📖 Diffusion Policy 论文精读笔记

> **论文全名**: *Diffusion Policy: Visuomotor Policy Learning via Action Diffusion*
> **作者**: Cheng Chi, Zhenjia Xu, Siyuan Feng 等（哥伦比亚大学 + MIT + Toyota Research）
> **发表**: RSS 2023 (arXiv: 2303.04137)
> **阅读日期**: Day 7
> **项目页**: https://diffusion-policy.cs.columbia.edu/

🎯 一句话总结

把图像生成领域大火的扩散模型（Diffusion Model）搬到机器人控制上：不再让策略直接输出"下一步怎么动"，而是从纯噪声出发，经过反复去噪，"画"出一整段未来动作序列。 这样做天然能处理"同一个场景有多种合理做法"的问题（多模态动作分布），训练还特别稳定。

与我课题的核心联系 🔗

> **模糊指令 → 多种合理动作 → Diffusion 的多模态建模能力正好适用！**
> 当工人说"把那个东西放过去"时，可能有3种合理的执行方式（拿不同的零件、放到不同的位置）。传统行为克隆（BC）只能输出一个"平均动作"（可能哪个都不对），而 Diffusion Policy 能表示这个多峰分布，采样时选择其中一个合理模式。你做消歧时，可以把 Diffusion Policy 当作"强执行器底座"——消歧模块负责缩小候选范围，Diffusion Policy 负责在确定目标后稳定执行。

📌 必读部分 vs 可跳过部分

部分	内容	重要性	建议
Figure 1	三种策略范式对比图	⭐⭐⭐	**必须看懂**，理解全文的起点
Figure 2	整体架构流程图	⭐⭐⭐	**必须看懂**，CNN-based 和 Transformer-based 两条路线
Section 2.1-2.3	DDPM 公式 + 如何改成策略	⭐⭐⭐	**必读核心**，先抓直觉再看公式
Section 3.1	网络架构选择（CNN vs Transformer）	⭐⭐⭐	必读，理解 FiLM 条件化和 Cross Attention
Section 3.2	视觉编码器设计	⭐⭐	了解 ResNet-18 + Spatial Softmax 即可
Section 4.1-4.3	Diffusion 的三大优势分析	⭐⭐⭐	**核心亮点**，多模态 + 位置控制 + 动作序列
Section 4.5	与控制理论的联系	⭐	可暂时跳过（LQR 推导细节）
Section 5	仿真实验结果	⭐⭐	看 Table 1-4 的数字感受性能差距
Section 6	真实机器人实验	⭐⭐⭐	**必读**，Push-T 和 Mug Flipping 的分析

部分	内容	重要性	建议
Appendix 数学推导	详细证明	⭐	**可跳过**，先抓直觉

🧠 核心概念拆解

1. 为什么需要 Diffusion Policy？——三种策略范式的对比

论文 Figure 1 是全文灵魂，它对比了三种让机器人"决定怎么动"的方式：

(a) 显式策略（Explicit Policy，显式策略）

直觉比喻：像一个考试只写一个答案的学生。

给模型一张图片（观测），它直接输出一个动作（比如"向右移动3cm"）。问题是：如果正确答案有多个（向左绕和向右绕都行），模型只能输出一个"折中"的平均值——可能刚好撞上障碍物。

常见做法：用高斯混合模型（GMM，Gaussian Mixture Model，多个高斯分布叠加在一起）来表示多个可能答案，但你需要提前猜"有几个答案"，而且混合太多就不稳定。

(b) 隐式策略（Implicit Policy，隐式策略）

直觉比喻：像一个给每个候选答案打分的评委。

模型不直接说"应该做什么"，而是学一个能量函数（Energy Function）$E_\theta(\mathbf{o}, \mathbf{a})$：给观测 $\mathbf{o}$ 和候选动作 $\mathbf{a}$，输出一个"这个动作有多不好"的分数。推理时找能量最低的动作。

问题：训练时需要"负样本"（故意造一些错误动作来对比），这个过程很不稳定。代表方法 IBC（Implicit Behavioral Cloning）训练时经常 loss 剧烈震荡。

(c) 扩散策略（Diffusion Policy）⭐

直觉比喻：像一个画家，从一张全是噪点的画开始，一笔一笔擦去噪点，最终呈现出一幅清晰的画。只不过这里"画"的不是图像，而是一段机器人动作序列。

核心思想：

1. **训练时**：拿到专家的真实动作序列，往上面加噪声，然后训练一个网络学会"预测加了什么噪声"
2. **推理时**：从纯随机噪声开始，反复调用这个网络去噪 K 次，最终得到一段干净的动作序列

为什么它天然能处理多模态？

想象一个山谷地形——有两个低谷（两种合理动作）。扩散过程就像从山顶随机滚一个小球下来：初始位置偏左就滚到左边的谷，偏右就滚到右边的谷。每次采样的随机噪声不同，小球就可能落入不同的谷。这就是多模态！

而传统回归（显式策略）则像是强行让小球停在两个谷的中间——那里其实是山脊，反而是最差的位置。

2. DDPM 核心公式解读（去噪扩散概率模型）

DDPM（Denoising Diffusion Probabilistic Model，去噪扩散概率模型）是扩散策略的数学基础。

去噪过程（推理时用）

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

逐项翻译：

- x^K：起点，纯高斯噪声（什么都不知道的"白纸"）
- x^0：终点，我们想要的干净动作序列
- $\epsilon_\theta(x^k, k)$：噪声预测网络，"猜"当前的 x^k 里混了多少噪声
- γ：去噪步长，类似梯度下降的学习率（learning rate）
- α：缩放系数
- $\mathcal{N}(0, \sigma^2 I)$：每步额外加一点小随机扰动，防止陷入局部最优
- $k = K, K-1, ..., 1, 0$：从第 K 步迭代到第 0 步

另一个直觉：整个过程等价于在能量场（Energy Landscape）上做梯度下降：

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

噪声预测网络 ϵ_θ 本质上在预测能量函数的梯度 $\nabla E(x)$，也就是"当前动作应该往哪个方向修正"。噪声调度（Noise Schedule，控制每步去噪力度的时间表）就像是梯度下降中的学习率调度。

训练过程

$$\mathcal{L} = \text{MSE}(\epsilon^k, \epsilon_\theta(x^0 + \epsilon^k, k))$$

用人话说：

1. 从数据集取一个真实动作序列 x^0
2. 随机选一个时间步 k，加上对应强度的噪声 ϵ^k
3. 把加噪后的 $x^0 + \epsilon^k$ 和时间步 k 喂给网络
4. 网络输出它猜测的噪声 ϵ_θ
5. 损失函数就是"猜的噪声"和"真实噪声"之间的均方误差（MSE）

这个训练过程极其简单稳定——就是一个回归任务！不需要 GAN 的对抗训练，不需要 IBC 的负样本。这是 Diffusion Policy 训练稳定的根本原因。

3. 从图像生成到机器人控制——两个关键改造

DDPM 原本用于生成图片，要用于机器人控制，需要两个改造：

改造 1：输出变成动作序列

原来 x 是一张图片，现在 $x = A_t = [a_t, a_{t+1}, ..., a_{t+T_p-1}]$ 是一段 **未来动作序列**。

不是只预测"下一步做什么"，而是一次性预测未来 T_p 步的动作。这就像下棋时不只想下一步，而是想好接下来的整个战术。

为什么要预测一整段？ 三大好处：

1. **时序一致性（Temporal Action Consistency）**：避免每步独立采样时的动作抖动。想象你在画一条线，如果每个点独立画，可能歪歪扭扭；但如果一笔画一段，自然流畅
2. **对闲置动作的鲁棒性（Robustness to Idle Actions）**：人类遥操作时经常有停顿，如果模型逐步学，容易过拟合于"不动"这个伪模式
3. **天然适配扩散模型的高维生成能力**：扩散模型本来就擅长处理高维输出

改造 2：以视觉观测作为条件（Visual Conditioning）

从 $p(A_t)$ 变成 $p(A_t|O_t)$，即 **条件生成**：给定当前看到的图像，生成相应的动作序列。

关键设计：视觉特征 O_t 只提取一次，然后在所有 K 步去噪迭代中重复使用。这大大节省了计算量！（如果每次去噪都重新跑一遍图像编码器，速度会慢 K 倍。）

---

4. 滚动时域控制（Receding Horizon Control）⭐

这是让 Diffusion Policy 在真实机器人上跑起来的关键工程设计。

直觉比喻：就像开车时你会看远处规划路线（预测 T_p 步），但只踩当前这一脚油门（执行 T_a 步），然后根据最新路况重新规划。

具体来说，论文定义了三个时间参数：

- T_o（Observation Horizon，观测时域）：用最近 T_o 帧图像作为输入
- T_p（Prediction Horizon，预测时域）：一次预测未来 T_p 步的动作
- T_a（Action Horizon，动作时域）：只执行前 T_a 步，然后丢弃剩余的，重新规划

为什么不把预测的全部执行完？

因为执行过程中环境会变化（比如物体被碰歪了）。只执行一小段就重新看一眼，重新规划，就能兼顾：

- **长远规划**：预测 T_p 步保证动作有前瞻性和连贯性
- **实时响应**：每隔 T_a 步就可以纠偏

论文实验发现 $T_a = 8$ **步左右是最优的**：太短（$T_a = 1$）相当于逐步决策，回到了老问题；太长（$T_a = 128$）则反应迟钝。

哲学思考 💡：这体现了"计划与执行的平衡"——过度计划会僵化，过度随机会混乱。好的策略是"想远一点，但只走一小步，然后根据新信息重新想"。这不仅是机器人控制的智慧，也是生活的智慧。

---

5. 两种网络架构

论文探索了两种噪声预测网络 ϵ_θ 的实现方式：

(a) CNN-based（1D 时序卷积网络）

- 使用 1D 卷积处理动作序列的时间维度
- 通过 **FiLM（Feature-wise Linear Modulation，特征逐通道线性调制）** 注入视觉条件
- FiLM 本质上就是：$\text{output} = a \cdot x + b$，其中 a, b 由视觉特征和时间步编码生成
- 同时注入扩散时间步 k 和观测特征 O_t

优点：对平滑、连续的动作效果好（比如推物体）
缺点：卷积的平滑特性会导致高频动作变化被"磨平"（over-smoothing，过度平滑）

(b) Transformer-based（时序扩散 Transformer）

- 把动作序列的每个时间步做成 token
- 用 **Cross Attention（交叉注意力）** 将视觉特征注入每一层 Transformer
- 使用 **Causal Attention Mask（因果注意力掩码）**：每个动作 token 只能看到它自己和之前的 token，不能"偷看"未来

优点：能更好地捕捉高频动作变化和速度控制
缺点：计算量略大

选择建议：大多数操作任务用 CNN 版本就够了；如果你的工业场景需要精细的速度控制或快速方向变化，试试 Transformer 版本。

---

6. 视觉编码器（Visual Encoder）

使用 **ResNet-18**（从头训练，不用预训练权重），但做了两个关键修改：

1. **Spatial Softmax Pooling** 替代全局平均池化：保留空间位置信息。全局平均池化会把"哪里有东西"的信息丢掉，Spatial Softmax 则告诉你"关键点在图像的哪个位置"——对机器人操作任务至关重要
2. **GroupNorm** 替代 BatchNorm：当用 EMA（Exponential Moving Average，指数移动平均，一种训练稳定化技巧）时，BatchNorm 的统计量会和 EMA 冲突，GroupNorm 没这个问题

论文发现端到端训练（从头一起训练视觉编码器和策略网络）效果最好，比用预训练的 R3M 或 ImageNet 特征都好。

---

7. 推理加速：DDIM

原始 DDPM 需要 K=100 步去噪，太慢了！论文使用 **DDIM（Denoising Diffusion Implicit Models，去噪扩散隐式模型）** 的技巧：

- 训练时用 100 步
- 推理时只用 **10 步**
- 在 Nvidia 3080 GPU 上只需 **0.1 秒推理延迟**，满足实时控制要求（10Hz 控制频率）

📊 关键实验结果

整体性能

对比方法	说明	Diffusion Policy 优势
LSTM-GMM	LSTM + 高斯混合模型	平均提升 46.9%
IBC	隐式行为克隆	训练稳定性远超
BET	行为 Transformer	更好的多模态表达
BC-RNN	带 RNN 的行为克隆	避免动作抖动

关键发现

1. 多模态处理能力遥遥领先

在 Block Push（推方块）任务中，需要按任意顺序推两个方块。Diffusion Policy 比第二名提升 **32%**。在 Kitchen（厨房）任务中更是提升 **213%**。

2. 位置控制 > 速度控制（对 Diffusion Policy 来说）

这是一个反直觉的发现！大多数之前的工作用速度控制（velocity control），但 Diffusion Policy 搭配位置控制（position control）效果更好。

原因：

- 位置控制下动作的多模态性更明显（向左5cm vs 向右5cm），Diffusion 能更好地建模
- 速度控制有累积误差（compounding error，误差像滚雪球一样越滚越大），位置控制则不会

3. 对延迟鲁棒

即使有 4 步的系统延迟（从看到图像到执行动作之间的时间差），Diffusion Policy 的位置控制版本依然表现稳定。这对工业场景很重要——工厂网络通信总有延迟。

4. 真实机器人表现

- **Push-T 任务**（推 T 形方块到目标位置）：95% 成功率，接近人类水平。对比之下，IBC 0%，LSTM-GMM 20%
- **Mug Flipping**（翻转杯子）：90% 成功率，展现出从未被示教过的"连续推→重新抓取"的创新行为
- **鲁棒性测试**：遮挡相机3秒、人为干扰物体位置 → Diffusion Policy 都能立即重新规划并恢复

🔑 核心术语表

英文术语	中文翻译	通俗解释
Diffusion Model	扩散模型	从噪声逐步去噪生成目标的生成式模型
DDPM	去噪扩散概率模型	扩散模型的具体数学框架
DDIM	去噪扩散隐式模型	DDPM 的加速版，推理步数可大幅减少
Visuomotor Policy	视觉运动策略	看图片 → 输出动作的策略
Behavior Cloning (BC)	行为克隆	直接模仿专家动作的监督学习方法
Multi-modal Distribution	多模态分布	同一输入对应多个合理输出的概率分布
Receding Horizon	滚动时域	预测一长段但只执行一小段，然后重新规划
FiLM Conditioning	特征调制条件化	用条件信息（图像/时间步）缩放和偏移特征
Noise Schedule	噪声调度	控制每步去噪力度的超参数时间表
Score Function	评分函数	概率分布对数密度的梯度，指向"高概率方向"
Stochastic Langevin Dynamics	随机朗之万动力学	用梯度+噪声从分布中采样的方法
Energy-Based Model (EBM)	基于能量的模型	通过最小化能量函数来找最优动作
Spatial Softmax	空间 Softmax	池化时保留关键点空间位置的方法
Position Control	位置控制	直接告诉机器人"去哪个坐标"
Velocity Control	速度控制	告诉机器人"以什么速度移动"
Action Horizon (T_a)	动作时域	预测的动作序列中实际执行的步数
Prediction Horizon (T_p)	预测时域	一次去噪预测的总动作步数

英文术语	中文翻译	通俗解释
Observation Horizon (T_o)	观测时域	输入的历史图像帧数

🏗️ 与课题方向的深度连接

场景：工业机械臂 + 模糊指令

假设工人说："**把那个螺丝拧到那里**"

```
指令: "把那个螺丝拧到那里"
    |
    ├──→ 动作模式1: 拿红色螺丝放左边 (概率 40%)
    ├──→ 动作模式2: 拿蓝色螺丝放右边 (概率 35%)
    └──→ 动作模式3: 拿扳手放中间  (概率 25%)

传统 BC: 输出一个"平均"动作（可能哪个都不对）
Diffusion: 能表示这个多峰分布，采样时选择其中一个完整模式
```

你的系统架构设想

```
模糊指令 "把那个东西放过去"
      |
      ▼
┌──────────────────────────┐
| 消歧模块（你的核心贡献） | ← Grounding DINO 找候选 + Ask-to-Clarify 交互
| "你指的是左边的螺丝吗？" |
└──────────────────────────┘
      | 消歧后明确指令
      ▼
┌──────────────────────────┐
| Diffusion Policy         | ← 接收明确目标，生成稳定的动作序列
| （强执行器底座）         |   多模态能力处理"执行路径"的多样性
└──────────────────────────┘
```

关键启示

1. **Diffusion 处理的多模态和你处理的歧义不完全相同**：Diffusion 处理的是"执行层面的多模态"（同一个目标有多条路径），你的课题处理的是"语义层面的歧义"（指令本身不明确）。两者互补！
2. **消歧模块可以利用 Diffusion 的不确定性**：如果 Diffusion Policy 在当前观测下生成的动作分布特别"分散"（方差大），说明当前情况确实模糊，可以作为触发澄清提问的信号。
3. **位置控制的启示**：工业场景中精确的位置控制非常重要（"螺丝孔在 x=15.3mm"），Diffusion Policy 恰好和位置控制搭配最好。

💡 思考题（用于自查理解）

1. **为什么 Diffusion Policy 比 IBC（隐式行为克隆）训练更稳定？**
   > 提示：IBC 需要负样本估计归一化常数；Diffusion 绕开了这个问题，只需要预测噪声。
2. **如果把动作时域 T_a 设为 1，Diffusion Policy 退化成了什么？**
   > 提示：每步都重新规划，失去了时序一致性优势，但保留了多模态建模能力。
3. **为什么视觉特征只提取一次而不是每次去噪都重新提取？**
   > 提示：观测在一次推理中不会变化；重复提取 K 次是浪费计算。
4. **在你的工业场景中，什么情况下 Diffusion Policy 的多模态特性是有益的，什么情况下需要先消歧？**
   > 提示：路径多样性（绕左边还是右边抓）→ 有益；目标歧义（拿哪个螺丝）→ 需要先消歧。

📚 延伸阅读建议

论文	关系	阅读优先级
DDPM (Ho et al., 2020)	Diffusion Policy 的数学基础	如果你想深挖扩散数学，再读
DDIM (Song et al., 2021)	推理加速的核心技术	了解即可
Diffuser (Janner et al., 2022)	扩散用于轨迹规划（非策略）	对比理解"规划 vs 策略"
IBC (Florence et al., 2021)	隐式策略的代表，本文主要对比对象	推荐略读
π0 (2024)	基于 Diffusion 的最新 VLA	你的下一篇精读

⚙️ 局限性与未来方向

1. **继承了行为克隆的缺点**：示教数据质量差或不够多时效果会下降
2. **推理延迟高于简单方法**：LSTM-GMM 只需一次前向传播，Diffusion 需要 10 次去噪迭代。虽然用 DDIM 加速到了 0.1 秒，但对需要 1000Hz 控制的精密任务可能还不够
3. **缺少语言条件**：原版 Diffusion Policy 不接受语言指令输入——这正是你的课题可以扩展的方向！后续 π0 等工作已经在解决这个问题

笔记生成时间: Day 7 学习阶段
下一步: Day 7 下午 + Day 8 上午继续精读论文细节，Day 8 下午跑通 Diffusion Policy 仿真 demo