Changepoint-Aware World Models: Detecting Dynamics Shifts and Recovering by Forgetting Stale Replay in Model-Based RL

Everest Yang
Brown University
everest_yang@brown.edu

***Abstract*—A robot's learned model of its own dynamics is only valid until those dynamics change: actuators wear, payloads shift, and joints stiffen. A model-based agent that keeps training as if nothing happened adapts slowly, dragged back by a replay buffer full of stale experience. We present *Changepoint-Aware World Models* (CAWM), a DreamerV3 agent that detects an abrupt dynamics shift from its *own* internal prediction error, using an online CUSUM test against a rolling baseline that fires only on abrupt change rather than on slow learning drift. It then *forgets stale replay*, keeping the learned representation while flushing obsolete data. On simulated locomotion under two robot-relevant shifts, doubled gravity and halved actuator gain, CAWM recovers substantially faster than passive retraining. It also beats a strong baseline that respawns a fresh dynamics model on detection, the deep-world-model analogue of model-bank methods. With the response triggered at the shift, CAWM gains $+95$ to $+153$ return in the first 30k post-shift frames over three seeds, while matching that respawn at asymptote. Running the detector in closed loop reproduces this gain on the gravity shift. The benefit holds across both shift types, and is largest when the shift is severe enough that old data is genuinely obsolete.**

I. Introduction

A model-based reinforcement learning agent builds an internal model of how its actions move the world, and then plans or imagines inside it. For a robot, that is a model of its own body and contacts. But a real robot's dynamics are not stationary: actuators degrade, payloads are picked up, joints stiffen with wear, terrain changes underfoot. When the dynamics shift abruptly, a world model trained on the old regime is suddenly wrong, and an agent that keeps training recovers slowly, held back by a replay buffer dominated by stale transitions.

This raises two coupled questions. First, can an agent *detect* that its dynamics have shifted using only signals it already computes, without an external change oracle? Second, once it knows, what is the right *response*? We study both, and separate them: detection first, then the response on top.

Our **primary contribution** is that in a deep latent world model, the most effective response to a detected shift is to *forget stale experience* while *keeping the learned representation*. We detect the shift with an online CUSUM test [21] on the world model's own dynamics-prediction loss, against a *rolling trailing baseline* that tracks the agent's slow learning drift so that only abrupt changes trigger it. On detection we cap the replay buffer, evicting obsolete data and let the world model re-fit. The encoder and policy are untouched and only the data the model trains on is refreshed.

Our **secondary contribution** is a controlled comparison against the natural alternative: respawning a fresh dynamics model on detection, the deep-world-model analogue of model-bank changepoint methods [4]. We find that respawning incurs a sharp re-adaptation transient, a second crash as the fresh model relearns from scratch, whereas forgetting avoids it. The two reach comparable asymptotic performance, so the advantage of CAWM is *faster recovery*. This holds across two robot-relevant shift types and three seeds each.

In summary, we contribute: (i) an online changepoint detector that reads a deep world model's internal prediction error against a rolling baseline. (ii) A forgetting response that recovers faster than passive retraining and faster than respawning a fresh model, at comparable asymptote, across a gravity shift and an actuator-wear shift. (iii) A conditional finding delimiting when forgetting helps, namely when the shift is severe enough that old data is obsolete.

II. Related Work

World models and model-based RL: Learning a model of the environment and planning or training a policy inside it is a long-standing idea [10, 12, 23, 6, 15]. DreamerV3 [14], building on DreamerV2 [13], learns a latent world model and trains an actor-critic inside it, and is our base agent. Such models, and the actor-critics they build on [11], are typically studied under stationary dynamics. Exploration methods that act on model disagreement [24] detect where the model is uncertain, but target a fixed environment, not a changing one.

Changepoint detection and adaptation in RL: MBCD [4] runs a multivariate CUSUM over an ensemble of dynamics predictors and switches a context model on detection. Our respawn baseline is its analogue inside a single deep world model. Closest to our detector, Zollicoffer et al. [29] flag a shift from a DreamerV2 world model's prediction misalignment, but for *detection only*, leaving recovery out of scope. We instead detect *and* respond. ARCADE [28] tempers a Bayesian dynamics prior on a changepoint, the closest neighbour to our forgetting, but freezes an offline representation and adapts a linear decoder rather than learning end-to-end. Other non-stationary methods instead adapt the policy, by detecting a behaviour change [19], inferring a hidden latent context [5, 8, 27], or meta-learning across dynamics [3]. More broadly, continual RL studies the stability-plasticity trade-off our forgetting navigates [16, 1], including the loss of plasticity stale data induces [17, 20, 7].

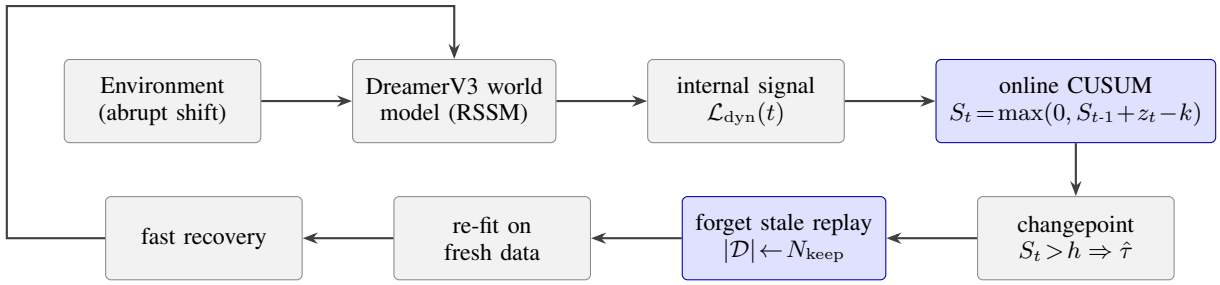

Fig. 1. Changepoint-Aware World Model. The agent monitors its own world-model prediction error $\mathcal{L}_{\text{dyn}}$. An online CUSUM test against a rolling trailing baseline fires only on an abrupt dynamics shift (blue: the changepoint machinery). On detection it forgets stale replay, capping the buffer while keeping the encoder, so the world model re-fits the new dynamics and the agent recovers. The MBCD-style baseline instead respawns the dynamics model at $\hat{\tau}$.

Robot adaptation: Rapid Motor Adaptation [18] adapts a legged policy to payload, wear, and terrain by inferring a latent context online. We share the motivation, but address it inside a world model by detecting the change and refreshing its data, rather than context inference in a model-free policy.

III. Method

Problem: An agent acts in an environment whose transition dynamics P_θ are stationary until an unknown changepoint τ, after which they switch abruptly to $P_{\theta'}$ and remain fixed. The agent observes only its own states, actions, and rewards. The changepoint τ and the new dynamics are unknown. We measure post-shift performance by episode return over fixed windows relative to τ: a *transient* window $[0, 30\text{k})$ frames, an *asymptote* window $[30\text{k}, 90\text{k})$, and their union, the *cumulative* window, which is a return-based proxy for adaptation regret.

Detecting the shift: A DreamerV3 world model already computes a dynamics-prediction loss $\mathcal{L}_{\text{dyn}}(t)$ (the KL between the posterior and prior latent transition) each update. When the true dynamics shift, the model's predictions stop matching observation and $\mathcal{L}_{\text{dyn}}$ rises. The difficulty is that $\mathcal{L}_{\text{dyn}}$ is also non-stationary *before* any shift, as the agent is still learning, so a fixed-threshold test false-alarms on the learning curve itself. We therefore run a one-sided CUSUM test [21, 2] against a *rolling trailing baseline*. We standardize $\mathcal{L}_{\text{dyn}}(t)$ by the mean and standard deviation of a trailing window, lagged to exclude the current point. The statistic $S_t = \max(0, S_{t-1} + z_t - k)$ then accumulates only sustained positive deviations and fires when $S_t > h$. Because the baseline slides with the agent's slow drift, gradual learning is absorbed and only an abrupt jump crosses the threshold. The slack k and threshold h trade detection delay against false alarms, and we use a fixed $k{=}0.5$ and $h{=}8$. We monitor the dynamics loss rather than a one-step latent surprise computed on fresh transitions which we found too weak and noisy to trigger reliably. Both are internal, control-relevant signals the model produces in the spirit of the value-equivalence view of what a model is for [9].

Responding by forgetting: On detection, CAWM caps the replay buffer to its most recent N_{keep} transitions. DreamerV3 already evicts the oldest episodes down to a configured size. A single change to that size flushes stale pre-shift data within roughly N_{keep} frames, after which the world model trains only on post-shift experience. The encoder, actor, and critic are left intact, and only the data is refreshed. This differs from prioritized experience replay [22], which reweights how often stored transitions are sampled but never discards them as obsolete. It is instead the deep-world-model counterpart of tempering a prior on a changepoint [28]: keep what generalizes (the representation), discard what is now wrong (the data).

Respawn baseline: As a controlled alternative we replace the forgetting step with an MBCD-style *respawn*: on detection, reinitialize the world model's latent dynamics and prediction heads in place (a fresh "context model"), keeping the encoder and policy, and reset the model optimizer. We emphasize this is an MBCD-*style* response adapted to a deep world model, not a literal reimplementation of MBCD's ensemble-and-model-bank system. It isolates the single question of whether to *discard the model* or merely *discard the data* on a shift.

IV. Experiments

Setup: We use the `walker_walk` locomotion task from the DeepMind Control suite [25] on the MuJoCo physics engine [26], with a DreamerV3 agent. We inject two robot-relevant abrupt shifts: gravity $\times 2.0$ (a payload / load-change analogue) and actuator-gain $\times 0.5$ (a motor-wear analogue), each fired at a fixed frame. To compare responses on identical histories we train one pre-shift checkpoint to convergence and branch three arms that resume it and undergo the shift: *passive* (continue training, full buffer), *CAWM* (forget on the shift), and *MBCD-style* (respawn on the shift). To isolate the response from detector-timing noise, the three-arm comparisons below trigger the response at the known shift frame (oracle timing). We then close the loop with the online detector and report it separately, verifying it fires within a few hundred frames of the shift with no pre-shift false alarm. We report mean $\pm$ std over three seeds.

Recovery. Figure 2 shows the dynamics. Under both shifts, passive retraining never recovers, pulled down by its stale buffer. CAWM recovers quickly and to the highest level. The MBCD-style respawn suffers a second, sharp crash right after the shift, the cost of relearning a dynamics model from scratch, before re-climbing to roughly CAWM's level. Table I quantifies this over the three windows. CAWM beats passive decisively everywhere (up to $+294$ at asymptote under motor-wear). Against the MBCD-style respawn, CAWM's advantage

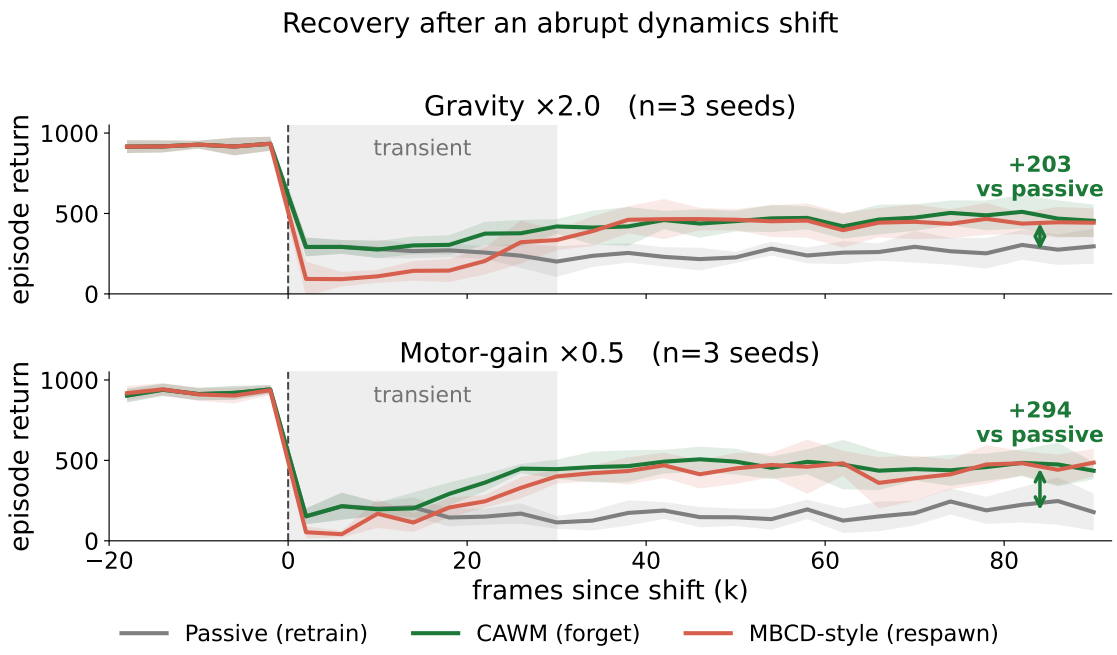

Fig. 2. Episode return versus frames since the shift, for the gravity (top) and motor-wear (bottom) shifts, mean over three seeds (shaded: std), with the transient window highlighted. Passive retraining stays stuck, CAWM recovers fastest, and the MBCD-style respawn crashes hardest immediately after the shift as its fresh model relearns before catching up. The green arrow marks CAWM's asymptotic gain over passive.

TABLE I
MEAN POST-SHIFT RETURN (WALKER_WALK, THREE SEEDS). WINDOWS ARE FRAMES SINCE THE SHIFT. THE Δ VALUES DISCUSSED IN THE TEXT ARE CAWM MINUS EACH BASELINE.

Shift	Window	Passive	MBCD-style	CAWM
Gravity $\times 2.0$	Transient	268	171	**323**
	Asymptote	256	441	**459**
	Cumulative	260	351	**414**
Motor-gain $\times 0.5$	Transient	172	183	**278**
	Asymptote	173	440	**467**
	Cumulative	173	354	**404**

is concentrated in the *transient*: $+153$ on gravity and $+95$ on motor-wear in the first 30k frames, robust across all three seeds. CAWM never pays the respawn's re-adaptation crash. At *asymptote* the two are comparable ($+18$ and $+27$, within seed variation), so forgetting recovers faster, not higher. The cumulative window, a return-based regret proxy, favors CAWM by $+63$ and $+50$.

Consistency across shift types. The two shifts are physically unrelated, a global force change versus a per-actuator gain change, yet produce the same structure (Table I, Fig. 3). The same detect-and-forget mechanism holds across both shifts without per-shift tuning, evidence that the effect is about *the response to non-stationarity*, not a particular perturbation.

Closed-loop detection. The comparisons above trigger the response at the known shift frame to isolate the response mechanism from detector-timing noise. We also ran the full loop with the online detector in control. On the gravity $\times 2.0$ shift, the CUSUM test fires 200 to 1000 frames after the true changepoint across three independent seeds, with *no* pre-shift false alarm despite the agent's ongoing learning drift. That drift is the precise failure mode a fixed baseline suffers and the rolling baseline removes. The detector-triggered forgetting then recovers to the same level as the oracle-timed response, beating passive by $+203 \pm 18$ over the asymptote window. Because the detection delay and the buffer-flush lag are small relative to the multi-thousand-frame window over which passive stays broken, closing the loop costs little: the recovery benefit is preserved end-to-end with essentially no added machinery.

Robustness across seeds. The transient advantage over the respawn is the consistent part of the result: CAWM wins the first-30k window in all three seeds of both shifts. The asymptotic margin behaves differently, on motor-wear it shrinks across seeds ($+56$, $+23$, $+2$), so the two arms converge and their per-seed ordering there is not stable. We therefore report the asymptote as a tie and claim only faster recovery, which the data robustly support. The CAWM-over-passive gap is large and stable in every window and seed.

When does forgetting help? Forgetting is not universally good. Under a milder gravity shift ($\times 1.5$) the passive agent self-heals to roughly its pre-shift return ($\sim$900) because much of its old experience is still valid, and forgetting then offers no benefit, plateauing lower ($\sim$716) as it discards data that is still useful. Forgetting helps precisely when the shift is severe enough that pre-shift data is obsolete (gravity $\times 2.0$, gain $\times 0.5$), where passive retraining is permanently dragged down (Table I). Whether to forget is thus shift-magnitude dependent, which the changepoint statistic's magnitude can

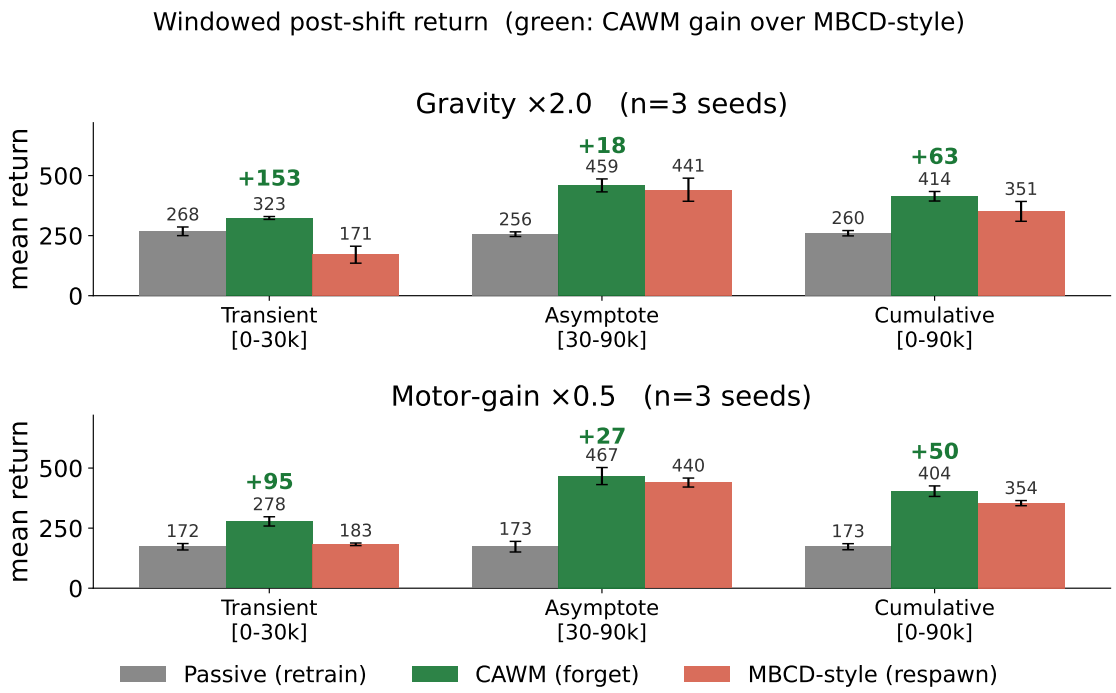

Fig. 3. Windowed mean return for the gravity (top) and motor-wear (bottom) shifts, three seeds (error bars: std). Green annotations give CAWM's gain over the MBCD-style respawn: large in the transient, shrinking to a tie at asymptote, on both shifts.

inform but a fixed "always forget" rule ignores.

Implementation details. The base agent is DreamerV3 in its default proprioceptive configuration (action repeat 2). A single pre-shift checkpoint is trained for 150k frames to convergence and saved with its replay buffer. All three arms resume it and run 130k further frames, of which the last 90k are post-shift. Forgetting caps the replay buffer from its pre-shift size ($\sim 1.5\times 10^5$ transitions) to $N_{\text{keep}}=10^4$, after which DreamerV3's existing oldest-first eviction flushes stale data within $\sim N_{\text{keep}}$ frames. The detector consumes one $\mathcal{L}_{\text{dyn}}$ value per logging step (every 100 frames) with CUSUM slack $k=0.5$, threshold $h=8$, a trailing baseline of the last 100 values (lag 5), light exponential smoothing, and a 60-step warm-up before it may fire. It thus adds one scalar test per logging step and no extra network evaluations. The respawn baseline reinitializes the latent dynamics and prediction heads and resets the model optimizer, keeping the encoder and actor-critic. Forgetting and respawn thus differ only in whether the learned dynamics are discarded.

Scope. The detector assumes a roughly converged pre-shift baseline, matching an agent that has settled before its dynamics change. The seed arms share that pre-shift checkpoint and vary post-shift stochasticity. Broadening to more tasks, gradual drift, and fully independent seeds is future work.

V. Discussion

These results suggest that, for a model-based agent facing abrupt dynamics change, the leverage is in *what to discard*. Discarding the data while keeping the representation (CAWM) recovers faster than discarding the model (respawn) and far faster than discarding nothing (passive): the learned encoder still generalizes across the shift, so only the dynamics need re-fitting. An abrupt shift changes how states evolve, not how observations map to states. That is why the representation survives while the transition model alone must be relearned. The advantage is a recovery *transient*, not an asymptotic ceiling, and it appears only when the shift is severe enough to render old data obsolete. The detector also adds little cost. It reuses a loss the world model computes and introduces no extra networks, so the mechanism drops into any Dreamer-style agent without changing the base learner. The most immediate next direction is breadth across robot morphologies, extending the same detect-and-forget comparison to the cheetah and quadruped bodies to test whether the transient advantage holds as the dynamics grow higher-dimensional. The conditional result also motivates graded forgetting scaled to the changepoint magnitude rather than a fixed cap, stronger non-stationary baselines such as a recency-weighted buffer and an explicit model bank, and calibrating the detector to a target false-alarm rate.

VI. Conclusion

We presented Changepoint-Aware World Models, which detect an abrupt dynamics shift from a deep world model's own prediction error and recover by forgetting stale replay while keeping the representation. Across two robot-relevant shifts and three seeds, this recovers faster than both passive retraining and respawning a fresh dynamics model, at comparable asymptote.

References

[1] David Abel, André Barreto, Benjamin Van Roy, Doina Precup, Hado van Hasselt, and Satinder Singh. A definition of continual reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2307.11046.

[2] Ryan Prescott Adams and David J. C. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[3] Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations (ICLR)*, 2018. arXiv:1710.03641.

[4] Lucas N. Alegre, Ana L. C. Bazzan, and Bruno C. da Silva. Minimum-delay adaptation in non-stationary reinforcement learning via online high-confidence change-point detection. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2021. arXiv:2105.09452.

[5] Xiaoyu Chen, Xiangming Zhu, Yufeng Zheng, Pushi Zhang, Li Zhao, et al. An adaptive deep rl method for non-stationary environments with piecewise stable context. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2212.12735.

[6] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. arXiv:1805.12114.

[7] Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632:768–774, 2024.

[8] Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016. arXiv:1308.3513.

[9] Christopher Grimm, André Barreto, Satinder Singh, and David Silver. The value equivalence principle for model-based reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2011.03506.

[10] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. arXiv:1809.01999.

[11] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018. arXiv:1801.01290.

[12] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1912.01603.

[13] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.02193.

[14] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[15] Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning (ICML)*, 2022. arXiv:2203.04955.

[16] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022. arXiv:2012.13490.

[17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences (PNAS)*, 114(13):3521–3526, 2017.

[18] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: Rapid motor adaptation for legged robots. In *Robotics: Science and Systems (RSS)*, 2021. arXiv:2107.04034.

[19] Zihe Liu, Jie Lu, Guangquan Zhang, and Junyu Xuan. A behavior-aware approach for deep reinforcement learning in non-stationary environments without known change points. *arXiv preprint arXiv:2405.14214*, 2024.

[20] Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2022. arXiv:2205.07802.

[21] E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

[22] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations (ICLR)*, 2016. arXiv:1511.05952.

[23] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588:604–609, 2020.

[24] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning (ICML)*, 2020. arXiv:2005.05960.

[25] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez,

Yazhe Li, et al. DeepMind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[26] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[27] Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst continual structured non-stationarity. In *International Conference on Machine Learning (ICML)*, 2021. arXiv:2006.10701.

[28] Rishabh Dev Yadav, Avirup Das, Hongyu Song, Samuel Kaski, and Wei Pan. ARCADE: Adaptive robot control with online changepoint-aware bayesian dynamics learning. *arXiv preprint arXiv:2512.14331*, 2025.

[29] Geigh Zollicoffer, Kenneth Eaton, Jonathan Balloch, Julia Kim, Wei Zhou, Robert Wright, and Mark Riedl. Novelty detection in reinforcement learning with world models. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2310.08731.