ONE FOR ALL, ALL FOR ONE: COORDINATED MULTI-AGENT DIFFUSION STEERING VIA STOCHASTIC OPTIMAL CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Many structured outputs consist of interacting components, so modelling them with a single generative model requires learning both the component distributions and their interactions. We pursue a modular alternative: reuse independently trained component generators and learn only how to coordinate them to produce coherent structured outputs. Our framework, Coordinated Multi-Agent Diffusion Steering (CMDS), treats frozen pretrained diffusion models as *reusable generative primitives* and coordinates their reverse processes through a learned *control*. We formulate coordination as a stochastic optimal control problem, balancing an assembly-level reward that specifies the desired properties of the combined output against deviations from the pretrained dynamics. The learned control amortises this optimisation, allowing reuse across new task instances. Experiments show that CMDS can recover a known target distribution, satisfy different spatial constraints with the same trained control, and recover individual sources from blurred mixtures. Across multi-agent maze navigation, articulated robot planning, and text-conditioned human motion, CMDS turns frozen component models into coordinated multi-agent generators.

1 INTRODUCTION

Producing coherent structured outputs involves two distinct problems: modelling individual components and coordinating them when they are used together. End-to-end generative models absorb both into a single generative distribution. We instead build on a long-standing tradition of modular and compositional generative modelling (Grenander, 1978; Bienenstock et al., 1996; Williams, 2025; Du & Kaelbling, 2024). Component generators, pretrained without joint-system supervision, capture component-level variability, while coordination is learned separately at the level of the structured output. Diffusion models (Ho et al., 2020; Song et al., 2021; Lai et al., 2025) are well suited to this decomposition, as each pretrained reverse process provides a reusable component generative prior whose dynamics can be steered via a learned control while the underlying model remains fixed. Prior work on compositional diffusion generation typically resolves the required coordination at sampling time for each generated instance (Liu et al., 2022; Yang et al., 2023; Cao et al., 2025). This is sensible for one-off tasks, but becomes costly when related coordination problems recur across many instances. In settings where related coordination problems recur across many instances–for example, across relational design specifications, source-separation observations, or multi-agent start-goal configurations–the associated computation can instead be amortised into a reusable control. Building on recent work showing how controls or amortised samplers can steer pretrained diffusion processes towards task-specific target distributions (Domingo-Enrich et al., 2024; 2025; Venkatraman et al., 2024), we thus ask:

> *How can we learn an amortised control that coordinates frozen component diffusion models using a reward on their assembled output, while remaining close to their pretrained dynamics?*

A framework for coordinated generation. The proposed framework begins with independently trained component generators, whose distributions form a reference product distribution over component configurations. An assembly map combines their outputs into a candidate, while an

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.

assembly-level reward evaluates the candidate. A learned control then coordinates the component generators towards high-reward joint outputs. To make this concrete, consider planning the movements of several agents through a shared environment. Each frozen copy of a single-agent trajectory model generates a path through the environment, using an agent-specific start and end as conditioning inputs. Stacking these paths produces a candidate multi-agent plan. Although each path may avoid walls in isolation, the assembled plan may still be invalid if agents collide. This illustrates the role of the assembly-level reward. It expresses requirements not captured by the independently trained component models and provides the signal used to coordinate their generation. Generally, the framework requires *no training examples of valid structured outputs*, only **(i)** component-level data or pretrained component generators; **(ii)** an assembly map; and **(iii)** an assembly-level reward.

Continuous-time generative modelling. For each component i, let G^i denote the generator induced by the reverse-time sampling process of a pretrained diffusion model (Anderson, 1982; Haussmann & Pardoux, 1986; Song et al., 2021). We write $\hat{X}_s^i \in \mathbb{R}^d$ for component i's state and $\hat{X}_s = (\hat{X}_s^1, \dots, \hat{X}_s^N) \in \mathbb{R}^D$, $D = Nd$, for the joint state; hats denote reverse-time quantities. We use $s \in [0,1]$ for reverse (generative) time with $s = 0$ corresponding to the noise state and $s = 1$ to the generated sample. The reverse process for component i starts from $\hat{X}_0^i \sim \hat{p}_0^i$, where $\hat{p}_0^i$ is a standard Gaussian, and terminates at $\hat{X}_1^i \sim \tilde{\pi}^i$, where $\tilde{\pi}^i$ denotes the learned component distribution. Suppressing the index i, we write the reference reverse-time diffusion as,

$$\mathrm{d}\hat{X}_s = \hat{f}_s(\hat{X}_s)\,\mathrm{d}s + \hat{\sigma}_s\,\mathrm{d}\bar{W}_s, \qquad \hat{X}_0 \sim \hat{p}_0. \tag{1}$$

Here, $\hat{f}_s$ denotes the learned reverse drift, $\hat{\sigma}_s$ is the reverse-time diffusion scale, and $\bar{W}_s$ is a Brownian motion in reverse time. The terminal law of $\hat{X}_1$ is the learned model distribution $\tilde{\pi}$, which approximates the corresponding exact terminal law π. For N pretrained component samplers, their independent reference processes form the joint product reverse process $\hat{X}_s = (\hat{X}_s^1, \dots, \hat{X}_s^N)$, with $\hat{X}_0^i \sim \hat{p}_0^i$ and $\hat{X}_1^i \sim \tilde{\pi}^i$. Its terminal marginal is therefore the product law $\rho = \tilde{\pi}^1 \otimes \cdots \otimes \tilde{\pi}^N$.

Controlled diffusion and stochastic optimal control (SOC). Diffusion sampling admits a SOC interpretation in which a reward tilt is realised by controlling a reference diffusion under a quadratic path-action penalty (Nüsken & Richter, 2021; Berner et al., 2024; Uehara et al., 2024; Domingo-Enrich et al., 2025). For a single diffusion model ($N = 1$), the problem can be written as,

$$\begin{aligned} \min_{u \in \mathcal{U}} \quad & \mathbb{E}_{\hat{\mathbb{P}}^u}\left[\int_0^1 \frac{1}{2}\|u_s(\hat{X}_s^u)\|^2\,\mathrm{d}s - \lambda r(\hat{X}_1^u)\right], \\ \text{s.t.} \quad & \mathrm{d}\hat{X}_s^u = \left[\hat{f}_s(\hat{X}_s^u) + \hat{\sigma}_s u_s(\hat{X}_s^u)\right]\mathrm{d}s + \hat{\sigma}_s \mathrm{d}\bar{W}_s, \qquad \hat{X}_0^u \sim \hat{p}_0. \end{aligned} \tag{2}$$

Here, $u \in \mathcal{U}$ is an admissible control field, $\hat{\mathbb{P}}^u$ is the path measure induced by the controlled process $\hat{X}^u$, $\lambda > 0$ controls the reward strength, $\hat{f}_s$ and $\hat{\sigma}_s$ denote the reverse-time drift and diffusion scale of the pretrained model, respectively, $\bar{W}_s$ is a Brownian motion, and $\hat{p}_0$ is the initial noise distribution. The quadratic running cost corresponds to the control energy and, under the Girsanov conditions and a shared initial law, equals the relative entropy between the controlled and reference path measures (Nüsken & Richter, 2021; Berner et al., 2024). We extend this SOC formulation from a single diffusion process to the joint state $\hat{X}_s$ of multiple pretrained diffusion agents.

Contributions. Our contributions can be summarised as follows:

- **Coordination as modular post-training.** We formulate compositional generation as stochastic optimal control over the joint reverse dynamics of frozen pretrained component diffusion models. The independent component processes define a reference product law, while an assembly-level reward induces a coordinated reward tilt, realised by a learned control that couples their reverse dynamics.
- **Comprehensive empirical validation.** Across four settings, we demonstrate stochastic-control target-law recovery, amortised relational assembly, compositional posterior inference for source separation, and multi-agent coordination using frozen component models. In PointMaze (Fu et al., 2020), CMDS coordinates up to four agents with high joint success and generalises to unseen start–goal regions. Experiments with publicly available KUKA

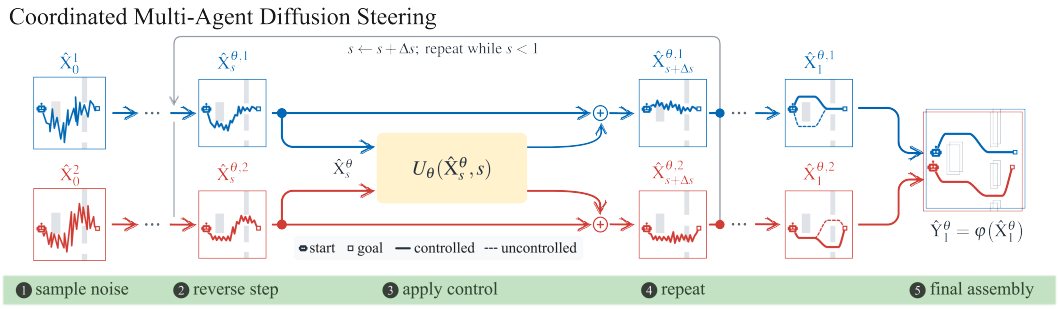


Figure 1: **Overview of CMDS.** Frozen component diffusion models are sampled concurrently, while a joint control conditions on the multi-agent state and applies component-wise controls throughout the reverse process, yielding a coordinated assembled output.

energy-based checkpoints (Luo et al., 2024) and a single-person MDM prior (Tevet et al., 2023) extend the evaluation to multi-arm planning and multi-person motion.

Further discussion on relevant background concepts can be found in Appendix A.

2 Coordinated Multi-Agent Diffusion Steering (CMDS)

We consider N pretrained diffusion agents, each generating one component of a structured output. Using the joint state $\hat{X}_s$ defined above, the assembled state is $\hat{Y}_s = \varphi(\hat{X}_s)$, where $\varphi : \mathbb{R}^D \to \mathbb{R}^m$ is a fixed assembly operator. Without coordination, the component processes evolve independently and need not produce a globally consistent assembly. We extend the single-process SOC formulation in eq. (2) to the joint state, with $u_s : \mathbb{R}^D \to \mathbb{R}^D$ now denoting the coordination control.

Importantly, CMDS is non-autoregressive across components. All N reverse processes evolve concurrently along the same generative-time axis, and at each time s, the control $u_s^i(\hat{X}_s)$ applied to agent i may depend on the current state of every agent. No component is therefore generated and fixed before the others. This avoids imposing an arbitrary ordering over components and enables corrections throughout generation, which is particularly important when the assembly-level criterion depends on their joint configuration. The component updates may also be evaluated in parallel, subject to the cost of the coordination modules.

With this joint representation, coordination amounts to steering the combined reverse process so that its assembled terminal sample satisfies an assembly-level criterion while remaining close to the independent pretrained dynamics. Writing $x = (x^1, \cdots, x^N) \in \mathbb{R}^D$, we denote the joint reference reverse drift by $\hat{f}_s(x) = (\hat{f}_s^1(x^1), \ldots, \hat{f}_s^N(x^N))$. The controlled terminal assembly is $\hat{Y}_1^u = \varphi(\hat{X}_1^u) = \varphi(\hat{X}_1^{u,1}, \ldots, \hat{X}_1^{u,N})$, and we formulate the problem as the SOC objective,

$$\min_u \mathbb{E}_{\hat{\mathbb{P}}^u} \left[\int_0^1 \frac{1}{2} \sum_{i=1}^N \left\| u_s^i(\hat{X}_s^u) \right\|^2 \mathrm{d}s - \lambda r\left(\varphi(\hat{X}_1^u)\right) \right], \tag{3}$$

where $r(\varphi(\hat{X}_1^u))$ is a reward of the assembled terminal output and measures the global consistency, and the quadratic control cost penalises deviation from the pretrained agents' distribution. The central hypothesis of CMDS is that locally valid components can be learned, while the dependencies required for global validity can be introduced through an assembly-level criterion. Further details on the forward noising process, its time reversal, the associated score-based parameterisation, and the controlled reverse-time SDE are provided in Appendices B.1 and B.2.

2.1 Learning Coordinated Reverse Dynamics

The exact optimal path distribution $\hat{\mathbb{P}}^{u^\star}$ is generally intractable. We therefore approximate the non-parametric optimal control $u_s^\star(x)$ by learning a parametrised neural control field,

$$U_\theta(x, s) = \left(U_\theta^1(x, s), \ldots, U_\theta^N(x, s)\right).$$

The component controls may either share parameters across agents or use agent-specific parameters; in both cases, each U_θ^i conditions on the full joint state. The resulting controlled reverse diffusion implicitly defines a stochastic generator $G_\theta = (G_\theta^1, \dots, G_\theta^N)$, with component dynamics,

$$\mathrm{d}\hat{X}_s^{\theta,i} = \left[\hat{f}_s^i(\hat{X}_s^{\theta,i}) + \hat{\sigma}_s\, U_\theta^i(\hat{X}_s^\theta, s)\right]\mathrm{d}s + \hat{\sigma}_s\,\mathrm{d}\bar{W}_s^i, \qquad i = 1, \dots, N. \tag{4}$$

We denote the corresponding assembled process by $\hat{Y}_s^\theta := \varphi(\hat{X}_s^\theta)$. Equivalently, $G_\theta^i : \hat{X}_0 \mapsto \hat{X}_1^{\theta,i}$ denotes the stochastic solution map induced by the controlled reverse dynamics, with $\hat{X}_0^i \sim \hat{p}_0^i$. The corresponding training objective balances control cost and global reward,

$$\mathcal{J}(\theta) = \mathbb{E}_{\hat{\mathbb{P}}^\theta}\left[\int_0^1 \frac{1}{2}\sum_{i=1}^N \|U_\theta^i(\hat{X}_s^\theta, s)\|^2 \mathrm{d}s - \lambda r(\varphi(\hat{X}_1^\theta))\right], \tag{5}$$

where $\hat{\mathbb{P}}^\theta$ is the path distribution induced by the parametrised controlled process.

Adjoint matching CMDS instantiation (Domingo-Enrich et al., 2025). To train the control efficiently, we use an adjoint matching (AM) version of the original objective. Along sampled controlled trajectories, an adjoint variable estimates the direction in which each component should move to improve the final global reward. The control is trained to match this optimal correction,

$$\mathcal{L}_{\mathrm{AM}}(\theta) = \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^{\bar{\theta}}}\left[\int_0^1 \left\| U_\theta(\hat{X}_s^{\bar{\theta}}, s) + \hat{\sigma}_s a_s^{\bar{\theta}}\right\|^2 \mathrm{d}s\right], \qquad \bar{\theta} = \mathrm{stopgrad}(\theta) \tag{6}$$

where $a_s^{\bar{\theta}}$ denotes the adjoint evaluated along the frozen controlled trajectory $\hat{X}^{\bar{\theta}}$, solving,

$$\frac{\mathrm{d}}{\mathrm{d}s}a_s^{\bar{\theta}} = -D_x\left(\hat{f}_s(\hat{X}_s^{\bar{\theta}}) + \hat{\sigma}_s U_{\bar{\theta}}(\hat{X}_s^{\bar{\theta}}, s)\right)^\top a_s^{\bar{\theta}} - \frac{1}{2}\nabla_x \sum_{i=1}^N \|U_{\bar{\theta}}^i(\hat{X}_s^{\bar{\theta}}, s)\|^2,$$

$$a_1^{\bar{\theta}} = -\lambda D_x\varphi(\hat{X}_1^{\bar{\theta}})^\top \nabla_y r(\hat{Y}_1^{\bar{\theta}}),$$

where D_x denotes the Jacobian with respect to x. The full pathwise adjoint, its block-wise multi-agent form, and the corresponding adjoint matching surrogate are derived in Appendix C.

Strategies for learning the coordination control. CMDS is defined by jointly controlling the reference product process under the SOC objective in eq. (3), whereas AM is one possible strategy for learning the coordination control U_θ, rather than part of the framework. When the reverse solver and assembly-level reward are differentiable, U_θ may instead be learned by directly optimising the discretised SOC objective; we refer to reverse-mode differentiation through the solver transitions as *discrete-adjoint* optimisation (Domingo-Enrich, 2024). Alternatively, path-space objectives such as relative trajectory balance (Venkatraman et al., 2024) could be adapted to the joint controlled reverse process to target the same reward-tilted law, provided that the corresponding controlled-to-reference trajectory-density ratio is tractable. We adopt AM since it propagates gradient information without backpropagating through the complete computational graph of the sampling rollout.

2.2 Path-Space Interpretation of Coordination

The SOC objective in eq. (3) admits a path-space interpretation. Let $\hat{\mathbb{P}}$ denote the reference path law induced by the independent pretrained reverse diffusion processes, and let $\hat{\mathbb{P}}^u$ denote the controlled path law. By Girsanov's theorem (Øksendal, 2003), under the usual change-of-measure conditions and a shared initial law, the quadratic control cost equals the path-space relative entropy, $D_{\mathrm{KL}}\big(\hat{\mathbb{P}}^u \,\|\, \hat{\mathbb{P}}\big) = \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^u}[\int_0^1 \|u_s(\hat{X}_s^u)\|^2\,\mathrm{d}s]$. The control problem admits the path-space form,

$$u^\star \in \arg\min_u \left\{ D_{\mathrm{KL}}\big(\hat{\mathbb{P}}^u \,\|\, \hat{\mathbb{P}}\big) - \lambda\mathbb{E}_{\hat{\mathbb{P}}^u}\left[r\Big(\varphi(\hat{X}_1^u)\Big)\right]\right\}. \tag{7}$$

This formulation makes explicit that coordination balances the assembly-level reward of the assembled terminal object against deviation from the pretrained product process. By the standard variational identity underlying KL-regularised stochastic control, the optimal controlled path law is an exponential tilt of the reference law. Conditionally on the initial noise state,

$$\frac{\mathrm{d}\hat{\mathbb{P}}^{u^\star}\big(\cdot\,|\hat{X}_0\big)}{\mathrm{d}\hat{\mathbb{P}}\big(\cdot\,|\hat{X}_0\big)}\big(\hat{X}_\cdot\big) \propto \exp\Big(\lambda r\big(\varphi(\hat{X}_1)\big)\Big). \tag{8}$$

Thus, optimal coordination re-weights joint trajectories according to the global reward of their assembled output. The change of measure argument on the joint process is in Appendix B.3.

Remark 1 (Reward-induced coordination). In the idealised complete-noising limit, where $\hat{X}_0 \perp \hat{X}_1$, marginalising the optimal controlled path law yields the terminal distribution,

$$\rho^\star(x) \propto \rho(x)\, \exp(\lambda r(\varphi(x)))\,, \qquad x \in \mathbb{R}^D, \tag{9}$$

where $\rho = \tilde{\pi}^1 \otimes \cdots \otimes \tilde{\pi}^N$ is the terminal law of the independently sampled pretrained agents. When $r \circ \varphi$ is non-separable across components, this reward tilt breaks the product structure of ρ and introduces the task-specific dependencies required by the assembled system. The learned control provides a dynamic realisation of this change of measure. Rather than explicitly reweighting complete pretrained trajectories, it steers the joint reverse dynamics towards the reward-tilted terminal law. A formal statement is provided in Appendix B.4.

3 Experimental Evaluation

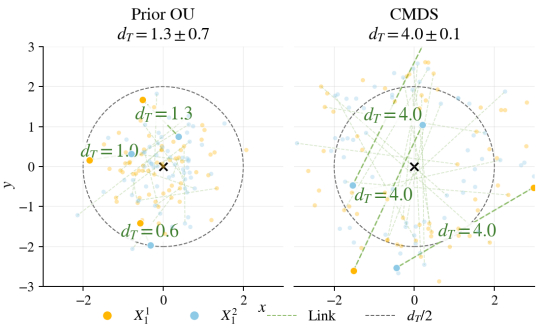


Figure 2: **Two-agent OU problem.** Terminal samples from the independent OU reference process and the rotation-equivariant CMDS control for a prescribed separation $d_T = 4$. Yellow and blue points denote the two agents, and each line joins the terminal states belonging to the same sample.

We evaluate CMDS along three axes: **(i)** a tractable stochastic-control problem with a known optimal solution, to assess recovery of the target distribution rather than constraint satisfaction alone; **(ii)** source separation, to assess posterior inference from blurred mixtures when paired source–mixture training data are unavailable; and **(iii)** multi-agent motion, to assess scalability to larger models and more agents using off-the-shelf pretrained checkpoints, including zero-shot transfer to changed geometry and additional agents without retraining the control. Additionally, we study constraint satisfaction and amortised coordination across relational specifications in Appendix D.2. These constrained visual-assembly experiments provide a controlled test-bed to isolate the effects of component factorisation, reward design, amortisation, and optimisation.

3.1 Two-agent OU coordination

Setup. We begin with a tractable two-agent stochastic optimal-control problem for which a value-function oracle is available. The experiment asks whether the learned control recovers the full stochastic-control target law, rather than satisfying its terminal-distance constraint. We consider two independent Ornstein–Uhlenbeck (OU) reference processes $X_t^1, X_t^2 \in \mathbb{R}^2$, with joint state $X_t = (X_t^1, X_t^2) \in \mathbb{R}^4$. The agents are initialised and evolve according to

$$X_0^1, X_0^2 \overset{\text{iid}}{\sim} \mathcal{N}(0, I_2), \qquad dX_t^i = -X_t^i\, dt + dW_t^i, \qquad t \in [0,1], \quad i \in \{1,2\},$$

where W^1 and W^2 are independent two-dimensional Brownian motions. Consequently, the joint initial law is $X_0 \sim \mathcal{N}(0, I_4)$. The reference components evolve independently, while coordination is induced by the terminal reward $r_{\text{OU}}(x^1, x^2) = -(\|x^2 - x^1\|_2 - d_T)^2$, with $d_T = 4$, and realised through the resulting joint optimal control. Because the problem reduces to the relative coordinate $R_t = X_t^2 - X_t^1$, we can compute a value-function oracle for the optimal stochastic control (refer to Appendix D.1). We also include importance sampling (IS) (Robert & Casella, 2004), using samples from the uncontrolled OU process re-weighted according to the terminal reward.

Constraint satisfaction. As we observe in fig. 2, the uncontrolled process yields a terminal separation of 1.3 ± 0.7, whereas the rotation-equivariant CMDS control achieves 4.0 ± 0.1. The equivariant control is used only to respect a known symmetry of the oracle target; it is not a requirement of CMDS.

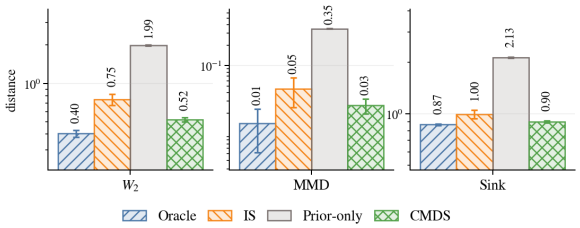


Figure 3: **Recovery of the SOC target.**

Distributional recovery. To evaluate recovery of the full terminal law, we compare samples against the stochastic-control target using the 2-Wasserstein distance (W_2), maximum mean discrepancy (MMD), and Sinkhorn (Sink) divergence over 5 seeds. As reported in fig. 3, CMDS approaches this finite-sample oracle floor[1] and outperforms importance sampling (IS). These results show that CMDS does more than reach the prescribed separation as it also recovers the distributional structure of the reward-tilted target law. Appendix D.1 shows that an unconstrained control can satisfy the scalar distance constraint while failing to recover angular structure of the target distribution.

3.2 Source separation

Compositional posterior formulation. We consider posterior inference in which N frozen pretrained diffusion samplers independently define component priors through their reverse processes, with reference terminal samples $\hat{X}_1^i \sim \tilde{\pi}^i$. Let $X = (X^1, \dots, X^N)$ denote the unknown component state, whose non-injective additive assembly is $Y = \varphi(X) := \sum_{i=1}^N X^i$, and suppose that we observe a blurred mixture $b = \mathrm{A}_\tau(Y) + \varepsilon$, where A_τ is a known Gaussian blurring operator with standard deviation τ, and $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2 I)$. Since the additive assembly φ is non-injective, the observation does not uniquely identify its constituent sources; the independently pretrained component priors therefore regularise the resulting compositional posterior over valid decompositions. Assuming Gaussian observation noise, we define the assembly-level reward $r_b(y) := -\frac{1}{2\sigma_\varepsilon^2}\|\mathrm{A}_\tau(y) - b\|^2$. The pretrained diffusion models define the reference product law $\rho = \tilde{\pi}^1 \otimes \cdots \otimes \tilde{\pi}^N$, while CMDS performs compositional posterior inference by coordinating their reverse dynamics, producing controlled terminal components $\hat{X}_1^\theta = (\hat{X}_1^{\theta,1}, \dots, \hat{X}_1^{\theta,N})$ whose assembly $\hat{Y}_1^\theta = \varphi(\hat{X}_1^\theta)$ jointly explains the observation b, without requiring training samples from the joint source distribution. For comparison, we adapt diffusion posterior sampling (DPS) (Chung et al., 2023) to the product prior, paralleling the posterior-sampling construction of Luan et al. (2025). At each reverse step, we evaluate the observation likelihood on the Tweedie estimates (Efron, 2011) of both processes and apply the resulting component-wise gradients to their reverse updates; details are given in Appendix D.3.

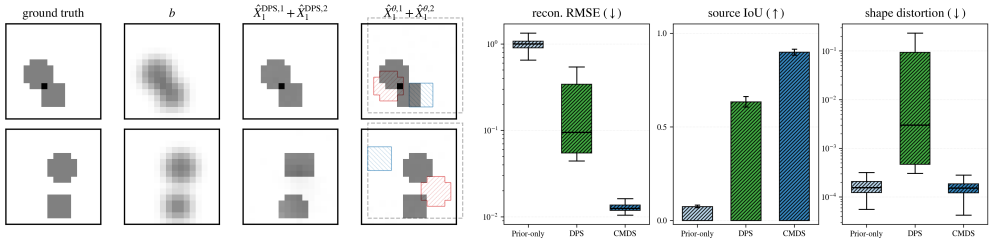


Figure 4: **Source separation from a blurred mixture** ($\tau = 1$)**.** Left: circle–square reconstructions for two held-out mixtures; red and blue hatching behind the CMDS reconstructions shows same-seed prior-only sources. Right: comparison of the frozen prior, DPS, and CMDS over 1,024 held-out mixtures, measuring clean-mixture recovery (RMSE, ↓), labelled-source overlap (IoU, ↑), and location-independent shape distortion (↓). Full metric definitions and additional results under milder blur ($\tau = 0.5$) are provided in Appendix D.3 and fig. 16.

[1]To indicate the finite-sample floor of each two-sample metric, we compare two independent sample sets from the value-function oracle, using the same number of samples as for each evaluated method. We refer to this oracle-to-oracle discrepancy as the finite-sample oracle baseline.

	Medium				Large			
	$N = 2$		$N = 4$		$N = 2$		$N = 4$	
Method	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑
Independent	0.174 ±0.005	0.195 ±0.008	0.068 ±0.005	0.010 ±0.005	0.195 ±0.016	0.219 ±0.000	0.039 ±0.000	0.049 ±0.005
Best-of-256	0.216 ±0.005	0.263 ±0.016	0.076 ±0.005	0.044 ±0.005	0.232 ±0.009	0.328 ±0.021	0.076 ±0.005	0.070 ±0.000
FK-Steering	0.797 ±0.028	0.820 ±0.034	0.398 ±0.036	0.331 ±0.016	0.924 ±0.030	0.953 ±0.008	0.622 ±0.009	0.729 ±0.023
DPS	0.964 ±0.024	0.956 ±0.016	0.958 ±0.025	0.919 ±0.033	**0.984** ±0.016	**0.977** ±0.000	0.880 ±0.030	0.927 ±0.009
CMDS	0.990 ±0.012	0.919 ±0.016	0.701 ±0.009	0.669 ±0.005	0.911 ±0.016	0.831 ±0.016	0.880 ±0.012	0.753 ±0.027
CMDS + TRG	**0.995** ±0.005	**0.992** ±0.000	**0.974** ±0.012	**0.935** ±0.005	0.966 ±0.005	0.898 ±0.016	**0.966** ±0.012	**0.977** ±0.000

Table 1: **PointMaze joint success.** Mean ± standard deviation (std) across sampling seeds. Training-seed variability is reported separately in Appendix D.4.8. Each pool contains 128 instances per layout and agent count. For CMDS and CMDS+TRG, we train five independent seeds and report the seed with median Pool 1 success for each setting, breaking ties using Pool 2.

Recovery under ambiguity. As shown in fig. 4, CMDS achieves the strongest overall recovery under the challenging $\tau = 1$ blur. It yields the lowest clean-mixture reconstruction error and the highest source intersection over union (IoU), while maintaining the low shape distortion induced by the frozen component priors. DPS substantially improves source localisation over independent prior sampling, but markedly distorts the components and yields a larger reconstruction error than CMDS. This localisation–distortion trade-off reflects the non-identifiability of the problem. The likelihood depends only on the assembled mixture $X^1 + X^2$, through $\mathrm{A}_\tau(X^1 + X^2)$, and therefore contains no direct information about the differential mode $X^1 - X^2$ that distinguishes the individual sources. Consequently, the step-wise likelihood corrections used by DPS can improve the reward by redistributing or deforming mass in ways that are poorly aligned with the pretrained component distributions. By contrast, rather than applying step-wise likelihood guidance, CMDS learns joint feedback over the product of the pretrained reverse processes, while the quadratic path-action penalty favours small coordinated deviations from the frozen dynamics, helping preserve component structure. Quantitative results under milder blur are provided in Appendix D.3.

3.3 From Single-Agent Priors to Coordinated Multi-Agent Motion

Coordination across motion domains. We investigate multi-agent navigation problems where agents move through a shared environment, between prescribed start- and end-points, while satisfying task-specific constraints and avoiding collisions. We consider planar navigation (PointMaze) (Fu et al., 2020), articulated robot planning (KUKA) (Luo et al., 2024), and text-conditioned human motion (MDM) (Tevet et al., 2023). Pretrained single-agent priors generate component trajectories, the assembly operator combines them, and an assembly-level reward coordinates their movement. Throughout, the component priors remain frozen and only the coordination control is learned. PointMaze and KUKA condition the component models on prescribed start and end points. For MDM, component models are text-conditioned and end points are supplied to the coordination control. Success requires every agent to satisfy the task-specific goal, avoid collisions, and pass task-specific feasibility checks. Further details are in Appendices D.4 to D.6.

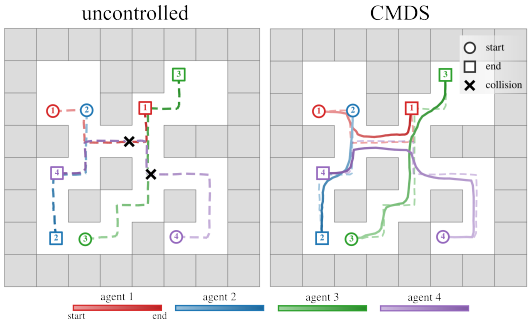


Figure 5: **PointMaze (Medium) coordination.** Dashed (--) and solid (—) paths use the same initial noise. Uncontrolled trajectories collide, whereas CMDS steers to avoid collisions while preserving their start–end pairs.

PointMaze: amortised multi-agent navigation. We pretrain one conditional single-agent trajectory prior for each of the *"Medium"* and *"Large"* layouts (Fu et al., 2020), then coordinate two or four copies. The reward penalises insufficient wall clearance, inter-agent collisions, path length, and trajectory acceleration. We evaluate on two sets of unseen start–end instances. Pool 1 places starts and goals in navigable grid squares used during control training, whereas Pool 2 uses squares excluded from control training (Appendix D.4.3). Table 1 and fig. 5 compare independent sampling,

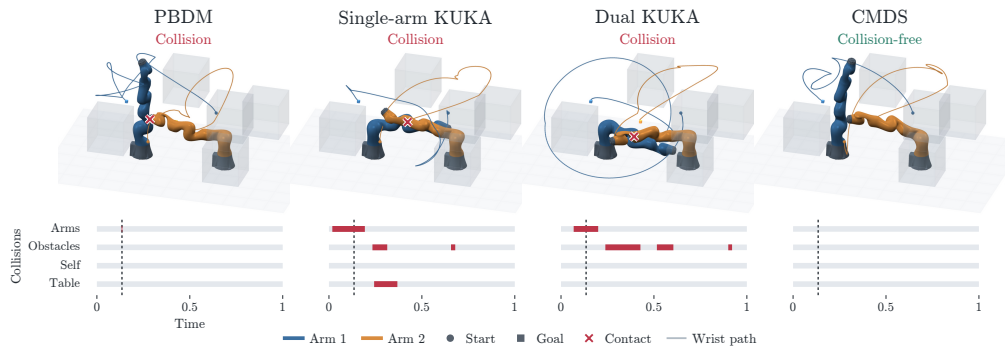


Figure 6: **A dual-arm coordination example.** Poses share normalised path progress $s = 0.135$; crosses mark arm–arm contact, and red intervals mark strict mesh collisions. "Time" denotes normalised path progress, not execution time. PBDM uses $K = 6$, $R = 5$, versus $K = 1$ for the other methods; its modified, native-accepted path fails strict checking between knots. This outcome-selected illustration is not a matched-budget comparison. Additional synchronised poses and clearance traces are provided in Appendix D.5.

Best-of-256, FK-Steering (Singhal et al., 2025), DPS (Chung et al., 2022), and our learned control. We evaluate two parameterisations of the coordination control: CMDS uses an attention-based control conditioned on the joint multi-agent state, while CMDS+TRG additionally incorporates the *Tweedie reward gradient* (TRG), namely the reward gradient evaluated at the Tweedie estimate (Denker et al., 2024; Venkatraman et al., 2024). CMDS+TRG improves or matches the reported base-CMDS rates, achieving Pool 2 success between 89.8% and 99.2% across the reported settings; DPS is the best performing on *"Large"* with two agents. Across five independent training seeds per setting, CMDS+TRG achieves mean Pool 2 success between 90.6% and 98.3%, with training-seed standard deviations no larger than 1.5 percentage points (Appendix D.4.8). To stabilise adjoint-matching training against large residuals, we use a pseudo-Huber regression loss in place of the squared loss; the calibration procedure and comparison are detailed in Appendix D.4.7.

KUKA: multi-arm planning from single-arm priors. We coordinate two seven-joint robot arms in a shared environment containing obstacles. Each task specifies the obstacle layout and the start and goal configurations of both arms. A shared CMDS+TRG control coordinates frozen pretrained single-arm diffusion models to generate their paths jointly. Publicly released checkpoints (Luo et al., 2024) predict noise as an energy gradient, $\epsilon_\psi = \nabla_x E_\psi$. Adjoint propagation therefore requires Hessian–vector products through the energy, while the checkpoint parameters remain frozen. Strict success requires both arms to reach their goals, remain within the allowed angular range of every robot joint, and pass checks for inter-arm collisions, self-collisions, and collisions with the obstacles and worktable. We evaluate on 413 tasks and denote by K the number of initial candidate multi-arm plans generated per task. With one candidate plan, CMDS+TRG achieves 77.3% strict success, compared with 11.4% for independent single-arm sampling and 51.0% when adding collision guidance (see table 2). Using the separately pretrained two-arm diffusion prior, guided Dual KUKA achieves 74.3%, while PBDM achieves 60.9% using the same prior within its native candidate-selection and refinement procedure. We also apply the learned control without retraining to shifted or enlarged obstacles, wider arm-base spacing, and additional arms (table 3). On a separate matched set of 209 tasks with five arms and four candidates per task ($K = 4$), CMDS achieves 68.3% strict success, compared with 48.3% for guided single-arm priors. Additional information is provided in Appendix D.5.

Method	Strict success (%, ↑)	Planning time (s, ↓)
PBDM	60.9 ±8.0	0.427 ±0.003
Dual KUKA + guidance	74.3 ±5.7	0.356 ±0.002
Dual KUKA prior	36.3 ±8.7	0.274 ±0.033
Single-arm priors + guidance	51.0 ±0.6	0.355 ±0.001
Single-arm priors	11.4 ±1.5	**0.193** ±0.002
CMDS+TRG	**77.3** ±0.7	0.368 ±0.002

Table 2: **Dual KUKA planning with one initial joint candidate.** Mean ± sample std. Planning time is the median value and includes scene setup, generation, verification and optional refinement.

Method	K	Geometry transfer			Number of arms		
		Shifted obstacles	Larger obstacles	Wider bases	3	4	5
Single-arm KUKA	1	53.4 ±4.0	14.3 ±1.3	49.2 ±2.4	33.8 ±0.7	28.4 ±1.7	19.3 ±2.0
+ guidance	4	83.2 ±2.8	42.1 ±4.1	84.0 ±0.7	69.1 ±1.5	55.7 ±3.5	48.3 ±2.4
Dual KUKA	1	68.9 ±3.9	31.1 ±10.0	58.2 ±8.1	–	–	–
+ guidance	4	89.7 ±0.8	66.9 ±3.8	87.8 ±2.8	–	–	–
CMDS	1	**74.0** ±1.7	**42.7** ±3.7	**72.1** ±2.2	**59.5** ±0.3	**52.3** ±3.4	**41.9** ±1.7
	4	**91.6** ±0.5	**73.0** ±1.0	**91.8** ±0.8	**86.8** ±1.0	**76.2** ±2.6	**68.3** ±1.2

Table 3: **Zero-shot KUKA generalisation.** Strict success (%), mean ± sample std. across three sampling seeds, for $K = 1, 4$. The left block reports geometry transfer; the right block reports transfer to larger numbers of arms.

Method	Success (%) ↑	Collision free (%) ↑	Goal err. (m) ↓	Time (s) ↓
Best-of-256	0.0 ±0.0	16.7 ±5.9	2.456 ±0.146	69.90 ±0.29
DPS	0.0 ±0.0	61.7 ±10.8	3.751 ±0.284	8.57 ±0.02
PCD	4.2 ±2.9	8.3 ±2.9	0.500 ±0.000	9.99 ±0.06
PCD++	75.0 ±6.6	**98.3** ±2.3	0.487 ±0.001	145.34 ±4.46
CMDS	**90.0** ±6.3	90.0 ±6.3	**0.158** ±0.008	**0.54** ±0.00

Table 4: **Four-person motion coordination.** Mean ± sample std. across five sampling seeds on 24 scenes; CMDS uses one trained control. Failures count as zero successes. Full metrics appear in table 15.

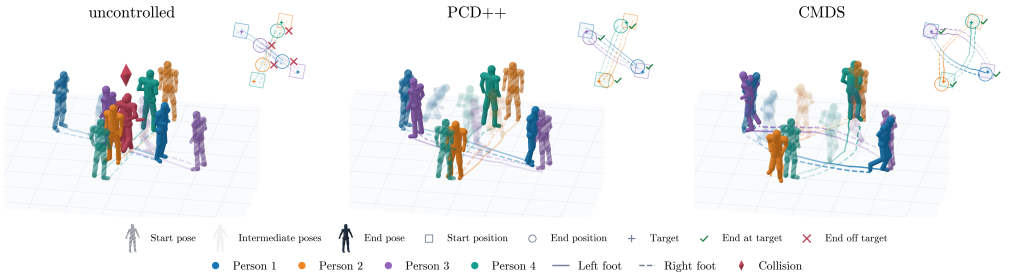


Figure 7: **Four-person motion coordination.** We compare uncontrolled diffusion, PCD++, and CMDS on the four-person coordination task. The prompts are *"A person moves forward at a jogging pace."* for Person 1 (●) and Person 4 (●), and *"A person walks forward."* for Person 2 (●) and Person 3 (●). Solid and dashed curves show left- and right-foot trajectories, respectively, with intermediate poses faded; the top-down insets summarise the trajectories and target completion (✓/×). Red diamonds (♦) mark inter-person collisions. Here, uncontrolled diffusion produces collisions and misses targets, whereas PCD++ and CMDS reach all targets without collisions. Full results are reported in Appendix D.6; fig. 22 extends the qualitative comparison to all baselines.

MDM: from single-person motion to coordinated crowds. We use the 50-step MDM checkpoint (Tevet et al., 2023) to coordinate four independently text-conditioned people over six-second motions (see Appendix D.6). A generated scene is successful when all four people move from their starting positions (□) to their targets (+), shown in the top-down insets of fig. 7, and their motions pass both the inter-person collision and kinematic-validity checks. Baselines include Best-of-256, DPS, PCD (Luan et al., 2025), and our PCD++ extension, which jointly optimises noisy motions through the frozen denoiser for goal attainment and clearance (see Algorithm 2). Table 4 and fig. 7 report 90% joint success for CMDS versus 75% for PCD++. PCD++ has the higher collision-free rate, but lower joint success and longer sampling time.

4 Conclusion

We introduced CMDS, a modular post-training framework for coordinating frozen pretrained diffusion models through stochastic optimal control. An assembly-level reward specifies joint requirements, while an amortised control couples the component reverse processes under a quadratic path-action penalty. Experiments demonstrate target-law recovery in a tractable setting, amortised relational assembly, and source recovery, while released KUKA and MDM priors extend coordination to multi-arm planning and multi-person motion. Transfer to additional robot arms further illustrates the reuse enabled by learning coordination separately from component generation. Together, these results show that frozen component models can support structured tasks beyond their individual training settings through learned interactions. Future work includes improving reward design, characterising how robust matching and control parameterisation affect target-law approximation, and extending coordination to larger, more heterogeneous generative models.

AI USE STATEMENT

In this work, we used generative AI tools to assist with text editing, reformatting, and polishing; LaTeX formatting and styling; and software-code review, debugging, and editing. We also used generative AI to generate code for visualisation and presentation of results; all such code was reviewed by the authors and was not used to implement the methods or produce the experimental results reported in this paper. All AI-assisted text and code were reviewed and edited by the authors. AI-assisted code was manually inspected and tested before use.

We did not use generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results; assisting with translation is not applicable to this work.

We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

REFERENCES

Brian D. O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. doi: 10.1016/0304-4149(82)90051-5. URL https://doi.org/10.1016/0304-4149(82)90051-5.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1737–1752. PMLR, 2023. URL https://proceedings.mlr.press/v202/bar-tal23a.html.

Riccardo Barbano, Alexander Denker, Zeljko Kereta, Runchang Li, and Francisco Vargas. CMAD: Cooperative multi-agent diffusion via stochastic optimal control. *arXiv preprint arXiv:2602.10933*, 2026. doi: 10.48550/arXiv.2602.10933. URL https://arxiv.org/abs/2602.10933.

Jonathan T. Barron. A general and adaptive robust loss function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4331–4339, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Barron_A_General_and_Adaptive_Robust_Loss_Function_CVPR_2019_paper.html.

Julius Berner, Lorenz Richter, and Karen Ullrich. An optimal control perspective on diffusion-based generative modeling. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2211.01364.

Elie Bienenstock, Stuart Geman, and Daniel Potter. Compositionality, MDL priors, and object recognition. In *Advances in Neural Information Processing Systems*, volume 9, 1996. URL https://proceedings.neurips.cc/paper/1996/hash/17fafe5f6ce2f1904eb09d2e80a4cbf6-Abstract.html.

Denis Blessing, Julius Berner, Lorenz Richter, Carles Domingo-Enrich, Yuanqi Du, Arash Vahdat, and Gerhard Neumann. Trust region constrained measure transport in path space for stochastic optimal control and inference. *arXiv preprint arXiv:2508.12511*, 2025. URL https://arxiv.org/abs/2508.12511.

Gerard Brunick and Steven Shreve. Mimicking an itô process by a solution of a stochastic differential equation. *The Annals of Applied Probability*, 23(4):1584–1628, 2013. doi: 10.1214/12-AAP881. URL https://doi.org/10.1214/12-AAP881.

Jiahang Cao, Qiang Zhang, Hanzhong Guo, Jiaxu Wang, Hao Cheng, and Renjing Xu. Modality-composable diffusion policy via inference-time distribution-level composition, 2025. URL https://arxiv.org/abs/2503.12466.

Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of the 1st International Conference on Image Processing*, volume 2, pp. 168–172. IEEE, 1994. doi: 10.1109/ICIP.1994.413553. URL https://doi.org/10.1109/ICIP.1994.413553.

Hyungjin Chung, Jeongsol Kim, Michael T. Mccann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022. URL https://arxiv.org/abs/2209.14687.

Hyungjin Chung, Jeongsol Kim, Michael T. Mccann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OnD9zGAGT0k.

Alexander Denker, Francisco Vargas, Shreyas Padhy, Kieran Didi, Simon Mathis, Vincent Dutordoir, Riccardo Barbano, Emile Mathieu, Urszula J. Komorowska, and Pietro Lio. DEFT: Efficient fine-tuning of diffusion models by learning the generalised h-transform. In *Advances in Neural Information Processing Systems*, volume 37, pp. 19636–19682. Curran Associates, Inc., 2024. doi: 10.52202/079017-0620. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/22d258dfbdf840ccbf266bbc545dd95f-Abstract-Conference.html.

Carles Domingo-Enrich. A taxonomy of loss functions for stochastic optimal control. *arXiv preprint arXiv:2410.00345*, 2024. URL https://arxiv.org/abs/2410.00345.

Carles Domingo-Enrich, Jiequn Han, Brandon Amos, Joan Bruna, and Ricky T. Q. Chen. Stochastic optimal control matching. In *Advances in Neural Information Processing Systems*, volume 37, pp. 112459–112504. Curran Associates, Inc., 2024. doi: 10.52202/079017-3573. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/cc32ec39a5073f61d38c338d963df30d-Abstract-Conference.html.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xQBRrtQM8u.

Yilun Du and Leslie Pack Kaelbling. Position: Compositional generative modeling: A single model is not all you need. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11721–11732. PMLR, 2024. URL https://proceedings.mlr.press/v235/du24d.html.

Yilun Du, Conor Durkan, Robin Strudel, Joshua B. Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and MCMC. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8489–8510. PMLR, 2023. URL https://proceedings.mlr.press/v202/du23a.html.

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. doi: 10.1198/jasa.2011.tm11181. URL https://doi.org/10.1198/jasa.2011.tm11181.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020. URL https://arxiv.org/abs/2004.07219.

Ulf Grenander. *Lectures in Pattern Theory: Volume 2: Pattern Analysis*, volume 24 of *Applied Mathematical Sciences*. Springer, 1978. doi: 10.1007/978-1-4684-9354-2. URL https://doi.org/10.1007/978-1-4684-9354-2.

Ulrich G. Haussmann and Etienne Pardoux. Time reversal of diffusions. *The Annals of Probability*, 14(4):1188–1205, 1986. doi: 10.1214/aop/1176992362. URL https://doi.org/10.1214/aop/1176992362.

Akio Hayakawa, Masato Ishii, Takashi Shibuya, and Yuki Mitsufuji. MMDisCo: Multi-modal discriminator-guided cooperative diffusion for joint audio and video generation. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VZ1DqfYqUw.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019. URL https://arxiv.org/abs/1912.12180.

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

Peter J. Huber. Robust estimation of a location parameter. In *Breakthroughs in Statistics: Methodology and Distribution*, pp. 492–518. Springer, 1992. doi: 10.1007/978-1-4612-4380-9_35. URL https://doi.org/10.1007/978-1-4612-4380-9_35.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

Thomas G. Kurtz. Martingale problems for conditional distributions of markov processes. *Electronic Journal of Probability*, 3:1–29, 1998. doi: 10.1214/EJP.v3-31. URL https://doi.org/10.1214/EJP.v3-31.

Chieh-Hsin Lai, Yang Song, Dongjun Kim, Yuki Mitsufuji, and Stefano Ermon. The principles of diffusion models. *arXiv preprint arXiv:2510.21890*, 2025. URL https://arxiv.org/abs/2510.21890.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. Syncdiffusion: Coherent montage via synchronized joint diffusions. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/9ee3a664ccfeabc0da16ac6f1f1cfe59-Abstract-Conference.html.

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models. In *European Conference on Computer Vision*, pp. 423–439. Springer, 2022. URL https://www.ecva.net/papers/eccv_2022/papers_ECCV/html/6940_ECCV_2022_paper.php.

Haofei Lu, Dongqi Han, Yifei Shen, and Dongsheng Li. What makes a good diffusion planner for decision making? In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=7BQkXXM8Fy.

Hao Luan, Yi Xian Goh, See-Kiong Ng, and Chun Kai Ling. Projected coupled diffusion for test-time constrained joint generation. *arXiv preprint arXiv:2508.10531*, 2025. URL https://arxiv.org/abs/2508.10531.

Yunhao Luo, Chen Sun, Joshua B. Tenenbaum, and Yilun Du. Potential based diffusion motion planning. *arXiv preprint arXiv:2407.06169*, 2024. URL https://arxiv.org/abs/2407.06169.

Luigi Negro. Sample distribution theory using coarea formula. *Communications in Statistics – Theory and Methods*, 53(5):1864–1889, 2024. URL https://doi.org/10.1080/03610926.2022.2112507.

Edward Nelson. *Dynamical Theories of Brownian Motion*. Princeton University Press, 1967. doi: 10.1515/9780691219615. URL https://doi.org/10.1515/9780691219615.

Nikolas Nüsken and Lorenz Richter. Solving high-dimensional hamilton–jacobi–bellman pdes using neural networks: Perspectives from the theory of controlled diffusions and measures on path space. *Partial Differential Equations and Applications*, 2(4):48, 2021. doi: 10.1007/s42985-021-00102-x. URL https://doi.org/10.1007/s42985-021-00102-x.

Bernt Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Springer, 6 edition, 2003. doi: 10.1007/978-3-642-14394-6. URL https://doi.org/10.1007/978-3-642-14394-6.

Lasse Peters, Laura Ferranti, Andrea Bajcsy, and Javier Alonso-Mora. Coordinated diffusion: Generating multi-agent behavior without multi-agent demonstrations. *arXiv preprint arXiv:2605.11485*, 2026. URL https://arxiv.org/abs/2605.11485.

Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, 2 edition, 2004. doi: 10.1007/978-1-4757-4145-2. URL https://doi.org/10.1007/978-1-4757-4145-2.

Yorai Shaoul, Itamar Mishani, Shivam Vats, Jiaoyang Li, and Maxim Likhachev. Multi-robot motion planning with diffusion models. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AUCYptvAf3.

Guni Sharon, Roni Stern, Ariel Felner, and Nathan R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015. doi: 10.1016/j.artint.2014.11.006. URL https://doi.org/10.1016/j.artint.2014.11.006.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen Mckeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55810–55827. PMLR, 2025. URL https://proceedings.mlr.press/v267/singhal25b.html.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-Or, and Amit Haim Bermano. Human motion diffusion model. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SJ1kSyO2jwu.

Masatoshi Uehara, Yulai Zhao, Kevin Black, Ehsan Hajiramezanali, Gabriele Scalia, Nathaniel Lee Diamant, Alex M. Tseng, Tommaso Biancalani, and Sergey Levine. Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024. URL https://arxiv.org/abs/2402.15194.

Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, Alexandre Adam, Jarrid Rector-Brooks, Yoshua Bengio, Glen Berseth, and Nikolay Malkin. Amortizing intractable inference in diffusion models for vision, language, and control. In *Advances in Neural Information Processing Systems*, volume 37, pp. 76080–76114. Curran Associates, Inc., 2024. doi: 10.52202/079017-2422. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/8b21a7ea42cbcd1c29a7a88c444cce45-Abstract-Conference.html.

Lirui Wang, Jialiang Zhao, Yilun Du, Edward H. Adelson, and Russ Tedrake. PoCo: Policy composition from and for heterogeneous robot learning. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.127. URL https://www.roboticsproceedings.org/rss20/p127.html.

Christopher K. I. Williams. Structured generative models for scene understanding. *International Journal of Computer Vision*, 133:2845–2867, 2025. doi: 10.1007/s11263-024-02316-z. URL https://doi.org/10.1007/s11263-024-02316-z.

Zhutian Yang, Jiayuan Mao, Yilun Du, Jiajun Wu, Joshua B. Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Compositional diffusion-based continuous constraint solvers. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3242–3265. PMLR, 06–09 Nov 2023.

Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. MADiff: Offline multi-agent learning with diffusion models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0136. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/07e278a120830b10aae20cc600a8c07b-Abstract-Conference.html.

Appendix Overview

A Related Work **15**

B Diffusion and Path-Space Formulation of CMDS **17**

B.1 Forward and Reverse Diffusion Processes 18

B.2 Controlled Reverse Diffusions and the Assembled Process 19

B.3 Path-Space Change of Measure and KL Cost 20

B.4 Optimal Coordinated Path and Terminal Laws 21

B.5 Pushforward to the Assembled Process 23

C Adjoint Matching Optimisation **24**

C.1 Full Pathwise Adjoint 25

C.2 Adjoint Matching Instantiation of CMDS 25

D Experiments **31**

D.1 Two-Agent OU Coordination: A Stochastic-Control Sanity Check 31

D.2 Relational Visual Assembly 34

D.3 Source Separation 39

D.4 PointMaze Collision Coordination 42

D.5 Multi-Arm KUKA Coordination from Frozen Single-Arm Priors 51

D.6 Goal-Directed Multi-Person Motion Coordination 58

A Related Work

Modular and compositional generative modelling. Modular generative modelling has a long history in pattern theory and structured scene representations (Grenander, 1978; Bienenstock et al., 1996; Williams, 2025), and recent work advocates composing specialised generators rather than learning every combination within a single model (Du & Kaelbling, 2024). For diffusion models, this perspective has motivated compositional visual generation through combinations of conditional scores (Liu et al., 2022) and sampling methods for composed energy-based models (Du et al., 2023). Related approaches compose learned constraint factors (Yang et al., 2023) or diffusion policies across skills, domains, and input modalities (Wang et al., 2024; Cao et al., 2025). CMDS follows this modular perspective while explicitly distinguishing component states from their assembled output. Each frozen generator supplies a component, and an assembly-level reward specifies the interactions introduced through learned joint control.

Joint generation with pretrained diffusion models. A related line of work coordinates multiple pretrained diffusion processes directly during generation. MultiDiffusion binds multiple denoising paths through shared image-generation constraints (Bar-Tal et al., 2023), while SyncDiffusion synchronises image patches using perceptual-similarity gradients (Lee et al., 2023). MMDisCo coordinates frozen single-modal diffusion models using a discriminator trained on paired audio–video examples (Hayakawa et al., 2025). In contrast to paired-data supervision, CMDS learns coordination from an assembly-level reward under a path-space control objective, without requiring training examples of valid joint outputs.

Cost-based coordination of diffusion processes. Coordinated Diffusion (CoDi; Peters et al., 2026) and Projected Coupled Diffusion (PCD; Luan et al., 2025) are particularly close to our setting. CoDi constructs a KL-regularised cooperative objective relative to single-agent base policies and

guides their diffusion processes simultaneously. Its sampling-based guidance estimator uses online cost evaluations without requiring cost gradients. PCD likewise introduces interactions between pretrained diffusion processes at test time, combining cost-based guidance with projections onto prescribed constraint sets. CMDS shares the use of separately pretrained generators and a joint criterion, but learns a reusable control across task instances under a quadratic path-action penalty. Coordination is therefore amortised into a joint-state control, rather than reconstructed solely through inference-time guidance. The approaches also differ in their constraint treatment: PCD explicitly projects onto supplied constraint sets, whereas the finite reward penalties used in our experiments do not guarantee hard constraint satisfaction. Our current adjoint-based implementation additionally assumes differentiable rewards, unlike CoDi's cost estimator. Since CoDi and PCD act at inference time, they are complementary to amortised coordination: their guidance or projection mechanisms could in principle be applied on top of a learned CMDS control when additional instance-specific constraints must be imposed at sampling time.

Diffusion-based multi-agent planning. Diffuser formulates planning as guided generation of trajectories (Janner et al., 2022). Multi-agent extensions such as MADiff learn coordinated behaviour from offline multi-agent trajectories (Zhu et al., 2024). Closer to our data assumptions, Multi-robot Multi-model planning Diffusion (MMD) combines single-agent diffusion priors with constraint-based search and replanning (Shaoul et al., 2025). Our motion experiments similarly begin from frozen component priors, but learn coordination separately from component generation. In PointMaze, we coordinate copies of a single-agent trajectory prior across start–goal instances; in KUKA, we coordinate released single-arm diffusion priors for multi-arm planning; and in the MDM experiment, we coordinate copies of a released text-conditioned single-person motion prior to generate collision-aware multi-person motion. Across these settings, the component models are not trained on the corresponding multi-agent task, while coordination is amortised into a reusable joint control rather than resolved solely through instance-specific search or guidance.

Inference-time steering and compositional posterior inference. Diffusion posterior sampling (DPS) conditions pretrained diffusion models on observations through approximate likelihood guidance evaluated using denoised estimates (Chung et al., 2023). FK-Steering instead uses reward-dependent particle reweighting and resampling during generation (Singhal et al., 2025); its particles represent alternative complete candidates, rather than the components of one assembled output. Our source-separation experiment places the observation model after an additive assembly and performs inference over the constituent sources. We compare against DPS adapted to the same product prior, while learning observation-dependent joint feedback under a path-action penalty. Amortisation is compatible with additional test-time information: our Tweedie reward-gradient parameterisation retains explicit reward-gradient features within the learned control. The test-time coordination mechanisms of PCD (Luan et al., 2025) and CoDi (Peters et al., 2026) are closely related to the *cooperative DPS* (CDPS) baseline introduced in our earlier work (Barbano et al., 2026). CDPS replaces a learned cooperative control with per-step guidance obtained from a joint cost evaluated on the agents' Tweedie estimates. PCD explicitly derives an analogous posterior-sampling variant, in which a coupling cost evaluated on Tweedie-denoised component estimates is differentiated through each diffusion process. CoDi follows the same general principle of augmenting the product of single-agent diffusion scores with an online joint cost-guidance term, but estimates this correction gradient-free. These methods therefore provide non-amortised, inference-time alternatives to CMDS, whereas CMDS learns a reusable joint feedback control across task instances.

Stochastic control and amortised diffusion steering. Optimal-control perspectives connect diffusion generation and reward-based fine-tuning to controlled dynamics and relative-entropy regularisation (Berner et al., 2024; Uehara et al., 2024). Stochastic optimal control matching and adjoint matching develop regression-based approaches for learning such controls (Domingo-Enrich et al., 2024; 2025). DEFT learns a generalised h-transform correction while keeping the pretrained diffusion model frozen (Denker et al., 2024), and relative trajectory balance amortises posterior sampling through a path-space balance objective (Venkatraman et al., 2024). Building on these foundations, CMDS coordinates a reference product process through an assembly-level reward and joint-state feedback, with the component generators remaining frozen. Adjoint matching is one learning strategy within this framework; our discrete implementation uses the full controlled-transition adjoint, rather than the lean-adjoint variant of Domingo-Enrich et al. (2025).

B DIFFUSION AND PATH-SPACE FORMULATION OF CMDS

Key Takeaways: The pretrained models define an independent reference process. The N component diffusion models form a product reverse process, and because the reference density factorises across agents, its reverse drift acts agent-wise. Cross-agent dependence is introduced only by the coordination control (Appendix B.1). **The assembled process has well-defined marginal dynamics, but itself need not be Markov.** CMDS controls the full component state $\hat{X}_s$, while the assembled state $\hat{Y}_s = \varphi(\hat{X}_s)$ is obtained through the assembly map. Its fixed-time law is the pushforward of the joint-component-state law, and its instantaneous marginals satisfy an exact projected Fokker-Planck equation, but $\hat{Y}$ need not be Markovian when φ discards information about dynamics (Appendix B.2). **Coordination is a minimum-KL change of controls in path measure.** The quadratic control action equals the relative entropy from the independent pretrained product process eq. (35), so CMDS trades assembly reward against the smallest path-space deviation from the component priors (Appendix B.3). **The optimal law is reward tilting.** The SOC optimum satisfies

$$\frac{\mathrm{d}\hat{\mathbb{P}}^\star}{\mathrm{d}\hat{\mathbb{P}}}(\hat{X}_\cdot) = \frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)},$$

where the $\hat{X}_0$-dependent normaliser preserves the prescribed initial marginal. In the complete-noising limit, $\hat{X}_1$ is independent of $\hat{X}_0$, and this reduces to the terminal reward tilt in eq. (42). The reward-tilting formulation holds for general assembly-level rewards. In the particularly relevant and interesting case in which $r \circ \varphi$ is non-separable across components, the exponential tilt generally no longer factorises, so the resulting target law introduces intended dependencies between components that have no intended dependencies under the reference product law (Appendix B.4). **The change of measure can be transferred to the assembled process.** If φ is injective, the assembled path-density ratio can be written directly through φ^{-1}. If φ is non-injective, the assembled density ratio is instead obtained by conditionally averaging the component-space likelihood ratio over all trajectories mapping to the same assembly (Appendix B.5).

We use the notation summarised in table 5. Particularly, $s \in [0,1]$ denotes reverse, generative time; hatted variables denote reverse-time quantities; $i \in \{1,\dots,N\}$ is an agent index; and $\hat{X}_s = (\hat{X}_s^1,\dots,\hat{X}_s^N) \in \mathbb{R}^D$, with $D = Nd$. The fixed assembly map $\varphi : \mathbb{R}^D \to \mathbb{R}^m$ differs from experiment to experiment, and $\hat{Y}_s = \varphi(\hat{X}_s)$. The system-level reward is r and $\lambda > 0$ is its strength.

Quantity	Notation
Variables and processes	
Abstract component & assembly	$X^i,\ X,\ Y = \varphi(X)$
Uncontrolled reverse process	$\hat{X}_s^i,\ \hat{X}_s,\ \hat{Y}_s$
Generic controlled reverse process	$\hat{X}_s^{u,i},\ \hat{X}_s^u,\ \hat{Y}_s^u$
Parametrised controlled reverse process	$\hat{X}_s^{\theta,i},\ \hat{X}_s^\theta,\ \hat{Y}_s^\theta$
Stop-gradient controlled rollout	$\hat{X}_s^{\bar{\theta},i},\ \hat{X}_s^{\bar{\theta}},\ \hat{Y}_s^{\bar{\theta}}$
Forward-time OU experiment	$X_t^i,\ X_t^{\theta,i}$
Densities and distributions	
Initial reverse-time noise density	$\hat{p}_0$
Initial forward-time data density	p_0
Terminal density of the uncontrolled reverse process	$\hat{p}_1$
Terminal forward-time noise density	p_1
Exact target law of a generic component	π
Learned terminal law of a generic component	$\tilde{\pi}$
Joint terminal law of the independent pretrained samplers	$\rho = \tilde{\pi}^1 \otimes \cdots \otimes \tilde{\pi}^N$

Table 5: **Notation convention.** Hatted variables denote reverse-time quantities. The superscript u denotes a generic control, while θ denotes the process induced by the parametrised control U_θ. For the continuous-time process notation, quantities without a control superscript refer to the uncontrolled reference process, unless stated otherwise. In the discrete-time algorithms and implementation descriptions, X_ℓ denotes the rollout state, with the control superscript suppressed when the generating method is clear from the context.

Standing assumptions. Throughout the Appendix, we work under the following convenient sufficient assumptions. For each $i \in \{1, \dots, N\}$, the drift $(t, x^i) \mapsto f_t^i(x^i)$ is jointly measurable, locally Lipschitz in x^i, uniformly on compact time intervals, and has at most linear growth. The diffusion scale $\sigma : [0,1] \to [0,\infty)$ is continuous and satisfies $\sigma_t > 0$ for $t \in (0,1)$. The joint forward process admits strictly positive one-time densities p_t, and the map $p : (0,1) \times \mathbb{R}^D \to (0,\infty)$, defined by $p(t,x) := p_t(x)$, belongs to $C^{1,2}((0,1) \times \mathbb{R}^D)$ and satisfies the integrability conditions required for the time-reversal identities below.

B.1 Forward and Reverse Diffusion Processes

For clarity, we present the derivation for N component processes, each taking values in $\mathbb{R}^d$ and driven by a common scalar diffusion schedule, so that the full component state is their concatenation,

$$X_t = (X_t^1, \dots, X_t^N) \in \mathbb{R}^D, \qquad D = Nd. \tag{10}$$

Under the standing assumptions stated above, we define $\{X_t\}_{t\in[0,1]}$ by the time-forward SDE in a general form

$$\mathrm{d}X_t = f_t(X_t)\,\mathrm{d}t + \sigma_t\,\mathrm{d}W_t, \qquad W_t \text{ is } D\text{-dimensional}, \quad X_t \in \mathbb{R}^D. \tag{11}$$

X_t is the expanded process corresponding to the concatenation of the N agents, with $X_t^i \in \mathbb{R}^d$. Here $W_t = ((W_t^1)^\top, \dots, (W_t^N)^\top)^\top$, where $W^1, \dots, W^N$ are independent d-dimensional Wiener processes, and the initial component states are sampled independently. We have $D = N \times d$ and

$$X_t = \begin{bmatrix} X_t^1 \\ \vdots \\ X_t^N \end{bmatrix}, \qquad f_t(X_t) = \begin{bmatrix} f_t^1(X_t^1) \\ \vdots \\ f_t^N(X_t^N) \end{bmatrix}. \tag{12}$$

Consequently, the reference process factorises across agents, and its one-time density satisfies $p_t(x) = \prod_{i=1}^N p_t^i(x^i)$. By the Fokker–Planck equation, the corresponding forward density p_t satisfies

$$\partial_t p_t(x) = -\nabla_x \cdot \big(f_t(x)\, p_t(x)\big) + \frac{1}{2}\,\sigma_t^2\,\Delta_x p_t(x). \tag{13}$$

Under the standard regularity assumptions ensuring time reversal (Nelson, 1967; Anderson, 1982; Haussmann & Pardoux, 1986), define

$$\hat{X}_s := X_{1-s}, \qquad \hat{p}_s := p_{1-s}, \qquad \hat{\sigma}_s := \sigma_{1-s} \qquad s \in [0,1]. \tag{14}$$

Then $\hat{X}_s$ solves the reverse-time SDE

$$\mathrm{d}\hat{X}_s = \hat{f}_s(\hat{X}_s)\,\mathrm{d}s + \hat{\sigma}_s\,\mathrm{d}\bar{W}_s, \qquad \hat{f}_s(x) := -f_{1-s}(x) + \hat{\sigma}_s^2\,\nabla_x \log p_{1-s}(x), \tag{15}$$

where $\bar{W}_s$ is a standard D-dimensional Brownian motion with respect to the reverse-time filtration. Since the reference density factorises across agents, the reverse drift also acts component-wise,

$$\hat{f}_s(x) = \begin{bmatrix} \hat{f}_s^1(x^1) \\ \vdots \\ \hat{f}_s^N(x^N) \end{bmatrix}, \qquad \hat{f}_s^i(x^i) = -f_{1-s}^i(x^i) + \hat{\sigma}_s^2 \nabla_{x^i} \log p_{1-s}^i(x^i).$$

Its density satisfies

$$\partial_s \hat{p}_s(x) = -\nabla_x \cdot \big(\hat{f}_s(x)\,\hat{p}_s(x)\big) + \frac{1}{2}\,\hat{\sigma}_s^2\,\Delta_x \hat{p}_s(x). \tag{16}$$

Let $\varphi : \mathbb{R}^D \to \mathbb{R}^m$ be the fixed assembly map. We assume that $\varphi \in C^2(\mathbb{R}^D; \mathbb{R}^m)$ and that its first and second derivatives are bounded. This condition includes the linear masking, stacking, and additive assembly maps considered in our experiments. We define

$$\hat{Y}_s = \varphi(\hat{X}_s) \in \mathbb{R}^m. \tag{17}$$

Its marginal law is the push-forward measure

$$\hat{\mu}_s^Y := \varphi_{\#}\big(\hat{p}_s(x)\,\mathrm{d}x\big). \tag{18}$$

Whenever $\hat{\mu}_s^Y$ is absolutely continuous with respect to Lebesgue measure, we denote its density by $\hat{p}_s^Y$.

B.2 Controlled Reverse Diffusions and the Assembled Process

We control the joint component process, and hence the assembled process, through a normalised drift perturbation. Let

$$u : [0,1] \times \mathbb{R}^D \to \mathbb{R}^D, \qquad (s,x) \mapsto u_s(x),$$

be an admissible Markov control such that the controlled SDE below admits a weak solution and satisfies the integrability conditions. We take the controlled and reference processes to share the same initial law, $\hat{X}_0^u \sim \nu_0$. Then the controlled reverse diffusion is

$$\mathrm{d}\hat{X}_s^u = \Big[\hat{f}_s(\hat{X}_s^u) + \hat{\sigma}_s u_s\big(\hat{X}_s^u\big)\Big]\,\mathrm{d}s + \hat{\sigma}_s\,\mathrm{d}\bar{W}_s, \tag{19}$$

where $\bar{W}$ is a standard D-dimensional Brownian motion under the controlled law $\hat{\mathbb{P}}^u$. When the law of $\hat{X}_s^u$ admits a density, denoted $\hat{p}_s^u$, this density satisfies

$$\partial_s \hat{p}_s^u(x) = -\nabla_x \cdot \Big(\big[\hat{f}_s(x) + \hat{\sigma}_s u_s(x)\big]\hat{p}_s^u(x)\Big) + \frac{1}{2}\,\hat{\sigma}_s^2\,\Delta_x \hat{p}_s^u(x). \tag{20}$$

In practice, the drift perturbation may be parameterised using assembled features, for example as $U_\theta(x,s) = \tilde{U}_\theta(x,\varphi(x),s)$, while remaining a full-state feedback map from $\mathbb{R}^D$ to $\mathbb{R}^D$.

Applying Itô's formula to $\hat{Y}_s^u := \varphi(\hat{X}_s^u)$ gives

$$\mathrm{d}\hat{Y}_s^u = \Big[D_x\varphi(\hat{X}_s^u)\,\hat{f}_s(\hat{X}_s^u) + \frac{1}{2}\,\hat{\sigma}_s^2\,\Delta_x\varphi(\hat{X}_s^u) + \hat{\sigma}_s D_x\varphi(\hat{X}_s^u)\,u_s(\hat{X}_s^u)\Big]\,\mathrm{d}s \tag{21}$$

$$+\,\hat{\sigma}_s\,D_x\varphi(\hat{X}_s^u)\,\mathrm{d}\bar{W}_s. \tag{22}$$

Here, $\Delta_x\varphi := (\Delta_x\varphi_1, \ldots, \Delta_x\varphi_m)$ is understood component-wise. For the linear masking, stacking, and additive assembly maps used in our experiments, $\Delta_x\varphi = 0$.

Equivalently, introducing

$$\hat{b}_s^\varphi(x) := D_x\varphi(x)\,\hat{f}_s(x) + \frac{1}{2}\,\hat{\sigma}_s^2\,\Delta_x\varphi(x), \tag{23}$$

we may write

$$\mathrm{d}\hat{Y}_s^u = \Big[\hat{b}_s^\varphi(\hat{X}_s^u) + \hat{\sigma}_s D_x\varphi(\hat{X}_s^u)\,u_s(\hat{X}_s^u)\Big]\mathrm{d}s + \hat{\sigma}_s\,D_x\varphi(\hat{X}_s^u)\,\mathrm{d}\bar{W}_s. \tag{24}$$

We can also recover the evolution of marginals. For $\hat{\mu}_s^{u,Y}$-almost every y, define the conditional projected coefficients,

$$\overline{b}_s^u(y) := \mathbb{E}_{\hat{\mathbb{P}}^u}\Big[\hat{b}_s^\varphi(\hat{X}_s^u) + \hat{\sigma}_s D_x\varphi(\hat{X}_s^u)u_s(\hat{X}_s^u)\,\Big|\,\hat{Y}_s^u = y\Big], \tag{25}$$

$$\overline{Q}_s^u(y) := \mathbb{E}_{\hat{\mathbb{P}}^u}\Big[\hat{\sigma}_s^2 D_x\varphi(\hat{X}_s^u)D_x\varphi(\hat{X}_s^u)^\top\,\Big|\,\hat{Y}_s^u = y\Big]. \tag{26}$$

Then, for test functions $\psi \in C_c^2(\mathbb{R}^m)$, the one-time marginals $\hat{\mu}_s^{u,Y} = \mathrm{Law}(\hat{Y}_s^u)$ satisfy

$$\frac{\mathrm{d}}{\mathrm{d}s}\int \psi(y)\,\hat{\mu}_s^{u,Y}(\mathrm{d}y) = \int\Big[\nabla_y\psi(y)\cdot\overline{b}_s^u(y) + \frac{1}{2}\,\mathrm{Tr}\Big(\overline{Q}_s^u(y)D_y^2\psi(y)\Big)\Big]\,\hat{\mu}_s^{u,Y}(\mathrm{d}y), \tag{27}$$

where $D_y^2\psi$ denotes the Hessian of ψ. If $\hat{\mu}_s^{u,Y}(\mathrm{d}y) = \hat{p}_s^{u,Y}(y)\mathrm{d}y$, then $\hat{p}_s^{u,Y}$ satisfies, in the sense of distributions,

$$\partial_s\hat{p}_s^{u,Y}(y) = -\nabla_y\cdot\big(\overline{b}_s^u(y)\hat{p}_s^{u,Y}(y)\big) + \frac{1}{2}\sum_{j,k=1}^m \partial_{y_j}\partial_{y_k}\Big((\overline{Q}_s^u(y))_{jk}\hat{p}_s^{u,Y}(y)\Big). \tag{28}$$

Remark 2 (Projected marginal dynamics and the law of the assembly). The eq. (28) characterises only the one-time marginals of $\hat{Y}^u$, and it should be distinguished from the standard Fokker-Planck equation of a Markov diffusion on the assembled space. Even though $\hat{X}_s^u$ is

Markovian, the assembled process $\hat{Y}_s^u = \varphi(\hat{X}_s^u)$ need not be Markovian if φ drops information about dynamics. The coefficients $\bar{b}_s^u$ and $\bar{Q}_s^u$ defined above average over unresolved component states consistent with the same assembly state y. Under the conditions of Brunick & Shreve (2013, Corollary 3.7), there exists a weak solution of an SDE with drift $\bar{b}_s^u$ and covariance $\bar{Q}_s^u$ whose one-time marginals coincide with those of $\hat{Y}^u$. If, moreover, the controlled projected drift $\hat{b}_s^\varphi(x) + \hat{\sigma}_s D_x\varphi(x) u_s(x)$ and the projected covariance $\hat{\sigma}_s^2 D_x\varphi(x) D_x\varphi(x)^\top$ depend on x only through $y = \varphi(x)$, and the associated martingale problem on the assembled space is well posed, then $\hat{Y}^u$ is Markov (Kurtz, 1998, Corollary 3.5 and Remark 3.6). At each fixed s, independently of this Markov question, the law of the assembled state is simply the pushforward

$$\hat{\mu}_s^{u,Y} = \varphi_\#(\hat{p}_s^u(x)\mathrm{d}x). \tag{29}$$

For $m = D$, if φ is injective and $\det D\varphi(x) \neq 0$ for almost every $x \in \mathbb{R}^D$ in Lebesgue measure, the law of $\hat{Y}_s^u$ admits a density given by

$$\hat{p}_s^{u,Y}(y) = \frac{\hat{p}_s^u(\varphi^{-1}(y))}{|\det D\varphi(\varphi^{-1}(y))|}, \tag{30}$$

for almost every $y \in \varphi(\mathbb{R}^D)$ with zero density outside $\varphi(\mathbb{R}^D)$. More generally, for $m \leq D$, if the rank of $D\varphi$ equals m for almost every x, an explicit expression for the density of $\hat{Y}$ can be given by a generalised coarea formula (Negro, 2024) under appropriate conditions but without injectivity. Without these assumptions, eq. (29) still holds, but an explicit expression of the density (with respect to Lebesgue measure) may not exist.

B.3 Path-Space Change of Measure and KL Cost

Let $\hat{\mathbb{P}} := \hat{\mathbb{P}}^{u=0}$ and $\hat{\mathbb{P}}^u$ denote the path-space laws on $C([0,1], \mathbb{R}^D)$ induced by the reference and controlled reverse diffusions, respectively. Throughout this subsection, $\hat{X}$ denotes the canonical coordinate process on path space. For completeness, we allow the controlled and reference processes to have potentially different initial marginals, although CMDS uses the common-initial-law setting. Let

$$e_0 : C([0,1], \mathbb{R}^D) \longrightarrow \mathbb{R}^D, \qquad e_0(\omega) := \omega_0, \tag{31}$$

and define

$$\nu_0 := (e_0)_\# \hat{\mathbb{P}}, \qquad \nu_0^u := (e_0)_\# \hat{\mathbb{P}}^u.$$

Assume that $\nu_0^u \ll \nu_0$, that $D_{\mathrm{KL}}(\nu_0^u \| \nu_0) < \infty$, and that Novikov's condition holds under the reference law:

$$\mathbb{E}_{\hat{\mathbb{P}}}\left[\exp\left(\frac{1}{2}\int_0^1 \left\|u_s(\hat{X}_s)\right\|^2 \mathrm{d}s\right)\right] < \infty. \tag{32}$$

Girsanov's theorem (Øksendal, 2003) then gives

$$\frac{\mathrm{d}\hat{\mathbb{P}}^u}{\mathrm{d}\hat{\mathbb{P}}}(\hat{X}_\cdot) = \frac{\mathrm{d}\nu_0^u}{\mathrm{d}\nu_0}(\hat{X}_0) \exp\left(\int_0^1 u_s(\hat{X}_s) \cdot \mathrm{d}\bar{W}_s - \frac{1}{2}\int_0^1 \left\|u_s(\hat{X}_s)\right\|^2 \mathrm{d}s\right), \tag{33}$$

where $\bar{W}$ denotes the Brownian motion under the reference law $\hat{\mathbb{P}}$. In the common case $\nu_0^u = \nu_0$, the initial-density factor disappears.

Taking the expectation of the log-density ratio under the controlled law gives

$$D_{\mathrm{KL}}\big(\hat{\mathbb{P}}^u \,\|\, \hat{\mathbb{P}}\big) = D_{\mathrm{KL}}\big(\nu_0^u \,\|\, \nu_0\big) + \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^u}\left[\int_0^1 \left\|u_s(\hat{X}_s^u)\right\|^2 \mathrm{d}s\right]. \tag{34}$$

In particular, if $\nu_0^u = \nu_0$, then

$$D_{\mathrm{KL}}\big(\hat{\mathbb{P}}^u \,\|\, \hat{\mathbb{P}}\big) = \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^u}\left[\int_0^1 \left\|u_s(\hat{X}_s^u)\right\|^2 \mathrm{d}s\right]. \tag{35}$$

For a neural parameterisation using assembled features, the same identities apply with

$$u_s(x) = U_\theta(x, s) = \tilde{U}_\theta(x, \varphi(x), s),$$

provided that the corresponding controlled process satisfies the admissibility and integrability conditions above.

Finally, define the pathwise assembly map

$$\Phi : C([0,1], \mathbb{R}^D) \longrightarrow C([0,1], \mathbb{R}^m), \qquad \big(\Phi(\omega)\big)_s := \varphi(\omega_s), \tag{36}$$

and the induced assembled path laws

$$\hat{\Pi} := \Phi_\# \hat{\mathbb{P}}, \qquad \hat{\Pi}^u := \Phi_\# \hat{\mathbb{P}}^u. \tag{37}$$

Since $\hat{\mathbb{P}}^u \ll \hat{\mathbb{P}}$, the data-processing inequality gives

$$D_{\mathrm{KL}}\big(\hat{\Pi}^u \,\|\, \hat{\Pi}\big) \le D_{\mathrm{KL}}\big(\hat{\mathbb{P}}^u \,\|\, \hat{\mathbb{P}}\big). \tag{38}$$

Thus, the relative-entropy cost of controlling the assembled paths is bounded above by the corresponding cost on the full component path space.

B.4 Optimal Coordinated Path and Terminal Laws

The idea of the proof of the result in this section mainly follows (Domingo-Enrich et al., 2025, Section 4). Recall that $\hat{\mathbb{P}}$ and $\hat{\mathbb{P}}^{u^\star}$ denote the uncontrolled reference path measure and the optimal controlled path measure, respectively. In addition, we define an evaluation map e_1 at time 1,

$$e_1 : C([0,1], \mathbb{R}^D) \to \mathbb{R}^D, \qquad e_1(\omega) = \omega_1. \tag{39}$$

The marginal distributions at time 1 are then defined as

$$\rho^\star := (e_1)_\# \hat{\mathbb{P}}^{u^\star} \qquad \rho := (e_1)_\# \hat{\mathbb{P}}. \tag{40}$$

Based on these, we can formulate the result mentioned in section 2.2 as the following result.

Proposition: The optimal and reference distributions of the full path $\hat{X}$ satisfy

$$\hat{\mathbb{P}}^{u^\star}(\mathrm{d}\hat{X}) = \hat{\mathbb{P}}(\mathrm{d}\hat{X}) \frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}, \tag{41}$$

where $Z(\hat{X}_0)$ is a constant only depending on $\hat{X}_0$.

Furthermore, if $\hat{X}_1$ is independent of $\hat{X}_0$, then their marginal distributions at time 1 satisfy

$$\rho^\star(\mathrm{d}\hat{X}_1) \propto \rho(\mathrm{d}\hat{X}_1) \exp\Big(\lambda r(\varphi(\hat{X}_1))\Big) \tag{42}$$

Proof. By the tower property and definition of KL divergence, we can rewrite our SOC objective,

$$D_{\mathrm{KL}}(\hat{\mathbb{P}}^u \| \hat{\mathbb{P}}) - \lambda \mathbb{E}_{\hat{\mathbb{P}}^u}\Big[r(\varphi(\hat{X}_1))\Big] \tag{43}$$

$$= \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u}\left[\log \frac{\mathrm{d}\hat{\mathbb{P}}^u}{\mathrm{d}\hat{\mathbb{P}}}(\hat{X}) - \lambda r(\varphi(\hat{X}_1))\right] \tag{44}$$

$$= \mathbb{E}_{\hat{X}_0\sim\nu_0}\left[\mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}\left(\log \frac{\mathrm{d}\hat{\mathbb{P}}^u}{\mathrm{d}\hat{\mathbb{P}}}(\hat{X}) - \lambda r(\varphi(\hat{X}_1))\right)\Big|\hat{X}_0\right] \tag{45}$$

$$= \mathbb{E}_{\hat{X}_0\sim\nu_0}\left[D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} \| \hat{\mathbb{P}}|_{\hat{X}_0}) - \lambda \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}[r(\varphi(\hat{X}_1))]\right], \tag{46}$$

so the SOC objective can be rewritten as

$$\min_u \left\{\mathbb{E}_{\hat{X}_0\sim\nu_0}\left[D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} \| \hat{\mathbb{P}}|_{\hat{X}_0}) - \lambda \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}[r(\varphi(\hat{X}_1^u))]\right]\right\} \tag{47}$$

Let $\hat{\mathbb{P}}|_{\hat{X}_0}$ and $\hat{\mathbb{P}}^u|_{\hat{X}_0}$ denote path measures of the uncontrolled process in eq. (15) and the controlled processes in eq. (19) starting from the same $\hat{X}_0$ respectively. By the Girsanov theorem, we have

$$\frac{\mathrm{d}\hat{\mathbb{P}}|_{\hat{X}_0}}{\mathrm{d}\hat{\mathbb{P}}^u|_{\hat{X}_0}} = \exp\left(-\int_0^1 u(\hat{X}_s^u, s)\mathrm{d}\bar{W}_s - \frac{1}{2}\int_0^1 \|u(\hat{X}_s^u, s)\|^2 \mathrm{d}s\right) \tag{48}$$

$$\iff \log\left(\frac{\mathrm{d}\hat{\mathbb{P}}|_{\hat{X}_0}}{\mathrm{d}\hat{\mathbb{P}}^u|_{\hat{X}_0}}\right) = -\int_0^1 u(\hat{X}_s^u, s)\mathrm{d}\bar{W}_s - \frac{1}{2}\int_0^1 \|u(\hat{X}_s^u, s)\|^2 \mathrm{d}s, \tag{49}$$

then we get

$$D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} \| \hat{\mathbb{P}}|_{\hat{X}_0}) = \mathbb{E}_{\hat{X}^u \sim \hat{\mathbb{P}}^u|_{\hat{X}_0}}\left[\frac{1}{2}\int_0^1 \|u(\hat{X}_s^u, s)\|^2 ds \middle| \hat{X}_0\right] \tag{50}$$

Define the path measure $\hat{\mathbb{P}}^{u^\star}$ by

$$\frac{\mathrm{d}\hat{\mathbb{P}}^{u^\star}|_{\hat{X}_0}}{\mathrm{d}\hat{\mathbb{P}}|_{\hat{X}_0}} := \frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}, \qquad Z(\hat{X}_0) := \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}|_{\hat{X}_0}}\left[\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)\right], \tag{51}$$

Notice that

$$\begin{aligned} D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} || \hat{\mathbb{P}}^{u^\star}|_{\hat{X}_0}) &= \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}\left[\log\frac{\mathrm{d}\hat{\mathbb{P}}^u|_{\hat{X}_0}}{\mathrm{d}\hat{\mathbb{P}}^{u^\star}|_{\hat{X}_0}}\right] && (52)\\ &= \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}\left[\log\frac{\mathrm{d}\hat{\mathbb{P}}^u|_{\hat{X}_0}}{\mathrm{d}\hat{\mathbb{P}}|_{\hat{X}_0}} - \lambda r(\varphi(\hat{X}_1)) + \log\Big(Z(\hat{X}_0)\Big)\right] && (53)\\ &= D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} || \hat{\mathbb{P}}|_{\hat{X}_0}) - \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}[\lambda r(\varphi(\hat{X}_1))] + \log\Big(Z(\hat{X}_0)\Big) && (54)\end{aligned}$$

Rearranging terms, we get

$$D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} || \hat{\mathbb{P}}|_{\hat{X}_0}) - \mathbb{E}_{\hat{X}\sim\hat{\mathbb{P}}^u|_{\hat{X}_0}}[\lambda r(\varphi(\hat{X}_1^u))] = D_{\mathrm{KL}}(\hat{\mathbb{P}}^u|_{\hat{X}_0} || \hat{\mathbb{P}}^{u^\star}|_{\hat{X}_0}) - \log\Big(Z(\hat{X}_0)\Big). \tag{55}$$

Since relative entropy is non-negative and vanishes if and only if its arguments coincide, the conditional measure-level objective is uniquely minimised when $\hat{\mathbb{P}}^u_{|\hat{X}_0} = \hat{\mathbb{P}}^\star_{|\hat{X}_0}$. Hence, $\hat{\mathbb{P}}^{u^\star}_{|\hat{X}_0}$ is the unique minimiser of the conditional path-space objective.

Equation (51) is equivalent to

$$\hat{\mathbb{P}}^{u^\star}|_{\hat{X}_0}(\mathrm{d}\hat{X}) = \frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}\hat{\mathbb{P}}|_{\hat{X}_0}(\mathrm{d}\hat{X}) \tag{56}$$

Multiplying both sides of the equation by $\nu_0(\hat{X}_0)$, we get

$$\begin{aligned} \hat{\mathbb{P}}^{u^\star}(\mathrm{d}\hat{X}) &= \hat{\mathbb{P}}^{u^\star}|_{\hat{X}_0}(\mathrm{d}\hat{X})\,\nu_0(\mathrm{d}\hat{X}_0)\\ &= \frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}\hat{\mathbb{P}}|_{\hat{X}_0}(\mathrm{d}\hat{X})\,\nu_0(\mathrm{d}\hat{X}_0)\\ &= \frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}\hat{\mathbb{P}}(\mathrm{d}\hat{X}),\end{aligned} \tag{57}$$

which finishes the proof of eq. (41).

Applying the endpoint map

$$e_{0,1} : C([0,1], \mathbb{R}^D) \to \mathbb{R}^D \times \mathbb{R}^D, \qquad e_{0,1}(\omega) := (\omega_0, \omega_1),$$

to eq. (41) yields

$$(e_{0,1})_{\#}\hat{\mathbb{P}}^{u^\star}(\mathrm{d}x_0, \mathrm{d}x_1) = \frac{\exp(\lambda r(\varphi(x_1)))}{Z(x_0)}(e_{0,1})_{\#}\hat{\mathbb{P}}(\mathrm{d}x_0, \mathrm{d}x_1). \tag{58}$$

Applying independence between $\hat{X}_1$ and $\hat{X}_0$, we can split $\hat{\mathbb{P}}(\mathrm{d}\hat{X}_0, \mathrm{d}\hat{X}_1) = \nu_0(\mathrm{d}\hat{X}_0)\rho(\mathrm{d}\hat{X}_1)$, and then evaluate the expectation on $\hat{X}_0$ to get

$$\rho^\star(\mathrm{d}\hat{X}_1) = \mathbb{E}_{\hat{X}_0\sim\nu_0}\left[\frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}\rho(\mathrm{d}\hat{X}_1)\right] \tag{59}$$

$$= \rho(\mathrm{d}\hat{X}_1)\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)\,\mathbb{E}_{\hat{X}_0\sim\nu_0}\left[\frac{1}{Z(\hat{X}_0)}\right] \tag{60}$$

$$\propto \rho(\mathrm{d}\hat{X}_1)\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big), \tag{61}$$

which finishes the proof of eq. (42).

Relation to probabilistic couplings. A coupling of $\tilde{\pi}^1, \ldots, \tilde{\pi}^N$ is a joint law on the product space whose i-th marginal is $\tilde{\pi}^i$. The reference product law $\tilde{\pi}^1 \otimes \cdots \otimes \tilde{\pi}^N$ is their independent coupling. By contrast, the reward tilt above generally changes both the dependence structure and the component marginals, and the optimal terminal law is therefore not, in general, a coupling of the original pretrained laws. We instead use coupled dynamics to describe the system constructed through a joint control whose component-wise actions may depend on the full multi-agent state.

B.5 Pushforward to the Assembled Process

Starting from the optimal path-law characterisation established in Appendix B.4, we now examine the corresponding change of measure after pushing the reference and controlled processes through the assembly map φ. This distinction is particularly relevant when φ is non-injective, since multiple component trajectories may correspond to the same assembled trajectory.

By similar results discussed in Blessing et al. (2025), and Domingo-Enrich et al. (2024), we already have that, for the objective and terminal cost we considered

$$\mathcal{J}(u) := \mathbb{E}_{\hat{\mathbb{P}}^u}\left[\int_0^1 C(\hat{X}_s^u, s) + \frac{1}{2}\|u_s(\hat{X}_s^u)\|^2\,\mathrm{d}s + g(\hat{X}_1^u)\right] \tag{62}$$

$$= \mathbb{E}_{\hat{\mathbb{P}}^u}\left[\int_0^1 \frac{1}{2}\|u_s(\hat{X}_s^u)\|^2\,\mathrm{d}s - \lambda r\Big(\varphi(\hat{X}_1^u)\Big),\right] \tag{63}$$

where C and g denote general running cost and terminal cost, respectively, and we consider zero running cost and take the terminal cost to be negative reward $g = -\lambda r \circ \varphi$ in this paper. The parametrised version of the objective expression is eq. (72). If the function $g(x)$ is regular enough, the Radon–Nikodym derivative of the optimal path measure $\hat{\mathbb{P}}^{u^\star}$ is given as

$$\frac{\mathrm{d}\hat{\mathbb{P}}^{u^\star}}{\mathrm{d}\hat{\mathbb{P}}}(\hat{X}) = \frac{\exp\big(-\mathcal{W}(\hat{X})\big)}{Z(\hat{X}_0)}, \tag{64}$$

where $\hat{\mathbb{P}}$ is the path measure of the uncontrolled process $\hat{X}$, Z is a normalizing constant and

$$\begin{aligned}\mathcal{W}(\hat{X}) &:= \int_0^1 C(\hat{X}_s, s)\,\mathrm{d}s + g(\hat{X}_1)\\ &= -\lambda r(\varphi(\hat{X}_1)),\end{aligned}$$

Equivalently, their densities give

$$p_{\hat{X}^{u^\star}}(x(\cdot)) = \frac{\exp\left(\lambda r(\varphi(x_1))\right)}{Z(x_0)} p_{\hat{X}^{u=0}}(x(\cdot)),$$

where $x(\cdot) = \{x(t)\}_{t\in[0,1]}$ is a whole trajectory rather than a single point. This means that we can use the above relation to generate samples of $\hat{X}^{u^\star}$ and $\hat{X}$, and then apply φ on those generated samples to get samples of $\hat{Y}^{u^\star}$ and $\hat{Y}$ respectively.

If φ is injective, then we further get

$$\begin{aligned}\frac{\mathrm{d}(\varphi_{\#}\hat{\mathbb{P}}^{u^\star})}{\mathrm{d}(\varphi_{\#}\hat{\mathbb{P}})}(\hat{Y}) &= \frac{1}{Z(\varphi^{-1}(\hat{Y}_0))}\exp\Big(\lambda r(\hat{Y}_1)\Big)\exp\left(-\int_0^1 C(\varphi^{-1}(\hat{Y}_s),s)\mathrm{d}s\right)\\ &= \frac{\exp\Big(\lambda r(\hat{Y}_1)\Big)}{Z(\varphi^{-1}(\hat{Y}_0))}\end{aligned}$$

where $\varphi_{\#}\hat{\mathbb{P}}^{u^\star}$ and $\varphi_{\#}\hat{\mathbb{P}}$ denote path measures of optimal control $Y^{u^\star}$ and uncontrolled Y respectively. They are actually just push-forward measures of $\hat{\mathbb{P}}^{u^\star}$ and $\hat{\mathbb{P}}$ under φ respectively.

If φ is not injective, we have

$$\frac{\mathrm{d}(\varphi_{\#}\hat{\mathbb{P}}^{u^\star})}{\mathrm{d}(\varphi_{\#}\hat{\mathbb{P}})}(\hat{Y}) = \mathbb{E}_{\hat{\mathbb{P}}}\left[\frac{\exp\Big(\lambda r(\varphi(\hat{X}_1))\Big)}{Z(\hat{X}_0)}\exp\left(-\int_0^1 C(\hat{X}_s,s)\mathrm{d}s\right)\Big|\varphi(\hat{X})=\hat{Y}\right] \tag{65}$$

$$= \mathbb{E}_{\hat{\mathbb{P}}}\left[\frac{\exp\Big(\lambda r(\hat{Y}_1)\Big)}{Z(\hat{X}_0)}\Big|\varphi(\hat{X})=\hat{Y}\right] \tag{66}$$

$$= \exp\Big(\lambda r(\hat{Y}_1)\Big)\,\mathbb{E}_{\hat{\mathbb{P}}}\left[\frac{1}{Z(\hat{X}_0)}\Big|\varphi(\hat{X})=\hat{Y}\right], \tag{67}$$

where notations without subscripts denote the whole paths or processes.

C Adjoint Matching Optimisation

Key Takeaways: The adjoint turns the global assembly objective into local learning signals. Starting from the terminal reward gradient, it propagates feedback backwards along the reverse trajectory, accounting for the remaining control cost and cross-agent dependencies introduced by the joint controller (Appendix C.1). **We retain the full adjoint, rather than the lean variant.** Backward replay differentiates both the controlled transition and the quadratic control cost, retaining all state dependencies not removed by the task-specific detach conventions (Appendix C.2). **Discrete matching uses noise-normalised controls and next-state adjoints.** The epsilon residual induces $w_{\theta,\ell} = (\gamma_\ell/\varsigma_\ell)\delta\epsilon_{\theta,\ell}$, which is trained using the matching residual $w_{\theta,\ell} + \varsigma_\ell a_{\ell+1}$. Rollout states and adjoint targets are detached during regression (Algorithm 1).

We use reverse, generative time $s \in [0,1]$. In this subsection, $\hat{f}_s$ denotes the reference reverse drift and $\hat{\sigma}_s$ denotes the corresponding reverse-time diffusion scale. Let

$$\hat{X}_s^\theta = (\hat{X}_s^{\theta,1},\ldots,\hat{X}_s^{\theta,N}), \qquad \hat{Y}_s^\theta = \varphi(\hat{X}_s^\theta, s), \tag{68}$$

where $\varphi : \mathbb{R}^D \times [0,1] \to \mathbb{R}^m$ can be a time-dependent assembly function in general cases. The controlled CMDS process is

$$\mathrm{d}\hat{X}_s^\theta = \Big(\hat{f}_s(\hat{X}_s^\theta) + \hat{\sigma}_s U_\theta(\hat{X}_s^\theta, s)\Big)\,\mathrm{d}s + \hat{\sigma}_s\,\mathrm{d}\bar{W}_s. \tag{69}$$

Here,

$$U_\theta(x,s) = \big(U_\theta^1(x,s),\ldots,U_\theta^N(x,s)\big) \in \mathbb{R}^D \tag{70}$$

is the actual full-state normalised control field. The block form of eq. (69) is

$$\mathrm{d}\hat{X}_s^{\theta,i} = \Big[\hat{f}^i(\hat{X}_s^\theta, s) + \hat{\sigma}_s U_\theta^i(\hat{X}_s^\theta, s)\Big]\,\mathrm{d}s + \hat{\sigma}_s\,\mathrm{d}\bar{W}_s^i, \qquad i = 1,\ldots,N. \tag{71}$$

If the reference component diffusions are independent, then $\hat{f}^i(\hat{X}^\theta_s, s) = \hat{f}^i(\hat{X}^{\theta,i}_s, s)$; the derivation below does not require this simplification.

The approximated objective in the neural network is

$$\mathcal{J}(\theta) = \mathbb{E}_{\hat{\mathbb{P}}^\theta}\left[\int_0^1 C(\hat{X}^\theta_s, s) + \frac{1}{2}\|U_\theta(\hat{X}^\theta_s, s)\|^2 \,\mathrm{d}s + g(\hat{X}^\theta_1)\right]. \tag{72}$$

In this work, we consider the relatively simpler case $\varphi(x, s) = \varphi(x)$ which does not change with time, and treat $\varphi(x, s)$ as $\varphi(x)$ in the following derivations.

C.1 Full Pathwise Adjoint

Fix a sampled controlled path $\hat{X}^\theta$. The full adjoint corresponds to the gradient of the cost-to-go from $\hat{X}_s$

$$a_s := \nabla_{\hat{X}_s}\left[\int_s^1 C(\hat{X}^\theta_\tau, \tau) + \frac{1}{2}\|U_\theta(\hat{X}^\theta_\tau, \tau)\|^2 \,\mathrm{d}\tau + g(\hat{X}^\theta_1)\right]. \tag{73}$$

With column-vector adjoints and the Jacobian convention, we have:

$$D_x F(x) = \left(\partial_{x_j} F_i(x)\right)_{i,j}. \tag{74}$$

The full adjoint solves

$$\frac{\mathrm{d}}{\mathrm{d}s} a_s = -D_x \hat{f}_\theta(\hat{X}^\theta_s, s)^\top a_s - \nabla_x\left(C(\hat{X}^\theta_s, s) + \frac{1}{2}\|U_\theta(\hat{X}^\theta_s, s)\|^2\right), \tag{75}$$

$$a_1 = \nabla_x g(\hat{X}^\theta_1) = -\lambda D_x \varphi(\hat{X}^\theta_1)^\top \nabla_y r(\hat{Y}^\theta_1). \tag{76}$$

We can further develop the expression of the Jacobian and the gradient:

$$D_x \hat{f}_\theta(x, s) = D_x \hat{f}_s(x) + \hat{\sigma}_s D_x U_\theta(x, s). \tag{77}$$

Moreover,

$$\nabla_x\left(C(x, s) + \frac{1}{2}\|U_\theta(x, s)\|^2\right) = D_x U_\theta(x, s)^\top U_\theta(x, s) + \nabla_x C(x, s), \tag{78}$$

where $D_x U_\theta^\top U_\theta$ denotes the gradient of $\frac{1}{2}\|U_\theta\|^2$.

Write $a_s = (a_s^1, \ldots, a_s^N)$ with $a_s^i \in \mathbb{R}^d$. The equivalent block-wise representation of eq. (75) is given as

$$\frac{\mathrm{d}}{\mathrm{d}s} a_s^i = -\sum_{j=1}^N \left[D_{x^i} \hat{f}_s^j(\hat{X}^\theta_s) + \hat{\sigma}_s D_{x^i} U_\theta^j(\hat{X}^\theta_s, s)\right]^\top a_s^j \tag{79}$$

$$-\sum_{j=1}^N \left[D_{x^i} U_\theta^j(\hat{X}^\theta_s, s)\right]^\top U_\theta^j(\hat{X}^\theta_s, s) \tag{80}$$

$$-\partial_{x^i} C(\hat{X}^\theta_s, s), \tag{81}$$

with terminal condition

$$a_1^i = -\lambda D_{x^i} \varphi(\hat{X}^\theta_1)^\top \nabla_y r(\hat{Y}^\theta_1) \tag{82}$$

for each i.

C.2 Adjoint Matching Instantiation of CMDS

In Proposition 2 and Appendices E.2 and E.3 of Domingo-Enrich et al. (2025), it is shown that, under the assumptions stated there, the adjoint matching loss in eq. (6) and the original objective in eq. (72) share the same gradient with respect to θ. Moreover, in the corresponding control-space

formulation, the unique critical point of the adjoint matching loss in eq. (6) is the optimal control for the original objective in eq. (72).

Let $\bar{\theta} = \mathrm{stopgrad}(\theta)$, and sample paths from the frozen controlled process

$$\hat{X}^{\bar{\theta}} \sim \hat{\mathbb{P}}^{\bar{\theta}}. \tag{83}$$

Let $a^{\bar{\theta}}$ solve eq. (75)–eq. (76) along $\hat{X}^{\bar{\theta}}$, with all dependence on $\bar{\theta}$ stopped. The basic CMDS adjoint matching objective used in our work and shown in the main part of this paper is

$$\mathcal{L}_{\text{CMDS-AM}}(\theta) = \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^{\bar{\theta}}}\left[\int_0^1 \left\| U_\theta(\hat{X}_s^{\bar{\theta}}, s) + \hat{\sigma}_s a_s^{\bar{\theta}} \right\|^2 \mathrm{d}s\right]. \tag{6}$$

Block-wise,

$$\mathcal{L}_{\text{CMDS-AM}}(\theta) = \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^{\bar{\theta}}}\left[\int_0^1 \sum_{i=1}^{N} \left\| U_\theta^i(\hat{X}_s^{\bar{\theta}}, s) + \hat{\sigma}_s a_s^{\bar{\theta},i} \right\|^2 \mathrm{d}s\right]. \tag{84}$$

The gradients of $\mathcal{J}$ can be recovered using the adjoint (Domingo-Enrich et al., 2025).

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}\left[\int_0^1 D_\theta U_\theta(\hat{X}_s^\theta, s)^\top \Big(U_\theta(\hat{X}_s^\theta, s) + \hat{\sigma}_s a_s \Big)\, \mathrm{d}s\right]. \tag{85}$$

C.2.1 Relation to Lean Adjoint Matching and Agent-Wise Structure

The adjoint matching formulation above uses the full adjoint, so it keeps the same gradient with respect to the parameter θ as direct optimisation over the original stochastic control objective. Domingo-Enrich et al. (2025) also proposed a lean adjoint, obtained by removing terms depending on control from the adjoint dynamics, since their expectation equals 0 at the optimum. Consequently, the resulting matching loss has a different gradient from the original SOC objective whenever the optimum is not achieved, but the unique optimum is proven to be the same as the optimal control under their continuous control-affine assumptions. In CMDS, it is formulated as a control-affine SOC problem, and thus the same lean adjoint construction can be applied to the continuous time CMDS formulation. Specifically, the lean version of the original full adjoint dynamics (eq. (75)) is

$$\frac{\mathrm{d}\tilde{a}_s}{\mathrm{d}s} = -\left[D_x \hat{f}_s(\hat{X}_s^\theta)\right]^\top \tilde{a}_s - \nabla_x C(\hat{X}_s^\theta, s), \tag{86}$$

while $\tilde{a}_1$ is defined to be the same as a_1 in eq. (76). The corresponding adjoint matching objective is

$$\mathcal{L}_{\text{LeanAM}}(\theta) = \frac{1}{2}\mathbb{E}_{\hat{\mathbb{P}}^{\bar{\theta}}}\left[\int_0^1 \left\| U_\theta(\hat{X}_s^{\bar{\theta}}, s) + \hat{\sigma}_s \tilde{a}_s^{\bar{\theta}} \right\|^2 \mathrm{d}s\right]. \tag{87}$$

Agent-wise decomposition The simplification is natural for the reference product process. As pretrained agents evolve independently, the state Jacobian of $\hat{f}_s(x)$ is in the block diagonal form

$$D_x \hat{f}_s(x) = \mathrm{diag}(D_{x^1} \hat{f}_s^1(x^1), D_{x^2} \hat{f}_s^2(x^2), ..., D_{x^N} \hat{f}_s^N(x^N)),$$

and thus the corresponding lean adjoint in agent-wise form satisfies

$$\frac{\mathrm{d}\tilde{a}_s^i}{\mathrm{d}s} = -\left[D_{x^i} \hat{f}_s^{\,i}(\hat{X}_s^{\theta,i})\right]^\top \tilde{a}_s^i - \nabla_{x^i} C(\hat{X}_s^\theta, s). \tag{88}$$

In the main CMDS setting considered in this work, $C = 0$, and thus the backward propagation of $\tilde{a}$ separates among agents after the terminal adjoint has been initialised. Importantly, this does not remove the coordination as

$$\tilde{a}_1^i = -\lambda D_{x^i} \varphi(\hat{X}_1^\theta)^\top \nabla_y r(\hat{Y}_1^\theta). \tag{89}$$

Therefore, a precise interpretation is that the assembly reward performs joint credit assignment across agents at the terminal time, and the lean adjoint dynamics subsequently propagate the information assigned to each agent backward through each agent's own frozen dynamics.

Computational Interpretation For factorised reference dynamics, the lean-adjoint propagation only needs agent-wise vectors. If the cost of each component model is fixed, the computational cost of this lean-adjoint backward propagation grows linearly with respect to the number of agents, and can be parallelly computed. On the other hand, the full adjoint additionally differentiates through the joint control U_θ, and its block form contains cross-agent terms $D_{x^i} U_\theta^j$ representing dependencies introduced by the coordination network. The lean adjoint matching removes these interactions from the adjoint propagation through the reference dynamics. Note that this does not imply that the overall CMDS computation is necessarily linear in the number of agents N, as the evaluation of the assembly reward and joint control could potentially involve higher-order interactions among different agents.

Relation to our implementation For the reasons discussed above, in this work we retain the full adjoint rather than the lean variant. In particular, the discrete DDIM recursion below differentiates the controlled transition and includes the quadratic control-cost derivative, thereby preserving the full cross-agent dependencies that are not detached by the task-specific implementation. A discrete lean CMDS variant could instead propagate adjoints through the frozen base transitions only, in analogy with the lean DDIM construction of Domingo-Enrich et al. (2025).

C.2.2 Discrete-time adjoint matching for DDIM sampling

We use the rollout convention of Algorithm 1: K is the number of DDIM transitions, $\ell = 0, \dots, K-1$ indexes transitions, and $\zeta_0 > \cdots > \zeta_{K-1}$ are their native diffusion timesteps. The rollout runs from the initial noise state X_0 to the terminal sample X_K. Let $X_\ell = (X_\ell^1, \dots, X_\ell^N) \in \mathbb{R}^D$ denote the current realised joint state along the reverse DDIM trajectory, with the control superscript suppressed, and let

$$Y_\ell := \varphi(X_\ell), \qquad \Xi_\ell = (\Xi_\ell^1, \dots, \Xi_\ell^N) \sim \mathcal{N}(0, I_D),$$

with $\Xi_\ell^i \sim \mathcal{N}(0, I_d)$. For agent i, parameterise the fine-tuned denoiser as

$$\epsilon_{\theta,\ell}^i(x, y) := \epsilon_{\mathrm{b},\ell}^i(x^i) + \delta\epsilon_{\theta,\ell}^i(x, y), \qquad x = (x^1, \dots, x^N), \qquad y = \varphi(x), \tag{90}$$

where $\epsilon_{\mathrm{b},\ell}^i(x^i) := \epsilon_{\mathrm{b}}^i(x^i, \zeta_\ell)$. The control parameters are shared across DDIM transitions; the native diffusion timestep ζ_ℓ is supplied through the time input rather than indexing a separate parameter vector. In the two-agent implementation,

$$\delta\epsilon_{\theta,\ell}^1 = C_{\theta_1}(x^1, y, x^2, \zeta_\ell), \qquad \delta\epsilon_{\theta,\ell}^2 = C_{\theta_2}(x^2, y, x^1, \zeta_\ell).$$

For each DDIM transition, let $\bar\alpha_{\mathrm{src},\ell}$ and $\bar\alpha_{\mathrm{dst},\ell}$ denote its source and destination cumulative alphas, respectively. Here $\bar\alpha_{\mathrm{src},\ell} = \bar\alpha_{\zeta_\ell}$, and $\bar\alpha_{\mathrm{dst},\ell} = \bar\alpha_{\zeta_{\ell+1}}$ for $\ell < K-1$; the terminal destination value is modified as specified below. For $\eta > 0$, define

$$\varsigma_\ell := \eta \left[\frac{1 - \bar\alpha_{\mathrm{dst},\ell}}{1 - \bar\alpha_{\mathrm{src},\ell}} \left(1 - \frac{\bar\alpha_{\mathrm{src},\ell}}{\bar\alpha_{\mathrm{dst},\ell}} \right) \right]^{1/2}, \tag{91}$$

$$\gamma_\ell := \left[1 - \bar\alpha_{\mathrm{dst},\ell} - \varsigma_\ell^2 \right]_+^{1/2} - \left(\frac{\bar\alpha_{\mathrm{dst},\ell}}{\bar\alpha_{\mathrm{src},\ell}} \right)^{1/2} (1 - \bar\alpha_{\mathrm{src},\ell})^{1/2}. \tag{92}$$

The component DDIM map is

$$F_\ell(z, \epsilon, \xi) := \left(\frac{\bar\alpha_{\mathrm{dst},\ell}}{\bar\alpha_{\mathrm{src},\ell}} \right)^{1/2} z + \gamma_\ell \epsilon + \varsigma_\ell \xi. \tag{93}$$

At the terminal reverse transition $\ell = K-1$, we use $\bar\alpha_{\mathrm{dst},\ell} = (1 + \bar\alpha_{\mathrm{src},\ell})/2$, rather than $\bar\alpha_{\mathrm{dst},\ell} = 1$. Hence $\varsigma_\ell > 0$ at that transition as well, and its control cost is included.

The base and controlled component updates, evaluated from the same current state and using the same noise, are

$$X_{\ell+1}^{\mathrm{b},i} := F_\ell\big(X_\ell^i, \epsilon_{\mathrm{b},\ell}^i(X_\ell^i), \Xi_\ell^i\big), \tag{94}$$

$$X_{\ell+1}^{\theta,i} := F_\ell\big(X_\ell^i, \epsilon_{\theta,\ell}^i(X_\ell, Y_\ell), \Xi_\ell^i\big). \tag{95}$$

Using the same noise $\Xi_\ell^i \sim \mathcal{N}(0, I_d)$ in the base and controlled DDIM steps, the one-step displacement induced by the epsilon residual is

$$X_{\ell+1}^{\theta,i} - X_{\ell+1}^{\mathrm{b},i} = \gamma_\ell \delta\epsilon_{\theta,\ell}^i(X_\ell, Y_\ell). \tag{96}$$

The normalised discrete control is therefore

$$\begin{aligned} w_{\theta,\ell}^i(X_\ell) &:= \frac{1}{\varsigma_\ell}\left(X_{\ell+1}^{\theta,i} - X_{\ell+1}^{\mathrm{b},i}\right) \\ &= \frac{\gamma_\ell}{\varsigma_\ell}\delta\epsilon_{\theta,\ell}^i\big(X_\ell, \varphi(X_\ell)\big) \approx \sqrt{\Delta s_\ell}\, U_\theta^i(X_\ell, s_\ell), \end{aligned} \tag{97}$$

where $\varsigma_\ell > 0$ denotes the stochastic scale of DDIM transition ℓ. For the continuous-time interpretation, s_ℓ belongs to a generative-time grid $0 = s_0 < \cdots < s_K = 1$, with $\Delta s_\ell := s_{\ell+1} - s_\ell > 0$. The equality preceding the approximation is exact; the final relation is the continuous-time interpretation of the discrete control. Consequently,

$$\frac{1}{2}\sum_{i=1}^N \left\|w_{\theta,\ell}^i(X_\ell)\right\|^2$$

is the one-step discrete SOC control cost.

Define the joint base and controlled DDIM maps by

$$F_\ell^{\mathrm{b}}(X_\ell, \Xi_\ell) := \left(X_{\ell+1}^{\mathrm{b},1}, \ldots, X_{\ell+1}^{\mathrm{b},N}\right), \tag{98}$$

$$F_\ell^{\theta}(X_\ell, \Xi_\ell) := \left(X_{\ell+1}^{\theta,1}, \ldots, X_{\ell+1}^{\theta,N}\right) = X_{\ell+1}^{\theta}. \tag{99}$$

The base map is a one-step counterfactual used to define the discrete control; only the controlled map advances the sampled trajectory.

As in the continuous-time formulation, let $\bar\theta = \mathrm{stopgrad}(\theta)$, and let $\{X_\ell^{\bar\theta}\}_{\ell=0}^K$ denote a trajectory generated by the frozen controlled DDIM maps $F_\ell^{\bar\theta}$. Define the terminal cost

$$g(x) := -\lambda r(\varphi(x)).$$

The full discrete adjoint used by the CMDS objective is initialised at K and propagated for $\ell = K-1, \ldots, 0$:

$$\begin{aligned} a_K^{\bar\theta} &= \nabla_x g(X_K^{\bar\theta}) \\ &= -\lambda D_x\varphi(X_K^{\bar\theta})^\top \nabla_y r\left(\varphi(X_K^{\bar\theta})\right), \end{aligned} \tag{100}$$

$$a_\ell^{\bar\theta} = D_x F_\ell^{\bar\theta}\big(X_\ell^{\bar\theta}, \Xi_\ell\big)^\top a_{\ell+1}^{\bar\theta} + \nabla_x\left[\frac{1}{2}\sum_{i=1}^N \left\|w_{\bar\theta,\ell}^i(x)\right\|^2\right]\Bigg|_{x=X_\ell^{\bar\theta}}. \tag{101}$$

Here $a_{\ell+1}^{\bar\theta}$ is the costate of the post-transition state $X_{\ell+1}^{\bar\theta}$; it is paired with the action at transition ℓ before the recursion is propagated to $a_\ell^{\bar\theta}$.

Equivalently, define

$$\Delta_\ell^{\bar\theta}(x, \xi) := F_\ell^{\bar\theta}(x, \xi) - x. \tag{102}$$

Then the adjoint recursion can be written as

$$a_\ell^{\bar\theta} = a_{\ell+1}^{\bar\theta} + \nabla_x\left[\left\langle a_{\ell+1}^{\bar\theta}, \Delta_\ell^{\bar\theta}(x, \Xi_\ell)\right\rangle + \frac{1}{2}\sum_{i=1}^N \left\|w_{\bar\theta,\ell}^i(x)\right\|^2\right]\Bigg|_{x=X_\ell^{\bar\theta}}. \tag{103}$$

Because the DDIM stochastic term $\varsigma_\ell\Xi_\ell$ is additive and independent of the current state, the state derivative in the adjoint recursion is unchanged if Ξ_ℓ is set to zero during adjoint re-computation.

The discrete CMDS adjoint matching loss is

$$\mathcal{L}_{\mathrm{CMDS}}(\theta) = \mathbb{E}\left[\sum_{\ell\in\mathcal{K}}\frac{1}{2}\sum_{i=1}^N \left\|w_{\theta,\ell}^i\big(\mathrm{stopgrad}(X_\ell^{\bar\theta})\big) + \varsigma_\ell\,\mathrm{stopgrad}\big(a_{\ell+1}^{\bar\theta,i}\big)\right\|^2\right], \tag{104}$$

where $\mathcal{K} \subseteq \{0, \dots, K-1\}$ is the set of stochastic DDIM transitions selected for the matching loss. The adjoint recursion itself is evaluated over all DDIM transitions; when $\mathcal{K}$ is a strict subset, the matching terms are not importance re-weighted. The implementation evaluates eq. (101) through vector–Jacobian products of the controlled joint map $F_\ell^{\bar{\theta}}$, while including $\frac{1}{2}\sum_i \|w^i_{\bar{\theta},\ell}\|^2$ in the adjoint recursion. The complete discrete-time adjoint-matching procedure, including the controlled rollout, backward adjoint replay, and control regression, is summarised in Algorithm 1. It therefore implements the full discrete adjoint of the controlled DDIM map under the task-specific detach conventions, including all non-detached cross-agent derivatives induced by φ and the controls. Note that in our implementation, the matched-transition set $\mathcal{K}$ is resampled at each optimisation iteration, while the adjoint recursion is always evaluated over all reverse transitions. Note that in Algorithm 1, we report a generic context builder $\mathcal{C}_\kappa : (X_\ell, \hat{X}_{1|\ell}, Y_\ell, \hat{Y}_{1|\ell}, \zeta_\ell) \mapsto \mathcal{I}_\ell$, which is experiment-specific. Thus, we use the basic, full-adjoint form of adjoint matching, discretised directly through the DDIM transitions. Unlike the lean-adjoint variant of Domingo-Enrich et al. (2025), the backward replay differentiates the controlled transition and the quadratic control cost.

Algorithm 1 Adjoint matching for CMDS (discrete time)

Require: frozen component denoisers ϵ_{b}^i, $i = 1, \dots, N$; parametrised component-wise residual $\delta\epsilon_\theta$; assembly map φ; task-dependent context builder $\mathcal{C}_\kappa$; terminal reward r_κ; reward strength λ; reverse-time grid $\zeta_0 > \cdots > \zeta_{K-1}$; joint controlled reverse maps F_ℓ^θ, signed epsilon-displacement coefficients γ_ℓ, and stochastic scales $\varsigma_\ell > 0$; matched transitions $\mathcal{K} \subseteq \{0, \dots, K-1\}$; batch size B

Sample task conditions $\kappa_1, \dots, \kappa_B$ # e.g., target $d_{\Delta x}$, observation b, or starts/goals $\{(s_i, g_i)\}_{i=1}^N$

Sample $X_0^{i,(b)} \sim \mathcal{N}(0, I)$ and independent rollout noise $\Xi_\ell^{i,(b)} \sim \mathcal{N}(0, I)$

$\bar{\theta} \leftarrow \mathrm{stopgrad}(\theta)$

1. Controlled rollout without gradient tracking

for $\ell = 0, \dots, K-1$ **do**

  for $i = 1, \dots, N$ **do**

    $\epsilon_{\mathrm{b},\ell}^i \leftarrow \epsilon_{\mathrm{b}}^i(X_\ell^i, \zeta_\ell)$

    Compute $\hat{X}_{1|\ell}^i$ # Tweedie: $\hat{X}_{1|\ell}^i = \frac{X_\ell^i - \sqrt{1-\bar{\alpha}_{\zeta_\ell}}\,\epsilon_{\mathrm{b},\ell}^i}{\sqrt{\bar{\alpha}_{\zeta_\ell}}}$

  end for

  $\hat{X}_{1|\ell} \leftarrow \left(\hat{X}_{1|\ell}^1, \dots, \hat{X}_{1|\ell}^N\right)$

  $Y_\ell \leftarrow \varphi(X_\ell), \qquad \hat{Y}_{1|\ell} \leftarrow \varphi(\hat{X}_{1|\ell})$

  $\mathcal{I}_\ell \leftarrow \mathcal{C}_\kappa\left(X_\ell,\, \hat{X}_{1|\ell},\, Y_\ell,\, \hat{Y}_{1|\ell},\, \zeta_\ell\right)$ # The control for agent i depends on the full multi-agent state

  $\left(\delta\epsilon_{\bar{\theta},\ell}^1, \dots, \delta\epsilon_{\bar{\theta},\ell}^N\right) \leftarrow \delta\epsilon_{\bar{\theta}}(\mathcal{I}_\ell)$

  for $i = 1, \dots, N$ **do**

    $w_{\bar{\theta},\ell}^i \leftarrow \frac{\gamma_\ell}{\varsigma_\ell}\delta\epsilon_{\bar{\theta},\ell}^i$

    $X_{\ell+1}^i \leftarrow F_\ell\left(X_\ell^i,\, \epsilon_{\mathrm{b},\ell}^i + \delta\epsilon_{\bar{\theta},\ell}^i,\, \Xi_\ell^i\right)$ # Controlled DDIM rollout; see eqs. (90), (93) and (97)

  end for

  Store X_ℓ

end for

2. Backward discrete-adjoint replay

$a_K \leftarrow \nabla_{X_K}\left[-\lambda \sum_{b=1}^B r_{\kappa_b}\left(\varphi(X_K^{(b)})\right)\right]$ # Terminal costate

for $\ell = K-1, \dots, 0$ **do**

  $\tilde{a}_{\ell+1} \leftarrow \mathrm{stopgrad}(a_{\ell+1})$ # Destination costate

  Re-evaluate $F_\ell^{\bar{\theta}}(x, 0)$ and $w_{\bar{\theta},\ell}(x)$ at $x = \mathrm{stopgrad}(X_\ell)$, using the task-specific detach conventions # Differentiate all non-detached state routes; keep $\bar{\theta}$ fixed

$$a_\ell \leftarrow \mathrm{stopgrad}\left[\tilde{a}_{\ell+1} + \nabla_x\left(\left\langle \tilde{a}_{\ell+1}, F_\ell^{\bar{\theta}}(x,0) - x\right\rangle + \frac{1}{2}\sum_{b=1}^B\sum_{i=1}^N \left\| w_{\bar{\theta},\ell}^{i,(b)}(x)\right\|^2\right)\Bigg|_{x=X_\ell}\right].$$

end for

3. Control regression

for $\ell \in \mathcal{K}$ **do**

  Build $\mathcal{I}_\ell = \mathcal{C}_\kappa\left(\mathrm{stopgrad}(X_\ell), \hat{X}_{1|\ell}, Y_\ell, \hat{Y}_{1|\ell}, \zeta_\ell\right)$ using domain-specific detach conventions

  $\left(\delta\epsilon_{\theta,\ell}^1, \dots, \delta\epsilon_{\theta,\ell}^N\right) \leftarrow \delta\epsilon_\theta(\mathcal{I}_\ell)$

  for $i = 1, \dots, N$ **do**

    $w_{\theta,\ell}^i \leftarrow \frac{\gamma_\ell}{\varsigma_\ell}\delta\epsilon_{\theta,\ell}^i$

    $e_{\theta,\ell}^i \leftarrow w_{\theta,\ell}^i + \varsigma_\ell\,\mathrm{stopgrad}\left(a_{\ell+1}^i\right)$ # Adjoint-matching regression residual

  end for

end for

$$\mathcal{L}_{\mathrm{AM}}(\theta) \leftarrow \frac{1}{2B}\sum_{b=1}^B\sum_{\ell\in\mathcal{K}}\sum_{i=1}^N \left\| e_{\theta,\ell}^{i,(b)}\right\|^2$$ # PointMaze uses the self-calibrated pseudo-Huber variant; see Appendix D.4.7

$\theta \leftarrow \textsc{OptimizerStep}(\theta,\, \nabla_\theta \mathcal{L}_{\mathrm{AM}}(\theta))$

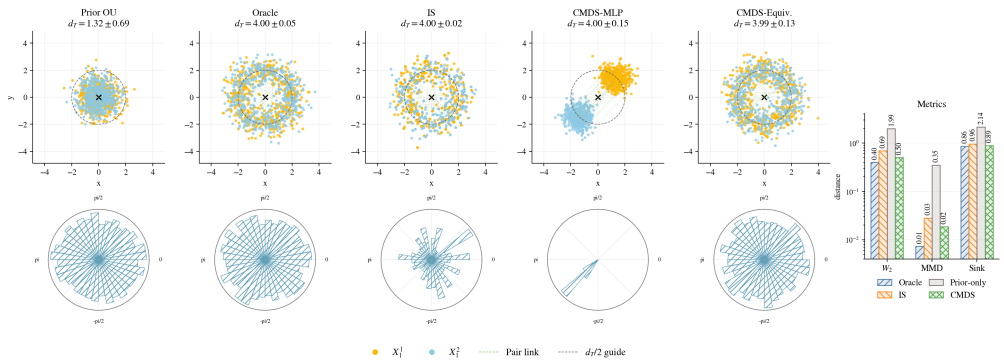


Figure 8: **Two-agent SOC toy: discrete adjoint.** Two agents start from an OU prior and are rewarded for reaching terminal configurations with a prescribed separation. We compare the uncontrolled prior, a value-function oracle, an importance-sampled reference (IS), and the learned control.

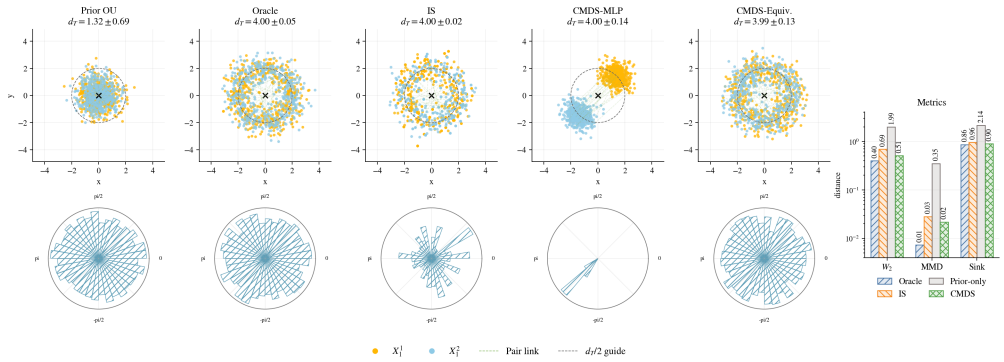


Figure 9: **Two-agent SOC toy: adjoint matching.** Two agents start from an OU prior and are rewarded for reaching terminal configurations with a prescribed separation. We compare the uncontrolled prior, a value-function oracle, an importance-sampled reference (IS), and the learned control.

D Experiments

D.1 Two-Agent OU Coordination: A Stochastic-Control Sanity Check

Key Takeaways: Least-action coordination recovers the target law. Across figs. 8 and 9, we compare the learned terminal distribution against the stochastic-control oracle using the 2-Wasserstein distance (W_2), maximum mean discrepancy (MMD), and Sinkhorn divergence. Together with the terminal-separation and angular diagnostics, these metrics show whether the control recovers the full reward-tilted distribution, rather than merely satisfying the prescribed distance constraint. **Relational structure matters.** A rotation-equivariant control preserves the symmetry of the target law, whereas an unconstrained MLP can satisfy the distance constraint while collapsing onto a preferred direction. **Constraint satisfaction alone may not be sufficient.** A control may achieve the correct terminal separation while remaining far from the oracle distribution. In particular, the angular histograms expose symmetry collapse that is not visible from the distance constraint alone, while W_2, MMD, and Sinkhorn divergence quantify this distributional mismatch. **The experiment isolates the control mechanism.** In this setting the reference dynamics are known and an exact reduced value-function oracle is available, providing a controlled sanity check that the coordination objective learns the intended low-action path-space perturbation.

To isolate the stochastic-control mechanism underlying our coordination framework, we consider two agents, each evolving in $\mathbb{R}^2$ according to an Ornstein–Uhlenbeck (OU) reference process. This experiment is a low-dimensional sanity check in which the reference dynamics are known and a high-accuracy value-function oracle can be computed in a reduced coordinate. We use ordinary forward control time $t \in [0, 1]$ in this experiment, and therefore omit the reverse-time hats used for diffusion sampling elsewhere in the paper. Let $X_t = (X_t^1, X_t^2) \in \mathbb{R}^4$ denote the joint state. The

controlled dynamics are:

$$\mathrm{d}X_t^i = \left[-X_t^i + U_\theta^i(X_t, t)\right]\mathrm{d}t + \mathrm{d}W_t^i, \qquad i \in \{1, 2\},$$

where W^1 and W^2 are independent two-dimensional Brownian motions and the uncontrolled drift $-X_t^i$ pulls each agent toward the origin. The two agents are coupled only through the terminal relational cost,

$$g(X_1) = \beta\left(\|X_1^2 - X_1^1\| - d_T\right)^2,$$

which encourages their terminal separation to match a prescribed distance d_T. The corresponding stochastic optimal-control objective is

$$J(\theta) = \mathbb{E}_{\mathbb{P}^\theta}\left[g(X_1) + \frac{\alpha}{2}\int_0^1 \sum_{i=1}^2 \left\|U_\theta^i(X_t, t)\right\|^2 \mathrm{d}t\right],$$

where $\mathbb{P}^\theta$ denotes the path law induced by the controlled dynamics, $\alpha > 0$ controls the path-action penalty, and $\beta > 0$ controls the strength of the terminal constraint. In the uncontrolled process, both agents concentrate near the origin, and their terminal separation is typically substantially smaller than d_T. A successful control should therefore separate the agents sufficiently to satisfy the relational constraint while avoiding unnecessary translation of their centre of mass.

Reduced value-function oracle. Since the two agents have the same component-wise OU dynamics, their common and relative modes can be decoupled. Since the terminal cost only depends on the relative displacement and their quadratic action is minimised with zero common mode control (i.e., $q_t = \dfrac{u_t^1 + u_t^2}{2}$), the control problem can be reduced to the relative coordinate

$$R_t := X_t^2 - X_t^1 \in \mathbb{R}^2.$$

For controls u_t^1, u_t^2, define the relative control $v_t := u_t^2 - u_t^1$. For any fixed v_t, the minimum-action decomposition across the two agents is

$$u_t^1 = -\frac{1}{2}v_t, \qquad u_t^2 = \frac{1}{2}v_t,$$

and hence

$$\frac{\alpha}{2}\left(\|u_t^1\|^2 + \|u_t^2\|^2\right) = \frac{\alpha}{4}\|v_t\|^2.$$

The relative controlled process therefore satisfies

$$\mathrm{d}R_t = (-R_t + v_t)\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}B_t,$$

where $B_t = (W_t^2 - W_t^1)/\sqrt{2}$ is a standard two-dimensional Brownian motion. Writing

$$g_R(r) := \beta\left(\|r\| - d_T\right)^2,$$

the reduced value function is

$$V(t, r) := \inf_v \mathbb{E}\left[g_R(R_1) + \frac{\alpha}{4}\int_t^1 \|v_s\|^2 \mathrm{d}s \,\middle|\, R_t = r\right].$$

Its Hamilton–Jacobi–Bellman equation is

$$\partial_t V - r^\top \nabla_r V + \Delta_r V - \frac{1}{\alpha}\|\nabla_r V\|^2 = 0, \qquad V(1, r) = g_R(r).$$

Introducing the desirability function

$$\psi(t, r) := \exp\left(-\frac{1}{\alpha}V(t, r)\right),$$

linearises the HJB equation. Equivalently, by the Feynman–Kac representation,

$$\psi(t, r) = \mathbb{E}\left[\exp\left(-\frac{1}{\alpha}g_R(R_1)\right)\,\middle|\, R_t = r\right] = \mathbb{E}\left[\exp\left(-\frac{\beta}{\alpha}\left(\|R_1\| - d_T\right)^2\right)\,\middle|\, R_t = r\right],$$

where $\frac{\beta}{\alpha}$ matches the λ in the SOC formulation, and $V(t,r) = -\alpha \log \psi(t,r)$. Under the uncontrolled OU reference process,

$$\mathrm{d}R_t = -R_t\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}B_t,$$

so that

$$R_1 \mid R_t = r \sim \mathcal{N}\left(e^{-(1-t)}r, \left(1 - e^{-2(1-t)}\right) I_2\right).$$

Since the terminal cost depends only on $\|R_1\|$, the value function is radial, and the conditional distribution of $\|R_1\|$ is Rice distributed. Consequently, $\psi(t,r)$ can be evaluated accurately by one-dimensional quadrature over the terminal radius. Minimising the reduced Hamiltonian gives

$$v^\star(t,r) = -\frac{2}{\alpha}\nabla_r V(t,r) = 2\nabla_r \log \psi(t,r),$$

with the corresponding minimum-action split

$$u_t^{1,\star} = -\frac{1}{2}v^\star(t,R_t), \qquad u_t^{2,\star} = \frac{1}{2}v^\star(t,R_t).$$

This defines the radial stochastic-control oracle used in figs. 8 and 9. The HJB construction above directly introduces the optimal change of path measure through $\psi(t,r) = \exp(-V(t,r)/\alpha)$, which solves the backward Kolmogorov equation of the uncontrolled relative OU process

$$\mathrm{d}R_t = -R_t\mathrm{d}t + \sqrt{2}\mathrm{d}B_t,$$

with terminal condition $\psi(1,r) = \exp(-g_R(r)/\alpha)$. Then, by Itô's formula, under the uncontrolled law,

$$\mathrm{d}\log\psi(t,R_t) = \sqrt{2}\nabla_r \log\psi(t,R_t)\cdot \mathrm{d}B_t - \|\nabla_r \log\psi(t,R_t)\|^2\mathrm{d}t,$$

and thus

$$\frac{\psi(t,R_t)}{\psi(0,R_0)} = \exp\left(\int_0^t \frac{v^\star(s,R_s)}{\sqrt{2}}\cdot \mathrm{d}B_s - \frac{1}{2}\int_0^t \left\|\frac{v^\star(s,R_s)}{\sqrt{2}}\right\|^2 \mathrm{d}s\right).$$

By Girsanov's theorem, the RHS of the above equation is exactly the Radon-Nikodym derivative of the optimally controlled path measure of R_t with respect to its uncontrolled counterpart up to time t. Evaluating at $t = 1$, it gives

$$\frac{\mathrm{d}\mathbb{P}_R^\star}{\mathrm{d}\mathbb{P}_R}(R_\cdot) = \frac{\exp(-g_R(R_1)/\alpha)}{\psi(0,R_0)}.$$

Since our above formulation has shown that controls are only applied to R_t and the common mode is uncontrolled and since $g(X_1) = g_R(R_1)$ in this OU example, substituting these relations back to the two-agent process gives

$$\frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}}(X_\cdot) = \frac{\exp(-g(X_1)/\alpha)}{\psi(0, X_0^2 - X_0^1)}.$$

The importance-sampling baseline approximates this same change of measure by drawing trajectories from the uncontrolled OU process and weighting each sample proportionally to $\dfrac{\exp(-g(X_1)/\alpha)}{\psi(0,R_0)}$.

Learning the control. We train the same feedback-control parameterisation using either direct discrete adjoint (a.k.a., differentiable optimisation of $J(\theta)$ or differentiable simulations) or an adjoint-matching surrogate (Domingo-Enrich, 2024). For the latter, let $\bar{\theta} = \mathrm{stopgrad}(\theta)$ and sample a trajectory $X^{\bar{\theta}} \sim \mathbb{P}^{\bar{\theta}}$ without gradient tracking. At terminal time, the component-wise adjoints are

$$a_1^{i,\bar{\theta}} = \nabla_{X_1^i} g(X_1^{\bar{\theta}}), \qquad i \in \{1,2\}.$$

For this low-dimensional diagnostic, we deliberately use a simplified reference-drift adjoint obtained by stopping the state dependence of the neural feedback control during adjoint propagation. Since the reference OU drift is $f(x) = -x$, the corresponding adjoint equation is

$$\frac{\mathrm{d}}{\mathrm{d}t}a_t^{i,\bar{\theta}} = -D_x f(X_t^{i,\bar{\theta}})^\top a_t^{i,\bar{\theta}} = a_t^{i,\bar{\theta}},$$

with closed-form backward solution $a_t^{i,\bar{\theta}} = e^{t-1}a_1^{i,\bar{\theta}}, i \in \{1,2\}$. This reference-drift adjoint is intentionally used only as an analytic diagnostic in the OU experiment. It should be distinguished from the full CMDS adjoint, which additionally differentiates the controlled drift and the running control cost with respect to the joint state.

For the individual control u_t^i, the local Hamiltonian contribution is

$$\frac{\alpha}{2}\|u_t^i\|^2 + \left(a_t^{i,\bar{\theta}}\right)^\top u_t^i.$$

Its first-order optimality condition is $\alpha u_t^i + a_t^{i,\bar{\theta}} = 0$, and therefore $u_t^{i,\star} = -\frac{1}{\alpha}a_t^{i,\bar{\theta}}$. We accordingly train the control using

$$\mathcal{L}_{\mathrm{AM}}^{\mathrm{OU}}(\theta) = \frac{1}{2}\mathbb{E}_{\mathbb{P}^{\bar{\theta}}}\left[\int_0^1 \sum_{i=1}^{2}\left\|U_\theta^i(X_t^{\bar{\theta}},t) + \frac{1}{\alpha}a_t^{i,\bar{\theta}}\right\|^2 \mathrm{d}t\right].$$

This surrogate regresses the learned feedback controls onto the least-action correction implied by the terminal relational cost and the reference OU dynamics.

Control parameterisation. We consider two feedback-control parameterisations. The first is an unconstrained multilayer perceptron that receives the complete joint state and time, $U_\theta^i = U_\theta^i(X_t,t)$. The second explicitly encodes the rotational symmetry of the problem,

$$U_\theta^i(X_t,t) = c_\theta\left(\|X_t^j - X_t^i\|^2, t\right)\left(X_t^j - X_t^i\right), \qquad j \neq i.$$

This equivariant parameterisation is a natural inductive bias because both the reference dynamics and the terminal cost are invariant to a common rotation of the two-agent system. The optimal correction therefore acts along the relative displacement between the agents, pushing them apart when their predicted terminal separation is too small and pulling them together when it is too large. The comparisons in figs. 8 and 9 highlight the importance of this inductive bias. The uncontrolled OU process concentrates both agents near the origin and therefore produces terminal separations substantially below the target. The value-function oracle achieves the prescribed separation while preserving rotational symmetry. The importance-sampling baseline approximates the same terminal tilt but suffers from finite-sample variance. The unconstrained MLP control can substantially reduce the terminal-distance error, but it may select a preferred direction in the plane and thereby collapse the terminal-angle distribution. In contrast, the rotation-equivariant control achieves a terminal separation close to the target while maintaining an approximately uniform angular distribution, substantially closer to the oracle. Thus, terminal-distance accuracy alone is insufficient: a control may satisfy the relational constraint while distorting symmetries and diversity of the target law. This toy problem provides a controlled SOC sanity check in which the component dynamics are known, the global constraint is purely relational, and a reduced value-function oracle is available. The experiment tests whether a learned control can balance satisfaction of a terminal system-level constraint against quadratic path action. It therefore provides a low-dimensional analogue of the least-action principle underlying CMDS: instead of arbitrarily moving samples toward high reward, the control seeks the smallest path-space perturbation required to make independently evolving components satisfy a consistency criterion.

D.2 Relational Visual Assembly

Key Takeaways: Compositionality helps. When the sampler is factorised according to the object structure, coordination is substantially easier than forcing a single process to learn the full constraint through control alone due to positive inductive bias of the decomposition. **Rewards are fragile.** Reward-based coordination is only as reliable as the system-level signal used to define it. Poorly specified rewards can be over-optimised or hacked. **Least-work coordination.** The SOC objective favours assemblies that satisfy the global design while requiring the smallest deviation from the independent pretrained dynamics. Thus CMDS is not only searching for valid component combinations; it is learning a low-action path to valid assemblies. CMDS is sampling high-reward assemblies while paying a quadratic control/action cost that penalises deviation from the pretrained reverse diffusion dynamics.

This experiment is not reported in the main paper, but serves as a controlled testbed in which the coordination network can be trained cheaply, allowing us to study the method in detail from both

theoretical and computational perspectives. In particular, it lets us isolate the effects of component factorisation, optimisation strategy, and reward design, and examine how different rewards trade off constraint satisfaction, diversity, and the amount of control required to steer the pretrained processes.

Setup. We test two claims central to CMDS: whether a single amortised control can satisfy a family of relational constraints, and whether explicitly factorising the sampler by component simplifies relational steering. We consider one-channel 10×10 canvases with a white background and dark 2×2 squares. The target distribution consists of valid two-square canvases satisfying the prescribed constraints. We pretrain a single diffusion model G on canvases containing one square, and instantiate two independent frozen copies at sampling time. No samples from the valid two-square distribution are provided to CMDS during either pretraining or coordination. Their controlled terminal states are combined through the fixed left/right assembly

Method	Optim. scheme	ELBO (↑)	$\log Z_{\mathrm{IS}}$	Valid (↑)
$N = 1$ (1-square)	ADJ. MATCH.	-1303.1	-7.2	0.0%
	DISC. ADJ.	-1309.5	-7.2	0.0%
$N = 1$ (2-square)	ADJ. MATCH.	-289.1	-0.7	18.9%
	DISC. ADJ.	-173.4	-0.7	25.4%
CMDS	ADJ. MATCH.	-74.3	-1.6	**99.2%**
	DISC. ADJ.	**-61.2**	-1.6	98.5%

Table 6: **Classifier-only steering diagnostics.** We report $\log Z_{\mathrm{IS}}$ as an importance-sampling estimate of the log normalising constant of the reward-tilted target, providing a calibration diagnostic for the strength of the induced tilt.

$$\hat{Y}_1^\theta = \varphi(\hat{X}_1^{\theta,1}, \hat{X}_1^{\theta,2}) = M_L \odot \hat{X}_1^{\theta,1} + M_R \odot \hat{X}_1^{\theta,2},$$

where $M_L, M_R \in \{0,1\}^{10\times 10}$ mask the left and right halves and satisfy $M_L + M_R = 1$. Each component contributes only within its assigned half; its contribution outside that region is masked out. The masks therefore define the components' regions of influence, but do not ensure that a complete square appears in each visible half or that the two squares satisfy the requested relative separation. Independent sampling can consequently produce plausible single-square components but invalid assemblies.

Left-Right Steering. We first evaluate whether the agents can learn an additional drift on top of the pretrained dynamics that guides each square toward its prescribed region of interest. For this initial experiment, we train a simple multilayer perceptron to classify whether a square lies in the target half of the domain. The classifier-only assembly reward is

$$r_{\mathrm{clf}}(y) = \log \sigma(f_\psi(M_L \odot y)) + \log \sigma(f_\psi(M_R \odot y)),$$

where f_ψ denotes the trained classifier and each term scores square presence in its corresponding masked half. In fig. 11, we show samples of the aggregated canvas, illustrating that fine-tuning produces samples that satisfy the prescribed constraints. In both cases, each component model generates a square in its assigned region, so that the final stitched assembly is valid. The output of the first model is masked on the right half of the canvas, while the output of the second model is masked on the left half. We optimise the controls using both the discrete adjoint method (DISC. ADJ.) and adjoint matching (ADJ. MATCH.), as described in section 2.1. We compare our approach against two single-process baselines: (i) tilting the same one-square base model used as the component generator in CMDS; and (ii) tilting a generator trained directly to sample two 2×2 squares within the 10×10 canvas.

One-square baseline. The failure of the one-square baseline is expected as its pretrained support is misaligned with the target object class. Since the base model represents canvases with a single 2×2 square, producing a valid two-square configuration requires the control to create (or hallucinate) an additional square through tilting alone, rather than merely steering existing component structure.

Two-square baseline. The two-square baseline has the correct object count, but not the right factorisation. It can generate two-square canvases, yet the control must still solve the assignment problem of which square should satisfy which part of the left/right and distance reward (below). This creates an ambiguous credit-assignment problem. The reward gradients act on the whole image rather than on separately addressable components. In contrast, CMDS builds the assignment into the sampler by using one component process per square and applying the reward after assembly, so the same global signal induces more targeted, lower-action coordination.

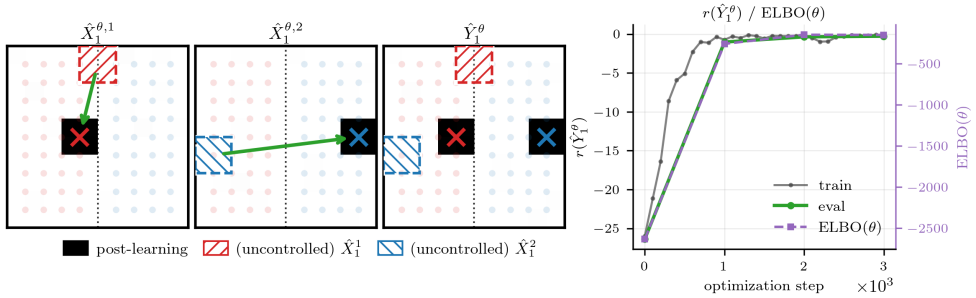


Figure 10: **Left-right steering overview.** Two copies of a pretrained single-square diffusion generator are coordinated through a controlled reverse process and assembled by the fixed masking operator $\hat{Y}_1^\theta = \varphi(\hat{X}_1^{\theta,1}, \hat{X}_1^{\theta,2})$. Black filled squares show the controlled terminal assembly $\hat{Y}_1^\theta$. Red and blue dashed outlines show the paired counterfactual uncontrolled terminal proposals $\hat{X}_1^1$ and $\hat{X}_1^2$, obtained from the same initial noise draw but with the learned control removed. Thus, each dashed outline indicates where the corresponding generator copy would have ended up under the reference product law $\hat{\mathbb{P}}$, before reward-based steering of the assembled canvas. The training curves report the terminal reward $r(\hat{Y}_1^\theta)$ and the path-space objective $\mathrm{ELBO}(\theta) = \mathbb{E}_{\hat{\mathbb{P}}^\theta}[\lambda r(\hat{Y}_1^\theta) - A_\theta]$, where A_θ denotes the accumulated control/action cost. The learned controls steer the two locally valid component generators into their assigned left/right regions while remaining close to the pretrained dynamics. The path-action cost remains low because the uncontrolled generator already places substantial mass near valid square configurations, so the control only needs a moderate path-space perturbation to steer samples into high-reward assemblies.

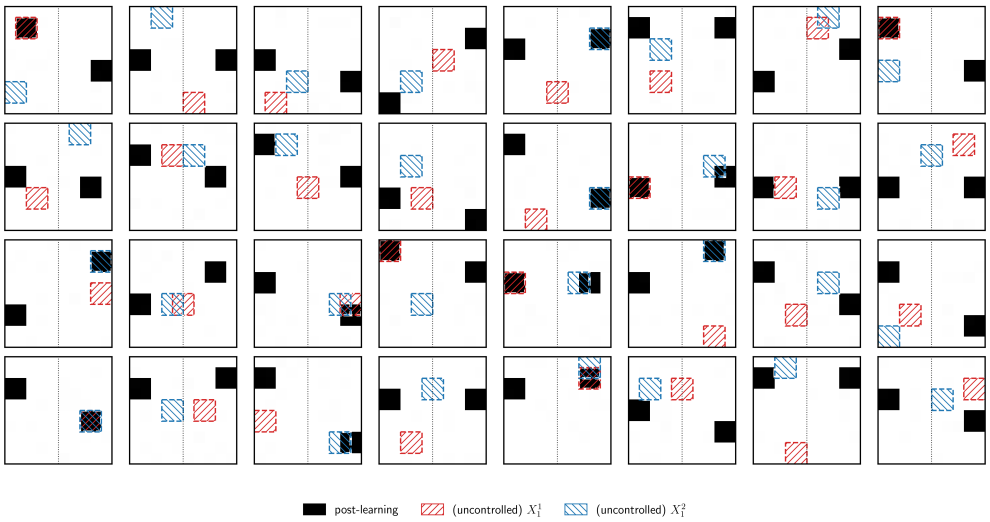


Figure 11: **Left-right steering (Samples).** Samples from the aggregated canvas after training. Black filled squares denote the post-learning assembled output. Red and blue dashed outlines denote the paired uncontrolled terminal proposals from the two generator copies, obtained from the same initial/noise draw but with the learned control removed. Thus, for each sample, the outlines show where that instance would have landed without reward-based steering of the assembled canvas. The learned controls steer each square into its assigned half of the canvas, yielding valid stitched assemblies under the fixed left/right masking operator.

Metrics. In tables 6 to 8, we report the path-space evidence lower bound (ELBO)

$$\mathrm{ELBO}(u) = \mathbb{E}_{\hat{\mathbb{P}}^u}\left[\lambda r\Big(\varphi(\hat{X}_1^u)\Big) - A_u\right],$$

Training scheme	ELBO$_{d_{\Delta x}=5}$ (↑)	$\log Z_{\mathrm{IS}}|_{d_{\Delta x}=5}$	Valid (↑)				
			$d_{\Delta x}=3$	$d_{\Delta x}=4$	$d_{\Delta x}=5$	$d_{\Delta x}=6$	Avg.
ADJ. MATCH.	-172.2	-5.4	**95.8%**	95.4%	**97.3%**	**97.1%**	**96.4%**
DISC. ADJ.	**-158.5**	-5.4	95.2%	**95.7%**	97.2%	95.5%	95.9%

Table 8: **Amortized steering diagnostics.** ELBO and $\log Z_{\mathrm{IS}}$ are evaluated at $d_{\Delta x}=5$. Validity is evaluated at every supported target using 4096 samples per target with matched random seeds across target conditions. Avg. is the macro-average over $d_{\Delta x}\in\{3,4,5,6\}$.

where $\hat{\mathbb{P}}^u$ is the controlled path measure and A_u is the accumulated control path-action. The normalizing constant of the reward-tilted target,

$$Z=\mathbb{E}_{\hat{\mathbb{P}}}\Big[\exp\Big(\lambda r\Big(\varphi(\hat{X}_1)\Big)\Big)\Big],$$

is estimated independently from uncontrolled samples $\hat{X}_1^{(j)}$ which follow the terminal time marginal of $\hat{\mathbb{P}}$ via

$$\log Z_{\mathrm{IS}}=\log\left(\frac{1}{M}\sum_{j=1}^{M}\exp\Big(\lambda r\Big(\varphi(\hat{X}_1^{(j)})\Big)\Big)\right).$$

Thus, $\log Z-\mathrm{ELBO}(u)$ is the path-space KL divergence from the controlled sampler to the reference path distribution reweighted by the exponential of the scaled terminal reward and normalised by Z. This unconditional reward tilt differs from the fixed-initial-law SOC optimum unless the conditional normaliser $Z(\hat{X}_0)$ is constant. In the numerical diagnostic, $\log Z$ is replaced by its estimate $\log Z_{\mathrm{IS}}$. Here A_θ denotes the quadratic path-action, i.e., the accumulated control cost along a parametrised controlled trajectory. Following Algorithm 1, we write X_ℓ for the controlled rollout state, suppressing its parameter superscript, and ζ_ℓ for the native diffusion time-step at transition ℓ.

$$A_\theta=\frac{1}{2}\sum_{\ell=0}^{K-1}\sum_{i=1}^{N}\left\|w_{\theta,\ell}^{i}(X_\ell)\right\|_2^2,$$

where $w_{\theta,\ell}^{i}$ is the normalised discrete control associated with component i at DDIM transition ℓ. For multi-component samplers, the norm includes the sum over all controlled processes and pixels. This is the discrete Girsanov control cost appearing in the path-space change of measure from the uncontrolled process $\hat{\mathbb{P}}$ to the controlled process $\hat{\mathbb{P}}^u$.

Coupled Steering. We impose a constraint that can only be satisfied through coordination. The assembly-level reward combines square presence with the requested horizontal separation:

$$r_{d_{\Delta x}}(y)=r_{\mathrm{clf}}(y)-\alpha|\Delta_x(y)-d_{\Delta x}|,$$

where $d_{\Delta x}$ is the target separation and α is the distance penalty weight. The achieved separation is

$$\Delta_x(y)=\mu_x(M_R\odot y)-\mu_x(M_L\odot y),$$

with μ_x denoting the horizontal foreground barycentre. This introduces coordination through an analytic distance term computed between the barycentres of the two squares. This term makes each agent's reward depend on the relative configuration of both squares, encouraging the learned drifts to coordinate rather than solve two independent localisation tasks. In this setting, we also compare

Method	Optim. scheme	ELBO (↑)	$\log Z_{\mathrm{IS}}$	Valid (↑)
$N=1$ (1-square)	ADJ. MATCH.	-1706.8	-22.8	0.0%
	DISC. ADJ.	-1565.0	-22.8	0.0%
$N=1$ (2-square)	ADJ. MATCH.	-1561.7	-4.5	8.3%
	DISC. ADJ.	-381.4	-4.5	9.6%
CMDS	ADJ. MATCH.	**-156.7**	-5.4	**97.9%**
	DISC. ADJ.	-190.9	-5.4	96.8%

Table 7: **Coupled steering diagnostics.** We report $\log Z_{\mathrm{IS}}$ as an importance-sampling estimate of the log normalising constant of the reward-tilted target, providing a diagnostic for the strength of the induced tilt. Single-process baselines struggle to satisfy the coupled validity constraint, whereas CMDS coordinates two pretrained one-square generators through the assembled canvas and achieves near-perfect validity with improved ELBO.

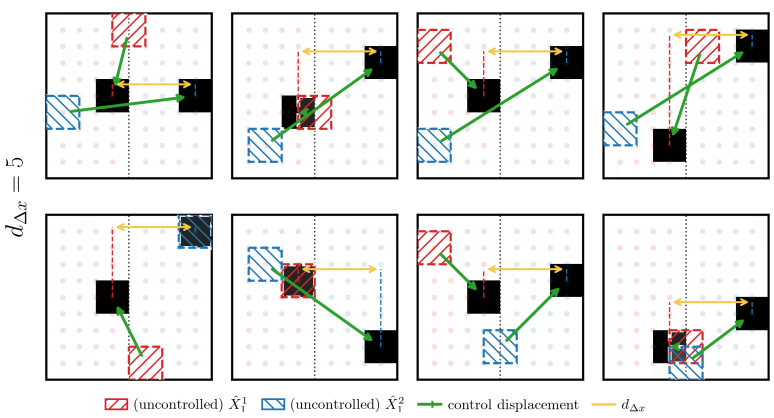


Figure 12: **Fixed-target coupled steering ($d_{\Delta x} = 5$).** Black squares show the controlled assembly, red and blue hatched squares the paired uncontrolled proposals, green arrows the induced corrections, and yellow arrows the target separation.

against the two single-process baselines. The failure for the two-square baseline becomes more pronounced for the distance constraint, where validity depends on the relative barycentres of the two squares. A unique control can improve the reward by moving mass in reward-favourable ways, but it does not inherit the component-level separation that makes the desired relation easy to express and control. This failure is therefore an inductive-bias failure, not just a capacity failure. The two-square generator represents valid objects, but its representation does not expose the component assignments needed for least-action steering.

Amortised coordination across design specifications. To test amortisation, we jointly train two agent-specific controllers with separate parameters, one acting on each frozen copy of the one-square reverse process, over $d_{\Delta x} \in \{3, 4, 5, 6\}$. The requested separation is represented by a four-way one-hot vector, broadcast over the spatial grid and concatenated to each controller input at every DDIM transition. Each controller outputs an additive residual to its corresponding frozen denoiser's noise prediction, thereby steering the two component processes jointly under the shared assembly-level reward. As shown in fig. 14, the amortised control steers the paired uncontrolled proposals into valid assemblies, while the achieved separations closely match the requested targets. The discrete adjoint-matching formulation is provided in Appendix C.2.2, and the control implementation is described below.

Control implementation. For the squares experiments, we attach two agent-specific convolutional control heads to two frozen copies of the same pretrained one-square denoiser. At each DDIM transition ℓ, the frozen denoiser provides Tweedie estimates $\hat{X}^i_{1|\ell}$ of the terminal component states, which are assembled as $\hat{Y}_{1|\ell} = \varphi(\hat{X}^1_{1|\ell}, \hat{X}^2_{1|\ell})$. The control head for component i receives the full two-agent configuration,

$$\mathrm{cat}_c\left[X^i_\ell,\ \hat{X}^i_{1|\ell},\ Y_\ell,\ \hat{Y}_{1|\ell},\ X^j_\ell,\ \hat{X}^j_{1|\ell}\right], \qquad Y_\ell = \varphi(X^1_\ell, X^2_\ell), \qquad j \neq i,$$

together with the embedding of the native diffusion timestep ζ_ℓ. Coordination therefore enters through joint conditioning: although the two heads have separate parameters and produce component-specific corrections, each control depends on both reverse processes and on their current and predicted assembly. Each head outputs an unbounded additive epsilon residual $\delta\epsilon^i_{\theta,\ell}$, which is added to the frozen denoiser prediction and induces the normalised DDIM control $w^i_{\theta,\ell}$. For the amortised variant, both heads are additionally conditioned on the requested separation $d_{\Delta x}$. The Tweedie features are recomputed at every transition and remain differentiable with respect to the component states, while the denoiser parameters remain frozen.

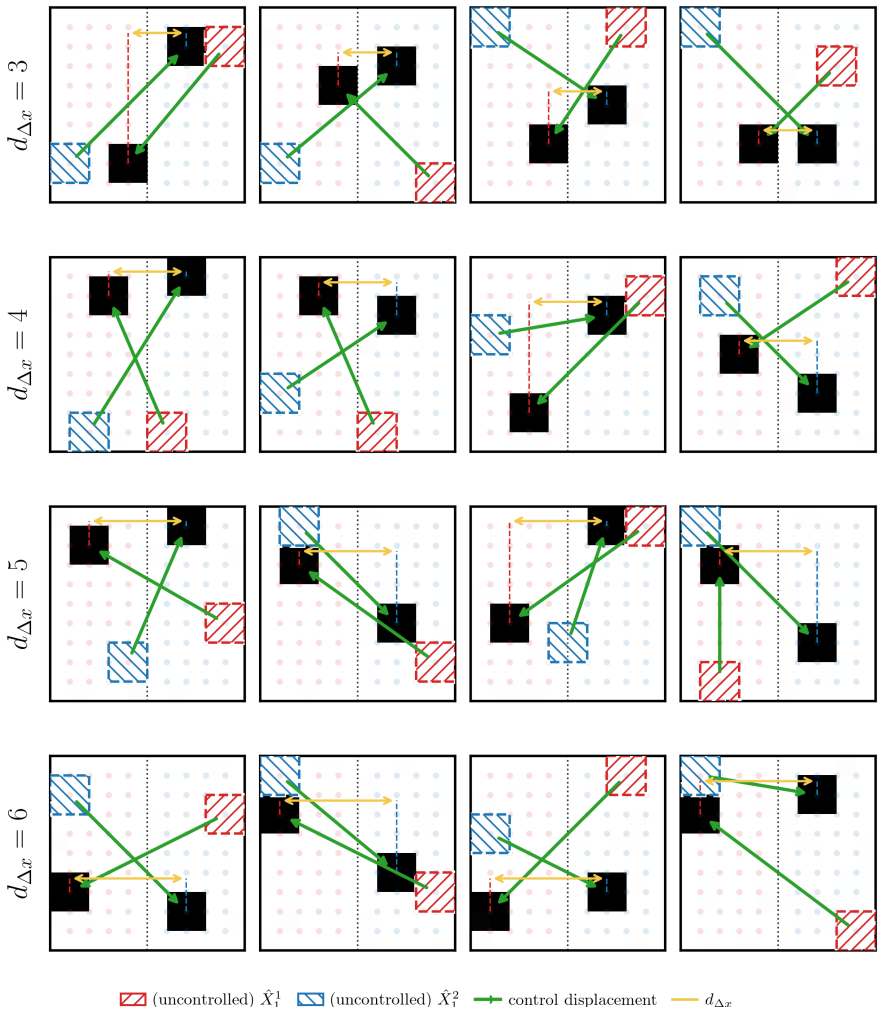


Figure 13: **Amortised coupled steering.** Samples from a single pair of agent-specific controls trained jointly over $d_{\Delta x} \in \{3, 4, 5, 6\}$. Black squares show the controlled assembly, red and blue hatched squares the paired uncontrolled proposals, and green arrows the induced corrections.

D.3 Source Separation

Key Takeaways: CMDS beats DPS. CMDS learns to coordinate components while minimising control cost, preserving their structural coherence. DPS has no equivalent mechanism and enforces the reward more aggressively, often at the expense of the individual components. **The control matters.** Providing the Tweedie-evaluated reward gradient directly to the control was important for effective coordination. **Performance under ambiguity.** For sufficiently ill-posed problems, CMDS recovers *a* plausible solution rather than necessarily *the* solution. Under stronger blur, this can even mean swapping component locations when that provides a lower-cost route to satisfying the reward. DPS, by contrast, tends to swap and deform components much more aggressively.

Source separation seeks to recover two or more latent signals from a combined observation. In many cases, datasets of individual source types and datasets of mixtures may each be available, while paired examples containing a mixture together with its constituent sources are unavailable or difficult to obtain. We investigate whether CMDS can coordinate samplers trained independently on the individual source distributions to explain an observed mixture without requiring paired source–mixture training data. The objective is to recover components whose assembly matches the observation while each component remains consistent with its respective pretrained distribution.

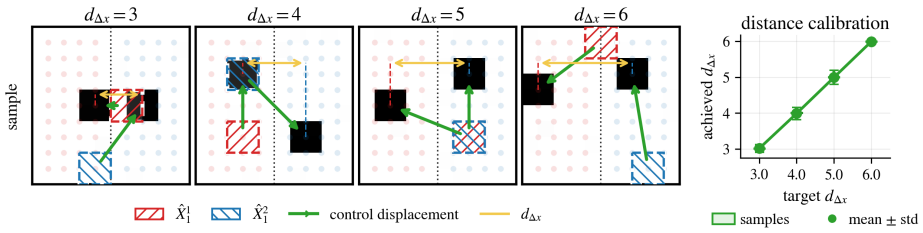


Figure 14: **Amortised assembly from copies of a single-square prior.** Two frozen copies of a generator pretrained only on one-square canvases are coordinated by a control and combined through a fixed left/right mask. Left: for each requested separation $d_{\Delta x} \in \{3, 4, 5, 6\}$, red and blue dashed outlines show paired uncontrolled terminal proposals obtained from the same noise, black squares show the controlled assembly, and green arrows show the induced corrections. Dots indicate feasible barycentre locations within the two masked halves. Right: the separation closely matches the requested target $d_{\Delta x}$, with points and error bars reporting the sample mean and standard deviation.

Task formalization. We consider $N = 2$ frozen pretrained samplers, G^1 and G^2, defining circle and square priors $\tilde{\pi}^1$ and $\tilde{\pi}^2$, respectively. The ground-truth terminal components are single-channel 16×16 images, $X_1^1, X_1^2 \in \{-1, +1\}^{1\times16\times16}$, with background value $+1$ and foreground value -1. The circle has radius 2.25 pixels and the square has side length 4 pixels; their locations are sampled independently within the image boundary. Their clean additive assembly is $Y_1 = \varphi(X_1) := X_1^1 + X_1^2 \in \{-2, 0, 2\}^{1\times16\times16}$.

Observation model. We observe $b = \mathrm{A}_\tau(Y_1)$, where A_τ denotes Gaussian convolution with a 5×5 kernel and blur standard deviation $\tau \in \{0.5, 1.0\}$. No additional observation noise is applied ($\varepsilon = 0$). Although the ground-truth sources are binary-valued, the blurred observations and generated sources are continuous-valued. Representative reconstructions under the two blur operators are shown in figs. 16 and 17.

Component priors. The circle and square priors are trained independently using 4096 procedurally sampled examples per component and epoch, defining the reference product law $\rho = \tilde{\pi}^1 \otimes \tilde{\pi}^2$. Each noise-prediction network projects a 32-dimensional sinusoidal time-step embedding and concatenates it spatially with the noisy image. The resulting tensor is processed by two 3×3, 64-channel convolutional layers with SiLU activations, a residual four-head spatial self-attention layer, and two further 3×3 convolutional layers producing a single-channel epsilon prediction. Both samplers use a linear 100-step diffusion schedule, $\beta_1 = 10^{-4}, \beta_{100} = 0.08$. They are trained for 1000 epochs using Adam, batch size 128, and a learning rate cosine-annealed from 10^{-3} to 10^{-5}, and are frozen.

Jointly controlled source processes. The reference process samples the circle and square independently. CMDS coordinates their reverse processes using a joint feedback control, producing

$$\hat{X}_1^\theta = \left(\hat{X}_1^{\theta,1}, \hat{X}_1^{\theta,2}\right), \qquad \hat{Y}_1^\theta = \varphi(\hat{X}_1^\theta) = \hat{X}_1^{\theta,1} + \hat{X}_1^{\theta,2}.$$

At each reverse step, the control outputs a separate additive epsilon-space control for each component process.

CMDS training. Following Algorithm 1, we use ℓ for rollout indices and ζ_ℓ for the corresponding native diffusion timesteps. At reverse transition ℓ, the control receives the current noisy states X_ℓ^1, X_ℓ^2, their Tweedie estimates $\hat{X}_{1|\ell}^1, \hat{X}_{1|\ell}^2$, the observation b, and the predicted observation

$$\hat{b}_\ell = A_\tau\left(\hat{X}_{1|\ell}^1 + \hat{X}_{1|\ell}^2\right).$$

It additionally receives the residual $b - \hat{b}_\ell$, the component-wise reward gradients, i.e., $\nabla_{\hat{X}_{1|\ell}^1} r_b$ and $\nabla_{\hat{X}_{1|\ell}^2} r_b$, and a sinusoidal embedding of ζ_ℓ. It outputs two additive epsilon-space controls, one for each reverse process. The control uses joint multi-scale processing with hidden width 32 and

one output head per component. We train the control by adjoint matching for $10{,}000$ optimisation steps. Each iteration samples 512 circle–square pairs, initial diffusion states, and rollout noise. Rollouts use 20 DDIM steps with $\eta = 1.0$, and adjoint matching uses 15 stochastic reverse steps. We use Adam with cosine learning-rate decay from 10^{-4} to 10^{-6}, gradient clipping at norm 1.0, and rejection of non-finite updates. An exponential moving average of the control parameters is maintained with decay 0.995.

Reward and control cost. The system-level reward is

$$r_b(y) = -\frac{1}{d}\,\|\mathrm{A}_\tau(y) - b\|_2^2\,, \qquad d = 16^2.$$

Training uses reward multiplier 1000 and the uniform quadratic control cost from the CMDS objective. The component-wise reward-gradient inputs are computed exactly through the known observation operator A_τ.

Source-separation control implementation. For source separation, we use a centralised joint control with a shared representation and two source-specific output heads, one for each frozen component diffusion model. At each DDIM transition ℓ, the shared control receives the current noisy states X_ℓ^1, X_ℓ^2, their Tweedie estimates $\hat{X}_{1|\ell}^1, \hat{X}_{1|\ell}^2$, the observation b, and the predicted observation

$$\hat{b}_\ell = \mathrm{A}_\tau\left(\hat{X}_{1|\ell}^1 + \hat{X}_{1|\ell}^2\right).$$

It is additionally conditioned on the measurement residual $b - \hat{b}_\ell$, the component-wise reward gradients, and the diffusion-time embedding ζ_ℓ. Coordination arises through the shared representation as both output heads have access to both reverse processes, their predicted clean sources, and their joint measurement mismatch before producing source-specific corrections. The two heads output additive epsilon residuals $\delta\epsilon_{\theta,\ell}^1$ and $\delta\epsilon_{\theta,\ell}^2$, which modify their respective frozen denoisers and induce the normalised DDIM controls $w_{\theta,\ell}^1$ and $w_{\theta,\ell}^2$. The reward-gradient features are detached, while the Tweedie features remain differentiable.

Baselines. The prior-only baseline samples the frozen component priors independently,

$$\hat{X}_1^1 \sim \tilde{\pi}^1, \qquad \hat{X}_1^2 \sim \tilde{\pi}^2,$$

and returns their assembly

$$\hat{Y}_1 = \hat{X}_1^1 + \hat{X}_1^2$$

without conditioning on b. We additionally adapt diffusion posterior sampling (DPS) (Chung et al., 2023) to the product of the two frozen component priors. At each reverse step, DPS evaluates the observation residual through the Tweedie estimates of both sources and applies the resulting component-wise data-fidelity gradients to their reverse updates. Writing $\hat{X}_{1|\ell}^i$ for the Tweedie estimate of component i at rollout index ℓ, its guidance is computed from

$$\nabla_{X_\ell^i}\left\|A_\tau\left(\hat{X}_{1|\ell}^1 + \hat{X}_{1|\ell}^2\right) - b\right\|_2\,, \qquad i \in \{1, 2\}.$$

In this sense, product-DPS provides a non-amortised inference-time analogue of coordination. The observation-dependent correction is recomputed for every mixture, whereas CMDS amortises this dependence into a learned joint feedback control.

Evaluation protocol. All methods are evaluated on the same 1024 procedurally generated circle–square instances with batch size 128. Every method receives the same ground-truth components X_1, observation b, and initial diffusion states. The controlled and prior-only rollouts share identical per-step noise, while DPS uses the same rollout-noise seed under its selected DDIM discretisation.

Evaluation metrics. We report three complementary properties of the recovered components. Clean-mixture reconstruction RMSE measures whether their assembly recovers the unblurred mixture:

$$\mathrm{RMSE}_{\mathrm{recon}} = \left[\frac{1}{d}\left\|\left(\hat{X}_1^1 + \hat{X}_1^2\right) - \left(X_1^1 + X_1^2\right)\right\|_2^2\right]^{1/2}.$$

In the following metric definitions, $\hat{X}_1^i$ denotes component i returned by the method being evaluated, with the method superscript suppressed, while X_1^i denotes the corresponding ground-truth component. This metric evaluates the recovered assembly but does not determine whether its structure has been assigned to the correct source. Source IoU compares the foreground mask of each recovered component with that of its corresponding labelled ground-truth component:

$$\mathrm{IoU}_{\mathrm{source}} = \frac{1}{2}\sum_{i=1}^{2}\frac{\left|M(\hat{X}_1^i)\cap M(X_1^i)\right|}{\left|M(\hat{X}_1^i)\cup M(X_1^i)\right|}.$$

The darkness representation and foreground mask are defined by $D(x) = \frac{1-\mathrm{clip}(x,-1,1)}{2}$, $M(x) = \{p : D(x)_p \geq 0.5\}$. Equivalently, a pixel is classified as foreground when its clipped model value is non-positive. Source IoU equals 1 when both recovered components occupy exactly the correct pixels, and 0 when neither overlaps its corresponding ground-truth component.

Shape distortion measures the difference between each recovered source and the nearest valid instance of its labelled prior shape, independently of location. Let $\mathcal{P}_1$ contain all 100 generator-valid circle placements and $\mathcal{P}_2$ all 169 generator-valid square placements. For each source, we define

$$\delta_i = \min_{T\in\mathcal{P}_i}\frac{\left\|D(\hat{X}_1^i)-T\right\|_2^2}{\|T\|_2^2}, \qquad \mathrm{Dist}_{\mathrm{shape}} = \frac{1}{2}\sum_{i=1}^{2}\delta_i.$$

Shape distortion is zero when each recovered component is a valid instance of its labelled prior shape at any generator-valid location. It increases when a component is blurred, deformed, missing, or has the wrong shape. Together, reconstruction RMSE, source IoU, and shape distortion measure assembly recovery, labelled source placement, and source-shape validity, respectively.

Additional evaluations. The aggregate comparisons over 1024 instances at $\tau = 0.5$ (low blur) and $\tau = 1.0$ (high blur) are shown in fig. 15 and fig. 4, respectively. CMDS achieves the lowest reconstruction RMSE and highest source IoU while retaining the low shape distortion of the frozen component priors. Representative reconstructions for the low-blur setting are shown in fig. 16; corresponding results under the stronger blur $\tau = 1.0$ are shown in fig. 17. Under stronger blur, the observation contains less information about the locations of the individual components, and the recovered circle and square can therefore exchange their ground-truth locations, as illustrated in the fourth row of fig. 17. Such decompositions preserve plausible component shapes and can produce a similar blurred assembly. The quadratic cost can consequently favour a nearby prior-consistent decomposition over the larger intervention required to recover the exact ground-truth placement.

D.4 PointMaze Collision Coordination

Key Takeaways: Multi-agent coordination can be amortised. A single control coordinates up to four frozen single-agent trajectory diffusion models, achieving high collision-free joint success and generalising to unseen start–goal regions. **Cross-agent communication must scale.** Factoring attention into temporal and cross-agent passes preserves the timing of predicted conflicts while reducing complexity from $O(N^2H^2)$ to $O(NH(N+H))$. **Tweedie reward gradients improve control.** Adding a learned time-dependent scaling of the reward gradient at the Tweedie estimate improves performance, with Pool 2 joint success ranging from 89.8% to 99.2%. **Robust regression stabilises adjoint matching.** Squared adjoint matching can be seed-sensitive in harder settings. The self-calibrated pseudo-Huber loss improves mean joint success in six of eight comparisons and reduces variance in the most unstable cases, but does not inherit the gradient identity or critical-point guarantee of squared adjoint matching.

This section provides detail on the protocol for the maze collision coordination experiment.

D.4.1 Task, State, and Reward

We first recall the setup of the task. Agent i generates a length-H trajectory $\tau_i = (p_i(0), \ldots, p_i(H-1)) \in \mathbb{R}^{H\times 2}$ using the same pretrained single-agent diffusion sampler G, instantiated independently

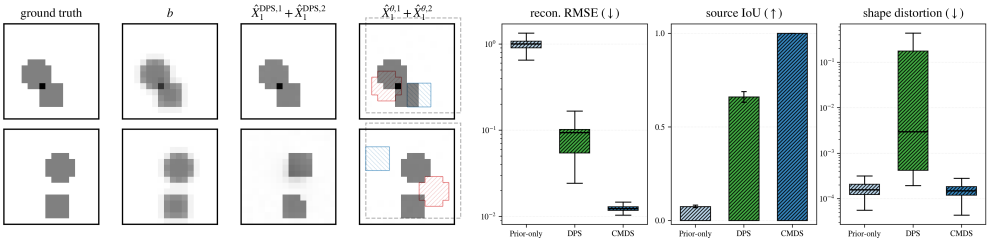


Figure 15: **Source separation from a blurred mixture ($\tau = 0.5$).** Each task contains a circle X_1^1 and a square X_1^2. We observe the blurred mixture $b = \mathrm{A}_{0.5}(X_1^1 + X_1^2)$ and recover the labelled sources. Left: two representative evaluation mixtures. The red and blue hatched shapes behind the CMDS reconstructions are the matched prior-only terminal sources. Right: results over 1024 evaluation mixtures for the frozen prior, DPS, and CMDS. Reconstruction RMSE measures whether the two recovered sources add up to the correct clean mixture. Source IoU measures whether each recovered object occupies the same pixels as its corresponding ground-truth object (↑): it is 1 for an exact match and 0 when neither recovered source overlaps its corresponding ground-truth source. The bars show the evaluation-set mean with 95% bootstrap confidence intervals computed from 10,000 resamples. Shape distortion measures whether source 1 still looks like a circle and source 2 like a square, irrespective of their location (↓). It is zero for an exact valid shape and increases when an object is blurred, deformed, missing, or has the wrong shape.

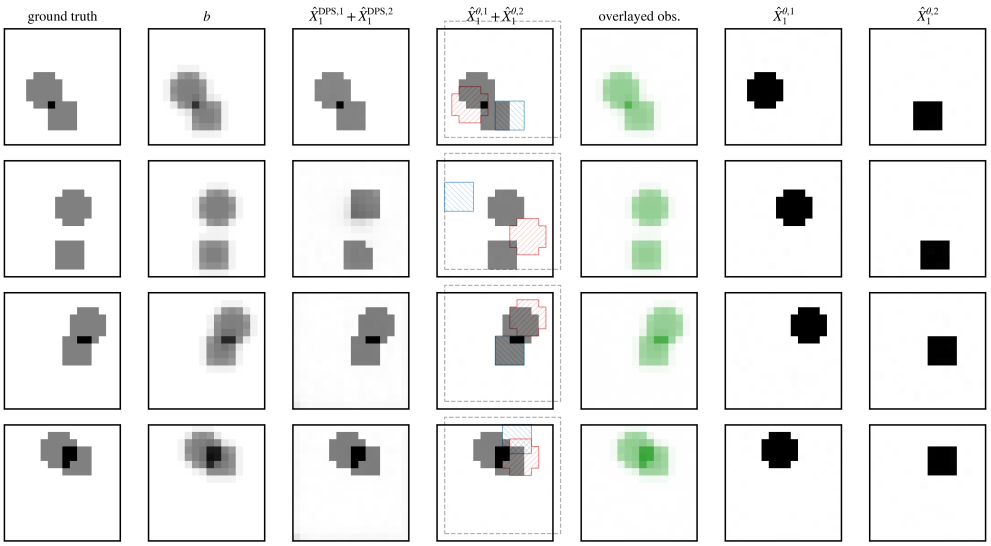


Figure 16: **Representative samples ($\tau = 0.5$).** Each row represents a different instance of the problem. The columns are: (i) the ground-truth mixture; (ii) the observation b; (iii) the DPS mixture reconstruction; (iv) the CMDS mixture reconstruction, with the matched prior-only circle and square shown using red and blue hatching, respectively; (v) the blurred CMDS reconstruction overlaid with the observation b; and (vi–vii) the CMDS estimates of the circle and square sources, respectively. In the measurement overlay, green indicates agreement between the predicted and observed mixtures, red indicates structure present only in the observation, and blue indicates structure present only in the predicted mixture.

for each agent. We use the discrete rollout convention of Algorithm 1. X_ℓ^i denotes agent i's native diffusion state, $\ell = 0, \ldots, K$ is the rollout index, and ζ_ℓ is the native diffusion time-step at transition ℓ. Thus X_0 is the initial noise and X_K is the terminal sample; the physical waypoint index $h \in \{0, \ldots, H-1\}$ is distinct from ℓ. The assembly operator stacks the N component states,

$$Y_\ell = \varphi(X_\ell) = (X_\ell^1, \ldots, X_\ell^N), \tag{105}$$

where φ is the identity on the concatenated state. We write $\hat{X}_{1|\ell}$ for the frozen prior's terminal-state estimate and $\hat{Y}_{1|\ell} = \varphi(\hat{X}_{1|\ell})$ for its assembly. Start and goal positions condition the reconstructed

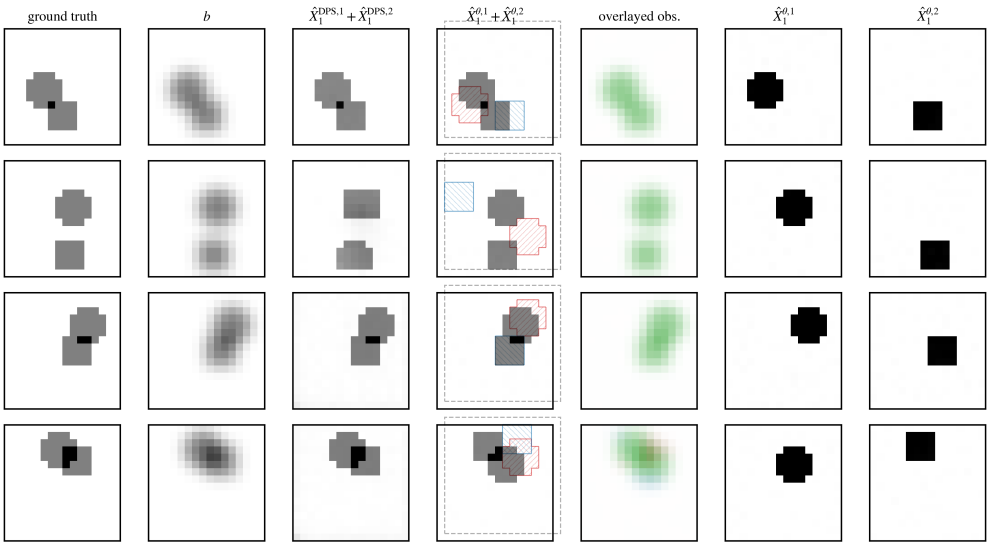


Figure 17: **Representative samples** ($\tau = 1.0$)**.** Each row represents a different instance of the problem. The columns are: (i) the ground-truth mixture; (ii) the observation b; (iii) the DPS mixture reconstruction; (iv) the CMDS mixture reconstruction, with the matched prior-only circle and square shown using red and blue hatching, respectively; (v) the blurred CMDS reconstruction overlaid with the observation b; and (vi–vii) the CMDS estimates of the circle and square sources, respectively. In the measurement overlay, green indicates agreement between the predicted and observed mixtures, red indicates structure present only in the observation, and blue indicates structure present only in the predicted mixture.

trajectory, which is learned as a residual modification to a straight-line base path interpolating between start and goal (Appendix D.4.4). Let $\kappa = \{(s_i, g_i)\}_{i=1}^N$ denote these fixed conditions and $\mathcal{R}_\kappa$ the residual-to-world reconstruction, so that

$$Z_\ell = \mathcal{R}_\kappa(Y_\ell), \qquad \hat{Z}_{1|\ell} = \mathcal{R}_\kappa(\hat{Y}_{1|\ell}).$$

Coordination is introduced by a shared control conditioned on these reconstructed joint trajectories. Goal reach is enforced structurally by the reconstruction and is not part of the reward.

D.4.2 Reward

For an assembled state Y, let

$$p_i(h; Y) := \big[\mathcal{R}_\kappa(Y)\big]_{i,h} \in \mathbb{R}^2$$

denote agent i's reconstructed world-space position at waypoint h. We suppress the dependence on Y below. The reward combines wall avoidance, inter-agent collision avoidance, and penalties on speed, acceleration, and path length. The terminal reward is evaluated at Y_K.

The path penalties use the waypoint grid $\mathcal{T} = \{0, \dots, H-1\}$. Collision penalties use a denser linearly interpolated grid $\mathcal{T}_{\mathrm{dense}}$, with $|\mathcal{T}_{\mathrm{dense}}| = 5(H-1)$ evaluation points. Positions at fractional waypoint indices are obtained by linear interpolation. We write $[a]_+ = \max\{a, 0\}$, so collision penalties vanish once the prescribed clearance is satisfied.

Wall penalty. Wall avoidance is enforced through the signed distance to the maze geometry. For each agent, we penalise only positions whose clearance from the nearest wall is smaller than the agent radius:

$$r_{\mathrm{wall}}^{(i)}(Y) = -\lambda_{\mathrm{wall}} \sum_{h \in \mathcal{T}_{\mathrm{dense}}} \big[r_{\mathrm{agent}} - \mathrm{sdf}(p_i(h))\big]_+^2 . \tag{106}$$

Here, $\mathrm{sdf}(\cdot)$ is positive in free space and negative inside an obstacle or outside the maze boundary, and $r_{\mathrm{agent}} = 0.12$ is the agent radius. The hinge therefore vanishes whenever the trajectory maintains at least one agent radius of wall clearance.

Inter-agent collision penalty. Collisions between agents are handled analogously by penalising insufficient pairwise separation along the densified trajectory. We define

$$r_{\text{pair}}(Y) = -\frac{\lambda_{\text{pair}}}{\binom{N}{2}} \sum_{i<j} \sum_{h \in \mathcal{T}_{\text{dense}}} \left[2r_{\text{agent}} + m - \|p_i(h) - p_j(h)\|\right]_+^2, \tag{107}$$

where m is an additional safety margin. The term is zero once two agents are separated by at least $2r_{\text{agent}} + m$, and the factor $\binom{N}{2}^{-1}$ averages the penalty over agent pairs.

Path regularisation. The collision terms constrain feasibility but do not by themselves favour short or smooth trajectories. We therefore include three per-agent regularisers on the raw waypoint grid $\mathcal{T}$: a one-sided penalty on excessive waypoint displacement, a discrete acceleration penalty, and total path length. These are

$$r_{\text{speed}}^{(i)}(Y) = -\lambda_{\text{speed}} \sum_{h=0}^{H-2} \left[\|p_i(h{+}1) - p_i(h)\| - v_{\max}\right]_+^2, \tag{108}$$

$$r_{\text{accel}}^{(i)}(Y) = -\lambda_{\text{accel}} \sum_{h=1}^{H-2} \|p_i(h{+}1) - 2p_i(h) + p_i(h{-}1)\|^2, \tag{109}$$

$$r_{\text{length}}^{(i)}(Y) = -\lambda_{\text{length}} \sum_{h=0}^{H-2} \|p_i(h{+}1) - p_i(h)\|. \tag{110}$$

The speed term is active only above $v_{\max}$, whereas acceleration and path length are penalised throughout the trajectory.

Total reward. Each agent receives its own wall and path penalties together with the shared inter-agent penalty:

$$\begin{aligned} r^{(i)}(Y) = r_{\text{wall}}^{(i)}(Y) + r_{\text{speed}}^{(i)}(Y) + r_{\text{accel}}^{(i)}(Y) \\ + r_{\text{length}}^{(i)}(Y) + r_{\text{pair}}(Y). \end{aligned} \tag{111}$$

For a world-space trajectory Z, let $r_\kappa^{\text{world}}(Z)$ denote this reward evaluated directly at positions $p_i(h) = Z_{i,h}$, so that $r_\kappa(Y) = r_\kappa^{\text{world}}(\mathcal{R}_\kappa(Y))$. The subscripted penalty weights are distinct from the outer SOC reward strength λ. We use

$$\left(\lambda_{\text{wall}}, \lambda_{\text{pair}}, \lambda_{\text{speed}}, \lambda_{\text{accel}}, \lambda_{\text{length}}\right) = (200, 400, 0, 1, 1),$$

with $m = 0.05$ and $v_{\max} = 0.15$. The Tweedie reward-gradient extension (Appendix D.4.5) uses the same reward definition, except that $\lambda_{\text{wall}} = 300$.

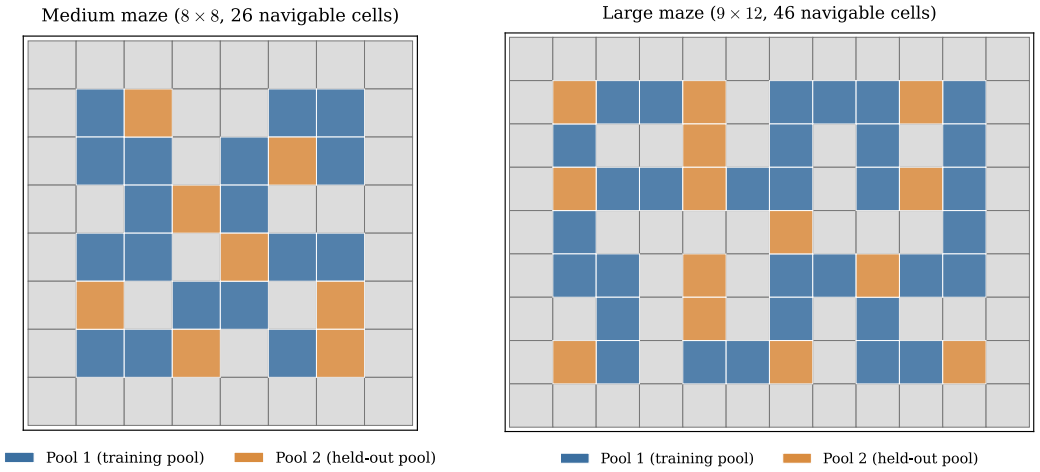
D.4.3 Data splits.

We use the medium and large PointMaze layouts (Fu et al., 2020). Each maze is mapped by a binary grid with 0 and 1 entries corresponding respectively to corridors and walls. The continuous PointMaze environment is obtained by fixing each grid entry to a physical size of 1.0×1.0. We use the medium and large layouts, whose cells are partitioned once, in advance, into disjoint training and test pools:

Layout	Grid	Navigable cells	Pool 1 cells	Pool 2 cells
Medium	8×8	26	18	8
Large	9×12	46	32	14

The Pool 1 and Pool 2 cells are disjoint, spatially interspersed across each maze and fixed across runs. The exact partition of cells is presented in fig. 18.

A training sample is obtained by sampling every start and goal cell independently and uniformly from the selected pool, adding continuous within-cell jitter, and retaining the draw only if:



(a) Medium (8×8): 26 navigable cells, comprising 18 training and 8 held-out cells.

(b) Large (9×12): 46 navigable cells, comprising 32 training and 14 held-out cells.

Figure 18: **Fixed PointMaze cell partitions.** Coordinates use zero-based $(\mathrm{row}, \mathrm{column})$ matrix indexing. The colours show the exact partition of cells between Pool 1 and Pool 2.

1. Every agent's uncontrolled path has a minimum length of 4.
2. The independently planned paths contain collisions.
3. Conflict-Based Search (CBS) (Sharon et al., 2015) finds a joint solution within a specified compute budget.

We report two held-out splits. Pool 1 contains unseen instances sampled from the training-cell pool; Pool 2 contains instances sampled from the disjoint test-cell pool. Every method and hyperparameter setting is evaluated on the same fixed set of 128 instances per pool.

D.4.4 Frozen Per-Agent Prior

Each layout uses a single-agent trajectory diffusion model, pretrained once and frozen throughout coordination training. We adopt the *Diffusion Veteran* configuration (Lu et al., 2025): a DiT-style denoiser with six adaLN-zero blocks, hidden width 256, four attention heads, and ≈ 7.3M parameters. The model is trained as a 100-step discrete-time DDPM with a linear schedule and mean-squared ϵ-prediction loss. This configuration reaches near-perfect (≈ 0.99) wall feasibility and closed-loop success on both layouts.

D.4.5 Control Architecture

Base control. The pretrained diffusion model represents each trajectory as a residual correction to the straight-line path between the start and goal. Before passing the trajectory to the control network, we reconstruct the corresponding absolute (x, y) positions in the maze. Cross-agent attention therefore operates on the agents' predicted physical positions rather than on the residual representation used by the diffusion model. At reverse transition ℓ, it receives the joint trajectory Z_ℓ, the reconstructed Tweedie estimate $\hat{Z}_{1|\ell}$, the start–goal conditions κ, and the native diffusion time-step ζ_ℓ. The trajectory and Tweedie estimate are concatenated along the channel axis and projected to width 344, with sinusoidal embeddings of diffusion time and physical waypoint position. An initial temporal self-attention block processes each agent's trajectory separately. Two axial blocks then alternate temporal self-attention within each agent, cross-agent attention at each shared waypoint, and a feed-forward layer. Both blocks retain the full waypoint resolution and use four attention heads, a feed-forward multiplier of 4, residual connections, and layer normalisation. Cross-agent attention includes a learned geometric bias. For each agent pair, an eight-dimensional feature collects relative position (2), distance (1), forward-differenced relative velocity (2), closing speed (1), and relative goal offset (2). An MLP with dimensions $8 \to 32 \to 4$ maps these features to an additive bias for

each attention head, applied through a custom scaled-dot-product implementation. Separate output heads read the local and communication representations to produce the epsilon residual $\delta\epsilon_{\theta,\ell}$. Both heads and the final layer of the geometric-bias MLP are zero-initialised, so the base control initially leaves the pretrained sampler unchanged. The network has ~ 4.3M parameters. Rather than applying attention jointly over all $N \times H$ agent–waypoint tokens, we factor it along the two natural axes of the trajectory tensor (Ho et al., 2019): temporal attention operates over waypoints within each agent, while cross-agent attention operates over agents at each shared waypoint. This reduces the per-block attention complexity from $O(N^2H^2)$ to $O(NH(N+H))$. For $H = 50$ and $N = 4$, this gives approximately 3.7 times fewer attention-pair evaluations. Temporal attention is batched over agents, and cross-agent attention is batched over waypoints. Shared weights and attention along the agent axis make the control equivariant to a consistent permutation of agents and their conditions (Lee et al., 2019).

Tweedie reward gradient extension. CMDS+TRG augments the base axial control with the joint reward gradient $g_\ell = \lambda^\star \, \nabla_Z r_\kappa^{\mathrm{world}}(Z)\big|_{Z=\hat{Z}_{1|\ell}}$, evaluated at the reconstructed Tweedie estimate $\hat{Z}_{1|\ell}$ (Denker et al., 2024; Venkatraman et al., 2024), where $\lambda^\star > 0$ is a fixed scale, distinct from the outer SOC reward strength λ. This is normalised to unit root-mean-square magnitude and clamped element-wise to $[-3, 3]$ to give $\hat{g}_\ell$ ($\lambda^\star$ cancels under this normalisation, so its value does not affect $\hat{g}_\ell$). This is combined additively with the base network:

$$\delta\epsilon_{\theta,\ell} = \mathrm{NN}_\theta\Big(Z_\ell, \hat{Z}_{1|\ell}, \hat{g}_\ell, \kappa, \zeta_\ell\Big) - s_\theta(\zeta_\ell)\, \hat{g}_\ell, \tag{112}$$

where $\delta\epsilon_{\theta,\ell}$ is the additive epsilon residual, inducing the normalised control $w_{\theta,\ell} = (\gamma_\ell/\varsigma_\ell)\delta\epsilon_{\theta,\ell}$ as in eq. (97). Here NN_θ is the same control described above, applied to a widened, three-channel input – the reconstructed noisy state, the reconstructed Tweedie estimate, and $\hat{g}_\ell$, concatenated along the channel axis (so it can condition on the reward gradient rather than only react to its additive contribution) – and $s_\theta(\zeta_\ell) = 1 + \tanh(\mathrm{MLP}_\theta(\zeta_\ell))$ is a single learned, time-dependent, agent-shared scalar gate (a function of the diffusion time-step only, not of $\hat{Z}_{1|\ell}$ or $\hat{g}_\ell$). The gate MLP is zero-initialised so that $s_\theta(\zeta_\ell) = 1$ at the start of training. This adds only $1{,}777$ parameters over the base control ($4{,}304{,}953$ total). We use this full configuration (both the network conditioning and the gated additive term) throughout; restricting to either component alone, or removing the network's access to $\hat{Z}_{1|\ell}$ or $\hat{g}_\ell$, trades off differently across cells but does not change the qualitative picture and is left to future ablation work.

D.4.6 Training Objectives

We apply the adjoint matching coordination objective introduced in section 2.1 to the stacked N-agent state. Adjoint matching rolls out the current controlled trajectory along the reverse grid with no gradient tracking. Starting from the terminal cost gradient $a_K = \nabla_{X_K}[-\lambda r_\kappa(Y_K)]$, it propagates an adjoint variable backward one DDIM step at a time. This backward pass replays individual controlled transitions rather than retaining the computational graph of the full multi-step sampler. The control is then trained to match the correction implied by the next-state adjoint $a_{\ell+1}$. All headline results use the robust regression variant described in Appendix D.4.7. The batch sizes keep the number of per-agent trajectories approximately constant at 512 per optimiser step. At low signal-to-noise ratios, the DiT prior can also produce excessively large $\hat{X}_{1|\ell}$ estimates. Following Janner et al. (2022), we clip $\hat{X}_{1|\ell}$ before converting it back to the equivalent noise prediction used in the control-penalty decomposition.

The adjoint matching configuration is summarised below.

Setting	Value
Optimiser steps	15,000
Initial learning rate	10^{-4} (Medium N=2, Medium N=4, Large N=2); 3×10^{-4} (Large N=4)
Learning-rate schedule	Cosine decay to 10^{-5}
EMA decay	0.9995
Clipping	Gradient norm 0.5, elementwise $\hat{X}_{1\|\ell}$ magnitude 4
Batch size	256 for $N = 2$, 128 for $N = 4$
Training instances	4096 joint instances for $N = 2$; 2048 for $N = 4$

The table above is CMDS's recipe. CMDS+TRG converges in fewer steps and uses a correspondingly slower schedule: 7,000 optimiser steps, initial learning rate 10^{-5} with cosine decay to 10^{-6}, and reward wall-weight $\lambda_{\text{wall}} = 300$. Batch size, training instances, EMA decay, and gradient/$\hat{X}_{1|\ell}$ clipping are shared between both variants.

D.4.7 Robust AM: pseudo-Huber versus squared ℓ_2

Motivation. Adjoint matching is an on-policy stochastic regression problem whose inputs and targets depend on the current control. Because adjoints are propagated through Jacobians evaluated along stochastic, control-dependent trajectories, finite-batch gradient estimates can have high variance despite being unbiased in expectation (Domingo-Enrich et al., 2025). The squared matching loss can amplify this variability because its gradient grows with the adjoint residual. For collision rewards, early trajectories often contain persistent agent collisions or deep wall penetrations, which can dominate minibatch adjoints and make optimisation unstable and strongly seed-dependent. This motivates a modification of adjoint matching's regression objective.

Objective. Using the rollout convention of Algorithm 1, with the frozen-rollout superscript suppressed, we write the joint matching residual as

$$e_{\theta,\ell} := \operatorname{col}_{i=1}^{N} \left[w^i_{\theta,\ell}(\operatorname{stopgrad}(X_\ell)) + \varsigma_\ell \operatorname{stopgrad}\big(a^i_{\ell+1}\big) \right] \in \mathbb{R}^D.$$

Here D is the dimension of the stacked residual after vectorisation ($D = 2NH$ for the full $H \times 2$ representation). This is the residual appearing in eq. (104). The normalised control is defined in eq. (97), and the adjoint target is obtained from the terminal condition eq. (100) and recursion eq. (101). To reduce the influence of rare high-norm residuals induced by these recursively propagated targets, we use a pseudo-Huber, or Charbonnier loss (Huber, 1992; Charbonnier et al., 1994; Barron, 2019). For a residual vector $e \in \mathbb{R}^D$, let $\operatorname{rms}(e) = \|e\|/\sqrt{D}$ be its root-mean-square magnitude and define

$$\rho_{\delta_\ell}(e) := D\delta_\ell^2 \left(\sqrt{1 + \frac{\|e\|^2}{D\delta_\ell^2}} - 1 \right). \tag{113}$$

The resulting robust matching objective is

$$\mathcal{L}_{\text{CMDS-PH}}(\theta) := \mathbb{E}\left[\sum_{\ell \in \mathcal{K}} \rho_{\delta_\ell}(e_{\theta,\ell}) \right]. \tag{114}$$

For $\|e\| \ll \sqrt{D}\delta_\ell$,

$$\rho_{\delta_\ell}(e) = \frac{1}{2}\|e\|^2 + O\left(\frac{\|e\|^4}{D\delta_\ell^2} \right),$$

so $\nabla_e^2 \rho_{\delta_\ell}(0) = I$ and the loss retains the local curvature of squared matching. For $\|e\|/(\sqrt{D}\delta_\ell) \to \infty$, it instead satisfies $\rho_{\delta_\ell}(e) = \sqrt{D}\delta_\ell \|e\| + O(1)$, and

$$\nabla_e \rho_{\delta_\ell}(e) = \frac{e}{\sqrt{1 + \|e\|^2/(D\delta_\ell^2)}}, \qquad \|\nabla_e \rho_{\delta_\ell}(e)\| \le \sqrt{D}\,\delta_\ell. \tag{115}$$

Because both the rollout and adjoint target are stopped in eq. (104),

$$\nabla_\theta \rho_{\delta_\ell}(e_{\theta,\ell}) = D_\theta w_{\theta,\ell}^\top \nabla_e \rho_{\delta_\ell}(e_{\theta,\ell}).$$

Thus, pseudo-Huber bounds the influence of an individual target in residual space, although the parameter gradient can still be amplified by the control Jacobian $D_\theta w_{\theta,\ell}$.

Self-calibration of the pseudo-Huber loss. Adjoint residual scales differ substantially across DDIM transitions, so a single global threshold would constrain some steps too aggressively and others too weakly. We first run a short calibration phase with the squared DDIM matching objective eq. (104) and record the per-instance RMS residual $\mathrm{rms}(e^{(b)}_{\theta,\ell})$ at each transition ℓ. We then set

$$\delta_\ell = \max\Big\{\delta_{\min}, Q_q\Big(\{\mathrm{rms}(e^{(b)}_{\theta,\ell})\}_{b\in\mathcal{D}_{\mathrm{cal}}}\Big)\Big\}. \tag{116}$$

Here, Q_q is a fixed empirical quantile over the calibration set $\mathcal{D}_{\mathrm{cal}}$; we freeze δ_ℓ for the remainder of training. The calibration hyperparameters are fixed before the multi-seed comparison and shared across mazes and agent counts. The underlying SOC objective remains eq. (72); only the squared regression surrogate eq. (104) is replaced. The gradient identity and critical-point guarantee for the basic squared objective eq. (6), established by Domingo-Enrich et al. (2025), do not automatically extend to eq. (114). This modified regression objective is studied empirically and need not preserve the minimiser of squared adjoint matching; its connection to the original stochastic-control objective remains an open question. Table 9 reports the full squared-ℓ_2-versus-pseudo-Huber comparison, held-out joint success, mean $\pm$ population standard deviation across 10 independent training seeds per (maze, N) cell. All runs use identical hyperparameters tuned for ℓ_2-AM, identical control architecture (Appendix D.4.5), and evaluation protocol. The pseudo-Huber matching loss (113) generally improves mean success and substantially reduces variance in challenging settings, such as the Large maze with 4 agents.

	Medium ($N=2$)		Medium ($N=4$)		Large ($N=2$)		Large ($N=4$)	
Matching loss	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑
Squared ℓ_2	0.940 ±0.020	0.884 ±0.048	**0.806** ±0.155	0.667 ±0.151	0.833 ±0.041	0.792 ±0.042	0.723 ±0.157	0.606 ±0.152
Pseudo-Huber	**0.988** ±0.012	**0.933** ±0.032	0.805 ±0.098	0.665 ±0.169	**0.907** ±0.037	**0.833** ±0.045	**0.873** ±0.027	**0.744** ±0.036

Table 9: **Adjoint matching training-seed variance and its mitigation.** Held-out joint success, mean $\pm$ population standard deviation across 10 independent training seeds per (maze, N) cell.

D.4.8 Training Variance

We report the final CMDS+TRG training variance. We train CMDS+TRG with 5 different seeds for each setting and report the mean and standard deviation of joint success on each pool.

	Medium ($N=2$)		Medium ($N=4$)		Large ($N=2$)		Large ($N=4$)	
Method	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑	Pool 1 ↑	Pool 2 ↑
CMDS+TRG	**0.998** ±0.003	**0.983** ±0.015	**0.984** ±0.005	**0.947** ±0.006	**0.973** ±0.009	**0.906** ±0.015	**0.975** ±0.006	**0.969** ±0.005

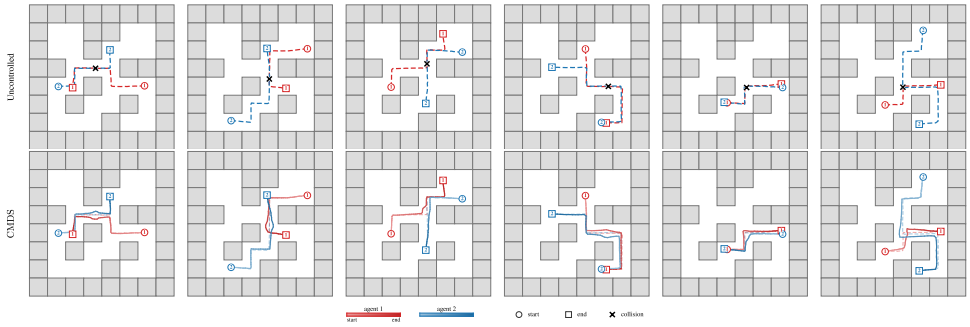
Table 10: **CMDS+TRG training variance.** Held-out joint success rate (Pool 1: train-cell instances, Pool 2: instances from entirely held-out maze cells), mean $\pm$ population standard deviation across 5 independent training seeds, evaluated on 128 held-out instances per split.

D.4.9 Baseline Suite and Protocol Matching

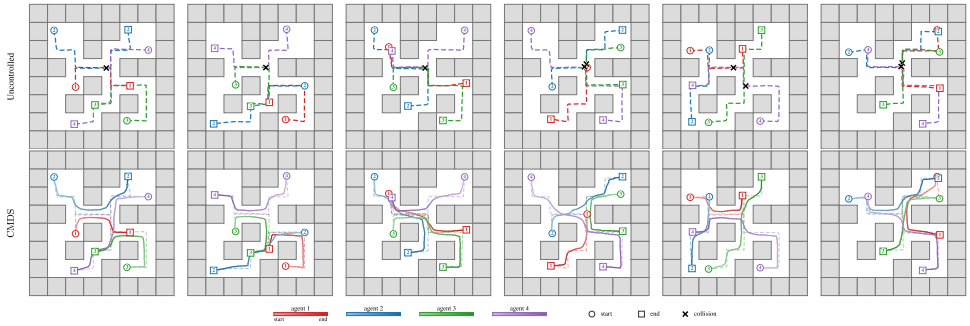
We compare against independent sampling, Best-of-256 reward selection, FK-Steering, and DPS test-time reward guidance. All baselines use the same frozen single-agent prior, trajectory representation, reward, agent radius $r_{\mathrm{agent}} = 0.12$, $\hat{X}_{1|\ell}$-clip of 4, and fixed evaluation sets of 128 instances per split. DPS and FK-Steering are tuned for each task.

D.4.10 Qualitative Examples

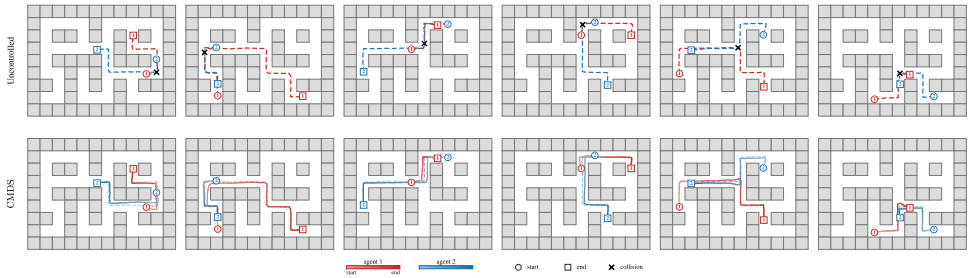
Figures 19I to 19IV give 6 held-out examples per (maze, N) cell, with the highest variation of the trajectory before and after introducing the control.



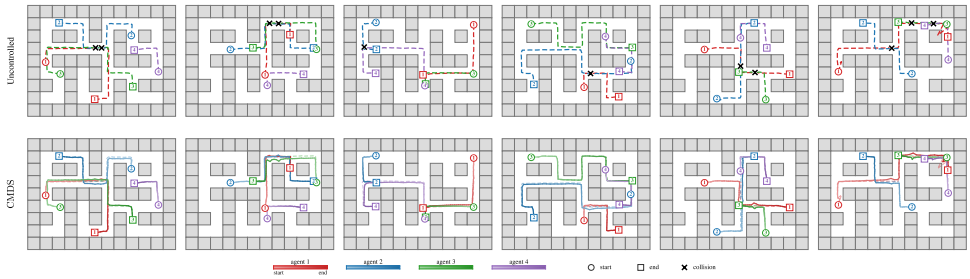
(I) Medium PointMaze ($N = 2$).



(II) Medium PointMaze ($N = 4$).



(III) Large PointMaze ($N = 2$).



(IV) Large PointMaze ($N = 4$).

Figure 19: **Qualitative maze-coordination examples.** For each maze and agent count, the top row is the uncontrolled trajectory (dashed) and the bottom row is the trajectory produced with the trained CMDS control (solid), for the same 6 held-out instances and the same initial noise.

D.5 Multi-Arm KUKA Coordination from Frozen Single-Arm Priors

Key Takeaways: Frozen energy-based priors can be coordinated. CMDS learns a shared control over independent seven-joint KUKA priors while keeping their pretrained weights fixed. Adjoint training differentiates the reverse dynamics through the learned energy using input Hessian–vector products, evaluated one transition at a time. **A two-arm controller generalises without retraining.** Trained to coordinate two frozen single-arm priors, CMDS provides zero-shot transfer to shifted or enlarged obstacles, wider base spacing, and three to five arms. It achieves higher mean strict success than the evaluated guidance baselines in all these transfer settings.

D.5.1 Task, representation, and data

The KUKA task follows Luo et al. (2024) and consists of two robotic arms whose single-arm trajectory priors are independently pretrained for obstacle-avoiding motion planning. Arm i generates joint angles $q^i = (q_1^i, \dots, q_H^i) \in \mathbb{R}^{H\times 7}$, with $H = 56$. All arms share a path-progress coordinate, and q_1^i, q_H^i are fixed to the requested start and goal. The planning problem is geometric. The paths have no physical time parameterisation, velocity limits, or execution guarantee. With the saved prior-normalisation bounds $q_{\mathrm{lo}}, q_{\mathrm{hi}}$, the normalised full path is $x = 2(q - q_{\mathrm{lo}})/(q_{\mathrm{hi}} - q_{\mathrm{lo}}) - 1$, with component-wise operations and an affine inverse. Only the $H - 2$ interior knots are sampled; endpoints are reinserted whenever the full trajectory is reconstructed. We denote the joint native state by $X = (X^1, \dots, X^N)$, where $X^i \in \mathbb{R}^{(H-2)\times 7}$ contains arm i's normalised interior coordinates. For fixed task conditions κ, comprising the endpoints, arm bases, and obstacles, the reconstructed assembly is $Y = \varphi(X) = q = (q^1, \dots, q^N)$; the dependence of φ on κ is suppressed. The two-arm state has $14(H - 2) = 756$ free coordinates. For N arms, $D = 7N(H - 2)$; norms on the native state and control are Euclidean after vectorisation, without averaging over arms or coordinates.

The arm bases are placed at $(-0.5, 0, 0)$ and $(0.5, 0, 0)$ metres with identity base rotations, above a shared table and among five axis-aligned box obstacles of half-extent 0.22 m. Single-arm priors receive obstacle centres expressed in each arm's local base coordinates; the controller and collision reward use world-frame (i.e. shared coordinate system) forward kinematics and the actual obstacle extents. This distinction matters under geometric transfer. The pretrained conditioning interface remains fixed, while the joint control observes the transformed scene.

Priors. We instantiate the priors from the pretrained checkpoints of Luo et al. (2024). The Dual-arm checkpoint is used only for evaluation and comparison with CMDS[2].

Controller-training and evaluation populations. We partition the 2,500 released training layouts into 2,250 controller-training layouts and 250 controller-validation layouts. Extracting 32 complete start–goal configuration pairs per layout gives 72,000 training queries. We retain the 50,212 queries whose start and goal configurations satisfy joint geometrical limits and pass the evaluation collision checks.

We select two start–goal configuration pairs from each of 100 validation layouts and 300 test layouts, using a fixed random seed. We discard pairs whose start or goal configurations violate joint limits or contain collisions. This leaves 134 validation tasks and 413 test tasks, with the test tasks spanning 264 layouts. All methods are evaluated on the same tasks. Tasks are selected before running any planner; valid endpoints do not guarantee that a collision-free path connects them. The test set was used during development and is therefore not an untouched final test set.

Zero-shot transfer. Each transformation below is applied to all 600 source test queries before endpoint filtering. Obstacle perturbations are deterministic per layout and shared across methods. All controller and prior weights, guidance strengths, and success predicates are held fixed. The resulting number of tasks is detailed in table 11.

For wider bases, joint-space endpoints and world-frame obstacle positions remain fixed, so world-space end-effector endpoints change. The three-arm construction preserves the native pair and assigns the third arm the first arm's endpoints from the next source query in the same layout. CMDS is architecturally compatible with an arbitrary number of arms without retraining.

[2] https://github.com/devinluo27/potential-motion-plan-release

Population	Construction	Evaluation tasks
In-domain	Released dual-arm scene and endpoints	413
Shifted obstacles	Horizontal Gaussian shifts, standard deviation 0.10 m	191
Larger obstacles	Each box half-extent multiplied by 1.2	121
Wider bases	Base positions moved to $x = \pm 0.6$ m	208
Three arms	Third base at $(0, 0.8, 0)$ m, identity rotation	172

Table 11: **KUKA evaluation populations.**

Transfer from two to five arms. For the arm-count comparison in the right-hand block of table 3, we select twenty start–goal tasks from each of the 300 test layouts, giving 6,000 tasks. Selection is fixed before running any planner. We keep the original two arms and obstacles unchanged, then add arms at $(0, 0.8, 0)$, $(0, -0.8, 0)$, and $(0.8, 0.8, 0)$ metres, all with the same base orientation as the original arms. Within each layout, we pair consecutive tasks in the fixed selection order. Arms three and four use the start and goal configurations of the partner task's first and second arms. Arm five uses the first arm's configurations from the first remaining task in lexicographic order, excluding the current task, its partner, and the two tasks used in the original evaluation.

Checking joint limits and endpoint collisions leaves 4,161, 1,513, 524, and 209 tasks for two, three, four, and five arms, respectively. We compare all arm counts on the same 209 tasks whose endpoints pass these checks in every setting. These results are conditional on endpoint feasibility for all four counts; they do not describe the full retained population at each count or certify that a feasible connecting path exists.

D.5.2 Energy-based priors and discrete adjoint matching

Energy parameterisation. The released temporal network f_ψ defines the ℓ_2 energy parameterisation of Du et al. (2023); Luo et al. (2024). For a normalised full path x, diffusion index t, and scene conditioning c, the noise predictor is

$$E_\psi(x, t, c) = \tfrac{1}{2}\|f_\psi(x, t, c)\|_2^2, \qquad \epsilon_\psi(x, t, c) = \nabla_x E_\psi(x, t, c). \tag{117}$$

The positive sign is the release's epsilon-prediction convention; it should not be confused with the sign of a score. Classifier-free guidance with weight $\omega_{\mathrm{cfg}} = 2$ gives $\epsilon_\psi^{\mathrm{cfg}} = \epsilon_\psi^{\varnothing} + \omega_{\mathrm{cfg}}(\epsilon_\psi^{c} - \epsilon_\psi^{\varnothing})$, equivalently the input gradient of the same linear combination of energies. These are learned trajectory energies, distinct from the geometric collision reward defined below.

Reverse transition and control. We use the state and transition-index convention of Algorithm 1, but retain $L = 10$ for the number of DDIM transitions in this subsection, since K denotes the number of initial joint trajectory candidates per task in the KUKA experiments. The rollout runs from the initial noise state X_0 to the terminal sample X_L, with transitions indexed by $\ell = 0, \dots, L-1$ and native diffusion timesteps $\zeta_\ell = 90 - 10\ell$. Thus L takes the role of the transition count K in Algorithm 1. Let $\bar{\alpha}_t$ denote the checkpoint's cumulative noise-schedule coefficient at native timestep t, and define

$$\bar{\alpha}_{\mathrm{src},\ell} = \bar{\alpha}_{\zeta_\ell}, \qquad \bar{\alpha}_{\mathrm{dst},\ell} = \begin{cases} \bar{\alpha}_{\zeta_{\ell+1}}, & 0 \le \ell < L-1, \\ 1, & \ell = L-1. \end{cases}$$

The frozen noise prediction is evaluated on each arm's full normalised path with fixed endpoints, then restricted to its free interior. We write the concatenated prediction as $\epsilon_{\mathrm{b},\ell}(X_\ell) = (\epsilon_{\mathrm{b},\ell}^i(X_\ell^i))_{i=1}^N$, with evaluation at ζ_ℓ, scene conditioning, and classifier-free guidance included. The reference tran-

sition first clips its clean estimate and recomputes the corresponding epsilon:

$$\begin{aligned}
\hat{X}^{\mathrm{clip}}_{1|\ell} &= \mathrm{clip}_{[-1,1]}\left(\frac{X_\ell - \sqrt{1-\bar{\alpha}_{\mathrm{src},\ell}}\,\epsilon_{\mathrm{b},\ell}(X_\ell)}{\sqrt{\bar{\alpha}_{\mathrm{src},\ell}}}\right),\\
\widetilde{\epsilon}_{\mathrm{b},\ell}(X_\ell) &= \frac{X_\ell - \sqrt{\bar{\alpha}_{\mathrm{src},\ell}}\,\hat{X}^{\mathrm{clip}}_{1|\ell}}{\sqrt{1-\bar{\alpha}_{\mathrm{src},\ell}}},\\
\varsigma_\ell^2 &= \eta^2 \frac{1-\bar{\alpha}_{\mathrm{dst},\ell}}{1-\bar{\alpha}_{\mathrm{src},\ell}}\left(1-\frac{\bar{\alpha}_{\mathrm{src},\ell}}{\bar{\alpha}_{\mathrm{dst},\ell}}\right),\\
\mu_{\mathrm{b},\ell}(X_\ell) &= \sqrt{\bar{\alpha}_{\mathrm{dst},\ell}}\,\hat{X}^{\mathrm{clip}}_{1|\ell} + \sqrt{1-\bar{\alpha}_{\mathrm{dst},\ell}-\varsigma_\ell^2}\,\widetilde{\epsilon}_{\mathrm{b},\ell}(X_\ell).
\end{aligned} \tag{118}$$

For the learned-control sampler, $\eta = 1$, so $\varsigma_\ell > 0$ for $0 \le \ell < L-1$, whereas $\varsigma_{L-1} = 0$. For an epsilon correction $\delta\epsilon_{\theta,\ell}(X_\ell)$, define on the stochastic transitions $0 \le \ell < L-1$

$$\begin{aligned}
\gamma_\ell &= \sqrt{1-\bar{\alpha}_{\mathrm{dst},\ell}-\varsigma_\ell^2} - \sqrt{\frac{\bar{\alpha}_{\mathrm{dst},\ell}}{\bar{\alpha}_{\mathrm{src},\ell}}}\sqrt{1-\bar{\alpha}_{\mathrm{src},\ell}},\\
w_{\theta,\ell}(X_\ell) &= \frac{\gamma_\ell}{\varsigma_\ell}\delta\epsilon_{\theta,\ell}(X_\ell),\\
\mu^\theta_\ell(X_\ell) &= \mu_{\mathrm{b},\ell}(X_\ell) + \varsigma_\ell w_{\theta,\ell}(X_\ell),\\
X_{\ell+1} &= \mu^\theta_\ell(X_\ell) + \varsigma_\ell \Xi_\ell, \qquad \Xi_\ell \sim \mathcal{N}(0, I_D).
\end{aligned} \tag{119}$$

At the deterministic final transition, no learned correction is applied:

$$X_L = \mu_{\mathrm{b},L-1}(X_{L-1}), \qquad w_{\theta,L-1} \equiv 0.$$

The final control is set to zero by convention, not computed through division by ς_{L-1}. Unlike the modified endpoint in Appendix C.2.2, KUKA retains the native deterministic endpoint; it contributes neither control energy nor a matching term.

Costates and matching targets. The implemented per-agent-normalised control objective is

$$\mathcal{J}_{\mathrm{KUKA}}(\theta) = \mathbb{E}\left[\frac{1}{N}\sum_{\ell=0}^{L-2}\frac{1}{2}\|w_{\theta,\ell}(X_\ell)\|_2^2 - \beta r_\kappa(\varphi(X_L))\right], \qquad \beta = 40. \tag{120}$$

For fixed N, multiplying this objective by N gives the equivalent unnormalised joint objective with terminal reward coefficient $\lambda = N\beta$. The adjoints below correspond to this rescaled objective. For a rollout under the fixed controller snapshot $\bar{\theta} = \mathrm{stopgrad}(\theta)$, suppress the rollout superscript and initialise the adjoint at the terminal reward, then propagate backward through every transition:

$$\begin{aligned}
a_L &= -N\beta \nabla_{X_L} r_\kappa(\varphi(X_L)),\\
a_{L-1} &= \left[D_X \mu_{\mathrm{b},L-1}(X_{L-1})\right]^{\mathsf{T}} a_L,\\
a_\ell &= \nabla_X\left[a_{\ell+1}^{\mathsf{T}}\mu^{\bar{\theta}}_\ell(X) + \tfrac{1}{2}\|w_{\bar{\theta},\ell}(X)\|_2^2\right]\Big|_{X=X_\ell}, \quad \ell = L-2,\dots,0.
\end{aligned} \tag{121}$$

The next-state adjoint is held fixed during each state differentiation. The deterministic final transition remains in adjoint propagation, but has no control-cost derivative. The squared matching objective uses the *next-state* adjoint, with both rollout states and targets stopped during controller fitting. With all stochastic transitions included, $\mathcal{K} = \{0,\dots,L-2\}$, we use

$$\begin{aligned}
e^i_{\theta,\ell} &= w^i_{\theta,\ell}(\mathrm{stopgrad}(X_\ell)) + \varsigma_\ell\,\mathrm{stopgrad}(a^i_{\ell+1}), \quad \ell \in \mathcal{K},\\
\mathcal{L}_{\mathrm{AM}}(\theta) &= \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\sum_{\ell\in\mathcal{K}}\frac{1}{2}\|e^i_{\theta,\ell}\|_2^2\right].
\end{aligned} \tag{122}$$

We store detached rollout states and recompute one transition at a time for eq. (121). Since $D_x\epsilon_\psi = \nabla_x^2 E_\psi$, a vector–Jacobian product through the reference mean requires a Hessian–vector product through the energy, evaluated by differentiating its input gradient. The energy weights remain frozen, but the input-gradient computation must retain its differentiation graph during this local re-computation. No dense Hessian or full-rollout graph is materialised. The predicted clean state, reward-gradient direction, and auxiliary geometry supplied to the controller are deliberately detached; the recursion differentiates the transition with these observations held fixed, as in the implemented computation graph.

Setting	Final configuration
Trajectory / reverse steps	56 knots / 10 DDIM steps
Stochasticity / prior conditioning	$\eta = 1$ / $\omega_{\text{cfg}} = 2$
Controller	Width 344, 4 heads, 2 axial blocks
Feed-forward / pair-bias hidden width	$4\times$ controller width / 32
Optimiser	Adam, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$
Updates / batch / microbatch	20,000 / 256 joint queries / 128
Learning rate	400-step warm-up to 10^{-4}, cosine decay to 10^{-5}
EMA / gradient-norm clipping	0.99 / 0.5
Reward coefficient / active matching steps	$\beta = 40$ / all 9 stochastic transitions
Robust calibration	First 500 updates; 95th percentile; floor 10^{-3}
Primary / supplementary training seed	0 / 1; final-update EMA weights

Table 12: **KUKA controller-training configuration.** Values are taken from the final saved training configuration; the priors remain frozen.

D.5.3 Collision reward, controller, and training

Differentiable collision reward. We divide each trajectory segment into four subsegments and compute arm-link positions at the sampled points. We approximate each link with four spheres to measure distances between arms and to obstacles and the table. Self-collision distances use a sampled-axis convex-hull approximation with a 2 mm offset. Intended contact between the fixed mounting link and the table is excluded.

For clearances d, safety margin m, and activation width $\delta = 0.01\,\text{m}$, we define

$$p(d; m) = \left[\frac{m + \delta - d}{\delta}\right]_+^2, \qquad P(d; m) = \operatorname{mean} p(d; m) + \max p(d; m), \tag{123}$$

where $[x]_+ = \max(x, 0)$. The mean penalises violations across the sampled points, while the maximum emphasises the worst violation for each arm or arm pair. For an assembled trajectory $Y = \varphi(X) = q$, the reward combines inter-arm, obstacle, table, and self-collision penalties:

$$r_\kappa(Y) = -\left[\frac{1}{\binom{N}{2}} \sum_{i<j} P(d^{ij}_{\text{arm}}; m) + \frac{1}{N} \sum_i \left(2P(d^i_{\text{obs}}; 0) + \tfrac{1}{2} P(d^i_{\text{table}}; 0) + 2P(d^i_{\text{self}}; 0)\right)\right].$$

The clearances are evaluated on Y in scene κ; these arguments are suppressed on the right-hand side.

All distances are in metres. Penalties activate within 1 cm of contact; the additional inter-arm margin m is zero in these experiments. Start and goal configurations are fixed exactly. We use no endpoint, path-length, or curvature penalty. The reward and control cost do not guarantee collision-free, smooth, or dynamically feasible motion.

Shared control. The controller follows the same implementation detailed in Appendix D.4.5. It first attends along each arm's trajectory, then applies two axial blocks with temporal and cross-arm attention, four heads, and hidden width 344. All arm-wise parameters are shared; no arm-identity embedding is used. Inputs include the noisy state, detached clean prediction, normalised reward gradient, endpoint and base conditions, and local kinematic features. During cross-arm attention, the controller uses the relative positions of each pair of arms, their clearance, goals, base positions, and motion to adjust how much attention they give each other. Consequently, the same controller can process an additional arm without changing parameter dimensions.

The controller is parameterised using the Tweedie reward gradient, CMDS+TRG; see eq. (112), and uses the pseudo-Huber loss in Appendix D.4.7. Detailed parameters are provided in table 12.

D.5.4 Baselines and PBDM refinement

Independent and joint priors. Single-arm KUKA samples each arm independently using the same frozen prior as CMDS; Dual KUKA uses the released joint prior. All methods use 56 way-

points, 10 DDIM steps, classifier-free weight $\omega_{\text{cfg}} = 2$, and fixed endpoints. Single-arm sampling uses $\eta = 1$; Dual KUKA retains its native $\eta = 0$ sampler. We also evaluated single-arm sampling with $\eta = 0$, but it did not improve performance. Each of the K candidates contains a path for every arm. Except for PBDM, methods return the first candidate that passes the common collision and endpoint checks.

Direct collision guidance. Guided single-arm and joint priors use the same collision reward as CMDS, with gradient scale 0.6 on the first nine DDIM transitions. The reward gradient is evaluated at the predicted clean path, without backpropagating through the prior. The guidance strength was selected from $\{0.1, 0.3, 0.6, 1.0, 1.7\}$ by validation success across $K = 1, 2, 3, 4$, then fixed for all tests.

PBDM selection and refinement. PBDM uses the released joint prior and planning procedure (Luo et al., 2024). The release defaults to 20 initial candidates with refinement disabled; our main PBDM results enable its optional refinement with the stated candidate budget K equal to 1 or 4.

1. Generate K candidates and return the first accepted by PBDM's own checker.
2. If none passes, try the first $\min(K, 6)$ candidates in order, allowing up to $R = 3$ repair attempts each. Repair partially re-noises a path, applies three DDIM steps, and uses the released segment acceptance and splicing rules. Return the first accepted repaired path, or report failure.
3. Evaluate the returned path with our common strict checker. Success requires passing both checks; strict rejection does not trigger another candidate or repair.

Dual KUKA and PBDM can therefore return different paths from the same candidates. Increasing K need not improve PBDM's strict success, because its native selection can return a different path. We record native acceptance separately. The budget K excludes repair attempts, but planning time includes all repair and checking. These results evaluate the released procedure under our protocol, rather than reproduce the original paper's reported scores.

Single-arm priors with adapted PBDM. This baseline uses independent single-arm priors with $\eta = 0$, 56 waypoints, and ten DDIM steps. During repair, the released partial denoiser runs separately for each arm, using local obstacle coordinates and fixed endpoints. Joint segments are accepted using PBDM's rules, with three repair attempts and three denoising steps per attempt. The adaptation supports three to five arms. It adds no learned control or collision-gradient guidance.

D.5.5 Strict evaluation, timing, and uncertainty

Strict success. An independent PyBullet mesh checker tests inter-arm, obstacle, table, and non-adjacent self collisions, excluding intended mounting contact. Each trajectory segment has at least eight subdivisions, with joint increments no larger than 0.04 radians. Success requires positive clearance, finite joint angles, joint limits satisfied within 10^{-6} radians, and endpoint errors no larger than 10^{-5} radians.

The additional inter-arm margin is zero; the reward's 1 cm activation width is not an evaluation threshold. Denser offline checks assess sensitivity to interpolation without changing the reported decisions or timings. These sampled checks do not guarantee collision freedom between every checked point.

Planning time. Each task generates its K candidates as one batch on a GH200 GPU using warm CUDA graphs checked against the eager implementation. Timing includes scene setup, generation, transfer, checking, and any repair through the final decision. It excludes model and asset loading, graph preparation, diagnostics, logging, and clean-up.

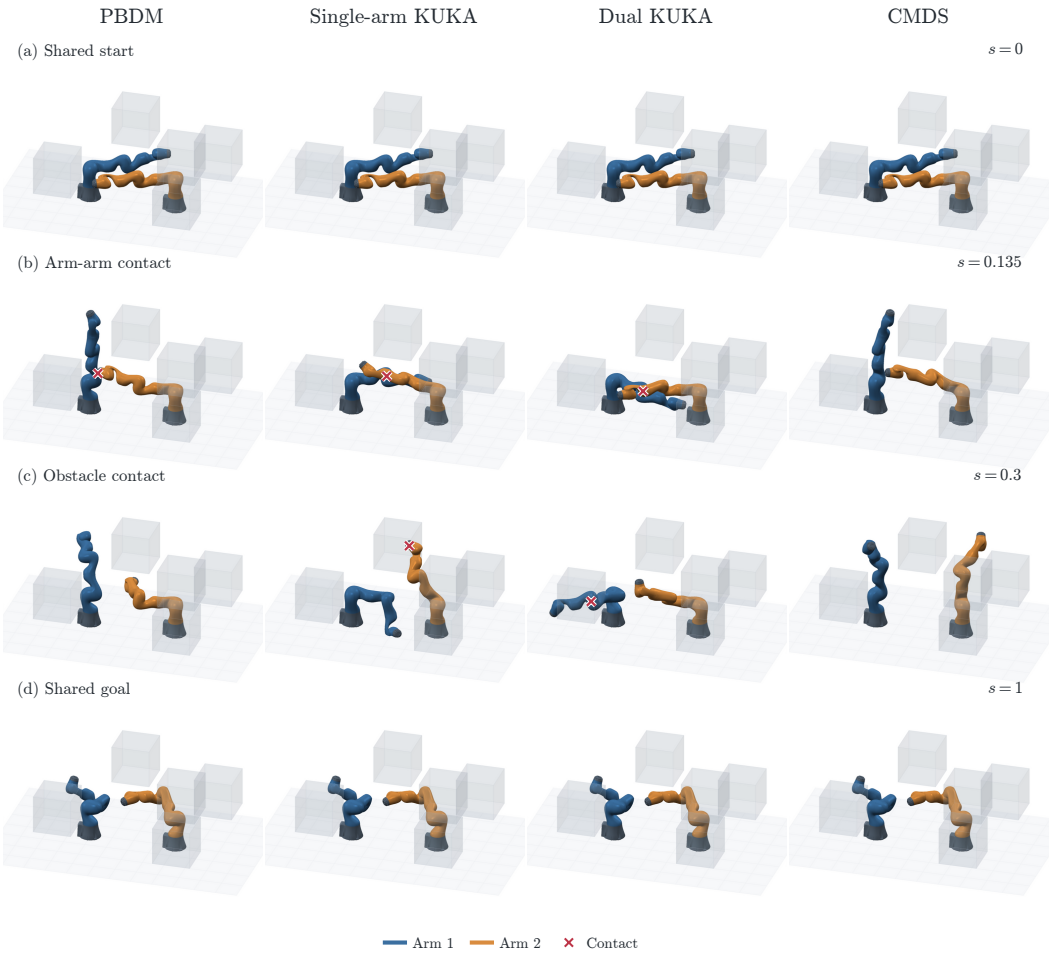


Figure 20: **Synchronised KUKA trajectories.**

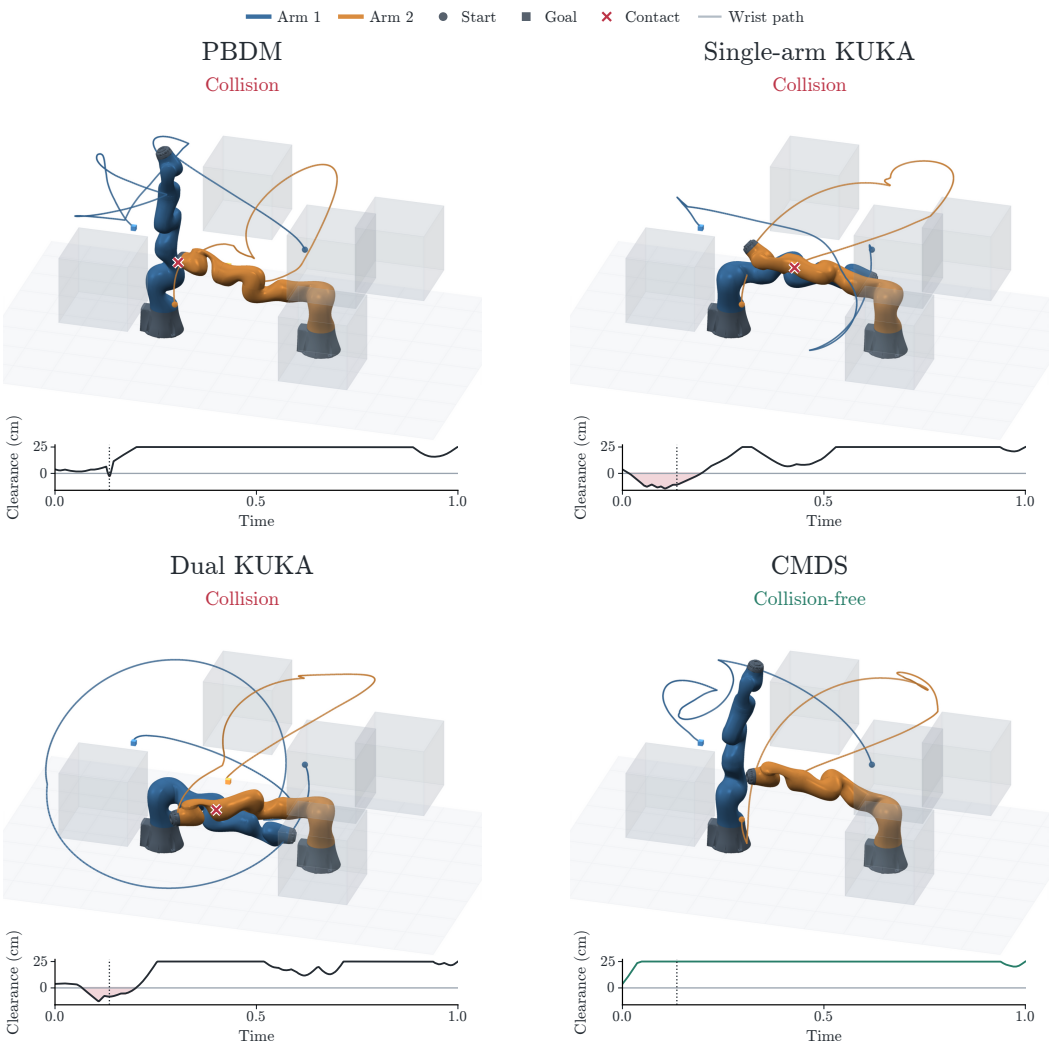


Figure 21: **KUKA poses and inter-arm clearance.**

D.6 GOAL-DIRECTED MULTI-PERSON MOTION COORDINATION

Key Takeaways: Frozen single-person priors can be coordinated into multi-person motion. A single shared control coordinates four independently text-conditioned MDM processes to reach prescribed destinations, avoid inter-person collisions, and maintain kinematic validity, without paired interaction data. **Amortisation makes coordination efficient.** CMDS achieves 90.0% joint success compared with 75.0% for PCD++, while requiring $0.54\,\mathrm{s}$ per assembly versus $145.34\,\mathrm{s}$ for PCD++.

We coordinate four text-conditioned human motions using a shared control and a frozen single-person prior. People cross a common region towards opposite destinations, requiring collision avoidance and consistent articulated motion.

D.6.1 TASK, REPRESENTATION, AND FROZEN MDM PRIOR

Each person $i \in \{1, \dots, N\}$, with $N = 4$, receives a caption c_i, a planar start $s_i \in \mathbb{R}^2$, an initial yaw ψ_i, and a destination $g_i = -s_i$. Starts are arranged on a randomly rotated ring with nominal radius $2.147\,\mathrm{m}$, radial jitter $\pm 0.15\,\mathrm{m}$, and angular jitter $\pm 10^\circ$. Yaw points towards the destination with an additional $\pm 10^\circ$ jitter. Scene conditions remain fixed during generation, and waiting and lateral detours are allowed. A motion contains $F = 120$ frames at $20\,\mathrm{Hz}$, represented by 263 normalised HumanML features per frame.

Notation and assembly. We retain the discrete state and indexing convention of Algorithm 1: the joint component state is $X_\ell = (X_\ell^1, \dots, X_\ell^N) \in \mathbb{R}^{N \times F \times 263}$, the reverse-time grid is $\zeta_\ell = K-1-\ell$, and the generated motion is X_K, with $K = 50$. Thus X_0 denotes Gaussian noise, rather than a clean motion, and ℓ is distinct from the physical frame index $f \in \{0, \dots, F-1\}$. We use τ for possibly fractional frame indices on the densified collision-checking grid. Each reverse transition updates the entire motion clip; the terminal reward is evaluated after denoising and can inspect all physical frames. The task condition is $\kappa = \{(c_i, s_i, g_i, \psi_i)\}_{i=1}^N$, together with the active-person and frame masks. A deterministic geometric decoder $D_i(\cdot; s_i, \psi_i)$ denormalises person i's features, reconstructs 22 joints, and applies the scene transform, enforcing $p_i(0) = s_i$ by construction. For a fixed task, the assembly retains both the native features and the decoded world geometry:

$$\varphi(X) := \big(X, J_\kappa(X)\big), \qquad J_\kappa(X) := \big(D_i(X^i; s_i, \psi_i)\big)_{i=1}^N. \tag{124}$$

We suppress the fixed scene dependence of φ in the notation. Retaining native features is necessary for the rotation, displacement, and contact-consistency costs, which cannot be evaluated from world-space joint positions alone. State variables, noise, and optimisation are restricted to active people and frames; padded coordinates remain zero throughout. For motion-feature tensors, $\|\cdot\|_2$ denotes the Euclidean norm after vectorisation, without averaging over people, frames, or features.

Frozen denoising and reverse transitions. We use the released HumanML MDM checkpoint trained for 50 diffusion steps, together with its frozen CLIP ViT-B/32 text encoder. Each person's frozen denoiser is conditioned on its own caption, not on the other people or the scene's starts and goals, with classifier-free guidance scale 2.5. This is a separately trained native 50-step model, not a respaced $1{,}000$-step checkpoint. Sampling retains its cosine schedule, fixed posterior variance, and unclipped clean predictions. Write $\alpha_t = 1 - \beta_t$, $\bar\alpha_t = \prod_{j=0}^t \alpha_j$, and $\bar\alpha_{-1} = 1$ for the saved native schedule. For a candidate joint state $Z = (Z^1, \dots, Z^N)$, define

$$\begin{aligned} \mathcal{D}_\ell^i(Z^i) &:= \frac{Z^i - \sqrt{1-\bar\alpha_{\zeta_\ell}}\,\epsilon_{\mathrm{b}}^i(Z^i, \zeta_\ell)}{\sqrt{\bar\alpha_{\zeta_\ell}}}, && 0 \le \ell < K, \\ \mathcal{D}_\ell(Z) &:= \big(\mathcal{D}_\ell^1(Z^1), \dots, \mathcal{D}_\ell^N(Z^N)\big), && \mathcal{D}_K(Z) := Z. \end{aligned} \tag{125}$$

Thus $\hat X_{1|\ell} = \mathcal{D}_\ell(X_\ell)$ is the clean-motion estimate, and $\hat Y_{1|\ell} = \varphi(\hat X_{1|\ell})$ is its assembly. MDM predicts the clean state directly; the expression above is its equivalent epsilon parameterisation, with classifier-free guidance included in ϵ_{b}^i. The denoising map $\mathcal{D}_\ell$ is distinct from the geometric decoder D_i. Although Algorithm 1 uses DDIM, the reverse maps here are MDM's native DDPM transitions:

$$F_\ell^{\mathrm{b}}(X_\ell, \Xi_\ell) := \mu_{\mathrm{b},\ell}(X_\ell) + \varsigma_\ell \Xi_\ell, \qquad \begin{cases} \varsigma_\ell > 0, & 0 \le \ell < K-1, \\ \varsigma_{K-1} = 0. \end{cases} \tag{126}$$

Here $\mu_{\mathrm{b},\ell}$ is the frozen classifier-free-guided DDPM posterior mean, acting component-wise. With $\epsilon_{\mathrm{b}}(Z,t) = (\epsilon_{\mathrm{b}}^i(Z^i,t))_{i=1}^N$, the native mean and effective variance are

$$\begin{aligned} \mu_{\mathrm{b},\ell}(Z) &= \frac{1}{\sqrt{\alpha_{\zeta_\ell}}}\left[Z - \frac{\beta_{\zeta_\ell}}{\sqrt{1-\bar{\alpha}_{\zeta_\ell}}}\epsilon_{\mathrm{b}}(Z,\zeta_\ell)\right], \\ \varsigma_\ell^2 &= \frac{\beta_{\zeta_\ell}(1-\bar{\alpha}_{\zeta_\ell-1})}{1-\bar{\alpha}_{\zeta_\ell}}. \end{aligned} \tag{127}$$

The final transition has $\zeta_{K-1} = 0$ and is deterministic. No denoiser evaluation or additional grid point ζ_K is needed for $\mathcal{D}_K = \mathrm{Id}$.

Controlled transition. We use $w_{\theta,\ell}$ for the normalised discrete control, consistently with Algorithm 1. For the stochastic transitions, CMDS uses

$$\begin{aligned} X_{\ell+1} &= F_\ell^\theta(X_\ell, \Xi_\ell) \\ &:= \mu_{\mathrm{b},\ell}(X_\ell) + \varsigma_\ell w_{\theta,\ell}(X_\ell) + \varsigma_\ell \Xi_\ell, \qquad \Xi_\ell \sim \mathcal{N}(0,I), \quad 0 \le \ell < K-1. \end{aligned} \tag{128}$$

Unlike the epsilon-residual parameterisation in Algorithm 1, the motion controller outputs $w_{\theta,\ell}$ directly, supplying the mean correction $\varsigma_\ell w_{\theta,\ell}$. For $\varsigma_\ell > 0$, this is the same normalisation $w_{\theta,\ell} = (F_\ell^\theta - F_\ell^{\mathrm{b}})/\varsigma_\ell$, with both maps evaluated at the same state and noise. The final update is $X_K = \mu_{\mathrm{b},K-1}(X_{K-1}) = \mathcal{D}_{K-1}(X_{K-1})$, with no learned correction, control-energy term, or matching loss at that transition. In particular, the positive-noise terminal modification used for DDIM in Algorithm 1 is not used here.

DDPM–DDIM control correspondence. Both samplers use the same control normalisation, but different network outputs. In Algorithm 1, an epsilon residual $\delta\epsilon_{\theta,\ell}$ induces the DDIM mean shift $\gamma_\ell \delta\epsilon_{\theta,\ell}$, giving $w_{\theta,\ell} = (\gamma_\ell/\varsigma_\ell)\delta\epsilon_{\theta,\ell}$ for that sampler. For the native DDPM kernel, the corresponding signed epsilon coefficient and equivalent residual are

$$\begin{aligned} \gamma_\ell^{\mathrm{DDPM}} &:= -\frac{\beta_{\zeta_\ell}}{\sqrt{\alpha_{\zeta_\ell}}\sqrt{1-\bar{\alpha}_{\zeta_\ell}}}, \\ \delta\epsilon_{\theta,\ell}^{\mathrm{equiv}} &:= \frac{\varsigma_\ell}{\gamma_\ell^{\mathrm{DDPM}}} w_{\theta,\ell}, \qquad 0 \le \ell < K-1. \end{aligned} \tag{129}$$

The sign is retained: $\gamma_\ell^{\mathrm{DDPM}}\delta\epsilon_{\theta,\ell}^{\mathrm{equiv}} = \varsigma_\ell w_{\theta,\ell}$. The motion implementation requires no such conversion because the controller outputs $w_{\theta,\ell}$ directly. For adjacent native timesteps, the same denoiser and schedule, and no clipping, stochastic DDIM with $\eta = 1$ reproduces the DDPM mean and posterior variance on stochastic transitions. This does not identify the complete rollouts: Algorithm 1 retains noise at its modified DDIM endpoint, whereas MDM uses its original control-free deterministic endpoint. Thus the adaptation changes the sampling kernel, output parameterisation, and endpoint treatment, rather than the underlying stochastic-control formulation.

D.6.2 Cooperative control architecture

The controller forms one token per person and physical frame. Its motion input concatenates the current noisy features, the prior's clean-motion estimate, and the decoded world-space joints:

$$\begin{aligned} \hat{J}_{\ell,i,f} &:= D_i(\hat{X}^i_{1|\ell}; s_i, \psi_i)_f, \\ z_{\ell,i,f} &:= \left[(X^i_\ell)_f,\ (\hat{X}^i_{1|\ell})_f,\ \mathrm{vec}(\hat{J}_{\ell,i,f})\right] \in \mathbb{R}^{263+263+66}. \end{aligned} \tag{130}$$

The 66 skeleton coordinates represent the 22 joints in metres.

We project $z_{\ell,i,f}$ to width 344 and add separate projections of the 512-dimensional CLIP embedding, the static condition $[s_i, g_i, \sin\psi_i, \cos\psi_i, F/196] \in \mathbb{R}^7$, and 32-dimensional sinusoidal embeddings of the physical frame f and native diffusion index ζ_ℓ, then apply SiLU. A local temporal encoder attends to each person's entire clip. Two subsequent blocks alternate temporal attention within a person, attention across people at the same frame, and a feed-forward network. Communication is aligned in physical time, with whole-motion context supplied by the temporal passes. Weights are shared across people, making the controller equivariant to a consistent permutation of people and their conditions.

Geometric communication. Person attention receives an additive bias computed from predicted planar root geometry. At a fixed reverse transition and physical frame, suppress these indices and write $\Delta p_{ij} = \hat{p}_j - \hat{p}_i$, $d_{ij}^{\mathrm{root}} = \|\Delta p_{ij}\|_2$, and $\Delta v_{ij} = \hat{v}_j - \hat{v}_i$, where velocities use 20 Hz backward differences. The bias is

$$b_{ij} = \mathrm{MLP}_{8\to 32\to 4}\left(\left[\Delta p_{ij},\, d_{ij}^{\mathrm{root}},\, \Delta v_{ij},\, \frac{-\Delta p_{ij}^{\top}\Delta v_{ij}}{\max(d_{ij}^{\mathrm{root}}, 10^{-4})},\, g_j - g_i\right]\right). \tag{131}$$

Its four outputs are added to the corresponding attention-head logits $q_i^{\top} k_j/\sqrt{86}$. The same bias is used in both interaction blocks. The root distance d_{ij}^{root} is distinct from the signed body-capsule clearance defined below.

Control output. Separate linear heads read the local encoder output H_{local} and the second interaction block output H_{joint}:

$$\begin{aligned} w_{\theta,\ell} = {} & W_{\mathrm{local}} H_{\mathrm{local}} + b_{\mathrm{local}} \\ & + W_{\mathrm{joint}} H_{\mathrm{joint}} + b_{\mathrm{joint}} \in \mathbb{R}^{N\times F\times 263}. \end{aligned} \tag{132}$$

This $4{,}863{,}042$-parameter control supplies joint awareness to the separate denoising processes. Both output heads and the final layer of the geometric-bias MLP are zero-initialised, so the initial controller leaves the frozen sampling process unchanged.

D.6.3 Assembly reward

For a completed joint motion X, write $Y = \varphi(X)$. The total cost combines collision avoidance, goal attainment, directional progress, and physical consistency:

$$\begin{aligned} C_\kappa(X) = {} & \lambda_{\mathrm{col}} E_{\mathrm{col}} + \lambda_{\mathrm{goal}} E_{\mathrm{goal}} + \lambda_{\mathrm{prog}} E_{\mathrm{prog}} \\ & + \lambda_{\mathrm{bone}} E_{\mathrm{bone}} + \lambda_{\mathrm{FK}} E_{\mathrm{FK}} + \lambda_{\mathrm{disp}} E_{\mathrm{disp}} \\ & + \lambda_{\mathrm{contact}} E_{\mathrm{contact}} + \lambda_{\mathrm{floor}} E_{\mathrm{floor}} + \lambda_{\mathrm{speed}} E_{\mathrm{speed}}. \end{aligned} \tag{133}$$

All terms are evaluated using X and $\varphi(X)$, with dependence on the fixed task suppressed on the right-hand side. We distinguish $C_\kappa^{\mathrm{train}}$, used to train CMDS, from C_κ^{base}, used for baseline selection and guidance. They use identical terms and normalisations, but different goal weights: $\lambda_{\mathrm{goal}} = 50$ for control training and $\lambda_{\mathrm{goal}} = 1$ for the baselines. The corresponding assembly-level rewards are

$$r_\kappa^{\mathrm{train}}(\varphi(X)) := -C_\kappa^{\mathrm{train}}(X), \qquad r_\kappa^{\mathrm{base}}(\varphi(X)) := -C_\kappa^{\mathrm{base}}(X). \tag{134}$$

The subscripted cost weights are distinct from the outer SOC reward multiplier λ, specified in Appendix D.6.4. Their recorded values are reported in table 14. All geometric distances below are in metres.

Body collision. Each person is represented by 15 fixed-radius capsule primitives, including a spherical head primitive. For capsule centreline segment $S_{ia}(\tau; Y)$ and radius ρ_a, the signed surface clearance is

$$d_{ij}^{ab}(\tau; Y) := \mathrm{dist}\big(S_{ia}(\tau; Y), S_{jb}(\tau; Y)\big) - \rho_a - \rho_b. \tag{135}$$

We divide each native frame interval into five linear subintervals and retain the final endpoint, giving the synchronised grid $\mathcal{T} = \{n/5 : n = 0, \ldots, 5(F-1)\}$, with $|\mathcal{T}| = 5(F-1)+1 = 596$. The collision penalty is

$$E_{\mathrm{col}} = \frac{1}{\binom{N}{2}|\mathcal{T}|} \sum_{i<j} \sum_{\tau\in\mathcal{T}} \sum_{a,b=1}^{15} \left(\frac{[0.03 - d_{ij}^{ab}(\tau; Y)]_+}{0.05}\right)^2, \qquad [z]_+ := \max(z, 0). \tag{136}$$

For $N = 4$, the denominator is $6|\mathcal{T}|$, as in the implemented cost. The penalty averages over person pairs and dense times, sums all 225 capsule pairs, and becomes zero at 3 cm clearance. This soft-cost margin is distinct from both PCD++'s nonnegative-clearance constraint and evaluation's 5 mm penetration tolerance. Table 13 specifies the primitive endpoints and radii.

Table 13: Body-capsule geometry for collision costs and checks. Endpoints are zero-based HumanML joint indices; the repeated head endpoint defines a sphere.

Primitive	Joint endpoints	Radius (m)
Pelvis	$(1, 2)$	0.120
Torso	$(0, 9)$	0.150
Shoulder span	$(16, 17)$	0.100
Upper torso–head	$(9, 15)$	0.100
Head	$(15, 15)$	0.110
Upper arms	$(16, 18),\ (17, 19)$	0.055
Forearms	$(18, 20),\ (19, 21)$	0.045
Thighs	$(1, 4),\ (2, 5)$	0.085
Lower legs	$(4, 7),\ (5, 8)$	0.065
Feet	$(7, 10),\ (8, 11)$	0.055

Destination and progress. Let $p_i(f; Y)$ be person i's planar root position, and set the goal tolerance to $\varepsilon_{\mathrm{g}} = 0.5\,\mathrm{m}$. On the final five frames, $\mathcal{W}_i = \mathcal{W} = \{F-5, \dots, F-1\} = \{115, \dots, 119\}$, we use

$$E_{\mathrm{goal}} = \frac{1}{N}\sum_{i=1}^{N} \frac{\min_{f\in\mathcal{W}_i} \|p_i(f;Y) - g_i\|_2^2}{\varepsilon_{\mathrm{g}}^2}, \tag{137}$$

$$\bar{q}_i = \frac{1}{|\mathcal{W}_i|}\sum_{f\in\mathcal{W}_i} \frac{(p_i(f;Y) - s_i)^\top (g_i - s_i)}{\|g_i - s_i\|_2^2}, \qquad E_{\mathrm{prog}} = \frac{1}{N}\sum_{i=1}^{N} [0.9 - \bar{q}_i]_+^2. \tag{138}$$

People may arrive at different frames within the window. Progress is the window-mean displacement along the start-to-goal direction; its threshold shapes training but is not an extra success condition.

Physical consistency. Let $Q_{ifj} \in \mathbb{R}^3$ be canonical decoded joints and J_{ifj} their world-space counterparts. Write L_{ifb} for the 21 bone lengths, $L_b^\star$ for fixed audit references, and Q^{FK} for joints reconstructed from the native 6D rotations using those reference lengths. For denormalised displacement channels V_{ifj}, root-local rotation A_{if}, and $\Delta Q_{ifj} = Q_{i,f+1,j} - Q_{ifj}$, the costs are

$$E_{\mathrm{bone}} = \left\langle (L/L^\star - 1)^2 \right\rangle, \qquad E_{\mathrm{FK}} = \frac{\langle \|Q - Q^{\mathrm{FK}}\|_2^2 \rangle}{3(0.05)^2}, \tag{139}$$

$$E_{\mathrm{disp}} = \frac{\langle \|V - A\Delta Q\|_2^2 \rangle}{3(0.05)^2}, \qquad E_{\mathrm{contact}} = \langle (H - H^\star)^2 \rangle, \tag{140}$$

$$E_{\mathrm{floor}} = \frac{\langle [-J_y]_+^2 \rangle}{(0.05)^2}, \qquad E_{\mathrm{speed}} = \left\langle [\|20\Delta Q\|_2 / v_{\max} - 1]_+^2 \right\rangle. \tag{141}$$

Angle brackets average over people, valid native frames or intervals, and the relevant bones or joints. The factors of three make vector errors coordinate-wise means. Encoded displacement is compared per frame, without multiplying it by the frame rate. Here $v_{\max}$ is the speed threshold in the soft cost. For denormalised contact channels H, corresponding to joints $7, 10, 8, 11$, the soft reference is $H^\star = \sigma((0.08 - h)/0.02)\,\sigma((0.2 - v)/0.05)$, using foot height h, speed v, and logistic sigmoid σ. Root speed and jerk are additional hard evaluation checks.

D.6.4 Training by adjoint matching

Only controller parameters are updated. Training scenes are generated online, with captions drawn from five families: walking, jogging, high-knee marching, forward jumping, and crouched walking. Each family has three phrasings. A quarter of scenes use one shared action family; the remainder sample family assignments conditioned on having at least two distinct families. Phrasings are sampled per person. Training uses generated trajectories, without paired interaction data.

Control objective and replay. The motion experiment uses reward strength $\lambda = 100$ and includes all stochastic transitions in the matched set $\mathcal{K} = \{0, \dots, K-2\}$. Its discrete stochastic-control

objective is

$$\mathcal{J}(\theta) = \mathbb{E}\left[\lambda C_\kappa^{\text{train}}(X_K) + \frac{1}{2}\sum_{\ell=0}^{K-2} \|w_{\theta,\ell}(X_\ell)\|_2^2\right]. \tag{142}$$

The expectation covers task conditions, initial noise, and rollout noise. The norm sums over people, physical frames, and all 263 motion features, without averaging; λ scales only the terminal cost. For a fixed task and state, the controlled and reference stochastic transitions have common covariance $\varsigma_\ell^2 I$ and mean difference $\varsigma_\ell w_{\theta,\ell}$, giving conditional KL divergence $\frac{1}{2}\|w_{\theta,\ell}\|_2^2$. With the same initial Gaussian distribution, the expected sum of these energies equals the discrete path-space KL relative to the frozen sampler at classifier-free guidance scale 2.5; the shared deterministic final map adds no conditional KL. This Gaussian mean-shift interpretation does not cover arbitrary nonzero corrections at zero-noise transitions, including deterministic DDIM with $\eta = 0$.

As in Algorithm 1, set $\bar{\theta} = \text{stopgrad}(\theta)$, generate a controlled rollout without gradient tracking, and replay individual transitions backwards. Suppressing the rollout superscript, the state adjoints satisfy

$$\begin{aligned} a_K &= \lambda \nabla_{X_K} C_\kappa^{\text{train}}(X_K), \\ a_{K-1} &= \left[D_X \mu_{\text{b},K-1}(X_{K-1})\right]^\top a_K, \\ a_\ell &= \left[D_X \mu_{\text{b},\ell}(X_\ell) + \varsigma_\ell D_X w_{\bar{\theta},\ell}(X_\ell)\right]^\top a_{\ell+1} \\ &\quad + \left[D_X w_{\bar{\theta},\ell}(X_\ell)\right]^\top w_{\bar{\theta},\ell}(X_\ell), \qquad \ell = K-2, \ldots, 0. \end{aligned} \tag{143}$$

The deterministic final transition remains in adjoint propagation, but contributes neither control energy nor a matching term. Each transition is differentiated using a fresh state variable at the stored value X_ℓ, with $\bar{\theta}$ fixed and the next-state adjoint detached. The controller's clean-motion, world-joint, text, and geometry-derived observations are detached and held fixed during this state differentiation, while its explicit noisy-state input and the frozen-prior mean remain differentiable. The recursion follows these observation-detach conventions, rather than differentiating through the recomputation of every controller observation; graphs are released between transitions.

Calibrated robust matching. Training regresses against detached next-state adjoints using

$$e_{\theta,\ell} := w_{\theta,\ell}\big(\text{stopgrad}(X_\ell)\big) + \varsigma_\ell \,\text{stopgrad}(a_{\ell+1}), \qquad \ell \in \mathcal{K}. \tag{144}$$

The control is evaluated with the same task and detached observations as in replay. With $D = NF \cdot 263 = 4 \cdot 120 \cdot 263$ motion-feature coordinates, the robust matching loss is

$$\mathcal{L}_{\text{PH}}(\theta) = \mathbb{E}\left[\sum_{\ell=0}^{K-2} D\delta_\ell^2 \left(\sqrt{1 + \frac{\|e_{\theta,\ell}\|_2^2}{D\delta_\ell^2}} - 1\right)\right]. \tag{145}$$

The first 500 successful updates use squared matching and collect per-instance RMS residuals $\|e_{\theta,\ell}\|_2/\sqrt{D}$. Each δ_ℓ is then fixed to their 95th percentile, floored at 10^{-3}. With squared matching, current-policy rollouts, and all stochastic transitions included, the matching gradient agrees with direct differentiation of eq. (142) under the same observation-detach conventions. Pseudo-Huber matching downweights large residuals and generally changes that gradient; it leaves the terminal reward, adjoint recursion, and quadratic control energy unchanged.

Raw and EMA checkpoints were validated during training. Selection ranked joint success, then terminal cost plus control energy, preferring earlier steps on ties. Evaluation uses raw step-15,000 weights from one seed-0 run. Inference requires no reward-gradient or projection iterations.

D.6.5 Baselines

Baselines share the frozen prior, native DDPM sampler, classifier-free guidance scale, scenes, decoder, and audit. Selection and guidance use C_κ^{base}, with weights $(100, 1, 1, 500, 5, 5, 1, 200, 10)$ in the order of eq. (133). These retain exactly the cost normalisations above; in particular, the baseline goal weight is 1, rather than the training value 50.

Best-of-256. Best-of-256 generates 256 independent complete four-person assemblies and returns the one with minimum baseline cost. Uncontrolled MDM samples the same frozen component priors independently and is used as a qualitative reference.

Table 14: Reward weights and controller training settings for the reported $N = 4$ experiment. Baselines share the cost normalisations but use a smaller goal weight.

Setting	Recorded value
$(\lambda_{\mathrm{col}}, \lambda_{\mathrm{goal}}, \lambda_{\mathrm{prog}})$, training	$(100, 50, 1)$
$(\lambda_{\mathrm{col}}, \lambda_{\mathrm{goal}}, \lambda_{\mathrm{prog}})$, baselines	$(100, 1, 1)$
$(\lambda_{\mathrm{bone}}, \lambda_{\mathrm{FK}}, \lambda_{\mathrm{disp}})$	$(500, 5, 5)$
$(\lambda_{\mathrm{contact}}, \lambda_{\mathrm{floor}}, \lambda_{\mathrm{speed}})$	$(1, 200, 10)$
Outer SOC reward multiplier λ	100, control training only
Optimiser / updates / learning rate	Adam / 15,000 / constant 10^{-4}
Effective batch size	8 scenes: microbatch 2, accumulation 4
Gradient-norm clipping / EMA decay	0.5 / 0.9995

DPS guidance. DPS differentiates the joint baseline reward of the clean-motion estimate through the frozen denoiser to the current noisy state:

$$\begin{aligned} v_\ell &:= \nabla_{X_\ell} r_\kappa^{\mathrm{base}}(\varphi(\mathcal{D}_\ell(X_\ell)))\,, \\ \widetilde{X}_{\ell+1} &:= \mathrm{stopgrad}\big(F_\ell^{\mathrm{b}}(X_\ell, \Xi_\ell) + \eta_{\mathrm{g}} v_\ell\big)\,. \end{aligned} \tag{146}$$

DPS sets $X_{\ell+1} = \widetilde{X}_{\ell+1}$. DPS, PCD, and PCD++ all use $\eta_{\mathrm{g}} = 0.1$ at every transition, including the deterministic final one. This coefficient directly multiplies the reward gradient: there is no ς_ℓ^2 factor and no separate application of the control-training multiplier $\lambda = 100$. All guidance gradients are with respect to state variables, with denoiser parameters frozen and no gradient propagation across outer diffusion steps. Unlike the learned control in eq. (128), this constant guidance also changes the deterministic endpoint; the subsequent PCD projections are not Gaussian mean shifts either. These baselines therefore do not inherit the learned controller's quadratic path-KL interpretation.

PCD and PCD++. The PCD baseline combines eq. (146) with a root-goal projector enforcing the final-window destination condition. The projector changes planar root increments and the corresponding stored joint-displacement channels while keeping headings, heights, and poses fixed. It minimises normalised-feature change within this restricted parameterisation, but does not enforce collision clearance or the kinematic-validity audit. PCD++ uses the same guidance update and replaces that projector with a joint optimisation of the proposed state. All 263 normalised feature channels of each active person and frame are free variables in this solve. Feasibility is checked on a freshly recomputed clean estimate at the candidate's next noise level, rather than on the clean estimate used for the preceding guidance update. At the final transition, the candidate is already a clean motion and $\mathcal{D}_K = \mathrm{Id}$.

Joint goal and collision constraints. Using the planar roots and signed capsule clearances defined above, set

$$\begin{aligned} c_{\mathrm{g},i}(Y) &:= \min_{f \in \mathcal{W}_i} \|p_i(f; Y) - g_i\|_2 - \varepsilon_{\mathrm{g}}, \\ c_{\mathrm{c},ijab\tau}(Y) &:= \delta_{\mathrm{c}} - d_{ij}^{ab}(\tau; Y), \\ V_\kappa(Y) &:= \max\left\{0,\ \max_i c_{\mathrm{g},i}(Y),\ \max_{\substack{i<j,\,a,b \\ \tau \in \mathcal{T}_{ij}}} c_{\mathrm{c},ijab\tau}(Y)\right\}. \end{aligned} \tag{147}$$

Here $\mathcal{T}_{ij}$ contains the checked times when both people are active; in the reported fixed-length experiment, $\mathcal{T}_{ij} = \mathcal{T}$ for every pair. We use $\varepsilon_{\mathrm{g}} = 0.5\,\mathrm{m}$ and $\delta_{\mathrm{c}} = 0$, so PCD++ requires nonnegative capsule clearance. The feasible set is

$$\mathcal{F}_\kappa := \big\{Y : V_\kappa(Y) = 0,\ p_i(0; Y) = s_i \text{ for all } i\big\}. \tag{148}$$

Starts are already enforced by the decoder. For a guided proposal $\widetilde{X}_{\ell+1}$, the projection targets

$$\underset{Z}{\text{minimise}} \quad \tfrac{1}{2}\|Z - \widetilde{X}_{\ell+1}\|_2^2 \qquad \text{subject to} \qquad \varphi(\mathcal{D}_{\ell+1}(Z)) \in \mathcal{F}_\kappa. \tag{149}$$

The distance is Euclidean in the active normalised motion features, not in decoded joint coordinates. The implemented solver approximates this constrained problem; it does not certify a nearest feasible point.

Penalty and numerical guards. The inner objective uses the squared-hinge penalty

$$\begin{aligned}\mathcal{P}_\kappa(Y;\omega_{\mathrm{g}},\omega_{\mathrm{c}}) := &\sum_i [c_{\mathrm{g},i}(Y)+\omega_{\mathrm{g}}]_+^2 \\ &+\sum_{i<j}\sum_{\tau\in\mathcal{T}_{ij}}\sum_{a,b=1}^{15}[c_{\mathrm{c},ijab\tau}(Y)+\omega_{\mathrm{c}}]_+^2.\end{aligned} \tag{150}$$

Unlike the soft collision cost, this penalty uses unnormalised sums of constraint violations. The small interior guards tighten the penalty only; acceptance uses the original constraints in eq. (147). At inner iterate m, let $J^{(m)}$ denote the active decoded world joints in $\hat{Y}^{(m)} = \varphi(\mathcal{D}_{\ell+1}(Z^{(m)}))$, and let ϵ_{mach} be the machine epsilon of their data type. The detached coordinate scale and guards are

$$\begin{aligned}S_m &:= \mathrm{stopgrad}\Big[\max\Big(1, \|J^{(m)}\|_\infty, \max_i \|g_i\|_\infty\Big)\Big], \\ \omega_{\mathrm{c}}^{(m)} &:= \max(10^{-7}, 64\epsilon_{\mathrm{mach}} S_m), \\ \omega_{\mathrm{g}}^{(m)} &:= \min(10^{-3}\varepsilon_{\mathrm{g}}, \omega_{\mathrm{c}}^{(m)}).\end{aligned} \tag{151}$$

The infinity norms denote the largest absolute coordinate. These guards are recomputed at every inner iterate and held fixed when differentiating that iterate's objective:

$$\mathcal{L}_m(Z) := \tfrac{1}{2}\|Z - \widetilde{X}_{\ell+1}\|_2^2 + \rho\,\mathcal{P}_\kappa\Big(\varphi(\mathcal{D}_{\ell+1}(Z)); \omega_{\mathrm{g}}^{(m)}, \omega_{\mathrm{c}}^{(m)}\Big). \tag{152}$$

Projection settings and failure handling. PCD++ permits at most $M = 100$ inner Adam updates per outer transition, with step size $\eta_{\mathrm{p}} = 0.01$, initial penalty $\rho_0 = 10$, and cap $\rho_{\max} = 10^7$. Adam uses $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and zero weight decay, with fresh optimiser state at every outer transition. Every five inner updates, the penalty is multiplied by 10, up to its cap, if the maximum violation has not decreased by more than 5% since the previous reference check. The solver checks all 15×15 capsule pairs for each pair of people at the 596 dense times, and checks goals in each person's final five native frames. It accepts the first verified feasible iterate, with at most M updates and $M + 1$ checks per transition. The accepted state is $Z^{(m)}$, not its denoised estimate, so it remains at the correct noise level for the next DDPM transition. Non-finite states, objectives, residuals, or gradients terminate the affected assembly with failure. Failure to find a feasible iterate within the budget also terminates that assembly, without an infeasible fallback. Failed assemblies count as failures in all binary success metrics.

Algorithm 2 gives the complete procedure. Each completed outer transition uses one guidance backward pass and at most M inner-optimisation backward passes, giving at most $K(1+M) = 5050$ backward passes per assembly under the reported settings. Early acceptance or failure reduces this count; feasibility checks additionally require forward computations. This counts joint backward evaluations, not separate passes for each person; at the final transition, inner gradients pass through the decoder but not through a denoiser, since $\mathcal{D}_K = \mathrm{Id}$. The solver does not certify convergence or global distance optimality. Feasibility covers only the stated goal constraints and capsule proxy at checked times, not continuous-time clearance or the complete kinematic-validity audit.

D.6.6 Additional Experimental Evaluations

Evaluation uses 24 fixed scenes drawn from three training action families: walking, jogging, and crouched walking. Thirteen scenes are homogeneous and eleven mix families. Every method is evaluated with five sampling seeds, giving 120 attempts per method. These seeds measure sampling variability for one controller, not training variability or held-out layout generalisation. Main figures compare uncontrolled MDM, PCD++, and CMDS; the appendix comparison also includes Best-of-256, DPS, and PCD. Matched views use the same sampling seed, identical scene conditions, and a common camera. Capsule poses and footpaths show body clearance and goal progress, with faint intermediate poses and emphasised starts and endpoints. Uncontrolled MDM supplies the visual prior reference; table 15 aggregates the full evaluation. Figure 22 extends the qualitative comparison to all evaluated baselines.

Algorithm 2 PCD++ with a DDPM rollout and joint Tweedie projection

Require: frozen DDPM maps $\{F_\ell^{\mathrm{b}}\}_{\ell=0}^{K-1}$, with $\varsigma_{K-1} = 0$; clean-estimate maps $\{\mathcal{D}_\ell\}_{\ell=0}^{K}$, with $\mathcal{D}_K = \mathrm{Id}$; assembly φ, task κ, and baseline reward r_κ^{base}; constraint residual V_κ and guarded penalty $\mathcal{P}_\kappa$; guidance strength η_{g}; maximum inner updates M; Adam step size η_{p}; penalty parameters $\rho_0, \rho_{\max}$
Conventions: optimise only active coordinates; keep padding zero. Any non-finite state, objective, residual, or gradient returns ProjectionFailure.

Sample $X_0^i \sim \mathcal{N}(0, I)$ and independent rollout noise $\Xi_\ell^i \sim \mathcal{N}(0, I)$, for all i, ℓ, on active coordinates
for $\ell = 0, \dots, K-1$ **do**
  1. Native DDPM proposal and DPS guidance
    $\hat{X}_{1|\ell} \leftarrow \mathcal{D}_\ell(X_\ell), \quad \hat{Y}_{1|\ell} \leftarrow \varphi(\hat{X}_{1|\ell})$
    $v_\ell \leftarrow \nabla_{X_\ell} r_\kappa^{\mathrm{base}}(\hat{Y}_{1|\ell})$ # Differentiate through the frozen denoiser with respect to its input
    $\widetilde{X}_{\ell+1} \leftarrow \mathrm{stopgrad}\big(F_\ell^{\mathrm{b}}(X_\ell, \Xi_\ell) + \eta_{\mathrm{g}} v_\ell\big)$ # Also guide at $\ell = K-1$; no variance or training-reward multiplier
  2. Joint optimisation of the proposed state
    $Z^{(0)} \leftarrow \widetilde{X}_{\ell+1}, \quad \rho \leftarrow \rho_0$; initialise fresh Adam state
    for $m = 0, \dots, M$ **do**
      $\hat{Y}^{(m)} \leftarrow \varphi\big(\mathcal{D}_{\ell+1}(Z^{(m)})\big)$ # Fresh clean estimate at the candidate's next noise level; $\mathcal{D}_K = \mathrm{Id}$
      $q_m \leftarrow V_\kappa(\hat{Y}^{(m)})$
      if $q_m = 0$ **then**
        $X_{\ell+1} \leftarrow \mathrm{stopgrad}(Z^{(m)})$ # Keep the corrected state, not its denoised estimate
        break
      end if
      if $m = M$ **then**
        return ProjectionFailure # Terminate this assembly; no infeasible fallback
      end if
      if $m = 0$ **then**
        $q_{\mathrm{ref}} \leftarrow q_m$
      else if $m \bmod 5 = 0$ **then**
        if $q_m \geq 0.95\, q_{\mathrm{ref}}$ **then**
          $\rho \leftarrow \min(10\rho, \rho_{\max})$
        end if
        $q_{\mathrm{ref}} \leftarrow q_m$
      end if
      Compute detached guards $\omega_{\mathrm{g}}^{(m)}, \omega_{\mathrm{c}}^{(m)}$ from $\hat{Y}^{(m)}$ using eq. (151)
      Define $\mathcal{L}_m(Z) := \frac{1}{2}\|Z - \widetilde{X}_{\ell+1}\|_{\mathrm{active}}^2 + \rho\, \mathcal{P}_\kappa\Big(\varphi(\mathcal{D}_{\ell+1}(Z)); \omega_{\mathrm{g}}^{(m)}, \omega_{\mathrm{c}}^{(m)}\Big)$
      $G_m \leftarrow \nabla_Z \mathcal{L}_m(Z)|_{Z=Z^{(m)}}$ # Hold the guards fixed; one backward pass per inner update
      $Z^{(m+1)} \leftarrow \mathrm{AdamStep}(Z^{(m)}, G_m, \eta_{\mathrm{p}})$
    end for
end for
return $X_K,\ \varphi(X_K)$ # At most $K(1+M)$ backward passes; 5050 for the reported settings

Table 15: **Four-person locomotion coordination.** All methods use the same 24 scenes with walking, jogging, and crouched walking, evaluated over five sampling seeds (120 attempts per method). Entries are mean ± sample standard deviation across the five seed summaries. Binary rates weight all 24 attempts per seed equally, including generation failures as zero successes; continuous geometry and time use available generated outputs. Timing excludes failed-attempt costs. Bold denotes the best mean, including ties.

Method	Joint success ↑ (%)	Collision-free ↑ (%)	Peak body overlap ↓ (cm)	All-person goals ↑ (%)	Goal error ↓ (m)	Kinematic validity ↑ (%)	Time ↓ (s/assembly)
Best-of-256	0.0 ±0.0	16.7 ±5.9	8.860 ±1.086	0.0 ±0.0	2.456 ±0.146	83.3 ±6.6	69.895 ±0.292
DPS	0.0 ±0.0	61.7 ±10.8	3.335 ±1.049	0.0 ±0.0	3.751 ±0.284	57.5 ±6.8	8.566 ±0.018
PCD	4.2 ±2.9	8.3 ±2.9	19.002 ±1.865	**100.0** ±0.0	0.500 ±0.000	27.5 ±4.8	9.985 ±0.064
PCD++	75.0 ±6.6	**98.3** ±2.3	**0.000** ±0.000	98.3 ±2.3	0.487 ±0.001	75.0 ±6.6	145.335 ±4.460
CMDS	**90.0** ±6.3	90.0 ±6.3	0.740 ±0.351	**100.0** ±0.0	**0.158** ±0.008	**100.0** ±0.0	**0.541** ±0.002

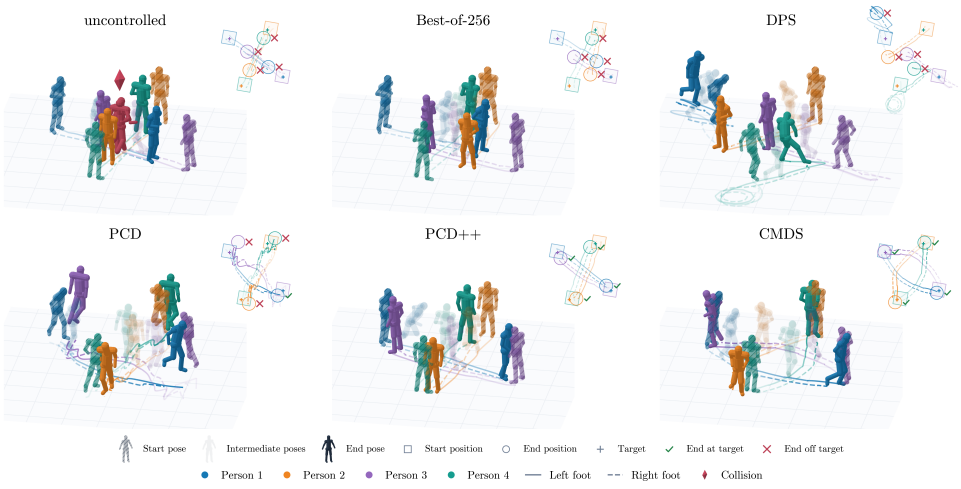


Figure 22: **Extended qualitative comparison of four-person motion generation.** We compare uncontrolled diffusion, Best-of-256, DPS, PCD, PCD++, and CMDS on the same four-person task shown in the main paper. The prompts are *"A person moves forward at a jogging pace."* for Person 1 (●) and Person 4 (●), and *"A person walks forward."* for Person 2 (●) and Person 3 (●). Top-down insets show starts (□), final positions (◯), and targets (+), with surrounding regions indicating the goal tolerance and symbols marking reached (✓) or missed (×) targets. Solid and dashed curves show left- and right-foot trajectories; intermediate poses are faded, while start and final poses are opaque. Red diamonds (♦) mark simultaneous inter-person body overlap. In this example, uncontrolled diffusion, Best-of-256, DPS, and PCD do not satisfy the full joint goal-and-collision constraint, whereas PCD++ and CMDS reach all goals without inter-person collisions.