THROW D4RT INTO THE FUTURE: MOTION WORLD MODELS FROM 4D RECONSTRUCTION PRIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Feedforward 4D reconstruction models learn a spatiotemporal prior of dynamic scenes from large collections of 4D-annotated video. Motion forecasters, in contrast, still rely on 3D trajectory annotations that are expensive to collect and limited in domain. We present FUTURED4RT, a world model that forecasts in the feature space of a frozen 4D reconstruction model and is trained on raw video and timestamps alone. A tokenizer compresses each transition between consecutive scene states into one continuous delta token, a predictor forecasts future tokens, and the frozen decoder of the reconstruction model reads 3D positions from the predicted states. We build the same pipeline on two backbones, D4RT and Point4D. The delta token carries transition-specific content, since the correct token lowers reconstruction loss by 95.1% against copying the preceding state while tokens of other transitions do not. Forecasts beat copying the last state in feature space, and updating the random query vectors of the predictor from the waypoints of one supplied track lowers the average 3D error of the other points on PointMotionBench by 12 to 22% with D4RT and 6 to 20% with Point4D. These gains use an encoding of the full clip, as in training. From three observed frames alone, the forecast stays at the level of holding the last observed position, which we report as the main limitation. The results indicate that a frozen 4D reconstruction model can supply the state space of a motion world model without motion labels.

1 INTRODUCTION

Forecasting 3D scene motion beyond a short video observation is a prerequisite for many real-world systems. Applications include robotic manipulation (Bharadhwaj et al., 2024; Wen et al., 2024; Xu et al., 2024), autonomous driving (Jiang et al., 2023; Zhong et al., 2023), human motion prediction (Karunratanakul et al., 2023; Rempe et al., 2023; Xie et al., 2024), and augmented reality (Grauman et al., 2022; Li et al., 2022). Existing methods learn to forecast motion from annotated object trajectories (Nayakanti et al., 2023; Salzmann et al., 2020), dense 2D point tracks (Thakkar et al., 2026), or sparse 3D point tracks (Yuan et al., 2024; Zhang et al., 2026b). These annotations are expensive to collect at scale. The resulting forecasters also remain restricted to the annotated domains and object categories.

Feedforward 4D reconstruction models offer an alternative source of spatiotemporal knowledge (Wang et al., 2025c; Zhang et al., 2026a; 2025b). Such a model maps a monocular video to the time-varying 3D geometry of a dynamic scene in one forward pass, and it learns this mapping from large collections of video with 4D annotations. Recent models such as D4RT (Zhang et al., 2026a) and Point4D (Jeon et al., 2026) separate an encoder from a decoder at a single interface. The encoder writes the whole video into one encoding, and independent queries, each naming a source point, a target time, and a camera frame, read 3D positions from it. Everything the decoder learns about the video passes through this encoding, so an encoding written by another module can be handed to the same frozen decoder. The encoding carries a spatiotemporal prior learned once from 4D-annotated data, and a forecaster built on it needs no motion annotation of its own.

A world model predicts how the state of an environment evolves (Ha & Schmidhuber, 2018; LeCun, 2022). We present FUTURED4RT, a world model that operates in the feature space of a frozen 4D reconstruction encoder. It uses the encoding of the observed frames as a dense scene state (Section 3.1). Consecutive states share most of their content because appearance and static geometry

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.

persist. A model that predicts each state in full devotes capacity and sequence length to content that remains unchanged. We therefore predict changes between states, compressing each transition into one continuous delta token (Kerssies et al., 2026) (Section 3.2). A learned tokenizer decoder reconstructs the next state from the delta token and the preceding state, so the token needs to encode only what changed. The world model predicts future delta tokens at the target timestamps (Section 3.3). During rollout, the frozen tokenizer decoder expands each predicted token into a dense state, and the frozen decoder of the reconstruction model reads the 3D position of any observed point at any future time (Section 3.4). Both training stages use only video and timestamps, without 3D trajectories, scene flow, or point tracks. We build the same pipeline on D4RT and on Point4D, which differ in architecture, input resolution and query form.

At each step, the predictor takes a random query vector that determines the predicted future. Best-of-many training updates the predictor using only the sampled prediction that best matches the observed future (Section 3.3). One token sequence fixes the future of the whole scene, so a constraint on one point constrains every other point. We steer the forecast by updating these query vectors through the frozen tokenizer decoder and the frozen reconstruction decoder, so that the predicted trajectory of one point fits its supplied waypoints while all model weights stay fixed (Section 3.5). On PointMotionBench (Zhang et al., 2026b), twenty updates from one supplied track lower the average 3D error of the other points by 12 to 22 percent with D4RT and by 6 to 20 percent with Point4D, against the unsteered forecast under identical inputs (Section 4.5). Without a supplied track, the forecast from observed frames alone is about equal to holding the last observed position (Section 4.4).

Our contributions are as follows.

- **Annotation-free 3D motion forecasting from 4D reconstruction priors.** We show that the feature space of a frozen feedforward 4D reconstruction model can serve as the state space of a world model trained on raw video, with two different backbones (Section 3). The frozen decoder of each backbone reads 3D positions from predicted states without fine-tuning, and correct D4RT tokens move decoded points half as far from the true-memory prediction as time-shuffled ones (Appendix B).
- **Compact delta-token dynamics.** A conditional tokenizer compresses each state transition into one continuous token, and a transformer predicts future tokens with one token per step instead of 256 or 326 state tokens (Section 3.2, Section 3.3). The correct D4RT token lowers reconstruction loss by 95.1% against copying the preceding state while wrong tokens stay at the copy level, and forecasts beat copying the last state in feature space at every measured context and horizon (Tables 2 and 3).
- **Steerable motion forecasts.** Updating the random query vectors from the waypoints of one track lowers the average error of all other points on all three PointMotionBench subsets for both backbones (Table 4).

2 Related Work

4D reconstruction and tracking. 4D reconstruction recovers the 3D position of every observed point at every time of a monocular video in a chosen camera frame. Optimization-based pipelines recover this geometry per scene by structure from motion (Schönberger & Frahm, 2016), bundle adjustment on dynamic video (Li et al., 2025; Zhang et al., 2022), dynamic radiance fields or Gaussians (Pumarola et al., 2021; Wang et al., 2025b; Wu et al., 2024), and test-time tracking (Wang et al., 2023). Feedforward models regress the pointmaps of static views in one pass (Leroy et al., 2024; Wang et al., 2025a; 2024), and dynamic variants add per-time pointmaps, a persistent state, scene flow, or world-frame tracks (Feng et al., 2025; Qian et al., 2026; Wang et al., 2025c; Zhang et al., 2025b). Point trackers follow selected pixels across frames in 2D (Doersch et al., 2022; 2023; Karaev et al., 2024) or in 3D (Xiao et al., 2024; Zhang et al., 2025a). D4RT (Zhang et al., 2026a) and Point4D (Jeon et al., 2026) instead read geometry with one query that names a source point, a target time, and a camera frame (Section 3.1). Queries are independent, and the target time is a plain input that admits a future frame, which we build on.

Motion forecasting. Embodied agents forecast scene motion from an observed video prefix because they must choose an action before they observe its outcome. Forecasters differ in the scene abstraction, which fixes what a forecast can answer, and in the output interface. Driving forecasters

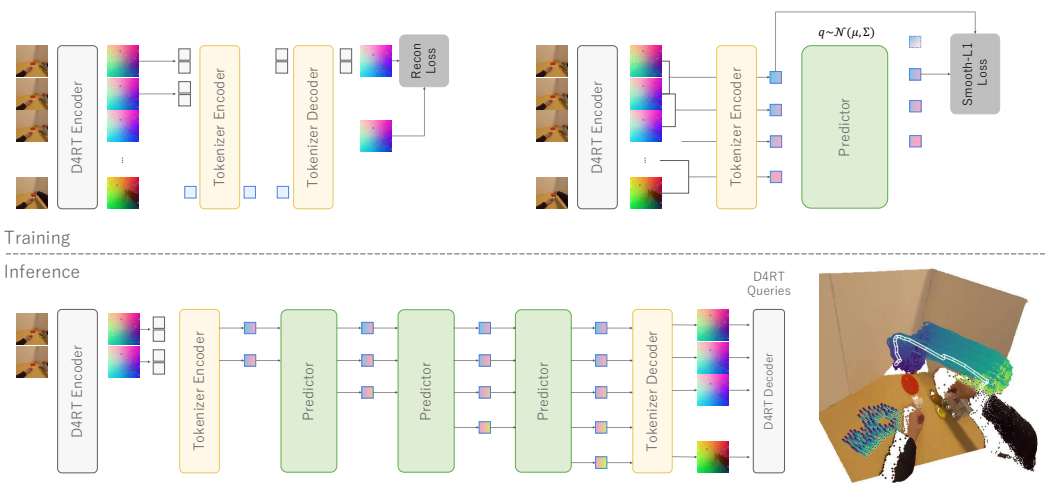


Figure 1: Overview of FUTURED4RT. *Training, left.* A frozen 4D reconstruction encoder, shown for D4RT, turns a clip into one dense state per step, visualized by PCA. The tokenizer encoder compresses each pair of consecutive states into one delta token, and the tokenizer decoder reconstructs the next state from that token and the preceding state. *Training, right.* With the tokenizer frozen, the predictor maps a random query vector η and the preceding delta tokens to the next token. *Inference.* From the observed frames, the predictor rolls out future tokens, the frozen tokenizer decoder expands them into future states, and the frozen reconstruction decoder answers point queries with 3D positions at future times. Neither stage uses 3D annotations.

predict a few agents of known category over a map prior (Nayakanti et al., 2023; Salzmann et al., 2020), which omits everything outside those categories, such as hands and cloth. Point-trajectory forecasters keep every point without categories, as dense 2D tracks (Thakkar et al., 2026) or as sparse 3D tracks for goal-designated points (Yuan et al., 2024; Zhang et al., 2026b). Closest to ours, geometry forecasters extend a feedforward reconstructor into the future, as 4D Gaussians (Song et al., 2026), time-queried point maps (Liu et al., 2026a), or VGGT tokens decoded by frozen layers (Sun et al., 2026). None of these interfaces takes the source point, target time, and camera frame as one query, so one answer needs a dense field plus indexing. We keep the category-free abstraction and the forecast-then-decode strategy of VGGT-World, but read the forecast through the query decoder of the reconstruction model (Section 3.4) and train without the trajectory annotations that point-trajectory forecasters need.

World models in frozen feature spaces. World models differ in the state they predict. The state can be future frames (Bruce et al., 2024; NVIDIA, 2025), a latent learned together with the dynamics (Hafner et al., 2025), or the features of a frozen visual encoder (Assran et al., 2025; Zhou et al., 2025). Features of a frozen image or video encoder describe appearance and semantics, so reading 3D motion from them needs a separately trained head. DeltaTok (Kerssies et al., 2026) compresses the change between consecutive frozen features into one token and forecasts in that token space. We inherit its conditional tokenizer and best-of-many predictor and change the state space to the encoding of a 4D reconstruction model, whose frozen decoder reads the forecast as 3D positions.

3 METHOD

FUTURED4RT forecasts 3D motion in the feature space of a frozen feedforward 4D reconstruction model. We build it on two such models, D4RT (Zhang et al., 2026a) and Point4D (Jeon et al., 2026), with the same recipe of a delta tokenizer, a token predictor, a rollout read out by the frozen decoder, and steering (Figure 1).

3.1 SCENE STATES FROM A FROZEN RECONSTRUCTION MODEL

Backbones. Both backbones encode a whole video with one bidirectional ViT-g of 40 blocks and read geometry with a separate 8-block decoder whose queries never attend to one another. D4RT encodes $2 \times 16 \times 16$ spatiotemporal patches, and its query names a source pixel with a 9×9 RGB patch, a target time and a camera frame. We use the public OpenD4RT checkpoint, trained on 48-frame inputs at 256 pixels.[1] Point4D encodes 14×14 per-frame patches at 252 pixels with alternating frame-local and global attention and one camera and one time token per frame, and its query is a 3D point of a source frame. Both backbones stay frozen in every experiment, and their decoders run in full precision.

Scene states. Let $V = (I_1, \ldots, I_T)$ be a clip of $T = 48$ frames. We define the state $X_k \in \mathbb{R}^{P \times d}$ of step k as the rows of one encoding of the complete clip that the decoder reads for that step. For D4RT a step is a two-frame tubelet, so a clip has 24 states of $P = 256$ spatial tokens of width $d = 1280$ from the memory $\mathcal{M}(V) = W_m E(V, a)$, the output of the encoder E after the projection W_m, whose last row is the projected aspect-ratio token a. For Point4D a step is one frame, so a clip has 48 states of $P = 326$ tokens of width $d = 1536$, namely the camera token, the time token and 324 patch tokens of the global half of the last encoder layer, which is the half its motion head reads.

One encoder call per clip. Both training stages read their states from a single encoder call on the complete clip, so one pass of the 40-block encoder supervises every step of the clip. The encoders are bidirectional, so every state also carries information about frames that come later in the clip. At inference only the observed frames exist, so the context states are built differently from the ones seen in training (Section 3.4), and Section 4 measures what this costs.

3.2 CONDITIONAL DELTA TOKENIZER

Following DeltaTok (Kerssies et al., 2026), we compress the change between consecutive states into one continuous vector. What is new here is the state space, the encoding of a frozen reconstruction model that a geometric decoder reads directly. For a consecutive pair of states the tokenizer encoder A_ϕ and decoder B_ψ compute

$$z_{k+1} = A_\phi(X_k, X_{k+1}) \in \mathbb{R}^{1 \times d}, \qquad \widetilde{X}_{k+1} = B_\psi(z_{k+1}, X_k) \in \mathbb{R}^{P \times d}. \tag{1}$$

The token is continuous, with no codebook and no quantization.

Encoder and decoder are twelve transformer blocks each (Appendix A). The encoder reads a learned token embedding with the rows of X_k and X_{k+1}, and the decoder reads the token with X_k, so the destination state reaches the decoder only through the one-token bottleneck.

Objective and training. We train A_ϕ and B_ψ jointly with an elementwise log-cosh loss,

$$\mathcal{L}_{\text{tok}}(\phi, \psi) = \mathbb{E}_{V,k}\left[\frac{1}{Pd}\sum_{p=1}^{P}\sum_{c=1}^{d}\log\cosh\Big(\widetilde{X}_{k+1,p,c} - X_{k+1,p,c}\Big)\right], \tag{2}$$

where for Point4D the mean is taken separately over the camera, time and patch rows and the three means are summed. One encoder pass supervises all adjacent pairs of a clip, 23 for D4RT and 47 for Point4D, and only the tokenizer parameters change. The D4RT tokenizer trains for 100,000 updates on 4 GPUs with 5 clips each, and the Point4D tokenizer for 10,000 updates on 6 GPUs with 8 clips each, both with AdamW at learning rate 10^{-3}.

The decoder needs the token. A conditional decoder could lower its error without reading the token, because X_k already predicts most of X_{k+1}. Section 4.2 tests this by decoding from the same X_k with the token replaced.

[1] `https://huggingface.co/Lijiaxin0111/OpenD4RT`, an open reimplementation of D4RT.

3.3 WORLD MODEL OVER DELTA TOKENS

After tokenizer training we freeze A_ϕ and B_ψ. One encoding of a clip gives the token sequence (z_k), where z_k connects X_{k-1} to X_k and carries the timestamp τ_k of its last frame. The predictor G_θ maps a random query vector, a window of earlier tokens and a target timestamp to the next token,

$$\widehat{z}_{k+1} = G_\theta(\eta_{k+1}, \mathcal{H}_k, \tau_{k+1}), \qquad \mathcal{H}_k = \{(z_j, \tau_j)\}_{j=k-C+1}^{k}, \tag{3}$$

with a sliding window that covers 8 frames, $C = 4$ tokens for D4RT and $C = 8$ for Point4D. The query vectors are drawn as $\eta \sim \mathcal{N}(0, \sigma^2 I)$ with $\sigma = 0.02$, one draw per predicted token, and independent draws give different futures.

G_θ has 12 blocks of width 768, each applying cross-attention from the prediction queries to the history tokens and a feedforward layer, with temporal rotary embeddings on the timestamps.

Best-of-many objective. To keep several plausible futures, we draw S query vectors per target and train the one that fits best,

$$\mathcal{L}_{\text{pred}}(\theta) = \mathbb{E}\left[\min_{1 \le s \le S} \ell_\beta\Big(G_\theta(\eta^{(s)}, \mathcal{H}_k, \tau_{k+1}),\, z_{k+1}\Big)\right], \tag{4}$$

where ℓ_β is the smooth-L_1 loss with $\beta = 0.1$ and $S = 256$. We select with detached weights and re-evaluate the selected candidate with gradients, and only θ changes.

Training. Each clip supplies every next-token target, 22 for D4RT and 46 for Point4D, and both predictors use AdamW at learning rate 10^{-4} on 4 GPUs. The D4RT predictor trains on 384 clips per update for 10,000 updates, about 42 hours, and we use update 9,750, which has the lowest validation loss. The Point4D predictor trains on 136 clips per update under a schedule of 100,000 updates, and we evaluate update 14,500, reached after about 45 hours.

3.4 ROLLOUT AND GEOMETRIC READOUT

Given the first K states of an encoding, we initialize the history with the observed tokens inside the window and the dense state with $\widehat{X}_K = X_K$. Each step draws a query vector, predicts a token and expands it,

$$\widehat{z}_{k+1} = G_\theta\Big(\eta_{k+1}, \widehat{\mathcal{H}}_k, \tau_{k+1}\Big), \qquad \widehat{X}_{k+1} = B_\psi\Big(\widehat{z}_{k+1}, \widehat{X}_k\Big). \tag{5}$$

The only inputs after the observed part are timestamps, and rollouts stay inside the 48-frame range of the backbone.

Readout. The predicted states have the width and layout of the encoding, so we put them back where the encoder would have written them,

$$\widehat{\mathcal{M}} = \Big[\, X_1 \,\|\, \cdots \,\|\, X_K \,\|\, \widehat{X}_{K+1} \,\|\, \cdots \,\|\, \widehat{X}_N \,\Big], \tag{6}$$

where N is the number of states per clip, 24 for D4RT and 48 for Point4D, $\|$ concatenates rows, and D4RT also appends the aspect-ratio row of the same encoding. With the frozen decoder $\mathcal{D}$ and position head W_{xyz} of the backbone, a query q with embedding $\mathrm{emb}(q, V)$ reads

$$\widehat{f}(q) = W_{xyz}\, \mathcal{D}\Big(\mathrm{emb}(q, V),\, \widehat{\mathcal{M}}\Big) \in \mathbb{R}^3. \tag{7}$$

For Point4D we lift the source pixel to 3D through the depth and intrinsics that Point4D predicts for the observed frames.

Observed-only inference. The benchmark supplies three real frames. We fill the remaining input slots with repeats of the last observed frame and encode this padded clip once. This gives $K = 2$ context states for D4RT, whose first tubelet holds frame 0 twice, and $K = 3$ for Point4D, and the rollout produces the remaining states. The predictor never saw such padded context states in training.

Full-clip context. In the full-clip setting we encode the real 48-frame clip, as in training, keep the same K context states and roll out the rest. The bidirectional encoder lets these context states carry information about the scored future frames, so full-clip results do not measure forecasting from three frames. We use this setting to evaluate the predictor and steering on the kind of state they were trained on.

Scale and anchor. We fit one scale per scene between the supplied and the decoded observed positions of the observed-only encoding, shared by all methods, and add predicted displacements relative to frame 2 to the supplied position at frame 2.

3.5 Steering with one supplied track

We adjust the random query vectors η of all rollout steps so that the decoded 3D trajectory of one supplied point, the guide, matches its supplied future waypoints, and we read every other point from the same adjusted forecast. All weights stay frozen, so steering tests whether the information in one track reaches the rest of the scene, and we make no wider claim about controllability. The guide is the smallest selected point identifier of a scene, fixed before we look at any prediction error, and it is excluded from every steering metric. When the guide has no visible future waypoint, steering returns the unsteered forecast.

Objective. Let $\mathcal{W}$ hold the visible guide waypoints (t, w) and let q_t^{g} be the guide query at target frame t. With η^0 the initial draw, we minimize

$$\mathcal{L}_{\mathrm{steer}}(\eta) = \frac{1}{\nu\,|\mathcal{W}|} \sum_{(t,w)\in\mathcal{W}} \left\| \widehat{f}(q_t^{\mathrm{g}}) - w \right\|_2 \;+\; \frac{\lambda}{D} \left\| \eta - \eta^0 \right\|_2^2, \tag{8}$$

where D is the number of entries of η, $\lambda = 10^{-3}$, and ν is the mean distance from the anchor to the waypoints, floored at 0.05 m. The readout depends on η through Equation 5 and Equation 6, and we differentiate the whole path. We run Adam on η at learning rate 0.05 and, after each step, rescale $\eta - \eta^0$ so that its root mean square stays at or below 0.5. Budgets of 5, 10 and 20 updates share one path, and each reports the iterate with the lowest guide error so far, including the initialization. Appendix D gives the numerics and the cost.

4 Experiments

4.1 Setup

Data. Both training stages use the training split of MolmoMotion-1M (Zhang et al., 2026b), from the subsets with reconstructed videos. These are Xperience (Ropedia, 2026), EgoDex (Hoque et al., 2026), Stereo4D (Jin et al., 2025) and HD-EPIC (Perrett et al., 2025), which together give 156,728 clips of at least 48 frames of egocentric manipulation and third-person scenes (Appendix A). A clip is 48 consecutive frames at stride 1, cropped with a fixed center square and resized to 256 pixels for D4RT or 252 pixels for Point4D, with no flip and no color jitter, and the loader returns frames and timestamps only. Feature-space validation uses 64 held-out clips of the test split whose videos are absent from training.

Benchmark and protocol. PointMotionBench (Zhang et al., 2026b) scores 3D point forecasts on WorldTrack (140 scenes, camera-at-target-time coordinates), DAVIS (Pont-Tuset et al., 2017) (87 scenes, camera frame of frame 0) and HOT3D (Banerjee et al., 2025) (1,066 scenes, fixed camera frame). Following the released runner, the history is the real frames 0 to 2, the scored future is the real frames 3 to 32, and each scene contributes up to 24 points with their supplied observed 3D history. We run one rollout per scene and report ADE, the mean 3D error in meters, FDE, the error at the last visible frame, and PWT, the fraction of point-times within 1, 2, 5, 10 and 20 cm averaged over thresholds. At test time we resize full frames so that the official query points stay inside the image, whereas training used center crops. No caption or language enters our inference, while MolmoMotion is language-conditioned and scores the best of 5 samples, so we list its published scores as a reference and not as a matched comparison.

Table 1: 3D point trajectory prediction on PointMotionBench. Baselines are copied from MolmoMotion (Zhang et al., 2026b) (best-of-5, language input where supported). **Bold** and underline mark the best and second best over the published rows, since our rows receive more input. Our rows use the released H3/F30 runner on 1,293 scenes with one rollout and no text, and are scored on held-out points with the guide excluded. †Full-clip context, which encodes all 48 real frames and keeps only the context states of the three observed frames (Section 3.4). ‡One future ground-truth guide track is supplied, so the row measures transfer from that track rather than forecasting. Steering uses 20 updates.

Paradigm	Model	Inputs		HOT3D			WorldTrack			DAVIS		
		Frames	Text	ADE↓	FDE↓	PWT↑	ADE↓	FDE↓	PWT↑	ADE↓	FDE↓	PWT↑
Non-parametric	Static	1	✗	0.180	0.316	0.293	0.167	0.317	0.390	2.281	4.360	0.085
	Extrapolate	3	✗	0.159	0.309	0.351	0.184	0.432	0.436	2.683	5.741	0.104
Pixel-space	Wan2.2-5B (Team Wan et al., 2025)	1	✓	0.200	0.308	0.253	0.852	1.046	0.090	3.074	5.192	0.051
	Cosmos Predict (NVIDIA, 2025)	5	✓	0.225	0.294	0.199	0.831	0.988	0.072	4.191	6.368	0.033
3D model	ObjectForesight (Soraki et al., 2026)	3	✗	0.129	0.192	0.353	–	–	–	–	–	–
	EgoScaler (Yoshida et al., 2025)	1	✓	0.170	**0.179**	0.218	–	–	–	–	–	–
	Robot4DGen (Liu et al., 2026b)	3	✗	0.212	0.271	0.112	0.548	0.704	0.121	2.120	3.382	0.081
2D track	Track2Act (Bharadhwaj et al., 2024)	1	✗	0.294	0.413	0.202	1.230	1.567	0.053	4.853	8.110	0.018
3D track	MolmoMotion-FM	1	✓	0.183	0.311	0.286	0.165	0.305	0.401	1.380	2.205	0.165
	MolmoMotion-FM	3	✓	0.135	0.255	0.382	0.158	0.295	0.438	1.480	2.520	0.130
	MolmoMotion-AR	1	✓	0.157	0.290	0.303	0.148	0.269	0.424	**1.146**	**1.843**	**0.199**
	MolmoMotion-AR	3	✓	**0.109**	0.217	**0.444**	**0.143**	**0.261**	**0.445**	1.227	2.108	0.153
Ours	FUTURED4RT, D4RT†	48	✗	0.197	0.331	0.256	0.149	0.270	0.328	2.019	3.784	0.074
	FUTURED4RT, D4RT, steered†‡	48	✗	0.174	0.295	0.285	0.116	0.203	0.400	1.633	3.121	0.101
	FUTURED4RT, Point4D†	48	✗	0.190	0.314	0.273	0.144	0.266	0.349	1.930	3.499	0.087
	FUTURED4RT, Point4D, steered†‡	48	✗	0.152	0.247	0.337	0.135	0.224	0.396	1.573	2.825	0.105

Table 2: Token usage probe of the D4RT tokenizer at update 100,000 on 552 adjacent pairs of 24 test clips. Every row decodes the same preceding state and changes only the token. Reduction is measured against copying the preceding state. Only the correct token beats copying by a wide margin.

Token condition	Loss ($\times 10^{-2}$) ↓	Reduction vs copy (%) ↑
Correct token	0.145	95.1
Token from another clip, same time	3.00	−0.9
Token from the same clip, another time	2.96	0.6
Mean token over all pairs	2.19	26.4
Zero token	3.70	−24.3
Copy the preceding state	2.98	0.0

4.2 WHAT THE DELTA TOKEN CARRIES

We decode each transition of 24 MolmoMotion test clips, 552 adjacent pairs, from the same preceding state with the D4RT token replaced (Table 2). The correct token lowers the loss by 95.1% against copying the preceding state, while a token from another clip at the same time or from the same clip at another time stays at the copy level. The mean token lowers the loss by 26.4%, which bounds the part of the update that all transitions share, so the decoder reconstructs the next state from the token and not from the preceding state alone. The Point4D tokenizer reconstructs the next state of 128 validation clips with 89.6% lower loss than copying the preceding state. The token content also reaches the decoded geometry, since D4RT memory slices rebuilt from correct tokens move decoded 3D points half as far from the true-memory prediction as slices rebuilt from time-shuffled tokens (Appendix B).

4.3 FORECASTING STATES IN FEATURE SPACE

Table 3 scores rollouts on the 64 validation clips against the states of the same encoding, as in training, for several numbers of observed and predicted frames. The D4RT forecast lowers the loss by 34.0% to 45.5% against copying the last observed state at every context length and horizon. For

Table 3: Future state prediction on 64 held-out clips under full-clip context, with context and horizon counted in frames. Losses are the mean smooth-L_1 of 20 sampled forecasts and of copying the last observed state, and the reduction is one minus their ratio. The last column covers all Point4D rows, including the camera and time tokens.

		D4RT			Point4D, patch tokens			Point4D, all
Context	Horizon	Mean ↓	Copy	Red. (%) ↑	Mean ↓	Copy	Red. (%) ↑	Red. (%) ↑
8	2	0.0717	0.1218	41.1	0.0992	0.1234	19.6	48.4
8	6	0.0804	0.1344	40.2	0.1240	0.1651	24.9	60.9
24	2	0.0715	0.1195	40.2	0.0944	0.1144	17.5	43.8
24	6	0.0781	0.1433	45.5	0.1207	0.1514	20.2	57.9
40	2	0.0758	0.1149	34.0	0.1048	0.1255	16.5	46.9
40	6	0.0818	0.1353	39.5	0.1363	0.1686	19.2	61.1

Point4D the reduction over all rows is 43.8% to 61.1%, but most of it comes from the camera and time tokens, and on the patch tokens, which are comparable to the D4RT states, the reduction is 16.5% to 24.9%. Both forecasts beat persistence in feature space everywhere we measured, but a lower feature loss does not by itself give a lower 3D error, which the benchmark tests next.

4.4 Comparison on PointMotionBench

Table 1 places our results under the published PointMotionBench results of MolmoMotion (Zhang et al., 2026b). Our rows use full-clip context (Section 3.4), and the steered rows also receive the waypoints of one guide track. Our rows therefore see more of the clip than the published rows, and the table does not rank them against each other. Under these inputs, steered D4RT reaches WorldTrack ADE 0.116 and FDE 0.203, below every published row, while the published baselines keep the best HOT3D and DAVIS scores. Without future frames, the forecast of either backbone stays at the level of holding the last observed position, at held-out WorldTrack ADE 0.171 for D4RT and 0.168 for Point4D against 0.168 for hold, and best-of-5 selection changes it by at most 3.7% (Appendix C). Two untested explanations for the gap to full-clip context are that padded context states differ from the training states, from which the Point4D predictor is nearly deterministic (five rollouts 0.9 mm apart on WorldTrack), and that we resize full frames at test time but trained on center crops.

4.5 Steering with one supplied track

Table 4 reports steering under full-clip context on held-out points, with the guide excluded from every number. For D4RT, twenty updates lower ADE by 22.2% on WorldTrack, 19.1% on DAVIS and 11.6% on HOT3D against the unsteered forecast, and the improvement is monotone in the budget on every subset and metric. For Point4D, twenty updates lower ADE by 18.5% on DAVIS and 20.0% on HOT3D. On WorldTrack its held-out ADE is lowest after 5 updates, 10.3% below unsteered, and rises with more updates while the guide error keeps falling, which suggests that the optimizer then fits the guide at the expense of the other points. Guide translation, which copies the ground-truth displacement of the guide to every other point, is far better on DAVIS and HOT3D and worse than steering on WorldTrack ADE and FDE, which suggests that the scored points of DAVIS and HOT3D largely move rigidly with the guide. We read these results as evidence that the information of one track reaches the other points through the shared token sequence, not as beating a guide-based heuristic.

In Figure 2, steering lowers held-out ADE in all four scenes, from 0.804 to 0.470 m on the selected DAVIS car, but the steered tracks still stop short of the ground truth.

5 Conclusion

We presented FutureD4RT, a world model that forecasts 3D motion in the feature space of a frozen feedforward 4D reconstruction model and trains on video and timestamps alone. With both D4RT and Point4D as backbones, one continuous delta token carries the content of each transition,

Table 4: Steering from the waypoints of one supplied track, ADE in meters on held-out points with the guide excluded. †Full-clip context (Section 3.4). ‡Also uses the waypoints of the guide track. Compare the steered rows with the unsteered row of the same backbone. Appendix C gives FDE and PWT.

	D4RT			Point4D		
Method	WorldTrack	DAVIS	HOT3D	WorldTrack	DAVIS	HOT3D
Observed only	0.171	2.282	0.223	0.168	2.151	0.223
Unsteered†	0.149	2.019	0.197	0.144	1.930	0.190
Steered, 5 updates†‡	0.126	1.750	0.187	0.129	1.748	0.173
Steered, 10 updates†‡	0.120	1.691	0.182	0.132	1.688	0.165
Steered, 20 updates†‡	0.116	1.633	0.174	0.135	1.573	0.152
Guide translation‡	0.153	0.572	0.043	0.153	0.572	0.043

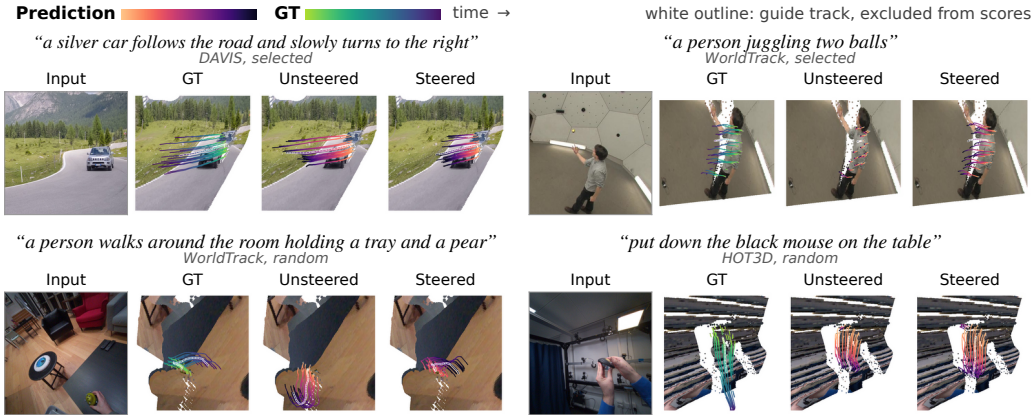


Figure 2: Forecast 3D tracks of the Point4D predictor on PointMotionBench. Each example shows the last observed frame and the ground-truth, unsteered and steered tracks rendered from one shared viewpoint over the Point4D reconstruction of that frame. Color runs from the last observed frame (light) to 30 frames later (dark). Unsteered and steered forecasts use full-clip context, and steering runs 20 updates from the guide track (white outline), which is excluded from all scores. The top row is selected for high steered PWT among scenes with large motion, and the bottom row is the first random draw of a fixed seed for its subset. Steering moves the tracks of the other points toward the ground truth, for example the whole car in the DAVIS example.

the predictor forecasts future states better than persistence in feature space, and steering its query vectors with the waypoints of one supplied track improves the forecast of the other points on all three PointMotionBench subsets. These results suggest that pretrained 4D reconstruction models can serve as state spaces for motion world models, without motion labels or retraining of the geometric readout.

Limitations and future work. From three observed frames alone, the forecast of either backbone stays at the level of holding the last observed position, and simple damped velocity beats it. The benchmark gains use full-clip context, which matches training but lets the context states carry information about the scored frames, and closing the gap to observed-only inference remains open. The token usage probe and the geometry round trip cover D4RT only, the Point4D predictor is evaluated at an intermediate checkpoint, and every benchmark number comes from a single run. We expect the observed-only gap to shrink when context states are built the same way in training and inference, for example by training on padded encodings or by predicting features before the global attention of the encoder, and we leave this to future work.

AI USE STATEMENT

We used large language model assistants to edit the writing and LaTeX of this manuscript. The authors designed the method, ran all experiments, and reviewed every part of the text.

ETHICS STATEMENT

This work trains and evaluates on publicly released video datasets, MolmoMotion-1M and PointMotionBench, which include egocentric and third-person recordings of people. We use them under their research licenses, collect no new data, and do not attempt to identify individuals. Forecasting human motion from video could be misused for surveillance, and our models are research prototypes that are not intended for deployment.

REFERENCES

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning. *arXiv preprint arXiv:2506.09985*, 2025.

Prithviraj Banerjee, Sindi Shkodrani, Pierre Moulon, Shreyas Hampali, Shangchen Han, Fan Zhang, Linguang Zhang, Jade Fountain, Edward Miller, Selen Basol, Richard Newcombe, Robert Wang, Jakob Julian Engel, and Tomas Hodan. HOT3D: Hand and Object Tracking in 3D from Egocentric Multi-View Videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

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 (ECCV)*, 2024.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative Interactive Environments. In *International Conference on Machine Learning (ICML)*, 2024.

Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. TAP-Vid: A Benchmark for Tracking Any Point in a Video. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2022.

Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, João Carreira, and Andrew Zisserman. TAPIR: Tracking Any Point with Per-Frame Initialization and Temporal Refinement. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10061–10072, 2023.

Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J. Black, Trevor Darrell, and Angjoo Kanazawa. St4RTrack: Simultaneous 4D Reconstruction and Tracking in the World. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8503–8513, 2025.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4D: Around the World in 3,000 Hours of Egocentric Video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

David Ha and Jürgen Schmidhuber. Recurrent World Models Facilitate Policy Evolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering Diverse Control Tasks through World Models. *Nature*, 640(8059):647–653, 2025.

Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning Dexterous Manipulation from Large-Scale Egocentric Video. In *International Conference on Learning Representations (ICLR)*, 2026.

Minsik Jeon, Jay Karhade, Deva Ramanan, and Shubham Tulsiani. Point4D: Long-range 4D Motion Reconstruction. *arXiv preprint arXiv:2609.09145*, 2026.

Chiyu Jiang, Andre Cornman, Cheolho Park, Benjamin Sapp, Yin Zhou, and Dragomir Anguelov. MotionDiffuser: Controllable Multi-Agent Motion Prediction using Diffusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4D: Learning How Things Move in 3D from Internet Stereo Videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10497–10509, 2025.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It Is Better to Track Together. In *European Conference on Computer Vision (ECCV)*, 2024.

Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided Motion Diffusion for Controllable Human Motion Synthesis. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Tommie Kerssies, Gabriele Berton, Ju He, Qihang Yu, Wufei Ma, Daan de Geus, Gijs Dubbelman, and Liang-Chieh Chen. A Frame is Worth One Token: Efficient Generative World Modeling with Delta Tokens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Yann LeCun. A Path Towards Autonomous Machine Intelligence. OpenReview, 2022. Version 0.9.2.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding Image Matching in 3D with MASt3R. In *European Conference on Computer Vision (ECCV)*, 2024.

Yiming Li, Ziang Cao, Andrew Liang, Benjamin Liang, Luoyao Chen, Hang Zhao, and Chen Feng. Egocentric Prediction of Action Target in 3D. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. MegaSaM: Accurate, Fast and Robust Structure and Motion from Casual Dynamic Videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Xinhang Liu, Pedro Miraldo, Suhas Lohit, Huaizu Jiang, Naoko Sawada, Yu-Wing Tai, Chi-Keung Tang, and Moitreya Chatterjee. Point4Cast: Streaming Dynamic Scene Reconstruction and Forecasting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026a.

Zeyi Liu, Shuang Li, Eric Cousineau, Siyuan Feng, Benjamin Burchfiel, and Shuran Song. Geometry-aware 4D Video Generation for Robot Manipulation. In *International Conference on Learning Representations (ICLR)*, 2026b.

Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S. Refaat, and Benjamin Sapp. Wayformer: Motion Forecasting via Simple & Efficient Attention Networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

NVIDIA. Cosmos World Foundation Model Platform for Physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

Toby Perrett, Ahmad Darkhalil, Saptarshi Sinha, Omar Emara, Sam Pollard, Kranti Kumar Parida, Kaiting Liu, Prajwal Gatti, Siddhant Bansal, Kevin Flanagan, et al. HD-EPIC: A Highly-Detailed Egocentric Video Dataset. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 23901–23913, 2025.

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 preprint arXiv:1704.00675*, 2017.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Shenhan Qian, Ganlin Zhang, Shangzhe Wu, and Daniel Cremers. Flow4R: Unifying 4D Reconstruction and Tracking with Scene Flow. In *European Conference on Computer Vision (ECCV)*, 2026.

Davis Rempe, Zhengyi Luo, Xue Bin Peng, Ye Yuan, Kris Kitani, Karsten Kreis, Sanja Fidler, and Or Litany. Trace and Pace: Controllable Pedestrian Animation via Guided Trajectory Diffusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Ropedia. Xperience-10M: A Large-Scale Egocentric Multimodal Dataset with Structured 3D/4D Annotations. `https://huggingface.co/datasets/ropedia-ai/xperience-10m`, 2026. Hugging Face dataset.

Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data. In *European Conference on Computer Vision (ECCV)*, 2020.

Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Qi Song, Yifei He, Chi Zhang, Zheng Fu, Xuhe Zhao, Mengmeng Yang, Kun Jiang, Rui Huang, and Diange Yang. Envision4D: Envisioning Visual Futures via Feed-forward 4D Gaussian Splatting for Autonomous Driving. *arXiv preprint arXiv:2606.10656*, 2026.

Rustin Soraki, Homanga Bharadhwaj, Ali Farhadi, and Roozbeh Mottaghi. ObjectForesight: Predicting Future 3D Object Trajectories from Human Videos. *arXiv preprint arXiv:2601.05237*, 2026.

Xiangyu Sun, Shijie Wang, Fengyi Zhang, Lin Liu, Caiyan Jia, Ziying Song, Zi Huang, and Yadan Luo. VGGT-World: Transforming VGGT into an Autoregressive Geometry World Model. In *European Conference on Computer Vision (ECCV)*, pp. 382–400, 2026.

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.

Neerja Thakkar, Shiry Ginosar, Jacob Walker, Jitendra Malik, João Carreira, and Carl Doersch. Forecasting Motion in the Wild. *arXiv preprint arXiv:2604.01015*, 2026.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual Geometry Grounded Transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking Everything Everywhere All at Once. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of Motion: 4D Reconstruction from a Single Video. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9660–9672, 2025b.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3D Perception Model with Persistent State. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025c.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jérôme Revaud. DUSt3R: Geometric 3D Vision Made Easy. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point Trajectory Modeling for Policy Learning. In *Robotics: Science and Systems (RSS)*, 2024.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. SpatialTracker: Tracking Any 2D Pixels in 3D Space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. OmniControl: Control Any Joint at Any Time for Human Motion Generation. In *International Conference on Learning Representations (ICLR)*, 2024.

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.

Tomoya Yoshida, Shuhei Kurita, Taichi Nishimura, and Shinsuke Mori. Generating 6DoF Object Manipulation Trajectories from Action Description in Egocentric Vision. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 17370–17382, 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.

Bowei Zhang, Lei Ke, Adam W. Harley, and Katerina Fragkiadaki. TAPIP3D: Tracking Any Point in Persistent 3D Geometry. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026a.

Jianing Zhang, Chenhao Zheng, Yajun Yang, Max Argus, Rustin Soraki, Winson Han, Taira Anderson, Chun-Liang Li, Shuo Liu, Jiafei Duan, Zhongzheng Ren, Jieyu Zhang, and Ranjay Krishna. MolmoMotion: Forecasting Point Trajectories in 3D with Language Instruction. *arXiv preprint arXiv:2606.18558*, 2026b.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. MonST3R: A Simple Approach for Estimating Geometry in the Presence of Motion. In *International Conference on Learning Representations (ICLR)*, 2025b.

Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T. Freeman. Structure and Motion from Casual Videos. In *European Conference on Computer Vision (ECCV)*, 2022.

Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. Guided Conditional Diffusion for Controllable Traffic Simulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning. In *International Conference on Machine Learning (ICML)*, pp. 79115–79135, 2025.

A TRAINING SETTINGS

Table 5 lists the settings of the four training runs. The tokenizer encoder input holds a learned token embedding and the rows of X_k and X_{k+1}, each with a learned source or destination embedding, and two-dimensional rotary position embeddings shared by the two grids. Its blocks are pre-norm transformer blocks with query-key normalization, gated attention, a SwiGLU feedforward layer and layer scale initialized at 10^{-5}, and a final layer normalization of the token row gives z_{k+1}. The decoder concatenates the token with X_k and reads the spatial rows of the output as $\widetilde{X}_{k+1}$. For Point4D, learned type embeddings mark the camera, time and patch rows, and the camera and time rows carry no rotary rotation. Each predictor block projects the keys and values of the history tokens from width d, and prediction queries never attend to one another. Both tokenizers are trained once and frozen before predictor training. The D4RT tokenizer uses rotary coordinate augmentation during training, and both frozen tokenizers later run with fixed coordinates. The validation loss of the Point4D predictor is still falling at update 14,500, the checkpoint we evaluate. The training clips come from the MolmoMotion-1M subsets with reconstructed videos, namely Xperience with 70,595 clips, EgoDex with 63,695, Stereo4D with 19,386, HD-EPIC with 3,050 and DROID with 2.

Table 5: Training settings of the tokenizers and predictors.

Setting	D4RT	Point4D
State per step	256 tokens × 1,280, two-frame tubelet	326 tokens × 1,536, one frame
Encoder input	48 frames, 256×256	48 frames, 252×252
Tokenizer blocks, token	12 + 12, one token of width 1,280	12 + 12, one token of width 1,536
Tokenizer pairs per clip	23	47
Tokenizer optimizer	AdamW, lr 10^{-3}, 5,000 warmup, weight decay 0.01, clip 0.01	
Tokenizer batch	4 GPUs × 5 clips	6 GPUs × 8 clips
Tokenizer updates, time	100,000, about 40 h	10,000, about 37 h
Encoder precision (tokenizer)	bfloat16 autocast	float32
Predictor	12 blocks, width 768, 12 heads, $S = 256$, $\sigma = 0.02$, smooth-L_1 with $\beta = 0.1$	
Window C, targets per clip	4 tokens, 22	8 tokens, 46
Predictor optimizer	AdamW, lr 10^{-4}, 1,000 warmup, weight decay 0.4, clip 0.01	
Predictor batch	4 GPUs × 96 clips	4 GPUs × 34 clips
Updates, evaluated update	10,000, 9,750	100,000 planned, 14,500
Time per update	15.0 s	10.7 s
Encoder precision (predictor)	bfloat16 autocast	bfloat16
Validation	64 held-out clips every 250 updates, 20 sampled futures	

B GEOMETRY ROUND TRIP

Feature loss does not say whether the frozen decoder reads the token content as geometry, so we rebuild D4RT memory slices from tokens, put them back into the memory of Equation 6 and query 3D points. The test uses 3 PointOdyssey clips, 96 tracks and all 48 target frames, with one scale per scene taken from the true-memory condition. The error against ground truth cannot separate the conditions in Table 6, because the reconstruction error of the frozen backbone dominates it. The usable signal is the drift from the prediction that the true memory produces. Correct tokens drift 0.148, less than half of the 0.303 that time-shuffled tokens produce, so the decoder reads the token as the geometry of that transition. Replacing every slice by its predecessor drifts least while having the largest feature error, 29.7% relative squared error against 2.8% for rebuilt slices, so feature loss overstates geometric fidelity. Our post-hoc explanation is that a time-shifted true memory stays on the encoder manifold while a rebuilt slice does not.

C ADDITIONAL POINTMOTIONBENCH RESULTS

Controls and observed-only forecasts. Table 7 scores all selected points with the history-only controls, our observed-only forecasts, their best-of-5 selection and the full-video teacher, which decodes an encoding of the whole real video and is an upper reference for the frozen backbone.

Table 6: Geometry round trip through the frozen D4RT decoder on 3 PointOdyssey clips with 96 tracks, in dataset world units. The first column is the error against ground truth, the second the distance from the true-memory prediction. Read the second column.

Memory condition	Error vs GT ↓	Drift from true memory ↓
True memory	0.519	0.000
All slices rebuilt from correct tokens	0.530	0.148
All slices rebuilt, tokens shuffled in time	0.527	0.303
All slices replaced by the previous slice	0.502	0.044
Only slice 12 rebuilt	0.520	0.016
Only slice 12 replaced by slice 11	0.518	0.012

Hold repeats the last observed position, constant velocity extrapolates the last observed velocity, and damped velocity uses 0.4 times that velocity with the factor fixed before the runs. Best-of-5 draws 5 rollouts and keeps, for every official 8-point record, the one with the lowest ADE against ground truth, which is our reading of the MolmoMotion selection because its released code does not implement it. Hold and the published Static baseline predict the same thing, yet reach HOT3D ADE 0.223 and 0.180, so the two scoring setups alone differ by 0.043 m, twice the published gap between Static and Extrapolate. The ADE of the observed-only forecast stays within 6% of hold for either backbone on every subset, damped velocity beats it everywhere, and best-of-5 lowers D4RT ADE by at most 3.7% and Point4D ADE by at most 0.5%.

Table 7: PointMotionBench on all selected points, ADE and FDE in meters, PWT in percent. †Reads the whole real video. Best-of-5 selects with ground truth.

	HOT3D			WorldTrack			DAVIS		
Method	ADE↓	FDE↓	PWT↑	ADE↓	FDE↓	PWT↑	ADE↓	FDE↓	PWT↑
Hold	0.223	0.369	23.4	0.169	0.300	31.6	2.163	4.138	8.4
Constant velocity	0.201	0.389	28.2	0.195	0.424	39.1	2.362	4.906	10.1
Damped velocity 0.4	0.193	0.333	26.0	0.146	0.293	36.9	1.907	3.792	9.4
D4RT, observed only	0.223	0.370	23.3	0.172	0.306	29.7	2.280	4.282	7.2
D4RT, observed only, best of 5	0.221	0.368	23.7	0.166	0.297	31.0	2.196	4.192	8.1
D4RT, full-video teacher†	0.166	0.274	29.1	0.072	0.122	51.1	1.549	2.767	10.6
Point4D, observed only	0.223	0.370	23.3	0.169	0.301	31.3	2.150	4.130	8.6
Point4D, observed only, best of 5	0.223	0.369	23.4	0.168	0.299	31.5	2.146	4.122	8.6
Point4D, full-video teacher†	0.152	0.243	34.5	0.100	0.173	44.8	1.786	3.102	10.2

Full steering results. Table 8 adds FDE and PWT to Table 4.

D STEERING DETAILS

Numerics. With the default fused attention kernels the D4RT backward pass is not deterministic, and twenty Adam steps through the rollout amplify the small gradient differences into visibly different steered forecasts. All D4RT steering therefore uses deterministic algorithms and a standard Adam implementation, which reproduces bit for bit. Point4D steering uses the same optimizer in eager mode with a fixed seed per scene, and its differentiable rollout reproduces the unsteered forecast exactly.

Cost. A warm unsteered D4RT forecast takes about 1.1 s per scene on one RTX PRO 6000, and 20 steering updates take about 9 s with a captured CUDA graph, after about 5.5 s of capture per process. Point4D steering without graph capture takes about 26 s per scene for 20 updates, including the full-clip encoding.

Table 8: Steering on held-out points with the guide excluded, ADE and FDE in meters, PWT in percent. †Full-clip context. ‡Uses the future track of the guide.

		WorldTrack			DAVIS			HOT3D		
Backbone	Method	ADE↓	FDE↓	PWT↑	ADE↓	FDE↓	PWT↑	ADE↓	FDE↓	PWT↑
D4RT	Observed only	0.171	0.305	29.8	2.282	4.290	7.2	0.223	0.370	23.3
	Unsteered†	0.149	0.270	32.8	2.019	3.784	7.4	0.197	0.331	25.6
	Steered, 5†‡	0.126	0.230	36.8	1.750	3.363	8.8	0.187	0.316	26.7
	Steered, 10†‡	0.120	0.215	38.5	1.691	3.231	9.5	0.182	0.308	27.3
	Steered, 20†‡	0.116	0.203	40.0	1.633	3.121	10.1	0.174	0.295	28.5
Point4D	Observed only	0.168	0.300	31.4	2.151	4.139	8.6	0.223	0.370	23.3
	Unsteered†	0.144	0.266	34.9	1.930	3.499	8.7	0.190	0.314	27.3
	Steered, 5†‡	0.129	0.225	37.8	1.748	3.184	9.7	0.173	0.285	29.9
	Steered, 10†‡	0.132	0.217	38.5	1.688	3.033	10.0	0.165	0.270	31.4
	Steered, 20†‡	0.135	0.224	39.6	1.573	2.825	10.5	0.152	0.247	33.7
	Guide translation‡	0.153	0.243	44.4	0.572	0.893	23.6	0.043	0.071	65.7