Open **D** ataLab

上 海 人 工 智 能 实 验 室
Shanghai Artificial Intelligence Laboratory

GGBench: A Geometric Generative Reasoning Benchmark for Unified Multimodal Models

Jingxuan Wei[†1], **Caijun Jia**[†1], **Xi Bai**[†1], **Xinglong Xu**[†1], **Siyuan Li**[2], **Linzhuang Sun**[1], **Bihui Yu**[1], **Conghui He**[2], **Lijun Wu**[2], **Cheng Tan**[2]

[1]University of Chinese Academy of Sciences, [2]Shanghai Artificial Intelligence Laboratory

The advent of Unified Multimodal Models (UMMs) signals a paradigm shift in artificial intelligence, moving from passive perception to active, cross-modal generation. Despite their unprecedented ability to synthesize information, a critical gap persists in evaluation: existing benchmarks primarily assess discriminative understanding or unconstrained image generation separately, failing to measure the integrated cognitive process of generative reasoning. To bridge this gap, we propose that geometric construction provides an ideal testbed as it inherently demands a fusion of language comprehension and precise visual generation. We introduce GGBench, a benchmark designed specifically to evaluate geometric generative reasoning. It provides a comprehensive framework for systematically diagnosing a model's ability to not only understand and reason but to actively construct a solution, thereby setting a more rigorous standard for the next generation of intelligent systems.

Date: January 15, 2026
Correspondence: Cheng Tan, tancheng@pjlab.org.cn

○ **Code**   🌐 **Website**   🤗 **Dataset**

Question： Using the digits $\{0, 6, 7, 8, 9\}$, how many two-digit numbers with distinct digits can be formed?

Solution： *A two-digit number cannot begin with 0, so the tens place has 4 admissible choices $\{6, 7, 8, 9\}$. For each choice, the units place can be any of the remaining 4 digits (including 0). Hence the count is $(4 \times 4 = 16)$. Equivalent argument: Total permutations of two distinct digits from five is (P(5,2)=20). Exclude the 4 invalid numbers with leading zero $\left(0x, (x \in \{6, 7, 8, 9\})\right): (20 - 4 = 16)$.*
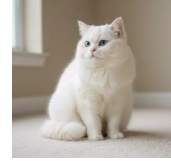***Answer:*** *16.*

(a) Understanding Only

Question： Chubby white cat with blue eyes, sitting, photorealistic, neutral background.

Okay, here's that image for you:
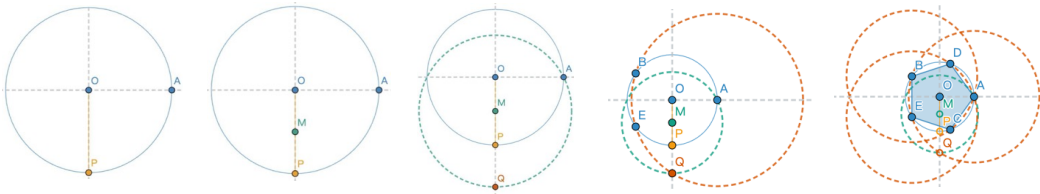
(b) Generation Only

Question： Draw a regular pentagon inscribed in a circle.

Step 1: Construct a radius OP passing through the center O and perpendicular to OA
Construction:
1. Construct the line OA.
2. Construct a line l passing through point O and perpendicular to the line OA.
3. Line l intersects circle c at two points; select either intersection as point P.

⋯

Step 4: With center A and radius AQ, mark points B and E on circle c
Construction:
1. With point A as the center and the length of the line segment AQ as the radius, construct an auxiliary circle c_A.
2. Circle c_A intersects the original circle c at two points. These are two additional vertices of the regular pentagon, B and E.

Step 5: Sequentially mark the remaining vertices C, D and connect to form the regular pentagon
Construction:
1. With B as the center and a radius of AQ, draw a circle. This circle intersects circle c at A and another point, C.
2. With E as the center and a radius of AQ, draw a circle. This circle intersects circle c at A and another point, D.
3. Connect points A, B, C, D, and E sequentially.

(c) Understanding & Generation

Figure 1: The paradigm shift to generative reasoning. Conventional benchmarks evaluate (a) Understanding or (b) Generation in isolation. GGBench introduces (c) integrated Understanding & Generation evaluation, requiring generative reasoning from Unified Multimodal Models.

1

1   Introduction

Artificial intelligence research is undergoing a profound paradigm shift, evolving from the foundational era of passive perception to the more sophisticated stage of contextual understanding [2], and now culminating in the frontier of active generation. This evolutionary leap is epitomized by the recent advent of Unified Multimodal Models (UMMs)[73], such as the groundbreaking GPT-4o[20] and Nano Banana [12], which demonstrate an unprecedented ability to fluently process, reason across, and synthesize information from a diverse spectrum of modalities. These UMMs can not only understand complex visual and textual inputs, but also generate rich outputs; notably, advanced systems like Unified-IO 2 [27] can even produce images, audio, or actions in response to multimodal instructions. Despite these breakthroughs, existing evaluation benchmarks lag behind – they primarily assess discriminative reasoning (e.g. selecting an answer or classifying a visual input) and thus fail to capture the generative dimension of intelligence [60]. In particular, current tests rarely challenge models to construct solutions (e.g. draw a diagram or formulate a step-by-step proof), which is often essential for domains like geometry. This gap is especially evident in geometric problem-solving, where genuine understanding is inseparable from the ability to plan and generate a diagram through multi-step spatial reasoning [55]. Indeed, current models struggle with this constructive aspect – even advanced multimodal systems have difficulty reliably interpreting complex geometric setups [13, 46, 11].

The trajectory of evaluation benchmarks over recent years reflects a push toward increasingly complex cognition, yet also reveals a gap in evaluating **generative reasoning**. Early benchmarks targeted single-modality reasoning or generation, primarily for Large Language Models (LLMs) in purely textual domains. For example, GSM8K [8] and MATH [17] established rigorous tests for mathematical problem-solving in text, requiring models to parse word problems and generate final answers through a step-by-step reasoning chain. Techniques such as chain-of-thought prompting greatly improved performance on these datasets [54]. Building on this progress, the community introduced multimodal reasoning challenges that incorporate visual context. Benchmarks like ScienceQA [28] require models to combine text understanding with image comprehension [45], and the recent MathVista suite [29] amalgamates 28 math and visual datasets to evaluate mathematical reasoning in visual contexts. These tasks, aimed at Multimodal LLMs (MLLMs), represent a critical step toward contextual understanding – models must interpret a question against an image or figure – but they remain largely discriminative. The model is still to answer or classify based on given content, rather than to create new content. The latest multimodal benchmarks have started to probe both understanding and generation capabilities, but often in a decoupled fashion. Efforts like MME [74], MM-Vet [67], and MMBench [26] evaluate a broad range of perceptual and cognitive tasks to score models on each ability individually. Critically, however, even these comprehensive benchmarks treat understanding and generation as separate modules. This fragmentation means we lack evaluation of the integrated cognitive process – the scenario where a system must simultaneously comprehend, reason, and generate a complex outcome.
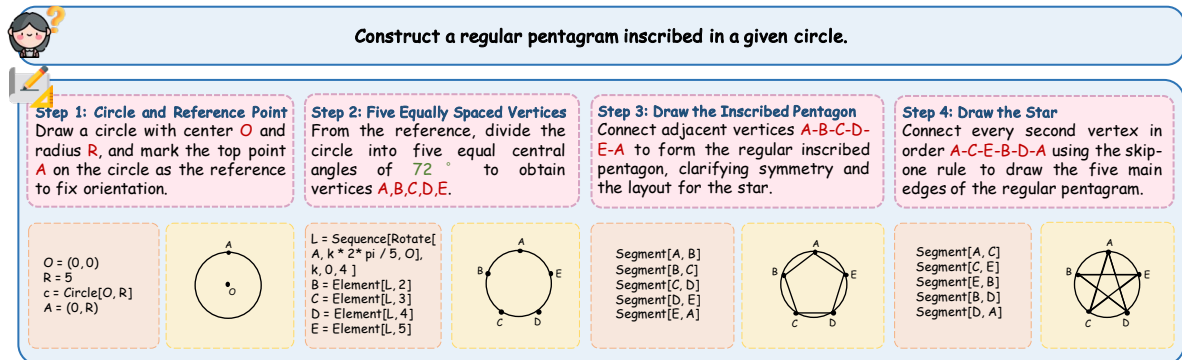

Figure 2: GGBench's step-by-step evaluation. Beyond traditional text-image pairs, GGBench provides executable code for each construction step, allowing for precise and automated verification.

Perform **generative reasoning** demands both cross-modal understanding of abstract concepts and the structured, stepwise generation of a coherent solution. We propose that geometric construction problems provide an ideal testbed for evaluating such integrated understanding–generation capabilities within a unified setting. Such problems inherently require a model to:

- Parse and comprehend abstract natural language instructions with domain-specific constraints;
- Formulate a multi-step plan grounded in formal geometric principles;
- Generating precise figures that satisfies the given constraints.

Performance on geometric construction tasks is directly verifiable, offering a clear and objective criterion for evaluation. Success in such tasks engages the full spectrum of intelligence: language understanding, mathematical reasoning, long-horizon planning, and visual generation. To enable this rigorous evaluation, we introduce GGBench, a comprehensive benchmark where each problem instance consists of precisely aligned text, code, and image modalities, as shown in Figure 2, facilitating a systematic analysis of the entire multimodal generative reasoning process.

2   Related Work

2.1   The Evolution of Mathematical Reasoning Benchmarks

The evaluation of mathematical reasoning in AI has progressed from discriminative question answering toward more generative, process-oriented tasks. Early benchmarks such as GeoQA [5] provides valuable testbeds but narrowly targets plane figures and evaluates models on answer accuracy. Several recent benchmarks like MATH-Vision (MATH-V) [48], MathVista [29], and MathVerse [72] have broadened the scope of evaluation by combining visual and textual information. Specialized datasets like PolyMath [14] evaluates general cognitive reasoning in multimodal settings, requiring models to combine visual pattern recognition with mathematical logic. MathScape [76] uses real-world photographs for math questions. GeoEval [71] compiled a comprehensive suite of geometry question. SolidGeo [51] focuses on solid/3D geometry, whereas VisAidMath [31] evaluates the use of visual aids such as drawing auxiliary lines in geometry. In a different vein, MMSciBench [65] evaluates multimodal scientific problems, covering both math and physics questions in textual and text+image formats. Beyond static images, researchers have explored temporal problem-solving paradigms: Video-MathQA [38] assesses long-horizon mathematical reasoning in educational videos, and NewtonBench [75] challenges LLM-based agents to discover scientific laws through interactive experimentation. MathOPEval [22] requires models to perform fine-grained visual operations by generating executable code like constructing, modifying, or annotating a diagram. DynaMath [78] introduces dynamic variations of visual math problems to test robustness. MaRVL-QA [36] uses unlabeled function plots and geometry transformations as inputs that minimizes textual cues.

Crucially, as researchers identified the limitations of purely result-oriented evaluation, new benchmarks have placed increasing emphasis on the reasoning process itself. MM-MATH [44] exemplifies this shift by augmenting outcome evaluation with process evaluation. We-Math [37] draws inspiration from human problem-solving principles and decomposes problem into sub-problems aligned with knowledge concepts. Math2Visual [47] introduces the task of generating pedagogically useful diagrams for math word problems. A model is given a math problem in text and must produce a diagram or visual illustration that meaningfully represents the problem's content. Most recently, Geoint-R1 [55] integrates auxiliary element construction with formal verification, and MathCanvas [41] enabled a unified multimodal model [10] to learn "when and how" to draw as part of its reasoning process. Despite this rich and rapidly evolving landscape, a critical gap persists. The majority of existing evaluations, even those that are process-oriented, ultimately reduce to either checking a final numerical or categorical answer or scoring a model's generated text against a reference solution. A fundamental challenge remains unaddressed: the lack of a benchmark that holistically evaluates a model's ability to generate a complete, multi-step, and formally verifiable geometric construction from scratch.

2.2 The Rise of Unified Multimodal Models

UMMs signifies a pivotal architectural shift, not merely in the unification of modalities, but more fundamentally in the unification of understanding and generation capabilities within a single framework [73]. This paradigm moves beyond specialized models towards a generalist capable of both interpreting multimodal inputs and generating structured outputs. Recent examples of UMMs include GPT-4o [20] and Gemini 2.5 [9], as well as powerful open-source alternatives such as Janus series [57, 6, 32], Qwen-VL [4, 52], and OmniBridge [58], are all designed to seamlessly transition between comprehension and synthesis. Other efforts include Apple's MM1 [33] and Google's Nano Banana (Gemini 2.5 Flash Image) [12]. There are also hybrid systems like MM-ReAct [64], which augment an LLM with external visual tool interfaces rather than a single unified network. Together, these developments mark a shift from modality-specific AI toward end-to-end multimodal intelligence, with UMMs learning unified representations that support cross-modal understanding and generation.

On the one hand, the understanding capabilities of these models have been rigorously validated across a vast landscape of benchmarks. They have achieved exceptional performance on testbeds like MMMU [68] for expert-level knowledge, MME-Unify [61] for holistic perception and cognition, and MathVista [29] for visual mathematical reasoning. Models explicitly designed for deep comprehension, such as Query-Kontext [43] and MAGUS [21], further demonstrate this proficiency. On the other hand, the generative prowess of UMMs is rapidly advancing, moving far beyond simple image captioning. Models like Bagel [10] and its successor Hyper-Bagel [30] showcase sophisticated generative abilities honed by novel training techniques. Application-specific models like ChartSketcher [19] can synthesize data visualizations from text, while generative models like Lingshu [62], VARGPT [77], and HaploOmni [59] exhibit advanced faculties for creating rich, context-aware multimodal content. The development of sophisticated evaluation frameworks like UniEval [23] and reward models such as Skywork-VL Reward [53] reflects a growing effort to quantify and guide complex generative behaviors. Despite these parallel advancements in understanding and generation, a critical gap persists in existing evaluation. Benchmarks have largely focused on assessing these two functions as separate capabilities. Consequently, there remains a lack of a framework for evaluating a model's ability to integrate these functions in a **generative reasoning** task—that is, tasks that *require a model to first deeply comprehend multimodal premises and then generate a structured, logically sound, and verifiable artifact as a solution*.

2.3 Code-based Evaluation for Reasoning Tasks

To enable rigorous evaluation, a growing body of work adopts code as a structured mechanism for assessment, leveraging the verifiable nature of code over free-form natural language explanations. MathCoder-VL [49] employs image-to-code supervision to ensure precise visual–textual alignment, while MATP-BENCH [15] integrates formal theorem proving in Lean, Coq, and Isabelle for verifiable multimodal logic. VeriEquivBench [69] proposes an equivalence-based validation metric for verifying consistency between generated code and specifications. InternLM-Math [66] unifies solving and verifying under a single framework with interleaved reasoning and code execution. DeepMath-103K [16] provides a large, contamination-free dataset with verifiable answers for reinforcement learning with rule-based rewards. MathQ-Verify [40] targets problem validity itself through a structured verification pipeline. CMMaTH [24] introduces a large Chinese multimodal math benchmark and an open-source automatic grader, while MARIO Eval [70] standardizes automated evaluation across mathematical datasets. U-MATH [7] assess university-level reasoning and meta-evaluate LLM judges. In parallel, QuesCo [34] captures holistic mathematical intent through contrastive pretraining.

While recent multimodal datasets [29, 37, 76, 48, 42, 50] have advanced the field, they seldom enforce a verifiable alignment between the natural language reasoning steps. To bridge this gap, GGBench extends the code-based evaluation to the visual geometric domain, enabling a holistic verification pipeline that tightly aligns textual reasoning, code execution, and visual output. This multimodal, verifiable setup allows GGBench to move beyond pattern matching and surface-level correctness.

3 The GGBench Benchmark

3.1 Overview

We introduce GGBench, a geometric generative reasoning benchmark designed to evaluate UMMs through end-to-end construction tasks. Unlike conventional discriminative benchmarks that evaluate models solely on final answer selection or numerical computation, GGBench operationalizes the assessment of whether a system can genuinely construct a geometrically valid solution.

A defining feature of GGBench is its tri-modal data structure, where each sample comprises precisely aligned text, code, and image components. This multi-faceted design enables a uniquely versatile and diagnostic evaluation framework. For instance, the complete, end-to-end task of generating a final image from a textual prompt serves as the primary test for the integrated generative reasoning of UMMs. Concurrently, the benchmark can be decoupled to probe specific capabilities: the natural language reasoning steps (text) and their translation into executable programs (code) can be used to assess the logical planning and code generation faculties of LLMs. Furthermore, the provided GeoGebra code acts as an unambiguous, machine-verifiable ground truth, offering a deterministic method for verifying the geometric correctness and precision of any generated figure. An overview of the multi-modal dataset construction pipeline is shown in Figure 3, which defines the stage annotations (a)–(f) referenced throughout the paper.

Figure 3: Overview of the GGBench data construction pipeline.

3.2 Dataset construction

Data collection and candidate pooling We manually search the web for public geometry problems and assemble a pool that spans classical constructions and contest-style tasks (Figure 3, stage (a)). To scale selection beyond purely manual effort, we use an LLM for assisted tagging and filtering (LLM-i in the figure) and then conduct human checking to retain items that are suitable for construction—i.e., problems with unambiguous geometric dependencies and diagrammatically actionable conditions (Figure 3, stage (b)). This stage emphasizes diversity in objects and relations (e.g., circles, tangents, incenters, parallels) while discarding under-specified prompts.

Prompt design and problem adaptation To elicit end-to-end construction in GeoGebra only, we design a *composite* prompt that pairs a textual specification with exemplar GGB code (Figure 3, stage (c)). In parallel, we prepare a reformulation prompt family that converts the selected problems into construction-oriented statements with explicit auxiliary objects and ordered dependencies (Figure 3, stage (d)). In practice, GPT-5 [35] serves as the primary authoring model during adaptation (LLM-ii), while GGB-oriented prompt templates are informed by prior engineering with Gemini 2.5 Pro [9] to improve syntactic fidelity. The outcome is a set of adapted problems whose reasoning steps align one-to-one with executable GGB operations.

Solution generation and visualization Given an adapted problem and its data-generation prompts, GPT-5 [35] (LLM-iii) produces synchronized outputs: (i) stepwise reasoning text, (ii) executable GGB code, and (iii) a rendered diagram obtained by running the code (Figure 3, stage (e)). This stage focuses on faithful construction: steps materialize auxiliary objects (e.g., angle bisectors, tangents, parallels), code reflects those steps, and the rendered figure visualizes the target configuration. At this point we obtain roughly $\sim 10,000$ draft instances prior to filtering.

Automated screening and expert finalization The final stage involved a rigorous two-tier filtering process to ensure dataset quality. First, we perform LLM-based quality control (LLM-iv) along three axes—*code executability*, *logical consistency of the construction process*, and *diagram correctness with respect to the intended figure*—to remove low-quality drafts (Figure 3, stage (f)). Subsequently, all surviving instances underwent a final review by domain experts who verified their geometric correctness, the sufficiency of the construction, and the precise consistency across the text, code, and image modalities. After this two-tier filtering, GGBench retains 1,411 high-quality items with tightly aligned text–code–image triplets. A detailed statistical analysis of the dataset is provided as below.

Table 1: Corpus-level statistics of GGBench.

Dataset size	
Total problems	1,411
Problem types	
Straightedge-and-compass construction	798
Geometric transformation construction	426
Analytic construction	187
Difficulty levels	
Easy	298
Medium	816
Hard	297
Images per problem	
3 images	7
4 images	227
5 images	860
6 images	283
7 images	34
Total images	7,165
Question length (tokens)	
Minimum	68
Maximum	483
Average	189.83

3.3 Dataset Analysis

Dataset Composition and Scale   GGBench comprises 1,411 GeoGebra-based construction problems that integrate reasoning, code, and visual outputs. Table 1 summarizes overall dataset statistics, including problem types, difficulty levels, image counts, and question lengths. Most problems belong to the medium difficulty (57.83%), balancing solvability and reasoning complexity. Straightedge-and-compass constructions dominate (56.6%), followed by geometric transformation and analytic constructions. Each problem includes between 3 and 7 diagrams, averaging 5.08 images per problem, providing dense visual evidence for evaluating generative reasoning.

Table 2: Distribution of reasoning categories in GGBench. Each problem may involve multiple categories, yielding 3,097 total tags across 1,411 problems.

Category	Count
Basic Constructions	1,063
Circle Properties & Constructions	931
Geometric Transformations	376
Triangle Properties & Constructions	280
Applications of Geometric Theorems	218
Polygon Properties & Constructions	107
Measurement & Ratios	92
Locus Constructions	30
Total category tags	3,097

Category Distribution and Difficulty   GGBench covers eight major categories of geometric reasoning, as detailed in Table 2. Each problem may involve multiple skills, resulting in 3,097 category annotations across the dataset. As shown in Figure 4, *Basic Constructions* and *Circle Properties & Constructions* are predominant across all difficulty levels, while advanced reasoning types—such as *Applications of Geometric Theorems* and *Measurement & Ratios*—become more frequent in the hard subset. This stratification ensures balanced coverage from elementary to advanced geometric reasoning.

Category Composition and Cognitive Coverage   The distribution in Table 2 reflects a deliberate cognitive gradient. The majority of tasks emphasize procedural reasoning (e.g., bisectors, parallels, and tangents), providing a stable foundation for evaluating geometric comprehension. Intermediate categories, including *Transformations* and *Triangles*, introduce abstraction through relational constraints and inter-object dependencies. Higher-order categories—such as *Theorem Applications* and *Measurement & Ratios*—demand symbolic reasoning, multi-step dependencies, and precision alignment between text and geometry. This stratification enables a fine-grained evaluation of a model's capabilities, from foundational geometric execution to advanced, abstract problem-solving.

Token Length and Reasoning Path   We report question length in tokens. As shown in Table 1, question length ranges from 68 to 483 tokens, averaging 189.83 tokens. This range reflects the multi-step reasoning nature of GGBench tasks, requiring the model to plan, describe, and validate geometric constructions. On average, each problem contains 5.08 rendered images (7,165/1,411), and carries 2.20 category tags (3,097/1,411), indicating that GGBench is both visually dense and multi-skill by design. The five-image setting is the most common (60.9%; 860/1,411), providing ample visual evidence for stepwise construction. While *Basic Constructions* and *Circle Properties & Constructions* dominate across the corpus, Figure 4 further shows a progressive shift toward higher-order skills (e.g., *Applications of Geometric Theorems*, *Measurement & Ratios*) in the hard subset.
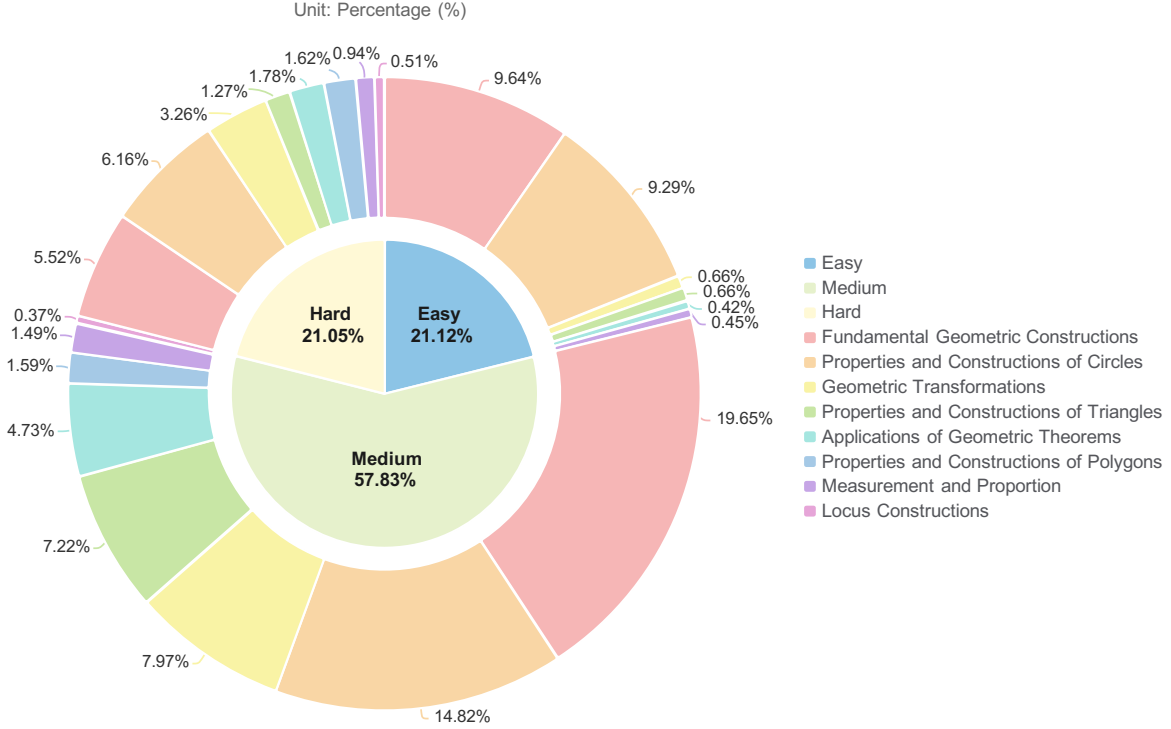
Figure 4: Difficulty distribution and category composition in GGBench. The inner ring shows the proportion of difficulty levels (Easy/Medium/Hard), while the outer ring presents category shares within each difficulty band, reflecting progressive complexity across reasoning types.

3.4 Comparison with Existing Benchmarks

Table 3 situates GGBench among representative multimodal math benchmarks along two groups of axes: (i) *capability coverage*—the proportion of samples that exercise *understanding*, *generation*, and *multi-step* reasoning; and (ii) *modality supervision*—whether problems provide aligned *text*, *images*, and *code*. We normalize modality coverage as a binary indicator in the table (✓/✗) for readability.

Generative construction vs. answer selection Most benchmarks [29, 76] emphasize recognition or answer selection. Even when generation is present [72], outputs are rarely required to be *constructive artifacts* that satisfy constraints. By contrast, every GGBench instance requires producing diagrams with formal constraints, shifting evaluation from choosing an answer to *constructing evidence*.

Tri-modal alignment and verifiability A central gap in prior work is the absence of code supervision. It is difficult to verify whether a model's reasoning aligns with the final diagram. GGBench provides 100% alignment between *text* (stepwise plan), *code* (executable construction), and *image* (rendered diagram), enabling process-level judging, program executability checks, and geometry-aware verification within a single benchmark.

Multi-step supervision While some datasets include multi-step elements [44, 72], they do not couple each step with executable constructions, limiting verifiability. GGBench enforces multi-step reasoning at *100%* and ties steps to code, ensuring that auxiliary objects and dependencies are operationalized rather than narrated.

Table 3: Comparison with existing multimodal mathematical benchmarks. Percentages under *Understanding/Generation/Multi-step* indicate the share of samples exercising each capability. *Text/Image/Code* report modality support (%). GGBench uniquely achieves full tri-modal coverage with executable code.

Benchmark	Understanding	Generation	Multi-step	Text	Image	Code
MathVista [29]	49.9	34.8	21.5	✓	✓	✗
MathVerse [72]	61.3	52.7	45.2	✓	✓	✗
MM-MATH [44]	31.0	26.5	82.0	✓	✓	✗
MathScape [76]	44.2	39.8	18.5	✓	✓	✗
GeoEval [71]	46.0	25.0	33.3	✓	✓	✗
WE-MATH [37]	58.7	51.9	42.6	✓	✓	✗
MMSciBench [65]	55.4	32.8	40.2	✓	✓	✗
MATH2VISUAL [47]	48.0	37.0	✗	✓	✓	✗
PolyMath [14]	54.6	✗	✗	✓	✗	✗
SOLIDGEO [51]	49.5	18.7	6.7	✓	✓	✗
GGBench	100.0	100.0	100.0	✓	✓	✓

Modal breadth and heterogeneity. Most benchmarks offer full text and image coverage (100%/100%), with exceptions such as MMSciBench [65] (90%/40%) and PolyMath [14] (100%/0%). GGBench maintains complete modality support (text/image/code all 100%), so reasoning, rendering, and execution can be evaluated cohesively.

Together, these dimensions position GGBench as a construction-centric benchmark that unifies understanding, reasoning, and generation under a verifiable framework. It complements prior understanding-oriented datasets, offering a rigorous testbed for the next generation of unified multimodal reasoning models.

4 Experiment

Setup. We evaluate existing Unified Multimodal Models on GGBench to assess their generative geometric reasoning capabilities. Each model receives the problem text and reference image as input. To ensure deterministic outputs, we fix the temperature at 0.0 during inference. This eliminates stochasticity and enforces consistent step-by-step reasoning. All code generation adheres strictly to the GeoGebra command syntax to ensure executability and structural correctness. Inference is parallelized for efficiency. All models are evaluated under identical settings with unified prompts and processing pipelines. Full prompt templates are detailed in Appendix.

4.1 Baselines

We evaluate models under two complementary tracks that correspond to distinct architectural paradigms: (A) end-to-end UMMs that directly generate diagram images from natural-language prompts, and (B) LLMs/LRMs that first produce explicit, executable construction code which is then rendered into diagrams. This dual-track evaluation allows us to quantify and analyze the gap between UMMs' immediate visual generation and the geometrically grounded, code-driven constructions produced by LLMs/LRMs.

Track A: End-to-end UMMs These systems take natural-language specifications and output diagram images step-by-step. We include: *Qwen-Image* [56], *Seedream 4.0* [39], *Janus* [57], *BAGEL* [10], and *Nano*

Banana [12]. Among them, *Qwen-Image* and *Seedream 4.0* produce only final images without stepwise explanations, whereas the other models generate multi-stage visual outputs for step evaluation.

Track B: Planning → Code → Render These models first produce a *textual plan* and then emit *GeoGebra code* to render into images: *GPT-4o* [20], *GLM-4.5V* [18], *Qwen3VL-235B-A22B* [63], *GPT-4* [1], *GPT-5* [35], *Claude Sonnet 4.5* [3], *Gemini-2.5-Pro* [9], *DeepSeek-R1* [25], *DeepSeek-V3.1* [25], and *Qwen3-14B* [63].

4.2   Metrics

To evaluate geometric generative reasoning, We adopt a four-stage protocol: (1) *Planning*, (2) *Middle Process*, (3) *Final Result*, and (4) *Overall Scores*. All automatic scores are produced by a VLM GPT-4o using fixed prompts.

(1) Planning (VLM-T) measures the model's ability to formulate a coherent step-by-step plan in natural language before any code or drawing is produced. Given the textual reasoning output, the VLM judge scores each response along three dimensions: (i) Logical coherence, (ii) Step completeness, and (iii) Geometric correctness. Each criterion is rated on a 1–5 scale and rescaled to [0,100]. Details of the judging rubric and prompt design appear in Appendix B.2.

(2) Middle process (VLM-I-Mid) To evaluate intermediate reasoning consistency, we concatenate all intermediate construction images into a chronological panel and feed it to the same VLM judge. The model assesses: (i) Step accuracy and (ii) Process consistency. Implementation and prompt details are provided in Appendix B.4.

(3) Final result (VLM-I-Res) assesses the geometric correctness of the final diagram relative to the reference solution. In addition, we also report pixel-level metrics. Details are in Appendix B.3.

(4) Overall scores The overall VLM score (VLM–I) is computed as the mean of the intermediate VLM-I-Mid and final scores VLM-I-Res. Human raters follow the same rubric as the VLM judge; inter-rater consistency and calibration procedures are detailed in Appendix B.6. We observe a strong Pearson correlation ($r = 0.9295$) between VLM and human scores, validating the high reliability of our automated evaluation for this task.

4.3   Main Results

Table 4 summarizes the overall performance across both tracks. End-to-end UMMs remain significantly behind code-driven models in nearly all evaluation dimensions. Even the strongest UMM, *Nano Banana*, ranks only in the middle tier of code-based systems, illustrating that direct visual generation still struggles to enforce geometric constraints. C"ode-level metrics of LLMs/LRMs are reported in Table 5.

The planning stage (VLM-T) evaluates a model's capacity to reason step-by-step before any code or drawing is produced. Interestingly, *Nano Banana* achieves planning scores comparable to several LLMs/LRMs. However, models such as *GPT-4o* and *GLM-4.5V*, despite generating coherent plans, often fail to produce executable code, limiting their downstream visual accuracy. This gap demonstrates that high-level reasoning alone is insufficient without grounding in executable geometry.

Intermediate and final constructions (VLM–I) Across both the intermediate (VLM{I_{mid}}) and final (VLM{I_{res}}) stages, code-driven models consistently outperform end-to-end generators. Explicitly reasoning through textual plans and producing executable constructions leads to markedly higher geometric correctness and visual coherence. While robust planning does not always guarantee flawless drawings, weak or inconsistent reasoning almost always results in invalid geometry—reinforcing the necessity of tightly coupled reasoning–generation pipelines. Moreover, purely pixel-based metrics (PSNR, SSIM, LPIPS) show only weak correlation with geometric validity: high perceptual similarity can mask structural errors such as misaligned intersections or missing tangencies. Hence, evalua-

Table 4: Main results on GGBench. Higher is better (↑) except for LPIPS (↓).

Model	Planning	Middle Process	Final Result				Overall Scores	
	VLM-T ↑	VLM-I-Mid ↑	VLM-I-Res ↑	LPIPS ↓	PSNR ↑	SSIM ↑	VLM-I ↑	Human ↑
End-to-end UMMs								
Qwen-Image [56]	-	-	22.75	56.39	58.23	48.06	22.75	25.56
Seedream 4.0 [39]	-	-	24.45	51.06	59.44	56.44	24.45	37.56
Janus [57]	33.85	21.69	19.76	57.74	57.76	60.97	20.73	19.46
BAGEL [10]	23.07	21.84	19.99	57.07	61.78	58.82	20.91	20.12
Nano Banana [9]	58.54	44.83	22.81	51.85	64.53	59.51	**33.82**	**45.75**
LLMs/LRMs								
GPT-4o [20]	59.73	26.19	2.66	95.43	5.45	5.69	14.43	23.04
GLM-4.5V [18]	53.32	25.63	5.02	52.91	12.19	12.94	15.33	30.14
Qwen3-14B [63]	58.65	39.30	12.97	78.81	23.92	24.81	26.13	38.23
Gemini 2.5 Pro [9]	38.50	37.41	15.80	68.39	37.17	39.73	26.61	44.68
DeepSeek-R1 [25]	61.16	62.42	20.48	66.06	37.94	37.59	41.45	49.55
GPT-4 [1]	55.66	50.99	15.10	67.35	35.26	38.31	33.04	55.99
Qwen3-VL [63]	56.40	49.55	23.94	39.40	52.33	58.71	36.74	66.77
DeepSeek-V3.1 [25]	60.24	73.13	26.41	57.21	48.33	50.12	49.77	68.12
Claude Sonnet 4.5 [3]	61.19	77.92	30.29	52.22	51.74	50.52	54.11	72.12
GPT-5 [35]	62.01	76.79	37.36	49.65	54.80	59.49	**57.08**	**83.06**

Table 5: Evaluation results on GGBench-Code across execution, similarity, and structural metrics.

Model	Pass@1 ↑	BLEU Mean ↑	RUBY Mean ↑	ROUGE-L Mean ↑	chrF Mean ↑	EditDist Mean ↓
GPT-4o [20]	7.87	8.33	12.85	27.41	29.17	78.92
GLM-4.5V [18]	14.25	10.28	17.53	30.48	38.32	79.61
Qwen3-14B [63]	34.30	14.04	23.85	38.07	44.36	76.89
Gemini 2.5 Pro [9]	32.67	24.21	40.56	50.50	65.87	71.14
DeepSeek-R1 [25]	54.36	18.46	33.68	42.69	60.11	73.64
GPT-4 [1]	50.53	17.31	28.54	44.62	51.25	70.34
Qwen3-VL- [63]	69.31	23.24	38.61	50.73	63.51	70.24
DeepSeekV3.1 [25]	70.09	20.44	35.14	47.36	59.24	70.87
Claude Sonnet 4.5 [3]	75.34	22.04	36.77	49.68	60.28	70.81
GPT-5 [35]	79.02	18.87	39.92	44.08	62.44	74.13

tion based on verifiable geometry, rather than appearance alone, offers a more reliable measure of multimodal reasoning quality.

Code-based analysis (Table 5) At the program level, *GPT-5* achieves the highest execution accuracy (pass@1=79.02), followed closely by *Claude Sonnet 4.5* (75.34) and *DeepSeek-V3.1* (70.09). Interestingly, code-similarity metrics (e.g., BLEU or ROUGE-L) do not necessarily predict executable success: *Gemini-2.5-Pro* attains high textual similarity yet a lower Pass@1, confirming that surface-level resemblance does not imply geometric equivalence. This observation validates the necessity of execution-based evaluation in GGBench.

Summary of findings In summary, reasoning-grounded models that produce and execute code achieve substantially higher geometric correctness, interpretability, and human-rated quality. *GPT-5* attains the best overall performance (VLM{I=57.08, Human=83.06), followed by *Claude Sonnet 4.5* and *DeepSeek-V3.1*. Despite recent advances, purely generative UMMs remain limited in enforcing spatial constraints, emphasizing the importance of explicit, verifiable construction pipelines for next-generation multimodal reasoning systems.

4.4 Performance by Construction Category

	Claude-Sonnet-4.5	GPT-5	DeepSeek V3.1	DeepSeek-R1	Janus	Qwen3-VL	GPT-4	Nano Banana	Gemini-2.5-pro	Qwen3-14B	Doubao-Seedream	Qwen-Image	BAGEL	GLM-4.5V	GPT-4o
Locus Construction	57.333	62.000	49.500	47.833	41.167	39.000	36.333	35.333	17.500	22.833	21.333	21.333	20.500	26.333	21.833
Measurement & Ratios	65.978	59.402	52.989	36.957	34.511	31.685	32.120	35.543	27.228	22.446	22.826	22.609	22.391	14.565	17.337
Polygon Properties & Constructions	65.888	59.953	64.579	45.467	39.393	35.654	36.729	33.037	33.364	30.794	25.607	23.925	20.794	15.561	15.000
Application of Theorems	65.550	66.720	56.284	49.954	41.560	39.312	39.289	34.335	29.633	29.151	23.578	22.569	22.317	19.656	15.000
Triangle Properties & Constructions	60.518	58.446	52.661	40.982	39.089	35.393	34.875	40.661	32.000	28.268	23.786	22.143	24.250	18.268	15.750
Geometric Transformations	67.620	67.327	53.936	45.319	40.785	37.580	35.319	38.590	35.053	26.955	24.787	22.181	22.553	19.455	16.662
Circle Properties & Constructions	66.460	71.631	64.503	56.433	47.037	44.898	43.914	36.941	33.866	36.160	24.898	22.738	21.048	21.449	21.963
Basic Constructions	63.811	65.211	64.887	53.835	46.687	44.323	43.351	38.741	28.299	33.938	24.173	22.989	22.397	20.742	21.579

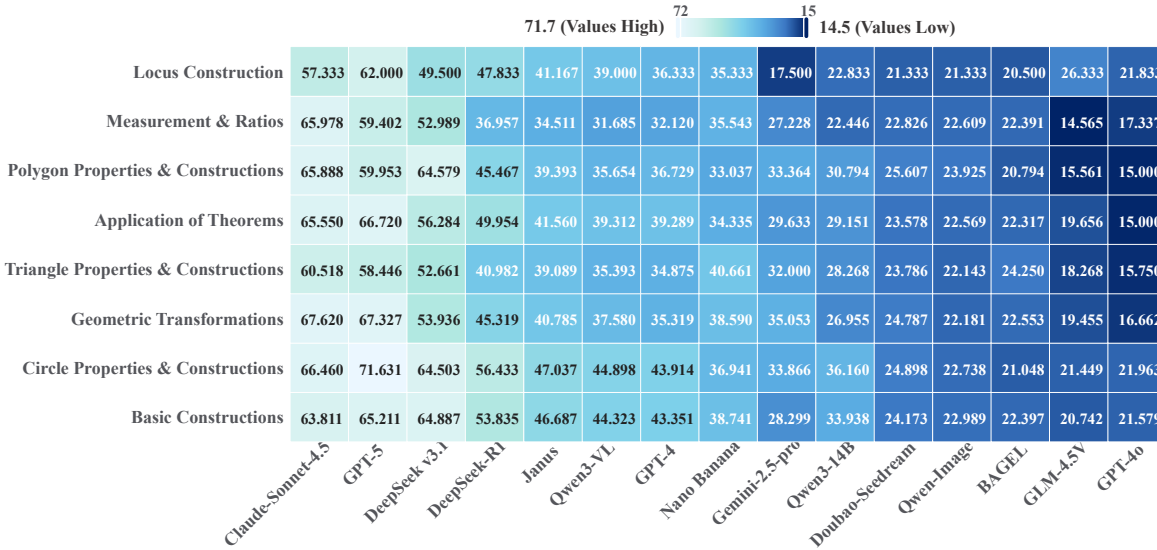71.7 (Values High)   72   15   14.5 (Values Low)

Figure 5: VLM-I scores across eight construction categories in GGBench. Each cell reflects the average multimodal reasoning quality for a model-category pair.

Figure 5 reports the average VLM–I scores across eight construction categories in GGBench, revealing distinct performance patterns among models. GPT-5 [35] achieves the highest overall accuracy, consistently outperforming other models across all categories. Its strength lies in maintaining both *symbolic precision* (accurate translation from plan to executable code) and *spatial coordination* (faithful geometric realization). Claude-Sonnet-4.5 [3] ranks second, closely followed by DeepSeek-V3.1 [25], particularly excelling in tasks requiring geometric regularity such as *Geometric Transformations* and *Circle Properties & Constructions*. These models exhibit stable reasoning across categories, indicating robust integration between linguistic and spatial modalities. Qwen3-VL [63] demonstrates strong performance on structurally constrained categories—such as *Basic Constructions* and *Triangle Properties & Constructions*—where relationships among objects are more rule-based, but its scores decline in *Measurement & Ratios* and *Applications of Theorems*, which require quantitative and theorem-level reasoning. This contrast suggests that while Qwen3-VL effectively grounds syntactic geometry, it struggles with deeper symbolic arithmetic integration. At the lower end, vision-first models such as GPT-4o [20], GLM-4.5V [18], and Qwen-Image [56] consistently underperform, with mean scores below 23 across most categories. Their weakness is especially evident in tasks requiring explicit relational constraints or quantitative precision, implying insufficient geometric grounding despite competent visual synthesis. Across all models, the most challenging categories are *Measurement & Ratios* and *Applications of Theorems*, where performance drops by 10–15 points relative to simpler procedural tasks. These categories require not only visual generation but also algebraic and deductive reasoning, emphasizing the need for symbolic–geometric alignment. Conversely, *Basic Constructions* and *Circle Properties* yield the highest scores overall, indicating that most models can handle canonical constructions but fail to generalize to theorem-driven or quantitative reasoning.

Discussion The category-wise trends highlight the diagnostic value of GGBench. By decomposing performance along interpretable geometric skills, the benchmark exposes which reasoning components—procedural, transformational, or quantitative—limit current multimodal systems. The results confirm that achieving high-fidelity diagram generation alone does not guarantee correct geometric reasoning, underscoring the necessity of construction-level, verifiable evaluation frameworks.

4.5 Performance by Question Type

Figure 6 presents the performance of all evaluated models across three geometric task types: analytic construction (AC), geometric transformation construction (GTC), and straightedge-and-compass construction (SCC), measured using the unified VLM-I metric. Clear distinctions emerge across categories. Models generally perform best on SCC tasks that emphasize rule-based geometric procedures, where GPT-5 [35] reaches 72.54, followed closely by Claude-Sonnet-4.5 [3] and DeepSeek V3.1 [25], both above 66. Performance on GTC tasks is moderately lower: Claude-Sonnet-4.5 records 66.26 and DeepSeek V3.1 achieves 53.53, reflecting the increased complexity of maintaining geometric invariants under rotation or reflection. Analytic construction tasks yield the lowest results, with GPT-5 scoring 51.90 and DeepSeek V3.1 reaching 46.07, likely due to the greater spatial flexibility and reduced structural constraints of such problems. UMMs such as Janus [57], BAGEL [10], and Seedream [39] remain consistently weaker due to the absence of symbolic reasoning, while text-only systems like DeepSeek V3.1 show competitive performance on structure-driven categories through accurate code synthesis. Overall, these findings indicate that model effectiveness is shaped jointly by modality alignment and the specific geometric reasoning paradigm required by the task.
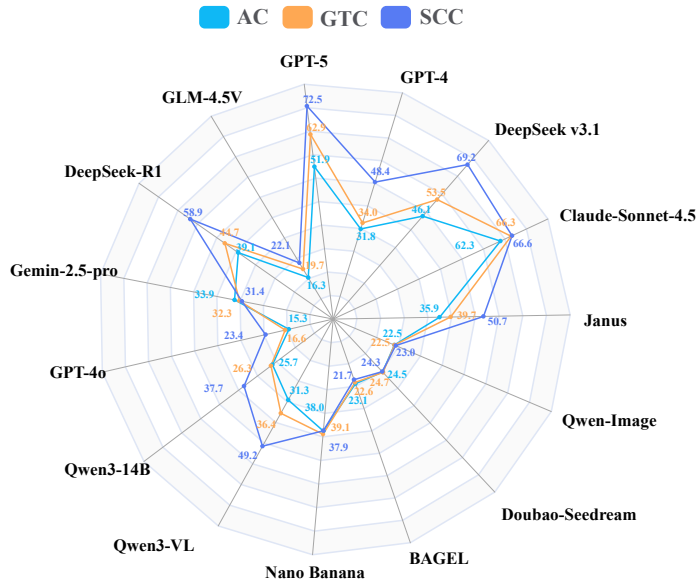

Figure 6: VLM-I scores across geometric task types: Analytic Construction (AC), Geometric Transformation Construction (GTC), and Straightedge-and-Compass Construction (SCC). Higher values indicate better performance.

Discussion Overall, performance varies systematically with the degree of symbolic structure embedded in the task. Tasks governed by explicit geometric rules (SCC) favor models with strong language-to-code translation ability, whereas transformation and analytic tasks expose weaknesses in maintaining geometric invariants and algebraic reasoning. These results underscore that multimodal effectiveness in GGBench is shaped jointly by a model's modality alignment and by the specific reasoning paradigm—procedural, transformational, or analytic—required by the construction type.

4.6 Performance by Difficulty Level

Figure 7 presents the VLM-I scores across three difficulty levels—Easy, Medium, and Hard—providing a comprehensive view of how reasoning complexity impacts model performance. The majority of
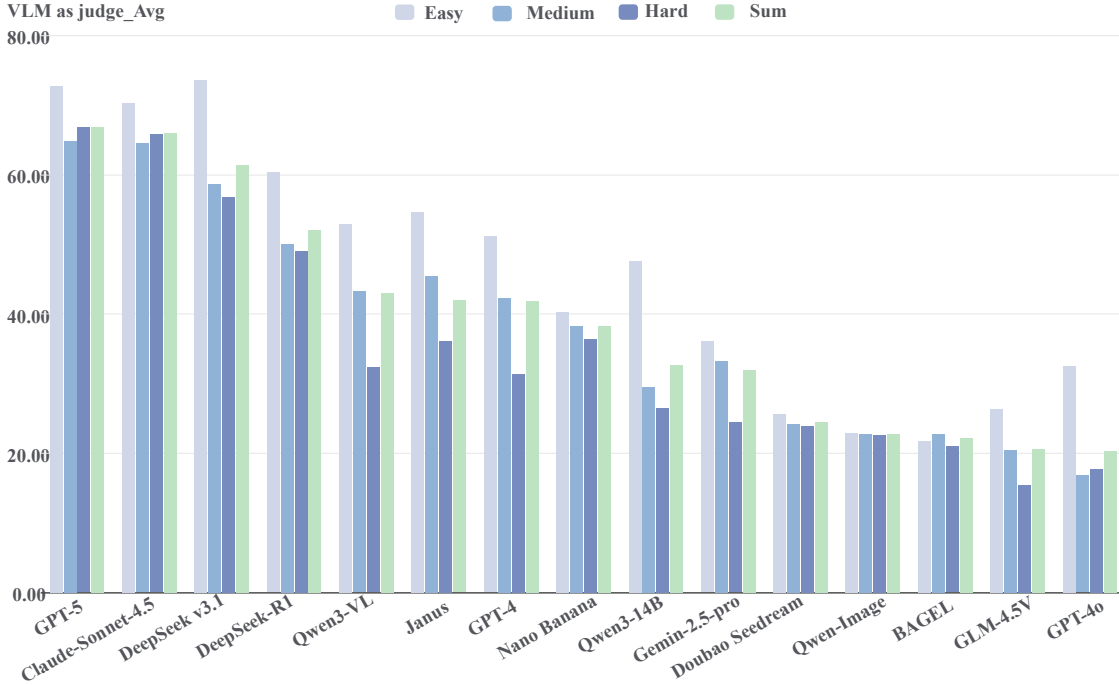
Figure 7: VLM-I performance across difficulty levels on GGBench. Bars represent *Easy*, *Medium*, *Hard*, and overall scores. VLM-I captures both intermediate reasoning quality and final visual correctness.

systems perform best on Easy problems and experience a steady decline as task difficulty increases, confirming the progressive design of GGBench. This pattern is especially pronounced for models like Qwen3-14B [63] and GPT-4 [1], which drop more than 20 points from Easy to Hard, suggesting that their performance is disproportionately sensitive to longer reasoning chains and higher geometric abstraction. In contrast, GPT-5 [35] maintains strong results across all difficulty tiers, scoring 72.70 on Easy and 66.77 on Hard, highlighting its robustness in both planning and execution. Claude-Sonnet-4.5 [3] follows a similar trend, while DeepSeek V3.1 [25] notably surpasses both on Easy problems with a peak of 73.57. UMMs like Nano Banana [9], Janus [57], and Seedream [39] show more stable but consistently lower performance across difficulties, indicating weaker adaptation to reasoning complexity despite visual fluency. These observations reinforce that difficulty scaling in GGBench effectively distinguishes between symbolic robustness and shallow pattern matching, making VLM-I a reliable diagnostic signal for geometric reasoning across complexity levels.

5 Error Analysis

To better understand failure patterns in geometric reasoning, we analyze incorrect outputs from benchmark evaluation and identify four common error types: (i) geometric logic errors, due to misapplied theorems; (ii) structural and contextual errors, involving confusion in figure containment or spatial relationships; (iii) conflation of construction and numeric goals; and (iv) code-level failures from syntax misuse or reserved keyword conflicts. Figure 8 presents a structural and contextual error. The task asks for a rectangle inscribed in a circle, but the model produces a circle inside a square, reversing the intended containment. This highlights a failure to ground symbolic reasoning in geometric constraints, where surface-level textual correctness leads to invalid constructions.
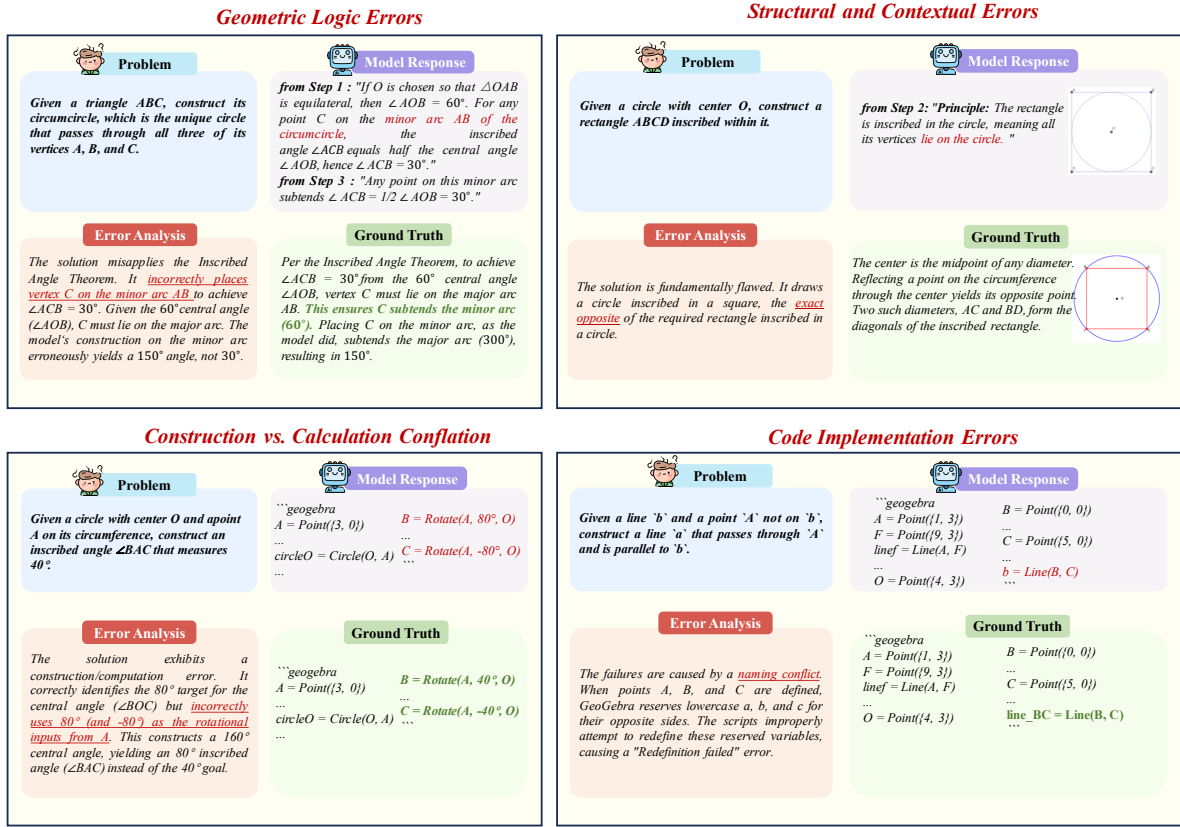
Figure 8: The common error analysis.

6 Conclusion

We introduce **GGBench**, the first benchmark explicitly designed to evaluate *geometric generative reasoning*—the ability of a model to not only understand and reason but also to construct verifiable solutions. Each problem in GGBench integrates natural-language instructions, executable GeoGebra code, and rendered diagrams, forming a tri-modal, interpretable, and fully verifiable testbed. Our unified evaluation protocol jointly measures textual planning quality, code executability and geometric equivalence, and visual diagram fidelity, providing a holistic view of multimodal reasoning competence. Comprehensive experiments across state-of-the-art UMMs and LLMs/LRMs reveal a consistent gap between end-to-end image generation and reasoning-grounded construction. While UMMs excel at perceptual synthesis, they often fail to satisfy precise geometric constraints; in contrast, code-generating models demonstrate higher logical coherence and geometric correctness but remain limited in visual expressivity. We hope GGBench will serve as a rigorous foundation for future research in grounded, verifiable multimodal intelligence. Beyond geometry, our framework points toward a broader paradigm for evaluating generative reasoning—linking *language, logic, and construction*—as a step toward AI systems capable of building as well as understanding.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv e-prints*, pages arXiv–2303, 2023.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[3] Anthropic. Claude sonnet 4.5 system card. Technical report, Anthropic PBC, 2025. Official system card describing Claude Sonnet 4.5 capabilities and safety evaluation. Available at: https://assets.anthropic.com/m/12f214efcc2f457a/original/Claude-Sonnet-4-5-System-Card.pdf.

[4] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[5] Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric Xing, and Liang Lin. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 513–523, 2021.

[6] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

[7] Konstantin Chernyshev, Vitaliy Polshkov, Ekaterina Artemova, Alex Myasnikov, Vlad Stepanov, Alexei Miasnikov, and Sergei Tilga. U-math: A university-level benchmark for evaluating mathematical skills in llms. *arXiv preprint arXiv:2412.03205*, 2024.

[8] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

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

[10] Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

[11] Chengqi Duan, Kaiyue Sun, Rongyao Fang, Manyuan Zhang, Yan Feng, Ying Luo, Yufang Liu, Ke Wang, Peng Pei, Xunliang Cai, et al. Codeplot-cot: Mathematical visual reasoning by thinking with code-driven images. *arXiv preprint arXiv:2510.11718*, 2025.

[12] Alisa Fortin, Guillaume Vernade, Kat Kampf, and Ammaar Reshi. Introducing Gemini 2.5 Flash Image: our state-of-the-art image generation and editing model. Google Developers Blog, Aug 2025. Accessed 2025-10-27, available online at: https://developers.googleblog.com/en/introducing-gemini-2-5-flash-image/.

[13] Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing HONG, Jianhua Han, Hang Xu, Zhenguo Li, et al. G-llava: Solving geometric problem with multi-modal large language model. In *The Thirteenth International Conference on Learning Representations*, 2025.

[14] Himanshu Gupta, Shreyas Verma, Ujjwala Anantheswaran, Kevin Scaria, Mihir Parmar, Swaroop Mishra, and Chitta Baral. Polymath: A challenging multi-modal mathematical reasoning benchmark. *arXiv preprint arXiv:2410.14702*, 2024.

[15] Zhitao He, Zongwei Lyu, Dazhong Chen, Dadi Guo, and Yi R Fung. Matp-bench: Can mllm be a good automated theorem prover for multimodal problems? *arXiv preprint arXiv:2506.06034*, 2025.

[16] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025.

[17] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[18] Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv e-prints*, pages arXiv–2507, 2025.

[19] Muye Huang, Lingling Zhang, Jie Ma, Han Lai, Fangzhi Xu, Yifei Li, Wenjun Wu, Yaqiang Wu, and Jun Liu. Chartsketcher: Reasoning with multimodal feedback and reflection for chart understanding. *arXiv preprint arXiv:2505.19076*, 2025.

[20] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[21] Jiulin Li, Ping Huang, Yexin Li, Shuo Chen, Juewen Hu, and Ye Tian. A unified multi-agent framework for universal multimodal understanding and generation. *arXiv preprint arXiv:2508.10494*, 2025.

[22] Xiaoyuan Li, Moxin Li, Wenjie Wang, Rui Men, Yichang Zhang, Fuli Feng, Dayiheng Liu, and Junyang Lin. Mathopeval: A fine-grained evaluation benchmark for visual operations of mllms in mathematical reasoning. *arXiv preprint arXiv:2507.18140*, 2025.

[23] Yi Li, Haonan Wang, Qixiang Zhang, Boyu Xiao, Chenchang Hu, Hualiang Wang, and Xiaomeng Li. Unieval: Unified holistic evaluation for unified multimodal understanding and generation. *arXiv preprint arXiv:2505.10483*, 2025.

[24] Zhongzhi Li, Ming-Liang Zhang, Pei-Jie Wang, Jian Xu, Rui-Song Zhang, Yin Fei, Zhi-Long Ji, Jin-Feng Bai, Zhen-Ru Pan, Jiaxin Zhang, et al. Cmmath: A chinese multi-modal math skill evaluation benchmark for foundation models. In *ACL*, pages 2690–2726, 2025.

[25] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *CoRR*, 2024.

[26] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer, 2024.

[27] Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision language audio and action. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26439–26455, 2024.

[28] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

[29] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

[30] Yanzuo Lu, Xin Xia, Manlin Zhang, Huafeng Kuang, Jianbin Zheng, Yuxi Ren, and Xuefeng Xiao. Hyper-bagel: A unified acceleration framework for multimodal understanding and generation. *arXiv preprint arXiv:2509.18824*, 2025.

[31] Jingkun Ma, Runzhe Zhan, Derek F Wong, Yang Li, Di Sun, Hou Pong Chan, and Lidia S Chao. Visaidmath: Benchmarking visual-aided mathematical reasoning. *arXiv preprint arXiv:2410.22995*, 2024.

[32] Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, et al. Janusflow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7739–7751, 2025.

[33] Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Anton Belyi, et al. Mm1: methods, analysis and insights from multimodal llm pre-training. In *European Conference on Computer Vision*, pages 304–323. Springer, 2024.

[34] Yuting Ning, Zhenya Huang, Xin Lin, Enhong Chen, Shiwei Tong, Zheng Gong, and Shijin Wang. Towards a holistic understanding of mathematical questions with contrastive pre-training. In *AAAI*, volume 37, pages 13409–13418, 2023.

[35] OpenAI. Gpt-5 system card. Technical report, OpenAI, 2025. Official system card document for GPT-5; available at: https://cdn.openai.com/pdf/8124a3ce-ab78-4f06-96eb-49ea29ffb52f/gpt5-system-card-aug7.pdf.

[36] Nilay Pande, Sahiti Yerramilli, Jayant Sravan Tamarapalli, and Rynaa Grover. Marvl-qa: A benchmark for mathematical reasoning over visual landscapes. *arXiv preprint arXiv:2508.17180*, 2025.

[37] Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Zhuoma GongQue, Shanglin Lei, Zhe Wei, Miaoxuan Zhang, et al. We-math: Does your large multimodal model achieve human-like mathematical reasoning? *arXiv preprint arXiv:2407.01284*, 2024.

[38] Hanoona Rasheed, Abdelrahman Shaker, Anqi Tang, Muhammad Maaz, Ming-Hsuan Yang, Salman Khan, and Fahad Shahbaz Khan. Videomathqa: Benchmarking mathematical reasoning via multimodal understanding in videos. *arXiv preprint arXiv:2506.05349*, 2025.

[39] Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, Zhiyao Guo, Xiaoxia Hou, Weilin Huang, Yixuan Huang, et al. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv e-prints*, pages arXiv–2509, 2025.

[40] Chengyu Shen, Zhen Hao Wong, Runming He, Hao Liang, Meiyi Qiang, Zimo Meng, Zhengyang Zhao, Bohan Zeng, Zhengzhou Zhu, Bin Cui, et al. Let's verify math questions step by step. *arXiv preprint arXiv:2505.13903*, 2025.

[41] Weikang Shi, Aldrich Yu, Rongyao Fang, Houxing Ren, Ke Wang, Aojun Zhou, Changyao Tian, Xinyu Fu, Yuxuan Hu, Zimu Lu, et al. Mathcanvas: Intrinsic visual chain-of-thought for multimodal mathematical reasoning. *arXiv preprint arXiv:2510.14958*, 2025.

[42] Wenhao Shi, Zhiqiang Hu, Yi Bin, Junhua Liu, Yang Yang, See Kiong Ng, Lidong Bing, and Roy Lee. Math-llava: Bootstrapping mathematical reasoning for multimodal large language models. In *EMNLP*, pages 4663–4680, 2024.

[43] Yuxin Song, Wenkai Dong, Shizun Wang, Qi Zhang, Song Xue, Tao Yuan, Hu Yang, Haocheng Feng, Hang Zhou, Xinyan Xiao, et al. Query-kontext: An unified multimodal model for image generation and editing. *arXiv preprint arXiv:2509.26641*, 2025.

[44] Kai Sun, Yushi Bai, Ji Qi, Lei Hou, and Juanzi Li. Mm-math: Advancing multimodal math evaluation with process evaluation and fine-grained classification. *arXiv preprint arXiv:2404.05091*, 2024.

[45] Cheng Tan, Jingxuan Wei, Zhangyang Gao, Linzhuang Sun, Siyuan Li, Ruifeng Guo, Bihui Yu, and Stan Z Li. Boosting the power of small multimodal reasoning models to match larger models with self-consistency training. In *European Conference on Computer Vision*, pages 305–322. Springer, 2024.

[46] Trieu Trinh and Thang Luong. Alphageometry: An olympiad-level ai system for geometry. *Google DeepMind*, 17, 2024.

[47] Junling Wang, Anna Rutkiewicz, April Yi Wang, and Mrinmaya Sachan. Generating pedagogically meaningful visuals for math word problems: A new benchmark and analysis of text-to-image models. *arXiv preprint arXiv:2506.03735*, 2025.

[48] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. *NeurIPS*, 37:95095–95169, 2024.

[49] Ke Wang, Junting Pan, Linda Wei, Aojun Zhou, Weikang Shi, Zimu Lu, Han Xiao, Yunqiao Yang, Houxing Ren, Mingjie Zhan, et al. Mathcoder-vl: Bridging vision and code for enhanced multimodal mathematical reasoning. *arXiv preprint arXiv:2505.10557*, 2025.

[50] Peijie Wang, Zhong-Zhi Li, Fei Yin, Dekang Ran, and Cheng-Lin Liu. Mv-math: Evaluating multimodal math reasoning in multi-visual contexts. In *CVPR*, pages 19541–19551, 2025.

[51] Peijie Wang, Chao Yang, Zhong-Zhi Li, Fei Yin, Dekang Ran, Mi Tian, Zhilong Ji, Jinfeng Bai, and Cheng-Lin Liu. Solidgeo: Measuring multimodal spatial math reasoning in solid geometry. *arXiv preprint arXiv:2505.21177*, 2025.

[52] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[53] Xiaokun Wang, Peiyu Wang, Jiangbo Pei, Wei Shen, Yi Peng, Yunzhuo Hao, Weijie Qiu, Ai Jian, Tianyidan Xie, Xuchen Song, et al. Skywork-vl reward: An effective reward model for multimodal understanding and reasoning. *arXiv preprint arXiv:2505.07263*, 2025.

[54] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[55] Jingxuan Wei, Caijun Jia, Qi Chen, Honghao He, Linzhuang Sun, Conghui He, Lijun Wu, Bihui Yu, and Cheng Tan. Geoint-r1: Formalizing multimodal geometric reasoning with dynamic auxiliary constructions. *arXiv preprint arXiv:2508.03173*, 2025.

[56] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv e-prints*, pages arXiv–2508, 2025.

[57] Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. In *CVPR*, pages 12966–12977, 2025.

[58] Teng Xiao, Zuchao Li, and Lefei Zhang. Omnibridge: Unified multimodal understanding, generation, and retrieval via latent space alignment. *arXiv preprint arXiv:2509.19018*, 2025.

[59] Yicheng Xiao, Lin Song, Rui Yang, Cheng Cheng, Zunnan Xu, Zhaoyang Zhang, Yixiao Ge, Xiu Li, and Ying Shan. Haploomni: Unified single transformer for multimodal video understanding and generation. *arXiv preprint arXiv:2506.02975*, 2025.

[60] Wulin Xie, Yi-Fan Zhang, Chaoyou Fu, Yang Shi, Bingyan Nie, Hongkai Chen, Zhang Zhang, Liang Wang, and Tieniu Tan. Mme-unify: A comprehensive benchmark for unified multimodal understanding and generation models. *arXiv preprint arXiv:2504.03641*, 2025.

[61] Wulin Xie, Yi-Fan Zhang, Chaoyou Fu, Yang Shi, Bingyan Nie, Hongkai Chen, Zhang Zhang, Liang Wang, and Tieniu Tan. Mme-unify: A comprehensive benchmark for unified multimodal understanding and generation models. *arXiv preprint arXiv:2504.03641*, 2025.

[62] Weiwen Xu, Hou Pong Chan, Long Li, Mahani Aljunied, Ruifeng Yuan, Jianyu Wang, Chenghao Xiao, Guizhen Chen, Chaoqun Liu, Zhaodonghui Li, et al. Lingshu: A generalist foundation model for unified multimodal medical understanding and reasoning. *arXiv preprint arXiv:2506.07044*, 2025.

[63] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv e-prints*, pages arXiv–2505, 2025.

[64] Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.

[65] Xinwu Ye, Chengfan Li, Siming Chen, Wei Wei, and Xiangru Tang. Mmscibench: Benchmarking language models on chinese multimodal scientific problems. *arXiv preprint arXiv:2503.01891*, 2025.

[66] Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, et al. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*, 2024.

[67] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

[68] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *CVPR*, pages 9556–9567, 2024.

[69] Lingfei Zeng, Fengdi Che, Xuhan Huang, Fei Ye, Xu Xu, Binhang Yuan, and Jie Fu. Veriequivbench: An equivalence score for ground-truth-free evaluation of formally verifiable code. *arXiv preprint arXiv:2510.06296*, 2025.

[70] Boning Zhang, Chengxi Li, and Kai Fan. Mario eval: Evaluate your math llm with your math llm–a mathematical dataset evaluation toolkit. *arXiv preprint arXiv:2404.13925*, 2024.

[71] Jiaxin Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Fei Yin, Cheng-Lin Liu, and Yashar Moshfeghi. Geoeval: Benchmark for evaluating llms and multi-modal models on geometry problem-solving. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 1258–1276, 2024.

[72] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *ECCV*, pages 169–186. Springer, 2024.

[73] Xinjie Zhang, Jintao Guo, Shanshan Zhao, Minghao Fu, Lunhao Duan, Jiakui Hu, Yong Xien Chng, Guo-Hua Wang, Qing-Guo Chen, Zhao Xu, et al. Unified multimodal understanding and generation models: Advances, challenges, and opportunities. *arXiv preprint arXiv:2505.02567*, 2025.

[74] Yunhang Shen Yulei Qin Mengdan Zhang, Xu Lin Jinrui Yang Xiawu Zheng, Ke Li Xing Sun Yunsheng Wu, Rongrong Ji Chaoyou Fu, and Peixian Chen. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 18, 2021.

[75] Tianshi Zheng, Kelvin Kiu-Wai Tam, Newt Hue-Nam K Nguyen, Baixuan Xu, Zhaowei Wang, Jiayang Cheng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Tianqing Fang, et al. Newtonbench: Benchmarking generalizable scientific law discovery in llm agents. *arXiv preprint arXiv:2510.07172*, 2025.

[76] Minxuan Zhou, Hao Liang, Tianpeng Li, Zhiyu Wu, Mingan Lin, Linzhuang Sun, Yaqi Zhou, Yan Zhang, Xiaoqin Huang, Yicong Chen, et al. Mathscape: Evaluating mllms in multimodal math scenarios through a hierarchical benchmark. *arXiv preprint arXiv:2408.07543*, 2024.

[77] Xianwei Zhuang, Yuxin Xie, Yufan Deng, Liming Liang, Jinghan Ru, Yuguo Yin, and Yuexian Zou. Vargpt: Unified understanding and generation in a visual autoregressive multimodal large language model. *arXiv preprint arXiv:2501.12327*, 2025.

[78] Chengke Zou, Xingang Guo, Rui Yang, Junyu Zhang, Bin Hu, and Huan Zhang. Dynamath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. *arXiv preprint arXiv:2411.00836*, 2024.

Appendix

A Data Curation Prompts

A.1 LLM-i - Reformulation Suitability Screening

To operationalize stage (b) in our pipeline (Figure 3), we employ *GPT-5 [35]* as a binary judge that decides whether a raw problem can be *reformulated into a geometric construction task*. The model returns a single token (true/false) without explanations, enabling high-throughput triage before manual verification. Figure 9 shows the exact prompt template used in our experiments.

LLM-i: Reformulation Suitability Judge

Role. You are a *Geometric Construction Feasibility Judge*. Given a math problem (text + optional diagram), your task is to **only decide** whether the problem is **suitable to be reformulated as a geometric construction problem**. **Output only one word:** true **or** false. Do not output any explanation, punctuation, or spaces.

Criteria
All of the following must hold (otherwise output false):
- **Geometric objects are explicit:** the statement involves planar geometric objects/relations (points, lines, angles, circles, triangles/polygons, parallels/perpendiculars, angle bisectors, midpoints, centers, tangents, intersections, etc.).
- **Convertible into a construction goal:** there is a clear object/position/figure/relation to be *constructed*, not merely a proof or numeric calculation.
- **Reasonable determinacy:** under common assumptions (not to scale; free choice of references), the target has a clearly attainable solution (unique or finitely many), not severely underdetermined.
Output false if the problem is purely algebraic/calculus/probability/statistics/equation solving or analytic calculations with no geometric construction goal.

Output requirements (must be strictly followed)
- Only output: true or false.
- Do not output any other characters, spaces, or line breaks.

Inputs
- Problem text: {PROBLEM_TEXT}
- Problem diagram: {OPTIONAL_DIAGRAM}

Figure 9: Prompt template used by **GPT-5 [35]** in pipeline stage (b) to screen whether a raw problem can be reformulated as a geometric construction task. The template enforces a single-token decision (true/false) with explicit positive criteria and disqualifiers, enabling high-throughput automatic triage prior to human verification.

A.2 LLM-ii - Geometric Problem Generation

At stage (d) of the GGBench pipeline (Figure 3), **LLM-ii** rewrites pre-screened geometry questions into formal *geometric construction problems*. The model transforms general geometry descriptions into construction-oriented formulations that specify given elements, target relations, and construction goals—without revealing solutions or GeoGebra code. It also assigns metadata including difficulty level, construction type, and core geometric skills. Figure 10 shows the prompt template used in this stage (abridged with ellipses for brevity).
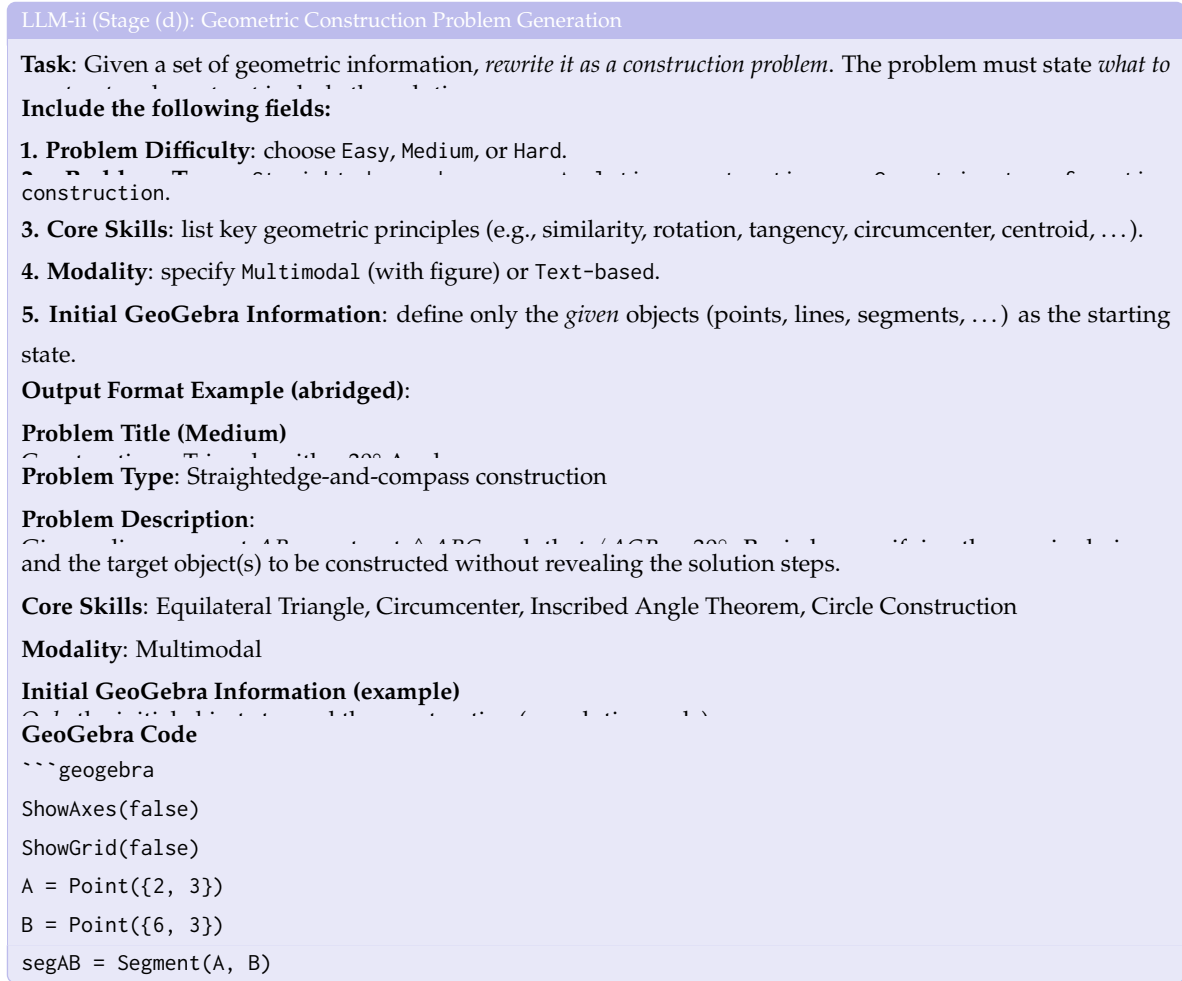
<div style="background-color: #e6e6f5; padding: 10px;">

LLM-ii (Stage (d)): Geometric Construction Problem Generation

Task: Given a set of geometric information, *rewrite it as a construction problem.* The problem must state *what to*

Include the following fields:

1. Problem Difficulty: choose Easy, Medium, or Hard.

construction.

3. Core Skills: list key geometric principles (e.g., similarity, rotation, tangency, circumcenter, centroid, ...).

4. Modality: specify Multimodal (with figure) or Text-based.

5. Initial GeoGebra Information: define only the *given* objects (points, lines, segments, ...) as the starting

state.

Output Format Example (abridged):

Problem Title (Medium)

Problem Type: Straightedge-and-compass construction

Problem Description:

and the target object(s) to be constructed without revealing the solution steps.

Core Skills: Equilateral Triangle, Circumcenter, Inscribed Angle Theorem, Circle Construction

Modality: Multimodal

Initial GeoGebra Information (example)

GeoGebra Code

```geogebra
ShowAxes(false)
ShowGrid(false)
A = Point({2, 3})
B = Point({6, 3})
segAB = Segment(A, B)
```

</div>

Figure 10: Prompt template for **LLM-ii** at stage (d), aligned with the LLM-iii box in layout and syntax.

A.3   LLM-iii - Answer Generation

Based on the adapted problems produced in stage (d), **LLM-iii** generates the final *geometric construction answer* at stage (e) (Figure 3). The model is instructed to output stepwise reasoning aligned with executable GeoGebra code; per-step snapshots are required for visual traceability. Figure 11 shows the prompt template (abridged with ellipses for brevity).

A.4   LLM-iv Prompt for Final Automatic Screening

At stage (f) of the GGBench pipeline (Figure 3), **LLM-iv** acts as the final *automatic construction judge*. It reviews the generated construction tasks and corresponding solutions produced by earlier stages, verifying whether each sample meets all structural, syntactic, and geometric requirements before inclusion in the final dataset. The model outputs a binary score (1 = pass, 0 = fail) together with diagnostic reasoning in JSON format. Figure 12 presents the abridged prompt template used for this purpose.

LLM-iii (Stage (e)): Geometric Construction *Answer* Generation

Task: Given a geometric construction problem, generate a **step-by-step construction answer** including: (1) stepwise construction from primitives (points, lines, circles) to the final figure; (2) *valid GeoGebra code for each step*; (3) *reasoning for each step* (e.g., perpendicular bisector, symmetry, tangency, . . .); (4) per-step snapshots showing figure evolution; (5) clear labels/annotations for all objects. **GeoGebra code requirements** (important): No comments and no blank lines; strictly follow GeoGebra syntax; define objects before use; only official commands; simple, non-self-intersecting polygons; consistent, descriptive names (`PointA`, `LineAB`, . . .). **Constraints & notes** (abridged): Verify commands exist and are valid; initialize points/lines/angles before dependent objects; avoid CW/CCW mistakes; ensure angle/ratio sources are stated; check intersections and angles; use consistent notation; angles displayed should be integers when required; . . . **Checklist**: (i) no unde-

fined objects; (ii) no spelling/syntax errors; (iii) strict GeoGebra compliance. If errors exist, revise. **Output structure**: Numbered steps; each step provides *Method*, *Principle*, and a GeoGebra code block. *Example (abridged)*: **Step 1: Construct auxiliary circles to locate the circumcenter** *Method*: draw c_1 centered at A with radius AB, and c_2 centered at B with radius BA. *Principle*: intersections of c_1 and c_2 are equidistant from A and B. **GeoGebra Code**

```geogebra
ShowAxes(false)
ShowGrid(false)
A = Point({2, 3})
B = Point({6, 3})
c1 = Circle(A, B)

c2 = Circle(B, A)
```

Continue stepwise until completion. Provide a final, single, self-contained GeoGebra code block that runs without errors.

Figure 11: Prompt template for **LLM-iii** at stage (e) in Figure 3, used to generate the complete *geometric construction answer* (stepwise reasoning aligned with executable GeoGebra). Ellipses (. . .) indicate omitted boilerplate for brevity.

A.5 Human Checking & Filtering Standards

At stage (b) of the GGBench pipeline (Figure 3), human annotators perform the first-level screening to ensure that all retained samples are geometrically meaningful, visually grounded, and executable when applicable. This step complements the automated filtering conducted by LLM-i and forms a high-quality foundation for downstream prompt adaptation (stage (d)). The manual filtering process follows three evaluation dimensions as outlined below.

(1) Geometric relevance. Each problem must explicitly present geometric semantics or spatial relationships. Accepted problems typically contain characteristic geometric terminology such as "point," "line," "angle," "circle," "triangle," "parallel," "perpendicular," "radius," or "area." Problems without explicit geometric keywords may still qualify if their content clearly implies geometric reasoning, such as construction, angle relations, or area derivation. In contrast, problems focused purely on algebraic manipulation, logical deduction, or other non-spatial reasoning are excluded.

(2) Image–text correspondence. Each problem must include an accompanying geometric diagram or schematic that accurately reflects the textual conditions and spatial configuration. The image should support reasoning or serve as visual evidence for the described construction. Samples are removed if their images lack geometric characteristics, contradict the textual description, or contain unrelated visual content such as natural photographs, tables, or decorative graphics.

(3) Code executability and consistency. For problems containing executable code (e.g., GeoGebra scripts, or LaTeX drawing commands), annotators manually verify that all code runs correctly in

---

LLM-iv (Stage (f)): Construction Compliance Verification Prompt

Task: Given a geometric construction task, verify whether a model-generated submission satisfies all structural, syntactic, and field-level requirements. The input includes the construction problem,

initial diagram code, and a stepwise solution with corresponding GeoGebra commands.

Evaluation requirements: (1) *Information extraction*: extract title, difficulty, problem type, description, core skills, modality, initial code block, and construction steps; (2) *Problem reasonableness*: target must

be clear; constraints must be sufficient; construction must be geometrically feasible; (3) *Structural and field compliance*: allowed types/modalities only; English-only text; a single initial GeoGebra code

block (no comments or blank lines); no undefined or malformed objects; (4) *Stepwise consistency*: each step must explain "what + why"; commands must match text; logical progression must be

followed.

Scoring rubric: (1) reasonable construction and full compliance. (2) any mandatory rule violated.

Output structure: Provide a brief analysis, followed by a strict binary score:

[Scoring Rationale]: explain format, compliance, and syntax checks (e.g., English-only, valid commands, no undefined objects, no blank lines, consistent use of ShowAxes, ZoomIn, etc.)

[Score]: X point(s)
[JSON]:

```
{"answer_score": [[X]]}
```
End with: "In summary, this submission deserves X point(s)."

Figure 12: Prompt template for **LLM-iv** at stage (f) in Figure 3, used to verify structural integrity, field compliance, and command-level correctness in GeoGebra-based construction scripts. The prompt outputs a strict 0/1 score based on format and execution validity.

standard environments. The generated output must align with the geometric meaning and textual description. Any instance with syntax errors, runtime failures, or inconsistencies between the produced figure and the problem statement is excluded from the dataset.

This threefold manual screening protocol ensures that all retained items exhibit clear geometric validity, accurate multimodal alignment, and functional code integrity—providing a robust and verifiable foundation for subsequent automated evaluation and dataset finalization.

A.6 Expert Validation Standards

At stage (f) of the GGBench pipeline (Figure 3), expert reviewers conduct the final validation to ensure the scientific soundness, structural completeness, and executable reliability of all retained samples. This stage serves as the last human-in-the-loop checkpoint before dataset finalization, guaranteeing that every item meets research-grade quality requirements. The expert validation process is guided by the following five criteria.

(1) Logical rigor of reasoning structure. Experts examine whether each problem demonstrates a complete and coherent reasoning chain that conforms to mathematical logic and geometric principles. Samples with missing premises, invalid inferential steps, or conclusions that cannot be justified by the given conditions are considered invalid.

(2) Consistency and correctness of drawing procedures. Problems involving geometric constructions, function plotting, or visual rendering are manually reviewed to ensure that all drawing instructions

and steps follow established geometric conventions. The generated figures must correspond precisely to the textual descriptions. Samples are rejected if inconsistencies occur between text and graphics, if proportions are distorted, or if essential annotations—such as angles, lengths, or intersection points—are missing.

(3) Accuracy and reproducibility of results. Experts verify that each final outcome, including computed values, geometric figures, or rendered visualizations, is both correct and reproducible. This involves cross-checking the stated results with mathematical reasoning and the actual rendering outputs. Any instance of computational error, logical inconsistency, or semantic mismatch with the problem intent is disqualified.

(4) Clarity and formal correctness of presentation. Each problem is reviewed for linguistic clarity, symbolic correctness, and graphical readability. The use of mathematical symbols and notations must follow formal conventions, and the visual layout must support interpretability in research contexts. Problems containing ambiguous expressions, inconsistent symbols, or irregular formatting are revised or excluded.

(5) Global coherence and research alignment. Finally, experts assess whether the overall logic, visual presentation, and code execution results are semantically aligned. Only problems that demonstrate complete coherence across content, structure, and visualization—and that fall within the intended scope of multimodal geometric reasoning—are retained in the finalized GGBench dataset.

This expert validation phase ensures that the final dataset satisfies high standards of mathematical soundness, visual consistency, and executable reliability, enabling robust benchmarking for future research in multimodal geometric reasoning and construction.

B   Evaluation

B.1   Experimental Setups

To ensure deterministic outputs, we fix the temperature at 0.0 during inferencet that eliminates stochasticity and enforces consistent step-by-step reasoning. All code generation adheres strictly to the GeoGebra command syntax to ensure executability and structural correctness. Inference is parallelized for efficiency. All models are evaluated under the same settings with prompts and pipelines.

B.2   VLM-T Scoring Prompt

Figure 13 illustrates the evaluation prompt used to assess VLM-T performance, one of the core dimensions in GGBench. This component focuses on evaluating the textual reasoning in solving geometric construction problems.

The evaluation process compares a model's generated construction steps against expert-written reference steps, emphasizing logical completeness, geometric correctness, and stepwise coherence. A rubric-based scoring from 1 to 5 is adopted, where higher scores indicate precise and faithful reasoning. To maintain scoring consistency, the evaluator is instructed to ignore stylistic variations and accept logically equivalent strategies that achieve the same geometric objectives. The final score serves as a proxy for the model's ability to translate visual reasoning tasks into accurate symbolic instructions, and directly contributes to the VLM-T metric reported in Table 4.

B.3   VLM-I-Res Scoring Prompt

Figure 14 shows the evaluation template used to obtain VLM-I-Res scores in GGBench.

> **LLM-text-score: Evaluation Prompt for VLM-T**
>
> **Role**: You are an evaluator of the "text steps" for geometric multimodal constructions.
> **Input**:
> - Problem description
> - Reference answer: standard construction steps
> - Model's answer: model's construction steps
>
> **Evaluation Criteria**:
> - **Completeness**: step order is clear; no skipped logic; dependency relations are explicit
> - **Accuracy**: key geometric operations are correctly described
> - **Geometric Consistency**: operations comply with geometric constraints
> - **Reference Equivalence**: alternative but equivalent strategies are acceptable
>
> **Scoring Rubric (1–5)**:
> - 5: Complete, rigorous, and equivalent to the reference answer
> - 4: Minor omissions or differences that do not affect reproducibility
> - 3: Most key points covered but with notable omissions or errors
> - 2: Numerous logical flaws or incoherent reasoning
> - 1: Invalid, unstructured, or irreproducible construction
>
> **Output Requirements**: Return a single Arabic numeral (1–5) only. No text, punctuation, or justification.

Figure 13: Prompt used to compute VLM-T scores. The evaluator assesses textual construction steps on logical, geometric, and structural grounds, returning a scalar score between 1 and 5.

This prompt guides GPT-4o to act as an impartial geometric evaluator that judges the consistency between a model's *final rendered diagram* and the *reference solution*. The design emphasizes geometric reasoning over superficial appearance, ensuring that perceptual similarity does not override structural correctness.

B.4   VLM-I-Mid Scoring Prompt

To evaluate the consistency and correctness of intermediate construction steps, we employ two complementary visual-judging criteria: **Step Accuracy (Step)** and **Process Consistency (Consist.)**. Each is independently scored by a frozen VLM (GPT-4o) using fixed prompts. The final $\text{VLM}\{I_{Mid}$ score is computed as the mean of the two ratings and rescaled to $[0, 100]$.

Prompt for Step Accuracy   Figure 15 presents the evaluation prompt used for measuring Step Accuracy. This metric quantifies the degree of alignment between each textual instruction and its corresponding geometric subfigure in the rendered construction sequence. The prompt guides the judge model to assess whether visual elements—including object naming, positional structure, and geometric relationships—faithfully match the symbolic description at each reasoning step. It ensures that the geometric process remains verifiable and syntactically grounded at the step level.

Prompt for Process Consistency   Figure 16 illustrates the prompt designed for evaluating Process Consistency. This indicator measures whether each step in a geometric construction logically inherits and extends the previous figure, maintaining coherent spatial and structural progression. The prompt enforces sequential reasoning validity, penalizing disjointed or skipped steps, reflecting the model's capacity for stable geometric evolution.

LLM-image-score: Evaluation Prompt for VLM-I-Res

Task: You are a geometric image consistency evaluator.

Input:

- Reference answer image
- Model's final image

Evaluation Criteria:

- Element completeness: whether the key geometric elements required by the problem are included (points, line segments, circles/auxiliary circles, points of tangency, tangents, annotations/labels, etc.).
- Topology & constraints: whether relative positions and geometric relationships (perpendicular, parallel, tangency, concyclicity/collinearity, angle/proportional relationships, etc.) are correct.
- Visual tolerance: do not penalize non-critical style differences such as line width, color, fonts; do not penalize translation/rotation/uniform scaling/mirroring (similarity transforms) unless explicitly forbidden by the problem.
- Overall judgment: prioritize the correctness of geometric topology and constraints; LPIPS is only a reference for perceptual similarity—if there is a conflict, geometric consistency takes precedence.

Scoring Rubric (1–5):

- 5: Elements, topology, and constraints are highly consistent
- 4: Basically consistent, only slight positional/style deviations
- 3: Mostly consistent; elements are complete but there are several minor errors/deviations
- 2: Only partially consistent; key elements are missing or constraint errors are obvious
- 1: Inconsistent with the problem intent or missing key elements (such as circles/auxiliary circles/points of tangency/tangents), or the image is unusable

Boundary Handling:

- Missing/corrupted/unreadable image → 1
- Text only with no image → 1

Output Requirements: Output only a single Arabic numeral (1–5) as the final result, with no other text, punctuation, or spaces.

Figure 14: Prompt used to compute the VLM–I-Res score. The VLM judge compares the model's final rendered diagram against the reference image and assigns a 1–5 rating based on geometric consistency and constraint satisfaction.

B.5 Prompt for Model Inference

To ensure fair and reproducible evaluation across heterogeneous model families, we design two complementary prompt templates corresponding to the two evaluation tracks in GGBench: (1) **LLM/LRM code–based prompting** for models that generate explicit *GeoGebra programs*, and (2) **UMM image–based prompting** for models that directly produce diagrams or visual construction steps. Both

> **VLM-judge: Prompt for Step Accuracy Evaluation (Step)**
>
> **Task**: Given a geometric construction problem and a rendered long image showing the step-by-step solution, judge whether each step image strictly matches its corresponding textual instruction in terms of naming, geometric structure, and positioning.
>
> **Scoring Rules**:
> - **5 points**: The image and text match perfectly, including naming, topological relationships, and quantity, with no extraneous or missing elements.
> - **4 points**: The image and text are mostly consistent, with only very slight visual deviations that do not affect understanding (e.g., a point label is slightly offset but named correctly).
> - **3 points**: The image meets the main structural requirements but has more than one minor error (e.g., confusing names, incorrect position of auxiliary points).
> - **2 points**: The image and text are difficult to correspond one-to-one; there are critical naming errors, connection errors, or missing elements.
> - **1 point**: The overall structure of the image is incorrect, naming is chaotic, it is completely disconnected from the text, or the image is irrelevant/completely incorrect.
>
> **Mandatory Clauses**:
> - If naming errors or omissions make it impossible to uniquely identify the target point/line, the score must not exceed **1 point**.
> - If the text requires connecting a specific pair of points, but the endpoints in the image cannot be reasonably matched (wrong position **and** wrong name), score **1 point**.
> - If a name is slightly misplaced but still identifiable in context, a score of **2 points** may be given.

Figure 15: Prompt used for Step Accuracy (Step) evaluation by the judge model. This criterion assesses alignment between visual steps and symbolic instructions.

prompts are carefully standardized to (i) unify task semantics, (ii) minimize ambiguity in geometric intent, and (iii) encourage interpretable, stepwise reasoning.

Figure 17 shows the template used to elicit explicit *text–to–code* reasoning. Each instance provides a natural–language construction topic and, when available, a reference diagram. Strict code–generation rules are enforced to guarantee syntactic validity and deterministic rendering—e.g., capitalized command names, no underscores, static drawings only, and a single coordinate frame. For ruler–and–compass problems, auxiliary traces must be preserved to make the reasoning chain fully inspectable. An abridged output example is provided in the prompt box of Fig. 17. This design encourages models to *reason, formalize, and construct* in a single pipeline, directly linking linguistic reasoning to executable geometry.

Figure 18 illustrates the template for end-to-end multimodal systems that generate drawings directly from text. These models receive the same construction topic but are instructed to produce only natural–language descriptions of drawing steps without emitting code. The rules emphasize clarity and procedural completeness: each step must describe a concrete geometric operation that would yield the corresponding figure; dynamic or stylistic embellishments are prohibited. If the problem involves ruler-and-compass construction, all traces should be retained in the generated image. This prompt design isolates the model's visual *generative reasoning* ability—its capacity to transform textual geometric constraints into coherent visual sequences—without relying on symbolic code execution.

B.6 Human Evaluation

To verify the reliability and consistency of our automatic evaluation metric (VLM-I), we conducted a systematic human evaluation involving 3 domain experts with backgrounds in mathematics education and geometric modeling. For each model, 100 samples were randomly selected and independently

---

VLM-judge: Prompt for Process Consistency Evaluation (Consist.)

Task: Evaluate whether each step reasonably builds upon the previous step's figure in a cumulative manner, avoiding skipped steps, missing procedures, or logical gaps.

Scoring Rules:

- **5 points**: Completely based on the previous step, structure is fully inherited, and the new construction is correct and complete.
- **4 points**: Mostly coherent, with only minor issues in inheritance clarity or naming deviations.
- **3 points**: The structural relationship between steps is roughly preserved, but there are missing key construction marks, logical skips, or incorrect step order.
- **2 points**: Most constructions are not inherited, or construction continues on an incorrect structure; the logical chain is unclear.
- **1 point**: Steps are severely incoherent, as if they are independent drawings, with no evolutionary relationship visible, or the figures in each step are completely unrelated, forming a broken chain.

Mandatory Clauses:

- If the long image contains only one single figure, the score for this dimension must not exceed **1 point**.
- If intermediate figures are missing or replaced by error messages, the score must be **1 point**.

Figure 16: Prompt used for Process Consistency (Consist.) evaluation by the judge model. This criterion checks visual and logical coherence across sequential construction steps.

scored to ensure representativeness and fairness.

Prior to the evaluation, all experts underwent standardized training to ensure familiarity with the assessment protocol, scoring rubric, and exemplar responses. A double-blind review mechanism was adopted to minimize subjective bias:

- Each sample was independently evaluated by two experts;

- If the score difference exceeded 1 point, a third expert acted as arbiter;

- The final score was the average of the three expert ratings.

Evaluation Criteria. Experts evaluated each model's generation by jointly considering its textual construction steps and the rendered figure. Scores were assigned based on three criteria: logical soundness of the geometric reasoning, accuracy of the drawing process, and overall consistency between text and image. A 1–5 scale was used:

- **5 – Perfect Match**: Both the textual steps and diagram are complete, logically sound, and closely match the reference. All geometric primitives (e.g., circles, auxiliary lines, intersections, tangents) are correctly included and rendered.

- **4 – Mostly Correct**: Minor deviations in position, style, or step detail, but the core reasoning and visual fidelity are intact.

- **3 – Partially Correct**: The key reasoning path is preserved, but notable omissions or inconsistencies exist (e.g., missing constructions, misaligned diagrams, or loose logical steps).

- **2 – Major Errors**: Either the text or diagram deviates significantly from the problem intent, omitting several key steps or introducing logical flaws.

- **1 – Invalid Result**: Severe inconsistency or mismatch between text and diagram, with critical geometric elements missing or misinterpreted.

Model Inference Prompt for LLMs/LRMs

Task:

According to the given drawing topic and image, create a construction using GeoGebra language. If the problem requires a ruler-and-compass construction, retain all auxiliary traces. Output step-by-step drawing instructions, geometric principles, and executable GeoGebra code.

Code Generation Rules:

1. Follow GeoGebra syntax strictly (capitalize commands; no underscores).

2. Use `Point({x, y})` for points and `Vector((x, y))` or `Vector(Point1, Point2)` for vectors.

3. Retain all auxiliary elements in the final figure.

4. Use a single coordinate system; no comments or blank lines.

5. Only static drawings are allowed (no animations).

Output Example:

Problem Title: Construct the incircle of triangle ABC.

Each step includes **Construction**, **Principle**, and **GeoGebra Code** sections.

GeoGebra Code (abridged):

```
ShowAxes(false)
ShowGrid(false)
A = Point({1, 5})
B = Point({0, 1})
C = Point({7, 2})
...
bisA = AngleBisector(C, A, B)
bisB = AngleBisector(A, B, C)
I = Intersect(bisA, bisB)
Circle(I, A)
...
ZoomIn(0, 0, 8, 6)
```

Figure 17: Prompt used for LLMs/LRMs.

Correlation Analysis. To assess alignment between human judgments and automated scores, we computed the Pearson correlation coefficient (r) between VLM-I scores and average human ratings. As shown in Figure 19, the correlation is exceptionally high ($r = 0.9295$), confirming that our automated metric closely reflects human-perceived quality and faithfully captures model performance trends.

C Case Study

C.1 Examples across Difficulty Levels

To illustrate the range of geometric reasoning tasks in GGBench, we present three representative examples drawn from the benchmark, each aligned with a specific difficulty level. Figures 20–22 visualize problems designed to probe distinct levels of spatial abstraction, procedural depth, and symbolic control.

Figure 20 corresponds to an entry-level task grounded in elementary circle geometry. The problem involves constructing chords that pass through given points while remaining parallel to two specified diameters. Despite its simplicity, solving the problem requires maintaining directional consistency, identifying appropriate auxiliary constructions, and accurately positioning endpoints within a bounded coordinate frame. It targets the model's ability to handle localized spatial constraints and execute basic primitives such as 'Segment', 'ParallelLine', and 'Point'.

Model Inference Prompt for UMMs

Task:
According to the given drawing topic, create a drawing that meets the requirements. If the problem requires the use of a ruler and compass construction, please retain the traces of ruler and compass construction. Finally, only output the drawing steps.

Drawing Step Generation Rules:

1. Strictly follow the description of the drawing steps and clearly explain each step.
2. The output should only include detailed step descriptions, without any code.
3. The steps should describe the drawing process, ensuring that each description leads to the creation of the corresponding image.
4. The steps should be clear and concise, avoiding dynamic effects or code.

Output Example:
Problem Title: Given two circles of different radii c_1 (center O_1, radius r_1) and c_2 (center O_2, radius r_2), which do not intersect. Using only ruler and compass construction, draw the two external tangents to these two circles.

Drawing (abridged):

1. Construct the angle bisector of ($\angle$ BAC).
2. Construct the angle bisector of ($\angle$ ABC).

Principle: The incenter of a triangle (the center of the incircle) is the intersection of its three angle bisectors. According to the angle bisector theorem, any point on an angle bisector is equidistant from the two sides of the angle. Therefore, the intersection of the two angle bisectors will be equidistant from the three sides of the triangle.

Figure 18: Prompt used for UMMs (e.g., Janus, Bagel, Qwen-Image, Nano Banana, Seeddream).

Figure 21 illustrates a transformation-based task involving a regular pentagon and an inscribed triangle. The objective is to rotate the triangle about one vertex and identify the resulting intersections with the original figure. This setup tests the model's understanding of rigid motions, label consistency, and intersection logic. It requires chaining multiple dependencies while preserving geometric invariants such as distance and angle, and it penalizes any deviation from transformation semantics or object reuse policies.

Figure 22 represents a high-complexity construction spanning multiple geometric phases. Starting from a tangent circle, the problem unfolds into a sequence of regular polygon constructions—inscribing a square, then an octagon, followed by a 16-gon. The solution involves hierarchical decomposition, recursive angle bisection, and repeated application of symmetric placement rules. Success depends on long-horizon reasoning, internal consistency across stages, and robustness in symbolic command generation under rigid geometric constraints.
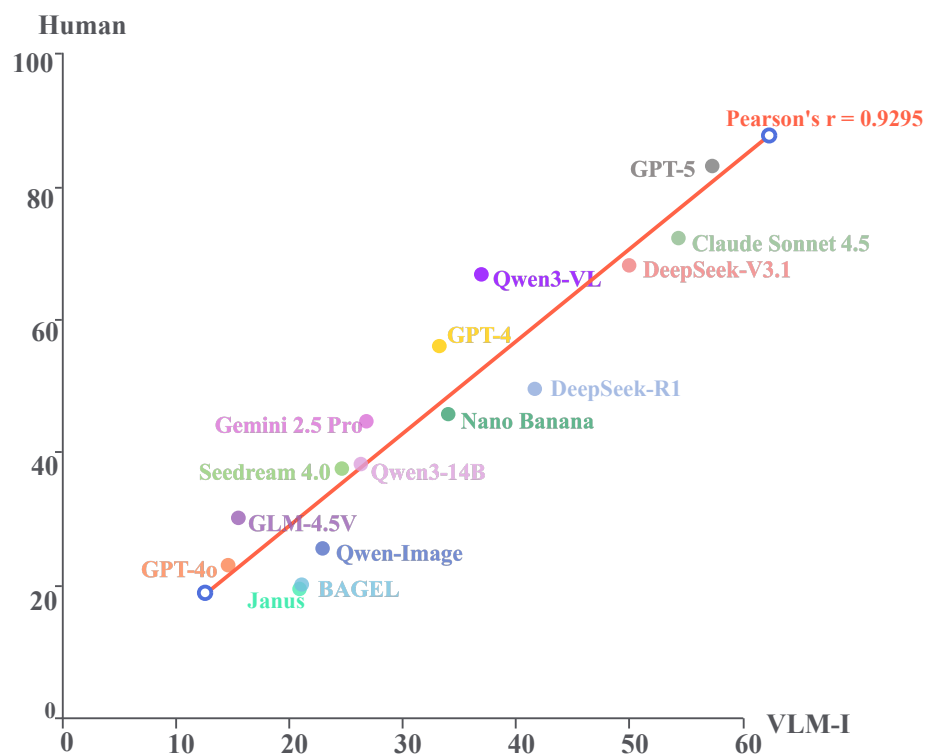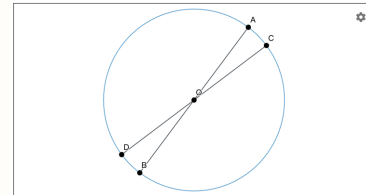
Figure 19: Correlation between VLM-I and human evaluation.

Difficulty: Easy

Involved Concepts: Circle Properties&Constructions;
Basic Constructions
Problem:
Given a circle with center O and two diameters AB and CD, construct a chord FD passing through D and parallel to diameter AB. Also, construct a chord BE passing through B and parallel to diameter CD.

Solution:
Step 1: Construct chord FD parallel to diameter AB
Method:
1.Draw an auxiliary line through point D parallel to the diameter AB.
2.This line intersects the circle at D and another point F; locate this second intersection.
3.Connect F and D to form the required chord FD.
Principle:*A parallel line through a given point defines a chord whose endpoints lie on the circle.*

Step 2: Construct chord BE parallel to diameter CD
Method:
1.Draw an auxiliary line through point B parallel to the diameter CD.
2.This line intersects the circle at B and another point E; locate this second intersection.
3.Connect B and E to form the required chord BE, completing the construction.
Principle:*Following the same reasoning, the line through B parallel to diameter CD determines chord BE that meets the given conditions. The final figure shows both chords and their corresponding auxiliary lines.*

Figure 20: Example of an easy-level task.

Difficulty: Medium

Involved Concepts:
Geometric Transformations;Polygon Properties & Constructions
Problem:
Given a pentagon ABCDE, rotate the triangular section ADE around vertex A by an angle of -60° (clockwise) to obtain the triangle AE'D' Construct the resulting composite figure and find the intersection point F of the original side CD and the new side E'D'.

Solution:
Step 1: Rotate vertices D and E around point A
Method:
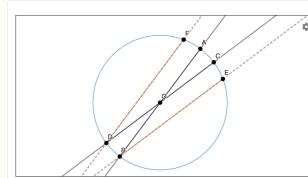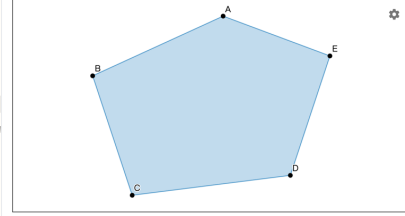*1.Define the rotation angle α = −60°.*
2.Rotate points D and E around A to obtain D' and E'.
3.Keep ABC fixed; mark the original D and E as hollow points, and draw AD and AE with dashed lines to show the initial state.
Principle: Rotation is a rigid transformation preserving distances from the center (e.g., AD = AD'); all points rotate by the same angle.

Step 2: Construct the composite figure
Method:
1.Draw the fixed part of the original figure, which is the quadrilateral ABCD, as a filled polygon.
2.Draw the rotated triangle AE'D' as another filled polygon to represent the new position of the transformed part.
Principle: The final figure is a composition of a static part (ABCD) and a dynamically transformed part (AE'D'). Representing them as distinct polygons helps visualize the result of the transformation.

Step 3: Construct the intersection point F
Method:
Define an auxiliary line lineCD passing through points C and D.
Define another auxiliary line lineEprimeDprime passing through the rotated points E' and D'.
Use the Intersect command to find the common point F of these two lines.
Principle: Two distinct, non-parallel lines in a plane intersect at exactly one point. This point F lies on both the boundary of the fixed part and the boundary of the rotated part.

Step 4: Add final annotations
Method:
Add a circular arc using the Angle command to visually represent the 60° rotation from E to E' around A.
Ensure all key points (A, B, C, D, E, D', E', F) are clearly labeled and styled for readability.
Adjust the final view to encompass the entire construction.
Principle: Annotations and clear labeling are crucial for communicating the steps and results of a geometric construction, making the relationships between elements explicit.

Figure 21: Example of a medium-level task.

Difficulty: Hard

Involved Concepts: Circle Properties &Constructions;
Basic Constructions;Polygon Properties & Constructions

Problem: Given a horizontal line `l` and a point `A` on it, construct a circle with radius 3 that is tangent to `l` at `A`. Then, inscribe a regular 16-gon in this circle, using `A` as one of its vertices.

Solution:
Step 1: Construct the tangent circle
Method:
1.Construct a line perpLine perpendicular to the given line l at point A. The center of the tangent circle must lie on this line.
2.To place the center O at a distance of 3 units (the radius) from A, draw a circle centered at A with radius 3.
3.The intersection of perpLine and this new circle gives the center O.
4.Draw the final circle c with center O and radius 3.
Principle: The radius of a circle is perpendicular to the tangent line at the point of tangency.

Step 2: Construct the vertices of an inscribed square
Method:
1.The point of tangency A is the first vertex of our polygon, V1.
2.Find the point V9 diametrically opposite to V1 by intersecting perpLine with circle c.
3.Construct a new line perpDiameter passing through the center O and perpendicular to the line segment V1V9.
4.The intersection points of perpDiameter with circle c give two more vertices, V5 and V13. The points V1, V5, V9, V13 form an inscribed square.
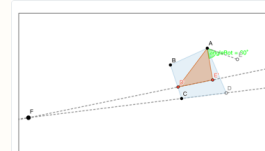*Principle: The diagonals of a square are perpendicular bisectors of each other. This construction divides the circle into four 90° arcs.*

Step 3: Construct the vertices of an inscribed octagon
Method:
1.Bisect the four 90°central angles formed by the vertices of the square (V1OV5, V5OV9, etc.).
2.The intersection of each angle bisector with the circle c yields a new vertex.
3.This process creates vertices V3, V7, V11, and V15, which,together with the previous vertices, form a regular octagon.
Principle: Bisecting the central angles of a regular n-goninscribed in a circle results in the vertices of a regular 2n-gon.

Step 4: Construct the final vertices of the 16-gon
Method:
1.Repeat the bisection process on the 45°central angles of the octagon.
2.This final set of bisections creates the remaining 8 vertices: V2, V4, V6,
V8, V10, V12, V14, and V16.
3.All 16 vertices for the regular 16-gon are now defined.
Principle: This is the second application of the principle that bisecting the central angles of a regular n-gon produces a regular 2n-gon. Here, we go from an 8-gon to a 16-gon.

Step 5: Draw the regular 16-gon and finalize the figure
Method:
1.Connect the 16 vertices (V1 through V16) in sequential order to form the regular 16-gon.
2.Adjust the colors and styles to highlight the final polygon and ensure the construction is clear and aesthetically pleasing.
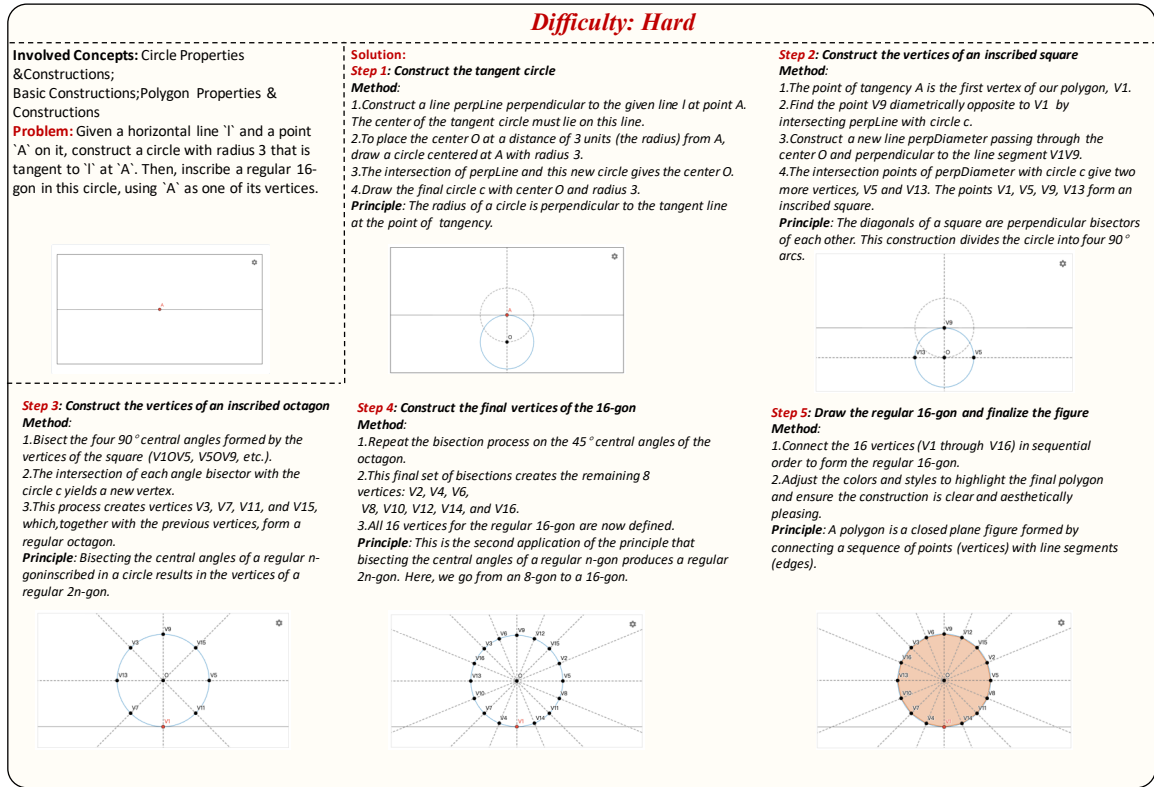*Principle: A polygon is a closed plane figure formed by connecting a sequence of points (vertices) with line segments (edges).*

Figure 22: Example of a hard-level task.