KINEMATIC TRANSITION MODEL FOR TRAINING-FREE ONLINE HUMAN MOTION CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Interactive human motion synthesis aims to continually generate plausible motion while responding to changing user inputs and environmental constraints. Recent autoregressive motion models enable efficient online generation, but their real-time control typically relies on predefined conditioning signals, specialized control interfaces, or objective-specific training. In practical scenarios, however, control objectives may be newly introduced, changed, or combined during execution, requiring the motion generator to adapt without retraining while maintaining real-time performance. To address this challenge, we propose Kinematic Transition Model (KTM) for training-free online human motion control. Starting from the observed motion, KTM constructs an analytical kinematic continuation from its boundary dynamics and represents possible futures as acceleration corrections to this continuation. We then model future generation through a small number of latent stochastic transitions in this boundary-attached kinematic space, capturing the one-to-many evolution toward plausible future motions. Importantly, the same transition representation provides an efficient interface for training-free control. At each transition step, the predicted transition is kept fixed as a local motion prior while a future endpoint is optimized under the current inference-time objective. The optimized endpoint analytically defines the next controlled transition, allowing new or changing differentiable objectives to steer motion generation without control-specific training, additional motion-model computations, or backpropagation through the learned generator during optimization. Experiments demonstrate that KTM enables high-quality online motion synthesis with flexible control over unseen objectives while maintaining real-time inference.

1 INTRODUCTION

Interactive human motion generation aims to continually produce plausible motion while responding to changing user inputs or environmental constraints. Recent autoregressive motion models have made substantial progress toward online and real-time motion synthesis by predicting the next frame or motion segment from the generated motion history (Shi et al., 2024; Chen et al., 2024b; Zhao et al., 2025; Xiao et al., 2025; Wang et al., 2026; Zhao et al., 2026). Beyond fast generation, however, practical scenarios may require introducing, changing, or combining control objectives during execution. For example, a character moving toward a waypoint may encounter a moving obstacle and must immediately avoid it while continuing toward the original target, requiring multiple objectives to be considered simultaneously as the motion progresses. A practical interactive motion generator should be able to support newly specified or changing objectives *without retraining* (i.e., *training-free*), while continuing to generate motion within a real-time budget. Retraining the motion model for each new objective is impractical, while repeatedly sampling or optimizing through the motion generator may not meet the latency requirement for online generation.

Despite recent progress, existing methods do not jointly demonstrate continual online generation, control under unseen inference-time objectives, and real-time execution through a single generic inference-time control interface. As summarized in Table 1, training-free methods such as DNO (Karunratanakul et al., 2024) and MaskControl (Pinyoanuntapong et al., 2025) can introduce new objectives at inference, but are designed for offline motion generation. Conversely, online methods such as AAMDM (Li et al., 2024), MotionBricks (Wang et al., 2026), and ARDY (Zhao et al., 2026) achieve efficient interactive control through task-specific query, primitive, or learned

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.

Table 1: Comparison of representative kinematic motion control methods. *Unseen Objective* denotes support for control objective functions that can be specified or changed at inference through the same generic control interface, without objective-specific training or adaptation. *Real-time Unseen-Objective Control* requires the same generic control interface for unseen objectives to operate within the real-time inference budget. ✓ indicates that the published method demonstrates the criterion, while × indicates that it does not.

Method	Control Interface	Online Generation	Unseen Objective	Real-time Unseen-Objective Control
DNO (Karunratanakul et al., 2024)	*Diffusion* noise optimization	×	✓	×
MaskControl (Pinyoanuntapong et al., 2025)	*Masked-token* logit optimization	×	✓	×
AAMDM (Li et al., 2024)	*Diffusion* query guidance	✓	×	×
A-MDM (Shi et al., 2024)	*Diffusion* score-based candidate selection	✓	✓	×
DART (Zhao et al., 2025)	*Latent diffusion* noise optimization / RL policy	✓	✓	×
PRIMAL (Zhang et al., 2025)	*Diffusion* analytical-gradient guidance	✓	✓	×
MotionBricks (Wang et al., 2026)	*Masked-token* smart-primitive conditioning	✓	×	×
ARDY (Zhao et al., 2026)	*Diffusion* spatial-constraint conditioning	✓	×	×
KTM (Ours)	***Kinematic transition* real-time control**	✓	✓	✓

conditioning interfaces rather than newly specified objective functions. A-MDM (Shi et al., 2024), DART (Zhao et al., 2025), and PRIMAL (Zhang et al., 2025) come closer by supporting objective-based inference-time control, but their inference-time control interface relies on candidate sampling, iterative optimization, or guidance that is not demonstrated for unseen objectives within the real-time constraints. This leaves a key gap in jointly supporting flexible training-free objective control and efficient online real-time generation through the same generic control procedure.

Addressing this gap presents two key challenges. *First*, unlike offline generation, online human motion control always starts from an already moving character. When the objective changes, the model needs a way to express how the upcoming motion should transition from its current dynamics while remaining consistent with the observed motion (Xiao et al., 2025). This motivates a future transition representation that is explicitly attached to the observed motion and describes changes over the same upcoming motion window, so that each transition state meaningfully represents how the future dynamics evolve from the current observed motion. This should provide a structured control space where the control objective can propagate necessary changes across the future motion coherently, rather than adjusting individual poses independently. *Second*, flexible inference-time control must remain efficient under a real-time budget (Wang et al., 2026). An effective generative process should therefore operate on this structured future state through a finite number of stochastic transitions, allowing the specified objective to steer the future transition without repeatedly sampling from or backpropagating through the learned motion model.

To tackle these challenges, we introduce **Kinematic Transition Model (KTM)** for training-free online human motion control. Starting from the observed motion, KTM constructs an analytical kinematic continuation from its boundary dynamics and represents possible futures as acceleration corrections to this continuation. This defines a boundary-attached *kinematic transition space* over the same upcoming motion window, where each transition explicitly represents how the future dynamics change relative to the current motion. KTM then generates motion through a small number of stochastic finite transitions in this space, capturing the one-to-many evolution toward different plausible futures. Importantly, the kinematic transition space also provides a structured interface for efficient training-free control. At each step, the predicted transition serves as a local motion prior, while a future endpoint is optimized in this space under the control objective. The optimized endpoint analytically determines the next controlled transition, allowing unseen objectives to steer online generation without control-specific retraining or additional evaluations of the learned motion model during endpoint optimization.

In summary, our contributions are: *(i)* we address the underexplored setting of training-free online motion control under real-time constraints, where unseen objectives are introduced only at inference using the same pre-trained motion model and control interface; *(ii)* we formulate online future generation as finite stochastic transitions in a kinematic acceleration-correction space, where the process starts from an analytical continuation of the observed motion dynamics, and each transition state represents how the dynamics evolves toward plausible motions over the same upcoming future; and *(iii)* we present training-free transition control in this kinematic space, using each predicted transition as a fixed local motion prior and optimizing its future endpoint to define the next controlled transition state, enabling unseen objectives to steer online generation without control-specific training.

2 KINEMATIC TRANSITION MODEL FOR TRAINING-FREE ONLINE CONTROL

Problem setting. Let $X = (x_{1-L}, \dots, x_0)$ denote the observed motion history of length L, $Y = (y_1, \dots, y_F)$ the future motion over the next F frames. We aim at addressing training-free online control, where a differentiable objective $G(Y)$ can be newly specified, changed, or combined with other objectives during inference. For example, the current objective may describe trajectory following or obstacle avoidance. Meanwhile, the motion model is trained only once on motion data, without supervision from these inference-time objectives or control-specific data. Our goal is to continually generate plausible future motion under the specified objective, using the same pre-trained model and control procedure within a real-time budget.

Overview. We formulate online motion synthesis as a stochastic transition from the kinematic future implied by the observed history motion to plausible actual futures. We first construct an analytical continuation from the observed boundary dynamics and represent possible futures as *acceleration corrections* to this continuation (Sec. 2.1). We then generate these corrections through a small number of latent finite transitions, capturing the one-to-many nature of future motion while preserving its kinematic relationship to the observed motion (Sec. 2.2).

The same transition formulation enables training-free online control (Sec. 2.3). At each finite step, the predicted transition serves as a local motion prior for optimizing a future endpoint under the current inference-time objective. The optimized endpoint defines the next analytical transition, whose sampled state conditions the subsequent step. Because the predicted transition remains fixed during endpoint optimization, control requires no additional evaluations of or backpropagation through the learned motion model.

2.1 KINEMATIC FUTURE TRANSITION REPRESENTATION

We begin by using the motion at the observed boundary to construct a reference for the upcoming frames. In addition to the current pose, the boundary velocity and acceleration describe how the motion is evolving. By propagating the observed boundary dynamics over the upcoming future frames, we obtain a kinematic future continuation that serves as the starting point for transition generation. We represent the actual future through the acceleration corrections with respect to this kinematic continuation, so both the continuation and the actual future are reconstructed from the same observed boundary.

Kinematic future continuation. Given the observed history X, we define its boundary state as (x_0, v_0, a_0), where x_0 is the last observed pose, and v_0 and a_0 are its corresponding velocity and acceleration computed using backward differences. We construct a reference future acceleration sequence $\bar{A} = (\bar{a}_1, \dots, \bar{a}_F)$ by setting $\bar{a}_f = \rho_f a_0$. Here, $\rho_f = \exp(-\eta f \Delta t)$ is a drift factor that gradually reduces the contribution of the boundary acceleration over the future window to mitigate drifted future, with a fixed rate parameter $\eta \geq 0$ and motion-frame interval Δt.

To reconstruct motion from an acceleration sequence $A = (a_1, \dots, a_F)$, we define a kinematic operator $\kappa(X, A)$ that construct the future by integrating the motion dynamics forward from the observed boundary, using the future acceleration at each frame:

$$v_f = v_{f-1} + \Delta t\, a_f, \qquad y_f = y_{f-1} + \Delta t\, v_f, \qquad y_0 = x_0. \tag{1}$$

For joint rotation, angular accelerations are computed using the logarithm map, and rotations are reconstructed on $\mathrm{SO}(3)$ using the exponential map and rotation composition. Under the kinematic operator κ, all reconstructed futures share the same initial pose and velocity state at the boundary of the observed motion history. Thus, every future motion can be represented through its subsequent acceleration dynamics while remaining attached to the observed boundary over the future window.

Acceleration-correction state. Applying the kinematic operator to the reference acceleration sequence $\bar{A}$ gives the kinematic future continuation $\bar{Y} = \kappa(X, \bar{A})$. We then represent each training ground-truth future sequence $Y^\star$ through the acceleration correction $Z^\star$ with respect to the reference acceleration $\bar{A}$. Specifically, given the ground-truth acceleration sequence $A^\star$, we define

$$Z^\star = A^\star - \bar{A}, \qquad Y^\star = \kappa(X, \bar{A} + Z^\star). \tag{2}$$

Then, we use $Z \in \mathbb{R}^{F \times d_a}$ to denote an arbitrary transition state, which contains acceleration corrections for all F future frames, where d_a is the number of acceleration coordinates per frame. The

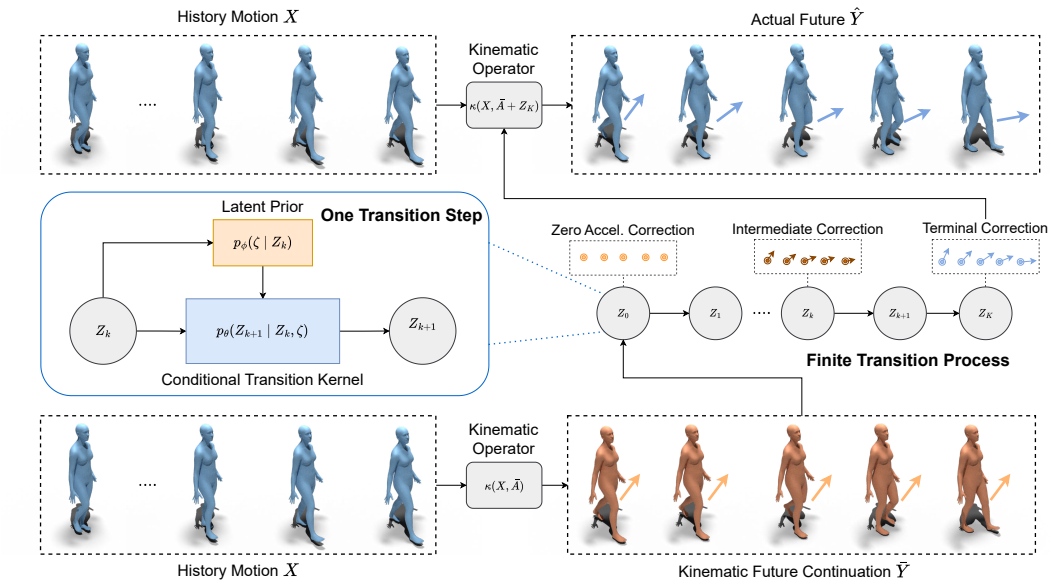


Figure 1: Kinematic Transition Model. From the observed motion X, we construct a boundary-attached kinematic future continuation $\bar{Y}$ and represent its evolution toward a plausible actual future $\hat{Y}$ through acceleration corrections. Starting from the zero acceleration-correction state Z_0, a small number of finite transitions progressively produce the terminal correction Z_K over the same upcoming future interval. Each transition step is modeled by a latent $\zeta \sim p_\phi(\zeta \mid Z_k)$ conditioning the transition kernel $p_\theta(Z_{k+1} \mid Z_k, \zeta)$, which stochastically evolves the state Z_k to Z_{k+1}.

corresponding future motion is reconstructed as $Y(Z) = \kappa(X, \bar{A} + Z)$. Accordingly, $Z = 0$ gives the kinematic continuation $\bar{Y}$, while $Z = Z^\star$ recovers the ground-truth future $Y^\star$.

This construction provides the transition process with a shared representation in the kinematic acceleration correction space. The future continuation, intermediate transition state, and actual future all describe the same upcoming frames and share the same observed boundary. Therefore, each transition models how the future acceleration dynamics evolve with respect to the reference acceleration sequence $\bar{A}$, while remaining kinematically attached to the observed motion and and exposing these dynamics as a structured space for inference-time control. In the following section, we model this evolution through a small number of learned stochastic transitions in this acceleration-correction Z space.

2.2 Kinematic Motion Transition Prior

The kinematic representation in Sec. 2.1 defines future generation in the kinematic acceleration-correction space of Z. Starting from $Z_0 = 0$, which corresponds to the kinematic continuation $\bar{Y}$, we generate a future through a small number of K stochastic transitions $Z_0 \to Z_1 \to \cdots \to Z_K$, where k indexes the transition step and is different from the motion-frame index f. Each $Z_k \in \mathbb{R}^{F \times d_a}$ represents an intermediate transition state for the acceleration corrections over the same future window, while the terminal state Z_K specifies the actual future motion through $\kappa(X, \bar{A}+Z_K)$. For compact notation, we omit additional conditioning inputs, including the observed history and bridge time condition, and write the learned distributions as conditioned on Z_k.

An observed motion history can admit multiple plausible futures, so the transition from the current state Z_k to Z_{k+1} is generally one-to-many. We model this multimodal transition distribution as

$$P_k(Z_{k+1} \mid Z_k) = \int p_\theta(Z_{k+1} \mid Z_k, \zeta)\, p_\phi(\zeta \mid Z_k)\, d\zeta, \tag{3}$$

where ζ is a stochastic latent variable that captures the variation among plausible transitions from the current state, predicted by the latent prior $p_\phi(\zeta \mid Z_k) = \mathcal{N}(\zeta; \mu_\phi(Z_k), \Sigma_\phi(Z_k))$. Given a sampled latent ζ, we then predict the conditional transition kernel parameterized by an explicit Gaussian distribution $p_\theta(Z_{k+1} \mid Z_k, \zeta) = \mathcal{N}(Z_{k+1}; \mu_\theta(Z_k, \zeta), \Sigma_\theta(Z_k, \zeta))$, from which we can sample the next transition state Z_{k+1}. In Eq.3, different latent samples induce different transition mode, while

their marginal P_k can represent the multimodal ways the current kinematic transition state may evolve toward the plausible actual futures.

At inference, we sequentially generate the future through the K finite transition steps. At each step k, we sample a transition latent $\zeta \sim p_\phi(\zeta \mid Z_k)$ and obtain the corresponding conditional transition $p_\theta(Z_{k+1} \mid Z_k, \zeta)$. This transition characterizes how the current kinematic state Z_k can evolve toward a plausible next state. The sampled next state Z_{k+1} then becomes the input to the subsequent transition, and the process continues. After K transition steps, the future motion is constructed using the kinematic operator as $\hat{Y} = \kappa(X, \bar{A} + Z_K)$.

To train the transition model, we use the ground-truth future motion from the dataset to construct an analytical Gaussian reference transition at each finite step to supervise the learned kernel p_θ. The detailed training procedure is provided in the Appendix B.6.

The learned transition model defines a stochastic generative process that evolves from the kinematic continuation of the observed motion toward plausible future motions in the acceleration-correction space. Because this space represents each future as a boundary-attached correction to the current motion dynamics, control objectives can directly optimize how those dynamics should change through the kinematic operator without repeatedly computing the learned motion model, making it well suited to training-free online control.

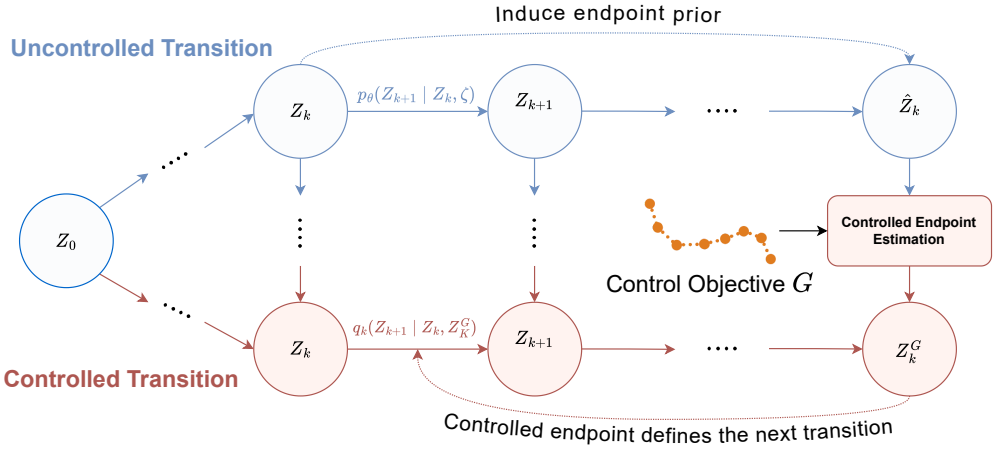


Figure 2: Training-free online transition control. At each finite step k, the predicted latent-conditioned transition induces a local endpoint prior, illustrated by the upper branch. The inference-time objective G shifts its preferred endpoint $\hat{Z}_K$ to Z_K^G, which defines the analytical next-state transition $q_k(Z_{k+1} \mid Z_k, Z_K^G)$ in the lower branch. The sampled Z_{k+1} conditions the subsequent step, where a new prior and controlled endpoint are estimated. The learned transition remains fixed during each endpoint optimization.

2.3 Training-free Online Transition Control

We control the finite transition process introduced in Sec. 2.2 toward future motions that better satisfy an inference-time objective G. At each step, the sampled latent-conditioned transition $p_\theta(Z_{k+1} \mid Z_k, \zeta)$ provides a local motion prior, which we keep fixed while estimating a controlled future endpoint. A future endpoint Z_K has two complementary roles in our formulation. It specifies a complete future motion through $\kappa(X, \bar{A} + Z_K)$, while also determining how the current state Z_k transitions toward that future. This allows us to control each finite transition directly through its future endpoint.

Controlled transition process. For a candidate future endpoint Z_K, we use a linear-Gaussian diffusion bridge (Zhou et al., 2024a; Zheng et al., 2025) to define the stochastic transition from the current state Z_k toward that endpoint:

$$q_k(Z_{k+1} \mid Z_k, Z_K) = \mathcal{N}\big(\alpha_k Z_k + \beta_k Z_K,\, \sigma_k^2 I\big)\,, \tag{4}$$

where α_k, β_k, and σ_k^2 are analytically determined by the chosen reference variance schedule for step k. Thus, selecting a future endpoint also determines the distribution of the next transition

state. In our implementation, we fix the transition covariance to the analytical bridge variance, $\Sigma_\theta(Z_k, \zeta) = \sigma_k^2 I$, and learn only the mean to improve stability.

We formulate transition control by assigning higher probability to future endpoints Z_K that better satisfy G. We define the terminal desirability of an endpoint as $h_K(Z_K) = \exp[-G(\kappa(X, \bar{A} + Z_K))/\lambda]$, where $\lambda > 0$ controls the influence of the objective. Let $p(Z_K \mid Z_k)$ denote the ideal uncontrolled endpoint distribution. Motivated by Doob's h-transform (Doob et al., 1984), we define the corresponding controlled endpoint distribution as

$$p_G(Z_K \mid Z_k) \propto h_K(Z_K)\, p(Z_K \mid Z_k). \tag{5}$$

Since each endpoint Z_K determines both a future motion and the corresponding analytical transition in Eq. 4, $p_G(Z_K \mid Z_k)$ describes the ideal distribution over future motion controlled by the objective G. In other words, the control objective determines which plausible future endpoints should receive higher probability via p_G, which in turn shifts the next transition toward state that leading to those futures more likely. However, the endpoint distribution $p(Z_K \mid Z_k)$ is unknown and intractable at inference, so $p_G(Z_K \mid Z_k)$ cannot be calculated directly. We use the latent-conditioned transition modeled in Sec. 2.2 to construct a tractable local endpoint prior and estimate a controlled endpoint within the sampled transition.

Controlled endpoint estimation. At step k, we sample $\zeta \sim p_\phi(\zeta \mid Z_k)$ and obtain the conditional transition $p_\theta(Z_{k+1} \mid Z_k, \zeta) = \mathcal{N}(Z_{k+1}; \mu_k, \Sigma_k)$. We keep this predicted transition fixed throughout endpoint optimization. Using the analytical transition in Eq. 4, we obtain an approximated endpoint distribution induced by the density of this local transition as:

$$\widehat{Z}_K = \frac{\mu_k - \alpha_k Z_k}{\beta_k}, \qquad \widetilde{p}_k(Z_K \mid Z_k, \zeta) = \mathcal{N}\left(Z_K; \widehat{Z}_K, \frac{\Sigma_k}{\beta_k^2}\right). \tag{6}$$

We use $\widetilde{p}_k$ as a local endpoint prior induced by the predicted transition and estimate the controlled endpoint by

$$Z_K^G = \arg\min_{Z_K} [\underbrace{\frac{1}{\lambda} G\big(\kappa(X, \bar{A} + Z_K)\big)}_{\text{objective satisfaction}} - \underbrace{\log \widetilde{p}_k(Z_K \mid Z_k, \zeta)}_{\text{transition plausibility}}]. \tag{7}$$

For the sampled latent ζ, Eq. 7 can be interpreted as a conditional maximum-a-posteriori (MAP) estimation of $h_K(Z_K)\widetilde{p}_k(Z_K \mid Z_k, \zeta)$ using the transition-induced endpoint surrogate in Eq. 6, which provides a tractable approximation to the controlled endpoint distribution in Eq. 5 At the terminal step, where the transition variance vanishes to zero, we replace the negative log-prior with $\|Z_K - \mu_k\|_2^2$.

The controlled endpoint Z_K^G from Eq. 7 determines the analytical controlled transition $q_k(Z_{k+1} \mid Z_k, Z_K = Z_K^G)$ in Eq. 4, from which we sample the next controlled state Z_{k+1}. This state conditions the subsequent step, where a new latent is sampled from $p_\phi(\zeta \mid Z_{k+1})$ to induce an updated transition prior and controlled endpoint, allowing each transition to adapt to the preceding controlled update and progressively steer the transition chain toward the objective. The sampled latent ζ and predicted transition remain fixed only within the current endpoint optimization. For differentiable G, gradients of the objective term pass through G and the kinematic operator κ, while the local prior is evaluated from the fixed predicted Gaussian. This requires no additional computation of or back-propagation through the learned transition model during controlled endpoint estimation, facilitating efficient online control.

Online generation and control. Algorithm 1 summarizes one online generation and control cycle. Starting from the kinematic continuation with $Z_0 = 0$, each finite step predicts one latent-conditioned transition prior, estimates a controlled endpoint, and samples the corresponding controlled next state. After K transitions, the controlled future is reconstructed as $Y^G = \kappa(X, \bar{A} + Z_K)$ and used to update the observed motion history. A new boundary state and kinematic continuation are then constructed for the next future window under the objective specified at that time. Each cycle requires only K computations of p_ϕ and p_θ, while objective-dependent computation is confined to the endpoint optimization. Because G is introduced only through this inference-time optimization, the same pre-trained transition model can continually respond to new, changing, or combined objectives without control-specific training or model retraining.

Algorithm 1 Training-free online transition control

Input: Motion history X, optional condition $\mathbf{c}$, objective G, transition grid $\{\tau_k\}_{k=0}^{K}$
Output: Controlled future motion Y^G
1: Compute the boundary velocity v_0 and acceleration a_0.
2: Construct the reference acceleration $\bar{A}$ and kinematic continuation $\bar{Y} = \kappa(X, \bar{A})$.
3: Initialize the transition state $Z_0 = 0$.
4: **for** $k = 0, \ldots, K-1$ **do**
5: Sample the transition latent $\zeta \sim p_\phi(\zeta \mid Z_k)$.
6: Predict the local transition prior $p_\theta(Z_{k+1} \mid Z_k, \zeta)$.
7: Estimate the controlled future endpoint Z_K^G using Eq. 7.
8: Sample the next controlled state $Z_{k+1} \sim q_k(Z_{k+1} \mid Z_k, Z_K^G)$.
9: **end for**
10: Reconstruct the controlled future $Y^G = \kappa(X, \bar{A} + Z_K)$.
11: Update the generated sequence and motion history for the next control cycle.

3 EXPERIMENTS

Our base model is trained on AMASS (Mahmood et al., 2019) motion data with online action label from BABEL (Punnakkal et al., 2021), without any control-specific supervision. Unless otherwise stated, all experiments use the same trained model and inference procedure, with control objectives introduced only at inference time. We evaluate whether our KTM can support new, changing, and combined objectives under online real-time execution, including sequential waypoint navigation, moving-obstacle avoidance, 3D end-effector control. Control performance is evaluated in terms of objective satisfaction, motion quality and semantic consistency, and inference efficiency. We compare with representative online motion control methods (Zhao et al., 2025; Zhang et al., 2025) and further investigate the key components of the proposed kinematic transition control.

3.1 ONLINE WAYPOINT NAVIGATION WITH DYNAMIC OBSTACLES

We first evaluate online control under sequential waypoint navigation and dynamic obstacle avoidance. The character is given one waypoint at a time during generation, and after reaching the current waypoint, the next waypoint is provided and becomes the new navigation target, while multiple static and moving obstacles may appear along the route. This setting requires the objective to change as new waypoints are sequentially introduced and to combine goal reaching with obstacle avoidance when necessary. All methods use the same initial motions and environment configurations. Fig. 3(a) visualizes an example online rollout with sequential waypoints and dynamic obstacle avoidance control. We refer to Appendix F for details of experiments, and refer to the supplementary video for qualitative results.

We compare with recent online control methods PRIMAL-G (Zhang et al., 2025) using 10 and 50-step diffusion guidance, and DART-RL (Zhao et al., 2025) using its real-time control policy retrained on the task objective. We also include the uncontrolled baseline as a reference. For control performance, we report waypoint success rate (SR) and collision rate (Coll.). Motion quality is evaluated using neural Riemannian pose distance (ARD) (He et al., 2024) and foot skating following PRIMAL (Zhang et al., 2025). Efficiency is measured by generation frame rate (FPS) and average online control-cycle latency (Latency), reported in seconds.

Table 2: Online sequential-waypoint and dynamic-obstacle control. DART-RL uses a control policy retrained specifically for the navigation task, whereas KTM and PRIMAL-G require no control-specific training. Best results among controlled methods are shown in bold.

			Control Performance		Motion Quality		Efficiency	
Method	Training-Free	Real-Time	SR↑	Coll.↓	ARD ($\times 10^{-3}$)↓	Skate↓	FPS↑	Latency (s)↓
Uncontrolled Reference	–	–	7.63%	28.02%	0.379	0.086	86.76	0.098
PRIMAL-G (10-step)	✓	×	42.28%	28.23%	0.434	0.202	31.82	0.286
PRIMAL-G (50-step)	✓	×	73.01%	25.18%	0.705	0.358	12.26	0.496
DART-RL	×	✓	61.72%	35.04%	0.752	0.189	**92.02**	**0.092**
KTM (ours)	✓	✓	**89.19%**	**18.06%**	**0.384**	**0.129**	52.18	0.134

Table 2 clearly shows our KTM achieves the strongest overall control performance while preserving better motion quality than the competing methods. Particularly, KTM reaches an SR of 89.19%

with only 18.06% collision rate, outperforming PRIMAL-G with 50 diffusion steps (73.01% SR and 25.18% collision) while running more than four times faster. Despite more precise control, KTM also preserves motion quality close to the uncontrolled reference, with an ARD of 0.384 compared with 0.379. PRIMAL-G provides competitive training-free control but incurs high inference cost and does not meet the real-time and desirable control performance, whereas DART-RL achieves the highest generation speed at 92.02 FPS through a control-specific policy but exhibits considerably worse control performance (61.72% SR and 35.04% collision) despite being specifically trained for this control task. In contrast, KTM requires no control-specific training and achieves 89.19% SR at 52.18 FPS with 0.134 s latency, while maintaining real-time execution. These results highlight the gap between flexible training-free control and efficient real-time control in existing works, showing that our KTM effectively achieves both within a single control framework.

3.2 Online End-Effector Control

We further evaluate whether the same training-free control interface can support precise articulated control using the 3D wrist-reaching setting of WANDR (Diomataris et al., 2024). We adapt the original protocol to evaluate sequential online control setting, where the wrist target changes and the character must redirect from its current generated state. The same target sequence is used for all methods. Following WANDR, we additionally compare with GOAL (Taheri et al., 2022). Both GOAL and WANDR obtain reaching capability through task-specific training, whereas KTM receives each wrist target only as a specified inference-time objective. In addition to success rate (SR), we report distance-to-goal (DTG) following WANDR to measure reaching precision. Figure 3(b) shows a representative online sequence where the character adapts to a changing 3D wrist target.

Table 3: Online 3D end-effector control. We adapt the WANDR reaching protocol to sequential targets that change during online generation. GOAL and WANDR learn goal-conditioned interaction or reaching during training, while KTM introduces the wrist target only at inference time.

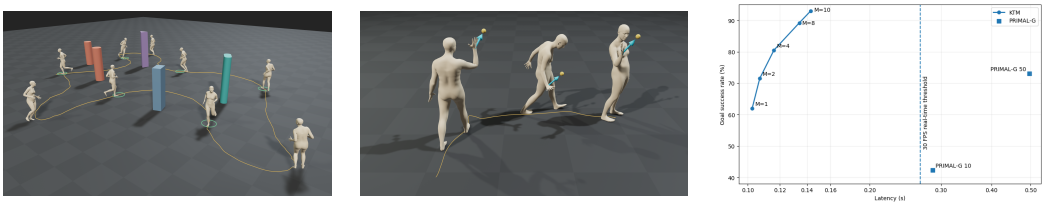
	Setting		Control Performance		Motion Quality		Efficiency	
Method	Training-Free	Real-Time	SR↑	DTG (cm)↓	ARD ($\times 10^{-3}$)↓	Skate↓	FPS↑	Latency (s)↓
GOAL	×	×	4.82%	115.22	0.863	0.351	–	1.209
WANDR	×	✓	35.89%	56.85	0.552	**0.189**	**102.08**	**0.078**
KTM (ours)	✓	✓	**42.15%**	**38.51**	**0.521**	0.208	45.89	0.162

As shown in Table 3, KTM achieves competitive reaching accuracy (with 42.15% SR and 38.51 cm DTG, compared with 35.89% SR and 56.85 cm DTG for WANDR) despite receiving no end-effector control supervision, while maintaining high-quality motion and real-time execution. In contrast, GOAL and WANDR acquire their reaching capability from task-specific and goal-conditioned training. Together with Sec. 3.1, these results validate our training-free control interface across different objective structures. Importantly, all objectives are introduced only at inference and handled using the same trained motion model and control procedure. We note that the broad space of control objectives is inherently open-ended and cannot be exhaustively evaluated in empirical validation, so we consider representative online scenarios spanning global trajectory, environmental constraints, and articulated control. This demonstrates that KTM can be generalized and is not tied to a particular task-specific control module.

3.3 Ablation Study

We investigate the three key components that distinguish our kinematic transition control formulation. *w/o kinematic transition representation* replaces the proposed kinematic transition representation with a pose-space transition process starting from Gaussian noise; *w/o transition prior* removes the learned local prior during control endpoint optimization; *w/o progressive transition control* applies endpoint control optimization for a single-step transition process, instead of progressively across the K finite transitions, under the same total optimization budget. All variants use the same model capacity and are evaluated on the online navigation task.

Table 4 examine the complementary role of each component in our transition control method. Removing the kinematic transition representation degrades both control performance and motion quality, demonstrating the benefit of evolving the future from a boundary-attached kinematic continuation rather than an unstructured pose-space state. Similarly, without the transition prior, the con-



(a) (b) (c)

Figure 3: **Online control tasks and control–efficiency analysis.** (a) Sequential waypoint navigation with dynamic obstacles. (b) Sequential 3D end-effector control with changing wrist targets. (c) Control–efficiency trade-off under different endpoint-optimization budgets; the dashed line denotes the 30 FPS real-time threshold.

Table 4: Ablation of the proposed kinematic transition control on online navigation task.

	Control Performance		Motion Quality	
Model Variant	SR↑	Coll.↓	ARD ($\times 10^{-3}$)↓	Skate ↓
w/o Kinematic Transition Representation	72.84%	28.91%	0.672	0.248
w/o Transition Prior	78.36%	22.74%	0.741	0.286
w/o Progressive Transition Control	58.57%	31.26%	0.426	0.151
KTM (ours)	**89.19%**	**18.06%**	**0.384**	**0.129**

trolled motion becomes less plausible with substantially degraded quality, confirming its role in regularizing objective-driven updates toward learned motion dynamics. Moreover, removing progressive transition control process reduces control accuracy under the same optimization budget, showing that progresively steering the finite transition process is more effective than performing a single endpoint refinement. Together, these results confirm the contributions of the kinematic transition representation, learned transition prior, and the progressive control process for effective trainng-free online motion control.

We further analyze the control–efficiency trade-off by changing the transition control budget M, defined as the maximum number of objective-and-gradient evaluations used to optimize the controlled endpoint at each transition step. As shown in Fig. 3(c), increasing M improves control accuracy at higher inference cost with more latency, while KTM maintains a favorable real-time trade-off compared with PRIMAL's diffusion guidance.

Limitations. Our KTM operates in a kinematic motion space and does not explicitly enforce physical feasibility or interaction dynamics. Each control cycle is performed over short online horizons using a local transition prior under real-time latency constraints, which may limit performance for objectives that require long-term planning or highly precise constraint satisfaction. Finally, the effectiveness of control depends on the coverage and quality of the learned motion prior, particularly for motions outside the training distribution. Potential research directions could extend KTM with physics-aware priors and longer-horizon planning while preserving real-time control, which we leave for future work.

4 Conclusion

We presented KTM, a kinematic transition model for training-free online human motion control under real-time constraints. Instead of treating control as a separate trained module or an expensive refinement stage, KTM makes the generative transition itself controllable. Starting from the observed motion, it constructs a boundary-attached kinematic continuation and models plausible futures through a small number of stochastic transitions in acceleration-correction space. Each predicted transition then provides a local motion prior for optimizing the future under a newly specified objective, allowing control to progressively steer generation without control-specific training or backpropagation through the learned motion model. Experiments on several online control settings demonstrate that the same trained model can respond to changing and previously unseen objectives while maintaining motion quality and real-time execution. We believe this provides a practical step toward general interactive motion synthesis that can adapt to new control requirements at runtime rather than requiring a dedicated controller for each task.

REFERENCES

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Roni Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *ICLR*, 2024.

German Barquero, Sergio Escalera, and Cristina Palmero. Seamless Human Motion Composition with Blended Positional Encodings. In *CVPR*, 2024.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *NeurIPS*, 2024a.

Rui Chen, Mingyi Shi, Shaoli Huang, Ping Tan, Taku Komura, and Xuelin Chen. Taming diffusion probabilistic models for character control. In *SIGGRAPH*, 2024b.

Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. Executing Your Commands via Motion Diffusion in Latent Space. In *CVPR*, pp. 18000–18010, 2023.

Rishabh Dabral, Muhammad Hamza Mughal, Vladislav Golyanik, and Christian Theobalt. MoFusion: A Framework for Denoising-Diffusion-based Motion Synthesis. In *CVPR*, pp. 9760–9770, 2023.

Wenxun Dai, Ling-Hao Chen, Jingbo Wang, Jinpeng Liu, Bo Dai, and Yansong Tang. MotionLCM: Real-time controllable motion generation via latent consistency model. In *ECCV*, 2024.

Markos Diomataris, Nikos Athanasiou, Omid Taheri, Xi Wang, Otmar Hilliges, and Michael J Black. Wandr: Intention-guided human motion generation. In *CVPR*, pp. 927–936, 2024.

Joseph L Doob et al. *Classical potential theory and its probabilistic counterpart*, volume 19. Springer, 1984.

Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2Motion: Conditioned Generation of 3D Human Motions. In *ACM MM*, 2020.

Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating Diverse and Natural 3D Human Motions From Text. In *CVPR*, pp. 5152–5161, 2022.

Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. MoMask: Generative Masked Modeling of 3D Human Motions. In *CVPR*, pp. 1900–1910, 2024.

Wen Guo, Yuming Du, Xi Shen, Vincent Lepetit, Xavier Alameda-Pineda, and Francesc Moreno-Noguer. Back to MLP: A Simple Baseline for Human Motion Prediction. In *WACV*, pp. 4809–4819, 2023.

Yannan He, Garvita Tiwari, Tolga Birdal, Jan Eric Lenssen, and Gerard Pons-Moll. Nrdf: Neural riemannian distance fields for learning articulated pose priors. In *CVPR*, pp. 1661–1671, 2024.

Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. MoGlow: Probabilistic and Controllable Motion Synthesis Using Normalising Flows. *ACM Transactions on Graphics*, 2020.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33: 6840–6851, 2020.

Daniel Holden, Taku Komura, and Jun Saito. Phase-Functioned Neural Networks for Character Control. *ACM Transactions on Graphics*, 2017.

Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. Learned Motion Matching. *ACM Transactions on Graphics*, 2020.

William Huang, Yifeng Jiang, Tom Van Wouwe, and C. Karen Liu. Constrained diffusion with trust sampling. In *NeurIPS*, 2024.

Xiaoyu Huang, Takara Truong, Yunbo Zhang, Fangzhou Yu, Jean Pierre Sleiman, Jessica Hodgins, Koushil Sreenath, and Farbod Farshidian. Diffuse-CLoC: Guided diffusion for physics-based character look-ahead control. *ACM Transactions on Graphics*, 44(4):132:1–132:12, 2025.

Xiaofei Hui, Bo Yan, Haoxuan Qu, Hossein Rahmani, and Jun Liu. Training-free controllable human motion generation under heterogeneous constraints. In *ECCV*, 2026.

Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. MotionGPT: Human Motion as a Foreign Language. In *NeurIPS*, 2023.

Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *ICCV*, pp. 2151–2162, 2023.

Korrawe Karunratanakul, Konpat Preechakul, Emre Aksan, Thabo Beeler, Supasorn Suwajanakorn, and Siyu Tang. Optimizing diffusion noise can serve as universal motion priors. In *CVPR*, 2024.

Tianyu Li, Calvin Qiao, Guanqiao Ren, KangKang Yin, and Sehoon Ha. AAMDM: Accelerated auto-regressive motion diffusion model. In *CVPR*, pp. 1813–1823, 2024.

Qiayuan Liao, Takara E. Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C. Karen Liu. BeyondMimic: From motion tracking to versatile humanoid control via guided diffusion. *Science Robotics*, 11(117):eadx8924, 2026.

Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel van de Panne. Character controllers using motion VAEs. *ACM Transactions on Graphics*, 2020.

Dong C. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

Hanchao Liu, Xiaohang Zhan, Shaoli Huang, Tai-Jiang Mu, and Ying Shan. Programmable motion generation for open-set motion control tasks. In *CVPR*, pp. 1399–1408, 2024.

Hanchao Liu, Fang-Lue Zhang, Shining Zhang, Tai-Jiang Mu, and Shi-Min Hu. Towards highly-constrained human motion generation with retrieval-guided diffusion noise optimization. In *CVPR*, pp. 38291–38301, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Tiezheng Ma, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Progressively generating better initial guesses towards next stages for high-quality human motion prediction. In *CVPR*, pp. 6437–6446, 2022.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *ICCV*, pp. 5442–5451, 2019.

Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction. In *ICCV*, pp. 9489–9497, 2019.

Wei Mao, Miaomiao Liu, and Mathieu Salzmann. History repeats itself: Human motion prediction via motion attention. In *ECCV*, 2020.

Julieta Martinez, Michael J. Black, and Javier Romero. On Human Motion Prediction Using Recurrent Neural Networks. In *CVPR*, pp. 4674–4683, 2017.

Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *CVPR*, pp. 10975–10985, 2019.

Dario Pavllo, David Grangier, and Michael Auli. QuaterNet: A Quaternion-based Recurrent Model for Human Motion. In *BMVC*, 2018.

Mathis Petrovich, Michael J. Black, and Gül Varol. Action-Conditioned 3D Human Motion Synthesis with Transformer VAE. In *ICCV*, 2021.

Mathis Petrovich, Michael J. Black, and Gül Varol. TEMOS: Generating Diverse Human Motions from Textual Descriptions. In *ECCV*, 2022.

Mathis Petrovich, Or Litany, Umar Iqbal, Michael J. Black, Gül Varol, Xue Bin Peng, and Davis Rempe. Multi-Track Timeline Control for Text-Driven 3D Human Motion Generation. In *CVPR Workshops*, 2024.

Ekkasit Pinyoanuntapong, Muhammad Usama Saleem, Korrawe Karunratanakul, Pu Wang, Hongfei Xue, Chen Chen, Chuan Guo, Junli Cao, Jian Ren, and Sergey Tulyakov. MaskControl: Spatio-temporal control for masked motion synthesis. In *ICCV*, 2025.

Abhinanda R Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J Black. Babel: Bodies, action and behavior with english labels. In *CVPR*, pp. 722–731, 2021.

Shiyi Qi, Kun He, and Mingmou Liu. SDDBMs: Soft denoising diffusion bridge models. *arXiv preprint arXiv:2608.08594*, 2026.

Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. HuMoR: 3D human motion model for robust pose estimation. In *ICCV*, 2021.

Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler, Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne, Jan Kautz, Simon Yuen, and Sanja Fidler. Kimodo: Scaling controllable human motion generation. *arXiv preprint arXiv:2603.15546*, 2026.

Yonatan Shafir, Guy Tevet, Roy Kapon, and Amit H. Bermano. Human Motion Diffusion as a Generative Prior. In *ICLR*, 2024.

Neta Shaul, Uriel Singer, Itai Gat, and Yaron Lipman. Transition matching: Scalable and flexible generative modeling. In *NeurIPS*, 2025.

Yi Shi, Jingbo Wang, Xuekun Jiang, Bingkun Lin, Bo Dai, and Xue Bin Peng. Interactive character control with auto-regressive motion diffusion models. *ACM Transactions on Graphics*, 43(4), 2024.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *NeurIPS*, 2015.

Sebastian Starke, Ian Mason, and Taku Komura. DeepPhase: Periodic Autoencoders for Learning Motion Phase Manifolds. *ACM Transactions on Graphics*, 41(4), 2022.

Omid Taheri, Vasileios Choutas, Michael J Black, and Dimitrios Tzionas. Goal: Generating 4d whole-body motion for hand-object grasping. In *CVPR*, pp. 13253–13263, 2022.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H. Bermano. Human Motion Diffusion Model. In *ICLR*, 2023.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit Bermano, and Michiel van de Panne. CLoSD: Closing the loop between simulation and diffusion for multi-task character control. In *ICLR*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Weilin Wan, Zhiyang Dou, Taku Komura, Wenping Wang, Dinesh Jayaraman, and Lingjie Liu. Tlcontrol: Trajectory and language control for human motion synthesis. In *ECCV*, pp. 37–54, 2024.

Tingwu Wang, Olivier Dionne, Michael De Ruyter, David Minor, Davis Rempe, Kaifeng Zhao, Mathis Petrovich, Ye Yuan, Chenran Li, Zhengyi Luo, Brian Robison, Xavier Blackwell, Bernardo Antoniazzi, Xue Bin Peng, Yuke Zhu, and Simon Yuen. MotionBricks: Scalable real-time motions with modular latent generative model and smart primitives. *ACM Transactions on Graphics*, 2026.

Yixiao Wang, Chen Tang, Lingfeng Sun, Simone Rossi, Yichen Xie, Chensheng Peng, Thomas Hannagan, Stefano Sabatini, Nicola Poerio, Masayoshi Tomizuka, and Wei Zhan. Optimizing diffusion models for joint trajectory prediction and controllable generation. In *ECCV*, 2024a.

Zhenzhi Wang, Jingbo Wang, Yixuan Li, Dahua Lin, and Bo Dai. Intercontrol: Zero-shot human interaction generation by controlling every joint. In *NeurIPS*, 2024b.

Akihisa Watanabe, Qing Yu, Edgar Simo-Serra, and Kent Fujiwara. Projflow: Projection sampling with flow matching for zero-shot exact spatial motion control. *arXiv preprint arXiv:2602.22742*, 2026.

Lixing Xiao, Shunlin Lu, Huaijin Pi, Ke Fan, Liang Pan, Yueer Zhou, Ziyong Feng, Xiaowei Zhou, Sida Peng, and Jingbo Wang. MotionStreamer: Streaming motion generation via diffusion-based autoregressive model in causal latent space. In *ICCV*, pp. 10086–10096, 2025.

Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion GANs. In *ICLR*, 2022.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. OmniControl: Control any joint at any time for human motion generation. In *ICLR*, 2024.

Qing Yu, Akihisa Watanabe, and Kent Fujiwara. Causal Motion Diffusion Models for Autoregressive Motion Generation. In *CVPR*, 2026a.

Zichao Yu, Ming Li, Wenyi Zhang, Difan Zou, and Weiguo Gao. Proflow: Zero-shot physics-consistent sampling via proximal flow guidance. *arXiv preprint arXiv:2601.20227*, 2026b.

Ye Yuan and Kris Kitani. DLow: Diversifying Latent Flows for Diverse Human Motion Prediction. In *ECCV*, 2020.

Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. PhysDiff: Physics-guided human motion diffusion model. In *ICCV*, 2023.

Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating Human Motion From Textual Descriptions With Discrete Representations. In *CVPR*, pp. 14730–14740, 2023.

Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. MotionDiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*, 2022.

Yan Zhang, Michael J. Black, and Siyu Tang. We are More than Our Joints: Predicting how 3D Bodies Move. In *CVPR*, pp. 3371–3381, 2021.

Yan Zhang, Yao Feng, Alpar Cseke, Nitin Saini, Nathan Bajandas, Nicolas Heron, and Michael J. Black. PRIMAL: Physically reactive and interactive motor model for avatar learning. In *ICCV*, 2025.

Yiming Zhang, Baojia Han, Weichen Wang, Ximing Li, Fausto Giunchiglia, Bo Yang, Wei Pang, Xiaoyue Feng, and Renchu Guan. HumanBBD: Human motion prediction with brownian bridge diffusion. *Neurocomputing*, 698:134309, 2026.

Kaifeng Zhao, Gen Li, and Siyu Tang. DartControl: A diffusion-based autoregressive motion model for real-time text-driven motion control. In *ICLR*, 2025.

Kaifeng Zhao, Mathis Petrovich, Haotian Zhang, Tingwu Wang, Siyu Tang, and Davis Rempe. ARDY: Autoregressive diffusion with hybrid representation for interactive human motion generation. *ACM Transactions on Graphics*, 2026.

Kaiwen Zheng, Guande He, Jianfei Chen, Fan Bao, and Jun Zhu. Diffusion bridge implicit models. In *ICLR*, 2025.

Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. In *ICLR*, 2024a.

Wenyang Zhou, Zhiyang Dou, Zeyu Cao, Zhouyingcheng Liao, Jingbo Wang, Wenjia Wang, Yuan Liu, Taku Komura, Wenping Wang, and Lingjie Liu. EMDM: Efficient motion diffusion model for fast, high-quality human motion generation. In *ECCV*, 2024b.

A Related Work

Interactive kinematic motion generation. Human motion generation has long been studied through conditional latent models, diffusion models, and token-based autoregressive models (Guo et al., 2020; 2022; Petrovich et al., 2021; 2022; Tevet et al., 2023; Dabral et al., 2023; Zhang et al., 2022; Barquero et al., 2024; Chen et al., 2023; Zhang et al., 2023; Jiang et al., 2023; Guo et al., 2024), while earlier interactive character animation systems focus on synthesizing controllable motion from trajectory, motion phase, or user-command inputs (Holden et al., 2017; Henter et al., 2020; Ling et al., 2020; Holden et al., 2020). Recent autoregressive motion models further support online synthesis by continually predicting future motion from the generated history (Shi et al., 2024; Li et al., 2024; Chen et al., 2024b; Zhao et al., 2025; Zhang et al., 2025; Xiao et al., 2025; Yu et al., 2026a; Wang et al., 2026; Zhao et al., 2026). However, real-time controllability in these methods is commonly achieved through trained conditioning or specialized control interfaces. CAMDM (Chen et al., 2024b) and AAMDM (Li et al., 2024) use learned conditioning or predefined query guidance. PRIMAL (Zhang et al., 2025) derives efficient analytical guidance for movement speed and facing direction while using adaptation modules for additional control tasks. MotionBricks (Wang et al., 2026) uses smart primitives to translate user commands or game events into keyframe constraints, and ARDY (Zhao et al., 2026) learns text and kinematic constraint conditioning. Conversely, more flexible training-free control can instead introduce unseen objectives at inference, but typically incurs substantial computation cost during generation. A-MDM (Shi et al., 2024) evaluates candidate futures under a user-defined score, while DART (Zhao et al., 2025) optimizes latent diffusion noise for general spatial objectives, with reinforcement learning used as an alternative for online control. Their inference-time control interface are not demonstrated to operate in real time during online generation. Similarly, offline methods such as DNO (Karunratanakul et al., 2024), RGDNO (Liu et al., 2026), or MaskControl (Pinyoanuntapong et al., 2025) support objectives introduced at inference through iterative noise or token-logit optimization, but do not address continual online generation. Thus, existing training-free control methods for unseen objectives do not demonstrate continual real-time generation, while existing real-time online control methods rely on trained or specialized control interfaces rather than generic training-free control for unseen objectives. We aim to address both requirements within a unified framework. Our focus is training-free online kinematic motion control that can continually respond to unseen differentiable objectives using the same pre-trained motion model and inference-time control process under an end-to-end real-time online budget.

Kinematic motion priors and stochastic transitions. Human motion priors and prediction models have explored temporal structure through residual, dynamical, and stochastic motion representations (Martinez et al., 2017; Pavllo et al., 2018; Mao et al., 2019; 2020; Ma et al., 2022; Ling et al., 2020; Rempe et al., 2021; Yuan & Kitani, 2020; Zhang et al., 2021; Starke et al., 2022; Guo et al., 2023; Shafir et al., 2024). For efficient stochastic generation, DDGAN (Xiao et al., 2022) models multimodal large-step denoising transitions, while EMDM (Zhou et al., 2024b) applies latent-conditioned multimodal transitions to few-step human motion generation. Transition Matching (Shaul et al., 2025) further formulates generative modeling through learned finite stochastic transition kernels. Related diffusion bridge models connect informative source and target distributions (Zhou et al., 2024a; Zheng et al., 2025; Qi et al., 2026), while HumanBBD (Zhang et al., 2026) projects the observed source and target sequence into a DCT latent space and applies Brownian-bridge diffusion between them for stochastic motion prediction. These works provide mechanisms for stochastic generation, large-step transitions, and general source-to-target modeling. KTM defines a motion-specific stochastic transition process in a kinematic space induced by the observed motion for fast and adaptive online control. The process starts from the kinematic future implied by the observed boundary dynamics and evolves in an acceleration-correction space, so every transition state explicitly represents how the upcoming motion can change from its current dynamics toward plausible actual futures. Each transition state is directly reconstructable as motion through the same kinematic operator, while each learned finite transition provides an probabilistic motion prior of these future-dynamics changes, which is used for training-free online control.

Training-free motion guidance and control. Controllable human motion generation has been studied through learned conditioning, spatial guidance, and inference-time refinement (Karunratanakul et al., 2023; Xie et al., 2024; Dai et al., 2024; Wan et al., 2024; Wang et al., 2024b; Petrovich et al., 2024; Yuan et al., 2023; Tevet et al., 2025; Rempe et al., 2026). More flexible training-free

methods introduce control objectives directly at inference. ProgMoGen (Liu et al., 2024) expresses motion requirements as programmable objectives, DNO (Karunratanakul et al., 2024) optimizes diffusion noise under objectives defined on the generated motion, and MaskControl (Pinyoanuntapong et al., 2025) performs zero-shot control through token-logit optimization. Trust Sampling (Huang et al., 2024) balances constraint satisfaction with the learned motion prior, while ProjFlow (Watanabe et al., 2026) projects predicted clean motions under kinematic spatial constraints and MIC (Hui et al., 2026) supports heterogeneous inference-time constraints through stochastic control. Related general generative diffusion guidance methods optimize estimated clean or terminal states and transfer the resulting corrections to the sampling process (Bansal et al., 2024; Wang et al., 2024a; Yu et al., 2026b; Chen et al., 2024a), but are not developed specifically for human motion generation or online control. Recent physics-based methods, including Diffuse-CLoC (Huang et al., 2025) and BeyondMimic (Liao et al., 2026), also support inference-time objectives for real-time control. However, these methods require learning state–action policies from physics-based simulation or distilled motion-tracking controllers, whereas we focus on kinematic synthesis from real motion data, without physics-based tracking policy in simulated environment. We perform training-free control directly on the learned finite kinematic transitions in the boundary-attached acceleration-correction space. At each online control cycle, the predicted transition is kept fixed as a local motion prior while its future endpoint is optimized under the control objective. The resulting controlled endpoint analytically defines the next controlled transition, enabling new or changing objectives without control-specific training or additional motion-model computation during optimization.

B Finite Transition Model: Derivation and Learning

Overview. The purpose of this section is to explain how the finite transition model is constructed and learned. During training, each ground-truth future provides a known endpoint, allowing us to analytically construct the bridge transition toward that future. At inference, however, the endpoint is unknown and may correspond to many plausible futures. To handle this, we use the endpoint-conditioned bridge transitions to learn their marginalized transition distribution through a latent model. We first derive the analytical finite transition in Sec. B.2, then define the marginalized learning target and training objective in Sec B.3 and B.4, followed by its theoretical justification in Sec. B.5.

For readers primarily interested in our practical training and generation process, we highly encourage referring directly to Sec. B.6, which summarizes the complete training and inference algorithms without requiring the theoretical derivations.

B.1 Notation and Background on Diffusion Bridge

Table 5: Main notation for the finite transition process.

Symbol	Meaning
$f = 1, \dots, F$	Motion-frame index within the future window
Δt	Physical time interval between motion frames
$\tau \in [0, 1]$	Continuous bridge time
$\{\tau_k\}_{k=0}^{K}$	Finite bridge-time grid, with $\tau_0 = 0$ and $\tau_K = 1$
$k = 0, \dots, K$	Finite transition-step index
Z_τ	Kinematic transition state at continuous bridge time τ
$Z_k := Z_{\tau_k}$	Kinematic state at finite transition step k
$Z^\star$	Ground-truth terminal state available during training
Z_K	Terminal state produced by the K-step transition process
s, t	Arbitrary bridge times satisfying $0 \le s < t \le 1$
$q_{s\to t}$	Analytical endpoint-conditioned reference transition
q_k	Reference transition from τ_k to τ_{k+1}
Q_k	Endpoint-marginalized target transition
P_k	Learned latent-marginalized transition

Notation. Our formulation involves two different notions of time. Motion-frame time describes the actual human motion in the physical time axis, while bridge time parameterizes the stochastic

transition process used to generate that motion. We use $f \in \{1, \dots, F\}$ for the motion-frame index and Δt for the physical time interval between motion frames. Separately, we use $\tau \in [0, 1]$ for continuous bridge time and discretize it using $0 = \tau_0 < \tau_1 < \cdots < \tau_K = 1$. The index k denotes a finite transition step and is unrelated to the motion-frame index f. Table 5 summarizes the main notation used throughout the paper.

When discussing the continuous bridge, its terminal state occurs at $\tau = 1$. During training this terminal state is the ground-truth correction $Z^\star$, while on the finite grid it corresponds to the final state Z_K. Therefore, we avoid using the notation Z_1 for the continuous endpoint, since Z_1 already denotes the first discrete transition state in the finite process. For arbitrary bridge times s and t, Z_s and Z_t denote the corresponding continuous bridge states.

Diffusion bridge. A diffusion bridge describes a stochastic process that is conditioned to start from a given source state and reach a specified terminal state. (Zhou et al., 2024a; Zheng et al., 2025) Consider a standard reference diffusion process

$$dZ_\tau = f(Z_\tau, \tau)\, d\tau + g(\tau)\, dW_\tau, \qquad \tau \in [0, 1]. \tag{8}$$

Conditioning this process on a fixed terminal endpoint $Z_{\tau=1} = Z^\star$ gives a diffusion bridge. Let $h^\star_\tau(Z_\tau) = p_{\text{ref}}(Z^\star \mid Z_\tau)$ denote the reference transition density evaluated at the specified terminal endpoint. Doob's h-transform (Doob et al., 1984) gives the endpoint-conditioned process as

$$dZ_\tau = \left[f(Z_\tau, \tau) + g(\tau)^2 \nabla_{Z_\tau} \log h^\star_\tau(Z_\tau)\right] d\tau + g(\tau)\, dW_\tau. \tag{9}$$

This conditioning changes the process so that it reaches $Z^\star$ at $\tau = 1$, while intermediate states remain stochastic.

For a Markov reference process, the same bridge can be described directly through its transition distribution. Specifically, for arbitrary bridge times $0 \le s < t < 1$,

$$q_{s\to t}(Z_t \mid Z_s, Z^\star) = \frac{p_{\text{ref}}(Z_t \mid Z_s)\, p_{\text{ref}}(Z^\star \mid Z_t)}{p_{\text{ref}}(Z^\star \mid Z_s)}. \tag{10}$$

Our model uses a linear-Gaussian reference process, for which this conditional transition has an analytical Gaussian form. We derive its mean and variance for an arbitrary interval $s \to t$ in Sec. B.2. This analytical transition is then evaluated on the finite grid $\{\tau_k\}_{k=0}^K$ to construct the reference transition kernels used for training our model. The main paper uses the same finite transition formulation for generation and control.

Doob's h-transform for terminal desirability. The same h-transform is not limited to conditioning on one fixed endpoint. More generally, we can define a positive terminal desirability function $h_1(z) > 0$ over possible terminal states z at bridge time $\tau = 1$ (Doob et al., 1984). The desirability of an intermediate state is

$$h_s(Z_s) = \mathbb{E}[h_1(Z_{\tau=1}) \mid Z_s]\,. \tag{11}$$

The transformed transition from s to t is

$$P^h_{s\to t}(Z_t \mid Z_s) = P_{s\to t}(Z_t \mid Z_s) \frac{h_t(Z_t)}{h_s(Z_s)}. \tag{12}$$

Thus, states that are more likely to lead to desirable terminal outcomes receive higher probability under the transformed process.

B.2 Derivation of the Analytical Finite Bridge Transitions

We now derive the analytical transition used to construct each finite transition step. Starting from a current bridge state Z_s, the reference process gives a distribution over a later state Z_t. When the target endpoint $Z^\star$ is known, this distribution is conditioned toward states that can also reach $Z^\star$ at bridge time $\tau = 1$.

Reference transition. Following the linear-Gaussian reference diffusion formulation used in DDBMs (Zhou et al., 2024a), we consider the isotropic linear process

$$dZ_\tau = \ell(\tau) Z_\tau\, d\tau + g(\tau)\, dW_\tau, \tag{13}$$

where $\ell(\tau)$ and $g(\tau)$ define the reference drift and noise schedule. Its transition between arbitrary bridge times $s < t$ is Gaussian, with the variance schedule:

$$\boxed{\bar{\alpha}_{s\to t} = \exp\left(\int_s^t \ell(r)\, dr\right), \qquad \bar{\sigma}^2_{s\to t} = \int_s^t \bar{\alpha}^2_{r\to t} g(r)^2\, dr.} \tag{14}$$

This defines the reference Gaussian transition from s to t

$$p_{\text{ref}}(Z_t \mid Z_s) = \mathcal{N}\big(Z_t;\, \bar{\alpha}_{s\to t} Z_s,\, \bar{\sigma}^2_{s\to t} I\big)\,. \tag{15}$$

Similarly, from an intermediate state Z_t to the terminal state at $t = 1$, we have the following transition

$$p_{\text{ref}}(Z^\star \mid Z_t) = \mathcal{N}\big(Z^\star;\, \bar{\alpha}_{t\to 1} Z_t,\, \bar{\sigma}^2_{t\to 1} I\big)\,. \tag{16}$$

Conditioning on the target endpoint. During training, the ground-truth endpoint $Z^\star$ is available from the training dataset. The bridge transition from an arbitrary state Z_s to Z_t is

$$q_{s\to t}(Z_t \mid Z_s, Z^\star) \propto \underbrace{p_{\text{ref}}(Z_t \mid Z_s)}_{\text{transition from the current state}} \times \underbrace{p_{\text{ref}}(Z^\star \mid Z_t)}_{\text{ability to reach the endpoint}}\,. \tag{17}$$

The first term describes where the process can move from Z_s, while the second favors intermediate states that can still reach the target endpoint. Since both terms are Gaussian, their product is also Gaussian.

Substituting the reference transitions gives

$$q_{s\to t}(Z_t \mid Z_s, Z^\star) \propto \exp\left[-\frac{1}{2}\left(\frac{\|Z_t - \bar{\alpha}_{s\to t} Z_s\|_2^2}{\bar{\sigma}^2_{s\to t}} + \frac{\|Z^\star - \bar{\alpha}_{t\to 1} Z_t\|_2^2}{\bar{\sigma}^2_{t\to 1}}\right)\right]. \tag{18}$$

Collecting the quadratic and linear terms in Z_t gives the variance

$$\frac{1}{\sigma^2_{s\to t}} = \frac{1}{\bar{\sigma}^2_{s\to t}} + \frac{\bar{\alpha}^2_{t\to 1}}{\bar{\sigma}^2_{t\to 1}}, \tag{19}$$

and the corresponding Gaussian mean is

$$\sigma^2_{s\to t}\left(\frac{\bar{\alpha}_{s\to t} Z_s}{\bar{\sigma}^2_{s\to t}} + \frac{\bar{\alpha}_{t\to 1} Z^\star}{\bar{\sigma}^2_{t\to 1}}\right) = \alpha_{s\to t} Z_s + \beta_{s\to t} Z^\star. \tag{20}$$

Therefore, the endpoint-conditioned finite transition is

$$q_{s\to t}(Z_t \mid Z_s, Z^\star) = \mathcal{N}\big(Z_t;\, \alpha_{s\to t} Z_s + \beta_{s\to t} Z^\star,\, \sigma^2_{s\to t} I\big)\,. \tag{21}$$

Using

$$\bar{\sigma}^2_{s\to 1} = \bar{\alpha}^2_{t\to 1} \bar{\sigma}^2_{s\to t} + \bar{\sigma}^2_{t\to 1}, \tag{22}$$

the coefficients simplify to

$$\boxed{\alpha_{s\to t} = \frac{\bar{\alpha}_{s\to t}\bar{\sigma}^2_{t\to 1}}{\bar{\sigma}^2_{s\to 1}}, \qquad \beta_{s\to t} = \frac{\bar{\alpha}_{t\to 1}\bar{\sigma}^2_{s\to t}}{\bar{\sigma}^2_{s\to 1}}, \qquad \sigma^2_{s\to t} = \frac{\bar{\sigma}^2_{s\to t}\bar{\sigma}^2_{t\to 1}}{\bar{\sigma}^2_{s\to 1}}} \tag{23}$$

Equation 21 has an intuitive interpretation. The transition mean is a weighted combination between the current state Z_s and the target endpoint $Z^\star$. As the process approaches the terminal time, the transition becomes more influenced by $Z^\star$, while its variance decreases to zero.

Sampling an intermediate state for training. To supervise the transition at bridge time τ_k, we first need a current state Z_k. Since the process starts from $Z_0 = 0$, we obtain its distribution by setting $s = 0$ and $t = \tau_k$ in Eq. 21:

$$q_{0\to\tau_k}(Z_k \mid Z_0 = 0, Z^\star) = \mathcal{N}\big(Z_k;\, \beta_{0\to\tau_k} Z^\star,\, \sigma^2_{0\to\tau_k} I\big)\,. \tag{24}$$

We directly sample Z_k from this analytical distribution during training, without generating the preceding transition steps.

Finite transition steps. Our model uses the fixed bridge-time grid $0 = \tau_0 < \tau_1 < \cdots < \tau_K = 1$. Calculating the analytical bridge coefficients (Eq. 23) on this fixed bridge-time grid defines the finite transition schedule used throughout our main paper:

$$\boxed{\alpha_k := \alpha_{\tau_k \to \tau_{k+1}}, \qquad \beta_k := \beta_{\tau_k \to \tau_{k+1}}, \qquad \sigma_k^2 := \sigma^2_{\tau_k \to \tau_{k+1}}} \tag{25}$$

The analytical reference transition for step k is

$$q_k(Z_{k+1} \mid Z_k, Z^\star) = \mathcal{N}\big(Z_{k+1};\, \alpha_k Z_k + \beta_k Z^\star,\, \sigma_k^2 I\big)\,. \tag{26}$$

where $q_k := q_{\tau_k \to \tau_{k+1}}$ denotes the reference transition over the k-th finite bridge interval.

At the final step, the bridge reaches its prescribed endpoint. Thus, $\alpha_{K-1} = 0$, $\beta_{K-1} = 1$, and $\sigma^2_{K-1} = 0$, giving $Z_K = Z^\star$. We handle this deterministic step separately in the training objective.

In summary, the same analytical bridge provides the two distributions needed during training:

$$\boxed{\begin{aligned} q_{0\to\tau_k}(Z_k \mid Z_0 = 0, Z^\star) \quad & \text{samples the current transition state,} \\ q_k(Z_{k+1} \mid Z_k, Z^\star) \quad & \text{provides its target transition kernel.} \end{aligned}} \tag{27}$$

B.3 Marginal Transition and Learned Kernel

The analytical transition in Eq. 26 is conditioned on a known target endpoint $Z^\star$. This endpoint is available during training, but it is unknown at inference, where the observed motion may correspond to many plausible futures. Consequently, the transition model needs to capture the possible next states after accounting for these different future endpoints.

For clarity, we omit additional conditioning inputs to the learned distributions, such as the observed motion history and optional condition. Their detailed network inputs are described in the implementation details.

Marginal reference transition kernel. At transition step k, different plausible futures correspond to different target endpoints $Z^\star$. Marginalizing the analytical reference transition over these possible endpoints gives

$$\boxed{Q_k(Z_{k+1} \mid Z_k) = \int q_k(Z_{k+1} \mid Z_k, Z^\star)\, p_{\text{data}}(Z^\star \mid Z_k)\, dZ^\star.} \tag{28}$$

Here, $p_{\text{data}}(Z^\star \mid Z_k)$ denotes the distribution of plausible future endpoints that given the current intermediate transition state. More precisely, this conditional distribution is induced by first sampling a ground-truth future from the motion data and then sampling Z_k from its endpoint-conditioned analytical bridge in Eq. 24.

Each endpoint $Z^\star$ defines a different Gaussian reference transition through Eq. 26. Therefore, although each endpoint-conditioned transition is Gaussian, the marginalized transition Q_k can represent multiple plausible transition outcomes and is generally not Gaussian. We use Q_k as the target transition distribution that the model aims to learn.

Learned transition prior. Since the possible future endpoints are unknown at inference, we model the marginal transition using a latent variable ζ:

$$\boxed{P_k(Z_{k+1} \mid Z_k) = \int p_\theta(Z_{k+1} \mid Z_k, \zeta)\, p_\phi(\zeta \mid Z_k)\, d\zeta.} \tag{29}$$

The latent prior $p_\phi(\zeta \mid Z_k)$ captures the variation among possible transitions from the current state. Given a sampled latent ζ, the conditional transition is

$$p_\theta(Z_{k+1} \mid Z_k, \zeta) = \mathcal{N}(Z_{k+1}; \mu_\theta(Z_k, \zeta), \Sigma_\theta(Z_k, \zeta))\,, \tag{30}$$

where μ_θ and Σ_θ are the predicted transition mean and variance, respectively. In our implementation, we fix the conditional transition variance to the reference bridge variance, $\Sigma_\theta(Z_k, \zeta) = \sigma_k^2 I$, and

learn only the transition mean. We found this makes training more stable than directly learning the transition variance.

Different latent samples can induce different conditional Gaussian transition distributions, allowing their marginal $P_k(Z_{k+1} \mid Z_k)$ to represent multiple plausible ways that the current state may evolve. Our learning objective is to make the learned transition approximate the target marginal transition:

$$P_k(Z_{k+1} \mid Z_k) \approx Q_k(Z_{k+1} \mid Z_k). \tag{31}$$

The future endpoint $Z^\star$ is used only during training to construct the analytical reference transitions. The next subsection explains how these endpoint-conditioned transitions are used to train P_k without requiring the future endpoint at inference.

B.4 Transition Kernel Learning

Recall from Eq. 31 that our key learning target is the marginal transition $Q_k(Z_{k+1} \mid Z_k)$, while the latent transition model defines its learned approximation $P_k(Z_{k+1} \mid Z_k)$.

Directly evaluating Q_k would require integrating over all possible future endpoints, which is intractable. During training, however, each ground-truth future provides one target endpoint $Z^\star$, together with its analytical endpoint-conditioned transition $q_k(Z_{k+1} \mid Z_k, Z^\star)$. We use these available transitions to learn the marginal transition without explicitly evaluating the endpoint integral.

For a randomly sampled transition step k, we first sample the current state

$$Z_k \sim q_{0\to\tau_k}(Z_k \mid Z_0 = 0, Z^\star) \tag{32}$$

from the analytical bridge in Eq. 24. The same endpoint $Z^\star$ then defines the target transition $q_k(Z_{k+1} \mid Z_k, Z^\star)$.

Since the marginal transition in Eq. 28 (our key learning target) depends on the unknown distribution of future endpoints $p_{\text{data}}(Z^\star \mid Z_k)$, we leverage the ground-truth endpoint available in each training sample to capture this endpoint variation into a latent space. Inspired by conditional variational latent learning (Sohn et al., 2015), we use a training-only endpoint encoder $e_\omega(\zeta \mid Z_k, Z^\star)$ that maps the current state and the observed ground-truth endpoint to a future-informed latent distribution. A sampled latent ζ then conditions $p_\theta(Z_{k+1} \mid Z_k, \zeta)$ to predict the transition associated with that future.

Unlike a standard conditional VAE Sohn et al. (2015) that directly encodes and then reconstructs the example training motion, our latent space conditions a finite stochastic transition kernel. The endpoint encoder is used to associate the observed future with its endpoint-conditioned reference transition, while the latent prior and conditional transition jointly approximate the marginal transition kernel Q_k.

Transition learning objective. The endpoint encoder, latent prior, and conditional transition kernel are jointly trained using

$$\boxed{\mathcal{L}_{\text{KTM}} = \mathcal{L}_{\text{trans}} + \mathcal{L}_{\text{align}},} \tag{33}$$

where

$$\mathcal{L}_{\text{trans}} = \mathbb{E}\left[D_{\text{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) \,\|\, p_\theta(Z_{k+1} \mid Z_k, \zeta))\right], \tag{34}$$

with $\zeta \sim e_\omega(\zeta \mid Z_k, Z^\star)$, and

$$\mathcal{L}_{\text{align}} = \mathbb{E}\left[D_{\text{KL}}(e_\omega(\zeta \mid Z_k, Z^\star) \,\|\, p_\phi(\zeta \mid Z_k))\right]. \tag{35}$$

The expectations are taken over the training examples, sampled transition step k, sampled intermediate state Z_k, and latent ζ where applicable.

The transition loss $\mathcal{L}_{\text{trans}}$ train the predicted conditional transition to match the analytical reference transition associated with the ground-truth future. The latent alignment loss $\mathcal{L}_{\text{align}}$ aligns the future-informed encoder distribution with the latent prior $p_\phi(\zeta \mid Z_k)$ available at inference.

At inference, the endpoint $Z^\star$ is unknown and the endpoint encoder cannot be used. Instead, transition latents are sampled directly from $p_\phi(\zeta \mid Z_k)$. The alignment loss allows the latent prior to capture the transition variation observed through the endpoint encoder during training. The endpoint

encoder is used only as an auxiliary training component and is not part of the learned inference-time transition $P_k(Z_{k+1} \mid Z_k)$.

At the final transition step, the analytical bridge reaches the target endpoint deterministically, so its variance becomes zero. We replace $\mathcal{L}_{\text{trans}}$ at this step with the endpoint matching loss $\mathcal{L}_{\text{end}} = \|\mu_\theta(Z_{K-1}, \zeta) - Z^\star\|_2^2$. The latent alignment loss remains unchanged at the terminal step. Inference requires only the latent prior p_ϕ and the learned conditional transition kernel p_θ.

B.5 Theoretical Insight into our Transition Learning Objective

Our training objective uses the analytical transition associated with one observed endpoint $Z^\star$ at a time to supervise the learned kernel, while the target transition $Q_k(Z_{k+1} \mid Z_k)$ marginalizes over all plausible future endpoints. Here, we show that they are connected and provide a theoretical justification for our transition kernel learning objective. Specifically, the training objective provides a tractable upper bound on the divergence between the learned transition P_k and the desired marginal transition Q_k, without explicitly constructing the intractable Q_k. When this bound is tight and under ideal conditions, the learned transition recovers the target marginal transition (up to a model-independent constant).

Proposition 1. ***(lower bound for our transition learning objective).*** *For any non-terminal transition step k and current state Z_k,*

$$\begin{aligned}
&\mathbb{E}_{Z^\star \mid Z_k}\Big[\mathbb{E}_{\zeta \sim e_\omega(\zeta \mid Z_k, Z^\star)}\left[D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) \,\|\, p_\theta(Z_{k+1} \mid Z_k, \zeta))\right] \\
&\qquad\qquad + D_{\mathrm{KL}}(e_\omega(\zeta \mid Z_k, Z^\star) \,\|\, p_\phi(\zeta \mid Z_k))\Big] \\
&\geq D_{\mathrm{KL}}(Q_k(Z_{k+1} \mid Z_k) \,\|\, P_k(Z_{k+1} \mid Z_k)) \\
&\qquad + \underbrace{\mathbb{E}_{Z^\star \mid Z_k}\left[D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) \,\|\, Q_k(Z_{k+1} \mid Z_k))\right]}_{\textit{constant w.r.t. the learned model}}.
\end{aligned} \tag{36}$$

The last term on the right depends only on the target transition distribution and is constant with respect to the learned model parameters.

Or informally, Eq. 36 can be viewed as

$$\boxed{\begin{aligned}&\text{transition kernel learning objective} \;\geq\; \text{marginal transition error} \\ &\qquad\qquad\qquad\qquad + \text{constant independent of the learned model.}\end{aligned}} \tag{37}$$

Proof. For a given endpoint $Z^\star$, the transition $\mathcal{L}_{\text{trans}}$ and latent alignment $\mathcal{L}_{\text{align}}$ terms can be written together as

$$\begin{aligned}
&\mathbb{E}_{\zeta \sim e_\omega(\zeta \mid Z_k, Z^\star)}\left[D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) \,\|\, p_\theta(Z_{k+1} \mid Z_k, \zeta))\right] \\
&\quad + D_{\mathrm{KL}}(e_\omega(\zeta \mid Z_k, Z^\star) \,\|\, p_\phi(\zeta \mid Z_k)) \\
&= D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) e_\omega(\zeta \mid Z_k, Z^\star) \,\|\, p_\theta(Z_{k+1} \mid Z_k, \zeta) p_\phi(\zeta \mid Z_k)).
\end{aligned} \tag{38}$$

Marginalizing the latent variable ζ cannot increase the KL divergence. Since marginalizing the learned joint distribution gives $P_k(Z_{k+1} \mid Z_k)$, we obtain

$$\begin{aligned}
&\mathbb{E}_{Z^\star \mid Z_k}\Big[\mathbb{E}_{\zeta \sim e_\omega(\zeta \mid Z_k, Z^\star)}\left[D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) \,\|\, p_\theta(Z_{k+1} \mid Z_k, \zeta))\right] \\
&\qquad\qquad + D_{\mathrm{KL}}(e_\omega(\zeta \mid Z_k, Z^\star) \,\|\, p_\phi(\zeta \mid Z_k))\Big] \\
&\geq \mathbb{E}_{Z^\star \mid Z_k}\left[D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z^\star) \,\|\, P_k(Z_{k+1} \mid Z_k))\right].
\end{aligned} \tag{39}$$

Using the definition of the target marginal transition in Eq. 28, the term on the right can be decomposed exactly as

$$\begin{aligned}&\mathbb{E}_{Z^\star|Z_k}\left[D_{\mathrm{KL}}(q_k(Z_{k+1}\mid Z_k,Z^\star)\,\|\,P_k(Z_{k+1}\mid Z_k))\right]\\&\quad=D_{\mathrm{KL}}(Q_k(Z_{k+1}\mid Z_k)\,\|\,P_k(Z_{k+1}\mid Z_k))\\&\qquad\qquad+\mathbb{E}_{Z^\star|Z_k}\left[D_{\mathrm{KL}}(q_k(Z_{k+1}\mid Z_k,Z^\star)\,\|\,Q_k(Z_{k+1}\mid Z_k))\right].\end{aligned}\tag{40}$$

Combining Eq. 39 and 40 gives Eq. 36, which completes the proof. □

Discussion. Proposition 1 provides a direct interpretation of our training objective. Specifically, Each training future provides one endpoint-conditioned transition $q_k(Z_{k+1}\mid Z_k,Z^\star)$, while averaging these transitions over possible future endpoints gives the target marginal transition $Q_k(Z_{k+1}\mid Z_k)$. The second term on the right of Eq. 36 depends only on these target transitions and does not depend on the learned model. Therefore, minimizing our tractable transition kernel learning loss $\mathcal{L}_{\mathrm{KTM}}$ reduces an upper bound on the divergence between the learned and the intractable target marginal transitions, without explicitly computing the endpoint integral in Eq. 28.

Corollary 1. ***(our learning objective recovers the target marginal transition).*** *If the learned latent prior $p_\phi(\zeta\mid Z_k)$ and conditional transition $p_\theta(Z_{k+1}\mid Z_k,\zeta)$ can match the corresponding training distributions such that the marginal transition error in Eq. 36 becomes zero,*

$$\begin{aligned}&\mathbb{E}_{Z^\star|Z_k}\Big[\mathbb{E}_{\zeta\sim e_\omega(\zeta|Z_k,Z^\star)}\left[D_{\mathrm{KL}}(q_k(Z_{k+1}\mid Z_k,Z^\star)\,\|\,p_\theta(Z_{k+1}\mid Z_k,\zeta))\right]\\&\qquad\qquad+D_{\mathrm{KL}}(e_\omega(\zeta\mid Z_k,Z^\star)\,\|\,p_\phi(\zeta\mid Z_k))\Big]\\&=\mathbb{E}_{Z^\star|Z_k}\left[D_{\mathrm{KL}}(q_k(Z_{k+1}\mid Z_k,Z^\star)\,\|\,Q_k(Z_{k+1}\mid Z_k))\right],\end{aligned}\tag{41}$$

then the learned marginal transition recovers the target marginal transition:

$$\boxed{P_k(Z_{k+1}\mid Z_k)\approx Q_k(Z_{k+1}\mid Z_k).}\tag{42}$$

Discussion. Corollary 1 gives a justification on the learning target of our training objective. This shows that, under the ideal convergence condition, our endpoint-conditioned transition and latent alignment losses are consistent with learning the desired marginal transition without explicitly constructing Q_k. In practice, the learned distributions may not satisfy this condition perfectly due to model capacity or optimization error, so P_k provides an approximation to the target marginal transition Q_k.

B.6 Practical Training and Inference Algorithm

The previous sections formally describe the transition process and learning objective. Here, we summarize our practical training and generation processes with few-step transitions for online motion synthesis, in Algorithm 2 and 3.

During training, each motion sequence from the dataset provides a ground-truth future endpoint $Z^\star$. Rather than generating the full transition process, we randomly select one step k, directly sample its current intermediate state Z_k from the analytical bridge, and construct the corresponding target transition $q_k(Z_{k+1}\mid Z_k,Z^\star)$. The training-only endpoint encoder e_ω uses the observed endpoint to sample a future-informed latent, while the conditional transition model p_θ and latent prior p_ϕ are jointly optimized using the transition kernel learning loss in Eq. 33. Thus, each training update effectively supervises one finite transition without going through the whole process.

At inference, the ground-truth endpoint and endpoint encoder are no longer available. Starting from $Z_0=0$, the model instead samples a latent from the learned latent prior p_ϕ and generates the next state from the corresponding conditional transition p_θ. Repeating this procedure for K finite steps produces the terminal state Z_K, which is then transformed into the future motion through the kinematic operator κ. The generated motion then updates the motion history, and the same procedure is repeated for the next online generation cycle. The same learned networks are shared across all transition steps, while their conditioning inputs are described in the implementation details.

Algorithm 2 Transition Kernel Training

Input: Training motion dataset $\mathcal{D}$
Input: Bridge-time grid $\{\tau_k\}_{k=0}^{K}$
while not converged **do**
  Sample history-future pair $(X, Y^\star) \sim \mathcal{D}$.
  Construct $\bar{A}$ and $\bar{Y} = \kappa(X, \bar{A})$.
  Compute $A^\star$ and $Z^\star = A^\star - \bar{A}$.
  Sample transition step $k \in \{0, \dots, K-1\}$.
  Sample the intermediate transition state $Z_k \sim q_{0\to\tau_k}(Z_k \mid Z_0 = 0, Z^\star)$.
  Construct the target transition $q_k(Z_{k+1} \mid Z_k, Z^\star)$.
  Sample the training latent $\zeta \sim e_\omega(\zeta \mid Z_k, Z^\star)$.
  Predict $p_\theta(Z_{k+1} \mid Z_k, \zeta)$ and $p_\phi(\zeta \mid Z_k)$.
  Compute $\mathcal{L}_{\mathrm{KTM}}$ in Eq. 33.
  Update (θ, ϕ, ω).
end while
return trained model parameters (θ, ϕ, ω).

Algorithm 3 Few-Step Online Inference

Input: Current motion history X
Input: Bridge-time grid $\{\tau_k\}_{k=0}^{K}$
Construct $\bar{A}$ and $\bar{Y} = \kappa(X, \bar{A})$.
Initialize $Z_0 = 0$.
for $k = 0, \dots, K-1$ **do**
  Sample transition latent $\zeta_k \sim p_\phi(\zeta \mid Z_k)$.
  Predict the conditional transition $p_\theta(Z_{k+1} \mid Z_k, \zeta_k)$.
  if $k < K-1$ **then**
    Sample $Z_{k+1} \sim p_\theta(Z_{k+1} \mid Z_k, \zeta_k)$.
  else
    Set $Z_K = \mu_\theta(Z_{K-1}, \zeta_{K-1})$ (terminal state).
  end if
end for
Reconstruct the generated future $\hat{Y} = \kappa(X, \bar{A} + Z_K)$.
Update the motion sequence and history.
return generated future $\hat{Y}$.

C Discussion of our Kinematic Future Representation and Transition Structure

Beyond providing an analytical continuation of the observed motion, our kinematic representation defines the reference and structure of the finite transition process. The continuation provides a meaningful boundary-attached starting future, while acceleration corrections describe how its upcoming dynamics change. Through the same kinematic operator κ, this representation also determines how stochastic perturbations propagate over future frames and how deviations from a predicted transition are measured during control. We discuss these properties (for Euclidean coordinates) in the following.

C.1 Boundary-Attached Transition States

Given the observed motion history X, we construct the reference acceleration $\bar{A}$ from its boundary dynamics and obtain $\bar{Y} = \kappa(X, \bar{A})$. Any future acceleration A is represented by its correction from this reference,

$$Z = A - \bar{A}, \qquad Y(Z) = \kappa(X, \bar{A} + Z). \tag{43}$$

Thus,

$$Z = 0 \Longrightarrow Y(Z) = \bar{Y}, \qquad Z^\star = A^\star - \bar{A} \Longrightarrow Y(Z^\star) = Y^\star. \tag{44}$$

Our finite transition process begins from $Z_0 = 0$, which corresponds to continuing the observed motion dynamics over the upcoming future window. As the process evolves through $Z_0 \to Z_1 \to \cdots \to Z_K$, each Z_k represents how the acceleration dynamics of these same future frames have changed from the reference continuation.

Moreover, all transition states are reconstructed through the same kinematic operator $\kappa(X, \cdot)$ using the same observed boundary pose and velocity in X, so the transition process does not move between independently defined future sequences. Conversely, it evolves different acceleration dynamics over a consistent future interval while remaining attached to the motion that has already been observed. This gives both the source state and every intermediate transition state a meaningful interpretation in the kinematic motion space.

The reference continuation and the acceleration-correction coordinates play complementary roles. The continuation $\bar{Y}$ provides the meaningful starting future, while Z describes how its subsequent dynamics should change. In the next subsection, we show that these coordinates also induce a specific temporal structure on stochastic perturbations and on deviations from a predicted transition.

C.2 Temporal Structure of Stochastic Transitions and Control

Consider two transition states Z and $\hat{Z}$, with $\Delta Z = Z - \hat{Z}$, and let Δy_f denote the difference between their reconstructed poses at future frame f. Since both futures share the same observed boundary, the kinematic construction in Eq. 1 of the main paper gives

$$\Delta y_f = (\Delta t)^2 \sum_{j=1}^{f} (f - j + 1)\, \Delta z_j. \tag{45}$$

Thus, an acceleration change at frame j affects that frame and all subsequent poses, with earlier changes accumulating over a longer portion of the future.

Transition structure. Consider a conditional transition $Z_{k+1} = \mu_k + \sigma_k \epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$. Although the perturbation is isotropic in acceleration-correction space, its reconstructed motion perturbation satisfies

$$\Delta y_f = \sigma_k (\Delta t)^2 \sum_{j=1}^{f} (f - j + 1)\epsilon_j.$$

The induced covariance between future frames f and r is therefore

$$\mathrm{Cov}\,(\Delta y_f, \Delta y_r) = \sigma_k^2 (\Delta t)^4 \sum_{j=1}^{\min(f,r)} (f - j + 1)(r - j + 1)\, I. \tag{46}$$

Hence, isotropic transition noise in acceleration space becomes temporally correlated in motion space because future frames share the effects of earlier acceleration perturbations. By contrast, independent isotropic perturbations applied directly to future poses would not induce this temporal coupling.

Control structure. The same representation also provide a meaningful temporal structure our inference-time control. Recall that the controlled endpoint objective in Eq. 7 balances the control objective with the negative log-density of the local endpoint prior $\widetilde{p}_k(Z_K \mid Z_k, \zeta)$. Since this prior is Gaussian with mean $\hat{Z}_K$ and variance Σ_k/β_k^2, its contribution is proportional to $\|Z_K - \hat{Z}_K\|_2^2$. Let $\Delta Z_K = Z_K - \hat{Z}_K$ and let Δy_f denote the corresponding difference between the candidate and predicted future motions. Because both futures share the same observed boundary, their acceleration difference satisfies $\Delta z_f = (\Delta y_f - 2\Delta y_{f-1} + \Delta y_{f-2})/(\Delta t)^2$. Therefore,

$$\|\Delta Z_K\|_2^2 = \frac{1}{(\Delta t)^4} \sum_{f=1}^{F} \|\Delta y_f - 2\Delta y_{f-1} + \Delta y_{f-2}\|_2^2, \tag{47}$$

where $\Delta y_0 = \Delta y_{-1} = 0$ follows from the shared boundary state.

Thus, the transition-prior term in our control objective does not penalize pose deviations independently. In motion space, it induces a second-order temporal penalty on deviations from the predicted future, favoring controlled adjustments that remain close to its predicted acceleration dynamics.

Together, these results clarify the structural role of the kinematic future representation. The continuation provides a boundary-attached reference state, while acceleration-correction coordinates induce a shared temporal geometry for stochastic transitions and control. The formulation does not increase the set of representable future motions; rather, it changes how future states are referenced, perturbed, and regularized throughout the transition process.

D Discussion of our Training-Free Transition Control

Our control formulation uses each predicted finite transition as a local motion prior while optimizing its future endpoint under the inference-time objective. Here, we discuss the interpretation of this optimization and its relation to existing endpoint-guidance methods.

D.1 MAP Interpretation of Transition Control

Recall that a candidate endpoint Z_K defines both a complete future motion through $\kappa(X, \bar{A} + Z_K)$ and an analytical next-state transition $q_k(Z_{k+1} \mid Z_k, Z_K)$. Meanwhile, the learned transition model predicts $p_\theta(Z_{k+1} \mid Z_k, \zeta)$, which is kept fixed during endpoint optimization.

When both transitions use the same transition variance, the negative log-prior in our endpoint MAP (maximum-a-posteriori) optimizaiton objective is equivalent to the divergence between these two transitions (up to an independent constant C):

$$-\log \widetilde{p}_k(Z_K \mid Z_k, \zeta) = D_{\mathrm{KL}}(q_k(Z_{k+1} \mid Z_k, Z_K) \,\|\, p_\theta(Z_{k+1} \mid Z_k, \zeta)) + C. \tag{48}$$

Therefore, our endpoint optimization can be interpreted as balancing objective satisfaction with deviation from the transition prior predicted by the motion model. The endpoint is thus controlled through the finite transition that it induces, rather than optimized only as an isolated future motion.

This also connects training and control. During training, ground-truth endpoints define analytical transitions that supervise the learned kernel to transition toward plausible future motions. During control, the learned kernel is instead kept fixed and regularizes the analytical transition induced by the controlled endpoint, so the controlled motion can stay plausible. Together with Sec. C.2, this shows that control is regularized directly in the acceleration-correction space used by the learned transition model, with coherent kinematic propagation across the future sequence.

D.2 Relation to Existing Endpoint Guidance

Optimizing a predicted clean or terminal motion sample and projecting it back to the generative process is also used by related guidance methods. For example, ProjFlow (Watanabe et al., 2026) corrects predicted clean motion under kinematic spatial constraints.

The main distinction is that our generative process is itself constructed as a controllable kinematic transition process. The future motion dynamics evolves through a small number of stochastic states in the boundary-attached acceleration-correction space, where every state describes how the same upcoming future changes from the observed motion and is directly constructed through the kinematic operator κ. Because the future endpoint also determines the analytical next-state transition, control can act directly on the generative transition itself. The learned transition provides the local prior for this endpoint update, while the kinematic representation determines how the resulting change propagates over the future motion. As shown in Sec. C.2, deviations from the predicted transition correspond to temporally coupled changes in future acceleration dynamics rather than independent pose corrections.

This coupling between kinematic representation and transition control process is also what makes our formulation suitable for training-free and real-time online synthesis. Each finite step both advances the stochastic generation process and exposes a structured future endpoint that can be optimized under the online control objective. Since only a small number of such transitions are required, and the predicted transition remains fixed during endpoint optimization, new objectives can steer the same generative process without retraining or repeated computations of the learned motion model. The controlled state then becomes the input to the next transition, allowing generation and control to proceed together throughout the online sequence rather than as separate generation and refinement stages.

E Model Architecture and Implementation Details

This section provides the architecture and implementation details of our transition model and online controller. We first describe the learned components used by the finite transition process, then discuss the inference budget for online control, followed by the practical training and control settings used in our implementation.

E.1 Model Architecture

Our motion transition model contains three learned components: the training-only endpoint encoder e_ω, the latent prior p_ϕ, and the transition network p_θ. The same networks are shared across all finite transition steps, with the bridge time τ_k provided as an additional conditioning input.

Specifically, the endpoint encoder takes $(X, Z^\star, \tau_k)$ as input and outputs the mean and log-variance of a 64-dimensional Gaussian latent distribution $e_\omega(\zeta \mid X, Z_k, Z^\star, \tau_k,)$. The latent prior takes (X, Z_k, τ_k) as input and outputs the mean and log-variance of the corresponding inference-time latent distribution $p_\phi(\zeta \mid X, Z_k, \tau_k)$. Both networks are implemented using 6-layer Transformers (Vaswani et al., 2017) with hidden dimension 384 and 8 attention heads. During training, ζ is sampled from the endpoint encoder e_ω, while at inference it is sampled from the latent prior p_ϕ. Subsequently, the transition network p_θ takes (X, Z_k, τ_k, ζ) as input and directly predicts the next-state Gaussian transition kernel with mean μ_θ, together with the analytical bridge variance defined in Sec. B.2. We implement this network using an 8-layer Transformer (Vaswani et al., 2017) with hidden dimension 384 and 8 attention heads.

Each motion frame is linearly projected into a 384-dimensional Transformer token. The bridge time τ_k is mapped to a continuous-time sinusoidal embedding, while the transition latent ζ are projected to the same hidden dimension and provided as conditioning tokens. The endpoint encoder is used only during training and is not evaluated during normal inference. Generation requires only the latent prior p_ϕ and transition network p_θ.

E.2 Online Execution and Inference Budget

At each online generation cycle, we first construct the kinematic future continuation from the current motion history and initialize the transition process with $Z_0 = 0$. For each of the K finite transition steps, the latent prior is computed once to sample ζ, followed by one forward pass of the transition network to predict the conditional transition kernel. This transition is then kept fixed while its endpoint is optimized under the control objective. The resulting controlled endpoint defines the analytical next-state transition, whose sampled state becomes the input to the subsequent step. After the final transition, the controlled future is reconstructed through the kinematic operator and used to update the motion history for the next online cycle.

Let M denote the control-optimization budget (e.g., maximum number of control objective and gradient evaluations) at each finite transition step. If C_ϕ and C_θ denote the costs of computing the latent prior and transition network, and C_{ctrl} denotes the cost of one endpoint-control evaluation, the cost of one controlled transition step is approximately

$$C_{\text{step}} = C_\phi + C_\theta + MC_{\text{ctrl}}, \tag{49}$$

and one complete online control cycle has complexity

$$C_{\text{cycle}} = \mathcal{O}(K\,(C_\phi + C_\theta + MC_{\text{ctrl}}))\,. \tag{50}$$

Importantly, the sampled latent and predicted transition prior remain fixed during the M control evaluations. Thus, increasing the control-optimization budget does not introduce additional evaluations of, or backpropagation through, the learned motion model. The additional computation is confined to the inference-time objective, kinematic reconstruction, and analytical transition-prior term.

This separates learned-model computation from objective-dependent control computation. For a fixed small number of finite transitions K, the learned model is evaluated only K times per online cycle, while the control budget M can be adjusted according to the available inference budget depending on the control objective.

E.3 Implementation Details

Our experiments are conducted using a single NVIDIA RTX 4090 GPU. We construct the reference future from the observed boundary velocity and acceleration as described in the main paper, using $\eta = 1$ for the acceleration drift rate. Each online control cycle generates a future window of $F = 8$ frames at 30 FPS.For translational coordinates, the kinematic operator propagates position using linear velocity and acceleration. For joint rotations, angular differences are computed through the logarithm map and rotations are reconstructed through exponential-map composition on $\mathrm{SO}(3)$. The transition state is the acceleration correction $Z = A - \bar{A}$, and all generated endpoints are reconstructed through the same kinematic operator κ. We use the VE reference bridge schedule following Zhou et al. (2024a), with $K = 5$ uniformly spaced finite transition steps over bridge-interval $\tau \in [0, 1]$. We train the model using AdamW (Loshchilov & Hutter, 2019) optimizer with learning

rate 10^{-4}, batch size 64, and uniformly sampled transition steps. For training-free control, endpoint optimization is initialized from the endpoint $\hat{Z}_K$ induced by the predicted transition, while the sampled latent and predicted transition remain fixed throughout optimization. We use L-BFGS (Liu & Nocedal, 1989) with learning rate 1.0 and control strength $\lambda = 8$, limiting each controlled transition to 8 objective evaluations. At the terminal transition, where the analytical bridge variance vanishes, we use the ℓ_2-regularizer on the endpoint distance. No gradients are propagated through the latent prior or transition network during endpoint optimization.

F Experimental Details

F.1 Online Protocol and Metrics

Online protocol. All methods are evaluated in continuous motion rollouts using the same initial motions and task configurations. Motion is generated at 30 Hz in $F = 8$-frame primitives. At each control cycle, the method generates a future motion primitive, commits the generated frames, and then updates its motion history and the currently active control target for the next cycle. History frames remain fixed once they are generated and committed. At each cycle, only the currently active target is provided to the control inference, while the next target is revealed only after the current target has been successfully satisfied. The same scenario, target sequence, and random seed are shared across methods whenever their control interface supports the corresponding task. The total limit for each full rollout sequence is 60 seconds, since some methods cannot complete all the objective targets.

Control metrics. We define success rate (SR) generically as the percentage of specified control targets that are successfully satisfied:

$$\mathrm{SR} = 100 \times \frac{\#\text{ satisfied targets}}{\#\text{ specified targets}}.$$

The task-specific satisfaction criterion is defined in the corresponding task protocol in Sec. F.2. For navigation, targets are sequential waypoints; for end-effector control, targets are active 3D wrist locations. Targets not satisfied before they expire or the rollout terminates are counted as unsuccessful.

For dynamic obstacle avoidance, collision rate (Coll.) is the percentage of obstacles that are collided with at least once during motion execution:

$$\mathrm{Coll.} = 100 \times \frac{\#\text{ collided obstacles}}{\#\text{ total obstacles}}.$$

For end-effector control, we additionally report distance-to-goal (DTG) following WANDR (Diomataris et al., 2024), computed as the closest wrist–target distance while each target is active and averaged across targets.

Motion quality and semantic consistency. We evaluate pose plausibility using averaged neural Riemannian distance (ARD) (He et al., 2024) and foot skating following PRIMAL (Zhang et al., 2025). peak jerk (PJ) following DART (Zhao et al., 2025) to evaluate temporal motion smoothness.

Inference efficiency. Generation speed (FPS) is computed as

$$\mathrm{FPS} = \frac{\#\text{ committed motion frames}}{\text{total online inference time}}.$$

Latency is the mean wall-clock time of one online control cycle, reported in milliseconds. Runtime includes controller preparation, motion generation and control, and the update required before the next cycle, while model initialization, rendering, and offline metric computation are excluded. Unless otherwise stated, all metrics are averaged across evaluation scenarios.

F.2 Online Task Setting

Waypoint navigation with dynamic obstacles. Each rollout contains a sequence of waypoints that are revealed one at a time; the next waypoint becomes active only after the current one is

reached within 0.2m. The benchmark contains 1,000 scenarios spanning obstacle-free, static, moving, mixed, and challenging layouts under locomotion action such as walk, run, or jog. Obstacles use varying shapes with randomly varying sizes, placements, and trajectories. Moving-obstacle motion is deterministic and trajectory is provided to the controller over the prediction horizon. We do not evaluate obstacle motion prediction. When the direct route is obstructed, a lightweight geometric planner provides an intermediate detour target. The general navigation objective is described as follows:

$$G_{\text{nav}} = G_{\text{waypoint}} + G_{\text{heading}} + G_{\text{collision}},$$

where G_{waypoint} encourages progress toward the current waypoint or detour target, G_{heading} encourages the character to face the corresponding travel direction, and $G_{\text{collision}}$ penalizes insufficient clearance between the predicted body and nearby obstacles. Collision is checked using all 22 SMPL-X (Pavlakos et al., 2019) body joints, with collision spheres centered at each joint by a fixed collision radius to approximates the local human body volume, rather than representing the character by the root alone. Moving obstacles are evaluated at their corresponding time interval.

End-effector control. Following the 3D reaching setting of WANDR (Diomataris et al., 2024), we control the wrist toward a target position in 3D space. We adapt the protocol to continuous online generation by sequentially changing the active wrist target during the rollout, requiring the character to redirect from its current generated state without restarting the motion. The same target sequence and change schedule are used for all compared methods. Following WANDR, a target is considered successfully reached when the wrist comes within 10 cm of the target while it is active. No reaching-specific supervision or target-conditioning signal is used when training KTM.

F.3 Baseline and Ablation Settings

Baselines. For navigation, the *Uncontrolled* baseline uses the only base KTM model without applying inference-time control. PRIMAL-G (Zhang et al., 2025) applies training-free diffusion guidance using 10 or 50 denoising steps following DDPM schedule Ho et al. (2020). For DART-RL (Zhao et al., 2025), we retrain its reinforcement-learning-based controller on our navigation task objective, providing a control-trained real-time baseline. For dynamic end-effector control, we compare with WANDR (Diomataris et al., 2024) and GOAL (Taheri et al., 2022) using their goal-conditioned reaching models. All methods use the same initial motions and task sequences/scenarios whenever supported by their control interface.

Ablation models. The *w/o kinematic transition representation* variant replaces the boundary-attached acceleration-correction process with a Gaussian-initialized pose-space transition model of the same capacity and is retrained from scratch. *w/o transition prior* removes the learned local-prior term during endpoint optimization. *w/o progressive transition control* retains all K generative transitions but applies endpoint control only at the final transition using the same total optimization budget. For the control–efficiency analysis, we vary the per-transition control budget M while keeping all other settings fixed. All ablations are evaluated under the same online navigation protocol.

G LLM Usage Statement

We used Large Language Models (LLMs) as assistive tools for improving manuscript writing, debugging auxiliary code, and helping design the synthetic benchmarks used in our experimental validation. All AI-assisted content and code were carefully reviewed and verified by the authors. The research idea, problem formulation, proposed method, experimental analysis, and core algorithmic implementation are entirely the original work of the authors. The authors take full responsibility and ownership for all conceptual and experimental aspects of this work.