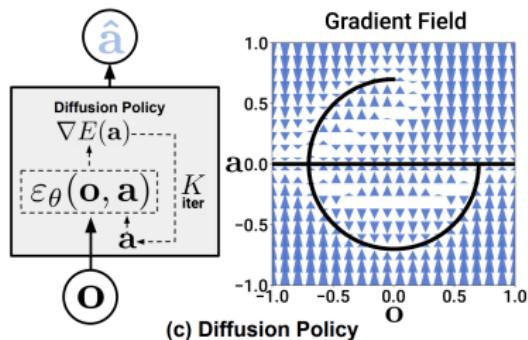


Diffusion Policy

Visuomotor Policy Learning via Action Diffusion

A. Buynitsky

Nov 7th, 2024



Outline

- ① TLDR
- ② Related Work
- ③ Background & Intuition
- ④ Diffusion for Visuomotor Policies
- ⑤ Diffusion Properties
- ⑥ Key Findings

Outline

① TLDR

② Related Work

③ Background & Intuition

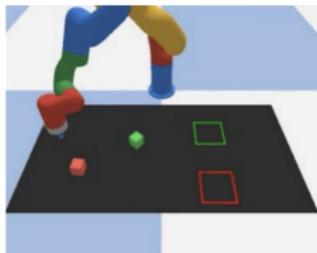
④ Diffusion for Visuomotor Policies

⑤ Diffusion Properties

⑥ Key Findings

High Level: Learn visiomotor policy through a diffusion process

Results: 46.9% average improvement across 12 tasks (sim + real life)



Block Push

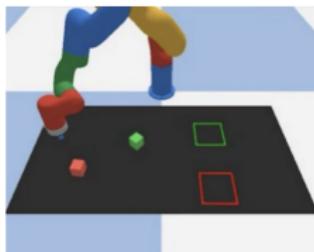


Kitchen

	BlockPush		Kitchen			
	p1	p2	p1	p2	p3	p4
LSTM-GMM [29]	0.03	0.01	1.00	0.90	0.74	0.34
IBC [12]	0.01	0.00	0.99	0.87	0.61	0.24
BET [42]	0.96	0.71	0.99	0.93	0.71	0.44
DiffusionPolicy-C	0.36	0.11	1.00	1.00	1.00	0.99
DiffusionPolicy-T	0.99	0.94	1.00	0.99	0.99	0.96

High Level: Learn visiomotor policy through a diffusion process

Results: 46.9% average improvement across 12 tasks (sim + real life)



Block Push



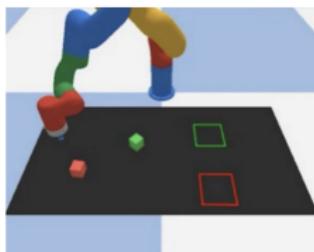
Kitchen

Research Group: Shuran Song;
MIT, Columbia and Toyota Research
Institute (TRI)

	BlockPush		Kitchen			
	p1	p2	p1	p2	p3	p4
LSTM-GMM [29]	0.03	0.01	1.00	0.90	0.74	0.34
IBC [12]	0.01	0.00	0.99	0.87	0.61	0.24
BET [42]	0.96	0.71	0.99	0.93	0.71	0.44
DiffusionPolicy-C	0.36	0.11	1.00	1.00	1.00	0.99
DiffusionPolicy-T	0.99	0.94	1.00	0.99	0.99	0.96

High Level: Learn visiomotor policy through a diffusion process

Results: 46.9% average improvement across 12 tasks (sim + real life)



Block Push



Kitchen

	BlockPush		Kitchen			
	p1	p2	p1	p2	p3	p4
LSTM-GMM [29]	0.03	0.01	1.00	0.90	0.74	0.34
IBC [12]	0.01	0.00	0.99	0.87	0.61	0.24
BET [42]	0.96	0.71	0.99	0.93	0.71	0.44
DiffusionPolicy-C	0.36	0.11	1.00	1.00	1.00	0.99
DiffusionPolicy-T	0.99	0.94	1.00	0.99	0.99	0.96

Research Group: Shuran Song;
MIT, Columbia and Toyota Research
Institute (TRI)

Future Work:

- Apply Diffusion Policy to RL to avoid suboptimal data
- Accelerate inference time + decrease computational costs with advancements from modern diffusion methods

Outline

① TLDR

② Related Work

③ Background & Intuition

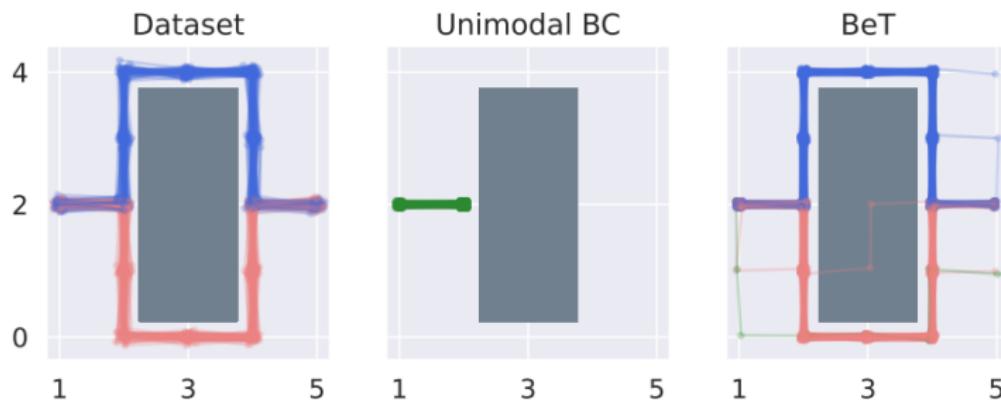
④ Diffusion for Visuomotor Policies

⑤ Diffusion Properties

⑥ Key Findings

Behavior Transformer (BeT)

Large Scale human datasets have wide variances with multiple modes



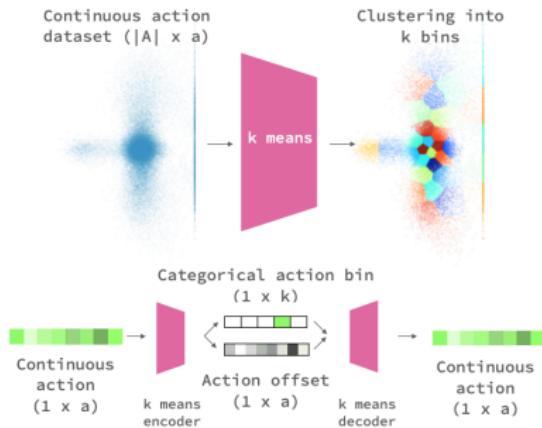
Limitation of BC: Current bc assume that data is drawn from unimodal experts solving a single task

Behavior Transformer (BET)

Proposed Solution:

1. Descretize action dataset into k bins with k-means with offset
2. Train a transformer with two heads to predict an action bin and an offset for that bin

A. Continuous action binning

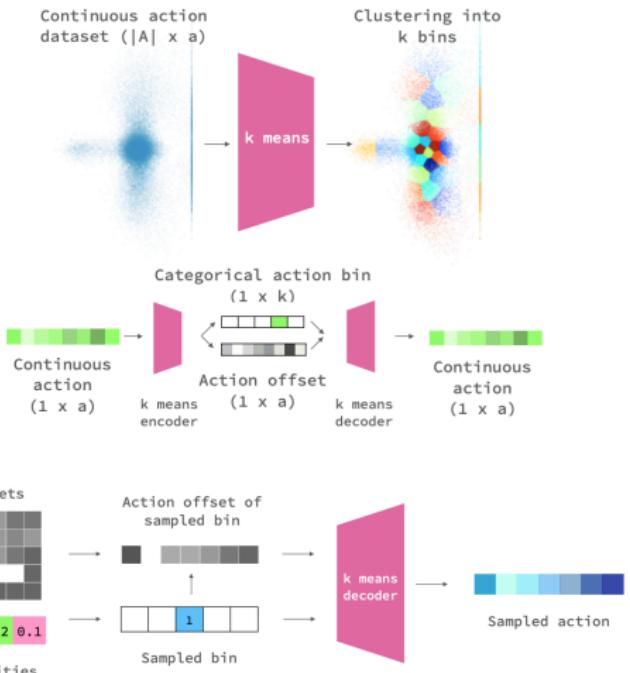


Behavior Transformer (BET)

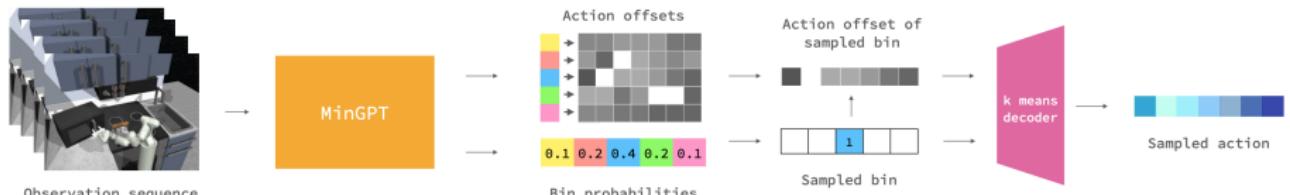
Proposed Solution:

1. Descretize action dataset into k bins with k-means with offset
2. Train a transformer with two heads to predict an action bin and an offset for that bin

A. Continuous action binning



Inference:

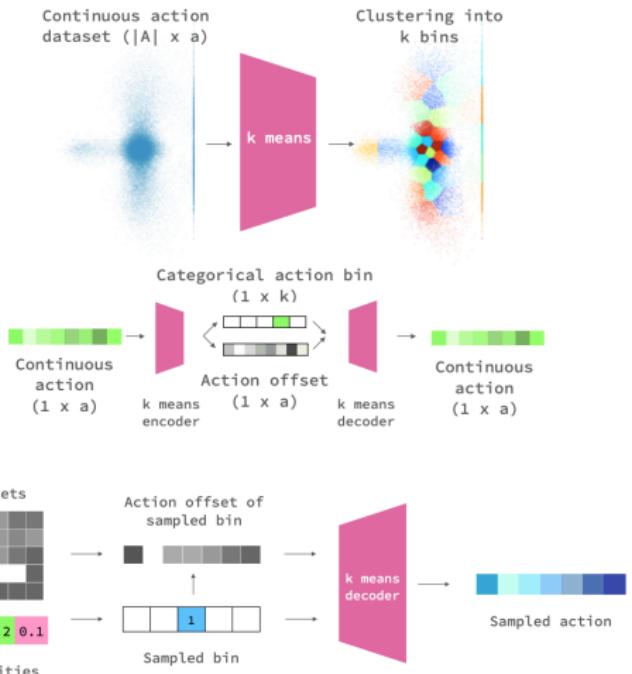


Behavior Transformer (BET)

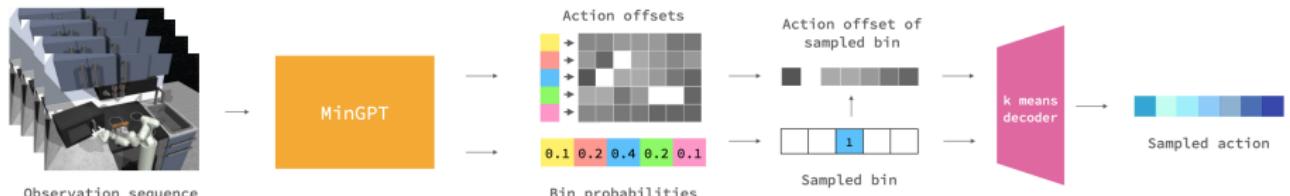
Proposed Solution:

1. Descretize action dataset into k bins with k-means with offset
2. Train a transformer with two heads to predict an action bin and an offset for that bin

A. Continuous action binning



Inference:

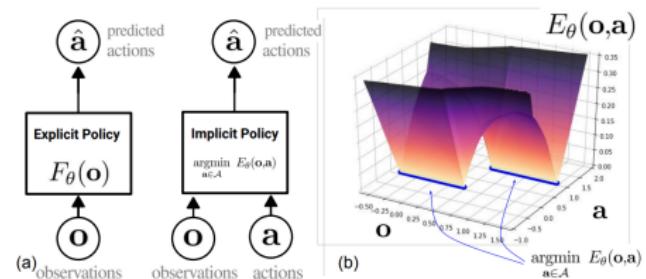


Limitation: Inconsistent sampling from action bins

Implicit BC

BC Reformulation: Use Energy-Based models instead of explicit policies

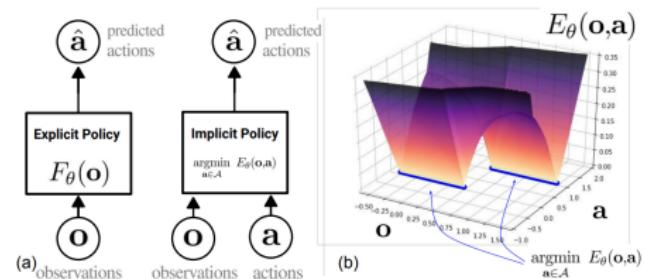
Energy Models: Scalar energy function captures relationship between states and actions



Implicit BC

BC Reformulation: Use Energy-Based models instead of explicit policies

Energy Models: Scalar energy function captures relationship between states and actions



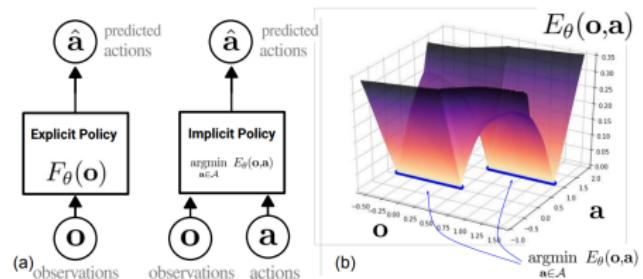
Inference: solve optimization problem (i.e. with gradient descent) for:

$$\hat{a} = \arg \min_{a \in A} E(o, a)$$

Implicit BC

BC Reformulation: Use Energy-Based models instead of explicit policies

Energy Models: Scalar energy function captures relationship between states and actions



Inference: solve optimization problem (i.e. with gradient descent) for:

$$\hat{a} = \arg \min_{a \in A} E(o, a)$$

Limitation: Increases complexity, training and inference time, and computational cost

BC-RNN / LSTM-RNN

Behavioral Cloning learns a policy $\pi_\theta(s)$ to clone actions from a set of demonstrations:

$$\arg \min_{\theta} \mathbb{E}_{(s,a) \sim D} \|\pi_\theta(s) - a\|^2$$

BC-RNN / LSTM-RNN

Behavioral Cloning learns a policy $\pi_\theta(s)$ to clone actions from a set of demonstrations:

$$\arg \min_{\theta} \mathbb{E}_{(s,a) \sim D} \|\pi_\theta(s) - a\|^2$$

Method:

- RNN models temporal dependencies $(s_t, a_t, \dots, s_{t+T}, a_{t+T})$ via hidden state
- predicts sequence of action and is unrolled one step at a time:

$$a_t, h_{t+1} = \pi_\theta(s_t, h_t)$$

BC-RNN / LSTM-RNN

Behavioral Cloning learns a policy $\pi_\theta(s)$ to clone actions from a set of demonstrations:

$$\arg \min_{\theta} \mathbb{E}_{(s,a) \sim D} \|\pi_\theta(s) - a\|^2$$

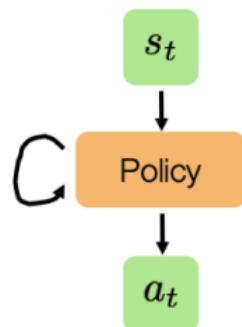
Method:

- RNN models temporal dependencies $(s_t, a_t, \dots, s_{t+T}, a_{t+T})$ via hidden state
- predicts sequence of action and is unrolled one step at a time:

$$a_t, h_{t+1} = \pi_\theta(s_t, h_t)$$

Limitation

1. Biased towards one mode
2. Brittle performance due to changes in observation space and hyperparameters



Outline

- ① TLDR
- ② Related Work
- ③ Background & Intuition
- ④ Diffusion for Visuomotor Policies
- ⑤ Diffusion Properties
- ⑥ Key Findings

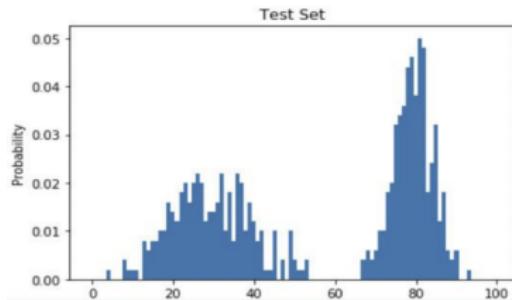
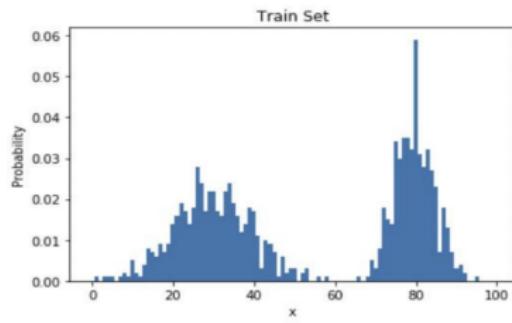
Learning

Assume that all Data (image, speech, etc.) is sampled from a distribution

Learning

Assume that all Data (image, speech, etc.) is sampled from a distribution

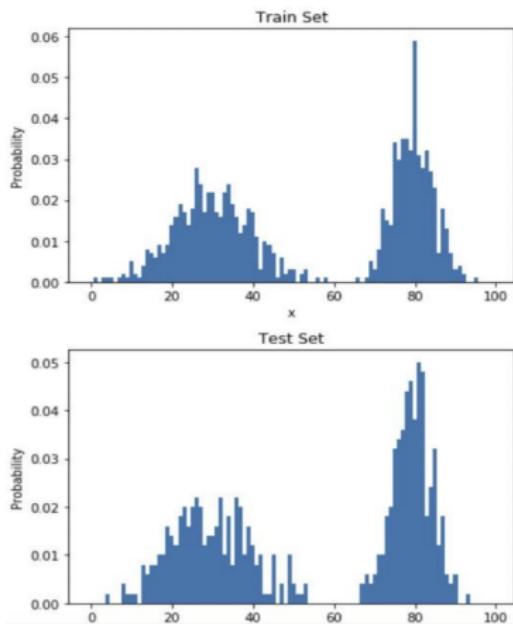
Generative Model: learn training distribution $P_\theta(x)$ from dataset



Learning

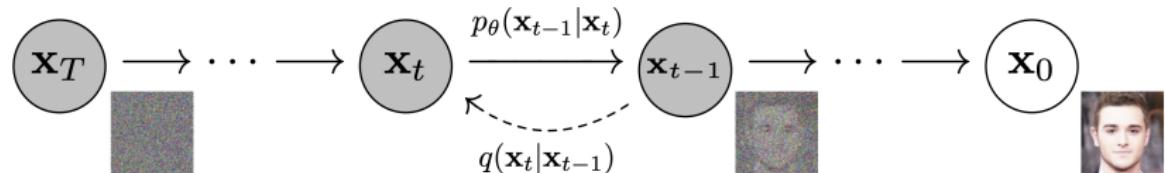
Assume that all Data (image, speech, etc.) is sampled from a distribution

Generative Model: learn training distribution $P_\theta(x)$ from dataset



Diffusion Models model the output as a denoising process, performing T denoising steps to produce a series of intermediate actions with decreasing levels of noise.

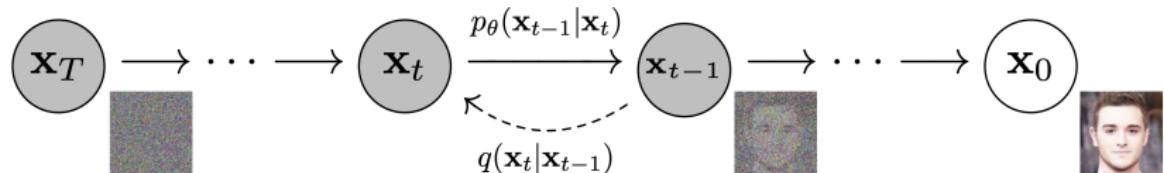
Diffusion Overview (Forward Process)



Forward Process ($q(x_t | x_{t-1})$): Markov chain that gradually adds Gaussian Noise according to variance schedule $\beta_t \in [\beta_1, \dots, \beta_T]$

$$q(x_{1:T} | x_0) := \prod_{t=1}^T q(x_t | x_{t-1}), \quad q(x_t | q_{t-1}) := \mathbf{N}(x_t; \sqrt{1 - \beta_t} x_{t-1}, \beta_T \mathbf{I})$$

Diffusion Overview (Forward Process)



Forward Process ($q(x_t | x_{t-1})$): Markov chain that gradually adds Gaussian Noise according to variance schedule $\beta_t \in [\beta_1, \dots, \beta_T]$

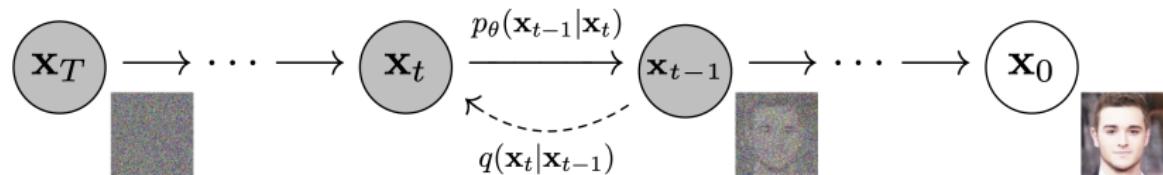
$$q(x_{1:T} | x_0) := \prod_{t=1}^T q(x_t | x_{t-1}), \quad q(x_t | q_{t-1}) := \mathbf{N}(x_t; \sqrt{1 - \beta_t} x_{t-1}, \beta_T \mathbf{I})$$

Reparametrization for $q(x_t | x_0)$ allows for sampling x_t at arbitrary t :

$$q(x_t | x_0) = N(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) \mathbf{I}) \quad \text{where:}$$

$$\bar{\alpha}_t := \prod_{s=1}^t \alpha_s \quad \text{and} \quad \alpha_s = 1 - \beta_t$$

Diffusion Overview (Backward Process)



Backward Process ($p_\theta(x_{t-1} | x_t)$): Markov chain with learned Gaussian transitions initialized with $p(x_T) = \mathbf{N}(x_T; 0, \mathbf{I})$

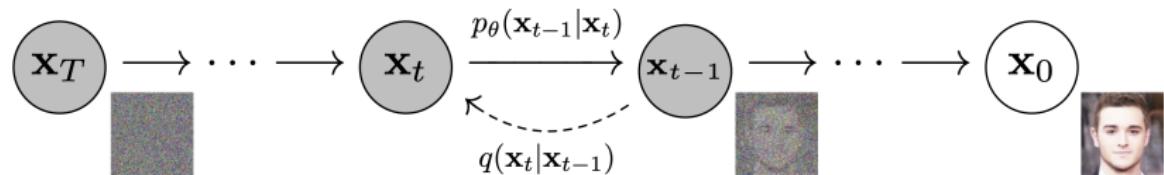
$$p_\theta(x_{0:T}) := p(x_t) \prod_{t=1}^T p_\theta(x_{t-1} | x_t) \quad (1)$$

$$p_\theta(x_{t-1} | x_t) := \mathbf{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (2)$$

$\Sigma_\theta(x_t, t) = \sigma_t^2 \mathbf{I}$: Fix to untrained time dependent constants (i.e. $\sigma_t^2 = \beta_t$)

$\mu_\theta(x_t, t)$: Reparametrize to predict the noise at step t with $\epsilon_\theta(x_t, t)$

Diffusion Overview (Backward Process)



Backward Process ($p_\theta(x_{t-1} | x_t)$): Markov chain with learned Gaussian transitions initialized with $p(x_T) = \mathbf{N}(x_T; 0, \mathbf{I})$

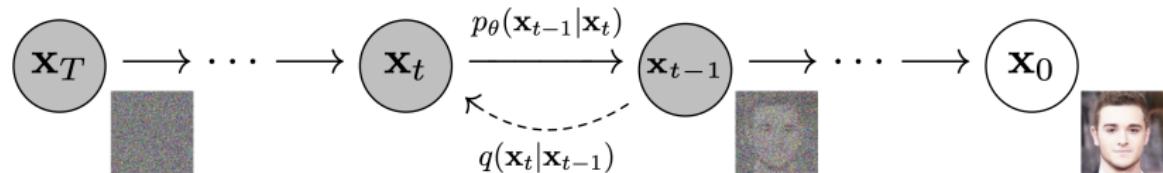
$$p_\theta(x_{0:T}) := p(x_t) \prod_{t=1}^T p_\theta(x_{t-1} | x_t) \quad (1)$$

$$p_\theta(x_{t-1} | x_t) := \mathbf{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (2)$$

$\Sigma_\theta(x_t, t) = \sigma_t^2 \mathbf{I}$: Fix to untrained time dependent constants (i.e. $\sigma_t^2 = \beta_t$)

$\mu_\theta(x_t, t)$: Reparametrize to predict the noise at step t with $\epsilon_\theta(x_t, t)$

Diffusion Overview (Backward Process)



Backward Process ($p_\theta(x_{t-1} | x_t)$): Markov chain with learned Gaussian transitions initialized with $p(x_T) = \mathbf{N}(x_T; 0, \mathbf{I})$

T Denoising iterations performed with decreasing levels of noise producing:
 x_T, x_{T-1}, \dots, x_0 following:

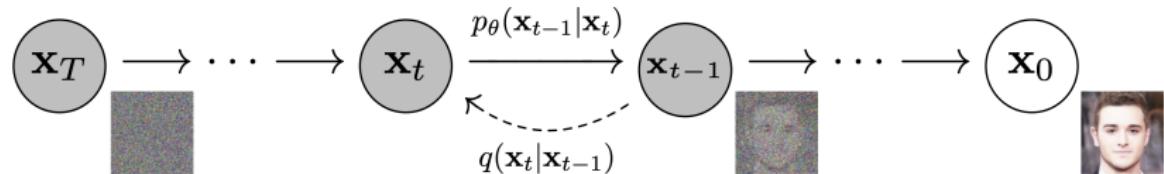
$$x_{t-1} = \alpha(x_t - \gamma \epsilon_\theta(x_t, t) + \mathbf{N}(0, \sigma^2, I)) \quad (3)$$

Above can be interpreted as a single noisy gradient step:

$$x' = x - \gamma \nabla E(x)$$

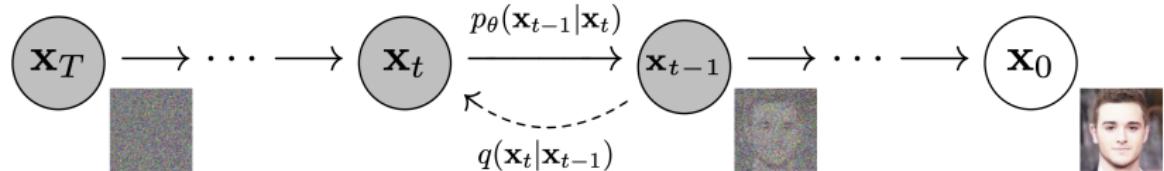
Where $\epsilon_\theta(x, t)$ predicts $E(x)$ with learning rate γ and parameters α and σ^2

Diffusion Training



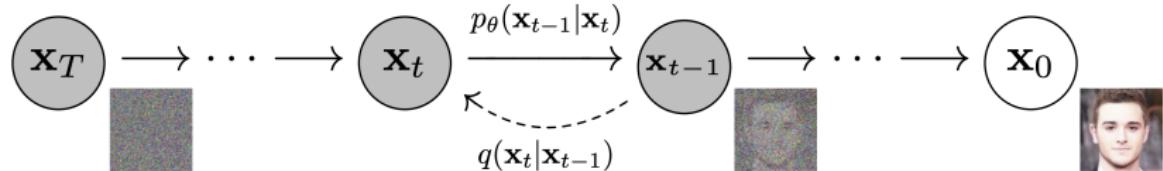
1. Randomly sampling examples x_0 from training dataset

Diffusion Training



1. Randomly sampling examples x_0 from training dataset
2. For each sample randomly select denoising iteration $k \in [1, \dots, T]$ and noise ϵ_k with appropriate variance for k

Diffusion Training



1. Randomly sampling examples x_0 from training dataset
2. For each sample randomly select denoising iteration $k \in [1, \dots, T]$ and noise ϵ_k with appropriate variance for k
3. Noise Prediction Network ϵ_θ predicts noise from noisy data:

$$\mathcal{L} = MSE(\epsilon_k, \epsilon_\theta(x_0 + \epsilon_k, k)) \quad (4)$$

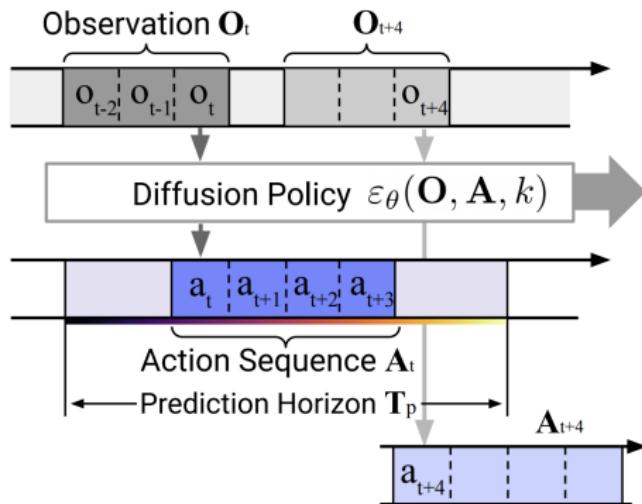
Outline

- ① TLDR
- ② Related Work
- ③ Background & Intuition
- ④ Diffusion for Visuomotor Policies
- ⑤ Diffusion Properties
- ⑥ Key Findings

Modifying Image Diffusion to Robotic Diffusion

Action Sequence Prediction:

- At step t policy takes latest T_o steps of observation data O_t
- Policy predicts T_p action steps
- The first T_a of T_p steps are executed



Modifying Image Diffusion to Robotic Diffusion

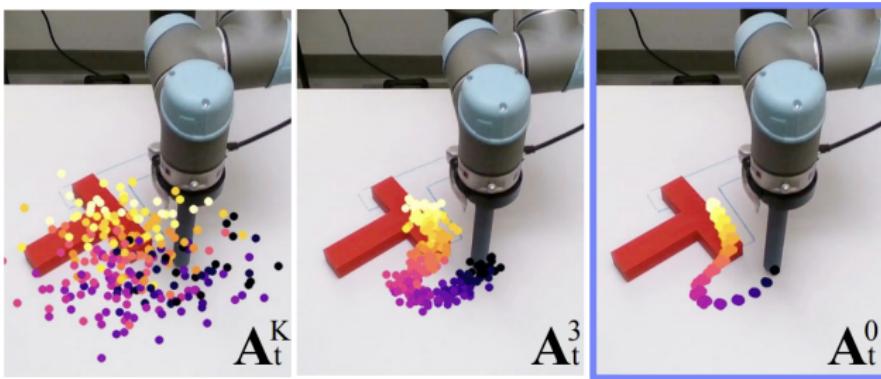
Visual Observation Conditioning: Learn conditional distribution $p(A_t | O_t)$ instead of joint distribution $p(A_t, O_t)$

Modify denoising iterations (3) to:

$$A_t^{k-1} = \alpha(A_t^k - \gamma\epsilon_\theta(O_t, A_t^k, k) + \mathbf{N}(0, \sigma^2, I))$$

Modify MSE Loss (4) to:

$$\mathcal{L} = MSE(\epsilon_k, \epsilon_\theta(O_t, A_t^0 + \epsilon^k, k))$$



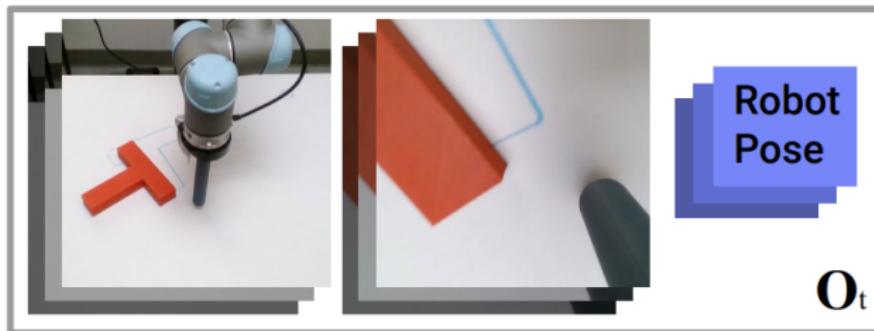
Vision Encoder

Choice of noise prediction network ϵ_θ is **independent** of visual encoders

Vision encoder maps raw image sequence to latent embedding O_t from multiple camera views

Use a standard **Resnet-18** with the following modifications:

- Replace global average pooling with spatial softmax pooling
- Replace BatchNorm with GroupNorm



End-to-end performs better than pre-trained vision encoders.

Option 1: CNN-Based Diffusion Policy

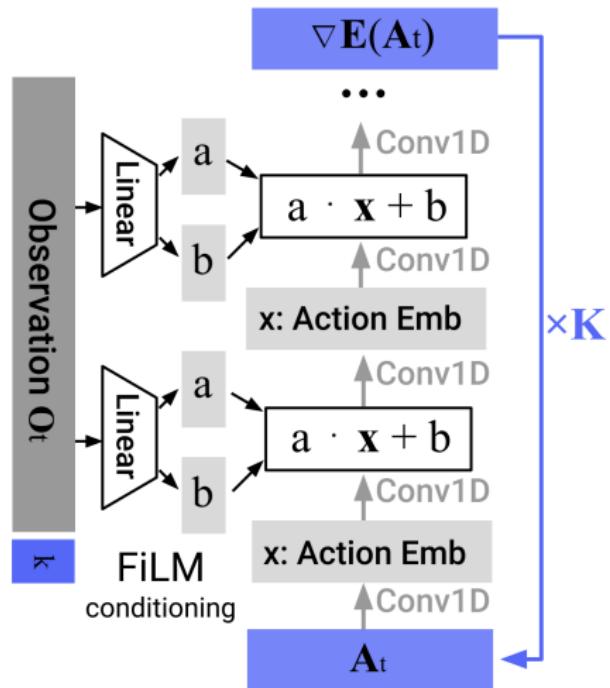
1D temporal CNN: learn training distribution $P_\theta(x)$ from dataset

Method:

- Condition on O_t using FiLM layers
- Only Predict Action Trajectory

Result:

- Ok performance on most tasks
- Poor performance when desired action sequence changes rapidly



Option 2: Time-series Diffusion Transformer

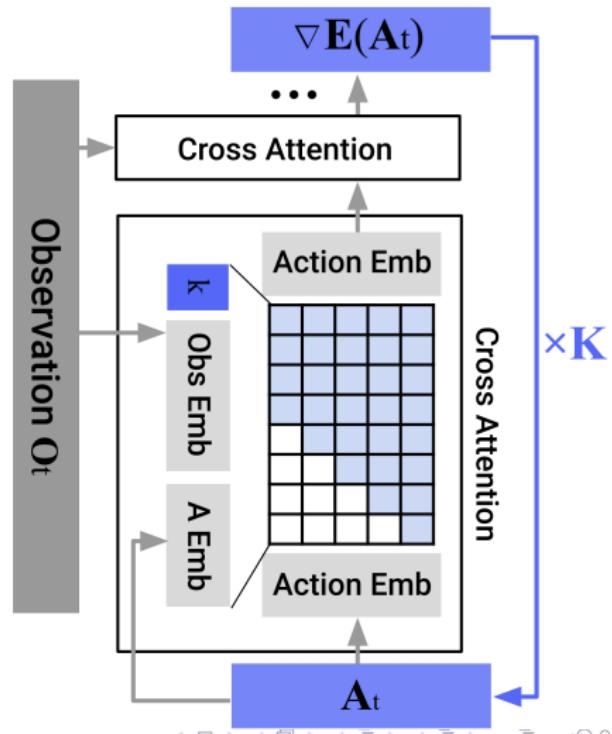
Time-Series Transformer: Adopted from MinGPT

Transformer Input:

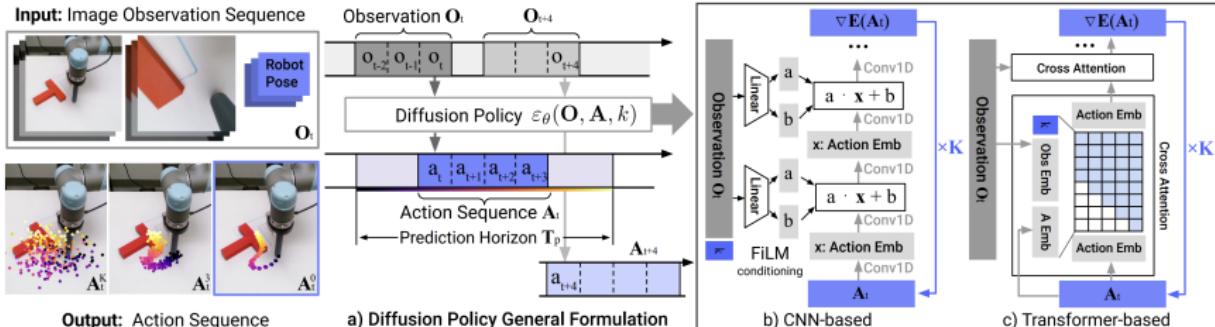
- Noisy Actions A_t^k
- Embedding for diffusion iteration k
- Observations O_t transformed into input tokens by MLP

Result:

- Best performing policy on even complicated tasks
- Sensitive to hyperparameter tuning



Final Result:



Outline

- ① TLDR
- ② Related Work
- ③ Background & Intuition
- ④ Diffusion for Visuomotor Policies
- ⑤ Diffusion Properties
- ⑥ Key Findings

Convergence Basins

A_t^K is stochastically initialized:

- Help specify different convergence basins for final action prediction A_t^0

A_t^K is stochastically perturbed:

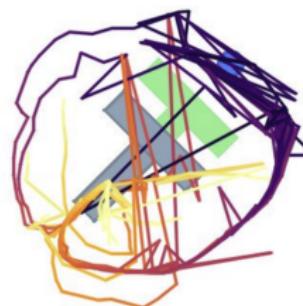
- K iterations enable individual action samples to converge and move between different action basins.



Diffusion Policy



LSTM-GMM



BET



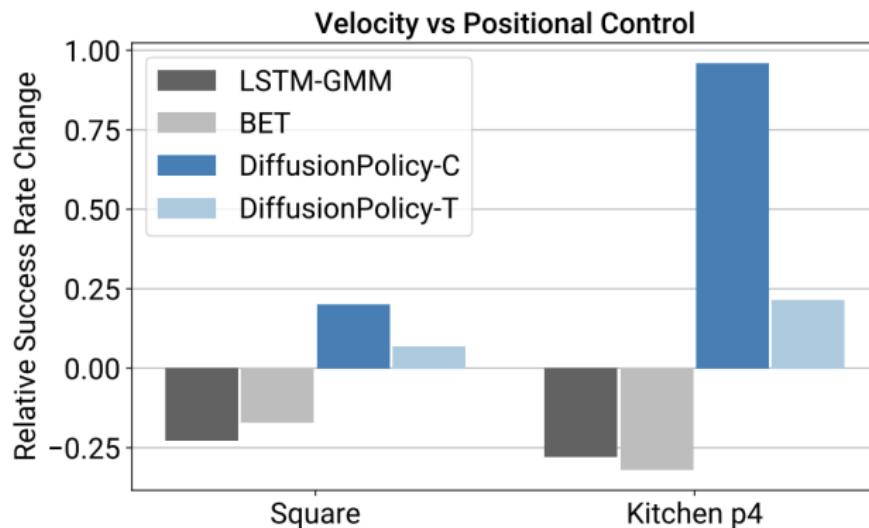
IBC

Robotic Control

Diffusion Policy with position-control outperforms velocity-control

Reasoning:

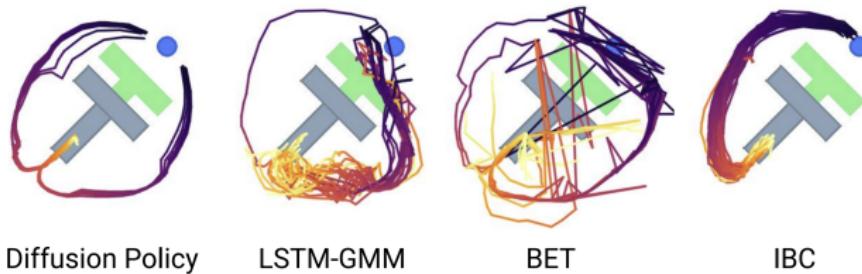
- action multimodality more pronounced in position-control
- position control suffers less from compounding errors than velocity control



Benefits of Action Prediction

Temporal Action Consistency: DDPM represents action in a high-dimensional action sequence overcoming:

1. **Temporal Action Consistency:** Alternate approach (BC-RNN / BET) represent actions as independent multimodal distributions leading to jittery actions
2. **Robustness to Idle Actions:** Single step policies overfit to pausing behavior (BC-RNN + IBC get stuck)



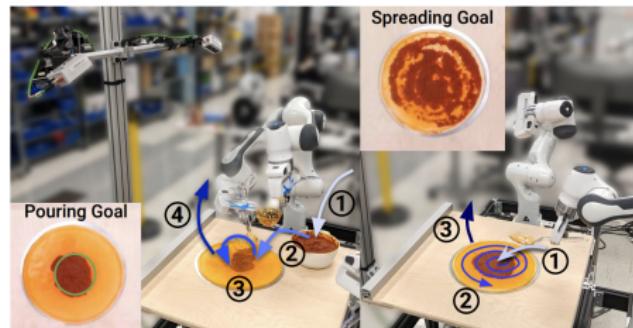
Outline

- ① TLDR
- ② Related Work
- ③ Background & Intuition
- ④ Diffusion for Visuomotor Policies
- ⑤ Diffusion Properties
- ⑥ Key Findings

Diffusion Policy Benefits

Duffusion policy ...

1. expresses short and long horizon multimodality
2. better leverages position control
 - Baseline methods work better with velocity control
3. has a better action horizon tradeoff
 - Horizon > 1 help policy predict consistent actions
 - Compensate for idle portions
 - Optimal action horizon of 8 steps



	Pour		Spread	
	IoU	Succ	Coverage	Succ
Human	0.79	1.00	0.79	1.00
LSTM-GMM	0.06	0.00	0.27	0.00
Diffusion Policy	0.74	0.79	0.77	1.00

Thank you!

Have a great rest of your Day!!!