# Matrix-Game 2.0: An Open-Source, Real-Time, and Streaming Interactive World Model


## Abstract

Recent advances in interactive video generations have demonstrated diffusion model's potential as world models by capturing complex physical dynamics and interactive behaviors. However, existing interactive world models depend on bidirectional attention and lengthy inference steps, severely limiting real-time performance. Consequently, they are hard to simulate real-world dynamics, where outcomes must update instantaneously based on historical context and current actions. To address this, we present Matrix-Game 2.0, an interactive world model generates long videos on-the-fly via few-step auto-regressive diffusion. Our framework consists of three key components: (1) A scalable data production pipeline for Unreal Engine and GTA5 environments to effectively produce massive amounts ($\sim$ 1350 hours) of interactive video data; (2) An action injection module that enables frame-level mouse and keyboard input as interactions; (3) A few-step distillation for real-time, streaming video generation. Matrix-Game 2.0 can generate high-quality minute-level videos across diverse scenes at an ultra-fast speed of 25 FPS. We open-source our model weights, codebase, and data production pipeline to advance research in interactive world modeling.


## 1 Introduction

World models [9, 18, 25] have gained significant attention due to their capability to understand real-world interactions and predict future states or behaviors [41]. By enabling intelligent agents to perceive their surroundings and respond to actions, these models reduce the cost of real-world trials and facilitate interactive simulation. Consequently, world models show great promise in fields such as game engines [8, 23, 35], autonomous driving, and spatial intelligence [2, 40].

Recent advances in video generation models [3, 4, 16, 29, 37] have shown remarkable progress in learning knowledge from large-scale real-world datasets, ranging from physical laws to interactive scenes. This demonstrates their huge potential to serve as world models. Among the various research directions in this domain, interactive long video generation [4, 13] has become increasingly important due to its practical applications, where long videos must be generated in real-time in response to a continuous stream of user input. Specifically, when conditioned on user actions like camera movements and keyboard inputs, the model generates frames progressively, enabling real-time user interaction.

Despite impressive progress in interactive video generation [7–9, 21, 23], existing methods suffer from several significant challenges:

• Lack of large-scale, high-quality interactive video datasets with rich annotations for training, such as accurate actions and camera dynamics, due to the high cost and difficulty of collection.

• Latency issues with bidirectional video diffusion models [18, 21, 52], where generating a single frame requires processing the entire video. This makes them unsuitable for real-time, streaming applications where the model must adapt to dynamic user commands and produce frames on the fly. The quadratic scaling of compute and memory requirements with respect to frame length, along with


Figure 1: **Real-time Interactive Generation Results.** We introduce Matrix-Game 2.0, a real-time interactive video generation model. By integrating action modules and few-step distillation, it can auto-regressively produce high-quality interactive videos given an input image in 25 FPS. The demonstrated results cover various scenes and diverse styles, demonstrating its powerful generation capabilities.

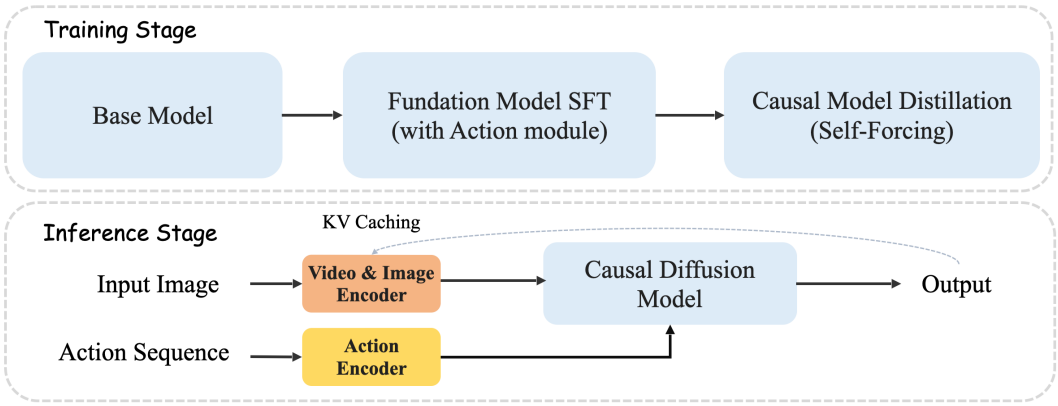



Figure 2: **Pipelines of Matrix-Game 2.0.**

the high number of denoising iterations, makes long video generation computationally intensive and economically impractical.

• Error accumulation in existing auto-regressive video diffusion models. While these models generate the next frame based on previous frames, they often suffer from error accumulation during generation, leading to degraded video quality over time.

To address these critical challenges in real-time interactive generation, we present Matrix-Game 2.0 - a novel framework specifically designed to achieve both real-time performance and robust generalization across diverse scenarios. First, our technical core features a video diffusion transformer with integrated action control, distilled into a causal auto-regressive model via Self-Forcing-based techniques. This architecture supports both training and inference through an efficient KV caching mechanism, achieving 25 FPS generation on a single H800 GPU while maintaining minute-long temporal consistency and precise action controllability - even in complex wild scenes beyond the training distribution.

The model's strong generalization capability is enabled by another innovation of ours: a comprehensive data production pipeline that solves fundamental limitations in interactive training data. The pipeline is based on Unreal Engine, including a Navigation Mesh-based Path Planning System for data diversity and Quaternion Precision Optimization modules for accurate camera control. Moreover, for Grand Theft Auto V(GTA5) environment, we developed a data recording system using Script Hook integration, which enables synchronized capture of visual content with corresponding user interactions. Together, these components produce large-scale datasets with frame-level annotations, addressing two critical needs: (1) precise alignment between visual content and control signals, and (2) effective modeling of dynamic in-game interactions.

By simultaneously tackling the challenges of efficiency and controllability, Matrix-Game 2.0 makes significant strides in world modeling by introducing an efficient framework tailored for real-time simulation and interaction. To support continued progress in this area, we will release the code and weights of the pre-trained models, along with the code for our data production pipeline.

## 2 Related Work

### 2.1 Controllable Video Generation

With the rapid advancement of diffusion models [12, 30, 32], significant progress has been made in visual content generation for videos [3, 10, 20, 34, 45, 53, 55]. Most recent approaches have transitioned toward bidirectional attention Transformer-based architectures [24, 36] or autoregressive models [13, 48], enabling modern video diffusion models to synthesize high-quality, temporally coherent, and substantially longer videos. This rapid evolution of video generation models has further driven the development of world models that leverage video diffusion techniques to implicitly learn physical laws, object dynamics, and causal relationships from raw video data [1, 22, 23, 42, 43] for complex environment simulation.

Controllable video generation serves as a core component of world simulation. Generally, control signals span multiple modalities and can be categorized into scene controllability and action controllability. Extensive prior work has explored scene controllability, including [10, 16, 28, 38], which leverages text, images, or 3D scene priors to regulate the scenes in generated videos. Beyond scene control, action controllability, achieved through camera angles [44] or trajectories [11, 39], has also emerged as a prominent research focus. These efforts have yielded promising advancements in both the visual quality and controllability of generated videos. Certain world model-based approaches [8, 9, 23, 49, 52] further support both scene and action controllability. However, constrained by computational resources and video length limitations, most contemporary models still struggle to achieve real-time video generation.

### 2.2 Long-context Video Generation

Current video generation models are typically constrained to videos of $\leq 10$ seconds due to limited long video training data and prohibitive computational costs. Existing methods for long-context video generation can be broadly categorized into two types: those that combine multiple video segments, and those that employ autoregressive approaches. For multi-segment video generation, a simple yet effective approach is to generate multiple overlapping segments of fixed length [6, 26, 52]. Other works [46, 54] adopt a two-stage pipeline, first generating keyframes and then applying frame interpolation. In contrast, autoregressive models offer a natural advantage for variable-length video generation. For example, methods such as Diffusion Forcing [4], CausVid [48], and Self-Forcing [13] combine autoregressive modeling with diffusion techniques to achieve promising results in long video synthesis. However, these approaches remain largely confined to conventional Text-to-Video (T2V) and Image-to-Video (I2V) tasks, leaving the challenge of generating long, interactive videos unexplored.

### 2.3 Real-Time Video Generation

Diverse approaches currently exist to achieve real-time video generation. The primary methods involve increasing the compression ratio of the VAE, performing knowledge distillation to reduce the number of sampling steps in diffusion models, or combining KV Cache with a causal transformer to autoregressively infer the next frame. LTX-Video [10] achieves generation times shorter than the video duration on an H100 GPU by optimizing VAE compression ratios and applying model distillation techniques [31, 47, 51]. Works such as Next-Frame Diffusion [7], Self-Forcing [13], CausVid [48], and Oasis [8] leverage the characteristics of autoregressive models, combined with knowledge distillation, to enable efficient few-step generation. Although these works achieve real-time video generation, most of them cannot support real-time interaction. While Oasis manages to enable real-time interaction, its visual quality degrades rapidly during the inference of long videos. In our work, we follow the training paradigm of Self-Forcing to allow few-step inference, achieving not only ultra-long video generation but also maintaining stable and consistent frame quality.

## 3 Data Pipeline Development

We design and implement comprehensive data production pipelines to facilitate large-scale training of interactive video generation models. Specifically, our work addresses two key challenges: (1) generating gaming video data precisely aligning with keyboard and camera signal annotations, and (2) enabling interactive video capture mechanisms to better model dynamic in-game interactions. For practical deployment, we develop and curate a diverse dataset production pipeline comprising both static and dynamic scenes sourced from the Unreal Engine and the GTA5 simulation environment.

### 3.1 Unreal Engine-based Data Production

The development of high-performance interactive video generation models requires large-scale datasets featuring precisely synchronized visual content and control signals like precisely aligned keyboard input and camera parameters. While existing datasets often lack accurate temporal alignment between game-play footage and corresponding inputs, our Unreal Engine-based pipeline systematically addresses this gap through controlled synthetic data generation. Unreal Engine's precise

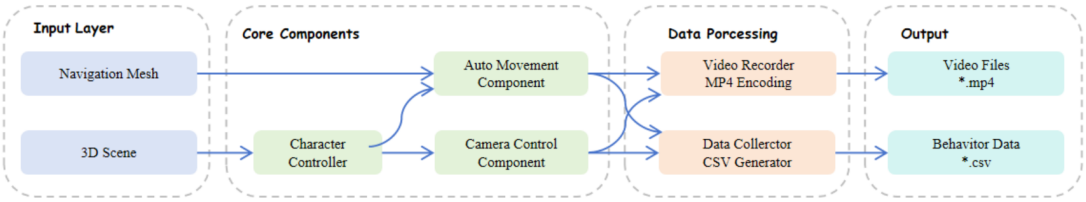



Figure 3: **Overview of Our Data Production Pipeline based on Unreal Engine.**

environmental control and deterministic rendering make it particularly suitable for creating scalable, multi-modal training data with guaranteed annotation accuracy.

### 3.1.1 Framework Design

As illustrated in Figure 3, our Unreal Engine-based data pipeline takes a navigation mesh and a 3D scene as input. The system then employs automated movement and camera control modules to simulate agent navigation and dynamic viewpoint transitions. Finally, the resulting visual data and corresponding action annotations are recorded and exported through an integrated MP4 encoder and CSV generator.

The key innovations of our system comprise: (1) a navigation mesh-based path planning module to enable diverse trajectory generation; (2) a precise system input and camera control mechanism to ensure accurate action and viewpoint alignment; and (3) a structured post-processing pipeline for high-quality data curation. Detailed descriptions of each component are provided below.

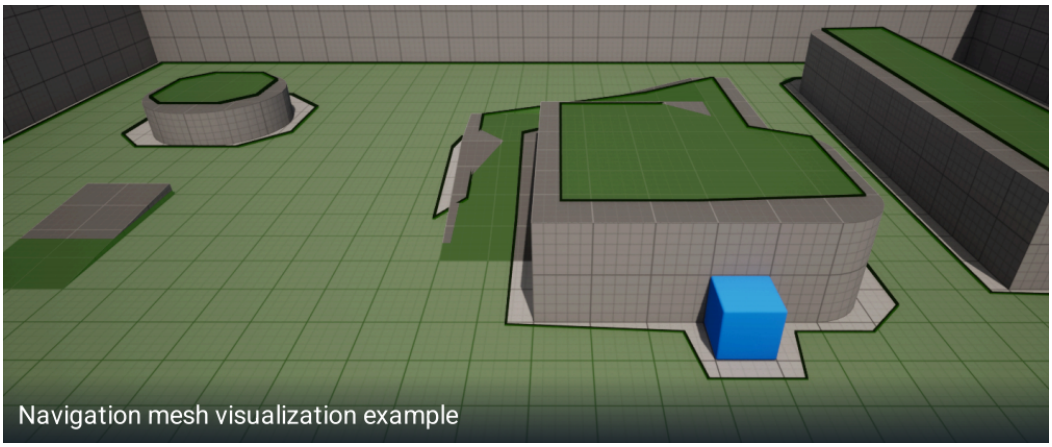



Figure 4: **An example for Our Navigation System.**

**Navigation Mesh-based Path Planning System** To enhance the realism and behavioral diversity of the generated training data, we developed an advanced navigation mesh–based path planning system that facilitates dynamic and adaptive movement of non-player characters (NPCs). This system supports real-time, deterministic path planning, a critical requirement for producing reproducible and high-fidelity training data.

Our implementation builds upon Unreal Engine's native NavMesh infrastructure, augmented with customized path-planning optimizations that reduce the average query latency to less than 2 ms. Furthermore, the system introduces controlled stochasticity in agent behavior, enabling diverse and contextually coherent movement patterns while strictly adhering to logical navigation constraints. This approach substantially enhances the richness of the training corpus by introducing realistic agent interaction dynamics and movement trajectories, thereby improving the generalization capacity of downstream video generation models. A navigation example is shown in Figure 4. The green area in the picture shows the area where the agent can move freely, preventing the agent from hitting the walls and getting stuck.

**Precise Input and Camera Control** We integrated Unreal Engine's Enhanced Input system to enable simultaneous capture of multiple keyboard inputs with millisecond-level precision. The system maintains a synchronized buffer of input events aligned with rendered frames to ensure accurate input–visual synchronization for training:

$$\text{Input}_{\text{frame}_i} = (\{k_1, k_2, ..., k_n\}, \text{timestamp}_i) \tag{1}$$

where each input state $k_j$ represents a specific key press or release event aligned with frame $i$.

To eliminate a critical error rate of 0.2% in camera rotation calculations, we implemented quaternion precision optimization by using double precision arithmetic in intermediate calculations. This optimization reduced rotation errors to a level that is effectively negligible.

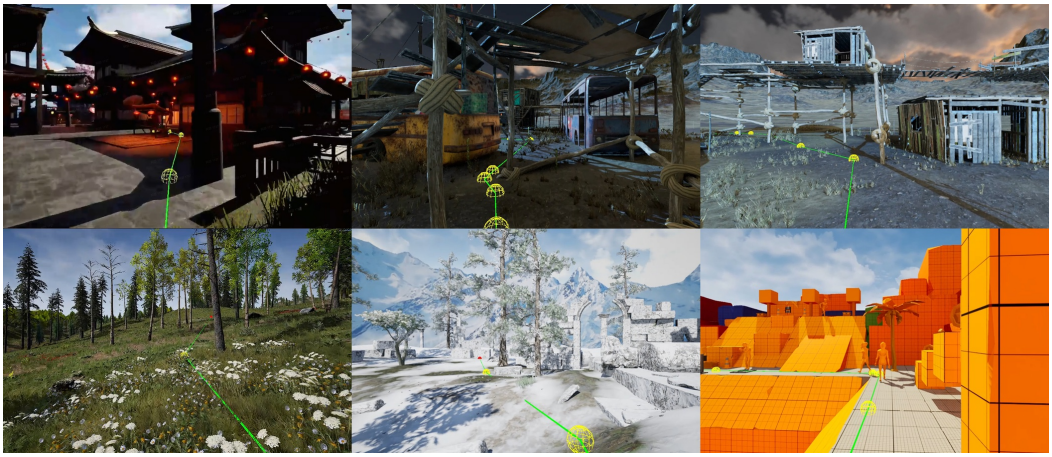


Figure 5: **Trajectory Examples of Collected Unreal Engine Data.**

**Data Curation** We developed a video frame filtering algorithm based on OpenCV to detect and eliminate temporally redundant frames, thereby enhancing data efficiency. A velocity-based validation mechanism was further introduced to identify and exclude invalid samples characterized by zero or negative velocity, which typically indicate stationary or physically implausible motion states:

$$\text{validity} = \begin{cases} 1 & \text{if } ||\vec{v}|| > \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $\vec{v}$ represents the velocity vector and $\epsilon$ is a small positive threshold to account for the precision of floating points. This criterion ensures the retention of only semantically meaningful motion data for subsequent model training.

**Multi-thread Pipeline Accelerating** The data processing pipeline was redesigned to support multi-thread execution, enabling dual-stream data production on a single RTX 3090 GPU. The system employs separate rendering threads in conjunction with shared memory pools for efficient resource utilization. Some representative trajectory examples are illustrated in Figure 5. The green line segments represent the path of the agent. In complex scenarios, reasonable paths can also be planned.

### 3.2 GTA5 Interactive Data Recording System

To facilitate the acquisition of richly interactive dynamic scenes, we developed a comprehensive recording system within GTA5 using Script Hook integration, which enables synchronized capture of visual content with corresponding user actions.

We implemented a custom plugin architecture using Script Hook V to establish a recording pipeline within the GTA5 environment. The plugin simultaneously captures mouse and keyboard operations with frame-accurate synchronization. Each item collected includes the RGB frame and the corresponding mouse and keyboard operations.

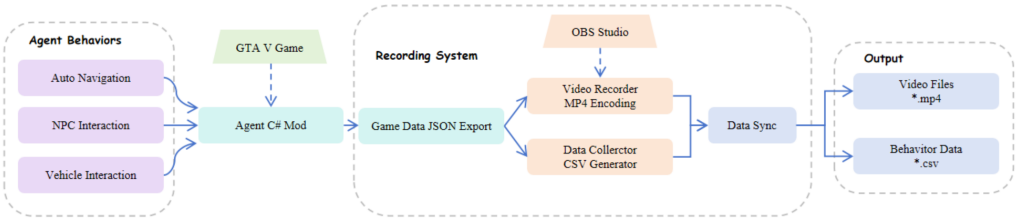



Figure 6: **Overview of Our GTA5 Interactive Data Recording System.**

As illustrated in Figure 6, Our system comprises three main components: Agent Behaviors, GTA V Game Environment, and Recording System. The Agent Behaviors module includes autonomous navigation, NPC interaction, and vehicle interaction capabilities, which are integrated into the GTA V game through a custom C modification. The game exports behavioral data in JSON format to the Recording System, which utilizes OBS Studio for video capture with MP4 encoding and a Data Collector for CSV generation. A synchronization mechanism ensures temporal alignment between video frames and behavioral data, producing synchronized video files (.mp4) and behavioral datasets (.csv) as the final output. dynamic control mechanisms, including autonomous navigation, NPC interaction, and vehicle interaction, can be selectively enabled to generate interactive scenarios from first-person or third-person perspectives. Environmental parameters such as vehicle density, NPC number, weather patterns, and time-of-day settings can be adjusted to simulate a wide variety of dynamic scenarios, enhancing the diversity and realism of the collected data. Specifically, the vehicle density parameter is configurable within the range $[0.1, 2.0]$ , while the NPC density parameter spans the interval $[0.2, 1.5]$.

To obtain an optimal viewpoint during vehicle navigation simulations, the system ensures precise camera alignment through per-tick positional updates, maintaining an optimal and consistent viewpoint relative to the vehicle throughout the simulation:

$$\text{Camera}_{position} = \text{Vehicle}_{position} + \text{offset} \times \text{rotation} \tag{3}$$

Building upon the vehicle dynamics, the system infers and logs the corresponding keyboard inputs, thereby generating a comprehensive and temporally aligned interaction data encompassing velocity, acceleration and steering angle.

Figure 7: **Trajectory Examples of Collected GTA5 Data.**

Additionally, we developed a runtime system to dynamically access navigation mesh information, facilitating intelligent camera positioning and motion prediction. This system performs queries on the navigation mesh data structure to extract spatial constraints and valid traversal paths, thereby enabling optimal planning of the camera trajectory. The navigation mesh query process involves

real-time spatial data retrieval and path validation to ensure that camera movements are confined within navigable regions while preserving optimal viewing angles for effective data acquisition.

### 3.3 Quantative Data Evaluation

We collected over 1.2 million video clips through our data curation pipeline, which demonstrated robust performance in several key metrics. The overall accuracy of the data exceeded 99%, and the system achieved a 50-fold improvement in the precision of the camera rotation. Furthermore, the pipeline supported dual concurrent data streams per GPU, effectively doubling production efficiency. A representative trajectory example is shown in Figure 7. The game environment in GTA5 is complex and diverse. The lines in the picture represent the movement path of the agent. We can plan a reasonable path to prevent the agent from colliding or blocking, effectively improving the accuracy of the data.

## 4 Methods

In this section, we present the overall architecture and key components of Matrix-Game 2.0. First, we train a foundation model using our diverse dataset collection, as detailed in Section 4.1. Subsequently, Section 4.2 describes our distillation approach that transforms this foundation model into a few-step autoregressive variant, enabling real-time generation of extended video sequences while maintaining visual quality.

### 4.1 Foundation Model Architecture

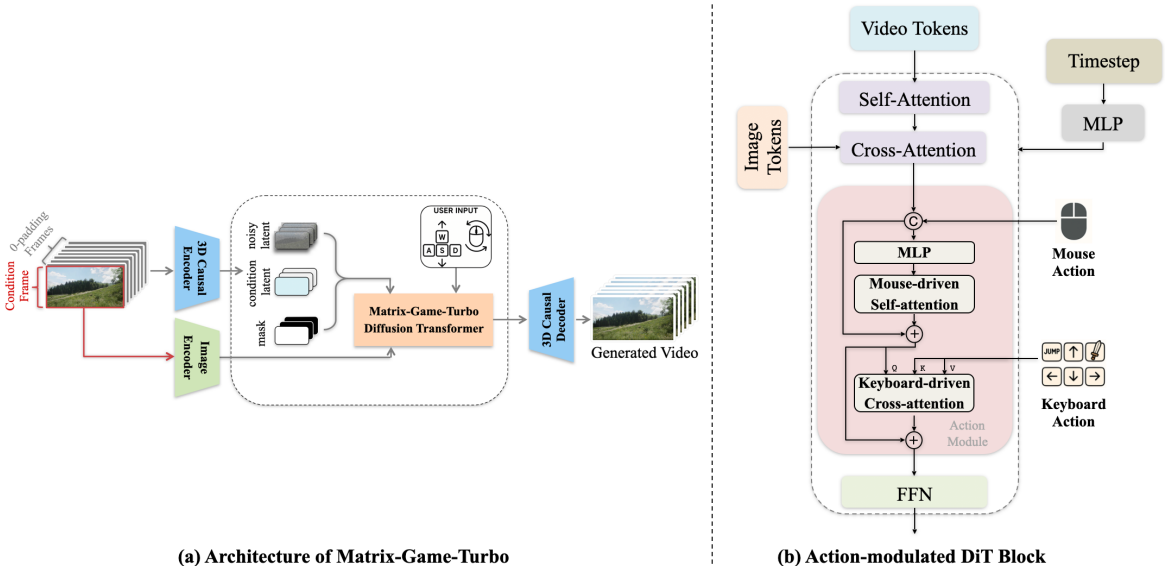
We propose Matrix-Game 2.0, a novel framework to vision-driven world model that explores intelligence capable of understanding and generating the world without relying on language descriptions. In contemporary works, text guidance has become the dominant modality for controlling — examples include SORA [22], HunyuanVideo [17], and Wan [37], all of which leverage text descriptions for generation. However, such methods often introduce semantic priors that bias the generation toward linguistic reasoning rather than physical laws, thereby impeding the model's ability to grasp the fundamental properties of the visual world.

In contrast, Matrix-Game 2.0 eliminates all forms of language input, focusing solely on learning spatial structures and dynamic patterns from image. This de-semanticized modeling approach is inspired by the concept of spatial intelligence [40], emphasizing that the model's capabilities should stem from intuitive understanding of visual and physical laws rather than abstract semantic scaffolding.

As shown in Figure 8(a), Matrix-Game 2.0 takes a single reference image and corresponding action sequences as input, generating a physically plausible video. A 3D Causal VAE [15, 50] is first employed to compress raw video data along both spatial and temporal dimensions — by a factor of $8\times8$ in space and 4 in time — enhancing training efficiency and modeling capability. The image input is encoded by 3D VAE encoder and an image encoder [27] as a condition input. Guided by input actions provided by users, the Diffusion Transformer(DiT) generates a visual token sequence, which is subsequently decoded into a coherent video through a 3D VAE decoder.

To enable interactive control between users and generated contents, Matrix-Game 2.0 incorporates an action module to achieve controllable video generation. Inspired by the control design paradigms of GameFactory [49] and Matrix-Game [52], we embed frame-level action signals into the DiT blocks, as illustrated in Figure 8(b). The injected action signals are divided into two categories: discrete movement actions via keyboard inputs, and continuous viewpoint actions via mouse actions.

We follow the design [49] and [52] to align these control signals with corresponding latent embeddings. Continuous mouse actions are directly concatenated to the input latent representations, forwarded through an MLP layer, and then passed through a temporal self-attention layer. Furthermore, keyboard actions are queried by the fused features through a cross-attention layer, leading to precise controllability for interactions. Different from Matrix-Game [52], we use Rotary Positional Encoding [33] (RoPE) to replace the sin-cos embeddings added to keyboard inputs to facilitate long video generation.




**(a) Architecture of Matrix-Game-Turbo** **(b) Action-modulated DiT Block**

Figure 8: **Overview of Matrix-Game 2.0 Architecture.** The foundation model is derived from the Wan [38] I2V design. By removing the text branch and adding action modules as in Matrix-Game [52], the model predicts next frames only from visual contents and corresponding actions.

### 4.2 Real-time Interactive Auto-Regressive Video Generation

Unlike Matrix-Game [52] which employs a full-sequence diffusion model limited to fixed-length generation, we develop an auto-regressive diffusion framework for real-time long video synthesis. Our approach transforms the bidirectional foundation model into an efficient auto-regressive variant through Self-Forcing [13], which addresses exposure bias by conditioning each frame on previously self-generated outputs rather than ground truth. This significantly reduces the error accumulation characteristic of Teacher Forcing [14] or Diffusion Forcing [4] approaches.

The distillation process comprises two key phases: student initialization and DMD-based Self-Forcing training. We first initialize the student generator $G_\phi$ with weights from the base model, then construct a dataset of ODE trajectories $\{x_t^i\}_{i=1}^N$ using random action sequences, with $t$ is sampled from 3 steps subset of $[0, T]$. During training, block-wise causal masks are applied to attention layers. As shown in Figure 9, we first sample a sequence of noisy input with $N$ frames from the ODE trajectories and split it into $L$ chunks with independent timesteps $\{x_T^i\}_{i=1}^L$. The student generator takes corresponding actions as input and backwards with the regression loss between the denoised output and clean output:

$$\mathcal{L}_{\text{student}} = \mathbb{E}_{x,t^i} \left\| G_\phi \left( \{x_{t^i}^i\}_{i=1}^L, \{c^i\}_{i=1}^L, \{t^i\}_{i=1}^L \right) - \{x_0^i\}_{i=1}^L \right\|^2 \tag{4}$$

The subsequent DMD phase (Figure 10) aligns the student's distributions $p_{\theta,t}\left(x_t^{1:N}\right)$ with the teacher model's $p_{real,t}\left(x_t^{1:N}\right)$ through Self-Forcing. Critically, the generator samples previous frames from its own distribution rather than the training data, ensuring training-inference consistency and mitigating error accumulation.

The KV-caching mechanism enables efficient sequential generation by maintaining a fixed-length cache of recent latents and action embeddings. Our rolling cache implementation automatically manages memory by evicting oldest tokens when exceeding capacity, supporting infinite-length generation. To address potential training-inference discrepancies in image-to-video scenarios where the first frame may be excluded during long video inference, we constrain the KV-cache window size. This forces the model to rely more on its learned priors and understanding to the input actions for generation, improving robustness by making initial frames invisible to subsequent latent frames during training.

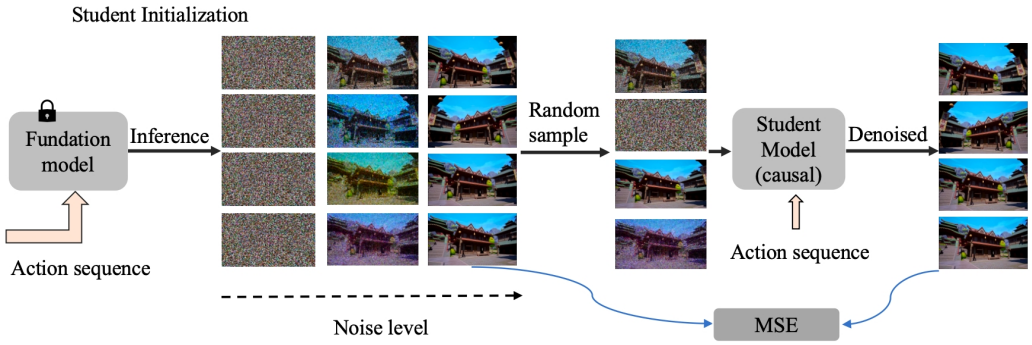



Figure 9: **Causal Student Model Initialization via ODE Trajectories.** The proposed initialization method stabilizes subsequent distillation training by deriving a few-step causal student model from the bidirectional teacher model through optimal ODE trajectory sampling.

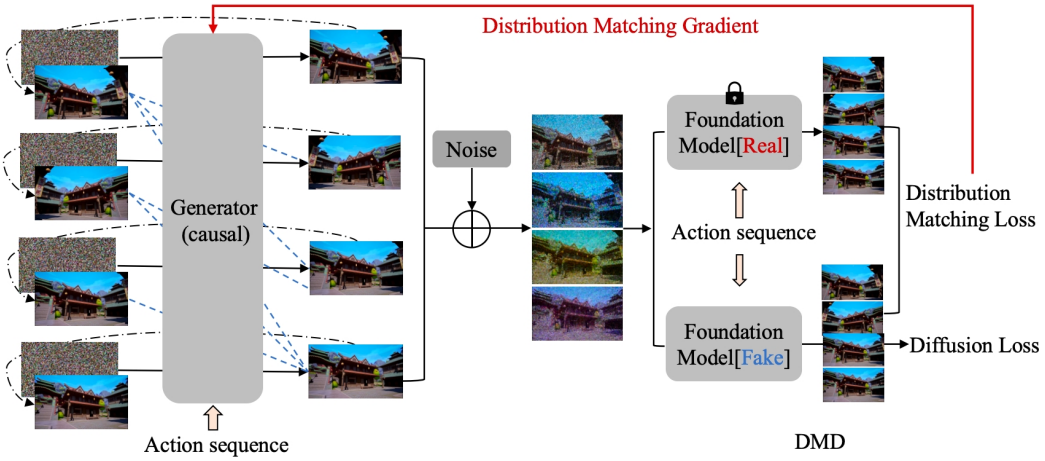



Figure 10: **Overview of Causal Diffusion Model Training via Self-Forcing.** The distillation process aligns the student model's distributions with the teacher model's through self-conditioned generation. This approach effectively mitigates error accumulation while maintaining the bidirectional model's generation quality.

# 5 Experiments

## 5.1 Experiment Settings

### 5.1.1 Implementation Details

For training the foundation model, we initialize our model with SkyReels-V2-I2V-1.3B [5], which adopts the Wan 2.1 [38] architecture. The 1.3B variant provides an optimal balance between generation quality and computational efficiency, enabling real-time speed and high-quality generation performance. We remove the text injection modules from the released checkpoint. To stabilize the whole training process, we firstly fine-tune the model for 5k steps. After that, action modules are added into each DiT block, leading to the total model size as 1.8B. We train the foundation model for 120k steps with learning rate=2e-5, batch size=256. The video data is clipped into 57-frame sections across training process.

For distillation, we firstly collect 40k ODE pairs and fine-tune the causal student model for 6k steps, with subsequent 4k training steps via DMD-based self-forcing. The learning rate is 6e-6. The frame chunk and attention local size are set as 3 and 6, respectively. Additionally, self-forcing is a data-free training method, allowing us to manually design the action sequence distribution for aligning better for input actions from users rather than random action sequences produced by automatic scripts.

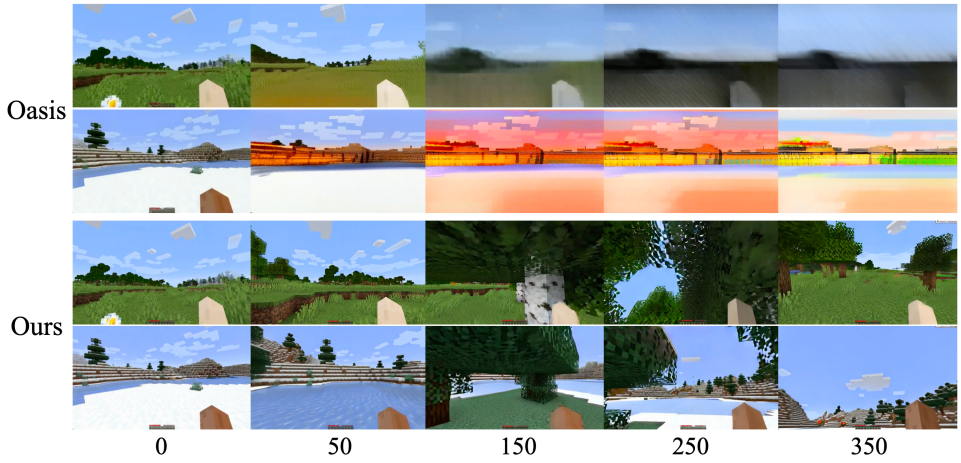



Figure 11: **Qualitative Comparisons on Minecraft Scene Generations.** Compared to Oasis [8], our model shows superior visual performance in long interactive video generations.

| Model | Visual Quality | | Temporal Quality | | Action Controllability | | Physical Understanding | |
|---|---|---|---|---|---|---|---|---|
| | Image Quality ↑ | Aesthetic ↑ | Temporal Cons. ↑ | Motion smooth. ↑ | Keyboard Acc. ↑ | Mouse Acc. ↑ | Obj. Cons. ↑ | Scenario Cons. ↑ |
| Oasis [8] | 0.27 | 0.27 | 0.82 | **0.99** | 0.73 | 0.56 | 0.18 | **0.84** |
| Ours | **0.61** | **0.50** | **0.94** | 0.98 | **0.91** | **0.95** | **0.64** | 0.80 |

Table 1: **Quantitative Comparisons on Minecraft Scene Generations.**

### 5.1.2 Dataset

The training dataset, produced by the pipeline in Section 3, consists of about 800-hour action-annotated video at 360p resolution. The data includes 153-hour Minecraft video data and 615-hour Unreal Engine data, arranged into 65 frames for each video clip. For real-world scenes, we utilize the open-source Sekai dataset [19], obtaining an additional 85 hours of training data after data curation. Given that the environment navigation speed and FPS in the Sekai dataset is different from that of Unreal Engine scenes, we perform frame resampling on the SEKAI data to align the temporal dynamics and movements. To validate the universality of our framework, we further collect 574-hour GTA-driver data and 560-hour Temple Run game data, featuring dynamic scenes interaction for additional fine-tuning. All the videos are resized to 360×640 resolution.

### 5.1.3 Evaluation Metrics and Baselines

We assess our universal real-time model using the comprehensive GameWorld Score Benchmark [52] introduced in Matrix-Game. This benchmark provides a multi-dimensional evaluation framework examining four critical capabilities: visual quality, temporal quality, action controllability and physical rule understanding. Given the current scarcity of open-source interactive world models, we conduct separate evaluations for two distinct domains: Minecraft and wild scenes. For Minecraft environments, we compare against Oasis [8] as our primary baseline, while employing YUME [21] for more complex wild scene generation tasks. All experiments utilize a standardized 597-frame composite action sequence, with evaluation performed on 32 Minecraft scenes and 16 diverse wild scene images to ensure representative coverage of different environmental conditions.

## 5.2 Generation Results

We present comprehensive qualitative and quantitative evaluations comparing Matrix-Game 2.0 against state-of-the-art baselines across multiple domains, including long video generation in both Minecraft environments and wild scenes, as well as extended generation results for GTA driving scenarios and TempleRun gameplay.

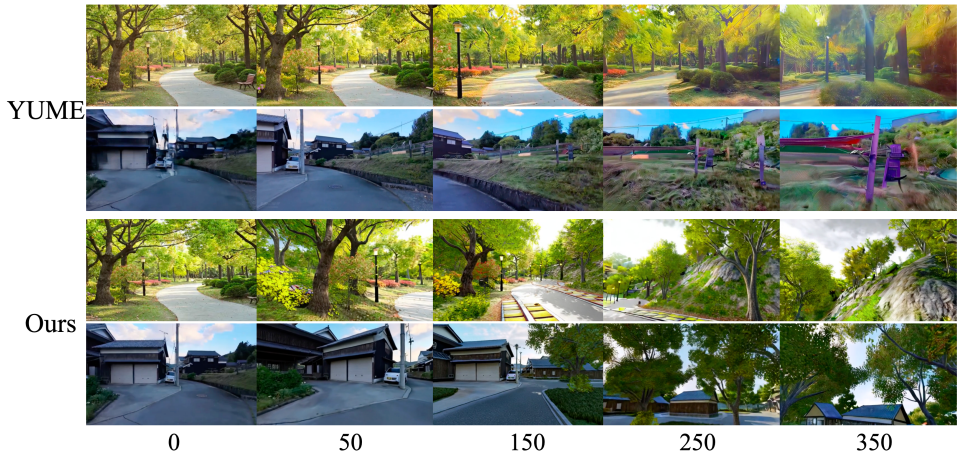



Figure 12: **Qualitative Comparisons on Wild Scene Generations.** For wild image inputs, Matrix-Game 2.0 exhibits strong generalization capabilities, fast generation speed, and accurate interaction responses.

| Model | Visual Quality | | Temporal Quality | | Physical Understanding | |
|---|---|---|---|---|---|---|
| | Image Quality ↑ | Aesthetic ↑ | Temporal Cons. ↑ | Motion smooth. ↑ | Obj. Cons. ↑ | Scenario Cons. ↑ |
| YUME [21] | 0.65 | 0.48 | 0.85 | **0.99** | **0.77** | **0.80** |
| Ours | **0.67** | **0.51** | **0.86** | 0.98 | 0.71 | 0.76 |

Table 2: **Quantitative Comparisons on Wild Scene Generations.**

### 5.2.1 Minecraft Scene Results

Figure 11 and Table 1 demonstrate Matrix-Game 2.0's superior performance compared to Oasis [8]. While Oasis exhibits significant quality degradation after several dozen frames, our model maintains long-term consistency throughout extended generation sequences. Quantitative metrics reveal substantial improvements across most evaluation dimensions, though we observe marginally lower scores in scene consistency and action smoothness. We attribute this to Oasis's tendency to produce static frames after collapse, which inflates these particular metrics.

### 5.2.2 Wild Scene Results

Our comparison with YUME [21] in Figure 12 reveals Matrix-Game-V2's stronger robustness in wild scene generation. YUME develops noticeable artifacts and color saturation issues after several hundred frames, while ours maintains stable style fidelity. Moreover, the generation speed of YUME maintains slow, which is hard to be directly applied for interactive world modeling.

Table 2 shows the quantitative results. Since the action controllability assessment in GameWorld Score Benchmark was designed specifically for Minecraft evaluation, it cannot be directly applied to wild scenes. Empirical results demonstrate that YUME exhibits significantly degraded action control performance in out-of-domain scenarios, while our method maintains robust controllability.

### 5.2.3 More Qualitative Results

Figure 13 showcases Matrix-Game 2.0's exceptional capability for long video generation with minimal quality degradation. The model's strong domain adaptability is further evidenced by its performance in diverse scenarios including GTA driving environments (Figure 14) and TempleRun game (Figure 15), demonstrating its potential as a foundation for world modeling.


Figure 13: **Long Video Generations of Matrix-Game 2.0.** The real-time generation results demonstrate excellent visual quality and precise action controllability when generating long videos.

### 5.3 Ablation Studies

#### 5.3.1 Different KV-cache Local Size

The KV-cache mechanism plays a crucial role in maintaining contextual information during Matrix-Game 2.0's auto-regressive generation process. Our investigation reveals an important trade-off in cache size selection: while larger caches (9 latent frames) theoretically provide richer historical context, they paradoxically lead to earlier onset of visual artifacts (Figure 16). Comparative analysis shows that models with 6-frame caches demonstrate superior long-term generation quality, with significantly reduced distortion and degradation artifacts.

We attribute this phenomenon to an over-reliance on cached information during generation. With larger cache sizes, the model increasingly depends on stored cache rather than actively correcting accumulated errors through learned capability of model itself. This creates a compounding effect where artifacts in early frames become more momerized through the cache mechanism, ultimately being treated as valid scene elements. Our findings suggest that moderate cache sizes (6 frames) provide a balance between context preservation and error correction capability.

#### 5.3.2 Comparative Analysis of Acceleration Techniques

To achieve real-time generation at 25 FPS, we systematically optimized both the diffusion model and VAE components through several key modifications. First, we integrated the efficient Wan2.1-VAE architecture with caching mechanism, significantly accelerating the decoding process for extended video sequences. Second, we strategically employ action modules only in the first half of DiT


Figure 14: **Generation Results under GTA5 driving scenes.**

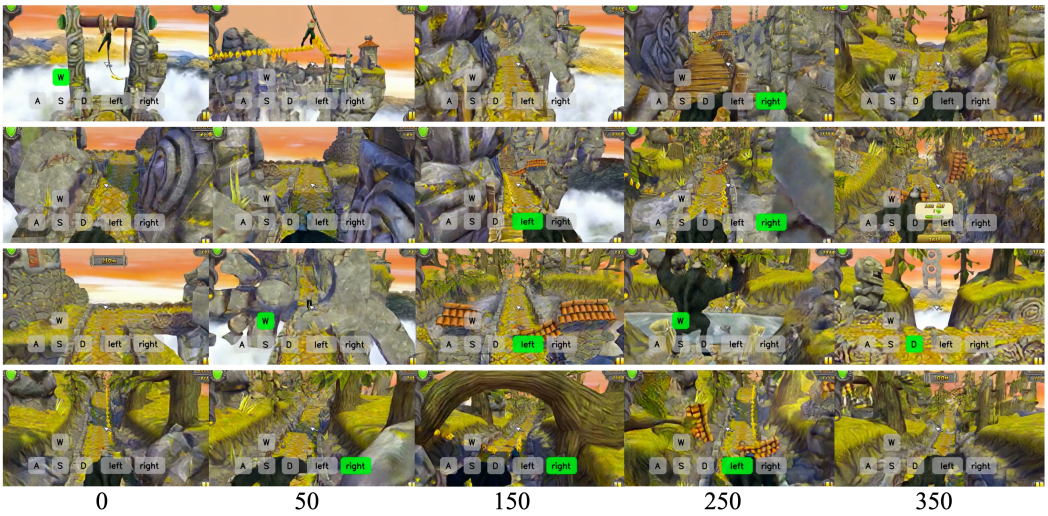



Figure 15: **Generation Results under the Parkour Game TempleRun scene.**

blocks, and reduce the denoising steps from 4 to 3 in the distillation process. The quantitative comparisons are shown in Table 3. Quantitative comparisons shown in Table 3 demonstrate that these acceleration strategies can achieve 25 FPS while maintaining generation quality, resulting in an optimal speed-quality trade-off.

## 6 Conclusion

Matrix-Game 2.0 represents a significant advancement in real-time interactive video generation through three key innovations. First, we developed a comprehensive data production pipeline that overcomes previous limitations in obtaining high-quality training data for interactive scenarios. Our systemetic pipeline based on Unreal Engine, together with the video recording framework verified in GTA5 environments, establish new standards for scalable production of action-annotated video data at unprecedented fidelity.

Building upon this foundation, we introduced an autoregressive diffusion framework that uniquely combines action-conditioned modulation with Self-Forcing training. This approach effectively mitigates the error accumulation problem that has traditionally plagued long video synthesis while maintaining real-time performance. Through systematic optimizations of both the diffusion process

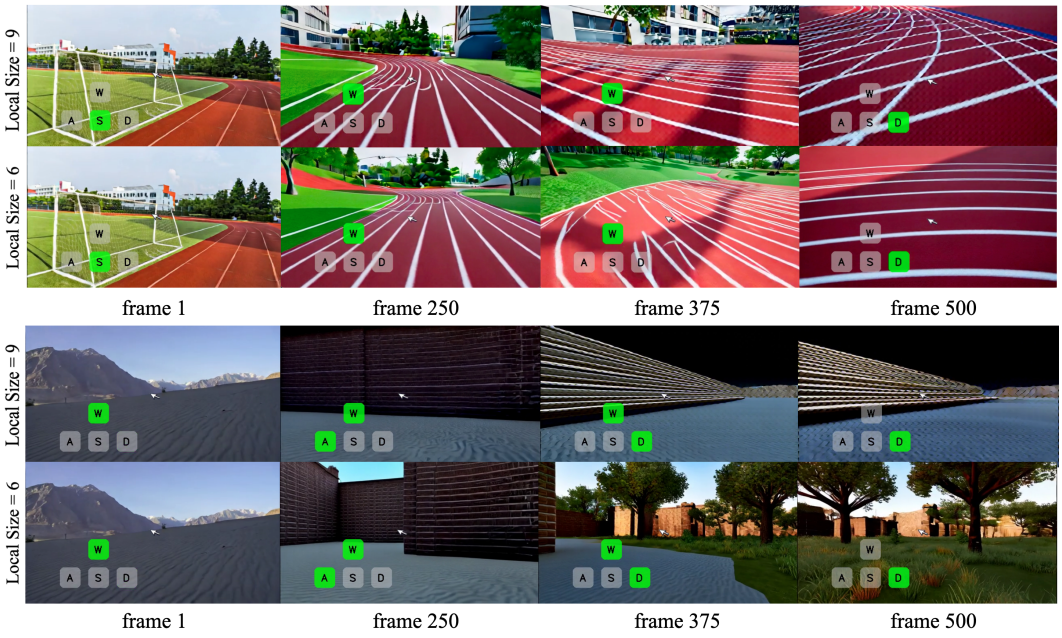



Figure 16: **Qualitative Comparison on Using Different Local Size for KV-cache**. Larger local size cause artifacts in long sequences while smaller local size can keep a balance between visual quality and content fidelity.

| Acceleration Techniques | Visual Quality | | Temporal Quality | | Action Controllability | | Physical Understanding | | Speed |
|---|---|---|---|---|---|---|---|---|---|
| | Image ↑ | Aesthetic ↑ | Temporal ↑ | Motion ↑ | Keyboard ↑ | Mouse ↑ | Object ↑ | Scenario ↑ | FPS ↑ |
| (1) +VAE Cache | **0.61** | **0.51** | 0.93 | 0.97 | 0.91 | **0.95** | **0.68** | **0.81** | 15.49 |
| (2) (1)+Halving action modules | **0.61** | **0.51** | **0.94** | 0.97 | 0.92 | **0.95** | 0.63 | **0.81** | 21.03 |
| (3) (2)+Reduce denoising steps (4→3) | **0.61** | 0.50 | **0.94** | **0.98** | **0.91** | **0.95** | 0.64 | 0.80 | **25.15** |

Table 3: **Quantitative Comparisons of Different Acceleration Techniques.** While maintaining comparable generation quality metrics, our combined acceleration techniques achieve 25 FPS throughput, enabling real-time on-the-fly video generation.

and VAE architecture, we achieved a generation speed of 25 FPS - fast enough for seamless human-in-the-loop interaction.

Extensive experiments demonstrate that Matrix-Game 2.0 sets new benchmarks for interactive generation systems, delivering excellent performance in both visual quality and action controllability. The model's ability to maintain temporal coherence during extended interactions while responding precisely to user inputs represents a substantial step forward for applications requiring real-time world simulation.

### 6.1 Limitations

While demonstrating strong performance, Matrix-Game 2.0 has several limitations that point to future research directions. First, the model shows limited generalization capability when handling out-of-domain (OOD) scenes - for example, moving the camera upward or step forward for a long time in OOD scenes may result in over-saturated or degraded results. Second, the current 360×640 resolution output falls short of state-of-the-art video generation models that typically produce higher-definition results. Third, while the auto-regressive diffusion model enables long video generation, maintaining content consistency and history over long video generations remains challenging due to the lack of explicit memory mechanisms for history preservation.

We note that these limitations present clear pathways for improvement. The generalization and resolution issues can be improved through expanded training data domain and model architecture scaling. Moreover, the last limitation could be addressed by integrating compatible memory retrival


Figure 17: **Bad cases.** Matrix-Game-V2 sometimes fails when handling out-of-domain scenes, like producing over-saturated (left) or degraded (right) results.

mechanisms without compromising real-time performance. These directions will be the focus of our future work towards making interactive video generation truly practical for real-world applications.

## References


[1] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos: world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

[2] Philip J. Ball, Jakob Bauer, Frank Belletti, Bethanie Brownfield, Ariel Ephrat, Shlomi Fruchter, Agrim Gupta, Kristian Holsheimer, Aleksander Holynski, Jiri Hron, Christos Kaplanis, Marjorie Limont, Matt McGill, Yanko Oliveira, Jack Parker-Holder, Frank Perbet, Guy Scully, Jeremy Shar, Stephen Spencer, Omer Tov, Ruben Villegas, Emma Wang, Jessica Yung, Cip Baetu, Jordi Berbel, David Bridson, Jake Bruce, Gavin Buttimore, Sarah Chakera, Bilva Chandra, Paul Collins, Alex Cullum, Bogdan Damoc, Vibha Dasagi, Maxime Gazeau, Charles Gbadamosi, Woohyun Han, Ed Hirst, Ashyana Kachra, Lucie Kerley, Kristian Kjems, Eva Knoepfel, Vika Koriakin, Jessica Lo, Cong Lu, Zeb Mehring, Alex Moufarek, Henna Nandwani, Valeria Oliveira, Fabio Pardo, Jane Park, Andrew Pierson, Ben Poole, Helen Ran, Tim Salimans, Manuel Sanchez, Igor Saprykin, Amy Shen, Sailesh Sidhwani, Duncan Smith, Joe Stanton, Hamish Tomlinson, Dimple Vijaykumar, Luyu Wang, Piers Wingfield, Nat Wong, Keyang Xu, Christopher Yew, Nick Young, Vadim Zubov, Douglas Eck, Dumitru Erhan, Koray Kavukcuoglu, Demis Hassabis, Zoubin Gharamani, Raia Hadsell, Aäron van den Oord, Inbar Mosseri, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 3: A new frontier for world models. 2025.

[3] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[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] Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

[6] Xinyuan Chen, Yaohui Wang, Lingjun Zhang, Shaobin Zhuang, Xin Ma, Jiashuo Yu, Yali Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Seine: Short-to-long video diffusion model for generative transition and prediction. In *The Twelfth International Conference on Learning Representations*, 2023.

[7] Xinle Cheng, Tianyu He, Jiayi Xu, Junliang Guo, Di He, and Jiang Bian. Playing with transformer at 30+ fps via next-frame diffusion. *arXiv preprint arXiv:2506.01380*, 2025.

[8] Decart. Oasis: A universe in a transformer. 2024.

[9] Junliang Guo, Yang Ye, Tianyu He, Haoyu Wu, Yushu Jiang, Tim Pearce, and Jiang Bian. Mineworld: a real-time and open-source interactive world model on minecraft. *arXiv preprint arXiv:2504.08388*, 2025.

[10] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

[11] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. *arXiv preprint arXiv:2404.02101*, 2024.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

[13] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

[14] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024.

[15] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

[16] 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.

[17] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. Hunyuanvideo: A systematic framework for large video generative models, 2025.

[18] Jiaqi Li, Junshu Tang, Zhiyong Xu, Longhuang Wu, Yuan Zhou, Shuai Shao, Tianbao Yu, Zhiguo Cao, and Qinglin Lu. Hunyuan-gamecraft: High-dynamic interactive game video generation with hybrid history condition. *arXiv preprint arXiv:2506.17201*, 2025.

[19] Zhen Li, Chuanhao Li, Xiaofeng Mao, Shaoheng Lin, Ming Li, Shitian Zhao, Zhaopan Xu, Xinyue Li, Yukang Feng, Jianwen Sun, et al. Sekai: A video dataset towards world exploration. *arXiv preprint arXiv:2506.15675*, 2025.

[20] Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024.

[21] Xiaofeng Mao, Shaoheng Lin, Zhen Li, Chuanhao Li, Wenshuo Peng, Tong He, Jiangmiao Pang, Mingmin Chi, Yu Qiao, and Kaipeng Zhang. Yume: An interactive world generation model. *arXiv preprint arXiv:2507.17744*, 2025.

[22] OpenAI. Sora: Video generation models as world simulators. https://openai.com/index/video-generation-models-as-world-simulators/, 2024.

[23] J Parker-Holder, P Ball, J Bruce, V Dasagi, K Holsheimer, C Kaplanis, A Moufarek, G Scully, J Shar, J Shi, et al. Genie 2: A large-scale foundation world model. *URL: https://deepmind. google/discover/blog/genie-2-a-large-scale-foundation-world-model*, 2024.

[24] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *International Conference on Computer Vision*, pages 4195–4205, 2023.

[25] Ryan Po, Yotam Nitzan, Richard Zhang, Berlin Chen, Tri Dao, Eli Shechtman, Gordon Wetzstein, and Xun Huang. Long-context state-space video world models. *arXiv preprint arXiv:2505.20171*, 2025.

[26] Haonan Qiu, Menghan Xia, Yong Zhang, Yingqing He, Xintao Wang, Ying Shan, and Ziwei Liu. Freenoise: Tuning-free longer video diffusion via noise rescheduling. *arXiv preprint arXiv:2310.15169*, 2023.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[28] Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. Gen3c: 3d-informed world-consistent video generation with precise camera control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[30] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[31] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

[32] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[33] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *CoRR*, abs/2104.09864, 2021.

[34] 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.

[35] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[37] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong,

Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models, 2025.

[38] Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

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

[40] World Labs. Generating worlds. https://www.worldlabs.ai/blog, 2025.

[41] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 1(2):6, 2023.

[42] Sherry Yang, Yilun Du, Seyed Kamyar Seyed Ghasemipour, Jonathan Tompson, Leslie Pack Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *International Conference on Learning Representations*, 2024.

[43] Sherry Yang, Jacob C Walker, Jack Parker-Holder, Yilun Du, Jake Bruce, Andre Barreto, Pieter Abbeel, and Dale Schuurmans. Position: video as the new language for real-world decision making. In *International Conference on Machine Learning*, 2024.

[44] Shiyuan Yang, Liang Hou, Haibin Huang, Chongyang Ma, Pengfei Wan, Di Zhang, Xiaodong Chen, and Jing Liao. Direct-a-video: Customized video generation with user-directed camera movement and object motion. In *ACM SIGGRAPH*, pages 1–12, 2024.

[45] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[46] Shengming Yin, Chenfei Wu, Huan Yang, Jianfeng Wang, Xiaodong Wang, Minheng Ni, Zhengyuan Yang, Linjie Li, Shuguang Liu, Fan Yang, et al. Nuwa-xl: Diffusion over diffusion for extremely long video generation. *arXiv preprint arXiv:2303.12346*, 2023.

[47] 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.

[48] Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22963–22974, 2025.

[49] Jiwen Yu, Yiran Qin, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. Gamefactory: Creating new games with generative interactive videos. In *International Conference on Computer Vision*, 2025.

[50] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. In *International Conference on Learning Representations*, 2024.

[51] Yifan Zhang and Bryan Hooi. Hipa: enabling one-step text-to-image diffusion models via high-frequency-promoting adaptation. *arXiv preprint arXiv:2311.18158*, 2023.

[52] Yifan Zhang, Chunli Peng, Boyang Wang, Puyi Wang, Qingcheng Zhu, Fei Kang, Biao Jiang, Zedong Gao, Eric Li, Yang Liu, and Yahui Zhou. Matrix-game: Interactive world foundation model. *arXiv preprint arXiv:2506.18701*, 2025.

[53] Yifan Zhang, Daquan Zhou, Bryan Hooi, Kai Wang, and Jiashi Feng. Expanding small-scale datasets with guided imagination. In *Advances in Neural Information Processing Systems*, volume 36, pages 76558–76618, 2023.

[54] Canyu Zhao, Mingyu Liu, Wen Wang, Weihua Chen, Fan Wang, Hao Chen, Bo Zhang, and Chunhua Shen. Moviedreamer: Hierarchical generation for coherent long visual sequence. *arXiv preprint arXiv:2407.16655*, 2024.

[55] Longtao Zheng, Yifan Zhang, Hanzhong Guo, Jiachun Pan, Zhenxiong Tan, Jiahao Lu, Chuanxin Tang, Bo An, and Shuicheng Yan. Memo: Memory-guided diffusion for expressive talking video generation. *arXiv preprint arXiv:2412.04448*, 2024.

# A Past Frame

**Algorithm 1** PastFrame Training

**Require:** Denoise timesteps $\{t_1, \ldots, t_T\}$
**Require:** Diffusion model $G_\theta$
**Require:** Number of past frames $k$
**Require:** Video data $z, x$ (correspond to the first and second halves of a video segment.)
**Require:** ground truth past frame $z_k, x_k$
**loop**
  **for** $i = T, \ldots, 1$ **do**
    **if** $i = 1$ **then**
      Enable gradient computation
      Set $\hat{z}_0 \leftarrow G_\theta(z_{t_i}; t_i, z_k)$
    **else**
      Disable gradient computation
      Set $\hat{z}_0 \leftarrow G_\theta(z_{t_i}; t_i, z_k)$
    **end if**
  **end for**
  Sample $\epsilon \sim \mathcal{N}(0, I)$
  Set $x_{t_i} \leftarrow \Psi(x_0, \epsilon, t_i)$
  Get generated latent $\hat{x}_k \leftarrow z_0^{-n:-1}$
  Enable gradient computation
  Set $\hat{x}_0 \leftarrow G_\theta(x_{t_i}; t_i, \hat{x}_k)$
  Update $\theta$
**end loop**

One key parallel between our foundation model and existing video generation works is their shared reliance on generating fixed-length video clips via image-to-world modeling, a paradigm that diverges significantly from the infinite-world modeling capabilities we aim to achieve. To enable infinite video generation, we have explored two primary approaches. First, we advanced the autoregressive video generation framework of Matrix Game v1 [52] by incorporating previously generated frames as input during training to predict subsequent video segments. This modification aims to bridge the train-test gap and extend the temporal length of generated sequences. Second, we introduce a self-forcing mechanism to facilitate autoregressive generation of videos with theoretically infinite duration.

To revisit the past frame strategy in Matrix Game v1: we concatenate the latent representations of past video frames with the current video latent along the channel dimension, which is then fed into DIT as input. To enhance model robustness, during training we inject random Gaussian noise a probability of 0.2 into these past frame latents to mitigate the impact of error accumulation. Furthermore, we apply Classifier-Free Guidance (CFG) to the past frames to improve the quality of generated video.

However, such training suffers from a critical train-test divergence: during inference, past frames are model-generated rather than ground-truth (GT) frames, and this mismatch amplifies error accumulation—manifesting as severe quality degradation in Version 1 beyond 4 generated segments. To address this gap, during training we explicitly utilize model-generated past frame latents to predict subsequent video segments, aligning training dynamics with inference-time behavior. Our detailed algorithm is illustrated as shown in Algorithm 1. During training, we perform 30-step sampling to generate $z_0$ from the previous video segment $z$ using our foundation model, producing the generated past frame $\hat{x}_k$. We then use $\hat{x}_k$ as the conditional input for generating the subsequent video segment. To reduce GPU memory usage, we only enable gradient computation at the final sampling step. Forward process $x^i_{t^i} = \Psi(x^i, \epsilon^i, t^i) = \alpha_{t^i} x^i + \sigma_{t^i} \epsilon^i$, where $\alpha_{t^i}, \sigma_{t^i}$ are pre-defined noise schedule within a finite time horizon $t^i \in [0, 1000]$ and $\epsilon^i \sim \mathcal{N}(0, I)$ is Gaussian noise.