ICML
International Conference
On Machine Learning

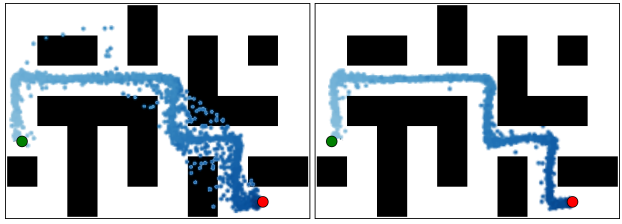
Improving Diffusion Planners by Self-Supervised Action Gating with Energies

Yuan Lu[1] Dongqi Han[2] Yansen Wang[2] Dongsheng Li[2]

[1]University College London, [2]Microsoft Research

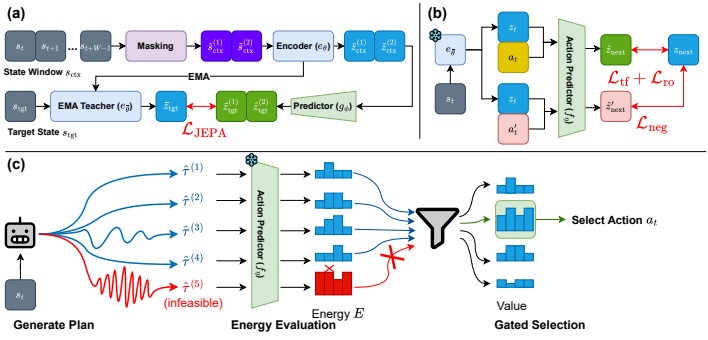
Why BC diffusion planners still fail

- Diffusion planners sample many future trajectories and rank them with a learned value score.
- But the highest-value candidate can have an infeasible early prefix.
- In receding-horizon control, a bad prefix is costly because execution commits to it before replanning.
- We want a **feasibility signal** that is separate from reward/value and can be added at test time.



Left: value-only selection can drift through obstacles. Right: SAGE suppresses those infeasible prefixes.

SAGE: inference-time feasibility gating



Stage 1: pretrain a JEPA encoder on offline state windows.

Stage 2: train an action-conditioned latent predictor for short transitions.

Inference: compute an energy on the first K steps of each candidate plan and gate value-based ranking.

SAGE score

$$E(\hat{\tau}) = \frac{1}{K}\sum_{k=0}^{K-1} \left\| f_\eta(z_k, a_k) - z_{k+1} \right\|_1, \qquad i^* = \arg\max_{i \in \mathcal{I}_t} J(\hat{\tau}^{(i)}) - \lambda E(\hat{\tau}^{(i)})$$

Low energy means the prefix looks locally consistent with dataset-supported dynamics.

Training and Sampling SAGE

Algorithm 1 Encoder Pre-Training

Require: Offline data $\mathcal{D}$, window W, offsets $\mathcal{K}$

Initialise Encoder e_θ, EMA teacher $e_{\bar{\theta}}$, predictor g_ϕ
for each step **do**
  Sample t and $\mathcal{K}$; set $s_{\text{ctx}} \leftarrow (s_t, \dots, s_{t+W-1})$
  Views: $\tilde{s}^{(1)}_{\text{ctx}}, \tilde{s}^{(2)}_{\text{ctx}} \leftarrow \text{Mask}(s_{\text{ctx}})$
  Targets: $s^{(k)}_{\text{tgt}} \leftarrow s_{t+W-1+k}$ for $k \in \mathcal{K}$
  $z^{(i)}_{\text{ctx}} \leftarrow e_\theta(\tilde{s}^{(i)}_{\text{ctx}}), \quad \bar{z}^{(k)}_{\text{tgt}} \leftarrow e_{\bar{\theta}}(s^{(k)}_{\text{tgt}})$
  $\hat{z}^{(i,k)}_{\text{tgt}} \leftarrow g_\phi(z^{(i)}_{\text{ctx}}, k)$
  Incur loss $\mathcal{L} \leftarrow \sum_{i\in\{1,2\},k\in\mathcal{K}} \mathcal{L}_{\text{JEPA}}\big(\hat{z}^{(i,k)}_{\text{tgt}}, \bar{z}^{(k)}_{\text{tgt}}\big)$
  Update θ, ϕ; update EMA $\bar{\theta}$
end for
Return frozen $e_{\bar{\theta}}$

Algorithm 2 SAGE Inference

Require: Planner p_θ, encoder $e_{\bar{\theta}}$, predictor f_η, score J, prefix K, keep-rate $\mathcal{P}$, penalty λ

for each decision step t **do**
  Sample $\{\hat{\tau}^{(i)}_t\}_{i=1}^{C} \sim p_\theta(\tau \mid s_t)$
  Compute $\{E(\hat{\tau}^{(i)}_t)\}_{i=1}^{C}$ on the first K steps (Eq. 11)
  Keep $\mathcal{I}_t \leftarrow$ lowest-energy $\mathcal{P}$ fraction
  Choose $i^* \leftarrow \arg\max_{i\in\mathcal{I}_t} \big(J(\hat{\tau}^{(i)}_t) - \lambda E(\hat{\tau}^{(i)}_t)\big)$
  Execute $a_t \leftarrow \hat{a}^{(i^*)}_t$ and observe s_{t+1}
end for

In Stage II, we learn an action-conditioned short-horizon predictor in the frozen JEPA latent space. The Predictor f_η is trained with three complementary objectives.

Teacher-forced one-step loss. This encourages accurate next-latent prediction under ground-truth prefixes:

$$\mathcal{L}_{\text{tf}} = \sum_{j=0}^{W-1} \left\| \hat{z}_{t+1+j} - z_{t+1+j} \right\|_1 .$$

Short-horizon rollout loss. This enforces consistency under autoregressive application of f_η:

$$\mathcal{L}_{\text{ro}} = \left\| \hat{z}_{t+H_{\text{ro}}} - z_{t+H_{\text{ro}}} \right\|_1 .$$

Here $\hat{z}_{t+H_{\text{ro}}}$ is obtained by rolling out f_η from z_t using $a_{t:t+H_{\text{ro}}-1}$.

Action-usage hinge. We permute actions within the batch to form a' and compute

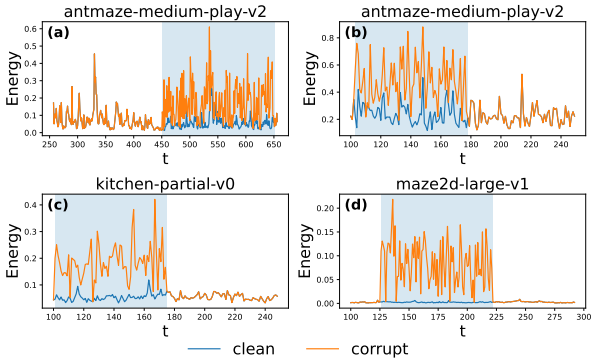
$$E_{\text{neg}} = \sum_j \left\| \hat{z}'_{t+1+j} - z_{t+1+j} \right\|_1 , \qquad \mathcal{L}_{\text{neg}} = \big[m - E_{\text{neg}}\big]_+ .$$

The final action-conditioned objective is

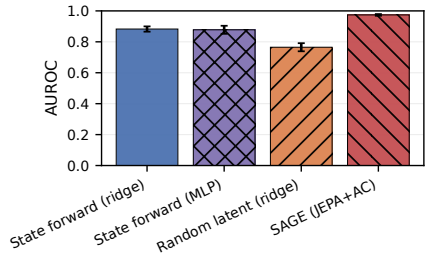
$$\mathcal{L}_{\text{AC}} = \mathcal{L}_{\text{tf}} + \lambda_{\text{ro}}\mathcal{L}_{\text{ro}} + \lambda_{\text{neg}}\mathcal{L}_{\text{neg}}.$$

After training, f_η is frozen and used to compute the consistency energy.

Does the energy really measure feasibility?



Clean transitions stay low-energy; energy in the corrupted window increases sharply.

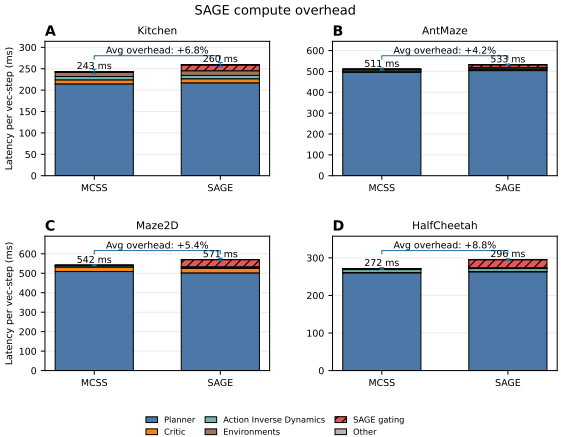


- Best feasibility discrimination among compared predictors.
- Mean AUROC is about **0.98**, above state-forward and random-latent baselines.
- It acts like an executability test for candidate prefixes.

Main benchmark result: better planning by better selection

		IL	Non-Diffusion Policies			Diffusion Policies			Diffusion Planners				
Dataset	**Environment**	BC	BCQ	CQL	IQL	DQL*	IDQL*	Diffuser*	RGG	LoMAP	LDCQ	DV*	SAGE (Ours)
Medium-Expert	HalfCheetah	35.8	64.7	62.4	86.7	95.5 ± 0.1	95.9	88.9 ± 0.3	90.8 ± 0.3	91.1 ± 0.2	90.2 ± 0.9	92.7 ± 0.3	95.4 ± 0.1
Medium-Expert	Hopper	111.9	110.9	98.7	91.5	111.1 ± 1.4	108.6	103.3 ± 1.3	109.6 ± 2.3	110.6 ± 0.3	113.3 ± 0.2	108.8 ± 0.5	111.3 ± 0.5
Medium-Expert	Walker2d	6.4	57.5	110.0	109.6	111.6 ± 0.0	112.7	106.9 ± 0.2	107.8 ± 0.1	109.2 ± 0.1	109.3 ± 0.4	108.6 ± 0.0	109.4 ± 0.1
Medium	HalfCheetah	36.1	40.7	44.4	47.4	52.3 ± 0.2	51.0	42.8 ± 0.3	44.0 ± 0.3	45.4 ± 0.1	42.8 ± 0.7	50.4 ± 0.0	51.6 ± 0.0
Medium	Hopper	29.0	57.5	58.0	66.3	96.5 ± 1.3	65.5	74.3 ± 1.4	82.5 ± 4.3	93.7 ± 1.5	66.2 ± 1.7	80.9 ± 1.2	83.9 ± 1.2
Medium	Walker2d	6.6	53.1	79.2	78.3	86.8 ± 0.2	85.5	79.6 ± 0.6	81.7 ± 0.5	79.9 ± 1.2	69.4 ± 3.5	82.8 ± 0.1	84.8 ± 0.1
Medium-Replay	HalfCheetah	38.4	38.2	46.2	44.2	47.9 ± 0.0	45.9	37.7 ± 0.5	41.0 ± 0.2	39.1 ± 1.0	41.8 ± 0.4	45.8 ± 0.1	46.5 ± 0.2
Medium-Replay	Hopper	11.3	33.1	48.6	94.7	101.6 ± 0.0	92.1	93.6 ± 0.4	95.2 ± 0.5	97.6 ± 0.6	86.3 ± 2.5	91.6 ± 0.0	91.8 ± 0.0
Medium-Replay	Walker2d	11.8	15.0	26.7	73.9	98.2 ± 0.1	85.1	70.6 ± 1.6	78.3 ± 4.4	78.7 ± 2.2	68.5 ± 4.3	84.1 ± 0.5	85.3 ± 0.3
Average		31.9	52.0	63.9	77.0	**89.1**	82.1	77.5	81.2	82.8	81.6	82.9	**84.4**
Mixed	Kitchen	47.5	8.1	51.0	51.0	55.1 ± 1.6	66.5	52.5 ± 2.5	—	—	62.3 ± 0.5	73.6 ± 0.1	74.5 ± 0.3
Partial	Kitchen	33.8	18.9	49.8	46.3	65.5 ± 1.4	66.7	55.7 ± 1.3	—	—	67.8 ± 0.8	90.0 ± 0.4	96.6 ± 0.2
Average		40.7	13.5	50.4	48.7	60.3	66.6	54.1	—	—	65.1	81.8	**85.6**
Antmaze-Large	Diverse	0.0	2.2	61.2	47.5	70.6 ± 3.7	67.9	27.3 ± 2.4	—	39.3 ± 2.5	57.7 ± 1.8	76.0 ± 1.8	77.0 ± 1.7
Antmaze-Large	Play	0.0	6.7	53.7	39.6	81.3 ± 3.1	63.5	17.3 ± 1.9	—	20.7 ± 3.8	—	76.4 ± 2.0	82.1 ± 1.9
Antmaze-Medium	Diverse	0.0	0.0	15.8	70.0	82.6 ± 3.0	84.8	2.0 ± 1.6	—	36.0 ± 3.7	68.9 ± 0.7	85.1 ± 1.3	88.0 ± 1.7
Antmaze-Medium	Play	0.0	0.0	14.9	71.2	87.3 ± 2.7	84.5	6.7 ± 5.7	—	40.7 ± 4.3	—	89.0 ± 1.6	91.0 ± 2.0
Average		0.0	2.2	36.4	57.1	80.5	75.2	13.3	—	34.2	—	81.6	**84.5**
Maze2D	Large	5.0	6.2	12.5	58.6	186.8 ± 1.7	90.1	123.0	148.3 ± 1.4	151.9 ± 2.7	150.1 ± 2.9	197.4 ± 1.6	200.6 ± 1.4
Maze2D	Medium	30.3	8.3	5.0	34.9	152.0 ± 0.8	89.5	121.5	130.0 ± 0.9	131.0 ± 0.9	125.3 ± 2.5	150.7 ± 1.1	150.8 ± 1.0
Maze2D	Umaze	3.8	12.8	5.7	47.4	140.6 ± 1.0	57.9	113.9	128.3 ± 0.8	126.0 ± 0.3	134.2 ± 4.0	136.8 ± 1.3	137.9 ± 1.0
Average		13.0	9.1	7.7	47.0	159.8	79.2	119.5	135.5	136.3	136.5	161.6	**163.1**

Why this is practical



- Only lightweight encoder/predictor evaluations on a short prefix.
- Overhead is modest: about **4% to 9%** across domains.
- A **plug-in reliability layer** for existing diffusion planners.

Take-home message

Value says desirable; SAGE energy says executable. Together they plan better.