Perspective-Aware Reasoning in Vision-Language Models via Mental Imagery Simulation

Phillip Y. Lee[1], Jihyeon Je[2], Chanho Park[1], Mikaela Angelina Uy[3]
Leonidas Guibas[2], Minhyuk Sung[1]
[1]KAIST, [2]Stanford University, [3]NVIDIA
{phillip0701, mhsung}@kaist.ac.kr

Figure 1: We introduce **Abstract Perspective Change** (APC), a framework that empowers VLMs to adopt arbitrary perspectives for spatial reasoning. As demonstrated by the examples above, APC significantly enhances VLM's ability to *imagine a scene from alternative viewpoints*, overcoming the inherent egocentric bias that constrains the spatial reasoning of existing VLMs to the camera's viewpoint.

Abstract: We present a framework for perspective-aware reasoning in vision-language models (VLMs) through mental imagery simulation. Perspective-taking—the ability to perceive an environment or situation from an alternative viewpoint—is a key benchmark for human-level visual understanding, essential for environmental interaction and collaboration with autonomous agents. Despite advancements in spatial reasoning within VLMs, recent research has shown that modern VLMs significantly lack perspective-aware reasoning capabilities and exhibit a strong bias toward egocentric interpretations. To bridge the gap between VLMs and human perception, we focus on the role of mental imagery, where humans perceive the world through abstracted representations that facilitate perspective shifts. Motivated by this, we propose a framework for perspective-aware reasoning, named Abstract Perspective Change (APC), that effectively leverages vision foundation models, such as object detection, segmentation, and orientation estimation, to construct scene abstractions and enable perspective changes. Our experiments on synthetic and real-image benchmarks, compared with various VLMs, demonstrate significant improvements in perspective-aware reasoning with our framework, further outperforming fine-tuned spatial reasoning models and novel-view-synthesis-based approaches.

Keywords: Vision-Language Models, Spatial Reasoning

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

1 Introduction

Vision-language models (VLMs) have made remarkable progress, positioning themselves as a crucial backbone for general-purpose physical AI agents. The growing research efforts to improve VLMs' spatial reasoning capabilities [1, 2, 3, 4] reflect this potential. Early VLMs were limited to basic tasks such as visual question answering (VQA) and image captioning [5, 6, 7]. However, recent advancements have enabled them to perform complex visual reasoning [8, 9, 10, 11, 12, 13] and extract spatial properties, including spatial relationships, relative sizes, and distances [14, 1]. Further techniques such as instruction-tuning and vision-centric adapters, have expanded their capabilities, allowing depth-aware [3, 15] and region-aware [2, 16] spatial reasoning.

Despite these advances, progress remains largely confined to *egocentric* spatial reasoning, and even the latest VLMs struggle with *allocentric* reasoning—answering questions from perspectives other than the camera's (Fig. 2). Allocentric reasoning is crucial for high-level planning, environmental interaction, and collaboration with autonomous agents [17, 18, 19]. Moreover, it serves as a key benchmark for human-level spatial understanding. However, as analyzed by Zhang *et al.* [20], most VLMs exhibit a strong bias toward an *egocentric* perspective. Even when explicitly prompted to adopt an *allocentric* viewpoint, VLMs often revert to egocentric interpretations [21, 22, 20, 23]. Recent efforts to enhance spatial reasoning remain focused on improving egocentric reasoning [3, 2, 4], leaving allocentric reasoning largely unaddressed.

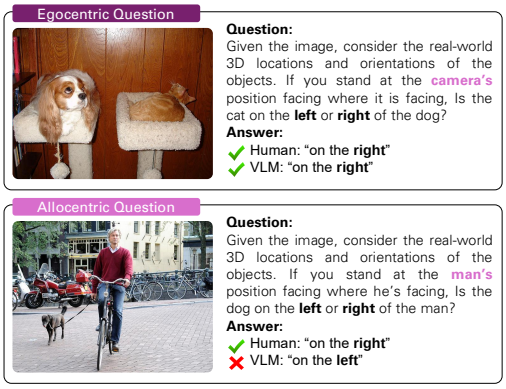

Figure 2: **Egocentric vs. Allocentric.** While VLMs perform well when questions are asked from an egocentric (i.e. camera's) perspective, they struggle when the same questions are posed from an allocentric perspective, showing a strong bias toward egocentric reasoning.

To bridge the gap between VLMs and human perspective reasoning, we ask: *What cognitive process allows humans to effortlessly shift perspectives?* Unlike current VLMs, humans seamlessly form internal representations of the physical world, making perspective reasoning an intuitive and natural process. The mechanism of creating internal representations, known as *mental imagery* [24, 25, 26, 27, 28], plays a fundamental role in cognition, enabling us to simulate visual, spatial, and conceptual scenarios. This ability allows for abstraction beyond immediate perception, facilitating sophisticated spatial reasoning tasks such as mentally rotating objects, inferring occlusions, and envisioning alternative viewpoints [29, 30, 31].

A key aspect of mental imagery is that it is not simply the process of visualizing a clear image from different perspectives; rather, it involves forming an *abstract representation* of a scene that encodes essential spatial information and can be reinterpreted from a new perspective. From a computational standpoint, such an abstract representation is particularly advantageous, as equipping VLMs with the imaginative capability to generate novel views remains extremely challenging. In contrast, constructing an abstract representation requires significantly less computation and can be achieved procedurally.

Inspired by this, we introduce a novel framework for adapting perspectives in VLMs by simulating the mental imagery process and modifying the perspective in the given prompt. Our goal is to leverage the strengths of both VLMs and recent vision foundation models, such as object detection [32, 33, 34], image segmentation [35, 36], and orientation estimation [37]. The proposed framework takes an image and a perspective-based question as input and operates through three key stages. First, by simulating the mental imagery process, it builds an abstract representation of the scene in the input image. The VLM parses the prompt to identify objects in the image, while vision foundation models extract the center and orientation information of each object in 3D space. Second, the VLM analyzes the prompt to determine the reference object from whose perspective the question is asked, and transforms the abstraction to be aligned with that perspective. Finally,

a new *prompt* is generated by reinterpreting the scene from the reference object's perspective. We explore different formats for rendering the scene abstraction: (1) a text-based representation, where objects are described using numerical 3D coordinates, and (2) an image-based representation, where objects are visualized as colored boxes corresponding to the original image. The newly generated prompt is fed to the VLM to obtain the final answer. Note that our framework is *not* designed for a specific type of allocentric question. We leverage the egocentric reasoning capabilities of VLMs by performing allocentric-to-egocentric prompt conversion through scene abstraction, removing the perspective-related barrier in the question while preserving its original intent in the new prompt.

In our experiments on COMFORT++ [20] and 3DSRBench [38], our method achieves robust spatial reasoning across a variety of tasks and perspectives. In constrast, baseline VLMs and previous frameworks for spatial reasoning often struggle with even simple viewpoint shifts, reconfirming a notable bias toward the camera's perspective. These results highlight how our abstraction-based representation significantly enhances the spatial reasoning capabilities of VLMs beyond their default egocentric perspectives.

2 Related Work

2.1 Spatial Reasoning with VLMs

Building on the remarkable advancements of vision-language models (VLMs) [10, 39, 14, 11, 12], recent studies have adapted VLMs for real-world spatial reasoning. Numerous evaluations revealed that VLMs struggle on even elementary spatial-perception tasks [40, 41, 42, 43, 44] and higher-level spatial reasoning based on images or videos [45, 46, 47, 48, 49, 50]. SpatialVLM [3] tackles this issue with a data-synthesis pipeline that injects rich spatial cues, while Cambrian-1 [1] introduces an architecture purposed for improved spatial reasoning. Another line of work allows VLMs to utilize richer vision-centric data such as points, depth maps or segmentation masks through fine-tuning [51, 15, 52, 48] or employing auxiliary encoders [2]. Taking a different approach, other works exploit the planning and programming abilities of language models, building LLM/VLM-in-the-loop systems that call external vision modules as needed [53, 54, 55]. Notably, SpatialPIN [4] extracts dense visual priors from multiple vision foundation models [56, 57] and uses a VLM [8] to combine and interpret this information.

2.2 Visual Perspective-Taking

Visual perspective-taking (VPT) is the ability to imagine an alternative viewpoint, whether from another person's perspective or a different camera angle. This ability is essential for fundamental human skills such as navigation, spatial awareness, and social interaction [58, 25, 59, 60]. To be regarded as a general vision agent capable of human-like reasoning, a VLM should possess robust perspective-taking abilities. However, recent analyses reveal that current VLMs fail to shift to allocentric perspectives, showing a strong bias toward the egocentric viewpoint of a given image [20, 38, 23, 21, 22]. Zhang *et al.* [20] propose a synthetic evaluation protocol to assess whether VLMs can adopt different frames of reference (i.e. perspective). Likewise, 3DSRBench [38] includes real image-question pairs asked from an object's viewpoint, and finds that recent VLMs still demonstrate near chance level on perspective-related tasks. These findings suggest that while VLMs are rapidly improving in both complex visual reasoning [8, 9, 10, 11, 12] and basic spatial reasoning [14, 1, 3, 15, 2, 16], their abilities remain confined to the egocentric viewpoint, posing a significant barrier to human-like reasoning. Recently, SAT [61] proposed to improve VLMs' allocentric reasoning through instruction-tuning, yet it remains restricted to left/right relations with the need for annotations. In this work, we empower VLMs to reason from *arbitrary* perspectives, by reformulating any spatial reasoning task into their default egocentric viewpoint, resulting in a generalizable framework.

2.3 Visual Prompting

Visual prompting frames an input image as an *instruction* for a VLM, functioning similarly to how text prompts guide language models [62, 63, 64, 65, 66, 67, 68]. Numerous studies have demonstrated its effectiveness by exploiting the inherent image comprehension capabilities of VLMs.

Set-of-Marks [69] augments each object in an image with its corresponding segmentation mask for more fine-grained visual grounding. Visual Sketchpad [70] provides tool-based framework for VLMs to utilize drawing tools to annotate images for complex tasks such as math problem solving and visual search. Recent research further proposes *visual* chain-of-thought (CoT) pipelines [71, 72, 67, 73, 74, 75, 76] that visualize intermediate reasoning steps as images and feed them back to the model as auxiliary inputs. This visual feedback loop has proven effective for spatial tasks, as it anchors textual reasoning to concrete visual cues [71, 73]. Building on this idea, we propose to transform an abstraction of a given scene and feed it back to a VLM in the form of a visual prompt, offering a new way for the model to reason from arbitrary viewpoints.

3 Method: Abstract Perspective Change

Our goal is to enable VLMs to solve spatial reasoning tasks from any given perspective (Fig. 2). Let us call the entity of the target perspective as the *reference viewer*. Since VLMs inherently approach spatial reasoning from an egocentric perspective [20], we propose to reformulate perspective-specific questions to align with the reference viewer's egocentric perspective. Inspired by theories in mental imagery [25, 24, 28], we begin by explicitly building an *abstraction* of the scene and use it as a foundation for shifting perspectives (Fig. 3).

Overview of APC. We call our approach Abstract Perspective Change (APC), which consists of three main stages. (1) First, APC constructs a coarse 3D abstraction of the scene from the input image by selecting and extracting objects of interest using off-the-shelf vision modules (Sec. 3.1), drawing inspiration from human mental imagery [25]. (2) Next, APC selects a reference viewer for the spatial reasoning task among the objects of interests in the constructed scene abstraction. This determines "where to look from". Such a formulation allows the conversion of the allocentric reasoning problem to an egocentric spatial reasoning task by performing a *perspective change* that transforms the base coordinate system of the abstraction from the original camera view to that of the reference viewer (Sec. 3.2). (3) Finally, the transformed abstracted scene, which can now be posed as an egocentric problem, is fed back into the VLM for spatial reasoning (Sec. 3.3). We explore two alternative representations when providing the VLM with transformed astract scene information: 1) directly feeding numerical 3D coordinates of each object as a text prompt (numerical prompt), and 2) generating an abstract rendering of the scene as viewed by the reference perspective (visual prompt). An illustration of our APC pipeline is shown in Fig. 4, and we detail each step as follows.

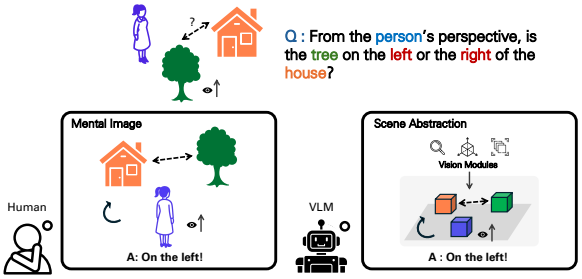

Figure 3: **Mental Imagery Simulation.** Inspired by how humans employ mental imagery to reason across different perspectives (left), we propose a similar process for VLMs, by constructing an explicit abstraction of the input scene and using it as a foundation for perspective changes (right).

3.1 Scene Abstraction

APC begins by building a coarse 3D abstraction of the scene. Given an image I and a spatial reasoning question Q, we define the abstraction of a scene as the set $S_E := \{O_i\}_{i=1}^n$ composed of objects of interest from the question Q. Here, E denotes that the abstraction is defined in the camera's egocentric coordinate system, and the number of objects of interest n is determined by the VLM based on Q. Each O_i corresponds to an object of interest in the image and is represented as a tuple (t_i, c_i, p_i), where t_i is the object's description, $c_i \in \mathbb{R}^3$ is its 3D position, and $p_i \in \mathbb{S}^3$ is a unit vector that indicates its orientation. Additionally, the camera is also included as an object of interest. This abstraction provides a minimal yet sufficient information in order to perform perspective changes, and mirrors how humans draw and rotate mental images when reasoning with perspectives [28, 25]. It allows for our APC to convert an allocentric problem to an egocentric spatial reasoning task, which VLMs can better solve [20]. More details are described below.

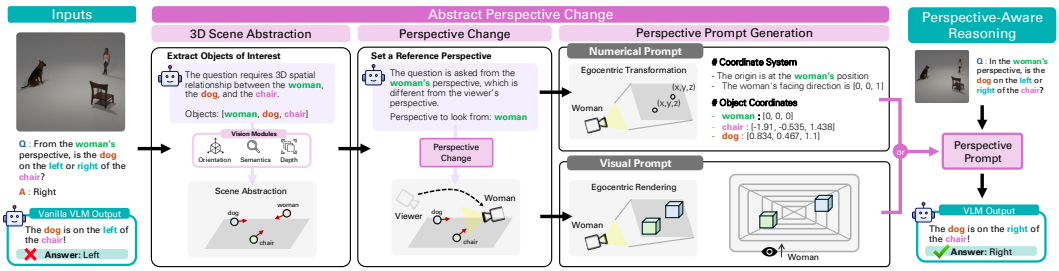

Figure 4: **Pipeline Overview of APC.** Our proposed framework consists of three stages. 1) Scene Abstraction (Sec. 3.1): APC first detects the objects of interest and build a coarse 3D abstraction of the scene using off-the-shelf vision foundation models. 2) Perspective Change (Sec. 3.2): Then, a reference perspective is set and the abstraction is transformed into the reference viewer's egocentric coordinate frame. 3) Perspective Prompting (Sec. 3.3): Finally, APC passes the transformed scene to the VLM by producing (1) a numerical (textual) prompt or (2) an abstract visual prompt, and poses the question of interest from the reference perspective.

Extracting Objects of Interest. To determine which objects in the image should be included in the scene abstraction S_E, we provide the image I and the question Q to the VLM and instruct it to identify the list of objects necessary for answering the question. The VLM then returns the list of objects of interest, specified by their name, which we denote as t_i. The detailed instruction prompts are included in the **Appendix (Sec. E)**.

Building Object Abstractions. Given the list of objects of interest, we complete our abstracted scene representation by extracting the position and orientation of each object O_i using off-the-shelf vision foundation models. To obtain the 3D position of O_i, we first query GroundingDINO [32] with image I and the object description t_i and obtain its 2D bounding box b_i. We then crop I with b_i, and utilize SAM [77] to obtain a precise segmentation mask for O_i. Next, we extract the metric depth map of I using DepthPro [78] and unproject the pixels within the segmentation mask to 3D. Subsequently, the position c_i is obtained by taking the median coordinate of this 3D point cloud. For further implementation details, please refer to the **Appendix (Sec. C)**. Estimating the orientation p_i for each object O_i is also necessary to perform the desired perspective transformation. We utilize OrientAnything [37], which returns the object's frontal orientation within the camera coordinate system. For this, we crop the image with b_i and feed the cropped image to OrientAnything to obtain O_i's orientation, hence completing our scene abstraction representation.

3.2 Perspective Change

With egocentric scene abstraction S_E for a given image I and question Q, APC then determines the reference viewer and performs perspective change to obtain a transformed scene abstraction from the reference viewer's perspective. This effectively converts an allocentric problem into an egocentric task, which VLMs find easier to handle.

Setting a Reference Perspective. APC first determines "where to look from" by selecting a reference viewer from the set of objects of interest. For this, we provide the spatial reasoning question Q to the VLM and instruct it to identify the reference perspective from which the question should be answered. We denote the extracted reference perspective as A, and provide the complete instruction for perspective extraction in the **Appendix (Sec. E)**.

Transforming Scene Abstraction. After identifying the reference viewer, we then transform the original camera-based scene abstraction S_E into the reference viewer's egocentric coordinate system. Specifically, we apply coordinate transformation from the camera's frame to that of the reference viewer A. In the resulting abstraction S_A, the reference viewer A is placed at the origin, and its orientation is aligned with the z-axis. This step supports APC's main objective of reframing a general perspective question—typically an allocentric problem—into the reference viewer's egocentric viewpoint, making it an egocentric task. Finally, we provide S_A to the VLM so it can answer the question Q from A's perspective. We describe this stage more in depth below.

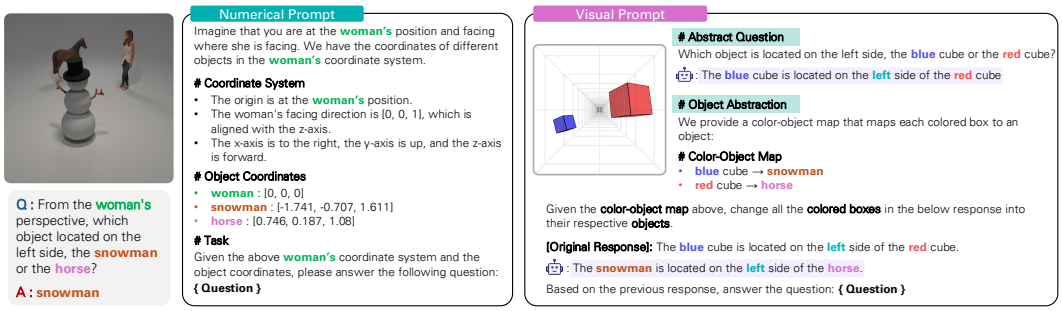

Figure 5: **Perspective Prompt Samples.** We explore two variations of perspective prompting, numerical (left) and visual (right). Numerical (textual) prompt is generated by directly utilizing the 3D coordinate and orientation information. To generate the Visual prompt, we first place a colored cube at each object's identified 3D position then render the scene at the reference viewpoint, which results in an egocentric depiction of the scene. In addition, we construct an abstract question along with object-color mapping to ground the abstracted view.

3.3 Perspective Prompting

The final step of APC involves generating a prompt from the transformed scene abstraction S_A to feed as input for the VLM. That is, how is the VLM asked with the transformed, now egocentric spatial reasoning task? We refer to our generated prompt as the *perspective prompt* for image I and question Q. Since VLMs can take images and text inputs, we explore two choices for the representation of this prompt: numerical (textual) and visual.

Numerical (Textual) Prompt. Recall that an object abstraction in the transformed scene abstraction S_A consists of the object's textual description, its corresponding 3D position, and its orientation, i.e. $O'_i = (t_i, c'_i, p'_i)$. Hence, a straightforward approach is to directly feed this information into the VLM. Specifically, we include the 3D position c'_i in a predefined instruction template and instruct the VLM to directly solve the question Q.

Visual Prompt. Our goal is to let VLMs "view the scene from A's perspective"; thus an alternative choice for the perspective prompt is a visualization of our abstraction S_A. We begin by assigning each object an equal-sized cube, with each cube's position matching the objects' positions c'_i. We then render these cubes from the reference viewer A's vantage point, generating an egocentric depiction of the scene abstraction. To distinguish between objects, each cube is assigned a unique color. When providing this information to the VLM, we modify the original question Q to reflect the abstract visual representation. Specifically, we replace object names (e.g. *"dog"*) with their corresponding colored cubes (e.g. *"red box"*), forming an abstract question Q^*. Refer to Fig. 5 for an example of an obtained abstraction question. Putting it all together, the VLM receives as a prompt both the abstract rendered image—showing colored cubes—and the reformulated question Q^*. This allows it to answer spatial reasoning questions originally posed from *arbitrary* perspectives by reasoning with this abstract, egocentric visual prompt. More details on the visual prompting process are presented in the **Appendix (Sec. C.3)**.

4 Results

In this section, we present the experimental results of our APC across a range of spatial reasoning tasks that include specified reference perspectives. We compare APC to multiple baseline methods and show how our abstraction-based allocentric-to-egocentric reasoning framework enables the VLM to handle alternative perspectives. We use Qwen2.5-VL [11] as our backbone VLM.

4.1 Evaluation Settings

Benchmarks. We validate our APC on both synthetic [20] and real-world [38] benchmarks in which the spatial reasoning requires perspective changes. Sample image-question pairs from each benchmark are shown in Fig. 6.

- **COMFORT++:** Zhang *et al.* [20] introduce COMFORT, a benchmark synthesis protocol designed to evaluate VLMs on perspective-aware spatial reasoning. It employs a simple Blender [79] rendering pipeline to place multiple objects in a synthetic scene with one reference viewer and various other objects. Each scene poses a spatial reasoning question from the reference viewer's perspective. Building on COMFORT, we construct four types of spatial reasoning tasks that require a reference viewer different from the camera: *left/right*, *closer/further*, *visibility*, and *facing*.
- **3DSRBench:** Ma *et al.* [38] introduce a 3D spatial reasoning benchmark based on MS-COCO images [80]. We focus on three categories that require an allocentric viewpoint: *left/right*, *visibility*, and *facing*. Note that we recast the original *front/behind* question in 3DSRBench into a *visibility* question using the same images. We provide further discussion on the dataset and the evaluation protocol in the **Appendix (Sec. D)**.

Baselines: VLMs. We benchmark our APC against multiple state-of-the-art VLMs, including both open-source and proprietary models. For open-source, we include LLaVA-NeXT [39], LLaVA-OneVision [14], Molmo [12], and Qwen2.5-VL [11]. We also include proprietary models: GPT-4o [8] and Gemini-2.0-Flash [9]. We refer to these as *pure VLMs*. Additionally, we compare against *grounded VLMs*, which include models explicitly tuned for spatial reasoning, such as SpatialVLM [3] and SpatialRGPT [2]. We also include SpatialPIN [4], which leverages interactions between VLMs and vision foundation models for complex spatial reasoning.

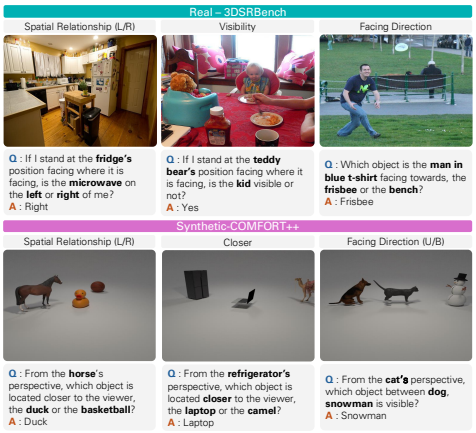

Figure 6: Example image-question pairs from 3DSRBench [38] and COMFORT++ [20] benchmarks. The tasks probe spatial reasoning across left-right relations, object visibility, closeness, and the facing direction of objects.

Baselines: Dense Reconstruction. To compare APC with standard dense reconstruction techniques for novel view synthesis, we introduce two baselines. First, we extend SpatialPIN [4] to include our perspective change phase (Sec. 3.2). We use the generated meshes from its original pipeline and render the meshes from the reference perspective, and denote this extension as SpatialPIN*. Refer to the **Appendix (Sec. B)** for more details. Second, we adopt ViewCrafter [81], a novel view synthesis method designed for single-image inputs. For both baselines, we synthesize a novel view according to the reference perspective's relative pose, and feed the resulting image to the VLM for spatial reasoning.

4.2 Evaluation on COMFORT++ [20]

Tab. 1 (cols 2-5) provides quantitative comparisons on COMFORT++. Here, APC-Vis refers to our visual prompt, and APC-Num corresponds to the numerical prompt. Even though the benchmark consists of objects rendered in a simple, synthetic scene (see Fig. 6), we find that most pure VLMs (rows 3-9) struggle with the *left/right* task, hovering around chance level with the best performing model LLaVA-OneVision scoring only 55.33%. This confirms earlier observations [20] that VLMs fail to adopt alternative perspectives. Even specialist VLMs designed for spatial reasoning perform poorly, with SpatialVLM at 46.0% and both SpatialRGPT and SpatialPIN also exhibiting low accuracy. We observed that SpatialRGPT often generates hallucinated responses unrelated to the instruction, thereby resulting in low accuracy. While SpatialPIN*—employing perspective change—shows better performance, low-quality meshes often bottleneck further improvements (refer to **Sec. B** for more discussions). In contrast, our APC significantly outperforms these baselines, achieving 89.67% accuracy with a visual prompt and 88.67% with a numerical (textual) prompt.

For the *closer* task, some VLMs show relatively high accuracy (79.00% for both LLaVA-OneVision and Cambrian-1), likely since they can also solve the question by comparing object distances directly from the egocentric viewpoint. Even in this case, APC achieves higher accuracy, attaining 96%

Method	COMFORT++ [20]				3DSRBench [38]		
	left/right	closer	visibility	facing	left/right	visibility	facing
Random	50.00	50.00	50.00	50.00	50.00	50.00	50.00
LLaVA-NeXT [39]	48.00	47.33	40.00	39.00	34.10	41.57	50.29
LLaVA-OneVision [14]	55.33	79.00	50.94	38.33	32.09	46.51	60.12
Molmo [12]	36.33	35.67	31.88	29.00	19.77	22.97	32.08
Qwen2.5-VL [11]	43.33	74.33	51.25	43.00	34.96	45.06	53.47
Cambrian-1 [1]	52.00	79.00	57.50	41.00	40.97	49.71	<u>65.03</u>
GPT-4o [8]	41.00	61.33	53.75	38.67	2.01	40.12	47.70
Gemini-2.0-Flash [9]	43.67	26.00	40.31	13.00	24.93	57.65	55.20
SpatialVLM [3]	46.00	41.67	42.81	29.33	22.35	46.51	47.11
SpatialRGPT [2]	27.08	33.90	29.25	1.33	25.98	27.19	42.55
SpatialPIN [4]	19.62	23.96	48.43	43.91	11.10	42.40	11.66
SpatialPIN* [4]	59.80	70.45	49.84	50.51	50.10	52.30	28.86
ViewCrafter [81]	32.33	53.00	38.75	37.46	28.41	22.47	18.31
`APC-Num` **(Ours)**	<u>88.67</u>	**96.00**	<u>71.25</u>	<u>62.00</u>	<u>71.92</u>	<u>62.79</u>	60.98
`APC-Vis` **(Ours)**	**89.67**	<u>94.33</u>	**90.00**	**88.33**	**72.78**	**67.44**	**66.47**

Table 1: **Quantitative Comparisons.** Purple ( ) represents *pure VLMs*, green ( ) represents *grounded VLMs*, and red ( ) represents *dense reconstruction-based* frameworks. Gray ( ) corresponds to our `APC`. **Bold** and <u>underline</u> indicate the best and the second-best result for each column, respectively. `APC-Num` and `APC-Vis` refer to our method employing numerical prompt and visual prompt, respectively.

when using a numerical prompt. Moreover, for *visibility* and *facing* categories, the baseline models perform at near-chance levels, failing to take the reference perspectives into account. Notably, `APC` exhibits a performance gap between visual and numerical prompts, with the visual prompt outperforming the numerical one by +18.75% and +26.33%, respectively. We attribute this difference to trivial logical errors that VLMs often make when relying on numerical coordinates. In contrast, for these two tasks the visual prompt requires only simple visual perception, mitigating such logical errors and achieving more accurate results.

4.3 Evaluation on 3DSRBench [38]

Tab. 1 (cols 6-8) presents quantitative comparisons on 3DSRBench, which includes real images. Compared to the synthetic environment in COMFORT++, using real images introduces additional noise into both the VLMs' visual reasoning and our `APC`'s scene abstraction phase. For the *left/right* task, baseline VLMs consistently fall under 50% accuracy, including the grounded VLMs. Even SpatialPIN* with perspective change only reaches 50.10%, at near chance-level. We find that using ViewCrafter to generate a novel view from the reference perspective yields 28.41% accuracy, due to the noise and hallucinations during generation. We provide visualizations of the rendered views from both SpatialPIN* [4] and ViewCrafter [81] in the **Appendix (Sec. B)**, along with more discussions on the different between our abstraction-based approach and the dense reconstruction-based approaches. Compared to other baselines, `APC` consistently achieves accuracies above 60% both with visual and numerical prompts, showing that our framework is robust to real images.

For *visibility* task, while our method outperforms the baselines, the accuracy appears lower than previous tasks—67.44% with the visual prompt and 62.79% with the numerical prompt. We attribute this decline to the noise in the scene abstraction phase, particularly errors in detected orientations or centroids. This issue is evident when compared to the *visibility* task in COMFORT++, which employs the same question format but with a simpler scene. Lastly, for the *facing* task, Cambrian-1 achieves 64.03%, yet our `APC` with a visual prompt still leads at 66.47%. Qualitative examples on 3DSRBench are shown in Fig. 7.

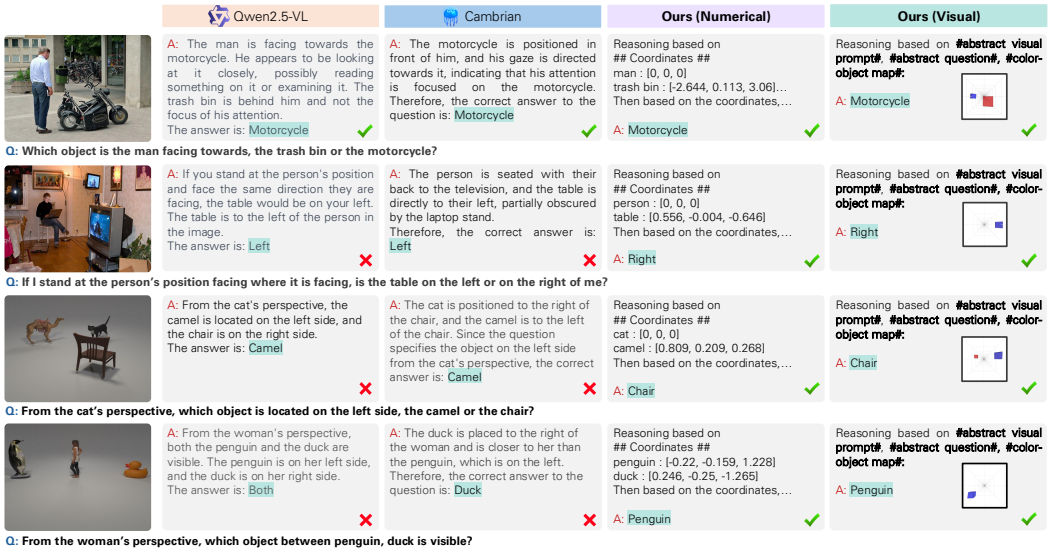

Figure 7: **Spatial Reasoning with Perspective Change.** Recent VLMs such as Qwen2.5-VL [11] and Cambrian-1 [1] often struggle with spatial reasoning tasks that require a shift to a specific reference viewpoint. In constrast, our APC effectively handles such perspective changes by constructing a scene abstraction and delivering the transformed view through a simple prompting technique.

4.4 Probing the Perspective-Awareness of VLMs

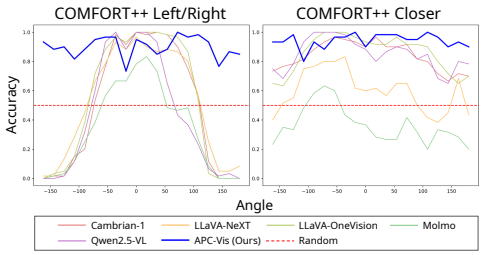

Figure 8: Each plot shows accuracy versus the angular offset θ between the camera and the reference viewpoint. While baselines show clear degradation at certain ranges of θ, APC retains robust accuracy across all angles, demonstrating strong perspective-aware reasoning.

Finally, we analyze the *perspective-awareness* of each method by assessing spatial reasoning accuracy across different viewpoints. Specifically, we select two tasks—*left/right* and *closer*—and construct 60 scenes similar following our setting in COMFORT++. Each scene is rendered from 20 evenly spaced azimuth angles. We define θ as the angular offset between the camera's orientation and the reference viewer's orientation. Here, for $\theta = 0°$ the camera is aligned with the reference perspective, while $\theta = 180°$ indicates that the reference viewer is facing towards the camera.

The results are shown in Fig. 8. For the *left/right* task (left), the baselines exhibit clear bell-shaped curves, achieving near perfect accuracy when θ is close to $0°$ (egocentric) but rapidly declining as the magnitude of θ increases (allocentric). In contrast, APC maintains consistently high accuracy across all angles, demonstrating strong *perspective-aware* reasoning. For the *closer* task (right), baseline models also show noticeable accuracy drops, especially near the leftmost and rightmost θ ranges. APC consistently achieves over 80% accuracy, robustly handling viewpoints regardless of their deviation from the egocentric perspective.

5 Conclusion

In this work, we introduced APC, a framework empowering VLMs with the capability of perspective-aware reasoning. Our key idea is to simulate the mental imagery process of humans, abstracting the scene in an image to facilitate allocentric-to-egocentric perspective shifts, and in turn convey the transformed view to the VLM in the form of a prompt. The scene abstraction is constructed using vision foundation models for object detection, segmentation, and orientation estimation. The reframed prompt from the new perspective, either in text or image form, is then processed by VLMs, leveraging their egocentric reasoning capabilities. As shown by our experiment on both synthetic and real spatial reasoning benchmarks, APC enables robust accuracy across diverse perspectives, thereby opening new possibilities of VLMs on real-world spatial tasks.

Acknowledgements

This work was supported by the NRF of Korea grant (RS-2023-00209723); IITP grants (RS-2019-II190075, RS-2022-II220594, RS-2023-00227592, RS-2024-00399817, RS-2025-25443318); and the Technology Innovation Program (RS-2025-02317326), all funded by the Korean government (MSIT and MOTIE), as well as grants from the DRB-KAIST SketchTheFuture Research Center. L. Guibas acknowledges the ARL grant (W911NF-21-2-0104) and a Vannevar Bush Faculty Fellowship.

References

[1] P. Tong, E. Brown, P. Wu, S. Woo, A. J. V. IYER, S. C. Akula, S. Yang, J. Yang, M. Middepogu, Z. Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. In *NeurIPS*, 2024.

[2] A.-C. Cheng, H. Yin, Y. Fu, Q. Guo, R. Yang, J. Kautz, X. Wang, and S. Liu. Spatialrgpt: Grounded spatial reasoning in vision language model. In *NeurIPS*, 2024.

[3] B. Chen, Z. Xu, S. Kirmani, B. Ichter, D. Sadigh, L. Guibas, and F. Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *CVPR*, 2024.

[4] C. Ma, K. Lu, T.-Y. Cheng, N. Trigoni, and A. Markham. Spatialpin: Enhancing spatial reasoning capabilities of vision-language models through prompting and interacting 3d priors. In *NeurIPS*, 2024.

[5] J. Li, D. Li, S. Savarese, and S. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

[6] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

[7] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022.

[8] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[9] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[10] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[11] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[12] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models. *arXiv preprint arXiv:2409.17146*, 2024.

[13] Y. Hu, O. Stretcu, C.-T. Lu, K. Viswanathan, K. Hata, E. Luo, R. Krishna, and A. Fuxman. Visual program distillation: Distilling tools and programmatic reasoning into vision-language models. In *CVPR*, 2025.

[14] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[15] W. Cai, I. Ponomarenko, J. Yuan, X. Li, W. Yang, H. Dong, and B. Zhao. Spatialbot: Precise spatial understanding with vision language models. In *ICRA*, 2025.

[16] M. Heo, M.-H. Chen, D.-A. Huang, S. Liu, S. Radhakrishnan, S. J. Kim, Y.-C. F. Wang, and R. Hachiuma. Omni-rgpt: Unifying image and video region-level understanding via token marks. In *CVPR*, 2025.

[17] L. Fan, G. Wang, Y. Jiang, A. Mandlekar, Y. Yang, H. Zhu, A. Tang, D.-A. Huang, Y. Zhu, and A. Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *NeurIPS*, 2022.

[18] D. Yang, K. Yang, Y. Wang, J. Liu, Z. Xu, R. Yin, P. Zhai, and L. Zhang. How2comm: Communication-efficient and collaboration-pragmatic multi-agent perception. In *NeurIPS*, 2023.

[19] Z. Yang, C. Garrett, D. Fox, T. Lozano-Pérez, and L. P. Kaelbling. Guiding long-horizon task and motion planning with vision language models. In *ICRA*, 2025.

[20] Z. Zhang, F. Hu, J. Lee, F. Shi, P. Kordjamshidi, J. Chai, and Z. Ma. Do vision-language models represent space and how? evaluating spatial frame of reference under ambiguities. In *ICLR*, 2025.

[21] D. Linsley, P. Zhou, A. K. Ashok, A. Nagaraj, G. Gaonkar, F. E. Lewis, Z. Pizlo, and T. Serre. The 3d-pc: a benchmark for visual perspective taking in humans and machines. In *ICLR*, 2025.

[22] G. Góral, A. Ziarko, M. Nauman, and M. Wołczyk. Seeing through their eyes: Evaluating visual perspective taking in vision language models. *arXiv preprint arXiv:2409.12969*, 2024.

[23] W. Zhang, W. E. Ng, L. Ma, Y. Wang, J. Zhao, B. Li, and L. Wang. Sphere: A hierarchical evaluation on spatial perception and reasoning for vision-language models. *arXiv preprint arXiv:2412.12693*, 2024.

[24] B. Nanay. Mental imagery. *The Stanford Encyclopedia of Philosophy*, 2021. URL https://plato.stanford.edu/archives/win2021/entries/mental-imagery/.

[25] R. Finke. Principles of mental imagery, 1989.

[26] A. Paivio. *Imagery and Verbal Processes (1st ed.)*. Psychology Press, 1979.

[27] S. M. Kosslyn, T. M. Ball, and B. J. Reiser. Visual images preserve metric spatial information: Evidence from studies of image scanning. *Journal of Experimental Psychology: Human Perception and Performance*, 1978.

[28] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171 (3972):701–703, 1971.

[29] G. G. Cole, S. Samuel, and M. J. Eacott. A return of mental imagery: The pictorial theory of visual perspective-taking. *Consciousness and Cognition*, 2022.

[30] V. A. Diwadkar and T. P. McNamara. Viewpoint dependence in scene recognition. *Psychological science*, 1997.

[31] N. Burgess. Spatial memory: how egocentric and allocentric combine. *Trends in cognitive sciences*, 2006.

[32] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *ECCV*, 2024.

[33] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[34] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.

[35] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. In *ICCV*, 2023.

[36] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv*, 2024.

[37] Z. Wang, Z. Zhang, T. Pang, C. Du, H. Zhao, and Z. Zhao. Orient anything: Learning robust object orientation estimation from rendering 3d models. *arXiv preprint arXiv:2412.18605*, 2024.

[38] W. Ma, H. Chen, G. Zhang, C. M. de Melo, A. Yuille, and J. Chen. 3dsrbench: A comprehensive 3d spatial reasoning benchmark. *arXiv preprint arXiv:2412.07825*, 2024.

[39] H. Liu, C. Li, Y. Li, B. Li, Y. Zhang, S. Shen, and Y. J. Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

[40] S. K. Ramakrishnan, E. Wijmans, P. Kraehenbuehl, and V. Koltun. Does spatial cognition emerge in frontier models? In *ICLR*, 2025.

[41] Y. Tang, A. Qu, Z. Wang, D. Zhuang, Z. Wu, W. Ma, S. Wang, Y. Zheng, Z. Zhao, and J. Zhao. Sparkle: Mastering basic spatial capabilities in vision language models elicits generalization to composite spatial reasoning. *arXiv preprint arXiv:2410.16162*, 2024.

[42] J. Wang, Y. Ming, Z. Shi, V. Vineet, X. Wang, S. Li, and N. Joshi. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. In *NeurIPS*, 2024.

[43] P. Rahmanzadehgervi, L. Bolton, M. R. Taesiri, and A. T. Nguyen. Vision language models are blind. In *ACCV*, 2024.

[44] X. Fu, Y. Hu, B. Li, Y. Feng, H. Wang, X. Lin, D. Roth, N. A. Smith, W.-C. Ma, and R. Krishna. Blink: Multimodal large language models can see but not perceive. In *ECCV*, 2024.

[45] A. Kamath, J. Hessel, and K.-W. Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In *EMNLP*, 2023.

[46] F. Liu, G. Emerson, and N. Collier. Visual spatial reasoning. In *EMNLP*, 2023.

[47] F. Shiri, X.-Y. Guo, M. Far, X. Yu, R. Haf, and Y.-F. Li. An empirical analysis on spatial reasoning capabilities of large multimodal models. In *EMNLP*, 2024.

[48] C. H. Song, V. Blukis, J. Tremblay, S. Tyree, Y. Su, and S. Birchfield. Robospatial: Teaching spatial understanding to 2d and 3d vision-language models for robotics. In *CVPR*, 2025.

[49] J. Yang, S. Yang, A. W. Gupta, R. Han, L. Fei-Fei, and S. Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *CVPR*, 2025.

[50] C. Li, C. Zhang, H. Zhou, N. Collier, A. Korhonen, and I. Vulić. Topviewrs: Vision-language models as top-view spatial reasoners. In *EMNLP*, 2024.

[51] W. Yuan, J. Duan, V. Blukis, W. Pumacay, R. Krishna, A. Murali, A. Mousavian, and D. Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. In *CORL*, 2024.

[52] Y. Liu, D. Chi, S. Wu, Z. Zhang, Y. Hu, L. Zhang, Y. Zhang, S. Wu, T. Cao, G. Huang, et al. Spatialcot: Advancing spatial reasoning through coordinate alignment and chain-of-thought for embodied task planning. *arXiv preprint arXiv:2501.10074*, 2025.

[53] D. Surís, S. Menon, and C. Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *ICCV*, 2023.

[54] T. Gupta and A. Kembhavi. Visual programming: Compositional visual reasoning without training. In *CVPR*, 2023.

[55] D. Marsili, R. Agrawal, Y. Yue, and G. Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. In *CVPR*, 2025.

[56] L. Jin, J. Zhang, Y. Hold-Geoffroy, O. Wang, K. Blackburn-Matzen, M. Sticha, and D. F. Fouhey. Perspective fields for single image camera calibration. In *CVPR*, 2023.

[57] M. Liu, R. Shi, L. Chen, Z. Zhang, C. Xu, X. Wei, H. Chen, C. Zeng, J. Gu, and H. Su. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In *CVPR*, 2024.

[58] P. D. Sette, M. Bindemann, and H. J. Ferguson. Visual perspective-taking in complex natural scenes. *Quarterly Journal of Experimental Psychology*, 2022.

[59] H. H. Clark and S. E. Brennan. Grounding in communication. *Perspectives on Socially Shared Cognition*, 1991.

[60] C. Beckham, M. Weiss, F. Golemo, S. Honari, D. Nowrouzezahrai, and C. Pal. Visual question answering from another perspective: Clevr mental rotation tests. *Pattern Recognition*, 2023.

[61] A. Ray, J. Duan, R. Tan, D. Bashkirova, R. Hendrix, K. Ehsani, A. Kembhavi, B. A. Plummer, R. Krishna, K.-H. Zeng, et al. Sat: Spatial aptitude training for multimodal language models. *arXiv preprint arXiv:2412.07755*, 2024.

[62] J. Wu, Z. Zhang, Y. Xia, X. Li, Z. Xia, A. Chang, T. Yu, S. Kim, R. A. Rossi, R. Zhang, et al. Visual prompting in multimodal large language models: A survey. *arXiv preprint arXiv:2409.15310*, 2024.

[63] B. Liu, Y. Dong, Y. Wang, Y. Rao, Y. Tang, W.-C. Ma, and R. Krishna. Coarse correspondence elicit 3d spacetime understanding in multimodal language model. In *CVPR*, 2025.

[64] X. Lei, Z. Yang, X. Chen, P. Li, and Y. Liu. Scaffolding coordinates to promote vision-language coordination in large multi-modal models. *arXiv preprint arXiv:2402.12058*, 2024.

[65] Y. Wu, Y. Wang, S. Tang, W. Wu, T. He, W. Ouyang, P. Torr, and J. Wu. Dettoolchain: A new prompting paradigm to unleash detection ability of mllm. In *ECCV*, 2024.

[66] L. Yang, Y. Wang, X. Li, X. Wang, and J. Yang. Fine-grained visual prompting. In *NeurIPS*, 2023.

[67] Q. Zhou, R. Zhou, Z. Hu, P. Lu, S. Gao, and Y. Zhang. Image-of-thought prompting for visual reasoning refinement in multimodal large language models. *arXiv preprint arXiv:2405.13872*, 2024.

[68] A. Shtedritski, C. Rupprecht, and A. Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. In *ICCV*, 2023.

[69] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, and J. Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*, 2023.

[70] Y. Hu, W. Shi, X. Fu, D. Roth, M. Ostendorf, L. Zettlemoyer, N. A. Smith, and R. Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *NeurIPS*, 2024.

[71] C. Li, W. Wu, H. Zhang, Y. Xia, S. Mao, L. Dong, I. Vulić, and F. Wei. Imagine while reasoning in space: Multimodal visualization-of-thought. *arXiv preprint arXiv:2501.07542*, 2025.

[72] D. Rose, V. Himakunthala, A. Ouyang, R. He, A. Mei, Y. Lu, M. Saxon, C. Sonar, D. Mirza, and W. Y. Wang. Visual chain of thought: bridging logical gaps with multimodal infillings. *arXiv preprint arXiv:2305.02317*, 2023.

[73] W. Wu, S. Mao, Y. Zhang, Y. Xia, L. Dong, L. Cui, and F. Wei. Mind's eye of llms: Visualization-of-thought elicits spatial reasoning in large language models. In *NeurIPS*, 2024.

[74] Z. Chen, Q. Zhou, Y. Shen, Y. Hong, Z. Sun, D. Gutfreund, and C. Gan. Visual chain-of-thought prompting for knowledge-based visual reasoning. In *AAAI*, 2024.

[75] H. Shao, S. Qian, H. Xiao, G. Song, Z. Zong, L. Wang, Y. Liu, and H. Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. *NeurIPS*, 2024.

[76] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *CVPR*, 2025.

[77] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *ICCV*, 2023.

[78] A. Bochkovskii, A. Delaunoy, H. Germain, M. Santos, Y. Zhou, S. R. Richter, and V. Koltun. Depth pro: Sharp monocular metric depth in less than a second. In *ICLR*, 2025.

[79] B. O. Community. Blender - a 3d modelling and rendering package, 2018. URL http://www.blender.org.

[80] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[81] W. Yu, J. Xing, L. Yuan, W. Hu, X. Li, Z. Huang, X. Gao, T.-T. Wong, Y. Shan, and Y. Tian. Viewcrafter: Taming video diffusion models for high-fidelity novel view synthesis. *arXiv preprint arXiv:2409.02048*, 2024.

[82] Z. Wang, Z. Zhang, T. Pang, C. Du, H. Zhao, and Z. Zhao. Orient anything: Learning robust object orientation estimation from rendering 3d models. *arXiv*, 2024.

[83] Y. Nie, X. Han, S. Guo, Y. Zheng, J. Chang, and J. J. Zhang. Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *CVPR*, 2020.

[84] G. Brazil, A. Kumar, J. Straub, N. Ravi, J. Johnson, and G. Gkioxari. Omni3d: A large benchmark and model for 3d object detection in the wild. In *CVPR*, 2023.

[85] J. Yao, H. Gu, X. Chen, J. Wang, and Z. Cheng. Open vocabulary monocular 3d object detection. *arXiv preprint arXiv:2411.16833*, 2024.

[86] M. Dahnert, J. Hou, M. Nießner, and A. Dai. Panoptic 3d scene reconstruction from a single rgb image. In *NeurIPS*, 2021.

[87] A.-Q. Cao and R. De Charette. Monoscene: Monocular 3d semantic scene completion. In *CVPR*, 2022.

[88] Y. Nie, A. Dai, X. Han, and M. Nießner. Learning 3d scene priors with 2d supervision. In *CVPR*, 2023.

[89] M. Liu, C. Xu, H. Jin, L. Chen, M. Varma T, Z. Xu, and H. Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. In *NeurIPS*, 2023.

[90] Dawson-Haggerty et al. trimesh. URL https://trimesh.org/.

[91] H. Duan, J. Yang, Y. Qiao, X. Fang, L. Chen, Y. Liu, X. Dong, Y. Zang, P. Zhang, J. Wang, et al. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. In *ACM MM*, 2024.

[92] Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *ECCV*, 2024.

[93] M. Deitke, R. Liu, M. Wallingford, H. Ngo, O. Michel, A. Kusupati, A. Fan, C. Laforte, V. Voleti, S. Y. Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. In *NeurIPS*, 2023.

Appendix

In this appendix, we first discuss the limitations of our work and potential directions for future work (Sec. A). We then analyze the failure cases of two dense reconstruction-based baselines—SpatialPIN* [4] and ViewCrafter [81]—in allocentric reasoning scenarios (Sec. B). We describe the implementation details of our APC framework (Sec. C) and provide details on the evaluation setups (Sec. D). Finally, we provide the text prompts used in each stage of our method (Sec. E).

A Limitations and Future Work

Our APC framework empowers VLMs with perspective-aware spatial reasoning, but its use of multiple vision foundation models [32, 77, 82] introduces additional memory usage compared with running the VLM alone. In our experiments, we ran inference on two NVIDIA RTX 3090 GPUs each with 24GB VRAM.

Moreover, the robustness of the scene abstraction stage depends on the accuracy its detection, segmentation, and orientation modules. To quantify each module's impact, we conducted an ablation study by substituting its predictions with the oracle ground truth. The left-hand table in Fig. A.1 shows the ablated accuracy on the *left/right* split of COMFORT++, while the right-hand figures depict representative failure cases. Injecting oracle information consistently improves performance, indicating that APC's accuracy is indeed influenced by each module's robustness, yet does not work as a critical bottleneck.

	Accuracy
APC-Vis	89.67
+ det./seg./depth	91.67
+ ori.	96.67
+ det./seg./depth/ori.	97.33

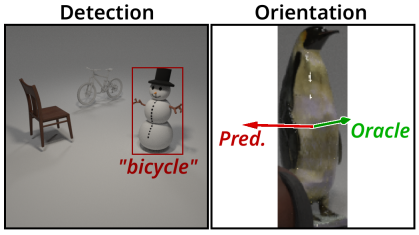

Figure A.1: **Ablation.** Accuracy of APC with oracle values (left), and failure cases of the vision modules (right).

While in this work we introduced a minimal yet effective form of 3D abstraction for perspective change in VLMs, exploring richer scene abstractions from images could offer an promising direction for future research—such as the use of 3D bounding boxes [83, 84, 85] and coarse, semantic 3D scene reconstructions [86, 87, 88].

B Analysis on Dense Reconstruction Baselines

In this section, we further discuss the dense reconstruction-based baselines introduced in Sec. 4.1. In contrast to APC's abstraction-based approach, another intuitive approach for perspective-aware spatial reasoning is to perform a dense 3D reconstruction of the scene and then render a novel view from the target perspective. This new view can then be provided to the VLM instead of the visual prompt used in Sec. 3.3. We explore two such approaches that involve dense 3D reconstruction process: (1) a modified version of SpatialPIN [4], which directly lifts objects from the image into meshes and renders them from the target view, and (2) ViewCrafter [81], which synthesizes novel views by using an intermediate point cloud reconstruction. As the original SpatialPIN [4] does not include a rendering phase for novel target perspectives, we refer to our extended pipeline as SpatialPIN*. For the inference of SpatialPIN*, we used One-2-3-45 [89] in contrast to One-2-3-45++ [57] in the original paper due to the limited access of the API.

While a dense reconstruction-based approach may appear to be an obvious alternative to our abstraction-based framework, our experiments show that constructing an accurate and descriptive view of the target perspective is challenging and expensive. As illustrated in Fig. A.2, the synthesized novel views from both SpatialPIN* (row 1) and ViewCrafter (row 2) are often excessively

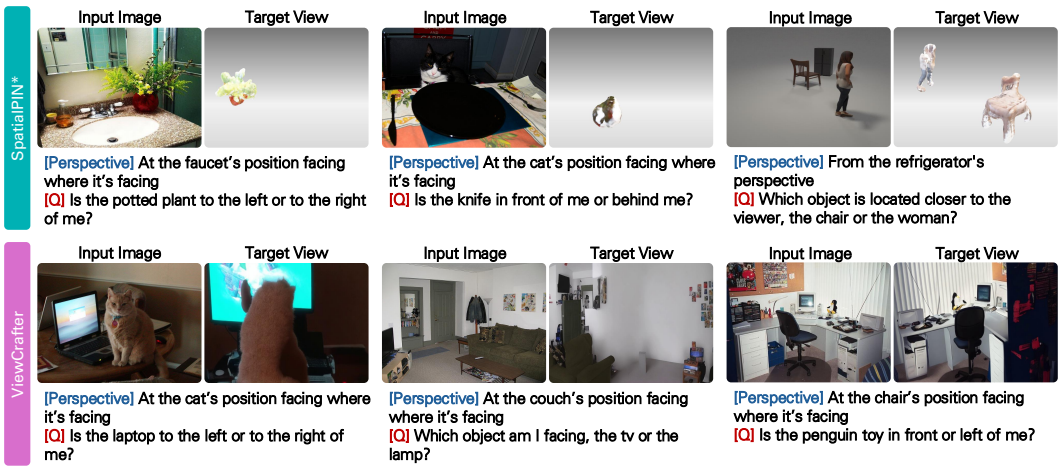

Figure A.2: **Dense Reconstruction Baseline Examples.** Novel views synthesized by SpatialPIN* [4] and ViewCrafter [81] both display noisy and inaccurate objects and scene structures lacking the original context of the input image, thereby leading to low accuracy when VLMs are fed the images as a visual input for spatial reasoning.

noisy and fail to preserve the context of the input image. Consequently, providing these reconstructed views to the VLM for spatial reasoning results in lower accuracy as previously shown in Tab. 1. In addition, both methods incur notably longer inference times due to the dense 3D reconstruction steps, as shown in Tab. B.1. In contrast, as in our APC, constructing an minimal abstraction of the scene with precise mappings between the original objects and their abstractions not only yields more accurate reasoning but also substantially reduces inference time.

Method	SpatialPIN* [4]	ViewCrafter [81]	APC (Ours)
Time (s)	336.21	260.57	17.47

Table B.1: **Inference Time Comparison.** Both dense reconstruction-based baselines [4, 81] require over 14 times the inference time of our APC to answer a single question.

C Implementation Details

In this section, we provide the implementation details of our APC framework in Sec. 3. As the backbone VLM, we used `Qwen2.5-VL-7B-Instruct`[1].

C.1 Scene Abstraction

Detection Refinement with VLM. While GroundingDINO [32] excels in object detection, it often struggles when the input text prompt is complex. We add a simple refinement stage utilizing the VLM for improved detection accuracy. For each object description t_i we keep GroundingDINO's predicted candidates whose confidence exceeds a threshold s, then select the top k candidates. The corresponding image crops are laid out in a grid, and we query the VLM to select the crop that best matched t_i. We set $s = 0.15$ and $k = 5$. Fig. C.1 illustrates a case in which the initial GroundingDINO output is incorrect but is corrected by this refinement step.

Filtering Outliers. To obtain the 3D position of each object abstraction $O_i \in S_E$, we unproject the segmented pixels using the predicted depth map. To handle outliers caused by background pixels being included in the segmentation masks, we filter out the points whose depth values fall outside the range $[0.9d_i, 1.1d_i]$, where d_i is the mode depth within the mask. We then assign the coordinate-wise median of the remaining points in the remaining points as the 3D position c_i of object O_i.

[1] `https://huggingface.co/Qwen/Qwen2.5-VL-7B-Instruct`

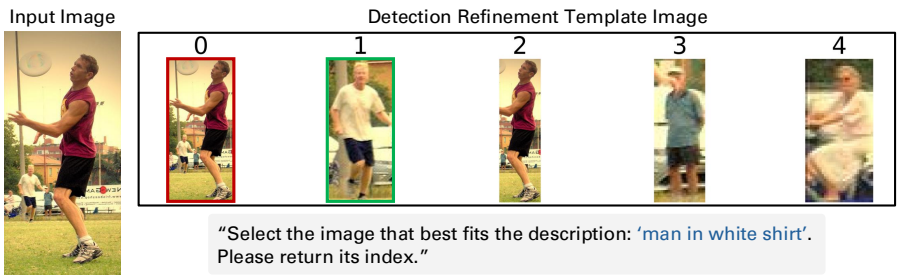

Figure C.1: **Detection Refinement Example.** Starting with candidate detections from GroundingDINO [32], we select the top k predictions and present them as a grid of cropped images (right). We then query the VLM to return the index that best aligns with the input text prompt. Red indicates GroundingDINO's initial choice and green indicates the refined choice.

C.2 Egocentric Rephrasing

Recall that our `APC` converts an *allocentric* question Q—originally stated with respect to a reference viewpoint A—into an *egocentric* one posed from A itself. To ensure compatibility with the perspective prompts introduced in Sec. 3.3, we remove the explicit perspective descriptions from Q. In practice, we query the VLM to rewrite Q, excluding the phrases that mention a reference perspective. In turn we obtain a perspective-agnostic reformulation of the task, which is then used in each type of perspective prompt.

C.3 Visual Prompt Rendering.

To render a *visual prompt* from the transformed scene abstraction $S_A = \{O'_i\}_{i=1}^n$ as shown in Fig. 5, we use the Trimesh renderer [90]. Note that S_A is defined in the coordinate system of the reference perspective A. Each object O'_i is converted to an equal-sized cube with distinct colors, and the visual prompt is obtained by rendering the scene accordingly. Given the camera in S_A faces in the positive z-direction, only the objects with $z > 0$ appear in the visual prompt. Objects with $z \leq 0$ are considered to be out of view (not visible) from perspective A.

Normalization. To prevent cubes from appearing too small or large in the visual prompt, we normalize the coordinates of S_A, ensuring z values lie within a predefined range $[z_{\text{min}}, z_{\text{max}}]$. Likewise, we scale the x, y coordinates into a fixed range $[-d^*, d^*]$ to keep objects within the view frustum.

Camera Translation. By default, we place the camera at reference viewer's position—the origin of S_A. As an exception, for the *left/right* task in 3DSRBench [38], we shift the camera backward along the z-axis to ensure all objects in the scene appear in the visual prompt. This adjustment is applied to match the benchmark's setup, where an object that lies *behind* and *to the right* of a reference viewer is still treated as being on the right side from that viewer's perspective.

D Evaluation Details

In this section, we provide further details on our evaluation setup in Sec. 4. Each VLM response is scored with a two-step process that combines *exact matching* and *LLM-assisted* evaluation. We first perform exact matching: if the response consists solely of the correct option index or the exact answer phrase, we label it as correct. Otherwise, we pass the entire response to an LLM along with the answer to determine its correctness. For this, we used the judgment prompt template from VLMEvalKit [91].

Following 3DSRBench [38], we employ CircularEval [92], which takes into account VLM's response consistency by permuting the answer options for each image-question pair. The VLM is considered to be correct for a question Q only if it selects the correct across all permutations. CircularEval is applied for both COMFORT++ [20] and 3DSRBench [38].

To construct the COMFORT++ benchmark for each task, we first collected 7 object meshes from the original implementation [20] and additional 6 meshes from Objaverse-XL [93]. For the *left/right*

and *closer* tasks, we arranged three objects in a predefined layout, designating one as the reference viewer, and added random perturbations to the objects' x, y coordinates to diversify the layouts. We prepared 60 scenes and rendered each from 20 evenly spaced azimuth viewpoints. Then, we randomly sampled five views per scene, resulting in a total of 300 images for each task. For the *visibility* task, we created 160 scenes, each containing a reference viewer and single target object positioned so that the object is either visible or invisible from the viewer's perspective. We rendered each scene two opposite viewpoints, yielding 320 images. Finally, for the *facing* task, we arranged three objects in a linear configuration, setting the central object as the reference viewer, and oriented it to face either one of the two remaining objects. Each scene is rendered once, resulting in 300 images in total.

For 3DSRBench [38], we used the original *left/right* and *facing* criteria. We recasted the *front/behind* task as a *visibility* judgment for two reasons: (i) the provided task can be more naturally interpreted as deciding whether an object is visible from the reference object's viewpoint, and (ii) VLMs struggle to infer that an object is *behind* it when the object is not present in the image itself. This adjustment better serves our goal of measuring the egocentric and allocentric reasoning capabilities of VLMs.

E Details on Text Prompts

In this section, we present the text prompts used at each stage of our APC pipeline. To guide the VLM towards the desired response format, we include examplar question-answer pairs for in-context learning. For the text prompt fed along with the visual prompt, we add simple prompt engineering to help suppress hallucinations: we (i) define the the relation *"facing towards"* and (ii) explicitly that the larger object is considered as being closer to the viewer—an assumption that holds since our abstraction assigns equal size to every object.

(1) Scene Abstraction (Sec. 3.1) — Extracting Objects of Interest.

```
# Situation Description
Given an image and a spatial-reasoning question, identify all entities mentioned
in the question.

# Example
[Question] You are standing at the airplane's position, facing where it is
facing.  Is the person on your left or right?
[Detect] [airplane, person]

# Your Task
Now, given the question below, list the entities that appear in the question.

[Question] {Question}
[Detect]
```

(2) Perspective Change (Sec. 3.2) — Setting a Reference Perspective

```
Given a question about spatial reasoning, we want to extract the perspective
of the question.  If the question is from the camera's perspective, return
++camera++.

# Example
[Question] From the woman's perspective, is the tree on the left or right?
[Perspective] ++woman++

# Your Task
Given the question below, please specify the perspective from which the question
is asked.
You must return in the format:  [Perspective] ++object_name++

[Question] {Question}
[Options] obj1, obj2, ..., camera
```

```
[Perspective]
```

(3) Egocentric Rephrasing (Sec. C.2)

```
From a sentence with a perspective description, we need to remove the perspective
description.

# Example
[Question] From the car's perspective, which is on the right side:  the person or
the tree?
[Output] Which is on the right side:  the person or the tree?

# Your Task
Given the question below, please remove the perspective description.

[Question] {Question}
[Output]
```

(4) Perspective Prompting (Sec. 3.3) — Visual Prompt.

```
This is an image of a 3D scene.

- The viewer is facing towards the object that is closest to the center.
- A larger object is closer to the viewer compared to a smaller object.

# Task
Based on the image, please answer the following question.

{Question}

Please only return the answer.
```

(5) Perspective Prompting (Sec. 3.3) — Numerical Prompt.

```
Imagine that you are at the {src_obj}'s position and facing where it is facing.
We have the coordinates of different objects in {src_obj}'s coordinate system.

# Coordinate System
- The origin is at the {src_obj}'s position.
- The {src_obj}'s facing direction is [0, 0, 1], which is aligned with the
z-axis.
- The x-axis is to the right, the y-axis is up, and the z-axis is forward.

# Object Coordinates
[...]

# Task
Given the above {src_obj}'s coordinate system and the object coordinates, please
answer the following question:

[Question] {Question}
```