APPENDIX

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 $\left(\hat{I}, w_{1:M}\right) = \Omega(I, a_{1:M})$
  Query VLM $P_{\text{VLM}}\left(w \mid \hat{I}, \ell\right)$ 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

A. Additional Related Work

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 [28], [29] 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 [55], [58], [66], 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.

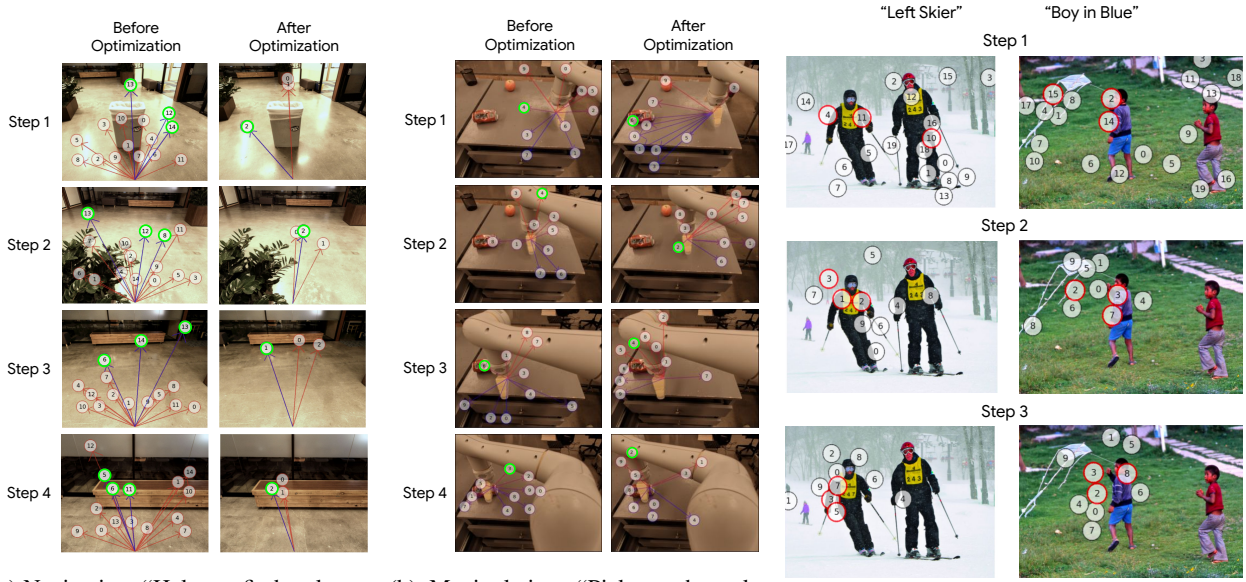
Foundation models for robot reasoning and control. In recent years, foundation models have shown impressive results in robotics from high-level reasoning to low-level control [13], [19]. Many early works investigated robotic reasoning and planning regimes where LLMs and language outputs are well suited [1], [8], [21], [23], [31], [32], [34], [41], [47], [51], [63]. 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 [7], [12], [14], [20], [43], [44] 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 [22], [35], [62]. 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], [15], [25], [38], [42], [45]. Unlike these works, we show how VLMs can be applied *zero-shot* to low-level control of multiple real robot platforms.

B. 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 F.

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 Appendix L. 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 7. More details on the results can be found in Appendix Section H.

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 J.



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

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

(c) RefCOCO spatial reasoning

Fig. 5: (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.

RAVENS [64]. 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. [64]. This action space allows us to evaluate higher-level action representations. More details on the results can be found in Appendix Section I.

C. 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 M and results are in Figure 8 and examples in Figure 5.

D. Experiments on Prompting.

Text prompts. To understand the effect of different text prompts, we experiment with several design choices, with numbers reported in Appendix H. 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.

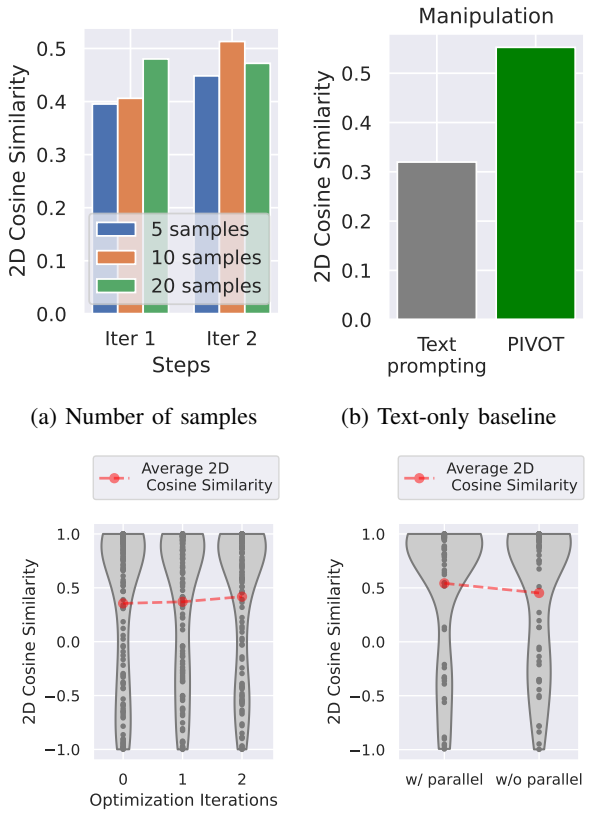
Visual prompts. Aspects of the style of visual prompts has been examined in prior works [46], [59], such as the color, size, shading, and shape. Herein, we investigate aspects central to PIVOT– the number of samples and the importance of the visual prompt itself. An ablation over the number of samples is shown in Figure 6 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.

E. 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 9). In particular, we compare PIVOT using four sizes of the Gemini family of models [16] 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 [16].

F. Mobile Manipulator Navigation Offline Evaluation

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



(a) Number of samples (b) Text-only baseline

(c) Iterations (d) Parallel calls

Fig. 6: Offline evaluation results for manipulation tasks with cosine similarity (higher is better).

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.

Evaluation Results. Table III 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.

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,

TABLE III: 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

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 IV. For GPT-4V, the text baseline incurs higher mean and standard deviation of errors across all tasks.

TABLE IV: Reasoning with Image Annotations vs. with Text for Navigation offline evaluations measured in L2 loss (lower the better).

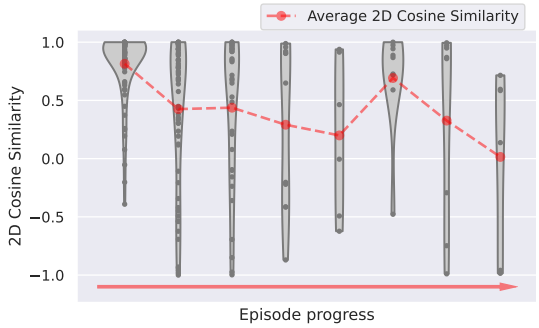
Method	In-View	Semantic	Out-of-View
Image	0.21 ± 0.002	0.23 ± 0.012	0.44 ± 0.04
Text	0.26 ± 0.15	0.35 ± 0.14	0.46 ± 0.31

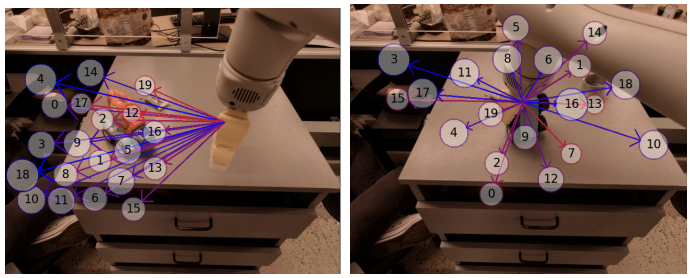
G. Mobile Manipulator Manipulation Online Evaluation

In addition to the quantitative evaluation trials for the real-world manipulation experts described in Section IV-B, 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.

H. Mobile Manipulator Manipulation Offline Evaluation

Using the offline mobile manipulator dataset described in Section B, 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





(a) Easy scenario (b) Hard scenario

Fig. 7: 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.

Fig. 8: 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.

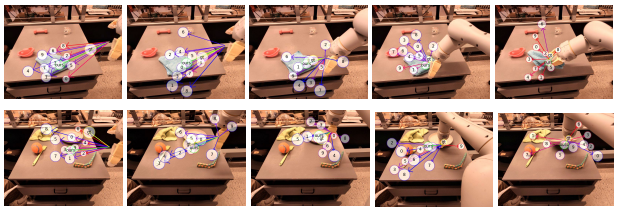


Fig. 12: Two episodes of mobile manipulator manipulation offline evaluation. It shows our method can generate reasonable actions following the arrow annotations.

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 M. 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 described in Fig. 7 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.

I. 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 M 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 often lacks the ability to precisely choose points that allow it to execute the task successfully in one action.

J. 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

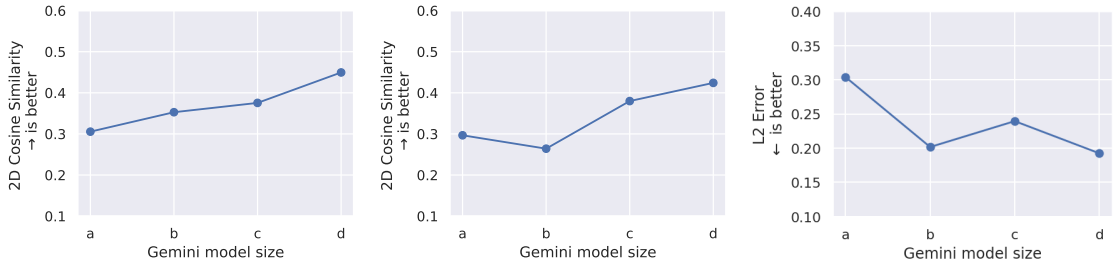


Fig. 9: Scaling results of first iteration visual prompting performance across Gemini model [16] sizes show that PIVOT scales well with improved VLMs. Left and center plots are manipulation (pick up objects, move one object next to another), right plot is navigation.

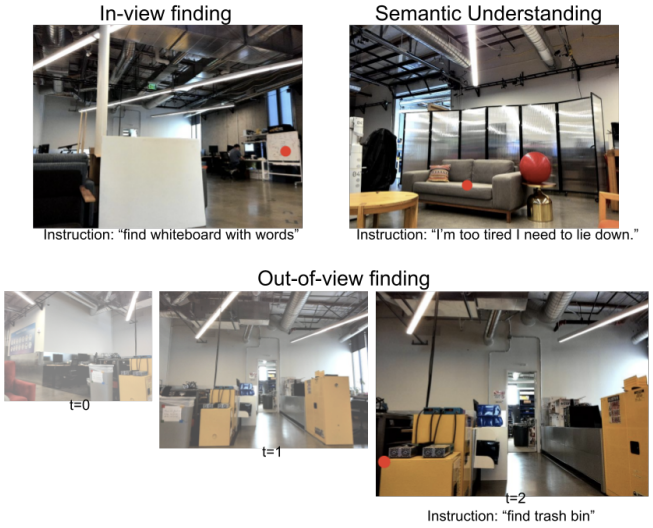


Fig. 10: Example tasks in the offline navigation dataset from different task categories. Red dot denotes the ground truth target.

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 V, 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.

K. 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).

L. 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 (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 determining when to

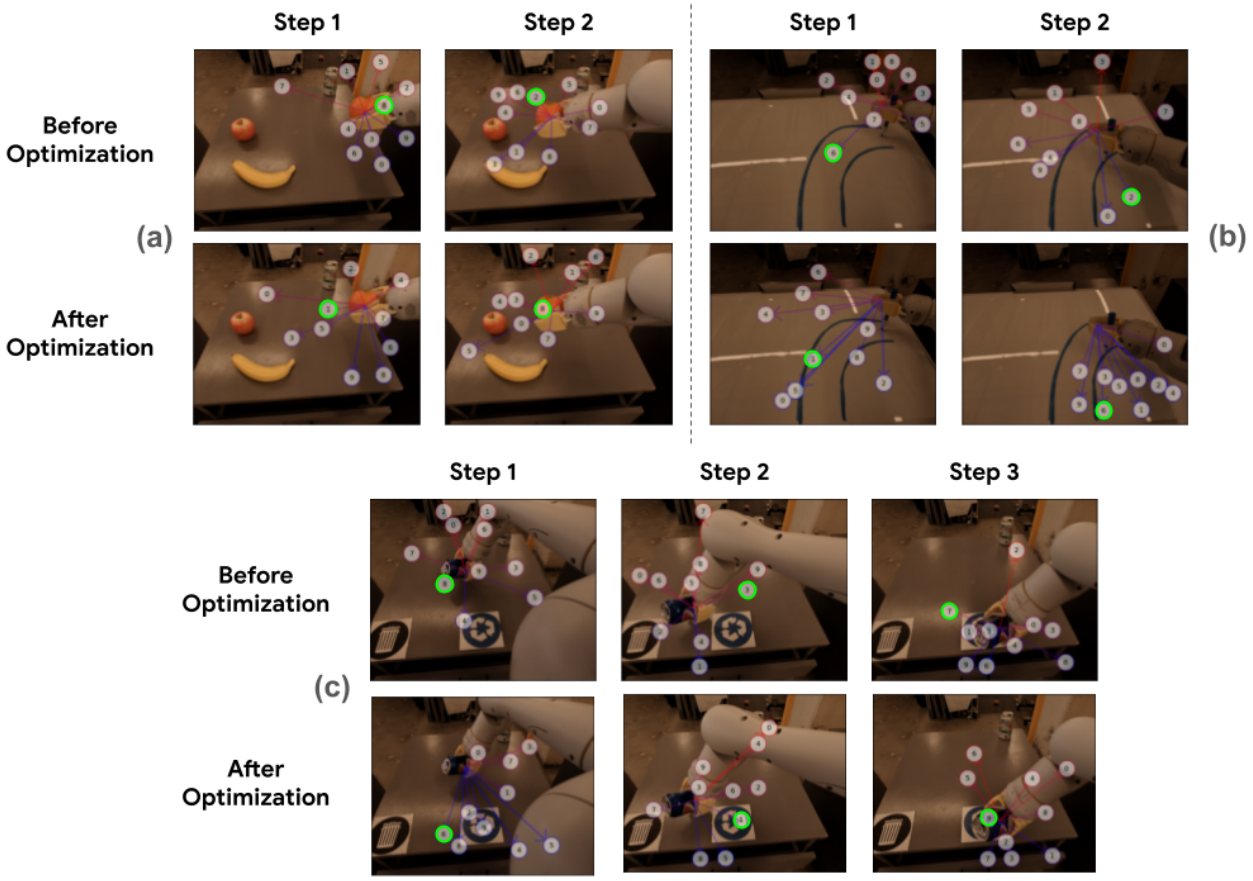


Fig. 11: Evaluating PIVOT on real world mobile manipulator tabletop manipulation scenarios which require a combination of semantic reasoning and action understanding. Using 3 optimization iterations on the real world mobile manipulator, we see promising successes for (a) "move the orange to complete 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".

TABLE V: 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

grasp, whether to lift an object, or approaching an object from the correct side to push. This is visualized in Figure 7, 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.

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

TABLE VI: 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%

TABLE VII: 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%

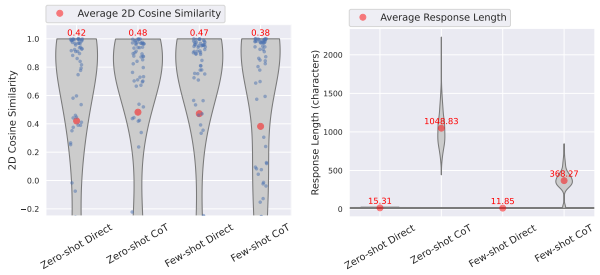


Fig. 13: Ablation of few-shot vs. zero-shot and CoT vs. direct performance on manipulation domain. The best performing combination is zero-shot CoT. However, direct models can achieve similar performance with much fewer output tokens thus more token efficient.

with a history of actions as input context to the VLM to guide future generated actions.

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.

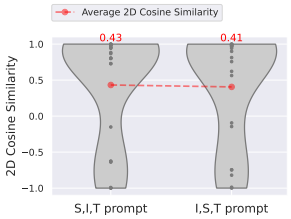


Fig. 14: Ablation of order of preamble, image, and task on mobile manipulation domain. We found it is beneficial to put the image closer to the end of the prompt, though the effect is marginal. P, I, T means preamble, followed by image and task description, and I, P, T means image followed by preamble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

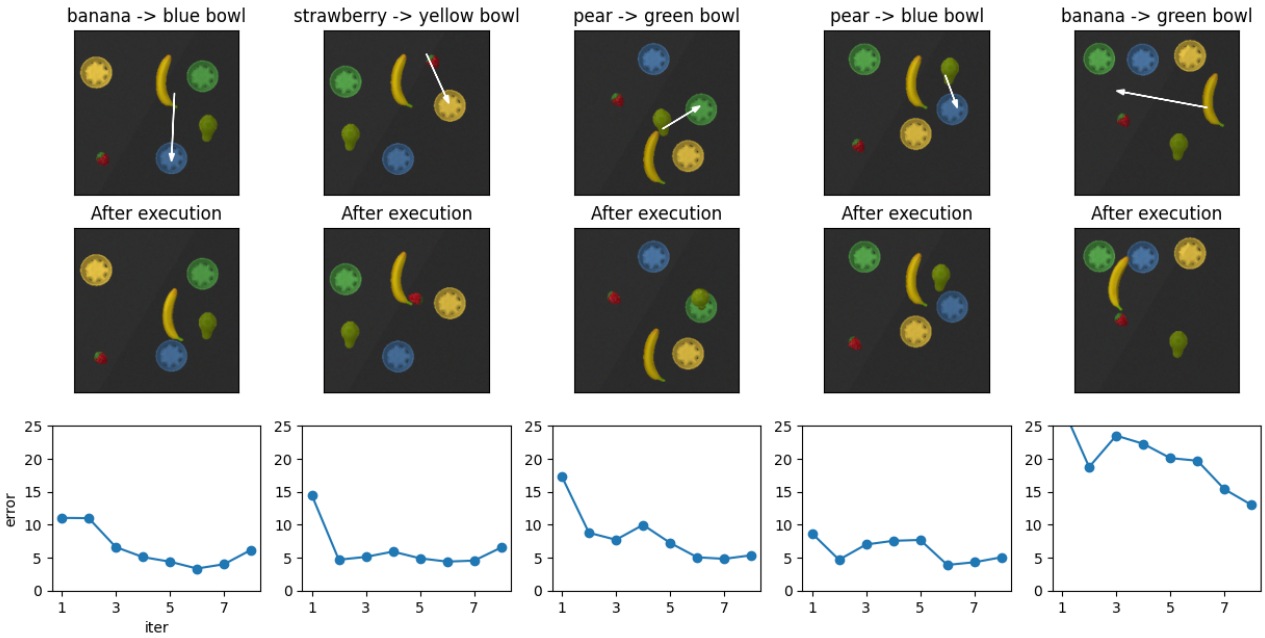


Fig. 15: RAVENS evaluations. Each column shows a different task instance. Title: pick object followed by place object. Top row: initial image with pick and place locations predicted by VLM indicated by white arrow. Middle row: result after executing action. Bottom row: L2 distance between predicted and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

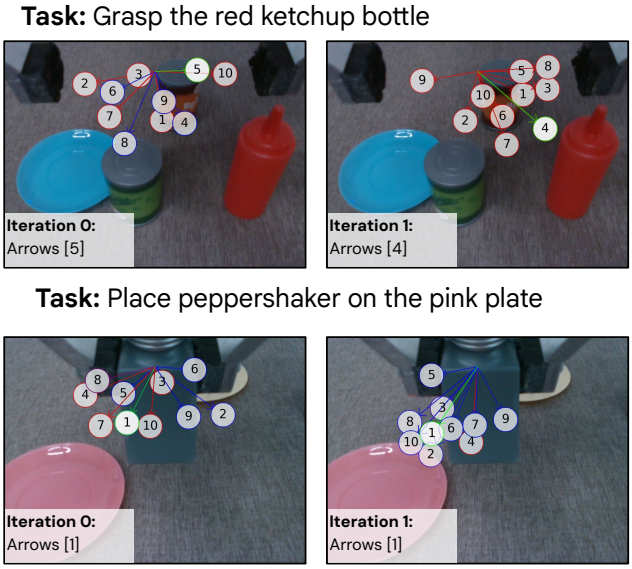


Fig. 16: Rollouts on the Franka environment.

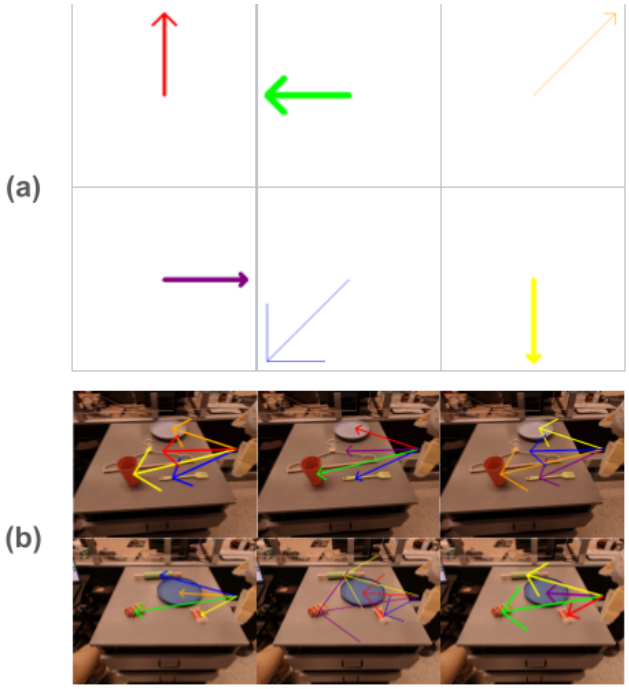


Fig. 17: Examples of procedurally generated datasets studying the robustness of VLMs for understanding visual annotation arrow styles. (a) focuses on absolute direction understanding of single arrows on blank backgrounds. (b) focuses on object-relative arrow understanding in realistic scenes.

M. Prompts

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": [] }
```

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": [] }
```

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.
```

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
```

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.
```