PLAN, GENERATE, REPLAN: HIERARCHICAL SYMBOLIC MUSIC GENERATION WITH BOUNDARY-AWARE HISTORY CORRECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Music creation is an iterative process: composers repeatedly revise what has been written and decide how the music should develop next. Supporting this process requires modeling how music evolves under user instructions. We introduce the Music World Model (MWM), a hierarchical latent world model for controllable long-form symbolic music generation. MWM represents musical content as latent states and requested changes as explicit actions. A phrase-level planner predicts structural context, while a segment-level executor recursively generates musical states under this context and the requested action. Users can update actions at phrase boundaries, and the planner refreshes its predictions from generated history to account for how the music has evolved. However, mismatches between new plans and preceding context can enter this history and influence subsequent plans, causing structural drift. We therefore introduce Boundary-Aware History Correction (BHC), an inference-time mechanism that combines context anchoring with persistent correction memory to reconcile planned slow trajectories with preceding context. BHC carries correction information across phrase boundaries while keeping learned weights and external actions fixed, without future reference music. To support learning these action-conditioned dynamics, we construct *Slakh-Action* from Slakh2100 Redux, aligning multi-track symbolic music with segment-level states, phrase-level summaries, and structural action annotations. **An anonymous interactive demo is available at `demo-czv.pages.dev`.**

1 INTRODUCTION

> "Use the dynamics of the system to take you in the direction you wanted to go."
>
> —Brian Eno, "Composers as Gardeners," 2011

A creator does not need to specify every musical event in advance. Instead, the creator defines materials, constraints, and interventions, allowing the system to develop them toward a desired musical direction.

Symbolic music models support conditional infilling Malandro (2024), structural conditioning Wang et al. (2024), and flexible track-level completion Lv et al. (2025). Figure 1(a) illustrates the diffusion formulation: musical conditions guide repeated denoising of a corrupted representation until a musical sample is obtained Wang et al. (2024); Huang et al. (2024). In Figure 1(b), a conditional Transformer instead extends a sequence by predicting each token from preceding tokens and supplied conditions Qu et al. (2025); Bhandari et al. (2025). Both formulations support controllable generation, but iterative creation also requires tracking how a requested change affects the music as it unfolds. For example, from the same passage, a creator may request a denser or sparser next phrase and revise that request after hearing the result. The model must connect each intervention to the current musical context and carry its consequences into subsequent generation.

Figure 1(c) makes this relationship explicit: a world model separates the current state from the requested action and predicts how the state evolves under that action Hansen et al. (2024); Hafner et al. (2025). We adopt this state-transition perspective in the *Music World Model* (MWM), a hierarchical latent world model for long-form multi-track symbolic music. A phrase-level planner

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.

predicts structural context, or a *plan*, while a segment-level executor recursively predicts musical states under the plan and a temporally extended structural action. After each phrase, the planner refreshes its predictions from generated history, and the creator can update the action for the next phrase. This *plan–generate–replan* loop allows the same observed history to develop into different action-conditioned continuations, with later plans responding to the music actually generated.

To learn these dynamics, we introduce *Slakh-Action*, an extension of Slakh2100 Redux that annotates structural changes between consecutive phrases. The resulting multi-track state–action trajectories provide supervision and evaluation data for action-conditioned musical dynamics.

Long-horizon world-model rollouts are prone to compounding errors, as recursively feeding model-generated predictions back as context can induce a train–inference distribution mismatch and cause local prediction errors to accumulate over time Huang et al. (2025); Zhang et al. (2025); Liu et al. (2026); Lin et al. (2026a). In music generation, this issue takes on a distinct form because structural predictions directly shape the music being generated. An inaccurate plan can lead the current phrase away from its intended musical structure. The generated music is then reused as historical context for subsequent replanning, thereby enabling the structural error to affect future predictions. As this process repeats, local planning errors can accumulate and result in progressive deviations from the intended musical structure. We refer to this music-specific failure mode as *predicted-plan structural drift*. To mitigate this drift, we introduce *Boundary-Aware History Correction* (BHC), an inference-only mechanism that corrects reused structural context at replanning boundaries while leaving the learned model and external actions unchanged.

On *Slakh-Action*, MWM's hierarchical planner–executor achieves the lowest mean RBDE against six baselines on 102 held-out songs, improving over NotaGen by 12.1%. Counterfactual density tests yield first-phrase directional and ordering accuracies of 83.5% and 86.8%, versus 18.5% and 21.9% with actions masked. Matched ablations show that replanning from generated history lowers mean density and role-structure errors compared with a fixed plan in the evaluated long-horizon window. For 64-bar continuation, BHC closes 29.1% of the terminal decoded-F1 gap to a diagnostic rollout using target-derived reference plans, reaching 42.6% under the strongest tested persistent plan perturbation without retraining.

In summary, our contributions are as follows:

- We introduce MWM, a hierarchical action-conditioned world model for long-form symbolic music generation that couples segment-level musical dynamics with phrase-level structural planning.
- We introduce BHC, an inference-only correction with no additional learned weights that combines context anchoring with a persistent boundary correction to mitigate structural drift while keeping the learned model and external actions fixed.
- We build *Slakh-Action* by augmenting Slakh2100 with phrase-level structural actions. Objective and subjective evaluations confirm MWM's generation quality, while counterfactual and long-horizon studies validate action responsiveness and BHC's structural stability.

2 Related Work

As illustrated in Figure 1(a), Whole-Song uses cascaded diffusion over form, phrases, and local content Wang et al. (2024), while Diff-Symbo combines text-controlled latent diffusion with autoregressive continuation Lin et al. (2026b). Figure 1(b) illustrates autoregressive generation: MuPT and NotaGen use ABC-based Transformers for multi-track pretraining and classical-music generation Qu et al. (2025); Wang et al. (2025); BEAT organizes events into uniform temporal steps Qian et al. (2026); Text2MIDI conditions an autoregressive decoder on text Bhandari et al. (2025); Music2Music uses track-aware reconstruction for instrumentation transformations Ou et al. (2025); and Amadeus combines autoregressive note prediction with bidirectional diffusion over note attributes Su et al. (2026). The latter six are our external baselines. Anchored Cyclic Generation (ACG) reuses generated musical features to guide continuation and mitigate error accumulation Cao et al. (2026). BHC also exploits history, but corrects plan-derived structural context at replanning boundaries and carries correction information across phrases with frozen parameters.

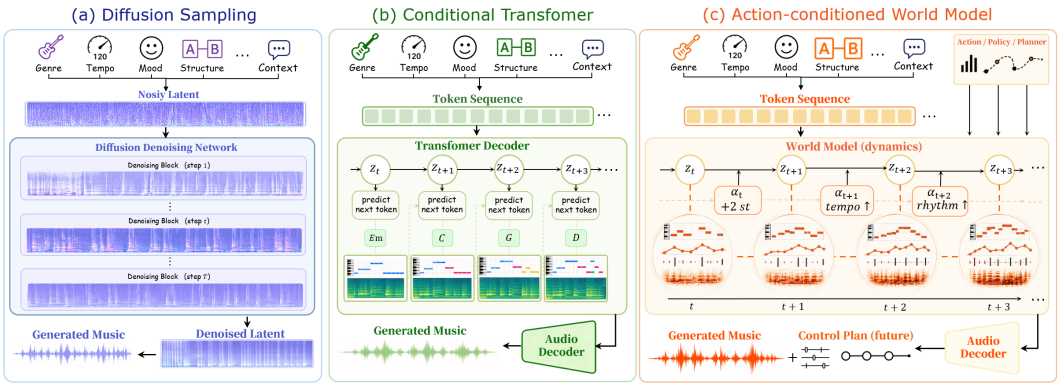


Figure 1: Controllable music generation paradigms. (a) Diffusion models iteratively denoise conditioned latents. (b) Conditional Transformers autoregressively predict tokens from preceding sequences and conditions. (c) Action-conditioned world models predict state transitions under interventions, supporting different trajectories from the same state and recursive long-horizon planning.

Action-conditioned world models (Figure 1(c)) separate states, actions, and transitions. TD-MPC2 uses latent dynamics for trajectory optimization, while DreamerV3 learns from imagined rollouts Hansen et al. (2024); Hafner et al. (2025). Recent predictive music models learn transitions or structured representations from audio and symbolic music Wang et al. (2026); Hawley (2026). MWM couples phrase-level planning with segment-level execution to recursively generate multi-track states under structural actions and replan from generated history. Paired predicted-plan and teacher-plan rollouts diagnose structural drift and its downstream effects.

3 METHOD

For coherent long-horizon generation under *user instructions*, we use three components: *latent musical states* and *actions* for representing musical content and user instructions (Sec. 3.1); a *hierarchical music world model* (MWM) for phrase-level planning and recursive segment-level execution (Sec. 3.2); and *Boundary-Aware History Correction* (BHC) for reducing phrase-boundary *structural drift* from *closed-loop replanning* without future reference music (Sec. 3.3).

3.1 MUSICAL STATES AND ACTIONS

To interface with user instructions, MWM separates *what the music is* from *how it should change*, representing them as a latent musical state and an action, respectively. This allows the same musical history to evolve differently under different instructions.

We partition each piece into two-bar multi-track segments $\mathbf{x}_t$ containing note onsets, sustain, and velocity, and group $M = 4$ segments into an eight-bar *phrase*, with $t = (\tau - 1)M + m$ for member m of phrase τ. Segment-level states preserve local detail, while phrase-level summaries support structural planning. A neural codec encodes and reconstructs each segment as

$$\mathbf{z}_t = E_\phi(\mathbf{x}_t), \qquad \hat{\mathbf{x}}_t = D_\psi(\mathbf{z}_t), \tag{1}$$

where E_ϕ and D_ψ are the encoder and decoder. The state $\mathbf{z}_t = [\mathbf{z}_t^g; \mathbf{z}_t^\ell]$ combines global features pooled for phrase planning with local detail. During generation, MWM recursively predicts latent states that D_ψ decodes into symbolic music.

To represent the musical attributes controlled by user instructions, we describe segment m of phrase τ by an eight-dimensional control descriptor $\mathbf{u}_{\tau,m}$. Its entries are the segment tempo, note density, and six role-activity indicators for melody, bass, chords, drums, accompaniment, and other. Averaging these descriptors over the M segments gives the phrase-level control descriptor

$$\mathbf{c}_\tau = \frac{1}{M} \sum_{m=1}^{M} \mathbf{u}_{\tau,m}. \tag{2}$$

This phrase-level descriptor summarizes the absolute controls of phrase τ. To represent a requested change between phrases, let $\mathcal{A}$ denote the controllable coordinates and define the *action* as the projected phrase-level difference:

$$\mathbf{a}_t = \Delta \mathbf{c}_\tau = P_{\mathcal{A}} \left(\mathbf{c}_{\tau+1} - \mathbf{c}_\tau \right), \qquad t \in \mathcal{J}_{\tau+1}, \tag{3}$$

where $\mathcal{J}_{\tau+1} = \{\tau M + 1, \ldots, (\tau+1)M\}$ and $P_{\mathcal{A}}$ retains the controllable coordinates of this phrase-level difference and sets all other coordinates to zero. The same action is applied to every segment in $\mathcal{J}_{\tau+1}$. For the next phrase, $\mathbf{u}_{\tau+1,m}$ specifies the tempo, density, and role activity for segment m. For example, requesting a sparser phrase gives a negative density action while leaving instrumentation coordinates unchanged; a zero action preserves the commanded phrase-level attributes while allowing the musical state to evolve.

Training actions are differences between adjacent observed phrase descriptors. Our primary cross-model evaluation uses all-zero actions and planner-predicted segment-level descriptors from the observed prefix to generate the next phrase. Separate command-conditioned evaluations supply actions and $M = 4$ descriptors $\mathbf{u}_{\tau+1,1:M}$ per phrase through external schedules. The interactive LLM interface maps natural-language instructions to actions for user-directed generation. In both evaluation settings, latent states and notes remain model-generated without future reference music. Action normalization uses fixed statistics; details are given in Appendix A.2.

3.2 Hierarchical Music World Model

Beyond local state–action transitions, long-form generation requires coordination across segments and phrases. MWM therefore combines phrase-level planning of longer-range structure with recursive segment-level execution under the requested action.

Phrase-level planning. To provide the segment-level executor with a phrase-level target, we represent each phrase τ with three inputs. The phrase representation $\mathbf{p}_\tau = G(\mathbf{z}_{(\tau-1)M+1:\tau M})$ averages its global latent features. The nine-dimensional summary $\mathbf{c}_\tau^{\text{plan}}$ contains mean tempo, mean density, within-phrase density variation, and six mean role-activity values, while $\mathbf{d}_\tau = [\mathbf{u}_{\tau,1}; \ldots; \mathbf{u}_{\tau,M}]$ concatenates the segment-level descriptors in phrase order. Given the latest L_p phrases, the planner F_θ predicts the structured plan for the next phrase:

$$\hat{\mathcal{S}}_{\tau+1} = F_\theta \left(\mathbf{p}_{\tau-L_p+1:\tau}, \mathbf{c}^{\text{plan}}_{\tau-L_p+1:\tau}, \mathbf{d}_{\tau-L_p+1:\tau} \right) = \{\hat{\mathbf{g}}_{\tau+1}, \hat{\mathbf{m}}_{\tau+1,1:N}, \hat{\mathbf{h}}_{\tau+1,1:K}, \hat{\mathbf{s}}_{\tau+1}\}. \tag{4}$$

Here $\hat{\mathcal{S}}_{\tau+1}$ contains four coordinated predictions. The global output $\hat{\mathbf{g}}_{\tau+1}$ summarizes the next phrase and its controls; $\hat{\mathbf{m}}_{\tau+1,1:N}$ provides a latent-and-control prediction for each segment position; $\hat{\mathbf{h}}_{\tau+1,1:K}$ provides pooled latent-and-control summaries over the longer horizon; and $\hat{\mathbf{s}}_{\tau+1}$ is the pooled latent used for slow conditioning. We set $N = M = 4$ to align the segment slots with the four segment positions, and $K = 4$ for four future phrase targets.

Plan-to-trajectory conversion and segment-level executor. The planner produces a phrase-level plan, while the executor operates segment by segment. We therefore map $\hat{\mathcal{S}}_{\tau+1}$ to M slow conditions through *plan-to-trajectory conversion*. For each position $m = 1, \ldots, M$, the trajectory builder projects segment-slot and horizon predictions into the slow-state space and combines them with the pooled slow prediction:

$$\mathbf{v}^{\text{raw}}_{\tau+1,m} = \hat{\mathbf{s}}_{\tau+1} + \frac{\lambda_{\text{mem}}}{M} \sum_{i=1}^{M} P_{\text{mem}}(\hat{\mathbf{m}}_{\tau+1,i}) + \lambda_{\text{hor}} P_{\text{hor}} \left(\hat{\mathbf{h}}_{\tau+1,m} \right), \qquad m = 1, \ldots, M. \tag{5}$$

Here P_{mem} and P_{hor} map the segment-slot and horizon outputs to the slow-state dimension, while λ_{mem} and λ_{hor} control their contributions. The resulting pre-adapter trajectory $\mathbf{V}_{\tau+1} = \mathbf{v}^{\text{raw}}_{\tau+1,1:M}$ contains one slow conditioning vector for each segment of the next phrase.

Because the new trajectory is predicted from refreshed history, a shared row-wise adapter reconciles each planned condition with the slow context from the preceding phrase. For segment position m, let $\eta_m = (m-1)/(M-1)$ denote its normalized position within the phrase. The adapter combines the planned condition $\mathbf{v}^{\text{raw}}_{\tau+1,m}$, the last stored pre-adapter slow state $\mathbf{v}^{\text{ctx}}_\tau$, the pooled slow prediction $\hat{\mathbf{s}}_{\tau+1}$, and η_m through a learned residual:

$$\mathbf{r}_{\tau+1,m} = \mathbf{v}^{\text{raw}}_{\tau+1,m} + r_\xi \left(\mathbf{v}^{\text{raw}}_{\tau+1,m}, \mathbf{v}^{\text{ctx}}_\tau, \hat{\mathbf{s}}_{\tau+1}, \eta_m \right). \tag{6}$$

Applying the shared adapter row-wise converts $\mathbf{V}_{\tau+1}$ into the adapted trajectory $\mathbf{R}_{\tau+1} = [\mathbf{r}_{\tau+1,1}; \ldots; \mathbf{r}_{\tau+1,M}]$, whose row for segment $t+1$ is inserted into the executor's slow-condition history when that segment is generated. The adapter reduces discontinuity between the newly planned trajectory and the preceding slow context while preserving the planned change. We retain the pre-adapter trajectory $\mathbf{V}_{\tau+1}$ across the phrase boundary, while $\mathbf{R}_{\tau+1}$ supplies the slow conditions used by the executor. BHC therefore modifies $\mathbf{V}_{\tau+1}$ before adaptation.

The **segment-level executor** predicts the next latent musical state from latent, descriptor, and slow-condition histories. Here $\mathbf{z}_t$ and $\mathbf{z}_{t+1}$ denote current and next states; $\bar{\mathbf{z}}_t$ is the generated-history state, and $\hat{\mathbf{z}}_{t+1}$ is the predicted next state. Given supplied segment-level descriptor $\mathbf{u}^{\text{in}}_{t+1}$, latent-history length L_z, and slow-history length L_R, the executor predicts

$$\hat{\mathbf{z}}_{t+1} = \bar{\mathbf{z}}_t + f_\omega\big(\bar{\mathbf{z}}_{t-L_z+1:t}, \bar{\mathbf{u}}_{t-L_z+1:t}, \bar{\mathbf{R}}_{t-L_R+1:t}; \mathcal{N}(\mathbf{a}_{t+1}), \mathbf{u}^{\text{in}}_{t+1}\big)\,. \tag{7}$$

Here f_ω is the executor's learned transition function that predicts the latent-state residual. The executor receives the action and supplied descriptors through projections, together with the global, segment-slot, and horizon plan branches. Repeating this transition for M segments under the phrase's fixed action produces one phrase, which is decoded through D_ψ. Because the action and segment-level descriptors are explicit inputs, MWM can change the requested action or supplied controls at phrase boundaries while keeping the planner and executor fixed.

Replanning from generated history. MWM refreshes the planner after each phrase using generated latents and controls, adapting subsequent plans to its rollout. Let τ_0 be the final observed phrase. After generating phrase $\tau+1$, path q forms

$$\hat{\mathbf{p}}^q_{\tau+1} = G\left(\hat{\mathbf{z}}^q_{\tau M+1:(\tau+1)M}\right). \tag{8}$$

The planner history then combines observed and generated states:

$$\bar{\mathbf{p}}^q_\ell = \begin{cases} \mathbf{p}_\ell, & \ell \le \tau_0, \\ \hat{\mathbf{p}}^q_\ell, & \ell > \tau_0, \end{cases} \qquad \hat{\mathcal{S}}^q_{\tau+2} = F_\theta\left(\bar{\mathbf{p}}^q, \bar{\mathbf{c}}^{\text{plan},q}, \bar{\mathbf{d}}^q\right). \tag{9}$$

At each refresh, the latest L_p entries are used. The refreshed plan replaces the previous one, while fast-state and slow-context histories persist. This closes the planning–execution loop, but also allows structural prediction errors to propagate through the history used by later plans, motivating BHC.

We pretrain and freeze the codec, then jointly fit the planner, projections, adapter, and executor. All learned components remain frozen during evaluation; BHC is inference-only and adds no learned weights. Full objectives, budgets, and checkpoint details are given in Appendix B.2.

3.3 Boundary-Aware History Correction

Replanning from generated history lets MWM respond to the music it has produced, but also allows structural errors to persist. At each phrase boundary, a new plan replaces the previous one while the executor's latent and slow-context histories continue. A predicted trajectory can therefore be plausible in isolation yet inconsistent with the preceding phrase. Executing it changes the generated history on which later plans depend, allowing the mismatch to propagate across phrases. We call the resulting deviation *predicted-plan structural drift*.

To limit this drift, we seek to connect each refreshed plan to the inherited slow context while retaining information about earlier corrections. Since future reference music is unavailable during generation, we use the stored slow context as the boundary reference. We therefore introduce Boundary-Aware History Correction (BHC), an inference-only mechanism that combines *context anchoring* to moderate the current plan's departure from this reference with *correction memory* to carry previous adjustments into later phrases. BHC acts on the pre-adapter trajectory so that correction affects both the conditions supplied to the executor and the slow context retained at the next boundary.

Context anchoring. To reconcile continuity with the new plan's proposed evolution, we first anchor its slow trajectory to the preceding phrase. At the boundary before phrase $\tau+1$, let $\mathbf{V}^{\text{raw,BHC}}_{\tau+1} = \mathbf{v}^{\text{raw}}_{\tau+1,1:M}$ be the pre-adapter trajectory produced from the BHC path's predicted plan by Equation 5. Let $\mathbf{v}^{\text{ctx}}_{\tau+1,0}$ be the final pre-adapter slow state from the preceding phrase, observed

at initialization and thereafter obtained from the previous corrected rollout. We retain a position-dependent fraction of each raw row's displacement from this reference:

$$\mathbf{v}^{\mathrm{A}}_{\tau+1,m} = \mathbf{v}^{\mathrm{ctx}}_{\tau+1,0} + \alpha_m \left(\mathbf{v}^{\mathrm{raw}}_{\tau+1,m} - \mathbf{v}^{\mathrm{ctx}}_{\tau+1,0}\right). \tag{10}$$

Here α_m controls the balance between continuity and planned change: $\alpha_m = 1$ leaves the row unchanged, while $\alpha_m = 0$ copies the boundary state. Intermediate values reduce the departure from the inherited context while retaining part of the planned displacement, before the trajectory reaches the adapter and executor. The equivalent context-regularized objective is given in Appendix C.2.

Correction memory and propagation. The corrected endpoint carries slow context into the next phrase, but does not explicitly record how the preceding plan was adjusted. A refreshed plan may introduce a similar mismatch. To retain this correction information across replanning steps, BHC records the endpoint displacement caused by anchoring:

$$\boldsymbol{\delta}^{\mathrm{A}}_{\tau+1} = \mathbf{v}^{\mathrm{A}}_{\tau+1,M} - \mathbf{v}^{\mathrm{raw}}_{\tau+1,M}. \tag{11}$$

We summarize these displacements in a smoothed memory $\bar{\boldsymbol{\delta}}_\tau$. The memory accumulated through phrase τ supplies a carryover adjustment to the anchored trajectory for phrase $\tau + 1$:

$$\tilde{\mathbf{v}}^{\mathrm{BHC}}_{\tau+1,m} = \mathbf{v}^{\mathrm{A}}_{\tau+1,m} + g\beta_m \bar{\boldsymbol{\delta}}_\tau, \tag{12}$$

where $g \geq 0$ controls the carryover strength. The coefficients β_m, with $1 \geq \beta_1 \geq \cdots \geq \beta_M \geq 0$, make the carryover contribution non-increasing across positions, allowing the new plan to play a greater role at later positions. With zero initial memory, the first corrected phrase uses anchoring only. To retain persistent adjustments while smoothing variation across boundaries, BHC updates the memory after each phrase:

$$\bar{\boldsymbol{\delta}}_{\tau+1} = \begin{cases} \boldsymbol{\delta}^{\mathrm{A}}_{\tau+1}, & \tau = \tau_0, \\ \rho\bar{\boldsymbol{\delta}}_\tau + (1-\rho)\boldsymbol{\delta}^{\mathrm{A}}_{\tau+1}, & \tau > \tau_0. \end{cases} \tag{13}$$

The update records only the anchoring displacement $\boldsymbol{\delta}^{\mathrm{A}}$, so the propagated term is not directly accumulated again in memory. The parameter ρ controls retention of the previous correction, while β_m controls its within-phrase decay. For $0 \leq \rho < 1$ and bounded anchoring displacements, the correction memory remains bounded.

Corrected rollout. The corrected trajectory conditions the executor through the frozen adapter. At each boundary, BHC retains the corrected pre-adapter endpoint, generated latent and control histories, and smoothed correction memory, while keeping learned weights and commanded actions fixed. To assess drift correction, we compare predicted-plan (PP), teacher-plan (TT), and BHC rollouts under matched weights, observed prefixes, actions, and segment-level controls. PP uses predicted slow trajectories; TT supplies target-derived slow references to define the diagnostic PP–TT gap; BHC corrects trajectories from its rollout history without future reference music. Coefficients are selected on 81 development songs and fixed for held-out testing (Appendix B.2).

4 Experiments

4.1 Experimental Setup

All experiments on Slakh-Action follow Slakh2100 Redux's predefined song-level splits. An 81-song development dataset, excluded from codec and predictor training, is used solely for inference-time BHC coefficient selection; all reported evaluations use held-out test data. Our primary cross-model evaluation uses an 8-bar prefix, zero actions, and planner-predicted controls to generate an 8-bar continuation. Separate command-conditioned evaluations measure counterfactual action responses and 32-/64-bar recursive prediction with matched prefixes and external commands. Dataset details and evaluation protocols are given in Appendices A.1 and D.3.

4.2 Cross-Model Evaluation

We compare MWM and MWM+BHC with six public baselines on 102 held-out songs, each generating an 8-bar continuation from the same 8-bar MIDI prefix. Baselines use released checkpoints and native inference settings; MWM and MWM+BHC use zero actions and planner-predicted

Table 1: Objective evaluation of 8-bar continuation on 102 songs (song-level mean ± SD). All values except RBDE are scaled by 10. C-JSD/R-JSD denote Chroma/Rhythm JSD; bold marks the best unrounded mean.

Method	RBDE↓	C-JSD↓	SC↑	R-JSD↓	GC↑	RSE↓	TSSE↓
BEAT	$6.77_{\pm 0.68}$	$5.09_{\pm 0.74}$	$9.21_{\pm 0.38}$	$4.68_{\pm 0.67}$	$8.15_{\pm 0.51}$	$4.32_{\pm 0.62}$	$2.81_{\pm 0.55}$
Music2Music	$6.63_{\pm 0.64}$	$3.63_{\pm 0.58}$	$\mathbf{9.84}_{\pm 0.21}$	$4.41_{\pm 0.63}$	$8.57_{\pm 0.45}$	$4.35_{\pm 0.59}$	$1.11_{\pm 0.31}$
MuPT	$3.74_{\pm 0.51}$	$4.06_{\pm 0.62}$	$9.18_{\pm 0.39}$	$3.45_{\pm 0.54}$	$8.90_{\pm 0.37}$	$3.40_{\pm 0.52}$	$1.24_{\pm 0.34}$
NotaGen	$3.07_{\pm 0.44}$	$3.48_{\pm 0.55}$	$8.66_{\pm 0.46}$	$2.77_{\pm 0.48}$	$8.95_{\pm 0.36}$	$2.70_{\pm 0.45}$	$0.90_{\pm 0.27}$
Amadeus	$5.28_{\pm 0.59}$	$3.59_{\pm 0.57}$	$9.65_{\pm 0.28}$	$4.24_{\pm 0.61}$	$9.13_{\pm 0.33}$	$3.70_{\pm 0.51}$	$0.92_{\pm 0.28}$
Text2MIDI	$5.44_{\pm 0.61}$	$5.09_{\pm 0.71}$	$9.50_{\pm 0.31}$	$4.64_{\pm 0.66}$	$8.24_{\pm 0.48}$	$3.50_{\pm 0.50}$	$1.59_{\pm 0.38}$
MWM	$\mathbf{2.70}_{\pm 0.41}$	$3.30_{\pm 0.51}$	$9.84_{\pm 0.20}$	$2.57_{\pm 0.44}$	$9.20_{\pm 0.31}$	$2.62_{\pm 0.42}$	$1.01_{\pm 0.29}$
MWM + BHC	$2.95_{\pm 0.43}$	$\mathbf{2.70}_{\pm 0.47}$	$9.66_{\pm 0.25}$	$\mathbf{2.28}_{\pm 0.41}$	$\mathbf{9.27}_{\pm 0.29}$	$\mathbf{2.56}_{\pm 0.43}$	$\mathbf{0.76}_{\pm 0.24}$

Table 2: Subjective listening evaluation on a five-point scale (mean ± SD; higher is better). Each method receives 145 submissions from 29 listeners rating five matched excerpts.

Method	Overall↑	Musicality↑	Coherence↑	Continuity↑	Structure↑
BEAT	$3.62_{\pm 0.60}$	$3.71_{\pm 0.64}$	$3.64_{\pm 0.61}$	$3.55_{\pm 0.58}$	$3.58_{\pm 0.68}$
Music2Music	$3.57_{\pm 0.64}$	$3.68_{\pm 0.60}$	$3.55_{\pm 0.62}$	$3.59_{\pm 0.66}$	$3.53_{\pm 0.62}$
MuPT	$3.79_{\pm 0.62}$	$3.84_{\pm 0.61}$	$3.81_{\pm 0.58}$	$3.72_{\pm 0.59}$	$3.69_{\pm 0.60}$
NotaGen	$3.93_{\pm 0.56}$	$\mathbf{4.06}_{\pm 0.54}$	$3.91_{\pm 0.58}$	$3.87_{\pm 0.56}$	$3.94_{\pm 0.58}$
Amadeus	$3.96_{\pm 0.63}$	$3.98_{\pm 0.61}$	$3.89_{\pm 0.59}$	$4.04_{\pm 0.62}$	$\mathbf{4.11}_{\pm 0.64}$
Text2MIDI	$3.73_{\pm 0.58}$	$3.79_{\pm 0.63}$	$3.76_{\pm 0.60}$	$3.68_{\pm 0.62}$	$3.71_{\pm 0.60}$
MWM	$4.04_{\pm 0.56}$	$4.03_{\pm 0.58}$	$\mathbf{4.08}_{\pm 0.55}$	$3.99_{\pm 0.65}$	$4.02_{\pm 0.59}$
MWM + BHC	$\mathbf{4.08}_{\pm 0.62}$	$4.01_{\pm 0.64}$	$4.04_{\pm 0.64}$	$\mathbf{4.09}_{\pm 0.63}$	$4.08_{\pm 0.62}$

segment-level descriptors from the observed prefix. Shared metrics measure role density (RBDE), pitch/rhythm distributions (Chroma/Rhythm JSD), role relationships (RSE), temporal self-similarity (TSSE), and scale/groove consistency (SC/GC), with equal weight per song. Table 1 shows that MWM matches reference role densities more closely than all six baselines: its RBDE is 2.70, compared with the best baseline RBDE of 3.07 (NotaGen). MWM+BHC achieves the best mean on Chroma JSD, Rhythm JSD, GC, RSE, and TSSE; its TSSE of 0.76, compared with NotaGen's 0.90, indicates closer agreement with the reference's temporal organization. Relative to MWM, BHC improves these five measures while increasing density error and lowering scale consistency. Metrics and protocols are in Appendix D; matched architecture ablations are in Appendix E.1.

4.3 Subjective Evaluation

Twenty-nine listeners each rated five matched held-out excerpts from all eight methods, with method identities hidden and presentation order randomized. They scored overall quality, musicality, local coherence, long-horizon continuity, and structural consistency on a five-point scale, yielding 1,160 submissions. Rendering, rating anchors, and statistical procedures are detailed in Appendix E.2. Table 2 shows that MWM+BHC receives the highest mean overall-quality (4.08) and continuity (4.09) scores, while MWM leads local coherence (4.08). The strongest method varies by dimension: NotaGen leads musicality and Amadeus structural consistency. BHC's gains over MWM are small (0.04 for overall quality and 0.10 for continuity), and all five paired 95% confidence intervals include zero (Appendix E.2); the study therefore provides no conclusive evidence that objective gains translate into perceptual improvements on these five excerpts.

4.4 Counterfactual Action Evaluation

On the 32-bar test dataset (102 songs, 1,024 windows), we compare MWM, MWM+BHC, and an action-masked control under five density-command levels. Commands are held for four phrases or reset to the original value at phrase 3. We measure direction accuracy, target error, response

Table 3: Counterfactual density-action evaluation on the 32-bar test set (102 songs, 1,024 windows). Panels use target response, five-level dose response, and reversal eligibility rules. Brackets show 95% paired song-bootstrap confidence intervals.

Density target response (test split; all eligible points)

Method	Dir.↑	Target Err.↓	Success↑	Off-target↓	Late Err.↓
MWM (action masked)	18.49% [15.53,21.63]	0.745 [0.730,0.765]	0.72% [0.27,1.30]	0.0022 [0.0018,0.0026]	0.799 [0.776,0.822]
MWM	83.46% [79.27,87.31]	0.720 [0.706,0.739]	3.47% [1.95,5.18]	0.0080 [0.0069,0.0094]	0.751 [0.733,0.769]
MWM + BHC	81.68% [77.05,86.15]	0.719 [0.709,0.730]	3.04% [1.66,4.68]	0.0077 [0.0065,0.0090]	0.779 [0.758,0.801]

Density dose response (test split; all eligible points)

Method	ρ_O ↑	Order Acc.↑	Gain→ 1	Off-target↓	Late ρ_O ↑
MWM (action masked)	-0.047 [-0.148,0.057]	21.90% [18.16,25.92]	-0.001 [-0.001,0.000]	0.0022 [0.0018,0.0027]	-0.885 [-0.932,-0.835]
MWM	0.881 [0.832,0.923]	86.79% [83.22,90.13]	0.043 [0.032,0.057]	0.0070 [0.0061,0.0080]	-0.379 [-0.516,-0.243]
MWM + BHC	0.860 [0.810,0.905]	84.27% [80.29,88.12]	0.037 [0.028,0.049]	0.0068 [0.0058,0.0080]	-0.707 [-0.792,-0.617]

Density reversal (test split; all eligible points)

Method	Dir.↑	Target Err.↓	Success↑	Off-target↓	Late Err.↓
MWM (action masked)	18.83% [16.20,21.63]	0.205 [0.173,0.243]	11.29% [9.22,13.51]	0.0020 [0.0017,0.0023]	0.285 [0.239,0.336]
MWM	83.60% [80.23,86.91]	0.187 [0.158,0.220]	60.03% [53.66,65.78]	0.0067 [0.0056,0.0081]	0.241 [0.206,0.282]
MWM + BHC	81.10% [77.33,84.92]	0.196 [0.163,0.234]	57.12% [50.45,63.13]	0.0073 [0.0063,0.0085]	0.270 [0.223,0.323]

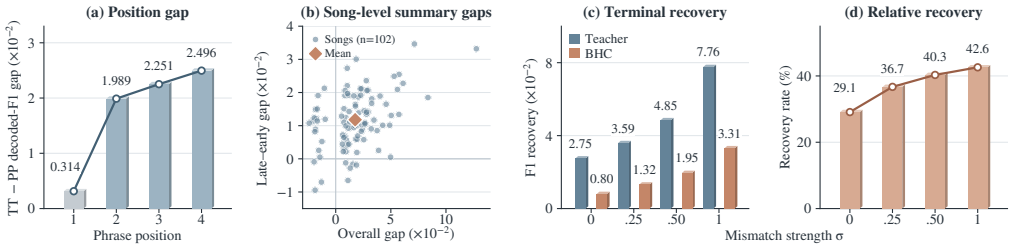


Figure 2: PP–TT diagnostics and plan-mismatch robustness. (a,b) F1 gaps by position and song (32 bars; circles: songs; diamond: mean). (c,d) Terminal recovery and BHC/TT recovery ratio under increasing mismatch (64 bars). F1 values in (a–c) use 10^{-2} units.

ordering and gain, off-target changes, and reversal recovery. Success requires the correct direction and target error within 0.25 training-split standard deviations; gain 1 denotes matched response magnitude. Table 3 shows strong initial command sensitivity: MWM achieves 83.5% directional and 86.8% ordering accuracy, versus 18.5% and 21.9% with actions masked, while MWM+BHC achieves 81.7% and 84.3%, respectively. Both methods respond to command direction and relative strength in the first phrase, although response gains of 0.043 and 0.037 remain far below the target of 1. Definitions, eligibility, and aggregation are in Appendix D.4.

4.5 Long-Horizon Effectiveness and Plan Stability

We compare predicted-plan (PP), teacher-plan (TT), and BHC rollouts with matched prefixes, commands, and model weights. TT supplies target-derived slow plans as a diagnostic reference. The first-phrase PP–TT diagnostic uses the 32-bar test dataset (102 songs, 1,024 windows) with non-slow plan branches disabled; full 32-/64-bar rollouts activate all branches, with 44 songs and 396 windows at 64 bars. We measure decoded-F1 gains over PP, using the terminal step for full rollouts, and rollout-averaged latent/logit RMS distances to TT. Windows and six fixed inference settings are averaged within song, then songs receive equal weight (Appendix C.1). Figure 2(a,b) shows larger PP–TT F1 gaps toward later phrase positions, consistent with plan-related drift during recursive execution. Figure 3(a,b) shows that BHC recovers 18.7% and 29.1% of the PP–TT terminal-F1 gap at 32 and 64 bars, respectively, with an absolute F1 gain of 0.008 at 64 bars. The lower latent-state and decoder-logit distances in Figure 3(c,d) further show that BHC brings internal rollouts closer to TT, linking history correction to more stable long-horizon generation.

Table 4: Ablation comparison of BHC components. F1 gain is measured at the terminal step relative to MWM; latent and logit reductions are decreases in rollout-averaged RMS distance to TT. The 32- and 64-bar rows use the 102- and 44-song test datasets, respectively.

Method	F1 Gain↑	95% CI	Latent Red.↑	Logit Red.↑
32 bars: 102-song test dataset				
Full BHC	**+0.0031**	**[-0.0021,+0.0085]**	0.050	0.019
Anchoring only	0.0026	[-0.0024,+0.0072]	**0.061**	**0.027**
Propagation only	0.0010	[-0.0030,+0.0046]	0.032	0.013
EMA slow-plan	0.0012	[-0.0028,+0.0051]	0.036	0.015
Reverse propagation	0.0005	[-0.0031,+0.0034]	0.025	0.011
Norm-matched random	−0.0009	[-0.0045,+0.0014]	−0.006	−0.002
64 bars: 44-song test dataset				
Full BHC	**+0.0080**	**[+0.0030,+0.0178]**	**0.165**	**0.088**
Anchoring only	0.0068	[+0.0005,+0.0132]	0.119	0.062
Propagation only	0.0045	[+0.0007,+0.0093]	0.072	0.038
EMA slow-plan	0.0023	[-0.0012,+0.0059]	0.051	0.024
Reverse propagation	0.0012	[-0.0026,+0.0050]	0.035	0.019
Norm-matched random	−0.0002	[-0.0013,+0.0011]	0.012	0.005

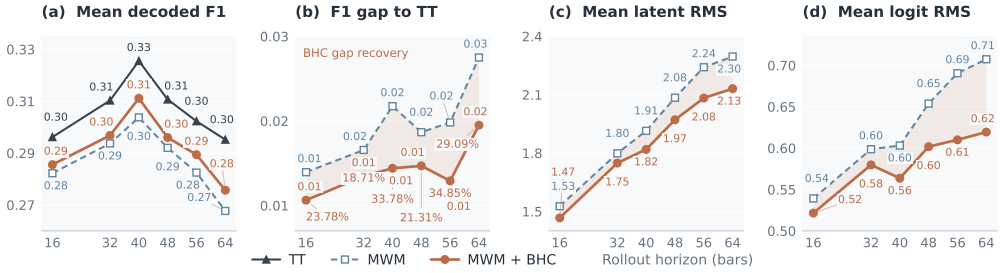


Figure 3: Full recursive rollout on the test split. (a) Terminal decoded F1. (b) F1 gaps to TT; percentages show the fraction recovered by BHC. (c,d) Latent and logit RMS distances to TT, averaged over rollout steps and equally across songs. The test datasets contain 102 and 44 songs for 32- and 64-bar continuation, respectively.

4.6 Mechanism and Robustness to Plan Mismatch

All rows use the frozen checkpoint, prefixes, action schedules, decoder, and six deterministic settings, with no future reference music or teacher plans. Controls remove or reverse correction branches; terminal F1 and latent/logit RMS evaluation uses 10,000 paired song-bootstrap replicates.

Table 4 compares the five controls. At 32 bars, all F1-gain intervals include zero and anchoring gives the largest RMS reductions. At 64 bars, full BHC gains 0.0080 F1 versus 0.0068 for anchoring and 0.0045 for propagation, while reducing latent/logit distances by a further 0.046/0.026 over anchoring. EMA, reverse propagation, and random updates give gains of 0.0023, 0.0012, and −0.0002. Under the paired stress process in Appendix C.2, BHC recovery rises from 0.0080 to 0.0331 as mismatch increases from 0 to 1, retaining 29.1%–42.6% of teacher recovery (Figure 2(c,d)). The stress test probes controlled robustness and does not estimate the empirical planner-error distribution.

5 Conclusion

MWM couples phrase-level planning with segment-level execution for controllable symbolic music generation. On Slakh-Action, MWM achieves the lowest RBDE, while MWM+BHC leads five other metrics. Counterfactual evaluations confirm density-command following in the first phrase. At 64 bars, BHC improves terminal decoded F1 by 0.0080 and recovers 29.1% of the PP–TT gap, reaching 42.6% under the strongest mismatch. Thus, inference-time history correction improves long-horizon prediction without retraining or future reference music.

REFERENCES

Keshav Bhandari, Abhinaba Roy, Kyra Wang, Geeta Puri, Simon Colton, and Dorien Herremans. Text2midi: Generating symbolic music from captions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23478–23486, 2025. doi: 10.1609/aaai.v39i22.34516.

Boyu Cao, Lekai Qian, Dehan Li, Haoyu Gu, Mingda Xu, and Qi Liu. Anchored cyclic generation: A novel paradigm for long-sequence symbolic music generation, 2026.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. doi: 10.1038/s41586-025-08744-2.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Scott H. Hawley. Helping music co-creation agents 'listen' well: Hierarchical self-supervised world models for understanding and generation, 2026.

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

Yujia Huang, Adishree Ghatare, Yuanzhe Liu, Ziniu Hu, Qinsheng Zhang, Chandramouli Shama Sastry, Siddharth Gururani, Sageev Oore, and Yisong Yue. Symbolic music generation with non-differentiable rule guided diffusion. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 19772–19797. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24g.html.

Haoxin Lin, Siyuan Xiao, Yi-Chen Li, Zhilong Zhang, Yihao Sun, Chengxing Jia, and Yang Yu. ADM-v2: Pursuing full-horizon roll-out in dynamics models for offline policy learning and evaluation. In *International Conference on Learning Representations*, 2026a.

Zhiwei Lin, Jun Chen, Boshi Tang, Weihao Wu, Yang Jing, Yaolong Ju, Fan Fan, and Zhiyong Wu. Diff-Symbo: Text-controlled long-duration symbolic music generation using autoregressive latent diffusion model, 2026b. URL https://arxiv.org/abs/2608.05222.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. In *International Conference on Learning Representations*, 2026.

Ang Lv, Xu Tan, Peiling Lu, Wei Ye, Shikun Zhang, Jiang Bian, and Rui Yan. GETMusic: Generating music tracks with a unified representation and diffusion framework. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 10144–10152, 2025. doi: 10.24963/ijcai.2025/1127.

Martin E. Malandro. Composer's assistant 2: Interactive multi-track MIDI infilling with fine-grained user control. In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pp. 438–445, 2024.

Longshen Ou, Jingwei Zhao, Ziyu Wang, Gus Xia, Qihao Liang, Torin Hopkins, and Ye Wang. Unifying symbolic music arrangement: Track-aware reconstruction and structured tokenization. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Lekai Qian, Haoyu Gu, Jingwei Zhao, and Ziyu Wang. Beat: Tokenizing and generating symbolic music by uniform temporal steps. In *Proceedings of the International Conference on Machine Learning*, 2026.

Xingwei Qu, Yuelin Bai, Yinghao Ma, Ziya Zhou, Ka Man Lo, Jiaheng Liu, Ruibin Yuan, Lejun Min, Xueling Liu, Tianyu Zhang, Xinrun Du, Shuyue Guo, Yiming Liang, Yizhi Li, Shangda Wu, Junting Zhou, Tianyu Zheng, Ziyang Ma, Fengze Han, Wei Xue, Gus Xia, Emmanouil Benetos, Xiang Yue, Chenghua Lin, Xu Tan, Stephen W. Huang, Jie Fu, and Ge Zhang. MuPT: A generative symbolic music pretrained transformer. In *International Conference on Learning Representations*, 2025.

Hongju Su, Ke Li, Lan Yang, Honggang Zhang, and Yi-Zhe Song. Amadeus: Autoregressive model with bidirectional attribute modelling for symbolic music. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 40910–40928. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1898.

Yashan Wang, Shangda Wu, Jianhuai Hu, Xingjian Du, Yueqi Peng, Yongxin Huang, Shuai Fan, Xiaobing Li, Feng Yu, and Maosong Sun. NotaGen: Advancing musicality in symbolic music generation with large language model training paradigms. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 10207–10215, 2025. doi: 10.24963/ijcai.2025/1134.

Ziyu Wang, Lejun Min, and Gus Xia. Whole-song hierarchical generation of symbolic music using cascaded diffusion models. In *International Conference on Learning Representations*, 2024.

Ziyu Wang, Kun Fang, and Yann LeCun. Music-JEPA: Learning a world model of sound from action, 2026.

Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, Xun Cao, and Wei Yin. Epona: Autoregressive diffusion world model for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27220–27230, 2025.

A Data, Representation, and Action Interface

A.1 Data and symbolic representation

Slakh2100 Redux is partitioned by song before segmentation (Table 5). Each segment spans two $4/4$ bars at 16 grid cells per bar; four segments form an eight-bar phrase. Unreadable sources, unsupported meters, and segments without an active role are excluded; pitches outside MIDI 0–127 are ignored. The 81-song development dataset is excluded from both codec and predictor training and is used only for inference-only BHC coefficient selection. All evaluations use the untouched test split. Internal long-horizon continuation windows require a 48-bar observed prefix; the cross-model comparison uses its separate native 8-bar prefix protocol.

Table 5: Data partitions and uses after preprocessing.

Partition / use	Songs	Contents
Frozen codec base pretraining	1,160	Train-split segments
Joint predictor training	1,160	9,847 eligible windows
BHC coefficient development	81	Inference-only selection
Full test split	150	15,470 segments; 3,662 phrases
8-bar cross-model comparison	102	Shared test dataset; native 8-bar continuation
32-bar test continuation	102	1,024 windows
64-bar test continuation	44	396 windows

A segment is $\mathbf{x}_t \in [0,1]^{6\times32\times128\times3}$, with channels for binary onset o, binary sustain q, and velocity $v/127$; activity is $o \vee q$. Roles are melody, bass, chords, drums, accompaniment, and other, in that order. Assignment prioritizes drum/bass metadata, then instrument class, pitch, and monophony tests; same-role stems merge by channelwise maxima. These role labels are produced by a computer annotation pipeline. We manually spot-checked sampled assignments and boundary cases as a quality-control step, including role identity and carry-in sustain behavior. Quantization precedes boundary clipping, retaining carry-in sustain without a new onset. A phrase has shape $[4, 6, 32, 128, 3]$; members are joined along time before event extraction to preserve sustained notes.

A.2 Control descriptors and actions

The segment descriptor is $\mathbf{u}_t = [b_t/240, n_t/32, m_{t,0}, \ldots, m_{t,5}] \in \mathbb{R}^8$: b_t is starting tempo in BPM, n_t counts source notes intersecting the segment (including carry-in notes), and $m_{t,r}$ indicates role activity. Phrase controls and actions are

$$\mathbf{c}^s_\tau = \frac{1}{4}\sum_{m=1}^{4} \mathbf{u}^s_{\tau,m}, \qquad \mathbf{a}^s_{\tau+1} = P_{\mathcal{A}}(\mathbf{c}^s_{\tau+1} - \mathbf{c}^s_\tau), \quad s \in \{\mathrm{obs}, \mathrm{cmd}\}. \tag{14}$$

$P_{\mathcal{A}}$ retains controllable coordinates and zeros the rest. Training uses observed descriptors. The primary cross-model evaluation fixes all eight action coordinates to zero and uses segment-level control descriptors predicted by the planner from the observed prefix. These predicted descriptors condition the executor; their differences do not replace the zero action inputs. This evaluation uses no external control schedule or LLM-generated action. Separate command-conditioned evaluations use externally supplied actions and descriptors. The interactive LLM API maps natural-language instructions to the same normalized action coordinates. In paired PP–BHC–TT comparisons, the four segment-level control descriptors per phrase are fixed in advance and shared across paths, without future-reference or TT-derived inputs. They remain in raw coordinates during execution; frozen training standard deviations scale losses only. In these paired rollouts, supplied descriptors update executor histories and planner summaries; predicted descriptors are auxiliary outputs. A hold command preserves the commanded control state while music can evolve. In cross-model generation, planner-predicted descriptors instead condition the executor during generation.

B Model Architecture and Training

B.1 Model Architecture and History Buffers

Table 6 lists dimensions without the batch axis. Planner and executor Transformers use causal attention, pre-layer normalization, GELU, zero dropout, and final normalization of the last token.

Table 6: MWM architecture; $M = N = K = 4$.

Component	Specification
Segment latent	$10{,}496 = 512$ global $+6 \times 128$ track $+6 \times 8 \times 192$ local coordinates
Codec decoder	Width 192; 3 layers; 6 heads; feed-forward width 768
Planner / executor	Width 768; 8 / 12 layers; 12 heads; feed-forward width 3,072
Plan heads	Global: 521; member: $4 \times 10{,}504$; horizon: 4×521; slow: 512
Latent increment heads	Global: $768 \to 512 \to 512$; detail: $1{,}280 \to 768 \to 9{,}984$
Slow adapter	Residual MLP: $1{,}537 \to 1{,}024 \to 512$; zero-initialized output layer
History lengths	Planner: 6 phrases; executor: 4 latent, control, and slow rows

Codec and planner. The encoder maps each role grid through a shared $12{,}288 \to 512$ affine layer and layer normalization. A $3{,}072 \to 1{,}024 \to 512$ global MLP uses GELU and intermediate layer normalization. Each role's 128-D track state is conditioned on the global state. Eight four-step chunks per role are projected $1{,}536 \to 192$, combined with learned time embeddings, layer normalization, and projected track states, then affinely mapped to local tokens. Latents concatenate global, track, and local states; the first 512 coordinates form the slow state. The decoder projects the global, six track, and 48 local states to width 192. Its 192 role/time queries attend to this memory using GELU, pre-layer normalization, zero dropout, and learned role/time/type embeddings. An affine head and sigmoid produce 128 pitches and three channels per query.

Each planner token sums projections of the pooled global latent (512-D), phrase summary (9-D), and flattened member controls (32-D), without additional positional embeddings. The summary orders mean tempo, mean density, density standard deviation, and six mean role activities. Global/slow heads are affine; member/horizon heads add learned slot vectors and use two-layer GELU MLPs of hidden width 768. Outputs concatenate latent then summary for the global head, controls then full latent for each member, and summary then pooled latent for each horizon.

Plan-to-trajectory conversion and segment-level executor. This conversion maps the four-slot phrase plan to four slow-state rows, one for each segment generated in the next phrase. The resulting rows are passed through the shared adapter and then supplied to the recurrent executor at the corresponding segment positions. With affine maps $P_{\mathrm{mem}} : 10{,}504 \to 512$ and $P_{\mathrm{hor}} : 521 \to 512$, the raw trajectory is

$$\mathbf{v}_m = \hat{\mathbf{s}} + \frac{\lambda_{\mathrm{mem}}}{4} \sum_{i=1}^{4} P_{\mathrm{mem}}(\hat{\mathbf{m}}_i) + \lambda_{\mathrm{hor}} P_{\mathrm{hor}}(\hat{\mathbf{h}}_m), \quad m = 1, \ldots, 4. \tag{15}$$

Both scales equal one in full rollouts; horizon slots map directly to current-phrase conditioning positions. The shared adapter concatenates $\mathbf{v}_m$, the last stored pre-adapter state, $\hat{\mathbf{s}}$, and $(m-1)/3$, applies its GELU MLP, and adds the result to $\mathbf{v}_m$.

The executor sums projections of latent/control/slow histories into 768-D tokens. Its global increment is scaled by 0.25 and concatenated with the last hidden state to predict the detail increment; both heads use GELU. The concatenated increment is added to the last latent. Next-segment controls are projected into every token with scale 0.5 and coordinate scales $(0, 1, 2)$ for tempo, density, and roles. Member-dependent latent modulation, plan-derived control offsets, and horizon latent residuals have scales $(0.1, 0.1, 0.05)$; multiplicative member modulation uses $\tanh$.

The prefix initializes latent/control histories; its last phrase's mean global latent fills four slow rows. Within a phrase, the executor appends its predicted latent, supplied control, and adapted slow row to its buffers. The horizon residual is added after the four-step executor rollout. At the boundary, stored histories use full predicted latents, supplied controls, and *pre-adapter* slow rows. Pooled generated global latents and supplied control summaries update the six-phrase planner history. BHC modifies the trajectory before adaptation and retains its corrected pre-adapter context and 512-D memory.

B.2 Training Objectives, Optimization, and Parameter Selection

The reported predictor is a fixed-budget, fixed-configuration instance of MWM (architecture in Appendix B.1). After codec pretraining, the codec is frozen and the planner, projections, adapter, and executor are jointly trained from random initialization. Table 7 gives budgets; data roles are in Table 5.

Table 7: Training budgets, both using AdamW.

Stage	Batch	Learning rate	Budget
Frozen codec base pretraining	8	2×10^{-4}	5 epochs
Joint planner/executor	32	10^{-4}	3,500 updates

Codec objective. Let $\langle\cdot\rangle$ denote an elementwise mean and $B_w(p, y) = -\langle wy \log p + (1 - y) \log(1 - p)\rangle$. For sigmoid outputs $\hat{o}, \hat{q}, \hat{v}$ and target activity $a = \mathbb{1}[o + q > 0]$,

$$\mathcal{L}_{\text{codec}} = B_8(\hat{o}, o) + 0.75B_4(\hat{q}, q) + 2\langle|\max(\hat{o}, \hat{q}) - a|\rangle + \frac{\sum a|\hat{v} - v|}{\max(1, \sum a)}. \tag{16}$$

The epoch-five checkpoint supplies cached targets $\mathbf{z} = E_\phi(\mathbf{x}) = [\mathbf{z}^g; \mathbf{z}^d]$; the detail block contains track and local coordinates.

Joint objective. Each eligible window contains up to six prefix phrases (zero-filled when missing) and five contiguous future phrases. State losses supervise the first future phrase's four recursively predicted executor states without detaching gradients; the horizon head supervises the first four future phrases. BHC is disabled. For coordinate scales $\boldsymbol{\sigma} > 0$, define

$$Q_{\boldsymbol{\sigma}}(\hat{\mathbf{y}}, \mathbf{y}) = \left\langle ((\hat{\mathbf{y}} - \mathbf{y}) \oslash \boldsymbol{\sigma})^2 \right\rangle. \tag{17}$$

Frozen training standard deviations are $\boldsymbol{\sigma}_z$ and $\boldsymbol{\sigma}_u$; phrase scales $\boldsymbol{\sigma}_c$ contain the tempo scale, two density scales, and six role scales. Head losses L_g, L_s, L_m, L_h use Q with scales $[\boldsymbol{\sigma}_z^g; \boldsymbol{\sigma}_c]$, $\boldsymbol{\sigma}_z^g$, $[\boldsymbol{\sigma}_u; \boldsymbol{\sigma}_z]$, and $[\boldsymbol{\sigma}_c; \boldsymbol{\sigma}_z^g]$, respectively, against the corresponding latent/control targets. Global and slow targets use the first future phrase's pooled global latent; global targets also include its nine controls. Member targets use the corresponding segment, and horizon targets use the corresponding future phrase. L_u additionally supervises member control coordinates. The objective is

$$\mathcal{L}_{\text{joint}} = 0.5Q_{\boldsymbol{\sigma}_z^g}(\hat{Z}^g, Z^g) + Q_{\boldsymbol{\sigma}_z^d}(\hat{Z}^d, Z^d) + 0.25L_g + 3L_s + 0.25L_m + 0.25L_h + 0.25L_u. \tag{18}$$

State losses average equally across four segments. Training conditions on observed future member controls while plans and latent feedback are model-generated. The primary cross-model evaluation replaces observed future controls with planner-predicted descriptors and uses zero actions; separate command-conditioned evaluations use the external schedules described in Appendix A.2.

Joint AdamW uses $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, weight decay 0.01, and constant learning rate. Each update samples 32 distinct windows uniformly from 9,847 eligible windows, accumulates two microbatches of 16, and clips the global gradient norm to 1.0. Float32 is used with autocast/TF32 disabled. The 112,000 window draws cover all eligible windows, allowing repeats across updates. The update-3,500 checkpoint is prespecified; test data are not used for training or selection.

BHC coefficients. Table 8 is fixed for all test horizons. The 81-song development sweep evaluates 26 joint configurations subject to prespecified bounds on action-response changes, then maximizes

$$G(\vartheta) = \frac{1}{6|H|} \sum_{s=1}^{6} \sum_{h \in H} \left(\text{F1}_{h,s}^{\text{MWM+BHC}(\vartheta)} - \text{F1}_{h,s}^{\text{MWM}} \right), \quad H = \{32, 64\} \text{ bars}. \tag{19}$$

Candidate values and constraint thresholds are fixed before evaluation. The selected configuration is frozen before testing; BHC adds no learned checkpoint and all learned parameters remain fixed.

Table 8: Fixed inference coefficients for BHC.

Parameter	Value
Anchoring $\alpha_{1:4}$	$(0.4, 0.7, 0.575, 0.35)$
Gain g; decay ρ	0.75; 0.75
Propagation $\beta_{1:4}$	$(1, 0.75, 0.5, 0.25)$
λ_{mem}; λ_{hor}	1; 1
Global control branch; activity threshold	Enabled; 0.5

C Receding-Horizon Rollout and BHC

C.1 Receding-Horizon Rollout and PP–TT–BHC Pairing

Each path starts from a private copy of the same six-phrase prefix, including four latent/control/slow rows and six planner entries. Paths share commanded controls, frozen weights, decoder, scoring targets, window IDs, and inference settings. PP and BHC plan from their own generated histories; BHC additionally corrects its pre-adapter trajectory. TT substitutes target-derived slow conditioning while fast-state execution remains recursive, so it is a nondeployable diagnostic. No reference fast state is appended to any path.

Paired phrase rollout. For each phrase and each path:

1. Load the fixed four-row command schedule. Build the predicted trajectory (PP/BHC) or supply the target slow reference (TT). For BHC, apply the anchoring/memory recurrence in Appendix C.2, using its own context and memory.
2. Apply the shared adapter and run four recursive executor steps, updating within-phrase buffers as specified in Appendix B.1. Add the horizon residual, decode, and score against the aligned reference.
3. Retain generated latents, supplied controls, and pre-adapter slow rows. Pool generated latents and append them with command summaries to PP/BHC planner histories; refresh TT's slow reference at the next boundary.

The first BHC phrase uses anchoring alone and initializes memory with the full endpoint displacement; later phrases use the previous memory. The within-phrase PP–TT diagnostic uses only the first phrase, observed prefix slow context, and disabled non-slow plan branches. Chained rollouts keep all branches active.

Units and pairing. The within-phrase diagnostic and 32-bar rollout use the 32-bar test dataset in Table 5; the 64-bar rollout and controlled mismatch study use its 64-bar test dataset. Counterfactual panels use eligible songs and decision points in the 32-bar test dataset. The diagnostic scores four segment positions per window (4,096 per path/setting; 24,576 across six settings). Chained rollouts score four or eight phrase steps at 32 or 64 bars. All evaluated tensors were finite and scored. The six settings are deterministic inference configurations of the *same checkpoint*, not independent training seeds or stochastic generations. Pairing uses setting/song/window/phrase/local-step IDs before aggregation.

Windows and settings are averaged within song, then songs receive equal weight. The PP–TT diagnostic averages the four position-specific gaps in Figure 2(a), giving $\mathrm{F1}_{\mathrm{TT}} - \mathrm{F1}_{\mathrm{PP}} = 0.017625$ with 95% CI $[0.012392, 0.022305]$; pooled/window scores are secondary. All trajectory, position, component, and mismatch contrasts use 10,000 paired song-bootstrap replicates and percentile intervals (2.5th–97.5th), without cross-horizon multiplicity correction.

C.2 BHC Derivations and Boundary-Conditioning Mechanism

Let $\mathbf{s}_{j,m}$ be the BHC path's raw pre-adapter row at phrase j, position $m \in \{1, \ldots, M\}$, with $M = 4$. The prefix supplies context $\mathbf{r}_0$; later contexts are the final corrected pre-adapter rows.

Anchoring. For $\kappa_m, \lambda_m \geq 0$ with $\kappa_m + \lambda_m > 0$, consider

$$J(\mathbf{v}) = \frac{\kappa_m}{2}\|\mathbf{v} - \mathbf{s}_{j,m}\|_2^2 + \frac{\lambda_m}{2}\|\mathbf{v} - \mathbf{r}_{j-1}\|_2^2. \tag{20}$$

Its Hessian is $(\kappa_m + \lambda_m)I \succ 0$; setting its gradient to zero yields the unique minimizer

$$\mathbf{A}_{j,m} = \mathbf{r}_{j-1} + \alpha_m(\mathbf{s}_{j,m} - \mathbf{r}_{j-1}), \qquad \alpha_m = \frac{\kappa_m}{\kappa_m + \lambda_m}. \tag{21}$$

BHC specifies α_m directly, equivalent to $(\kappa_m, \lambda_m) = (\alpha_m, 1 - \alpha_m)$, including endpoints. For any norm, $\|\mathbf{A}_{j,m} - \mathbf{r}_{j-1}\| = \alpha_m\|\mathbf{s}_{j,m} - \mathbf{r}_{j-1}\|$.

Correction memory. Separate the anchoring displacement from its smoothed memory:

$$\boldsymbol{\delta}_j^{\mathrm{A}} = \mathbf{A}_{j,M} - \mathbf{s}_{j,M} = (1 - \alpha_M)(\mathbf{r}_{j-1} - \mathbf{s}_{j,M}), \qquad \mathbf{C}_{j,m} = \mathbf{A}_{j,m} + g\beta_m\bar{\boldsymbol{\delta}}_{j-1}. \tag{22}$$

Here $\mathbf{r}_j = \mathbf{C}_{j,M}$, $\bar{\boldsymbol{\delta}}_0 = \mathbf{0}$, $g \geq 0$, and $1 \geq \beta_1 \geq \cdots \geq \beta_M \geq 0$. Update memory by

$$\bar{\boldsymbol{\delta}}_j = \begin{cases} \boldsymbol{\delta}_1^{\mathrm{A}}, & j = 1, \\ \rho\bar{\boldsymbol{\delta}}_{j-1} + (1 - \rho)\boldsymbol{\delta}_j^{\mathrm{A}}, & j \geq 2, \end{cases} \quad 0 \leq \rho < 1. \tag{23}$$

Thus the first phrase anchors only and retains its full displacement, without a factor $1 - \rho$. Memory excludes the propagated term in $\mathbf{C}_{j,M}$; including it would change the inherited coefficient to $\rho + (1 - \rho)g\beta_M$. Unrolling gives

$$\bar{\boldsymbol{\delta}}_j = \rho^{j-1}\boldsymbol{\delta}_1^{\mathrm{A}} + (1 - \rho)\sum_{i=2}^{j}\rho^{j-i}\boldsymbol{\delta}_i^{\mathrm{A}}. \tag{24}$$

The nonnegative coefficients sum to one, so memory lies in the convex hull of past anchoring displacements.

Conditional bound. If $\|\boldsymbol{\delta}_j^{\mathrm{A}}\|_2 \leq D < \infty$ for all j and g is finite, convexity and the triangle inequality imply

$$\begin{aligned} \|\bar{\boldsymbol{\delta}}_j\|_2 \leq D, \qquad \|\mathbf{C}_{j,m} - \mathbf{A}_{j,m}\|_2 \leq g\beta_m D, \\ \|\mathbf{C}_{j,m} - \mathbf{s}_{j,m}\|_2 \leq (1 - \alpha_m)\|\mathbf{r}_{j-1} - \mathbf{s}_{j,m}\|_2 + g\beta_m D. \end{aligned} \tag{25}$$

If raw-to-context mismatches are uniformly bounded by E, one may take $D = (1 - \alpha_M)E$. For two memories driven by the same displacement sequence, their difference contracts by ρ at each recursive update.

Anchoring-only and propagation-only controls. Anchoring-only sets the propagation gain to zero, so the pre-adapter trajectory is $\mathbf{C}_{j,m} = \mathbf{A}_{j,m}$ and the retained context is $\mathbf{r}_j = \mathbf{A}_{j,M}$. Propagation-only retains the anchoring residual calculation and memory update, but removes the direct anchoring update from the trajectory. With the coefficients in Table 8, it computes

$$\boldsymbol{\delta}_j^{\mathrm{aux}} = (1 - \alpha_M)(\mathbf{r}_{j-1} - \mathbf{s}_{j,M}), \tag{26}$$

$$\mathbf{C}_{j,m}^{\mathrm{P}} = \mathbf{s}_{j,m} + g\beta_m\bar{\boldsymbol{\delta}}_{j-1}^{\mathrm{P}}, \qquad \mathbf{r}_j = \mathbf{C}_{j,M}^{\mathrm{P}}. \tag{27}$$

Memory starts at zero and follows Equation 23, substituting $\boldsymbol{\delta}_j^{\mathrm{aux}}$ for $\boldsymbol{\delta}_j^{\mathrm{A}}$. The first phrase therefore uses the raw trajectory and initializes memory with the full auxiliary displacement; propagation can first affect the second phrase. Each control computes its context and residuals from its own rollout. The propagation-only control keeps α_M at its fixed value for residual calculation: setting all $\alpha_m = 1$ would eliminate the residual and keep the initially zero memory at zero.

Causality and perturbations. Given fixed external controls, each BHC update uses only the prefix, fixed parameters, and its own previous states. Induction over phrases therefore excludes future targets and teacher plans from BHC; commanded actions remain identical across PP/BHC/TT even as generated histories diverge. This establishes that the supplied action schedule is unchanged, while the counterfactual table measures the separate change in its realized response.

For the controlled mismatch process used in the main-text stress test, we perturb the raw slow trajectory by a persistent mismatch:

$$\mathbf{V}_\tau^{\text{stress}} = \mathbf{V}_\tau^{\text{raw}} + \mathbf{b}_\tau, \qquad \mathbf{b}_{\tau+1} = 0.9\mathbf{b}_\tau + \sigma\boldsymbol{\epsilon}_\tau, \quad \boldsymbol{\epsilon}_\tau \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{28}$$

Here τ indexes phrase boundaries, σ controls mismatch strength, and the same paired perturbations are applied to the compared rollouts. For this process, $\gamma = 0.9$ and independent standard Gaussian innovations give

$$\mathbf{b}_k = \gamma^k \mathbf{b}_0 + \sigma \sum_{i=0}^{k-1} \gamma^{k-1-i} \boldsymbol{\epsilon}_i, \qquad \mathrm{Cov}(\mathbf{b}_k \mid \mathbf{b}_0) = \sigma^2 \frac{1-\gamma^{2k}}{1-\gamma^2} I. \tag{29}$$

Thus σ is innovation SD, with limiting coordinate SD $\sigma/\sqrt{1-\gamma^2}$. At fixed context, adding bias $\mathbf{b}$ changes $\mathbf{A}_{j,m}$ by $\alpha_m \mathbf{b}$ and $\boldsymbol{\delta}_j^{\mathrm{A}}$ by $-(1-\alpha_M)\mathbf{b}$.

D Metrics and Evaluation Protocols

Prediction/reference share the canonical grid and roles. Predicted onset/sustain masks use ≥ 0.5 and reference masks use > 0.5; activity A is their logical OR. Velocity is excluded from the seven decoded-space metrics. RBDE, Rhythm JSD, Chroma JSD, RSE, and TSSE are lower-is-better; SC and GC are higher-is-better. The metrics are reported independently, without a composite score. For display in the main cross-model table, we use the reporting units $\widetilde{m} = 10m$ for Chroma JSD, SC, Rhythm JSD, GC, RSE, and TSSE, while RBDE is reported as $\widetilde{m} = m$. All metric definitions, aggregation, bootstrap contrasts, and ranking decisions use the raw metric m; the fixed factor only changes the displayed units. Let B be the number of complete bars, $S = 16$, and $\epsilon = 10^{-8}$.

D.1 Decoded-space metrics

For all histogram comparisons, use base-2 $\mathrm{JSD}(p, q) = \frac{1}{2}\mathrm{KL}_2(p\|m) + \frac{1}{2}\mathrm{KL}_2(q\|m)$ with $m = (p+q)/2$. If both histograms are empty the distance is zero; if only one is empty it is one. Otherwise add ϵ to raw counts before unit-mass normalization.

RBDE and Rhythm JSD. Role density is $d_r = B^{-1}\sum_{b,s,p} A_{r,b,s,p}$, giving

$$\mathrm{RBDE} = \frac{1}{6}\sum_{r=1}^{6} \left| \log \frac{\hat{d}_r + \epsilon}{d_r + \epsilon} \right|. \tag{30}$$

All roles, including silent roles, remain in this unclipped mean. Rhythm JSD averages six role-wise JSDs of 16-bin onset histograms $h_{r,k} = \sum_{b,p} O_{r,b,k,p}$.

Chroma JSD and RSE. Chroma JSD averages aligned-bar JSDs of 12-bin pitch-class activity histograms pooled over time and the five pitched roles, excluding drums; valid-bar counts are retained. For RSE, each of six roles has a 128-bin activity-weighted pitch histogram $P_{r,p} = \sum_t A_{r,t,p}$. Their pairwise JSDs form a symmetric matrix M with zero diagonal:

$$\mathrm{RSE} = \frac{\|\hat{M} - M\|_F}{\|M\|_F + \epsilon}. \tag{31}$$

No separate role-activity relationship is added; drum keys retain their role-specific identity rather than representing harmonic register.

Scale consistency (SC). SC is computed directly from non-drum pitched note events in the decoded eight-bar window. Let c_p count events with pitch class $p \in \{0, \ldots, 11\}$, and let $\mathcal{S}$ contain the 12 major and 12 natural-minor pitch-class scales. We report the largest fraction of events covered by one candidate scale:

$$\mathrm{SC} = \max_{S \in \mathcal{S}} \frac{\sum_{p \in S} c_p}{\sum_{p=0}^{11} c_p}. \tag{32}$$

Drum events and pitch metadata are ignored. If the window contains no non-drum pitched event, SC is undefined for that window only.

Groove consistency (GC). All true note onsets, including drum onsets, are quantized on a four-step- per-beat grid, giving 16 steps per $4/4$ bar. Let $g_{b,s} \in \{0, 1\}$ indicate whether any track starts a note at bar b and step s in the eight-bar window. After flattening the 8×16 binary onset matrix in temporal order, GC is one minus the mean adjacent binary Hamming change:

$$\mathrm{GC} = 1 - \frac{1}{8S - 1} \sum_{i=1}^{8S-1} \mathbf{1}[g_i \neq g_{i-1}]. \tag{33}$$

The window extent is fixed to eight complete bars; carry-in notes remain active at the left boundary but do not create an artificial onset.

TSSE. Each bar concatenates 12 chroma, 16 rhythm, six role-onset, six role-activity, and one mean-polyphony feature (41 dimensions). Each block is independently L1-normalized; zero blocks remain zero. Polyphony counts active role–pitch cells across the bar, divided by S. The cosine self-similarity matrix Q sets zero/zero similarity and its diagonal to one, and zero/nonzero similarity to zero. For strict upper-triangle indices $\mathcal{U}$,

$$\mathrm{TSSE} = \frac{\|(\hat{Q} - Q)_{\mathcal{U}}\|_2}{\|Q_{\mathcal{U}}\|_2 + \epsilon}. \tag{34}$$

Reference norms are not clipped. With fewer than two bars, TSSE is undefined and excluded only from its own mean.

D.2 F1 AND DIAGNOSTIC DISTANCES

Activity precision, recall, and F1 use canonical-cell counts: $P = TP/(TP+FP)$, $R = TP/(TP+FN)$, and $F1 = 2TP/(2TP + FP + FN)$; empty denominators yield one. Pooled F1 combines counts before evaluation; macro F1 averages values at the stated sample/window/song level. Latent and decoder-logit distances to TT are coordinatewise RMS errors, averaged over scored steps within a phrase or horizon. TT never enters PP/BHC histories.

D.3 CROSS-MODEL TEST DATASET AND AGGREGATION

The cross-model comparison uses the shared 102-song test dataset in Table 5, with matched song IDs, aligned references, and a native eight-bar continuation target. BEAT, Music2Music, MuPT, NotaGen, Amadeus, and Text2MIDI use publicly released checkpoints. For repositories with a single checkpoint, we use that release; for those with multiple checkpoints, we use the version associated with the published paper. Each baseline retains its repository's original inference code and sampling settings, without modification or Slakh2100 retraining or fine-tuning. All methods receive the same raw eight-bar multi-track MIDI prefix; the prefix is converted with each repository's released preprocessing code into its native representation. Text-capable methods receive the empty-text condition used by their released prefix adapter, while methods without a text input receive no text. After generation, every output is converted back to MIDI and processed by one shared Python post-processing utility before scoring; no manual note editing, padding, or score-based retry is applied. Native sampling budgets are retained, so computation budgets are not equalized. MWM is PP; MWM+BHC adds correction with the same weights. TT appears only in diagnostics, not the generation or listening tables. For this cross-model table, every action input to MWM and MWM+BHC is the all-zero vector. Both models use segment-level control descriptors predicted by the planner from the shared observed prefix; no manually authored action or control-descriptor schedule is supplied.

Each decoded continuation is cropped to eight complete $4/4$ bars using the same offset for prediction and reference. Every method is evaluated from the same 102-song test dataset under the shared scoring protocol. The cross-model ranking uses each method's valid decoded outputs from this shared test dataset. For each metric, valid seeds are averaged within the exact matched window, windows are averaged within song, and songs receive equal weight. An undefined value is excluded only from that metric's mean. The native h8 table is the primary cross-model result; the centered h32 crop is a secondary diagnostic. Event sample counts are averaged within windows, summed within songs, then pooled for precision/recall/F1. Paired scalar contrasts bootstrap song-mean differences; event contrasts resample paired song counts and recompute pooled F1. Cross-model paired contrasts

use shared song draws and 10,000 song-bootstrap replicates with seed 43. The trajectory diagnostics instead use song-macro F1 and 10,000 replicates as specified in Appendix C.1.

D.4 Counterfactual Control Metrics

We score counterfactuals from the decoded continuation using the same descriptor extractor as above. Commanded controls and auxiliary predicted controls are never used as measurements. Let i index a decision point, m a method, $k \in \{1, 2, 3, 4\}$ the generated phrase, and j the edited coordinate. Let $c_{0,i,j}$ be the mean of the four member controls in the last complete observed prefix phrase, $t_{i,j}$ the commanded target, and $s_j > 0$ the standard deviation of coordinate j computed on the training split. The paired zero-action branch uses the same prefix, method, decoder, sampling seed, and non-edited commanded coordinates while holding the edited coordinate at $c_{0,i,j}$. From the decoded outputs define

$$y_{m,i,k,j} = f_j(\widehat{X}_{m,i,k}), \qquad z_{m,i,k,j} = f_j(\widehat{X}^{\,0}_{m,i,k}), \qquad r_{m,i,k,j} = y_{m,i,k,j} - z_{m,i,k,j}, \tag{35}$$

where f_j is the phrase-level mean descriptor and $\Delta_{i,j} = t_{i,j} - c_{0,i,j}$ is the requested change. A condition is eligible before generation only when its command lies in the fixed support, the edited coordinate is nonzero, and the required prefix/context is available; eligibility is never inferred from the generated result.

For a continuous edit, the first-phrase response metrics are

$$\mathrm{Dir}_{m,i,j} = \mathbf{1}\{\Delta_{i,j} r_{m,i,1,j} > 0\}, \tag{36}$$

$$\mathrm{TargetErr}_{m,i,j} = \frac{|y_{m,i,1,j} - t_{i,j}|}{s_j}, \tag{37}$$

$$\mathrm{Success}_{m,i,j} = \mathrm{Dir}_{m,i,j}\ \mathbf{1}\{\mathrm{TargetErr}_{m,i,j} \le 0.25\}. \tag{38}$$

Zero response and reverse response therefore count as incorrect direction. Late Err. uses the same Target Err. formula and the same eligible command set with $k = 4$. For the non-target coordinates, let $\mathcal{J}_i$ be the continuous coordinates other than j and let $\mathcal{R}_i$ be the six role-activity coordinates. We report

$$\mathrm{OffTarget}_{m,i,k,j} = \frac{1}{2}\left[\frac{1}{|\mathcal{J}_i|}\sum_{q\in\mathcal{J}_i}\frac{|y_{m,i,k,q} - z_{m,i,k,q}|}{s_q} + \frac{1}{|\mathcal{R}_i|}\sum_{q\in\mathcal{R}_i}\mathbf{1}\{y_{m,i,k,q} \neq z_{m,i,k,q}\}\right]. \tag{39}$$

The two groups receive equal weight; a group with no independently readable coordinate is reported as N/A rather than silently changing the denominator.

For a five-level dose response, use $\mathcal{L} = \{-1, -0.5, 0, 0.5, 1\}$ and let $v_{m,i,k}(\ell)$ be the measured target coordinate at level ℓ, after division by s_j. Define the response relative to the zero level by

$$q_{m,i,k}(\ell) = v_{m,i,k}(\ell) - v_{m,i,k}(0). \tag{40}$$

The dose statistics are

$$\rho_{0,m,i,k} = \mathrm{Corr}\big(\mathrm{rank}(\ell), \mathrm{rank}(v_{m,i,k}(\ell))\big), \tag{41}$$

$$\mathrm{OrderAcc}_{m,i,k} = \frac{1}{\binom{5}{2}}\sum_{a<b}\mathbf{1}\{v_{m,i,k}(\ell_b) > v_{m,i,k}(\ell_a)\}, \tag{42}$$

$$\mathrm{Gain}_{m,i,k} = \frac{\sum_{\ell\in\mathcal{L}}\ell\, q_{m,i,k}(\ell)}{\sum_{\ell\in\mathcal{L}}\ell^2}. \tag{43}$$

Ranks use average ranks for ties. If all five realized values are equal, ρ_0 is defined as zero and the condition is retained in the dose denominator. Equal values in an ordering pair do not count as correctly ordered, so the Order Acc. denominator is ten. Gain$\to 1$ equals one when the realized normalized response is exactly $q(\ell) = \ell$; it is the zero-intercept least-squares slope and is not clipped. The density-response Spearman statistic ρ_0 in Table 3 uses this rank-correlation definition before the stated song-level aggregation.

Reversal response. For each eligible nonzero edit $\Delta_{i,j} = t_{i,j} - c_{0,i,j}$, the reversal schedule holds $t_{i,j}$ for phrases 1 and 2 and restores $c_{0,i,j}$ for phrases 3 and 4. The commanded change at the reversal boundary is

$$\Delta^{\mathrm{rev}}_{i,j} = c_{0,i,j} - t_{i,j} = -\Delta_{i,j}. \tag{44}$$

Let $y^{\mathrm{rev}}_{m,i,k,j}$ denote the descriptor measured from the decoded output under this schedule. The realized reversal response is the change from phrase 2 to phrase 3 within the same rollout:

$$r^{\mathrm{rev}}_{m,i,j} = y^{\mathrm{rev}}_{m,i,3,j} - y^{\mathrm{rev}}_{m,i,2,j}. \tag{45}$$

The reversal panel of Table 3 reports

$$\mathrm{Dir}^{\mathrm{rev}}_{m,i,j} = \mathbf{1}\{\Delta^{\mathrm{rev}}_{i,j} r^{\mathrm{rev}}_{m,i,j} > 0\}, \tag{46}$$

$$\mathrm{TargetErr}^{\mathrm{rev}}_{m,i,j} = \frac{|y^{\mathrm{rev}}_{m,i,3,j} - c_{0,i,j}|}{s_j}, \tag{47}$$

$$\mathrm{Success}^{\mathrm{rev}}_{m,i,j} = \mathrm{Dir}^{\mathrm{rev}}_{m,i,j}\ \mathbf{1}\{\mathrm{TargetErr}^{\mathrm{rev}}_{m,i,j} \le 0.25\}. \tag{48}$$

Thus direction uses the change between the pre-reversal and restored commands, while target error uses the restored absolute target. Zero realized change counts as an incorrect direction. Late Err. evaluates $|y^{\mathrm{rev}}_{m,i,4,j} - c_{0,i,j}|/s_j$. Off-target uses Equation 39 at $k = 3$, comparing the reversal output with the paired zero-action output on the non-edited coordinates. Eligibility is determined by the initial nonzero edit, command support, and available context before generation; restoring the original target does not remove the condition from the reversal denominator.

All metrics retain the predeclared eligible-attempt set. Complete decoded outputs are required for continuous errors and off-target values; a failed or incomplete generation remains in the evaluation and contributes zero to Dir. and Success, while continuous metrics are reported on the common complete-output set. No failed condition is removed because it has an unfavorable response. For each metric, executor seeds are averaged within a $(\text{song}, \text{decision point}, \text{command})$ cell, commands are averaged within a decision point, decision points are averaged within a song, and songs receive equal weight. Dose statistics are computed within each seed and decision point before applying this same aggregation. Confidence intervals resample songs with replacement, keep all of a sampled song's decision points, commands, seeds, and phrases, recompute the complete statistic, and use the 2.5th and 97.5th percentiles.

E Supplementary Experiments

E.1 Seven-Metric Architecture Ablation

We evaluate five retrained configurations under a matched architecture-ablation protocol: Full MWM, No-plan, Open-loop, No-action, and Executor-only. All configurations use the same frozen codec, 102-song test dataset, continuation windows, command schedules, training data, 3,500-update budget, and training seeds $\{7, 23, 29\}$; BHC is disabled. No-plan masks all plan-derived paths, Open-loop holds the initial plan fixed, No-action masks the action/control paths, and Executor-only removes the phrase planner and slow adapter while retaining the segment-level action-conditioned executor. The first panel scores the native first eight generated bars. The second scores bars 12–20 of the first 32 generated bars, providing a centered long-horizon readout. Metrics are computed on the decoded canonical grid, averaged over valid seed and window observations within song, and then averaged equally over songs.

The paired comparisons use 10,000 song-bootstrap replicates over the same 102 songs. At the native horizon, No-plan and Open-loop are close to Full, while No-action has the largest RBDE and rhythm error. Executor-only has the lowest RBDE, Chroma JSD, Rhythm JSD, and RSE in this fixed-budget setting. In the centered long-horizon window, Full has lower mean RBDE (1.204117 versus 1.218219) and RSE (0.216744 versus 0.217032) than Open-loop. These two comparisons show improved mean density and role-structure matching when the planner refreshes its plan at phrase boundaries. Across all five configurations, Executor-only has the lowest target-relative errors, while No-plan also improves on Full in RBDE, Chroma JSD, Rhythm JSD, and RSE. No-action has the largest RBDE, Rhythm JSD, RSE, and TSSE in this window.

This table reports control-conditioned prediction on the canonical grid. Cross-model generation scores use autonomous MIDI outputs under the protocol in Appendix D.3. The No-plan configuration removes plan-derived paths and uses a smaller trainable parameter subset.

Table 9: Main results of the matched architecture ablation. All rows use 102 matched songs. RSE denotes role-structure error. RBDE, Chroma JSD, Rhythm JSD, RSE, and TSSE are lower-is-better; SC and GC are higher-is-better.

Configuration	RBDE↓	Chroma JSD↓	SC↑	Rhythm JSD↓	GC↑	RSE↓	TSSE↓
Native first 8 bars							
Full	0.567405	0.322946	0.984719	0.138604	0.865196	0.157474	0.089307
No-plan	0.547049	0.321320	0.984526	0.137952	0.865109	0.155908	0.089069
Open-loop	0.544286	0.322410	0.984844	0.138379	0.865021	0.156361	0.089274
No-action	0.607198	0.320738	0.985984	0.140303	0.867851	0.159415	0.089620
Executor-only	0.489447	0.317519	0.984654	0.136735	0.863008	0.153186	0.089118
Centered 8 bars, bars 12–20 of the first 32 bars							
Full	1.204117	0.341424	0.981196	0.218930	0.860673	0.216744	0.082659
No-plan	1.148031	0.337998	0.980296	0.215627	0.861257	0.213832	0.082758
Open-loop	1.218219	0.339469	0.981347	0.218659	0.862307	0.217032	0.082413
No-action	1.316638	0.340839	0.983708	0.221429	0.863095	0.225836	0.083001
Executor-only	1.119229	0.333284	0.981329	0.211492	0.867122	0.212931	0.082380

E.2 Subjective Listening Evaluation

Twenty-nine listeners, including five with arranging experience, rated all five matched excerpts from each of eight methods: 40 excerpts, 145 submissions per method, 1,160 submissions, and 5,800 dimension scores. FluidSynth rendered MIDI to 44,100-Hz, 16-bit PCM audio using `Normalize`; the interface's `Common Prefix` option was disabled. Method labels were hidden, order randomized, and replay allowed.

All dimensions use integer scores 1–5; 2–4 are intermediate judgments between the anchors in Table 10. Records retain anonymous listener/group, excerpt/method, presentation order, and five scores. All listener–excerpt–method records are complete, and every listener's order contains all 40 conditions once.

Table 10: Listening-score anchors.

Dimension	Score 1	Score 5
Overall quality	Very poor overall listening quality.	Excellent overall listening quality.
Musicality	Rhythm, melody, and harmony sound unnatural or clash.	Rhythm, melody, and harmony work together naturally and expressively.
Local coherence	Neighboring events and short transitions sound disconnected or abrupt.	Neighboring events and short transitions form coherent musical passages.
Long-horizon continuity	Musical development breaks down or becomes disconnected across the excerpt.	Musical development remains coherent and flows naturally across the excerpt.
Structural consistency	Motifs, phrases, and instrumental roles lack clear organization.	Motifs, phrases, and instrumental roles form a clear, consistent organization.

Method means average listeners within excerpt, then the five excerpts, equivalently all 145 ratings. Main-table SDs use the 145 individual scores with denominator 144. Paired differences are computed within listener/excerpt; listeners have equal weight irrespective of experience. For each of 10,000 crossed bootstrap replicates, independently resample 29 listener IDs and five excerpt IDs, use the same draws for all methods, and average their Cartesian product. Percentiles 2.5 and 97.5 give method-mean and paired-difference CIs (Tables 11–12). All five BHC-minus-MWM intervals include zero. Results describe these five matched excerpts, with confidence intervals reported without multiplicity adjustment.

Table 11: 95% confidence intervals for mean listening scores from the paired crossed listener/excerpt bootstrap. All dimensions use a 1–5 scale.

Method	Overall	Musicality	Coherence	Continuity	Structure
BEAT	$[3.428, 3.800]$	$[3.476, 3.938]$	$[3.414, 3.855]$	$[3.352, 3.731]$	$[3.366, 3.779]$
Music2Music	$[3.345, 3.800]$	$[3.497, 3.876]$	$[3.345, 3.766]$	$[3.338, 3.841]$	$[3.297, 3.766]$
MuPT	$[3.607, 3.972]$	$[3.683, 4.014]$	$[3.662, 3.952]$	$[3.572, 3.855]$	$[3.524, 3.848]$
NotaGen	$[3.772, 4.090]$	$[3.883, 4.228]$	$[3.759, 4.076]$	$[3.717, 4.028]$	$[3.793, 4.083]$
Amadeus	$[3.717, 4.200]$	$[3.786, 4.172]$	$[3.662, 4.103]$	$[3.821, 4.255]$	$[3.903, 4.317]$
Text2MIDI	$[3.572, 3.897]$	$[3.566, 4.007]$	$[3.600, 3.917]$	$[3.483, 3.883]$	$[3.524, 3.890]$
MWM	$[3.834, 4.241]$	$[3.807, 4.234]$	$[3.897, 4.283]$	$[3.752, 4.228]$	$[3.828, 4.207]$
MWM + BHC	$[3.814, 4.310]$	$[3.724, 4.241]$	$[3.765, 4.276]$	$[3.841, 4.310]$	$[3.876, 4.283]$

Table 12: Paired listening-score differences, MWM+BHC minus MWM. Positive differences favor BHC; all five 95% intervals include zero.

Dimension	Mean difference	95% confidence interval
Overall	$+0.041$	$[-0.152, +0.234]$
Musicality	-0.021	$[-0.304, +0.255]$
Coherence	-0.041	$[-0.269, +0.179]$
Continuity	$+0.097$	$[-0.103, +0.297]$
Structure	$+0.062$	$[-0.152, +0.283]$

F Evaluation Scope

This study evaluates MWM on the automatically annotated Slakh2100 Redux corpus. The internal continuation protocol uses a 48-bar observed prefix; the eligible test datasets contain 102 songs at 32 bars and 44 songs at 64 bars.

The counterfactual study evaluates density control through the eight-dimensional action interface. Five command levels characterize response direction, ordering, and magnitude. First-phrase and fourth-phrase measurements describe the response at the initial and later rollout stages.

AI use statement

We used generative artificial intelligence (AI) tools to assist with language editing, grammar correction, and improvements to the clarity and readability of the manuscript, as well as code development, debugging, and implementation refinement. In addition, the system includes a large language model interface that translates natural-language instructions into structured musical control actions, as described in the methodology and Appendix A.2. Musical states and note sequences are generated by MWM through its learned dynamics and decoder. The authors reviewed and verified AI-assisted text and code and retained responsibility for the research design, experimental procedures, interpretation of results, and conclusions. We take full responsibility for the final content of this work, including all AI-assisted material.

Ethics statement

This work investigates controllable symbolic music generation using Slakh-Action, constructed from the publicly available Slakh2100 Redux dataset. Third-party datasets, software, and pretrained models remain subject to their respective licenses and terms of use. Our subjective listening evaluation involved 29 participants, from whom informed consent was obtained. Ratings were recorded using anonymous participant identifiers, with method identities hidden and presentation order randomized. We report aggregate statistics without disclosing participant identities. We recognize potential concerns related to music generation, including memorization of existing works, respect for creators' rights, and limitations in the stylistic coverage of training data. Our evaluation is restricted to the dataset and experimental settings described in this paper and does not establish generalization across musical styles or populations.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the implementation of MWM and Boundary-Aware History Correction (BHC) in an anonymous repository: `https://anonymous.4open.science/r/anonymous-repository-8B30/`. The main paper and Appendices A–F describe the construction of Slakh-Action, song-level data partitions, symbolic representations, action annotations, model architecture, training objectives, optimization settings, and parameter-selection procedures. They also specify the recursive rollout protocols, BHC updates and coefficients, baseline configurations, evaluation metrics, aggregation procedures, and statistical analyses. The supplementary material details the counterfactual control experiments, long-horizon continuation evaluations, ablation studies, and subjective listening protocol. These descriptions document the experimental settings and assumptions needed to reproduce and interpret the reported results.