# CAVC: CONVERGENCE-ADAPTIVE AUDIO-VISUAL CACHE FOR MODALITY-ASYMMETRIC DIFFUSION GENERATION

**Anonymous authors**
Paper under double-blind review

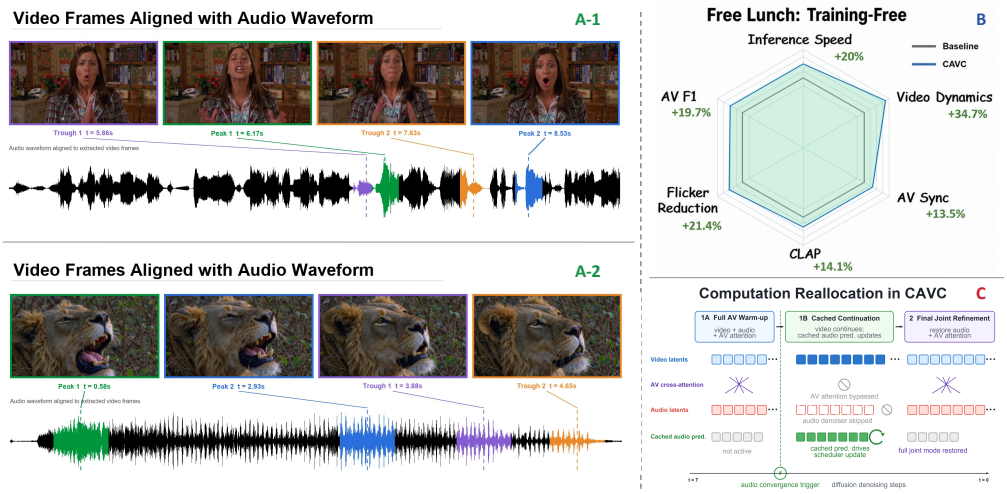


Figure 1: CAVC overview: synchronized generation examples (A), selected positive General-80 changes (B), and the branch schedule (C). Panel B is a selected-metric summary; the complete profile also includes the Naturalness decrease from 0.6349 to 0.490.

## ABSTRACT

Diffusion-based audio-visual generation methods typically perform joint denoising of audio and video latents throughout the entire sampling trajectory, repeatedly executing modality-specific denoising branches and cross-modal attention at every timestep. However, we observe that such uniform joint computation is not always necessary. The key is a modality-asymmetric recoverability: audio-side drift introduced by mid-stage caching can be substantially reduced by only a few terminal joint denoising steps, whereas video-side degradation caused by caching leads to persistent loss of motion and temporal details. This asymmetry suggests that inference acceleration should selectively cache recoverable audio-side computation while preserving full video refinement. Motivated by this observation, we propose **Convergence-Adaptive Audio-Visual Cache (CAVC)**, a training-free modality-asymmetric caching framework. CAVC adopts a sandwich-structured computation allocation: after detecting that audio denoiser outputs enter a cache-tolerant regime, it caches and reuses the audio denoiser output to bypass audio denoising and cross-modal attention in the middle stage, while the video branch continues independent spatio-temporal refinement; a few joint steps are then restored at the end to reduce accumulated cross-modal drift and improve synchronization. Experiments across multiple public benchmarks demonstrate that CAVC achieves 1.10–1.35× wall-clock acceleration, improves several visual dynamics and temporal statistics, and maintains competitive audio fidelity and audio-visual alignment.

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

# 1 INTRODUCTION

Text-to-audio-visual generation aims to synthesize semantically consistent video and temporally synchronized audio from text, becoming a pivotal task in multimodal generative modeling Singer et al. (2022); Ho et al. (2022); Blattmann et al. (2023); Huang et al. (2023); Zhang et al. (2024); He et al. (2025). While unimodal diffusion models excel independently Guo et al. (2023); Chen et al. (2023); Liu et al. (2024b), their temporal and semantic alignment requires joint denoising via audio-video cross-modal attention Ruan et al. (2023); Tang et al. (2023; 2024); Xing et al. (2024); Luo et al. (2023b). Despite its effectiveness, this paradigm introduces prohibitive inference costs by repeatedly executing modality-specific branches and dense cross-attention at every sampling step, creating a major bottleneck for large-scale deployment.

A straightforward remedy is to adopt existing diffusion acceleration techniques, including advanced ODE/SDE solvers Song et al. (2020); Lu et al. (2022); Zhao et al. (2023), knowledge distillation for few-step generation Salimans & Ho (2022); Luo et al. (2023a); Meng et al. (2023); Yin et al. (2024), and training-free feature caching across adjacent timesteps Ma et al. (2024); Bolya & Hoffman (2023); Wimbauer et al. (2024). However, directly applying these strategies to audio-visual generation is not always satisfactory. Reducing the entire sampling trajectory compresses refinement of both modalities, often harming video details and temporal dynamics. Meanwhile, modality-agnostic caching Ma et al. (2024) treats audio and video computation uniformly, ignoring their potentially different roles throughout denoising. These limitations raise a fundamental question: *is full audio-video joint computation necessary at every denoising step?*

We answer this question through the lens of a modality-asymmetric recoverability in joint audio-visual denoising. Importantly, we do not assume that audio predictions numerically converge faster than video predictions. Instead, CAVC exploits a recoverability asymmetry: audio-side drift induced by mid-stage caching can be substantially reduced by a few terminal joint steps, whereas video-side degradation leads to persistent loss of motion and temporal details that brief terminal recoupling cannot recover. In the evaluated backbone, this asymmetry is supported by our branch-freezing ablations, where caching or freezing the video branch yields high raw speedups but severely degrades video dynamics, while audio-only caching preserves video refinement and benefits from joint tuning.

Motivated by this observation, we propose **Convergence-Adaptive Audio-Visual Cache** (**CAVC**), a training-free modality-asymmetric caching framework with a sandwich-structured computation allocation. Once the audio denoiser output enters a cache-tolerant regime, CAVC caches and reuses the audio-side network output to bypass the audio denoising branch in subsequent middle steps; the audio latent is still updated by the original scheduler using the cached output, so its sampling trajectory continues without repeated audio forward passes. Simultaneously, CAVC bypasses audio-video cross-modal attention, allowing the video branch to proceed with independent spatio-temporal refinement conditioned on its visual state and textual guidance. Finally, CAVC restores full joint denoising for a few terminal steps. Empirically, this short recoupling reduces cross-modal drift and improves audio-visual alignment before decoding.

We also provide a conditional perturbation analysis of cached-audio and video-only residuals. Under local smoothness assumptions, the bounds expose how these residuals and cache duration affect deviation from the full sampler; recoverability is evaluated empirically. On AVGen-Bench, VABench T2AV, and General-80, CAVC achieves 1.10–1.35$\times$ speedups while improving visual dynamics and retaining competitive audio fidelity and audio-visual alignment, with metric-dependent trade-offs. Our main contributions are summarized as follows:

- **Modality-asymmetric recoverability.** We identify that audio-side drift introduced by mid-stage caching can be substantially reduced by a few terminal joint denoising steps, whereas video-side degradation is difficult to recover, motivating asymmetric computation allocation.
- **Sandwich-structured asymmetric caching.** We propose CAVC, a training-free framework that bypasses recoverable audio-side computation and cross-modal attention in the middle stage while preserving full video refinement, then restores minimal joint denoising at the end for synchronization refinement.
- **Favorable quality-efficiency trade-off.** Across three benchmarks, CAVC achieves 1.10–1.35$\times$ acceleration, improves several motion and temporal metrics, and maintains competitive audio fidelity and audio-visual alignment, supported by ablations and perturbation analysis.

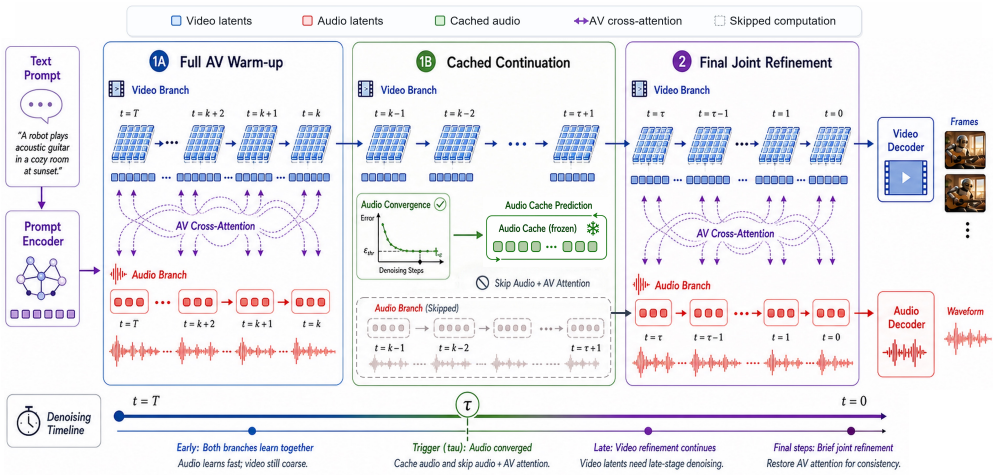


Figure 2: CAVC follows joint warm-up (1A), cached audio prediction with continued video computation (1B), and terminal joint refinement (2). The panel timelines denote the trigger by $k$ and the refinement boundary by $\tau$; the equations use $\tau$ and $t_{\text{ref}}$, respectively. The convergence inset denotes the adjacent-change trigger. Stage 2 retains native terminal joint computation.

## 2 Related Work

### 2.1 Audio-Visual Diffusion Generation

Multimodal generative modeling has advanced from independent unimodal video (Singer et al., 2022; Guo et al., 2023) and audio synthesis (Liu et al., 2023; 2024b) to audio-visual joint generation. Representative frameworks such as MM-Diffusion (Ruan et al., 2023) and CoDi (Tang et al., 2023) achieve this by passing audio and video latents through dual denoising branches fused via dense cross-modal attention. Recent advancements (Xing et al., 2024; Wang et al., 2026; Sun et al., 2025) employ latent aligners to enhance fine-grained synchronization. However, this tightly coupled paradigm mandates repeated execution of modality-specific branches and expensive cross-modal interactions (Zhao et al., 2024) at every denoising step, causing prohibitive computational overhead. Our work addresses this bottleneck by dynamically reallocating computation during inference, bypassing the need for architectural redesigns.

### 2.2 Acceleration of Diffusion Models

Accelerating diffusion inference predominantly reduces sampling steps via advanced solvers (Song et al., 2020; Lu et al., 2022; Zhao et al., 2023; Esser et al., 2024) or distillation (Luo et al., 2023a; Yin et al., 2024; Meng et al., 2023; Sauer et al., 2024). Complementary training-free approaches reduce per-step costs via token merging (Bolya et al., 2023; Wang et al., 2024) or structured pruning (Fang et al., 2023). Step reduction also changes the video refinement trajectory. CAVC instead preserves its number of updates and approximates selected branch evaluations within them.

### 2.3 Selective Feature Caching

DeepCache reuses high-level features, and TeaCache adapts reuse to changes in model representations (Ma et al., 2024; Liu et al., 2024a). ToCa selects tokens and layers, AdaCache adapts schedules to individual video generations, and PAB assigns different broadcast intervals to attention types (Zou et al., 2025; Kahatapitiya et al., 2025; Zhao et al., 2025). FasterCache also exploits temporal redundancy in video diffusion (Lyu et al., 2025). These methods already establish the value of heterogeneous reuse. CAVC studies a complementary allocation unit: an entire audio branch and its cross-modal interactions, while retaining every video evaluation and native terminal joint refinement. An audio-only TeaCache implementation with the same terminal phase would be a close controlled comparator; the reported experiments do not isolate that comparison.

# 3 METHOD

We propose CAVC (Figure 2), a training-free acceleration framework for asymmetric dual-stream audio-visual diffusion transformers, selectively approximating recoverable audio computation while keeping the video branch active throughout the native denoising schedule.

## 3.1 ASYMMETRIC PROBLEM FORMULATION

We consider a standard text-conditioned audio-visual diffusion model. Given text condition $c$, let $v_t$ and $a_t$ denote the video and audio latents at denoising step $t \in \{T, T-1, \dots, 1\}$ with noise level $\sigma_t$. Larger $t$ corresponds to higher noise, and denoising proceeds from $t = T$ to the final state at $t = 0$. For empirical trigger locations, we use the zero-based forward execution index $k = T - t$, which increases as $t$ decreases. The full joint denoising model simultaneously predicts denoised outputs for both modalities:

$$\hat{y}_v^t = f_v(v_t, a_t, c, \sigma_t), \qquad \hat{y}_a^t = f_a(v_t, a_t, c, \sigma_t), \tag{1}$$

where $f_v$ and $f_a$ denote the video and audio branches, mutually coupled through audio-to-video (A2V) and video-to-audio (V2A) cross-modal attention. The scheduler subsequently updates both latents:

$$v_{t-1} = S_t^v\big(v_t,\, \hat{y}_v^t\big)\,, \qquad a_{t-1} = S_t^a\big(a_t,\, \hat{y}_a^t\big)\,. \tag{2}$$

The branch ablations motivate asymmetric allocation in the evaluated configuration. We preserve video evaluations and approximate audio predictions during a middle interval. This policy retains video refinement, with metric-dependent trade-offs in audio, alignment, and motion.

**Modality-level allocation.** CAVC reuses the audio prediction, not the audio state. The scheduler still advances both latents at every step, while $g_v$ retains video self-/temporal attention and text conditioning. Joint warm-up establishes the initial interaction; the trigger selects a prompt-dependent transition; terminal joint updates act on the trajectory reached after approximation. This preserves the native state-update interface while reducing repeated neural evaluations. The terminal ablation measures recovery empirically, and the analysis below characterizes how approximation residuals propagate through the retained downstream joint updates.

## 3.2 CONVERGENCE-ADAPTIVE AUDIO-VISUAL CACHE

Driven by the recoverability asymmetry, CAVC adopts a sandwich-structured computation allocation with three phases: a full joint warm-up to establish cross-modal binding, an asymmetric cached continuation phase that bypasses recoverable audio-side computation while video continues full refinement, and a brief final joint refinement to reduce cross-modal drift.

**Stage 1A: Full Audio-Visual Warm-up.** For $T \geq t \geq \tau$, CAVC follows the full sampler above, establishing scene layout, audio categories, and cross-modal binding. Premature decoupling can introduce drift that terminal refinement does not reliably remove.

**Stage 1 (Trigger): Online Audio Convergence Detection.** Because prediction changes vary across prompts, a fixed trigger need not match every trajectory. CAVC thus employs adaptive online monitoring; for $t < T$, it computes the mean absolute change from the previously executed, higher-noise prediction at $t + 1$:

$$\Delta_a^t = \mathrm{mean}\big(\big|\hat{y}_a^t - \hat{y}_a^{t+1}\big|\big)\,. \tag{3}$$

The mean is over all elements of the audio prediction tensor for one sample; it is an absolute, unnormalized change. Caching triggers at timestep $\tau$ when $\Delta_a^t < \theta$ persists for $W$ execution steps. This threshold is an empirical proxy for a cache-tolerant regime (Figure 3, Table 3d), after which CAVC stores $\hat{y}_a^* = \hat{y}_a^\tau$.

**Stage 1B: Video-Only Continuation with Cached Audio.** Let $t_{\mathrm{ref}}$ denote the first timestep of terminal joint refinement, with $t_{\mathrm{ref}} = 3$ in our default setting. During the cached interval $\mathcal{I} = \{\tau - 1, \tau - 2, \dots, t_{\mathrm{ref}} + 1\}$, CAVC simultaneously bypasses the audio denoiser and AV cross-modal attention. The audio latent continues its trajectory using the cached audio denoiser output:

$$\hat{y}_a^t := \hat{y}_a^*, \qquad \tilde{a}_{t-1} = S_t^a(\tilde{a}_t,\, \hat{y}_a^*)\,. \tag{4}$$

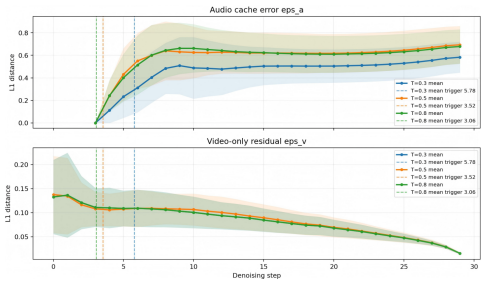


Figure 3: Empirical audio-prediction and video-only residual curves over execution index $k$. Dashed markers indicate trigger locations. Threshold labels refer to $\theta$, not the diffusion horizon $T$. The two curves illustrate the distinct approximation sources along the displayed trajectories.

The residual curves connect the allocation policy to its two approximation sources. Audio-prediction variation concerns reuse within one branch; the video-only residual concerns the cross-modal interactions omitted from the other branch. Their roles differ even when both are small on a selected trajectory. CAVC retains video computation throughout and restores joint updates at the end, allocating the approximation to a bounded middle interval.

The analysis below keeps these residuals separate. It characterizes how a chosen interval affects the final state, while the empirical curves motivate when reuse may be useful. The executable branch schedule and cache-state conventions are given in Appendix F.

Critically, CAVC caches the denoiser output rather than freezing the latent state directly, enabling the scheduler to continue smooth variance integration along the original noise schedule $\sigma_t$. The validity of this cross-noise-level output reuse is discussed in Appendix E. Simultaneously, the video branch proceeds independently:

$$\tilde{y}_v^t = g_v(\tilde{v}_t, c, \sigma_t)\,, \qquad \tilde{v}_{t-1} = S_t^v\big(\tilde{v}_t,\, \tilde{y}_v^t\big)\,, \tag{5}$$

where $g_v$ denotes the video denoiser with AV cross-attention modules bypassed via residual skip connections while fully preserving the video self-/temporal attention and text conditioning. This skips the cross-modal attention computation, whose dense interaction cost scales with $N_v N_a$, while keeping the video branch active to refine visual content at every scheduled step.

**Stage 2: Final Joint Refinement.** CAVC restores full joint denoising for $t = t_{\text{ref}}, \ldots, 1$, producing the final states at $t = 0$:

$$\tilde{y}_v^t = f_v(\tilde{v}_t, \tilde{a}_t, c, \sigma_t), \qquad \tilde{y}_a^t = f_a(\tilde{v}_t, \tilde{a}_t, c, \sigma_t), \tag{6}$$

$$\tilde{v}_{t-1} = S_t^v(\tilde{v}_t, \tilde{y}_v^t), \qquad \tilde{a}_{t-1} = S_t^a(\tilde{a}_t, \tilde{y}_a^t). \tag{7}$$

Stage 2 denotes the native terminal joint refinement, not additional steps appended to the default pipeline. Its length is varied only in the corresponding ablation. Table 4 reports changes in the audio and video metrics under this terminal-step sweep.

### 3.3 Theoretical Analysis

We formalize the deviation between CAVC and the full sampler under standard smoothness assumptions. The analysis separates residuals introduced during Stage 1B from their propagation through subsequent updates, linking cache duration and approximation magnitude to a conditional bound on the final output discrepancy.

**Error Sources.** For the conditional analysis, assume uniform bounds on two approximation residuals over the compared cache interval. Figure 3 illustrates the corresponding quantities on selected trajectories; uniform bounds remain an analytical assumption. The audio cache error $\epsilon_a$ captures the discrepancy of freezing the audio prediction:

$$\|f_a(v_t, a_t, c, \sigma_t) - \hat{y}_a^*\| \le \epsilon_a, \tag{8}$$

and the *video-only residual* $\epsilon_v$ captures the impact of discarding AV cross-attention:

$$\|f_v(v_t, a_t, c, \sigma_t) - g_v(v_t, c, \sigma_t)\| \le \epsilon_v. \tag{9}$$

The adaptive threshold is an empirical proxy for entering a cache-tolerant audio regime, while Stage 1A warm-up reduces the dominant cross-modal dependency before AV attention is bypassed.

**Latent Deviation Bound.** Assuming the scheduler satisfies local Lipschitz conditions with constants $\alpha_t$ (w.r.t. the latent) and $\beta_t$ (w.r.t. the prediction), the cumulative latent deviations $e_a^t =$

$\|a_t - \tilde{a}_t\|$ and $e_v^t = \|v_t - \tilde{v}_t\|$ within the cached interval $\mathcal{I} = \{\tau - 1, \tau - 2, \ldots, t_{\text{ref}} + 1\}$ satisfy:

$$e_a^{t-1} \leq \alpha_t \, e_a^t + \beta_t \, \epsilon_a, \tag{10}$$

$$e_v^{t-1} \leq (\alpha_t + \beta_t L_g) \, e_v^t + \beta_t \, \epsilon_v, \tag{11}$$

where $L_g$ is the local Lipschitz constant of $g_v$. Because Stage 1A includes the full update at trigger step $\tau$, the two trajectories agree at the start of Stage 1B, i.e., $e_a^{\tau-1} = e_v^{\tau-1} = 0$. Unrolling over $\mathcal{I}$ yields:

$$e_a^{t_{\text{ref}}} \leq \sum_{t=t_{\text{ref}}+1}^{\tau-1} \left( \prod_{j=t_{\text{ref}}+1}^{t-1} \alpha_j \right) \beta_t \, \epsilon_a, \tag{12}$$

$$e_v^{t_{\text{ref}}} \leq \sum_{t=t_{\text{ref}}+1}^{\tau-1} \left( \prod_{j=t_{\text{ref}}+1}^{t-1} (\alpha_j + \beta_j L_g) \right) \beta_t \, \epsilon_v. \tag{13}$$

These bounds characterize how terminal deviation depends on $\epsilon_a$, $\epsilon_v$, and caching duration $|\mathcal{I}|$ subject to the smoothness conditions on the updates used in the derivation.

**Output Quality Bound.** Let $E_t = e_a^t + e_v^t$. Assume that each terminal joint update in Stage 2 is locally Lipschitz in this product-state norm with constant $\kappa_t$, and define $K_{\text{ref}} = \prod_{t \in \mathcal{J}} \kappa_t$ for $\mathcal{J} = \{t_{\text{ref}}, t_{\text{ref}} - 1, \ldots, 1\}$. Let decoders $D_v$, $D_a$, and a smooth scalar output functional $Q$ be locally Lipschitz with constants $L_D^v$, $L_D^a$, and $L_Q$. The final output discrepancy satisfies:

$$\begin{aligned} |Q(y) - Q(\tilde{y})| &\leq L_Q \big( L_D^v e_v^0 + L_D^a e_a^0 \big) \\ &\leq L_Q \max\{L_D^v, L_D^a\} \, K_{\text{ref}} \big( e_v^{t_{\text{ref}}} + e_a^{t_{\text{ref}}} \big) . \end{aligned} \tag{14}$$

This conditional bound carries the Stage 1B deviations through the terminal joint updates via $K_{\text{ref}}$. Table 4 evaluates their empirical recovery behavior. The complete derivation, including nested state bounding and multi-step accumulation, is detailed in Appendix G.

**Reading the conditional bound.** The residual terms describe the approximation injected at a cached step, while the products of Lipschitz factors describe how later updates propagate it. A short reuse interval reduces the number of injected terms; a small residual reduces their magnitudes; and stable downstream updates reduce amplification. The factor $K_{\text{ref}}$ accounts for terminal joint computation without requiring those updates to be contractive. These dependencies explain the complementary roles of a stability-based trigger and a bounded reuse interval. The bound characterizes perturbation propagation, while the terminal-step ablation measures the resulting quality trade-offs in generated outputs.

### 3.4 Complexity Analysis

The idealized saved work is the skipped audio/cross-attention cost minus detection overhead. Video evaluation, terminal refinement, upsampling, and decoding remain. This explains why a long cache interval can coexist with modest end-to-end speedup; measured wall-clock time must also include offload and memory traffic (Appendix J.2).

**From skipped computation to elapsed time.** Let $f$ be the fraction of full-run time spent on cache-eligible audio/cross-attention computation in the middle stage, and $q$ the cost-weighted fraction of that computation omitted by reuse. Holding other costs fixed and ignoring cache-management overhead gives the accounting model

$$r_{\text{ideal}} = \frac{1}{1 - qf}, \qquad \frac{T_{\text{full}} - T_{\text{cache}}}{T_{\text{full}}} = qf. \tag{15}$$

The unchanged video computation and the two joint-computation phases impose a ceiling on acceleration even when audio reuse begins early. The fraction $f$ is a property of the actual execution configuration, including tensor residency; it cannot be recovered from a trigger index alone. This provides a useful interpretation of the method's granularity: CAVC preserves the iterative video trajectory while concentrating its approximation on one modality, so its objective is a favorable quality–latency operating point rather than maximal removal of network evaluations. Appendix J.2 details the distinction between trigger-position proxies and actual skipped calls.

Table 1: Evaluation of **Visual Quality** and **Low-Level Statistics** across three benchmarks. Low-level statistics are no-reference diagnostic metrics; definitions appear in Appendix C. ↑ indicates higher is better, and ↓ indicates lower is better. **Bold** indicates the best performance. CAVC provides a favorable quality-efficiency trade-off while achieving inference speedups.

| Benchmark | Method | Speedup ↑ | Visual Quality Huang et al. (2025) | | | | | Low-Level Statistics | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Subj.Con↑ | Bg.Con↑ | Mot.Smth↑ | Dyn.Deg↑ | Aes.Qual↑ | Natural↑ | Temporal↑ | Flicker↓ |
| AVGen-Bench | TeaCache Liu et al. (2024a) | 0.45× | 0.864 | 0.920 | 0.989 | 0.272 | 0.452 | 0.473 | 0.619 | 0.522 |
| | Cache-DiT DefTruth (2025) | 0.24× | 0.869 | 0.920 | 0.989 | 0.298 | 0.462 | 0.463 | 0.612 | 0.517 |
| | Full AV HaCohen et al. (2026) | 1.00× | 0.7895 | 0.8622 | 0.9200 | 0.655 | **0.524** | 0.5272 | 0.6534 | 0.3346 |
| | CAVC (Ours) | **1.10**× (+10.0%) | **0.874** (+10.7%) | **0.925** (+7.3%) | **0.991** (+7.7%) | **0.6701** (+2.3%) | 0.522 | **0.5540** (+5.1%) | **0.6600** | **0.2800** (-16.3%) |
| VABench T2AV | TeaCache Liu et al. (2024a) | 0.64× | 0.936 | 0.956 | 0.993 | 0.099 | 0.482 | 0.409 | 0.668 | 0.180 |
| | Cache-DiT DefTruth (2025) | 0.35× | **0.942** | **0.959** | **0.994** | 0.094 | 0.500 | 0.402 | 0.655 | 0.170 |
| | Full AV HaCohen et al. (2026) | 1.00× | 0.9280 | 0.9530 | **0.994** | 0.4320 | 0.5540 | 0.4193 | 0.6621 | **0.1608** |
| | CAVC (Ours) | **1.35**× (+34.7%) | 0.932 | 0.954 | **0.994** | **0.4470** (+3.5%) | **0.559** | **0.509** (+21.4%) | **0.718** (+8.4%) | 0.176 |
| General-80 | TeaCache Liu et al. (2024a) | 0.39× | 0.932 | 0.957 | 0.990 | 0.275 | 0.519 | 0.477 | 0.667 | 0.467 |
| | Cache-DiT DefTruth (2025) | 0.20× | 0.938 | **0.961** | 0.991 | 0.237 | 0.544 | 0.520 | 0.649 | 0.493 |
| | Full AV HaCohen et al. (2026) | 1.00× | 0.9460 | 0.9540 | 0.9920 | 0.3800 | 0.5560 | **0.6349** | 0.7136 | 0.3346 |
| | CAVC (Ours) | **1.20**× (+20.0%) | **0.949** | **0.961** | **0.993** | **0.512** (+34.7%) | **0.561** | 0.490 | **0.718** | **0.263** (-21.4%) |

Table 2: Evaluation of **Automated Audio & MLLM-Based Quality** and **Audio-Visual Alignment & Physics** across three benchmarks. AV-F1 is an onset-based diagnostic metric. ↑ indicates higher is better; **bold** indicates best performance.

| Benchmark | Method | Speedup ↑ | Automated Audio & MLLM Quality | | | | | Audio-Visual & Physics | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | DNSMOS↑ | CLAP↑ | Aud.Real↑ | Expr.↑ | Vis.Real↑ | AV-F1↑ | AV Sync↑ | Lo-Phy↑ |
| AVGen-Bench | TeaCache Liu et al. (2024a) | 0.45× | 1.607 | 0.366 | 4.055 | 4.264 | 4.609 | 0.601 | 0.429 | **3.936** |
| | Cache-DiT DefTruth (2025) | 0.24× | 1.676 | 0.418 | 4.068 | **4.319** | 4.566 | **0.614** | 0.249 | 3.911 |
| | Full AV HaCohen et al. (2026) | 1.00× | **1.706** | 0.435 | 3.925 | 4.070 | 4.475 | 0.5381 | **0.561** | 3.847 |
| | CAVC (Ours) | **1.10**× (+10.0%) | 1.688 (-1.1%) | **0.438** | **4.077** (+3.9%) | 4.294 (+5.5%) | **4.723** (+5.5%) | 0.5650 (+5.0%) | 0.525 (-6.4%) | 3.872 |
| VABench T2AV | TeaCache Liu et al. (2024a) | 0.64× | 1.324 | 0.294 | 3.912 | **4.392** | 4.468 | **0.659** | 0.549 | **4.503** |
| | Cache-DiT DefTruth (2025) | 0.35× | 1.399 | **0.366** | 3.918 | 4.386 | **4.497** | 0.652 | 0.470 | 4.455 |
| | Full AV HaCohen et al. (2026) | 1.00× | 1.383 | 0.359 | 3.678 | 4.072 | 4.309 | 0.5150 | **0.608** | 4.255 |
| | CAVC (Ours) | **1.35**× (+34.7%) | **1.498** (+8.3%) | 0.359 | **3.965** (+7.8%) | 4.333 (+6.4%) | 4.491 (+4.2%) | 0.552 (+7.2%) | 0.594 (-2.3%) | 4.231 |
| General-80 | TeaCache Liu et al. (2024a) | 0.39× | 1.197 | 0.313 | 4.188 | **4.575** | **4.812** | **0.688** | **0.563** | **4.350** |
| | Cache-DiT DefTruth (2025) | 0.20× | 1.270 | **0.379** | **4.225** | 4.562 | **4.812** | 0.682 | 0.493 | 4.263 |
| | Full AV HaCohen et al. (2026) | 1.00× | **1.306** | 0.284 | 4.000 | 4.160 | 4.620 | 0.5381 | 0.496 | 4.180 |
| | CAVC (Ours) | **1.20**× (+20.0%) | 1.283 (-1.8%) | 0.324 (+14.1%) | 4.125 (+3.1%) | 4.450 (+7.0%) | 4.650 | 0.644 (+19.7%) | **0.563** (+13.5%) | 4.300 (+2.9%) |

# 4 EXPERIMENTS

## 4.1 EXPERIMENTAL SETUP

**Benchmarks.** We evaluate on AVGen-Bench (Zhou et al., 2026), VABench T2AV (Hua et al., 2025), and General-80. They span general scenes and synchronization challenges; Appendix B details General-80.

**Implementation Details & Baselines.** CAVC is implemented on LTX-2.3-22B-dev (HaCohen et al., 2026) with the official distilled LoRA. We compare against Full AV, Cache-DiT, and TeaCache under matched generation specifications on one NVIDIA H20. Generation and cache settings are provided in Appendix A.

## 4.2 MAIN RESULTS

Tables 1–2 report the benchmark metrics and comparators in complementary groups. Results average seeds 0, 42, and 100; Appendix A describes the stability check. Comparison scope is detailed in Appendix H.

**Speed and deployment scope.** CAVC achieves 1.10–1.35× speedup over full sampling under the H20/offload configuration. The end-to-end measurement includes the retained video branch and the pipeline stages outside the cache interval, connecting the allocation policy to practical inference time. Adapted TeaCache and Cache-DiT runs reach 0.20–0.64× relative to Full AV in this deployment. These timings characterize the evaluated implementations and offload setting; component-level profiling would be needed to separate cache-state traffic, cache-hit frequency, and other overheads. Appendix J.2 specifies the timing scope for interpreting these configuration-dependent results.

Figure 4 complements the aggregate metrics with a selected example of the visual and audio outputs under the compared sampling configurations.

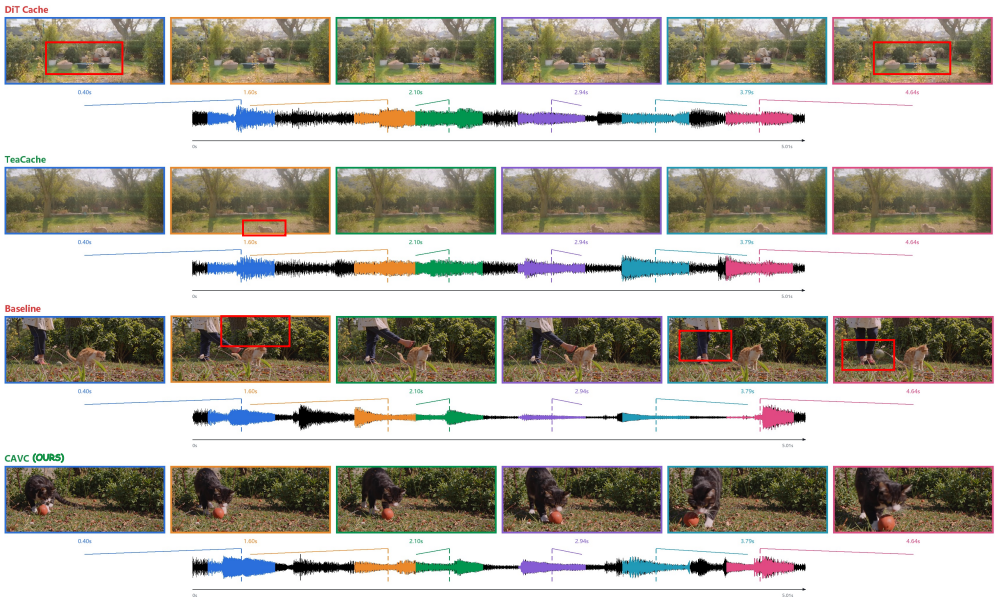


Figure 4: A reported cat-and-ball generation example under the evaluated configurations. The observed baseline slowdowns are specific to these implementations and the offload setup; this panel does not identify their cause. The single-example CAVC speedup is 1.15×.

Table 3: Comprehensive ablation studies. (a) Branch caching/freezing strategy. (b) Alternative caching strategies. (c) Phase processing components and terminal refinement. (d) Hyperparameter analysis on threshold ($\theta$) and window size ($W$). The configuration $\theta = 0.30, W = 3$ is selected as the default setting, balancing acceleration and recoverability. Audio@ denotes the average forward execution index $k_\tau$ at which audio caching is triggered, not the decreasing diffusion timestep $\tau$. ↑ indicates higher is better, and ↓ indicates lower is better.

(a) Branch Freezing Strategy

| Method (Freeze) | Speed↑ | Video Quality | | Audio & Align | |
|---|---|---|---|---|---|
| | | Bg.↑ | Dyn.↑ | AV-F1↑ | CLAP↑ |
| Video Only | 2.84× | 0.950 | 0.460 | 0.593 | **0.362** |
| Audio & Video | **3.01**× | 0.949 | 0.440 | **0.640** | 0.316 |
| Audio Only (Ours) | 1.19× | **0.960** (+1.2%) | **0.700** (+59.1%) | 0.634 | 0.312 |

(b) Alternative Caching Strategies

| Method | Speed↑ | Video Quality | | | Audio & Align | |
|---|---|---|---|---|---|---|
| | | Bg.↑ | Dyn.↑ | Aes.↑ | AV-F1↑ | CLAP↑ |
| Baseline | 1.00× | 0.958 | 0.440 | 0.562 | 0.617 | 0.287 |
| Cache Only | 1.00× | 0.960 | 0.440 | 0.566 | 0.641 | 0.291 |
| Fixed 20% | 1.09× | 0.960 | 0.500 | 0.563 | 0.643 | **0.304** |
| Fixed 80% | 1.01× | 0.959 | 0.460 | 0.560 | 0.618 | 0.290 |
| Dynamic (Ours) | **1.19**× (+19%) | **0.963** | **0.560** (+27%) | **0.572** (+1.8%) | **0.674** (+9.2%) | 0.301 |

(c) Stage Processing Ablation

| Method | Video Quality (VBench) | | | Align |
|---|---|---|---|---|
| | Subj.↑ | Smooth↑ | Dyn.↑ | Mot.↑ |
| Stage 1 (Base) | 0.936 | 0.989 | 0.540 | 1.140 |
| Stage 1 (Ours) | 0.938 | 0.989 | 0.580 (+7.4%) | 1.199 |
| Full Stages (Ours) | **0.953** (+1.8%) | **0.993** | **0.620** (+14.8%) | **1.283** (+12.5%) |

(d) Trigger Threshold and Window Size

| Threshold ($\theta$) | Window ($W$) | Time(s)↓ | Speedup↑ | Audio@ |
|---|---|---|---|---|
| *Baseline* | | 217.2 ± 0.7 | 1.00× | - |
| 0.15 | 3 | 192.8 ± 4.5 | 1.13× | 14.9 |
| 0.30 | 1 | 181.4 ± 1.0 | 1.20× | 1.5 |
| 0.30 | 2 | 182.4 ± 1.2 | 1.19× | 2.9 |
| **0.30** | **3** | **184.8 ± 2.9** | **1.18×** | **5.5** |
| 0.50 | 1 | 180.6 ± 0.6 | 1.20× | 1.1 |
| 0.50 | 2 | 181.4 ± 1.2 | 1.20× | 2.3 |
| 0.50 | 3 | 182.9 ± 0.9 | 1.19× | 3.3 |
| 0.80 | 1 | 180.4 ± 0.7 | 1.20× | 1.0 |
| 0.80 | 2 | 181.2 ± 0.6 | 1.20× | 2.0 |
| 0.80 | 3 | 182.5 ± 0.9 | 1.19× | 3.0 |

**Quality trade-offs.** On General-80, Dynamic Degree rises from 0.380 to 0.512, while Naturalness falls from 0.6349 to 0.490. A larger motion statistic could reflect either useful motion or excess variation after decoupling, and the present metrics do not separate these mechanisms. AV-F1 increases on all three benchmarks, but AV Sync falls from 0.561 to 0.525 on AVGen and from 0.608 to 0.594 on VABench. Onset coincidence and a learned temporal-offset measure assess different properties. These mixed results support a configurable approximation, with distinct quality trade-offs across the reported benchmarks and evaluation dimensions.

**Stage-wise effects.** Table 3c compares Stage 1 outputs with the complete schedule. Restoring terminal joint computation raises subject consistency from .938 to .953, motion smoothness from .989 to .993, and Dynamic Degree from .580 to .620. These within-panel changes (Appendix I) complement the terminal-step sweep below: they describe the full configuration's visual profile, while

the sweep separates the effects of different terminal lengths. The metric definitions in Appendix C distinguish motion prevalence from perceptual quality.

### 4.3 Branch Allocation and Terminal Refinement

**Terminal-step ablation.** Table 4 reports the terminal-refinement sweep. With audio caching, one step closes 81.5% of the CLAP gap to Full AV; three and five steps close 87.0% and 91.3%. These percentages use $(s_m - s_0)/(s_{\text{full}} - s_0)$ separately for each metric. The large first-step improvement highlights the role of restoring joint computation for audio–text alignment, with further CLAP recovery as terminal refinement increases.

Table 4: Terminal joint-step sweep after branch caching. Percentages show closure of the CLAP (audio) or Dynamics (video) gap from the 0-step condition to Full AV.

| Cached Branch | Stage 2 | Dyn.↑ | Aes.Qual↑ | AV Sync↑ | CLAP↑ |
|---|---|---|---|---|---|
| None (Full AV) | 3 steps | 0.600 | 0.602 | 0.511 | 0.271 |
| Audio (CAVC) | 0 steps | 0.560 | 0.563 | 0.495 | -0.005 (0.0% Rec.) |
| Audio (CAVC) | 1 step | 0.640 | 0.603 | 0.679 | 0.220 (81.5% Rec.) |
| Audio (CAVC) | 3 steps | 0.620 | 0.596 | 0.502 | 0.235 (87.0% Rec.) |
| Audio (CAVC) | 5 steps | 0.620 | 0.596 | 0.566 | 0.247 (91.3% Rec.) |
| Video | 0 steps | 0.360 (0.0% Rec.) | 0.286 | 0.290 | 0.300 |
| Video | 1 step | 0.320 (-16.7% Rec.) | 0.588 | 0.590 | 0.237 |
| Video | 3 steps | 0.360 (0.0% Rec.) | 0.575 | 0.456 | 0.289 |
| Video | 5 steps | 0.360 (0.0% Rec.) | 0.570 | 0.464 | 0.294 |

The sweep is non-monotonic: among audio-cached rows, one terminal step has the highest Dynamic Degree, aesthetic score, and AV Sync; five steps has the highest CLAP. The default three-step setting retains the native pipeline configuration. Thus, the sweep exposes a choice between audio–text alignment and other quality dimensions rather than a single optimum across metrics. Since terminal length also shortens the cached interval, each setting combines its approximation duration with the corresponding amount of joint refinement.

**Allocation preference.** The branch controls in Table 3a make the trade-off explicit. Audio-only caching has Dynamic Degree 0.700, compared with 0.460 for video-only and 0.440 for joint caching. However, video-only caching is faster (2.84× versus 1.19×) and has higher CLAP (0.362 versus 0.312); joint caching has slightly higher AV-F1 (0.640 versus 0.634). Retaining video computation is a design preference supported by the motion profile, with the accompanying audio and runtime trade-offs visible in the same comparison. This allocation is especially relevant when preserving an evolving visual scene is central to the output: video computation continues at every step, while audio-prediction reuse supplies the computational saving. The branch comparison makes that design choice explicit alongside its measured alternatives.

**Trigger behavior.** The adaptive policy yields 1.19× speedup and AV-F1 0.674 in the trigger-strategy ablation, versus 1.09× and 0.643 for the fixed-20% policy. Fixed-20% has slightly higher CLAP (0.304 versus 0.301), identifying a modest audio–text alignment trade-off alongside the runtime and onset-alignment gains. The threshold/window sweep records timing and trigger positions: $\theta = 0.15, W = 3$ delays the mean trigger to $k_\tau = 14.9$, while $\theta = 0.30, W = 3$ triggers at 5.5 with 1.18× speedup. Appendix D provides benchmark trigger statistics; repeated-seed stability is discussed separately in Appendix A.

## 5 Conclusion

CAVC introduces a training-free, modality-asymmetric allocation rule for joint audio–visual diffusion. Joint warm-up is followed by audio-prediction reuse under an online stability condition and terminal joint refinement, while the video branch remains active and the audio latent continues under its native scheduler. The reported LTX-2.3 deployment achieves 1.10–1.35× end-to-end speedup across three benchmarks. Branch interventions support retaining video computation for the measured motion profile, and the terminal sweep shows substantial recovery of audio alignment after prediction reuse. Conditional perturbation analysis relates the approximation residuals and cache duration to output deviation. Together, these results identify modality-level allocation as a useful way to reduce repeated joint computation without changing pretrained weights.

**Limitations and future work.** Quality trade-offs remain metric dependent, and the trigger is a local heuristic. Additional backbones, component-level timing, and broader prompt coverage would further characterize the operating range of this schedule.

## AI USE STATEMENT

Generative AI tools were used to assist with language editing, manuscript organization, and consistency checking. The authors take responsibility for the final manuscript, including its mathematical derivations, implementation descriptions, experimental values, citations, and claims.

## ETHICS STATEMENT

CAVC inherits the risks of its underlying audio–video generator, including misleading media and unauthorized imitation of people or voices. Acceleration provides no content-safety guarantee. Deployment should retain the generator's safeguards and require appropriate consent, provenance, and synthetic-content disclosure.

## REFERENCES

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4599–4603, 2023.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster, 2023. URL https://arxiv.org/abs/2210.09461.

Haoxin Chen, Menghan Xia, Yingqing He, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Jinbo Xing, Yaofang Liu, Qifeng Chen, Xintao Wang, et al. Videocrafter1: Open diffusion models for high-quality video generation. *arXiv preprint arXiv:2310.19512*, 2023.

etc. DefTruth, vipshop.com. Cache-dit: A pytorch-native inference engine with cache, parallelism, quantization and cpu offload for dits., 2025. URL https://github.com/vipshop/cache-dit.git. Open-source software available at https://github.com/vipshop/cache-dit.git.

Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. CLAP: Learning audio concepts from natural language supervision. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2023.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entekhabi, Jonas Saini, Yam Levi Baumann, et al. Scaling rectified flow models for image generation. In *International Conference on Machine Learning (ICML)*, 2024.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models, 2023. URL https://arxiv.org/abs/2305.10924.

Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023.

Yoav HaCohen, Benny Brazowski, Nisan Chiprut, Yaki Bitterman, Andrew Kvochko, Avishai Berkowitz, Daniel Shalem, Daphna Lifschitz, Dudu Moshe, Eitan Porat, Eitan Richardson, Guy Shiran, Itay Chachy, Jonathan Chetboun, Michael Finkelson, Michael Kupchick, Nir Zabari, Nitzan Guetta, Noa Kotler, Ofir Bibi, Ori Gordon, Poriya Panet, Roi Benita, Shahar Armon, Victor Kulikov, Yaron Inger, Yonatan Shiftan, Zeev Melumian, and Zeev Farbman. Ltx-2: Efficient joint audio-visual foundation model, 2026. URL https://arxiv.org/abs/2601.03233.

Xu He, Haoxian Zhang, Hejia Chen, Changyuan Zheng, Liyang Chen, Songlin Tang, Jiehui Huang, Xiaoqiang Liu, Pengfei Wan, and Zhiyong Wu. Just-dub-it: Video dubbing via joint audio-visual diffusion. *arXiv preprint arXiv:2501.00103*, 2025.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

Daili Hua, Xizhi Wang, Bohan Zeng, Xinyi Huang, Hao Liang, Junbo Niu, Xinlong Chen, Quanqing Xu, and Wentao Zhang. Vabench: A comprehensive benchmark for audio-video generation. *arXiv preprint arXiv:2512.09299*, 2025.

Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. In *International Conference on Machine Learning*, pp. 13916–13932. PMLR, 2023.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. VBench: Comprehensive benchmark suite for video generative models. *arXiv preprint arXiv:2311.17982*, 2024.

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. doi: 10.1109/TPAMI.2025.3633890.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S. Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15240–15252, 2025.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. *arXiv preprint arXiv:2411.19108*, 2024a.

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*, 2023.

Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:2871–2883, 2024b.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023a.

Simian Luo, Chuanhao Yan, Chenxu Hu, and Hang Zhao. Diff-foley: Synchronized video-to-audio synthesis with latent diffusion models, 2023b.

Zhengyao Lyu, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K Wong. Fastercache: Training-free video diffusion model acceleration with high quality. In *International Conference on Learning Representations*, volume 2025, pp. 33132–33156, 2025.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15762–15772, 2024.

Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14297–14306, 2023.

Ludan Ruan, Yiyang Ma, Huan Yang, Huiguo He, Bei Liu, Jianlong Fu, Nicholas Jing Yuan, Qin Jin, and Baining Guo. Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10219–10228, 2023.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Zhao Sun et al. Pav-dit: A cross-modal alignment projected latent diffusion transformer for synchronized audio-video generation. *OpenReview*, 2025.

Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. Any-to-any generation via composable diffusion. *Advances in Neural Information Processing Systems*, 36:16083–16099, 2023.

Zineng Tang, Ziyi Yang, Mahmoud Khademi, Yang Liu, Chenguang Zhu, and Mohit Bansal. Codi-2: In-context interleaved and interactive any-to-any generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27425–27434, 2024.

Jun Wang, Chunyu Qiang, Yuxin Guo, Yiran Wang, Xijuan Zeng, Chen Zhang, and Pengfei Wan. Klear: Unified multi-task audio-video joint generation. *arXiv preprint arXiv:2601.04151*, 2026.

Xinyu Wang et al. Vidtome: Video token merging for zero-shot video editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6211–6220, 2024.

Yazhou Xing, Yingqing He, Zeyue Tian, Xintao Wang, and Qifeng Chen. Seeing and hearing: Open-domain visual-audio generation with diffusion latent aligners. In *CVPR*, 2024.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024.

Yiming Zhang, Yicheng Gu, Yanhong Zeng, Zhening Xing, Yuancheng Wang, Zhizheng Wu, Bin Liu, and Kai Chen. Foleycrafter: Bring silent videos to life with lifelike and synchronized sounds. *arXiv preprint arXiv:2407.01494*, 2024.

Wangbo Zhao, Yizeng Han, Jiasheng Tang, Kai Wang, Yibing Song, Gao Huang, Fan Wang, and Yang You. Dynamic diffusion transformer. *arXiv preprint arXiv:2403.06977*, 2024.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:49842–49869, 2023.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. In *International Conference on Learning Representations*, 2025.

Ziwei Zhou, Zeyuan Lai, Rui Wang, Yifan Yang, Zhen Xing, Yuqing Yang, Qi Dai, Lili Qiu, and Chong Luo. Avgen-bench: A task-driven benchmark for multi-granular evaluation of text-to-audio-video generation, 2026. URL https://arxiv.org/abs/2604.08540.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *International Conference on Learning Representations*, 2025.

# A Detailed Experimental Protocol

**Random seeds and result aggregation.** The numerical performance results in the main-text tables are arithmetic means of runs with seeds 0, 42, and 100. For metric $m$, we first obtain its benchmark-level value $m_s$ within each run and report $\bar{m} = (m_0 + m_{42} + m_{100})/3$. An additional run with seed 1000 checks stability and is excluded from this mean. Results remain stable across the tested seeds. This check concerns variation under the stated generation configuration; prompt-set composition and backbone transfer remain separate evaluation questions. Sample counts and configuration parameters are fixed descriptors, not averaged measurements.

## A.1 Generation Configuration

Table 5 records the generation settings used for the reported experiments. Each sample is generated independently on one NVIDIA H20 GPU with 96 GB memory. CPU offload is enabled for all four methods in the reported runs.

Table 5: Reported generation configuration.

| | |
|---|---|
| Base model | LTX-2.3-22B-dev |
| Distilled LoRA | `ltx-2.3-22b-distilled-lora-384-1.1`, strength 0.8 |
| Spatial upscaler | `ltx-2.3-spatial-upscaler-x2-1.1` |
| Denoising steps | 30 |
| Aggregation seeds | 0, 42, 100 |
| Stability-check seed | 1000 |
| AVGen-Bench and VABench | $1280 \times 704$, 241 frames, 24 FPS |
| General-80 | $1536 \times 1024$, 121 frames, 24 FPS |
| Video CFG / Audio CFG | 3.0 / 7.0 |
| Video STG / Audio STG | 1.0 / 1.0 |
| Video/Audio modality scale | 3.0 / 3.0 |
| Video/Audio guidance rescale | 0.7 / 0.7 |
| STG block | 28 |
| Numerical precision | BF16 |
| Hardware | One NVIDIA H20 (96 GB) per generated sample |
| Batch size / generation concurrency | 1 / 1 |
| CPU offload | Enabled for all reported methods |

## A.2 Acceleration-Method Configuration

The behavior-affecting cache parameters are listed in Table 6. Cache-DiT uses its official LTX2 implementation. TeaCache uses a public implementation adapted to the LTX-2.3 dual-stream pipeline. The parameter table specifies the evaluated adaptation; comparisons with other implementations must also control their cache policies and execution environment.

Table 6: Reported cache settings. No maximum is imposed by $-1$.

| | |
|---|---|
| CAVC: convergence threshold $\theta$ / window $W$ | 0.30 / 3 |
| Cache-DiT: warm-up steps | 8 |
| Cache-DiT: front/back compute blocks | 8 / 0 |
| Cache-DiT: residual-difference threshold | 0.08 |
| Cache-DiT: max cached / continuous cached steps | $-1$ / $-1$ |
| Cache-DiT: TaylorSeer / transformer compilation | disabled / disabled |
| TeaCache: `rel_l1_thresh` | 0.20 |
| TeaCache: active interval | 0.15–1.00 of the denoising trajectory |
| TeaCache: polynomial coefficients | [21.4700694, $-12.8016453$, 2.31279151, 0.792487521, 0.00969274326] |

For TeaCache, the relative first-block change is

$$d_k = \frac{\mathrm{mean}(|x^{(k)} - x^{(k-1)}|)}{\max(\mathrm{mean}(|x^{(k-1)}|), 10^{-8})}. \tag{16}$$

The polynomial mapping of $d_k$ is accumulated. A residual is reused while the accumulated value remains below 0.20; otherwise, a full transformer evaluation is performed and the accumulator is reset.

## B General-80 Prompt Set

General-80 contains a General-40 split and an AV-hard-40 split. General-40 covers natural scenes, cities, people, animals, sports, cooking, industry, transportation, stage performances, and dynamic physical events. AV-hard-40 is balanced across the eight categories in Table 7, with five prompts per category.

Table 7: General-80 composition.

| | |
|---|---|
| General | 40 |
| Speech and lip synchronization | 5 |
| Music and pitch | 5 |
| Environmental sound | 5 |
| Physical-event synchronization | 5 |
| Animal vocalization | 5 |
| Mechanical sound and motion | 5 |
| Crowd ambience | 5 |
| Multiple audio-visual events | 5 |
| Total | 80 |

Prompts were selected to provide broad visual coverage, observable sound sources or event times, balanced AV-hard categories, and a mixture of single-event and multi-event scenes. The list is deterministic and does not depend on online sampling. The split composition specifies coverage, but does not provide an estimate of performance variance across independently sampled prompt sets.

# C Evaluation Metrics and Evaluators

## C.1 VBench Visual Metrics

We use the VBench implementation (Huang et al., 2024). The definitions below list six visual dimensions defined by the evaluator and their evaluators.

**Subject consistency** DINO ViT-B/16. Mean non-negative feature similarity to the first and preceding frames.

**Background consistency** CLIP ViT-B/32. Analogous temporal similarity using CLIP image features.

**Motion smoothness** AMT-S. Agreement between interpolated and observed intermediate frames.

**Dynamic degree** RAFT. Fraction of videos exceeding the resolution-adjusted optical-flow criterion.

**Aesthetic quality** CLIP ViT-L/14 + LAION predictor. Mean predicted aesthetic score over sampled frames.

**Imaging quality** MUSIQ-SPAQ. No-reference per-frame quality, scaled by VBench to $[0, 1]$.

The visual metric suite uses a higher-is-better convention; the main visual-quality table reports five of these dimensions. Dynamic degree measures the presence of motion and should not, by itself, be interpreted as overall perceptual quality.

## C.2 Audio Quality and MLLM Evaluators

**DNSMOS.** We use the DNSMOS P.835 local implementation with `sig_bak_ovr.onnx` and `model_v8.onnx`. Audio is converted to mono 16 kHz, and the reported value is the calibrated overall MOS estimate (higher is better).

**CLAP.** We use LAION `clap-htsat-unfused` (Elizalde et al., 2023). Audio is converted to mono 48 kHz and compared with the audio prompt. The score is

$$S_{\text{CLAP}} = \cos(e_{\text{text}}, e_{\text{audio}}). \tag{17}$$

**Qwen2.5-Omni evaluators.** The Audio Reality, Visual Reality, and Expressiveness dimensions follow the VABench evaluator protocol (Hua et al., 2025) with Qwen2.5-Omni-7B. Decoding is deterministic (`do_sample=False`). Videos are sampled at 1 FPS with 4–8 frames and at most 150,528 input pixels. Each evaluator returns one integer score from 1 to 5 and a timestamped reason in a fixed JSON object. Audio Reality evaluates scale–pitch, material–timbre, and action-energy consistency; Visual Reality evaluates object permanence, biomechanics, rendering, temporal coherence, and material interaction; Expressiveness evaluates emotional alignment and narrative function. These model-based assessments are protocol-dependent automated scores rather than human ratings.

### C.3 Audio-Visual Alignment and Physics

**AV Sync.** SyncFormer produces an absolute temporal offset $|\hat{o}_{\text{SyncFormer}}|$ in seconds. The raw offset is lower-is-better. For the unified tables, we map it to

$$S_{\text{AVSync}} = \max\left(0, 1 - \frac{|\hat{o}_{\text{SyncFormer}}|}{0.5}\right). \tag{18}$$

The transformed score lies in $[0, 1]$ and is higher-is-better; zero offset maps to 1, and offsets of 0.5 seconds or more map to 0.

**VideoPhy2 / Lo-Phy.** We use the VideoPhy2 evaluator distributed with AVGen-Bench (Zhou et al., 2026). It uniformly samples 32 frames, uses deterministic decoding, and returns a 1–5 assessment of local physical plausibility. The table reports the arithmetic mean (higher is better).

**AV-F1.** AV-F1 is a no-reference onset-alignment diagnostic. Visual onsets are local peaks in frame difference above the 75th percentile; audio onsets are local peaks in STFT spectral flux above the 75th percentile. Events are matched bidirectionally within $\pm 3$ video frames. With the resulting precision $P$ and recall $R$,

$$S_{\text{AV-F1}} = \frac{2PR}{P + R}, \tag{19}$$

with a score of zero when $P + R = 0$. Higher values indicate greater agreement between detected audio and visual event times; this handcrafted diagnostic is not a learned semantic evaluator.

### C.4 Pure-Pixel No-Reference Diagnostics

The pure-pixel diagnostics resize the long side to at most 480 pixels and evaluate at most 128 frames. Except for the audio onset extraction used by AV-F1, they require no pretrained weights. The definitions below specify these diagnostics.

**Sharpness** Variance of the grayscale Laplacian. Higher.

**Colorfulness** Hasler–Suesstrunk opponent-color statistic. Higher.

**Naturalness** Luminance/gradient kurtosis proximity to reference values. Higher.

**Temporal consistency** $1/(1 + \text{std}(d)/(\text{mean}(d) + \varepsilon))$ for frame differences $d$. Higher.

**Flicker** Mean second difference of frame-average brightness changes. Lower.

**Motion smoothness** Consecutive Farneback-flow directional agreement, mapped to $[0, 1]$. Higher.

**Artifact ratio** Fraction of $8 \times 8$ patches with local-variance $z$-score above 3. Lower.

**AV-F1** Bidirectional onset matching within $\pm 3$ frames. Higher.

## D Audio Cache Trigger Curves

CAVC monitors the audio denoiser-output change $\Delta_a^t$ and activates caching after it remains below threshold $\theta$ for $W$ consecutive execution steps. Diffusion time $t$ decreases from $T$ to 0, whereas the forward execution index $k$ used in the plots and trigger statistics increases. This is an online cacheability criterion, not a claim that audio numerically converges faster than video.

Table 8: Default trigger statistics ($\theta = .30$, $W = 3$). Proxy denotes $1 - \overline{k_\tau}/30$.

| Statistic | AVGen | VABench |
|---|---|---|
| Median $k_\tau$ | 4 | 4 |
| Mean $k_\tau$ | 5.29 | 4.91 |
| Proxy (%) | 82.4 | 83.6 |
| Time (s) | 329.69 | 392.73 |
| Speedup | 1.10 | 1.35 |

## E Validity of Cross-Noise-Level Prediction Reuse

In Stage 1B, the cached audio denoiser output $\hat{y}_a^* = \hat{y}_a^\tau$ is reused at lower noise levels $t < \tau$, while the scheduler continues to update the audio latent:

$$\tilde{a}_{t-1} = S_t^a(\tilde{a}_t, \hat{y}_a^*). \tag{20}$$

Thus, CAVC does not freeze the audio latent or the noise schedule. Let the full prediction residual during the cached interval satisfy $\|\hat{y}_a^t - \hat{y}_a^*\| \le \epsilon_a$, and assume that $S_t^a$ is locally Lipschitz in its latent and prediction arguments with constants $\alpha_t$ and $\beta_t$. The latent deviation then obeys

$$e_a^{t-1} \le \alpha_t e_a^t + \beta_t \epsilon_a, \tag{21}$$

and, because the full trigger-step update gives $e_a^{\tau-1} = 0$,

$$e_a^{t_{\text{ref}}} \le \sum_{t=t_{\text{ref}}+1}^{\tau-1} \left( \prod_{j=t_{\text{ref}}+1}^{t-1} \alpha_j \right) \beta_t \epsilon_a. \tag{22}$$

This conditional bound relates prediction-reuse error to residual magnitude, scheduler sensitivity, and reuse duration. The residual curves and terminal-refinement ablations provide complementary empirical measurements of approximation and recovery.

## F Execution Order and Cache State

The prediction being cached is the native audio output supplied to the scheduler. It is reused in that parameterization; no conversion to an assumed noise or clean-sample prediction is introduced by the equations. The audio state and noise level still advance at every update. The first warm-up evaluation initializes the previous prediction and cannot itself establish a consecutive-change test. The persistence counter resets when the threshold test fails. Once the cache activates, it remains fixed until the terminal joint phase.

If the persistence condition is never met before Stage 2, the trajectory uses joint computation throughout, with detector overhead. A trigger at the boundary cannot remove a terminal joint evaluation. The number of full warm-up calls is $n_\tau = k_\tau + 1$ because $k_\tau$ is zero based and the trigger step itself uses the full model. This distinction is necessary when relating the reported trigger positions to saved calls.

**Local stability and cumulative reuse.** The detector controls subsequent evaluations, while both schedulers continue to advance. Appendix J.1 distinguishes the observed stability window from the cumulative prediction error after activation.

**Algorithm 1** CAVC branch schedule for one denoising trajectory

**Require:** Ordered timesteps, joint model $(f_v, f_a)$, video-only model $g_v$, threshold $\theta$, window $W$, native schedulers
Initialize latents, cache-inactive flag, previous prediction, and counter $b = 0$
**for** each timestep $t$ in execution order **do**
  **if** $t$ belongs to Stage 2 **then**
    Evaluate $(\hat{y}_v, \hat{y}_a) = (f_v, f_a)(\tilde{v}_t, \tilde{a}_t, c, \sigma_t)$
  **else if** cache is active **then**
    Evaluate $\hat{y}_v = g_v(\tilde{v}_t, c, \sigma_t)$ and reuse $\hat{y}_a = \hat{y}_a^*$
  **else**
    Evaluate joint predictions $(\hat{y}_v, \hat{y}_a)$
    Update $b$ using the adjacent audio-output test if a previous output exists
    Save the current audio output as the previous output
    **if** $b \geq W$ **then**
      Set $\hat{y}_a^* = \hat{y}_a$ and activate cache for subsequent middle steps
    **end if**
  **end if**
  Update both latents with their native schedulers and selected predictions
**end for**
Decode the final audio and video latents

# G Detailed Theoretical Derivation of the Output Deviation Bound

This section derives a conditional deviation bound for CAVC. The result depends on bounded prediction residuals and local Lipschitz assumptions for the scheduler, the video-only network, terminal joint updates, decoders, and the scalar evaluation function. It characterizes how approximation errors can propagate; it is not an unconditional perceptual-quality guarantee.

## G.1 Problem Formulation and Assumptions

We formalize the exact joint denoising process and the approximated CAVC process to define the error metrics.

**Definition of Variables.** Let $v_t$ and $a_t$ denote the exact video and audio latents at denoising step $t \in \{T, T-1, \ldots, 1\}$, generated by the standard full joint workflow using the exact models $f_v(t) = f_v(v_t, a_t, c, \sigma_t)$ and $f_a(t) = f_a(v_t, a_t, c, \sigma_t)$. Denoising maps the state at $t$ to that at $t-1$ and terminates at $t = 0$. Conversely, let $\tilde{v}_t$ and $\tilde{a}_t$ denote the approximated latents produced by the CAVC framework. We quantify the cumulative deviation at step $t$ using the $L_2$ norms:

$$e_a^t = \|a_t - \tilde{a}_t\|, \qquad e_v^t = \|v_t - \tilde{v}_t\| \tag{23}$$

**Bounded-Residual Assumptions.** Let $\mathcal{I} = \{\tau - 1, \tau - 2, \ldots, t_{\text{ref}} + 1\}$ denote the cached interval and let $\hat{y}_a^*$ be the full audio prediction cached at trigger step $\tau$. We assume that finite constants $\epsilon_a$ and $\epsilon_v$ exist over $\mathcal{I}$ such that

$$\|f_a(t) - \hat{y}_a^*\| \leq \epsilon_a \tag{24}$$

Let $g_v(t) = g_v(v_t, c, \sigma_t)$ denote the isolated video network where audio cross-attention modules are explicitly bypassed. The bounded *video-only residual* incurred by removing audio conditioning is defined as:

$$\|f_v(v_t, a_t, c, \sigma_t) - g_v(v_t, c, \sigma_t)\| \leq \epsilon_v \tag{25}$$

**Lipschitz Constraints.** We assume the updating scheduler $S_t(x, p)$ satisfies local Lipschitz continuity with respect to both the latent state $x$ (with constant $\alpha_t$) and the network prediction $p$ (with constant $\beta_t$):

$$\|S_t(x, p) - S_t(x', p)\| \leq \alpha_t \|x - x'\| \tag{26}$$

$$\|S_t(x, p) - S_t(x, p')\| \leq \beta_t \|p - p'\| \tag{27}$$

These inequalities are assumptions over the evaluated trajectory, not consequences of the trigger threshold alone. Figure 3 empirically measures the corresponding residual behavior.

### G.2 Single-Step Deviation Recurrence

We derive the propagation of errors from step $t$ to $t-1$ for both modalities independently.

**Audio Branch Single-Step Deviation.** Expanding the definition of the audio error at $t-1$:

$$e_a^{t-1} = \|S_t^a(a_t, f_a(t)) - S_t^a(\tilde{a}_t, \hat{y}_a^*)\| \tag{28}$$

To decouple the error caused by the drifted state $\tilde{a}_t$ from the error caused by the cached prediction $\hat{y}_a^*$, we introduce an intermediate virtual state $S_t^a(\tilde{a}_t, f_a(t))$ and apply the triangle inequality:

$$\begin{aligned} e_a^{t-1} \leq \|S_t^a(a_t, f_a(t)) - S_t^a(\tilde{a}_t, f_a(t))\| \\ + \|S_t^a(\tilde{a}_t, f_a(t)) - S_t^a(\tilde{a}_t, \hat{y}_a^*)\| \end{aligned} \tag{29}$$

Applying the Lipschitz constraints (Eq. 26 and 27):

$$e_a^{t-1} \leq \alpha_t \|a_t - \tilde{a}_t\| + \beta_t \|f_a(t) - \hat{y}_a^*\| \tag{30}$$

Substituting the bounds, we obtain the audio recurrence:

$$e_a^{t-1} \leq \alpha_t e_a^t + \beta_t \epsilon_a \tag{31}$$

**Video Branch Single-Step Deviation (Nested Bounding).** The derivation for the video branch is more intricate because the network structure itself changes from $f_v$ to $g_v$. The deviation is:

$$e_v^{t-1} = \|S_t^v(v_t, f_v(t)) - S_t^v(\tilde{v}_t, g_v(\tilde{v}_t))\| \tag{32}$$

Inserting the intermediate state $S_t^v(\tilde{v}_t, f_v(t))$ and applying the triangle inequality yields:

$$e_v^{t-1} \leq \alpha_t e_v^t + \beta_t \|f_v(t) - g_v(\tilde{v}_t)\| \tag{33}$$

The critical challenge is evaluating the prediction discrepancy $\|f_v(t) - g_v(\tilde{v}_t)\|$, which encapsulates two distinct sources of error: the architectural change (removing audio) and the cascading effect of inputting a drifted state $\tilde{v}_t$ into the altered network $g_v$.

We disentangle these by inserting another intermediate term, $g_v(v_t)$, applying the triangle inequality a second time:

$$\|f_v(t) - g_v(\tilde{v}_t)\| \leq \|f_v(t) - g_v(v_t)\| + \|g_v(v_t) - g_v(\tilde{v}_t)\| \tag{34}$$

The first term is exactly the structurally induced video-only residual, bounded by $\epsilon_v$. For the second term, assuming the isolated video network $g_v$ is locally Lipschitz continuous with constant $L_g$:

$$\|g_v(v_t) - g_v(\tilde{v}_t)\| \leq L_g \|v_t - \tilde{v}_t\| = L_g e_v^t \tag{35}$$

Consolidating these gives the total prediction discrepancy:

$$\|f_v(t) - g_v(\tilde{v}_t)\| \leq \epsilon_v + L_g e_v^t \tag{36}$$

Substituting this back into the scheduler recurrence, we derive the final single-step bound for the video branch:

$$e_v^{t-1} \leq (\alpha_t + \beta_t L_g) e_v^t + \beta_t \epsilon_v \tag{37}$$

The additional $\beta_t L_g$ term accounts for evaluating the video-only network at a perturbed input. Comparing the algebraic forms of these bounds does not prove that realized video errors exceed realized audio errors.

### G.3 Multi-Step Accumulation

We unroll the recurrences over $\mathcal{I}$. Because Stage 1A includes the full update at trigger step $\tau$, the exact and approximate states agree at the beginning of Stage 1B, so $e_a^{\tau-1} = e_v^{\tau-1} = 0$. The audio deviation at the end of Stage 1B is

$$e_a^{t_{\text{ref}}} \leq \sum_{t=t_{\text{ref}}+1}^{\tau-1} \left( \prod_{j=t_{\text{ref}}+1}^{t-1} \alpha_j \right) \beta_t \epsilon_a. \tag{38}$$

Here, an empty product is defined as one. Let $h_t$ denote the discrete integration step size and additionally assume $\alpha_t \leq 1 + Lh_t$ and $\beta_t \leq Bh_t$, with $H = \sum_{t=t_{\text{ref}}+1}^{\tau-1} h_t$. Using $1 + x \leq \exp(x)$ gives

$$e_a^{t_{\text{ref}}} \leq B\epsilon_a H \exp(LH), \tag{39}$$

We apply the identical logic to the video recurrence (Eq. 37). The effective state amplification factor is bounded by:

$$\alpha_t + \beta_t L_g \leq 1 + (L + BL_g)h_t \tag{40}$$

and therefore

$$e_v^{t_{\text{ref}}} \leq B\epsilon_v H \exp((L + BL_g)H). \tag{41}$$

### G.4 Terminal Joint Propagation

Stage 2 applies the same full joint update to the exact and approximate states. Define the product-state deviation $E_t = e_a^t + e_v^t$ and assume that each terminal joint update is locally Lipschitz in this norm with constant $\kappa_t$. For the terminal interval $\mathcal{J} = \{t_{\text{ref}}, t_{\text{ref}} - 1, \ldots, 1\}$,

$$E_0 \leq K_{\text{ref}} E_{t_{\text{ref}}}, \qquad K_{\text{ref}} = \prod_{t \in \mathcal{J}} \kappa_t. \tag{42}$$

The bound allows either contraction or amplification; the empirical recoverability experiment in Table 4 measures the behavior of the actual terminal updates.

### G.5 Output Deviation Bound

Let the video and audio decoders, $D_v$ and $D_a$, and a scalar evaluation function $Q$ be locally Lipschitz on the compared trajectories with constants $L_D^v, L_D^a$, and $L_Q$, respectively.

The final scalar-output discrepancy between the full sampler and CAVC is

$$\begin{aligned} &|Q(\text{full}) - Q(\text{CAVC})| = \\ &\qquad |Q(D_v(v_0), D_a(a_0)) - Q(D_v(\tilde{v}_0), D_a(\tilde{a}_0))| \end{aligned} \tag{43}$$

Applying the triangle inequality and the stated local assumptions yields

$$|Q(\text{full}) - Q(\text{CAVC})| \leq L_Q \left( L_D^v e_v^0 + L_D^a e_a^0 \right) \tag{44}$$

Combining Equations 39–42 with Equation 44 makes the dependence on $\epsilon_a$, $\epsilon_v$, cached duration $H$, and the assumed local constants explicit. This is an output-deviation characterization under the listed assumptions. It neither guarantees an improvement in any perceptual metric nor rules out mode collapse or other failure modes.

## H Interpreting the Complete Benchmark Results

Tables 1–2 in the main text report the complete comparison. Speedup is relative to Full AV in the same deployment. Boldface identifies the best values in each metric's stated direction, not statistically significant differences. The speedups below unity are configuration-level observations; they should not be read as the best achievable operating points of those methods.

## I Detailed Ablations

Tables 3a–3d in the main text use the reported configurations. Comparisons should be made within each panel: their different Full AV reference values do not define a shared normalization across panels. The branch controls identify which branch computation is reused or disabled; CAVC specifically reuses denoiser predictions while updating the audio latent. They do not establish an equivalent latent-freezing implementation for every control.

# J Interpreting the Detector and Runtime

## J.1 Adjacent variation and future cache error

For predictions $p_i$ in forward execution order, the triangle inequality gives

$$\|p_k - p_{k_\tau}\| \le \sum_{i=k_\tau+1}^{k} \|p_i - p_{i-1}\|. \tag{45}$$

The detector observes only the terms before cache activation. Even if each future term were bounded by $\delta$, the cumulative bound would grow as $(k-k_\tau)\delta$. Without such a future-variation assumption, the observed threshold provides no upper bound on subsequent reuse error. This is why the theory treats $\epsilon_a$ as a separate assumption. The mean absolute detector also has a different normalization from the $L_2$ residual in the proof; its threshold cannot be substituted for $\epsilon_a$ without a dimension-dependent conversion and additional assumptions. No monotone decrease of audio residuals is required by the implementation or established by the analysis.

## J.2 Timing arithmetic and deployment interpretation

Table 9 expresses the reported speedups as elapsed-time savings, computed as $1 - 1/r$. These are algebraic transformations of the aggregate results, not additional timing measurements.

Table 9: Derived end-to-end time savings from reported speedups.

| Benchmark | Speedup $r$ | Time saved $100(1 - 1/r)$ |
|---|---|---|
| AVGen-Bench | 1.10 | 9.09% |
| VABench T2AV | 1.35 | 25.93% |
| General-80 | 1.20 | 16.67% |

Using Equation 15, $f$ is the full-run time fraction attributable to cache-eligible middle-stage computation, and $q$ is the cost-weighted fraction omitted. The zero-overhead accounting model gives $r = (1 - qf)^{-1}$. Fixed video and terminal costs bound the saving. This identity is an accounting illustration, not a substitute for measured component times: offload transfers and cache residency can change with the policy. The reported data do not identify their individual costs.

The trigger-position percentages in Table 8 equal $100(1 - \overline{k_\tau}/30)$: 82.4% and 83.6% after rounding. They are position proxies, not skipped-call fractions. Under the unified $T$-step schedule in Section 3.2, the trigger call belongs to warm-up and $t_{\text{ref}}$ calls are reserved for Stage 2. A trajectory that activates caching therefore omits $n_{\text{skip}} = \max\{0, T - k_\tau - 1 - t_{\text{ref}}\}$ audio evaluations. Any empirical average must also account for trajectories that never trigger. This expression reconciles the call accounting with the method; the position proxy alone is not a measured cache-hit count.

## J.3 Metric scope and statistical interpretation

DNSMOS is an automated quality predictor and is not a listening study. CLAP measures text–audio embedding similarity, while AV-F1 detects onset coincidence and AV Sync measures a transformed predicted offset. A method can improve one without improving the others. Dynamic Degree records whether motion exceeds an optical-flow criterion; excessive motion or drift may also increase it. Naturalness is a handcrafted no-reference diagnostic and is not an independently calibrated probability of realistic video. Neither integer MLLM judgments nor thresholded event-count metrics satisfy the smooth scalar-functional assumption automatically, so the theoretical output bound must not be interpreted as a bound on every table column.

The seed protocol in Appendix A applies alongside the fixed batch size and hardware/offload policy in Table 5. Timing dispersion in the hyperparameter panel describes runtime variation, while the terminal-refinement entries summarize quality across the aggregation seeds. Neither quantity is a confidence interval for a quality difference.