VLASIM: WORLD MODELLING VIA VLM-DIRECTED ABSTRACTION AND SIMULATION FROM A SINGLE IMAGE

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative video models, a leading approach to world modeling, face fundamental limitations. They often violate physical and logical rules, lack interactivity, and operate as opaque black boxes ill-suited for building structured, queryable worlds. To overcome these challenges, we propose a new paradigm focused on distilling a single image into a tractable, abstract representation optimized for simulation. We introduce VLASim, a framework where a Vision-Language Model (VLM) acts as an intelligent agent to orchestrate this process. The VLM autonomously constructs a grounded (2D or 3D) scene representation by selecting from a suite of vision tools, and accordingly chooses a compatible physics simulator (e.g., rigid body, fluid) to act upon it. VLASim can then infer latent dynamics from the static scene to predict plausible future states. Our experiments show that this combination of intelligent abstraction and adaptive simulation results in a versatile world model capable of producing high quality simulations across a wider range of dynamic scenarios.

1 INTRODUCTION

Understanding and forecasting how the visual world evolves is a core challenge for building intelligent systems. Humans can observe a static scene and infer not only its current structure but also how it might change over time and in response to different actions. This capability underlies essential skills such as planning, decision-making, and causal reasoning. Replicating this physical intuition in AI hinges on creating robust "world models" to predict potential futures. In recent years, the dominant paradigm for building such models has been large-scale video generation. By training on immense visual corpora, these models have achieved remarkable success in synthesizing complex, dynamic scenes with impressive visual realism, suggesting they are learning a powerful, albeit implicit, model of our world only from 2D observations and text descriptions Wan et al. (2025); Google Deepmind (2025b). However, as most video models are only conditioned on image and text inputs, they lack a native mechanism for physical interaction, which limits their utility for tasks that require physical intuition. While some models incorporate action conditioning (Song et al., 2025; Google Deepmind, 2025a), the action space is typically limited to simple transformations, such as changes in camera viewpoint, rather than complex physical interventions.

Despite their visual prowess, pixel-space models result in critical and systematic failures that limit their use for robust interaction (Motamed et al., 2025; Kang et al., 2024). First, they frequently generate physically implausible scenarios. Video models learn statistical correlations from pixels and do not enforce any physical plausibility constraints, which results in their outputs often violating fundamental principles of object permanence, collision, and causality. For example, the number and size of objects can change erratically, or objects can accelerate without a corresponding force. This failure to deduce underlying principles is not limited to 3D physics; these models are equally unable to infer the structured representations and simple, deterministic rules required to simulate abstract 2D environments like Conway's Game of Life Conway et al. (1970). Second, these models operate as opaque 'black boxes'. The generated scene is not a structured, queryable world but a sequence of pixels. Consequently, it is impossible to inspect the environment's underlying state or apply novel physical actions beyond those observed during training. Ultimately, the world inside these models is a passive movie to be watched, not a dynamic environment to be acted upon. To build

useful world models, we need approaches that yield structured, interactive, and physically grounded representations.

Separate from generative video models, another significant line of research has focused on reconstructing 3D or 4D scene representations from images. Foundational models like DUST3R (Wang et al., 2024) have demonstrated impressive capabilities in producing dense and accurate geometric reconstructions, while methods based on Neural Radiance Fields (NeRFs) Mildenhall et al. (2021) excel at generating photorealistic novel views. However, the primary objective of both lines of work is to capture a scene's geometry and appearance, not its underlying physical nature. This focus on 3D geometry also renders them, by design, inapplicable to abstract 2D environments governed by logical rules, such as cellular automata like Conway's Game of Life and grid-based games like Snake. The resulting representations, e.g., point clouds or radiance fields, are not amenable to tractable simulation. Most methods do not decompose scenes into simplified primitives or, crucially, infer the physical properties (e.g., mass, friction, elasticity) necessary for a physics engine. Consequently, while these methods can show what a scene looks like from a new angle, they cannot predict what will happen next in response to physical forces, leaving a critical gap for simulation-ready world models. While some follow-up work has attempted to retrofit these representations for 3D physics (Zhang et al., 2024b; Petitjean et al., 2023), such efforts are typically restricted to narrow classes of simulation and fall short of a truly versatile and generalizable system.

In this paper, we introduce `VLASim`, a novel framework for world modeling that moves away from direct pixel prediction and instead builds an explicit, structured world representation. Instead of predicting pixels, our primary goal is to distill a visually complex image into a tractable abstract representation, illustrated in figure 1. This representation intentionally discards physically irrelevant information (like fine-grained textures or static backgrounds) to create a structured world model optimized for simulation. Our framework achieves this through a Vision-Language Model (VLM) that acts as a central agent, orchestrating three core innovations. First, the VLM acts as an intelligent tool-using agent to construct a grounded representation. It is equipped with a versatile suite of vision modules, including segmentation, 3D reconstruction, and primitive fitting, and autonomously decides which tools to deploy. This allows the representation to be grounded in the scene's native dimensionality; for instance, applying the full 3D pipeline for spatial environments while recognizing that such tools are irrelevant for planar ones, such as Conway's Game of Life. Second, the choice of representation and simulator is co-dependent and adaptive. The VLM jointly determines the type of abstraction and the most appropriate physics simulator to act upon it: a scene with blocks is abstracted into a rigid body model and paired with a rigid body solver, while one with water is represented as a particle system paired with a fluid dynamics engine. Finally, this structured world model enables `VLASim` to infer latent dynamics, predicting a scene's likely evolution from the static image alone based on physical cues and the text description of the scene. Through comprehensive experiments, we show this combination of intelligent abstraction, adaptive simulation, and inferred dynamics results in a world model that significantly outperforms prior methods in producing high-quality, physically and logically plausible simulations across a wide range of scenarios.

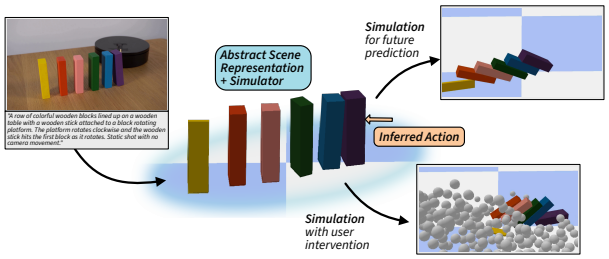

Figure 1: Overview of `VLASim`.

2 Previous Work

Video Models Recent advances in generative methods have established large-scale video models as a dominant paradigm for modeling world dynamics. State-of-the-art models have demonstrated a remarkable ability to synthesize high-fidelity and temporally coherent videos from text and image inputs (OpenAI, 2024; Google Deepmind, 2025b; Runway, 2025; Wan et al., 2025). These models use diffusion or flow-matching approaches in a compressed latent space for video generation. While most models only condition on image and text inputs, some recent methods have enabled

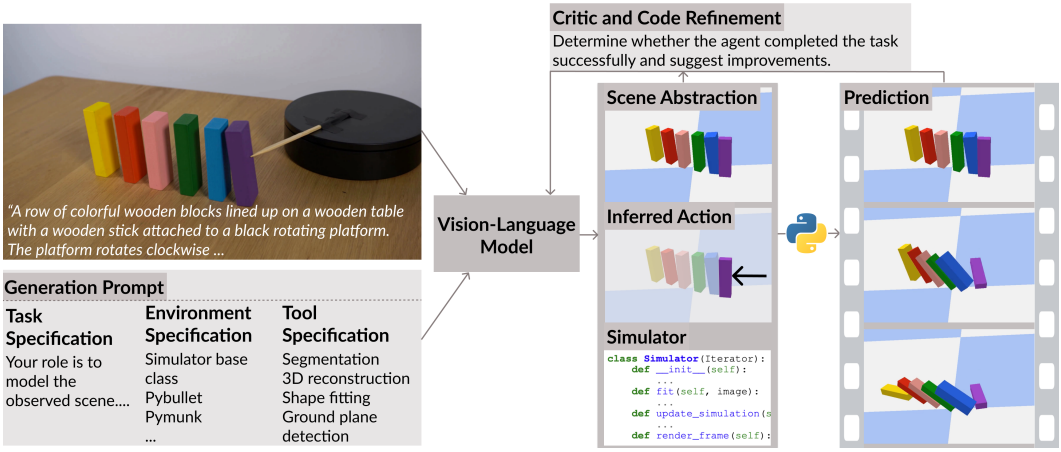

Figure 2: Illustration of our method. `VLASim` takes a single image and text description as input. We design a generation prompt that the VLM uses to generate a scene abstraction, along with a simulator of the scene and any actions that can be inferred from the input. The simulator code is then executed to generate the future predictions. We can also generate other diverse videos by interactively changing the actions. Finally, a code and refinement step is capable of automatically correcting any errors in the predictions and generate fixes to the intial prediction.

conditioned on other parameters, such as camera parameters, for more controllable video generation (Huang et al., 2025; Song et al., 2025; Google Deepmind, 2025a). However, it is very difficult to interact with these models outside of the control parameters used during training. It is generally impossible to query the state of an object, apply a novel physical force, or explore alternative outcomes under different conditions. These models do not explicitly reason in a structured space, and instead directly perform computations in frame space. This leads their outputs to frequently violate fundamental principles of the real world (Motamed et al., 2025; Li et al., 2025; Kang et al., 2024). Common failure cases include the violation of object permanence, where objects may inexplicably appear or vanish, and inconsistent causality, where actions do not have plausible consequences. The lack of explicit, structured reasoning and interactivity in pixel-space video models limits their utility as robust world models, motivating our shift towards an explicit, simulation-based approach.

Scene Reconstruction and Simulation 3D reconstruction from images has achieved considerable success in recent years. Models like DUST3R (Wang et al., 2024) and its follow-ups (Zhang et al., 2024a; Wang et al., 2025; Feng* et al., 2025) produce dense geometric reconstructions from images, while methods based on Neural Radiance Fields (NeRFs) (Yu et al., 2021; Tewari et al., 2023) and 3D Gaussians (3DGS) (Szymanowicz et al., 2024; Charatan et al., 2024) and their 4D extensions (Wu et al., 2024; Tretschk et al., 2021; Yang et al., 2023; Yunus et al., 2024) excel at generating photorealistic novel views. While most approaches do not inherently decompose the scene into discrete, object-centric components, some methods have made progress with the help of features from vision-language models Kerr et al. (2023); Jatavallabhula et al. (2023). Additionally, most neural radiance methods are optimized for appearance rather than physics, making them poorly suited for interactive simulation. Some work has attempted to model physics, for instance, by combining 3D representations with physics engines (Zhang et al., 2024b; Petitjean et al., 2023; Feng et al., 2024; Li et al., 2023; Wu et al., 2015; Le et al., 2025; Chen et al., 2025; Kairanda et al., 2025; Xie et al., 2024), but they all only model limited physical phenomena. Concurrent work, PhysGen3D Chen et al. (2025), reconstructs object-centric 3D scenes and performs simulation with a fixed material point method. In addition, it only takes an image as input and cannot reason about input text. In this work, we use tools from scene reconstruction and simulation methods but do not used a fixed pipeline as the VLM is free to select scene representations and simulators best suitable for any input.

Program Synthesis with VLMs Our work is informed by recent advances in using Vision-Language Models (VLMs) as agents that synthesize programs to solve complex visual tasks. A foundational paradigm is visual program synthesis for querying. Methods like ViperGPT (Surís et al., 2023) and VisProg (Gupta & Kembhavi, 2023) can parse a complex visual query into a se-

quence of steps, generating code that calls various vision APIs (e.g., object detectors, depth estimators) to arrive at a final answer. LayoutGPT Feng et al. (2023), which uses an LLM to generate a complete scene layout, including the sizes, positions, and relationships of different objects Other research has shown that a VLM can evolve interpretable visual classifiers (Chiquier et al., 2024) and design interpretable programs to describe underlying scientific laws Mall et al. (2025). A second major application is in high-level planning and robotics. This research aims to create agents that can reason about the world to perform actions. VisualPredicator (Liang et al., 2024), for example, learns neuro-symbolic predicates that classify the state of the world for a symbolic planner. Others, like VoxPoser Huang et al. (2023), use LLMs to synthesize 3D affordances that guide a low-level motion planner. The common thread in this research is that the VLM's role is to generate a plan or a set of actions for an agent to execute within an existing environment.

3 VLASIM: WORLD MODELLING VIA VLM-DIRECTED ABSTRACTION AND SIMULATION FROM A SINGLE IMAGE

The input to `VLASim` is a pair consisting of a single image and a text prompt that describes the scene. The goal of `VLASim` is to convert this static input into a dynamic, interactive world model. This process is orchestrated by a central Vision-Language Model (VLM), which generates a complete "world program" in Python ready for execution. This program consists of three key components: (1) *A Grounded Abstract Representation*: The VLM selects from a suite of vision tools to construct a 2D or 3D model of the scene, optimized for simulation, (2) *Inferred Latent Dynamics*: It predicts the most likely implicit action from the visual and textual cues, which serves as the initial condition for the simulation, (3) *A Selected Simulator*: It determines the most compatible simulation engine (e.g., rigid body, fluid, rule-based) to simulate the scene's dynamics. Once generated, this world program is executed to predict a plausible future. Because the program describes an explicit and structured world, it can also be modified with novel user-defined actions to imagine diverse futures.

3.1 PROMPTING FOR WORLD PROGRAM GENERATION

The core of `VLASim` lies in guiding a powerful Vision-Language Model (VLM) to generate a complete, executable world program. Instead of fine-tuning, we steer the model's behavior at inference time using a comprehensive, multi-part prompt. The prompt begins with a high-level task specification in natural language. This instruction outlines the overall objective: to analyze a user-provided image and text description and produce a self-contained Python script that simulates the scene's future. This main directive then embeds two more structured components to formalize the task: environment specification that provides the structural code template, and tool specification, which provides the API definitions for the suite of perception tools the VLM can use.

Task Specification The task specification is a high-level, natural language instruction. It directs the VLM to analyze the user-provided inputs and produce a self-contained Python script that simulates the scene's future, making use of the other prompts to structure its output and call the necessary tools. A condensed version is shown in Figure 3.

Environment Specification The environment specification provides the formal scaffolding for the VLM's code generation task. Its central element is a Python Simulator base class that the VLM must use to derive the model from. This base class defines the core methods the VLM must implement, enforcing the entire simulation logic from scene setup to frame-by-frame execution. Additionally, the environment provides the VLM with a list of existing python libraries that it can rely on, pointing it to common simulation implementations. This ensures the VLM's output is structurally compatible with our execution environment, as illustrated in Figure 5.

Tool Specification Finally, the tool specification provides the VLM with the API of a diverse toolkit used for scene understanding and simulation setup. These helper functions are not required to be used by the VLM, but often help in the final solution. The API is organized into several categories: (1) Core Perception tools for open-vocabulary segmentation and 3D point cloud estimation; (2) Geometric Processing functions for fitting planes, cleaning data, and abstracting objects into primitive shapes; and (3) Simulation Interface methods that directly add objects (e.g., rigid meshes,

```
# ROLE: Computer Scientist

# TASK: Generate an executable Python class `VideoSimulation`
# that simulates the future of the scene from the input image
# and caption: "[CAPTION]".

# KEY PRINCIPLES:
# 1. Minimal Abstraction: Determine if the scene is fundamentally
# 2D or 3D and use the simplest required representation
# 2. Activating Agents: Model the *effect* of an external agent
# (e.g., a hand pushing a block), not the agent itself.
# 3. Robustness: Prioritize robustness to sensor noise.
```

Figure 3: A condensed version of the task specification provided to the VLM, outlining its role and key principles.

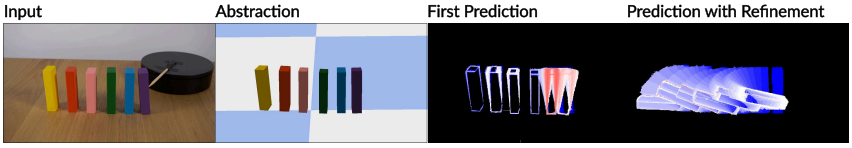

Figure 4: The VLM also acts as a critic. If the result is not perceived to be correct, a new result is generated. The spatiotemporal visualisations in the last two columns visualise the motion in the video as a static frame. In this example, the second block from the left has incorrect motion in the first prediction, that is then resolved with a better dynamics prediction in the last column.

```python
class Simulator(Iterator):
    def __init__(self, frame_size=(1024, 576), api: API=None, fps=30):
        """Initializes the simulator."""
    def fit(self, image: np.ndarray, text: str):
        """Fit the simulator's parameters to the provided image."""
    def update_simulation(self, dt: float):
        """Update the simulation by one timestep dt."""
    def render_frame(self):
        """Render the next frame of the simulation."""
# Available libraries: numpy, scipy, pybullet, pygame, punk..
```

Figure 5: The environment specification implements the base class, which defines the required structure for the VLM's generated Python code, and also provides a list of relevant Python libraries.

soft bodies, or particles) into physics engines. This rich set of tools allows the VLM to translate its conceptual understanding of a scene into the precise, low-level code required to instantiate and run a simulation, as exemplified in Figure 6.

```python
class API:
    def segment(self, image: np.ndarray, objects: List[str]):
        """Segments the image."""
    def fit_3d_shape(self, point_cloud: np.ndarray, shape_class: str):
        """Fits a 3D primitive ('cuboid', 'sphere', etc.) to a
        point cloud and returns its parameters."""
    def generate_surface_mesh(self, vertices, indices, mass=0.0):
        """Creates a mesh from a vertex mesh."""
```

Figure 6: The API provided in the tool specification defines a rich set of functions for perception and geometric processing.

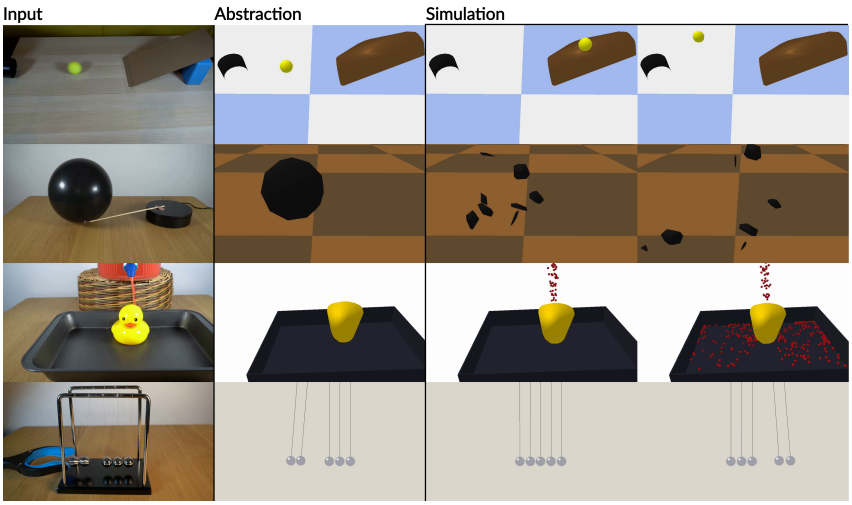

Figure 7: `VLASim` generates an abstraction of the scene and a simulator that can together be used to predict future scene states. From top to bottom: (Rigid Body) ball moving on an incline, (Thermodynamics) balloon bursting on interaction with flame, (Fluid) juice falling on a rubber duck, (2D Simulation) a cradle, (Logic) Conway's Game of Life.

3.2 Perception Toolbox

To power the API exposed in the tool specification, we implement a robust suite of perception and geometry modules that the VLM can call to perform a wide range of tasks. These functions are implemented to aid the inference of VLM. The VLM does not not re-implement or update the implementation of these updates, and it is free to ignore these implementations if it does not find any use for them.

3.2.1 2D Perception and Gometry

Open-Vocabulary segmentation The `segment` API function is built upon a state-of-the-art open-vocabulary segmentation pipeline. We first use Gemini Perception (Comanici et al., 2025) to estimate the bounding boxes for the query, and then Segment Anything (Kirillov et al., 2023) to compute dense segmentation maps from those boxes. The VLM infers the important and relevant objects in the scene to query this function. This function is used in almost all scenes to develop the right abstract scene representation.

Geometry Helpers The `fit_2D_shape` function fits simple 2D gemetric primitives, such as disks and polygons, to selcted regions. This helps in developing the right abstractions that can be used for tractable simulation.

3.2.2 3D Perception and Geometry

Single Image 3D Estimation We use VGGT (Wang et al., 2025) to estimate dense 3D point maps from a single image as the `pts3d` function. This model is also used to implement the `intrinsics` call that computes the camera paramters.

Geometry Helpers These tools rely on established computer graphics algorithms. The `predict_ground_plane` function uses a RANSAC-based approach to robustly fit a plane to a point cloud, allowing the system to establish a world coordinate frame. The `fit_3d_shape` function performs robust, RANSAC-based, fitting of simple geometric primitives to point clouds. This is an important component, as it not only abstracts the shape for tractable simulation, but also computes a complete shape from incomplete point clouds. The `generate_surface_mesh` and `add_soft_body` create meshes and soft bodies from point clouds.

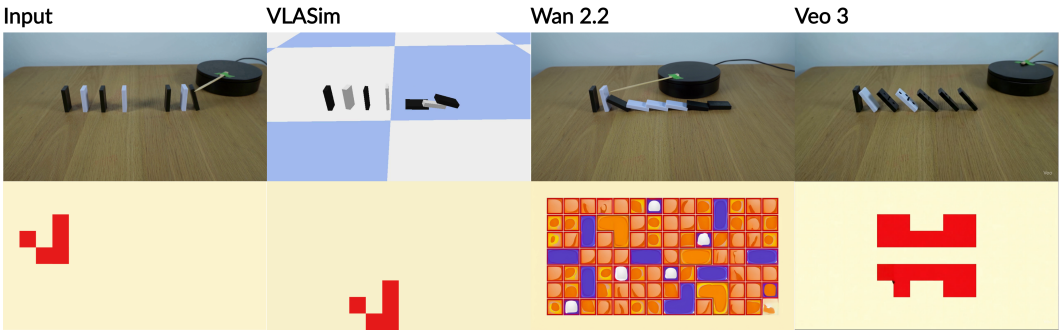

Figure 8: Comparisons of our approach with video generation models Wan 2.2 and Veo 3. Compared to both of these models, our approach is able to follow physical principles such as object permanence and the effect of gravity (Top); while on the Game of Life dataset our approach is able to correctly infer future patterns based on the true rules of the game (Bottom).

3.3 Critic and Code Refinement

While a single generation pass can produce high-quality results, high complexity in scenes can lead to errors in the initial code or inaccurate scene fits. To enhance the robustness of our system, we introduce an automated feedback loop: a two-stage Critic and Refinement process, as illustrated in Figure 2. This allows the system to identify and correct its own mistakes.

Critic Stage. In the first stage, a VLM is prompted to act as a critic. Along with the text caption, the critic is provided with the initial frame of the generated simulation and, crucially, a spatiotemporal colormap that visualizes all dynamic activity over the simulation's duration (blue for early motion, red for late), see Figure 4 for a visualization. The critic's task is to assess the correctness of the simulation's initial conditions and physical setup, not its visual quality. It then outputs a structured JSON object containing a boolean flag evaluating the accuracy of the simulation, and a list of suggested improvements.

Refinement Stage. If the critic deems the simulation inaccurate, the second stage begins. A VLM is prompted to act as a 'code refiner'. It receives the original, flawed Python code generated in the first pass, along with the specific suggested improvements from the critic's JSON feedback. Its task is to debug and rewrite the code to address the identified issues, producing a final, corrected `VideoSimulation` class. This self-correction capability improves the quality and physical plausibility of the final output.

Automated Debugging. Separate from the semantic feedback loop, we also implement a process for handling runtime errors that produce no simulation outputs. If the generated Python code fails to execute due to an error (e.g., from incorrect API usage or unexpected perception tool outputs), we automatically capture the full error traceback. A VLM is then prompted to act as a debugger. It is provided with the original flawed code, the full environment and API specifications, and the captured traceback. Its sole task is to correct the code based on the error message, allowing the system to recover from common programming mistakes and increasing the overall success rate of the generation process.

4 Experiments

We conduct a comprehensive set of experiments to validate our approach. Our evaluation is structured around four primary goals. First, we assess the physical plausibility of our generated simulations and benchmark them against state-of-the-art video models. Second, to demonstrate the versatility of our approach, we showcase results across a wide variety of physical phenomena, such as rigid-body dynamics and fluid interactions. Third, we validate that our method's programmatic output creates an interpretable scene abstraction that users can directly interact with and modify.

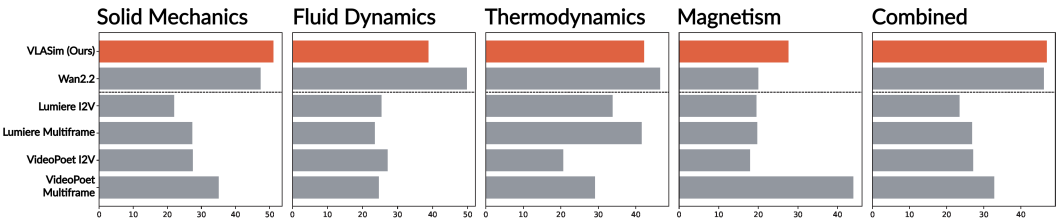

Figure 9: The modified Physics-IQ score of our approach compared with a selection of video generation models. We exclude evaluations on visual-based phenomena, and as such we do not consider the *Optical* category of Physics-IQ, as well as the *MSE* metric for the evaluation.

Finally, we conduct ablation studies in Sec 6.2 to analyze the contribution of each core component of our system.

Implementation Details. Our implementation uses Gemini Comanici et al. (2025) as the core Vision-Language Model agent. All GPU computations are performed on NVIDIA H200 GPUs. The inference time for generating a complete world program from a single input image and prompt is approximately 10 minutes. All code will be made publicly available.

Baselines. We benchmark `VLASim` against several state-of-the-art video generation models. Our primary baseline is Wan2.2 (Wan et al., 2025), as it represents the leading open-source model. We also compare with Lumiere (Bar-Tal et al., 2024) and VideoPoet (Kondratyuk et al., 2023). Finally, we include a select number of examples from Veo3 (Google Deepmind, 2025b)[1].

Benchmark and Metrics. For quantitative evaluation, we use the PhysicsIQ benchmark (Motamed et al., 2025). This dataset is composed of real-world videos capturing a diverse set of physical phenomena, categorized into the following areas: solid mechanics, fluid dynamics, magnetism, thermodynamics, and optics. As our method focuses on physical dynamics rather than visual appearance, we exclude the optics category from our evaluation.

We adopt three of the four metrics proposed by PhysicsIQ to evaluate motion and action. We exclude the Mean Squared Error (MSE) metric, as our goal is to model physically plausible motion via abstract simulation rather than achieve photorealistic pixel-level consistency. The metrics we use are: Spatial IoU (evaluating where the action happened), Weighted Spatial IoU (evaluating where and how much action happened), and Spatiotemporal IoU (evaluating when and where the action happened). Following the original benchmark protocol, we combine these three components to produce a final score out of 100, where a higher score indicates a more physically accurate prediction.

To specifically evaluate the logical reasoning capabilities of our method on a deterministic, rule-based system, we introduce a benchmark based on Conway's Game of Life (Conway et al., 1970). We created a test set of 10 distinct initial scenes. For each scene, the task is to generate a simulation program from a single input frame that correctly predicts the evolution of the board over subsequent steps. The accompanying text caption explicitly identifies the scene as Conway's Game of Life, tasking the model to apply the game's known rules. We evaluate the accuracy of the predicted frames using the F1 score, which is computed by comparing the state of each cell (live or dead) in the predicted grid against the ground truth, treating live cells as the positive class.

Qualitative Results. Figure 7 presents a selection of our qualitative results across various challenging scenarios. As shown in the top three rows, `VLASim` successfully generates physically plausible simulations for scenes involving complex solid body dynamics, as well as thermodynamics, as well as fluid interactions. The fourth row demonstrates the model's ability to select an appropriate level of abstraction. For this predominantly planar scene, our method correctly infers that a simpler 2D abstraction and simulation is sufficient, generating a program that is both computationally efficient and accurate for future prediction. The generated abstractions models the important components of the scene while intentionally discarding distracting, high-frequency visual details

[1]Due to the significant costs associated with Veo3, a full-scale evaluation was prohibitive for this paper.

and appearance. The goal is not to achieve photorealism, but to focus exclusively on producing a plausible and accurate simulation.

Figure 8 provides a direct qualitative comparison of VLASim against the state-of-the-art video models, Wan2.2 Wan et al. (2025) and Veo3 Google Deepmind (2025b). While the baseline models generate visually detailed outputs, they often exhibit common physical inconsistencies. For example, in the top row, both baselines change the number of visible blocks in the scene, and also do not correctly model the effect of the gap between the blocks. In contrast, VLASim generates simulations where the objects behave as distinct entities governed by consistent physical laws. The blocks collide plausibly. This highlights a fundamental advantage of our approach: the baselines attempt to learn physics implicitly within a high-dimensional pixel space, making them prone to such artifacts. Our method, by generating a program for an explicit, rule-based physics engine, inherently enforces object permanence and consistent dynamics.

Quantitative Results. Figure 9 plots the quantitative results on the PhysicsIQ benchmark. The scores show that VLASim performs on par with Wan2.2, the leading open-source video model. However, these quantitative metrics fail to capture the full picture of physical plausibility. As is evident in our qualitative comparisons (Figure 8) and supplementary video, the outputs from Wan2.2 frequently exhibit non-physical artifacts—such as objects unnaturally merging that demonstrate a lack of a true underlying physics model. This discrepancy suggests that the IoU-based metrics of PhysicsIQ, while useful for tracking general motion, are not sensitive enough to penalize these critical, common-sense violations. Our method, which is governed by an explicit physics engine, avoids such artifacts by design, a crucial advantage not fully reflected in the final score. We note that physical prediction is often non-deterministic. To account for this, for both our method and Wan2.2, we generate three distinct outputs and report the best score. Scores for other models are taken directly from the original PhysicsIQ paper, as their models were not available for our re-evaluation.

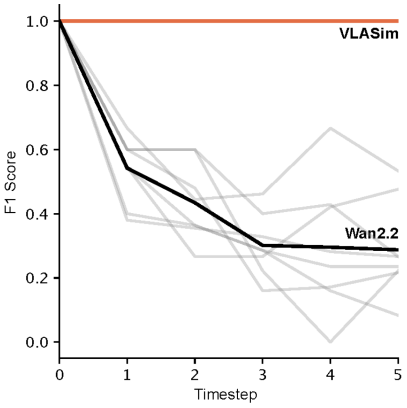

Figure 10: Results on Conway's Game of Life. One timestep corresponds to a single frame step, and an F1 score is calculated between the cells considered 'alive' in the predicted video and in the ground truth.

Finally, to evaluate performance on a purely logical and rule-based task, we present the results of our Conway's Game of Life benchmark in Figure 10. Here, VLASim significantly outperforms Wan2.2, achieving a perfect F1 score. This result highlights a fundamental difference between the two approaches. As a pixel-prediction model, Wan2.2 attempts to generate the visual patterns of the game's evolution but consistently fails to adhere to the strict, deterministic rules, leading to cumulative errors. This demonstrates the inherent advantages of an explicit, program-synthesis approach for tasks that require precise, rule-based reasoning.

5 Conclusion

In this work, we introduced VLASim, a new paradigm for building dynamic world models from static images. We have shown that by tasking a Vision-Language Model (VLM) with world program synthesis, it is possible to generate explicit, executable simulations that are physically plausible, interactive, and versatile. Our experiments demonstrate that this approach avoids the common physical artifacts of pixel-prediction models and excels at tasks requiring precise, rule-based reasoning. This programmatic approach represents a significant step towards creating more grounded and interactive world models. We believe our work points to a broader shift in how we build autonomous agents. Instead of relying on monolithic, end-to-end models that learn an opaque representation of the world, VLASim functions as a compositional agent that reasons about the world and writes code to model it.

REFERENCES

Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19457–19467, 2024.

Boyuan Chen, Hanxiao Jiang, Shaowei Liu, Saurabh Gupta, Yunzhu Li, Hao Zhao, and Shenlong Wang. Physgen3d: Crafting a miniature interactive world from a single image. *CVPR*, 2025.

Mia Chiquier, Utkarsh Mall, and Carl Vondrick. Evolving interpretable visual classifiers with large language models. In *European Conference on Computer Vision*, pp. 183–201. Springer, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

John Conway et al. The game of life. *Scientific American*, 223(4):4, 1970.

Haiwen Feng*, Junyi Zhang*, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J. Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36: 18225–18250, 2023.

Yutao Feng, Yintong Shang, Xuan Li, Tianjia Shao, Chenfanfu Jiang, and Yin Yang. Pie-nerf: Physics-based interactive elastodynamics with nerf. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4450–4461, 2024.

Google Deepmind. Genie 3: A new frontier for world models. `https://deepmind.google/discover/blog/genie-3-a-new-frontier-for-world-models/`, 2025a.

Google Deepmind. Veo: a text-to-video generation system. Technical report, 2025b. URL `https://storage.googleapis.com/deepmind-media/veo/Veo-3-Tech-Report.pdf`.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14953–14962, 2023.

Tianyu Huang, Wangguandong Zheng, Tengfei Wang, Yuhao Liu, Zhenwei Wang, Junta Wu, Jie Jiang, Hui Li, Rynson WH Lau, Wangmeng Zuo, and Chunchao Guo. Voyager: Long-range and world-consistent video diffusion for explorable 3d scene generation. *arXiv preprint arXiv:2506.04225*, 2025.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023.

Navami Kairanda, Marc Habermann, Shanthika Naik, Christian Theobalt, and Vladislav Golyanik. Thin-shell-sft: Fine-grained monocular non-rigid 3d surface tracking with neural deformation fields. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11373–11383, 2025.

Bingyi Kang, Yang Yue, Rui Lu, Zhijie Lin, Yang Zhao, Kaixin Wang, Gao Huang, and Jiashi Feng. How far is video generation from world model? – a physical law perspective. *arXiv preprint arXiv:2411.02385*, 2024.

Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 19729–19739, 2023.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4015–4026, 2023.

Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Grant Schindler, Rachel Hornung, Vighnesh Birodkar, Jimmy Yan, Ming-Chang Chiu, et al. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023.

Long Le, Ryan Lucas, Chen Wang, Chuhao Chen, Dinesh Jayaraman, Eric Eaton, and Lingjie Liu. Pixie: Fast and generalizable supervised learning of 3d physics from pixels. *arXiv preprint arXiv:2508.17437*, 2025.

Chenyu Li, Oscar Michel, Xichen Pan, Sainan Liu, Mike Roberts, and Saining Xie. Pisa experiments: Exploring physics post-training for video diffusion models by watching stuff drop. *arXiv preprint arXiv:2503.09595*, 2025.

Xuan Li, Yi-Ling Qiao, Peter Yichen Chen, Krishna Murthy Jatavallabhula, Ming Lin, Chenfanfu Jiang, and Chuang Gan. Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. *arXiv preprint arXiv:2303.05512*, 2023.

Yichao Liang, Nishanth Kumar, Hao Tang, Adrian Weller, Joshua B Tenenbaum, Tom Silver, João F Henriques, and Kevin Ellis. Visualpredicator: Learning abstract world models with neuro-symbolic predicates for robot planning. *arXiv preprint arXiv:2410.23156*, 2024.

Utkarsh Mall, Cheng Perng Phoo, Mia Chiquier, Bharath Hariharan, Kavita Bala, and Carl Vondrick. Disciple: Learning interpretable programs for scientific visual discovery. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 29258–29267, 2025.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? *arXiv preprint arXiv:2501.09038*, 2025.

OpenAI. Video generation models as world simulators. `https://openai.com/index/sora/`, 2024.

Automne Petitjean, Yohan Poirier-Ginter, Ayush Tewari, Guillaume Cordonnier, and George Drettakis. Modalnerf: Neural modal analysis and synthesis for free-viewpoint navigation in dynamically vibrating scenes. In *Computer Graphics Forum*, volume 42, pp. e14888. Wiley Online Library, 2023.

Runway. Gen-4. `https://runwayml.com/research/introducing-runway-gen-4`, 2025.

Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. *arXiv preprint arXiv:2502.06764*, 2025.

Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11888–11898, 2023.

Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10208–10217, 2024.

Ayush Tewari, Tianwei Yin, George Cazenavette, Semon Rezchikov, Josh Tenenbaum, Frédo Durand, Bill Freeman, and Vincent Sitzmann. Diffusion with forward models: Solving stochastic inverse problems without direct supervision. *Advances in Neural Information Processing Systems*, 36:12349–12362, 2023.

Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12959–12970, 2021.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20697–20709, 2024.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20310–20320, June 2024.

Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. *Advances in neural information processing systems*, 28, 2015.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4389–4398, 2024.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4578–4587, 2021.

Raza Yunus, Jan Eric Lenssen, Michael Niemeyer, Yiyi Liao, Christian Rupprecht, Christian Theobalt, Gerard Pons-Moll, Jia-Bin Huang, Vladislav Golyanik, and Eddy Ilg. Recent trends in 3d reconstruction of general non-rigid scenes. In *Computer Graphics Forum*, volume 43, pp. e15062. Wiley Online Library, 2024.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024a.

Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y. Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T. Freeman. PhysDreamer: Physics-based interaction with 3d objects via video generation. In *European Conference on Computer Vision*. Springer, 2024b.

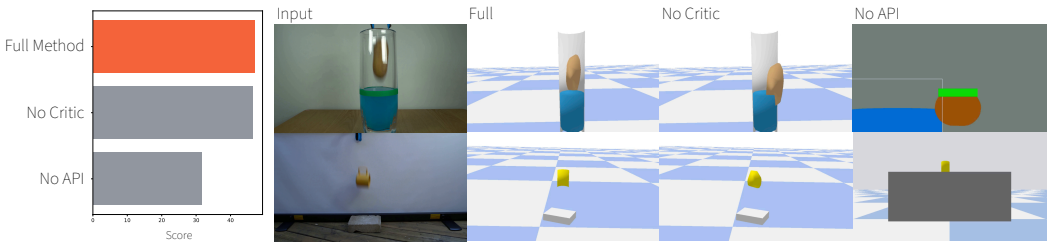

Figure 11: Ablations. We show quantitative results on left, and qualitative on the right. Both critic and the API improve the quality of our results.

6 Appendix

6.1 Quantitative Results

In table 1 we provide the numbers that were used in the plots in Figure 9.

Table 1: Results on Physics-IQ dataset. Subset excludes *Optical* category as wel as *MSE* component of metric. Our method marked with *.

Physics-IQ	Solid Mechanics	Fluid Dynamics	Thermodynamics	Magnetism	Total
VLASim*	**51.1**	38.8	42.2	27.6	**47.0**
Wan 2.2	47.3	**49.7**	**46.5**	20.0	46.2
Lumiere I2V	22.0	25.4	33.8	19.5	23.5
Lumiere MultiF	27.3	23.5	41.5	19.7	26.9
Runway	27.5	27.2	20.7	17.9	27.1
VideoPoet MultiF	35.1	24.6	29.1	**44.0**	32.8

6.2 Ablations

We conduct ablation studies to analyze the contribution of each core component of our system. The results, summarized in Figure 11, demonstrate the importance of each module. We use the same metric as in the main paper, and evaluate on one split of PhysicsIQ dataset. First, we evaluate a variant of our model without access to the perception toolbox API ('No API'). This version performs poorly, both quantitatively and qualitatively, failing to generate coherent or accurate simulations which match the input images. This result confirms that the VLM's ability to ground its reasoning in the explicit scene information provided by the perception tools is critical to its success.

Next, we analyze the impact of the critic-and-refinement loop ('No Critic'). Quantitatively, this variant performs similarly to our full method on the PhysicsIQ benchmark. However, we observe a noticeable improvement in the visual quality and physical plausibility of the final simulations when the critic is enabled. This suggests that while the initial program generated by the VLM is often functionally correct, the refinement loop is crucial for correcting subtle errors and improving the overall quality of the simulation.

6.3 Limitations

A key limitation we observe is the system's sensitivity to errors in the upstream perception toolbox. As a compositional system, the quality of the final simulation is often contingent on the accuracy of tools like segmentation and depth estimation. A failure in one of these modules—for example, misidentifying an object's shape or its 3D position—can lead the VLM to generate a semantically incorrect world program, even if that program is syntactically valid. The VLM currently has no mechanism to question or correct a faulty tool output.