# EAGC 2026 World Model Challenge: Public Environments, Tracks, Rules and Baselines

Xutao Sun[1,*], Dongyan Huang[2,3,*], Yixing Gao[1,*], Hongxia Xie[1], Jifeng Hu[1], Yan Wu[4],Hui Bu[5,*], Haifeng Zhong[1]
[1]School of Artificial Intelligence, Jilin University, Changchun, China
[2]International Graduate School, Tsinghua University, Shenzhen, China
[3]Joylift AI, Shenzhen, China
[4]A*STAR, Singapore
[5]AISHELL Technology Co., Ltd. Beijing, China

***Abstract*—The EAGC 2026 World Model Challenge is a simulation-based international algorithmic challenge for evaluating executable world models in embodied intelligence. Recent progress in model-based reinforcement learning, embodied simulation, imitation learning, and language-conditioned agents has shown strong potential for learning predictive representations and action policies. However, many benchmarks still emphasize final task success, offline prediction, or public task instances, making it difficult to measure whether a system constructs a reusable internal model and uses it for planning, execution, adaptation, and recovery. EAGC 2026 addresses this issue by evaluating world-model systems in standardized simulated environments with hidden tests, structured model-output files, and centralized scoring. The challenge contains two tracks. Track 1 evaluates unseen-environment exploration and closed-loop task execution, where an agent explores a partially unseen indoor environment, builds a queryable world model, receives a natural-language task, and recovers from a controlled exception. Track 2 evaluates one-demonstration-driven rapid adaptation, where a system receives a single expert demonstration and must infer transferable task variables, constraints, preconditions, and success criteria before executing an unseen instance. This paper presents the public resource policy, evaluation workflow, scoring criteria, submission requirements, and baseline results for both tracks.**

***Index Terms*—Embodied AI, world model, simulation benchmark, closed-loop execution, rapid adaptation, robot learning.**

## I. Introduction

Embodied agents must perceive the environment, maintain task-relevant state, select actions, and revise their decisions after new observations. This requires more than passive recognition or one-step control. An effective world model should organize spatial topology, object states, affordances, task constraints, and action consequences into a representation that can be queried and used during execution. Fig. 1 illustrates the two task scenarios used by EAGC 2026: unseen-environment exploration with closed-loop execution, and one-demonstration-driven adaptation on a new manipulation instance.

World models have long been studied in reinforcement learning and control. Neural world-model methods learn compact latent dynamics for policy learning and planning [1], [2], [3], [4], [5]. Related model-based methods further show that task-relevant dynamics can support planning even when the learned model is not a complete physical simulator [6], [7], [8]. These works provide important algorithmic foundations, but standard control benchmarks usually report reward or success rate. The internal model is rarely checked as a structured, queryable object.

Embodied simulation platforms have also advanced rapidly. AI2-THOR, Habitat, Gibson, VirtualHome, Habitat 2.0, ProcTHOR, ALFRED, and BEHAVIOR-1K provide interactive indoor environments, navigation tasks, rearrangement tasks, language instructions, and household activity settings [9], [10], [11], [12], [13], [14], [15], [16]. For manipulation, RLBench, ManiSkill2, MimicGen, LIBERO, RoboCasa, and RoboCasa365 provide task families, demonstrations, object variations, and generalization protocols [18], [19], [20], [21], [22], [23]. These resources are valuable for training and task design, but a competition still needs a clear boundary between public resources and hidden evaluation.

Recent language-conditioned and foundation-model-based agents expand the solution space for embodied tasks [24], [25], [26], [27], [28], [29], [30], [31], [33], [34], [35]. Their progress also makes evaluation more difficult. A high task success rate may come from memorized public tasks, reactive policies, manually engineered prompts, or large pretrained priors. EAGC 2026 therefore evaluates not only task outcomes, but also whether the submitted system maintains a structured model, updates it over time, exposes it through verifiable files, and uses it when unexpected changes occur.

The challenge is organized as an online evaluation followed by an on-site final presentation in early August 2026. It uses standardized simulated environments and hidden evaluation episodes. Participants submit executable runnable containers and structured model files. During official evaluation, inference, memory updates, lightweight online adaptation, and declared local models are allowed; internet access, online model APIs, hidden-ground-truth access, manual remote control, reverse engineering of hidden assets, and large-scale retraining are prohibited.

The rest of this paper is organized as follows. Section II introduces public resources and the evaluation workflow. Section III describes Track 1. Section IV describes Track 2. Section V summarizes common rules and submission require-

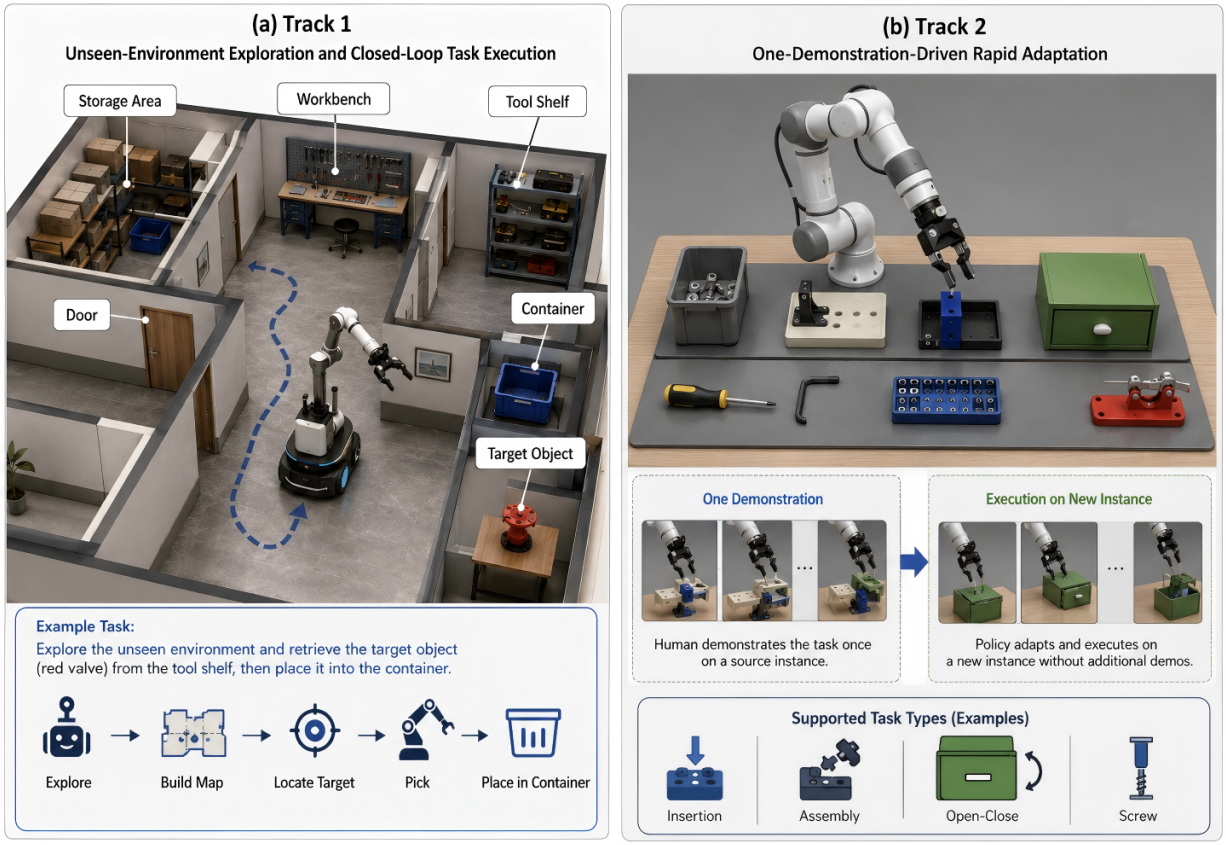


Fig. 1. Typical task scenarios of EAGC 2026 World Model Challenge. Track 1 evaluates exploration, world-model construction, target localization, manipulation, and placement in a hidden indoor environment. Track 2 evaluates whether a system can transfer a single source demonstration to a new manipulation instance.

ments. Section VI lists important dates. Section VII concludes the paper.

## II. Public Resources and Evaluation Workflow

This section describes the resource policy and evaluation workflow of the challenge. Different from dataset-release challenges that provide a single official training corpus, EAGC 2026 uses public environments, simulators, and benchmark tasks as training resources and task-design references. Official scores are produced only through hidden evaluation. Fig. 2 summarizes the competition workflow, from registration and public resource release to qualification, hidden validation, final frozen submission, official testing, and on-site presentation.

### A. Public Resource Policy

Public resources are used for training, validation, method development, and comparison. They are not direct copies of official hidden tests. Participants may also use private data, synthetic data, and public pretrained models, provided that sources, licenses, model weights, and external dependencies are declared in the technical report.

Table I summarizes the recommended resources. For Track 1, the recommended combination is ProcTHOR, Habitat Challenge, and ALFRED. For Track 2, the recommended combination is RLBench, MimicGen, and LIBERO. ManiSkill2, RoboCasa, RoboCasa365, and BEHAVIOR-1K are useful for expanded training and future task extensions.

### B. Hidden Evaluation

Official evaluation is conducted in hidden simulated environments. The organizers run the submitted systems under the official evaluation configuration and collect scores, logs, and structured model files. As shown in Fig. 2, qualification results determine finalists, and finalists submit a frozen version for official final testing. Hidden environments follow the announced task semantics and evaluation protocol, while scene layouts, instance parameters, exception triggers, and scoring seeds are not released.

The runtime permits inference, persistent memory update, local planning, and lightweight online adaptation within the official time budget. It prohibits internet access, online model APIs, hidden-ground-truth access, manual intervention, large-scale retraining, and redistribution or reverse engineering of hidden assets. These restrictions preserve a clear boundary between offline preparation and official testing.

### C. Structured Model Files

Each submitted system must expose its internal model through structured files. Track 1 submits `world_model.json` and `episode_log.jsonl`.

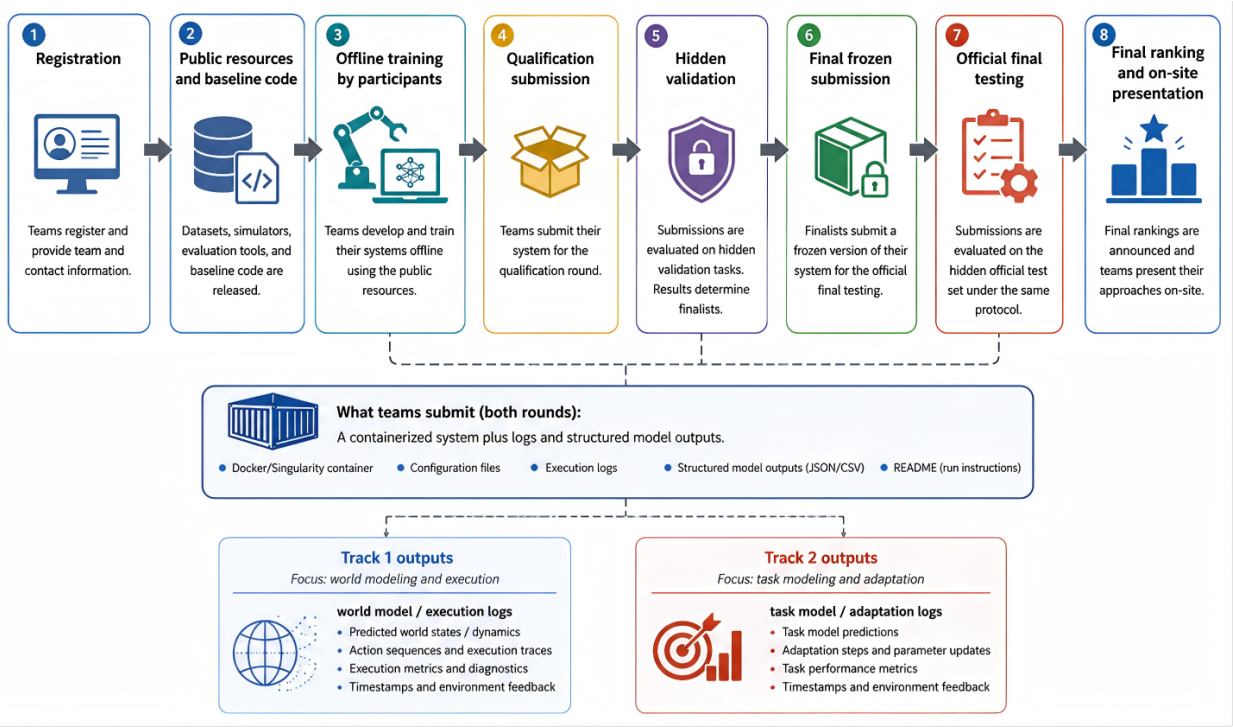


Fig. 2. Evaluation workflow and submission protocol. Participants train offline with public resources and submit runnable containers, logs, and structured model outputs for qualification and final evaluation. Track 1 requires world-model and execution logs, while Track 2 requires task-model and adaptation logs.

TABLE I
RECOMMENDED PUBLIC RESOURCES AND SUGGESTED ROLES. THESE RESOURCES ARE USED FOR TRAINING AND TASK-DESIGN REFERENCE. OFFICIAL RANKING IS COMPUTED ON HIDDEN ENVIRONMENTS AND HIDDEN INSTANCES.

| Track | Public Resource | Main Capability | Recommended Role |
|---|---|---|---|
| Track 1 | ProcTHOR / ProcTHOR-10K | Procedural indoor environments and cross-scene generalization | Primary public training environment and procedural reference |
| | Habitat Challenge / ReplicaCAD | Mobile manipulation and rearrangement protocol | Reference for evaluation setting and hidden-test protocol |
| | ALFRED | Language instructions and long-horizon household tasks | Task-template and instruction-design reference |
| | BEHAVIOR-1K | Everyday activities and rich object interactions | High-difficulty reference for future or stress settings |
| Track 2 | RLBench | Manipulation task families and demonstrations | Primary public training set and baseline platform |
| | MimicGen | Demonstration expansion and task-variant generation | Data-generation tool for one-shot/few-shot adaptation |
| | LIBERO | Object, goal, and spatial transfer suites | Reference for controlled generalization splits |
| | ManiSkill2 | Object variation and physical interaction | Resource for geometry and contact-rich variations |
| | RoboCasa / RoboCasa365 | Household manipulation and diverse kitchen scenes | Extended resource for future editions |

Track 2 submits `task_model.json` and `episode_log.jsonl`. The evaluator may inspect these files during post-evaluation review to check temporal consistency, topology and object-state correctness, affordance or precondition prediction, recovery traces, and agreement between model files and action logs.

## III. TRACK 1: UNSEEN-ENVIRONMENT EXPLORATION AND CLOSED-LOOP TASK EXECUTION

Track 1 evaluates whether an agent can construct and use an executable world model in a partially unseen indoor environment. As shown in Fig. 1(a) and Fig. 3, the task requires the agent to reach the final state through exploration, modeling, planning, execution, model update, and exception recovery in a single closed-loop run.

TABLE II
OFFICIAL PROCEDURE FOR TRACK 1.

| Stage | Time Budget | Requirement |
|---|---|---|
| Autonomous exploration | 20 min | Build topology, object states, and queryable model |
| Task reception and planning | 10 min | Parse instruction and generate subgoals and paths |
| Autonomous execution | 60 min | Execute, update the model, replan, and finish the task |

### A. Task Setting and Procedure

Each Track 1 episode lasts 90 minutes and contains three stages. In the exploration stage, the system discovers room connectivity, key objects, interactive states, and a queryable world model. In the task reception stage, it receives a natural-language instruction and generates an execution plan from its own model. In the execution stage, it completes the task while updating the model and recovering from a controlled exception. The visual sequence in Fig. 3 summarizes this

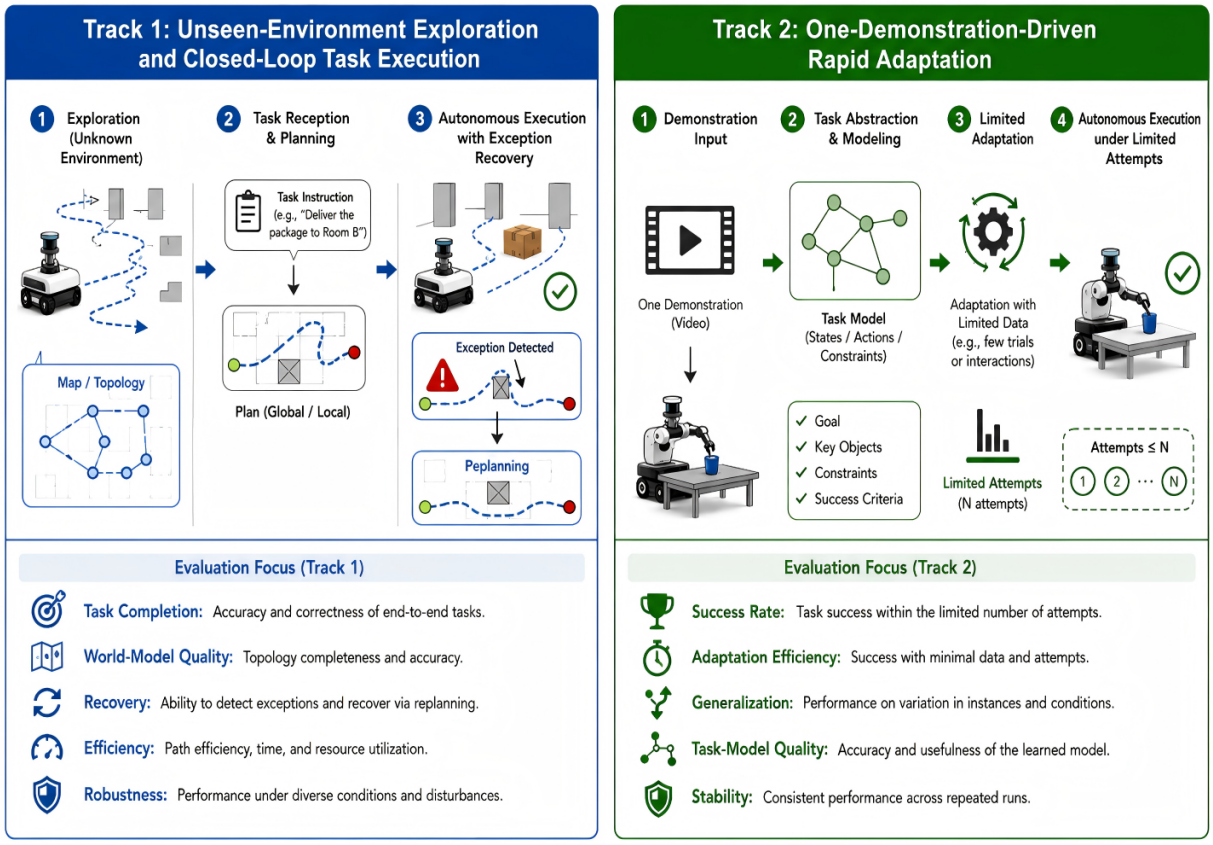


Fig. 3. Track definitions and evaluation focus. Track 1 contains exploration, task reception and planning, and autonomous execution with exception recovery. Track 2 contains demonstration input, task abstraction, limited adaptation, and autonomous execution under limited attempts.

TABLE III
CONTROLLED EXCEPTIONS IN TRACK 1.

| Exception Type | Example | Capability Evaluated |
|---|---|---|
| Door-lock change | A planned path becomes blocked | Topology update and replanning |
| Object relocation | A target is moved to a backup area | Object-state update and retrieval |
| Container unavailable | A planned container cannot be used | Functional substitute reasoning |
| Tool substitution | A specified tool is unavailable | Affordance modeling and correction |

exploration–planning–execution protocol.

For Track 1, each hidden episode is evaluated as one continuous run over the prescribed exploration, planning, and execution stages. Unrestricted resets, repeated trial-and-error exploration, and manual intervention are not allowed.

### B. Controlled Exception Library

Track 1 uses a fixed exception library to keep the first edition controlled and reproducible. At most one exception is triggered in each official episode. Every exception preserves at least one recoverable solution path.

### C. Evaluation and Ranking

Track 1 uses five normalized scoring components, matching the evaluation focus in Fig. 3: task completion, world-model quality, exception recovery and replanning, execution efficiency, and robustness and safety. The episode score is

$$S_{\mathrm{T1}} = 0.40S_{\mathrm{task}} + 0.20S_{\mathrm{model}} + 0.20S_{\mathrm{rec}} + 0.10S_{\mathrm{eff}} + 0.10S_{\mathrm{safe}}. \quad (1)$$

Task completion measures whether the main objective and key subgoals are completed. World-model quality checks topology, object states, affordances, and queryability. Exception recovery measures detection, model update, and valid alternative planning. Efficiency and safety penalize redundant actions, invalid operations, deadlocks, and unstable behavior.

When hidden standard and hidden challenge splits are both used, the final score is

$$S_{\mathrm{final}} = 0.6\,\mathrm{mean}(S_{\mathrm{hidden\_std}}) + 0.4\,\mathrm{mean}(S_{\mathrm{hidden\_ch}}). \quad (2)$$

If only one hidden split is used, the final score is the arithmetic mean over all hidden episodes. The leaderboard reports total score, task success rate, model-verification score, recovery score, invalid actions, and average runtime.

### D. Rules

Teams may train with public environments, private data, synthetic data, and pretrained models, but must disclose all resources. During official evaluation, the system may use inference, memory updates, planning, and lightweight online

TABLE IV
PRELIMINARY TRACK 1 BASELINE RESULTS. "STD." AND "CH." DENOTE STANDARD AND CHALLENGE-STYLE VALIDATION EPISODES.

| System | Std. Score | Ch. Score | Final Score | Task Succ. (%) | Model Verif. (%) | Recovery (%) | Invalid Act./Ep. ↓ | Latency (ms) ↓ | Peak Mem. (GB) ↓ |
|---|---|---|---|---|---|---|---|---|---|
| Random/scripted sanity | 10.6 | 6.8 | 9.1 | 2.5 | 10.0 | 0.0 | 27.4 | 16 | 0.7 |
| Heuristic map planner | 32.8 | 23.9 | 29.2 | 27.8 | 44.0 | 13.5 | 13.6 | 38 | 1.1 |
| Memory-free policy | 33.5 | 24.2 | 29.8 | 31.6 | 20.5 | 8.8 | 12.4 | 68 | 4.8 |
| PathDreamer-style planner | 40.6 | 30.8 | 36.7 | 34.8 | 35.0 | 18.2 | 10.7 | 360 | 8.5 |
| Structured WM + replanning | 48.2 | 36.5 | 43.5 | 41.0 | 58.0 | 31.5 | 8.4 | 520 | 10.2 |

TABLE V
OFFICIAL PROCEDURE FOR TRACK 2.

| Stage | Time Budget | Requirement |
|---|---|---|
| Demonstration reception | 5 min | Receive one standardized expert demonstration |
| Task parsing and adaptation | 15–20 min | Infer state variables, constraints, and policy updates |
| Autonomous execution | 25–30 min | Execute the task on the hidden instance |

adaptation only. It may not access hidden ground truth, internet services, online model APIs, remote human control, or hidden asset files. Ties are resolved by task success rate, world-model verification score, recovery score, invalid action count, and runtime.

### E. Baselines

Track 1 uses five reference baselines. The random/scripted sanity baseline checks whether submission, logging, and scoring interfaces operate correctly. The heuristic map planner stores discovered connectivity and object locations, then uses rule-based decomposition and shortest-path planning. The memory-free policy uses the current observation and instruction without persistent model memory. The PathDreamer-style planner uses visual look-ahead prediction for unvisited viewpoints [38]. The structured world-model baseline augments look-ahead with a queryable topology, object inventory, affordance table, and exception-triggered replanning. Preliminary results are shown in Table IV. The results show that explicit state maintenance and replanning improve the baseline score, while the reference systems still leave substantial room for participating teams.

## IV. TRACK 2: ONE-DEMONSTRATION-DRIVEN RAPID ADAPTATION

Track 2 evaluates whether a system can infer a transferable task model from one expert demonstration and execute an unseen instance under limited attempts. As shown in Fig. 1(b) and Fig. 3, the focus is not trajectory reproduction alone, but task abstraction, state-transition modeling, action-precondition reasoning, and rapid adaptation.

### A. Task Setting and Procedure

Each Track 2 episode lasts 50–60 minutes. The system first receives one standardized expert demonstration, then parses the task and adapts within the official budget, and finally executes the task on a hidden instance. This demonstration-to-new-instance setting corresponds to Fig. 1(b). Teams may make no more than three official attempts per instance.

The demonstration packet may include video, RGB-D observations, action trajectories, state-change annotations, or key-stage labels. Hidden instances vary in geometry, pose, friction, tool availability, and goal parameters.

TABLE VI
TASK FAMILIES FOR TRACK 2.

| Task Family | Representative Requirement |
|---|---|
| Insertion | Insert parts with unseen shape, pose, or clearance |
| Screwing/rotation | Operate knobs, nuts, valves, or rotational mechanisms |
| Assembly | Complete 2–3 step part combination and alignment |
| Opening/closing | Operate lids, latches, bolts, or sliding covers |
| Two-step tool use | Retrieve a tool and then tighten, pry, press, or align |

### B. Task Families

The first edition uses constrained task families to keep the evaluation implementable. Table VI lists the families and representative requirements.

### C. Evaluation and Ranking

Track 2 evaluates each hidden instance by a weighted sum of five normalized components:

$$S_{\mathrm{T2}} = 0.45S_{\mathrm{succ}} + 0.20S_{\mathrm{adapt}} + 0.15S_{\mathrm{gen}} + 0.10S_{\mathrm{model}} + 0.10S_{\mathrm{stab}}. \tag{3}$$

where $S_{\mathrm{succ}}$, $S_{\mathrm{adapt}}$, $S_{\mathrm{gen}}$, $S_{\mathrm{model}}$, and $S_{\mathrm{stab}}$ denote task success, rapid adaptation, generalization, task-model quality, and execution stability, respectively. The final ranking is based on the average score over all hidden instances.

### D. Rules

Teams may use public demonstrations, private demonstrations, synthetic data, and pretrained models for offline training. During evaluation, the system receives only the official demonstration packet for the hidden instance. It may adapt within the time budget but may not call online services, inspect hidden parameters, or perform large-scale retraining. The submitted system must output `task_model.json` and `episode_log.jsonl`. Ties are resolved by success rate, task-model verification score, average attempts, adaptation time, and runtime.

### E. Baselines

Track 2 uses five reference baselines. The T-OSVI-style imitation baseline uses a one-shot visual imitation policy without an explicit world model. The AWDA-style waypoint baseline predicts waypoints from the demonstration. The demo-parser baseline segments stages and calls scripted controllers. The OSVI-WM-style trajectory baseline follows

TABLE VII
PRELIMINARY TRACK 2 BASELINE RESULTS. SCORES ARE AVERAGED OVER TASK FAMILIES. ADAPTATION TIME EXCLUDES SIMULATOR STEPPING.

| System | Insertion | Rotation | Assembly | Open/Close | Tool Use | Avg. Score | Succ. (%) | Task Model (%) | Attempts ↓ | Adapt. Time (min) ↓ |
|---|---|---|---|---|---|---|---|---|---|---|
| T-OSVI-style imitation | 31.4 | 34.8 | 22.6 | 36.5 | 18.2 | 28.7 | 25.0 | 12.0 | 2.8 | 2.1 |
| AWDA-style waypoint | 35.2 | 37.9 | 27.5 | 40.8 | 21.3 | 32.5 | 30.5 | 15.0 | 2.6 | 2.4 |
| Demo-parser + controllers | 42.6 | 39.7 | 31.0 | 44.5 | 25.2 | 36.6 | 34.0 | 46.0 | 2.4 | 3.8 |
| OSVI-WM-style trajectory | 53.5 | 49.0 | 40.4 | 55.8 | 36.7 | 47.1 | 44.5 | 32.0 | 2.1 | 4.6 |
| OSVI-WM + task model | 57.8 | 52.4 | 43.6 | 59.5 | 40.6 | 50.8 | 48.0 | 55.0 | 1.9 | 5.5 |

TABLE VIII
SUBMISSION PACKAGE.

| Item | Content |
|---|---|
| Executable system | Runnable container with models and runtime dependencies |
| Structured files | `world_model.json` or `task_model.json`, plus logs |
| Technical report | Method, model design, training resources, compute, failures |
| Resource disclosure | Public/private/synthetic data, pretrained models, dependencies |
| Open-source statement | Optional code-release and reproducibility information |

world-model-guided latent trajectory prediction [39]. The OSVI-WM plus task-model baseline adds explicit stage, constraint, and success-condition outputs. Preliminary results are reported in Table VII. The results indicate that trajectory-level world modeling and explicit task-structure outputs improve the baseline score, while the reference systems remain moderate lower bounds for the challenge.

## V. COMMON RULES AND SUBMISSION REQUIREMENTS

This section summarizes rules shared by both tracks. The shared submission and evaluation flow is shown in Fig. 2.

### A. Participation Format

Each team submits an executable runnable container, model weights, run instructions, dependency list, structured model files, technical report, training-resource disclosure, and optional open-source statement. The container must include the inference system, memory update module, online adaptation module, and planner or policy module. Code release is not mandatory, but it is encouraged for reproducibility and special awards.

### B. Training and Evaluation Boundary

Participants may train models before the competition with public resources, private data, synthetic data, and pretrained models. All data sources, model sources, compute resources, and external dependencies must be declared. During official evaluation, only inference, memory updates, local planning, and lightweight online adaptation are allowed. Large-scale retraining, internet access, online APIs, manual control, and hidden-ground-truth access are prohibited.

### C. Submission Package

Table VIII lists the required files. These files correspond to the container, logs, and structured outputs shown in Fig. 2. A submission that fails to run, does not output required files, or accesses disallowed information may be rejected or rerun.

TABLE IX
IMPORTANT DATES OF EAGC 2026 WORLD MODEL CHALLENGE.

| Milestone | Date |
|---|---|
| Registration opens | May 8, 2026 |
| Dataset and code release | May 19, 2026 |
| Result submission channel opens | May 30–June 5, 2026 |
| Baseline paper | June 5, 2026 |
| Result testing period | June 6–June 9, 2026 |
| Registration deadline | June 15, 2026 |
| Environment v1.0 freeze | June 16, 2026 |
| Qualification submission | June 17–June 26, 2026 |
| Hidden validation test | June 27–July 3, 2026 |
| Finalists announced | July 5, 2026 |
| Final frozen submission | July 8–July 15, 2026 |
| Official final testing | July 16–July 24, 2026 |
| Log audit, rerun, and reproducibility check | July 25–July 28, 2026 |
| Final ranking released to teams | July 29, 2026 |
| Public results and on-site final presentation | Early August 2026 |

### D. Evaluation Integrity

Official scores are determined by the hidden checker. The evaluator may inspect model files and action logs to verify consistency. Top-ranked teams may undergo replay checks, log audits, and reruns. The final report should include total score, success rate, model-verification score, stability violations, latency, peak memory, and training-resource disclosure.

## VI. IMPORTANT DATES

Table IX lists the challenge timeline. The milestones follow the workflow in Fig. 2. Deadlines follow 23:59 Anywhere on Earth (AoE), unless otherwise specified.

## VII. CONCLUSIONS

This paper introduced the EAGC 2026 World Model Challenge, a simulation-based competition for evaluating executable world models in embodied intelligence. The challenge contains two complementary tracks: unseen-environment exploration with closed-loop task execution, and one-demonstration-driven rapid adaptation. Both tracks evaluate final task outcomes together with structured model outputs, temporal updates, recovery behavior, and runtime efficiency. By separating public training resources from hidden official tests, EAGC 2026 provides a controlled benchmark for comparing world-model algorithms under reproducible conditions.

## REFERENCES

[1] D. Ha and J. Schmidhuber, "Recurrent world models facilitate policy evolution," in *Advances in Neural Information Processing Systems*, 2018.

[2] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *Proc. Int. Conf. Machine Learning*, 2019.

[3] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *Proc. Int. Conf. Learning Representations*, 2020.

[4] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering Atari with discrete world models," in *Proc. Int. Conf. Learning Representations*, 2021.

[5] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104*, 2023.

[6] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature*, vol. 588, pp. 604–609, 2020.

[7] K. Gregor, D. J. Rezende, F. Besse, Y. Wu, H. Merzic, and A. van den Oord, "Shaping belief states with generative environment models for reinforcement learning," in *Advances in Neural Information Processing Systems*, 2019.

[8] R. Sekar, O. Rybkin, K. Pertsch, M. C. Machado, J. A. D. Bagnell, and S. Levine, "Planning to explore via self-supervised world models," in *Proc. Int. Conf. Machine Learning*, 2020.

[9] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: An interactive 3D environment for visual AI," *arXiv preprint arXiv:1712.05474*, 2017.

[10] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A platform for embodied AI research," in *Proc. IEEE/CVF Int. Conf. Computer Vision*, 2019, pp. 9339–9347.

[11] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: Real-world perception for embodied agents," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, 2018.

[12] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba, "VirtualHome: Simulating household activities via programs," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, 2018.

[13] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. S. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," in *Advances in Neural Information Processing Systems*, 2021.

[14] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, J. Salvador, K. Ehsani, W. Han, E. Kolve, A. Farhadi, A. Kembhavi, and R. Mottaghi, "ProcTHOR: Large-scale embodied AI using procedural generation," in *Advances in Neural Information Processing Systems*, 2022.

[15] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "ALFRED: A benchmark for interpreting grounded instructions for everyday tasks," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, 2020, pp. 10740–10749.

[16] C. Li, F. Xia, R. Martín-Martín, and S. Savarese, "BEHAVIOR-1K: A benchmark for embodied AI with 1000 everyday activities and realistic simulation," *arXiv preprint arXiv:2403.09227*, 2024.

[17] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "SAPIEN: A simulated part-based interactive environment," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, 2020.

[18] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "RLBench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[19] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su, "ManiSkill2: A unified benchmark for generalizable manipulation skills," *arXiv preprint arXiv:2302.04659*, 2023.

[20] 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," in *Proc. Conf. Robot Learning*, 2023.

[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] 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," *arXiv preprint arXiv:2406.02523*, 2024.

[23] S. Nasiriany, S. Nasiriany, A. Maddukuri, and Y. Zhu, "RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots," in *Proc. Int. Conf. Learning Representations*, 2026.

[24] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and where pathways for robotic manipulation," in *Proc. Conf. Robot Learning*, 2022.

[25] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan, "VIMA: General robot manipulation with multimodal prompts," in *Proc. Int. Conf. Machine Learning*, 2023.

[26] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, B. Ichter, A. Irpan, E. Jang, R. Jauregui Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng, "Do as I can, not as I say: Grounding language in robotic affordances," in *Proc. Conf. Robot Learning*, 2022.

[27] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence, "PaLM-E: An embodied multimodal language model," in *Proc. Int. Conf. Machine Learning*, 2023.

[28] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "RT-1: Robotics transformer for real-world control at scale," in *Robotics: Science and Systems*, 2023.

[29] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Quiambao, K. Rao, M. Ryoo, P. Sermanet, A. Singh, J. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, B. Zitkovich, and A. Zeng, "RT-2: Vision-language-action models transfer web knowledge to robotic control," in *Proc. Conf. Robot Learning*, 2023.

[30] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter, "Inner monologue: Embodied reasoning through planning with language models," in *Proc. Conf. Robot Learning*, 2022.

[31] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, "Voyager: An open-ended embodied agent with large language models," *arXiv preprint arXiv:2305.16291*, 2023.

[32] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-Actor: A multi-task transformer for robotic manipulation," in *Proc. Conf. Robot Learning*, 2023.

[33] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2023.

[34] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "ProgPrompt: Generating situated robot task plans using large language models," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2023.

[35] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "VoxPoser: Composable 3D value maps for robotic manipulation with language models," in *Proc. Conf. Robot Learning*, 2023.

[36] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. S. Ryoo, A. Stone, and D. Kappler, "Open-vocabulary queryable scene representations for real world planning," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2023.

[37] T. Zhang, M. Hu, C. Xia, and Y. Zhu, "GR-1: Unleashing large-scale video generative pre-training for visual robot manipulation," *arXiv preprint arXiv:2312.13139*, 2023.

[38] J. Y. Koh, H. Lee, Y. Yang, J. Baldridge, and P. Anderson, "PathDreamer: A world model for indoor navigation," in *Proc. ICCV*, 2021.

[39] R. G. Goswami, P. Krishnamurthy, Y. LeCun, and F. Khorrami, "OSVI-WM: One-shot visual imitation for unseen tasks using world-model-guided trajectory generation," in *Proc. NeurIPS*, 2025.