Diffusion Predictive Control with Constraints

Ralf Römer RALF.ROEMER@TUM.DE
Alexander von Rohr ALEX.VON.ROHR@TUM.DE
Angela P. Schoellig ANGELA.SCHOELLIG@TUM.DE
Technical University of Munich, Germany; TUM School of Computation, Information and Technology, Learning Systems and Robotics Lab; Munich Institute of Robotics and Machine Intelligence (MIRMI)

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

Diffusion models have become popular for policy learning in robotics due to their ability to capture high-dimensional and multimodal distributions. However, diffusion policies are stochastic and typically trained offline, limiting their ability to handle unseen and dynamic conditions where novel constraints not represented in the training data must be satisfied. To overcome this limitation, we propose diffusion predictive control with constraints (DPCC), an algorithm for diffusion-based control with explicit state and action constraints that can deviate from those in the training data. DPCC incorporates model-based projections into the denoising process of a trained trajectory diffusion model and uses constraint tightening to account for model mismatch. This allows us to generate constraint-satisfying, dynamically feasible, and goal-reaching trajectories for predictive control. We show through simulations of a robot manipulator that DPCC outperforms existing methods in satisfying novel test-time constraints while maintaining performance on the learned control task.

Keywords: Diffusion Policies, Imitation Learning, Predictive Control, Robotics

1. Introduction

Recent advances in using diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) for control and robotics have demonstrated their potential in applications such as robot manipulation (Chi et al., 2023) and locomotion (Huang et al., 2024). These works highlight how diffusion models excel at learning policies directly from diverse demonstrations, capturing multimodal behavior, and handling high-dimensional state and action spaces. However, as diffusion models are trained offline to generate samples via an iterative stochastic denoising process, they don't account for hard constraints, especially if these are not present in the training data. This limits their applicability in controlling robots under changing operation conditions and in real-world dynamic environments where novel and potentially time-varying constraints, such as avoiding obstacles, must be satisfied.

In contrast, the ability to modify a plan according to unforeseen circumstances is a hallmark of model-based control and planning approaches, such as model predictive control (MPC) (Rawlings et al., 2017). These methods leverage a model of the system dynamics to generate feasible trajectories that satisfy constraints despite external disturbances by incorporating feedback. However, MPC requires a cost function, which is often difficult to formulate for complex robotics tasks.

These limitations motivate combining the expressiveness and flexibility of diffusion-based control policies with the ability of MPC to satisfy hard constraints. Different strategies have been proposed to incorporate constraints into diffusion models, either during training (Bastek et al., 2024), inference (Carvalho et al., 2023; Christopher et al., 2024), or both (Ajay et al., 2023) (see Section 2 for a more detailed discussion). In this work, we propose Diffusion Predictive Control with

Constraints (DPCC), an algorithm that extends diffusion policies to operate under unseen state and action constraints. DPCC achieves this by integrating repeated model-based projections into the trajectory denoising process. In addition, we adopt a constraint-tightening mechanism to account for errors in the dynamics model. At each timestep, DPCC generates a batch of predicted trajectories that are dynamically feasible, constraint-satisfying, and perform the learned task, and applies the first action from a selected trajectory. In summary, our main contributions are the following:

- We show that generating goal-reaching trajectories that are guaranteed to satisfy constraints can be achieved by incorporating model-based projections into the backward diffusion process.
- We propose additional constraint tightening to account for model errors and a selection mechanism for the trajectories generated by the diffusion model to improve task performance.
- We evaluate our approach in simulations of a robotic manipulator and demonstrate its superior performance in satisfying novel constraints while still reliably solving the task.

These contributions collectively advance the state of the art in diffusion-based control policies, enabling their deployment in safety-critical environments by satisfying novel constraints.

2. Related Work

Diffusion-Based Control: Diffusion models have recently been applied to various decision-making tasks such as imitation learning (IL) (Pearce et al., 2023; Chi et al., 2023; Chen et al., 2023), offline reinforcement learning (Janner et al., 2022; Ajay et al., 2023) and motion planning (Carvalho et al., 2023; Power et al., 2023). While some works generate only one action per timestep (Pearce et al., 2023; Reuss et al., 2023), most approaches adopt a receding horizon control strategy. This can be done by directly predicting state-action trajectories, either using a single (Janner et al., 2022) or two separate diffusion models (Zhou et al., 2024), or by predicting state (or high-level action) trajectories and using a separate controller (Ajay et al., 2023; Chi et al., 2023). In this regard, we follow Diffuser (Janner et al., 2022) as this approach allows us to ensure dynamic feasibility.

Diffusion Models with Constraints: Many generative modeling tasks require generating samples that are not only from the same distribution as the training data but also adhere to certain constraints. If the constraints are always the same, a residual loss can be added to the training objective (Bastek et al., 2024). A more flexible approach is to sample from a conditional distribution, where the conditioning variable represents a parameterization of the constraints. Classifier-free guidance (Ho and Salimans, 2022) trains additional diffusion models for each condition and can be used to encourage satisfaction of constraints seen in the training data or novel combinations of those constraints (Ajay et al., 2023), but requires more labeled data. It has also been proposed to formulate constraints via cost functions and add their gradients to the backward diffusion process (Carvalho et al., 2023; Kondo et al., 2024), which is conceptually similar to classifier guidance (Dhariwal and Nichol, 2021). However, training loss modification and model conditioning can only encourage but not guarantee constraint satisfaction of the generated samples. Post-processing methods impose constraints on the generated samples by modifying them after the last denoising step, usually by solving an optimization problem (Giannone et al., 2023; Power et al., 2023; Mazé and Ahmed, 2023). As the optimization problem does not consider the unknown data likelihood, post-processing may result in samples that significantly deviate from the data distribution. To address this problem, the integration of projections into the denoising process has recently been investigated. However, these approaches either disregard the system dynamics and deviations between the learned and the actual distribution (Römer et al., 2024), resulting in frequent constraint violations in closed-loop operation or are too computationally expensive for sequential decision-making (Christopher et al., 2024).

3. Problem Statement

We consider a dynamical system with state $\boldsymbol{s}_t \in \mathcal{S}$ and action $\boldsymbol{a}_t \in \mathcal{A}$ at timestep t that is governed by the discrete-time dynamics

$$\boldsymbol{s}_{t+1} = \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \boldsymbol{w}_t, \tag{1}$$

where $\boldsymbol{f}$ is known, and $\boldsymbol{w}_t$ is an unknown disturbance (or model mismatch) bounded by $\|\boldsymbol{w}_t\|_2 \leq \gamma$ for all t. We aim to control the system (1) such that a goal $\boldsymbol{g} \in \mathcal{G}$ is reached, which is indicated by a binary indicator function $\phi : \mathcal{S} \times \mathcal{G} \to \{0, 1\}$. For this, we assume the availability of a dataset

$$\mathcal{D} = \left\{ \boldsymbol{\tau}_{\mathrm{e}}^{(n)} = \left(\boldsymbol{s}_0^{(n)}, \boldsymbol{a}_0^{(n)}, \ldots, \boldsymbol{s}_{T_n}^{(n)}, \boldsymbol{a}_{T_n}^{(n)}, \boldsymbol{g}^{(n)} \right) \right\}_{n=1}^{N} \tag{2}$$

containing N demonstrations of system (1) performing the desired task, i.e., $\phi\big(\boldsymbol{s}_{T_n}^{(n)}, \boldsymbol{g}^{(n)}\big) = 1$ for all $n \in \mathbb{I}_1^N = \{1, \ldots, N\}$. The dataset has been collected by an unknown stochastic expert policy $\boldsymbol{\pi}_{\mathrm{e}}$, i.e., $\boldsymbol{a}_t \sim \boldsymbol{\pi}_{\mathrm{e}}(\cdot|\boldsymbol{s}_t, \boldsymbol{g})$. We consider the demonstrations (2) to be multimodal, i.e., they contain multiple distinct ways to reach the goal (Jia et al., 2024; Urain et al., 2024). Therefore, we aim to use a diffusion model to learn a stochastic policy $\boldsymbol{\pi}$ from the data (2) via imitation learning. In addition, our objective is to satisfy novel and potentially time-varying state and action constraints

$$\boldsymbol{s}_t \in \mathcal{S}_t \subseteq \mathcal{S}, \quad \boldsymbol{a}_t \in \mathcal{A}_t \subseteq \mathcal{A}, \qquad \forall t, \tag{3}$$

at test-time, where we assume the sets $\mathcal{S}_t$ and $\mathcal{A}_t$ to be closed for all t. We refer to the constraints (3) as *novel* because we do not assume that they are satisfied by some or all of the demonstrations in the training dataset (2). Such novel constraints can arise, for example, when deploying a learned robot policy in an environment with moving obstacles or when system specifications, such as torque or velocity limits, are different at test time than during data collection.

Problem 1 *Given a demonstration dataset* (2)*, how can we obtain a diffusion policy* $\boldsymbol{\pi}$ *that can control the system* (1) *to reach a desired goal and satisfy the constraints* (3)*?*

4. Background on Trajectory Diffusion

Diffusion models are generative models for learning an unknown target distribution q from samples $\boldsymbol{\tau}^0 \sim q(\cdot)$, which we consider to be trajectories $\boldsymbol{\tau}^0 = (\boldsymbol{s}_{0:T}, \boldsymbol{a}_{0:T})$ of system (1). The main idea is to gradually transform the data into noise and learn a reverse process to reconstruct the data from pure noise (Sohl-Dickstein et al., 2015). Denoising diffusion probabilistic models (DDPM) (Ho et al., 2020) introduce latent variables $\boldsymbol{\tau}^1, \ldots, \boldsymbol{\tau}^K$ and construct a forward diffusion process

$$q\big(\boldsymbol{\tau}^k | \boldsymbol{\tau}^{k-1}, k\big) = \mathcal{N}\big(\sqrt{1-\beta_k}\boldsymbol{\tau}^{k-1}, \beta_k \boldsymbol{I}\big), \tag{4}$$

where $k = 1, \ldots, K$ is the diffusion time step and the values $\beta_{1:k} \in (0, 1)^k$ are determined by a noise schedule. Since the transition dynamics (4) are Gaussian, we can compute marginals in closed form as $q\big(\boldsymbol{\tau}^k | \boldsymbol{\tau}^0, k\big) = \mathcal{N}\left(\sqrt{\bar{\alpha}_k}\boldsymbol{\tau}^0, (1-\bar{\alpha}_k)\boldsymbol{I}\right)$, where $\alpha_k = 1 - \beta_k$, $\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_k$. The noise schedule and the number of diffusion steps K are chosen such that $q\big(\boldsymbol{\tau}^K | \boldsymbol{\tau}^0, K\big) \approx \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, i.e., the forward process gradually transforms the trajectories data into Gaussian noise. This process is reversed by the learnable backward diffusion (or denoising) process

$$p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^{k-1} | \boldsymbol{\tau}^k, k\big) = \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^k, k\big), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^k, k\big)\big), \qquad p\big(\boldsymbol{\tau}^K\big) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{5}$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ can be parameterized by neural networks. The training objective is to match the joint distributions in the forward and backward process, i.e., $q(\boldsymbol{\tau}^{0:K}) = q(\boldsymbol{\tau}^0)\prod_{k=1}^{K} q(\boldsymbol{\tau}^k|\boldsymbol{\tau}^{k-1}, k)$ and $p_{\boldsymbol{\theta}}(\boldsymbol{\tau}^{0:K}) = p(\boldsymbol{\tau}^K)\prod_{k=1}^{K} p_{\boldsymbol{\theta}}(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k, k)$, by maximizing the evidence-lower bound (ELBO) (Ho et al., 2020). The variance is often set to $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{\tau}^k, k) = \sigma_k^2 \boldsymbol{I}$, where $\sigma_k^2 = \beta_k \frac{1-\bar{\alpha}_{k-1}}{1-\bar{\alpha}_k}$, and the mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ is learned indirectly by learning to predict the noise added to $\boldsymbol{\tau}^0 \sim q(\cdot)$ via the surrogate loss function $\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{k\sim\mathrm{Unif}(1,K),\boldsymbol{\tau}^0\sim q(\cdot),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\sqrt{\bar{\alpha}_k}\boldsymbol{\tau}^0 + \sqrt{1-\bar{\alpha}_k}\boldsymbol{\epsilon}, k\right)\right\|_2\right]$, where $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{\tau}^k, k) = \frac{\sqrt{1-\bar{\alpha}_k}}{\beta_k}\left(\boldsymbol{\tau}^k - \sqrt{\alpha_k}\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{\tau}^k, k)\right)$. After training with this loss, we can generate trajectories from the learned distribution $\boldsymbol{\tau}^0 \sim p_{\boldsymbol{\theta}}(\cdot)$ by iterating through the backward diffusion process (5). It is also possible to learn and sample from a conditional distribution $\boldsymbol{\tau}^0 \sim p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c})$, where $\boldsymbol{c}$ is some context, via methods such as inpainting (Sohl-Dickstein et al., 2015), classifier guidance (Dhariwal and Nichol, 2021) or classifier-free guidance (Ho and Salimans, 2022).

5. Methodology

In this section, we present the DPCC algorithm. We explain the use of diffusion models for receding horizon control and show how to incorporate novel constraints in the backward diffusion process via model-based projections. Lastly, we account for model errors using constraint tightening and introduce two trajectory selection criteria. The DPCC method is summarized in Algorithm 1.

5.1. Diffusion-Based Receding Horizon Control

We address the problem of learning a control policy from an offline dataset (2) via conditional generative modeling (Janner et al., 2022; Ajay et al., 2023). For this, we consider state-action trajectories $\boldsymbol{\tau} = (\boldsymbol{s}_{t:t+H}, \boldsymbol{a}_{t:t+H})$ of horizon length $H+1$ of system (1). The expert policy $\boldsymbol{\pi}_{\mathrm{e}}$ induces a conditional trajectory distribution $q(\boldsymbol{\tau}|\boldsymbol{c})$, where $\boldsymbol{c} = (\boldsymbol{s}_t, \boldsymbol{g})$, which is generally unknown. Utilizing the samples from $q(\boldsymbol{\tau}|\boldsymbol{c})$ in (2), we train a diffusion model to learn a trajectory distribution

$$p_{\boldsymbol{\theta}}(\boldsymbol{\tau}|\boldsymbol{c}) \approx q(\boldsymbol{\tau}|\boldsymbol{c}), \tag{6}$$

as described in Section 4, where the learned backward diffusion process is given by

$$p_{\boldsymbol{\theta}}(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k, k, \boldsymbol{c}) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{\tau}^k, k, \boldsymbol{c}), \sigma_k^2 \boldsymbol{I}\right), \qquad p(\boldsymbol{\tau}^K) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{7}$$

As the learned distribution (6) implicitly encodes information about the dynamics (1) and how to solve the task, we can use it for receding horizon control: At time t, given the current state $\boldsymbol{s}_t$ and the goal $\boldsymbol{g}$, we sample a future trajectory $\boldsymbol{\tau}_{t:t+H|t} = (\boldsymbol{s}_{t:t+H|t}, \boldsymbol{a}_{t:t+H|t})$ from (6) and apply the first action $\boldsymbol{a}_{t|t}$. Here $\boldsymbol{a}_{t+i|t}$ denotes the action prediction for time $t+i$ generated at time t (Borrelli et al., 2017). This process is repeated until the goal is reached, i.e., $\phi(\boldsymbol{s}_t, \boldsymbol{g}) = 1$. A major limitation of this approach is that it cannot take into account constraints of the form (3). In the following, we will therefore discuss how to incorporate such constraints into diffusion-based receding-horizon control.

5.2. Constraint- and Model-Based Trajectory Diffusion

Diffusion predictive control as discussed in Section 5.1 generates predicted trajectories from the learned distribution (6). In that way, the approach implicitly encourages satisfaction of constraints that were already present in the demonstration dataset (2), such as state and action bounds. However, the controlled system may still violate these constraints because the denoising process (7) is

stochastic and the true trajectory distribution can generally only be approximated. Moreover, our main goal is to satisfy novel constraints of the form (3). As a first step, we need to ensure that these constraints are satisfied by the open-loop trajectories $\boldsymbol{\tau}_{t:t+H|t}$ predicted at timestep t, i.e.,

$$\boldsymbol{\tau}_{t:t+H|t} \in \mathcal{Z} = \left\{ \boldsymbol{\tau} = (\boldsymbol{s}_{t:t+H}, \boldsymbol{a}_{t:t+H}) \middle| \, \boldsymbol{s}_{t'} \in \mathcal{S}_{t'}, \boldsymbol{a}_{t'} \in \mathcal{A}_{t'}, \; \forall t' \in \mathbb{I}_t^H \right\}. \tag{8}$$

One way to enforce (8) is to perform a projection of the denoised trajectory $\boldsymbol{\tau}^0 = \boldsymbol{\tau}_{t:t+H|t}$ into the set $\mathcal{Z}$ as $\Pi_{\mathcal{Z}}(\boldsymbol{\tau}^0) = \arg\min_{\tilde{\boldsymbol{\tau}} \in \mathcal{Z}} \left\| \boldsymbol{\tau}^0 - \tilde{\boldsymbol{\tau}} \right\|_2$. However, since this projection only takes into account the state and action constraints (3), using the projected trajectory for receding-horizon control has two drawbacks: The resulting trajectory may not be dynamically feasible anymore, and it may not be suitable for achieving the goal $\boldsymbol{g}$. We can mitigate the first problem by taking into account the dynamics (1) and applying a model-based constraint set projection to $\boldsymbol{\tau}^0$, which is defined by

$$\Pi_{\mathcal{Z}_{\boldsymbol{f}}}(\boldsymbol{\tau}) = \underset{\tilde{\boldsymbol{\tau}} = (\boldsymbol{s}_{t:t+H|t}, \boldsymbol{a}_{t:t+H|t}) \in \mathcal{Z}}{\arg\min} \; \|\boldsymbol{\tau} - \tilde{\boldsymbol{\tau}}\|_2^2 \tag{9a}$$

$$\text{s.t.} \qquad \boldsymbol{s}_{t'+1|t} = \boldsymbol{f}(\boldsymbol{s}_{t'|t}, \boldsymbol{a}_{t'|t}), \qquad \forall t' \in \mathbb{I}_t^H, \tag{9b}$$

where $\mathcal{Z}_{\boldsymbol{f}} = \left\{ \boldsymbol{\tau} = (\boldsymbol{s}_{t:t+H}, \boldsymbol{a}_{t:t+H}) \middle| \, \boldsymbol{\tau} \in \mathcal{Z}, \boldsymbol{s}_{t+1} = \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{a}_t), \forall t \in \mathbb{I}_t^H \right\}$ is assumed to be non-empty, and the projection cost is denoted by $c_{\mathcal{Z}_{\boldsymbol{f}}}(\boldsymbol{\tau}) = \|\boldsymbol{\tau} - \Pi_{\mathcal{Z}_{\boldsymbol{f}}}(\boldsymbol{\tau})\|_2^2$. Since the projection is goal-independent, it may render the trajectory less useful for completing the task. As the training dataset (2) consists of dynamically feasible and goal-reaching trajectories, these two properties are implicitly encoded in the learned trajectory distribution (6), which is defined by the backward process (7). Thus, we aim to modify (7) only as much as necessary to guarantee constraint satisfaction.

We approach this problem via control as inference (Toussaint, 2009) and introduce a binary variable $\mathcal{O} \in \{0, 1\}$ that is related to the feasibility of a trajectory $\boldsymbol{\tau}$, i.e., whether $\boldsymbol{\tau} \in \mathcal{Z}_{\boldsymbol{f}}$. We can then formulate our objective as sampling trajectories from the conditional distribution

$$p_{\boldsymbol{\theta}}(\boldsymbol{\tau} | \mathcal{O} = 1) \propto p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) p(\mathcal{O} = 1 | \boldsymbol{\tau}) \tag{10}$$

instead of the original learned distribution (6), where we have omitted the conditioning on the context $\boldsymbol{c}$ for brevity. If the likelihood $p(\mathcal{O}|\boldsymbol{\tau})$ is defined as

$$p(\mathcal{O} = 1 | \boldsymbol{\tau}) = \begin{cases} 1, & \text{if } \boldsymbol{\tau} \in \mathcal{Z}_{\boldsymbol{f}} \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

sampling from (10) is guaranteed to yield feasible trajectories. In principle, this could be performed through rejection sampling from the learned distribution (6), but this becomes too computationally inefficient if samples $\boldsymbol{\tau} \sim p_{\boldsymbol{\theta}}(\cdot)$ are unlikely to lie within $\mathcal{Z}_{\boldsymbol{f}}$. Instead, we can sample from (10) more efficiently if the likelihood $p(\mathcal{O}|\boldsymbol{\tau})$ takes a different form than (11).

Theorem 1 *Let $\mathcal{Z}_{\boldsymbol{f}}$ be a closed convex set, $\sigma_k > 0$, $\forall k \in \mathbb{I}_1^K$, and let the feasibility likelihood be defined by $p(\mathcal{O} = 1 | \boldsymbol{\tau}, k) \propto \exp\left(-\frac{1}{2\sigma_k^2} d(\boldsymbol{\tau}, \mathcal{Z}_{\boldsymbol{f}})^2 \right)$, where $d(\boldsymbol{\tau}, \mathcal{Z}_{\boldsymbol{f}}) = \min_{\tilde{\boldsymbol{\tau}} \in \mathcal{Z}_{\boldsymbol{f}}} \|\tilde{\boldsymbol{\tau}} - \boldsymbol{\tau}\|_2$ is the distancee between $\boldsymbol{\tau}$ and $\mathcal{Z}_{\boldsymbol{f}}$. Then, we can approximately sample from (10) via the modified denoising process*

$$p_{\boldsymbol{\theta}}(\boldsymbol{\tau}^{k-1} | \boldsymbol{\tau}^k, k, \mathcal{O} = 1) \approx \mathcal{N}(\Pi_{\mathcal{Z}_{\boldsymbol{f}}}(\boldsymbol{\mu}_{\boldsymbol{\theta}}^k), \sigma_k^2 \boldsymbol{I}), \qquad p(\boldsymbol{\tau}^K, \mathcal{O}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{12}$$

where the learned mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}^k = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{\tau}^k, k)$ is the same as in (7), and $\Pi_{\mathcal{Z}_{\boldsymbol{f}}}$ is defined in (9).

Algorithm 1: DPCC: Diffusion Predictive Control with Constraints.

Input: Diffusion model $\boldsymbol{\epsilon_\theta}$, goal $\boldsymbol{g}$, dynamics $\boldsymbol{f}$, state constraints $\mathcal{S}_{0,1,\dots}$, action constraints $\mathcal{A}_{0,1,\dots}$.
Set $t = 0$.
Compute tightened state constraints $\tilde{\mathcal{S}}_{0,1,\dots}$ via (17).
while *goal* $\boldsymbol{g}$ *not reached* **do**
  Get current state $\boldsymbol{s}_t$ and set $\boldsymbol{c} = (\boldsymbol{s}_t, \boldsymbol{g})$.
  Sample a trajectory batch from noise: $\boldsymbol{\tau}_{t:t+H|t}^{K,1:B} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.
  for $k = K, \dots, 1$ **do**
    Trajectory denoising step: $\tilde{\boldsymbol{\tau}}_{t:t+H|t}^{k-1,1:B} \sim \mathcal{N}\big(\boldsymbol{\mu_\theta}(\boldsymbol{\tau}_{t:t+H|t}^{k,1:B}, k, \boldsymbol{c}), \sigma_k^2 \boldsymbol{I}\big)$.
    Model-based projection into the feasible set: $\boldsymbol{\tau}_{t:t+H|t}^{k-1,1:B} = \Pi_{\tilde{\mathcal{Z}}_{\boldsymbol{f}}}\big(\tilde{\boldsymbol{\tau}}_{t:t+H|t}^{k-1,1:B}\big)$.
  end
  Select a trajectory $\boldsymbol{\tau}^* = \boldsymbol{\tau}_{t:t+H|t}^{0,i}$ from $\boldsymbol{\tau}_{t:t+H|t}^{0,1:B}$ via temporal consistency or projection costs (Section 5.4).
  Apply the first action $\boldsymbol{a}_{t|t}$ in $\boldsymbol{\tau}^*$.
  Set $t \leftarrow t + 1$.
end

Proof Incorporating the conditioning on $\mathcal{O}$ into the Markovian backward diffusion process gives

$$p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k, \mathcal{O}, k\big) \propto p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k, k\big) p\big(\mathcal{O}|\boldsymbol{\tau}^{k-1}, k\big), \tag{13}$$

where $p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k, k\big) = \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}^k, \sigma_k^2 \boldsymbol{I}\big)$. As the likelihood $p(\mathcal{O}|\boldsymbol{\tau}, k)$ is smooth with respect to $\boldsymbol{\tau}$ by definition, its logarithm at $\boldsymbol{\tau} = \boldsymbol{\tau}^{k-1}$ can be approximated using a first-order Taylor expansion around $\boldsymbol{\mu}_{\boldsymbol{\theta}}^k$, which gives $\log p(\mathcal{O}|\boldsymbol{\tau}^{k-1}, k) \approx \log p\big(\mathcal{O}|\boldsymbol{\mu}_{\boldsymbol{\theta}}^k, k\big) + \big(\boldsymbol{\tau}^{k-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}}^k\big)^\mathsf{T} \boldsymbol{v}(\mathcal{O})$, where $\boldsymbol{v}(\mathcal{O}) = \nabla_{\boldsymbol{\tau}} \log p(\mathcal{O}|\boldsymbol{\tau}, k)|_{\boldsymbol{\tau}=\boldsymbol{\mu}_{\boldsymbol{\theta}}^k}$. This approximation allows us to rewrite (13) as

$$p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k, k, \mathcal{O}\big) \approx \mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}^k + \sigma_k^2 \boldsymbol{v}(\mathcal{O}), \sigma_k^2 \boldsymbol{I}\big), \tag{14}$$

as shown in the derivation of classifier guidance (Dhariwal and Nichol, 2021). Since $\mathcal{Z}_{\boldsymbol{f}}$ is closed and convex by assumption, there exists a unique projection $\boldsymbol{z} = \Pi_{\mathcal{Z}_{\boldsymbol{f}}}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}^k\big)$ for each $\boldsymbol{\mu}_{\boldsymbol{\theta}}^k$; see (Bazaraa et al., 2006), Theorem 2.4.1. Thus, we can write $\boldsymbol{v}(\mathcal{O} = 1)$ as

$$\boldsymbol{v}(\mathcal{O} = 1) = -\frac{1}{\sigma_k^2} d\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}^k, \mathcal{Z}_{\boldsymbol{f}}\big) \nabla_{\boldsymbol{\tau}} d(\boldsymbol{\tau}, \mathcal{Z}_{\boldsymbol{f}})|_{\boldsymbol{\tau}=\boldsymbol{\mu}_{\boldsymbol{\theta}}^k} = \frac{1}{\sigma_k^2}\big(\boldsymbol{z} - \boldsymbol{\mu}_{\boldsymbol{\theta}}^k\big). \tag{15}$$

Inserting (15) into (14) and replacing $\boldsymbol{z}$ by $\Pi_{\mathcal{Z}_{\boldsymbol{f}}}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}^k\big)$ yields (12), which concludes the proof. ■

Theorem 1 assumes that $\mathcal{Z}_{\boldsymbol{f}}$ is convex. This is true, for example, if the constraint sets in (3) are convex and the dynamics (1) are linear. Moreover, with the likelihood definition in Theorem 1, $p(\mathcal{O} = 1|\boldsymbol{\tau}, k) > 0$ for $\boldsymbol{\tau} \notin \mathcal{Z}_{\boldsymbol{f}}$. Consequently, sampling from (10) via (12) is not strictly guaranteed to yield samples $\boldsymbol{\tau}^0 = \boldsymbol{\tau}_{t:t+H|t} \in \mathcal{Z}_{\boldsymbol{f}}$, i.e., trajectories are not guaranteed to satisfy (3).

Nonetheless, Theorem 1 provides theoretical justification to address constraint satisfaction via iterative projections in the denoising process. To ensure that $\boldsymbol{\tau}^0 \in \mathcal{Z}_{\boldsymbol{f}}$ for any variance schedule $\sigma_{1:K}$, we slightly modify (12) and apply the projection *after* adding the noise. This yields the model-informed denoising step with deterministic constraint satisfaction

$$\boldsymbol{\tau}^{k-1} = \Pi_{\mathcal{Z}_{\boldsymbol{f}}}\big(\boldsymbol{\mu_\theta}\big(\boldsymbol{\tau}^k, k, \boldsymbol{c}\big) + \sigma_k \boldsymbol{\epsilon}_k\big), \qquad \boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{16}$$

which we use in DPCC. Here, we have included the context $\boldsymbol{c}$ again. By using (16), We denote the samples projected distribution as $\boldsymbol{\tau}^0 \sim p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{c}, \mathcal{Z}_{\boldsymbol{f}})$.

5.3. Constraint Tightening

The dynamics model $\boldsymbol{f}$ used in the projection (9) is only an approximation of the true system (1), which is subject to a model mismatch $\boldsymbol{w}_t$. Hence, feasibility of the predicted trajectory, i.e., $\boldsymbol{\tau}_{t:t+H|t} \in \mathcal{Z}_{\boldsymbol{f}}$, does not guarantee that the actual future states $\boldsymbol{s}_{t+1}, \boldsymbol{s}_{t+2}, \dots$ will satisfy the constraints (3). We take this into account by tightening the constraints for the predicted states.

Theorem 2 *Let $\boldsymbol{s}_0 \in \mathcal{S}_0$, $\boldsymbol{g} \in \mathcal{G}$, $\mathcal{B}_\gamma$ denote the ℓ_2-norm ball of radius γ and $\ominus$ the Minkowski set difference, and define the tightened constraint sets for all t as*

$$\tilde{\mathcal{S}}_{t+1} = \mathcal{S}_{t+1} \ominus \mathcal{B}_\gamma. \tag{17}$$

Then, if at each timestep $t = 0, 1, \dots$, we sample $\boldsymbol{\tau}_{t:t+H|t} = (\boldsymbol{s}_{t:t+H|t}, \boldsymbol{a}_{t:t+H|t}) \sim p_{\boldsymbol{\theta}}\big(\,\cdot\,|\boldsymbol{c}, \tilde{\mathcal{Z}}_{\boldsymbol{f}}\big)$, where $\tilde{\mathcal{Z}}_{\boldsymbol{f}} = \big\{\boldsymbol{\tau} = (\boldsymbol{s}_{t:t+H}, \boldsymbol{a}_{t:t+H}) \,|\, \boldsymbol{s}_{t'} \in \tilde{\mathcal{S}}_{t'}, \boldsymbol{a}_{t'} \in \mathcal{A}_{t'},\ \forall t' \in \mathbb{I}_t^H\big\}$, and apply the action $\boldsymbol{a}_{t|t}$ to system (1), all future states satisfy the constraints (3), i.e., $\boldsymbol{s}_t \in \mathcal{S}_t$, $\forall t \in \{1, 2, \dots\}$.

Proof Let $\boldsymbol{s}_t \in \mathcal{S}_t$. Due to the definition of $p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}|\boldsymbol{c}, \tilde{\mathcal{Z}}_{\boldsymbol{f}}\big)$ via (16), sampling $\boldsymbol{\tau}_{t:t+H|t} \sim p_{\boldsymbol{\theta}}\big(\boldsymbol{\tau}|\boldsymbol{c}, \tilde{\mathcal{Z}}_{\boldsymbol{f}}\big)$ implies $\boldsymbol{\tau}_{t:t+H|t} \in \tilde{\mathcal{Z}}_{\boldsymbol{f}}$. Consequently, the predicted next state satisfies both $\boldsymbol{s}_{t+1|t} \in \tilde{\mathcal{S}}_{t+1}$ and $\boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{a}_{t|t}) = \boldsymbol{s}_{t+1|t}$. Inserting the latter into the dynamics (1) gives $\boldsymbol{s}_{t+1} = \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{a}_{t|t}) + \boldsymbol{w}_t = \boldsymbol{s}_{t+1|t} + \boldsymbol{w}_t$. The model mismatch is bounded by $\|\boldsymbol{w}_t\|_2 \leq \gamma$ by assumption, so we can write $\boldsymbol{s}_{t+1} \in \tilde{\mathcal{S}}_{t+1} \oplus \mathcal{B}_\gamma = (\mathcal{S}_{t+1} \ominus \mathcal{B}_\gamma) \oplus \mathcal{B}_\gamma \subseteq \mathcal{S}_{t+1}$. As $\boldsymbol{s}_0 \in \mathcal{S}_0$, the result follows by induction. ■

5.4. Trajectory Selection

By using a generative diffusion model for predictive control, we can generate not just one trajectory at teach timestep, but a batch of B candidate trajectories denoted by $\boldsymbol{\tau}_{t:t+H|t}^{0,1:B}$. Many existing works (Chi et al., 2023; Ajay et al., 2023) apply the actions from a trajectory randomly selected from the batch. However, this does not take into account that the candidate trajectories may be diverse and not equally well suited for the task. To improve control performance, we propose two different criteria for selecting a trajectory $\boldsymbol{\tau}_{t:t+H|t}^{0,i(t)}$ from the sampled batch:

- **Temporal Consistency (DPCC-T)**: Frequent replanning using a multimodal trajectory distribution can result in alternating between different behavior modes, which may impact task performance. We can avoid this by selecting the trajectory deviating the least from the previous timestep via $i(t) = \arg\min_j \big\|\boldsymbol{\tau}_{t:t+H-1|t}^{0,j} - \boldsymbol{\tau}_{t:t+H-1|t-1}^{0,i(t-1)}\big\|_2$.
- **Cumulative Projection Cost (DPCC-C)**: We aim to preserve as much information as possible from the learned trajectory distribution (6) despite the modifications to the denoising process (7). This motivates selecting the trajectory that has been modified the least by the projection operation in (16) during the whole denoising process, i.e., $i(t) = \arg\min_j \sum_{k=1}^{K} c_{\mathcal{Z}_{\boldsymbol{f}}}\big(\tilde{\boldsymbol{\tau}}_{t:t+H|t}^{k-1,j}\big)$, where $\tilde{\cdot}$ denotes the trajectories before applying the projection; see Algorithm 1.

6. Evaluation and Discussion

Our simulation experiments primarily aim at answering the following questions:

- **Q1**: Can our proposed DPCC algorithm satisfy novel constraints and still solve the learned task?
- **Q2**: How does the proposed constraint-informed trajectory denoising method (16) perform compared to existing approaches for incorporating constraints into diffusion models?
- **Q3**: How important is the accuracy of the dynamics model used in the projections (9)?

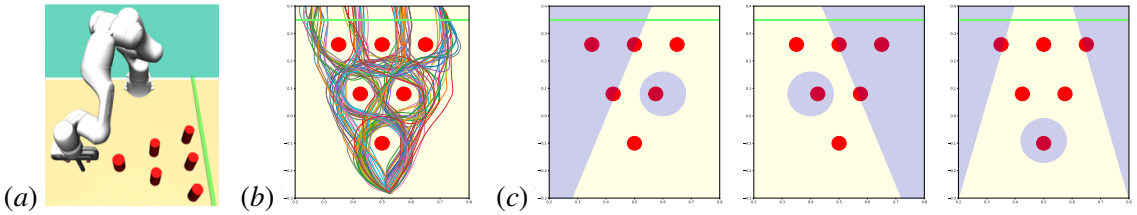

Figure 1: Experiments: (a) Simulation environment, where the objective is to reach the green line with the end-effector without collisions. (b) Multimodal trajectory distribution in the training dataset. (c) Novel test-time constraints (blue).

6.1. Setup

We conduct our experiments[1] in the `Avoiding` simulation environment (Jia et al., 2024) shown in Fig. 1 (a). The task is for a robot manipulator to reach a line (green) with its end-effector without colliding with one of the six obstacles (red). The state $\boldsymbol{s}_t \in \mathbb{R}^4$ consists of the current and desired end-effector positions in the 2D plane, and the action $\boldsymbol{a}_t \in \mathbb{R}^2$ contains the desired Cartesian velocities, which are sent to a low-level controller. The training data set $\mathcal{D}$ contains 96 demonstrations, 4 for each of the 24 different ways of navigating around the six obstacles, resulting in a highly multimodal expert trajectory distribution; see Fig. 1 (b). We consider different formulations of novel state constraints, which are defined by circular and halfspace areas that the end-effector must not enter, as shown in Fig. 1 (c). To ensure that the generated trajectories do not violate the control action bounds, we define $\mathcal{A}_t = \mathcal{A}$, where $\mathcal{A}$ is the smallest bounding box containing all actions from the demonstration dataset (2). The action constraints are defined as $\mathcal{A}_t = \mathcal{A}$, where $\mathcal{A}$ is the smallest bounding box containing all actions from the demonstration dataset (2). We do not assume knowledge of the dynamics of the low-level controller. Instead, we approximate the system dynamics (1) by a simple Euler integration, i.e., $\boldsymbol{s}_{t+1} = \boldsymbol{s}_t + \left[\boldsymbol{a}_t^\mathsf{T}, \boldsymbol{a}_t^\mathsf{T}\right]^\mathsf{T} t_\mathrm{s} + \boldsymbol{w}_t$, where t_s is the sampling time, and the model mismatch $\boldsymbol{w}_t$ accounts for the low-level controller and the numerical error of the Euler integration. The maximum episode length is 300 timesteps. We estimate the upper bound γ on the model mismatch based on 100 policy rollouts without constraints to be $\gamma = 0.025$.

We use a 1D U-Net (Janner et al., 2022) as the diffusion model backbone $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$, employ the cosine noise schedule (Nichol and Dhariwal, 2021) for $\beta_{1:K}$ with $K = 20$ diffusion steps, and condition on the current state using inpainting. At each timestep, we sample a batch of $B = 4$ trajectories with horizon length $H + 1 = 8$. The state constraints render the feasible sets $\tilde{\mathcal{Z}}_{\boldsymbol{f}}$ non-convex, and solve the resulting nonlinear optimization problems in the projections (9) using an SLSQP solver (Virtanen et al., 2020; Kraft, 1994). Computing an action with DPCC takes about 80 ms on a workstation with 64 GB RAM, an NVIDIA Geforce RTX 4090 GPU, and an Intel Core i7-12800HX CPU.

We compare DPCC against three baselines for satisfying constraints with diffusion policies:

- **Guidance**: The constraints are parameterized via cost functions, and their gradients are used to guide the denoising process towards the feasible set (Carvalho et al., 2023; Kondo et al., 2024). We conduct an ablation study to determine suitable weights for the gradient terms.
- **Post-Processing**: The trajectories are projected into the set $\mathcal{Z}_{\boldsymbol{f}}$ only after the last denoising iteration (Giannone et al., 2023; Power et al., 2023).
- **Model-Free**: The projection only takes into account the constraints (3), but not the system dynamics (Römer et al., 2024).

1. Our code is available at github.com/ralfroemer99/dpcc.

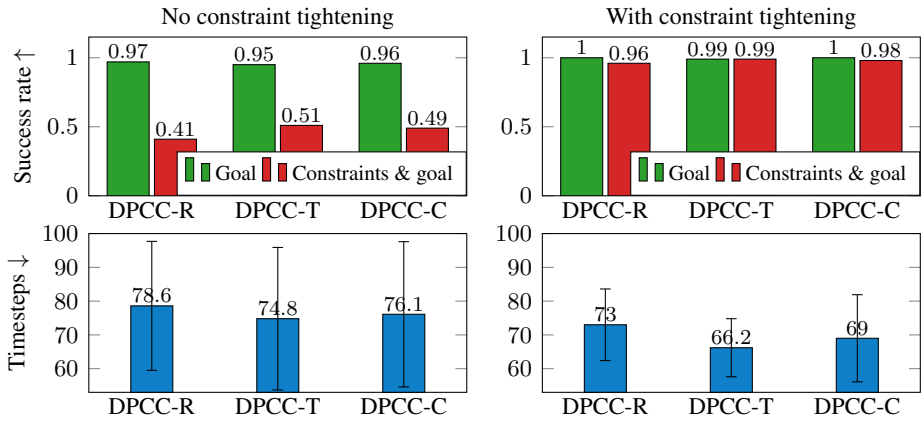

Figure 2: Impact of our constraint tightening method and the trajectory selection criterion (DPCC-R: random, DPCC-T: temporal consistency, DPCC-C: cumulative projection cost) on the success rates and the number of timesteps needed.

None of these prior works use constraint tightening, but we evaluate their performance with and without our constraint tightening method to ensure the comparison is fair.

We use four evaluation metrics: The number of timesteps to reach the goal, the success rates of 1) reaching the goal and 2) satisfying the constraints and reaching the goal, and the number of timesteps in which the constraints are violated. All results are averaged over five training seeds and ten test seeds for each constraint set formulation.

6.2. Results

To answer **Q1**, we conduct an ablation study with regard to two key components of DPCC: The constraint-tightening method (Section 5.3) and the trajectory selection mechanism (Section 5.4). For the latter, we also compare against selecting randomly, referred to as DPCC-R. The results, including standard deviations, are shown in Fig. 2. Projecting onto the non-tightened state constraints $\mathcal{S}_t$ only results in constraint satisfaction success rates below 50%. By utilizing our constraint-tightening approach, we achieve close to 100% success rate for all three trajectory selection criteria. The main performance difference between the selection criteria is the number of time steps needed to reach the goal, which is higher for DPCC-R than for DPCC-T and DPCC-C. One reason for this is that DPCC-T and, to a lesser extent, DPCC-C result in more temporally consistent closed-loop behavior, as shown in Fig. 3. These results show the potential of exploiting the fact that diffusion models can generate multiple trajectories at each timestep without increased computation.

We report the results of our baseline comparison (**Q2**) in Table 1 and also include Diffuser (Janner et al., 2022), which does not take into account constraints at all. DPCC-C has the highest success rate and the smallest number of constraint violations and, on average, reaches the goal significantly faster than all other methods. This highlights that DPCC retains very good task performance by sampling approximately from the conditional distribution (10), which encodes both the learned goal-reaching behavior and constraint satisfaction. We find that for cost guidance, the trajectory is often either not pushed out of the unfeasible region completely, resulting in poor constraint performance, or pushed far away from the boundary of the feasible set, such that reaching the goal requires more timesteps. The model-free projections perform poorly in our experiments. This shows that although the learned distribution (6) contains information about the dynamics, the learned mean $\boldsymbol{\mu_\theta}$ cannot restore a denoised trajectory's dynamic feasibility if it is destroyed by iterative model-free

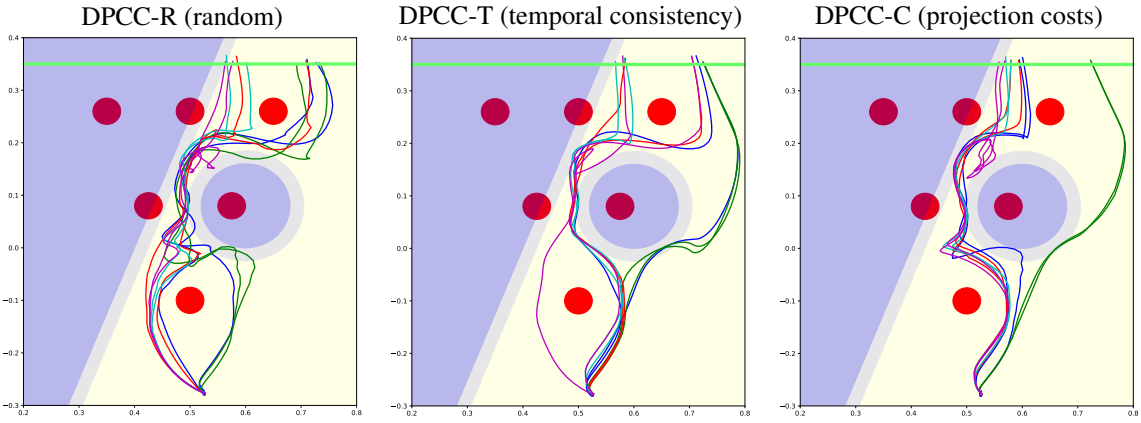

Figure 3: Closed-loop trajectories with DPCC for different trajectory selection criteria and five training seeds, which are indicated by the trajectories' colors. The tightened constraints are visualized in light blue. With our proposed trajectory selection criteria (DPCC-T and DPCC-C), we obtain a smoother behavior and shorter time to reach the goal.

	Constraint tightening	Timesteps	Goal	Constraints & goal	# Constraint violations
Diffuser	-	$76.7^{\pm 12.7}$	**1.00**	0.07	$17.8^{\pm 12.1}$
Guidance	no	$74.5^{\pm 11.8}$	0.99	0.09	$17.3^{\pm 12.6}$
	yes	$75.6^{\pm 12.4}$	0.96	0.13	$17.4^{\pm 14.2}$
Post-Processing	no	$84.5^{\pm 21.5}$	0.95	0.32	$8.2^{\pm 13.9}$
	yes	$79.1^{\pm 13.9}$	**1.00**	0.96	$0.1^{\pm 0.5}$
Model-Free	no	$76.7^{\pm 12.4}$	0.99	0.07	$17.8^{\pm 12.4}$
	yes	$76.1^{\pm 12.1}$	0.99	0.07	$18.0^{\pm 12.0}$
DPCC-C (ours)	no	$76.1^{\pm 21.5}$	0.96	0.49	$6.0^{\pm 11.5}$
	yes	$\mathbf{69.0^{\pm 12.9}}$	**1.00**	**0.98**	$\mathbf{0.0^{\pm 0.3}}$

Table 1: Comparison of DPCC against other approaches for diffusion-based receding-horizon control with constraints.

projections, resulting in a trajectory that the system cannot follow. Post-Processing performs better than the other baselines but also needs significantly more timesteps than DPCC-C.

We have seen that neglecting the dynamics in the projections results in poor constraint satisfaction. To better understand the impact of the dynamics model used in (9) (**Q3**), we consider a mismatch between the assumed sampling time $\hat{t}_\mathrm{s}$ and its true value t_s. The results provided in Table 2 demonstrate that even with a significant deviation by a factor of 4, the constraints can be satisfied in most cases. This shows that the iterative constraint set projections yield much better results with even a very inaccurate dynamics model than when using no model.

$\hat{t}_\mathrm{s}/t_\mathrm{s}$	Timesteps	Goal	Constraints & goal	# Constraint violations
0.25	$85.7^{\pm 16.4}$	**1.00**	0.86	$0.3^{\pm 0.8}$
0.5	$73.1^{\pm 9.8}$	**1.00**	**0.99**	$\mathbf{0.0^{\pm 0.3}}$
1	$\mathbf{69.0^{\pm 12.9}}$	**1.00**	0.98	$\mathbf{0.0^{\pm 0.3}}$
2	$76.6^{\pm 14.8}$	0.99	0.95	$0.3^{\pm 2.1}$
4	$152.0^{\pm 26.3}$	0.88	0.77	$0.6^{\pm 1.8}$

Table 2: Impact of the model mismatch between the dynamics used in the constraint set projection (9), which assume a sampling time $\hat{t}_\mathrm{s}$, and the true dynamics with sampling time t_s, for DPCC-C.

7. Conclusion

DPCC combines the expressivity of diffusion models for offline policy learning with the ability of predictive control to satisfy constraints online in closed-loop operation. We show that incorporating model-based projections into the trajectory denoising process allows us to sample future trajectories that are constraint-satisfying, dynamically feasible, and suitable for solving the learned task. Our experiments do not consider time-varying constraints, but DPCC can handle them directly without modifications. In future work, we aim to include additional notions of safety, such as stability.

Acknowledgements

Ralf Römer gratefully acknowledges the support of the research group ConVeY funded by the German Research Foundation under grant GRK 2428. This work has been supported by the Robotics Institute Germany, funded by BMBF grant 16ME0997K.

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 *International Conference on Learning Representations (ICLR)*, 2023.

Jan-Hendrik Bastek, WaiChing Sun, and Dennis M Kochmann. Physics-Informed Diffusion Models. *arXiv preprint arXiv:2403.14404*, 2024.

Mokhtar S Bazaraa, Hanif D Sherali, and Chitharanjan M Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, 2006.

Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

Joao Carvalho, An T Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion Planning Diffusion: Learning and Planning of Robot Motions with Diffusion Models. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1916–1923, 2023.

Lili Chen, Shikhar Bahl, and Deepak Pathak. PlayFusion: Skill Acquisition via Diffusion from Language-Annotated Play. In *Conference on Robot Learning (CoRL)*, pages 2012–2029, 2023.

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 (RSS)*, 2023.

Jacob K Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained Synthesis with Projected Diffusion Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat GANs on Image Synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:8780–8794, 2021.

Giorgio Giannone, Akash Srivastava, Ole Winther, and Faez Ahmed. Aligning Optimization Trajectories with Diffusion Models for Constrained Design Generation. *Advances in Neural Information Processing Systems*, 36:51830–51861, 2023.

Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Xiaoyu Huang, Yufeng Chi, Ruofeng Wang, Zhongyu Li, Xue Bin Peng, Sophia Shao, Borivoje Nikolic, and Koushil Sreenath. DiffuseLoco: Real-Time Legged Locomotion Control with Diffusion from Offline Datasets. In *Conference on Robot Learning (CoRL)*, 2024.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with Diffusion for Flexible Behavior Synthesis. In *Proc. of the International Conference on Machine Learning*, volume 162 of *PMLR*, pages 9902–9915, 2022.

Xiaogang Jia, Denis Blessing, Xinkai Jiang, Moritz Reuss, Atalay Donat, Rudolf Lioutikov, and Gerhard Neumann. Towards Diverse Behaviors: A Benchmark for Imitation Learning with Human Demonstrations. In *International Conference on Learning Representations (ICLR)*, 2024.

Kota Kondo, Andrea Tagliabue, Xiaoyi Cai, Claudius Tewari, Olivia Garcia, Marcos Espitia-Alvarez, and Jonathan P How. Cgd: Constraint-Guided Diffusion Policies for UAV Trajectory Planning. *arXiv preprint arXiv:2405.01758*, 2024.

Dieter Kraft. Algorithm 733: TOMP–Fortran modules for optimal control calculations. *ACM Transactions on Mathematical Software (TOMS)*, 20(3):262–281, 1994.

François Mazé and Faez Ahmed. Diffusion models beat GANs on topology optimization. In *Proc. of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9108–9116, 2023.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved Denoising Diffusion Probabilistic Models. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 8162–8171. PMLR, 2021.

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. In *International Conference on Learning Representations (ICLR)*, 2023.

Thomas Power, Rana Soltani-Zarrin, Soshi Iba, and Dmitry Berenson. Sampling Constrained Trajectories Using Composable Diffusion Models. In *IROS 2023 Workshop on Differentiable Probabilistic Robotics: Emerging Perspectives on Robot Learning*, 2023.

James Blake Rawlings, David Q Mayne, Moritz Diehl, et al. *Model Predictive Control: Theory, Computation, and Design*, volume 2. Nob Hill Publishing Madison, 2017.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-Conditioned Imitation Learning using Score-based Diffusion Policies. In *Robotics: Science and Systems (RSS)*, 2023.

Ralf Römer, Lukas Brunke, Martin Schuck, and Angela P Schoellig. Safe Offline Reinforcement Learning using Trajectory-Level Diffusion Models. In *ICRA 2024 Workshop Back to the Future: Robot Learning Going Probabilistic*, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 2256–2265. PMLR, 2015.

Marc Toussaint. Robot Trajectory Optimization using Approximate Inference. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 1049–1056, 2009.

Julen Urain, Ajay Mandlekar, Yilun Du, Mahi Shafiullah, Danfei Xu, Katerina Fragkiadaki, Georgia Chalvatzaki, and Jan Peters. Deep Generative Models in Robotics: A Survey on Learning from Multimodal Demonstrations. *arXiv preprint arXiv:2408.04380*, 2024.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

Guangyao Zhou, Sivaramakrishnan Swaminathan, Rajkumar Vasudeva Raju, J Swaroop Guntupalli, Wolfgang Lehrach, Joseph Ortiz, Antoine Dedieu, Miguel Lázaro-Gredilla, and Kevin Murphy. Diffusion Model Predictive Control. *arXiv preprint arXiv:2410.05364*, 2024.