Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.080333

ARTICLE

KG-HoT: Knowledge-Grounded Hybrid Chain-of-Thought for Geometry Problem Solving

Meihuizi Jia[1,*], **Hongyan Ran**[1] **and Shanshan Li**[2]

[1]School of Artificial Intelligence and Computer Science (School of Software), Northwest Normal University, Lanzhou, China
[2]Beijing Jinghang Research Institute of Computing and Communication, Beijing, China

*Corresponding Author: Meihuizi Jia. Email: jiameihuizi@nwnu.edu.cn

Received: 06 February 2026; Accepted: 06 May 2026; Published: 15 June 2026

ABSTRACT: Large language models (LLMs) have demonstrated considerable ability in solving various tasks via Chain-of-Thought (CoT) prompting, which has precipitated extensive research into their application for complex mathematical reasoning problems. However, current research on mathematical reasoning with CoT predominantly focuses on textual mathematical tasks, such as math word problems, while paying limited attention to multimodal geometric scenarios. To bridge this gap, we propose KG-HoT, a model that harnesses the generative and comprehension capabilities of Multimodal large language models (MLLMs) to enhance complex geometric problem-solving in multimodal systems. Our knowledge-grounded approach enables MLLMs to generate hybrid chains-of-thought operating on dual tracks—language-based reasoning and program-based reasoning—which serve as teaching signals for smaller models. Furthermore, we design an instruction tuning framework that trains these dual reasoning tracks collaboratively within a unified architecture, enabling mutual enhancement and efficient knowledge distillation for complex geometric problem solving. Extensive experimental results demonstrate that KG-HoT achieves superior performance compared to existing approaches on multiple geometry problem-solving benchmarks.

KEYWORDS: Geometry problem solving; multimodal large language models; chain-of-thought; mathematical reasoning; instruction tuning

1 Introduction

Geometry Problem Solving (GPS) represents a pivotal and long-standing challenge within artificial intelligence, requiring the integration of advanced mathematical reasoning, geometric visual understanding, and domain knowledge application. Its importance and complexity have drawn increasing attention from both Computer Vision (CV) and Natural Language Processing (NLP) communities [1–3].

Research in geometry problem solving has evolved from rule-based matching and symbolic reasoning to deep learning approaches. Symbolic solvers [2,4–6] typically employ syntactic parsers to translate problems and diagrams into formal languages, then execute symbolic reasoning using techniques such as path searching and condition matching. As shown in Fig. 1a, the geometric elements, relational constraints, and target objective are first transformed into formal language representations, followed by systematic derivation to yield the final solution. Despite their interpretability, these methods suffer from complex rule engineering requirements and poor generalization to real-world unstructured data. To address these limitations, neural solvers [1,3,7,8] employ hybrid encoders to jointly embed diagrams and text, generating solution programs through sequence modeling. As illustrated in Fig. 1b, the approach begins with predefining basic geometric

relations (e.g., *Equal*, *Double*), arithmetic operators (e.g., *Add*, *Minus*), and constants. Subsequently, the model generates a solution program, for instance "*minus C_3 N_0 minus C_2 V_0*", which upon execution yields the final result. Despite achieving some success, annotating solution programs requires domain expert guidance, making the process expensive and time-consuming. Moreover, these models demonstrate limited understanding of geometric knowledge and generate solution programs without explicit reasoning processes.

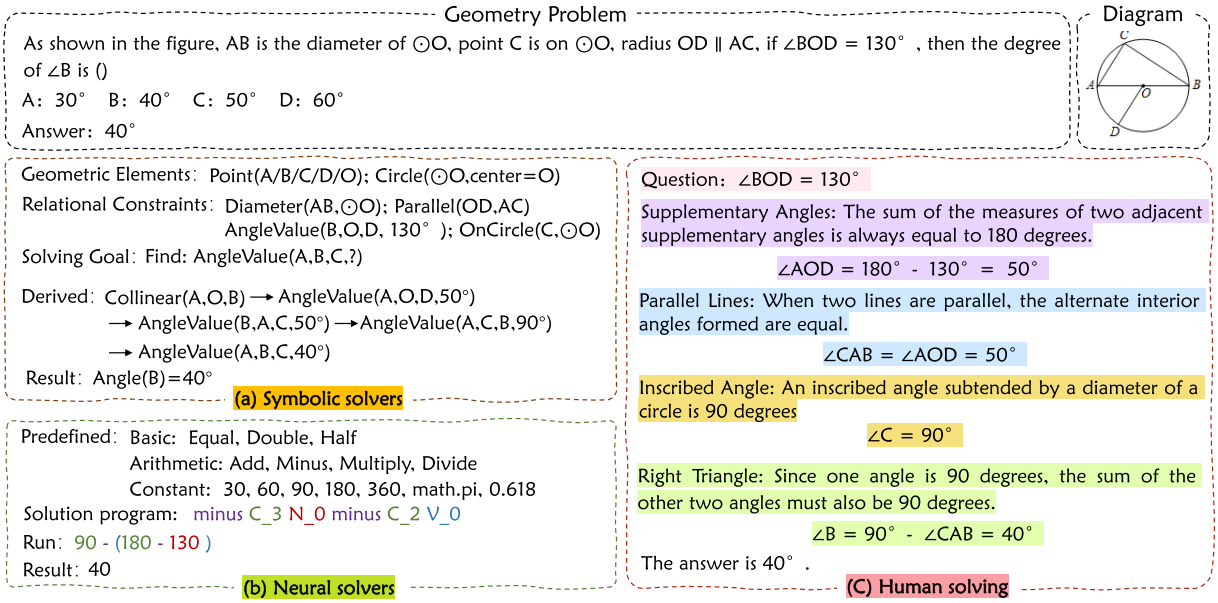

Figure 1: Comparison of three approaches for geometric reasoning. (**a**) symbolic solvers using predefined rules, (**b**) neural solvers with solution programs, (**c**) human solving with step-by-step theorem-based reasoning.

Recent breakthroughs in large language models [9,10] have introduced new research paradigms for mathematical reasoning problems [11–14]. These models advance mathematical reasoning tasks, such as math word problems, through Chain-of-Thought (CoT) reasoning techniques that generate intermediate reasoning steps before deriving final answers. However, text-only processing approaches exhibit inherent limitations in geometric diagrams understanding and spatial relationship reasoning. While several studies have explored Multimodal Chain-of-Thought (MCoT) reasoning [15–18], these efforts primarily target natural science domains. Multimodal geometry problems present unique challenges due to their complex textual descriptions, specialized geometric information, and implicit domain knowledge. When solving complex geometry problems, human experts seamlessly integrate textual and diagrammatic analysis, apply relevant geometric theorems, and construct solution paths. As shown in Fig. 1c, humans start from the given condition $\angle BOD = 130°$ and derive the answer through step-by-step reasoning using geometric theorems and diagram relationships. For example, to calculate $\angle CAB$, they utilize *Supplementary Angles* and *Parallel Lines* theorems, combined with the relationships in the diagram where $\angle AOD$ and $\angle BOD$ are supplementary angles, and $\angle AOD$ and $\angle CAB$ are alternate interior angles. This human problem-solving process reveals three core capabilities that effective models must possess: 1) multimodal understanding to accurately parse geometric elements and spatial relationships across modalities; 2) knowledge grounding to identify applicable theorems and align them with visual elements; 3) hybrid reasoning to combine symbolic manipulation with precise numerical computation.

To enhance multimodal geometry problem-solving capabilities and effectively bridge the gap between existing methods and human cognitive processes, we propose the **K**nowledge-**g**rounded **H**ybrid **C**hain-**o**f-**T**hought (**KG-HoT**) paradigm for Geometry Problem Solving. Our approach consists of two complementary stages: 1) generating knowledge-grounded hybrid reasoning chains from multimodal large language models as teaching signals; 2) distilling these signals into lightweight student models for efficient geometric reasoning. In the teaching signal generation stage, we construct diverse reasoning chains through zero-shot instructions. Specifically, guided by geometric knowledge, we build knowledge paths from concepts to theorems to visual mappings, and implement Python functions for programmatic theorem invocation. By integrating geometric problems, diagram information, standard answers, and geometric knowledge, we guide MLLMs to generate two complementary reasoning chains: 1) Knowledge-grounded Language Chain-of-Thought (KG-CoT), featuring a three-tier progressive structure comprising geometric concept chains (identifying fundamental concepts), theorem chains (determining reasoning rules), and mapping chains (establishing knowledge-diagram associations), ultimately producing solution paths; 2) Knowledge-grounded Program Chain-of-Thought (KG-PoT), transforming reasoning into executable Python sequences that combine semantic clarity with computational precision. To achieve knowledge transfer, we propose a multi-chain learning strategy that jointly optimizes KG-CoT and KG-PoT within a unified framework, enabling synergistic enhancement through instruction tuning. The complementary reasoning paths–natural language and programmatic–grounded in shared geometric knowledge collectively strengthen the student model's problem-solving capabilities. Experimental results demonstrate significant performance improvements across multiple geometric benchmarks. Our contributions are summarized as follows:

- We propose KG-HoT framework, a knowledge distillation framework that transfers geometric reasoning capabilities from large models to lightweight models.
- We design a multi-chain joint learning strategy that combines the three-tier progressive reasoning of KG-CoT with the programmatic verification of KG-PoT, enhancing both the accuracy and interpretability of lightweight models in geometry problem solving.
- We conduct comprehensive experiments across multiple geometry benchmarks to evaluate our approach. Results demonstrate that KG-HoT achieves significant performance improvements over baseline methods and delivers competitive results compared to existing advanced models.

2 Related Work

2.1 Geometry Problem Solving

Multimodal geometric reasoning is a challenging mathematical reasoning task, which can be categorized into symbolic solvers [2,4–6] and neural solvers [1,3,7,8,19]. Most neural solvers follow the encoder-decoder framework and focus on dataset construction. Many studies [1,3,7] leverage pre-training and fine-tuning to improve performance. Chen et al. [1] build the GeoQA dataset and propose NGS, the first deep learning method for GPS, with two pre-training tasks for better cross-modal alignment. Chen et al. [7] present the unified benchmark UniGeo and introduce Geoformer, a multi-task transformer for geometric calculation and proving, enhanced by math expression pre-training. Zhang et al. [3] construct the fine-grained PGPS9K dataset and design PGPSNet with an MLM-based semantic pre-training strategy, which converts diagrams into text clauses for effective feature representation. Liang et al. [8] propose UniMath to handle diverse multimodal math problems, and augment the vocabulary with tokenized image representations from a trainable VQ-VAE [20,21]. Ning et al. [19] develop a symbol-character aware model, using self-supervised learning and masked image modeling to improve diagram understanding. These methods generate fixed solution programs, but expert annotation leads to high costs. Recently, large language models have advanced geometric problem solving. Zhao et al. [22] systematically survey GPS in the large

model era, covering benchmarks, multimodal parsing, and reasoning paradigms. Gao et al. [23] release the Geo170K dataset with over 170K geometric image-caption and question-answer pairs, and develop G-LLaVA for multimodal GPS. In addition, chain-of-thought prompting has become a key technique for multi-step logical reasoning. GeomVerse [24] validates that CoT fine-tuning significantly boosts reasoning ability on complex geometry problems, and GNS [25] further improves accuracy and interpretability by combining symbolic parsing with CoT. How to more effectively apply geometric knowledge for GPS remains an important open direction.

2.2 Chain-of-Thought Reasoning

Chain-of-Thought (CoT) reasoning has emerged as a crucial paradigm for enhancing complex reasoning capabilities in large language models. Wei et al. [11] pioneer CoT prompting, significantly improving model performance on arithmetic, commonsense, and symbolic reasoning tasks through intermediate reasoning steps. Kojima et al. [26] advance this with zero-shot CoT, showing that simply adding "Let's think step by step" can elicit reasoning capabilities. To enhance CoT robustness, researchers explore multiple directions. Wang et al. [27] propose Self-Consistency, which improves answer reliability through multi-path sampling and voting mechanisms. Zhang et al. [28] construct diverse exemplars via automatic clustering. Yao et al. [29] and Besta et al. [30] introduce Tree-of-Thoughts (ToT) and Graph-of-Thoughts (GoT), respectively, extending linear reasoning to tree and graph structures. In multimodal scenarios, Zhang et al. [18] propose Multimodal-CoT, employing a two-stage framework for sequential rationale generation and answer inference. Rose et al. [16] introduce Visual Chain-of-Thought, incorporating visual information to address logical gaps in text-based reasoning. However, existing work primarily focuses on optimizing single reasoning chains, with limited exploration of synergies between different chain types. We propose a hybrid chain-of-thought approach for more effective geometry problem solving through the complementary fusion of language and program reasoning chains.

3 Methodology

3.1 Preliminary

We observe that state-of-the-art MLLMs, such as GPT, Claude, and DeepSeek, although capable of understanding general visual tasks, have difficulty with geometric problems, even those simple for humans. As shown in Fig. 2, GPT-4o and DeepSeek yield erroneous results of 90° and 60°, respectively. Taking GPT-4o as an example, when supplemented with geometric theorems (Fig. 2, left), this model achieves improved reasoning but still struggles with visual interpretation; when given explicit mappings between theorems and visual elements (Fig. 2, right), this model successfully produces correct answers. This progressive improvement indicates that model performance hinges on the integration of geometric domain knowledge and visual cues. Therefore, developing effective geometric problem-solving frameworks requires deep integration of geometric knowledge with visual reasoning.

We propose KG-HoT, a knowledge distillation framework that leverages a teacher-student architecture to transfer geometric reasoning capabilities effectively. As depicted in Fig. 3, the framework operates in two distinct stages. (1) **Knowledge-grounded hybrid chain-of-thought generation**, where a multimodal large language model serves as the teacher model to extract problem-relevant theorems and properties, subsequently generating two complementary reasoning chains based on domain knowledge: a language-based chain-of-thought that describes reasoning steps in natural language, and a program-based chain-of-thought that encodes geometric relationships into python programs for computational solving. (2) **Hybrid chain-based knowledge distillation**, where the hybrid chains serve as supervision signals to train a lightweight student model through instruction tuning, enabling resource-efficient geometric reasoning.

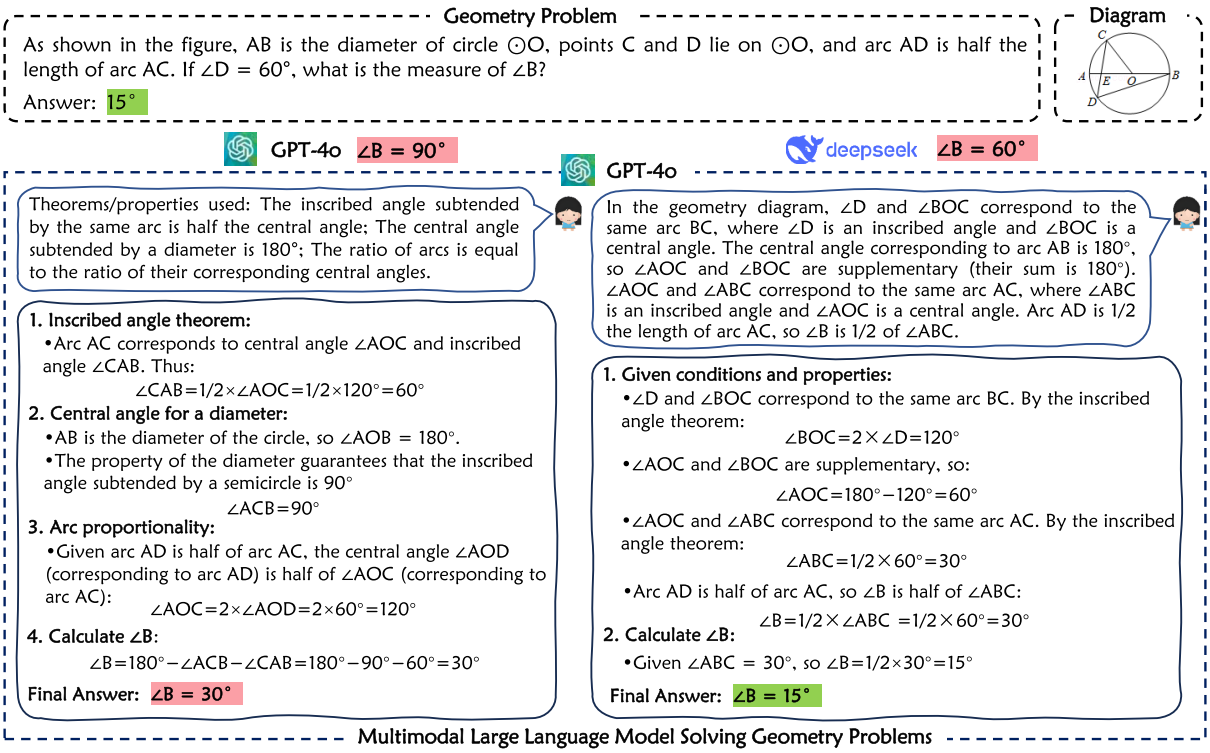

Figure 2: Geometric reasoning by multimodal large language models with progressive knowledge enhancement. Left: Models with geometric theorems show improved but imperfect reasoning. Right: Models with theorem-visual mappings achieve correct solutions.

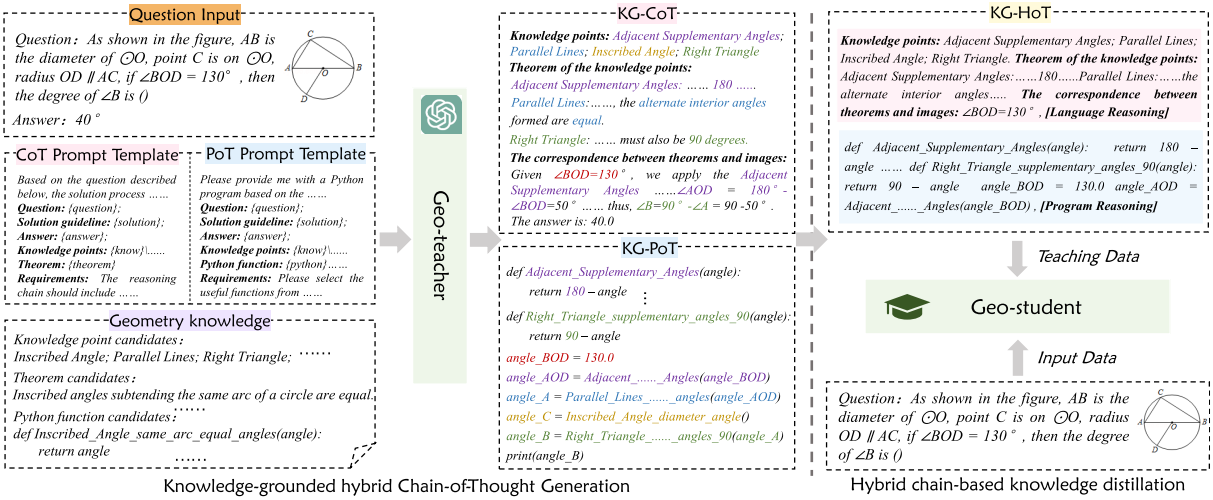

Figure 3: KG-HoT framework.

3.2 KG-HoT

Large language models have accumulated comprehensive knowledge bases and superior linguistic comprehension through extensive pre-training on large-scale corpora. Leveraging this foundation, our KG-HoT model employs geometric knowledge to guide multimodal large language models in generating

high-quality hybrid chains-of-thought through carefully designed prompt templates. Specifically, we first collect key geometric knowledge as follows:

- Knowledge concepts: Basic theoretical knowledge essential for solving specific problem types (e.g., *Parallel Lines*, *Inscribed Angles*, *Pythagorean theorem*). The knowledge concept candidate set is denoted as $K = \{k_1, k_2, ..., k_n\}$, where n is the number of geometric knowledge.
- Theorems: Rigorously proven mathematical propositions with general validity (e.g., *Alternate interior angles formed by parallel lines are equal*). All collected theorems are denoted as: $\mathcal{T} = \{T_{k_1}, T_{k_2}, ..., T_{k_n}\}$, where $T_{k_i} = \{t^1_{k_i}, t^2_{k_i}, ..., t^m_{k_i}\}$ represents the candidate set of theorems associated with knowledge concept k_i, with m being the number of such theorems.
- Python function candidates: For each knowledge concept and its corresponding theorems, we predefine a series of python functions whose names consist of the knowledge concept name and a theorem summary (e.g., *Parallel_Lines_Alternate_Interior_Angles*). This nomenclature establishes explicit mappings between functions and geometric knowledge. All predefined python functions are denoted as $\mathcal{P} = \{P_{k_1}, P_{k_2}, ..., P_{k_n}\}$, where $P_{k_i} = \{p^1_{k_i}, p^2_{k_i}, ..., p^m_{k_i}\}$ represents the python function candidate set for the knowledge concept k_i, with m being the number of theorems in k_i.

We collect 48 candidate geometric knowledge concepts, 137 candidate geometric theorems, and 137 candidate Python functions. All collected knowledge concepts, theorems, and corresponding Python functions are available in the data/collection folder at https://github.com/jmhz24/HG-HoT. Fig. 4 illustrates representative examples of the collected knowledge.

Knowledge concepts	Theorem	Python Function
Inscribed Angle	The inscribed angles subtended by the same arc are equal.	`def Inscribed_Angle_same_arc_equal_angles(angle):` `  return angle`
	The inscribed angle subtended by a diameter is 90°.	`def Inscribed_Angle_diameter_angle():` `  return 90`

	The inscribed angle subtended by the same arc is half the central angle.	`def Inscribed_Angle_relationship_with_central_angle(central_angle=None, inscribed_angle=None):`
		`  if central_angle:`
		`  return 1/2 * central_angle`
		`  else:`
		`  return 2 * inscribed_angle`

Parallel Lines	If two lines are parallel, the alternate interior angles formed are equal.	`def Parallel_Lines_alternate_interior_angles(angle):` `  return angle`
	If two lines are parallel, the corresponding angles formed are equal.	`def Parallel_Lines_corresponding_angles(angle):` `  return angle`
	If two lines are parallel, the consecutive interior angles formed are supplementary.	`def Parallel_Lines_consecutive_interior_angles_180(angle):` `  return 180 - angle`

Pythagorean	In a right-angled triangle, the square of the hypotenuse is equal to the sum of the squares of the other two legs.	`def Right_Triangle_pythagorean_theorem_add(leg_a=None, leg_b=None):`
		`  return math.sqrt(leg_a ** 2 + leg_b ** 2)`
		`def Right_Triangle_pythagorean_theorem_minus(leg_a=None, hypotenuse_c=None):`
		`  return math.sqrt(hypotenuse_c ** 2 - leg_a ** 2)`

Figure 4: Samples of geometric knowledge mapping between knowledge concepts, theorems, and python functions.

3.2.1 Knowledge-Grounded Language-Based CoT Generation

Given the extensive knowledge scope required for geometric problems, we adopt a two-step zero-shot prompting strategy to mitigate potential interference from irrelevant knowledge in multimodal large language models. This approach comprises: (1) knowledge concept identification and (2) high-quality chain-of-thought generation. Knowledge concepts refer to fundamental geometric principles (e.g., *Inscribed Angle*; *Parallel Lines*) that are essential for problem-solving. For knowledge concept identification, we design

a knowledge selection/generation prompt template as follows, into which the i-th training sample X^i is inserted:

> [Instruction]; Question: $[X_q^i]$; Diagram: $[X_v^i]$; Simple solution process: $[X_e^i]$; Answer: $[X_a^i]$; Geometric knowledge concept candidates: $[K]$; [Output format].

In this template, X_q^i, X_v^i, X_e^i and X_a^i denote the question, diagram, simple solution process, and answer from the i-th sample, respectively. The [Instruction] slot provides the following directive: *Given the question, geometric diagram, simple solution process, and answer, identify and select the relevant knowledge concepts from the candidates below to solve this question. If the provided candidates do not fully cover the required concepts, generate additional knowledge concepts based on your understanding*. The [output format] should include two parts: *(1) Selected concepts from candidates: list the relevant concepts chosen from the provided candidates; (2) Additional required concepts: list any new concepts not in the candidates but necessary for solving the problem, or state "N/A" if unable to generate additional concepts*. Upon processing this prompt, MLLMs generate a filtered knowledge concept set $X_K^i = \{k_1^i, k_2^i, ..., k_l^i\} \subseteq K$, where $K = \{k_1, k_2, ..., k_n\}$ and k_j^i denotes the j-th selected concept for sample X^i. Based on the selected knowledge concepts, the corresponding theorems $X_T^i = \bigcup_{j=1}^{l} T_{k_j^i}$ and Python functions $X_P^i = \bigcup_{j=1}^{l} P_{k_j^i}$ are collected, where $T_{k_j^i}$ and $P_{k_j^i}$ are drawn from the theorem collection $\mathcal{T} = \{T_{k_1}, T_{k_2}, ..., T_{k_n}\}$ and Python function collection $\mathcal{P} = \{P_{k_1}, P_{k_2}, ..., P_{k_n}\}$ defined in Section 3.2, with l denoting the number of knowledge concepts relevant to sample X^i.

Subsequently, we construct the following prompt template for language-based chain-of-thought generation based on the identified knowledge concepts set:

> [Basic Instruction]; Question: $[X_q^i]$; Diagram: $[X_v^i]$; Simple solution process: $[X_e^i]$; Answer: $[X_a^i]$; Geometric knowledge concepts: $[X_K^i]$; Theorem sets corresponding to each geometric knowledge concept: [For each $k_j^i \in X_K^i$: Theorem set for k_j^i is $[T_{k_j^i}]$]; [Detailed Instruction].

The prompt template design incorporates multiple key slots. Note that each theorem set $T_{k_j^i} = \{t_{k_j^i}^1, t_{k_j^i}^2, ..., t_{k_j^i}^m\}$ may contain multiple theorem statements. The [Simple solution process] and [Answer] slots provide contextual information to guide MLLMs in generating more reliable reasoning processes (filled with "N/A" when no guidance is available). The [Basic Instruction] slot contains the task description, prompting the model to generate reasoning chains based on the problem, image, knowledge concepts, and theorem candidates. The [Detailed Instruction] slot establishes three critical constraints. First, it requires MLLMs to follow a three-step reasoning process: inferring potential knowledge concepts, determining relevant theorems, and establishing knowledge-image mapping relationships. This coarse-to-fine thinking pattern helps student models learn structured reasoning and promotes multimodal information alignment. Second, it mandates the model to select necessary theorems from the candidate set when available, while leveraging its own understanding to generate appropriate reasoning for "N/A" fields, promoting reasoning reliability and robustness for out-of-scope problems. Third, it specifies the output format and requires concise expression to prevent redundant content caused by hallucinations. Fig. 5 shows an example of a language-based chain-of-thought prompt template.

Ultimately, this template guides MLLMs to produce a language-based chain $\mathcal{C}_L$ composed of: **knowledge concept chain → knowledge theorem chain → relationship chain between different knowledge and geometric images**.

Basic Instruction: [*Please generate a reasoning chain for solving this problem based on the following question, geometric image, simple solution guidance, answer, knowledge points, and their corresponding theorem candidates.*];
Question: [*As shown in the figure, AB is the diameter of ⊙O, point C is on ⊙O, radius OD ∥ AC, if ∠BOD = 130° , then the degree of ∠B is ()*];
Geometric Image:[] ;
Simple solution process:[*The key to solving this problem lies in understanding the theorems of parallel lines and inscribed angles*.];
Answer: [*40 °*];
Geometric knowledge concepts: [*Inscribed Angle; Parallel Lines; Supplementary Angles; Right Triangle*];
Theorem candidates corresponding to each geometric knowledge: [*Theorem candidates for geometric knowledge concept* ***Inscribed Angle*** *are* ***The inscribed angles subtended by the same arc are equal, The inscribed angle subtended by a diameter is 90°***, *and* ***The inscribed angle subtended by the same arc is half the central angle***; *Theorem candidates for geometric knowledge concept* ***Parallel Lines*** *are* ***If two lines are parallel, the alternate interior angles formed are equal***, ***If two lines are parallel***, ***the corresponding angles formed are equal***, *and* ***If two lines are parallel, the consecutive interior angles formed are supplementary***. *Theorem candidates for geometric knowledge concept* ***Supplementary Angles*** *are* ***The sum of the measures of two supplementary angles that share a common side is always equal to 180 degrees***; *Theorem candidates for geometric knowledge concept* ***Right Triangle*** *are* ***Pythagorean Theorem: In a right triangle, the square of the length of the hypotenuse is equal to the sum of the squares of the lengths of the other two sides***, ***Since one angle is 90 degrees, the sum of the other two angles must also be 90 degrees***, ***and In a right-angled triangle, the length of the segment from the midpoint of the hypotenuse to the right-angle vertex is half the length of the hypotenuse***.];
Detailed Instruction: [*Generate the reasoning chain following these three steps: 1. Infer the implicit knowledge points in the problem; 2. Identify the geometric theorems involved in the problem; 3. Locate the correspondences between different knowledge and geometric figures. Please select the necessary information from theorem candidates based on the knowledge points for solving this problem. If any field shows 'N/A', leverage your own understanding to generate the appropriate reasoning content based on the question. After generating the reasoning chain, output: The answer is: followed by the answer. The generated reasoning process should be as concise as possible.*]

Figure 5: Sample of language-based chain-of-thought prompt template.

3.2.2 Knowledge-Grounded Program-Based CoT Generation

The program-based chain-of-thought is designed to facilitate student models' learning of reasoning processes through program abstraction while establishing multimodal mappings between program variables and visual elements. Building upon the knowledge concepts from Section 3.2.1, our program-based chain-of-thought prompt template incorporates the following core components:

> [Basic Instruction]; Question: $[X_q^i]$; Diagram: $[X_v^i]$; Simple solution process: $[X_e^i]$; Answer: $[X_a^i]$; Geometric knowledge concepts: $[X_K^i]$; Python function sets corresponding to each geometric knowledge concept: [For each $k_j^i \in X_K^i$: Python function set for k_j^i is $[P_{k_j^i}]$]; [Detailed Instruction].

The template reuses the basic slots from the language-based version (problem, simple solution process, answer, knowledge concepts) but replaces theorem candidates with Python function candidates. Note that each Python function set $P_{k_j^i} = \{p_{k_j^i}^1, p_{k_j^i}^2, ..., p_{k_j^i}^m\}$ may contain multiple Python functions. The [Basic Instruction] requires the model to generate complete Python programs based on the input information. The [Detailed Instruction] establishes critical constraints: selecting and invoking relevant functions from the candidate set when available, while using the model's own knowledge to implement necessary logic for "N/A" fields; maintaining consistency between variable names and image elements; generating concise, executable code without comments or redundant output. These constraints ensure that the generated programs both embody reasoning logic and remain practically executable, even for problems with incomplete knowledge coverage. Fig. 6 shows an example of a program-based chain-of-thought prompt template.

This template guides MLLMs to produce an executable programmatic chain C_P **that encodes the complete reasoning process**.

Basic Instruction: [*Please provide me with a Python program based on the following question, geometric image, simple solution guidance, answer, knowledge points, and their corresponding python program candidates.*];
Question: [*As shown in the figure, AB is the diameter of ⊙O, point C is on ⊙O, radius OD ∥ AC, if ∠BOD = 130°, then the degree of ∠B is ()*];
Geometric Image:[] ;
Simple solution process:[*The key to solving this problem lies in understanding the theorems of parallel lines and inscribed angles.*];
Answer: [*40 °*];
Geometric knowledge concepts : [*Inscribed Angle ; Parallel Lines; Supplementary Angle; Right Triangle*];
Python program candidates corresponding to each geometric knowledge: [

The python program corresponding to the knowledge concept ***Inscribed Angle*** *includes:*

```
def Inscribed_Angle_same_arc_equal_angles(angle):
    return angle
def Inscribed_Angle_diameter_angle():
    return 90
def Inscribed_Angle_relationship_with_central_angle(central_angle=None, inscribed_angle=None):
    if central_angle:
        return 1/2 * central_angle
    else:
        return 2 * inscribed_angle
```

The python program corresponding to the knowledge concept ***Parallel Lines*** *includes:*

```
def Parallel_Lines_alternate_interior_angles(angle):
    return angle
def Parallel_Lines_corresponding_angles(angle):
    return angle
def Parallel_Lines_consecutive_interior_angles_180(angle):
    return 180 - angle
```

The python program corresponding to the knowledge concept ***Right Triangle*** *includes:*

```
import math
def Right_Triangle_pythagorean_theorem_add(leg_a=None, leg_b=None):
    return math.sqrt(leg_a ** 2 + leg_b ** 2)

def Right_Triangle_pythagorean_theorem_minus(leg_a=None, hypotenuse_c=None):
    return math.sqrt(hypotenuse_c ** 2 - leg_a ** 2)

def Right_Triangle_supplementary_angles_90(angle):
    return 90 - angle

def Right_Triangle_medians(hypotenuse=None, median=None):
    if hypotenuse:
        return (1 / 2) * hypotenuse
    else:
        return 2 * median
```

The python program corresponding to the knowledge concept ***Supplementary Angle*** *includes:*

```
def Supplementary_Angles(angle):
    return 180 - angle
```

];
Detailed Instruction: [Please select the useful functions from these candidate functions to solve this problem, and then create a complete program. Candidate functions for the knowledge point can be selected more than once. For any 'N/A' fields, use your knowledge to implement the necessary logic. Do not include comments or text in the program; just generate the program directly and briefly. When printing, do not add extra strings, just print the variable directly. Please avoid instances where the generated Python program calls certain functions, but the called functions are not included within the program. Please ensure that every function is used. Ensure that the program is executable.]

Figure 6: Sample of program-based chain-of-thought prompt template.

3.2.3 Hybrid Chain-of-Thought

To fully leverage the complementary advantages of different reasoning paradigms, we integrate language-based and program-based chains-of-thought into unified training samples $\mathcal{C} = \{\mathcal{C}_L, \mathcal{C}_P\}_{MIX}$. As illustrated in Fig. 7, the language-based chain simulates natural language reasoning to cultivate conceptual understanding capabilities, while the program-based chain implements symbolic computation to improve reasoning precision. By integrating these dual paradigms, student models develop both intuitive insight and computational accuracy, enabling comprehensive geometric problem-solving proficiency.

3.3 Knowledge Distillation

Inspired by Orca2 [31], we employ the Prompt Erasure technique, where student models receive only task descriptions and teacher-generated hybrid chains-of-thought, without exposure to the detailed prompts used to guide the teacher model. This design aims to enable student models not only to learn reasoning steps but also, more importantly, to autonomously select appropriate reasoning strategies. Therefore, during training, we provide only generic instructions such as: “Please analyze the given problem, incorporate the image information, and provide solution steps with the answer.”

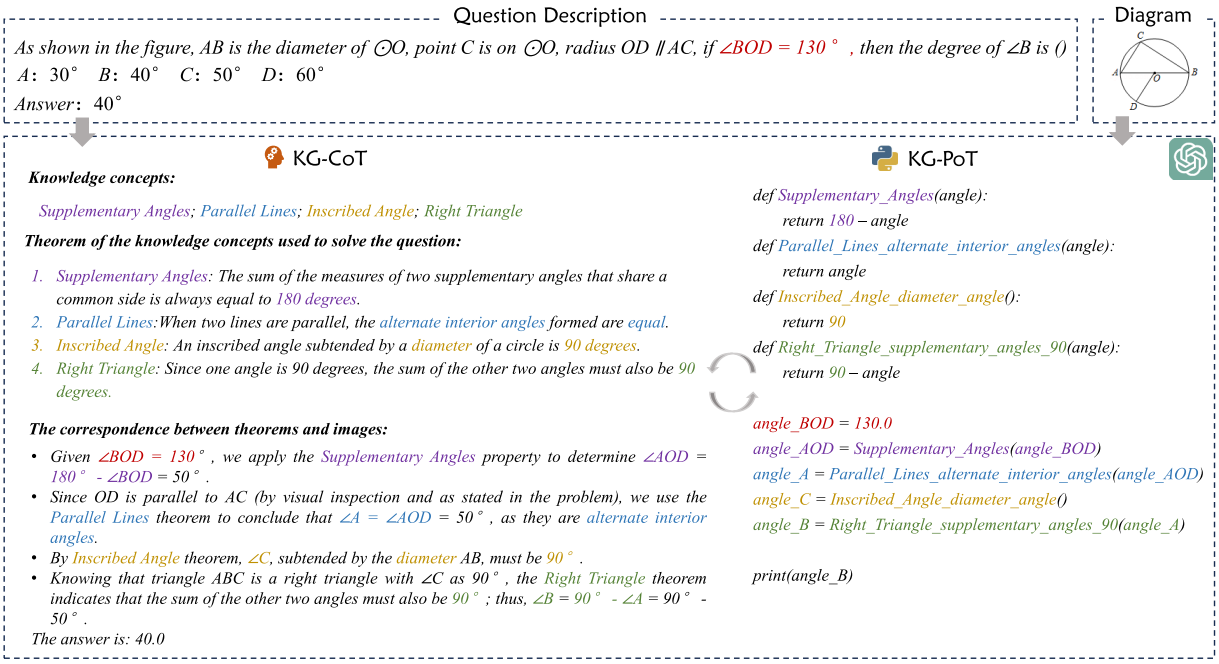

Figure 7: Sample of hybrid chain-of-thought combining KG-CoT (language-based reasoning) and KG-PoT (program-based reasoning).

The student model training follows this format: inputs comprise generic instructions $\mathcal{I}$, problem descriptions $\mathcal{Q}$, and geometric images $\mathcal{V}$; outputs consist of hybrid chains-of-thought $\mathcal{C}$. We employ special tokens [Language Reasoning] and [Program Reasoning] to distinguish between the two chain types, appending these tokens to the end of all inputs. Both modalities are jointly optimized during training, with the loss function defined as shown in Eq. (1):

$$\mathcal{L}(\boldsymbol{\Theta}) = -\sum_{i=1}^{|\mathcal{C}|} log\boldsymbol{p}(\boldsymbol{c_i}|\mathcal{X}_{input}, \boldsymbol{c_{<i}}; \boldsymbol{\Theta}), \tag{1}$$

where $\boldsymbol{\Theta}$ represents trainable parameters, $|\mathcal{C}|$ denotes the output data length, and $\mathcal{X}^{input}$ represents input data. For language-based and program-based chains, the inputs are $\mathcal{X}_{input\_L} = \{\mathcal{I}, \mathcal{Q}, \mathcal{V}, [Language\ Reasoning]\}$ and $\mathcal{X}_{input\_P} = \{\mathcal{I}, \mathcal{Q}, \mathcal{V}, [Program\ Reasoning]\}$, respectively, with corresponding outputs $\mathcal{C}_L$ and $\mathcal{C}_P$.

During inference, the model receives generic instructions, problem descriptions, and geometric diagrams. The visual encoder (CLIP ViT-L/14 in LLaVA-v1.5) extracts diagram features and maps them to the same embedding space dimension as the large language model through a two-layer MLP projection layer, which are then concatenated with text embeddings to form multimodal representations. Based on these representations, the model autoregressively generates output sequences, as shown in Eq. (2):

$$\mathcal{T} = \mathcal{F}(\mathcal{I}, \mathcal{Q}, \mathcal{V}; \boldsymbol{\Theta}), \tag{2}$$

where $\mathcal{F}$ denotes the student model, specifically a small-scale multimodal language model and $\mathcal{T}$ represents the generated reasoning process. By appending special tokens [Language Reasoning] and [Program Reasoning] to the corresponding input data, we prompt the student model to generate the respective types of chains-of-thought.

4 Experiments

We validate KG-HoT through comprehensive experiments on GeoQA and GeoQA+ datasets, conducting baseline comparisons, ablation, and further analysis.

4.1 Experiments Settings

4.1.1 Datasets

- GeoQA [1] contains 5010 geometry multiple-choice problems from Chinese mathematics examinations for grades 6–12, covering angle calculation, length calculation, and other types (e.g., area calculation). Each problem is annotated with simple solution guidance and manually crafted solution programs. The dataset is split into training/validation/test sets with a 7:1.5:1.5 ratio. We utilize the English version provided by [7].
- GeoQA+ [32] extends the GeoQA training set with an additional 2518 problems, totaling 7528, while retaining the original validation and test sets. Compared to GeoQA, this dataset introduces more challenging problems with expanded knowledge coverage and difficulty gradients, maintaining consistent annotation schemes.

Table 1 presents the data splits for both datasets.

Table 1: Statistics of the multimodal geometric problem-solving dataset.

Datasets	Numbers			
	All	Training	Validation	Test
GeoQA	5010	3509	745	754
GeoQA+	7528	6027	746	755

4.1.2 Implementation Details

Following [1,7,32], we employ accuracy as the evaluation metric and beam search (beam size = 10) for prediction generation. To ensure a fair comparison, answer options are not provided during either training or inference. GPT-4V (gpt-4-vision-preview, temperature = 0.7) serves as the teacher model for generating hybrid chains-of-thought, while the 7B-parameter LLaVA-v1.5 [33] functions as the student model, featuring an architecture comprising the Vicuna-7B-v1.5 language model, CLIP ViT-L/14 visual encoder, and a two-layer MLP projection layer. Both datasets and prompt templates are in English, with the input image resolution set to 224×224 pixels. We implement rigorous quality control during data generation. Linguistic chains are processed with answer extraction via regularization, incorrect answer filtering, and length constraints (50–2000 tokens), while program chains are verified for runtime correctness and result accuracy with length limits (50–1000 tokens). Failed samples are iteratively regenerated (up to 3 attempts), with manual annotation for persistent failures. For language-based chains, answers are extracted by identifying the "The answer is:" marker, while program-based chains obtain results through Python code execution. For evaluation, we employ a hybrid answer selection strategy that prioritizes program execution results for their precision, but falls back to language-based outputs when program results don't match any given answer choices (indicating potential errors). The answer choices serve only as validity checks during evaluation, not as model inputs.

To evaluate the comprehensiveness of our 48-concept knowledge base, we analyzed its coverage on the test sets. The knowledge base achieves 98.54% sample-level coverage (98.54% of test samples contain at

least one relevant concept) and 99.94% concept-level coverage (99.94% of all required concept instances are covered). For the small fraction of out-of-scope cases, our prompt templates explicitly instruct models to leverage their inherent knowledge, ensuring robust performance across all test problems.

We employ LoRA (Low-Rank Adaptation) [34] for efficient fine-tuning on all linear transformation layers (rank = 128, alpha = 256, dropout = 0.05) and the AdamW optimizer [35] (lr = 2e−4, β_1 = 0.9, β_2 = 0.999, eps = 1e−8) with cosine learning rate scheduling (3% warmup) and no weight decay. Training uses mixed precision (fp16) with gradient accumulation steps of 4 and gradient clipping at 1.0. Experiments are conducted on 4 NVIDIA A6000 GPUs (48 GB memory each) with an effective batch size of 28, achieving model convergence after 14 epochs. All experiments use fixed random seeds (42, 2024, 3047) for reproducibility, including data shuffling, model initialization, and dropout operations.

4.1.3 Baseline Models

We select the following 10 baseline models and provide human performance on the dataset:

- **FiLM** [36], which introduces Feature-wise Linear Modulation layers applying affine transformations to intermediate features based on input conditions for conditional visual reasoning.
- **RN** [37], which designs a plug-and-play Relation Networks module for relational reasoning across visual QA, textual QA, and complex reasoning tasks.
- **MCAN** [38], which enhances visual question answering performance through deep Modular Co-Attention Networks.
- **Seq2Prog+Diagram** [1], which builds on the Seq2Prog framework inspired by [39], using attention-based GRU for text encoding, ResNet for image features, and feature concatenation for multimodal fusion.
- **BERT2Prog + Diagram** [1], which replaces the encoder in Seq2Prog+Diagram with BERT, as derived from Chen et al. [1].
- **NGS** [1], which introduces Neural Geometry Solver fusing multimodal features via attention and employing two pre-training tasks (geometric jigsaw position and element prediction) for enhanced text-diagram representation.
- **Geoformer** [7], which unifies geometric calculation and proof as sequence generation with mathematical expression pre-training.
- **DPE-GPS** [32], which augments training data and employs Dual Parallel text Encoders for processing problem texts of varying lengths.
- **SCA-GPS** [19], which proposes Symbol Character-Aware modeling enhanced through self-supervised learning and masked image modeling for geometric diagram understanding.
- **LLaVA***, which directly applies LLaVA [33] to geometry datasets, trained on GeoQA/GeoQA+ for solution program generation with identical parameters to KG-HoT.

4.2 Experimental Results

4.2.1 Main Results

Tables 2 and 3 present performance comparisons across models on GeoQA and GeoQA+, respectively. Bold values in the tables indicate the best performance in each column. KG-HoT achieves optimal performance across all problem types, with improvements of 7.2% and 7.0% over the same architecture LLaVA* baseline, validating the effectiveness of hybrid chain-of-thought learning. The comparative results on GeoQA reveal distinct differences between two modeling paradigms among baseline models. FiLM, RN, and MCAN employ VQA classification approaches with inferior performance, indicating that classification

paradigms are unsuitable for complex geometric reasoning. In contrast, solution program generation models (BERT2Prog+Diagram through LLaVA*) demonstrate significant performance gains. Among these, early feature concatenation methods (Seq2Prog, BERT2Prog) show limited effectiveness; NGS and Geoformer utilize the T5 framework, achieving 4.0% and 1.5% improvements through pre-training, respectively; DPE-GPS and SCA-GPS design specialized modules for long problem sequences and symbolic problems, respectively, further enhancing performance. LLaVA*, leveraging powerful foundational capabilities, surpasses all specialized models without custom design, outperforming DPE-GPS and SCA-GPS by 4.1% and 2.7%, respectively. Compared to these models, our proposed KG-HoT achieves a 7.2% improvement. GeoQA+ augments the original dataset with 2518 challenging problems, forming a mix-training set. As shown in Table 3, KG-HoT achieves 75.1% accuracy on this dataset, leading all other methods. Compared to manual annotation of solution programs by experts, our method simply uses prompt templates to generate training data from large models, significantly reducing annotation costs. This approach enables models to learn both human-readable reasoning steps and formalized computational processes while maintaining performance, facilitating broader application deployment.

Table 2: Results of KG-HoT and comparative models on GeoQA dataset. Results for KG-HoT and LLaVA* are averaged over 3 runs with different random seeds. Other baseline results are taken from their respective papers.

Methods/Models		All	Angle	Length	Others
Human	Text-Only	63.0	58.1	71.7	55.6
	Text-Diagram	92.3	94.3	90.5	87.0
w/o solution program	FiLM [36]	31.7	34.0	29.7	24.1
	RN [37]	38.0	42.8	32.5	29.6
	MCAN [38]	39.7	45.0	34.6	25.9
Solution program	BERT2Prog + Diagram [1]	50.3	63.4	33.2	38.9
	Seq2Prog + Diagram [1]	52.6	63.6	39.2	37.0
	NGS [1]	56.7	67.5	44.5	37.0
	NGS-Auxiliary [1]	60.7	72.0	47.0	44.4
	Geoformer [7]	60.9	72.2	48.8	–
	Geoformer + Pre-training [7]	62.5	75.5	48.8	–
	DPE-GPS [32]	62.7	74.9	47.7	50.0
	SCA-GPS [19]	64.1	74.9	50.1	55.5
	LLaVA* [33]	66.8 ± 0.4	77.4 ± 0.3	55.2 ± 0.5	51.4 ± 0.5
Hybrid-thought	KG-HoT (Ours)	**74.0 ± 0.3**	**82.5 ± 0.3**	**65.0 ± 0.4**	**55.6 ± 0.5**

Table 3: Results of KG-HoT and comparative models on GeoQA+ dataset. Results for KG-HoT and LLaVA* are averaged over 3 runs with different random seeds. Other baseline results are taken from their respective papers.

Methods/Models		All	Angle	Length	Others
Mix-train	NGS [1]	61.2	72.3	47.7	46.3
	DPE-NGS [32]	66.0	75.6	54.4	51.9
	LLaVA* [33]	68.1 ± 0.4	77.7 ± 0.3	57.6 ± 0.5	48.2 ± 0.5
	KG-HoT (Ours)	**75.1 ± 0.3**	**82.3 ± 0.3**	**64.4 ± 0.4**	**53.2 ± 0.5**

4.2.2 Ablation Study

To validate the contributions of different chain-of-thought components, we conduct ablation study by removing particular component from it. Table 4 presents results when removing language-based and program-based chains-of-thought, respectively. When using only program-based chains (removing language-based reasoning), accuracy drops by 6.0% and 5.8% on GeoQA and GeoQA+, respectively, indicating the importance of natural language reasoning for semantic understanding. Similarly, when using only language-based chains (removing program-based reasoning), we observe more significant decreases, highlighting the necessity of structured computation for precise numerical calculations. These consistent results across both datasets confirm that our proposed hybrid chain-of-thought training strategy requires both reasoning paradigms as essential components, with each contributing unique capabilities that jointly enhance the model's geometric reasoning performance.

Table 4: Ablation study of KG-HoT on GeoQA and GeoQA+ test sets.

Methods/Datasets	GeoQA				GeoQA+			
	All	Angle	Length	Others	All	Angle	Length	Others
KG-HoT	74.0	82.5	65.0	55.6	75.1	82.3	64.4	53.2
w/o KG-CoT	68.0	76.3	56.2	53.7	69.3	79.5	58.4	50.1
w/o KG-PoT	55.2	64.0	46.1	55.0	56.1	64.9	48.3	47.0

4.3 Further Discussions

4.3.1 Analysis of Hybrid Chain-of-Thought Effectiveness

To verify the synergistic effects of hybrid chains of thought, we design two comparative experiments: 1) KG-HoT with hybrid training (decoding programs only) vs. KG-PoT with program-only training; 2) KG-HoT with hybrid training (decoding language only) vs. KG-CoT with language-only training. As shown in Table 5, hybrid chain-of-thought training outperforms single-type chain training even under single-output paradigm: for program generation, KG-HoT surpasses KG-PoT by an average of 3.4% (with angle problems showing the highest improvement at 4.6%); for language generation, KG-HoT exceeds KG-CoT by an average of 2.7%. This bidirectional improvement demonstrates synergistic gains between the two chains of thought during training, where learning from both paradigms simultaneously enhances each individual reasoning pathway.

Table 5: Comparison of hybrid chain-of-thought and single-type chain-of-thought. Hybrid training evaluated under both program-only and language-only decoding paradigms vs. corresponding single-type training baselines.

Methods	Program-Based Problem Solving				Language-Based Problem Solving			
	All	Angle	Length	Others	All	Angle	Length	Others
KG-PoT	68.0	76.3	56.2	53.7	–	–	–	–
KG-CoT	–	–	–	–	55.2	64.0	46.1	55.0
KG-HoT	71.4	80.9	59.0	55.6	57.9	66.2	47.7	52.3

4.3.2 Analysis of Multi-Chain Joint Learning Effectiveness

To validate the effectiveness of end-to-end hybrid chain-of-thought training, we compare it with two-stage training approaches where auxiliary information is first generated then used for final reasoning. As shown in Tables 6 and 7 the results indicate that end-to-end KG-HoT significantly outperforms two-stage methods across both output paradigms. For program generation, KG-HoT exceeds "Q-K; QK-PoT" (first generating knowledge concepts). "Q-KT; QKT-PoT" (first generating knowledge concepts and theorems), and "Q-CoT; QCoT-PoT" (first generating language-based chain-of-thought) by 2.5%, 3.7%, and 5.3%, respectively. For language-based generation, KG-HoT similarly surpasses "Q-PoT; QPoT-CoT". The performance degradation in two-stage methods primarily stems from error propagation, where generation errors in the first stage accumulate and impair second-stage reasoning. In contrast, our end-to-end hybrid training avoids such cascading errors through mutual guidance between different chain-of-thought paradigms, effectively enhancing geometric reasoning capabilities. To illustrate these pipeline variants, we include examples of each permutation's outputs in the examples folder of https://github.com/jmhz24/HG-HoT.

Table 6: Comparison of end-to-end vs. pipeline training methods for program-based chain-of-thought. Pipeline methods use two-stage training where auxiliary information (Q: Question, K: Knowledge concepts, T: Theorems, CoT: Language-based chain-of-thought) is generated first then used for PoT generation.

	Methods	All	Angle	Length	Others
End-to-End	**KG-HoT (Program-Only)**	**71.4**	**80.9**	**59.0**	**55.6**
Pipeline	1: Q-K; 2: QK-PoT	68.9	78.3	55.5	54.7
	1: Q-KT; 2: QKT-PoT	67.7	76.8	54.7	54.4
	1: Q-CoT; 2: QCoT-PoT	66.1	75.2	53.9	53.3

Table 7: Comparison of end-to-end vs. pipeline training methods for language-based chain-of-thought. Pipeline method generates program-based chain-of-thought (PoT) first then uses them for CoT generation.

	Methods	All	Angle	Length	Others
End-to-End	KD-MoT (Language-only)	55.2	64.0	46.1	55.0
Pipeline	1: Q-PoT; 2: QPoT-CoT	54.9	63.1	45.6	54.3

4.3.3 Generalization to Additional Benchmarks

To evaluate generalizability beyond GeoQA/GeoQA+, we conduct zero-shot tests on two external benchmarks: Geometry3K (601 questions) [40] and MathVista-Geometry (208 geometry problems from MathVista testmini) [41]. As shown in Table 8, our model achieves accuracies of 51.2% on Geometry3K and 62.0% on MathVista-Geometry. We further dissect source-wise performance within MathVista-Geometry in Fig. 8. The subset includes 62 problems each from GeoQA+, UniGeo and Geometry3K, and 22 from GEOS. Model performance declines consistently across sources, dropping from 72.6% on GeoQA+ to 45.5% on GEOS, due to inherent discrepancies in problem design and presentation. Two main reasons account for this performance disparity. First, Geometry3K contains minimal-text questions (e.g., "Find x") with geometric information embedded solely in diagrams, whereas our training data from GeoQA/GeoQA+ features rich textual descriptions that effectively trigger knowledge retrieval. Second, GEOS features comparative questions (e.g., "Which is greatest?") demanding multi-choice reasoning. As we exclude answer options during training to avoid implicit guidance, our model is not optimized for such comparative reasoning

formats, resulting in degraded performance. These results verify the superiority of our method on text-rich geometric problems. Meanwhile, they highlight the necessity of breaking the reliance on text-only retrieval triggers. Future work will further enhance the model capability to cope with visually implicit problems and comparative reasoning tasks, so as to strengthen generalization across diverse geometric benchmarks.

Table 8: Zero-shot generalization performance on additional geometry benchmarks.

Method Dataset	Geometry3K	MathVista-Geometry
KG-HoT	51.2	62.0

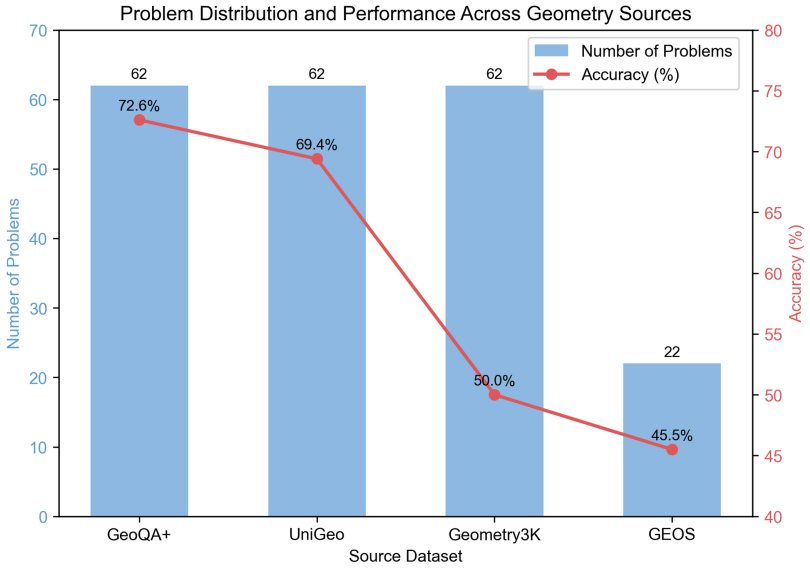

Figure 8: Question distribution and generalization performance across four geometry datasets in the MathVista-Geometry subset.

5 Limitations

While our hybrid chain-of-thought framework combines linguistic and programmatic reasoning with elaborate data generation and quality control, several limitations remain to be discussed. First, the current framework prioritizes reliable training data construction but lacks explicit self-correction during inference. Although our knowledge-guided generation and multi-stage verification (answer checking, length filtering, iterative validation and manual annotation) effectively mitigate hallucinations in data creation, the model cannot perform real-time error detection for intermediate reasoning steps at inference time. Second, all experiments are conducted on high-quality public datasets with well-organized problem descriptions and standard geometric diagrams. Our framework is therefore not equipped to handle ill-defined, ambiguous or incomplete real-world cases with vague statements, missing diagrams or imprecise visual inputs. Third, the model adopts a simple heuristic to select outputs between language and programmatic reasoning based on answer-option validation, without dynamically evaluating the reliability of each reasoning chain during inference. These limitations point to promising future directions for improving the framework's adaptability, reasoning reliability and practical applicability in real-world scenarios.

6 Conclusion and Future Work

In this work, we propose a knowledge-guided hybrid chain-of-thought (KG-HoT) framework to tackle high annotation costs and poor interpretability in multimodal geometry problem solving. Our method leverages knowledge-grounded prompts to generate two complementary reasoning chains. The language-based chain conducts progressive three-level reasoning consisting of knowledge recall, theorem application, and solution derivation for interpretability, while the program-based chain enables precise mathematical computation. The two chains are end-to-end distilled into lightweight models for mutual reinforcement and synergistic learning. Experimental results validate that KG-HoT achieves competitive performance across benchmarks and greatly reduces annotation costs. Future work will incorporate self-correction and real-time error detection for intermediate reasoning steps, adapt the framework to ill-defined and ambiguous geometric problems, develop advanced selection strategies for linguistic and programmatic reasoning, and explore knowledge representation to support complex geometric reasoning and cross-domain generalization.

Acknowledgement: The authors sincerely thank their co-authors for their continuous technical support, guidance, and assistance with the experimental equipment used in this work.

Funding Statement: This work is supported by the Youth Science and Technology of Gansu Province (26JRRA493, 24JRRA148), the Northwest Normal University Young Teachers Research Capacity Promotion Plan (NWNU-LKQN2027-19, NWNU-LKQN2024-22).

Author Contributions: Meihuizi Jia: Conceptualization, Methodology, Software, Writing—original draft, Funding acquisition. Hongyan Ran: Supervision, Writing—review & editing, Funding acquisition. Shanshan Li: Supervision. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The code and supplementary resources, including geometric knowledge concepts, theorems, and corresponding Python functions, are publicly available at https://github.com/jmhz24/HG-HoT.

Ethics Approval: This study involved no human participants or animal experiments, and thus ethical review and approval were not required.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen J, Tang J, Qin J, Liang X, Liu L, Xing E, et al. GeoQA: a geometric question answering benchmark towards multimodal numerical reasoning. In: Findings of the Association for Computational Linguistics (ACL-IJCNLP). Kerrville, TX, USA: Association for Computational Linguistics; 2021. p. 513–23.
2. Seo M, Hajishirzi H, Farhadi A, Etzioni O, Malcolm C. Solving geometry problems: combining text and diagram interpretation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2015 Sep 17–21; Lisbon, Portugal. p. 1466–76.
3. Zhang ML, Yin F, Liu CL. A multi-modal neural geometric solver with textual clauses parsed from diagram. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI); 2023 Aug 19–25; Macao, China. p. 3374–82.
4. Sachan M, Dubey K, Xing E. From textbooks to knowledge: a case study in harvesting axiomatic knowledge from textbooks to solve geometry problems. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2017 Sep 9–11; Copenhagen, Denmark. p. 773–84.
5. Zhang X, Zhu N, He Y, Zou J, Huang Q, Jin X, et al. Formalgeo: the first step toward human-like IMO-level geometric automated reasoning. arXiv:2310.18021. 2023.
6. Trinh TH, Wu Y, Le QV, He H, Luong T. Solving olympiad geometry without human demonstrations. Nature. 2024;625:476–82.

7. Chen J, Li T, Qin J, Lu P, Lin L, Chen C, et al. Unifying geometry logical reasoning via reformulating mathematical expression. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2022 Dec 7–11; Abu Dhabi, United Arab Emirates. p. 3313–23.
8. Liang Z, Yang T, Zhang J, Zhang X. Unimath: a foundational and multimodal mathematical reasoner. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2023 Dec 6–10; Singapore. p. 7126–33.
9. Yue X, Qu X, Zhang G, Fu Y, Huang W, Sun H, et al. MAmmoTH: building math generalist models through hybrid instruction tuning. In: The Twelfth International Conference on Learning Representations (ICLR); 2024 May 7–11; Vienna, Austria.
10. Comanici G, Bieber E, Schaekermann M, Pasupat I, Sachdeva N, Dhillon I, et al. Gemini 2.5: pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv:2507.06261. 2025.
11. Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, et al. Chain-of-thought prompting elicits reasoning in large language models. Adv Neural Inf Process Syst. 2022;35:24824–37. doi:10.52202/068431-1800.
12. Chen W, Ma X, Wang X, Cohen WW. Program of thoughts prompting: disentangling computation from reasoning for numerical reasoning tasks. Trans Mach Learn Res. 2023 [cited 2026 Apr 1]. Available from: https://openreview.net/forum?id=YfZ4ZPt8zd.
13. Yu J, He R, Ying Z. Thought propagation: an analogical approach to complex reasoning with large language models. In: The Twelfth International Conference on Learning Representations (ICLR); 2024 May 7–11; Vienna, Austria.
14. Wang PY, Liu TS, Wang C, Li Z, Wang Y, Yan S, et al. A survey on large language models for mathematical reasoning. ACM Comput Surv. 2026;58(8):209. doi:10.1145/3786333.
15. Yao Y, Li Z, Zhao H. Beyond chain-of-thought, effective graph-of-thought reasoning in large language models. arXiv:2305.16582. 2023.
16. Rose D, Himakunthala V, Ouyang A, He R, Mei A, Lu Y, et al. Visual chain of thought: bridging logical gaps with multimodal infillings. arXiv:2305.02317. 2023.
17. Wang L, Hu Y, He J, Xu X, Liu N, Liu H, et al. T-SCIQ: teaching multimodal chain-of-thought reasoning via large language model signals for science question answering. In: Thirty-Eighth AAAI Conference on Artificial Intelligence; 2024 Feb 20–27; Vancouver, BC, Canada. p. 19162–70.
18. Zhang Z, Zhang A, Li M, Zhao H, Karypis G, Smola A. Multimodal chain-of-thought reasoning in language models. Trans Mach Learn Res. 2024. doi:10.59350/73qcj-wyt28.
19. Ning M, Wang QF, Huang K, Huang X. A symbolic characters aware model for solving geometry problems. In: Proceedings of the 31st ACM International Conference on Multimedia; 2023 Oct 29–Nov 3; Ottawa, ON, Canada. p. 7767–75.
20. van den OA, Vinyals O, Kavukcuoglu K. Neural discrete representation learning. In: 31st Conference on Neural Information Processing Systems (NeurIPS); 2017 Dec 4–9; Long Beach, CA, USA. p. 6309–18.
21. Razavi A, van den OA, Vinyals O. Generating diverse high-fidelity images with VQ-VAE-2. In: 33rd Conference on Neural Information Processing Systems (NeurIPS); 2019 Dec 8–14; Vancouver, BC, Canada. p. 14866–76.
22. Zhao Y, Wang X, Liu J, King I, Huang Z. Towards geometry problem solving in the large model era: a survey. arXiv:2506.02690. 2025.
23. Gao J, Pi R, Zhang J, Ye J, Zhong W, Wang Y, et al. G-LLaVA: solving geometric problem with multi-modal large language model. arXiv:2312.11370. 2023.
24. Kazemi M, Alvari H, Anand A, Wu J, Chen X, Soricut R. GeomVerse: a systematic evaluation of large models for geometric reasoning. arXiv:2312.12241. 2023.
25. Ning M, Zhou Z, Wang Q, Huang X, Huang K. GNS: solving plane geometry problems by neural-symbolic reasoning with multi-modal LLMS. In: Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence; 2025 Feb 25–Mar 4; Philadelphia, PA, USA. p. 24957–65.
26. Kojima T, Gu SS, Reid M, Matsuo Y, Iwasawa Y. Large language models are zero-shot reasoners. Adv Neural Inf Process Syst. 2022;35:22199–213. doi:10.52202/068431-1613.

27. Wang X, Wei J, Schuurmans D, Le QV, Chi EH, Narang S, et al. Self-consistency improves chain of thought reasoning in language models. In: The Eleventh International Conference on Learning Representations (ICLR); 2023 May 1–5; Kigali, Rwanda.
28. Zhang Z, Zhang A, Li M, Smola A. Automatic chain of thought prompting in large language models. In: The Eleventh International Conference on Learning Representations (ICLR); 2023 May 1–5; Kigali, Rwanda.
29. Yao S, Yu D, Zhao J, Shafran I, Griffiths TL, Cao Y, et al. Tree of thoughts: deliberate problem solving with large language models. In: 37th Conference on Neural Information Processing Systems 2023 (NeurIPS); 2023 Dec 10–16; New Orleans, LA, USA. p. 11809–22.
30. Besta M, Blach N, Kubicek A, Gerstenberger R, Podstawski M, Gianinazzi L, et al. Graph of thoughts: solving elaborate problems with large language models. In: Thirty-Eighth AAAI Conference on Artificial Intelligence; 2024 Feb 20–27; Vancouver, BC, Canada. p. 17682–90.
31. Magister LC, Mallinson J, Adamek J, Malmi E, Severyn A. Teaching small language models to reason. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers); 2023 Jul 9–14; Toronto, ON, Canada. p. 1773–81.
32. Cao J, Xiao J. An augmented benchmark dataset for geometric question answering through dual parallel text encoding. In: Proceedings of the 29th International Conference on Computational Linguistics (COLING); 2022 Oct 12–17; Gyeongju, Republic of Korea. p. 1511–20.
33. Liu H, Li C, Wu Q, Lee YJ. Visual instruction tuning. In: 37th Conference on Neural Information Processing Systems (NeurIPS); 2023 Dec 10–16; New Orleans, LA, USA.
34. Yu Y, Yang CHH, Kolehmainen J, Shivakumar PG, Gu Y, Ren SRR, et al. Low-rank adaptation of large language model rescoring for parameter-efficient speech recognition. In: 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU); 2023 Dec 16–20; Taipei, Taiwan. p. 1–8.
35. Kingma DP, Ba J. Adam: a method for stochastic optimization. In: International Conference on Learning Representations (ICLR); 2015 May 7–9; San Diego, CA, USA.
36. Perez E, Strub F, De Vries H, Dumoulin V, Courville A. Film: visual reasoning with a general conditioning layer. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2018 Feb 2–7; New Orleans, LA, USA. p. 3942–51.
37. Santoro A, Raposo D, Barrett DG, Malinowski M, Pascanu R, Battaglia P, et al. A simple neural network module for relational reasoning. In: 31st Conference on Neural Information Processing Systems (NeurIPS); 2017 Dec 4–9; Long Beach, CA, USA. p. 4967–76.
38. Yu Z, Yu J, Cui Y, Tao D, Tian Q. Deep modular co-attention networks for visual question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019 Jun 15–20; Long Beach, CA, USA. p. 6281–90.
39. Amini A, Gabriel S, Lin S, Koncel-Kedziorski R, Choi Y, Hajishirzi H. Mathqa: towards interpretable math word problem solving with operation-based formalisms. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT); 2019 Jun 2–7; Minneapolis, MN, USA. p. 2357–67.
40. Lu P, Gong R, Jiang S, Qiu L, Huang S, Liang X, et al. 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 (ACL/IJCNLP); 2021 Aug 1–6; Virtual. p. 6774–86.
41. Lu P, Bansal H, Xia T, Liu J, Li C, Hajishirzi H, et al. MathVista: evaluating mathematical reasoning of foundation models in visual contexts. In: The Twelfth International Conference on Learning Representations (ICLR); 2024 May 7–11; Vienna, Austria.