Enhance Multimodal Consistency and Coherence for Text-Image Plan Generation

Xiaoxin Lu Ranran Haoran Zhang Yusen Zhang Rui Zhang
The Pennsylvania State University, State College, PA, USA
`{xzl5514, haoranz6, yfz5488, rmz5227}@psu.edu`

Abstract

People get informed of a daily task plan through diverse media involving both texts and images. However, most prior research only focuses on LLM's capability of textual plan generation. The potential of large-scale models in providing text-image plans remains understudied. Generating high-quality text-image plans faces two main challenges: ensuring consistent alignment between two modalities and keeping coherence among visual steps. To address these challenges, we propose a novel framework that generates and refines text-image plans step-by-step. At each iteration, our framework (1) drafts the next textual step based on the prediction history; (2) edits the last visual step to obtain the next one; (3) extracts PDDL-like visual information; and (4) refines the draft with the extracted visual information. The textual and visual step produced in stage (4) and (2) will then serve as inputs for the next iteration. Our approach offers a plug-and-play improvement to various backbone models, such as Mistral-7B, Gemini-1.5, and GPT-4o. To evaluate the effectiveness of our approach, we collect a new benchmark consisting of 1,100 tasks and their text-image pair solutions covering 11 daily topics. We also design and validate a new set of metrics to evaluate the multimodal consistency and coherence in text-image plans. Extensive experiment results show the effectiveness of our approach on a range of backbone models against competitive baselines. Our code and data are available at `https://github.com/psunlpgroup/MPlanner`.

1 Introduction

Recently, there has been growing attention on employing LLMs for planning, the task of decomposing a high-level goal into a sequence of executable steps (Valmeekam et al., 2022; Hao et al., 2023). LLMs have demonstrated strong capabilities in generating textual plans, enabling applications in robotics, virtual assistants, and instructional content generation (Huang et al., 2022; Liu et al., 2023; Silver et al., 2024). However, textual plans alone can be insufficient, as many real-world

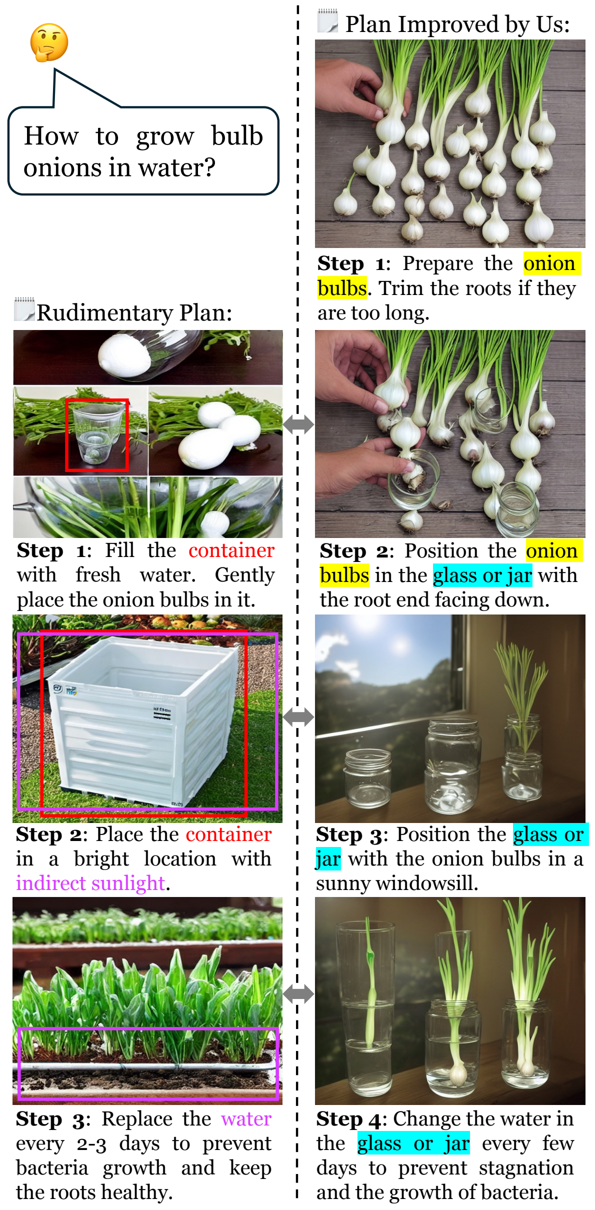

How to grow bulb onions in water?
Rudimentary Plan:
Step 1: Fill the container with fresh water. Gently place the onion bulbs in it.
Step 2: Place the container in a bright location with indirect sunlight.
Step 3: Replace the water every 2-3 days to prevent bacteria growth and keep the roots healthy.
Plan Improved by Us:
Step 1: Prepare the onion bulbs. Trim the roots if they are too long.
Step 2: Position the onion bulbs in the glass or jar with the root end facing down.
Step 3: Position the glass or jar with the onion bulbs in a sunny windowsill.
Step 4: Change the water in the glass or jar every few days to prevent stagnation and the growth of bacteria.

Figure 1: Plans generated by GPT-4o (left) and our framework (right). Our framework maintains higher consistency between images and texts, and achieves higher coherence among images across different steps.

tasks require both textual instructions and visual demonstrations for clarity. Multimodal task planning, which can be formulated as a paired text-image sequence generation problem conditioned on the task goal (Lu et al., 2024), enhances comprehensibility and usability by leveraging the complementary strengths of language and vision. Despite its potential, multimodal planning remains an underexplored problem, with prior approaches struggling to maintain consistency between modalities and coherence across visual steps.

Multimodal task planning faces two main challenges: ensuring coherence through visual steps and alignment between two modalities. Figure 1 illustrates these challenges through a comparative example of generating plans for the input goal "growing bulb onions in water". The left-hand side shows the plan generated by GPT-4o, and the right-hand side presents the corresponding plan generated by our framework. The baseline plan, although generally reasonable, exhibits critical issues in both visual coherence and text-image alignment. First, it fails to maintain visual consistency between step 1 and step 2, as highlighted in red. The container depicted in step 1 is a transparent glass, but in step 2, it becomes an opaque white planter, disrupting coherence. Second, as highlighted in purple, step 2's image shows a location with direct sunlight, contradicting the textual instruction that specifies "indirect sunlight". Furthermore, step 3 entirely diverges from the intended goal of growing onions *in water*, as it depicts bulbs planted in soil.

On the other hand, our framework effectively improves visual coherence and text-image consistency. As shown on the right-hand side of Figure 1, our approach maintains a uniform depiction of the onion bulbs and their container throughout the process. The glass or jar remains consistent in shape and size from step 2 to step 4, avoiding abrupt visual changes. Moreover, the visual steps accurately reflect the textual descriptions through all steps. For instance, step 3 correctly depicts a sunny windowsill, aligning with the text's instruction to position the jar in such a location.

These improvements stem from our novel autoregressive framework. At each iteration, the text generator drafts the current step based on the task goal and previous steps. The image generator then produces a corresponding visual representation conditioned on the last visual step. An image interpreter subsequently extracts structured information from the generated image, which the text generator uses to refine the step draft. This iterative process enhances visual coherence with a text-image-to-image model and ensures text-image consistency through cross-modality prompting. Compared to vanilla approaches that simply concatenate an LLM with a text-to-image model, our framework effectively mitigates the challenges of multimodal task planning, leading to more coherent and consistent instructional sequences.

To evaluate our framework, we collect a dataset from two popular websites: Instructables and wikiHow. Our dataset consisting of 1100 examples in 11 categories, providing rich multimodal information of procedural solutions to a variety of daily tasks. We adopt a set of metrics including conventional automatic measurements, LLM evaluations, and human evaluations to comprehensively evaluate planning performance in three aspects: textual plan quality, visual plan quality, and textual-visual plan alignment. To demonstrate the generalizability of our approach, we evaluate our framework with 3 different backbones: Mistral-7B (Jiang et al., 2023), Gemini-1.5-flash (Team et al., 2024), and GPT-4o (Hurst et al., 2024). For every backbone, we compare our framework with various baselines. Extensive experiment results show our framework outperforms all baselines, especially in terms of the two concerns we aim to address.

In summary, our contributions are three-fold:

- We propose a novel framework to address both visual coherence challenge and text-image alignment challenge of multimodal planning problem;
- We collect a dataset of daily tasks covering diverse domains and complexity levels to evaluate the text-image planning performance.
- We empirically show the effectiveness of our framework with extensive experimental results and visualization examples.

2 Related Work

Task Planning Task planning is broadly studied in various scenarios of virtual environment (Zhao et al., 2023; Gao et al., 2023; Hu et al., 2024), embodied environment (Huang et al., 2022; Song et al., 2023; Zhang et al., 2024), and daily life (Oswald et al., 2024; Wu et al., 2022). Despite classical planning algorithms (Hoffmann and Nebel, 2001; Alarnaouti et al., 2023), they are primarily applicable with restrictions such as fully observable environments with pre-defined actions and objects. It prohibits their usage in open-domain daily scenarios, igniting research interest in solutions from LLM advancements (Kambhampati et al., 2024).

LLMs, possessing a rich amount of commonsense knowledge and impressive reasoning capability, are competent in such contexts. Huang et al. (2022) studies LLMs as zero-shot planners and shows their great planning potential in a virtual environment. Liu et al. (2023) combines the LLM with classical planners to get reliable solutions to robotic tasks in an embodied environment. Arora and Kambhampati (2023) tries to add an external verifier to improve the planning capabilities of a finetuned LLM. Wang et al. (2023) seeks to study textual plan generation provided visual states as supplements. Despite these, only rare efforts contribute to studying multimodal planning. Lu et al. (2024) first explores the potential of LLMs and text-to-image models to generate image-text paired plans. However, their approach is limited by the single-shot generation framework and reliance on image captions that inadequately capture object interactions, leading to information loss and visual incoherence in planning sequences.

Vision and Language Diffusion models (Ho et al., 2020; Ramesh et al., 2021; Rombach et al., 2022) have revolutionized image generation conditioned on texts. However, in specific circumstances where the models are expected to output coherent images, they exhibit poor performance because of the lack of visual context knowledge (Lu et al., 2024). This leads to the introduction of advanced image editing models (Kawar et al., 2023). Building upon the success of image generation models, they are capable of generating images conditioned on both reference images and text instructions. For example, Brooks et al. (2023) achieves image editing by fine-tuning the stable diffusion model on (original image, instruction, edited image) triplets. Zhang et al. (2023) enables precise spatial conditioning controls including edges, depth maps, pose information, etc. Souček et al. (2024) infuses video data into a diffusion model to generate images depicting how actions lead to object state transformations. However, they all tend to maintain the outline of objects in the original image while only editing the color, texture, or style. Thus, they are still inadequate in generating coherent images for planning, which typically involves scenario change and object transformation.

3 Dataset

Despite existing datasets in the daily task planning area, they either lack image modality (Koupaee and Wang, 2018; Valmeekam et al., 2023a), task domain diversity (Yagcioglu et al., 2018; Valmeekam et al., 2023b), or are not intended for plan generation (Yang et al., 2021). Therefore, to benchmark text-image plan generation, we collect a dataset of daily task plans covering various topics.

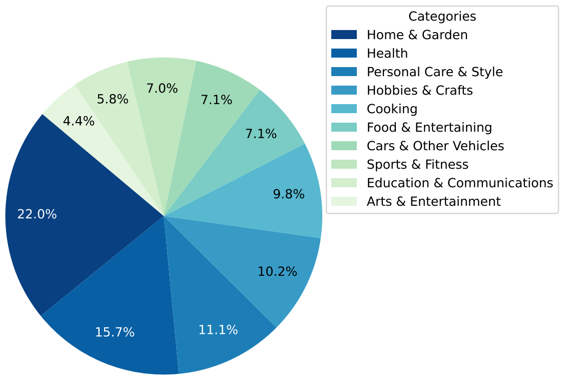


Figure 2: The distribution of our dataset.

Inspired by previous work, we consider two popular websites affording procedural daily task instructions: Instructables[1] and wikiHow[2]. Both data sources provide various modalities and cover diverse daily task categories. The overall category distribution of our dataset is shown in Figure 2. Compared with previous public datasets, our dataset is rich in task categories and plan modalities. Furthermore, we ensure high data quality by manually filtering out malicious content and curating the collected plans to remove noises such as the authors' information and personal stories. Please see Appendix A for more dataset statistics and demonstrations.

Instructables To study how our framework performs on simple daily tasks, we randomly collect 100 plans from Instructable. Specifically, we choose "Cooking" category as it is a common planning scenario. Plans from Instructables are published by users who would like to share their experience in achieving a specific goal. Therefore, their plans are more brief and more casual.

wikiHow We randomly sample 1,000 plans from wikiHow expert articles. In 19 categories, we select 10 that best fit into the multimodal planning context. The remaining categories, such as family life, relationships, philosophy and religion, etc, often involve sensitive information. Moreover, they are not suitable for our planning problem since the provided "plans" are more like "advice" with-

[1]https://www.instructables.com/
[2]https://www.wikihow.com/

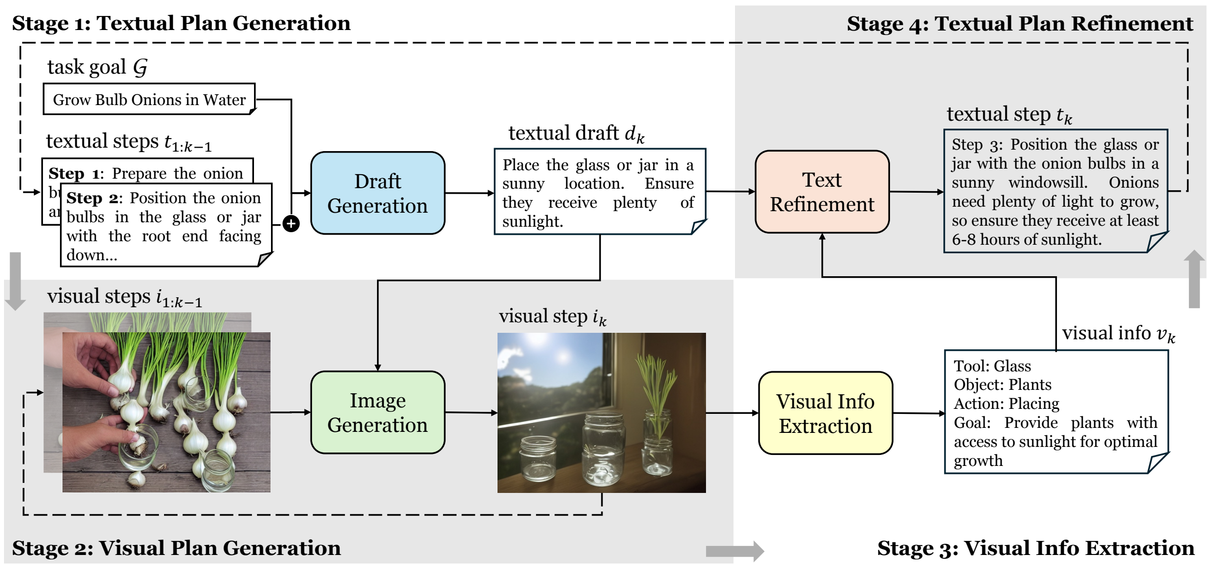

Stage 1: Textual Plan Generation
Stage 4: Textual Plan Refinement
task goal $\mathcal{G}$
Grow Bulb Onions in Water
textual steps $t_{1:k-1}$
Step 1: Prepare the onion
Step 2: Position the onion bulbs in the glass or jar with the root end facing down...
Draft Generation
textual draft d_k
Place the glass or jar in a sunny location. Ensure they receive plenty of sunlight.
Text Refinement
textual step t_k
Step 3: Position the glass or jar with the onion bulbs in a sunny windowsill. Onions need plenty of light to grow, so ensure they receive at least 6-8 hours of sunlight.
visual steps $i_{1:k-1}$
Image Generation
visual step i_k
Visual Info Extraction
visual info v_k
Tool: Glass
Object: Plants
Action: Placing
Goal: Provide plants with access to sunlight for optimal growth
Stage 2: Visual Plan Generation
Stage 3: Visual Info Extraction

Figure 3: Overview of our autoregressive framework at time step k. Stage 1: the text plan generator takes the input task goal $\mathcal{G}$ and output textual steps $t_{1:k-1}$ from previous time steps to predict d_k. Stage 2: the image generator outputs the visual step i_k conditioned on d_k and the last visual step i_{k-1}. Stage 3: the text generator extracts formatted visual information from the generated i_k as v_k. Stage 4: the text generator refines d_k with v_k to generate t_k. Dotted Arrow: The output t_k and i_k then serve as the input for the next time step $k+1$.

out explicit temporal consistency and dependency through the "steps". Our collected wikiHow articles are reviewed, edited, or authored by domain experts verified by the platform. Therefore, these plans are longer, more detailed, and more formal.

4 Approach

Task Formulation Given a task goal $\mathcal{G}$ as the input, the output of multimodal planning is a sequence of steps $\mathcal{S} = \{s_1, s_2, ..., s_n\}$. Each step consists of an instructional text t_k and its corresponding image i_k, so we have $s_k = (t_k, i_k), k \in \{1, 2, ..., n\}$. At every step k, t_k and i_k should be semantically consistent. Both textual plan $\mathcal{S}_t = \{t_1, ..., t_n\}$ and visual plan $\mathcal{S}_i = \{i_1, ..., i_n\}$ should be coherent through steps.

Framework Overview As shown in Figure 3, our framework generates plan steps iteratively. At each step, 4 cross-modality stages collaboratively contribute to the multimodal plan generation. In the first stage, the textual plan drafter takes the input task goal $\mathcal{G}$ and the previous textual steps $t_{1:k-1}$ to draft the current textual step d_k. In stage two, the image generator edits the last visual step i_{k-1} conditioned on d_k to get the current visual step i_k. In stage three, an image interpreter extracts formatted visual information v_k from the generated i_k. Last, the textual plan refiner collects v_k and uses it to refine the original textual step draft d_k to be t_k. After the current iteration is complete, both t_k and i_k are added to history steps for the next iteration. The overall design ensures the coherence of visual steps by prompting the image generation module to predict the next image based on both textual instruction and the previous visual state. Combining text-to-image and converse image-to-text in a loop guarantees consistency between the generated textual plan and its visual counterpart. Detailed prompts for each module are in Appendix B.

4.1 Textual Plan Drafting

As LLMs embed a rich amount of commonsense knowledge, we prompt them with the task goal $\mathcal{G}$ at the first iteration to get the textual step draft d_1. At later iterations, we concatenate $\mathcal{G}$ with previous textual steps $t_{1:k-1}$ to draft d_k:

$$d_k = \begin{cases} \mathbf{G}_t(\mathcal{G}), & k = 1 \\ \mathbf{G}_t(\mathcal{G}, \text{Concat}(t_1, t_2, ..., t_{k-1})), & k > 1 \end{cases} \quad (1)$$

where $\mathbf{G}_t$ denotes the text generation model.

4.2 Visual Plan Generation

To maintain coherence through visual steps, we employ InstructPix2Pix (Brooks et al., 2023), a prevailing image editing model. Different from image generation models, InstructPix2Pix is conditioned on both the text prompt and the input image. In our framework, it ensures the generation of the current visual step i_k is always aware of the last visual state i_{k-1}. This helps improve the coherence through the visual plan $\mathcal{S}_i = \{i_1, i_2, ..., i_n\}$.

$$i_k = \mathbf{G}_i(d_k, i_{k-1}) \tag{2}$$

where $\mathbf{G}_i$ denotes the image generation model.

4.3 Textual Plan Refinement

We further employ a textual plan refinement mechanism for two reasons: 1) to improve the consistency between textual step and visual step; 2) to complement the textual step draft with implicit knowledge embedded in the image.

Visual Information Extraction First, we adopt a visual information-infused model as the image interpreter to extract visual information. Specifically, we adopt the idea of planning domain definition language (PDDL) from the classical planning field (Fox and Long, 2003). We design a pseudo-PDDL (pPDDL) with structured representations that are applicable in our planning scenario. As displayed in Figure 3, we specify this pPDDL as 4 different types of information embedded in a single image: involved objects, tools, the action, and the goal.

$$v_k = \mathbf{E}(i_k) \tag{3}$$

$\mathbf{E}$ is a language model to generate pPDDL.

Visual Information Incorporation Last, we feed the extracted visual information into the plan generator to revise the textual step draft.

$$t_k = \mathbf{G}_t(d_k, v_k) \tag{4}$$

5 Experiment Settings

5.1 Baselines

To demonstrate the effectiveness of our model, we compare with three types of strong multimodal planning baselines based on state-of-the-art LLMs and diffusion models:

- Vanilla LLM baselines including GEMINI-1.5-FLASH, GPT-4O, and LLaVa with instruction finetuning using Mistral (M&L). The backbone LLM generates textual plan in one turn, and Stable Diffusion model generates the visual plan according to parsed textual steps.
- SD: Stable Diffusion (Rombach et al., 2022) generates one visual step at each turn given the task goal $\mathcal{G}$, and the backbone LLM describes each visual step to form a textual plan.
- We also compare with a state-of-the-art multimodal planning framework TIP (Lu et al., 2024). Compared with the first type baseline, it leverages cross-modality prompting with a T2I-Bridge and an I2T-Bridge to further improve the performance. For fair comparison, we use TIP on top of the same backbone language model choices as ours.

5.2 Evaluation Metrics

Textual Plan Evaluation We conventionally choose BertScore (Zhang et al., 2020) and ROUGE (R-1, R-2, R-L) (Lin, 2004) to automatically measure the semantic similarity between generated textual plans and reference textual plans. Considering the open-ended attribute of the daily task planning problem, automatic metrics do not suffice to evaluate plan quality. Therefore, we also include four qualitative metrics: correctness, executability, coherence, and informativeness. We use Claude-3.5-Sonnet as the LLM judge to score generated plans with awareness of reference plans and have three human annotators verify the LLM evaluation's reliance. Please see Appendix B for more details.

Text-Image Evaluation Following the common practice, we use CLIP score to examine the semantic alignment between textual steps and their visual counterparts. Like textual plan evaluation and visual plan evaluation, we use an LLM judge and human annotators to perform evaluations as well.

Visual Plan Evaluation Evaluating visual plans is nontrivial due to four cases through the plan: scenario coherence/change when actions are conducted in the same workplace/different workplaces; object coherence/change when involved objects are unchanged/changed in their states by some actions. Therefore, conventional image similarity measurements like FID (Heusel et al., 2017) do not fit into this context. To this end, we first convert every image to a textual description of the background, salient objects, and the involved action (if any). Then, we employ the perplexity score (PPL) to check if consecutive descriptions are coherent *given the action corresponding to the later step*. In addition, we also conduct LLM evaluations and human evaluations.

5.3 Implementation Details

Backbone LLMs To examine the effectiveness of our approach on both open-source and closed-source LLMs, we choose 3 different models as the backbone: MISTRAL-7B, GEMINI-1.5-FLASH, and GPT-4O. In all experiments, the backbone LLM

Backbone & Dataset	Method	Automatic Evaluation						LLM Evaluation					
		BertScore ↑	R-1 ↑	R-2 ↑	R-L ↑	CLIP ↑	PPL ↓	Corr. ↑	Exec. ↑	Coh. ↑	Info. ↑	T-I ↑	I-I ↑
Gemini & Instructables	GEMINI	0.835	26.3	7.00	24.7	17.84	5.98	4.81	4.84	**4.96**	4.85	1.59	2.33
	SD	0.807	20.7	5.20	19.7	8.59	**4.85**	1.17	0.74	0.85	2.33	1.04	1.29
	TIP	0.812	22.2	5.00	20.5	18.66	6.11	3.15	3.10	3.54	3.57	1.73	2.15
	OURS	**0.842**	**30.6**	**9.20**	**28.2**	**26.38**	5.49	**4.85**	**4.89**	4.91	**4.94**	**2.64**	**2.41**
Gemini & wikiHow	GEMINI	0.847	26.1	7.30	24.4	15.61	5.92	4.83	4.84	**4.97**	4.82	1.69	2.32
	SD	0.806	14.4	2.00	13.7	9.03	**4.94**	0.79	0.70	0.81	1.94	1.28	1.13
	TIP	0.812	21.5	5.40	20.4	14.58	6.01	3.32	3.47	3.51	3.82	1.58	2.31
	OURS	**0.850**	**29.0**	**9.00**	**27.0**	**20.23**	5.13	**4.89**	**4.88**	4.89	**4.91**	**2.42**	**2.40**
GPT & Instructables	GPT	0.827	27.8	7.40	26.0	12.32	5.75	4.90	4.87	**4.97**	4.83	1.53	2.47
	SD	0.805	19.4	4.30	18.4	9.65	**5.09**	1.33	0.92	0.79	2.04	1.10	1.24
	TIP	0.840	29.8	8.20	28.0	13.19	6.27	3.78	3.25	3.63	3.61	1.68	2.30
	OURS	**0.849**	**33.7**	**10.3**	**31.5**	**27.14**	5.21	**4.93**	**4.90**	4.93	**4.93**	**2.47**	**2.76**
GPT & wikiHow	GPT	0.850	30.0	9.00	28.1	11.29	5.83	4.84	4.86	**4.97**	4.87	1.56	2.35
	SD	0.811	20.3	4.80	19.2	10.37	**5.17**	1.19	0.80	0.84	2.12	1.08	1.13
	TIP	0.843	29.9	8.70	27.9	11.81	5.97	3.50	3.71	3.88	3.68	1.73	2.21
	OURS	**0.856**	**33.2**	**10.5**	**30.9**	**24.62**	5.30	**4.88**	**4.91**	4.90	**4.94**	**2.58**	**2.68**
M&L & Instructables	M&L	0.829	30.8	9.10	28.8	19.36	6.03	4.79	4.58	**4.80**	4.67	1.51	2.26
	SD	0.807	20.7	5.20	19.7	11.21	**5.31**	0.83	0.75	0.71	1.96	1.06	1.25
	TIP	0.842	30.5	9.50	29.3	20.07	5.99	4.02	3.85	4.07	3.73	1.82	2.18
	OURS	**0.848**	**32.5**	**10.1**	**30.1**	**26.39**	5.47	**4.81**	**4.74**	4.66	**4.74**	**2.58**	**2.70**
M&L & wikiHow	M&L	0.836	30.0	9.00	28.1	15.82	6.19	4.75	4.62	**4.77**	4.60	1.53	2.29
	SD	0.809	20.0	4.40	19.3	10.19	**5.20**	0.78	0.77	0.80	2.03	1.10	1.19
	TIP	0.839	30.1	8.90	27.8	15.47	6.16	3.84	3.68	3.92	3.70	1.84	2.07
	OURS	**0.851**	**31.7**	**10.0**	**29.6**	**19.60**	5.27	**4.83**	**4.75**	4.71	**4.78**	**2.45**	**2.53**

Table 1: Main results. The best results are in **bold**, and the second best results are underlined. Textual Plan Evaluation: BertScore, R-1, R-2, R-L, Corr.(correctness), Exec.(executability), Coh.(coherence), Info.(informativeness). Text-Image Evaluation: CLIP, T-I (text-image). Visual Plan Evaluation: PPL, I-I (image-image).

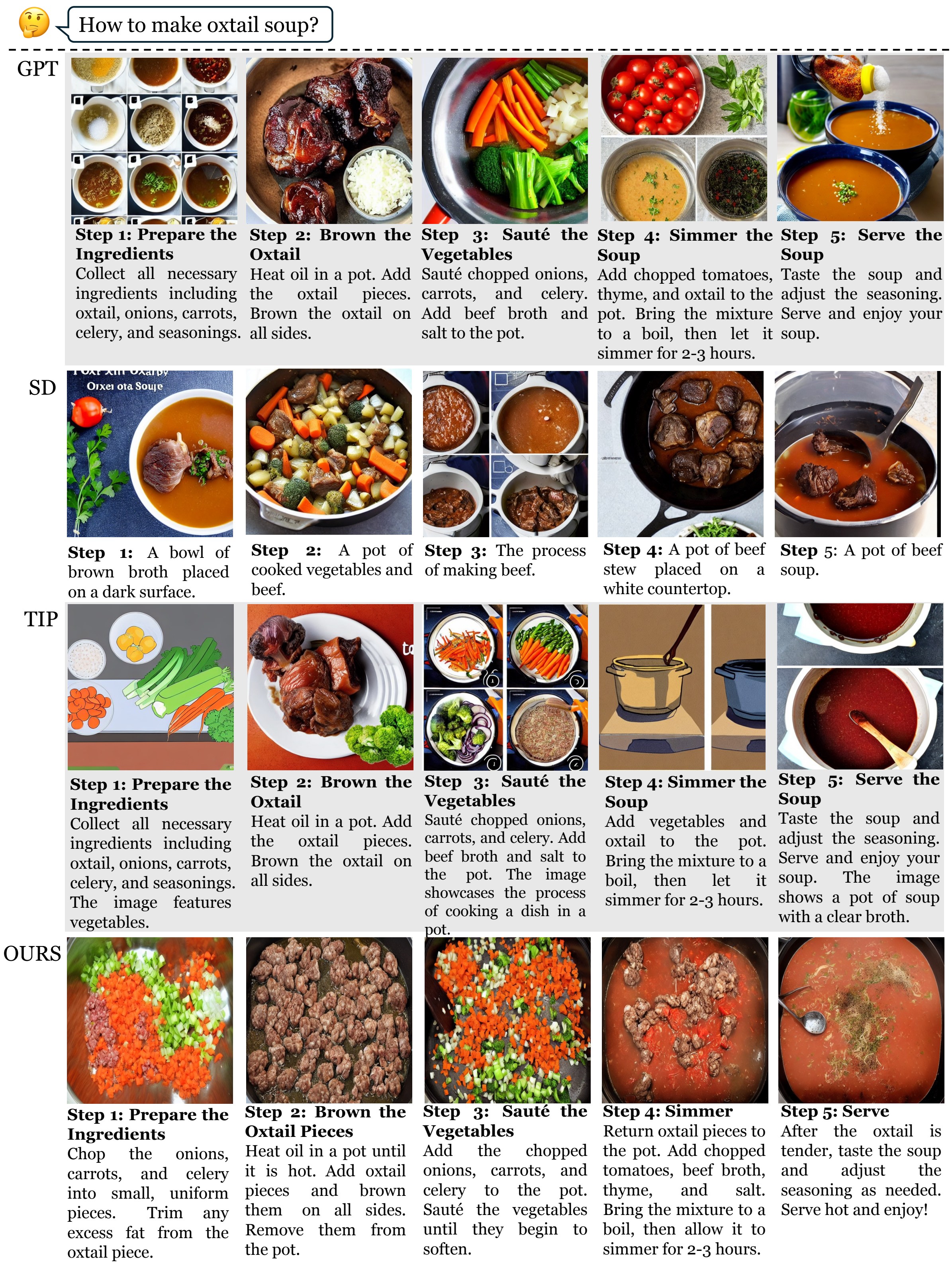
works both as the draft generator and the text refinement editor. Since GEMINI-1.5-FLASH and GPT-4O can also interpret images, we use them as the visual information extractor in our framework. For experiments with MISTRAL-7B backbone, we choose INSTRUCTBLIP-VICUNA-7B, a general-purpose MLLM tuned on diverse tasks, to extract visual information in the framework.

InstructPix2Pix Finetuning To make the image generator better align with our context, we fine-tune INSTRUCTPIX2PIX on a re-purposed dataset collected from wikiHow (Yang et al., 2021). The original dataset includes more than 60,000 tasks covering all categories in wikiHow. We sample 20,000 of them in the scope of our selected categories as shown in Figure 2. Furthermore, we transform each plan into a series of $\{i_{k-1}, t_k, i_k\}$ triplets, where i_{k-1} and i_k are two consecutive images, and t_k the text corresponding to the later image. The fine-tuning aims to promote INSTRUCTPIX2PIX's capability to generate coherent i_k given i_{k-1} and t_k as input. We split the re-purposed dataset by tasks in 0.9/0.05/0.05 for training/validation/test. We follow most training hyperparameters in the original work but change the maximum number of epochs to 50. At the end of training, we achieve training loss of 0.100 and validation loss of 0.105.

6 Results and Analysis

6.1 Main Results

Table 1 shows automatic measurements and LLM evaluation results with three different backbone models. Despite value differences across three sets of experiments, they exhibit the same trends in all three aspects. OURS approach obtains consistently higher scores on all automatic measurements of textual plans. In LLM evaluation, the coherence of textual plans generated by OURS approach ranks second to the vanilla baselines GEMINI, GPT, and M&L. The potential cause is that generating the whole plan in one shot prevents semantic conflicts and temporal inconsistency. It is also noteworthy that all baselines and OURS approach obtain better scores in terms of textual plan evaluation, but obtain worse scores in terms of visual plan and text-image pair evaluation on wikiHow tasks. The observation is accountable provided different plan lengths and language styles of two data sources as we discussed in Section 3 and Appendix A.



GPT

Step 1: Prepare the Ingredients Collect all necessary ingredients including oxtail, onions, carrots, celery, and seasonings.

Step 2: Brown the Oxtail Heat oil in a pot. Add the oxtail pieces. Brown the oxtail on all sides.

Step 3: Sauté the Vegetables Sauté chopped onions, carrots, and celery. Add beef broth and salt to the pot.

Step 4: Simmer the Soup Add chopped tomatoes, thyme, and oxtail to the pot. Bring the mixture to a boil, then let it simmer for 2-3 hours.

Step 5: Serve the Soup Taste the soup and adjust the seasoning. Serve and enjoy your soup.

SD

Step 1: A bowl of brown broth placed on a dark surface.

Step 2: A pot of cooked vegetables and beef.

Step 3: The process of making beef.

Step 4: A pot of beef stew placed on a white countertop.

Step 5: A pot of beef soup.

TIP

Step 1: Prepare the Ingredients Collect all necessary ingredients including oxtail, onions, carrots, celery, and seasonings. The image features vegetables.

Step 2: Brown the Oxtail Heat oil in a pot. Add the oxtail pieces. Brown the oxtail on all sides.

Step 3: Sauté the Vegetables Sauté chopped onions, carrots, and celery. Add beef broth and salt to the pot. The image showcases the process of cooking a dish in a pot.

Step 4: Simmer the Soup Add vegetables and oxtail to the pot. Bring the mixture to a boil, then let it simmer for 2-3 hours.

Step 5: Serve the Soup Taste the soup and adjust the seasoning. Serve and enjoy your soup. The image shows a pot of soup with a clear broth.

OURS

Step 1: Prepare the Ingredients Chop the onions, carrots, and celery into small, uniform pieces. Trim any excess fat from the oxtail piece.

Step 2: Brown the Oxtail Pieces Heat oil in a pot until it is hot. Add oxtail pieces and brown them on all sides. Remove them from the pot.

Step 3: Sauté the Vegetables Add the chopped onions, carrots, and celery to the pot. Sauté the vegetables until they begin to soften.

Step 4: Simmer Return oxtail pieces to the pot. Add chopped tomatoes, beef broth, thyme, and salt. Bring the mixture to a boil, then allow it to simmer for 2-3 hours.

Step 5: Serve After the oxtail is tender, taste the soup and adjust the seasoning as needed. Serve hot and enjoy!

Figure 4: A qualitative comparison of plans output by all three baselines and our approach.

6.2 Visual Coherence Analyses

From Table 1, Ours approach obtains consistent improvement in terms of PPL score and I-I score by LLM judge. Although Sd baseline achieves the best PPL score in all cases, the underlying reason is that the image generation model is incapable of actual "task planning". When the planning instructions and task goal $\mathcal{G}$ are fed into it, it concentrates on the project or action mentioned in $\mathcal{G}$ and outputs images about the topic. As shown in Figure 4, the visual plan generated by Sd fails to showcase expected object transitions in temporal logic. Therefore, the translated textual "steps" are semantically similar, further leading to lower PPL score.

For other baselines, we observe that they tend to involve sharp transitions in the visual plan. Figure 4 intensively shows such failures in maintaining visual coherence. For example, the visual plan generated by Gpt presents a cooking course where consecutive steps appear irrelevant. It is due to the ignorance of its image generation module about the last visual step. In contrast, the Ours approach addresses this challenge by predicting the next image conditioned on both the textual instruction and the last visual step. Consequently, the cooking course is smooth with natural object transitions.

6.3 Text-Image Consistency Analyses

Ours approach outperforms all baselines in terms of CLIP score and T-I score by LLM judge. Ours approach ensures high consistency between two modalities with two passes that first use the textual draft to instruct image generation and then refine the draft with extracted information from the generated image. As presented in Figure 4, the visual step in Tip's plan digresses from the task goal and its corresponding textual instruction. It is hard to observe "oxtail" from step 3 on. The cause is that without awareness of the previous steps, the image generation module is easily attracted by the most mentioned objects in the text. On the other hand, Ours approach counteracts the textual noise with visual information from the last step.

6.4 Human Evaluation

To complement our automatic and LLM evaluation, we conduct a human evaluation by recruiting 3 annotators. We randomly select 50 tasks as a validation dataset. Its category distribution is consistent with the overall dataset. We choose the strong baseline Gpt as one candidate and Ours with the same backbone as the other. We show the plans generated by these two models in random order, and the annotators are asked to make comparative annotations (win/tie/lose) between the generated plans in three aspects: textual plan quality, visual plan coherence, and text-image alignment. Please check Appendix C.2 for more evaluation details.

Eval.	Ours v.s. Gpt-4o			κ
	Win	Tie	Lose	
Text	34%	53%	13%	0.521
Image	77%	15%	8%	0.604
T-I	81%	15%	4%	0.699

Table 2: Human evaluation results. The κ scores demonstrate moderate to substantial inter-annotator agreement.

Table 2 shows the human evaluation results. It verifies the effectiveness of Ours approach in all 3 dimensions and is generally in line with automatic and LLM evaluations. Compared with the baseline Gpt, Ours approach obtains slight improvements in textual plan quality. Regarding visual plan coherence and text-image alignment, Ours approach exhibits significant superiority.

6.5 Ablation Studies

To further examine the effects of our formatted visual information design, we implement three ablation studies with backbone gpt-4o including (1) w Des which replaces the formatted visual information v_k in our framework with general image descriptions; (2) w Img which directly feeds the generated image i_k and the draft d_k into the multimodal text refinement module and obtains the output textual step t_k; and (3) pPddl-to-Nl which conversely asks the draft generator to provide formatted draft as image generation guidance and translates it into natural language afterward. The experiment results are shown in Table 3.

Formatted v.s. General Visual Information Aiming to study pPDDL's necessity, we edit the prompt in stage 3 to make the visual information extractor describe i_k in natural language (NL). The results in Table 3 indicate a performance drop induced by this modification. From its intermediate outputs, we find that without a pre-defined format restriction, the image-to-text model tends to generate lengthy, unstructured descriptions with much attention on minutia. They introduce noise to text refinement in stage 4. As our framework is autoregressive, the noise accumulates with time steps, harming both the textual and visual plans.

Dataset	Method	Automatic Evaluation						LLM Evaluation					
		BertScore ↑	R-1 ↑	R-2 ↑	R-L ↑	CLIP ↑	PPL ↓	Corr. ↑	Exec. ↑	Coh. ↑	Info. ↑	T-I ↑	I-I ↑
Instructables	W DES	0.841	0.295	0.083	0.279	14.71	5.92	4.63	4.17	4.34	4.58	1.76	2.39
	W IMG	0.836	0.257	0.069	0.245	16.48	5.90	4.40	4.24	4.19	4.47	1.72	2.26
	PPDDL-TO-NL	0.837	0.261	0.079	0.243	12.04	6.25	4.18	4.20	3.93	4.02	1.58	2.09
	OURS	**0.849**	**0.337**	**0.103**	**0.315**	**27.14**	**5.21**	**4.93**	**4.90**	**4.93**	**4.93**	**2.47**	**2.76**
wikiHow	W DES	0.847	0.309	0.092	0.290	12.04	5.89	4.60	4.28	4.40	4.84	1.85	2.34
	W IMG	0.849	0.298	0.090	0.278	11.97	5.76	4.36	4.31	4.48	4.77	1.79	2.18
	PPDDL-TO-NL	0.840	0.279	0.081	0.255	11.35	6.38	4.14	4.02	3.95	4.33	1.45	1.97
	OURS	**0.856**	**0.332**	**0.105**	**0.309**	**14.62**	**5.30**	**4.88**	**4.91**	**4.90**	**4.94**	**2.58**	**2.68**

Table 3: Ablation study using GPT-4o backbone.

Formatted v.s. Raw Visual Information Given that GEMINI-1.5-FLASH and GPT-4O are infused with visual knowledge, we seek to explore whether these models can directly use raw image i_k to refine the draft d_k. In this case, stage 3 is skipped. Table 3 demonstrates the failure of depending on their image interpretation capability to refine textual plans. Without an external visual information extractor, they are likely to roughly append their image interpretations to the input drafts, leading to worse plan coherence.

NL-to-pPDDL v.s. pPDDL-to-NL To determine whether the pPDDL functions better than NL for image generation, we exchange the order of generating texts in two formats. In this ablation study, the draft generation module is prompted to generate a pPDDL step d_k in stage 1. In stage 3, we ask the visual information extractor to describe i_k in NL v_k. In stage 4, the text refinement is still based on both d_k and v_k.

From Table 3, OURS approach outperforms the pPDDL-to-NL method. Inspecting the concrete generation results, we observe that the pPDDL-first design can not yield sufficient information for the image generation module in stage 2. The highly compact drafts make it difficult to generate visual counterparts that align with textual instructions. It further disables the visual information extractor from generating accurate image descriptions. The refined textual steps are often accordingly ambiguous. Therefore, the overall evaluation of this method is poor in all aspects.

6.6 Sensitivity to Task Complexity

We observe that diverse task categories introduce different task complexities. For instance, "Home & Garden" tasks are moderately challenging, and both textual and visual plans provide similar information. "Hobbies & Craft" tasks are often complex and require delicate actions. In this case, the visual plan is more informative than the textual plan. "Education & Communications" tasks are relatively abstract. The textual plan is better for such tasks since the visual plan always depicts concrete actions or scenarios. To study the effectiveness of our framework over varying task complexities, we sample 100 tasks and manually annotate them with complexity high, medium, or low.

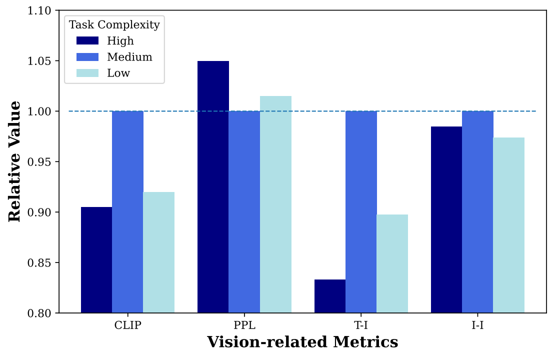


Figure 5: Visual step coherence and text-image alignment across different task complexity levels. All values are normalized relative to the results from medium-complexity tasks. For PPL score, the lower, the better.

Figure 5 demonstrates our approach yields better visual plans when challenged with medium-complexity tasks. Involved actions in such plans are often concrete while not elaborate, enabling the image generation model to visualize them precisely. However, it struggles with high- and low-complexity tasks. Depicting abstract or delicate actions still exceeds the capacity of current models.

7 Conclusion

Our work studies an underexplored problem of text-image plan generation. We identify two main challenges: ensuring visual coherence and text-image alignment, and propose a novel framework to address them accordingly. For evaluation, we collect a dataset of daily tasks covering diverse domains and task complexities. Substantial experiment results demonstrate the effectiveness of our approach on a range of various backbone models, especially in terms of the two challenges we aim to address.

Limitations

While our approach exhibits promising results in improving text-image plan generation, it is noteworthy that our work has several limitations.

First, given that LLMs are trained on vast amounts of data, data leakage is inevitable. This inherent characteristic potentially contributes to their strong performance in textual plan generation, as similar patterns may exist in their training data. Second, the quality of visual plans generated by fine-tuned InstructPix2Pix suggests room for improvement. Although it is capable of maintaining visual coherence when the scenario and object have minor transitions, we still observe unexpected incoherence when the textual instructions indicate significant workspace change. Last, the measurement of visual coherence through text-based metrics is indirect. The textual descriptions converted from images may not fully capture the nuanced visual relationships and coherence patterns that exist in the original images, potentially affecting the validity of our evaluation.

These limitations shed light on potential directions for future work, including the exploration of image editing models that better fit into the planning context, the development of visual coherence metrics that function in the image space, etc.

Ethics Statement

Instructables and wikiHow are two public online platforms licensed under CC BY-NC-SA 3.0 and CC BY-NC-SA 4.0, respectively. Our data collection process complies with their licensing terms, as both licenses permit academic use with proper attribution. Our work primarily focuses on generating text-image plans for daily tasks. Therefore, we exclude potentially inappropriate content in the process of data collection and manually inspect data quality. To prevent privacy leakage, we anonymize any personal information of the plan authors. Our human evaluation is conducted by three graduate students who are co-authors of this paper. They participate in the evaluation process as part of their research contribution and are acknowledged through co-authorship.

Last, our approach relies on LLMs, which may produce inconsistent or biased results. While designed for daily task planning, this approach could potentially be misused for malicious intents. Future work should investigate these risks and develop additional safeguards.

Acknowledgments

This work was supported by NSF CAREER Award IIS-2338418.

References

Diaeddin Alarnaouti, George Baryannis, and Mauro Vallati. 2023. Reformulation techniques for automated planning: A systematic review. *Preprint*, arXiv:2301.10079.

Daman Arora and Subbarao Kambhampati. 2023. Learning and leveraging verifiers to improve planning capabilities of pre-trained language models. *Preprint*, arXiv:2305.17077.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. *Preprint*, arXiv:2211.09800.

M. Fox and D. Long. 2003. Pddl2.1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:61–124.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. 2023. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Preprint*, arXiv:2312.11970.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Preprint*, arXiv:2006.11239.

J. Hoffmann and B. Nebel. 2001. The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302.

Hanxu Hu, Hongyuan Lu, Huajian Zhang, Yun-Ze Song, Wai Lam, and Yue Zhang. 2024. Chain-of-symbol prompting elicits planning in large langauge models. *Preprint*, arXiv:2305.10276.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *Preprint*, arXiv:2201.07207.

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

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. 2024. Llms can't plan, but can help planning in llm-modulo frameworks. *Preprint*, arXiv:2402.01817.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. 2023. Imagic: Text-based real image editing with diffusion models. *Preprint*, arXiv:2210.09276.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *Preprint*, arXiv:1810.09305.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. 2023. Llm+p: Empowering large language models with optimal planning proficiency. *Preprint*, arXiv:2304.11477.

Yujie Lu, Pan Lu, Zhiyu Chen, Wanrong Zhu, Xin Eric Wang, and William Yang Wang. 2024. Multimodal procedural planning via dual text-image prompting. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10931–10954, Miami, Florida, USA. Association for Computational Linguistics.

James Oswald, Kavitha Srinivas, Harsha Kokel, Junkyu Lee, Michael Katz, and Shirin Sohrabi. 2024. Large language models as planning domain generators. *Preprint*, arXiv:2405.06650.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. *Preprint*, arXiv:2102.12092.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. *Preprint*, arXiv:2112.10752.

Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B Tenenbaum, Leslie Kaelbling, and Michael Katz. 2024. Generalized planning in pddl domains with pretrained large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20256–20264.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. *Preprint*, arXiv:2212.04088.

Tomáš Souček, Dima Damen, Michael Wray, Ivan Laptev, and Josef Sivic. 2024. Genhowto: Learning to generate actions and state transformations from instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6561–6571.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2023a. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. *Advances in Neural Information Processing Systems*, 36:38975–38987.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. 2023b. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Qingyun Wang, Manling Li, Hou Pong Chan, Lifu Huang, Julia Hockenmaier, Girish Chowdhary, and Heng Ji. 2023. Multimedia generative script learning for task planning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 986–1008, Toronto, Canada. Association for Computational Linguistics.

Te-Lin Wu, Alex Spangher, Pegah Alipoormolabashi, Marjorie Freedman, Ralph Weischedel, and Nanyun Peng. 2022. Understanding multimodal procedural knowledge by sequencing multimodal instructional manuals. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4525–4542, Dublin, Ireland. Association for Computational Linguistics.

Semih Yagcioglu, Aykut Erdem, Erkut Erdem, and Nazli Ikizler-Cinbis. 2018. Recipeqa: A challenge dataset for multimodal comprehension of cooking recipes. *Preprint*, arXiv:1809.00812.

Yue Yang, Artemis Panagopoulou, Qing Lyu, Li Zhang, Mark Yatskar, and Chris Callison-Burch. 2021. Visual goal-step inference using wikihow. *Preprint*, arXiv:2104.05845.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. *Preprint*, arXiv:2302.05543.

Ruijun Zhang, Xianda Guo, Wenzhao Zheng, Chenming Zhang, Kurt Keutzer, and Long Chen. 2024. Instruct large language models to drive like humans. *Preprint*, arXiv:2406.07296.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *Preprint*, arXiv:1904.09675.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2023. Large language models as commonsense knowledge for large-scale task planning. *Preprint*, arXiv:2305.14078.

A Dataset

A.1 Data Sources

	Instructables	wikiHow
# of tasks	100	1,000
# of task categories	1	10
Avg. # of steps per task	7.20	33.30
Avg. # of words per step	9.76	45.84

Table 4: Statistics of data collected from two sources.

To provide more insight into the two sources of our dataset, we further present both quantitative and qualitative comparisons between their plans. As discussed in Section 3, they exhibit differences in plan lengths and language styles. As Table 4 reports, the average length of wikiHow plans exceeds Instructable plans in both the number of steps per plan and the number of words per step.

Figure 6 demonstrates an intuitive distinction between two data sources. The left-hand side plan from Instructables is brief in textual descriptions while the right-hand side plan from wikiHow is elaborate with details including execution tips and potential outcomes. The accompanying images are also different in style.

A.2 Task Complexity Classification

Our task complexity classification is based on the relative informativeness of textual versus visual planning modalities, reflecting the inherent challenges of different procedural domains.

High complexity tasks This type of task typically involves intricate procedures that require delicate actions, precise object transitions, or spatial reasoning. For such tasks, visual plans significantly outperform textual descriptions in conveying critical procedural information. Figure 7 shows a task“make an origami pinwheel”, where visual demonstration of folding sequences and spatial relationships is essential for successful execution. In this case, OURS approach faces constraints imposed by current text-image-to-image model capabilities. Despite the fine-tuning, these models struggle to accurately depict subtle hand movements, precise folding sequences, and fine-grained spatial transformations essential for such a complex crafting task.

Medium complexity tasks This type of task often involves moderately challenging procedures where textual and visual plans provide comparable information. Figure 8 shows a task “make simple muffins with pancake mix”, where both modalities effectively convey the sequential steps and key procedural elements. OURS framework achieves optimal performance in such tasks where the procedural knowledge does not demand extreme precision or highly abstract concepts.

Low complexity tasks This type of task encompasses relatively abstract or conceptual procedures where textual plans are more informative than visual representations. Figure 9 shows a task “ask about application status following an interview”, where the procedural knowledge is primarily communicative and contextual rather than visually demonstrable. In this case, the underlying textual instructions are inherently abstract and focus on communicative strategies, timing, and contextual considerations. Therefore, the visual plan generated by OURS approach becomes less descriptive and struggles to represent non-concrete actions, resulting in generic and superficial representations.

B Experiment Settings

B.1 Prompt Design

In this section, we present our prompt design as a reference. Table 5 shows the prompts we use in all four stages of our framework with backbone Gemini-1.5-flash. We make only trivial adjustments for the other two backbone models and omit them for brevity and clarity.

B.2 Computational Costs

Our framework incurs computational costs from two primary sources: LLM inference during testing and one-time InstructPix2Pix fine-tuning. During inference, each task requires approximately 21 LLM calls distributed across plan drafting ($\sim$7 calls), visual information extraction ($\sim$7 calls), and iterative plan refinement ($\sim$7 calls), plus inference from our fine-tuned InstructPix2Pix model

How to make cranberry walnut bread pudding?

User Plan from Instructables:

Step 1: Melt and Cut
Melt your butter in the microwave
Cube your bread slices
preheat oven to 350 degrees F

Step 2: Mix
Poor the coconut milk into a large bowl
Add the apple sauce, sugar and vanilla paste, mix well

Step 3: Stir in Butter
Stir in the melted butter

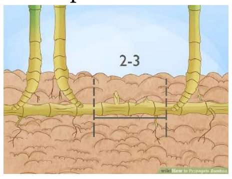
Step 4: Coat the Cubes
Add all the bread cubes to the milk mixture, stir to coat evenly

Step 5: Prepare Baking Dish
Pour the coated bread cubes into your baking dish

Step 6: Bake
Bake for 45 minutes, until the top in golden brown
It's best when served warm and topped with ice cream

How to grow new bamboo from rhizomes?

Expert Plan from wikiHow:

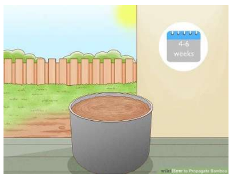


Step 1: Cut off a portion of the rhizome with 2-3 growth buds using a gardening knife.
Cut off a portion of rhizome with 2-3 growth buds using a gardening knife. Carefully remove the dirt away from the root system of your bamboo plant. Find a portion of the rhizome that has 2 or 3 growth buds, or the areas where stalks grow from. You may have to trim the stalks down to collect the rhizome. Use a sharp knife to remove the portion.

Step 2: Lay the rhizome horizontally in a pot with the buds facing up.
Have a layer of potting soil in the pot. Place the side where the stalks of bamboo grow face-up. If you left some of the stalk attached to the rhizome, keep those ends out of the soil.

Step 3: Cover the rhizome with 3 inches (7.6 cm) of potting soil.
Bury the rhizome so it can start to develop and grow. Press on the soil firmly so it has complete contact with the rhizome.

Step 4: Water the soil with a watering can.
The soil should be deeply moist, but there should not be any muddy water on the surface. Stick your finger into the soil down to the second knuckle to make sure that the soil is damp.

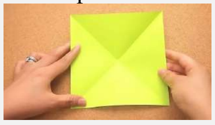


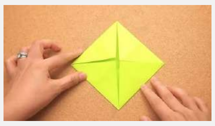
Step 5: Keep the pots in the shade for 4-6 weeks.
Keep the pot out of direct sunlight. The best place to keep it is next to a shady exterior wall or under the cover of a large tree. It will take 4 to 6 weeks before your bamboo sprouts and grows through the soil again.

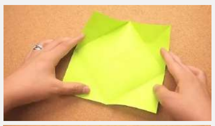
Figure 6: Two example plans sampled from Instructables data (left) and wikiHow data (right) respectively.

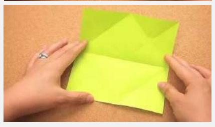
How to make an origami pinwheel?

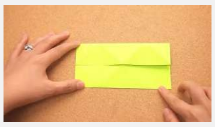
Expert Plan from wikiHow:

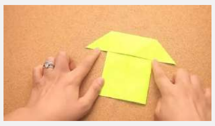

Step 1: Find a square piece of paper
If it's origami paper, start with the white side up. Fold it in half diagonally both ways. Then, unfold.

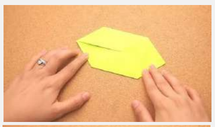

Step 2: Fold all four corners to the center, in the same manner as making a fortune teller
This is called a blintz fold.

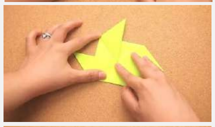

Step 3: Unfold

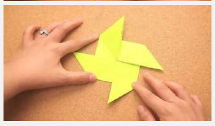

Step 4: Fold it in half
This is called a book or greeting card fold. Then, unfold.


Step 5: Fold each side to the center so that it looks like opening French doors
This is called a cupboard fold. Do not unfold this.


Step 6: Look at the diagonal creases at the very top
Pull them up and pinch them so they stick up. Then pull them down to the middle so that the shape appears as in the image.


Step 7: Turn the origami around so that the unfolded end is facing away from you and the folded end is facing you
Repeat step six on the unfolded side.

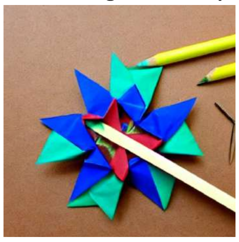

Step 8: Take the left flap on the top and fold it up, as shown in the image


Step 9: On the bottom portion, fold the right flap down
It's finished!

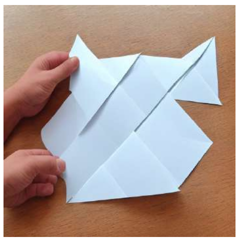
Step 10: Make use of your origami pinwheel

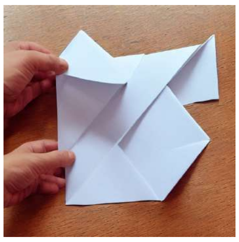
Step 11: Finished

Plan generated by Our approach:


Step 1: Gather Materials
Collect all the necessary materials for making an origami pinwheel. You will need a square piece of origami paper, a pencil, a ruler, a pair of scissors, a pushpin, and a small wooden stick or straw.

Step 2: Create Diagonal Folds
Take your square piece of origami paper and fold it in half diagonally to form a triangle. Make sure to crease the fold well. Then, unfold the paper and fold it diagonally in the opposite direction to form another triangle. Crease this fold well too.


Step 3: Cut Along the Diagonal Folds
Using a pair of scissors, carefully cut along each of the diagonal folds starting from the corners of the paper towards the center. Stop cutting about 1 inch (2.5 cm) away from the center point where the diagonal folds intersect.


Step 4: Fold and Secure the Flaps
Take one corner of each triangular flap and fold it towards the center of the paper. Make sure to fold the same corner of each flap (e.g., always fold the right corner). Once all four corners are folded towards the center, use a pushpin to secure them in place by piercing through all the layers at the center point.

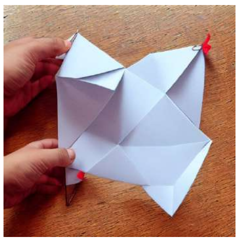

Step 5: Attach the Pinwheel to the Stick
Remove the pushpin from the center of the pinwheel while keeping the folded flaps in place. Then, use it to attach the pinwheel to the wooden stick or straw. Make sure to leave a small gap between the pinwheel and the stick so that the pinwheel can spin freely. Once attached, your origami pinwheel is complete and ready to spin!

Figure 7: Expert plan from wikiHow vs. plan generated by OURS approach for a high-complexity task *How to make an origami pinwheel.*

How to make simple muffins with pancake mix?

Expert Plan from Instructables:

Step 1: Preheat the oven to 180°C/350°F
Prepare a muffin pan with nonstick spray or cupcake cases.

Step 2: Add the pancake mix to a mixing bowl
Pour in the milk, eggs and maple syrup. Mix to combine.

Step 3: Transfer the batter to the muffin pan
Using a spoon or scoop, fill the muffin holes about two thirds each.

Step 4: Place in the oven
Bake for 13 to 15 minutes. The muffins are ready when a toothpick inserted comes out clean.

Step 5: Allow to cool
Serve fresh.

Plan generated by Our approach:

Step 1: Gather Ingredients and Supplies
Description: You will need pancake mix, milk, eggs, vegetable oil, and any optional add-ins like blueberries or chocolate chips. Additionally, ensure you have a mixing bowl, a whisk or spoon for mixing, a muffin tin, and muffin liners or non-stick spray.

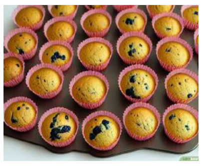
Step 2: Preheat the Oven
Preheat your oven to 375°F (190°C) to ensure it reaches the correct temperature by the time you are ready to bake the muffins.

Step 3: Prepare the Muffin Tin
Line the muffin tin with muffin liners or lightly spray it with non-stick spray to prevent the muffins from sticking. Ensure each cup is evenly coated to make it easier to remove the muffins once they are baked.

Step 4: Mix the Ingredients
In a mixing bowl, combine 2 cups of pancake mix, 1 cup of milk, 2 eggs, and 1/4 cup of vegetable oil. Stir until the batter is smooth and well combined. If you are using any optional add-ins like blueberries or chocolate chips, gently fold them into the batter.

Step 5: Fill the Muffin Tin
Spoon the batter into the prepared muffin tin, filling each cup about 2/3 full to allow room for the muffins to rise. Make sure to distribute the batter evenly to ensure uniform baking.

Step 6: Bake the Muffins
Place the muffin tin in the preheated oven and bake for 15-20 minutes. Once done, allow the muffins to cool in the tin for a few minutes before transferring them to a wire rack to cool completely.



Step 7: Cool the Muffins
Remove the muffin tin from the oven and let the muffins cool in the tin for about 5 minutes. Enjoy your muffins once they have cooled down to a comfortable temperature.

Figure 8: Expert plan from Instructables vs. plan generated by OURS approach for a medium-complexity task *How to make simple muffins with pancake mix*.

How to ask about application status following an interview?

Expert Plan from wikiHow:

I was calling to follow-up on the status of my application for the Managing Editor position
wikiHow

Step 1: Follow the interviewer's provided timeline before placing a call
Did you ask the interviewer to give you a specific timeline of when they'd get back to you? If so, let that entire time period pass before attempting to call them. If you don't have a set timeline to follow, wait at least 1 week before touching base. Don't call an interviewer earlier than this. If you call really early on, they might not have any updates—plus, you'll likely come off as pushy.

Step 2: Think about what you want to say before calling
Jot down a list of topics you'd like to discuss during your call so you aren't fumbling for things to chat about. Your script might include: a polite greeting; a brief intro; your main inquiry; any questions. Leave a voicemail where you mention your name, the position you applied for, and when you interviewed...

Have you reached a decision for the Managing Editor Position?

Step 3: Politely request an update
How you phrase your question can depend on different factors, like how many interviews you've had. If you've passed the final stage of interviews, a question like "Have you decided on [this position]?" can be a good way to go. If you just interviewed once, a more vague question like "Is my application still being considered for [this position]?" could be a better option...

Step 4: Highlight your qualifications for the position
A phone call can be a good opportunity to give a short run-down of why you're a good option for the position in question. You might highlight: Your experiences that directly apply to the company/job you applied for What you can offer to the company Why this opportunity is important to you

Step 5: Thank them for their time
The interviewer is taking time out of their day to follow-up and chat with you, so it's important to thank them for their continued time and consideration.

Plan generated by Our approach:

Step 1: Gather Necessary Information
Ensure you have all relevant details at hand, including the interviewer's name, the position you applied for, the date of the interview, and the contact number. Additionally, be prepared to describe the tool, object, action, and goal related to your application process. This preparation will help you make a professional and informed inquiry.

Step 2: Choose an Appropriate Time to Call
Select a time during regular business hours, ideally mid-morning or mid-afternoon, to ensure the interviewer is likely to be available and not too busy. Avoid calling during lunch hours or late in the day. If you are unsure about the best time, consider asking the interviewer at the end of your interview when they would prefer to be contacted for follow-up inquiries.

Step 3: Prepare a Polite and Professional Script
Draft a brief and courteous script to guide your conversation. This will help you stay focused and ensure you cover all necessary points. Having a script will help you communicate clearly and professionally.

Step 4: Make the Call
Dial the interviewer's contact number during the chosen appropriate time. Ensure you are in a quiet environment to avoid any background noise. When the interviewer answers, use your prepared script to inquire about the status of your application. Be polite, concise, and professional throughout the conversation. If the interviewer asks for any additional information or clarification, be ready to provide it promptly and accurately.

Figure 9: Expert plan from wikiHow vs. plan generated by OURS approach for a low-complexity task *How to ask about application status following an interview*.

Stage	Prompt
Draft Generation	You are a helpful planning assistant. Let's break down how to achieve the following goal step by step: GOAL: $\mathcal{G}$ PREVIOUS STEPS: $t_{1\ldots k-1}$ What is the next specific, actionable step toward achieving this goal? Please format your response as: STEP [k]: [Step Title] [step descriptions] After providing the step, indicate if the goal is achieved [YES/NO].
Image Generation	A clear, detailed photograph showing d_k, high quality, realistic with natural lighting
Visual Info Extraction	The provided image shows a step to $\mathcal{G}$. Please analyze it and extract the following information: 1. Objects: List salient objects in the image 2. Tools: Identify any tools, equipment, or instruments being used 3. Actions: Describe the specific actions being performed 4. Goal: Based on the visible actions and context, what appears to be the intended goal? Please format your response as: OBJECTS: [object list] TOOLS: [tool list] ACTIONS: [action list] GOAL: [state the apparent goal]
Text Refinement	You are a helpful planning assistant. Let's improve a step to $\mathcal{G}$ with visual information. Original Step: d_k Visual Information Extracted: v_k The improved step should: 1. Be more specific about the objects and tools involved 2. Provide clearer action descriptions 3. Maintain alignment with the overall goal Please format your response as: [improved step descriptions]

Table 5: Prompt templates we use in experiments with backbone model Gemini-1.5-flash.

for generating images at each planning step. The InstructPix2Pix fine-tuning represents a one-time computational investment, requiring 4 hours on 4x NVIDIA A100 80GB GPUs using 7,500 text-image-to-image pairs.

C Evaluation

Our comprehensive evaluation strategy employs three complementary assessment dimensions. This multi-faceted approach provides a complete picture of our framework's performance, addressing the inherent limitations of individual evaluation methods while leveraging their respective strengths. The convergent evidence across all three dimensions strengthens the validity of our findings.

C.1 LLM Evaluation

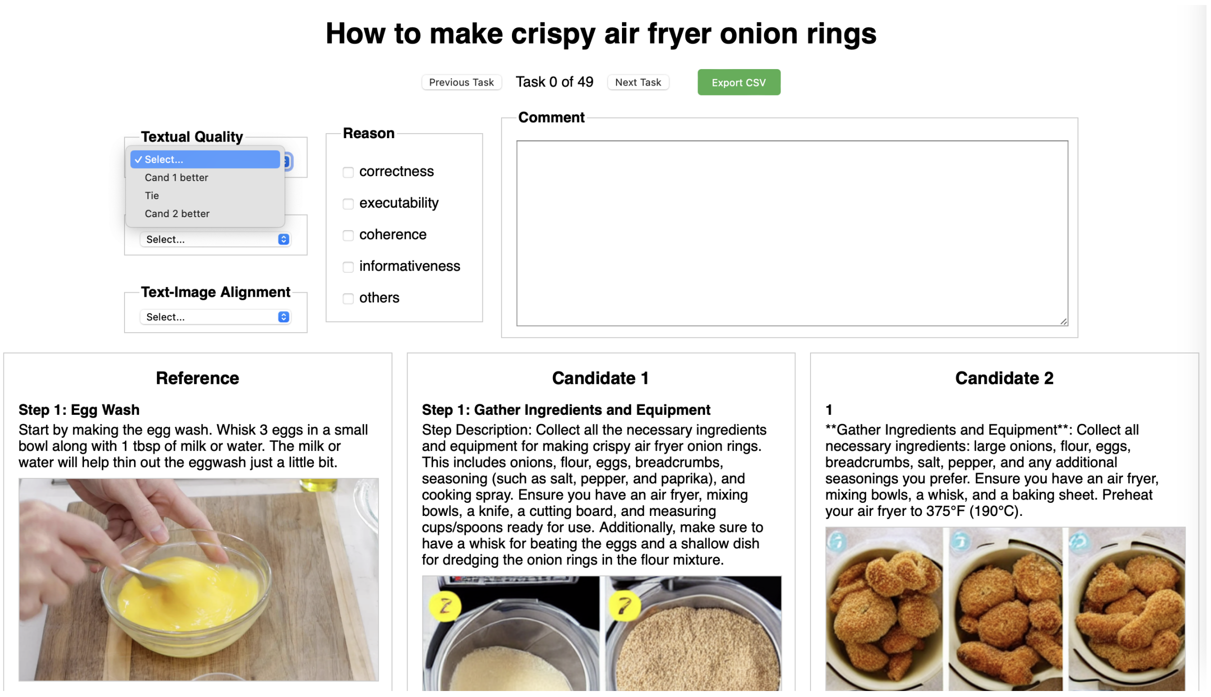
Table 6 presents prompts we use for LLM evaluation. Regarding textual plan evaluation, we define four aspects with inspirations from Huang et al. (2022)'s metrics design: correctness, executability, coherence, and informativeness. Their definitions are shown in Table 6. For visual plan evaluation, the evaluator Claude-3.5-Sonnet receives two images (i_{k-1}, i_k) in addition to the textual prompt. We prompt the MLLM to measure if i_k logically follows i_{k-1} considering the potential effect rendered by the step description t_k. For text-image alignment evaluation, the evaluator receives an image i_k. We prompt the MLLM to measure if i_k is semantically aligned with t_k. The grading criteria are elaborated in Table 6.

C.2 Human Evaluation

To facilitate human evaluation, we design an annotation tool as shown in Figure 10. In addition to two candidate plans, we also provide the reference article from the original data source (Instructables and wikiHow) in case annotators are unfamiliar with the task field. For every evaluation aspect, we provide three options: Candidate 1 better, Tie, and Candidate 2 better. To align with LLM evaluation, we list potential reasons annotators do not choose "Tie" for "Textual Quality". For example, if the annotator chooses "Candidate 1 better" when they evaluate textual plans, and they find its superiority over Candidate 2 is mainly in "coherence" and "informativeness", they are required to tick these two reasons. Furthermore, there is a text field for annotators to mark down their observations regarding any plan quality issues.

Evaluation	Prompt
Textual Plan	You are a helpful evaluation assistant. Please assess the following plan to $\mathcal{G}$ against the provided reference plan using these four criteria: 1. Correctness: Does the plan contain all necessary steps that align with the reference? This involves checking if the steps are complete. 2. Executability: How practical and actionable are the steps? This involves checking if they can be implemented in a real-world setting. 3. Coherence: Are all steps logically connected to each other? This involves checking if there are temporal conflicts or redundancy. 4. Informativeness: Does the plan provide sufficient detail? This involves checking if it provides enough information to understand the plan. Grading scale: 1-Poor 2-Fair 3-Good 4-Very Good 5-Excellent Reference Plan: [reference plan $\mathcal{R}$] Plan to Evaluate: [evaluated plan $\mathcal{P}$] Please provide a numeric score and a brief justification for each criterion.
Visual Plan	You are a helpful evaluation assistant. Please assess how well Image 2 continues from Image 1 considering the provided step description. Step description: t_k Grading Scale: 1-Poor: images appear unrelated or contradictory 2-Fair: slight logical connection but major inconsistencies 3-Good: clear connection but some inconsistencies 4-Very good: strong connection with minor inconsistencies 5-Excellent: perfect logical progression Please provide a numeric score and a brief justification.
Text-Image Alignment	You are a helpful evaluation assistant. Please evaluate how well the provided image aligns with the given step description. Step description: t_k Grading Scale: 1-Poor: image appears unrelated to the step 2-Fair: image partially reflects the step but has major mismatches 3-Good: image mostly reflects the step with some mismatches 4-Very good-mage clearly reflects the step with minor mismatches 5-Excellent: image perfectly represents the step Please provide a numeric score and a brief justification.

Table 6: Prompt templates we use for LLM evaluation.



Step 1: Egg Wash
Start by making the egg wash. Whisk 3 eggs in a small bowl along with 1 tbsp of milk or water. The milk or water will help thin out the eggwash just a little bit.

Step 1: Gather Ingredients and Equipment
Step Description: Collect all the necessary ingredients and equipment for making crispy air fryer onion rings. This includes onions, flour, eggs, breadcrumbs, seasoning (such as salt, pepper, and paprika), and cooking spray. Ensure you have an air fryer, mixing bowls, a knife, a cutting board, and measuring cups/spoons ready for use. Additionally, make sure to have a whisk for beating the eggs and a shallow dish for dredging the onion rings in the flour mixture.

1
Gather Ingredients and Equipment: Collect all necessary ingredients: large onions, flour, eggs, breadcrumbs, salt, pepper, and any additional seasonings you prefer. Ensure you have an air fryer, mixing bowls, a whisk, and a baking sheet. Preheat your air fryer to 375°F (190°C).

Figure 10: User interface of our designed annotation tool.