SHIFTWM: WORLD MODELS THAT MOVE WHAT THEY SAW

Anonymous authors
Paper under double-blind review

ABSTRACT

World models let an agent evaluate actions before executing them, and forecasting in the patch-feature space of a frozen visual foundation model keeps them compact. Current latent world models, such as DINO-WM and V-JEPA 2-AC, regenerate the whole feature grid at every future step and feed their predictions back as inputs. Yet in manipulation and surgical scenes most content persists and only moves, so regeneration corrupts the static background and lets errors compound. We introduce SHIFTWM, a latent world model that forecasts by *moving what it has already seen*: for every future step it predicts an action-conditioned soft transport of nearby observed patch features, learned without flow supervision, a gate between moved and kept features and a correction for new content, and it decodes all horizons in parallel. As an output head, it also attaches to open-source world models. On held-out real-robot video from DROID, Language-Table and the bimanual IWS tasks, and on seven surgical tasks from Open-H, SHIFTWM has the lowest error of all matched output heads, 7.6% below a direct predictor on DROID; as a plug-in, it lowers the error of the 1.3B-parameter V-JEPA 2-AC by 15.7%.

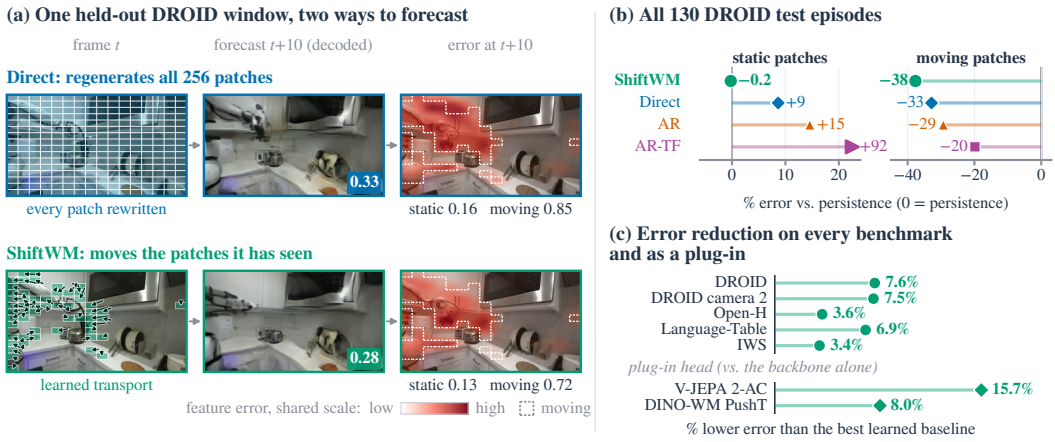


Figure 1: **SHIFTWM moves what it has seen instead of regenerating it.** (a) One held-out DROID window (Section D.5); Direct, a matched regression head, rewrites every patch, whereas SHIFTWM moves the arm's patches (arrows; dashed: the 25% that change most). (b) Error vs. persistence. (c) % lower error than the best learned baseline on every benchmark; plug-ins: vs. the same backbone without the head.

1 INTRODUCTION

Robots need world models, which predict how observations change in response to actions so that an agent can evaluate candidate actions before executing them (Hafner et al., 2019). Recent latent world models forecast in the patch-feature space of a frozen self-supervised encoder such as DINOv2 or V-JEPA 2, which keeps them compact and lets them plan without generating pixels (Zhou et al., 2024; Assran et al., 2025; Maes et al., 2026; Sobal et al., 2026). In robot manipulation and surgical

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.

robotics, such a model has to forecast several seconds ahead, respond to the commanded actions and generalise to scenes it has not seen.

Current latent world models regress the complete grid of patch features at every future step, and most feed their own predictions back as inputs. In manipulation and surgical video, however, most of the scene persists, as the table, tissue, tools and robot move but are rarely created or destroyed. Regenerating this persistent content causes three failures. (i) Static background must be rewritten exactly, yet on held-out DROID every regression predictor we train with the same encoder and architecture has higher error on static patches than copying the last frame, a baseline we call *persistence*. (ii) Moving content must be synthesised as a displacement in feature space, a target that grows with the motion and the feature difference between where content comes from and where it goes. (iii) Recursive rollouts compound both errors: on DROID, a DINO-WM-style predictor trained with teacher forcing, on true past features instead of its own predictions, has higher error than persistence once it is rolled out. Pixel and feature forecasters exploit motion by warping (Finn et al., 2016; Liu et al., 2017; Šarić et al., 2020), but latent world models for planning do not. On the moving patches of held-out DROID, the quarter with the largest true feature change, picking the observed feature in each patch's 7×7 neighbourhood that is closest to the true future (an oracle move) removes 51% of the error of persistence.

We introduce SHIFTWM, a latent world model that forecasts by *moving what it has already seen* (Figures 1 and 2). For every future step, each patch selects where its content comes from among nearby observed features, conditioned on the actions, while a gate keeps static content in place and a small correction adds what is new. SHIFTWM decodes every horizon in parallel from observed features, so its errors never feed back, and because it is an output head, its gated transport attaches to open-source world models.

We evaluate SHIFTWM against predictors that differ only in the output head, on held-out real-robot and surgical video, attach it to V-JEPA 2-AC and DINO-WM, and test whether the future is in fact a transport of observed content.

In this paper, our major contributions are:

1. **Forecasting by moving, not regenerating.** We introduce SHIFTWM, a latent world model that forecasts each future patch by moving features it has already observed, guided by the actions, instead of regenerating them. Unlike existing latent world models, which synthesise the features of moving content and mostly roll out recursively, it reuses what it has seen and predicts all steps in parallel.
2. **A plug-in head that upgrades existing world models.** Because SHIFTWM changes only how the forecast is formed, it attaches to open-source world models without changing their backbone or training recipe. On V-JEPA 2-AC it lowers forecast error by 15.7%, raising the error it removes relative to copying the last frame from 24.6% to 36.4%, and on DINO-WM it lowers error on PushT by 8.0% (Section 5.2).
3. **Lowest error across robots and surgery.** In a matched comparison, SHIFTWM has the lowest error on every benchmark: DROID (including an unseen camera), Language-Table, three bimanual IWS tasks and seven surgical tasks, and it wins on 98% of DROID test episodes (Section 5.1).
4. **The future is mostly the present, moved.** Choosing the best observed feature in each patch's window (an oracle move) removes 51% of the persistence error on moving content; SHIFTWM recovers 79% of this gain, and with a 1×1 window, which cannot move content, its advantage disappears (Sections 5.3 and 5.4).

2 RELATED WORK

Latent world models. Predicting dynamics in the embedding space of a frozen encoder keeps world models small enough to plan with: DINO-WM predicts DINOv2 patch features for model-predictive control (Zhou et al., 2024; Oquab et al., 2023), V-JEPA 2-AC post-trains an action-conditioned predictor on DROID (Assran et al., 2025; Khazatsky et al., 2024), and joint-embedding and foundation-feature forecasters extend this setting (Sobal et al., 2026; Maes et al., 2026; Balestriero & LeCun, 2025; Terver et al., 2025; Baldassarre et al., 2025; Karypidis et al., 2026; Boduljak et al., 2025; Porcher et al., 2026). DDP-WM separates dynamic regions from a static background, RLA-WM models residual latent actions (Yin et al., 2026; Zhang et al., 2026a), and recent analyses study

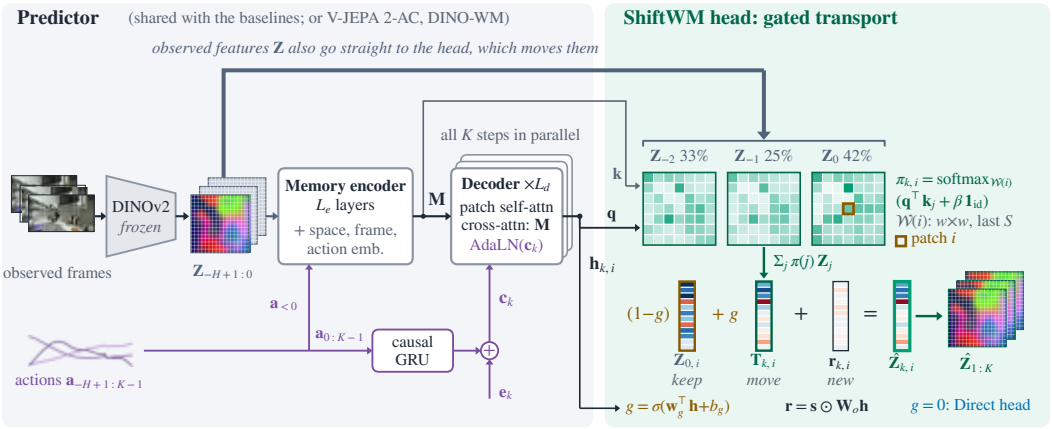


Figure 2: **SHIFTWM architecture.** Left: a predictor, shared with every baseline or taken from an open-source world model, turns the observed DINOv2 grids and the actions into a query $\mathbf{h}_{k,i}$ for every patch i and step k, all K steps in parallel. Right: the SHIFTWM head attends over a $w \times w$ window in the last S observed grids (Equation (1); weights π from a held-out DROID window, k=10), moves the observed features, $\mathbf{T} = \sum_j \pi(j)\mathbf{Z}_j$, and outputs $(1-g)\mathbf{Z}_0 + g\mathbf{T} + \mathbf{r}$ (Equation (2)); with g=0 it is the Direct baseline. Vectors: 12 principal components of one patch.

which latent spaces make predictions action-relevant (Jha et al., 2026; Yeom et al., 2026; Zhang et al., 2026b; Huang et al., 2026); pixel-space and model-based reinforcement-learning world models are complementary (Bar et al., 2025; Agarwal et al., 2025; Hafner et al., 2025; Hansen et al., 2024). All these predictors synthesise the features of moving content, including DDP-WM, which keeps static regions fixed but still regresses dynamic ones, whereas SHIFTWM forms each forecast as a corrected transport of observed features and attaches to them as a head.

Multi-step prediction. Iterating a one-step model compounds its errors (Venkatraman et al., 2015), yet most latent world models are rolled out recursively after teacher-forced training, sometimes with a rollout loss (Zhou et al., 2024; Assran et al., 2025); latent overshooting (Hafner et al., 2019), anchor frames (Luan et al., 2026) and decision-oriented objectives (Wang et al., 2026a) reduce the resulting drift. Direct prediction of every horizon avoids feedback but still re-synthesises static and moving content, which SHIFTWM keeps or moves instead.

Motion and feature warping. Moving existing content has a long history in pixel prediction, with action-conditioned kernels (Finn et al., 2016), dynamic filters (Jia et al., 2016), adversarial transformations (Vondrick & Torralba, 2017), skip connections (Ebert et al., 2017), resampling along predicted displacements (Reda et al., 2018; Liu et al., 2017) and stochastic or diffusion variants (Lee et al., 2018; Voleti et al., 2022). In feature space, features are propagated along optical flow (Zhu et al., 2017), forecast for segmentation (Luc et al., 2017; 2018), warped along a forecast motion field and blended with a regression (F2M; Šarić et al., 2020), or copied forward by attention (Oh et al., 2019), and recent robot world models add 3D scene flow, optical-flow velocity or object slots to condition a generator (Guo et al., 2026; Syed et al., 2026; Wu et al., 2022; Mosbach et al., 2024). SHIFTWM carries move-and-correct into latent world models: unlike F2M, it learns an action-conditioned soft transport over a local window of several observed frames from the forecasting loss alone, gates it against an exact identity path, decodes every horizon in parallel and attaches as a plug-in head, and it has lower error than F2M-style warping with optical flow (Section 5.3).

3 METHOD

SHIFTWM is a latent world model whose output is a *motion* of the feature grids it has observed. For every future step and every patch, it predicts where in a transport window of observed features the patch's content comes from, whether the patch stays or moves, and a correction for content that motion cannot explain, as Figure 2 shows with weights from a held-out DROID window. For

example, when the gripper moves one patch to the right, each patch along its path copies the gripper's features from its left neighbour in the last frame, the background keeps its own features, and only newly exposed regions need the correction. Sections 3.1 and 3.2 define the task and the predictor, Sections 3.3 and 3.4 the transport output and its loss, and Section 3.5 shows why this output suits scenes whose content mostly moves.

3.1 Problem setting

A frozen encoder E maps an image to a grid of $N = G \times G$ patch features $\mathbf{Z}_t \in \mathbb{R}^{N \times C}$. Given H observed grids $\mathbf{Z}_{-H+1:0}$, the actions between them $\mathbf{a}_{-H+1:-1}$, and a future action sequence $\mathbf{a}_{0:K-1}$, a world model predicts $\hat{\mathbf{Z}}_{1:K}$. The forecast $\hat{\mathbf{Z}}_k$ may depend only on $\mathbf{a}_{<k}$ (causality). Features are standardised per channel with train-split statistics shared across positions.

3.2 Memory encoder and action queries

Memory encoder. Each observed patch feature is projected to width d and receives a spatial embedding, a frame embedding and an embedding of the action applied after that frame. L_e transformer layers over all $H \cdot N$ tokens give memory states $\mathbf{M} \in \mathbb{R}^{H \times N \times d}$. **Action-prefix queries.** A unidirectional GRU over embedded actions gives a causal prefix state, to which a horizon embedding $\mathbf{e}_k$ is added to form $\mathbf{c}_k$. Query tokens start from the memory states of the last frame, and L_d decoder layers apply self-attention across patches and cross-attention to $\mathbf{M}$, each modulated by $\mathbf{c}_k$ through adaptive layer normalisation (AdaLN) with zero-initialised gains (Peebles & Xie, 2023). The result is a query $\mathbf{h}_{k,i} \in \mathbb{R}^d$ for patch i at step k, and all K steps are decoded in parallel.

Algorithm 1 SHIFTWM forecast

Input: grids $\mathbf{Z}_{-H+1:0}$, actions $\mathbf{a}_{-H+1:K-1}$
Output: forecasts $\hat{\mathbf{Z}}_{1:K}$

$\mathbf{M} \leftarrow \mathrm{Encoder}(\mathbf{Z}_{-H+1:0}, \mathbf{a}_{<0})$
for $k = 1, \dots, K$ (in parallel) **do**
  $\mathbf{c}_k \leftarrow \mathrm{GRU}(\mathbf{a}_{<k}) + \mathbf{e}_k$
  $\mathbf{h}_k \leftarrow \mathrm{Decoder}(\mathbf{M}, \mathbf{c}_k)$
  for each patch i **do**
    $\pi_{k,i} \leftarrow$ Eq. (1) over $\mathcal{W}(i)$
    $\mathbf{T}_{k,i} \leftarrow \sum_j \pi_{k,i}(j)\, \mathbf{Z}_j$
    $g_{k,i} \leftarrow \sigma(\mathbf{w}_g^\top \mathbf{h}_{k,i} + b_g)$
    $\mathbf{r}_{k,i} \leftarrow \mathbf{s} \odot \mathbf{W}_o \mathbf{h}_{k,i}$
    $\hat{\mathbf{Z}}_{k,i} \leftarrow (1 - g_{k,i})\mathbf{Z}_{0,i} + g_{k,i}\mathbf{T}_{k,i} + \mathbf{r}_{k,i}$
  end for
end for
return $\hat{\mathbf{Z}}_{1:K}$

3.3 Gated transport

A regression head would write every future feature from $\mathbf{h}_{k,i}$, whereas the transport reads it from the observed grids. For query patch i, let $\mathcal{W}(i)$ be the $w \times w$ spatial window around i in each of the last $S \le H$ observed frames. The transport is attention over this window, with a learned bias toward keeping each patch in place:

$$\pi_{k,i}(j) \;=\; \mathrm{softmax}_{j \in \mathcal{W}(i)} \Big(\tfrac{1}{\sqrt{d_{\text{key}}}} (\mathbf{W}_{\text{qry}} \mathbf{h}_{k,i})^\top (\mathbf{W}_{\text{key}} \mathbf{M}_j) + \beta\, \mathbf{1}[j = (0,i)] \Big), \tag{1}$$

where $\mathbf{W}_{\text{qry}}, \mathbf{W}_{\text{key}}$ project to key width d_{key}, j indexes a source patch and frame, and β is a learned identity bias toward the same location in the last frame. With $w{=}7$ and $S{=}3$, each patch scores 147 observed candidates, so the transport adds a cost linear in the number of patches, with a fifth of the candidates of a global window over the same frames. The transport moves *observed* features, not latent states, $\mathbf{T}_{k,i} = \sum_j \pi_{k,i}(j)\, \mathbf{Z}_j$. A per-patch gate $g_{k,i} = \sigma(\mathbf{w}_g^\top \mathbf{h}_{k,i} + b_g)$, with sigmoid σ, and a correction $\mathbf{r}_{k,i} = \mathbf{s} \odot \mathbf{W}_o \mathbf{h}_{k,i}$ give

$$\hat{\mathbf{Z}}_{k,i} \;=\; (1 - g_{k,i})\, \mathbf{Z}_{0,i} \;+\; g_{k,i}\, \mathbf{T}_{k,i} \;+\; \mathbf{r}_{k,i}. \tag{2}$$

The first term keeps the last observed feature in place, the second moves observed content, and the third adds what no observed feature contains; with $g{=}0$ the output reduces to the Direct baseline $\mathbf{Z}_0 + \mathbf{r}$, so the transport term is the only change (Algorithm 1 lists the steps). $\mathbf{W}_o$ is zero-initialised and $b_g = -3$, so an untrained model starts close to persistence, and $\mathbf{s} \in \mathbb{R}^C$ is a learned per-channel scale.

3.4 Training

We minimise the mean squared error over all horizons, $\mathcal{L}_{\text{pred}} = \frac{1}{K} \sum_k \|\hat{\mathbf{Z}}_k - \mathbf{Z}_k\|^2$. Anchoring every forecast to observed features makes it easy to ignore the actions, since a forecast that stays close to

$\mathbf{Z}_0$ scores well on average. An optional action-contrastive loss (Equation (10), Section C.2) enters only as an ablation, and all forecasting and plug-in results use the prediction loss alone, like every baseline.

3.5 Motion model

We motivate Equation (2) with a simple generative model, drawn for one patch in Figure 3. Suppose that each future patch copies one observed patch, plus zero-mean new content:

$$\mathbf{Z}_k \;=\; \mathbf{P}_k\,\mathbf{Z}_0 + \boldsymbol{\varepsilon}_k, \tag{3}$$

where $\mathbf{P}_k \in \{0,1\}^{N\times N}$ is a *motion* matrix whose row i selects the source patch $\rho(i)$ that patch i copies ($\mathbf{P}_k = \mathbf{I}$ is a static scene; we write the model for the last frame, and with S frames the sources range over all S grids) and $\boldsymbol{\varepsilon}_k$ is new content with $\mathbb{E}[\boldsymbol{\varepsilon}_k \mid \mathbf{P}_k, \mathbf{Z}_0, \mathbf{a}] = \mathbf{0}$. Motion is uncertain given the context $c = (\mathbf{Z}_{\le 0}, \mathbf{a}_{<k})$.

Proposition 1 (The optimal forecast is a transport)**.** *Under Equation (3), the MSE-optimal forecast is $\mathbb{E}[\mathbf{Z}_k \mid c] = \bar{\mathbf{P}}_k\mathbf{Z}_0$ with $\bar{\mathbf{P}}_k = \mathbb{E}[\mathbf{P}_k \mid c]$, a row-stochastic matrix. If the support of each row of $\mathbf{P}_k$ lies in the transport window $\mathcal{W}$, then $\bar{\mathbf{P}}_k\mathbf{Z}_0$ can be approximated to arbitrary precision by the transport of Equation (1) with $g{=}1$ and $\mathbf{r}{=}\mathbf{0}$, given logits that the decoder can express.*

A sufficiently expressive head could reach this optimum as well, but transport parameterises it exactly (①), and the correction $\mathbf{r}$ absorbs what the model leaves out, such as changes in viewpoint, lighting and shape. An additive head $\mathbf{Z}_0{+}\mathbf{W}_o\mathbf{h}$ instead has to write out the displacement (②). For a deterministic motion with sources $\rho(i)$, its target is $\mathbf{R} = (\mathbf{P}-\mathbf{I})\mathbf{Z}_0$ with $\|\mathbf{R}\|_F^2 = \sum_i \|\mathbf{z}_{0,\rho(i)} - \mathbf{z}_{0,i}\|^2$, the feature difference along the motion, whereas ShiftWM needs $\mathbf{r} = \mathbf{0}$ for any such motion within the window. This concerns what each head must represent, not what it can learn. On static patches the identity path is exact, whereas a regression head must output exactly zero. Finally, a recursive predictor that is L-Lipschitz with one-step error at most δ has a k-step error bounded by $\delta\sum_{j=0}^{k-1} L^j$, which grows geometrically for $L > 1$ (Venkatraman et al., 2015) (③). ShiftWM decodes each horizon from observed features, so its errors never feed back. Section B.2 gives the derivations.

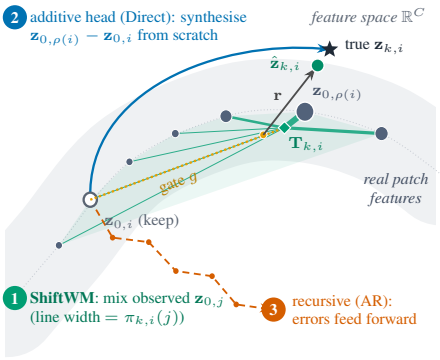


Figure 3: **How each head reaches one patch's future feature** (circle: last observed; star: true future). ① ShiftWM mixes nearby observed features (line width $= \pi$) into $\mathbf{T}$, gated, plus a small correction $\mathbf{r}$. ② A direct additive head synthesises the whole displacement. ③ An autoregressive (AR) head feeds its errors forward.

Testable predictions. If the future is a transport, choosing among the episode's own observed features should remove most of the error on moving content, and features from an unrelated episode should not (Section 5.3); the absolute gain should concentrate on moving patches while static patches are copied (Section 5.3); and the gain over the teacher-forced AR-TF should grow with the horizon (Section 5.1).

4 Experiments

We first compare ShiftWM with predictors that differ only in the output head on held-out robot and surgical video (Section 5.1), then attach its head to two open-source world models (Section 5.2), test whether its gains come from moving observed content (Section 5.3), and remove one component at a time (Section 5.4). Model selection uses validation splits only, and every test split is scored once.

Datasets. **DROID** (Khazatsky et al., 2024) contains episodes (recorded task demonstrations) of a real Franka robot arm, split by recording session. We train on the primary exterior camera and test zero-shot transfer to the second; one step is 0.33 s, its 35-D action block holds the five commanded gripper poses and openings of that step, and we forecast $K{=}10$ steps (3.3 s). **Open-H Hamlyn** (Open-H below) (Nelson et al., 2026) contains seven surgical training tasks, two on table-top phantoms (artificial practice models) and five on ex-vivo porcine tissue, recorded with the dVRK

Table 1: **Held-out forecasting.** Left: feature MSE ↓ per benchmark (mean over horizons unless noted; training seeds: DROID 3, Open-H 1, Language-Table 1, IWS 1). Right: on DROID, error on the 25% of patches that change most (moving) and on the rest (static), and action-ranking accuracy, the share of windows where the true actions give a lower error than another episode's. **Best**, second; †: paired 95% bootstrap CI vs. the best learned baseline excludes 0 (resampling recording sessions for DROID, source recordings for IWS, episodes otherwise); –: not run or not applicable; bottom row: vs. the best learned baseline.

	forecast error by benchmark						DROID analysis		
	DROID		DROID cam 2	Open-H	Language-	IWS	moving	static	action
Method	avg	k=10	(zero-shot)	surgical	Table	bimanual	↓	↓	rank. ↑
Persistence	0.2493	0.3264	0.2850	0.4437	0.3646	0.7408	0.7264	0.0903	–
Linear extrapolation	4.8117	11.8550	5.2971	10.8242	8.4367	–	–	–	–
AR-TF (DINO-WM-style)	0.2753	0.4168	0.3140	0.2854	0.1475	0.2635	0.5822	0.1730	84.9
AR (rollout-trained)	0.2059	0.2639	0.2440	0.2348	0.1220	0.2328	0.5135	0.1034	81.0
Direct (cross-attn)	0.1955	0.2496	0.2327	0.2338	0.1197	0.2331	0.4878	0.0981	89.8
SHIFTWM (ours)	**0.1807**†	**0.2312**†	**0.2153**†	**0.2255**†	**0.1115**†	**0.2249**†	**0.4527**†	**0.0901**†	**92.0**†
error reduction vs. best learned baseline	**+7.6%**	**+7.3%**	**+7.5%**	**+3.6%**	**+6.9%**	**+3.4%**	**+7.2%**	**+8.2%**	**+2.2 pt**

(da Vinci Research Kit) robot, 80-D action blocks for its two arms and episode-disjoint splits per task. **Language-Table** (Lynch et al., 2023) shows an xArm robot arm pushing rigid blocks (a fixed Open-X subset; O'Neill et al., 2024), and **IWS** (Wang et al., 2026b; Zhang et al., 2026a) records real bimanual arms pushing a T-block, handling a box and manipulating a rope, scored on the official validation handles (a start frame and its 60 commands, from recordings disjoint from all training data; Section C.1). For the plug-in study, we use the simulated PushT task (pushing a T-shaped block to a target pose) with the official **DINO-WM** code. Table 4 lists sizes, splits and licences, and Figure 9 maps all benchmarks, predictors and evaluations.

Baselines. All learned predictors share a frozen DINOv2-S/14 encoder (16×16 patches, no pooling), the same memory encoder and decoder (width 384, $L_e{=}L_d{=}4$, ≈22M parameters), optimiser, schedule, steps and batch size, and differ only in the output head and rollout. **Direct** replaces Equation (2) with $\mathbf{Z}_0 + \mathbf{W}_o\mathbf{h}_k$; it has the same cross-patch attention as SHIFTWM, so it controls for spatial communication. **AR** predicts a residual to the most recent grid and is rolled out recursively, with gradients through the rollout. **AR-TF** follows DINO-WM, as it is trained with teacher forcing to predict full next-step features and rolled out at test time. **Persistence** copies $\mathbf{Z}_0$, and **Linear** extrapolates the last feature difference, which multiplies frame-to-frame feature noise by up to K and is therefore far worse than persistence. As open-source world models, we use V-JEPA 2-AC and DINO-WM with and without the SHIFTWM head.

Metrics and statistics. The primary metric is standardised feature MSE on held-out test windows of $H{=}3$ observed and $K{=}10$ future steps ($K{=}12$ on IWS), averaged over horizons k (steps ahead) and at the final step. We also report *skill*, the fraction of persistence error a model removes, which is scale-free and therefore compares models in different feature spaces. Intervals are paired bootstraps over episodes (recording sessions for DROID, source recordings for IWS), and every dagger in Table 1 survives Holm correction; Section C gives all further details.

5 Results and Discussion

5.1 Held-out forecasting

Table 1 compares SHIFTWM with predictors that share its frozen encoder and architecture and differ only in how the forecast is formed. On held-out DROID, SHIFTWM has the lowest error of all methods, 7.6% below Direct, 12.2% below the rollout-trained AR and 34.3% below the DINO-WM-style AR-TF, which predicts full features rather than changes and is worse than persistence from $k{=}5$ on. It removes 27.5% of the persistence error, against 21.6% for Direct and 17.4% for AR. **The gains transfer.** Zero-shot on DROID's second exterior camera, which no model saw in training, SHIFTWM is 7.5% below Direct and 11.8% below AR. On Language-Table, where a robot pushes rigid blocks, it is 6.9% and 8.6% below them (24.4% below AR-TF), and on the seven Open-H surgical tasks 3.6% and 4.0%. On the three real bimanual IWS tasks it is 3.5% and 3.4% below them, with the lowest mean error on each task (Figure 6). Paired 95% intervals of the gain over the best baseline are [6.8,

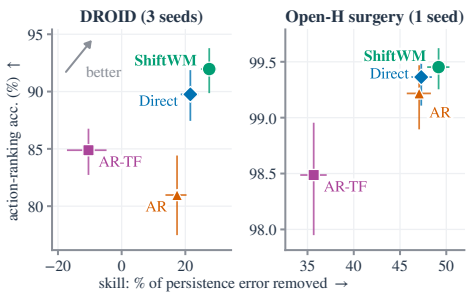


Figure 4: **Better forecasts without weaker action sensitivity.** Skill vs. action-ranking accuracy (95% CIs); SHIFTWM is highest on both axes on DROID and Open-H.

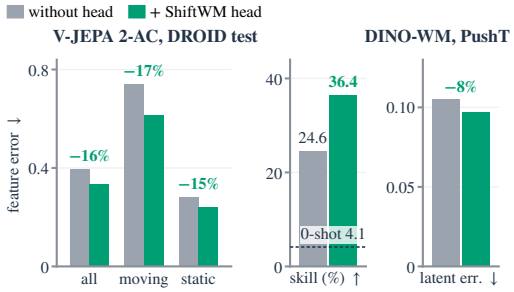


Figure 5: **SHIFTWM as a plug-in head**, same model, data and schedule with and without it. V-JEPA 2-AC: DROID test, 2 seeds (dashed: zero-shot); DINO-WM: PushT.

8.3]% on DROID (resampling 57 test sessions), [6.7, 8.3]% on its second camera, [6.4, 7.4]% on Language-Table, [3.3, 3.9]% on Open-H and [2.1, 4.5]% on IWS.

The gains hold across horizons and tasks (Figure 8f). The gain over AR-TF grows from 10% at k=1 to 45% at k=10 on DROID as its errors compound (Section 3.5), and the gain over Direct is positive at every horizon on every dataset. Lower error does not come from ignoring the actions: SHIFTWM also ranks actions best on DROID (92.0% against 89.8% for Direct) and on Open-H, so it is highest on both axes (Figure 4). Section D.1 gives the per-horizon, per-task and action-ranking results.

5.2 PLUG-IN HEADS FOR OPEN-SOURCE WORLD MODELS

Because SHIFTWM is an output head, we attach its gated transport, without the correction, to two open-source world models without other changes. The head blends the backbone's prediction $\mathbf{B}$ with transported observed features, $\hat{\mathbf{Z}} = (1-g)\mathbf{B} + g\,\mathbf{T}$, with the gate initialised near zero so that training starts from the unmodified model. Each pair shares data, schedule, loss and seed.

V-JEPA 2-AC. The released model (Assran et al., 2025) (trained on DROID) removes 4.1% of persistence error zero-shot on our test windows, 24.6% after fine-tuning, and 36.4% after fine-tuning *with* the head under the identical budget (Figure 5). The head thus lowers error by 15.7%, and by 16.9% on moving patches, for 0.16M added parameters. Its gate opens during fine-tuning, and validation error is lower at every evaluation for both seeds (Section D.2). Because V-JEPA 2-AC was pre-trained on DROID, its zero-shot row may be optimistic, and both fine-tuned rows share this starting point, so the head's gain is measured against the same model, data and budget.

DINO-WM. With the official code (Zhou et al., 2024), the head lowers teacher-forced latent error, the one-step feature error given true past features, by 8.0% on PushT and also lowers the LPIPS of its decoded frames (Table 3).

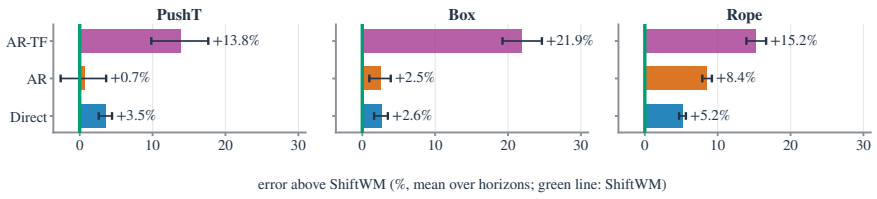


Figure 6: **IWS per task:** error of each baseline above SHIFTWM (200 official handles per task, mean over k, seed 0; paired 95% CIs over the 10 source recordings per task).

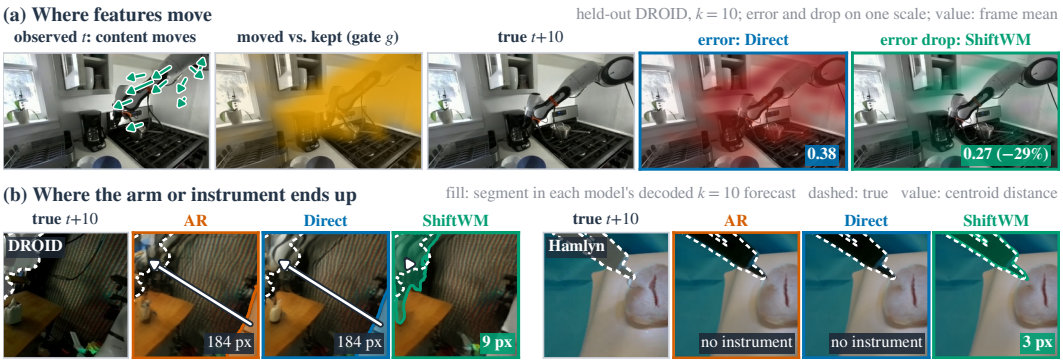


Figure 7: **What the transport does** (held-out k=10; selected windows, averages in Figure 8 and Section D.4). (a) On DROID the gate is high on the moving arm, arrows show where its features come from, and SHIFTWM lowers Direct's error on the arm (green: lower, violet: higher; one scale). (b) Each model's decoded forecast segmented with SAM 2.1 (dashed: true position; value: centroid distance).

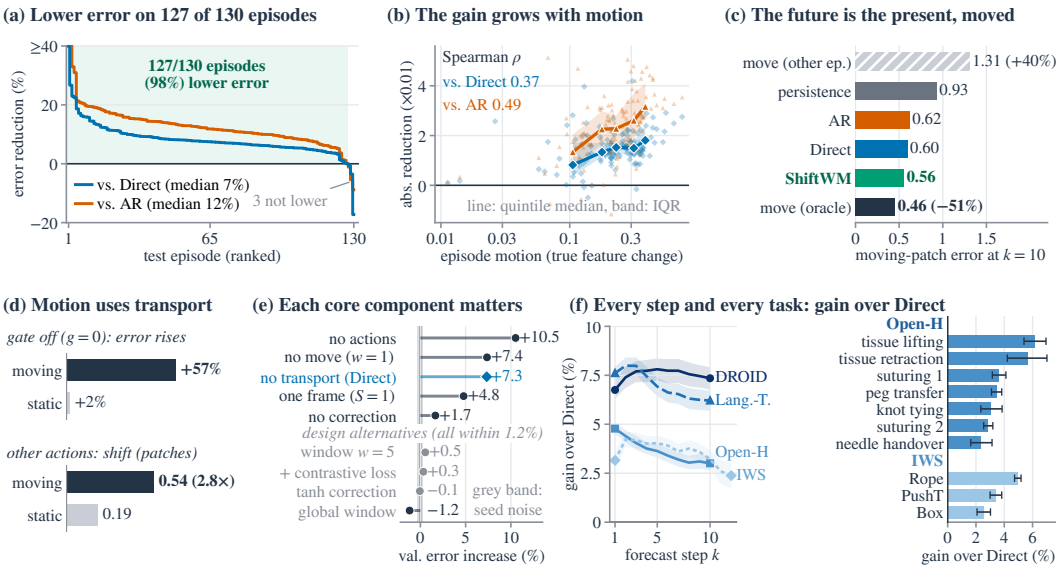


Figure 8: **Where and why SHIFTWM wins** (held-out DROID, seed 0, unless noted). (a) Per-episode error reduction vs. Direct and AR, ranked (mean over k). (b) Absolute reduction vs. episode motion (lines: quintile medians, bands: IQR). (c) Moving-patch error at k=10; oracle move: best observed feature in the transport window; other ep.: the same oracle with candidates from an unrelated episode. (d) Gate knock-out and action swap. (e) Ablations on validation (grey band: ±0.3%, above the 0.2% seed range). (f) Gain over Direct per step (left; DROID: 3 seeds) and per task (right; mean over k), 95% paired CIs.

5.3 WHY TRANSPORT HELPS

The gains concentrate on moving content. The regression target of Section 3.5 predicts gains on moving content and exact copies of static content. We split every test window into *moving* patches, the 25% with the largest true feature change (a property of the data shared by all methods), and *static* patches. On moving patches, SHIFTWM has 7.2% lower error than Direct and 11.8% lower than AR, and on static patches it is the only learned model at or below persistence, as regenerating the grid corrupts background that should not change (Table 1, right). Moving patches, a quarter of the grid, thus carry about 60% of the absolute error reduction over Direct, concentrated on the arm and gripper (Figure 7a). Per episode, SHIFTWM has lower error than each of Direct and AR on 98% of

the 130 DROID test episodes (three losses each, none in all three seeds), and its absolute gain grows with motion (Figure 8a,b). The moving arm also lands in the right place: labelling each forecast patch by its most similar observed patch (cosine) under the SAM 2.1 arm mask of the last observed frame, without any decoder, SHIFTWM has the highest arm IoU on moving DROID windows, +2.0 points above Direct and +4.2 above AR (paired intervals exclude zero; Section D.4), and on DROID decoded forecasts place the arm where it should be (Figure 7b).

The trained model moves content through the transport. Setting the gate to zero at test time, so that each forecast becomes $\mathbf{Z}_0 + \mathbf{r}$, raises the error by 56.8% on moving patches but only 1.8% on static ones, so content moves through the transport, not the correction. Swapping in another episode's future actions changes the transport field 2.8× more on moving than on static patches, so the actions steer *where* features are moved (Figure 8d).

The future is largely the present, moved. As a model-free check of Proposition 1, we pick, for each moving patch of 3,923 DROID test windows, the observed feature in its transport window closest to the true future, an *oracle move*. At $k{=}10$ it cuts the error of persistence by 51%, whereas the same oracle with candidates from an unrelated test episode is 40% *worse* than persistence (18% even from the same lab), so the future is largely the episode's own observed content, displaced (Figure 8c). SHIFTWM recovers 79% of this gain (Direct 71%, AR 65%); as Direct differs only in the transport, the gap between the shares is what transport adds, and SHIFTWM has lower error than Direct and AR in all 100 cells of 10 horizons by 10 motion deciles (Section D.3). The learned transport is also more than flow warping: its error is 25.4% below warping $\mathbf{Z}_0$ with RAFT flow (as in F2M's warping branch) and 18.1% below warping with the true future flow, and pure transport without the correction is already 17.9% below the true-flow warp.

5.4 ABLATIONS

Figure 8e changes one component at a time on DROID, scored on validation; the full model's seed range is 0.2%. **The gain comes from moving content.** Replacing the transport head by a direct regression head raises the error by 7.3%, a $1{\times}1$ window that keeps the gate and correction but cannot move content by +7.4%, and a single source frame ($S{=}1$) by 4.8%. Removing the correction costs +1.7%, and removing the actions costs most (+10.5%), so the forecasts follow the commanded motion. Design alternatives ($5{\times}5$ or global window, $\tanh$ correction, contrastive loss) stay within 1.2%; the global window lowers error slightly (-1.2%) but ranks actions less well (92.7 vs. 93.4%) and scores 768 instead of 147 candidates.

6 CONCLUSION

SHIFTWM forecasts by moving what it has already seen and correcting only what is new. Against matched predictors, it has the lowest held-out error on DROID (including an unseen camera), Language-Table, three bimanual IWS tasks and seven surgical tasks, and it wins on 98% of DROID test episodes. As a plug-in head, it lowers the error of the 1.3B-parameter V-JEPA 2-AC by 15.7% with 0.16M added parameters. Move-and-correct is a simple and general output parameterisation, and we encourage latent world models to form their forecasts from what they have observed.

Limitations. Content that no observed frame contains, such as newly exposed background or an object entering the view, cannot be moved and must come from the correction: even the oracle move removes only 51% of the persistence error on moving content (Figure 8c), and for the rest SHIFTWM falls back to regression, like its baselines. Its forecasts are also deterministic: under uncertain motion the MSE-optimal forecast averages the plausible sources (Proposition 1), which stochastic feature forecasters (Boduljak et al., 2025; Porcher et al., 2026) could complement. Two properties of the data bound the evaluation (Section C.1): Language-Table segments may share a play session across splits, and the Hamlyn videos are external-camera views of practice tasks, not clinical endoscopy.

REPRODUCIBILITY STATEMENT

Code, configurations, per-episode evaluation records and trained checkpoints will be released; the model is also packaged as an installable library (Section B.4). Section B.1 lists every hyperparameter.

Dataset splits are deterministic and documented in Section C.1. All datasets are public, namely DROID (CC-BY-4.0), Open-H-Embodiment (CC-BY-4.0), Language-Table via Open-X (CC-BY-4.0), the IWS recordings of the RLA-WM release (no licence stated; used for research as released) and the DINO-WM releases.

Ethics statement

The surgical data are public recordings of robotic procedures on table-top phantoms and ex-vivo porcine tissue, released under CC-BY-4.0. We make no clinical claims, and the models are not intended for clinical use.

AI use statement

We used large language model (LLM) tools to explore related research and to polish the writing of this paper. The authors take full responsibility for the content of this work.

References

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Federico Baldassarre, Marc Szafraniec, Basile Terver, Vasil Khalidov, Francisco Massa, Yann LeCun, Patrick Labatut, Maximilian Seitzer, and Piotr Bojanowski. Back to the features: Dino as a foundation for video world models. *arXiv preprint arXiv:2507.19468*, 2025.

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

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15791–15801. IEEE, 2025.

Gabrijel Boduljak, Yushi Lan, Christian Rupprecht, and Andrea Vedaldi. Vfmf: World modeling by forecasting vision foundation model features. *arXiv preprint arXiv:2512.11225*, 2025.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. volume 12, pp. 23, 2017.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. volume 29, 2016.

Jun Guo, Xiaojian Ma, Yikai Wang, Min Yang, Huaping Liu, and Qing Li. Flowdreamer: A rgb-d world model with flow-based motion representations for robot manipulation. *IEEE Robotics and Automation Letters*, 11(3):2466–2473, 2026.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025.

Nick Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024.

Yufan He, Pengfei Guo, Mengya Xu, Zhaoshuo Li, Andriy Myronenko, Dillan Imans, Bingjie Liu, Dongren Yang, Mingxue Gu, Yongnan Ji, et al. Cosmos-h-surgical: Learning surgical robot policies from videos via world modeling. *arXiv preprint arXiv:2512.23162*, 2025.

Siqiao Huang, Partha Kaushik, Michael Chen, Hengkai Pan, Kaiwen Geng, Omar Chehab, Fernando Moreno-Pino, and Max Simchowitz. Nano world models: A minimalist implementation of future video prediction. *arXiv preprint arXiv:2605.23993*, 2026.

Saurav Jha, Artem Zholus, Sarath Chandar, et al. Reconstruction or semantics? what makes a latent space useful for robotic world models. *arXiv preprint arXiv:2605.06388*, 2026.

Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. volume 29, 2016.

Efstathios Karypidis, Ioannis Kakogeorgiou, Spyridon Gidaris, and Nikos Komodakis. Dino-foresight: Looking into the future with dino. volume 38, pp. 163779–163811, 2026.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

Saurabh Koju, Saurav Bastola, Prashant Shrestha, Sanskar Amgain, Yash Raj Shrestha, Rudra PK Poudel, and Binod Bhattarai. Surgical vision world model. In *MICCAI Workshop on Data Engineering in Medical Imaging*, pp. 1–10. Springer, 2025.

Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

Chenxin Li, Hengyu Liu, Yifan Liu, Brandon Y Feng, Wuyang Li, Xinyu Liu, Zhen Chen, Jing Shao, and Yixuan Yuan. Endora: Video generation models as endoscopy simulators. In *International conference on medical image computing and computer-assisted intervention*, pp. 230–240. Springer, 2024.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pp. 38–55. Springer, 2024.

Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 4473–4481. IEEE, 2017.

Po-Chien Luan, Zimin Xia, Wuyang Li, Yang Gao, and Alexandre Alahi. Drift-resistant navigation world model with anchored epipolar guidance. *arXiv preprint arXiv:2605.24761*, 2026.

Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 648–657. IEEE, 2017.

Pauline Luc, Camille Couprie, Yann Lecun, and Jakob Verbeek. Predicting future instance segmentation by forecasting convolutional features. In *European Conference on Computer Vision*, pp. 593–608. Springer, 2018.

Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

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

Malte Mosbach, Jan Niklas Ewertz, Angel Villar-Corrales, and Sven Behnke. Sold: Slot object-centric latent dynamics models for relational manipulation learning from pixels. 2024.

Nigel Nelson, Juo-Tung Chen, Jesse Haworth, Xinhao Chen, Lukas Zbinden, Dianye Huang, Alaa Eldin Abdelaal, Alberto Arezzo, Ayberk Acar, Farshid Alambeigi, et al. Open-h-embodiment: A large-scale dataset for enabling foundation models in medical robotics. *arXiv preprint arXiv:2604.21017*, 2026.

Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9225–9234. IEEE, 2019.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Francois Porcher, Nicolas Carion, Karteek Alahari, and Shizhe Chen. Flow matching in feature space for stochastic world modeling. *arXiv preprint arXiv:2606.29059*, 2026.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, volume 2025, pp. 28085–28128, 2025.

Fitsum A Reda, Guilin Liu, Kevin J Shih, Robert Kirby, Jon Barker, David Tarjan, Andrew Tao, and Bryan Catanzaro. Sdc-net: Video prediction using spatially-displaced convolution. In *European conference on computer vision*, pp. 747–763. Springer, 2018.

Josip Šarić, Marin Oršić, Tonći Antunović, Sacha Vražić, and Siniša Šegvić. Warp to the future: Joint forecasting of features and feature motion. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10645–10654. IEEE, 2020.

Uladzislau Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim GJ Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. volume 38, pp. 43905–43941, 2026.

Shahram Najam Syed, Arthur Jakobsson, Haoran Hao, and Jeffrey Ichnowski. Intercepting the future: Latent-space predictive world model for dynamic vla manipulation. *arXiv preprint arXiv:2606.02486*, 2026.

Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pp. 402–419. Springer, 2020.

Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What drives success in physical planning with joint-embedding predictive world models? *arXiv preprint arXiv:2512.24497*, 2025.

Arun Venkatraman, Martial Hebert, and J Bagnell. Improving multi-step prediction of learned time series models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. volume 35, pp. 23371–23385, 2022.

Carl Vondrick and Antonio Torralba. Generating the future with adversarial transformers. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2992–3000. IEEE, 2017.

Jiawei Wang, Ke Rui, Yushen Zuo, Yichun Feng, and Minglei Li. Decision-metric alignment in latent world models: Diagnostics and action-conditioned objectives for mpc planning. *arXiv preprint arXiv:2608.18746*, 2026a.

Yixuan Wang, Rhythm Syed, Fangyu Wu, Mengchao Zhang, Aykut Onol, Jose Barreiros, Hooshang Nayyeri, Tony Dear, Huan Zhang, and Yunzhu Li. Interactive world simulator for robot policy training and evaluation. *arXiv preprint arXiv:2603.08546*, 2026b.

Ziyi Wu, Nikita Dvornik, Klaus Greff, Thomas Kipf, and Animesh Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models. 2022.

Jewon Yeom, Hanseul Kim, Jeongjae Park, Sungmok Jung, Jaejin Lee, and Taesup Kim. What makes video world model latents action-relevant: Prediction over reconstruction. *arXiv preprint arXiv:2606.07687*, 2026.

Shicheng Yin, Kaixuan Yin, Weixing Chen, Yang Liu, Guanbin Li, and Liang Lin. Ddp-wm: Disentangled dynamics prediction for efficient world models. *arXiv preprint arXiv:2602.01780*, 2026.

Xinyu Zhang, Zhengtong Xu, Yutian Tao, Yeping Wang, Yu She, and Abdeslam Boularias. Learning visual feature-based world models via residual latent action. *arXiv preprint arXiv:2605.07079*, 2026a.

Zhenghao Zhang, Yuanxiang Wang, Zhenyu Guan, Yujia Yang, Bingkang Shi, Tianyu Zong, Hongzhu Yi, Guoqing Chao, Xingchen Chen, Tiankun Yang, et al. Delta-jepa: Learning action-sensitive world models via latent difference decoding. *arXiv preprint arXiv:2606.31232*, 2026b.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. 2024.

Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 4141–4150. IEEE, 2017.

A Evaluation overview and further related work

Figure 9 maps every benchmark to the evaluation axes it serves and the predictors evaluated on it.

A.1 Further related work

Pixel-space generative world models learn action-controllable video simulators, from navigation (NWM; Bar et al., 2025) to foundation models for physical AI (Cosmos; Agarwal et al., 2025) and interactive environments (Genie; Bruce et al., 2024), and model-based reinforcement learning learns task-specific latent dynamics jointly with rewards and policies (Hafner et al., 2019; 2025; Hansen et al., 2024). These systems optimise sample fidelity or online return with their own encoders and data, so they do not share our setting of a frozen encoder with targets in common feature coordinates, and we treat them as complementary. Surgical world modelling is likewise driven by generative video models, such as Endora for endoscopy video (Li et al., 2024), SurgWM with latent-action control (Koju et al., 2025) and Cosmos-H-Surgical for policy learning (He et al., 2025), whereas we use Open-H dVRK recordings (Nelson et al., 2026) to test kinematics-conditioned forecasting in feature space.

B Extended method

B.1 Model details and hyperparameters

The memory encoder attends over all $H \cdot N{=}768$ tokens of the three observed grids, each query-decoder layer applies self-attention, cross-attention to memory and an MLP under AdaLN-zero, and actions are embedded by a 2-layer MLP and a single-layer GRU of width d. The transport uses

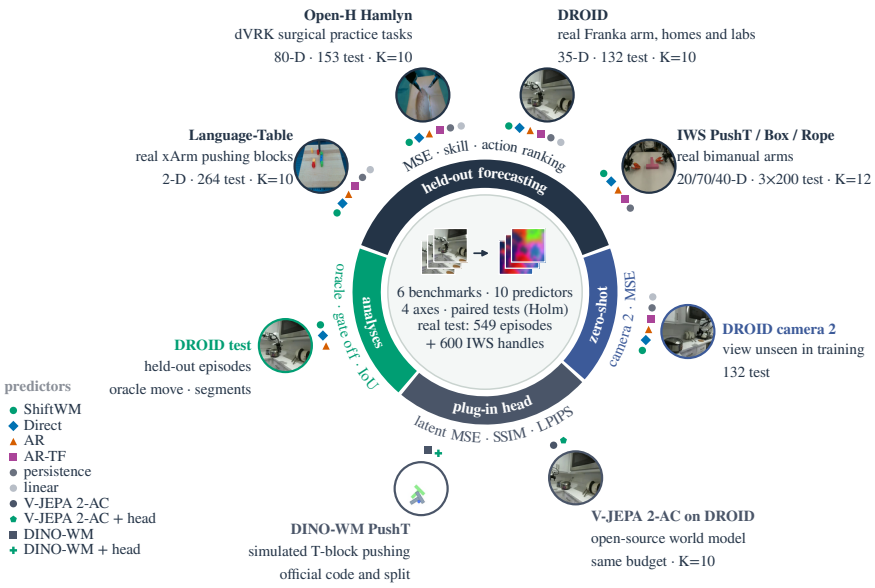


Figure 9: **Evaluation overview.** Evaluation axes (inner ring, with metrics), benchmarks with their data (real frames) and the predictors evaluated on each (markers).

key dimension 64 and a 7×7 window over $S{=}3$ source frames (147 candidates per query), and the correction head applies LayerNorm, a zero-initialised linear map to C and a per-channel scale initialised at 1. Table 2 lists all sizes and settings.

Table 2: **Hyperparameters**, shared by all learned predictors and datasets unless noted. IWS uses $K{=}12$.

Width / heads / MLP ratio	384 / 6 / 4	Optimiser	AdamW, $\beta{=}(0.9, 0.95)$
Memory encoder layers L_e	4	Peak LR / warm-up	$3{\times}10^{-4}$ / 400 steps
Query decoder layers L_d	4	Schedule	cosine, 8k steps
History H / horizon K	3 / 10	Batch size	48 windows
Transport window w / sources S	7 / 3	Weight decay / clip	0.05 / 1.0
Key dim / identity bias init	64 / 4.0	Precision	bf16 autocast
Gate bias init	-3	Selection	best val. MSE (every 1k steps)
Action-contrastive λ	0 (ablation: 0.5, margin 0.05)	Seeds	DROID 3, others 1

B.2 Proofs

Throughout, $c = (\mathbf{Z}_{\le 0}, \mathbf{a}_{<k})$ is the context, $\mathbf{e}_j$ the j-th standard basis vector, and $\mathbf{z}_{0,j}$ the j-th row of $\mathbf{Z}_0$.

Proof of Proposition 1. Optimal forecast. Under squared error the optimal forecast is the conditional mean. Since $\mathbf{Z}_0$ is determined by c and the new content has zero conditional mean, the tower property gives

$$\mathbb{E}[\mathbf{Z}_k \mid c] = \mathbb{E}[\mathbf{P}_k \mid c]\,\mathbf{Z}_0 + \mathbb{E}\big[\mathbb{E}[\boldsymbol{\varepsilon}_k \mid \mathbf{P}_k, c] \,\big|\, c\big] = \bar{\mathbf{P}}_k \mathbf{Z}_0. \tag{4}$$

Structure of $\bar{\mathbf{P}}_k$. Row i of $\mathbf{P}_k$ is $\mathbf{e}_{\rho(i)}^\top$, so row i of $\bar{\mathbf{P}}_k$ is $\sum_j \Pr[\rho(i) = j \mid c]\,\mathbf{e}_j^\top$, a probability vector whose support lies in $\mathcal{W}(i)$ whenever the support of $\rho(i)$ does.

Realisation by transport. With $g = 1$ and $\mathbf{r} = \mathbf{0}$, Equation (2) gives $\hat{\mathbf{z}}_{k,i} = \sum_{j \in \mathcal{W}(i)} \pi_{k,i}(j)\,\mathbf{z}_{0,j}$. Let $\mathcal{S}_i$ be the support of row i of $\bar{\mathbf{P}}_k$, $u = |\mathcal{W}(i) \setminus \mathcal{S}_i|$, and choose logits $\ell_j = \log \bar{P}_{k,ij}$ for $j \in \mathcal{S}_i$

and $\ell_j = -\Lambda$ otherwise. Since $\sum_{j\in\mathcal{S}_i} \bar{P}_{k,ij} = 1$,

$$\pi_{k,i}(j) = \frac{\bar{P}_{k,ij}}{1+ue^{-\Lambda}} \;\; (j \in \mathcal{S}_i), \qquad \pi_{k,i}(j) = \frac{e^{-\Lambda}}{1+ue^{-\Lambda}} \;\; (j \notin \mathcal{S}_i), \tag{5}$$

so the ℓ_1 distance to row i of $\bar{\mathbf{P}}_k$ is $2ue^{-\Lambda}/(1+ue^{-\Lambda}) \le 2ue^{-\Lambda}$ and

$$\left\|\hat{\mathbf{z}}_{k,i} - (\bar{\mathbf{P}}_k \mathbf{Z}_0)_i\right\| \;\le\; 2u\, e^{-\Lambda} \max_{j\in\mathcal{W}(i)} \|\mathbf{z}_{0,j}\| \;\xrightarrow[\Lambda\to\infty]{}\; 0. \tag{6}$$

With $S > 1$ source frames the same argument holds with $\mathcal{W}(i)$ ranging over patches of all observed frames. □

Regression target of Section 3.5. An additive head outputs $\mathbf{Z}_0 + \mathbf{R}$. Matching $\mathbf{P}\mathbf{Z}_0$ requires

$$\mathbf{R} = (\mathbf{P} - \mathbf{I})\mathbf{Z}_0, \qquad \mathbf{R}_i = \mathbf{z}_{0,\rho(i)} - \mathbf{z}_{0,i}, \qquad \|\mathbf{R}\|_F^2 = \sum_i \left\|\mathbf{z}_{0,\rho(i)} - \mathbf{z}_{0,i}\right\|^2. \tag{7}$$

For SHIFTWM, take the one-hot transport $\pi_{k,i} = \mathbf{e}_{\rho(i)}$ (the limit $\Lambda \to \infty$ above, admissible since $\rho(i) \in \mathcal{W}(i)$) and $g = 1$. Then $\hat{\mathbf{z}}_{k,i} = \mathbf{z}_{0,\rho(i)} = (\mathbf{P}\mathbf{Z}_0)_i$ for every i, so the required correction is $\mathbf{r} = \mathbf{0}$. □

Compounding bound of Section 3.5. Let the recursive rollout be $\hat{\mathbf{Z}}_j = f(\hat{\mathbf{Z}}_{j-1})$ with $\hat{\mathbf{Z}}_0 = \mathbf{Z}_0$, the truth $\mathbf{Z}_j = f(\mathbf{Z}_{j-1}) + \boldsymbol{\eta}_j$ with $\|\boldsymbol{\eta}_j\| \le \delta$, and $e_j = \|\hat{\mathbf{Z}}_j - \mathbf{Z}_j\|$. By the triangle inequality and the Lipschitz property,

$$e_j \;\le\; \left\|f(\hat{\mathbf{Z}}_{j-1}) - f(\mathbf{Z}_{j-1})\right\| + \|\boldsymbol{\eta}_j\| \;\le\; L\, e_{j-1} + \delta, \qquad e_0 = 0. \tag{8}$$

By induction on k (the case $k = 1$ is $e_1 \le \delta$; if $e_{k-1} \le \delta \sum_{j=0}^{k-2} L^j$ then $e_k \le L\delta \sum_{j=0}^{k-2} L^j + \delta$),

$$e_k \;\le\; \delta \sum_{j=0}^{k-1} L^j \;=\; \begin{cases} \delta\, \dfrac{L^k - 1}{L - 1}, & L \neq 1, \\ k\,\delta, & L = 1, \end{cases} \tag{9}$$

which grows geometrically for $L > 1$. A direct predictor forms $\hat{\mathbf{Z}}_k$ from the observed inputs $(\mathbf{Z}_{\le 0}, \mathbf{a})$ alone, so its error at horizon k is its per-horizon error δ_k and does not depend on its predictions at other horizons. □

B.3 PLUG-IN HEADS FOR OPEN-SOURCE WORLD MODELS

Table 3: **SHIFTWM as a plug-in head for open-source world models.** Same data, schedule and loss for each pair; only the head differs. Top: V-JEPA 2-AC on DROID test (its own feature space). Bottom: DINO-WM on PushT, official code; teacher-forced latent error on validation; SSIM, LPIPS: quality of decoded frames.

V-JEPA 2-AC (ViT-g, 1.3B) on DROID	MSE ↓	moving ↓	static ↓	skill (%) ↑
V-JEPA 2-AC, zero-shot	0.504	0.815	0.400	4.1
V-JEPA 2-AC, fine-tuned (2 seeds)	0.396	0.741	0.281	24.6
V-JEPA 2-AC, fine-tuned + SHIFTWM head (2 seeds)	**0.334**	**0.616**	**0.240**	**36.4**

Env	DINO-WM (ViT-S/14)	latent err. ↓	SSIM ↑	LPIPS ↓
PushT	DINO-WM	0.1053	0.976	0.014
PushT	DINO-WM + SHIFTWM head	**0.0969**	**0.977**	**0.013**

For an open-source backbone with prediction $\mathbf{B}_k$, the head outputs $\hat{\mathbf{Z}}_k = (1 - g_k)\mathbf{B}_k + g_k\mathbf{T}_k$, where $\mathbf{T}_k$ is the local 7×7 soft transport, with identity bias, of *observed* encoder tokens. The gate $g_k = \sigma(\mathbf{w}^\top \mathbf{h} + b)$ starts at $\mathbf{w} = \mathbf{0}$ and $b = -4$, so training starts numerically from the unmodified backbone, and nothing else in the backbone, loss or schedule changes. Table 3 gives the resulting errors.

V-JEPA 2-AC. Queries come from the predictor's normalised hidden state for each output block (1024-D), and keys and values from the layer-normalised ViT-g/16 tokens (16×16, 1408-D) of the last S=3 observed frames, with a learned per-source key offset (0.16M parameters). The fine-tuning

objective is the released one (teacher-forced L1 plus rollout L1), and during rollouts the predictor consumes $\hat{\mathbf{Z}}_k$ as its own prediction, while $\mathbf{T}_k$ only ever reads observed tokens.

DINO-WM. We subclass the official `ViTPredictor`, otherwise unmodified. Queries come from the predictor's output token, and keys and values from its input token at the same slot on the 14×14 DINOv2 patch grid ($S=1$); proprioceptive and action channels are left untouched. After the first rollout step, that input token is DINO-WM's own previous prediction, so the head then moves predicted rather than observed features, unlike SHIFTWM and the V-JEPA 2-AC head. The extra parameters are drawn from a private random generator, so data order, initialisation and dropout of the base predictor are identical in both arms.

B.4 SOFTWARE

The forecaster, the plug-in head and the evaluation helpers are packaged as a small Python library, `shiftwm` (MIT licence, depending only on PyTorch and NumPy), separate from the research code. It exposes `ShiftWM`, the forecaster of Section 3.2, `ShiftHead`, the backbone-agnostic plug-in head of Section B.3 with its gate closed at initialisation, `forecast`, which also returns the transport weights, gate and correction, `transport_field`, the expected source offset of every patch, and `skill`. Its tests confirm that the package reproduces the research implementation (maximum absolute difference 0 on the trained DROID checkpoint) and that `ShiftHead` starts at the backbone's prediction.

C EXPERIMENTAL DETAILS

C.1 DATASETS AND SPLITS

Table 4: **Datasets.** Every forecasting task predicts $K=10$ ($K=12$ on IWS) future feature grids from $H=3$ observed frames and the actions. Step: one model time step. Splits are disjoint at the split unit. DINO-WM PushT uses the official release and split.

Dataset	Recorded scenes	Episodes (train/val/test)	Step	Action	Split	Licence
DROID (24 shards)	real Franka arm in homes, labs and offices; two exterior cameras	851 / 143 / 132	0.33 s	35-D	session	CC-BY-4.0
Open-H Hamlyn	dVRK arms on 7 surgical practice tasks (knot tying, needle grasp and handover, peg transfer, two suturing tasks, tissue lifting, tissue retraction); scene camera	708 / 152 / 153	0.33 s	80-D	episode	CC-BY-4.0
Language-Table	xArm pushing coloured blocks; fixed camera (Open-X subset)	2,079 / 246 / 264	0.1 s	2-D	episode	CC-BY-4.0
IWS PushT / Box / Rope	real bimanual T-block, box and rope tasks (Wang et al., 2026b)	510–512 / 90 / 200	0.17 s	20/70/40-D	recording	as released

Table 4 summarises the datasets. **DROID.** We use 24 official full-release shards with 1,126 episodes from 14 sites, split by recording session with a salted hash (316/59/58 sessions in train/val/test). Frames are subsampled from 15 to 3 Hz, each action block concatenates the 5 commanded poses and gripper positions of one step, and the second exterior camera records the same instants from another viewpoint. Two of the 132 DROID test episodes are shorter than one $H+K=13$-step window, so per-episode analyses use 130. **Open-H Hamlyn.** The `observation.images.color` stream, which the release labels "endoscope", is an Intel RealSense scene camera (848×480). Each action block holds five commands, each an 8-D pose-and-gripper target per arm; 6 of 1,019 episodes are shorter than 14 steps and removed, and splits are 70/15/15 per task by a hash of the episode id. **Language-Table.** We use 14 of the 1,024 Open-X shards of a fixed release (6,161 episodes) at the native 10 Hz and keep the 2,589 episodes long enough for $H+K=13$ steps, split 81/9/10% by a salted hash of the episode id. The release has no id of the source play episode, so segments of one play session can fall into different splits; for only 1 of 264 test segments does a training segment start or end with a frame closer to the test boundary frame than the median one-step frame change. **IWS.** We use the IWS subset of the RLA-WM release (Wang et al., 2026b; Zhang et al., 2026a), real bimanual recordings of pushing a T-block, handling a box and manipulating a rope from one RGB camera at 30 fps (640×480, resized to 224×224). One model step groups five native command rows (1/6 s), so the official 60-row horizon is $K=12$ steps (2 s), and the action block concatenates the five recorded command vectors (4-D for PushT, 14-D for box, 8-D for rope). The test set is the 200 official validation handles per task, drawn from 10 reserved recordings, and each handle provides a single

observed frame, which we replicate into the H-frame history with neutral past actions. Training and checkpoint selection use the remaining upstream training recordings, split by recording (about 510 training and 90 validation recordings per task).

C.2 Baselines, training and cost

All learned baselines share the memory encoder, decoder, width, depth, optimiser and schedule of ShiftWM, and differ only in the output head and rollout. AR is trained through its full 10-step rollout, and AR-TF, following Zhou et al. (2024), regresses absolute next-step features with teacher forcing and is rolled out autoregressively at test time. We evaluate the released V-JEPA 2-AC (ViT-g encoder, 300M predictor) zero-shot on exactly our test windows with its own preprocessing, action convention and rollout, and fine-tune it on our training split with and without the head under an identical budget. ShiftWM has 21.75M parameters against 21.70M for Direct, AR and AR-TF, and because ShiftWM and Direct decode all K steps in parallel, a 10-step forecast takes one decoder pass instead of ten. One training run takes 0.55 h for ShiftWM, 0.47 h for Direct, 0.91 h for AR and 0.84 h for AR-TF (median, two runs sharing one H200).

Optional action-contrastive loss. A hinge with margin m and stop-gradient sg compares the error under the true future actions with the error under actions borrowed from another sample in the batch,

$$\mathcal{L}_{\text{act}} = \max\big(0,\ m - (\|\hat{\mathbf{Z}}(\tilde{\mathbf{a}}) - \mathbf{Z}\|^2 - \mathrm{sg}\,\|\hat{\mathbf{Z}}(\mathbf{a}) - \mathbf{Z}\|^2)\big), \qquad \mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda\,\mathcal{L}_{\text{act}}. \tag{10}$$

All forecasting and plug-in results use λ=0, like every baseline, and the loss enters only as an ablation (Section 5.4).

C.3 Statistical procedures

Each test episode contributes the mean of its windows (stride 2), and tables report the mean over training seeds. For a paired comparison of methods A and B, we compute per-episode differences (averaged over seeds) and draw 10,000 bootstrap resamples of episodes, of recording sessions for DROID and DROID camera 2, or of the source recordings (10 per task, stratified by task) for IWS, to form a percentile 95% interval. Table daggers use unadjusted intervals; for every dagger of Table 1 no resample crosses zero ($p \le 10^{-4}$), so all remain significant after Holm correction over the three learned baselines of each dataset. Training seeds per method are 3 for DROID, 2 for V-JEPA 2-AC and 1 for Open-H, Language-Table, IWS, DINO-WM and the ablations, and on DROID the full model's seed range is 0.2% of its validation error.

D Extended results

D.1 Forecasting per horizon, task and action ranking

The advantage of ShiftWM does not come from a single horizon or task: Direct and AR-TF lie above ShiftWM at every step on every benchmark and on each of the seven surgical tasks (Figure 10), AR-TF falls further behind with the horizon as its errors compound, and AR, the weakest model at short horizons, approaches ShiftWM at the longest ones. Figure 6 in the main text gives the three IWS tasks separately.

Action ranking. A forecast that stays close to $\mathbf{Z}_0$ can score well on average while barely responding to the actions, so we also measure the *action-ranking error*, the share of test windows in which another episode's actions yield a lower forecast error than the true actions. ShiftWM combines the highest skill with the lowest ranking error on DROID and Hamlyn, and on Language-Table ranking errors are rare for all methods (Figure 11a–c). Its gain over Direct is positive in every quintile of episode motion on all three datasets (Figure 11d).

D.2 Plug-in training dynamics

On V-JEPA 2-AC, validation error with the head is lower at every evaluation, its gate opens within the first few hundred steps, and the test gain holds at every horizon (Figure 12a–c). On DINO-WM PushT, the head lowers both the teacher-forced and the open-loop error (Figure 12d).

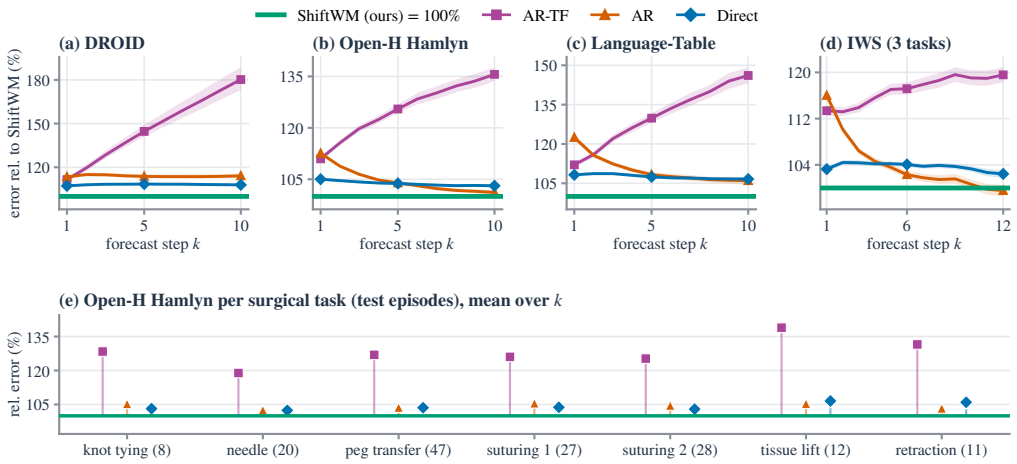


Figure 10: **Error of each baseline relative to SHIFTWM** (SHIFTWM = 100%; above the green line: SHIFTWM better) on held-out test data. (a–d) Per forecast step; bands: paired 95% bootstrap (sessions for DROID, episodes elsewhere); IWS weights its three tasks equally. (e) Per Open-H surgical task, mean over k (number of test episodes in brackets).

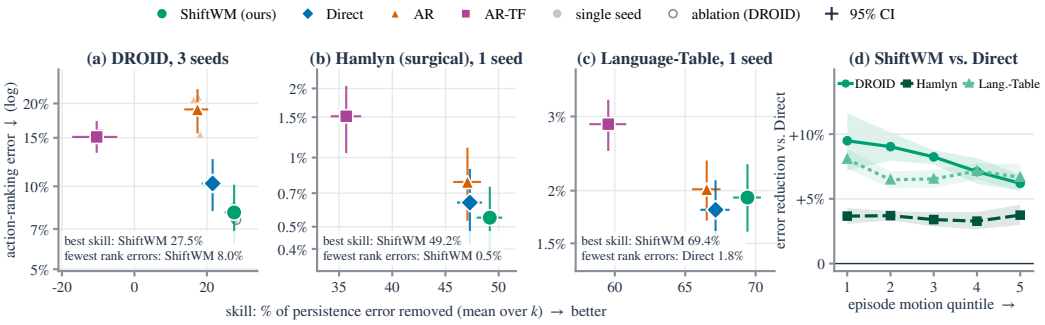


Figure 11: **Forecast skill versus action ranking (test).** (a–c) Lower right is better; large markers: seed means with 95% CIs; hollow: DROID ablations. (d) Error reduction of SHIFTWM over Direct per quintile of episode motion.

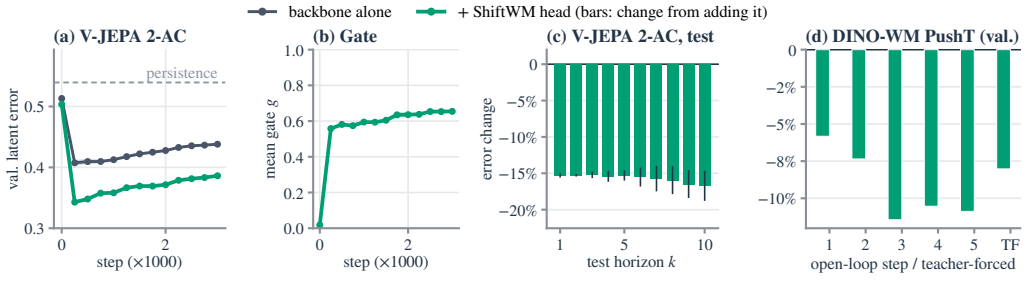


Figure 12: **Plug-in heads during and after training**, same data, schedule and loss with and without the head. (a) V-JEPA 2-AC validation latent error (2 seeds; band: range). (b) Mean gate of its head. (c) Change of test error per horizon (whiskers: all seed pairings). (d) DINO-WM on PushT (1 seed): change of validation latent error per open-loop step and teacher-forced (TF).

D.3 THE FUTURE IS A TRANSPORT OF OBSERVED CONTENT

The *oracle move* picks, for each moving patch, the observed candidate in its transport window closest to the true future; it uses the future to choose and is therefore a model-free reference, not a predictor. With candidates from an unrelated test episode the same oracle is 40% worse than persistence, and 18% worse even when that episode comes from the same lab, so its gain comes from the episode's own content. The share of its gain a model recovers is $(\text{err}_{\text{pers}} - \text{err}_{\text{model}})/(\text{err}_{\text{pers}} - \text{err}_{\text{oracle}})$, with err_{pers} the persistence error, computed over all moving patches of 3,923 windows from 130 DROID test episodes. SHIFTWM has lower error than both Direct and AR in every cell of horizon and motion decile (Figure 13a,b), with the largest gain over Direct for moderate-to-large motion at long horizons and over AR on nearly static patches, where the recursive rollout accumulates error on content that should not change. It recovers a larger share of the oracle gain at every horizon (Figure 13c), and its advantage is not driven by a few outliers, as it has lower error than Direct on 64% of moving patches (Figure 13d). Per episode, the absolute error reduction grows with motion (Figure 8b), and the relative reduction stays positive in every motion quintile (Figure 11d).

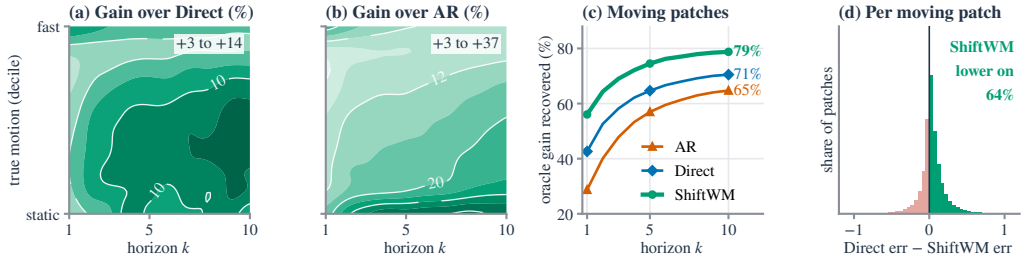


Figure 13: **Where the gains lie (held-out DROID).** (a,b) Error reduction of SHIFTWM per horizon and decile of true patch change (%). (c) Share of the oracle-move gain recovered on moving patches. (d) Per-patch error difference to Direct at k=10 on moving patches (green: SHIFTWM lower).

D.4 BEYOND FEATURE ERROR

Feature warping with optical flow. We warp the last observed grid $\mathbf{Z}_0$ with RAFT (Teed & Deng, 2020) flow pooled to the patch grid, on the same DROID test windows and metric as Table 1. *Flow extrapolation*, as in the warping branch of F2M (Šarić et al., 2020), scales the flow between the last two observed frames by k, whereas the *oracle warp* uses the flow from the true future frame and is thus a strong reference for a pure warp of $\mathbf{Z}_0$; SHIFTWM improves on both, as its gate keeps static content, where estimated flow is noisy, and its correction adds content that no warp of $\mathbf{Z}_0$ contains.

Segmentation view. Feature error does not show *where* a forecast puts the arm, so we read a segmentation out of every forecast. Grounding DINO (Liu et al., 2024) detects the arm or the instruments on the last observed frame and SAM 2.1 (Ravi et al., 2025) tracks the mask through the true future, and we keep windows with a confident, stable track covering 3–128 of the 256 patches (1,332 of 1,500 DROID and 1,017 of 1,500 Hamlyn windows). A forecast patch is foreground if it is more similar (cosine) to a foreground than to a background patch of $\mathbf{Z}_0$, moving windows have below-median mask overlap between t and $t+K$, and placement is the centroid distance. On DROID, SHIFTWM has the highest arm IoU on moving windows, with paired intervals that exclude zero against Direct and AR, and the lowest placement error, whereas on Hamlyn, where the instruments move less within ten steps, the three predictors are close (Figure 14).

Decoded pixels. To show forecasts as images (Figure 1a, Figure 16), we train one feature-to-RGB decoder per dataset on true training features (L1 + LPIPS) and apply it unchanged to every method's forecast.

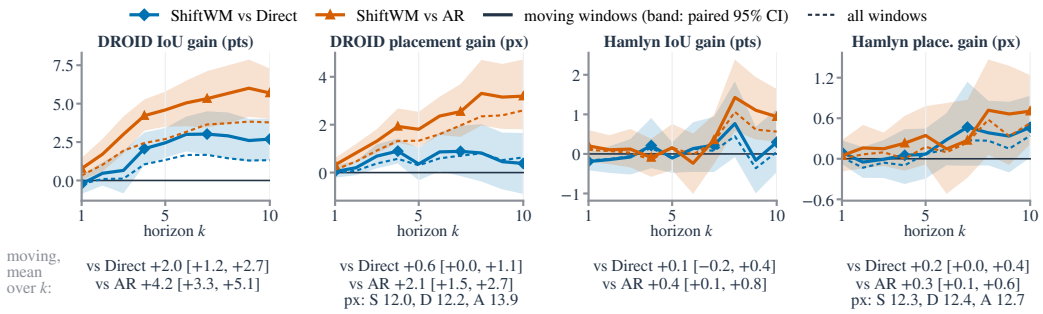


Figure 14: **Segmentation read out of the forecasts.** Paired gain of SHIFTWM over Direct and AR per horizon: IoU (points) and placement error removed (px of the displayed frame); bands: paired 95% CIs on moving windows. Below: moving-window means over k with CIs, and mean placement error of SHIFTWM (S), Direct (D) and AR (A).

D.5 QUALITATIVE RESULTS

Example selection. Every method is shown on the same window, and the examples illustrate rather than replace the aggregate results. Figure 1, the transport weights of Figure 2 and Figure 7 use held-out DROID windows among the 30% with the most motion that have the largest step-10 advantage of SHIFTWM over the better of Direct and AR (over Direct for Figure 7), and Figure 1a decodes step-10 features with the decoder of Section D.4. Figure 15 selects, among test windows in the top half of true motion, the three DROID windows (distinct episodes) and the Hamlyn window with the largest relative k=10 gain over the better of Direct and AR.

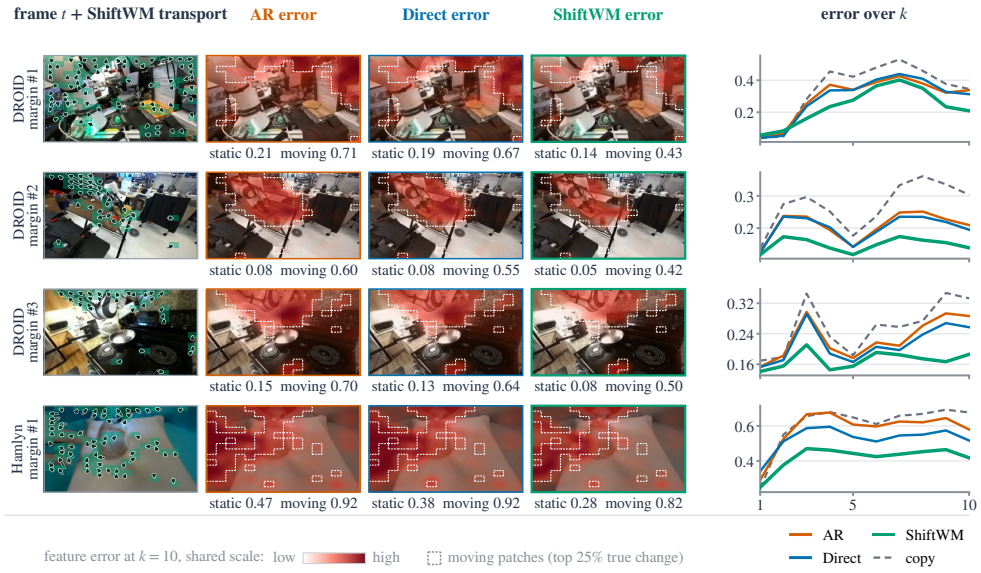


Figure 15: **Largest-margin test windows.** DROID and Hamlyn windows (top half of motion) with the largest k=10 gain of SHIFTWM over the better of Direct and AR. Error maps share one scale (dashed: moving patches).

Where the arm and the instruments go. Figure 16 segments the arm or instruments with SAM 2.1 in each method's decoded k=10 forecast, on the moving windows with the largest placement advantage of SHIFTWM (labeller verified on the true future), which illustrate the paired averages of Figure 14.

Further benchmarks. Figure 17 shows, by the same selection rule, the best Language-Table window, the best V-JEPA 2-AC plug-in window and the best window of the unseen second DROID camera; on

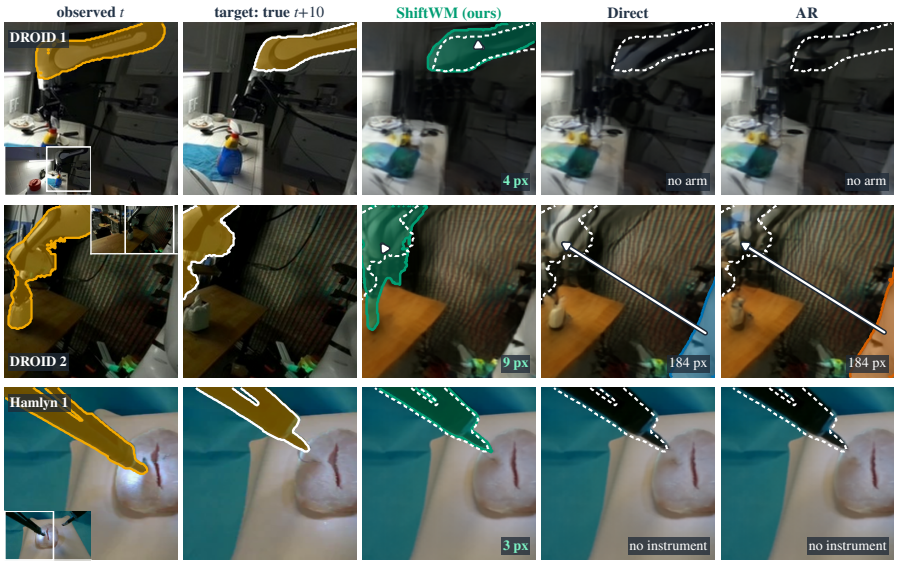


Figure 16: **Selected windows: where each forecast puts the arm or instruments.** Each column is that method's own decoded k=10 forecast of the same held-out window, segmented with SAM 2.1 (filled; dashed: true position at t+10; badges: centroid distance to the true mask).

Language-Table, Direct and AR blur or lose blocks that SHIFTWM keeps in place, on V-JEPA 2-AC the head recovers the arm's token structure, and on the unseen camera SHIFTWM has the lowest error on the moving arm, where AR's rollout drifts most.

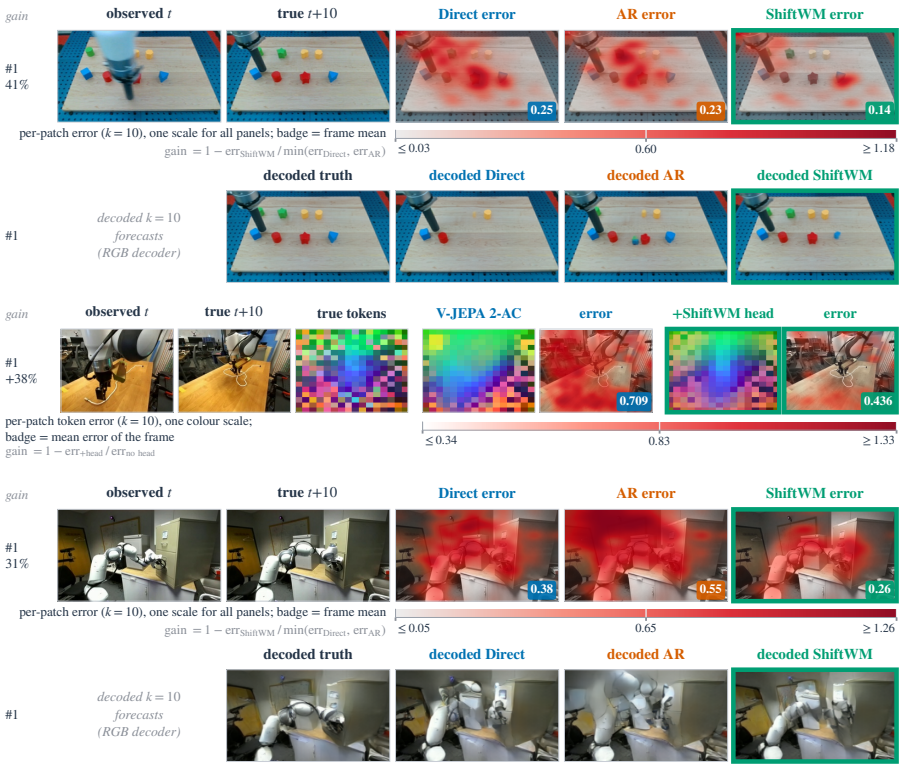


Figure 17: **Further benchmarks** (held-out windows with the largest k=10 gain). Top: Language-Table, per-patch error on one scale and each model's decoded forecast. Middle: V-JEPA 2-AC on DROID with and without the SHIFTWM head; tokens in a shared PCA colour space. Bottom: DROID's second camera, unseen in training (zero-shot).