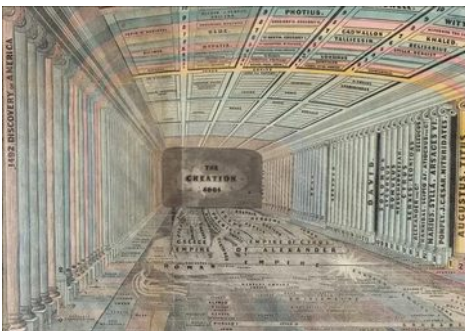


AI That Uses Imagery in Chain-of-Thought (CoT) Reasoning

Originally published September 28, 2026 at <https://www.unite.ai/ai-that-uses-imagery-in-chain-of-thought-cot-reasoning/>



We think in pictures a lot – well, most of us: in humans, diminished capacity for visual imagination has been [shown in research](#) to be associated with lower academic performance. In terms of memorization (our need to remember things, not [the AI bugbear](#)), the considerable semantic power of strong or vivid imagery has been [used for millennia](#) as an aide to learning:



Emma Willard's "Temple of Time" (1846), an educational visualization that turns chronology into physical space, arranging historical periods and figures within a receding architectural perspective. Willard designed such images as visual mnemonics, believing graphic representations could help students form a coherent mental picture of history. [Source](#)

However, the field of mnemonics concerns data *ingestion* rather than processing or interpretation, where people vary considerably in the 'styles' of thinking that they draw on.

Personally, I [think in shapes](#), and have done for as long as I have been thinking – with decades, years, ideas and people represented in some way by relative geometry and dynamic blocks of volume. Others think primarily [in numbers](#); others still in [smells or taste](#).

For AI, the possible range of synesthesia methods is more limited; naturally, a Large Language Model (LLM) can think in text, since this is its native encoding type, above the level of [embeddings](#); and LLMs have been shown to encode numbers [as concepts](#) rather than pure variable values, demonstrating a propensity to 'think in numbers'.

But to what extent can an LLM be said to be 'thinking in shapes', or 'thinking in images', in the way humans tend to?

The fact that one [gets different responses](#) from an LLM to the same question presented in different languages signifies that those relationships can be very specific and brittle, and [hard-coded to specific text](#) in a language domain (i.e., 'Spanish'), rather than addressing a higher-level domain that transcends and yet contains language*.

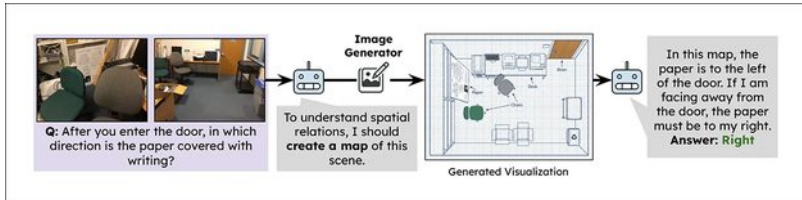
Therefore a multimodal LLM (i.e., a Vision Language Model, or VLM) that can generate images *purely for its own internal Chain of Thought (COT)* processes might logically have an advantage (or at least a literally alternate point of view).

Novel Viewpoints

With this in mind, a recent research collaboration from Germany has put forward an image-focused VLM called *RelmaGin*, which leverages image generation as a *ductile reasoning mechanism*.

Though prior work has 'bolted-on' image-based components, these facilities were neither intrinsic nor essential to the core workflow; conversely, the authors' new system is designed to exploit the very recent capabilities of VLMs to usurp and take over the functionality of [specialized tools](#) and methods, [such as depth perception](#).

In the example below, from the [new paper](#) – titled *Reasoning with Image Generation* – the AI must interpret two views (leftmost images), neither of which contain all the information necessary to solve a task. Therefore the model is able to use the information it does have to interpret a wider and more complete picture of the scene (the map with the sub-caption 'Generated Visualization', third from left below).



From the new paper, an example of *RelmaGin* generating a top-down map from two incomplete views, giving the model a unified visual representation from which

RelmaGin is not tied to one particular kind of visual transformation, but rather can complete missing puzzle regions; remove occlusions for counting; draw trajectories for collision prediction; combine multiple views into spatial maps; convert dashed paths into solid lines for tracing; and generate depth maps for depth reasoning.

More importantly, the system can regulate its own use of this internal tool-set, in line with recent emergent abilities of VLMs to automatically address particular challenges with apposite tools, such as [depth estimation](#). Most of *RelmaGin*'s functionality as described is summoned up by calls to `generate_image` tool, a function that can intervene visually as necessary, and without human supervision or invocation.

The authors state:

'Because generate_image accepts arbitrary natural-language instructions, the agent can perform a vast range of transformations; the question is which transformation actually helps for a given task.'

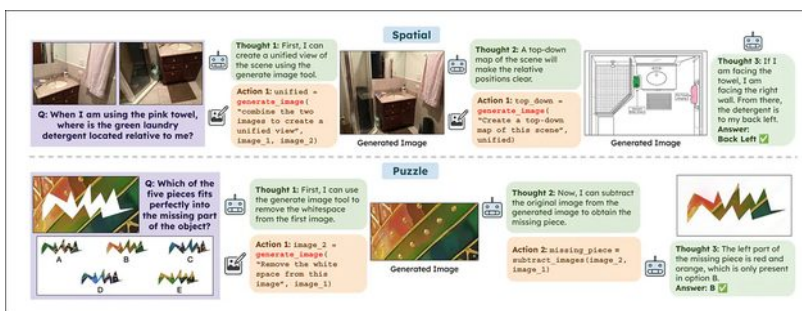
'[Standard] practice is to specify the transformation through handcrafted in-context examples that demonstrate the desired strategy (e.g., generate a depth map for depth reasoning). This requires manual effort per task and limits generalization to new tasks.'

'[We] ask whether such strategies can be discovered automatically. Since the strategy is conveyed to the agent through its prompt, discovering a strategy reduces to optimizing the prompt: we instantiate an iterative loop in which a proposal model generates candidate prompts, evaluates them on a small development set, and refines based on observed successes and failures.'

Tested across three VLMs and six visual reasoning tasks, *RelmaGin* broadly outperformed both unaided reasoning and specialist vision tools, with the use of a single tool.

The Approach

RelmaGin operates as a loop: at each step, the VLM considers the question, the original images, and anything it has already generated, and decides what to do next. This can include ordinary image operations such as cropping or overlaying, or a call to `generate_image`, which creates a new image intended specifically to make the problem easier to reason about:



A step-by-step illustration of *RelmaGin* reasoning through spatial and visual-puzzle problems. In both examples, the model generates a new image during reasoning additional input for subsequent steps toward the answer.

The generated image is then fed back to the VLM, becoming part of the material available for the next reasoning step, and the process can repeat. In the image above we can see these facets illustrated for the spatial task: the model first combines two photographs into a single view, then turns that result into a top-down map before answering the question.

For the puzzle task, it generates an altered version of the puzzle that isolates the missing region, then uses conventional image subtraction to identify the answer.

In this way, image generation becomes part of the reasoning process itself, rather than an end product for the user; indeed, these interstitial images are intended solely for the CoT processes of the AI, and not for the direct consumption of the end-user.

At each turn, the VLM chooses its own next action from the context accumulated so far. That action can be another reasoning step; a conventional image operation; or a natural-language instruction to generate_image. Whatever the chosen tool returns is added to the context, allowing the model to inspect the result and decide whether to transform it again; use another tool; or proceed to its final answer.

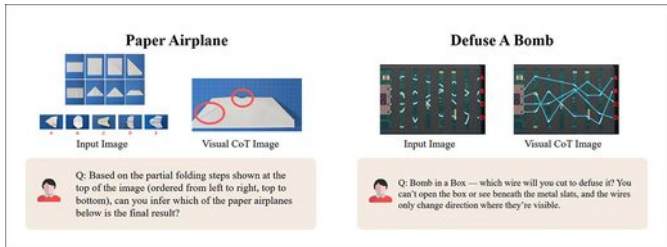
Gaining Agency

A central problem is deciding what *kind* of image would actually make a particular task easier. RelmaGin decides this by experimenting with different instructions to the agent, testing candidate prompts on examples with known answers, and using successful and failed attempts to obtain better prompts. Further iterations of this loop eventually produce the best-performing strategies.

The reasoning model and image generator themselves are not retrained during this process; only the instructions governing how the agent uses its tools are optimized. The researchers therefore describe the process as ‘automated discovery’ of visual reasoning strategies, without providing task-specific strategies or reasoning examples themselves.

Configuration and Tests

RelmaGin was evaluated across six visual reasoning tasks: *depth reasoning* ([Blink](#) – identifying which of two points is closer to the camera); *puzzle completion* ([Mira](#) – selecting the correct piece for a missing image region); *occlusion counting* ([CAPTURE](#) – counting objects including hidden ones); *collision prediction* ([Spatial457](#) – predicting which object a moving target will hit); *spatial reasoning* ([MMSI](#) – determining relationships between objects across different views); and *path tracing* – a new benchmark requiring models to follow dashed lines connecting numbers to letters.



From the paper ‘MIRA, a Benchmark for Visual Chain-of-Thought’, striking examples of visual imagery in chain-of-thought processes. [Source](#)

The VLM backbones tested were [Gemini-3.1-Pro](#); [GPT-5](#); and the open-weights [Qwen-3.5-27B](#). Image generation was handled primarily by [Nano-Banana-Pro](#), with the open-weights [FLUX.2 \[dev\]](#) and [Qwen-Image-Edit-2511](#) also tested. Gemini-3.1-Pro was used as the selector for image-generation test-time scaling.

Of the two baselines used for comparison, the ‘vanilla’ VLM dubbed by the paper’s authors ‘No Tools’ answered directly from the query and images, without tools or code execution, while the 2024 outing [Visual Sketchpad](#) offered in this case a fixed set of specialist vision and image-manipulation tools – but no open-ended image generation.

For the MMSI spatial task, both baselines were given a similar amount of computing power to RelmaGin, by generating 20 answers from each baseline, and using the answer that appeared most often.

Performance was measured by multiple-choice accuracy for all tasks except occlusion counting, which was assessed using [symmetric mean absolute percentage error](#), comparing predicted and actual object counts. Results were averaged across three runs, with [standard error](#) also reported.

Model	Approach	Puzzle ↑ (MIRA)	Occ. Count ↓ (CAPTURE)	Collision ↑ (Spatial457)	Spatial ↑ (MMSI)	Tracing ↑ (Ours)	Depth ↑ (BLINK)
Gemini-3.1-Pro	No Tools	37.2±2.6	8.9±0.4	60.3±0.9	52.0±0.0	71.5±1.2	91.7±0.3
	Sketchpad	38.5±8.9	9.7±0.2	65.0±0.8	57.0±0.7	74.7±1.9	99.2±0.0
	REIMAGIN	42.3±2.2	7.1±0.6	69.7±1.2	59.0±0.6	89.0±1.6	94.6±1.5
GPT-5	No Tools	29.5±3.4	12.5±0.6	44.7±1.9	36.0±1.5	48.5±1.2	83.9±1.6
	Sketchpad	16.7±1.3	11.7±0.4	51.0±1.0	37.7±0.3	41.7±1.7	88.7±0.7
	REIMAGIN	44.9±5.6	7.4±0.1	54.0±1.5	43.7±1.3	73.5±0.4	89.2±1.3
Qwen-3.5-27B	No Tools	34.6±2.2	12.2±0.3	51.8±1.5	45.0±1.2	35.7±1.2	83.5±0.3
	Sketchpad	25.6±2.6	8.7±0.0	48.0±1.5	46.7±0.3	32.5±2.4	91.1±0.0
	REIMAGIN	46.2±2.2	6.8±0.6	61.3±2.3	54.3±0.9	78.5±0.4	88.3±0.3

Test results comparing RelmaGin with No Tools and Visual Sketchpad across three VLMs and six visual reasoning tasks. Higher scores are better except for Occl is better. RelmaGin achieved the best result in the majority of model-task combinations.

The same human-defined strategy examples were used across all three models. RelmaGin produced better results than both baselines on most tasks, with particularly clear gains in puzzle completion, occlusion counting and path tracing.

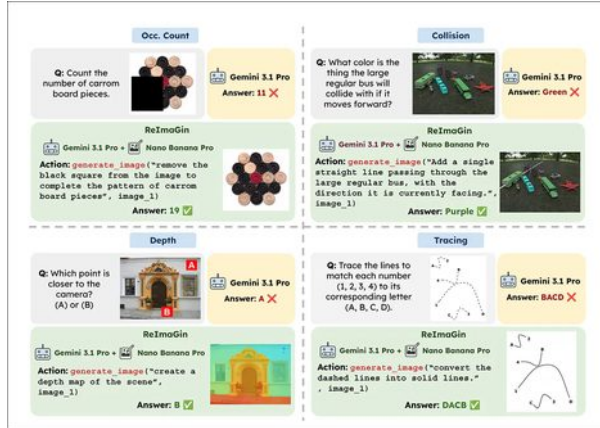
With GPT-5, for example, puzzle accuracy increased from 29.5% with No Tools and 16.7% with Visual Sketchpad, to 44.9% with RelmaGin. Ablation tests confirmed that the generative image component is essential to the system’s performance.

Open-weights image generators were also tested in place of Nano-Banana-Pro: FLUX.2 [dev] and Qwen-Image-Edit-2511 produced comparable results on some simpler tasks, particularly [inpainting](#), but were less consistent on more demanding transformations such as depth estimation and path tracing. Similar results were obtained when Qwen-3.5-27B was used as the VLM:

Image Gen. Model	Puzzle ↑ (MIRA)	Occ. Count ↓ (CAPTURE)	Collision ↑ (Spatial457)	Spatial ↑ (MMSI)	Tracing ↑ (Ours)	Depth ↑ (BLINK)
No Tools	34.6	12.2	51.8	45.0	35.7	83.5
Nano-Banana-Pro	46.2	6.8	61.3	54.3	78.5	88.3
FLUX.2 [dev]	44.2	6.1	45.5	38.0	66.0	85.5
Qwen-Image-Edit-2511	32.1	8.8	50.0	42.3	48.0	80.3

Test results comparing image generators with Qwen-3.5-27B as the VLM. Nano-Banana-Pro achieved the best result on five of six tasks, while FLUX.2 [dev] and (gains over No Tools on several tasks.

The authors also carried out qualitative tests on four tasks:



Qualitative examples across four tasks show how RelmaGin was used to augment Gemini-3.1-Pro's reasoning. In each case, generated visual representations were incorporated into the reasoning process to help solve the task.

RelmaGin was not reliable in every case: in some circumstances, an inaccurate transformation was produced by the image generator, such as a floor-plan with objects in the wrong positions; in others, an accurate generated image was subsequently misread by the VLM.

In a manual audit of 120 generated images, 75% were found to have faithfully performed the requested transformation. When this was achieved, the final answer was correct 87% of the time, compared with 43% when the generated image was inaccurate.

The authors conclude:

'Our results suggest two broader conclusions. First, image generation can act as a general visual imagination mechanism within multimodal reasoning, reducing the need to engineer a separate tool for each new transformation.'

'Second, the effectiveness of this approach depends not only on the underlying models but also on the reasoning strategy used to decide what to visualize and how to use the result.'

'The gains from automatic strategy discovery show that the models can leverage their understanding of visual transformations to discover relevant strategies without human-authored task-specific strategies or reasoning.'

Conclusion

Self-tooling decisions of the kind investigated in this study are a fascinating development, not least because it appears that VLM development is naturally evolving towards this approach in any case. I am curious to see if such all-purpose VLMs will eventually perform as well as dedicated tools and frameworks such as the [Segment](#) ecostructure; and if therefore those who deal exclusively in these 'side tasks' may end up using a sledgehammer to crack a nut.

It is rather difficult to believe that VLMs could evolve the requisite discipline to overtake and maintain a lead on dedicated solutions like [local fine-tuning](#), where an entire model is given over to one task and [often left useless](#) for any other in the process. Time will tell.

* An arguable definition for the domain of imagery, which we [learn long before](#) we pick up any language skills.

First published Monday, September 28, 2026

Writer on machine learning, domain specialist in human image synthesis. Former head of research content at Metaphysic.ai, until its dissolution into DNEG's Brahma.ai.

Website: martinanderson.ai

Contact: martin@martinanderson.ai