

KNOWING IS NOT SEEING: LIMITS OF PHYSICAL PROBLEM SOLVING IN VLMS

**Karim Elmaaroufi^{1,2}, Kevin Chon, Justin Svegliato¹, Lakshya A Agrawal¹,
Matei Zaharia¹, Sanjit A. Seshia¹**

¹EECS, UC Berkeley ²Embodied Science
elmaaroufi@berkeley.edu

ABSTRACT

Perception enables humans and animals to interact with the world. Combined with our knowledge of physical laws, we rely on perception to complete many tasks such as tracking a ball or driving a car. While we learn these physical laws through perception, proprioception, and physical interaction, Vision Language Models (VLMs) have seemingly learned the same physical laws strictly through natural language. In this paper, we leverage existing puzzles designed to test physical problem solving in humans to instead evaluate VLMs. Unlike prior work evaluating only artificial agents, we leverage games designed for and tested on humans. We design a new reinforcement learning environment over the original game that allows VLMs to play the game in multiple rounds. With a direct comparison between humans and VLMs on the same 28 levels with the same number of attempts, we show that humans average a 61% pass rate within 4 attempts, while the best-performing VLM without access to code representing the game state solves only 9/28 levels. Only Gemini 3 Pro with a textual JSON representation of the game state exceeds human-level performance (23/28), revealing that physical reasoning is unlocked through text, not vision. Our results suggest that while VLMs can describe Newton’s Laws of Motion in detail, they cannot apply these principles to solve challenging physical problems that humans can often solve in just a few attempts.

1 INTRODUCTION

Vision-Language Models (VLMs) now achieve expert-level performance on a range of scientific question-answering benchmarks, including biology, chemistry, and physics (noa; Rein et al., 2023). Their strong performance on topics such as classical mechanics suggests these models possess sophisticated physical reasoning capabilities learned during training. However, when asked to apply this knowledge to physical tasks—such as placing tools to redirect falling objects or construct makeshift levers—VLMs fail catastrophically, even when humans can solve them in a handful of attempts. This reveals a fundamental limitation: although VLMs possess substantial knowledge about the physical world, this knowledge does not reliably translate to solving physical tasks when those tasks are encoded visually.

Instead of introducing a new benchmark, we build an interface that enables VLMs to interact with a well-established, theoretically grounded game designed by cognitive scientists to probe human physical reasoning—*The Virtual Tools Game* (Allen et al., 2020). By observing VLMs playing this game, we identify three key findings. First, by providing the full game state in text form, Gemini 3 Pro nearly triples its performance, showing that mechanical and physical reasoning is more effectively unlocked when problem solving in text rather than image space. Second, VLMs constantly place tools in positions that are unfaithful to their stated plans. Third, without a textual representation of game state, VLMs cannot learn from their past failed attempts, repeating their same failed strategy even when shown trajectory visualizations of their failures. These findings agree with prior work and support the conclusion that VLMs do not reason in visual space but only in text.

2 BACKGROUND AND RELATED WORK

Human adults have a remarkable ability to rapidly understand, simulate, and act in the real world, often within fractions of a second. At the limits of human performance, Formula 1 drivers navigate turns at speeds exceeding 230 mph while avoiding contact with other drivers. Tom Brady, the winningest NFL quarterback in history, averaged only 2.5 seconds in his final season to observe all

Table 1: Experimental settings for VLM gameplay.

Setting	Description
Zero-Shot	Image only. Screenshot + minimal task prompt; no examples or state.
Zero-Shot + JSON	Image + state JSON. Same prompt with full level JSON (objects, physics, goal).
Few-Shot	Image + examples. Zero-shot prompt + 3 curated (image, placement, reasoning) e.g.
Few-Shot + JSON	Image + examples + JSON. Few-shot prompt + JSON for level & all examples
GEPA	Same as zero-shot but system prompt is optimized by GEPA on train levels.

21 other players on the field, select a receiver, and deliver a precise pass while avoiding tackles. While much of a professional athlete’s ability is shaped by training, there are certain principles and laws that they know apply even at home, away from their competition. In general, for all humans, what do we learn and when do we learn it?

Developmental studies indicate that physical world knowledge emerges before language. In fact, many principles of the physical world have been detected in infants “as early as we can measure it” (Hespos & vanMarle, 2012). By two months of age, infants demonstrate knowledge of containment (i.e., objects can be placed inside other objects) and non-interpenetration (i.e., two objects cannot occupy the same physical space) (Hespos & Baillargeon, 2001). At three months, they demonstrate 3D spatial awareness (Kestenbaum et al., 1987), knowledge of gravity (Baillargeon) (i.e., unsupported objects must fall), and object permanence (Baillargeon) (i.e., an object is still present even if occluded from view).

Given that VLMs have a rich knowledge over language, existing work has studied whether VLMs also possess physical world knowledge, something humans have already learned by the time they learn language. Kang et al. (2025) develop a simple 2D simulation environment that supports collisions and Newtonian kinematics. Their results show that current video generation models “mimic their closest training examples” instead of learning general physical rules like Newton’s Laws of Motion. Le et al. (2025) establish similar findings, and Zhao et al. (2025) find similar limitations in agents trained through reinforcement learning algorithms like PPO (Schulman et al., 2017), A3C (Mnih et al., 2016), SAC (Haarnoja et al., 2018), and DDPG (Lillicrap et al., 2019). In more complex environments like games, VLMs “struggle to translate descriptive physical knowledge into precise, predictive control” (Xu et al., 2025; Xue et al., 2021; Taesiri et al., 2025), and in embodied applications (Li et al.). These results extend beyond physical understanding in VLMs to generative models that produce videos with stunning visual realism but lack correct knowledge of physical dynamics (Zhang et al., 2025; Motamed et al., 2025; Zhao et al., 2025; Bansal et al., 2024).

3 PLAYING THE VIRTUAL TOOLS GAME

The Virtual Tools Game (Allen et al., 2020) was designed to evaluate mechanical reasoning—“the ability to spontaneously repurpose objects in our environment to accomplish a novel goal.” The game is based on principles in Newtonian kinematics like elasticity, friction, and gravity. There are 18 levels with corresponding results from 50 human participants allowing for “precise, quantifiable comparisons between humans and machine agents.” There are an additional 10 validation levels. In each level of the game, the objective is to move a red object into a green goal area and keep it there for 2 seconds. To achieve this, a number of high-level concepts must be used. We group the 28 available levels into five concepts: levers (5), drops (4), blocking (5), bridging (9), and chaining (5). These concepts are immediately obvious as they must be applied by selecting one of three tools in each level. Using the tools directly is not always possible, instead requiring using concepts like gravity in the physical world. However, such tool use is not unique to artificial environments like games. Tool use informed by physical understanding has been observed in American crows that time the use of cars as nutcrackers (Grobecker & Pietsch, 1978) and in chimpanzees that exhibit aimed throwing behavior (Goodall, 1964). More broadly, mechanical reasoning—both in this game and in the real world—requires long-horizon causal reasoning. Reflecting the difficulty of the task, even human participants could not solve all levels, with some levels exhibiting success rates below 40% despite more than 20 attempts.

Agent Gameplay We build a new interface over the original game to allow for a simple interaction for VLMs without compromising game functionality. Similar to an RL environment, we define an *init*, *reset*, and *step* interface. The *step* interface allows for a single action that includes a choice of tool and Cartesian coordinates (x, y) of where the center of the tool should be placed. The simulation stops and a binary reward is returned once there is no movement in the environment.

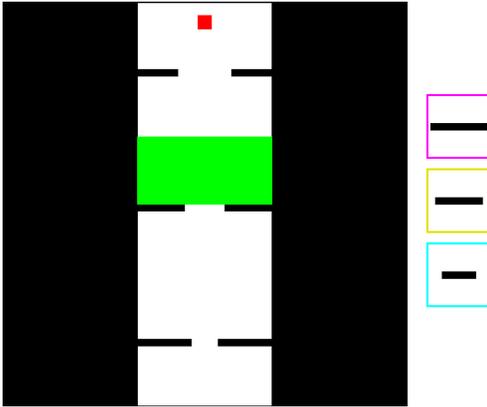


Figure 1: The Elevator Level. When the game starts, the red block will fall as it is unsupported. The winning strategy is to block it from falling past the green goal zone by placing a large enough horizontal rectangular tool. This example image along with the winning strategy, tool, and coordinates is provided as one of three example solutions during few-shot experiments.

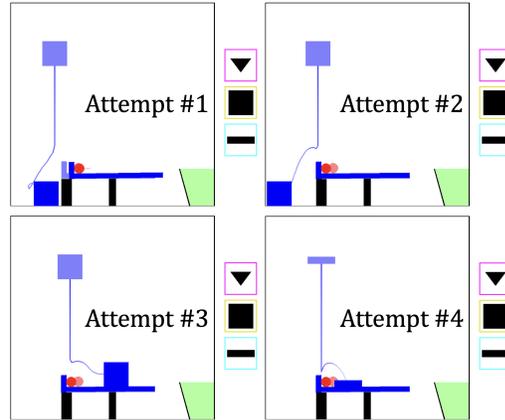


Figure 2: The Catapult Level. GPT-5 was asked to play this level with 4 attempts. In all attempts, it committed to the strategy, “drives it (*the ball*) along the platform and off the end into the green bin”. Despite knowledge of gravity, GPT-5 ignores the possibility of gravity pulling the ball down before it could reach the goal area. New strategies are never explored.

A running example of a game level, *Elevator*, is provided in Figure 1. In this level, a red block is suspended in air and three tools are shown on the right side. All three tools are horizontal rectangles of various sizes. Upon selection of a tool, a VLM can then choose where to place it. They can place it anywhere except in black areas (these are immovable). The key insight in this level is to recognize that gravity will cause the unsupported block to fall. To have the block remain in the green area for two seconds, we merely have to select the largest rectangular object and place it such that it blocks the bottom of the green area so that the red block cannot fall all the way through.

We design five experiments for our VLMs as shown in Table 1. An image of the starting environment and tool palette, an explanation of the task, and reminders of physical laws such as gravity are given to the VLM (see Appendix A.1 for the prompt). Due to context limits, the VLM is allowed four attempts per level. On success, it advances to the next level. On failure, a trajectory visualization is appended to the prompt and the VLM is asked to analyze the failure before retrying, enabling it to learn from previous mistakes. An example trajectory image is in Appendix Figure 4. When a round ends, we return to the original prompt without any trajectories. The first four experiments are: zero-shot; zero-shot with the full game state represented as JSON; few-shot (with 3 solved examples, one of which is our running example, the Elevator level); and few-shot with JSON. An example of JSON game state including parameterized physics settings for properties such as elasticity and gravity is in Appendix Figure 9. The VLM may choose to use Chain-of-Thought (CoT) (Wei et al., 2023) in its decision making, but the end of the response must contain a JSON containing four fields: the selected object, the x coordinate, the y coordinate, and a brief sentence explaining why it made its selection. An example response with CoT is in Appendix A.4.

In the last experiment, we use Genetic-Pareto (GEPA), a prompt optimizer that uses VLMs to reflect on both successful and failed trajectories to propose system prompt updates (Agrawal et al., 2025). In GEPA, natural language reflection is used to generate high-level rules, or in our case, physical laws and principles that govern these trajectories, which are then incorporated back into the system prompt. If the new prompts improve performance, they are added to a candidate pool and are used to advance the Pareto frontier of the VLMs’ attempts. To further aid the VLM’s reflection process, we include a rule-based method to generate additional feedback from an attempt. For example, our feedback method may look for attempts where the VLM placed the object in an invalid position and provide this as explicit feedback to the reflective model, which will then synthesize new rules to include in a new candidate system prompt. We use the same model, GPT-5, as both the VLM that plays the game and the one that does the reflection with the zero-shot prompt as the initial prompt.

Table 2: VLM gameplay performance across experimental settings compared to human baseline. We list the number of levels solved with a maximum of 4 attempts per level. In GEPA, we list scores both before and after prompt optimization with GPT-5 as both the task and reflection model.

	GPT-5	Qwen3-VL-4B-Inst	Gemini 3 Pro	Claude 4.5 Sonnet
Humans	Avg. pass rate: 61% across 28 levels (range: 4–98%; see Appendix A.6)			
Zero-Shot	5/28	1/28	8/28	4/28
Zero-Shot + JSON	8/28	1/28	22/28	1/28
Few-Shot	4/28	1/28	9/28	3/28
Few-Shot + JSON	6/28	0/28	23/28	3/28
GEPA (GPT-5)	5/28 → 5/28	—	—	—

4 RESULTS

We summarize the results of the five experiments using four VLMs in Table 2. Claude 4.5 Sonnet, Qwen3 VL 4B, and GPT-5 all cannot solve more than 30% of levels. Human participants solving the same levels under the same 4-attempt limit average a 61% pass rate (range: 4–98%; Appendix A.6); without JSON state, even the best VLM (Gemini 3 Pro, 9/28) falls far below human performance, and only with a textual representation does Gemini (23/28) exceed human-majority levels. Notably, we observe that few-shot examples with solutions do not lift the performance of any of the four models, which is in stark contrast to text-only domains where few-shot learning has led to much success (Brown et al., 2020; Parnami & Lee, 2022). However, providing the full game state does positively affect one model, Gemini 3 Pro. Gemini’s performance is nearly tripled when the JSON representation of the game state is provided, but for the other three models, providing the state does not meaningfully affect their performance. Finally, in the GEPA experiment, we see that the new synthesized prompt does not offer additional gains over the initial zero-shot prompt.

Existing work shows that LLMs can accurately answer questions about the physical world (Hendrycks et al., 2021; Wang et al., 2024). However, our experiments show that these models cannot effectively use this knowledge. We perform a qualitative analysis over the responses to understand this failure. Here, we find that the VLMs produce reasonable strategies but cannot put them to use. A representative example is in Appendix A.4. In this example, Gemini 3 Pro generates a reasoning trace with the correct strategy, but the desired tool rotation/placement are incorrect and unfaithful to its plan. Similar results were observed for all VLMs and experiments. Semantic analysis grouped by failure categories is provided in Appendix A.5.

We create heatmaps of VLM gameplay to visualize their strategies and determine if they learn from failures. Heatmaps for all levels are in Appendix Figures 10 & 11. We see that VLMs struggle to learn from their failed attempts without the game state. For example, in the Shafts B level, both zero-shot and few-shot attempts are often made in illegal areas, and the VLMs never attempt a new high-level plan. However, when the game state is provided, if they choose an illegal action, they learn from it and shift to the correct strategy of trying to drop an object down one of the two shafts. One explanation for why a VLM cannot learn without a text game representation is that VLMs struggle to reason in vision space. Prior work has found that despite image tokens comprising more than 90% of input tokens, a similar percentage of attention activations occur in the language layers and not vision layers (Chen et al., 2025). Elmaaroufi et al. (2025) found that VLMs cannot learn spatial reasoning when their language layers were frozen despite the rest of the model being unfrozen.

We hope that identifying these limitations—both in learning from visual attempts and in generating actions faithful to plans—will motivate further research in visual reasoning.

5 CONCLUSION

In this work, we repurpose a cognitive science game designed to evaluate mechanical reasoning, to perform the same test on VLMs. By leveraging this game, we benefit from a theoretically principled probe of visual reasoning abilities already measured and verified in humans. Under the same 4-attempt limit, humans average 61% across all levels while most VLMs solve under 30% from vision alone, confirming a large human-VLM gap. While VLMs can speak convincingly of the physical world and can even formulate nearly perfect plans, they cannot implement their knowledge in the physical world. Their actions are often unfaithful to the reasoning traces which do contain the correct strategies. Lastly, we find that VLMs cannot learn from their past mistakes without a textual representation of game state, implying that reasoning only happens in text space.

6 ETHICS

This paper presents work whose goal is to advance the field of AI models that can understand and interact in the real world. There are many potential societal consequences of such AI agents, however, this work is a small step towards understanding current limitations towards that goal. At this time, there are no consequences of which we feel must be specifically highlighted here.

7 REPRODUCIBILITY

The Virtual Tools Game (Allen et al., 2020) is publicly available for all to play. The prompts we use are provided in the appendix. Lastly, after the review period, we will release our code to simplify reproducibility.

REFERENCES

- GPQA Diamond. URL <https://epoch.ai/benchmarks/gpqa-diamond>.
- Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziemis, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning, July 2025. URL <http://arxiv.org/abs/2507.19457>. arXiv:2507.19457 [cs].
- Kelsey R. Allen, Kevin A. Smith, and Joshua B. Tenenbaum. Rapid trial-and-error learning with simulation supports flexible tool use and physical reasoning. *Proceedings of the National Academy of Sciences*, 117(47):29302–29310, November 2020. doi: 10.1073/pnas.1912341117. URL <https://www.pnas.org/doi/10.1073/pnas.1912341117>.
- Renee Baillargeon. PhysicalReasoningin Infancy.
- Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. VideoPhy: Evaluating Physical Commonsense for Video Generation. 2024. doi: 10.48550/ARXIV.2406.03520. URL <https://arxiv.org/abs/2406.03520>. Version Number: 2.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL <https://arxiv.org/abs/2005.14165>.
- Shiqi Chen, Tongyao Zhu, Ruochen Zhou, Jinghan Zhang, Siyang Gao, Juan Carlos Niebles, Mor Geva, Junxian He, Jiajun Wu, and Manling Li. Why is spatial reasoning hard for vlms? an attention mechanism perspective on focus areas, 2025. URL <https://arxiv.org/abs/2503.01773>.
- Karim Elmaaroufi, Liheng Lai, Justin Svegliato, Yutong Bai, Sanjit A. Seshia, and Matei Zaharia. Graid: Enhancing spatial reasoning of vlms through high-fidelity data generation, 2025. URL <https://arxiv.org/abs/2510.22118>.
- Jane Goodall. Tool-Using and Aimed Throwing in a Community of Free-Living Chimpanzees. *Nature*, 201(4926):1264–1266, March 1964. ISSN 1476-4687. doi: 10.1038/2011264a0. URL <https://www.nature.com/articles/2011264a0>.
- David B Grobecker and Theodore W Pietsch. Crows use automobiles as nutcrackers. *The Auk*, 95(4):23, 1978.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL <https://arxiv.org/abs/1801.01290>.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL <https://arxiv.org/abs/2009.03300>.
- Susan J. Hespos and Renée Baillargeon. Reasoning about containment events in very young infants. *Cognition*, 78(3):207–245, 2001. ISSN 0010-0277. doi: [https://doi.org/10.1016/S0010-0277\(00\)00118-9](https://doi.org/10.1016/S0010-0277(00)00118-9). URL <https://www.sciencedirect.com/science/article/pii/S0010027700001189>.
- Susan J. Hespos and Kristy vanMarle. Physics for infants: characterizing the origins of knowledge about objects, substances, and number. *WIREs Cognitive Science*, 3(1):19–27, 2012. ISSN 1939-5086. doi: 10.1002/wcs.157. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/wcs.157>. eprint: <https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wcs.157>.
- 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, June 2025. URL <http://arxiv.org/abs/2411.02385>. arXiv:2411.02385 [cs].
- Roberta Kestenbaum, Nancy Termine, and Elizabeth S. Spelke. Perception of objects and object boundaries by 3-month-old infants. *British Journal of Developmental Psychology*, 5(4):367–383, 1987. ISSN 2044-835X. doi: 10.1111/j.2044-835X.1987.tb01073.x. URL <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.2044-835X.1987.tb01073.x>. eprint: <https://bpspsychub.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2044-835X.1987.tb01073.x>.
- Minh-Quan Le, Yuanzhi Zhu, Vicky Kalogeiton, and Dimitris Samaras. What about gravity in video generation? post-training newton’s laws with verifiable rewards, 2025. URL <https://arxiv.org/abs/2512.00425>.
- Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Li Erran Li, Ruohan Zhang, Weiyu Liu, Percy Liang, Li Fei-Fei, Jiayuan Mao, and Jiajun Wu. Embodied Agent Interface: Benchmarking LLMs for Embodied Decision Making.
- Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019. URL <https://arxiv.org/abs/1509.02971>.
- Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016. URL <https://arxiv.org/abs/1602.01783>.
- Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles?, February 2025. URL <http://arxiv.org/abs/2501.09038>. arXiv:2501.09038 [cs].
- Archit Parnami and Minwoo Lee. Learning from few examples: A summary of approaches to few-shot learning, 2022. URL <https://arxiv.org/abs/2203.04291>.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof qa benchmark, 2023. URL <https://arxiv.org/abs/2311.12022>.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.
- Mohammad Reza Taesiri, Abhijay Ghildyal, Saman Zadootaghaj, Nabajeet Barman, and Cor-Paul Bezemer. VideoGameQA-Bench: Evaluating Vision-Language Models for Video Game Quality Assurance, December 2025. URL <http://arxiv.org/abs/2505.15952>. arXiv:2505.15952 [cs].
- Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhui Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL <https://arxiv.org/abs/2406.01574>.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL <https://arxiv.org/abs/2201.11903>.

Xinrun Xu, Pi Bu, Ye Wang, Börje F. Karlsson, Ziming Wang, Tengtao Song, Qi Zhu, Jun Song, Zhiming Ding, and Bo Zheng. DeepPHY: Benchmarking Agentic VLMs on Physical Reasoning, August 2025. URL <http://arxiv.org/abs/2508.05405>. arXiv:2508.05405 [cs].

Cheng Xue, Vimukthini Pinto, Chathura Gamage, Peng Zhang, and Jochen Renz. Hi-Phy: A Benchmark for Hierarchical Physical Reasoning, August 2021. URL <http://arxiv.org/abs/2106.09692>. arXiv:2106.09692 [cs].

Chenyu Zhang, Daniil Cherniavskii, Antonios Tragoudaras, Antonios Vozikis, Thijmen Nijdam, Derck W. E. Prinzhorn, Mark Bodracska, Nicu Sebe, Andrii Zadaianchuk, and Efstratios Gavves. Morpheus: Benchmarking Physical Reasoning of Video Generative Models with Real Physical Experiments, October 2025. URL <http://arxiv.org/abs/2504.02918>. arXiv:2504.02918 [cs].

Pengyu Zhao, Jinan Xu, Ning Cheng, Huiqi Hu, Xue Zhang, Xiuwen Xu, Zijian Jin, Fandong Meng, Jie Zhou, and Wenjuan Han. Bridging the reality gap: A benchmark for physical reasoning in general world models with various physical phenomena beyond mechanics. *Expert Systems with Applications*, 270:126548, April 2025. ISSN 0957-4174. doi: 10.1016/j.eswa.2025.126548. URL <https://www.sciencedirect.com/science/article/pii/S0957417425001708>.

A APPENDIX

A.1 GAME INSTRUCTIONS

The following block contains the game-play instructions we provide to all VLMs. In particular, we explicitly mention the presence of physical principles such as gravity, friction, and elasticity. If the VLM succeeds, the game is over. However, if it fails, we provide a screenshot of what happened. An example screenshot is shown in Figure 4.

A.2 TRAJECTORIES

Whenever there are unsuccessful attempts, we record them and present them as an image. All objects that did move are first presented as opaque versions of themselves, followed by an opaque line with the final resting position presented as the object fully colored in. An example of GPT-5's second attempt at the Spiky level is shown in Figure 4.

A.3 VISUALIZING ATTEMPTS

We design heatmaps to help us better visualize the attempts of VLMs. Two examples are provided in Figure 5 and Figure 6. The full examples are provided in Figure 10 and Figure 11 where each heatmap represents a game level showcasing where each VLM selected which specific object at what specific position.

```

You are playing a physics puzzle game. Your goal is to get the
red ball into the green goal container.

The rules of the game:
- Get any red object into any green area to win the level
- Solve the level with a single action (any tool, any position)
- Solve the level in as few attempts as possible

WORLD SIZE: {world_width}x{world_height} pixels
COORDINATE SYSTEM: Origin at bottom-left, Y increases upward

The available tools will be shown in the screenshot on the right.
There are always 3 tools available.
Listed from top to bottom, they are: {tool_names}

Recall that this is a realistic physics simulation, so:
- Gravity pulls objects downward
- Objects have friction and elasticity
- Placed tools are physical objects that interact with other
objects
- The ball will roll, slide, and bounce

CONSTRAINTS:
- Tool cannot overlap existing objects
- Ball must stay in goal for 2 seconds to win
- You have {max_attempts} total attempts (this is attempt
{attempt_num})
{previous_attempts}{result_screenshot_info}
{level_json_section}
Respond with ONLY a JSON object in this exact format (no
markdown, no extra text):
{{
  reasoning: Brief explanation of strategy,
  tool: obj1,
  x: 250.5,
  y: 300.0
}}

Choose coordinates carefully based on the images. Remember Y
increases upward from bottom-left origin.

```

Figure 3: Gameplay instructions used in all prompts.

A.4 FAILURE TO CONVERT A PLAN INTO ACTION

In this setting (shown in Figure 7), Gemini 3 Pro has access to few-shot examples and the full game state in JSON. On its third attempt, Gemini correctly identifies the second object (highlighted in yellow) as the object to use. Based on its past failures, it also identifies the correct plan, *"Its shape (an asymmetric "T" or hook) creates a natural pivot point on its longer left leg, allowing it to swing clockwise like a hammer"*. It also correctly identifies that drop height matters, *"This is high enough to generate speed but low enough to maintain a stable pivot upon landing on the table."* The full reasoning trace is shown below. Despite the nearly perfect plan, the execution is poor. Firstly, the necessary rotation is incorrect. As correctly noted, the hook must be dropped from a height that is neither too high nor too low, but the right edge of the hook must hit the floating obstacle so that a *counter-clockwise* rotation is imparted which would cause the left side of the hook to strike the ball and send it to the right.

A.5 SEMANTIC ANALYSIS

We categorize each level into five semantic concepts: levers, drops, bridging, chaining, and blocking. We observe that failure rates vary substantially by the semantic concept of the level. Levers show

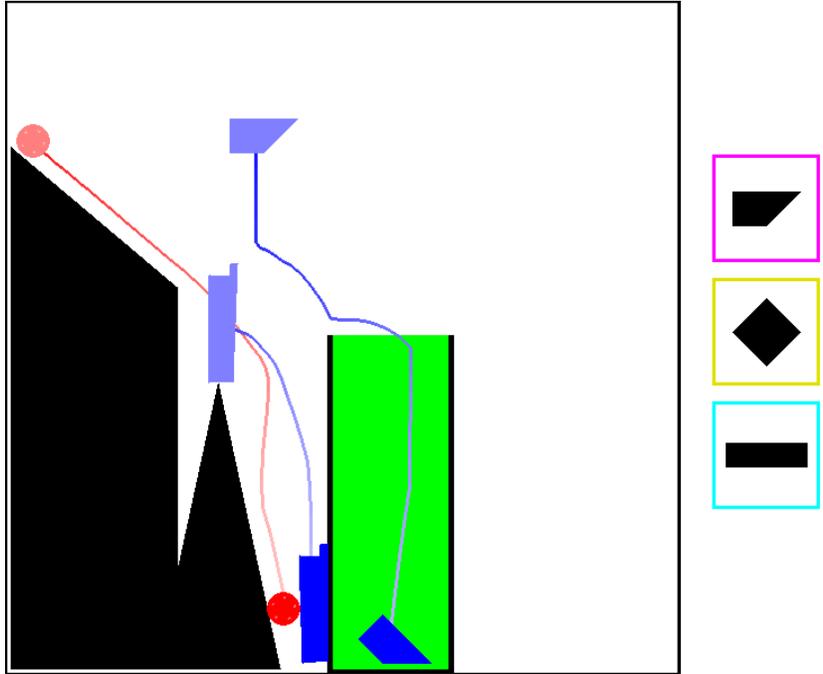


Figure 4: The Spiky Level. In this attempt, the top object was selected and dropped at a height similar to the red ball. When the game commenced, the ball and object fell resulting in the object hitting the teetering block which then toppled over and fell with the ball outside of the goal area. The paths of the objects are shown through their colored lines, their initial positions in opaque color, and their final positions in solid color.

Table 3: Mapping of game levels to the five semantic concepts.

Concept	Levels
levers	Catapult, Launch_A, Launch_B, Launch_v2, SeeSaw
drops	Falling_A, Falling_B, Falling_v2, Trap
blocking	Prevention_A, Prevention_B, Remove, Shove, Spiky
bridging	Basic, Basic_v2, Balance, Bridge, Collapse, Gap, Table_A, Table_B, Table_v2
chaining	Chaining, Shafts_A, Shafts_B, Unbox, Unsupport

the highest failure rate (83.8%), followed by drops (catch/guide) at 80.0%; chaining is the easiest (73.0%), with blocking (including hazard-avoidance levels such as Spiky) at 74.7% and bridging at 73.3%. Across methods, the Few-Shot + JSON method achieves the lowest overall failure rate (61.9%), while Zero-Shot and Few Shot perform worst (around 84%); Few-Shot + JSON method yields the largest gains on drops, bridging, and chaining. Across models, Gemini 3 Pro has the lowest overall failure rate (44.1%), Qwen3-VL the highest (96.4%), and Claude 90.2% (over 112 runs).

A.6 PER-LEVEL COMPARISON: HUMANS VS. VLMS (ALL CONDITIONS)

Tables 5–8 present per-level breakdowns comparing human success rates (limited to 4 attempts, matching the VLM setting) against VLM pass/fail outcomes under all four prompting conditions. Human percentages reflect the fraction of participants who solved each level within their first 4 placements. Human rates are identical across conditions since humans do not receive different prompting treatments.

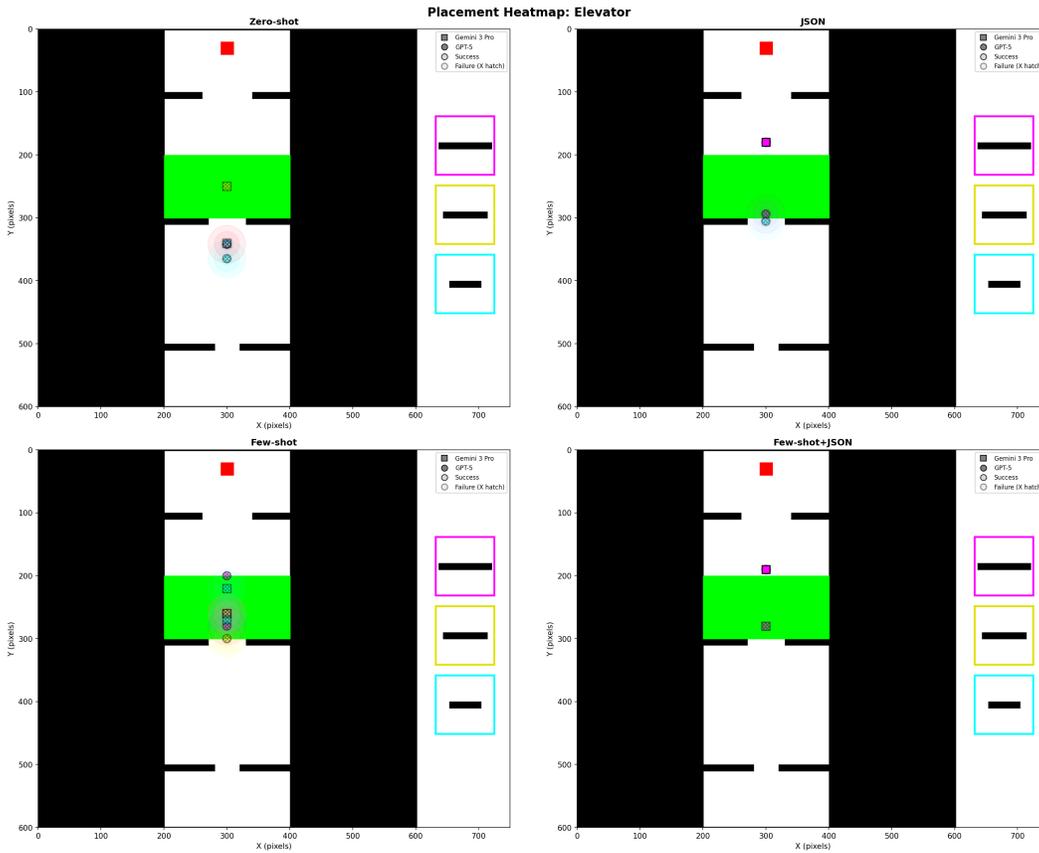


Figure 5: The Elevator Level. Without a JSON representation of game state, VLMs are more likely to repeatedly select the same kind of object and not explore different objects despite being presented with their past failure.

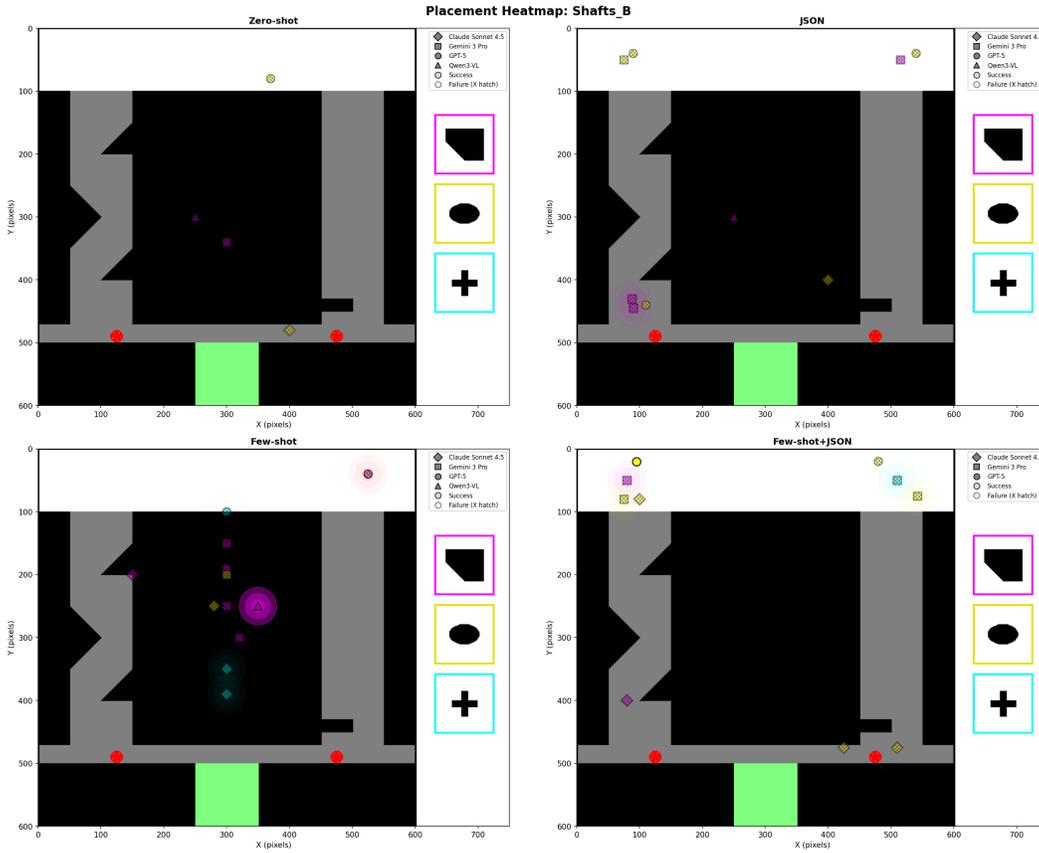


Figure 6: The Shafts B Level. Without a JSON representation of game state, VLMs are more likely to repeatedly select illegal positions (the black area).

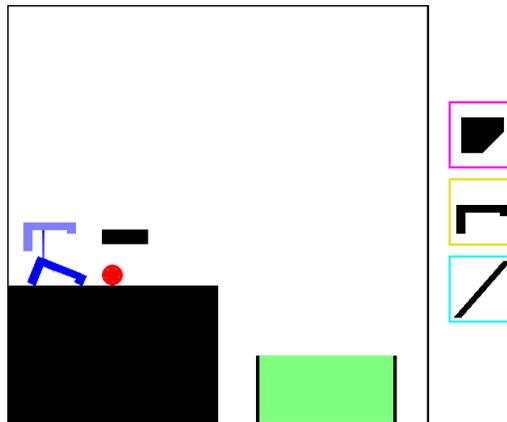


Figure 7: The Launch B Level. Gemini 3 Pro fails to solve this level in the full 4 attempts in all 4 experiments.

Table 4: Failure rate by level concept (5 concepts, 4 models).

Concept	Fail%	Fail / Total Attempts	N levels
levers	83.8	62/74	5
drops	80.0	48/60	4
blocking	74.7	56/75	5
bridging	73.3	99/135	9
chaining	73.0	54/74	5

Based on the analysis of previous attempts and the available tools, the strategy is to use `obj2` (the L-shaped/hook tool) as a swinging hammer.

`**Reasoning:**`

- `**Tool Choice**`: The stick (`obj3`) used in previous attempts successfully hit the ball but failed to impart enough velocity to clear the gap into the goal. `obj2` has a larger surface area (approx 1160 vs 800) and thus likely more mass, which should result in a more powerful impact. Its shape (an asymmetric T or hook) creates a natural pivot point on its longer left leg, allowing it to swing clockwise like a hammer.
- `**Placement (X)**`: We want the hammer head (the top-right part of `obj2`) to swing down and hit the ball. The pivot point will be the bottom of the left leg. To maximize reach while avoiding the Blocker (left edge at `x=135`), we place the tool as far right as possible. `obj2` has a half-width of 37. Placing it at `x=96` puts the right edge at 133, leaving a small safety margin (2px) from the Blocker. This placement sets the pivot at $96 - 37 = 59$. The swing radius to the tip is ≈ 84 . Max reach is $59 + 84 = 143$, which overlaps the ball (left edge 135) by roughly 8 pixels, ensuring a solid hit.
- `**Placement (Y)**`: A drop is needed to generate rotational velocity. Dropping from `y=320` provides a fall of roughly 90-120 pixels (depending on part). This is high enough to generate speed but low enough to maintain a stable pivot upon landing on the table.

The tool will fall, its left leg will hit the table, and the center of mass (to the right of the leg) will cause it to rotate clockwise, swinging the heavy top section into the back-left of the ball, propelling it forward off the table and into the goal.

```
{
  reasoning: Use obj2 as a heavier hammer. Placed at x=96, the left
    leg acts as a pivot, swinging the top bar into the ball. The mass
    is greater than the stick, providing more impulse to clear the
    pit.,
  tool: obj2,
  x: 96.0,
  y: 320.0
}
```

Figure 8: Output of Gemini 3 Pro after several attempts on the Launch B Level (Figure 7).

```

{
  world: {
    dims: [600, 600],
    bts: 0.01,
    gravity: 200,
    defaults: {density: 1, friction: 0.5, elasticity: 0.5, color:
black, bk_color: white},
    objects: {
      _LeftWall: {type: Poly, color: black, density: 0, vertices:
[[-1,-1],[-1,601],[1,601],[1,-1]]},
      _BottomWall: {type: Poly, color: black, density: 0, vertices:
[[-1,-1],[-1,1],[601,1],[601,-1]]},
      _RightWall: {type: Poly, color: black, density: 0, vertices:
[[599,-1],[599,601],[601,601],[601,-1]]},
      _TopWall: {type: Poly, color: black, density: 0, vertices:
[[-1,599],[-1,601],[601,601],[601,599]]},
      Slope: {type: Poly, color: black, density: 0, vertices:
[[0,0],[0,250],[200,120],[200,0]]},
      SeeSaw: {type: Poly, color: blue, density: 1, vertices:
[[220,90],[220,110],[420,110],[420,90]]},
      Goal: {type: Container, color: black, density: 0,
points: [[440,80],[440,2],[540,2],[540,80]], width: 10,
innerColor: green, outerColor: black},
      Ball: {type: Ball, color: red, density: 1, position:
[30,270], radius: 20}
    },
    blocks: {},
    constraints: {},
    gcond: {type: SpecificInGoal, goal: Goal, obj: Ball, duration: 1}
  },
  tools: {
    obj1: [[[-10,-40],[-10,40],[10,40],[10,-40]]],
    obj2: [[[-10,-30],[-10,30],[10,30],[10,-30]]],
    obj3: [[[-40,-20],[-40,20],[40,20],[40,-20]]]
  },
  toolNames: [obj1, obj2, obj3],
  sucText: Get the red ball into the goal
}

```

Figure 9: JSON representing the full initial game state of the SeeSaw level.

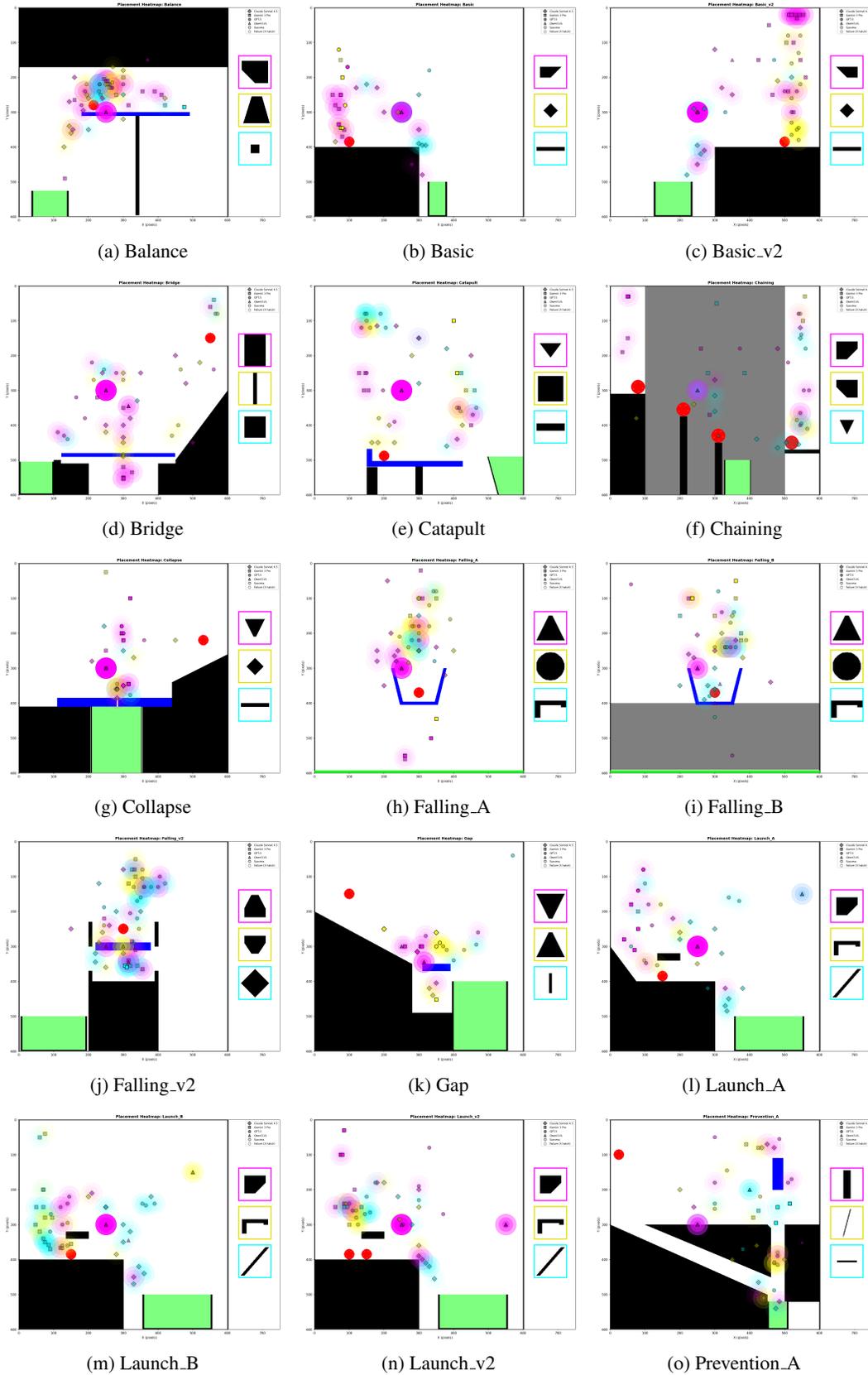


Figure 10: Placement figures for all game levels (page 1 of 2). Each panel shows the initial level with overlaid placement attempts by model (shape) and tool (color).

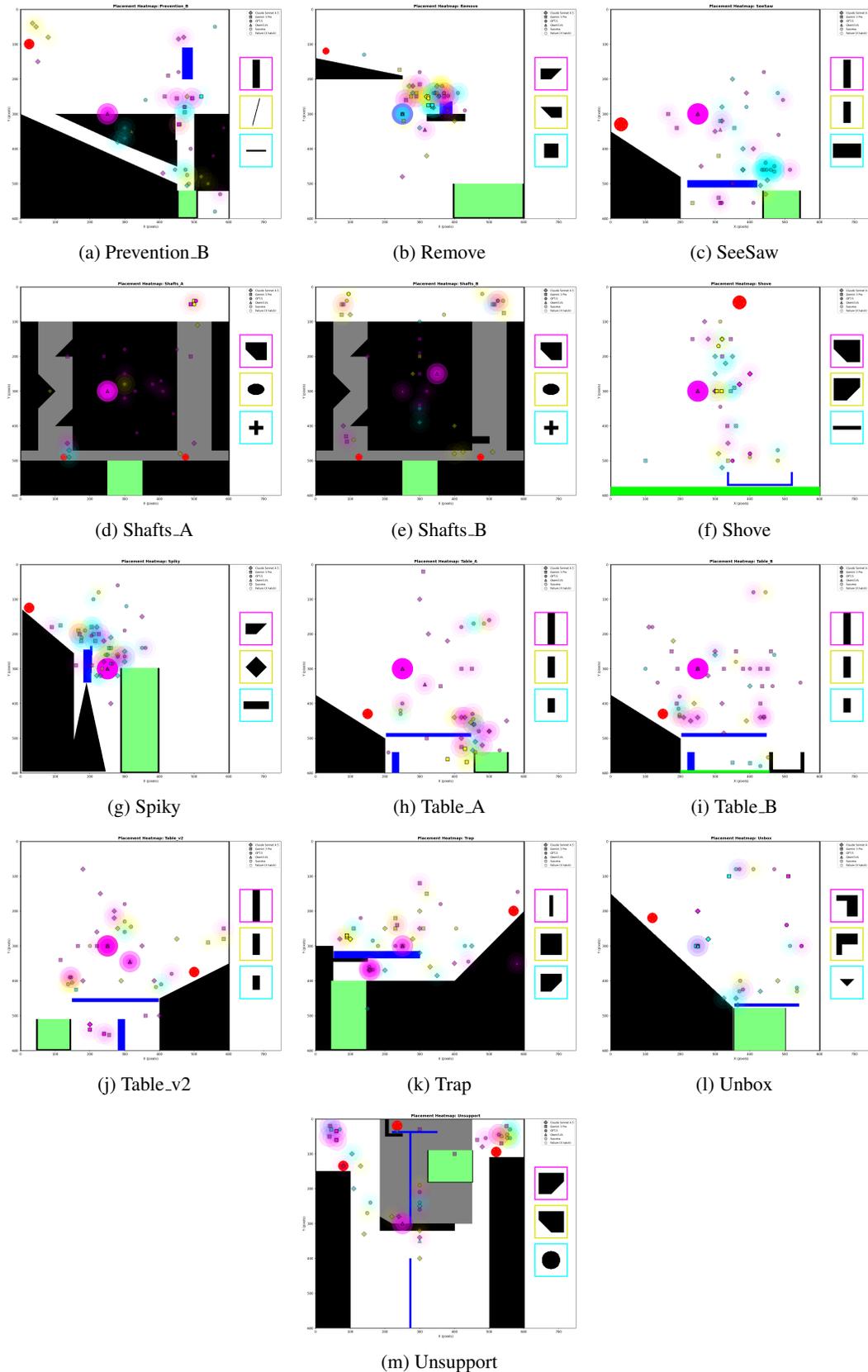


Figure 11: Placement figures (page 2 of 2). Continued from Figure 10.

Table 5: Per-level comparison for the **Zero-Shot** condition. Human% shows the percentage of participants who solved the level within 4 attempts. VLM columns show ✓ (pass) or × (fail). Human pass rates are based on $N=94$ participants for the 18 original levels and $N=50$ for the 10 validation levels. Among the original levels, 6 form matched A/B pairs (e.g., Falling_A vs. Falling_B); each participant played all 12 unmatched levels plus one randomly assigned variant from each pair, yielding $N=46-48$ per matched-pair variant.

Level	Human% (4 att.)	GPT-5	Qwen3-VL	Gemini 3 Pro	Claude 4.5 Son.
<i>Original Levels</i>					
Basic	91.5%	✓	×	✓	×
Bridge	69.1%	×	×	×	×
Catapult	69.1%	×	×	×	×
Chaining	62.8%	×	×	×	×
Falling_A	47.8%	×	×	×	×
Falling_B	31.2%	×	×	✓	×
Gap	85.1%	×	×	✓	✓
Launch_A	93.5%	✓	×	✓	×
Launch_B	20.8%	×	×	×	×
Prevention_A	82.6%	×	×	×	×
Prevention_B	43.8%	×	×	×	×
SeeSaw	37.2%	×	×	×	×
Shafts_A	97.9%	×	×	×	×
Shafts_B	84.8%	×	×	×	×
Table_A	65.2%	×	×	×	×
Table_B	8.3%	×	×	×	×
Unbox	94.7%	×	✓	✓	✓
Unsupport	62.8%	×	×	×	×
<i>Validation Levels</i>					
Balance	26.0%	×	×	×	×
Basic_v2	36.0%	×	×	✓	×
Collapse	94.0%	✓	×	✓	×
Falling_v2	62.0%	×	×	×	×
Launch_v2	50.0%	×	×	✓	×
Remove	78.0%	✓	×	×	✓
Shove	82.0%	✓	×	×	✓
Spiky	4.0%	×	×	×	×
Table_v2	62.0%	×	×	×	×
Trap	78.0%	×	×	×	×
Total	—	5/28	1/28	8/28	4/28

Table 6: Per-level comparison for the **Zero-Shot + JSON** condition. Human% shows the percentage of participants who solved the level within 4 attempts. VLM columns show ✓ (pass) or × (fail). Human pass rates are based on $N=94$ participants for the 18 original levels and $N=50$ for the 10 validation levels. Among the original levels, 6 form matched A/B pairs (e.g., Falling_A vs. Falling_B); each participant played all 12 unmatched levels plus one randomly assigned variant from each pair, yielding $N=46-48$ per matched-pair variant.

Level	Human% (4 att.)	GPT-5	Qwen3-VL	Gemini 3 Pro	Claude 4.5 Son.
<i>Original Levels</i>					
Basic	91.5%	✓	×	✓	×
Bridge	69.1%	×	×	✓	×
Catapult	69.1%	×	×	✓	×
Chaining	62.8%	×	×	✓	×
Falling_A	47.8%	×	×	✓	×
Falling_B	31.2%	×	×	✓	×
Gap	85.1%	✓	×	✓	×
Launch_A	93.5%	✓	×	✓	×
Launch_B	20.8%	×	×	×	×
Prevention_A	82.6%	×	×	✓	×
Prevention_B	43.8%	×	×	✓	×
SeeSaw	37.2%	×	×	×	×
Shafts_A	97.9%	✓	×	✓	×
Shafts_B	84.8%	×	×	×	×
Table_A	65.2%	×	×	✓	×
Table_B	8.3%	×	×	×	×
Unbox	94.7%	✓	✓	✓	×
Unsupport	62.8%	×	×	✓	×
<i>Validation Levels</i>					
Balance	26.0%	×	×	×	×
Basic_v2	36.0%	×	×	✓	×
Collapse	94.0%	×	×	✓	×
Falling_v2	62.0%	×	×	✓	×
Launch_v2	50.0%	×	×	✓	×
Remove	78.0%	✓	×	✓	×
Shove	82.0%	✓	×	✓	×
Spiky	4.0%	×	×	×	×
Table_v2	62.0%	×	×	✓	✓
Trap	78.0%	✓	×	✓	×
Total	—	8/28	1/28	22/28	1/28

Table 7: Per-level comparison for the **Few-Shot** condition. Human% shows the percentage of participants who solved the level within 4 attempts. VLM columns show ✓ (pass) or × (fail). Human pass rates are based on $N=94$ participants for the 18 original levels and $N=50$ for the 10 validation levels. Among the original levels, 6 form matched A/B pairs (e.g., Falling_A vs. Falling_B); each participant played all 12 unmatched levels plus one randomly assigned variant from each pair, yielding $N=46-48$ per matched-pair variant.

Level	Human% (4 att.)	GPT-5	Qwen3-VL	Gemini 3 Pro	Claude 4.5 Son.
<i>Original Levels</i>					
Basic	91.5%	✓	×	✓	×
Bridge	69.1%	×	×	×	×
Catapult	69.1%	×	×	×	×
Chaining	62.8%	×	×	×	×
Falling_A	47.8%	×	×	✓	×
Falling_B	31.2%	×	×	×	×
Gap	85.1%	×	×	✓	✓
Launch_A	93.5%	×	×	✓	×
Launch_B	20.8%	×	×	×	×
Prevention_A	82.6%	×	×	✓	×
Prevention_B	43.8%	×	×	✓	×
SeeSaw	37.2%	×	×	×	×
Shafts_A	97.9%	×	×	×	×
Shafts_B	84.8%	×	×	×	×
Table_A	65.2%	×	×	✓	×
Table_B	8.3%	×	×	×	×
Unbox	94.7%	✓	✓	✓	✓
Unsupport	62.8%	×	×	×	×
<i>Validation Levels</i>					
Balance	26.0%	×	×	×	×
Basic_v2	36.0%	×	×	×	×
Collapse	94.0%	✓	×	✓	×
Falling_v2	62.0%	×	×	×	×
Launch_v2	50.0%	×	×	×	×
Remove	78.0%	×	×	×	×
Shove	82.0%	✓	×	×	✓
Spiky	4.0%	×	×	×	×
Table_v2	62.0%	×	×	×	×
Trap	78.0%	×	×	×	×
Total	—	4/28	1/28	9/28	3/28

Table 8: Per-level comparison for the **Few-Shot + JSON** condition. Human% shows the percentage of participants who solved the level within 4 attempts. VLM columns show ✓ (pass) or × (fail). Human pass rates are based on $N=94$ participants for the 18 original levels and $N=50$ for the 10 validation levels. Among the original levels, 6 form matched A/B pairs (e.g., Falling_A vs. Falling_B); each participant played all 12 unmatched levels plus one randomly assigned variant from each pair, yielding $N=46-48$ per matched-pair variant.

Level	Human% (4 att.)	GPT-5	Qwen3-VL	Gemini 3 Pro	Claude 4.5 Son.
<i>Original Levels</i>					
Basic	91.5%	✓	×	✓	×
Bridge	69.1%	×	×	✓	×
Catapult	69.1%	×	×	✓	×
Chaining	62.8%	×	×	✓	×
Falling_A	47.8%	×	×	✓	×
Falling_B	31.2%	×	×	✓	×
Gap	85.1%	✓	×	✓	✓
Launch_A	93.5%	×	×	✓	×
Launch_B	20.8%	×	×	×	×
Prevention_A	82.6%	×	×	✓	×
Prevention_B	43.8%	×	×	✓	×
SeeSaw	37.2%	×	×	✓	×
Shafts_A	97.9%	✓	×	✓	×
Shafts_B	84.8%	✓	×	×	×
Table_A	65.2%	×	×	✓	×
Table_B	8.3%	×	×	×	×
Unbox	94.7%	×	×	✓	×
Unsupport	62.8%	×	×	✓	×
<i>Validation Levels</i>					
Balance	26.0%	×	×	✓	×
Basic_v2	36.0%	×	×	×	×
Collapse	94.0%	✓	×	✓	✓
Falling_v2	62.0%	×	×	✓	×
Launch_v2	50.0%	×	×	✓	×
Remove	78.0%	×	×	✓	×
Shove	82.0%	✓	×	✓	×
Spiky	4.0%	×	×	×	×
Table_v2	62.0%	×	×	✓	×
Trap	78.0%	×	×	✓	✓
Total	—	6/28	0/28	23/28	3/28