Hybrid Diffusion for Simultaneous Symbolic and Continuous Planning

Sigmund H. Høeg[1], Aksel Vaaler[1], Chaoqi Liu[2], Olav Egeland[1], and Yilun Du[3]

***Abstract*—Constructing robots to accomplish long-horizon tasks is a long-standing challenge within artificial intelligence. Approaches using generative methods, particularly Diffusion Models, have gained attention due to their ability to model continuous robotic trajectories for planning and control. However, we show that these models struggle with long-horizon tasks that involve complex decision-making and, in general, are prone to confusing different modes of behavior, leading to failure. To remedy this, we propose to augment continuous trajectory generation by simultaneously generating a high-level symbolic plan. We show that this requires a novel mix of discrete variable diffusion and continuous diffusion, which dramatically outperforms the baselines. In addition, we illustrate how this hybrid diffusion process enables flexible trajectory synthesis, allowing us to condition synthesized actions on partial and complete discrete conditions. Website: hybrid-diffusion.github.io**

I. Introduction

In the quest for general-purpose robotics, learning from demonstrations has proven a widely applicable paradigm. The task of imitation learning is mainly that of absorbing large amounts of demonstrations, including diverse behaviors. A performant technique for this task is to apply diffusion models [32, 17] for modeling robotic behavior. In addition to being stable to train, they allow for flexible guidance through conditioning and composition [18, 1, 20, 4]. They are, as a result, ubiquitous in a number of robotic systems, such as open-loop trajectory modelling [18, 9, 20], closed-loop action inference [7, 27, 8, 36], or as modules in composite systems [34, 25, 21]. However, diffusion models often struggle to form long-horizon, non-smooth plans, which restricts them to modeling dense trajectories in Cartesian space [1]. This limits their ability to do long-horizon decision-making tasks. A motivating example is shown in Figure 1, where a trajectory-level diffusion model is used to model robotic trajectories for sorting three blocks. In this example, the sampled trajectories do not lead to the blocks being sorted, despite the demonstrations always terminating in a sorted state.

In classical motion planning, directly constructing a long-horizon continuous motion plan is typically infeasible, as the space of possibilities increases exponentially with the temporal horizon. Therefore, Task-and-Motion Planning (TAMP) methods typically exploit the connection between symbolic and continuous motion plans [11] to simplify and reduce the overall size of the search space. Symbolic planners construct symbolic abstracted plans that transfer the system to the goal state, whereas continuous motion planning aims to find corresponding motion plans that correspond to this symbolic plan [12]. Inspired by this insight, we aim to answer the question *Can we endow diffusion models with a notion of symbolic planning, to improve long-horizon planning performance?*

We find that this is indeed possible and advantageous, and propose Hybrid Diffusion Planner (`HDP`) which simultaneously couples both continuous and symbolic plan generation. Using a joint objective in constructing symbolic plans through masked diffusion [30, 2] and continuous plans through continuous diffusion [17], it effectively combines the tasks of symbolic and continuous motion planning. Surprisingly, we find that introducing the joint task of symbolic planning boosts performance for continuous planning dramatically (Figure 1), enabling models to reason in both continuous and discrete modalities. Furthermore, it offers transparency, as the model outputs a corresponding discrete plan that describes the continuous motion.

In addition, `HDP` enables flexible conditional sampling at inference. By fixing a partial or complete symbolic plan through inpainting, `HDP` can generate a continuous plan that satisfies the specified constraints. Simultaneously, by fixing a partial or complete continuous plan, we can infer a corresponding symbolic interpretation of the continuous plan. Such flexible conditioning allows `HDP` to be easily controlled and used for diverse tasks outside of explicit plan generation.

Overall, our contributions are threefold: **(1)** We introduce Hybrid Diffusion Planner, a novel diffusion-based planner that uses a coupled discrete and continuous diffusion process for generating both symbolic and continuous motion plans. **(2)** We illustrate how such joint generation enables `HDP` to scale to tasks of increasing complexity substantially better than baseline approaches that directly generate continuous actions. **(3)** We empirically illustrate the efficacy of `HDP` on simulated and real robotics tasks and demonstrate `HDP`'s flexible conditioning capabilities.

II. Related Work

Long-horizon Planning. Our work is related to the problem of Task-and-Motion Planning (TAMP) [11], which simultaneously integrates both symbolic and discrete plans for tasks in long-horizon manipulation. While TAMP methods [13, 12] and methods incorporating TAMP planners [37, 23, 10] aim to solve these tasks, they have some face limitations due to

[1]Department of Mechanical and Industrial Engineering, Norwegian University of Science and Technology (NTNU) {sigmund.hoeg, aksel.vaaler,olav.egeland}@ntnu.no

[2] University of Illinois Urbana-ChampaignUniversity of Illinois Urbana-Champaign, chaoqil2@illinois.edu

[3] Harvard University, ydu@seas.harvard.edu

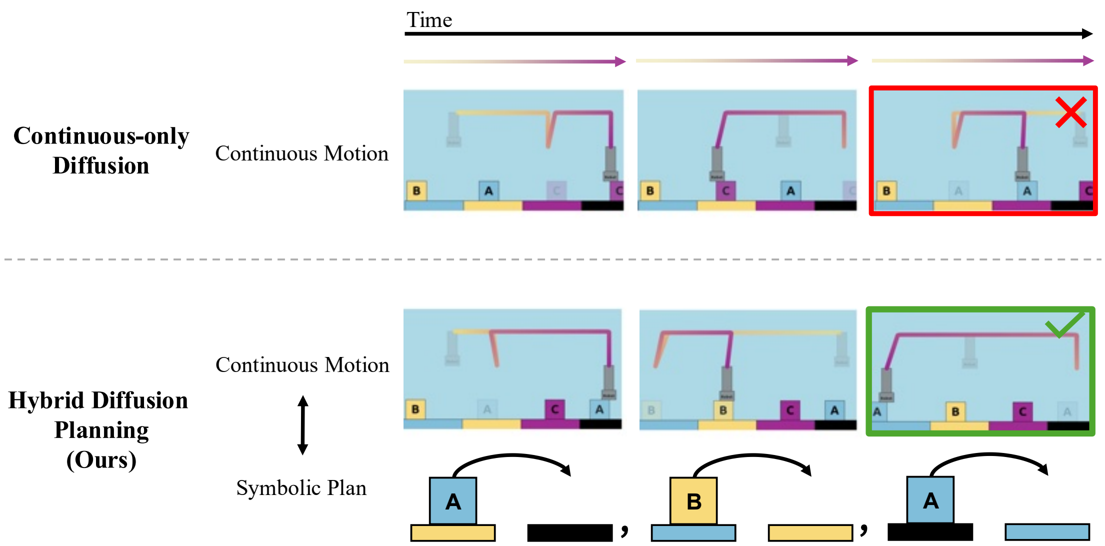


Fig. 1: **Hybrid Diffusion Planning.** Continuous diffusion models struggle when constructing plans for long-horizon decision-making tasks, such as this task of alphabetically sorting three blocks (top row). Our method, Hybrid Diffusion Planner, jointly constructs symbolic and discrete plans, enabling more robust performance over long-horizon tasks (bottom row).

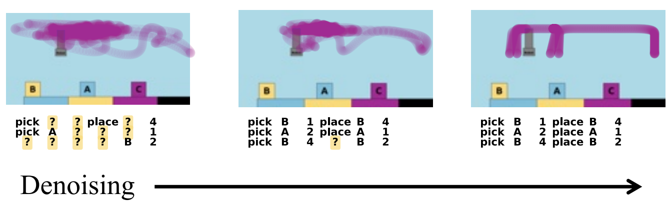


Fig. 2: **Denoising process** of discrete and continuous plans.

relying on a symbolic planner. These planners require full observability of the environment state, which is difficult in real-life environments. In addition, specifying the goal state is not a trivial task. They also assume that the available actions, with associated preconditions and effects, are known in advance. This limits their flexibility and application to real-life systems, and they require significant engineering for each new task considered. In contrast, `HDP` learns long-horizon behavior directly from data and is directly applicable to new tasks. While works such as Transporter Networks and derivatives [35, 31] are able to perform long-horizon tasks, they are restricted to pick-place tasks, while `HDP` lays no restrictions on the action modality.

Planning with Diffusion Models. Several works have shown that Diffusion Models excel at modelling distributions over trajectories [18, 26], with Diffuser by Janner et al. [18] as a seminal work showing their application to planning. Beneficial for planning is diffusion models' flexibility during sampling, such as inpainting [18], composition with auxiliary cost functions [4, 29, 20], and classifier-free guidance [16, 1]. Because of their capabilities, they have been widely applied in robotic systems [33]. However, these works only consider diffusion models for continuous motion planning, while our method also models the symbolic plan using diffusion.

III. Hybrid Diffusion Planning

We are interested in solving the task of long-horizon planning of robotic motion given demonstration data. Given an initial observation of the environment $\mathbf{O}$, the planner is tasked to predict a feasible continuous plan over a large number of time steps T, where our overall goal is to generate a action trajectory $\mathbf{A}_c \in R^{T \times D_a}$ that results in the robot completing the task.

We present Hybrid Diffusion Planner (`HDP`) for solving this task. To enable effective planning over very long time hori-

Algorithm 1 Hybrid Diffusion Training

Require: Dataset $\mathcal{D}$, Denoiser $D_\theta(\cdot)$

while not converged **do**
- Sample $(\mathbf{A}_c, \mathbf{A}_d, \mathbf{O}) \sim \mathcal{D}$
- Sample diffusion steps $k_c, k_d \sim \mathcal{U}[0, 1]$
- $\mathbf{A}_c^{k_c} \sim q_{\text{DDPM}}(\mathbf{A}_c^{k_c} | \mathbf{A}_c)$
- $\mathbf{A}_d^{k_d} \sim q_{\text{MD4}}(\mathbf{A}_d^{k_d} | \mathbf{A}_d)$
- $(\hat{\epsilon}, \hat{\mu}) \leftarrow D_\theta(\mathbf{A}_c^{k_c}, \mathbf{A}_d^{k_d}, k_c, k_d, \mathbf{O})$
- $\mathcal{L} \leftarrow \mathcal{L}_{\text{DDPM}} + \lambda \mathcal{L}_{\text{MD4}}$
- $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$

end while

return $D_\theta(\cdot)$

Algorithm 2 Hybrid Diffusion Planning

Require: Trained denoiser D_θ, Observation $\mathbf{O}$, Planning horizons h_c, h_d

$\mathbf{A}_c \leftarrow \mathcal{N}(\mathbf{0}_{h_c}, \mathbf{I}_{h_c})$

$\mathbf{A}_d \leftarrow [e_m]_{h_d}$

for each diffusion step **do**
- $(\hat{\epsilon}, \hat{\mu}) \leftarrow D_\theta(\mathbf{A}_c, \mathbf{A}_d, \mathbf{O}, \mathbf{k}, \mathbf{k})$
- $\mathbf{A}_c \sim q_{\text{DDPM}}(\mathbf{A}_c^{k-1} | \mathbf{A}_c^k, \hat{\epsilon})$
- $\mathbf{A}_d \sim q_{\text{MD4}}(\mathbf{A}_d^{k-1} | \mathbf{A}_d^k, \hat{\mu})$

end for

return $\mathbf{A}_c, \mathbf{A}_d$

zons, HDP simultaneously predicts both a continuous action trajectory $\mathbf{A}_c$ as well as a symbolic sequence of actions $\mathbf{A}_d$, by modeling the joint distribution over continuous and discrete plans using two coupled diffusion processes. We first describe modeling the continuous plan and discrete plan separately with diffusion in Section III-A and III-B. We then introduce our hybrid diffusion procedure, HDP, which jointly models both discrete and continuous planning simultaneously, enabling us to effectively solve long-horizon planning tasks.

A. Modeling the Continuous Plan with Continuous Variable Diffusion

For modeling continuous action plans, HDP uses a continuous diffusion process. Specifically, we apply DDPM [17] for trajectory generation, following Diffuser from Janner et al. [18]. During training, a continuous trajectory $\mathbf{A}_c$ is added noise with a magnitude proportional to the diffusion step k

$$q_{\text{DDPM}}(\mathbf{A}_c^k|\mathbf{A}_c^0) = \mathcal{N}(\mathbf{A}_c^k; \sqrt{\bar{\alpha}_k}\mathbf{A}_c^0, (1-\bar{\alpha}_k)\mathbf{I}), \tag{1}$$

where α_k given by the noise schedule determines the signal-to-noise ratio for a given diffusion step. The model is tasked to predict the noise component ϵ given a noisy sample

$$\mathcal{L}_{\text{DDPM}} = MSE(\epsilon, \hat{\epsilon}). \tag{2}$$

During sampling, the trained model outputs the noise component ϵ, and x^{k-1} is sampled from

$$q_{\text{DDPM}}(\mathbf{A}_c^{k-1}|\mathbf{A}_c^k, \epsilon) = \mathcal{N}(\mathbf{A}_c^{k-1}; \mu(\mathbf{A}_c^k, k, \epsilon), \sigma_k^2\mathbf{I}), \tag{3}$$

where $\mu(\mathbf{A}_c^k, k) = \frac{1}{\sqrt{\alpha_k}}\left(\mathbf{A}_c^k - \xi_k\epsilon\right)$ is the predicted mean with ξ_k given by the diffusion schedule.

B. Modelling the Symbolic Plans with Discrete Variable Diffusion

For modeling the discrete action plan, HDP deploys MD4 [30], which offers a simple and performant framework for diffusion of discrete variables. This is done through *masked diffusion* [30, 2, 3], where the forward diffusion process is defined such that each token is masked with an increasing probability when moving along the diffusion axis. This is done by representing each action $\mathbf{A}_{d,i}$ in the discrete action sequence as one-hot encoded variables with $m+1$ possible states, where the last state corresponds to a masked state e_m. The transition matrix $\bar{Q}(k)$ transfers tokens to this absorbing masked state:

$$q_{\text{MD4}}(\mathbf{A}_d^k|\mathbf{A}_d^0) = \text{Cat}(\mathbf{A}_d^k; \bar{Q}(k)^\top\mathbf{A}_d^0). \tag{4}$$

Here, $\text{Cat}(x;p)$ denotes a categorical distribution where p is the vector of probabilities and the transition matrix is $\bar{Q}(k) = \alpha_k I + (1-\alpha_k)\mathbf{1}e_m^\top$, which places increasing weight on the masked class at higher diffusion steps. As in continuous variable diffusion, α_k is given by the diffusion schedule.

The learned reverse process is parameterized with a network predicting logits over all possible discrete actions, $\hat{\mu} = \mu_\theta(\mathbf{A}_d^k, k) \in R^{m+1}$, where the probability of the masked class is set to 0. During training, the model is trained with a cross-entropy loss over the masked tokens in a partially-masked sequence

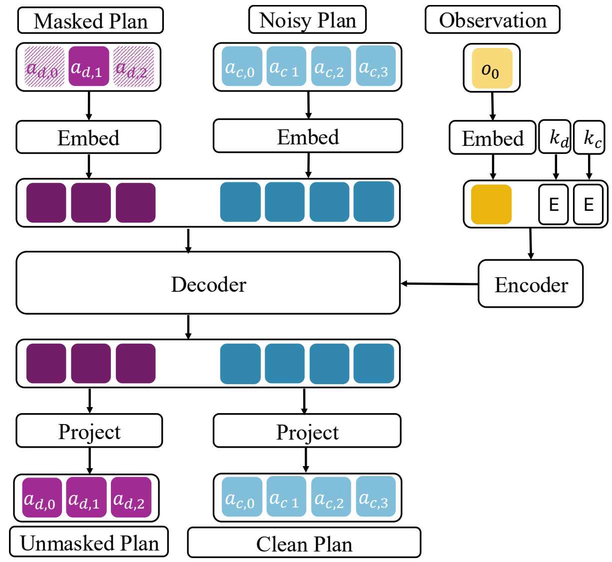


Fig. 3: **Architecture** for hybrid denoising of the symbolic and continuous plan.

$$\mathcal{L}_{\text{MD4}} = \sum_{i:\mathbf{A}_{d,i}^k=m} w_k \mathcal{L}_{\text{cross-entropy}}(\hat{\mu}, \mathbf{A}_{d,i}^0), \tag{5}$$

where w_k is a weighting term given by the diffusion schedule.

At sampling time, the sequence is instantiated as a fully masked sequence, and the tokens will be sampled from the predicted categorical distribution over all tokens in the vocabulary

$$q_{\text{MD4}}(\mathbf{A}_d^{k-1}|\mathbf{A}_d^k, \hat{\mu}) = \text{Cat}(\mathbf{A}_d^{k-1}, \bar{R}^\top\mathbf{A}_d^k), \tag{6}$$

$$\text{where } \bar{R} = I + \gamma_k e_m(\hat{\mu} - e_m), \tag{7}$$

where γ_k is given by the diffusion schedule. Intuitively, a sample will, with a given probability, transfer from the masked class to a sample from the predicted categorical distribution, at each reverse step. The diffusion schedule is designed to reveal tokens with increasing probability when moving backward along the diffusion axis [30].

C. Hybrid Diffusion Planning: Jointly Modeling Continuous and Discrete Plans

Next, we discuss Hybrid Diffusion Planner (HDP), which constructs a hybrid diffusion process jointly over continuous plan $\mathbf{A}_c$ and discrete plan $\mathbf{A}_d$.

Training. To jointly learn a diffusion process over a paired continuous and discrete plans $\mathbf{A}_c$ and $\mathbf{A}_d$, we corrupt each plan with independently sampled noise levels k_c and k_d to form noisy trajectories $\mathbf{A}_c^{k_c}$, $\mathbf{A}_d^{k_d}$. We add noise to each modality in separate ways: for the continuous plan, we add continuous noise to each element following Equation 1, and for the discrete plan, we mask out tokens following Equation 4.

Given both corrupted plans, the denoiser is tasked with reversing the corruption of both plans, predicting the continuous

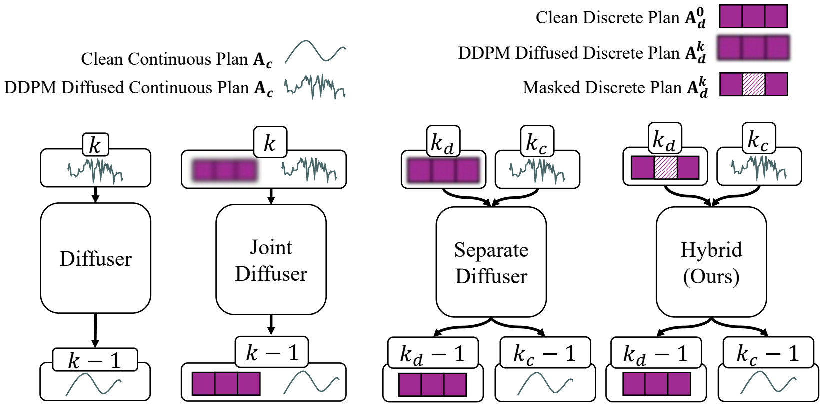


Fig. 4: **Denoiser designs.**In contrast to baselines, we model the discrete and continuous plan jointly, using masked diffusion for the symbolic plan.

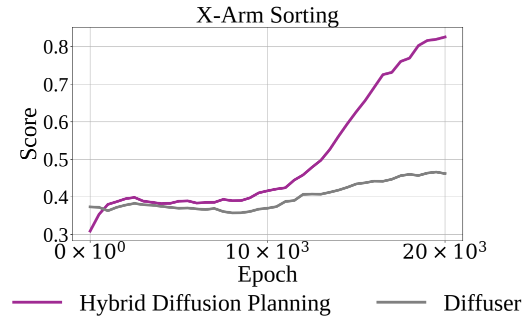


Fig. 5: **Performance over time.** `HDP` quickly learns the planning task compared to Diffuser.

noise component of $\mathbf{A}_c^{k_c}$ and the unmasking probabilities of $\mathbf{A}_d^{k_d}$ *simultaneously* using two separate heads

$$(\hat{\epsilon}, \hat{\mu}) = D_\theta(\mathbf{A}_c^{k_c}, \mathbf{A}_d^{k_d}, \mathbf{O}, k_c, k_d). \tag{8}$$

The above formulation enables the denoiser to have access to both noisy discrete and continuous plans and leverage information across both plans to accurately denoise $\mathbf{A}_c^{k_c}$ and $\mathbf{A}_d^{k_d}$.

To train the denoising network, we use a weighted sum of a continuous denoising loss on $\mathbf{A}_c^{k_c}$ (Equation 2) and discrete denoising loss on $\mathbf{A}_d^{k_d}$ (Equation 5) on both modalities

$$\mathcal{L} = \mathcal{L}_{\text{DDPM}} + \lambda \mathcal{L}_{\text{MD4}}, \tag{9}$$

where we set $\lambda = \frac{1}{30}$ to balance out the magnitude of the losses. An overview of the training procedure is outlined in Algorithm 1.

We train with independently sampled levels of corruption k_c and k_d. Similar to [6], this enables `HDP` to more accurately learn the correspondence between plans. For example, if the discrete plan is fully unmasked ($k_d = 0$) during a training iteration, the denoiser can learn to exploit the information in the discrete sequence when denoising the continuous trajectory. In contrast, when the discrete plan is fully masked $k_d = K_d$, the denoiser can learn to denoise the continuous trajectory unconditionally without masked information. In addition, independent noise levels further enable flexible sampling, which we discuss further next.

Sampling. Training `HDP` on independently sampled noise levels k_c and k_d enables a variety of different sampling techniques for generating plans from the joint distribution $p_\theta(\mathbf{A}_c, \mathbf{A}_d)$, depending on the order in which we denoise $\mathbf{A}_c^{k_c}$ and $\mathbf{A}_d^{k_d}$. For instance, after initializing $\mathbf{A}_c^{K_c}$ to Gaussian noise and $\mathbf{A}_d^{K_d}$ to a fully masked vector, we can first construct a clean discrete plan $\mathbf{A}_d^0$ (sampling from Equation 7), before refining $\mathbf{A}_c^{K_c}$ (sampling from Equation 3), or alternatively first construct a clean continuous plan $\mathbf{A}_c^0$ before refining the discrete plan $\mathbf{A}_d^{K_d}$.

One natural sampling technique is to iteratively denoise both plans simultaneously, allowing intermediate plan generation across both modalities to inform each other. Assuming an equal number of denoising steps for each process, both plans will be produced jointly in $N = K_d = K_c$ denoising steps. We outline this in Algorithm 2, and our experimental results in Section IV show that this procedure works well in practice.

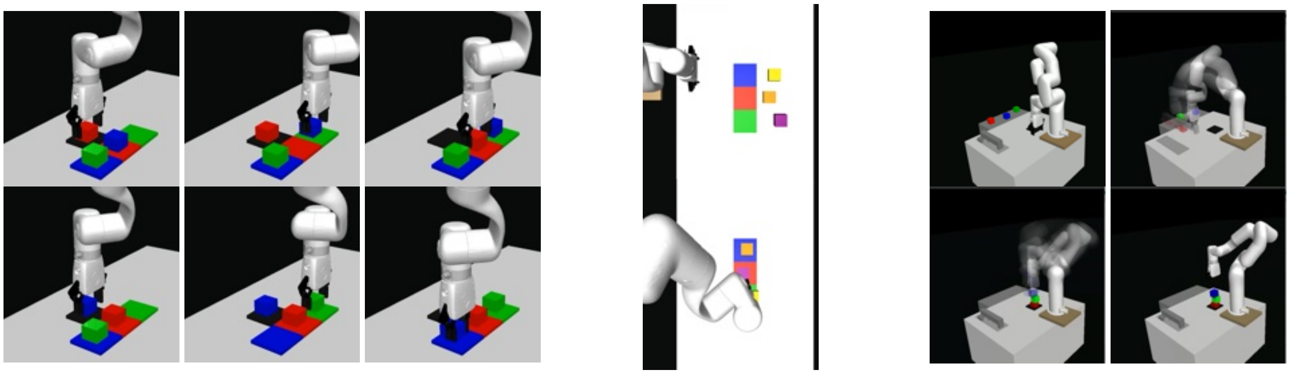
In addition, we can further modify the sampling procedure to sample from the conditional distributions, $p_\theta(\mathbf{A}_d | \mathbf{A}_c)$ given a specified continuous plan $\mathbf{A}_c$ or $p_\theta(\mathbf{A}_c | \mathbf{A}_d)$ given a specified discrete plan $\mathbf{A}_c$. To do this, we can pass the conditioned plan $\mathbf{A}_c$ or $\mathbf{A}_c$ with noise level 0 to the denoiser, and run sampling on the other plan modality. In addition, we can similarly condition on partially clean or unmasked plans. We show how this can be used in practice in Section IV-C.

Architecture. We base our architecture on the GPT-style transformer architecture used by Chi et al. [7], where we modify the architecture to take in the discrete plan along with its diffusion step. To do this, we first embed the discrete plan before concatenating it to the continuous plan, and passing the result to the decoder, where it is processed along with the encoded observations. The architecture is illustrated in Fig. 3.

IV. Experimental Evaluation

In our experiments, we measure the performance of `HDP` on long-horizon decision-making tasks and compare it to baselines. We evaluate our method on robotic manipulation tasks, including both simulated and real tasks, in Section IV-A. In subsection IV-B, we systematically evaluate the robustness of each method when faced with increasing task complexity. Finally, we demonstrate the diverse sampling capabilities of `HDP` in IV-C.

To properly assess the capabilities of each method, we develop a set of simulated and real benchmarks with long-horizon tasks that require both precision and complex decision-making, in contrast to existing benchmarks for robotic imitation learning [22], which often focus on evaluating policies for relatively short-horizon tasks. Benchmarks focusing on sequential task execution, such as CALVIN [24], test whether the policy can perform an arbitrary sequence of skills. In this setup, the policy is given a predetermined task sequence, which differs from our setup, where the planner itself must infer a sequence of actions to reach a goal state.



(a) 3 Block Sorting Task (b) Arrange Block Task (c) Tool-Use Task

Fig. 6: **Robotic Manipulation Tasks.**

We benchmark our method against Diffuser [18], which only models the continuous trajectory, implemented using the transformer architecture from Chi et al. [7] without causal masking. To identify essential components when modeling both plan modalities, we additionally form two baselines that explicitly model both plan modalities:

- **Joint Diffuser** Represents the symbolic plan as a continuous sequence and concatenates it to the motion plan, resulting in a single variable. These are always joined throughout the DDPM diffusion process, meaning that the architecture accepts only a single diffusion step, k. This represents an incremental change from the Diffuser baseline by providing the symbolic plan during training.
- **Separate Diffuser** Constructed to measure the effect of modeling the continuous and symbolic plan with two separate, independent DDPM diffusion processes. This results in the model taking different noise-corrupted symbolic and continuous plans, which can differ in the level of noise corruption.

An overview of all considered methods is shown in Figure 4.

A. Robotic Manipulation Experiments

To confirm that our method can tackle action spaces of high dimensions, we evaluate `HDP` on two tasks using an X-Arm robotic manipulator simulated in a MuJoCo environment, as well as on a real-world setup using a Franka Emika Panda robotic arm.

X-Arm Sorting. Three blocks are initialized in three slots on the table in random order, and the task is completed when all the blocks are sorted alphabetically, each in its own slot. The reward is given by the number of blocks in their correct place, with 100% indicating fully sorted. To ensure complete correspondence between the discrete and continuous plans, we collect 200 demonstrations using a scripted planner based on an in-place sorting algorithm. Using Cycle sort [15], the demonstrator swaps two blocks using an auxiliary slot as temporary storage. We concurrently log the corresponding discrete sequences using a vocabulary consisting of *Block Identifiers:* $\{A, B, C\}$, *Actions:* $\{\text{Pick up}, \text{Place}\}$, and *Slot IDs:* $\{\text{Slot } 1, \ldots \text{Slot } 4\}$. In addition to determining a valid symbolic plan, the models must construct a kinematically feasible continuous plan, as the initial position of the blocks is randomized within each slot. This requires the planner to create a motion that picks up the block at the correct location within each slot.

Method	X-Arm Sorting	Arrange Blocks	Tool Use
Diffuser	46%	67%	38%
Joint Diffuser	41%	61%	48%
Separate Diffuser	38%	62%	43%
Hybrid (Ours)	**83%**	**74%**	**60%**

TABLE I: **Average reward** over three seeds for different methods.

Arrange Blocks. 3 blocks are placed randomly at the table, and the planner is tasked to place the blocks in each slot. The demonstrations are collected using a scripted planner that randomly matches blocks to slots, creating highly multimodal demonstrations.

Tool-use experiment. To further illustrate the multi-step performance of `HDP`, we construct a task where the robot must use a "hook" tool to reach blocks outside of its workspace, and then stack them. This task extends beyond simple pick-and-place operations, as the robot must utilize the tool to pull the blocks into its workspace. The scripted expert demonstrates multimodal behaviour by randomly selecting to drag blocks into the workspace or stack blocks that have already been pulled.

Real-world Sorting. We also evaluate the models on a real-life version of the 3 Block Sorting task. The blocks are labeled with ArUco markers [14, 28], and their position are tracked using RGB-D images from a RealSense camera. We evaluate the methods with 10 trials, corresponding to 10 different seeds. All methods are evaluated on the same permutation, which is guaranteed by the operator shuffling the block according to a permutation specified by the seed. An episode is successful if the blocks are sorted by the end of the episode, and the operator never has to intervene, which happens before collisions. We use the Deoxys [38] framework with an operational-space controller, leading to compliant behavior when colliding. The results are presented in Fig. 8.

We show the results for the simulated benchmark in Figure 6. Our method dramatically outperforms the Diffuser baselines

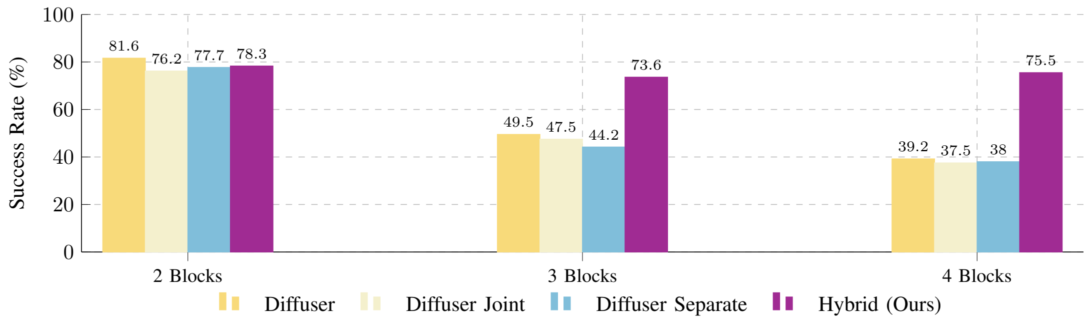


Fig. 7: **Performance on a sorting task with increasing number of blocks.** We report average results over 3 seeds, over the last 10 checkpoints, on 50 evaluations for all methods.

Method	Reward
Diffuser	20%
Joint Diffuser	10%
Separate Diffuser	0%
Hybrid (Ours)	**70%**

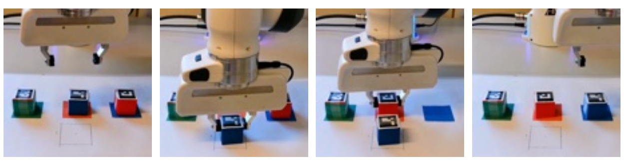


Fig. 8: **Real-world sorting task.** *Top:* Performance of different methods averaged over 10 trials. *Bottom:* Rollout of HDP completing the sorting task.

on the *X-Arm Sorting* task, showing that HDP is performant for long-horizon robotic tasks. Even for the less complex *Arrange Blocks*, HDP shows higher performance, showing that the method is also applicable to simpler, yet multimodal, tasks. The training curve in Figure 5 shows that a significant gap forms between HDP and Diffuser during training, which we hypothesize is related to the discovery of the connection between the symbolic and continuous plans. The results for *Real-world Sorting* in Fig. 8 show that HDP prevails here too, with a 50% performance gap to Diffuser. In addition to the wide performance gap, there is also a significant difference in failure modes. A majority of Diffuser failures are not due to collisions, but rather to inconsistent plans, as demonstrated in Fig. 1, resulting in a final unsorted sequence of blocks. HDP, however, only fails due to imprecise motion, such as missing blocks, but the overall plan remains consistent.

B. Robustness to Task Complexity

To explore the performance of the methods as planning complexity increases, we evaluate them on a 2D sorting task. We construct a suite of three tasks, with an increasing number of blocks, using a demonstration collection procedure similar to *X-Arm Sorting*. We report the results in Figure 7. While the baseline performance drops monotonically with increasing task complexity, HDP is remarkably robust to the complexity increase.

C. Conditional Sampling with Hybrid Diffusion Planning

In addition to the dramatic performance gain of our method, the inclusion of modeling over discrete plans allows for many practical ways of conditional sampling, such as conditioning on a symbolic sequence as shown in Figure 9. To verify the correspondence between the symbolic and continuous plan, we train HDP on the *Arrange Blocks* task for 20k epochs, reaching 100% task performance. Conditioning this model on the symbolic plan presented in Fig. 9 results in corresponding block placements in every one of 20 trials.

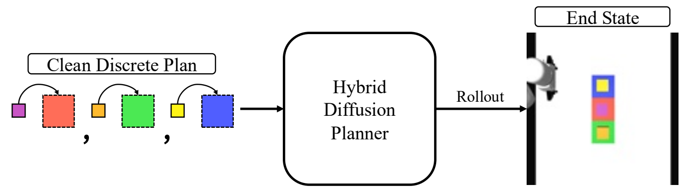


Fig. 9: **Conditioning on symbolic plans.** HDP allows for conditioning on a discrete plan when generating the continuous motion trajectory.

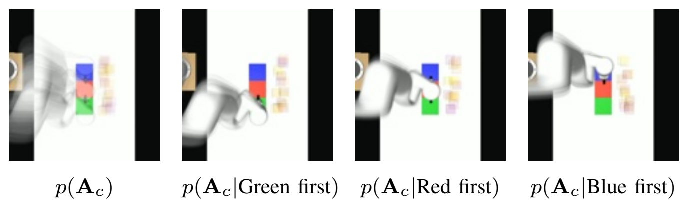


Fig. 10: **Flexible Symbolic Conditioning.** Conditioning HDP on different partially specified symbolic plans restricts the action plans to the specified slot (multiple action plans overlaid on image).

Due to the masking of the discrete plan during training, conditioning the planner on a partly complete plan is within its training distribution. This is performed by initializing the symbolic plan with selected elements unmasked. For example, for the *Arrange Block* task, only specifying which slot to place the first block at will result in HDP filling the continuous and remaining symbolic plan. Figure 10 shows an overlay of 20 rollouts with this conditioning, proving that HDP indeed accepts such conditioning and has 100% adherence. This conditioning is more fine-grained than typical language-conditioned policies, which accept only complete sentences.

V. Conclusion

We present HDP, a hybrid diffusion planning algorithm that learns to plan consistent, long-horizon plans from demonstrations. We illustrate how such a system improves long-horizon planning performance and further how it enables flexible and controllable planning.

References

[1] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is Conditional Generative Modeling all you need for Decision-Making?, July 2023. URL http://arxiv.org/abs/2211.15657. arXiv:2211.15657 [cs].

[2] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured Denoising Diffusion Models in Discrete State-Spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

[3] Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A Continuous Time Framework for Discrete Denoising Models, October 2022. URL http://arxiv.org/abs/2205.14987. arXiv:2205.14987.

[4] Joao Carvalho, An T. Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion Planning Diffusion: Learning and Planning of Robot Motions with Diffusion Models, August 2023. URL http://arxiv.org/abs/2308.01557. arXiv:2308.01557 [cs].

[5] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked Generative Image Transformer, February 2022. URL http://arxiv.org/abs/2202.04200. arXiv:2202.04200 [cs].

[6] 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, 2024.

[7] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation, July 2023. ISBN 978-0-9923747-9-2. doi: 10.15607/RSS.2023.XIX.026. URL http://www.roboticsproceedings.org/rss19/p026.pdf.

[8] Sudeep Dasari, Oier Mees, Sebastian Zhao, Mohan Kumar Srirama, and Sergey Levine. The Ingredients for Robotic Diffusion Transformers, October 2024. URL http://arxiv.org/abs/2410.10088. arXiv:2410.10088 [cs].

[9] Zibin Dong, Jianye Hao, Yifu Yuan, Fei Ni, Yitian Wang, Pengyi Li, and Yan Zheng. DiffuserLite: Towards Real-time Diffusion Planning, February 2024. URL http://arxiv.org/abs/2401.15443. arXiv:2401.15443 version: 4.

[10] 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)*, pages 1412–1419, October 2024. doi: 10.1109/IROS58592.2024.10802746. URL https://ieeexplore.ieee.org/document/10802746/?arnumber=10802746. ISSN: 2153-0866.

[11] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated Task and Motion Planning, October 2020. URL http://arxiv.org/abs/2010.01083. arXiv:2010.01083.

[12] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. PDDLStream: Integrating Symbolic Planners and Blackbox Samplers via Optimistic Adaptive Planning, March 2020. URL http://arxiv.org/abs/1802.08705. arXiv:1802.08705.

[13] Caelan Reed Garrett, Chris Paxton, Tomas Lozano-Perez, Leslie Pack Kaelbling, and Dieter Fox. Online Replanning in Belief Space for Partially Observable Task and Motion Problems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5678–5684, Paris, France, May 2020. IEEE. ISBN 978-1-72817-395-5. doi: 10.1109/ICRA40945.2020.9196681. URL https://ieeexplore.ieee.org/document/9196681/.

[14] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and R. Medina-Carnicer. Generation of fiducial marker dictionaries using Mixed Integer Linear Programming. *Pattern Recognition*, 51:481–491, March 2016. ISSN 0031-3203. doi: 10.1016/j.patcog.2015.09.023. URL https://www.sciencedirect.com/science/article/pii/S0031320315003544.

[15] B. K. Haddon. Cycle-Sort: A Linear Sorting Method. *The Computer Journal*, 33(4):365–367, January 1990. ISSN 0010-4620. doi: 10.1093/comjnl/33.4.365. URL https://doi.org/10.1093/comjnl/33.4.365.

[16] Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance, July 2022. URL http://arxiv.org/abs/2207.12598. arXiv:2207.12598 [cs].

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

[18] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with Diffusion for Flexible Behavior Synthesis. In *Proceedings of the 39th International Conference on Machine Learning*, pages 9902–9915. PMLR, June 2022. URL https://proceedings.mlr.press/v162/janner22a.html. ISSN: 2640-3498.

[19] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational Diffusion Models, April 2023. URL http://arxiv.org/abs/2107.00630. arXiv:2107.00630 [cs, stat].

[20] Yunhao Luo, Chen Sun, Joshua B. Tenenbaum, and Yilun Du. Potential Based Diffusion Motion Planning, July 2024. URL http://arxiv.org/abs/2407.06169. arXiv:2407.06169 [cs].

[21] Yunhao Luo, Utkarsh A. Mishra, Yilun Du, and Danfei Xu. Generative Trajectory Stitching through Diffusion Composition, March 2025. URL http://arxiv.org/abs/

2503.05153. arXiv:2503.05153 [cs].

[22] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What Matters in Learning from Offline Human Demonstrations for Robot Manipulation, September 2021. URL http://arxiv.org/abs/2108.03298. arXiv:2108.03298 [cs].

[23] Ajay Mandlekar, Caelan Garrett, Danfei Xu, and Dieter Fox. Human-in-the-Loop Task and Motion Planning for Imitation Learning, October 2023. URL http://arxiv.org/abs/2310.16014. arXiv:2310.16014 [cs].

[24] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks, July 2022. URL http://arxiv.org/abs/2112.03227. arXiv:2112.03227 [cs].

[25] Utkarsh A. Mishra, Shangjie Xue, Yongxin Chen, and Danfei Xu. Generative Skill Chaining: Long-Horizon Skill Planning with Diffusion Models, October 2023. URL http://arxiv.org/abs/2401.03360. arXiv:2401.03360 [cs].

[26] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating Human Behaviour with Diffusion Models, March 2023. URL http://arxiv.org/abs/2301.10677. arXiv:2301.10677 [cs, stat].

[27] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-Conditioned Imitation Learning using Score-based Diffusion Policies. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation, July 2023. ISBN 978-0-9923747-9-2. doi: 10.15607/RSS.2023.XIX.028. URL http://www.roboticsproceedings.org/rss19/p028.pdf.

[28] Francisco J. Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76:38–47, August 2018. ISSN 0262-8856. doi: 10.1016/j.imavis.2018.05.004. URL https://www.sciencedirect.com/science/article/pii/S0262885618300799.

[29] Kallol Saha, Vishal Mandadi, Jayaram Reddy, Ajit Srikanth, Aditya Agarwal, Bipasha Sen, Arun Singh, and Madhava Krishna. EDMP: Ensemble-of-costs-guided Diffusion for Motion Planning, September 2023. URL http://arxiv.org/abs/2309.11414. arXiv:2309.11414.

[30] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and Generalized Masked Diffusion for Discrete Data, June 2024. URL http://arxiv.org/abs/2406.04329. arXiv:2406.04329.

[31] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. CLIPort: What and Where Pathways for Robotic Manipulation, September 2021. URL http://arxiv.org/abs/2109.12098. arXiv:2109.12098 [cs].

[32] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, November 2015. URL http://arxiv.org/abs/1503.03585. arXiv:1503.03585 [cond-mat, q-bio, stat].

[33] Toshihide Ubukata, Jialong Li, and Kenji Tei. Diffusion Model for Planning: A Systematic Literature Review, August 2024. URL http://arxiv.org/abs/2408.10266. arXiv:2408.10266.

[34] Zhou Xian, Nikolaos Gkanatsios, Theophile Gervet, Tsung-Wei Ke, and Katerina Fragkiadaki. ChainedDiffuser: Unifying Trajectory Diffusion and Keypose Prediction for Robotic Manipulation. In *Proceedings of The 7th Conference on Robot Learning*, pages 2323–2339. PMLR, December 2023. URL https://proceedings.mlr.press/v229/xian23a.html. ISSN: 2640-3498.

[35] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Ayzaan Wahid, Vikas Sindhwani, and Johnny Lee. Transporter Networks: Rearranging the Visual World for Robotic Manipulation, January 2022. URL http://arxiv.org/abs/2010.14406. arXiv:2010.14406 [cs].

[36] Tony Z. Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Seyed Kamyar Seyed Ghasemipour, Chelsea Finn, and Ayzaan Wahid. ALOHA Unleashed: A Simple Recipe for Robot Dexterity. In *Proceedings of The 8th Conference on Robot Learning*, pages 1910–1924. PMLR, January 2025. URL https://proceedings.mlr.press/v270/zhao25b.html. ISSN: 2640-3498.

[37] Zihan Zhou, Animesh Garg, Dieter Fox, Caelan Garrett, and Ajay Mandlekar. SPIRE: Synergistic Planning, Imitation, and Reinforcement Learning for Long-Horizon Manipulation, October 2024. URL http://arxiv.org/abs/2410.18065. arXiv:2410.18065 [cs].

[38] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. VIOLA: Imitation Learning for Vision-Based Manipulation with Object Proposal Priors. In *Proceedings of The 6th Conference on Robot Learning*, pages 1199–1210. PMLR, March 2023. URL https://proceedings.mlr.press/v205/zhu23a.html. ISSN: 2640-3498.

Experiment	Epochs	# Demos	H_c	H_d	D_{ac}	Vocab. Size
X-Arm Sorting	5000	200	145	108	4	10
Arrange Blocks	5000	4000	25	15	4	9
Tool-Use	5000	4000	61	15	4	6
Real-world Sorting	5000	200	145	108	4	10
2 Block Sorting	6000	200	73	72	3	8
3 Block Sorting	6000	200	109	108	3	10
4 Block Sorting	6000	200	145	144	3	12

TABLE II: **Hyperparameters for all tasks.** H_c and H_d are the continuous and discrete planning horizons, respectively. D_{ac} denotes the dimensionality of the continuous action space.

APPENDIX A
EXPERIMENTAL DETAILS

A. Training and Evaluation Details

For the simulated results, we follow the evaluation procedure of Chi et al. [7], and train the models for three seeds in parallel. They are evaluated with 50 environment initializations at regular intervals during training, and the average score is calculated over the last 10 checkpoints over the three seeds, effectively capturing performance over 1500 initializations. Training is done using NVIDIA A100 GPUs, with training times ranging from 30 minutes for *Real-world Sorting* to 15 hours for *Rearrange Blocks*.

We provide hyperparameters for each experimental setup in Table II. We use a batch size of 64 for all experiments, and set the architecture with hyperparameters in Table III.

B. Architecture hyperparameters

See the Diffusion Policy codebase [7] for details on the original architecture. The discrete diffusion level k_d is processed with a learned embedding consisting of two linear layers with a Mish non-linearity before concatenating with the embedding of continuous variable diffusion step k_c and observation $\mathbf{O}$.

Parameter	Value
Num layers	8
Num heads	4
Emb. dim.	256
Drop emb. prob.	0.0
Drop atten. prob.	0.3
Causal Attention	Disabled

TABLE III: **Transformer architecture hyperparameters.**

APPENDIX B
ALGORITHMIC DETAILS

This section outlines details on combining MD4 [30] with DDPM [17] during training and sampling.

During training, the diffusion steps for the discrete variable diffusion are sampled from a continuous uniform distribution $k_d \sim \mathcal{U}[0, 1]$. We sample the continuously distributed diffusion step k_d with the low-discrepancy sampling [19] following Shi et al. [30]. For the continuous variable diffusion, DDPM expects a sample from a categorical distribution over the set of all training levels $k_c \sim \mathcal{U}\{0, \ldots, N-1\}$.

At sampling time, the continuous diffusion iterates through the diffusion levels $\{N-1, \ldots, 0\}$, while the discrete variable diffusion applies a cosine masking schedule [5]. The variable i iterates from 0 to $N-1$, and

$$t = \cos\left(\frac{\pi i}{2N}\right)$$
$$s = \cos\left(\frac{\pi(i+1)}{2N}\right),$$

where t is passed through the model (as k_d), and both s and t are used for denoising the sample. See Shi et al. [30] for further details on MD4 sampling.

APPENDIX C
ROLLOUT VISUALIZATIONS

Figure 11 shows the rollout of all methods on the *Real-world Sorting* task, all with the same permutation. All methods except Separate Diffuser plan a non-collision sequence. However, `HDP` is the only method that solves the task.

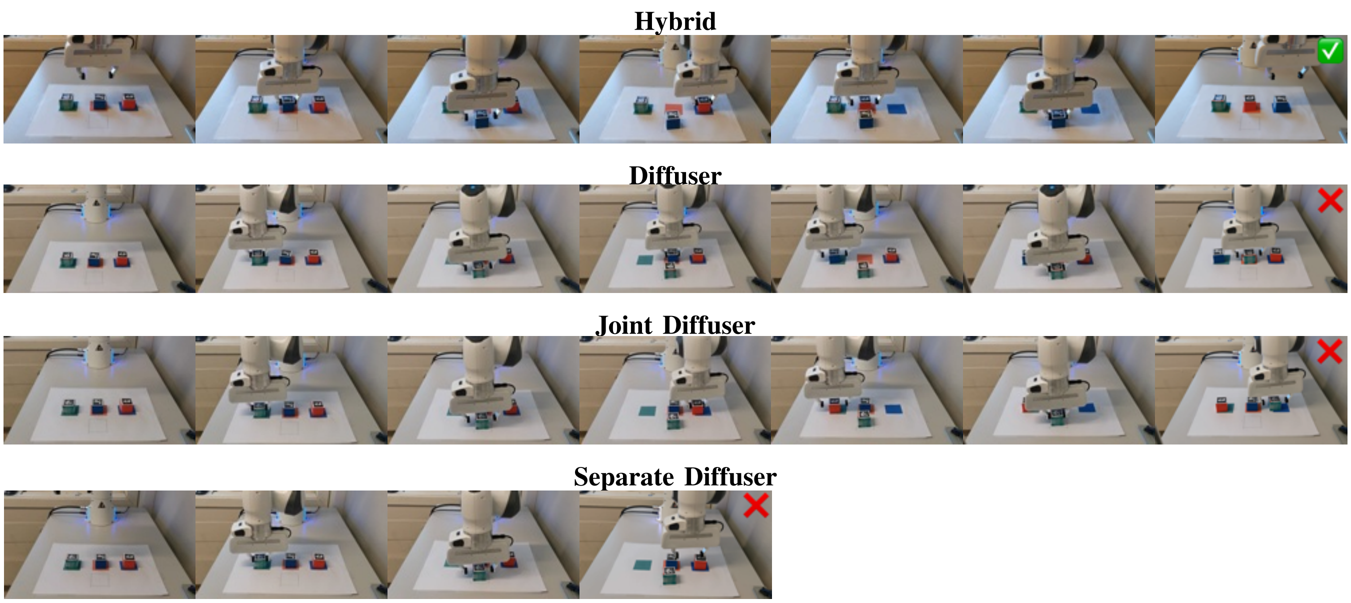


Fig. 11: **Real-world sorting rollouts** for all methods.