

Table30 V2: Evaluating Generalized Models by Real Robots at Scale

RoboChallenge Dexmal Team
Dexmal

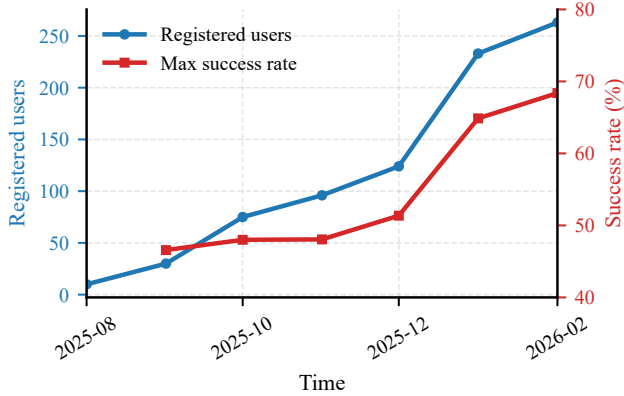


Figure 1. User activity on the previous Table30 benchmark. The SOTA success rates of the tasks are rapidly increasing as users accumulate.

Abstract

In this work we present Table30 V2, an upgraded version to the Table30 benchmark [8] on the real-robot-based large-scale evaluation system of RoboChallenge. Its main emphasis is on testing the generalization aspect of the models, which we believe will be the key feature of the next-generation embodied models. For this purpose, we take measures to rework the benchmark task set to cover the enhanced strength of the models, and more importantly, we fundamentally refactored the learning-testing protocol to exercise the generalization ability of the models. We will support multi-task, zero-shot and out-of-domain paradigm of evaluation while keeping the core openness-and-fairness spirit of the RoboChallenge campaign untouched. A preview version of the benchmark will be online as part of the CVPR 2026 workshop competition of GigaBrain Challenge 2026.

1. Introduction

Embodied policies like VLA or WMA made tremendous progress on robots in the recent period, as witnessed by the rapid improving results on the Table30 benchmark (Fig. 1)

hosted by the RoboChallenge program. With the online real-robot-based evaluation paradigm, we can clearly observe the evolving capabilities of the models. However, as the SOTA entries accumulate, the benchmark itself is prone to saturation. This pushes us to defined an upgraded benchmark set for evaluating future models.

Our core emphasis is on the **generalization** ability of the models. The reason is as follows. Firstly, the current generation of open-source models like $\pi_{0.5}$ [2, 3, 5, 6, 9, 11] have already been highly successful in single task focused finetuning setting. From our practice, even very seemingly hard task can be well tackled by careful curation of the training set, and the model is usually finetuned to a point where it loses most capabilities outside the task. So we suggest that testing models on a large set of skills will be a better way to reveal the true capability of the model. Second, even in the single-task finetuning setting (common in real world applications), a more generalizable model requires much less effort in data preparation. A better starting point benefits model convergence, and hence loosen the constraint on data coverage. Finally, object-level and background-level generalization to unseen settings is a must-have in real world applications. An intelligent model is not believed to work if we have to strictly reproduce all its input conditions. So we need to test these aspects of the model for more realistic evaluation.

Testing for generalization is a non-trivial objective. We decompose this target into three directions of improvements to our benchmark. **Task-level** expansion of the benchmark suite introduces more challenging or realistic tasks. They are to test the upper-bound of model capability in highly adaptive and dexterous jobs, which we believe is a consequence of the model being more generalized. **Protocol-level** re-design of the learning and testing requirements create new tracks of multi-task, zero-shot and out-of-domain testing. We need to tailor our testing procedure so that the intended testing method matches the online evaluation format. **System-level** improvements to the underlying evaluation infrastructure aims to increase the total throughput of the tests. As we want stabilized results for more test protocols, the necessary number of evaluation runs will be multiple times of the status quo. So we need the throughput to

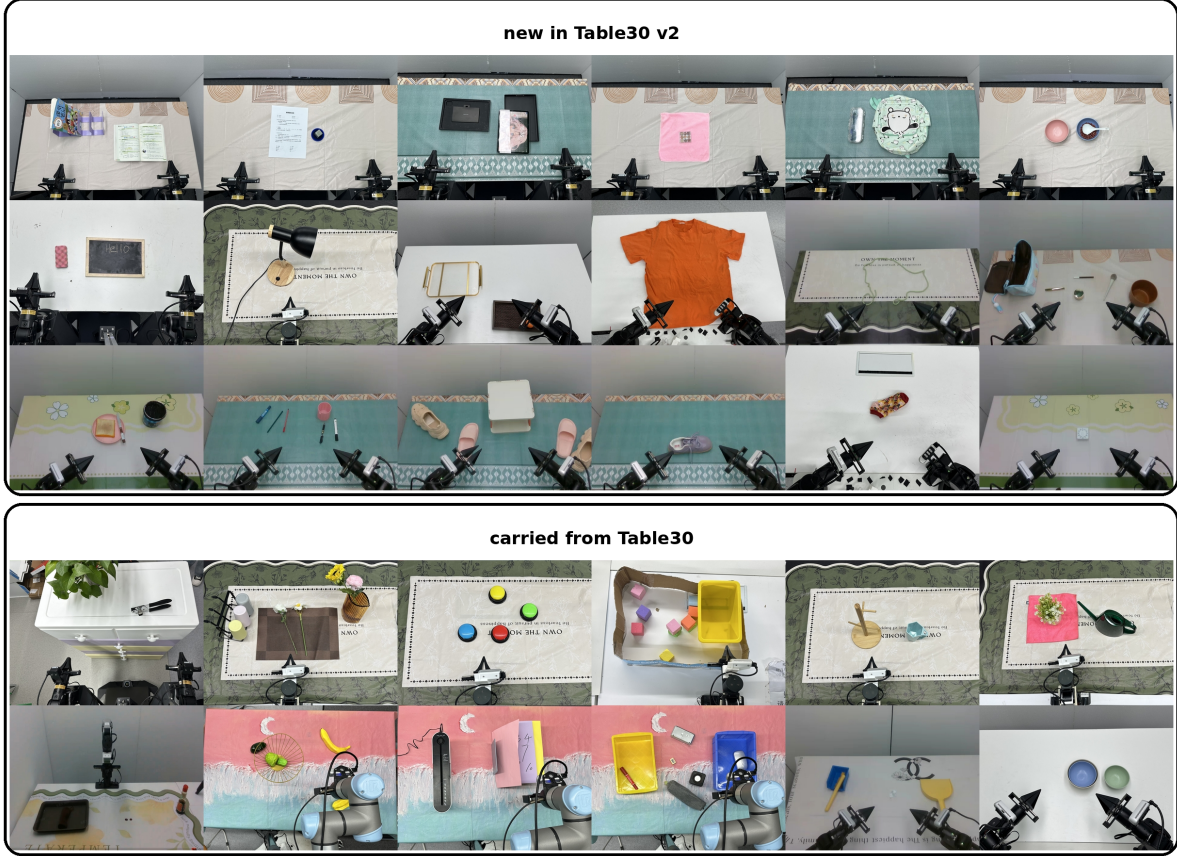


Figure 2. The task set of Table30 V2. It includes 18 freshly designed bimanual tasks, and 12 tasks carried from Table30.

match the new requirements.

Accomplishing the goals above require careful revisit and design of the benchmark system. We will cover these aspects in the following sections.

The key delivery of Table30 V2 are as follows:

- An upgraded benchmark task set for large-scale real-robot-based evaluation of embodied policies;
- Proposed method to test multi-task and out-of-domain generalization;
- Improved evaluation system throughput for online real-robots.

2. Related Works

Testing embodied by real robot is an increasingly popular topic in the community. The early works of [1] and [12] initiated the on-line evaluation format, but does not include a large scale task set. The RoboChallenge [8] demonstrates the feasibility of online and large scale real-robot based testing with 30 tasks. Following the work, [10] proposes another benchmark with 20 tasks.

There are also non-centralized real-robot based eval systems proposed for scalability. The [4] and [7] methods let

the user submit their videos for judgement.

This benchmark is moving along the trend of centralized, real-robot-based and online evaluation, yet still the largest in terms of number of tasks.

3. Benchmark Tasks

3.1. New Tasks

As the capability of the embodied models expand, we find there are several important aspects that are not covered in the previous version of Table30. These lays the core motivation of the expansion of the task set.

Handling soft continuum. There are objects like rope, stripe or cloth that can change its shape or even topology in a continuous manner. This creates vast variation in the state of the system, as a state of the object can hardly be produced from run to run. The model need to handle a never seen shape or state of the object.

Using tools. In realistic tasks, the robot may need to manipulate a tool to accomplish the task. This tests the robot’s ability to understand the tool-object spatial relationship and exercises its precision of control.

Task Name	Description
Pick opened book	The robot picks up opened books from the the table to a shelf.
Tie a knot	The robot grips both ends of the strip, forms a square knot, and pulls it tight.
Stamp on paper	The robot presses the stamp onto an ink pad and stamps the paper.
Sort cosmetics	The robot puts the cosmetics into the brush pot and the tray.
Spread jam	The robot spreads jam on the bread.
Box the tablet	The robot puts the tablet and accessories into the box.
Wrap with cloth	The robot places the item on the cloth and wraps it around from all sides.
Pick the pen	The robot picks up the pen on the table into pen container.
Pack up Stationery box	The robot opens the schoolbag and puts the stationery box on the desk into it.
Pair the shoes	The robot put four shoes onto the shelf in two pairs.
Remove laces	The robot removes the laces from the shoes, and place them on the table.
Scoop the beans	The robot scoop beans to a bowl.
Wipe whiteboard	The robot wipes the whiteboard with an eraser.
Set the timer	The robot sets a timer to a specific time.
Turn on lamp switch	The robot turns on a lamp switch by pressing the button on it.
Move the tray	The robot uses dual arms to pick up a tray and places it into a basket.
Folding T-shirt	The robot folds the T-shirt in half twice and put it in the corner of the table.
Pack socks	The robot turns the socks inside-out, fold them and put in the corner of the table.
Store the opener	The robot store the opener into the drawer.
Sweep the table	The robot sweeps the dust to into the dustpan.
Arrange the flowers	The robot puts the flowers into a vase.
Press four buttons	The robot presses four buttons sequentially on the table.
Search green boxes	The robot finds all green boxes and put them to basket.
Hang the cup	The robot hangs the cup on a rack.
Water the flowers	The robot waters the flowers with a watering can.
Wipe the table	The robot wipes the table with tissue paper.
Arrange fruits in basket	The robot move fruits on the table to a basket.
Shred the paper	The robot puts the paper into a shredder.
Sort electronic products	The robot puts stationary into the yellow box and electronic products to the blue box.
Stack bowls	The robot stack three bowls on the table.

Table 1. Description of the tasks

Simultaneous two-hand operation. In the previous version, for the bimanual robots usually only one arm is in action at the same time. We deliberately add tasks that require the two hands to operate at the same time. This also challenges the control precision, as the relative pose of the hands will be in a constrained state when moving together.

The result of our thinking leads to 18 newly designed tasks. They are listed in Tab. 1 and illustrated in Fig. 2. Almost all these tasks are executed on bimanual robots.

We also carry 12 tasks from the previous version of Table30. They are dedicated to test the performance of single-arm-based jobs, or the jobs using dual-arm asynchronously. We select the tasks that are most discriminative and popular in the community.

3.2. New Robot Platform

We find the bimanual robots to be the most popular in the community. In the previous version, only one machine type is used, the Cobot Magic Aloha. So in this version, we

add a new type of robot, the DOS-W1. It is also a mobile bimanual manipulation, yet the arm becomes Airbot Play. The cameras are keep consistent to the Aloha setup, the RealSense series being used. The gripper in the DOS-W1 setup has a triangle tip, enabling it to more fine-grained manipulation tasks.

Addition of the robot type enables us to verify whether the model’s behavior is consistent across the arm types. In the previous version, the “performance on bimanual” is identical to “performance on Aloha”. Now we can separate the two concepts. Also, the DOS-W1 has a lower listing price that other bimanual manipulators, so we can better amortize the hardware cost.

We also phased out the Franka Research 3 robot, as we observed the tasks it is executing can well be replaced by other arms while the hardware cost is greatly reduced. The UR5 arm strikes a good balance between cost and performance. The ARX5 arm is selected due to its good endurance and low cost.

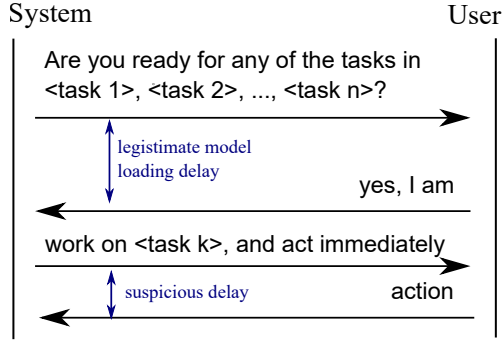


Figure 3. Task-agnostic online interaction protocol. The user gets the exact task seconds before the first interaction starts. This probes potential change-of-model behavior. However, it is a necessary-but-not-sufficient test.

3.3. Statistics and Analysis

Across the 30 tasks, we find a good distribution among the machine types. The DOS-W1 receives 12 tasks, the Aloha 8 tasks. For the single-arm setup, the ARX5 handles 7 tasks, while the UR5 implements 3 tasks. For each type of robot, we purchased the number of robots roughly proportional to the tasks on it, so the evaluation workload can be evenly distributed among the machine types.

As for the tasks, 7 of them involves using a tool, 9 of them requires two-hand coordination, 6 of them requires high position accuracy. And these three sets of task intersect with each other, so we have a full combination of the task difficulties.

As for the application areas, most of the tasks are kept as natural skills in everyday life. We removed tasks like sandwich making that tortures the hardware more than the model. The tasks are natural both to human and the robot, so what we are evaluating will be fully targeted on the intelligence part.

4. Evaluation Protocols

Nowadays we want to test the generalization ability of the model, and for an online testing system there are extra requirements that we must meet.

The first is **fairness**. We should ensure that the testing method will not bias towards a particular user or model. The outcome is ideally i.i.d. samples of the “true” performance of the model on real machines. Also, we need to make sure that cheating on the system, especially using more restrictive protocols, requires extra work than benign users. This will not stop users’ from claiming false results, but it will reduce the probability.

The second is **maintainability**. We need to make sure the benchmark service can run for a long duration of time. When some machines are under maintenance, the service

should not stop. This requirement is strengthened when we shift from the single-task protocol to the generalized settings. When we conduct evaluation of one model on multiple tasks simultaneously, we need to make sure all the dependent machines are working. So we need to take more measures to avoid single-point-of-failure in the system.

4.1. Multi-task and Multi-machine Setup

In all our evaluations, we require that the model handles multiple tasks of one machine type, and it should operate well on any instance of the machine inside one type.

In the previous version of Table30, each task only belongs to one machine instance. This creates many hot spots in the machine workload, and when one machine is down, the whole benchmark cannot be completed. We decide to get rid of this weakness. However, it is well known that with the same model and the same task, performance among the machine instances will vary. It is impossible to exactly reproduce the camera and arm parameters across instances. To alleviate this, we collect data on all of the machine instances of the same machine type for each task. We observed models trained on the mixture of machine instances tend to generalize across small variations in the set-up. Further, to ensure fairness, when an evaluation job is submitted, the machine instance to do one task is randomly selected. And as we require multiple tasks to be tested in one row, the randomness will be amortized – it will be harder to roll for the “lucky machine” for a particular task.

As for the multi-task requirement, the main motivation is to relief the burden on the participants. We observe that when the tasks are evaluated individually, the participants spend very long time on tuning the model parameters of each individual task. Though this may lead to progress on the maximum achieved score, it deviates from what we really want to measure. Also, training 30 individual models consume a great amount of GPU hours, hampering labs and students from joining the race. So we decide that the benchmark will be multi-task models only (the generalist setting in Table30). This is in line with the trend to test the generalized models. The multi-task setting can be seen as a mini version of the task-level generalization, although we do not emphasize on this.

It is not easy (or even possible) to enforce a multi-task single-model algorithm. However, we still decide to augment the current protocol so that participants have more tendency to use real multi-task models. The idea is that the nowadays models are usually huge in parameter count, so loading them will take a measurable amount of time. Given this, we use a timing challenge to test if the model is really multi-tasked, or dynamically switched. As seen in Fig. 3, when a task is about to start, we first waits for the model to claim that it is fully loaded and ready to take control for a set of tasks. Only then, the actual task will be

given to the model as the instruction prompt. The model is supposed to generate its response with a reasonable delay time (hundreds of milliseconds for typical models). If the gap time is huge (on the order of several seconds or tens of seconds), there is a strong cue that another model is swapped in. Again, this protocol does not stop cheating – the participant can load many models on multiple machines in parallel and routes the request by the prompt. However, doing this incurs increased cost, so we just use this machinery as an auxiliary check of the nature of the model. The main constraint is still enforced by morality, as the participant need to agree our terms of “following the protocols strictly” when they submit their entries, and we encourage the users to open-source their models if they want to claim a prize in the competitions.

4.2. The Zero-shot Setting

When we talk about generalization, we need to introduce unseen factors to the system. So in addition to the usual mode of using the same distribution across training and inference, we create a dedicated mode of “zero-shot” testing.

Currently we consider two level of zero-shot generalizations:

- **Object-level zero-shot.** We will use unseen objects to test the model.
- **Environment-level zero-shot.** We will put the task in an unseen background / environment setting.

There are also more aspects we can test for, e.g. task-level or prompt-level zero-shot generalization. However, we decide to focus on the above two setups, as they are already hard enough.

For the object-level zero-shot testing, we will prepare dedicated objects in the same category as described in the task description. We make sure that in the demonstration data these objects do not appear. The objects will vary in shape, size or color, so the model need to get a better understanding of its tasks instead of memorizing the grasp position.

For the environment-level zero-shot testing, we will mainly use new backgrounds to test the model. Also we will make slight perturbations to the system, e.g. the height of the table so that the dynamics may change. As the exact details in each perturbation is not revealed to the participant, the model may encounter unanticipated environmental factor change, so it must be trained in a generalizable way.

We do not release demonstration data to the participant on these zero-shot objects or settings. However, to ensure fairness, we still stick to the rule that the same set of objects will be used to test all models. So there is a theoretical way of cheating by downloading the successful runs of other models as training data. We again leaves this rule enforcement to the participants themselves: It is not allowed to train the testing objects by any means (including purchas-

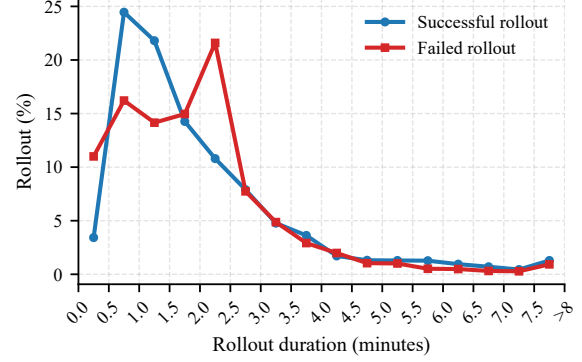


Figure 4. Running time distribution of the tasks in Table30.

ing the objects and record data themselves).

Given the two levels of zero-shot settings, there are $2 \times 2 = 4$ combination of protocols. We will rank the models separately in each track. Also, a model can claim itself to be of task-level zero-shot (using no demonstration data provided by us) or generalizable to different machine types (using one single model for evaluation on the four machine types). We do not check for this, while we encourage the participants to disclose the details of their methods if they claim a stronger level of generalization.

4.3. The Out-of-domain (OOD) Setting

To make the protocol future-proof, we propose another more challenging setting, which we call the OOD setup. In this setup, the task is not changed, and the environment / objects are largely the same. However, all others can be changed, in a totally unknown way. The positions of the objects can be in a very peculiar way (but still within the physical reach of the robot), the table may be changed into a sofa, etc. Also, the test cases can be adversarially set up for the pure reason of finding weakness in the model.

This mode is not intended to be ranked, as we will be constantly changing the exact test cases so that they are essentially unknown to the participants, and we will attempt to devise “bad cases” depending on previous behavior of the tested models. This is a self-check more than a competitive evaluation. It studies whether given the task description (and possibly a few demonstrations) the model learns the essential purpose of the task, in spite of all other specific factor like the shape / color / arrangement of the objects.

This setting will be setup online when we observe that most models tested by the system are already achieving saturating performance. We hope it will help shed light on future directions of improvement when we want to push the robots to work in real scenarios.

5. Evaluation System

In our past evaluations, we find the most important issue is the throughput of the evaluation system. As the number of users increased, a noticeable delay exists after the submission of the evaluation request. This slows down the iteration cycle of the participants, so this is one of the biggest challenge that we need to invest in.

The first technical issue we see is the overly long execution time of some tasks. See Fig. 4 for example. Some rollout goes for 7 or more minutes, even the demonstration data is typically below 1 minute. They consume a significant portion of time. We empirically observe the long runs. The long running time is mostly attributed to long response delay: The model takes several seconds to give an action after receiving the observations. Another cause is too many retries in each phase (we limit the number of retries to 5 in each step, so there may be tens of retries in one successful run). Also, some models purposely slow down the action chunk in order to obtain better success rate.

To handle this, the first approach we will take is to limit the total running time of a rollout. The model will be halted (and scored according to current progress) after a task-specific timeout. The timeout will be determined according to our baseline implementations and the general distribution of the user’s submissions. Again, according to Fig. 4, most successful rollouts achieved its success in a very short time, so the time limit should not be altering the overall performance of the models, as it will impact the maximum achievable results if some “inference-time tricks” are taken. Also, we will be adding a **time to complete** score to the leaderboard, showing the average time of the model to obtain its success rate. This should encourage the participants to be aware of the running time of their policies, and hence reduce the occupied time on the real machines.

There are also other measures that we will take to reduce latency. We will loose the requirement on reproducing the initial state to a reference image from pixel-level alignment to rough alignments, as we are testing the generalization of models anyway. This significantly speeds up task preparation time as we observe in practice. We have already been spreading the evaluations to multiple instances of the same machine type as we described in the previous section, so the duty cycle of the machines may be improved. Finally, we increased the number of machines for an increased upper bound. After all these measures, we hope the user can get a better experience in our system.

6. Evaluation Activities

The number of attracted users in Table30 goes far beyond our expectation. Not only institutional participants like the Embodied Intelligence start-ups submit their models to compete for SOTA, but also individual researchers or stu-

dents create valid submissions to exercise their models. We collected very valuable feedback on the content, operation and rules of the system. Especially, the request for more testing bandwidth and arguments around fairness push us to think fundamentally of how future evaluations should look like. This version of Table30 V2 roots the idea from both our organizers and users.

Perhaps the most fundamental change in terms of user base is our switch to multi-task evaluations. Previously, tuning for 30 different tasks is both laborious and costly. So most individual contributors cannot get a full score averaged over all tasks, and on the leaderboard we can only see institution backed participants. We hope our change enable more individual researchers and students to join this activity.

We will setup an initial version of this benchmark in the CVPR 2026 Workshop Competition held by GigaBrain Challenge 2026 and invite participants to compete. After the competition, we plan to continue the operation of the benchmark set so that all researchers worldwide can utilize this infrastructure. The version used by the competition will be a preview of the full benchmark due to schedule constraints. Advanced features like zero-shot or OOD will not be used to score the participants. These entries will be used as the seed users so that we can continuously improve our system before being open to public.

7. Concluding Remarks

In this paper we describe the Table30 V2 benchmark, a successor of Table30 that aims the next generation of stronger embodied models. The core emphasis is on generalization, which is manifested by the expansion of task set, addition of test protocols and improvements of the overall system.

We hope our work will inspire future researchers on both constructing their own in-house evaluation facilities and designing new models based on the findings in the evaluations. Also, we welcome anyone interested in the area to join the Table30 V2 evaluation activity to compare with each other in a shared ground and ignite ideas for future research.

References

- [1] Pranav Atreya, Karl Pertsch, Tony Lee, Moo Jin Kim, Arhan Jain, Artur Kuramshin, Clemens Eppner, Cyrus Neary, Edward Hu, Fabio Ramos, et al. Roboarena: Distributed real-world evaluation of generalist robot policies. *arXiv preprint arXiv:2506.18123*, 2025. 2
- [2] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025. 1
- [3] Kevin Black, Noah Brown, James Darpinian, Karan Dhaliwal, Danny Driess, Adnan Esmail, Michael Robert Equi,

Chelsea Finn, Niccolo Fusai, Manuel Y Galliker, et al. *pi*_{0.5}: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025. [1](#)

- [4] Yiting Chen, Kenneth Kimble, Edward H Adelson, Tamim Asfour, Podshara Chanrungmaneeikul, Sachin Chitta, Yash Chitambar, Ziyang Chen, Ken Goldberg, Danica Kragic, et al. Manipulationnet: An infrastructure for benchmarking real-world robot manipulation with physical skill challenges and embodied multimodal reasoning. *arXiv preprint arXiv:2603.04363*, 2026. [2](#)
- [5] GigaBrain Team, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Jie Li, Jindi Lv, Jingyu Liu, Lv Feng, et al. Gigabrain-0.5 m*: a vla that learns from world model-based reinforcement learning. *arXiv preprint arXiv:2602.12099*, 2026. [1](#)
- [6] Spirit AI Team. Spirit-v1.5: Clean data is the enemy of great robot foundation models, 2026. Accessed: 2026-03-21. [1](#)
- [7] Ziyu Wang, Chenyuan Liu, Yushun Xiang, Runhao Zhang, Qingbo Hao, Hongliang Lu, Houyu Chen, Zhizhong Feng, Kaiyue Zheng, Dehao Ye, et al. The great march 100: 100 detail-oriented tasks for evaluating embodied ai agents. *arXiv preprint arXiv:2601.11421*, 2026. [2](#)
- [8] Adina Yakefu, Bin Xie, Chongyang Xu, Enwen Zhang, Erjin Zhou, Fan Jia, Haitao Yang, Haoqiang Fan, Haowei Zhang, Hongyang Peng, et al. Robochallenge: Large-scale real-robot evaluation of embodied policies. *arXiv preprint arXiv:2510.17950*, 2025. [1](#), [2](#)
- [9] En Yu, Haoran Lv, Jianjian Sun, Kangheng Lin, Ruitao Zhang, Yukang Shi, Yuyang Chen, Ze Chen, Ziheng Zhang, Fan Jia, et al. Dm0: An embodied-native vision-language-action model towards physical ai. *arXiv preprint arXiv:2602.14974*, 2026. [1](#)
- [10] Sun Yu, Cao Meng, Yang Ping, Xu Rongtao, Xu Runze, Ma Liang, Gan Roy, Zhai Andy, Wang Hao, Liang Lucy, Wang Qian, Laptev Ivan, D Reid Ian, and Liang Xiaodan. Maniparena: Comprehensive real-world evaluation of reasoning-oriented generalist robot manipulation, 2026. Accessed 2026-03-21. [2](#)
- [11] Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, Ethan Yin, Hao Wang, Hugo Zhou, James Wang, Lights Shi, et al. Igniting vlms toward the embodied space. *arXiv preprint arXiv:2509.11766*, 2025. [1](#)
- [12] Zhiyuan Zhou, Pranav Atreya, You Liang Tan, Karl Pertsch, and Sergey Levine. Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world. *arXiv preprint arXiv:2503.24278*, 2025. [2](#)