Latent Space Prediction Model in Game Playing

Jingyuan Liu
1012871910, jyuan.liu@mail.utoronto.ca
github.com/jerryliujy/improved-World-Model-2018

Introduction

Human beings develop intuitive physical common senses through continuous perception and action. In *Thinking, Fast and Slow* (Kahneman, 2011), the cognitive processes of human beings are classified as System 1 (fast, intuitive) and System 2 (slow, logical). System 1 excels in quick, heuristic decisions, while System 2 relies on logical reasoning for more accurate judgments.

Recent advancements in reasoning LLMs represent a significant step in the evolution of deep learning, especially in System 2-style reasoning shown in deep reasoning models like Openai's o3 and Deepseek's r1. However, in multi-modal scenarios, current large models often fail to generalize to the physically grounded world, where logic is more intuitive. For instance, a human instinctively knows to grasp a knife by its handle, not its blade. Teaching a robotic agent this same behavior requires extensive training data and a suitable model architecture. Such tasks, trivial for humans, remain laborious for deep learning models.

Deep neural networks have achieved remarkable success in domains such as image classification, audio recognition, and text generation, but face significant limitations in areas like video generation and motion planning. A promising solution is to equip models with a better understanding of real-world physics. To model these physics and dynamics, this project is built upon the idea of **world model**: a generative model trained to learn and predict an environment's dynamics from raw sensory inputs. The core concept is to simulate the environment's dynamics through state transitions. An optimally trained model should faithfully simulate the real world, aiding downstream applications such as execution guidance or policy exploration in a virtual environment.

The model developed in this project derives from the World Model proposed by Ha and Schmidhuber (2018), which combines a Variational Autoencoder (VAE) for perception with an MDN-RNN for prediction. I introduce two key architectural enhancements to improve upon this framework:

1. The VAE is replaced with a **Vector Quantized-Variational Autoencoder (VQ-VAE)** (van den Oord et al., 2017) to achieve higher-fidelity and more stable visual representations.

2. The RNN-based predictor is replaced with a **Transformer**, which is better suited for modeling the long-range dependencies inherent in predicting future states.

The training process involves three stages, and the model is trained in an unsupervised manner. For evaluation, I plan to place my model inside game environments. Game playing provides controlled, repeatable simulations of real-world dynamics, causality, and agent interaction, which is perfect for developing and evaluating models of physical understanding.

Illustration

The dataset consists of episodes, where each episode is a sequence $(s_0, a_0, r_0, , \dots, s_{T-1}, a_{T-1}, r_{T-1}, s_T)$, with s, a, and r representing the state (image observation), action, and reward, respectively. Figure 1 illustrates the model architecture. The training process is divided into three stages. In Stage 1, only the VQ-VAE is trained to learn a robust latent representation for image observations. In Stage 2, the VQ-VAE is frozen, and the Transformer model is trained to predict the latent representation of the next state using the current latent state and action as input. In Stage 3, both the VQ-VAE and Transformer are frozen, and the controller is trained using an evolution strategy. At test time, the controller generates an action, and the environment provides the subsequent state as feedback. For more details, please refer to Section Data Processing and Section Architecture.

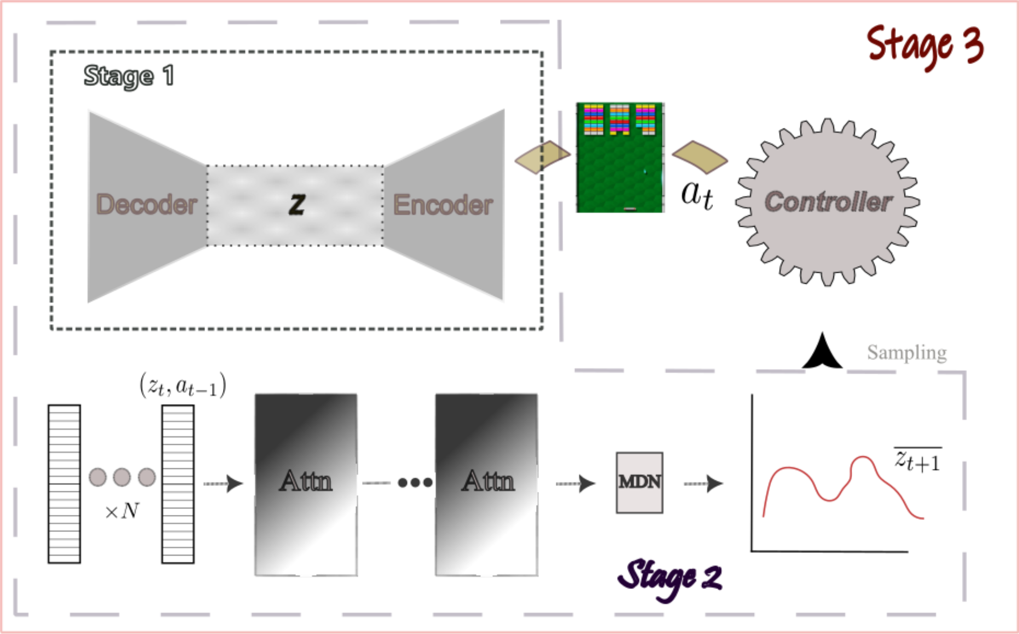

Figure 1: Diagram of the model architecture.

Background & Related Work

The term **"world model"** refers to a generative model that learns a compressed, internal representation of an environment to predict future states. A key feature of this approach is its ability to reason about the world with an implicit understanding of physics and causality, moving beyond pattern generation to grounded simulation. Research in this active field can be broadly categorized into two interconnected efforts: improving the inner space representation and advancing future state prediction (Ding et al., 2025).

The foundation of modern world models was laid by works like Ha and Schmidhuber (2018), which used a VAE to encode observations into a latent space where future states were predicted. This paradigm of learning dynamics in a compressed space proved highly effective. Subsequent research has significantly advanced the sophistication of this inner space, most notably through the Dreamer series (Hafner et al., 2019, 2020, 2024; Zhao et al., 2024), which introduced the Recurrent State-Space Model (RSSM) to jointly learn deterministic and stochastic dynamics, leading to state-of-the-art performance in model-based reinforcement learning.

Concurrently, another branch of research has focused on scaling up future state prediction using massive generative models, exemplified by OpenAI's Sora (Brooks et al., 2024) and Google's Genie (Bruce et al., 2024). These models demonstrate a remarkable ability to generate high-fidelity, long-horizon video sequences. However, they often rely on immense scale rather than novel architectures tailored for physical modeling, and can still struggle with ensuring long-term causal and physical consistency.

Another interesting architecture is JEPA(Joint Embedding Prediction Architecture). It extends the idea of latent space training from (Ha and Schmidhuber, 2018), while focusing on predicting future representations in an abstract embedding space across various modalities. The input modality is not only image or video(Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025; Bardes et al., 2023), but also language(Huang et al., 2025), audio(Fei et al., 2024), 3D space(Saito et al., 2025), graph(Skenderi et al., 2025), and time series(Verdenius et al., 2024).

While these avenues have pushed the boundaries of model-based RL and large-scale generation, the foundational architecture of the 2018 World Model remains a subject of interest. Its simple, modular design makes it an excellent testbed for understanding the impact of individual components. This work develops a model that modernizes the original 2018 World Model architecture. By incorporating a more powerful representation module (VQ-VAE) and prediction module (Transformer), this project aims to achieve improved performance in both state representation and future prediction.

Data Processing

This work evaluates the proposed model on two distinct Gymnasium environments (Brockman et al., 2016): CarRacing-v2 (Klimov, 2016) and Breakout-v5 (from the Arcade Learning Environment (Bellemare et al., 2013; Machado et al., 2018)).

- **CarRacing-v2** CarRacing is a 2D top-down racing environment that requires the agent to learn from pixels. The track is generated randomly for each trial, and the agent is rewarded for visiting as many tiles as possible in the least amount of time. Specifically, The reward is -0.1 every frame and +1000/N for every track tile visited, where N is the total number of tiles visited in the track. The agent controls three continuous actions: steering left/right, acceleration, and brake.
- **Breakout-v5** Breakout is a game where the player move a paddle and hit the ball in a brick wall at the top of the screen. The goal is to destroy the brick wall. The reward returned depends the number of bricks destroyed. The agent controls four discrete actions: noop, fire, left and right.

The choice of these two environments is deliberate, as they present complementary challenges that allow for a thorough evaluation of the world model's capabilities. The original 2018 World Model was benchmarked on CarRacing; I include Breakout to test the model's performance in a setting with fundamentally different characteristics. The key distinctions are:

1. **Action**: CarRacing features a continuous action space (steering, acceleration, brake), posing a difficult control problem. In contrast, Breakout uses a simple discrete action space (noop, fire, left, right).

2. **Dynamics**: The physics in CarRacing are relatively smooth and predictable. Breakout's dynamics are highly sensitive and chaotic, where minor changes in paddle position can lead to vastly different ball trajectories.

3. **Object**: The CarRacing environment is visually simple and mostly static, containing only the car and the track. Breakout features multiple dynamic objects (paddle, ball, bricks) with varying states and positions.

4. **Reward**: CarRacing provides a dense reward signal, offering constant feedback. Breakout has sparse rewards, granted only when bricks are destroyed, making credit assignment more challenging.

Evaluating our model across these divergent settings provides a robust assessment of its ability to learn different types of environmental dynamics and visual complexities.

The data collection and pre-processing pipelines are tailored for each environment, so I will discuss them one by one.

- **CarRacing-v2** I plan to generate a custom dataset following the methodology of the original World Model paper (Ha and Schmidhuber, 2018). The data generation script is copied from this repo, and you can refer to Algorithm 1 for more details. The process is speeded up with CPU parallelization, and the dataset is constructed in HDF5 format. After collection, the raw observation contains the game information display at the bottom, and it needs to be cropped out.
- **Breakout-v5** For Breakout environment, I utilize the pre-collected expert-v0 dataset available through the Minari library (Younis et al., 2024), which can be accessed at this URL. The expert agent implemented through CleanRL PPO Impala can be used to generate more training data. After data collection, to reduce computation and storage, the input image will be downsampled to smaller resolutions like 96×96. Then the image is cropped out to avoid

Algorithm 1 Pseudocode for CarRacing Dataset Generation

```
procedure GENERATE_WORLD_MODEL_DATASET($num\_episodes, max\_steps, num\_workers$)
    // Phase 1: Generate episode data in parallel.
    for each worker from 1 to $num\_workers$ do
        In Parallel Do:
            Assign a subset of episodes to this worker.
            For each assigned episode:
                Run a simulation loop for $max\_steps$.
                    At each step, store the processed image, action, reward, and done status.
                    If the episode terminates early, reset the environment and continue.
                Save the complete episode data to a unique temporary file.
    end for
    wait for all workers to finish.

    // Phase 2: Consolidate all temporary files into a single dataset.
    Get a list of all temporary episode files.
    Create a final, consolidated HDF5 file.
    Initialize datasets (‘images‘, ‘actions‘, etc.) in the final file sized for all episodes.
    for each temporary episode with index $i$ do
        Read the data from the temporary file.
        Write this data into the $i$-th position of the datasets in the final file.
        Delete the temporary file.
    end for
    return the consolidated dataset file.
end procedure
```

unrelated information at the top and the bottom. Since color information is not critical for gameplay, all images are converted to greyscale for further training and evaluation.

Architecture

The model consists of three components-perceptor, predictor and controller.

1. **Perceptor**: The Perceptor's role is to encode high-dimensional pixel observations into a low-dimensional latent representation. This component is trained in an unsupervised manner to reconstruct the original image from its latent encoding.

The original World Model employed a Variational Autoencoder (VAE). However, VAEs are susceptible to "posterior collapse," a training pathology where the model learns to ignore the latent variable to minimize the KL divergence term in its loss function (Lucas et al., 2019). To mitigate this issue, I replace the VAE with a Vector Quantized-Variational Autoencoder (VQ-VAE). The VQ-VAE leverages a discrete, learnable codebook to map encoder outputs to the nearest codebook entry. This design enforces a richer, more structured latent space and effectively prevents posterior collapse.

Regarding implementation, the encoder is a Convolutional Neural Network (CNN) with residual connections, and the decoder is its symmetric counterpart, using transposed convolutions for upsampling. While the decoder is essential for training, it is discarded during inference.

2. **Predictor**: The Predictor is responsible for modeling the temporal dynamics of the environment within the latent space. Its goal is to predict the next latent state z_{t+1} given a history of past latent states and actions.

The 2018 World Model used an MDN-RNN for this task. While effective, RNNs are inherently sequential, limiting parallelization, and struggle to capture long-range dependencies due to the vanishing and exploding gradient problems. Consequently, I replace the recurrent backbone with a Transformer architecture. Specifically, I employ a decoder-only Transformer (Vaswani et al., 2017), which is well-suited for autoregressive sequence generation. Causal masking is applied to ensure that the prediction for a given timestep only depends on past information. At each timestep t, the Transformer takes a sequence of the last N state-action pairs $[(z_{t-N+1}, a_{t-N}), (z_{t-N+2}, a_{t-N+1}), \ldots, (z_t, a_{t-1})]$ as input and outputs z_{t+1}. The context window N may be kept relatively small, as the game environments chosen require more immediate reactivity than long-term planning.

While the Transformer backbone is powerful, it is inherently deterministic. To model the stochastic nature of the game environments, I retain the Mixture Density Network (MDN) from the original architecture as the prediction head. The MDN takes the final output from the Transformer and projects it into the parameters (means, variances, and mixture weights) of a Gaussian Mixture Model (GMM). Instead of a single deterministic prediction, this allows the model to output a probability distribution over possible next states. Sampling from this distribution enables the Controller to explore more diverse trajectories during planning.

3. **Controller**: The Controller's objective is to determine the optimal action a_t given the current state representation from the Perceptor and Predictor. Since evolutionary algorithms or reinforcement learning is not available for this project, I plan to adopt the same simple yet effective architecture and training strategy as the 2018 World Model: a single-layer linear network trained with the Covariance Matrix Adaptation Evolution Strategy (CMA-ES). This can be trivial to train with the script provided in 2018 World Model, and therefore does not introduce anything beyond the class.

As the Controller is trained separately while the Perceptor and Predictor are held fixed, this straightforward design allows me to isolate and clearly evaluate the impact of the architectural improvements to the world model itself. More complex policy learning methods (e.g., actor-critic reinforcement learning) are viable alternatives but are left as avenues for future studying.

Baseline Model

- **World Model (2018)(Ha and Schmidhuber, 2018)** 2018 World Model is the foundational architecture my project builds upon. It consists of a VAE perception module and an MDN-RNN prediction module, with a controller trained via an evolution strategy. This serves as my primary baseline. A direct comparison against this model quantifies the overall performance improvement gained from modernizing both the perceptor (to VQ-VAE) and the predictor (to Transformer).
- **MDN-Transformer (Ablation Study)** This baseline is a hybrid model that combines the original VAE from the 2018 World Model with my proposed Transformer-based predictor. This model serves as a crucial ablation study. By changing only the predictor while keeping the perceptor fixed, this comparison is designed to specifically isolate and measure the performance contribution of the Transformer architecture, independent of the improvements from the VQ-VAE.
- **VQ-VAE (Ablation Study)** Similarly, this baseline combines VQ-VAE with MDN-RNN from 2018 World Model. This comparison is designed to specifically isolate and measure the performance contribution of VQ-VAE.
- **DreamerV2(Zhao et al., 2024)** DreamerV2 is a state-of-the-art model-based reinforcement learning agent. It features a sophisticated Recurrent State-Space Model (RSSM) for dynamics prediction and employs a learned actor-critic policy for control. It is included not for a direct architectural comparison, but to provide an upper-bound benchmark. This situates my model's performance within the broader landscape of contemporary and more complex methods in model-based control. Evaluatiing DreamerV2 may be a challenge, but I will give it a try when time and hardware resources are available.

Ethical Considerations

All data utilized in this project is sourced from publicly available, open-source resources. Specifically, the CarRacing dataset was generated using the open-source scripts provided by Ha and Schmidhuber (2018), while the Breakout dataset is the publicly available expert-v0 dataset from the Minari library (Younis et al., 2024). The scope of this research is confined to simulated game environments and does not involve interaction with human participants. Consequently, the primary ethical considerations stem not from data privacy or human subjects, but from the potential implications of the world model architecture itself and its future applications.

A key capability of world models, as demonstrated by Ha and Schmidhuber (2018), is the ability to train agents entirely within their simulated latent space. This introduces a significant ethical risk when overfitting to the simulation. An agent might learn to exploit flaws or inaccuracies in the simulated physics to maximize its reward—a behavior that would be ineffective or dangerous if transferred to a real-world environment.

This risk is amplified by the consequence-free nature of simulated training. Within the simulation, an agent can learn destructive or malicious policies (e.g., crashing a car intentionally to test an edge case) without any real-world cost. If such policies are not properly constrained, their transfer to real-world systems—such as autonomous vehicles or robotics—could pose a significant threat to safety and property.

References

Assran, M., Bardes, A., Fan, D., Garrido, Q., Howes, R., Komeili, M., Muckley, M., Rizvi, A., Roberts, C., Sinha, K., Zholus, A., Arnaud, S., Gejji, A., Martin, A., Robert Hogan, F., Dugas, D., Bojanowski, P., Khalidov, V., Labatut, P., Massa, F., Szafraniec, M., Krishnakumar, K., Li, Y., Ma, X., Chandar, S., Meier, F., LeCun, Y., Rabbat, M., and Ballas, N. (2025). V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*.

Assran, M., Duval, Q., Misra, I., Bojanowski, P., Vincent, P., Rabbat, M., LeCun, Y., and Ballas, N. (2023). Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv preprint arXiv:2301.08243*.

Bardes, A., Garrido, Q., Ponce, J., Rabbat, M., LeCun, Y., Assran, M., and Ballas, N. (2024). Revisiting feature prediction for learning visual representations from video. *arXiv:2404.08471*.

Bardes, A., Ponce, J., and LeCun, Y. (2023). Mc-jepa: A joint-embedding predictive architecture for self-supervised learning of motion and content features.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym.

Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., and Ramesh, A. (2024). Video generation models as world simulators.

Bruce, J., Dennis, M., Edwards, A., Parker-Holder, J., Shi, Y., Hughes, E., Lai, M., Mavalankar, A., Steigerwald, R., Apps, C., Aytar, Y., Bechtle, S., Behbahani, F., Chan, S., Heess, N., Gonzalez, L., Osindero, S., Ozair, S., Reed, S., Zhang, J., Zolna, K., Clune, J., de Freitas, N., Singh, S., and Rocktäschel, T. (2024). Genie: Generative interactive environments.

Ding, J., Zhang, Y., Shang, Y., Zhang, Y., Zong, Z., Feng, J., Yuan, Y., Su, H., Li, N., Sukiennik, N., Xu, F., and Li, Y. (2025). Understanding world or predicting future? a comprehensive survey of world models.

Fei, Z., Fan, M., and Huang, J. (2024). A-jepa: Joint-embedding predictive architecture can listen.

Ha, D. and Schmidhuber, J. (2018). Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*, pages 2451–2463. Curran Associates, Inc. `https://worldmodels.github.io`.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2020). Dream to control: Learning behaviors by latent imagination.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. (2019). Learning latent dynamics for planning from pixels. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2555–2565. PMLR.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. (2024). Mastering diverse domains through world models.

Huang, H., LeCun, Y., and Balestriero, R. (2025). Llm-jepa: Large language models meet joint embedding predictive architectures.

Kahneman, D. (2011). *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York.

Klimov, O. (2016). Carracing-v2. `https://gymnasium.farama.org/environments/box2d/car_racing/`.

Lucas, J., Tucker, G., Grosse, R. B., and Norouzi, M. (2019). Understanding posterior collapse in generative latent variable models. In *DGS@ICLR*.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M. J., and Bowling, M. (2018). Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562.

Saito, A., Kudeshia, P., and Poovvancheri, J. (2025). Point-jepa: Joint embedding predictive architecture for 3d point cloud self-supervised learning. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Skenderi, G., Li, H., Tang, J., and Cristani, M. (2025). Graph-level representation learning with joint-embedding predictive architectures.

van den Oord, A., Vinyals, O., and kavukcuoglu, k. (2017). Neural discrete representation learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Verdenius, S., Zerio, A., and Wang, R. L. (2024). Lat-pfn: A joint embedding predictive architecture for in-context time-series forecasting. *arXiv preprint arXiv:2405.10093*.

Younis, O. G., Perez-Vicente, R., Balis, J. U., Dudley, W., Davey, A., and Terry, J. K. (2024). Minari.

Zhao, G., Wang, X., Zhu, Z., Chen, X., Huang, G., Bao, X., and Wang, X. (2024). Drivedreamer-2: Llm-enhanced world models for diverse driving video generation.