FROM REPRESENTATION LEARNING TO RESHAPING LATENT GEOMETRY IN WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent world models encode high-dimensional observations into compact representations, where the dynamics can be learned and used for planning. Their representation quality is typically evaluated based on few-step prediction accuracy, while goal-conditioned planning aims to minimize the distance between long-term predictions and an encoded goal. In this work, we show that without proper supervision, the representation's *metric consistency* is poor, meaning that latent distances do not align well with the optimal cost of reaching the goal. We then reshape latent geometry using these costs and generalize the method through temporal ordering within recorded trajectories. Next, we show that geometry reshaping does not require extensive visual representation learning. We introduce an inexpensive test for representation sufficiency, based on the recovery of task-relevant state variables. We show empirically that once this criterion is met, visual training can stop and geometry reshaping in latent space alone can significantly improve planner performance. After two epochs of visual training in TwoRoom, the representation already satisfies our sufficiency criterion, but CEM planning success is only 11%. Freezing the encoder and continuing training in latent space raises success to 88% with proportional task-distance supervision and 90% with temporal ordering, compared with 27% using the base objective. Continued expensive visual training without reshaping plateaus at 37%.

1 INTRODUCTION

Latent world models encode high-dimensional visual observations into compact representations and predict how these representations evolve under actions (Hafner et al., 2019; 2020; Hansen et al., 2022; Sobal et al., 2025; Zhou et al., 2025; Maes et al., 2026). When planning with these models, candidate action sequences are rolled out using the learned dynamics and ranked using the Euclidean distance between the predicted terminal latent state and an encoded goal (Zhou et al., 2025; Maes et al., 2026). The latent representation therefore serves two roles: it provides a compressed state for the dynamics model, and its geometry is used to compare imagined outcomes.

Standard world-model training typically optimizes one-step latent prediction accuracy, but does not explicitly supervise the latent geometry. Consequently, distances in latent space may fail to reflect distances in task space, leading to poor planner performance. For example, in the TwoRoom environment, two states that are close in task space but separated by a wall may have a small distance in latent space even though reaching one from the other requires a long path around the wall.

To formalize this observation, we introduce the notion of *metric consistency*: a latent embedding is metrically consistent if pairwise Euclidean distances in the latent space preserve the ranking induced by optimal task distances, defined as the minimum costs of feasible trajectories between states. For goal-conditioned planning, this means that candidate outcomes closer to the goal in task space are also ranked accordingly in latent space. We first show that without appropriate supervision, latent representations exhibit poor metric consistency. We then provide two alternative regularizations to shape the latent geometry and improve metric consistency. When optimal task distances are available, we encourage latent distances to be proportional to them, providing a strong supervisory signal for metric consistency. We generalize this reshaping through temporal ordering within recorded trajectories, which encourages the latent distance to a trajectory endpoint to decrease over time. Importantly, we show that latent reshaping can be performed either by retraining the visual encoder

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.

or by introducing a lightweight latent adapter while freezing the upstream visual features, with the latter being significantly cheaper.

Given the effectiveness of our latent reshaping for planner performance, we next ask whether visual representation learning can stop once task-relevant information is encoded. Intuitively, if task-relevant information is already encoded, further visual training may be unnecessary, and reshaping latent geometry could be enough to make this information accessible to the planner and thus improve planning success. To this end, we study *representation sufficiency*, the ability to recover selected task-relevant state variables from individual representations with a fixed-capacity nonlinear probe. We show that once an early representation satisfies the representation sufficiency criterion, we can freeze the visual encoder and continue training a residual latent adapter and the predictor to improve metric consistency.

While our framework and diagnostic tools are general, we empirically evaluate them using LeWorldModel (Maes et al., 2026) on the long-range TwoRoom navigation benchmark, using A^* shortest paths as privileged supervision and agent position as the task-relevant state variable. After two epochs of visual training, agent position is recoverable with sub-pixel error, but planning success is only 11%. We then freeze the encoder, train the predictor for 2,000 updates, and jointly train the adapter and predictor for 20,000 updates. Mean planning success across three training seeds reaches 27.4% with the base objective, 88.1% with proportional task-distance supervision, and 89.8% with temporal ordering.

Our work is related to prior methods that shape latent spaces via temporal straightening (Wang et al., 2026) or by fitting goal-conditioned value functions and quasimetrics (Ma et al., 2023; Wang et al., 2023; Destrade et al., 2026). Local straightening constrains consecutive steps. We supervise the distances used to rank world-model predictions against a goal. A lightweight adapter reshapes these distances while keeping the visual encoder and the planner's Euclidean scoring rule fixed.

To summarize, our contributions are:

- We quantify agreement between latent and task distances using *metric consistency*, and evaluate two objectives to improve it: proportional task-distance supervision and temporal ordering.
- We show that improving *metric consistency* substantially improves planning success, particularly for long-range tasks.
- We show empirically that *representation sufficiency* can serve as a stopping criterion for expensive visual training: task-relevant state can become recoverable early in training, at which point metric consistency can be improved purely through substantially cheaper latent-space adaptation.

2 Related Work

World models and action search. PlaNet (Hafner et al., 2019) plans online with CEM by maximizing predicted rewards, while Dreamer (Hafner et al., 2020) learns an actor-critic from imagined trajectories and TD-MPC2 (Hansen et al., 2024) combines predicted rewards with a learned terminal Q-value. These methods therefore select actions through reward or value estimates rather than distances between learned representations.

We instead build on the world-model architectures DINO-WM (Zhou et al., 2025) and LeWorldModel (Maes et al., 2026), where candidates are scored by their latent distance to an encoded goal. DINO-WM learns dynamics over frozen pretrained features, whereas LeWorldModel jointly learns its visual representation and predictor using prediction and a SIGReg regularizer (Balestriero & LeCun, 2025). This feature-prediction approach is also related to I-JEPA and V-JEPA (Assran et al., 2023; Bardes et al., 2024).

Terver et al. (2026) systematically study the design choices that make such models effective for planning. They find CEM with an L_2 cost to perform best overall and observe that accurate long-horizon rollouts do not necessarily yield successful planning. We complement this analysis by studying how metric consistency affects planning success.

Parthasarathy et al. (2025) identify distribution shift induced by gradient-based planning as a source of world-model prediction error and fine-tune the model on planner-generated and adversarial trajectories. Their intervention targets prediction along planned trajectories, whereas ours targets the metric used to rank predicted outcomes.

Learning latent geometry. Several methods explicitly shape latent geometry to improve planning. Temporal straightening encourages alignment of consecutive latent displacements and improves gradient-based planning in the settings studied by Wang et al. (2026). Value-Implicit Pre-training (VIP) learns a visual embedding in which the negative Euclidean distance between an observation and a goal image represents a goal-conditioned value function (Ma et al., 2023). Quasimetric reinforcement learning uses the structure of optimal goal-conditioned values to learn task distances (Wang et al., 2023). Destrade et al. (2026) similarly train JEPA representations so that a distance or quasi-distance between state embeddings approximates the negative goal-conditioned value. Our objectives use direct distance regression or temporal-ordering supervision rather than Bellman or expectile-based value fitting. Complementary to the above methods, we also investigate how early visual representations can support subsequent geometry adaptation with a frozen encoder and introduce an empirically validated stopping criterion.

3 Reshaping Latent Geometry

3.1 World models and latent planning

Let $o_t \in \mathcal{O}$ be a visual observation and $a_t \in \mathcal{A}$ an action. The encoder f_θ maps each observation o_t to a latent representation:

$$z_t = f_\theta(o_t). \tag{1}$$

Let $h \geq 1$ denote the maximum context length. The predictor F_ψ takes up to the h most recent available latent representations and the corresponding actions and predicts the next latent representation:

$$z_{t+1} = F_\psi\big(z_{t-h+1:t}, a_{t-h+1:t}\big). \tag{2}$$

The encoder and predictor are jointly trained to minimize

$$\mathcal{L}_{\text{base}} := \mathcal{L}_{\text{pred}} + 0.09\,\mathcal{L}_{\text{SIGReg}}, \tag{3}$$

where $\mathcal{L}_{\text{pred}}$ is the latent prediction loss and $\mathcal{L}_{\text{SIGReg}}$ is the Sketched Isotropic Gaussian Regularization term, which encourages an isotropic Gaussian distribution of representations (Balestriero & LeCun, 2025). We use mean-squared error for one-step latent prediction from encoded observation histories.

We use these learned dynamics for goal-conditioned planning. Given a goal observation o_g, the planner searches for an action sequence $\mathbf{a} = (a_t, \ldots, a_{t+H-1})$. The predictor is rolled out autoregressively over the candidate sequence to obtain a predicted terminal latent z_{t+H}. Candidate sequences are evaluated using the distance of the terminal to the latent goal. In other words, we solve the latent-space optimal control problem

$$\begin{aligned} &\min_{\mathbf{a}} \ \|z_{t+H} - z_g\|_2^2, \\ &\text{s.t. } z_{k+1} = F_\psi(z_{t:k}, a_{t:k}), \quad k = t, \ldots, t+H-1, \\ &\qquad z_t = f_\theta(o_t), \\ &\qquad z_g = f_\theta(o_g). \end{aligned} \tag{4}$$

Here, F_ψ uses at most the h most recent available latent states and actions.

3.2 Metric consistency

We define *metric consistency* as agreement between latent distances and optimal task distances when ranking states relative to a goal. Let $d^\star_{ig}$ denote the minimum cost of a feasible trajectory from state s_i to goal s_g. In TwoRoom, this cost is path length. For a representation r, let $\ell_{ij} = \|r(o_i) - r(o_j)\|_2$.

Each comparison $(i, j, g) \in \mathcal{C}$ ranks two states relative to the same goal, with $d^\star_{ig} \neq d^\star_{jg}$. We measure metric-consistency error as

$$\epsilon_{\mathrm{MC}}(r) = \frac{1}{|\mathcal{C}|} \sum_{(i,j,g)\in\mathcal{C}} \mathbf{1}\left[\left(\ell_{ig} - \ell_{jg}\right)\left(d^\star_{ig} - d^\star_{jg}\right) \leq 0\right], \tag{5}$$

Zero error indicates perfect agreement, while random ordering gives an expected error of 0.5.

3.3 Reshaping parameterizations

We aim to improve metric consistency by fine-tuning the model using geometric losses added to $\mathcal{L}_{\mathrm{base}}$ that encourage metric consistency:

$$\mathcal{L}_{\mathrm{base}} + \lambda_i \mathcal{L}_i \tag{6}$$

where λ_i and $\mathcal{L}_i$ are the relative weights and geometric losses with i in $\{\mathrm{s}, \mathrm{geo}, \mathrm{order}\}$. The representation could be trained either by updating the visual encoder or by training a latent adapter over frozen visual features. We focus on the adapter, which allows subsequent training to operate entirely in latent space using cached representations.

In adapter-based reshaping, the visual backbone and original projection head are frozen. A residual latent adapter transforms their outputs:

$$y_t = A_\phi(z_t). \tag{7}$$

We apply A_ϕ to the current encoded observation and to the goal, so that prediction and planning operate in the same adapted space. In this formulation, we evaluate all losses, including the base loss, on the adapted representations.

We freeze the visual encoder and jointly train the adapter and predictor. Importantly, we can choose ϕ to be significantly smaller than θ. In particular, we choose the same size for y_t as for z_t and use a shallow MLP for A_ϕ, which makes adapter training inexpensive.

Next, we study different objectives to improve metric consistency by aligning latent-distance ordering with the optimal cost of reaching one state from another. We first review temporal straightening, and then introduce the proportional shortest-path objective and the temporal-ordering objective. For simplicity, we use z_t as a symbol for $\mathcal{L}_{\mathrm{s}}$ and $\mathcal{L}_{\mathrm{geo}}$; in practice, this would be replaced by y_t when a latent adapter is used.

3.3.1 Temporal straightening

Temporal straightening aligns consecutive latent displacement directions. For $v_t = z_{t+1} - z_t$, the objective is

$$\mathcal{L}_{\mathrm{s}} = -\mathbb{E}_t \left[\frac{v_t^\top v_{t+1}}{\|v_t\|_2 \|v_{t+1}\|_2} \right]. \tag{8}$$

This objective encourages locally straight trajectories in latent space, and has been shown to improve the performance of latent-space gradient-based planners (Wang et al., 2026). However, as we will show, it does not substantially improve the planning performance for the considered long-range task solved to high precision.

3.3.2 Proportional task-distance supervision

For a batch $\mathcal{B}$ of state pairs, let $d^\star_{ij}$ denote the optimal distance between states i and j. For each batch, we fit the scale $\alpha_{\mathcal{B}}$ by least squares so that $\ell_{ij} \approx \alpha_{\mathcal{B}} d^\star_{ij}$:

$$\alpha_{\mathcal{B}} = \frac{\sum_{(i,j)\in\mathcal{B}} \ell_{ij} d^\star_{ij}}{\sum_{(i,j)\in\mathcal{B}} (d^\star_{ij})^2}. \tag{9}$$

The geometric loss is defined as

$$\mathcal{L}_{\mathrm{geo}} = \frac{1}{|\mathcal{B}|} \sum_{(i,j)\in\mathcal{B}} h_1\left(\ell_{ij} - \alpha_{\mathcal{B}} d^\star_{ij}\right), \tag{10}$$

where h_1 is the Huber loss with threshold one. We use $\lambda_{\text{geo}} = 3$ for the weight of this loss. The derivation of the optimal scale and definition of the Huber loss are provided in Appendix B.

Proportional distances preserve the ordering induced by shortest-path distances, so minimizing $\mathcal{L}_{\text{geo}}$ encourages metric consistency. Note that all-zero representations also yield $\alpha_{\mathcal{B}} = 0$ and $\mathcal{L}_{\text{geo}} = 0$, but $\mathcal{L}_{\text{SIGReg}}$ prevents this collapse.

3.3.3 TEMPORAL ORDERING

Temporal ordering uses the order of states within recorded trajectories as an alternative source of geometric supervision. This assumes that later states tend to be closer to the trajectory endpoint.

Let $z_i = f_\theta(o_i)$ denote the frozen representation before adaptation and $y_i = A_\phi(z_i)$ its adapted representation. For a recorded trajectory window ending at timestep G, we define

$$d_i = \|y_i - y_G\|_2, \qquad \rho = \sqrt{\mathbb{E}_{i<G}[d_i^2]}. \tag{11}$$

We normalize the distance difference $d_j - d_i$ by ρ, the root-mean-square distance to the endpoint over the window. Unlike the start-to-end distance alone, ρ also accounts for intermediate states that may lie farther from the endpoint.

For $i < j < G$, we encourage states to be increasingly close to the endpoint, so that $d_j < d_i$. For a batch $\mathcal{B}$ of such windows, the temporal-ordering term is

$$\ell_{\text{order}}(\mathcal{B}) = \sqrt{\mathbb{E}_{w\in\mathcal{B},\, i<j<G_w} \left[\text{softplus}\left(\frac{d_j - d_i}{\rho} \right)^2 \right]}, \tag{12}$$

where $\text{softplus}(x) = \log(1 + \exp(x))$. Distances and normalization are computed within each window w. This smooth penalty penalizes ordering violations and encourages separation between states at similar distances from the endpoint. The root-mean-square aggregation gives greater weight to comparisons with larger penalties. In our experiments, $\mathcal{L}_{\text{order}}$ sums this term over the two trajectory sources described in Appendix A.6.

3.4 REPRESENTATION SUFFICIENCY

We investigate whether geometric supervision can improve planning once task-relevant state variables are recoverable. Therefore, we seek a signal that indicates when representation sufficiency is reached, allowing us to freeze the encoder and continue training the adapter and predictor in latent space. Let $s \in \mathcal{S}$ denote selected task-relevant state variables. For each representation r, we fit a representation-specific MLP probe q_{η_r} on a training split, and measure held-out recovery error:

$$\epsilon_{\text{probe}}(r) = \underset{(o,s)\in\mathcal{D}_{\text{test}}}{\text{median}} \left\| q_{\eta_r}(r(o)) - s \right\|_2 . \tag{13}$$

We call a representation *sufficient* for the chosen variables, probe family, and data distribution when this error falls below a task-specific threshold. For example, in TwoRoom, $s = (x, y)$ and we use sub-pixel median position error as sufficiency criterion.

This test is sufficient but not necessary, as the probe is not guaranteed to recover the information encoded in the representation.

4 EXPERIMENTAL RESULTS

We use LeWorldModel with 192-dimensional latent representations. The long-range TwoRoom benchmark uses 300 fixed start–goal pairs, with goals taken 75 primitive actions after the start along recorded trajectories. A trial succeeds if the agent comes within 3 pixels of the goal within 150 primitive actions. We solve the planning problem using either CEM (Rubinstein, 1999) or Adam (Kingma & Ba, 2014). CEM uses 300 candidates, 30 elites, and ten iterations. In TwoRoom, each planned action a_k concatenates five consecutive primitive actions. The predictor estimates the latent state after this five-action block. With $H = 5$, each candidate plan therefore contains 25 primitive actions. We execute the selected sequence, then encode the new observation and replan.

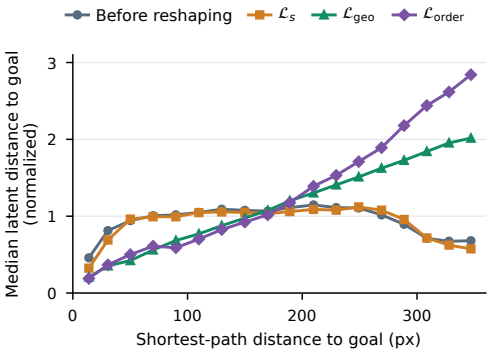


Figure 1: **Distance correlation study.** We plot latent distances before reshaping and after 10,000 adapter and predictor updates with $\mathcal{L}_s$, $\mathcal{L}_{\text{geo}}$, or $\mathcal{L}_{\text{order}}$ in TwoRoom. Each point shows the median distance within a shortest-path distance bin, divided by the model's median over all state–goal pairs. Adapted curves show means across three training seeds; bands show the minimum–maximum range.

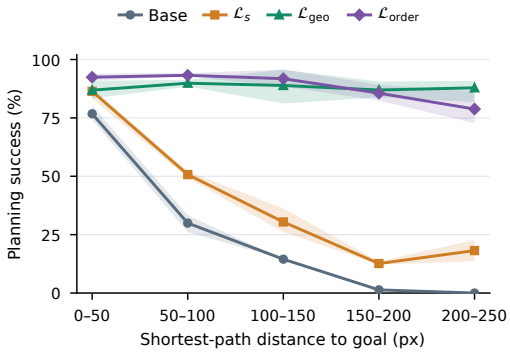


Figure 2: **Planning success study.** We plot planning success versus shortest-path distance from start to goal in TwoRoom. All methods use the same epoch-2 visual checkpoint and training budget. Base uses $\mathcal{L}_{\text{base}}$; the other methods add $\mathcal{L}_s$, $\mathcal{L}_{\text{geo}}$, or $\mathcal{L}_{\text{order}}$. Lines show means across three training seeds, and shaded bands show the minimum–maximum range. All methods are evaluated on the same 300 tasks.

4.1 Metric consistency improves planning performance

We compare four methods for latent reshaping: using only $\mathcal{L}_{\text{base}}$ as the objective, or one of the three losses introduced in Section 3.3. All four methods start from the same model after two epochs of visual training. We freeze the encoder and train the predictor for 2,000 updates, then jointly train the adapter and predictor for 20,000 updates. During joint training, Base uses $\mathcal{L}_{\text{base}}$ only; the other methods add to that objective either $\mathcal{L}_s$, $\mathcal{L}_{\text{geo}}$, or $\mathcal{L}_{\text{order}}$, respectively. We repeat these two stages with three training seeds, keeping the visual checkpoint and evaluation tasks fixed. For temporal ordering, we combine recorded dataset windows with paths generated by A^* and supervise their temporal order. We evaluate latent geometry after 10,000 adapter and predictor updates and planning after 20,000 updates. Appendix A gives the complete planning and replication protocol.

Metric consistency. Figure 1 shows how fine-tuning aligns latent and task-space distances. Before reshaping, we observe no clear correlation between the two, especially at large distances, indicating low metric consistency. After latent reshaping, the distances are strongly correlated, both for proportional task-distance supervision and for temporal ordering.

For a more qualitative evaluation of distance alignment, Figure 3 compares the spatial ordering induced by latent distances with shortest-path ordering for two goals. Before reshaping, latent distances vary irregularly across the environment and poorly reflect the paths through the doorway. Both $\mathcal{L}_{\text{geo}}$ and $\mathcal{L}_{\text{order}}$ produce spatial orderings that more closely resemble the shortest-path reference.

As a final evaluation of metric consistency, Figure 4 quantifies this improvement using the pairwise ordering error introduced in (5). At longer distances, the original representation gives errors around or above the 50% random-ordering level. Proportional task-distance supervision and temporal ordering reduce these errors, with $\mathcal{L}_{\text{geo}}$ achieving lower errors than $\mathcal{L}_{\text{order}}$ over most distances. Temporal straightening alone does not improve long-range ordering.

Planning performance. Next, we study how planner performance compares for the four reshaping methods. Figure 2 shows planning success as a function of shortest-path distance from start to goal. Across the 300 tasks, mean success is 27.4% for Base, 42.1% with straightening, 88.1% with shortest-path geometry, and 89.8% with temporal ordering. The differences are most pronounced at long distances. Base and straightening fall below 20% mean success in both bins beyond 150 px, while proportional task-distance supervision and temporal ordering maintain mean success above 75% in every bin. We show in Appendix A.3 that the improvement also holds with Adam, where

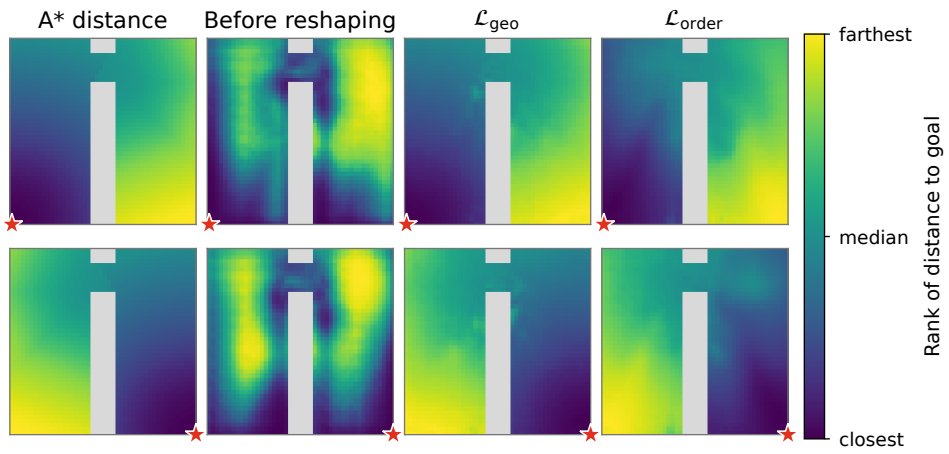


Figure 3: **Latent-distance visualization in 2D.** We plot the distances to two goals in TwoRoom before and after latent reshaping. Stars mark the goals. Colors rank states by distance to the goal, using shortest-path distance in the first column and latent distance in the remaining columns. Adapted maps average these ranks over three training seeds after 10,000 updates.

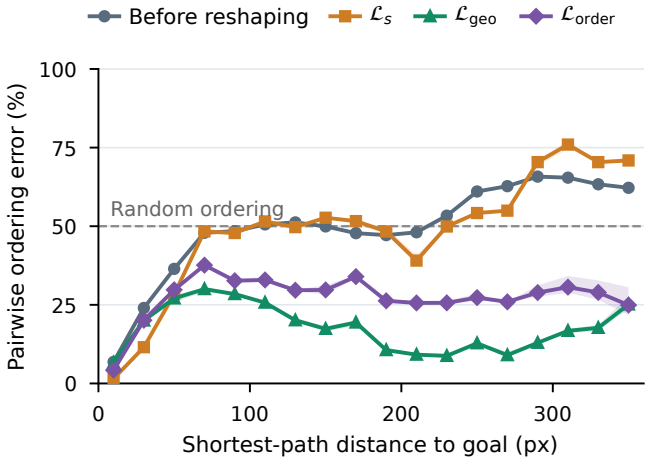
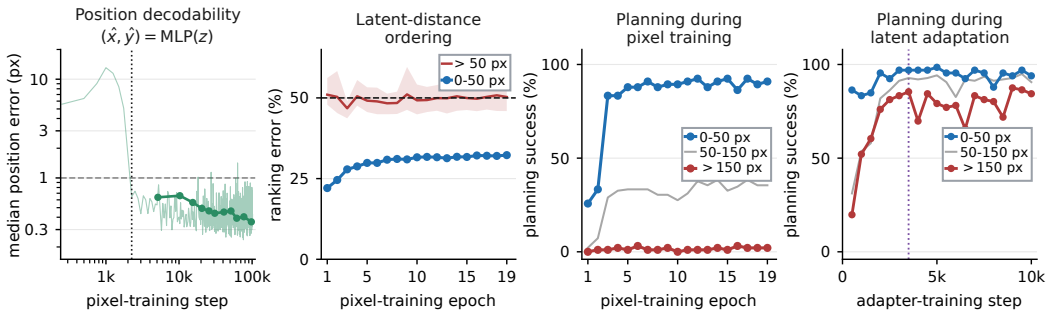


Figure 4: **Metric consistency study.** Pairwise ordering error before and after 10,000 adaptation updates in TwoRoom. Lower values indicate closer agreement with shortest-path ordering; 50% corresponds to random ordering. Adapted curves show means across three training seeds, with minimum–maximum bands.

proportional task-distance supervision raises mean planning success from 28.4% to 76.6% using the same checkpoints.

4.2 Early sufficiency enables latent-only reshaping

We next ask how early visual training can stop while still supporting effective latent reshaping. Figure 5a shows that representation sufficiency is reached after 2,250 visual-training updates, when median position-probe error first falls below one pixel. Figure 5b shows that ordering is more accurate near the goal than at longer distances. At short distances, ordering error increases during visual training while planning improves (Figure 5c). This diagnostic compares encoded states, while planning also depends on predicted trajectories. We see in Figure 5c that planning improves slowly without latent reshaping: success is 6.7%, 11.0%, and 32.0% at epochs 1, 2, and 3, respectively, then remains in the 32–39% range through epoch 19 and ends at 37.0%. For completeness, Figure 5d



a b c d

Figure 5: **State recovery and planning geometry develop at different rates.** Left to right: position-probe error; latent-versus-task ranking error; planning during visual training; and planning during adaptation from the frozen epoch-2 encoder. Distance bands use grid-based A* distances. Planning success rates are measured over 300 tasks. Panel (d) shows a separate single-seed adaptation run without the initial predictor-only stage. Shading in (b) shows the range across distance bands.

reports planning during latent reshaping with proportional task-distance supervision, starting from the frozen epoch-2 encoder. Its horizontal axis counts adapter and predictor updates after freezing.

We use the epoch-2 checkpoint, which already satisfies the representation sufficiency criterion.

Finally, Figure 6 shows one of our core results. We examine how the duration of visual training affects planning after latent reshaping. Each visual checkpoint receives 2,000 predictor-only updates followed by 20,000 adapter and predictor updates. With proportional task-distance supervision, mean planning success rises from 12.1% to 78.3% when visual training increases from 2,000 to 3,000 updates. Over the same interval, agent-position probe error falls below one pixel (Figure 5a). Starting from the epoch-2 checkpoint, the same procedure reaches 88.1%.

With the same adaptation budget, temporal straightening reaches at most 42.1% mean planning success across these four visual checkpoints (Appendix A.8).

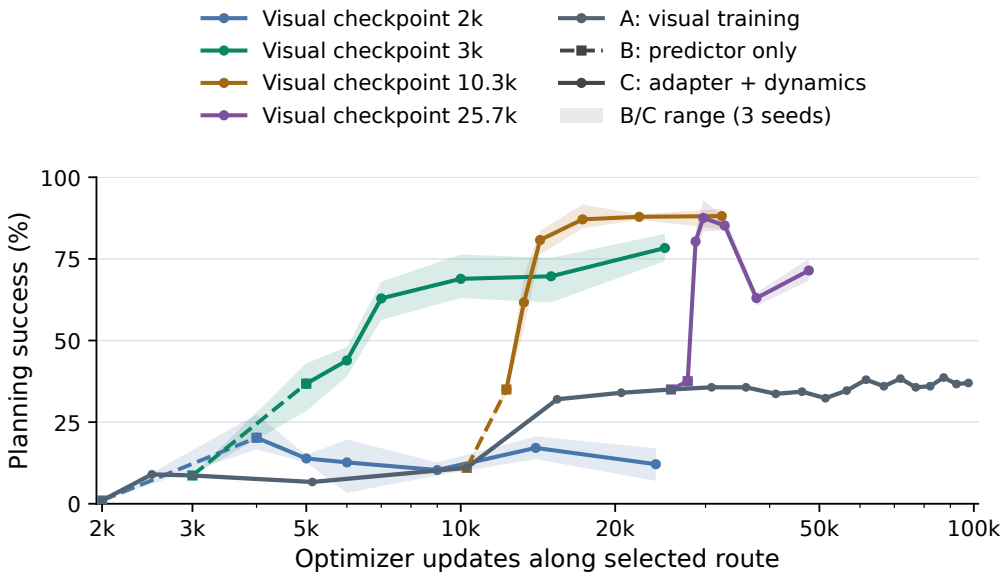


Figure 6: **Effectiveness of geometric reshaping with increasing amounts of visual training.** We use $\mathcal{L}_{\text{geo}}$ in addition to the base cost for training. Colors identify four visual-training checkpoints. The gray curve shows one phase-A visual-training run; dashed phase-B and solid phase-C segments show means across three independent training seeds, with minimum–maximum bands. All evaluations use the same 300 tasks.

5 Limitations

Our experiments focus on TwoRoom with LeWorldModel and use map information for shortest-path supervision and A^*-generated trajectories. The adapter and predictor are trained jointly, so the comparison measures the effect of adding geometric supervision to their training. Representation sufficiency is evaluated through agent-position recovery. Further experiments should assess the stopping criterion across environments and measure wall-clock training costs.

6 Conclusion

Representation sufficiency and metric consistency describe complementary properties of latent representations. In TwoRoom, both proportional task-distance supervision and temporal ordering improve metric consistency and long-range planning. An early sufficient representation supports these improvements through latent adaptation with a frozen visual encoder.

Future work should evaluate temporal ordering and representation sufficiency in other environments, including manipulation tasks such as PushT and Cube. Refining the stopping criterion could help balance visual training cost and planning performance across these tasks.

AI USE STATEMENT

Generative AI tools assisted with literature identification, manuscript restructuring and LaTeX formatting. All technical content was developed by the authors.

REPRODUCIBILITY STATEMENT

The model interface, adaptation objectives, and principal evaluation protocol are described in Sections 3–4. Appendix A documents planning settings, diagnostic data, geometric sampling, and replication. Appendix B provides details of the proposed objectives.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024.

Matthieu Destrade, Oumayma Bounou, Quentin Le Lidec, Jean Ponce, and Yann LeCun. Value-guided action planning with JEPA world models. World Modeling Workshop, 2026. arXiv:2601.00844.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Nicklas A. Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8387–8406, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations*, 2023.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

Arjun Parthasarathy, Nimit Kalra, Rohun Agrawal, Yann LeCun, Oumayma Bounou, Pavel Izmailov, and Micah Goldblum. Closing the train-test gap in world models for gradient-based planning. *arXiv preprint arXiv:2512.09929*, 2025.

Reuven Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1(2):127–190, 1999.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. arXiv preprint arXiv:2502.14819, 2025.

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.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, 2023.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim G. J. Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*, 2026.

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*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135, 2025.

A Experimental Details

A.1 Model and adaptation

The visual representation is the output of the LeWorldModel encoder and its original projection head, with dimension $d_z = 192$. Each rollout starts from a single encoded observation. The predictor then incorporates its own predictions into a context of up to three latent states. We distinguish the released model from checkpoints of the from-scratch visual-training run used in the sufficiency analysis.

In the controlled adapter experiments reported here, the visual encoder and original projector remain frozen. Predictor-only training updates the predictor, action encoder, and prediction head. Joint adaptation additionally updates the latent adapter, while the visual encoder and original projection head remain frozen. Table 1 gives the adapter architecture, optimization settings, and objective weights. Adapted next-state targets are not detached. SIGReg acts on adapted real representations. For task-distance supervision, the geometric term acts on pairs of real encoded observations and has no direct gradient path through the predictor.

Visual training uses a random $90/10$ split of dataset windows. For latent adaptation, we split the first $5{,}000$ episodes into $4{,}500$ training and 500 validation episodes, using seed 3072.

Parameter	Value
Objective weights	$\lambda_s = 1$, $\lambda_{\text{geo}} = 3$, $\lambda_{\text{order}} = 3$
SIGReg weight	0.09
Adapter hidden layers	Two layers of width 512, each LayerNorm $\rightarrow$ Linear $\rightarrow$ GELU
Adapter output	192 dimensions; residual scale 0.1
Adapter initialization	Final linear layer initialized to zero
Optimizer	AdamW, learning rate 10^{-4}, weight decay 10^{-3}
Prediction batch	64 windows of 8 frames
Ordering batch	8 dataset windows and 8 A^* paths; 9 states each
Ordering sampling	Dataset windows span 8–20 predictor steps; path states are uniform in arc length
Ordering comparisons	All $i < j < G$, excluding pairs at or below the frozen-distance floor
Ordering normalization	Numerical floor for ρ^2; gradients pass through ρ (Appendix A.6)

Table 1: Latent-adaptation settings. Probe fitting and evaluation-task selection are described in Appendices A.4 and A.2.

A.2 Planning protocol

TwoRoom is a visual navigation environment with two rooms connected by a doorway. The agent must reach a target position specified by a goal image, passing through the doorway when necessary. We count a trial as successful when the agent comes within 3 pixels of the target within 150 environment actions. This tolerance is tighter than the environment's default of 16 pixels and requires precise goal reaching. All compared methods use the same success criterion and evaluation tasks.

We select one start–goal pair per episode from 300 episodes with indices at least $8{,}000$, stratifying by shortest-path distance over approximately 0–250 px. The same tasks are used for all methods.

Table 2 summarizes the long-range TwoRoom evaluation. Each predictor step advances the latent state by five primitive actions. We plan over five such steps, giving a sequence of 25 primitive actions. The agent executes this sequence before replanning from its latest observation, within a total budget of 150 primitive actions.

Adam (Kingma & Ba, 2014) optimizes the same terminal latent cost with eight parallel initializations, 100 updates, and learning rate 0.03. Including final scoring, it uses 808 rollout evaluations, whereas CEM uses $3{,}001$. These counts do not equate computational cost: Adam also performs backward passes through predicted trajectories.

The goal offset is a separation along a recorded trajectory, not a guarantee of shortest-path length. Distance-resolved results instead group tasks using $d^\star$. Longer distance bands contain more tasks

Parameter	Value
Latent dimension	192
Predictor context	Up to 3 latent states
Primitive actions per model block	5
Planning horizon	5 blocks
Actions executed before replanning	25
CEM candidates per iteration	300
CEM elites	30
CEM iterations	10
Returned CEM sequence	Final elite mean
Reference-goal offset	75 primitive actions
Trial budget	150 primitive actions
Success radius	3 px
Evaluation tasks	300

Table 2: Long-range TwoRoom planning configuration.

requiring passage through the doorway, so distance and obstacle topology are not independently controlled.

A.3 Comparison of CEM and Adam

Table 3 compares CEM and Adam on the same Base and proportional task-distance models as Figure 2. Both planners use the same 300 start–goal pairs and the settings in Appendix A.2. Both benefit from geometric supervision, with CEM achieving higher success in the longer-distance bins.

		CEM		Adam	
$d^\star$ (px)	Tasks	$\mathcal{L}_{\text{base}}$	$\mathcal{L}_{\text{base}} + 3\mathcal{L}_{\text{geo}}$	$\mathcal{L}_{\text{base}}$	$\mathcal{L}_{\text{base}} + 3\mathcal{L}_{\text{geo}}$
$[0, 50)$	66	76.8 [74.2, 80.3]	86.9 [83.3, 90.9]	85.4 [78.8, 89.4]	97.0 [95.5, 98.5]
$[50, 100)$	69	30.0 [26.1, 33.3]	89.9 [88.4, 91.3]	32.9 [26.1, 37.7]	82.6 [81.2, 85.5]
$[100, 150)$	69	14.5 [14.5, 14.5]	88.9 [81.2, 95.7]	8.2 [7.2, 10.1]	75.8 [75.4, 76.8]
$[150, 200)$	74	1.4 [1.4, 1.4]	86.9 [83.8, 90.5]	0.5 [0.0, 1.4]	59.0 [54.1, 62.2]
$[200, 250)$	22	0.0 [0.0, 0.0]	87.9 [81.8, 90.9]	1.5 [0.0, 4.5]	57.6 [50.0, 63.6]
All	300	27.4 [26.0, 28.3]	88.1 [84.0, 90.3]	28.4 [26.3, 29.7]	76.6 [74.7, 78.3]

Table 3: Planning success (%) with CEM and Adam in TwoRoom. Models start from the same epoch-2 visual checkpoint and receive 2,000 predictor-only updates followed by 20,000 adapter and predictor updates. Values are means across three training seeds; brackets show minimum–maximum ranges.

A.4 State probes and ranking diagnostics

We fit a new two-hidden-layer position probe for each representation. The reported error is the median Euclidean coordinate error in pixels. The probe fitted during visual training (Figure 5a) and the separate before/after adaptation probe use the settings in Table 4.

The training-stage geometry study (Figure 5b) uses 24 goals and 9,000 test states, giving 216,000 state–goal distances. For each goal and each distance band, we draw 40,000 pairs of states with replacement and retain comparisons whose shortest-path distances differ by more than 1 px. All checkpoints use the same comparisons, and the two states in each comparison share a goal. In Figure 5b, the red curve averages the four bands beyond 50 px, and the shading shows the range across these bands.

	During visual training	Before/after adaptation
Hidden widths	256, 256	256, 256
Activation	ReLU	GELU
Optimizer	AdamW	Adam
Learning rate	3×10^{-3}	10^{-3}
Weight decay	10^{-4}	10^{-4}
Batch size	1,024	1,024
Fitting budget	1,500 updates	Up to 100 epochs; validation patience 15
Training frames	5,954	30,000
Validation frames	—	9,000
Test frames	1,989	9,000
Evaluation frequency	Every 250 visual updates	Before and after 2,500 adaptation updates

Table 4: Position-probe fitting protocols. Training frames come from episodes $[0, 5000)$ and test frames from $[6500, 8000)$. The before/after protocol samples six frames per episode and uses episodes $[5000, 6500)$ for validation. Frame samples are fixed across checkpoints.

For the distance-resolved diagnostic in Figure 4, each 20 px band contains up to $8{,}000$ comparisons across 16 goals. Each comparison uses two states relative to the same goal, with a shortest-path distance difference of 5–20 px. There are $144{,}613$ comparisons across the 19 bands.

Ranking comparisons can share states and goals. Their number should therefore not be interpreted as a number of independent Bernoulli trials. The random-ordering reference is an expected error of 0.5, not a claim that every distance band is exactly at chance.

A.5 Shortest-path supervision

The reference distance is computed with A* on a discretized map. It approximates continuous collision-free path length and is not an exact continuous-space geodesic.

The pair pool is balanced across nineteen half-open distance bands:

$$[0, 20), [20, 40), \ldots, [360, 380) \quad \text{pixels}.$$

Each band contains 128 pairs, giving $2{,}432$ pairs in the supervision pool. An update samples four pairs from every band, producing a 76-pair geometry batch. The batch-dependent scale in Equation (9) is fitted from these pairs.

A.6 Temporal-ordering supervision

We compute the temporal-ordering loss separately on dataset windows and A^* paths, then sum the two terms with equal weights:

$$\mathcal{L}_{\text{order}} = \ell_{\text{order}}(\mathcal{B}_{\text{data}}) + \ell_{\text{order}}(\mathcal{B}_{\text{path}}). \tag{14}$$

The resulting objective is $\mathcal{L}_{\text{base}} + 3\mathcal{L}_{\text{order}}$. Each ordering update uses eight dataset windows and eight A^* paths, with nine states per window or path. Dataset windows span 8–20 predictor steps; A^* states are sampled at uniform arc-length intervals along the full path. All pairs $i < j < G$ are compared, except those whose squared distance in the frozen representation is at most ε. Within each source batch, $\varepsilon = 10^{-6}m + 10^{-12}$, where m is the median positive squared pairwise distance in the frozen representation. We use $\rho_w = \sqrt{\max(\mathbb{E}_{i<G}[d_i^2], \varepsilon)}$. Gradients pass through ρ_w; the floor is computed from detached frozen representations.

A.7 Replication and checkpoint provenance

The objective comparison, A–B–C trajectories, and planner comparison use three training seeds per objective and visual checkpoint. The A–B–C comparison across visual checkpoints uses Base, $\mathcal{L}_s$, and $\mathcal{L}_{\text{geo}}$; $\mathcal{L}_{\text{order}}$ is evaluated from the epoch-2 checkpoint. For the planner comparison, CEM and

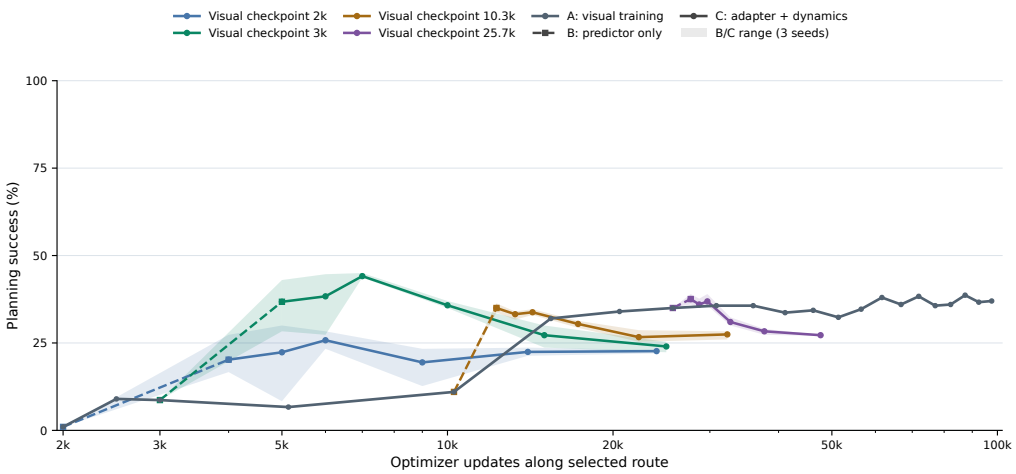


Figure 7: **Effectiveness of fine-tuning with increasing amounts of visual training.** Colors identify the four visual checkpoints. The gray curve shows visual training; dashed segments show predictor-only training and solid segments show joint adapter and predictor training. After freezing, lines show means across three training seeds, with minimum–maximum bands. All evaluations use the same 300 tasks.

Adam evaluate the same six models: three trained with $\mathcal{L}_{\text{base}}$ and three with $\mathcal{L}_{\text{base}} + 3\mathcal{L}_{\text{geo}}$. All six models use the same epoch-2 visual encoder, $2{,}000$ predictor-only updates, and $20{,}000$ adapter and predictor updates.

Without the initial predictor-only stage, $2{,}000$ joint adapter and predictor updates yield 34.3% planning success with the base objective and 83.0% when proportional task-distance supervision is added (Table 5). Each condition uses one training seed.

Objective	Planning success (%)
$\mathcal{L}_{\text{base}}$	34.3
$\mathcal{L}_{\text{base}} + 3\mathcal{L}_{\text{geo}}$	83.0

Table 5: Direct adaptation from the frozen epoch-2 visual checkpoint for $2{,}000$ updates. CEM evaluates the same 300 tasks.

Figure 5d shows a separate adaptation run without the initial predictor-only stage. Planning success reaches 91.3% after $3{,}500$ updates. Median position-probe error is 0.587 px before adaptation and 0.583 px after $2{,}500$ updates. On the same test states, metric-consistency error decreases from 0.395 to 0.170.

The 300 planning tasks are organized into ten groups of 30. Repeated evaluation or variation across these groups does not estimate variation across independently trained world models. The pixel-training and longer adaptation trajectories likewise concern checkpoints along training runs rather than independent replications of every point.

A.8 Planning trajectories across visual checkpoints

Figures 7 and 8 extend the comparison in Figure 6 to Base and temporal straightening across the same four visual checkpoints. The epoch-2 Base and $\mathcal{L}_{\text{geo}}$ endpoints are the checkpoints evaluated with both planners in Table 3.

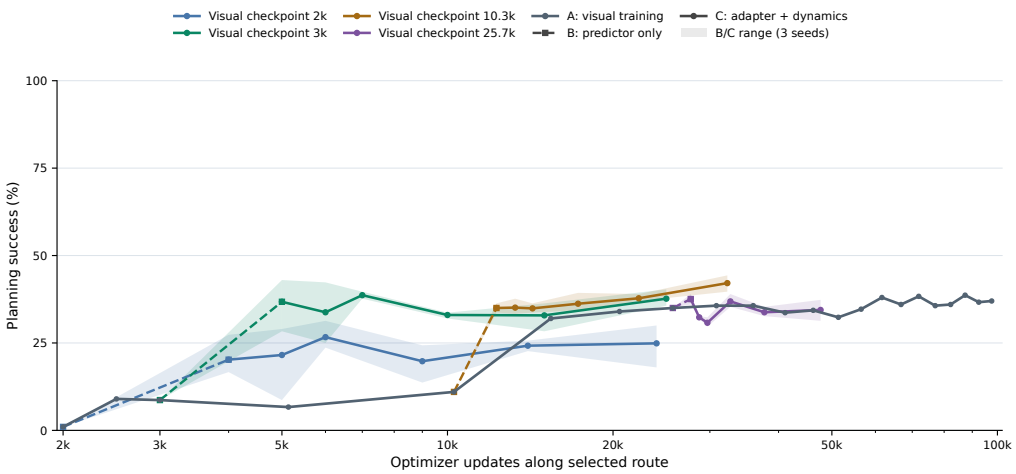


Figure 8: TwoRoom planning success with temporal straightening. Phase C adds $\mathcal{L}_s$ to $\mathcal{L}_{\text{base}}$. Colors, training stages, and uncertainty bands follow the Base control in Figure 7. All evaluations use the same 300 tasks.

A.9 Straightening-weight sensitivity with and without encoder freezing

We compare full-model tuning with frozen-encoder adaptation across straightening weights on the same 300 short TwoRoom tasks. Every checkpoint is trained for 10,000 adaptation updates and evaluated with a 16 px success radius.

Figure 9 shows that frozen-encoder adaptation retains higher planning success at the largest tested straightening weights. Preliminary end-to-end and adapter-based sweeps use different optimization recipes, so they show that both parameterizations can reach strong performance but do not support a matched comparison of stability or final performance.

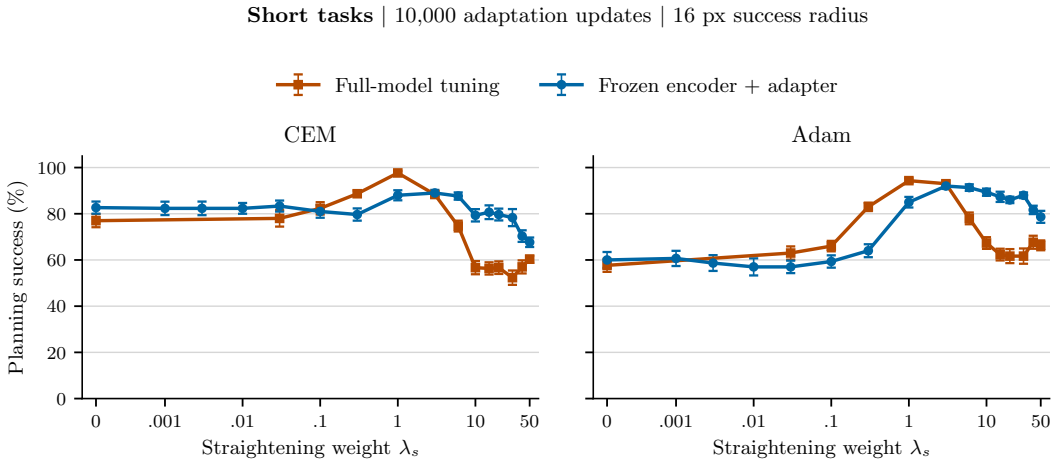


Figure 9: Straightening-weight sensitivity on short TwoRoom tasks. Full-model tuning updates the visual encoder, while frozen-encoder adaptation trains a residual adapter and predictive components. Each point corresponds to one trained model. Error bars show the standard error across ten groups of 30 evaluation tasks and therefore reflect task variation, not variation across training seeds.

B ADDITIONAL DETAILS ON GEOMETRIC OBJECTIVES

For a geometry batch $\mathcal{B}$, the fitted scale minimizes

$$Q_{\mathcal{B}}(\alpha) = \sum_{(i,j)\in\mathcal{B}} \left(\ell_{ij} - \alpha d^{\star}_{ij}\right)^2 . \tag{15}$$

Differentiating gives

$$\frac{\partial Q_{\mathcal{B}}}{\partial \alpha} = 2\alpha \sum_{(i,j)\in\mathcal{B}} (d^{\star}_{ij})^2 - 2 \sum_{(i,j)\in\mathcal{B}} \ell_{ij} d^{\star}_{ij}, \tag{16}$$

yielding Equation (9) when the denominator is positive. The fitted value is then detached before evaluating the geometric residuals.

The Huber penalty is

$$h_{\delta}(e) = \begin{cases} \frac{1}{2}e^2, & |e| \le \delta, \\ \delta\left(|e| - \frac{1}{2}\delta\right), & |e| > \delta, \end{cases} \qquad \delta = 1. \tag{17}$$

The scale is therefore an ordinary least-squares fit, not the minimizer of this Huber objective. Its closed form removes the need to tune a fixed physical-to-latent scale, but does not remove the geometric-loss weight or Huber threshold.

If every adapted latent is identical, then $\ell_{ij} = 0$, $\alpha_{\mathcal{B}} = 0$, and the geometric loss vanishes. The proportional-distance objective alone is thus not a non-collapse mechanism.