Planning as Inpainting: A Generative Framework for Realistic Embodied Path Planning

Cheng-Fu Yang[♠] **Haoyang Xu**[♠] **Te-Lin Wu**[♠] **Xiaofeng Gao**[♣] **Kai-Wei Chang**[♠] **Feng Gao**[♣]
[♠] University of California, Los Angeles [♣] Amazon
{cfyang, ericxu19, telinwu, kwchang}@cs.ucla.edu, {gxiaofen, fenggo}@amazon.com*

Abstract

Embodied Path planning, the process of determining optimal navigation paths or trajectories for robotic operations, is pivotal for the autonomy of robots in the wild. Generative methods have shown great promise in avoiding myopic decision by predicting the entire trajectory simultaneously. However, whether such type of method generalizes to more realistic settings — with high-dimensional state-space and potential partial observability, remains an open question. To address these problems, we propose a generative framework, "planning-as-inpainting", reconceptualizing path planning via utilizing the environmental map as a dynamic canvas to "inpaint" the predicted trajectories. This approach enables effectively leveraging the high-dimensional observations throughout the planning process due to its capability of: (1) precisely capturing the intricate environmental nuances, and (2) preserving the presented spatial relationships and physical constraints. To tackle the prevalent issue of model hallucinating future decisions when planning under partial observability, our framework integrates language conditioning mechanism. This mechanism is utilized to ground and infer the target position within the environment, increasing the accuracy and reliability of the plan. The proposed framework achieves promising performance across various embodied AI tasks, including vision-language navigation, object manipulation, and task planning in a realistic egocentric environment, highlighting its capability of handling the complexities of real-world scenarios.

1 Introduction

Recent advancements in embodied AI, spanning Computer Vision, Vision Language, Machine Learning, and Robotics, have significantly broadened the autonomy of robots to undertake diverse tasks in changing environments, such as precise object manipulation and multimodal navigation [1–7]. Central to this progress is path planning, which can be succinctly defined as the determination of navigation paths or the trajectories for a robotic arm's end effector. This critical process ensures the optimal routing or movement needed to accomplish specific objectives, thereby enhancing robot adaptability and interaction with complex environments.

Practically, path planning in embodied AI has been approached by learning a policy where the induced plans (trajectories) either maximize the return from the environment [8–19], or mimic expert behaviors [20–22, 7, 5]. In addition, modularized methods [23–25] combine the two aforementioned and other domain-specific modules to optimize performance.

On the other hand, the introduction of generative policy frameworks has ushered in a new era of innovation in path planning methodologies [26–28]. This body of work, including significant contributions from contemporary research [29–32], has successfully applied denoising diffusion

*This work is not related to these authors' position at Amazon.

Preprint. Under review.

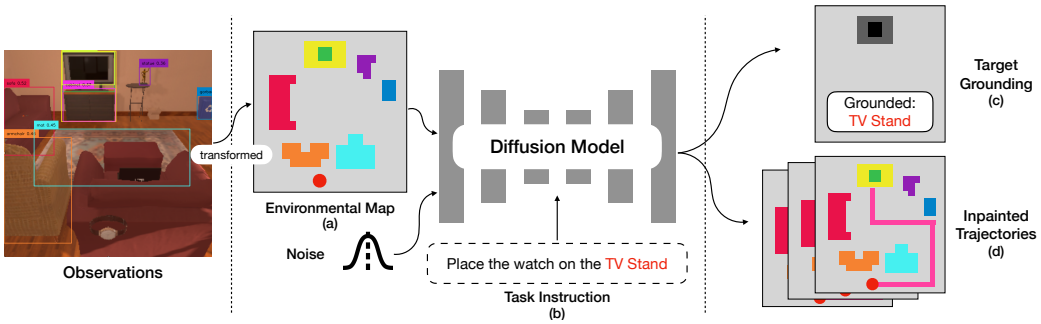

Figure 1: An illustration of our "planning as in-painting" framework. (a) The environmental map is constructed from high-dimensional observations, encapsulating both semantic meaning and spatial layout. (b) Task instructions are integrated with the generative model via cross attention. Our model is tasked to (c) ground the target object within the environment, or infer its approximate position when it is not visible. Subsequently, the model (d) 'inpaints' predicted trajectories onto the map, guided by these instructions and the observed data.

models [33] to focus on predicting the entire trajectories simultaneously. By leveraging a condensed vector representation of the environment coupled with cross-attention mechanisms, these strategies enable models to develop multimodal policies and address the challenge of myopic decision-making endemic to prior techniques.

Despite these innovations offering a broader perspective on path planning, they also pose a fundamental challenge in fully capturing the complex environmental and spatial dynamics present in realistic settings. The simplified environmental representations may not encapsulate the full spectrum of spatial and contextual details essential for accurate path planning. Furthermore, the issue of partial observability presents a persistent obstacle. For instance, during navigation tasks, a model might fail to accurately localize a target object within its initial position due to limited visibility or obstruction. As highlighted in the literature [27, 34], these methods can be susceptible to hallucinating future decisions under conditions of incomplete information.

In response, our work introduces a novel paradigm for path planning, conceptualizing it as an *image inpainting* challenge. As illustrated in Fig. 1, going beyond a single condensed vector and cross attention mechanism for environmental context, our strategy involves directly utilizing the environmental map as a canvas, where predicted trajectories are 'painted' in. This approach enables our model to effectively leverage the high-dimensional observations throughout the planning process. It boasts several advantages: firstly, it allows for a richer, more detailed representation of the surroundings, which captures the complexities and nuances of real-world environments more effectively. Secondly, by treating the environment as a canvas for inpainting, the model inherently considers the spatial relationships and physical constraints present in the environment.

Furthermore, the inpainting formulation intrinsically allows our model to incorporate language instruction clues into the environmental context via cross-modal attention. Specifically, we introduce a grounding objective within our model's planning phase, regularizing the model to localize the objects specified in the language instructions. Even if the objects are not observable, our model can still infer their approximate locations based on instructed waypoints or reference points. This significantly diminishes the risk of hallucinating the plan when predict with incomplete environmental information.

We evaluate our proposed method in three distinct simulated agent environments, including visual-language navigation, robotic arm object manipulation, and task planning in realistic egocentric environment. Our experimental results show the effectiveness of the proposed framework in all three evaluated environments, compared to previous methods including RL-based baselines, prior diffusion-based generative policy and modular approaches. Extensive ablation analysis further underscores the benefits of our "planning as in-painting" modeling and the grounding objective in achieving robust path planning.

In summary, the contribution of our work are three-folds:

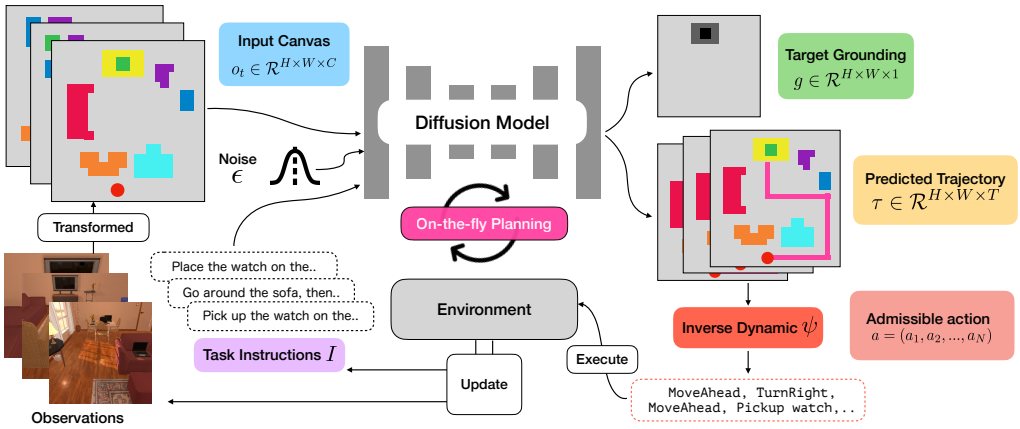

Figure 2: Architecture of our "planning as in-painting" framework. The process begins by transforming observations from the environment into task feature map o_t, which serves as the canvas for our inpainting algorithm. Along with the task instruction, the diffusion model grounds the target object referred in the instruction, and predicts the state trajectories τ. It is later translated into a sequence of actions a using the inverse dynamic module ψ. After executing a in the environment, the model gathers new observations and instructions, which are used to predict new trajectories. We refer this iterative process as "on-the-fly planning".

- We introduce "planning as in-painting", a novel path planning approach that utilizes the environmental map as a canvas, which enables our model to effectively leverage the high-dimensional environmental context throughout the planning process.
- Our method integrates language instruction clues into the environmental context, employing a grounding objective that substantially reduces hallucinations in scenarios with incomplete information.
- Through comprehensive experiments and analysis, we demonstrate the effectiveness of our proposed method across various simulated environments, highlighting its practical applicability and robustness.

2 Formulation and Method

In this section, we reveal the formulation and details of our inpainting framework and the grounding objective, which in combined is able to handle both fully and partially observable environments.

2.1 Notations.

We hereby define the notations that would be used throughout the rest of the paper: Given o which is the feature map transformed from agent observation, as well as the textual task instruction I, our model is tasked to generate the plan in the space of $\mathbf{x}$. This plan, $\mathbf{x}$, is composed of a trajectory, τ, and the grounding result of the targeted goal g. To clearly distinguish between the sequential timesteps inherent in task execution and those in the diffusion process, we utilize T to denote the timestep of the planning horizon, and k to represent the timestep within the diffusion process.

2.2 Path Planning as Image Inpainting

In order to generate reliable plans in complex and partially observable environments, it is crucial for our model to extract semantic and spatial context from the environment. In addition, the ability to leverage the linguistic cues is also imperative to complete the task. These requirements closely mirror the challenges in the image inpainting process, where model is tasked to generate images that are not only coherent with the existing canvas but also consistent with the given instructions. Inspired by the accomplishments of Denoising Diffusion Model (DDM) in the domain of image inpainting, we adapt a similar architectural paradigm to our planning context. Specifically, we formally define our model's objective to learn the conditional probability $p(\mathbf{x}|o, I)$, where $\mathbf{x}$ represents the output plan, o denotes the observed feature map from the agent's sensory data, and I embodies the task instruction provided in natural language, as illustrated in Fig. 2. We will now describe the input and output streams of our model in detail.

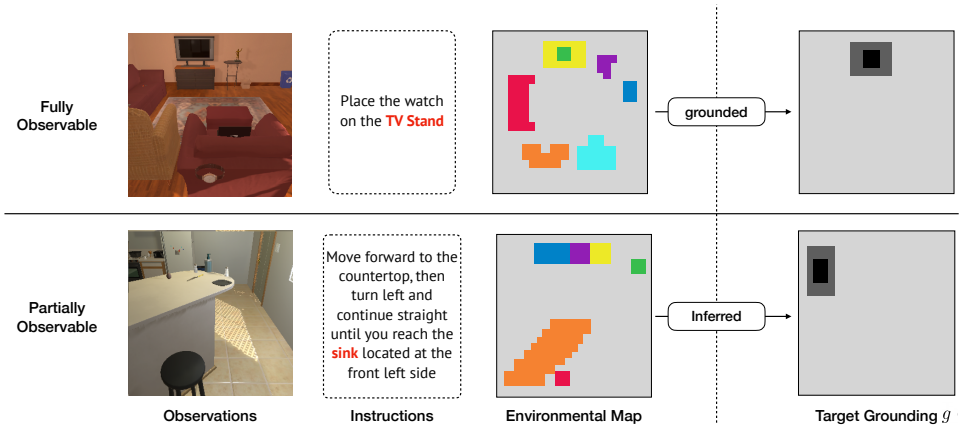

Figure 3: Illustration of the proposed grounding objective. In the fully observable environment, our model is tasked to directly *localize* the target object. While in the partial observable setting, this objective can still be enforced to guide our model to infer the approximate target position from the observed part and the instruction.

Input Canvas. In the practical implementation, capturing the intricacies of a real-world environment requires processing high-dimensional observational data. For instance, in navigation tasks, the observation o is initially obtained through techniques such as SLAM (Simultaneous Localization and Mapping) [35], which yields a 3D representation of the environment. Subsequently, this 3D data is processed, specifically by pooling along the z-dimension, to derive a 2D semantic map. This map is then utilized for navigation, where each pixel encodes the presence of objects detected within a specific grid area, facilitating a condensed yet informative representation of the environment for path planning. Accordingly, we define the input canvas at timestep t as $o_t \in \mathcal{R}^{H\times W\times C}$, where $H \times W$ is the dimension of the environment, and C indicates the channels conveying various types of environmental information. For example, C could represent the number of semantic classes or the dimension of visual features extracted from the field of view (FoV) observations by visual sensors.

Output planning space. As discussed in our previous subsection, the output plan $\mathbf{x}$ consists of both a trajectory τ and the grounded target g. We therefore define the output space as $\mathbf{x} = [\tau \oplus g] \in \mathcal{R}^{H\times W\times(T+1)}$. Here, $H \times W$ is delineates the environmental dimensions, while T pertains the temporal dimension of the trajectory τ, with $\tau = (s_{t+1}, s_{t+2}, ..., s_{t+T})$ being the trajectory of the predicted states starting from the next time step. An additional channel is allocated to encapsulate the grounded target information, which will be further elaborated in Sec. 2.3.

Execute the predicted plan. To execute the planned output, we translate the state sequence $\tau = (s_{t+1}, s_{t+2}, ..., s_{t+T})$ into admissible action sequences $a = (a_1, a_2, ..., a_N)$ using the inverse dynamics module ψ. Specifically, we have $a = \psi(s_t, s_{t+1})$. It is important to note that unlike prior models [27, 26], our model does not constrain state-to-action mapping to be one-to-one; in fact, we typically have $T < N$. This efficiency stems from the inpainting formulation, which enables our model to synthesize the plan within a space and dimension that mirrors the actual environment. As a result, this allows for encoding the spatial progression between different states into the same channel, significantly reducing the computation cost of the planning process. For example, during object manipulation, our model is able to forecast both the initial and final positions of the manipulator's end effector within the same channel.

2.3 Planning in Partially Observable Environment.

In real-world embodied tasks, the agent is often only expose to partial environment. In other words, the environment is partially observable. Thus the predicted plan tends to be unreliable with insufficient information [34]. In order to collect sufficient information for a more accurate plan, the planning algorithm often relies on exploratory probing to the environment, *e.g.*, random walking to *explore* an unseen environment, or attempting manipulating an occluded object in a trial and error manner. However, the stochacity of such exploratory phase hinders an efficient (and sometimes effective) solution of the plan and may suffer from catastrophic failures when the plan is deployed in the wild.

To alleviate such an issue, we introduce a grounding objective to localize the target objects referred in the task instruction I. This is visualized in Fig. 3, where our model is shown localizing the target object in a fully observable environment. Conversely, in a partially observable setting, the grounding objective is still applicable, guiding the model to infer the target's approximate location using available observations and instructions. This additional grounding step produces a variable $g \in \mathcal{R}^{H \times W \times 1}$ that represents the grounded target information. We therefore reformulate the objective of our model into:

$$p(\mathbf{x}|o_t, I) = p(\tau, g|o_t, I) = p(\tau|g, o_t, I)p(g|o_t, I) \tag{1}$$

We predict the joint distribution of the plan τ together with an estimation of the goal g given the current observation o_t and language instruction I. The right-hand side is actually a general formulation of decision making with partially observable environment. It provides a succinct theoretical justification to answer why the target grounding g could help improving the accuracy of the plan τ. Our experiments also show the effectiveness of incorporating g into the formulation.

2.4 Training and Inference

Training. We follow the supervised training paradigm used in DDM [33]. Concretely, each trajectory $\tau = (s_{t+1}, s_{t+2}, ..., s_{t+T}), \tau \in \mathcal{R}^{H \times W \times T}$ within the environment is paired with a corresponding task instruction I, task feature maps $o_t \in \mathcal{R}^{H \times W \times C}$, and goal state estimation $g \in \mathcal{R}^{H \times W}$. During training, we commence by sampling noise $\epsilon \sim \mathcal{N}(0, 1), \epsilon \in \mathcal{R}^{H \times W \times T}$ and a specific time step $k \sim \mathcal{U}(0, K)$. For simplicity, we rewrite $\mathbf{x} = (\tau, g)$. Subsequently, we construct noisy trajectories $x_k(\mathbf{x}) = \sqrt{\alpha_k}\mathbf{x} + \sqrt{1 - \alpha_k}\epsilon$ by systematically injecting noise, to introduce stochasticity that is essential for the diffusion model to learn the underlying denoising process. Our noise prediction model ϵ_θ is jointly conditioned on noisy input $x_k(\mathbf{x})$, task feature map o and the instruction I to predict the noise. By minimizing the discrepancy between the ground-truth noise ϵ and the predicted one, our model learns to reverse the diffusion process. Thus, our training objective can be written as:

$$\mathcal{L}(\theta) = \mathbb{E}_{q(\mathbf{x}_{1:K}, \boldsymbol{\epsilon}_{1:K}|\mathbf{x}_0)} \left[\sum_{k=1}^{K} \|\boldsymbol{\epsilon}_k - \epsilon_\theta(x_k(\mathbf{x}), k, o, I)\|^2 \right] \tag{2}$$

Nevertheless, unlike the common practice in conditional image generation, we choose not to take advantage of *Classifier Free Guidance (CFG)* for learning. Due to the nature of instruction-following, both the predicted trajectory and the goal estimation have to be conditioned on the instructions; otherwise, they are meaningless. We also found significant performance drop while using CFG.

On-the-fly planning. In most of the time, the accuracy of the generated plan generally improves as more information becomes observable. Consequently, we can apply our inpainting framework in a recurrent way, updating the plan as the task progresses. We refer to this iterative planning process as "on-the-fly planning", as illustrated in Fig. 2. Due to page limitations, the full algorithmic details are presented in the appendix.

3 Experiments

Our experiments are designed to answer the following research questions: (i) Can the proposed "planning as inpainting" formulation function effectively in realistic scenarios characterized by high-dimensional data and partial observability? (ii) Does our 'planning as inpainting' formulation outperform existing generative and reinforcement learning policies in path planning tasks, and if so, to what extent do the inpainting formulation and grounding objective contribute to this performance enhancement? (iii) How well does our method incorporate language instructions into the decision-making process, and does it demonstrate the capacity to generalize to novel task instructions?

Specifically, to answer question (i), begin by assessing our framework within the realistic, egocentric setting of the ALFRED benchmark **ALFRED** [3], a complex indoor environment featuring a **variety of objects and tasks** that require an agent to navigate and interact with its surroundings based on specific instructions. In response to research question (ii), for a thorough analysis and fair comparison, we perform experiments within the controlled setting of the **CompILE** environment [36]. This structured **multi-goal grid world** requires the model to interpret instructions and sequentially navigate to designated objects. To answer question (iii), we utilize the **Kuka Robot** [37], designed for **block-stacking and rearrangement** tasks. In this setting, we challenge our model with a range

Table 1: Experimental results of our method on ALFRED. Note that SR denotes task success rate, and GC denotes goal condition success rate. PLW indicates the path-length-weighted scores. PIP indicates our planning-as-inpainting framework.

Method	**Seen**				**Unseen**			
	SR	PLWSR	GC	PLWGC	SR	PLWSR	GC	PLWGC
EmBERT [38]	**37.4**	**28.8**	**44.6**	**36.4**	5.7	3.1	15.9	9.3
E.T. [5]	34.7	24.6	42.0	31.0	3.5	1.8	13.6	8.0
LACMA [7]	36.9	27.5	42.8	33.8	8.2	5.1	18.0	12.2
FILM [23]	25.4	10.3	38.8	14.8	20.4	8.1	32.7	12.2
FILM + PIP (**Ours**)	31.2	15.3	42.0	17.8	**26.1**	**10.0**	**39.7**	**12.5**
Improvements	↑ 5.8	↑ 5.0	↑ 3.2	↑ 3.0	↑ 5.7	↑ 1.9	↑ 7.0	↑ 0.3

of instructions varying in complexity to observe how effectively it incorporates these directives into its decision-making process.

In the following sections, we provide detailed explanations of the configurations for each environment, the corresponding evaluation metrics, the baseline models to compare with, and the experimental results of the proposed method. Model architecture and hyper-parameters are in the appendix.

3.1 ALFRED

Settings. To evaluate our framework's efficacy in realistic, complex settings, we utilize ALFRED [3]—a simulated household environment for instruction-following tasks. Here, an agent must comprehend language directives to locate and interact with specific objects. Each directive comprises two components: a navigation instruction to guide the agent towards the target, and a subsequent command specifying the object for interaction. Given that the target object is typically not visible from the starting position, the agent is required to infer from instruction cues to successfully locate and interact with target object. We assessed our methods and baselines on the validation set, distinguishing between *seen* and *unseen* environments. The *seen* category refers to environments that were included in the training data, whereas the *unseen* are those that the agent was not exposed to during training. Our evaluation on the *unseen* split offers a direct measure of our method's ability to generalize to new environments not encountered during training.

Baselines. For baselines, we consider end-to-end methods Embodied BERT [38], Episodic Transformer [5], and LACMA [7], which demonstrate great performance in the *seen* environment. Meanwhile, we consider modular methods like FILM [23], which show great generalizability in the *unseen* environment.

Evaluation Metrics. We consider task success rates (SR), goal-conditioned success rates (GC) and the path-length-weighted (PLW) for the metrics. SR quantifies the proportion of tasks in which the agent successfully completes all subgoals, whereas GC calculates the fraction of subgoals achieved at the end of the task, and PLW adjusts the success rate by the efficiency of the path taken, reflecting a balance between accuracy and task completion efficiency.

Results. Our empirical results on the ALFRED dataset validate the efficacy of the proposed "planning as inpainting" (PIP) formulation in environments characterized by high-dimensionality and partial observability. As depicted in Table 1, our FILM + PIP method shows substantial improvements across multiple metrics when compared to baseline models. In seen environments, the Success Rate (SR) and Path-Length Weighted Success Rate (PLWSR) have increased by 5.8 and 5.0 percentage points, respectively, while the Goal Conditioned (GC) metric and the Path-Length Weighted Goal Conditioned (PLWGC) show enhancements of 3.2 and 3.0 percentage points. Notably, in unseen environments, our approach demonstrates even more significant improvements: a 5.7 percentage point increase in SR and a notable 7.0 percentage point enhancement in GC. These increases suggest that our formulation not only copes effectively with the intricacies of high-dimensional data but also exhibits strong generalization capabilities in unseen settings.

Table 2: Experimental results of our method on CompILE. Note that SR denotes task success rate, and GD denotes goal distance. Path Precision, Recall, and F1 are metrics evaluating if model have followed the instruction faithfully.

Method	MO1G					P-MO2G				
	SR	GD ↓	Prec.	Rec.	F1	SR	GD ↓	Prec.	Rec.	F1
BC	23.20	6.49	4.42	66.55	8.29	19.19	8.27	4.84	74.09	82.61
CQL [39]	3.03	9.06	4.75	48.82	8.65	3.03	9.77	4.87	80.93	86.69
TD3 + BC [17]	40.2	2.83	15.60	83.35	26.28	0.01	6.87	8.31	21.01	11.91
IQL [19]	45.7	3.22	22.31	51.28	31.09	0.05	5.99	10.91	27.55	15.62
DT [18]	24.42	3.86	34.31	22.90	27.47	20.51	5.75	28.66	36.69	32.18
DP [29]	0.004	7.46	3.76	80.35	7.18	0.0	9.58	2.83	87.12	5.48
DD [27]	0.02	7.19	4.75	79.58	8.96	0.0	8.99	3.69	82.53	7.06
Ours	**99.50**	**0.03**	**99.63**	**99.42**	**99.52**	**99.90**	**0.002**	**93.34**	**96.48**	**94.88**
Ours - o	0.0	9.65	2.38	74.19	4.61	0.0	10.99	1.72	73.48	3.36
Ours - g	97.40	0.23	99.70	96.64	98.15	11.45	6.38	21.95	20.64	21.2

3.2 Multi-Goal Grid World: CompILE

Settings. We first consider a synthetically built grid-world environment, CompILE [36]. To further increase the problem's complexity, we add five more different objects and augment the map size to 16×16. The environment contains at most 10 obstacles that the agent cannot step on. In this grid map, an agent is required to navigate to multiple targets according to a natural language instruction. For each episode, the target objects are randomly initialized.

For quantitative analysis, we examine two different settings with increasing complexities: 1. Multi-object and one goal (**MO1G**): the environment contains up to 8 objects, and the agent's objective is to navigate to the object indicated by the instruction. 2. Partially observable (**P-MO2G**) setting: the target object is not visible to the agent at the beginning; however, the instruction will provide guidance on a reference object that the agent can first navigate to. This setup evaluates the model's capacity to deduce the accurate goal state and navigate to a target that is not visible initially. For example, *'First, go to the apple, then turn right and head towards the tomato'*, where the tomato is not initially visible.

Evaluation Metrics. We employ several metrics for evaluation: task success rate (SR); and goal distance (GD), measuring the Euclidean distance from the agent's final position to the target. Inspired by [40], to test if the agent has faithfully followed the instructions, we define the overlap between the predicted and ground-truth paths as true positives and the missed ground-truth path points as false negatives. False positives are defined as non-path predictions that are incorrectly marked as the path.

Baselines. We benchmark the proposed method against two families of methods: 1. RL policies: We consider behavioral cloning (BC), Conservative Q-Learning (CQL) [39], TD3+BC [17], Implicit Q-Learning (IQL) [19], Decision Transformer (DT) [18], and 2. Generative policies: Diffusion Policy (DP) [29], and the cutting-edge Decision Diffuser (denoted as DD) [27]. Due to page limits, we describe the implementation details of our method and baselines in the appendix section.

Results. The performances of both our method and the baselines are shown in Table 2. In comparison to RL policies, our model exhibits greater adherence to task instructions, reflected in markedly higher precision, recall, and F1 scores. Against generative policies, our utilization of environmental information through the planning-as-inpainting framework leads to significant performance boosts in all metrics. Moreover, while traditional RL and generative methods falter in partially observable settings, our model exhibits remarkable robustness, preserving high levels of accuracy amidst uncertainty.

Ablation Analysis. We present the performance of the ablated versions of our model in Table 2. We alter the observation space from a 2D semantic map to a 1D condensed vector in the '-o' version of the model. The significant performance drop highlights the importance of the inpainting formulation, which enables our model to effectively utilize environmental context for planning. On the other hand, while the removal of the grounding objective g appears to have a negligible impact in fully observable environments, there is a significant performance degradation in partially observable settings. This underscores the efficacy of our grounding objective, designed to mitigate the risk of hallucinated

Table 3: Experimental results of our method on Kuka. Note that all numbers represent the task success rate, which is the number of satisfied constraints divided by the number of specified constraints. We trained our method using either a one-hot vector or natural language instructions to represent the task constraints. Here, "SI" refers to tests conducted with a simple instruction set, whereas "CI" indicates the use of a complex instruction set for testing.

Method	**Single Constraint**		**Multi Constraint**	
	Stack	Rearrange	Stack	Rearrange
BC	4.00	20.00	0.0	0.0
CQL [39]	0.0	5.00	0.0	0.0
Diffuser [26]	45.60	58.90	-	-
DD [27]	58.00	62.70	60.30	67.20
Ours w/ one-hot	97.05	97.44	96.94	97.15
Ours w/ SI	**99.50**	**99.80**	**98.38**	**98.85**
Ours w/ CI	98.70	97.80	97.33	78.55

planning in environments with incomplete information. Due to space limit, we provide the additive ablation analysis in the appendix section.

3.3 Robotic Arm Block Stacking: Kuka

Settings. To investigate the efficacy of our "planning as in-painting" framework in manipulation tasks with constraints specified, we conduct additional experiments in the Kuka environment [37]. In this environment, the following two tasks are performed. 1. **Stacking**: assemble a tower consisting of four blocks. 2. **Rearrangement**: place one set of blocks on top of another. Additional constraints that specify the spatial relationship of the blocks are applied as well. For example, the constraints may require the red block needs to be on top of a blue block. The configurations may result in multiple structures, such as a single tower and two separate ones, presenting an out-of-distribution challenge for the model to interpret/execute the stack-place relations.

In addition, we perform additional analysis to assess the model's ability to understand and execute complex, language-based instructions and its capacity to adapt and generalize these instructions to novel scenarios. Specifically, our training involved two specific types of instructions: 1. "*Stack block1 on top of block2*" and 2. "*Stack block2 below block1*". For testing, we expanded the instruction set by including paraphrased versions generated by a Large Language Model (LLM), enriching the diversity of our test cases. Details of the exact instructions are provided in the appendix. This process yielded 10 distinct paraphrases for each original instruction, culminating in a comprehensive set of 20 varied command phrases for thorough evaluation.

Baselines. We compare our methods with CQL, BC, Diffuser [26], and Decision Diffuser [27] (DD). For Diffuser and DD, we directly report the number from their paper. For CQL and BC, we utilize one-hot vectors to represent the constraint. For our method, we have three different variants. The first one is conditioned on the same one-hot vector as the constraint, the second and the third one are trained with the linguistic constraints mentioned in the previous paragraph. They are differentiated by their testing conditions: the second variant is assessed using simple instruction constraints (SI), while the third is evaluated under complex instruction constraints (CI).

Results. From the results presented in Table 3, it is observed that Behavioral Cloning (BC) and Conservative Q-Learning (CQL) are unable to successfully complete the task due to their goal-specific nature. Specifically, they struggle to link varying objectives with appropriate planned trajectories. In contrast, our method and Decision Diffuser (DD) demonstrate significantly better adaptability across all test scenarios. Despite this, DD does not perform as well as our method. The key advantage of our in-painting framework lies in its ability to plan within a representation space that accurately reflects the real-world environment's spatial dimensions. This approach ensures the preservation of spatial context, which is essential for the successful execution of manipulation tasks.

Analysis on language condition. From Table 3 one can see that using language as condition results in the best version of the model, necessitates the integration of linguistic condition. Notably, our model demonstrates considerable resilience; even when tested against complex instructions that were *unseen* during the training phase, it sustains a performance level that is on par with the simpler

instruction set. This evidences the robustness of our model in integrating linguistic cues and adapting to increased complexity without a significant drop in success rates.

4 Conclusions

In conclusion, our "planning as in-painting" redefines path planning by treating the environmental map as a canvas for trajectory prediction. It leverages a richer representation of the environment and improves adaptability in realistic scenarios. By integrating linguistic instructions, our method reduces planning errors even with incomplete information. Our comprehensive testing across multiple environments showcases the robustness and effectiveness of our framework.

References

[1] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018.

[3] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[4] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems*, volume 34, pages 251–266, 2021.

[5] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952, 2021.

[6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[7] Cheng-Fu Yang, Yen-Chun Chen, Jianwei Yang, Xiyang Dai, Lu Yuan, Yu-Chiang Frank Wang, and Kai-Wei Chang. Lacma: Language-aligning contrastive learning with meta-actions for embodied instruction following. *arXiv preprint arXiv:2310.12344*, 2023.

[8] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

[9] Michael Janner, Karthik Narasimhan, and Regina Barzilay. Representation learning for grounded spatial reasoning. In *Transactions of the Association for Computational Linguistics*, volume 6, pages 49–61. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , 2018.

[10] John D Co-Reyes, Abhishek Gupta, Suvansh Sanjeev, Nick Altieri, Jacob Andreas, John DeNero, Pieter Abbeel, and Sergey Levine. Guiding policies with language via meta-learning. In *International Conference on Learning Representations*, 2019.

[11] Shao-Hua Sun, Te-Lin Wu, and Joseph J Lim. Program guided agent. In *International Conference on Learning Representations*, 2019.

[12] Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme q-learning: Maxent rl without entropy. *arXiv preprint arXiv:2301.02328*, 2023.

[13] Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36, 2024.

[14] Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:31278–31291, 2022.

[15] Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

[16] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

[17] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

[18] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[19] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[20] Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In *Advances in neural information processing systems*, volume 30, 2017.

[21] Ziyu Wang, Josh S Merel, Scott E Reed, Nando de Freitas, Gregory Wayne, and Nicolas Heess. Robust imitation of diverse behaviors. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[22] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *CVPR*, 2018.

[23] So Yeon Min, Devendra Singh Chaplot, Pradeep Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. Film: Following instructions in language with modular methods. *arXiv preprint arXiv:2110.07342*, 2021.

[24] Yuki Inoue and Hiroki Ohashi. Prompter: Utilizing large language model prompting for a data efficient embodied instruction following. *arXiv preprint arXiv:2211.03267*, 2022.

[25] Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. A persistent spatial semantic representation for high-level natural language instruction execution. In *Conference on Robot Learning*, pages 706–717. PMLR, 2022.

[26] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[27] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[28] Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

[29] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[30] Joao Carvalho, An T Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1916–1923. IEEE, 2023.

[31] Zhou Xian, Nikolaos Gkanatsios, Theophile Gervet, Tsung-Wei Ke, and Katerina Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[32] Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. *arXiv preprint arXiv:2302.01877*, 2023.

[33] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[34] Pedro A Ortega, Markus Kunesch, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Joel Veness, Jonas Buchli, Jonas Degrave, Bilal Piot, Julien Perolat, et al. Shaking the foundations: delusions in sequence models for interaction and control. *arXiv preprint arXiv:2110.10819*, 2021.

[35] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[36] Thomas Kipf, Yujia Li, Hanjun Dai, Vinicius Zambaldi, Alvaro Sanchez-Gonzalez, Edward Grefenstette, Pushmeet Kohli, and Peter Battaglia. Compile: Compositional imitation learning and execution. In *International Conference on Machine Learning (ICML)*, 2019.

[37] Günter Schreiber, Andreas Stemmer, and Rainer Bischoff. The fast research interface for the kuka lightweight robot. In *IEEE workshop on innovative robot control architectures for demanding (Research) applications how to modify and enhance commercial controllers (ICRA 2010)*, pages 15–21. Citeseer, 2010.

[38] Alessandro Suglia, Qiaozi Gao, Jesse Thomason, Govind Thattai, and Gaurav Sukhatme. Embodied bert: A transformer model for embodied, language-guided visual task completion. *arXiv preprint arXiv:2108.04927*, 2021.

[39] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1179–1191. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf`.

[40] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *ACL*, 2019.

[41] Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In *Conference on Robot Learning*, pages 767–782. PMLR, 2018.

[42] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, 2020.

[43] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2022.

[44] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4497–4506, 2021.

[45] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 104–120. Springer, 2020.

[46] Qiaozi Gao, Govind Thattai, Xiaofeng Gao, Suhaila Shakiah, Shreyas Pansare, Vasu Sharma, Gaurav Sukhatme, Hangjie Shi, Bofei Yang, Desheng Zheng, et al. Alexa arena: A user-centric interactive platform for embodied ai. *arXiv preprint arXiv:2303.01586*, 2023.

[47] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, 2020.

[48] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

[49] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444, 2022.

[50] Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

[51] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[52] Chuang Gan, Siyuan Zhou, Jeremy Schwartz, Seth Alter, Abhishek Bhandwaldar, Dan Gutfreund, Daniel LK Yamins, James J DiCarlo, Josh McDermott, Antonio Torralba, et al. The threedworld transport challenge: A visually guided task-and-motion planning benchmark for physically realistic embodied ai. *arXiv preprint arXiv:2103.14025*, 2021.

[53] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, et al. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975*, 2020.

[54] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

[55] Linxi Fan, Guanzhi Wang, De-An Huang, Zhiding Yu, Li Fei-Fei, Yuke Zhu, and Anima Anandkumar. Secant: Self-expert cloning for zero-shot generalization of visual policies. *arXiv preprint arXiv:2106.09678*, 2021.

[56] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning*, pages 4732–4741. PMLR, 2018.

[57] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6741–6749, 2019.

[58] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. In *NeurIPS*, 2021.

[59] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL-HLT*, 2019.

[60] Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. Babywalk: Going farther in vision-and-language navigation by taking baby steps. In *ACL*, 2020.

[61] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *EMNLP*, 2019.

[62] Yi Zhu, Yue Weng, Fengda Zhu, Xiaodan Liang, Qixiang Ye, Yutong Lu, and Jianbin Jiao. Self-motivated communication agent for real-world vision-dialog navigation. In *ICCV*, 2021.

[63] Raphael Schumann and Stefan Riezler. Analyzing generalization of vision and language navigation to unseen outdoor areas. In *ACL*, 2022.

[64] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[65] Wenlong Huang, Fei Xia, Dhruv Shah, Danny Driess, Andy Zeng, Yao Lu, Pete Florence, Igor Mordatch, Sergey Levine, Karol Hausman, et al. Grounded decoding: Guiding text generation with grounded models for robot control. *arXiv preprint arXiv:2303.00855*, 2023.

[66] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[67] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[68] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[69] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[70] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[71] Boyuan Chen, Fei Xia, Brian Ichter, Kanishka Rao, Keerthana Gopalakrishnan, Michael S Ryoo, Austin Stone, and Daniel Kappler. Open-vocabulary queryable scene representations for real world planning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11509–11522. IEEE, 2023.

[72] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[73] Nur Muhammad Mahi Shafiullah, Chris Paxton, Lerrel Pinto, Soumith Chintala, and Arthur Szlam. Clip-fields: Weakly supervised semantic fields for robotic memory. *arXiv preprint arXiv:2210.05663*, 2022.

[74] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.

[75] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[76] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

[77] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[78] Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems*, 35:31199–31212, 2022.

[79] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[80] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[81] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[82] Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Pathdreamer: A world model for indoor navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14738–14748, 2021.

[83] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[84] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[85] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[86] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[87] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

[88] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. *arXiv preprint arXiv:2310.07896*, 2023.

[89] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[90] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[91] Yilun Dai, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *arXiv preprint arXiv:2302.00111*, 2023.

[92] Russ Tedrake. Underactuated robotics. 2022. *URL https://underactuated. csail. mit. edu*, page 1.

[93] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[94] James A Sethian. Fast marching methods. *SIAM review*, 41(2):199–235, 1999.

A Related Work

Conventional Approaches in Embodied AI. In the realm of Embodied AI, several core tasks have attracted researchers' attention. These tasks typically include object manipulation [37, 41–44] and navigation [2, 36, 3, 45–47].

For object manipulation, the diversity of tasks requires agents to acquire a versatile skill set tailored to specific task requirements. Thus, specialized solutions have been developed for instruction following [48, 43], constraint satisfaction [49, 37, 50, 27], rearrangement [51–53], tabletop manipulation [54, 55, 43], and goal-conditioned planning [56]; each designed to equip agents with the capabilities needed to tackle the particular challenges posed by different scenarios.

Visual navigation, on the other hand, requires agents to navigate in complex environments using natural language instructions [57, 58, 40, 59–63]. Techniques such as sequence-to-sequence models [5, 38, 7] demonstrate strong navigation capability in the seen environment, yet their performance can falter in novel environments. On the other hand, modular approaches [23–25] learn to model the semantic mapping to mitigate the visual discrepancies across environments. However, the disjoint nature between the visual module and the language planning module can hinder the agents from executing complex instructions. Our proposed framework aims at bridging the efficiency of planning in the feature space and the strong distribution modeling capability of generative models, to devise a more efficient framework that better utilizes a given instruction.

Foundation Models in Embodied AI. The remarkable performance in both language (sequence) understanding and generation draws rapidly growing attention of Large Language Models (LLMs), even in the computer vision and robotics communities. Increasingly more recent works have integrated LLM into a variety of solutions for addressing embodied tasks, with a particular focus on reasoning and planning. Works such as [64–66] have successfully anchored the high-level semantic knowledge of LLMs into executable low-level actions. Meanwhile, numerous works [67–74] exploit Vision-and-Language Models (VLMs) to achieve either task or object generalization. Lastly, there are also abundant literature [75–78] that utilize LLMs for effective long-horizon planning. While these methods demonstrate proficiency in tasks within the well-explored environment, how these methods can cope with real-world partial observability, still remains an open research question. To this end, our method proactively addresses this environmental uncertainty by incorporating the belief prediction. This enables our model to infer the unobserved parts of the environment during test time and dynamically adjust its strategies, effectively handling the complexities of real-world environments.

Generative Methods in Embodied AI. To enhance agent's learning, a series of studies have suggested modeling the environment's dynamics with a world model, as seen in the contributions of [79–82]. Recently, diffusion models have shown remarkable capabilities in modeling and generating high-dimensional data, as evidenced by [33, 83–86]. Building on top of this, research works [29, 87–90, 32, 31, 30, 71] have leveraged diffusion models to deal with the inherent stochasticity in complex policy landscapes. Concurrently, certain studies [27, 91, 26] have shifted focus to modeling state trajectories, a method that accounts for the typically non-smooth and high frequency nature of action sequences, which poses a challenge for predictive modeling, as discussed in [92]. These methods have been effective within their respective domains. Our model extends these concepts, incorporating deep fusion methods inspired by text-to-image applications, as showcased by [86].

B Implementation Details

Generally, we use `UNet2DConditionModel` as the backbone of our diffusion model. For training, we set the batch size to 64 and the total training epoch to be 200. Our learning rate begins at 0 and linearly increases to 0.0002 over the first 5 epochs, after which it undergoes a cosine decay back to 0. For noise scheduling strategies, we utilize DDIM [93] and a cosine beta schedule where the noise level follows a squared cosine function with a capping mechanism. For both training and inference, we set the diffusion step to 100. We now detail task-specific parameters and implementations.

B.1 ALFRED

Ours. Planning in a complex and realistic environments is extremely challenging. As a result, following [23, 25], we first use an object detector to get the semantic map of the agent's field-of-view

(FoV) image, together with the depth map from the depth estimation model, we can reconstruct the environment in a 3D point cloud. Each point in the point cloud contains the semantic category of the object. By applying a max-pooling operation across the z-dimension, we derive a 2-dimensional semantic map. In this framework, each pixel of the map is directly associated with the object located at that point. Building on [23], we transform the 2D semantic map into a three-dimensional grid of dimensions $48 \times 48 \times 512$. Each cell within this grid contains a 512-dimensional textual descriptor for the object it corresponds to. These descriptors are derived using the `CLIPTextModel`, which extracts the word embeddings for the object labels.

For the ALFRED dataset, our model's input is designed with 514 channels. The first 512 channels represent the grid, while the remaining two channels introduce Gaussian noise that aids the generation process. The model outputs two channels representing the predicted object trajectories and the target locations.

Our UNet architecture consists of six layers: three downsampling layers (implemented as `CrossAttnDownBlock2D`, `CrossAttnDownBlock2D`, `DownBlock2D`) and three upsampling layers (implemented as `UpBlock2D`, `CrossAttnUpBlock2D`, `CrossAttnUpBlock2D`). The output channels for these layers are 256, 512, and 1024 in ascending order of depth. Language instructions are incorporated into the model by first converting them into textual embeddings of shape $L \times 512$, where L is the length of the instructions. To assimilate these text features into the model's representations, we map them from $L \times 512$ to $L \times 1024$ before applying cross-attention at this higher dimension.

In a departure from the Fast Marching Method (FMM) [94] utilized by [23], our novel planning-as-inpainting technique is employed to direct the agents' plan to complete their tasks.

B.2 Multi-Goal Grid World: CompILE

Ours. For the Multi-Object One Goal (**MO1G**) setting, the environmental representation involves converting a 16×16 grid into a corresponding $16 \times 16 \times 512$ representation grid. Within this grid, each cell hosts a 512-dimensional textual embedding that characterizes the object present. This architecture mirrors the one utilized in our ALFRED experiments, maintaining identical input/output channel configurations and UNet block structures. With multiple objects in the scene, textual instructions is leveraged to specify the goal, which are subsequently encoded into $L \times 512$ dimensional textual embeddings.

For partially-observable experiment (**P-MO2G**), the target objects are made invisible to the agent initially, but there would be a reference object that is visible to the agent. In the context of navigation, reference object often serves as landmark that the agent can reference to, which assists the agent in inferring the location of the target object. We adopt the same parameters and architecture as the one we have in the **MO1G** experiment.

Baselines. In detailing the implementation specifics for our baselines within the grid environment. We evaluate our method against two families of methods: 1. RL policies: We consider behavioral cloning (BC), Conservative Q-Learning (CQL) [39], TD3+BC [17], Implicit Q-Learning (IQL) [19], Decision Transformer (DT) [18], and 2. Generative policies: Diffusion Policy (DP) [29], and the cutting-edge Decision Diffuser (denoted as DD) [27]. For RL policies, we adhered to a simple yet effective architecture. This architecture comprises two convolutional layers followed by two fully connected layers. We represent the various objects within the grid as distinct integers to signify the object classes, simplifying the state representation. For our model's hyperparameters, we set the soft update parameter τ to 0.001 and the discount factor γ to 0.99, which balances immediate and future rewards. The network architecture features 16 channels for convolutional processes and a layer size of 40 for the fully connected layers, ensuring adequate complexity for pattern recognition within the environmental data. The epsilon-greedy strategy for action selection starts with an epsilon value of 1, annealing to a minimum of 0.01 over $10,000$ frames to balance exploration with exploitation. The replay buffer is set to a size of $300,000$ to maintain a comprehensive record of past experiences, and the learning rate for the model is configured at 0.001 to facilitate steady convergence during training. For generative policy, we tailor the grid representation to fit the original UNet-1D architecture by flattening it into a one-dimensional vector. For the training schedule parameters such as epochs, learning rate (LR), and diffusion steps, we adopt the configurations delineated in our established methodology.

Table 4: Experimental results of our method on the ALFRED test set. Note that SR denotes task success rate, and GC denotes goal condition success rate. PLW indicates the path-length-weighted scores. PIP indicates our planning-as-inpainting framework.

Method	**Seen**				**Unseen**			
	SR	PLWSR	GC	PLWGC	SR	PLWSR	GC	PLWGC
EmBERT [38]	31.8	23.4	39.2	31.3	7.5	3.6	16.3	10.4
E.T. [5]	28.9	20.1	36.3	27.8	4.7	2.6	14.9	8.3
LACMA [7]	**32.4**	**24.1**	**40.5**	**31.7**	9.2	5.8	20.1	13.5
FILM [23]	26.7	10.8	36.9	14.5	22.2	9.6	33.9	13.5
FILM + PIP (**Ours**)	28.7	12.9	38.6	16.1	**24.7**	**10.5**	**36.3**	**14.9**
Improvements	↑ 2.0	↑ 2.1	↑ 1.7	↑ 1.6	↑ 2.5	↑ 0.9	↑ 2.4	↑ 1.4

B.3 Robotic Arm Block Stacking: Kuka

Ours. For both stacking and rearrangement experiment. We use a template instruction to describe the final configuration of the cube. An example instruction looks like this: *Stack green cube on top of red cube. Stack blue cube on top of green cube. Stack yellow cube on top of blue cube.* Our diffusion planner is asked to model the position of the end effector of the robotic arm. We first convert the environment into a 2D grid, which contains the initial configuration of the cube. As a result, the input of our diffusion model contains 3 channels. The first one is the location of each cube, followed by 2 noisy channels. The first output channel is the trajectories of the end effector, and the second channel is the estimated location of the cube or target placement location.

Baselines. Similarly, for RL policies like Behavior Cloning (BC) and Conservative Q-Learning (CQL), we have the same parameters as the one we have in the CompILE experiment. We first decompose the stacking task into moving a single cube once. The reward function is designed as the distance to the target position, which initially is the location of the cube to pick, and the location to place the cube (which is often on top of another cube). For generative policies like Diffuser [26] and Decision Diffuser (DD) [27] we choose to directly report the number from their paper.

C ALFRED Test Set Result

Since our method is built upon FILM [23], it's essential to showcase the advancements achieved by our planning model over the baseline. However, as reported in the repository of the ALFRED [3] dataset, the simulator of the environment could have machine-dependent behavior, which introduces randomness in the reproducibility of the method. To mitigate this, we conducted five evaluation runs of FILM on our system and report the highest score in Table 4. Our model demonstrates a notable improvement in generalization to unseen environments. As indicated in the table, our method achieves a Success Rate (SR) of 24.7% and a Goal Condition (GC) score of 36.3% in unseen scenarios, which is a significant enhancement over the FILM's 22.2% SR and 33.9% GC. This improvement underscores the efficacy of our planning model, particularly in navigating novel environments.

D On the Fly Planning

As outlined in Section 2.4, agents operating in environments with partial information may require iterative replanning to enhance the precision of their paths. The detailed procedure is provided in Algorithm 1, which centralizes on iteratively refining the plan's accuracy by incorporating newly acquired information during the plan's execution. This approach, as opposed to random exploration, leverages the diffusion model's ability to learn from optimal trajectories. Specifically, executing an initial segment of the trajectory $\tau = s_{t+1:t+T}$ is more effective given the model's grounding in empirical data. With each step taken, beginning with s_{t+1} from trajectory τ at time t, the agent uncovers additional environmental details. This ongoing process of exploration enables the diffusion model to continually update the trajectory τ and target grounding g, resulting in a progressively comprehensive understanding of the environment. Consequently, this method is poised to elevate the overall success rate of the final execution.

Algorithm 1 On-the-fly planning with diffusion planner.

Inputs:
Planning Horizon: T, Instruction: I
Initialize:
$o_t \leftarrow o_0, s_t \leftarrow s_0, t \leftarrow 0$
for t = 0 to T **do**
  $\tau_K \sim \mathcal{N}(0, I)$, $g_K \sim \mathcal{N}(0, I)$
  $\mathbf{w}_K = (\tau_K, g_K)$
  for k = K to 1 **do**
    $z \sim \mathcal{N}(0, I)$ if $k > 1$, else $z = 0$
    $\mathbf{w}_{k-1} \leftarrow \frac{1}{\sqrt{\alpha_k}}(\mathbf{w}_k - \frac{1-\alpha_k}{\sqrt{1-\bar{\alpha}_k}}\epsilon_\theta(\mathbf{w}_k, k)) + \sigma_k z$
  end for
  $\tau, g = \mathbf{w}_0$
  $s_{t+1}, ..., s_{t+T} \leftarrow \tau$
  $a = (a_1, a_2, ..., a_N) \leftarrow \psi(s_t, s_{t+1})$
  $o_{t+1} \leftarrow Env(s_t, a)$
  $t \leftarrow t + 1$, $o_t \leftarrow o_{t+1}$, $s_t \leftarrow s_{t+1}$
  if o_t is terminated **then**
    break
  end if
end for

Table 5: Analysis of the on-the-fly planning on the CompILE dataset. We evaluate our model under the partially-observable setting to showcase the adaptability of our method when planning under uncertainty. .

Models	**P-MO2G**				
	SR	GD ↓	Path Prec.	Path Rec.	Path F1
1-shot	28.60	4.05	31.07	33.16	32.08
2-shot	54.70	2.33	47.97	51.52	49.68
On-the-fly (**Ours**)	**99.90**	**0.002**	**93.34**	**96.48**	**94.88**

The results presented in Table 5 reveals the effectiveness of our on-the-fly planning approach on the CompILE dataset under the partially-observable setting. On-the-fly planning refers to the model's iterative process of updating its representation of the environment and adjusting its plan accordingly, continuing until the agent reaches the target or exceed the maximum allowed steps. In the one-shot setting, which model only predicts the path at once, while the Success Rate (SR) is modest at 28.60%, the model significantly improves in the two-shot case, achieving an SR of 54.70%. Notably, the on-the-fly strategy excels with an impressive SR of 99.9%, demonstrating near-perfect performance. Goal Distance (GD) is minimized to 0.002 in the on-the-fly setting, indicating high precision in achieving the specified objectives. Path Precision (Path Prec.), Recall (Path Rec.), and F1 scores reflect this high accuracy and reliability, with the on-the-fly planning outperforming one-shot and two-shot approaches with scores of 93.34%, 96.48%, and 94.88%, respectively. These results underscore the adaptability and efficiency of our on-the-fly planning method when planning under uncertainty.

E Qualitative Analysis

In this section, we extend our visual analysis to include datasets from CompILE [36] and ALFRED [3]. Beyond illustrating the predicted trajectories, our visualizations also encompass target grounding g results. This comprehensive study facilitates the understanding of the contribution of our proposed method.

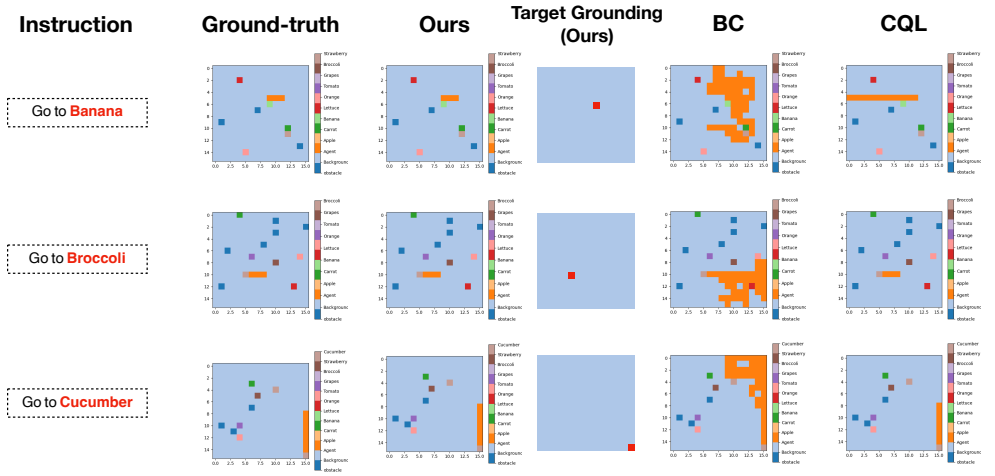

Figure 4: Qualitative Results on CompILE under multi-object and one goal setting ((MOG1). The target objects are marked in bold red. The predicted target position is highlighted in red.

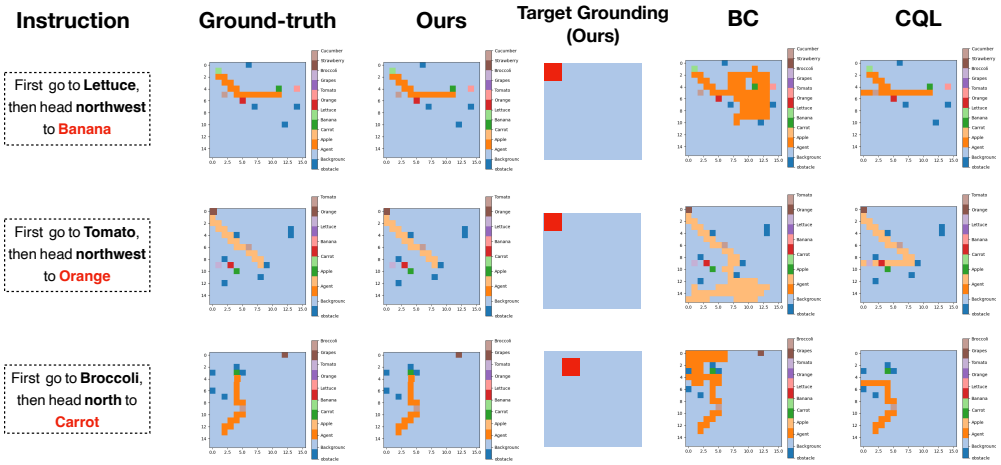

Figure 5: Qualitative Results on CompILE under partially-observable setting. The reference objects are highlighted in bold black and target objects in bold red. The inferred target position is highlighted in red.

E.1 CompILE

The qualitative results depicted in Fig. 4 offer a comparative analysis of our model's performance against baseline models BC and CQL in the CompILE dataset under two settings: multi-object single goal (**MOG1**) and partially-observable multi-object goal (**P-MOG2**). In the MOG1 scenarios, our model demonstrates a high fidelity in replicating the ground-truth paths towards single target objectives such as Banana and Cucumber, with the targets distinctly indicated in bold red. The baseline models, BC and CQL, display varying degrees of accuracy, with some paths showing indirect routes to the target.

In the more complex partially-observable setting, our model's ability to navigate first to a visible reference object and then to an initially unseen target object is showcased in Fig. 5. The reference objects serve as waypoints, and our model's trajectories indicate a clear understanding of this two-step process. For example, paths that first go to Lettuce and then head to Banana are more nuanced and exhibit strategic planning that is more aligned with the ground truth compared to the BC and CQL models, which either demonstrate indirect routes or miss the target altogether. The consistent alignment of our model's predicted paths with the ground-truth data across both MOG1 and P-MOG2 settings underscores its robustness and effectiveness in complex task execution.

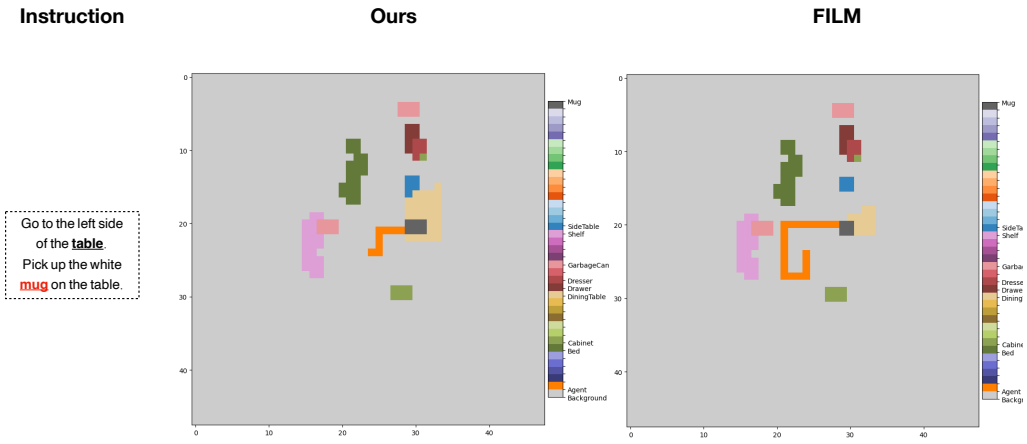

Figure 6: Qualitative Results on ALFRED. The reference objects are highlighted in bold black and target objects in bold red.

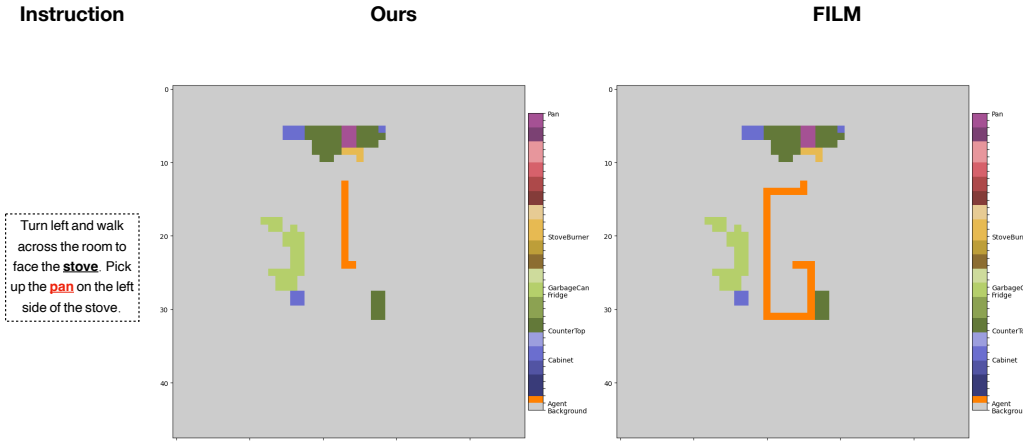

Figure 7: Qualitative Results on ALFRED. The reference objects are highlighted in bold black and target objects in bold red.

Our grounding objective is proposed to address the challenge of hallucination in models tasked with planning under conditions of partial observability. As demonstrated in the both figures, our framework's language-conditioning mechanism effectively refines the model's ability to infer target positions. This is evident from the closer alignment of our predicted trajectories and target groundings with the ground truth, as compared to the baseline models BC and CQL. By harnessing linguistic cues, our method not only contextualizes spatial reasoning but also enhances the plan's precision and reliability, leading to an increased rate of successful task completions.

E.2 ALFRED

The qualitative evaluations on the ALFRED dataset showcase our model's advanced capability to first identify the reference object and then proceed directly to the target object without the necessity for extensive exploration. For instance, in the task of retrieving the white mug from the table in Fig. 6, our model accurately pinpoints the table as the reference object, then moves decisively to the mug. This contrasts with the FILM model, which seems to engage in an initial exploratory phase before locating the mug. Similarly, in Fig. 7 when tasked with picking up the pan near the stove, our model efficiently recognizes the stove as the reference point and then the pan as the target, following an optimized path that demonstrates a clear understanding of the sequence of actions. The FILM model, however, appears to require a broader search of the environment, indicating a less efficient task execution process. These observations underline our model's proficiency in sequential task execution within complex and partially observable spaces.

F Limitations

Our model demonstrates promising results in both navigation and object manipulation tasks; however, it is important to recognize the boundaries within which it operates. We summarize the limitations of our method as follows:

Complex and Diverse Task Instructions. The performance discrepancy observed between the synthetic dataset (CompILE) and the realistic dataset (ALFRED), coupled with the outcomes from the instruction manipulation experiment detailed in Sec. 3.3, indicates that the primary challenge for our model is the diversity and complexity of real-world instructions. Addressing this challenge calls for future work to focus on enhancing the model's ability to parse and reason about varied linguistic constructs and the subtleties inherent in real-world instructions.

Our planning space is two-dimensional. Although our model excels at reasoning for certain navigation and manipulation tasks, it operates within a two-dimensional planning space, which restricts its applicability to environments that require three-dimensional spatial reasoning. This limitation presents an opportunity for future research to expand the model's capabilities into the third dimension, allowing for more comprehensive interaction within more complex environments.

On-the-fly planning is time-consuming. While on-the-fly planning did significantly improve our model's performance, the requirement to initiate the diffusion process from scratch for each prediction is a time-consuming procedure. Moving forward, to tackle the issue, we envision two primary avenues for enhancement. On one hand, optimizing the diffusion process for faster convergence or developing methods to resume from intermediate diffusion states could be pivotal. On the other end, we can develop an algorithm to determine whether we need to re-plan or not, which could reduce the computational demands and time costs associated with on-the-fly planning.