When Does Auxiliary Modality Matter in Solving Geometric Problems?
A Comprehensive Study of Textual, Formal, and Visual Modalities

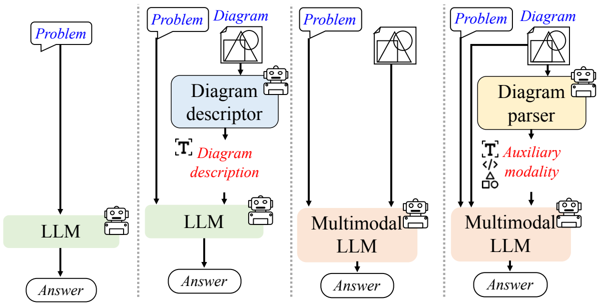
Hyuk Namgoong, Jeesu Jung, Yerim Han and Sangkeun Jung[*]
Department of Computer Science & Engineering,
Chungnam National University, Republic of Korea
`{hyuk199, jisu.jung5, namu.rim2, hugmanskj}@gmail.com`

Abstract

Large Language Models (LLMs) face challenges in integrating linguistic and spatial reasoning, which limits their performance on geometry problems. While prior work has attempted to bridge this gap using diagram parsers with multimodal models, a systematic comparison of how various auxiliary modalities and their combinations affect performance has been lacking. To address this, we present a systematic study of four auxiliary modalities—formal diagram facts (CDL), natural language representations ($\mathrm{T}_{\mathrm{CDL}}$), diagram descriptions (DES), and image augmentations (IMG)—on a range of open- and closed-source multimodal LLMs. Our analysis reveals a compelling dichotomy in the effectiveness of these modalities. While formal representations like CDL and $\mathrm{T}_{\mathrm{CDL}}$ offer a modest performance lift, diagram descriptions (DES) cause a dramatic split: they significantly boost the accuracy of open-source LLMs which often struggle with visual parsing, while often misleading more capable closed-source models and causing a performance drop. This highlights a critical trade-off between augmenting input with helpful information and introducing misleading noise, demonstrating that the efficacy of auxiliary modalities is heavily dependent on the inherent capabilities of the underlying model.

1 Introduction

Large Language Models (LLMs) have shown strong reasoning performance on mathematical benchmarks. Prompting methods like few-shot and chain-of-thought (CoT) (Brown et al., 2020; Wei et al., 2023) further boost these capabilities. However, LLMs often face performance limits on benchmarks such as GSM8K (Cobbe et al., 2021) without external support like additional information or verification. This gap widens in **geometry**, where linguistic and spatial reasoning must be integrated.



(a) LLM-Based Mathematics Solver (b) Descriptor-LLM Geometry Solver (c) General Multimodal LLM Solver (d) Diagram Parser-Multimodal LLM Solver

Figure 1: Comparison of four geometry-solving pipelines. (a) A text-only LLM solves the problem using only the natural-language statement. (b) A descriptor listing every diagram element and relation provides textual input to the LLM. (c) A multimodal LLM processes the statement and the raw diagram image. (d) A parser-extracted auxiliary modality supplies multimodal LLM.

In general, geometry problems combine natural language descriptions and diagrams as input for a solver LLM. This requires finding reference (e.g., AB bisects $\angle CAD$) from both visual and linguistic inputs. As Figure 1a illustrates, *text-only* LLMs inherently lack visual context. Efforts to mitigate this, such as connecting diagram descriptors to NLP pipelines (Seo et al., 2015) as shown in Figure 1b, have been attempted. However, these systems often mirrored the limitations of the diagram descriptors themselves, frequently missing information or misinterpreting fine-grained details (Hao et al., 2022). While employing a diagram parser with existing multimodal models, as shown in Figure 1d (Zhang et al., 2022, 2025), has advanced geometry problem-solving.

However, effectively integrating visual, textual, and formal modalities remains a challenge. To address this, we augment geometry problems with four *auxiliary modalities*:

- **CDL** (condition declaration language): a formal

---

[*] Corresponding author

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 2: Short Papers, pages 76–92
March 24-29, 2026 ©2026 Association for Computational Linguistics

language that lists every geometric element and its explicit relations (Zhang et al., 2024b).

- **T$_{\text{CDL}}$**: a natural language representation derived from CDL.
- **DES** (Diagram description): an exhaustive caption enumerating all entities and relations.
- **IMG** (Image augmentations): additional images generated from the original diagram, providing visual analogies.

These modalities encompass formal, linguistic, and pictorial signals that an LLM might otherwise infer.. However, this field lacks a direct comparison of how each modality and their combinations affect performance, a gap we address in this study.

Our analysis reveals a compelling dichotomy in multimodal learning effectiveness across different model architectures. While formal diagram representations such as CDL and T$_{\text{!CDL}}$ offer a modest performance lift, DES causes a dramatic split. It significantly boosts open-source LLMs, which struggle to parse diagrams, by up to +15.0% accuracy. Conversely, DES often misleads more capable closed LLMs, causing performance to drop by as much as -6.8%. This highlights a critical trade-off between providing helpful information and introducing misleading noise, which depends heavily on the inherent capabilities of models.

2 Related Works

Datasets for Geometric Problems For evaluating general models, multimodal datasets featuring diagrams and fine-grained annotations have been utilized (Cao and Xiao, 2022; Hao et al., 2022). More recently, datasets such as GEOEVAL (Zhang et al., 2024a) and PGPS9K (Zhang and Moshfeghi, 2024) have emerged to assess multimodal LLMs. These datasets offer diverse problem types and difficulties, covering planar, solid, and analytic geometry problems.

Multimodal LLMs for Geometry Multimodal LLMs benefit from explicit reasoning prompts. CoT (Wei et al., 2023) enhances mathematical accuracy, and integrated frameworks like GeoGen and GeoLogic (Pan et al., 2025) improve performance on geometric problem benchmarks. However, these models still struggle with diagram misinterpretations and hallucinations.

Enhancing Multimodal LLMs with Auxiliary Geometric Information Recent research integrates auxiliary inputs into LLMs to explicitly represent geometric structures and reduce reasoning errors. Early systems (Seo et al., 2015) converted diagram primitives to symbolic clauses. More recent models parse diagrams into graph representations through vision pipelines (Zhang et al., 2022). FORMALGEO (Zhang et al., 2024b) introduced CDL for formal geometric representation, which (Zhu et al., 2025) leveraged for domain-specific logic. GeoSCN (Hossain et al., 2025) further enhances spatial reasoning by injecting geometric features into multimodal attention layers. These approaches highlight the effectiveness of structured representations in improving LLM geometric reasoning.

In our work, we present the first systematic analysis of four auxiliary modalities—CDL, T_{CDL}, diagram descriptions, and image augmentations.

3 Auxiliary-Modality Analysis

3.1 Definitions of Auxiliary-Modality

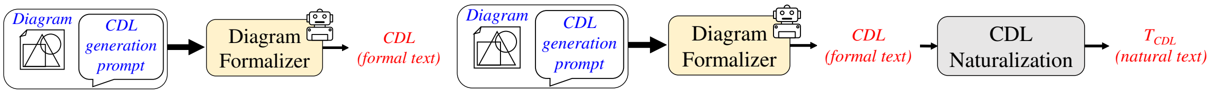
We evaluate four *auxiliary modalities* and their construction pipelines.

CDL A formal language representing the diagram as a graph of points, lines, circles, and relations (e.g., `Shape(AO,OD,DA)`, `Shape(DC,CE,ED)`, `Collinear(AOC)`). We retain only the subgraph covering *collinearity* and *cocircularity*, extracted by the Diagram Formalizer of (Zhang et al., 2025). The Diagram Formalizer takes the original diagram and a CDL type-specifying prompt as input, and outputs the corresponding CDL, as shown in Figure 2a. In our experiments, as illustrated in Figure 4a, we generate both the construction CDL and its calibrated variant, employing the calibrated CDL in our evaluations.

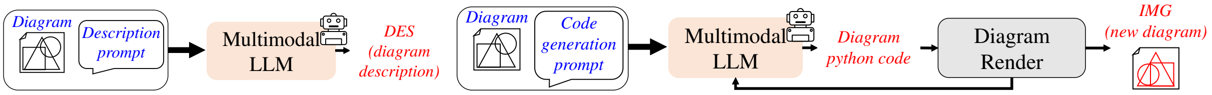
T$_{\text{CDL}}$ A natural-language rendering of construction CDL, as shown in Figure 2b. CDL naturalization performs the following tasks:

- *Shape* definitions are mapped to “triangle” or “quadrilateral” based on the number of sides and points; otherwise, to “shape.”
- *Collinearity* is expressed as “Point `[point]` lies on the same line.”
- *Cocircularity* is expressed as “Point `[point]` lies on a circle centered at point `center`.”

DES A textual description generated by guiding a multimodal LLM to narrate all visible diagram elements and relations, as shown in Figure 2c. Using the prompt in Figure 6, the model examines the diagram and produces a natural-language explanation of all visible components (points, lines, angles, shapes) and relationships such as parallelism, perpendicularity, lengths, and symmetry.



(a) CDL generation using Diagram Formalizer (b) $\mathrm{T_{CDL}}$ generation via natural-language conversion with Diagram Formalizer



(c) Diagram description generation using a multimodal LLM (d) Additional diagram generation using a multimodal LLM. If rendering fails, the multimodal LLM retries generation.

Figure 2: Our auxiliary modality generation pipeline takes a diagram and a prompt as input, producing the Auxiliary Modality (shown in red). Below is an overview of the four generation modes: (a) CDL Generation: Achieved using the Diagram Formalizer. (b) $\mathrm{T_{CDL}}$ Generation: Performed via natural-language conversion. (c) Diagram Description Generation: Generated with a multimodal LLM. (d) Additional Diagram Generation: Produced by a multimodal LLM. The robot icon denotes the multimodal LLM, while the gray rounded box indicates the algorithm executor.

IMG Image augmentations, as shown in Figure 2d, where visually similar diagrams are synthesized. Using the prompt in Figure 5, the LLM is instructed to generate Python scripts that reproduce the original geometry under diverse transformations (e.g., rotation, flipping, affine transforms) while preserving all connections and labels. If an error occurs, the model is provided with the error message and faulty code and tasked to correct it.

Thus, CDL and $\mathrm{T_{CDL}}$ supply formal or textual structure, while DES and IMG provide automatically generated visual context. DES and IMG are generated using GPT-o4-mini (OpenAI, 2025).

3.2 Dataset and Model Settings

We evaluate all modality combinations on the FormalGeo7K (Zhang et al., 2024b) and Geometry3K (Lu et al., 2021) test sets, selecting them for comparative analysis. FormalGeo7K embeds comprehensive problem information directly within its textual statements, whereas Geometry3K provides a more simplified representation, thus serving as an effective contrast for evaluation.

We used closed LLMs including GPT-o4-mini (2025-04-16) and Gemini 2.5-flash (DeepMind, 2025), evaluated with and without its *'think'* reasoning option. For open-source multimodal capabilities, we utilized Qwen-2.5-VL (72B-Instruct) (Bai et al., 2025), Llama-4-Scout (17B-16E–Instruct) (AI, 2025) and Gemma-3 (27b-it) (Team, 2025).

4 Experiment Analysis Result

As shown in Figure 1d, we experimented with auxiliary modalities both individually and combined. We compared their average accuracy against a baseline model that receives only the problem statement and diagram, as shown in Figure 1c. We also analyzed performance differences between open and closed LLMs. Table 1 provides a numerical summary of all modality combination results, with detailed settings in Appendix A.

4.1 Auxiliary Modalities

Auxiliary modalities enhance the performance of the base model. Figure 1d shows that CDL and TCDL have a positive impact on FormalGeo7K. TCDL showed a small but significant effect, with a maximum increase of +3.6% on FormalGeo7K and +2.1% on Geometry3K. However, the performance drop in some models using $\mathrm{T_{CDL}}$ suggests that they vary in their ability to leverage formal diagram input versus natural-language rendering. The significant difference between open and closed LLMs regarding DES will be discussed later.

Conversely, IMG has a very negligible effect compared to other auxiliary modalities. Table 3 shows, IMG results in a similar mixture of performance gains and losses across all LLMs. The performance variance is notably low at +0.23% on Geometry3K and +0.22% on FormalGeo7K, which is smaller than that of other auxiliary modalities.

4.2 Open vs. Closed LLM

A clear performance difference emerges between open and closed LLMs based on DES. Open LLMs often misinterpret information from a diagram, while closed LLMs demonstrate a superior understanding of visual data, leading to higher baseline performance. The following discussion will therefore focus on the results obtained with and without DES, with detailed examples from Qwen-2.5-VL and GPT-o4-mini provided in Appendix E.

Open LLMs, which have lower baseline performance, show significant improvement when DES

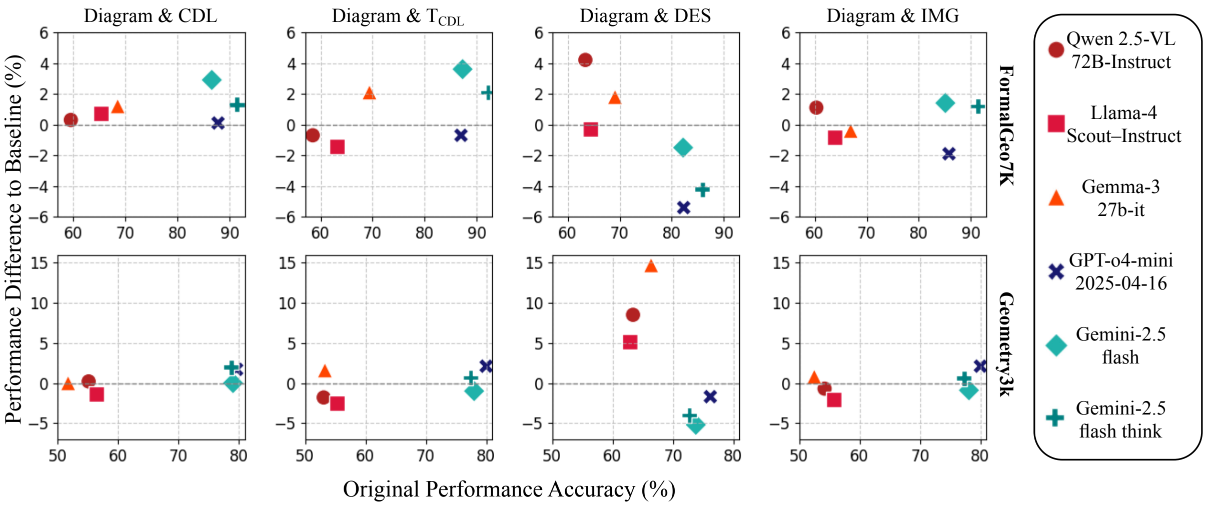


Figure 3: Performance impact of various auxiliary modalities when added to a baseline input consisting of a diagram and a problem description. The experiments are conducted on the FormalGeo7K and Geometry3K test datasets, with each row of charts corresponding to one of the datasets. Each column shows the result of adding a single type of auxiliary modality. The markers represent various models. The x-axis plots the accuracy achieved with the auxiliary modality, while the y-axis shows the performance difference from the baseline.

is added. On Geometry3K, where problem information is simplified, they achieved up to a $+15.0\%$ accuracy increase. This is likely due to their difficulty in extracting key information from diagrams.

Conversely, closed LLMs show a performance drop with DES. On FormalGeo7K, they experienced up to a -6.8% accuracy decrease, as problem information is already present in the text. This suggests DES can contain factual errors (e.g., incorrect angles or lengths) that mislead these LLMs.

4.3 Combined Modalities

Combining multiple auxiliary modalities shows similar trends to using them individually. The combination of CDL and $\mathrm{T}_{\mathrm{CDL}}$ shows a slight improvement over using them separately, while combining DES with other modalities results in a significant change in performance.

For open LLMs, the inclusion of DES with other multiple auxiliary modalities generally leads to an improvement compared to using DES alone. Conversely, closed LLMs show a performance drop when DES is included in a combination. While Qwen-2.5-VL shows a performance increase of +1.1% on FormalGeo7K, a decrease of up to -1.6% on Geometry3K also occurs. Furthermore, for closed LLMs, the combination of CDL and $\mathrm{T}_{\mathrm{CDL}}$ can sometimes mitigate the performance drop caused by DES, as seen on FormalGeo7K where the decrease was reduced -4.2% to -1.9%.

4.4 Discussion

Our experimental results indicate that:

- Auxiliary modalities generally enhance performance, although a performance drop is observed in some models.
- DES has contrasting effects, significantly boosting open LLM performance while often degrading closed LLM performance due to factual errors that can mislead more capable models.
- CDL and $\mathrm{T}_{\mathrm{CDL}}$ consistently provide a positive impact, improving performance and mitigating the drop caused by DES in closed LLMs.
- Combining modalities follows individual trends, with open LLMs benefiting from DES combinations while closed LLMs use other modalities to mitigate the DES-induced performance drop.

5 Conclusion

We have presented a systematic study of four auxiliary modalities for enhancing geometry problem solving with multimodal LLM. Our findings indicate that auxiliary modalities generally enhance performance, but we observed a performance drop in some models. A key observation was the contrasting effect of DES, which provided a significant performance boost for open-source LLMs but frequently degraded the performance of closed-source models, likely due to potential factual errors within the descriptions. Building on these results, we propose two main directions for future research: first, exploring gating mechanisms for high-capacity models to enable selective use of auxiliary information; and second, extending formal modalities like CDL to represent a richer set of geometric relations.

Limitations

Our work has several limitations. The data spectrum of our experiments is limited to planar geometry problems, which restricts the generalizability of our findings to more complex geometric domains. Furthermore, our reliance on a single metric of final answer accuracy fails to comprehensively assess the quality of the model's reasoning process or the correctness of its intermediate steps. Future work should address these limitations by exploring a broader range of geometric problems and developing more nuanced evaluation metrics.

We leverage GPT-generated DES and IMG without systematic human evaluation or strict quality controls. As a result, the impact of noisy or incorrect auxiliary inputs—and potential mitigation strategies—remains underexplored. However, because our goal was not to produce optimal DES or IMG but to assess their utility in a realistic setting, these imperfections are aligned with our experimental objectives.

Acknowledgments

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(No. RS-2025-0055621731482092640101), Institute of Information and Communications Technology Planning and Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190004, Development of semi-supervised learning language intelligence technology and Korean tutoring service for foreigners), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2022-00155857, Artificial Intelligence Convergence Innovation Human Resources Development (Chungnam National University)) and research fund of Chungnam National University.

References

Meta AI. 2025. The llama 4 herd: Launching *Llama 4 Scout*. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`. Blog post. Accessed 18 May 2025.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Jie Cao and Jing Xiao. 2022. An augmented benchmark dataset for geometric question answering through dual parallel text encoding. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1511–1520, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Google DeepMind. 2025. Gemini 2.5: Our most intelligent ai model. `https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/`. Blog post. Accessed 18 May 2025.

Google. 2025. Gemini 2.0: Flash, flash-lite, pro. `https://developers.googleblog.com/en/gemini-2-family-expands/`. Accessed: 2025-05-20.

Yihan Hao, Mingliang Zhang, Fei Yin, and Linlin Huang. 2022. Pgdp5k: A diagram parsing dataset for plane geometry problems. *Preprint*, arXiv:2205.09947.

Md.Shamim Hossain, Shamima Aktar, Naijie Gu, Weiyong Liu, and Zhangjin Huang. 2025. GeoSCN: A novel multimodal self-attention to integrate geometric information on spatial–channel network for fine-grained image captioning. *Expert Systems with Applications*, 272:126692. `https://www.sciencedirect.com/science/article/pii/S0957417425003148`.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. 2021. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In *The 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

OpenAI. 2025. Gpt-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Blog post. Accessed 18 May 2025.

Yicheng Pan, Zhenrong Zhang, Pengfei Hu, Jiefeng Ma, Jun Du, Jianshu Zhang, Quan Liu, Jianqing Gao, and Feng Ma. 2025. Enhancing the geometric problem-solving ability of multimodal llms via symbolic-neural integration. *Preprint*, arXiv:2504.12773.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1476, Lisbon, Portugal. Association for Computational Linguistics.

Gemma Team. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Jiaxin Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Fei Yin, Cheng-Lin Liu, and Yashar Moshfeghi. 2024a. 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, Bangkok, Thailand. Association for Computational Linguistics.

Jiaxin Zhang and Yashar Moshfeghi. 2024. Gold: Geometry problem solver with natural language description. *Preprint*, arXiv:2405.00494.

Ming-Liang Zhang, Fei Yin, Yi-Han Hao, and Cheng-Lin Liu. 2022. Plane geometry diagram parsing. *Preprint*, arXiv:2205.09363.

Xiaokai Zhang, Na Zhu, Yiming He, Jia Zou, Qike Huang, Xiaoxiao Jin, Yanjun Guo, Chenyang Mao, Yang Li, Zhe Zhu, Dengfeng Yue, Fangzhen Zhu, Yifan Wang, Yiwen Huang, Runan Wang, Cheng Qin, Zhenbing Zeng, Shaorong Xie, Xiangfeng Luo, and Tuo Leng. 2024b. Formalgeo: An extensible formalized framework for olympiad geometric problem solving. *Preprint*, arXiv:2310.18021.

Zeren Zhang, Jo-Ku Cheng, Jingyang Deng, Lu Tian, Jinwen Ma, Ziran Qin, Xiaokai Zhang, Na Zhu, and Tuo Leng. 2025. Diagram formalization enhanced multi-modal geometry problem solver. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Na Zhu, Xiaokai Zhang, Qike Huang, Fangzhen Zhu, Zhenbing Zeng, and Tuo Leng. 2025. Fgeo-parser: Autoformalization and solution of plane geometric problems. *Symmetry*, 17(1).

A Experiments Setting

Dataset Setting FormalGeo7K is an Olympiad-style geometry dataset comprising geometric diagrams, problem statements, and answers, all annotated in Conditional Declaration Language (CDL). In our experiments, we follow the original split and reserve 15% of the data (1,050 instances) as a held-out test set.

Performance Setting We compute accuracy by comparing each reference answer in FormalGeo7K with the corresponding output from our multimodal LLM. As shown in Figure 9, the model's response follows the template "The final answer is: boxed{{your_answer}}". We extract the value inside the box and perform a direct match against the ground truth. Any mismatches are subjected to a secondary verification step using the Gemini-2.0-flash-lite (Google, 2025) and the prompt in Figure 7. To account for equivalent but syntactically different representations (e.g., fractions vs. decimals or alternate algebraic forms), we apply normalization and equivalence checking before final evaluation.

Environment Setting We accessed transformer models via the OpenAI API, Google Cloud packages and Openrouter API[1], while all other open-source LLMs were executed on Nvidia A6000 GPUs. Performance metrics were averaged over five independent experimental runs.

B Prompt Configurations

In this appendix, we detail the prompts used throughout our pipeline. Except for the mathematical equivalence verification prompt (Fig. 7) and the code–error retry prompt (Fig. 5b), each prompt begins by providing the geometric diagram as input:

First, the Diagram Formalizer is invoked with the diagram to produce the construction CDL (Fig. 4a), and its output is shown in Fig. 4b. Next, the multimodal LLM receives the diagram along with a request to generate Python code for a transformed variant (Fig. 5) or to produce a concise natural-language description of the diagram's elements and relations (Fig. 6). Finally, the unified solving prompt (Fig. 8) feeds the diagram and selected auxiliary modalities to the model to elicit a boxed answer.

Dataset	Data Type				Opened LLM			Closed LLM		
	CDL	T_{CDL}	DES	IMG	Qwen 2.5-VL 72B-Instruct	Llama-4 Scout–Instruct	Gemma-3 27b-it	GPT-o4-mini 2025-04-16	Gemini-2.5 flash	Gemini-2.5 flash think
FormalGeo7K	Baseline				59.2	64.6	67.3	87.7	83.7	90.1
	O				59.5 (↑0.3)	65.3 (↑0.7)	68.5 (↑1.2)	87.8 (↑0.1)	86.6 (↑2.9)	91.4 (↑1.3)
		O			58.5 (↓0.7)	63.2 (↓1.4)	69.4 (↑2.1)	87.0 (↓0.7)	87.3 (↑3.6)	92.2 (↑2.1)
			O		63.4 (↑4.2)	64.3 (↓0.3)	69.1 (↑1.8)	82.3 (↓5.4)	82.2 (↓1.5)	85.9 (↓4.2)
				O	60.3 (↑1.1)	63.8 (↓0.8)	66.9 (↓0.4)	85.8 (↓1.9)	85.1 (↑1.4)	91.3 (↑1.2)
	O	O			60.0 (↑0.8)	64.8 (↑0.2)	69.4 (↑2.1)	86.1 (↓1.6)	87.4 (↑3.7)	91.0 (↑0.9)
	O		O		61.6 (↑2.4)	64.1 (↓0.5)	67.5 (↑0.2)	82.2 (↓5.5)	81.8 (↓1.9)	86.7 (↓3.4)
	O			O	57.8 (↓1.4)	64.0 (↓0.6)	69.0 (↑1.7)	85.9 (↓1.8)	86.7 (↑3.0)	92.4 (↑2.3)
		O	O		63.3 (↑4.1)	65.2 (↑0.6)	69.1 (↑1.8)	83.0 (↓4.7)	82.6 (↓1.1)	87.9 (↓2.2)
		O		O	59.5 (↑0.3)	64.7 (↑0.1)	68.8 (↑1.5)	85.9 (↓1.8)	87.3 (↑3.6)	93.8 (↑3.7)
			O	O	63.5 (↑4.3)	64.3 (↓0.3)	68.4 (↑1.1)	80.9 (↓6.8)	81.6 (↓2.1)	87.9 (↓2.2)
	O	O	O		63.1 (↑3.9)	63.5 (↓1.1)	68.4 (↑1.1)	82.9 (↓4.8)	82.8 (↓0.9)	87.2 (↓2.9)
	O	O		O	58.1 (↓1.1)	64.3 (↓0.3)	70.6 (↑3.3)	87.0 (↓0.7)	87.6 (↑3.9)	91.3 (↑1.2)
	O		O	O	63.4 (↑4.2)	63.4 (↓1.2)	68.3 (↑1.0)	82.3 (↓5.4)	81.0 (↓2.7)	88.1 (↓2.0)
		O	O	O	64.5 (↑5.3)	64.4 (↓0.2)	69.8 (↑2.5)	83.4 (↓4.3)	83.0 (↓0.7)	89.3 (↓0.8)
	O	O	O	O	64.2 (↑5.0)	64.7 (↑0.1)	68.0 (↑0.7)	83.7 (↓4.0)	82.7 (↓1.0)	88.2 (↓1.9)
Geometry3k	Baseline				54.9	57.8	51.7	77.9	79.0	76.7
	O				55.1 (↑0.2)	56.4 (↓1.4)	51.7 (0.0)	79.6 (↑1.7)	79.0 (0.0)	78.7 (↑2.0)
		O			53.1 (↓1.8)	55.3 (↓2.5)	53.3 (↑1.6)	80.0 (↑2.1)	78.0 (↓1.0)	77.4 (↑0.7)
			O		63.4 (↑8.5)	62.9 (↑5.1)	66.4 (↑14.7)	76.2 (↓1.7)	73.8 (↓5.2)	72.7 (↓4.0)
				O	54.2 (↓0.7)	55.8 (↓2.0)	52.5 (↑0.8)	80.0 (↑2.1)	78.1 (↓0.9)	77.3 (↑0.6)
	O	O			53.7 (↓1.2)	55.3 (↓2.5)	52.5 (↑0.8)	79.0 (↑1.1)	78.5 (↓0.5)	78.7 (↑2.0)
	O		O		62.4 (↑7.5)	61.2 (↑3.4)	65.8 (↑14.1)	76.1 (↓1.8)	74.0 (↓5.0)	73.0 (↓3.7)
	O			O	54.2 (↓0.7)	55.7 (↓2.1)	52.0 (↑0.3)	79.2 (↑1.3)	78.8 (↓0.2)	78.6 (↑1.9)
		O	O		61.6 (↑6.7)	62.2 (↑4.4)	66.7 (↑15.0)	76.6 (↓1.3)	73.8 (↓5.2)	73.1 (↓3.6)
		O		O	53.0 (↓1.9)	55.5 (↓2.3)	53.1 (↑1.4)	79.5 (↑1.6)	78.1 (↓0.9)	77.9 (↑1.2)
			O	O	62.1 (↑7.2)	61.9 (↑4.1)	66.2 (↑14.5)	76.7 (↓1.2)	73.9 (↓5.1)	72.9 (↓3.8)
	O	O	O		62.0 (↑7.1)	60.8 (↑3.0)	66.4 (↑14.7)	75.8 (↓2.1)	74.3 (↓4.7)	73.6 (↓3.1)
	O	O		O	53.7 (↓1.2)	55.1 (↓2.7)	52.5 (↑0.8)	79.1 (↑1.2)	78.7 (↓0.3)	78.8 (↑2.1)
	O		O	O	62.2 (↑7.3)	61.1 (↑3.3)	65.9 (↑14.2)	76.1 (↓1.8)	73.8 (↓5.2)	73.2 (↓3.5)
		O	O	O	61.6 (↑6.7)	61.7 (↑3.9)	66.5 (↑14.8)	76.4 (↓1.5)	73.8 (↓5.2)	73.2 (↓3.5)
	O	O	O	O	61.8 (↑6.9)	60.9 (↑3.1)	66.2 (↑14.5)	75.8 (↓2.1)	74.1 (↓4.9)	73.5 (↓3.2)

Table 1: Impact of auxiliary modalities on test dataset. Columns list base multimodal models and reasoning models. Cells report accuracy (%) with difference from *baseline* in parentheses. Shade red indicates gain; blue for drop.

C Experiments Performance

In the appendix, Table 1 presents the numerical performance result across multiple experimental settings. The table includes the baseline performance obtained using the original diagrams and geometry problems, as well as the results for all combinations of auxiliary modalities.

Each result represents the average accuracy (%) computed over five independent runs. Accuracy values are reported alongside their differences from the *baseline* in parentheses. Performance gains are highlighted in red, while performance drops are indicated in blue.

Table 2 shows the standard deviation of performance for various LLMs on the FormalGeo7K and Geometry3k datasets. The numbers in parentheses indicate the consistency of each model's results across different input combinations.

D Worked Examples

In this appendix, we present an end-to-end example of our pipeline to demonstrate how the Gemini-2.5-flash model processes a single geometry problem. This example is illustrated in Figures 9, 10, 11, 12, and 13. The correct answer for this problem is 70, and the reference CDL is as follows: Shape(AO,OD,DC,OCA), Shape(OA,AD,DO), Shape(CD,DB,BC), Shape(CB,OBC), Shape(BD,DA,OAB), Collinear(ODB), Collinear(ADC), Cocircular(O,ABC).

---

[1] https://openrouter.ai/

Dataset	Data Type				Opened LLM			Closed LLM		
	CDL	T_{CDL}	DES	IMG	Qwen 2.5-VL 72B-Instruct	Llama-4 Scout–Instruct	Gemma-3 27b-it	GPT-o4-mini 2025-04-16	Gemini-2.5 flash	Gemini-2.5 flash think
FormalGeo7K	Baseline				59.2 (±1.0)	64.6 (±0.8)	67.3 (±0.5)	87.7 (±0.4)	83.7 (±0.9)	90.1 (±1.0)
	O				59.5 (±2.1)	65.3 (±1.5)	68.5 (±0.5)	87.8 (±0.9)	86.6 (±0.9)	91.4 (±0.7)
		O			58.5 (±0.6)	63.2 (±1.5)	69.4 (±0.3)	87.0 (±1.6)	87.3 (±0.7)	92.2 (±0.4)
			O		63.4 (±0.8)	64.3 (±0.4)	69.1 (±0.7)	82.3 (±0.9)	82.2 (±0.3)	85.9 (±0.2)
				O	60.3 (±0.4)	63.8 (±0.8)	66.9 (±1.9)	85.8 (±0.7)	85.1 (±0.5)	91.3 (±0.2)
	O	O			60.0 (±1.1)	64.8 (±1.5)	69.4 (±0.8)	86.1 (±0.6)	87.4 (±0.5)	91.0 (±1.2)
	O		O		61.6 (±0.6)	64.1 (±0.6)	67.5 (±1.2)	82.2 (±0.7)	81.8 (±0.9)	86.7 (±0.5)
	O			O	57.8 (±0.5)	64.0 (±0.8)	69.0 (±0.5)	85.9 (±0.5)	86.7 (±0.6)	92.4 (±0.3)
		O	O		63.3 (±0.7)	65.2 (±0.7)	69.1 (±1.4)	83.0 (±0.3)	82.6 (±0.5)	87.9 (±0.7)
		O		O	59.5 (±0.8)	64.7 (±1.1)	68.8 (±0.6)	85.9 (±0.9)	87.3 (±0.5)	93.8 (±0.5)
			O	O	63.5 (±0.6)	64.3 (±0.3)	68.4 (±1.1)	80.9 (±1.0)	81.6 (±0.8)	87.9 (±0.9)
	O	O	O		63.1 (±0.9)	63.5 (±0.4)	68.4 (±1.1)	82.9 (±0.7)	82.8 (±0.9)	87.2 (±0.6)
	O	O		O	58.1 (±0.8)	64.3 (±0.4)	70.6 (±0.5)	87.0 (±0.9)	87.6 (±0.7)	91.3 (±0.7)
	O		O	O	63.4 (±0.9)	63.4 (±0.6)	68.3 (±0.7)	82.3 (±0.8)	81.0 (±0.9)	88.1 (±0.3)
		O	O	O	64.5 (±0.7)	64.4 (±0.8)	69.8 (±0.4)	83.4 (±1.0)	83.0 (±0.8)	89.3 (±0.9)
	O	O	O	O	64.2 (±0.8)	64.7 (±0.7)	68.0 (±0.8)	83.7 (±0.5)	82.7 (±1.0)	88.2 (±0.5)
Geometry3k	Baseline				54.9 (±0.2)	57.8 (±0.7)	51.7 (±1.2)	77.9 (±0.9)	79.0 (±1.5)	76.7 (±0.8)
	O				55.1 (±1.1)	56.4 (±0.2)	51.7 (±1.4)	79.6 (±1.5)	79.0 (±0.1)	78.7 (±0.9)
		O			53.1 (±0.4)	55.3 (±2.2)	53.3 (±0.9)	80.0 (±0.7)	78.0 (±0.6)	77.4 (±0.7)
			O		63.4 (±1.1)	62.9 (±0.9)	66.4 (±1.1)	76.2 (±0.8)	73.8 (±0.4)	72.7 (±0.4)
				O	54.2 (±1.6)	55.8 (±0.7)	52.5 (±1.0)	80.0 (±1.3)	78.1 (±0.9)	77.3 (±0.8)
	O	O			53.7 (±0.9)	55.3 (±1.4)	52.5 (±0.8)	79.0 (±0.6)	78.5 (±1.1)	78.7 (±1.3)
	O		O		62.4 (±0.6)	61.2 (±0.6)	65.8 (±0.8)	76.1 (±0.7)	74.0 (±0.3)	73.0 (±0.7)
	O			O	54.2 (±0.6)	55.7 (±0.8)	52.0 (±0.9)	79.2 (±0.9)	78.8 (±0.5)	78.6 (±0.9)
		O	O		61.6 (±0.9)	62.2 (±0.9)	66.7 (±0.9)	76.6 (±0.6)	73.8 (±0.7)	73.1 (±0.8)
		O		O	53.0 (±0.5)	55.5 (±1.2)	53.1 (±0.4)	79.5 (±0.9)	78.1 (±0.7)	77.9 (±0.6)
			O	O	62.1 (±0.5)	61.9 (±0.7)	66.2 (±0.5)	76.7 (±0.7)	73.9 (±0.6)	72.9 (±0.5)
	O	O	O		62.0 (±0.5)	60.8 (±0.7)	66.4 (±0.4)	75.8 (±0.8)	74.3 (±0.5)	73.6 (±0.6)
	O	O		O	53.7 (±0.9)	55.1 (±1.3)	52.5 (±1.2)	79.1 (±0.7)	78.7 (±0.5)	78.8 (±0.7)
	O		O	O	62.2 (±0.6)	61.1 (±0.6)	65.9 (±0.4)	76.1 (±0.8)	73.8 (±0.5)	73.2 (±0.6)
		O	O	O	61.6 (±0.8)	61.7 (±0.9)	66.5 (±0.9)	76.4 (±0.6)	73.8 (±0.7)	73.2 (±0.5)
	O	O	O	O	61.8 (±0.7)	60.9 (±0.7)	66.2 (±0.5)	75.8 (±0.9)	74.1 (±0.7)	73.5 (±0.6)

Table 2: Standard deviation of performance for various LLMs on the FormalGeo7K and Geometry3k datasets. The values in parentheses beside the main performance scores represent the standard deviation.

E Illustrative Examples of DES Impact

For a direct comparison between open and closed LLMs with and without DES, we also include examples from Qwen 2.5-VL and GPT-o4-mini. These examples, shown in Figures 14a, 14b, 15a, and 15b, were conducted on Geometry3k problems, using different problems for each model.

As shown in Figure 14, open LLMs like Qwen 2.5-VL leverage external knowledge from auxiliary modalities like DES to improve performance. For instance, in Figure 14a, the model correctly identifies the formula for the sum of a pentagon's interior angles as $180(n-2)$ and calculates the sum to be $540°$. However, it fails because it incorrectly interprets the given angles ($x°$, $(x+4)°$, $(x+10)°$, and $(x-6)°$) as interior angles, when they are in fact exterior angles. This error is prevented in Figure 14b where the DES explicitly states that the angles are "exterior angle, and those five angles have been labeled," thereby guiding the model to apply the correct geometric principle.

Conversely, as illustrated in Figure 15, closed LLMs like GPT-o4-mini show a performance decrease when provided with external knowledge from DES. Figure 15b shows a case where the DES provides incorrect information, specifically stating that "corresponding angles facts give". This causes the LLM to establish a faulty premise, such as "$\angle 2 = \angle 1 = 50°$, $\angle 8 = \angle 6$ ", which leads to an incorrect solution. This problem is avoided in Figure 15a, where the absence of DES prevents the model from being misled by erroneous external information.

Diagram Formalizer prompt:

Based on the image, predict the construction_cdl and calibrate it.

(a) Prompt to the Diagram Formalizer for generating construction CDL.

Diagram Formalizer output:

construction_cdl:
Shape(OC,CD,DO),Shape(AO,OD,DA),Shape(OB,BC,CO),Shape(AB,BO,OA),Collinear(BOD),Collinear(AOC)
calibrate construction_cdl:
Shape(AO,OD,DA),Shape(AB,BO,OA),Shape(OB,BC,CO),Shape(DO,OC,CD),Collinear(AOC),Collinear(BOD)

(b) Example construction CDL output from the Diagram Formalizer.

Figure 4: Input and output of Diagram Formalizer. (a) The prompt instructing the Diagram Formalizer to generate only the construction CDL; (b) A sample set of construction CDL statements produced by the Diagram Formalizer. Prompt begins with the geometric diagram as input.

Diagram code generation:

Given a geometric diagram composed of basic shapes (e.g., triangles, rectangles, circles, line) and labeled elements (e.g., points like A, B; lines like AB; angles like 90°), generate **different Python scripts** that use matplotlib to recreate the diagram with **visually distinct layouts** while **preserving all geometric information**.

Each of the scripts must:
- Represent the **same geometric structure and connections** as the original image.
- Alter only the **visual appearance**, using transformations such as:
  - Whole-image rotation
  - Horizontal or vertical flipping
  - Per-shape flipping or local rotation
  - Symmetry-based shape reordering
  - Matrix-based transforms like affine or transpose
  - Other transformation

Labeling and Annotation Rules:
- For point labels (e.g., A, B), segment labels (e.g., AB), and angles (e.g., 90°):
  - Place text **close to but not overlapping** lines, shapes, or other labels.
  - Use **manual position adjustments** (small shifts in x/y) to ensure readability.
  - Angle values should appear inside or near arcs, segment names near the middle of lines, and point names near (but offset from) the points.
  - Ensure that each label remains visually associated with the correct element, even after transformation.

Connection Rules:
- If a line originally connects point A to point B, it must continue connecting A to B after transformation.
- Do **not** create incorrect or new connections (e.g., connecting A to a different point like W).

Code Constraints:
- Write all code inline — **do not define any new functions**.
- Use matplotlib.use('Agg') at the top to disable GUI rendering.
- Do **not** call plt.show().
- Save each image using plt.savefig().

Output:
Generate five complete Python code blocks (one per variation), each implementing a visually distinct rendering of the same geometric structure.

(a) Prompt to the multimodal LLM for generating code that creates a new, transformed diagram given the original input diagram. Prompt begins with the geometric diagram as input.

Re generating diagram code:

You are a highly skilled Python graphics programmer. Fix broken matplotlib code.

You previously generated matplotlib Python code for a geometric diagram.
There was an error when running code. Please revise and correct the code.

Here is the broken code:
{broken_code}
And here is the error message that occurred during execution:
{error_message}
Please fix the above code and return a fully working Python code block that follows all original constraints:
- Do not define new functions.
- Save the image using `plt.savefig(...)`.
- Use `matplotlib.use("Agg")` and avoid `plt.show()`.

(b) Prompt to the multimodal LLM for retries code generation upon errors.

Figure 5: Prompts to the multimodal LLM for (a) generating code to create a new, transformed diagram given the original input diagram, and (b) retrying code generation upon encountering errors. Blue text indicates external inputs—the erroneous code and accompanying error message.

Diagram description generation:

Look at the geometric diagram and briefly describe its components (points, lines, angles, and shapes).
Mention any notable relationships between elements, such as parallel lines, perpendicularity, equal lengths, or symmetry.

Figure 6: Prompt to the multimodal LLM for generating a natural-language description of the input diagram. Prompt begins with the geometric diagram as input.

Verifying mathematical equivalence prompt:

You are given two mathematical expressions: a ground truth answer and a predicted answer. Your task is to decide whether they are **mathematically equivalent**.
- If they are equivalent (even if written differently, e.g., fractions vs decimals, different forms), output **True**.
- If they are not equivalent, output **False**

To make this determination, follow these guidelines:
1. You may use **symbolic manipulation** or **numerical evaluation** methods.
2. Assume the **domain of variables is real numbers**, unless context clearly implies otherwise (e.g., integer powers, specific functions).
3. When performing numerical comparisons, consider a **standard floating-point precision tolerance**.
4. **IMPORTANT SPECIAL RULE:** For the purpose of this comparison task, a **numerical value N without units should be considered mathematically equivalent to N° (N degrees)**. This rule applies even though they are mathematically distinct in standard contexts.
5. Output **True** only if you are **confident** the expressions are mathematically equivalent based on these rules. Otherwise, output **False**

Only output a single word: **True** or **False**
Here are the two expressions:

Ground truth answer:
{answer}
Predicted answer:
{prediction}

Now, determine if they are mathematically equivalent.
Answer with only **True** or **False**.

Figure 7: Prompt for verifying mathematical equivalence between the ground-truth and predicted answers.

Geometry problem-solving prompts:

{*IMG*
The first image is the original geometric diagram.
The following images are different redrawn versions of the same diagram, preserving the exact geometric relationships.}
The first image is the original geometric diagram.
The following images are different redrawn versions of the same diagram, preserving the exact geometric relationships.
Using the provided geometric image and question, always end with the final answer in the following format:
The final answer is: \\boxed{{your_answer}}
The question is:
{Conditional Declaration Language (CDL):
construction_cdl: *CDL*}
{The diagram structure is as follows, T_{CDL}}
{The following is a description of the same geometric image.
DES}
{Question of geometric problem}

Figure 8: Prompt for solving the problem: red text denotes auxiliary modalities, blue text denotes the corresponding additional inputs provided when each modality is included, and purple text denotes the problem statement, which is always appended. Prompt begins with the geometric diagram as input.

Multimodal LLM input:
As shown in the diagram, ∠BCA=35°, the center of ⊙O is O. Find ∠BOA.

Multimodal LLM output:
The diagram shows a circle with center O. Points A, B, and C are on the circle. Angle BCA is an inscribed angle subtending arc AB. Angle BOA is the central angle subtending arc AB.
The relationship between an inscribed angle and the central angle that subtend the same arc is that the central angle is twice the inscribed angle. In this case, both $angle text{BCA}$ and $angle text{BOA}$ subtend the arc AB.
Therefore, the measure of the central angle $angle text{BOA}$ is twice the measure of the inscribed angle $angle text{BCA}$.
We are given that $angle text{BCA} = 35^circ$.
Using the relationship between the central angle and the inscribed angle, we have:
$angle text{BOA} = 2 * angle text{BCA}$
$angle text{BOA} = 2 * 35^circ$
$angle text{BOA} = 70^circ$

The final answer is $boxed{70}$.

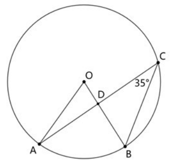


Figure 9: Baseline input and model output. The original diagram is shown in the lower right corner.

Multimodal LLM input CDL:
Using the provided geometric image and question, always end with the final answer in the following format:
The final answer is: boxed{{your_answer}}
The question is:
Conditional Declaration Language (CDL):
construction_cdl:Shape(AO,OD,DC,OCA),Shape(OA,AD,DO),Shape(CD,DB,BC),Shape(CB,OBC),Shape(BD,DA,OAB),Collinear(ODB),Collinear(ADC),Cocircular(O,ABC)
As shown in the diagram, ∠BCA=35°, the center of ⊙O is O. Find ∠BOA.

Multimodal LLM output:
The given image shows a circle with center O, and points A, B, C, and D on the circle. We are given that ∠BCA = 35°. We need to find ∠BOA.
In a circle, the angle subtended by an arc at the center is twice the angle subtended by the same arc at any point on the remaining part of the circle.\nThe arc AB subtends ∠BCA at point C on the circle and ∠BOA at the center O.
Therefore, the relationship between ∠BOA and ∠BCA is:
∠BOA = 2 * ∠BCA

We are given that ∠BCA = 35°.
Substituting this value into the formula, we get:\n∠BOA = 2 * 35°
∠BOA = 70°

Therefore, the measure of angle BOA is 70 degrees.

The final answer is $boxed{70}$.

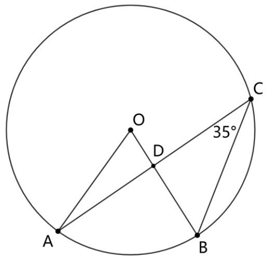


Figure 10: CDL input sample. The added CDL text is highlighted in red; the diagram remains in the lower right corner.

Multimodal LLM input with T_{CDL} :
The diagram structure is as follows, Segments [AO, OD, DC, OCA] form a shape, Segments [OA, AD, DO] form a triangle, Segments [CD, DB, BC] form a triangle, Segments [CB, OBC] form a shape, Segments [BD, DA, OAB] form a shape,[Point O, Point D, Point B] are at same line,[Point A, Point D, Point C] are at same line, Points [A, B, C] lie on a circle centered at point O.
As shown in the diagram, ∠BCA=35°, the center of ⊙O is O. Find ∠BOA.

Multimodal LLM output:
The diagram shows a circle with center O. Points A, B, and C are on the circle. We are given that ∠BCA = 35°. We need to find ∠BOA.
The angle subtended by an arc at the center is twice the angle subtended by the same arc at any point on the remaining part of the circle.
In this case, ∠BCA is the angle subtended by the arc AB at point C on the circumference. ∠BOA is the angle subtended by the same arc AB at the center O.\nTherefore, according to the theorem, ∠BOA = 2 * ∠BCA.\nGiven that ∠BCA = 35°, we can substitute this value into the formula:
∠BOA = 2 * 35°
∠BOA = 70°

The final answer is $\\boxed{70}$.

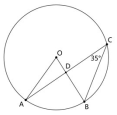
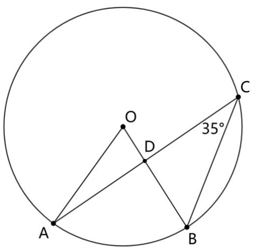


Figure 11: Text-CDL (T_{CDL}) input. The T_{CDL} segment is highlighted in red; the diagram is in the lower right corner.

Multimodal LLM input with DES:
The following is a description of the same geometric image.
Here's what you see in the picture, and the relationships you can read off at a glance:



1. The big circle
   - Points A, B and C all lie on the circumference.
   - O is the center of the circle.

2. Radii
   - Segments OA, OB (and implicitly OC) are all radii and hence OA = OB = OC.
   - OB is drawn so that it meets the chord AC at an interior point D.

3. Chords and inscribed triangle
   - AC and BC are chords of the circle, and together with AB they form the inscribed triangle ABC.
   - The arc AB (the "bottom" arc of the circle) has been thickened to show that it is the arc subtended by both the central angle at O and the inscribed angle at C.
4. Angles
   - At C the inscribed angle ∠ACB is marked 35°, and it intercepts arc AB.
   - The central angle ∠AOB intercepts that same arc AB, and in circle geometry one has $\angle AOB = 2 \cdot \angle ACB = 70^\circ$.
5. Intersection point D
   - D is simply the point where the radius OB cuts the chord AC. (In many problems the line from the center to a chord's midpoint is perpendicular to the chord, but here no right-angle mark is shown, so D need not be the midpoint nor $OD \perp AC$.)

6. Isosceles triangles and symmetry
   - Triangles OAB and OBC are isosceles (two sides are radii).
   - There is rotational symmetry of order two if you swap A with C and reflect across OB, but the 35°marking breaks full symmetry.

In summary: you have a circle with center O, equal-length radii OA, OB, OC; two chords AC and BC making an inscribed triangle ABC; a 35°inscribed angle at C; the corresponding central angle at O; and the chord AC is cut by the radius OB at point D.
As shown in the diagram, ∠BCA=35°, the center of ⊙O is O. Find ∠BOA.

Multimodal LLM output:
According to the information given in the problem description and the diagram, $angle BCA$ is an inscribed angle that subtends arc AB. The central angle that subtends the same arc AB is $angle BOA$. The relationship between an inscribed angle and a central angle subtending the same arc is that the central angle is twice the inscribed angle.
Therefore, $angle BOA = 2 cdot angle BCA$.\nGiven that $angle BCA = 35^circ$, we can calculate $angle BOA$ as follows:\n$angle BOA = 2 cdot 35^circ$\n$angle BOA = 70^circ$

The final answer is $boxed{70}$.

Figure 12: DES input sample. The generated description (DES) is shown in red; the diagram is in the lower right corner.

Multimodal LLM input with IMG:
As shown in the diagram, ∠BCA=35°, the center of ⊙O is O. Find ∠BOA.

Multimodal LLM output:
The given image shows a circle with center O. Points A, B, and C are on the circumference of the circle. Angle $angle BCA = 35^circ$. We need to find the measure of angle $angle BOA$.

In a circle, the angle subtended by an arc at the center is twice the angle subtended by the same arc at any point on the remaining part of the circle.
In the given diagram, the arc AB subtends the angle $angle BCA$ at point C on the circumference and the angle $angle BOA$ at the center O.
Therefore, according to the theorem, the angle at the center is twice the angle at the circumference subtended by the same arc.
So, we have:
$angle BOA = 2 cdot angle BCA$

We are given that $angle BCA = 35^circ$.
Substituting the value of $angle BCA$ into the equation, we get:
$angle BOA = 2 cdot 35^circ$
$angle BOA = 70^circ$

Thus, the measure of angle $angle BOA$ is $70^circ$.

The final answer is: $boxed{70}$.

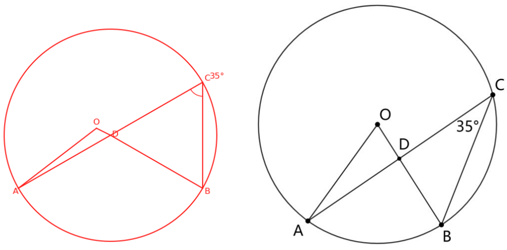
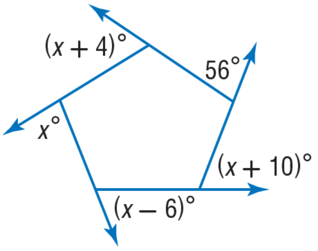


Figure 13: IMG input sample. The auxiliary image (IMG) are highlighted in red and placed to the left of the original diagram.

Qwen 2.5-VL input :
<image>Find the value of x

Qwen 2.5-VL output:
To find the value of (x) in the given pentagon, we need to use the fact that the sum of the interior angles of a pentagon is (540^circ). The formula for the sum of the interior angles of an (n)-sided polygon is:
[180(n-2)]
For a pentagon ((n = 5)):
[180(5-2) = 180 times 3 = 540^circ]
Given the angles in the pentagon are (x^circ), ((x+4)^circ), (56^circ), ((x+10)^circ), and ((x-6)^circ), we can set up the equation:
[x + (x+4) + 56 + (x+10) + (x-6) = 540]
Simplify the left-hand side:
[x + x + 4 + 56 + x + 10 + x - 6 = 540]
Combine like terms:
[4x + 64 = 540]
Subtract 64 from both sides:
[4x = 476]
Divide by 4:
[x = 119]

Thus, the value of (x) is:
[boxed{119}]



Answer: 74

(a) Input and output of the Qwen 2.5-VL without DES.

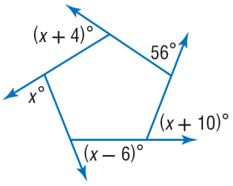
Qwen 2.5-VL input with *DES* :
The following is a description of the same geometric image.The figure is built around a single five-sided (pentagonal) "core" whose sides have all been "prolonged" into rays (the little arrowheads simply show that each side keeps going). At each of the five vertices the extension creates an exterior angle, and those five angles have been labeled (in order around the pentagon)

- x°
- (x + 4)°
- 56°
- (x + 10)°
- (x – 6)°

No sides are marked parallel or perpendicular, there are no tick-marks denoting equal lengths, and there is no obvious line of symmetry. The one key relationship implied is that these five exterior angles together "go all the way around" the pentagon, so their sum is 360°.
<image>Find the value of x

Qwen 2.5-VL output:
To find the value of (x), we start by using the fact that the sum of the exterior angles of any polygon is always (360^circ).
For this pentagon, the exterior angles are given as (x^circ), ((x + 4)^circ), (56^circ), ((x + 10)^circ), and ((x - 6)^circ).
We can write the equation for the sum of these exterior angles:
[x + (x + 4) + 56 + (x + 10) + (x - 6) = 360]
Next, we simplify the left-hand side of the equation by combining like terms:
[x + x + 4 + 56 + x + 10 + x - 6 = 360]
[4x + 64 = 360]
Now, we solve for (x) by isolating it on one side of the equation. First, subtract 64 from both sides:
[4x = 360 - 64]
[4x = 296]
Then, divide both sides by 4:
[x = frac{296}{4}]
[x = 74]

Thus, the value of (x) is:[boxed{74}]

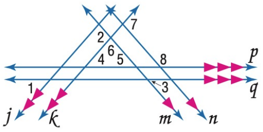


Answer: 74

(b) Input and output of the Qwen 2.5-VL with DES.

Figure 14: Input and output of the Qwen 2.5-VL on a Geometry3k example. The figures compare the performance with and without a diagram description (DES). The correct answer is shown in the bottom-right corner of the image.

GPT-o4-mini input:
<image>In the figure, $m∠1 = 50$ and $m∠3 = 60$. Find the measure of $\\angle 8$



GPT-o4-mini output:
Here is the quickest way to see it. Look at the right-hand "transversal" m cutting the two parallel lines p (top) and q (bottom). At the lower intersection with q it makes the interior (co-interior) angle
∠3 = 60°
At the upper intersection with p it makes the interior (co-interior) angle ∠8. Co-interior angles on the same side of a transversal sum to 180°, so
∠8 + ∠3 = 180°
∠8 + 60°= 180°
∠8 = 120°.

The final answer is: \\boxed{120}

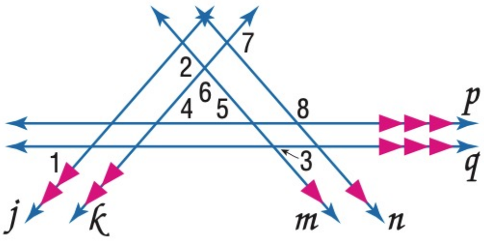
Answer: 120

(a) Input and output of the GPT-o4-mini without DES.

GPT-o4-mini input with *DES*:
The following is a description of the same geometric image.
The picture is built around two horizontal lines "p" (top) and "q" (bottom), each marked with the same triple-arrow symbol to show that p //q. Four slanted lines (transversals) cross between them:
1. Two "up-going" rays start at a single point on the lower line q (their endpoints on q are marked j and k) and meet p at two distinct points.
2. Two "down-going" rays start at two distinct points on p and re-meet q at points m and n.
Where each transversal crosses p or q an angle is numbered (1 at the first up-ray on q, 2 at its intersection with p, 3 and 7 on the first down-ray, 8 on the second down-ray), and where the up- and down-rays cross each other inside the strip there are the three central angles 4, 5, 6 (in fact around each crossing you have the usual "vertical" pairs of equal angles).
Notable relationships:
- Because p//q, corresponding angles are equal (for example angle 1 on q = angle 2 on p, angle 3 = angle 7, angle 6 = angle 8, etc.).
- Each crossing of two lines produces vertical-angle pairs (e.g. in the middle 4 and 6 are opposite angles, as are 5 and the un-numbered one opposite it).
- Alternate-interior angles are equal, and any pair of consecutive interior angles along a single transversal sum to 180°.
- The four transversals enclose a little crossed quadrilateral between p and q whose opposite angles are equal (a consequence of parallel–transversal angle rules).

Points j, k, m and n simply mark the feet of those four rays on the lower line, the two upper intersection points are left unlabeled, and all the pink arrows on p and q serve only to indicate "these two lines are parallel.
<image>In the figure, $m∠1 = 50$ and $m∠3 = 60$. Find the measure of $\\angle 8$

GPT-o4-mini output:
First, note that since p//q the corresponding-angles facts give
(1) ∠2 = ∠1 = 50°
(2) ∠7 = ∠3 = 60°
(3) ∠8 = ∠6
Now look at the little "triangle" whose vertices are
– P_1 (UL ∩ p), with interior angle ∠2 = 50°,
– P_3 (DL ∩ p), with interior *interior-of-that-triangle* angle ∠7 = 60°,
– X = (UL ∩ DL), the crossing inside, whose angle there is ∠6.
Because P_1P_3X is a triangle, its interior angles sum to 180°, so
∠2 + ∠7 + ∠6 = 180°
⇒50°+ 60°+ ∠6 = 180°
⇒∠6 = 180°−110°=70°.
But from (3) above ∠8 = ∠6, hence
∠8 = 70°.



The final answer is: \\boxed{70}

Answer: 120

(b) Input and output of the GPT-o4-mini with DES.

Figure 15: Input and output of the GPT-o4-mini on a Geometry3k example. The figures compare the performance with and without a diagram description (DES). The correct answer is shown in the bottom-right corner of the image.