SynGeo: Synergizing Seeing and Proving through Revisable Geometric States

Tianyi Xu[1*] **Liu Yang**[1*] **Wenjun Gao**[1*] **Junyu Ou**[1] **Zhe Zhao**[2] **Haibin Wen**[1]
Maolin Wang[1] **Ye Wei**[1†]
[1] City University of Hong Kong, Hong Kong, China
[2] Stanford University, Stanford, California, United States

Abstract

Geometry problem solving is a canonical testbed for machine intelligence, requiring systems to interpret diagrams, ground symbolic constraints, and perform rigorous deduction. Yet current approaches expose a persistent gap between seeing and proving: multimodal large language models can flexibly inspect diagrams but may hallucinate unsupported relations, while symbolic solvers provide checkable derivations but are brittle to incomplete or misgrounded formalization. We introduce **SynGeo**, a state-centric framework for **syn**ergizing seeing and proving in **geo**metry problem solving by making the geometric representation revisable during inference. SynGeo first constructs a predicate state from diagram–text evidence and tests it with symbolic reasoning. When proof search fails or stagnates, symbolic diagnostics guide image-grounded revisiting, repairing solver-incompatible predicates before another deductive attempt. When the symbolic route remains unresolved, a complementary MLLM branch provides an image-grounded reasoning path from the problem evidence. On Geometry3K and PGPS9K, SynGeo achieves state-of-the-art performance across both *Choice* and *Completion* settings. With GPT-4o as the backbone, it reaches 90.2% Choice and 89.0% Completion on Geometry3K, and 87.5% Choice and 85.4% Completion on PGPS9K. Ablations studies demonstrate the benefits of feedback-guided state revision and conditional complementary reasoning, supporting a broader view of geometry problem solving as an adaptive interaction between seeing and proving.

1 Introduction

Geometry problem solving (GPS) has long served as a challenging benchmark for mathematical intelligence, requiring systems to jointly interpret diagram–text information and perform deductive reasoning [Seo et al., 2015, Lu et al., 2021, Peng et al., 2023, Wu et al., 2024, Ping et al., 2026]. Unlike text-only mathematical tasks, GPS distributes its evidence across two modalities: points, lines, arcs, numerical labels, and spatial relations are embedded in the diagram, while constraints and the target question are expressed in language. A solver must therefore *see*, grounding geometric entities and relations in the figure and aligning them with text, and *prove*, chaining these grounded relations into a derivation that reaches the target [Seo et al., 2015, Lu et al., 2021]. Recent multimodal reasoning benchmarks show that even strong models struggle to sustain fine-grained visual grounding and long-range geometric deduction within a single problem [Lu et al., 2023, Wang et al., 2024, Zhang et al., 2024b,a].

Existing approaches reveal a fundamental trade-off between these two capabilities. Multimodal large language models (MLLMs) read diagrams directly and can tolerate the messy parts of geometry

*Equal contribution.
†Corresponding author: ye.wei@cityu.edu.hk

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

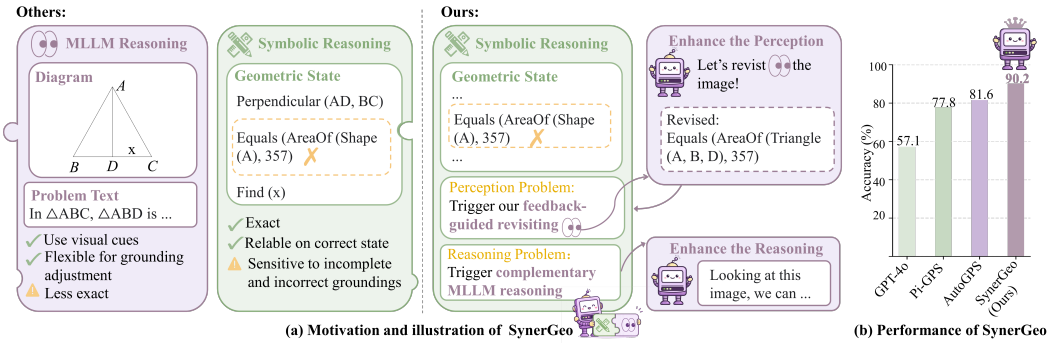


Figure 1: **Motivation and performance of SynGeo.** (a) Geometry problem solving requires both image-grounded seeing and symbolic proving. MLLM reasoning provides flexible access to visual evidence but can be imprecise, while symbolic reasoning provides verifiable deduction but relies on a complete and correctly grounded geometric state. SynGeo treats these two reasoning modes as complementary pieces: feedback-guided revisiting repairs perception-side errors in the geometric state, and complementary MLLM reasoning supports cases where symbolic deduction alone is insufficient. (b) Performance gains on Geometry3K under the choice setting. SynGeo substantially outperforms GPT-4o, Pi-GPS, and AutoGPS, demonstrating the effectiveness of combining feedback-guided revisiting with complementary MLLM reasoning.

problems, such as implicit incidences, ambiguous shaded regions, and labels that float between nearby segments [Hurst et al., 2024, Bai et al., 2025, Gao et al., 2023, Huang et al., 2025]. Yet their flexibility can come at the cost of reliability: derivations may skip steps, invent relations, or even reach the correct answer through reasoning that does not survive inspection. Symbolic and neuro-symbolic solvers take the opposite route: they parse problems into formal predicates, then apply theorem rules, algebraic transformations, or proof-graph search over them [Lu et al., 2021, Peng et al., 2023, Wu et al., 2024, Ping et al., 2026]. Their answers can come with checkable derivations, but this rigor comes with fragility: one missing incidence, one misread arc, or one ambiguous shaded region can leave proof search with nowhere to go, even when the original diagram still contains the evidence needed for recovery. As illustrated in Figure 1, these two paradigms behave like complementary pieces of a geometry solver: one keeps the diagram open and reasons with flexible visual access, while the other demands a clean geometric state and rewards it with a verifiable proof. The central question is therefore not whether geometry solvers should rely on perception or proof, but how these two modes can be made to cooperate.

Recent systems improve the formalization side of GPS by using diagrammatic information more explicitly. Pi-GPS shows that diagram context can help resolve textual ambiguities through rectification and verification [Zhao et al., 2025]. Other neuro-symbolic systems further demonstrate the value of formal representations and theorem-based reasoning for producing checkable solutions [Lu et al., 2021, Peng et al., 2023, Wu et al., 2024, Ping et al., 2026]. These advances confirm that formalization quality is critical for reliable geometry solving. We take a complementary view: symbolic failure should not only be treated as an unsuccessful solve, but also as a diagnostic signal for repairing the state on which solving depends. When proof search stalls on a predicate, the predicate itself may reveal what the system has misread; when reasoning stagnates, the gap often points to a relation that exists in the diagram but is absent from the current state.

We build **SynGeo** to synergize seeing and proving through a revisable geometric state, represented as formal predicates extracted from the diagram and text. SynGeo first constructs an initial state and reasons over it with a symbolic proof process. When reasoning cannot close the goal, SynGeo returns diagnostic feedback rather than only a failed answer, indicating which predicate cannot be grounded, which sub-goal has stalled, or what evidence appears to be missing. Two recovery routes use this feedback: *feedback-guided revisiting* sends the diagnostic signal and the original image back to the MLLM to revise the erroneous geometric state in place, while *complementary MLLM reasoning* is invoked when even a repaired state remains insufficient for symbolic solving.

SynGeo treats the geometric state as a writable interface that connects perception and symbolic reasoning. The perception module constructs and repairs the geometric relations, while the symbolic solver validates the resulting state, identifies inconsistencies or missing information, and performs

formal deduction once the state satisfies symbolic solvability requirements. This interaction preserves the rigor of symbolic proof when possible, while retaining access to image-grounded reasoning when formal deduction alone is insufficient. We evaluate SynGeo on Geometry3K and PGPS9K under both choice and completion settings. Across both benchmarks, SynGeo achieves strong performance against direct MLLM baselines and neuro-symbolic solvers, supporting the view that robust GPS requires coordinated state repair and reasoning complementarity rather than a single fixed inference path. Our main contributions are summarized as follows:

- We reframe geometry problem solving as a synergy between seeing and proving, where symbolic failures are treated as diagnostic signals for repairing perception-side geometric states rather than as terminal reasoning failures.
- We introduce **SynGeo**, a framework built around a revisable geometric state. SynGeo combines symbolic diagnosis, feedback-guided revisiting, and Complementary MLLM Reasoning to repair solver-incompatible states and recover from unresolved symbolic routes.
- We achieve state-of-the-art results on Geometry3K and PGPS9K under both *Choice* and *Completion* settings. With GPT-4o as the backbone, SynGeo reaches 90.2% Choice accuracy and 89.0% Completion accuracy on Geometry3K, and 87.5% Choice accuracy and 85.4% Completion accuracy on PGPS9K, improving over AutoGPS by up to 13.6 percentage points.

2 Related Work

2.1 Geometry Problem Solving

Geometry problem solving has been studied through symbolic, neural, and neural-symbolic paradigms. Early systems combine text and diagram interpretation to recover geometric entities and relations from multimodal problem statements [Seo et al., 2015]. Inter-GPS introduces a formal language for geometry problems and solves them with symbolic reasoning [Lu et al., 2021]. Subsequent symbolic and neural-symbolic methods improve theorem selection, search, and explainability. GeoDRL uses reinforcement learning to guide deductive reasoning [Peng et al., 2023], while E-GPS combines a top-down solver with a bottom-up generator for explainable geometry solving [Wu et al., 2024]. Neural solvers such as PGPSNet and LANS learn multimodal representations from diagrams and text for answer prediction [Zhang et al., 2023, Li et al., 2024c]. Recent MLLM-based systems, including G-LLaVA and Vision-R1, adapt multimodal large language models to geometry or visual mathematical reasoning through geometry-specific data construction or reasoning-oriented training [Gao et al., 2023, Huang et al., 2025].

Recent work has increasingly focused on improving formalization quality. Pi-GPS [Zhao et al., 2025] uses diagrammatic information to resolve textual ambiguities through rectification and verification [Zhao et al., 2025]. AutoGPS [Ping et al., 2026] combines multimodal formalization with deductive symbolic reasoning and uses symbolic feedback to refine formal descriptions [Ping et al., 2026]. These methods show that accurate diagram–text grounding is crucial for reliable GPS. SynGeo builds on this insight by extending symbolic feedback beyond formalization refinement. Proof-process outcomes are used to revise the evolving geometric state during inference, while unresolved formal routes can transition to image-grounded reasoning.

2.2 Multimodal and Neural-Symbolic Mathematical Reasoning

The rise of MLLMs has motivated benchmarks for visual mathematical reasoning. MathVista evaluates mathematical reasoning in visual contexts [Lu et al., 2023], MathVision focuses on multimodal mathematical reasoning across core domains including geometry [Wang et al., 2024], MathVerse studies whether MLLMs truly use diagram information in visual math problems [Zhang et al., 2024b], and GeoEval targets geometric reasoning in plane and solid geometry [Zhang et al., 2024a]. These benchmarks show that visual mathematical reasoning requires fine-grained grounding, symbolic notation alignment, and multi-step deduction. General MLLMs such as GPT-4o and Qwen-VL provide strong multimodal capabilities [Hurst et al., 2024, Bai et al., 2025], but they can still produce fluent yet unsupported reasoning chains.

Neural-symbolic methods address this limitation by combining neural flexibility with symbolic rigor. AlphaGeometry demonstrates that neural guidance can be combined with symbolic deduction for

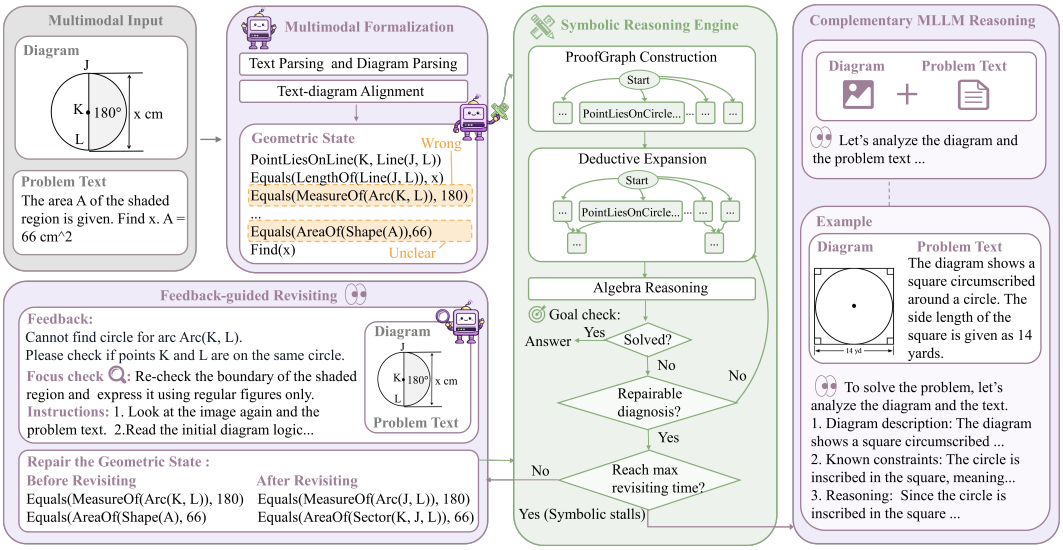


Figure 2: Overview of SynGeo. SynGeo constructs a revisable geometric state from diagram–text evidence, tests it with symbolic reasoning, and uses symbolic diagnostics to guide state repair. If the symbolic route remains unresolved after revisiting, complementary MLLM reasoning provides an image-grounded reasoning path over the original multimodal evidence.

olympiad-level geometry theorem proving [Trinh et al., 2024]. Related work explores autoformalization, theorem proving, inequality reasoning, and reliable data generation with symbolic verification or neural guidance [Wu et al., 2022, Li et al., 2024a,b]. Most existing systems, however, still separate representation construction from downstream reasoning. SynGeo instead treats the geometric state as a revisable interface between image-grounded perception and symbolic proof, allowing perception and reasoning to cooperate over the same evolving representation.

3 Method

SynGeo operationalizes the synergy between seeing and proving through a revisable geometric state, as illustrated in Figure 2 and summarized in Algorithm 1. Given a geometry problem, SynGeo first constructs an initial predicate state from the diagram and text, tests this state with a symbolic reasoning engine, and revises the state when symbolic diagnostics reveal repairable formalization or grounding errors. If the symbolic route remains unresolved after revisiting, SynGeo invokes complementary MLLM reasoning over the original multimodal evidence.

3.1 Revisable Geometric State and Initialization

Given a diagram image I and problem text T, SynGeo represents the current interpretation of the problem as a geometric state

$$S_t = \{l_1, \ldots, l_m, \mathrm{Find}(q)\},$$

where each l_i is a logic-form predicate describing a geometric object, relation, or numerical constraint, and $\mathrm{Find}(q)$ specifies the target quantity. The state may include point-line incidence, circle membership, parallelism, perpendicularity, length and angle equations, area constraints, and other geometry-specific relations.

The initial state is constructed by a predicate-based multimodal formalization frontend:

$$S_0 = \mathcal{F}(I, T).$$

Here, $\mathcal{F}$ denotes the overall initialization pipeline that converts diagram–text evidence into a solver-facing predicate state. Following the formal GPS paradigm of Inter-GPS [Lu et al., 2021] and the implementation pipeline of AutoGPS [Ping et al., 2026], this frontend follows a structured formalization process rather than directly prompting an MLLM to generate the final answer. It uses

Algorithm 1 SynGeo Inference

Require: image I, text T, max revisits K
  $S_0 \leftarrow \mathcal{F}(I, T)$ ▷ predicate-based multimodal formalization
  for $t = 0, \ldots, K$ **do**
    $(y_t, d_t) \leftarrow \mathcal{P}(S_t)$ ▷ symbolic reasoning and diagnosis
    if y_t is solved **then**
      return y_t
    end if
    if $\mathrm{Repairable}(d_t)$ and $t < K$ **then**
      $S_{t+1} \leftarrow \mathcal{R}(I, T, S_t, d_t)$ ▷ feedback-guided revisiting
    else
      break
    end if
  end for
  $y_{\mathrm{m}} \leftarrow \mathcal{M}(I, T)$ ▷ complementary reasoning over problem evidence
  return y_{m}

an FPN-based visual annotation model [Lin et al., 2017] to detect geometrically important points, a PGDP-Net-style diagram parser [Zhang et al., 2022] to extract diagram-side primitives and relations, a rule-based text parser to convert the problem statement into pseudo logic forms, and a multimodal alignment module to link textual references with visual entities and produce solver-compatible predicates. Further implementation details of the formalization frontend are provided in Appendix C.

This initialization is structured but not assumed to be perfect. Text-side forms may still contain unresolved placeholders, such as `AreaOf(Shape($))`, and diagram-side predicates may be syntactically valid but visually misgrounded, for example when an arc or region is attached to the wrong geometric object, as illustrated in Figure 2. SynGeo therefore treats S_0 as an initial hypothesis rather than a final commitment. During inference, symbolic reasoning tests the current state, exposes incomplete or incompatible predicates, and triggers revisiting when the diagnostic feedback is actionable.

At iteration t, the symbolic reasoning engine attempts to solve the problem over the current state, producing $(y_t, d_t) = \mathcal{P}(S_t)$. Here, y_t is the proof-derived answer if the target is reached, and d_t is diagnostic feedback otherwise. If symbolic reasoning succeeds, SynGeo returns y_t. If d_t indicates a repairable state-level issue, feedback-guided revisiting updates the state as $S_{t+1} = \mathcal{R}(I, T, S_t, d_t)$. If the symbolic route remains unresolved after the allowed revisiting attempts, SynGeo invokes complementary MLLM reasoning, denoted by $y_{\mathrm{m}} = \mathcal{M}(I, T)$. The final prediction prioritizes the symbolic answer when available and otherwise returns the normalized MLLM answer.

3.2 Symbolic Reasoning and Diagnosis

The symbolic reasoning engine serves two roles: proof search and diagnosis. Given a state S_t, the engine first validates whether the predicates are syntactically well-formed, self-consistent, and sufficiently complete for symbolic solving. Some missing relations can be completed by symbolic closure, but malformed or misgrounded predicates cannot be repaired by closure alone. This distinction separates deductive completion from perception-side state repair.

When validation succeeds, the engine expands a proof graph over the current state,

$$G_t = (V_t, E_t),$$

where nodes represent predicates, derived equations, or quantities, and directed hyperedges represent theorem applications or algebraic transformations. The graph is expanded by alternating deductive and algebraic reasoning. Deductive expansion applies geometry rules such as incidence propagation, angle relations, circle and arc properties, similarity, congruence, polygon properties, and area formulas. Algebraic reasoning performs substitution, simplification, numerical evaluation, and equation solving. The symbolic route succeeds when the graph derives a solution literal of the form $\texttt{Equals}(q, v)$, where q is the target quantity and v is the derived value.

When validation or proof-graph expansion fails, the engine returns diagnostic feedback instead of only a failed answer. The feedback may indicate unparsable predicates, malformed objects, inconsistent relations, missing target-related predicates, invalid grounding, stalled subgoals, or search failures. For example, an unresolved region placeholder or an arc that cannot be grounded to a valid circle can

be exposed as a state-level error, as shown in Figure 2. These diagnostic signals identify which parts of the current state prevent symbolic solving and provide guidance for feedback-guided revisiting. Implementation details are provided in Appendices C and D.

3.3 Feedback-guided Revisiting and Complementary Reasoning

When diagnostic feedback suggests a repairable state-level issue, SynGeo performs feedback-guided revisiting. The revisiting module receives the diagram–text evidence, the current state S_t, and diagnostic feedback d_t, then re-examines the evidence and edits the state in place. Its goal is not to generate a free-form solution, but to produce a revised geometric state that can be tested again by symbolic reasoning.

The reviser is constrained to make minimal, grounded edits: it preserves correct predicates whenever possible, and every added or repaired predicate must be supported by the diagram or text. This prevents unsupported relations from being introduced merely to satisfy the symbolic engine. For example, when feedback indicates that a shaded-region predicate is underspecified, the reviser re-checks the visual boundary and rewrites the region using solver-compatible geometric objects when possible, as illustrated in Figure 2.

The revised state S_{t+1} is sent back to the symbolic engine, forming a state revision loop. In this loop, symbolic reasoning guides perception by identifying what is missing, malformed, or incompatible, while revisiting repairs the state on which proof search depends. Thus, symbolic failure becomes a diagnostic signal rather than only a stopping condition.

Some problems remain unresolved even after revisiting, for example when the formal language does not express a useful visual shortcut or theorem search does not reach the required derivation. In such cases, SynGeo invokes complementary MLLM reasoning. Unlike revisiting, this branch does not aim to revise the logic-form state; instead, it provides an image-grounded reasoning path from the diagram–text evidence when the symbolic route remains insufficient. SynGeo therefore coordinates seeing and proving through the evolving state of a single problem, rather than combining independent solvers by voting or redundant sampling.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate SynGeo on two standard geometry problem-solving benchmarks, Geometry3K [Lu et al., 2021] and PGPS9K [Zhang et al., 2023]. The Geometry3K dataset contains $3{,}002$ geometry problems and is split into training, validation, and test sets with a $0.7 : 0.1 : 0.2$ ratio; following prior work, we evaluate on its test set, which contains 601 diagram-text geometry problems [Lu et al., 2021]. PGPS9K is a larger plane-geometry benchmark with $9{,}022$ problems paired with diagrams and fine-grained annotations; under the commonly used problem-type split, it contains $8{,}022$ training problems and $1{,}000$ test problems [Zhang et al., 2023, Li et al., 2024c]. Both datasets require joint interpretation of the diagram and problem text, making them suitable for evaluating neural, symbolic, neuro-symbolic, and multimodal geometry solvers.

Evaluation protocol. Following prior work [Lu et al., 2021, Zhang et al., 2023, Ping et al., 2026], we report answer accuracy under two evaluation settings. In the *Choice* setting, the solver is given the candidate answer options and is evaluated by whether it selects the correct option. In the *Completion* setting, the solver must directly generate the final answer without access to candidate choices, and the prediction is judged by matching the target value under the benchmark evaluation protocol.

Baselines. We compare SynGeo with three groups of representative methods. The first group contains direct MLLM baselines, including Claude 3.5 Sonnet [Anthropic, 2024], GPT-4o [Hurst et al., 2024], and Qwen3-VL-32B-Thinking [Bai et al., 2025]. The second group contains neural geometry solvers, including NGS [Chen et al., 2021], Geoformer [Chen et al., 2022], PGPSNet [Zhang et al., 2023], and LANS [Li et al., 2024c]. The third group includes symbolic and neuro-symbolic solvers, including InterGPS [Lu et al., 2021], E-GPS [Wu et al., 2024], GeoDRL [Peng et al., 2023], Pi-GPS [Zhao et al., 2025], and AutoGPS [Ping et al., 2026]. These baselines cover direct multimodal reasoning, learned geometry-specific solving, and structured reasoning based on formal representations and symbolic deduction.

Table 1: Performance comparison on Geometry3K and PGPS9K. We report accuracy under both Choice and Completion settings.

Method	Geometry3K		PGPS9K	
	Choice	*Completion*	*Choice*	*Completion*
MLLMs				
Claude 3.5 Sonnet [Anthropic, 2024]	56.4	32.0	45.9	27.6
GPT-4o [Hurst et al., 2024]	57.1	46.3	36.3	33.4
Qwen3-VL-32B-Thinking [Bai et al., 2025]	51.7	43.3	72.7	42.1
Neural Solvers				
NGS [Chen et al., 2021]	58.8	35.3	46.1	34.1
Geoformer [Chen et al., 2022]	59.3	36.8	47.3	35.6
PGPSNet [Zhang et al., 2023]	77.9	65.0	70.4	62.7
LANS [Li et al., 2024c]	82.3	71.3	73.8	66.1
Neuro-symbolic Solvers				
InterGPS [Lu et al., 2021]	63.5	50.6	66.2	57.4
E-GPS [Wu et al., 2024]	67.9	–	–	–
GeoDRL [Peng et al., 2023]	68.4	–	–	–
Pi-GPS [Zhao et al., 2025]	77.8	70.6	61.4	69.8
AutoGPS [Ping et al., 2026]	81.6	75.4	81.5	75.3
SynGeo (GPT-4o)	**90.2**	**89.0**	**87.5**	**85.4**
SynGeo (Qwen3-VL-32B-Thinking)	83.69	84.69	78.40	80.20

Implementation details. SynGeo is evaluated with GPT-4o [Hurst et al., 2024] and Qwen3-VL-32B-Thinking [Bai et al., 2025] as multimodal backbones. For initial state construction and symbolic proof-graph reasoning, SynGeo follows the predicate-based formal GPS paradigm of prior systems [Lu et al., 2021, Ping et al., 2026]; implementation details are provided in Appendices C and D. Unless otherwise specified, the full system enables symbolic reasoning, feedback-guided revisiting, and complementary MLLM reasoning. For ablation studies, we evaluate the contribution of each mechanism on both Geometry3K and PGPS9K, and further analyze when the geometric state should be revisited.

4.2 Main Results

We first compare SynGeo with representative MLLM, neural, and neuro-symbolic geometry solvers on Geometry3K and PGPS9K. As shown in Table 1, SynGeo achieves the best performance across all four evaluation settings. With GPT-4o as the backbone, SynGeo obtains 90.2% Choice accuracy and 89.0% Completion accuracy on Geometry3K, and achieves 87.5% Choice accuracy and 85.4% Completion accuracy on PGPS9K. These results demonstrate strong and consistent performance across datasets and answer formats.

Compared with prior neuro-symbolic solvers, SynGeo consistently outperforms AutoGPS, the strongest prior method in Table 1. On Geometry3K, SynGeo improves Choice accuracy from 81.6% to 90.2%, and Completion accuracy from 75.4% to 89.0%. On PGPS9K, it improves Choice accuracy from 81.5% to 87.5%, and Completion accuracy from 75.3% to 85.4%. The larger improvements in the Completion setting suggest that SynGeo is particularly helpful when the solver must generate the answer directly without relying on candidate options.

SynGeo also substantially improves over direct MLLM baselines. Using the same GPT-4o backbone, SynGeo increases Geometry3K Choice accuracy from 57.1% to 90.2%, and Completion accuracy from 46.3% to 89.0%. On PGPS9K, it increases Choice accuracy from 36.3% to 87.5%, and Completion accuracy from 33.4% to 85.4%. With Qwen3-VL-32B-Thinking as the backbone, SynGeo reaches 83.69% Choice and 84.69% Completion accuracy on Geometry3K, and 78.40% Choice and 80.20% Completion accuracy on PGPS9K. These results show that the framework is not

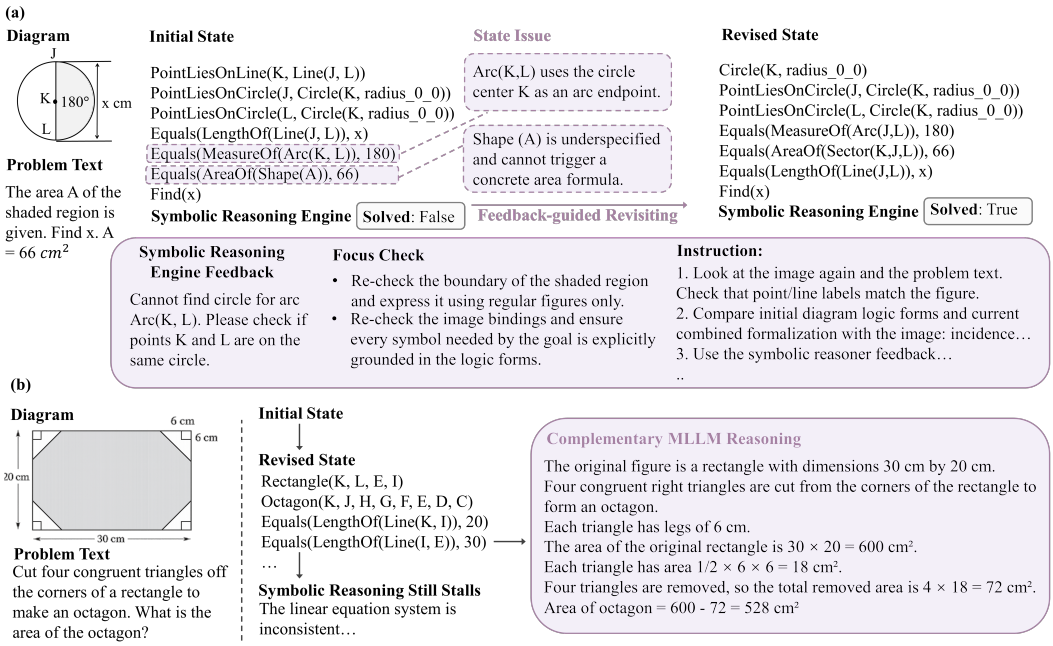


Figure 3: Case study of SynGeo's recovery mechanisms. (a) Feedback-guided Revisiting uses symbolic diagnostics to localize a misgrounded arc predicate and an underspecified region predicate, then repairs the geometric state and enables symbolic reasoning to succeed. (b) Complementary MLLM Reasoning is invoked when the symbolic route remains unresolved, providing an image-grounded reasoning path over the original diagram and problem text.

tied to a single MLLM backbone and support the benefit of allowing image-grounded perception and symbolic reasoning to interact during inference.

4.3 Case Study

This section illustrates how SynGeo recovers from symbolic failures through state repair or complementary image-grounded reasoning. Figure 3 shows two representative cases.

In Figure 3 (a), symbolic reasoning exposes a repairable state-level error. The initial state contains a misgrounded arc predicate and an underspecified shaded-region predicate, causing the symbolic engine to fail. Instead of treating this as a terminal failure, SynGeo uses the diagnostic feedback and focus checks to guide revisiting. The repaired state replaces the invalid arc and grounds the shaded region as a sector, allowing symbolic reasoning to succeed.

In Figure 3 (b), the symbolic route remains unresolved after state construction and revisiting. SynGeo therefore invokes Complementary MLLM Reasoning, which solves the problem by using the original diagram and text to form an image-grounded decomposition. Together, the two cases show that SynGeo can either repair the formal state when the symbolic feedback is actionable, or fall back to complementary reasoning when the formal route remains insufficient.

4.4 Ablation Study

We analyze the key design choices in SynGeo. The ablation study is organized around three questions corresponding to the main mechanisms in Figure 2: whether Complementary MLLM Reasoning helps, whether Feedback-guided Revisiting helps, and when the geometric state should be revisited.

Does Complementary MLLM Reasoning help? Table 2 analyzes the contribution of Complementary MLLM Reasoning using the same inference trajectories with and without the complementary branch. On Geometry3K, complementary reasoning increases Choice accuracy from 75.4% to 90.2% and Completion accuracy from 75.4% to 89.0%. On PGPS9K, it improves Choice accuracy from

Table 2: Effect of Complementary MLLM Reasoning.

Variant	Geometry3K		PGPS9K	
	Choice	*Completion*	*Choice*	*Completion*
w/o Complementary MLLM Reasoning	75.4	75.4	75.7	75.6
w Complementary MLLM Reasoning	90.2	89.0	87.5	85.4

Table 3: Effect of Feedback-guided Revisiting.

Variant	Geometry3K		PGPS9K	
	Choice	*Completion*	*Choice*	*Completion*
w/o Feedback-guided Revisiting	86.4	84.5	85.4	84.3
w Feedback-guided Revisiting	90.2	89.0	87.5	85.4

75.7% to 87.5% and Completion accuracy from 75.6% to 85.4%. These results indicate that complementary image-grounded reasoning provides additional recovery when the symbolic route remains unresolved.

Does Feedback-guided Revisiting help? Table 3 evaluates the contribution of Feedback-guided Revisiting by disabling post-symbolic state revision while retaining symbolic reasoning and Complementary MLLM Reasoning. On Geometry3K, feedback-guided revisiting improves Choice accuracy from 86.4% to 90.2% and Completion accuracy from 84.5% to 89.0%. On PGPS9K, it improves Choice accuracy from 85.4% to 87.5% and Completion accuracy from 84.3% to 85.4%. These results show that feedback-guided revisiting provides consistent improvements by repairing geometric states based on symbolic diagnostics, while symbolic and complementary reasoning remain effective without this revision step.

When should the geometric state be revisited? Table 4 compares two stages for applying revisiting on Geometry3K. Revisiting during formalization performs a pre-revisiting step immediately after the initial multimodal formalization and before symbolic solving. At this stage, the system can only rely on formalization-level signals, such as unresolved placeholders, missing goals, dropped salient predicates, or visually risky relations. It achieves 80.0% Choice accuracy and 79.2% Completion accuracy. In contrast, revisiting after symbolic testing is triggered after the current state has been tested by the symbolic engine. Thus, the revisiting prompt can use concrete solver diagnostics, such as parsing failures, stalled goals, exceptions, or inconsistent equations, to localize the state repair. This post-revisiting strategy reaches 90.2% Choice accuracy and 89.0% Completion accuracy, supporting our design choice of using symbolic testing to guide when and how the geometric state should be revised.

Table 4: Effect of revisiting stage on Geometry3K.

Variant	*Choice*	*Completion*
Revisiting during formalization	80.0	79.2
Revisiting after symbolic testing	**90.2**	**89.0**

5 Conclusion

In this work, we presented SynGeo, a geometry problem-solving framework that connects image-grounded seeing and symbolic proving through a revisable geometric state. Instead of treating formalization as a fixed preprocessing step, SynGeo uses symbolic reasoning to test the current state, expose missing or misgrounded predicates, and trigger feedback-guided revisiting when repair is possible. When symbolic reasoning remains unresolved, complementary MLLM reasoning provides an image-grounded reasoning path. Experiments on Geometry3K and PGPS9K show that this coordination improves over direct MLLM baselines and prior neuro-symbolic solvers, while ablations confirm the contributions of both revisiting and complementary reasoning. Current evaluations are limited to plane geometry, and extending SynGeo to settings such as solid geometry may require richer visual grounding and more dynamic perception–reasoning interaction. More broadly, SynGeo

points to a general direction for multimodal reasoning systems: symbolic failures can serve not only as stopping conditions, but also as feedback for repairing what the system sees.

References

Anthropic. Introducing claude 3.5 sonnet. `https://www.anthropic.com/news/claude-3-5-sonnet`, 2024.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

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.

Jiaqi Chen, Tong Li, Jinghui Qin, Pan Lu, Liang Lin, Chongyu Chen, and Xiaodan Liang. Unigeo: Unifying geometry logical reasoning via reformulating mathematical expression. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 3313–3323, 2022.

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. *arXiv preprint arXiv:2312.11370*, 2023.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Xu Tang, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

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.

Zenan Li, Yifan Wu, Zhaoyu Li, Xinming Wei, Fan Yang, Xian Zhang, and Xiaoxing Ma. Autoformalize mathematical statements by symbolic equivalence and semantic consistency. *Advances in Neural Information Processing Systems*, 37:53598–53625, 2024a.

Zenan Li, Zhi Zhou, Yuan Yao, Yu-Feng Li, Chun Cao, Fan Yang, Xian Zhang, and Xiaoxing Ma. Neuro-symbolic data generation for math reasoning. *Advances in Neural Information Processing Systems*, 37:23488–23515, 2024b.

Zhong-Zhi Li, Ming-Liang Zhang, Fei Yin, and Cheng-Lin Liu. Lans: A layout-aware neural solver for plane geometry problem. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2596–2608, 2024c.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6774–6786, 2021.

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.

Shuai Peng, Di Fu, Yijun Liang, Liangcai Gao, and Zhi Tang. Geodrl: A self-learning framework for geometry problem solving using reinforcement learning in deductive reasoning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13468–13480, 2023.

Bowen Ping, Minnan Luo, Zhuohang Dang, Chenxi Wang, and Chengyou Jia. AutoGPS: Automated geometry problem solving via multimodal formalization and deductive reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PVtZnUh04m.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1466–1476, 2015.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

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. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

Wenjun Wu, Lingling Zhang, Jun Liu, Xi Tang, Yaxian Wang, Shaowei Wang, and Qianying Wang. E-gps: Explainable geometry problem solving via top-down solver and bottom-up generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13828–13837, 2024.

Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. *Advances in neural information processing systems*, 35:32353–32368, 2022.

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, 2024a.

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

Ming-Liang Zhang, Fei Yin, and Cheng-Lin Liu. A multi-modal neural geometric solver with textual clauses parsed from diagram. In *IJCAI*, 2023.

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 *European Conference on Computer Vision*, pages 169–186. Springer, 2024b.

Junbo Zhao, Ting Zhang, Jiayu Sun, Mi Tian, and Hua Huang. Pi-gps: Enhancing geometry problem solving by unleashing the power of diagrammatic information. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1526–1536, 2025.

Appendix Overview

This appendix provides supplementary material for SynGeo and is organized to help readers locate implementation details, additional analyses, and qualitative examples. Appendix A discusses the broader impact of SynGeo for reliable multimodal reasoning and educational use. Appendix B discusses limitations and future directions. Appendix C describes how the initial geometric state is constructed from diagram–text evidence. Appendix D explains symbolic validation, proof-graph expansion, and diagnostic feedback. Appendix **??** reports a progressive construction ablation, complementing the removal ablations in the main paper. Appendix E provides additional case studies for the three main inference routes: Feedback-guided Revisiting, Complementary MLLM Reasoning, and successful symbolic proof. Appendix F collects implementation and reasoning details, including API-based inference, the symbolic solving loop, proof-graph abstraction, feedback fields, revisiting prompts, trigger signals, prompt-to-repair examples, and reasoning complementarity.

A Broader Impact

SynGeo studies geometry problem solving as a coordination between image-grounded perception and symbolic reasoning. Its broader impact lies in suggesting how multimodal reasoning systems can expose, test, and revise intermediate representations rather than only output final answers. By using symbolic failures as diagnostic feedback, SynGeo may help make reasoning processes more transparent and easier to inspect. At the same time, the system can still produce incorrect results due to misgrounded diagrams, incomplete formalization, limited theorem coverage, or unreliable complementary MLLM reasoning. Therefore, SynGeo should be viewed as an assistive reasoning framework rather than an autonomous decision maker, especially in educational or other high-stakes settings where human verification remains necessary.

B Limitations and Future Directions

The current evaluation focuses on plane-geometry benchmarks, including Geometry3K and PGPS9K. While these datasets provide a controlled setting for studying diagram–text grounding, symbolic diagnosis, state repair, and complementary reasoning, more complex settings such as solid geometry may introduce stronger perceptual challenges. For example, three-dimensional relations can be hidden by projection, object boundaries may overlap, and useful constraints may depend on viewpoint, depth, or implicit spatial structure. Extending SynGeo to such settings may require richer visual grounding and more dynamic perception–reasoning interaction.

SynGeo also depends on the coverage of its formal predicate interface and symbolic rule system. The current predicate language and theorem library support many plane-geometry problems, but problems requiring unsupported objects, transformations, or higher-dimensional relations may remain difficult for the symbolic route. Future work may extend the predicate language, broaden the theorem library, and develop richer state representations for more general geometric settings.

Finally, the revisiting strategy can be made more adaptive. The current system triggers revisiting from symbolic diagnostics and uses a bounded number of repair attempts. This is effective for localized repairs, such as invalid arc grounding or underspecified shaded regions. In more complex settings, especially solid geometry, repair may require multi-step revisiting over different visual cues, higher-level spatial interpretations, and groups of dependent predicates. Future work may therefore study adaptive revisiting policies that decide what to inspect, how many times to revisit, and when to switch from state repair to complementary reasoning.

C Initial Geometric State Construction

C.1 Formalization Frontend

SynGeo constructs its initial geometric state using a predicate-based formalization frontend following AutoGPS [Ping et al., 2026]. This frontend is used as an initialization module; SynGeo focuses on how the resulting state is tested, repaired, and complemented during inference.

Given a raw diagram I, the frontend first performs key-point annotation. Following AutoGPS, a pre-trained visual annotation model based on the FPN architecture [Lin et al., 2017] detects geometrically important points, including vertices, intersections, and circle centers, and assigns point labels to them. This produces an annotated diagram

$$I' = (I, P_t, Y),$$

where P_t denotes the detected key points and Y denotes their assigned labels.

The annotated diagram is then processed by a diagram parser. Following AutoGPS, this parser is based on PGDP-Net [Zhang et al., 2022] and extracts diagram-side geometric primitives and relations:

$$(P_g, R) = \mathcal{D}(I').$$

Here, P_g contains primitives such as line segments, circles, and arcs, while R contains relations such as point-line incidence, point-circle membership, parallelism, perpendicularity, and numerical labels.

In parallel, a rule-based text parser converts the problem text T into pseudo logic forms:

$$T' = \mathcal{T}(T) = \{l'_1, l'_2, \ldots, l'_{k'}, \mathrm{Find}(q')\}.$$

The text-side output is a pseudo formalization because some expressions may not yet be fully grounded in the diagram. For example, a shaded-area expression may appear as `AreaOf(Shape($))`, where the actual geometric region must still be resolved from visual evidence.

The diagram-side parses and text-side pseudo logic forms form the pre-formalization

$$F = (P_g, R, T').$$

A multimodal alignment module then takes the annotated diagram I' and the pre-formalization F as input. It aligns textual references with visual entities, resolves placeholders, fills missing global information, and rewrites ambiguous expressions into solver-compatible predicates. The output is the initial geometric state

$$S_0 = \mathcal{A}(I', F) = \{l_1, l_2, \ldots, l_m, \mathrm{Find}(q)\}.$$

SynGeo treats S_0 as an initial hypothesis rather than a fixed formalization, because errors can still arise from key-point annotation, diagram parsing, text parsing, or cross-modal alignment.

D Symbolic Proof Graph and Diagnostic Feedback

D.1 Symbolic Validation and Proof-Graph Expansion

SynGeo uses an AutoGPS-style symbolic reasoning backend to test and solve over the current geometric state [Ping et al., 2026]. Given a state S_t, the symbolic engine first validates whether its predicates are syntactically well-formed, self-consistent, and sufficiently complete for symbolic solving. Some implicit relations can be added by symbolic closure. For example, line-level incidence or perpendicularity may imply corresponding segment-level relations. However, validation cannot repair predicates whose visual grounding is wrong, such as an arc whose endpoint is actually a circle center. Such failures are returned as diagnostic feedback for revisiting.

After validation, the symbolic engine constructs a proof graph. The proof graph is a directed hypergraph $G_t = (V_t, E_t)$, where each node $v \in V_t$ corresponds to a known or derived predicate, equation, or quantity, and each hyperedge $e \in E$ corresponds to a theorem application or algebraic transformation. A reasoning step has the form

$$\{p_1, \ldots, p_m\} \xrightarrow{\rho} \{q_1, \ldots, q_n\},$$

where p_i are premise predicates, q_i are conclusion predicates, and ρ is a geometric theorem or algebraic operation. The graph is initialized by connecting a start node to all predicates in S_t as known facts.

The engine expands the graph with two complementary operations. Deductive expansion applies geometry rules, such as triangle definitions, angle relations, circle and arc properties, similarity, congruence, polygon properties, and area formulas. Algebraic expansion performs stepwise symbolic computation, including substitution, simplification, numerical evaluation, and equation solving. The symbolic route succeeds when the expanded graph contains a solution literal $\texttt{Equals}(q, v)$, where q is the target quantity and v is the derived value. If a solution is found, the final proof can be obtained by tracing a compact subgraph from the start node to the solution node.

Table 5: Progressive performance summary of SynGeo. The SynGeo entries report cumulative correct-answer coverage within the full evaluation.

Variant	Geometry3K		PGPS9K	
	Choice	*Completion*	*Choice*	*Completion*
Base (GPT-4o)	57.1	46.3	36.3	33.4
+ Symbolic Reasoning Engine	72.7	73.0	73.7	73.8
+ Feedback-guided Revisiting	75.4	75.4	75.7	75.6
+ Complementary MLLM Reasoning	**90.2**	**89.0**	**87.5**	**85.4**

D.2 Diagnostic Feedback for Revisiting

SynGeo uses the symbolic engine not only as a solver, but also as a diagnostic component. When validation or proof-graph expansion fails, the engine returns feedback that can identify unparsable predicates, malformed objects, inconsistent relations, missing target-related predicates, invalid grounding, or reasoning stagnation. These signals are passed to the revisiting module together with the original diagram, problem text, and current state.

For example, if the engine reports that it cannot find a circle for an arc predicate, revisiting re-checks which points lie on the circle and whether the arc endpoints are correctly grounded. If an area formula cannot be instantiated because the state contains a placeholder such as `Shape(A)`, revisiting re-checks the boundary of the shaded region and rewrites it as a regular geometric object when possible. This is the point at which SynGeo differs from a fixed formalization pipeline: symbolic failure is not only a stopping condition, but also a signal for repairing the geometric state.

Table 5 presents a progressive view of SynGeo, illustrating how correct-answer coverage increases as its reasoning mechanisms are incorporated.

The direct GPT-4o baseline achieves 57.1% Choice and 46.3% Completion accuracy on Geometry3K, and 36.3% Choice and 33.4% Completion accuracy on PGPS9K. Within SynGeo, Symbolic Reasoning achieves 72.7% and 73.0% on Geometry3K, and 73.7% and 73.8% on PGPS9K, respectively.

Feedback-guided Revisiting further increases cumulative accuracy to 75.4% in both settings on Geometry3K, and 75.7% Choice and 75.6% Completion on PGPS9K. With Complementary MLLM Reasoning, the full system achieves 90.2% Choice and 89.0% Completion accuracy on Geometry3K, and 87.5% Choice and 85.4% Completion accuracy on PGPS9K.

These results support the progressive design of SynGeo: symbolic reasoning provides a structured deductive foundation, feedback-guided revisiting repairs state-level errors, and complementary reasoning improves performance when symbolic deduction remains unresolved.

E Case Studies

This section provides qualitative examples for the main inference routes of SynGeo. The cases are organized into three groups so that readers can distinguish different system behaviors. Feedback-guided Revisiting cases show how symbolic feedback helps repair solver-incompatible states. Complementary MLLM Reasoning cases show how SynGeo recovers when the symbolic route remains unresolved but the diagram–text evidence still supports an image-grounded solution path. Successful symbolic proof cases show how a solver-compatible state leads directly to a proof-derived answer.

F Implementation and Reasoning Details

This section collects implementation and reasoning details that support the method description in the main paper. It first describes the inference setting and symbolic solving loop, then clarifies several table-based components used by SynGeo: the proof-graph abstraction, symbolic feedback fields, revisiting prompt, revisiting triggers, prompt-to-repair examples, and the complementarity among inference routes.

Algorithm 2 Symbolic reasoning over a geometric state

Require: geometric state S_t, target q, maximum depth D
Initialize proof graph $\mathcal{G}$ from predicates in S_t
if q is already derivable in $\mathcal{G}$ **then**
  return proof-derived answer
end if
for $d = 1, \ldots, D$ **do**
  if d is odd **then**
    Perform algebraic expansion
      substitute known representatives, evaluate constants, solve linear and univariate equations
  else
    Perform deductive expansion
      apply definitions, theorem rules, and equivalence transitive closure
  end if
  if q is derivable **then**
    Extract a minimal proof subgraph for q
    return proof-derived answer and proof trace
  end if
end for
return unsolved status and symbolic diagnostics

F.1 API-Based Inference

All MLLM components in SynGeo are run through API-based inference without training or fine-tuning new model parameters. The experiments use fixed model backbones and prompting protocols for diagram–text understanding, state revisiting, and complementary reasoning, while the symbolic reasoning backend is executed locally. We report the model backbones, prompts, datasets, evaluation settings, and symbolic-solver configuration in the paper and appendix, but do not report detailed token usage, API-call counts, memory usage, or per-run execution time.

F.2 Symbolic Solving Loop

SynGeo uses the symbolic reasoning engine as a state-testing and proof-generation component. We do not modify the symbolic theorem system itself; instead, SynGeo uses its proof results and failure diagnostics to decide whether the current geometric state is solver-compatible or should be revisited.

Given a geometric state S_t, the symbolic engine initializes a proof graph whose nodes are logic-form predicates and whose hyperedges correspond to theorem applications, definitions, substitutions, or algebraic transformations. The solver first checks whether the target is already available. If not, it alternates between deductive expansion and algebraic expansion until the goal is reached or the search budget is exhausted. Algorithm 2 summarizes this symbolic solving loop.

F.3 Proof Graph Abstraction

This subsection clarifies the proof-graph abstraction used by the symbolic engine. The proof graph records how initial predicates, derived predicates, theorem applications, and algebraic operations are connected during symbolic solving. Table 6 summarizes the graph components used by SynGeo and explains how each component participates in proof search.

F.4 Symbolic Diagnostic Feedback

A key role of symbolic reasoning in SynGeo is diagnostic. When symbolic solving succeeds, SynGeo uses the proof-derived answer; when it fails, the wrapper returns structured feedback explaining why the current state cannot support proof search. Table 7 summarizes the feedback fields passed to the revisiting module and explains how they guide state repair.

Table 6: Proof graph abstraction used by the symbolic engine.

Component	Meaning
Node	A logic-form predicate, such as `Equals(LengthOf(Line(X,Y)),7)`, `IsMidsegmentOf(Line(X,Y),Triangle(R,S,T))`, or `AreaOf(Sector(X,W,Y))`.
Hyperedge	A reasoning step connecting premise predicates to conclusion predicates. It may correspond to a geometry theorem, a definition, substitution, constant evaluation, or equation solving.
Deductive expansion	Applies geometry definitions and theorem rules, then closes equivalent predicates through transitivity.
Algebraic expansion	Substitutes known values, evaluates constant expressions, solves linear equation systems, and solves univariate equations when applicable.
Goal check	For `Find(q)`, the solver checks whether a predicate of the form `Equals(value,q)` can be derived. For proof goals, it checks whether the target predicate is matched in the proof graph.
Proof extraction	Once the target is reached, the engine extracts a minimal subgraph leading to the target and converts it into a step-by-step proof trace.

Table 7: Symbolic feedback fields used by SynGeo.

Field	Role in SynGeo
`solved`	Indicates whether the target is derived from the current state.
`solution_predicate`	Stores the derived answer predicate, such as `Equals(14,LengthOf(Line(S,T)))`.
`proof`	Stores the natural-language proof trace extracted from the minimal proof subgraph.
`failed_to_parse`	Lists malformed or unsupported predicates. These are prioritized during revisiting.
`goal_repr`	Records the target that the solver attempted to derive, helping revisiting focus on target-related objects.
`error`	Stores solver exceptions, such as invalid arc grounding, unsupported equations, inconsistent equation systems, or timeout-related failures.
`revisit_reasons`	Converts symbolic failure into natural-language reasons for state repair.
`focus_checks`	Converts symbolic failure into image re-check instructions, such as re-checking arc endpoints, circle centers, shaded-region boundaries, or numeric label ownership.

F.5 Feedback-Guided Revisiting Prompt

This subsection presents the prompt template used for Feedback-guided Revisiting. The prompt is designed to make revisiting state-focused rather than solution-focused: the model is asked to repair the logic-form state using symbolic feedback and diagram–text evidence, not to produce a free-form answer. The template also enforces minimal edits and visual grounding to reduce unsupported predicate insertion.

Feedback-guided revisiting prompt template

Input fields. The revisiting prompt is instantiated with: the diagram image, problem text, predicate definitions, initial diagram logic forms, initial text logic forms, the current combined formalization, uncertainty score, revisit reasons, focused visual checks, symbolic reasoner feedback, and retrieved memory guidelines.

Instruction template. Revisit the image and problem text. Compare the initial diagram logic forms and the current combined formalization against the image. Use the symbolic reasoner feedback to locate where parsing or solving failed. If there are failed-to-parse predicates, fix them first. If the solver cannot derive the goal, add only visually grounded constraints and do not invent hidden equalities. Prefer minimal edits: keep correct predicates and only repair or replace wrong or missing ones. End with exactly one plaintext code block containing the full corrected logic forms, one predicate per line.

Table 8: Trigger signals for Feedback-guided Revisiting. SynGeo invokes revisiting when the symbolic route exposes a repairable state-level issue.

Trigger signal	Source	Interpretation
`failed_to_parse`	Symbolic parser	Some predicates are syntactically invalid or unsupported by the symbolic engine. The revisiting module is asked to rewrite them into valid logic forms.
Symbolic exception	Symbolic solver	The solver raises an exception during proof construction or algebraic reasoning, such as invalid arc grounding, unsupported equations, or inconsistent equation systems.
Goal not derived	Proof graph search	The current state is parseable, but the target predicate or target value cannot be reached, suggesting missing constraints or insufficient grounding.
Timeout / stagnation	Solver controller	Deductive or algebraic expansion fails to make progress within the search budget, often indicating an over-expanded or noisy state.
Placeholder forms	Formalization state	Predicates such as `Shape(A)` or unresolved symbols remain in the state and must be grounded as regular geometric objects.
Risky visual relations	Revisit analyzer	Relations involving arcs, circles, tangency, perpendicularity, shaded regions, or angle labels are marked as visually risky and re-checked against the image.

Table 9: Prompt-to-repair examples for Feedback-guided Revisiting. Symbolic feedback is converted into focused image re-checks, which then repair the geometric state.

Case	Feedback	Focused re-check	Repair action
2704	The symbolic parser rejects `Equals(AreaOf(Shape(A)),357)`.	Re-check the labeled area and rewrite unresolved placeholders as concrete geometric objects.	Replace `AreaOf(Shape(A))` with `AreaOf(Triangle(A,B,D))`, enabling the triangle-area equation $357 = 34x/2$.
2839	The solver cannot find a circle for `Arc(K,L)`, because K is the circle center rather than an arc endpoint.	Re-check the circle center, on-circle points, valid arc endpoints, and shaded-region boundary.	Replace `Arc(K,L)` with `Arc(J,L)`, and ground the shaded region as `Sector(K,J,L)`.
2431	The symbolic engine times out after 1800 seconds under the current search depth.	Re-check whether the state contains unnecessary auxiliary equalities or noisy relations that enlarge proof search.	Simplify the state to the essential regular-hexagon predicate and side-length constraints, allowing a concise law-of-cosines derivation.

F.6 Revisiting Trigger Signals

This subsection explains when SynGeo decides to invoke Feedback-guided Revisiting. The trigger signals come from different stages of the symbolic route, including parsing, proof-graph search, solver exceptions, and timeout or stagnation. Table 8 lists the trigger types, their sources, and how SynGeo interprets them as repairable state-level issues.

F.7 Prompt-to-Repair Examples

This subsection illustrates how symbolic feedback is converted into concrete state repairs. Each row corresponds to a representative revisiting case: the symbolic engine first reports a failure signal, SynGeo converts it into focused visual checks, and the revisiting module edits the predicate state accordingly. Table 9 provides examples of this feedback-to-repair chain.

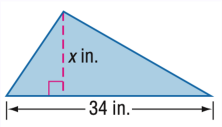
Table 10: Complementarity between symbolic reasoning, Feedback-guided Revisiting, and Complementary MLLM Reasoning. The three routes address different failure modes rather than acting as redundant solvers.

Route	Best suited for	Typical evidence used	Example behavior
Symbolic reasoning	Solver-compatible states with sufficient predicates and theorem-applicable structure.	Formal predicates, proof graph expansion, theorem rules, and algebraic equations.	Directly proves mid-segment, rectangle-area, Heron, or sector-area cases with checkable theorem-level derivations.
Feedback-guided revisiting	States that are close to solvable but contain malformed, missing, noisy, or visually misgrounded predicates.	Symbolic diagnostics plus the original image and text.	Repairs `Shape(A)` into a triangle, fixes invalid arc endpoints, or simplifies an over-expanded regular-hexagon state.
Complementary MLLM reasoning	Cases where the symbolic route remains unresolved, but the diagram supports a concise visual or high-level geometric strategy.	Diagram image, problem text, answer choices, and image-grounded reasoning.	Uses rectangle-minus-triangles, circle-minus-polygon, similar-figure area ratio, or inscribed-circle diameter reasoning when proof search stalls.

F.8 Reasoning Complementarity

This subsection summarizes how the three inference routes in SynGeo complement one another. Symbolic reasoning, Feedback-guided Revisiting, and Complementary MLLM Reasoning are not redundant solvers; they are used for different state conditions and failure modes. Table 10 compares when each route is most useful, what evidence it relies on, and what behavior it typically produces.

Feedback-guided Revisiting: Grounding an Underspecified Area Predicate



Problem. Find x. $A = 357$ in^2.

Choices. A. 21, B. 22, C. 23, D. 24. Ground truth: A.

Initial issue. The initial formalization contains the unresolved predicate `Equals(AreaOf(Shape(A)), 357)`. The symbolic parser cannot use `Shape(A)` as a regular geometric object, so the area constraint cannot participate in proof-graph expansion.

Symbolic feedback and focused re-check

Feedback. The symbolic reasoner rejects the predicate `Equals(AreaOf(Shape(A)), 357)`.

Focused re-check. Repair predicates that the symbolic reasoner could not parse, especially malformed equations and unresolved placeholders. Re-check and rewrite the area predicate in valid geometry syntax.

State repair after revisiting

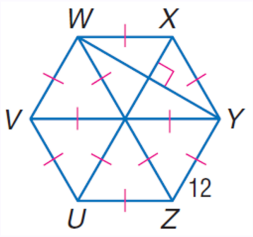
```
Triangle(A, B, D)
Line(A, D)
Line(A, C)
Line(B, C)
Line(C, D)
PointLiesOnLine(C, Line(A, D))
Perpendicular(Line(A, C), Line(B, C))
Equals(LengthOf(Line(C, B)), x)
Equals(LengthOf(Line(A, D)), 34)
Equals(AreaOf(Triangle(A, B, D)), 357)
Find(x)
```

Symbolic proof after revisiting

1. The revisiting step grounds the ambiguous region `Shape(A)` as $\triangle ABD$.
2. Since C lies on AD and $BC \perp AD$, BC is the height of $\triangle ABD$.
3. The triangle area formula gives
$$Area(\triangle ABD) = \frac{AD \cdot BC}{2}.$$
4. Substitute $AD = 34$, $BC = x$, and $Area(\triangle ABD) = 357$:
$$357 = \frac{34x}{2}.$$
5. Solving the equation gives
$$x = 21.$$
6. The repaired symbolic route therefore predicts choice A.

Figure 4: Feedback-guided Revisiting: grounding an underspecified area predicate.

Feedback-guided Revisiting: Simplifying an Over-Expanded Formal State



Problem. In regular hexagon $UVWXYZ$, the side length is 12 cm. Find WY.

Choices. A. 12, B. $6\sqrt{6}$, C. $12\sqrt{3}$, D. 36. Ground truth: C.

Initial issue. The symbolic engine times out after 1800 seconds. Here the problem is not a malformed predicate, but an over-expanded state that makes proof search unnecessarily large.

Symbolic feedback and focused re-check

Feedback. Timeout after 1800 seconds with `reasoner_max_depth=20`.

Focused re-check. Consider simplifying the formalization or reducing ambiguous relations so the solver search space stays smaller.

State repair after revisiting

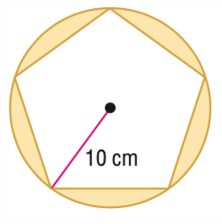
```
Regular(Hexagon(U,V,W,X,Y,Z))
Equals(LengthOf(Line(U,V)), 12)
Equals(LengthOf(Line(V,W)), 12)
Equals(LengthOf(Line(W,X)), 12)
Equals(LengthOf(Line(X,Y)), 12)
Equals(LengthOf(Line(Y,Z)), 12)
Equals(LengthOf(Line(Z,U)), 12)
Find(LengthOf(Line(W,Y)))
```

Symbolic proof after revisiting

1. The revisiting step keeps only the essential regular-hexagon structure and side-length constraints.
2. In a regular hexagon, the interior angle at X is 120°, so
$$\angle WXY = 120^\circ.$$
3. The adjacent sides satisfy $WX = XY = 12$.
4. Apply the law of cosines in $\triangle WXY$:
$$WY^2 = WX^2 + XY^2 - 2 \cdot WX \cdot XY \cos(120^\circ).$$
5. Substitute $WX = XY = 12$:
$$WY^2 = 12^2 + 12^2 - 2 \cdot 12 \cdot 12 \cdot \left(-\frac{1}{2}\right) = 432.$$
6. Therefore,
$$WY = 12\sqrt{3} \approx 20.78.$$
7. The repaired symbolic route predicts choice C.

Figure 5: Feedback-guided Revisiting: simplifying an over-expanded formal state.

Complementary MLLM Reasoning Case: Circle Minus Regular Pentagon



Problem. Find the area of the shaded region formed by the circle and regular polygon. Round to the nearest tenth.

Choices. A. 76.4, B. 237.8, C. 266.6, D. 314.2. Ground truth: A.

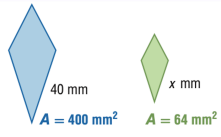
Symbolic status. The symbolic engine cannot create an equation for an unsupported angle predicate, reporting an error around `Angle(A,C,D)`. The formal route stalls even though the visual structure directly suggests a regular-pentagon area subtraction.

Complementary MLLM reasoning trace

1. The diagram is interpreted as a circle of radius 10 with an inscribed regular pentagon.
2. The circle area is
$$\pi r^2 = 100\pi \approx 314.159.$$
3. The regular pentagon is decomposed into 5 congruent isosceles triangles. Each triangle has central angle
$$\frac{360^\circ}{5} = 72^\circ.$$
4. The area of one triangle is
$$\frac{1}{2} \cdot 10^2 \cdot \sin 72^\circ = 50 \sin 72^\circ.$$
Hence the pentagon area is
$$5 \cdot 50 \sin 72^\circ \approx 237.775.$$
5. The shaded region is the circle area minus the pentagon area:
$$100\pi - 237.775 \approx 76.384.$$
6. Rounding gives 76.4, corresponding to choice A.

Figure 6: A complementary MLLM case where the image-grounded route uses a regular-polygon decomposition that the symbolic engine fails to instantiate.

Complementary MLLM Reasoning Case: Similar-Figure Area Ratio



Problem. For the pair of similar figures, use the given areas to find x.

Choices. A. 6.4, B. 16, C. 100, D. 250. Ground truth: B.

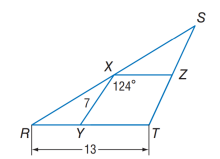
Symbolic status. The symbolic engine reports an inconsistent equation system over expanded angle constraints. The high-level similar-figure relation is present, but proof search becomes entangled in unnecessary geometric constraints.

Complementary MLLM reasoning trace

1. The two figures are similar. Their areas are 64 mm^2 and 400 mm^2.
2. The corresponding side of the larger figure is 40 mm, and the corresponding side of the smaller figure is x.
3. For similar figures, the area ratio equals the square of the side-length ratio:
$$\frac{64}{400} = \left(\frac{x}{40}\right)^2.$$
4. Since $\frac{64}{400} = 0.16$, we have
$$\frac{x}{40} = \sqrt{0.16} = 0.4.$$
5. Therefore
$$x = 40 \cdot 0.4 = 16,$$
corresponding to choice B.

Figure 7: A complementary MLLM case where the concise similar-figure invariant avoids an over-expanded symbolic equation system.

Symbolic Proof Case: Midsegment Theorem



Problem. XY and XZ are midsegments of $\triangle RST$. Find ST.

Choices. A. 7, B. 13, C. 14, D. 20. Ground truth: C.

Geometric state.

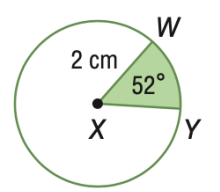
```
PointLiesOnLine(X, Line(R, S))
PointLiesOnLine(Y, Line(R, T))
Equals(LengthOf(Line(X, Y)), 7)
IsMidsegmentOf(Line(X,Y), Triangle(R,S,T))
Find(LengthOf(Line(S,T)))
```

Symbolic proof trace

1. **Known facts.** The solver has $XY = 7$, X on RS, Y on RT, and XY is a midsegment of $\triangle RST$.
2. **Evaluation.** From $XY = 7$, compute
$$2 \cdot XY = 14.$$
3. **Theorem application.** By the triangle midsegment theorem,
$$ST = 2 \cdot XY.$$
4. **Transitivity.** Since $2 \cdot XY = 14$ and $ST = 2 \cdot XY$, the solver derives
$$ST = 14.$$
5. The proof-derived answer is 14, corresponding to choice C.

Figure 8: A successful symbolic proof case where a theorem-level relation directly closes the target.

Symbolic Proof Case: Sector Area Formula



Problem. Find the area of the shaded sector. Round to the nearest tenth.

Choices. A. 0.9, B. 1.8, C. 10.8, D. 12.6. Ground truth: B.

Geometric state.

```
Circle(X, radius_0_0)
Equals(RadiusOf(Circle(X)), 2)
Equals(MeasureOf(Angle(W, X, Y)), 52)
Sector(X, W, Y)
Find(AreaOf(Sector(X, W, Y)))
```

Symbolic proof trace

1. **Known facts.** The solver has radius $r = 2$, central angle $52°$, and the target sector.
2. **Angle normalization.** The engine represents the angle in radians:
$$\theta = \pi \cdot \frac{52}{180}.$$
3. **Theorem application.** Apply the sector-area formula:
$$\text{Area} = \frac{r^2\theta}{2}.$$
4. **Substitution.**
$$\text{Area} = \frac{2^2 \cdot \pi \cdot (52/180)}{2} \approx 1.815.$$
5. Rounded to the nearest tenth, the answer is 1.8, corresponding to choice B.

Figure 9: A successful symbolic proof case where formal angle normalization and sector-area computation produce the final answer.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions and scope. They present SynGeo as a geometry problem-solving framework that combines multimodal state construction, symbolic reasoning, feedback-guided state repair, and complementary MLLM reasoning, and the claims are supported by the reported results and ablations on Geometry3K and PGPS9K.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations of the proposed framework, including its focus on plane-geometry benchmarks, dependence on MLLM-based diagram–text grounding, coverage limits of the predicate interface and theorem library, and the need for more adaptive revisiting policies in more complex geometry settings.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present formal theoretical results such as theorems or proofs. Its contributions are methodological and empirical.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the datasets, evaluation settings, model backbones, prompting protocols, symbolic-solver configuration, ablation settings, and implementation details needed to reproduce the main experimental results. The appendix further provides details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The supplementary materials include anonymized code, prompts, processed evaluation files, and reproduction instructions for the reported experiments. These materials will be made publicly available after the review process.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer.
   - The instructions should contain the exact command and environment needed to run to reproduce the results.
   - The authors should provide instructions on data access and preparation.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines.
   - At submission time, to preserve anonymity, the authors should release anonymized versions if applicable.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The paper specifies the experimental setting, including datasets, task formats, compared methods, model backbones, prompting protocols, symbolic reasoning configuration, and evaluation metrics. Additional implementation details are provided in the appendix and supplementary materials.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The paper reports quantitative comparisons and ablation results across Geometry3K and PGPS9K, but it does not conduct formal statistical significance tests or consistently report confidence intervals for all experiments. We therefore answer No and interpret the empirical comparisons as benchmark-level evidence rather than formal significance claims.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated.
- The method for calculating the error bars should be explained.
- The assumptions made should be given.
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [No]

Justification: The paper describes the model backbones and API-based inference settings used for the MLLM components, together with the locally executed symbolic-solver setup and evaluation configuration. However, it does not report detailed compute-resource information such as token usage, number of API calls, memory usage, or execution time for each experiment, and we therefore answer No.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research is conducted on geometry problem solving and does not involve harmful applications, sensitive personal data, or human-subject experiments. The submitted materials are anonymized and the work is intended for scientific evaluation of reasoning systems.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses the potential positive impact of improving reliable geometry reasoning and educational problem-solving support. It also acknowledges possible risks, such as over-reliance on automated solvers, propagation of incorrect reasoning traces, and the need for human oversight in educational or high-stakes use.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include malicious or unintended uses, fairness considerations, privacy considerations, and security considerations.
    - The authors should consider possible harms that could arise when the technology is used as intended, when it gives incorrect results, and from misuse.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk pretrained model, image generator, or scraped sensitive dataset. The released supplementary materials, if any, consist of anonymized code, prompts, processed evaluation files, and reproduction scripts for geometry problem-solving experiments.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards.
    - Datasets that have been scraped from the Internet could pose safety risks.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

Justification: The paper cites the original sources of the datasets, baselines, symbolic tools, and model APIs used in the experiments. Appendix X summarizes the corresponding asset sources, versions or access paths where available, and license or terms-of-use information.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license should be included for each asset.
- For scraped data from a particular source, the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not introduce or release a new standalone dataset, model, or reusable artifact beyond the manuscript and appendix materials.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets if applicable.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects. The evaluations are conducted using benchmark datasets, automatic accuracy metrics, symbolic reasoning outcomes, and model-based reasoning analyses rather than participant studies.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or human-subject research, and therefore IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval or equivalent may be required for any human subjects research.
- The authors are expected to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs and multimodal LLMs are important components of the proposed geometry problem-solving framework, rather than being used only for writing assistance. The paper describes their roles in diagram–text understanding, predicate-state construction or revision, and complementary image-grounded reasoning, with model configurations and prompting details provided in the method section and appendix.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.