KEPLER4D: CONTROLLABLE FUTURE VIDEO GENERATION VIA 4D SCENE STATE EVOLUTION

Anonymous authors
Paper under double-blind review

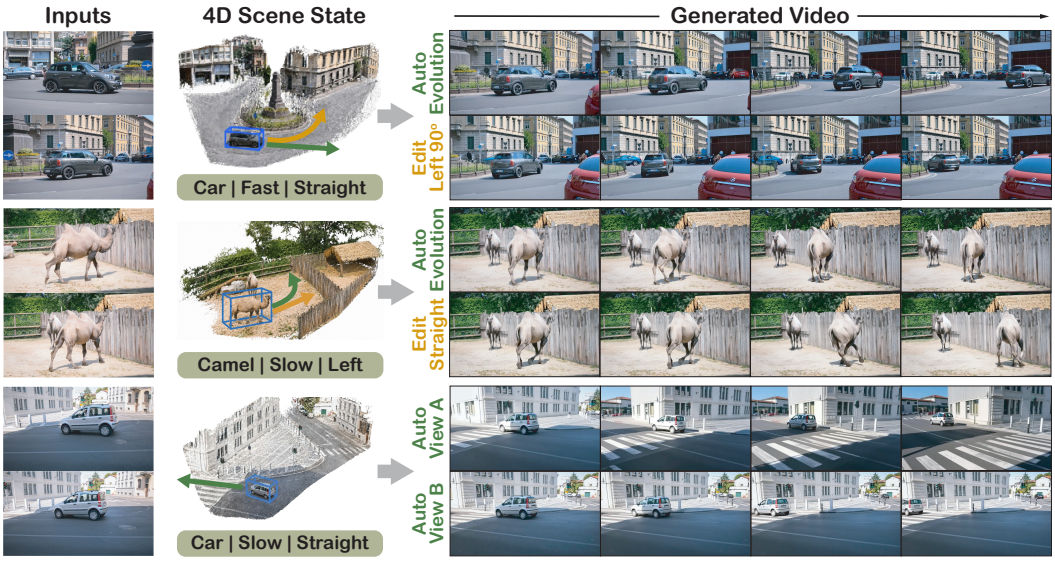

Figure 1: Kepler4D enables future video generation through explicit trajectory evolution and control in a 4D scene state. Our method organizes static geometry, foreground motion histories, and semantic context into the scene state, predicting scene-aware 3D trajectories to guide generation. It supports automatic future evolution, language-guided trajectory editing, and view-consistent rendering across viewpoints, producing plausible future motion while preserving the reconstructed scene structure.

ABSTRACT

Video world models aim to preserve scene structure and predict how dynamic objects evolve beyond visual observations. We present Kepler4D, a framework for future video generation through explicit 4D scene state evolution. Given a monocular video, Kepler4D constructs a shared 3D representation of background geometry, object motion histories, coarse spatial supports, and semantic context. Chain-of-Motion summarizes observed motion and uses a vision-language model to select structured speed and heading decisions and decide whether to bound object-center height from below. A deterministic rollout converts these decisions into future object trajectories for inspection and editing before synthesis. We render the evolving proxies into geometric controls for a pretrained video generator, separating coarse object motion from the synthesis of appearance and articulation. Experiments on real-world videos demonstrate that Kepler4D enables controllable object motion and plausible future rollout while preserving scene consistency.

1 INTRODUCTION

Video world models aim to preserve scene structure and predict plausible future dynamics from visual observations (Chen et al., 2025; Duan et al., 2026; Wang et al., 2025; Wu et al., 2025). Future video generation is valuable for anticipating scene changes and exploring possible outcomes beyond the observed frames, supporting scenario generation for autonomous driving (Hu et al., 2023; Wang et al., 2024a), visual planning for robotic manipulation (Dharmarajan et al., 2026; Zhou et al., 2024), and action-controllable simulation of interactive environments (Bruce et al., 2024; Yang et al., 2023).

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.

Recent video world models have made progress in maintaining scene structure. Reconstructed spatial memory, geometric caches, and rendered controls improve consistency across camera viewpoints (Ren et al., 2025; Wang et al., 2025; Wu et al., 2025). Other methods address dynamic subjects and events that evolve outside the current view (Chen et al., 2026; Duan et al., 2026). Geometry-based controllers also enable camera redirection (Yu et al., 2025) and joint control of camera and object trajectories (Zheng et al., 2026). However, these scene-consistency and geometric-control approaches typically leave future motion to video synthesis or require external trajectories. Joint motion prediction and editing therefore benefit from a shared scene representation that connects observed motion and scene context to explicit future object trajectories and their rendering as video generation controls.

We propose **Kepler4D**, a controllable video world model that observes motion history, predicts scene-aware 3D trajectories, and renders them as motion guidance for future generation. At its core is a structured 4D scene state that organizes static scene geometry, foreground motion histories, coarse spatial supports, and semantic context in a shared world frame. Static geometry anchors scene structure, while foreground motion states provide the basis for predicting future object trajectories. By evolving object centers within this shared 4D state, Kepler4D makes future motion explicit and available for inspection and editing before video synthesis. This formulation supports automatic future evolution, language-guided trajectory editing, and rendering under different camera viewpoints.

Kepler4D combines state construction, motion reasoning, and controllable rendering. First, it lifts the input video into background geometry and an object proxy bank, where each proxy stores semantic identity, an observed 3D trajectory, and coarse spatial support. Second, Chain-of-Motion (CoM) summarizes observed motion and uses a vision-language model to infer structured motion decisions from these summaries and scene context. Deterministic rollout converts decisions into future 3D trajectories for inspection or editing before synthesis. Third, predicted or edited trajectories are rendered with background geometry as geometric control maps under a target camera sequence. These controls guide a pretrained video generator, with background geometry anchoring layout and moving proxies guiding foreground displacement. The generator synthesizes appearance and local articulation; trajectories expose coarse object motion throughout prediction, editing, and synthesis.

We first evaluate static novel view synthesis on DL3DV, followed by automatic dynamic continuation and language-guided trajectory editing on DAVIS to assess scene rendering and future motion control. Experiments demonstrate that Kepler4D enables controllable object motion and plausible future rollout while preserving scene consistency. Ablation studies further show the contributions of observed motion history, semantic context, and rendered trajectory guidance to future video synthesis.

Our contributions are:

- We propose Kepler4D, a controllable video world model that integrates future motion prediction and video synthesis through explicit evolution of one shared 4D scene state.
- We introduce Chain-of-Motion, which uses observed motion and scene context to infer structured motion decisions and expands them into editable future 3D trajectories through deterministic rollout to guide future video generation with explicit object motion control.
- We evaluate automatic evolution, language-guided trajectory editing, and static rendering on real-world videos, ablating motion history, semantic context, and 3D trajectory guidance.

2 Related Work

Video World Models and Geometric Control. Video world models (Bruce et al., 2024; Chen et al., 2025; Duan et al., 2026; Hu et al., 2023; Wang et al., 2025; Wu et al., 2025) aim to simulate scene evolution from visual observations and control signals. Spatial-memory methods use reconstructed geometry to preserve scene structure across generated views (Ren et al., 2025; Wang et al., 2025; Wu et al., 2025). LiveWorld (Duan et al., 2026) and hybrid-memory approaches (Chen et al., 2026) model dynamic events outside the current view. For viewpoint control, camera-conditioned generators encode target poses (He et al., 2025; Wang et al., 2024b), while TrajectoryCrafter (Yu et al., 2025) and FreeOrbit4D (Cao et al., 2026) use reconstructed geometry to redirect camera trajectories in monocular videos. Geometry Forcing improves 3D consistency by aligning intermediate diffusion features with pretrained geometric features during training (Wu et al., 2026). VerseCrafter supports joint camera and object control by rendering background geometry and per-object 3D Gaussian

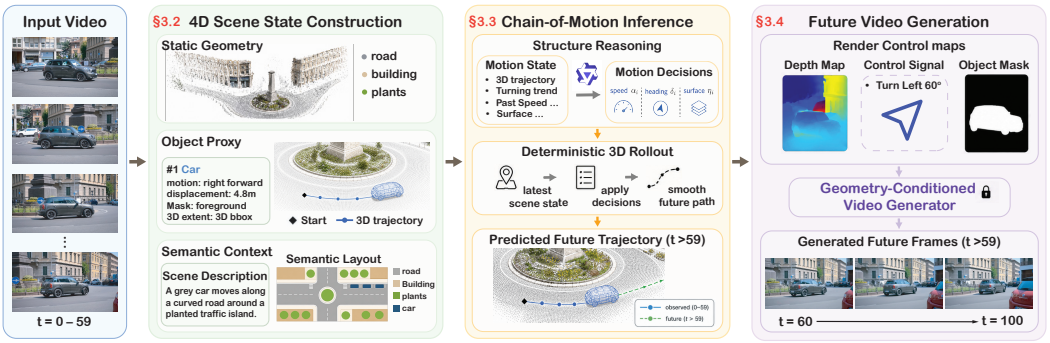

Figure 2: **Overview of Kepler4D.** Given an observed video, Kepler4D constructs a structured 4D scene state comprising static scene geometry, foreground motion histories, and semantic context. Chain-of-Motion reasons over the observed motion and scene context to infer structured motion decisions, which a deterministic 3D rollout expands into future object trajectories. These trajectories are rendered with static geometry into depth maps, object control signals, and masks to guide a pretrained video generator. This design separates explicit object motion prediction from video synthesis, enabling controllable generation of future videos while preserving the scene structure.

trajectories into conditioning maps (Zheng et al., 2026). Kepler4D renders editable 3D trajectories predicted from observed motion and scene context as geometric controls for future video generation.

Motion Reasoning and Future Prediction. Motion prediction methods (Moon et al., 2024; Seff et al., 2023; Yang et al., 2025a;b; Zhang et al., 2026) forecast trajectories from scene observations, motion histories, or language instructions. MotionLM formulates multi-agent forecasting as autoregressive prediction of discrete motion tokens (Seff et al., 2023), while VisionTrap incorporates visual features and textual supervision into trajectory prediction (Moon et al., 2024). VLIPP (Yang et al., 2025b) and TrajVLM-Gen (Yang et al., 2025a) use VLMs to predict coarse motion that guides video synthesis. SeeU learns continuous 4D dynamics from reconstructed observations to generate videos at unobserved times and viewpoints (Yuan et al., 2026a). MolmoMotion predicts 3D point trajectories from visual history and language goals for downstream control and generation (Zhang et al., 2026), while OptiWorld optimizes trajectories for task goals under physical constraints (Yuan et al., 2026b). Kepler4D combines structured VLM decisions with deterministic 3D rollout to predict scene-aware, editable future object-center trajectories, enabling automatic future evolution and controllable video generation. These trajectories provide direct object motion controls before future video synthesis.

3 Methodology

Kepler4D reconstructs an observed scene in 3D and predicts future object motion to guide controllable future video generation. As shown in Fig. 2, given an observed video $\mathcal{V}^{\text{src}} = \{I_t\}_{t=0}^{T_o-1}$, we construct a shared 3D representation of static scene geometry, foreground object proxies, and semantic context (§ 3.1, § 3.2). Chain-of-Motion then reasons over motion history and scene context to predict 3D object trajectories for the next T_f frames (§ 3.3). Finally, we render the predicted or edited trajectories with static geometry, producing geometric control maps that guide future video synthesis (§ 3.4).

3.1 4D Scene State and Its Evolution

Our goal is to construct a proxy world from an observed video and use its predicted evolution to guide future video generation. The representation combines scene geometry for locating and rendering objects with motion histories and semantic context for predicting their motion. We express geometry and motion in a shared reconstructed 3D coordinate frame and define the 4D scene state as

$$\mathcal{S} = (\mathcal{G}, \mathcal{P}, \mathcal{A}), \qquad \mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{N}. \tag{1}$$

Here, $\mathcal{G}$ stores static background geometry, $\mathcal{P}$ contains N foreground object proxies, and $\mathcal{A}$ stores semantic scene context. The background geometry provides a spatial reference for object trajectories and scene rendering. Each proxy $\mathbf{p}_i$ records a semantic label ℓ_i, an observed 3D center history $\mathcal{H}_i$,

and a coarse spatial extent $\mathbf{e}_i$. The center history informs motion prediction, while the spatial extent defines an ellipsoid for rendering the proxy. Scene descriptions and local layout cues in $\mathcal{A}$ provide additional context for predicting object motion. Together, these components provide a common representation for predicting future object motion and rendering the predicted future evolution.

To represent scene changes beyond the observed video, we evolve the proxy world into a sequence of future states by updating object centers along predicted 3D trajectories. Given T_o observed frames and a prediction horizon of T_f frames, the state at each future frame $t = T_o, \ldots, T_o + T_f - 1$ is

$$\widehat{\mathcal{S}}_t = \left(\mathcal{G}, \{(\ell_i, \widehat{\mathbf{c}}_i^t, \mathbf{e}_i)\}_{i=1}^N, \mathcal{A}\right), \tag{2}$$

where $\widehat{\mathbf{c}}_i^t$ is the predicted center of object i at time t. Background geometry, semantic context, and object labels and extents remain fixed throughout the predicted state sequence. Rendering the predicted or edited state sequence under target camera poses converts future scene evolution into geometric controls for video generation. These controls encode scene layout and object displacement, while the video generator synthesizes fine appearance and local articulation over the future frames.

3.2 Scene State Construction

From the observed video, we recover scene geometry and identify foreground objects for motion prediction. A shared 3D coordinate frame describes their motion relative to the scene across frames. Specifically, a pretrained geometry estimator (Lin et al., 2026) predicts per-frame depth D_t, camera intrinsics K_t for image projection, and camera poses $\mathcal{C}_t$ specifying camera position and orientation.

Using this geometric reference, we identify objects and extract scene context for motion prediction. A vision-language model (Bai et al., 2025) processes the first and last observed frames to produce object labels, bounding boxes, and a scene description. The labels define ℓ_i, while the scene description and local layout cues form $\mathcal{A}$. We use the bounding boxes to initialize SAM 2 (Ravi et al., 2025), which tracks object regions across frames to produce a mask $\mathcal{M}_i^t$ identifying pixels of object i at time t.

To recover object motion in 3D, we combine these masks with the estimated depth and camera parameters to map object pixels into the shared coordinate frame. Denoting this mapping by Π^{-1}, we estimate each proxy center as the mean of the visible object points reconstructed within its mask:

$$\mathbf{c}_i^t = \frac{1}{|\mathcal{M}_i^t|} \sum_{u \in \mathcal{M}_i^t} \Pi^{-1}(u, D_t(u), K_t, \mathcal{C}_t)\,. \tag{3}$$

Here, u denotes a pixel location and $|\mathcal{M}_i^t|$ is the number of pixels in the object mask. These observed centers form the motion history $\mathcal{H}_i$ for future trajectory prediction. We estimate the spatial extent $\mathbf{e}_i$ from reconstructed object points to define the ellipsoid rendered at each predicted future position.

Finally, we construct static background geometry from multiple input frames to retain the observed surroundings during future object motion. Using the same mapping, we transform pixels outside the foreground masks into the shared 3D coordinate frame, and aggregate the resulting points into $\mathcal{G}$. This aggregation incorporates background regions visible in different frames into the scene used for future rendering. Together, these steps construct $\mathcal{S}$, providing motion histories and semantic context for subsequent trajectory prediction and scene geometry for rendering predicted scene evolution.

3.3 Chain-of-Motion Inference

To evolve the reconstructed scene, we use motion histories to describe recent movement and scene context to infer how it may continue or change. Inspired by structured spatial reasoning (Zhen et al., 2026), we introduce Chain-of-Motion (CoM), which first extracts motion attributes, then infers motion decisions, and finally computes future 3D trajectories. These decisions translate semantic reasoning about future behavior into numerical position updates for rendering the future 3D scene.

Motion-state extraction. To support motion reasoning, we encode each object's recent movement and local scene conditions in a compact motion state $\mathbf{m}_i$. The function Ψ_{state} extracts these attributes from the object proxy $\mathbf{p}_i$ and semantic context $\mathcal{A}$ for subsequent object motion prediction:

$$\mathbf{m}_i = \Psi_{\text{state}}(\mathbf{p}_i, \mathcal{A}) = \left[\mathbf{c}_i^{T_o-1}, \mathbf{v}_i, s_i, \mathbf{a}_i, \kappa_i, h_i, \mathbf{e}_i, q_i\right], \tag{4}$$

where $\mathbf{c}_i^{T_o-1}$ is the latest observed center, while $(\mathbf{v}_i, s_i, \mathbf{a}_i, \kappa_i)$ denote velocity, speed, acceleration, and turning trend estimated from $\mathcal{H}_i$. The remaining entries are the height threshold h_i used by the optional lower-bound constraint, object extent $\mathbf{e}_i$, and semantic cues q_i about the category or surroundings. These attributes relate recent motion to the scene for subsequent motion decisions.

Structured motion decisions. Given $\mathbf{m}_i$, a pretrained VLM interprets recent movement in its scene context to infer future behavior. We express this behavior as structured decisions about object speed, motion direction, and whether to apply a lower height bound. With an optional language instruction ℓ, the inference module Φ_{CoM} produces these decisions together with a textual motion description:

$$(\alpha_i, \delta_i, \eta_i, d_i) = \Phi_{\mathrm{CoM}}(\mathbf{m}_i, \mathcal{A}, \ell), \tag{5}$$

where α_i specifies whether to maintain, increase, or decrease speed, while δ_i selects whether to continue in the current direction or turn. The decision η_i determines whether to apply a lower bound to the predicted object-center height during trajectory computation. The motion description d_i guides video synthesis, whereas structured decisions govern numerical position updates. These decisions can also be edited to specify alternative future object motion within the same reconstructed scene.

Deterministic trajectory rollout. To render the evolving scene, we convert these decisions into a 3D object position for each future frame. Deterministic trajectory rollout applies fixed numerical update rules from the latest observed position and velocity. Specifically, Γ_{motion} maps the motion decisions to numerical parameters $\mathbf{u}_i$, which $\mathcal{I}_{\mathrm{traj}}$ uses to compute positions over the next T_f frames:

$$\mathbf{u}_i = \Gamma_{\mathrm{motion}}\left(\alpha_i, \delta_i, \eta_i, h_i\right), \qquad \widehat{\mathcal{T}}_i = \mathcal{I}_{\mathrm{traj}}\left(\mathbf{c}_i^{T_o-1}, \mathbf{v}_i, \mathbf{u}_i; T_f\right), \tag{6}$$

At each step, speed and heading decisions adjust the displacement used to update the object center. When enabled, the height constraint imposes h_i as a lower bound on the predicted object-center height. We then apply temporal smoothing to reduce abrupt changes between successive predicted positions. The resulting trajectory $\widehat{\mathcal{T}}_i = \{\widehat{\mathbf{c}}_i^t\}_{t=T_o}^{T_o+T_f-1}$ supplies the centers in Eq. 2, converting predicted or edited motion into future scene configurations for rendering under target camera poses.

3.4 Future Video Generation

We generate future video by rendering the evolving proxy world under target camera poses and using the resulting controls to guide video synthesis. The future scene states specify scene layout and object motion, while the video generator synthesizes fine appearance details absent from the coarse proxy representation. We adopt the geometric control representation and pretrained video generation components of VerseCrafter (Zheng et al., 2026) to guide future synthesis from the scene states.

For each future state, we render complementary background and foreground controls under the corresponding target camera pose. The background controls encode the appearance and depth of the static scene. Foreground objects are represented by coarse 3D Gaussian ellipsoids whose centers follow the predicted or edited trajectories and whose sizes are determined by the proxy supports. Rendering these proxies produces foreground color, depth, and motion masks that indicate where objects should appear in each future frame. Both sets of controls are derived from the shared world frame, allowing camera motion and object motion to jointly guide synthesis across all future frames. Here, changing the target camera pose changes the projection of the reconstructed state, whereas editing an object trajectory changes its center positions within that state. The same explicit motion plan can therefore be rendered under alternative target camera paths, and each object-motion edit remains defined in the shared scene rather than in the image coordinates of a particular camera view.

To preserve visual continuity, we use the last observed frame I_{T_o-1} as an appearance reference for the scene and its foreground objects. We also construct a text prompt p_{text} from CoM's motion descriptions d_i to complement the geometric controls with semantic guidance for object motion. The generator combines these appearance, geometric, and motion cues to generate the future video:

$$\widehat{\mathcal{V}} = \Phi_{\mathrm{VG}}\Big(I_{T_o-1}, \mathcal{R}\Big(\{\widehat{\mathcal{S}}_t\}_{t=T_o}^{T_o+T_f-1}, \mathcal{C}^{\mathrm{tar}}\Big), p_{\mathrm{text}}\Big). \tag{7}$$

Here, $\mathcal{R}$ renders the background and foreground controls for each state under the target camera sequence $\mathcal{C}^{\mathrm{tar}}$, and Φ_{VG} is the pretrained generator. The implementation appendix summarizes the model components and inference settings used here. By evolving the proxy world, Kepler4D supports automatic video continuation, while trajectory edits allow users to specify alternative object motion.

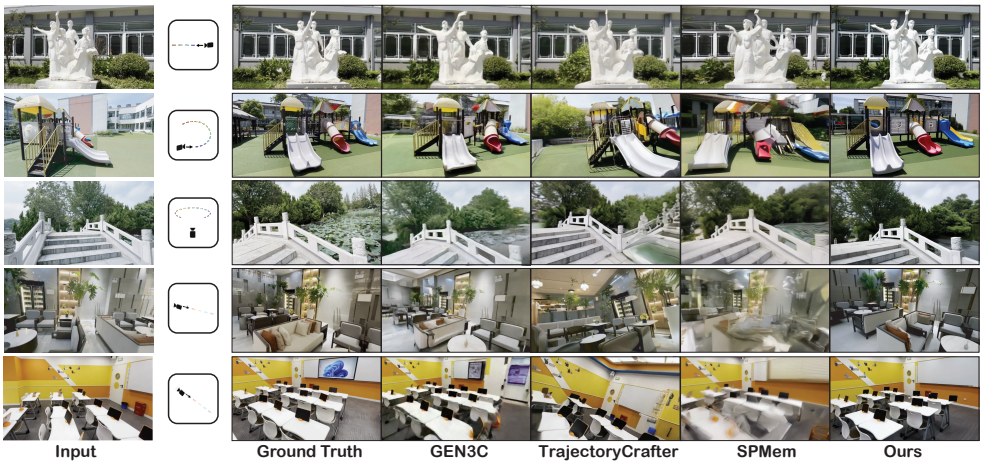

Figure 3: **Static novel view synthesis on DL3DV.** We compare Kepler4D with GEN3C, TrajectoryCrafter, and SPMem on indoor and outdoor scenes under target camera motion. Each row shows an input view, the target camera path, the held-out reference view, and the generated results. In these examples, Kepler4D preserves the scene layout and appearance details across viewpoint changes.

4 EXPERIMENTS

We first introduce the datasets and evaluation metrics. We then evaluate Kepler4D across three settings: static view synthesis in § 4.1, automatic future video generation in § 4.2, and controllable future video generation in § 4.3. These experiments assess scene rendering under camera motion, continuation of observed object dynamics, and control over future object movement, respectively.

Datasets. We evaluate static view synthesis on the 55-scene DL3DV-Evaluation release (Ling et al., 2024), using the first 49 frames as observations and the provided camera trajectories for future viewpoints. Dynamic continuation and motion editing draw on the 90 sequences of the DAVIS 2017 train and validation splits (Perazzi et al., 2016; Pont-Tuset et al., 2017). For automatic continuation, we split 38 selected sequences from this pool into observed prefixes and held-out continuations. We estimate camera trajectories from the full sequences with Depth Anything 3 (Lin et al., 2026) and use future poses as external rendering conditions. Within each setting, all methods share observed frames; camera-controllable methods share the target camera path. Appendix B describes the dataset usage and temporal alignment procedures for these evaluations and the role of the target camera paths.

Evaluation Metrics. We evaluate reference similarity, visual quality, and motion accuracy. PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) measure similarity to held-out frames. FID (Heusel et al., 2017) compares feature distributions of generated and reference frames. VBench (Huang et al., 2024) assesses visual quality, temporal consistency, and motion activity. Dynamic Degree measures activity, not accuracy. For dynamic continuation, Flow-EPE measures the full-image optical-flow difference between generated and reference videos, reflecting camera and object motion, while RotErr measures camera rotation error. For motion editing, we report the camera-yaw-based TurnErr proxy and VBench Imaging Quality. TurnErr reflects camera turning, not object heading independent of camera motion. PSNR measures similarity to the recorded continuation, which may not reflect the requested motion. Appendix C provides the metric definitions and evaluation protocols used here and explains the interpretation of each of the reported scores.

4.1 STATIC NOVEL VIEW SYNTHESIS

We first evaluate the static setting on the DL3DV dataset, where foreground evolution is absent and the task reduces to rendering a static scene under future camera viewpoints. When no dynamic foreground object is detected, Kepler4D uses an empty object proxy bank and relies solely on the reconstructed background state for geometry-conditioned video synthesis. This setting evaluates background reconstruction and camera-conditioned generation, without testing CoM motion reasoning itself.

Table 1: **Static view synthesis on DL3DV (Ling et al., 2024).** We compare Kepler4D with three baselines using reference fidelity, FID, and VBench (Huang et al., 2024) metrics. We also report peak GPU memory and per-clip inference time under the evaluated model and inference configurations.

	Reconstruction Fidelity			Generation Quality					Efficiency	
Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Imag. ↑	MotSm. ↑	BgC. ↑	Overall ↑	Mem. (GB)	Time (min)
GEN3C Ren et al. (2025)	11.67	0.2963	0.5706	63.99	0.6991	**0.9863**	0.9314	0.8041	54.73	43
TrajectoryCrafter Yu et al. (2025)	11.90	0.3114	0.5791	74.23	0.6709	0.9712	0.9372	0.8005	37.81	10.4
SPMem Wu et al. (2025)	13.19	0.3611	0.5273	56.92	0.5819	0.9776	0.9134	0.8175	45.57	9.7
Ours	**13.55**	**0.3675**	**0.3995**	**41.37**	**0.7606**	0.9745	**0.9448**	**0.8322**	62.46	5.2

Table 2: **Automatic evolution on DAVIS (Pont-Tuset et al., 2017).** All methods receive the same observed video and camera path. We report reconstruction fidelity, generation, and motion fidelity.

	Reconstruction Fidelity			Generation Quality				Motion Fidelity	
Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Imag. ↑	DynDeg ↑	Overall ↑	Flow-EPE ↓	RotErr ↓
TrajectoryCrafter Yu et al. (2025)	14.349	0.3593	0.5145	214.13	0.6595	0.1622	0.7064	17.409	7.5400
SPMem Wu et al. (2025)	14.641	0.3837	0.5102	265.13	0.5870	**0.7838**	0.7832	20.138	18.549
VerseCrafter Zheng et al. (2026)	14.090	0.3649	**0.4556**	303.80	0.6896	0.6486	0.7773	15.400	8.9610
Ours	**14.921**	**0.3924**	0.4561	**202.64**	**0.6970**	**0.7838**	**0.8029**	**15.220**	**5.0200**

Results. As shown in Figure 3, in indoor and outdoor scenes, Kepler4D better preserves scene layout and fine textures under target camera motion for wide-baseline views, where baselines show geometric distortion, surface inconsistency, or texture drift. These results support our static background representation and geometry-conditioned rendering for static novel view synthesis.

Table 1 shows competitive overall static generation results for Kepler4D. Reconstruction gains indicate that scene-state rendering better preserves scene structure in the target views, while lower LPIPS and FID suggest that the generated views retain finer appearance details and visual realism.

Although GEN3C achieves the highest Motion Smoothness, Kepler4D performs better on most reconstruction and VBench metrics, balancing viewpoint accuracy, visual quality, and scene consistency under target camera motion. Efficiency results indicate a runtime–memory trade-off under the evaluated configurations. Kepler4D uses 62.46 GB of peak GPU memory, compared with 37.81–54.73 GB for the baselines, while also recording the lowest reported per-clip inference time of 5.2 minutes.

4.2 Automatic Dynamic Evolution

We next evaluate dynamic world evolution, where the task is to continue foreground motion from a monocular video while rendering the scene along a target future camera path. Experiments are conducted on DAVIS, where all methods receive the same observed video and target camera trajectory, but no object trajectories, mask sequences, or language motion commands. This setting tests whether object evolution can be inferred from visual observations without externally specified motion control.

Results. Table 2 evaluates Kepler4D in a challenging setting where foreground motion is inferred from the observed video without object trajectories or language motion commands. Kepler4D obtains favorable motion-fidelity scores while maintaining competitive reconstruction and generation quality. The lower Flow-EPE indicates closer overall image motion to the reference video, while the lower RotErr indicates more accurate following of the target camera rotation. The reconstruction and perceptual scores show that the complete pipeline maintains competitive visual quality during automatic continuation. SPMem maintains competitive pixel-level fidelity but has higher image-motion and camera-rotation errors in this evaluation. VerseCrafter achieves the lowest LPIPS, while its Flow-EPE and RotErr are higher than those of Kepler4D. Together, these results support explicit trajectory guidance within the complete pipeline for automatic future generation from video inputs.

Figure 4 provides a qualitative comparison of these differences. The observed video already shows a parkour runner vaulting over a low wall. The future scene contains a railing along the runner's route, and another vault is a plausible continuation given the observed activity and layout. The reference video and Kepler4D both show the person moving forward and vaulting over the railing. TrajectoryCrafter shows a viewpoint change without a vault, SPMem shows apparent intersections between the person and railing, and VerseCrafter shows a lateral deviation before the vault. This example illustrates a plausible continuation produced by the full pipeline: predicted center trajectories provide coarse motion guidance, while the video generator synthesizes appearance and local articulation.



Figure 4: **Automatic future evolution.** All methods use the input video without externally supplied object trajectories or motion commands. Methods with camera control are given the same target camera path. Kepler4D depicts the runner moving forward and vaulting over the railing, as in the reference. TrajectoryCrafter shows a viewpoint change without a vault. SPMem shows apparent intersections between the runner and railing. VerseCrafter shows a lateral deviation before the vault.

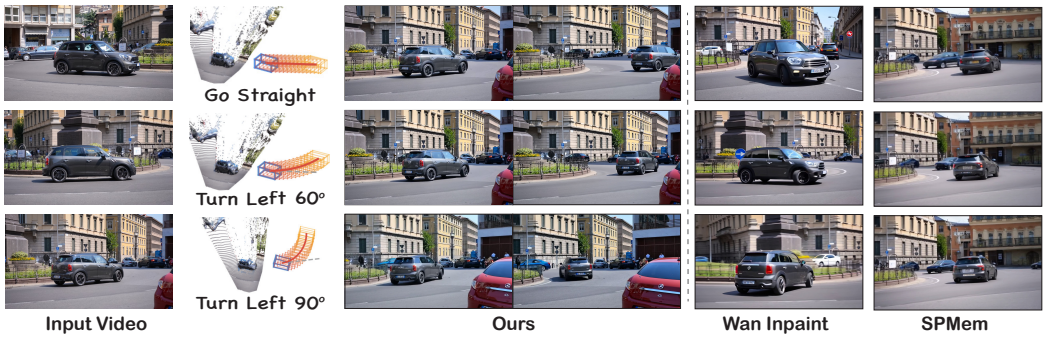

Figure 5: **Language-guided controllable future generation.** Given the same input video, the user specifies the intended motion through a high-level language command. Kepler4D infers an object-level trajectory from the command, shown in the second column. Baselines receive the same language command but not the inferred trajectory or control maps. With explicit trajectory control, Kepler4D better realizes the intended object motion while preserving scene layout and consistency.

4.3 Controllable Dynamic Generation

We further evaluate language-guided dynamic control on DAVIS. Unlike § 4.2, where motion is inferred without user guidance, this setting provides high-level commands such as going straight or turning left by a desired angle. Kepler4D converts these commands into object-level motion plans and renders the edited scene state using geometry-conditioned control maps. Baselines receive the same language commands, but not Kepler4D's generated trajectory visualizations or control maps.

Results. Figure 5 illustrates distinct future generations from the same input video under different language commands. Kepler4D translates these commands into object-level trajectories and renders them as spatial guidance, producing command-dependent changes in the generated motion. Wan Inpaint (Team Wan et al., 2025) and SPMem receive the same commands without these rendered trajectory controls and exhibit different motion responses in the displayed examples. Table 3 reports the highest PSNR and imaging quality for Kepler4D, together with the lowest TurnErr under the stated evaluation protocol. These scores do not directly measure future object-heading accuracy.

Table 3: **Language-guided dynamic control under identical inputs.** All methods receive the same observed video and language command.

Method	PSNR↑	TurnErr↓	Imag.↑	DynDeg↑
Wan Inpaint	11.52	278.52	0.7177	**0.831**
SPMem	15.18	127.49	0.6855	0.774
Ours	**15.27**	**85.720**	**0.7970**	0.792

Table 4: **Ablation on trajectory-conditioned control.** We compare language prompting, CoM reasoning without and with trajectory rendering.

Variant	SSIM↑	PSNR↑	FID↓	Imag.↑
Qwen prompt	0.2622	15.175	79.53	0.747
CoM w/o traj.	0.2598	15.106	60.49	0.761
CoM w/ traj.	**0.2692**	**15.189**	**48.76**	**0.765**

Table 5: **Trajectory perturbation study.** We perturb the GT trajectory to evaluate the robustness of trajectory-conditioned video rendering.

Control	AngleErr↓	SurfErr↓	Imag.↑	DynDeg↑
GT traj.	**5.78**	**0.040**	**0.6939**	**0.7862**
Left 60°	11.37	0.048	0.6815	0.7562
Down 60°	26.92	0.056	0.6848	0.7116

Table 6: **Scene state component ablation.** We remove each scene-state component to assess its role in motion reasoning and video synthesis.

Variant	Flow-EPE↓	SurfErr↓	DynDeg↑	Imag.↑
w/o Motion Hist.	18.742	0.049	0.6421	0.6954
w/o Sem. Context	16.831	0.057	0.7616	0.6918
w/o Surface Cue	15.364	0.054	0.7593	0.6872
Full	**15.220**	**0.043**	**0.7838**	**0.6970**

Effect of motion conditioning. We examine what motion descriptions and rendered trajectories contribute to generation. The Qwen Prompt baseline uses the scene description from scene-state construction, which describes the observed objects within their surroundings. CoM w/o traj. instead provides a motion description that tells the generator how those objects are expected to move. The CoM w/ traj. variant in this comparison additionally renders the predicted trajectories, specifying where the objects should appear within the scene over future frames. This gives the generator spatial guidance to accompany the motion description. Table 4 shows that CoM's motion description improves FID and imaging quality over scene prompting, with further gains after adding rendered trajectories. These results suggest that describing future movement provides guidance for synthesis, while specifying its spatial progression through rendered trajectories contributes beyond text alone.

4.4 Ablation Study

Trajectory perturbation study. We evaluate whether coarse trajectory-centered control can steer trajectory-conditioned rendering by perturbing the ground-truth future 3D trajectory with the input video, target camera path, and generator fixed. As shown in Table 5, GT trajectory control yields the lowest AngleErr and SurfErr and the highest imaging quality and dynamic degree. AngleErr increases from 5.78 to 11.37 when rotating the trajectory 60° left and to 26.92 under the downward perturbation, indicating reduced adherence to the specified control directions. SurfErr also increases from 0.040 to 0.048 and 0.056, respectively, while imaging quality remains below that of the unperturbed control. These results characterize how directional adherence and synthesis quality change under trajectory perturbations, with the downward perturbation producing the largest reported directional error.

Scene state component ablation. We ablate components of the 4D scene state while keeping the same input video, target camera path, and rendering pipeline. As shown in Table 6, removing motion history causes the largest increase in Flow-EPE and drop in DynDeg. Removing semantic context also increases Flow-EPE. Removing surface cues has a smaller effect on Flow-EPE, but increases SurfErr and reduces imaging quality. Within the evaluated generation pipeline, these comparisons support using motion history, semantic context, and surface cues together to guide future generation.

5 Conclusion

Kepler4D presents a structured approach to dynamic video world modeling by representing future evolution as explicit object-centric 3D trajectories inferred from monocular observations. Its 4D scene state separates scene structure, foreground motion, and semantic context into a shared, editable intermediate representation, while Chain-of-Motion reasoning and deterministic rollout expose future dynamics as interpretable control signals before rendering. By combining trajectory-grounded reasoning with geometry-conditioned video generation, Kepler4D supports controllable future object motion under target camera paths, while the video generator supplies appearance details and local articulation. Experiments across static rendering, automatic dynamic evolution, and language-guided control demonstrate the utility of an explicit motion representation for future video generation.

AI USE STATEMENT

Generative AI assistants were used only to polish the manuscript for clarity, grammar, and presentation. The research ideas, methods, experiments, and conclusions were developed and verified by the authors. Pretrained vision-language models used as components of our method are explicitly described in the corresponding methodology and implementation sections. All AI-assisted manuscript edits were reviewed by the authors, who take responsibility for the final content of this work.

ETHICS STATEMENT

Our experiments use existing public research datasets and do not involve new human-subject data collection. Kepler4D is developed for research on controllable future video generation and scene evolution. As generated videos represent possible continuations rather than verified future events, they should be interpreted accordingly. Applications in safety-critical or real-world decision-making settings would require additional validation and appropriate safeguards.

REPRODUCIBILITY STATEMENT

We provide implementation and evaluation details in the appendix, including model configurations, scene-state construction, trajectory rollout, geometric controls, evaluation protocols, and metric definitions. We will release the code and evaluation to support reproduction and further research.

REFERENCES

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Maria Elisabeth Bechtle, Feryal Behbahani, Stephanie C.Y. 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 *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4603–4623. PMLR, 2024.

Wei Cao, Hao Zhang, Fengrui Tian, Yulun Wu, Yingying Li, Shenlong Wang, Ning Yu, and Yaoyao Liu. Freeorbit4d: Training-free arbitrary camera redirection for monocular videos via foreground-complete 4d reconstruction. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12, 2026.

Junyi Chen, Haoyi Zhu, Xianglong He, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Yang Zhou, Zizun Li, Zhoujie Fu, Jiangmiao Pang, et al. DeepVerse: 4D autoregressive video generation as a world model. *arXiv preprint arXiv:2506.01103*, 2025.

Kaijin Chen, Dingkang Liang, Xin Zhou, Yikang Ding, Xiaoqiang Liu, Pengfei Wan, and Xiang Bai. Out of sight but not out of mind: Hybrid memory for dynamic video world models. *arXiv preprint arXiv:2603.25716*, 2026.

Karthik Dharmarajan, Wenlong Huang, Jiajun Wu, Li Fei-Fei, and Ruohan Zhang. Dream2Flow: Bridging video generation and open-world manipulation with 3D object flow. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

Zicheng Duan, Jiatong Xia, Zeyu Zhang, Wenbo Zhang, Gengze Zhou, Chenhui Gou, Yefei He, Feng Chen, Xinyu Zhang, and Lingqiao Liu. LiveWorld: Simulating out-of-sight dynamics in generative video world models. In *European Conference on Computer Vision (ECCV)*, 2026.

Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. CameraCtrl: Enabling camera control for video diffusion models. In *International Conference on Learning Representations (ICLR)*, pp. 100433–100464, 2025.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, pp. 6626–6637, 2017.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self Forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, volume 38, pp. 167283–167308, 2025. doi: 10.52202/085713-5576.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818, 2024.

LightX2V Contributors. LightX2V: Light video generation inference framework. GitHub repository: `https://github.com/ModelTC/LightX2V`, 2025.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *International Conference on Learning Representations (ICLR)*, pp. 141261–141285, 2026.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, Xuanmao Li, Xingpeng Sun, Rohan Ashok, Aniruddha Mukherjee, Hao Kang, Xiangrui Kong, Gang Hua, Tianyi Zhang, Bedrich Benes, and Aniket Bera. DL3DV-10K: A large-scale scene dataset for deep learning-based 3D vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22160–22169, 2024.

Seokha Moon, Hyun Woo, Hongbeen Park, Haeji Jung, Reza Mahjourian, Hyung-gun Chi, Hyerin Lim, Sangpil Kim, and Jinkyu Kim. VisionTrap: Vision-augmented trajectory prediction guided by textual descriptions. In *European Conference on Computer Vision (ECCV)*, pp. 361–379, 2024. doi: 10.1007/978-3-031-72658-3_21.

Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 724–732, 2016.

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.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *International Conference on Learning Representations (ICLR)*, pp. 28085–28128, 2025.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. GEN3C: 3D-informed world-consistent video generation with precise camera control. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6121–6132. IEEE, 2025.

Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S. Refaat, Rami Al-Rfou, and Benjamin Sapp. MotionLM: Multi-agent motion forecasting as language modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8579–8590, 2023.

Team Wan et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, pp. 402–419, 2020. doi: 10.1007/978-3-030-58536-5_24.

Jiahao Wang, Luoxin Ye, TaiMing Lu, Junfei Xiao, Jiahan Zhang, Yuxiang Guo, Xijun Liu, Rama Chellappa, Cheng Peng, Alan Yuille, et al. EvoWorld: Evolving panoramic world generation with explicit 3D memory. *arXiv preprint arXiv:2510.01183*, 2025.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. DriveDreamer: Towards real-world-driven world models for autonomous driving. In *European Conference on Computer Vision (ECCV)*, pp. 55–72, 2024a. doi: 10.1007/978-3-031-73195-2_4.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024b.

Haoyu Wu, Diankun Wu, Tianyu He, Junliang Guo, Yang Ye, Yueqi Duan, and Jiang Bian. Geometry Forcing: Marrying video diffusion and 3D representation for consistent world modeling. In *International Conference on Learning Representations*, pp. 119186–119209, 2026.

Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video world models with long-term spatial memory. In *Advances in Neural Information Processing Systems*, volume 38, pp. 49371–49393, 2025. doi: 10.52202/085713-1651.

Fan Yang, Zhiyang Chen, Yousong Zhu, Xin Li, and Jinqiao Wang. From seeing to predicting: A vision-language framework for trajectory forecasting and controlled video generation. *arXiv preprint arXiv:2510.00806*, 2025a.

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

Xindi Yang, Baolu Li, Yiming Zhang, Zhenfei Yin, Lei Bai, Liqian Ma, Zhiyong Wang, Jianfei Cai, Tien-Tsin Wong, Huchuan Lu, and Xu Jia. VLIPP: Towards physically plausible video generation with vision and language informed physical prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12360–12370, 2025b.

Mark Yu, Wenbo Hu, Jinbo Xing, and Ying Shan. TrajectoryCrafter: Redirecting camera trajectory for monocular videos via diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 100–111, 2025.

Yu Yuan, Tharindu Wickremasinghe, Zeeshan Nadir, Xijun Wang, Yiheng Chi, and Stanley H. Chan. SeeU: Seeing the unseen world via 4D dynamics-aware generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11150–11162, 2026a.

Yu Yuan, Jianhao Yuan, Xijun Wang, Daiqing Li, Liu He, Lu Ling, and Stanley H. Chan. OptiWorld: Optimal control for video world generation under physical constraints. *arXiv preprint arXiv:2606.00499*, 2026b.

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*, 2026.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018.

Haoyu Zhen, Xiaolong Li, Yilin Zhao, Han Zhang, Sifei Liu, Kaichun Mo, Chuang Gan, and Subhashree Radhakrishnan. 3D-Layout-R1: Structured reasoning for language-instructed spatial editing. In *European Conference on Computer Vision (ECCV)*, 2026.

Sixiao Zheng, Minghao Yin, Wenbo Hu, Xiaoyu Li, Ying Shan, and Yanwei Fu. VerseCrafter: Dynamic realistic video world model with 4D geometric control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 40277–40290, 2026.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61885–61896. PMLR, 2024.

TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

This appendix provides additional details on the components, motion interface, geometric controls, and evaluation of Kepler4D. We retain the notation of the main paper: $\mathcal{S} = (\mathcal{G}, \mathcal{P}, \mathcal{A})$ denotes the 4D scene state, $\mathcal{H}_i$ is the observed center history of object i, $\mathbf{e}_i$ is its coarse spatial extent, and $\widehat{\mathcal{T}}_i$ denotes its future center trajectory. Observed frames are indexed by $t = 0, \ldots, T_o - 1$, and the future horizon contains T_f frames.

APPENDIX CONTENTS

A Implementation Details 15
- A.1 Pretrained Components and Inference Configuration 15
- A.2 Geometry, Foreground Identification, and Spatial Support 15
- A.3 Motion Summaries and Structured Decisions 16
- A.4 Speed and Heading Updates in Deterministic Rollout 17
- A.5 Geometric Control Representation 18

B Dataset Usage and Evaluation Settings 18

C Evaluation Metrics 19
- C.1 Metrics by Evaluation Setting 19
- C.2 Reference Similarity and Perceptual Quality 19
- C.3 Image Motion and Camera Motion 20
- C.4 Control-Related Measurements 20

D Baseline Scope and Ablation Interpretation 21
- D.1 Motion Description and Trajectory Guidance 21
- D.2 Trajectory Perturbations and Scene-State Components 22

E Visualization of Chain-of-Motion Trajectory Prediction 23
- E.1 Perception and Motion Evidence 23
- E.2 Example of a Motion Decision 23

F Limitations and Future Work 24

A Implementation Details

A.1 Pretrained Components and Inference Configuration

Kepler4D combines pretrained models for geometry estimation, visual perception, mask propagation, and video synthesis. Depth Anything 3 (Lin et al., 2026) provides depth, intrinsics, and camera poses. Qwen2.5-VL-7B-Instruct (Bai et al., 2025) is used for scene understanding and structured motion decisions, while SAM 2.1 Hiera-Large (Ravi et al., 2025) propagates foreground masks. Future video synthesis uses Wan2.1-T2V-14B (Team Wan et al., 2025) with the pretrained geometric-conditioning components of VerseCrafter (Zheng et al., 2026). The object trajectories and their rendered controls connect these components without requiring the video generator to produce an explicit 3D motion plan itself.

The inference implementation supports the Wan2.1-T2V-14B CFG- and step-distilled LoRA weights (Hu et al., 2022) released by LightX2V (LightX2V Contributors, 2025). The model card attributes their four-step bidirectional distillation to training code adapted from Self Forcing (Huang et al., 2025). The launcher configures a rank-64 v2 LoRA; we do not perform distillation ourselves.[1] Table A1 lists the default synthesis configuration exposed by the inference launcher. These are inference defaults; clip length and the observed/future partition belong to the evaluation setting rather than to the model architecture. The reported hardware is an NVIDIA RTX PRO 6000 Blackwell GPU with 96 GB memory.

Table A1: Default video-synthesis launcher settings; listed spatial dimensions are height × width.

Component or setting	Configuration
Video backbone	Wan2.1-T2V-14B
Geometric conditioning	Pretrained VerseCrafter GeoAdapter
Acceleration	LightX2V CFG/step-distillation LoRA; merge weight 1.0
Numerical precision	bfloat16
Sampler	Flow-matching Euler scheduler
Denoising steps	6
Classifier-free guidance scale	1.0
Schedule shift	5.0
GeoAdapter context scale	1.0
Output resolution	720×1280
Output frame rate	16 frames per second
Random seed	2025
Negative prompt	None

The Qwen motion-decision call uses a temperature of 0.1. Its output is parsed as structured JSON before trajectory generation. Model loading can be amortized through persistent inference services; the alternative standalone interface loads the same synthesis components for each invocation. Consequently, cold-start and warm-service timings represent different workloads. Runtime depends on the generator, acceleration settings, and whether model initialization is included.

A.2 Geometry, Foreground Identification, and Spatial Support

Geometric observations. Let $C_t^{\mathrm{c2w}} = (R_t, \mathbf{t}_t)$ denote a camera-to-world pose, and let $\widetilde{\mathbf{u}} = (u_x, u_y, 1)^\top$ be a homogeneous pixel coordinate. The inverse projection used by the state construction can be written as

$$\mathbf{x}_t(\mathbf{u}) = R_t \left[D_t(\mathbf{u}) K_t^{-1} \widetilde{\mathbf{u}} \right] + \mathbf{t}_t. \tag{A1}$$

This expression makes explicit the roles of depth, calibration, and camera pose in relating image observations to object motion. Here depth is measured along the camera's optical axis. Image resizing requires the corresponding rescaling of focal lengths and principal-point coordinates before

[1] Upstream model card and LoRA releases: `lightx2v/Wan2.1-T2V-14B-StepDistill-CfgDistill-Lightx2v`, `loras/`. The configured filename is `lightx2v_T2V_14B_cfg_step_distill_v2_lora_rank64_bf16.safetensors`, also published in Kijai's converted LightX2V release.

unprojection. Spatial extents and object-center trajectories use the same reconstructed coordinate system.

For longer observed videos, geometry inference can be performed in overlapping temporal chunks. The chunking interface uses 60-frame windows with 30-frame overlap. Adjacent reconstructions are aligned through a similarity transform estimated from their shared camera poses. The estimated scale is applied to depth as well as camera translation, so the two remain compatible when constructing scene points. This procedure concerns consistency within the observed reconstruction.

Foreground identity and masks. The first and last observed frames are provided to the perception VLM. Its structured output contains object identifiers, semantic labels, endpoint bounding boxes, a scene description, and scene-context cues. The endpoint boxes initialize SAM 2 mask propagation across the observed video. Propagation in both temporal directions allows a prompt on a later frame to contribute to earlier masks. The resulting object identifiers associate masks across time and connect each motion decision to the corresponding foreground proxy. The center average in Eq. 3 is defined for a nonempty mask with valid depth samples. A fully occluded object has no directly observed center in that frame.

Coarse spatial support. The spatial extent $\mathbf{e}_i$ describes the support used for geometric control, not a detailed object mesh. A single object keeps the same extent during its future trajectory, while its center changes. This construction separates translation guidance from the appearance and local articulation synthesized by the video generator. In particular, a person's limbs or a vehicle's fine shape are not explicitly represented by the coarse support.

A.3 Motion Summaries and Structured Decisions

CoM separates measured motion from semantic decisions. The motion extractor summarizes displacement, speed, acceleration, and turning trend; the VLM then chooses how the motion should continue in context. These quantities supplement the scene description and the endpoint images, rather than requiring the VLM to recover all numerical motion information directly from pixels.

Displacement and speed. Let $n_i > 1$ denote the number of valid reconstructed centers retained in an observed track, and let $\mathbf{c}_i^{\text{first}}$ and $\mathbf{c}_i^{\text{last}}$ be the directly reconstructed centers at its first and last valid frames. The implementation uses the endpoint estimate

$$\mathbf{v}_i = \frac{\mathbf{c}_i^{\text{last}} - \mathbf{c}_i^{\text{first}}}{n_i - 1}, \qquad s_i = \|\mathbf{v}_i\|_2. \tag{A2}$$

Frames without a valid reconstructed center are omitted, without filling temporal gaps. The denominator counts retained-observation steps rather than the elapsed frame-index difference. Motion is therefore expressed in reconstructed coordinate units per retained-observation step, equivalent to units per sampled-frame interval only when the retained frames are consecutive. A track with negligible displacement is assigned zero speed, avoiding division by a nearly zero displacement when forming its direction. The stationary cue is relative to the object's spatial extent, so the semantic decision receives both motion magnitude and an object-scale reference.

Turning and acceleration cues. Turning trend is obtained from signed changes between short planar displacement vectors. The motion extractor uses displacements spanning three retained-observation steps in the recent history, rejects nearly zero displacements, takes a median over valid signed changes, and limits the estimated turn rate to $[-8, 8]$ degrees per retained-observation step. The recent-history window contains at most 20 trajectory samples. Acceleration is summarized from changes in recent step lengths. The velocity and turn-rate estimates are used numerically for per-frame rollout without correcting for observation gaps. These estimates are cues for selecting future behavior, rather than a claim that noisy monocular reconstruction recovers exact physical velocities or forces.

Decision vocabulary. The speed decision α_i selects maintaining, increasing, or decreasing speed, and the heading decision δ_i selects straight continuation, a left turn, or a right turn. As defined in Eq. 5, the response also includes η_i, which selects whether to apply the lower bound h_i to the

predicted object-center height. This bound constrains the center trajectory; it does not represent physical contact or enforce an object mesh to lie on a supporting surface. The motion description d_i provides text guidance for synthesis, associating the geometric controls and the semantic description with the same predicted behavior.

The JSON parser validates the action vocabulary. An invalid speed action is replaced by maintaining speed, and an invalid heading action is replaced by straight continuation. If a tracked object is omitted from a parsed response, the decision list is completed using these conservative defaults. This validation addresses the structured control interface; it does not guarantee that a semantically plausible decision will be inferred in every scene.

A.4 Speed and Heading Updates in Deterministic Rollout

The numerical rollout starts from the latest observed position and estimated velocity. Here we detail how Γ_{motion} in Eq. 6 numerically instantiates the structured speed and heading decisions as rollout parameters. Let $k = 0, \ldots, T_f - 1$ index future steps, and let $g_{\alpha_i}(k)$ be the displacement multiplier associated with the speed decision. The implemented schedules are

$$g_{\alpha_i}(k) = \begin{cases} 1, & \text{maintain}, \\ \min(2, 1 + 0.02k), & \text{accelerate}, \\ \max(0.3, 1 - 0.02k), & \text{decelerate}. \end{cases} \tag{A3}$$

Here accelerate and decelerate implement the increase and decrease decisions of the main paper. The multiplier acts on the displacement at each step. It is not repeatedly multiplied into the stored velocity magnitude, so an acceleration action does not induce an unintended exponential speed increase.

For a selected per-step turn ω_i, the planar velocity update uses

$$Q(\omega_i) = \begin{bmatrix} \cos\omega_i & -\sin\omega_i \\ \sin\omega_i & \cos\omega_i \end{bmatrix}, \qquad \mathbf{v}_{i,\parallel}^{(k+1)} = Q(\omega_i)\mathbf{v}_{i,\parallel}^{(k)}. \tag{A4}$$

In the rollout coordinate convention, $\mathbf{v}_{i,\parallel}$ contains the X and Z components, and the Y component is unchanged by this rotation. The initial velocity is $\mathbf{v}_i^{(0)} = \mathbf{v}_i$. Turn rates stated in degrees are converted to radians before evaluating $Q(\omega_i)$. The speed and heading contribution to the position update is then

$$\widetilde{\mathbf{c}}_i^{(k+1)} = \widetilde{\mathbf{c}}_i^{(k)} + g_{\alpha_i}(k)\mathbf{v}_i^{(k+1)}, \qquad \widetilde{\mathbf{c}}_i^{(0)} = \mathbf{c}_i^{T_o - 1}. \tag{A5}$$

Equations A3–A5 describe the speed and heading substep, with step $k + 1$ corresponding to absolute frame $T_o + k$; the complete motion decision and trajectory interface are defined in § 3.3 of the main paper.

To realize the structured heading decision δ_i within Γ_{motion}, automatic left/right actions use the measured turn-rate magnitude when it exceeds 0.3 degrees per step; otherwise, a category-derived surface cue selects a default magnitude. The defaults are 1.5 degrees for water, 2 degrees for road, 3 degrees for ground, and 2 degrees for an unknown cue. The chosen side determines the sign. Straight continuation uses zero additional turn. The turn-rate rules above describe automatic continuation. For language-guided editing, the turning angle specified in the instruction is passed to the deterministic rollout to construct a corresponding 3D object-center trajectory. The resulting trajectory is rendered into geometric controls to guide video synthesis.

The height decision is applied to the object-center coordinate in the local height frame. When η_i enables the constraint, a proposed height below h_i is raised to h_i; otherwise, the proposed height is retained. The other position coordinates retain their speed-and-heading updates. This is the optional center-height lower bound in Eq. 6, not a projection of the object onto a physical surface. Its behavior depends on the estimated height reference and the selected decision.

After rollout, a one-dimensional Gaussian filter with a standard deviation of one sampled frame smooths the trajectory along time. The first and last generated center positions are retained from the unsmoothed segment, preserving its endpoints. Smoothing reduces abrupt variations in the control path; it does not add articulated motion or simulate forces.

A.5 Geometric Control Representation

The renderer represents each foreground object by a Gaussian with mean $\widehat{\mathbf{c}}_i^t$ and a diagonal covariance derived from its fixed spatial extent. If $\mathbf{e}_i = (e_{i,1}, e_{i,2}, e_{i,3})$ contains full axis lengths, the parameterization is

$$\Sigma_i = \mathrm{diag}(\sigma_{i,1}^2, \sigma_{i,2}^2, \sigma_{i,3}^2), \quad \sigma_{i,j} = \max\left(\frac{e_{i,j}}{2\rho}, 10^{-3}\right), \qquad \rho = 2.5. \tag{A6}$$

When the floor is inactive, the ρ-standard-deviation ellipsoid has half-axis lengths $\mathbf{e}_i/2$. The standard-deviation floor is expressed in the reconstructed coordinate units and prevents degenerate covariance axes. The covariance remains fixed during a rollout, while the mean follows the object trajectory. Changing the center therefore changes the spatial guidance without requiring an explicit prediction of detailed object shape.

The synthesis interface consumes four control videos and a mask video, as summarized in Table A2. Foreground color identifies the projected object support; it is not a photorealistic reconstruction of object texture. Appearance information is supplied by the last observed image and the pretrained generator. Background and foreground depth provide geometric context for combining the controls and their visibility.

Table A2: Control streams supplied to the pretrained synthesis interface. The combined background/foreground visualization is diagnostic and distinct from the separate conditioning streams.

Stream	Information represented
Background RGB	Appearance of the rendered background geometry
Background depth	Depth of the rendered background geometry
Foreground Gaussian RGB	Projected object support and identity color
Foreground Gaussian depth	Depth of the coarse foreground support
Merged mask	Spatial mask used by the video-conditioning interface

Appearance and text conditioning. The last observed frame I_{T_o-1} is the appearance reference. The synthesis loader inserts this reference into the first background-conditioning slot and marks the reference slot accordingly in the input mask. The motion description produced by CoM supplies the default synthesis prompt. An explicit prompt or a prompt file can override that description at the generation interface. Text specifies semantic motion while the geometric streams specify spatial guidance; neither stream is a substitute for the other.

Temporal length. The video VAE uses a temporal compression factor of four. For a requested output length $L_{\mathrm{req}} > 1$, the synthesis loader uses

$$L_{\mathrm{out}} = 4\left\lfloor \frac{L_{\mathrm{req}} - 1}{4} \right\rfloor + 1. \tag{A7}$$

Lengths of the form $4k + 1$ avoid truncation at this interface. The encoded clip length, the role of its appearance-reference slot, and the future prediction horizon are separate bookkeeping quantities. Evaluation compares the generated continuation with the corresponding held-out portion of the source video, as described below.

B Dataset Usage and Evaluation Settings

Static view synthesis. We use the 55-scene DL3DV-Evaluation release of DL3DV (Ling et al., 2024) for static evaluation.[2] This release is independent of the original DL3DV-10K dataset and distinct from its 140-scene NVS benchmark. The first 49 frames serve as observations; held-out views along the provided camera trajectories serve as reconstruction references. All methods receive the same observed frames, and methods supporting camera control receive the same target path. This setting evaluates background reconstruction and geometry-conditioned synthesis, with no foreground motion reasoning to assess. Static performance therefore does not independently measure the accuracy of CoM.

[2]Dataset release and scene list: `huggingface.co/datasets/DL3DV/DL3DV-Evaluation`.

Automatic dynamic continuation. The dynamic evaluation draws on the 90 sequences of the DAVIS 2017 train and validation splits (60 train and 30 validation) (Perazzi et al., 2016; Pont-Tuset et al., 2017). Automatic continuation uses 38 of these sequences: 24 train and 14 validation sequences. Each is divided into an observed prefix and a held-out continuation. The selected sequences contain 75–104 frames: 37 exceed 75 frames, while `car-roundabout` contains 75 frames, with 35 observed and 40 held-out frames. Thus, the 38-sequence set is not defined by a strict greater-than-75-frame rule. All methods receive the same observed video; no future object trajectory, mask sequence, or language motion command is supplied as an external control. Descriptions inferred from the observed video are internal model outputs.

DAVIS does not provide ground-truth camera poses. We estimate camera trajectories from each full sequence with Depth Anything 3 (Lin et al., 2026) and provide the future poses as external rendering conditions, shared by all camera-controllable methods. Thus, this setting evaluates continuation under a supplied future camera path rather than prediction of the camera motion. The held-out video provides one realized continuation against which reference-based metrics are computed.

Language-guided generation. The controlled cases are constructed from the same 90-sequence DAVIS 2017 train-and-validation pool. This setting additionally provides a high-level motion command. Kepler4D converts the command into object-motion guidance before synthesis. The text-conditioned baselines receive the same command, without Kepler4D's inferred trajectories or rendered foreground controls. This comparison examines the complete route from a shared language instruction to generated motion, rather than comparing generators supplied with identical geometric conditioning tensors. Camera-controllable methods use the same DA3-estimated camera path. Reference-frame comparisons use the corresponding recorded continuation, while TurnErr provides the restricted camera-turning proxy defined in Appendix C.

Temporal correspondence. For static evaluation, frame index represents progress along the prescribed view sequence. Outputs of different lengths are uniformly resampled to the reference sequence before frame-wise comparison. Dynamic evaluation instead uses wall-clock time: the common duration of the reference and generated clips defines the comparison interval, reference-rate timestamps are sampled, and the nearest generated frame is selected at each timestamp. Selected frames are resized to the reference resolution. This distinction avoids equating equal frame indices with equal elapsed time when frame rates differ.

Interpretation of the reference. A recorded continuation is useful for measuring similarity, but does not exhaust the set of plausible futures. In an edited sequence, following a requested motion may decrease similarity to the unedited recording. We therefore report reference similarity together with perceptual quality and motion-related measurements, and interpret each according to the quantity it actually measures.

C Evaluation Metrics

C.1 Metrics by Evaluation Setting

Static view synthesis uses PSNR, SSIM, LPIPS, and FID to compare held-out views, together with VBench image quality and temporal-consistency scores. Automatic dynamic continuation adds Dynamic Degree, Flow-EPE, and RotErr to characterize motion activity, image motion, and camera rotation. Language control is evaluated with PSNR, TurnErr, Imaging Quality, and Dynamic Degree. Trajectory perturbations use AngleErr, SurfErr, Imaging Quality, and Dynamic Degree; scene-state ablations use Flow-EPE, SurfErr, Dynamic Degree, and Imaging Quality. These measurements describe complementary properties of the complete generation pipeline.

C.2 Reference Similarity and Perceptual Quality

PSNR and SSIM. PSNR measures pixel-level reconstruction similarity after spatial and temporal alignment. With RGB values in $[0, 255]$, it is computed from the mean squared error as $10 \log_{10}(255^2/\mathrm{MSE})$. SSIM (Wang et al., 2004) measures local structural similarity between corresponding RGB frames. Frame-level measurements are aggregated within a sequence and then across

evaluated sequences. These metrics compare the generated images with the recorded reference; they do not establish whether an alternative edited motion is correct.

LPIPS and FID. LPIPS (Zhang et al., 2018) measures perceptual distance using the AlexNet-based implementation, with RGB inputs normalized to $[-1, 1]$. FID (Heusel et al., 2017) compares the distributions of Inception-V3 features pooled from aligned generated and reference frames. Lower LPIPS indicates greater perceptual similarity to the reference, while lower FID indicates a closer feature distribution. FID is a set-level statistic rather than a trajectory-following measurement.

VBench. We use VBench (Huang et al., 2024) in custom-input mode. Table A3 summarizes the dimensions used in the paper. Imaging Quality and the temporal-consistency measures evaluate properties of the generated video without requiring its motion to match the recorded continuation. Dynamic Degree assesses motion activity, so a larger value does not by itself imply more accurate object motion. The displayed dimensions are interpreted individually; the Overall column should not be reconstructed by averaging only the subset of dimensions visible in a results table.

Table A3: Interpretation of the VBench dimensions used for video evaluation in the main paper.

Dimension	Measured property
Aesthetic Quality	Visual aesthetic quality
Imaging Quality (Imag.)	Image quality, including sharpness and artifacts
Motion Smoothness (MotSm.)	Smoothness of temporal transitions
Subject Consistency	Consistency of foreground visual identity
Background Consistency (BgC.)	Consistency of background appearance
Dynamic Degree (DynDeg)	Presence of sufficiently strong motion

C.3 Image Motion and Camera Motion

Flow-EPE. Dense optical flow is estimated with RAFT-Large (Teed & Deng, 2020) for both the generated and reference videos. For $T \geq 2$ aligned frames indexed $0, \ldots, T-1$, let F_t denote flow from frame $t-1$ to frame t. Over image domain Ω, the endpoint error is

$$\text{Flow-EPE} = \frac{1}{T-1} \sum_{t=1}^{T-1} \frac{1}{|\Omega|} \sum_{\mathbf{u} \in \Omega} \left\| F_t^{\text{gen}}(\mathbf{u}) - F_t^{\text{ref}}(\mathbf{u}) \right\|_2 . \tag{A8}$$

The flow field is evaluated over the image, so this quantity reflects the combined image motion caused by camera movement, object displacement, and local appearance changes. It does not isolate foreground trajectory prediction from the rest of the generation pipeline.

RotErr. We adapt the rotation error of CameraCtrl (He et al., 2025), which sums per-frame geodesic errors over a clip. Our version averages these errors and reports them in degrees, avoiding a direct dependence on the number of evaluated frames. We compare camera rotations estimated by DA3-Large from generated and reference videos. For aligned frames indexed $0, \ldots, T-1$, R_t below denotes the camera rotation relative to the first frame, so $R_0^{\text{gen}} = R_0^{\text{ref}} = I$. The rotation error is the mean geodesic distance over the remaining frames:

$$\text{RotErr} = \frac{180}{\pi(T-1)} \sum_{t=1}^{T-1} \arccos\left(\text{clip}\left[\frac{\text{tr}(R_t^{\text{gen}} (R_t^{\text{ref}})^\top) - 1}{2}, -1, 1 \right] \right) . \tag{A9}$$

The factor $180/\pi$ expresses the result in degrees. Clipping the arccosine argument prevents numerical overflow outside its valid interval. This metric measures camera rotation rather than the heading of an independently moving foreground object.

C.4 Control-Related Measurements

TurnErr. The turning proxy compares a commanded angle θ_{cmd} with a camera-yaw quantity estimated from the generated clip:

$$\text{TurnErr} = \left| \widehat{\theta}_{\text{gen}} - \theta_{\text{cmd}} \right| , \qquad \widehat{\theta}_{\text{gen}} = \max_t \left| \sum_{\tau=1}^{t} \Delta \widehat{\psi}_\tau \right| . \tag{A10}$$

Here $\Delta\widehat{\psi}_\tau$ is the estimated camera-yaw increment; all angles are in degrees and $\theta_{\mathrm{cmd}} \geq 0$ denotes the commanded turning magnitude. For the car-roundabout example, background features are tracked across consecutive frames and a RANSAC homography rejects outliers. Median horizontal feature displacement is converted to a per-frame yaw change and accumulated over the clip. This yields the camera-motion signal used by the proxy. As specified in the main paper, TurnErr reflects camera turning and does not independently recover object heading. Its use of an absolute peak also means that it does not, on its own, distinguish left from right or a final angle from a transient maximum. TurnErr is reported as a diagnostic of generated camera turning and is not used to establish object-level compliance with the motion instruction. In particular, when the target camera path is fixed, accurate object turning does not necessarily imply a low TurnErr. Figure 5 provides qualitative examples of the object-motion responses to different instructions.

AngleErr. In the trajectory analysis, AngleErr compares the specified motion direction $\mathbf{d}_{\mathrm{ctrl}}$ with the direction $\mathbf{d}_{\mathrm{gen}}$ obtained from reconstructed, mask-tracked generated object centers:

$$\mathrm{AngleErr} = \arccos\left(\mathrm{clip}\left[\frac{\mathbf{d}_{\mathrm{gen}}^{\top}\mathbf{d}_{\mathrm{ctrl}}}{\|\mathbf{d}_{\mathrm{gen}}\|_2\|\mathbf{d}_{\mathrm{ctrl}}\|_2}, -1, 1\right]\right). \tag{A11}$$

Angles are converted to degrees for reporting. This angular comparison is defined for nonzero directions expressed in a common coordinate frame and concerns adherence to a supplied control direction, rather than the ability to predict the reference future trajectory from observations.

SurfErr. Using the notation of the evaluation, z_t^{gen} is the generated object-center height in the local support coordinate and h_{sup} is the estimated object-center support height inferred from the observed trajectory. The reported quantity is

$$\mathrm{SurfErr} = \frac{1}{T}\sum_{t=1}^{T} |z_t^{\mathrm{gen}} - h_{\mathrm{sup}}|\,. \tag{A12}$$

Here z_t^{gen} denotes the scalar height, not the raw Z component used in the rollout plane. The quantity has units of the reconstructed height coordinate and measures deviation from an estimated center-height reference. It does not compare the object's boundary with a physical supporting surface and does not measure contact accuracy. Lower SurfErr indicates closer adherence to this estimated height reference. Its interpretation depends on the reconstruction, the tracked object centers, and the chosen local coordinate. Here h_{sup} denotes the evaluation reference, while h_i in § 3.3 is the optional lower-bound parameter of trajectory rollout.

D Baseline Scope and Ablation Interpretation

The baselines test complementary ways of connecting observations, geometry, and video generation. We use the same observed video and, when supported, the same target camera trajectory for all methods. We follow official inference pipelines where available, adapting input formatting, frame resolution, and camera conventions to the evaluation setting. SPMem (Wu et al., 2025) provides spatial memory for scene-consistent generation. GEN3C (Ren et al., 2025) uses a 3D cache as geometric guidance. TrajectoryCrafter (Yu et al., 2025) redirects the camera trajectory of a monocular source video. VerseCrafter (Zheng et al., 2026) provides joint camera/object geometric controls, including object trajectories. These capabilities differ from inferring a future object trajectory through CoM and then rendering that trajectory for synthesis.

The language-control comparison also includes Wan Inpaint, which uses the Wan2.1 family (Team Wan et al., 2025). The label denotes the baseline's use in this experiment rather than a separate model family. In the language-guided setting, its motion instruction is supplied as text. It does not receive the object trajectories or foreground control maps inferred by Kepler4D.

D.1 Motion Description and Trajectory Guidance

The motion-conditioning study separates a description of the observed scene from a description of future motion, and then adds spatial trajectory guidance. Table A4 summarizes this distinction. The comparison asks what the generation pipeline gains from these different forms of motion information; it is not a direct evaluation of a VLM's standalone 3D forecasting accuracy.

Table A4: Text and trajectory guidance supplied by the three tested motion-conditioning variants.

Variant	Text guidance	CoM trajectory guidance
Qwen prompt	Observed scene description	Absent
CoM w/o traj.	Predicted motion description	Not rendered into foreground controls
CoM w/ traj.	Predicted motion description	Rendered future object trajectories

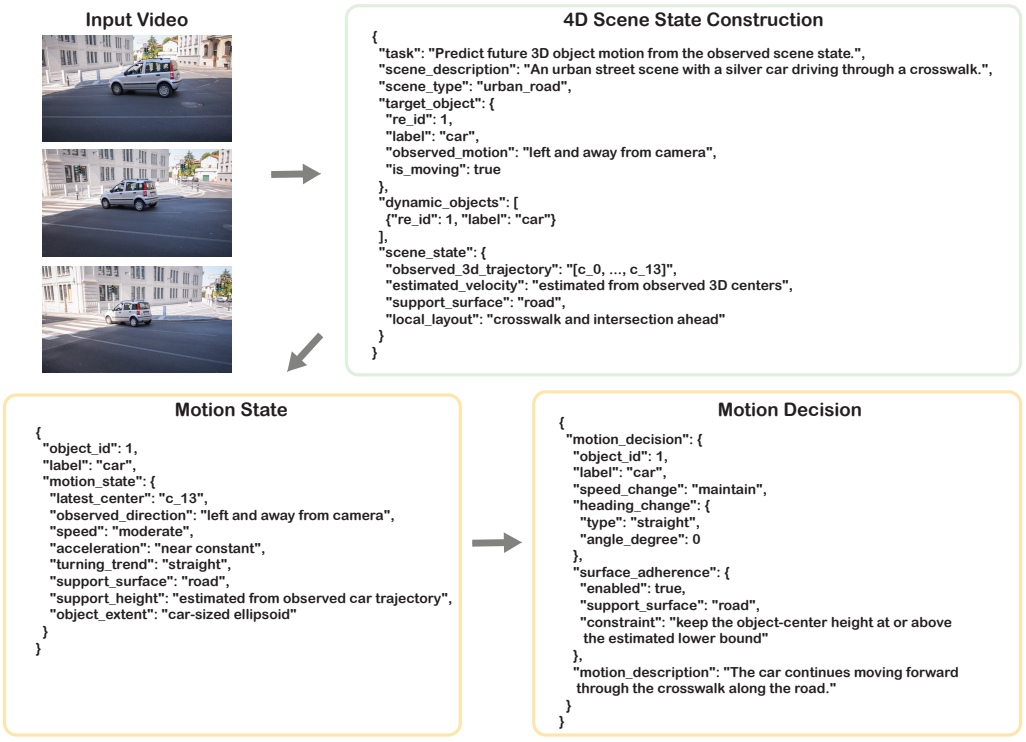

Figure 6: **Schematic illustration of Chain-of-Motion inputs and outputs.** CoM uses observed motion and semantic scene context to select speed and heading actions, decide whether to apply a lower bound to object-center height, and produce a motion description. The structured decisions guide deterministic 3D trajectory rollout, while the description provides text conditioning for video synthesis. The fields illustrate the interface and are not numerical logs from a benchmark sequence.

D.2 Trajectory Perturbations and Scene-State Components

The trajectory-control analysis uses a reference trajectory derived from the source video and directional perturbations of that trajectory. It examines the response of the synthesis pipeline to changed geometric guidance. AngleErr and deviation from the estimated object-center support height characterize properties of the generated response. This study is distinct from automatic motion prediction, because a source-derived trajectory is supplied as a control.

The scene-state component study examines observed motion history, semantic context, and surface-related cues within the same generation pipeline. Motion history describes how the object has moved; semantic context describes its identity and surroundings; surface-related cues provide additional context for interpreting likely motion. The reported changes in Flow-EPE, DynDeg, and Imaging Quality are interpreted as effects on the complete pipeline. In particular, a change in Dynamic Degree measures a change in motion activity, and does not by itself establish that one trajectory prediction is more accurate.

E VISUALIZATION OF CHAIN-OF-MOTION TRAJECTORY PREDICTION

Figure 6 schematically illustrates the scene information, motion state, and structured decisions used by Chain-of-Motion to guide deterministic future 3D object-center trajectory rollout. The scene state records background geometry, dynamic-object identities, observed object centers and extents, and semantic scene context. These quantities connect the input video to a structured motion interface in the reconstructed 3D frame.

The `surface_adherence` field in the schematic example denotes the decision η_i in Eq. 5. In our rollout, it enables the estimated object-center height lower bound h_i, rather than enforcing contact with a reconstructed road surface.

For each object, motion-state extraction summarizes the latest center, velocity, speed, acceleration, turning trend, height threshold, coarse extent, and semantic cues, as in Eq. 4. CoM then predicts speed and heading decisions, the optional center-height lower-bound decision, and a textual motion description. Deterministic rollout converts the structured decisions into future center positions. The predicted or edited trajectory becomes rendered foreground guidance under the target camera path, while the motion description provides semantic guidance for synthesis. The generator supplies appearance and local articulation beyond this coarse trajectory representation.

The interface below provides schematic examples of the connection shown in Figure 6. These examples describe fields and their roles; they are not numerical logs from a benchmark sequence.

E.1 PERCEPTION AND MOTION EVIDENCE

The perception query asks the VLM to identify foreground objects in the first and last observed frames, assign the same identity across the two frames, return endpoint boxes, and summarize the scene. Object boxes then initialize mask tracking. The motion query augments the resulting description with measured direction, speed, acceleration, turning trend, and stationary cues. Table A5 relates these quantities to their roles.

Table A5: Roles of observed information and motion decisions in the structured CoM interface.

Information	Role
Endpoint images	Visual evidence for object identity and scene context
Object identifier and label	Associate masks, history, decisions, and controls
Observed motion summary	Numerical evidence for speed and heading decisions
Scene description	Context for a plausible continuation
Speed, heading, and height-bound decisions	Inputs to deterministic trajectory updates
Motion description	Semantic guidance for video synthesis

E.2 EXAMPLE OF A MOTION DECISION

Consider a tracked car whose observed history indicates forward movement along a road. A schematic motion query can be expressed as follows:

> Given the observed scene, the endpoint images, and the measured motion of each tracked object, select speed and heading actions and decide whether to apply the estimated lower height bound to its future center positions. Return the decisions with the corresponding object identifier and a concise description of the expected motion for video synthesis.

A possible motion-related subset of the response is

Field	Schematic value
Object identifier	1
Object label	Car
Speed action	Maintain
Heading action	Straight
Height-bound decision	Enabled
Motion description	The car continues moving forward along the road.

The deterministic rollout turns the selected actions and the observed motion state into future center positions, applying the height threshold only when the corresponding decision enables it. Those positions become geometric foreground guidance, while the motion description becomes text conditioning. The two outputs serve different purposes: the trajectory specifies where the coarse object support moves, and the description conveys the intended behavior to the pretrained generator. Fine appearance and articulation are synthesized during video generation.

F Limitations and Future Work

Scope of the motion representation. Kepler4D targets editable object-center motion under a supplied camera path. This compact interface lets the same trajectory support motion prediction, language editing, and geometric rendering, while the pretrained generator supplies appearance and local articulation. Fixed coarse supports do not explicitly model joint motion, shape deformation, or contact dynamics, so geometric guidance alone does not guarantee physically correct interactions. Extending the interface with articulated supports and contact constraints is a natural direction for applications that require finer physical control.

Reliability of reconstructed state. The scene state is estimated from monocular video using pretrained depth, camera, segmentation, and language models. Occlusion and changes in visible surface area can shift a reconstructed center even when the physical center moves differently. Kepler4D exposes these estimates as explicit motion histories, making their role in subsequent decisions inspectable, but does not currently propagate calibrated uncertainty through the rollout. Confidence-aware tracking and joint refinement of geometry and motion offer ways to improve robustness to difficult observations without changing the trajectory-based control interface.