GameFactory: Creating New Games with Generative Interactive Videos (Supplementary Materials)

Additional results of action control in Minecraft and open-domain scenarios can be found in our homepage at https://yujiwen.github.io/gamefactory/.

A. Details of Minecraft Dataset

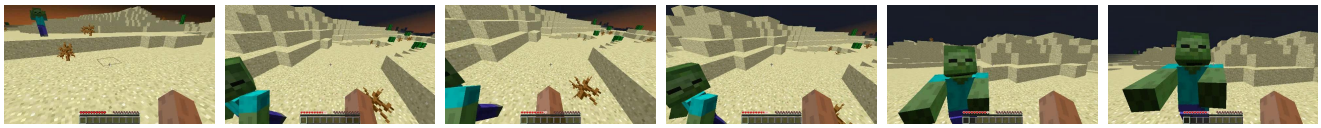
A.1. Basic Information

For data collection, we use Minedojo [3] to obtain Minecraft snapshots which contain a wide array of details within the agent's surroundings. To mitigate biases introduced by human player habits, we uniformly sampled the frequency and duration of each action and randomly combined them to enhance action generalization (e.g., pressing two keys simultaneously or combining keyboard and mouse operations). Subsequently, the agent was connected to the MineDojo simulation environment to execute the action sequences and collect corresponding observations.

To further enhance the diversity and generalization of the generated game scenes, we preconfigured three biomes (forest, plains, desert), three weather conditions (clear, rain, thunder), and six different times of day ("Starting of a day," "Noon, sun is at its peak," "Sunset," "Beginning of night," "Midnight, moon is at its peak," "Beginning of sunrise"). This process resulted in the generation of 2,000 video clips with action annotations, each containing 2,000 frames.

A.2. Data PreProcessing

After obtaining the full video, we preprocess it in two steps: (1) **Slicing:** Randomly sample consecutive sequences of k frames from the full video to form new video clips. Sampling continues until the total number of sampled frames reaches n times the total frame count of the original video. In our experimental setup, $k = 81$, which is slightly larger than the generation length of the video diffusion model (77), and $n = 3$; (2) **Text Annotation:** We use the open-source multimodal large language model MiniCPM-V [10] to annotate the sliced video clips with text. The corresponding prompts and annotation examples are shown in Fig. A.



Q: Describe the scenery and objects in the Minecraft video.
A: The video showcases a **desert biome** in the game Minecraft, characterized by **sandy terrain** and **sparse vegetation**. The environment includes **sand blocks** forming the ground and **cacti** scattered around as the only **plant life** visible. In the background, there are rock formations that add to the arid **landscape's ruggedness**. A notable feature is a **green, pixelated entity** resembling a **zombie-like figure** with outstretched arms, which remains stationary throughout the video. The sky transitions from **orange hues** at the horizon to **darker shades** above, suggesting either **sunrise or sunset**. The player's HUD (Heads-Up Display) is consistently present, indicating health and inventory status, but no actions such as mining or crafting are depicted within these frames.

Figure A. An example of video clip annotation, where words describing scenes and objects are highlighted in red and bolded.

Table A. **Details of Action Space.** The term *Control Signal* refers to the raw input signals utilized for training purposes, while the *Action Interface* represents the corresponding interface in the MineDojo platform that maps these input signals to actionable commands.

Behavior	**Control Signal**	**Action Interface**
forward	W key	Interface_1
back	S key	Interface_1
left	A key	Interface_2
right	D key	Interface_2
jump	space key	Interface_3
sneak	shift key	Interface_3
sprint	ctrl key	Interface_3
vertical perspective movement	mouse movement(yaw)	Interface_4
horizontal perspective movement	mouse movement(pitch)	Interface_5

Move left and encounter an obstacle

Move forward and hit a wall

Figure B. Demonstration of the learned response to collision, one of the most common interactions in Minecraft navigation. Note that the text below each video frame is a descriptive label of the content, not a text prompt provided to the model.

A.3. Details of Action Space

We use the part of the action space of Minedojo [3] which encompasses nearly all actions available to human players, including keypresses, mouse movements, and clicks. We used keypresses and mouse movements as our control signal. The specific binary actions used in our setup are listed in Tab A. Interface_1 to Interface_5 represent different MineDojo interfaces, where mutually exclusive actions (e.g., moving forward and backward) are assigned to the same interface. Mouse movements are represented by the offset of the mouse pointer from the center of the game region. For each frame in the video, we calculate the cumulative offset relative to the first frame, and this absolute offset is used as input to the model.

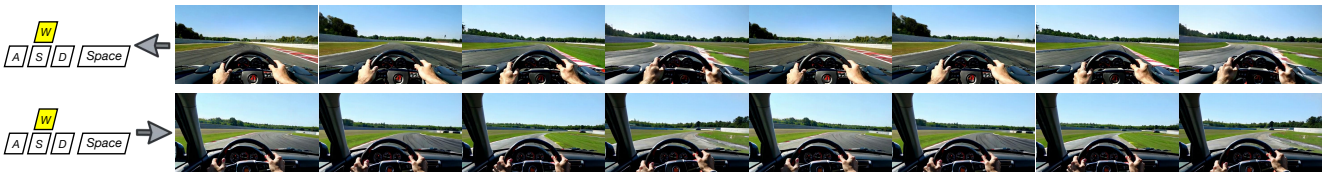
B. Supplementary Experimental Results

B.1. Interaction in generative Minecraft videos.

For navigation agents, collision is one of the most critical physical interactions in a simulator. Since our data collection process involves randomly generated scenes, our dataset naturally includes numerous examples of collisions. In these cases, even when action inputs are provided, the agent's behavior should ideally remain stationary. Such corner cases can impact the learning of the action control module, especially when data volume is limited. However, during inference, we observed that our model has developed collision detection ability and provides appropriate interaction feedback, as shown in Fig. B.

B.2. More Inspiration from Examples of Racing Games

In Fig. C, we discuss an intriguing example. Since our collected Minecraft data is from a first-person perspective, the learned action space primarily generalizes to first-person scenes. Here, we experimented with a racing game scenario using a prompt for a car. Interestingly, we observed that the model's learned yaw control for the mouse seamlessly generalized to steering control in the racing game. Additionally, certain directional controls, such as moving backward or sideways, were diminished, an adaptation that aligns well with typical controls in a racing game, where these actions are rarely needed. This example not only highlights the strong generalization capabilities of our method but also raises the question: could there exist a larger, more versatile action space that encompasses a wider range of game controls, extending beyond first-person and racing games? This example also leads us to wonder whether the racing game scenario could have applications



Prompt: On a racing track, from a first person perspective, one can see holding a steering wheel.

Figure C. Our model demonstrates the ability to generalize to a different game type, a racing game. Interestingly, the yaw control learned in Minecraft seamlessly transfers to steering control in the racing game, while unrelated actions, such as moving backward, left, or right, and pitch angle adjustments, automatically diminish.

in autonomous driving. If we were to collect an action dataset from an autonomous driving simulation environment, could a pre-trained model then generate unlimited open-domain autonomous driving data? Our exploration of scene generalization within generative game engines may hold valuable insights for other fields as well.

C. Potential of Generalizable World Model

We propose that the **GameFactory** we have developed is not merely a tool for creating new games but a **Generalizable World Model** with far-reaching implications. This model has the capability to generalize physical knowledge learned from small-scale labeled datasets to open-domain scenarios, addressing challenges in areas like autonomous driving [4, 5] and embodied AI [2, 7, 9, 11], which also face limitations due to the lack of large-scale action-labeled datasets. The generalizable world model has two key applications from different perspectives:

- **As a data producer:** It transfers knowledge from small labeled datasets to open-domain scenarios, enabling the generation of diverse unlimited action-annotated data that closely approximates real-world complexity.
- **As a simulator:** It provides an environment for directly training agents to perform real-world tasks [1, 6, 8], closely approximating real-world conditions. By enabling controlled and diverse scenario generation, including extreme situations that are difficult to capture in real-world data collection, it facilitates the development of policy models that are exposed to a wide range of environments and interactions, thereby improving their robustness and generalizability, and aiding in overcoming the challenges of sim-to-real transfer.

References

[1] Anurag Ajay, Seungwook Han, Yilun Du, Shuang Li, Abhi Gupta, Tommi Jaakkola, Josh Tenenbaum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[2] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. *arXiv preprint arXiv:2412.03572*, 2024. 3

[3] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 1, 2

[4] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. *arXiv preprint arXiv:2405.17398*, 2024. 3

[5] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023. 3

[6] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 3

[7] Yiran Qin, Zhelun Shi, Jiwen Yu, Xijun Wang, Enshen Zhou, Lijun Li, Zhenfei Yin, Xihui Liu, Lu Sheng, Jing Shao, et al. Worldsimbench: Towards video generation models as world simulators. *arXiv preprint arXiv:2410.18072*, 2024. 3

[8] Yiran Qin, Enshen Zhou, Qichang Liu, Zhenfei Yin, Lu Sheng, Ruimao Zhang, Yu Qiao, and Jing Shao. Mp5: A multi-modal open-ended embodied system in minecraft via active perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16307–16316, 2024. 3

[9] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023. 3

[10] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024. 1

[11] Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024. 3