DoomShift: How Much Adaptation Do World Models Need Under Domain Shift?

Rohan Nagabhirava
Carnegie Mellon University
Pittsburgh, PA, United States
rnagabhi@andrew.cmu.edu

Keerthana Chirumamilla
Carnegie Mellon University
Pittsburgh, PA, United States
kchiruma@andrew.cmu.edu

Changliu Liu
Carnegie Mellon University
Pittsburgh, PA, United States
cliu6@andrew.cmu.edu

Abstract: World models have become capable simulators, but they are evaluated almost exclusively on held-out rollouts from the domain they were trained on. A small domain shift, such as a new map of the same game, lighting, textures or enemies, can make their outputs drift even when the underlying physics and game engine are unchanged. With DoomShift, a controlled version of this shift in Doom, we measure what breaks and how much adaptation is needed to repair it. We train three diffusion backbones, initialized from pretrained diffusion image models, on four Doom maps and evaluate them on 13 unseen maps. All three preserve their response to controls but fail to reproduce the new scene: scene PSNR drops by 2.8–3.3 dB and LPIPS rises by 0.126–0.145 across the three backbones. This consistent gap shows that action-conditioned behavior transfers more reliably than scene appearance, independent of architecture or model scale. Yet the lost appearance is relatively cheap to recover: rank-16 adapters trained on only eight episodes with 2% of the original training updates recover 72–76% of the LPIPS rise and about half of the lost PSNR. Together, these results suggest that adapting a world model to a new environment may require restoring the scene's appearance rather than relearning the underlying game.

Keywords: World Models, Diffusion Models, Few-shot Adaptation

1 Introduction

World models learned from recorded play now render whole games in real time, predicting each update to the world from the player's actions in place of complex, hand-crafted game engines [1, 2, 3]. However, they are only scored against ground truth on held-out rollouts from their training environment [1, 2, 4]: none has been shown to generalize to a new environment in the same game.

A learned simulator is useful only where it is faithful and an agent or a robot moving to a new room, level or workspace needs to know how much of its world model still holds and the cost of fixing the rest. Game worlds let us pose this question with the engine held fixed and only the environment changed: a new map of the same game preserves the physics and the controls exactly and only the layout, textures and lighting differ. Yet unchanged dynamics alone do not guarantee a faithful render: pretrained video and world models are typically moved to a new domain only through a further full fine-tune [5] or a dedicated adapter on the frozen backbone [6] and a model trained on a handful of maps tends to fall back on the appearance it already knows once the surroundings change. Pretraining one model on every map a deployment might meet is never complete and training a separate model per map from scratch is prohibitively costly. We therefore explore what it takes to close the domain gap that opens when a world model meets a shift in which the underlying physics and game engine are exactly the same and only the surroundings, scenario or scene change: what the model retains, what it loses and how few episodes and updates recover the rest.

In short, a world model moved to a new map keeps playing the game but draws the wrong world and a small adapter trained on eight episodes draws the map back (Figure 1). Our contributions:

10th Conference on Robot Learning (CoRL 2026), Austin TX, USA.



Figure 1: **Zero-shot vs. adapted world models under domain shift.** (a) On training maps, the prediction is close to the ground truth; (b) on an unseen map, it is not, until adapted on eight episodes.

- We introduce DoomShift, a benchmark that isolates a scene shift: 17 Doom maps share the same physics and game engine and differ only in their surroundings, with four maps for training and 13 held out. We release the recordings, models, decoders and adapters.
- We show that a scene shift breaks appearance, not control: across three diffusion backbones, the models keep their response to controls on the unseen maps while their perceptual error (LPIPS) roughly doubles and their scene PSNR drops 2.8–3.3 dB (Table 1).
- We measure how much adaptation repairs the shift: a standard LoRA adapter trained on eight episodes of an unseen map recovers 72–76% of the lost perceptual quality and roughly half of the lost PSNR, using only 2% of the original training updates (Table 1).

2 Related work

Game world models. Our work is most closely related to diffusion models that simulate a game from the player's actions: GameNGen runs Doom inside a diffusion model [1], DIAMOND learns Atari and Counter-Strike in pixel space [2], Matrix-Game 2 scales the recipe to open worlds [3] and MultiGen trains a single Doom model across 100 procedurally generated maps [4]. We build on GameNGen's backbone and training procedure and evaluate on maps the model never saw.

Generalization of world models. Two domain shifts are common for world models: a change in dynamics or a change in only the scene while the physics and engine are unchanged. Most evaluations test the first: world models have been evaluated on a new robot [7], game [6] or simulator [5], where the dynamics change along with the scene. For indoor navigation, Pathdreamer is evaluated in buildings not seen during training [8], the closest analogue to an unseen map, but its scenes are static and it does not study adaptation. Doom maps have been held out, but only to test the agents that play them [9, 10] and game world models that do show new scenes generate them from an image, with no recorded frames to score against [11, 12]. To our knowledge, no published game world model has been scored against recorded frames on held-out maps of its own game.

Adaptation. Pretrained video and world models are adapted to a new domain with a small adapter on a frozen backbone, as in AVID [6] and Vista [13], or with a short full fine-tune, as in AdaWorld [5]. We use low-rank adaptation (LoRA) [14] for its efficiency. None of these works reports what adapting to one new scene costs in episodes and updates, which is what we measure.

3 Method

Data. We collect the data by running the Arnold agent [9] through ViZDoom [15] deathmatches: free-for-all matches where each player scores by killing the others. Arnold ships with 17 Doom maps; each has its own layout, textures and lighting. For every episode, we let Arnold play against eight bots for up to 150 seconds. We record every tic, the engine's 1/35 s simulation step, capturing a 320×240 frame and the 19 buttons the engine executed. Four maps (2 to 5) are our training maps: we train on 500 episodes of each and score on 25 further validation episodes per map (512

windows). The other thirteen maps (1, 6 to 17) are never trained on: we record 24 new episodes on each, using 16 for adaptation and holding out eight (256 windows) for scoring. In total we collect 2,412 episodes, about 96 hours of play, 12.1 million frames and 1.8 TB of recordings.

Models and training. We choose three pretrained backbones: the SD 1.4 U-Net [16] (860M parameters) as the reference, following GameNGen [1]; PixArt-α [17] (a 628M transformer in the same latent space) to see the effect of a different architecture; and SD 3.5 Medium [18] (a 2.27B MMDiT with 16-channel latents) to see the effect of a larger model.

We turn all three models into next-tic world models under one shared recipe (Figure 2). We give each model 32 context tics so it can see motion, train with v-prediction [19], minimizing $\mathrm{E}\,\|v_\theta(z_t, t, \mathbf{c}, a) - v\|^2$ with $v = \sqrt{\bar{\alpha}_t}\,\epsilon - \sqrt{1-\bar{\alpha}_t}\,z_0$ for the next latent z_0, its noised version z_t, the 32 context latents $\mathbf{c}$ and the action a, for stable training across noise levels and add context noise augmentation as in GameNGen [1] so the model tolerates its own rollout errors. We train each backbone for 200k updates at batch size 32, keep an EMA at 0.9999 in fp32 and sample with 10-step DDIM [20], which we found to be a strong balance between quality and efficiency.

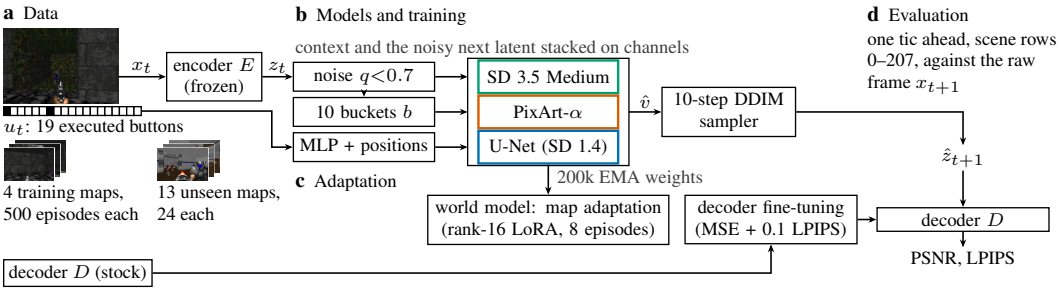


Figure 2: **Method overview.** (a) Data, (b) models and training, (c) adaptation and decoder fine-tuning, (d) evaluation against the raw frame.

Metrics. We score the part of each frame that shows the world (rows 0 to 207) and leave out the HUD rows below it, which do not change with the map.

For accuracy, we compute PSNR against the raw frame and for realism, LPIPS [21]. The *turn-response* score measures control: we flip the turn direction of the last action and count the fraction of windows in which the predicted view's horizontal shift from the last context frame reverses direction. The score is 0 for a model that ignores actions and 0.89 for a perfect model, because in 11% of windows the recorded turn moves the view too little to measure in one tic.

Decoder fine-tuning. As in GameNGen [1], we fine-tune each backbone's decoder on training-map frames with the encoder frozen, adding an LPIPS term to the loss (MSE plus 0.1 LPIPS); this happens once, before adaptation, the decoder stays frozen during it and it never sees an unseen map. This raises the scene reconstruction from 26.9 to 28.4 dB for the SD 1 decoder, which the U-Net and PixArt-α share and from 30.0 to 31.8 dB for the SD 3.5 decoder.

Adaptation. For each unseen map we adapt each backbone with rank-16 LoRA [14] on its attention layers [13] and train its action and input embeddings in full [22]: 0.49, 4.1 and 1.8% of the U-Net, PixArt-α and SD 3.5. We train on eight episodes of the map and score on its eight held-out episodes at 0 to 8k updates. We found that updates, not data, limit adaptation: a single episode gave 75–101% of the gain of sixteen, so eight episodes keep data from limiting the update curves. We report 4k updates throughout, 2% of the pretraining updates and $\sim$1% of its compute, which gives 96% of the gain at 8k. As a baseline, we fine-tune all of the U-Net's parameters on the same episodes.

4 Results

Zero-shot evaluation on unseen maps. On unseen maps, the models preserve control but fail to reproduce the new scene (Table 1). Turn-response scores remain close to those on the training maps

Table 1: **Zero-shot vs. adapted results on training and unseen maps.** Medians over each row's maps. Map difficulty: terciles of the U-Net's zero-shot LPIPS. Recovered: median over maps of $(M_{4k} - M_0)/(M_{\mathrm{tr}} - M_0)$; M: PSNR or LPIPS, 0: zero-shot, 4k: after 4k updates, tr: training.

	PSNR (dB) ↑		LPIPS ↓		Turn-response ↑		Recovered (%)	
Model, maps	0	4k	0	4k	0	4k	PSNR	LPIPS
SD 1.4 U-Net, training maps	25.20		0.158		0.855			
unseen maps (all 13)	22.30	23.60	0.303	0.210	0.809	0.840	48	72
unseen easy (7, 9, 10, 11)	23.38	24.92	0.255	0.170	0.836	0.863	173	87
unseen medium (6, 8, 12, 14, 15)	22.80	23.98	0.303	0.210	0.809	0.828	48	68
unseen hard (1, 13, 16, 17)	21.65	22.56	0.391	0.233	0.768	0.838	29	65
full fine-tune (all 13)	22.30	23.98	0.303	0.184	0.809	0.852	54	88
PixArt-α, training maps	25.18		0.159		0.842			
unseen maps (all 13)	22.38	23.67	0.285	0.204	0.793	0.844	47	76
SD 3.5 Medium, training maps	25.38		0.126		0.840			
unseen maps (all 13)	22.09	23.83	0.262	0.166	0.805	0.832	48	76

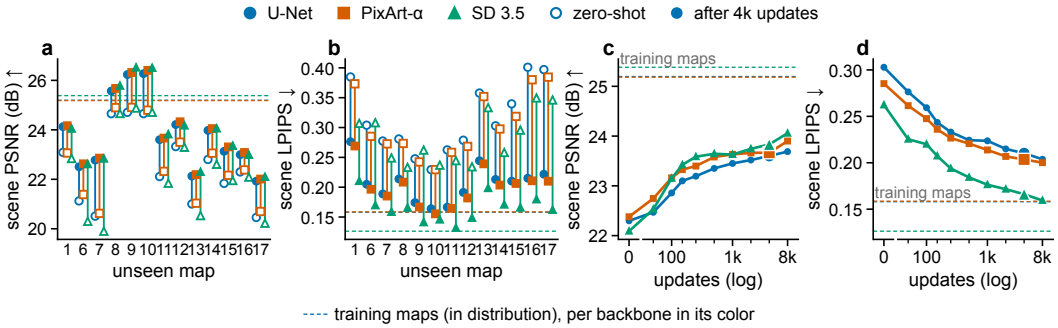


Figure 3: **Zero-shot vs. adapted results per unseen map.** (a, b) Scene PSNR and LPIPS, zero-shot (open) and after 4k adaptation updates (filled). (c, d) Median statistics against adapter updates.

(0.79–0.81 vs. 0.84–0.86), while scene PSNR drops by 2.8–3.3 dB and LPIPS roughly doubles. Every unseen map has worse LPIPS than every training map. The rollouts show the same pattern: weapon position, motion and turning remain consistent, but walls, floors and textures are often replaced by appearances learned from the training maps (Figure 1). This holds across the U-Net, PixArt-α and SD 3.5, so the failure is not specific to one architecture or model scale.

Adaptation cost. Adaptation primarily repairs the scene rather than the model's response to actions (Table 1, Figure 3). After 4k updates on eight episodes, the adapters recover 72–76% of the LPIPS rise and about half of the lost PSNR (above 100% where a map passes the training-map level), while turn-response scores change only slightly. The map's textures, colors and lighting return (Figure 1) without substantially changing the already-preserved control behavior.

Most of this recovery happens early: half of the LPIPS rise is recovered within a median of 150–250 updates, and extending training to 8k provides little additional gain. Full U-Net fine-tuning recovers more of the LPIPS rise (88% vs. 72%), but requires 205× more trainable parameters and a 48 instead of a 16 GB GPU, and causes 2.7× more forgetting on the training maps (1.08 vs. 0.40 dB).

5 Conclusion

DoomShift shows that under domain shift, world models largely preserve control but lose the appearance of the new scene. Across three diffusion backbones, lightweight LoRA adaptation on only eight episodes restores much of this lost perceptual quality without relearning the game dynamics.

Limitations. We study one game, evaluate one tic ahead, and measure only basic control.

References

[1] D. Valevski, Y. Leviathan, M. Arar, and S. Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://arxiv.org/abs/2408.14837.

[2] E. Alonso, A. Jelley, V. Micheli, A. Kanervisto, A. Storkey, T. Pearce, and F. Fleuret. Diffusion for world modeling: Visual details matter in Atari. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://arxiv.org/abs/2405.12399.

[3] X. He, C. Peng, Z. Liu, B. Wang, Y. Zhang, Q. Cui, F. Kang, B. Jiang, M. An, Y. Ren, B. Xu, H.-X. Guo, K. Gong, S. Wu, W. Li, X. Song, Y. Liu, Y. Li, and Y. Zhou. Matrix-Game 2.0: An open-source, real-time, and streaming interactive world model. *arXiv preprint arXiv:2508.13009*, 2025.

[4] R. Po, D. J. Zhang, A. Hertz, G. Wetzstein, N. Wadhwa, and N. Ruiz. MultiGen: Level-design for editable multiplayer worlds in diffusion game engines. *arXiv preprint arXiv:2603.06679*, 2026.

[5] S. Gao, S. Zhou, Y. Du, J. Zhang, and C. Gan. AdaWorld: Learning adaptable world models with latent actions. In *International Conference on Machine Learning (ICML)*, 2025. URL https://proceedings.mlr.press/v267/gao25u.html.

[6] M. Rigter, T. Gupta, A. Hilmkil, and C. Ma. AVID: Adapting video diffusion models to world models. *Reinforcement Learning Journal*, 6:737–764, 2025. URL https://rlj.cs.umass.edu/2025/papers/Paper64.html.

[7] Y. Chen, J. Yang, Y. Xu, Q. Ma, K. He, P. Li, K. Wang, Z. He, X. Wu, J. Liu, N. Liu, Y. Huang, and L. Wang. XEWorld: Can action-conditioned world models generalize to unseen robot embodiments? *arXiv preprint arXiv:2608.05799*, 2026.

[8] J. Y. Koh, H. Lee, Y. Yang, J. Baldridge, and P. Anderson. Pathdreamer: A world model for indoor navigation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. doi:10.1109/ICCV48922.2021.01447.

[9] G. Lample and D. S. Chaplot. Playing FPS games with deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2017. doi:10.1609/aaai.v31i1.10827.

[10] M. Wydmuch, M. Kempka, and W. Jaśkowski. ViZDoom competitions: Playing Doom from pixels. *IEEE Transactions on Games*, 11(3):248–259, 2019. doi:10.1109/TG.2018.2877047.

[11] Z. Tong, Y. Jin, H. Lai, Z. Wang, Z. Xing, K. Cheng, H. Xu, Z. Pu, S. Zhu, R. Feng, J. Zhao, Y. Zhang, H. Tang, and L. Shao. SCOPE: Simulating cross-game operations in playable environments for FPS world models. *arXiv preprint arXiv:2605.23345*, 2026.

[12] J. Bruce, M. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, Y. Aytar, S. Bechtle, F. Behbahani, S. Chan, N. Heess, L. Gonzalez, S. Osindero, S. Ozair, S. Reed, J. Zhang, K. Zolna, J. Clune, N. de Freitas, S. Singh, and T. Rocktäschel. Genie: Generative interactive environments. In *International Conference on Machine Learning (ICML)*, 2024. URL https://proceedings.mlr.press/v235/bruce24a.html.

[13] S. Gao, J. Yang, L. Chen, K. Chitta, Y. Qiu, A. Geiger, J. Zhang, and H. Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://arxiv.org/abs/2405.17398.

[14] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2106.09685.

[15] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games (CIG)*, 2016. doi:10.1109/CIG.2016.7860433.

[16] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. doi:10.1109/CVPR52688.2022.01042.

[17] J. Chen, J. Yu, C. Ge, L. Yao, E. Xie, Y. Wu, Z. Wang, J. Kwok, P. Luo, H. Lu, and Z. Li. PixArt-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2310.00426.

[18] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, D. Podell, T. Dockhorn, Z. English, and R. Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

[19] T. Salimans and J. Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2202.00512.

[20] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://arxiv.org/abs/2010.02502.

[21] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. doi:10.1109/CVPR.2018.00068.

[22] E. Xie, L. Yao, H. Shi, Z. Liu, D. Zhou, Z. Liu, J. Li, and Z. Li. DiffFit: Unlocking transferability of large diffusion models via simple parameter-efficient fine-tuning. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. doi:10.1109/ICCV51070.2023.00390.