PIVOT: Iterative Visual Prompting Elicits Actionable Knowledge for VLMs

Soroush Nasiriany[*,†,1,3], **Fei Xia**[*,1], **Wenhao Yu**[*,1], **Ted Xiao**[*,1], **Jacky Liang**[1], **Ishita Dasgupta**[1], **Annie Xie**[2], **Danny Driess**[1], **Ayzaan Wahid**[1], **Zhuo Xu**[1], **Quan Vuong**[1], **Tingnan Zhang**[1], **Tsang-Wei Edward Lee**[1], **Kuang-Huei Lee**[1], **Peng Xu**[1], **Sean Kirmani**[1], **Yuke Zhu**[3], **Andy Zeng**[1], **Karol Hausman**[1], **Nicolas Heess**[1], **Chelsea Finn**[1], **Sergey Levine**[1], **Brian Ichter**[*,1]

[1]Google DeepMind, [2]Stanford University, [3]The University of Texas at Austin
Correspond to: {soroushn, xiafei, magicmelon, tedxiao, ichter}@google.com
Website: pivot-prompt.github.io and HuggingFace: https://huggingface.co/spaces/pivot-prompt/pivot-prompt-demo

arXiv:2402.07872v1 [cs.RO] 12 Feb 2024

Vision language models (VLMs) have shown impressive capabilities across a variety of tasks, from logical reasoning to visual understanding. This opens the door to richer interaction with the world, for example robotic control. However, VLMs produce only textual outputs, while robotic control and other spatial tasks require outputting continuous coordinates, actions, or trajectories. How can we enable VLMs to handle such settings without fine-tuning on task-specific data?

In this paper, we propose a novel visual prompting approach for VLMs that we call Prompting with Iterative Visual Optimization (PIVOT), which casts tasks as iterative visual question answering. In each iteration, the image is annotated with a visual representation of proposals that the VLM can refer to (e.g., candidate robot actions, localizations, or trajectories). The VLM then selects the best ones for the task. These proposals are iteratively refined, allowing the VLM to eventually zero in on the best available answer. We investigate PIVOT on real-world robotic navigation, real-world manipulation from images, instruction following in simulation, and additional spatial inference tasks such as localization. We find, perhaps surprisingly, that our approach enables zero-shot control of robotic systems without any robot training data, navigation in a variety of environments, and other capabilities. Although current performance is far from perfect, our work highlights potentials and limitations of this new regime and shows a promising approach for Internet-Scale VLMs in robotic and spatial reasoning domains.

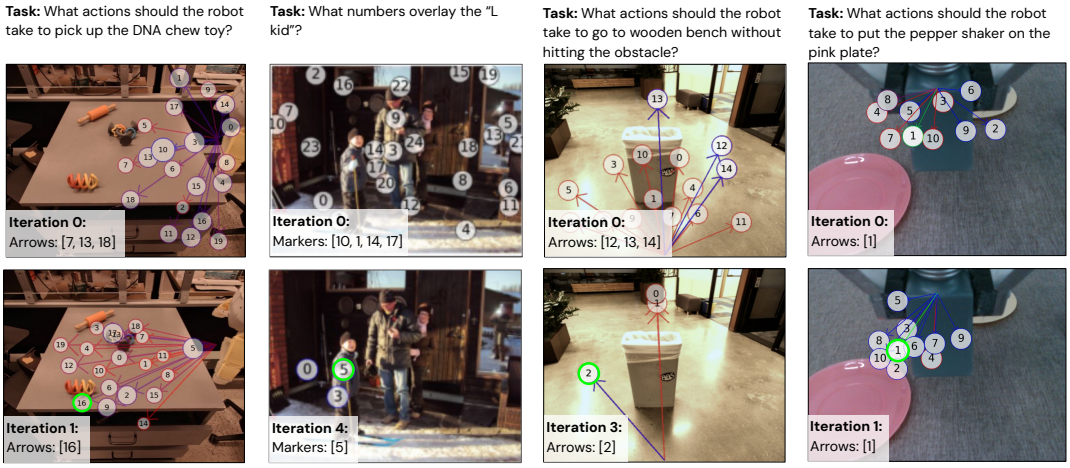


Figure 1 | Prompting with Iterative Visual Optimization (PIVOT) casts spatial reasoning tasks, such as robotic control, as a VQA problem. This is done by first annotating an image with a visual representation of robot actions or 3D coordinates, then querying a VLM to select the most promising annotated actions seen in the image. The best action is iteratively refined by fitting a distribution to the selected actions and requerying the VLM. This procedure enables us to solve complex tasks that require outputting grounded continuous coordinates or robot actions utilizing a VLM without any domain-specific training.

© 2024 Google DeepMind. All rights reserved
[*] Equal contribution, ordering randomly decided. [†] Work done while a student researcher at Google DeepMind.

1. Introduction

Large language models (LLMs) have shown themselves capable of solving a broad range of practical problems, from code generation to question answering and even logical deduction [5, 3, 52]. The extension of LLMs to multi-modal inputs, resulting in powerful vision-language models (VLMs), enables models that handle much richer visual modalities [16, 37, 2, 9], which makes it feasible to interact not only with natural language but directly with the physical world. However, most VLMs still only *output* textual answers, seemingly limiting such interactions to high-level question answering. Many real-world problems are inherently spatial: controlling the trajectory of a robotic arm, selecting a waypoint for a mobile robot, choosing how to rearrange objects on a table, or even localizing keypoints in an image. Can VLMs be adapted to solve these kinds of embodied, physical, and spatial problems? And can they do so zero shot, without additional in-domain training data? In this work, we propose an iterative prompting method to make this possible and study the limits and potentials for zero-shot robotic control and spatial inference with VLMs.

Our proposed method is based on a simple insight: although VLMs struggle to produce precise spatial outputs directly, they can readily select among a discrete set of coarse choices, and this in turn can be used to *refine* this set to provide more precise choices at the next iteration. At each iteration of our iterative procedure, we annotate the image with candidate proposals (i.e., numbered keypoints as in Yang et al. [59]) drawn from a proposal distribution, and ask the VLM to rank the degree to which they perform the desired task. We then *refine* this proposal distribution, generate new candidate proposals that are clustered around better regions of the output space, and repeat this procedure. With this optimization approach, the entire loop can be viewed as an iterative optimization similar to the cross-entropy method [11], with each step being framed as a visual question compatible with current VLMs without any additional training. In Figure 1 and throughout this work, we use robot control as a running example, wherein candidates are numbered arrows.

Equipped with our method for extracting spatial outputs from VLMs, we study the limits and potentials of zero-shot VLM inference in a range of domains: robotic navigation, grasping and rearranging objects, language instructions in a simulated robotic benchmark, and non-robot spatial inference through keypoint localization. It is important to note that in all of these domains, we use state-of-the-art vision language models, namely GPT-4 [37] and Gemini [17], *without any modification or finetuning*. Our aim is not necessarily to develop the best possible robotic control or keypoint localization technique, but to study the limits and potentials of such models. We expect that future improvements to VLMs will lead to further quantitative gains on the actual tasks. The zero-shot performance of VLMs in these settings is far from perfect, but the ability to control robots in zero shot without *any* robotic data, complex prompt design, code generation, or other specialized tools provides a very flexible and general way to obtain highly generalizable systems.

Our main contribution is thus an approach for visual prompting and iterative optimization with VLMs, applications to low-level robotic control and other spatial tasks, and an empirical analysis of potentials and limitations of VLMs for such zero-shot spatial inference. We apply our approach to a variety of robotic systems and general visually-grounded visual question and answer tasks, and evaluates the kinds of situations where this approach succeeds and fails. While our current results are naturally specific to current state-of-the-art VLMs, we find that performance improves with larger, more performant VLMs. Thus, as VLM capabilities continue to improve with time, we expect our proposed approach to improve in turn.

2. Related Work

Visual annotations with VLMs. With the increasing capabilities of VLMs, there has been growing interest in understanding their abilities to understand visual annotations [60, 46, 57, 65], improving such capabilities [6, 56], as well as leveraging them for perception or decision-making tasks [18, 59, 53, 26, 33]. Shtedritski et al. [46] identify that VLMs like CLIP [40] can recognize certain visual annotations. Yang et al. [60] perform a more comprehensive analysis on the GPT-4 model and demonstrate its ability to understand complex visual annotations. Yang et al. [59] demonstrates how such a model can solve visual reasoning tasks by annotating the input image with object masks and numbers. Several works too have applied visual prompting methods to web navigation tasks [26, 57, 65], obtaining impressive-zero shot performance. Our work builds upon these works: instead of taking proposals as given or generating the proposals with a separate perception systems, PIVOT generates proposals randomly, but then adapt the distribution through iterative refinement. As a result, we can obtain relatively precise outputs through multiple iterations, and do not require any separate perception system or any other model at all besides the VLM itself.

Prompt optimization. The emergence of few-shot in context learning within LLMs [5] has lead to many breakthroughs in prompting. Naturally prompt optimization has emerged as a promising approach, whether with gradients [29, 28] or without gradients, e.g., with human engineering [27] or through automatic optimization in language space [66]. These automatic approaches are most related to our work and have shown that language-model feedback [39], answer scores [66, 58, 55], and environment feedback [49] can significantly improve the outputs of LLMs and VLMs. A major difference between these prior methods and ours is that our iterative prompting uses refinement of the *visual* input, by changing the visual annotations across refinement steps. We optimize prompts "online" for a specific query rather than offline to identify a fixed prompt, and show that our iterative procedure leads to more precise spatial outputs.

Foundation models for robot reasoning and control. In recent years, foundation models have shown impressive results in robotics from high-level reasoning to low-level control [13, 19]. Many early works investigated robotic reasoning and planning regimes where LLMs and language outputs are well suited [21, 64, 1, 22, 34, 41, 47, 32, 31, 51, 8]. To apply foundation models to control tasks, several promising approaches have emerged. One line of work has shown that foundation-model-selected subgoals are an effective abstraction to feed into policies for navigation [12, 44, 7, 20, 43, 14] and manipulation [10, 45]. Another abstraction that has been shown to be effective for control is LLM generated rewards, which can be optimized within simulation [23, 62, 35]. Others have investigated code writing LLMs to directly write code that can be executed via control and perceptive primitives [30, 48, 54]. On simple domains, even few-shot prompting language models has been shown to be capable of control [36, 50], while finetuned foundation models have yielded significantly more capable VLM-based controllers [4, 45, 25, 42, 15, 38]. Unlike these works, we show how VLMs can be applied *zero-shot* to low-level control of multiple real robot platforms.

3. Prompting with Iterative Visual Optimization

The type of tasks this work considers have to be solved by producing a value $a \in \mathcal{A}$ from a set $\mathcal{A}$ given a task description in natural language $\ell \in \mathcal{L}$ and an image observation $I \in \mathbb{R}^{H \times W \times 3}$. This set $\mathcal{A}$ can, for example, include continuous coordinates, 3D spatial locations, robot control actions, or trajectories. When $\mathcal{A}$ is the set of robot actions, this amounts to finding a policy $\pi(\cdot|\ell, I)$ that emits an action $a \in \mathcal{A}$. The majority of our experiments focus on finding a control policy for robot actions. Therefore, in the following, we present our method of PIVOT with this use-case in mind. However, PIVOT is a general algorithm to generate (continuous) outputs from a VLM.

3.1. Grounding VLMs to Robot Actions through Image Annotations

We propose framing the problem of creating a policy π as a Visual Question Answering (VQA) problem. The class of VLMs we use in this work take as input an image I and a textual prefix w_p from which they generate a distribution $P_{\text{VLM}}(\cdot|w_p, I)$ of textual completions. Utilizing this interface to derive a policy raises the challenge of how an action from a (continuous) space $\mathcal{A}$ can be represented as a textual completion.

The core idea of this work is to lift low-level actions into the *visual language* of a VLM, i.e., a combination of images and text, such that it is closer to the training distribution of general vision-language tasks. To achieve this, we propose the *visual prompt mapping*

$$\left(\hat{I}, w_{1:M}\right) = \Omega(I, a_{1:M}) \tag{1}$$

that transforms an image observation I and set of candidate actions $a_{1:M}$, $a_j \in \mathcal{A}$ into an annotated image $\hat{I}$ and their corresponding textual labels $w_{1:M}$ where w_j refers to the annotation representing a_j in the image space. For example, as visualized in Fig. 1, utilizing the camera matrices, we can project a 3D location into the image space, and draw a visual marker at this projected location. Labeling this marker with a textual reference, e.g., a number, consequently enables the VLM to not only be queried in its natural input space, namely images and text, but also to refer to spatial concepts in its natural output space by producing text that references the marker labels. In Section 4.4 we investigate different choices of the mapping (1) and ablate its influence on performance.

3.2. Prompting with Iterative Visual Optimization

Representing (continuous) robot actions and spatial concepts in image space with their associated textual labels allows us to query the VLM P_{VLM} to judge if an action would be promising in solving the task. Therefore, we can view obtaining a policy π as solving the optimization problem

$$\max_{a \in \mathcal{A}, w} \; P_{\text{VLM}}\left(w \,\middle|\, \hat{I}, \ell\right) \quad \text{s.t.} \quad \left(\hat{I}, w\right) = \Omega(I, a). \tag{2}$$

Intuitively, we aim to find an action a for which the VLM would choose the corresponding label w after applying the mapping Ω. In order to solve (2), we propose an iterative algorithm, which we refer to as Prompting with Iterative Visual Optimization. In each iteration i the algorithm first samples a set of candidate actions $a^{(i)}_{1:M}$ from a distribution $P_{\mathcal{A}^{(i)}}$ (Figure 2 (a)). These candidate actions are then mapped onto the image I producing the annotated image $\hat{I}^{(i)}$ and the associated action labels $w^{(i)}_{1:M}$ (Figure 2 (b)). We then query the VLM on a multiple choice-style question on the labels $w^{(i)}_{1:M}$ to choose which of the candidate actions are most promising (Figure 2 (c)). This leads to set of best actions to which we fit a new distribution $P_{\mathcal{A}^{(i+1)}}$ (Figure 2 (d)). The process is repeated until convergence or a maximum number of steps N is reached. Algorithm 1 and Figure 2 visualize this process.

3.3. Robust PIVOT with Parallel Calls

VLMs can make mistakes, causing PIVOT to select actions in sub-optimal regions. To improve the robustness of PIVOT, we use a parallel call strategy, where we first execute E parallel PIVOT instances and obtain E candidate actions. We then aggregate the selected candidates to identify the final action output. To aggregate the candidate actions from different PIVOT instances, we compare two approaches: 1) we fit a new action distribution from the E action candidates and return the fitted action distribution, 2) we query the VLM again to select the single best action from the E actions. We

find that by adopting parallel calls we can effectively improve the robustness of PIVOT and mitigate local minima in the optimization process.

Algorithm 1 Prompting with Iterative Visual Optimization

Given: image I, instruction ℓ, action space $\mathcal{A}$, max iterations N, number of samples M
Initialize: $\mathcal{A}^{(0)} = \mathcal{A}$, $i = 0$
while $i < N$ **do**
  Sample actions $a_{1:M}$ from $P_{\mathcal{A}^{(i)}}$
  Project actions into image space and textual labels $(\hat{I}, w_{1:M}) = \Omega(I, a_{1:M})$
  Query VLM $P_{\text{VLM}}(w \mid \hat{I}, \ell)$ to determine the most promising actions
  Fit distribution $P_{\mathcal{A}^{(i+1)}}$ to best actions
  Increment iterations $i \leftarrow i + 1$
end while
Return: an action from the VLM best actions

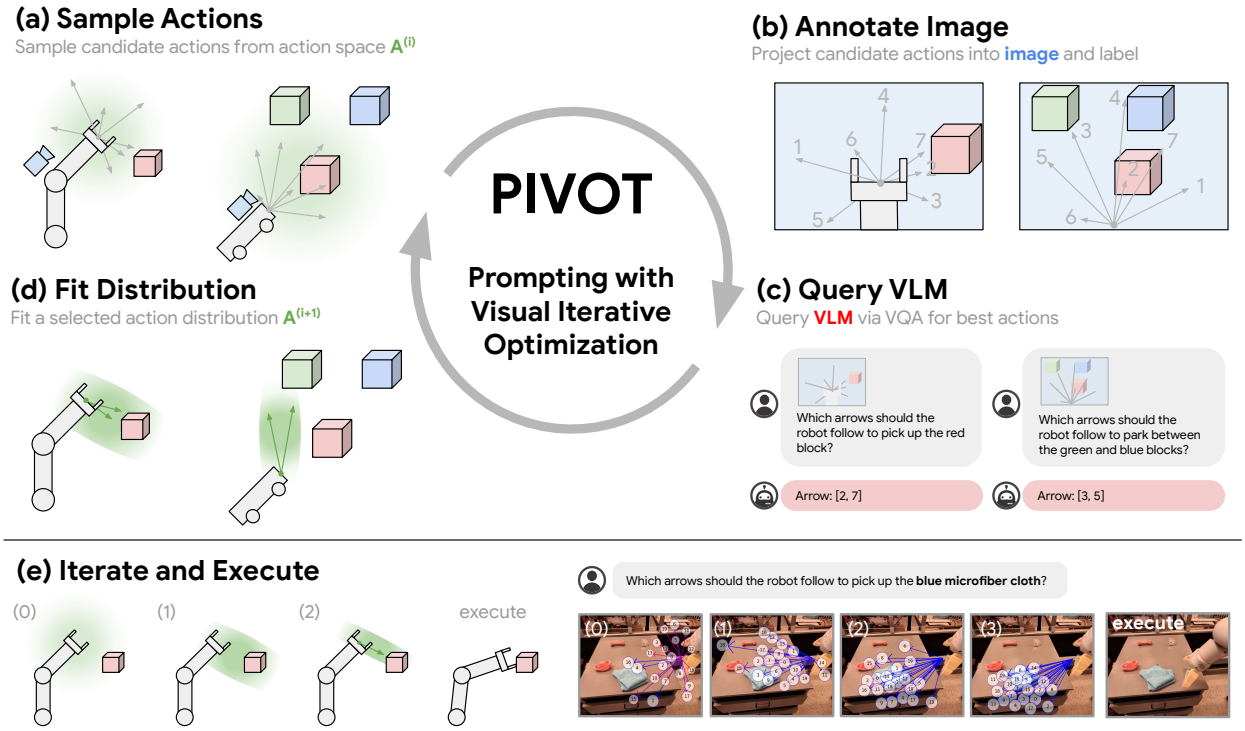


Figure 2 | Prompting with Iterative Visual Optimization produces a robot control policy by iteratively (a) sampling actions from an action distribution $\mathcal{A}^{(i)}$, (b) projecting them into the image space and annotating each sample, (c) querying a VLM for the best actions, and (d) fitting a distribution to the selected actions to form $\mathcal{A}^{(i+1)}$. (e) After a set number of iterations, a selected best action is executed.

3.4. PIVOT Implementation

Our approach can be used to query the VLM for any type of answer as long as multiple answers can be simultaneously visualized on the image. As visualized in Figure 1, for the visual prompting mapping Ω, we represent actions as arrows emanating from the robot or the center of the image if the embodiment is not visible. For 3D problems, the colors of the arrows and size of the labels indicate forward and backwards movement. We label these actions with a number label circled at the end

of the arrow. Unless otherwise noted, the VLM used herein was GPT-4V [37]. For creating the text prompt w_p, we prompt the VLM to use chain of thought to reason through the problem and then summarize the top few labels. The distributions $P_{\mathcal{A}}$ in Algorithm 1 are approximated as isotropic Gaussians.

4. Experiments

We investigate the capabilities and limitations of PIVOT for visuomotor robotic control and visually grounded (e.g., spatial) VQA. Our primary examples involve action selection for control because (a) it requires fine-grained visual grounding, (b) actions can be difficult to express in language, and (c) it is often bottlenecked by visual generalization, which benefits from the knowledge stored within pre-trained VLMs. We aim to understand both the strength and weaknesses of our approach, and believe that (i) identifying these limitations and (ii) understanding how they may be alleviated via scaling and by improving the underlying foundation models are main contributions of this work. Specifically, we seek to answer the questions:

1. How does PIVOT perform on robotic control tasks?
2. How does PIVOT perform on object reference tasks?
3. What is the influence of the different components of PIVOT (textual prompting, visual prompting, and iterative optimization) on performance?
4. What are the limitations of PIVOT with current VLMs?
5. How does PIVOT scale with VLM performance?

4.1. Robotics Experimental Setup

We evaluate PIVOT across the following robot embodiments, which are visualized in Figure 3 and described in detail in Appendix A:

- Mobile manipulator with a head-mounted camera for both navigation (2D action space, Figure 3 (a) and manipulation tasks (4D end-effector relative Cartesian (x, y, z) and binary gripper action space, Figure 3 (b).
- Franka arm with a wrist-mounted camera and a 4D action space (end-effector relative Cartesian (x, y, z) and gripper). Results shown in Appendix F.
- RAVENS [63] simulator, with an overhead camera and a pick and place pixel action space. Results shown in Appendix E.

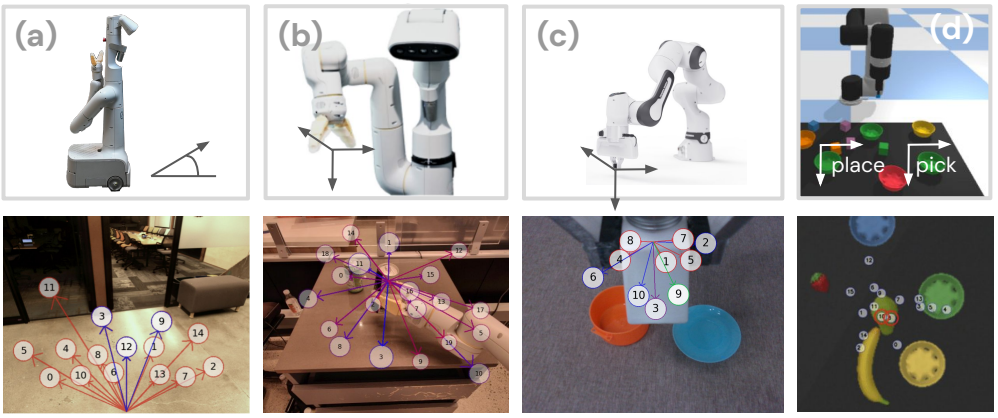
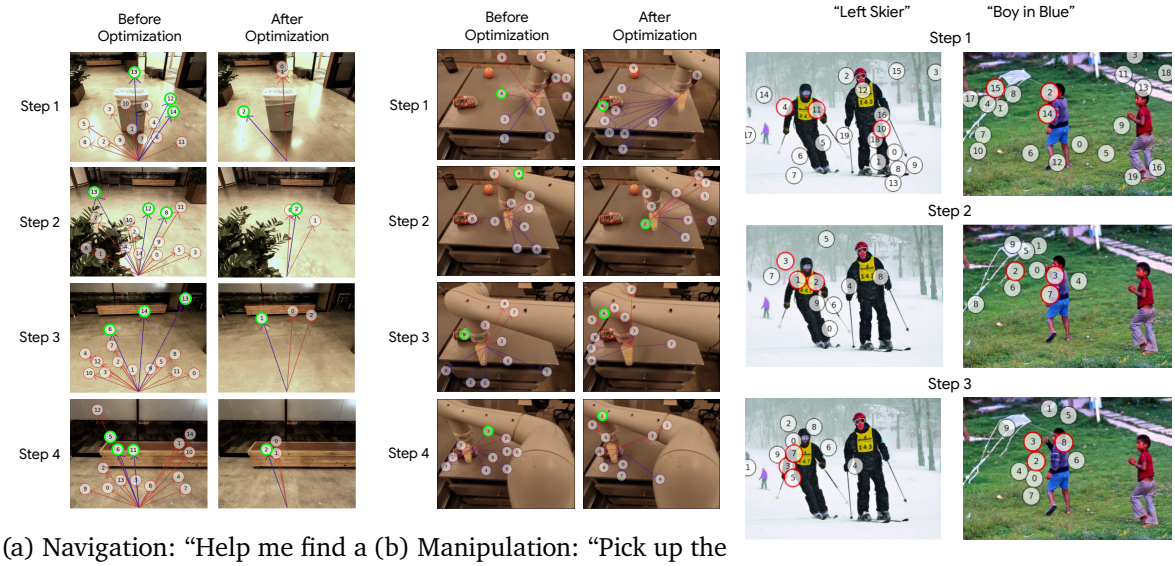


Figure 3 | We evaluate PIVOT on several robot embodiments including: a mobile manipulator for (a) navigation and (b) manipulation, (c) single Franka arm manipulation, and (d) tabletop pick-and-place [63].

4.2. Zero-shot Robotic Control in the Real World



(a) Navigation: "Help me find a place to sit and write"

(b) Manipulation: "Pick up the coke can"

(c) RefCOCO spatial reasoning

Figure 4 | (a) An example rollout on a real-world navigation task. We use three parallel calls to generate samples. (b) An example rollout on a real-world manipulation task, where actions selected by PIVOT with 3 iterations are directly executed at every step. PIVOT improves the robustness and precision of robot actions, enabling corrective behavior such as in Step 2. (c) An example rollout on RefCOCO questions.

Our first set of real robot experiments evaluate PIVOT's ability to perform zero-shot robotic control with mobile manipulator navigation and manipulation, and Franka manipulation. These highlight the flexibility of PIVOT, as these robots vary in terms of control settings (navigation and manipulation), camera views (first and third person), as well as action space dimensionalities. For example, Figure 4 illustrates several qualitative rollouts of PIVOT and the action samples (projected onto the images) as it steps through the iteration process. Note that after optimization, selected actions are more precisely positioned on target objects and areas of interest (most relevant to the input language instructions), without any model fine-tuning. For goal-directed navigation tasks, we quantitatively evaluate PIVOT by measuring the success rates of whether it enables the mobile manipulator to reach its target destination (provided as a language input to PIVOT). For manipulation, we evaluate performance via three metrics (i) whether the robot end-effector reaches the relevant object (reach), (ii) efficiency via the number of action steps before successful termination (steps), and (iii) the success rate at which the robot grasps the relevant object (grasp – when applicable).

Table 1 | Navigation success rate on the mobile manipulator in Figure 3 (a). We observe that iterations and parallel calls improve performance.

Task	No Iteration No Parallel	3 Iterations No Parallel	No Iteration 3 Parallel	3 Iterations 3 Parallel
Go to orange table with tissue box	25%	50%	**75%**	**75%**
Go to wooden bench without hitting obstacle	25%	50%	**75%**	50%
Go to the darker room	25%	50%	75%	**100%**
Help me find a place to sit and write	75%	50%	**100%**	75%

Results on both navigation and manipulation tasks (shown in Tables 1 and 2) demonstrate that (i) PIVOT enables non-zero task success for both domains, (ii) parallel calls improves performance (in terms of success rates) and efficiency (by reducing the average number of actions steps), and (iii)

increasing the number of PIVOT iterations also improves performance. Appendix F and E presents

Table 2 | Manipulation results on the mobile manipulator shown in Figure 3 (b), where "Reach" indicates the rate at which the robot successfully reached the relevant object, "Steps" indicates the number of steps, and "Grasp" indicates the rate at which the robot successfully grasped the relevant object (when applicable for the task). We observe that while all approaches are able to achieve some non-zero success, iteration and parallel calls improve performance and efficiency of the policy.

	No Iterations No Parallel			3 Iterations No Parallel			3 Iterations 3 Parallel		
Task	Reach	Steps	Grasp	Reach	Steps	Grasp	Reach	Steps	Grasp
Pick coke can	50%	4.5	0.0%	67%	**3.0**	33%	**100%**	**3.0**	**67%**
Bring the orange to the X	20%	4.0	-	**80%**	**3.5**	-	67%	**3.5**	-
Sort the apple	67%	3.5	-	**100%**	3.25	-	75%	**3.0**	-

results on real Franka arm and a simulated RAVENS domain.

4.3. Zero-shot Visual Grounding

In addition to robotic control tasks, we also examine PIVOT for reference localization tasks from RefCOCO [61], which evaluates precise and robust visual grounding. To this end, we evaluate GPT-4V with 3 rounds of PIVOT on a random subset of 1000 examples from the RefCOCO testA split. We find strong performance even in the first iteration with modest improvement over further iterations. Prompts used are in Appendix H and results are in Figure 5 and examples in Figure 4.

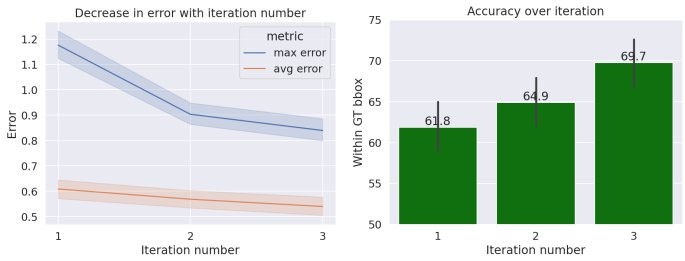


Figure 5 | RefCOCO quantitative results. (Left) Normalized distance between the center of the ground truth bounding box and the selected circle. (Right) Accuracy as measured by whether the selected circle lies within the ground truth bounding box.

We provide an interactive demo on HuggingFace with a few demonstrative images as well as the ability to upload new images and questions; available here.

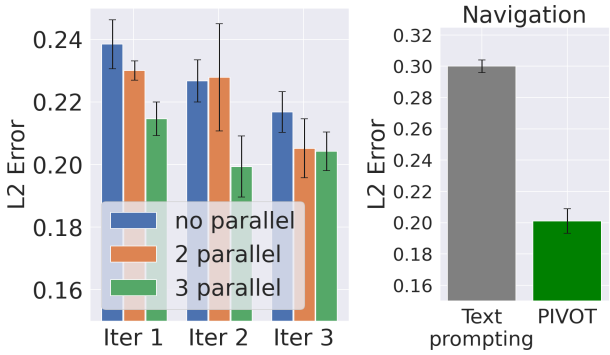
4.4. Offline Performance and Ablations

In this section, we examine each element of PIVOT (the text prompt, visual prompt, and iterative optimization) through an offline evaluation, allowing a thorough evaluation without requiring execution on real robots. To do this, we use demonstration data as a reference and compute how similar the action computed by PIVOT is to the ground-truth expert action.

For the manipulation domain, we obtain the reference robot action from the RT-X dataset [38] and compute the cosine similarity of the two actions in the camera frame as our metric. This metric measures how VLM choice is "aligned" with human demonstrations. For example, a 0.5 cosine similarity in 2D space correspond to $\arccos(0.5) = 60°$. As our actions can be executed a maximum delta along the chosen Cartesian action direction, we have found this metric more informative than others, e.g., mean squared error. For the navigation domain, we use a human-labeled dataset from navigation logs and compute the normalized L2 distance between the selected action and the point of interest in camera frame as our metric. More information on each offline dataset can be found in Appendix D and B.

Text prompts. To understand the effect of different text prompts, we experiment with several design choices, with numbers reported in Appendix D. We investigate the role of zero-shot, few-shot, chain of thought, and direct prompting; we find that zero-shot chain of thought performs the best, though few-shot direct prompting is close and more token efficient. We also experiment over the ordering of the image, preamble, and task; finding that preamble, followed by image, followed by task performs best, though by a small margin.

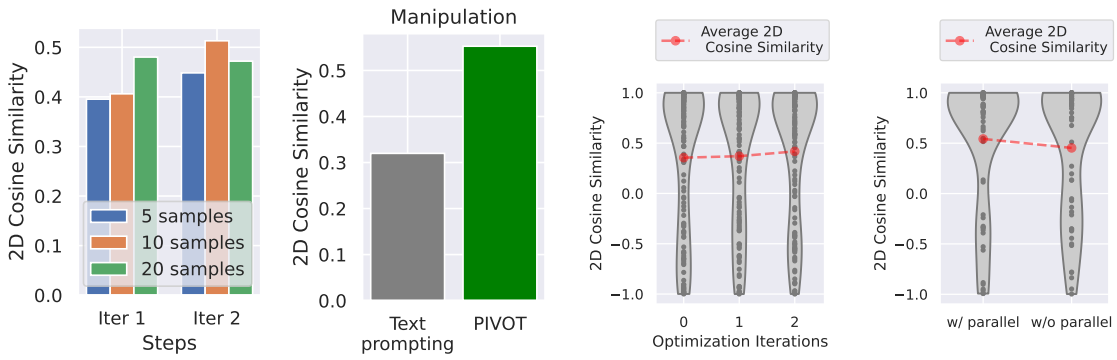
Visual prompts. Aspects of the style of visual prompts has been examined in prior works [59, 46], such as the color, size, shading, and shape. Herein, we investigate aspects central to PIVOT– the number of samples and the importance of the visual prompt itself. An ablation over the number of samples is shown in Figure 7 where we note an interesting trend: more samples leads to better initial answers, but worse optimization. Intuitively, a large number of samples supports good coverage for the initial answer, but with too many samples the region of the image around the correct answer gets crowded and causes significant issues with occlusions. For our tasks, we found 10 samples to best trade off between distributional coverage and maintaining sufficient visual clarity.



(a) (b)

Figure 6 | Offline evaluation results for navigation task with L2 distance (lower is better). Ablation over (6a) iterations and parallel calls and (6b) text-only baseline.

To understand the necessity of the visual prompt itself, we compare to a language only baseline, where a VLM selects from a subset of language actions that map to robotic actions. For the manipulation task, the VLM is given an image and task and selects from move "right", "'left", "up", and "down". A similar navigation benchmark is described in Appendix B. We see in Figure 7 and Figure 6 that PIVOT outperforms text by a large margin. We note here that we do not compare to learned approaches that require training or finetuning as our focus is on zero-shot understanding. We believe many such approaches would perform well in distribution on these tasks, but would have limited generalization on out of distribution tasks.



(a) Number of samples (b) Text-only baseline (c) Iterations (d) Parallel calls

Figure 7 | Offline evaluation results for manipulation tasks with cosine similarity (higher is better).

Iterative optimization. To understand the effect of the iterative optimization process, we ablate over the number of iterations and parallel calls. In Figures 5, 6, and 7, we find that increasing iterations improves performance, increasing parallel calls improves performance, and crucially doing

both together performs the best. This echos the findings in the online evaluations above.

4.5. Scaling

We observe that PIVOT scales across varying sizes of VLMs on the mobile manipulator offline evaluation (results measured in terms of cosine similarity and L2 error between PIVOT and demonstration data ground truth in Figure 8). In particular, we compare PIVOT using four sizes of the Gemini family of models [17] which we labeled a to d, with progressively more parameters. We find that performance increases monotonically across each model size. Although there are still significant limitations and capabilities gaps, we see this scaling as a promising sign that PIVOT can leverage next-generation foundation models with increasing model size and capabilities [17].

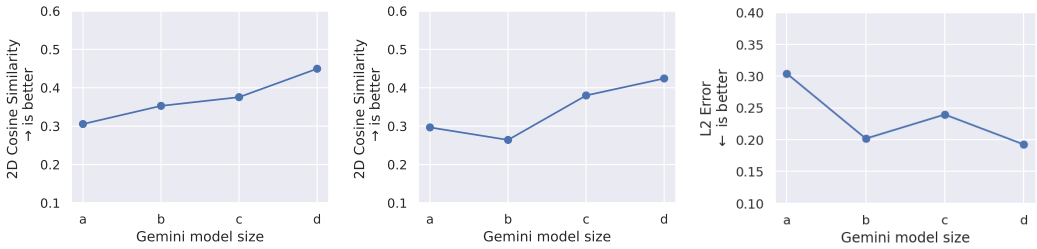


Figure 8 | Scaling results of first iteration visual prompting performance across Gemini model [17] sizes show that PIVOT scales well with improved VLMs. Left and center plots are manipulation (pick up objects, move one object next to another), right plot is navigation.

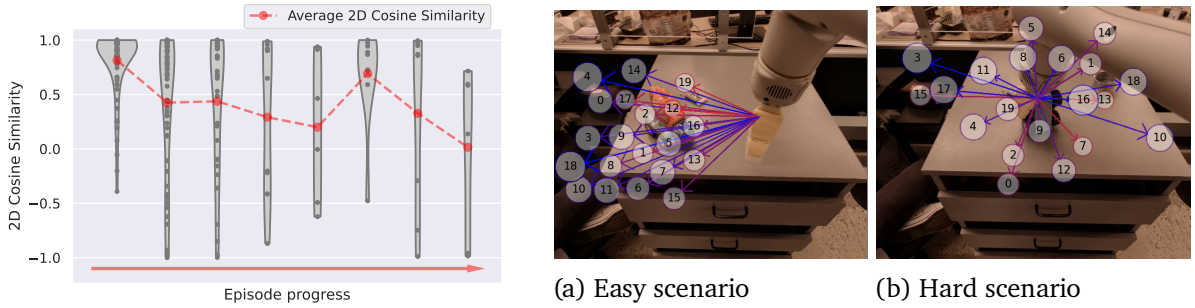
4.6. Limitations

In this work, we evaluate PIVOT using state-of-the-art VLMs and their zero-shot capabilities. We note that the base models have not been trained on in-domain data for robotic control or physical reasoning represented by visual annotation distributions. While the exact failure modes may be specific to particular underlying VLMs, we continue to observe trends which may reflect broad limitation areas. We expect that future VLMs with improved generalist visual reasoning capabilities will likewise improve in their visual annotation and robotics reasoning capabilities, and the general limitations of PIVOT on current state-of-the-art VLMs may serve to highlight potential risks and capabilities gaps, that point to interesting open areas for future work.

3D understanding. While VLMs only take 2D images as visual inputs, in principle the image annotations and transformations applied via PIVOT can represent 3D queries as well. Although we examined expressing depth values as part of the annotations using colors and label sizes (and described what they map to within a preamble prompt), we have observed that none of the VLMs we tested are capable of reliably choosing actions based on depth. Beyond this, generalizing to higher dimensional spaces such as rotation poses even additional challenges. We believe more complex visuals (e.g. with shading to give the illusion of depth) may address some of these challenges, but ultimately, the lack of 3D training data in the underlying VLM remains the bottleneck. It is likely that training on either robot specific data or with depth images may alleviate these challenges.

Interaction and fine-grained control. During closed-loop visuomotor tasks (e.g., for first-person navigation tasks, or manipulation task with hand-mounted cameras), images can often be characterized by increasing amounts of occlusion, where the objects of interest can become no longer visible if the cameras are too close. This affects PIVOT and the VLM's capacity for decision-making e.g., determining when to grasp, whether to lift an object, or approaching an object from the correct side to push. This is visualized in Figure 9, where errors over the trajectory are shown. These errors are a

result of both occlusions, resolution of the image, but perhaps more crucially, a lack of training data from similar interactions. In this case, training on embodied or video data may be a remedy.



(a) Easy scenario (b) Hard scenario

Figure 9 | PIVOT performance over "move near" trajectories, which pick up an object and move them near another. Initially performance is high, but decreases as the robot approaches the grasp and lift (due to objects being obscured and the VLM not understanding the subtlety of grasping). After the grasp, the performance increases as it moves to the other object, but again decreases as it approaches.

Greedy behavior. Though we find iterative optimization alleviates many simple errors, we also find that the underlying VLM often displays greedy, myopic behaviors for multi-step decision-making tasks. For instance, given the task "move the apple to the banana", the VLM may recommend immediately approaching the banana rather than the apple first. We believe these mistakes may lessen with more capable VLMs, or with more in-domain examples provided either via fine-tuning or via few-shot prompting with e.g., a history of actions as input context to the VLM to guide future generated acitons.

Vision-language connection reasoning errors. We find that though overall the thought process of the VLM is reasonable, it stochastically connects the thought process to the incorrect arrow. This issue appears to be a challenge of autoregressive decoding, once the number is decoded, the VLM must justify it, even if incorrect, and thus hallucinates an otherwise reasonable thought process. Many of these errors are remedied through the optimization process of PIVOT, but we believe further improvements could be made with tools from robust optimization.

5. Conclusion

PIVOT presents a promising step towards leveraging VLMs for spatial reasoning zero-shot, and suggests new opportunities to cast traditionally challenging problems (e.g., low-level robotic control) as vision ones. PIVOT can be used for tasks such as controlling a robot arm that require outputting spatially grounded continuous values with a VLM zero shot. This is made possible by representing spatial concepts in the image space and then iteratively refining those by prompting a VLM. Built on iterative optimization, PIVOT stands to benefit from other sampling initialization procedures, optimization algorithms, or search-based strategies. Furthermore, we have identified several limitations of current state-of-the-art models that limits performance herein (e.g., 3D understanding and interaction). Therefore, adding datasets representing these areas presents an interesting avenue for future work; along with directly finetuning task specific data. More importantly, though, we expect the capabilities of VLMs to improve over time, hence the zero-shot performance of PIVOT is likely to improve as well, as we have investigated in our scaling experiments. We believe that this work can be seen as an attempt to unify internet-scale general vision-language tasks with physical problems in the real world by representing them in the same input space. While the majority of our experiments focus on robotics, the algorithm can generally be applied to problems that require outputting continuous values with a VLM.

Acknowledgements

We thank Kanishka Rao, Jie Tan, Carolina Parada, James Harrison, Nik Stewart, and Jonathan Tompson for helpful discussions and providing feedback on the paper.

References

[1] Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[2] Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

[3] Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

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

[5] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[6] Cai, M., Liu, H., Mustikovela, S. K., Meyer, G. P., Chai, Y., Park, D., and Lee, Y. J. Making large multimodal models understand arbitrary visual prompts. *arXiv preprint arXiv:2312.00784*, 2023.

[7] Chen, B., Xia, F., Ichter, B., Rao, K., Gopalakrishnan, K., Ryoo, M. S., Stone, A., and Kappler, D. Open-vocabulary queryable scene representations for real world planning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11509–11522. IEEE, 2023.

[8] Chen, B., Xu, Z., Kirmani, S., Ichter, B., Driess, D., Florence, P., Sadigh, D., Guibas, L., and Xia, F. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. *arXiv preprint arXiv:2401.12168*, 2024.

[9] Chen, X., Djolonga, J., Padlewski, P., Mustafa, B., Changpinyo, S., Wu, J., Ruiz, C. R., Goodman, S., Wang, X., Tay, Y., et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023.

[10] Cui, Y., Niekum, S., Gupta, A., Kumar, V., and Rajeswaran, A. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for Dynamics and Control Conference*, pp. 893–905. PMLR, 2022.

[11] De Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. A tutorial on the cross-entropy method. *Annals of operations research*, 134:19–67, 2005.

[12] Dorbala, V. S., Sigurdsson, G., Piramuthu, R., Thomason, J., and Sukhatme, G. S. Clip-nav: Using clip for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2211.16649*, 2022.

[13] Firoozi, R., Tucker, J., Tian, S., Majumdar, A., Sun, J., Liu, W., Zhu, Y., Song, S., Kapoor, A., Hausman, K., et al. Foundation models in robotics: Applications, challenges, and the future. *arXiv preprint arXiv:2312.07843*, 2023.

[14] Gadre, S. Y., Wortsman, M., Ilharco, G., Schmidt, L., and Song, S. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23171–23181, 2023.

[15] Gao, J., Sarkar, B., Xia, F., Xiao, T., Wu, J., Ichter, B., Majumdar, A., and Sadigh, D. Physically grounded vision-language models for robotic manipulation. *arXiv preprint arXiv:2309.02561*, 2023.

[16] Gemini, T., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[17] Gemini Team, G. Gemini: A family of highly capable multimodal models. Technical report, Google, 2023. URL https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf.

[18] Gu, J., Kirmani, S., Wohlhart, P., Lu, Y., Arenas, M. G., Rao, K., Yu, W., Fu, C., Gopalakrishnan, K., Xu, Z., et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv preprint arXiv:2311.01977*, 2023.

[19] Hu, Y., Xie, Q., Jain, V., Francis, J., Patrikar, J., Keetha, N., Kim, S., Xie, Y., Zhang, T., Zhao, Z., et al. Toward general-purpose robots via foundation models: A survey and meta-analysis. *arXiv preprint arXiv:2312.08782*, 2023.

[20] Huang, C., Mees, O., Zeng, A., and Burgard, W. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10608–10615. IEEE, 2023.

[21] Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pp. 9118–9147. PMLR, 2022.

[22] Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[23] Huang, W., Wang, C., Zhang, R., Li, Y., Wu, J., and Fei-Fei, L. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[24] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015.

[25] Jiang, Y., Gupta, A., Zhang, Z., Wang, G., Dou, Y., Chen, Y., Fei-Fei, L., Anandkumar, A., Zhu, Y., and Fan, L. Vima: General robot manipulation with multimodal prompts. *arXiv*, 2022.

[26] Koh, J. Y., Lo, R., Jang, L., Duvvur, V., Lim, M. C., Huang, P.-Y., Neubig, G., Zhou, S., Salakhutdinov, R., and Fried, D. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*, 2024.

[27] Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[28] Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[29] Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[30] Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., and Zeng, A. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

[31] Lin, K., Agia, C., Migimatsu, T., Pavone, M., and Bohg, J. Text2motion: From natural language instructions to feasible plans. *arXiv preprint arXiv:2303.12153*, 2023.

[32] Liu, B., Jiang, Y., Zhang, X., Liu, Q., Zhang, S., Biswas, J., and Stone, P. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

[33] Liu, D., Dong, X., Zhang, R., Luo, X., Gao, P., Huang, X., Gong, Y., and Wang, Z. 3daxiesprompts: Unleashing the 3d spatial task capabilities of gpt-4v. *arXiv preprint arXiv:2312.09738*, 2023.

[34] Liu, Z., Bahety, A., and Song, S. Reflect: Summarizing robot experiences for failure explanation and correction. *arXiv preprint arXiv:2306.15724*, 2023.

[35] Ma, Y. J., Liang, W., Wang, G., Huang, D.-A., Bastani, O., Jayaraman, D., Zhu, Y., Fan, L., and Anandkumar, A. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

[36] Mirchandani, S., Xia, F., Florence, P., Ichter, B., Driess, D., Arenas, M. G., Rao, K., Sadigh, D., and Zeng, A. Large language models as general pattern machines. *arXiv preprint arXiv:2307.04721*, 2023.

[37] OpenAI. Gpt-4v(ision) system card. Technical report, OpenAI, 2023. URL https://openai.com/research/gpt-4v-system-card.

[38] Padalkar, A., Pooley, A., Jain, A., Bewley, A., Herzog, A., Irpan, A., Khazatsky, A., Rai, A., Singh, A., Brohan, A., et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[39] Pryzant, R., Iter, D., Li, J., Lee, Y. T., Zhu, C., and Zeng, M. Automatic prompt optimization with" gradient descent" and beam search. *arXiv preprint arXiv:2305.03495*, 2023.

[40] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

[41] Raman, S. S., Cohen, V., Rosen, E., Idrees, I., Paulius, D., and Tellex, S. Planning with large language models via corrective re-prompting. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022.

[42] Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T., et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

[43] Shah, D., Equi, M. R., Osiński, B., Xia, F., Ichter, B., and Levine, S. Navigation with large language models: Semantic guesswork as a heuristic for planning. In *Conference on Robot Learning*, pp. 2683–2699. PMLR, 2023.

[44] Shah, D., Osiński, B., Levine, S., et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on Robot Learning*, pp. 492–504. PMLR, 2023.

[45] Shridhar, M., Manuelli, L., and Fox, D. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pp. 894–906. PMLR, 2022.

[46] Shtedritski, A., Rupprecht, C., and Vedaldi, A. What does clip know about a red circle? visual prompt engineering for vlms. *arXiv preprint arXiv:2304.06712*, 2023.

[47] Silver, T., Dan, S., Srinivas, K., Tenenbaum, J. B., Kaelbling, L. P., and Katz, M. Generalized planning in pddl domains with pretrained large language models. *arXiv preprint arXiv:2305.11014*, 2023.

[48] Singh, I., Blukis, V., Mousavian, A., Goyal, A., Xu, D., Tremblay, J., Fox, D., Thomason, J., and Garg, A. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11523–11530. IEEE, 2023.

[49] Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., and Anandkumar, A. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[50] Wang, Y.-J., Zhang, B., Chen, J., and Sreenath, K. Prompt a robot to walk with large language models. *arXiv preprint arXiv:2309.09969*, 2023.

[51] Wang, Z., Cai, S., Liu, A., Ma, X., and Liang, Y. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.

[52] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[53] Wen, L., Yang, X., Fu, D., Wang, X., Cai, P., Li, X., Ma, T., Li, Y., Xu, L., Shang, D., et al. On the road with gpt-4v (ision): Early explorations of visual-language model on autonomous driving. *arXiv preprint arXiv:2311.05332*, 2023.

[54] Wu, J., Antonova, R., Kan, A., Lepert, M., Zeng, A., Song, S., Bohg, J., Rusinkiewicz, S., and Funkhouser, T. Tidybot: Personalized robot assistance with large language models. *arXiv preprint arXiv:2305.05658*, 2023.

[55] Xu, H., Chen, Y., Du, Y., Shao, N., Wang, Y., Li, H., and Yang, Z. Gps: Genetic prompt search for efficient few-shot learning. *arXiv preprint arXiv:2210.17041*, 2022.

[56] Xu, J., Zhou, X., Yan, S., Gu, X., Arnab, A., Sun, C., Wang, X., and Schmid, C. Pixel aligned language models. *arXiv preprint arXiv:2312.09237*, 2023.

[57] Yan, A., Yang, Z., Zhu, W., Lin, K., Li, L., Wang, J., Yang, J., Zhong, Y., McAuley, J., Gao, J., et al. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. *arXiv preprint arXiv:2311.07562*, 2023.

[58] Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*, 2023.

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

[60] Yang, Z., Li, L., Lin, K., Wang, J., Lin, C.-C., Liu, Z., and Wang, L. The dawn of lmms: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*, 9(1):1, 2023.

[61] Yu, L., Poirson, P., Yang, S., Berg, A. C., and Berg, T. L. Modeling context in referring expressions. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pp. 69–85. Springer, 2016.

[62] Yu, W., Gileadi, N., Fu, C., Kirmani, S., Lee, K.-H., Arenas, M. G., Chiang, H.-T. L., Erez, T., Hasenclever, L., Humplik, J., et al. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

[63] Zeng, A., Florence, P., Tompson, J., Welker, S., Chien, J., Attarian, M., Armstrong, T., Krasin, I., Duong, D., Sindhwani, V., et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pp. 726–747. PMLR, 2021.

[64] Zeng, A., Attarian, M., Ichter, B., Choromanski, K., Wong, A., Welker, S., Tombari, F., Purohit, A., Ryoo, M., Sindhwani, V., et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

[65] Zheng, B., Gou, B., Kil, J., Sun, H., and Su, Y. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*, 2024.

[66] Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., and Ba, J. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022.

Appendix

A. Robotic Embodiments

Mobile Manipulator Navigation. Shown in Figure 3 (a), we use a mobile manipulator platform for navigation tasks. We use the image from a fixed head camera and annotate the image with arrows originating from the bottom center of the image to represent the 2D action space. After PIVOT identifies the candidate action in the pixel space, we then use the on-board depth camera from the robot to map it to a 3D target location and command the robot to move toward the target (with a maximum distance of 1.0m). We evaluate PIVOT on both a real robot and on an offline dataset. For real robot evaluation, we designed four scenarios where the robot is expected to reach a target location specified either through an object of interest (e.g. find apple) or through an indirect instruction (e.g. find a place to take a nap). For offline evaluation, we created a dataset of 60 examples from prior robot navigation data with labeled ground truth targets. More details on the task and dataset can be found in Appendix Section B.

Mobile Manipulator Manipulation. Shown in Figure 3 (b), we use a mobile manipulator platform for manipulation tasks. We use the image from a fixed head camera and annotate the image with arrows originating from the end-effector in camera frame, for which each arrow represents a 3D relative Cartesian end-effector position (x, y, z). To handle the z-dimension height, we study two settings: one where height is represented through color grading (a red to blue spectrum) and one where the arm only uses fixed-height actions. Gripper closing actions are not shown as visual annotations but instead expressed through text prompts. Note that although the end-effector has rotational degrees of freedoms, we fix these due to the difficulty of expressing them with visual prompting, as is discussed in Section 4.6. We evaluate PIVOT on both real robot and an offline dataset. For real robot evaluation, we study three tabletop manipulation tasks which require combining semantic and motion reasoning. Success criteria consists of binary object reaching success, number of steps taken for successful reaching trajectories, and grasping success when applicable. For offline evaluation, we use demonstration data from the RT-X mobile manipulator dataset [38]. We sample 10 episodes of pick demonstrations for most of our offline evaluations, and 30 episodes of move near demonstrations for our interaction Figure 9. More details on the results can be found in Appendix Section D.

Franka. Shown in Figure 3 (c) we use the Franka for manipulation. We use the image from a wrist mounted camera and annotate the image with arrows originating from the center of the camera frame, for which each arrow represents a 3D relative Cartesian end-effector position $(x, y, z$, where the z dimension is captured with a color spectrum from red to blue). We examine both pick tasks and place tasks, with 5 objects for each task. More details on the results can be found in Appendix Section F.

RAVENS [63]. Show in Figure 3 (d), we use the RAVENS simulation domain for pick and place manipulation. We use the image from an overhead camera and annotate the image with pick and place locations, following the action representation in Zeng et al. [63]. This action space allows us to evaluate higher-level action representations. More details on the results can be found in Appendix Section E.

B. Mobile Manipulator Navigation Offline Evaluation

B.1. Dataset

We create an offline dataset of 60 examples using images collected from the on-robot camera sensor by walking the robot in an indoor environment. For each example, we provide an instruction and a associated location in the image space as the target. We categorize our tasks into three types: 1) in-view finding, where the robot is tasked to approach an object within the line of sight, 2) semantic understanding, where the instruction implicitly refers to an object in view 3) out-of-view finding, where the object of interest is not visible from the current view with arrow annotations, but can be seen in past images from different locations. Figure 10 shows examples of the three task categories.

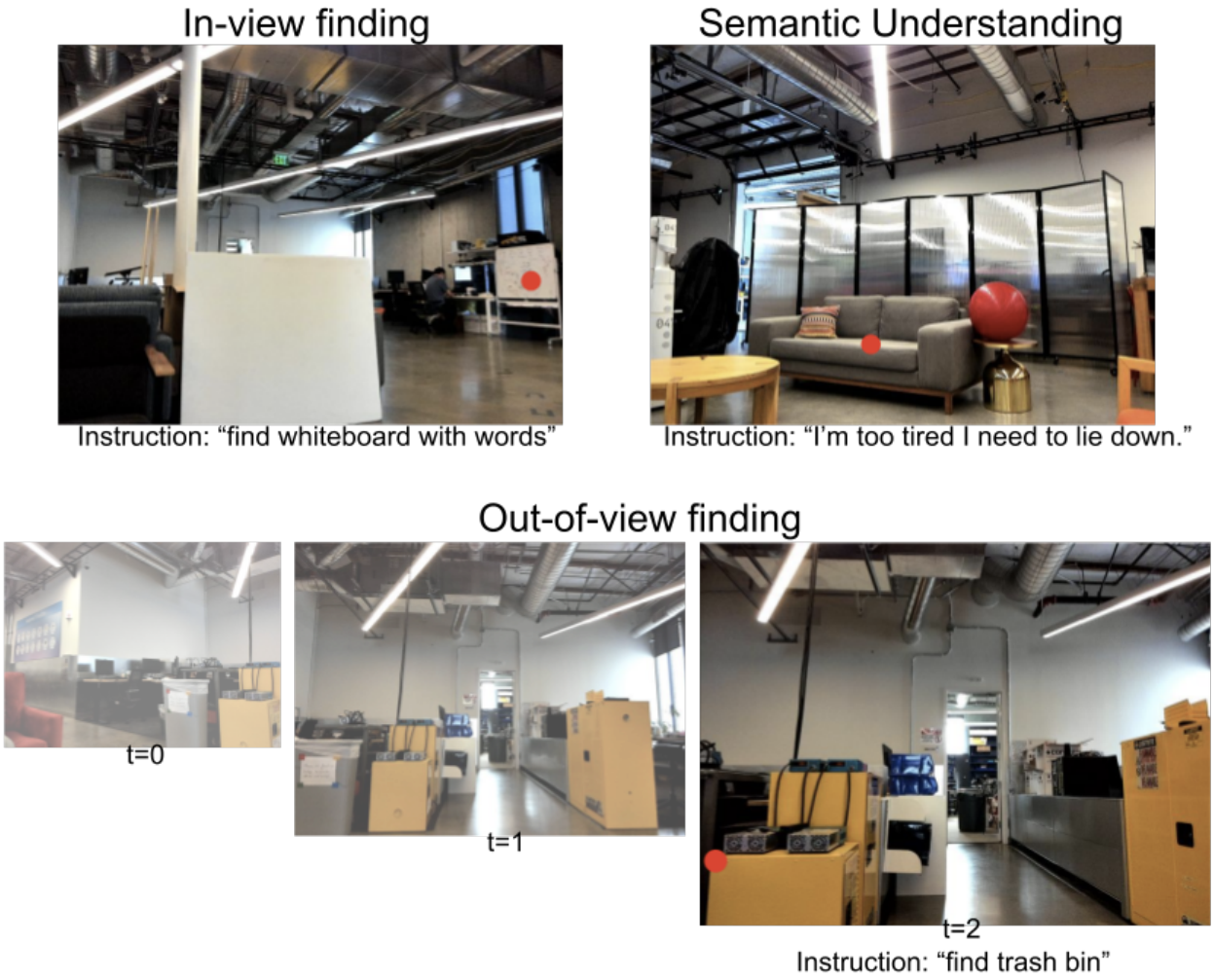


Figure 10 | Example tasks in the offline navigation dataset from different task categories. Red dot denotes the ground truth target.

B.2. Evaluation Results

Table 3 shows the detailed evaluation results of PIVOT on the offline navigation dataset. We measure the accuracy of the PIVOT output by its deviation from the target point in image space normalized by the image width and break it down into the three task categories. We report mean and standard deviation for three runs over the entire dataset.

As seen in the table, by using the parallel call to robustify the VLM output we see significant improvements over running VLM only once (0 parallel) and running PIVOT for multiple iterations also improves accuracy of the task. However, increasing the parallel calls or the iteration number further did not achieve notably better performance.

Table 3 | Navigation offline evaluation measured in L2 loss (lower the better).

In-View Tasks			
	1 iter	2 iter	3 iter
0 parallel	0.21 ± 0.002	0.21 ± 0.007	0.19 ± 0.007
2 parallel	0.19 ± 0.004	0.2 ± 0.012	0.18 ± 0.005
3 parallel	0.19 ± 0.003	0.17 ± 0.007	0.17 ± 0.009
Semantic Tasks			
	1 iter	2 iter	3 iter
0 parallel	0.23 ± 0.012	0.2 ± 0.006	0.19 ± 0.025
2 parallel	0.26 ± 0.015	0.21 ± 0.02	0.2 ± 0.02
3 parallel	0.21 ± 0.01	0.19 ± 0.04	0.19 ± 0.01
Out-of-View Tasks			
	1 iter	2 iter	3 iter
0 parallel	0.44 ± 0.04	0.38 ± 0.015	0.39 ± 0.032
2 parallel	0.38 ± 0.001	0.39 ± 0.02	0.39 ± 0.02
3 parallel	0.37 ± 0.01	0.38 ± 0.026	0.39 ± 0.05

We compared our proposed approach, which reasons in image-space with image annotations, with reasoning in text without annotated images. In this text-based baseline, we provide the same image and navigation query to the VLM, but we ask the VLM to imagine that the image is split into 3 rows and 3 columns of equal-sized regions and output the name of one of those regions (e.g. “top left", "bottom middle"). We then compute the distance between the center of the selected region to the ground truth target point. Given that we are not performing iterative optimization with this text baseline, we compare its results against PIVOT with just 1 iteration and 0 parallel. See results in Table 4. For GPT-4V, the text baseline incurs higher mean and standard deviation of errors across all tasks.

Table 4 | Reasoning with Image Annotations vs. with Text for Navigation offline evaluations measured in L2 loss (lower the better).

Method	In-View	Semantic	Out-of-View
Image	0.21 ± 0.002	0.23 ± 0.012	0.44 ± 0.04
Text	0.26 ± 0.15	0.35 ± 0.14	0.46 ± 0.31

C. Mobile Manipulator Manipulation Online Evaluation

In addition to the quantitative evaluation trials for the real-world manipulation experts described in Section 4.2, we also showcase additional evaluation rollouts in Figure 11. Qualitatively, we find that PIVOT is able to recover from inaccuracies in action prediction, such as those which may result from imperfect depth perception or action precision challenges.

D. Mobile Manipulator Manipulation Offline Evaluation

Using the offline mobile manipulator dataset described in Section A, we additionally ablate the text prompt herein. In Figure 13 we consider the performance of zero-shot and few-shot prompting as well as chain of thought [52] and direct prompting. We find in general that neither is a panacea, though zero-shot chain of thought performs best, few-shot direct prompting performs similarly and is significantly more token efficient. In Figure 14 we consider the effect that the order of the prompt has on performance. The distinct elements of the prompt are the preamble (which describes the high level goal), the task (which describes the specific task the robot is attempting to perform), and the image. Examples of these prompts can be seen in Appendix Section H. We find a small amount of variation in performance between orders, with preamble, image, and task resulting in the highest performance. We hypothesize that this order most closely mirrors the training mixture.

To illustate the limitation of our method decribed in Fig. 9 better, we visualize two episodes of the mobile manipulator manipulation offline eval in Fig. 12. The figure shows that at the beginning of the episode where it is clear where to move, our method tend to generate accurate predictions while in the middle of the episode where there are interactions, our method struggles to generate correct actions.

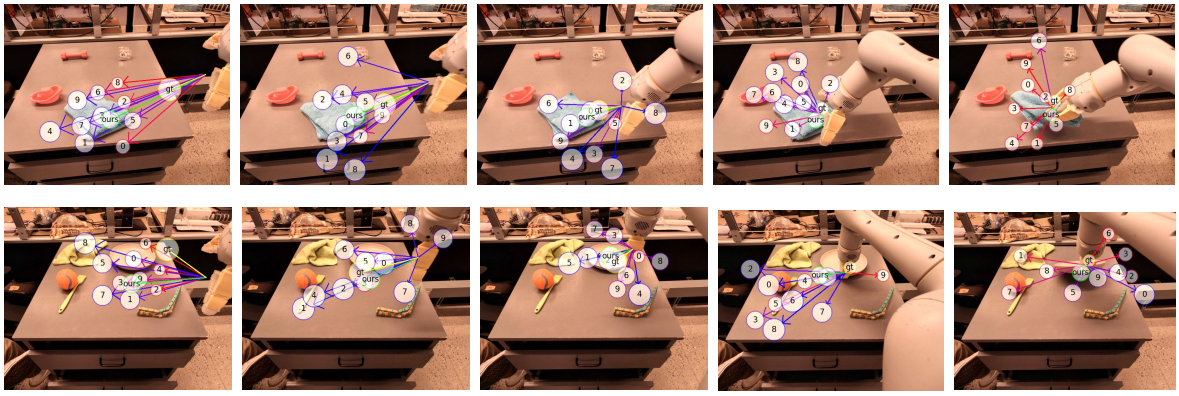


Figure 12 | Two episodes of mobile manipulator manipulation offline evaluation. It shows our method can generate reasonable actions following the arrow annotations.

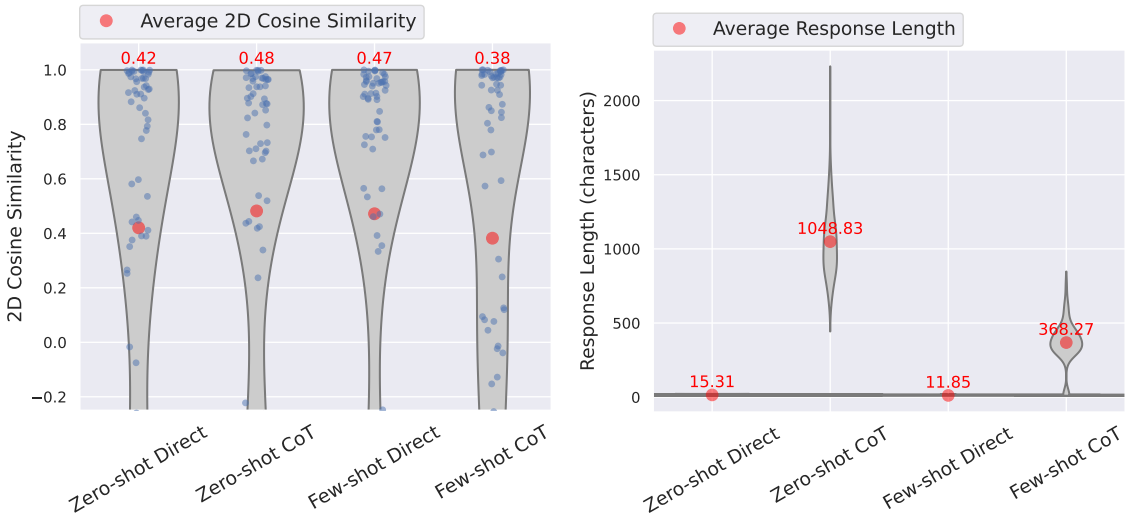


Figure 13 | Ablation of few-shot vs. zero-shot and CoT vs. direct performance on manipulation domain. The best performing combination is zero-shot CoT. However, direct models can achieve similar performance with much fewer output tokens thus more token efficient.

E. RAVENS Online Simulation Evaluation

We create a suite of evaluation tasks in which the robot must pick a specified fruit and place it in a specified bowl. There are three fruits in the scene (banana, strawberry, pear) and three bowls with different colors (blue, green, yellow). Each task takes the form "pick the {fruit} and place it in the {color} bowl." Given the task goal, we parse the source object and the target object, and independently prompt the VLM to get the pick and place locations corresponding to these two objects respectively. Refer to Appendix H for the prompt we use. In Figure 15 we report evaluation over five random instances. Here we specifically report the error with respect to ground truth pick and place locations over each iteration of visual prompting. We see that the error generally decreases in the first few iterations and eventually converges. In most settings the chosen pick and place locations are close to the desired objects, yet the VLM lacks the often ability to precisely choose points that allow it to execute the task successfully in one action.

F. Franka Online Evaluation

We evaluate PIVOT in a real world manipulation setting using a Franka robot arm with a wrist-mounted camera and a 4D relative Cartesian delta action space. We study 7 tabletop manipulation tasks involving grasping and placing various objects, and analyze three version of PIVOT with varying numbers of optimization iterations and number of parallel PIVOT processes. Each task is evaluated for two trials, for which we record intermediate reaching success rates for reaching the correct XY and YZ proximities for the target object (where in the camera frame the x-axis is into and out of the page, the y-axis is left and right, and the z axis is up and down), as well as the overall number of timesteps taken for successful trials. As shown in Table 5, we find that all instantiations of PIVOT are able to achieve non-zero success, but increasing the number of optimization iterations and number of parallel processes increases performance and stability. Rollouts are shown in Figure 16.

Table 5 | Manipulation results on the real-world Franka setting shown in Figure 3 (c), where "XY" and "YZ" indicate success rates for reaching the relevant object XY and YZ proximities respectively and "Steps" indicates the number of steps taken if successful finished the task. We observe that while all approaches are able to achieve some non-zero success, iteration and parallel calls improve performance and efficiency of the policy.

	No Iterations No Parallel			3 Iterations No Parallel			3 Iterations 3 Parallel		
Task	XY	YZ	Steps	XY	YZ	Steps	XY	YZ	Steps
Place saltshaker on the blue plate	0%	0%	-	0.5%	0%	-	50%	50%	3.0
Place peppershaker on the pink plate	100%	100%	8.0	100%	100%	3.5	50%	50%	4.0
Grasp the pink cup	50%	50%	7.0	0%	50%	-	0%	50%	-
Grasp the pepper shaker	50%	50%	8.0	0%	50%	-	0%	50%	-
Grasp the blue cup	0%	50%	-	0%	50%	-	0%	50%	-
Grasp the red ketchup bottle	0%	50%	-	0%	0%	-	100%	100%	6.0
Grasp the can	0%	0%	-	0%	0%	-	50%	50%	3.0
Average	25%	38%	7.8	28%	31%	3.5	34%	59%	4.4

G. Visual Annotation Sensitivity

Inspired by prior works which find interesting biases and limitations of modern VLMs on understanding visual annotations [46, 59, 60], we analyze the ability of state-of-the-art VLMs to understand various types of arrow annotations. We generate two synthetic datasets: one toy dataset of various styles of CV2 [24] arrows overlaid on a white background, and a more realistic dataset of various styles of object-referential arrows overlaid on a real-world robotics scene. The datasets adjust parameters such as arrow color, arrow thickness, and relative arrowhead size. In the first dataset, we query VLMs to classify the direction of the arrows, which studies the effect of styling on the ability of VLMs to understand absolute arrow directions; examples are shown in Figure 17. In the second dataset, we query VLMs to select the arrow which points at a specified object out of multiple objects, which studies the effect of styling on the ability of VLMs to understand relative and object-centric arrow directions. The second dataset contains scenes with various objects, which we categorize into "Easy" (plates, boxes, cubes), "Medium" (cups, bags, mugs), "Hard" (hangers, toys), and "Very Hard" (brushes, eccentric objects).

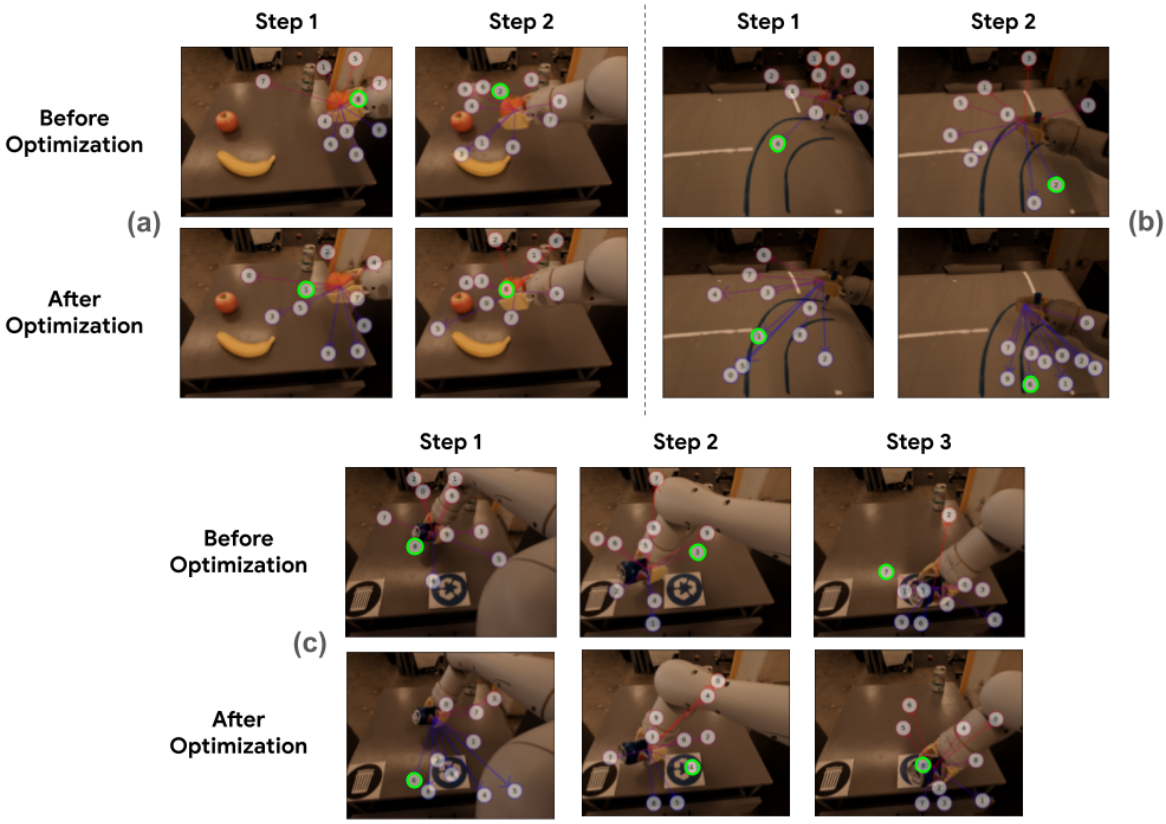


Figure 11 | Evaluating PIVOT on real world mobile manipulator tabletop manipulation scenarios which require a combination of semantic reasoning and action understanding. Using 3 optimization iterations on the real world mobile manipulator, we see promising successes for (a) “move the orange to complete the smiley face represented by fruits”, (b) “use the marker to trace a line down the blue road”, and (c) “sort the object it is holding to the correct piece of paper”.

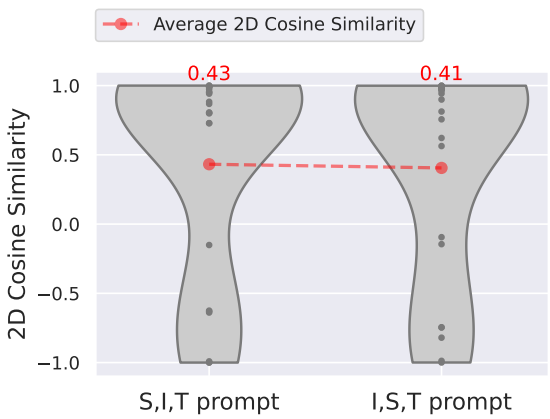


Figure 14 | Ablation of order of preamble, image, and task on mobile manipulation domain. We found it is beneficial to put the image closer to the end of the prompt, though the effect is marginal. P, I, T means preamble, followed by image and task description, and I, P, T means image followed by preamble and task description.

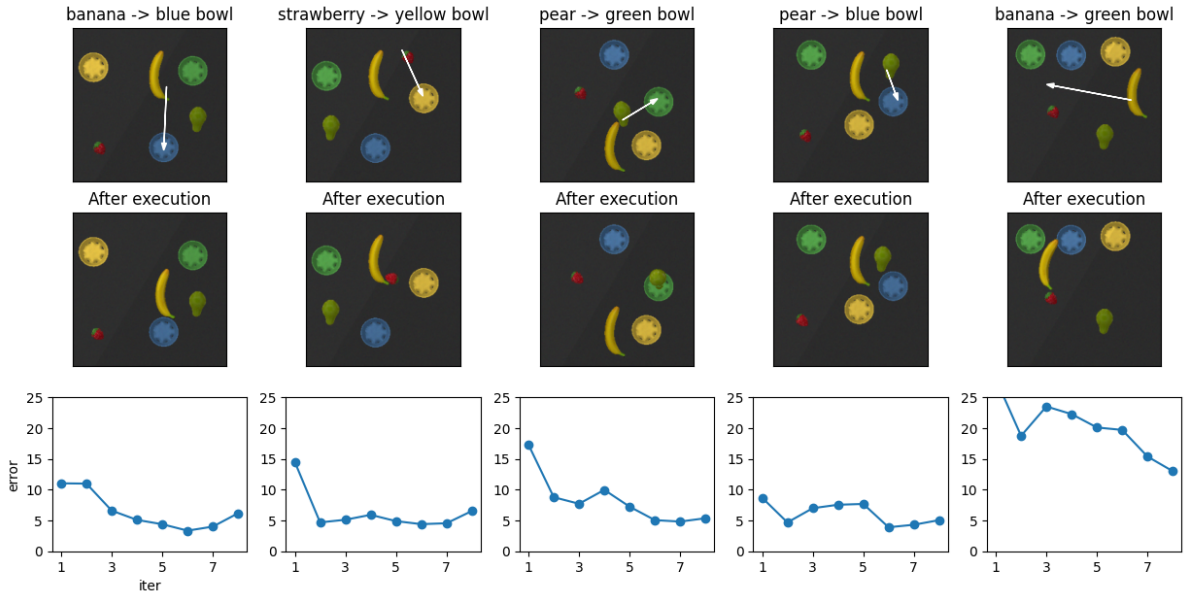


Figure 15 | RAVENS evaluations. Each column shows a different task instance. Title: pick object followed by place object. Top row: initial image with pick and place locations predicted by VLM indicated by white arrow. Middle row: result after executing action. Bottom row: L2 distance between predicted and ground truth locations (averaged for both pick location and place location), over iterations.

Table 6 | Visual annotation arrow robustness of VLMs on a synthetic toy arrow dataset. For various colored arrows with different thicknesses, different sized arrowheads, and different absolute directions, we evaluate the robustness of GPT-4V on correctly classifying the absolute arrow direction.

	Arrow Thickness			Arrowhead Size			Direction			
Color	2	4	6	0.1	0.3	0.5	up+right	down+right	up+left	down+left
red	96%	92%	96%	97%	94%	88%	100%	75%	75%	92%
orange	92%	88%	96%	100%	91%	84%	100%	100%	50%	83%
yellow	88%	88%	100%	100%	94%	84%	93%	100%	75%	67%
green	96%	92%	96%	100%	100%	88%	100%	92%	92%	83%
blue	92%	92%	88%	91%	91%	88%	100%	17%	100%	100%
purple	100%	96%	96%	97%	97%	97%	100%	92%	92%	92%

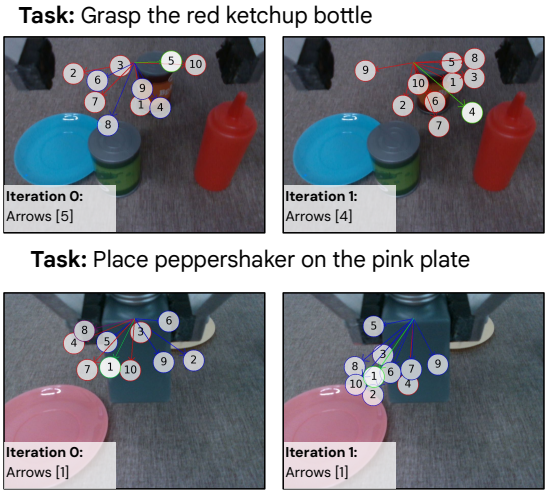


Figure 16 | Rollouts on the Franka environment.

Table 7 | Visual annotation arrow robustness of VLMs on an object-referential arrow dataset. For various colored arrows with different thicknesses, different sized arrowheads, and different absolute directions, we evaluate the robustness of GPT-4V on correctly selecting the arrow which refers to a specified object.

	Arrow Thickness			Arrowhead Size			Target Object			
Color	2	4	6	0.1	0.3	0.5	Easy	Medium	Hard	Very Hard
red	42%	33%	33%	50%	33%	25%	44%	100%	0%	0%
orange	25%	25%	25%	25%	25%	25%	0%	100%	0%	0%
yellow	67%	58%	50%	83%	58%	33%	100%	33%	56%	44%
green	50%	58%	50%	83%	58%	33%	100%	33%	56%	44%
blue	42%	36%	33%	36%	50%	25%	100%	33%	22%	0%
purple	33%	50%	50%	58%	58%	17%	89%	22%	56%	11%

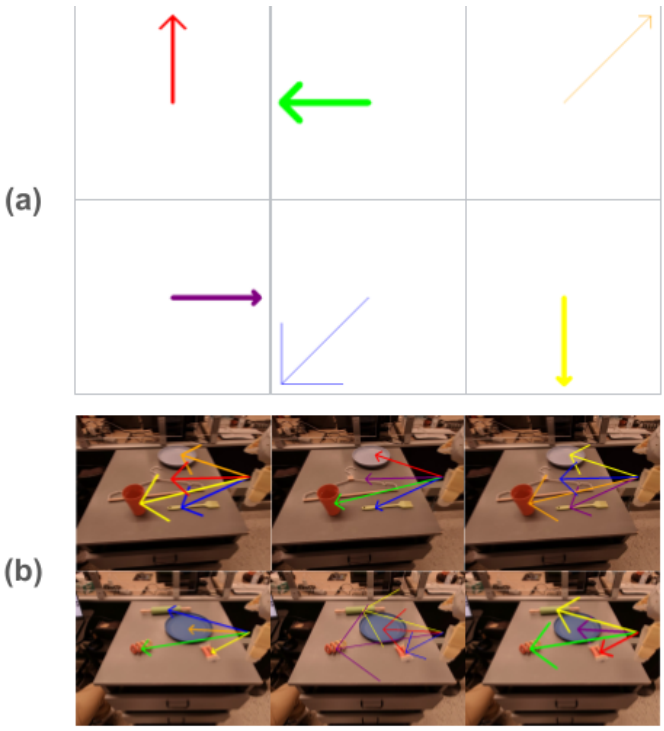


Figure 17 | Examples of procedurally generated datasets studying the robustness of VLMs for understanding visual annotation arrow styles. (a) focuses on absolute direction understanding of single arrows on blank backgrounds. (b) focuses on object-relative arrow understanding in realistic scenes.

H. Prompts

H.1. RefCOCO prompt

```
Your goal is to find the OBJECT in this scene.  I have annotated the image with numbered
circles.  Choose the 3 numbers that have the most overlap with the OBJECT. If there
are no points with overlap, then don't choose any points.  You are a five-time world
champion in this game.  Give a one sentence analysis of why you chose those points.
Provide your answer at the end in a json file of this format:
{"points":  [] }
```

H.2. Navigation prompt

```
I am a wheeled robot that cannot go over objects.  This is the image I'm seeing right
now.  I have annotated it with numbered circles.  Each number represent a general
direction I can follow.  Now you are a five-time world-champion navigation agent and
your task is to tell me which circle I should pick for the task of:  {INSTRUCTION}?
Choose {K} best candidate numbers.  Do NOT choose routes that goes through objects.
Skip analysis and provide your answer at the end in a json file of this form:
{"points":  [] }
```

H.3. RAVENS prompt

```
which number markers are closest to the {OBJECT}?  Reason and express the final answer
as 'final answer' followed by a list of the closest marker numbers.
```

H.4. Manipulation online eval prompt

Direct

```
 What number arrow should the robot follow to task?
```

```
 Rules:  - You are looking at an image of a robot in front of a desk trying to arrange
objects.  The robot has an arm and a gripper with yellow fingers.  - The arrows in the
image represent actions the robot can take.  - Red arrows move the arm farther away from
the camera, blue arrows move the arm closer towards the camera.  - Smaller circles are
further from the camera and thus move the arm farther, larger circles are closer and
thus move the arm backwards.  - The robot can only grasp or move objects if the robot
gripper is close to the object and the gripper fingers would stably enclose the object -
Your answer must end with a list of candidate arrows which represent the immediate next
action to take ( 0.3 seconds).  Do not consider future actions between the immediate
next step.  - If multiple arrows represent good immediate actions to take, return
all candidates ranked from worst to best.  - A general rule of thumb is to return 1-4
candidates.  Instruction:  Reason through the task first and at the end summarize the
correct action choice(s) with the format, "Arrow:  [<number>, <number>, etc.]." Task:
task
```

H.5. Manipulation offline eval prompt

Direct

```
Summary: The arrows are actions the robot can take. Red means move the arm forward
(away from the camera), blue means move the arm backwards (towards the camera). Smaller
circles are further from the camera and thus move the arm forward, larger circles are
closer and thus move the arm backwards. Do not output anything else, direct answer ith
the format, Arrow: [<number>, <number>, etc.]. IMG, Task: What are the best arrows
for the robot follow to pick white coat hanger?
```

CoT

```
Summary: The arrows are actions the robot can take. Reason through the task first and
at the end summarize the correct action choice(s) with the format, Arrow: [<number>,
<number>, etc.]. Description: The robot can only grasp or move objects if the gripper
is around the object and closed on the object. Red means move the arm forward (away
from the camera), blue means move the arm backwards (towards the camera). Smaller
circles are further from the camera and thus move the arm forward, larger circles are
closer and thus move the arm backwards. You must include this summarization. IMG,
Task: What are the best arrows for the robot follow to pick catnip toy?
```

Few-shot Direct

```
Summary: (same as above) IMG, Task: Erase the writing on the whiteboard. Arrow: [5,
10], IMG, Task: Pick up the iced coffee can. Arrow: [1], IMG, Task: Pick up the
string cheese. Arrow: [8, 15, 3, 13], IMG, Task: pick white coat hanger.
```

Few-shot CoT

```
Summary: (same as above) IMG, Task: Erase the writing on the whiteboard. The robot
is holding an eraser, so it should move it over the marker on the whiteboard. The
following arrows look promising: 5. This arrow moves the eraser over the writing and
away from the camera and thus towards the whiteboard. 10. This arrow too moves the
eraser over the writing and has an even smaller circle (and more red) and thus more
towards the whiteboard. Arrow: [5, 10], IMG, Task: ... Arrow: [5, 10], IMG, Task:
... Arrow: [8, 15, 3, 13], IMG, Task: pick oreo.
```