ForeAct: Steering Your VLA with Efficient Visual Foresight Planning —- Supplementary Materials

A. Appendix

A.1. Table of Contents

- Sec. A.2: We present additional qualitative foresight generation results, particularly in out-of-distribution scenarios to highlight the strong generalization of our model.
- Sec. A.3: The detailed descriptions of all our real-world tasks with visual examples.
- Sec. A.4: Definition of in-distribution and out-of-distribution tasks.
- Sec. A.5: Evaluation on different VLMs' performance as the high-level task planner.
- Sec. A.6: The prompt we use for the VLM model.
- Sec. A.7: A video demo.

A.2. Qualitative Foresight Generation Results

Figure 2 presents further qualitative results of our foresight image generation model. All scenarios shown are out-of-distribution, highlighting the strong capability and generalization of our model. This further demonstrates the effectiveness of our large-scale pretraining. We provide a detailed discussion of these scenarios below.

- **First Row**: While all the objects are present in our dataset, they have never been co-located within the same scene. Although the dataset contains bowls, we do not include any scenes in which objects are placed inside a bowl.
- **Second Row**: This scenario is similar to Pick_Veg; however, we introduce fruits (an apple and a banana) into the scene. Notably, neither apples nor bananas are present in our dataset.
- **Third Row**: There is already an apple on the plate. In the Pick_Veg task, however, we place only a single object onto the plate.
- **Fourth Row**: This scenario represents a completely unseen setting: all objects, including the new vegetables, fruits, and the glass bowl, are not present in our dataset.
- **Fifth Row**: This scenario is similar to Clean_Rubb; however, instead of placing rubbish into the bin, we place fruits into it.
- **Sixth Row**: This task is entirely novel and includes new objects such as a knife, a fork, and a paper cup. The paper cup has never been positioned on a plate in our dataset.

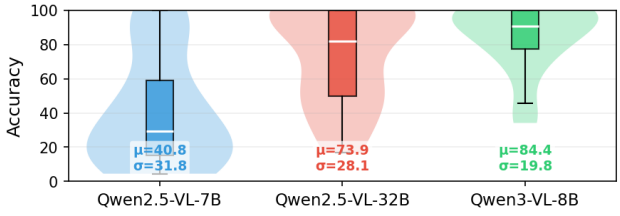


Figure 1. **Subtask Planning Accuracy across Different VLMs.**

A.3. Detailed Task Descriptions

In Table 2 and 3, we provide a comprehensive breakdown, with visual examples, of the 11 real-world robotic tasks used in our evaluation suite. To rigorously evaluate the models' performance, we intentionally introduce distributional shifts between training and inference in some of the testbenches, such as increasing the number of objects, randomizing initial poses, or introducing novel distractors. Please refer to the tables for a summary of the operational protocol and data distribution for each task.

A.4. Definition of In-Distribution and Out-Of-Distribution Tasks

The in-distribution setting corresponds to tasks and environments that match those seen during training, although the object layouts may differ. Since layout variations are also present in the training data, we consider these cases as in-distribution. The out-of-distribution setting involves changes to either the task or the environment. For example, the robot may be required to manipulate a different object within the same environment, such as picking the corn in the packing flower environment. For the in-distribution evaluation, we sample 50 observations from the test split of our dataset; for the out-of-distribution evaluation, we design and collect 50 new observations.

A.5. Evaluating Performance of VLMs

To assess the generalization capability of our planner, we employ a rigorous LLM-as-a-Judge protocol. Specifically, we utilize Gemini-3-pro-preview [**?**] to evaluate the semantic correctness of the VLM-predicted subtasks on our long-horizon real-world video benchmark. As illustrated in

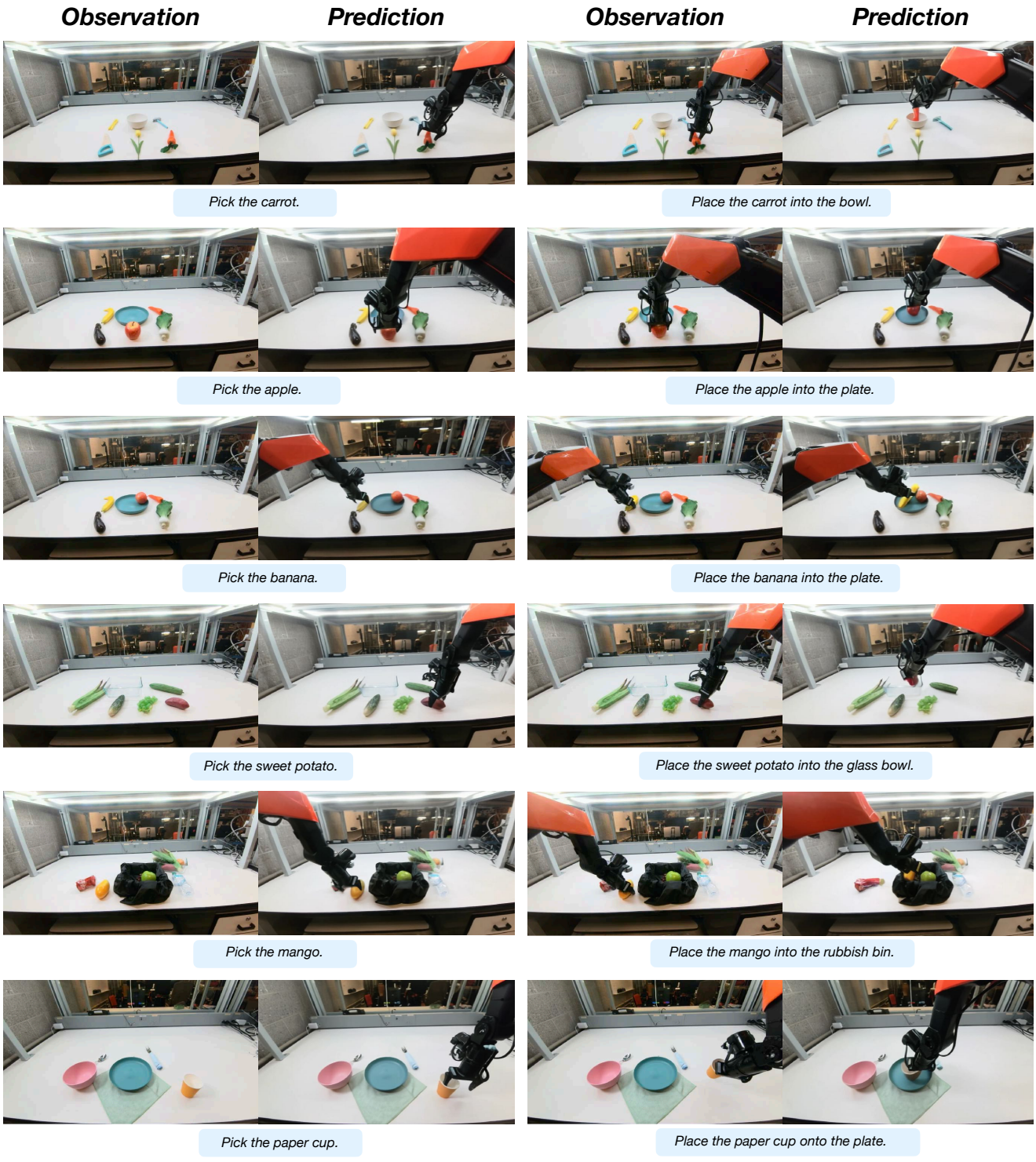


Figure 2. **Qualitative Foresight Image Generation Results.**

Figure 1, our framework demonstrates strong model scalability. Both Qwen2.5-VL-32B and Qwen3-VL-8B achieve competitive performance, effectively steering the robot through complex tasks. Conversely, the significant performance drop observed with the smaller Qwen2.5-VL-7B model validates the discriminative nature of our benchmark.

Table 1. **The prompt template used for the VLM planner.** The variable `{Task}` represents the high-level human instruction.

SYSTEM / INITIAL PROMPT ($t = 0$)

You are a robot controller. Please plan to finish the task in several steps. And give instruction for each step in a concise way.
The task is to "`{Task}`".

RULES:

- During the job, I will continuously give you an observation image of the current state.
- Based on the observation, please judge if the last instruction has been finished.
  - If yes, give me the instruction for the next step.
  - If no, repeat the instruction of the ongoing subtask.
- You're not required to describe the observation. Only output the instruction for each subtask.

Now, you are only required to output instruction for the first step.

VISUAL INPUT: `[Initial Observation Image]`

FOLLOW-UP PROMPT ($t > 0$)

VISUAL INPUT: `[Current Observation Image]`

Pay attention to the latest observation. Firstly, judge if the last instruction has been finished. Secondly, if yes, give me the instruction for the next step; if no, repeat the instruction of the ongoing subtask.
Your answer should be concise and deterministic.
Remember, your Overall Task is "`{Task}`".

A.6. VLM Prompt

To ensure reproducibility, we include the exact prompt templates we used for the VLM. The interaction is divided into two stages: the Initial Planning Phase ($t = 0$) and the Closed-Loop Monitoring Phase ($t > 0$). At $t = 0$, upon receiving the first observation, the system applies the *Initial Prompt* to specify the task and generate the first subtask. For all subsequent steps ($t > 0$), the system transitions to the *Follow-up Prompt*, where the VLM uses the latest observation to assess progress on the previous instruction and produce the next one. The full templates for both phases are provided in Table 1.

A.7. Demo Videos

To provide a more intuitive understanding of our system's real-world robustness, we include three video demonstrations corresponding to the Out-of-Distribution (OOD) scenarios as we mentioned in Section 4.3 of the main paper. These demos showcase how our Visual Foresight Planning framework successfully generalizes to unseen scenarios where baseline methods typically fail.

- **Demo 1: Complex Compositions.** The robot is tasked with picking multiple vegetables sequentially, despite being trained only on single-object manipulation.
- **Demo 2: Novel Objects.** The robot is tasked with picking fruits and placing them on a plate, whereas the training data only contained vegetable manipulation.
- **Demo 3: New Configurations.** The robot performs the Clean_Rubb task under unseen spatial layouts and background settings. This highlights the planner's robustness to novel environmental conditions and object positions.

All videos are available at https://github.com/mit-han-lab/foreact.

Table 2. **Overview of robot tasks and descriptions in our real-world dataset** (Part I).

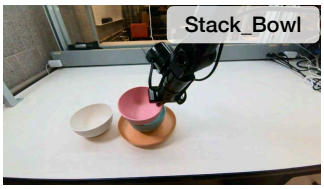
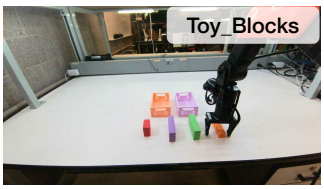
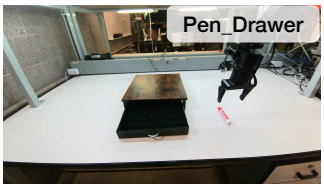
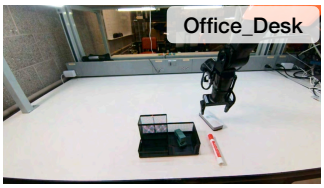
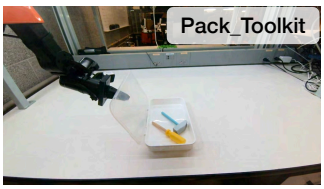
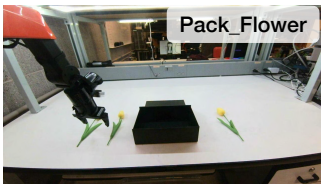
Task Example	Detailed Description
	**Protocol:** The robot must identify and pick a specific vegetable from a set of five and place it onto a plate. **Data:** Fine-tuning episodes exclusively feature single-object manipulation. Evaluation follows this setting unless otherwise specified (e.g., Fig. **??**).
	**Protocol:** The robot is required to pick a bowl and place it onto a matching color-coded plate. **Data:** Both training and inference setups involve a single bowl and three plates on the workspace. The presence of a color-matched plate is guaranteed in every episode.
	**Protocol:** The robot is required to pick up bowls one by one and stack them onto the plate. **Data:** Both training and inference involve stacking three bowls. During evaluation, the bowls can either be pre-placed on the desk or introduced at any time.
	**Protocol:** The robot is required to pick up rubbish items and dispose of them into a bin. **Data:** Items are randomly positioned on the workspace during both training and inference. Background settings are varied across episodes.
	**Protocol:** The robot sorts various toy blocks into boxes with matching colors. **Data:** The workspace contains 2 boxes and 4 blocks of different shapes and colors. While training episodes involve picking two blocks, evaluation requires sorting 2 to 3 blocks.
	**Protocol:** The robot picks up pens from the desk, places them inside a drawer, and then closes the drawer. **Data:** Training data exclusively involves manipulating a single pen. Evaluation requires the robot to place two pens into the drawer.
	**Protocol:** The robot is required to organize a cluttered desk by placing items into a designated organizer. **Data:** The task involves individual object placements, including a whiteboard eraser, a stapler, and a pen. The robot is required to sort all three objects in both training and evaluation.

Table 3. **Overview of robot tasks and descriptions in our real-world dataset** (Part II).

Task Example	Detailed Description
	**Protocol:** The robot identifies a specific tool and places it into a target container. **Data:** Both training and evaluation involve single-object manipulation. The task includes 5 distinct tools (hammer, hand saw, screwdriver, pliers, and wrench).
	**Protocol:** The robot distinguishes between N95 and surgical masks, sorting them into separate boxes. **Data:** Both training and evaluation involve two types of masks. The positions of masks are randomly initialized.
	**Protocol:** The robot retrieves designated tools, put them in the toolkit, and then closes the toolkit lid. **Data:** Both training and evaluation involve two target objects. Unseen objects were introduced during the evaluation to test zero-shot abilities.
	**Protocol:** The robot picks up flowers on the desk and place them into a box. **Data:** Both training and evaluation involve 2 to 3 flowers. Additional distractors, such as corn and grapes, are added to the workspace during evaluation.