Make A Game: A Novel Paradigm for Interactive Game Rendering

Supplementary Materials

A. Related Works

A. Video Diffusion Models

In recent years, diffusion models have demonstrated significant potential in the field of video generation, particularly following the introduction of Latent Diffusion Models (LDM) [1], which achieved groundbreaking results in works like MagicVideo [2]. As a result, diffusion models have become the de facto standard for video generation. Based on existing research, video diffusion models can be broadly categorized into two primary architectures—U-Net and Transformer—and can be divided into two major tasks: image-to-video and text-to-video.

U-Net-based architectures were prevalent in earlier works, such as models like VideoFactory [3], ModelScope [4], AnimateAnything [5], and Stable Video Diffusion [6]. These models have shown promising results in realism, text adherence, temporal consistency, and dynamics, and typically leverage pre-trained text-to-image models combined with various temporal mixing layers to handle the intrinsic temporal dimension of video data [7] [8]. However, due to the inherent limitations of the U-Net architecture, these methods often face scalability challenges and limited multimodal controllability, restricting higher-quality and more complex video generation.

To overcome these limitations, Transformer-based architectures have recently emerged as a promising new direction [9], gradually becoming the next generation of video diffusion models. Sora [10] and its open-source implementations (Opensora [11], Opensora-plan [12]), CogVideoX (based on MMDiT), and mochi1 (based on AsymmDiT [13]) exemplify this paradigm shift, demonstrating superior scalability and enhanced multimodal integration capabilities. These models have achieved remarkable improvements in video length, resolution, and visual quality for both text-to-video and image-to-video generation tasks, thus paving the way for future research aiming at stronger interactivity and higher video quality. Nevertheless, the exploration of gaming video generation and its interactive controllability remains relatively under-explored, calling for more extensive investigation in the field.

B. Interactive Game Simulation and World Models

Traditional interactive game simulation typically relies on game engines like Unreal and Unity, which maintain game world states, handle game logic, and render continuous image streams based on scene geometry representations and user interactions. However, developing such engines is time-intensive and resource-demanding, motivating recent research to explore neural network models as alternatives to conventional game engines.

Early approaches such as GameNGEN [14], DIAMOND [15], and Oasis [16] leverage diffusion models to simulate playable environments in games like Atari, DOOM, and Minecraft for real-time, highly dynamic scene generation and user interaction. Nonetheless, these works often lack the capacity to customize or generate new game visuals according to user preferences. Consequently, more recent studies extend beyond conventional playable scenarios to develop more flexible and user-specific simulation frameworks. For instance, UniSim and Pandora employ supervised learning to predict future frames given a video segment and action prompts, while PVG and Genie focus on

unsupervised action learning from video data. Similar to these works, GameGAN [17], GameNGen [14], and DIAMOND [15] also investigate playable simulations of early games such as Atari and DOOM, integrating gaming agents for interactive gameplay. Oasis uses diffusion models to simulate Minecraft in real time, including both graphics and some internal game systems, but it has not delved deeply into the complexities of next-generation game environments.

Addressing this gap, MAG pushes the boundaries of game simulation by offering more intricate environments, dynamic events, diverse characters, and complex actions with a high degree of realism and variety. This method can generate subsequent frames based on the current video segment and player-provided multi-modal signals, ensuring that the system is both visually compelling and contextually coherent with player actions. By bridging the gap between simpler game simulations and the sophisticated demands of next-generation open-world games, MAG provides a robust foundation for advanced interactive game experiences.

B. Detailed Explanation of Multi-Level Haar Wavelet Transform in WF-VAE

WF-VAE leverages multi-level Haar wavelet transforms to extract low-frequency energy flows from videos and introduces an energy flow pathway, combined with a lossless causal cache mechanism, to achieve efficient compression and reconstruction, significantly reducing computational overhead while enhancing video generation quality.

The multi-level Haar wavelet transform is a key component of WF-VAE, enabling efficient video compression by decomposing video signals

$$V \in \mathbb{R}^{c \times t \times h \times w} \tag{1}$$

into low-frequency and high-frequency components to prioritize critical information while reducing computational costs. The Haar scaling filter

$$h = \sqrt{\frac{1}{2}}[1, 1] \tag{2}$$

acts as a low-pass filter to capture approximation coefficients, while the Haar wavelet filter

$$g = \sqrt{\frac{1}{2}}[1, -1] \tag{3}$$

serves as a high-pass filter for extracting detail coefficients. For a given video V, the 3D Haar wavelet transform at layer l is defined as:

$$S_{ijk}^{(l)} = S^{(l-1)} * (f_i \otimes f_j \otimes f_k), \tag{4}$$

where $f_i, f_j, f_k \in \{h, g\}$, $*$ denotes convolution, and $\otimes$ represents the tensor product. Starting from

$$S^{(0)} = V, \tag{5}$$

this process generates eight sub-band components:

$$W^{(l)} = \{S_{hhh}^{(l)}, S_{hhg}^{(l)}, S_{hgh}^{(l)}, S_{ghh}^{(l)}, S_{hgg}^{(l)}, S_{ggh}^{(l)}, S_{ghg}^{(l)}, S_{ggg}^{(l)}\}, \tag{6}$$

where $S_{hhh}^{(l)}$ represents the low-frequency component, and $S_{ggg}^{(l)}$ captures high-frequency details.

To achieve multi-scale compression, the transform recursively applies the low-frequency component of the previous layer

$$S^{(l)} = S^{(l-1)}_{hhh} \tag{7}$$

to subsequent layers, enabling compression rates: $4 \times 8 \times 8$ (temporal $\times$ height $\times$ width) through combinations of 3D and 2D wavelet transforms.

During decoding, the inverse wavelet transform (IWT) reconstructs the video by combining low-frequency and high-frequency components, expressed as:

$$S^{(l-1)}_{hhh} = \text{IWT}(W^{(l)}) + S^{(l-1)}_{\text{outflow},hhh}. \tag{8}$$

At the final layer, the reconstructed video is obtained as:

$$\hat{V} = \text{IWT}(W^{(1)}) + S^{(1)}_{\text{outflow},hhh}. \tag{9}$$

To further reduce computational costs, WF-VAE introduces an energy flow pathway for low-frequency components, expressed as:

$$S^{(l)}_{\text{concat}} = \text{Concat}(S^{(l)}_{hhh}, S^{(l)}_{\text{backbone}}), \tag{10}$$

where $S^{(l)}_{\text{backbone}}$ represents the feature maps from the backbone network. This pathway allows low-frequency energy to bypass the backbone, preserving critical information in the latent representation while minimizing the reliance on dense 3D convolutions.

By leveraging this structure, WF-VAE achieves efficient video compression with state-of-the-art reconstruction quality and significantly reduced computational requirements.

C. The mathematics of Flow Matching

Flow Matching for video diffusion models aims to learn a continuous transformation (or flow) from a data distribution of video sequences to a simple prior distribution (often Gaussian) in a unified time interval. Let us denote a video sequence as a tensor $\mathbf{x} \in \mathbb{R}^{T \times H \times W \times C}$, where T is the temporal dimension (number of frames), H and W are the spatial dimensions, and C is the number of channels. We introduce a continuous variable $s \in [0, 1]$ to parameterize the transformation from the data manifold at $s = 0$ to the prior manifold at $s = 1$. The core idea of Flow Matching is to train a velocity field $\mathbf{v}_\theta(\mathbf{x}, s)$ such that it aligns with an ideal flow that transports each sample $\mathbf{x}^0$ at $s = 0$ (drawn from the data distribution) to a corresponding sample $\mathbf{x}^1$ at $s = 1$ (drawn from the prior distribution).

To formalize this process, we define the following ordinary differential equation (ODE):

$$\frac{d\mathbf{x}(s)}{ds} \;=\; \mathbf{v}_\theta\big(\mathbf{x}(s), s\big), \tag{11}$$

where $\mathbf{x}(0) = \mathbf{x}^0$ is drawn from the empirical data distribution of video sequences, and $\mathbf{x}(1) = \mathbf{z}$ is expected to follow a tractable prior (standard Gaussian). The function $\mathbf{v}_\theta(\mathbf{x}, s)$ is parameterized by deep neural networks (U-Net or Transformer architectures adapted for spatiotemporal data).

To train $\mathbf{v}_\theta$, we construct a *reference flow* $\dot{\mathbf{x}}(s)$ from a known diffusion or noising process that smoothly bridges the data distribution at $s = 0$ and the prior distribution at $s = 1$. In practice, one approach is to sample $\mathbf{x}(s)$ from a forward SDE or from controlled perturbations of the data, and

record the instantaneous velocity $\dot{\mathbf{x}}(s)$. Flow Matching training then minimizes the mean squared error between the learned velocity field $\mathbf{v}_\theta(\mathbf{x}(s), s)$ and the reference flow $\dot{\mathbf{x}}(s)$:

$$\mathcal{L}(\theta) \;=\; \mathbb{E}_{s,\mathbf{x}(0),\mathbf{x}(s)}\Big[\big\|\mathbf{v}_\theta\big(\mathbf{x}(s), s\big) \;-\; \dot{\mathbf{x}}(s)\big\|^2\Big], \tag{12}$$

where the expectation is taken over the random draw $s \sim \mathrm{Uniform}(0, 1)$, the data sample $\mathbf{x}(0) \sim p_{\text{data}}(\mathbf{x})$, and the intermediate state $\mathbf{x}(s)$ obtained from the reference flow process. By minimizing (12), the model learns a velocity field that approximates the transport direction at every point $(\mathbf{x}(s), s)$.

Once the velocity field $\mathbf{v}_\theta$ is trained, generation proceeds by drawing a random sample $\mathbf{x}(1)$ from the prior distribution (a Gaussian tensor for the entire spatiotemporal volume) and solving the following ODE in reverse from $s = 1$ down to $s = 0$:

$$\frac{d\mathbf{x}(s)}{ds} \;=\; -\mathbf{v}_\theta\big(\mathbf{x}(s), s\big), \tag{13}$$

with $\mathbf{x}(1)$ carrying noise drawn from the prior. Integrating (13) backward in s yields a sample $\mathbf{x}(0)$ at $s = 0$, which ultimately corresponds to a synthesized video sequence in the data space.

In summary, the Flow Matching framework for video diffusion models:

- Introduces a continuous parameter s to blend between data and prior distributions.
- Defines a learnable velocity field $\mathbf{v}_\theta(\mathbf{x}, s)$ in the ODE (11).
- Constructs a reference flow $\dot{\mathbf{x}}(s)$ to guide learning via the objective (12).
- Generates new videos by sampling from the prior and integrating the negative velocity field (13) in reverse.

Because each video $\mathbf{x}$ is treated as a high-dimensional tensor, the neural networks parameterizing $\mathbf{v}_\theta$ are adapted for spatiotemporal data. By matching the flow at every intermediate step in s, the model learns a faithful mapping that can synthesize realistic and temporally coherent video sequences while providing a principled framework for bridging data and prior distributions.

D. MAG Three-Stage Training Scheme and Corresponding Data Preparation Details

A. *Overview*

MAG aims to achieve two major goals:

- Extract generic visual and action representations from 3D game images and videos, establishing robust generative comprehension for complex scenes.
- Enable multimodal interactive video generation, focusing on controllable character motion to create "playable" game video segments.

To more efficiently reach these objectives, the training process is divided into three stages:

- **Stage 1: Image Pretraining.** A large-scale pretraining on massive open-world 3D game images to fully finetune the base weights.
- **Stage 2: Video Backbone (UC-3DMMDiT) Training.** Similar to CogVideoX, it involves video data cleaning and recaption, as well as multi-resolution and multi-length bucketed training, ensuring both image-to-video generalization and temporal consistency.
- **Stage 3: APB (Action Prompt Block) Finetuning.** Inspired by the InstructNet concept in GameGenX, this step leverages high-quality "character motion" data for multimodal interactive control, particularly for character actions.

MAG reuses an existing VAE module, named WFVAE, without additional training. Improvements and finetuning efforts primarily focus on the core MAG backbone and its APB plugin.

a) Training setup overview: The distributed mixed-precision training of MAG is implemented using HuggingFace's Accelerate library and the DeepSpeed ZeRO3 architecture, with memory optimization achieved via the XFormer architecture. MAG consists of 8.8 billion parameters and required approximately 2880 Nvidia H100 GPU hours for training. Techniques such as Explicit Uniform Sampling and bucket training are employed to accelerate convergence.

b) Inference setup overview: The inference process is implemented on a single Nvidia H100 GPU. We observed that excessive sampling steps not only significantly degrade the real-time performance of MAG but also impair the interactive capabilities of APB. Thus, we adopt a 10-step DPM solver++ for sampling. Additionally, to mitigate the accumulation of artifacts during the autoregressive inference process, we use a relatively small classifier-free guidance (CFG) weight of 1.3.

c) Additional Notes: This approach integrates and adapts insights from CogVideoX and GameGenX, while also referencing data filtering and caption generation methods from models such as OpenSora and GameNGen. By leveraging these experiences, we carefully design both data curation workflows and model architectures to optimize results in generating realistic, interactive game scenes.

B. Data Preparation and Cleaning Process

We conduct multiple rounds of filtering, cleaning, and labeling on raw data sources to ensure that the training set is sufficiently diverse yet meets requirements regarding visual quality, action coherence, and scene relevance.

a) Data Acquisition:

- **Data Sources:**
  1) Recorded gameplay videos and trailers of open-world 3D games sourced from online platforms, primarily in high resolutions (720p, 1080p, 4K).
  2) Locally exported images or video clips from game engines, allowing for UI-free or clean-background footage in specific scenarios.
  3) Publicly available datasets containing 3D game-themed art assets or screenshots, serving as auxiliary material.
- **Data Types:**
  - Large volumes of static images (used in both the first and second stages for hybrid image-video training).
  - Video clips of varying duration (4–16 seconds, generally at 24 fps, optionally mixing 30 fps or 60 fps).
- **Example Scale:**
  - Images: about 2M to 5M.
  - Videos: around 1M short clips (4–16 seconds each), covering over 100 major 3D open-world game titles (RPG/ACT/FPS, etc.).

b) Filtering and Labeling:

1) **Manual or Semi-Automated Filtering:** Remove non-playable segments, heavily edited footage, or clips with excessive UI elements. Assign key labels: game title, gameplay type, perspective (first-person vs. third-person), and so on.
2) **Clip Segmentation:** Use tools like TransNetV2 or PyScene to detect scene boundaries within longer videos, splitting them into 4–16 second segments. Discard segments shorter than 4 seconds or highly repetitive.

3) **Automated Filtering:**
   - Aesthetic scoring (CLIP-AVA) to remove clips with poor visual clarity.
   - Optical flow or motion detection (UniMatch) to preserve segments with coherent movement, discarding overly static or heavily jittery ones.
   - Content similarity (VideoCLIP) to reduce redundancy within the same game.
   - Camera/character motion analysis (CoTrackerV2) to mark camera trajectories and detect the presence of character movements.

c) Text Caption Generation and Processing:

- **Image Captions:** We apply a general-purpose vision-language model (BLIP2) to generate preliminary labels, then refine or summarize these descriptions using GPT-4o to ensure completeness. Important attributes like scene type, presence of characters, lighting, and style are extracted for each image.
- **Video Captions:** Similar to CogVideoX and GameGenX, we uniformly sample 8 frames per video segment and combine them into a single prompt for GPT-4o to produce a structured description. The resulting captions may include multiple levels of detail (summary, characters, environment, atmosphere, video events or actions). For some high-quality data, we also provide specific interactive instructions derived from frame-to-frame differences ("move right," "dusk setting," "summon mount").

d) High-Quality Character Motion Data: For the third stage, we select data that highlights clear, well-defined human character movements while avoiding scenes where background clutter obscures motion details. Emphasis is placed on maintaining consistent skeletal and motion information. Additional features: keypoints or motion vectors may be extracted to assist in training.

C. Three-Stage Training Flow

The MAG training pipeline can be summarized as follows:

1) **Image Pretraining (Building on FLUX.1 dev Base Weights).** We finetune the FLUX.1 dev pretrained weights on large-scale open-world game images, thereby establishing an initial visual and semantic inductive bias. This step uses only static images, on the order of 2–5 million, captured from various 3D game environments.
2) **Video Backbone (UC-3DMMDiT) Training.** Taking cues from CogVideoX, the model is adapted to video data for temporal consistency and motion generation while still leveraging large-scale image data. Specifically, we filter and recaption videos, combining them with the image dataset (25% images + 75% videos). The training scheme features multi-resolution and multi-length buckets: (256×256, 6 frames), (512×512, 16 frames), (720p, 48 frames), each allocated a certain sampling probability. This unified UC-3DMMDiT backbone supports both unconditional and conditional modes.
3) **APB (Action Prompt Block) Finetuning.** Inspired by GameGenX's InstructNet, this plugin is introduced to handle character motion or action prompts in an "on-demand" manner. While the Stage 2 backbone remains frozen, the APB module learns to incorporate multimodal instructions (text, keyboard input, or auxiliary video cues) to control character actions and environmental transitions. This stage primarily uses carefully curated character motion footage (4–8 seconds each), annotated with structured commands like "move right for three seconds, then jump" or "change weather to overcast. The APB adapts features within each transformer block, enabling the model to respond to real-time user-driven prompts and generate new frames accordingly.

By following this three-stage strategy combined with the associated data preparation procedures, MAG inherits low-level visual features from WFVAE [18] and progressively establishes a robust open-world gaming prior (Stage 1). Then it refines its motion understanding and dynamic generation capability by training on both images and videos at varied resolutions and durations (Stage 2). Finally, the APB plugin is finetuned on high-quality human motion data to enable controllable character actions and real-time interactive generation (Stage 3). This pipeline efficiently leverages existing models and insights from both Cogvideox [19], culminating in a more comprehensive framework for generating "playable" game footage in real time.

E. Metrics Introduction

To evaluate the performance of MAG in game video generation, we adopt a comprehensive set of metrics, some of which are inspired by VBench [20] and VBench++ [21] frameworks. These metrics are designed to address both the temporal quality and the alignment between the generated content and the input prompts, considering the unique demands of game environments. Specifically, we evaluate the consistency of video elements, the realism of motion, the aesthetic quality, and the alignment between generated content and user instructions. Below, we provide a structured overview of each metric:

- **Subject Consistency (SC):** This metric evaluates whether the main subject in the video (e.g., a game character, vehicle, or creature) maintains a consistent appearance across all frames. In the context of game video generation, this is crucial to ensure that protagonists or key elements do not deform or change unexpectedly. We calculate consistency using DINO [22] feature similarity, and prompts are designed to include movable subjects performing dynamic actions to thoroughly test this aspect.
- **Background Consistency (BC):** Background consistency measures the temporal stability of the environment throughout the video. For example, in a game scene, the background (e.g., buildings, landscapes) should not shift, flicker, or distort during gameplay. We evaluate this using CLIP [23] feature similarity to ensure that the background remains coherent and steady.
- **Temporal Flickering (TF):** This metric assesses the temporal stability of high-frequency details across frames. In game videos, flickering can disrupt immersion, especially in textures or lighting effects. We check for flickering by extracting static frames and computing the average absolute difference between consecutive frames, ensuring smooth transitions.
- **Motion Smoothness (MS):** Motion smoothness evaluates whether the movements of characters, objects, or the camera follow realistic and fluid trajectories, reflecting real-world physics. For example, in a game scene, a character's walking or shooting motion should appear natural. We use motion priors from video frame interpolation models to measure the smoothness of generated motion.
- **Dynamic Degree (DD):** While static videos might score well in other consistency metrics, this metric evaluates whether the generated content includes dynamic and substantial motion. For instance, in an FPS game, actions like running, jumping, or explosions should be present to capture the essence of the gameplay. We use RAFT to estimate the level of motion in the synthesized videos.
- **Aesthetic Quality:** This metric measures the artistic and visual appeal of the generated video frames. For game videos, factors like color harmony, scene composition, and photorealism are crucial for maintaining immersion. We use the LAION aesthetic predictor to assess attributes such as layout, richness of colors, and artistic quality.

- **FID:** Fréchet Inception Distance (FID) evaluates the overall fidelity of the generated frames by comparing them to real-world references. A lower FID score indicates that the generated frames are closer in quality to real game content.
- **Color:** This metric assesses whether the colors of objects in the video match the descriptions in the input prompts. For example, if the prompt specifies "a red car," the generated video should include a red car. We use GRiT to annotate and compare the actual and expected object colors.
- **Spatial Relationship (SR):** In addition to generating objects with the correct attributes, it is important to ensure that their spatial relationships align with the prompt. For example, in a game scene, if the prompt specifies "a tree next to a house," the generated video should reflect this arrangement. We adopt a rule-based evaluation method to assess spatial alignment.
- **Scene:** This metric evaluates whether the generated video depicts the correct environment as described in the prompt. For instance, if the input specifies "a battlefield," the video should contain elements associated with a battlefield, not a cityscape. We use Tag2Text to annotate the generated scenes and compare them to the prompt descriptions.
- **Appearance Style (AS):** Style consistency ensures that the generated video matches the stylistic requirements of the prompt. For example, if the prompt specifies "cyberpunk style," the video should exhibit characteristics such as neon lighting and futuristic elements. We calculate CLIP feature similarity between the generated frames and the specified style descriptions to evaluate alignment.
- **Overall Consistency (OC):** This metric provides a comprehensive evaluation of the alignment between the generated video and the input prompt, focusing on both semantic and stylistic consistency. We use ViCLIP to calculate text-to-video alignment based on generic prompts, ensuring that the generated content meets both functional and aesthetic expectations.

This set of metrics, addressing both technical and perceptual aspects, ensures MAG's performance is rigorously evaluated across all critical dimensions relevant to game video generation.

F. Prompt Used in Qualitative experiment

Prompt Used in Qualitative experiment

In a FPS game, players embody a stealthy special forces operative on a covert daytime mission to infiltrate an enemy-controlled island. Navigating a sun-drenched landscape, players move silently through the sandy beaches, their boots leaving tracks in the warm sand that glisten under the relentless sun. As they advance, they enter the dense jungle, where the rustling leaves and snapping twigs underfoot must be navigated with care to avoid detection. The dilapidated structures provide both hazards and opportunities for ambush, with long shadows creating pockets of darkness that can hide an operative or an enemy. Camouflaged enemies patrol the area, their radios crackling with static as they communicate, unaware of the player's presence. Players must use sandbags and debris for cover, their hearts pounding as they listen for the telltale signs of an enemy's approach. Armed with an assault rifle for engagements and a trusty sidearm for close-quarters combat, players must manage their ammunition wisely, all while keeping an eye on the HUD for health, ammo status, and a map that guides them towards their critical mission objectives.

Fig. 1. Prompt:Players navigate through a vibrant, neon-lit metropolis at night. Starting from a skyscraper's helipad, they leap across rooftops, dodging obstacles like police cars and falling banners. Below, the streets are bustling with traffic and pedestrians, with collectible coins and energy drinks scattered about. The game features dynamic elements such as moving elevators and retractable bridges, challenging players to react swiftly against a backdrop of urban noise and light rain. The interface includes an energy bar, score, and timer, adding to the sense of urgency in the run.

G. More qualitative experimental results

Due to the attachment size limitation, only one set of results is presented here. Additional comparisons are available on the MAG project homepage. Please click the link (anonymous) to access the relevant information: https://icme2025sub.github.io/MAG/.

H. Comparison of ablation experiment structures

Due to the attachment size limitation, we have presented the architecture comparison on the MAG project homepage. Please click the link (anonymous) to view the relevant information: https://icme2025sub.github.io/MAG/.

I. Human Evaluation

To assess the real-world performance of MAG, we randomly recruited 100 volunteers and conducted a human evaluation based on four aspects:

(1) **Safety**: Measured by the **Safe Rate**, which represents the proportion of volunteers who deemed MAG-generated content to be safe and appropriate [24]. (2) **Realism**: Measured by the **Confusion Rate**, which indicates the percentage of volunteers who were unable to correctly distinguish between

TABLE I

THE TABLE LISTS MAG'S PERFORMANCE ACROSS THE FOUR HUMAN EVALUATION METRICS: **SAFE RATE**, **REALISM (CONFUSION RATE)**, **INSTRUCTION ADHERENCE**, AND **VIDEO-TEXT CONSISTENCY**. THE **SAFE RATE** WAS DIRECTLY REPORTED BY VOLUNTEERS, WHILE THE OTHER METRICS WERE DERIVED FROM VOTING RESULTS IN FPS GAME SCENARIOS.

Safe rate	Confusion Rate	Instruction Adherence	Consistency
100%	44%	91%	87%

MAG-generated visuals and those produced by a real game engine. (3) **Instruction Adherence**: Evaluates whether MAG-generated visuals effectively respond to user-provided control instructions for character movements. (4) **Video-Text Consistency**: Assesses whether the visuals generated by MAG align with the input text content.

During the experiment, each volunteer participated in 10 voting sessions based on FPS (First-Person Shooter) game scenarios. The results, as shown in TableI, indicate that MAG achieved a **Safe Rate** of 100%, ensuring that all generated content was safe and non-harmful. The **Confusion Rate** was close to 50%, demonstrating that MAG's outputs are approaching the realism of visuals generated by real game engines. Moreover, MAG achieved over 85% in both **Instruction Adherence** and **Video-Text Consistency**, highlighting its strong controllable generation capabilities.

REFERENCES

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[2] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng, "Magicvideo: Efficient video generation with latent diffusion models," 2023.

[3] Wenjing Wang, Huan Yang, Zixi Tuo, Huiguo He, Junchen Zhu, Jianlong Fu, and Jiaying Liu, "Swap attention in spatiotemporal diffusions for text-to-video generation," 2024.

[4] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang, "Modelscope text-to-video technical report," 2023.

[5] Zuozhuo Dai, Zhenghao Zhang, Yao Yao, Bingxue Qiu, Siyu Zhu, Long Qin, and Weizhi Wang, "Animateanything: Fine-grained open domain image animation with motion guidance," 2023.

[6] 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.

[7] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai, "Animatediff: Animate your personalized text-to-image diffusion models without specific tuning," 2024.

[8] Jinbo Xing, Menghan Xia, Yong Zhang, Haoxin Chen, Wangbo Yu, Hanyuan Liu, Xintao Wang, Tien-Tsin Wong, and Ying Shan, "Dynamicrafter: Animating open-domain images with video diffusion priors," *arXiv preprint arXiv:2310.12190*, 2023.

[9] William Peebles and Saining Xie, "Scalable diffusion models with transformers," *arXiv preprint arXiv:2212.09748*, 2022.

[10] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun, "Sora: A review on background, technology, limitations, and opportunities of large vision models," 2024.

[11] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You, "Open-sora: Democratizing efficient video production for all," March 2024.

[12] Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, Tanghui Jia, Junwu Zhang, Zhenyu Tang, Yatian Pang, Bin She, Cen Yan, Zhiheng Hu, Xiaoyi Dong, Lin Chen, Zhang Pan, Xing Zhou, Shaoling Dong, Yonghong Tian, and Li Yuan, "Open-sora plan: Open-source large video generation model," *arXiv preprint arXiv:2412.00131*, 2024.

[13] Genmo Team, "Mochi 1," https://github.com/genmoai/models, 2024.

[14] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter, "Diffusion models are real-time game engines," 2024.

[15] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret, "Diffusion for world modeling: Visual details matter in atari," in *Thirty-eighth Conference on Neural Information Processing Systems*, 2024.

[16] Decart and Spruce Campbell Julian Quevedo, Quinn McIntyre, "Oasis: A universe in a transformer," 2024.
[17] Seung Wook Kim, Yuhao Zhou, Jonah Philion, Antonio Torralba, and Sanja Fidler, "Learning to simulate dynamic environments with gamegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1231–1240.
[18] Zongjian Li, Bin Lin, Yang Ye, Liuhan Chen, Xinhua Cheng, Shenghai Yuan, and Li Yuan, "Wf-vae: Enhancing video vae by wavelet-driven energy flow for latent video diffusion model," *arXiv preprint arXiv:2411.17459*, 2024.
[19] 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.
[20] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu, "VBench: Comprehensive benchmark suite for video generative models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
[21] Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu, "Vbench++: Comprehensive and versatile benchmark suite for video generative models," *arXiv preprint arXiv:2411.13503*, 2024.
[22] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," 2022.
[23] 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.
[24] Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting, "Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.