A. Details on Experimental Setup

We conduct an evaluation on VIMA-BENCH [21], a benchmark suite for multimodal robot learning, which is built on the Ravens robot simulator [50]. VIMA-BENCH supports extensible collections of objects and textures to compose multi-modal prompts and to procedurally generate a large number of tasks. Specifically, each task instance defines a multimodal prompt that interleaves texts and images and the type of objects for manipulation.

A.1. Details of Benchmarks and Tasks

VIMA-BENCH contains various tasks ranging from simple object manipulation to multi-object manipulation and we select three representative long-horizon manipulation tasks in VIMA-BENCH- *visual rearrangement*, *visual constraints*, and *visual reasoning*, which represent different levels of complexity and requirements for embodied manipulation.

Visual Rearrangement Rearrange target objects in the workspace to match goal configuration shown in prompts. Note that to achieve the goal configuration, distractors may need to be moved away first.

- **Prompt:** `Rearrange objects to this setup {scene} and then restore.`
- **Description:** Objects in the scene placeholder `{scene}` are target objects to be manipulated and rearranged. In the workspace, the same target objects are spawned randomly, potentially with distractors randomly spawned as well. With a pre-defined distractor conflict rate, the position of each distractor has this probability to occupy the position of any target object such that the rearrangement can only succeed if moving away that distractor first. After reach the given goal configuration, and then within the allowed max steps restore all target objects to their initial configurations.
- **Success Criteria:** The configuration of target objects in the workspace matches that specified in the prompt, and then within the allowed max steps restore all target objects to their initial configurations.
- **Oracle Trajectory:** Shown in Fig. 7 with its multimodal prompt.

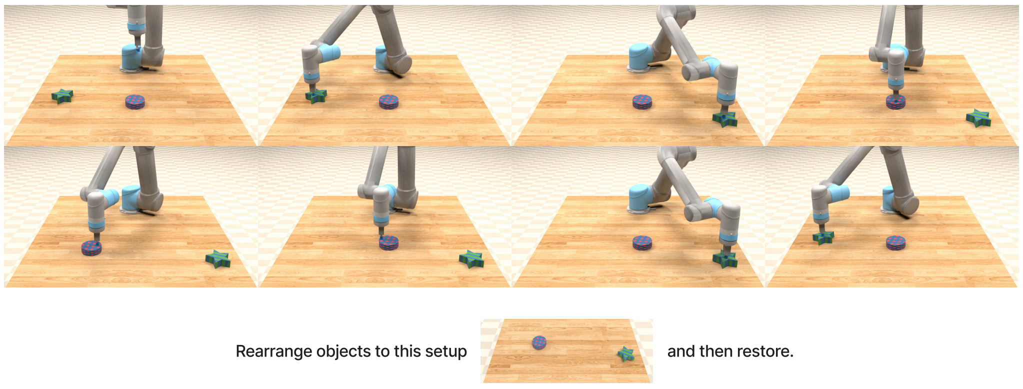

Figure 7. The example in Visual Rearrangement tasks.

Visual Reasoning Pick and place the target object specified in the prompt into different containers in order then restore to the initial container.

- **Prompt:** `Put` $\{$`object`$\}_1$ `into` $\{$`object`$\}_2$ `. Finally restore it into its original container.`
- **Description:** The object in the image placeholder $\{$`object`$\}_1$ is the target object to be manipulated across the task. There are more than one target containers (e.g. "`Put` $\{$`object`$\}_1$ `into` $\{$`object`$\}_2$ `then` $\{$`object`$\}_3$`. Finally restore it into its original container`" for two target containers to be placed in order). The rest of spawned containers naturally becomes distractors.
- **Success Criteria:** The target object is first put into multiple containers following the specific order. Finally it should be restored into its original container.
- **Oracle Trajectory:** Shown in Fig.8 with its multimodal prompt.

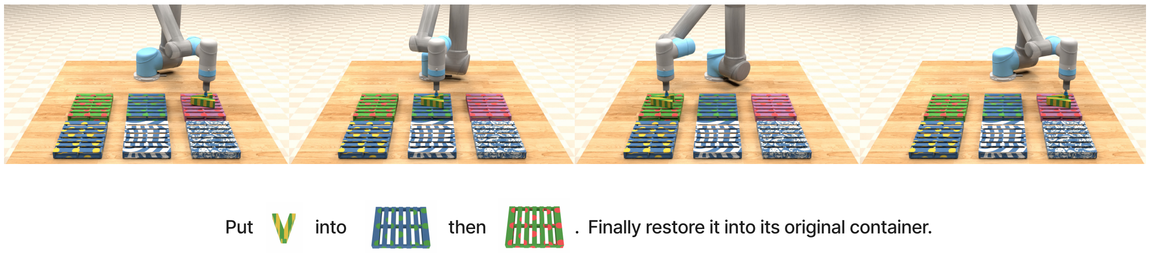

Figure 8. The example in Visual Reasoning tasks.

<u>**Visual Constraints**</u> Sweep the designated number of objects into a specified region without exceeding the boundary.

- **Prompt:** `Sweep {quantifier} {object} into {bounds} without exceeding {constraint}.`
- **Description:** `{object}` is the image placeholder of the target object to be swept spawned with a random amount in the workspace. Distractors have the same amount, same shape, but different color from target objects. `{quantifier}` is the text placeholder to determine the target quantity of objects to be wiped, sampled from `any`, `one`, `two`, `three`, and `all`. `{bounds}` is the image placeholder for a three-sided rectangle as the goal region. `{constraint}` is the constraint line.
- **Success Criteria:** The exact number of target objects to be swept are all inside the specified region. Potential failure cases include 1) any distractor being wiped into the region, 2) target object exceeding the constraint, or 3) incorrect number of target objects being swept into the goal region.
- **Oracle Trajectory:** Shown in Fig. 9 with its multimodal prompt.

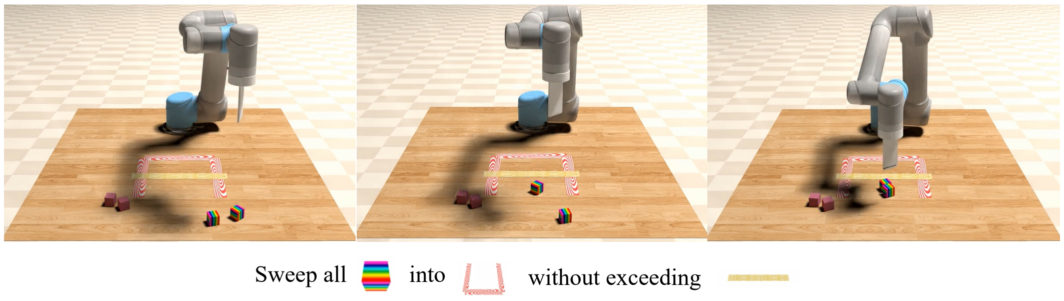

Figure 9. The example in Visual Constraints tasks.

A.2. Zero-shot Generalization Evaluation

Considering that some of the original evaluation levels showed minimal differences in test performance from VIMA-BENCH, we made slight consolidations and adjustments and introduced more challenging combinatorial tasks to test the combinatorial generalization. Specifically, we evaluate the generalization ability across three levels of increasing difficulty: placement generalization which randomizes the novel placement of objects (L1), object generalization which provides the objects with novel attributes (L2), and task combinatorial generalization which complexes the prompts with extra novel instructions (L3). Each level deviates more from the training distribution and is thus strictly harder than the previous one.

1. **Placement generalization.** All prompts are seen verbatim during training, but only the placement of objects on the tabletop is randomized at testing;
2. **Object generalization.** All textures and objects are seen during training, but new combinations of them appear in testing;
3. **Novel combinatorial task generalization.** Tasks with novel prompts and combined with additional manipulation demand at test time.

As Fig. 6 shows, CoTDiffusion exhibits strong zero-shot generalization on unfamiliar objects, colors, and shapes at different placements by explicitly visualizing them into coherent subgoals grounded on new concepts. This avoids re-grounding concepts from multi-modal prompts in the low-level foundation model for the placement and object. For L3 generalization, taking an example, we complex the prompt such as 'rearrange ... then rotate/twist/stack ...' to demand extra tasks such as rotating, twisting degrees, or stacking objects. CoTDiffusion achieves outstanding gain in the zero-shot performance of combinatorial tasks. When CoTDiffusion visualizes novel concepts into goal images, the foundation model can still accomplish the stack by simply achieving the provided subgoal, with no need to inherently understand novel skills like stack. This demonstrates the power of 'a image is worth a thousand words' - the subgoal images facilitate the generalization of the foundation model to unseen objects and novel combinatorial tasks while the foundation model simply achieves them with reuse of simple known primitives.

A.3. Training Dataset

VIMA-Bench can generate large quantities of expert trajectories via scripted oracle agents [1], and the ground truth keyframes that can be leveraged for the training of CoTDiffusion. For each task, we collect expert demonstrations with scripted oracle engine, including 10K oracle trajectories with annotated ground truth keyframe provided by VIMA-Bench. The keyframes serve as important supervision signals for CoTDiffusion training.

B. Pesudocodes of Framework

Algorithm 1 The training of CoTDiffusion for robotics manipulation.

```
Input: Training dataset $\mathcal{D}^{train}$ with pair-wised keyframes $\tau_x = \{\hat{x}_i\}_{i=1}^N$ and the action trajectories $\tau_a^i = \{\hat{a}_{i,t}\}_{t=1}^T$ and multi-modal prompt $\mathcal{P}$, multi-modal encoder and vision encoder $\mathcal{V}$, the semantic alignment module $\mathcal{S}$, conditional diffusion model $\mathcal{E}$ and foundation model $\mathcal{F}$ for action planning.

Pre-training Coarse Semantic Alignment Module
for each iteration do
    Sample the expert trajectories $\tau_x, \mathcal{P}$ from the $\mathcal{D}^{train}$ as mini-batch $\mathcal{B}$
    Calculate the adjacent keyframe $x_i \sim \tau_x$ contrasts reveal object manipulations as mask residuals patch $\hat{m}_i = x_{i+1} - x_i$ from the mini-batch $\mathcal{B}$
    Infer the semantic align tokens $z^i_{\text{align}}$ by the semantic alignment module $\mathcal{S}_\theta(x_0, x_i, \mathcal{P})$
    Predict the mask residual patch $\hat{m}_i$ with the additional mask decoder $\mathcal{D}_\omega(z^i_{\text{align}})$
    Update $\omega$, $\theta$ on mini-batch $\mathcal{B}$ according the following loss:
    $\mathcal{L}_{\theta,\omega} = -\mathbb{E}_{(\tau_x,\mathcal{P})\sim\mathcal{D}^{train}} || \sum_{i=1}^N \hat{m}_i - \mathcal{D}_\omega(m_i|z^i_{\text{align}})\mathcal{S}_\theta(z^i_{\text{align}}|x_0, x_i, \mathcal{P})||$
end for

Training Fine-grained Diffusion Model
for each iteration do
    Sample the expert trajectories $\tau_x, \mathcal{P}$ from the $\mathcal{D}^{train}$ as mini-batch $\mathcal{B}$
    Infer the semantic align tokens $z^i_{\text{align}}$ by the semantic alignment module $\mathcal{S}_\theta(x_0, x_i, \mathcal{P})$ and we discard the previously trained mask decoder $\mathcal{D}_\omega(z^i_{\text{align}})$, with $z^i_{\text{align}}$ as conditioned information guidance.
    Update the fine-grained conditioned diffusion model in a classifier-free manner according the following loss:
    $L_\phi = -\mathbb{E}_{(\tau_x,\mathcal{P})\sim\mathcal{D}^{train}}[\underbrace{\|\hat{x}_i - \mathcal{E}_\phi\left(x_{i-1}, z^i_{\text{align}}, \mathcal{P}\right)\|}_{\text{Forward Generation}} + \underbrace{\|\hat{x}_{i-1} - \mathcal{E}_\phi\left(x_i, z^i_{\text{align}}, \mathcal{P}\right)\|}_{\text{Backward Generation}}]$
end for

Training of Goal-conditioned Foundation Model
for each iteration do
    Sample the oracle action trajectories $\tau_a^i$ and the pair-wised subgoal images trajectories $\tau_x$ from the $\mathcal{D}^{train}$ as mini-batch $\mathcal{B}$
    Sample action sequence $\{a_{i,t}\}_{t=1}^T$ from the given subgoal image $x_i$ and the current observation $x_{i,t}$
    Update $\psi$ on mini-batch $\mathcal{B}$ according the following loss:
    $\mathcal{L}_\psi = -\mathbb{E}_{(\tau_x,\tau_a^i)\sim\mathcal{D}^{train}} || \sum_{t=1}^T \hat{a}_{i,t} - p_\psi(a_{i,t}|x_{i,t}, x_i)||$
end for
```

[1] https://github.com/vimalabs/VIMABench/tree/main/scripts/oracle

C. Model Architecture

In this section, we provide comprehensive details about CoTDiffusion model architecture as well as other adapted baseline methods.

C.1. Multi-modal Prompt Tokenization

Following VIMA-BENCH, there are three types of input formats in multi-modal prompts, namely (1) **text inputs**, (2) **images of full scenes**, and (3) **images of single objects**. Since VIMA has made considerable improvements mainly on tokenization and condition mechanisms for multi-modal prompts, for a fair comparison, we directly borrow the tokenization mechanism without any modification from VIMA. Specifically, For **text inputs**, we follow the standard pipeline in NLP to first tokenize raw language to discrete indices through pre-trained `t5-base` tokenizer and then obtain corresponding word tokens from the embedding look-up of the pre-trained `t5-base` model. For **images of full scenes**, we first parse the scene through a fine-tuned Mask R-CNN detection model [13] to extract individual objects. Each object representation contains a bounding box and a cropped image, captured by a bounding box encoder MLP to obtain the bounding box feature vector and ViT to obtain the image feature individually. Finally, an object token is obtained by concatenating the bounding box feature and the image feature and mapping to the embedding dimension. For **single object images**, we obtain tokens similarly, using a dummy bounding box. After obtaining a sequence of prompt tokens, we follow VIMA to pass it through a pre-trained `t5-base` encoder to obtain an encoded prompt and add adapter MLP between object tokens and the T5 encoder.

C.2. Semantic Alignment Module

The semantic alignment module consists of fusion modules which consist of several self-attention blocks and several cross-attention modules to track the progress and critically relies on extracting semantic information from the generated subgoal. Specifically, we use 3 cross-attention blocks with 6 causal attention layers and cross-attention heads as 8 and set the embedding dimension as 768. The fusion layer consists of 3 self-attention blocks with 8 heads. The pseudocodes of semantic alignment module can refer to Pseudocode 1. The aligned token encapsulates cues about 'which subgoal has reached now' and 'which subgoal should reach next', further injected into the diffusion model as precise semantic guidance to steer the generation of next subgoal. Through multi-stage cross-frame attention mechanisms, the triple alignment module can capture the cross-modal semantic correlation and achieve dynamic grounding of generated subgoal image into the multi-modal prompts to identify progress along the CoT reasoning chain.

```
def semantic_alignment_tokenizer(prompt_tokens, initial_obs_tokens, current_obs_tokens, pos_embd):
    # concat initial_obs_tokens and current_obs_tokens with prompt_tokens seperately and refined
    through fusion module which consists of several self-attention blocks
    initial_concat_tokens = concat([initial_obs_tokens, prompt_tokens]) + pos_embd
    initial_attn__tokens = causal_attn(q=initial_concat_tokens, v=initial_concat_tokens)

    current_concat_tokens = concat([current_obs_tokens, prompt_tokens])  + pos_embd
    current_attn__tokens = causal_attn(q=current_concat_tokens, v=current_concat_tokens)

    # peform cross-attention to capture a contrast representation
    contrast_tokens = cross_attn(q=initial_attn__tokens, kv=current_attn__tokens)

    # concatenate contrast tokens with prompt tokens into another fusion module to capture progress
    tokens
    contrast_concat_tokens = concat([contrast_tokens, prompt_tokens])
    progress_tokens = causal_attn(q=contrast_concat_tokens, kv=contrast_concat_tokens)

    # finally perform cross-attention between progress tokens and current obs tokens
    align_tokens = cross_attn(q=current_obs_tokens, kv=progress_tokens)

    # the last token is the inferred aligned token for mask decoder and diffusion generation
    predict_align_tokens = align_tokens[-1]
    return predict_align_tokens
```

Pseudocode 1. Semantic alignment module that compares current generated subgoal against initial observation and prompts, serving as an information bottleneck to capture the key semantic cues about the subgoal-prompt matching to locate the current chain stage and deliver to the diffusion model for next subgoal generation.

C.3. Observation Encoding

We follow the same procedure as in VIMA to obtain flattened object tokens from the RGB observations of full scenes. Since RGBs from both front and top-down views are provided, we order the object tokens by concatenating front view tokens following the order of $\left[\text{front, top-down}\right]$. We perform one-hot encoding of the end effector state and concatenate the encoded state with the object tokens. Finally, we transform the concatenated object-state representation into observation tokens. Additionally, we find that in complex control task scenarios, traditional abstract planners like VIMA have high demands for rich visual observations from diverse views. If the observation tokens are reduced to a single view, there will be a certain degree of performance decline, especially for long-horizon multi-object manipulation. Therefore, to demonstrate the robustness of limited observations of visual planners, we also designed a single-view version for comparison. Empirically, the performance with the front view tends to be slightly better than that of the top view.

D. Details of Baselines

D.1. Gato

Gato [41] introduces a decoder-only model that solves tasks from multiple domains including robotics, game, image captioning, language modeling, etc. Different tasks are specified by supplying the model with an initial sequence of corresponding tokens. For example, in tasks involving decision-making, these tokens include observation and action tokens. For a fair comparison, we provide the same conditioning as CoTDiffusion, including multi-modal tokenized prompts and the same observation image. We borrow the Gato [41] architecture from the modified implementation for multi-modal prompts [2] and use it for training a language conditioned policy with imitation learning. We use the 10k expert trajectories from VIMA-BENCH in each manipulation task for training, in an autoregressive manner to optimize the causal behavior cloning objective.

D.2. Flamingo

Flamingo [2] is a vision-language model that learns to generate textual completion in response to multi-modal prompts. It embeds a variable number of prompt images into a fixed number of tokens via the Perceiver Resampler module [19], and conditions the language decoder on encoded prompts by cross-attention. The original Flamingo does not interface with embodied agents. VIMA adapts it for manipulation tasks by replacing the output layer with robot action heads and using tokenized rollout histories as input during inference. For a fair comparison, we borrow the same adapted architecture used in VIMA [3] and train it end-to-end with causal behavior cloning loss, with the same 10k expert trajectories like Gato.

D.3. VIMA

VIMA policy [21] propose the VisuoMotor Attention (VIMA) agent to solve robot manipulation from multimodal prompts with a Transformer [47] Encoder-Decoder architecture. It encodes the multimodal prompts that interleave textual and visual tokens with a pre-trained language model. The autoregressive action decoding of VIMA is conditioned on the prompt embedding via cross-attention layers that alternate with the causal self-attention. Instead of directly operating on the raw RGB images, VIMA adopts the object-centric representation by cropping objects from both prompt and observation images and forming them as a sequence of object tokens with pixel coordinate information. For fair comparisons, we directly borrow the prompt tokenization mechanism from VIMA without making any improvements, illustrated in Appendix C.1. VIMA predicts each action dimension independently and trains its model via behavior cloning for an entire trajectory.

D.4. SuSIE

SuSIE[4], the concurrent work for visual planning in manipulation tasks on the CALVIN [31], incorporating pretrained image-editing model and the low-level goal-conditioned policy. SuSIE chooses Instruct Pix2Pix [6] as the pre-trained image-editing model and finetuned with the dataset of language-labeled video clips and robot trajectories from CALVIN. Then SuSIE can produce valid subgoals $_{\text{edited}}$ given an initial image $_{\text{orig}}$ and a language label l.

There are several key differences between SuSIE and CoTDiffusion. First, SuSIE utilizes a pre-trained image editing model, whereas CoTDiffusion learns a classifier-free diffusion model as a visual planner from scratch. However, we emphasize that CoTDiffusion provides a general framework that could easily incorporate other image-synthesis or image-editing techniques. Second, SuSIE lacks the capability to decompose general prompts and reason about complex multi-step instructions. It can only generate a single-step subgoal conditioned on the current state and short instructions. In contrast,

[2] https://github.com/vimalabs/VIMA/blob/main/vima/policy/vima_gato_policy.py
[3] https://github.com/vimalabs/VIMA/blob/main/vima/policy/vima_flamingo_policy.py

CoTDiffusion can logically decompose complex and general instructions into an ordered subgoal image sequence in a chain-of-thought manner.

To enable fair comparison, we finetune SuSIE on the same dataset as CoTDiffusion, following the intructPix2Pix approach. Specifically, we treat the current observation, instructions, and next subgoal keyframe as *curr*, *goals*, and *subgoals* inputs to SuSIE. We borrow the finetuning hyperparameters used for SuSIE on the CALVIN dataset and use the AdamW optimizer [26] with a learning rate of 1e-4, a linear warmup of 800 steps. We track an exponential moving average (EMA) of the model parameters with a decay rate of 0.999 and use the EMA parameters at test time. We train for 50k steps with a batch size of 64 on 8 v100 Nvidia GPUs, which takes 2 days. Moreover, SuSIE lacks intrinsic chain-of-thought reasoning capabilities, struggling to generate logically progressing subgoal sequences as coherent plans from general prompts. To enable comparison, we grant SuSIE the manually decomposed prompts into privileged step-wise sub-prompts which are unavailable in fair settings, denoted as ‘SuSIE + sub-prompts’.

E. Implementation Details

E.1. Details of High-level Diffusion Model

We utilize a classifier-free guidance diffusion model [16] for high-level visual planners, considering the open-vocabulary instruction in multi-modal. Our implementation references codebase of the classifier-free diffusion [4]. We initially attempted to finetune Stable Diffusion [43] directly on the keypoint images from VIMA-BENCH by denoising in the latent space. However, we found the observation resolution in VIMA-BENCH to be not high enough, leading to greater detail loss through the latent encoder and making effective denoising more difficult. The removal of fine-grained details through the latent space is critical for the manipulation subgoals, making the unsatisfactory subgoal images struggle to guide the low-level foundation model to plan action sequences. To this end, we choose to perform denoising in the original space instead of latent space and we train the classifier-free diffusion model from scratch. Specifically, we first train a coarse semantic alignment module for 10k steps with a batch size of 128, using 8 v100 Nvidia GPUs for training in parallel for 10 hours. We use the AdamW optimizer [26] with a learning rate of 2e-4. Then we train for 50k steps with a batch size of 32, using 16 v100 Nvidia GPUs for training in parallel for 40 hours. We use the AdamW optimizer [26] with a learning rate of 1e-4, a linear warmup of 1k steps, and weight decay of 0.01. We track an exponential moving average (EMA) of the model parameters with a decay rate of 0.999 and use the EMA parameters at test time. The strength of classifier-free guidance ω sets to 2.0, and we use the DDIM sampler [46] with 50 sampling steps.

E.2. Details of Multi-Modal Encoder

For the multi-modal encoder, we integrate the vision encoder and text encoder jointly. For text, we use t5-base as a text tokenizer and set embedding dimension as 768. For image, we use 4 ViT layers with 24 heads and a width of 768 to extract the image embedding. Then we use t5-base as pretrained language model to aggregate the object visual tokens and prompt text tokens. We also test the varying size of the pre-trained T5 encoder with t5-small(30M), t5-base(111M) and t5-large(368M). The differences among the variants are not significant, which is also mentioned in VIMA, for the potential reason that the prompt provided in VIMA-BENCH is not challenging enough.

E.3. Details of Low-level Foundation Model

The foundation model can be parameterized as an image-conditioned planner that infers the action $a_{i,t}$ given the current observation $x_{i,t}$ and the generated subgoal image g_i: $\tau_a^i = \{a_{i,t}\}_{t=1}^{T} \sim \prod_{t=1}^{T} p_\psi(a_{i,t}|x_{i,t}, g_i)$. For the low-level foundation model for action planning, we implement it as a simple 5-layer MLP to only possess basic single-object manipulation primitives. The input to the foundation model contains the generated subgoal image and the current observation and we utilize the same ViT as visual encoder used in the multimodal encoder to ensure consistent visual tokenization. We explore two options for aggregating the visual tokens of the subgoal image and current observation. The first simply concatenates the tokens and passes them through a self-attention block followed by several MLPs to predict the action tokens. The second option treats the subgoal and observation tokens as keys and queries into a cross-attention block for fusion before the MLP layers. Experiments show slightly better performance with the cross-attention approach, though overall differences are minor.

We use the AdamW optimizer [26] with a learning rate of 1e-4, a linear warmup of 500 steps, and weight decay of 0.01. We use 3 cross-attention blocks with cross-attention heads as 8 and set the embedding dimension as 768, the same as ViT. We train with a batch size of 64 for 5e5 steps on a single v100 Nvidia GPU, which takes 15 hours.

[4] https://github.com/lucidrains/classifier-free-guidance-pytorch

F. Additional Experiment Results

F.1. Horizon Length with Various Subgoal Numbers

Here we conducted an additional experiment by increasing the number of objects and the corresponding manipulation steps for rearrangement, reasoning, and constraints tasks to evaluate the success rate of different methods on more complex and longer-horizon manipulation tasks. The results in Fig. 10 demonstrate that CoTDiffusion enjoys better robustness for longer horizons compared to abstract planners, with the benefit from the explicit visual subgoals providing improved guidance for following complex instructions. SuSIE can also explicitly generate visual subgoals, but it lacks intrinsic chain-of-thought decomposition capacities for general complex prompts, limiting its support for complex long-horizon tasks. CoTDiffusion does not require explicit prompt decomposition and can produce subgoal images in an implicit chain-of-thought fashion.

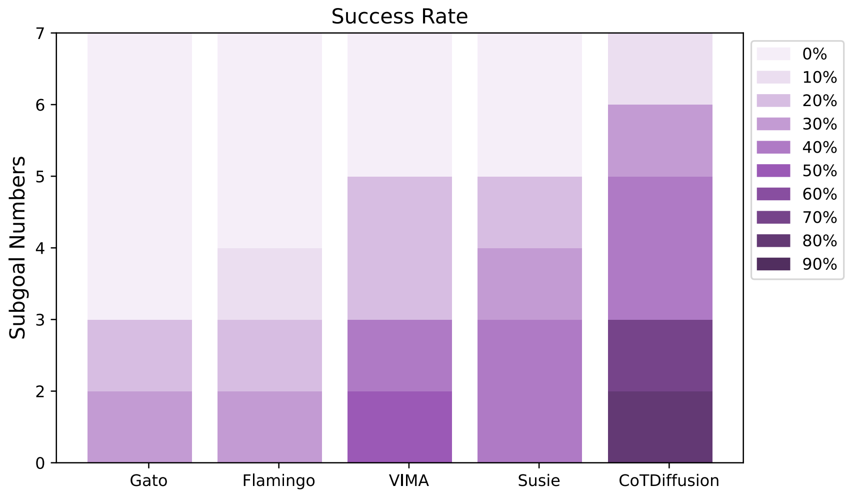

Figure 10. The evaluation of various methods on more complex manipulation task with longer horizon.

F.2. Model Capacity of Component Modules

We conducted additional ablation experiments on the model capacity of some key components, including the semantic alignment module, T5 tokenizer, and low-level foundation model. The results are shown in Tab. 4. The model size of T5 tokenizer has little impact, while the foundation model requires larger model capacity to ensure accurate execution and achievement of generated subgoals. A medium-sized model works best for the semantic alignment module. Based on the experiments, we chose T5-base, a 20M semantic alignment module, and a 40M foundation model as our final model configuration.

Model Size	**Rearrange**	**Reasoning**	**Constraints**	**Overall**
Semantic Alignment Module				
*2M	53.7 ± 1.4	48.4 ± 2.4	72.0 ± 4.5	58.0 ± 2.7
*4M	56.2 ± 1.8	50.3 ± 2.5	80.1 ± 4.3	62.2 ± 2.9
*20M	$\mathbf{59.0 \pm 1.7}$	51.7 ± 2.6	$\mathbf{83.1 \pm 4.7}$	$\mathbf{64.6 \pm 3.0}$
*40M	57.9 ± 1.3	$\mathbf{52.3 \pm 2.6}$	82.8 ± 4.9	64.3 ± 3.0
T5 Tokenizer				
*t5-small(30M)	57.5 ± 1.6	45.6 ± 2.5	78.4 ± 4.2	60.5 ± 2.8
*t5-base(111M)	$\mathbf{59.0 \pm 1.7}$	$\mathbf{51.7 \pm 2.6}$	83.1 ± 4.7	$\mathbf{64.6 \pm 3.0}$
*t5-large(368M)	58.6 ± 1.4	50.3 ± 2.4	$\mathbf{84.4 \pm 3.9}$	64.4 ± 2.7
Foundation Model				
*5M	50.2 ± 2.5	47.7 ± 3.2	70.3 ± 5.8	56.1 ± 3.8
*16M	56.9 ± 2.0	50.8 ± 2.5	82.0 ± 5.1	63.2 ± 3.2
*40M	$\mathbf{59.0 \pm 1.7}$	$\mathbf{51.7 \pm 2.6}$	$\mathbf{83.1 \pm 4.7}$	$\mathbf{64.6 \pm 3.0}$

Table 4. The evaluations on success rate of different model capacity of component modules.

F.3. More Complex environments

Additionally, we select long-horizon tasks from RLBench [20] and CALVIN [32] for further comparisons. For more detailed investigation for strong control policy, we add Act3D [12] and Diffusion Policy [7] into baselines. The evaluation resutls are shown in Tab. 5. For both Act3D and the language conditioned Diffusion Policy (DP) we modified, our visual subgoal chains and hierarchical framework demonstrate advantages.

Benchmark	**Act3D**	**LangCond-DP**	**Ours(+GCBC)**	**Ours(+DP)**
RLBench	$68.3\% \pm 2.9$	$59.4\% \pm 2.4$	$73.6\% \pm 5.1$	$\mathbf{81.0\% \pm 3.7}$
CALVIN	$38.1\% \pm 3.0$	$43.8\% \pm 4.3$	$49.7\% \pm 2.1$	$\mathbf{53.2\% \pm 1.6}$

Table 5. The evaluations of success rate on more challenging benchmarks.

F.4. Ablation Studies For Align Tokens

Additionally, we ablate the networks within the semantic alignment module by degrading align tokens to several intermediate tokens Fig. 2 and verify the effectiveness of designed triple-aligned network. The results are shown in Tab. 6.

Variant	**FID** $\downarrow$	**CLIPScore** $\uparrow$	**Succss Rate** $\uparrow$	**Generalization** $\uparrow$
CoTDiffusion	$\mathbf{10.2 \pm 0.8}$	$\mathbf{0.93 \pm 0.04}$	$\mathbf{64.6\% \pm 3.0}$	$\mathbf{51.1\% \pm 5.3}$
- $z_{\mathbf{contrast}}$	15.9 ± 2.0	0.84 ± 0.12	$41.4\% \pm 5.6$	$22.0\% \pm 4.6$
- $z_{\mathbf{progress}}$	17.8 ± 1.3	0.72 ± 0.18	$43.7\% \pm 5.1$	$17.9\% \pm 4.9$
- $z_{\mathbf{concat}}$	20.1 ± 2.4	0.75 ± 0.11	$28.3\% \pm 6.2$	$12.3\% \pm 4.6$

Table 6. The detailed ablation studies on different align tokens.

G. Qualitative Examples

Additionally, we would like to showcase some success examples as well as failure modes simultaneously.

G.1. Success Examples

Here we select VIMA as the representative method in abstract planner and compare it against CoTDiffusion on the visual rearrange in Fig. 11, visual reasoning in Fig. 12, visual constraints tasks in Fig. 13. As shown in the figures, for long-horizon tasks, VIMA directly maps complex instructions and prompts to actions in an end-to-end manner. This leads to difficulties in accurately following the instructions, while CoTDiffusion can generate intermediate subgoal images to guide the action planning process as visual planners. Without explicit subgoal images to correct accumulating deviations over longer action sequences, VIMA exhibits much lower success rates on complex long-horizon tasks. In contrast, by providing explicit visual milestones for guidance, CoTDiffusion enjoys enhanced instruction following and reasoning for accomplishing such temporally extended manipulation tasks. Inspired by the motto 'show, don't tell', our work aims to translate the general multi-modal prompts into coherent subgoal images instead of text step by step, serving as visual landmarks to enhance the instruction following in the long horizon.

We also tried providing some novel concepts not seen during training, such as 'rotate' or 'stack', as the L3 generalization experiments. The qualitative examples are illustrated in Fig. 14 and Fig. 16. CoTDiffusion can visualize novel concepts into goal images, the foundation model can still accomplish the stack by simply achieving the provided subgoal, with no need to inherently understand novel skills like stack. This highlights the advantage of using subgoal images over language descriptions - the subgoal images enable generalizing the foundation model to new objects and task concepts, while relying only on basic manipulation primitives to accomplish the visualized goals. The subgoal images bridge the gap between high-level comprehension and low-level executable actions, proving the power of 'a image is worth a thousand words' .

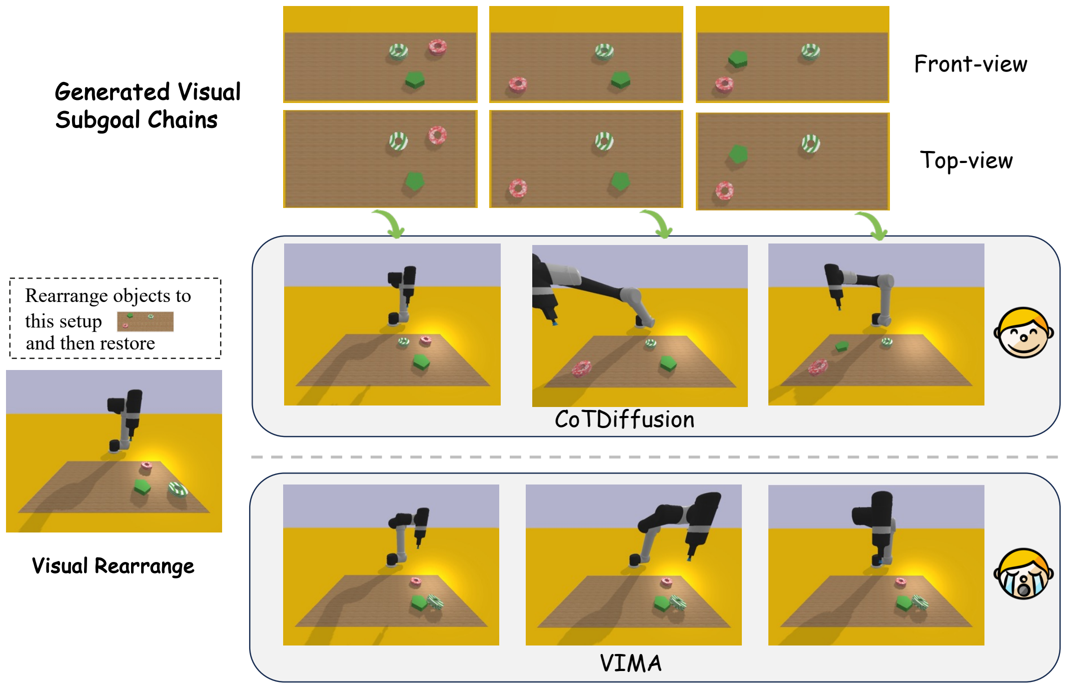
Generated Visual Subgoal Chains
Front-view
Top-view
Rearrange objects to this setup and then restore
CoTDiffusion
Visual Rearrange
VIMA

Figure 11. Visualization of CoTDiffusion and VIMA in visual rearrange long-horizon tasks with multi-modal prompts in VIMA-BENCH.

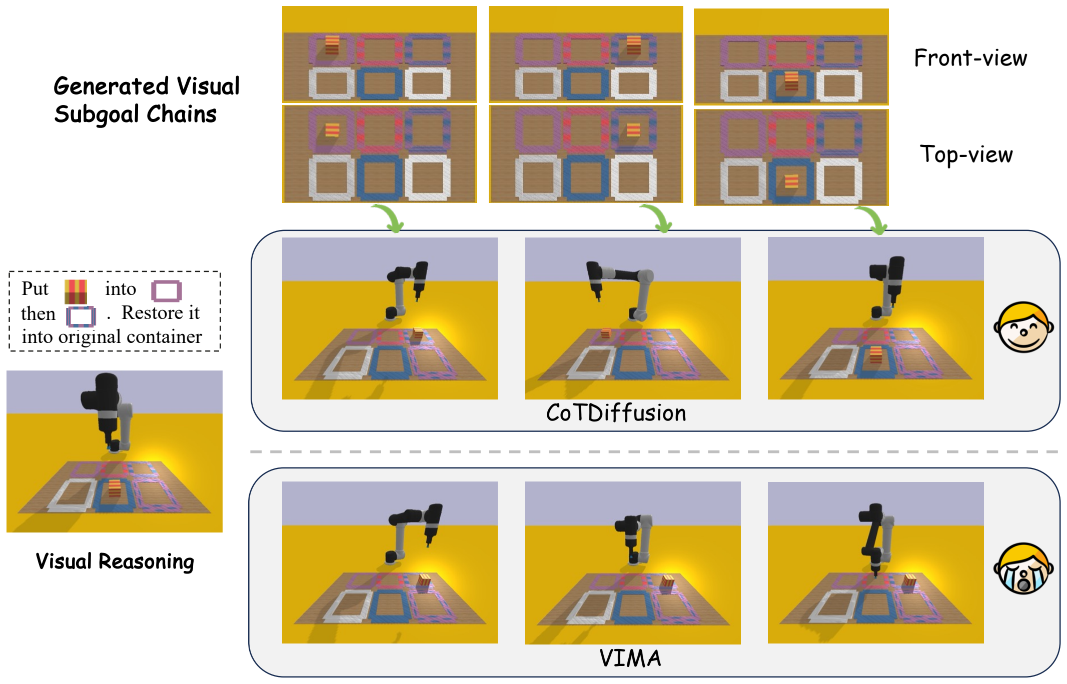
Generated Visual Subgoal Chains
Front-view
Top-view
Put into then . Restore it into original container
CoTDiffusion
Visual Reasoning
VIMA

Figure 12. Visualization of CoTDiffusion and VIMA in visual reasoning long-horizon tasks with multi-modal prompts in VIMA-BENCH.

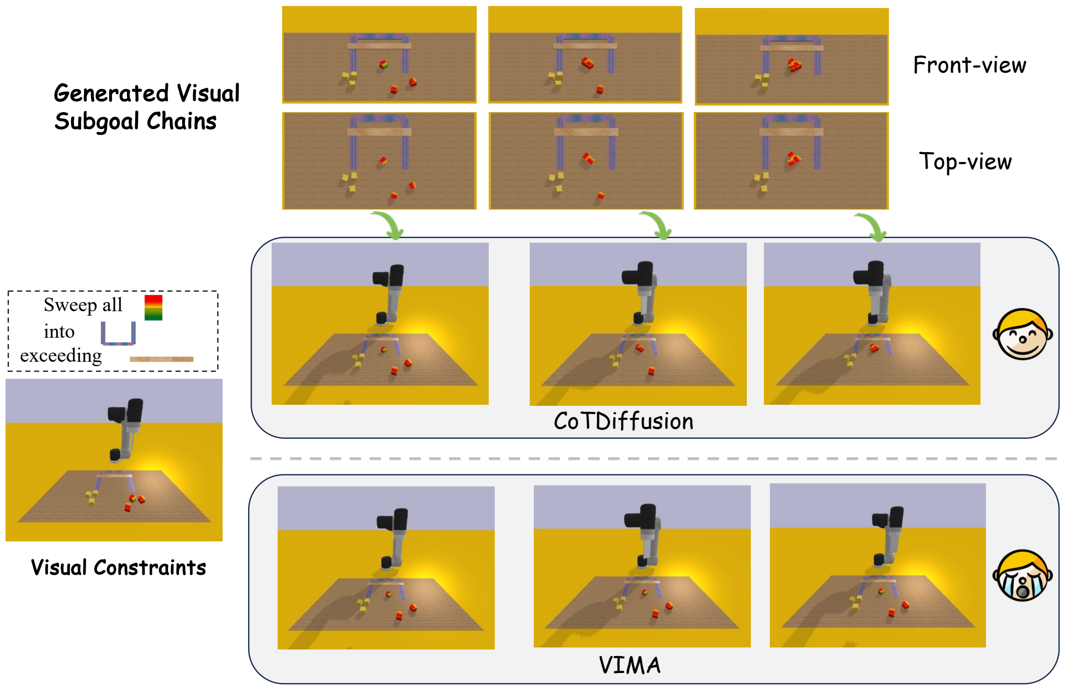

Figure 13. Visualization of CoTDiffusion and VIMA in visual constraints long-horizon tasks with multi-modal prompts in VIMA-BENCH.

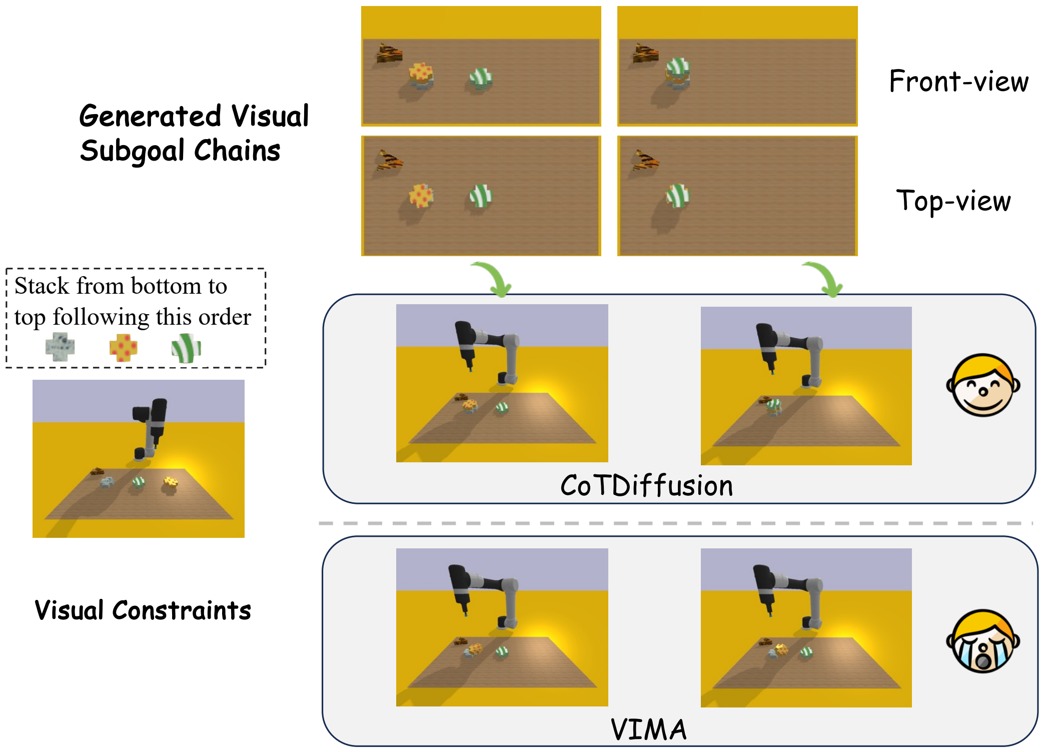
Generated Visual Subgoal Chains
Front-view
Top-view
Stack from bottom to top following this order
Visual Constraints
CoTDiffusion
VIMA

Figure 14. Visualization of CoTDiffusion and VIMA in novel generation tasks with novel concept with ‘stack’ in VIMA-BENCH.

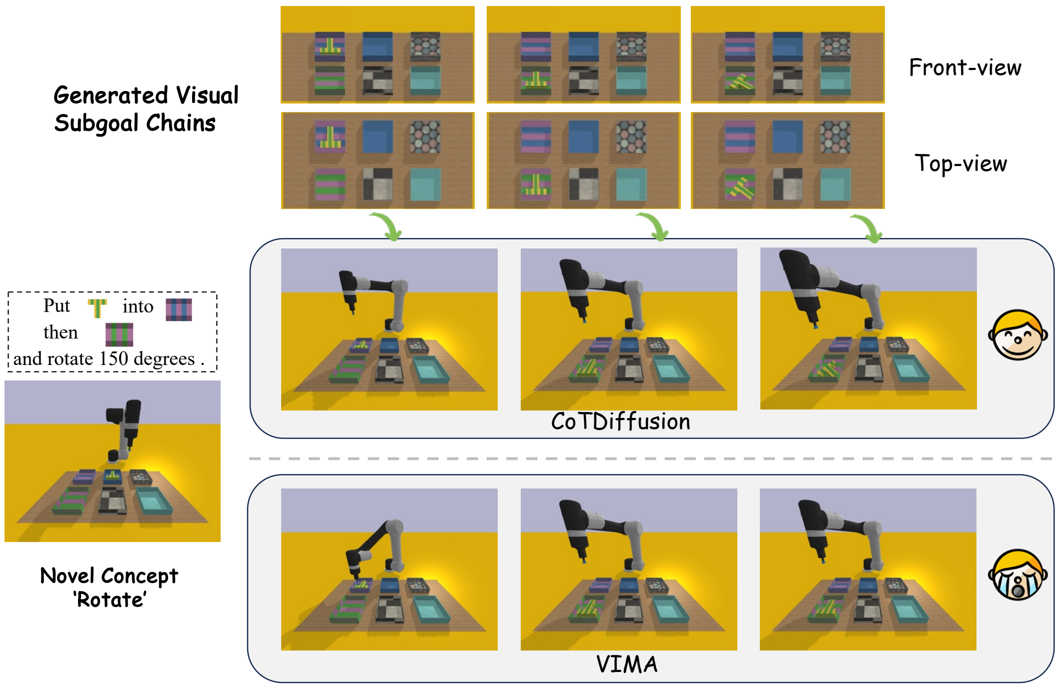
Generated Visual Subgoal Chains
Front-view
Top-view
Put into then and rotate 150 degrees .
Novel Concept 'Rotate'
CoTDiffusion
VIMA

Figure 15. Visualization of CoTDiffusion and VIMA in novel generation tasks with novel concept with 'rotate' in VIMA-BENCH.

G.2. Failure Analysis

We conduct more evaluations on CALVIN [32] and we design tasks like "put red box from slider to drawer" as occluded observation. The first subgoal generated is somewhat false as opened slider is empty, for no idea about red box in it. But with closed-loop manner, next generated subgoal explicitly appears the red box with actual observation for correction. CoTDiffusion generates next subgoal in closed-loop based on updated state and actual progress captured by triple alignment module, which means limited execution error can be implicitly considered and absorbed during iterations. However, failure still occurs when error is too large to cover, as future work.

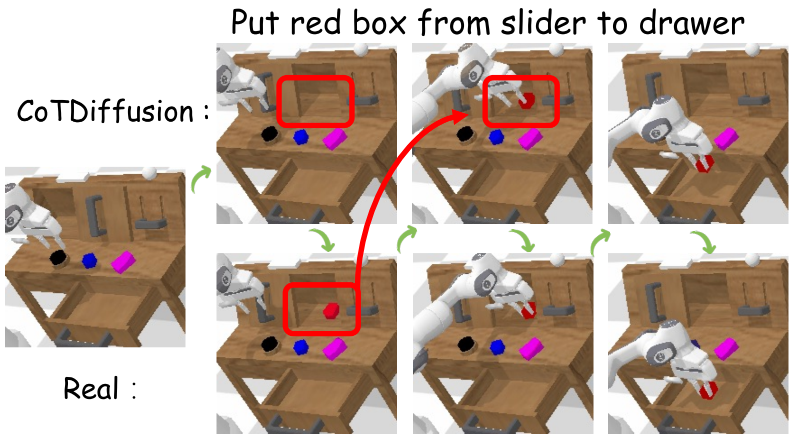

Figure 16. The failure case for generation in CALVIN and can be corrected by the closed-loop updated observations.

H. Limitation & Future Work

Although CoTDiffusion currently realizes implicit chain-of-thought visual reasoning via designed architecture, it still struggles with decomposing more complex instructions involving dozens of object manipulation subtasks. Additionally, the generation of reliable visual goals that can support cross-domain or cross-embodiment environments remains challenging. We believe incorporating commonsense knowledge from MLLMs like GPT-4V could empower CoTDiffusion with more generalizable and promising reasoning capabilities. Fine-tuning the framework on rationalized token sequences from such MLLMs provides a potential direction to enhance the visual planner into a more universally capable framework.

Another limitation is that visual planners like CoTDiffusion which generate subgoal images can suffer from slow sampling speeds for real robotics scenarios. Even using accelerated methods like DDIM [46], generating one subgoal image takes over ten seconds, which is difficult to tolerate for real-time manipulation. Going forward, we could consider more powerful diffusion sampling techniques like DPM++ solver [27]to achieve faster generation. Additionally, combining image inversion and conditional editing techniques during half of the denoising process could allow modifying existing subgoals instead of generating entirely new images from scratch each time. This incremental editing approach represents a promising direction to avoid costly full sampling and could help address the runtime limitations.

Overall, CoTDiffusion introduces a novel ‘generate subgoal images before act’ visual planning framework, equipped with chain-of-thought reasoning capacities for complex multi-modal prompts. Overall, our work paints a promising direction toward complex manipulation tasks by translating the general instruction into coherent subgoal images in a chain-of-thought manner as visual milestones to anchor the low-level foundation model execution. We believe the decoupled framework from visual planner and action planner highlights the motto ‘a subgoal image is worth a thousand words instruction’.