# Consolidating Specialist Representations into a Shared World Model

Anonymous authors
Paper under double-blind review

## Abstract

Independently trained world-model specialists provide task-specific representations whose latent coordinates need not be compatible with a shared predictor. We study how to consolidate these representations into one task-conditioned world model. Our approach freezes specialist encoders, aligns their latent spaces, and trains a shared model with cached supervision and a gradual transition to student representations. For quantized supervision, applying the same similarity map to a latent space and its codebook preserves assignments within that codebook. We compare continuous, discrete, and hybrid supervision on two-, three-, and six-task collections of pixel-based control problems. In our current single-training-seed evaluation, continuous and hybrid consolidation achieve 80.3% macro-average success on six tasks, compared with 50.0% for our native joint-training baseline; aligning hybrid supervision improves its point estimate from 75.3% to 80.3%. These results motivate further study of representation alignment for specialist consolidation, while leaving its robustness across training seeds and the causes of supervision-type differences to be established.

## 1 Introduction

A world model supports control by predicting how actions change a latent representation of the environment. Sharing one model across tasks is attractive at deployment, but the standard route, joint training on all tasks, couples representation learning, dynamics learning, and inter-task trade-offs in a single optimization. The cost is visible in our experiments: a native six-task model we train reaches 92% closed-loop success on one task while collapsing to 2% on another, although a capable single-task specialist exists for every task in the collection.

We therefore start from the specialists and ask how to consolidate what they have already learned into one shared model. Each specialist has solved, for its own task, the problems that joint training must solve for all tasks at once; the open problem is the supervision interface between specialists and a shared student. That interface has three parts: independently trained encoders assign unrelated coordinates to the same image, so raw teacher features are not comparable across tasks; a task's discrete codebook is meaningful only relative to its own encoder's geometry, so quantized supervision must move together with its representation; and the interface is useful only if the learned representation supports closed-loop planning—a property that reconstruction or feature-similarity measurements do not establish, so we evaluate every design choice through control.

Our pipeline (Figure 1) proceeds in three offline stages. Extract: freeze every specialist encoder, fit a task-local codebook in its latent space, and cache per-frame latents, code assignments, and soft assignments. Align: fit a similarity map from each source space onto a chosen reference space and apply the same map to that task's latents and its codebook; because the map rescales all within-codebook distances by a common factor, nearest-code assignments are preserved exactly. Consolidate: train one task-conditioned student—shared encoder, predictor, and heads, distinguished only by task embeddings and a common padded action interface—from the cached supervision, while a schedule shifts the predictor's inputs and targets from teacher to student latents. Deployment uses the student alone: no specialist



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.

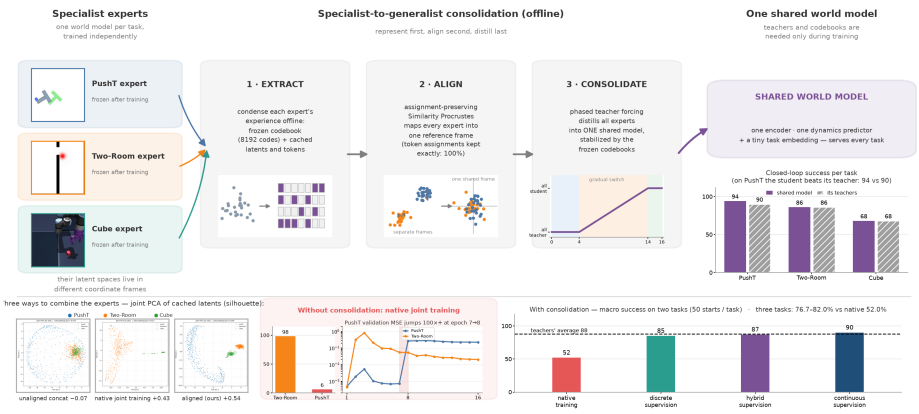


Figure 1: Specialist-to-generalist consolidation. Top: three frozen, independently trained experts live in different latent coordinate frames; offline stages condense each expert's experience into a frozen codebook with cached latents and tokens (Extract), map every expert into one reference frame with assignment-preserving similarity transforms (Align, Section 3.2), and distill all experts into one shared model through a gradual teacher-to-student switch (Consolidate); teachers and codebooks are needed only during training. Bottom: joint PCA of the cached latents shows the experts' frames do not coincide (left); native joint training on two tasks collapses PushT control while its validation error jumps by more than two orders of magnitude at epochs 7–8 (middle; Appendix C.5); with consolidation, two-task macro success reaches 85–90% versus 52% for native training (right; dashed line: teacher average; point estimates), three-task results reach 76.7–82.0%, and six-task results appear in Table 2.

encoders, no caches, no codebook lookups. One objective covers three supervision choices—continuous, discrete, and hybrid—that differ only in what the teacher signal is made of, so their control consequences can be compared under a fixed pipeline.

Our contributions are:

- a consolidation pipeline for closed-loop control that turns frozen specialist representations, and optionally their codebooks, into aligned cached supervision for one task-conditioned student whose deployment requires no teachers, caches, or codebook queries;
- an alignment treatment with an exact interface property—one similarity map applied to a latent space and its codebook preserves within-codebook nearest-code assignments, proved and audited in floating point—measured against an unaligned-consolidation control on held-out feature pairs and closed-loop success;
- a task-wise comparison of the three supervision choices on two-, three-, and six-task pixel-control collections, reporting macro, worst-task, and teacher-gap structure, quantization diagnostics, and failure cases.

In our current single-training-seed evaluation, consolidation raises six-task macro-average success from 50.0% under native joint training to 75.3% unaligned and 80.3% aligned hybrid; continuous supervision matches the aligned hybrid point estimate, discrete supervision reaches 79.0%, and the six specialists average 81.0%. Two effects must be kept apart: introducing specialist supervision without alignment already accounts for 25.3 of the 30.3 percentage points between the native baseline and aligned hybrid, and the complete alignment treatment contributes the remaining 5.0, with one task (Scene) moving down; all results are single-run point estimates, and we flag where this limits interpretation.

# 2 Related Work

## 2.1 Consolidating specialists into one shared model

Training specialists and consolidating their knowledge is an established alternative to joint optimization: knowledge distillation transfers teacher predictions to a student (Hinton et al., 2015), and multi-agent variants consolidate task-specific policies (Rusu et al., 2016; Parisotto et al., 2016; Teh et al., 2017). Representation compatibility is likewise addressed by knowledge amalgamation (Shen et al., 2019), where Luo et al. (2019) learn a common feature space for heterogeneous teachers before training the student. Closest in spirit, vocabulary-oriented model fusion aligns token spaces of independently trained language models before merging (Wan et al., 2024a;b; Zhang et al., 2025; 2026b). Neither multi-teacher consolidation nor vocabulary compatibility is unique to our setting; our question is how to use them for action-conditioned latent dynamics while retaining the specialists' closed-loop control, with continuous, discrete, and hybrid targets compared under one pipeline.

## 2.2 Aligning representation spaces and vocabularies

Our alignment stage uses established geometry: bilingual embedding alignment fits linear maps from paired examples (Mikolov et al., 2013), constrains them to the orthogonal Procrustes problem (Xing et al., 2015), and admits closed-form solutions (Smith et al., 2017). Model stitching connects separately trained networks through learned interfaces (Bansal et al., 2021); relative representations communicate through shared-anchor similarities (Moschella et al., 2023); and the Platonic representation hypothesis frames cross-model representational convergence (Huh et al., 2024). In world models specifically, Li et al. (2026) report geometric correspondence and stitching compatibility, and multi-way generalized Procrustes analysis aligns several models into one frame (Achara et al., 2026), of which our per-task fits against one fixed reference are a simple instance. Vocabulary transfer also predates this work: WECHSEL initializes subword embeddings by dictionary-anchored Procrustes (Minixhofer et al., 2022). We instead transport each specialist's features and code vectors together and distinguish the resulting task-local assignment invariance from the assignments and soft targets recomputed on the pooled bank; the empirical question is whether the resulting supervision helps a shared predictor, beyond the algebraic fact.

## 2.3 World models for control

Latent world models support planning from pixels (Hafner et al., 2019), and the JEPA family formalizes prediction in embedding space rather than pixel space (Assran et al., 2023; Bardes et al., 2024). DINO-WM plans in frozen pretrained visual features (Zhou et al., 2024), and LeWorldModel (LeWM)—which supplies our specialists—trains such an architecture stably end to end from raw pixels (Maes et al., 2026b). Our experiments are built on the stable-worldmodel platform (Maes et al., 2026a), whose standardized baseline implementations, specialist checkpoints, and control-evaluation protocols are the infrastructure this paper starts from; the native joint-training baselines we evaluate are its LeWM-based reference models, making it the closest baseline line of work. Shared models can also be trained directly: TD-MPC2 conditions one model on tasks (Hansen et al., 2024), Newt combines demonstration pretraining with language-conditioned multi-task learning (Hansen et al., 2025), and Mixture-of-World Models introduces modular latent dynamics (Zhang et al., 2026a); task interference under joint training is well documented (Standley et al., 2020), and our native baseline is one implementation at the evaluated scale rather than a claim about joint learning in general. World-model distillation is also established (Yamada et al., 2023; Kuzmenko & Shvai, 2025; Robertshaw et al., 2026), but starts from a single generalist or pools trajectories; we start from several fixed specialists and construct the representation interface for one task-conditioned student. Discrete latents have substantial precedent (van den Oord et al., 2017; Hafner et al., 2021; Micheli et al., 2023; Scannell et al., 2025), including feature-guided tokenization (Huang et al., 2024; Dong et al., 2023); we adopt these principles to build task-local targets and study how the supervision choice affects a shared control model.

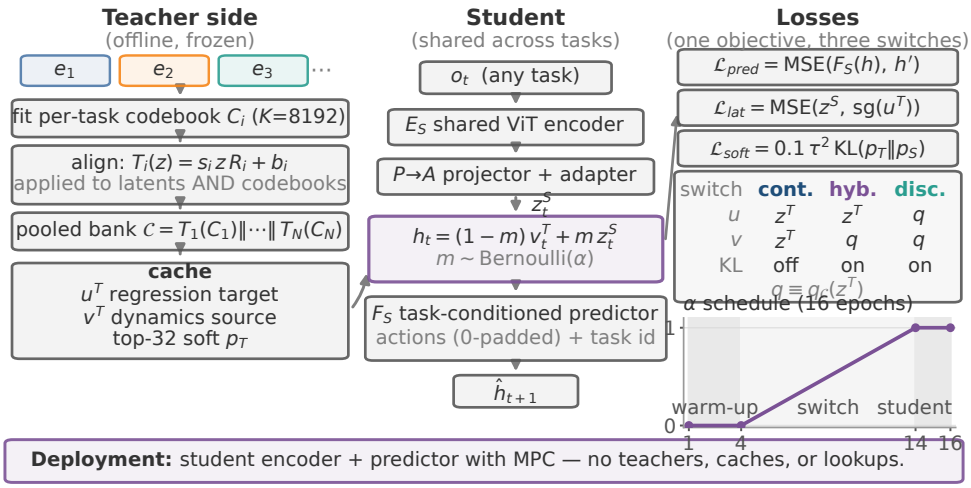


Figure 2: The consolidation pipeline and its single training objective. Left: frozen specialists supply per-task latents and codebooks; one similarity transform $T_i$ maps each source space and its codebook into the reference frame, and a pooled bank $\mathcal{C}$ plus a per-frame cache (regression target $u^T$, dynamics source $v^T$, soft targets $p_T$) are built offline. Middle: the shared, task-conditioned student mixes teacher and student latents through $h_t = (1-m)v_t^T + mz_t^S$ with $m \sim \text{Bernoulli}(\alpha)$ and predicts ahead. Right: the three losses and the switches that instantiate continuous, hybrid, and discrete supervision; the $\alpha$ schedule transitions from teacher-source to student-representation training over 16 epochs. Deployment keeps only the student and MPC.

## 3 Method

### 3.1 Setting and representation extraction

We are given $N$ tasks with offline trajectories $\mathcal{D}_i = \{(o_t, a_t, o_{t+1})\}$ and trained specialists. Let $e_i(o) \in \mathbb{R}^d$ denote the feature used by specialist $i$ for latent prediction, including its projection layers ($d = 192$ throughout). We freeze $e_i$ and use its outputs as representation supervision. The student shares its visual encoder, projector, adapter, predictor, and prediction head across tasks; a task embedding and a common padded action interface identify the relevant dynamics, and consolidation assumes access to task identity and each task's offline data (Figure 2; the full backbone with dimensions appears in Figure 6, Appendix B).

For quantized supervision, we fit and freeze a task-local codebook $C_i = \{c_{ik}\}_{k=1}^K$ in the original specialist space, with nearest-code quantization

$$q_{C_i}(z) = c_{ik^*}, \qquad k^* = \arg\min_k \|z - c_{ik}\|_2^2 \tag{1}$$

and $K = 8192$ in the multi-task experiments. Quantization error measures $\|z - q_{C_i}(z)\|_2$: a change in the target representation, not a lower bound on how accurately a student can fit that code vector. The continuous variant uses the frozen feature directly.

### 3.2 Aligning features and codebooks

Specialist coordinates are not comparable across tasks, so we map every source space into one reference before pooling supervision. We use PushT as the reference task, indexed by $r$. Feature pairs come from sampling images of the pooled task datasets through both $e_i$ and $e_r$; they compare two encodings of the same image, not correspondences between different physical states. For each source we fit a similarity map on the alignment-training partition,

$$T_i(z) = s_i z R_i + b_i, \qquad (s_i, R_i, b_i) = \arg \min_{s_i>0,\, R_i^\top R_i = I,\, b_i} \sum_{o \in \mathcal{A}_{\text{train}}} \|s e_i(o) R + b - e_r(o)\|_2^2, \tag{2}$$

Table 1: Supervision choices. All variants retain representation regression with weight 1. The predictor remains continuous, including in the variant called discrete supervision.

| Variant | Regression target $u_t^T$ | Dynamics source $v_t^T$ | Soft weight |
|---|---|---|---|
| Continuous (M4) | $z_t^T$ | $z_t^T$ | 0 |
| Discrete (M5) | $q_{\mathcal{C}}(z_t^T)$ | $q_{\mathcal{C}}(z_t^T)$ | 0.1 |
| Hybrid (M2) | $z_t^T$ | $q_{\mathcal{C}}(z_t^T)$ | 0.1 |

with row-vector features and $T_r$ the identity; centering, an SVD of the cross-covariance, and a scalar least-squares fit give the closed form (Appendix A). We apply the same frozen map to source features and their codebook: the transformed teacher feature of source task $i$ is $z^T = T_i(e_i(o))$, written $z_t^T$ at sequence time $t$, and the transformed codebook is $\widetilde{C}_i = T_i(C_i)$. Superscript $T$ marks teacher quantities; transposition is always written $^\top$. Held-out pairs measure how the fit generalizes beyond the images used to compute it.

Within-codebook assignment invariant. For any $s_i > 0$ and orthogonal $R_i$,

$$\|T_i(z) - T_i(c)\|_2^2 = s_i^2 \|z - c\|_2^2. \tag{3}$$

Multiplication by the common factor $s_i^2$ preserves the ordering of all distances, so a unique nearest-code index within $C_i$ is unchanged when feature and codebook are transformed together (ties require a consistent rule). This elementary property justifies transporting a task-local quantizer with its representation; it makes no claim about semantic agreement between tasks.

We form the final bank by concatenating the transformed codebooks, $\mathcal{C} = \widetilde{C}_1 \,\|\, \cdots \,\|\, \widetilde{C}_N$, retaining all $NK$ entries (the pipeline's candidate-merging stage accepted no candidates in our runs; Appendix B), and recompute hard assignments and soft targets against this entire bank. A nearest code from another task can now be selected, so Equation 3 does not guarantee preservation of these pooled assignments. For each frame, the cache stores $z_t^T$, its final hard index, and the indices and probabilities of its 32 nearest entries in $\mathcal{C}$.

Scaling also affects soft supervision: at fixed temperature the distance softmax shifts under $T_i$ (within one transformed codebook the compensating temperature is $s_i^2\tau$). Our runs use a fixed $\tau = 1$ on the final bank, so aligned-versus-unaligned comparisons evaluate the full transformation—norms, loss scales, and soft targets together—rather than rotation alone.

### 3.3 One objective, three supervision choices

The student representation is $z_t^S = A(P(E_S(o_t)))$, with visual encoder $E_S$, projector $P$, and learned linear adapter $A$. We distinguish the representation-regression target $u_t^T$ from the teacher source $v_t^T$ used for dynamics training; Table 1 defines the three instantiations—M2 (hybrid), M4 (continuous), M5 (discrete)—and Section 4 adds the unaligned control M0 and the native baseline M3.[1] Here $q_{\mathcal{C}}$ denotes quantization against the pooled bank.

To connect fixed teacher supervision to the deployed representation, we draw one Bernoulli variable $m$ per sequence and use it for every time step:

$$h_t = (1 - m)v_t^T + m z_t^S, \qquad m \sim \text{Bernoulli}(\alpha). \tag{4}$$

Let $F_S$ include the task-conditioned predictor and its output head; with three context steps,

$$\mathcal{L}_{\text{pred}} = \text{MSE}(F_S(h_{1:3}, a_{1:3}, i), h_{2:4}), \qquad \mathcal{L}_{\text{latent}} = \text{MSE}\big(z_{1:4}^S, \text{sg}(u_{1:4}^T)\big). \tag{5}$$

The teacher cache is constant. When $m = 1$ the prediction target $h_{2:4}$ is not detached: gradients reach the student through both the predictor input and the next-feature target, a convention shared by all three variants.

[1] A merged-vocabulary variant M1 was never run because the recorded codebook merge count is zero (Appendix B).

For soft supervision, let $I_t$ be the teacher's top-32 code indices; the teacher distribution is normalized on $I_t$ and zero elsewhere, and the student distribution is normalized over all entries:

$$p_T(k \mid t) = \frac{\mathbf{1}[k \in I_t] \exp(-\|z_t^T - c_k\|^2/\tau)}{\sum_{j \in I_t} \exp(-\|z_t^T - c_j\|^2/\tau)}, \tag{6}$$

$$p_S(k \mid t) = \frac{\exp(-\|z_t^S - c_k\|^2/\tau)}{\sum_{j=1}^{NK} \exp(-\|z_t^S - c_j\|^2/\tau)}. \tag{7}$$

The total objective averages across tasks, sequences, and time steps:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{latent}} + \lambda_{\text{soft}} \tau^2 \mathbb{E}_t \, \mathrm{KL}(p_T(\cdot \mid t) \,\|\, p_S(\cdot \mid t)) \,. \tag{8}$$

Continuous supervision disables only the KL term. Hybrid and discrete supervision share their quantized dynamics source and soft targets and differ in the regression target, so their comparison isolates that switch; continuous versus hybrid changes both source and KL term, so that comparison alone cannot identify either component.

### 3.4 Scheduled source mixing and planning

Training runs 16 epochs in three phases: four with $\alpha = 0$ (teacher-source warm-up), ten with $\alpha$ increasing linearly to one, and two with $\alpha = 1$ (Figure 2, right); representation supervision stays active throughout. The final phase trains on student encodings of recorded observations—offline training, not environment interaction or recurrent replacement by predicted frames.

At deployment, the model encodes current observations and goal image, predicts future features under candidate action sequences, and minimizes the latent goal-distance objective with MPC, updating the context as actions execute. This path uses only the student encoder, projector, adapter, conditioning, predictor, and prediction head; teacher encoders, alignment transforms, caches, and codebook lookups are training-time components.

## 4 Experimental Protocol

Tasks and models. We evaluate nested collections: PushT and Two-Room; those tasks plus Cube; and those three plus Scene, Reacher, and HumanoidMaze, each trained independently. The tasks span planar manipulation, navigation, articulated manipulation, and locomotion, with raw action dimensions from 2 to 21. All students use a 192-dimensional representation and a six-layer predictor; actions are grouped into blocks of five environment steps, zero-padded to a shared width, and encoded with task identity. Appendix B specifies architectures, interfaces, and specialist sources.

Comparisons. M3 is the platform's native joint-training LeWorldModel baseline (Maes et al., 2026b;a) with prediction loss and SIGReg latent-variance regularization, without specialist features. M0 uses hybrid consolidation with identity alignment and an unaligned concatenated bank; M2 adds similarity alignment, and M4 and M5 instantiate continuous and discrete supervision. We also report the original continuous specialists and separately trained single-task codebook students (hybrid configuration; not fully discrete teachers, and they do not supply the multi-task caches). Specialist performance is a reference point, not an upper bound.

Optimization and comparison scope. Consolidation uses AdamW, mixed bfloat16 precision, and the 4/10/2 schedule; six-task runs use a global batch of 384 sequences per task, and M3 trains 16 epochs with SIGReg weight 0.09 (full configuration in Appendix B). Epoch counts alone do not establish equal optimizer updates, task exposure, or tuning budgets, so we read M3 as the evaluated native baseline rather than an exhaustive test of joint training; the results also do not measure the cost of training the specialists.

Table 2: Six-task closed-loop success (%). Macro averages weight tasks equally; Min is the lowest task score. These are point estimates under Section 4's evaluation. The specialist and single-task student rows use one model per task (the latter trained with the hybrid configuration); the remaining rows each use one shared model.

| Model | PushT | Two-Room | Cube | Scene | Reacher | HumanoidMaze | Macro | Min |
|---|---|---|---|---|---|---|---|---|
| Specialists | 90 | 86 | 68 | 64 | 94 | 84 | 81.0 | 64 |
| Single-task students | 78 | 84 | 68 | 64 | 98 | 86 | 79.7 | 64 |
| Native joint (M3) | 2 | 92 | 60 | 52 | 6 | 88 | 50.0 | 2 |
| Unaligned hybrid (M0) | 58 | 88 | 70 | 60 | 88 | 88 | 75.3 | 58 |
| Aligned hybrid (M2) | 88 | 90 | 70 | 54 | 92 | 88 | 80.3 | 54 |
| Aligned continuous (M4) | 86 | 88 | 70 | 62 | 94 | 82 | 80.3 | 62 |
| Aligned discrete (M5) | 76 | 90 | 70 | 60 | 92 | 86 | 79.0 | 60 |

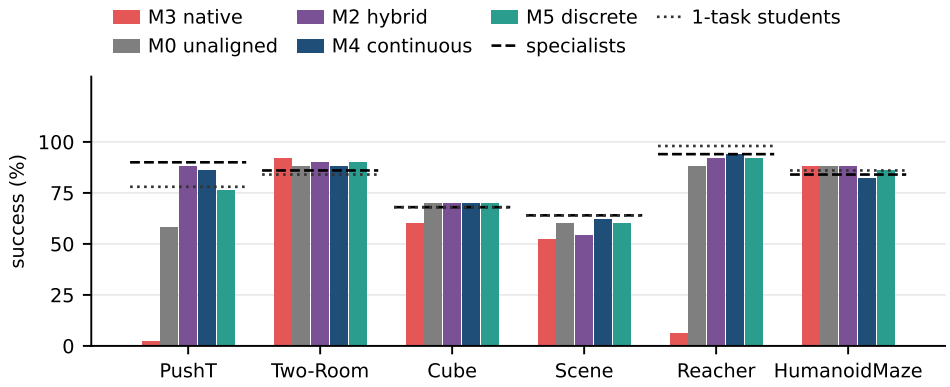


Figure 3: Six-task per-task success for the shared models of Table 2 (50 fixed starts per task). Dashed lines mark the original continuous specialists; dotted lines mark the separately trained single-task hybrid students (the two reference lines coincide on Cube and Scene, where the values are equal). Exact values appear in Table 2.

Evaluation and reporting. The reported runs use training seed 3072 and evaluate 50 fixed starts per task with MPC seed 42, shared by all methods within a collection. CEM uses 300 candidate sequences, 30 iterations, and 30 elites; planning and receding horizons are five action blocks. We report task success percentages, their equally weighted macro-average, and the minimum task score. These are observed point estimates without equivalence or significance claims; the two-point resolution of a task score is a counting increment, not a confidence interval, and diagnostics with their own episode counts are labeled where used. Further controlled evaluations are specified in Appendix D, which contains no result claims.

## 5 Results

### 5.1 Specialist supervision restores closed-loop control

Table 2 and Figure 3 show that consolidation improves the observed multi-task result over the native baseline. M3 reaches 92% on Two-Room and 88% on HumanoidMaze but only 2% on PushT and 6% on Reacher. Introducing specialist supervision without alignment (M0) raises the macro-average from 50.0% to 75.3%, recovering PushT and Reacher to 58% and 88%. Aligned hybrid and continuous supervision each reach 80.3%, close in point estimate to the specialists' 81.0%, which does not establish equivalence.

The task-level breakdown separates broad improvement from remaining weaknesses: M4 attains the highest minimum task success (62%, versus 2% for M3 and 54% for M2), while Scene stays below its 64% specialist reference under every shared model. The breakdown also

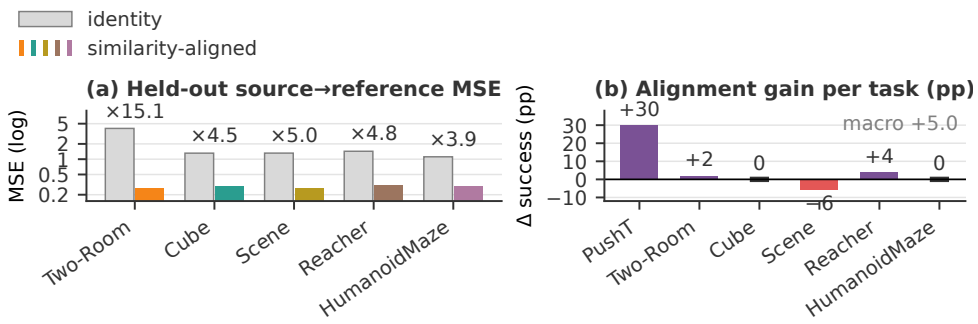


Figure 4: Alignment evidence on the six-task collection. (a) Held-out source-to-reference MSE before (identity) and after fitting the similarity map, for the five source tasks (PushT is the reference frame); annotation gives the improvement factor. (b) Per-task change in closed-loop success when alignment is added to hybrid consolidation (M2−M0, percentage points); the aggregate gain of +5.0 macro points includes a negative effect on Scene. Point estimates; held-out alignment diagnostics in Appendix C.2.

shows why improvement should not be attributed to alignment alone: specialist supervision without alignment already accounts for 25.3 of the 30.3 percentage points between M3 and M2.

### 5.2 The alignment treatment helps hybrid consolidation, with task-dependent effects

Alignment increases the hybrid macro-average in every evaluated collection: M2 improves on M0 by +7.0, +4.7, and +5.0 points on the two-, three-, and six-task collections (Table 7, Appendix C); only the M2−M0 pairing speaks to alignment, since the continuous and discrete variants differ from M0 in supervision type as well. Figure 4(b) shows where the six-task gain comes from: +30 points on PushT, +4 on Reacher, +2 on Two-Room, no change on Cube and HumanoidMaze, and −6 on Scene—the positive aggregate contains a negative task effect. Because M0 and M2 also differ in transformed target scales and soft distributions (Section 3.2), these observations support the complete alignment treatment without identifying a separate geometric cause; a scale-matched control is specified in Appendix D.2.

Held-out feature measurements complement this view (Figure 4(a)): fitting reduces source-to-reference MSE by factors of 3.86–15.08 and reaches cosine similarities of 0.72–0.75 with $R^2$ of 0.146–0.240 (Appendix C.2)—a partial correspondence, not complete feature agreement. Task-local nearest-code preservation is 1.0 for four source encoders and 0.999949 for Reacher in the floating-point audit, consistent with Equation 3; the audit does not measure cross-task code reuse in the pooled cache.

### 5.3 Continuous and quantized supervision trade control differently

The three supervision forms reach similar macro-averages on six tasks (80.3, 80.3, 79.0), but the per-task results in Table 2/Figure 3 show the differences behind the aggregate: M2 leads on PushT and HumanoidMaze and ties on Two-Room; M4 leads on Scene and Reacher; and M5 gives up 12 points of PushT success relative to M2 in exchange for a six-point gain on Scene and a two-point loss on HumanoidMaze. Since M2 and M5 share the dynamics source and KL term, their comparison localizes the change to the regression target; continuous versus hybrid changes two components at once, so neither comparison isolates quantization.

Single-task measurements show why quantization needs its own control evaluation (Figure 5(a)): on PushT, increasing $K$ from 512 to 8192 cuts mean relative feature reconstruction error from 49.73% to 20.62%, discrete-supervision success rises from 17.5% to 62.0%, while hybrid success stays between 74.0% and 76.0%; on Cube, discrete success rises from

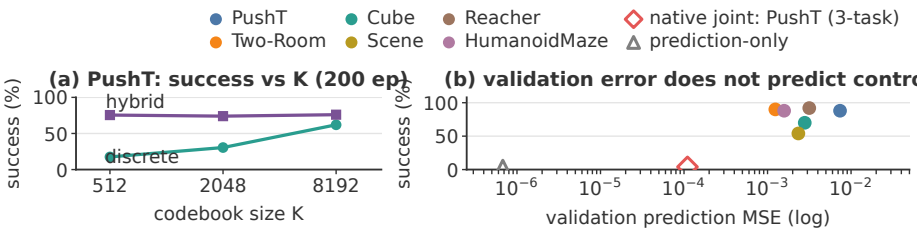


Figure 5: Quantization sensitivity and loss diagnostics. (a) PushT codebook-size sweep (200 held-out episodes): discrete-supervision success rises with $K$ while hybrid supervision changes little; point estimates, not fitted trends (Cube sweep in Table 9). (b) Validation prediction MSE vs. closed-loop success: the six consolidated tasks (filled), the native joint baseline on PushT (three-task run, approximate final MSE), and the prediction-only run (MSE at epoch 40; success of its selected epoch-10 checkpoint, 200 held-out episodes). Failing models validate as low as or lower than every succeeding task.

54% to 70% (Table 9). These observations are consistent with task-dependent sensitivity to target quantization; they do not show that quantization explains every consolidation gap.

### 5.4 Validation prediction error does not diagnose control

Low validation error and poor control coexist across settings (Figure 5(b)). In the three-task native model, PushT validation MSE is approximately $1.1 \times 10^{-4}$ despite 4% success; in the six-task native model, PushT and Reacher validate at $9.8 \times 10^{-5}$ and $9.5 \times 10^{-5}$ (final-epoch records) while succeeding on 2% and 6% of starts. A prediction-only PushT run (no specialist supervision, no SIGReg) drives validation MSE to $6.768 \times 10^{-7}$, yet its selected checkpoint only reaches 3.5% success over 200 held-out episodes, while the consolidated model's tasks validate one to two orders of magnitude higher and succeed 54–92% of the time. The prediction-only diagnostic removes two constraints at once, so it cannot attribute the failure to either; internal losses must be checked against closed-loop evaluation.

## 6 Discussion and Limitations

Frozen specialist features can supervise one predictive model with useful control across heterogeneous tasks, turning consolidation into a supervised interface problem rather than a from-scratch multi-task optimization; alignment asks the narrower question of whether preprocessing helps, and the completed runs answer positively for the hybrid macro estimate, with task-specific exceptions (Section 5.2).

The scope of these conclusions is bounded. All specialists share one latent dimension, the reference is fixed to PushT, and a similarity map cannot correct arbitrary nonlinear differences between encoders; the pooled bank retains task-origin entries rather than a compact shared vocabulary; and collections are consolidated once, offline, so these runs make no claims about continual retention, scaling, or capacity independence. Specialist sources are heterogeneous, the native baseline's optimization is not budget-matched, and all numbers are single-run point estimates (Appendix D). Scene's residual failure and the supervision-dependent trade-offs show that matching supervision to task quantization difficulty remains open.

## 7 Conclusion

We studied how to consolidate independently trained specialist representations into one shared world model: cache frozen features and codebooks, map each source space and codebook into a common frame, and train a task-conditioned student that gradually takes over its own representation, substantially outperforming native joint training. Consolidation is thus a practical route to shared predictive control.

## Reproducibility Statement

Section 3 defines the transformations, training objective, gradient conventions, and inference procedure. Section 4 and Appendix B describe the model, datasets, optimization, and evaluation settings. Appendix A proves the within-codebook invariance, and Appendix C reports additional completed measurements. Tables distinguish observed success estimates from unexecuted validation protocols. Anonymized training, alignment, cache-construction, and evaluation code will be released with the supplementary material.

## Ethics Statement

This study concerns offline learning and control in simulated environments. Its experimental claims are restricted to those environments; successful simulated planning does not establish suitability for deployment on physical robots or in safety-critical settings.

## AI Use Statement

Generative AI tools assisted with literature organization, research framing, experiment design feedback, interpretation of existing results, mathematical exposition and proof drafting, English writing, reference formatting, figure preparation, and LaTeX preparation. Numerical results in this draft are transcribed from existing experiment records; proposed experiments are labeled separately. The authors are responsible for the final content and its verification.

## References

Akshit Achara, Tatiana Gaintseva, Mateo Mahaut, Pritish Chakraborty, Viktor Stenby Johansson, Melih Barsbey, Emanuele Rodolà, and Donato Crisostomi. Multi-way representation alignment. arXiv preprint arXiv:2602.06205, 2026. URL https://arxiv.org/abs/2602.06205.

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 CVPR, 2023. URL https://arxiv.org/abs/2301.08243.

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. Revisiting model stitching to compare neural representations. In NeurIPS, 2021. URL https://papers.neurips.cc/paper/2021/hash/01ded4259d101feb739b06c399e9cd9c-Abstract.html.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Russell Blackwell, Liheng Li, Corentin Martinet, Damien Vincent, Michael Rabbat, Mahmoud Assran, Yann LeCun, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. arXiv preprint arXiv:2404.08471, 2024. URL https://arxiv.org/abs/2404.08471.

Xiaoyi Dong, Jianmin Bao, Ting Zhang, Dongdong Chen, Weiming Zhang, Lu Yuan, Dong Chen, Fang Wen, Nenghai Yu, and Baining Guo. PeCo: Perceptual codebook for BERT pre-training of vision transformers. In AAAI, 2023. URL https://arxiv.org/abs/2111.12710.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In ICML, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In ICLR, 2021. URL https://arxiv.org/abs/2010.02193.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In ICLR, 2024. URL https://arxiv.org/abs/2310.16828.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Learning massively multitask world models for continuous control. arXiv preprint arXiv:2511.19584, 2025. URL https://arxiv.org/abs/2511.19584.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015. URL https://arxiv.org/abs/1503.02531.

Zhichao Huang, Chutong Meng, and Tom Ko. RepCodec: A speech representation codec for speech tokenization. In ACL, 2024. URL https://arxiv.org/abs/2309.00169.

Minyoung Huh, Brian Cheung, Tony Wang, and Phillip Isola. Position: The platonic representation hypothesis. In ICML, 2024. URL https://arxiv.org/abs/2405.07987.

Dmytro Kuzmenko and Nadiya Shvai. TD-MPC-Opt: Distilling model-based multi-task reinforcement learning agents. arXiv preprint arXiv:2507.01823, 2025. URL https://arxiv.org/abs/2507.01823.

Wenhow Li, Chengwei Ma, Hui Xiong, Ying-Cong Chen, and Lei Zhang. Platonic representation hypothesis on world models. arXiv preprint arXiv:2608.23720, 2026. URL https://arxiv.org/abs/2608.23720.

Sihui Luo, Xinchao Wang, Gongfan Fang, Yao Hu, Dapeng Tao, and Mingli Song. Knowledge amalgamation from heterogeneous networks by common feature learning. In IJCAI, pp. 3087–3093, 2019. URL https://www.ijcai.org/Proceedings/2019/428.

Lucas Maes, Quentin Le Lidec, Luiz Facury, Nassim Massaudi, Ayush Chaurasia, Francesco Capuano, Richard Gao, Taj Gillin, Dan Haramati, Damien Scieur, Yann LeCun, and Randall Balestriero. stable-worldmodel: A platform for reproducible world modeling research and evaluation. arXiv preprint arXiv:2605.21800, 2026a. URL https://arxiv.org/abs/2605.21800.

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, 2026b. URL https://arxiv.org/abs/2603.19312.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In ICLR, 2023. URL https://arxiv.org/abs/2209.00588.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. arXiv preprint arXiv:1309.4168, 2013. URL https://arxiv.org/abs/1309.4168.

Benjamin Minixhofer, Fabian Paischer, and Navid Rekabsaz. WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In NAACL, 2022. doi: 10.18653/v1/2022.naacl-main.293. URL https://arxiv.org/abs/2112.06598.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. In ICLR, 2023. URL https://arxiv.org/abs/2209.15430.

Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. In ICLR, 2016. URL https://arxiv.org/abs/1511.06342.

Harry Robertshaw, Maxence Boels, Nikola Fischer, Sebastien Ourselin, Christos Bergeles, Alejandro Granados, and Thomas C. Booth. Progressive experience fusion for multi-task world model control in endovascular navigation. arXiv preprint arXiv:2608.18647, 2026. URL https://arxiv.org/abs/2608.18647.

Andrei A. Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. In ICLR, 2016. URL https://arxiv.org/abs/1511.06295.

Aidan Scannell, Mohammadreza Nakhaei, Kalle Kujanpää, Yi Zhao, Kevin Sebastian Luck, Arno Solin, and Joni Pajarinen. Discrete codebook world models for continuous control. In ICLR, 2025. URL https://arxiv.org/abs/2503.00653.

Chengchao Shen, Xinchao Wang, Jie Song, Li Sun, and Mingli Song. Amalgamating knowledge towards comprehensive classification. In AAAI, pp. 3068–3075, 2019. doi: 10.1609/aaai.v33i01.33013068. URL https://arxiv.org/abs/1811.02796.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In ICLR, 2017. URL https://arxiv.org/abs/1702.03859.

Trevor Standley, Amir R. Zamir, Dawn Sun, Weilong Ouyang, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In ICML, 2020. URL https://arxiv.org/abs/1905.07553.

Yee Whye Teh, Victor Bapst, Wojciech M. Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. In Advances in Neural Information Processing Systems, volume 30, 2017. URL https://arxiv.org/abs/1707.04175.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In Advances in Neural Information Processing Systems, volume 30, 2017. URL https://papers.nips.cc/paper/7210-neural-discrete-representation-learning.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. In ICLR, 2024a. URL https://arxiv.org/abs/2401.10491.

Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. FuseChat: Knowledge fusion of chat models. arXiv preprint arXiv:2408.07990, 2024b. URL https://arxiv.org/abs/2408.07990.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1006–1011, 2015. URL https://aclanthology.org/N15-1104/.

Jun Yamada, Marc Rigter, Jack Collins, and Ingmar Posner. TWIST: Teacher-student world model distillation for efficient sim-to-real transfer. arXiv preprint arXiv:2311.03622, 2023. URL https://arxiv.org/abs/2311.03622.

Boxuan Zhang, Weipu Zhang, Zhaohan Feng, Wei Xiao, Jian Sun, Jie Chen, and Gang Wang. Mixture-of-world models: Scaling multi-task reinforcement learning with modular latent dynamics. arXiv preprint arXiv:2602.01270, 2026a. URL https://arxiv.org/abs/2602.01270.

Huiyi Zhang, Zijian Li, Xiaocheng Feng, Weitao Ma, Xiaoliang Yang, Yichong Huang, and Bing Qin. ACTD: Anchor-based cross-tokenizer distillation with residual regularization. In EMNLP, 2026b. URL https://arxiv.org/abs/2608.29662.

Qizhen Zhang, Prajjwal Bhargava, Chloe Bi, Chris X. Cai, Jakob Foerster, Jeremy Fu, Punit Singh Koura, Ruan Silva, Sheng Shen, Emily Dinan, Suchin Gururangan, and Mike Lewis. BTS: Harmonizing specialized experts into a generalist LLM. arXiv preprint arXiv:2502.00075, 2025. URL https://arxiv.org/abs/2502.00075.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. arXiv preprint arXiv:2411.04983, 2024. URL https://arxiv.org/abs/2411.04983.

# A Similarity Alignment and Assignment Preservation

Fitting the transformation. Let $X, Y \in \mathbb{R}^{n\times d}$ contain paired source and reference row vectors. Write their means as $\bar{x}, \bar{y}$ and centered matrices as $X_c, Y_c$. For fixed scale and orthogonal matrix, the minimizing translation is $b = \bar{y} - s\bar{x}R$. The centered objective is

$$\|sX_cR - Y_c\|_F^2 = s^2\|X_c\|_F^2 + \|Y_c\|_F^2 - 2s\,\mathrm{tr}(R^\top X_c^\top Y_c). \tag{9}$$

With $X_c^\top Y_c = U\Sigma V^\top$, an orthogonal optimum is $R = UV^\top$, giving $s = \mathrm{tr}(\Sigma)/\|X_c\|_F^2$. We require nonzero source variance and a strictly positive fitted scale. The implementation permits reflections; it does not constrain $\det R = 1$. It computes the fit in float64 and stores the frozen parameters in float32. The inverse map is $T^{-1}(y) = ((y-b)/s)R^\top$.

Assignment preservation. Let $C = \{c_1, \ldots, c_K\}$, $T(z) = szR + b$, $s > 0$, and $R^\top R = I$. Since $R$ is square, $RR^\top = I$, and for every $k$,

$$\|T(z) - T(c_k)\|_2^2 = s^2(z - c_k)RR^\top(z - c_k)^\top = s^2\|z - c_k\|_2^2. \tag{10}$$

Multiplication by a common positive number preserves the ordering of all distances. Thus the set of minimizing indices is unchanged. If the minimizer is unique, its index is unchanged; if several entries tie, a consistent index-based tie rule gives the same result in exact arithmetic. Floating-point rounding can change near ties, motivating the numerical audit in Appendix C.2.

The argument applies to one transformed codebook. It does not compare $T_i(z)$ with $T_j(c)$ for $i \neq j$, nor does it imply equality of soft distributions at a fixed temperature. It guarantees neither a shared semantic label for two codewords nor preservation of a predictor's control performance.

# B Implementation and Evaluation Details

## B.1 Tasks, specialist sources, and action interface

Table 3: Task interfaces for the six-task collection. Block width is raw action dimension times frame skip. All tasks use frame skip 5 and latent dimension 192. "Released" denotes the original compatible specialist checkpoint; "scratch" denotes a specialist trained for this project.

| Task | Action | Block | Specialist | Offline dataset identifier |
|---|---|---|---|---|
| PushT | 2 | 10 | Released | galilai-group/lewm-pusht |
| Two-Room | 2 | 10 | Released | tworoom.h5 |
| Cube | 5 | 25 | Released | cube_single_expert.h5 |
| Scene | 5 | 25 | Scratch | scene_play_v0.h5 |
| Reacher | 2 | 10 | Released | reacher.h5 |
| HumanoidMaze | 21 | 105 | Scratch | humanoidmaze_medium_ navigate_v0.h5 |

Table 3 lists the task interfaces. The shared action encoder pads action blocks to the maximum width in the task collection, which is 105 in the six-task setting, then maps them to 192 dimensions. The task embedding identifies the dynamics associated with each observation sequence. Specialist sources differ, so aggregate specialist scores summarize available references rather than a uniformly trained upper bound. Our training, alignment, cache-construction, and evaluation code is implemented on the stable-worldmodel platform (Maes et al., 2026a), and the "released" specialist checkpoints are provided by its authors.

## B.2 Architecture and training

Figure 6 shows the module-level backbone with tensor dimensions—the frozen specialist lane that produces cached supervision, and the shared student lane that is deployed with CEM planning. Table 4 summarizes the consolidation configuration.

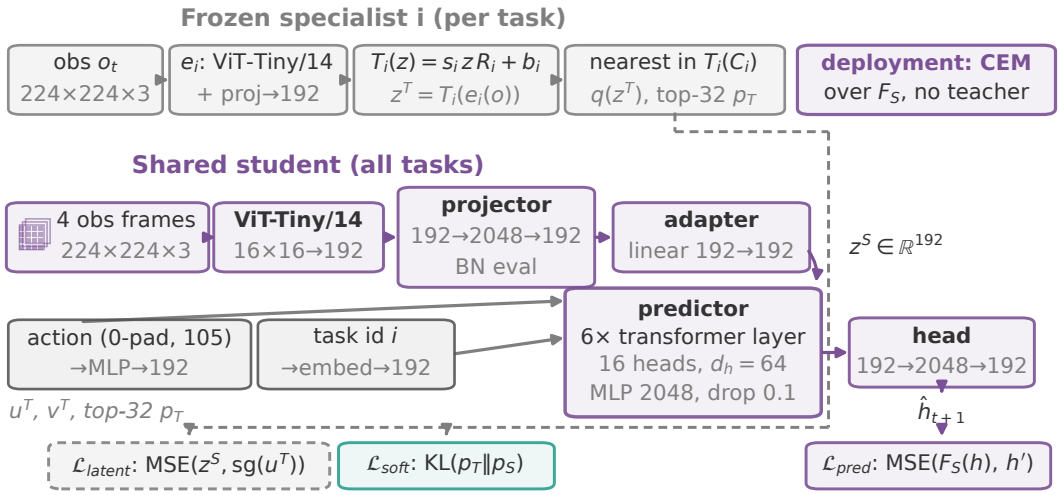


Figure 6: Network backbone. Top (gray): each frozen specialist encodes observations with its ViT-Tiny/14 encoder and projection; the fitted similarity map $T_i$ transports latents and codebooks into the reference frame, and nearest-code lookup supplies the quantized target $q(z^T)$ and top-32 soft distribution. Bottom (purple): the shared student encodes a four-frame observation stack (ViT-Tiny/14, projector 192→2048→192 with evaluation-mode normalization, linear adapter), conditions a six-layer, 16-head transformer predictor on padded action blocks and a task embedding, and predicts ahead through a 192→2048→192 head; at deployment only this lane remains, searched by CEM. Loss strip: prediction, representation-regression, and soft-KL terms (Section 3.3).

Table 4: Configuration of the six-task consolidation runs. Learning-rate arrows denote the initial and terminal rates within each phase.

| Component | Setting |
|---|---|
| Images and encoder | $224 \times 224$ RGB, ImageNet input normalization, ViT-Tiny, patch size 14, random student initialization |
| Representation | Dimension 192; projector and prediction head are $192 \to 2048 \to 192$ MLPs with batch normalization; linear $192 \to 192$ adapter; the projector's normalization layers stay in evaluation mode during consolidation, so their running statistics remain fixed and only affine parameters update |
| Predictor | 6 transformer layers, 16 heads, head dimension 64, MLP width 2048, dropout 0.1, embedding dropout 0 |
| Sequences | 4 observed frames; 3 context steps and 3 shifted prediction targets |
| Data split | Training fraction 0.9, split seed 3072 |
| Optimizer | AdamW, betas $(0.9, 0.999)$, weight decay 0.001, gradient clipping 1.0 |
| Batch and precision | 96 sequences/task/GPU, 4 GPUs, 2304 total sequences per update for 6 tasks; mixed bfloat16 |
| Phase lengths | 4 / 10 / 2 epochs; $\alpha = 0$ / linear $0 \to 1$ / 1 |
| Encoder LR | $10^{-3} \to 10^{-4}$ / $10^{-4} \to 10^{-5}$ / $10^{-5} \to 10^{-6}$ |
| Predictor LR | $10^{-4} \to 10^{-5}$ / $10^{-4} \to 5 \cdot 10^{-5}$ / $5 \cdot 10^{-5} \to 5 \cdot 10^{-6}$ |
| Warm-up fraction | 0.05 / 0.01 / 0 in the three phases, followed by cosine decay |
| Loss weights | Prediction 1, representation 1, soft KL 0.1 (0 for M4) |
| Codebooks | 8192 entries/task, dimension 192, top-32 teacher distribution, $\tau = 1$, distance chunks of 2048 |

Alignment samples up to 32,768 images from each task dataset. Each image is encoded by both the source specialist and the reference, including images outside either specialist's original task. The fitting fraction is 0.9 with split seed 4242; the remaining pairs supply

the reported alignment diagnostics. Alignment fitting uses a batch size of 512. The cache uses batches of 192 per GPU. Cached teacher latents and top-32 probabilities are stored in float16; hard and top-32 indices are stored as unsigned integers. The alignment holdout measures feature-fit generalization; it is distinct from a held-out control evaluation and is not a claim of trajectory-disjoint data splitting.

The original codebook construction ran a sequential unbalanced-optimal-transport fusion stage with candidate merging. Its recorded merge count is zero for the two-, three-, and six-task runs, yielding banks of 16,384, 24,576, and 49,152 entries respectively. The effective bank used here is therefore concatenation of transformed entries, with no measured vocabulary compression. Cache construction recomputes the hard index and top-32 probabilities on that bank. The shared training implementation can materialize code vectors for diagnostics even when the continuous objective does not use them.

### B.3 MPC protocol and reporting units

The configured CEM planner samples 300 candidate action sequences per iteration, retains 30 elites, and runs 30 iterations. It plans five action blocks and executes a receding horizon of five blocks, each block containing five raw actions. Start states, goal construction, task termination conditions, and episode budgets follow the corresponding task evaluation configuration. The reported multi-task tables use 50 fixed starts and planner seed 42. Additional PushT diagnostics use 200 held-out episodes, as labeled; their percentages must not be mixed with the 50-start comparison as if they were the same evaluation.

For success indicators $y_{ij} \in \{0, 1\}$ on task $i$, we report $S_i = 100 \sum_j y_{ij}/n_i$, $S_{\text{macro}} = N^{-1} \sum_i S_i$, and $S_{\min} = \min_i S_i$. Each task has equal weight regardless of trajectory count, action dimension, or number of images. A teacher gap is a difference in percentage points under the same evaluation protocol. We do not infer uncertainty across training runs from individual evaluation episodes.

## C Additional Completed Results

### C.1 Task-wise results for smaller collections

Table 5: Two-task success point estimates (%). Each shared model is trained on this two-task collection.

| Model | PushT | Two-Room | Macro | Min |
|---|---|---|---|---|
| Specialists | 90 | 86 | 88.0 | 86 |
| Single-task hybrid students | 78 | 84 | 81.0 | 78 |
| Native joint (M3) | 6 | 98 | 52.0 | 6 |
| Unaligned hybrid (M0) | 78 | 82 | 80.0 | 78 |
| Aligned hybrid (M2) | 90 | 84 | 87.0 | 84 |
| Aligned continuous (M4) | 94 | 86 | 90.0 | 86 |
| Aligned discrete (M5) | 88 | 82 | 85.0 | 82 |

The number of tasks changes the alignment pool, action interface, and training problem. These tables (two-task point estimates in Table 5, three-task in Table 6) are separate feasibility measurements; their differences do not measure the retention of an unchanged model or establish a scaling trend.

### C.2 Held-out alignment measurements

The maximum coordinate round-trip errors for these sources are $9.06 \times 10^{-6}$, $3.10 \times 10^{-6}$, $3.10 \times 10^{-6}$, $3.34 \times 10^{-6}$, and $2.86 \times 10^{-6}$, respectively. The preservation audit is numerically consistent with the task-local result, while the moderate $R^2$ values indicate that alignment leaves substantial feature variation unexplained. In the smaller collections, the corresponding fits improve held-out MSE by factors of 8.06 (Two-Room, three-task pool) and 3.36

Table 6: Three-task success point estimates (%). Each shared model is retrained on the three-task collection, rather than extended sequentially from its two-task counterpart.

| Model | PushT | Two-Room | Cube | Macro | Min |
|---|---|---|---|---|---|
| Specialists | 90 | 86 | 68 | 81.3 | 68 |
| Single-task hybrid students | 78 | 84 | 68 | 76.7 | 68 |
| Native joint (M3) | 4 | 92 | 60 | 52.0 | 4 |
| Unaligned hybrid (M0) | 74 | 84 | 74 | 77.3 | 74 |
| Aligned hybrid (M2) | 92 | 86 | 68 | 82.0 | 68 |
| Aligned continuous (M4) | 94 | 86 | 64 | 81.3 | 64 |
| Aligned discrete (M5) | 82 | 84 | 64 | 76.7 | 64 |

Table 7: Macro-average success (%) across independently trained task collections. The final row is the aligned-minus-unaligned hybrid difference in percentage points.

| Model | Two tasks | Three tasks | Six tasks |
|---|---|---|---|
| Specialists | 88.0 | 81.3 | 81.0 |
| Native joint (M3) | 52.0 | 52.0 | 50.0 |
| Unaligned hybrid (M0) | 80.0 | 77.3 | 75.3 |
| Aligned hybrid (M2) | 87.0 | 82.0 | 80.3 |
| Aligned continuous (M4) | 90.0 | 81.3 | 80.3 |
| Aligned discrete (M5) | 85.0 | 76.7 | 79.0 |
| M2 − M0 (pp) | +7.0 | +4.7 | +5.0 |

(Cube), with the same partial-agreement pattern; anchor pools differ across collections, so these factors are not directly comparable.

### C.3 Codebook reconstruction and control

The PushT reconstruction sweep uses a frozen specialist and trains each codebook for 100 epochs. It samples 262,144 features for codebook training and validation (235,929 training and 26,215 validation), and 32,768 independent features for testing. Initialization uses k-means++ on 65,536 features. A trainable code table minimizes squared reconstruction error at indices assigned by an exponential-moving-average table; the latter updates with momentum 0.99. AdamW and cosine learning-rate decay optimize the table, and the checkpoint minimizes validation quantization error. The final EMA table supplies the frozen codes. For $K = 512$ and 8192, mean absolute reconstruction errors are 6.8794 and 2.8373, and mean relative errors are 49.73% and 20.62%. At $K = 8192$, validation uses 6,980 entries, normalized perplexity is 65.91%, and the largest code probability is 0.0916%. Occupancy indicates how features use the dictionary; it does not establish semantic sharing or planning quality. Figure 7 shows the per-$K$ validation convergence and utilization of the same sweep.

Forcing quantization into the original PushT specialist at inference gives 58% success when applied to encoded features and 24% when also applied recurrently, compared with the original 90% on that 50-start evaluation. Training a single-task hybrid student gives 78% on the same comparison protocol. This is evidence that quantization placement and adaptation matter; it is not a comparison of fully discrete teachers supplying the shared model.

### C.4 Prediction-only diagnostic

A separate PushT run trains for 40 epochs with prediction loss and without both specialist supervision and SIGReg. At epoch 40, its training MSE is $9.369 \times 10^{-8}$ and validation MSE is $6.768 \times 10^{-7}$. Its checkpoint selected at epoch 10 achieves 3.5% success over 200 held-out episodes. These loss and control measurements describe different checkpoints in the same training run. Because both constraints are absent, the run is not a single-factor ablation of quantization, specialist supervision, or regularization. It illustrates why near-

Table 8: Six-task held-out source-to-PushT alignment measurements. Ratio is identity MSE divided by aligned MSE. Preserve measures task-local nearest-code index agreement, not pooled-bank agreement.

| Source | Identity MSE | Aligned MSE | Ratio | Cosine | $R^2$ | Preserve |
|---|---|---|---|---|---|---|
| Two-Room | 4.0351 | 0.2677 | 15.08 | 0.7488 | 0.2364 | 1.000000 |
| Cube | 1.3149 | 0.2900 | 4.53 | 0.7459 | 0.1728 | 1.000000 |
| Scene | 1.3176 | 0.2663 | 4.95 | 0.7447 | 0.2402 | 1.000000 |
| Reacher | 1.4326 | 0.2993 | 4.79 | 0.7327 | 0.1460 | 0.999949 |
| HumanoidMaze | 1.1152 | 0.2892 | 3.86 | 0.7199 | 0.1749 | 1.000000 |

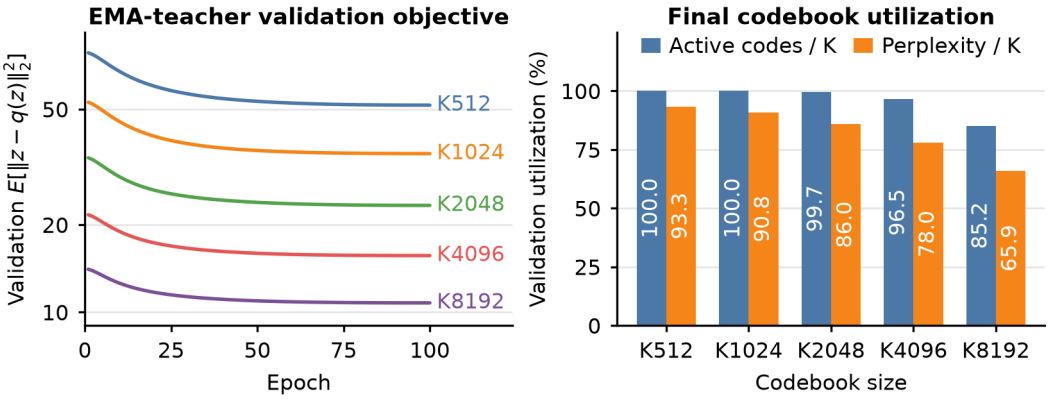


Figure 7: Diagnostics for the codebook-size sweep on PushT specialist features: per-$K$ validation convergence (left) and code-utilization measures versus $K$ (right). Convergence is comparable across $K$, while larger codebooks leave a growing fraction of entries unused.

zero latent loss is insufficient evidence of a useful control model; under the same held-out protocol, the $K$=8192 discrete-supervision student reaches 62% (Table 9). Figure 8 shows the convergence and the per-checkpoint control results.

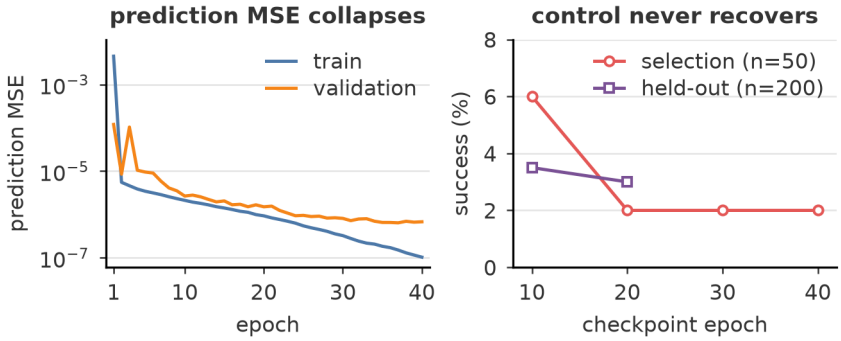


Figure 8: The prediction-only PushT diagnostic of Section 5.4: training and validation prediction MSE fall to $\sim 10^{-7}$ (left, log scale), while closed-loop success remains at a few percent at every evaluated checkpoint (right; selection on 50 starts, held-out on 200).

### C.5 Training dynamics and qualitative diagnostics

Figure 9 examines how the native joint baseline fails during training in the two- and three-task runs: the weighted SIGReg term dominates the objective's improvement while PushT prediction quality does not follow, and PushT validation error either jumps abruptly (two-task) or stays deceptively lowest (three-task) while closed-loop success collapses. Figure 10

Table 9: Completed single-task supervision sweep, success (%). PushT uses 200 held-out episodes and Cube uses 50. PushT checkpoints are selected on 50 starts before testing on 200 disjoint starts. Cube reports the best score on its 50-start evaluation. These are separate from the single-task hybrid reference checkpoints in the multi-task tables.

| $K$ | PushT discrete | PushT hybrid | Cube discrete |
|---|---|---|---|
| 512 | 17.5 | 75.5 | 54 |
| 2048 | 30.5 | 74.0 | 64 |
| 8192 | 62.0 | 76.0 | 70 |

shows the corresponding convergence of aligned hybrid consolidation, with the three schedule phases shaded. Figure 11 shows a qualitative latent-space view of the same contrast on the PushT rotation sweep.

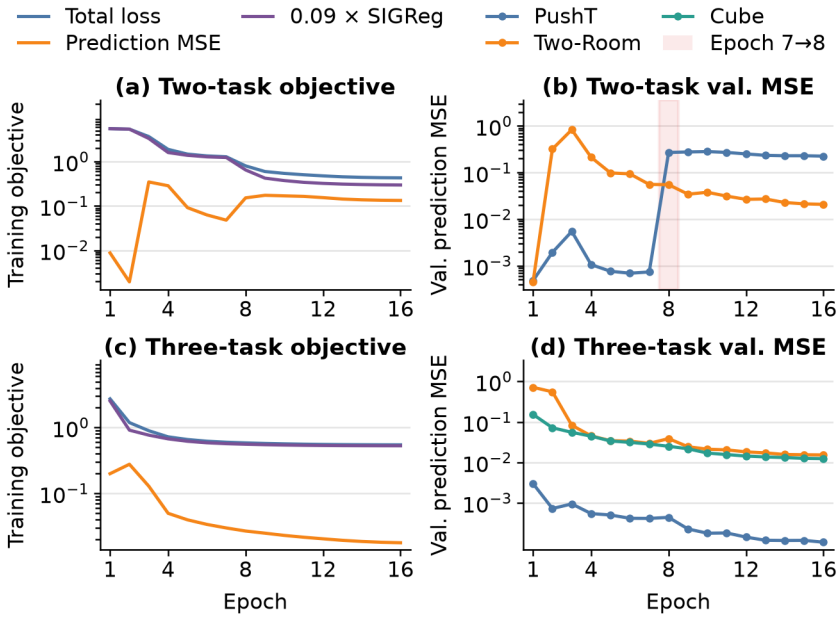


Figure 9: Training dynamics of the native joint baseline in the two-task (top row) and three-task (bottom row) runs. Left: objective terms (total loss, prediction MSE, weighted SIGReg; log scale). Right: per-task validation prediction MSE (log scale). In the two-task run, PushT validation MSE jumps by more than two orders of magnitude between epochs 7 and 8 (highlighted); in the three-task run, PushT validates lowest of the three tasks while its closed-loop success is 4% (Table 6). Neither failure is anticipated by the validation error alone.

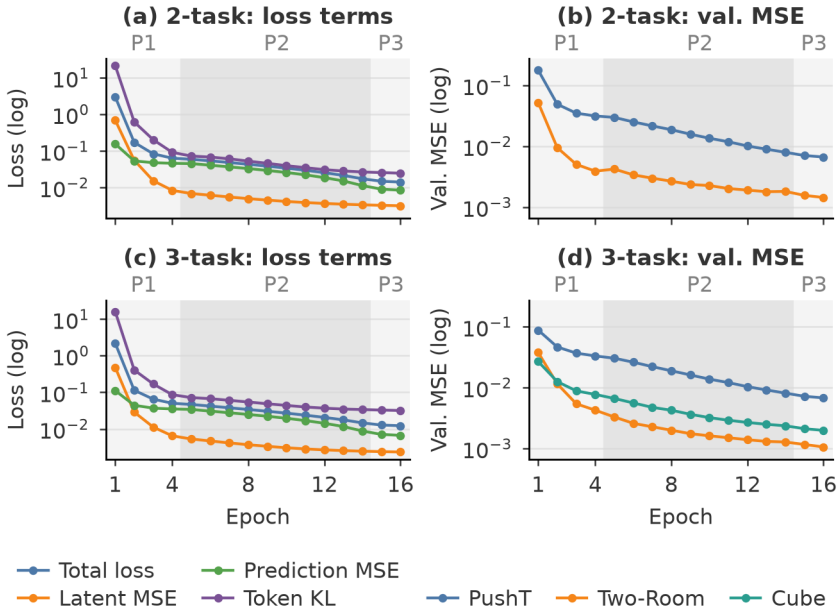


Figure 10: Training convergence of aligned hybrid consolidation (M2) in the two-task (top row) and three-task (bottom row) runs: loss terms (left) and per-task validation student prediction MSE (right), with the three schedule phases shaded (P1/P2/P3 = the 4/10/2-epoch teacher-warm-up, switching, and student-representation phases). Curves decrease smoothly with no divergence or abrupt jumps.

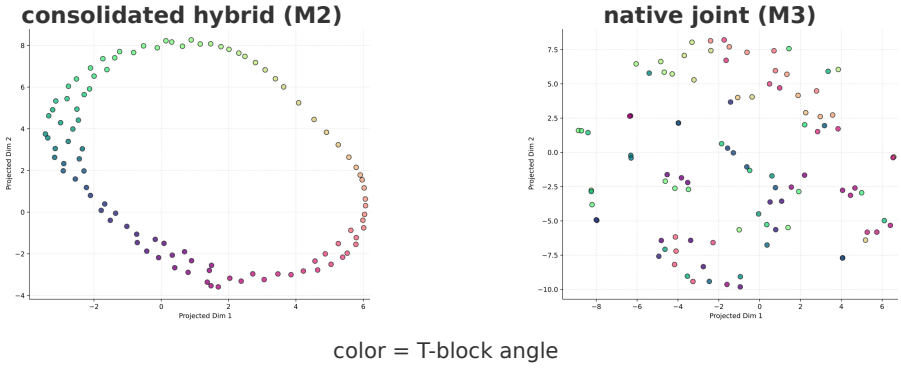


Figure 11: Qualitative latent topology on the PushT rotation sweep (two-task run): t-SNE projections of one hundred states varying only the T-block angle, whose physical sweep traces a unit circle; color encodes the angle. Left: the consolidated hybrid student (M2) forms a closed ring with monotonically progressing color, matching the underlying circle. Right: the native joint model (M3) scatters the same states into color-mixed clusters without ring structure. This projection is qualitative evidence, consistent with the control gap between the two models; it is not used as a quantitative measurement.

# D Planned Validation Protocols

Status. The experiments below are planned or in progress. This appendix reports no outcomes and supplies no evidence for the claims in the main paper.

## D.1 Independent training runs and matched native baseline

Question. Does the observed advantage over the native joint-training baseline persist across independently initialized training runs and comparable optimization budgets?

Design. We will prioritize the six-task collection and train M3, M0, M2, and M4 with at least three independent training seeds. M5 will use the same seeds if the compute budget permits. The historical run may be reused only if its artifact hashes, data split, initialization policy, and complete training protocol match the frozen experiment specification. We will match global per-task batch size, optimizer updates, per-task sample exposures, student architecture, numerical precision, and checkpoint selection. The native baseline will include the existing shared SIGReg setting and a per-task SIGReg alternative selected using validation data with the same tuning budget. Removing SIGReg will be treated as a diagnostic, not the sole native baseline.

All methods will use the same fixed evaluation-start manifest, with per-episode outcomes retained for paired comparisons. Shared frozen teachers, alignment fits, and caches may be reused across student seeds when their manifests match: this estimates variability of student training conditional on those artifacts. Robustness to retraining the teachers is a separate experiment. Neither cached supervision nor identical evaluation starts makes two student initializations the same training run.

Outputs and analysis. Every run will record the code revision, full configuration, initialization seed, split identifier, checkpoint rule, task exposures, update count, and GPU time. We will report each seed's task-wise success, macro success, worst-task success, and teacher gap. Training seeds will be the independent units for between-run uncertainty. Paired method differences will use the same seed and evaluation starts; any hierarchical bootstrap will preserve the seed and task structure rather than treating episodes as independent training runs. With few seeds, we will show the individual estimates and avoid interpreting a narrow conditional episode interval as strong evidence of training robustness.

Training cost will be reported both conditional on specialists already being available and including specialist training where that cost can be recovered. A baseline will be labeled budget-matched only after the recorded optimizer updates and task exposures have been checked; discrepancies will be disclosed rather than hidden by equal epoch counts.

## D.2 Separating alignment from target scale

Question. Does fitting source-to-reference correspondence add value beyond centering and matching overall feature scale?

Design. For the three-task collection, we will compare identity, center-and-norm matching, and full similarity alignment under continuous supervision. The center-and-norm control will use the same training feature pairs as the similarity fit, with

$$T_i^{\mathrm{norm}}(z) = \gamma_i(z - \mu_i) + \mu_r, \qquad \gamma_i = \frac{\sqrt{\mathbb{E}\|z_r - \mu_r\|^2}}{\sqrt{\mathbb{E}\|z_i - \mu_i\|^2}}. \tag{11}$$

This comparison removes the soft-token term and keeps the regression and dynamics-source choices continuous. Hybrid versions will separately evaluate the combined effects of code assignment and soft supervision. In both cases we will log target RMS norms and task-wise loss magnitudes; a center-and-norm control does not by itself guarantee identical scales to a least-squares similarity fit. A scale-normalized analysis will be needed before attributing any residual gain solely to orientation or correspondence.

We will rebuild caches for each transform and record task-local assignment agreement, pooled-bank assignment changes, and the fraction of selected codes originating from another task. We will also record top-32 entropy and the relation between transform scale and temperature. Temperature compensation within one task-local bank will not be presented as exact compensation for comparisons across differently transformed banks. A non-PushT reference, initially Two-Room, will test dependence on the chosen reference under the same data and training budget.

Decision rule. A repeatable advantage over the normalization control would support a more specific alignment claim. If the advantage disappears, the paper will describe target normalization as a sufficient explanation within the tested protocol. Either outcome can be inserted into the existing alignment subsection without changing the problem setting or method definition.

### D.3 Factorial analysis of supervision and source scheduling

The three reported instantiations are not a complete factorial ablation. To separate representation regression, dynamics source, and soft supervision, we will evaluate the configurations in Table 10. Let C denote the continuous feature and Q its final-bank quantization. Existing configurations will be rerun under the frozen multi-seed protocol where necessary; the table specifies designs rather than completed measurements.

Table 10: Planned supervision-factor configurations. No outcomes are reported. Consecutive configurations within each listed comparison change one factor.

| Regression | Dynamics source | KL weight | Purpose |
|---|---|---|---|
| C | C | 0 | Continuous reference (M4) |
| C | Q | 0 | Quantized-source effect relative to M4 |
| C | Q | 0.1 | KL effect at fixed source (M2) |
| Q | Q | 0 | Code regression without KL |
| Q | Q | 0.1 | Code regression with KL (M5) |
| C | C | 0.1 | Optional KL effect at continuous source |

We will first evaluate the schedule with a student-source-from-start control ($\alpha = 1$ throughout) and a teacher-source control ($\alpha = 0$ throughout), keeping representation supervision, updates, and all other losses fixed. The teacher-source control still trains the student encoder through representation supervision and evaluates with that student at deployment. We will compare these controls with the existing 4/10/2 schedule. Evaluations will report both prediction error and MPC success, since lower training loss alone does not determine a preferred schedule.

### D.4 Failure analysis

We will measure latent variance and effective rank, action sensitivity of predictions, and prediction error along evaluated trajectories for the native baseline's failing tasks and for Scene. Rank-based diagnostics will use common sample counts, centering, normalization, and feature dimension; otherwise scale differences can obscure their interpretation. Cross-task code selection will be reported as an assignment statistic, not semantic discovery. A contradictory outcome in any of the planned experiments will require revising the affected main-text claim, not moving the contradiction into an appendix.