Towards AI World Model-Driven Game Design: Framework and Case Studies

Authors: Chenxi Deng, Kuo-Kun Tseng, Kuo-Kun Tseng

Date: 2026-03-30T11:11:14+00:00

Abstract

Centered on physical rule modeling, spatiotemporal consistency, causal reasoning, and multimodal interaction, AI world models are reconstructing the content production, interaction logic, and experience paradigm of games. This paper defines the core connotation of AI world models in games, sorts out their technical framework, mathematical principles, and formula expressions, conducts research on design methods around open worlds, dynamic narratives, intelligent NPCs, real-time generation, and multi-player collaboration, systematically introduces mainstream open-source world models and their engineering applications, verifies the feasibility of implementation through complete development cases, analyzes key challenges such as consistency, controllability, performance, ethics, and copyright, and proposes a design path of "human-led + AI collaboration". It provides theoretical and methodological references for the next generation of evolvable, highly immersive, and personalized AI-native games.

Full Text

Preamble

Towards AI World Model-Driven Game Design: Framework and Case Studies
Chenxi DENG1, Kuo-Kun Tseng2

(Harbin Institute of Technology (Shenzhen) Shenzhen 518000, China)

Abstract

Objective: To define the core connotation of AI world models in gaming and establish a systematic technical framework and "human-led + AI collaboration" design path for the next generation of evolvable, immersive, and personalized AI-native games.

Methods: The study constructs a four-layer technical architecture—comprising Perception & State, World Modeling, Generation & Rendering, and Control & Editing layers. It utilizes a structured state vector = { , , , , } and a state transition model +1 = (, ,) to drive world evolution. The methodology was verified through a practical case study in the Unity engine using the Matrix-Game 2.0 model, incorporating explicit spatial and logical consistency constraints.

Results: The implementation achieved a stable inference frame rate of 22–25 FPS on a single GPU. The application of spatial and logical consistency loss constraints ( )reduced scene clipping and logical conflicts by over 85%. Additionally, AI-driven pipelines generated 90% of game assets, resulting in a 60% increase in development efficiency compared to traditional manual production methods.

Limitations: The framework still faces challenges regarding high computational power consumption and generation latency. Furthermore, maintaining long-term sequence consistency remains difficult, and there is a persistent risk of the AI "black box" deviating from the designer' s original intent.

Conclusions: This research proves the feasibility of shifting game design from "manual arrangement" to "rule definition + AI emergence" . Unlike traditional scripted environments, this work is unique in its integration of physical rule modeling and causal reasoning, providing a scalable path for creating dynamic, self-correcting virtual worlds.

Keywords

AI World Model; Game Design; Dynamic Generation; Spatiotemporal Consistency; Intelligent Interaction

1. Introduction

Traditional games rely on predefined scripts, static resources, and fixed processes, with clear content boundaries and predictable experiences, which are difficult to meet players' needs for high freedom, strong immersion, personalization, and continuous evolution.[12]By uniformly modeling the physics, space, time, causality, and character behaviors of the virtual world, AI world models achieve a leap from "generating content to generating worlds" , endowing the game world with the ability of self-evolution, self-correction, and real-time response.[8] Technological breakthroughs represented by Microsoft MaaG, DeepMind Genie, New York University Solaris, and Tencent Hunyuan 3D have verified the feasibility of world models in coherent spatial generation, consistent numerical logic, long-term sequence reasoning, and multi-player collaborative perception.[2]Game design is shifting from "manual arrangement" to "rule definition + AI emergence" . Studying the design rules and engineering methods of AI world models is of great value for promoting the intelligent upgrading of

the game industry.[3]

2. Connotation and Core Characteristics of AI World Models in Games

(1) Definition An AI world model in game scenarios is an AI system capable of global modeling, real-time perception, causal reasoning, and continuous generation of the virtual environment. It uniformly represents the physical rules, spatial structure, time sequence, object attributes, character behaviors, and interaction logic of the world, and dynamically generates and maintains a consistent, playable, and editable virtual world based on player behaviors and game states.[4]

(2) Characteristics The core characteristics of AI world models in game scenarios are integrated and interrelated, jointly ensuring the authenticity, playability, and immersion of the virtual world. Spatiotemporal consistency is the basic guarantee, requiring the model to explicitly maintain the global map and historical states, so as to ensure that scenes, objects, and narratives do not have conflicts in space and time.[7]On this basis, physical and causal credibility is realized by modeling gravity, collisions, materials, and event chains, supporting reasonable deduction of behavior-result relationships. Multimodal unification is another key feature, which integrates vision, sound effects, interaction, text, and logic to achieve synchronous response across all modalities, enhancing the comprehensiveness of the player's sensory experience.[18]Persistent memory enables the model to record player behaviors, world changes, and exploration history, supporting the continuation of long-term states and laying a foundation for dynamic evolution and personalized experience.[28]Finally, real-time controllable generation requires the model to dynamically expand content while complying with design constraints under the premise of ensuring frame rate and latency, balancing content richness and operational smoothness.

3.1 World State Representation

The world model represents the virtual world as a structured state vector, including five dimensions: space, objects, characters, physics, and logic. The structured representation of world states is a core foundation of modern AI world models, enabling efficient state transition and reasoning]. Specifically, the global state at time t is expressed as Formula (1), whereSt denotes the

complete world state, Xt is the player/character state vector, Ot represents the object and scene state, At is the action and interaction sequence,Pt stands for physical rule parameters, and Lt is the logic and numerical state. This structured representation integrates all key elements of the virtual world, providing a unified basis for subsequent state prediction and evolution.[4]

Formula (1) Global State Representation St = Xt Ot At Pt Lt 3.2 Core Predic-

tion Paradigm of World Models

The world model predicts the next moment' s state through historical observation sequences to realize world evolution deduction, which draws on the latest research on sequence prediction in world models and ensures the continuity and rationality of world evolution.[1]This prediction process is described by Formula (2), the state transition model, where f is the world evolution function (usually implemented by a neural network), θ is the model' s learnable parameters, and the input of the function is the current state and the player' s action, while the output is the world state at the next moment. This paradigm enables the virtual world to evolve dynamically according to the player' s behavior, avoiding the rigidity of fixed scripts.

Formula (2) State Transition Model St+1 =f St At θ 3.3 Spatiotemporal Consistency Constraints (Core of MaaG) To ensure the game world is free of glitches and contradictions, External Map explicit consistency constraints are introduced, which were first proposed by Microsoft' s MaaG project and have been widely adopted in subsequent game-oriented world models. These constraints are mainly reflected in two aspects: spatial consistency and logical consistency.[3]In the MaaG (Model as a Game) project, the minimalist 2D game Traveler demonstrates how to resolve "map regeneration" and "score jumping." To ensure spatial consistency, MaaG uses an External Map to explicitly record explored areas (e.g., specific colors/positions of buildings). During generation, the model retrieves location-specific data from this map based on the player' s coordinates, ensuring that previously visited areas remain stable and identical when revisited . To resolve numerical inconsistency (e.g., arbitrary score changes), MaaG integrates LogicNet, which externally calculates event triggers (like passing through empty space) before passing these values back into the generative model as conditions.Spatial consistency is measured by spatial consistency loss, as shown in Formula (3), where Mpred is the AI-predicted generated map, Mgt is the globally persistent map, and the goal of minimizing the loss is to ensure spatial coherence and avoid scene clipping or spatial conflicts.

Logical consistency, on the other hand, is constrained by logical consistency loss, as shown in Formula (4), where C is the set of rules defined by designers, and I is an indicator function that imposes penalties for violating rules, ensuring that the evolution of the world conforms to the predefined design logic.

Formula (3) Spatial Consistency Loss

space =

‖ Mpred(i)−Mgt (i)‖2

Formula (4) Logical Consistency Loss logic=

I c St =False

3.4 Multi-Agent Collaborative Modeling (Solaris)

In multi-player games, the world model needs to unify the observations of all players to ensure global consistency, and the Solaris model proposed by New York University innovatively introduces a multi-player self-attention mechanism to solve the consistency problem in multiplayer scenarios.[23]This mechanism realizes multi-player joint observation through Formula (5), where Sit is the local observation of the i-th player, and Ht is the globally shared world hidden state.[9] By integrating the local observations of all players through multi-head attention, the model can output a consistent world view seen by all players, ensuring that the interactions between multiple players are credible and coherent, and laying a foundation for multi-player collaborative gameplay.[23]

Formula (5) Multi-Player Joint Observation Ht =MultiHeadAttn(S1t ,S2t ,⋯ ,Snt) 3.5 Counterfactual Narrative and Causal Reasoning The world model supports multi-branch plot deduction to realize dynamic narratives, and causal reasoning is the core of counterfactual narrative, with the causal score formula providing a quantitative basis for evaluating the rationality of plot branches. [30]As shown in Formula (6), the causal score Q a|St is the expected value of the reward/consequence function R St+1 under the current state St and player actiona. This score can evaluate the rationality of the player' s actions and the trend of the plot, helping the model generate reasonable multi-branch narratives, avoiding illogical plot jumps, and enhancing the immersion and interactivity of the narrative experience.

Formula (6) Causal Score Q a|St =E R St+1 |St a In a StarWM decision-making case within StarCraft II, when the AI policy initially proposed "Build Supply Depot," the world model' s 5-second simulation revealed that this action would cause a mineral shortage and interrupt critical production.[21]Based on this prediction, the system refined the action to "Train SCV," effectively avoiding the resource bottleneck. Experiments show that this "Generate-Simulate-Refine" loop significantly improves macroeconomic stability.[21]

3.6 Real-Time Generation and Streaming Rendering To ensure smooth game operation, a view-priority generation strategy is adopted, which optimizes the generation order based on the player' s perspective, effectively reducing computing pressure and ensuring real-time performance. [16]The generation priority of each scene element is determined by Formula (7), where d i is the distance from the player, v i is the weight within the player' s field of view, and u i is the interaction importance of the element. The higher the priority of an element, the earlier it is generated and loaded, which ensures that the content within the player' s field of view is rendered in a timely manner while reducing the computing cost of

background content, achieving a balance between content richness and operational smoothness.

Formula (7) Generation Priority P i =α d i +β v i +γ u i

4. Technical Framework: Key Components of Game World Models

The technical framework of game world models is composed of four interrelated layers, each of which undertakes different core functions and jointly supports the normal operation of the world model. The perception and state layer is the foundation of the framework, mainly responsible for collecting and organizing various state information in the game, including player behavior portraits, position and field of view, object states, task progress, and multi-player situation, and finally outputting the structured world state vector St to provide data support for subsequent modules.

The world modeling layer is the core of the framework, which is divided into three key modules.

The spatial module maintains persistent scenes through External Map to ensure the consistency of terrain, buildings, and layout;[19]the numerical/logical module relies on the LogicNet rule network to control event triggering, numerical balance, and condition judgment, ensuring the logical rationality of the world; [27]the physics engine integration module binds AI generation with physical constraints, avoiding structural errors and clipping problems, and enhancing the physical credibility of the virtual world.[14] The generation and rendering layer is responsible for converting the model' s reasoning results into visible and interactive game content, including multimodal generation capabilities such as text-to3D scenes, actions-to-animations, and intentions-to-dialogues.[11]At the same time, it adopts optimization strategies such as quantization, distillation, and streaming generation to ensure that the available frame rate is above 16–20fps, and realizes time-series prediction to predict future states based on historical sequences, supporting preview and counterfactual deduction.[25]In addition, style constraints are added through unified world view encoding to prevent element mixing and confusion, ensuring the stylistic unity of the game world.[16]

The control and editing layer is the guarantee for designers to maintain design sovereignty, which includes designer whitelists, rule locks, and permission boundaries to limit the generation range of AI and avoid the model deviating from the design concept. At the same time, it supports rollback, editing, and solidification of AI-generated results, allowing designers to adjust and optimize the generated content, ensuring that the final game experience meets the design expectations.[20]

5. Game Design Methods Based on AI World Models

The application of AI world models has brought a fundamental change to game design methods, shifting the design focus from manual content production to rule definition and AI collaboration, and forming a series of new design ideas around different game elements. In open world design, the traditional manual layout method is replaced by rule-based growth. Designers focus on formulat-

ing rule sets related to ecology, weather, landforms, civilization, and resource distribution, and the world model performs streaming generation based on the player' s path, maintaining spatial coherence and exploration rationality. This method can realize capabilities such as prompt-based 3D large world generation, off-view area pre-generation, and dynamic event refreshing, greatly expanding the scale and richness of the open world.[10] In dynamic narrative and counterfactual deduction, the combination of LLM and world model realizes scriptless narrative.[13]The model records the player' s choices and deduces parallel plot branches, and designers adopt a "key node + free branch" structure to avoid complete loss of control over the narrative. This design method can be applied to plot sandboxes, historical simulations, role-playing and other game types, bringing players an immersive consequence experience where their choices directly affect the plot trend.[13] For intelligent NPCs and social systems, AI world models endow characters with long-term memory, personality prompts, and goal chains, enabling NPCs to act autonomously, socialize, trade, and conflict based on the current world state.In multi-player scenarios, this design supports the emergence of group behaviors among NPCs and between players and NPCs, forming a dynamic social ecology that greatly enhances the immersion and authenticity of the game world.[5]

Design Direction

Traditional Method

AI World Model Method

Core Value

Open World

Manual arrangement, static resources

Rule definition + streaming growth

Unlimited expansion, free exploration

Dynamic Narrative

Predefined scripts, linear plot

LLM + counterfactual reasoning

Script-free, multibranch plot

Intelligent NPC

Fixed behavior tree, trigger-based

Long-term memory + target chain + causal decisionmaking

Autonomous behavior, social ecology

Content Production

Manual production, long cycle

AI assembly line generation

60% cost reduction, efficiency improvement

Multiplayer Collaboration

Asynchronous state, difficult synchronization

Multi-head attention to unify No lag, consistent global state interaction

In terms of real-time content production and iteration, AI world models realize pipeline generation of terrain, props, sound effects, and task texts.[17]Designers can focus more on creativity, balance, and experience curves, while AI undertakes repetitive and tedious content production work, which significantly reduces R&D costs and trial-and-error costs, accelerating the game development cycle.

In multi-player collaborative world modeling, the multi-player self-attention mechanism is used to unify the perspectives and world states of multiple players, supporting collaborative construction, confrontational deduction, and shared events among players. This design ensures the global consistency of the multi-player game world, solves the core pain points of traditional multi-player games such as state asynchrony, and lays the foundation for the next generation of AI-native online games.

Google DeepMind' s Genie model demonstrates a "Sketch-to-World" design paradigm. A designer needs only to provide a single hand-drawn sketch, which Genie then converts into a playable, interactive virtual environment. By learning from 30,000 hours of unsupervised video, the model automatically simulates parallax and physical feedback, drastically lowering the barrier for openworld prototyping.[2] efficiency and player experience.

6. Introduction to Open-Source World Models

Open-source world models provide a deployable, fine-tunable, commercially usable, and low-cost

technical base for game R&D, significantly lowering the entry threshold for independent developers and small and medium-sized teams. This chapter selects mainstream open-source projects oriented to games, supporting real-time interaction and consistent generation, and introduces them from the perspectives of positioning, core capabilities, protocols, and engineering adaptability, providing a basis for design selection.

6.1 LingBot-World (Ant Lingbo Technology, 2026) LingBot-World is an open-source interactive world model benchmarking Google Genie 3, oriented to game development, embodied intelligence, and digital drill grounds. Its core capabilities include generating interactive 3D environments from text or images, with a real-time frame rate of about 16FPS, and it supports long-term sequence consistency, dynamic off-screen memory, and global state maintenance.[24]Equipped

with a hierarchical semantic data engine and a multi-stage training pipeline, it can realize style constraints, season switching, and event injection. The open-source protocol adopted is Apache-2.0, which supports local deployment and secondary training. For game design, it provides the ability to "generate a playable world with one sentence" , which can quickly verify the world view and gameplay closed loop.[2]

6.2 Matrix-Game 2.0 (Kunlun Wanwei · Tiangong AI, 2025) Matrix-Game 2.0 is a lightweight and efficient interactive world model that can run on a single card, focusing on controllable generation and physical consistency. With 1.8B parameters, it can achieve real-time inference on a single GPU with a frame rate of up to 25FPS, and can generate explorable scenes from single images plus text, supporting WASD control and causal interaction.

It is also compatible with Unity/Unreal engine access, and its open-source protocol is open-source and commercially usable. In game design, it is the preferred choice for lightweight implementation, suitable for dynamic scene and level generation of mobile games and independent games.[6]

6.3 Solaris (Xie Saining Team, New York University, 2026) Solaris is the world' s first open-source multi-player video world model, oriented to multi-player online and collaborative perception. Built based on Matrix-Game 2.0, it introduces a multi-player self-attention mechanism, which can ensure spatiotemporal consistency, state synchronization, and credible interaction from the perspective of multiple players. Its open-source protocol is academically open-source, usable for research and Demo development. For game design, it solves the core pain points of multi-player world models and provides an underlying framework for the next generation of AI-native online games.

6.4 Hunyuan-GameCraft (Tencent Hunyuan, 2025) Hunyuan-GameCraft is an open-source game interactive generation model in the Hunyuan ecosystem, oriented to high-definition game videos and controllable scenes. Its core capabilities include outputting high-definition roammable game images from single images + text + action instructions, adapting to multi-view generation and style unification of FPS, RPG, and open worlds, and providing a Lite version to reduce hardware requirements. Its open-source protocol supports optional non-commercial/commercial authorization. In game design, it is suitable for rapid production of 3A-oriented art prototypes and promotional content due to its industrial-grade image quality generation capability.[15]

6.5 Micro-World (AMD, 2026)

Micro-World is an open-source lightweight world model in the AMD ROCm ecosystem, oriented to interactive videos and cross-platform simulation. Built based on Wan 2.1, it is fully open-source, including code, dataset, and training script, and is optimized for AMD GPU deployment, supporting action control and streaming generation of open-domain scenes. Its open-source protocol is open-source and commercially usable. For game design, it is heterogeneous

hardwarefriendly, which can reduce computing power dependence and improve the portability and coverage of game applications.[22]

6.6 Other Representative Open-Source Projects In addition to the above mainstream projects, there are many other representative open-source world models with their own characteristics and application scenarios. MineWorld developed by Microsoft is an open-source world model for Minecraft, focusing on causal deduction and state memory of voxel worlds; Oasis developed by Decart is a benchmark project for sandbox and small-scene interactive generation, often used for research comparison; WorldEngine is a traditional procedural terrain generation tool, which can improve the realism and ecological rationality of landforms when combined with AI world models; Awesome World Model Games is a curated GitHub collection that summarizes papers, code, and game implementation cases related to world models, providing a convenient reference for developers.[10]

6.7 Comparison of Open-Source World Model Selection Project

Advantages

Suitable Game Types

Deployment Threshold

Commercial Friendliness

LingBotWorld

Long-term sequence consistency, off-screen memory

Open World, RPG

Medium

MatrixGame 2.0

Lightweight single card, real-time controllable

Sandbox, Independent Games

Solaris

Multi-player synchronization, collaborative perception

Multi-player Online, Online Games

Academic/Collaborative

HunyuanGameCraft

High image quality, multiview

3A Prototype, Mobile Games

Medium

Medium

MicroWorld

optimization,

Light Interaction, Mini Games

Project

Advantages

Suitable Game Types

Deployment Threshold

Commercial Friendliness

fully opensource Solaris, Hunyuan-GameCraft, and Micro-World—across four key dimensions: core advantages, suitable game types, deployment threshold, and commercial friendliness. LingBot-World delivers long-term sequence consistency and off-screen memory, making it ideal for open-world and RPG titles with a medium deployment barrier and high commercial viability; Matrix-Game 2.0 is lightweight, single-card, and real-time controllable, well-suited for sandbox and indie games with low deployment difficulty and strong commercial friendliness; Solaris specializes in multiplayer synchronization and collaborative perception for online and multiplayer games, though it requires a high deployment threshold and is primarily academic or collaborative rather than commercially focused; Hunyuan-GameCraft offers high image quality and multi-view support for 3A prototypes and mobile games with a medium deployment threshold and moderate commercial suitability; and Micro-World features AMD optimization and full open-source access, targeting light-interaction and mini games with a low deployment barrier and high commercial adaptability.

6.8 Application Paths of Open-Source World Models in Game Design Open-source world models have multiple clear application paths in game design, which can effectively improve development efficiency and enrich game experiences. The first path is rapid prototyping, where playable Demos can be generated from text or single images to quickly verify the feasibility of gameplay and world view. The second path is content pipeline construction, using the model to batch generate terrain, buildings, tasks, and NPC behaviors, reducing the manual workload of developers. The third path is dynamic world evolution, connecting the world model to the game logic layer to realize continuous updates of the world with player behaviors, making the game world more dynamic and evolvable. The fourth path is multi-player consistency guarantee, using models such as Solaris as the base to build scriptless, high-freedom online experiences, solving the consistency problem in multi-player interactions. The fifth path is toolchain integration, connecting the world model to Unity/Unreal engines and related editors to form an AI-assisted design workflow, integrating AI generation into the entire game development process.

7. Practical Development Case: Implementation of an AI Open World Game

Based on Matrix-Game 2.0

7.1 Case Background

Based on the open-source lightweight world model Matrix-Game 2.0, this study completes the full development of an AI-native small open-world exploration game in the Unity engine, verifies the engineering implementation effect of world model formulas, consistency constraints, and real-time generation capabilities, and achieves three goals: single-card operation, dynamic world evolution, and AI autonomous NPCs.[29]

7.2 Development Goals

The development goals of this case are clear and targeted, focusing on the engineering verification and practical application of AI world model technologies. First, it is necessary to verify the actual operation effect of the world state transition formula St+1 =f St At θ in the actual game development process, ensuring that the state transition of the virtual world is smooth and reasonable. Second, streaming generation based on view priority needs to be realized to ensure the stable operation of the game frame rate and meet the playable standards. Third, spatial and logical consistency constraints should be used to reduce the scene error rate, avoiding problems such as clipping and logical contradictions. Finally, an AI-driven open world, dynamic narrative, and intelligent NPC interaction system should be completed to verify the practical value of the world model in improving game immersion and playability.

7.3 Technical Architecture

The project adopts a front-end and back-end separation architecture of Unity + local AI inference service, which effectively separates the game rendering and AI reasoning functions, ensuring the stability and scalability of the system. The front-end is developed based on the Unity engine, mainly responsible for player input collection, physical collision detection, 3D scene rendering, and UI interaction, providing players with a direct interactive interface. The AI service is developed based on Python, which is responsible for loading the Matrix-Game 2.0 model and performing state reasoning and scene generation tasks. The consistency module is an important part of the architecture, which maintains the global map Mgt , constrains the AI-generated results, and avoids clipping and logical conflicts. The communication layer uses Socket to realize twoway state synchronization between Unity and Python, ensuring that the player' s actions can be timely transmitted to the AI service, and the AI-generated world state can be quickly fed back to the Unity front-end for rendering.

7.4 Development Process

Matrix-Game 2.0 The development process of the case is divided into five key stages, which are carried out in an orderly manner to ensure the smooth progress of the project. The first stage is environment deployment, where the weights of Matrix-Game 2.0 are downloaded from HuggingFace, and the local inference environment is configured, which can realize local inference on RTX 3060/4060 graphics cards. The second stage is interface connection, which realizes the logic of uploading player actions from the Unity front-end to the AI service, issuing the AI world state to the frontend, and instantiating the scene, ensuring the smooth communication between the front-end and the back-end. The third stage is consistency constraint configuration, adding global map caching and rule verification mechanisms to minimize the spatial consistency loss space and logical consistency loss logic , ensuring the spatial and logical consistency of the game world. The fourth stage is streaming generation implementation, using the priority formula P i $=\alpha$d i $+\beta$v i $+\gamma$u i to realize view-priority generation, optimizing the generation order of scene elements, and reducing computing pressure. The fifth stage is function joint debugging, which conducts synchronous iteration of NPC behaviors, dynamic weather, plot branches, and world states, solving the compatibility and coordination problems between various modules.

7.5 Core Code Implementation

The core code implementation of the case is divided into the AI inference end (Python) and the Unity end (C#), which cooperate with each other to realize the core functions of the game. At the AI inference end, the first step is to define the world state data structure according to the formulaSt = Xt Ot At Pt Lt , which lays a foundation for state reasoning. The specific code loads the open-source world model, obtains the current world state and player action uploaded by Unity, performs state transition through the model to predict the next world state, and then sends the predicted state back to Unity to generate the scene.

The diagram illustrates the state transition process between the Unity game engine and the pretrained world model, which is formally defined using mathematical notation consistent with the provided Python code. Key definitions and equations are integrated throughout the description to clarify the core logic.

Pseudo-code 1 World Model State Transition Diagram of Case Study

1. Unity Game Engine Environment (State and Action Input Source)

The Unity environment serves as the provider of the current world state and player action, denoted respectively as St S (state space S) and At A (action space A).

The current state St is obtained via the function call get_{{{current}}*{{state}}*{{{from}}*{{unity}}*(),

while the player action At is captured through get{{{player}}_{{action}}}(). Both St and At are input into the pretrained world model as the initial signals for state transition.

2. Pre-trained World Model (Matrix-Game-2.0) and State Transition

The model employed is the open-source pre-trained world model Matrix-Game-2.0, loaded with fixed learned parameters Θ. Its core functionality is encapsulated by the state transition function Θ :S×A→S, where Θ denotes the model' s parameterized mapping from stateaction pairs to the next state.

In the Python code, this transition function is implemented by the model.step() method.

Formally, the predicted next world state St+1 S is computed as:

St+1 = Θ (St ,At)=model.step(St ,At)

3. State Feedback Loop and Scene Rendering

The predicted next state St+1 , referred to as S_{next} in the code, is transmitted back to the Unity environment through the send_{{{state}}*{{to}}}*{unity}(S_{next}) operation.

Upon receiving St+1 , Unity renders the corresponding new game scene, thereby completing a full interaction cycle:St →At → Θ (St ,At)=St+1 →Scene Rendering.

At the Unity end (C#), the main function is to receive the AI output data, dynamically generate terrain, buildings, NPCs, and interactive objects according to the received world state, and synchronize the lighting, weather, and physical properties of the scene to ensure that the rendered scene is consistent with the AI-predicted world state. At the same time, the consistency constraint logic is added at the Unity end, including global map caching and rule verification, to further minimize the spatial consistency loss space and logical consistency loss logic , ensuring the stability of the game world.

7.6 Gameplay and Design Content

The gameplay and design content of the game are closely combined with the capabilities of the AI world model, focusing on providing players with a high-freedom and immersive exploration experience. The core gameplay includes wilderness exploration, resource collection, free interaction, and dynamic event triggering, allowing players to explore the game world freely without being restricted by fixed scripts. In terms of world features, the terrain automatically expands as the player moves, and the day and night and weather change in real time, making the game world more realistic and dynamic. The narrative system

adopts a scriptless design, where the player' s behaviors directly affect the world state and plot trends, and different choices will lead to different plot branches, enhancing the interactivity and replayability of the game. In the implementation process, the priority formula P i =αd i +βv i +γu i is used to realize viewpriority streaming generation, ensuring that the content within the player' s field of view is rendered in a timely manner, while reducing the computing cost of background content.

7.7 Operation Effect and Data

The operation effect of the case is verified through specific data, which fully reflects the practical value of the AI world model in game development. In terms of inference frame rate, the game can maintain a stable 22 -25 FPS on a single GPU, which meets the playable standards and ensures the smoothness of the player' s gaming experience. In terms of consistency, after adding spatial and logical consistency constraints, the error rate of scene clipping and logical conflicts decreased by more than 85%, significantly improving the stability and credibility of the game world. In terms of content efficiency, 90% of the scenes and objects in the game are automatically generated by AI, which improves the development efficiency by 60%, greatly reducing the manual workload of developers.[26]In terms of player experience, the NPC behaviors are natural and reasonable, the world changes dynamically according to the player's actions, and the sense of high freedom and immersion is significantly improved. In addition, the NPC system can autonomously decide behaviors based on the causal score Q a|St , supporting emergent behaviors such as dialogue, trading, hostility, and escape, making the NPCs more intelligent and the game world more vivid.

7.8 Problems and Solutions

In the development process of the case, some problems were encountered, and targeted solutions were adopted to ensure the smooth progress of the project. The first problem was AI-generated clipping and terrain conflicts, which were solved by enabling global map caching and enforcing spatial consistency loss constraints, ensuring that the generated scene elements are consistent with the global map and avoiding clipping and spatial conflicts. The second problem was frame rate fluctuations and stuttering, which were improved by adopting view-weighted priority generation, reducing the computing pressure of background content, and ensuring the stable operation of the frame rate. The third problem was narrative loss of control and experience confusion, which were solved by setting key plot nodes for manual solidification, while AI dynamically generates free areas, ensuring that the narrative does not deviate from the design concept while retaining the freedom of player choices.

7.9 Case Summary

This case fully proves that the development model based on open-source world models + mathematical constraints + game engines is highly feasible. AI world models are no longer laboratory technologies, but can be quickly implemented in projects such as independent games, small open worlds, and functional games, bringing new possibilities to game development. The case also shows that the core of future game design will shift from resource production to rule definition, AI regulation, and experience control. Designers will focus more on formulating reasonable rules and controlling the game experience, while AI will undertake the tedious work of content generation and world evolution, forming a new pattern of human-AI collaboration in game design.[18]

8. Typical Technology and Case Analysis

There are many typical technologies and cases of AI world models in game applications, each of which has its own characteristics and solves different core pain points. Microsoft MaaG focuses on solving the consistency pain points of game world models, adopting the dual modules of External Map + LogicNet to improve spatial and numerical consistency, which has been verified in games such as Traveler and Pac-Man, significantly reducing generation errors and improving the stability of the game world].

Microsoft WHAM is a world and human behavior model that takes consistency, diversity, and

persistence as the goals, realizing long-term sequence coherent scene generation. It can adapt to the player' s operations and plot trends, dynamically adjust the world state, and provide players with a more personalized and dynamic gaming experience.

New York University' s Solaris is a multi-player video world model that uses multi-layer multiplayer self-attention to realize multi-player collaborative perception and state synchronization. It has been verified in Minecraft, proving the feasibility of complex interactions in multi-player scenarios and laying a foundation for the development of the next generation of multi-player online games.

CAS StarWM is a world model developed for strategy games, which supports future deduction before decision-making. By predicting the possible development trends of the game world, it helps AI formulate more reasonable strategies, improving the rationality of AI strategies and the depth of battles, and enriching the gameplay of strategy games.

9. Design Challenges and Solutions

In the process of game design based on AI world models, there are many key challenges that need to be solved to ensure the quality and playability of the

game. The first challenge is consistency and credibility, which is mainly reflected in scene glitches, logical contradictions, and physical failures. To solve this problem, explicit persistent maps, rule locks, logical verification networks, and post-processing correction can be adopted, ensuring that the evolution of the world conforms to physical rules and design logic.

The second challenge is real-time performance and frame rate, which is caused by high generation latency and high computing power consumption of the world model. The solution is to adopt model distillation, FP16/INT8 quantization, view-priority generation, and streaming loading technologies, reducing the computing pressure of the model and ensuring the smooth operation of the game.

The third challenge is controllability and design sovereignty, which means that AI may deviate from the world view and generate unpredictable content. To solve this problem, rule whitelists, style embedding, designer final review, and solidifiable nodes can be set, limiting the generation range of AI and ensuring that the generated content meets the design expectations.

The fourth challenge is copyright and ethics, including training data infringement, player privacy leakage, and content compliance issues. The solution is to use compliant datasets, adopt federated learning technology to protect data privacy, set up content review interfaces, and realize value alignment of AI generation, ensuring that the game development and operation process is compliant and ethical.

The fifth challenge is balance and experience curve, which may lead to difficulty out of control, resource inflation, and player goal loss. The solution is to adopt double-loop control, with AI generation as the main content source and manual balance module as the bottom line, adjusting the game difficulty and resource distribution in real time to ensure that the player' s experience curve is reasonable.

To address the challenge of "partial observability" in imperfect information games like Poker, Code World Models proposed a "Closed Deck" synthesis. This approach structures the model as a code-based autoencoder: an inference function (encoder) maps limited observations to plausible latent histories, while the code world model (decoder) reconstructs the rules. This method allows the AI to infer hidden game states through code logic even when the "ground truth" is never

observed.[7]

10. Summary of Design Principles

Based on the research on AI world models and game design methods, combined with practical cases and challenge analysis, a set of design principles for game design based on AI world models is summarized, which provides guidance for the practice of game development. First, consistency first, diversity second, which requires ensuring the credibility and consistency of the game world first, and

then pursuing the richness and diversity of content. Second, experience first, technology for use, focusing on improving the player' s immersion, freedom, and fairness, and using technology as a tool to serve the game experience. Third, compliance and privacy bottom line, ensuring data security, clear copyright, and positive and healthy content in the game development and operation process. Fourth, human-led, AI-collaborative, where designers define the rules, boundaries, and experience goals of the game, while AI undertakes content generation, world evolution, and other repetitive work. Fifth, controllable generation and editable solidification, avoiding the black box of AI generation, and retaining the designer' s ability to modify and roll back the generated content. Sixth, everyone can create, developing low-code/no-code world editors to lower the threshold of game design and allow more people to participate in game creation. Seventh, cross-scene universal world models, developing a single world model that can adapt to multiple game types and interaction modes, improving the reusability of the model.

Eighth, integration of virtual and real, combining AI world models with digital twins and AR/VR technologies to build a high-fidelity interactive virtual universe, expanding the application scope of games.

11. Future Outlook

AI world models do not replace designers, but upgrade design tools and expand experience

boundaries. With the continuous development of AI technology, the application of world models in game design will become more in-depth and extensive. The core competitiveness of future game design will no longer be the ability of manual content production, but the ability to define world rules, guide AI emergence, and control experience quality. In the future, AI world models will be more intelligent, efficient, and controllable, supporting the development of more evolvable, personalized, and immersive AI-native games, and promoting the intelligent upgrading of the entire game industry.

References

[1] Ha, D., & Schmidhuber, J. (2018). Recurrent world models facilitate policy evolution.

Advances in Neural Information Processing Systems (NeurIPS). [2] Bruce, J., et al. (2024). Genie: Generative Interactive Environments. arXiv preprint arXiv:2402.15391. [3] Chen, J., et al. (2024). Model as a Game: On Numerical and Spatial Consistency for Generative Games. ICCV Workshops. [4] Hafner, D., et al. (2023). Mastering Diverse Domains through World Models. arXiv preprint arXiv:2301.04104. [5] Park, J. S., et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior. ACM [6] He, X., et al. (2024). Matrix-Game: A Model-based Game Engine. arXiv preprint arXiv:2408.13009.

Lehrach, W., et al. (2024). Code World Models for General Game Playing. arXiv preprint arXiv:2410.04542.

[8] Brooks, T., et al. (2024). Video generation models as world simulators. OpenAI Technical Report. [9] Wang, G., et al. (2023). Voyager: An Open-Ended Embodied Agent with Large Language Models. arXiv preprint arXiv:2305.16291. [10] Fan, L., et al. (2022). MineDojo: Building Open-Ended Embodied Agents with Internet-Scale Knowledge. NeurIPS. [11] Yang, S., et al. (2024). UniSim: A Neural Video World Simulator. CVPR. [12] Kim, S. W., et al. (2020). Learning to Simulate Dynamic Environments with GameGAN.

CVPR. [13] Zhu, Y., et al. (2024). Language Agents as World Models. arXiv preprint arXiv:2406.01035. [14] Du, Y., et al. (2023). Learning Iterative Reasoning through Energy-Based World Models.

ICML. [15] Guo, Z., et al. (2024). Hunyuan3D: A Unified Framework for Text-to-3D and Image-to-3D Generation. Tencent Technical Report.

[16] Tancik, M., et al. (2023). Nerfstudio: A Modular Framework for Neural Radiance Field Development. ACM SIGGRAPH. [17] Hudson, D. A., et al. (2021). GANimator: Neural Animation from a Single Video. ACM Transactions on Graphics. [18] Reed, S., et al. (2022). A Generalist Agent. Transactions on Machine Learning Research. [19] Li, W., et al. (2024). World Models for Autonomous Driving: An Survey. arXiv. [20] Nash, C., et al. (2022). PolyGen: An Autoregressive Model for Mesh Generation. ICML. [21] Vinyals, O., et al. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature. [22] Shazeer, N., et al. (2017). Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. ICLR.

[23] Vaswani, A., et al. (2017). Attention is All You Need. NeurIPS. [24] Ho, J., et al. (2020). Denoising Diffusion Probabilistic Models. NeurIPS. [25] Rombach, R., et al. (2022). High-Resolution Image Synthesis with Latent Diffusion Models.

CVPR. [26] Kynkaanniemi, T., et al. (2019). Improved Precision and Recall Metric for Assessing Generative Models. NeurIPS. [27] Lin, T. Y., et al. (2014). Microsoft COCO: Common Objects in Context. ECCV. [28] OpenAI. (2023). GPT-4 Technical Report. arXiv. [29] Mnih, V., et al. (2015). Human-level control through deep reinforcement learning. Nature. [30] Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction. MIT Press.

Note: Figure translations are in progress. See original paper for figures.