0.35±0.18	0.65±0.26
Avg. Rotational Jitter	(↓)	**0.37**±0.14	0.81±0.39	0.44±0.21	0.64±0.21
Reset Rate (%)	(↓)	**28.57**	42.03	34.43	38.14
Willing to Use Again (1-5)	(↑)	**4.33**±0.52	4.17±1.17	3.83±1.33	3.17±1.47
Subjective Comfort (1-5)	(↑)	**4.50**±0.55	4.33±1.03	3.67±1.21	3.33±1.03

TABLE II: Summary of Key Performance Metrics Across Input Devices in User Study (Mean ± Std. Dev.). We observe that the use of COBALT with a smartphone improves performance as measured by both quantitative and qualitative metrics. Note that path length and jitter are unitless.

TABLE III: Pose Evaluation Task Average Positional and Rotational Error Across Input Devices. Smartphones yielded significantly lower errors than the other input modalities.

Device	Position Error	Rotation Error
Phone	**0.13**±0.06	**0.29**±0.17
VR Headset	0.20±0.20	0.51±0.39
3D Mouse	0.25±0.13	1.36±0.53
Keyboard	0.16±0.06	0.87±0.28

task (Table III) showed that smartphones achieved the lowest position and rotation errors, demonstrating their effectiveness for capturing high-quality, precise demonstrations. Subjective feedback corroborates these results, with users rating smartphones and VR higher on willingness to use again and subjective comfort. Note that these tasks were conducted in MuJoCo-based environments, so the translational and rotational metrics in Table II and Table III are unitless and should be interpreted only in relative terms.

For bimanual tasks, dual-smartphone control was identified as a low-cost alternative to VR demonstrations. Models were trained on a small dataset (60 demonstrations) of dual-smartphone and VR data. With BC-RNN and BC-Transformer, we obtained success rates of 22% and 26%, respectively. Success rates were calculated based on 100 rollouts. These results validate the use of COBALT for bimanual teleoperation.

(RQ2: Curriculum Effectiveness) The training curriculum had a demonstrably positive impact. Participants who underwent training exhibited significantly lower reset rates (Figure 4a) and reduced mean execution times across all devices (Figure 4b). This suggests the curriculum effectively onboarded users, improving data collection speed and quality by reducing errors and increasing device familiarity.

To assess statistical significance, we compared groups with and without the training curriculum using independent tests. For completion time, a one-sided t-test tested whether participants who received the curriculum executed downstream tasks in a shorter amount of time. The result yielded a test statistic of -4.26 and a p-value less than 0.0001, indicating statistical significance. For reset rate, we conducted a two-sample proportion test to determine whether the training curriculum reduced the frequency of resets. This test produced a statistic of -4.88 and a p-value of 1×10^{-6}, also significant. Together, these results demonstrate that a structured training curriculum substantially improves both the efficiency and reliability of trajectories collected by novice teleoperators.

B. *RQ3 & RQ4: Scalability, Latency, and Cost Efficiency*

Guided by the findings that smartphones and VR offer superior performance and user experience, we evaluated the scalability of COBALT for crowdsourced data collection.

We deployed COBALT on Google Cloud Platform (GCP) following the architecture in Figure 2. We varied the number of concurrent teleoperators connecting to a single GPU instance running vectorized Isaac Lab environments from one user up to eight to observe scaling effects. We measured:

a) Average latency: Time from user input action to client session service receiving the action.

b) Simulation control loop time: Time taken by the simulation environment to process user commands and update its state.

c) Resource utilization: RAM and VRAM usage.

d) Cost per 1,000 demonstrations: Estimated cost for 1,000 demonstrations based on cloud pricing.

(RQ3: Scalability & Performance) Our architecture demonstrated effective scaling, sustaining multiple concurrent users while maintaining interactive performance. Although teleoperation remained feasible when scaled to eight concurrent users, we decided to limit active sessions on one GPU to four to ensure the best user experience (Table IV). Memory utilization marginally increased when scaling users, demonstrating that CPU-bound processes such as WebSocket communication and WebRTC streaming impose minimal overhead on overall system performance. Additionally, we conducted system load

# Users	Avg. Latency (ms)	Med. Sim Step (ms)	Peak VRAM (GB)	Peak RAM (GB)
1	1.70 ± 4.95	45.68 ± 0.09	3.40 ± 0.00	5.17 ± 0.00
2	6.16 ± 4.97	51.08 ± 0.02	3.41 ± 0.00	5.37 ± 0.00
4	4.79 ± 4.82	60.06 ± 0.08	3.55 ± 0.00	5.70 ± 0.06
8	7.08 ± 4.93	79.31 ± 0.04	3.88 ± 0.00	6.66 ± 0.02

TABLE IV: System Scaling. System Performance vs. Number of Concurrent Users per GPU (NVIDIA T4). As the number of concurrent clients on a single GPU increases, latency and memory utilization grow sublinearly.

TABLE V: COBALT Pilot Dataset Statistics

Task	Demonstrations	Hours
Lift	1,294	1.99
Pour	1,026	4.92
Stack	1,112	5.77
Cleanup	1,023	6.86
Assembly	1,007	7.14
Kitchen	1,284	17.16
User Study	764	6.77
Total	7,510	50.61

testing by simulating *256 concurrent clients* distributed over *8 GPUs*. COBALT sustained this workload in a distributed fashion with a median simulation-step latency of 186.7 ms. Importantly, our results show that this latency does not scale linearly with client count, indicating that more powerful GPUs may deliver *superlinear scaling* improvements.

(RQ4: Cost Efficiency) Cloud deployment offers flexibility and scalability, but without effective system resource utilization, scaling can be expensive. By serving multiple clients on a single GPU, COBALT substantially cuts these costs. On an NVIDIA T4 instance, priced at $0.92 per hour (estimated GCP cost), assuming each user completes 120 demonstrations in that hour, the cost for 1,000 demonstrations in a non-vectorized setting would be $7.67. With COBALT, this cost drops to just $1.92. This equates to a near *4x reduction* in data collection expenses on entry-level hardware. COBALT can flexibly adjust concurrency to achieve an optimal cost–performance balance, scaling to support 12 concurrent users on a high-end GPU like the NVIDIA RTX 6000.

C. *RQ5: Data Validation via Behavior Cloning*

Finally, to demonstrate the practicality of crowdsourced data, we collected a large pilot dataset using COBALT, leveraging insights from our initial studies. We then used our pilot dataset to train several imitation learning policies.

We crowdsourced the collection of 6,746 human demonstrations using only smartphones across several benchmark tasks in Isaac Lab (Table V, Figure 5a). These tasks included four Isaac Lab environments created from scratch and two modified stock environments. Data quality was maintained by filtering based on performance metrics (Section III-C), selecting demonstrations within the 50th percentile for total path length to remove suboptimal trajectories. We trained standard behavior cloning algorithms (BC-RNN, BC-Transformer [32]) and state-of-the-art methods like Action Chunking with Transformers (ACT) [6] and Diffusion Policy (DP) [1] on this curated dataset. Policy performance was evaluated based on task success rates over 50 rollouts per task.

The policies trained on the COBALT-collected data achieved high success rates on the majority of tasks from our task suite (Table VI), confirming the effectiveness of metric-based filtering and overall quality of crowdsourced demonstrations. Note that BC-RNN and BC-TF were trained on additional low-dimensional observations, specifically task-relevant object poses. Capturing additional data or observations (such as another camera view) may improve these results. Nonetheless, these results confirm that COBALT can produce datasets effective for training robot manipulation policies, and that data collected via accessible devices like smartphones captures sufficient fidelity and diversity.

TABLE VI: BC Results Per Task. Data collected with COBALT is capable of achieving a variety of tasks using SOTA algorithms.

Task	BC-RNN	BC-TF	ACT	DP
Lift	**1.00**	0.84	0.88	**1.00**
Pour	**0.68**	0.36	0.36	0.54
Stack	0.00	0.00	**0.60**	0.58
Cleanup	0.72	0.20	0.92	**0.94**
Assembly	0.36	0.10	0.32	**0.50**
Kitchen	0.04	0.02	0.10	**0.12**

D. *RQ6: Real-Robot Compatibility*

In addition to our simulation environments, we validated our smartphone teleoperation pipeline on multiple platforms, including a Franka Panda arm and bimanual YAM arms (Figure 5b). We configured our system to connect the mobile app directly to a server running on the robot's host machine over our local network, enabling low-latency, real-time control. To ensure safe deployment on hardware, we incorporate safety mechanisms into the teleoperation pipeline. For example, the smartphone automatically disables robot control if its velocity exceeds a threshold, preventing accidental motions (e.g., if the user drops the phone). We validated our Franka setup by collecting 98 expert demonstrations on the standard Lift task and then training a BC-RNN policy on this data, achieving a 52% success rate over 25 rollouts. Although lower than the simulation results, this confirms the ability to use COBALT with real hardware. We expect performance to improve with more real-world data collection on COBALT, which we leave to future work. Nonetheless, these results confirm that our phone-based interface generalizes effectively from simulation to real-world hardware with minimal setup.

V. CONCLUSION

By lowering the barrier to entry for remote teleoperation, COBALT aims to democratize large-scale dataset creation for imitation learning. Through a comprehensive user study, we demonstrate that integrating accessible devices, user-friendly interfaces, and robust networking infrastructure can significantly improve the quality and efficiency of data

collection. We also establish core metrics to rank operators, assess server performance, evaluate a device's effectiveness in producing demonstrations, and identify the most reliable and high-quality demos. Our infrastructure enabled the crowdsourcing of a high-quality pilot dataset with over 7,500 trajectories, which we used to successfully train imitation learning models. While platforms like COBALT address the critical bottleneck of operator availability and data volume, they highlight the emergence of a new potential bottleneck: *the creation of diverse, high-quality simulation environments*. As collecting demonstrations at scale becomes easier, achieving task diversity and reliable generalization will require a parallel effort in scaling task design. Ultimately, future progress in robot learning will depend on both accessible data collection platforms like COBALT as well as consistent scaling of novel, complex environments.

ACKNOWLEDGMENTS

We thank all of our user study participants for providing valuable data and feedback to improve our platform.

REFERENCES

[1] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *arXiv preprint arXiv:2303.04137*, 2023.

[2] K. Fang, Y. Zhu, A. Garg, A. Kuryenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *Robotics: Science and Systems (RSS)*, 2018.

[3] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox, "Rvt: Robotic view transformer for 3d object manipulation," in *Conference on Robot Learning*. PMLR, 2023.

[4] S. Lee, Y. Wang, H. Etukuru, H. J. Kim, N. M. M. Shafiullah, and L. Pinto, "Behavior generation with latent actions," *arXiv preprint arXiv:2403.03181*, 2024.

[5] A. Mete, H. Xue, A. Wilcox, Y. Chen, and A. Garg, "Quest: Self-supervised skill abstractions for learning continuous control," *arXiv preprint arXiv:2407.15840*, 2024.

[6] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," 2023. [Online]. Available: https://arxiv.org/abs/2304.13705

[7] C. Crawl, "Common crawl dataset," 2008. [Online]. Available: https://registry.opendata.aws/commoncrawl/

[8] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, *et al.*, "Laion-5b: An open large-scale dataset for training next generation image-text models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 25 278–25 294, 2022.

[9] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[10] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," *arXiv preprint arXiv:2403.12945*, 2024.

[11] Q. Vuong, S. Levine, H. R. Walke, K. Pertsch, A. Singh, R. Doshi, C. Xu, J. Luo, L. Tan, D. Shah, *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," in *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*, 2023.

[12] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, *et al.*, "Bridgedata v2: A dataset for robot learning at scale," in *Conference on Robot Learning*. PMLR, 2023.

[13] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[14] NVIDIA, "Nvidia isaac sim," 2022.

[15] S. Dass, W. Ai, Y. Jiang, S. Singh, J. Hu, R. Zhang, P. Stone, B. Abbatematteo, and R. Martín-Martín, "Telemoma: A modular and versatile teleoperation system for mobile manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2403.07869

[16] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," 2018. [Online]. Available: https://arxiv.org/abs/1811.02790

[17] 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," 2023. [Online]. Available: https://arxiv.org/abs/2310.17596

[18] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu, "Robocasa: Large-scale simulation of everyday tasks for generalist robots," 2024. [Online]. Available: https://arxiv.org/abs/2406.02523

[19] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, p. 3740–3747, June 2023. [Online]. Available: http://dx.doi.org/10.1109/LRA.2023.3270034

[20] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," 2022. [Online]. Available: https://arxiv.org/abs/2009.12293

[21] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: Benchmarking knowledge transfer for lifelong robot learning," *arXiv preprint arXiv:2306.03310*, 2023.

[22] J. Wong, A. Tung, A. Kurenkov, A. Mandlekar, L. Fei-Fei, S. Savarese, and R. Martín-Martín, "Error-aware imitation learning from teleoperation data for mobile manipulation," 2021. [Online]. Available: https://arxiv.org/abs/2112.05251

[23] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," 2019. [Online]. Available: https://arxiv.org/abs/1911.04052

[24] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, "Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators," 2024. [Online]. Available: https://arxiv.org/abs/2309.13037

[25] A. Nicolle, "Threats, bans, and competition: Ripple effects in the global smartphone market," Nov. 2024. [Online]. Available: https://ssrn.com/abstract=5038275

[26] S. Chen, C. Wang, K. Nguyen, L. Fei-Fei, and C. K. Liu, "Arcap: Collecting high-quality human demonstrations for robot learning with augmented reality feedback," *arXiv preprint arXiv:2410.08464*, 2024.

[27] Y. Park, J. S. Bhatia, L. Ankile, and P. Agrawal, "Dexhub and dart: Towards internet scale robot data collection," *arXiv preprint arXiv:2411.02214*, 2024.

[28] V. Dhat, N. Walker, and M. Cakmak, "Using 3d mice to control robot manipulators," in *2024 19th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2024, pp. 896–900.

[29] J. Kim, M. Song, Y. Lee, M. Jung, and P. Kim, "An empirical evaluation of four off-the-shelf proprietary visual-inertial odometry systems," 2022. [Online]. Available: https://arxiv.org/abs/2207.06780

[30] N. Walker, X. Yang, A. Garg, M. Cakmak, D. Fox, and C. Pérez-D'Arpino, "Fast explicit-input assistance for teleoperation in clutter," 2024. [Online]. Available: https://arxiv.org/abs/2402.02612

[31] T. Wu, M. Wu, J. Zhang, Y. Gan, and H. Dong, "Graspgf: Learning score-based grasping primitive for human-assisting dexterous grasping," 2023. [Online]. Available: https://arxiv.org/abs/2309.06038

[32] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," 2021. [Online]. Available: https://arxiv.org/abs/2108.03298