GENERATIVE TRAJECTORY MODELLING THROUGH MASKED DIFFUSION

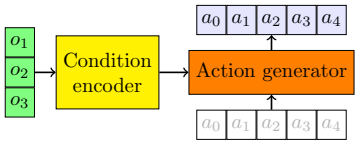
Anonymous authors
Paper under double-blind review

ABSTRACT

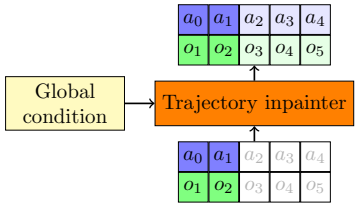
Behavioural cloning aims to imitate expert behaviour through supervision: a model is trained to predict the expert's actions from the observations it encountered. However, this framing discards the generative structure of trajectories, which arise from an interplay between the expert's policy and the environment's latent dynamics. Generative modelling is therefore a more natural fit. We motivate this perspective through four benefits: richer auxiliary learning signals, improved generalisation, temporal coherence, and robustness to unseen dynamics. Building on this, we propose a method for generative trajectory modelling that combines diffusion models with masked modelling, framing future trajectory prediction as a trajectory "inpainting" problem. We find this approach performs best under extremely high masking ratios of over 95%, reflecting the temporal redundancy in trajectories. Across a range of tasks, our method outperforms existing alternatives in efficiency (steps to task completion), accuracy (lower deviation from expert actions, both in-distribution and out-of-distribution), and generalisation to novel dynamics. To our knowledge, this is also the first demonstration that masked modelling can be effectively combined with denoising diffusion models.

1 INTRODUCTION

Behavioural cloning remains one of the prevailing frameworks for training control agents from offline expert demonstrations. It is essential in real-world settings where reward functions are challenging to engineer (Amodei et al., 2016) or where safety concerns and cost constraints make trial-and-error learning impractical, as in robotics and autonomous driving. Behavioural cloning trains a model to imitate an expert policy using offline demonstration datasets. A variety of approaches have been used to clone policies, including both supervised learning (Pomerleau, 1988; Ross & Bagnell, 2010; Ross et al., 2011; Shafiullah et al., 2022) and diffusion modelling (Pearce et al., 2023; Chi et al., 2024), but the basic paradigm remains the same: the model is trained to predict the next action or sequence of actions taken by the expert conditioned on one or more observations the expert was seeing (Fig. 1a).



(a)



(b)

Yet expert trajectories – sequences of actions and observations – contain significant structure that is lost when conditioning on observations or predicting actions alone. Observations and actions evolve smoothly across time and are correlated through the policy and the environment's transition structure. Thus standard approaches to behaviour cloning, despite their success, do not make optimal use of the training data, and behaviour cloning can be better served by modelling such larger-scale structure in trajectories. An agent that generatively models expert trajectories can capture such large-scale structure and use it to predict future continuations of histories for decision-making. As sequences are generated by an interplay between expert policy and transition dynamics, this implies learning an implicit model of the environment's dynamics as well. Such a model can actually help an agent solve tasks beyond the reach of traditionally cloned policies.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Consider the following two examples. In the first, an agent must catch a moving prey. Solving this requires understanding how the prey moves and then adapting accordingly. Since the prey's location is an observation, this amounts to modelling observational dynamics. In a second example, an agent must sail straight through a windy sea. But the wind changes from day to day, forcing the agent to change its actions if it wants to keep a straight course. While the observations do not change, the transition function – how actions map to motion – does. An agent that accounts for this shift in dynamics will be more successful than one that does not.

Masked modelling has been immensely successful in training language and vision models (Devlin et al., 2019; He et al., 2022). But it is particularly apposite for trajectory modelling. Many problems in control can be framed in terms of masked modelling. Future prediction corresponds to the case where the future is masked and the history is visible; inverse dynamics to the case where an action is masked; and goal conditioning to the case where the initial and final states are visible but the rest of the trajectory is masked. Indeed, prior work (Wu et al., 2023; Liu et al., 2022; Carroll et al., 2022) has noted that masked trajectory modelling unifies a wide range of inference problems commonly encountered in control. All these problems requires a strong understanding of both expert behaviour and the environment's transition structure. Masked trajectory modelling thus provides a richer learning signal. It forces the model to learn a more robust representation of both the expert policy and environment dynamics that it can apply to any downstream problem requiring this understanding. Considering this, we adapt masked modelling as our approach to behavioural cloning.

Masked modelling has traditionally been performed using either standard bidirectional transformers (Devlin et al., 2019) or masked autoencoders (He et al., 2022). Here, we leverage the greater expressivity of diffusion models (Ho et al., 2020; Song & Ermon, 2019; 2020; Dhariwal & Nichol, 2021a), as it has proven highly successful in modelling complex policies (Janner et al., 2022; Chi et al., 2024; Pearce et al., 2023). While standard diffusion models map pure Gaussian noise to clean data samples, our modification processes input where only masked segments are noised. Architecturally, this approach processes entire trajectories through a single stream, preserving their structural integrity rather than splitting them into histories and futures handled by distinct components.

We thus introduce a generative approach that models expert trajectories holistically. Our approach rests on three design choices: a unified architecture, training with masked-modelling, and a diffusion backbone (Fig. 1b). We hypothesise that this combination yields policies that more faithfully clone expert behaviour and are more robust at inference time. We refer to our models as *inpainting models*, in analogy to the inpainting problem in computer vision, where a model must fill in the missing parts of a partially masked image – in our case, we inpaint masked trajectories. Through a range of tasks, we empirically demonstrate the benefits of this approach. Our agent is generally more successful at solving tasks, solves them faster than other baselines, adapts better when it must exploit the dynamics of the environment, and is more robust to noise at inference time. Our approach also approximates ground truth policies more faithfully when known. Our results establish the importance of generatively modelling trajectories in a holistic manner. To our knowledge, this is also the first demonstration of the feasibility of combining masked modelling with diffusion models.

2 Related work

Behavioural cloning Traditionally, behaviour cloning trains networks to replicate expert behaviour through supervision. Initial approaches used multi-layer perceptrons (Pomerleau, 1988; Michie, 1993; Ross et al., 2011), but later work shifted to transformers Shafiullah et al. (2022). ALOHA (Zhao et al., 2023) introduced action chunking to improve temporal coherency. Subsequently, Pearce et al. (2023) and Chi et al. (2024) adapted diffusion models to more expressively clone policies. These works fell short of modelling whole, extended trajectories, something we consider in this work.

Masked modelling Masked modelling has proven immensely successful for pretraining bidirectional transformers in both language and vision. Introduced in BERT (Devlin et al., 2019), it set a new state-of-the-art in language modelling. In vision, it is the method of choice for pretraining vision transformers for image (Dosovitskiy et al., 2021; He et al., 2022; Oquab et al., 2023; Assran et al., 2023; Bardes et al., 2024) and video modelling (Tong et al., 2022b; Gupta et al., 2023; Assran et al., 2025; Kuhn et al., 2026). Masked-modelling has also been applied to trajectory modelling, using both masked autoencoders (He et al., 2022; Liu et al., 2022; Wu et al., 2023) and standard

bidirectional transformers (Carroll et al., 2022). However, most approaches to masked modelling rely on deterministic regression to predict masked tokens, making them ill-suited for modelling multi-modal distributions (Chi et al., 2024). Our work addresses this gap by marrying the expressivity of diffusion models with the rich learning signal provided by random, bidirectional mask modelling.

Diffusion modelling Diffusion models are the method of choice for generatively modelling high-dimensional, continuous data, particularly natural images (Song & Ermon, 2019; 2020; Ho et al., 2020; Dhariwal & Nichol, 2021b; Ramesh et al., 2022; Saharia et al., 2022) and control policies Janner et al. (2022); Ajay et al. (2023); Pearce et al. (2023); Chi et al. (2024). Diffusion policies (Chi et al., 2024) have successfully cloned multi-modal expert policies for simulated and real-world control. Masking has previously been considered in diffusion models for control (Chen et al., 2024; Zhu et al., 2025), where noise levels serve as partial masks. But these approaches did not consider random, bidirectional masking Devlin et al. (2019) which as well-established benefits as a pre-training objective. Diffusion forcing (Chen et al., 2024) is constrained to causal masking. And the masking considered in Unified World Models (Zhu et al., 2025) is not readily extendible to longer horizons with random masking positions. In contrast, we combine the training benefits of bidirectional masking with diffusion models.

3 Masked modelling of expert trajectories

3.1 Behavioural cloning as a generative modelling problem

Behavioural cloning is conventionally framed as a supervised learning problem. A model $\pi_\theta(a_t|o_t)$ is trained to replicate the expert's action a_t when seeing observation o_t through either a cross-entropy loss for discrete actions or an L2 loss for continuous ones. In some formulations (Chi et al., 2024), the prediction is extended to a horizon of H actions conditioned on a history of h past observations,

$$\pi_\theta(a_{t+H-1} \dots a_t | o_t \dots o_{t-h+1}). \tag{1}$$

In this paper, we take the perspective that behavioural cloning is more naturally framed as a generative modelling problem. The dataset of expert trajectories is viewed as a set of samples from a data generating distribution that combines the expert policy $\pi_E(a_t|o_t, a_{t-1}, \dots o_0)$ with the environment dynamics $P(o_{t+1}|a_t, o_t, a_{t-1}, \dots o_0)$.[1] The model π_θ is then trained to approximate the distribution

$$P(o_{t+H}, a_{t+H-1}, \dots, o_{t+1}, a_t | h_t) = \prod_{k=0}^{H-1} P(o_{t+k+1}|a_{t+k}, h_{t+k})\, \pi_E(a_{t+k}|h_{t+k}), \tag{2}$$

where $h_t = (o_0, \dots, a_{t-1}, o_t)$ is the history up to time t. In other words, π_θ is trained to predict the future extension of h_t. At inference time, we generate a prediction of the future extension, execute the first k actions in our prediction for some hyperparameter $k \leq H$, and then make another prediction with the new history. The predicted observations are never used. However, as we shall show empirically, they serve a purpose as a learning signal. The relation in (2) makes explicit the trajectory distribution's dependence on the environmental dynamics, particularly how dynamics ultimately affect control decisions made further into the future in the prediction horizon. Predicting observations forces π_θ to more directly model this dependence. The alternative is to marginalise out all future observations, yielding an action-only distribution,

$$\pi_\theta(a_{t+H-1}, \dots, a_t | h_t) = \int \mathrm{d}o_{t+H} \cdots \mathrm{d}o_{t+1}\ \pi_\theta(o_{t+H}, a_{t+H-1}, \dots, o_{t+1}, a_t | h_t), \tag{3}$$

which brings the prediction problem more aligned with the distribution in (1).[2] The dependence on environmental dynamics is now more implicit. However, from the learning perspective, the model never takes advantage of any learning signal that observations could have offered. Rather, the model is limited to capturing correlations between actions without a better understanding of the intermediate relationship between those actions. It is like predicting that a car will sometimes start and sometimes stop without realising it only stops in response to traffic signals.

[1] For generality, we work in the partially-observable setting, and for notational simplicity, we fold the latent state dynamics and observation model into the dynamics function $P(o_{t+1}|a_t\ o_t\ a_{t-1} \dots o_0)$.

[2] Recovering an observation-only conditioning along the lines of (1) would additionally require multiple non-trivial applications of Bayes rule combined with further marginalisation.

Future trajectory prediction can be regarded as a special case of masked modelling. Given a trajectory $\boldsymbol{\tau} = (o_0, a_1, o_1, \ldots, a_{t+H-1}, o_{t+H})$, we define a mask $M\colon \boldsymbol{\tau} \to \boldsymbol{\tau}_+$ that removes observations and actions from $\boldsymbol{\tau}$ to yield the visible portion of the trajectory $\boldsymbol{\tau}_+$. We then define an unmasking distribution $P_M(\boldsymbol{\tau}|\boldsymbol{\tau}_+)$ over the original $\boldsymbol{\tau}$ given only $\boldsymbol{\tau}_+$. The distribution in (2) is one special case of this. But in general, $\boldsymbol{\tau}_+$ will also be missing segments from the start and middle of $\boldsymbol{\tau}$ and inferring those segments will depend implicitly on the policy and transition dynamics through Bayes' rule. Inferring missing data from the middle of a time series is known as the smoothing problem in Bayesian inference and is generally solved by a forward-backward algorithm that uses transition dynamics to work both forward from the known initial segment and backward from the known tail Murphy (2022). The distributions implied by $P(\boldsymbol{\tau}|\boldsymbol{\tau}_+)$ generalises this to multiple missing segments from anywhere in $\boldsymbol{\tau}$. We train a single $\pi_\theta(\boldsymbol{\tau}|\boldsymbol{\tau}_+)$ to approximate $P(\boldsymbol{\tau}|\boldsymbol{\tau}_+)$ for all possible $\boldsymbol{\tau}_+$ in an amortised fashion, encouraging π_θ to acquire a more robust representation of the transition structure.

3.2 Diffusion modelling with randomly masked inputs

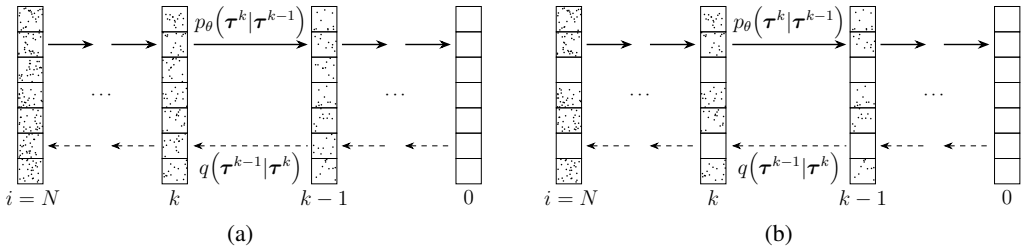


Figure 2: (a) In canonical diffusion models, a forward process (dashed arrows) incrementally corrupts a clean input ($i = 0$) along a Markov chain with Gaussian noise until reaching pure noise ($i = N$). A diffusion model $p_\theta\left(\boldsymbol{\tau}^k|\boldsymbol{\tau}^{k-1}\right)$ inverts this via the reverse process (solid arrows). (b) In the masked setting, the Markov chain only noises and denoises the masked squares.

We draw on denoising diffusion probabilistic models (DDPM) to approximate $P(\boldsymbol{\tau}|\boldsymbol{\tau}_+)$. DDPMs are amongst the most expressive generative models available for modelling continuous, high-dimensional data (Ho et al., 2020; Dhariwal & Nichol, 2021a). They learn to synthesise data by reversing a gradual noising process. In a fixed forward process (right-to-left dashed arrows in Fig. 2a), a clean input is incrementally corrupted with Gaussian noise following a fixed noise schedule. In the learnt reverse process, a model incrementally denoises the corrupted input until recovering the clean original (left-to-right solid arrows in Fig. 2a). DDPMs implement the reverse process by predicting the noise added at each forward step. Generation is performed by initialising samples with isotropic Gaussian noise and then iteratively denoising through the learnt reverse process until arriving at a clean sample. The resulting distribution should approximate the original data distribution.

Here, we present our modification of the diffusion process to perform trajectory inpainting. Let $\boldsymbol{M}$ denote a randomly sampled binary masking tensor. It has the same dimensionality as the trajectory tensor $\boldsymbol{\tau}$, is one whenever the corresponding element of $\boldsymbol{\tau}$ is masked out, and is zero otherwise. We use $\boldsymbol{M}$ to split $\boldsymbol{\tau}$ into its masked and visible components, $\boldsymbol{\tau}_- = \boldsymbol{M} \odot \boldsymbol{\tau}$ and $\boldsymbol{\tau}_+ = (\mathbf{1} - \boldsymbol{M}) \odot \boldsymbol{\tau}$ respectively, where $\odot$ denotes element-wise product and $\mathbf{1}$ is a tensor of ones. Note that $\boldsymbol{\tau} = \boldsymbol{\tau}_- + \boldsymbol{\tau}_+$.

The forward process is defined as a sequence of Gaussian transitions

$$q(\boldsymbol{\tau}^i|\boldsymbol{\tau}^{i-1}) := \mathcal{N}\left(\boldsymbol{\tau}^i;\ \sqrt{1-\beta_i}\,\boldsymbol{\tau}_-^{i-1} + \boldsymbol{\tau}_+,\ \beta_i \boldsymbol{M} \odot \mathbf{I}\right), \tag{4}$$

where $i = 1 \ldots N$ is the noising step, $\boldsymbol{\tau}^i$ and $\boldsymbol{\tau}_-^i$ are the partially noised versions of $\boldsymbol{\tau}$ and $\boldsymbol{\tau}_-$, and β_i is the noise level at step i. Note that Gaussian noise is only added to the masked components of $\boldsymbol{\tau}$ (right-to-left dashed arrows of Fig. 2b). The unmasked components remain uncorrupted and their variance in (4) is effectively zero; strictly speaking, their distribution should be understood as a delta distribution. Note also that when all of $\boldsymbol{\tau}$ is masked, $\boldsymbol{M} = \mathbf{1}$, we recover the original formulation of the forward process. A diffusion model is trained to approximate the reverse transition

$$p_\theta(\boldsymbol{\tau}^{i-1}|\boldsymbol{\tau}^i) := \mathcal{N}\left(\boldsymbol{\tau}^{i-1};\ \boldsymbol{\mu}_\theta(\boldsymbol{\tau}^i, i),\ \boldsymbol{\Sigma}^i\right), \tag{5}$$

where the mean $\boldsymbol{\mu}_\theta(\boldsymbol{\tau}^i, i)$ is a neural network model and the variance $\boldsymbol{\Sigma}^i$ can also be learnt but is, in practice, often set to a fixed noise schedule. Both the forward and reverse transitions are Markovian,

so the entire chain of forward and reverse transitions constitute Markov chains (dashed and solid arrows respectively in Fig. 2b). At generation time, we initialise $\boldsymbol{\tau}^N$ by sampling its unmasked components from an isotropic Gaussian, $p(\boldsymbol{\tau}^N|\boldsymbol{\tau}_+) = \mathcal{N}\left(\boldsymbol{\tau}^N;\ \boldsymbol{\tau}_+,\ \boldsymbol{M} \odot \boldsymbol{I}\right)$, and then pass this sample through the Markov chain defined by (5) until arriving at a sample from the clean distribution,

$$p_\theta(\boldsymbol{\tau}^0|\boldsymbol{\tau}_+) = \int p(\boldsymbol{\tau}^N|\boldsymbol{\tau}_+) \prod_i^N p_\theta(\boldsymbol{\tau}^{i-1}|\boldsymbol{\tau}^i) d\boldsymbol{\tau}^{1:N}. \tag{6}$$

This approximates the original data distribution $p(\boldsymbol{\tau}^0|\boldsymbol{\tau}_+)$. The function $\boldsymbol{\mu}_\theta(\boldsymbol{\tau}^i, i)$ can be learnt directly by optimising a variational lower bound on the negative log likelihood $\mathbb{E}[-\log p_\theta(\boldsymbol{\tau}^0|\boldsymbol{\tau}_+)]$. However, it is more common to train a network $\boldsymbol{\epsilon}_\theta$ to directly predict the corrupting noise $\boldsymbol{\epsilon}^i$ as this yields better results in practice,

$$\mathcal{L}(\theta) := \mathbb{E}_{i,\boldsymbol{\tau}^0,\boldsymbol{\epsilon},\boldsymbol{M}}\left[\|\boldsymbol{M} \odot \boldsymbol{\epsilon}^i - \boldsymbol{\epsilon}_\theta(\boldsymbol{\tau}^i, i)\|^2\right]. \tag{7}$$

The product on $\boldsymbol{M}$ ensures that the target noise for the unmasked portion of $\boldsymbol{\tau}$ is zero. Ho et al. (2020) show that reverse transitions of the diffusion process are equivalent to Langevin sampling steps where one can simply subtract $\boldsymbol{\epsilon}_\theta(\boldsymbol{\tau}^i, i)$ from $\boldsymbol{\tau}^i$ to obtain $\boldsymbol{\tau}^{i-1}$.

We implement our diffusion model as a diffusion transformer (Peebles & Xie, 2023), based on the diffusion policy transformers of Chi et al. (2024). We refer to the tensors that a diffusion process acts on as the *canvas*. In Chi et al. (2024), histories are conditioned *exogenous* to the canvas (Fig. 1a), being processed by a separate encoder that then guides the diffusion process through cross-attention. In contrast, our conditioning is *endogenous* to the canvas (Fig. 1b) and does not require a separate encoder. With endogenous conditioning, the history-future partitioning of the canvas becomes an inference-time decision. We still allow for exogenous conditioning but only of global context that is not part of the trajectory, such as task specifications through static goals or instructions.

We refer the reader to Supplement A for additional details on the models.

4 Empirical Evaluation

We evaluated our algorithm on a variety of tasks to test its overall performance, robustness, and ability to generalise in different ways. Our inpainting agent performed better overall, solved tasks in fewer steps, better generalised across latent dynamics, and was more robust to inference-time perturbations.

4.1 Tasks

Push-T Introduced by Chi et al. (2024), push-T tests an algorithm's ability to clone multi-modal policies. The agent pushes a T-block onto a green target region using a point effector (Fig. 3, left). Initial effector and block positions are randomised. Success is measured by coverage with the target area. Training data consists of 200 human teleoperated demonstrations. Behavioural cloning is notoriously brittle to inference-time distributional shifts, so to assess robustness and generalisation, we introduce several variants of the task:

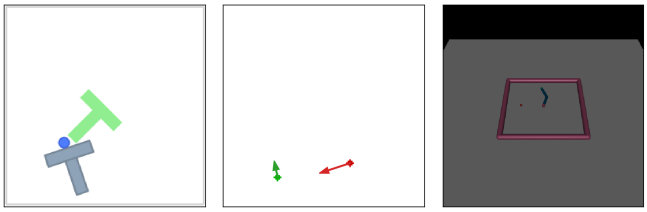


Figure 3: Push-T (left), Hunter-prey (middle), and Reacher (right).

- **Vanilla**: The original task as introduced by Chi et al. (2024).
- **Noisy observations**: Gaussian pixel noise corrupts observations, akin to sensor noise or failure in robotics settings. The environment generates observation o, but the agent sees $o + \xi$, with $\xi \sim \mathcal{N}(0, \sigma^2 = 25)$.
- **Noisy actions**: Gaussian control noise corrupts agent's actions, akin to actuator noise. The agent attempts action a, but $a + \xi$ gets executed instead, where $\xi \sim \mathcal{N}(0, \sigma^2 = 25)$.

- **Random dynamics**: This setting tests the agent's ability to infer and generalise across latent environment dynamics. Formally, we vary how an action a maps to the next observation o' under the transition function $P(o'|o, a)$. Specifically, we randomise the effector's responsiveness to actions, through its gain and damping factors, across episodes. The agent must infer this responsiveness from its recent actions and their observed effects, and adjust its behaviour accordingly.

Hunter-Prey We introduce hunter-prey (Fig. 3, middle) as another task for assessing agent performance in dynamic environments. Hunter-prey is a tracking task in which the agent (hunter) must intercept a moving target (prey). The prey follows elliptical orbits with random deviations, and orbits are randomised across episodes. This varies how observations evolve over time. Formally, this varies how the current observation o maps to the next observation o' under the transition function $P(o'|o, a)$. At evaluation, the prey follows orbits never before seen during training. To intercept the prey, the agent must therefore anticipate its motion by learning its dynamics and generalising them to unseen orbits. We refer the reader to Supplement D for the expert policy.

Reacher Reacher (Fig. 3, right) is a classic, continuous control environment from the gymnasium library (Towers et al., 2026). The agent controls a two-joint arm from a random starting configuration towards a target configuration. We modify reacher to assess how an agent's cloned policy extrapolates beyond the training region. Specifically, we define a closed-form expert policy through inverse kinematics to specify a mapping from the current and target configurations to an action, $\pi : (x_{curr},\ x_{target}) \rightarrow a$. During training, the agent sees trajectories originating from randomly sampled starting configurations but terminating within designated training quadrants. It must recover the closed-form policy from these demonstrations and then extrapolate it to the held-out quadrants. While this might be seen as a hard, out-of-distribution generalisation task, it is better understood as a test of function-approximation quality. We refer the reader to Supplement F for the expert policy.

4.2 Baselines

We compare against diffusion policy (Chi et al., 2024), arguably the state-of-the-art in behavioural cloning and the approach most similar to ours. In building the inpainting model, we endeavoured to minimise any changes from the diffusion policy along dimensions irrelevant to our contribution. As such, the codebase, architectures and hyperparameters are significantly unified between the two algorithms. To better understand how our design choices contribute to the inpainting agent's performance, we also ran these variants of the diffusion policy:

- **obs** $\rightarrow$ **action**: The original version of the diffusion policy. The agent conditions on observations and predicts actions.
- **obs** $\rightarrow$ **obs+action**: The diffusion policy conditions on observations but predicts observations and actions. This tests the importance of observation prediction to performance.
- **obs+action** $\rightarrow$ **action**: The diffusion policy conditions on observations and actions but predicts actions only. This tests the importance of action conditioning to performance.
- **obs+action** $\rightarrow$ **obs+action (all-to-all)**: The diffusion policy conditions on observations and actions and predicts observations and actions. This fully matches the inpainting agent in conditioining information and prediction targets. Only endogenous conditioning and masked-modelling are ablated.

Evaluation protocol We report results averaged across 3 seeds and N_{task} episodes. N_{task} is 50 for push-T (following Chi et al. (2024)) and 100 for hunter-prey and reacher. We follow Chi et al. (2024) and report either performance of the best test checkpoint or average across the last 10 checkpoint.

4.3 Masking policies

We first identified the optimal masking policy in push-T and kept it fixed across all subsequent tasks. It is known from vision and language (Devlin et al., 2019; He et al., 2022; Tong et al., 2022a) that the optimal masking ratio – the proportion of tokens to randomly mask – increases with the degree of input redundancy, and smoothly evolving trajectories carry high redundancy. Here, we obtained

best performance at $p = 0.88$ and 0.98 (Fig. 4). By comparison, a fixed masking pattern that matches the inference-time mask, where the history tokens are visible and the rest are masked, outperforms random masking at the same ratio ($p = 0.75$) but still falls short of random masking at higher ratios. This result highlights the value of random masking at high ratios, which extracts more learning signal from the same dataset than training on the inference-time pattern alone.

While $p = 0.88$ had a marginally higher score, the performance of $p = 0.98$ converged more quickly, as measured by the area under the curve (AUC), and solved the task in fewer steps (episode length). We therefore used $p = 0.98$ in the rest of our experiments.

	Score	AUC	Episode length
$p = 0.25$	0.16/0.12	0.12	300/300
$p = 0.5$	0.24/0.14	0.15	300/300
$p = 0.75$	0.91/0.87	0.63	272/284
$p = 0.88$	**0.98/0.94**	0.82	202/223
$p = 0.98$	0.97/0.91	**0.84**	**195/213**
$p = 1.0$	0.38/0.22	0.25	299/300
inference mask ($p = 0.75$ fixed)	0.94/0.91	0.81	243/255

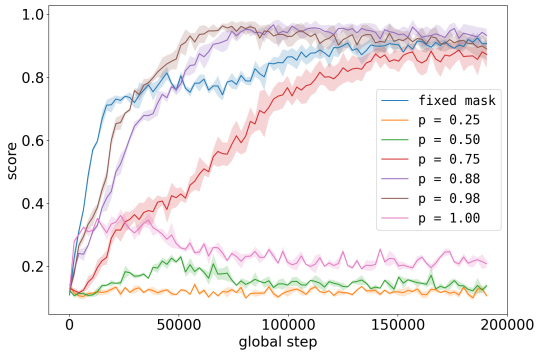


Figure 4: Push-T masking ratio sweep for inpainting ($d = 512$). **Left:** score, AUC, and episode length for each training masking ratio p, and for the inference-mask variant with $p = 0.75$ fixed. Paired entries show (max performance) / (average of last 10 checkpoints); best values in bold. **Right:** score for all p values considered.

4.4 PUSH-T

Success rate and solution efficiency in vanilla push-T To better understand the behaviour of our inpainting agents, we compared its performance against baselines on the original push-T benchmark. Our inpainting agent achieves a success rate of 0.97, outperforming the diffusion policy's rate of 0.94 (Table 1). The difference is even more noticeable when comparing episode length, a metric of efficiency. The inpainting agent needs 13% fewer steps to solve the task (195 vs 229 steps). We also compared the inpainting agent against a variant of diffusion policy that both conditioned on and predicted actions and observations (diffusion policy, all-to-all), just like the inpainting model. While this helped performance, it still could not match inpainting performance. These results show that inpainting agents make more efficient use of the same training data. Conditioning on actions and predicting observations, while important, is not sufficient to match inpainting performance. Both the masked modelling objective and endogenous conditioning are also essential.

Table 1: Push-T results for each agent. For each variant, the first row shows maximum checkpoint success rate and the second shows episode length. For perturbed variants, the fraction of the vanilla success rate is given in parentheses. All models use $d = 512$.

	Diffusion Policy	Diffusion Policy (all-to-all)	Inpainting (all-to-all)
Vanilla	0.94 229	0.94 215	**0.97** **195**
Rand. dyn.	0.81 (86%) 267	0.89 (94%) 242	**0.93** (95%) **240**
Noisy obs	0.84 (89%) 284	0.85 (91%) 277	**0.90** (92%) **273**
Noisy action	0.89 (95%) 273	0.79 (84%) 290	**0.93** (96%) **263**

Robustness to inference-time noise Behavioural cloning is notoriously brittle to distribution shifts at inference time (Ross & Bagnell, 2010). We hypothesised that a generative model of trajectories that more faithfully approximates the latent expert distribution could better mitigate this by supporting superior generalisation. To that end, we assessed the robustness of cloned policies to both control and observation noise at inference time. In real-world settings, observation noise is analogous to sensor

failure or reduced visibility due to poor weather in autonomous driving. Control noise may reflect actuator failure. In these evaluations, noise is introduced at inference time only; agents continue to be trained on the vanilla, push-T human data. This makes these assessments entirely out-of-distribution.

Performance declined across all agents, but the inpainting agent was the least affected (Table 1). Compared to the diffusion policy, its absolute success rate was higher (0.90 vs 0.84 for noisy observation and 0.93 vs 0.89 for noisy action), and it continued solving the tasks in fewer steps (273 vs 284 for noisy observation and 263 vs 273 for noisy action). We also measured the performance drop relative to the agents' own performance in the vanilla environment. The relative decline was greater for diffusion policy (92% vs 89% for noisy observation and 96% vs 95% for noisy control).

Unlike the diffusion policy, the inpainting agent conditions on both observations and actions, which may mitigate observation noise by providing redundant information. To account for this, we also evaluated an all-to-all variant of diffusion policy on these tasks. Relative to the standard diffusion policy, performance marginally improved when observations were noisy (0.85 vs 0.84 success rate and 277 vs 284 episode length) but declined when control was noisy (0.79 vs 0.89 success rate and 290 vs 273 episode length). When observations are unreliable, a clean action signal can provide an additional source of information, but when actions are noisy, they can mislead the agent. Crucially, since the all-to-all agent and the inpainting agent condition on the same information, this cannot account for the inpainting agent greater robustness (0.90 vs 0.85 success rate in noisy observation and 0.93 vs 0.79 success rate in noisy action). This suggests that the inpainting agent learns a better trajectory model, which in turn supports generalisation to weakly out-of-distribution settings.

Latent dynamics inference Agents in the physical world often face scenarios in which environmental dynamics vary. A boat trying to sail in a straight line must adjust its actions to wind conditions. Here, we randomised the point effector responsiveness across episodes. Like our sailing example, the agent could follow the same path in physical space if it desired, but it would have to perform a different sequence of actions to do so. This introduces an element of dynamics aliasing, as an agent cannot infer the mapping from actions a to next observations o' by tracking observations alone.

As we see in Table 1, the inpainting agent outperforms the diffusion policy agent by a significant margin on this task as well (0.93 vs 0.81 success rate). Yet in spite of dynamics aliasing, the diffusion policy's performance does not collapse. The randomness in dynamics is inherently unpredictable for this agent and effectively acts as control noise. When given action information, its performance does improve but not to the level of the inpainting agent (0.89 vs 0.93 success rate). Again, this shows that endogenous conditioning and masked modelling are important.

Table 2: Performance in hunter-prey. We report maximum performance / mean across the last 10 checkpoints.

	Hunter-Prey (d=256)	
	Catch rate	Episode length
Diffusion Policy	0.39/0.29	394/419
Diffusion Policy (all-to-act)	0.34/0.18	400/445
Diffusion Policy (obs-to-all)	**1.00/1.00**	**40/40**
Diffusion Policy (all-to-all)	**1.00/1.00**	116/126
Inpainting (all-to-all)	**1.00**/0.99	53/55

4.5 HUNTER-PREY

Consistent with our hypothesis that predicting observations helps an agent model the prey's dynamics, the inpainting agent solves hunter-prey perfectly while diffusion policy struggles (Table 2). To test this further, we augmented diffusion policy by including either actions in its history or observations in its predictions. The catch rates consistently show that observation prediction is needed to solve the task well. Episode lengths, meanwhile, show that action conditioning degrades diffusion policy performance. Accordingly, the obs-to-all variant, which predicts observations without action conditioning, performs best among the diffusion policy variants. Crucially, the inpainting agent completes episodes substantially faster than the all-to-all variant, despite sharing its conditioning and prediction targets. This again indicates that the inpainting agent's advantage is not due solely to all-to-all conditioning and prediction. Endogenous conditioning and masked modelling remain important, likely because they improve the temporal coherence of the generated trajectories.

4.6 REACHER

Reacher is a goal-conditioning task. A goal specifies what the agent must achieve. As such, the inpainting agent conditions on it exogenously through cross-attention (Fig. 1b). Diffusion policy already conditions exogenously, so we simply concatenated the goal to observation vectors. We compared this against separate tokenisation but found this worked better.

Table 3: Reacher generalization when training on goals in one quadrant and testing on goals in held-out quadrants. Both models use $d = 768$. Mean error is over 100 goals per quadrant.

	Success rate ↑		Mean error ↓	
	Train	Test	Train	Test
Diffusion Policy	**1.00**	0.29	0.0053	0.0430
Inpainting	**1.00**	**0.53**	**0.0020**	**0.0205**

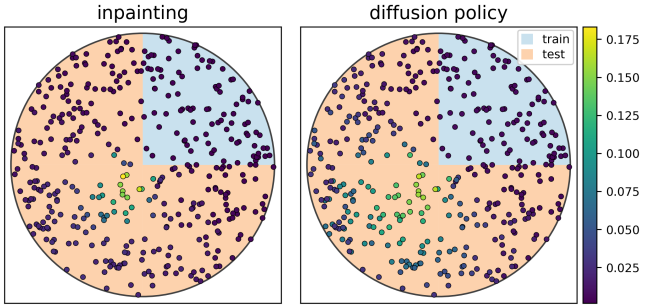


Figure 5: Mean-squared error of each policy with respect to the ground truth, trained on goals in cyan region. **Left:** Inpainting. **Right:** Diffusion policy. Each dot shows an evaluation goal at the corresponding location. Colours show average error across 400 random start configurations, three seeds, the top three checkpoints per seed, and five generation samples per start-goal pair.

Across all train-test splits, the inpainting agent outperformed the diffusion policy, particularly in the held-out quadrants (Table 3). To determine whether this was because the inpainting agent more closely approximates the ground truth policy, we compared each policy's actions against those of the ground truth policy. Specifically, we sampled 400 start configurations and 100 goals in each quadrant. For each configuration-goal pair, we computed the difference between the expert action and the action outputted by each policy at its three best test-performance checkpoints. The results in Fig. 5 reveal a noticeably smaller error for the inpainting agent. This suggests that our approach learns a better approximation of the ground truth policy, which may explain its ability to extrapolate better beyond the training regime.

5 DISCUSSION

Our work suggests several important connections to other areas of control as well as important directions for future work.

Trajectory model inherently contain world models. By conditioning on the last action, they predict the next observation. This implicit model was essential to our inpainting agent's performance in dynamical tasks like hunter-prey, and this connection suggests behavioural cloning is not entirely divorced from world modelling, as noted by Zhu et al. (2025). Cloned policies can draw on implicit world models to adapt. Nevertheless, behaviourally cloned world models would be limited in scope. As the training data is narrow and restricted to expert trajectories, much of the world is unexplored

Masked trajectory modelling unifies many problems in control, including goal conditioning and inverse dynamics (Liu et al., 2022; Wu et al., 2023; Carroll et al., 2022). Our approach is naturally suited to these problems, and comparing it against prior approaches is an exciting direction to explore.

6 CONCLUSION

In this work, we introduce a novel approach to generative behavioural cloning based on masked trajectory modelling through diffusion. Masking forces the model to acquire a more robust understanding of the trajectory structure and transition dynamics. Our approach shows superior performance on a variety of tasks, better generalisation and stronger robustness. These results establish trajectory inpainting through masked diffusion as an important new approach to behavioural cloning.

AI USE STATEMENT

In this work, we used generative AI tools in the form of coding agents for software development tasks. These encompassed activities such as model development, data pipelines, generation of plots and tables, and training environments implementation. Regarding this later point, we used coding agents to create "Hunter-Prey" from scratch, starting from design specifications of our own and iterating on ideas about how to introduce random dynamics in the expert policy. We also used AI coding agents for implementing the modified variations of both "Push-T" and "Reacher". As a way of providing an accurate representation of the implementation details in the paper, the supplementary materials sections that detail environment dynamics were mostly written with AI-assistance, where we extracted the relevant code files, and prompted models with those files and the task of generating a latex-formatted section that mathematically described the environment dynamics. Both sections were later manually checked to ensure correctness and alignment in style.

Additionally, we used generative AI tools in their browser interface for literature review and discovery, where we checked all AI-proposed references by searching for the original source material and reviewing it ourselves to verify their quality, relevance and scope, much as the workflow used when reviewing literature using a regular search engine. We discarded AI suggestions whenever we considered them inaccurate in their claims. We have not used generative AI tools for the writing the core content of this manuscript beyond the sections disclaimed in this statement, with minor exceptions limited to refining the phrasing of some sentences. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies imitation learning in simulated control environments (Push-T, Hunter-Prey and Reacher). It involves no human subjects, personal data or real-world deployment. The Push-T demonstrations come from the publicly released dataset of Chi et al. (2024), and the Hunter-Prey and Reacher demonstrations are generated by scripted experts. We are not aware of ethical concerns specific to this work beyond the general considerations that apply to learning-based robot control.

REPRODUCIBILITY STATEMENT

We provide anonymised source code as supplementary material, containing the training entry point, the environment implementations, and a configuration file for every experiment reported in this paper: the three conditioning strategies on Push-T and its evaluation-time variants, the masking-ratio sweep, the eight-arm Hunter-Prey ablation, and the Reacher quadrant-coverage experiments. Architecture and optimisation settings for every environment are listed in Table 6; the environment variants and their sampling distributions are specified in Appendix. Datasets are not redistributed but are reproducible: the Hunter-Prey and Reacher demonstrations are generated by the scripted experts included in the code, and Push-T uses the publicly released demonstrations of Chi et al. (2024), with the included relabelling script producing the action-relabelled copy used by the random-dynamics variant. Every reported number is the mean over three training seeds, with the per-seed statistic and the number of evaluation episodes stated in each table caption. Evaluation metrics are recorded during training and stored in the checkpoints, and the code includes the scripts that extract them and regenerate the tables and figures.

REFERENCES

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding

predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, June 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=QaCCuDfBk2.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016. URL https://arxiv.org/abs/1606.01540.

Micah Carroll, Orr Paradise, Jessy Lin, Raluca Georgescu, Mingfei Sun, David Bignell, Stephanie Milani, Katja Hofmann, Matthew Hausknecht, Anca Dragan, and Sam Devlin. Uni[MASK]: Unified inference in sequential decision problems. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=GisHNaleWiA.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion, 2024. URL https://arxiv.org/abs/2407.01392.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=a7APmM4B9d.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, pp. 02783649241273668, 2024. doi: 10.1177/02783649241273668. URL https://doi.org/10.1177/02783649241273668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021a. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021b. URL https://openreview.net/forum?id=AAWuCvzaVt.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale.

In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Daniel T. Gillespie. Exact numerical simulation of the Ornstein-Uhlenbeck process and its integral. *Physical Review E*, 54(2):2084–2091, 1996. doi: 10.1103/PhysRevE.54.2084.

Agrim Gupta, Jiajun Wu, Jia Deng, and Fei-Fei Li. Siamese masked autoencoders. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 40676–40693. Curran Associates, Inc., 2023. doi: 10.52202/075280-1771. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7ffb9f1b57628932518505b532301603-Paper-Conference.pdf.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16000–16009, June 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 1273–1286. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/099fe6b0b444c23836c4a5d07346082b-Paper.pdf.

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

Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, Berlin, Heidelberg, 1992. doi: 10.1007/978-3-662-12616-5.

Lukas Kuhn, Lucas Maes, Giuseppe Serra, Quentin Le Lidec, Yann LeCun, Randall Balestriero, and Florian Buettner. Levjepa: Efficient scalable video pretraining without the heuristics, 2026. URL https://arxiv.org/abs/2608.27395.

Fangchen Liu, Hao Liu, Aditya Grover, and Pieter Abbeel. Masked autoencoding for scalable and generalizable decision making. In *Advances in Neural Information Processing Systems*, 2022. URL http://arxiv.org/abs/2211.12740.

Donald Michie. Knowledge, learning and machine intelligence. In Leon S. Sterling (ed.), *Intelligent Systems*, pp. 1–19. Springer, Boston, MA, 1993. doi: 10.1007/978-1-4615-2836-4_1. URL https://doi.org/10.1007/978-1-4615-2836-4_1.

Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL http://probml.github.io/book1.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Pv1GPQzRrC8.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Dean A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D. Touretzky (ed.), *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988. URL https://proceedings.neurips.cc/paper_files/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *CoRR*, abs/2204.06125, 2022. URL https://arxiv.org/abs/2204.06125.

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/ross10a.html.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *CoRR*, abs/2205.11487, 2022. URL https://arxiv.org/abs/2205.11487.

Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=agTr-vRQsa.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 12438–12448. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92c3b916311a5517d9290576e3ea37ad-Paper.pdf.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 10078–10093. Curran Associates, Inc., 2022a. doi: 10.52202/068431-0732. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/416f9cb3276121c42eebb86352a4354a-Paper-Conference.pdf.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training, 2022b. URL https://arxiv.org/abs/2203.12602.

Mark Towers, Ariel Kwiatkowski, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Kallinteris Andreas, Markus Krimmel, Arjun KG, Rodrigo De Lazcano Perez-Vicente, J K Terry, Andrea Pierré, Sander V Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=qPMLvJxtPK.

G. E. Uhlenbeck and L. S. Ornstein. On the theory of the brownian motion. *Physical Review*, 36(5): 823–841, 1930. doi: 10.1103/PhysRev.36.823.

Philipp Wu, Arjun Majumdar, Kevin Stone, Yixin Lin, Igor Mordatch, Pieter Abbeel, and Aravind Rajeswaran. Masked trajectory models for prediction, representation, and control. In *International Conference on Machine Learning*, 2023.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets, 2025. URL https://arxiv.org/abs/2504.02792.

A Implementational details

Trajectories are represented in our canvas as a sequence of tokens consisting, at each time step, of an observation vector concatenated with the action vector taken in response to that observation,

$$\boldsymbol{\tau} = \left(\begin{bmatrix} \boldsymbol{a}_0 \\ \boldsymbol{o}_0 \end{bmatrix}, \cdots \begin{bmatrix} \boldsymbol{a}_t \\ \boldsymbol{o}_t \end{bmatrix}, \cdots \begin{bmatrix} \boldsymbol{a}_T \\ \boldsymbol{o}_T \end{bmatrix} \right). \tag{8}$$

There are two reasons for this. We expect actions and observations to evolve continuously across time, therefore concatenating them together gives us a vector that should also evolve continuously, whereas interleaving observations and actions, as is common in other work (Chen et al., 2021; Janner et al., 2021), would break this continuity. We hypothesised that continuously evolving representations would more readily reveal the larger scale structure of trajectories as well as be easier for a diffusion process to model. The other reason is that interleaving observations and actions would double the context window length and thereby quadruple the computational complexity of attention. This is the reason prior work on sequence modelling likewise chose to use a concatenated tokenisation (Janner et al., 2022; Wu et al., 2023). Masking is performed at the level of individual actions and observations instead of the level of tokens. To avoid one modality dominating over the other in the reconstruction loss, we computed the losses (7) for actions and observations separately and then normalised each by their respective dimensionalities before summing to obtain the overall loss.

Both baselines and our method condition on a history of length T_h and predict a horizon of length T_p. For our inpainting models, this gives a total canvas length of $T = T_h + T_p$. Importantly, however, the division of T into T_h and T_p only needs to be done at inference time for our approach, whereas it is a training time decision for the baselines as it determines the architecture. To improve temporal coherency and reduce replanning frequency, the diffusion policy of Chi et al. (2024) also commits to taking the first T_a of T_p actions before generating another T_p actions. We take the same approach here.

B Supplementary results

We present additional empirical results with the full set of ablations in Hunter-Prey in Table 4, which includes additional results on different mapping strategies implemented for the inpainting model. We also extend generalisation results for Reacher in Table 5, with a broader selection of train-test splits quadrants.

C Agent hyper parameters

Table 6 references the hyper parameters used for the inpainting model in each environment that we tested.

Table 4: Max test mean score and last 10 checkpoints mean score for conditioning format (exogenous or endogenous) and map type (observations to actions, observations to all, all to actions, and all to all).

	Hunter-Prey (d=256)	
	Catch rate	Episode length
Diffusion Policy	0.39/0.29	394/419
Diffusion Policy (all-to-act)	0.34/0.18	400/445
Diffusion Policy (obs-to-all)	**1.00/1.00**	**40/40**
Diffusion Policy (all-to-all)	**1.00/1.00**	116/126
Inpainting (obs-to-act)	0.39/0.31	391/416
Inpainting (all-to-act)	0.42/0.34	385/405
Inpainting (obs-to-all)	**1.00/1.00**	44/47
Inpainting (all-to-all)	**1.00**/0.99	53/55

Table 5: Reacher generalization across three goal splits. Models are trained on goals from the listed quadrants and tested on the remaining ones. d is the model dimension. Mean error is computed over 100 goals per quadrant.

	Success rate ↑		Mean error ↓	
	Train	Test	Train	Test
Train q1, test q2+q3+q4				
Diffusion Policy ($d = 768$)	**1.00**	0.29	0.0053	0.0430
Inpainting ($d = 768$)	**1.00**	**0.53**	**0.0020**	**0.0205**
Train q1+q3, test q2+q4				
Diffusion Policy ($d = 768$)	0.99	0.98	0.0051	0.0054
Inpainting ($d = 768$)	**1.00**	**0.99**	**0.0028**	**0.0030**
Train q1+q2+q3, test q4				
Diffusion Policy ($d = 1024$)	0.99	**0.97**	0.0038	0.0039
Inpainting ($d = 1024$)	**1.00**	**0.97**	**0.0024**	**0.0026**

D HUNTER-PREY ENVIRONMENT.

The task is a pursuit game on an $8 \times 8\,\mathrm{m}$ torus, with each coordinate in $[-4, 4]$ and wrapped independently. The time step is $\Delta t = 0.05\,\mathrm{s}$. Let $\mathrm{wrap}(z) = ((z + 4) \bmod 8) - 4$, applied to each component. The shortest displacement from hunter to prey is

$$d_t = \mathrm{wrap}(x_{p,t} - x_{h,t})\,, \tag{9}$$

and an episode ends when $\|d_t\| < 0.2\,\mathrm{m}$ or when the step limit is reached. The reward is 1 on capture and $-0.01\,\|d_t\|$ otherwise. The observation is (x_h, v_h, x_p, v_p), with wrapped positions and no relative vector.

Hunter The action is a 2D acceleration $a_t = (a_x, a_y)$ with $|a_x|, |a_y| \le 8\,\mathrm{m/s}^2$. The dynamics are

$$v_{h,t+1} = \mathrm{clip}(0.9\, v_{h,t} + a_t \Delta t,\ -4,\ 4)\,, \tag{10}$$

$$x_{h,t+1} = \mathrm{wrap}(x_{h,t} + v_{h,t+1} \Delta t)\,, \tag{11}$$

where the clip is applied to each velocity component (in m/s), so velocity persists across steps. The hunter starts at rest at a uniformly random position.

Prey For each level seed, an ellipse with focus $f = (f_x, f_y)$ is sampled,

$$r(\varphi) = \frac{l}{1 + e\cos(\varphi - \theta_0)}, \qquad c(\varphi) = f + r(\varphi)\,(\cos\varphi, \sin\varphi), \tag{12}$$

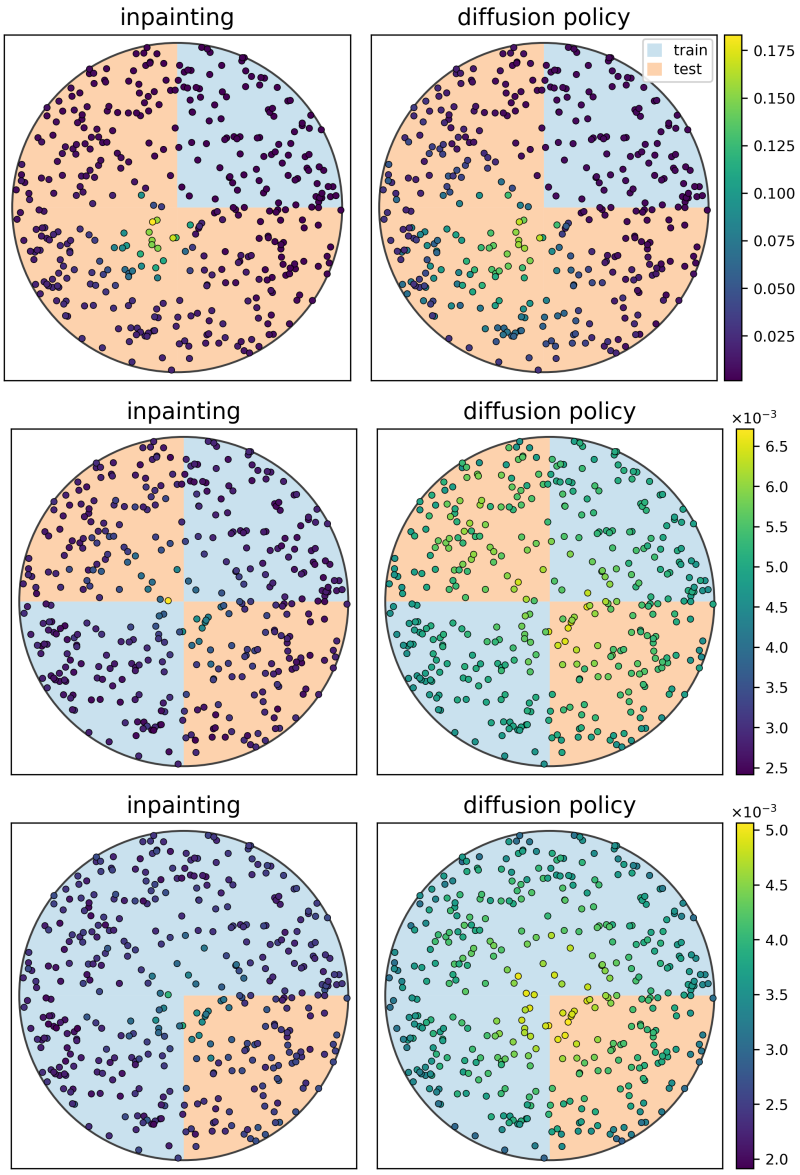


Figure 6: Mean-squared error of each policy with respect to the ground truth. Left column: inpainting. Right column: diffusion policy. Top row: trained on q1 goals. Middle row: trained on q1+q3 goals. Bottom row: trained on q1+q2+q3 goals. Each dot shows a goal at the corresponding location. Colours show average error across 400 random configurations, five action samples per configuration, three seeds, the top three checkpoints per seed.

with parameters

$$\begin{aligned} e &\sim U(0, 0.6), & l_{\text{frac}} &\sim U(0.3, 0.8), & l &= 4\, l_{\text{frac}}\,(1-e), \\ \theta_0 &\sim U(0, 2\pi), & \varphi_0 &\sim U(0, 2\pi), & f_x, f_y &\sim U(-4, 4), \\ s &\sim U\{-1, +1\}, & \rho &\sim U(0.30, 0.42), & \|v_p\| &= 4\rho. \end{aligned} \tag{13}$$

Here φ is the polar angle about the focus f, e is the eccentricity, θ_0 is the direction of the point of the ellipse closest to the focus, l is the semi-latus rectum, φ_0 is the initial angle, and s is the direction of rotation. The prey's constant speed $\|v_p\|$ (1.2 to 1.68 m/s) is a fraction ρ of the hunter's terminal speed of 4 m/s. Starting from φ_0, the angle φ_t advances in direction s by the amount that covers an arc length $\|v_p\|\Delta t$ along c. The reference heading $\psi_{\text{ref}}(\varphi_t)$ is the angle of the tangent to c at φ_t, taken in the direction of motion. The curve c therefore only supplies this heading. The prey starts on

Table 6: Architecture and training hyperparameters of the masked modelling agent. For Reacher, values separated by a slash correspond to the $d = 768$ and $d = 1024$ models respectively.

	Push-T	Hunter-Prey	Reacher
Architecture			
History length T_h	2	4	4
Prediction horizon T_p	14	12	12
Joint canvas $T_h + T_p$	16	16	16
Action horizon T_a	8	8	4
Model layers	8	8	8
Transformer heads	4	4	4
Model dimension d	512	256	768 / 1024
Attention dropout probability	0.3	0.1	0.1
Optimisation			
Batch size	256	256	256
Optimiser	AdamW	AdamW	AdamW
Optimiser (β_1, β_2)	(0.9, 0.95)	(0.9, 0.95)	(0.9, 0.95)
Weight decay	10^{-3}	10^{-3}	10^{-3}
Learning rate	10^{-4}	10^{-4}	10^{-4}
Learning rate scheduler	cosine	cosine	cosine
Learning rate warmup steps	1000	1000	1000
Training epochs	4500	500	1600 / 1000
Diffusion			
Denoising steps	100	100	100
Noise schedule	squared cosine	squared cosine	squared cosine
Starting noise β_0	10^{-4}	10^{-4}	10^{-4}
Final noise β_{99}	0.02	0.02	0.02

it, $x_{p,0} = c(\varphi_0)$, and then moves along $\psi_t = \psi_{\text{ref}}(\varphi_t) + \delta_t$, where δ_t follows an Ornstein-Uhlenbeck process (Uhlenbeck & Ornstein, 1930), discretized with an Euler-Maruyama step (Kloeden & Platen, 1992):

$$\delta_{t+1} = (1 - \kappa\,\Delta t)\,\delta_t + \sigma\sqrt{\Delta t}\;\varepsilon_t, \qquad \varepsilon_t \sim \mathcal{N}(0,1), \quad \delta_0 = 0, \quad \kappa = \sigma = 0.3. \tag{14}$$

At these values the discretization is nearly identical to the exact transition (Gillespie, 1996). The prey's velocity and position are

$$v_{p,t} = \|v_p\|\,(\cos\psi_t, \sin\psi_t), \tag{15}$$

$$x_{p,t+1} = \text{wrap}(x_{p,t} + v_{p,t}\,\Delta t)\,. \tag{16}$$

The ellipse therefore fixes only the reference turning schedule, and since nothing pulls the prey back onto it, the realized path drifts away from the ellipse. Only easy levels are used.

Expert demonstrations. Demonstrations are generated by a greedy pursuit expert that accelerates at the maximum magnitude toward the prey, with a damping term inside a radius of $0.3\,\text{m}$ to avoid overshooting:

$$a_t = 8\,\frac{d_t}{\|d_t\|} \;-\; 2\,(v_{h,t} - v_{p,t})\,\mathbf{1}[\|d_t\| < 0.3]\,. \tag{17}$$

where $\mathbf{1}[\cdot]$ is the indicator function. The dataset contains 1000 successful episodes in low-dimensional mode. At each reset a level is drawn uniformly from $\{0, \ldots, 199\}$, so each level appears about five times on average. The environment seed is not fixed during collection, so repetitions of a level share the prey's ellipse and speed but differ in the hunter's initial position and in the heading noise. Episodes are capped at 400 steps, and only those ending in a capture are kept, while time-outs are discarded and do not count toward the 1000. For each step we store the 8-dimensional observation before the step, $(x_h, v_h, x_p, v_p)_t$, and the expert action a_t computed from it. Levels 200 to 249 are reserved for valiation. Test rollouts use levels 250 to 259, which were never collected, and each rollout fixes its level and seed so evaluation is deterministic.

Table 7: Hunter-Prey dataset description.

	Hunter-Prey (easy)
Episodes	1000
Steps	39831
Mean length	39.8
Median length	39
Min length	1
Max length	108
Obs dim	8
Action dim	2

E PUSH-T ENVIRONMENT.

We use the standard Push-T task (Chi et al., 2024), unmodified, and describe here only the three evaluation-time variations we introduce. Training is identical across the vanilla and two noise variants: the same human-demonstration dataset, with no noise and no randomized dynamics. The random-dynamics variant is the exception. Since a demonstration's actions are setpoints, they are only valid for the gains under which they were recorded, so we train it on an action-relabelled copy of the same demonstrations. For each episode we draw $k_p \sim \mathcal{U}(50, 150)$ and $\nu \sim \mathcal{U}(0.05, 0.5)$, set $k_v = \nu k_p$, and rewrite every action as the counterfactual setpoint that reproduces the recorded acceleration under those gains, $a'_t = x_t + (\ddot{x}_t + k_v \dot{x}_t)/k_p$ with $\ddot{x}_t = k_p^0(a_t - x_t) - k_v^0 \dot{x}_t$ and the default gains $k_p^0 = 100$, $k_v^0 = 20$. The observed trajectory is unchanged, so each copy is a valid demonstration of the same behaviour under different dynamics; we generate five independent copies per episode. Evaluation samples the same gain ranges bimodally, as described above, so the policy is trained on uniform coverage of the range and evaluated at its extremes.

Random dynamics. In push-T, the agent operates a point controller to push the T-block to its goal. Actions are converted to accelerations through the control model,

$$\ddot{\boldsymbol{x}} = k_p \cdot (\boldsymbol{u} - \boldsymbol{x}) - k_v \cdot \dot{\boldsymbol{x}}, \tag{18}$$

where $\boldsymbol{x}$ is the controller's position, dots denote temporal derivatives, $\boldsymbol{u}$ is the agent's action, and k_p and k_v are gain and damping parameters respectively. Given the controller's initial position and velocity and the sequences of actions taken, we can integrate this equation to exactly determine the controller's trajectory through space. To change environmental dynamics between episodes, we randomised both k_p and k_v.

Both k_p and the damping ratio $\rho_k = k_v/k_p$ are each resampled from a bimodal log-normal distribution favoring one of two extremes, so that episodes are either noticeably sluggish or noticeably twitchy rather than close to the default ($\rho_k = 20/100 = 0.2$). Formally, with $\mathrm{pick}(x, y, b) = x$ if $b = 0$ and y if $b = 1$,

$$b, b' \sim \mathrm{Bernoulli}(0.5), \tag{19}$$

$$k_p = \mathrm{clip}\big(\mathrm{pick}(50, 150, b) \cdot e^{\xi},\ 50,\ 150\big), \qquad \xi \sim \mathcal{N}\Big(0,\ (0.08 \ln 3)^2\Big), \tag{20}$$

$$\rho_k = \mathrm{clip}\Big(\mathrm{pick}(0.05, 0.5, b') \cdot e^{\xi'},\ 0.05,\ 0.5\Big), \qquad \xi' \sim \mathcal{N}\Big(0,\ (0.08 \ln 10)^2\Big), \tag{21}$$

$$k_v = k_p\, \rho_k, \tag{22}$$

where k_v is drawn as a fraction ρ_k of k_p rather than independently.

The dynamics model (18) defines a second-order equation of motion which, given the control actions, we can integrate to exactly reconstruct the controller's trajectory through space. However, since we are constrained to human teleoperated data, we used the equations of motion to relabel the actions for each trajectory so that the controller would still follow the same path in physical space for the new dynamics. We, in fact, generated five different relabellings for each trajectory, and because all five relabellings follow the same path in physical space, an agent that conditions on observations only has no way of telling them apart and therefore cannot infer the latent dynamics.

Noisy observations. Independent Gaussian noise is added to every coordinate of the observation at each step,

$$o_t = \mathrm{clip}\big(o_t^{\mathrm{true}} + \varepsilon_t,\ 0,\ 512\big)\,, \qquad \varepsilon_t \sim \mathcal{N}(0,\ \sigma^2 = 25^2), \tag{23}$$

where o_t^{true} is the noise-free observation. A noise magnitude of $\sigma^2 = 25$ pixels is about 5% of the field width.

Noisy actions. Independent Gaussian noise is added to the action before it reaches the environment,

$$a'_t = \mathrm{clip}(a_t + \varepsilon_t,\ 0,\ 512)\,, \qquad \varepsilon_t \sim \mathcal{N}(0,\ \sigma^2 = 25^2), \tag{24}$$

and a'_t, not a_t, is passed to the PD controller. This is execution noise, since the policy's next observation reflects the true state resulting from a'_t, not from the action it emitted.

Both noise wrappers are applied outside the environment, on top of the vanilla physics, and both use a fixed noise seed (42) across evaluation episodes.

F Reacher environment

We use a two-link planar reacher with the physics and assets of `Reacher-v2` (Brockman et al., 2016). The action $a_t \in [-1, 1]^2$ is a joint-torque vector, and the observation is the 8-dimensional vector

$$o_t = [\cos\theta_0,\ \cos\theta_1,\ \sin\theta_0,\ \sin\theta_1,\ \dot\theta_0,\ \dot\theta_1,\ g_x,\ g_y], \tag{25}$$

where θ are the joint angles and g the goal position. An episode succeeds once the fingertip stays within $r = 0.02$ of the goal for 10 consecutive steps, and is truncated after 250 steps. To test generalisation, we vary which region of the workspace the demonstrations cover.

Quadrant coverage. Goals lie in a disc of radius $R = 0.2$ around the arm's base, divided into four quadrants q1–q4. Training goals in quadrant k are drawn from a Gaussian centred on the quadrant's centroid c_k, rejecting samples that fall outside the quadrant or the disc, and the initial joint angles are drawn uniformly,

$$g \sim \mathcal{N}(c_k,\ \sigma^2 I_2), \qquad \sigma = 0.08, \tag{26}$$

$$\theta_0 \sim \mathcal{U}(-\pi, \pi), \qquad \theta_1 \sim \mathcal{U}(-3, 3). \tag{27}$$

We train on three nested coverages: q1, q1+q3, and q1+q2+q3.

Expert. Demonstrations come from a computed-torque expert. At the start of each episode it solves inverse kinematics for a target configuration $\theta^\star$ and keeps it fixed. The joint error $e = \theta^\star - \theta$ is wrapped to $(-\pi, \pi]$ for the unlimited joint θ_0 and taken directly for the joint-limited θ_1. The expert drives this error along a critically damped reference response by inverting the arm's true rigid-body dynamics, which realises the response exactly whenever the command is not saturated,

$$\ddot\theta_{\mathrm{des}} = \omega_n^2 e - 2\zeta\omega_n\dot\theta, \qquad \omega_n = 2\,\mathrm{rad/s},\quad \zeta = 1, \tag{28}$$

$$\tau = \big(M(\theta) + I_a\big)\,\ddot\theta_{\mathrm{des}} + h(\theta, \dot\theta) + D\dot\theta, \qquad a_t = \mathrm{clip}(\tau/\kappa,\ -1,\ 1)\,, \tag{29}$$

where M is the joint-space mass matrix, I_a the armature inertia, h the Coriolis, centrifugal and gravity bias, D the joint damping and κ the actuator gear ratio.

Evaluation. Each policy is evaluated on 100 goals per quadrant, in two regimes. *In-distribution* goals lie in the trained quadrants; *held-out* goals lie in the quadrants absent from training (q2+q3+q4, q2+q4 and q4 respectively). Both regimes sample goals uniformly over their quadrant,

$$g \sim \mathcal{U}\big([-R, R]^2\big), \tag{30}$$

with the same quadrant and disc rejection, so the two differ only in which region is sampled, not in how. Held-out performance therefore measures generalisation to regions of the workspace the demonstrations never visited.