Maintaining Demonstration Quality in a 100-Robot Teleoperation Pipeline

Kunal Kapoor, Shiraz Khan, Joseph McCalmon, Jesse Michel, Arif Mohammed,
Katerina Nikiforova, Adam Oppenheimer, Victor Szabo, David Watkins
Tutor Intelligence, Watertown, MA
Email: jesse@tutorintelligence.com

***Abstract*—At fleet scale, maintaining demonstration quality and consistency becomes a first-order engineering problem. When collecting dexterous manipulation data, the sheer number of people defining tasks, teleoperating, and annotating data is a challenge for data quality and consistency. We study these problems in Data Factory 1 (DF1), a 100-robot fleet of bimanual manipulators that supports 1,000 hours of teleoperated demonstrations per day from a large, internationally distributed operator pool.**

Each stage of the pipeline has its own failure modes and opportunities for improvement. (1) During teleoperation, interface latency degrades quality and throughput. We implement PTeleop, a teleoperation system that uses proprioception to improve teleoperation speed by 35.8% and produce over 10% fewer mistakes. (2) To ensure quality at scale, we monitor adherence to the task specification without sacrificing behavioral diversity, using both a human supervisor and an in-house labeling system that scores episodes within minutes of capture. (3) During data preprocessing, operator speed varies widely across local and remote teleoperators, which we address with velocity normalization. (4) We do in-house large-scale data annotation to accurately grade demonstration quality, which enables us to monitor operator performance and provide rapid interventions and to model reward for use in data filtering and correcting task labels. (5) To discover failures at scale we use model rollouts to surface failures and use teleoperator interventions to provide corrections using 1:N robot to human supervision. Taken together, we provide a blueprint for producing high-quality real robot data at scale.

I. A Data Factory

DF1 (Fig. 1), the largest robot data factory in the United States when it was constructed, is a fleet of 100 Sonny bimanual manipulators in Watertown, MA, that runs continuously to collect demonstrations for training vision-language-action (VLA) policies. Demonstrations arrive in two modes, both driven by our international team of remote operators ("Tutors"). In *1:1 control*, one operator demonstrates a task on one robot; in *1:N supervision*, one operator monitors many autonomous rollouts and intervenes. We trained our Ti0 VLA model entirely on DF1 data.

In this abstract, we focus on maintaining quality at scale, and provide a comprehensive technical report as well [11]. We found early on that policy performance improved appreciably when demonstrations were collected consistently across operators. For example, at a few hundred demonstrations, direct communication with a single operator kept collection consistent. However, for 100 continuously operating robots, this mechanism was no longer sufficient. In this work, we treat demonstration quality as an explicit objective at each stage of the pipeline: (1) reducing teleoperation latency with PTeleop, (2) monitoring specification adherence while preserving behavioral diversity, (3) normalizing speed variation during preprocessing, (4) grading demonstrations for reward-weighted training and data filtering, and (5) using model rollouts and teleoperator interventions to surface and correct failures.

Authors are listed in alphabetical order.

Fig. 1. **Data Factory 1 (DF1).** A 100-robot fleet of Sonny bimanual manipulators at Tutor Intelligence HQ in Watertown, MA, collecting up to 1,000 hours of teleoperated demonstrations per day.

II. The Collection Interface

The waiting-to-act problem. In conventional 2D-video teleoperation, operators move, wait for visual confirmation that the robot has reached the expected state, and only then continue. Under even modest latency, the operator pauses frequently to check if the robot is in the expected state, which we call the *waiting-to-act* problem. This is a challenge for both throughput and demonstration quality, as the resulting trajectories are slower and have inconsistent speed, forcing policies to deal with multimodality in whether to move or pause.

Proprioceptive teleoperation. *Proprioceptive teleoperation* is a teleoperation approach (other examples include Goertz [2], Zhao et al. [12], Kent et al. [4]) in which (1) the operator views a real-time 3D representation of the robot's workspace, (2) there is a one-to-one relationship between the distance the operator's hands move and the distance the robot's grippers move, and (3) the operator receives physical

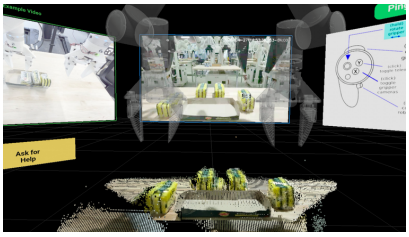

Fig. 2. **The PTeleop interface.** A live workspace reconstruction from multi-view point clouds, with RGBD cameras producing real-time point-cloud state at under 50 ms latency. The robot, objects, and scene geometry share a single 3D frame, enabling continuous manipulation without pausing to confirm robot or scene state.

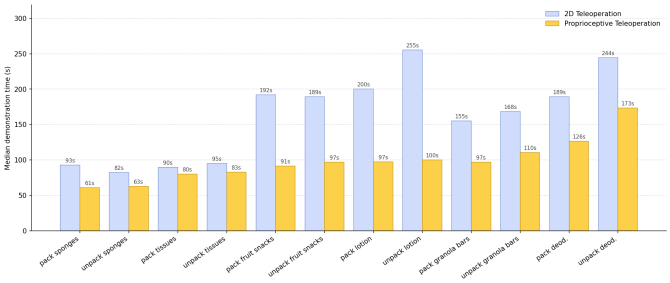

Fig. 3. **PTeleop speeds up demonstrations and reduces mistakes.** Median demonstration time before and after switching from 2D teleoperation to proprioceptive VR teleoperation, across twelve tasks.

feedback (e.g., haptic feedback) from the robot's contact with the scene. PTeleop, our VR implementation of proprioceptive teleoperation, renders a live workspace reconstruction from multi-view point clouds (under 50 ms update time) together with a ghost robot that mirrors the state of the physical system in real time, communicating where the robot will be next, and vibrates the VR controllers to signal successful grasps (Fig. 2). Thus in PTeleop, operators use their proprioception to perform open-space motions based on anticipated end-effector positions. Since motion in open space dominates picking-and-packing time, avoiding unnecessary pauses over large distances is particularly important. In other words, while latency is the same, operators can better use proprioception to perform tasks.

Results. Evaluated across twelve tasks, PTeleop improved teleoperation speed by 35.8% and produced over 10% fewer mistakes in demonstrations relative to 2D teleoperation (Fig. 3). New operators reach our data-quality standards within 6 hours of practice. We monitor median demonstration time and mistake rate per task as standing quality checks. Improvements in both suggest a better overall interface.

III. High-Quality Curation

Monitoring specification adherence at scale. Each task carries an explicit specification (e.g., picking one item at a time, left to right). Curation must flag off-spec demonstrations while preserving the within-spec behavioral diversity that policies need to generalize. We do this continuously, through human supervision and automated labeling, while maintaining diversity by encouraging operators to vary demonstrations and randomizing the scene.

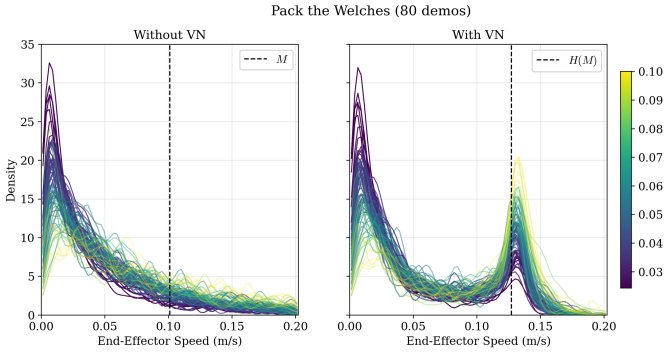

Fig. 4. **Velocity normalization.** Per-episode end-effector speed distributions before and after VN, where each line is one demonstration. The dashed line marks the 80th-percentile speed used as the upper breakpoint.

Episode labeling system. We built an in-house labeling system that scores each episode within minutes of capture, at a throughput of thousands of hours per day. The labels surface mistakes to operators and grade quality for training (Sec. V).

IV. Data Preprocessing

Speed-induced multimodality. Two demonstrations of the same pick-and-place task, one performed at twice the speed of the other, trace nearly identical spatial paths. Yet when sampled at a fixed rate (e.g. 20 FPS) they produce dramatically different frame-to-frame action displacements, forcing the policy to reconcile conflicting action labels for visually similar observations. Speed variation across operators and within a single demonstration can therefore introduce harmful multimodality into the behavior-cloning objective and reduce sample efficiency [9, 10].

Velocity normalization. Velocity normalization addresses both inter- and intra-episode speed variation by changing which frames are selected during downsampling, so that the model perceives a consistent speed distribution. It operates in two stages. Stage 1 aligns inter-episode speed (different demonstrators' overall paces) by rescaling each episode according to its characteristic speed (the 30th percentile of its end-effector speed distribution, with the speedup clamped to $[0.75, 1.5]$). Stage 2 compresses the remaining intra-episode variation (speed changes within a single demonstration) by applying a smooth, monotonic speed mapping that leaves contact-rich low speeds unchanged while compressing the high-speed tail. Frames near gripper open/close events are protected by a 0.5 s window, and idle frames are dropped, except for the final idle segment, which is preserved so that the policy learns to come to rest.

Empirical analysis. Across 30 sampled demonstrations, velocity normalization reduces cross-episode speed variance by 62% (Fig. 4). We monitor the per-episode speed distributions before and after preprocessing.

V. Modeling Demonstration Quality

Mixed-quality data. A trained behavior-cloning policy is only as good as the demonstrations it imitates, and large-scale

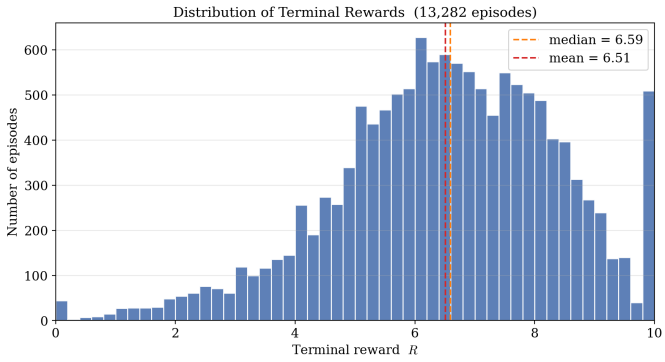

Fig. 5. **Distribution of fused terminal reward** $R \in [0, 10]$ across 13,282 demonstrations. Mixed-quality data has a broad quality distribution; reward-weighting lets us train on it without hard filtering.

collection tends to yield mixed quality, since even episodes a human would call "successful" often contain failed grasp attempts and irrelevant motions. Rather than filtering data to a hard pass/fail, we model demonstration quality as a scalar reward, which converts mixed-quality teleoperation data into a reward-weighted imitation objective [7].

Label types and fusion. Human labelers provide five complementary label types (task-specific multiple-choice questions, pairwise comparisons, timestamped thumbs up/down, an overall quality rating, and subtask-completion markers), which are fused into a single terminal reward $R \in [0, 10]$ per episode and shaped into a per-frame progress reward using subtask segmentation (piecewise progress) and thumbs events (Gaussian bumps) [1, 5, 8, 6]. We use this pipeline to collect 13,282 demonstrations (Fig. 5). We find that high-quality runs yield value-function curves that climb smoothly to a high terminal reward, while low-quality runs show drops at thumbs-down events and plateau below the maximum. The fusion is modular, so a label type's weight can be adjusted as it proves more or less predictive, without changing the rest of the pipeline.

A quality feedback loop. The modeled quality serves a second purpose beyond training. Because each episode carries a graded reward and the labels behind it, systematic quality deficits become visible per operator and per task. Persistent low scores for an operator can prompt targeted coaching, while a behavior that scores poorly across many operators often signals that the task specification or high-level instructions need revision. Quality modeling and collection thus close the loop: the reward we model to weight training also tells us which operators to coach and which task descriptions to make more specific.

Approaches that did not work. A coarse three-category quality label (good/medium/bad) proved redundant with the multiple-choice questions and insufficiently precise, and was removed. Additional label modalities help only when they are non-redundant and predictive enough to justify their operator-time cost. We also examined whether a pretrained reward model, Robometer [5], could reduce the labeling cost. On our data its progress curves were noisy and poorly correlated with actual progress, plateauing after a few pick-places and

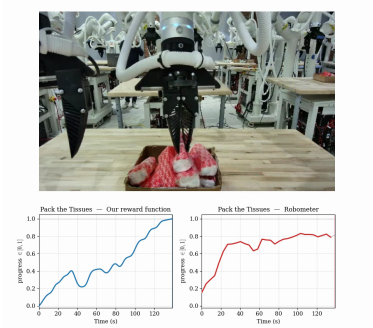

Fig. 6. **In-domain reward models track progress more reliably.** On a successful "pack the tissues" demonstration, our in-domain human-feedback reward (left) rises monotonically to completion, while Robometer [5] (right) plateaus after a few pick-places, reflecting its training-distribution gap.

not reaching completion on successful runs, because our camera viewpoints, object sets, and scene clutter differ from its training distribution. However, fine-tuning the model on our data did not close this gap (Fig. 6). In our setting, dense in-domain labels remained necessary.

VI. Discovering Failures at Scale

Correction data from on-policy rollouts. The end-to-end collection and curation pipeline above is sufficient to train policies that exceed 80% success on long-horizon tasks. In our experience, reaching substantially higher success appears to require expert corrections collected from on-policy rollouts [3]. Fleet scale makes these inexpensive to obtain. An edge case that a single robot might encounter once in eight hours of operation surfaces within about five minutes across a 100-robot fleet, so the benefit of scale is not just more demonstrations but a roughly $100\times$ faster discovery of failures worth correcting. In 1:N supervision, when the policy reaches such a case, flagged automatically or by a human supervisor, an operator assumes control within seconds and demonstrates the recovery. We track time-to-first-occurrence of edge cases to gauge how quickly the fleet surfaces failures.

VII. Discussion

Taken together, these stages suggest that demonstration quality is best treated as an explicit objective at each step of the pipeline, since a deficiency at any single stage limits the quality attainable downstream. Two key factors are proprioceptive teleoperation, which bounds trajectory quality before curation, and low-latency monitoring of task-specification adherence, which is essential for maintaining diversity. Velocity normalization and reward-weighting keep varied-quality data usable for training, while rollout-driven failure discovery and correction surface rare edge cases at fleet scale. Commodity hardware (Meta Quest 3S, Nvidia 5060 Ti) enables the operator and labeling workforce to scale internationally. Near-term work targets dexterity beyond parallel-jaw grasping and lower teleoperation latency.

Acknowledgments

We thank the Tutor Intelligence build, manufacturing, and operations teams in Boston, Mexico City, and Manila, whose demonstrations and labels make this work possible.

REFERENCES

[1] Qiyu Chen, Jialu Yu, Mac Schwager, Pieter Abbeel, Yide Shentu, and Philip Wu. SARM: Stage-aware reward modeling for long horizon robot manipulation. In *ICLR*, 2026. arXiv:2509.25358.

[2] Raymond C. Goertz. Master-slave manipulator. Technical Report ANL-4311, Argonne National Laboratory, 1949.

[3] Michael Kelly, Christian Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *ICRA*, 2019. arXiv:1810.02890.

[4] David Kent, C. Saldanha, and Sonia Chernova. A comparison of remote robot teleoperation interfaces for general object manipulation. In *HRI*, 2017.

[5] Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S. Huang, Luke Zettlemoyer, Dieter Fox, Yu Xiang, Anqi Li, Andreea Bobu, Abhishek Gupta, Stephen Tu, Erdem Biyik, and Jesse Zhang. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons, 2026. arXiv:2603.02115.

[6] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, 1999.

[7] X. B. Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. arXiv:1910.00177.

[8] Physical Intelligence. $\pi^*_{0.6}$: a vla that learns from experience. https://pi.website/blog/pistar06, 2025.

[9] Modi Shi, Li Chen, Jin Chen, Yuxiang Lu, Chiming Liu, Guanghui Ren, Ping Luo, Di Huang, Maoqing Yao, and Hongyang Li. Is diversity all you need for scalable robotic manipulation?, 2025. arXiv:2507.06219.

[10] Jiawei Tang, Yuxuan Sun, Yichen Zhao, Sheng Yang, Yixuan Lin, Zhen Zhang, Jiayi Hou, Yuxiang Lu, Zihan Liu, and Song Han. VLASH: Real-time VLAs via future-state-aware asynchronous inference, 2025. arXiv:2512.01031.

[11] Tutor Intelligence. Data Factory 1 Technical Report. Technical report, Tutor Intelligence, 2026. URL https://cdn.prod.website-files.com/68fa203de60162c631624d70/69f3b223d5ec2d664d4a497c_5441a2fae0791729c87f025e68f26443_tutor_intelligence_data_factory_1_technical_report.pdf.

[12] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *RSS*, 2023. arXiv:2304.13705.