TRACE: Textual Reasoning for Affordance Coordinate Extraction

Sangyun Park[1,*]   Jin Kim[2,3,*]   Yuchen Cui[4,5]   Matthew S. Brown[2,3,†]

[1]ABB Robotics, Seoul, Republic of Korea
[2]Department of Radiology, University of California, Los Angeles, CA, USA
[3]Center for Computer Vision and Imaging Biomarkers, University of California, Los Angeles, CA, USA
[4]Computer Science Department, University of California, Los Angeles, CA, USA
[5]Robot Intelligence Lab, University of California, Los Angeles, CA, USA

{sang-yun.park@kr.abb.com, kimjin116@g.ucla.edu, yuchencui@cs.ucla.edu, mbrown@mednet.ucla.edu}

Abstract

*Vision-Language Models (VLMs) struggle to translate high-level instructions into the precise spatial affordances required for robotic manipulation. While visual Chain-of-Thought (CoT) methods exist, they are often computationally intensive. In this work, we introduce **TRACE (Textual Reasoning for Affordance Coordinate Extraction)**, a novel **methodology** that integrates a textual **Chain of Reasoning (CoR)** into the affordance prediction process. We use this methodology to create the **TRACE dataset**, a large-scale collection created via an autonomous pipeline that pairs instructions with explicit textual rationales. By fine-tuning a VLM on this data, our model learns to externalize its spatial reasoning before acting. Our experiments show that our **TRACE-tuned model** achieves state-of-the-art performance, reaching **48.1%** accuracy on the primary Where2Place (W2P) benchmark (a 9.6% relative improvement) and **55.0%** on the more challenging W2P(h) subset. Crucially, an ablation study demonstrates that performance scales directly with the amount of reasoning data used, confirming the CoR's effectiveness. Furthermore, analysis of the model's attention maps reveals an interpretable reasoning process where focus shifts dynamically across reasoning steps. This work shows that training VLMs to generate a textual CoR is an effective and robust strategy for enhancing the precision, reliability, and interpretability of VLM-based robot control. Our dataset and code are available at* https://github.com/jink-ucla/TRACE.

1. Introduction

The quest to build generalist robots capable of performing diverse manipulation tasks has been significantly advanced by the advent of Vision-Language Models (VLMs) and Vision-Language-Action (VLA) models [24, 32]. These models show immense promise by leveraging vast commonsense knowledge to interpret natural language instructions and visual scenes for high-level task planning [10, 14]. However, a critical gap remains between high-level reasoning and the precise, low-level spatial understanding required for physical manipulation [20, 31]. Effectively grounding ambiguous language commands in the physical world of robot affordances is a fundamental challenge [1, 23], limiting the reliability of VLM-driven robots in unstructured environments.

Recent research has explored several avenues to bridge this gap. One popular approach is to have VLMs predict intermediate action representations, such as affordance keypoints, which serve as targets for low-level controllers [11, 15, 27, 29]. While effective, these methods often treat the VLM as a black-box predictor, without an explicit, interpretable reasoning process. Concurrently, a new wave of research has focused on enabling VLMs to "think visually" through a visual Chain-of-Thought (CoT). These approaches allow models to generate intermediate visual states, such as future goal images [13, 32], edit the input image to refocus attention [5, 22], or perform guided visual search to find relevant details [3, 26, 28, 30]. These methods demonstrate that iterative, multi-step visual processing is crucial for complex reasoning. However, they can be computationally intensive and often rely on external tools or generative models, creating a complex pipeline for robotic control.

In this work, we argue that for a VLM to generate pre-

---

*Equal contribution.
†Corresponding author.

Question

The image features two objects, one enclosed by a red rectangle and the other surrounded by a green rectangle. Locate and indicate several points within the vacant area that lies between these delineated objects. Your answer should be formatted as a list of tuples, i.e. [(x1, y1), (x2, y2), ...], where each tuple contains the x and y coordinates of a point satisfying the conditions above. The coordinates should be between 0 and 1, indicating the normalized pixel locations of the points in the image.

Step1. Identify the reference objects and target region
The user has highlighted two objects with red and green borders. The request asks to find points in the vacant area situated between these two outlined objects.

Step2. Determine the goal's subtype
The user is asking for points in an empty space between two distinct objects. This indicates the goal is to find an independent empty space, fitting the Placement Affordance subtype, rather than a part of one of the referenced surfaces.

Step3. Define the search space
The vacant area lies on the white surface of the appliance (likely a counter or table) that is positioned between the red-outlined object (a white rectangular object with a red border) and the green-outlined object (a yellow rectangular object).

Step4. Generate the output
Points are selected from this identified vacant area on the appliance's surface, and their coordinates are normalized to fall between 0 and 1 to produce the final answer.

Final Answer
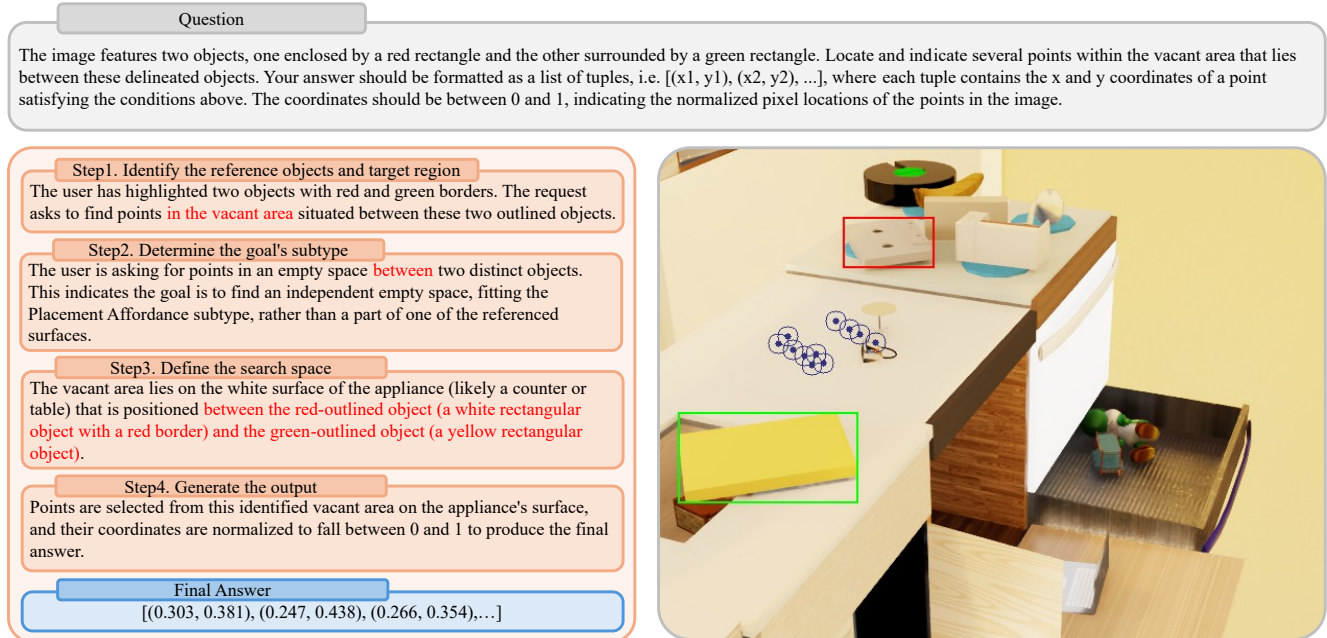[(0.303, 0.381), (0.247, 0.438), (0.266, 0.354),…]

Figure 1. An example data point from our TRACE reasoning dataset illustrating its overall structure. Each entry consists of an image and a corresponding natural language question that requires spatial reasoning, such as finding the vacant area between the two delineated objects. The dataset also provides the explicit, multi-step reasoning process required to solve the instruction. This process includes identifying reference objects, determining the goal's subtype as a "Placement Affordance", defining the search space on the appliance's surface, and generating the final coordinates as a list of tuples.

cise and reliable low-level actions, it must not only predict what to do but also reason about why it is doing it. Inspired by the success of CoT prompting in enhancing the reasoning capabilities of LLMs [25], and its recent application in robotics [7, 8], we introduce a framework that integrates a textual 'Chain of Reasoning' (CoR) into the spatial affordance prediction process, as illustrated in Figure 1. Our approach differs from visual CoT methods by focusing on generating an interpretable textual rationale for a grounded visual action. This strategy directly leverages the VLM's linguistic strengths and is computationally simpler than generating intermediate images.

To achieve this, we introduce the **TRACE (Textual Reasoning for Affordance Coordinate Extraction) methodology**. This methodology includes a new data generation pipeline used to create the large-scale **TRACE dataset**, created by enhancing the RoboPoint [29] data generation pipeline. Our pipeline now automatically generates explicit textual reasoning steps that justify the selection of ground-truth action points. By fine-tuning a VLM on this augmented dataset, we teach it to externalize its reasoning process, leading to more accurate and robust predictions for low-level manipulation.

- We introduce the **TRACE methodology** and the resulting large-scale **TRACE dataset**, which pairs language instructions with both keypoint affordances and a CoR, cre-

ated via a modified autonomous data generation pipeline.
- We demonstrate that instruction-tuning a VLM with our CoR data significantly improves its spatial understanding and the precision of its predicted action points for robotic manipulation tasks.
- Our experiments show that our **TRACE-tuned model (RoboPoint+TRACE)** achieves **48.1%** accuracy on the challenging Where2Place benchmark, a statistically significant **9.6% relative improvement** over the original RoboPoint model, showcasing a more interpretable and reliable path for VLM-based control.

2. Related Work

Our work is situated at the intersection of VLM-driven robotic manipulation, spatial affordance prediction, and CoT reasoning. We build upon methods for learning visual affordances while drawing a sharp contrast with the emerging field of visual CoT.

2.1. Chain-of-Thought for Robotics and Vision

The concept of CoT prompting, introduced by Wei *et al.* [25], demonstrated that prompting large language models to generate intermediate textual steps significantly improves their performance on complex reasoning tasks. This paradigm has been foundational, inspiring a new direction in how models approach problem-solving. Our work adapts
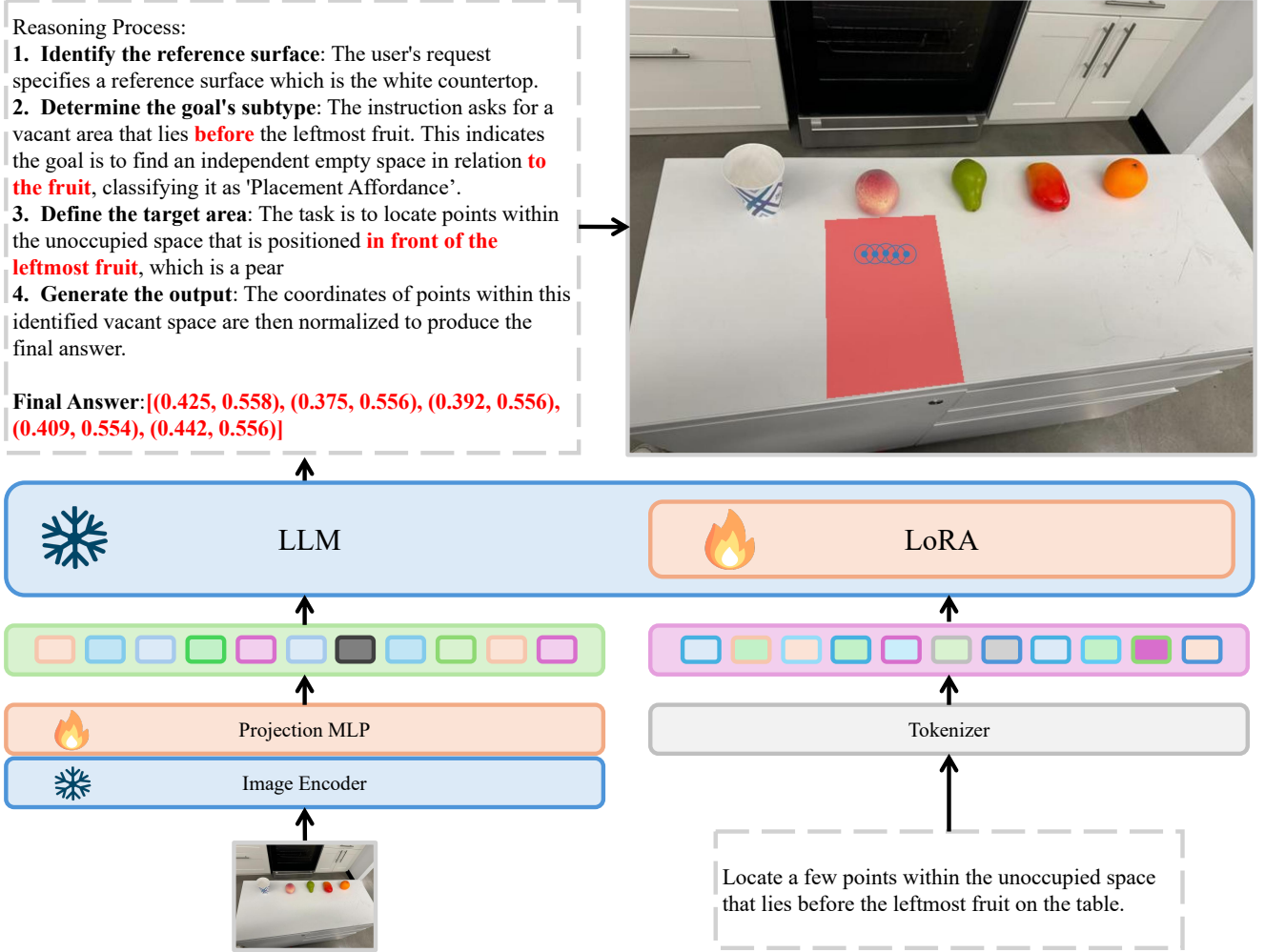
Reasoning Process:
1. Identify the reference surface: The user's request specifies a reference surface which is the white countertop.
2. Determine the goal's subtype: The instruction asks for a vacant area that lies **before** the leftmost fruit. This indicates the goal is to find an independent empty space in relation **to the fruit**, classifying it as 'Placement Affordance'.
3. Define the target area: The task is to locate points within the unoccupied space that is positioned **in front of the leftmost fruit**, which is a pear
4. Generate the output: The coordinates of points within this identified vacant space are then normalized to produce the final answer.

Final Answer:[(0.425, 0.558), (0.375, 0.556), (0.392, 0.556), (0.409, 0.554), (0.442, 0.556)]

LLM

LoRA

Projection MLP

Image Encoder

Tokenizer

Locate a few points within the unoccupied space that lies before the leftmost fruit on the table.

Figure 2. Overview of the model's reasoning pipeline. Given an image and a corresponding natural language instruction, the system begins a multi-step reasoning process. The model first determines the subtype of the goal and establishes the relevant reference surface based on the image. It then defines the target area by interpreting the request. This process utilizes an Image Encoder, Tokenizer, and a Large Language Model (LLM). Finally, a Projection MLP generates the output, which consists of normalized coordinates for points within the identified vacant space.

this core idea, which we term a textual CoR, to the visuo-motor domain of spatial grounding for robotics.

Recently, similar textual reasoning strategies have been applied to robotics. For instance, Hao *et al.* have explored a "Chain of Robotic Actions Reasoning" to generate sequences of high-level actions for manipulation tasks [8] and humanoid locomotion [7]. These methods show that breaking down a complex instruction into a textual plan improves task success. Our work shares this emphasis on textual, step-by-step reasoning. However, instead of generating high-level action plans, we apply the reasoning process to a different problem: justifying the selection of precise, low-level spatial coordinates for manipulation. Our CoR provides an explicit rationale for *why* a specific point in space is chosen, directly linking high-level reasoning to

low-level physical grounding.

2.2. Contrasting with Visual CoT

In parallel, a significant body of work has explored a **Visual CoT**, where reasoning is externalized through intermediate visual states rather than text. A prominent approach is the generation of future images that act as subgoals. For example, CoT-VLA [32] generates images of anticipated future scenes, while other work focuses on visualizing intermediate steps to guide the reasoning process [13].

Other forms of Visual CoT involve editing or interacting with the input image. REFOCUS [5] generates code to programmatically edit the visual input, such as by masking or highlighting regions, to focus the model's attention. Similarly, other methods enable the model to perform in-

teractive reasoning by selectively revisiting parts of an image [3] or conducting a guided visual search to find missing details [26].

While these visual and interactive methods are powerful, they often introduce significant computational overhead from image generation, editing, or multi-step visual processing. Our textual CoR framework is proposed as a lightweight yet effective alternative. By grounding the reasoning process in language—a native modality for VLMs—we improve the precision of spatial predictions without the complexity and computational cost associated with generating or manipulating visual data. Our approach leverages the model's innate linguistic capabilities to directly enhance its spatial understanding.

3. Method

Our approach enhances a VLM's ability to ground language in the physical world by fine-tuning it on our novel **TRACE** dataset. This dataset is specifically designed to integrate a textual **CoR** into the learning process, teaching the model not only to predict precise spatial affordances but also to articulate the reasoning behind them. This section details the model architecture, the dataset construction, and the optimization strategies employed.

3.1. Model Architecture

Our model architecture is composed of three core components: a base language model, a vision encoder, and a multimodal projector. For our primary results, we use **Vicuna-v1.5-13B** [2] as the large language model. **For computationally intensive experiments, such as our ablation study and attention visualization, we utilize a smaller Vicuna-v1.5-7B variant to facilitate analysis.** The vision backbone is a pretrained **CLIP-ViT-Large-Patch14-336** [21] model, where we extract visual features from the penultimate layer to capture rich semantic information. These visual tokens are then mapped into the language model's embedding space via a **2-layer MLP projector** that uses a GELU activation function. To ensure memory-efficient and fast computation, the model implementation leverages **Flash Attention 2** [4]. All prompts and responses adhere to the standard `Vicuna v1` conversation template.

3.2. TRACE Dataset and Preprocessing

The cornerstone of our work is the **TRACE dataset**, a large-scale collection of **200,000 training samples** designed to teach VLMs explicit reasoning. The dataset is a composite of two data sources: 100,000 novel reasoning-augmented samples and 100,000 standard visual instruction-tuning samples to match the data scale of the original RoboPoint model [29].

The first component, which is our primary contribution, consists of **100,000 samples with an explicit textual**

Chain of Reasoning (CoR). To create these, we took the image and instruction pairs from the original RoboPoint datasets and used the Gemini API to synthetically generate a step-by-step rationale for each one. This process, detailed in Appendix 8, prompted the model to deconstruct the task into distinct cognitive steps: identifying reference objects, determining the goal's subtype (e.g., "Placement Affordance"), defining the target area, and justifying the final coordinate selection, as illustrated in Figure 1.

The second component consists of an additional **100,000 samples** from the LVIS [6] and VQA [17] datasets. These samples follow a standard instruction-following format without the explicit CoR and ensure that our model is trained on a dataset of comparable size to the baseline, isolating the effect of our reasoning-augmented data.

The learning task for the entire dataset is formulated as an autoregressive prediction problem, where the model must generate both a textual rationale (for the TRACE-specific data) or a direct answer, followed by target coordinates, as shown in Figure 2. Given an input image and a language instruction, the model is trained to generate a response that first outputs the text and then provides the target 2D coordinates, normalized as $\{(x_i, y_i) \mid x_i, y_i \in [0, 1]\}$. This approach teaches the model to connect its reasoning directly to its final spatial prediction.

To manage this large-scale dataset efficiently, we implement several key preprocessing and data-loading optimizations. We enable **lazy preprocessing** to process data on the fly, significantly reducing memory consumption. All images are **padded to a square aspect ratio** to create uniform input for the vision encoder. To further improve training efficiency, we optimize batching by **grouping samples by modality length**, which minimizes the amount of padding required. Finally, to prevent data-loading bottlenecks, the pipeline is supported by **12 dataloader workers**.

3.3. Training and Optimization

We employ two distinct training strategies corresponding to our two model variants.

For our main 13B model, whose results are reported in Table 1, we perform **full fine-tuning (FFT)** for one epoch on the **TRACE dataset**. All weights of the language model are updated during this process to achieve maximum performance.

For the 7B model, used for the ablation study and attention analysis presented in Figure 4 and Figure 5, we use a parameter-efficient approach. Specifically, we employ **Low-Rank Adaptation (LoRA)** [9] with a rank of $r = 128$ and a scaling factor of $\alpha = 256$. This allows us to adapt the model by updating only the LoRA adapters applied to the language model's linear layers.

Both training configurations use the **AdamW optimizer** [18] with a global learning rate of 2×10^{-6} and

Question : Pinpoint several spots in the vacant area that lies to the right of the glass container.

Ground Truth : From dataset masks

TRACE
Reasoning Summary : The empty countertop space to the right of the glass container. This area is visually distinct from the glass container itself and the appliances on the counter
Final Answer : [(0.694, 0.471), (0.716, 0.475),…]

Robotpoint
Final Answer : [(0.806, 0.478), (0.838, 0.479),…]

SpaceLLaVA
Final Answer (bbox) : (0.5, 0.67, 0.64, 0.87)

gpt-4o
Final Answer (bbox) : (0.65, 0.35, 1.0, 0.55)

gemini-2.5-flash-lite
Final Answer (bbox) : (0.44, 0.25, 0.55, 0.35)

Figure 3. A qualitative comparison of TRACE with other leading models on a sample from our reasoning dataset. The given instruction is to "Pinpoint several spots in the vacant area that lies to the right of the glass container".

Table 1. Main results on spatial affordance prediction benchmarks. The metric is the percentage of predicted points falling within the ground-truth mask, averaged over three runs. Our model, TRACE, outperforms all baselines across all benchmarks.

Model	RoboRefIt	W2P	W2P (h)
RoboPoint(FFT)+**TRACE**	**42.9% ± 0.8**	**48.1% ± 0.1**	**55.0% ± 3.5**
RoboPoint(FFT)	41.7% ± 0.6	43.9% ± 0.6	46.9% ± 4.2
RoboPoint(LoRA)+**TRACE**	**48.1% ± 2.8**	**43.7% ± 4.1**	**41.2% ± 7.3**
RoboPoint(LoRA)	40.6% ± 3.0	36.1% ± 1.3	30.7% ± 0.2
SpaceLLaVA [19]	20.0% ± 0.5	15.0% ± 1.6	13.6% ± 2.1
GPT-4o [12]	6.5% ± 0.8	18.7% ± 2.6	17.8% ± 4.8
Gemini	5.2% ± 0.1	7.8% ± 0.2	6.6% ± 0.2

a **cosine annealing scheduler** with a 3% warmup period. To ensure memory efficiency, we utilize **bfloat16 (bf16)** mixed-precision and enable **gradient checkpointing**.

4. Experimental Results

We evaluate our model on challenging spatial affordance prediction tasks to demonstrate that integrating a textual CoR significantly improves performance. Our experiments are designed to answer: 1) How does our **TRACE-tuned model** perform against state-of-the-art VLMs and the original RoboPoint? 2) Is the performance gain statistically significant?

4.1. Benchmarks

Benchmarks: We evaluate performance on two real-world benchmarks:
- **RoboRefIt:** A 250-image dataset featuring cluttered scenes where objects can only be distinguished by relational language instructions [19].
- **Where2Place (W2P):** A challenging 100-image dataset for identifying free space based on relational language. It includes a difficult subset, **Where2Place (h)**, with 30 examples containing relation types not seen during training [29].

Baselines: We compare our model against several strong baselines:
- **RoboPoint:** The original VLM from Yuan et al. [29], which serves as our primary baseline.

- **GPT-4o:** A state-of-the-art proprietary VLM used in a zero-shot prompting setup [12].
- **SpaceLLaVA:** An open-source VLM specialized for spatial reasoning tasks [16].
- **Gemini:** A state-of-the-art proprietary VLM used in a zero-shot prompting setup [24].

4.2. Performance on Spatial Affordance Prediction

Our primary finding is that instruction-tuning with the **TRACE dataset** leads to consistent improvements in spatial affordance prediction across all benchmarks. As shown in Table 1, our **TRACE-tuned model (RoboPoint+TRACE)** achieves **48.1% accuracy on the Where2Place benchmark**, a notable improvement over the 43.9% accuracy of the original RoboPoint. This represents a relative improvement of 9.6%.

To provide a qualitative understanding of these performance differences, Figure 3 illustrates a direct comparison on a challenging sample from our reasoning dataset, where our **TRACE-tuned model** correctly identifies the target area while other models struggle.

The performance gains are even more pronounced on the more challenging **Where2Place (h)** subset, where our **TRACE-tuned model** outperforms the RoboPoint by a significant margin (55.0% vs. 46.9%). These results demonstrate that the explicit textual reasoning process helps the model better resolve the ambiguity in complex, unseen relational instructions. Both our **TRACE-tuned model** and the original RoboPoint significantly outperform other specialist and generalist VLMs, highlighting the strength of the underlying architecture and data generation pipeline.

4.3. Ablation Study on Reasoning Data

To conclusively demonstrate that our primary contribution—the textual Chain of Reasoning (CoR) data—is responsible for the performance improvements, we conducted an ablation study. For this analysis, we utilized the 7B LoRA variant of our model to efficiently assess the impact of data scaling. We trained the model on progressively larger subsets of the TRACE dataset, from 0% (the baseline model trained without any CoR data) to 100% (the full 200,000-sample dataset).

The results, presented in Figure 4, confirm a strong, positive correlation between the quantity of reasoning data and model performance. Specifically, as the training data scales from 0% to 100%, performance on **RoboRefIt** improves by **7.5** points (from 40.6% to 48.1%), on **Where2Place (W2P)** by **7.6** points (from 36.1% to 43.7%), and on **Where2Place (h)** by a substantial **10.5** points (from 30.7% to 41.2%). The positive linear trend for each benchmark is statistically significant, confirming a robust dose-dependent relationship between CoR data and performance.

Notably, the performance on the challenging **Where2Place (h)** subset, which contains unseen relational concepts, exhibits the steepest improvement, with a **34.2% relative gain** over its baseline. This result strongly indicates that exposure to the explicit reasoning process is particularly crucial for enhancing the model's ability to generalize to novel and more complex instructions. The consistent upward trend across all benchmarks validates that the CoR is not just an explanatory artifact but a vital component for improving spatial grounding.

4.4. Analysis of the Reasoning Mechanism

To better understand *why* fine-tuning with the TRACE dataset improves performance, we analyzed the model's internal state by visualizing its attention map during the reasoning process. This provides insight into how the model utilizes the textual Chain of Reasoning (CoR) to solve spatial tasks.

Figure 5 presents such a visualization for the instruction, "Find the free space in front of the window on the left." The attention heatmap reveals a dynamic process. During the initial steps—(1) "Identify Reference Object" and (2) "Define Target Area"—the model's attention is weak and diffuse as it establishes a general context. However, attention becomes more **focused** during step (3), "Determine Goal's Subtype." While still diffuse, a **new, distinct high-attention region (the bright orange spot) emerges** over the general target area under the window. This suggests that classifying the instruction is a critical juncture where the model solidifies its understanding of the task.

Crucially, in the final step (4), "Generate Output," there is almost no visual attention focused on the area where the final coordinates are predicted. This observation supports our central hypothesis: the model leverages the completed textual CoR as its primary guide for action generation, rather than relying on continuous, intensive visual grounding. The reasoning chain acts as a powerful intermediate representation that bridges the gap between ambiguous language and precise spatial coordinates, directly contributing to the improved accuracy shown in Table 1.

4.5. Statistical Validation

To validate the significance of our improvements over the original RoboPoint, we performed a two-sample t-test. The analysis shows that the performance gain on the primary **Where2Place (W2P) benchmark is statistically significant** ($p = 0.022 < 0.05$). This confirms that the CoR provides a meaningful improvement for identifying affordances in free space. While the improvements on the RoboRefIt ($p = 0.36$) and the challenging Where2Place (h) ($p = 0.27$) subsets were not statistically significant at the $p < 0.05$ level, our model consistently outperformed the baseline across all categories, demonstrating the general effectiveness of our approach.
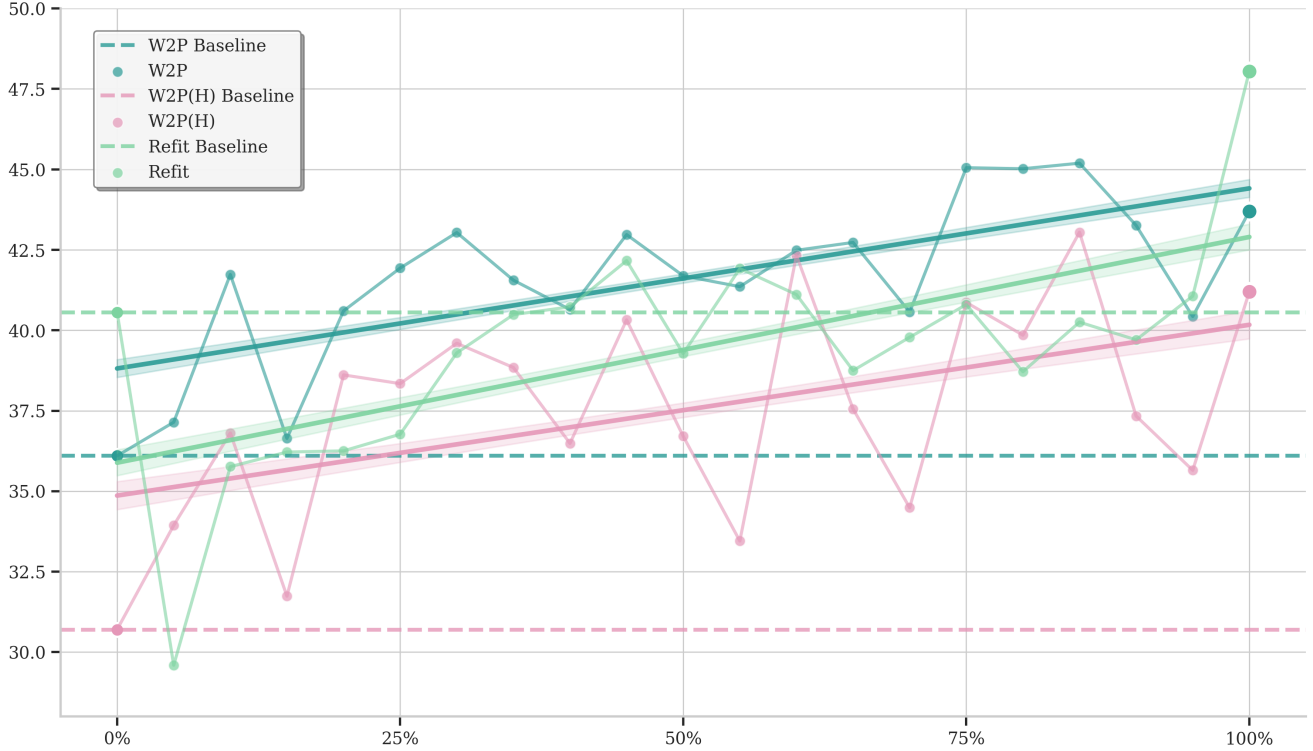
Figure 4. Ablation study on the impact of reasoning data. The plot shows the performance on the RoboRefIt, Where2Place (W2P), and Where2Place (h) benchmarks as the percentage of the TRACE reasoning dataset used for training is increased from 0% (baseline) to 100%. The solid lines represent the trend (linear regression), while the shaded areas indicate the 95% confidence interval. Performance across all tasks consistently improves with more reasoning data, providing strong evidence for the effectiveness of our approach.

5. Conclusion

In this work, we introduced TRACE, a framework that enhances spatial affordance prediction by integrating an explicit textual Chain of Reasoning (CoR). We demonstrated through comprehensive experiments that fine-tuning a VLM on our large-scale TRACE dataset enables it to generate more accurate and interpretable low-level actions. Our approach's success is supported by three key findings: 1) Our model achieves state-of-the-art performance, outperforming strong baselines with a statistically significant **9.6% relative improvement** on the primary Where2Place benchmark. 2) Our ablation study proves that these gains are directly attributable to the CoR data, as performance scales with the quantity of reasoning examples provided during training. 3) Our analysis of attention maps provides qualitative evidence of an interpretable, multi-step reasoning process, validating our method's design. Together, these results show that training for textual reasoning is a lightweight yet powerful method for improving the grounding of language in robotic actions.

5.1. Limitations and Future Work

While promising, our work has limitations. The reasoning chains in our dataset are synthetically generated and may not capture the full complexity of human thought processes. Additionally, while our attention analysis provides insight, the model lacks a mechanism to explicitly control this process or report confidence estimates for its predictions.

For future work, we plan to explore more sophisticated and less-structured reasoning generation. We also aim to extend the CoR framework to a wider range of robotic tasks, including multi-step manipulation and navigation. Finally, investigating methods to leverage the insights from attention maps to further improve model reliability represents a promising avenue for creating more capable and trustworthy VLM-driven robots.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. 1
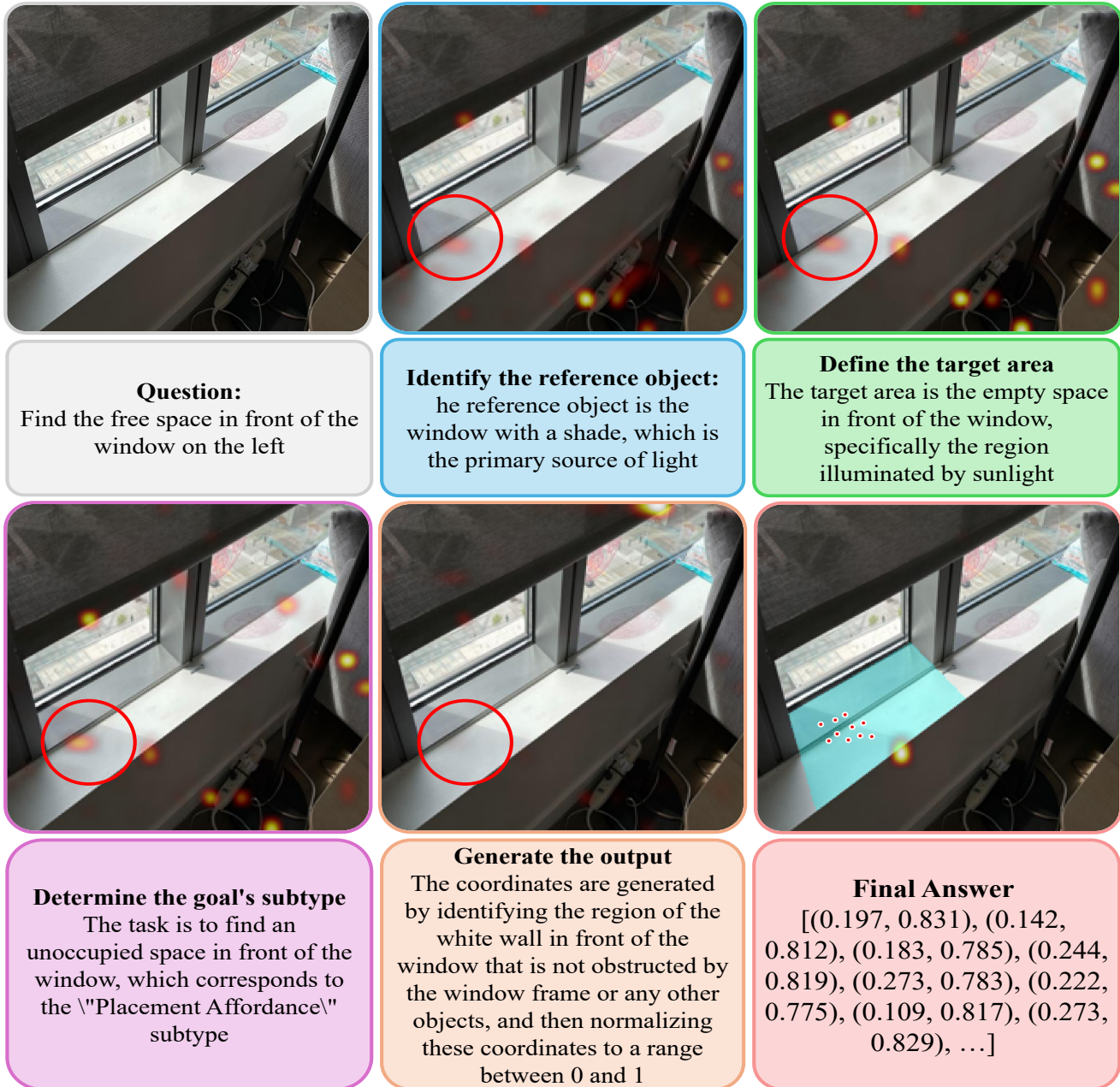
[2] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng,

Figure 5. A visualization of the model's reasoning attention map for the instruction: "Find the free space in front of the window on the left." The figure illustrates the model's focus across the four-step textual reasoning process. The attention heatmap is overlaid on the input image. (1) **Identify Reference Object** and (2) **Define Target Area**: The model exhibits diffuse, weak attention during the initial steps of identifying the window and defining the target area. (3) **Determine Goal's Subtype**: Attention begins to **focus** as the model classifies the task. A **distinct high-attention region (bright orange) emerges** over the target area, suggesting this is a critical reasoning step. (4) **Generate Output**: In the final step, there is minimal attention on the image, especially over the predicted points (shown as dots), indicating the model relies on its completed textual reasoning to generate the final coordinates rather than direct, concurrent visual evidence.

Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023. 4

[3] Jiwan Chung, Junhyeok Kim, Siyeol Kim, Jaeyoung Lee, Min Soo Kim, and Youngjae Yu. Don't look only once: Towards multimodal interactive reasoning with selective visual

revisitation. *arXiv preprint arXiv:2505.18842*, 2025. 1, 4

[4] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023. 4

[5] Xingyu Fu, Minqian Liu, Zhengyuan Yang, John Corring, Yijuan Lu, Jianwei Yang, Dan Roth, Dinei Florencio, and Cha Zhang. Refocus: Visual editing as a chain of thought for structured image understanding. *arXiv preprint arXiv:2501.05452*, 2025. 1, 3

[6] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 4

[7] Yu Hao, Geeta Chandra Raju Bethala, Niraj Pudasaini, Hao Huang, Shuaihang Yuan, Congcong Wen, Baoru Huang, Anh Nguyen, and Yi Fang. Embodied chain of action reasoning with multi-modal foundation model for humanoid loco-manipulation. *arXiv preprint arXiv:2504.09532*, 2025. 2, 3

[8] Yu Hao, Fan Yang, and Nicholas Fang. Cora: A chain of robotic actions reasoning model for autonomous robotic arm manipulation. In *2025 11th International Conference on Automation, Robotics, and Applications (ICARA)*, pages 165–169. IEEE, 2025. 2, 3

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 4

[10] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. 1

[11] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024. 1

[12] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 5, 6

[13] Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. Imagine while reasoning in space: Multimodal visualization-of-thought. *arXiv preprint arXiv:2501.07542*, 2025. 1, 3

[14] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022. 1

[15] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024. 1

[16] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023. 6

[17] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024. 4

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 4

[19] Yuhao Lu, Yixuan Fan, Beixing Deng, Fangfu Liu, Yali Li, and Shengjin Wang. Vl-grasp: a 6-dof interactive grasp policy for language-oriented objects in cluttered indoor scenes. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 976–983. IEEE, 2023. 5

[20] Mingjie Pan, Jiyao Zhang, Tianshu Wu, Yinghao Zhao, Wenlong Gao, and Hao Dong. Omnimanip: Towards general robotic manipulation via object-centric interaction primitives as spatial constraints. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17359–17369, 2025. 1

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 4

[22] Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. *Advances in Neural Information Processing Systems*, 37:8612–8642, 2024. 1

[23] Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. Robospatial: Teaching spatial understanding to 2d and 3d vision-language models for robotics. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15768–15780, 2025. 1

[24] Gemini Robotics Team, S Abeyruwan, J Ainslie, JB Alayrac, MG Arenas, T Armstrong, A Balakrishna, R Baruch, M Bauza, M Blokzijl, et al. Gemini robotics: Bringing ai into the physical world, 2025. *URL https://arxiv.org/abs/2503.20020*, 2025. 1, 6

[25] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 2

[26] Penghao Wu and Saining Xie. V?: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13084–13094, 2024. 1, 4

[27] Rongtao Xu, Jian Zhang, Minghao Guo, Youpeng Wen, Haoting Yang, Min Lin, Jianzheng Huang, Zhe Li, Kaidong Zhang, Liqiong Wang, et al. A0: An affordance-aware hierarchical model for general robotic manipulation. *arXiv preprint arXiv:2504.12636*, 2025. 1

[28] Yichang Xu, Gaowen Liu, Ramana Rao Kompella, Sihao Hu, Tiansheng Huang, Fatih Ilhan, Selim Furkan Tekin, Zachary Yahn, and Ling Liu. Language-vision planner and executor for text-to-visual reasoning. *arXiv preprint arXiv:2506.07778*, 2025. 1

[29] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024. 1, 2, 4, 5

[30] Jiarui Zhang, Mahyar Khayatkhoei, Prateek Chhikara, and Filip Ilievski. Mllms know where to look: Training-free perception of small visual details with multimodal llms. *arXiv preprint arXiv:2502.17422*, 2025. 1

[31] Enyu Zhao, Vedant Raval, Hejia Zhang, Jiageng Mao, Zeyu Shangguan, Stefanos Nikolaidis, Yue Wang, and Daniel Seita. Manipbench: Benchmarking vision-language models for low-level robot manipulation. *arXiv preprint arXiv:2505.09698*, 2025. 1

[32] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1702–1713, 2025. 1, 3

TRACE: Textual Reasoning for Affordance Coordinate Extraction

Supplementary Material

6. Additional Qualitative Results

To further illustrate the performance of our model, Figures 6 through 7 provide a qualitative comparison of **TRACE** against the **RoboPoint (Baseline)**, **SpaceLLaVA**, **GPT-4o**, and **Gemini** models. The examples are drawn from both the **Where2Place** and **RoboRefIt** benchmarks. Across these challenging scenarios, TRACE not only provides more accurate affordance points but also generates an explicit Chain of Reasoning (CoR) that makes its decision-making process transparent and interpretable. This contrasts with the baseline models, which can misinterpret spatial relations or fail to precisely locate the target region.

7. Details on Reasoning-Augmented Data Generation

As described in the main paper, our 200,000-sample TRACE dataset is a composite. This section provides details on the generation of the novel **100,000 reasoning-augmented samples**, which form our core contribution.

These samples were created by taking image-instruction pairs from the RoboRefIt and Where2Place benchmarks (50,000 from each) and using a large language model to generate a detailed Chain of Reasoning (CoR) for each one.

The reasoning chains were generated using the **Gemini 2.5 Flash** model ('gemini-2.5-flash-lite-preview-06-17') via its API. We prompted the model to produce a step-by-step rationale that logically connects the visual context and the language instruction to the final selection of ground-truth points. The prompt was structured to deconstruct the task into distinct cognitive steps, including: 1) Identifying reference objects in the scene, 2) Determining the goal's subtype (e.g., Placement Affordance), 3) Defining the specific target area, and 4) Explaining how the final points were generated within that area.

This process, which took approximately 40 hours of API computation time, resulted in the rich, interpretable data format shown in Figure 1 of the main paper. This 100k subset, when combined with 100k standard instruction-following samples from VQA and LVIS, comprises the full training dataset used to fine-tune TRACE.

8. TRACE Dataset Generation Details

The TRACE dataset was created to explicitly teach Vision-Language Models to generate a textual Chain of Reasoning (CoR) before predicting spatial affordances. The generation process involved leveraging a powerful large language model to create detailed rationales for existing image-instruction pairs.

First, we randomly sampled a total of 100,000 examples, drawing 50,000 from the RoboRefIt benchmark and 50,000 from the Where2Place benchmark. For each sampled image, instruction, and ground-truth affordance point set, we programmatically generated a corresponding reasoning chain.

The reasoning chains were generated using the **gemini-2.5-flash-lite-preview-06-17** model via its API. We prompted the model to produce a step-by-step rationale that logically connects the visual context and the language instruction to the final selection of points. The prompt was structured to deconstruct the task into distinct cognitive steps, including: 1) Identifying reference objects in the scene, 2) Determining the goal's subtype (e.g., Placement Affordance), 3) Defining the specific target area, and 4) Explaining how the final points were generated within that area.

This process, which took approximately 40 hours of API computation time, resulted in the rich, interpretable data format shown in Figure 1 of the main paper. This dataset of 100,000 reasoning-augmented samples forms the core of the training data used to fine-tune TRACE.

9. TRACE Dataset Examples

To provide a clearer view of our training data, Figures 8 and 9 show sample data points from the TRACE dataset. Each example includes the input instruction, the synthetically generated Chain of Reasoning, and the ground-truth coordinates used for fine-tuning our model.

Question : Find a few locations in the free space that is in front of the steamer.

Ground Truth : From dataset masks

TRACE
Reasoning Process:
1. Identify the reference object : The user's request specifies a steamer. Visually, this refers to **the white and blue appliance** with a handle, located on the right side of the image.
2. Identify the target area : The instruction asks for locations in the free space that is **in front of the steamer**. This means the search area is the empty space directly in front of the steamer, on the same surface.
3. Define the action : The task is to find points within this identified **free space in front of the steamer**.
4. Explain the output format : The coordinates of these points are then normalized to values between 0 and 1 to generate the final answer.
Final Answer : [(0.613, 0.783), (0.595, 0.856), …]

Robotpoint
Final Answer : [(0.720, 0.865), (0.328, 0.867), …]

SpaceLLaVA
Final Answer (bbox) : (0.0, 0.67, 0.14, 0.87)

gpt-4o
Final Answer (bbox) : (0.531, 0.583, 0.656, 0.729)

gemini-2.5-flash-lite
Final Answer (bbox) : (0.39, 0.34, 0.57, 0.57)

Question : Pinpoint some points within the vacant space that is in between the phone and measuring tape.

Ground Truth : From dataset masks

TRACE
Reasoning Process:
1. Identify the reference object : The user's request specifies a reference object, which is **the wooden desk**.
2. Determine the goal's subtype : The instruction asks for points within the vacant space in between the phone and the measuring tape. This indicates the goal is **to find an independent empty space**, classifying it as Placement Affordance.
3. Define the target area : The target area is the empty space located on the desk surface, specifically **between the phone and the measuring tape**.
4. Generate the output : The coordinates of points within this vacant space are then normalized to produce the final answer.
Final Answer : [(0.634, 0.621), (0.619, 0.617), …]

Robotpoint
Final Answer : [(0.705, 0.542), (0.689, 0.544), …]

SpaceLLaVA
Final Answer (bbox) : (0.44, 0.66, 0.62, 0.87)

gpt-4o
Final Answer (bbox) : (0.375, 0.625, 0.5, 0.75)

gemini-2.5-flash-lite
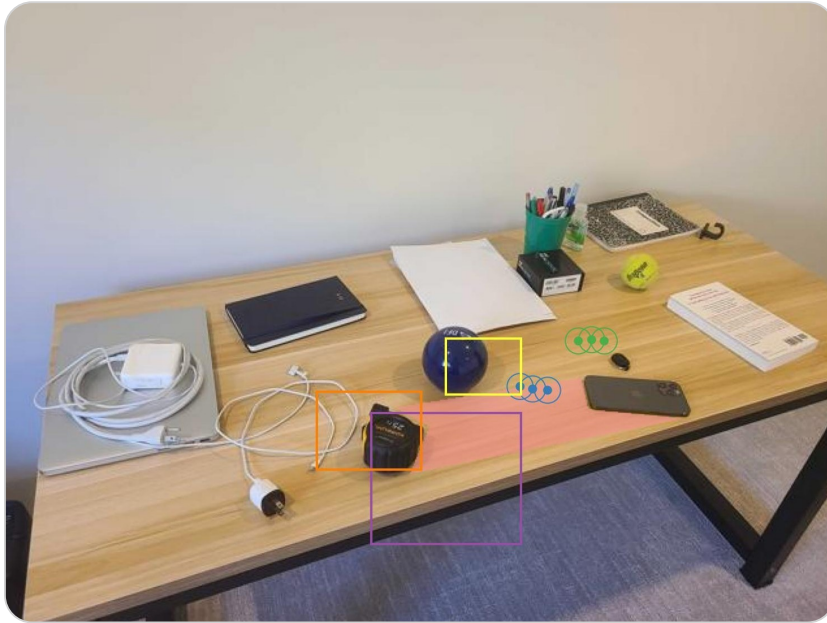**Final Answer (bbox)** : (0.53, 0.54, 0.62, 0.63)

Figure 6. Qualitative results on the **Where2Place** benchmark. These examples showcase tasks that require identifying vacant space based on complex relational instructions. In both cases, TRACE's explicit Chain of Reasoning helps it correctly interpret the spatial relationship and generate precise affordance points. In contrast, baseline models either misinterpret the relation or provide less accurate localization.

Question : Locate several points within the area related to 'would you please hand me the rubik's cube.

Ground Truth : From dataset masks

TRACE
Reasoning Process:
1. Identify the reference object : The user's request specifies a reference object related to "would you please hand me the rubik's cube". In the image, **the rubik's cube is a colorful, multi-faceted object**.
2. Identify the target object : The instruction asks for points within the area related to 'would you please hand me the rubik's cube'. This means we need to find points **on the surface of the rubik's cube** itself.
3. Define the action : The task is to locate several points on the surface of the rubik's cube, specifically **on the colored faces of the cube**.
4. Explain the output format : The coordinates of these points are then normalized to values between 0 and 1 to generate the final answer.
Final Answer : [(0.692, 0.769), (0.673, 0.762), …]

Robotpoint
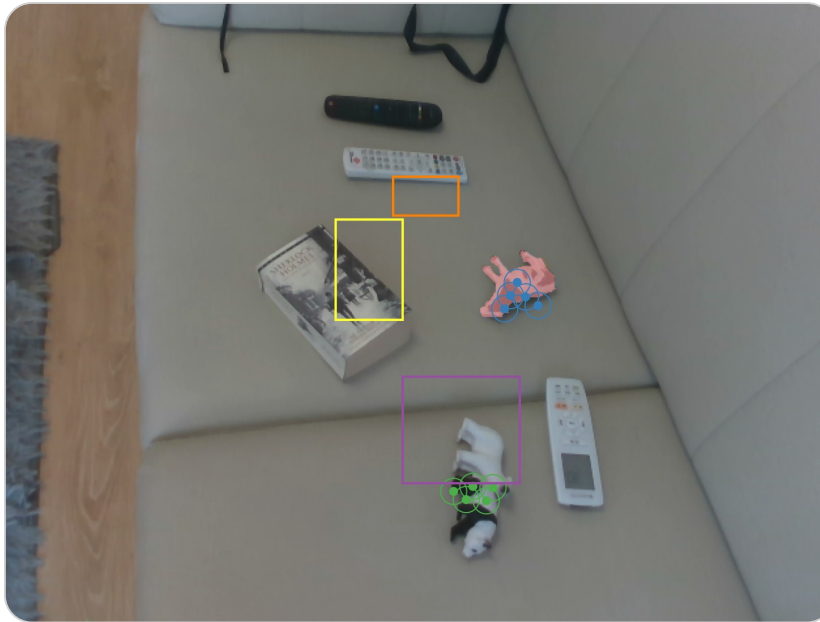**Final Answer :** [(0.466, 0.673), (0.442, 0.671), …]

SpaceLLaVA
Final Answer (bbox) : (0.38, 0.56, 0.57, 0.86)

gpt-4o
Final Answer (bbox) : (0.55, 0.6, 0.60, 0.65)

gemini-2.5-flash-lite
Final Answer (bbox) : (0.42, 0.32, 0.53, 0.44)

Question : Locate several points within the area related to 'could you fetch me the black and white toy cow

Ground Truth : From dataset masks

TRACE
Reasoning Process:
1. Identify the reference object : The user's request specifies a reference object, which is **the black and white toy cow.**
2. Define the target area : The instruction asks to locate points within the area related to the toy cow. This means we need to find points **on the surface of the couch** where the toy cow is situated.
3. Determine the goal's subtype: The request asks for points within the area related to the toy cow. This indicates a Placement Affordance subtype, as we are looking for a location on the **surface where the toy cow is placed**.
4. Generate the output : The coordinates of points on the couch surface, specifically on the area where the toy cow is located, are then normalized to produce the final answer.
Final Answer : [(0.602, 0.492), (0.627, 0.473), …]

Robotpoint
Final Answer : [(0.580, 0.798), (0.556, 0.796), …]

SpaceLLaVA
Final Answer (bbox) : (0.48, 0.60, 0.62, 0.77)

gpt-4o
Final Answer (bbox) : (0.469, 0.251, 0.547, 0.344)

gemini-2.5-flash-lite
Final Answer (bbox) : (0.40, 0.35, 0.48, 0.51)

Figure 7. Qualitative results on the **RoboRefIt** benchmark, which tests the ability to ground language instructions to specific objects in cluttered environments. TRACE successfully uses its reasoning process to identify the target objects and predict accurate affordance points. The comparison highlights how the textual rationale aids in disambiguating objects and improves the precision of the final output compared to baseline methods.

"from": "human"
"value": "<image>
The image features an item encased within a red rectangular border. Pinpoint a few locations in the vacant area that is in front of the object highlighted by the rectangle. Your answer should be formatted as a list of tuples, i.e. [(x1, y1), (x2, y2), ...], where each tuple contains the x and y coordinates of a point satisfying the conditions above. The coordinates should be between 0 and 1, indicating the normalized pixel locations of the points in the image.
"from": "gpt"
"value": "**Reasoning Process:**
Identify the reference object: The user has highlighted an object with a red border, which is a pair of white high-heeled shoes.
2. **Define the target area:** The request asks for a vacant area that is in front of the reference object. This means the search space is the floor area directly ahead of the shoes.
Determine the goal's subtype: The request is to find a vacant area in front of the object, not on the object itself. This aligns with the \"Placement Affordance\" subtype, as it's about finding a place *near* the object.
Generate the output: The coordinates of points located on the floor in front of the shoes are then normalized to produce the final answer.
Final Answer:
[(0.664, 0.650), (0.695, 0.644), (0.648, 0.623), (0.769, 0.606), (0.723, 0.617), (0.753, 0.613), (0.672, 0.623), (0.706, 0.623), (0.795, 0.602), (0.816, 0.600), (0.680, 0.646), (0.688, 0.625)]"

"from": "human"
"value": "<image>
The image features an item encircled by a red rectangle. Locate a few spots within the vacant area situated at the rear of the encircled item. Your answer should be formatted as a list of tuples, i.e. [(x1, y1), (x2, y2), ...], where each tuple contains the x and y coordinates of a point satisfying the conditions above. The coordinates should be between 0 and 1, indicating the normalized pixel locations of the points in the image.
"from": "gpt"
"value": "**Reasoning Process:**
1. **Identify the reference object:** The user has encircled an item with a red rectangle. This item is a blue bowl.
2. **Determine the goal's subtype:** The request asks to find points in a \"vacant area situated at the rear of the encircled item.\" This indicates a search for an empty space that is not part of the item itself but is located behind it. Therefore, this aligns with the 'Placement Affordance' subtype.
3. **Define the target area:** The goal is to find an empty space located behind the blue bowl. This means we need to search the area on the black surface behind the bowl.
4. **Generate the output:** The coordinates of a point on this vacant surface, behind the blue bowl, are then normalized to produce the final answer.
Final Answer:
[(0.750, 0.858)]"

Figure 8. This image contains two data examples from the **Free Space Reference** dataset, with each example showing the prompt, the model's reasoning, and the final coordinate output.
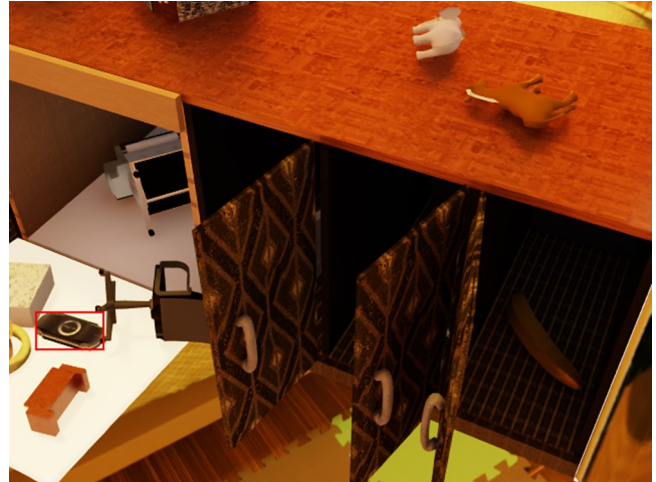
"from": "human"
"value": "<image>
The image features an area outlined by a red rectangle. Locate several points on an item that is placed on top of the marked area. Your answer should be formatted as a list of tuples, i.e. [(x1, y1), (x2, y2), ...], where each tuple contains the x and y coordinates of a point satisfying the conditions above. The coordinates should be between 0 and 1, indicating the normalized pixel locations of the points in the image.
"from": "gpt"
"value": "**Reasoning Process:**
1. **Identify the reference area:** The user's request specifies a red rectangular outline in the image. This outline clearly delineates the top surface of the white counter.
2. **Identify the target item:** The request asks to locate points on an \"item\" that is \"placed on top of the marked area.\" Visually, a white spherical object with a green circular marking is situated directly on this marked counter surface. This object is therefore the target.
3. **Determine the points to locate:** The request asks for several points on the surface of this target object. The provided coordinates represent specific locations on the surface of the white sphere.
4. **Explain the output format:** The coordinates are normalized pixel locations, ranging from 0 to 1, which is the required output format for specifying points on the object
Final Answer:
[(0.325, 0.179), (0.327, 0.131), (0.284, 0.152), (0.300, 0.208), (0.294, 0.110), (0.353, 0.154)]"

"from": "human"
"value": "<image>
The image features an item encircled by a red rectangle. Locate several points on an item that is situated behind the encircled item. Your answer should be formatted as a list of tuples, i.e. [(x1, y1), (x2, y2), ...], where each tuple contains the x and y coordinates of a point satisfying the conditions above. The coordinates should be between 0 and 1, indicating the normalized pixel locations of the points in the image.
"from": "gpt"
"value": "**Reasoning Process:**
1. **Identify the reference object:** The user's request encircles an item with a red rectangle. This encircled item is identified as a black cell phone, located on a white surface.
Identify the target object: The instruction asks for points on an item \"situated behind\" the encircled cell phone. Visually inspecting the area behind the cell phone, there is a dark, patterned object. This object is identified as a cabinet door with a distinctive diamond pattern and a handle.
3. **Define the action:** The task requires locating multiple points on the surface of this identified cabinet door.
Explain the output format: The coordinates for these points are then normalized to values between 0 and 1 to generate the final answer.
Final Answer:
[(0.256, 0.325), (0.228, 0.356), (0.181, 0.335), (0.277, 0.394), (0.209, 0.433), (0.169, 0.377), (0.239, 0.415), (0.244, 0.285), (0.214, 0.308), (0.184, 0.294), (0.202, 0.390), (0.217, 0.481), (0.150, 0.340)]"

Figure 9. This image contains two data examples from the **Object Reference** dataset, with each example showing the prompt, the model's reasoning, and the final coordinate output.