Integrating Object-Centric Learning with Model-Based Reinforcement Learning

A Study on Enhanced Sample Efficiency and Generalization in Atari Pong
Master thesis by Your Full Name (Student ID: 1234567)
Date of submission: 23. September 2025

1. Review: Prof. Dr. Supervisor Name
2. Review: Dr. Second Reviewer
Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Department of Computer Science

Institute of Computer Science

Machine Learning Research Group

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 und § 23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Your Full Name, die vorliegende master thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Fall eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung gemäß § 23 Abs. 7 APB überein.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 23. September 2025

Y. Name

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

List of Abbreviations

RL	Reinforcement Learning
MBRL	Model-Based Reinforcement Learning
MFRL	Model-Free Reinforcement Learning
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
PPO	Proximal Policy Optimization
A3C	Asynchronous Advantage Actor-Critic
DQN	Deep Q-Network
RSSM	Recurrent State Space Model
KL	Kullback-Leibler
MSE	Mean Squared Error
RGB	Red-Green-Blue
GPU	Graphics Processing Unit
JAX	Just After eXecution

1. Introduction

1.1. Motivation and Problem Statement

Reinforcement Learning (RL) has achieved remarkable success in various domains, from game playing to robotics. However, traditional RL approaches face significant challenges in terms of sample efficiency and generalization capabilities. Model-free methods, while effective in specific environments, often require millions of environment interactions to learn optimal policies. Model-based approaches attempt to address this limitation by learning explicit world models, but they frequently struggle with the complexity of high-dimensional observations and the compounding errors in long-horizon planning.

Object-centric representations offer a promising avenue for addressing these challenges by decomposing complex scenes into meaningful object-level abstractions. By focusing on relevant entities and their relationships rather than raw pixel representations, agents can potentially achieve better sample efficiency and improved generalization to unseen scenarios.

1.2. Research Questions and Objectives

This thesis addresses the following research questions:

1. How can object-centric representations be effectively integrated into model-based reinforcement learning frameworks?
2. What are the benefits of this integration in terms of sample efficiency and generalization capabilities?
3. How does the proposed approach compare to existing model-free and pixel-based model-based methods?

The primary objectives of this work are:

- Design and implement an integrated object-centric model-based RL system
- Evaluate the approach on the Atari Pong environment
- Analyze the impact of object-centric representations on learning efficiency
- Assess the interpretability and generalization capabilities of the learned models

1.3. Contributions

The main contributions of this thesis are:

1. A novel integration of object-centric state representations with model-based reinforcement learning
2. Implementation of multiple world model architectures optimized for structured state spaces
3. Comprehensive experimental evaluation demonstrating improved sample efficiency
4. Analysis of the interpretability benefits of object-centric world models
5. Open-source implementation enabling reproducible research

1.4. Thesis Structure

This thesis is organized as follows: Chapter 2 provides the necessary background on reinforcement learning and object-centric representations. Chapter 3 presents the theoretical framework and integration strategy. Chapter 4 details the technical implementation. Chapter 5 presents experimental results and analysis. Chapter 6 discusses the implications and limitations of the approach. Chapter 7 concludes the thesis and outlines future research directions.

2. Background and Related Work

2.1. Reinforcement Learning Fundamentals

2.1.1. Model-Free vs. Model-Based Approaches

There are two main categories in Reinforcement learning. First of all Model-Free Reinforcement Learning focousses on training an agent without explicitly learning a model of the environment. There is no prediction of future states but purely an agent that takes in observation and directly computes Decisions. Whereas Model Based Reinforcement Learning usually consists of a world model part that tries to make predicitons on how the environment will behave in the future. This can be done using actual predicition of next observations directly or using some kind of latent space and making predictions there.

2.1.2. Sample Efficiency Challenges

Sample Efficiency so the optimisation of the amount of rollouts that is sampled in the actual environment can have multiple motivations. On the one hand the actual environment can be something in the physical world where there are contraints on the amount of times a physical object like a robot can move or how fast one can generate real samples. On the other hand computing states in the actual environment might me more costly and can therefore lower training speed depending on the complexity of the environment. For example running a game can be more resource intensive than computing a next (latent) state in the internal representation. Model based approaches usually vastly improve sample effiency because the data hungry step of training the agent can be done in the worldmodel. Meaning the worldmodel creates artificial rollouts (or parts of them) in which the actor can then do its training. The worldmodel has to of course sample training data in the actual environment but it usually requires way less training steps.

2.1.3. Generalization Problems

Moreover Model-Free approaches sometimes suffer from generalization problems. They overfit on a certain state of the environment. A common thing is the background color of the game if the actor is training on visual input. Thus changing the background which would have zero impact on the performance of a human player can drastically decrease the performance of an agent. Model Based Approaches usually come with some form of abstraction be it latent space or object centric things which usually do not suffer that much from simple changes in the environment. Also due to the error of the Environment Prediction an actor might see more diverse things to begin with.

2.2. Model-Based Reinforcement Learning

2.2.1. World Model Learning

There are multiple way to to train a world model. As mentioned above already you can predict latent states or directly predict future observations. There are also multiple different architectures which can do the job. From a simple MLP over LSTMs to Transformers there have been many different approiaches to that same problem

2.2.2. Planning with Learned Models

2.2.3. DreamerV2 Architecture

Explain in general what dreamer does what are the components and how do they play together

2.2.4. DreamerV2 TD Lambda Targets

The λ-target is defined recursively as:

$$V_t^\lambda = \hat{r}_t + \hat{\gamma}_t \cdot \begin{cases} (1-\lambda) v_\xi(\hat{z}_{t+1}) + \lambda V_{t+1}^\lambda & \text{if } t < H \\ v_\xi(\hat{z}_H) & \text{if } t = H \end{cases} \tag{2.1}$$

where $\hat{r}_t$ represents the predicted reward, $\hat{\gamma}_t$ is the predicted discount factor, $v_\xi(\hat{z}_t)$ is the critic's value estimate, and $\lambda = 0.95$ controls the weighting between immediate and future rewards. This formulation creates a weighted average of n-step returns, where longer horizons receive exponentially decreasing weights.

In my case I dont have z but simply the observation itself. So no latent space.

2.2.5. DreamerV2 Actor Loss

The actor network in DreamerV2 employs a sophisticated loss function that combines multiple gradient estimators to achieve both learning efficiency and convergence stability. The actor aims to maximize the same λ-return targets used for critic training, incorporating intermediate rewards directly rather than relying solely on terminal value estimates.

The combined actor loss function is formulated as:

$$\mathcal{L}(\psi) = \mathbb{E}\left[\sum_{t=1}^{H-1}\Big[-\rho \ln p_\psi(\hat{a}_t|\hat{z}_t)\,\text{sg}(V_t^\lambda - v_\xi(\hat{z}_t)) \quad \text{[REINFORCE]} \right. \tag{2.2}$$

$$- (1-\rho)V_t^\lambda \quad \text{[Dynamics Backprop]} \tag{2.3}$$

$$\left. - \eta\mathcal{H}[a_t|\hat{z}_t] \quad \text{[Entropy Regularization]}\Big]\right. \tag{2.4}$$

The loss function incorporates three distinct components:

REINFORCE Gradients: REINFORCE algorithm, which maximizes the log-probability of actions weighted by their advantage values. The advantage is computed to be the difference between the lambda-return and the critic's estimate, with gradients stopped around the targets (denoted by sg stop gradient) to prevent interference with critic learning

Entropy Regularization: The third term encourages exploration by maximizing the entropy of the action distribution. The entropy coefficient η controls the trade-off between exploitation and exploration, with higher values promoting more diverse action selection.

The weighting parameter ρ determines the relative contribution of REINFORCE versus straight-through gradients. For discrete action spaces like Atari, DreamerV2 typically uses $\rho = 1$ (pure REINFORCE) with $\eta = 10^{-3}$, while continuous control tasks benefit from $\rho = 0$ (pure dynamics backpropagation) with $\eta = 10^{-4}$.

2.2.6. DreamerV2 Critic Loss

The critic network in DreamerV2 serves as a value function approximator that estimates the expected discounted sum of future rewards from any given latent state. This component is essential for both the λ-target computation and providing baseline estimates for the REINFORCE algorithm, making it a cornerstone of the learning process.

The critic is trained using temporal difference learning with the λ-targets as regression targets. The loss function is formulated as a squared error between the critic's predictions and the computed λ-returns:

$$\mathcal{L}(\xi) = \mathbb{E}\left[\sum_{t=1}^{H-1} \frac{1}{2}\left(v_\xi(\hat{z}_t) - \text{sg}(V_t^\lambda)\right)^2\right] \tag{2.5}$$

where $v_\xi(\hat{z}_t)$ represents the critic's value estimate for latent state $\hat{z}_t$, and $\text{sg}(V_t^\lambda)$ denotes the λ-target with stopped gradients. The gradient stopping prevents the critic's learning from interfering with the target computation, maintaining the stability of the temporal difference updates.

Several key design choices enhance the critic's learning efficiency:

Target Network Stabilization: Following the approach used in Deep Q-Networks, DreamerV2 employs a target network that provides stable targets for critic learning. The target network is a delayed copy of the critic parameters, updated every 100 gradient steps. This approach prevents the rapid changes in the critic from destabilizing the learning targets.

Trajectory Weighting: The loss terms are weighted by cumulative predicted discount factors to account for episode termination probabilities. This weighting ensures that states likely to lead to episode endings receive appropriate emphasis during training.

Compact State Representation: Unlike traditional value functions that operate on high-dimensional observations, the DreamerV2 critic leverages the compact latent states $\hat{z}_t$ learned by the world model. This representation provides several advantages: reduced computational complexity, better generalization across similar states, and improved learning efficiency due to the structured nature of the latent space.

The critic architecture consists of a multi-layer perceptron with ELU activations and approximately 1 million trainable parameters. The network outputs a single scalar value representing the expected return from the input state, enabling efficient batch processing of imagined trajectories during training.

2.2.7. TD Lambda Targets

2.3. Object-Centric Representations

2.3.1. Limitations of Pixel-Based Representations

2.3.2. Object-Centric Learning Approaches

2.3.3. Relational Reasoning in RL

2.4. Current Limitations and Research Gaps

2.4.1. Lack of Integration Between Approaches

2.4.2. Misalignment Problems

2.4.3. Generalization Challenges

3. Methodology

3.1. Theoretical Framework

3.1.1. Object-Centric World Model Design

3.1.2. Integration Strategy with Model-Based RL

3.1.3. Expected Benefits Analysis

3.2. Implementation Approach

3.2.1. Environment Selection (Atari Pong)

3.2.2. Object-Centric State Representation

3.2.3. World Model Architecture Design

3.3. Experimental Design

3.3.1. Baseline Comparisons

3.3.2. Evaluation Metrics

3.3.3. Ablation Studies

4. Implementation

4.1. Environment and Setup

4.1.1. Pong Environment Characteristics

4.1.2. Object-Centric State Extraction

4.1.3. Frame Stacking and Preprocessing

4.2. World Model Architecture

4.2.1. LSTM-Based World Model (PongLSTM)

4.2.2. Alternative Architectures Explored

4.2.3. State Normalization and Stability

4.3. Actor-Critic Integration

4.3.1. DreamerV2-Style Actor-Critic

4.3.2. Policy Learning in Imagined Rollouts

4.3.3. Lambda-Return Computation

4.4. Training Pipeline

4.4.1. Experience Collection

4.4.2. World Model Training

4.4.3. Policy Optimization

5. Experiments and Results

5.1. Experimental Setup

5.1.1. Hyperparameters and Configuration

5.1.2. Hardware and Implementation Details

5.1.3. Evaluation Protocol

5.2. World Model Performance

5.2.1. Prediction Accuracy Analysis

5.2.2. Long-Term Rollout Quality

5.2.3. Model Stability Assessment

5.3. Policy Learning Results

5.3.1. Sample Efficiency Comparison

5.3.2. Final Performance Evaluation

5.3.3. Learning Curve Analysis

5.4. Ablation Studies

5.4.1. Impact of Object-Centric Representations

5.4.2. Architecture Component Analysis

5.4.3. Reward Function Design Effects

5.5. Visualization and Interpretability

5.5.1. Real vs. Model Comparison

5.5.2. Learned Representations Analysis

5.5.3. Policy Behavior Visualization

6. Discussion

6.1. Analysis of Results

6.1.1. Strengths of the Proposed Approach

6.1.2. Limitations and Challenges

6.1.3. Comparison with Existing Methods

6.2. Technical Insights

6.2.1. World Model Design Choices

6.2.2. Training Stability Issues

6.2.3. Reward Engineering Importance

6.3. Implications for Object-Centric RL

6.3.1. Benefits of Integration

6.3.2. Generalization Potential

6.3.3. Scalability Considerations

6.4. Future Research Directions

6.4.1. More Complex Environments

6.4.2. Improved Object Discovery

6.4.3. Multi-Object Scenarios

7. Conclusion

7.1. Summary of Contributions

7.2. Key Findings

7.3. Limitations and Future Work

7.4. Final Remarks

A. Implementation Details

A.1. Code Structure and Organization

A.2. Hyperparameter Sensitivity Analysis

A.3. Additional Experimental Results

B. Technical Specifications

B.1. Hardware Requirements

B.2. Software Dependencies

B.3. Reproducibility Guidelines

C. Supplementary Figures and Tables