JTT: RETHINKING POINT TRAJECTORY DIFFUSION FOR DIVERSE AND RIGID MOTION FROM AN IMAGE

Anonymous authors
Paper under double-blind review

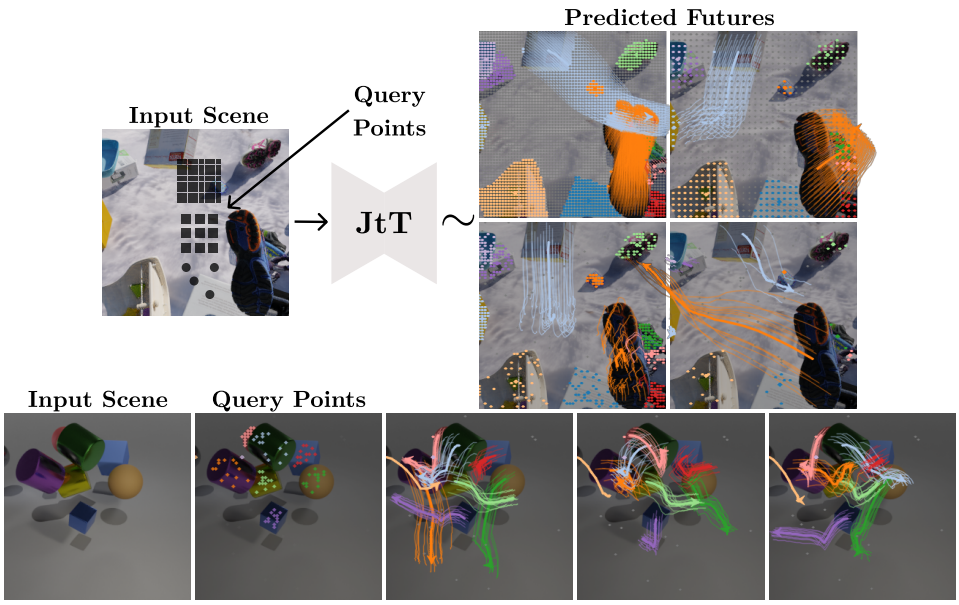

Figure 1: Our method, JTT, infers diverse, physically plausible motion for an arbitrary number of query points, supporting flexible query patterns across diverse camera viewpoints and scene settings.

ABSTRACT

We study forecasting rigid-body motion from a single image: generating diverse and physically plausible future trajectories for arbitrary points in a scene, without building rigidity into the architecture or relying on geometric inputs such as depth. Like video generators, recent trajectory generators fail at this task, even in simple synthetic Kubric scenes, despite scaling up model size and training on real data. We show how to solve this with the right architecture and parameterization. Our model, JTT, is a simple diffusion transformer that operates directly in coordinate space and decodes arbitrary and dense query patterns, without the trajectory VAEs or fixed grids assumed in prior work. Through a systematic study of the design space of point trajectory diffusion, we find that diffusing per-frame velocities is the only parameterization that captures rigidity, that it must be paired with the right timestep schedule, and that adding full-resolution RGB at the query points resolves the boundary failures of prior methods. However, evaluating such 2D models rigorously is itself non-trivial. To our knowledge, we are the first to observe that physical plausibility and diversity do not improve together during training: larger models, as used in prior work, memorize the single ground-truth future per training image, becoming more rigid while covering fewer possible futures. Yet prior work has paid little attention to this, so we show how model rankings depend on model size, metric, and stopping time, and adopt a stricter, checkpoint-averaged evaluation protocol. Under this protocol, JTT improves over prior methods on both distributional and rigidity metrics and approaches ground-truth rigidity, with only about 50M parameters.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

We study the problem of forecasting the motion of rigid objects from a single image. These predictors are increasingly used in robot manipulation to generate target motions for policies to execute, represented either as videos (Ko et al., 2023; Bharadhwaj et al., 2024a; Patel et al., 2025) or as point trajectories (Wen et al., 2024; Bharadhwaj et al., 2024b; Xu et al., 2024; Yuan et al., 2024). Motion forecasting is also used in model-based control (Yang et al., 2025; Mazzaglia et al., 2024) and controllable video generation (Polyak et al., 2024; Wan et al., 2025; Brooks et al., 2024).

Since an image does not fully determine the state of the scene, many futures are possible, and a useful predictor must capture their distribution rather than a single realization. The goal is to make such realizations physically plausible. Unfortunately, even the best generative forecasting models struggle with physical plausibility. Video generators often violate object permanence and basic physical laws (Kang et al., 2025; Motamed et al., 2025; Bansal et al., 2025; 2026); in particular, they may fail to predict rigid motion even when fine-tuned for forecasting in simple synthetic scenes (Boduljak et al., 2026). Some authors simply assume that objects are rigid and only predict rigid motion (Ko et al., 2023; Bharadhwaj et al., 2024b; Patel et al., 2025), but this is not general.

Using 2D point trajectories as the representation for motion forecasting has several advantages: it is general, it guarantees object permanence and temporal coherence by construction, it avoids the need to recover 2D motion from generated pixels (Ko et al., 2023; Patel et al., 2025), and it yields more rigid motion than videos under an identical architecture (Boduljak et al., 2026). Furthermore, as for video models, large quantities of real-world training data can be obtained by using reliable 2D point trackers (Doersch et al., 2023; 2024; Karaev et al., 2025) for pseudo-labeling videos (Bharadhwaj et al., 2024b; Zhang et al., 2026b; Thakkar et al., 2026; Lu et al., 2026).

However, point trajectories alone do not guarantee plausible motion: even on the synthetic Kubric-A scenes, which contain only a few rigid objects, no prior trajectory generator achieves exact rigidity (Table 2). Instead, they trade off plausibility and diversity: MotionForecasting (Thakkar et al., 2026) is the most rigid but has the worst distributional fidelity, whereas WHN (Boduljak et al., 2026) is the opposite. They also often miss small objects and object parts and generate jittery backgrounds (Fig. 4).

In this paper, we study in depth how to learn to predict diverse and plausible 2D point trajectories from single images. We deliberately avoid injecting hard inductive biases. We also avoid using additional observations of the scene, such as depth. We consider this to be a particularly interesting case for two reasons. First, it is a hard setting for the model: without additional observations, the distribution of possible futures is highly multimodal and difficult to capture even using diffusion (Lavoie et al., 2025). Second, 2D pseudo-labels can be obtained with reasonable accuracy by modern trackers (Doersch et al., 2023; 2024; Karaev et al., 2025), but 3D pseudo-labels remain much less accurate (Koppula et al., 2024; Xiao et al., 2024; 2025) and harder to get, which makes our case of practical interest.

We are not the first to consider this an interesting setting. However, we have identified two gaps in current best practices that we believe are limiting progress in this area. The first is a modeling gap and the second is an evaluation gap. Our contribution is to characterize and address them.

On the **modeling gap**, existing trajectory generators (Table 1) compromise on at least one of three quality axes: (i) *physical plausibility*, respecting physical invariants such as the rigidity of solid objects; (ii) *query flexibility*, querying any number of points at arbitrary 2D locations at inference; and (iii) *query density*, predicting motion everywhere in the image rather than at a sparse, fixed set of points. The default approach to improving quality is to scale the model and training data, or to introduce latent RGB and video representations, or custom VAE losses and rollout fine-tuning (Boduljak et al., 2026; Zhang et al., 2026b). Here we take a step back and ask whether a simple but well-designed diffusion model that operates directly in trajectory (coordinate) space can do better. We show it can, given four design choices that prior works treat inconsistently: (1) what to diffuse: absolute coordinates, offsets relative to the query, or velocities; (2) how to inject visual information: sampled image features or full-resolution RGB; (3) which timestep schedule to use for flow matching; and (4) which inductive biases help, in particular implicit (e.g., reconstruction model features) or explicit (e.g., world coordinates) 3D geometry. Most importantly, diffusing per-frame velocities is the only parameterization that captures rigidity, and adding full-resolution RGB at the query points resolves the boundary failures of prior methods, which use low-resolution image features (Fig. 4).

Building on these findings, we introduce the *Just trajectory Transformer* (JTT; Fig. 2), which predicts the per-frame velocities of arbitrary query points in an input frame. A small *support set* of query tokens does the heavy lifting, attending to each other and to multi-layer image features, while all other *non-support* queries decode their trajectories cheaply by cross-attending the support set. Unlike Perceiver IO (Jaegle et al., 2021; 2022), D4RT (Zhang et al., 2026a), and CoTracker3 (Karaev et al., 2025), which use latent or support tokens for *tracking*, we use them in a generative model and randomly re-partition the queries at every training step, so that a single model, trained with 1 to 1024 query points, decodes arbitrary and dense query patterns without retraining (Fig. 1) and lets users trade compute for accuracy. JTT requires no VAE, auxiliary losses, rollout fine-tuning, or pre-trained video latents; with only about 50M parameters, it outperforms prior methods on FVMD and rigidity.

On the **evaluation gap**, we note that assessment in prior works is often unreliable, partly due to reporting a single checkpoint when model ranking can change substantially from one to the next, and partly due to only reporting results on real videos, which, each providing a single realization, make it difficult to measure distributional coverage. In general, rigorous evaluation is non-trivial. For instance, rigidity cannot be measured directly from 2D point trajectories and must be assessed via proxy metrics. Furthermore, there is a clear trade-off between physical plausibility and distributional coverage.

By studying the training dynamics of trajectory generators in a controlled setting, to our knowledge for the first time, we find that larger models converge faster but then memorize the single ground-truth future per training image, becoming more rigid while covering fewer possible futures (Fig. 3), so that which model appears best depends on model size, metric, and stopping time. We therefore evaluate with a checkpoint-averaged protocol on the Kubric-A benchmark of Boduljak et al. (2026), whose multiple ground-truth futures per image enable rigorous evaluation of stochastic predictors; on real videos, a model that memorizes the single observed future (or predicts the implausible average) would even appear best.

We summarize our contributions as follows: (1) We introduce JTT, a transformer architecture and diffusion parameterization that learns the 2D motion of rigid bodies with neither rigidity built into the architecture nor extra geometric inputs, while decoding arbitrary and dense query patterns. It improves over prior methods on distributional and rigidity metrics, approaching ground-truth rigidity, while using a smaller model and simpler design than many prior works. (2) We systematically study the design space of point trajectory diffusion and show that diffusing per-frame velocities is the only parameterization that captures rigidity, that it depends on the right timestep schedule, and that full-resolution RGB at the query points resolves boundary failures; these choices should take precedence over more complex model architectures, such as introducing trajectory VAEs. (3) To our knowledge, we are the first to show that physical plausibility and diversity do not improve together during the training of point trajectory generators: larger models converge faster but then memorize the single ground-truth future per training image, improving rigidity while degrading distributional fidelity, so that the ranking of models depends on model size and on the evaluated checkpoint. We therefore adopt a checkpoint-averaged protocol for rigorous evaluation. (4) We show that explicit oracle geometry, such as inverse depth at query points or generation in world coordinates, improves performance, but, in terms of rigidity, our 2D-only model closes most of the gap between prior 2D methods and the 3D oracle, verifying our design; integrating 3D reconstruction models is promising future work.

2 Related work

Visual trajectory forecasting. Visual trajectory forecasting has progressed quickly, but existing methods have limitations that our work addresses. **Track2Act** (Bharadhwaj et al., 2024b) is a 600M-parameter diffusion model that predicts absolute coordinates for arbitrary query points, but it conditions on a single ResNet feature vector, which limits spatial resolution and causes it to hallucinate objects and miss small ones. **WHN** (Boduljak et al., 2026) runs latent rectified flow over a fixed 32×32 grid via a VAE with patch-level DINO conditioning, improving diversity and showing the importance of global reasoning over large point sets. However, the fixed grid prevents arbitrary query patterns, the VAE increases training complexity, and jitter persists in generated samples, even on Kubric-A, with degraded accuracy near boundaries. **TrajLoom** (Zhang et al., 2026b) extends WHN's latent flow matching to video-conditioned generation at a larger scale. It diffuses VAE

Table 1: **Design space of image-conditioned trajectory generators.** *Queries*: where trajectories can be requested at inference; *Dense*: decodes grids denser than in training; *Space*: where diffusion is performed; *Target*: the diffused trajectory representation; *: estimated (Section A.3).

	Flexibility		Representation			
Method	**Queries**	**Dense**	**Space**	**Target**	**Image conditioning**	**Parameters**
Track2Act	arbitrary	✗	coords.	absolute x_t	global vector	600M
WHN	fixed 32×32	✗	VAE latent	absolute x_t	DINOv2 patches	100/250M
TrajLoom	fixed 15×26	✗	VAE latent	offset $x_t - x_c$	Wan2.1 latents	250M
MotionForecasting	arbitrary	✗	coords.	velocity Δ_t	DINOv3 at query	100M
GPTF	arbitrary	✗	coords.	absolute x_t	DINOv3 patches	30/230M*
JTT (ours)	**arbitrary**	✓	**coords.**	**velocity Δ_t**	**RGB + DINO patches**	**50/220M**

latents on a fixed 15×26 grid conditioned on Wan2.1 (Wan et al., 2025) RGB-video latents, and introduces custom VAE losses along with rollout fine-tuning. **MotionForecasting** (Thakkar et al., 2026) adapts Track2Act to wildlife footage by removing the goal image and replacing the ResNet with DINOv3 (Siméoni et al., 2025a). The method samples low-resolution image features, resulting in failure modes similar to those of Track2Act. **GPTF** (Lu et al., 2026) unifies point tracking and forecasting in one generative model, diffusing absolute coordinates given DINOv3 patch features.

JTT differs from these methods in three ways. First, we operate directly in coordinate space without any VAE. Second, we condition on full-resolution RGB and multi-layer DINO features rather than low-resolution global features or RGB-video VAE latents. Third, we decouple the query distribution from the architecture via the support/non-support partition, so the same model can predict with arbitrary configurations of query points, including dense query patterns (Fig. 1).

Architectures supporting arbitrary configurations of query points. An important property of our design is support for arbitrary query configurations. Our design is inspired by Perceiver (Jaegle et al., 2021) and Perceiver IO (Jaegle et al., 2022), which introduce the idea of using a latent representation of fixed size paired with an arbitrary collection of output queries to read it out. This decouples the size of model inputs, internal state, and outputs; D4RT (Zhang et al., 2026a) applies this idea to dynamic 4D reconstruction with randomly resampled read-write tokens. Closer to our domain, in the context of 2D tracking, CoTracker3 (Karaev et al., 2025) (and its predecessor (Karaev et al., 2024a)) use a small set of *support* tracks that interact through self-attention and a large set of *non-support* tracks that read from them. Our support/non-support decomposition (Section 3.3) uses a similar idea for *generating* instead of estimating tracks. Unlike in prior work, the partition is randomized at every training step to keep the model general. Thus, a single model trained with 1 to 1,024 points per scene generalizes to dense query patterns at inference time.

3 METHOD

We study the problem of visual trajectory forecasting. Formally, a trajectory is a sequence $\boldsymbol{x} = (x_0, \ldots, x_T) \in \mathbb{R}^{(T+1)\times2}$ of 2D points $x_t = (u_t, v_t) \in \mathbb{R}^2$ that starts at a query location $q \equiv (u_0, v_0)$ (i.e., $x_0 = q$). The model is given as input an image $I \in \mathbb{R}^{H\times W\times3}$ as well as a collection of N query points $\mathcal{Q} = \{q_n\}_{n=1}^N$. The goal of the model is to sample corresponding trajectories from the conditional distribution $p(\{\boldsymbol{x}_n\}_{n=1}^N \mid I, \mathcal{Q})$. Our problem formulation is more flexible than some prior works; in particular, compared to Boduljak et al. (2026), the set of query points $\mathcal{Q}$ is supplied at inference rather than being fixed. On the other hand, unlike Bharadhwaj et al. (2024b), the model is only given the image I as evidence; it is not given a goal frame, segmentation, or correspondences.

We introduce a new model, JTT, to solve this problem. Our design, illustrated in Fig. 2, is (i) *generative*, sampling multiple distinct futures per scene; (ii) *flexible*, accepting an arbitrary number of query points at arbitrary 2D locations; and (iii) *dense*, scaling to thousands of points per scene on a single GPU. We take a deliberately minimal approach and train a single diffusion transformer that operates directly in coordinate space. In particular, we use no VAEs, auxiliary losses, or pre-trained video latents. While the design is simple, we find numerous choices critical to achieving good performance. We discuss these next.

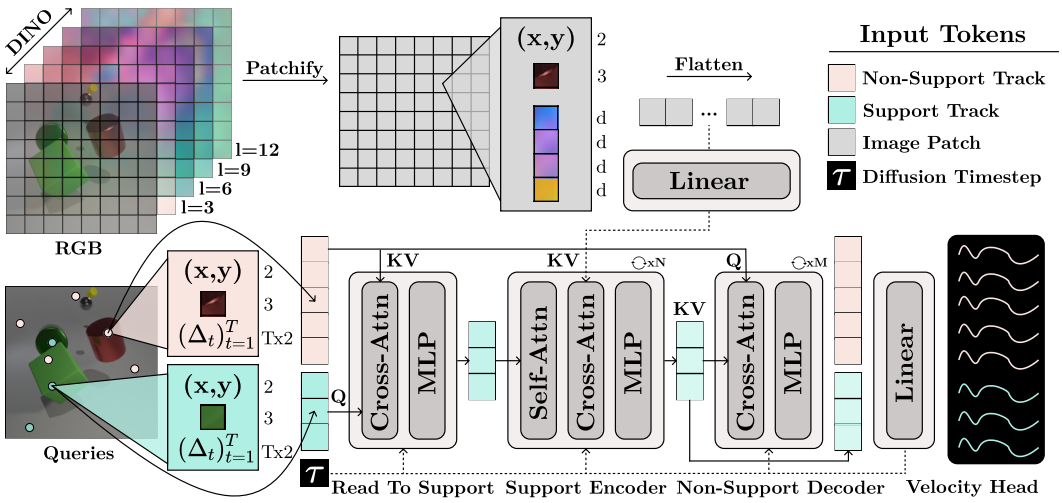

Figure 2: **Overview of JTT.** Given an input image and query points, image patch tokens (gray) are formed by concatenating patch position (x, y), RGB content, and multi-layer DINOv2 features ($l \in \{3, 6, 9, 12\}$), while query points are tokenized analogously with position, sampled RGB, and noised per-frame velocities $(\Delta_t)_{t=1}^T$, and then split into a support set (cyan) and a non-support set (pink). A *Read-To-Support* cross-attention block initializes support tokens from non-support tokens. The *Support Encoder* ($\times N$) refines the support set via interleaved self-attention and cross-attention to image patches. The *Non-Support Decoder* ($\times M$) reads out from the encoded support set in parallel; non-support tokens never attend to each other or back to the image, making dense readout linear in M and reusable for any query pattern at inference. A linear *Velocity Head* predicts the rectified-flow velocities, which are accumulated to recover trajectories. All blocks are AdaLN-modulated by the diffusion timestep τ.

3.1 Trajectory Parameterization

We parameterize each trajectory as the sequence of *instantaneous velocities* $\Delta_t \equiv x_t - x_{t-1} \in \mathbb{R}^2$, $t = 1, \ldots, T$ and target those for generation using denoising diffusion. We isotropically normalize the velocities across the dataset so the diffusion target has roughly unit variance. The full trajectory for query q is recovered at inference as $x_t = q + \sum_{i=1}^{t} \Delta_i$. Compared with *absolute coordinates* x_t (Track2Act, WHN), which couple the diffusion target to the query location, and *grid-cell offsets* $x_t - x_c$ (TrajLoom), which assume a fixed inference grid, per-frame velocities are translation-equivariant in the query, place the target on a near-zero, dataset-normalized scale, and decouple the trajectory from the absolute pixel grid. Empirically, we found that this helps the model learn the motion of rigid objects under arbitrary query patterns.

3.2 Image Conditioning

We patchify the image I into a regular $P \times P$ grid and form one token per patch by concatenating (i) the 2D pixel center $(u, v) \in \mathbb{R}^2$, (ii) the RGB patch itself (stacked to a $3P^2$ vector), and (iii) DINOv2 (Oquab et al., 2023) features extracted from *multiple* transformer layers $l \in \{3, 6, 9, 12\}$, each contributing a d-dimensional descriptor, analogous to the multi-layer aggregation in DPT (Ranftl et al., 2021). A single linear layer maps these vectors to a fixed hidden dimension, yielding P^2 image tokens that the trajectory tokens cross-attend to. We found that including full-resolution RGB patches instead of a single global vector (Track2Act) or only a low-resolution feature grid (MotionForecasting) works better, particularly at image boundaries, and reduces hallucinated and missed objects that sometimes affect prior work. We also observed that multi-layer DINO features perform better than relying solely on the final layer.

3.3 Architecture

The JTT model (Fig. 2) is a diffusion transformer that takes as input the trajectory velocity Δ_τ at a noise level τ, the image I, and the query set $\mathcal{Q}$, and outputs a denoising signal $\hat{\boldsymbol{v}}_\theta(\Delta_\tau, I, \mathcal{Q}, \tau)$ (see

Section 3.4). Hence, the transformer must process the velocities Δ encoded by the query tokens as well as the input image I and noise level τ.

The bulk of the computation is taken up by the query tokens, for which we use a Perceiver-IO-style (Jaegle et al., 2021; 2022) encoder-decoder. The idea is to split the N query points $\mathcal{Q}$ into a *support* set of S tokens (cyan in the figure) and a *non-support* set of $M = N - S$ tokens (pink). Support tokens attend to the image features and reason globally; non-support tokens are decoded cheaply by reading information off the support set in parallel. By comparison, self-attention across all query tokens would scale quadratically in their number N, which would make dense prediction (e.g., using a 256×256 grid of queries) all but intractable. CoTracker3 (Karaev et al., 2025) and D4RT (Zhang et al., 2026a) sidestep this problem by using a latent representation of fixed dimensionality, which acts as a bottleneck. We retain that efficiency, but make the support set fully flexible. Users can specify any set of queries $\mathcal{Q}$ and still control the compute budget by choosing the size $S \ll N$ of the support set, without having to retrain the model. The diffusion timestep τ modulates layer norms via AdaLN (Peebles & Xie, 2023) in every block.

3.4 Training and sampling

We train JTT with a rectified-flow (stochastic-interpolant) objective (Lipman et al., 2022; Liu et al., 2023b; Albergo et al., 2023). With ground-truth velocities $\Delta_1 \in \mathbb{R}^{T\times 2}$ (isotropically normalized as noted above) and $\Delta_0 \sim \mathcal{N}(0, I)$, the linear diffusion path is $\Delta_\tau = (1-\tau)\Delta_0 + \tau\Delta_1$ with constant diffusion velocity $\boldsymbol{v} = \Delta_1 - \Delta_0$ (not to be confused with the trajectory velocity). The model $\hat{\boldsymbol{v}}_\theta(\Delta_\tau, I, \mathcal{Q}, \tau)$ is trained by minimizing

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left\|\hat{\boldsymbol{v}}_\theta(\Delta_\tau, I, \mathcal{Q}, \tau) - (\Delta_1 - \Delta_0)\right\|_2^2\right],$$

with τ sampled from a logit-normal distribution (Esser et al., 2024) rather than uniformly: a uniform schedule overfits the single observed future per initial condition, while logit-normal up-weights mid-noise levels and yields sharper, more diverse samples (Table 3).

For each video, we take the initial frame, sample $N \sim \mathcal{U}[1, 1024]$ pixel queries, and use the next $T = 24$ frames as ground truth. For each N, tokens are randomly partitioned into a support set ($S \leq S_{\max}$) and a non-support set ($N - S$). This randomization is a strong inductive prior: trajectories must be consistent under arbitrary subsampling and orderings, and any token may be promoted to or demoted from the support role. We use no data augmentation, rollout fine-tuning, or auxiliary objectives. At inference, we draw $\Delta_0 \sim \mathcal{N}(0, I)$ and solve the SDE with the learned drift via 1000 Euler–Maruyama steps, then accumulate the predicted velocities. We use this sampler because it gives the best rigidity (LRTL); a 10-step Euler ODE sampler achieves similar rigidity and even better scores on the other metrics (Table 8). Details are in Section C.

4 Experiments

We begin by studying the interplay between model size, sample diversity, and physical plausibility during training, which determines how models should be selected and compared. We then show that JTT outperforms prior work, explain step by step which design choices lead to this performance, and finally compare it with oracles given privileged 3D geometry.

Data. We run most experiments on Kubric-A, strictly following the protocol of WHN (Boduljak et al., 2026): sequences of $T = 24$ frames rendered with the Kubric simulator (Greff et al., 2022), with multiple ground-truth futures per initial condition obtained by re-simulating the physics under different initial velocities. As established in WHN, this setup is necessary for rigorous evaluation of a stochastic motion predictor: real videos supply only one future per scene and no instance-level rigidity ground truth, ruling out both distributional metrics and physically grounded ones such as LRTL. We also use Kubric-D, with richer 3D scene geometry, to assess explicit geometric inputs. We verify our findings on diversity and rigidity qualitatively on real videos from Cityscapes (Cordts et al., 2016) and Physics 101 (Wu et al., 2016) (Sections E and F; videos in the supplement).

Metrics. We follow the evaluation protocol of Boduljak et al. (2026); lower is better for all metrics. Since motion prediction from a single image is multimodal, we report the *Best-of-K* MSE between K generated and K simulated futures per image. We measure distributional fidelity with *FVMD* (Liu

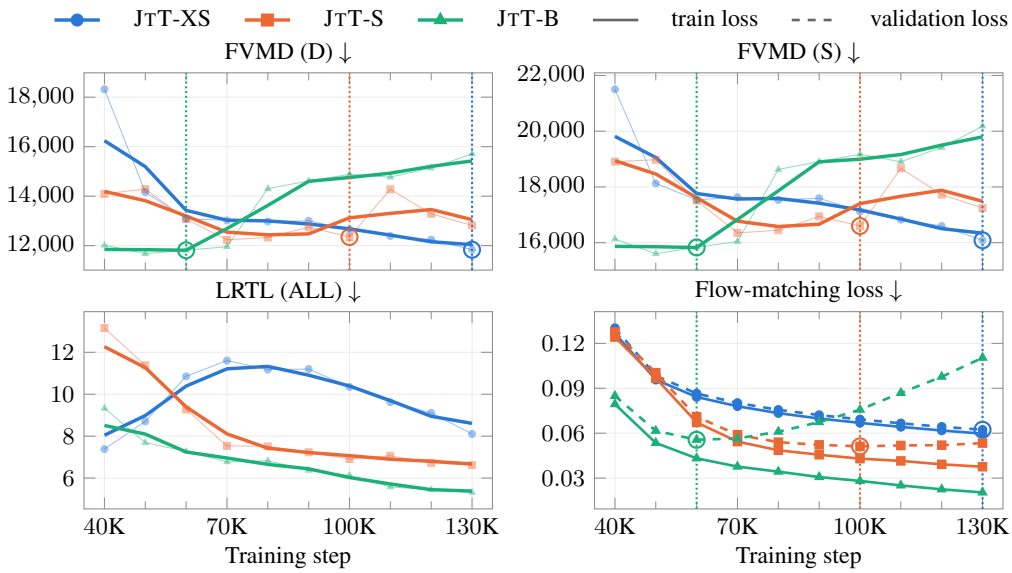

Figure 3: **Evolution of downstream metrics and flow-matching loss during baseline training** (Kubric-A; lower is better). Larger models attain their best FVMD early and then memorize, as their validation flow-matching loss (dashed) rises while the training loss (solid) keeps falling; dotted lines mark each model's lowest validation loss (thin: checkpoints; thick: 3-checkpoint moving average).

et al., 2024), a Fréchet distance between trajectory features, computed over the whole dataset (D) and per image (S). Directly evaluating rigidity is impossible for 2D outputs, since the image-plane projection of a rigid motion is not itself rigid; we therefore measure physical plausibility with the proxy metric *LRTL* (Karazija et al., 2024; Boduljak et al., 2026), the rank-5 reconstruction error of trajectories grouped by object using simulated masks, which is low for rigid motion. FVMD and LRTL are complementary: FVMD can miss local inconsistencies such as shuffled velocities, while LRTL is minimized by static trajectories.

Model size, flow-matching loss, and downstream metrics. Prior work on trajectory forecasting typically reports a single checkpoint, sometimes just for one model size (Bharadhwaj et al., 2024b; Thakkar et al., 2026), without analyzing how metrics evolve during training. In Fig. 3, we show how model size affects the flow-matching loss and downstream metrics over training for JTT-XS, -S, and -B (exact values in Table 9). Larger models converge faster but show clear signs of overfitting: their validation flow-matching loss (on the official Kubric-A validation split, disjoint from the test set; averaged over timesteps 0.1–0.9) rises while the training loss keeps falling; for B, it is lowest at 60K steps and nearly doubles by 130K, and S follows later and more mildly. The first effect of overfitting is that the model collapses to predicting a single mode for each input image, evidenced by an increase in FVMD, while LRTL (and Best-of-K, Fig. 6) keeps improving, with the LRTL of B eventually falling below that of the ground truth (5.31 vs. 6.68 at 130K). XS, in contrast, does not overfit: its training and validation losses stay close, and its FVMD decreases steadily. Overall, plausibility (LRTL) and diversity (FVMD) do not improve together, and rankings depend on when training stops: at 50K steps, B has the lowest FVMD (D) of the three models; at 130K, the highest.

Validation loss and model selection. The validation flow-matching loss is indicative of distributional fidelity: its minimum coincides with the best FVMD (D) and (S) for XS at 130K (and, for FVMD (S), also at 200K when trained twice as long; Table 10); it is one checkpoint later for B (60K vs. 50K), a 1.2–1.4% FVMD difference within the checkpoint-to-checkpoint noise of FVMD, whose moving average is indeed lowest at 60K; and it is later for S (100K vs. 70K), whose validation loss is nearly flat from 90K to 120K. This suggests that the validation loss, which needs one forward pass per sample, could serve as a cheap proxy for FVMD, which requires sampling many futures per image. To avoid single-checkpoint results, we evaluate an EMA checkpoint every 10K steps and report the average over the last five (60K–100K), together with the best among them, as Average (Best).

Comparison with prior work. Table 2 compares JTT, without (Sparse) and with the support architecture (Dense), with prior trajectory generators on the benchmark of WHN. Since checkpoint

Table 2: **Motion generation quality on Kubric-A, following WHN.** Prior methods: single checkpoint; JTT: Average (Best) of the last five checkpoints. JTT improves over prior methods on FVMD and LRTL, nearing the ground truth; GPTF is best on Best-of-K. †: reimplementation; *: estimated.

Model	Size	FVMD (D) ↓	FVMD (S) ↓	Best-of-K ↓	LRTL (ALL) ↓
GT	—	4,941	8,845	110.85	6.68
Track2Act (from WHN)	600M	16,735	22,509	250.80	15.80
WHN (L, latent-space)	250M	13,745	17,838	127.00	14.10
WHN (B, coordinate-space)	50M	14,519	18,551	137.80	23.70
MotionForecasting†	100M	20,178	26,379	186.47	8.20
GPTF	30M*	—	17,786	95.60	14.60
JTT					
S (Sparse)	50M	12,546 (12,238)	16,769 (16,351)	160.25 (129.77)	7.69 (6.91)
B (Sparse)	200M	13,514 (11,812)	17,716 (15,829)	124.37 (115.24)	6.67 (6.13)
S (Dense)	55M	11,807 (11,591)	15,695 (15,452)	135.85 (128.69)	6.84 (6.64)
B (Dense)	230M	13,266 (12,333)	17,437 (15,958)	130.44 (118.76)	6.80 (6.13)

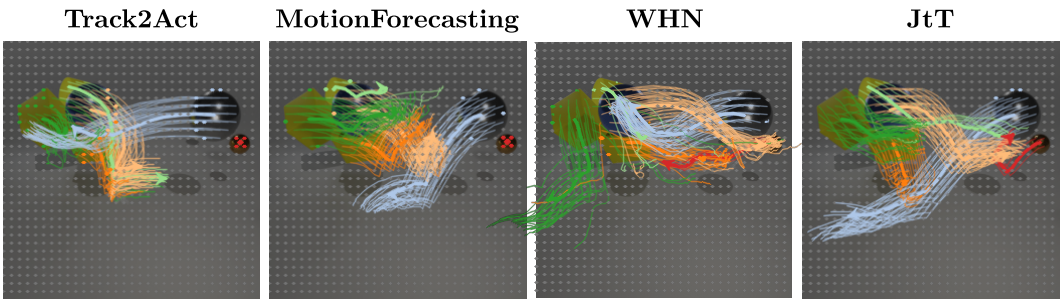

Figure 4: **Qualitative results on Kubric from WHN (Boduljak et al., 2026).** Baselines often miss smaller objects (red) or object parts (dark green), and frequently fail to capture boundary points accurately (peach), primarily due to their low-resolution conditioning. In contrast, our method reliably captures objects and their parts while achieving near-perfect boundary accuracy. This highlights our token design, which uses full-resolution RGB at the query points.

selection is crucial (Fig. 3), we report both the average over our last five checkpoints and the best among them (all in Table 9), whereas for prior methods we copy their single reported checkpoint; although this favors prior methods, every JTT variant improves over all of them on FVMD (D), FVMD (S), and rigidity, with LRTL close to the ground truth (6.67–7.69 vs. 6.68). GPTF obtains a better Best-of-K, but at the expense of rigidity (LRTL 14.60): its samples are closer to the ground truth pointwise, yet less physically plausible. Consistent with Table 3, prior methods that diffuse absolute coordinates (Track2Act, WHN, GPTF) obtain poor rigidity (LRTL 14.10–23.70), whereas MotionForecasting, the only one diffusing velocities, is the most rigid among them (8.20); JTT nevertheless outperforms MotionForecasting on all metrics without using the occlusion labels it requires, highlighting the benefit of full-resolution RGB at the query points.

Baseline details. We ablate one design choice at a time, starting from a baseline that corresponds to Fig. 2 with every query in the support set (Sparse): the Support Encoder refines all query tokens via self-attention and cross-attention to image patches, and the Velocity Head decodes them directly. It samples RGB at the query points and uses uniform time sampling (logit-normal in Table 2).

Parameterization and timestep sampling. With all variants normalized isotropically and the rest of the setup fixed, relative parameterizations almost always outperform the absolute one, but only velocities capture rigidity and surpass the previous state of the art on every best-case metric (Table 3). With uniform timestep sampling, however, relative parameterizations lose sample diversity; logit-normal sampling alleviates this, so we adopt velocities with logit-normal sampling throughout.

Image conditioning. Sampling RGB at full resolution at the query point substantially improves Best-of-K (182.73 → 138.27) and rigidity (LRTL 8.49 → 6.82) and resolves the boundary failures of prior methods (Fig. 4), whereas DINO features bilinearly sampled at the query point, as in MotionForecasting, degrade most metrics (Table 5 in the appendix). For the image tokens, adding

Table 3: **The impact of diffusion target parameterization and timestep scheduling.** Relative parameterizations outperform absolute coordinates; diffusing velocities with logit-normal timestep sampling gives the best overall trade-off between diversity and rigidity.

Model	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
$\tau \sim \mathrm{Uniform}[0, 1]$					
absolute	18,695 (15,750)	23,719 (20,772)	153.54 (149.67)	14.76 (11.65)	13.22 (11.56)
velocities	19,927 (17,785)	25,914 (23,518)	125.31 (122.19)	5.12 (4.81)	5.38 (5.17)
offsets	13,915 (13,164)	19,100 (18,245)	133.68 (124.97)	9.74 (8.61)	9.90 (8.93)
$\tau \sim \mathrm{LogitNormal}(0, 1)$					
absolute	20,284 (19,140)	24,678 (23,801)	125.15 (118.90)	25.59 (21.87)	19.25 (17.16)
velocities	14,136 (12,386)	18,523 (16,589)	138.27 (122.00)	6.82 (6.43)	6.35 (6.10)
offsets	13,883 (13,207)	17,814 (17,013)	145.56 (128.79)	15.91 (14.00)	13.82 (12.24)

Table 4: **Explicit oracle geometry helps.** Inverse depth or 3D world coordinates mainly improve Best-of-K, while in rigidity our 2D-only model closes most of the gap from prior 2D methods to the 3D oracle. Kubric-A/D are re-rendered with 2D–3D correspondence, so numbers differ from Table 2.

Model	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
Kubric-A					
2D	14,097 (12,080)	20,074 (17,790)	120.04 (111.50)	6.16 (5.75)	5.92 (5.70)
2D + Inverse Depth	13,946 (12,560)	19,827 (17,962)	92.51 (83.98)	5.59 (4.21)	5.97 (4.77)
3D	12,909 (12,162)	18,518 (17,641)	95.02 (92.37)	4.37 (3.86)	4.82 (4.38)
Kubric-D					
2D	19,210 (17,848)	28,412 (25,903)	94.16 (89.68)	2.05 (1.95)	1.69 (1.60)
2D + Inverse Depth	18,652 (17,223)	27,796 (25,746)	64.03 (56.80)	1.81 (1.55)	1.80 (1.59)
3D	19,500 (18,955)	29,409 (28,629)	76.77 (73.22)	1.95 (1.67)	1.93 (1.72)

register tokens (Darcet et al., 2024) to DINOv2 improves all metrics (e.g., Best-of-K 138.27 $\rightarrow$ 126.00), and multi-layer (DPT) features further improve Best-of-K and foreground rigidity; other extractors do not help consistently (Table 7), so we use RGB with DINOv2 layers $3, 6, 9, 12$.

Dense support architecture. Beyond enabling dense decoding, the support architecture improves FVMD and foreground rigidity for both S and B (details in Table 6 in the supp. mat.): for S, FVMD (D) 12,546 $\rightarrow$ 11,807 and LRTL (FG) 6.57 $\rightarrow$ 5.82; for B, 13,514 $\rightarrow$ 13,266 and 6.01 $\rightarrow$ 5.87. For S, it improves every metric; the slight Best-of-K and overall LRTL degradation for B mirrors the overfitting trends of Fig. 3 in reverse: we attribute it to the harder objective of decoding all queries through a randomly re-partitioned support set, which alleviates overfitting.

Does explicit 3D geometry help? For these experiments, we re-render Kubric-A and Kubric-D with direct point-to-point correspondence between 2D and 3D trajectories, so that the 2D and 3D variants are trained and evaluated on identical points; absolute numbers therefore differ slightly from Table 2. Oracle geometry from the simulator, as inverse depth at the query points or queries and trajectories in 3D world coordinates, helps mainly in Best-of-K (Table 4). In rigidity, however, our 2D-only model closes most of the gap between prior 2D methods and the 3D oracle (LRTL 6.16 vs. 4.37, compared to 8.20–23.70 for prior work; Table 2), verifying the effectiveness of our design.

5 Conclusion

We showed that diverse and plausible rigid-body motion can be forecast from a single image with JTT, a simple diffusion transformer operating directly in trajectory space, without trajectory VAEs, fixed grids, or rigidity priors, on synthetic multi-future data. Its key ingredients are diffusing per-frame velocities with logit-normal timestep sampling, full-resolution RGB at the query points, and a randomized support set for arbitrary and dense queries. As plausibility and diversity do not improve together during training, fair comparison requires checkpoint-averaged evaluation on such data, under which JTT outperforms prior methods on distributional and rigidity metrics. We hope our findings promote better evaluation practices in the field; we leave rigorous real-data evaluation to future work.

AI USE STATEMENT

In this work, we used generative AI tools to polish the paper, improving clarity and rephrasing text, and to identify additional relevant sources, which we read and thoroughly checked manually. We have *not* used generative AI tools in the implementation of our method, nor for research ideation, experiment design, or synthetic data generation, which were carried out by the authors. However, we *did* use generative AI to help write visualization code, for instance to output plots and animations for the paper and for debugging. We have reviewed all AI-assisted work; in particular, we checked the resulting plots and visualizations and ensured that they are correct and faithfully reflect our results. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. VideoPhy: Evaluating physical commonsense for video generation. *Proc. ICLR*, 2025.

Hritik Bansal, Clark Peng, Yonatan Bitton, Roman Goldenberg, Aditya Grover, and Kai-Wei Chang. VideoPhy-2: A challenging action-centric physical commonsense evaluation in video generation. In *Proc. ICLR*, 2026.

Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2Act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024a.

Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2Act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *Proc. ECCV*, 2024b.

Gabrijel Boduljak, Laurynas Karazija, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. What happens next? Anticipating future motion by generating point trajectories. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=t1vMYl1yhe.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023. URL https://arxiv.org/abs/2212.06817.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. Technical report, OpenAI, 2024.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. CVPR*, 2016.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for EPIC-KITCHENS-100. *International Journal of Computer Vision (IJCV)*, 130(2):331–355, 2022. doi: 10.1007/s11263-021-01531-2.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=2dnO3LLiJ1.

Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement, 2023. URL https://arxiv.org/abs/2306.08637.

Carl Doersch, Pauline Luc, Yi Yang, Dilara Gokay, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ignacio Rocco, Ross Goroshin, João Carreira, and Andrew Zisserman. BootsTAP: Bootstrapped training for tracking-any-point. In *Proc. ACCV*, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024. URL https://arxiv.org/abs/2403.03206.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *Proc. ICCV*, 2017.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: a scalable dataset generator. In *CVPR*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Proc. NeurIPS*, 2020.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention, 2021.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *Proc. ICLR*, 2022.

Bingyi Kang, Yang Yue, Rui Lu, Zhijie Lin, Yang Zhao, Kaixin Wang, Gao Huang, and Jiashi Feng. How far is video generation from world model: A physical law perspective. In *Proc. ICML*, 2025.

Nikita Karaev, Ignacio Rocco, Ben Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024a.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together, 2024b. URL https://arxiv.org/abs/2307.07635.

Nikita Karaev, Iurii 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 International Conference on Computer Vision (ICCV)*, 2025.

Laurynas Karazija, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Learning segmentation from point trajectories. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to act from actionless videos through dense correspondences, 2023. URL https://arxiv.org/abs/2310.08576.

Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, João Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. TAPVid-3D: a benchmark for tracking any point in 3D. *arXiv*, 2407.05921, 2024.

Samuel Lavoie, Michael Noukhovitch, and Aaron Courville. Compositional discrete latent code for high fidelity, productive diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=1iSnpztjbD.

Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Flow-grounded spatial-temporal video prediction from still images, 2018. URL https://arxiv.org/abs/1807.09755.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv.cs*, abs/2210.02747, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023a.

Jiahe Liu, Youran Qu, Qi Yan, Xiaohui Zeng, Lele Wang, and Renjie Liao. Fréchet video motion distance: A metric for evaluating motion consistency in videos. *arXiv preprint arXiv:2407.16124*, 2024.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Proc. ICLR*, 2023b.

Xuanchen Lu, Ang Cao, Chao Feng, and Andrew Owens. Generative point tracking and forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Pietro Mazzaglia, Tim Verbelen, Bart Dhoedt, Aaron Courville, and Sai Rajeswar. GenRL: multimodal-foundation world models for generalization in embodied agents. In *Proc. NeurIPS*, 2024.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models learn physical principles from watching videos? *arXiv*, 2501.09038, 2025.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. In *arXiv preprint arXiv:2304.07193*, 2023.

Karran Pandey, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, Niloy J. Mitra, and Paul Guerrero. Motion modes: What could happen next?, 2024. URL https://arxiv.org/abs/2412.00148.

Shivansh Patel, Shraddhaa Mohan, Hanlin Mai, Unnat Jain, Svetlana Lazebnik, and Yunzhu Li. Robotic manipulation by imitating generated videos without physical demonstrations, 2025. URL https://arxiv.org/abs/2507.00990.

William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie Gen: A cast of media foundation models. *arXiv*, 2410.13720, 2024.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proc. ICCV*, 2021.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. *Advances in neural information processing systems*, 29, 2016.

Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Motion-I2V: Consistent and controllable image-to-video generation with explicit motion modeling, 2024. URL https://arxiv.org/abs/2401.15977.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3, 2025b. URL https://arxiv.org/abs/2508.10104.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint*, 2025a.

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.

Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Z. Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1723–1736. PMLR, 2023.

Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders, 2016. URL https://arxiv.org/abs/1606.07873.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang,

Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning, 2024. URL https://arxiv.org/abs/2401.00025.

Jiajun Wu, Joseph Lim, Hongyi Zhang, Joshua Tenenbaum, and William Freeman. Physics 101: Learning physical object properties from unlabeled videos. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 39.1–39.12. BMVA Press, September 2016. ISBN 1-901725-59-6. doi: 10.5244/C.30.39. URL https://dx.doi.org/10.5244/C.30.39.

Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. SpatialTracker: tracking any 2d pixels in 3d space. *arXiv*, 2404.04319, 2024.

Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. SpatialTrackerV2: 3D point tracking made easy. In *Proc. ICCV*, 2025.

Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. In *Conference on Robot Learning (CoRL)*, 2024.

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 *Proc. CVPR*, 2025.

Chengbo Yuan, Chuan Wen, Tong Zhang, and Yang Gao. General flow as foundation affordance for scalable robot learning. In *Conference on Robot Learning (CoRL)*, 2024.

Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K. Barral, Raia Hadsell, Zoubin Ghahramani, Andrew Zisserman, Junlin Zhang, and Mehdi S. M. Sajjadi. Efficiently reconstructing dynamic scenes one D4RT at a time. In *Proc. CVPR*, 2026a.

Zewei Zhang, Jia Jun Cheng Xian, Kaiwen Liu, Ming Liang, Hang Chu, Jun Chen, and Renjie Liao. TrajLoom: Dense future trajectory generation from video, 2026b. URL https://arxiv.org/abs/2603.22606.

A Extended related work

This section expands the discussion of Section 2 with details omitted for space.

A.1 Image-conditioned trajectory forecasting

We forecast object motions as 2D point trajectories from a single image. Existing methods differ in what they predict (absolute vs. relative coordinates) and how they model uncertainty (deterministic vs. generative).

Deterministic regression of absolute coordinates. ATM (Wen et al., 2024) pseudo-labels LIBERO (Liu et al., 2023a) with CoTracker (Karaev et al., 2024b) and trains a deterministic transformer to regress trajectories of 32 active points; Tra-MoE (Yang et al., 2025) adds a mixture-of-experts head. Regressing a single output per query, these methods cannot represent the multimodal future implied by a single image, and are biased toward active points by training-time supervision.

Generative diffusion of absolute coordinates. Track2Act (Bharadhwaj et al., 2024b) samples 400 trajectories with a denoising diffusion model (Ho et al., 2020) conditioned on an image and a goal image, pseudo-labeled from Goyal et al. (2017); Damen et al. (2022); Brohan et al. (2023); Walke et al. (2023). While stochastic, it diffuses absolute coordinates and ingests the image as a single global vector with a 600M-parameter network, which in practice hallucinates objects and misses small ones. A separate, training-free method (Pandey et al., 2024) guides a pre-trained image-to-flow model (Shi et al., 2024) with hand-crafted energies, but only handles one pre-segmented object.

Latent trajectory diffusion. These methods generate trajectories in a learned latent space, typically via a VAE (Walker et al., 2016; Li et al., 2018); see Section A.2 for details. Closest to our setting is **WHN** (Boduljak et al., 2026), which trains a β-VAE over a fixed 32×32 trajectory grid and runs rectified flow (Lipman et al., 2022; Liu et al., 2023b) in the resulting latent space, conditioned on patch-level DINOv2 (Oquab et al., 2023) features. WHN is generative and uses patch-level conditioning, but the VAE introduces *jitter* and *boundary errors*, and the fixed 32×32 grid couples training and inference patterns, limiting the flexibility required for downstream applications. JTT builds on WHN's generative, image-grounded formulation and rigorous evaluation, but removes the VAE entirely, generates per-frame velocities directly in coordinate space, and supports arbitrary inference patterns at any density. Following WHN, we evaluate motion plausibility with FVMD (Liu et al., 2024) and the rigidity-based LRTL score (Karazija et al., 2024), in the spirit of Salimans et al. (2016); Unterthiner et al. (2018); Huang et al. (2024); Bansal et al. (2025), since pixel-level metrics are poor proxies for motion generation (Boduljak et al., 2026).

A.2 Earlier latent-VAE trajectory methods

Walker et al. (2016) use a DCT-based linear compression of trajectory offsets together with a VAE to sample futures from static images. The linear compression has limited expressivity, and the evaluation does not test physical plausibility or distributional coverage. Li et al. (2018) use a VAE to generate optical flow maps from a static image and translate them to RGB; this inherits the failure modes of frame-pair flow (drift under occlusion, no long-range consistency) and uses pixel-level metrics rather than motion-aware ones. Both serve as early demonstrations that latent generative modeling of motion is viable, but neither addresses the multi-object, rigidity-aware setting we target.

A.3 Recent work in detail

TrajLoom. TrajLoom (Zhang et al., 2026b) extends the latent-VAE formulation of WHN to *video* conditioning at larger scale: it diffuses VAE latents on a fixed grid using Wan2.1 (Wan et al., 2025) RGB video latents and trains with auxiliary VAE losses and rollout fine-tuning. The contribution is essentially an offset parameterization relative to the grid, plus additional video signal at larger scale. The underlying recipe, latent diffusion over a fixed grid, and its limitations (no arbitrary query patterns, no density beyond the training grid) carry over, while the offset-based parameterization is suboptimal, as we show in this work.

MotionForecasting. MotionForecasting (Thakkar et al., 2026) ("Forecasting Motion in the Wild") extends Track2Act to in-the-wild forecasting on *wildlife* footage and is, architecturally, Track2Act

with the goal image dropped and ResNet-50 swapped for DINOv3 (Siméoni et al., 2025b). The model is a joint-attention transformer over query points, which scales quadratically in the number of queries and cannot run densely. The relative-coordinate parameterization is adopted but never ablated, so its importance is not established. The evaluation runs WHN *zero-shot* on animal videos, even though WHN is trained only on Kubric, so it remains unclear whether the proposed method outperforms WHN under matched supervision. Even in this zero-shot setting, WHN still outperforms MotionForecasting on the acceleration-based distributional metrics. The model also inherits the low-resolution image conditioning of Track2Act, and consequently the same failure mode at object boundaries. The setting also differs from ours: a single animal moving with simple, mostly translational motion, rather than scenes with multiple interacting objects and rigid-body collisions. In addition, MotionForecasting requires occlusion labels as input, whereas we deliberately minimize the number of inputs. Finally, Table 5 shows that, within our architecture and on the benchmark of Boduljak et al. (2026), bilinearly sampling DINO features at the query points, as in MotionForecasting, is worse than sampling full-resolution RGB on every metric except foreground rigidity.

Reimplementation of MotionForecasting. Neither the implementation nor the dataset of Thakkar et al. (2026) is publicly available, so we reimplement the method ourselves. We start from the Track2Act codebase and strictly follow the architecture and training recipe described in the paper. We set the trajectory horizon to $T = 24$ Kubric timesteps, matching prior work and our method, and use ground-truth occlusion labels and the same isotropic delta normalization as for our method. We train for 140 epochs, as in the paper, which amounts to about 180K steps. In contrast, the checkpoints we report for our models are trained for at most 100K steps, since larger models overfit beyond that point (Fig. 3).

Parameter count of GPTF. No public implementation of Lu et al. (2026) is available, so we estimate its size from the architecture described in the paper. On Kubric, the model is a DiT with 6 blocks of width $d = 384$; each block has temporal, spatial, and point–image attention ($3 \times 4d^2$), an MLP, which we assume has ratio 4 ($8d^2$), and AdaLN modulation for its four sublayers ($12d^2$), giving about $32d^2 \approx 4.7$M parameters per block and about 28M for all six. The feature projection, upsampling layer, embeddings, and output layer add about 2M, for about 30M in total. The same estimate for the Kinetics model (12 blocks of width 768) gives about 230M. As for JTT, we exclude the pre-trained image encoder (DINOv3-S).

B MORE RESULTS

Tables 5 to 7 report the full conditioning, support-architecture, and feature-extractor ablations, and Table 11 the per-checkpoint results behind Table 2.

Table 5: **Impact of full-resolution inputs on motion generation.** RGB alone yields strong results, yet more explicit representations, such as segmentation and inverse depth, further improve performance.

Token with	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
Nothing	13,458 (13,006)	17,662 (17,147)	182.73 (175.60)	8.49 (7.61)	5.95 (5.56)
DINO	14,470 (13,901)	19,189 (18,357)	184.19 (170.87)	8.94 (8.48)	6.15 (5.92)
RGB	14,136 (12,386)	18,523 (16,589)	138.27 (122.00)	6.82 (6.43)	6.35 (6.10)
Inverse Depth	15,572 (13,653)	20,116 (17,716)	88.72 (81.63)	4.11 (3.82)	4.56 (4.30)
Segm (binary)	13,232 (12,753)	17,281 (16,816)	93.49 (86.18)	5.84 (5.29)	6.58 (5.99)
Segm (instance)	13,356 (12,888)	17,666 (17,205)	104.64 (97.77)	6.01 (5.24)	6.77 (5.93)

Sampler. Table 8 compares our default 1000-step SDE (Euler–Maruyama) sampler with Euler integration of the probability-flow ODE. The SDE sampler needs at least 250 steps to be competitive, whereas the ODE sampler is robust to the number of steps: with only 10 steps, it improves both FVMD variants and Best-of-K while using $100\times$ fewer function evaluations, at the cost of slightly lower rigidity (LRTL 7.05 vs. 6.77).

Scale. Table 9 compares JTT-XS, JTT-S, and JTT-B, evaluated every 10K steps up to 130K steps. These trends are easier to see in Fig. 3, enlarged in Figs. 5 and 6; Fig. 7 shows the flow-matching loss over the full 200K steps. Larger models converge faster: B reaches its lowest FVMD at 50K steps and S at 70K steps. Afterwards, they start to memorize the single ground-truth future available per training

Table 6: **Adding a support-point architecture.** Besides enabling dense decoding, the support-point architecture (Dense) improves FVMD and foreground rigidity (LRTL (FG)) over the baseline (Sparse) for both model sizes, and improves every metric for S. The slight degradation in Best-of-K and LRTL (ALL) for B is expected, as the Dense model solves a harder task, and is not visible qualitatively.

Model	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
B	13,514 (11,812)	17,716 (15,829)	124.37 (115.24)	6.67 (6.13)	6.01 (5.58)
B (Dense)	13,266 (12,333)	17,437 (15,958)	130.44 (118.76)	6.80 (6.13)	5.87 (5.31)
S	12,546 (12,238)	16,769 (16,351)	160.25 (129.77)	7.69 (6.91)	6.57 (6.26)
S (Dense)	11,807 (11,591)	15,695 (15,452)	135.85 (128.69)	6.84 (6.64)	5.82 (5.46)

Table 7: **The impact of the image feature extractor on motion quality** on Kubric-A. All models diffuse velocities; only the feature extractor changes. Adding register tokens (Reg) improves all metrics, and multi-layer (DPT) features further improve Best-of-K and foreground rigidity. Larger or alternatively pretrained extractors (DINOv2 G, 4RC, VGGT-Omega, VJEPA2.1) do not yield consistent gains. Results are reported as Average (Best).

Feature extractor	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
DINOv2 S	14,136 (12,386)	18,523 (16,589)	138.27 (122.00)	6.82 (6.43)	6.35 (6.10)
DINOv2 S+Reg	13,441 (12,793)	17,623 (16,763)	126.00 (116.75)	6.67 (5.95)	6.21 (5.86)
DINOv2 S+Reg+DPT	13,514 (11,812)	17,716 (15,829)	124.37 (115.24)	6.67 (6.13)	6.01 (5.58)
DINOv2 G+Reg+DPT	14,519 (12,522)	18,482 (16,442)	123.72 (104.64)	6.75 (5.88)	6.04 (5.57)
4RC+DPT	13,363 (12,227)	17,441 (16,284)	128.20 (121.22)	6.40 (5.56)	5.88 (5.29)
VGGT-Omega+DPT	13,821 (12,824)	18,045 (17,054)	150.18 (125.42)	7.27 (6.43)	6.53 (6.09)
VJEPA2.1+DPT	13,988 (12,257)	18,274 (16,416)	146.00 (131.63)	6.71 (5.95)	6.02 (5.65)

Table 8: **Effect of the sampler and number of function evaluations (NFE)** on Kubric-A, for a single JiT-B (Sparse) checkpoint at 70K steps. Our default is the 1000-step SDE (Euler–Maruyama) sampler.

Sampler	NFE	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
ODE	10	11,614	15,263	115.59	7.05	6.41
	25	11,404	15,221	126.82	7.95	7.18
	50	11,365	15,266	133.83	8.35	7.51
	100	11,357	15,311	138.16	8.59	7.71
	250	11,359	15,348	141.45	8.74	7.84
	1000	11,362	15,370	143.29	8.83	7.90
SDE	10	32,396	39,223	2,797.51	50.47	31.02
	25	21,258	26,333	337.13	16.87	13.49
	50	17,487	22,372	260.94	12.45	10.95
	100	14,070	18,700	186.54	9.78	8.87
	250	11,901	16,122	139.93	7.65	6.93
	1000	11,957	16,043	126.12	6.77	6.19

image instead of learning the distribution of futures: qualitatively, their samples collapse toward one future per image. This is also visible in the metrics, as their FVMD increases while Best-of-K and LRTL continue to improve. In contrast, the FVMD of XS decreases almost monotonically throughout training. Since sampling and evaluation are expensive, we did not sample and evaluate S and B beyond 130K steps, as they had clearly started to memorize by then; their flow-matching loss over the full 200K steps is shown in Fig. 7. We evaluated XS further, up to 200K steps (Table 10): it keeps improving and reaches FVMD (D) 11,771, Best-of-K 120.78, and LRTL 6.52, comparable to our larger models (Table 2). Hence, optimizing plausibility (LRTL) does not necessarily optimize diversity: a model can produce increasingly rigid trajectories while covering fewer of the possible futures, as reflected in its FVMD. This makes model selection tricky, since no single checkpoint is best on all metrics, and it has direct consequences for evaluation: depending on model size and training time, the ranking of models can be reversed. For example, at 50K steps, B has the lowest FVMD (D) of the three models (11,676 vs. 14,165 for XS), whereas at 130K steps, it has the highest (15,708 vs.

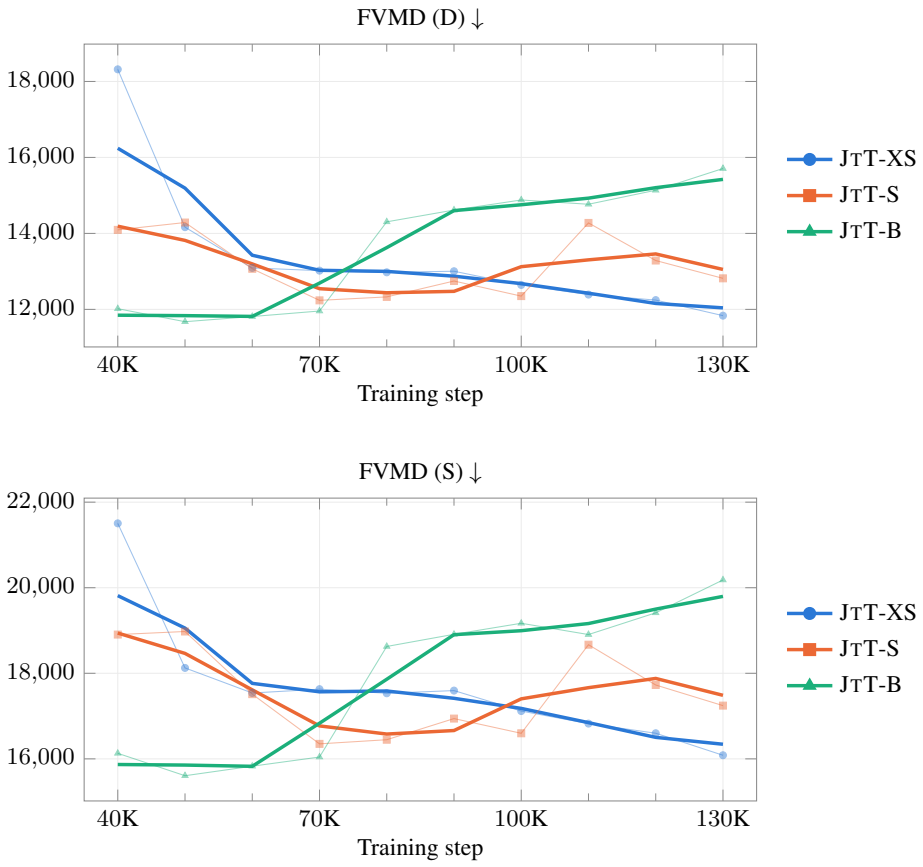

Figure 5: **Distributional fidelity during training** (Kubric-A; lower is better), enlarged from Fig. 3. B and S reach their best FVMD at 50K and 70K steps, respectively, and then degrade as they memorize the single ground-truth future per training image, whereas the FVMD of XS decreases steadily. Thin lines: evaluated checkpoints; thick lines: 3-checkpoint moving average.

11,835 for XS). Prior work largely leaves this model selection implicit, typically reporting a single checkpoint without discussing how it was chosen, which makes it difficult to know which model is better on which metric. To avoid results that depend on a single, arbitrarily chosen checkpoint, we therefore report the average over the last five checkpoints up to 100K steps, together with the best among them (Section 4).

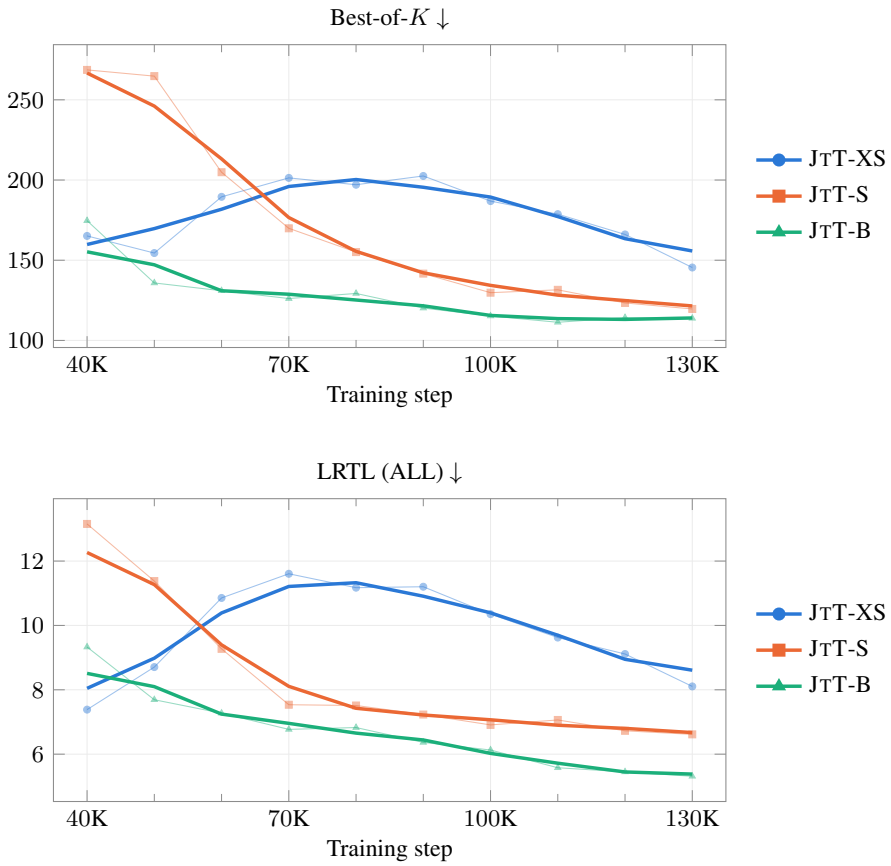

Figure 6: **Best-of-K and rigidity during training** (Kubric-A; lower is better), complementing Fig. 3. Best-of-K and LRTL keep improving for S and B even after their FVMD degrades, as their samples collapse toward the single ground-truth future per training image. Thin lines: evaluated checkpoints; thick lines: 3-checkpoint moving average.

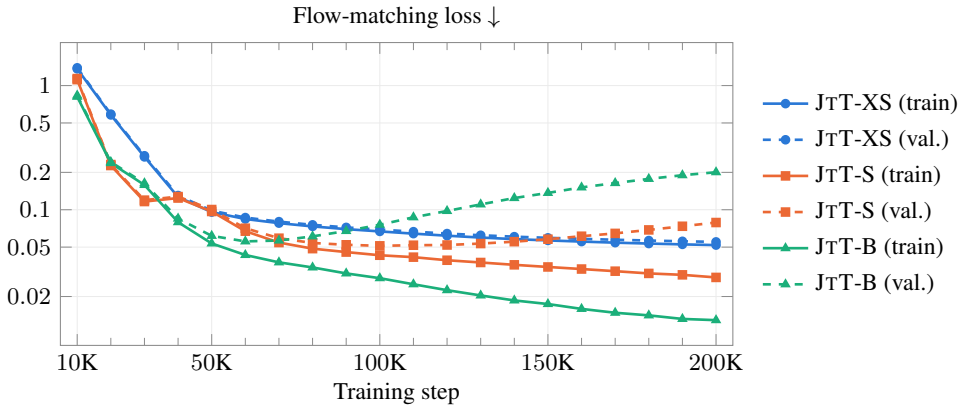

Figure 7: **Flow-matching loss over the full training run** (Kubric-A; log scale), averaged over diffusion timesteps 0.1–0.9, complementing Fig. 3. Solid: training loss; dashed: validation loss. The validation loss of B is lowest at 60K steps and of S at 100K steps, after which both rise while their training losses keep falling; for XS, the two losses stay close and still decrease at 200K steps.

Table 9: **Model performance through training for different model sizes** on Kubric-A, up to 130K steps. Larger models reach their best FVMD earlier (B at 50K, S at 70K steps), after which FVMD increases while Best-of-K and LRTL keep improving; the FVMD of XS decreases almost monotonically.

Model	Step	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
XS	40K	18,319	21,505	165.15	7.38	5.76
	50K	14,165	18,125	154.48	8.71	7.28
	60K	13,092	17,539	189.52	10.85	9.36
	70K	13,020	17,629	201.34	11.60	10.16
	80K	12,976	17,533	197.07	11.17	9.97
	90K	13,004	17,596	202.53	11.20	9.97
	100K	12,644	17,118	186.86	10.35	9.36
	110K	12,390	16,824	178.74	9.62	8.85
	120K	12,245	16,599	166.08	9.11	8.49
	130K	11,835	16,084	145.46	8.11	7.68
S	40K	14,093	18,905	268.66	13.15	9.12
	50K	14,286	18,977	264.77	11.38	8.37
	60K	13,069	17,512	204.86	9.27	7.41
	70K	12,238	16,351	169.90	7.54	6.26
	80K	12,329	16,447	155.07	7.51	6.46
	90K	12,747	16,942	141.66	7.23	6.45
	100K	12,350	16,598	129.77	6.91	6.30
	110K	14,276	18,668	131.56	7.06	6.31
	120K	13,286	17,726	123.37	6.73	6.32
	130K	12,817	17,245	119.63	6.61	6.18
B	40K	12,018	16,130	174.66	9.33	6.91
	50K	11,676	15,606	135.84	7.69	6.43
	60K	11,812	15,829	130.98	7.27	6.50
	70K	11,957	16,043	126.12	6.77	6.19
	80K	14,305	18,627	129.30	6.83	5.95
	90K	14,615	18,911	120.19	6.37	5.82
	100K	14,880	19,169	115.24	6.13	5.58
	110K	14,768	18,906	111.33	5.58	5.34
	120K	15,136	19,414	114.22	5.45	5.37
	130K	15,708	20,181	113.80	5.31	5.30

Table 10: **Convergence of JTT-XS** (under 10M parameters) trained for 200K steps on Kubric-A.

Model	Step	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
XS	50K	14,165	18,125	154.48	8.71	7.28
	100K	12,644	17,118	186.86	10.35	9.36
	150K	11,737	15,873	130.47	7.21	7.12
	200K	11,771	15,773	120.78	6.52	6.68

Table 11: **Per-checkpoint results of the Sparse and Dense models** on Kubric-A, from which the Average (Best) values in Tables 2 and 6 are computed. Several metrics are non-monotonic across checkpoints.

Model	Step	FVMD (D)	FVMD (S)	Best-of-K	LRTL (ALL)	LRTL (FG)
S	60000	13068.82	17512.31	204.86	9.27	7.41
	70000	12237.69	16350.72	169.90	7.54	6.26
	80000	12328.60	16447.45	155.07	7.51	6.46
	90000	12746.66	16941.53	141.66	7.23	6.45
	100000	12349.56	16597.45	129.77	6.91	6.30
S (Dense)	60000	11957.53	15733.35	137.26	6.64	5.46
	70000	11630.86	15452.36	138.41	6.65	5.63
	80000	11591.28	15592.10	139.02	7.24	6.03
	100000	12049.17	16003.89	128.69	6.80	6.17
B	60000	11812.41	15828.66	130.98	7.27	6.50
	70000	11957.17	16042.63	126.12	6.77	6.19
	80000	14304.91	18627.16	129.30	6.83	5.95
	90000	14615.27	18911.21	120.19	6.37	5.82
	100000	14880.14	19169.12	115.24	6.13	5.58
B (Dense)	60000	12333.12	15958.49	120.30	6.13	5.31
	70000	12738.38	16730.78	118.76	6.35	5.66
	80000	13391.11	17745.91	141.90	8.05	6.47
	90000	13769.36	18143.02	131.45	6.76	5.90
	100000	14096.63	18606.89	139.77	6.72	6.00

C IMPLEMENTATION DETAILS

Table 12 lists the JTT variants, Table 13 the architecture hyperparameters shared by all variants, and Table 14 the training and sampling hyperparameters.

Table 12: JTT model variants.

Model Variant	Enc. Depth	Dec. Depth	Hidden Size (d_{model})	Heads	Head Dim
JTT-XS	6	2	192	3	64
JTT-S	12	2	384	6	64
JTT-B	12	4	768	12	64

Table 13: Shared architecture hyperparameters.

Parameter	Default	Description
`traj_length`	24	Number of frames/timesteps in the trajectory sequence.
`in_channels`	2	Input coordinate dimensions (e.g., x and y).
`frame_feature_size`	384	Dimensionality of the pre-extracted vision features.
`frame_feature_patch_size`	14	Spatial patch size used by the vision backbone.
`mlp_ratio`	4.0	Expansion factor for the hidden dimension in the MLP blocks.
`num_freq_bands`	8	Number of frequency bands for Fourier/Sin-Cos featurization.
`sample_channels`	3	Number of channels to sample from the reference feature map.
`sample_patch_size`	1	Spatial size of the local patch sampled around each query point coordinate.
`sample_method`	`bilinear`	Interpolation method for coordinate-based grid sampling.
`attn_drop`	0.0	Dropout probability for attention weights (applied to middle layers).
`proj_drop`	0.0	Dropout probability for linear projections (applied to middle layers).

Table 14: Hyperparameters for JiT.

	Parameter	Value
Model	Stochastic Interpolant Path	Linear (Rectified-Flow)
	Stochastic Interpolant Target	velocity
	Network Prediction	velocity
	Parameterization	velocities Δ_t
	Feature Extractor	DINOv2
	Feature Extractor Size	Small
	Feature Extractor Registers	4
	Feature Extractor Layers	3, 6, 9, 12
	Layer-Normalize Features	True
	Query Sample Patch Size	1
Training	Optimizer	AdamW
	Learning Rate	6×10^{-5}
	Learning Rate Schedule	constant
	Weight Decay	0.0
	Adam Betas	(0.9, 0.999)
	EMA Decay	0.9999
	Global Batch Size	64
	Total Steps	100,000
	Global Seed	42
	Number of Queries	$\mathrm{Uniform}[1, 1024]$
	Timestep (τ) Distribution	$\mathrm{LogitNormal}(0, 1)$
Sampling	SDE Diffusion	SBDM
	Solver	Euler-Maruyama
	Solver Steps	1000
	Last Step Type	Mean
	Last Step Size	$\frac{1}{1000}$
Hardware	Physical Nodes	1
	GPUs per Node	2
	GPU Type	NVIDIA GH200
	Workers	8
	Training Max Walltime	24 hours

D QUALITATIVE RESULTS ON KUBRIC-A

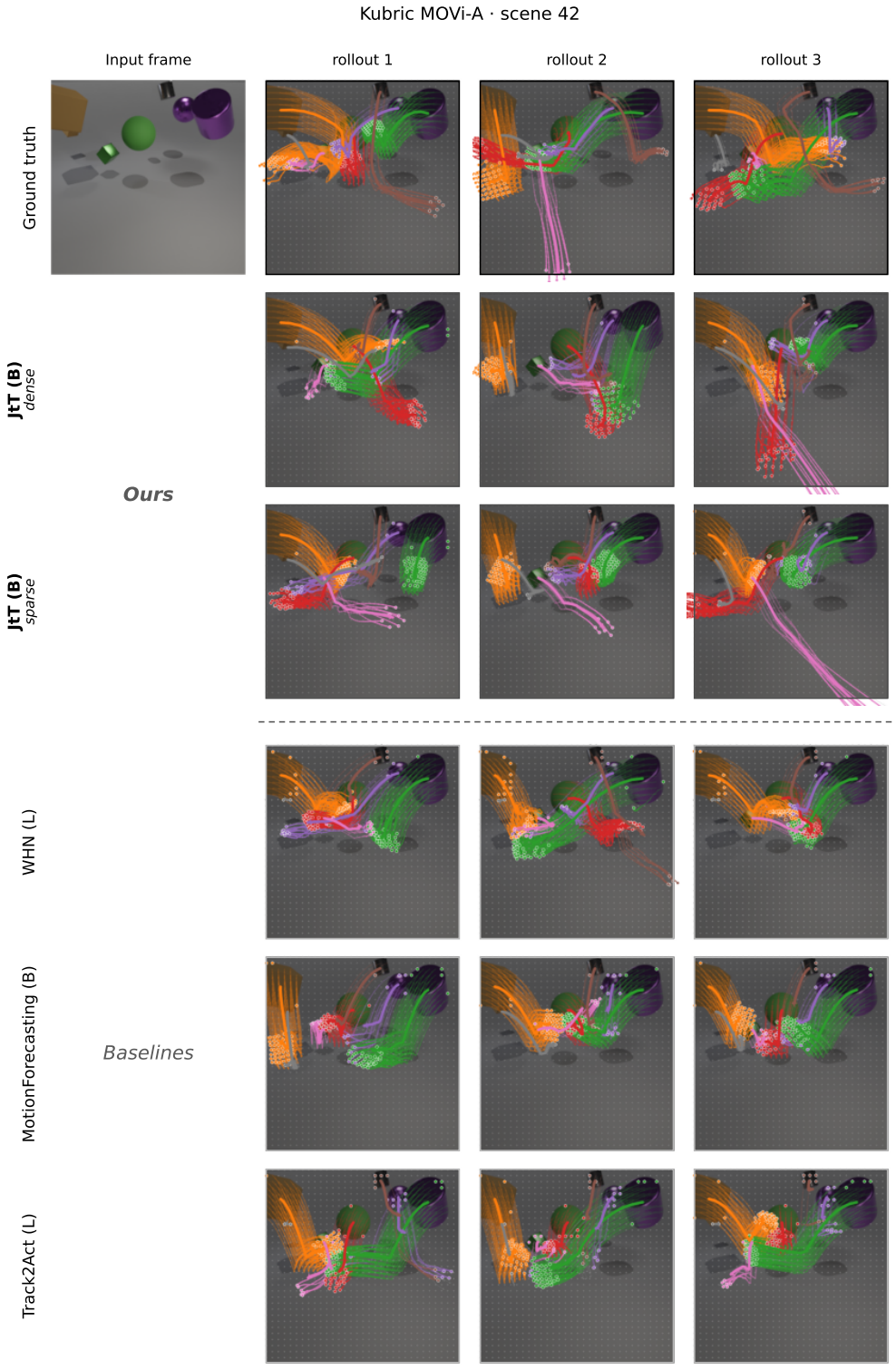

Figure 8: **Comparison on the Kubric benchmark of WHN (Boduljak et al., 2026).** Scene 42.

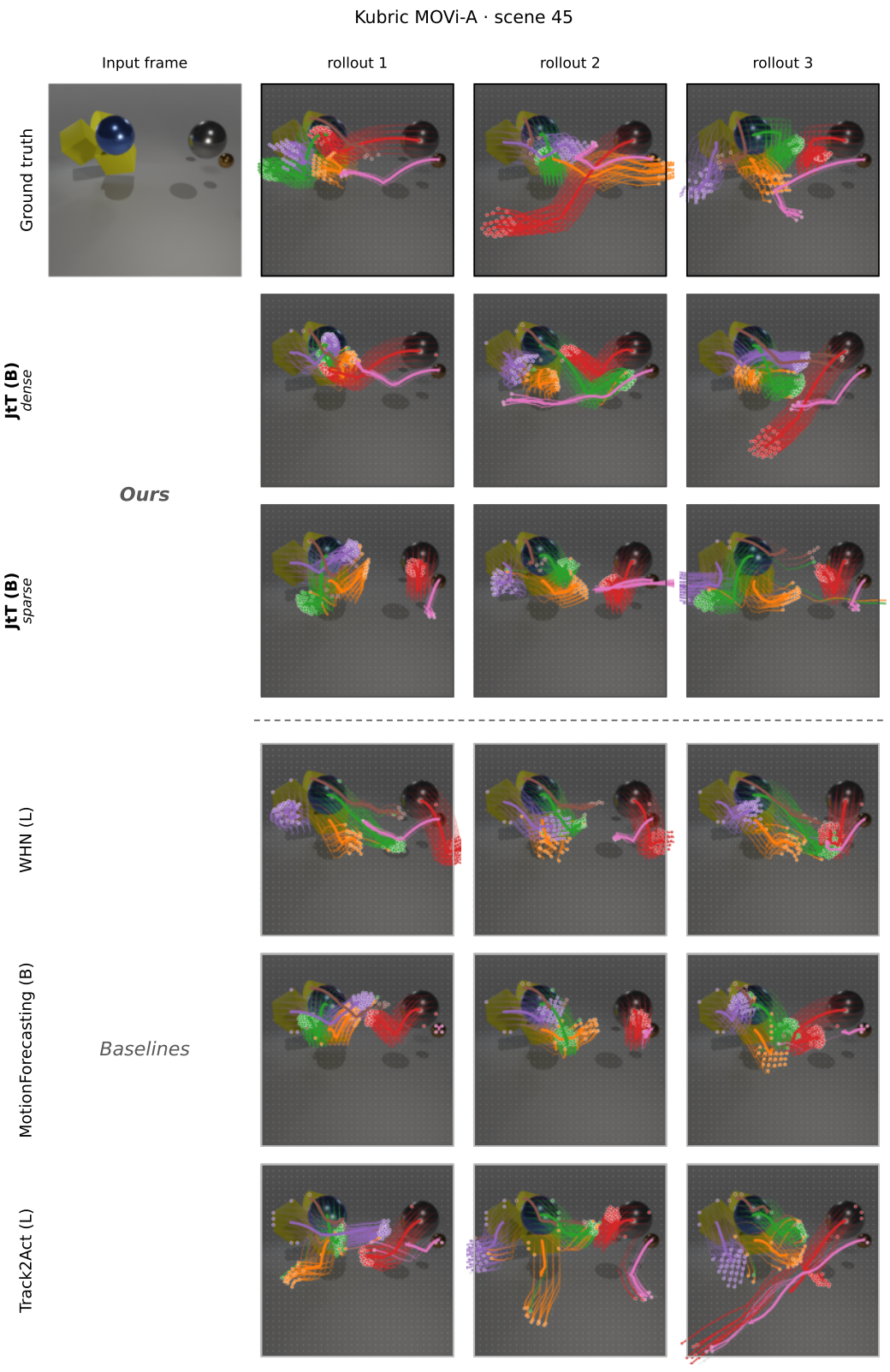

Figure 9: **Comparison on the Kubric benchmark of WHN (Boduljak et al., 2026).** Scene 45.

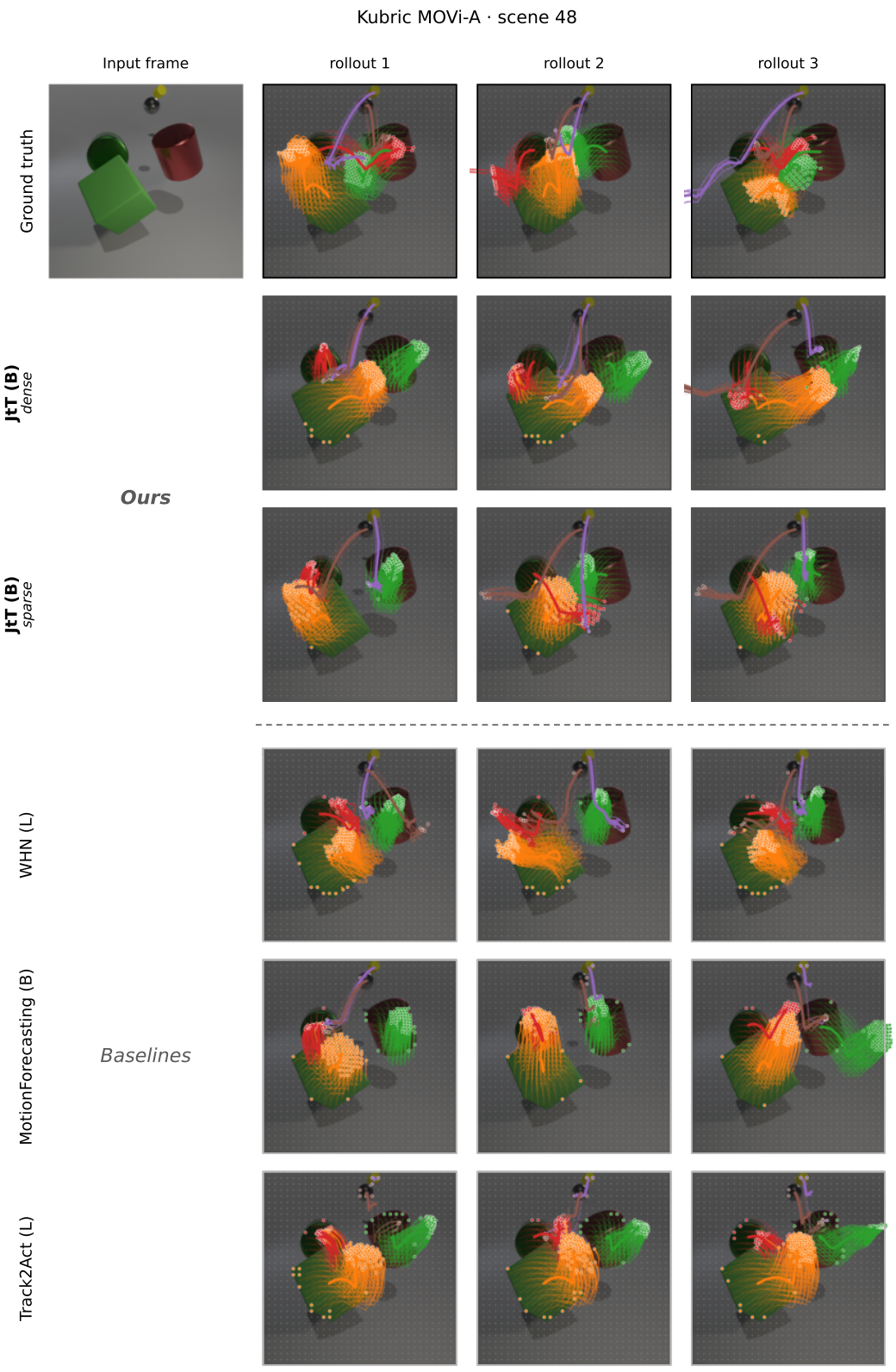

Figure 10: **Comparison on the Kubric benchmark of WHN (Boduljak et al., 2026).** Scene 48.

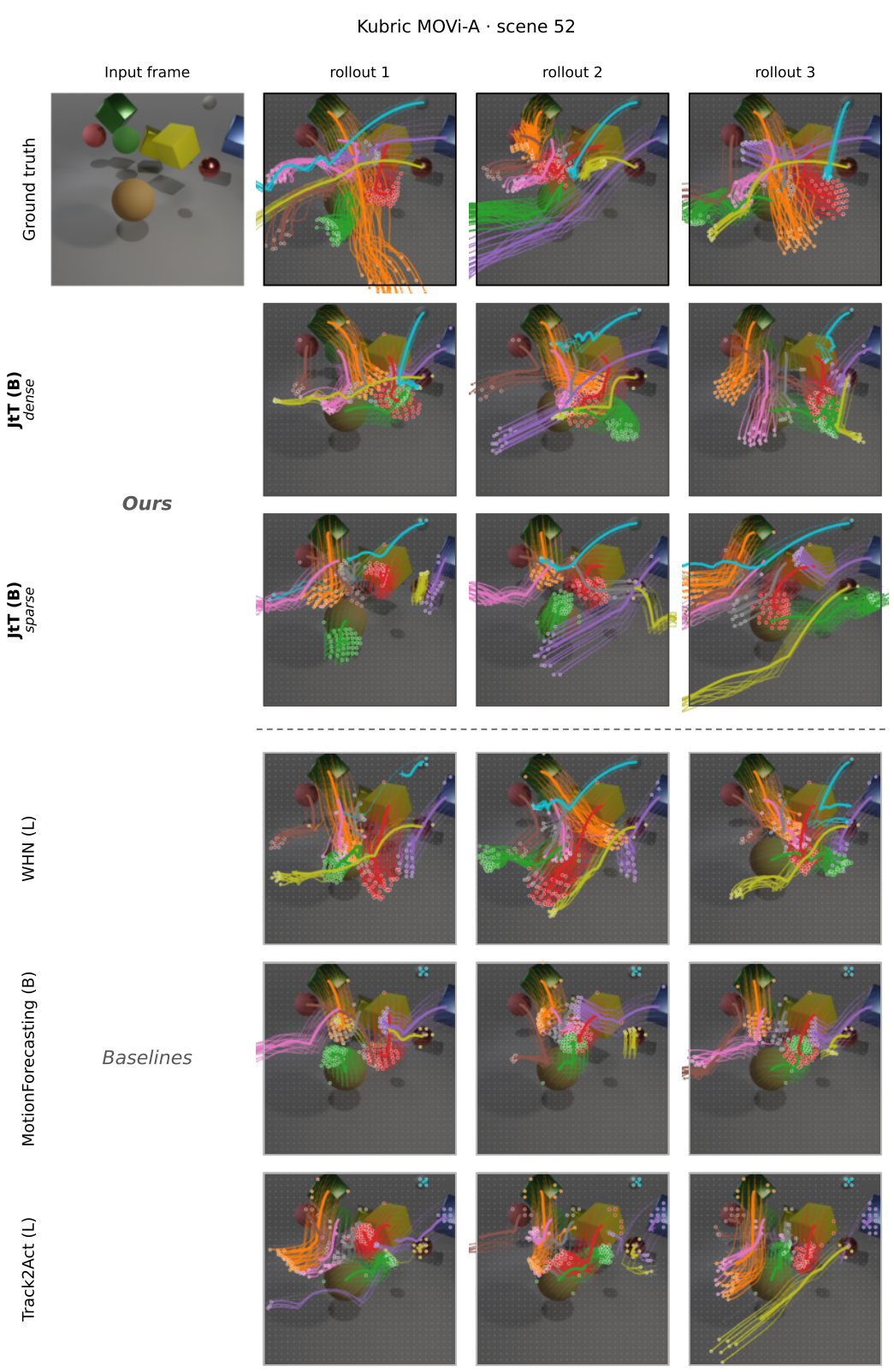

Figure 11: **Comparison on the Kubric benchmark of WHN (Boduljak et al., 2026).** Scene 52.

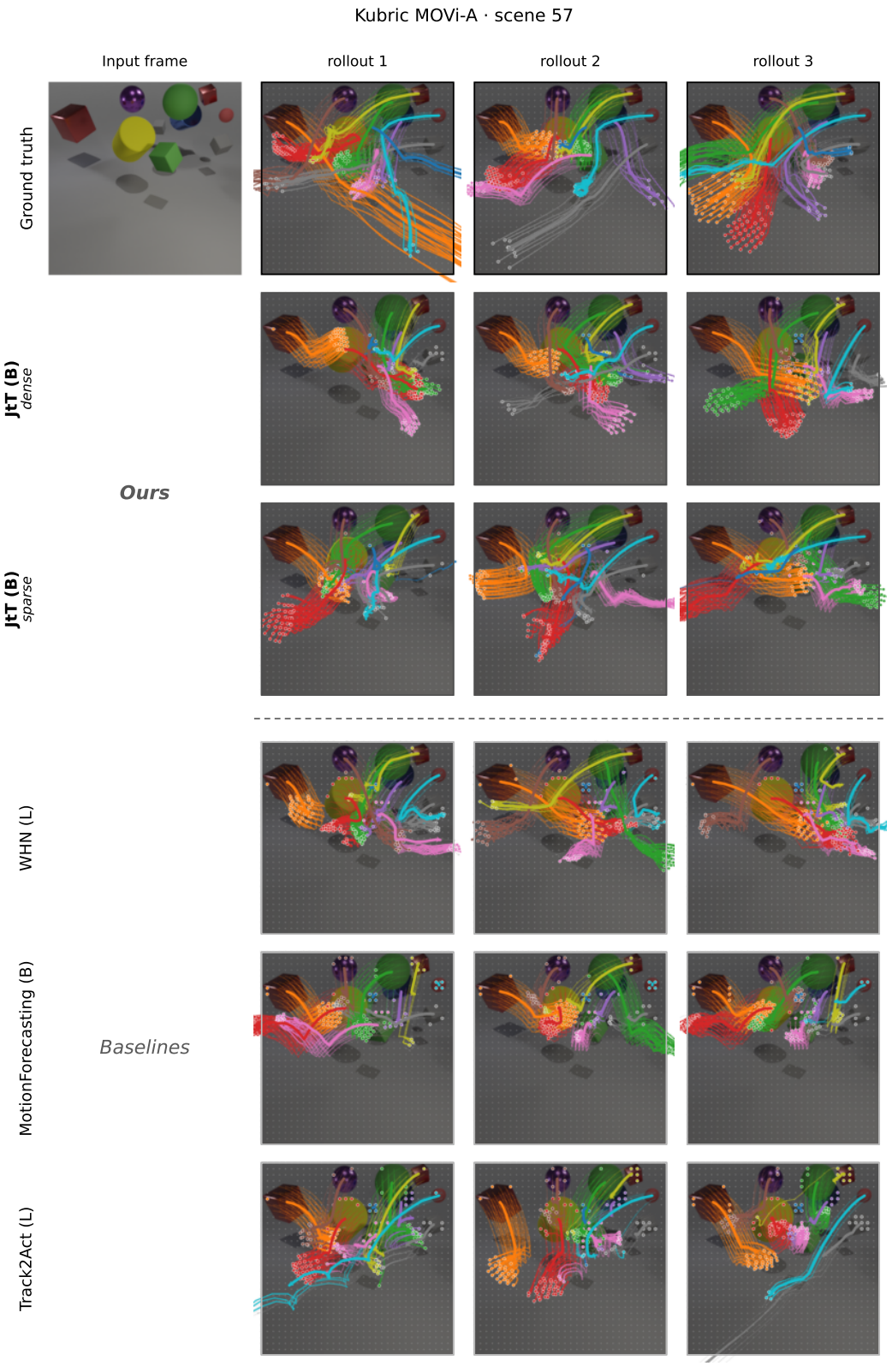

Figure 12: **Comparison on the Kubric benchmark of WHN (Boduljak et al., 2026).** Scene 57.

E QUALITATIVE RESULTS ON PHYSICS 101

Following WHN (Boduljak et al., 2026), we also train JTT on real videos from Physics 101 (Wu et al., 2016), which records simple physical experiments, and show samples on held-out sequences. As for Cityscapes (Section F), real videos provide a single observed future per image and no object-level rigidity ground truth, so a rigorous evaluation is not possible and we report qualitative results only (Figs. 13 to 15). JTT generates rigid motion for rigid bodies, such as objects sliding down a ramp or falling onto a table. It also captures the interaction between rigid and non-rigid bodies, such as a hand releasing an object or a block entering water, and propagates forces through contact, such as a sliding object pushing another one on collision. Finally, it reflects different kinds of forces acting on rigid bodies, such as gravity and buoyancy on a wooden block dropped into water. JTT also retains its ability to generate diverse and plausible futures on real data: for the same input frame, samples differ in how objects slide, fall, and bounce, while remaining consistent with the ground truth (Fig. 16).

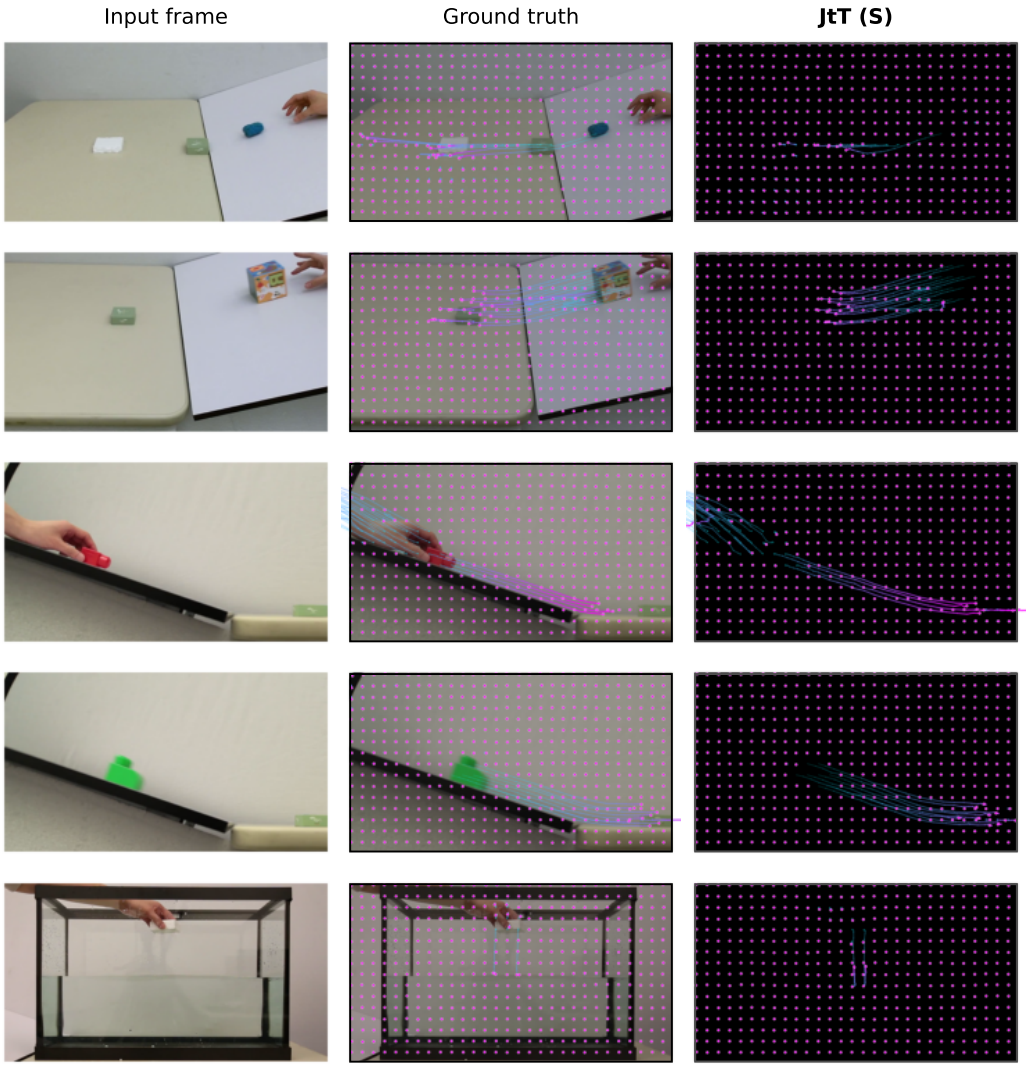

Figure 13: **Qualitative results on Physics 101** (Wu et al., 2016), following the training and evaluation setup of WHN (Boduljak et al., 2026).

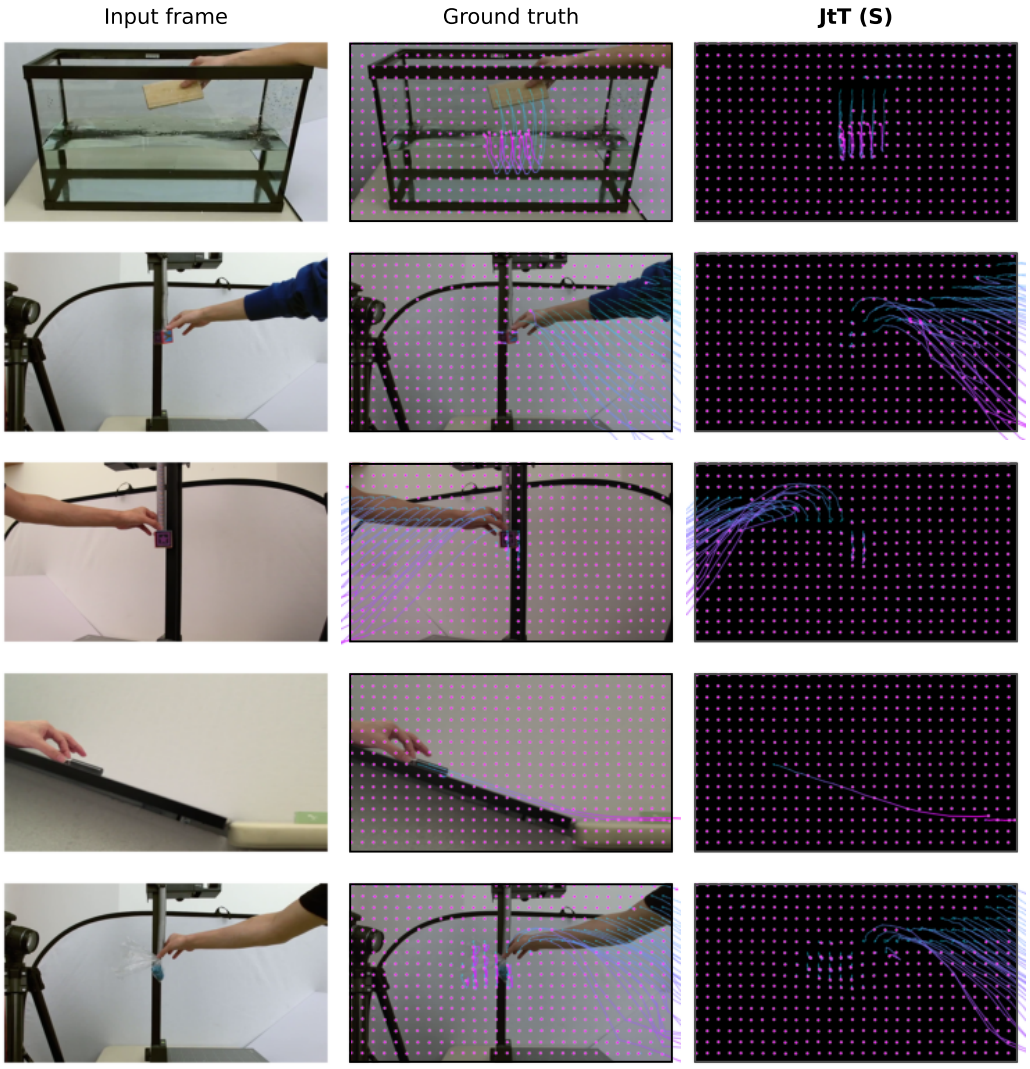

Figure 14: **More qualitative results on Physics 101** (Wu et al., 2016), following the training and evaluation setup of WHN (Boduljak et al., 2026).

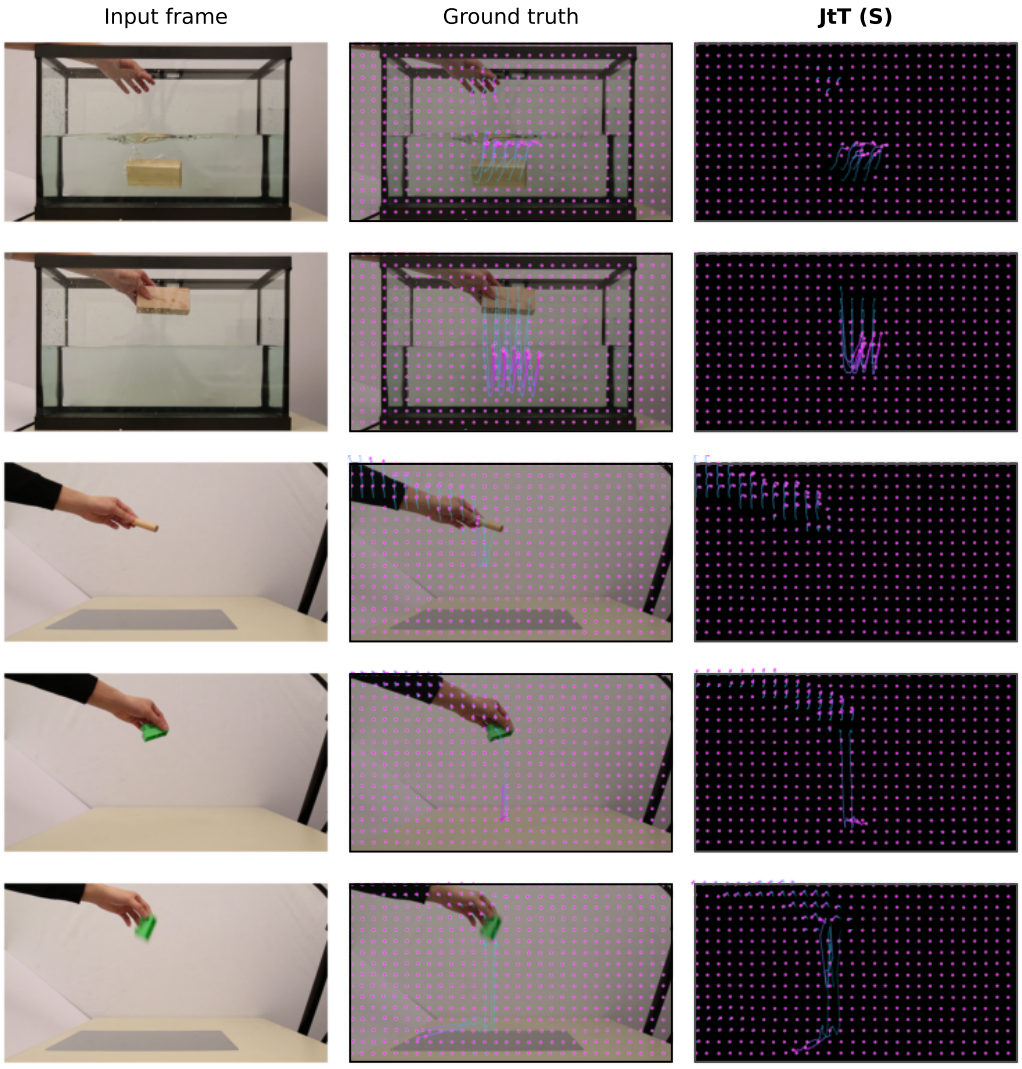

Figure 15: **More qualitative results on Physics 101** (Wu et al., 2016), following the training and evaluation setup of WHN (Boduljak et al., 2026).

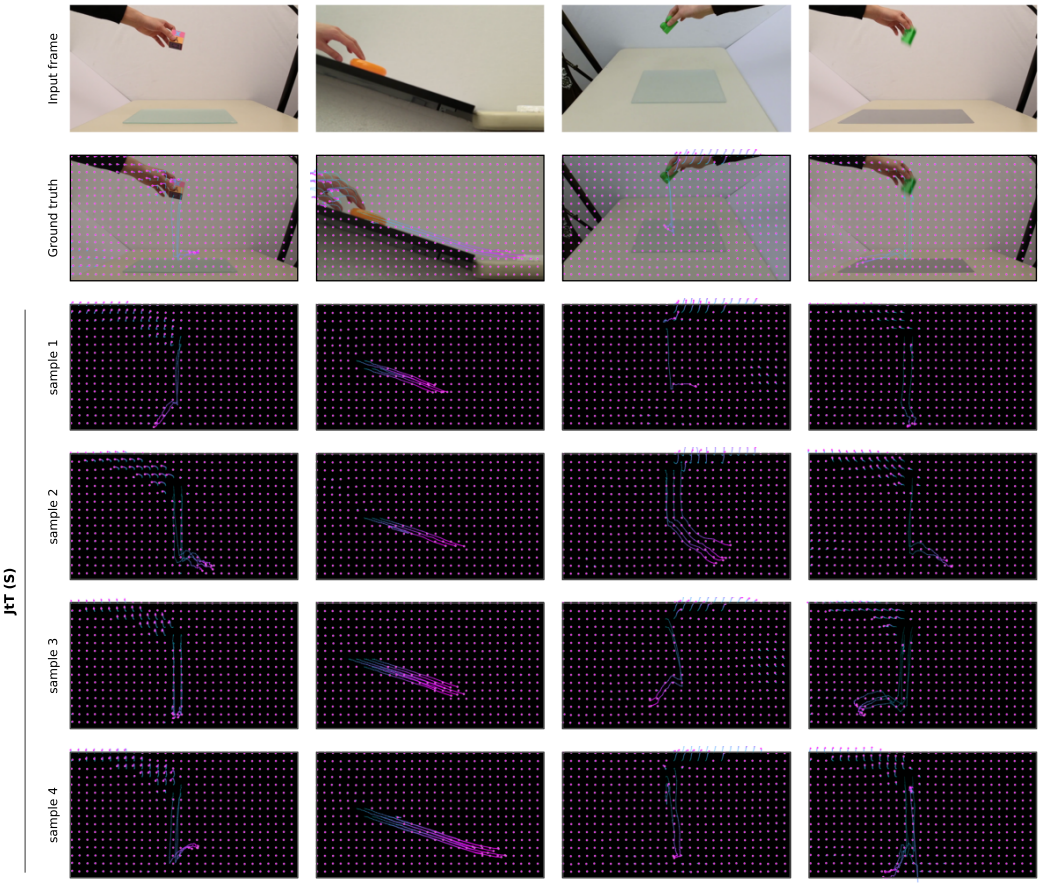

Figure 16: **Sample diversity on Physics 101** (Wu et al., 2016). Four samples from JɪT per input frame, following the training and evaluation setup of WHN (Boduljak et al., 2026).

F Qualitative results on Cityscapes

To test whether JTT transfers to real data, we train it on Cityscapes (Cordts et al., 2016), using the training and evaluation data of WHN (Boduljak et al., 2026). This analysis is qualitative: real videos provide a single observed future per image and no object-level rigidity ground truth, so the rigorous evaluation of our synthetic experiments (Section 4) is not possible here. The training set has 2,975 sequences, the same order of magnitude as Kubric-A. We train for the same number of steps as on Kubric-A, with identical hyperparameters (Table 14), and stop training once we observe overfitting on the held-out set. Figures 17 and 18 show samples. Qualitatively, JTT preserves the rigidity of objects that should move rigidly, such as cars, and handles egocentric camera motion well. It also generalizes to some extent to deformable objects such as pedestrians: their overall motion is plausible, but less precise than for rigid objects. We also verify that JTT samples diverse futures consistent with the ground truth (Fig. 19). It does, and the samples are reasonable, although less diverse than on Kubric-A, likely because it is trained on about $5\times$ less data.

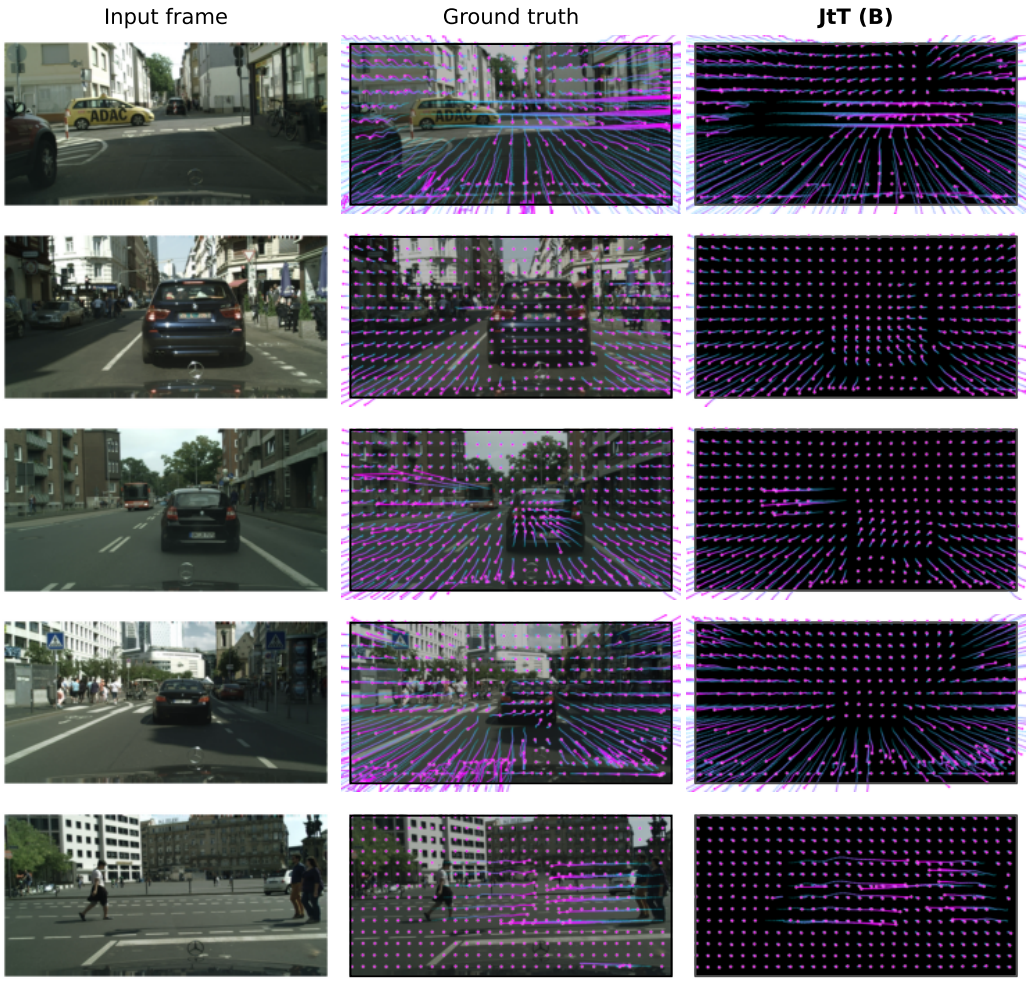

Figure 17: **Qualitative results on Cityscapes** (Cordts et al., 2016), using the training and evaluation split of WHN (Boduljak et al., 2026).

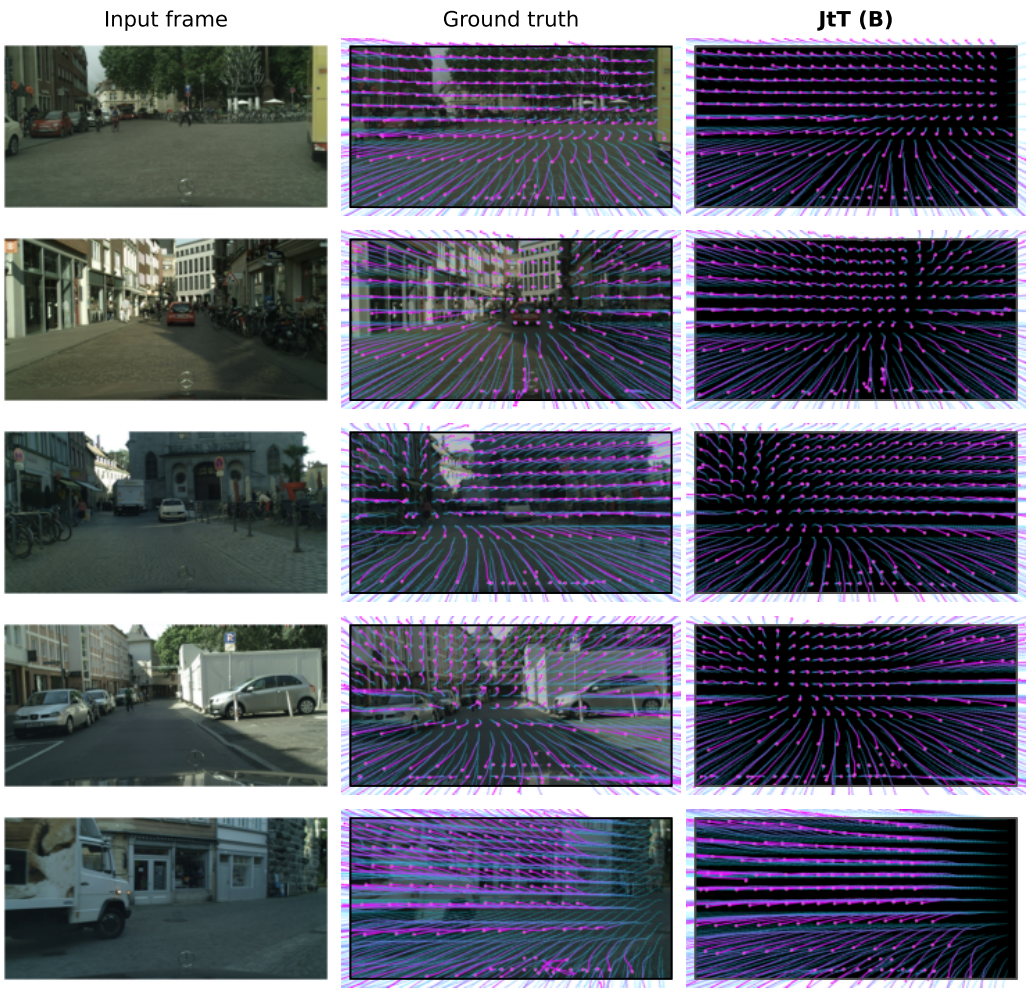

Figure 18: **More qualitative results on Cityscapes** (Cordts et al., 2016), using the training and evaluation split of WHN (Boduljak et al., 2026).

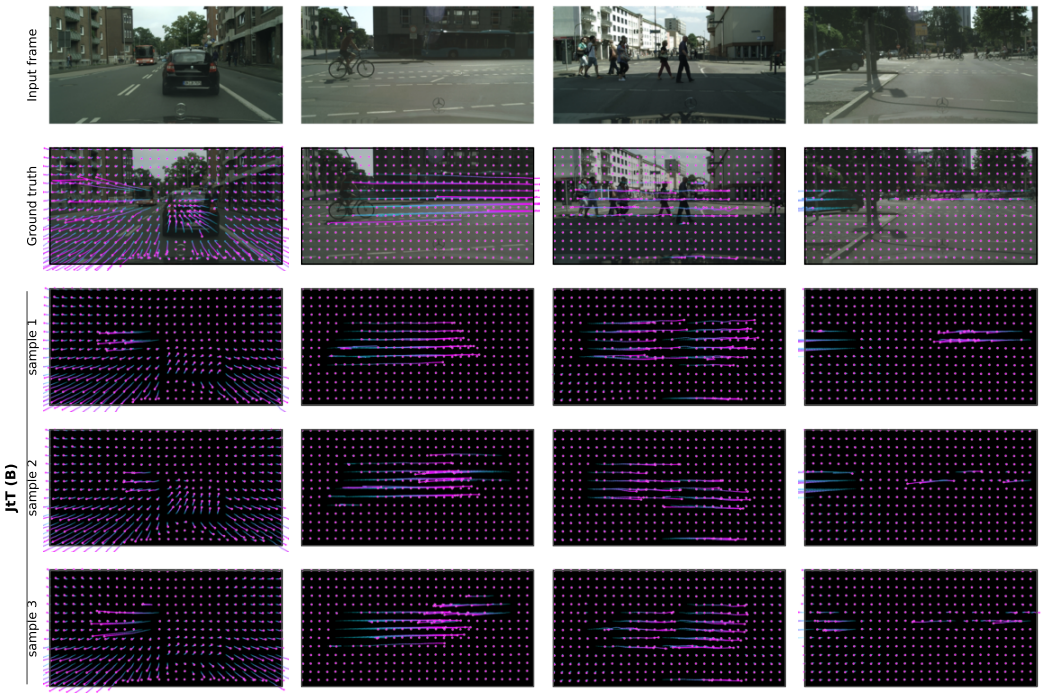

Figure 19: **Sample diversity on Cityscapes** (Cordts et al., 2016). Three samples from JTT per input frame, using the training and evaluation split of WHN (Boduljak et al., 2026).