VeriGraph: Scene Graphs for Execution Verifiable Robot Planning

Daniel Ekpo, Mara Levy, Saksham Suri, Chuong Huynh, Abhinav Shrivastava

***Abstract*— Recent advancements in vision-language models (VLMs) offer potential for robot task planning, but challenges remain due to VLMs' tendency to generate incorrect action sequences. To address these limitations, we propose VeriGraph, a novel framework that integrates VLMs for robotic planning while verifying action feasibility. VeriGraph employs scene graphs as an intermediate representation, capturing key objects and spatial relationships to improve plan verification and refinement. The system generates a scene graph from input images and uses it to iteratively check and correct action sequences generated by an LLM-based task planner, ensuring constraints are respected and actions are executable. Our approach significantly enhances task completion rates across diverse manipulation scenarios, outperforming baseline methods by 58% for language-based tasks and 30% for image-based tasks.**

I. Introduction

For robots to be able to solve complex manipulation problems in the real world, they need to understand the physical world around them, including object locations and relationships between objects in the scene. Humans intuitively understand spatial relationships between objects in the world and can use this understanding to develop efficient and executable plans to complete tasks. Consider the example of organizing a cluttered room. Humans can quickly understand which objects are out of place based on their understanding of how objects are supposed to relate to each other. For example, it seems intuitive that a book should be on a shelf, not on a cup. Robots struggle to perceive the world around them the way humans do.

Additionally, physical constraints in the real world restrict the order in which actions can be executed. For example, if a glass cup is on a book, which is on a desk, the robot must pick up the cup first and place it on the desk before picking up the book. Because of these constraints, robots need to understand the relationships between objects in the scene. If the robot does not understand that the cup is on the book, it might not factor that into its planning and may try to pick up the book first, which could result in the cup falling and breaking.

Recent advances in large language models (LLMs) and vision-language models (VLMs) have opened up new possibilities for robot task planning [1, 2]. These models demonstrate impressive reasoning capabilities and world knowledge. Prior work [3–5] used LLMs to generate Planning Domain Definition Language (PDDL), which can be used by classical planners to create a task plan. While the results have been promising, PDDL is inherently restrictive and does not generalize well [6, 7]. Other lines of work use VLMs to generate high-level task plans directly using images [8–10]. While the results are remarkable, relying on raw pixel data for complex manipulation can be suboptimal. Pixel-level information is often noisy and may contain extraneous details irrelevant to the high-level planning task.

To address the scene representation issue, we propose VeriGraph, a novel approach that utilizes scene graphs as an intermediate representation for robot task planning. Scene graphs have proven particularly valuable in robotic task planning [11–14]. Their structured nature enables the abstraction of object-level details into symbolic graphs, making them robust to noise while retaining essential information about object interactions. For example, a scene graph might encode that a "cup is on the table" or a "spoon is inside the cup," providing a framework for reasoning about actions such as moving the spoon or rearranging the objects in the scene. This abstraction is especially important for tasks where the physical appearance of individual objects is not important. One such task is using a reference scene to arrange objects in another scene to look like the reference scene. Because of the scene graph representation, VeriGraph can solve this task significantly better than methods that rely on raw pixel data.

Despite their powerful nature, VLMs are prone to failures in planning, often requiring multiple iterations of prompting to achieve the correct result [15]. Approaches such as [10] attempt to solve this problem by inserting a human directly into the loop. However, this is time-consuming and requires constant human supervision. To address the plan verification and correcting problem we add an iterative planning component to VeriGraph. The structured nature of scene graphs allows VeriGraph to represent each action in the plan as graph operations. For example, moving an object from one location to another can be represented with an edge manipulation operation. This representation allows VeriGraph to quickly check for constraint violations and iterate with the task planner to generate valid action sequences. This setup, shown in Figure 2, allows for more accurate and robust planning.

For some tasks, it might be easier to specify the goal state via language instructions, while for some tasks a reference image is sufficient. To be able to support both types of task specification, VeriGraph supports flexible goal specification, allowing manipulation tasks to be defined through either target scene images or natural language instructions. The system can generate goal scene graphs from these inputs, providing a unified planning framework across different task specifications. Note that for reference images, VeriGraph does not require the exact same scene, only contextually similar scene (e.g., refer to Figure 1). This versatility makes VeriGraph applicable to various real-world scenarios where

University of Maryland, College Park, MD USA. Correspondence to: Daniel Ekpo daniekpo@umd.edu

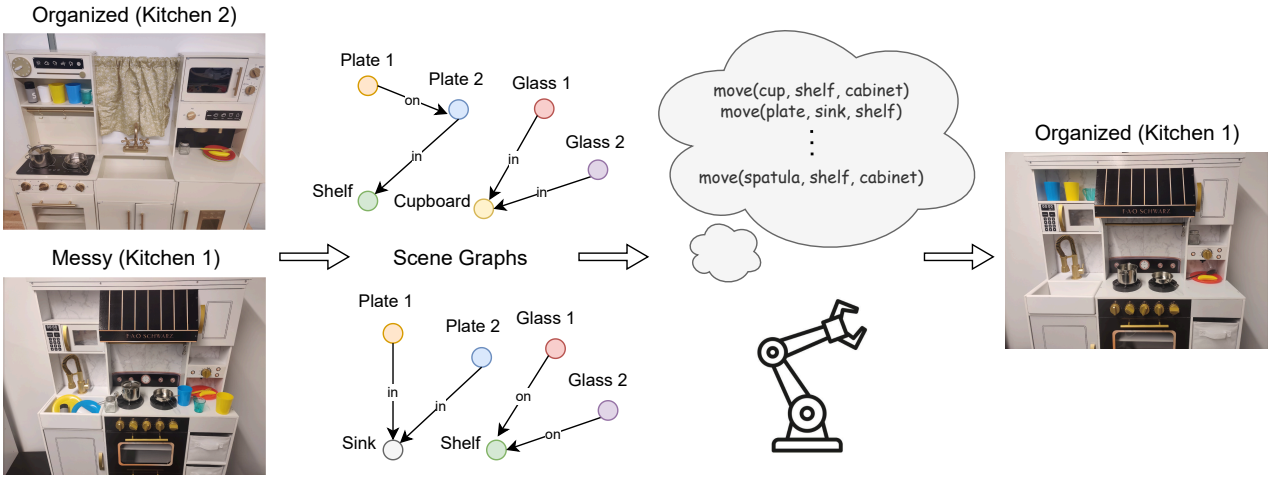

Fig. 1: VeriGraph is able to utilize an initial scene image and a reference image which may or may not be from the same setting. Using the two images, our approach generates the corresponding scene graphs. Using a VLM as the planner along with execution-verifiability, we generate and execute a plan using the robot.

goals may be communicated in different formats.

We show that VeriGraph beats existing methods that rely on raw pixels as input to the task planner while being execution-verifiable. Our main contributions are as follows:

- We present a modular and fast approach that uses scene graphs to enhance planning with LLMs, improving the understanding of spatial relationships and constraints.
- We propose an iterative planning and verification mechanism that uses scene graphs to represent and verify action sequences, enhancing the system's ability to identify and correct constraint violations without human intervention.
- We utilize VLMs to generate goal scene graphs based on a reference image or language instruction to create a unified goal specification method.

II. Related Works

A. Scene Graph in Planning

Scene graphs have been used in computer vision for symbolic representation. Scene graphs can represent object relationships in images [16]. These representations have been used for different tasks like image generation [17–22], image/video captioning [23–25], and visual question answering [26–28]. Scene graphs can encode useful scene information without getting affected by pixel-level noise. This has increased the use of scene graphs in robotics. [29] propose a graph representation of the robot's environment for navigation by representing nodes as semantic places and edges as navigational behaviors. [30] use 3D scene graphs to train a reinforcement learning algorithm policy for navigation. They maintain a graph that encodes information about the observed scene and use a graph neural network (GNN) to encode graph features which are passed to the policy to predict a distribution over the action space. GRID [31] uses an existing scene graph generator to get an initial graph which is passed to a VLM encoder that acts as an 'instructor'. A separate robot graph is passed to the encoder which represents the robot's state. These are then combined through GNNs to predict actions. [32] generate a scene graph using an object detector and use the scene graph for cluttered scene exploration for question answering and planning. SARP [13] uses multi-view images to generate scene graphs for robot planning using a partially observable Markov decision process solver. [11] uses an object detector and pose estimator to create geometric and symbolic graphs and uses a graph neural network and symbolic reasoning for motion planning. SG-Bot [33] uses scene graphs to imagine the goal scene for a reconstruction task. [34] uses contact graphs and Graph Edit Distance for planning. Our work is similar to theirs in how we model actions as graph operations. However, we use an LLM as the task planner, which gives us much more flexibility and allows us to use natural language instructions to describe the tasks. An LLM planner also provides better generalization over multiple tasks, objects, and actions.

B. Planning with LLMs/VLMs

LLMs [35–37] and VLMs [38–40] have emerged as strong agents for open world reasoning and have shown good understanding of the real world. Due to their good reasoning and understanding of object relations, actions, and context have also been applied to robotics as agents to help solve tasks, especially planning.

On the side of LLMs, multiple works have explored them as planners [41, 42]. LLMs have also been used for embodied agents to generate plans that can be executed [8, 43, 44]. While [43] is similar to our work in the sense that we provide language feedback to the model, our work differs in the sense that our original input to the model is a scene graph, and we continue to use only the scene graph as feedback to the planner, reducing the amount of data and computation required. SayCan [8] demonstrate good results using their approach. However, using a learned value function to verify the output requires training the value function and can require a lot of data and computation. ConceptGraphs [12] fuse the output of 2D foundation models to generate a 3D scene graph and use an LLM to generate plans based on the 3D scene graph.

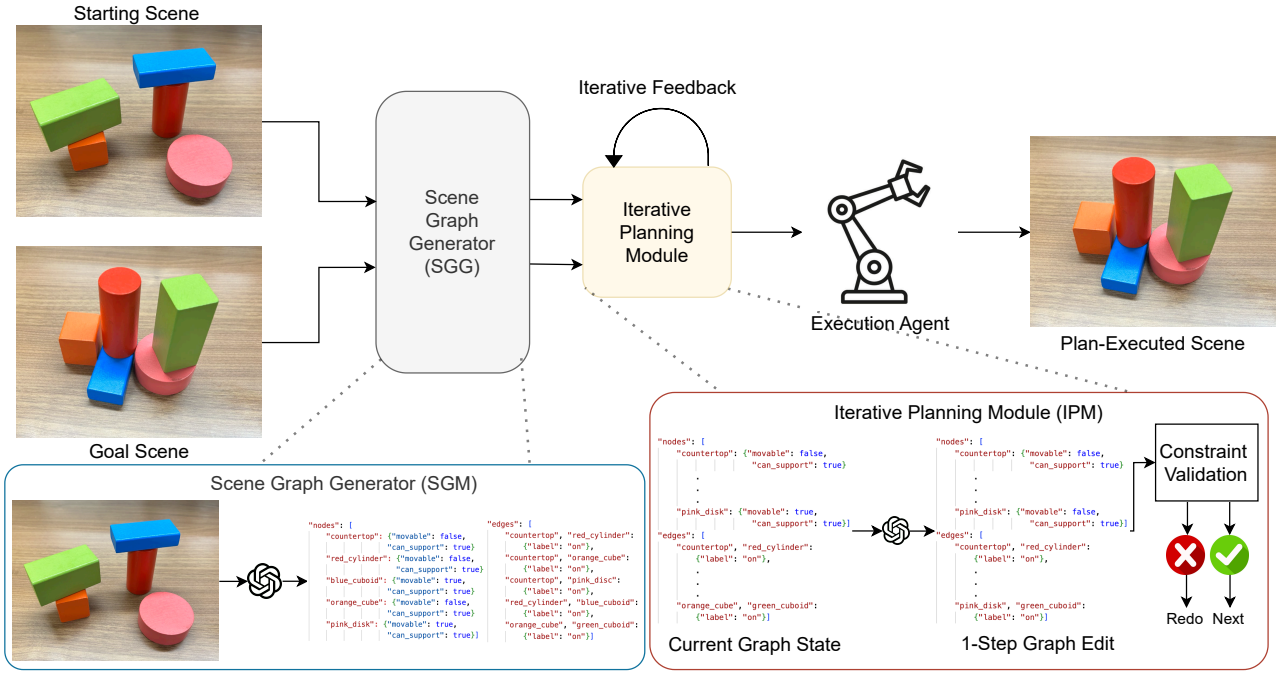

Fig. 2: Overview of VeriGraph. Two images are input: the start scene (current state) and the goal scene (desired state). A scene graph generator extracts objects and relationships from each image, which are then processed by the iterative planning module. This module evaluates suggested actions from the VLM, checking for constraint violations. If a violation occurs, the VLM suggests a new action; if not, the action is executed. This loop continues until the environment matches the goal scene.

Recent VLM developments have made them an even better candidate for robotics since they can take visual data and raw pixels as input, unlike LLMs, which can only work with text. This allows them to perceive the scene as is without being heavily dependent on the input prompt. Recently, RT-2 [45] incorporated VLMs directly into end-to-end robotic control. PaLM-E [46] trained a large model specifically as an agent that can take multimodal inputs and perform robotics tasks. ViLa [9], which is the closest to our work, uses a large VLM directly for their task. While they show good results, we compare with them and show how directly looking at pixels might be sub-optimal for robot planning. Instead, our approach uses an intermediate scene graph representation to make the planning verifiable and more accurate, thus showing improved performance.

C. *Execution-Verification*

While correctness is important when designing a task execution plan, it is also essential for it to be plausible. SayPlan [14] assumes an existing 3D scene graph, which they use to interact with the LLM. They use a graph simulator to verify the LLM-generated task plans and show that their approach works for multi-room setups. CoPAL [47] proposes corrective planning by using different layers of encapsulation. Some works [48] modify an existing reinforcement learning algorithm to condition on natural language feedback from the environment. They automatically generate the language feedback based on the current goal and the agent's current actions. This is similar to our approach of providing feedback to the planner based on the current graph state and predicted action. REFLECT [49] introduces a framework to query the LLM planner to reason about failures based on the hierarchical summary of the robot's past experiences generated from multisensory observations. They show that the failure explanation can help the LLM correct the failure and complete the task. Voyager [50] introduces an LLM learner that can learn executable skills as it interacts with the environment. It writes code to interact with the environment and correct itself with feedback received from the environment. ViLa [9] uses execution to verify the plan by feeding the current state of the environment to the model at every step using visual inputs. Our approach, on the other hand, can generate execution-verifiable plans by relying on the current scene graph for constraint checking. This makes verifying the affordances and plausibility of an action is especially quick and efficient.

III. Our Approach: VeriGraph

VeriGraph takes in an image depicting an initial scene, along with either a target image portraying the desired goal scene state or instructions detailing modifications to the initial scene. It generates the initial and goal scene graphs and uses them to predict actions to transform the initial scene into the target scene. The scene graph generation method is discussed in Section III-A, and action constraints are discussed in Section III-B. Given a pair of scene graphs, the planner, discussed in Section III-C and Section III-D, generates a high-level plan instructing a robot to transform the initial scene into the goal scene.

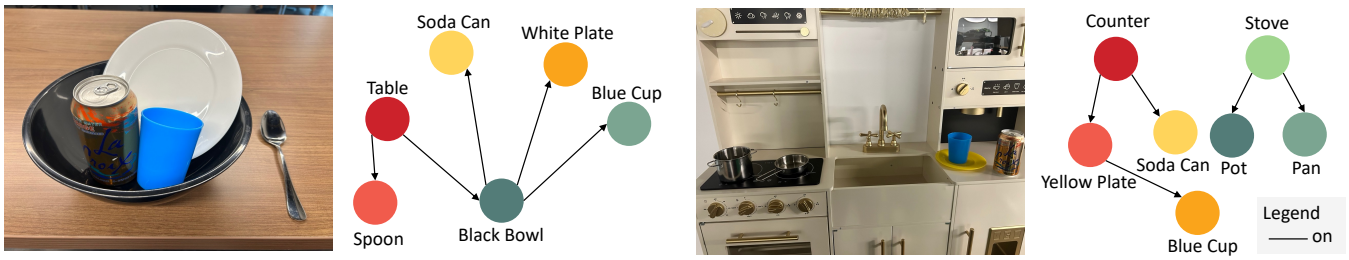

Fig. 3: An example of how the scene graphs are structured for individual images. First, nodes are created for each object in the image, and then edges are added to represent the relationships between different objects. The relationship is represented with a solid line. Relationships are directional and go towards the object that is "on" another object. This is so objects on top are represented as leaf nodes and are not blocked from moving.

A. Scene Graph Generation

Given an image I of a scene, the goal of scene graph generation is to create a graph that accurately represents the scene's structure. The scene graph comprises a set of vertices V, representing objects in the scene, and a set of edges E, describing the relationships between objects in V. R represents a set of possible relations between objects in the scene. An edge $e_{uv} \in E$ between two vertices u and v in the scene is then defined as $e_{uv} = \{u, v, r\}$ where $r \in R$ and $u, v \in V$. The scene graph for image I is thus represented as $G = \{V, E\}$.

In VeriGraph, the set of relations R includes basic spatial relations such as {in, on}. However, this set is flexible and easily adapted for other tasks. To address the issue of varying object names (e.g., tabletop, table, and countertop for the same object), we maintain a global dictionary of unique object names D for scene graph generation. This dictionary encompasses all objects that could be present in any scene. The scene graph generator SGG takes the image I, the global dictionary D, the set of relations R and the task description T and returns a scene graph. We define the graph generator as

$$SGG(I, R, D, T) \rightarrow G = \{V, E\} \tag{1}$$

When generating scene graphs, T is set to null except for the target scene graph for tasks where the target scene is described using natural language, in which case the task instruction/goal scene description T is used in the graph generation process. VeriGraph uses SGG to generate the initial and goal scene graphs. An example scene graph is shown in Figure 3.

B. Constraint Validation

Every action has a set of preconditions that must be met before execution. For instance, before performing the "move" action on a plate, any items on the plate must first be removed. Additionally, post-conditions must be satisfied for the action to be considered successful. In the example given, the plate must end up on the new supporting object. In VeriGraph, these conditions are represented as a set of constraints $\mathcal{C}$.

VeriGraph uses the current graph to validate constraints. The vertex v associated with the action must exist in the graph, and its in/out edges must satisfy specific conditions. For instance, if v is being moved, VeriGraph checks if v supports any other objects by ensuring no edges from v to any other nodes exist.

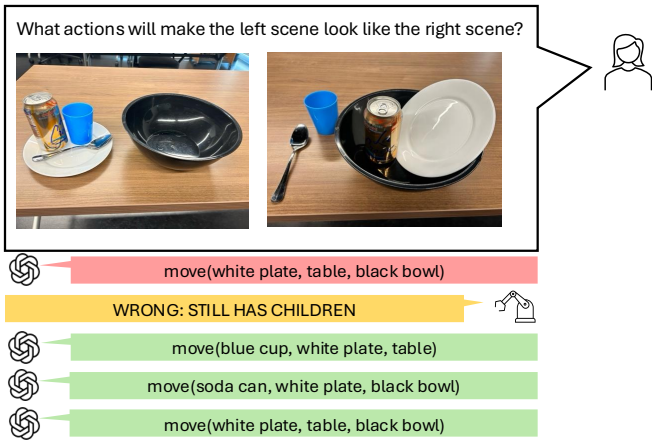

Fig. 4: Iterative planning: The planner suggests the first action. Our model detects that the plate cannot be moved due to objects on top and requests a new plan. The planner responds with a better action, continuing until the task is complete.

After constraints are validated, VeriGraph updates the current graph state to reflect the execution of the action. The specific changes to the graph depend on the action taken. For a "move" operation, the edge representing the initial support relationship is removed, and a new edge is created for the new support relationship. The next action in the sequence is then validated, and the graph is modified accordingly. The final graph's nodes and edges are compared against the goal scene graph. If they match, the plan is considered successful.

C. Task Planning

Given the initial scene graph G_{init} and the target scene graph G_{tgt}, the task planner $\mathcal{P}$ generates actions that can be executed on the initial scene to transform it into the target graph. Let $\mathcal{A}$ be the set of all high-level actions that a robot can perform, the sequence of actions $a = \{a_1, a_2, \cdots, a_n\}$ such that $a \in \mathcal{A}$ predicted by the planner $\mathcal{P}$ must complete the given task while adhering to the constraints $\mathcal{C}$. We define the task planner as

$$a = \mathcal{P}(G_{\text{init}}, G_{\text{tgt}}, \mathcal{C}, \mathcal{A}). \tag{2}$$

An example of such a plan can be seen in Figure 4.

D. Iterative Planning

The planner described in Section III-C outputs the full action sequence without any feedback mechanism to refine the plan. Our experiments showed that this approach often fails in difficult tasks because LLMs tend to forget constraints. To address this issue, we designed an iterative planner $\mathcal{P}_{\text{iter}}$ that receives feedback $\mathcal{F}$ about the proposed action sequences and corrects the plan accordingly.

The planner, $\mathcal{P}_{\text{iter}}$, is given $\mathcal{A}$, $\mathcal{C}$, G_{init}, and G_{tgt} and asked to output at most k high-level actions and an end token. VeriGraph attempts to perform the actions, and VeriGraph returns feedback, $\mathcal{F}$, as well as the current graph state. If there is a constraint violation in the proposed actions, the error count τ is increased. The feedback and new graph state are then passed to $\mathcal{P}_{\text{iter}}$, and the iteration continues until either the number of errors reaches the error threshold $\mathbf{t}$ or the end token is received. The iterative planning process is described in Algorithm 1.

Algorithm 1 Iterative Planning Algorithm

Input: G_{init}, G_{tgt}, $\mathcal{C}$, $\mathcal{A}$, $\mathcal{F}$
Output: $a = \{a_1, \cdots, a_n\}$
Initialize current graph: $G_{\text{current}} \leftarrow G_{\text{init}}$
Initialize error count: $\tau \leftarrow 0$
while $\tau < \mathbf{t}$ **do**
  a, end_token $\leftarrow \mathcal{P}_{\text{iter}}(G_{\text{current}}, G_T, \mathcal{C}, \mathcal{A}, \mathcal{F})$
  if end_token **then**
    break
  end if
  $G_{\text{current}}, \mathcal{F} \leftarrow$ execute actions a
  if $\mathcal{F}$ has constraint violation **then**
    $\tau \leftarrow \tau + 1$
  end if
end while

IV. Experimental Details

In this section, we outline the details of our experiments. The evaluation dataset is introduced in Section IV-A, followed by task description in Section IV-B. All baselines are mentioned in Section IV-C and finally Section IV-D discusses the results.

A. Dataset

We design three scenes—kitchen, tabletop, and block scenes—to evaluate VeriGraph's performance and compare against baseline methods. Each scene has multiple configurations with varying numbers of objects and placements. We vary the number of objects between three and seven. Ground truth scene graphs for each scene were created using GPT-4V(ision) (GPT-4V) [51], and corrections were made manually when necessary. Figure 5 shows some example images from the dataset.

B. Tasks

We created three task groups - rearrange, language instruction, and stacking - with varying difficulty levels. Each task has a ground truth goal scene graph for evaluating the planner. The predicted actions from the planner are executed on the initial scene graph, and the transformed graph is then compared against the ground truth goal scene graph. The different task groups are described below.

Fig. 5: Example scenes from the evaluation dataset; (top) blocks, (middle) kitchen, and (bottom) tableware scene.

TABLE I: Results on scene graph generation.

Scene	Method	F1 Score		Accuracy
		Nodes	Edges	
Blocks	LLava	0.30	0.07	0.00
	Gemini	**1.00**	0.93	0.73
	GPT-4V	**1.00**	**1.00**	**1.00**
Kitchen	LLava	0.50	0.05	0.00
	Gemini	0.78	0.59	0.04
	GPT-4V	**0.98**	**0.87**	**0.65**
Tableware	LLava	0.52	0.16	0.00
	Gemini	0.95	0.90	0.39
	GPT-4V	**0.99**	**0.95**	**0.89**

Stacking Task. We use block scenes of different configurations for the stacking task. The model is given an image of the initial scene and asked to stack all the blocks into one stack. The final order is arbitrary here, so there's no ground truth scene graph. Instead, the final scene graph is checked to ensure a single pile of blocks. Some block scenes already have multiple incomplete stacks, so the planner must unstack the other stacks and complete one.

Language Instruction Task. This consists of an initial scene and a language instruction. The instruction is either direct commands, e.g., "Move pan to the stovetop," or a description of the desired goal state of the scene, e.g., "I need the positions of the pan and pot swapped." The model is asked to predict actions to execute the given instruction on the scene.

Reference Image Instruction Task. The model is given an initial scene and a structurally similar scene as the goal state and asked to predict a sequence of actions to transform the initial scene into the goal scene. The scene graph of the goal scene is used as the goal scene graph.

C. Baselines

We compare VeriGraph against the following baselines: (a) **ViLa** [9]. prompts the vision-language model (VLM) with an image and a language instruction or another image

TABLE II: Planning results for different scenes. We report the success rate, where success is when the task was completed.

Task	SayCan	ViLa	Ours (Direct)	Ours
Stacking	0.07	0.62	0.35	**0.65**
Language Instruction	0.17	0.43	**0.73**	0.65
Ref. Image Instruction (Blocks)	-	0.27	0.67	**0.86**
Ref. Image Instruction (Kitchen)	0.00	0.05	0.50	**0.55**

as the goal state. For a fair comparison, we use a prompt identical to ours and remove all references to scene graphs. We execute the proposed actions on the ground truth initial scene graph and compare the final graph against the ground truth graph. (b) **SayCan** [8] prompts the LLM with a textual representation of the scene. The text contains a list of all the objects in the scene. We implemented a similar setup using GPT-4 as the LLM. Since SayCan cannot understand the spatial relationships between objects in the scene because it only receives a list of all objects, we only evaluate it on image-language tasks. The objects from vertices in the ground truth scene graph are the scene observation for this baseline.

D. Experiments and Results

Scene graph generation: As mentioned earlier, VLMs are utilized in our scene graph generation model. Here, we evaluate some popular VLMs, such as Gemini 1.5 Pro [52], LLava [53], and GPT-4V [51], without any in-context examples. For GPT-4V and Gemini 1.5 Pro, we use the official Python SDK. Ollama [54] is used to run LLava locally. The same prompt is used for all three models. For our experiments, the set of relations R and global dictionary D are predefined and included in the prompt. We evaluate accuracy via an F1-score for the generated nodes and edges compared to the ground truth scene graph. Full accuracy is based on a perfect match of both nodes and edges. The three models are compared across the scenes in Table I. GPT-4V performs notably better than Gemini and LLava across all scenes. We used GPT-4V as the scene graph generator SGG for other experiments based on these results.

Planning: We used GPT-4 [38] as the task planner $\mathcal{P}$. We evaluate our approach on all three task groups presented in Section IV-B and show results for our approach in Table II. Compared to ViLa, we improve on language and image instruction planning tasks. For the image-based start and goal state, we see an average improvement of $\sim$0.57. For the language instruction task, we outperform SayCan by $\sim$0.56.

This improvement is because of the efficient representation of the scene as a scene graph. Scene graph representations help planning by structuring the setup so the LLM does not need to interpret actions precisely from the image. Additionally, they allow for iterative correction during the actual planning stage. Most of the failure cases in our approach are caused by inaccurate scene graphs. This is addressable because as scene graph generation methods improve, our planning will improve with them. To test how much scene graphs affect planning, we passed the ground truth scene graphs to the planner and observed that the iterative planner proposed successful plans $\sim$100% of the time.

TABLE III: Ablation on error threshold and number of actions per iteration used in VeriGraph.

(a) Effect of error threshold.

Error Threshold (τ)	2	3	5	10
Accuracy (%)	20	85	80	90

(b) Varying of # of actions / iter.

# of Actions	2	3	5	10
Accuracy (%)	85	85	85	85

Our experiment results in Table II show that the iterative planner (last column) achieves higher accuracy than the baselines and our non-iterative planner. This improvement is expected since the iterative planner receives feedback about the plan and replans accordingly.

E. Ablation Study

Iterative vs. non-iterative planner. Our experiment results in Table II show that the iterative planner ('Ours') achieves higher accuracy than the non-iterative planner ('Ours (Direct)') for most tasks. This improvement is expected since the iterative planner receives feedback about the plan and can modify it accordingly. We are able to utilize such a correction and iterative planning model due to our scene graph representation.

Error thresholds. We observed that setting the error threshold to 2 resulted in the worst performance. While setting it to 10 yielded the best performance, it was more time-consuming. We found that setting the threshold to 5 provided a good balance between accuracy and speed.

Number of actions per iteration. We tested 1, 2, 5, and 10 actions per iteration. Although there was no significant difference in accuracy, we ultimately chose to use 3 actions per iteration for optimal performance.

V. Conclusion

In this paper, we present VeriGraph, a novel approach for generating high-level task plans for robot object manipulation using scene graphs. We demonstrate that scene graphs provide an efficient representation of scene information and show that our method outperforms methods that rely on raw pixel values for planning. We also introduced an iterative approach where the task planner's proposed actions are evaluated and corrected before execution on actual objects, enhancing the robustness of the planning process. Additionally, our method offers an efficient way to evaluate high-level robot task-planning algorithms. As scene graph generation algorithms continue to improve in accuracy, the effectiveness of our approach will correspondingly increase. This highlights the potential for further advancements in high-level robot task planning using scene graphs.

We note that current scene graph generation methods suffer from common problems in occlusion and poor performance in out-of-distribution settings. GPT-4v performed well for some scenes however, it failed to generate accurate scene graphs for some scenes. Further research is needed to improve the accuracy of scene graph generators. While this work has demonstrated promise in using scene graphs for planning, in the future, we want to investigate better ways to generate reliable scene graphs for more complex scenes.

References

[1] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International conference on machine learning*. PMLR, 2022, pp. 9118–9147.

[2] W. Huang, F. Xia, D. Shah, D. Driess, A. Zeng, Y. Lu, P. Florence, I. Mordatch, S. Levine, K. Hausman *et al.*, "Grounded decoding: Guiding text generation with grounded models for robot control," *arXiv preprint arXiv:2303.00855*, 2023.

[3] N. Simon and C. Muise, "A natural language model for generating pddl," in *ICAPS KEPS workshop*, 2021.

[4] B. Wang, Z. Wang, X. Wang, Y. Cao, R. A Saurous, and Y. Kim, "Grammar prompting for domain-specific language generation with large language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[5] J. Oswald, K. Srinivas, H. Kokel, J. Lee, M. Katz, and S. Sohrabi, "Large language models as planning domain generators," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 34, pp. 423–431, May 2024. [Online]. Available: https://ojs.aaai.org/index.php/ICAPS/article/view/31502

[6] J. Espasa, I. Miguel, P. Nightingale, A. Z. Salamon, and M. Villaret, "Challenges in modelling and solving plotting with pddl," 2023. [Online]. Available: https://arxiv.org/abs/2310.01470

[7] X. Zhang, Z. Altaweel, Y. Hayamizu, Y. Ding, S. Amiri, H. Yang, A. Kaminski, C. Esselink, and S. Zhang, "Dkprompt: Domain knowledge prompting vision-language models for open-world planning," 2024. [Online]. Available: https://arxiv.org/abs/2406.17659

[8] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng, "Do as i can, not as i say: Grounding language in robotic affordances," 2022.

[9] Y. Hu, F. Lin, T. Zhang, L. Yi, and Y. Gao, "Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning," *arXiv preprint arXiv:2311.17842*, 2023.

[10] B. Li, P. Wu, P. Abbeel, and J. Malik, "Interactive task planning with language models," 2023.

[11] Y. Zhu, J. Tremblay, S. Birchfield, and Y. Zhu, "Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs," 2021.

[12] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa, C. Gan, C. M. de Melo, J. B. Tenenbaum, A. Torralba, F. Shkurti, and L. Paull, "Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning," 2023.

[13] S. Amiri, K. Chandan, and S. Zhang, "Reasoning with scene graphs for robot planning under partial observability," 2022.

[14] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf, "Sayplan: Grounding large language models using 3d scene graphs for scalable task planning," in *7th Annual Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=wMpOMO0Ss7a

[15] K. Valmeekam, M. Marquez, S. Sreedharan, and S. Kambhampati, "On the planning abilities of large language models-a critical investigation," *Advances in Neural Information Processing Systems*, vol. 36, pp. 75 993–76 005, 2023.

[16] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F.-F. Li, "Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations," Feb. 2016. [Online]. Available: http://arxiv.org/abs/1602.07332

[17] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1219–1228.

[18] X. Zhao, L. Wu, X. Chen, and B. Gong, "High-quality image generation from scene graphs with transformer," in *2022 IEEE International Conference on Multimedia and Expo (ICME)*, 2022, pp. 1–6.

[19] G. Mittal, S. Agrawal, A. Agarwal, S. Mehta, and T. Marwah, "Interactive image generation using scene graphs," *arXiv preprint arXiv:1905.03743*, 2019.

[20] S. Tripathi, A. Bhiwandiwalla, A. Bastidas, and H. Tang, "Using scene graph context to improve image generation," *arXiv preprint arXiv:1901.03762*, 2019.

[21] B. Zhao, L. Meng, W. Yin, and L. Sigal, "Image generation from layout," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8584–8593.

[22] L. Yang, Z. Huang, Y. Song, S. Hong, G. Li, W. Zhang, B. Cui, B. Ghanem, and M.-H. Yang, "Diffusion-based scene graph to image generation with masked contrastive pre-training," *arXiv preprint arXiv:2211.11138*, 2022.

[23] L. Gao, B. Wang, and W. Wang, "Image captioning with scene-graph based semantic concepts," in *Proceedings of the 2018 10th international conference on machine learning and computing*, 2018, pp. 225–229.

[24] Y. Zhong, L. Wang, J. Chen, D. Yu, and Y. Li, "Comprehensive image captioning via scene graph decomposition," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer, 2020, pp. 211–229.

[25] X. Yang, J. Peng, Z. Wang, H. Xu, Q. Ye, C. Li, M. Yan, F. Huang, Z. Li, and Y. Zhang, "Transforming visual scene graphs to image captions," *arXiv preprint*

arXiv:2305.02177, 2023.

[26] C. Zhang, W.-L. Chao, and D. Xuan, "An empirical study on leveraging scene graphs for visual question answering," *arXiv preprint arXiv:1907.12133*, 2019.

[27] S. Lee, J.-W. Kim, Y. Oh, and J. H. Jeon, "Visual question answering over scene graph," in *2019 First International Conference on Graph Computing (GC)*, 2019, pp. 45–50.

[28] V. Damodaran, S. Chakravarthy, A. Kumar, A. Umapathy, T. Mitamura, Y. Nakashima, N. Garcia, and C. Chu, "Understanding the role of scene graphs in visual question answering," *arXiv preprint arXiv:2101.05479*, 2021.

[29] G. Sepulveda, J. C. Niebles, and A. Soto, "A deep learning based behavioral approach to indoor autonomous navigation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4646–4653.

[30] Z. Ravichandran, L. Peng, N. Hughes, J. D. Griffith, and L. Carlone, "Hierarchical representations and explicit memory: Learning effective navigation policies on 3d scene graphs using graph neural networks," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 9272–9279.

[31] Z. Ni, X.-X. Deng, C. Tai, X.-Y. Zhu, X. Wu, Y.-J. Liu, and L. Zeng, "Grid: Scene-graph-based instruction-driven robotic task planning," 2023.

[32] Y. Deng, Q. Sima, D. Guo, H. Liu, Y. Wang, and F. Sun, "Scene graph for embodied exploration in cluttered scenario," 2023.

[33] G. Zhai, X. Cai, D. Huang, Y. Di, F. Manhardt, F. Tombari, N. Navab, and B. Busam, "Sg-bot: Object rearrangement via coarse-to-fine robotic imagination on scene graphs," 2023.

[34] Z. Jiao, Y. Niu, Z. Zhang, S.-C. Zhu, Y. Zhu, and H. Liu, "Sequential Manipulation Planning on Scene Graph," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Kyoto, Japan: IEEE, Oct. 2022, pp. 8203–8210. [Online]. Available: https://ieeexplore.ieee.org/document/9981735/

[35] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[36] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.

[37] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen *et al.*, "Palm 2 technical report," *arXiv preprint arXiv:2305.10403*, 2023.

[38] OpenAI, :, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph, "Gpt-4 technical report," 2024.

[39] D. Surís, S. Menon, and C. Vondrick, "Vipergpt: Visual inference via python execution for reasoning," 2023.

[40] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," *arXiv preprint arXiv:2310.03744*, 2023.

[41] P. Pramanick, H. B. Barua, and C. Sarkar, "Decomplex: Task planning from complex natural instructions by a collocating robot," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6894–6901.

[42] S. G. Venkatesh, R. Upadrashta, and B. Amrutur, "Translating natural language instructions to computer programs for robot manipulation," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1919–1926.

[43] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter, "Inner monologue: Embodied reasoning through planning with language models," 2022.

[44] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2022.

[45] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[46] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023.

[47] F. Joublin, A. Ceravola, P. Smirnov, F. Ocker, J. Deigmoeller, A. Belardinelli, C. Wang, S. Hasler, D. Tanneberg, and M. Gienger, "Copal: Corrective planning of robot actions with large language models," 2023.

[48] S. McCallum, M. Taylor-Davies, S. V. Albrecht, and A. Suglia, "Is feedback all you need? leveraging natural language feedback in goal-conditioned reinforcement learning," 2023.

[49] Z. Liu, A. Bahety, and S. Song, "Reflect: Summarizing robot experiences for failure explanation and correction," 2023.

[50] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, "Voyager: An open-ended embodied agent with large language models," 2023.

[51] OpenAI, "Openai (2023)," 2023. [Online]. Available: https://openai.com/index/gpt-4v-system-card/

[52] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, D. Silver, M. Johnson, I. Antonoglou, J. Schrittwieser, A. Glaese, J. Chen, E. Pitler, T. Lillicrap, A. Lazaridou, O. Firat, J. Molloy, M. Isard, P. R. Barham, T. Hennigan, B. Lee, F. Viola, M. Reynolds, Y. Xu, R. Doherty, E. Collins, C. Meyer, E. Rutherford, E. Moreira, K. Ayoub, M. Goel, J. Krawczyk, C. Du, E. Chi, H.-T. Cheng, E. Ni, P. Shah, P. Kane, B. Chan, M. Faruqui, A. Severyn, H. Lin, Y. Li, Y. Cheng, A. Ittycheriah, M. Mahdieh, M. Chen, P. Sun, D. Tran, S. Bagri, B. Lakshminarayanan, J. Liu, A. Orban, F. Güra, H. Zhou, X. Song, A. Boffy, H. Ganapathy, S. Zheng, H. Choe, Ágoston Weisz, T. Zhu, Y. Lu, S. Gopal, J. Kahn, M. Kula, J. Pitman, R. Shah, E. Taropa, M. A. Merey, M. Baeuml, Z. Chen, L. E. Shafey, Y. Zhang, O. Sercinoglu, G. Tucker, E. Piqueras, M. Krikun, I. Barr, N. Savinov, I. Danihelka, B. Roelofs, A. White, A. Andreassen, T. von Glehn, L. Yagati, M. Kazemi, L. Gonzalez, M. Khalman, J. Sygnowski, A. Frechette, C. Smith, L. Culp, L. Proleev, Y. Luan, X. Chen, J. Lottes, N. Schucher, F. Lebron, A. Rrustemi, N. Clay, P. Crone, T. Kocisky, J. Zhao, B. Perz, D. Yu, H. Howard, A. Bloniarz, J. W. Rae, H. Lu, L. Sifre, M. Maggioni, F. Alcober, D. Garrette, M. Barnes, S. Thakoor, J. Austin, G. Barth-Maron, W. Wong, R. Joshi, R. Chaabouni, D. Fatiha, A. Ahuja, G. S. Tomar, E. Senter, M. Chadwick, I. Kornakov, N. Attaluri, I. Iturrate, R. Liu, Y. Li, S. Cogan, J. Chen, C. Jia, C. Gu, Q. Zhang, J. Grimstad, A. J. Hartman, X. Garcia, T. S. Pillai, J. Devlin, M. Laskin, D. de Las Casas, D. Valter, C. Tao, L. Blanco, A. P. Badia, D. Reitter, M. Chen, J. Brennan, C. Rivera, S. Brin, S. Iqbal, G. Surita, J. Labanowski, A. Rao, S. Winkler, E. Parisotto, Y. Gu, K. Olszewska, R. Addanki, A. Miech, A. Louis, D. Teplyashin, G. Brown, E. Catt, J. Balaguer, J. Xiang, P. Wang, Z. Ashwood, A. Briukhov, A. Webson, S. Ganapathy, S. Sanghavi, A. Kannan, M.-W. Chang, A. Stjerngren, J. Djolonga, Y. Sun, A. Bapna, M. Aitchison, P. Pejman, H. Michalewski, T. Yu, C. Wang, J. Love, J. Ahn, D. Bloxwich, K. Han, P. Humphreys, T. Sellam, J. Bradbury, V. Godbole, S. Samangooei, B. Damoc, A. Kaskasoli, S. M. R. Arnold, V. Vasudevan, S. Agrawal, J. Riesa, D. Lepikhin, R. Tanburn, S. Srinivasan, H. Lim, S. Hodkinson, P. Shyam, J. Ferret, S. Hand, A. Garg, T. L. Paine, J. Li, Y. Li, M. Giang, A. Neitz, Z. Abbas, S. York, M. Reid, E. Cole, A. Chowdhery, D. Das, D. Rogozińska, V. Nikolaev, P. Sprechmann, Z. Nado, L. Zilka, F. Prost, L. He, M. Monteiro, G. Mishra, C. Welty, J. Newlan, D. Jia, M. Allamanis, C. H. Hu, R. de Liedekerke, J. Gilmer, C. Saroufim, S. Rijhwani, S. Hou, D. Shrivastava, A. Baddepudi, A. Goldin, A. Ozturel, A. Cassirer, Y. Xu, D. Sohn, D. Sachan, R. K. Amplayo, C. Swanson, D. Petrova, S. Narayan, A. Guez, S. Brahma, J. Landon, M. Patel, R. Zhao, K. Villela, L. Wang, W. Jia, M. Rahtz, M. Giménez, L. Yeung, J. Keeling, P. Georgiev, D. Mincu, B. Wu, S. Haykal, R. Saputro, K. Vodrahalli, J. Qin, Z. Cankara, A. Sharma, N. Fernando, W. Hawkins, B. Neyshabur, S. Kim, A. Hutter, P. Agrawal, A. Castro-Ros, G. van den Driessche, T. Wang, F. Yang, S. yiin Chang, P. Komarek, R. McIlroy, M. Lučić, G. Zhang, W. Farhan, M. Sharman, P. Natsev, P. Michel,

Y. Bansal, S. Qiao, K. Cao, S. Shakeri, C. Butterfield, J. Chung, P. K. Rubenstein, S. Agrawal, A. Mensch, K. Soparkar, K. Lenc, T. Chung, A. Pope, L. Maggiore, J. Kay, P. Jhakra, S. Wang, J. Maynez, M. Phuong, T. Tobin, A. Tacchetti, M. Trebacz, K. Robinson, Y. Katariya, S. Riedel, P. Bailey, K. Xiao, N. Ghelani, L. Aroyo, A. Slone, N. Houlsby, X. Xiong, Z. Yang, E. Gribovskaya, J. Adler, M. Wirth, L. Lee, M. Li, T. Kagohara, J. Pavagadhi, S. Bridgers, A. Bortsova, S. Ghemawat, Z. Ahmed, T. Liu, R. Powell, V. Bolina, M. Iinuma, P. Zablotskaia, J. Besley, D.-W. Chung, T. Dozat, R. Comanescu, X. Si, J. Greer, G. Su, M. Polacek, R. L. Kaufman, S. Tokumine, H. Hu, E. Buchatskaya, Y. Miao, M. Elhawaty, A. Siddhant, N. Tomasev, J. Xing, C. Greer, H. Miller, S. Ashraf, A. Roy, Z. Zhang, A. Ma, A. Filos, M. Besta, R. Blevins, T. Klimenko, C.-K. Yeh, S. Changpinyo, J. Mu, O. Chang, M. Pajarskas, C. Muir, V. Cohen, C. L. Lan, K. Haridasan, A. Marathe, S. Hansen, S. Douglas, R. Samuel, M. Wang, S. Austin, C. Lan, J. Jiang, J. Chiu, J. A. Lorenzo, L. L. Sjösund, S. Cevey, Z. Gleicher, T. Avrahami, A. Boral, H. Srinivasan, V. Selo, R. May, K. Aisopos, L. Hussenot, L. B. Soares, K. Baumli, M. B. Chang, A. Recasens, B. Caine, A. Pritzel, F. Pavetic, F. Pardo, A. Gergely, J. Frye, V. Ramasesh, D. Horgan, K. Badola, N. Kassner, S. Roy, E. Dyer, V. C. Campos, A. Tomala, Y. Tang, D. E. Badawy, E. White, B. Mustafa, O. Lang, A. Jindal, S. Vikram, Z. Gong, S. Caelles, R. Hemsley, G. Thornton, F. Feng, W. Stokowiec, C. Zheng, P. Thacker, Çağlar Ünlü, Z. Zhang, M. Saleh, J. Svensson, M. Bileschi, P. Patil, A. Anand, R. Ring, K. Tsihlas, A. Vezer, M. Selvi, T. Shevlane, M. Rodriguez, T. Kwiatkowski, S. Daruki, K. Rong, A. Dafoe, N. FitzGerald, K. Gu-Lemberg, M. Khan, L. A. Hendricks, M. Pellat, V. Feinberg, J. Cobon-Kerr, T. Sainath, M. Rauh, S. H. Hashemi, R. Ives, Y. Hasson, E. Noland, Y. Cao, N. Byrd, L. Hou, Q. Wang, T. Sottiaux, M. Paganini, J.-B. Lespiau, A. Moufarek, S. Hassan, K. Shivakumar, J. van Amersfoort, A. Mandhane, P. Joshi, A. Goyal, M. Tung, A. Brock, H. Sheahan, V. Misra, C. Li, N. Rakićević, M. Dehghani, F. Liu, S. Mittal, J. Oh, S. Noury, E. Sezener, F. Huot, M. Lamm, N. D. Cao, C. Chen, S. Mudgal, R. Stella, K. Brooks, G. Vasudevan, C. Liu, M. Chain, N. Melinkeri, A. Cohen, V. Wang, K. Seymore, S. Zubkov, R. Goel, S. Yue, S. Krishnakumaran, B. Albert, N. Hurley, M. Sano, A. Mohananey, J. Joughin, E. Filonov, T. Kępa, Y. Eldawy, J. Lim, R. Rishi, S. Badiezadegan, T. Bos, J. Chang, S. Jain, S. G. S. Padmanabhan, S. Puttagunta, K. Krishna, L. Baker, N. Kalb, V. Bedapudi, A. Kurzrok, S. Lei, A. Yu, O. Litvin, X. Zhou, Z. Wu, S. Sobell, A. Siciliano, A. Papir, R. Neale, J. Bragagnolo, T. Toor, T. Chen, V. Anklin, F. Wang, R. Feng, M. Gholami, K. Ling, L. Liu, J. Walter, H. Moghaddam, A. Kishore, J. Adamek, T. Mercado, J. Mallinson, S. Wandekar, S. Cagle, E. Ofek, G. Garrido, C. Lombriser, M. Mukha, B. Sun, H. R. Mohammad, J. Matak, Y. Qian, V. Peswani, P. Janus, Q. Yuan, L. Schelin, O. David, A. Garg, Y. He, O. Duzhyi, A. Älgmyr, T. Lottaz, Q. Li, V. Yadav, L. Xu, A. Chinien, R. Shivanna, A. Chuklin, J. Li, C. Spadine, T. Wolfe, K. Mohamed, S. Das, Z. Dai, K. He, D. von Dincklage, S. Upadhyay, A. Maurya, L. Chi, S. Krause, K. Salama, P. G. Rabinovitch, P. K. R. M, A. Selvan, M. Dektiarev, G. Ghiasi, E. Guven, H. Gupta, B. Liu, D. Sharma, I. H. Shtacher, S. Paul, O. Akerlund, F.-X. Aubet, T. Huang, C. Zhu, E. Zhu, E. Teixeira, M. Fritze, F. Bertolini, L.-E. Marinescu, M. Bölle, D. Paulus, K. Gupta, T. Latkar, M. Chang, J. Sanders, R. Wilson, X. Wu, Y.-X. Tan, L. N. Thiet, T. Doshi, S. Lall, S. Mishra, W. Chen, T. Luong, S. Benjamin, J. Lee, E. Andrejczuk, D. Rabiej, V. Ranjan, K. Styrc, P. Yin, J. Simon, M. R. Harriott, M. Bansal, A. Robsky, G. Bacon, D. Greene, D. Mirylenka, C. Zhou, O. Sarvana, A. Goyal, S. Andermatt, P. Siegler, B. Horn, A. Israel, F. Pongetti, C.-W. L. Chen, M. Selvatici, P. Silva, K. Wang, J. Tolins, K. Guu, R. Yogev, X. Cai, A. Agostini, M. Shah, H. Nguyen, N. Donnaile, S. Pereira, L. Friso, A. Stambler, A. Kurzrok, C. Kuang, Y. Romanikhin, M. Geller, Z. Yan, K. Jang, C.-C. Lee, W. Fica, E. Malmi, Q. Tan, D. Banica, D. Balle, R. Pham, Y. Huang, D. Avram, H. Shi, J. Singh, C. Hidey, N. Ahuja, P. Saxena, D. Dooley, S. P. Potharaju, E. O'Neill, A. Gokulchandran, R. Foley, K. Zhao, M. Dusenberry, Y. Liu, P. Mehta, R. Kotikalapudi, C. Safranek-Shrader, A. Goodman, J. Kessinger, E. Globen, P. Kolhar, C. Gorgolewski, A. Ibrahim, Y. Song, A. Eichenbaum, T. Brovelli, S. Potluri, P. Lahoti, C. Baetu, A. Ghorbani, C. Chen, A. Crawford, S. Pal, M. Sridhar, P. Gurita, A. Mujika, I. Petrovski, P.-L. Cedoz, C. Li, S. Chen, N. D. Santo, S. Goyal, J. Punjabi, K. Kappaganthu, C. Kwak, P. LV, S. Velury, H. Choudhury, J. Hall, P. Shah, R. Figueira, M. Thomas, M. Lu, T. Zhou, C. Kumar, T. Jurdi, S. Chikkerur, Y. Ma, A. Yu, S. Kwak, V. Ähdel, S. Rajayogam, T. Choma, F. Liu, A. Barua, C. Ji, J. H. Park, V. Hellendoorn, A. Bailey, T. Bilal, H. Zhou, M. Khatir, C. Sutton, W. Rzadkowski, F. Macintosh, K. Shagin, P. Medina, C. Liang, J. Zhou, P. Shah, Y. Bi, A. Dankovics, S. Banga, S. Lehmann, M. Bredesen, Z. Lin, J. E. Hoffmann, J. Lai, R. Chung, K. Yang, N. Balani, A. Bražinskas, A. Sozanschi, M. Hayes, H. F. Alcalde, P. Makarov, W. Chen, A. Stella, L. Snijders, M. Mandl, A. Kärrman, P. Nowak, X. Wu, A. Dyck, K. Vaidyanathan, R. R, J. Mallet, M. Rudominer, E. Johnston, S. Mittal, A. Udathu, J. Christensen, V. Verma, Z. Irving, A. Santucci, G. Elsayed, E. Davoodi, M. Georgiev, I. Tenney, N. Hua, G. Cideron, E. Leurent, M. Alnahlawi, I. Georgescu, N. Wei, I. Zheng, D. Scandinaro, H. Jiang, J. Snoek, M. Sundararajan, X. Wang, Z. Ontiveros, I. Karo, J. Cole, V. Rajashekhar,

L. Tumeh, E. Ben-David, R. Jain, J. Uesato, R. Datta, O. Bunyan, S. Wu, J. Zhang, P. Stanczyk, Y. Zhang, D. Steiner, S. Naskar, M. Azzam, M. Johnson, A. Paszke, C.-C. Chiu, J. S. Elias, A. Mohiuddin, F. Muhammad, J. Miao, A. Lee, N. Vieillard, J. Park, J. Zhang, J. Stanway, D. Garmon, A. Karmarkar, Z. Dong, J. Lee, A. Kumar, L. Zhou, J. Evens, W. Isaac, G. Irving, E. Loper, M. Fink, I. Arkatkar, N. Chen, I. Shafran, I. Petrychenko, Z. Chen, J. Jia, A. Levskaya, Z. Zhu, P. Grabowski, Y. Mao, A. Magni, K. Yao, J. Snaider, N. Casagrande, E. Palmer, P. Suganthan, A. Castaño, I. Giannoumis, W. Kim, M. Rybiński, A. Sreevatsa, J. Prendki, D. Soergel, A. Goedeckemeyer, W. Gierke, M. Jafari, M. Gaba, J. Wiesner, D. G. Wright, Y. Wei, H. Vashisht, Y. Kulizhskaya, J. Hoover, M. Le, L. Li, C. Iwuanyanwu, L. Liu, K. Ramirez, A. Khorlin, A. Cui, T. LIN, M. Wu, R. Aguilar, K. Pallo, A. Chakladar, G. Perng, E. A. Abellan, M. Zhang, I. Dasgupta, N. Kushman, I. Penchev, A. Repina, X. Wu, T. van der Weide, P. Ponnapalli, C. Kaplan, J. Simsa, S. Li, O. Dousse, F. Yang, J. Piper, N. Ie, R. Pasumarthi, N. Lintz, A. Vijayakumar, D. Andor, P. Valenzuela, M. Lui, C. Paduraru, D. Peng, K. Lee, S. Zhang, S. Greene, D. D. Nguyen, P. Kurylowicz, C. Hardin, L. Dixon, L. Janzer, K. Choo, Z. Feng, B. Zhang, A. Singhal, D. Du, D. McKinnon, N. Antropova, T. Bolukbasi, O. Keller, D. Reid, D. Finchelstein, M. A. Raad, R. Crocker, P. Hawkins, R. Dadashi, C. Gaffney, K. Franko, A. Bulanova, R. Leblond, S. Chung, H. Askham, L. C. Cobo, K. Xu, F. Fischer, J. Xu, C. Sorokin, C. Alberti, C.-C. Lin, C. Evans, A. Dimitriev, H. Forbes, D. Banarse, Z. Tung, M. Omernick, C. Bishop, R. Sterneck, R. Jain, J. Xia, E. Amid, F. Piccinno, X. Wang, P. Banzal, D. J. Mankowitz, A. Polozov, V. Krakovna, S. Brown, M. Bateni, D. Duan, V. Firoiu, M. Thotakuri, T. Natan, M. Geist, S. tan Girgin, H. Li, J. Ye, O. Roval, R. Tojo, M. Kwong, J. Lee-Thorp, C. Yew, D. Sinopalnikov, S. Ramos, J. Mellor, A. Sharma, K. Wu, D. Miller, N. Sonnerat, D. Vnukov, R. Greig, J. Beattie, E. Caveness, L. Bai, J. Eisenschlos, A. Korchemniy, T. Tsai, M. Jasarevic, W. Kong, P. Dao, Z. Zheng, F. Liu, F. Yang, R. Zhu, T. H. Teh, J. Sanmiya, E. Gladchenko, N. Trdin, D. Toyama, E. Rosen, S. Tavakkol, L. Xue, C. Elkind, O. Woodman, J. Carpenter, G. Papamakarios, R. Kemp, S. Kafle, T. Grunina, R. Sinha, A. Talbert, D. Wu, D. Owusu-Afriyie, C. Du, C. Thornton, J. Pont-Tuset, P. Narayana, J. Li, S. Fatehi, J. Wieting, O. Ajmeri, B. Uria, Y. Ko, L. Knight, A. Héliou, N. Niu, S. Gu, C. Pang, Y. Li, N. Levine, A. Stolovich, R. Santamaria-Fernandez, S. Goenka, W. Yustalim, R. Strudel, A. Elqursh, C. Deck, H. Lee, Z. Li, K. Levin, R. Hoffmann, D. Holtmann-Rice, O. Bachem, S. Arora, C. Koh, S. H. Yeganeh, S. Põder, M. Tariq, Y. Sun, L. Ionita, M. Seyedhosseini, P. Tafti, Z. Liu, A. Gulati, J. Liu, X. Ye, B. Chrzaszcz, L. Wang, N. Sethi, T. Li, B. Brown, S. Singh, W. Fan, A. Parisi, J. Stanton, V. Koverkathu, C. A. Choquette-Choo, Y. Li, T. Lu, A. Ittycheriah, P. Shroff, M. Varadarajan, S. Bahargam, R. Willoughby, D. Gaddy, G. Desjardins, M. Cornero, B. Robenek, B. Mittal, B. Albrecht, A. Shenoy, F. Moiseev, H. Jacobsson, A. Ghaffarkhah, M. Rivière, A. Walton, C. Crepy, A. Parrish, Z. Zhou, C. Farabet, C. Radebaugh, P. Srinivasan, C. van der Salm, A. Fidjeland, S. Scellato, E. Latorre-Chimoto, H. Klimczak-Plucińska, D. Bridson, D. de Cesare, T. Hudson, P. Mendolicchio, L. Walker, A. Morris, M. Mauger, A. Guseynov, A. Reid, S. Odoom, L. Loher, V. Cotruta, M. Yenugula, D. Grewe, A. Petrushkina, T. Duerig, A. Sanchez, S. Yadlowsky, A. Shen, A. Globerson, L. Webb, S. Dua, D. Li, S. Bhupatiraju, D. Hurt, H. Qureshi, A. Agarwal, T. Shani, M. Eyal, A. Khare, S. R. Belle, L. Wang, C. Tekur, M. S. Kale, J. Wei, R. Sang, B. Saeta, T. Liechty, Y. Sun, Y. Zhao, S. Lee, P. Nayak, D. Fritz, M. R. Vuyyuru, J. Aslanides, N. Vyas, M. Wicke, X. Ma, E. Eltyshev, N. Martin, H. Cate, J. Manyika, K. Amiri, Y. Kim, X. Xiong, K. Kang, F. Luisier, N. Tripuraneni, D. Madras, M. Guo, A. Waters, O. Wang, J. Ainslie, J. Baldridge, H. Zhang, G. Pruthi, J. Bauer, F. Yang, R. Mansour, J. Gelman, Y. Xu, G. Polovets, J. Liu, H. Cai, W. Chen, X. Sheng, E. Xue, S. Ozair, C. Angermueller, X. Li, A. Sinha, W. Wang, J. Wiesinger, E. Koukoumidis, Y. Tian, A. Iyer, M. Gurumurthy, M. Goldenson, P. Shah, M. Blake, H. Yu, A. Urbanowicz, J. Palomaki, C. Fernando, K. Durden, H. Mehta, N. Momchev, E. Rahimtoroghi, M. Georgaki, A. Raul, S. Ruder, M. Redshaw, J. Lee, D. Zhou, K. Jalan, D. Li, B. Hechtman, P. Schuh, M. Nasr, K. Milan, V. Mikulik, J. Franco, T. Green, N. Nguyen, J. Kelley, A. Mahendru, A. Hu, J. Howland, B. Vargas, J. Hui, K. Bansal, V. Rao, R. Ghiya, E. Wang, K. Ye, J. M. Sarr, M. M. Preston, M. Elish, S. Li, A. Kaku, J. Gupta, I. Pasupat, D.-C. Juan, M. Someswar, T. M., X. Chen, A. Amini, A. Fabrikant, E. Chu, X. Dong, A. Muthal, S. Buthpitiya, S. Jauhari, N. Hua, U. Khandelwal, A. Hitron, J. Ren, L. Rinaldi, S. Drath, A. Dabush, N.-J. Jiang, H. Godhia, U. Sachs, A. Chen, Y. Fan, H. Taitelbaum, H. Noga, Z. Dai, J. Wang, C. Liang, J. Hamer, C.-S. Ferng, C. Elkind, A. Atias, P. Lee, V. Listík, M. Carlen, J. van de Kerkhof, M. Pikus, K. Zaher, P. Müller, S. Zykova, R. Stefanec, V. Gatsko, C. Hirnschall, A. Sethi, X. F. Xu, C. Ahuja, B. Tsai, A. Stefanoiu, B. Feng, K. Dhandhania, M. Katyal, A. Gupta, A. Parulekar, D. Pitta, J. Zhao, V. Bhatia, Y. Bhavnani, O. Alhadlaq, X. Li, P. Danenberg, D. Tu, A. Pine, V. Filippova, A. Ghosh, B. Limonchik, B. Urala, C. K. Lanka, D. Clive, Y. Sun, E. Li, H. Wu, K. Hongtongsak, I. Li, K. Thakkar, K. Omarov, K. Majmundar, M. Alverson, M. Kucharski, M. Patel, M. Jain, M. Zabelin, P. Pelagatti, R. Kohli, S. Kumar, J. Kim, S. Sankar, V. Shah, L. Ramachandruni, X. Zeng, B. Bariach, L. Weidinger, T. Vu, A. Andreev, A. He, K. Hui,

S. Kashem, A. Subramanya, S. Hsiao, D. Hassabis, K. Kavukcuoglu, A. Sadovsky, Q. Le, T. Strohman, Y. Wu, S. Petrov, J. Dean, and O. Vinyals, "Gemini: A family of highly capable multimodal models," 2024. [Online]. Available: https://arxiv.org/abs/2312.11805

[53] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," 2023.

[54] github, "Github," 2024. [Online]. Available: https://github.com/ollama/ollama