澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

Diffusion Policy: Visuomotor Policy Learning via Action Diffusion

GONG Ying

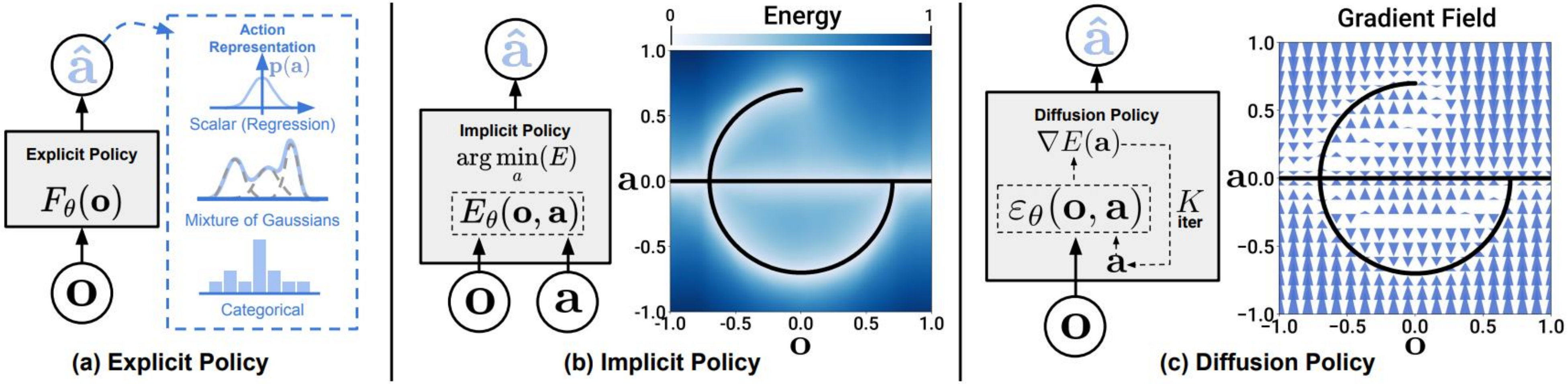
gong.ying@connect.umac.mo

2024.1.12

Contents

1. Introduction

- Policy learning from demonstration can be formulated as the supervised regression task of learning to map observations to actions.
- Prior work attempts to explore different action representations from explicit to implicit to better capture multi-modal distributions.
- This work introduces a new form of robot visuomotor policy that generates behavior via a "conditional denoising diffusion process on robot action space", Diffusion Policy.



1. Introduction

- **Diffusion policy** infers the action-score gradient, conditioned on visual observations, for K denoising iterations.
    I. Expressing multimodal action distributions.
    II. High-dimensional output space.
    III. Stable training.
- **Contributions:**
    I. Closed-loop action sequences.
    II. Visual conditioning.
    III. Time-series diffusion transformer.
- Consistent performance boost across all benchmarks with an average improvement of **46.9%**. (12 tasks from 4 benchmarks)

2. Diffusion Policy Formulation

Visuomotor robot policies are formulated as Denoising Diffusion Probabilistic Models (DDPMs).

- **Denoising Diffusion Probabilistic Models**

  Starting from x^K sampled from Gaussian noise, the DDPM performs K iterations of denoising to produce a series of actions with decreasing levels of noise, x^k, x^{k-1}... x^0, until a desired noise-free output x^0 is formed.

$$\boldsymbol{x}^{k-1} = \alpha(\boldsymbol{x}^k - \gamma\varepsilon_\theta(\boldsymbol{x}^k, k) + N(0, \sigma^2 I)),$$

  where ε_θ is the noise prediction network and $N(0, \sigma^2 I)$ is Gaussian noise added. It can be interpreted as a noisy gradient descent step:

$$\boldsymbol{x}' = \boldsymbol{x} - \gamma\nabla E(\boldsymbol{x}).$$

  Where $\varepsilon_\theta(\boldsymbol{x}, k)$ predicts the gradient field $\nabla E(\boldsymbol{x})$ and γ is the learning rate. An α slightly smaller than 1 improves stability.

2. Diffusion Policy Formulation

- **DDPM Training**

  The training process starts by randomly drawing unmodified examples, $\boldsymbol{x}^0$, from the dataset. For each sample, we randomly select a denoising iteration *k* and then sample a random noise ε^k with appropriate variance for iteration *k*. The noise prediction network is asked to predict the noise from the data sample with noise added.

$$L = \mathrm{MSE}\left(\varepsilon^k, \varepsilon_\theta\left(x^0 + \varepsilon^k, k\right)\right),$$

  Minimizing the loss function L also minimizes the variational lower bound of the KL-divergence between the data distribution $p(\boldsymbol{x}^0)$ and the distribution of samples drawn from the DDPM $q(\boldsymbol{x}^0)$.

2. Diffusion Policy Formulation

- **Diffusion for Visuomotor Policy Learning**

  DDPMs are typically used for image generation (x is an image), we use a DDPM to learn robot visuomotor policies. It requires 2 modifications:

  I. Change the output **x** to represent robot actions:

  Closed-loop action-sequence prediction

  II. Make the denoising processes conditioned on input observation $\boldsymbol{O}_t$:

  Visual observation conditioning

2. Diffusion Policy Formulation

- **Diffusion for Visuomotor Policy Learning**
  I. Closed-loop action-sequence prediction
  To encourage temporal consistency and smoothness in long-horizon planning while allowing prompt reactions to unexpected observations, the action-sequence prediction produced by a diffusion model is integrated with receding horizon control:
  At time step t the policy takes the latest T_O steps of observation data $\boldsymbol{O}_T$ as input and predicts T_P steps of actions, of which T_a steps of actions are executed on the robot without re-planning. Here we define T_O as the observation horizon, T_P as the action prediction horizon and T_a as the action execution horizon.

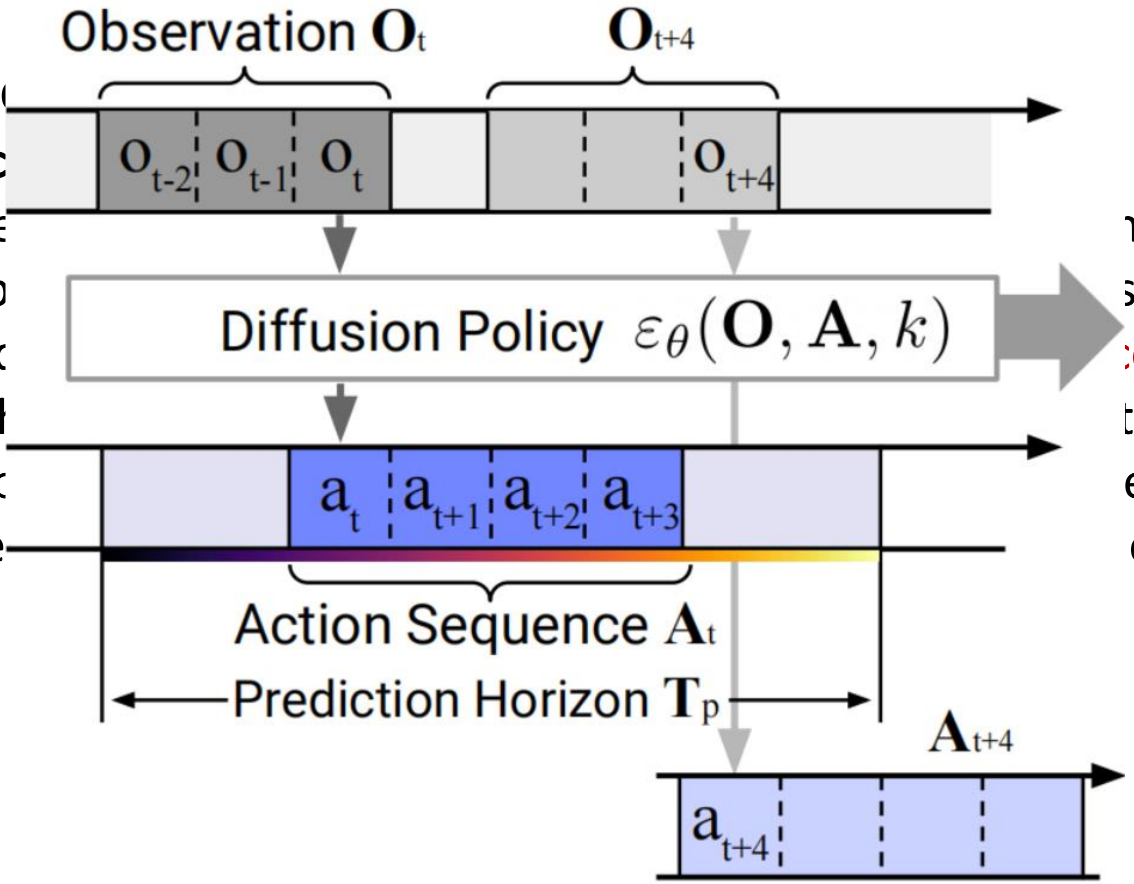
$$\boxed{o_1, \boxed{o_2, o_3}, o_4}, o_5, \ldots, \boxed{o_{t-2}, o_{t-1}, o_t}$$

$T_O = 3$

2. Diffusion Policy Formulation

- **Diffusion for Visu**
  - **I. Closed-loop ac**

    To encourage te … planning while
    allowing promp … sequence prediction
    produced by a c … control:
    At time step t th … ta $\boldsymbol{O}_T$ as input and
    predicts T_P step … ed on the robot without
    re-planning. He … e action prediction
    horizon and T_a



2. Diffusion Policy Formulation

- **Diffusion for Visuomotor Policy Learning**

  II. Visual observation conditioning

  We use a DDPM to approximate the conditional distribution $p(\boldsymbol{A}_t|\boldsymbol{O}_t)$ instead of the joint distribution $p(\boldsymbol{A}_t, \boldsymbol{O}_t)$. It allows the model to predict actions conditioned on observations without inferring future states, speeding up the diffusion process and improving the accuracy of generated actions. To capture $p(\boldsymbol{A}_t|\boldsymbol{O}_t)$, we modify $\boldsymbol{x}^{k-1} = \alpha(\boldsymbol{x}^k - \gamma\varepsilon_\theta(\boldsymbol{x}^k, k) + N(0, \sigma^2 I))$ to:

$$\boldsymbol{A}_t^{k-1} = \alpha(\boldsymbol{A}_t^k - \gamma\varepsilon_\theta(\boldsymbol{O}_t, \boldsymbol{A}_t^k, k) + N(0, \sigma^2 I)),$$

  And the training loss is modified from $L = \mathrm{MSE}\left(\varepsilon^k, \varepsilon_\theta(x^0 + \varepsilon^k, k)\right)$ to:

$$L = \mathrm{MSE}\left(\varepsilon^k, \varepsilon_\theta(\boldsymbol{O}_t, \boldsymbol{A}_t^0 + \varepsilon^k, k)\right),$$

  The exclusion of $\boldsymbol{O}_t$ from the output of the denoising process significantly improves inference speed and better accommodates real-time control.

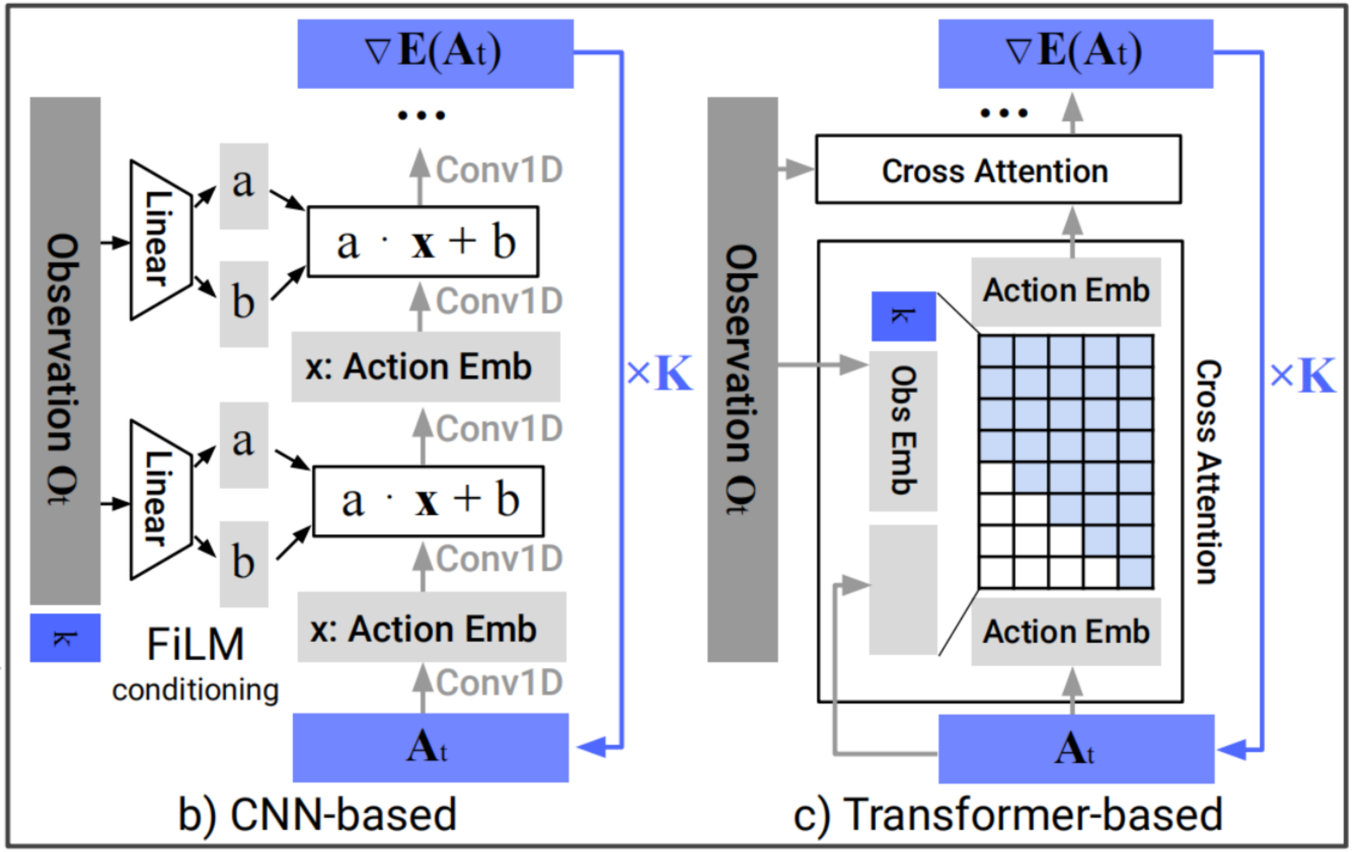
3. Key Design Decisions

- **Network Architecture Options**

Choice of NN architectures for ε_θ.

I. **CNN-based diffusion policy**
Needs less tuning, but performs poorly when the desired action sequence changes quickly and sharply.

II. **Time-series diffusion transformer**
More sensitive to hyperparameters, but performs well if the task is complex or action changes often.



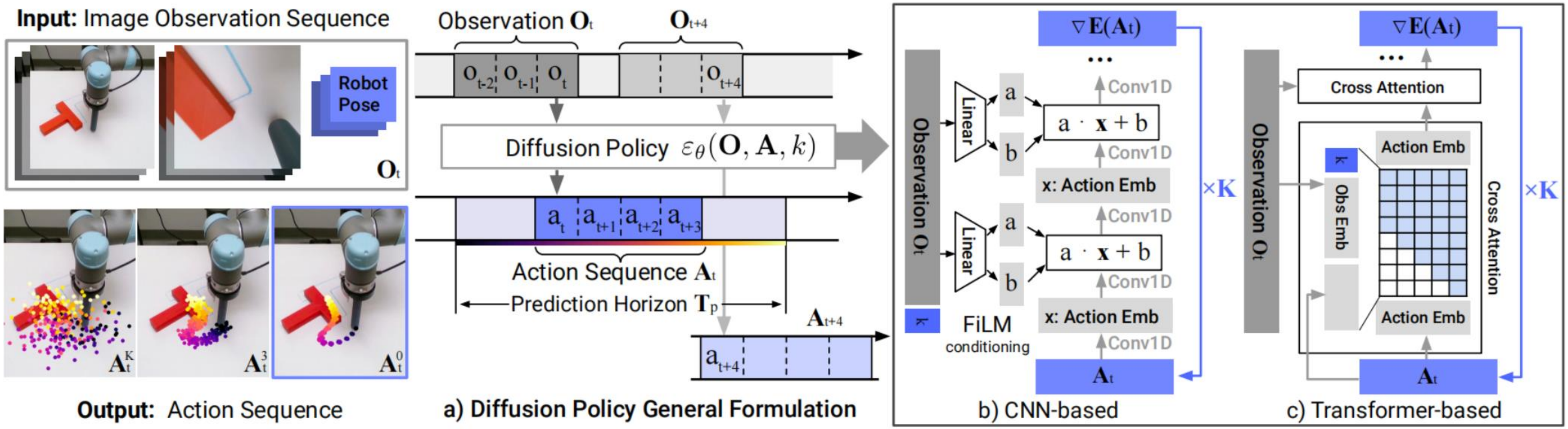
3. Key Design Decisions

- **Visual Encoder**
  - The visual encoder maps the raw image sequence into a latent embedding $\boldsymbol{O}_t$ and is trained end-to-end with the diffusion policy.
  - ResNet-18 is used as the encoder with the following modifications:
    I. Replace the global average pooling with a spatial softmax pooling to maintain spatial information.
    II. Replace BatchNorm with GroupNorm for stable training.
- **Noise Schedule**

  It's empirically found that the Square Cosine Schedule proposed in iDDPM works best for the tasks.
- **Accelerating Inference for Real-time Control**

  Denoising Diffusion Implicit Models (DDIM) approach.

4. Diffusion Policy Overview



Thank you.