SciTextures: Collecting and Connecting Visual Patterns, Models, and Code Across Science and Art

Sagi Eppel, Alona Strugatski

Abstract

The ability to connect visual patterns with the processes that form them represents one of the deepest forms of visual understanding. Textures of clouds and waves, the growth of cities and forests, or the formation of materials and landscapes are all examples of patterns emerging from underlying mechanisms. We present the Scitextures dataset, a large-scale collection of textures and visual patterns from all domains of science, tech, and art, along with the models and code that generate these images. Covering over 1,200 different models and 100,000 images of patterns and textures from physics, chemistry, biology, sociology, technology, mathematics, and art, this dataset offers a way to explore the connection between the visual patterns that shape our world and the mechanisms that produce them. Created by an agentic AI pipeline that autonomously collects and implements models in standardized form, we use SciTextures to evaluate the ability of leading AI models to link visual patterns to the models and code that generate them, and to identify different patterns that emerged from the same process. We also test AIs ability to infer and recreate the mechanisms behind visual patterns by providing a natural image of a real-world pattern and asking the AI to identify, model, and code the mechanism that formed the pattern, then run this code to generate a simulated image that is compared to the real image. These benchmarks show that vision-language models (VLMs) can understand and simulate the physical system beyond a visual pattern. The dataset and code are available at this URL

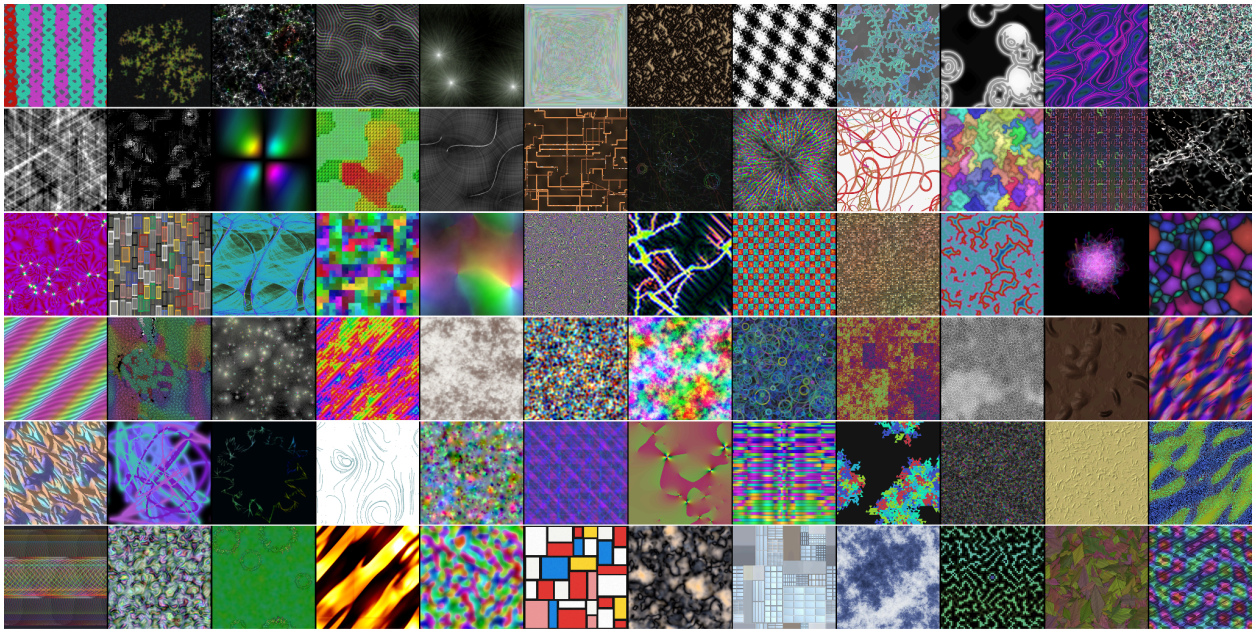

Figure 1: The SCItexture is a large-scale collection of textures and visual patterns, each combined with the generating code (model, simulation, or method) that produced it. The dataset contains 100,000 images from over 1,200 systems, spanning diverse domains of science and art.

AI Hub, Weizmann Institute of Science. (sagieppel@gmail.com, alonast@gmail.com)

The Scitextures Dataset is available at**: Zenodo**

1. Introduction

The visual world is a tapestry of textures and visual patterns emerging from the underlying physical reality[1-11]. The ability to look at a visual pattern and infer the process that formed it represents one of the deepest forms of visual understanding[12,13]. Almost every visual pattern arises from an underlying mechanism, from the self-assembly of materials to the growth of crystals, colonies, and ecosystems, and even the formation of landscapes, plants, and galaxies. Each of these spatial patterns is a visible trace of the hidden dynamics that generate them. Understanding and modeling how such patterns arise and how they connect to the mechanisms that produce them is a central pursuit across science, mathematics, and visual art[1-11]. Numerous works on this problem are scattered across a vast range of domains (Figures 1,2). While some works have sought to treat visual patterns more generally[1-9], to the best of our knowledge, no comprehensive dataset exists that collects these patterns and their corresponding models across a wide range of scientific domains.

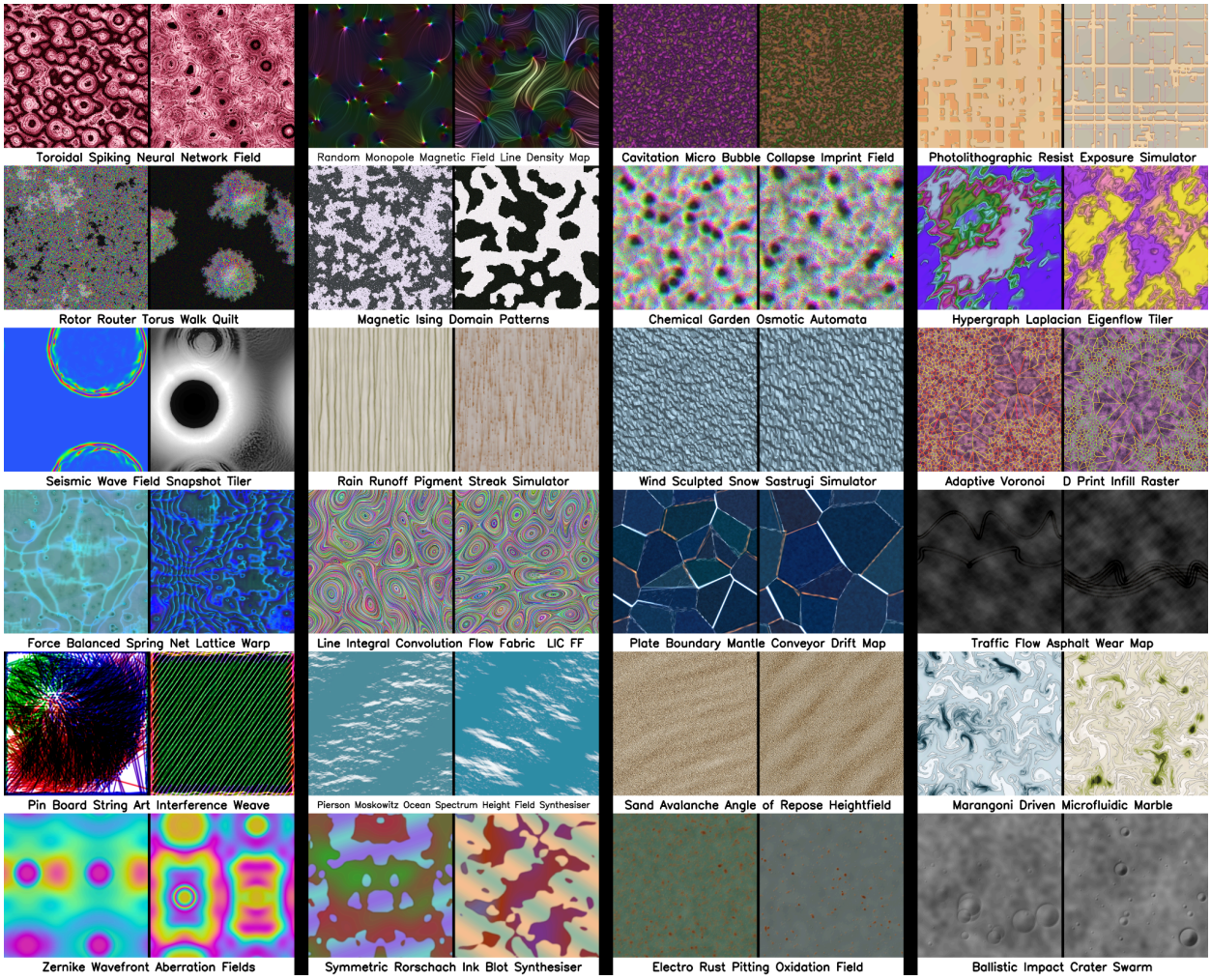

Figure 2. ***Sample image pairs from Scitextures.*** **Each pair of adjacent images was generated by the same generative model, with the model's name shown below the images. Scitextures include 80 images per model, covering over 1,200 different models along with their corresponding simulation code.**

In contrast, computer vision and AI researchers have developed various methods and datasets that explore texture and pattern perception without being limited to specific domains[14,15,45-50]. However, these approaches primarily focus on classifying or matching images or textures of the same type, while ignoring the underlying systems that generate them. As a result, they do not evaluate whether AI systems truly understand the processes behind an image. This work aims to address these limitations by introducing a large-scale, domain-general dataset that brings together visual patterns and textures from across science, technology, and art, alongside the models and code that simulate and generate them (Figures 1,2). This was achieved through an automated, agentic AI pipeline that collects, adapts, and implements models from a wide range of fields in a unified format (Figures 1,2).

A second goal of this work is to probe the depth of AI's visual understanding by assessing its ability to identify and reconstruct the underlying generative processes behind visual patterns. To this end, we introduce three novel benchmarking tasks:

a) Im2Code: Testing the AI ability to match images of visual patterns to the code that generated them (Figure 3).

b) Im2Im: Assessing the AI ability to identify different images that originate from the same underlying process (Figure 3).

c) Im2Sim2Im: Presenting an AI with a real-world image of a pattern, asking it to identify the process that produced it, write code to simulate that process, and run it to generate a synthetic image (Figure 4.left). The generated image is then compared to the real-world input image to assess the model's success (Figure 4.right).

Together, these tasks provide a general framework for evaluating whether AI models possess a deeper, mechanistic understanding of visual patterns. Note that AI in this work refers to Vision Language Models (VLMs).

1.1. Main contributions

a) This work introduces the first large-scale, general dataset that collects textures and visual patterns spanning multiple scientific domains, together with the models and code used to generate them. It enables data-driven exploration of visual patterns that transcend specific domains, linking visual form to underlying mechanisms and offering a new lens on one of science's most universal phenomena.
b) Understanding a system is inherently tied to the ability to model it; likewise, identifying and modeling the mechanism behind an image reflects a deep form of visual understanding, one largely unexplored in computer vision. The methods and dataset presented here provide a means to evaluate this fundamental yet underexplored capability and show that leading AIs can understand and model visual patterns across multiple levels of abstraction.

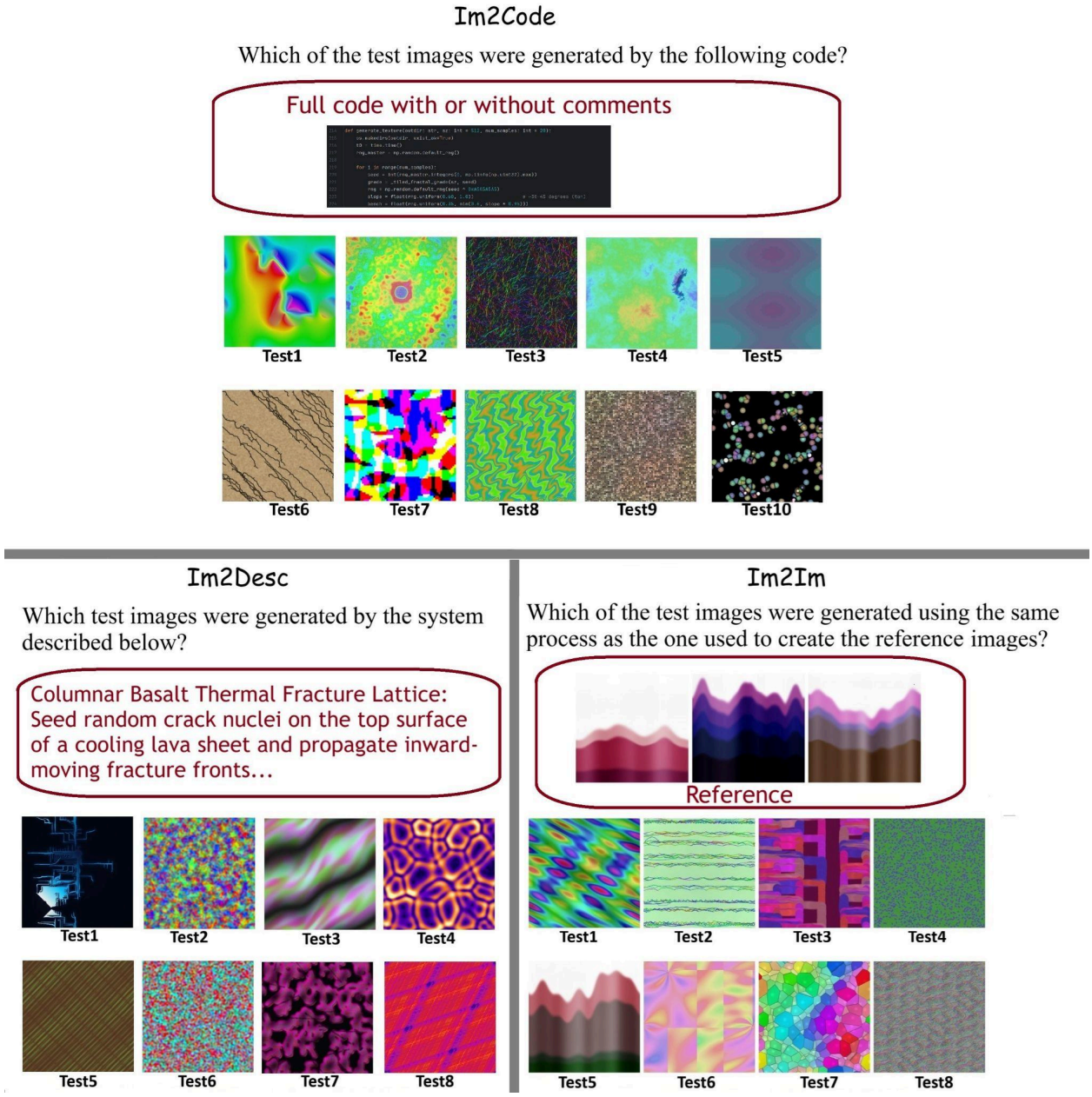

Figure 3. Different approaches for evaluating AI's ability to connect visual patterns with the underlying process that generated them. Im2Code: The AI is given a piece of code (with or without comments) and several images, and must identify which image(s) were generated by that code. Im2Desc: The AI receives a textual description of a generative model along with several images produced by different systems, and must determine which image(s) were generated by the process described. Im2Im: The AI is given a few reference images generated by the same model, along with several test images, and must identify which test image(s) were created by the same process as the reference images.

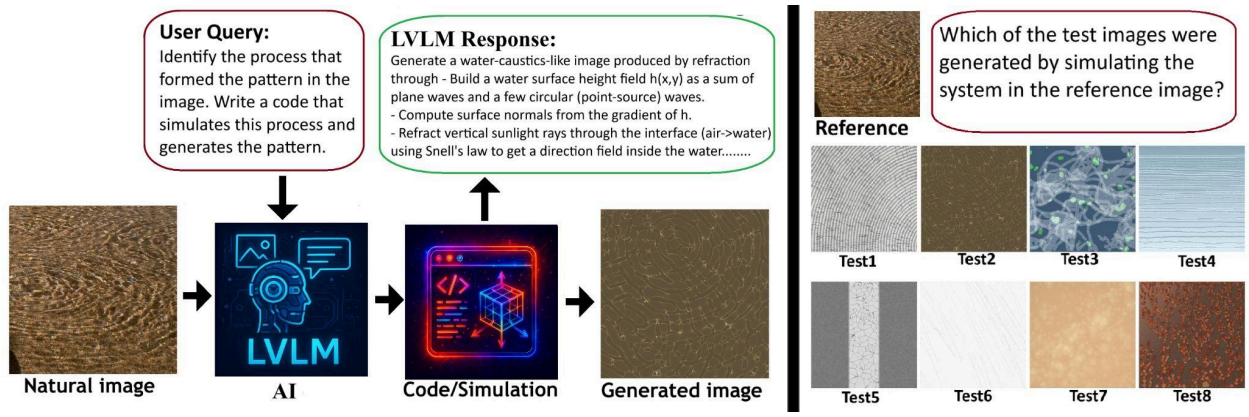

Figure 4. *Im2Sim2Im* approach for testing an AI's ability to identify and model the underlying process behind a real-world visual pattern. Left: The AI receives an image of a real-world pattern, infers the underlying process, implements it as code, and runs the code to generate simulated images. Right: A matcher evaluates and ranks the simulated image by identifying which test image best matches the reference image containing the real-world pattern. The hypothesis is that if the matching process (right) is accurate, it can be indirectly used to evaluate the quality of the generated simulation (left).

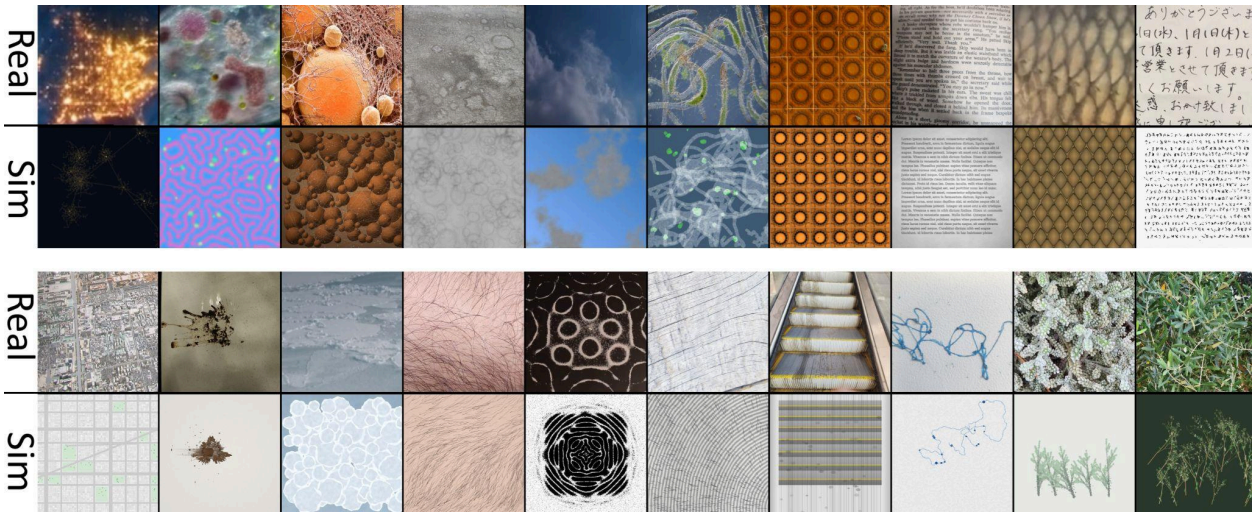

Figure 5. Result from the Im2Sim2Im (Figure 4, left). The "Real" image is a picture of a real-world pattern. The "Sim" image (below the real image) is the image generated by the code/simulation created by the AI in an attempt to model the system in the real image.

2. SciTextures Dataset

The SciTextures dataset is a large-scale collection of images featuring visual patterns and textures from a wide range of scientific and artistic domains, along with the corresponding models and generation code that produce them (Figures 1,2). The dataset spans over 1,270 distinct generative models, ranging from simple systems and mathematical functions, such as the Ising model and the Game of Life, to simulations of cities, materials, chemical reactions,

biological growth, and many others. Each model is accompanied by standardized, flexible code that generates a given number of images at arbitrary resolutions, as well as 80 images produced by the model. Most images are colored and seamlessly tileable, achieved through the use of periodic boundary conditions. In total, the dataset includes over 100,000 images, 1,270 models, and around 300,000 lines of code.

2.1. Dataset Generation: Using Agentic AI to Collect and Implement Models

Identifying visual patterns and modeling the mechanisms that generate them is a central aspect of many domains in science and art[1-12]. From modeling materials and chemical processes to simulating crowd dynamics, weather systems, or digital art, each field develops its own specialized models. As a result, a vast number of models are dispersed across diverse domains, making manual collection and standardization infeasible. To address this, we developed an agentic AI pipeline that autonomously collects, implements, and standardizes models capable of generating visual patterns.

2.2. Agentic AI pipeline for Dataset Generation

The model gathering process begins with an AI agent proposing a set of models for visual pattern generation, either from existing scientific and artistic principles or inferred from natural images. Each model is implemented as executable code, debugged, and reviewed by multiple AI systems to ensure conceptual soundness. Generated images are manually screened for quality and diversity. A manual audit of a few sampled scripts confirmed the reliability of the automated modeling pipeline. All scripts and prompts are supplied in the S.I. More details for each step are presented below:

1) Texture-Model Suggestion: The AI agent (GPT-5) is tasked with proposing a set of models capable of generating visual patterns (five ideas per round) based on either established methods, like published models and simulations from any domain of science, tech, and art, or original concepts. Alternatively, the agent is provided with a natural image of a real-world pattern (Figure 4) and asked to infer and model the process that produced it. The agent has access to all existing models in the dataset and is asked to avoid and filter repeating ideas, and suggest models from as many fields as possible.

2) Code Implementation: The AI agent is then instructed to independently implement each proposed model as standardized code that generates any number of images at variable resolutions.

3) Inspection and Debugging: The generated code is executed, and both the outputs and source code undergo multiple rounds of inspection and debugging. Even when the code runs successfully, additional reviews are performed by multiple AI agents (GPT-5, DeepSeek R1, and Claude Sonnet 4.5) to detect potential conceptual or logical errors.

4) Image Inspection: Few images from each model are inspected both manually and automatically to identify problematic outputs, such as pure noise (manually), uniform images

(automatic), or models that produce repetitive, overly similar results (model must be capable of generating an unlimited variety of distinct images).

5) Model Ranking: Each model is evaluated by additional AI agents (Claude Sonnet 4.5 and DeepSeek R1), ranking how accurately the model simulates the target system (See Sec 2.3 for details).

6) Manual code inspection: Since manually inspecting 300,000 lines of code is not realistic, we have sampled 25 models with which we are familiar and manually inspected them, and could not find any major errors.

2.3. Model Fidelity and Granularity

The degree to which a model faithfully represents the underlying mechanism of a system can differ widely. For example, ocean waves can be simulated using complex fluid and wind dynamics or as a simple sinusoidal function. Each model in the dataset was evaluated and ranked with five categories of fidelity by Claude 4.5 Sonnet. The ranking categories are as follows: **1) Accurate (4%)**: Models that faithfully and quantitatively reproduce the system. **2) Good Approximation (42%):** Models that capture the system's behavior with reasonable accuracy, though not exhaustively. **3) Toy Models (40%)**: Simplified models that capture the core qualitative behavior but lack quantitative fidelity. **4) Weak Approximations (1%)**: Models that represent only minor aspects of the system, missing most critical dynamics. 5) Inspired By (13%): Models, often artistic/graphics, that aim to replicate the system's appearance rather than its underlying mechanisms.

2.4. Prompts and Creativity

We used multiple sets of prompts to guide the AI pipeline; some sets direct the AI toward suggesting established models, while others encourage it to generate novel ideas. The AI identified models across a wide range of disciplines, surpassing, by far, the knowledge of any single person. Even for canonical models, substantial variability exists in parameters, visualization choices, and the projection of patterns into images, providing room for creativity in implementation. When prompted to prioritize creativity, the AI frequently leverages known functions and models, recombining them in novel configurations. In most cases, the models produced colorful images and seamless, tileable patterns (with periodic boundaries), making the results valuable as visual art assets. All prompts are supplied together with the generation code.

3. Testing AI's ability to Extract Model and Code From an Image

The ability to form a representative model of the system beyond a given observation is a core aspect of true understanding. Applied to visual comprehension, this means going beyond surface-level pattern recognition to infer the underlying generative process behind an image. A direct way to evaluate this form of understanding is to test whether AI can analyze an image and identify, or even reconstruct, the system that produced it. We introduce three novel tasks to

assess the AI's ability to link visual patterns to their underlying generative models. Each task was used to evaluate leading Vision Language Models (VLMs), and the results appear in Table 1.

Im2Code/Im2Desc (Image-to-Code Matching): The most straightforward approach involves matching images to their generative code. The AI receives either the full code or verbal descriptions of generative processes as references, along with several test images, and must identify which set of images was created by the process (Figure 3).

Im2Im (Image-to-Image Model Matching): This task provides the AI with a set of reference images all generated by the same process, plus several test images. The AI must identify which test image was produced by the same generative process as the references (Figure 3).

Im2Sim2Im (Image-to-Simulation-to-Image): The most challenging test involves reconstructing the generative model from a real-world image (Figure 4.left). Given a natural photograph of a pattern or texture, the AI must identify the underlying physical process that created it, implement that process as executable code, and run the simulation to generate new images. Unlike the previous tasks, where the AI was required to select from predefined options, here the AI task is to reconstruct the generative model from scratch. The key challenge in this approach is evaluating the accuracy of AI-inferred models. Since the final output is an image generated by the model (Figure 4.left), we propose using visual similarity between the input natural image and the generated output image as a proxy for model accuracy. This approach builds on the observation that both AI and humans excel at the Im2Im matching task (Figure 3), reliably identifying images produced by the same process with high accuracy (Table 1: Im2Im). We leverage this robust matching capability as an evaluation mechanism. We present an AI evaluator with the original natural reference image alongside a test set containing the simulated image (generated by the inferred model) plus several distractor images from alternative models. The matcher identifies which test image best replicates the process that formed the reference image (Figure 4, left). It's clear that the accuracy of this matching task should be correlated to the accuracy of the model that generates the image (Assuming an accurate evaluator). The rationale for the metric is: if a model accurately captures the underlying generative process, it should produce images that match the reference better than images produced from alternative models (that were generated in response to different images). While the correlation between image similarity and matching accuracy and true model fidelity is imperfect, this limitation is inevitable. As far as we know, no universal metric exists for evaluating general-purpose models across arbitrary domains. The ability of a model to recreate observable outputs (in this case, visual patterns) provides a practical and intuitive measure of model quality (Figure 5). Hence, the ability of a model to generate a pattern that looks like the observed pattern counts as a strong indication of its validity [1-5].

Table 1: Results of various tests. All tests were done with 100 questions each, and 10 choices per question, meaning a random guess score is 10%. Im2Desc (Figure 3): matching the model description to the image (matching one description to images from 10 different models, with 3 reference images from each model). Im2Code (Figure 3): Identifying which image was generated by the code, this can be done for code with and without comments (Same format as Im2Code). Im2Im (Figure 3): Identifying which images were generated by the same model (With 3 reference images generated by one model and 10 test images each generated by a different model). Im2Sim2Im (Figure 4): The VLM is given a natural image and is asked to identify the process that generates the pattern in the image, model it, and generate a simulated image. The matching accuracy of the simulated image to the input natural image is used to evaluate the model accuracy (Figure 4). Matching was done with a single real image as reference and 10 synthetic test images, each made by a different model.

Model	Im2Desc	Im2Code (Comments)	Im2Code (Clean)	Im2Model2Code	Im2Model2Im	Im2Im	Im2Sim2Im
GPT_5	44%	**58%**	**50%**	**38%**	48%	**95%**	74%
GPT_5_mini	43%	50%	49%	43%	**50%**	94%	77%
Gemini_2.5_flash	44%	44%	51%	33%	47%	94%	**79%**
Gemini_2.5_pro	**50%**	50%	55%	35%	38%	92%	78%
Qwen25_VL_72B	29%	37%	30%	20%	18%	73%	41%
Llama_4 Maverick_17B	38%	42%	39%	20%	28%	87%	54%
Gemma_3n_E4B	11%	8%	13%	13%	22%	52%	-
Llama_4 Scout_17B	32%	27%	32%	14%	25%	85%	54%
Grok_4_fast reasoning.	13%	14%	14%	11%	21%	47%	62%
Grok_4_fast non_reasoning.	12%	10%	13%	12%	14%	36%	56%
Grok_4.	13%	10%	14%	14%	20%	44%	71%

3. Results

The results of the tests are summarized in Table 1. These results show that most large vision language models (VLMs) can successfully connect the images to the models and code that formed them, with performance substantially exceeding random chance (10% Table 1). All VLMs except Grok4 achieved high accuracy in matching different images generated by the same model (Im2Im; Table 1, Figure 3), with GPT-5 and Gemini Pro 2.5 reaching accuracies up to 95%. In the more challenging Im2Code task of matching images to both the model and the code that produced them, most VLMs achieved accuracies in the 40–60% range. Notably, performance was comparable when image matching to code (Im2Code) and to descriptions (Im2Desc) (Table 1, Fig. 3). This suggests that VLMs extract similar information from code and textual model descriptions. In the Im2Code task, we compared performance on matching images to the clean code (without comments) and annotated code (with comments). Both conditions yielded similar accuracy, indicating that comments provided no additional information to the VLMS. When tasked with identifying and replicating the generative process behind real-world

pattern images (Im2Sim2Im; Figures 4, 5), VLMs produced impressive results (Table 1). Showing VLM's ability to produce models that captured key physical aspects of the real systems, at least at a simplified toy model level (Figure 5).

What Does the AI See? For all tests, the VLMs were asked to explain their answers. These explanations reveal that the VLMs mostly focus on inferring visual features (such as smooth transitions and scanline-like patterns) from the code and match them to features extracted from the images. While the visual features inferred from both code and images were mostly accurate, they tended to be general and often lacked distinctiveness, leading to matching errors.

3.1. Im2Model2Code: From Image to Model to Code

When matching images to code, all VLMs appeared to prioritize inferring visual patterns from the code/model over extracting the underlying generative process from the image (see above). This suggests that the VLMs' approach to this task relies more heavily on their ability to infer visual features from code than on their ability to reverse-engineer the model from the image. To address this limitation, we redesigned the test with a two-stage process. First, one VLM was asked to examine each image and infer the generative process/model behind it. These textual descriptions were then provided (without the images) to a second LLM, which was asked to match the described systems to the corresponding code. This approach forces the first model to extract the underlying model from the image, rather than simply inferring visual patterns from code, and the second LLM to use only model description and not image patterns for matching. As shown in Table 1 (Im2Model2Code vs Im2Code), this two-stage approach yielded lower accuracy but still performed above random chance. This indicates that most VLMs possess a limited but clear ability to infer generative models from images. We also inspected a few inferred models to qualitatively validate this and found that, for simple cases, these models matched the physical system in the image.

3.2. Im2Sim2Im: Results Extracting Model From Image

The Im2Sim2Im test (Figure 4) involves giving the AI an image of a real-world pattern and asking it to identify the process that forms this pattern, implement it in code, and run the code to generate a simulated image. This task required creating models and code, with only an image as a reference, making it considerably more challenging than previous tests. Most VLMs easily identified the visual patterns and their physical sources. This is not surprising since the test images depicted well-known phenomena (foam, dust, clouds, plants, cities; Figure 5). Accurately simulating many of these patterns is complex and often computationally infeasible. Consequently, in the majority of cases, the VLMs generated simplified toy models rather than accurate simulations. The relatively good scores VLMs got in this test (Table 1:Im2Sim2Im) imply that the generated models capture the main essence of the system. A notable result was the ability of leading VLMs to deal with complex systems by representing each of their components in different levels of abstraction. For example, when presented with an image of

sunlight reflecting on water waves (Figure 4), GPT-5 represented the waves as a simple sinusoidal function, then used the surface normals with Snell's law to calculate light reflections and refractions from both the water surface and pool floor (Figure 4). This demonstrates an impressive capacity to integrate crude approximations (sinusoids for waves) with real physics (Snell's law) to model complex systems using only a single reference image. This ability to represent different elements of the system in different levels of abstraction is a sign of deep understanding. While the generated images were crude (Figure 5), they captured essential system characteristics and demonstrated that the VLMs understood both the patterns and their underlying physics. For instance, when presented with a printed book page, GPT simulated a Markovian chain of letters (Figure 5). Smaller VLMs, such as GPT-5-mini and Gemini-2.5 Flash, often produced overly simplistic but more tunable models. Interestingly, these simplified models were frequently better tuned to the input image and more successfully replicated pattern appearance compared to more physically accurate simulations of the same patterns made by larger VLMs. This led to these smaller VLMs outperforming their larger counterparts in this task (Table 1:Im2Sim2Im). Whether less accurate simulations that are more tunable and hence can be better fitted to the observed pattern can be considered superior to more physically accurate but less tunable models is an open question in science.

4. Conclusion

This work leverages recent advances in Agentic AI to tackle two fundamental challenges in visual understanding: the formation of visual patterns and the modeling of their underlying mechanisms. Our first contribution is a large-scale dataset of visual patterns and textures spanning science, technology, and art. Critically, the dataset includes not only images but also the models and code that generate them, enabling deep exploration of the relationship between visual patterns and the processes that produce them. Beyond its wide range of potential applications in fields that rely on textures and patterns[42-50], this dataset opens the door to general exploration of the connection between visual patterns and the mechanisms that form them. Understanding the processes underlying visual patterns, rather than merely recognizing them, is a core aspect of genuine comprehension, yet remains largely unexplored in computer vision. To address this gap, we introduce three novel evaluation methods that assess AI’s capacity to link images to the systems that formed them. Our results demonstrate that VLMs are capable of inferring models and mechanisms from images. Demonstrating the ability to understand and model processes in multiple layers of abstraction. This level of understanding transcends traditional image classification and description, marking a significant step toward general visual intelligence. Developing the tools to explore, harness, and enhance this capability constitutes a central challenge for future research in computer vision and AI.

5. Related Works

Identifying visual patterns and the processes that generate them is central to both science[1-8] and visual art[9-11]. While numerous models, simulations, and graphics techniques[50] exist for these tasks, they remain scattered across disparate fields. Few works have explored this territory in a more unified way[1-9, 50], often focusing on common mathematical principles like fractals[8,9], cellular automata[6], self-organization and emergence[2-5] that universally recur across different systems.

Datasets of visual patterns and textures typically focus on specific phenomena and rarely include generation models. The largest source of linked visual patterns and executable scripts comes from OpenGL/WebGL shaders: short web scripts used to generate real-time graphics and simple videos in websites. The most extensive collection of such scripts is ShaderToy[16,17], containing over 51,000 shaders that can each generate video output. However, these shaders are designed for CGI and artistic projects with real-time rendering capabilities (~979 frames per second[17]), not for scientific models and simulations, which typically cannot run in real time and prioritize physical accuracy over visual aesthetics. Another source of texture-program pairs comes from node-graph systems for PBR materials and shaders, though these remain art-focused and limited to surface textures[18]. Scientific datasets that combine code and patterns are typically domain-specific, concentrating on areas like fractals[19], metamaterials[20], morphogenesis[21,22], or procedurally generated training data[23,24].

This work also relates to visual program synthesis, which involves inferring underlying programs from images. Works in this field focused on extracting simple code from technical diagrams such as CAD, SVG, Webpages, or code diagrams[25-37]. While early approaches used specialized neural networks[26-29], recent work has begun exploring foundation vision-language models (VLMs)[34]. One example task involves reconstructing the node-graph that generated a given texture image[34-37] (node-graphs being a form of visual scripting used to create textures). An older but related approach is exemplar-based texture synthesis, which can be viewed as simplified model inference from images[38-40]. This approach focuses on identifying the statistical building blocks[41] of a pattern or finding system parameters that replicate it.

6. Reference

1. Ball, Philip. The self-made tapestry: pattern formation in nature. Oxford university press, 1998.
2. Turing, Alan Mathison. "The chemical basis of morphogenesis." Bulletin of mathematical biology 52.1 (1990): 153-197.
3. Brent, Sandor B. "Prigogine's model for self-organization in nonequilibrium systems: Its relevance for developmental psychology." Human Development 21.5/6 (1978): 374-387.

4. Thomson, J. Arthur. "On growth and form." (1917): 21-22.
5. Cross, Mark C., and Pierre C. Hohenberg. "Pattern formation outside of equilibrium." Reviews of modern physics 65.3 (1993): 851.
6. Wolfram, Stephen, and M. Gad-el-Hak. "A new kind of science." Appl. Mech. Rev. 56.2 (2003): B18-B19.
7. Hofstadter, Douglas R. Gödel, Escher, Bach: an eternal golden braid. Basic books, 1999.
8. Mandelbrot, Benoit B. "The fractal geometry of nature/Revised and enlarged edition." New York (1983).
9. Flake, Gary William. The computational beauty of nature: Computer explorations of fractals, chaos, complex systems, and adaptation. MIT press, 2000.
10. https://www.youtube.com/@SebastianLague
11. https://www.youtube.com/@EmergentGarden
12. Gregory, Richard L. "Eye and brain: The psychology of seeing." (1966): 1-288.
13. Rock, Irvin. "The logic of perception." (1983).
14. Oquab, Maxime, et al. "Dinov2: Learning robust visual features without supervision." arXiv preprint arXiv:2304.07193 (2023).
15. Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In ICML, 2020.
16. Jeremias, Pol, and Inigo Quilez. "Shadertoy: Learn to create everything in a fragment shader." SIGGRAPH Asia 2014 Courses. 2014. 1-15.
17. Baradad, Manel, et al. "Procedural image programs for representation learning." Advances in Neural Information Processing Systems 35 (2022): 6450-6462.
18. https://www.blenderkit.com/
19. Kataoka, Hirokatsu, et al. "Pre-training without natural images." Proceedings of the Asian Conference on Computer Vision. 2020.
20. Makatura, Liane, et al. "MetaGen: A DSL, Database, and Benchmark for VLM-Assisted Metamaterial Generation." arXiv preprint arXiv:2508.17568 (2025).
21. Lomas, Andy. "Cellular forms: an artistic exploration of morphogenesis." SIGGRAPH studio. 2014.
22. Lomas, Andy. "Species Explorer: An interface for artistic exploration of multi-dimensional parameter spaces." Electronic visualisation and the arts. BCS Learning & Development, 2016.
23. Cobbe, Karl, et al. "Leveraging procedural generation to benchmark reinforcement learning." International conference on machine learning. PMLR, 2020.
24. Bray, Mark-Anthony, et al. "Workflow and metrics for image quality control in large-scale high-content screens." Journal of biomolecular screening 17.2 (2012): 266-274.
25. Pearl, Ofek, et al. "Geocode: Interpretable shape programs." Computer Graphics Forum. Vol. 44. No. 1. 2025.

26. Carlier, Alexandre, et al. "Deepsvg: A hierarchical generative network for vector graphics animation." Advances in Neural Information Processing Systems 33 (2020): 16351-16361.
27. Reddy, Pradyumna, et al. "Im2vec: Synthesizing vector graphics without vector supervision." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.
28. Du, Tao, et al. "Inversecsg: Automatic conversion of 3d models to csg trees." ACM Transactions on Graphics (TOG) 37.6 (2018): 1-16.
29. Wu, Rundi, Chang Xiao, and Changxi Zheng. "Deepcad: A deep generative network for computer-aided design models. in 2021 ieee." CVF International Conference on Computer Vision (ICCV). 2021.
30. Beltramelli, Tony. "pix2code: Generating code from a graphical user interface screenshot." Proceedings of the ACM SIGCHI symposium on engineering interactive computing systems. 2018.
31. Doris, Anna C., et al. "CAD-Coder: An Open-Source Vision-Language Model for Computer-Aided Design Code Generation." arXiv preprint arXiv:2505.14646 (2025).
32. Ren, Xinxing, Qianbo Zang, and Zekun Guo. "SimuGen: Multi-modal Agentic Framework for Constructing Block Diagram-Based Simulation Models." arXiv preprint arXiv:2506.15695 (2025).
33. Bates, Averi, et al. "Unified modeling language code generation from diagram images using multimodal large language models." Machine Learning with Applications (2025): 100660.
34. Li, Beichen, et al. "VLMaterial: Procedural Material Generation with Large Vision-Language Models." arXiv preprint arXiv:2501.18623 (2025).
35. Hu, Yiwei, et al. "Generating procedural materials from text or image prompts." ACM SIGGRAPH 2023 conference proceedings. 2023.
36. Xing, Youxin, et al. "A Survey of Inverse Recovery Methods of Highly Realistic Surface Materials." Journal of Computer-Aided Design & Computer Graphics.
37. Guerrero, Paul, et al. "Matformer: A generative model for procedural materials." arXiv preprint arXiv:2207.01044 (2022).
38. Raad, Lara, et al. "A survey of exemplar-based texture synthesis." arXiv preprint arXiv:1707.07184 (2017).
39. Mumford, David, and Agnès Desolneux. Pattern theory: the stochastic analysis of real-world signals. CRC Press, 2010.
40. Liu, Jun, et al. "Perception-driven procedural texture generation from examples." Neurocomputing 291 (2018): 21-34.
41. Julesz, Bela. "A theory of preattentive texture discrimination based on first-order statistics of textons." Biological Cybernetics 41.2 (1981): 131-138.
42. Ambientcg, free pbr textures repositories. https:// ambientcg.com/.
43. freepbr, free pbr textures repositories. https:// freepbr.com/.
44. Polyhaven free hdri repository. https://polyhaven. Com.

45. Drehwald, Manuel S., et al. "One-shot recognition of any material anywhere using contrastive learning with physics-based rendering." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023.
46. Cimpoi, Mircea, et al. "Describing textures in the wild." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.
47. Sharma, Prafull, et al. "Materialistic: Selecting similar materials in images." ACM Transactions on Graphics 42.4 (2023).
48. Sagi Eppel, Li, Jolina, Manuel Drehwald, and Alan Aspuru-Guzik. "Infusing synthetic data with real-world patterns for zero-shot material state segmentation." Advances in Neural Information Processing Systems 37 (2024): 60237-60259.
49. Vecchio, Giuseppe, and Valentin Deschaintre. "Matsynth: A modern pbr materials dataset." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024.
50. Raistrick, Alexander, et al. "Infinite photorealistic worlds using procedural generation." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023.