

# TalkingMachines: Real-Time Audio-Driven FaceTime-Style Video via Autoregressive Diffusion Models

Chetwin Low Weimin Wang

Character AI

{chetwinlow, weiminwang}@character.ai

arXiv:2506.03099v1 [cs.SD] 3 Jun 2025

Abstract

In this paper, we present TalkingMachines—an efficient framework that transforms pretrained video generation models into real-time, audio-driven character animators. TalkingMachines enables natural conversational experiences by integrating an audio large language model (LLM) with our video generation foundation model. Our primary contributions include: (1) We adapt a pretrained SOTA image-to-video DiT into an audio-driven avatar generation model of 18 billion parameters; (2) We enable infinite video streaming without error accumulation through asymmetric knowledge distillation from a bidirectional teacher model into a sparse causal, autoregressive student model; (3) We design a high-throughput, low-latency inference pipeline incorporating several key engineering optimizations such as: (a) disaggregation of the DiT and VAE decoder across separate devices, (b) efficient overlap of inter-device communication and computation using CUDA streams, (c) elimination of redundant recomputations to maximize frame-generation throughput. Please see demo videos here- <https://aaxwaz.github.io/TalkingMachines/>

Figure 1: TalkingMachines provides a framework to generate highly dynamic, immersive FaceTime

experiences based on different character styles.

Equal contribution.

## 1 Introduction

The emergence of video foundation models has revolutionized content creation, enabling high-quality video generation from text prompts and images. However, existing models face significant challenges

when applied to real-time interactive applications like video conferencing. The primary limitation

is their reliance on bidirectional attention mechanisms, which require processing the entire video sequence all at once—where each individual frame relies on future frames. This architectural constraint introduces prohibitive latency for streaming applications.

In this work, we present TalkingMachines, a novel approach that transforms a pretrained bidirectional

video diffusion model into an efficient autoregressive system capable of real-time, audio-driven character animation. Our method addresses three key challenges:

1. Adaptation of pretrained models: We leverage the knowledge embedded in large-scale video foundation models while adapting them for audio-driven generation through specialized attention mechanisms and training strategies.

2. Infinite-length generation: Through our modified Distribution Matching Distillation (DMD) [29,

- 28, 30] approach, we enable autoregressive generation without the error accumulation typically

associated with sequential predictions, allowing for arbitrarily long video streams.

3. Real-time performance: By distilling the model to 2 diffusion steps and implementing system level optimizations, we achieve real-time latency suitable for interactive applications.

## 2 Background

### 2.1 Flow Matching Models

In our framework, we leverage a pretrained image-to-video (I2V) model from WAN 2.1 [25], which

was originally trained using the flow matching paradigm. We adapt this model for our TIA2V (Text-Image-Audio to Video) generation task by introducing audio as an additional conditioning signal alongside the existing image and text inputs.

Flow matching provides a stable and efficient alternative to traditional stochastic diffusion training by

modeling deterministic trajectories in latent space via velocity prediction. Rather than sampling with

stochastic differential equations, flow matching treats generation as an ordinary differential equation

(ODE) integration problem, enabling more stable training and inference.

During fine-tuning, given a target latent  $x_1$  (i.e., a video latent from ground truth frames), we sample

a Gaussian noise latent  $x_0 \sim N(0, I)$  and a timestep  $t \in [0, 1]$  from a logit-normal distribution.

We then compute an intermediate latent using a linear interpolation, following the Rectified Flow

formulation [10]:

$$x_t = tx_1 + (1-t)x_0$$

The ground truth velocity is simply the difference between target and noise latents:

$$v_t = dx_t$$

$$dt = x_1 - x_0$$

(1)

(2)

We condition the model on three modalities: text embedding  $ctxt$ , an initial reference image  $cimg$ , and audio embeddings  $caud$ . The model is trained to predict the velocity field from this conditioning.

The training objective becomes:

$$LFM = \mathbb{E}_{x_0, x_1, t} \|u(x_t, ctxt, cimg, caud, t; \theta) - (x_1 - x_0)\|_2$$

(3)

where  $u(x_t, ctxt, cimg, caud, t; \theta)$  denotes the predicted velocity from the model. This adaptation retains the benefits of flow matching—such as training stability and efficient sampling—while extending the model’s expressiveness to multimodal conditioning. By incorporating temporally aligned audio features into the pretrained I2V architecture, we enable high-quality, lip-synced video

generation driven by real-time conversational audio inputs.

2

### 2.2 Distribution Matching Distillation

Distribution Matching Distillation (DMD) addresses the computational burden of multi-step sampling

in diffusion models by training efficient few-step generators. Unlike traditional distillation

methods

that preserve individual sampling trajectories, DMD operates at the distribution level, enabling greater

f

lexibility in student model design.

The fundamental principle behind DMD involves minimizing the reverse KL divergence between dis

tributions generated by a multi-step teacher model and a few-step student generator. The optimization

objective can be expressed through the reverse KL divergence:

$$\text{LDMD} = \mathbb{E}_t[\text{KL}(\text{p}_{\text{gen}}, t \mid \text{p}_{\text{data}}, t)]$$

The gradient of this objective admits a tractable approximation:

$$\nabla_{\phi} \text{LDMD} \approx \mathbb{E}_t[\nabla_{\phi} \text{KL}(\text{p}_{\text{gen}}, t \mid \text{p}_{\text{data}}, t)]$$

$$\approx -\mathbb{E}_t(\text{sdata}(\Psi(\text{G}_{\phi}(\cdot), t), t) - \text{sgen}(\Psi(\text{G}_{\phi}(\cdot), t), t)) \partial \text{G}_{\phi}(\cdot))$$

(4)

(5)

$\partial \phi$  d

(6)

where  $\text{G}_{\phi}$  denotes the student generator parameterized by  $\phi$ ,  $\Psi$  represents the forward diffusion process, and  $\text{sdata}, \text{sgen}$  are score functions estimated from the data and generated distributions respectively.

### 3 Related Work

#### 3.1 Video Foundation Models

Recent advances in video generation have been driven by diffusion-based approaches, particularly

those using Transformer architectures. State-of-the-art models [2, 11, 25, 14, 20] have demonstrated

impressive generation quality and visual fidelity. However, these models are primarily designed for

offline generation, where the entire video sequence is processed at once. The bidirectional attention

mechanisms employed in these architectures, while beneficial for maintaining temporal coherence

and visual quality, introduce prohibitive latency for real-time streaming applications where frames

must be generated on-demand.

#### 3.2 Audio-Driven Animation

Previous work on audio-driven facial animation has explored various approaches, from 3D morphable

models to GAN-based methods. Recent diffusion-based approaches like EMO[22] and Hallo3[6] have shown promise but still suffer from high latency and limited style diversity. Our work builds on

these foundations while addressing their limitations for real-time applications.

#### 3.3 Real-time Talking Head Generation

Recent real-time talking head generation methods, such as VASA-1 [26] and MegaPortraits [9], employ a common architecture centered around a disentangled facial latent space. These

approaches

typically use an appearance encoder to extract static identity features, and a motion encoder or predictor to represent facial dynamics as a low-dimensional temporal sequence—often compact 1D

vectors. At inference time, a lightweight generative model predicts the motion sequence from audio

and expression cues, which is then combined with the static appearance embedding via a decoder to

synthesize realistic talking head animations in real time.

The computational efficiency of these methods enables real-time generation on consumer-grade GPUs without requiring advanced optimization techniques. However, their architectural

assumptions

inherently limit generality. Specifically, the reliance on rigid and non-rigid 3D warping tailored for faces makes these models difficult to extend to full-body avatars, non-realistic videos such as animations or general-purpose video synthesis. Additionally, the compactness of the motion representation, while efficient, constrains the model’s ability to capture complex or large-scale motion

beyond head and facial dynamics.

3

In contrast, our approach explores the use of large-scale Diffusion Transformers, which offer strong

generative capabilities. To meet the demands of real-time performance, we complement these models

with sparse causal attention and targeted engineering optimizations, enabling synthesis beyond facial

domains while preserving responsiveness.

### 3.4 Model Distillation

Knowledge distillation has emerged as a powerful technique for accelerating diffusion models.

Progressive distillation and consistency models have shown success in reducing sampling steps.

DMD offers unique advantages by allowing architectural flexibility between teacher and student models, which we exploit to enable the bidirectional-to-causal transformation.

## 4 Method

### 4.1 Model Architecture

Our model architecture builds upon the WAN2.1 [25] 14B I2V model, which supports video genera

tion from a ground truth image and text prompt. We extend this with:

**Audio Cross Attention with Attention Mask:** We modified the Cross Attention layer to include support for the input of audio sequence. Basically within each Cross Attention layer, we initialize new layers to generate keys and values from the input audio tokens, which are then cross-attended to

the queries of the corresponding latent frame tokens. Similar to MagicInfinite [27], we employ local

attention masks that focus on facial regions during audio cross-attention. This improves lip-sync accuracy by reducing interference from non-facial tokens such as noise and background visual signals.

**Audio Projection Layer:** A 1.2B parameters audio module that processes raw audio token em

beddings before passing into cross-attention layers. Inspired by Hallo3[6], the audio tokens are temporally aligned with video frames using a local window-based approach: each latent frame will attend only to the audio tokens within a window of five latent frames centered around itself. Considering that speech-driven motion—such as lip movement, head gestures, and body dynamics—is predominantly influenced by temporally local audio features, this localized attention helps the model to learn audio-visual correlations faster and better.

**Speaking/Silence Mode:** For each face of the training clip, we detect whether it is actually in Speaking or Silence mode. When it is in Silence mode, we will use a special embedding (e.g. zero embedding) instead of the original audio embedding to represent the status, and pass that for cross attention. We are able to use a lipsyncing evaluation model such as SyncNet[5] to detect the (face, audio) similarity score (i.e. conf). For videos that have multiple faces, we will select the face that has the highest SyncNet score as the face in Speaking mode, and the rest in Silence mode. We also manually filter the training data in Silence mode to ensure there are no dubbed cases. During inference, we will enable the Silence mode for the given character by passing in the corresponding special embedding when it is the character’s listening phase. This also makes it possible for Multi

Character generation cases- where there are multiple faces within the video being generated, and we are able to switch the modes among them depending on the turns of speech.

#### 4.2 Asymmetric Distillation with Modified CausVid

The core innovation of our approach is extending and modifying the CausVid [30] framework, which distills the knowledge of a bidirectional teacher model into an autoregressive student model. Similar to CausVid, to reduce first frame latency, we break up the original context length of our bidirectional teacher (21 latent frames) into smaller chunks, maintaining bidirectional attention within chunks, but introducing causal attention masking across chunks, enabling the autoregressive student to generate shorter segments of video frames sequentially.

However, our approach diverges from CausVid in several key aspects. CausVid employs an approach akin to Diffusion Forcing [4], where each chunk is assigned an independent randomly sampled timestep of gaussian noise. As a result, chunks attend to preceding chunks that may be at different timesteps of noise. In contrast, we observe that few-step generators typically exhibit a piecewise flow trajectory. Allowing chunks to attend to other chunks of mismatched timesteps would be

suboptimal.

To address this, we maintain the sampled timestep as a global attribute shared across all chunks

4

within a video, aligning more closely with the pre-training set up. Furthermore, we observe that speech-driven motion only requires a small temporal attention window to have smooth temporary

continuity, as such we designed the following sparse causal attention masking, effectively reducing

inference memory requirements by requiring a smaller KV cache and also faster inference due to small attention KV context length, without any quality degradation.

For each 81-frame video clip (21 latent frames after VAE encoding), we:

1. Chunked Processing: Divide the latent sequence into 7 chunks of 3 latent frames each
2. Sparse Attention Pattern: Each token in chunk  $c_t$  attends to:
  - All tokens within the current chunk  $c_t$
  - All tokens in the previous chunk  $c_{t-1}$
  - All tokens in the starting chunk  $c_0$  (containing the ground truth image)

This attention pattern ensures temporal continuity while preventing error accumulation, as the model

always has access to the clean reference image in  $c_0$ .

The mathematical formulation of our sparse attention can be expressed as:

$\text{Attention}(Q_t, K, V) = \frac{\exp(Q_t K^T)}{\sum \exp(Q_t K^T)}$

$\sqrt{d_V}$

where  $K$  and  $V$  include tokens from  $\{c_0, c_{t-1}, c_t\}$  only.

(7)

We employ DMD to distill our model from 24 NFEs ( $12 \times 2$  with CFG) to just 2 NFEs. The key modifications include the following (please refer to Figure 2 for illustrated workflow).

1. Mixed Training Data: We combine real training clips with synthetic samples generated by the student model using a (image, audio) pair in the sparse causal attention, improving generalization to unseen styles. We do this in progressive training fashion- we start with only real training data until the student model converges to some extent, before we leverage it to generate synthetic samples on the fly during training.
2. Sparse Causal Attention: Each chunk only attends to current chunk  $c_t$  (bidirectional within current chunk), previous chunk  $c_{t-1}$ , and the starting chunk  $c_0$ .
3. Regression Loss without GAN: We added regression loss on top of student model's prediction in addition to DMD loss to stabilize the training.

#### 4.3 System Optimizations

**Score-VAE Disaggregation:** Traditionally, latent diffusion models (LDMs)[18] exhibit significantly

higher computational latency during the denoising process compared to the subsequent VAE [13] decoding stage—often to the extent that decoding latency is considered negligible. However, with

the growing adoption of transformer-based[24] architectures as the standard backbone for diffusion

models, this assumption no longer holds. The performance gap between the diffusion model and the VAE is narrowing, owing to the proliferation of hardware- and software-level optimizations

for transformers, including fast attention kernels[8, 7] and fused MLP operations. Furthermore, in the context of real-time generation, where latency constraints are stringent, any inefficiency is non-trivial, motivating our Score-VAE disaggregation server design to support real-time generation. In a single-GPU configuration, VAE decoding and device-to-host (d2h) transfer of the output tensor account for approximately 30% of the end-to-end generation time per video chunk. This bottleneck is further amplified when employing Sequence Parallelism (SP)[12] to distribute the diffusion computation across multiple GPUs. As the diffusion workload becomes increasingly parallelized, the relative cost of VAE decoding dominates, ultimately limiting scalability and becoming the primary obstacle to real-time streaming performance.

To evaluate whether a system meets real-time requirements, we define a key success criterion: the time taken to generate a sequence of video frames must be consistently less than the duration of the video sequence itself. To measure this, we introduce the Time Taken Between Chunks (TTBC) as a metric for comparison across different server configurations (illustrated in Figure 3). In Cases 1 and 2, where self-contained model servers are used (i.e., each GPU independently performs both diffusion

5

Figure 2: Overview of the DMD training workflow for TalkingMachines. The diagram illustrates the asymmetric distillation process where a bidirectional teacher model is distilled into an autoregressive student model. The training incorporates mixed data generation with synthetic samples from the student model, sparse causal attention patterns across chunks, and a combination of DMD loss with

regression loss for stable convergence.

and decoding), we observe diminishing returns when scaling from 1 to 2 GPUs. The improvement in

TTBC is minimal, and neither configuration achieves the latency required for real-time generation. In

contrast, Case 3 implements a 2-GPU disaggregated server, where one GPU is dedicated to diffusion

(worker) and the other to VAE decoding (master). This design allows the worker to maximize diffusion throughput while overlapping decoding operations on the master GPU. Even without distributed diffusion, the 2-GPU disaggregated server outperforms the self-contained server with same number of GPUs in TTBC. Finally, in Case 4, we scale the number of diffusion workers to two

while maintaining a single dedicated decoder. This configuration demonstrates superior scalability, with improved TTBC compared to Case 2. Both Cases 3 and 4 are able to consistently meet the real-time threshold for video generation.

Figure 3: Runtime analysis comparing the latency of various server designs like a simple self contained server, and our Score-VAE disaggregation server, both with and without Sequence Parallelism.

6

Efficient computation-communication overlap with CUDA streams: The disaggregation server design introduces additional inter-device communication, as the master rank must broadcast the inputs to, and receive outputs from, the workers ranks. We leverage NCCL collectives to manage these data transfers efficiently. Crucially, we make use of additional CUDA[16] streams to overcome

the blocking nature of these NCCL collectives and ensuring the VAE decoding can be done in parallel

with these collectives, thereby significantly improving inference throughput.

KV & Embedding Cachings: During inference, we cache all key-value pairs for chunk  $ct-1$  and  $c0$ , for each timestep, over all transformer blocks, eliminating redundant computations.

Other

embeddings such as timestep embedding, guidance embedding, and context embedding are also cached to improve inference speed.

## 5 Experiments

### 5.1 Datasets

We prepare high quality, human centric videos containing 1.5M clips that are first split by scene detection [3], and then are filtered based on aesthetics[19] and motion scoring[21], as well as using

OCR[1] model to filter out videos that contain too much text. The clips are with minimum 4s in duration, and contain speakers of diverse poses and distances to camera. The majority of the clips are

focused on a single speaker, with a small percentage containing multiple speakers.

Our model’s target resolution is 512x512, and supports diverse aspect ratios of input video frames that

fall within the total area of 512x512. Using a lower resolution than the original pretrained model’s

resolution of 480x832 also makes inference faster as it reduces the context length of the transformer

blocks, which reduces the burden for streaming. Based on our internal evaluation, 512x512 works

well for both desktop and mobile-based devices.

### 5.2 Training Setup

Our end to end training is split into three stages. 1) WAN 2.1 pretrained model warm-up. 2)

Audio

pretraining. 3) Distillation and autoregressive training. Our training starts with WAN2.1 I2V 480P,

which has the original resolution of 480x832, and larger than our target resolution of 512x512.



- Stage 1 Warmup:— Total training steps: 9k— GPUs: 128 H100 for 1 day— Target: adapting the pretrained model to our new resolution as well as human centric training data.— Model weights: Learning all layers without freezing
- Stage 2 Audio Pretraining:— Total training steps: 30k— GPUs: 384 H100 for 5 days— Target: Learning the new audio layers and lip-syncing capabilities.— Model weights: Initializing new audio layers, and freezing all non-audio parameters.
- Stage 3 Sparse Autoregressive Distillation:— Total training steps: 20k— GPUs: 128 H100 for 10 days— Target: Distilling the model into 2 diffusion steps, with causal sparse attention.— Model weights: Learning all layers without freezing.

### 5.3 Training Infrastructure and Strategies

As described in section 5.2, we choose 512x512 as the target resolution not only for real-time generation, but also faster training iterations. At this resolution, the DiT model parameters, gradients and optimizer states consume approximately 204GB of GPU memory in bfloat16, while activations

7

reaching up to several terabytes of GPU memory. Through activation checkpointing and parameter sharding, we are able to fit the peak activation memory within each GPU, eliminating the need for

model parallelism and enabling training with data-parallel variants alone.

In addition to the DiT backbone, training requires several auxiliary models—namely, the VAE, and

Text, Image, and Audio encoders—to produce the necessary embeddings. While each of these modules is smaller than DiT, they still contribute significant memory overhead. To optimize memory

allocation for activations, we employ DeepSpeed ZeRO Stage 3 [17] to shard the encoder modules.

This strategy frees over 20GB of GPU memory with negligible impact on per-step performance, striking a favorable trade-off between embedding generation time and activation capacity.

Stage 1 and Stage 2 training of the DiT model are conducted using DeepSpeed ZeRO Stage 2 on NVIDIA H100 GPUs, leveraging GPUDirect-TCPX for high-speed interconnect. This configuration

provides a strong baseline for efficient scaling under data-parallel training.

In Stage 3, corresponding to the DMD training, the training setup requires three full copies of the diffusion model to reside in GPU memory—two of which are trainable—totalling 450GB of GPU

memory, thereby necessitating a more aggressive memory management strategy. Initial attempts using ZeRO Stage 3 are found to scale poorly as GPU counts increase, primarily due to its global parameter sharding, which introduces substantial cross-device communication overhead.

To address this bottleneck and improve throughput, we implement Hybrid Sharded Data Parallel (HSDP) with PyTorch FSDP[31] API. This strategy enables parameter, gradient, and optimizer states

sharding within localized GPU groups, thereby significantly reducing cross-node communication and

improving memory efficiency.

Additionally, experiments are conducted on a cluster equipped with NVIDIA H200 GPUs and InfiniBand. We observe a 2x speed up in per-step training time, primarily due to faster inter-node bandwidth and the ability to fit all relevant tensors within a single node, effectively eliminating the

overhead of cross-node sharding.

Scaling training to a 720×720 resolution increases the total activation memory footprint by approximately

3–4×, which exceeds the capacity of a single GPU even with activation checkpointing enabled.

Since both ZeRO and FSDP are fundamentally data-parallel approaches, no amount of sharding will

reduce activations.

To address this, we rely on Sequence Parallelism (SP)[12], a form of intra-layer parallelism that partition sequences across devices, hence reducing per-GPU computation. In our implementation, we

integrate SP as part of a 2D parallelism strategy, where we shard model parameters, gradients and

optimizer states (via ZeRO/HSDP), while splitting the computation and activations further within a

smaller set of GPUs.

#### 5.4 Distillation Ablations

To better understand the trade-offs between chunk size and diffusion steps in our distillation process,

we conduct an ablation study using a 2×2 factorial design. Specifically, we vary the chunk size (3 vs.

7) and the number of diffusion steps (2 vs. 4), and evaluate performance using perceptual metrics and GPU efficiency, based on an internal video dataset containing mainly realistic videos of talking

persons. The metrics include Fréchet Video Distance (FVD) [23] and two variants of SyncNet-based

lip-sync confidence: Sync-C and Sync-D [5].

Chunk Size Steps FVD↓ Sync-C↑ Sync-D↓ ScoreGPUs VAEGPUs

7

4

7

3

3

2

4

2

208.2

235.1

215.4

238.1

6.28

6.13

5.89  
 5.83  
 8.94  
 8.77  
 8.55  
 8.73  
 4  
 2  
 2  
 1  
 1  
 1  
 1  
 1

Table 1: Ablation study of distillation setups with varying chunk sizes and diffusion steps based on our internal eval dataset. GPU columns indicate the number of H100 GPUs used for the score model and the VAE decoder respectively.

8  
 Overall, we observe that the lip-sync quality (as indicated by Sync-C and Sync-D) remains relatively consistent across different settings, with only minor fluctuations. This suggests that all configurations offer robust lip-sync performance. The perceptual quality, as measured by FVD, shows a slight degradation when both the chunk size is reduced from 7 to 3 and the number of diffusion steps is reduced from 4 to 2. However, this degradation is modest. What is particularly notable is the trade-off in computational cost. The most compute-efficient setting (chunk size 3 with 2 diffusion steps) requires only 1 H100 GPU for the score model, compared to 4 H100s in the highest quality setting. This results in a significant reduction in hardware cost and energy consumption, while still delivering acceptable generation quality. Thus, one may choose a configuration based on their specific compute budget and quality expectations.

If resource constraints are critical, the 3×2 setup provides a compelling balance between performance and efficiency; if maximum perceptual fidelity is required, the 7×4 setup is preferred.

## 6 Applications

We demonstrate a real-time FaceTime-style application that integrates TalkingMachines with audio large language models (LLMs) to showcase the practical deployment of our system in interactive video communication scenarios.

### 6.1 System Architecture

Our demonstration system consists of three main components:

- Audio LLM Integration: We integrate mainstream audio LLMs that generate real-time spoken responses, enabling natural conversational interactions with users.

- Video Generation Server: Our TalkingMachines model is deployed on a cloud server with H100 GPUs, where the score model and VAE decoder operate on separate GPU resources to generate synchronized lip-synced animations from audio input.
- WebRTCStreaming: We utilize LiveKit [15], a cloud-based WebRTC service, to handle real-time video streaming and client connections.

## 6.2 System Workflow

The system operates through a distributed pipeline where user audio is captured via the web interface, processed by the audio LLM to generate conversational responses, and forwarded to our video generation server. The generated video frames are synchronized with audio and streamed back to clients through the WebRTC service, enabling real-time interactive conversations with AI-generated avatars.

## 6.3 Demonstration Results

Our implementation successfully achieves real-time performance suitable for interactive video calls, with users able to access the system through standard web browsers across desktop and mobile devices. The system demonstrates the practical viability of deploying advanced video generation models in real-time communication applications, establishing a foundation for AI-powered interactive media experiences.

## 7 Conclusion, Limitations and Future Work

In this work, we demonstrate how audio-driven animation and sparse causal distillation can be effectively applied to a powerful pretrained video generation model, transforming it into a streaming capable network that supports real-time, infinite-length FaceTime-style video generation. The resulting model is capable of animating images of arbitrary styles—including photorealistic, anime, and 3D avatars—with natural, high-quality lip-synced speech, when paired with mainstream audio large language models (LLMs).

9

We also detail the system-level engineering efforts required to reduce computational bottlenecks in the context of real-time streaming, including optimizations in GPU allocation, communication computation overlap, and memory reuse. These design choices are critical for enabling the model to

run with minimal latency in practical deployment scenarios.

Our training methodology is designed to strike a strong balance between leveraging the generalization

capabilities of the pretrained model and extending it with new audio-driven behaviors. Notably, our adaptation introduces significant capabilities—such as expressive lip synchronization and dynamic motion range—using only a modest amount of paired audio-video training data and compute resources.

Despite these advantages, our current approach has limitations. The audio conditioning components

were introduced only in the later stages of training, meaning that the pretrained model did not benefit

from large-scale audio-video supervision during its initial learning. The large audio projection layers, while effective, were trained on a relatively small subset of data with limited iterations. This

bottleneck constrains the model’s scalability and expressiveness in more diverse or challenging audio-driven scenarios.

Looking forward, it would be promising to explore large-scale pretraining strategies that incorporate

audio conditions earlier in the process. In particular, joint modeling of video and audio from massive

paired datasets may enable even stronger multimodal representations, improved lip-sync fidelity, and

more robust performance across domains and languages.

#### Project Contributions

- Project Leader: Weimin Wang
- Modeling & Deployment: Chetwin Low, Weimin Wang
- Data: Chetwin Low, Weimin Wang, Ryan Vilim, Diego De La Torre, Calder Katyal, Zhiwei Jia, Michal Wolski
- WebRTC(LiveKit): Yi Cui

#### References

[1] PaddlePaddle Authors. Paddleocr, awesome multilingual ocr toolkits based on paddlepaddle. <https://github.com/PaddlePaddle/PaddleOCR>, 2020.

[2] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe

Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators, 2024. OpenAI Technical Report.

[3] Brandon Castellano. Pyscenedetect: Python and opencv-based scene cut/transition detection program & library.

[4] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

[5] Joon Son Chung and Andrew Zisserman. Out of time: automated lip sync in the wild. In *Computer Vision–ACCV 2016 Workshops: ACCV 2016 International Workshops, Taipei, Taiwan*,

November 20-24, 2016, Revised Selected Papers, Part II 13, pages 251–263. Springer, 2017.

[6] Jiahao Cui, Hui Li, Yun Zhan, Hanlin Shang, Kaihui Cheng, Yuqi Ma, Shan Mu, Hang Zhou, Jingdong Wang, and Siyu Zhu. Hallo3: Highly dynamic and realistic portrait image animation with video diffusion transformer. *arXiv preprint arXiv:2412.00733*, 2024.

[7] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

[8] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

10

[9] Nikita Drobyshhev, Jenya Chelishhev, Taras Khakhulin, Aleksei Ivakhnenko, Victor

Lempitsky,  
and Egor Zakharov. Megaportraits: One-shot megapixel neural head avatars, 2023.

[10] Patrick Esser, Robin Rombach, Andreas Blattmann, and Björn Ommer. Rectified flow matching: Towards fast and stable training of continuous normalizing flows. arXiv preprint arXiv:2402.14851, 2024.

[11] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. arXiv preprint arXiv:2205.15868, 2022.

[12] Sam Ade Jacobs, Masahiro Tanaka, Chengming Zhang, Minjia Zhang, Shuaiwen Leon Song, Samyam Rajbhandari, and Yuxiong He. Deepspeed ulysses: System optimizations for enabling training of extreme long sequence transformer models. arXiv preprint arXiv:2309.14509, 2023.

[13] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. Foundations and Trends® in Machine Learning, 12(4):307–392, 2019.

[14] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. arXiv preprint arXiv:2412.03603, 2024.

[15] LiveKit. Livekit: Open-source webrtc infrastructure for real-time audio and video, 2025. Accessed: 2025-05-27.

[16] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with cuda: Is cuda the parallel programming model that application developers have been waiting for? Queue, 6(2):40–53, 2008.

[17] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models, 2020.

[18] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[19] Christoph Schuhmann, Andreas Biewald, Aran Komatsuzaki, et al. Laion-aesthetics: Predicting the aesthetic quality of images, 2022. Accessed: 2025-05-27.

[20] Genmo Team. Mochi 1. <https://github.com/genmoai/models>, 2024.

[21] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16, pages 402–419. Springer, 2020.

[22] Linrui Tian, Qi Wang, Bang Zhang, and Liefeng Bo. Emo: Emote portrait alive generating expressive portrait videos with audio2video diffusion model under weak conditions. In European Conference on Computer Vision, pages 244–260. Springer, 2024.

[23] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. In arXiv preprint arXiv:1812.01717, 2019.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez,

\_ukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[25] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu,

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

[26] Sicheng Xu, Guojun Chen, Yu-Xiao Guo, Jiaolong Yang, Chong Li, Zhenyu Zang, Yizhong Zhang, Xin Tong, and Baining Guo. Vasa-1: Lifelike audio-driven talking faces generated in real time, 2024.

11

[27] Hongwei Yi, Tian Ye, Shitong Shao, Xuancheng Yang, Jiantong Zhao, Hanzhong Guo, Terrance

Wang, Qingyu Yin, Zeke Xie, Lei Zhu, et al. Magicinfinite: Generating infinite talking videos with your words and voice. *arXiv preprint arXiv:2503.05978*, 2025.

[28] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and

Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024.

[29] 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*, pages 6613–6623, 2024.

[30] Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman,

and Xun Huang. From slow bidirectional to fast causal video generators. *arXiv preprint arXiv:2412.07772*, 2024.

[31] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright,

Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. Pytorch fsdp: Experiences on scaling fully sharded data parallel, 2023.