FLOMO: 3D Scene Flow as a World Action Model Intermediate for Learning from Human Video

Jeremy A. Collins*, Namra Patel*, Ayush Agarwal*, Shitij Govil*, and Animesh Garg
Georgia Institute of Technology

***Abstract*— Pretrained video generation models distill internet-scale motion priors into robot policies, but full video is a redundant signal for action prediction. We argue that dense 3D motion is the medium through which video pretraining best transfers to robot actions, and that predicting motion rather than video affords better sample-efficiency for action prediction. FloMo is a 5B video-generation model fine-tuned to predict 3D scene-flow and action tokens. Rendering scene flow as RGB and encoding it into the same latent space as video enables a pretrained video generation backbone to efficiently adapt to predict 3D scene flow. Co-trained on action-free human video and action-labeled robot teleoperation data, our model benefits from human-video pretraining and generalizes to unseen objects, motion primitives, and tasks. Replacing motion prediction with a video prediction objective degrades out-of-distribution generalization, while adding a video prediction target alongside motion lowers policy success rather than improving it. When motion, video, and action are predicted jointly, action tokens attend far more to motion than to video, suggesting that predicting video wastes model capacity. Together, these results indicate that motion effectively mediates the transfer from video pretraining to action. More information can be found at flomo-wam.github.io.**

I. Introduction

Pretrained video generation models can be used to distill internet-scale motion priors into robot policies. Transfer from video to action is typically attributed to information overlap between RGB videos and the actions that produce them: a model that predicts the world's next pixels has implicitly learned the dynamics that link action to outcome [1]–[3]. Video, however, is a far higher-entropy signal than the action it implies. Videos that differ in texture, lighting, material, and background depict the same underlying motion and imply the same action, so a world action model (WAM) trained to predict pixels spends capacity on appearance that action prediction never uses [4]–[6].

WAMs vary on how tightly they couple the video backbone to the action head, but they share a prediction target: future pixels. We question this assumption and ask: *which prediction target lets video pretraining transfer to action most directly?* We argue that the answer is 3D motion. Given the current observation, the action that produces a future state depends on how the scene transformed rather than on appearance and texture. Predicted frames that differ only in lighting or background share the same motion and imply the same action. Predicting motion rather than pixels hence targets a lower-entropy signal and incorporates this invariance instead of forcing the model to learn it, leaving the action head a tighter information bottleneck it can decode directly.

Our insight is that a pretrained video VAE already provides a latent space for motion. We represent motion as the 3D displacement of each tracked scene point over time in a canonical frame independent of camera motion, otherwise known as *3D scene flow*, and render this vector field as an RGB image. Rendering scene flow as RGB lets us encode it with a pretrained video VAE, giving scene-flow tokens the same latent representation format as video without needing to train a new tokenizer. This lets us finetune only the backbone to predict motion.

We instantiate this in FLOMO, a Wan 2.2 5B DiT that, given an image and language instruction, flow-matches 3D scene-flow and action tokens, with scene flow encoded into the same video latent space the model was pretrained on. We co-train on action-free human video and action-labeled robot teleoperation data, and find that predicting motion alone improves both in-distribution policy success and out-of-distribution generalization, relative to predicting video or predicting video and motion jointly.

Our contributions are:

1) **Motion as a WAM prediction target (§III).** We argue that WAMs should predict motion rather than video: motion is a lower-complexity, texture-invariant target an action head can decode directly, and the priors of video pretraining are carried by the pretrained backbone, not by the prediction target. We realize this by rendering 3D scene flow as an RGB image and encoding it with the pretrained video VAE, so scene-flow tokens share the video latent space and a single backbone flow-matches motion and action without a new tokenizer.
2) **Motion is the most effective prediction target for policy learning (§IV-B.1, §IV-B.2).** Across real-robot manipulation, predicting motion yields higher in-distribution success and out-of-distribution generalization than predicting video or jointly predicting video and motion; adding a video target interferes with action learning rather than complementing it.
3) **Generalization from human video (§IV-C).** Our model generalizes to objects, motion primitives, and tasks seen only in human data.

II. Related Work

A. Learning Robot Policies from Action-Free Videos

Action-labeled robot demonstrations are expensive, while large-scale action-free videos of humans, robots, and objects

*Equal contribution.
Correspondence: sgovil9@gatech.edu

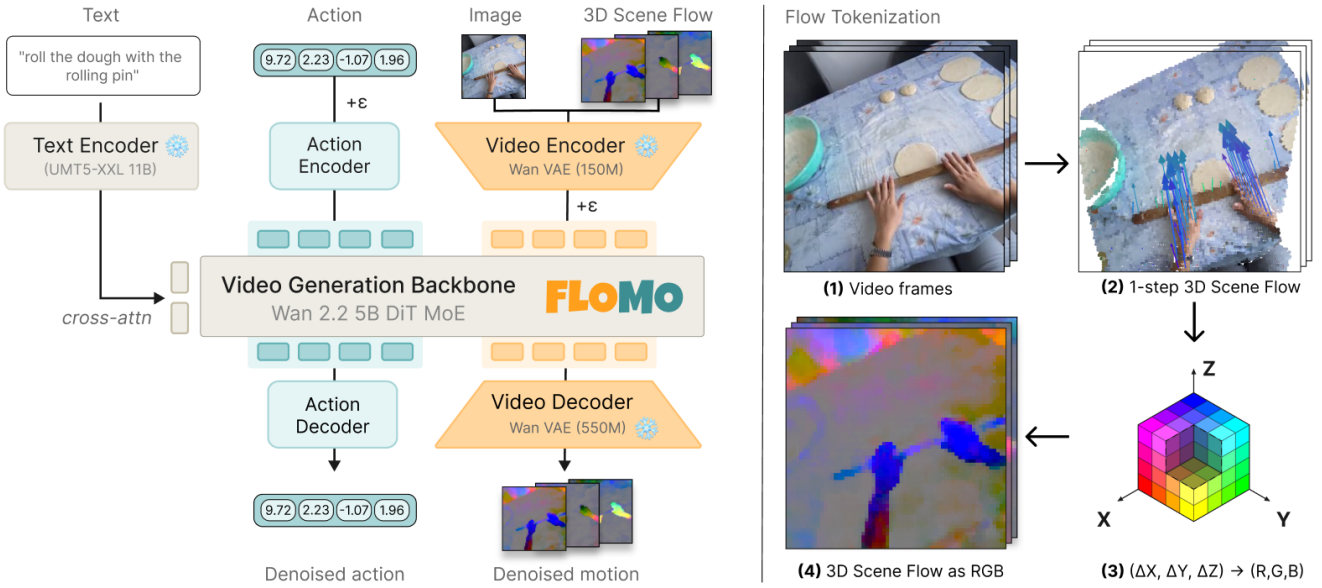


Fig. 1: Given an image and text prompt, FloMo jointly predicts dense 3D motion (scene flow) and robot actions using a pretrained video generation model using flow-matching. To tokenize scene flow, we transform cumulative 3D velocities into RGB videos that are encoded by the pretrained video backbone's video encoder.

are abundant. One line of work uses passive video to pretrain visual representations, rewards, or value functions ([7], [8], [9], [10]). These methods improve data efficiency but do not learn explicit predictive models of future dynamics.

Another line of work learns latent actions or transition variables from unlabeled videos through self-supervised objectives such as future prediction, frame consistency, or reconstruction ([11], [12], [13]). While these methods reduce reliance on action labels, their latents may encode visual information useful for reconstruction but unnecessary for control.

Finally, work on learning from human videos studies cross-embodiment alignment or explicit retargeting to transfer human demonstrations to robots ([14], [15], [16], [17], [18]). These works focus on bridging the human-robot embodiment gap; our work instead focuses on the prediction target inside of a WAM.

B. Motion and Geometry-Aware Video Prediction

Raw pixel prediction is an expensive and indirect objective for learning physical dynamics, motivating methods that introduce motion or geometric intermediates. Some use 2D motion as an interface between action-free video and robot actions, predicting future image-point trajectories and conditioning policies on them ([4], [5]). Others use flow or motion fields as an interface linking video prediction and manipulation ([19], [6], [20], [21]), or separate motion and temporal reasoning from appearance and camera movement in video generation ([22]).

Closest to our representation choice are methods that model dynamic scenes with compact trace spaces, 3D scene flow, or point-track representations ([23], [24], [25]). These works show that 3D motion fields are a natural substrate for future dynamics prediction. Rather than building a bespoke latent space for motion, we render scene flow into a pretrained video-generation latent space.

C. World Action Models

Pretrained video-generation models provide motion priors for robot learning because they are trained on large-scale videos of physical interactions and object motion. Recent work uses these models for control by coupling future video prediction with action generation. One family generates task-conditioned robot videos and then recovers or co-generates actions through inverse dynamics models or latent actions ([26], [27], [1], [2]). These approaches preserve the pretrained video model, but deployment is expensive and performance depends heavily on video fidelity. A second family avoids explicit video rollout at test time by extracting intermediate latents from a pretrained video backbone and training an action head on top ([28], [29], [30]). This reduces inference cost, but action prediction remains limited by the quality of the extracted video features.

More tightly coupled world action models jointly model visual features and actions with shared or entangled architectures ([31], [32], [33], [3], [34]). Our work studies a complementary axis: what future state should the world model predict? Instead of pixels, video latents, or action tokens, we predict 3D scene flow, discarding texture, lighting, and material details that are irrelevant for inverse dynamics.

III. Method

Existing WAMs predict future video jointly with actions, forcing the backbone to model details that low-level control doesn't require. FLOMO predicts 3D scene flow in place of video. We predict within the latent space of a pretrained video generator (Wan 2.2), so the model inherits large-scale video priors through its backbone. Our model is co-trained on

egocentric human video and robot teleoperation, predicting both scene flow and actions from an initial observation and a language instruction. Figure 1 gives an overview of the architecture.

A. Problem Formulation

We assume access to two types of datasets: an *egocentric video dataset* $\mathcal{V} = \{(\mathbf{o}_{0:T-1}, \ell)\}$ of RGB observation sequences $\mathbf{o}_{0:T-1} \in \mathbb{R}^{T\times H\times W\times 3}$ ($T = 16$) paired with language instructions ℓ, and a *robot interaction dataset* $\mathcal{R} = \{(\mathbf{o}_{0:T-1}, \ell, \mathbf{a})\}$ that additionally provides an action chunk $\mathbf{a} = (\mathbf{a}_0, \ldots, \mathbf{a}_{T-1}) \in \mathbb{R}^{T\times d_a}$ over a d_a-dimensional action space. For all clips in both datasets, we extract a 3D scene-flow sequence $\mathbf{F}_{0:T-1}$ in an offline preprocessing step (Sec. III-B). Given an initial observation $\mathbf{o}_0$ and instruction ℓ, our model predicts future scene flow and actions; the full joint distribution our architecture represents is:

$$p(\mathbf{F}_{0:T-1},\ \mathbf{a} \mid \mathbf{o}_0,\ \ell)\,, \tag{1}$$

B. 3D Scene Flow as the Video-Action Intermediate

The action producing a future state depends on how the scene moves, not on how it looks, motivating motion rather than pixels as the prediction target. 2D optical flow captures motion but conflates camera ego-motion with task-relevant motion. We instead predict 3D scene flow in a canonical frame, which isolates the geometric transition between states while discarding appearance and camera motion.

Scene-flow extraction and canonical frame. We extract 3D point trajectories in an offline preprocessing step using SpatialTrackerV2 [35], an off-the-shelf 3D point tracker. For each clip in $\mathcal{V}$ and $\mathcal{R}$, we slide a window of $T = 16$ frames and place $Q = 4096$ query points on a 64×64 grid covering the first frame. The tracker back-projects these queries to 3D and propagates them across the window, yielding trajectories $\{\mathbf{p}_i(u, v)\}_{i=0}^{T-1}$. We express every trajectory in the camera frame of the window's first (conditioning) timestep, so the coordinate origin is the current camera's focal point. This makes the representation invariant to camera ego-motion and yields a motion field that primarily reflects object and hand movement. Extraction runs at $\approx$2.4 s per 16-frame window on a single NVIDIA L40S GPU.

Cumulative scene flow. Rather than supervising on instantaneous per-frame velocities, we predict *cumulative* scene flow, defined as the total 3D displacement of each point from the first frame to frame i:

$$\mathbf{F}_i(u, v) \;=\; \mathbf{p}_i(u, v) \;-\; \mathbf{p}_0(u, v), \tag{2}$$

so $\mathbf{F}_0 = \mathbf{0}$ by construction. We find cumulative scene flow to be a smoother target than instantaneous scene flow: it reduces the per-frame tracking noise that dominates the supervision signal, letting fine-grained motion like fingertip and object movement be predicted accurately.

C. Tokenized Scene Flow

We render the cumulative scene flow $\mathbf{F}_{0:T-1}$ as an RGB video $\mathbf{F}^{\text{rgb}}$ by clipping each displacement channel to its global per-dataset 1st/99th percentile range and rescaling to $[0, 1]$, then treat it exactly like an RGB video. We encode it with the frozen Wan 2.2-TI2V VAE encoder $\mathcal{E}$:

$$\mathbf{z}^f \;=\; \mathcal{E}\big(\mathbf{F}^{\text{rgb}}\big) \;\in\; \mathbb{R}^{T_z\times c\times H_z\times W_z}, \tag{3}$$

where T_z is the number of latent frames (temporal stride $s_T = 4$, with the first frame encoded on its own to match the TI2V backbone), $c = 48$ is the VAE latent channel count, and $H_z \times W_z$ are the latent spatial dimensions (spatial stride 16). Representing scene flow as an RGB video lets us reuse the pretrained VAE without modification, so scene flow and video latents share similar distributions and the backbone does not significantly diverge from its video pretraining.

D. Architecture

FLOMO is instantiated from the pretrained Wan 2.2 5B Text-Image-to-Video DiT [36], extended with lightweight modality-specific encoder and decoder heads. The current observation is encoded by the frozen VAE, $\mathbf{z}^o = \mathcal{E}(\mathbf{o}_0)$, forming a clean conditioning prefix, while the frozen umT5-XXL text encoding $\boldsymbol{\tau} = \text{umT5}(\ell)$ [37] conditions the backbone through cross-attention. The cumulative scene-flow latents $\mathbf{z}^f$ and action tokens are the prediction targets; future RGB latents $\mathbf{z}^v = \mathcal{E}(\mathbf{o}_{1:T-1})$ are an additional target used only in video-prediction ablations. For the actions, flow matching is performed on the normalized d_a-dimensional action space. At noise level t, each noised action $a_{t,i} \in \mathbb{R}^{d_a}$ is mapped to a DiT token by a learned linear projection $\mathbb{R}^{d_a} \to \mathbb{R}^d$.

All target tokens are concatenated along the sequence dimension and jointly denoised by the shared DiT backbone. We fine-tune the backbone using LoRA [38] ($r = 64$, $\alpha = 128$, dropout 0), injecting adapters into the query/key/value/output projections of both self- and cross-attention and into both feed-forward layers of all 30 blocks; the Wan VAE and text encoder stay frozen. This leaves $\approx$161M trainable parameters on top of the frozen 5B backbone. We use bidirectional (full) attention, as in Wan, so every token attends to every other. Each prediction head can be independently activated or dropped per sample via loss masking, so the same backbone supports any subset of output modalities. By default, our model predicts scene flow and actions; our ablations reuse the same flexible architecture to additionally predict future video, or to predict video and actions without scene flow.

E. Training Objective

We adopt a flow-matching objective [39] over all prediction targets. For any clean target $\mathbf{x}_0 \in \{\mathbf{z}^f, \mathbf{a}, \mathbf{z}^v\}$, we form the noisy interpolant at a single noise level $t \sim \mathcal{U}[0, 1]$ shared across all modalities, $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, with velocity target $\mathbf{v} = \boldsymbol{\epsilon} - \mathbf{x}_0$. For actions, this interpolation is applied in the normalized action space before the learned action projection. The training loss is

$$\begin{aligned}\mathcal{L}(\theta) \;= \mathbb{E}_{\mathbf{x}_0,\,\boldsymbol{\epsilon},\,t}\Big[&\lambda_f\|\hat{\mathbf{v}}^f - \mathbf{v}^f\|_2^2 + \lambda_a\|\hat{\mathbf{v}}^a - \mathbf{v}^a\|_2^2 \\ &+ \lambda_v\|\hat{\mathbf{v}}^v - \mathbf{v}^v\|_2^2\Big],\end{aligned} \tag{4}$$

where $\lambda_a = 0$ for samples without action labels. By default, our model predicts motion and action only ($\lambda_f = 1$, $\lambda_a = 0.1$, $\lambda_v = 0$); the video term ($\lambda_v = 1$) is enabled only for ablations that predict future video. Actions use a 20-dimensional bimanual end-effector space (per arm: 3D position, 6D rotation, 1D gripper), normalized via 2nd/98th percentile clipping following [40].

F. Training Procedure

We train our model in two stages using a mixture of egocentric human video and robot interaction data. Both stages use AdamW with a constant learning rate and bf16 mixed precision. Mid-training runs ≈14,000 optimizer steps at global batch size 512 and learning rate 2.8×10^{-4}. Fine-tuning runs ≈1,500 steps at batch size 256 and learning rate 5.6×10^{-5}.

Stage 1: Mid-training. We train on the full data mixture combining human video and robot interaction datasets. Human video samples are supervised on the scene-flow loss only ($\lambda_a = 0$, $\lambda_v = 0$), while robot samples are supervised on scene flow and action. Dataset-specific batch weights control the relative sampling frequency. This stage exposes the backbone to diverse egocentric motion at scale, building a strong motion prior before any action-specific finetuning.

Stage 2: Fine-tuning. In a second stage, we exclusively train on robot interaction data, dropping the human video stream entirely. This sharpens action prediction without diluting the gradient signal with action-free samples, and we raise the action loss weight to $\lambda_a = 1$.

G. Inference

At test time, given an initial observation $\mathbf{o}_0$ and language instruction ℓ, our model jointly denoises its output heads by integrating the learned velocity field with the UniPC ODE solver [41] under a shifted noise schedule (shift 5) for $N = 10$ steps; we apply no classifier-free guidance. The robot executes the full predicted chunk of $T = 16$ actions before re-planning from the next observation. One cycle takes ≈0.7 s on a single NVIDIA H100 GPU, and asynchronous execution overlaps planning with robot motion.

IV. Experiments

We organize our experiments for FLOMO around three questions.

(1) Does predicting 3D motion rather than video improve policy performance? We compare FloMo with matched video-prediction variants and prior methods on in-distribution real-robot tasks, out-of-distribution real-robot tasks, and the RoboTwin benchmark (Secs. IV-B.1–IV-B.3).

(2) Does action-free human video enable out-of-distribution transfer? We remove the human-video portion of the training mixture and measure the resulting change in performance on novel objects, tasks, and motion primitives (Sec. IV-C).

(3) Does motion serve an effective intermediate for action prediction? We use an inverse-dynamics probe to measure how efficiently actions can be recovered from ground-truth motion versus ground-truth video, and analyze how the action stream allocates attention across modalities (Secs. IV-D.1 and IV-D.2).

A. Experimental Setup

Datasets. We train on a mixture of action-free egocentric human video and action-labeled bimanual YAM robot teleoperation. The human-video data is primarily drawn from **EgoVerse** [42]: 81 h of egocentric human video, filtered for diversity from a 1,362-hour corpus. The robot component comprises 10 h selected for diversity from 700 hours of **MolmoAct2-BimanualYAM** teleoperation [43], 3.7 h of internal task-directed YAM demonstrations, and 0.4 h of internal undirected YAM play. The physical YAM setup copies that of MolmoAct2. We also include 0.7 h of internal, action-free human demonstrations on tasks that overlap with robot data. Human clips provide video and scene-flow supervision. Robot demonstrations also provide action supervision. Mid-training uses the full mixture. Fine-tuning uses the in-house YAM demonstration and play data.

Baselines. We select baselines for each experiment according to each evaluation's main question. For in-distribution real-world evaluation, we compare FloMo with matched variants that substitute joint motion-action (MA) prediction for video-action (VA) prediction and video-motion-action (VMA) prediction to isolate whether action prediction benefits from motion, video, or both.

For out-of-distribution real-robot evaluation, we additionally compare against two prior works that represent close alternatives to our method: **AMPLIFY** [5], which predicts sparse, tokenized 2D point tracks as an intermediate between video and action, and **DreamZero** [3], which predicts future video jointly with actions. This comparison tests whether 3D motion transfers better than either image-space motion or pixel prediction under shifts in objects, tasks, and motion primitives. We train all the models on the data mixture described above until convergence.

On the RoboTwin [44] benchmark, we compare against $\boldsymbol{\pi_{0.5}}$ [45], **FAST-WAM** [34], and an action-only **Diffusion Policy** [46] (Behavior Cloning) baseline. These choices represent a pretrained Vision-Language-Action (VLA) policy, a video-based WAM, and a policy trained without any video pretraining. The Behavior Cloning model is a 20M parameter, 8-layer diffusion transformer with the Wan architecture. For Fast-WAM, we use the same rank-64 Wan-backbone LoRA configuration as FloMo and match the trainable capacity of its action head. We fine-tune $\pi_{0.5}$ end-to-end using its native training recipe, evaluating the VLA in its standard adaptation setting. Across methods, we hold the data and evaluation protocol fixed.

Evaluation protocol. For real-world rollouts, we report task success rate (SR), as the fraction of rollouts in which the policy completes the task. For the out-of-distribution evaluation, we also report task progress (TP), as the average fraction of task stages completed. We conduct 15 rollouts per task for the in-distribution real-robot evaluation and 10

Fig. 2: In-distribution success rates. Across 15 rollouts FloMo performs best on every task; the matched video–motion–action (VMA) and video–action (VA) variants lower success.

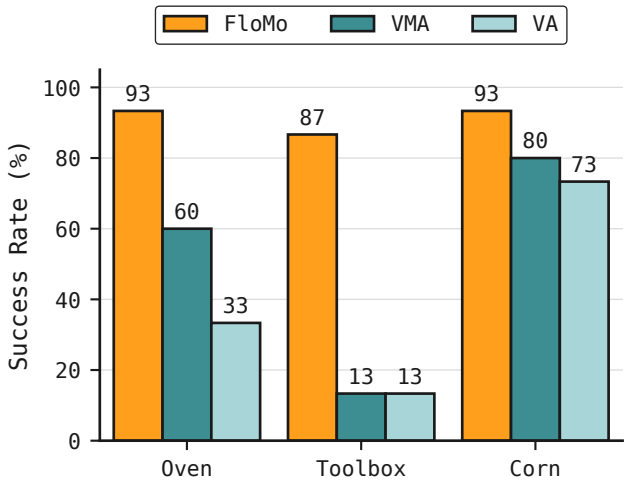


Table I: Out-of-distribution success rates and human-video ablation. We evaluate generalization across unseen motion primitives, tasks, and objects. The main comparison uses 10 rollouts per condition (30 total); the no-human-video ablation uses 5 rollouts per condition (15 total). SR = successful rollouts / total; TP = avg. fraction of subtasks completed (%).

Method	Close Cabinet		Pull String		Highlighter		Aggregate	
	SR	TP	SR	TP	SR	TP	SR	TP
VMA	0/10	0	0/10	0	2/10	45	2/30	15
VA	0/10	20	1/10	10	1/10	15	2/30	15
DreamZero	5/10	**75**	1/10	15	0/10	0	6/30	30
AMPLIFY	**6/10**	70	0/10	0	0/10	0	6/30	23
FLOMO (no human video)	0/5	20	1/5	50	0/5	20	1/15	30
FLOMO (MA)	**6/10**	70	**7/10**	**70**	**5/10**	**65**	**18/30**	**68**

rollouts per task for the out-of-distribution evaluation. The no-human-video ablation is done over 5 rollouts per task. In RoboTwin, we report success in 100 Easy-setting episodes per task and method.

B. Policy Evaluation

1) In-Distribution Real-Robot Tasks:

Setup. We evaluate on three contact-rich, precision tasks on the bimanual YAM platform, modelling that from MolmoAct2 [43]. *Oven* requires opening a toaster-oven door, retrieving a croissant from inside, and placing it on a plate. *Toolbox* requires opening a hinged lid via a small crevice, grasping a screwdriver, and placing it in the container. *Corn* requires grasping a corn cob and placing it in a bowl. We run 15 rollouts per task and report success rate. We compare FloMo against the VMA and VA prediction-target variants.

Results. Figure 2 reports in-distribution results. FloMo achieves the highest success rate on every task, with an average success rate of 91%, compared with 51% for VMA and 40% for VA. Under the matched architecture and training protocol, adding video prediction to the motion–action objective reduces average success by 40 percentage-points, while replacing motion prediction with video prediction reduces it by 51 percentage-points. The difference is largest in *Toolbox*, where FloMo achieves 87% success while both video-prediction variants achieve only 13%. Opening the toolbox requires locating a narrow crevice and excuting contact-rich, precise motion. Predicting 3D motion directly emphasizes the geometry of this contact.

2) Out-of-Distribution Real-Robot Generalization:

Setup. We probe generalization along three axes, each isolating a distinct distribution shift. *Close Cabinet* involves pushing a cabinet door shut, an unseen motion primitive in robot data during training. *Pull String* requires pulling a hanging string to a target mark, an unseen task. *Highlighter* introduces an unseen object (highlighter) which must be placed in a bowl. We run 10 rollouts per condition and report success rate (SR) and task progress (TP) for FloMo, VMA, VA, AMPLIFY [5], and DreamZero [3]. The no-human-video ablation in IV-C uses 5 rollouts per condition. Each task has a single intermediate stage: contacting the cabinet door for *Close Cabinet*, grasping the string for *Pull String*, and picking up the highlighter for *Highlighter*. TP is the fraction of these two stages a rollout completes, averaged over rollouts: 0 if it never reaches the intermediate stage, 50% if it reaches it but does not finish, and 100% if it completes the task.

Results. Table I reports out-of-distribution results. Among the full-data methods, FloMo is the only method that generalizes consistently across all three distribution shifts. It achieves an average success rate of 60% and average task progress of 68%, compared with 20% SR and 23% TP for AMPLIFY, 20% SR and 30% TP for DreamZero, and 7% SR and 15% TP for both VMA and VA.

All methods that predict pixels through a video objective (VMA, VA, DreamZero), or through image-space 2D tracks (AMPLIFY), degrade in performance. We hypothesize that FloMo retains performance because its 3D scene-flow representation is invariant to appearance and camera movement in pretraining and mid-training data. Removing the image-space prediction target recovers generalization, and adding the video target, even alongside motion, coincided with lower OOD success in these evaluations.

We attribute this gap to what each target encodes. A video objective binds the model to scene-specific appearance, texture, and object identity, none of which are preserved under the distribution shifts we test. One possible explanation is sensitivity to training-set appearance statistics. By construction, 3D scene flow discards these factors and retains only the geometric transition between states, a representation that is invariant to appearance and shared across embodiments. This is what allows FloMo to transfer a motion primitive such as string-pulling, observed only in human video, to the robot: on *Pull String*, FloMo succeeds on 7/10 rollouts while no baseline exceeds 1/10.

3) RoboTwin Benchmark:

Setup. We evaluate single-task adaptation on five RoboTwin tasks spanning bimanual grasping (*Pick Dual Bottles*), sequential placement (*Stack Bowls*), articulated-object manipulation (*Open Laptop*), precision placement (*Hanging Mug*), and bimanual transfer (*Handover Microphone*). For

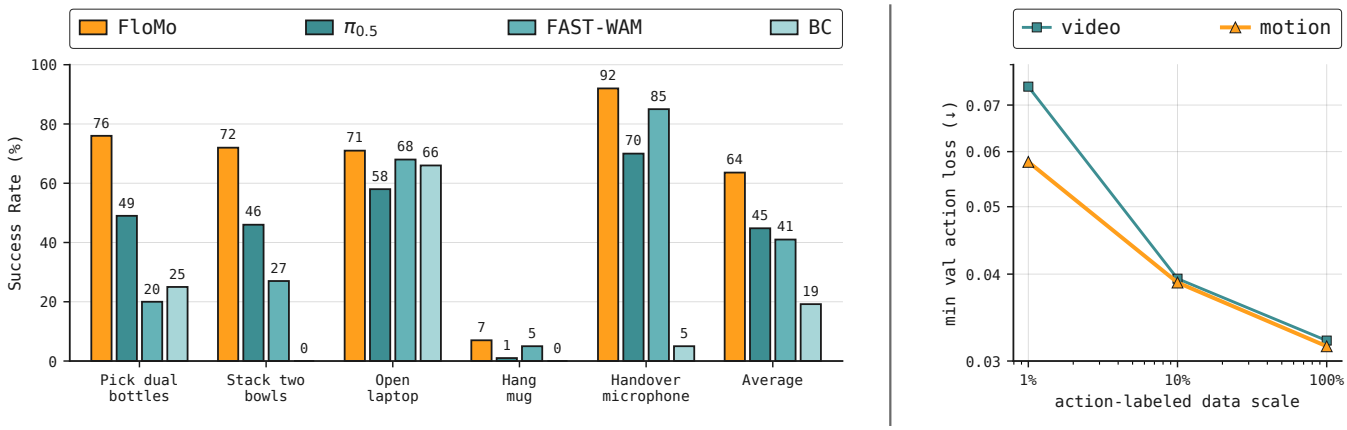


Fig. 3: (a) RoboTwin single-task evaluation. Recorded RoboTwin Easy success rates across five tasks, with 100 evaluations per task and method. Across all tasks, FloMo performs the best. **(b) Inverse-dynamics scaling.** Held-out action error versus action-data scale for two *ground-truth* conditioning signals (log-log). Recovering actions from motion is more sample-efficient than from video: the gap is largest at 1% of the data and narrows as data grows.

each task and method, we train a policy on the 50 official demonstrations. The runs use 5,000 gradient steps and a batch size of 256. We evaluate the models on 100 episodes from the RoboTwin Easy setting.

Results. Figure 3(a) shows that FloMo achieves the highest success rate on all five robotwin tasks. Averaged across tasks, FloMo achieves 63.6% success, compared with 44.8% for $\pi_{0.5}$, 41.0% for FAST-WAM, and 19.2% for BC. These results show that predicting 3D motion provides an effective interface for transferring video pretraining to action prediction across a distinct set of simulated tasks.

C. Transfer from Human Video

Table I reports both the full-data comparison and the no-human-video ablation. To determine the impact of human video, we train a separate FloMo variant with the ≈81 h EgoVerse subset and 0.7 h of internal action-free human demonstrations removed. Removing human video lowers average success from 60% to 7%, suggesting that much of FloMo's out-of-distribution performance comes from human video rather than robot demonstrations.

The per-subtask scores show where the model trained without human video fails. It can begin each task but rarely completes the novel motion: it contacts the cabinet on 2 of 5 trials but never closes it (0/5), grasps the string on 4 of 5 trials but successfully pulls it on only one, and grasps the highlighter on 2 of 5 trials but never places it in the bowl.

We verified this by direct text search of the training data. The robot data, both the MolmoAct2 episodes and the YAM demonstrations, contains no references to cabinets, strings, or markers. The EgoVerse subset, by contrast, contains all three motion classes explicitly: cabinet closing (e.g. "close cabinet door under sink"), string pulling (e.g. "pull string from spool"), and slender-object placement (e.g. "place pen lid on pen, place pen in bowl"), even though the exact word "highlighter" never appears. Since these motions are reachable only through the human video, the transfer in Table I likely originates there.

D. Analysis of Motion as an Action Intermediate

1) Inverse-Dynamics Sample Efficiency:

The policy results above establish motion as the more effective prediction target for WAMs, but the source of this gain remains ambiguous: it could stem from more accurate forward-dynamics predictions or a more decodable conditioning representation for inverse dynamics. We isolate the latter with a controlled inverse-dynamics probe. We repurpose FloMo to predict future action chunks from *ground-truth* future signals, eliminating forward-dynamics error so that any remaining difference in action error is attributable to the conditioning representation alone. We compare conditioning on ground-truth future scene flow against ground-truth future RGB, training each on $\{1\%, 10\%, 100\%\}$ of a filtered subset of **MolmoAct2-BimanualYAM** and reporting the minimum held-out action error.

Fig. 3(b) demonstrates that motion is the more sample-efficient signal: at 1% of the action data, it yields roughly 22% lower action error than video (0.058 vs. 0.074); the gap collapses to roughly $1\% - 2\%$ at scales above 10% of the dataset. Because the ground truth future is supplied in both conditions, this gap is a property of the representation quality rather than downstream performance, as in practice the actions are *jointly predicted* with the video or motion rather than conditioned on ground truth futures. This experiment shows, however, that motion exposes the action-relevant structure directly without losing predictive information, whereas recovering it from raw RGB video takes more examples. Motion is thus both an easier target to predict and the more efficient representation of the information needed to predict actions.

2) Action Attention Across Modalities:

To compare the importance of each modality for action prediction, we investigate the attention maps of the VMA variant. We hypothesize that if motion is an effective intermediate in our evaluated setting, then action tokens should attend predominantly to motion tokens rather than video tokens, and this pattern should persist across embodiments.

Figure 4 reports the share of action-query attention going

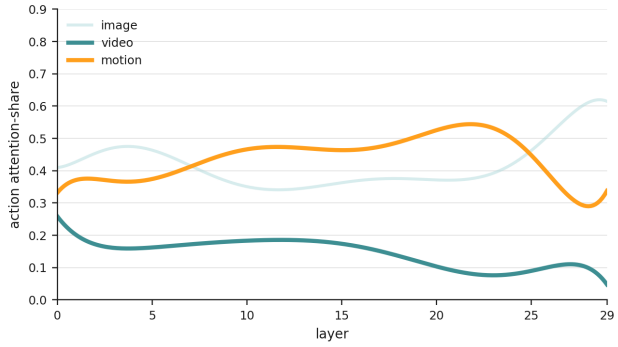


Fig. 4: Action attention concentrates on motion tokens. Share of action-query attention to each input modality, averaged over attention heads and robot validation rollouts, as a function of DiT layer. Attention to motion rises through the trunk and dominates at the late-middle layers, while attention to video stays lowest throughout; averaged over layers, action queries attend to motion roughly $3\times$ more than to video.

to each input modality (image, video, motion) as a function of DiT layer, aggregated across attention heads, averaged over $1,000$ held-out samples. Beginning from an approximately uniform distribution at the input layer, action queries progressively reallocate attention toward motion tokens through the network trunk, reaching $\sim 55\%$ at the late-middle layers. Attention to video exhibits the opposite trend, remaining the lowest across all layers and declining to $\sim 10\%$ at the deepest layers. Averaged over attention heads and layers, action queries attend to motion approximately $3\times$ more than to video. Taken together, these results indicate that the action stream draws predominantly on motion rather than video, supporting our hypothesis that 3D scene flow is an effective intermediate in our evaluated setting.

V. Discussion and Limitations

We argue that the priors of video pretraining are carried by a model's weights, not by what it is trained to predict. A world action model should target motion rather than pixels: an appearance-invariant geometric signal that an action head can decode directly. Rendering 3D scene flow into the latent space of a pretrained video generator lets a single backbone predict motion and action without a new tokenizer, inheriting internet-scale video priors through its weights. Across real-robot manipulation, predicting motion outperforms predicting video and jointly predicting video and motion, both in distribution and on unseen objects, motion primitives, and tasks; action queries attend far more to motion than to video; and a controlled inverse-dynamics probe shows motion is the more sample-efficient conditioning signal. These results point to motion as an effective intermediate in our evaluated setting.

Two limitations temper these results. First, extracting 3D scene flow adds an offline preprocessing step for every clip. Second, FloMo predicts motion and action jointly, and although the two usually agree, they sometimes diverge. A plausible predicted motion does not guarantee a correct action, because the action head is not constrained to be exactly consistent with the predicted flow, and distinct actions can in principle produce the same scene motion. In these cases, treating predicted motion as a proxy for the executed action breaks down, and tightening the coupling between the two predictions through mechanisms such as per-modality variance schedules or objectives to maximize mutual information between modalities is a natural direction for future work. Three additional failure modes occur: scene flow is predicted for the front camera only, so motion estimates degrade when the manipulated object is occluded in that view; instruction grounding is less reliable when a named object is unfamiliar or a similar distractor is present; and on occlusion-heavy clips the off-the-shelf tracker yields noisy trajectories that enter training as degraded targets.

References

[1] J. Jang, S. Ye, Z. Lin, J. Xiang, J. Bjorck, Y. Fang, F. Hu, S. Huang, K. Kundalia, Y.-C. Lin *et al.*, "DreamGen: Unlocking generalization in robot learning through video world models," *arXiv preprint arXiv:2505.12705*, 2025. [Online]. Available: https://arxiv.org/abs/2505.12705

[2] J. Liang, P. Tokmakov, R. Liu, S. Sudhakar, P. Shah, R. Ambrus, and C. Vondrick, "Video generators are robot policies," *arXiv preprint arXiv:2508.00795*, 2025. [Online]. Available: https://arxiv.org/abs/2508.00795

[3] S. Ye, Y. Ge, K. Zheng, S. Gao, S. Yu, G. Kurian, S. Indupuru, Y. L. Tan, C. Zhu, J. Xiang *et al.*, "World action models are zero-shot policies," *arXiv preprint arXiv:2602.15922*, 2026. [Online]. Available: https://arxiv.org/abs/2602.15922

[4] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel, "Any-point trajectory modeling for policy learning," in *Robotics: Science and Systems (RSS)*, 2024. [Online]. Available: https://arxiv.org/abs/2401.00025

[5] J. A. Collins, L. Cheng, K. Aneja, A. Wilcox, B. Joffe, and A. Garg, "AMPLIFY: Actionless motion priors for robot learning from videos," *arXiv preprint arXiv:2506.14198*, 2025. [Online]. Available: https://arxiv.org/abs/2506.14198

[6] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song, "Flow as the cross-domain manipulation interface," in *Conference on Robot Learning (CoRL)*, 2024. [Online]. Available: https://arxiv.org/abs/2407.15208

[7] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3M: A universal visual representation for robot manipulation," in *Conference on Robot Learning (CoRL)*, 2022. [Online]. Available: https://arxiv.org/abs/2203.12601

[8] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang, "VIP: Towards universal visual reward and representation via value-implicit pre-training," in *International Conference on Learning Representations (ICLR)*, 2023, notable-Top-25% (Spotlight). [Online]. Available: https://arxiv.org/abs/2210.00030

[9] C. Bhateja, D. Guo, D. Ghosh, A. Singh, M. Tomar, Q. Vuong, Y. Chebotar, S. Levine, and A. Kumar, "Robotic offline RL from internet videos via value-function pre-training," *arXiv preprint arXiv:2309.13041*, 2023. [Online]. Available: https://arxiv.org/abs/2309.13041

[10] N. Dashora, D. Ghosh, and S. Levine, "ViVa: Video-trained value functions for guiding online RL from diverse data," *arXiv preprint arXiv:2503.18210*, 2025. [Online]. Available: https://arxiv.org/abs/2503.18210

[11] J. Bruce, M. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps *et al.*, "Genie: Generative interactive environments," in *International Conference on Machine Learning (ICML)*, 2024. [Online]. Available: https://arxiv.org/abs/2402.15391

[12] S. Ye, J. Jang, B. Jeon, S. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin *et al.*, "Latent action pretraining from videos," in *International Conference on Learning Representations (ICLR)*, 2025. [Online]. Available: https://arxiv.org/abs/2410.11758

[13] Y. Chen, Y. Ge, W. Tang, Y. Li, Y. Ge, M. Ding, Y. Shan, and X. Liu, "Moto: Latent motion token as the bridging language for learning robot manipulation from videos," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. [Online]. Available: https://arxiv.org/abs/2412.04445

[14] R. Punamiya, D. Patel, P. Aphiwetsa, P. Kuppili, L. Y. Zhu, S. Kareer, J. Hoffman, and D. Xu, "EgoBridge: Domain adaptation for generalizable imitation from egocentric human data," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. [Online]. Available: https://arxiv.org/abs/2509.19626

[15] S. Kareer, D. Patel, R. Punamiya, P. Mathur, S. Cheng, C. Wang, J. Hoffman, and D. Xu, "EgoMimic: Scaling imitation learning via egocentric video," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2025. [Online]. Available: https://arxiv.org/abs/2410.24221

[16] H. Kim, J. Kang, H. Kang, M. Cho, S. J. Kim, and Y. Lee, "UniSkill: Imitating human videos via cross-embodiment skill representations," in *Conference on Robot Learning (CoRL)*, 2025. [Online]. Available: https://arxiv.org/abs/2505.08787

[17] J. Li, Y. Zhu, Y. Xie, Z. Jiang, M. Seo, G. Pavlakos, and Y. Zhu, "OKAMI: Teaching humanoid robots manipulation skills through single video imitation," in *Conference on Robot Learning (CoRL)*, 2024, oral. [Online]. Available: https://arxiv.org/abs/2410.11792

[18] P. Dan, K. Kedia, A. Chao, E. W. Duan, M. A. Pace, W.-C. Ma, and S. Choudhury, "X-Sim: Cross-embodiment learning via real-to-sim-to-real," in *Conference on Robot Learning (CoRL)*, 2025, oral. [Online]. Available: https://arxiv.org/abs/2505.07096

[19] C. Yuan, C. Wen, T. Zhang, and Y. Gao, "General flow as foundation affordance for scalable robot learning," in *Conference on Robot Learning (CoRL)*, 2024. [Online]. Available: https://arxiv.org/abs/2401.11439

[20] L.-H. Lin, Y. Cui, A. Xie, T. Hua, and D. Sadigh, "FlowRetrieval: Flow-guided data retrieval for few-shot imitation learning," in *Conference on Robot Learning (CoRL)*, 2024. [Online]. Available: https://arxiv.org/abs/2408.16944

[21] C. Gao, H. Zhang, Z. Xu, Z. Cai, and L. Shao, "FLIP: Flow-centric generative planning as general-purpose manipulation world model," in *International Conference on Learning Representations (ICLR)*, 2025. [Online]. Available: https://arxiv.org/abs/2412.08261

[22] S. Liu, X. Ren, T. Shen, H. Ling, S. Gupta, S. Wang, S. Fidler, and J. Gao, "MoRight: Motion control done right," *arXiv preprint arXiv:2604.07348*, 2026. [Online]. Available: https://arxiv.org/abs/2604.07348

[23] S. Lee, Y. Jung, I. Chun, Y.-C. Lee, Z. Cai, H. Huang, A. Talreja, T. D. Dao, Y. Liang, J.-B. Huang *et al.*, "TraceGen: World modeling in 3D trace space enables learning from cross-embodiment videos," *arXiv preprint arXiv:2511.21690*, 2025. [Online]. Available: https://arxiv.org/abs/2511.21690

[24] J. Guo, X. Ma, Y. Wang, M. Yang, H. Liu, and Q. Li, "FlowDreamer: A RGB-D world model with flow-based motion representations for robot manipulation," *IEEE Robotics and Automation Letters (RA-L)*, 2026. [Online]. Available: https://arxiv.org/abs/2505.10075

[25] A. Hung, B. P. Duisterhof, and J. Ichnowski, "3PoinTr: 3D point tracks for robot manipulation pretraining from casual videos," *arXiv preprint arXiv:2603.08485*, 2026. [Online]. Available: https://arxiv.org/abs/2603.08485

[26] Y. Du, M. Yang, B. Dai, H. Dai, O. Nachum, J. B. Tenenbaum, D. Schuurmans, and P. Abbeel, "Learning universal policies via text-guided video generation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. [Online]. Available: https://arxiv.org/abs/2302.00111

[27] P.-C. Ko, J. Mao, Y. Du, S.-H. Sun, and J. B. Tenenbaum, "Learning to act from actionless videos through dense correspondences," in *International Conference on Learning Representations (ICLR)*, 2024.

[28] Y. Wen, J. Lin, Y. Zhu, J. Han, H. Xu, S. Zhao, and X. Liang, "VidMan: Exploiting implicit dynamics from video diffusion model for effective robot manipulation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. [Online]. Available: https://arxiv.org/abs/2411.09153

[29] Y. Hu, Y. Guo, P. Wang, X. Chen, Y.-J. Wang, J. Zhang, K. Sreenath, C. Lu, and J. Chen, "Video prediction policy: A generalist robot policy with predictive visual representations," in *International Conference on Machine Learning (ICML)*, 2025, spotlight. [Online]. Available: https://arxiv.org/abs/2412.14803

[30] J. Pai, L. Achenbach, V. Montesinos, B. Forrai, O. Mees, and E. Nava, "mimic-video: Video-action models for generalizable robot control beyond VLAs," *arXiv preprint arXiv:2512.15692*, 2025. [Online]. Available: https://arxiv.org/abs/2512.15692

[31] C. Zhu, R. Yu, S. Feng, B. Burchfiel, P. Shah, and A. Gupta, "Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets," in *Robotics: Science and Systems (RSS)*, 2025. [Online]. Available: https://arxiv.org/abs/2504.02792

[32] S. Li, Y. Gao, D. Sadigh, and S. Song, "Unified video action model," in *Robotics: Science and Systems (RSS)*, 2025. [Online]. Available: https://arxiv.org/abs/2503.00200

[33] M. J. Kim, Y. Gao, T.-Y. Lin, Y.-C. Lin, Y. Ge, G. Lam, P. Liang, S. Song, M.-Y. Liu, C. Finn *et al.*, "Cosmos policy: Fine-tuning video models for visuomotor control and planning," *arXiv preprint arXiv:2601.16163*, 2026. [Online]. Available: https://arxiv.org/abs/2601.16163

[34] T. Yuan, Z. Dong, Y. Liu, and H. Zhao, "Fast-wam: Do world action models need test-time future imagination?" *arXiv preprint arXiv:2603.16666*, 2026. [Online]. Available: https://arxiv.org/abs/2603.16666

[35] Y. Xiao, J. Wang, N. Xue, N. Karaev, Y. Makarov, B. Kang, X. Zhu, H. Bao, Y. Shen, and X. Zhou, "SpatialTrackerV2: 3D point tracking made easy," *arXiv preprint arXiv:2507.12462*, 2025. [Online]. Available: https://arxiv.org/abs/2507.12462

[36] Team Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, D. Chen, F. Yu, H. Zhao, J. Yang *et al.*, "Wan: Open and advanced large-scale video generative models," *arXiv preprint arXiv:2503.20314*, 2025. [Online]. Available: https://arxiv.org/abs/2503.20314

[37] H. W. Chung, N. Constant, X. Garcia, A. Roberts, Y. Tay, S. Narang, and O. Firat, "UniMax: Fairer and more effective language sampling for large-scale multilingual pretraining," *arXiv preprint arXiv:2304.09151*, 2023. [Online]. Available: https://arxiv.org/abs/2304.09151

[38] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations (ICLR)*, 2022. [Online]. Available: https://openreview.net/forum?id=nZeVKeeFYf9

[39] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022. [Online]. Available: https://arxiv.org/abs/2210.02747

[40] J. Barreiros, A. Beaulieu, A. Bhat, R. Cory, E. Cousineau, H. Dai, C.-H. Fang, K. Hashimoto, M. Z. Irshad, M. Itkina *et al.*, "A careful examination of large behavior models for multitask dexterous manipulation," *Science Robotics*, vol. 11, no. 113, p. eaea6201, 2026.

[41] W. Zhao, L. Bai, Y. Rao, J. Zhou, and J. Lu, "UniPC: A unified predictor-corrector framework for fast sampling of diffusion models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 49 842–49 869, 2023.

[42] R. Punamiya, S. Kareer, Z. Liu, J. Citron, R.-Z. Qiu, X. Cai, A. Gavryushin, J. Chen, D. Liconti, L. Y. Zhu *et al.*, "EgoVerse: An egocentric human dataset for robot learning from around the world," *arXiv preprint arXiv:2604.07607*, 2026. [Online]. Available: https://arxiv.org/abs/2604.07607

[43] H. Fang, J. Duan, D. Clay, S. Wang, S. Liu, W. Huang, X. Fan, W.-C. Tsai, S. Chen, Y. R. Wang *et al.*, "MolmoAct2: Action reasoning models for real-world deployment," *arXiv preprint arXiv:2605.02881*, 2026. [Online]. Available: https://arxiv.org/abs/2605.02881

[44] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. Gu *et al.*, "Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation," *arXiv preprint arXiv:2506.18088*, 2025.

[45] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai *et al.*, "$\pi_{0.5}$: a vision-language-action model with open-world generalization," *arXiv preprint arXiv:2504.16054*, 2025. [Online]. Available: https://arxiv.org/abs/2504.16054

[46] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, vol. 44, no. 10-11, pp. 1684–1704, 2025.

APPENDIX I
IMPLEMENTATION DETAILS

A. Architecture

FloMo is initialized from the Wan 2.2 5B Text-Image-to-Video DiT [36]: a 30-layer transformer with hidden width d=3072, 24 attention heads (head dimension 128), feed-forward width 14,336, and a $(1, 2, 2)$ patchifier over the VAE latent grid. We use bidirectional (full) attention so every token attends to every other. On top of the shared backbone we add lightweight modality-specific heads:

- **Action head.** Each per-timestep action vector $\mathbf{a}_i \in \mathbb{R}^{d_a}$ (d_a=20) is mapped to a DiT token by a single linear projection $\mathbb{R}^{d_a} \to \mathbb{R}^d$ plus a learned per-position embedding; predicted action tokens are read back out by a single linear projection $\mathbb{R}^d \to \mathbb{R}^{d_a}$.
- **Video and scene-flow heads.** Video frames and RGB-rendered scene-flow frames are both encoded and decoded by the frozen Wan 2.2 VAE. On the trunk side they reuse the backbone's own patchify (Conv3d) and unpatchify (linear) projections rather than adding new layers, so these heads contribute no trainable parameters.
- **Conditioning.** The initial observation $\mathbf{o}_0$ is encoded by the frozen Wan VAE into $\mathbf{z}^o$, which forms a clean prefix in the token sequence that is never noised; the instruction $\boldsymbol{\tau}$ is encoded by the frozen umT5-XXL text encoder [37] (512 tokens, width 4096) and conditions the backbone through cross-attention. For the robot datasets the observation comprises three camera views (one front and two wrist), each VAE-encoded into its own prefix tokens; the human-video data uses a single view. Scene flow and video are predicted only in the front view.

We fine-tune the DiT with LoRA [38] (r=64, α=128, dropout 0), injecting adapters into the query/key/value/output projections of both self- and cross-attention and into both feed-forward linear layers of all 30 blocks; the Wan VAE and umT5 stay frozen. This leaves ≈161M trainable parameters (LoRA adapters plus the small action head) on top of the frozen 5B backbone.

B. Training Hyperparameters

Both stages use AdamW (β_1=0.95, β_2=0.999, weight decay 10^{-6}), gradient clipping at 1.0, bf16 mixed precision, and a constant learning rate. All inputs are processed at 256×256 resolution, giving a 16×16 Wan-VAE latent grid, and each example spans a 16-frame window: the model conditions on the initial frame and predicts the 16-step action chunk together with the video and scene flow over the window. Training uses a flow-matching objective with 1,000 timesteps. We augment every source except EgoVerse with random-resized-crop (scale $[0.85, 1.0]$; ratio $[0.98, 1.02]$) and color jitter (brightness, contrast, and saturation 0.25; hue 0.1). All models are fine-tuned from their respective epoch-28 mid-training checkpoints, using only in-house YAM data. Unless otherwise noted, fine-tuning evaluations use the epoch-3 checkpoint. Per-stage training settings are listed in Table II.

Table II: Per-stage training settings.

	Mid-training	**Fine-tuning**
Learning rate (constant)	2.8×10^{-4}	5.6×10^{-5}
Global batch size	512	256
Action loss weight λ_a	0.1	1.0
Optimizer steps	≈14,000	≈1,500

C. Inference

At test time, we draw the scene-flow and action tokens with a flow-matching sampler [39], integrating the learned velocity field with the UniPC ODE solver [41] under a shifted noise schedule (shift 5) for N=10 steps. We do not apply classifier-free guidance; the model is run on the conditional path only. We execute all 16 predicted actions per planning cycle, re-planning from the next observation. Each planning cycle (one joint motion-and-action denoising of N=10 steps through the full 5B trunk) takes ≈0.9 s on a single NVIDIA B200 GPU; asynchronous execution overlaps planning with robot motion.

D. Loss Weighting and Modality Masking

The training loss (Eq. 4) uses λ_f=1, λ_a=0.1, and λ_v=0 by default; the video term (λ_v=1) is enabled only for video-prediction ablations. During fine-tuning (Stage 2) we raise the action weight to λ_a=1. Per-sample modality masking sets λ_a=0 for action-free samples (human video), allowing co-training across datasets with mismatched modality coverage.

APPENDIX II
DATASET DETAILS

A. Source Datasets and Filtering

Below we describe each source dataset and how it is filtered to the subset used in training; Table III summarizes the result.

EgoVerse. EgoVerse [42] is a large egocentric human-manipulation video dataset; from it we use an 828 h subset collected by Mecka AI. From this we build a task-balanced subset by round-robin over tasks (one clip per task, then a second clip per task, and so on), giving a 200 h manifest of 8,578 clips over 3,696 tasks. We hold out 9 environments (35 tasks) for validation. From the training split we randomly subsample 16-frame windows down to a fixed budget, shuffling first so the kept windows cover all tasks rather than clustering on a few. Mid-training uses this subset, which spans ≈81 h of unique footage; no motion filtering is applied.

MolmoAct2-BimanualYAM. MolmoAct2-BimanualYAM [43] is a bimanual-YAM robot teleoperation dataset (737 recordings, three 640×360 views at 30 fps). We drop 7 "spell-out" tasks and round-robin across the remaining 25 coarse tasks to a task-balanced selection, of which ∼10 h (428 clips) is used for training.

In-house YAM. Teleoperated bimanual-YAM demonstrations of three tasks: put the corn in the bowl, open the oven and place the croissant on the plate, and open the toolbox and place the screwdriver inside. Each task includes randomized initial states as well as active data collection to enable data-efficient recovery from failure modes. This subset consists of

Table III: Dataset composition. "Tasks" counts distinct task labels (EgoVerse uses fine-grained activity categories, the others coarse manipulation tasks); "Clips" counts clips, videos, or demonstrations (one recording each); "Hours" is the span of unique footage used in training: the number of distinct frames appearing in any 16-frame window, divided by the source frame rate. All figures are for the subset actually used in each split.

	Train			Validation		
Dataset	**Tasks**	**Hours**	**Clips**	**Tasks**	**Hours**	**Clips**
EgoVerse	3,660	81.0	4,303	35	1.0	46
MolmoAct2-BimanualYAM	25	10.0	428	22	1.0	46
In-house YAM	4	3.7	1,029	4	0.4	113
In-house Human Data	11	0.7	710	11	0.1	83
In-house Play YAM	162	0.4	173	19	0.05	19

Table IV: Dataset sampling weights across training stages.

Dataset	**Mid-training weight**	**Fine-tuning weight**
EgoVerse	0.65	0.00
MolmoAct2-BimanualYAM	0.125	0.00
In-house YAM	0.125	0.75
In-house Human Data	0.05	0.00
In-house Play YAM	0.05	0.25

1,029 demonstrations totaling ≈3.7 h. Before training, data containing motion above or below pre-defined thresholds is filtered out.

In-house Human Data. Action-free human videos of the In-house YAM tasks, recorded from a single front-facing camera without action labels, so they supervise only the video and scene-flow heads. They cover the three robot tasks plus related variations (asparagus or pepper placed in a bowl or oven, pliers into the toolbox), for 710 demonstrations (≈0.7 h) across 12 splits.

In-house Play YAM. A small set of unstructured play demonstrations on the YAM robot (173 demonstrations, ≈0.4 h), filtered like the In-house YAM data and pooled as an auxiliary set during fine-tuning (Table IV).

B. Scene-Flow Extraction

We run SpatialTrackerV2 [35] on T=16-frame windows with Q=4096 query points uniformly arranged at t=0. We transform tracks into the current camera frame using estimated camera poses, then compute cumulative displacement (Eq. 2). Extraction runs at roughly 2.4 seconds per 16-frame window on a single NVIDIA L40S GPU, or about 72 GPU-hours per hour of 30 FPS video.

C. Sampling Weights

Table IV lists the weights that set how often each dataset is sampled in each training stage. Mid-training draws most samples from EgoVerse (0.65) to build a broad motion prior. It splits 0.25 evenly between the two robot datasets, so action learning is supervised throughout, and gives the small in-house human and play sets 0.05 each. Fine-tuning uses only action-labeled robot data: in-house YAM demonstrations (0.75) and in-house play (0.25).