Learning Long-term Motion Embeddings for Efficient Kinematics Generation

Nick Stracke[1,2,*] Kolja Bauer[1,2,*] Stefan Andreas Baumann[1,2]
Miguel Ángel Bautista[3] Josh Susskind[3] Björn Ommer[1,2]
[1] CompVis @ LMU [2] Munich Center for Machine Learning [3] Apple
compvis.github.io/long-term-motion

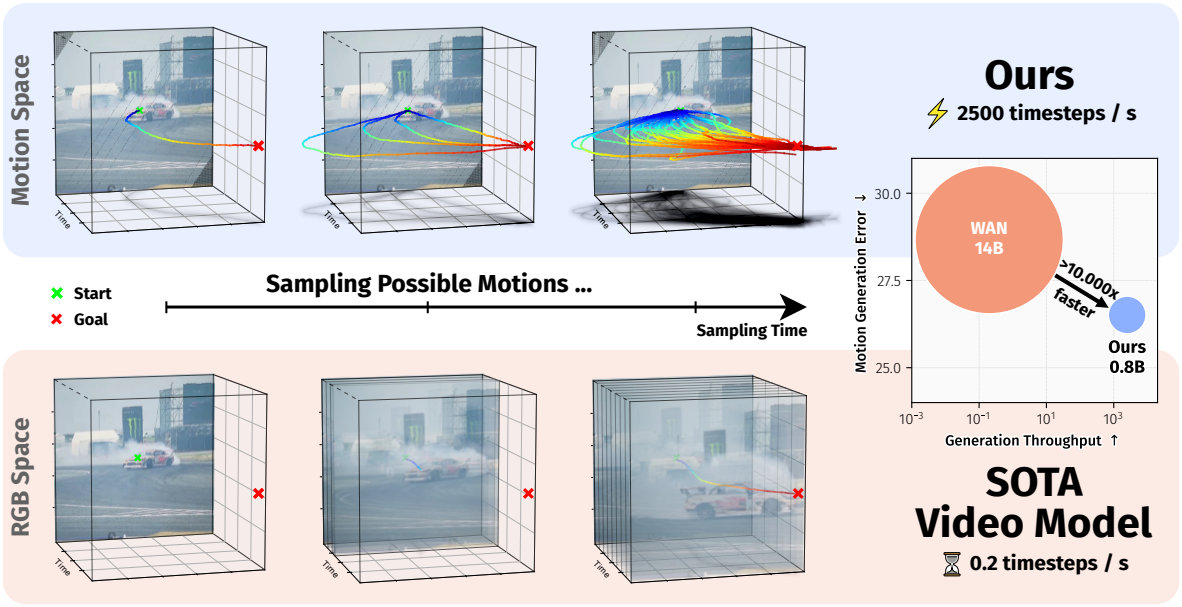


Figure 1. **Our approach enables extremely efficient, goal-conditioned kinematics generation and semantic motion reasoning.** We achieve this by learning a dense, temporally compressed motion space that allows goal-conditioned motion generation to be orders of magnitude faster than prior video models. While a video generative model has barely produced the first frame, our method can already generate multiple plausible motion trajectories connecting the start and goal, offering both speed and interpretability.

Abstract

Understanding and predicting motion is a fundamental component of visual intelligence. Although modern video models exhibit strong comprehension of scene dynamics, exploring multiple possible futures through full video synthesis remains prohibitively inefficient. We model scene dynamics orders of magnitude more efficiently by directly operating on a ***long-term motion embedding*** *that is learned from large-scale trajectories obtained from tracker models. This enables efficient generation of long, realistic motions that fulfill goals specified via text prompts or spatial pokes. To achieve this, we first learn a highly compressed motion embedding with a temporal compression factor of* $64\times$*. In this space, we train a conditional flow-matching model to generate motion latents conditioned on task descriptions. The resulting motion distributions outperform those of both state-of-the-art video models and specialized task-specific approaches.*

1. Introduction

Understanding and predicting motion is a fundamental trait of visual intelligence, yet current learning-based approaches with the goal of understanding and generating motion either focus on low-level motion information (optical flow or spatially sparse tracks) or conflate motion with appearance in video generative models, ultimately having to model per-pixel changes over time and requiring large amounts of compute to model this high-dimensional signal. In this paper, we propose to address these limitations by learning a long-term *motion embedding*: a compact, semantic latent

* Equal Contribution
Experimentation, including use of pre-trained models, was completed by university collaborators

representation that aggregates global kinematic structure, integrates information across trajectories, and models how motion evolves over extended time horizons. Unlike flow, which describes only instantaneous displacements, or tracks, which record sparse point correspondences, our embedding learns a continuous, scene-level representation of motion dynamics that generalizes beyond the observed samples and supports reasoning and generation in a unified latent space.

Our approach is rooted in two insights. First, learning useful motion representations requires more than tracking what moves: it demands reasoning about how things *can* move, how motions aggregate across objects, and forecasting what complex future behaviors are plausible in a given scene. Second, while direct modeling of video [13, 44, 49] or flow [30, 34] provides access to rich visual signals, these representations are high-dimensional and computationally intensive, and they cannot be strongly temporally compressed without significant information loss. In contrast, trajectories (tracks) are far more compact and interpretable, but lack the ability to generalize or aggregate motion contextually, and remain limited to the specific points sampled by the tracker.

To address these limitations, we introduce a two-stage framework that learns and predicts in a motion space for long-term kinematics. As the first step, we train a model to map sets of sparse tracker-derived motion samples into a continuous latent embedding. Unlike raw tracks, our latent motion space can be queried at any spatial position, enabling dense, context-aware motion predictions that go far beyond the tracker inputs. Secondly, we leverage the learned motion space for higher-order generative reasoning: a conditional flow matching model operates directly in the motion space, generating plausible, goal-directed motion under complex prompts such as text or spatial queries.

Our motion space thus sits at a new level of abstraction: longer than flow, richer than tracks, and far more efficient than video. It aggregates global kinematic context, supports reasoning about motion beyond observed samples, and provides a standardized interface for downstream models. By focusing on kinematics, i.e., how things *can* move, not just what happens frame-to-frame, we enable new forms of motion reasoning and generative modeling not possible with prior representations.

We demonstrate that our approach delivers substantial efficiency and reasoning gains across diverse motion tasks, outperforming raw tracks and enabling controllable motion generation with orders of magnitude greater compression than video models. Our framework thus establishes motion spaces as a powerful substrate for both large-scale motion understanding and semantic kinematics generation.

Contributions:

1. We propose a long-term motion embedding that captures the essential motion patterns of a scene from a single image, avoiding the overhead of full video generation.
2. We develop a goal-conditioned motion generator trained via flow matching on open-set data for efficient generative motion reasoning, operating entirely on the learned motion embedding.
3. We show that strong temporal compression improves learning, yielding better motion quality, more coherent dynamics, and faster training and inference.
4. Through extensive evaluations on open-set internet videos and robotics benchmarks, we show that our approach outperforms specialized trajectory predictors and video-based baselines while being substantially more efficient.

2. Related Work

Motion Representations Learning compact representations of motion has been studied from several perspectives, ranging from classical trajectory modeling to modern autoencoder-based approaches. Early works such as *An Uncertain Future* [43] explored stochastic VAEs for predicting future motion conditioned on a single image. More recent efforts adopt masked autoencoding or vector quantization to reconstruct trajectories or motion fields. Examples include trajectory autoencoders for anomaly detection or policy learning [2, 11], and two-stage models such as WHN [9] that encode dense grid trajectories and generate in latent space without temporal compression.

In contrast, our goal is to learn a *semantic and structured motion space* that captures generic kinematics to enable open-domain planning.

Motion in Video Models Most modern generative video models implicitly learn motion alongside appearance synthesis. Large-scale diffusion architectures such as Wan [44], Veo 3 [13] and others [1, 8, 17, 49] predict future frames without an explicit notion of motion. While such models excel at visual fidelity, the learned motion is entangled with texture and lighting, making kinematic reasoning and control difficult. Moreover, latent video autoencoders typically employ modest compression ratios (typically $4\times$–$8\times$), as stronger compression severely degrades visual detail. These modest compression ratios imply that large compute budgets are often needed to learn useful video generative models. Recent works have also explored world modeling in pre-trained feature spaces rather than RGB pixels, for example, by predicting the temporal evolution of DINO [31] features [3, 22, 54]. These approaches demonstrate that dynamics can be learned efficiently in a latent representation space, yet they do not provide an explicit notion of motion or kinematic structure. Their dense feature dynamics remain entangled with appearance and lack the interpretability and controllability of explicit trajectory-based motion representations. In contrast, our method models motion directly

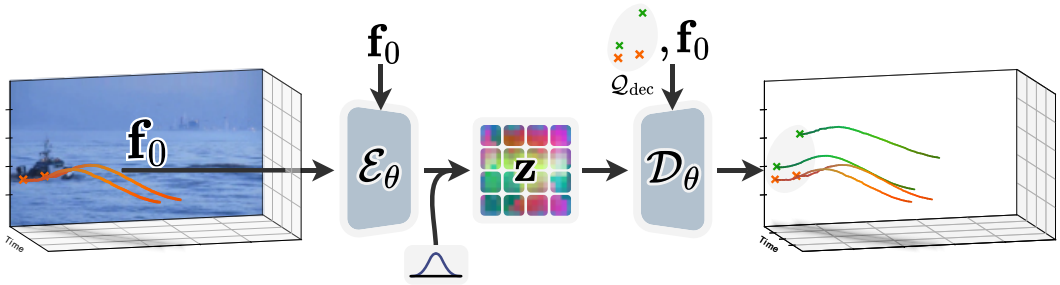


Figure 2. **Our approach to learn a dense motion space.** Sparse tracker trajectories and the start frame are encoded into a latent motion grid, which enables dense reconstruction at arbitrary spatial query points. The model jointly attends over trajectory tokens and frame features, producing temporally consistent, spatially dense motion predictions.

through sparse trajectories in a dedicated latent space, yielding a compact and semantically structured representation of kinematics.

Motion-conditioned Video Generation A separate line of work explicitly models motion as an intermediate representation before video synthesis. Motion I2V [34], MoVideo [25], and MOFA [30] first generates optical flow or depth maps that guide subsequent frame generation. VideoJam [10] combines motion and appearance branches through joint diffusion, while dragging-based methods [7, 15, 35, 36, 47] manipulate motion vectors interactively to steer local frame changes, provide intuitive spatial or semantic control over motion dynamics. Although these models explicitly handle motion, it remains defined at the pixel level and tied to visual rendering. In contrast, our framework models motion itself as the generative domain by learning a dedicated latent motion space, which can later drive downstream synthesis or reasoning tasks without relying on pixel data.

Goal-Conditioned Motion Generation Controllable motion generation has received increasing attention in both vision and robotics. Trajectory-based models such as ATM [46], Traj-MoE [48], Track2Act [6], and Amplify [11] predict trajectories conditioned on high-level cues like start frames or language, often for action policy learning. [39] uses a diffusion model to predict trajectories for animals based on a start frame and a brief motion history. Vision-Language-Action (VLA) [24, 51, 52] use trajectory prediction as a means of grounding vision-language models in physical actions. Our method builds on this idea of controllable generation but operates fully within the latent motion space learned by our autoencoder. The second stage employs a conditional flow matching model that generates motion latents conditioned on text or spatial pokes, enabling efficient and interpretable motion synthesis. Unlike prior approaches that predict explicit trajectories for downstream control or learn motion implicitly through appearance, our model unifies both perspectives by learning a generative motion prior in latent space that is compact and semantically grounded.

3. Method

We aim to first learn a compact and dense latent representation of trajectories obtained from off-the-shelf trackers that we can later use to model and manipulate kinematics.

We represent a track as a sequence of x, y coordinates in a normalized grid, i.e. $x, y \in [-1, 1]$. We denote a single track as $\mathbf{x}_i = ([x_0, y_0], ..., [x_t, y_t], ..., [x_{T-1}, y_{T-1}])$, with t being the timestep, representing the motion of a tracked point across time within a normalized image coordinate system.

3.1. Learning Motion Spaces

Overview To compress trajectories into a compact latent representation, we train a variational autoencoder [23] $\mathcal{F}_\theta = (\mathcal{E}_\theta, \mathcal{D}_\theta)$ that maps sets of sparse, partially masked trajectories to a latent grid and reconstructs them from it. The encoder $\mathcal{E}_\theta$ processes a set of trajectories $\mathbf{X} = \{\mathbf{x}_0, \dots, \mathbf{x}_{N-1}\}$, where each trajectory $\mathbf{x}_i = (\mathbf{x}_{i,0}, \dots, \mathbf{x}_{i,T-1})$ contains spatial coordinates $\mathbf{x}_{i,t} = [x_{i,t}, y_{i,t}]$, together with an embedding of the first video frame $\mathbf{f_0}$ (e.g., a DINO [54] feature map). The encoder produces a latent representation

$$\mathcal{E}_\theta : \Big(\underbrace{\{(\mathbf{x}_{i,t})_{t\in\mathbf{t}}\}_{i\in\mathcal{I}}}_{\text{trajectories}}, \underbrace{\mathbf{f}_0}_{\text{frame}}\Big) \mapsto \underbrace{\mathbf{z}}_{\text{latent}}, \tag{1}$$

with $\mathbf{z} \in \mathbb{R}^{H\times W\times D}$ where H, W denote the spatial grid size and D the number of latent channels.

The latent grid acts as a compact *motion space*, summarizing the kinematic content of all trajectories in a video. The decoder $\mathcal{D}_\theta$ reconstructs future motion at arbitrary spatial locations and time indices through a set of query tokens,

$$\mathcal{D}_\theta : \Big(\underbrace{\{\mathbf{x}_{j,0}\}_{j\in\mathcal{J}}}_{\text{query points } \mathcal{Q}_{\text{dec}}}, \mathbf{z}, \mathbf{f}_0\Big) \mapsto \underbrace{\{(\hat{\mathbf{x}}_{j,t})_{t\in\mathbf{t}}\}_{j\in\mathcal{J}}}_{\text{trajectories}}, \tag{2}$$

where $\mathcal{Q}_{\text{dec}} = \{\mathbf{x}_{j,0}\}_{j\in\mathcal{J}}$ denotes query points in normalized coordinates. Importantly, query points are not restricted

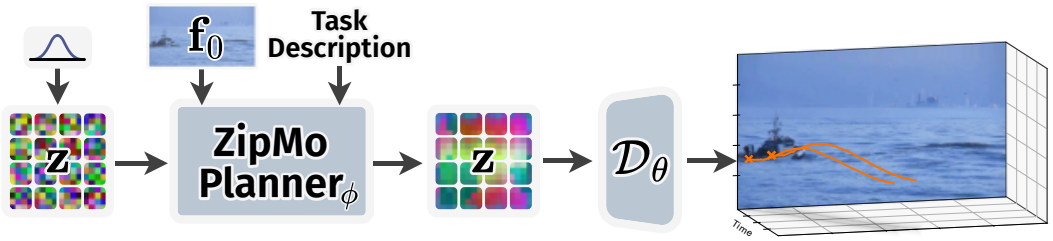


Figure 3. **Model architecture to generate in learned motion space.** We train a conditional flow matching model that learns a vector field over latent motion grids. We condition on either pokes [7] or text prompts, enabling controllable and semantically coherent motion synthesis in the learned motion space. The frame $\mathbf{f}_0$ provides context over the scene.

to those used during encoding. While the autoencoder is trained with sparse trajectories, the learned latent representation allows decoding motion at any position on the start frame, effectively enabling dense motion reconstruction.

Encoder Each individual trajectory sample $\mathbf{x}_{i,t}$ is Fourier-embedded using random frequencies drawn from a Gaussian distribution [38]. Temporal and trajectory identity information are encoded using a 3D rotary positional embedding (RoPE) [37]. It jointly encodes each sample's time index t and the start position $[x_{i,0}, y_{i,0}]$, which anchors the trajectory's identity. Token values and positional embeddings are thus computed as follows:

$$\mathrm{tok}(\mathbf{x}_{i,t}) = \mathrm{MLP}([\mathcal{F}(x_t) \mid \mathcal{F}(y_t)]) \quad \triangleright \text{ Encoding} \tag{3}$$

$$\mathrm{PE}(\mathbf{x}_{i,t}) = [\overbrace{\underbrace{\mathbf{R}(x_0)}_{d_k/4} \mid \underbrace{\mathbf{R}(y_0)}_{d_k/4}}^{\text{start position}} \mid \overbrace{\underbrace{\mathbf{R}(t)}_{d_k/4}}^{\text{time}} \mid \overbrace{\underbrace{\mathbf{1}}_{d_k/4}}^{\text{empty}}] \quad \triangleright \text{ RoPE} \tag{4}$$

The latent grid $\mathbf{z}$ is initialized as a learnable embedding broadcasted to the desired latent grid size $H \times W$. We differentiate between individual tokens of the latent grid using RoPE:

$$\mathrm{PE}(\mathbf{z}) = [\overbrace{\underbrace{\mathbf{R}(h)}_{d_k/4} \mid \underbrace{\mathbf{R}(w)}_{d_k/4}}^{\text{spatial position}} \mid \overbrace{\underbrace{\mathbf{1}}_{d_k/2}}^{\text{empty}}] \quad \triangleright \text{ RoPE} \tag{5}$$

We follow [4, 12] and only apply RoPE partially to encourage the model to rely more on semantic information instead of only positional information. In total, this yields $N \cdot T$ trajectory tokens jointly processed with the tokens of the latent grid $\mathbf{z}$. All tokens interact through global self-attention [42] with interleaved cross-attention to the start-frame features $\mathbf{f_0}$. The encoder outputs the mean and log-variance of a Gaussian posterior

$$q_\theta(\mathbf{z} \mid \mathbf{X}, \mathbf{f_0}) = \mathcal{N}\big(\boldsymbol{\mu}_\theta(\mathbf{X}, \mathbf{f_0}),\, \mathrm{diag}\big(\boldsymbol{\sigma}^2_\theta(\mathbf{X}, \mathbf{f_0})\big)\big)\,. \tag{6}$$

Decoder The decoder follows a masked-autoencoder (MAE) [18] design. Each query token encodes its temporal index t_q and start position $(x_{q,0}, y_{q,0})$ using RoPE similar to the encoder, and attends to the latent grid $\mathbf{z}$ and the start-frame features $\mathbf{f_0}$ via cross-attention. The decoded query tokens are projected to (x, y) coordinates through a small MLP, yielding motion predictions at the specified query positions.

Training follows a β-VAE objective [19] combining an L1 reconstruction loss and a KL regularization term:

$$\mathcal{L} = \underbrace{\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \big\|\mathcal{D}_\theta\big(\mathbf{x}_{i,0}; \mathbf{z}, \mathbf{f}_0\big) - \mathbf{x}_i\big\|_1}_{\text{reconstruction}} \tag{7}$$

$$+ \underbrace{\frac{1}{|\mathcal{J}_{\text{mae}}|}\sum_{j\in\mathcal{J}_{\text{mae}}} \big\|\mathcal{D}_\theta\big(\mathbf{x}_{j,0}; \mathbf{z}, \mathbf{f}_0\big) - \mathbf{x}_j\big\|_1}_{\text{masked reconstruction: } \mathcal{J}_{\text{mae}} \cap \mathcal{I} = \varnothing} \tag{8}$$

$$+ \underbrace{\beta\, D_{KL}[q_\theta(\mathbf{z} \mid \{(\mathbf{x}_{i,t})_{t\in\mathbf{t}}\}_{i\in\mathcal{I}}, \mathbf{f}_0) \parallel p(\mathbf{z})]}_{\text{latent regularization}} \tag{9}$$

where $\{\mathbf{x}_{i,0}\}_{i\in\mathcal{I}}$ are the query points corresponding to encoded trajectories (autoencoding loss) and $\{\mathbf{x}_{j,0}\}_{j\in\mathcal{J}_{\text{mae}}}$ are randomly held-out points not seen by the encoder (masked reconstruction loss). β controls the strength of the KL term.

This stage learns a semantic and structured motion bottleneck. The latent grid $\mathbf{z}$ captures the essential kinematics while enabling motion reconstruction at arbitrary spatial locations. Because trajectories are inherently low-dimensional and disentangled from appearance, the encoder achieves strong temporal compression without sacrificing semantic coherence. The resulting latent space provides a compact and interpretable *motion space*, forming the foundation for the subsequent flow-matching generation stage.

3.2. Reasoning in Motion Spaces

We train a conditional flow matching model that operates directly in the latent motion space learned by the first stage. The goal is to model a distribution over motion latents $\mathbf{z}$ that captures the diversity of plausible trajectories while allowing for flexible conditioning on, e.g., text prompts or pokes [7].

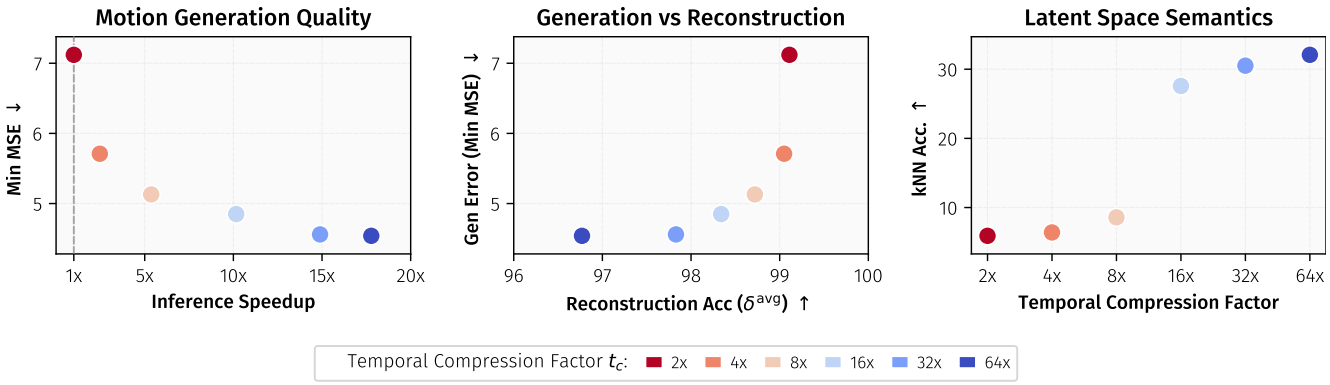


Figure 4. Temporal compression enables our model to generate plausible motions more efficiently. Under a fixed compute budget, both motion generation quality and inference throughput improve substantially with stronger compression (left), with only a minor reduction in reconstruction fidelity (middle). We attribute these gains to the reduced token count, which improves training efficiency, and to an increasingly semantic latent structure, as evidenced by higher kNN retrieval accuracy (right).

Given that the first stage already produces a compact and semantic motion embedding, learning to generate in this space becomes efficient.

Formally, we train a neural vector field $\mathbf{v}_\phi(\mathbf{z}_t, \mathbf{c}, t)$ to predict the instantaneous flow of a sample $\mathbf{z}_t$ along a continuous trajectory from a prior distribution $p_0(\mathbf{z}) = \mathcal{N}(0, I)$ to the empirical data distribution $p_1(\mathbf{z})$ of encoded motion latents. The conditioning variable $\mathbf{c}$ includes the start-frame embedding $\mathbf{f_0}$ and optionally additional control signals (language, pokes). The flow matching objective follows the continuous formulation of [26]:

$$\mathcal{L}_{\mathrm{FM}}(\phi) = \mathbb{E}_{t\sim\mathcal{U}(0,1)}\ \mathbb{E}_{\mathbf{z}_0,\mathbf{z}_1\sim p_0,\,p_1} \left[\|\mathbf{v}_\phi(\mathbf{z}_t, \mathbf{c}, t) - \mathbf{v}_t^*(\mathbf{z}_0, \mathbf{z}_1)\|_2^2\right], \tag{10}$$

where $\mathbf{z}_t = (1-t)\mathbf{z}_0 + t\mathbf{z}_1$ is the linear interpolation between noise and data, and $\mathbf{v}_t^* = \mathbf{z}_1 - \mathbf{z}_0$ is the target flow field driving samples toward the data manifold. In practice, $\mathbf{v}_\phi$ is implemented as a transformer-based denoiser that takes as input the noisy latent $\mathbf{z}_t$, the scalar timestep t, and conditioning features $\mathbf{c}$. Both the poke and the text conditional model use cross attention to integrate the conditioning signal. In the poke-conditional setting, we Fourier-embed the target poke positions and the target timestep, similar to Eq. (3). The pokes' start positions are encoded using RoPE (Eq. (4)). This design allows flexible conditioning on a variable number of pokes overall, as well as a variable number of pokes per trajectory, which can be placed arbitrarily throughout the temporal horizon.

4. Experiments

We evaluate our model's ability to compress motion and generate high-quality kinematics. First, we test our hypothesis that strong temporal compression enables efficient and accurate modeling of plausible motions. Second, we evaluate generative reasoning in this compressed motion space on both a closed-domain robotics setting and an open-domain general video setting. These experiments assess how well the model reasons about scene dynamics and plans trajectories that achieve goals specified through either text or poke.

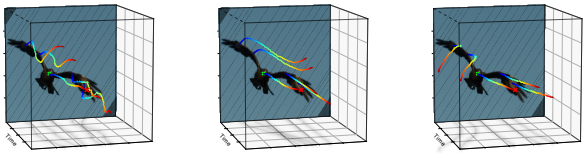


Figure 5. Example of multiple plausible motion hypotheses for the flight of an eagle, generated by our model from a single start frame. Our model produces diverse, physically coherent motions even in complex natural scenes, illustrating the expressiveness of the learned motion space.

Implementation Details The model used in all experiments has a latent grid with spatial dimensions of 16×16 and a temporal compression of $64\times$. Both VAE and Motion Planner are implemented as transformers, generally following LLaMA [40], with 340M and 530M parameters respectively. The goal conditioning is realized by cross-attending to either pokes for the open-domain setting or BERT [14] text embeddings, as is usual for LIBERO [46, 48]. For training, the models optimize their respective objective using AdamW [28] with $(0.9.0.95)$ betas. We provide additional implementation details of our model in Sec. B and Tab. G.

Datasets We consider both an open-domain and a closed-domain setting. For the open-domain setting, we train our models on KOALA-36M [45], aiming to capture broad world

Method	Modality	Speed	1 Poke			2 Pokes			4 Pokes			8 Pokes			Dense		
		timesteps/s ↑	Min ↓	Mean ↓	EPE ↓	Min ↓	Mean ↓	EPE ↓	Min ↓	Mean↓	EPE ↓	Min ↓	Mean ↓	EPE ↓	Min ↓	Mean ↓	EPE ↓
Motion-I2V [34]	Flow	21	135.7	429.7	19.7	106.5	336.2	17.1	87.4	245.7	14.4	54.5	121.7	11.2	46.9	71.7	8.8
Track2Act [6]	Tracks	180	-	-	-	-	-	-	-	-	-	-	-	-	138.7	156.1	20.9
ZipMo (Ours)	Latent	**2500**	**41.0**	**57.9**	**0.5**	**40.9**	**57.2**	**0.5**	**35.8**	**53.9**	**0.4**	**34.8**	**48.7**	**0.7**	**30.4**	**44.1**	**1.1**

Table 1. **Poked Motion Generation.** We compare against other methods that were trained on general video data and predict an explicit motion representation for multiple time steps. We report metrics for different conditioning densities to assess how well these models perform under varying levels of uncertainty. Track2Act [6] is end frame conditional, which is why we only report numbers for the dense case. Our approach outperforms other models while also being significantly faster.

knowledge and diverse motion patterns. The source video clips are up to eight seconds long, and we downsample them by skipping every other frame, resulting in frame rates between 12 and 15 fps. Training supervision is provided by pseudo ground-truth trajectories obtained using TapNext [53]. We filter out uncertain tracks as indicated by TapNext and train on tracks with 64 timesteps. We then evaluate on a subset of stock videos from Pexels [41], selected for scenes with a static camera that show complex motions and interactions, as introduced in [5]. See Sec. E for an ablation on tracker choice.

In the closed-domain setting, we train and evaluate on CoTracker3 [21] tracks obtained from the LIBERO benchmark [27], which focuses on structured robotic interaction scenarios. Together, these datasets allow us to assess how well the learned motion spaces generalize across domains and motion regimes.

Metrics We evaluate generated motions in terms of plausibility, diversity, and distributional fidelity. Our goal is to measure how well a model captures physically and semantically plausible motions in a scene. Since real-world data typically provides only a single ground-truth motion per video, evaluation must capture distributional quality despite the lack of a ground-truth distribution. To that end, we adopt metrics from prior work, primarily WHN [9], focusing on measuring fidelity, plausibility, and diversity.

To measure fidelity, we compute the minimum mean squared error (Min MSE), which measures the minimum distance between the ground truth (GT) motion and the distribution of generated samples. This metric correlates with the likelihood of the GT motion under the model's distribution, as high-likelihood regions are sampled more densely, reducing the minimum MSE. Motion diversity is measured with the mean squared error to the GT averaged over all samples (Mean MSE). A Mean MSE close to the Min can be seen as evidence for a collapsed motion distribution whereas a very high Mean MSE indicates potential implausible motion and outliers. Since our model is trained on open-set videos, the distribution of possible motions is highly multimodal, with many possible motions being plausible under the model. Metrics assuming unimodal (ie, deterministic) correspondence are therefore unsuitable. In cases where we condition models on pokes, we additionally measure conditioning adherence with the endpoint error (EPE). While no single metric independently guarantees meaningful motion generation, jointly considering Min MSE, Mean MSE, and EPE serves as a strong proxy for motion quality. All metrics are computed with tracks scaled to $[0, 128]$.

In the closed domain robotics setting, we measure the quality of generated motion embeddings by predicting a corresponding action sequence and executing it in simulation, allowing us to report success rates.

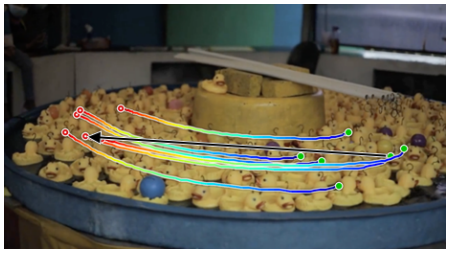
4.1. Semantic Motion Compression

We analyze the effectiveness of compressing motion across time using compression factors $t_c \in \{2, 4, 8, 16, 32, 64\}$. A compression factor of t_c means that t_c consecutive frames are aggregated into a single latent representation with no temporal dimension. Temporal compression enables the model to represent long-term scene dynamics more efficiently while reducing computational cost. We train models with varying temporal compression factors using a fixed compute budget. We evaluate the resulting motion generation quality in Fig. 4. Results clearly demonstrate that high temporal compression consistently improves motion generation quality while also significantly increasing inference efficiency. Increasing temporal compression also enhances the semantic structure of the motion space. This is reflected in a monotonically increasing kNN accuracy on a subset of SSv2 [16], shown in Fig. 4. Higher kNN accuracies indicate that semantically similar motions are grouped more closely in the latent space. We hypothesize that the improved motion generation quality under high temporal compression can be attributed to both the reduced token count, which increases training and inference efficiency, and a more semantically structured latent space. Additional results comparing performance over the course of training are provided in Fig. M, including two compute-matching schemes: matching either step time or batch size across models.

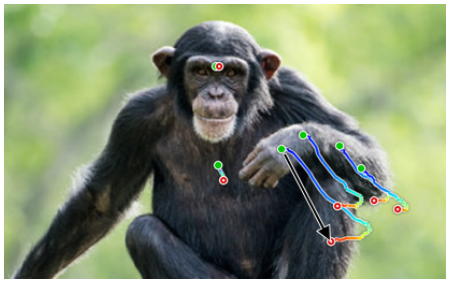
4.2. Motion Reasoning and Planning

We compare models in their ability to reason about physically plausible motions and plan trajectories that achieve desired goal states specified by prompts or end pokes.

(a) Path finding



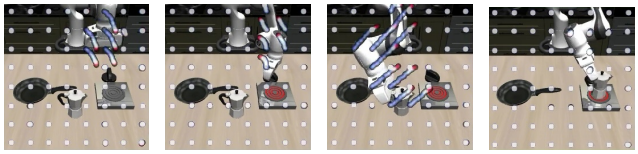
(b) Rotational Motion



(c) Joint Reasoning

Figure 6. Qualitative examples demonstrating diverse motion reasoning capabilities. These results highlight that the model captures semantic motion structure and generalizes to varied motion regimes.

(a) *Turn on the stove and put the moka pot on it*



(b) *Put the yellow-and-white mug in the microwave and close it*

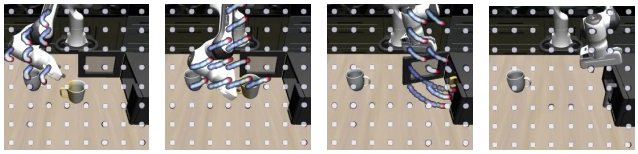


Figure 7. **LIBERO rollout samples.** Our track predictor forecasts tracks 16 steps ahead (visualized), enabling long-horizon planning. A policy head conditions on these predictions to select the next actions, with predictions updated after every new observation.

Model	LIBERO succ. rates ↑					
	10	90	Spatial	Goal	Object	Avg.
ATM [46]	39.3	48.4	68.5	**77.8**	68.0	60.4
Amplify [11]	62.0	**66.0**	69.0	75.0	85.0	71.4
ZipMo (Ours)	**66.3**	62.3	**91.3**	69.7	**98.0**	**77.5**
Tra-MoE [48]	28.5	–	62.5	81.0	73.5	61.4
ZipMo (Ours)	**61.0**	–	**85.7**	**82.3**	**92.0**	**80.3**

Table 2. Success rates on LIBERO tasks following the respective training and evaluation setup of ATM [46] (top) and of Tra-MoE [48] (bottom).

4.2.1. Trajectory Prediction

To comprehensively assess motion reasoning and planning, we conduct experiments in both a closed-domain robotics setting and an open-domain video setting.

In the open-domain setting, we investigate whether models trained on large-scale, open-set data have acquired transferable knowledge about how objects move and interact in the world. This evaluation tests each model's ability to generate physically plausible, goal-conditioned motions across diverse, unconstrained scenes. We compare models that directly predict motion, either as trajectories or optical flow. Evaluation is performed in a conditional setup, where the model is given a start frame and either a set of n target pokes or an end frame. The task is to generate plausible motions that transform the initial state into an end state consistent with the conditioning signals. To assess how well models handle different levels of conditioning sparsity and therefore uncertainty, we vary the strength of the conditioning signal by using $n \in \{1, 2, 4, 8\}$ pokes, along with a dense setting that uses the strongest conditioning signal supported by each method. Sparse conditioning provides more flexibility but can lead to more implausible motions, whereas dense conditioning offers stronger guidance but increases the challenge of reconciling all conditionings into a consistent motion. As shown in Tab. 1, our model consistently outperforms both flow-based and trajectory-based baselines across all conditioning sparsities. Qualitative results in Figs. 5 and 6 further demonstrate that our approach produces diverse, coherent, and physically plausible real-world motions. We show further results on DAVIS [32] and PhysicsIQ [29] in Sec. G.

In the closed-domain setting, we evaluate our model's ability to generate trajectories from high-level text instructions on the LIBERO robotics dataset [27]. Given a start frame and task description, the model predicts how the robot arm and surrounding objects should move to achieve the described goal, effectively performing motion planning from visual and linguistic input. Task descriptions in LIBERO are abstract (see Fig. 7), requiring the model to reason about both the current scene state and the specified goal to produce physically plausible and goal-directed trajectories. Results and evaluation details are presented in Sec. A.

4.2.2. Action Prediction

The LIBERO robotics setting allows us to rigorously evaluate generated motion embeddings by predicting corresponding robot actions, executing them in simulation and measuring task success. Following ATM [46] and Tra-MoE [48], we train a policy head that predicts robot actions from our generated conditional motion embeddings. Our model maintains a rolling motion plan over a horizon of T steps, updated at each timestep with new observations. At timestep t, it predicts a motion embedding encoding future motion for $\tilde{t} \in [t, t+T]$ (see Fig. 7), from which the policy head predicts the action at $t+1$. After executing this action, the model replans over the next horizon given the new observation. Since the policy head receives only the generated motion embeddings as input, it learns a mapping from motion embeddings to actions, effectively acting as an *inverse dynamics module*. This clear split ensures that the actual task-conditional reasoning and planning is done by the motion planner, not the policy head. We follow the evaluation setup of both baselines and report results in Tab. 2. Our model significantly outperforms both baselines across tasks, indicating superior understanding of scene dynamics. Further details are provided in Sec. A.

4.3. Comparison to Generative Video Models

To assess the effectiveness of modeling motion directly in our learned trajectory space, as opposed to modeling individual frames, we compare against state-of-the-art generative video models that also aim to learn world dynamics. The key distinction is that video models must jointly synthesize both appearance and motion, while our approach focuses exclusively on kinematics. Since we are interested in goal-directed motion rather than arbitrary dynamics, we focus on video models that can be conditioned on both a start frame and an end frame. This setup aligns with our conditioning scheme, where the model receives a start-frame embedding and a motion goal specified by pokes.

A direct comparison between our method and video models is challenging because videos do not provide explicit motion trajectories or any other motion representations. To obtain a comparable representation, we track the generated videos using CoTracker3 [21] (see Sec. F for TapNext), producing estimated trajectories $\text{tracks}_{\text{gen}} \in \mathbb{R}^{N\times T\times 2}$. However, tracking generated videos can introduce several sources of error, such as losing correspondences over time, resulting in incomplete or inconsistent motion estimates. In contrast, our method directly generates motion embeddings, providing globally consistent motion information without the risk of tracker drift or loss.

All methods are evaluated on a subset of Pexels [41] videos, which contain diverse and visually distinct motions. While this subset was held-out for our models, we cannot guarantee that this was also the case for the video models. For each video, we extract ground-truth trajectories by initializing trackers on the start frame and tracking them across time until the end frame, retaining only moving tracks that remain visible throughout. This yields a ground-truth trajectory set $\text{tracks}_{\text{gt}} \in \mathbb{R}^{N\times T\times 2}$. Our model is conditioned on $N = 40$ pokes $\mathbf{p}_{0\to T} \in \mathbb{R}^{N\times 2}$, each specifying the intended displacement from the start to the end frame, while video baselines are conditioned on the same start frame and the ground-truth end frame.

Model	Time ↓	Min MSE ↓	Mean MSE ↓	EPE ↓
Wan [44]	1h	28.67	57.02	4.68
Veo 3 [13]	?	36.18	94.00	6.21
ZipMo (Ours)	**1s**	**27.08**	**39.53**	**1.17**

Table 3. **Samples Matched:** We sample $k = 8$ times from each model and track the generated videos to report our distributional metrics as well as conditioning adherence with EPE. This is an unfavorable setting for us, as our model is much smaller. Still, we outperform the video while being orders of magnitude faster.

Model	Samples ↑	Min MSE ↓	Mean MSE ↓	EPE ↓
Wan [44]	1	64.20	64.20	5.23
Veo 3 [13]	1	65.99	65.99	5.84
ZipMo (Ours)	**> 10k**	**21.29**	**40.33**	**1.17**

Table 4. **Time Matched:** Setup similar to Tab. 3 but now matching wall clock time for sampling. Our performance lead increases drastically due to the efficiency of our approach.

We report quantitative results using our proposed motion metrics. Because sampling from video diffusion models is computationally expensive, we consider two evaluation regimes: (i) *Sample Matched* where both methods generate the same number of samples, and (ii) *Time Matched* where the wall-clock time for sampling is held constant, based on the average time required to generate a single video sample from the baseline. In the *Sample Matched* regime, we outperform Wan, with a larger gap for Veo. We hypothesize that this is because Veo tends to generate very high magnitude motions, which yields high errors if the ground truth video has more moderate motion.

5. Conclusion

We have introduced a framework that enables highly efficient modeling of scene dynamics by operating on a learned motion embedding. By leveraging large-scale tracker-derived trajectories, our approach captures long-term kinematic structure while heavily compressing temporal information by $64\times$. Operating on these motion embeddings, our conditional flow matching model efficiently generates realistic, long-horizon motions that fulfill goals specified via text prompts or spatial pokes. We empirically show that heavy temporal compression aids learning kinematics and enables our model to reason efficiently over long time. Across multiple benchmarks, our generated motion distributions significantly outperform those of both modern video models

and task-specific approaches, demonstrating that our motion embeddings provide a compact, expressive, and flexible representation for modeling kinematics.

Acknowledgments

This project has been supported by Apple, the Horizon Europe project ELLIOT (GA No. 101214398), the project "GeniusRobot" (01IS24083) funded by the Federal Ministry of Research, Technology and Space (BMFTR), the BMWE ZIM-project (No. KK5785001LO4) "conIDitional LoRA", the German Federal Ministry for Economic Affairs and Energy within the project "NXT GEN AI METHODS - Generative Methoden für Perzeption, Prädiktion und Planung", and the bidt project KLIMA-MEMES. The authors gratefully acknowledge the Gauss Center for Supercomputing for providing compute through the NIC on JUWELS/JUPITER at JSC and the HPC resources supplied by the NHR@FAU Erlangen.

Further, we would like to thank Owen Vincent for continuous technical support.

References

[1] Open AI. Sora 2, 2025. 2

[2] Kelsey R Allen, Carl Doersch, Guangyao Zhou, Mohammed Suhail, Danny Driess, Ignacio Rocco, Yulia Rubanova, Thomas Kipf, Mehdi S. M. Sajjadi, Kevin Patrick Murphy, Joao Carreira, and Sjoerd van Steenkiste. Direct motion models for assessing generated videos. In *Forty-second International Conference on Machine Learning*, 2025. 2

[3] 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. 2

[4] Federico Barbero, Alex Vitvitskyi, Christos Perivolaropoulos, Razvan Pascanu, and Petar Veličković. Round and round we go! what makes rotary positional encodings useful? In *The Thirteenth International Conference on Learning Representations*, 2025. 4

[5] Stefan Andreas Baumann, Jannik Wiese, Tommaso Martorella, Mahdi M. Kalayeh, and Björn Ommer. Envisioning the future, one step at a time, 2026. 6

[6] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pages 306–324. Springer, 2024. 3, 6

[7] Andreas Blattmann, Timo Milbich, Michael Dorkenwald, and Björn Ommer. ipoke: Poking a still image for controlled stochastic video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14707–14717, 2021. 3, 4

[8] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets. *CoRR*, abs/2311.15127, 2023. 2

[9] Gabrijel Boduljak, Laurynas Karazija, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. What happens next? anticipating future motion by generating point trajectories. *arXiv preprint arXiv:2509.21592*, 2025. 2, 6, 12

[10] Hila Chefer, Uriel Singer, Amit Zohar, Yuval Kirstain, Adam Polyak, Yaniv Taigman, Lior Wolf, and Shelly Sheynin. VideoJAM: Joint appearance-motion representations for enhanced motion generation in video models. In *Forty-second International Conference on Machine Learning*, 2025. 3

[11] Jeremy A Collins, Loránd Cheng, Kunal Aneja, Albert Wilcox, Benjamin Joffe, and Animesh Garg. Amplify: Actionless motion priors for robot learning from videos. *arXiv preprint arXiv:2506.14198*, 2025. 2, 3, 7, 13

[12] Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In *Forty-first International Conference on Machine Learning*, 2024. 4, 13, 14

[13] Google DeepMind. Veo, 2025. 2, 8

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019. 5

[15] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, et al. Motion prompting: Controlling video generation with motion trajectories. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1–12, 2025. 3

[16] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017. 6

[17] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024. 2

[18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 4

[19] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. 4

[20] Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, dahai li, Zhiyuan Liu, and Maosong Sun. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling*, 2024. 13, 14

[21] Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6013–6022, 2025. 6, 8

[22] Efstathios Karypidis, Ioannis Kakogeorgiou, Spyros Gidaris, and Nikos Komodakis. Dino-foresight: Looking into the future with dino. *arXiv preprint arXiv:2412.11673*, 2024. 2

[23] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[24] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Boyang Li, Shuo Liu, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact: Action reasoning models that can reason in space. In *Workshop on Making Sense of Data in Robotics: Composition, Curation, and Interpretability at Scale at CoRL 2025*, 2025. 3

[25] Jingyun Liang, Yuchen Fan, Kai Zhang, Radu Timofte, Luc Van Gool, and Rakesh Ranjan. Movideo: Motion-aware video generation with diffusion model. In *European Conference on Computer Vision*, pages 56–74. Springer, 2024. 3

[26] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. 5

[27] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023. 6, 7, 12

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5, 13, 14

[29] Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? *arXiv preprint arXiv:2501.09038*, 2025. 7, 15

[30] Muyao Niu, Xiaodong Cun, Xintao Wang, Yong Zhang, Ying Shan, and Yinqiang Zheng. Mofa-video: Controllable image animation via generative motion field adaptions in frozen image-to-video diffusion model. In *European Conference on Computer Vision*, pages 111–128. Springer, 2024. 2, 3

[31] 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. 2, 13, 14

[32] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 7, 15

[33] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020. 14

[34] Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, et al. Motion-i2v: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2, 3, 6, 15

[35] Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent YF Tan, and Song Bai. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8839–8849, 2024. 3

[36] Joonghyuk Shin, Daehyeon Choi, and Jaesik Park. Instantdrag: Improving interactivity in drag-based image editing. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–10, 2024. 3

[37] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. 4, 13, 14

[38] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020. 4

[39] Neerja Thakkar, Shiry Ginosar, Jacob C Walker, Jitendra Malik, João Carreira, and Carl Doersch. Forecasting motion in the wild. *arXiv preprint arXiv:2604.01015*, 2026. 3

[40] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 5

[41] UmiMarch. Openvideo: Pexels-raw (720p) video dataset. `https://github.com/UmiMarch/OpenVideo`, 2025. Accessed: 2025-11-10. 6, 8, 12

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[43] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European conference on computer vision*, pages 835–851. Springer, 2016. 2

[44] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 2, 8, 12

[45] Qiuheng Wang, Yukai Shi, Jiarong Ou, Rui Chen, Ke Lin, Jiahao Wang, Boyuan Jiang, Haotian Yang, Mingwu Zheng, Xin

Tao, et al. Koala-36m: A large-scale video dataset improving consistency between fine-grained conditions and video content. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 8428–8437, 2025. 5

[46] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 3, 5, 7, 8, 12, 13

[47] Weijia Wu, Zhuang Li, Yuchao Gu, Rui Zhao, Yefei He, David Junhao Zhang, Mike Zheng Shou, Yan Li, Tingting Gao, and Di Zhang. Draganything: Motion control for anything using entity representation. In *European Conference on Computer Vision*, pages 331–348. Springer, 2024. 3

[48] Jiange Yang, Haoyi Zhu, Yating Wang, Gangshan Wu, Tong He, and Limin Wang. Tra-moe: Learning trajectory prediction model from multiple domains for adaptive policy conditioning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6960–6970, 2025. 3, 5, 7, 8, 12, 13

[49] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 2

[50] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019. 14

[51] Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025. 3

[52] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *The Thirteenth International Conference on Learning Representations*, 2025. 3

[53] Artem Zholus, Carl Doersch, Yi Yang, Skanda Koppula, Viorica Patraucean, Xu Owen He, Ignacio Rocco, Mehdi S. M. Sajjadi, Sarath Chandar, and Ross Goroshin. Tapnext: Tracking any point (tap) as next token prediction. *arXiv preprint arXiv:2504.05579*, 2025. 6, 13, 14

[54] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024. 2, 3

Learning Long-term Motion Embeddings for Efficient Kinematics Generation

Supplementary Material

A. Additional Evaluation Details

LIBERO Trajectory Prediction We compare the accuracy of our model's predicted motions against both discriminative and generative baselines in Tab. E. Discriminative methods output a single set of trajectories for a given start frame and text prompt, while generative methods output distributions of possible trajectories. Across all tasks, our model significantly outperforms both discriminative and generative baselines, demonstrating superior understanding of the scene dynamics and the task at hand. For the generative methods, we report the Min MSE metric from WHN [9] for $k = 8$ samples (*Min*) and for fair comparison with discriminative methods, also for $k = 1$ (*Single*).

We closely follow the evaluation protocol of previous methods [9, 46, 48]: We split the videos into temporal windows of length T. Per temporal window, we extract $n = 32$ trajectories that exceed a certain variance threshold, i.e., are non-static. Given the start frame of the temporal window and the n starting positions, models have to predict the trajectories for the following T frames. All existing methods use a temporal window size of $T = 16$ [9, 46, 48]. Since our model predicts 64 frames, we interpolate the baseline window of 16 frames up to 64 frames during training. At inference, we subsample our model's prediction from 64 frames down to 16 frames to match the evaluation protocol. Pseudo-GT tracks are obtained through the exact preprocessing pipeline used by baselines [9, 46, 48]. The MSE metrics are computed on a resolution of 128×128. We train a single model jointly on both task suites (LIBERO-90 and LIBERO-10) and both views (side and effector).

LIBERO Action Prediction We train a policy head, also sometimes referred to as *Inverse Dynamics Module*, to predict robot actions from our generated motion embeddings. In the LIBERO setting, actions are 7-dimensional vectors. We instantiate the policy head as a shallow 6-layer transformer with dimensionality 768. During policy head training and policy rollouts, we perform 10 sampling steps for the motion planner to generate the conditional motion embeddings. The policy head cross-attends only to these motion embeddings, thus we can be sure that the planning is actually done by the motion planner. Following ATM [46] and Tra-MoE [48], we train the policy head with a simple L2 regression loss, effectively performing Behavioural Cloning (BC). We train the policy head for 12 hours on 16 H200s. For both the ATM [46] and Tra-MoE [48] baselines, we also follow their respective data configuration for training the motion planner. Unlike ATM [46], we train a single joint motion planner across all LIBERO task suites, which further increases task diversity and planning complexity for our model. To increase robustness of the motion planner during policy rollouts, we lock the DINOv2 image encoder and apply noise augmentation to the start frame during training. Further, we condition the motion planner on the current episode timestep and randomly drop out the start frame as an additional incentive to follow the task conditioning. As LIBERO rollouts are non-deterministic due to both stochasticity in the simulation environment and the model predictions, we report mean and standard deviation across seeds in Tab. F.

Model	LIBERO-90		LIBERO-10	
	Side	Effector	Side	Effector
Discriminative				
ATM [46]	47.82	123.01	59.10	131.58
Tra-MoE [48]	39.77	116.47	50.37	131.75
Generative				
$\text{WHN}_{\text{Single}}$ [9]*	17.89	57.64	26.18	63.47
$\text{ZipMo}_{\text{Single}}$	**8.83**	**45.23**	**10.73**	**46.27**
WHN_{Min} [9]*	10.99	32.01	13.86	35.93
$\text{ZipMo}_{\text{Min}}$	**5.96**	**27.78**	**7.43**	**25.80**
WHN_{Mean} [9]*	18.32	60.47	26.71	66.35
$\text{ZipMo}_{\text{Mean}}$	**9.18**	**45.71**	**9.05**	**46.55**

Table E. Comparison on text-conditioned trajectory prediction on the LIBERO [27] dataset, measured by MSE and following the evaluation setup of [9]. Numbers marked with an asterisk * are taken directly from the original paper, as no official checkpoints or training code were released. All other results were reproduced using the official checkpoints. The upper part of the table reports regression-based methods, while the lower part compares generative methods. The $_{\text{Single}}$ metric is the MSE for the first sample. All distributional metrics are computed over $k = 8$ samples per trajectory.

Video Models We compare our approach against two state-of-the-art generative video models that offer start and end frame conditioning, making them suitable baselines for our goal-conditional motion generation task. For Wan [44], we use the `Wan-AI/Wan2.1-FLF2V-14B-720P-Diffusers` implementation from Hugging Face and run it locally. Since Veo 3 is a closed-source model, we obtain samples via the fal.ai API. Due to the high computational and monetary cost associated with sampling from these models, we focus our evaluation on a curated subset of the Pexels dataset [41],

Model	LIBERO success rates ↑					
	10	90	Spatial	Goal	Object	Avg.
ATM [46]	39.3	48.4	68.5	**77.8**	68.0	60.4
Amplify [11]	62.0	**66.0**	69.0	75.0	85.0	71.4
ZipMo (Ours)	**66.3 ± 4.0**	62.3 ± 0.4	**91.3 ± 0.6**	69.7 ± 2.5	**98.0 ± 2.6**	**77.5 ± 1.8**
Tra-MoE [48]	28.5	–	62.5	81.0	73.5	61.4
ZipMo (Ours)	**61.0 ± 1.7**	–	**85.7 ± 2.1**	**82.3 ± 3.5**	**92.0 ± 2.0**	**80.3 ± 1.4**

Table F. Same setup as Table 2, with seed-wise variability reported for our method. For each experiment, we evaluate three random seeds, with 10 rollouts per task per seed, and report mean success rate ± standard deviation.

selecting 68 videos that feature unconstrained, real-world motion diversity.

To ensure a meaningful evaluation of motion generation, we carefully selected videos in which the primary moving objects remain clearly visible throughout the entire sequence. Our curated set features a diverse range of scenes, spanning various object categories including humans engaged in different activities, animals, vehicles, natural landscapes, and urban environments. The number of moving objects varies across videos, encompassing both single-object and multi-object scenarios. We further ensured diversity in motion by including clips with a wide spectrum of motion magnitudes, ranging from subtle, fine-grained movements to pronounced, large-scale motions.

Each original Pexels video has a framerate between 24 and 30 fps. We subsample each video with a frame skip of 1, selecting a contiguous window of 64 frames, corresponding to roughly 4-5 seconds of video. To establish ground-truth motion, we randomly sample 1024 query points in the first frame of each video and track them forward throughout the sequence. From these, we randomly select 40 dynamic (i.e., non-static) tracks, which serve as our final ground-truth trajectories for evaluation.

For both video models, we condition generation on the first and last frame of each clip, aligning the setup with our poke-conditional evaluation. The Wan model produces 81 frames at 12 fps, while Veo 3 generates 96 frames at 24 fps. To account for stochasticity and fairly assess distributional metrics, we sample $K = 8$ generations per start–end frame pair, resulting in a total of $68 \cdot 8 = 544$ generated video samples per model.

We then track the same 40 query points in each generated video using the same tracking procedure as was used to establish the ground truth tracks. Since both models output more frames than our ground truth sequences, we downsample the resulting tracks to match the 64-frame ground truth length. Finally, we compute the evaluation metrics outlined in the experiments section, comparing the predicted tracks from the generated videos to the ground truth to assess the quality and diversity of motion generation.

81 frames is the minimum number of frames that this version of Wan can generate. Shorter videos are not possible.

B. Implementation Details

We train our model in two stages: a variational autoencoder (VAE) that compresses tracking data into a latent grid representation, and a flow matching model that generates these latent representations conditioned on task specifications. Table Tab. G summarizes the key hyperparameters for both stages. Both models are trained on 10M video clips from diverse open-set videos, using TAPNext [53] to extract 1024 randomly sampled tracking positions per clip, with poke coordinates normalized to the range [-1,1].

The VAE employs a dual-encoder architecture with 12 layers each for self-attention and cross-attention processing, where tracking tokens attend to image tokens via cross-attention. Image features are extracted using a pretrained DINOv2 ViT-B/14 encoder [31], which remains frozen during initial training and is unlocked later when scaling to larger batch sizes. Tracker points are first Fourier embedded and processed with an MLP before passing them to the transformer. The VAE uses 3D RoPE [12, 37] as positional encoding, compressing the tracking data into a 16×16 latent grid with 16 channels. We set the KL divergence weight to $\beta = 1.0 \times 10^{-7}$ to regularize the latent space and use an L1 reconstruction loss. During early experiments, we evaluated alternative reconstruction losses (L2 and Huber) and alternative prediction targets (auto-regressive deltas and offsets to the start location). Directly predicting absolute coordinates in our normalized space with an L1 loss consistently gave the best performance. The VAE is trained with AdamW [28] using a constant learning rate of 1.0×10^{-4} and stable decay (WSD) [20], gradually scaling the batch size from 64 to 256 over 800k steps.

The second stage motion planner is a 24-layer transformer with 1024-dimensional self-attention and cross-attention, trained to perform flow matching in the learned latent space. Unlike the VAE, this model processes both image tokens and latent grid tokens through shared self-attention layers, while task specifications (either Fourier-embedded pokes

Parameter	ZipMo VAE	ZipMo Planner
Dataset	Open-Set Videos	Open-Set Videos
Number of clips	10M	10M
Tracker	TAPNext [53]	TAPNext [53]
Tracker positions	1024 random	1024 random
Poke scale	$[-1, 1]$	$[-1, 1]$
Batch size	$64 \rightarrow 256$	$512 \rightarrow 2048$
Optimizer	AdamW [28]	AdamW [28]
Peak LR	1.0×10^{-4}	1.0×10^{-4}
LR schedule	constant with WSD [20]	constant
Betas	$(0.9, 0.95)$	$(0.9, 0.95)$
Warm-up steps	300	300
Total Steps	800k	700k
Precision	bfloat16	bfloat16
Total Parameters	340M	530M
GPUs	$16 \rightarrow 64$ Nvidia H200	$16 \rightarrow 64$ Nvidia H200
Training Time	3 days	3 days
Depth	12 & 12	24
SA width	768	1024
CA width	768	1024
Normalization	RMSNorm [50]	RMSNorm [50]
FFN expand factor	3	3
Activation	SwiGLU [33]	SwiGLU [33]
Positional encoding	3D RoPE [12, 37]	2D RoPE [12, 37]
Image size	224×224	224×224
Image encoder	DINOv2 ViT-B/14 [31]	DINOv2 ViT-B/14 [31]
Latent width	16	16
Latent size	16×16	16×16
KL loss β	1.0×10^{-7}	-

Table G. Hyperparameters for our first stage VAE and motion planner. Ablation models use the same parameters, but were only trained on 4 Nvidia H200 GPUs for 24 hours.

or text embeddings) are provided via cross-attention. The motion planner uses 2D RoPE for positional encoding, as it operates on the 2D latent grid output by the VAE. For ablation experiments, we also evaluate using 3D RoPE in the motion planner in case the latent grid has a temporal dimension. Training follows a similar schedule to the VAE, scaling from 512 to 2048 batch size over 700k steps with a constant learning rate. Both models use RMSNorm[50], SwiGLU activations [33], and an FFN expansion factor of 3, and are trained in bfloat16 mixed precision on 16 to 64 Nvidia H200 GPUs, each requiring approximately 3 days of training time.

We conduct two ablations on the model design. Fig. J demonstrates that explicitly arranging latent tokens in a grid structure leads to slightly better performance compared to treating them as an unstructured sequence. Fig. K shows the performance of the flow matching model across different numbers of function evaluations (NFEs), demonstrating that the model achieves decent performance even with as few as 10 sampling steps, making it practical for more time-sensitive applications. All ablation models use identical hyperparameters to the main models but are trained on only 4 Nvidia H200 GPUs for 24 hours to enable rapid experimentation.

C. Track Densification and Inpainting

Trajectories provide a sparse sampling of motion and do not capture every pixel in a video. However, some application require or benefit from a dense a dense motion representation. Our approach naturally supports track densification, as illustrated in Fig. L.

This capability arises from two key properties of our framework. First, the motion planner (Sec. 3.2) always generates a dense latent motion grid, regardless of how dense or sparse the poke conditioning is, due to our training setup. Second, our first-stage autoencoder (Sec. 3.1) can be queried at arbitrary spatial locations. Thus, by converting available tracks into pokes and conditioning the motion planner on these inputs, we can generate a dense latent grid and subsequently obtain densely inpainted tracks by querying the autoencoder at all pixel locations. This procedure enables our model to reconstruct dense motion fields from sparse tracker inputs, providing high-resolution inpainting of trajectories where needed.

D. Effects of Temporal Compression

We extend our analysis of the temporal compression factor's influence on generation performance from Fig. 4. To that end, we show motion generation performance over the course of training in two compute-matched settings in Fig. M: matching step time across models and matching batch size across models. As discussed in the main paper, the number of tokens to be denoised is inversely proportional to the temporal compression factor. To illustrate, the model with $t_c = 2$ denoises $32\times$ as many tokens as the model with $t_c = 64$, which drastically increases the compute per sample both during training and inference. Results clearly demonstrate that our strong temporal compression is beneficial for motion generation performance, while also being much more efficient at inference time (see Fig. 4).

E. Tracker Model Ablation

To directly assess whether the learned motion space inherits tracker-specific biases, we ablate both the supervision source and its quality. Since the first stage embeds trackers into a smooth space, we report averaged reconstruction PCKs (δ^{avg}). Training with TapNext versus CoTracker3 yields near-identical reconstruction accuracy, and models generalize well when evaluated with the other tracker (Tab. H), indicating limited sensitivity to tracker choice.

We also stress-test supervision quality by degrading tracks during training. Dropping out tracks or training only on non-occluded tracks leads to graceful degradation on an evaluation split that includes occlusions (Tab. I), indicating robustness to imperfect supervision. While we acknowledge that tracker quality theoretically upper-bounds

Train \ Eval	TapNext	CoTracker3
TapNext	96.8	97.0
CoTracker3	96.3	97.3

Table H. Cross-tracker reconstruction accuracy δ^{avg}.

Ablation	$\delta^{\text{avg}} \uparrow$
Ours	96.8
+ track dropout	94.0
+ filter occlusions	93.2

Table I. Reconstruction accuracy degradation under systematic tracker degradation during training.

Model	Min MSE ↓	Mean MSE ↓	EPE ↓
DAVIS [32]			
Motion I2V [34]	222.2	307.0	16.37
ZipMo (Ours)	**155.1**	**233.0**	**0.83**
PhysicsIQ [29]			
Motion I2V [34]	177.8	225.1	12.4
ZipMo (Ours)	**90.60**	**143.65**	**0.76**

Table J. Dense Track prediction results on Davis *(top)* and PhysicsIQ *(bottom)*. We condition both models on the maximum number of pokes from the start to the end frames.

performance, our method outperforms baselines on downstream tasks (Tab. 2) and exceeds motion prediction performance of models trained with flow (Tab. 1) or RGB supervision (Tabs. 3 and 4).

F. Why CoTracker for Video Evaluation

We use CoTracker3 to obtain tracks from the generated videos in our SOTA evaluation (Sec. 4.3) because TapNext can lose tracks, resulting in missing values. This makes an unbiased challenging as it is unclear how to deal with these missing tracks. For completeness, we rerun the evaluation with tracks obtain from TapNext where we inpainting missing track values with the last observed position. This worsens metrics for video models (Min MSE↓: Wan 29→37, Veo 3 36→48).

G. Additional Track Prediction Results

We focus on static scenes to avoid conflating detailed object motion since heavy camera motion would dominate the metrics.We further evaluate our model on the DAVIS 2017 dataset, comprising 150 videos with often significant camera motion, as well as the solid mechanics split of Physics IQ, focusing on physical understanding. In both settings, we outperform Motion I2V, our strongest direct competitor, in its strongest Dense setting (Tab. J).

H. Addional Qualitative Examples

We provide additional qualitative examples in Figs. H and I.

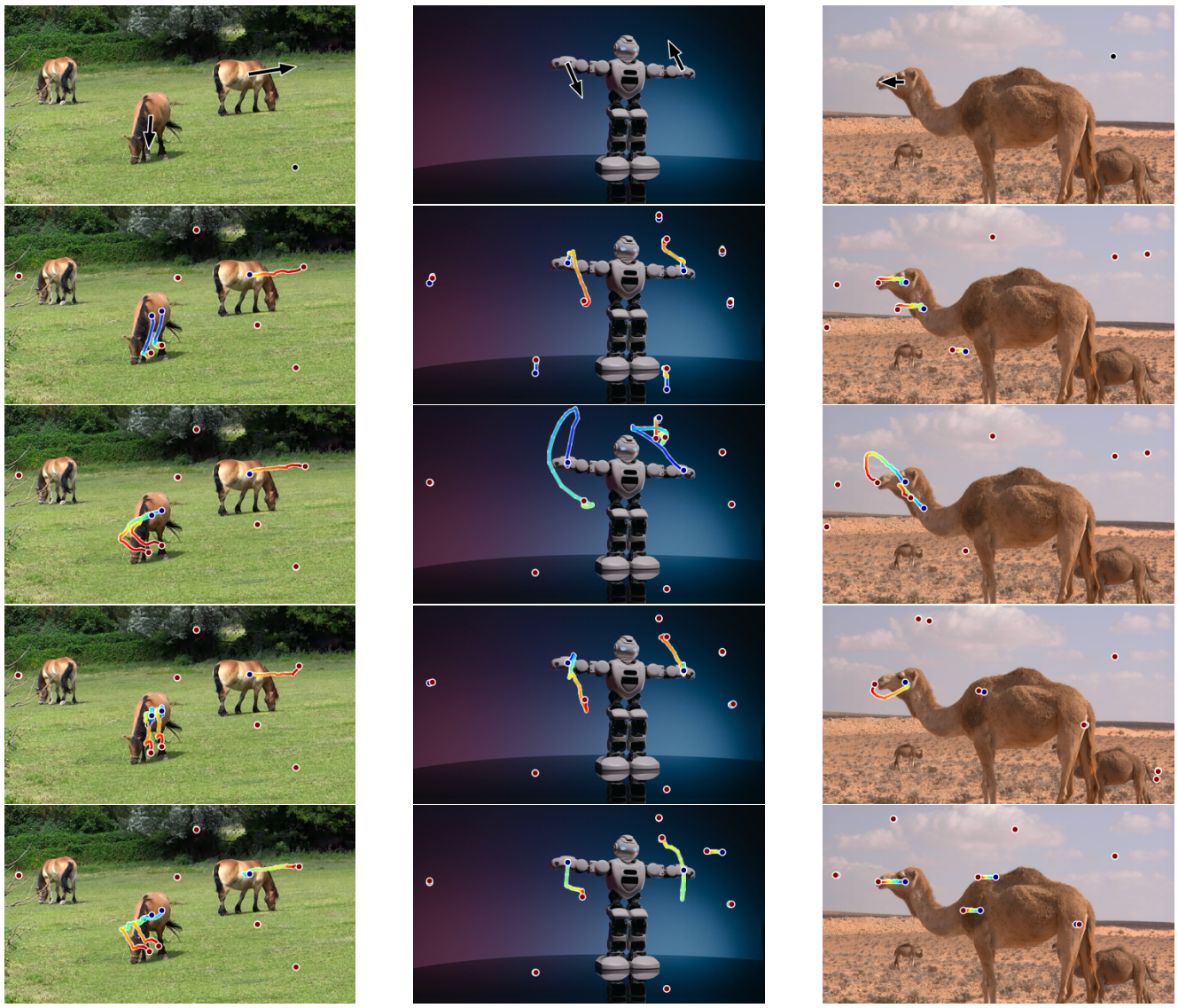
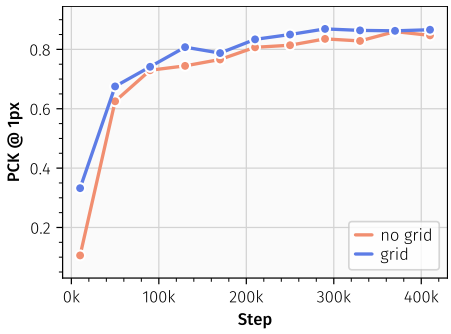
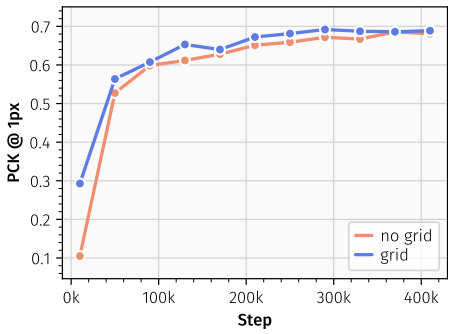


Figure H. **Additional qualitative samples.** The first image in every column is the prompt, with arrows indicating pokes, i.e., where a specific start position should end. The remaining four images are samples from our models, illustrating trajectories of randomly selected query points. The color gradient of the trajectories acts as an indicator of how fast the motion is happening. The gradient starts at blue, moves to green, yellow, and finally ends at red.

Figure I. **Additional qualitative samples.** The first image in every column is the prompt, with arrows indicating pokes, i.e., where a specific start position should end. The remaining four images are samples from our models, illustrating trajectories of randomly selected query points. The color gradient of the trajectories acts as an indicator of how fast the motion is happening. The gradient starts at blue, moves to green, yellow, and finally ends at red.



(a) AE PCK



(b) MAE PCK

Figure J. We ablate the effect of arranging latent tokens in a grid using 2D RoPE. We measure PCK using a 1 pixel threshold and report reconstruction performance for encoded positions (AE setting), as well as unseen positions (MAE setting). We find that providing positional information slightly increases performance.

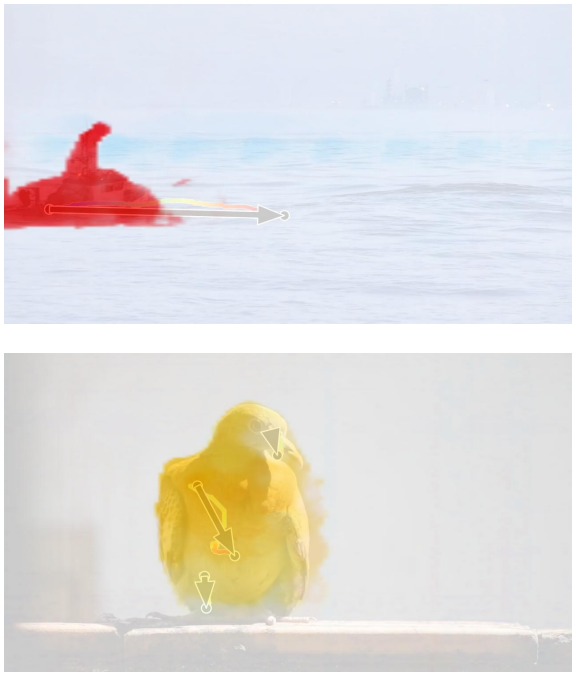


Figure L. Our model enables dense motion inference from sparse tracker inputs by converting observed tracks (pokes) into conditioning signals that specify desired endpoints in the final frame. The visualizations show dense flow toward the goal frame, illustrating how our approach infers globally coherent motion. Only the endpoint for each poke is provided; the resulting dense prediction demonstrates our model's ability to interpolate and inpaint plausible trajectories across the entire scene.

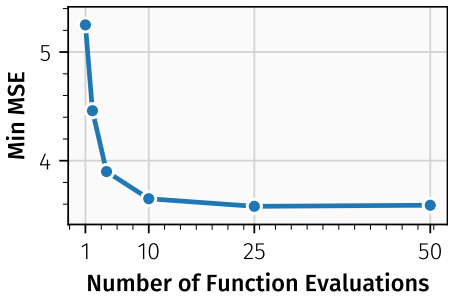
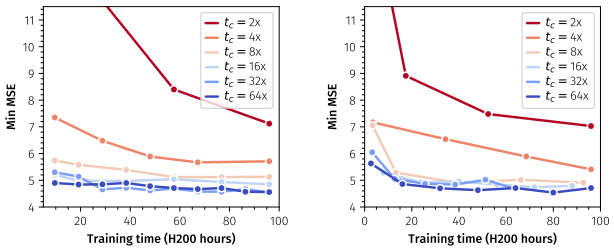


Figure K. Motion Generation performance (Min MSE) across different numbers of function evaluations (NFEs). Trajectories are densely conditioned on target positions, sampled $k = 128$ times per start frame.



(a) matched step time (b) matched batch size

Figure M. Motion generation quality over training for different temporal compression factors, measured against wall-clock time to account for varying computational cost. Matching step time (left) and batch size (right) both show that higher temporal compression yields more efficient learning and faster emergence of realistic motion.