ADAPTATION-READY JEPA: SOURCE TRAINING FOR TEST-TIME ADAPTIVE WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent world models often require test-time adaptation under deployment shifts, while existing methods mainly rely on self-supervised objectives that can be constructed directly at deployment. In contrast, supervision that explicitly shapes control-related representations, such as physical states, is often available only during source training. Such source-only supervision can improve the initial representation, but its constraints do not automatically carry over once the model is further updated by a different self-supervised objective. We therefore study a supervision-availability mismatch under a fixed deployment update: *how can training-only supervision improve the model produced by that update?* We propose **Adaptation-Ready JEPA (AR-JEPA)**, which keeps AdaJEPA's deployment update unchanged, differentiably simulates it during source training, and optimizes the source model through the resulting adapted model. A meta-learning-style post-update physical objective alone is insufficient in this setting, because the adapted object is itself the world model that must continue to support dynamics prediction and MPC. AR-JEPA therefore uses Post-Phys to preserve control-related physical information in the updated representation and Post-Dyn to preserve future latent-dynamics prediction of the same model, jointly maintaining*representation content* and *predictive functionality* after adaptation. After training, physical labels and the auxiliary head are removed, while real deployment retains AdaJEPA's original adaptation loss, updatable parameters, and MPC procedure. On Push-Object, AR-JEPA achieves the highest 13-OOD adapted success among the compared configurations (40.56%) and shows smaller future-dynamics and physical-representation degradation under an identical AdaJEPA update. PointMaze further shows that the relative roles of pre- and post-adaptation physical supervision depend on the environment.

1 INTRODUCTION

World models need to keep adapting their representations and predictions under deployment shifts. Recent test-time training/adaptation (TTT/TTA) methods mainly rely on self-supervised signals that can be reconstructed from deployment data. For example, MoVie adapts visual representations using interaction-based dynamics consistency, AdaPower drives test-time updates through spatiotemporal feature reconstruction, and WAM-TTT updates fast-weight memory through self-supervised video prediction (Yang et al., 2023; Huang et al., 2025b; Feng et al., 2026). Although these methods target different models and tasks, they share a common assumption: their test-time objectives can be constructed directly from deployment data.

However, useful supervision for world-model representation learning is not always available at deployment. In JEPA world models, the latent representation supports both future prediction and planning, so the information it retains directly affects control performance (Zhou et al., 2025; Terver et al., 2026). Prior work therefore uses additional sensing, depth, or physical states available during training to explicitly constrain this representation (Hu et al., 2024; Khan, 2026; Yan et al., 2026). Such training-only supervision can specify what the representation should retain, but cannot remain an adaptation objective once its labels are unavailable at deployment. More importantly, **when another objective continues to update the model after deployment, a representation constraint learned during source training does not automatically constrain the updated model.** We there-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

fore study a problem complementary to existing TTA methods: **when the deployment update is fixed, how can training-only supervision improve the world model produced by that update?**

We study this problem using physical supervision within AdaJEPA's closed-loop adaptation framework. AdaJEPA uses real interaction transitions and its native latent-prediction loss to update selected world-model parameters, and then replans with the updated model (Wang et al., 2026). In this setting, source-only physical supervision specifies which control-related information the model should retain, while real deployment adaptation is driven entirely by the latent-prediction objective. Therefore, directly combining ordinary physical representation learning with AdaJEPA still leaves a gap: **the former constrains only the model before the update, while the latter determines how the model changes afterward, and together they do not directly ensure that both control-related information and predictive ability are preserved after a prediction-based update.**

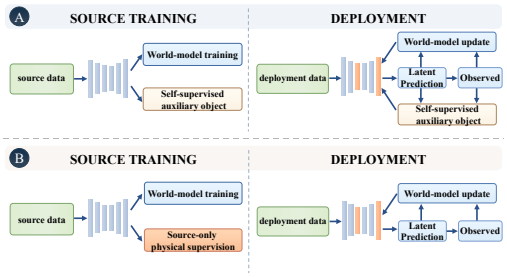


Figure 1: Availability of source supervision at deployment. (A) Self-supervised auxiliary objectives can be reconstructed from deployment data and used during both source training and test-time updates. (B) Source-only physical supervision is unavailable at deployment and therefore cannot directly enter later world-model updates. This difference creates the supervision-availability mismatch studied in this paper.

To make training-only supervision constrain the model produced by the fixed deployment update, we follow the gradient-based meta-learning principle of differentiating through a simulated parameter update (Finn et al., 2017; Bartler et al., 2022). This principle is not new: MT3, for example, optimizes an initialization through a supervised objective evaluated after self-supervised adaptation (Bartler et al., 2022). Our setting differs in what must be preserved after the update: the adapted object is itself a world model that must continue to support dynamics prediction and MPC. Therefore, post-update physical supervision alone constrains the representation content, but does not directly constrain the dynamics prediction of the updated model, motivating an additional post-update dynamics objective.

We therefore propose **Adaptation-Ready JEPA (AR-JEPA)**, which adds source-only physical supervision to base world-model training and differentiably simulates AdaJEPA's fixed deployment update. Post-Phys constrains the updated representation to retain control-related information, while Post-Dyn preserves future latent-dynamics prediction, targeting **representation content** and **predictive functionality**, respectively. After source training, the physical labels and auxiliary head are removed, and real deployment retains AdaJEPA's original update rule and MPC procedure. Thus, AR-JEPA changes **source training rather than deployment adaptation**.

We evaluate AR-JEPA on Push-Object and PointMaze from both closed-loop performance and update behavior. On Push-Object, AR-JEPA achieves the highest 13-OOD adapted performance among the compared configurations and exhibits smaller degradation in future dynamics and physical representation under an identical AdaJEPA update. Factorized experiments further show that the effects of pre- and post-adaptation physical supervision vary across environments.

Our main contributions are as follows:

- We study a *supervision-availability mismatch* in adaptive world models: supervision that specifies representation content may be available only during source training, while deployment adaptation must be driven by a different self-supervised objective. We further highlight that a source-stage representation constraint does not automatically persist in a model updated by another objective.

- We propose **AR-JEPA**. With the AdaJEPA deployment update kept fixed, AR-JEPA simulates this update during source training and uses Post-Phys and Post-Dyn to constrain the updated model's *representation content* and *predictive functionality*, respectively. This allows training-only supervision to directly benefit the world model produced by the future fixed update.

- We validate the design from both final control performance and identical-update response. AR-JEPA achieves the highest adapted performance on Push-Object and exhibits smaller future-dynamics and physical-representation degradation under the same AdaJEPA update. Factorized analyses across Push-Object and PointMaze further reveal the different roles of Pre-Phys, Post-Phys, and Post-Dyn.

2 Related Work

2.1 Test-Time Adaptation of World Models

World models trained on a fixed source distribution can become inaccurate when observations or environment dynamics shift at deployment (Deng et al., 2022; Jing & Boularias, 2025). Recent work therefore adapts world models online using signals available from deployment data (Nagabandi et al., 2019; Lee et al., 2020b; Seo et al., 2020). MoVie and AdaPower perform test-time updates using interaction-based dynamics consistency and feature reconstruction, respectively (Yang et al., 2023; Huang et al., 2025b). WorldAgen and AdaJEPA instead continue optimizing world models from real transitions using prediction objectives (Wan et al., 2026; Wang et al., 2026). These methods primarily design the deployment-time adaptation mechanism. AR-JEPA keeps AdaJEPA's update rule, updatable parameters, and MPC procedure fixed, and studies the complementary source-training problem of using training-only supervision to improve the model produced by that update.

2.2 Training-Time Privileged Supervision

Training-only privileged information is commonly used either to shape deployable representations or policies, or to recover hidden environment context at deployment (Hu et al., 2024; Huang et al., 2025a; Day et al., 2021; Byrd et al., 2025; Yu et al., 2026; Kumar et al., 2021; Miki et al., 2022; Loquercio et al., 2021; Lee et al., 2020a; Margolis et al., 2022; Baisero & Amato, 2022; Lambrechts et al., 2025). AR-JEPA does not recover physical variables at test time or use auxiliary predictions as world-model inputs. Instead, it studies the relation between privileged supervision and a subsequent parameter update: given a fixed deployment update, how can source-only physical supervision also constrain the model produced by that update?

2.3 Meta-Trained Test-Time Adaptation

Meta-training future test-time updates has clear precedents. MT3 combines a self-supervised inner update with a labeled objective on the adapted model, while Meta-Auxiliary Learning similarly trains reconstruction-driven updates to benefit downstream tasks (Bartler et al., 2022; Chi et al., 2021; Liu et al., 2023). WAM-TTT extends this principle to world-action models by updating fast-weight memory through video prediction and meta-training the updated memory for action generation (Feng et al., 2026). Thus, AR-JEPA does not claim novelty in the general meta-TTA principle. Its distinction is that the adapted object is itself the JEPA world model used for subsequent dynamics prediction and MPC. Post-Phys constrains the information retained by the adapted representation, but does not directly constrain predictive dynamics. AR-JEPA therefore adds Post-Dyn, and its factorized experiments compare Meta-Phys and Pre+Meta-Phys with their Post-Dyn counterparts to test whether generic post-adaptation supervision alone explains the full method.

3 Method

3.1 Problem Setting and the Fixed Deployment Update

We build on AdaJEPA's latent world model and closed-loop test-time adaptation framework (Wang et al., 2026). Let E_ϕ be the visual encoder and F_ψ the dynamics predictor. For compact notation, we write

$$z_t = E_\phi(o_t), \qquad \hat{z}_{t+1} = F_\psi(z_t, a_t), \tag{1}$$

while the actual implementation conditions on a short observation history and its action sequence. We denote the world-model parameters by $w = (\phi, \psi)$.

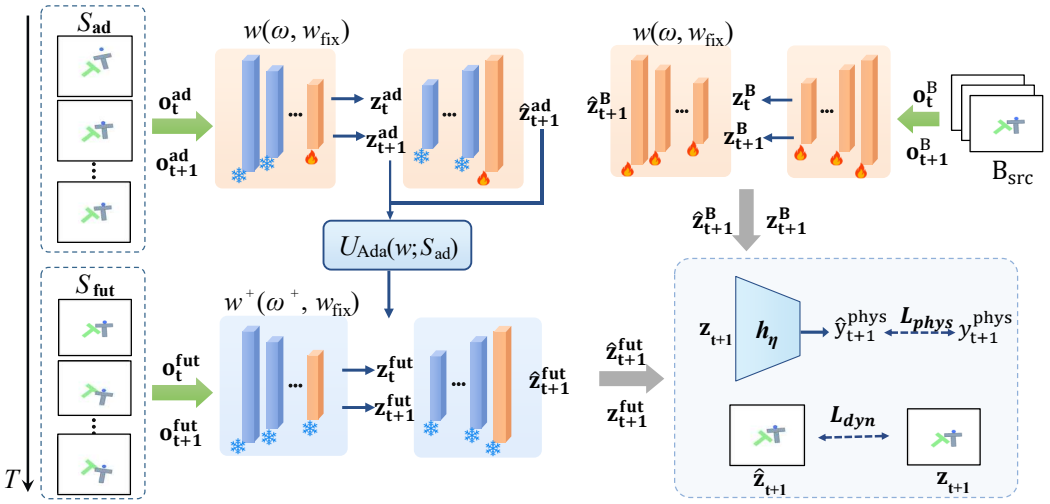


Figure 2: Source training of AR-JEPA. B_{src} is used for base world-model training and Pre-Phys. S_{ad} applies the fixed AdaJEPA update U_{Ada} to obtain the temporary model w^+, which is then evaluated on the later S_{fut} using Post-Phys and Post-Dyn. For visual simplicity, the figure writes the base source objective as $L_{\text{src}} = L_{\text{dyn}}$.

For a data collection $\mathcal{Q}$, the latent-prediction objective is

$$\mathcal{L}_{\text{pred}}(w; \mathcal{Q}) = \mathbb{E}_{(o_t, a_t, o_{t+1}) \sim \mathcal{Q}} \left[\ell_{\text{lat}} \left(\hat{z}_{t+1}(w), \text{sg}[z_{t+1}(w)] \right) \right], \tag{2}$$

where the target latent is encoded from the real successor observation and stopped from gradient propagation. Base source training follows AdaJEPA and is denoted by $\mathcal{L}_{\text{src}}(w; \mathcal{B}_{\text{src}})$.

We study source training under a fixed deployment update. Write $w = (\omega, w_{\text{fix}})$, where only ω is updated at test time. Given adaptation data $\mathcal{S}_{\text{ad}}$, AdaJEPA produces

$$w^+ = \mathcal{U}_{\text{Ada}}(w; \mathcal{S}_{\text{ad}}), \tag{3}$$

using its original latent-prediction loss, trainable parameter subset, and optimizer. We keep $\mathcal{U}_{\text{Ada}}$ unchanged and instead train the source model so that the resulting w^+ retains control-related information and effective predictive dynamics.

3.2 Source-Only Physical Representation Supervision

We use control-related physical information as training-only supervision because it directly specifies what the visual representation should retain. Let h_η be the physical auxiliary head and y_t^{phys} the corresponding target. Using the notation in Eq. (1), we define

$$\mathcal{L}_{\text{phys}}(w, \eta; \mathcal{Q}) = \mathbb{E}_{(o_t, y_t^{\text{phys}}) \sim \mathcal{Q}} \left[\ell_{\text{phys}} \left(h_\eta(z_t), y_t^{\text{phys}} \right) \right], \qquad z_t = E_\phi(o_t). \tag{4}$$

Push-Object uses spatial occupancy of the agent and manipulated object, while PointMaze uses position and velocity. Benchmark-specific inputs, targets, and auxiliary heads are detailed in Appendix A.2.

Ordinary source-only supervision is $\mathcal{L}_{\text{phys}}^{\text{pre}} = \mathcal{L}_{\text{phys}}(w, \eta; \mathcal{B}_{\text{src}})$. It constrains the source representation before adaptation, but not the representation produced after the deployment update. AR-JEPA therefore explicitly evaluates the updated model during source training.

3.3 Source Training for the Fixed Deployment Update

From each source sequence, we construct $\mathcal{S} = (\mathcal{S}_{\text{ad}}, \mathcal{S}_{\text{fut}})$. The former simulates the deployment update, while the temporally later $\mathcal{S}_{\text{fut}}$ evaluates its outcome. We first obtain $w^+ = \mathcal{U}_{\text{Ada}}(w; \mathcal{S}_{\text{ad}})$ using exactly the same AdaJEPA update as at deployment. Physical labels, the auxiliary head, rewards, goals, planning costs, and future targets do not enter this inner update.

Post-adaptation physical representation. To constrain what remains in the representation after adaptation, we use $\mathcal{L}_{\text{phys}}^{\text{post}} = \mathcal{L}_{\text{phys}}(w^+, \eta; \mathcal{S}_{\text{fut}})$. Thus, training-only physical supervision can shape the representation produced by the future deployment update without becoming a test-time loss.

Post-adaptation dynamics prediction. Preserving physical information does not directly preserve world-model prediction. Since the adapted model must still predict action outcomes and support MPC, we additionally define

$$\mathcal{L}_{\text{dyn}}^{\text{post}} = \frac{\mathcal{L}_{\text{pred}}(w^+; \mathcal{S}_{\text{fut}})}{\text{sg}[V_{\text{fut}}] + \epsilon}, \tag{5}$$

where V_{fut} is the variance of future target latents and is detached together with the target branch.

Post-Phys and Post-Dyn therefore target two complementary properties of the adapted model: *representation content* and *predictive functionality*.

3.4 AR-JEPA Source Training Objective

The full AR-JEPA objective is

$$\mathcal{L}_{\text{AR}} = \mathcal{L}_{\text{src}} + \lambda_{\text{pre}}\mathcal{L}_{\text{phys}}^{\text{pre}} + \lambda_{\text{post}}\mathcal{L}_{\text{phys}}^{\text{post}} + \lambda_{\text{dyn}}\mathcal{L}_{\text{dyn}}^{\text{post}}. \tag{6}$$

The inner AdaJEPA loss is used only to construct w^+ and is not added as another outer objective. Outer optimization differentiates Eq. (6) through $w^+ = \mathcal{U}_{\text{Ada}}(w; \mathcal{S}_{\text{ad}})$, so the post-adaptation objectives directly shape the source parameters according to the model produced by the fixed update. Loss weights, auxiliary-head optimization, and training schedules are given in Appendices A.2–A.5.

The temporary parameters w^+ are discarded after each outer update. After source training, physical labels and the auxiliary head are removed, and real deployment reuses AdaJEPA's original observation interface, transition feedback, latent-prediction loss, trainable parameters, optimizer, and MPC planner. **AR-JEPA therefore changes source training only; it adds no test-time supervision and does not modify the deployment update.**Pre-Phys, Post-Phys, and Post-Dyn also define the three factors used in our experimental analysis.

4 Experiments

We study how source-training objectives affect a *fixed* AdaJEPA deployment update. All methods share the same world-model architecture, deployment adaptation rule, and planner; only source training differs. Reference configurations are V0, Auxiliary-only, AR-Dyn, Meta-Phys, and Pre+Meta-Phys, while AR-PreAux, AR-PostAux, and AR-JEPA form the joint Phys+Dyn family. We organize the experiments around five questions: final closed-loop performance, response to the same deployment update, the additional effect of Post-Dyn, the complementarity of Phys and Dyn, and the placement of physical supervision.

4.1 Experimental Setup

We follow AdaJEPA's closed-loop control protocol (Wang et al., 2026). All methods use the same CEM planner, deployment latent-prediction loss, updatable parameter subset, and optimizer. Physical labels and auxiliary heads are removed before deployment, so no extra test-time supervision is introduced.

We use two complementary evaluation regimes. **Push-Object** mainly evaluates visual shifts using 6 shape shifts, 4 visual corruptions, and 3 color shifts, and reports the equal-domain average over 13 OOD domains. Its physical supervision is relatively weak and indirect, using agent/object occupancy to constrain the visual representation. **PointMaze** evaluates Low Density and High Damping dynamics shifts and directly regresses position and velocity from visual representations, providing a stronger state-level physical constraint. Thus, Push and Maze differ in both shift type and physical-supervision strength; we treat them as complementary regimes rather than attributing their differences to a single factor.

Table 1: **Main results and factorized analysis.** Panel A reports Push-Object 13-OOD performance and source-training factors; Panel B reports cross-domain statistics and PointMaze Dynamics-OOD performance.

Panel A: Source-training factors and Push-Object performance

	Source objectives			Push-Object		
Method	Pre	Post	Dyn	Frozen	Adapted	Gain [95% CI]
Reference configurations						
V0	×	×	×	33.64	37.38	+3.74 [1.79, 5.69]
Auxiliary-only	✓	×	×	34.72	37.18	+2.46 [0.21, 4.72]
AR-Dyn	×	×	✓	34.67	38.36	+3.69 [−0.31, 7.13]
Meta-Phys	×	✓	×	35.00	38.20	+3.20 [1.05, 5.25]
Pre+Meta-Phys	✓	✓	×	35.55	39.35	+3.80 [1.75, 5.80]
Joint Phys+Dyn variants						
AR-PreAux	✓	×	✓	34.56	38.82	+4.26 [2.51, 6.00]
AR-PostAux	×	✓	✓	35.18	38.56	+3.38 [1.54, 5.23]
AR-JEPA	✓	✓	✓	**36.62**	**40.56**	+3.95 [2.21, 5.74]

Panel B: Cross-domain adaptation profile and PointMaze

Method	HNAG ↑	Gain SD ↓	WQ ↑	+Domains ↑	PointMaze F→A
Reference configurations					
V0	6.33	4.17	−1.64	9/13	70.67→74.33
Auxiliary-only	3.67	3.85	−2.36	9/13	92.67→91.67
AR-Dyn	6.32	4.10	−1.13	11/13	83.00→82.33
Meta-Phys	6.20	4.08	−0.90	10/13	84.67→85.67
Pre+Meta-Phys	6.90	3.96	−0.30	11/13	87.33→87.67
Joint Phys+Dyn variants					
AR-PreAux	6.78	4.47	+0.10	**12/13**	89.00→**92.00**
AR-PostAux	7.28	3.79	−1.74	10/13	85.67→87.67
AR-JEPA	**7.44**	**3.77**	**+0.26**	**12/13**	86.67→88.33

Each evaluation condition uses three seeds with 50 episodes per seed, and Frozen/Adapted evaluations use paired task instances. We report Frozen success S_0, Adapted success S_A, and $\Delta S = S_A - S_0$. Push additionally reports HNAG, Gain SD, WQ, and positive-domain count. HNAG measures normalized adaptation benefit, while Gain SD, WQ, and positive-domain count summarize the consistency of gains across OOD domains, which we call *cross-domain adaptation stability*.

For the identical-update diagnostic,

$$\Delta L = L(w^+) - L(w), \qquad D_L = \left|\overline{\Delta L}\right|.$$

The sign of ΔL gives the loss-change direction, while D_L gives the mean response magnitude. We use D_L to describe *update-response stability*: a smaller value means only that the mean objective change induced by the same update is closer to zero, not that closed-loop performance or parameter/trajectory stability is necessarily better.

4.2 Final Closed-Loop Performance

On Push-Object, AR-JEPA achieves the highest Adapted success (40.56%). It also has the highest HNAG (7.44), the lowest Gain SD (3.77), a positive WQ (+0.26), and positive gain on 12/13 domains. By comparison, V0 has Gain SD 4.17, WQ -1.64, and positive gain on 9/13 domains. Thus, AR-JEPA improves both final performance and the consistency of adaptation across OOD domains. AR-PreAux has a larger raw Frozen-to-Adapted gain (+4.26 pp), showing that raw gain, final Adapted performance, and cross-domain adaptation stability capture different properties.

PointMaze shows a different ordering: AR-PreAux reaches the highest Adapted success (92.00%). At the family level, however, AR-PreAux, AR-PostAux, and AR-JEPA all have positive gains on both benchmarks, whereas Auxiliary-only and AR-Dyn both decrease on Maze. Thus, the joint Phys+Dyn family does not yield a universally best placement, but shows favorable adaptation across both the weak/indirect-Phys visual-shift regime and the strong/direct-Phys dynamics-shift regime.

Task-level performance alone cannot reveal whether these differences come from better source initialization or from different responses to the fixed update. We test the latter next.

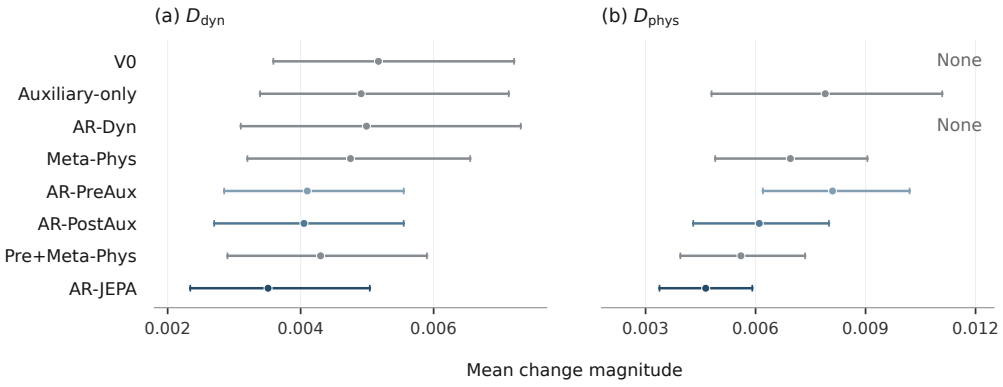


Figure 3: **Push-Object identical-update response.** Left: dynamics-response magnitude. Right: physical-response magnitude.

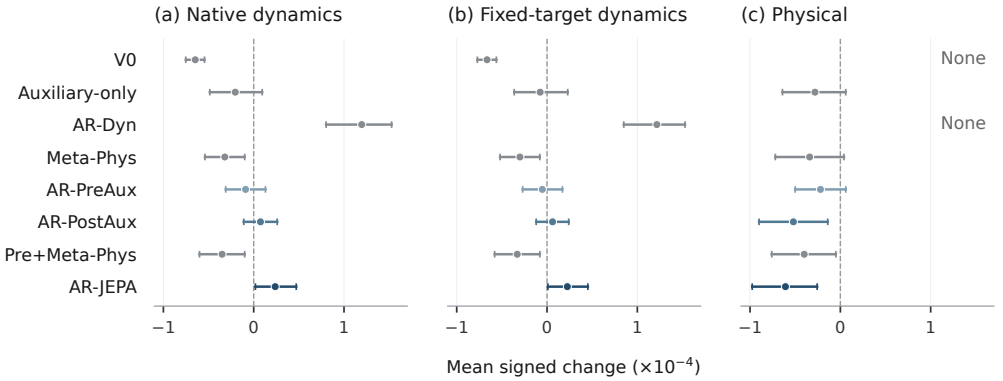


Figure 4: **PointMaze identical-update response.** Signed mean changes for native dynamics, fixed-target dynamics, and physical objectives. Distance to zero measures response magnitude; sign indicates direction.

4.3 Response to the Same Deployment Update

We fix adaptation data, updatable parameters, learning rate, and fresh Adam optimizer, and apply the same one-step AdaJEPA update to every checkpoint. Figures 3–4 show the model-level responses.

Push-Object. AR-JEPA has the smallest mean dynamics response and, among methods with a physical head, the smallest physical response. The partial variants and corresponding reference configurations lie above it. Thus, on Push, the highest final performance is accompanied by smaller mean dynamics and physical perturbations, consistent with higher update-response stability.

PointMaze. AR-PreAux and AR-PostAux have dynamics responses closest to zero, whereas AR-Dyn has the largest response. AR-JEPA lies between these cases but remains closer to zero than V0 and AR-Dyn. For the physical objective, Post-Phys variants have negative signed changes, indicating lower physical loss after the update.

Two observations follow. First, the same AdaJEPA update produces different responses depending on source training. Second, update-response stability is not equivalent to closed-loop performance: on Maze, the responses closest to zero do not yield the highest task success. We therefore use this diagnostic to characterize update behavior, not to define a new performance ranking.

Having established that source training changes fixed-update response, we next isolate the contribution of Post-Dyn.

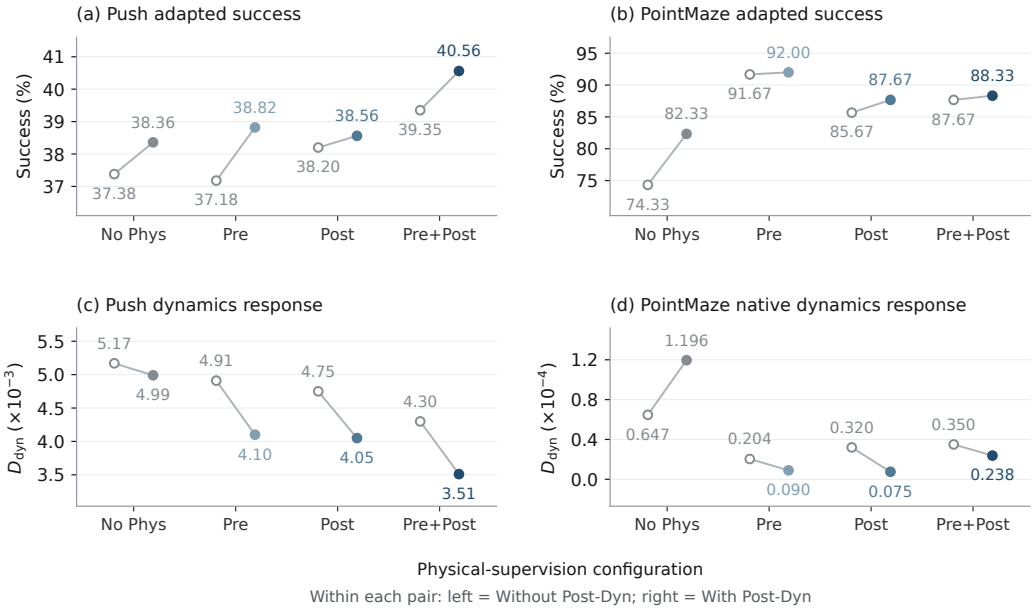


Figure 5: **Matched effect of Post-Dyn.** Each pair fixes physical supervision and varies Post-Dyn. Top: Adapted success. Bottom: dynamics-response magnitude.

4.4 Additional Effect of Post-Dyn

We include a set of controlled variants to disentangle the effects of physical supervision, its placement, and adaptation-aware optimization, while keeping the deployment update protocol fixed across methods.Each pair fixes the physical-supervision configuration and changes only whether Post-Dyn is enabled. Figure 5 summarizes the comparisons.

On Push, adding Post-Dyn increases the Adapted point estimate in all four pairs and moves the mean dynamics response closer to zero. Thus, in this regime, Post-Dyn improves both final performance and update-response stability.

On PointMaze, all four Adapted point estimates also increase, but the response depends on the representation constraint. With physical supervision, Post-Dyn moves the dynamics response closer to zero; without it, V0→AR-Dyn improves final success despite a larger immediate response.

Post-Dyn therefore does not simply "shrink" the update. It improves the task-level performance of the updated model, while its effect on update-response stability depends on the physical constraint already present in the source representation.

This dependence motivates treating dynamics and representation constraints jointly rather than as independent objectives.

4.5 Why Combine Phys and Dyn?

Phys-only and Dyn-only expose complementary limitations.

Physical supervision alone. On PointMaze, Auxiliary-only reaches 92.67% Frozen success, far above V0 at 70.67%, showing that strong/direct physical supervision can substantially improve the source representation. Yet its performance drops to 91.67% after adaptation. A better pre-update representation therefore does not automatically yield a more favorable future update response; optimizing the source representation and optimizing a model that will later be updated are different problems.

Dynamics constraint alone. AR-Dyn directly constrains post-update dynamics prediction and improves on Push, but drops from 83.00% to 82.33% on Maze. Post-update prediction alone is therefore insufficient, especially when the task depends more directly on physical state evolution.

The two objectives act at different points: **Phys constrains which control-related information is retained in the representation, while Post-Dyn constrains whether that representation still supports useful dynamics prediction after the fixed update.** All three joint variants maintain positive gains on both benchmarks, providing the main empirical support for this complementarity. The matched analysis further shows that Post-Dyn's effect on update-response stability depends on whether a physical representation constraint is already present.

With both constraints motivated, the remaining question is where physical supervision should be placed relative to the update.

4.6 Where Should Physical Supervision Be Applied?

Within the joint family, AR-PreAux, AR-PostAux, and AR-JEPA use Pre-Phys, Post-Phys, and Pre+Post, respectively.

In the **weak/indirect-Phys Push regime**, their Adapted success values are 38.82%, 38.56%, and 40.56%, so Pre+Post is highest. AR-JEPA also has the smallest Push dynamics and physical mean responses. Here, Pre+Post aligns higher task performance with higher update-response stability.

In the **strong/direct-Phys Maze regime**, the same methods reach 92.00%, 87.67%, and 88.33%, so final performance favors Pre-Phys. AR-PreAux and AR-PostAux have dynamics responses closest to zero, while Post-Phys variants show clearer reductions in physical loss. Again, response stability and closed-loop success are not interchangeable.

The two benchmarks therefore exhibit different placement preferences, consistent with their complementary regimes: Push uses weaker, indirect physical supervision, whereas Maze uses stronger, direct state supervision. We interpret this as *placement sensitivity across regimes*, rather than attributing the optimal placement to any single factor.

More generally, **Pre-Phys shapes the representation entering the deployment update, whereas Post-Phys constrains the representation produced by it.** Their relative value can vary with shift type and physical-supervision regime, making supervision placement a design dimension of fixed-update source training.

5 Conclusion

This paper studies a world-model adaptation problem caused by a supervision-availability mismatch. Control-related physical labels are available only during source training, while real deployment updates must use a different self-supervised prediction signal. The question is how these source-only physical labels can still improve the model after the update. AR-JEPA keeps AdaJEPA's deployment update, deployment-updatable parameter subset, and MPC procedure unchanged. It explicitly simulates the fixed update during source training. It trains the source model with post-adaptation dynamics prediction and physical representation objectives. The physical auxiliary head and physical labels are removed at deployment. No new test-time supervision is added.

Experiments show that AR-JEPA achieves the highest post-adaptation planning performance on the main Push-Object OOD evaluation. It also shows a favorable update profile across shifts. PointMaze shows that the relative value of pre/post physical supervision depends on the environment. These results support a more general view: for a world model that continues to update after deployment, source representation quality and the model's response to future updates should be considered together.

AI Usage Disclosure

In this work, we used generative AI tools to assist with literature retrieval and discovery, research ideation, drafting parts of the paper, and language editing and polishing. We reviewed and verified

all AI-assisted work, including retrieved references, research suggestions, and drafted or edited text. All final methodological choices, experimental results, analyses, and conclusions were determined and validated by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Andrea Baisero and Christopher Amato. Unbiased Asymmetric Reinforcement Learning under Partial Observability. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pp. 44–52, 2022. URL https://www.ifaamas.org/Proceedings/aamas2022/pdfs/p44.pdf.

Alexander Bartler, Andre Bühler, Felix Wiewel, Mario Döbler, and Bin Yang. MT3: Meta Test-Time Training for Self-Supervised Test-Time Adaption. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 151 of *Proceedings of Machine Learning Research*, pp. 3080–3090. PMLR, 2022. URL https://proceedings.mlr.press/v151/bartler22a.html.

Morgan Byrd, Jackson Crandell, Mili Das, Jessica Inman, Robert Wright, and Sehoon Ha. Privileged-Dreamer: Explicit Imagination of Privileged Information for Rapid Adaptation of Learned Policies. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3640–3646, 2025. doi: 10.1109/ICRA55743.2025.11127672. URL https://ieeexplore.ieee.org/document/11127672/.

Zhixiang Chi, Yang Wang, Yuanhao Yu, and Jin Tang. Test-Time Fast Adaptation for Dynamic Scene Deblurring via Meta-Auxiliary Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9137–9146, 2021. doi: 10.1109/CVPR46437.2021.00902. URL https://openaccess.thecvf.com/content/CVPR2021/html/Chi_Test-Time_Fast_Adaptation_for_Dynamic_Scene_Deblurring_via_Meta-Auxiliary_Learning_CVPR_2021_paper.html.

Ben Day, Alexander Norcliffe, Jacob Moss, and Pietro Liò. Meta-learning using privileged information for dynamics. In *ICLR 2021 SimDL Workshop*, 2021. URL https://simdl.github.io/files/10.pdf.

F. Deng, I. Jang, and S. Ahn. DreamerPro: Reconstruction-Free Model-Based Reinforcement Learning with Prototypical Representations. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4956–4975. PMLR, 2022. URL https://proceedings.mlr.press/v162/deng22a.html.

Yusen Feng, Bingchen Han, Jiangran Lyu, Kai Liu, Yixin Zheng, Yuxuan Wan, Weiheng Liu, Sun Han, Ruiqin Li, Yulong Zhang, Fangfu Liu, Xuesong Shi, Libin Liu, Yizhou Wang, Zhizheng Zhang, and He Wang. WAM-TTT: Steering World-Action Models by Watching Human Play at Test Time, 2026. URL https://arxiv.org/abs/2607.06988. arXiv preprint arXiv:2607.06988.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017. URL https://proceedings.mlr.press/v70/finn17a.html.

Edward S. Hu, James Springer, Oleh Rybkin, and Dinesh Jayaraman. Privileged Sensing Scaffolds Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=EpVe8jAjdx.

Dongchi Huang, Jiaqi Wang, Yang Li, Chunhe Xia, Tianle Zhang, and Kaige Zhang. PIGDreamer: Privileged Information Guided World Models for Safe Partially Observable Reinforcement Learning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 25859–25875. PMLR, 2025a. URL https://proceedings.mlr.press/v267/huang25ai.html.

Yuhang Huang, Shilong Zou, Jiazhao Zhang, Xinwang Liu, Ruizhen Hu, and Kai Xu. AdaPower: Specializing World Foundation Models for Predictive Manipulation, 2025b. URL https://arxiv.org/abs/2512.03538. arXiv preprint arXiv:2512.03538.

Eric Jing and Abdeslam Boularias. Bounding Distributional Shifts in World Modeling through Novelty Detection. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 21311–21317, 2025. doi: 10.1109/IROS60139.2025.11247183. URL https://doi.org/10.1109/IROS60139.2025.11247183.

Usman M. Khan. Depth-Regularized JEPA World Models Learn More Transferable Representations from Real Outdoor Robot Data, 2026. URL https://arxiv.org/abs/2607.16314. arXiv preprint arXiv:2607.16314.

A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid Motor Adaptation for Legged Robots. In *Robotics: Science and Systems XVII (RSS)*, 2021. doi: 10.15607/RSS.2021.XVII.011. URL https://roboticsproceedings.org/rss17/p011.html.

Gaspard Lambrechts, Damien Ernst, and Aditya Mahajan. A Theoretical Justification for Asymmetric Actor-Critic Algorithms. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 32375–32405. PMLR, 2025. URL https://proceedings.mlr.press/v267/lambrechts25a.html.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47):eabc5986, 2020a. doi: 10.1126/scirobotics.abc5986. URL https://leggedrobotics.github.io/rl-blindloco/.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware Dynamics Model for Generalization in Model-Based Reinforcement Learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5757–5766. PMLR, 2020b. URL https://proceedings.mlr.press/v119/lee20g.html.

Huan Liu, Zhixiang Chi, Yuanhao Yu, Yang Wang, Jun Chen, and Jin Tang. Meta-Auxiliary Learning for Future Depth Prediction in Videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5756–5765, 2023. doi: 10.1109/WACV56688.2023.00571. URL https://openaccess.thecvf.com/content/WACV2023/html/Liu_Meta-Auxiliary_Learning_for_Future_Depth_Prediction_in_Videos_WACV_2023_paper.html.

Antonio Loquercio, Elia Kaufmann, René Ranftl, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Learning high-speed flight in the wild. *Science Robotics*, 6(59):eabg5810, 2021. doi: 10.1126/scirobotics.abg5810. URL https://doi.org/10.1126/scirobotics.abg5810.

Gabriel B. Margolis, Ge Yang, Kartik Paigwar, Tao Chen, and Pulkit Agrawal. Rapid Locomotion via Reinforcement Learning. In *Proceedings of Robotics: Science and Systems*, 2022. doi: 10.15607/RSS.2022.XVIII.022. URL https://www.roboticsproceedings.org/rss18/p022.html.

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7 (62):eabk2822, 2022. doi: 10.1126/scirobotics.abk2822. URL https://leggedrobotics.github.io/rl-perceptiveloco/.

A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to Adapt in Dynamic, Real-World Environments Through Meta-Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/pdf?id=HyztsoC5Y7.

Younggyo Seo, Kimin Lee, Ignasi Clavera, Thanard Kurutach, Jinwoo Shin, and Pieter Abbeel. Trajectory-wise Multiple Choice Learning for Dynamics Generalization in

Reinforcement Learning. In *Advances in Neural Information Processing Systems 33*, 2020. URL https://papers.nips.cc/paper_files/paper/2020/hash/9739efc4f01292e764c86caa59af353e-Abstract.html.

Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What Drives Success in Physical Planning with Joint-Embedding Predictive World Models? *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/pdf?id=cHZn5Gdh8e.

Chi Wan, Kangrui Wang, Yuan Si, Pingyue Zhang, and Manling Li. WorldAgen: Unified State-Action Prediction with Test-Time World Model Training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18584–18592, 2026. doi: 10.1609/aaai.v40i22.38925. URL https://ojs.aaai.org/index.php/AAAI/article/view/38925.

Ying Wang, Oumayma Bounou, Yann LeCun, and Mengye Ren. AdaJEPA: An Adaptive Latent World Model, 2026. URL https://arxiv.org/abs/2606.32026. arXiv preprint arXiv:2606.32026.

Haodong Yan, Jiaguan Zhu, Mingyuan Jia, Ruiqing Yin, Junjie He, Zhide Zhong, Junfeng Li, Jinxuan Lu, Hengtao Li, Tianran Zhang, Jiayi Chen, Wenxuan Song, Wen Chen, Yuxiang Gao, and Haoang Li. Is Forward Prediction Enough? Physical State Grounding for JEPA World Models, 2026. URL https://arxiv.org/abs/2608.06799. arXiv preprint arXiv:2608.06799.

Sizhe Yang, Yanjie Ze, and Huazhe Xu. MoVie: Visual Model-Based Policy Adaptation for View Generalization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 21507–21523. Curran Associates, Inc., 2023. doi: 10.52202/075280-0940. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/43b77cef2a83a25aa27d3271d209e4fd-Abstract-Conference.html.

Shao-Yi Yu, Jen-Wei Wang, Maya Horii, Masayoshi Tomizuka, and Vikas Garg. Learning Context-Aware Neural ODE Dynamics for Adaptive Robotic Control. *IEEE Robotics and Automation Letters*, 11(8):9255–9262, 2026. doi: 10.1109/LRA.2026.3701574. URL https://research.aalto.fi/en/publications/learning-context-aware-neural-ode-dynamics-for-adaptive-robotic-c/.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A Method Implementation and Reproduction Details

This section provides benchmark-specific implementation details omitted from the main Method section. The full training configuration and code will be provided in the public implementation.

A.1 Base World Model and Source Training Objective

We follow AdaJEPA's latent-prediction training setup. The model predicts successor latents from a short observation history and the corresponding actions. The current encoder re-encodes real successor observations to produce prediction targets, and gradients are stopped on the target side. No separate teacher or EMA encoder is used.

In the main text, $\mathcal{L}_{\mathrm{src}}$ includes the base latent-prediction objective and the temporal representation regularization in the original implementation. The temporal regularization weights are 0.01 for Push-Object and 0.1 for PointMaze. The actual predictor is always conditioned on a short history; the one-step form in the main text only simplifies notation.

A.2 Physical Supervision

Push-Object uses spatial occupancy of the agent and manipulated object as the physical target. Real environment states are rasterized into two 64×64 occupancy channels, and the auxiliary head predicts the corresponding masks from a single-frame visual latent. Supervision combines balanced binary cross-entropy and Soft Dice.

PointMaze uses three consecutive visual latents to regress standardized two-dimensional positions and velocities,

$$[q_x, q_y, v_x, v_y].$$

Both auxiliary heads read only visual representations. They do not use actions, rewards, goals, or planning costs, and they do not serve as deployment-time state estimators. Physical-loss weights are benchmark-specific; full values and configurations will be released with the code.

A.3 Simulated AdaJEPA Update

The temporary source-training update strictly reuses the trainable parameter subset and optimizer configuration of the corresponding AdaJEPA deployment. This includes selected encoder-projection layers, the final Transformer block of the predictor, and the final normalization. The predictor and encoder-projection learning rates are 5×10^{-4} and 1×10^{-5}, respectively, with a fresh Adam optimizer state. Each simulated update starts from the current source parameters, and the temporary parameters are discarded after the current outer objective.

The Push-Object meta segment contains 5 adaptation transitions and 5 later evaluation targets. PointMaze uses one adaptation transition and one later target. Evaluation targets do not enter the inner update. Physical labels, auxiliary predictions, rewards, goals, and planning costs are also excluded from the inner adaptation objective.

A.4 Post-Adaptation Objectives

After the temporary adapted parameters w^+ are constructed, the adapted encoder re-encodes real later observations and the adapted predictor computes the post-adaptation dynamics objective. $\mathcal{L}_{\mathrm{dyn}}^{\mathrm{post}}$ uses detached target-variance normalization to reduce the effect of latent scale on the outer objective; gradients are stopped through both the target branch and the normalization statistics.

$\mathcal{L}_{\mathrm{phys}}^{\mathrm{post}}$ acts directly on visual representations obtained by passing real later observations through the adapted encoder, rather than on predictor rollout latents. Thus, Post-Phys and Post-Dyn constrain physical information and dynamics prediction after the update, respectively.

A.5 Source Optimization

Push-Object and PointMaze use their own predefined training steps, loss weights, and ramp schedules. We limit the gradient contribution of post-adaptation objectives to keep training stable. These

settings affect only source training and do not change the adaptation loss, updatable parameter subset, or optimizer at real deployment.

The physical auxiliary head always remains outside the AdaJEPA inner update. After training, temporary adapted parameters, physical labels, and the auxiliary head do not participate in real deployment.

A.6 Deployment Evaluation and Statistics

Push-Object uses CEM-based receding-horizon MPC with 200 candidates, 30 elites, 10 optimization iterations, and a planning horizon of 25. It executes 5 environment steps at a time. After each eligible feedback event, the model updates parameters and replans according to AdaJEPA's original rule. The main experiment performs one Adam update per feedback event.

The Push-Object 13-OOD aggregate contains 6 Shape, 4 Corruption, and 3 Color domains with equal weight. Frozen and Adapted evaluations use strictly paired task instances. The main confidence intervals use 20,000 paired hierarchical bootstrap samples. The bootstrap preserves the hierarchy of random seeds and their episodes and uses the same resampling indices across methods and Frozen/Adapted arms. The 95% confidence interval uses the 2.5 and 97.5 percentiles of the bootstrap distribution.

For Push-Object, domain-level adaptation gain is

$$G_d = S_{A,d} - S_{0,d}.$$

HNAG is computed within each domain as

$$\mathrm{HNAG}_d = \frac{S_{A,d} - S_{0,d}}{1 - S_{0,d}},$$

and then averaged across domains with equal weights. Gain SD is the sample standard deviation of the 13 values of G_d. WQ is the mean of the lowest 25% probability mass in the domain-gain distribution. Positive Domains counts domains where $G_d > 0$.