Beyond Words: Integrating Image-Based Context for Situation-Grounded Procedural Planning

Omkar Yadav
oyadav@umich.edu
Vatsal Joshi
jvatsal@umich.edu

1 Introduction

In recent years, language model (LM) based procedural planning techniques have considerably advanced (3). Generally, these methods have primarily operated in the text domain, focusing on optimizing efficiency, accuracy, and adaptability in goal-based planning (1) (4). However, many real-world applications, especially those involving embodied AI agents, require a richer, situational context that can only be achieved through visual cues.

Some interesting approaches have looked to improve small language model (SLM) based procedural planning for more practical real-time integration. One such framework is PlaSma, which uses symbolic knowledge distillation and verifier-guided beam search to generate step-by-step plans (2). We are particularly interested in this approach because the authors have shown that their beam search implementation has greatly improved the accuracy of procedural planning. However, unlike PlaSma we do not look to optimize SLMs - instead we look to apply the optimization techniques to LLMs to improve their accuracy on procedural planning.

This project proposes a novel extension of existing procedural planning techniques by incorporating an image modality. Specifically, we look to extend the work done by PlaSma by pairing each goal with a corresponding image that provides environmental or situational context. We believe this will help generate plans that are not only linguistically and logically coherent, but also context-aware.

Specifically we look to investigate the following research questions:

• **RQ1**: Does step-wise verifier-guided beam search improve procedural planning quality compared to full-plan generation?

• **RQ2**: How does the number of beams and candidates impact the quality of step-wise planning?

• **RQ3**: Does finetuning a Vision Language Model (VLM) on egocentric, chain-of-thought data improve planning compared to mere few-shot prompting?

2 Related Work and Motivation

Procedural planning has traditionally been framed as a task of decomposing high-level goals into ordered sequences of steps using language models (3). For example, PlaSma worked on improving the following:

- **Efficiency**: Distilling procedural knowledge from LLMs to SLMs.
- **Accuracy**: Improving plan quality with verifier-guided beam search.
- **Adaptability & Generalizability**: Introducing constrained and counterfactual planning to handle dynamic scenarios.

Despite these contributions, this work makes the assumption that their text based procedural planning was robust to generalize to different domains. Many tasks, such as embodied AI, involve spatial reasoning requiring environmental context that text alone cannot efficiently capture. Our proposal

seeks to bridge this gap by incorporating visual context to the training and inference stages of procedural planning. We hypothesize that this integration will enhance the agent's contextual understanding of the terrain as real-world landmarks and objects will be encoded. Ultimately, this will enable agents to be applied to downstream tasks such as robotics, navigation, and augmented reality.

2.1 Background: The PlaSma Approach

In the original PlaSma paper, the data generation process follows a three step process (2). First, the authors create a pool of goals that is generated using a large teacher model. Then, goals are paired with plans via few-shot prompting. Goal and plan pairs are then evaluated on several factors and filtered out based on these criteria. Ultimately, over 100,000 quality goal and plan pairs are created. Note that the authors of this paper also look at two variants: constrained planning and counterfactual replanning which look to incorporate constraints at different points of the goal generation. Once the dataset has been created, procedural knowledge distillation is performed through different objectives allowing the SLMs to better mimic the outputs of the LLMs.

Although the training process ensures that the SLMs are optimized for the procedural planning task, there are still some problems that occur during inference time. Namely, the SLM often makes temporally or causally related mistakes. To mitigate this, the authors employ a verifier-guided beam search mechanism to ensure the plans are more robust to these errors. This process works by producing multiple candidate steps that are evaluated by another model (the verifier). The verifier grades each of the potential steps based on correctness, consistency, and completeness. A function takes the grade from the verifier and the score from the original model and outputs a linear combination that determines the final score. The authors articulate that this beam search based inference process greatly improves the accuracy and robustness of the generated plans.

2.2 Related Work: Multimodal Procedural Planning via Dual Text-Image Prompting

We also look at another paper that introduces an agent called EmbodiedGPT (**?**). The authors of this paper are looking to enable embodied agents (robots) to autonomously accomplish long-horizon tasks in a physical environment. These types of tasks inherently require high levels of perception, reasoning, and action capabilities. Thus, the authors leverage chain-of-thought (COT) reasoning as recent work has shown that it improves LLMs' capabilities in challenging tasks. However, this task is clearly different as we have multi-modal inputs - thus introducing a new set of challenges.

The authors aim to answer the following three critical questions:

- **Domain Gap Adaptation:** How can we apply LLMs to robotics that may face large domain gaps?
- **Structured Planning through CoT:** How can we exploit LLM COT reasoning for structured planning?
- **Plan-to-Action Transformation:** How can we transform plans for downstream manipulation tasks?

Ultimately, the authors provide us with the following contributions that helped motivate our work. First, the EmbodiedGPT Model, which has multi-modal integration through visual/language understanding and executable language planning. Second, we have the EgoCOT Dataset that contains large-scale egocentric videos and COT text plans. For our project, the dataset is more useful because we are able to convert these video and text plans into an image-based planning task as described in Section 3 . It is important to note that our system is different from EmbodiedGPT as we have no interaction with the environment. In other words, we do not have a closed loop system - we are optimizing an open loop task.

3 Methodology

3.1 Architecture Overview

In this section, we will provide a brief overview of the architecture we will be implementing for the rest of the paper. The high level architecture is shown in Figure 1.

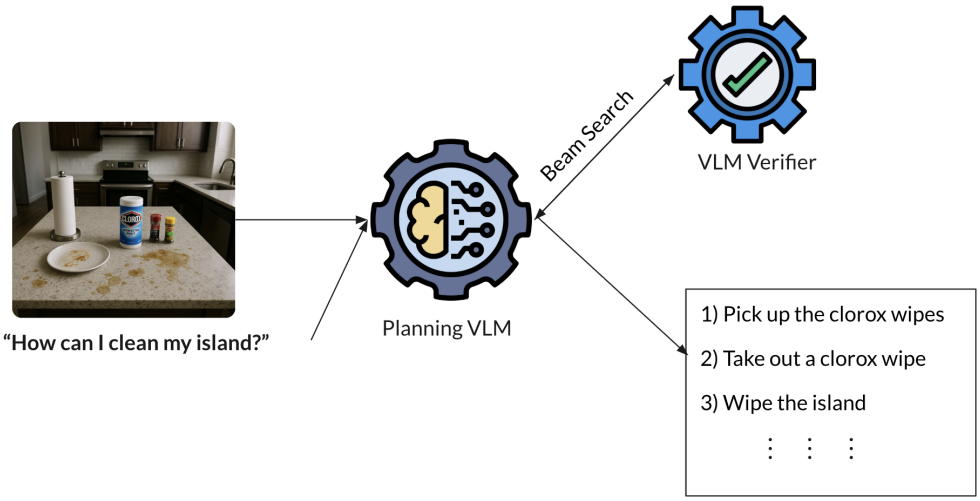

Figure 1: High level architecture overview

Concretely, we have two VLMs: one for generating the plans and the other to perform the step-wise beam search. As mentioned in Section 2, we extend the verifier-guided beam search proposed in the PlaSma architecture. Our Planning VLM takes in the original goal, image, and the plan so far and outputs one step. This step is then evaluated by the Verifier VLM which takes in the original goal, image, and the candidate step and provides a continuous score that can be interpreted as the quality of the step. It is important to note that this is different from the original verifier-guided beam search introduced in PlaSma. Namely, we omit the score coming from the planning model that is shown in Figure 2 and solely rely on the verifier. Additionally, we have an image that is fed into both the verifier and planning VLMs.

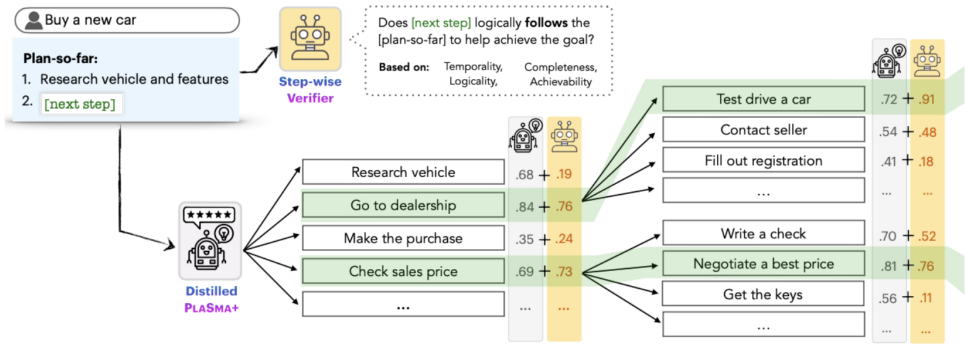

Figure 2: PlaSma Beam Search (2)

Although our pipeline is model agnostic, we needed to pick an architecture for our project. To determine our model of choice, we tested many different VLMs on multi-modal reasoning tasks. We found that that Qwen-VLM displayed strong reasoning / planning capabilities and was open source which was crucial for our finetuning. Specifically, we used Qwen/Qwen2.5-VL-3B-Instruct for size constraints.

3.2 Dataset Processing

To implement our proposed architecture, we require data to be formatted in a structured manner. Specifically, we require entries to have following attributes:

- **Original Goal:** Text entry that describes the high level goal
- **Initial State:** An image that describes the initial state
- **History:** A list of text entries that that has all previous steps. The length of the list can vary from [0, N] where N is the number of steps in the plan.
- **Target Step:** Text entry for the target step we want the model to output.

To construct this dataset, we transform the EgoCOT dataset. The original dataset is shown in Figure 3. Each entry contains a video: a frame of images, a caption: the high level goal, a plan: a list of steps, and a similarity score that we ignore.

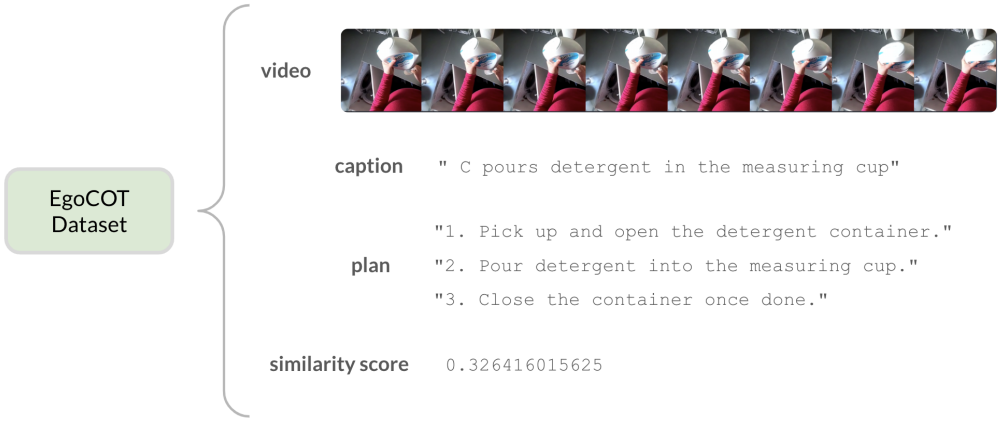

Figure 3: EgoCOT Dataset

First, we convert each of these plans into a list of steps. Then, for each subsequence of steps, we create a new data point for each possible next step prediction. So in the example shown in Figure 3 we create five unique data points - each with a different number of steps in the history. More formally, we have

$$\forall i \in [1, N+1], \quad \text{predict } P(s_i \mid s_0, s_1, \ldots, s_{i-1}),$$

Next, we add prefixes to each field that denote their use. Each candidate step starts with [STEP] and each goal starts with [GOAL]. Finally, we only take the first frame from the video and perform denormalization to make the image load more clearly.

3.3 Planning VLM

The training for the Planning VLM was quite straightforward as we are simply performing next step prediction based on our dataset. We utilize parameter efficient finetuning (PEFT) with low rank adaption adapters (LoRAs) to balance efficiency and performance. Our LoRA configuration had a rank=8, alpha=16 and a dropout=0.05. The finetuning was performed on 80% of the data with convergence after 3 epochs. The training loss can be seen in Figure 4, which confirms that the loss has converged.

Figure 4: Planning VLM Training Loss

3.4 Verifier VLM

As noted previously, the Verifier VLM must output a continuous score that describes the quality of the candidate step. However, since all of the plans in the EgoCOT dataset are valid, we only have positive examples to train our model. This is problematic as the verifier must have some understanding of what negative samples look like. Our solution was to use an LLM, GPT4o, to perturb the positive examples with common mistakes that we often observe in planning tasks. We accomplish this through few-shot prompting and end up with a more uniform dataset with respect to the output score.

In the end, we have 2,000 samples with an equal split of positive and negative examples to train our Verifier VLM. Again, we perform PeFT using LoRA and extract activations from the last layer. These activations are then fed into a linear layer for the regression task. Our LoRA configuration had a rank=8, alpha=32 and a dropout=0.1. The linear layer uses the same hidden size as the Qwen 2.5B VLM of 8,192. The regression setup is trained on 80% of the data until convergence which occurs after 3 epochs.

In the evaluation for our final presentation, we noticed that even with a low quantitative loss (MSE loss < 0.05), the VLM verifier performed noticeably poorly. We found many examples, where the verifier scored invalid steps high and valid steps low. Essentially, it seemed like the verifier did not have strong out of distribution generalization and was relying too much on the training examples we generated. Further, since our data was synthetically generated, we were unable to create a robust distribution over the quality of steps. The quality of the verifier is crucial to the success of the pipeline as we are omitting the score from the original model that exists in Figure 2. In other words, the verifier is the only model preserving the quality of the plans. Once we determined this issue, we decided to look into using state of the art proprietary VLMs. Specifically, we used GPT4 Turbo as it had strong reasoning capabilities. Through few-shot prompting we were able to qualitatively observe that this model had much better reasoning and generalization capabilities. Thus, we pivoted and leveraged the GPT4 Turbo API to perform the verifier scoring.

4 Evaluation

4.1 Setup and Evaluation Goals

To reiterate, the central objective of this project is to incorporate vision as a modality into procedural planning, specifically targeting embodied tasks which significantly benefit from environmental awareness. The purpose of our evaluation is to validate our proposed vision-language procedural planning framework by examining whether integrating visual context, employing step-wise verifier-guided beam search, and finetuning on an egocentric chain-of-thought dataset effectively enhance planning quality. Our evaluation is structured to address the three research questions introduced in Section 1.

All of our evaluations were conducted using a fixed dataset of 50 examples, randomly sampled and preprocessed (see Section 3.2) from the EgoCOT dataset, consisting of egocentric image-goal pairs. Although 50 examples might appear limited compared to typical LLM evaluations, this constraint was practical, given the high cost of GPT API calls required for verifier and judgment steps, as well as the computational and time limitations inherent in conducting numerous experimental runs. To ensure consistency across models and experimental settings, all planning prompts were constrained to produce a maximum of exactly five steps, with [EOP] tokens marking the end of a plan.

In our evaluation, we focus on pairwise comparison between two competing planning strategies. For each pair, we generate a plan from method A and method B, and then use an LLM (GPT-4-turbo) as a judge to decide which plan is superior based on criteria such as conciseness, minimality, and theoretical actionability in embodied agents. GPT-4-turbo is prompted to then return a score of 0 (Plan A better), 1 (Plan B better), or -1 (tie).

4.2 Large Language Models as Judges

The use of LLMs as judges is not a novel concept; recent work (6) has demonstrated that LLMs can achieve over 80% agreement rates with human experts on complex evaluation tasks. Specifically, GPT-4 models have been shown to match or even surpass human annotators in terms of consistency and judgment quality when evaluating chatbot outputs. This suggests that using an LLM-as-a-judge is a reliable, scalable, and cost-effective alternative to human annotation.

Given the fact that our evaluation criteria is relatively objective and that our planning outputs are structured, we believe GPT-4 is a suitable proxy for human evaluation. While PlaSma employed human evaluations via MTurk to assess procedural planning performance, we argue that LLM-as-a-judge offers comparable effectiveness with significantly reduced resource and time requirements.

4.3 RQ #1: Does Step-Wise Verifier-Guided Beam Search Improve Procedural Planning Quality Compared to Full-Plan Generation?

4.3.1 Motivation

Prior work, such as PlaSma, has shown that stepwise verifier-guided beam search dramatically improves planning quality in text-only environments. However, it's not certain whether we can see similar improvements in multimodal settings where plans must be grounded in visual observations. Beam search offers the potential advantage of refining each step based on image-grounded verifier feedback, but it also introduces pacing challenges: without visibility into the entire plan upfront, the model may struggle to balance immediate action optimality against global plan coherence.

4.3.2 Experimental Design

To investigate whether step-wise verifier-guided beam search improves procedural planning quality compared to full-plan generation (where an entire plan is decoded with one LLM pass), we designed a three-part experimental evaluation:

Experiment 1: Finetuned Qwen (1 beam, 1 candidate) vs. Full-Plan Base Qwen (few-shot prompted)

In our first experiment, we wish to establish a baseline comparison between basic step-wise planning and traditional full-plan generation. Specifically, we use the Qwen 2.5 VL-3B Planning model

fine-tuned on EgoCOT for next-step prediction. We configured the step-wise planner with only 1 beam and 1 candidate per step, effectively resulting in a sort of greedy decoding where the model produces the highest-scoring single candidate at each step without searching over any alternatives.

We compared this step-wise plan to a full-plan generated by a Base Qwen model (unfinetuned) using few-shot prompting. In this case, the model will reason about the full plan in one pass, with access to goal context but no verifier feedback at intermediate steps.

This experiment was designed to determine whether generating the entire plan in a single pass is better than incrementally constructing the plan step-by-step.

Experiment 2: Finetuned Qwen (2 beams, 3 candidates) vs. Full-Plan Base Qwen (few-shot prompted)

We already recognized that greedy step-wise planning (1 beam, 1 candidate) was unlikely to succeed against full-plan generation, which is why in Experiment 2 we introduced true search into the step-wise planner. We increased the search space by setting 2 beams and sampling 3 candidates at each expansion step, with selection guided by the verifier model.

The full-plan baseline remained the same: Base Qwen with few-shot full-plan prompting.

This experiment tests whether beam search can exploit the benefits of the verifier while mitigating the inherent short-sightedness of step-wise planning, allowing it to better compete with full-sequence planning.

Experiment 3: GPT-4 (2 beams, 3 candidates, few-shot prompted) vs. GPT-4 (Full-Plan few-shot prompted)

Finally, in Experiment 3, we aimed to control for model quality by repeating the evaluation using GPT-4-turbo via API calls instead of Qwen-3B. Since GPT-4-turbo possesses much stronger reasoning, planning, and language generation capabilities than smaller open-source VLMs, this comparison isolates the effect of model capacity.

Both strategies (step-wise beam search and full-plan generation) used few-shot prompting but differed in decoding style. In the beam search setting, GPT-4-turbo generated next-step candidates conditioned on prior steps and goal, with verifier scores guiding beam updates. In the full-plan setting, GPT-4-turbo produced the entire 5-step plan in one forward pass given the goal and examples.

This final experiment allows us to answer two questions simultaneously. Does beam search remain valuable even when applied to very strong base models like GPT-4? Are the deficiencies observed in earlier experiments simply due to model scale (e.g., Qwen's smaller size) or fundamental to the step-wise beam search design?

4.3.3 Results and Analysis

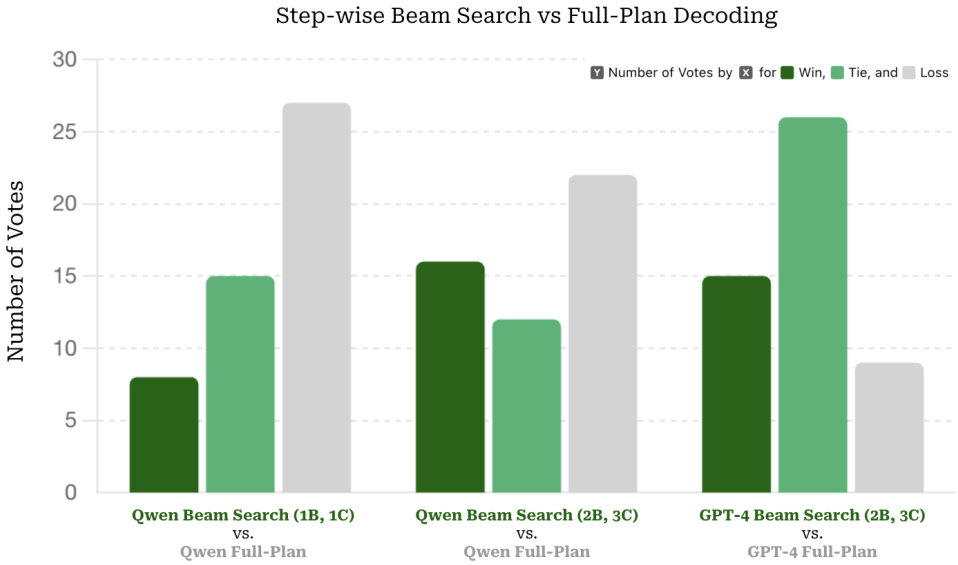

Figure 5: Comparison of step-wise beam search versus full-plan decoding across the three experimental settings. From left to right: **Experiment 1 -** Finetuned Qwen (k=1 beam, c=1 candidate) vs. Full-Plan Base Qwen (few-shot prompted), **Experiment 2 -** Finetuned Qwen (k=2 beams, c=3 candidates) vs. Full-Plan Base Qwen (few-shot prompted), **Experiment 3 -** GPT-4 (k=2 beams, c=3 candidates) vs. Full-Plan GPT-4 (few-shot prompted)

We present the results of all experiments in 5. When comparing simple step-wise planning (1 beam, 1 candidate) against full-plan generation by a few-shot Base Qwen model, full-plan generation decisively outperformed. Specifically, Full-Plan Base Qwen was preferred in 27 cases, compared to only 8 wins for Finetuned Qwen step-wise planning, with 15 ties. This outcome was expected since without any meaningful search, 1-beam, 1-candidate decoding lacks the ability to explore alternative actions or reason about the global plan structure.

However, when experimenting with richer beam search (2 beams, 3 candidates), we observed a substantial improvement. In this setting, Finetuned Qwen step-wise planning won 16 times, reducing Full-Plan Base Qwen's margin to 22 wins and increasing ties to 12. Now, the system is equipped to better evaluate candidate steps, correct early missteps, and incrementally assemble stronger plans. With that being said, full-plan generation still has a big advantage over beam search, which means the issues of limited global awareness still cause our beam search to fall behind.

To isolate whether this limitation is an issue specific to small models like Qwen, we repeated the experiment using GPT-4. Interestingly, when using GPT-4 with verifier-guided step-wise beam search, we saw 15 wins for beam search compared to 9 wins for full-plan generation, with 26 ties. This proves that with a sufficiently strong base model, beam search has the potential to outperform full-plan generation, surprisingly without any finetuning whatsoever.

From this set of experiments, it is clear that verifier-guided beam search is a viable strategy for vision-language procedural planning, but its effectiveness depends on the underlying model's planning and reasoning capacity. It is also clear that full-plan generation benefits from better pacing and global task structure awareness, especially for smaller models like Qwen.

4.4 RQ #2: How Does the Number of Beams and Candidates Impact the Quality of Step-Wise Planning?

4.4.1 Motivation

Two key hyperparameters in the verifier-guided step-wise beam search planning is the number of beams and the number of candidates sampled at each expansion step. Increasing beam width and candidate count should, in principle, improve plan robustness by better exploring possible continuations and filtering out suboptimal early terminations.

4.4.2 Experimental Design

To investigate this and determine a strong hyperparameter setup, we designed two experiments in a similar fashion.

Experiment 1: Finetuned Qwen (1 beam, 1 candidate) vs. Finetuned Qwen (1 beam, 3 candidate) We compared a step-wise beam search using k=1 beam and c=1 candidate against a beam search using k=1 beam and c=3 candidates. This evaluates in isolation the effect of increasing candidate diversity while keeping the number of beams fixed.

Experiment 2: Finetuned Qwen (1 beam, 3 candidate) vs. Finetuned Qwen (2 beam, 3 candidate)

We compared k=1 beam, c=3 candidates against k=2 beams, c=3 candidates, isolating the effect of introducing multiple beams for broader search, while holding the number of candidates constant.

4.4.3 Results and Analysis

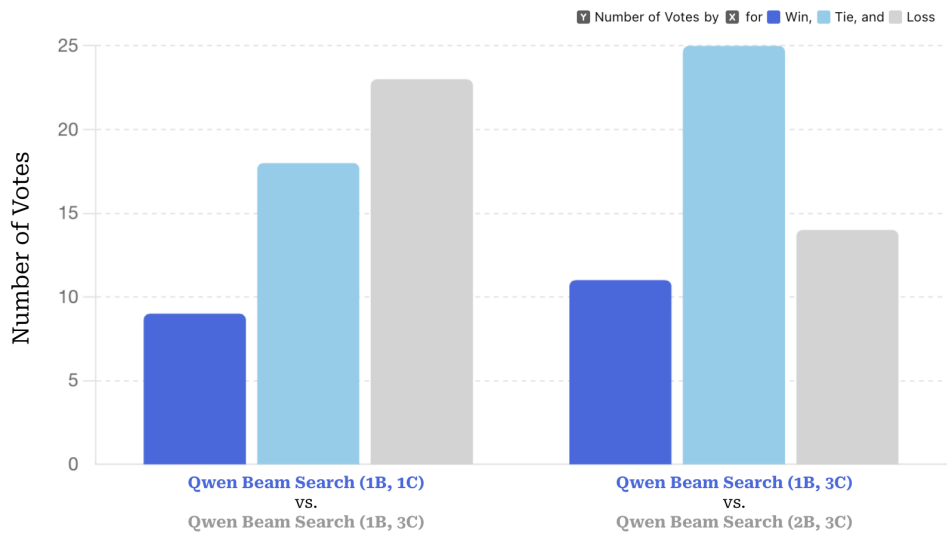

Figure 6: Comparison of varying numbers of beams and candidates. From left to right: **Experiment 1 -** Finetuned Qwen (k=1 beam, c=1 candidate) vs. Finetuned Qwen (k=1 beam, c=3 candidate), **Experiment 2 -** Finetuned Qwen (k=1 beams, c=3 candidates) vs. Finetuned Qwen (k=2 beam, c=3 candidate)

We present the results of these two experiments in Figure 6. When comparing Finetuned Qwen with 1 beam and 1 candidate against the same model using 1 beam and 3 candidates, we found a significant improvement. This aligns with our expectations that by sampling multiple candidates at each step, the planner avoids committing early to poor actions and has a greater chance to identify well-written and environment-grounded actions, as confirmed by the verifier.

However, we noticed that increasing the beam width from 1 to 2 (while keeping 3 candidates per expansion) yielded slightly more modest gains. One possible explanation for this is that while candidates at each step offer varied local actions, maintaining multiple beam paths across multiple

steps introduces competition among incomplete plans that may not diverge meaningfully unless further search depth or exploration strategies are added.

4.5 RQ #3: Does finetuning a VLM on egocentric, chain-of-thought data improve planning compared to mere few-shot prompting?

4.5.1 Motivation

To address our final research question, we worked on evaluating whether finetuning a VLM on the EgoCOT dataset genuinely improves stepwise planning compared to simply using few-shot prompting with a base model. Our rationale was that finetuning would enable VLMs to better adapt to egocentric tasks common in embodied AI scenarios and specifically enhance their proficiency in next-step prediction on our test set. On top of this, finetuning would be very useful in reinforcing adherence to the core principles of conciseness, minimality, and actionability.

4.5.2 Experimental Design

Experiment 1: Finetuned Qwen (2 beams, 3 candidates) vs. Base Qwen (2 beams, 3 candidates, few-shot prompted)

This experiment compares Qwen finetuned on EgoCOT against base Qwen that is taught to do next-step prediction with few-shot examples. Both setups have the same search strategy (k=2 beams, c=3 candidates) to isolate the impact of finetuning alone. We hope that these results would reveal whether exposure to egocentric, multimodal step-by-step reasoning data improves procedural planning quality beyond what can be achieved through prompting alone.

4.5.3 Results and Analysis

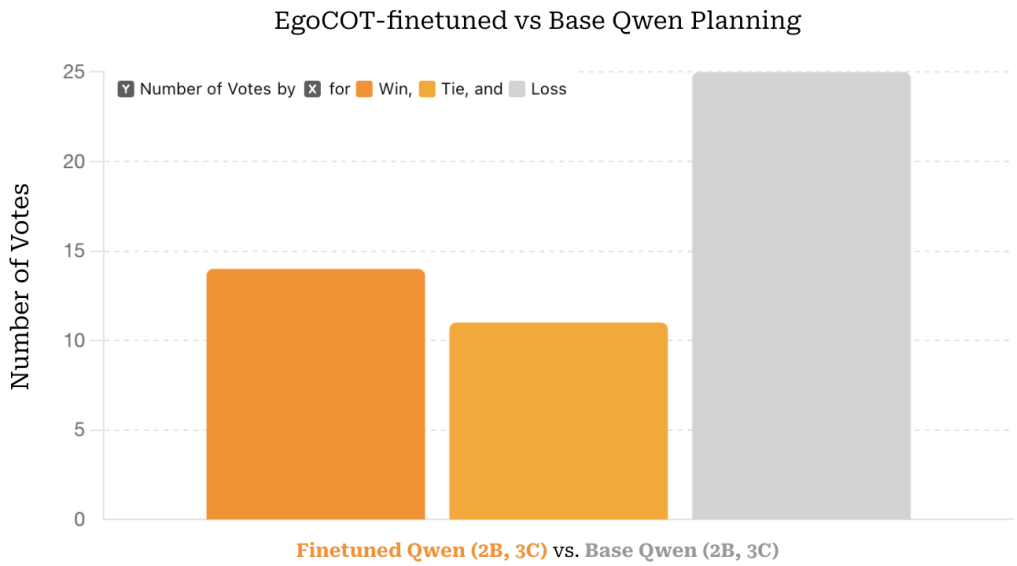

Figure 7: Comparison of EgoCOT-finetuned Qwen and Base Qwen performance in planning. **Experiment 1 -** Finetuned Qwen (k=2 beam, c=3 candidate) vs. Base Qwen (k=2 beam, c=3 candidate)

We present the results of this experiment in Figure 7. The results we observed were somewhat surprising. It essentially suggests that finetuning on EgoCOT does not yield consistent gains over simple crafted prompting for next-step generation.

We think one of the main factors for this was catastrophic forgetting where finetuning on a narrow dataset like EgoCOT may cause the model to lose some general reasoning capabilities, especially for planning tasks that do not perfectly matching the training examples.

Another important factor is that only about 15% of the EgoCOT examples contain an explicit [EOP] token for signaling plan termination. Because of this, when Qwen is finetuned on this data, it struggles with correctly pacing to 5 steps.

On inspection, we also saw that plans generated by finetuned Qwen are overly minimal. This is because EgoCOT plans are extremely terse and action-oriented, which while desirable for robotics, may penalize plans that require nuanced sequencing or elaboration.

5 Discussion

5.1 Step-wise Beam Search Effectiveness

From our experiments, we can extrapolate that Qwen2.5-3B may not have sufficient reasoning capabilities for our Planning VLM, which causes it to struggle to reason holistically about pacing, dependencies, and long-term task structure in a beam search pipeline. Although this Qwen model is known for its performance compared to other open source models, we were confined to the smaller version of this model due to resource constraints. We believe that the larger versions of this model (32B or 72B) may perform better because there seems to be a steep increase in performance at 32B parameters (5). Outside of the Qwen family, we also could have looked at state-of-the-art proprietary VLMs in the few-shot setting. In fact, Experiment 3 for RQ1 looks at evaluating GPT-4 in different generations processes. Although we do not gain any insights regarding the qualitative assessment of GPT from this experiment, manual observations proved that GPT had much more robust plans in both generation types. Thus, we hypothesize that these proprietary models could improve the overall quality of plans. However, it is important to consider that these models introduce monetary and time overhead since they cannot be run locally. In production, it is unreasonable to run GPT as the API call itself takes a few seconds and the cost is high.

However, we still think there are some promising solutions to improve performance even in smaller VLMs. Future prompting strategies could explicitly remind models of plan structure, for example by segmenting tasks into named phases (e.g., "Phase 1: Gather tools; Phase 2: Perform cleaning") to help maintain broader situational awareness across steps. Also, rather than starting step-wise decoding immediately, the model could first generate a rough full-plan outline (e.g., 5 short headlines) and then fill in each step via beam search with verification. Finally, another interesting idea could be to have the verifier model predict not just step likelihood, but also an expected number of steps remaining, helping guide when [EOP] should appear.

5.2 Number of candidates and beams

Our experiments showed that expanding the number of candidates per step significantly improves plan quality compared to increasing the number of beams. This means that local candidate exploration is crucial for eliminating poor intermediate actions and building more grounded plans. We think some interesting future directions could prioritize high candidate counts at early steps, when plans are most unconstrained, and then later switch over to less candidates to boost computational efficiency. Also an interesting branch and bound technique where beams are aggressively pruned based on early verifier scores, could help prevent wasted search on doomed paths.

5.3 EgoCOT dataset selection

The surprising finding through our experiments was that EgoCOT-finetuned Qwen did not outperform few-shot base Qwen. This shows us that there are deeper issues with the finetuning process and the dataset itself. While EgoCOT was the best dataset we could find for our use case, an exploration into datasets specifically used in embodied fields might be fruitful. Future datasets should emphasize correct plan terminations more explicitly or even use synthetic augmentation to generate more termination examples since they were so scarce in ours. Also, overly minimal step phrasing in EgoCOT led to under-detailed plans. A solution here could be Post-Hoc Elaboration, where after generating steps, an LLM could expand each step into a slightly more descriptive instruction.

5.4 LLMs as judges

While LLMs as judges are a scalable and practical judging framework, we must also discuss their limitations and biases in our use case.

In our qualitative analyses, we saw that GPT-4 tends to overvalue linguistic fluency and grammatical perfection over raw actionability or grounding, which is more useful in embodied control. It hesitates to penalize omissions of environment-specific actions (like ignoring an object that should have been manipulated based on the image). Also, since it's tuned for conversation, GPT-4 prioritizes human-preferred phrasing like "ensure the counter is clean" over arguably more robot-compatible explicit commands like "[STEP] wipe counter with Clorox." The latter is what is found more prominently in the EgoCOT dataset.

In the future, a possible solution could be to have task-specific finetuned judges where we would collect a set of expert-ranked plans and finetune a smaller LLM to mimic expert preferences. Another solution could be to include more objective metric-based scoring alongside LLM judgment that would measure things like object coverage (% of visible objects in the image referenced in plan steps) or step conciseness.

6 Conclusion

In this work, we investigate a novel planning task where we consider images as context for developing high level plans. Given the unexplored nature of this task, we create a multi-modal dataset that consists of images, goals, and plans transformed from the EgoCOT dataset. Additionally, we propose a framework for performing image based planning by extending the PlaSma inference method with this modality. We find that finetuning VLMs to a next step prediction setup may introduce catastrophic forgetting as discussed in Section 4. However, our other evaluations show that some combinations of beam search parameters can be useful in creating higher quality plans in this image-based setup.

There are some limitations / future work that we must consider when evaluating this work. First, we were bound to resource constraints on GPUs and by cost. This made it difficult to use more powerful models both locally and through API calls. Further, these constraints made it difficult to use larger datasets both in the training and evaluation steps. This could be an explanation as to why the verifier and planning VLMs performed so poorly as there was not enough data for training. Additionally, having only 50 samples for the evaluation makes it difficult to determine if the results we are seeing is merely from the variance in the data. However, running any more than 50 samples would have been difficult as each run required 1 dollar which is just not scalable.

Finally, we would like to have tried our approach in a closed-loop system. This would have enabled the model to understand if the plan it was generating could be revised to improve quality. The PlaSma paper looked at a counterfactual replanning setup to investigate this type of phenomenon, which would have been interesting. Another closed-loop setup we could have looked at is the embodied case similar to EmbodiedGPT. In this case, the model interacts with the environment getting direct feedback from each of its steps. Again, this enables the model to revise its plan as it generates steps. However, these setups have many more moving parts introducing overheads in computation and time.

7 Work Division

Omkar:

- Helped with scripts to preprocess the raw data
- Created verifier data using few-shot prompting
- Trained and evaluated different iterations of the verifier VLM
- Helped with final presentation and paper

Vatsal:

- Helped with scripts to preprocess the raw data
- Trained and evaluated different iterations of the planning VLM

- Created evaluation pipeline for LLMs as a judge
- Helped with final presentation and paper

References

[1] Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Ho, D., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jang, E., Ruano, R.J., Jeffrey, K., Jesmonth, S., Joshi, N.J., Julian, R., Kalashnikov, D., Kuang, Y., Lee, K.H., Levine, S., Lu, Y., Luu, L., Parada, C., Pastor, P., Quiambao, J., Rao, K., Rettinghouse, J., Reyes, D., Sermanet, P., Sievers, N., Tan, C., Toshev, A., Vanhoucke, V., Xia, F., Xiao, T., Xu, P., Xu, S., Yan, M., & Zeng, A. (2022). Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. arXiv Preprint. Retrieved from https://doi.org/10.48550/arXiv.2204.01691.

[2] Brahman, F., Bhagavatula, C., Pyatkin, V., Hwang, J.D., Li, X.L., Arai, H.J., Sanyal, S., Sakaguchi, K., Ren, X., & Choi, Y. (2024). PlaSma: Making Small Language Models Better Procedural Knowledge Models for (Counterfactual) Planning. ICLR 2024. Retrieved from https://doi.org/10.48550/arXiv.2305.19472.

[3] Chen, W., Koenig, S., & Dilkina, B. (2024). RePrompt: Planning by Automatic Prompt Engineering for Large Language Models Agents. arXiv Preprint. Retrieved from https://doi.org/10.48550/arXiv.2406.11132.

[4] Lu, Y., Lu, P., Chen, Z., Zhu, W., Wang, X.E., & Wang, W.Y. (2023). Multimodal Procedural Planning via Dual Text-Image Prompting. ICLR 2024 Conference Withdrawn Submission. Retrieved from arXiv (https://arxiv.org/abs/2305.01795).

[5] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL Technical Report. Retrieved from arXiv (https://arxiv.org/pdf/2502.13923).

[6] Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., & Stoica, I. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. Retrieved from arXiv (https://arxiv.org/pdf/2306.05685).

[7] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo, "EmbodiedGPT: Vision-Language Pre-Training via Embodied Chain of Thought," Retrieved from arXiv (https://arxiv.org/pdf/2305.15021).

8 Appendix - Reproducibility Checklist

All files relating to code and data can be found in this github link: https://github.com/omkar-yadav-12/beyond-words

- **Conceptual/Pseudocode Description**: Includes a conceptual outline and/or pseudocode description of AI methods introduced **Yes**
- **Clarity of Statements**: Clearly delineates statements that are opinions, hypotheses, and speculation from objective facts and results **Yes**
- **Pedagogical References**: Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper **Yes**

8.1 Theoretical Contributions

- Does this paper make theoretical contributions? **No**

8.2 Datasets

- Does this paper rely on one or more datasets? **Yes**
  - A motivation is given for why the experiments are conducted on the selected datasets **Yes**
  - All novel datasets introduced in this paper are included in a data appendix **Yes**
  - All novel datasets introduced in this paper will be made publicly available upon publication with a license allowing free research use **Yes**
  - All datasets drawn from existing literature are accompanied by appropriate citations **Yes**
  - All datasets drawn from existing literature are publicly available **Yes**
  - All non-public datasets are described in detail, explaining why publicly available alternatives are insufficient **N/A**

8.3 Computational Experiments

- Does this paper include computational experiments? **Yes**
  - Code required for pre-processing data is included in the appendix **Yes**
  - All source code for conducting and analyzing experiments is included in a code appendix **Yes**
  - All source code will be made publicly available upon publication with a license allowing free research use **Yes**
  - Code implementing new methods is commented and references steps from the paper **Yes**
  - If algorithms depend on randomness, seeding methods are described **Yes**
  - Computing infrastructure (hardware/software, GPU/CPU models, memory, OS, libraries) is specified **Yes**
  - Evaluation metrics are formally described with justification for their choice **Yes**
  - Number of algorithm runs used per result is stated **N/A** - we are unable to run multiple runs of our inference because of resource constraints.
  - Experimental analysis includes measures of variation, confidence, or distributional information **No** - Since we use GPT judge it is difficult to measure statistical attributes to recover distribution information.
  - Performance improvements are judged with appropriate statistical tests (e.g., Wilcoxon signed-rank) **N/A**
  - Final hyperparameters are listed **Yes**
  - Number and range of hyperparameters tried during development, along with selection criteria, are stated **Yes**