COMPOSITIONAL VISUAL PLANNING VIA INFERENCE-TIME DIFFUSION SCALING

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models excel at short-horizon robot planning, yet scaling them to long-horizon tasks remains challenging due to computational constraints and limited training data. Existing compositional approaches stitch together short segments by separately denoising each component and averaging overlapping regions. However, this suffers from instability as the factorization assumption breaks down in noisy data space, leading to inconsistent global plans. We propose that the key to stable compositional generation lies in enforcing boundary agreement on the estimated clean data (Tweedie estimates) rather than on noisy intermediate states. Our method formulates long-horizon planning as inference over a chain-structured factor graph of overlapping video chunks, where pretrained short-horizon video diffusion models provide local priors. At inference time, we enforce boundary agreement through a novel combination of synchronous and asynchronous message passing that operates on Tweedie estimates, producing globally consistent guidance without requiring additional training. Our training-free framework demonstrates significant improvements over existing baselines across 100 simulation tasks spanning 4 diverse sceness, effectively generalizing to unseen start-goal combinations that were not present in the original training data. Project website: https://comp-visual-planning.github.io/

1 INTRODUCTION

Generative diffusion models have shown strong capacity for modeling complex, high-dimensional distributions over images, videos, and robot plans. In planning, they offer a compelling alternative to per-instance optimization: instead of solving a new search problem for every start–goal pair, we can sample likely solutions from a learned generator. However, extending video-based planning to long horizons remains challenging: most backbones are trained on short clips, compute and memory scale unfavorably with sequence length, and long-range constraints (contacts, object persistence, and start–goal satisfaction) must be maintained throughout the rollout.

Classical planning methods, such as Task and Motion Planning (TAMP), decompose tasks into structured subproblems and enforce constraint satisfaction through symbolic operators, while hierarchical control methods first solve for a high-level task plan and then refine it into a low-level motion plan. Compositional diffusion planning follows these core principles but provides a data-driven, probabilistic alternative to hand-engineered classical and hierarchical planners. We adopt a compositional generation perspective on long-horizon planning, we compose plans from overlapping, short-horizon factors produced by a pretrained diffusion model. The central challenge is the consistency of this composition: during forward diffusion, noisy variables become entangled across time, breaking factorization assumptions behind common compositional heuristics (e.g., score averaging) (Zhang et al., 2023; Mishra et al., 2023; 2024; Bar-Tal et al., 2023) and yielding brittle behavior when long-range constraints must propagate.

Our key insight is to compose where diffusion model estimations are most reliable: on their Tweedie estimates, which provide a stable domain in which strong and explicit compositional heuristics can be applied. We formulate planning as inference in a chain-structured factor graph over overlapping video chunks. Local priors come from a short-horizon diffusion backbone; global coherence is enforced by boundary agreement on Tweedie predictions, not on noisy states. We propose two complementary message-passing mechanisms on these denoised variables: a synchronous scheme that views the chain as a Gaussian linear system and drives a single residual to zero with parallel, order-invariant updates, and an asynchronous scheme that propagates constraints via one-sided, stop-gradient targets for faster and more stable convergence. Both yield training-free guidance terms that we integrate into a DDIM sampler via diffusion-sphere guidance, balancing alignment with sample diversity.

We instantiate this approach for compositional visual diffusion planning, representing a plan as a sequence of images. Crucially, the method operates purely at inference time: the short-horizon diffusion backbone is trained once on short clips and then frozen; at test time we compose long-horizon plans via message passing on Tweedie estimates, with no additional training, fine-tuning, or task-specific adapters. The amount of test-time compute is controllable through the number of factors and guidance strength, enabling a direct plug-and-play use with existing backbones.

In summary, the key contributions of this work are: (1) A diffusion planning framework that models long-horizon plans as chain-structured factor graphs over video segments and enforces boundary agreement on Tweedie estimates rather than on noisy diffusion states. (2) Joint synchronous and asynchronous message passing over denoised variables, coupled with a training-free sampler that guides DDIM steps with diffusion-sphere guidance derived from message-passing residuals, preserving local sample quality, parallelism, and diversity while enforcing boundary agreement. (3) A compositional planning benchmark and empirical study demonstrating significant improvements in temporal coherence, static quality, and task success on held-out start–goal combinations compared with prior compositional baselines (Zhang et al., 2023) that operate on noisy diffusion states.

2 Related Work

Diffusion Models For Planning. A flurry of work has leveraged diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) for planning (Janner et al., 2022; Ajay et al., 2022; Dong et al., 2024a; He et al., 2023a; Ubukata et al., 2024; Lu et al., 2025; Chen et al., 2024a; Liang et al., 2023; Dong et al., 2024b), with various applications such as path finding (Carvalho et al., 2025; Luo et al., 2024), robotics (Pearce et al., 2023; Fang et al., 2024), and multi-agent (Zhu et al., 2024; Shaoul et al., 2024). Performance can be further improved by increasing test-time compute such as tree search (Feng et al., 2024a; Yoon et al., 2025b), hierarchical planning (Li et al., 2023; Chen et al., 2024b), and post-hoc refinement (Lee et al., 2024; Wang et al., 2022). Despite strong results, most prior work studies diffusion planning in low-dimensional state spaces that yield 2D trajectories. While recent work (Xu et al., 2025; Xie et al., 2025; Huang et al., 2024a) begins to consider more complex state spaces, they typically target simple or task-specific scenarios. In this paper, we investigate *visual diffusion planning*, where a plan is represented as a sequence of images, and we introduce a training-free sampling method that scales to significantly longer horizons and to unseen start–goal combinations.

Compositional Diffusion Generation. Compositional diffusion models are now well studied (Du et al., 2020; Garipov et al., 2023; Du & Kaelbling, 2024; Mahajan et al., 2024; Okawa et al., 2024; Thornton et al., 2025). One thread develops samplers for logical conjunctions of conditions, combining multiple prompts or constraints into coherent generations (Liu et al., 2022; Bradley et al., 2025; Zhang et al., 2025; Yang et al., 2023). A complementary thread scales the number of inference-time tokens while reusing models trained on short horizons—yielding wide-field panoramas (Zhang et al., 2023; Bar-Tal et al., 2023; Kim et al., 2024a; Lee et al., 2023), longer-duration videos (Wang et al., 2023; Kim et al., 2024b; 2025), and extended-horizon robotic plans (Zhang et al., 2023; Mishra et al., 2023; Luo et al., 2025). However, existing compositional methods suffer from instability when applied to noisy diffusion states, as they typically rely on score averaging or other heuristic combinations that assume factorization holds throughout the denoising process. In contrast, our approach operates on clean Tweedie estimates rather than noisy intermediate states, formulates the problem as factor graph inference with explicit boundary constraints, and employs principled message passing to maintain global consistency—yielding substantial improvements in both stability and plan quality over prior compositional planning methods.

Inference-Time Guidance for Diffusion Model. Inference-time guidance steers diffusion sampling without retraining, enabling adaptive, controllable behavior at test time. This flexibility has driven progress in image restoration (inpainting, deblurring) (Chung et al., 2024; Yang et al., 2024; Yu et al., 2023; Ye et al., 2024; Song et al., 2023a), style transfer (Bansal et al., 2023; He et al., 2023b), and robot motion/behavior generation (Liao et al., 2025; Black et al., 2025; Du & Song, 2025; Song et al., 2023b; Feng et al., 2024b). However, most existing methods steers a fixed-length output, we frame guidance as a form of message passing between tokens—allowing information to propagate across the sequence. This perspective lets us stitch together short behavioral fragments into long-range, temporally consistent visual plan.

3 Preliminaries

3.1 Factor Graph Formulation for Compositional Distributions

A factor graph $z = [u^1, u^2, \ldots, u^m]$ is a bipartite graph connecting factor nodes $\{x^i\}_{i=1}^n$ and variable nodes $\{u\}_{j=1}^m$, where $x^j \subseteq [u^1, u^2, \ldots, u^m]$. An undirected edge between x^i and u^j exists if and only if $u^j \in x^i$. Given a factor graph that represents the factorization of joint distribution, previous works DiffCollage approximate it with Bethe approximation (Zhang et al., 2023), and Generative Skill Chaining (GSC) extends the same formulation to robot task-and-motion planning as its follow-up work:

$$p(z_t) := \frac{\prod_{i=1}^n p(x_t^i)}{\prod_{j=1}^m p(u_t^j)^{d_j - 1}}, \tag{1}$$

where d_j is the degree of each variable u_j. Therefore, the estimated score is base on Bethe approximation is:

$$\nabla_{z_t} \log p(z_t) = \sum_{i=1}^n \nabla_{x_t^i} \log p(x_t^i) + \sum_{j=1}^m (1 - d_j) \nabla_{u_t^j} \log(u_t^j). \tag{2}$$

For example, consider a linear chain $z = [u^1, u^2, u^3, u^4, u^5]$ with factors $x_1 = [u^1, u^2, u^3]$ and $x_2 = [u^3, u^4, u^5]$. We can represent the joint noisy distribution as:

$$p(z_t) = \frac{p(u_t^1, u_t^2, u_t^3)\, p(u_t^3, u_t^4, u_t^5)}{p(u_t^3)}, \tag{3}$$

Given the linear chain graph, the overlapping variable u^3 (the one shared between neighboring factors x_1 and x_2) has degree $d = 2$, while the non-overlapping ones have $d = 1$ (i.e., their distribution come from individual factors). However, Bethe approximation(Eq. 1) holds in clean data (i.e., diffusion timestep $t = 0$), but does not hold when $t > 0$, we further prove its gap later (Theorem 1).

3.2 Diffusion model and Training-Free Guided Diffusion

Diffusion Models are a class of generative model that generates sample in the desired distribution from an initial Gaussian distribution $p(x_T)$ by iteratively performing a denoising process. It has a pre-defined forward process $q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) I)$, where $\bar{\alpha}$ is a scalar dependent on diffusion timestep t. In this work, we directly estimate Tweedie when training: $\mathbb{E}_{t, x_0, \epsilon_t}[\|x_0 - x_\theta(x_t, t)\|^2]$($x_0$ predictor) and apply a DDIM step when sampling:

$$x_{t-1} = \sqrt{\bar{\alpha}_t} x_{0|t} + \sqrt{1 - \bar{\alpha}_t - \sigma_t^2} \frac{x_t - \sqrt{\bar{\alpha}_t} x_{0|t}}{\sqrt{1 - \bar{\alpha}_t}} + \sigma_t \epsilon_t, \tag{4}$$

where estimated Tweedie $x_{0|t}$ is the output of $x_\theta(x_t, t)$.

Classifier Guidance (Dhariwal & Nichol, 2021) proposes to train a time dependent classifier in conditional generative tasks. Specifically, the conditional distribution $p(x_t|y)$ can be modeled by Bayes Rules $p(x_t|y) = p(x_t)p(y|x_t)/p(y)$: $\nabla_{x_t} \log p(x_t \mid y) = \nabla_{x_t} \log p(x_t) + \nabla_{x_t} \log p(y \mid x_t)$, where y represents the condition or measurement. This paper focuses on conditional guidance in a training-free manner, all the guidance in this paper is in training-free manner. In training-free guidance setting, instead of explicitly training a classifier, the guidance term $p(y \mid x_t)$ can be modeled as a potential function $\exp\left(-L(x_{0|t})\right)$, which simplifies to a gradient-descent update during inference time:

$$\nabla_{x_t} \log p(y \mid x_t) = \nabla_{x_t} \log \frac{\exp\left(-L(x_{0|t})\right)}{Z} = -\nabla_{x_t} L(x_{0|t}), \tag{5}$$

where $x_{0|t} = x_\theta(x_t, t)$ estimated by Tweedie's Predictor. Since $\exp\left(-L(x_{0|t})\right)$ is a point estimation of distribution of $\mathbb{E}_{x_0 \sim p(x_0|x_t)}[\exp(-L(x_0))]$, and the gap between them has a upper bound (Chung et al., 2024) and lower bound (Yang et al., 2024), Diffusion sphere guidance (Yang et al., 2024) is proposed to eliminate this gap based by formulating a constrained optimization problem over a hypersphere with the mean to be $\sqrt{\bar{\alpha}_t}x_{0|t} + \sqrt{1-\bar{\alpha}_t-\sigma^2}\frac{x_t-\sqrt{\bar{\alpha}_t}x_{0|t}}{\sqrt{1-\bar{\alpha}_t}}$ and radius $\sqrt{s}\sigma_t$(s is the shape of x_t), and derive a closed form solution for the update :

$$x_{t-1} = \sqrt{\bar{\alpha}_t}x_{0|t} + \sqrt{1-\bar{\alpha}_t-\sigma^2}\frac{x_t-\sqrt{\bar{\alpha}_t}x_{0|t}}{\sqrt{1-\bar{\alpha}_t}} - \sqrt{s}\sigma_t\frac{\nabla_{x_t}L(x_{0|t})}{\|\nabla_{x_t}L(x_{0|t})\|}. \tag{6}$$

4 Method

We formulate long-horizon planning as inference over a chain-structured factor graph of overlapping video chunks, where pretrained short-horizon diffusion models provide local priors. Our key innovation is enforcing boundary agreement on estimated clean data (Tweedie estimates) rather than noisy intermediate states, addressing the core limitation that factorization assumptions break down during diffusion sampling. We achieve this through novel synchronous and asynchronous message passing that operates on Tweedie estimates, producing globally consistent guidance without additional training. The approach involves formulating the planning problem as a factor graph (Section 4.1), deriving its distribution (Section 4.2), and sampling via our message passing scheme (Section 4.3).

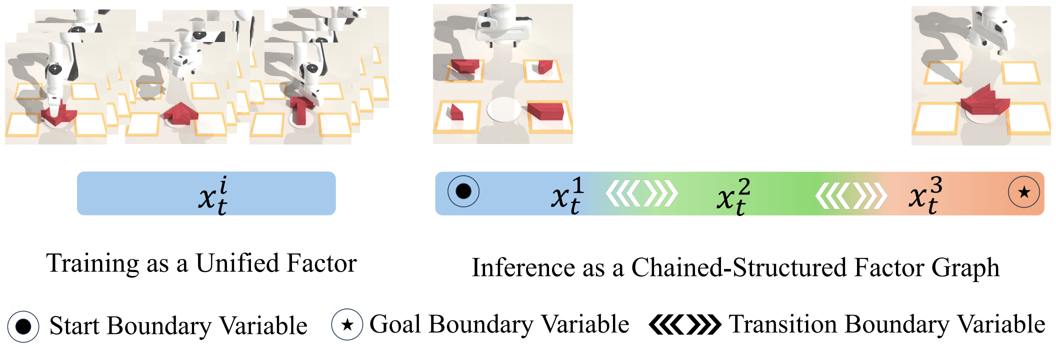
x_t^i
x_t^1
x_t^2
x_t^3
Training as a Unified Factor
Inference as a Chained-Structured Factor Graph
● Start Boundary Variable
★ Goal Boundary Variable
《《》》 Transition Boundary Variable

Figure 1: **Compositional Visual Planning via Inference Time Diffuser Scaling.** We train a short-horizon visual diffusion model on clips treated as a single factor. At inference, we scale visual planning horizon without retraining by chaining overlapping factors into a linear factor graph: the start and goal boundary variables are anchored at the ends, while neighboring factors exchange information through shared transition boundary variables.

4.1 Problem formulation

While a diffusion model can learn a prior over short, local behaviors, long-horizon planning requires additional structure to ensure feasibility. Beyond satisfying the start and the goal, intermediate pieces must stitch with local consistency. We therefore train a short-horizon diffusion model x_θ on local task segments; at test time, given a start image and a goal image, we sample from a Gaussian prior, partition the trajectory into overlapping chunks, generate each chunk with x_θ, and compose them into a coherent plan that is finally mapped back to an action sequence through inverse dynamics model.

We represent the plan as a linear chain $z = [u^1, \dots, u^m]$ and place n overlapping factors $x^i = [u^{2i-1}, u^{2i}, u^{2i+1}]$, $\quad i = 1, \dots, n$, each collecting three consecutive frames. The endpoints $u^1 = s$ and $u^m = g$ serve as the start and goal boundary variables. Let A_i and B_i denote linear selectors that extract the first and last frames of factor x^i, respectively. The feasibility of a plan is enforced by the following boundary agreements:

$$\begin{aligned} &\text{(Start/Goal Anchoring)} \quad A_1 x^1 = s, \qquad B_n x^n = g, \\ &\text{(Transition Boundary)} \quad B_i x^i = A_{i+1} x^{i+1}, \quad i = 1, \dots, n-1. \end{aligned} \tag{7}$$

This factorization reduces global planning to local generation with explicit boundary equalities and scales by reusing the same local model x_θ across time while preserving consistency via start–goal anchoring and transition agreements. All factors/variable in latent space encoded by the Cosmos tokenizer (et. al., 2025) into a compact latent representation; we perform planning entirely in this latent space rather than pixels, which reduces dimensionality to save compute.

4.2 Distribution of Factor Graph

Prior work (Zhang et al., 2023; Mishra et al., 2023; 2024) relies on the Bethe-style product of factors normalized by variables (Eq. 1), which is accurate on clean data. Forward diffusion, however, perturbs factorization assumption between factors and variables.

Theorem 1 (Noisy-Bethe Gap Theorem)**.** *Consider a linear chain $z = [u^1, u^2, u^3]$ with pairwise factors $[u^1, u^2]$ and $[u^2, u^3]$, where u^2 is the transition boundary variable. Assume the forward noising processes are $p(u_t^1, u_t^2 \mid u^1, u^2)$, $p(u_t^2, u_t^3 \mid u^2, u^3)$, and $p(u_t^2 \mid u^2)$. Let $a(u^2) = \int p(u^1, u^2) p(u_t^1, u_t^2 \mid u^1, u^2)\, du^1$, $b(u^2) = \int p(u^2, u^3) p(u_t^2, u_t^3 \mid u^2, u^3)\, du^3$, $c(u^2) = p(u^2) p(u_t^2 \mid u^2)$, $Z = \int c(u^2)\, du^2$, and $q(u^2) = c(u^2)/Z$. Denote by $p(u_t^1, u_t^2, u_t^3)$ the true noisy distribution and by $\hat{p}(u_t^1, u_t^2, u_t^3)$ the estimator from Eq. 1. Then the gap between true distribution and estimated distribution is:*

$$\Delta \;=\; p(u_t^1, u_t^2, u_t^3) - \hat{p}(u_t^1, u_t^2, u_t^3) \;=\; Z\, \mathrm{Cov}_{u^2 \sim q}\Big[\tfrac{a}{c},\, \tfrac{b}{c}\Big]. \tag{8}$$

Interpretation. The proof is in appendix A. We can view $a(u^2)$ as the left-factor message into the boundary u^2, $b(u^2)$ as the right-factor message into the boundary, and $c(u^2)$ as the local boundary evidence. Intuitively, $a(u^2)$ and $b(u^2)$ quantify how the left and right pairwise factors, after passing through their respective forward-noise channels, "vote" for different boundary values u^2. The term $c(u^2)$ provides the unary baseline that captures how plausible each boundary value is on its own (and how it transmits noise). The noisy-Bethe gap Δ is exactly the covariance—under the boundary weighting q—between the two relative gains $a(u^2)/c(u^2)$ and $b(u^2)/c(u^2)$. When these gains are uncorrelated (or proportional) across u^2, the covariance vanishes and the Bethe approximation remains accurate. Forward diffusion typically introduces shared, heteroscedastic distortions in u^2, which make the two gains rise and fall together; this produces a nonzero covariance and, consequently, a systematic gap.

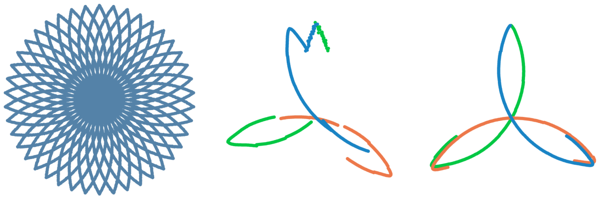
(a) Dataset Distribution (b) DiffCollage/GSC (c) Ours

Figure 2: Motivating toy example. We train a short-horizon diffusion model on circular *arc* clips (left). At test time, three 120° arc generators are composed to form a three-petal "flower".

Figure 2 illustrates the core failure mode of DiffCollage which is deployed based on noisy factorization assumption. A DiffCollage/GSC-style stitcher (middle) drifts and leaves boundary gaps, while our inference-time message passing (right) aligns shared boundaries and closes the loops. This mirrors the limitation of a Bethe-style product of factors (Eq. 1) under forward diffusion: noise corrupts factorization assumption. Motivated by the Noisy-Bethe gap, instead of enforcing dependencies directly among the noisy factors $x_t^{1:n}$, we impose them on the concatenated Tweedie (denoised) estimates $x_{0|t}^{1:n}$. Accordingly, our approximation to the factor-graph distribution is:

$$p(z_t) = \prod_{i=1}^{n} p(x_t^i) \cdot \exp(-L(x_{0|t}^{1:n})), \tag{9}$$

where $\exp(-L(x_{0|t}^{1:n}))$ acts as a potential that penalizes inconsistencies—and thus enforces dependencies—among the estimated clean variables $x_{0|t}^{1:n} = x_\theta(x_t^{1:n})$.

4.3 Jointly Synchronous and Asynchronous Message Passing

Message passing proceeds through *boundary factors*: when the transition boundaries together with the start and goal boundaries agree, the plan is feasible (see Eq. 7). We therefore optimize boundary agreement explicitly. Our synchronous scheme treats the chain as a Gaussian linear system and drives a single residual ($\Sigma^{-1} x_{0|t}^{1:n} = \eta$) to zero via parallel updates, but can be numerically stiff. Our asynchronous scheme uses one-sided, stop-gradient targets to propagate constraints forward and backward in a TD-style manner, yielding faster and more stable convergence at the cost of mild bias. Finally, diffusion-sphere guidance interpolates between unconditional sampling and loss-driven descent, balancing alignment and diversity.

4.3.1 Synchronous Message Passing

We encode the boundary condition as a Gaussian potential, $\psi_{i-1,i} := \exp(-\frac{1}{c_{i-1}}\|B_{i-1}x^{i-1}_{0|t} - A_i x^i_{0|t}\|^2)$, where c_{i-1} denotes the variance.

Theorem 2 (Synchronous Message Passing Constraint). *Let $x^{1:n} \in \mathbb{R}^{n\times tchw}$ denote the concatenated intermediate factors in a chain-structured factor graph with transition boundaries $\psi_{i-1,i}$. Given the start boundary $\psi_{s,1}$ and the goal boundary $\psi_{n,g}$, the joint constraints distribution over all intermediate factors is Gaussian:*

$$p_{\text{sync}}(x^{1:n} \mid s,\, g) \;\propto\; \exp(-\tfrac{1}{2}(x^{1:n})^T \Sigma^{-1} x^{1:n} \;+\; \eta^T x^{1:n}), \tag{10}$$

$$\textit{where } \Sigma^{-1} = \begin{bmatrix} \frac{A_1^T A_1}{c_0} + \frac{B_1^T B_1}{c_1} & -\frac{B_1^T A_2}{c_1} & & \\ -\frac{A_2^T B_1}{c_1} & \frac{A_2^T A_2}{c_1} + \frac{B_2^T B_2}{c_2} & -\frac{B_2^T A_3}{c_2} & \\ & -\frac{A_3^T B_2}{c_2} & \frac{A_3^T A_3}{c_2} + \frac{B_3^T B_3}{c_3} & \\ & & & \ddots \end{bmatrix}, \quad \eta = \begin{bmatrix} \frac{A_1^T s}{c_0} \\ 0 \\ 0 \\ \vdots \\ \frac{B_n^T g}{c_n} \end{bmatrix}.$$

The detailed proof is in appendix B. We perform synchronous message passing on the estimated Tweedie factors $x^{1:n}_{0|t}$ by penalizing the deviation from the consistent linear system $\Sigma^{-1}x^{1:n}_{0|t} = \eta$. In practice, we set $c_i = 1$ for all $i = 0, \ldots, n$.:

$$L_{sync} = \left\| \Sigma^{-1} x^{1:n}_{0|t} - \eta \right\| \tag{11}$$

Here, synchronous refers to a lockstep update schedule, in which all updates are computed from the same current iterate and applied simultaneously. This scheme preserves parallelism, eliminates order-dependent effects, and guides the Tweedie estimates toward satisfying the chain constraints.

4.3.2 Asynchronous Message Passing

While the synchronous objective is conceptually clean, the resulting hard consistency constraint is difficult to optimize and often exhibits slow or unstable convergence (Ortiz et al., 2021). To improve stability and speed, we adopt an asynchronous scheme with bootstrapped targets and stop-gradient, akin to temporal-difference updates Hansen et al. (2024); Li et al. (2025). Concretely, we optimize:

$$\begin{aligned} L_{async} = &\underbrace{\left\| s - A_1 x^1_{0|t} \right\| + \sum_{i=1}^{n-1} \gamma^i \left\| sg(B_i \hat{x}^i_{0|t}) - A_{i+1} x^{i+1}_{0|t} \right\|}_{\text{forward passing}} \\ &+ \underbrace{\sum_{i=1}^{n-1} \gamma^{n-i} \left\| B_i x^i_{0|t} - sg(A_{i+1} \hat{x}^{i+1}_{0|t}) \right\| + \left\| B_n x^n_{0|t} - g \right\|}_{\text{backward passing}}, \end{aligned} \tag{12}$$

where $sg(\cdot)$is the stop-gradient operator, $\hat{x}^{i+1}_{0|t}$ is a target produced by diffusion model with latest parameters, and $x^i_{0|t}$ is produced by an EMA of the model parameters. The discount γ down-weights messages as they move away from the start or the goal.

The boundary terms $\left\| s - A_1 x^1_{0|t} \right\|$ and $\left\| B_n x^n_{0|t} - g \right\|$ anchor the chain to the start and goal. The forward message loss penalizes mismatch between $B_i \hat{x}^i_{0|t}$ (outgoing) and $A_{i+1} x^{i+1}_{0|t}$ (incoming), with $sg(\cdot)$ enforcing one-way forward passing. Similarly, the backward message loss mirrors the same constraint in the reverse direction.

Algorithm 1 Compositional Generation

```
Require: Model EMA $x_\theta$, Latest Model $\hat{x}_\theta$
Hyperparameters: Diffusion Time Step $T$, number of factors chained $n$, guidance weight $g$
Sample $z_T \sim \mathcal{N}(0, I)$
Split $z_T$ to $n$ overlapping chunks $x_T^{1:n}$
for $t = T$ to $1$ do
    $x_{0|t}^{1:n} = x_\theta(x_t^{1:n})$                                   ▷ forward model passing
    $\hat{x}_{0|t}^{1:n} = \hat{x}_\theta(x_t^{1:n})$
    $\mu_{t-1}^{1:n} = \sqrt{\bar{\alpha}_t} x_{0|t}^{1:n} + \sqrt{1 - \bar{\alpha}_t - \sigma^2} \frac{x_t^{1:n} - \sqrt{\bar{\alpha}_t} x_{0|t}^{1:n}}{\sqrt{1-\bar{\alpha}_t}}$      ▷ DDIM Step
    $L = L_{sync} + L_{async}$                                  ▷ Jointly sync and async message passing
    $d^* = -\sqrt{s}\sigma_t \cdot \frac{\nabla_{x_t^{1:n}} L}{\|L\|}$                    ▷ Diffusion Sphere Guidance
    $d^{sample} = \sigma_t \epsilon_t$
    $d_m = d^{sample} + g(d^* - d^{sample})$
    $x_{t-1}^{1:n} = \mu_{t-1}^{1:n} + r \frac{d_m}{\|d_m\|}$
end for
merge chunks $x_0^{1:n}$ to get final plan $z_0$
return $z_0$
```

4.3.3 Diffusion-Sphere Guided Message Passing

Having derived differentiable losses for synchronous and asynchronous message passing, we adopt the training-free guidance of DSG (Yang et al., 2024). As noted by DSG (Eq. 6), stronger guidance improves alignment but can reduce sample diversity. To balance alignment and exploration, we interpolate between the unconditional sampling direction and the normalized descent direction induced by our loss:

$$d_m = d^{sample} + g_r(d^* - d^{sample}), \quad x_{t-1}^{1:n} = \mu_{t-1}^{1:n} + r \frac{d_m}{\|d_m\|}. \tag{13}$$

Here $d^{sample} = \sigma_t \epsilon_t$ is the unconditional annealing step, $d^* = -\sqrt{s}\, \sigma_t \cdot \frac{\nabla_{x_t^{1:n}} L}{\|\nabla_{x_t^{1:n}} L\|}$ is the steepest descent direction for our sync/async objective, and d_m is subsequently normalized to satisfy the spherical-Gaussian constraint.

4.4 Compositional Video Planning via Inference-Time Diffusion Scaling

The procedure for compositional generation is summarized in Algorithm 1. At a high level, the goal is to generate long-horizon trajectories by decomposing them into overlapping local factors and enforcing boundary agreement across those factors during the diffusion sampling process. At each timestep, DDIM provides the base update, while a joint synchronous–asynchronous message passing loss defines a residual that Diffusion Sphere Guidance interpolates against, steering updates toward agreement without collapsing diversity. The resulting updates are local and parallel across overlapping factors yet collectively converge to a feasible, consistent plan. After all steps, merging the denoised chunks yields a smooth, temporally aligned trajectory z_0.

For robot manipulation planning, we train a video diffusion model on randomly sampled short chunks from long-horizon demonstrations and an inverse dynamics model that predicts actions from consecutive frames. At test time, we condition the diffusion model on start and goal images and apply the compositional generation procedure to produce complete video plans. The resulting visual trajectory is converted into executable robot actions via the inverse dynamics model. The procedure is training-free, plug-and-play, and compatible with unconditional short-horizon diffusion backbones, enabling generalization to unseen start-goal combinations without task-specific retraining.

5 Experiments

We present experiment results on multiple robotic manipulation scenes spanning 100 tasks (18 in distribution, 82 out of distribution) of varying difficulties. Our objective is to investigate (1) the visual fidelity of the generated video plans (Section 5.1) (2) how the proposed compositional visual planning can generalize to long-horizon unseen tasks (Section 5.2) (3) how each proposed component affect the generation performance of our method (Section 5.3) (4) the effectiveness of our method in real robot manipulation tasks (Section 5.4).

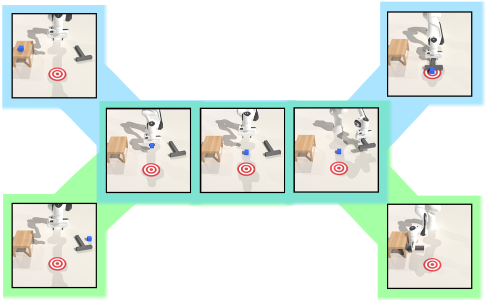

Figure 3: **Tool-Use setup.**This task involves 2 start and 2 goal configurations. We also evaluate on more challenging tasks. A complete list of task settings is provided in Appendix D.

Compositional Planning Benchmark. We develop a benchmark for compositional planning in robotic manipulation based on ManiSkill (Mu et al., 2021), where each scene contains N start states and N goal states, resulting in $N \cdot N$ tasks (i.e., different start/goal pairs) per scene, as shown in Figure 6. Our training datasets only contains demonstrations for N start-goal pairs. At test time, we evaluate the planner on both the N seen start–goal pairs (in-distribution) and the remaining $N \cdot N - N$ unseen pairs (out-of-distribution), since a capable planner should generalize to new combinations if the dataset covers all the required fragments. For example, in the Tool-Use setting (Figure 6), demonstrations cover only the blue or green regions; the planner must generalize across them to form cross-region plans unseen in the dataset but composable from its fragments. We address this type of generation by learning from short demonstration chunks randomly taken from long-horizon tasks and compositionally generates multiple chunks at inference time to construct the final plan.

Evaluation Setup. Given specified start and goal image as task context, our method first synthesize a sequence of frames as subgoals. We then use an MLP-based inverse dynamics model to predict the pose of end-effector for the robot to execute, conditioned on adjacent images. The inverse dynamics models are trained using the same demonstrations as the planner conditioned on adjacent images. We report both the video quality metrics and the success rates of the planners over the robotic manipulation tasks. An episode is counted as success if the target objects ends up the specified state within a small tolerance. For each environment, we report the success rate over all evaluation episodes. We evaluate all methods with 5 random seeds for each experiment and report the mean and standard deviation.

Baselines. For both policy-based and composition-based baseline, details are in Appendix C. For compositional baseline **DiffCollage/GSC** (where GSC is DiffCollage adapted to robotic planning), we condition on the start and goal images to generate an entire plan by noisy factorization Eq. 2, and then use an inverse dynamics model to execute that plan.

5.1 Video Generation Quality Study

Beyond reporting task success rate, we also evaluate the visual fidelity of the synthesized video plans. Even when rollouts verify feasibility, perceptual quality still warrants careful analysis. Accordingly, we score generated videos with VBench++ (Huang et al., 2024b), focusing on robotics-centric metrics that matter for control: *Dynamic Quality* (inter-frame), which includes (i) motion smoothness—capturing temporal stability of robot/object motion—and (ii) background consistency—testing whether the scene remains coherent over time; and *Static Quality* (frame-wise), which includes Aesthetic and Imaging Quality to ensure frames are clear and largely artifact-free. Our generation strategy substantially improves the time-dependent properties that matter for control: across all scenes and distributions, Motion Smoothness and Background Consistency far exceed DiffCollage (Zhang et al., 2023). This translates into dynamically executable trajectories and coherent spatiotemporal scenes. In terms of static quality, Aesthetic remains comparable, while Imaging shows a consistent and large advantage, directly reflecting fewer blurry frames and cleaner visuals.

5.2 Compositional Planning Benchmark

We present the robot manipulation success rates of 4 different escenes in Table 2. We separately report the success rates of IND and OOD tasks, where IND represents the N tasks seen in the training

Scene	Type	Dynamic Quality ↑				Static Quality ↑			
		Motion Smoothness		Background Consistency		Aesthetic		Imaging	
		DiffCollage	Ours	DiffCollage	Ours	DiffCollage	Ours	DiffCollage	Ours
Tool-Use	IND	0.45±0.06	**0.98**±0.03	0.48±0.05	**0.94**±0.03	0.47±0.02	**0.46**±0.02	0.42±0.03	**0.70**±0.02
	OOD	0.41±0.06	**0.96**±0.05	0.38±0.07	**0.89**±0.06	0.42±0.03	**0.48**±0.03	0.40±0.05	**0.70**±0.04
Drawer	IND	0.43±0.03	**0.85**±0.03	0.50±0.04	**0.86**±0.03	0.50±0.02	**0.54**±0.02	0.39±0.02	**0.73**±0.02
	OOD	0.44±0.05	**0.96**±0.05	0.41±0.06	**0.89**±0.05	0.45±0.03	**0.48**±0.03	0.42±0.04	**0.70**±0.04
Cube	IND	0.40±0.04	**0.96**±0.03	0.52±0.04	**0.90**±0.03	0.49±0.02	**0.51**±0.02	0.33±0.03	**0.65**±0.02
	OOD	0.42±0.06	**0.97**±0.05	0.39±0.06	**0.91**±0.05	0.51±0.03	**0.52**±0.03	0.41±0.05	**0.63**±0.04
Puzzle	IND	0.40±0.03	**0.96**±0.03	0.47±0.04	**0.90**±0.03	0.49±0.04	**0.48**±0.02	0.44±0.03	**0.70**±0.02
	OOD	0.39±0.06	**0.97**±0.05	0.48±0.06	**0.90**±0.05	0.46±0.03	**0.47**±0.03	0.38±0.05	**0.70**±0.04
Overall	IND	0.41±0.04	**0.94**±0.06	0.49±0.04	**0.90**±0.04	0.49±0.03	**0.50**±0.03	0.40±0.05	**0.70**±0.03
	OOD	0.40±0.06	**0.97**±0.05	0.45±0.07	**0.90**±0.05	0.46±0.04	**0.48**±0.03	0.39±0.05	**0.69**±0.05

Table 1: **Comparison across four scenes on Dynamic/Static Quality.** Our results are averaged over 5 seeds and standard deviations are shown after the ± sign.

data while OOD represents the $N \cdot N - N$ unseen tasks. We observe that DiffCollage fails at almost all tasks. Qualitatively, we find that the synthesized images of DiffCollage tend to be blurry or even unrealisitc, perhaps due to its score averaging sampling scheme. Such suboptimal images will further confuses the inverse dynamic models, cause unstable behaviors and failures. In contrast, our method achieves significantly higher success rates, indicating that the generated visual plans are realistic and accurate for the inverse dynamic model to follow. We also include several multiple representative policy learning baselines, such as Goal-Conditioned Diffusion Policy (GCDP). We notice that though strong policy learning baseline is able to perform well on IND tasks, their performance suffers from a significant degradation on OOD tasks. In contrast, our method—enabled by the graphical chain formulation and message passing—maintains stable performance regardless of the task distribution.

Scene	Type	LCBC	LCDP	GCBC	GCDP	DiffCollage	CompDiffuser	Ours
Tool-Use	IND	80±7	95±2	85±5	96±3	1±2	60±2	**97**±3
	OOD	15±3	37±8	13±6	42±13	0±0	51±3	**96**±2
Drawer	IND	35±6	54±6	30±5	50±6	0±0	20±5	**53**±5
	OOD	6±5	26±14	7±5	18±16	0±0	18±3	**52**±6
Cube	IND	28±3	58±5	26±4	60±3	0±0	32±8	**64**±10
	OOD	8±3	22±12	5±5	24±13	0±0	34±6	**65**±9
Puzzle	IND	23±5	48±5	19±6	47±3	0±0	10±3	**50**±11
	OOD	0±0	11±9	0±0	12±11	0±0	9±3	**50**±13
Overall	IND	33±18	57±15	30±21	56±16	0±1	17±2	**59**±17
	OOD	2±4	15±12	15±12	15±13	0±0	16±2	**54**±14

Table 2: **Quantitative Results on Compositional Planning Bench.** We benchmark our method on the 100 test-time tasks across 4 scenes with 30 episodes per task. Our results are averaged over 5 seeds and standard deviations are shown after the ± sign.

5.3 Ablation Studies

Jointly Synchronous & Asynchronous Message Passing. We compare the success rates of three variants of our test-time compositional sampling scheme in Cube Scene: Only Synchronous Loss (*Sync Only*), Only Asynchronous Loss (*Async Only*), and Joint Synchronous and Asynchronous Loss (*Sync & Async*), as shown in Figure 4. *Sync only* suffers from overly tight constraints that are difficult to optimize, leading to lower success rates. In contrast, the asynchronous variant performs better. Combining the two—*Sync & Async*—outperforms either alone, likely due to its more effective balance of constraint enforcement and flexibility.

Scaling in Sampling Steps. We study how the number of diffusion sampling steps affects the planning performance on Drawer Scene (Figure 5). Success rates improve as the number of steps increases, demonstrating that our method scales effectively with additional test-time compute. We hypothesize that taking more steps enables deeper cross-factor message passing through repeated denoising and guidance updates, which in turn reduces boundary inconsistencies and yields more accurate, temporally coherent plans.

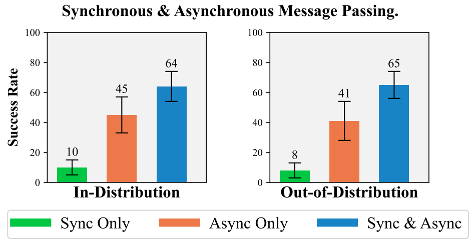

Figure 4: **Effect of synchronous and asynchronous message passing.** Results are averaged over 5 seeds on the Cube scene. Combining(*Sync & Async*) achieves the best results for both in-distribution and out-of-distribution tasks.

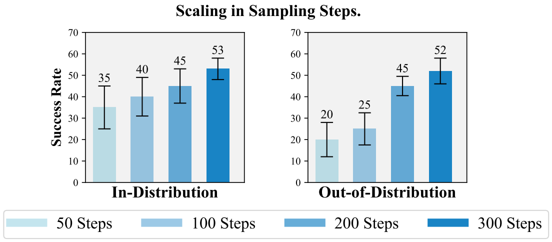

Figure 5: **Effect of sampling steps on planning performance.** Results are averaged over 5 seeds on Drawer Scene. Increasing the number of diffusion sampling steps improves success rates for both in-distribution and out-of-distribution tasks.

5.4 Real Robot Experiment

For the real-world experiments, we deploy our method on a Franka Emika Panda robotic manipulation, controlled at 20Hz using joint impedance control. Visual image observations are captured using an Intel RealSense D435 depth camera. For data collection, the robot is teleoperated using a Meta Quest 3 headset, with tracked Cartesian poses converted to joint configurations through inverse kinematics. Quantitative and qualitative results shows our compositional planner continues to produce coherent long-horizon behaviors and maintains strong performance in the real world. We present the corresponding results in table 3. More details are covered in appendix H.

Real Scene	**IND:Task1**	**IND:Task2**	**OOD:Task3**	**OOD:Task4**
DiffCollage	1/10	1/10	0/10	0/10
Ours	9/10	7/10	10/10	8/10

Table 3: **Real-robot success rates.** Our method substantially outperforms DiffCollage across both in-distribution (IND) and out-of-distribution (OOD) tasks on real hardware.

6 Discussion and Conclusion

Limitations. Our method has several limitations. First, it relies on the accuracy of the estimated clean data (Tweedie estimates) during denoising, since guidance losses are computed on these estimates. This sensitivity could be mitigated by performing multi-step Tweedie estimation or by scaling up training data and model capacity. Second, as in prior work, the number of test-time composed segments, n, must be specified manually. Developing procedures that automatically infer n from task structure and uncertainty would be an interesting future research direction. Lastly, our approach can be more computationally demanding than direct averaging-based sampling, because test-time guidance is implemented via gradient-based optimization(Table 6). Exploring lighter optimization schedules could potentially reduce this overhead.

Conclusion We introduced Compositional Visual Planning, an inference-time method that composes long-horizon plans by stitching overlapping video factors with message passing on Tweedie estimates. A chain-structured factor graph imposes global consistency, enforced via joint synchronous and asynchronous updates, while diffusion-sphere guidance balances alignment and diversity without retraining. Compositional Visual Planning is plug-and-play with short-horizon diffusion video prediction model, scales with test-time compute, and generalizes to unseen start–goal combinations. Beyond robotics, the framework is applicable to broader domains, such as panorama image generation and long-form text-to-video synthesis, which we leave for future exploration.

Reproducibility Statement

We promise to provide all the source code to reproduce the results in this paper, including the proposed algorithm and the evaluation benchmark.

LLM Usage Disclosure

We confirm that no large language models (LLMs) were used to generate, edit, or refine the scientific content, experimental design, results, or conclusions of this paper. All text, figures, algorithms, and analyses were produced entirely by the authors.

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B Tenenbaum, Tommi S Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2022. 2

Anonymous. Compositional diffusion with guided search for long-horizon planning. In *Submitted to The Fourteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=b8avf4F2hn. under review. 23

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models, 2023. URL https://arxiv.org/abs/2302.07121. 2

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation, 2023. URL https://arxiv.org/abs/2302.08113. 1, 2

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies, 2025. URL https://arxiv.org/abs/2506.07339. 2

Arwen Bradley, Preetum Nakkiran, David Berthelot, James Thornton, and Joshua M Susskind. Mechanisms of projective composition of diffusion models. *arXiv preprint arXiv:2502.04549*, 2025. 2

Joao Carvalho, An T Le, Piotr Kicki, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and adapting robot motion planning with diffusion models. *IEEE Transactions on Robotics*, 2025. 2

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015. URL https://arxiv.org/abs/1512.03012. 19

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024a. 2

Chang Chen, Fei Deng, Kenji Kawaguchi, Caglar Gulcehre, and Sungjin Ahn. Simple hierarchical planning with diffusion. *arXiv preprint arXiv:2401.02644*, 2024b. 2

Hyungjin Chung, Jeongsol Kim, Michael T. Mccann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems, 2024. URL https://arxiv.org/abs/2209.14687. 2, 3

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. URL https://arxiv.org/abs/2105.05233. 3

Zibin Dong, Jianye Hao, Yifu Yuan, Fei Ni, Yitian Wang, Pengyi Li, and Yan Zheng. Diffuserlite: Towards real-time diffusion planning. *Advances in Neural Information Processing Systems*, 37: 122556–122583, 2024a. 2

Zibin Dong, Yifu Yuan, Jianye HAO, Fei Ni, Yao Mu, YAN ZHENG, Yujing Hu, Tangjie Lv, Changjie Fan, and Zhipeng Hu. Aligndiff: Aligning diverse human preferences via behavior-customisable diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024b. 2

Maximilian Du and Shuran Song. Dynaguide: Steering diffusion polices with active dynamic guidance, 2025. URL https://arxiv.org/abs/2506.13922. 2

Yilun Du and Leslie Kaelbling. Compositional generative modeling: A single model is not all you need. *arXiv preprint arXiv:2402.01103*, 2024. 2

Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33:6637–6647, 2020. 2

NVIDIA et. al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025. 4

Xiaolin Fang, Caelan Reed Garrett, Clemens Eppner, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Dieter Fox. Dimsam: Diffusion models as samplers for task and motion planning under partial observability. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1412–1419. IEEE, 2024. 2

Lang Feng, Pengjie Gu, Bo An, and Gang Pan. Resisting stochastic risks in diffusion planners with the trajectory aggregation tree. In *Forty-first International Conference on Machine Learning*, 2024a. 2

Zeyu Feng, Hao Luan, Pranav Goyal, and Harold Soh. Ltldog: Satisfying temporally-extended symbolic constraints for safe diffusion-based planning. *IEEE Robotics and Automation Letters*, 9(10):8571–8578, October 2024b. ISSN 2377-3774. doi: 10.1109/lra.2024.3443501. URL http://dx.doi.org/10.1109/LRA.2024.3443501. 2

Timur Garipov, Sebastiaan De Peuter, Ge Yang, Vikas Garg, Samuel Kaski, and Tommi Jaakkola. Compositional sculpting of iterative generative processes. *arXiv preprint arXiv:2309.16115*, 2023. 2

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control, 2024. URL https://arxiv.org/abs/2310.16828. 6

Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *Advances in neural information processing systems*, 36:64896–64917, 2023a. 2

Yutong He, Naoki Murata, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Dongjun Kim, Wei-Hsiang Liao, Yuki Mitsufuji, J. Zico Kolter, Ruslan Salakhutdinov, and Stefano Ermon. Manifold preserving guided diffusion, 2023b. URL https://arxiv.org/abs/2311.16424. 2

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

Shuaiyi Huang, Mara Levy, Zhenyu Jiang, Anima Anandkumar, Yuke Zhu, Linxi Fan, De-An Huang, and Abhinav Shrivastava. Ardup: Active region video diffusion for universal policies. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8465–8472. IEEE, 2024a. 2

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Vbench++: Comprehensive and versatile benchmark suite for video generative models, 2024b. URL https://arxiv.org/abs/2411.13503. 8

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pp. 9902–9915. PMLR, 2022. 2

Jaihoon Kim, Juil Koo, Kyeongmin Yeo, and Minhyuk Sung. Synctweedies: A general generative framework based on synchronized diffusions, 2024a. URL https://arxiv.org/abs/2403.14370. 2

Jihwan Kim, Junoh Kang, Jinyoung Choi, and Bohyung Han. Fifo-diffusion: Generating infinite videos from text without training. *Advances in Neural Information Processing Systems*, 37: 89834–89868, 2024b. 2

Subin Kim, Seoung Wug Oh, Jui-Hsien Wang, Joon-Young Lee, and Jinwoo Shin. Tuning-free multi-event long video generation via synchronized coupled sampling. *arXiv preprint arXiv:2503.08605*, 2025. 2

Kyowoon Lee, Seongun Kim, and Jaesik Choi. Refining diffusion planner for reliable behavior synthesis by automatic detection of infeasible plans. *Advances in Neural Information Processing Systems*, 36, 2024. 2

Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. Syncdiffusion: Coherent montage via synchronized joint diffusions, 2023. URL https://arxiv.org/abs/2306.05178. 2

Shangzhe Li, Zhiao Huang, and Hao Su. Reward-free world models for online imitation learning, 2025. URL https://arxiv.org/abs/2410.14081. 6

Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In *International Conference on Machine Learning*, pp. 20035–20064. PMLR, 2023. 2

Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. In *International Conference on Machine Learning*, pp. 20725–20745. PMLR, 2023. 2

Qiayuan Liao, Takara E. Truong, Xiaoyu Huang, Guy Tevet, Koushil Sreenath, and C. Karen Liu. Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion, 2025. URL https://arxiv.org/abs/2508.08241. 2

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. Compositional visual generation with composable diffusion models. In *European Conference on Computer Vision*, pp. 423–439. Springer, 2022. 2

Haofei Lu, Dongqi Han, Yifei Shen, and Dongsheng Li. What makes a good diffusion planner for decision making? In *The Thirteenth International Conference on Learning Representations*, 2025. 2

Yunhao Luo, Chen Sun, Joshua B Tenenbaum, and Yilun Du. Potential based diffusion motion planning. *arXiv preprint arXiv:2407.06169*, 2024. 2

Yunhao Luo, Utkarsh A Mishra, Yilun Du, and Danfei Xu. Generative trajectory stitching through diffusion composition. *arXiv preprint arXiv:2503.05153*, 2025. 2

Divyat Mahajan, Mohammad Pezeshki, Ioannis Mitliagkas, Kartik Ahuja, and Pascal Vincent. Compositional risk minimization. *arXiv preprint arXiv:2410.06303*, 2024. 2

Utkarsh Aashu Mishra, Shangjie Xue, Yongxin Chen, and Danfei Xu. Generative skill chaining: Long-horizon skill planning with diffusion models. In *Conference on Robot Learning*, pp. 2905–2925. PMLR, 2023. 1, 2, 5, 16

Utkarsh Aashu Mishra, Yongxin Chen, and Danfei Xu. Generative factor chaining: Coordinated manipulation with diffusion-based factor graph. In *ICRA 2024 Workshop {\textemdash} Back to the Future: Robot Learning Going Probabilistic*, 2024. 1, 5, 16

Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021. 8

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024. 19

Maya Okawa, Ekdeep S Lubana, Robert Dick, and Hidenori Tanaka. Compositional abilities emerge multiplicatively: Exploring diffusion models on a synthetic task. *Advances in Neural Information Processing Systems*, 36, 2024. 2

Joseph Ortiz, Talfan Evans, and Andrew J. Davison. A visual introduction to gaussian belief propagation, 2021. URL https://arxiv.org/abs/2107.02308. 6

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. *The Eleventh International Conference on Learning Representations*, 2023. 2

William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748. 22

Yorai Shaoul, Itamar Mishani, Shivam Vats, Jiaoyang Li, and Maxim Likhachev. Multi-robot motion planning with diffusion models. *arXiv preprint arXiv:2410.03072*, 2024. 2

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015. 2

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=9_gsMA8MRKQ. 2

Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning (ICML)*, July 2023b. 2

James Thornton, Louis Bethune, Ruixiang Zhang, Arwen Bradley, Preetum Nakkiran, and Shuangfei Zhai. Composition and control with distilled energy diffusion models and sequential monte carlo. *arXiv preprint arXiv:2502.12786*, 2025. 2

Toshihide Ubukata, Jialong Li, and Kenji Tei. Diffusion model for planning: A systematic literature review. *arXiv preprint arXiv:2408.10266*, 2024. 2

Fu-Yun Wang, Wenshuo Chen, Guanglu Song, Han-Jia Ye, Yu Liu, and Hongsheng Li. Gen-l-video: Multi-text to long video generation via temporal co-denoising. *arXiv preprint arXiv:2305.18264*, 2023. 2

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. 2

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 19

Amber Xie, Oleh Rybkin, Dorsa Sadigh, and Chelsea Finn. Latent diffusion planning for imitation learning. *arXiv preprint arXiv:2504.16925*, 2025. 2

Zhengtong Xu, Qiang Qiu, and Yu She. Vilp: Imitation learning with latent video planning. *IEEE Robotics and Automation Letters*, 2025. 2

Lingxiao Yang, Shutong Ding, Yifan Cai, Jingyi Yu, Jingya Wang, and Ye Shi. Guidance with spherical gaussian constraint for conditional diffusion, 2024. URL https://arxiv.org/abs/2402.03201. 2, 3, 7

Zhutian Yang, Jiayuan Mao, Yilun Du, Jiajun Wu, Joshua B Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Compositional diffusion-based continuous constraint solvers. *arXiv preprint arXiv:2309.00966*, 2023. 2

Haotian Ye, Haowei Lin, Jiaqi Han, Minkai Xu, Sheng Liu, Yitao Liang, Jianzhu Ma, James Zou, and Stefano Ermon. Tfg: Unified training-free guidance for diffusion models, 2024. URL https://arxiv.org/abs/2409.15761. 2

Jaesik Yoon, Hyeonseo Cho, and Sungjin Ahn. Compositional monte carlo tree diffusion for extendable planning, 2025a. URL https://arxiv.org/abs/2510.21361. 24

Jaesik Yoon, Hyeonseo Cho, Doojin Baek, Yoshua Bengio, and Sungjin Ahn. Monte carlo tree diffusion for system 2 planning. *arXiv preprint arXiv:2502.07202*, 2025b. 2

Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model, 2023. URL https://arxiv.org/abs/2303.09833. 2

Jianrong Zhang, Hehe Fan, and Yi Yang. Energymogen: Compositional human motion generation with energy-based diffusion model in latent space. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 17592–17602, 2025. 2

Qinsheng Zhang, Jiaming Song, Xun Huang, Yongxin Chen, and Ming-Yu Liu. Diffcollage: Parallel generation of large content with diffusion models, 2023. URL https://arxiv.org/abs/2303.17076. 1, 2, 3, 5, 8, 16

Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. Madiff: Offline multi-agent learning with diffusion models. *Advances in Neural Information Processing Systems*, 37:4177–4206, 2024. 2

APPENDIX

A NOISY-BETHE GAP THEOREM

Theorem 1 (Noisy-Bethe Gap Theorem)**.** *Consider a linear chain $z = [u^1, u^2, u^3]$ with pairwise factors $[u^1, u^2]$ and $[u^2, u^3]$, where u^2 is the transition boundary variable. Assume the forward noising processes are $p(u_t^1, u_t^2 \,|\, u^1, u^2)$, $p(u_t^2, u_t^3 \,|\, u^2, u^3)$, and $p(u_t^2 \,|\, u^2)$. Let $a(u^2) = \int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2)\, du^1$, $b(u^2) = \int p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3)\, du^3$, $c(u^2) = p(u^2) p(u_t^2 \,|\, u^2)$, $Z = \int c(u^2)\, du^2$, and $q(u^2) = c(u^2)/Z$. Denote by $p(u_t^1, u_t^2, u_t^3)$ the true noisy distribution and by $\hat{p}(u_t^1, u_t^2, u_t^3)$ the estimator from Eq. 1. Then the gap between true distribution and estimated distribution is:*

$$\Delta \;=\; p(u_t^1, u_t^2, u_t^3) - \hat{p}(u_t^1, u_t^2, u_t^3) \;=\; Z\, \mathrm{Cov}_{u^2 \sim q}\Big[\tfrac{a}{c},\ \tfrac{b}{c}\Big]. \tag{8}$$

Proof. The true noisy distribution is:

$$\begin{aligned}
p(u_t^1, u_t^2, u_t^3) &= \int p(u_t^1, u_t^2, u_t^3, u^1, u^2, u^3)\, du^1 du^2 du^3 \\
&= \int p(u^1, u^2, u^3) p(u_t^1, u_t^2, u_t^3 | u^1, u^2, u^3)\, du^1 du^2 du^3 \\
&= \int \frac{p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2) p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3)}{p(u^2) p(u_t^2 | u^2)}\, du^1 du^2 du^3 \\
&= \int \frac{\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2) du^1 \int p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3) du^3}{p(u^2) p(u_t^2 | u^2)}\, du^2
\end{aligned} \tag{14}$$

The estimator used in (Zhang et al., 2023; Mishra et al., 2023; 2024) is:

$$\hat{p}(u_t^1, u_t^2, u_t^3) = \frac{\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2)\, du^1 du^2 \int p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3)\, du^2 du^3}{\int p(u^2) p(u_t^2 \,|\, u^2)\, du^2} \tag{15}$$

Define the left-factor message into u^2 as $a(u^2) = \int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2)\, du^1$, the right-factor message into u^2 as $b(u^2) = \int p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3)\, du^3$, and the local boundary evidence $c(u^2) = p(u^2) p(u_t^2 \,|\, u^2)$. Then $p(u_t^1, u_t^2, u_t^3) = \int \frac{ab}{c} du^2$ and $\hat{p}(u_t^1, u_t^2, u_t^3) = \frac{\int a\, du^2 \int b\, du^2}{\int c\, du^2}$.

Introduce a change of measure by setting $q(u^2) = \frac{c}{Z}$ with $Z = \int c\, du^2$. For the true distribution, we have:

$$\begin{aligned}
&p(u_t^1, u_t^2, u_t^3) \\
&= \int \frac{\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2) du^1 \int p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3) du^3}{p(u^2) p(u_t^2 | u^2)} \frac{p(u^2) p(u_t^2 | u^2)}{p(u^2) p(u_t^2 | u^2)} \frac{Z}{Z}\, du^2 \\
&= Z \int \frac{\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2) du^1}{p(u^2) p(u_t^2 | u^2)} \frac{\int p(u^2, u^3) p(u_t^2, u_t^3 \,|\, u^2, u^3) du^3}{p(u^2) p(u_t^2 | u^2)} \frac{p(u^2) p(u_t^2 | u^2)}{Z} du^2 \\
&= Z\, \mathbb{E}_{u \sim q(u^2)}[\tfrac{a}{c} \tfrac{b}{c}]
\end{aligned} \tag{16}$$

For the estimator, observe that:

$$\begin{aligned}
\int a(u^2)\, du^2 &= \int (\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2)\, du^1)\, du^2 \\
&= \int (\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2)\, du^1)\, \frac{p(u_2) p(u_t | u_2)}{p(u_2) p(u_t | u_2)} \frac{Z}{Z} du^2 \\
&= Z \int \frac{(\int p(u^1, u^2) p(u_t^1, u_t^2 \,|\, u^1, u^2)\, du^1)}{p(u_2) p(u_t | u_2)} \frac{p(u_2) p(u_t | u_2)}{Z} du^2 \\
&= Z\, \mathbb{E}_{u^2 \sim q(u^2)}[\tfrac{a}{c}]
\end{aligned} \tag{17}$$

By the same argument, $\int b(u^2)du^2 = Z\,\mathbb{E}_{u^2\sim q(u^2)}[\frac{b}{c}]$, therefore the estimated distribution:

$$\hat{p}(u_t^1, u_t^2, u_t^3) = Z\,\mathbb{E}_{u^2\sim q(u^2)}[\tfrac{a}{c}]\mathbb{E}_{u^2\sim q(u^2)}[\tfrac{b}{c}]. \tag{18}$$

Finally, the difference between the true and estimated distributions is:

$$\begin{aligned} \Delta &= p(u_t^1, u_t^2, u_t^3) - \hat{p}(u_t^1, u_t^2, u_t^3) \\ &= Z\,\mathbb{E}_{u\sim q(u^2)}[\tfrac{a}{c}\tfrac{b}{c}] - Z\,\mathbb{E}_{u^2\sim q(u^2)}[\tfrac{a}{c}]\mathbb{E}_{u^2\sim q(u^2)}[\tfrac{b}{c}] \\ &= Z\,\mathrm{Cov}[\tfrac{a}{c}, \tfrac{b}{c}]. \end{aligned} \tag{19}$$

This shows that the estimator departs from the true distribution by a covariance term under the reweighted boundary measure $q(u^2)$, scaled by Z. □

B Synchronous Message Passing

Theorem 2 (Synchronous Message Passing Constraint). *Let $x^{1:n} \in \mathbb{R}^{n\times tchw}$ denote the concatenated intermediate factors in a chain-structured factor graph with transition boundaries $\psi_{i-1,i}$. Given the start boundary $\psi_{s,1}$ and the goal boundary $\psi_{n,g}$, the joint constraints distribution over all intermediate factors is Gaussian:*

$$p_{\text{sync}}(x^{1:n} \,|\, s,\, g) \;\propto\; \exp(-\tfrac{1}{2}(x^{1:n})^T\Sigma^{-1}x^{1:n} \;+\; \eta^T x^{1:n}), \tag{10}$$

where $$\Sigma^{-1} = \begin{bmatrix} \frac{A_1^TA_1}{c_0} + \frac{B_1^TB_1}{c_1} & -\frac{B_1^TA_2}{c_1} & & \\ -\frac{A_2^TB_1}{c_1} & \frac{A_2^TA_2}{c_1} + \frac{B_2^TB_2}{c_2} & -\frac{B_2^TA_3}{c_2} & \\ & -\frac{A_3^TB_2}{c_2} & \frac{A_3^TA_3}{c_2} + \frac{B_3^TB_3}{c_3} & \\ & & & \ddots \end{bmatrix}, \quad \eta = \begin{bmatrix} \frac{A_1^Ts}{c_0} \\ 0 \\ 0 \\ \vdots \\ \frac{B_n^Tg}{c_n} \end{bmatrix}.$$

Proof. $p_{sync}(x_{1:n} \,|\, s,\, g)$ can be represented as a product of dependencies over all boundary variables:

$$p(x^{1:n} \,|\, s,\, g)_{sync} \;=\; \psi_{s,1}(s, x_1)\,\psi_{1,2}(x_1, x_2)\,\cdots\,\psi_{n-1}(x_{n-1}, x_n)\,\psi_{n,g}(x_n, g). \tag{20}$$

The aim of this equation is to express the joint distribution over all intermediate states, given initial and final state:

$$\begin{aligned} p(x^{1:n} \,|\, s,\, g) \;&=\; \psi_0(s, x_1)\,\psi_1(x_1, x_2)\,\cdots\,\psi_{n-1}(x_{n-1}, x_n)\,\psi_n(x_n, g) \\ &\propto\; \exp\Big(-\frac{1}{c_0}\left\|s - A_1x_1\right\|^2 - \frac{1}{c_1}\left\|B_1x_1 - A_2x_2\right\|^2 - \frac{1}{c_2}\left\|B_2x_2 - A_3x_3\right\|^2 \\ &\qquad -\;\cdots\;-\frac{1}{c_{n-1}}\left\|B_{n-1}x_{n-1} - A_nx_n\right\|^2 - \frac{1}{c_n}\left\|B_nx_n - g\right\|^2\Big) \\ &=\; \exp\Big(-\tfrac{1}{2}\,s^T\,\frac{I}{c_0}\,s \;+\; \tfrac{1}{2}\,s^T\,\frac{A_1}{c_0}\,x_1 \;+\; \tfrac{1}{2}\,x_1^T\,\frac{A_1}{c_0}\,s \;-\; \tfrac{1}{2}\,x_1^T\,\frac{A_1^TA_1}{c_0}\,x_1 \\ &\qquad -\tfrac{1}{2}\,x_1^T\,\frac{B_1^TB_1}{c_1}\,x_1 \;+\; \tfrac{1}{2}\,x_1^T\,\frac{B_1^TA_2}{c_1}\,x_2 \;+\; \tfrac{1}{2}\,x_2^T\,\frac{A_2^TB_1}{c_1}\,x_1 \;-\; \tfrac{1}{2}\,x_2^T\,\frac{A_2^TA_2}{c_1}\,x_2 \\ &\qquad -\tfrac{1}{2}\,x_2^T\,\frac{B_2^TB_2}{c_2}\,x_2 \;+\; \tfrac{1}{2}\,x_2^T\,\frac{B_2^TA_3}{c_2}\,x_3 \;+\; \tfrac{1}{2}\,x_3^T\,\frac{A_3^TB_2}{c_2}\,x_2 \;-\; \tfrac{1}{2}\,x_3^T\,\frac{A_3^TA_3}{c_2}\,x_3 \\ &\qquad -\tfrac{1}{2}\,x_3^T\,\frac{B_3^TB_3}{c_3}\,x_3 \;+\; \tfrac{1}{2}\,x_3^T\,\frac{B_3^TA_4}{c_3}\,x_4 \;+\; \tfrac{1}{2}\,x_4^T\,\frac{A_4^TB_3}{c_3}\,x_3 \;-\; \tfrac{1}{2}\,x_4^T\,\frac{A_4^TA_4}{c_3}\,x_4 \\ &\qquad\qquad\qquad\qquad\vdots \\ &\qquad -\tfrac{1}{2}\,x_n^T\,\frac{B_n^TB_n}{c_n}\,x_n \;+\; \tfrac{1}{2}\,x_n^T\,\frac{B_n^T}{c_n}\,g \;+\; \tfrac{1}{2}\,g^T\,\frac{B_n}{c_n}\,x_n \;-\; \tfrac{1}{2}\,g^T\,\frac{I}{c_n}\,g\Big) \\ &=\; \exp\big(-\frac{1}{2}x_{1:n}^T\Sigma^{-1}x_{1:n} \;+\; \eta^Tx_{1:n}\,\big), \end{aligned} \tag{21}$$

$$\text{where } \Sigma^{-1} = \begin{bmatrix} \frac{A_1^T A_1}{c_0} + \frac{B_1^T B_1}{c_1} & -\frac{B_1^T A_2}{c_1} & & \\ -\frac{A_2^T B_1}{c_1} & \frac{A_2^T A_2}{c_1} + \frac{B_2^T B_2}{c_2} & -\frac{B_2^T A_3}{c_2} & \\ & -\frac{A_3^T B_2}{c_2} & \frac{A_3^T A_3}{c_2} + \frac{B_3^T B_3}{c_3} & \\ & & & \ddots \end{bmatrix}, \eta = \begin{bmatrix} \frac{A_1^T s}{c_0} \\ 0 \\ 0 \\ \vdots \\ \frac{B_n^T g}{c_n} \end{bmatrix}.$$

C Baseline Implementations

To ensure clarity about how the prior baselines are instantiated in our setting, we briefly summarize the exact formulations of all baselines below. We'd like to clarify DiffCollage / GSC/ CompDiffuser are joint denoising over the full long-horizon trajectory, starting from a single long-horizon Gaussian.

Language-Conditioned Behavioral Cloning (LCBC). The policy uses a T5 text encoder to embed natural-language instructions. We concatenate the text embedding with image features extracted by a ResNet backbone, and feed the result into an MLP policy head. The model is trained in a supervised manner to predict a single action at each timestep, conditioned on both the language input and the current observation.

Language-Conditioned Diffusion Policy (LCDP). LCDP follows the same text encoding pipeline as LCBC with a T5 encoder, but replaces the MLP head with a Transformer-based policy head. The Transformer generates chunks of actions rather than single-step predictions, allowing multi-step reasoning conditioned on language and observations.

Goal-Conditioned Behavioral Cloning (GCBC). GCBC uses a ResNet backbone to encode both the current observation image and the goal image. The concatenated features are passed through an MLP policy head, which outputs a single action. This provides a goal-aware baseline without language conditioning.

Goal-Conditioned Diffusion Policy (GCDP). GCDP employs a ResNet backbone with a Transformer policy head, conditioned jointly on the current observation and the goal image. The model outputs action chunks, enabling multi-step planning toward the goal state.

DiffCollage / GSC. DiffCollage is a compositional test time generation method, and GSC refers to its adaptation for robotic planning. Given a start image and a goal image, the model directly generates an entire visual plan (a sequence of intermediate frames). It composes a long-horizon score following Eq. 2 and iteratively denoise the entire long trajectory. We then employ a separately trained inverse dynamics model to convert the visual plan into executable robot actions.

CompDiffuser. CompDiffuser is a trained joint-denoising baseline: it trains the short-horizon model to condition on its preceding and following noisy chunks. At test time, it subdivides the long trajectory into overlapping chunks, and denoises them jointly while conditioning each chunk on its neighbors. We then employ a separately trained inverse dynamics model to convert the visual plan into executable robot actions. CompDiffuser is SOTA joint-denoising baseline on OGBench 2D xy point maze navigation; however, its behavior on high-dimensional visual planning had not been evaluated prior to our experiments.

C.1 Implementation Details

For DiffCollage/GSC, we use the official codebase with recommended hyperparameters and exactly the same pretrained checkpoints as our method, since all methods are training-free in the same setup as ours. For CompDiffuser, we use the official codebase and match the model capacity to ours. And we select the checkpoint with the best validation loss. We intentionally disable replanning in order to evaluate the benefit coming from the algorithm itself, rather than from repeated replanning.

C.2 Sanity Test on OGBench

Env	**DiffCollage**	**CompDiffuser**	**Ours**
pointmaze-medium	100±0	100±0	100±0
pointmaze-large	100±0	100±0	100±0
pointmaze-giant	32±4	67±3	93±4

Table 4: **Results on OGBench.** Our reproduced results match the reported performance in CompDiffuser with only minor statistical variation.

To faithfully verify that our implementations are correct and reasonably tuned, we first evaluate on OGBench tasks, where CompDiffuser reports results for both DiffCollage/GSC and CompDiffuser. Our reproduced results match the reported performance with only minor statistical variation.

D Compositional Planning Tasks

D.1 Dataset and Simulation Setup

Assets. Our asset library combines 3D models and textures from ShapeNet (Chang et al., 2015) and RoboCasa (Nasiriany et al., 2024). We additionally apply simple high-quality texture(e.g., wood, plastic, metal finishes) to increase visual fidelity. All simulations are conducted in the SAPIEN engine (Xiang et al., 2020), which provides high-fidelity physics and rendering for robotic manipulation.

State and Action Space. The observation space consists solely of RGB images with resolution $256 \times 256 \times 3$, without access to privileged information such as depth or ground-truth states. The action space is parameterized by the end-effector position (3D Cartesian coordinates), orientation represented as a quaternion (4D), and a binary scalar controlling the gripper open/close state. During initialization, all object poses are randomized within a $0.2\,\mathrm{m}$ radius from their nominal positions, ensuring sufficient variability and out-of-distribution test cases.

Demonstrations. We provide 300 expert demonstrations for each of the N start–goal combinations across all scenes, resulting in thousands of trajectories spanning tool-use, drawer manipulation, cube rearrangement, and puzzle solving. Each demonstration is generated via scripted policies that guarantee feasibility and success. For the LCBC and LCDP baselines, we further annotate each demonstration with natural-language descriptions of the task. Tese annotations also help evaluate how well language-conditioned models can generalize across start–goal variations.

Success Condition. A rollout is considered successful if all objects reach their target configurations within a predefined spatial threshold.

D.2 Tool

The Tool scene (Figure 6) requires the robot to manipulate a tool in order to push the cube to the target location. Success is achieved when the cube reaches the designated target area within a fixed distance threshold. Direct manipulation is not possible, so the robot must use the provided tool to accomplish the task.

D.3 Drawer

The Drawer scene (Figure 7) requires the robot to manipulate drawers into closed states and use the brush to draw on the canvas. Depending on the start state, the brush may be located in different spaces, and the robot must go to the correct space and retrieve it. After grasping the brush, the drawer must often be closed again before drawing, to avoid collision between the arm and the drawer.

D.4 Cube

The Cube scene (Figure 8) requires the robot to manipulate multiple colored cubes, first arranging them into a prescribed order and then placing each cube into its designated goal region. This task evaluates the planner's ability to identify and distinguish between object colors, maintain the correct ordering, and execute precise placement into multiple targets.

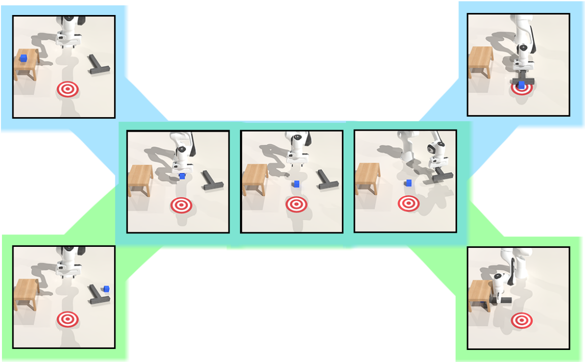

Figure 6: **Visualization of Tool Scene.**

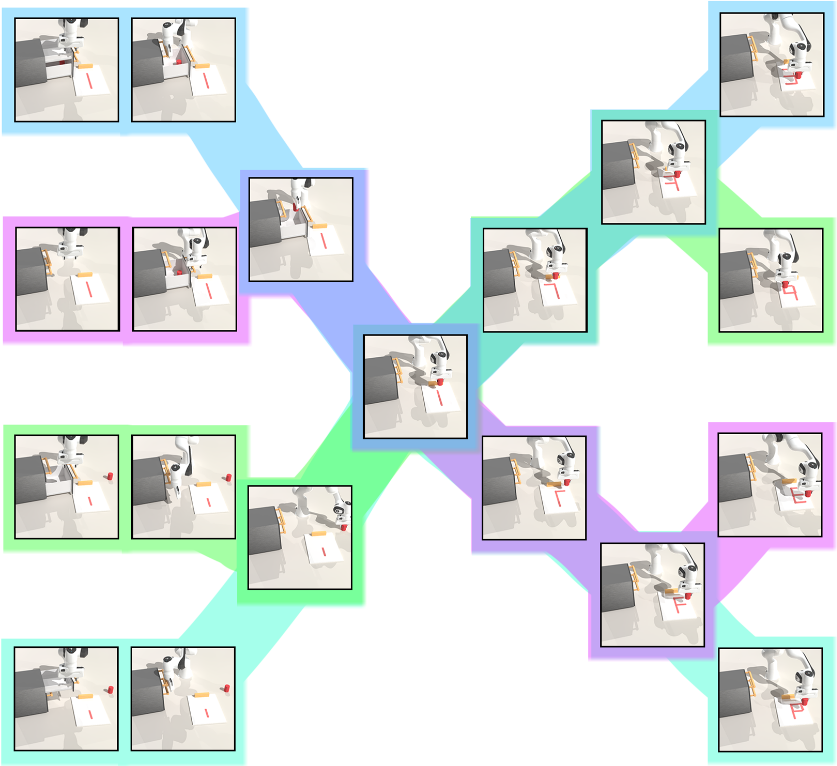

Figure 7: **Visualization of Drawer Scene.**

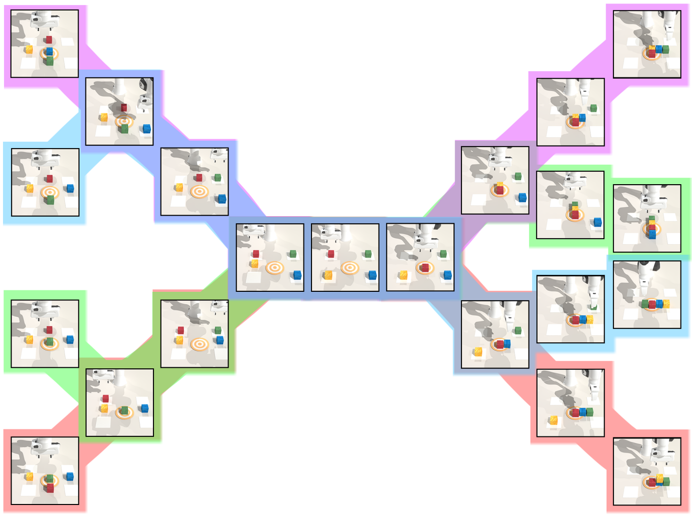

Figure 8: **Visualization of Cube Scene.**

D.5 PUZZLE

The Puzzle scene (Figure 9) poses the most challenging test of compositional planning. The robot must first arrange multiple colored blocks into a specific intermediate configuration, and then place them into distinct goal slots. This requires not only accurate object manipulation and ordering, but also the ability to chain together multiple sub-tasks that were only observed in isolation during demonstrations.

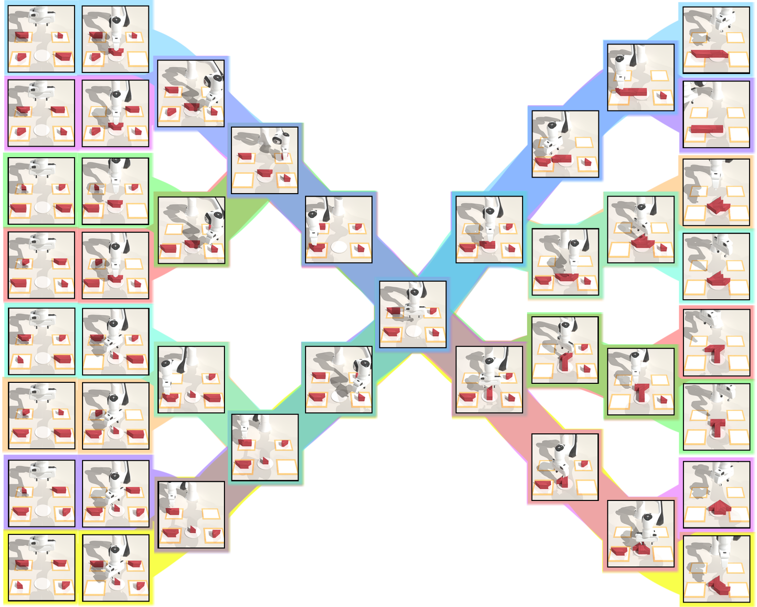

Figure 9: **Visualization of Puzzle Scene.**

E Implementation Details

Software. All experiments are conducted on Ubuntu 20.04.6 with Python 3.10 and PyTorch 2.2.1.

Training. Models are trained on NVIDIA H200 GPUs.

Deployment Hardware. For deployment, we use a single NVIDIA L40S GPU.

Model Inputs and Outputs. All observations are first encoded into tokens using the Cosmos tokenizer. We adopt DiT backbones for video generation, using DiT-L or DiT-XL (Peebles & Xie, 2023) depending on the scene. Video generation is performed entirely in the token space. For control, we employ a simple MLP-based inverse dynamics model, which predicts low-level actions conditioned on consecutive frames.

Hyperparameters. We report all hyperparameters used during both training and inference for full transparency. For fairness and reproducibility, we do not perform any hyperparameter search or use a learning rate scheduler; all experiments are conducted with fixed values throughout. This ensures that performance gains arise from the method itself rather than extensive hyperparameter tuning. We also keep hyperparameters consistent across different tasks and scenes, unless otherwise specified, to highlight the robustness of our approach.

Hyperparameter	**Value**
Diffusion Time Step	500
Batch Size	512
Optimizer	Adam
Learning Rate	1×10^{-4}
Iterations	1M
Discount Factor γ	0.6
Sampling Time Steps	300
Guidance Weight g_r	0.6

Table 5: **Relevant hyperparameters** used in our experiments.

F Deployment Time Study

F.1 Time Study & NFEs

All results in Table 6 are reported under the same setting of 300 DDIM steps. Compared to DiffCollage, our method incurs higher wall-clock time because it requires test-time backpropagation through the diffusion model in order to enforce consistency via message passing. This extra computation accounts for the increase in sampling time, but is necessary to achieve the significant gains in success rates reported in the main results. Our implementation is fully batched, so all factors require only a single forward pass at each diffusion steps. We follow the standard convention of counting only forward passes of the diffusion model as NFEs. At the same DDIM step count, our method requires twice the NFEs because it performs two forward passes per step. The backward update does not increase NFEs, but it does add a small amount of extra wall-clock time.

Scene	**# Models Composed**	**Sampling Time (s) ↓**	
		DiffCollage	**Ours**
Tool-Use	3	7.1	17.2
Drawer	5	18.9	30.4
Cube	5	21.1	31.7
Puzzle	6	30.4	61.8

Table 6: **Sampling time during deployment.** This is measured as the mean wall-clock time across all samples within a single scene.

Tool-Scene	NFE 300	NFE 400	NFE 500
IND	$1_{\pm 2}$	$0_{\pm 0}$	$3_{\pm 2}$
OOD	$0_{\pm 0}$	$2_{\pm 2}$	$1_{\pm 2}$

(a) DiffCollage/GSC

Tool-Scene	NFE 3000	NFE 4000	NFE 5000
IND	$1_{\pm 0}$	$2_{\pm 1}$	$5_{\pm 3}$
OOD	$1_{\pm 2}$	$3_{\pm 2}$	$2_{\pm 1}$

(b) DiffCollage+Repainting (10 RePaint steps)

Tool-Scene	NFE 300	NFE 400	NFE 500
IND	$60_{\pm 2}$	$62_{\pm 3}$	$64_{\pm 3}$
OOD	$51_{\pm 3}$	$52_{\pm 3}$	$49_{\pm 3}$

(c) CompDiffuser

Table 7: Success rates on tool-use scenes under different NFE budgets: (a) DiffCollage/GSC, (b) DiffCollage/GSC+Repainting with 10 RePaint steps, and (c) CompDiffuser. CompDiffuser consistently outperforms DiffCollage/GSC and DiffCollage+Repainting.

F.2 Hyperparameter Search over NFEs

On our high-dimensional tool scenes, we perform a hyperparameter search for DiffCollage/GSC and DiffCollage+Repainting(Anonymous (2025)) over the number of NFEs, allowing many more NFEs than our method. Even with substantially more NFEs than our sampler, the noisy-Bethe gap still leads to blurry and inconsistent video generations, which in turn keeps the success rates low. In these high-dimensional settings, CompDiffuser consistently outperforms DiffCollage/GSC and DiffCollage+Repainting. We report the corresponding results in table 7.

G Extension of Our Method

A key advantage of our training-free compositional framework is its plug-and-play flexibility. We demonstrate this with three extensions. (1) **Language conditioning:** replacing the goal image with a text prompt and adding a CLIP-style guidance loss yields performance comparable to goal-image conditioning. (2) **Physical constraints:** environment knowledge can be injected through simple penalty terms (e.g., collision loss), improving performance in challenging settings. (3) **Classical planning:** the asynchronous objective naturally supports tree-structured planning, and we provide a bidirectional RRT-style stitching variant for reference.

G.1 Extension to Language Conditioning

Conceptually, the boundary condition is an important way to extend the planning horizon, and in our training-free setting the goal image can be directly replaced by a text description. Concretely, we define a CLIP-style guidance loss L_{clip} that measures the semantic match between the generated goal frame at each denoising step and the goal text prompt, and we drop the original goal-image term in L_{sync} and L_{async}. In a preliminary goal-text setting, this variant achieves performance comparable to the goal-image version.

We build the language conditioning setting based on the Tool scene while the goal image is replaced with a goal text condition. In this setting, the text prompts correspond to two goal configurations: “the robot arm pushes the cube under the table” and “the robot arm pushes the cube over a red target.” We present the corresponding results in table 8.

Tool-Scene	**Goal Image**	**Goal Text**
IND	$97_{\pm 3}$	$95_{\pm 3}$
OOD	$96_{\pm 2}$	$96_{\pm 3}$

Table 8: Success rates for goal-image vs. goal-text boundary conditions on tool-use scenes. Goal-text guidance uses L_{clip} to enforce semantic match between the generated goal frame and the text description. We observe that language conditioning achieves performance comparable to goal-image conditioning.

G.2 Extension to Physical Constraints

Our method does not forcibly stitch short plans together: boundary consistency is imposed through a *soft* guidance penalty rather than a hard constraint. Our DSG-based guidance preserves the underlying data manifold, so if the demonstrations never exhibit behaviors such as "passing through a wall," the guided sampler has very little chance of generating such unrealistic trajectories.

Moreover, because our framework is fully training-free, we can incorporate additional physical constraints when available. For verifier design in the pointmaze domains, we use the ground-truth maze layout and compute a collision penalty from the position coordinates. If a point (x, y) is inside a maze wall box with center (c_x, c_y) and half-width d, we define the pointwise loss as the minimum distance to the four box walls:

$$L(x, y) = \min\big(x - (c_x - d),\, (c_x + d) - x,\, y - (c_y - d),\, (c_y + d) - y\big).$$

Summing this penalty along the trajectory yields $L_{\text{collision}}$, which we simply add to our guidance objective. As shown below, this explicit collision term further improves performance on the most challenging maze while leaving the easier settings unchanged. We present the corresponding results in table 9.

Env	**Ours**	**Ours + $L_{\text{collision}}$**
`pointmaze-medium`	$100_{\pm 0}$	$100_{\pm 0}$
`pointmaze-large`	$100_{\pm 0}$	$100_{\pm 0}$
`pointmaze-giant`	$93_{\pm 4}$	$97_{\pm 3}$

Table 9: Success rates on pointmaze environments with and without an explicit collision penalty $L_{\text{collision}}$. Incorporating $L_{\text{collision}}$ can further improve performance on more challenging long-horizon tasks.

G.3 Extension to Classical Planning

Extending our method to RRT-style, tree-structured planning is natural under the asynchronous objective. The objective decomposes into a forward term L_{fwd} that plans from the start and a backward term L_{bwd} that plans from the goal. We initialize a forward tree T_s rooted at s and a backward tree T_g rooted at g, and then iterate: (1) sample a batch of stitched paths using L_{fwd} to expand the forward tree, (2) sample a batch of stitched paths using L_{bwd} to expand the backward tree, and (3) connect T_s and T_g whenever newly added nodes fall below a distance threshold. On OGBench, this bidirectional stitching tree planner achieves a 100% success rate on `pointmaze-large`, and extending it to more complex settings will require additional exploration, which is beyond the scope of this work. Beyond classical sampling-based planners, our stitching framework naturally extends to Monte Carlo Tree Search as well, and recent work(Yoon et al. (2025a)) has already explored this direction.

Algorithm 2 Bidirectional RRT Compositional Tree Planner

```
Initialize forward tree $T_s \leftarrow \{s\}$, backward tree $T_g \leftarrow \{g\}$
while not connected do
    Sample a batch of stitched paths with $L_{\text{fwd}}$ to expand $T_s$ and collect new nodes $\mathcal{N}_s$
    Sample a batch of stitched paths with $L_{\text{bwd}}$ to expand $T_g$ and collect new nodes $\mathcal{N}_g$
    for each pair $(x, y)$ with $x \in \mathcal{N}_s, y \in T_g$ or $x \in T_s, y \in \mathcal{N}_g$ do
        if $\|x - y\| \leq \varepsilon$ then
            Connect $T_s$ and $T_g$ through $(x, y)$ and return the stitched path
        end if
    end for
end while
```

H Real Robot Experiment

H.1 Hardware Setup

The system setup is illustrated figure 10. We use a Franka Emika Panda robot controlled via a joint impedance controller running at 20 Hz for policy execution. For data collection, the robot is teleoperated using a Meta Quest 3 headset, with tracked Cartesian poses converted to joint configurations through inverse kinematics. RGB image are captured using an Intel RealSense D435 depth camera.

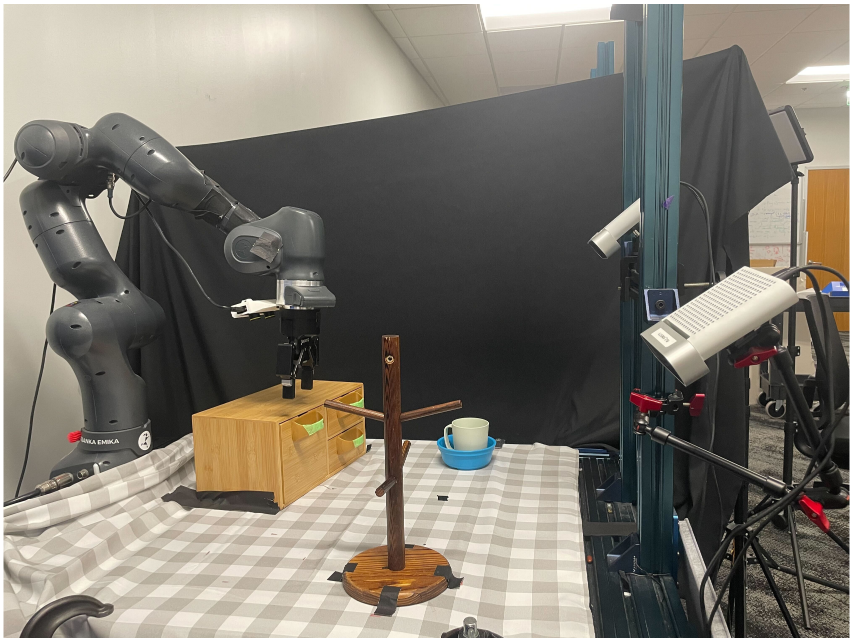

Figure 10: **Real World Robot Setup Overview.**

H.2 Qualitative Video Synthesis Results

H.2.1 In Distribution Evaluation

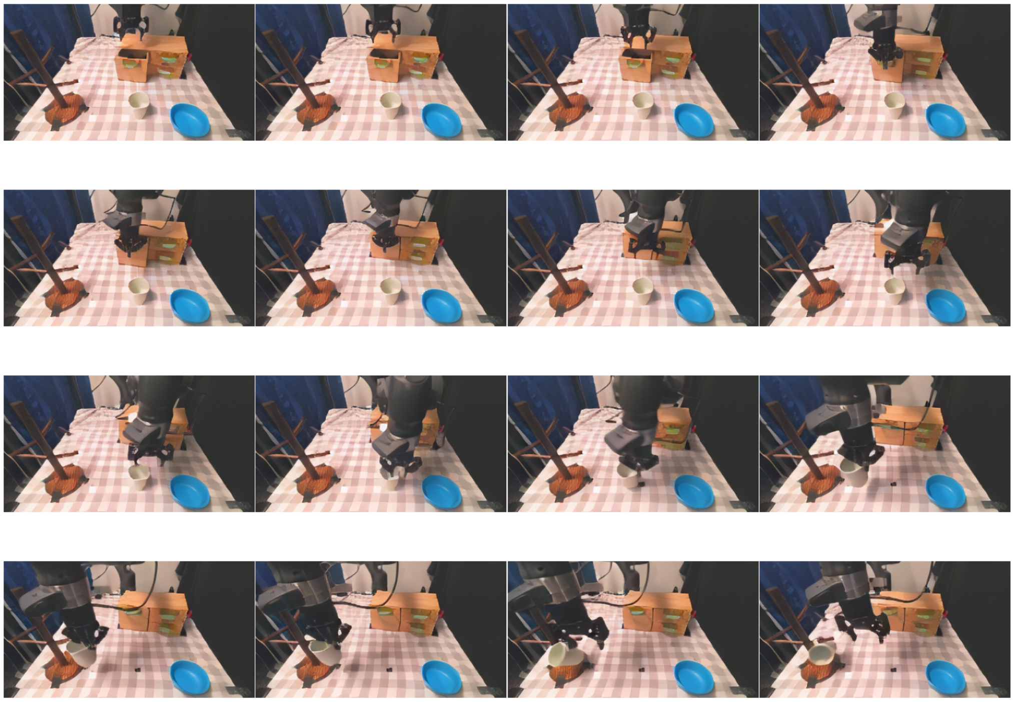

Figure 11: **Task1 Synthetic Plan.**

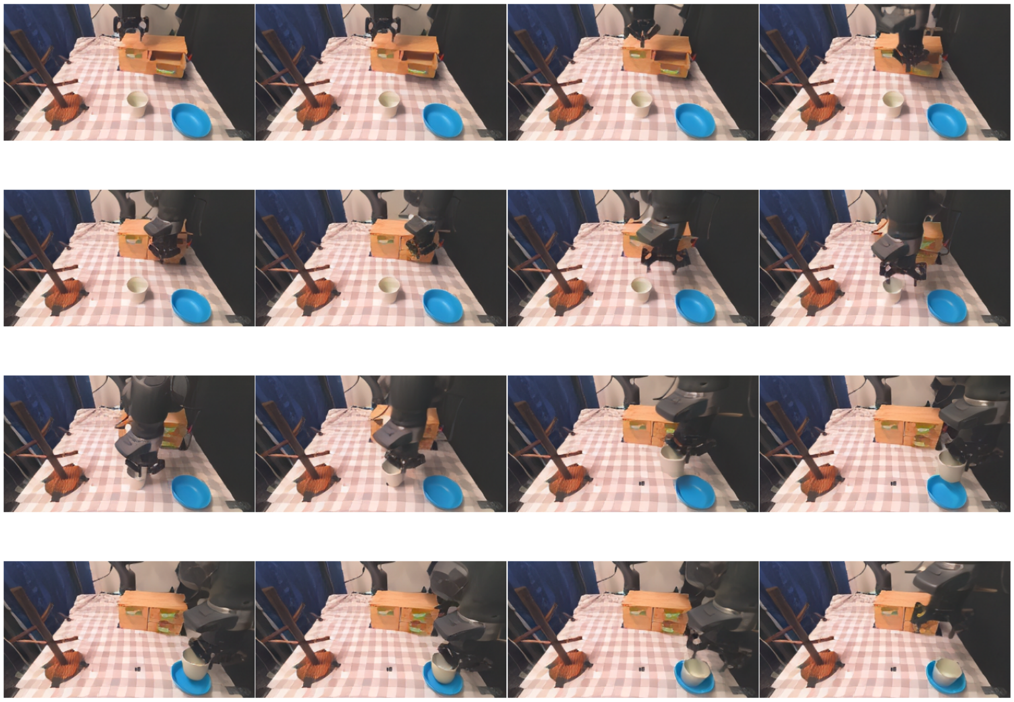

Figure 12: **Task2 Synthetic Plan.**

H.2.2 Out of Distribution Evaluation

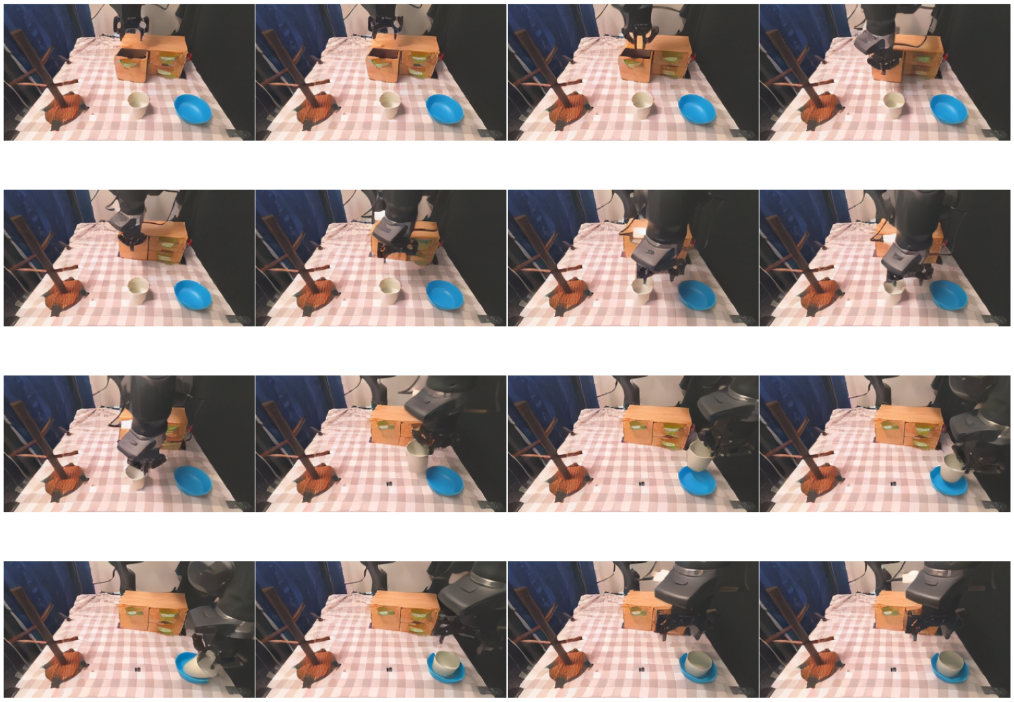

Figure 13: **Task3 Synthetic Plan.**

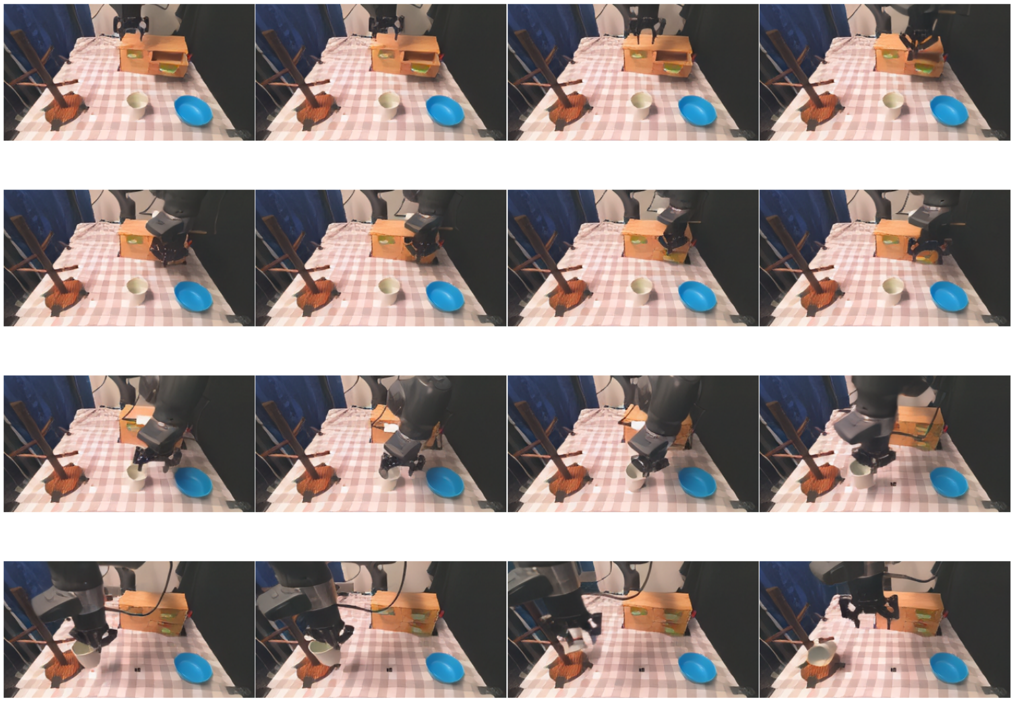

Figure 14: **Task4 Synthetic Plan.**

I Failure Case of Synchronous Message Passing

In our experiments, removing the discount factor does not cause plans to become globally incoherent, the generated videos remain smooth, and the intermediate states are generally consistent. The main failure mode is more subtle: the generated start and goal frames remain semantically correct but exhibit small spatial misalignments relative to the precise test-time requirement. This misalignment leads to plans that are coherent in motion but anchored to an incorrect spatial configuration.

For example, in our failure cases 1 and 2, the robot performs a reasonable picking motion for the hook, yet the entire sequence is centered around the wrong object location: the gray tool is consistently generated with a small but noticeable orientation offset from its true pose required by start. We hypothesize that without the discount factor, the guidance signals from the start and goal conditions are not sufficiently emphasized, allowing this spatial drift to persist throughout the whole plan.

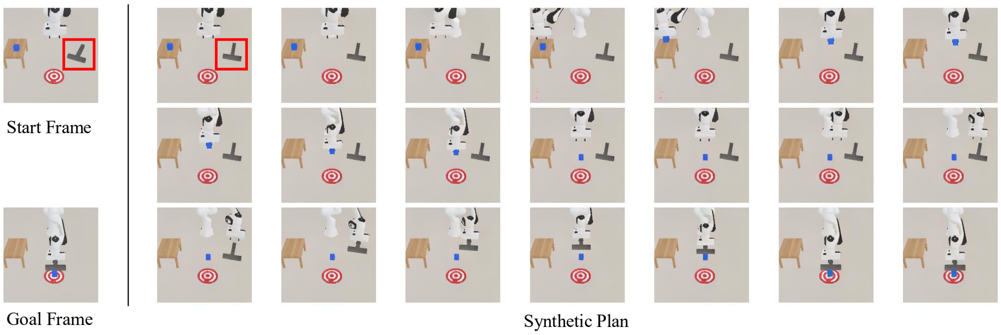

Figure 15: **Failure Mode of Synchronous Message Passing: Case 1.**

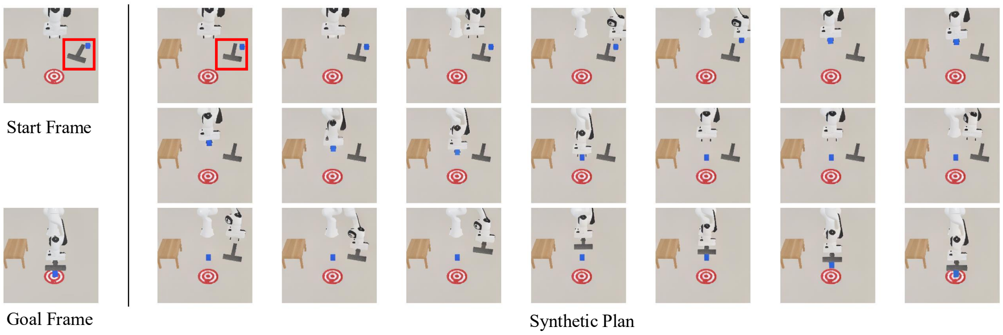

Figure 16: **Failure Mode of Synchronous Message Passing: Case 2.**