D-Cubed: Latent Diffusion Trajectory Optimisation for Dexterous Deformable Manipulation

Jun Yamada, Shaohong Zhong, Jack Collins, Ingmar Posner
Applied AI Lab
Oxford Robotics Institute
University of Oxford

Abstract: Mastering deformable object manipulation often necessitates the use of anthropomorphic, high-degree-of-freedom robot hands capable of precise, contact-rich control. However, current trajectory optimisation methods often struggle in these settings due to the large search space and the sparse task information available from shape-matching cost functions, particularly when contact is absent. In this work, we propose *D-Cubed*, a novel trajectory optimisation method using a latent diffusion model (LDM) trained from a task-agnostic play dataset to solve dexterous deformable object manipulation tasks. *D-Cubed* learns a skill-latent space that encodes short-horizon actions from a play dataset using a VAE and trains a LDM to compose the skill latents into a skill trajectory, representing a long-horizon action trajectory. To optimise a trajectory for a target task, we introduce a novel gradient-free guided sampling method that employs the Cross-Entropy method within the reverse diffusion process. In particular, *D-Cubed* samples a small number of noisy skill trajectories using the LDM for exploration and evaluates the trajectories in simulation. Then *D-Cubed* selects the trajectory with the lowest cost for the subsequent reverse process. This effectively explores promising solution areas and optimises the sampled trajectories towards a target task throughout the reverse diffusion process. Through empirical evaluation on a published benchmark of dexterous deformable object manipulation tasks, we demonstrate that *D-Cubed* outperforms traditional trajectory optimisation and competitive baseline approaches by a significant margin. Videos can be found at: https://applied-ai-lab.github.io/D-cubed.

Keywords: Trajectory Optimisation, Dexterous Deformable Object Manipulation, Latent Diffusion Model, Gradient-Free Guidance

1 Introduction

The realm of dexterous robot hand manipulation has made remarkable progress in recent years, in part due to advances in learning-based methods [1, 2, 3]. However, past research has focused predominantly on tasks that involve rigid objects [4, 5, 6, 7]. On the other hand, real-world manipulation tasks often present scenarios in which robots need to manipulate deformable objects, such as folding a piece of clothing [8], manipulating soft tissues [9] or shaping dough [10, 11].

One common approach to generating actions for a dexterous robot hand is trajectory optimisation, which optimises an action sequence by minimising a task-informed cost function. However, the application of trajectory optimisation is predominantly limited to rigid object manipulation [7] or relatively simple deformable object manipulation tasks with short horizons [12]. The primary challenges of optimising a trajectory for complex tasks such as those seen in dexterous deformable object manipulation stem from 1) the large search space due to the complexity of the task including the infinite dimensionality of deformable objects and high degrees of freedom (DoF) of the robot hand;

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

2) the large number of contacts associated with handling the objects; and 3) the limited task information that the cost function typically provides [13]. Commonly, the cost function to be optimised is defined as the distance between a target shape and the final shape of a deformable object after manipulation [14]. In this scenario, no task-relevant signal is available when no contact is made between the robot and the manipulated object, inhibiting the optimisation of a feasible trajectory.

In this work, we propose *D-Cubed*, Latent **D**iffusion for Trajectory Optimisation in **D**exterous **D**eformable Manipulation (see Fig 1). *D-Cubed* is a novel trajectory optimisation approach that leverages a latent diffusion model (LDM)[15] trained on a task-agnostic play dataset collected using a human hand. This dataset captures diverse representative robot hand motions, such as closing and opening the hand and moving individual fingers without object interaction, enabling reuse across a wide range of tasks. First, *D-Cubed* learns a skill-latent space that encodes short-horizon action sequences from the play dataset using a variational autoencoder (VAE). An LDM is then trained to compose these skill-latent representations into a skill trajectory, replicating motions of the robot hand found in the dataset. By leveraging the task-agnostic play dataset, the LDM, capable of generating diverse trajectories of meaningful robot hand motions, is applicable across diverse manipulation tasks (see Fig. 1). To find a performant action trajectory for a target task defined by a target object shape, given a shape-matching cost, we propose a novel gradient-free guided sampling method that employs a variation of the Cross-Entropy Method (CEM) [16, 17] within the reverse diffusion process. For each denoising step, the LDM generates a small number of noisy skill-latent trajectories to explore the solution space. Since the skill latent space is trained to represent smooth low-level action sequences, each skill in these noisy skill trajectories produces meaningful and consistent action trajectories that facilitate efficient exploration. These skill-latent trajectories are evaluated in a simulator using the shape-matching cost function, and the trajectory with the lowest task cost among the sampled trajectories is selected for further denoising in the reverse diffusion process. With each denoising step guided by the CEM, the noise in the chosen performant trajectory is gradually removed, refining it towards solutions with lower costs.

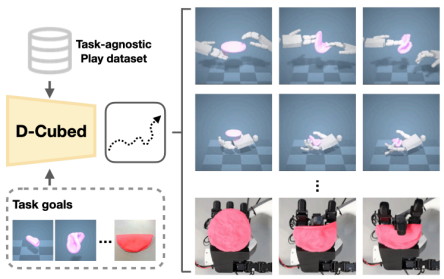

Figure 1: *D-Cubed* leverages a LDM trained from a task-agnostic play dataset to generate action trajectories for long-horizon dexterous deformable object manipulation tasks.

In summary, our contributions are three-fold: (1) we propose *D-Cubed*, a trajectory optimisation method using an LDM to solve challenging long-horizon dexterous manipulation tasks; (2) we introduce a novel gradient-free guided sampling method that employs the CEM within the reverse diffusion process to optimise a trajectory for a target task; and (3) we empirically demonstrate that *D-Cubed* significantly outperforms competitive baselines, including traditional trajectory optimisation methods such as gradient-based and sampling-based approaches.

2 Related Works

Several prior works note the importance of deformable object manipulation and introduce benchmark tasks for evaluating competing methodologies [12, 18, 19, 20, 14]. While most benchmarks focus on deformable object manipulation tasks with point-mass agents or parallel grippers that are incapable of dexterous manipulation, [14] proposes a suite of deformable object manipulation tasks with dexterous robot hands [21] built upon a differentiable physics engine.

Trajectory optimisation, including gradient-based and sampling-based, is a common approach to solving dexterous robot hand manipulation tasks by assuming access to an accurate dynamics model or simplified object geometries (e.g. [4, 5, 6, 7]). Gradient-based approaches directly optimise a task-informed cost function through a learned dynamics model [3, 22, 23] or differentiable simulator [12, 14, 24, 25] to find a performant action sequence. However, their application is limited to relatively

simple, short horizon tasks [12] due to convergence to locally optimal solutions caused by a lack of global task information, exacerbated by nonlinear contacts [14, 26]. Sampling-based methods such as CEM [16, 17] and MPPI [27] offer a simple approach by sampling actions for exploration, with MPPI applied to rigid object manipulation [7]. However, such sampling-based methods tend to be computationally expensive for large solution spaces, requiring many trajectory samples. Although previous work trains policies from expert demonstrations [28] or combine expert demonstrations with trajectory optimisation [14] to alleviate such issues, collecting expert demonstrations for each new task is considered expensive. Instead, in this work, a single task-agnostic play dataset containing representative hand movements is collected to form a structured skill-latent space that is used across a diverse range of tasks.

Diffusion models, a class of generative models, formulate data generation as an iterative denoising process [29, 15, 30]. Classifier guidance [31] is a common technique for using gradients to guide the sampling process of unconditional diffusion models to generate a desired sample, including a trajectory for a target task [32, 33, 34, 35]. However, classifier guidance struggles to guide sampling when gradients are inaccurate [36], such as when gradients are obtained from differentiable physics simulators [13]. In contrast, this work proposes gradient-free guidance that employs a variation of CEM to the reverse diffusion process for trajectory optimisation.

3 Preliminaries

Denoising Diffusion Probabilistic Models (DDPMs) DDPMs [15, 37] are a class of generative models that are trained by denoising a sequence of noise-corrupted inputs. For each training datum $\mathbf{x}_0 \sim q_{data}(\mathbf{x})$, the forward diffusion process constructs a Markov chain $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_N$ such that $q(\mathbf{x}_i|\mathbf{x}_{i-1}) = \mathcal{N}(\mathbf{x}_i; \sqrt{1-\beta_i}\mathbf{x}_{i-1}, \beta_i\mathbf{I})$ where β_i denotes a positive noise scale and subscript index i refers to the time step of the diffusion process. Then, the reverse process, which aims to remove noise from the noisy sample x_i, is defined as $p_{\boldsymbol{\theta}}(\mathbf{x}_{0:N}) = p(\mathbf{x}_N)\Pi_{i=1}^{N} p_{\boldsymbol{\theta}}(\mathbf{x}_{i-1}|\mathbf{x}_i)$, where $p(\mathbf{x}_N) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. The conditional distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_{i-1}|\mathbf{x}_i)$ is commonly modelled as a Gaussian distribution with mean $\mu_{\boldsymbol{\theta}}(\mathbf{x}_i, i)$ and covariance $\Sigma_\theta(\mathbf{x}_i, i)$:

$$\mu_\theta(\mathbf{x}_i, i) = \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_t}{1-\bar{\alpha}_i}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i}\mathbf{x}_i, \Sigma_{\boldsymbol{\theta}}(\mathbf{x}_i, i) = \sigma_i^2\mathbf{I} = \tilde{\beta}_i = \frac{1-\overline{\alpha}_i - 1}{1-\overline{\alpha}_i}\beta_i, \quad (1)$$

where $\alpha_i := 1 - \beta_i$ and $\overline{\alpha} := \Pi_{j=1}^{i}\alpha_j$.

Instead of predicting the noise, ϵ_i [15], added to the data, we train a diffusion model $G_\theta(x_i, i)$ to directly predict the clean datapoint, x_0, to simplify the objective [38]:

$$\mathcal{L}_{\text{diffusion}} = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), i \sim [1,N]}\left[||\mathbf{x}_0 - G(\mathbf{x}_i, i)||_2^2\right]. \quad (2)$$

Cross-Entropy Method (CEM) The CEM finds solutions to complex problems by iteratively refining a probability distribution, often modelled by a Gaussian distribution, to focus on promising solution areas. The CEM samples a population of solutions from a given distribution, evaluates them using a predefined cost function, and selects the top performing solutions to update the distribution.

4 Latent Diffusion Trajectory Optimisation

Given a representative dynamics model (e.g. a simulator), *D-Cubed* aims to find an action trajectory $\{\mathbf{a}^0, \mathbf{a}^1, \ldots, \mathbf{a}^T\}$, over time horizon T, that enables a dexterous robot hand to manipulate deformable objects to match a pre-defined goal shape. *D-Cubed* consists of three components: (1) a variational autoencoder (VAE) that learns a skill-latent, $\mathbf{z} \in \mathcal{Z}$, by encoding short-horizon action trajectories; (2) a latent diffusion model (LDM) that generates sequences of skill-latents that represent entire trajectories for exploration in the state space; and (3) trajectory optimisation using CEM within the reverse diffusion process for a target task. *D-Cubed* relies on a task-agnostic play dataset of action trajectories that cover a wide range of meaningful robot hand motions to learn a skill-latent space. This section describes the data collection process in Section 4.1, the LDM in Section 4.2, and the sampling method in Section 4.3. An overview of the method can be seen in Fig. 2.

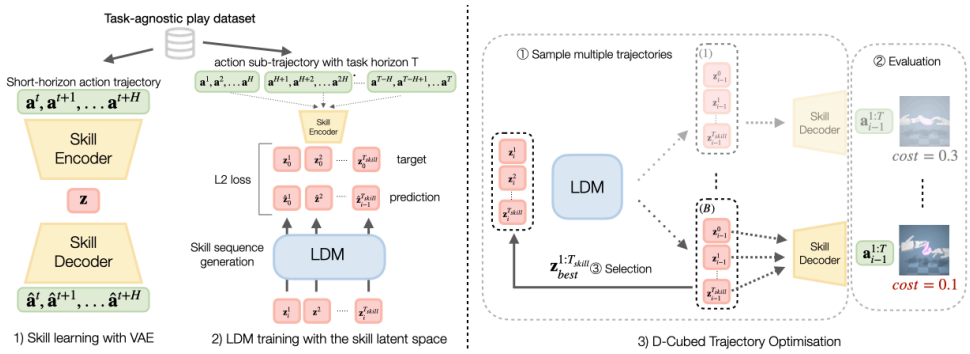

Figure 2: ***D-Cubed* overview.** (1) A VAE learns a skill latent representation $\mathbf{z}$ by reconstructing a short-horizon action sequence $\mathbf{a}^{t:t+H}$ randomly sampled from the task-agnostic play dataset. (2) A LDM learns to compose skills into a skill trajectory, representing a long-horizon action trajectory sampled from the dataset. (3) During trajectory optimisation, the LDM generates B skill trajectories $\{\mathbf{z}_i^{1:T_{skill}}\}^{|B|}$, where $T_{skill} = \frac{T}{H}$ is the length of skill trajectories. These trajectories are evaluated in a simulator, and the best sequence $\mathbf{z}_{best}^{1:T_{skill}}$ that minimises the cost is selected for the subsequent reverse process.

4.1 Data Collection

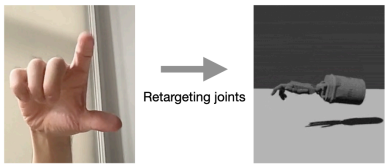

Figure 3: Data collection pipeline. Human hand joints are retargeted to robot hand joints to collect task-agnostic a play dataset designed to span the space of meaningful hand motions that form a skill latent space learnt by a VAE.

Collecting expert demonstrations for every new task is expensive due to the difficulty in manipulating deformable objects through teleoperation systems [14, 28]. To alleviate this issue, a single task-agnostic play dataset of robot hand trajectories $\mathcal{D}_{play}$ is collected per robot platform, without requiring interaction with deformable objects, allowing a human operator to readily collect the dataset without any training and enabling reuse across diverse tasks. This play dataset is designed to span the space of meaningful hand motions that can be performed by the given hardware and thus forming a skill latent space learnt by a VAE. This includes motions such as closing and opening the hand, moving individual fingers, as well as moving and flexing the wrist throughout the robot's workspace. The dataset is collected within 20 minutes by tracking the motion of a human hand and retargeting the human hand pose to a robot hand (see Fig. 3), similar to prior work [39]. This forms a dataset, denoted as D_{play}, comprising sequences of robot actions $\mathbf{a}_t$, where each action represents a relative change in the joint angles. For further details, see Appendix B.

4.2 Latent Diffusion Model as Skill Sampler

Long-horizon dexterous deformable manipulation tasks induce a large solution space due to the task complexity caused by the numerous contacts with deformable objects and the high DoF of a robot hand. As such, sampling low-level actions to find performant solutions is often infeasible because the sampled action trajectory is unlikely to correspond with meaningful robot hand motions that effectively explore the solution space. Instead, *D-Cubed* learns a skill latent space [40] that encodes a short-horizon action trajectory from the play dataset, which plays a significant role in efficiently exploring the search space of tasks. Specifically, a VAE, consisting of an encoder $q_\psi^{enc}(\mathbf{z}|\mathbf{a}^{t:t+H})$ and a decoder $p_\psi^{dec}(\mathbf{a}^{t:t+H}|\mathbf{z})$, is trained to reconstruct short-horizon action trajectories $\mathbf{a}^{t:t+H}$ randomly sampled from the play dataset $\mathcal{D}_{play}$ to learn the skill $\mathbf{z} \in \mathcal{Z}$, by optimising the ELBO objective:

$$\mathcal{L}^{\text{ELBO}} = \mathbb{E}_{\mathbf{z}\sim q_\psi(\mathbf{z}|\mathbf{a}^{t:t+H})} \log p_\psi^{dec}(\mathbf{a}^{t:t+H} \mid \mathbf{z}) - D_{\text{KL}}\left[q_\psi(\mathbf{z} \mid \mathbf{a}^{t:t+H}) \| p(\mathbf{z})\right]$$

where $p(\mathbf{z})$ is a Gaussian prior over the latent representation and H is the length of the short-horizon action sequence ($H = 10$). See Appendix C.1 for further details on the hyperparameters of the VAE.

Since the skill latent representation only encodes short-horizon actions of the robot hand, composing multiple skill latent representations into a long-horizon action trajectory is necessary for efficient exploration (e.g. to make meaningful contacts with an object). To achieve this, an LDM is trained to compose a sequence of skill-latent representations $\mathbf{z}^{1:T_{skill}}$, where $T_{skill} = \frac{T}{H}$ is the length of the skill trajectory, which reconstruct robot hand trajectories from the play dataset. The LDM is trained to optimise the following objective:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{\mathbf{z}^{1:T_{skill}} \sim \mathcal{D}_{play}, i \sim [1,N]} \left[||\mathbf{z}_0^{1:T_{skill}} - G_\theta(\mathbf{z}_i^{1:T_{skill}}, i)||_2^2 \right]. \tag{3}$$

where N is the number of diffusion steps, $\mathbf{z}_0^{1:T_{skill}}$ is a clean skill-latent trajectory, and $\mathbf{z}_i^{1:T_{skill}}$ is a noisy skill-latet trajectory after i forward diffusion steps. An LDM is chosen as it is a generative model proven to be capable of representing a complex multimodal distribution, like that of the play dataset [41]. In this work, we employ a transformer model as the backbone of the noise prediction model $G_\theta(\cdot, i)$ (see Appendix C.2 for further details).

By leveraging a task-agnostic play dataset that captures meaningful hand motions, the trained LDM can generate sequences of robot hand actions that effectively explore the action space and are reusable across a diverse range of tasks. As a result, the LDM needs to be trained only once from the single play dataset and can be applied to all tasks without retraining.

4.3 Trajectory Optimisation using Gradient-Free Guided Sampling

Traditional trajectory optimisation often struggles to solve dexterous deformable object manipulation tasks due to the large search space. This is further amplified by the limited global task information available from a cost function. Using the capability of the LDM with skill-latent space, the LDM generates diverse skill trajectories that represent long-horizon action trajectories of meaningful robot hand motions, leading to effective exploration of the state space. However, to solve a target task, guidance is required to direct the diffusion sampling process to converge towards high-performing trajectories. While classifier guidance is a common technique for guiding the reverse process using gradients, inaccurate [36] or noisy gradients such as those obtained from differentiable physics simulators [13] are unable to successfully guide the reverse diffusion process (see Section 5.3 for experimental results). To avoid such issues, we propose gradient-free guided sampling that employs the CEM [42] within the reverse diffusion process to optimise a trajectory for a target task. The reverse process can be viewed as analogous to the CEM optimisation steps, as *D-Cubed* evaluates generated action trajectories in a simulator and updates the parameters of a Gaussian distribution based on the trajectory with the lowest cost for each diffusion step (see Fig. 2 and Algorithm 1).

In particular, for each reverse step, a small number of noisy skill trajectories $\mathbf{z}^{1:T_{skill}}$ are sampled from a Gaussian distribution with a mean $\boldsymbol{\mu}_i$ predicted by the LDM (Line 3, 9), where $T_{skill} = \frac{T}{H}$ represents the horizon length of the skill-latent representations. Crucially, during the early stages of the reverse process, *D-Cubed* focuses on exploring the search space. This is because the variance $\Sigma_{\boldsymbol{\theta}}$ of the Gaussian distribution, determined by the noise scheduler (see Equation 1), is large, thereby generating diverse trajectories. During later steps of the reverse process, *D-Cubed* attempts to refine the trajectories for a target task as a result of the small variance of the scheduled distribution. The generated skill trajectories are decoded using the VAE into low-level action trajectories that are then evaluated in the simulator to obtain their respective scores (Line 5). While the generated skill sequences are noisy regarding their composition at early reverse diffusion steps, each short-horizon action sequence decoded from the skill latent representations remains smooth, which effectively promotes meaningful trajectories from the search space for efficient exploration.

Similar to how the CEM updates a Gaussian distribution based on the top-performing samples for each optimisation step (see Section 3), *D-Cubed* also updates a Gaussian distribution to search for more promising solution areas by predicting the mean $\boldsymbol{\mu}_i$ using the LDM given the trajectory with the lowest cost $\mathbf{z}_{best}^{1:T_{skill}}$ as input (see Equation 4 and Line 6):

$$\mu_\theta(\mathbf{z}_{best}^{1:T_{skill}}, i) = \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_t}{1-\bar{\alpha}_i} G_\theta(\mathbf{z}_{best}^{1:T_{skill}}, i) + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i} \mathbf{z}_{best}^{1:T_{skill}} \tag{4}$$

Algorithm 1 *D-Cubed* Trajectory Optimisation

1: **Require**: denoising model, $G_{\boldsymbol{\theta}}$; target state of deformable objects, $\mathbf{s}_{\text{target}}$, $T_{skill} = \frac{T}{H}$
2: **Initialise:** $C_{\text{best}} = \infty, \boldsymbol{\mu}_{\text{best}} = \text{None}$
3: $\{\mathbf{z}_N^1, \ldots, \mathbf{z}_N^{T_{skill}}\}^{|B|} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Sample B initial sequences of skill latent representations
4: **for** $i = N, N-1, \ldots, 1$ **do**
5: $\quad \mathbf{z}_{best}^{1:T_{skill}} \leftarrow \text{FINDBESTLATENTS}(\{\mathbf{z}_i^{1:T_s}\}^{|B|})$ ▷ Choose the best sequence of skill latents (Appendix E)
6: $\quad \boldsymbol{\mu}_i \leftarrow \boldsymbol{\mu}_\theta(\mathbf{z}_{best}^{1:T_{skill}})$ ▷ Predict a mean of a Gaussian distribution (see Eq. 4)
7: $\quad \text{cost} = \text{evaluate}(q_\psi^{dec}(\mathbf{a}^{1:T}|\boldsymbol{\mu}_i))$ ▷ Evaluate the predicted mean
8: $\quad$ **if** cost $<$ C_{best} **then** $\boldsymbol{\mu}_{\text{best}} \leftarrow \boldsymbol{\mu}_i$, $\text{C}_{best} \leftarrow$ cost
9: $\quad \{\mathbf{z}_{i-1}^1, \ldots, \mathbf{z}_{i-1}^{T_{skill}}\}^{|B|} \sim \mathcal{N}(\boldsymbol{\mu}_{best}, \sigma_{i-1}^2 \mathbf{I})$ ▷ Sample a batch B of sequences of skill latents
10: **return** $p_\psi^{dec}(\mathbf{a}^{1:T}|\mu_{\mathbf{best}})$

Although CEM normally requires several top-performing samples to compute the mean and variance of the Gaussian distribution, *D-Cubed* needs only a single top-performing sample because the mean at the next diffusion step is determined by the prediction from the LDM (see Line 6). Furthermore, in *D-Cubed*, the variance is updated based on a noise schedule (see Equation 1) for each reverse step. Intuitively, by choosing the lowest-cost trajectory for each diffusion step, the LDM removes noise from performant noisy trajectories for the subsequent reverse process. This leads to further refinement of the trajectory and minimisation of the cost function by exploring more promising solution areas.

Lastly, the mean $\boldsymbol{\mu}_i$ of the Gaussian distribution predicted by the LDM does not necessarily compose a better distribution for the following reverse steps because the LDM may make a poor prediction, leading to trajectory samples that have a higher cost. To address this issue, inspired by a variant of CEM [43, 44], the mean of the distribution is updated only when the current predicted mean $\boldsymbol{\mu}_i$ has a lower cost than the previous best mean $\boldsymbol{\mu}_{\text{best}}$ (see Line 8). This optimisation method, when paired with a LDM is empirically shown in Section 5.3 to generate performant trajectories.

5 Experiments

Our experimental evaluation is designed to answer the following questions: (1) How effective is *D-Cubed* in generating trajectories for dexterous deformable object manipulation? (2) How does our method compare to competitive baselines including traditional trajectory optimisation approaches and other methods that do not require expert demonstrations? (3) How important are the design decisions of *D-Cubed* in generating high-performance trajectories? In addition, we conduct real-world experiments to qualitatively assess whether the optimised action trajectories are executable on real hardware (see Appendix A.2). For further experimental details and resuls, see Appendix 4.3.

5.1 Experimental Setup

Simulated Environments: We evaluate *D-Cubed* on a publicly available benchmark that consists of a suite of six challenging dexterous deformable object manipulation tasks introduced in prior work [14]. The benchmark consists of three single-hand tasks (*Folding*, *Flip*, *Wrap*) and three dual-hand tasks (*Dumpling*, *Bun*, *Rope*). For dual-arm tasks, *D-Cubed* uses the same trained LDM to independently generate action sequences for each arm, producing two coordinated trajectories for each hand simultaneously. The cost function dictated by the benchmark is the Sinkhorn Divergence [45] which measures the difference between the manipulated and the target object shape using point clouds. Our experimental setup closely follows that of prior work [14] in that we evaluate *D-Cubed* and competitive baselines on tasks intended to form 5 different target shapes for each task. For more details on the tasks, see Appendix F and the prior work [14].

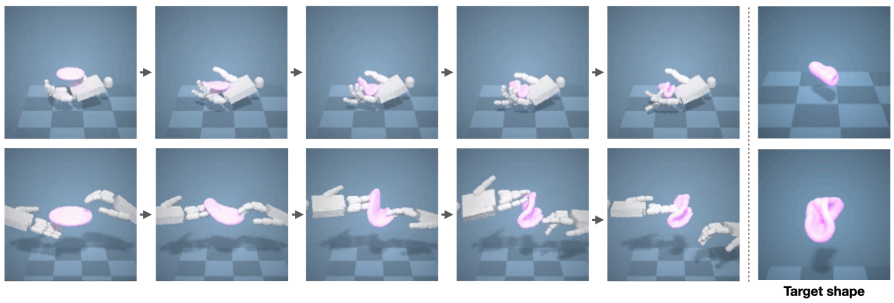

Figure 4: **Qualitative results of *D-Cubed*.** (Top) *Flip* and (Bottom) *Dumpling* task. In *Flip* task, the hand, using primarily the wrist and finger DoFs, is able to fold the plasticine into a configuration that is representative of the goal state. In *Dumpling* task, using two hands to deform the stationary plasticine, *D-Cubed*is able to manipulate the plasticine close to the target shape.

Env	Folding	Rope	Bun	Dumpling	Wrap	Flip
Grad TrajOpt	0.032 ± 0.061	0.079 ± 0.026	0.000 ± 0.000	0.032 ± 0.061	0.079 ± 0.026	0.000 ± 0.000
MPPI	0.002 ± 0.005	0.000 ± 0.000	0.000 ± 0.000	0.021 ± 0.042	0.000 ± 0.000	0.000 ± 0.000
Skill-based MPPI	0.020 ± 0.002	0.000 ± 0.000	0.048 ± 0.012	0.052 ± 0.034	0.000 ± 0.000	0.409 ± 0.001
PPO	0.361 ± 0.173	0.460 ± 0.257	0.069 ± 0.117	0.000 ± 0.000	0.000 ± 0.000	0.223 ± 0.328
LDM w/ Gradient guidance	0.050 ± 0.038	0.001 ± 0.001	0.019 ± 0.018	0.009 ± 0.014	0.016 ± 0.016	0.448 ± 0.080
Diffusion-ES	0.403 ± 0.227	0.192 ± 0.059	0.273 ± 0.092	0.179 ± 0.057	0.305 ± 0.007	0.678 ± 0.032
D-Cubed	$\mathbf{0.871 \pm 0.021}$	$\mathbf{0.741 \pm 0.031}$	$\mathbf{0.704 \pm 0.012}$	$\mathbf{0.699 \pm 0.037}$	$\mathbf{0.512 \pm 0.032}$	$\mathbf{0.909 \pm 0.025}$

Table 1: The averaged normalised improved EMD and standard deviation over 3 seeds are reported for each method. The scores for *Grad TrajOpt* and *PPO* are taken from previous work [14].

Evaluation Metric: Following [14], we report the normalised improvement in Earth-Mover distance (EMD) approximated by the Sinkhorn Divergence, calculated as $d(t) = \frac{d_0 - d_t}{d_0}$ where d_0 and d_t are the initial and current EMD values. When the normalised improvement is 1, the deformable object perfectly matches the target shape. A negative score from a large discrepancy is set to 0.

5.2 Baselines

We compare *D-Cubed* with the following state-of-the-art and competitive baselines that represent competing approaches capable of generating trajectories (for further details, see Appendix D):

- **Grad TrajOpt**: A gradient-based trajectory optimisation [12] that utilises the first-order gradients available from the benchmark simulator.
- **MPPI**: A sampling-based trajectory optimisation method [27] that samples a batch of short-horizon trajectories from a Gaussian distribution and updates the parameters of the distribution based on the top-performing trajectories.
- **Skill-based MPPI**: *Skill-based MPPI* is similar to *MPPI*, but operates in the skill-latent space. In contrast to *D-Cubed*, which uses an LDM to sample meaningful skill compositions, *Skill-based MPPI* must optimise such meaningful compositions by sampling diverse trajectories.
- **PPO**: Proximal Policy Optimisation (PPO) [46] generates closed-loop action sequences from point cloud inputs as an alternative to trajectory optimisation.
- **LDM w/ Gradient Guidance**: Using the learnt LDM to optimise a trajectory through the reverse process with gradient guidance using gradients from the simulator.
- **Diffusion-ES**: A concurrent method [47] that also proposes a gradient-free sampling method based on evolutionary search with a truncated diffusion process.

5.3 Trajectory Optimisation Results

***D-Cubed* is the highest performing method across all tasks.** Table 1 shows the normalised improvement in EMD for each task. *D-Cubed* outperforms the baseline methods by a significant margin in all tasks. These results indicate that *D-Cubed* effectively combines learnt skills to explore

the solution space using the LDM and exploits diverse sampled trajectories through gradient-free guided sampling to minimise a task-informed cost (see Appendix 4.3).

Traditional optimisation methods and RL baselines struggle due to poor exploration. *Grad TrajOpt* rarely obtains useful gradients from the shape-matching cost and struggles to find performant trajectories. *MPPI* and *Skill-based MPPI* focus on local short-horizon optimisation, missing better long-horizon solutions. In contrast, *D-Cubed* optimises the entire trajectory globally, enabling the dexterous hand to discover high-performing trajectories. This highlights the advantage of using an LDM trained to compose meaningful skill sequences, effectively narrowing the search to promising trajectories. Similarly, *PPO* underperforms relative to *D-Cubed* due to RL's difficulty in exploring high-dimensional state spaces, and must also be retrained for each task.

Gradient-guided diffusion and Diffusion-ES suffer from noisy or limited guidance. *LDM w/ Gradient guidance* performs poorly in all tasks, primarily due to noisy gradients from the simulator [13] and the lack of informative task signals, especially in tasks requiring extensive search or involving no object contact. *Diffusion-ES*, while initially generating clean trajectories and perturbing them through short diffusion steps, also struggles to explore the solution space sufficiently. The small perturbations limit its ability to recover when initial trajectories miss object contact.

Skill-level diffusion in *D-Cubed* enables efficient exploration. In contrast, *D-Cubed* explores the solution space more effectively by generating noisy skill trajectories at the beginning of the reverse diffusion process. This higher-level representation allows for broader exploration in the state space compared to low-level action trajectories generated by Diffusion-ES, leading to more efficient and successful planning outcomes.

5.4 Ablation Studies

Number of trajectories sampled during the reverse diffusion process. Fig. 5 shows the mean and Interquartile Mean (IQM) with 95% confidence intervals (CIs) [48] of normalised improvement in EMD averaged across all six tasks with different numbers of trajectories sampled during the reverse diffusion process (Line 9 in Algorithm 1). Fig. 5 indicates that sampling more trajectories during the reverse process significantly improves performance because *D-Cubed* can more effectively search the solution space.

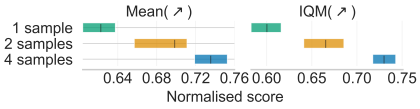

Figure 5: Ablation of the number of trajectories sampled during the reverse diffusion process (line 9 in Algorithm 1).

Efficacy of skill latent representations. Fig. 6 shows the performance of *D-Cubed* with and without the use of a skill latent across the six tasks. It is evident in simpler tasks, such as *Flip* and *Folding*, that the use of a skill-latent space does not significantly improve results. However, the performance of *D-Cubed* on *Rope* and *Wrap*, considerably harder tasks, is substantially better when using a skill-latent space. We reason that this is because *D-Cubed* with a skill-latent space can effectively search the solution space and tackle hard exploration problems.

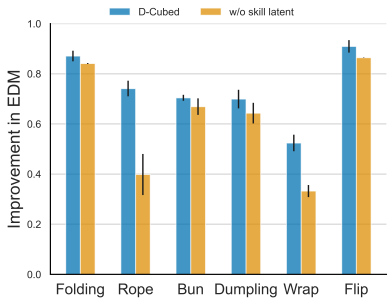

Figure 6: Comparison of *D-Cubed* w/ and w/o skill latent representations.

6 Conclusion

We present *D-Cubed*, a new trajectory optimisation method to solve long-horizon dexterous deformable object manipulation tasks using a latent diffusion model trained from a task-agnostic play dataset. *D-Cubed* leverages a novel gradient-free guided sampling method that adapts the CEM within the reverse diffusion process. The experimental results show that *D-Cubed* outperforms the traditional and competitive trajectory optimisation baselines by a significant margin, showing great promise for other challenging trajectory optimisation tasks.

6.1 Limitations

Experimental studies show that *D-Cubed* can generate a performant action sequence for dexterous deformable object manipulation tasks. However, *D-Cubed* cannot find realistic trajectories for all tasks because some benchmark tasks permit non-physical behaviours, such as table penetration and object floating. As a result, we are unable to conduct real-world experiments for every task, although we successfully demonstrate transfer for Flip (see Appendix A.2). Another limitation is the time required to generate a desired trajectory, which heavily depends on simulation evaluation time. However, using a faster, parallel simulator [49, 50] would greatly improve the speed of optimising a trajectory. Finally, *D-Cubed* is open-loop, making it unable to accommodate for discrepancies observed when executing the trajectory. In the future, we aim to close the loop, potentially by distilling the trajectories into a policy.

Acknowledgments

This work was supported by a UKRI/EPSRC Programme Grant [EP/V000748/1]. We would like to acknowledge the use of the University of Oxford Advanced Research Computing (ARC) and SCAN facilities in carrying out this work.

References

[1] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[2] T. Chen, J. Xu, and P. Agrawal. A system for general in-hand object re-orientation. In A. Faust, D. Hsu, and G. Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 297–307. PMLR, 08–11 Nov 2022.

[3] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.

[4] I. Mordatch, Z. Popović, and E. Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, page 137–144, 2012. ISBN 9783905674378.

[5] Y. Bai and C. K. Liu. Dexterous manipulation using both palm and fingers. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1560–1565, 2014. doi:10.1109/ICRA.2014.6907059.

[6] B. Sundaralingam and T. Hermans. Relaxed-rigidity constraints: kinematic trajectory optimization and collision avoidance for in-grasp manipulation. *Autonomous Robots*, 43(2):469–483, Feb 2019. doi:10.1007/s10514-018-9772-z.

[7] H. J. Charlesworth and G. Montana. Solving challenging dexterous manipulation tasks with trajectory optimisation and reinforcement learning. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1496–1506. PMLR, 18–24 Jul 2021.

[8] Y. Tsurumine, Y. Cui, E. Uchibe, and T. Matsubara. Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation. *Robotics and Autonomous Systems*, 112, 11 2018. doi:10.1016/j.robot.2018.11.004.

[9] A. Pore, E. Tagliabue, M. Piccinelli, D. Dall'Alba, A. Casals, and P. Fiorini. Learning from demonstrations for autonomous soft-tissue retraction. In *2021 International Symposium on Medical Robotics (ISMR)*, pages 1–7. IEEE, 2021.

[10] H. Shi, H. Xu, Z. Huang, Y. Li, and J. Wu. RoboCraft: Learning to See, Simulate, and Shape Elasto-Plastic Objects with Graph Networks. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi:10.15607/RSS.2022.XVIII.008.

[11] H. Shi, H. Xu, S. Clarke, Y. Li, and J. Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 642–660. PMLR, 06–09 Nov 2023.

[12] Z. Huang, Y. Hu, T. Du, S. Zhou, H. Su, J. B. Tenenbaum, and C. Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=xCcdBRQEDW.

[13] R. Antonova, J. Yang, K. M. Jatavallabhula, and J. Bohg. Rethinking optimization with differentiable simulation from a global perspective. In *6th Annual Conference on Robot Learning*, 2022. URL https://openreview.net/forum?id=Y_YUEEQMjQK.

[14] S. Li, Z. Huang, T. Chen, T. Du, H. Su, J. B. Tenenbaum, and C. Gan. Dexdeform: Dexterous deformable object manipulation with human demonstrations and differentiable physics. In *The Eleventh International Conference on Learning Representations*, 2023.

[15] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[16] R. Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99(1):89–112, 1997. ISSN 0377-2217. doi:https://doi.org/10.1016/S0377-2217(96)00385-2.

[17] M. Kobilarov. Cross-entropy motion planning. *The International Journal of Robotics Research*, 31(7):855–871, 2012. doi:10.1177/0278364912444543. URL https://doi.org/10.1177/0278364912444543.

[18] X. Lin, Y. Wang, J. Olkin, and D. Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 432–448. PMLR, 2021.

[19] S. Chen, Y. Xu, C. Yu, L. Li, X. Ma, Z. Xu, and D. Hsu. Daxbench: Benchmarking deformable object manipulation with differentiable physics. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1NAzMofMnWl.

[20] D. Blanco-Mulero, O. Barbany, G. Alcan, A. Colomé, C. Torras, and V. Kyrki. Benchmarking the sim-to-real gap in cloth manipulation, 2024.

[21] ShadowRobot. Shadowrobot dexterous hand., 2015. URL https://www.shadowrobot.com/products/dexterous-hand/.

[22] V. Kumar, E. Todorov, and S. Levine. Optimal control with learned local models: Application to dexterous manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–383, 2016. doi:10.1109/ICRA.2016.7487156.

[23] J. Yamada, C.-M. Hung, J. Collins, I. Havoutis, and I. Posner. Leveraging scene embeddings for gradient-based motion planning in latent space. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[24] Y. Qiao, J. Liang, V. Koltun, and M. Lin. Differentiable simulation of soft multi-body systems. *Advances in Neural Information Processing Systems*, 34:17123–17135, 2021.

[25] Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand. Difftaichi: Differentiable programming for physical simulation. In *International Conference on Learning Representations*, 2019.

[26] X. Lin, Z. Huang, Y. Li, D. Held, J. B. Tenenbaum, and C. Gan. Diffskill: Skill abstraction from differentiable physics for deformable object manipulations with tools. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Kef8cKdHWpP.

[27] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440, 2016. doi:10.1109/ICRA.2016.7487277.

[28] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[29] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265. PMLR, 07–09 Jul 2015.

[30] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

[31] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[32] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[33] M. Janner, Y. Du, J. Tenenbaum, and S. Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[34] Z. Liang, Y. Mu, M. Ding, F. Ni, M. Tomizuka, and P. Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. In *International Conference on Machine Learning*, pages 20725–20745. PMLR, 2023.

[35] M. Rigter, J. Yamada, and I. Posner. World models via policy-guided trajectory diffusion. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

[36] C.-H. Chao, W.-F. Sun, B.-W. Cheng, Y.-C. Lo, C.-C. Chang, Y.-L. Liu, Y.-L. Chang, C.-P. Chen, and C.-Y. Lee. Denoising likelihood score matching for conditional score-based data generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=LcF-EEt8cCC.

[37] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[38] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[39] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox. Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170. IEEE, 2020.

[40] K. Pertsch, Y. Lee, and J. Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pages 188–204. PMLR, 2021.

[41] L. Chen, S. Bahl, and D. Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning*, pages 2012–2029. PMLR, 2023.

[42] R. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Method. Comput. Appl. Prob.*, 1(2):127–190, sep 1999. ISSN 1387-5841. doi:10.1023/A:1010091220143. URL https://doi.org/10.1023/A:1010091220143.

[43] C. Pinneri, S. Sawant, S. Blaes, J. Achterhold, J. Stueckler, M. Rolinek, and G. Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning*, pages 1049–1065. PMLR, 2021.

[44] I. Szita and A. Lorincz. Online variants of the cross-entropy method, 2008.

[45] T. Séjourné, J. Feydy, F.-X. Vialard, A. Trouvé, and G. Peyré. Sinkhorn divergences for unbalanced optimal transport. *arXiv preprint arXiv:1910.12958*, 2019.

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[47] B. Yang, H. Su, N. Gkanatsios, T.-W. Ke, A. Jain, J. Schneider, and K. Fragkiadaki. Diffusion-es: Gradient-free planning with diffusion for autonomous driving and zero-shot instruction following. *arXiv preprint arXiv:2402.06559*, 2024.

[48] R. Agarwal, M. Schwarzer, P. S. Castro, A. Courville, and M. G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021.

[49] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance GPU based physics simulation for robot learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=fgFBtYgJQX_.

[50] G. Authors. Genesis: A universal and generative physics engine for robotics and beyond, December 2024. URL https://github.com/Genesis-Embodied-AI/Genesis.

[51] K. Shaw, A. Agarwal, and D. Pathak. Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning. *Robotics: Science and Systems (RSS)*, 2023.

[52] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[53] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[54] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

[55] A. Karpathy. NanoGPT. https://github.com/karpathy/nanoGPT, 2022.

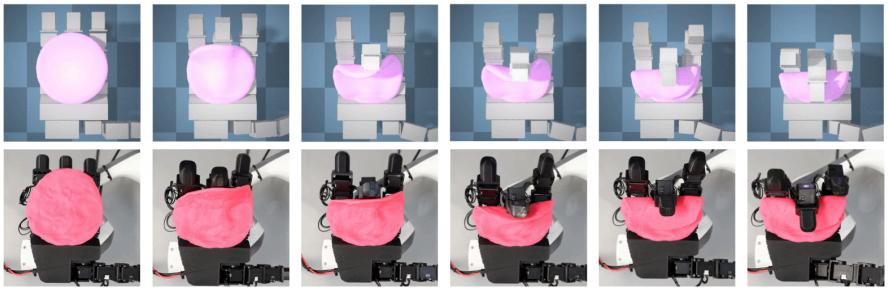

Figure 8: Qualitative results of *D-Cubed* using the LEAP hand in a real-world experiment. The LEAP hand effectively deforms the object, exhibiting similar deformation as observed in the simulation.

A Additional Analysis

A.1 Ablation of Additional Gradient Guidance

The performance disparity when *D-Cubed* also uses gradient guidance with the proposed gradient-free sampling is reported in Fig. 7. As gradient guidance does not demonstrate a statistically significant improvement in the score compared to *D-Cubed* without gradient guidance, the increased time required by the simulator to calculate gradients used for gradient guidance is not warranted. Additionally, this result adds further evidence to the premise that gradients from differentiable simulators are often sparse and uninformative [13].

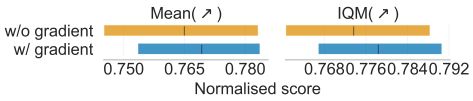

Figure 7: Comparison of performance with and without additional gradient guidance in our method. We report Mean and Interquartile Mean (IQM) of improvement in EDM averaged across all six tasks.

A.2 Qualitative Results in Real-World Environments

We qualitatively investigate whether an optimised trajectory from simulation can be transferred to real-world environments. Due to hardware limitations, we use a LEAP hand, a low-cost dexterous hand [51], instead of the Shadow hand [21]. Thus, the simulator is modified by replacing the Shadow hand with the LEAP hand. We evaluate the trajectory transfer on the *Flip* task, as this benchmark task makes the least number of simplifying assumptions compared to the real world. The other tasks permit object interpenetration with the table and unrealistic floating behaviour of the deformable object, rendering the evaluation impractical. In this experiment, given the known start state of the deformable object, we transfer the sequence of actions optimised in simulation to the real-world environment and control the hand in an open-loop manner. As shown in Fig. 8, the hand successfully *flips* the deformable object so that the object is folded within the hand in the real-world environment.

B Data collection Details

A task-agnostic play dataset of representative robot hand motions, including finger closing and opening and wrist movement, is collected. We use RGB data from a RealSense D435 camera to track human hand motion and re-target the human hand pose to a robot hand in the SAPIEN simulator [52], inspired by prior work [39]. We collect the play data for a duration of only 20 minutes, which corresponds to around $50K$ data points.

C Training Details

C.1 VAE

The VAE encoder and decoder both consist of a 4 layer LSTM [53] with 256 neurons per layer. In this work, we use a subsequence of actions with $H = 10$ to learn the skill-latent space. The VAE is trained using the Adam optimiser [54] with a learning rate of $1\text{e}{-4}$.

C.2 Latent Diffusion Models

We use the transformer architecture used in NanoGPT [55]. We report further hyperparameter details of the transformer denoiser network and diffusion in Table 2 and Table 3.

Table 2: Transformer Denoiser Network Hyperparameters

Parameter	Value
Optimiser	Adam
Learning rate	1e-4
Minibatch size	256
Embedding dimension	312
Batch size	256
Number of layers	6
Self-attention heads	4

Table 3: Diffusion Hyperparameters

Parameter	Value
Number of diffusion timesteps	200
Noise schedule	cosine
Noise schedule parameters s	0.008

C.3 D-Cubed Details

We report hyperparameters of D-Cubed used during trajectory optimisation steps in Table 4.

Table 4: D-Cubed Optimisation Hyperparameters

Parameter	Value
Number of diffusion timesteps	200
Number of samples	5

D Baseline Method Details

As we report the scores for gradient-based trajectory optimisation (TrajOpt) and PPO from prior work [14], we refer the reader to the prior work for further details.

D.1 MPPI

MPPI baseline samples 30 trajectories with a horizon of 15 steps. These parameters are chosen because they result in an optimisation time similar to *D-Cubed*. We report the hyperparameters for

MPPI in Table 5 in detail. In this experiment, we employ the publicly available MPPI implementation[1].

Table 5: MPPI and Skill-based MPPI Hyperparameters

Parameter	Value
planning horizon	15
Number of samples	30
Temperature	1.0
Initial noise mean	0.0
Initial noise std	1.0

D.2 Skill-based MPPI

Skill-based MPPI baseline samples skill-latent representations for effective exploration of the state space. We use the same hyperparameters as those of MPPI, except that the action sampled from a Gaussian distribution is skill-latent representations instead of low-level actions.

D.3 LDM w/ Gradient Guidance

LDM w/ Gradient Guidance baseline leverages gradient guidance [31] to generate a desired trajectory. In particular, first-order gradients from the differentiable physics simulator are used to guide the reverse process of the latent diffusion model. In our experiments, we denoise a noisy trajectory without gradient guidance for the first half of the diffusion steps so that a relatively clean trajectory can be obtained. For the rest of the diffusion steps, the following gradient guidance is applied:

$$\nabla_{\mathbf{x}_i} \log p_{\alpha_i}(\mathbf{x}_i|y) = \nabla_{\mathbf{x}_i} \log p_{\alpha_i}(\mathbf{x}_i) + \gamma \nabla_{\mathbf{x}_i} \log p(y|\mathbf{x}_i). \tag{5}$$

where y is the cost of the trajectory, $\nabla_{\mathbf{x}_i} \log p(y|\mathbf{x}_i)$ corresponds to the first-order gradients obtained from differentiable physics simulators, and γ is the scale of the gradient guidance. In our experiment, we use $\gamma = 1\text{e}{-4}$.

D.4 Diffusion-ES

Diffusion-ES, concurrent research [47], also optimises a trajectory using gradient-free guided sampling with a truncated diffusion process. While the prior work chooses the last trajectory of the optimisation process as output, we observe that it is often worse than the trajectories found in the middle of optimisation iterations. Thus, we report the score of the best trajectory found during the trajectory optimisation process. The hyperparameters used in Diffusion-ES is reported in Table 6. Following the original work [47], initially, the diffusion mutation steps start from 5 and the mutation step is linearly decayed to 1 over 200 search steps.

Table 6: Diffusion ES Hyperparameters

Parameter	Value
Mutation diffusion start steps	5
Mutation diffusion final steps	1
Population	5
Optimisation steps	200

[1] https://github.com/UM-ARM-Lab/pytorch_mppi/tree/master

E Gradient-Free Guided Sampling for Trajectory Optimisation

Algorithm 2 is a complete version of gradient-free guided sampling for trajectory optimisation in *D-Cubed*. To determine the best sequence of skill latent representations $\mathbf{z}_{best}^{1:T_{skill}}$, each skill sequence in a batch of B skill trajectories is evaluated in simulation, and the skill sequence that minimises a cost is selected as the best sequence (Line 5).

Algorithm 2 Gradient-Free Guided Sampling for Trajectory Optimisation in Reverse Diffusion Process

1: **Require**: denoising model, $G_{\boldsymbol{\theta}}$; target state of deformable objects, $\mathbf{s}_{\text{target}}$, $T_{skill} = \frac{T}{H}$
2: **Initialise:** $C_{\text{best}} = \infty$, $\boldsymbol{\mu}_{\text{best}} = \text{None}$
3: $\{\mathbf{z}_N^1, \ldots, \mathbf{z}_N^{T_{skill}}\}^{|B|} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Sample B initial sequences of latent representations
4: **for** $i = N, N-1, \ldots, 1$ **do**
5: $\quad \mathbf{z}_{best}^{1:T_{skill}} \leftarrow$ FINDBESTLATENTS$(\{\mathbf{z}_i^{1:T_s}\}^{|B|})$ ▷ Choose the best sequence of skill latents
6: $\quad \boldsymbol{\mu}_i \leftarrow \boldsymbol{\mu}_\theta(\mathbf{z}_{best}^{1:T_{skill}})$ ▷ Predict a mean of a Gaussian distribution (see Eq. 4)
7: $\quad \text{cost} = \text{evaluate}(q_\psi^{dec}(\mathbf{a}^{1:T}|\boldsymbol{\mu}_i))$ ▷ Evaluate the predicted mean
8: $\quad$ **if** cost $<$ C$_{\text{best}}$ **then**
9: $\quad\quad \boldsymbol{\mu}_{\text{best}} \leftarrow \boldsymbol{\mu}_i$, C$_{best} \leftarrow$ cost
10: $\quad \{\mathbf{z}_{i-1}^1, \ldots, \mathbf{z}_{i-1}^{T_{skill}}\}^{|B|} \sim \mathcal{N}(\boldsymbol{\mu}_{best}, \sigma_{i-1}^2 \mathbf{I})$ ▷ Sample a batch B of sequences of skill latents
11: **return** $p_\psi^{dec}(\mathbf{a}^{1:T}|\mu_{\mathbf{best}})$
12: **function** FINDBESTLATENTS$(\{\mathbf{z}^1, \ldots, \mathbf{z}^{T_{skill}}\}^{|B|})$
13: $\quad \text{cost}_{\text{best}}, \mathbf{z}_{\text{best}} = \infty, \text{None}$
14: $\quad$ **for** $\mathbf{z}^{1:T_{skill}} = \{\mathbf{z}^1, \ldots, \mathbf{z}^{T_{skill}}\}^{|B|}$ **do** ▷ Evaluate each sequence of latent representations in the batch
15: $\quad\quad \text{cost}_j = \text{evaluate}(p_\psi^{dec}(\mathbf{a}^{1:T}|\mathbf{z}^{1:T_{skill}}))$
16: $\quad\quad$ **if** cost $<$ cost$_{best}$ **then**
17: $\quad\quad\quad \text{cost}_{\text{best}} \leftarrow$ cost, $\mathbf{z}_{\text{best}} \leftarrow \mathbf{z}^{1:T_{skill}}$
$\quad$ **return** $\mathbf{z}_{\text{best}}$

F Task Details

F.1 Cost Function

The cost function used for trajectory optimisation is defined by Sinkhorn Divegence. Following the prior work [14], the *geomloss* library is used to define the cost function:

```
from geomloss import SamplesLoss
OT_LOSS = SamplesLoss(loss="sinkhorn", p=1, blur=0.0001)
```

F.2 Tasks

For single-hand task, such as *Folding* and *Wrap*, the action dimension is 26 (20 for actuators including finger joints and wrist, and 6 for the base). For in-hand manipulation tasks (*Flip*), a single hand with a fixed base is assumed, resulting in an action dimension of 20. In dual-hand environments, the action dimension is 52, allowing for a movable base for both hands. Since a VAE is trained to encode a single-arm action trajectory in the play dataset, an LDM generates a single-arm skill trajectory. Thus, to handle dual-hand tasks using *D-Cubed*, the LDM generates a trajectory for each arm. In the following, we describe the details of each task.

Folding: The initial position of the robot hand is above the dough, and the hand must fold the dough in four different directions: front, back, left, and right.

Wrap: The robot hand first picks up the plasticine ball and places it onto the dough shaped like a rope. Then, it pinches the side of the rope to wrap the ball inside it.

Flip: The robotic hand tosses the dough in the air to reshape and reposition it.

Bun: The two robotic hands deftly pinch and push the dough to form a bun-shaped object.

Rope: The right-hand grasps the rope on the right, lifts it, and places it above the left rope. Then, the left-hand bends the left rope.

Dumpling: To wrap a dumpling, the right hand first grasps the right side of the dough. While holding the dough with the right hand, the left hand lifts the left side of the dough. Finally, the two hands bring the two sides of the dough together and form it into a dumpling shape.