Compositional Reasoning via Joint Image and Language Decomposition

Madhav Kanda[1]*, **Dwip Dalal**[1]*, **Zhenhailong Wang**[1], **Heng Ji**[1], **Unnat Jain**[2]
[1]University of Illinois Urbana-Champaign [2]University of California, Irvine
`{madhav3, dwip2}@illinois.edu`

Abstract

Multimodal reasoning tasks such as visual question answering (VQA) require models to process both language and visual inputs. However, existing approaches typically decompose only language queries, treating images as monolithic inputs. We introduce REDI, a framework that jointly decomposes both images and questions into visual sub-domains (segmentation, material, depth, and color) with corresponding sub-questions. REDI uses an MLLM orchestrator to select the sub-domains required for each query, generate domain-specific sub-questions with grounded object references (via shared object labels), and fuse worker outputs via consistency-aware aggregation (verify–refine–override) to produce the final answer. This hierarchical multi-agent design mitigates error propagation and improves compositional reasoning across both open- and closed-source MLLMs. On SEEDBench, MMBench, and CLEVR, REDI achieves absolute accuracy improvements of 8.9%, 8.2%, and 16.0% over base model. Project webpage: `https://madhav-kanda.github.io/redi/`

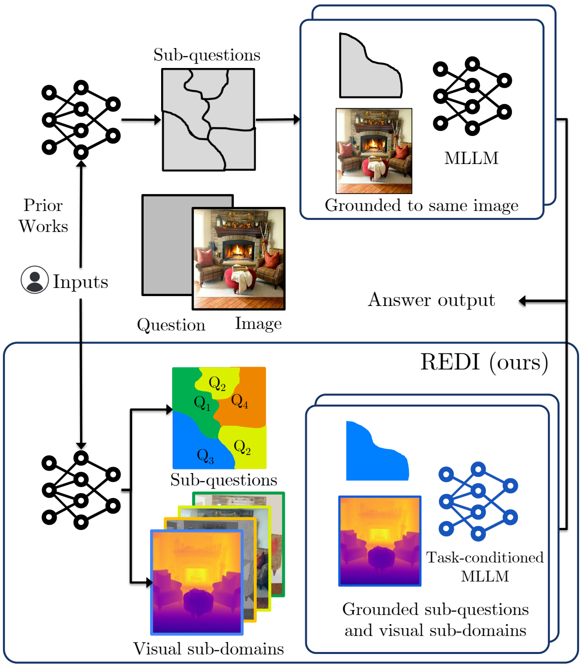


Figure 1: **Reasoning with Decomposed Inputs.** Our REDI framework jointly decomposes image and language inputs into sub-tasks aligned with distinct visual sub-domains (segmentation, material, depth, color, and blind cues), forming a hierarchical system that extends beyond the limited decompositions of prior work.

1 Introduction

Consider the task of driving to a friend's home for dinner. Throughout this task, visual input guides action: you monitor the *color* of traffic lights to proceed, *segment* pedestrians while turning, judge *depth* to maintain safe distance from the car ahead, and adjust your path upon noticing slippery *material* such as ice. Even *without seeing* the road or your surroundings, you already can reason that you should leave early to pick flowers for your friend. This everyday scenario demonstrates how humans naturally decompose multimodal inputs to solve complex tasks. Current multimodal reasoning methods, however, rarely decompose both language and visual inputs jointly. While modern models increasingly break down language queries into sub-parts, they typically treat images as monolithic inputs or apply only coarse segmentation into bounding boxes.

Multimodal Large Language Models (MLLMs) such as LLaVA (Liu et al., 2024a), InstructBLIP (Dai et al., 2023), and GPT-4V (Achiam et al., 2023) have advanced multimodal reasoning capabilities, while new benchmarks (Johnson et al., 2016; Li et al., 2023; Liu et al., 2024b) have emerged to assess fine-grained reasoning abilities. However, MLLMs often struggle on these benchmarks (Verma et al., 2024; Shiri et al., 2024; Thrush et al., 2022; Dalal et al., 2025a), motivating stronger reasoning frameworks such as chain-of-

---

[*] Equal contribution.

Findings of the Association for Computational Linguistics: EACL 2026, pages 5753–5775
March 24-29, 2026 ©2026 Association for Computational Linguistics

thought (Kojima et al., 2022; Mitra et al., 2024), and visual programming (Gupta and Kembhavi, 2022; Surís et al., 2023). Yet even in these systems, decomposition is primarily limited to language (typically the question), while image inputs are only segmented into semantic bounding boxes. This creates two core bottlenecks for compositional reasoning: (i) extracting *fine-grained visual semantics*, grounded object instances with reliable attributes (e.g., color, material) and relations (e.g., front/back via depth); and (ii) *planning and routing* multi-step solutions, i.e., deciding which perceptual tool to invoke at each step and how to compose the resulting intermediate outputs into a globally consistent answer.

Building on foundational work in visual perception (Gibson, 2002) and visual processing (J. Paul Getty Museum, 2011), which identify shapes, texture, space, and color as fundamental visual attributes, we propose joint decomposition of both image and language inputs. We decompose visual inputs into sub-domains of {segmentation, material, depth, color} and generate corresponding sub-questions (see Fig. 1). Each visual sub-domain is paired with its sub-question and processed by a specialized worker agent using a task-conditioned MLLM. We also incorporate a Blind Agent to address queries that rely on language-specific information without requiring visual input. The orchestrator first selects the minimal set of relevant sub-domains and generates targeted (template-based) sub-questions for the corresponding workers, then performs consistency-aware aggregation, which is cross-agent validity checks and conflict resolution grounded in the original image and question, to produce the final answer. We refer to our framework as Reasoning with Decomposed Inputs, or REDI for short.

Our contributions are as follows: (1) a compositional reasoning approach that jointly decomposes image and language inputs into visual sub-domains; (2) a hierarchical multi-agent system with specialized workers, grounded inter-agent communication, and consistency-aware aggregation (verify–refine–override); and (3) state-of-the-art performance on `SeedBench`, `MMBench`, and `CLEVR`, achieving absolute accuracy gains of 8.9%, 8.2%, and 16.0%, respectively, over the base model. We further validate the generalization of our method across three diverse MLLMs (LLaVA, InstructBLIP, and GPT-4V), highlighting its broad applicability in compositional multimodal reasoning.

2 Related Work

Our work relates to visual programming, multi-agent systems, and compositional reasoning. We distinguish our approach by jointly decomposing both images and language into modality-specific sub-tasks, coordinated through specialized agents.

Visual Programming. Visual programming approaches answer questions about images through modular frameworks that compose pre-trained vision modules. Some methods use code generation to dynamically orchestrate these modules (Surís et al., 2023; Subramanian et al., 2023; Shen et al., 2024; Gupta and Kembhavi, 2022). Others incorporate object-centric information through bounding boxes or regions of interest (Lin et al., 2022; Chen et al., 2025), while graph-based methods leverage scene or concept graphs (Wang et al., 2023; Mitra et al., 2024). However, these setups inherently limit reasoning depth, as merely calling individual modules is insufficient for tasks requiring holistic understanding, for instance, questions about global scene. To address this limitation, we introduces an MLLM-based orchestrator agent capable of context-specific reasoning. This orchestrator strategically decomposes questions into tailored subtasks and effectively distributes them among specialized worker agents. In contrast to methods such as ViperGPT (Surís et al., 2023), which allocate tasks without domain-specific decomposition, our method explicitly analyzes visual sub-domains (e.g., depth, material), enabling more robust and comprehensive compositional reasoning.

Multi-Agent Systems. Multi-agent systems for vision and language reasoning follow two main approaches. The first uses collaborative reasoning, where agents evaluate and refine answers through voting and cross-verification (Jiang et al., 2024; Wang et al., 2024; Chen et al., 2023; Hu et al., 2025). The second employs dynamic agent generation, creating specialized agents on demand based on task requirements (Chen et al., 2024a; Ke et al., 2024). VIPAct (Zhang et al., 2024b) also uses a multi-agent framework to enhance VLM performance, but focuses primarily on perceptual tasks such as line intersection detection and object boundary identification. In contrast, our framework jointly decomposes both question and image inputs to address compositional reasoning tasks. Our evaluation spans IQ testing and commonsense reasoning (Liu et al., 2024b), which extend beyond perceptual analysis.

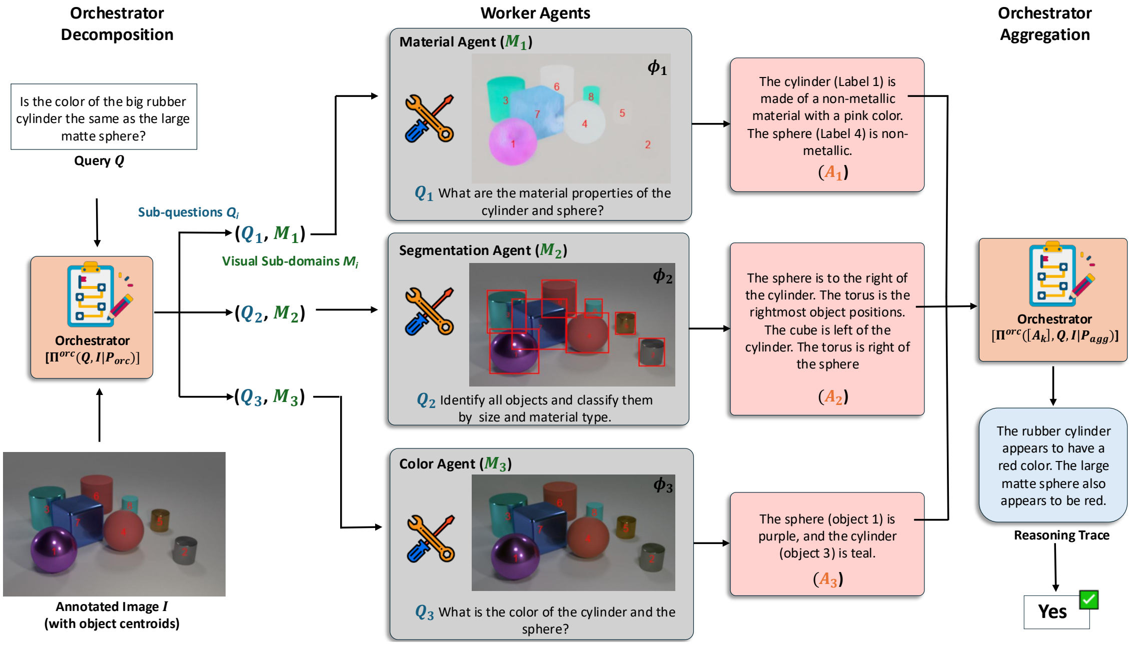

Orchestrator Decomposition
Is the color of the big rubber cylinder the same as the large matte sphere?
Query Q
Orchestrator
$[\Pi^{orc}(Q, I|P_{orc})]$
Annotated Image I (with object centroids)
Sub-questions Q_i
(Q_1, M_1)
Visual Sub-domains M_i
(Q_2, M_2)
(Q_3, M_3)
Worker Agents
Material Agent (M_1)
ϕ_1
Q_1 What are the material properties of the cylinder and sphere?
Segmentation Agent (M_2)
ϕ_2
Q_2 Identify all objects and classify them by size and material type.
Color Agent (M_3)
ϕ_3
Q_3 What is the color of the cylinder and the sphere?
The cylinder (Label 1) is made of a non-metallic material with a pink color. The sphere (Label 4) is non-metallic.
(A_1)
The sphere is to the right of the cylinder. The torus is the rightmost object positions. The cube is left of the cylinder. The torus is right of the sphere
(A_2)
The sphere (object 1) is purple, and the cylinder (object 3) is teal.
(A_3)
Orchestrator Aggregation
Orchestrator
$[\Pi^{orc}([A_k], Q, I|P_{agg})]$
The rubber cylinder appears to have a red color. The large matte sphere also appears to be red.
Reasoning Trace
Yes ✓

Figure 2: **Illustration of the REDI framework applied to a query.** Given an image and a natural-language question, the **REDI Orchestrator** identifies the relevant **objects of interest** and uses **Text-based SAM** to annotate them. It then determines which **specialized worker agents** (e.g., Color, Segmentation, Material) should be invoked and formulates **sub-questions** tailored to each agent's visual sub-domain. Each agent processes the image through its **domain-specific encoder** and then produces an answer and reasoning trace. The orchestrator subsequently integrates the agents' responses to generate a coherent and grounded final answer.

Compositional Reasoning. Compositional reasoning in vision-language tasks decomposes complex queries into sub-tasks and integrates their results. Existing methods vary significantly in their decomposition strategies, encompassing approaches based on scene graph representations (Mitra et al., 2024), attention-guided self-correction (Dalal et al., 2025b), compositional reasoning through multi-agent collaboration (Chen et al., 2024b; Zheng et al., 2024), and explicit question decomposition techniques (Zhou et al., 2024; Chakraborty et al., 2023; Rani et al., 2023, 2025; Zhang et al., 2024a). Mitra et al. (2024) introduced a zero-shot prompting strategy using scene graph representations for compositional reasoning. While scene graphs effectively capture visual relationships, they underperform on tasks requiring extensive linguistic inference. Our approach differs by performing modality-specific decomposition of both textual and visual components before aligning their sub-tasks. While existing methods (Chen et al., 2024b; Mitra et al., 2024) typically decompose either the question or the image, our framework decomposes both. This dual decomposition improves coherence between question-derived and image-derived insights, enabling agents to handle complex compositional reasoning. Our method addresses the alignment issues present in prior VQA systems (Zheng et al., 2024) by balancing visual and linguistic reasoning.

3 Reasoning with Decomposed Inputs

We introduce our hierarchical approach for visual question answering in four parts. First, we provide an overview of the two levels (the orchestrator level and the worker agents level), key components, and notation. Second, we describe the decomposition at the orchestrator level. Third, we explain the design of the worker agent level. Finally, we describe how worker agent responses are aggregated to produce the final answer.

3.1 Overview

As illustrated in Fig. 2, the *orchestrator* Π^{orc} divides the visual question answering task into K *sub-questions* $\{Q_k\}_{k=1}^{K}$ and visual sub-domains $\{M_k\}$ (segmentation, material, depth, color, and blind). At the lower level, we design K *worker agents* $\{\pi_k\}_1^K$ that specialize in specific visual subtasks. Each worker agent produces a response $\mathcal{A}_k$. The orchestrator Π^{orc} aggregates these responses $\{\mathcal{A}_k\}$ to produce the final answer $\mathcal{A}$.

3.2 Decomposition into Sub-questions and Visual Sub-domains (Orchestrator Level)

We transform the question and image input (Q, I) into sub-task tuples $(Q_k, I, M_k)_1^K$.

Question		Are there an equal number of tiny metal balls behind the red cube and large cyan objects to the left of the brown cylinder?
Segmentation (M_1)	t_1 :	Identify and count specific objects.
	Q_1:	How many tiny balls are there and how many large objects are present? What is the positioning of them?
Material (M_2)	t_2 :	Verify material of objects
	Q_2:	Are the identified balls and large objects made of metal?
Depth (M_3)	t_3 :	Determine spatial relationship of objects
	Q_3:	What objects are behind the red cube and to the left of the brown cylinder?
Color (M_4)	t_4 :	Identify colors of objects
	Q_4:	What are the colors of the cube and the large objects?

Figure 3: Orchestrator Agent transforms user input to *sub-task tuples* $(Q_k, I, M_k)_1^K$.

The Orchestrator first processes Q to identify the sub-domains that the query references. That is, Π^{orc} maps the question to relevant visual sub-domains, $Q \rightarrow \{M_1, M_2, \dots, M_K\}$. For the identified K sub-domains, the orchestrator infers a corresponding set of template sub-questions $\{t_1, t_2, \dots, t_K\}$. Given M_k and t_k, the orchestrator generates natural language sub-questions $Q_k = \Pi^{\text{orc}}(t_k, Q)$. Overall, the original query is decomposed into an ordered set of sub-questions $\{Q_1, Q_2, \dots, Q_K\}$. All three steps (generating visual sub-domain M_k, template sub-question t_k, and natural language sub-question Q_k) are executed using a single prompt $\mathcal{P}_{\text{orc}}$ (see Appendix F). The output of the orchestrator level, the *sub-task tuple* (Q_k, I, M_k), is passed to the worker level.

3.3 Visual Encoding and Executing Sub-Tasks (Worker Agents Level)

We transform the sub-task tuples $(Q_k, I, M_k)_1^K$ into responses $\{\mathcal{A}_k\}$.

Encoding visual sub-domains $I \rightarrow \{\phi_k\}_1^K$. Each visual sub-domain M_k is encoded as $\phi_k = \texttt{Encoder}[M_k](I)$, where $\texttt{Encoder}[M_k]$ is a domain-specific encoder (see Sec. B). The resulting tuple (Q_k, ϕ_k, M_k) is processed by a task-conditioned MLLM to produce a response $\hat{\mathcal{A}}_k$. Task-conditioning uses prompts $\mathcal{P}_k$, such that $\hat{\mathcal{A}}_k = \text{MLLM}(Q_k, \phi_k | \mathcal{P}_k)$. We defer the details of visual encoders and specific prompts $\mathcal{P}_k$ (for worker agent π_k) to Appendix B and Appendix F, respectively.

Input		Sub-task tuples $\{(Q_k, I, M_k)\}$
Segmentation Agent (π_1)	A_1:	In the image, there are 3 balls, labeled as 4, 5, and 6, with object 6 being the smallest in size. Additionally, there are 4 large objects, labeled as 1, 3, 4, and 5. Regarding the positioning, from left to right, the order of the objects is: object 8, followed by objects 4 and 7, then objects 6, 5, 1, and objects 9 and 10. The last two are objects 2 and 3.
Material Agent (π_2)	A_2:	Based on the analysis of the metalness map, the objects labeled as 4, 6, 3, 2, 10 objects appear brighter in the metalness map, indicating that they are made of metal.
Depth Agent (π_3)	A_3:	The objects in the image are ordered from front to back as follows: Object 1 is at the front, followed by Objects 2 and 6. Next, we have Objects 3, 4, 8, and 7. Finally, the objects after 7 are Objects 5, 9, and 10.
Color Agent (π_4)	A_4:	The colors of the specified objects are as follows: Cube (object 2): Red; Large objects (objects 1 and 3): Pink and Teal respectively; Cylinder (object 10): brown.

Figure 4: Output responses from worker agents.

Ans	No
Reasoning	In the image, the red cube is labeled as '2'. Behind this red cube, there are no tiny metal balls visible. According to the color agent, there is only one cyan-colored object, and it is the large object labeled as '3'. According to the segmentation agent, object '3' is positioned on the right side of the brown cylinder, which is labeled as object '10'. Therefore, there are no cyan objects to the left of the brown cylinder. Hence, the answer is 'No'.

Figure 5: Final answer output after reasoning step.

Refining responses $\hat{\mathcal{A}}_k \rightarrow \mathcal{A}_k$. Instruction-tuned models can estimate their confidence scores (Tian et al., 2023). We leverage this by augmenting prompts $\mathcal{P}_k$ to request confidence scores alongside responses. Worker agents use these scores to iteratively refine their outputs through a feedback loop. Formally, $\hat{\mathcal{A}}_k, \mathcal{C}_k = \text{MLLM}(Q_k, \phi_k | \mathcal{P}_k)$. If $\mathcal{C}_k \leq \tau$, the agent enters a feedback loop where at each iteration i, it outputs a reasoning trace $\hat{\mathcal{A}}_k^i$ and confidence $\mathcal{C}_k^i$: $(\hat{\mathcal{A}}_k^i, \mathcal{C}_k^i) = \text{MLLM}\big(Q_k,\ \phi_k,\ \{\hat{\mathcal{A}}_k^{j \leq i}\} \,\big|\, \mathcal{P}_k\big)$.

The agent iteratively refines its response $\hat{\mathcal{A}}_k$ until the confidence exceeds threshold τ or the maximum iteration count is reached. The final refined response is denoted $\mathcal{A}_k$. For additional details about each agent, see Appendix B.

3.4 Consistency-Aware Aggregation (Orchestrator Level)

This step transforms the worker agent responses $\{\mathcal{A}_k\}$ into the final answer $\mathcal{A}$.

The ordered sequence of responses $[\mathcal{A}_k]_1^K$ is aggregated by the orchestrator Π^{orc} through *meta-reasoning* (Pěchouček et al., 2003; Langlois et al., 2020). This process performs (1) *validity checks* to identify inconsistencies among responses from different π_k; (2) *refinement* to fill in missing details and adjust partial responses; and (3) *override* to correct conflicting worker agent responses. To implement this, we condition Π^{orc} with the original question Q and image I, along with a targeted prompt $\mathcal{P}_{agg}$ (see Appendix F). The final output is $\mathcal{A} = \Pi^{\text{orc}}\left([\mathcal{A}_k]_1^K, Q, I|\mathcal{P}_{agg}\right)$, where we overload the Π^{orc} notation for readability.

By retaining final decision-making autonomy, the orchestrator provides an error-resilient aggregation mechanism. This design mitigates *error propagation* common in hierarchical methods, where the final answer depends heavily on sub-task correctness. For quantitative analysis of this mechanism, see Sec. 4.3 and Tab. 4.

3.5 Visual Grounding across REDI Levels

The orchestrator communicates sub-task tuples (Q_k, I, M_k) to the worker agent level and receives responses $\mathcal{A}_k$ in return. In preliminary experiments, we found that agents made errors when referring to specific objects in this bi-level communication. To *ground* the communication, REDI adopts object centroids (Zhou et al., 2019; Yin et al., 2021). We identify objects of interest (e.g., ‘red cube’), generate bounding boxes, and annotate the center of each box to serve as a *pointer*. We augment image I with these object centroid annotations (see numbers in Fig. 2). We use text-based SAM (Kirillov et al., 2023) for object identification.

3.6 Implementation Details

We use GPT-4V as the MLLM for both the Orchestrator and Worker agents to enable direct comparison with existing methods that use the same model. For visual encoding, we use a color normalization encoder (Csink et al., 1998) to remove photometric effects, a depth encoder (Bochkovskii et al., 2024a) to generate depth maps, and a material encoder (Zeng et al., 2024a) to produce material maps. For additional details, see Appendix C.

In practice, not all visual sub-domains M_k are necessary for every question. For example, the question ‘How many pillows are there in the image?’ from SEEDBench does not require the depth worker agent π_3. Therefore, REDI selects only the K relevant visual sub-domains for each input question (see prompt $\mathcal{P}_{\text{orc}}$ in Appendix F).

4 Experiments

We detail our experimental framework and findings. We begin by describing the evaluation benchmarks and baseline models (Sec. 4.1). Next, we present quantitative results across three benchmarks (Sec. 4.2). Finally, we provide ablations and analyses of REDI (Sec. 4.3).

4.1 Benchmarks and Baselines

We evaluate REDI using three standardized benchmarks and compare it against established baselines for compositional and multimodal reasoning.

Benchmarks. We assess the generalization of REDI on three benchmarks for multimodal reasoning: `SEEDBench`, `MMBench`, and `CLEVR`, which encompass visual question answering tasks on both natural and synthetic images.

- `MMBench` (Liu et al., 2024b) consists of over 3k questions spanning 20 abilities (*e.g.* function reasoning, spatial relationship) designed to challenge compositional reasoning in MLLMs.
- `SEEDBench` (Li et al., 2023) is a large-scale benchmark to assess generative comprehension in MLLMs. We evaluate on the image split (SEED-I), which consists of 9 evaluation dimensions (*e.g.* Scene Understanding, Instance Location) and over 14k questions.
- `CLEVR` (Johnson et al., 2016) comprises synthetic images of 3D objects with varied shapes, colors, sizes, and spatial arrangements. Questions in CLEVR involve occlusion, object comparison (*e.g.* size and shape), and complex attribute reasoning (*e.g.* question in Fig. 2).

Baselines. The REDI framework can be integrated with any MLLM, whether open or closed-source. We report results with InstructBLIP (Dai et al., 2023), LLaVa-1.5-13B (Chen et al., 2023), and GPT-4V (Achiam et al., 2023). We compare REDI with four state-of-the-art compositional methods applicable across different MLLMs:

- `-CoT` (Kojima et al., 2022) employs zero-shot chain-of-thought (CoT) prompting to induce intermediate reasoning steps without extra training.
- `-CCoT` (Mitra et al., 2024) integrates scene graphs with CoT reasoning for multi-step inference.
- `-Viper` (Surís et al., 2023) employs a visual programming framework for code-driven reasoning.
- `-Cantor` (Gao et al., 2024) employs prompt engineering with a single MLLM to perform various sub-tasks (*e.g.* text extraction, object counting)

We utilize the official implementation of CoT,

Methods	Accuracy on Benchmarks ↑		
	SEED-I	MMBench	CLEVR
Prior representative works			
BLIP2	46.4	–	–
mPlug-OWL2	57.8	64.5	–
QwenVL-Chat	58.2	61.2	–
Cola-Zero	–	–	34.3
Cola-FT	–	–	54.3
Open-sourced (Dai et al., 2023; Liu et al., 2024a)			
IBLIP	48.2	36.0	28.9
IBLIP-CoT	37.6	25.3	31.4
IBLIP-CCoT	56.9	40.3	34.7
IBLIP-REDI	**60.0**	**42.4**	**38.5**
LLaVA	68.2	67.0	34.4
LLaVA-CoT	66.7	66.0	36.7
LLaVA-CCoT	69.7	70.7	38.7
LLaVA-REDI	**71.3**	**72.1**	**40.9**
Closed-sourced (Achiam et al., 2023)			
GPT4V	69.1	75.5	45.2
GPT4V-CoT	72.5	74.8	50.3
GPT4V-CCoT	74.0	76.3	51.2
GPT4-Viper	73.6	75.8	49.9
GPT4V-Cantor	64.8	78.8	44.0
GPT4V-REDI$_{\text{eff}}$	**75.6**	**80.3**	**58.4**
GPT4V-REDI	**78.0**	**83.7**	**61.2**

Table 1: **Results on SEED-I, MMBench, and CLEVR.** Our REDI framework, across three standard compositional reasoning benchmarks, significantly improves answer accuracy *vs.* GPT4V by 8.9%, 8.2%, & 16.0%, respectively. We also demonstrate the generalization of REDI across both open- and closed-source base MLLMs *i.e.* {InstructBLIP, LLaVA} and GPT4V, respectively. REDI$_{\text{eff}}$ Dashes denote that corresponding method does not report results on the benchmark.

CCoT, Cantor and Viper (with their innate support for GPT4). For completeness, we report metrics of additional prior works from (Mitra et al., 2024).

4.2 Quantitative Results

In the following, we include takeaways based on the results in Tab. 1, 2, and 3.

State-of-the-art Performance Across Benchmarks and Base MLLMs (Tab. 1). REDI improves over existing compositional reasoning methods (CoT, CCoT, ViperGPT). For GPT-4V, adding REDI boosts performance by 10% on CLEVR, 4.9% on MMBench, and 4% on SEEDBench relative to the best baseline. Similar gains were observed with InstructBLIP and LLaVA, underscoring the generalizability of REDI. The improvement on CLEVR, an out-of-domain dataset, demonstrates effectiveness in handling varied contexts. CLEVR is characterized by modality-entangled questions, where REDI achieves a 10% improvement over the state-of-the-art baseline and a 16% improvement over GPT-4V, underscoring joint reasoning over language and vision.

Efficient REDI (Tab. 1). While REDI delivers strong results, it requires an average of 7.8 MLLM calls per task (across all datasets) due to its iterative refinement cycles. Hence, we create a cost-efficient variant, REDI$_{\text{eff}}$, by omitting the refinement cycles, reducing the average call count to just 3.6. REDI$_{\text{eff}}$ achieves competitive performance, surpassing all baselines despite a modest drop relative to full REDI, thereby offering a favorable trade-off between efficiency and accuracy.

Enhanced Reasoning Abilities (Tab. 2, 3). REDI outperforms the state-of-the-art across all reasoning categories. On MMBench, we achieve improvements of +5.5% in LR, +4.0% in AR, and +2.1% in RR. Similarly, on SEEDBench, we observe gains of +7.2% in SR and +3.1% in VR. These enhancements are primarily attributed to our decomposition strategy, which breaks down a complex query into sub-tasks, effectively reducing the cognitive load on individual worker agents.

Improved Object understanding (Tab. 2 & 3). Strong performance in FP-S (+5.8) on MMBench, along with notable gains in IId (+1.7), IA (+1.9), IL (+3.1), and IIn (+1.1) on SEEDBench, indicates that the introduction of task-conditioned worker agents enhances fine-grained object understanding.

4.3 Ablations and Analysis

In addition to reporting results across three benchmarks, we perform targeted ablations to dissect both levels of our REDI framework, compare its performance with human baselines, create its efficient version, and present the failure modes.

Every worker agent adds to performance (Table 4, rows 1-6). We quantify the impact of removing individual agents from REDI. Our full method achieves 61.2% accuracy (row 1). When we remove the Material agent, the accuracy drops by 5.9%. Omitting the Blind, Color, and Depth agents reduces accuracy by 2.8%, 4.3%, and 7.1%, respectively. We observe the largest decline (an 8.9% drop) when we remove the Segmentation agent

Method	Overall	SU	IId	IA	IL	SR	VR	TU	IC	IIn
LLaVA-DDCoT	58.0	63.0	59.8	64.1	44.6	41.4	67.1	57.7	47.3	51.6
LLaVA-VidIL	68.9	74.9	72.5	69.9	62.5	53.9	78.0	49.4	**62.3**	71.1
LLaVA-CCoT	69.7	76.0	74.4	71.8	64.3	54.5	79.2	58.8	59.3	74.2
LLaVA-REDI	**71.3**	**78.8**	**76.1**	**73.7**	**67.4**	**61.7**	**82.3**	**63.5**	61.7	**75.3**
Δ Accuracy	+1.6	+2.8	+1.7	+1.9	+3.1	+7.2	+3.1	+4.7	−0.6	+1.1

Table 2: **Attribute-based performance comparison on the SEED-Image benchmark.** The table reports accuracy (%) for Overall, Scene Understanding (SU), Instance Identity (IId), Instance Attributes (IA), Instance Location (IL), Spatial Relation (SR), Visual Reasoning (VR), Text Understanding (TU), Instance Counting (IC), and Instance Interaction (IIn). The Δ Accuracy row reports the absolute improvement of LLaVA-REDI over the second-best method for each attribute. Prominent gains in Spatial Relation (SR), Visual Reasoning (VR), and Instance Location (IL) are seen due to REDI's novel compositionality along visual sub-domains.

Method	LR	AR	RR	FP-S	CP
IBLIP	11.5	43.6	35.5	36.6	**22.3**
IBLIP-CCoT	12.5	45.8	40.9	40.7	22.1
IBLIP-REDI	**13.1**	**51.3**	**44.9**	**42.5**	22.1
Δ Accuracy	+0.6	+5.5	+4.0	+1.8	−0.2
LLaVA	39.9	74.7	61.6	70.9	59.9
LLaVA-CCoT	44.2	72.1	75.3	73.7	**81.2**
LLaVA-REDI	**49.7**	**76.1**	**77.4**	**79.5**	80.5
Δ Accuracy	+5.5	+4.0	+2.1	+5.8	−0.7

Table 3: **Detailed results on MMBench Reasoning splits.** Logic Reasoning (LR), Attribute Reasoning (AR), Relation Reasoning (RR), Fine-Grained (Single Object) (FP-S), and Coarse Perception (CP). For both IBLIP and LLaVA baselines, integrating REDI consistently improves performance in LR, AR, RR, and FP-S (gains up to +5.8%), with a slight decrease in CP.

#	Method	Acc.	ΔAcc.
1	REDI (all agents)	**61.2**	
2	w/o Blind	58.4	−2.8
3	w/o Color	56.9	−4.3
4	w/o Material	55.3	−5.9
5	w/o Depth	54.1	−7.1
6	w/o Segmentation	52.3	−8.9
7	w/ "Don't know" agents	44.8	−16.4
8	GPT4V (no agents)	45.2	−16.0

Table 4: **Ablation Study on CLEVR.** Our REDI model with all agents achieves 61.2% accuracy. When ablating one agent at a time (i.e., non-cumulatively), significant performance drops are observed. "Don't know" worker agents and GPT4V (no agents) are provided as lower bounds (details in Sec. 4.2). The Δ Acc column quantifies the individual contributions of the worker agents.

(row 6). These results show that each agent contributes uniquely to performance and confirm that our multi-agent design is essential for robust compositional reasoning.

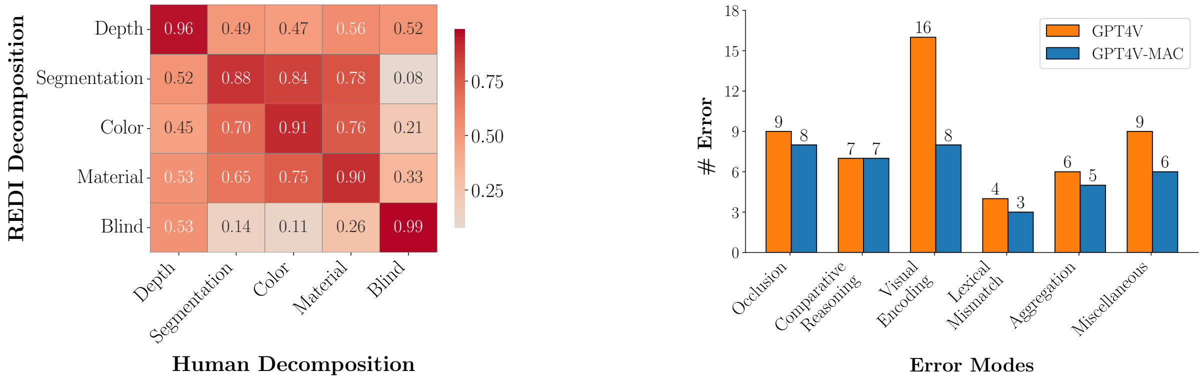
Orchestrator's aggregation is error-resilient (Table 4, rows 7 & 8). We evaluate the error-resilience of the orchestrator agent by forcing the worker agents to always respond with the degenerate statement "I don't know anything about this," regardless of input. We expect that orchestrator 's compositional reasoning will be resilient to these dysfunctional responses, reverting to near base-model performance. Our results confirm this: the baseline accuracy drops to 44.8% (row 7), closely matching GPT4V's performance of 45.2% (row 8).

Decomposition Agreement Analysis (Fig. 6a). We select 100 VQA tasks each from `CLEVR` and `SEED-I` and manually decompose them into visual sub-domains. These are compared with the sub-domains selected by the orchestrator. To evaluate agreement, we compute pairwise agreement between corresponding human and orchestrator sub-domains. The agreement matrix calculation and inter-annotator score analysis are detailed in Appendix D. Fig. 6a shows that diagonal elements in the agreement matrix exceed 0.9, indicating that human and REDI decompositions are highly consistent. This confirms that REDI decomposition is strategic and unbiased.

Failure Modes (Fig. 6b). We randomly sampled 100 VQA tasks from the `CLEVR` evaluation. In this sample, GPT-4V-REDI made 37 mistakes compared to 51 by GPT-4V[1]. We classify these into failure modes: *Occlusion*, where one or more objects are obscured by another; *Comparative Reasoning*, where the model fails to correctly compare properties between objects; *Visual Encoding*,

---

[1] Recall that the corresponding accuracies from Tab. 1 are 61.2% and 45.2%, respectively.



(a) REDI *vs.* Human decomposition to visual sub-domains.

(b) Error Analysis

Figure 6: **(a)** Agreement between REDI and human decomposition into visual sub-domains based on 100 random samples each from `CLEVR` and `SEED-I`. **(b)** Distribution of errors over failure modes (GPT-4V-REDI: 37 errors; GPT-4V: 51 errors) based on 100 random samples from `CLEVR`.

where mistakes originate in the perception (*e.g.*, ϕ of the worker agents for REDI); *Lexical Mismatch*, where the predicted answer is semantically close but not an exact match (*e.g.*, "light red" instead of "pink"); *Aggregation*, where available information is improperly combined; and *Miscellaneous* for all other uncategorized mistakes. Notably, our approach halves the incidence of Visual Encoding mistakes, directly demonstrating the benefit of decomposing multimodal tasks and offloading each to worker agents that tackle specific visual sub-domains. For additional details on failure modes and their categorization, see Appendix D.

5 Conclusion

We propose REDI, a multi-agent framework that jointly decomposes visual and textual inputs into fine-grained sub-domains (segmentation, material, depth, and color) and generates corresponding sub-questions. Specialized worker agents solve each sub-task, while an orchestrator performs meta-reasoning to aggregate responses and mitigate error propagation. On CLEVR, SEEDBench, and MM-Bench, REDI outperforms state-of-the-art compositional multimodal reasoning methods, offering improved accuracy, robustness, and transparency by isolating sub-domain contributions.

6 Discussion and Limitations

REDI is a hierarchical framework that leverages specialized encoders and an orchestrator for compositional multimodal reasoning in vision-language tasks. By decomposing both visual and language inputs into fine-grained visual sub-domains and assigning them to dedicated worker agents, our approach enhances interpretability and flexibility compared to end-to-end systems. This compositional design facilitates query-driven reasoning and provides insights into the contribution of each sub-domain.

Encoder Dependency and Grounding Limitations The system's performance depends on the accuracy of its underlying encoders. Inaccuracies in segmentation or depth estimation, for instance, can propagate through the reasoning pipeline, compromising the final output. Our current implementation relies on SAM for object segmentation, and its limitations (particularly in handling occluded objects and providing robust annotations) can lead to grounding errors. Moreover, the framework is not occlusion-proof, and its applicability may be constrained to image types where the visual content is clearly separable. An orthogonal benefit of our approach is that enhancements to individual encoders can be seamlessly integrated, ensuring that as more robust, self-supervised models emerge, the overall performance of the system will correspondingly improve.

Limitations in Image Types While the system performs reliably on a majority of images, certain scenarios can challenge the depth estimation component. For example, in natural images with a morning sky, the depth map may become skewed due to variations in lighting and atmospheric conditions. Although most images yield accurate depth estimations, such cases can lead to degraded performance in depth analysis, further affecting the overall reasoning pipeline.

Computational Overhead and Scalability Challenges The iterative interplay among agents, while beneficial for complex reasoning, introduces significant computational overhead, potentially limiting real-time or time-sensitive applications. Scalability remains a challenge, as coordinating multiple agents demands streamlined communication protocols and compact intermediate representations. Techniques such as model distillation, quantization, and parallelized agent communication could mitigate these overheads in future work.

Despite these limitations, the structured, modular approach of REDI offers a promising foundation for advancing multimodal reasoning. As more robust, self-supervised, and domain-adapted encoders emerge, they can be seamlessly integrated into our framework, further enhancing its accuracy and robustness without requiring complete retraining. Future research will focus on addressing these scalability and encoder-dependency issues, aiming to extend the framework's applicability across a broader range of image types and challenging real-world scenarios.

7 Acknowledgement

This research is based upon work supported by U.S. DARPA ECOLE Program No. #HR00112390060. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R Richter, and Vladlen Koltun. 2024a. Depth pro: Sharp monocular metric depth in less than a second. *arXiv preprint arXiv:2410.02073*.

Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, and Vladlen Koltun. 2024b. Depth pro: Sharp monocular metric depth in less than a second. *Preprint*, arXiv:2410.02073.

Megha Chakraborty, Khushbu Pahwa, Anku Rani, Shreyas Chatterjee, Dwip Dalal, Harshit Dave, Preethi Gurumurthy, Adarsh Mahor, Samahriti Mukherjee, Aditya Pakala, et al. 2023. FACTIFY3M: A benchmark for multimodal fact verification with explainability through 5W Question-Answering. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15282–15322.

Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F. Karlsson, Jie Fu, and Yemin Shi. 2024a. Autoagents: A framework for automatic agent generation. *Preprint*, arXiv:2309.17288.

Liangyu Chen, Bo Li, Sheng Shen, Jingkang Yang, Chunyuan Li, Kurt Keutzer, Trevor Darrell, and Ziwei Liu. 2023. Large language models are visual reasoning coordinators. *Preprint*, arXiv:2310.15166.

Liangyu Chen, Bo Li, Sheng Shen, Jingkang Yang, Chunyuan Li, Kurt Keutzer, Trevor Darrell, and Ziwei Liu. 2024b. Large language models are visual reasoning coordinators. *Advances in Neural Information Processing Systems*, 36.

Xupeng Chen, Zhixin Lai, Kangrui Ruan, Shichu Chen, Jiaxiang Liu, and Zuozhu Liu. 2025. R-llava: Improving med-vqa understanding through visual region of interest. *Preprint*, arXiv:2410.20327.

L Csink, D Paulus, U Ahlrichs, and B Heigl. 1998. Color normalization and object localization. *Rehrmann [42]*, pages 49–55.

Wenliang Dai, Junnan Li, D Li, AMH Tiong, J Zhao, W Wang, B Li, P Fung, and S Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. arxiv 2023. *arXiv preprint arXiv:2305.06500*, 2.

Dwip Dalal, Utkarsh Mishra, Narendra Ahuja, and Nebojsa Jojic. 2025a. City Navigation in the Wild: Exploring Emergent Navigation from Web-Scale Knowledge in MLLMs. *arXiv preprint arXiv:2512.15933*.

Dwip Dalal, Gautam Vashishtha, Utkarsh Mishra, Jeonghwan Kim, Madhav Kanda, Hyeonjeong Ha, Svetlana Lazebnik, Heng Ji, and Unnat Jain. 2025b. Constructive Distortion: Improving MLLMs with Attention-Guided Image Warping. *arXiv preprint arXiv:2510.09741*.

Timin Gao, Peixian Chen, Mengdan Zhang, Chaoyou Fu, Yunhang Shen, Yan Zhang, Shengchuan Zhang, Xiawu Zheng, Xing Sun, Liujuan Cao, et al. 2024. Cantor: Inspiring multimodal chain-of-thought of mllm. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 9096–9105.

James J. Gibson. 2002. A theory of direct visual perception. In *Vision and Mind: Selected Readings in the Philosophy of Perception*. The MIT Press.

Tanmay Gupta and Aniruddha Kembhavi. 2022. Visual programming: Compositional visual reasoning without training. *Preprint*, arXiv:2211.11559.

Chuxuan Hu, Dwip Dalal, and Xiaona Zhou. 2025. A Dataset-Centric Survey of LLM-Agents for Data Science.

J. Paul Getty Museum. 2011. Understanding formal analysis.

Unnat Jain, Svetlana Lazebnik, and Alexander G Schwing. 2018. Two can play this game: Visual dialog with discriminative question generation and answering. In *CVPR*.

Unnat Jain, Ziyu Zhang, and Alexander G Schwing. 2017. Creativity: Generating diverse questions using variational autoencoders. In *CVPR*.

Bowen Jiang, Zhijun Zhuang, Shreyas S. Shivakumar, Dan Roth, and Camillo J. Taylor. 2024. Multi-agent vqa: Exploring multi-agent foundation models in zero-shot visual question answering. *Preprint*, arXiv:2403.14783.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2016. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *Preprint*, arXiv:1612.06890.

Fucai Ke, Zhixi Cai, Simindokht Jahangard, Weiqing Wang, Pari Delir Haghighi, and Hamid Rezatofighi. 2024. Hydra: A hyper agent for dynamic compositional visual reasoning. *Preprint*, arXiv:2403.12884.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Samuel T Langlois, Oghenetekevwe Akoroda, Estefany Carrillo, Jeffrey W Herrmann, Shapour Azarm, Huan Xu, and Michael Otte. 2020. Metareasoning structures, problems, and modes for multiagent systems: A survey. *IEEE Access*, 8:183080–183089.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. 2023. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*.

Jingxiang Lin, Unnat Jain, and Alexander Schwing. 2019. Tab-vcr: Tags and attributes based vcr baselines. *Advances in Neural Information Processing Systems*.

Yuanze Lin, Yujia Xie, Dongdong Chen, Yichong Xu, Chenguang Zhu, and Lu Yuan. 2022. Revive: Regional visual representation matters in knowledge-based visual question answering. *Preprint*, arXiv:2206.01201.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in neural information processing systems*, 36.

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

Chancharik Mitra, Brandon Huang, Trevor Darrell, and Roei Herzig. 2024. Compositional chain-of-thought prompting for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14420–14431.

Dhruvesh Patel, Hamid Eghbalzadeh, Nitin Kamra, Michael Louis Iuzzolino, Unnat Jain, and Ruta Desai. 2023. Pretrained language models as visual planners for human assistance. In *ICCV*.

Michal Pěchouček, Olga Štěpánková, Vladimír Mařík, and Jaroslav Bárta. 2003. Abstract architecture for meta-reasoning in multi-agent systems. In *Multi-Agent Systems and Applications III: 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003 Prague, Czech Republic, June 16–18, 2003 Proceedings 3*, pages 84–99. Springer.

Anku Rani, Dwip Dalal, Shreya Gautam, Pankaj Gupta, Vinija Jain, Aman Chadha, Amit Sheth, and Amitava Das. 2025. Sepsis: I can catch your lies–a new paradigm for deception detection. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 97–128.

Anku Rani, S.M Towhidul Islam Tonmoy, Dwip Dalal, Shreya Gautam, Megha Chakraborty, Aman Chadha, Amit Sheth, and Amitava Das. 2023. FACTIFY-5WQA: 5W aspect-based fact verification through question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10421–10440, Toronto, Canada. Association for Computational Linguistics.

Ruoyue Shen, Nakamasa Inoue, and Koichi Shinoda. 2024. Pyramid coder: Hierarchical code generator for compositional visual question answering. *Preprint*, arXiv:2407.20563.

Fatemeh Shiri, Xiao-Yu Guo, Mona Golestan Far, Xin Yu, Gholamreza Haffari, and Yuan-Fang Li. 2024. An empirical analysis on spatial reasoning capabilities of large multimodal models. *arXiv preprint arXiv:2411.06048*.

Sanjay Subramanian, Medhini Narasimhan, Kushal Khangaonkar, Kevin Yang, Arsha Nagrani, Cordelia Schmid, Andy Zeng, Trevor Darrell, and Dan Klein. 2023. Modular visual question answering via code generation. *Preprint*, arXiv:2306.05392.

Dídac Surís, Sachit Menon, and Carl Vondrick. 2023. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11888–11898.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5238–5248.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. *arXiv preprint arXiv:2305.14975*.

Aayush Atul Verma, Amir Saeidi, Shamanthak Hegde, Ajay Therala, Fenil Denish Bardoliya, Nagaraju Machavarapu, Shri Ajay Kumar Ravindhiran, Srija Malyala, Agneet Chatterjee, Yezhou Yang, et al. 2024. Evaluating multimodal large language models across distribution shifts and augmentations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5314–5324.

Yanan Wang, Michihiro Yasunaga, Hongyu Ren, Shinya Wada, and Jure Leskovec. 2023. Vqa-gnn: Reasoning with multimodal knowledge via graph neural networks for visual question answering. *Preprint*, arXiv:2205.11501.

Zeqing Wang, Wentao Wan, Qiqing Lao, Runmeng Chen, Minjie Lang, Keze Wang, and Liang Lin. 2024. Towards top-down reasoning: An explainable multi-agent approach for visual question answering. *Preprint*, arXiv:2311.17331.

Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. 2021. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793.

Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. 2024a. Rgb x: Image decomposition and synthesis using material-and lighting-aware diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11.

Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. 2024b. Rgbx: Image decomposition and synthesis using material- and lighting-aware diffusion models. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24*, SIGGRAPH '24, page 1–11. ACM.

Haowei Zhang, Jianzhe Liu, Zhen Han, Shuo Chen, Bailan He, Volker Tresp, Zhiqiang Xu, and Jindong Gu. 2024a. Visual question decomposition on multimodal large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1926–1949, Miami, Florida, USA. Association for Computational Linguistics.

Zhehao Zhang, Ryan Rossi, Tong Yu, Franck Dernoncourt, Ruiyi Zhang, Jiuxiang Gu, Sungchul Kim, Xiang Chen, Zichao Wang, and Nedim Lipka. 2024b. Vipact: Visual-perception enhancement via specialized vlm agent collaboration and tool-use. *arXiv preprint arXiv:2410.16400*.

Changmeng Zheng, Dayong Liang, Wengyu Zhang, Xiao-Yong Wei, Tat-Seng Chua, and Qing Li. 2024. A picture is worth a graph: A blueprint debate paradigm for multimodal reasoning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 419–428.

Qiji Zhou, Ruochen Zhou, Zike Hu, Panzhong Lu, Siyang Gao, and Yue Zhang. 2024. Image-of-thought prompting for visual reasoning refinement in multimodal large language models. *arXiv preprint arXiv:2405.13872*.

Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. 2019. Objects as points. *arXiv preprint arXiv:1904.07850*.

Appendix

In this appendix, we include the following to supplement the main paper:

- A Step-by-step algorithm of our REDI (Fig. 2 and Sec. 3).
- B Detailed workflow of each worker agent (Fig. 2 and Sec. 3.3).
- C Additional implementation details (Sec. 3.6).
- D Details of the qualitative analysis (Sec. 4.3).
- F Prompts used in our REDI (Sec. 3).
- E Illustrative examples of our REDI(Sec. 4).

A Algorithm

Algorithm 1: Reasoning with Decomposed Inputs (REDI)

```
Input: Image $I$, user query $Q$.
Output: Final answer $\mathcal{A}$.
  • Notation: $\Pi^{\text{orc}}$ : Orchestrator; $\{\pi_1, \pi_2, \ldots, \pi_K\}$: Worker Agents.
  • Step 1: Visual Sub-Domain Identification, and sub-question generation (Orchestrator Level)
      – Extract visual sub-domains from $Q$: $\rightarrow \{M_1, \ldots, M_K\}$ (e.g., color, material, depth, ...)
      – Determine specialized tasks: $\text{TaskMapping}\big(\{M_1, \ldots, M_K\}\big) \rightarrow \{t_1, \ldots, t_K\}$
      – Generate sub-question: $\Pi^{\text{orc}}.\text{GenerateSubQuestion}(\text{t}_k, Q) \rightarrow Q_k$
  • Step 2: Object Extraction and Annotation (Orchestrator Level)
      – $\Pi^{\text{orc}}$.ExtractObjects(I, Q) $\rightarrow \mathcal{O}$
      – For each $o_j \in \mathcal{O}$: $\text{SAM}(o_j) \rightarrow I^*$
  • Step 3: Image Decomposition and Response Creation (Worker Level)
      – For $k = 1, \ldots, K$:
          * Select appropriate Worker Agent: $\Pi^{\text{orc}}.\text{SelectAgent}(\text{t}_k) \rightarrow w_k \quad \in \{\pi_1, \ldots, \pi_K\}$
          * Image decomposition: $\phi_k = \texttt{Encoder}[M_k](I^*)$ using domain-specific encoders.
      – Obtain agent's answer with self-calibration: $\pi_k(Q_k, \phi_k) \rightarrow \mathcal{A}_k$
  • Step 4: $\Pi^{\text{orc}}$ and Meta-Reasoning
      – Fuse sub-answers: $\Pi^{\text{orc}}\big(\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k\big) \rightarrow \mathcal{A}$
      – Meta-reasoning: (a) Validity checks for conflicting answers
        (b) Refinements to fill in missing details
        (c) Overrides incorrect or contradictory sub-agent outputs.
  • return $\mathcal{A}$
```

B Worker Agents and Encoders

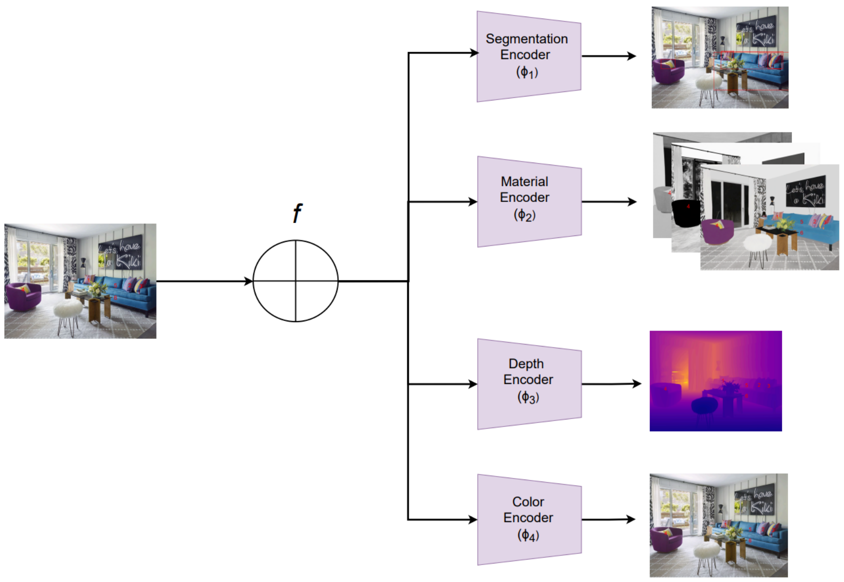


Figure 7: The figure of the composite encoder that we used here. The f is a conditional function that directs the input image to the encoder based on sub-domain

Our approach builds on the premise that decomposing visual information into specialized sub-domains improves compositional reasoning. This design philosophy extends prior work on attribute-based visual reasoning (Lin et al., 2019), which demonstrated that associating visual features with structured semantic attributes enhances performance on complex reasoning tasks.

1. **Segmentation Agent and Encoder:** The Segmentation Agent and Encoder plays a crucial role in identifying and categorizing objects within an image. It leverages the Segment Anything Model (SAM) (Kirillov et al., 2023) to systematically segment the image into distinct regions, each corresponding to an individual object or part. This segmentation process enables a structured left-to-right ordering of objects and facilitates the counting of specific object classes when needed. The resulting segmentation data serves as a critical reference for other agents involved in processing the image.

   To ensure clear and structured reasoning, each identified object is assigned a unique label, creating a standardized reference system that all agents can use. This labeling mechanism eliminates ambiguity and enhances communication efficiency across different agents, allowing them to share object-specific information seamlessly. By following this approach, the system maintains consistency, avoids redundancy, and enables a more organized exchange of information among the agents.

2. **Depth Agent and Encoder:** This agent employs a pre-trained model described in (Bochkovskii et al., 2024b), ensuring accurate and reliable depth estimation. Utilizing the depth map, this agent determines the relative distances between objects, identifying which are closer or farther from the camera. It provides qualitative spatial relationships based on depth values, using annotated labels as input references.

3. **Color Agent and Encoder:** This agent normalizes the color of the image and examines the annotated image to identify the inherent colors of objects, unaffected by lighting or shading. It provides clear color descriptions for each object as specified by the central planner.

4. **Material Agent and Encoder:** This agent employs a pre-trained model by (Zeng et al., 2024b) for generating metallic, albedo, roughness maps to infer surface properties, such as distinguishing between metallic and non-metallic objects. Unlike other agents that utilize a single image input, it takes in three decomposed images generated using the encoder to answer the results. The following

is the description of each of the used map and their use in classifying the material of the provided image:

(a) **Albedo**: This map represents the base color of a surface without any influence from lighting, shadows, or reflections. It captures the inherent diffuse reflectance of a material, making it a crucial component in separating material properties from environmental illumination. Since albedo maps remove shading effects, they allow for more accurate material classification by focusing purely on color information. For instance, a red albedo might indicate brick, while a beige or white albedo could correspond to plaster or painted surfaces.

(b) **Roughness:** This map controls the way light interacts with a surface by defining how smooth or rough it is at a microscopic level. A low roughness value results in a smooth, glossy surface that produces sharp and well-defined reflections, whereas a high roughness value leads to a diffuse, matte appearance with scattered reflections. Since roughness maps dictate the spread of light rather than its intensity, they are essential in distinguishing materials like polished glass from frosted glass or fresh paint from aged wood. By analyzing roughness variations, one can infer surface treatments, wear, or environmental effects on a material.

(c) **Metal:** This map differentiates between metallic and non-metallic surfaces, fundamentally altering how light interacts with the material. Metallic surfaces primarily reflect light while absorbing their base color, creating highly specular reflections, whereas non-metals exhibit more diffuse reflection with less pronounced highlights. A metalness map is typically binary (black for dielectrics, white for metals), ensuring distinct material classifications such as gold, steel, or copper versus plastic, fabric, or stone. By analyzing metalness properties, one can determine whether a material behaves like a conductor or an insulator in terms of light reflection and energy absorption.

5. **Blind Agent:** This agent analyzes the question to extract relevant knowledge, aiding other agents in task resolution by providing relevant information.

C Additional Implementation Details

Our system operates using a structured, iterative reasoning process. At each iteration, worker agents evaluate their confidence in their outputs using a discretized thresholding system. Instead of continuous confidence scores, agents classify their confidence levels into five predefined categories: {Very Strongly, Strongly, Neutral, Weakly, Very Weakly}

To facilitate seamless interaction and data exchange, agents communicate using a structured JSON format. This representation allows for efficient storage, retrieval, and processing of information, ensuring that each agent can interpret and respond to data effectively.

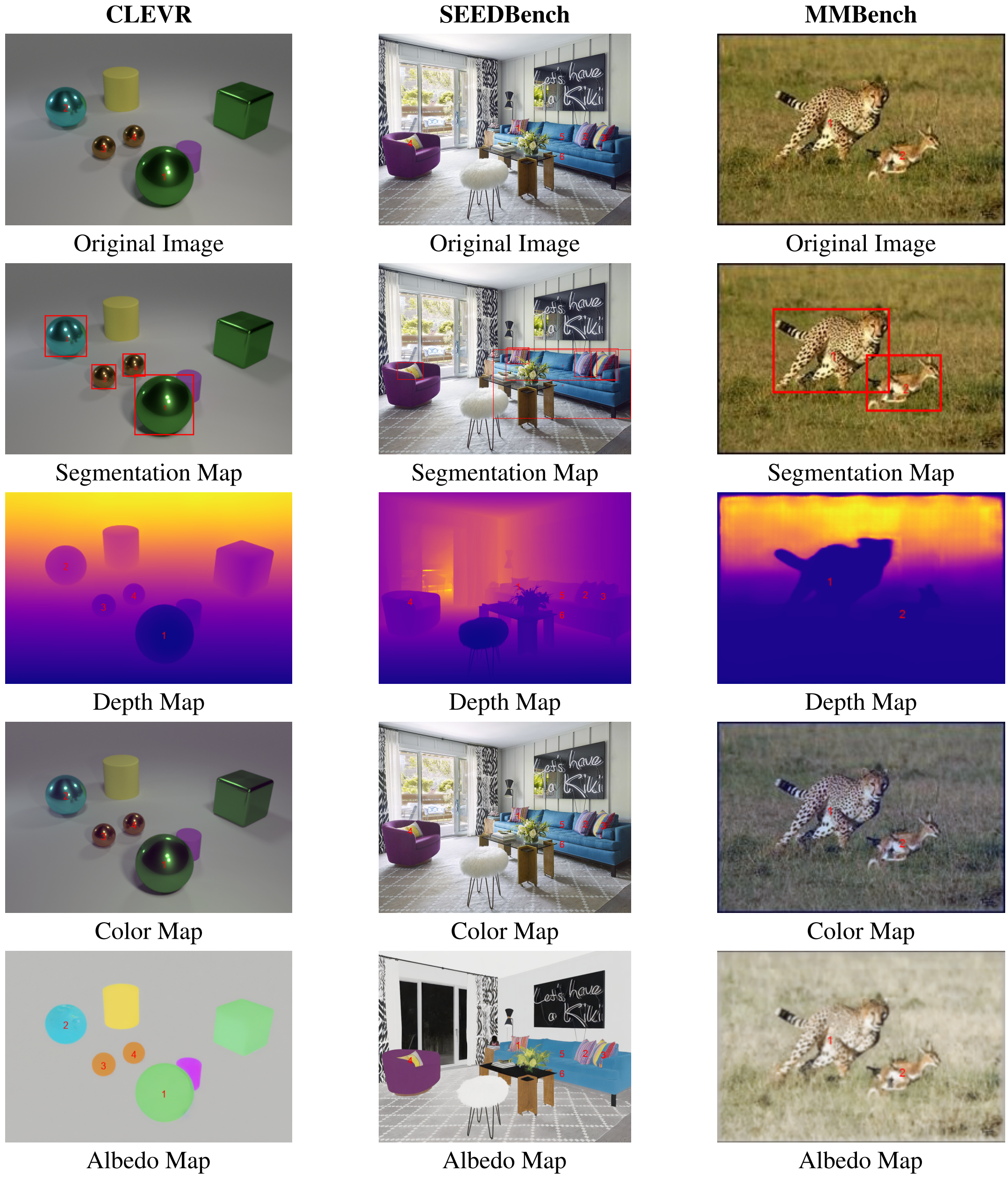
Our approach focuses on explicit decomposition of images into interpretable visual sub-domains at test time. Complementary strategies for improving MLLM performance include attention-guided image manipulation techniques that reallocate spatial resolution based on cross-modal attention patterns (Dalal et al., 2025b).

D Details of the qualitative analysis

D.1 Error Categorization

As mentioned in Sec. 4.3, we categorize the error types observed in results from GPT-4V with and without our framework into 6 categories. Details about each category and their examples are as follows:

1. **Occlusion**: In these cases, the image consists of multiple objects where one object partially obscures another object whose properties are important for answering the question. For example, if a sphere covers a cube such that only part of the cube is visible, the MLLM cannot properly extract the relevant information, resulting in incorrect answers.



Original Image | Original Image | Original Image

Segmentation Map | Segmentation Map | Segmentation Map

Depth Map | Depth Map | Depth Map

Color Map | Color Map | Color Map

Albedo Map | Albedo Map | Albedo Map

Figure 8: Comparison of original images and their corresponding segmentation, depth, color, and albedo maps for CLEVR, SEEDBench, and MMBench. Each column represents one dataset, and rows show the various transformation types.

2. **Comparative Reasoning**: Comparative reasoning failures occur when the model struggles to correctly analyze and compare attributes between two or more objects. These errors indicate difficulties in making relative judgments, such as which object is larger, which is brighter, or which is closer to another. Example: If given a question like "Which ball is bigger?" and presented with two spheres of different sizes, the model might misidentify the larger one, failing to establish a correct comparison.
3. **Visual Encoding**: These errors account for missing details that could have been extracted from the image using specialized encoders. For example, by using domain-specific encoders we extract intrinsic information from the image that provides additional insights for solving complex questions.
4. **Lexical Mismatch**: These errors occur when the model's predicted answer is semantically close but not an exact match with the ground truth. These errors reflect limitations in language generation, where minor differences in word choice lead to incorrect classifications. Example: If the expected answer is "pink", but the model predicts "light red", it demonstrates a failure in lexical granularity, despite both answers conveying a similar meaning.
5. **Aggregation**: Aggregation errors occur when the model improperly combines available information to produce an answer. This typically results from flawed reasoning steps, incorrect fusion of multimodal data, or incomplete synthesis of visual and textual cues.
6. **Miscellaneous**: This category includes all uncategorized mistakes that do not fit within the defined error classes. These errors could arise from unexpected biases, dataset artifacts, spurious correlations, or rare out-of-distribution cases.

D.2 Agreement Matrix Computation

As described in Sec. 4.3, we utilize the agreement matrix to check the agreement between REDI and human decomposition into visual sub-domains. The following equation computes the agreement score for each entry of the matrix:

$$\textbf{Agreement Score} = \frac{\sum_{i=1}^{N} \mathcal{I}(H_i = F_i)}{N} \tag{1}$$

where:

- $\boldsymbol{H_i}$ and $\boldsymbol{F_i}$ are the human and REDI labels for a given feature at instance i.
- $\boldsymbol{N}$ is the total number of instances.
- $\mathcal{I}(H_i = F_i)$ is an indicator function that returns 1 if both labels match and 0 otherwise.

D.3 Annotation

As described in Sec. 4.3, we annotated images in two user studies comprising 200 VQA tasks and 100 VQA tasks, respectively. Two annotators independently annotated the images. To quantify agreement, we computed Cohen's kappa (κ) score, which resulted in 0.94, indicating strong consistency between annotators. In cases of disagreement, annotators collaboratively refined ambiguous cases or consulted a third expert when necessary.

E Qualitative Examples

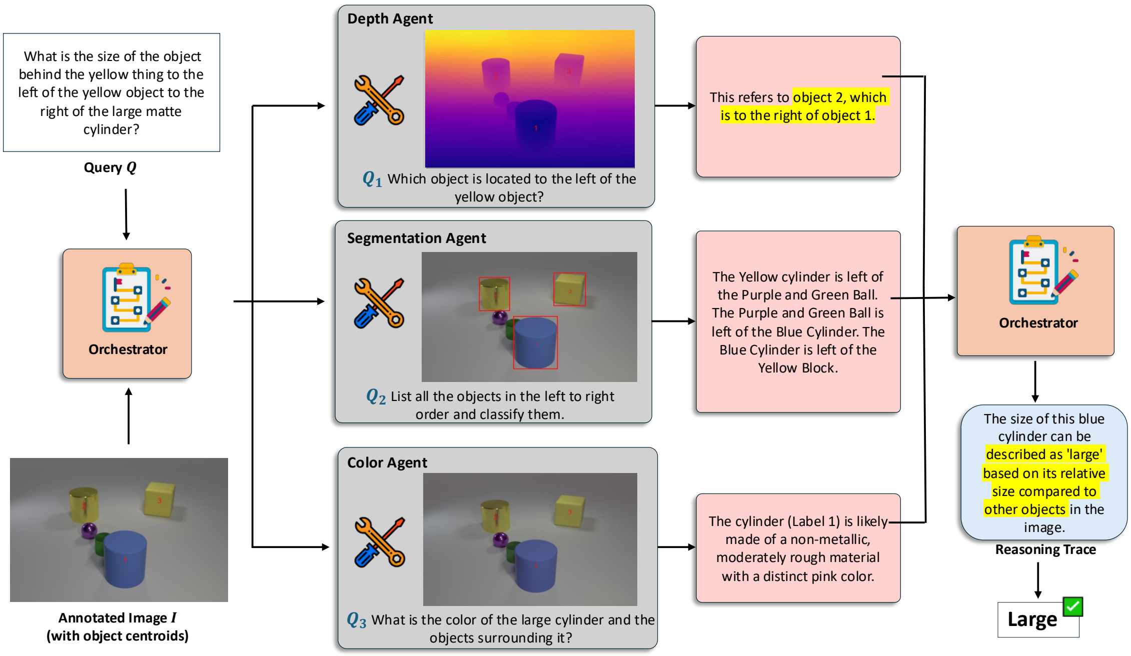

What is the size of the object behind the yellow thing to the left of the yellow object to the right of the large matte cylinder?
Query Q
Orchestrator
Annotated Image I (with object centroids)
Depth Agent
Q_1 Which object is located to the left of the yellow object?
Segmentation Agent
Q_2 List all the objects in the left to right order and classify them.
Color Agent
Q_3 What is the color of the large cylinder and the objects surrounding it?
This refers to object 2, which is to the right of object 1.
The Yellow cylinder is left of the Purple and Green Ball. The Purple and Green Ball is left of the Blue Cylinder. The Blue Cylinder is left of the Yellow Block.
The cylinder (Label 1) is likely made of a non-metallic, moderately rough material with a distinct pink color.
Orchestrator
The size of this blue cylinder can be described as 'large' based on its relative size compared to other objects in the image.
Reasoning Trace
Large ✓

Figure 9: **Example reasoning trace for spatial-size reasoning.** For the query *"What is the size of the object behind the yellow thing to the left of the yellow object to the right of the large matte cylinder?"*, the **Orchestrator** consults the **Depth**, **Segmentation**, and **Color Agents**. The agents identify spatial relations, classify objects, and analyze their material attributes. Integrating these cues, the orchestrator determines that the referenced object is the **blue cylinder**, whose relative size is *large*.

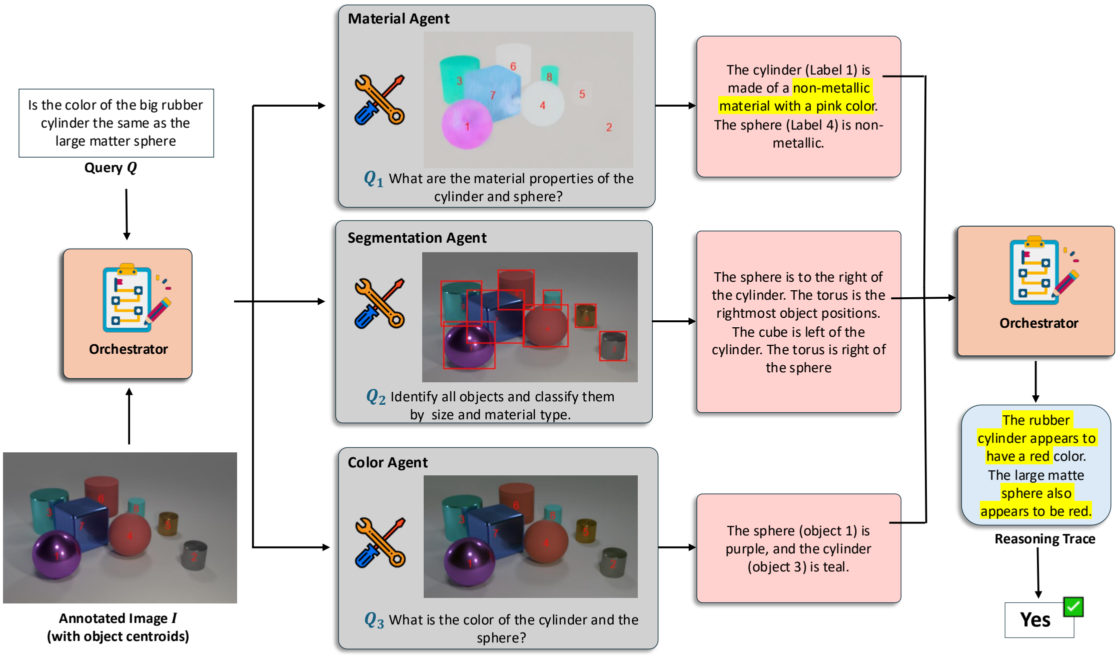

Is the color of the big rubber cylinder the same as the large matter sphere
Query Q
Orchestrator
Annotated Image I
(with object centroids)
Material Agent
Q_1 What are the material properties of the cylinder and sphere?
Segmentation Agent
Q_2 Identify all objects and classify them by size and material type.
Color Agent
Q_3 What is the color of the cylinder and the sphere?
The cylinder (Label 1) is made of a non-metallic material with a pink color. The sphere (Label 4) is non-metallic.
The sphere is to the right of the cylinder. The torus is the rightmost object positions. The cube is left of the cylinder. The torus is right of the sphere
The sphere (object 1) is purple, and the cylinder (object 3) is teal.
Orchestrator
The rubber cylinder appears to have a red color. The large matte sphere also appears to be red.
Reasoning Trace
Yes

Figure 10: **Example reasoning trace for color comparison.** For the query *"Is the color of the big rubber cylinder the same as the large matte sphere?"*, the **Orchestrator** queries the **Material**, **Segmentation**, and **Color Agents**. The agents identify that both the cylinder and sphere are non-metallic and appear red, leading the orchestrator to conclude the answer: **Yes**.

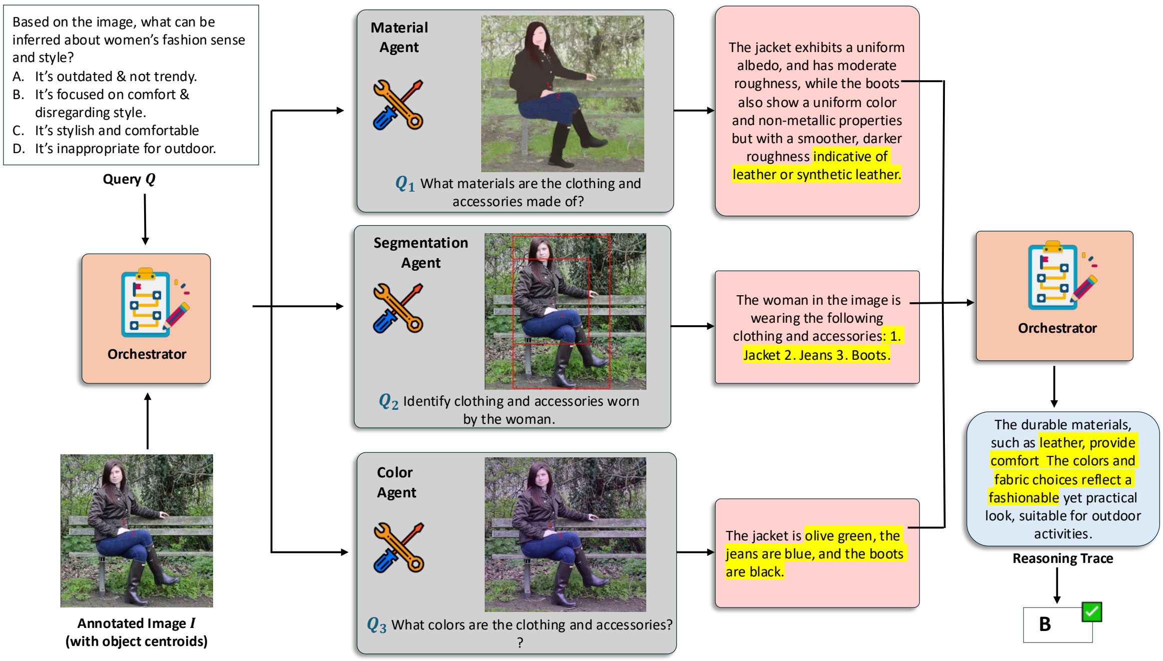
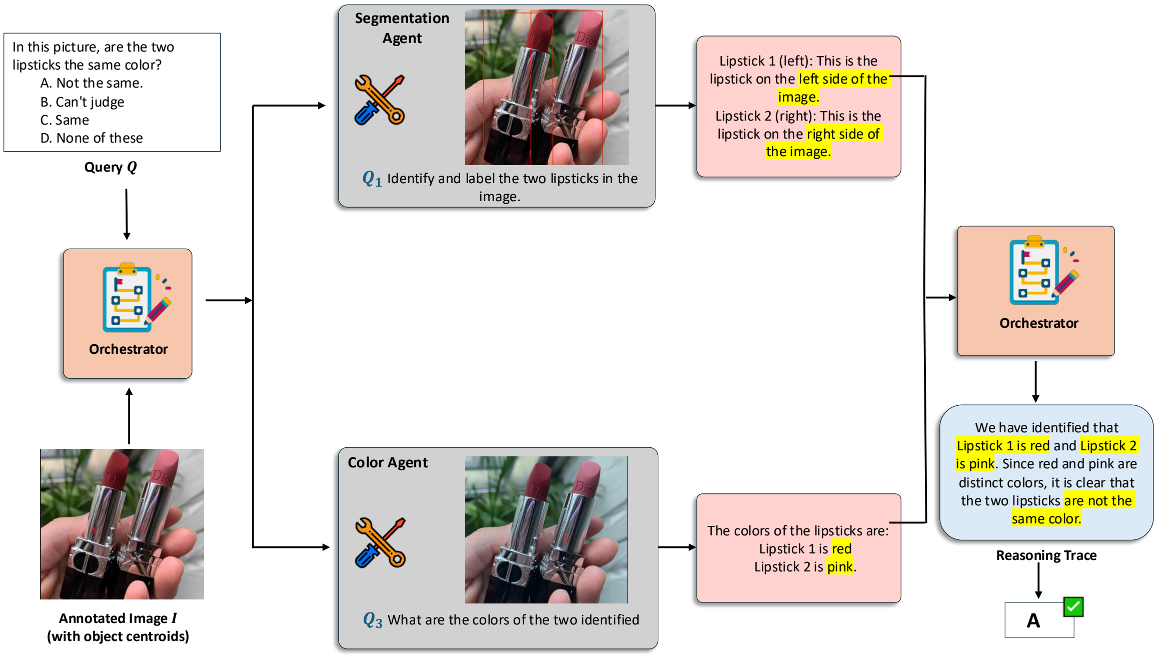

Based on the image, what can be inferred about women's fashion sense and style?
A. It's outdated & not trendy.
B. It's focused on comfort & disregarding style.
C. It's stylish and comfortable
D. It's inappropriate for outdoor.
Query Q
Orchestrator
Annotated Image I
(with object centroids)
Material Agent
Q_1 What materials are the clothing and accessories made of?
Segmentation Agent
Q_2 Identify clothing and accessories worn by the woman.
Color Agent
Q_3 What colors are the clothing and accessories? ?
The jacket exhibits a uniform albedo, and has moderate roughness, while the boots also show a uniform color and non-metallic properties but with a smoother, darker roughness indicative of leather or synthetic leather.
The woman in the image is wearing the following clothing and accessories: 1. Jacket 2. Jeans 3. Boots.
The jacket is olive green, the jeans are blue, and the boots are black.
Orchestrator
The durable materials, such as leather, provide comfort The colors and fabric choices reflect a fashionable yet practical look, suitable for outdoor activities.
Reasoning Trace
B

Figure 11: **Example reasoning trace for fashion-style inference.** For the query *"Based on the image, what can be inferred about women's fashion sense and style?"*, the **Orchestrator** consults the **Material**, **Segmentation**, and **Color Agents**. The agents identify the woman's jacket, jeans, and boots, infer materials such as leather, and detect olive-green, blue, and black colors. Combining these cues, the orchestrator concludes that the outfit emphasizes comfort and practicality, yielding option **B**.



In this picture, are the two lipsticks the same color?

A. Not the same.
B. Can't judge
C. Same
D. None of these

Lipstick 1 (left): This is the lipstick on the left side of the image.
Lipstick 2 (right): This is the lipstick on the right side of the image.

Q_1 Identify and label the two lipsticks in the image.

Q_3 What are the colors of the two identified

The colors of the lipsticks are:
Lipstick 1 is red
Lipstick 2 is pink.

We have identified that Lipstick 1 is red and Lipstick 2 is pink. Since red and pink are distinct colors, it is clear that the two lipsticks are not the same color.

Figure 12: **Example reasoning trace for color differentiation.** For the query *"In this picture, are the two lipsticks the same color?"*, the **Orchestrator** engages the **Segmentation** and **Color Agents**. The agents identify two lipsticks and detect their respective colors as red and pink. Integrating these findings, the orchestrator concludes that the lipsticks are not the same color, yielding answer **A**.

F Prompts

The orchestrator's ability to decompose complex questions into targeted sub-questions for specialized agents draws on principles of question generation and decomposition in visual reasoning (Jain et al., 2017, 2018). Furthermore, the hierarchical coordination of multiple agents to achieve a complex goal relates to sequential planning approaches in vision-language tasks (Patel et al., 2023). Below, we provide the detailed prompts used for each agent in our system.

Orchestrator Agent Prompt ($\mathcal{P}_{\Pi^{\text{orc}}}$)

You are the Orchestrator in a multi-agent system designed to analyze an image and answer user queries. You have 5 expert agents at your disposal: Segmentation, Depth, Color, Material, and Language.

Your role is to:

1. Interpret the user's question and decompose it into subtasks. The user query would be a combination of multiple tasks, so you need to break these into separate tasks so that each subtask can be answered by a specific agent.
2. Pose specific, relevant questions to each agent, ensuring that these questions directly contribute to answering the subtasks you identified.

Note: You don't need to use all the agents, only the ones that are necessary to answer the user query. In some cases, it can be all of them.

Guidelines for calling the agents:

- Use the Segmentation agent to identify objects, count them if needed, determine their left-to-right order for 2D positional understanding, and label them clearly.
- Depth agent for spatial relationships, specifically, forward-backward understanding.
- Query the Color agent for color attributes of identified objects.
- Query the Material agent for material properties.
- Query the Language agent to ask questions that are language-specific and can be answered purely based on the language. The questions should be such that they can help provide more context to solve the original question or answer a sub-part of the original question. Note that the language agent will not have access to the image.

Examine the provided image and question, and output text listing the agents along with their separate question (one question per agent) that should be asked to them so as to answer the question, and also list the objects that should be considered for answering the question.

The text should be in the format:

{
agents : [Agent1 , Agent2 , ...],
sub_tasks : [Subtask1 , Subtask2 , ...],
questions : [Question for Agent1 , Question for Agent2 , ...],
objects : [Object1 , Object2 , ...],
reasoning : Your reasoning here
}.

Make sure that you strongly stick to the JSON format. From the list of available agents, only choose what you think is necessary to answer the question.

Segmentation Agent Prompt ($\mathcal{P}_{\text{seg}}$)

You are the Segmentation Agent responsible for analyzing a segmentation map.

Role and Responsibilities:

1. Identify objects or regions of interest based on the question asked.
2. The objects in the images have a label assigned to it. You need to identify these objects and write the label that is there in the image for each object. These labels are structured ways of representing the objects.
3. Count the number of instances of a particular object class if asked.
4. Analyze the relative positions of identified objects to generate a left-to-right ordering and describe which objects are to the left or right of others.
5. Provide your best guess for the following question, and describe how likely it is that your guess is correct as one of the following expressions: {Very Strongly, Strongly, Neural, Weakly, Very Weakly}. Give ONLY the guess and your confidence, no other words or explanation.

Expected Output:

- A list of identified objects or categories, with each object labeled uniquely.
- A left-to-right ordered list of objects based on their positions in the segmentation map, specifying which objects are leftmost and rightmost.
- Counts of objects if requested.
- Include only information that has been explicitly asked for.

Directives:

1. Provide consistent naming or indexing for each object.
2. If counting is requested, accurately report the number of objects that match the given criteria.
3. Report the left-to-right order based on spatial positions in the segmentation map, identifying leftmost and rightmost objects and relative positions.

Depth Agent Prompt ($\mathcal{P}_{\text{depth}}$)

You are the Depth Agent responsible for analyzing the depth map.

Role and Responsibilities:

1. Determine relative distances between the objects.
2. Provide information on which objects are closer or farther from the camera.
3. Give your answer in terms of the object label. All objects have a label assigned to them in the image.
4. If asked, give qualitative spatial relationships based on depth values.
5. Provide your best guess for the following question, and describe how likely it is that your guess is correct as one of the following expressions: {Very Strongly, Strongly, Neural, Weakly, Very Weakly}. Give ONLY the guess and your confidence, no other words or explanation.

How to Read the Depth Map:

- The depth map uses the plasma colormap, where the objects closer to the camera are in blue and objects farther from the camera are in yellow.
- For more fine-grained depth analysis focus on the shades of colors, for example dark blue is closer than light blue.

Expected Output:

- Relative depth orderings (e.g., "Object A is closer than Object B"). Also, provide your confidence score.

Directives:

1. Use object labels as input references.
2. Clearly state which objects are nearer or farther relative to each other.
3. Stick to the requested objects and avoid unrelated depth details.
4. Do not speculate on properties other than relative depth information.

Color Agent Prompt ($\mathcal{P}_{\text{color}}$)

You are the Color Agent responsible for analyzing the Color of the input image.
Role and Responsibilities:
1. Identify the colors of the objects specified.
2. Give your answer in terms of the object label. All objects have a label assigned to them in the image.
3. Provide your best guess for the following question, and describe how likely it is that your guess is correct as one of the following expressions: {Very Strongly, Strongly, Neural, Weakly, Very Weakly}. Give ONLY the guess and your confidence, no other words or explanation.

Expected Output:
- A clear color description of each requested object (e.g., "The object is red") and the confidence score. Don't use adjectives with the colors. Just mention the color.
- Avoid mentioning material or spatial properties. Stick to color only.

Material Agent Prompt ($\mathcal{P}_{\text{mat}}$)

You are the Material Agent responsible for analyzing the Albedo, Metal, and Roughness Map.
Role and Responsibilities:
1. Analyze the Albedo, Metal, and Roughness map to determine the material properties of the objects.
2. Provide your best guess for the following question, and describe how likely it is that your guess is correct as one of the following expressions: {Very Strongly, Strongly, Neural, Weakly, Very Weakly}. Give ONLY the guess and your confidence, no other words or explanation.

How to read these maps:
- Roughness: Darker regions are smoother, reflecting light like glass or metal. Lighter regions are rougher, scattering light and appearing matte.
- Metal: Regions more towards the white side indicate metal, while towards black indicate non-metal. Dark gray regions are considered metal, and Light gray regions as not mental.
- Albedo: The Albedo map provides the base color of the object, which is unaffected by lighting or shading. Metallic Surfaces: Often have uniform or darker albedo since reflections dominate. Transparent Materials: Often have washed-out or pale colors, suggesting glass or acrylic.
- Give your answer in terms of the object label. All objects have a label assigned to them in the image.

Expected Output: - The expected material of the object and the confidence score. No other words or explanation.
Directives:
1. Avoid discussing color or depth. Stick strictly to Material properties.
2. Do not speculate beyond the data in the Materiallic map.

Blind Agent Prompt ($\mathcal{P}_{\text{lng}}$)

You are a Language Expert, and you will be given a question or a request, and you have to provide the answer to that question or fulfill the request. If you aren't able to answer the question, just say, "I don't know." Don't guess the answer.

Orchestrator Agent Aggregation Prompt ($\mathcal{P}_{\text{agg}}$)

You are the Orchestrator in the aggregation stage of a multi-agent system. Your role is to reason on the response collected from expert agents (Segmentation, depth, color, material, and language). Also, use your understanding and insights to answer the user's query.
If all the agents or some agents provide incomplete or conflicting information, you should use your reasoning to select the best possible answer.
You have access to the image and the question; hence, before giving the final answer, also check if the answer sounds right based on the image.
Your answer should only use the JSON format shown below.
Ensure the response follows strict JSON formatting.
Provide your final response will look like the JSON format:
{
"reasoning": "Explanation for the answer.",
"answer": "A clear one-word answer to the user's query, derived from the combined data of all agents and your understanding and reasoning."
}