Benchmarking Affordance Generalization with BusyBox

Dean Fortier[*] **Timothy Adamson**[†] **Tess Hellebrekers**[*]

Teresa LaScala[*] **Kofi Ennin**[‡] **Michael Murray**[*]

Andrey Kolobov[*] **Galen Mullins**[*]

Abstract:
Robot Foundation Models (RFMs) have been attracting the attention of researchers and practitioners thanks to their promise of *generalization*. Although single-task policies still offer competitive performance [1], RFMs are increasingly able to handle commands and environments unseen in their training set [2]. While generalization in vision and language space is undoubtedly important for robust versatile behaviors, a key meta-skill RFMs need to possess is *affordance generalization* – the ability to manipulate new objects with familiar physical features.

In this work, we present BusyBox, a physical benchmark for systematic semi-automatic evaluation of RFMs' affordance generalization. BusyBox consists of 6 modules with switches, sliders, wires, buttons, a display, and a dial. The modules can be swapped and rotated to create a multitude of BusyBox variations with different appearances but the same set of affordances. BusyBox's electronics, which can be readily purchased online at a low cost, can detect completion of a task and its substeps. We empirically demonstrate that generalization across BusyBox variants is highly challenging even for state-of-the-art publicly available RFMs such as π_0. To encourage researchers to evaluate their own RFMs on BusyBox and to propose new affordance generalization experiments, we have designed BusyBox to be easy to build in most robotics labs. We release the full set of CAD files for 3D-printing its parts as well as a bill of materials for (optionally) assembling its electronics. We also publish a dataset of language-annotated demonstrations that we collected using the common bimanual Mobile Aloha system [3] on the canonical BusyBox configuration and describe the detailed data collection protocol we followed. All of the released materials are available at URL `https://microsoft.github.io/BusyBox`.

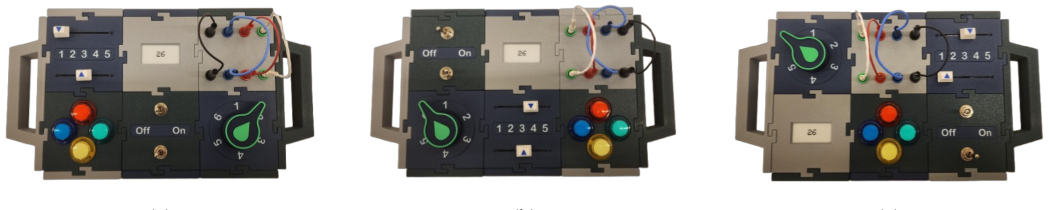
1 Introduction

Robot Foundation Models (RFMs) [4], also known as *Vision-Language-Action (VLA)* models [5], hold the promise of revolutionizing robot control as much as large language models have advanced natural language processing. In particular, they aim to make robot behaviors *general*, i.e., applicable across robot embodiments, tasks, and environments – including those not represented in the training data. Generalization takes many forms, and robotics researchers have created a number of benchmarks to evaluate them [6, 7, 8]. Some types of generalization studied in the context of

[*]Microsoft Research, {`v-defortier,tessh,telascal,michael.murray,akolobov,galenmullins`}`@microsoft.com`

[†]Genie, `timadamson21@yahoo.com`. Work done while at Microsoft Research.

[‡]Mississippi State University, `kofienninacheampong@gmail.com`. Work done while at Microsoft Research.



(a) (b) (c)

Figure 1: 3D-printed BusyBox configurations. Each of them consists of 6 modules: buttons, display, knob, sliders, switches, and wires. (a) is the "canonical" configuration, on which we collected a training set of demonstrations. (b) and (c) are two other configurations we chose arbitrarily for our experiments.

robotics carry over from language and vision, e.g., characterizing an RFM's ability to comprehend new instructions referencing familiar concepts or recognize camera images of known objects in new environments. However, robust physical interaction also requires RFMs to exhibit robotics-specific generalization flavors, such as understanding how an unfamiliar object can be manipulated based on its appearance and the robot's experience of handling other objects in the past. We call this capability *affordance generalization* [9]. Key interface elements in environments designed for people – buttons, switches, etc – are purposefully designed to look and function similarly across different objects in order to facilitate the generalization of basic affordances for the average person. Indeed, people see this capability as natural and *expected*. Accordingly, we posit that robots need to be able to use basic affordances not only to operate fluently in human environments but also to be *perceived* as intelligent and reliable.

This work introduces BusyBox (Figure 1), an open-source physical benchmark for systematically evaluating basic affordance generalization in RFMs. BusyBox is a 3D-printable device consisting of 6 interlocked modules that have buttons of various colors, switches, sliders with marked positions, wires with pluggable connectors, a knob, and a display. These control elements are commonly found on everyday items in home and industrial environments. However, BusyBox itself looks different from any objects likely to be present in RFMs's training data. A key feature of BusyBox is that its 6 modules can be easily swapped and rotated with respect to each other, giving rise to a family of distinct BusyBox instances with the same set of basic affordances. A person seeing a BusyBox instance for the first time can learn all its affordances in under a minute, including pulling out and inserting wires, rotating the knob, and reading information on the display. Having learned them on one BusyBox, the person will be able to use these affordances on any other BusyBox instances zero-shot. How close do RFMs come to this level of affordance generalization?

A second contribution of our work is an experiment protocol and a dataset of demonstrations (see Figure 2) for empirically answering questions like this. We collected 1042 trajectories across 7 task families representing BusyBox's affordances, e.g., *"rotate the knob to position 4"*, by teleoperating a Mobile Aloha system [3] on a designated "canonical" BusyBox instance (Figure 1a). Next, we used this dataset to finetune π_0 [10], widely considered to be state-of-the-art among open-weights RFMs, and evaluated this adapted model on the same tasks on three BusyBox configurations in Figure 1– the canonical one, on which we had collected the

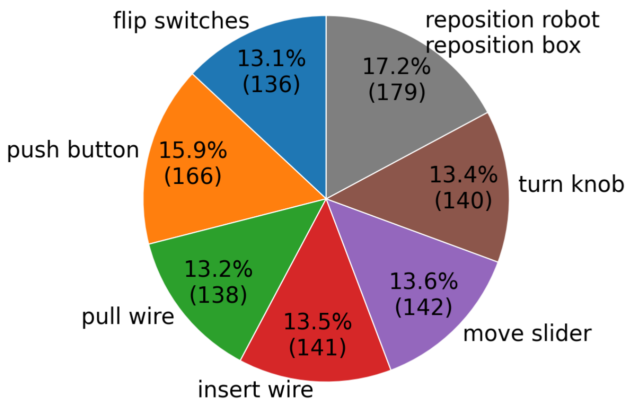


Figure 2: Breakdown of our dataset of 1042 BusyBox demonstrations by affordance category.

finetuning dataset, and two others. Our results show that the finetuned π_0 is far from perfect on the canonical BusyBox configuration, and fails completely on the other two. As this outcome demonstrates, basic affordance learning and generalization is a major area for improvement even for the strongest existing RFMs, and BusyBox as a benchmark is far from saturation.

Our study is only one example of experiments enabled by BusyBox. E.g., while we don't explore RFMs' spatial reasoning, BusyBox is highly suitable for empirical evaluations of it, on tasks such as *"pull the 2nd wire from the left"*. BusyBox can also be used for assessing the effectiveness of verbal corrections during task execution [11, 12]. More broadly, BusyBox was inspired by the interactive game *Keep Talking and Nobody Explodes* [13], which involves two players who need to communicate with each other in order to determine and execute a sequence of actions that defuses a bomb represented by BusyBox. BusyBox, as an implementation of this game, can be a rich environment for evaluating physical and digital AI agents and studying human-robot interaction.

Last but not least, we release the electronics designs that log BusyBox's state and allow for partly automating evaluations involving BusyBox. This is especially convenient for analyzing the execution of multistep tasks, e.g., *"move the top slider to 2, insert the red wire, and press the leftmost button"*, and makes BusyBox a physical counterpart of simulated benchmarks such as CALVIN [7].

In summary, our work's contributions are:

- BusyBox, a physical benchmark for evaluating basic affordance generalization and CAD designs for 3D-printing it.
- A language-annotated dataset of 1042 manipulation demonstrations, many of them bimanual, gathered on BusyBox's affordances.
- An experiment protocol for using this dataset or its equivalents for assesssing affordance generalization in RFMs, along with baseline empirical results on π_0.

All of the released materials are available at `https://microsoft.github.io/BusyBox`.

2 BusyBox design and instrumentation

The BusyBox system follows a modular architecture that uses easily reproducible 3D-printed components and instrumentation. To support robust and generalized policy evaluation, we provide scripts for automated assessment and data collection. All CAD files, source code, and instructions referenced herein are available on the project page at `https://microsoft.github.io/BusyBox`.

2.1 Modules

BusyBox comprises six distinct modules shown in Figure 1a, each representing a basic affordance. These modules are visually intuitive and engineered to be well within the physical manipulation capabilities of off-the-shelf 6-DoF robot arms with parallel grippers:

- **Display Module**: This module features an E Ink display and three LED indicators. It houses the main electronics of the BusyBox and provides visual feedback to both the RFM and the user. The E Ink display enhances information visibility for robots' cameras.
- **Buttons Module**: Consisting of four colored, illuminated buttons, this module requires the robot to press the correct button using a single arm. Rotating the module serves as a method to test policy generalization, as agents may otherwise memorize button positions.
- **Sliders Module**: This module includes two sliders, each adjustable between values 1 and 5. The primary challenge lies in determining which slider to move and ensuring accurate positioning.
- **Knob Module**: The knob can be rotated to a specified value between 1 and 6 and has a handle to facilitate its manipulation. Occlusions can complicate rotating the knob to a desired position.

- **Switches Module**: Featuring left and right switches with on/off positions, this module is visually simple but presents learning challenges, because the force required to flip the switch generally means that one arm must manipulate the switch while the other needs to pin BusyBox in place.
- **Wire Module**: This module involves inserting or unplugging colored wires. Both insertion and unplugging affordances are supported.

2.2 3D-printable design

One of our principal objectives in developing the BusyBox was to facilitate its replication in research laboratories. BusyBox also needed to be robust enough to handle rough handling as well as light enough to be picked up and moved by standard bimanual manipulators used in robot learning research. E.g., the common Aloha system with ViperX follower arms is capable of lifting only a 750 gram payload per arm. To this end, the body of BusyBox employs exclusively 3D-printed components in order to maximize BusyBox's strength-to-weight ratio and make it easy to manufacture.

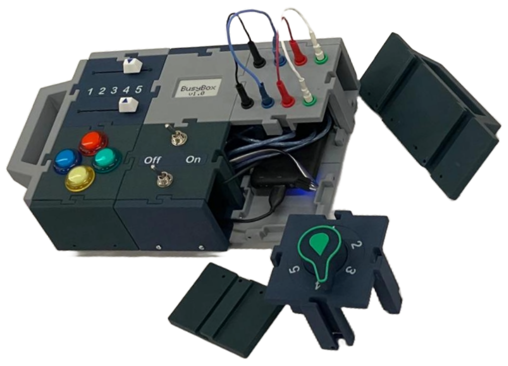


Figure 3: Disassembled BusyBox

As shown in Figure 3, BusyBox's components interlock using snap connectors, which streamlines both assembly and disassembly, thereby enabling rapid reconfiguration of the BusyBox. The perimeter of the BusyBox's interlocked modules is covered with side elements. Two of the side elements have handles, adding the affordances of conveniently picking up and rotating the BusyBox. The canonical BusyBox instance (Figure 1a) has 6 distinct modules in specific orientations, arranged in a three-by-two grid. To obtain other configurations, such as those in Figure 1b and Figure 1c, the modules can be easily permuted and rotated with respect to each other by 90, 180, or 270 degrees. BusyBox's design also allows multiple instances of the same module, alternative module arrangements such as two-by-two or two-by-one, and assemblies involving more than 6 modules.

We produced our BusyBox using a two-filament printer capable of printing in different colors. This enhances the legibility of textual elements, as demonstrated with high-contrast color combinations seen in Figure 3. For users lacking access to two-filament color printers, we have validated that single-color prints with manually painted highlights including the position numbers for the sliders and the knob are a practical alternative.

2.3 Instrumentation

To make both policy learning and evaluation easier, it is useful to have a mechanism that automatically records the state of all of BusyBox's controls at every time step. This not only helps determine if a task has been accomplished but is also valuable for tracking *progress* [10] when a task consists of several substeps, e.g., *"move the top slider to 2, insert the red wire, and press the leftmost button"*. With this in mind, we provide readily replicable *optional* electronic instrumentation that registers BusyBox state for live monitoring and/or recording it into demonstration trajectory files. We emphasize that even without this instrumentation, the BusyBox can be used for RFM evaluation.

The central control unit is a Raspberry Pi 0, which resides in the primary module alongside an E Ink display. This module must be included in the BusyBox configuration for the instrumentation to be functional. The Raspberry Pi 0 is augmented with a USB expansion board, enabling all individual components to interface via USB connections. We specifically selected the electrical components to be widely available and pre-instrumented to function as USB devices via Arduino serial communication. This architecture supports plug-and-play compatibility between BusyBox elements and the display module, enabling users to interchange modules freely. The instrumentation measurements are broadcast on the network at 10 Hz using a Raspberry Pi.

Task type	Instruction variants
Button	"Push the {color} button with the {left, right} gripper."
Slider	"Move the {top, bottom} slider to position{1, 2, 3, 4, 5}."
Knob	"Turn the knob to position {1, 2, 3, 4, 5, 6}."
Switch	"Flip the {left, right} switch {on, off} with the {left, right} gripper."
Pull Wire	"Pull the {red, black, blue, white} wire."
Insert Wire	"Insert the {red, black, blue, white} wire."
BusyBox pose	"Rotate BusyBox {clockwise, counter-clockwise}"
	"Move BusyBox {left, right, closer, away}"
Robot pose	"View BusyBox from above."
	"Move {left, right}, gripper to the {left, right}."
	"Open both grippers."
	"Open {left, right} gripper."

Table 1: Task/affordance types and instruction variants.

The wireless networking capabilities of the Raspberry Pi 0 allow users to remotely access and monitor the state of the BusyBox. When first powered on, the primary module displays the connection information on the E Ink display. For greater control, the users may also connect to the primary module directly via a USB-C interface. Power is supplied to all modules through the Raspberry Pi 0's USB ports, with the option of either a wired connection or a battery-powered setup. For battery operation, a standard 2500 mAh power brick is housed within the BusyBox.

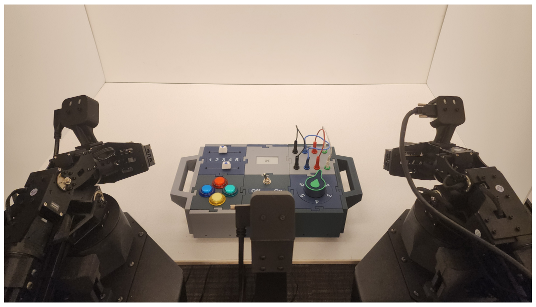


Figure 4: Illustration of our BusyBox data collection setup based on Mobile ALOHA.

To facilitate remote monitoring and control, we developed a lightweight Web server and browser-based interface for the BusyBox system. This interface provides real-time information on the operational status of each component of the BusyBox.

3 Data collection

To illustrate affordance generalization experiments enabled by BusyBox, we collected a finetuning dataset for adapting an RFM on BusyBox's affordances. To make this dataset useful for reproducing our experiments as well as for training RFMs, in this section we detail our data collection protocol and the dataset itself.

3.1 BusyBox dataset

We collected 1042 demonstrations data for the BusyBox task/affordance types listed in Table 1, with the breakdown of the number of demonstrations across affordances listed in Figure 2. Each task type was represented by by several possible language instructions. The instructions referenced color, relative position, or final position of the manipulated controls. For bi-manual setups, some of the language instructions also specified which manipulator to use. Table 1 covers the basic variations of the language instructions we recorded for each task. The number of variations likely to be encountered during deployment is far larger, since, e.g., the user may refer to a component by a variety of names, such as calling the rotating component a knob or dial. Our data collection workspace is shown in Figure 4.

3.2 Teleoperation instructions

3.2.1 Randomizing the initial state

Ensuring the diversity of initial states of the demonstrations is crucial for state coverage in the data and for learning robust policies. Given the number of factors of variation in the environment (positions of the sliders, the switches, the BusyBox itself, the robot, etc), we chose not to rely on the teleoperator to randomize the initial states along all these dimensions. Instead, generation of initial states is directed by a script that gives instructions to the teleoperator before the start of each demonstration. The parameters of each instruction are sampled uniformly at random, unless stated otherwise. We assume that, before the teleoperator executes these instructions, the robot is facing BusyBox directly, the BusyBox isn't rotated, the BusyBox is centered along the near edge of the workspace 2 inches from that edge, and all the wires are inserted, as in Figure 4:

- *Set the top slider to position s_{top} and the bottom slider to s_{bottom}*, where $s_{top}, s_{bottom} \sim \{1, 2, 3, 4, 5, \text{between 1 and 2}, \ldots, \text{between 4 and 5}\}$.
- *Set the knob to position s_{knob}*, where $s_{knob} \sim \{1, 2, 3, 4, 5, 6, \text{between 1 and 2}, \ldots, \text{between 6 and 1}\}$.
- *Set the top switch to position sw_{top} and the bottom switch to sw_{bottom}*, where $sw_{top}, sw_{bottom} \sim \{\text{on}, \text{off}\}$.
- With $p = 0.5$, leave all the wires inserted. Otherwise, for each wire independently with $p = 0.5$, pull out one end of that wire.
- *Rotate the BusyBox by roughly ϕ degrees clockwise*, where $\phi \sim \{-60, -40, -20, 0, 20, 40, 60\}$.
- *Move the BusyBox roughly x inches away and y inches to the right*, where $x \sim \{0, 2, 4\}$, $y \sim \{-4, -2, 0, 2, 4\}$.
- *Rotate the robot by roughly θ degrees clockwise*, where $\theta \sim \{-10, 10\}$.

3.2.2 Teleoperation style

To mitigate bias and unpredictable behavior in the finetuned models, the teleoperators were asked to follow the general rules for data collection:

1. Be efficient with movement. Demonstrations for tasks other than wire insertion should take no longer than 15 seconds. For wire insertion, demonstrations should not exceed 45 seconds.
2. Keep moving. Demonstrations should be active, only remain still when something dynamic is happening (i.e. dropping a wire).
3. End the demonstration when the requested task has been completed. There is no need to return to a neutral position.
4. Start from different initial positions before recording. These starting positions fall between the starting home position and somewhere above the BusyBox.
5. Whenever possible, ensure that both wrist cameras have a view of the parts of the environment needed for task at hand.

In every episode, the task is sampled uniformly at random from the set in Table 1 (expanded to include all the task variations). The initial state for a given episode is guaranteed to be constructed so that task's goal isn't achieved in it. E.g., if the task is *"Turn the knob to position 4 "*, then in the initial state sampling process above, '4' is excluded from the set of values for the knob before an initial state is sampled.

Additionally, teleoperators had to follow affordance-specific instructions:

1. **Buttons**: With grippers closed, press the button with one gripper. End recording just after the button is released and the gripper is a few inches above the button. If the button press was missed, retry at most once before ending the recording.

2. **Sliders**: With grippers closed, move the slider by pushing it in one direction or another until the slider reaches the position mentioned in the task description. If the value is overshot, correct the overshoot and end the recording.

3. **Pulling Wires**: Pull out only one end of the wire, leaving the other end inserted.

4. **Inserting Wires**: When inserting a wire, the demonstration should not exceed 45 seconds.

5. **Switches**: Brace the box by holding the handle with one gripper and move the switch to the desired position with the other gripper.

6. **Knobs**: With grippers closed, one gripper will nudge the knob until it is in the desired position and the other will view with its camera. If the target is overshot, just nudge in the other direction. If the box is moving while turning the knob, it is necessary for the second arm to brace the box.

4 Basic affordance generalization experiment

Using the dataset described in Section 3, we conducted a simple experiment meant to illustrate the utility of BusyBox by answering the conceptual question: how well does a state-of-the-art open-weights model do on affordance generalization?

In particular, we adopted π_0 [10] as the target RFM and measured:

- **Zero-shot success rate of π_0 (denoted as π_0-ZS)** on a set of BusyBox affordances on all 3 BusyBox configurations in Figure 1.

- **Few-shot success rate of π_0 (denoted as π_0-FS)** finetuned on our dataset, which was collected on the canonical BusyBox from Figure 1a, on a set of BusyBox affordances across on all 3 BusyBox configurations in Figure 1.

To measure the success rates, we chose 6 types of BusyBox affordances from – pressing button, moving sliders, turning the knob, flipping switches, pulling wires, and inserting wires – and sampled 10 instruction variants for each with replacement (see Table 1), for a total of 60 instructions. For each of $\{\pi_0\text{-ZS}, \pi_0\text{-FS}\}$ and each BusyBox configuration in Figure 1, our evaluation script commanded the Mobile Aloha to perform the 60 tasks in the same order, sampling the initial states as in Section 3.2.1 and setting the time horizon to 45 seconds for wire insertion and 30 seconds for all other tasks.

The results are presented in Table 2 and Table 3. Table 2 shows that without finetuning, π_0 completely fails in this experiment (see π_0-ZS). We attribute this not only to π_0's "unfamiliarity" with BusyBox, but also to its unfamiliarity with Mobile Aloha: the majority of π_0's pre-and post-training data appears to have come from other robot platforms. More interestingly, even the finetuned π_0 fails at *generalizing* affordances: it has some success only on the canonical BusyBoxconfiguration, on whose data it was finetuned. Indeed, according to our observations, π_0-FS's failures on non-canonical BusyBox variants were due to π_0-FS reaching for the wrong module.

Experiment	Successes	Trials	S.r.
π_0-FS_canonical	18	60	30.0%
π_0-FS_conf-1	0	60	0.0%
π_0-FS_conf-2	0	60	0.0%
π_0-ZS_canonical	0	60	0.0%
π_0-ZS_conf-1	0	60	0.0%
π_0-ZS_conf-2	0	60	0.0%

Table 2: Overall affordance generalization performance

Task	Successes	Trials	S.r.
pull_wire	4	10	40.0%
push_button	4	10	40.0%
move_slider	3	10	30.0%
turn_knob	5	10	50.0%
insert_wire	0	10	0.0%
flip_switches	2	10	20.0%

Table 3: Success Rate per Task for π_0-FS_canonical

5 Related Work

Compared to the existing evaluation tools for robot manipulation models, the novelty of our work is in introducing a physical benchmark for systematic evaluation of affordance generalization in a semi-automated way.

5.1 Physical Benchmarks

Physical benchmarks are common in robotic manipulation research and typically represent tasks that are difficult or tedious to simulate. Examples include the Functional Manipulation Benchmark (FMB) [14], FurnitureBench [15], NIST Task Boards [16], and Digital Robot Judge Task Boards (DR. J.) [17]. Like BusyBox, NIST Task Boards and FurnitureBench involve manipulating deformable and articulated objects such as wires and switches. Their focus, however, is on assessing a model's ability to drive difficult contact-rich manipulation rather than its ability to generalize. FMB has been used to benchmark generalization, but not specifically in the affordance space. DR. J. logs changes in the box state, which can be used to verify task completion but, like FMB, doesn't evaluate affordance generalization.

5.2 Simulated Benchmarks

Simulation-based benchmarks robots provide consistent and controllable environments that make task completion easier to measure reliably and record automatically. However, for modeling objects such as switches, wires, and sliders, with all their imperfections that frequently hamper manipulation, simulation has so far been too imprecise, too slow, or both. Common simulated task suites, including LIBERO [8] and SimplerEnv [18], don't involve complex objects like these, and evaluating affordance generalization isn't their focus. The simulation-based benchmark that involves operations most similar to those on BusyBox is BusyBoard [19]. However, BusyBoard is still very distinct, lacks open-source CAD files for reproducing it, and, like the other benchmarks, sidesteps evaluating affordance generalization.

5.3 Automated Evaluation on Real Hardware

AutoEval proposes an autonomous real-world evaluation framework for generalist robot manipulation policies, automating task orchestration, success detection, logging, and environment resets to reduce human supervision and enable long, unattended runs [20]. BusyBox is complementary; its sensor-instrumented modules (switches with detents, rotary knobs with indexed targets, sliders with visual setpoints, pushbuttons, and wire routing/pulling) provide unambiguous, per-module success signals and expose contact-rich, often bimanual behaviors that are hard to detect reliably with vision alone. In practice, BusyBox can serve as an AutoEval-style evaluation target.

6 Conclusion

We presented an instrumented, modular, reconfigurable BusyBox for benchmarking affordance generalization in RFMs on real hardware. BusyBox targets basic affordances such as flipping switches and plugging in audio cables, which are ubiquitous in home and industrial environments but are underrepresented in existing datasets and benchmarks. BusyBox comes with open-source CAD files and electronics design for easy reproduction in robotics research labs. Using a demonstration dataset, which we collected on BusyBox and are also open-sourcing in this work, we conducted an illustrative empirical study showing that even SOTA RFMs like π_0 currently struggle with affordance generalization. We hope that BusyBox will encourage more active and more reproducible research on this subject.

References

[1] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, S. K. S. Ghasemipour, C. Finn, and A. Wahid. Aloha unleashed: A simple recipe for robot dexterity. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 1910–1924. PMLR, 06–09 Nov 2025.

[2] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

[3] Z. Fu, T. Z. Zhao, and C. Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In *Conference on Robot Learning (CoRL)*, 2024.

[4] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang. On the opportunities and risks of foundation models, 2022. URL https://arxiv.org/abs/2108.07258.

[5] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, B. Ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal, N. Brown, A. Brohan, M. G. Arenas, and K. Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

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

[7] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7327–7334, 2022.

[8] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *NeurIPS-2023 Track on Datasets and Benchmarks*.

[9] Y. Ju, K. Hu, G. Zhang, G. Zhang, M. Jiang, and H. Xu. Robo-ABC: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *ECCV*, 2024.

[10] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. URL https://arxiv.org/abs/2410.24164.

[11] H. Liu, A. Chen, Y. Zhu, A. Swaminathan, A. Kolobov, and C.-A. Cheng. Interactive robot learning from verbal correction. In *CoRL LangRob Workshop*, 2023. URL https://arxiv.org/abs/2310.17555.

[12] L. X. Shi, Z. Hu, T. Z. Zhao, A. Sharma, K. Pertsch, J. Luo, S. Levine, and C. Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024. URL https://arxiv.org/abs/2403.12910.

[13] A. Pestaluky, B. Kane, and B. Fetter. Keep talking and nobody explodes. https://keeptalkinggame.com/, 2015. Steel Crate Games.

[14] J. Luo, C. Xu, F. Liu, L. Tan, Z. Lin, J. Wu, P. Abbeel, and S. Levine. Fmb: a functional manipulation benchmark for generalizable robotic learning. *International Journal of Robotics Research*, 2024.

[15] M. Heo, Y. Lee, D. Lee, and J. J. Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. In *Robotics: Science and Systems*, 2023.

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

[17] P. So, A. Sarabakha, F. Wu, U. Culha, F. J. Abu-Dakka, and S. Haddadin. Digital robot judge: Building a task-centric performance database of real-world manipulation with electronic task boards. *IEEE Robotics and Automation Magazine*, December 2024.

[18] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, S. Levine, J. Wu, C. Finn, H. Su, Q. Vuong, and T. Xiao. Evaluating real-world robot manipulation policies in simulation. In *CoRL*, 2024.

[19] Z. Liu, Z. Xu, and S. Song. Busybot: Learning to interact, reason, and plan in a busyboard environment. In *CoRL*, 2022. URL https://arxiv.org/abs/2207.08192.

[20] Z. Zhou, P. Atreya, Y. L. Tan, K. Pertsch, and S. Levine. Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world, 2025. URL https://arxiv.org/abs/2503.24278.