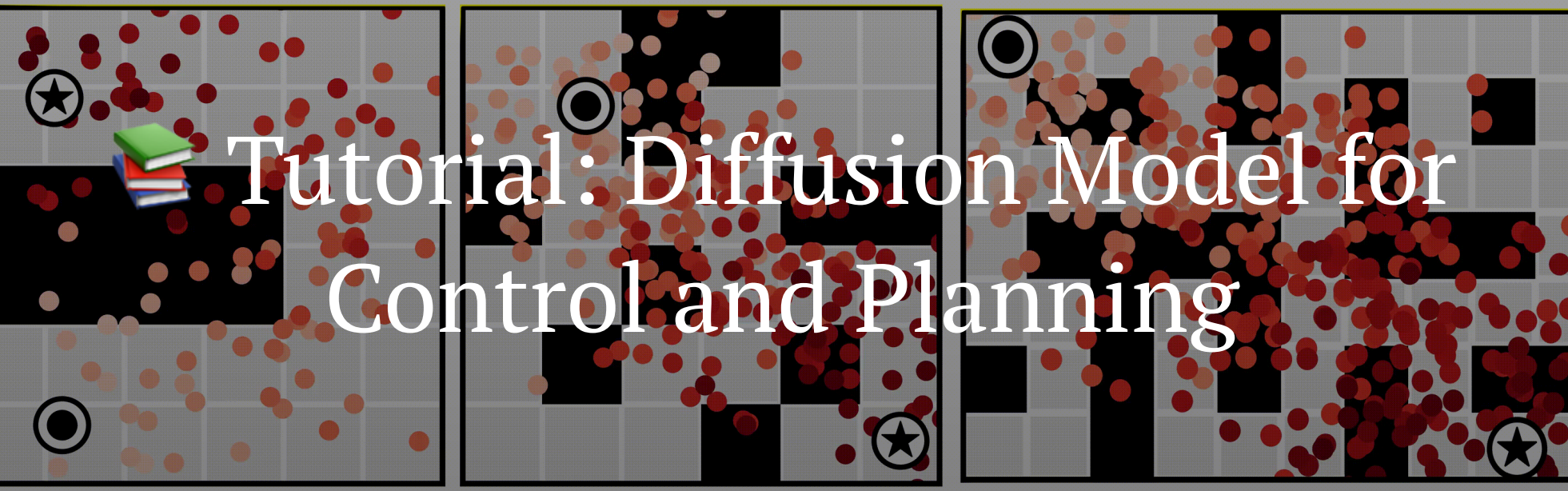


Tutorial: Diffusion Model for Control and Planning

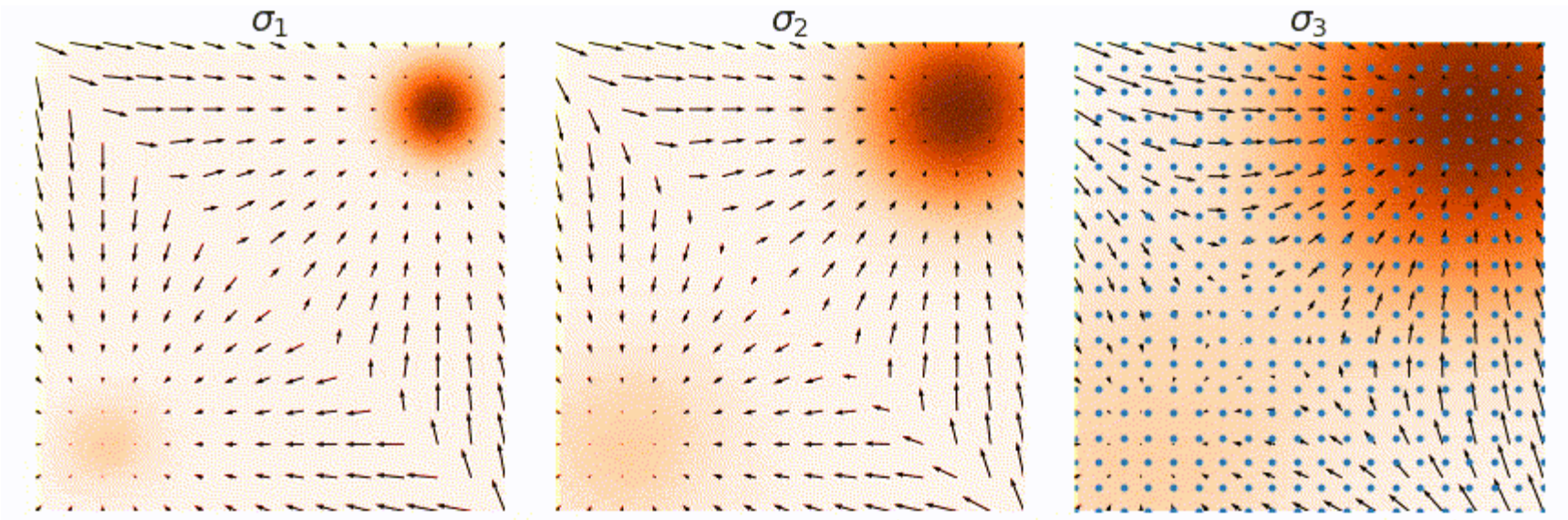
🔄 Recap: What is a Diffusion Model?

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.

🔄 Recap: Core of Diffusion Model

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.
- Core: Score function for sample generation and distribution description.

$$\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + c\nabla \log p\left(\boldsymbol{x}_i\right) + \sqrt{2c}\boldsymbol{\epsilon}, \quad i = 0, 1, \ldots, K$$



🔄 Recap: Advantages of Diffusion Model

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.
- Core: Score function for sample generation and distribution description.
- Advantages:

🔄 Recap: Advantages of Diffusion Model

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.
- Core: Score function for sample generation and distribution description.
- Advantages:
  - 🌟 Multimodal: Effective with multimodal distributions.

🔄 Recap: Advantages of Diffusion Model

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.
- Core: Score function for sample generation and distribution description.
- Advantages:
  - 🌟 Multimodal: Effective with multimodal distributions.
  - 📈 Scalable: Suits high-dimensional problems.

🔄 Recap: Advantages of Diffusion Model

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.
- Core: Score function for sample generation and distribution description.
- Advantages:
  - 🌟 Multimodal: Effective with multimodal distributions.
  - 📈 Scalable: Suits high-dimensional problems.
  - 🔒 Stable: Grounded in solid mathematics and training.

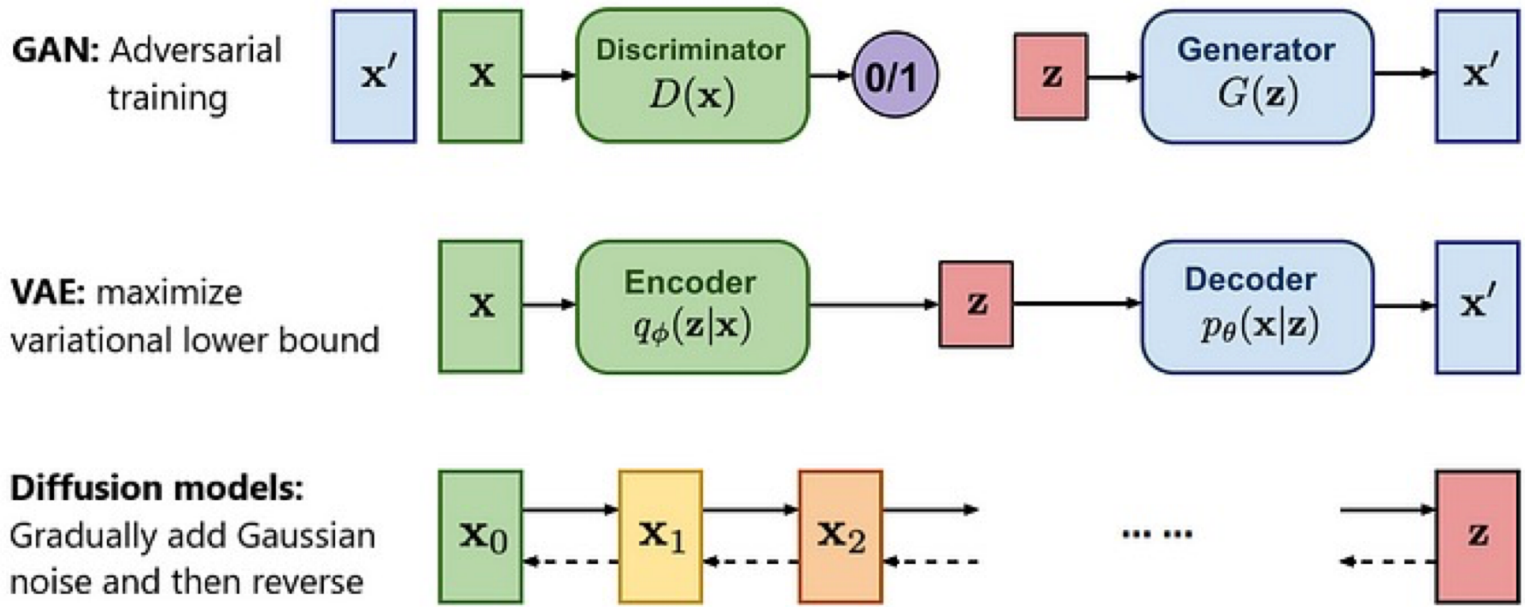
🔄 Recap: Advantages of Diffusion Model

- Keynote: Generative model for distribution matching.
- Applications: Image and text generation, creative tasks.
- Core: Score function for sample generation and distribution description.
- Advantages:
  - 🌟 Multimodal: Effective with multimodal distributions.
  - 📈 Scalable: Suits high-dimensional problems.
  - 🔒 Stable: Grounded in solid mathematics and training.
  - 🔄 Non-autoregressive: Predicts entire trajectories efficiently.

🚀 Motivation: Why a Generative Model in Control and Planning?

Generative Models in Control and Planning

- Generative Models: application in imitation learning to match expert data.
- Examples: GANs, VAEs in imitation learning.



🚀 Motivation: Why a Generative Model in Control and Planning?

Generative Models in Control and Planning

- Generative Models: application in imitation learning to match expert data.
- Examples: GANs, VAEs in imitation learning.
- GAN in GAIL: Discriminator learning and policy training.
  - Idea: Train a discriminator to distinguish between expert and agent data.
  - Limitation: Struggles with multimodal distributions, unstable training.

Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \tag{17}$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

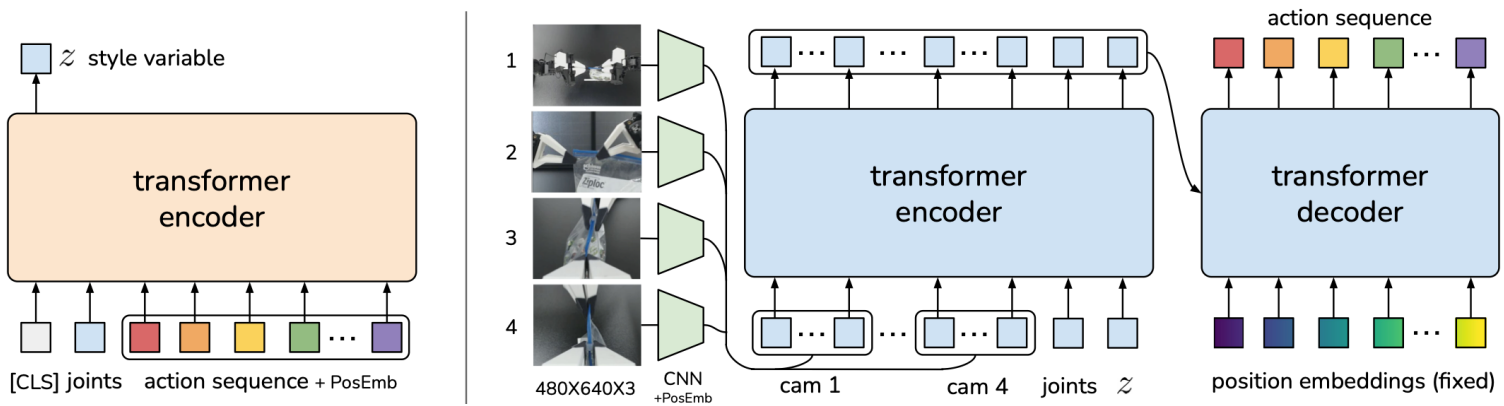
$$\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta), \quad \text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \tag{18}$$

6: **end for**

🚀 Motivation: Why a Generative Model in Control and Planning?

Generative Models in Control and Planning

- Generative Models: Crucial in control and planning.
- Examples: GANs, VAEs in imitation learning.
- VAE in ACT (ALOHA): Latent space learning for planning.
  - Idea: learn a latent space for planning and control. (generate action in chunks)
  - Limitation: hard to train.



Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware

Tony Zhao Vikash Kumar Sergey Levine Chelsea Finn

Stanford University UC Berkeley Meta

RSS 2023 Paper

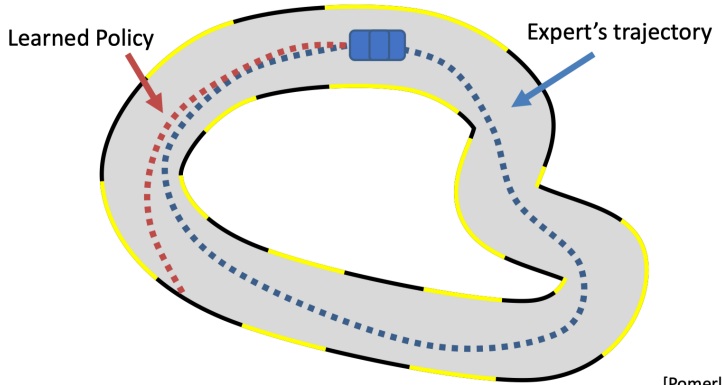
Abstract. Fine manipulation tasks, such as threading cable ties or slotting a battery, are notoriously difficult for robots because they require precision, careful coordination of contact forces, and closed-loop visual feedback. Performing these tasks typically requires high-end robots, accurate sensors, or careful calibration, which can be expensive and difficult to set up. *Can learning enable low-cost and imprecise hardware to perform these fine*

🚀 Motivation: Why a Generative Model in Control and Planning?

What to Learn with the Generative Model?

Scenario: Imitation Learning $\min \|p_\theta(\tau) - p_{\text{data}}(\tau)\|$

- Challenge: Match high-dimensional, multimodal trajectory distributions.
- Solution: Diffusion models for expressive distribution matching.
- Common Method: GAIL with adversarial training.
- Limitation: Struggles with multimodal distributions, unstable training.



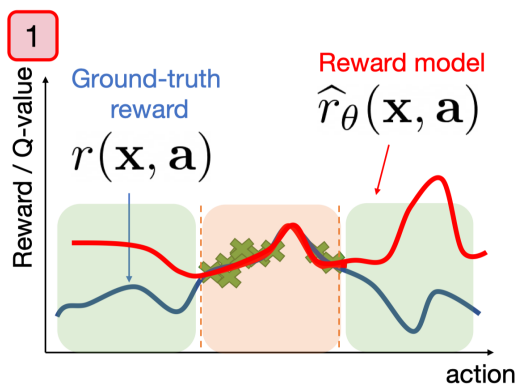
[Pomerleau89 Daume09]

🚀 Motivation: Why a Generative Model in Control and Planning?

What to Learn with the Generative Model?

Scenario: Offline Reinforcement Learning $\max J_\theta(\tau) \geq J_{\text{data}}(\tau) \text{s.t.} \|p_\theta(\tau) - p_{\text{data}}(\tau)\| < \epsilon$

- Challenge: Outperform demonstrations, ensure close action distribution.
- Solution: Diffusion models to match action distribution effectively.
- Common Method: CQL, penalizes out-of-distribution samples.
- Limitation: over-conservative.

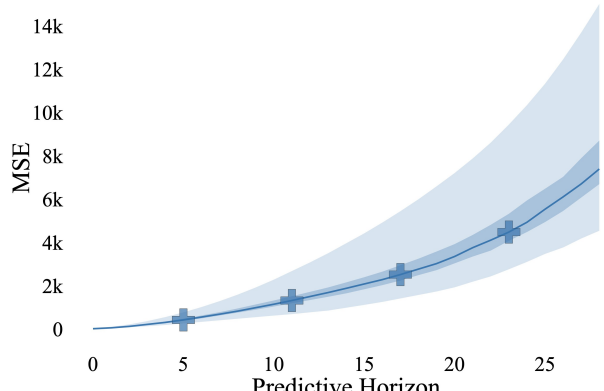


🚀 Motivation: Why a Generative Model in Control and Planning?

What to Learn with the Generative Model?

Scenario: Model-based Reinforcement Learning

- Challenge: Match dynamic model and policy's action distribution.
- Solution: Diffusion models for non-autoregressive, multimodal matching.
- Common method: planning with learned dynamics.
- Limitation: compounding error in long-horizon planning.



🚀 Motivation: Why a Generative Model in Control and Planning?

What to Learn with the Generative Model?

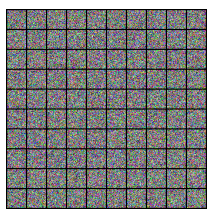
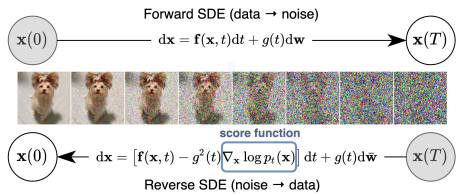
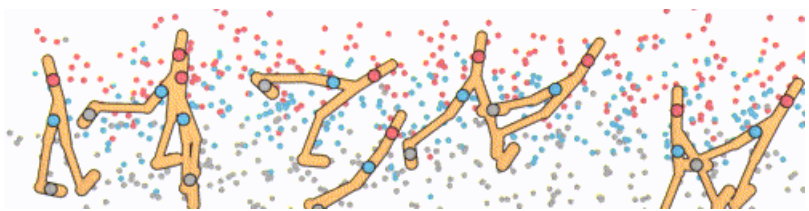
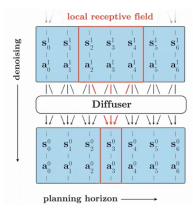
Key: using a powerful model to matching a high-dimensional, multimodal distribution.

- Action/Value distribution matching: grounded in demostrations -> offline RL.
- Trajectory distribution matching: dynamic feasibility and optimal trajectory distribution -> model-based RL.
- Transition distribution matching: dynamics matching in a non-autoregressive manner -> model-based RL.

🛠️ Practice: How to Use the Diffuser?

What to Diffuse?

- Most common: diffuse trajectory (`diffuser`).
- Diffused variable `x` : state, action sequence. $\tau = \{s_0, a_0, s_1, a_1, \ldots, s_T, a_T\}$.

Task	Thing's to Diffuse	How to Diffuse
Image Generation		
Planning		

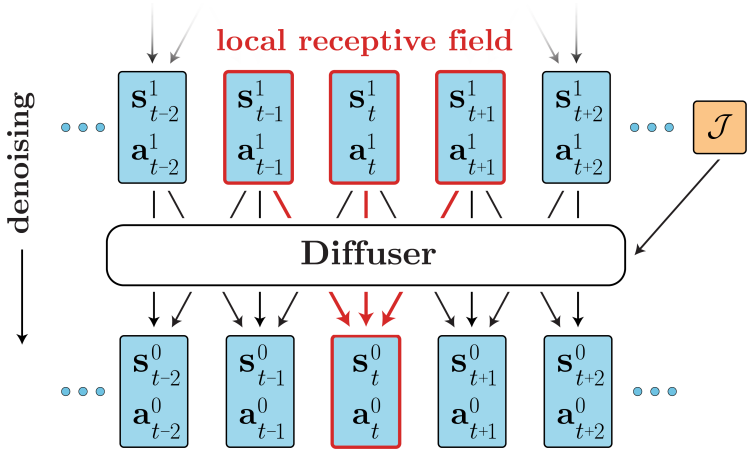
Planning with Diffusion for Flexible Behavior Synthesis

Michael Janner*, Yilun Du*, Joshua Tenenbaum, and Sergey Levine

ICML 2022 (long talk) Paper Code Colab BibTex

*equal contribution

Planning as denoising

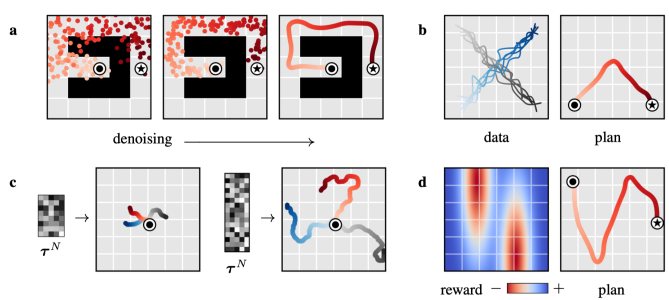


Diffuser is a denoising diffusion probabilistic model that plans by iteratively refining randomly sampled noise. The denoising process lends itself to flexible conditioning, by either using gradients of an objective function to bias plans toward high-reward regions or conditioning the plan to reach a specified goal.

🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

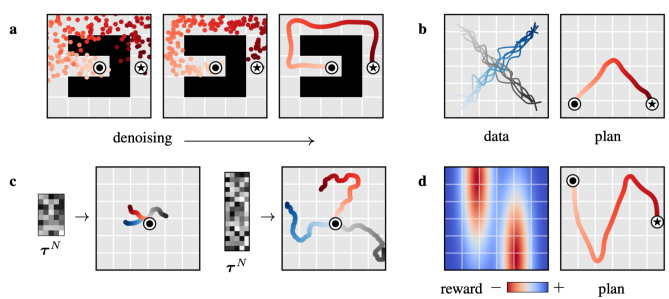
- Objective: make the trained model can generalize to new constraints and tasks.
- Common case: goal-conditioned, safety, new task etc.
- Possible Methods:



🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

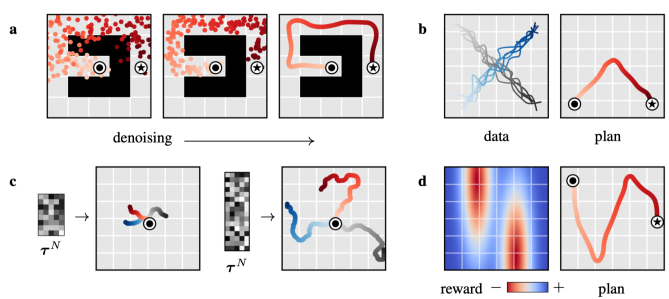
- Objective: make the trained model can generalize to new constraints and tasks.
- Common case: goal-conditioned, safety, new task etc.
- Possible Methods:
  - Guidance function (d): shift distribution with extra gradient.



🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

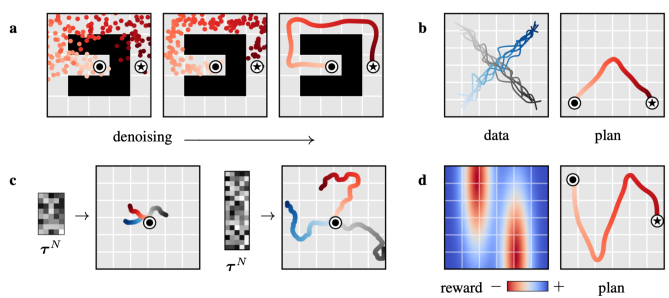
- Objective: make the trained model can generalize to new constraints and tasks.
- Common case: goal-conditioned, safety, new task etc.
- Possible Methods:
  - Guidance function (d): shift distribution with extra gradient.
  - Classifier-free method: learn a model can both represent conditional and unconditional distribution.



🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Objective: make the trained model can generalize to new constraints and tasks.
- Common case: goal-conditioned, safety, new task etc.
- Possible Methods:
  - Guidance function (d): shift distribution with extra gradient.
  - Classifier-free method: learn a model can both represent conditional and unconditional distribution.
  - Inpainting (a): fill in the missing part of the trajectory by fixing certain start and end state.

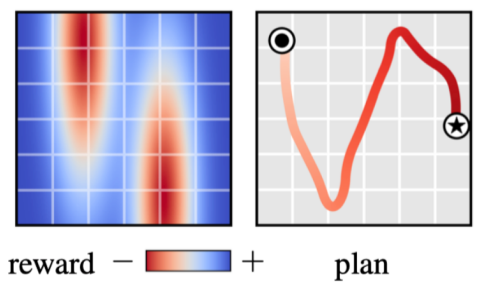


🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Guidance function: shift distribution with extra gradient.
  - Predefined the guidance function:
    - Method: shift distribution with a manually defined function
    - Limitation: Might lead to OOD samples, which break the learned diffusion process.

$$\tilde{p}_\theta(\boldsymbol{\tau}) \propto p_\theta(\boldsymbol{\tau}) h(\boldsymbol{\tau})$$
$$\boldsymbol{\tau}^{i-1} = \mathcal{N}\left(\mu + \alpha \Sigma \nabla \mathcal{J}(\mu), \Sigma^i\right)$$



🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Guidance function: shift distribution with extra gradient.
  - Predefined the guidance function:
    - Method: shift distribution with a manually defined function
    - Limitation: Might lead to OOD samples, which break the learned diffusion process.
  - Learned classifier:
    - Method: learning a classifier to distinguish between different constraints. (similar to GAN)
    - Limitation: Hard to tune parameters.

$$
\begin{aligned}
\nabla \log p\left(\boldsymbol{x}_t \mid y\right) &= \nabla \log \left(\frac{p\left(\boldsymbol{x}_t\right) p\left(y \mid \boldsymbol{x}_t\right)}{p(y)} \right) \\
&= \nabla \log p\left(\boldsymbol{x}_t\right) + \nabla \log p\left(y \mid \boldsymbol{x}_t\right) - \nabla \log p(y) \\
&= \underbrace{\nabla \log p\left(\boldsymbol{x}_t\right)}_{\text{unconditional score}} + \underbrace{\nabla \log p\left(y \mid \boldsymbol{x}_t\right)}_{\text{adversarial gradient}}
\end{aligned}
$$

🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Guidance function: shift distribution with extra gradient. ▶ leads to OOD samples
  - Predefined the guidance function:
    - Method: shift distribution with a manually defined function
    - Limitation: Might lead to OOD samples, which break the learned diffusion process.
  - Learned classifier:
    - Method: learning a classifier to distinguish between different constraints. (similar to GAN)
    - Limitation: Hard to tune parameters.

$$
\begin{aligned}
\nabla \log p\left(\boldsymbol{x}_t \mid y\right) &= \nabla \log \left(\frac{p\left(\boldsymbol{x}_t\right) p\left(y \mid \boldsymbol{x}_t\right)}{p(y)} \right) \\
&= \nabla \log p\left(\boldsymbol{x}_t\right) + \nabla \log p\left(y \mid \boldsymbol{x}_t\right) - \nabla \log p(y) \\
&= \underbrace{\nabla \log p\left(\boldsymbol{x}_t\right)}_{\text{unconditional score}} + \underbrace{\nabla \log p\left(y \mid \boldsymbol{x}_t\right)}_{\text{adversarial gradient}}
\end{aligned}
$$

🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Guidance function: shift distribution with extra gradient. ▶ leads to OOD samples
- Classifier-Free Method: learn a model can both represent conditional and unconditional distribution.
  - Method: drop out the condition term to learn a model can represent both conditional and unconditional distribution.

$$
\begin{aligned}
\nabla \log p\left(\boldsymbol{x}_t \mid y\right) &= \nabla \log p\left(\boldsymbol{x}_t\right) + \gamma\left(\nabla \log p\left(\boldsymbol{x}_t \mid y\right) - \nabla \log p\left(\boldsymbol{x}_t\right)\right) \\
&= \nabla \log p\left(\boldsymbol{x}_t\right) + \gamma \nabla \log p\left(\boldsymbol{x}_t \mid y\right) - \gamma \nabla \log p\left(\boldsymbol{x}_t\right) \\
&= \underbrace{\gamma \nabla \log p\left(\boldsymbol{x}_t \mid y\right)}_{\text{conditional score}} + \underbrace{(1-\gamma) \nabla \log p\left(\boldsymbol{x}_t\right)}_{\text{unconditional score}}
\end{aligned}
$$

🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Guidance function: shift distribution with extra gradient. ▶ leads to OOD samples
- Classifier-Free Method: learn a model can both represent conditional and unconditional distribution.

Guidance Function Method

Algorithm 1 Guided Diffusion Planning

Require Diffuser μ_θ, guide $\mathcal{J}$, scale α, covariances Σ^i
while not done **do**
  Observe state $\mathbf{s}$; initialize plan $\boldsymbol{\tau}^N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  for $i = N, \ldots, 1$ **do**
    `// parameters of reverse transition`
    $\mu \leftarrow \mu_\theta(\boldsymbol{\tau}^i)$
    `// guide using gradients of return`
    $\boldsymbol{\tau}^{i-1} \sim \mathcal{N}(\mu + \alpha\Sigma\nabla\mathcal{J}(\mu), \Sigma^i)$
    `// constrain first state of plan`
    $\boldsymbol{\tau}^{i-1}_{\mathbf{s}_0} \leftarrow \boldsymbol{s}$
  Execute first action of plan $\boldsymbol{\tau}^0_{\mathbf{a}_0}$

Classifier-Free Method

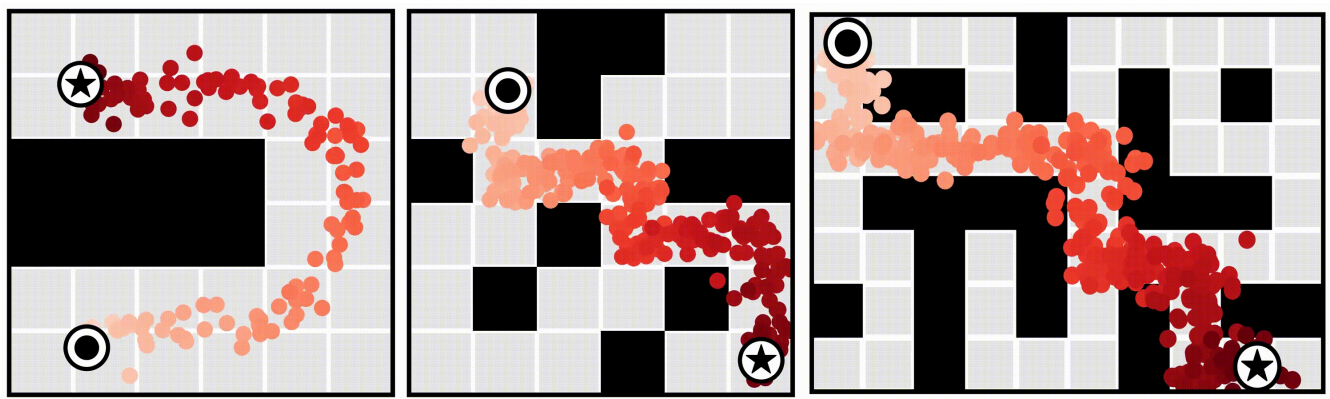
Algorithm 1 Conditional Planning with the Decision Diffuser

Input: Noise model ϵ_θ, inverse dynamics f_ϕ, guidance scale ω, history length C, condition $\boldsymbol{y}$
Initialize $h \leftarrow \texttt{Queue}(\texttt{length} = C)$, $t \leftarrow 0$ // Maintain a history of length C
while not done **do**
  Observe state s; $h.\texttt{insert}(s)$; Initialize $\boldsymbol{x}_K(\tau) \sim \mathcal{N}(0, \alpha I)$
  for $k = K \ldots 1$ **do**
    $\boldsymbol{x}_k(\tau)[:\texttt{length}(h)] \leftarrow h$ // Constrain plan to be consistent with history
    $\hat{\epsilon} \leftarrow \epsilon_\theta(\boldsymbol{x}_k(\tau), k) + \omega(\epsilon_\theta(\boldsymbol{x}_k(\tau), \boldsymbol{y}, k) - \epsilon_\theta(\boldsymbol{x}_k(\tau), k))$ // Classifier-free guidance
    $(\mu_{k-1}, \Sigma_{k-1}) \leftarrow \texttt{Denoise}(\boldsymbol{x}_k(\tau), \hat{\epsilon})$
    $\boldsymbol{x}_{k-1} \sim \mathcal{N}(\mu_{k-1}, \alpha\Sigma_{k-1})$
  end for
  Extract (s_t, s_{t+1}) from $x_0(\tau)$
  Execute $a_t = f_\phi(s_t, s_{t+1})$; $t \leftarrow t + 1$
end while

🛠️ Practice: How to Use the Diffuser?

How to Impose Constraints/Objectives?

- Guidance function: shift distribution with extra gradient. ▶ leads to OOD samples
- Classifier-Free Method: learn a model can both represent conditional and unconditional distribution.
- Inpainting: fill in the missing part of the trajectory by fixing certain start and end state.
  - Method: fix the start and end state, and fill in the missing part of the trajectory.



🛠️ Practice: How to Use the Diffuser?

- Common thing to diffuse: trajectory.
- Common way to impose constraints/add objectives: guidance function, classifier-free method, inpainting.

📚 Literatures: Recent Research Progress in Diffusion for RL/Control

A detailed summary of each method can be found here.

The key of diffusion: how to get the score function.

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \quad (\quad \underbrace{x}_{\text{what to diffuse}} \quad | \quad \underbrace{y}_{\text{how to impose constraints/objectives}} \quad)$$

📚 Literatures: Recent Research Progress in Diffusion for RL/Control

A detailed summary of each method can be found here.

The key of diffusion: how to get the score function.

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \quad (\quad \underbrace{x}_{\text{what to diffuse}} \quad | \quad \underbrace{y}_{\text{how to impose constraints/objectives}} \quad)$$

- How to get score function: data-driven v.s. analytical.

📚 Literatures: Recent Research Progress in Diffusion for RL/Control

A detailed summary of each method can be found here.

The key of diffusion: how to get the score function.

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \quad (\quad \underbrace{x}_{\text{what to diffuse}} \quad | \quad \underbrace{y}_{\text{how to impose constraints/objectives}} \quad)$$

- How to get score function: data-driven v.s. analytical.
- What to diffuse: sequential v.s. non-sequential.

📚 Literatures: Recent Research Progress in Diffusion for RL/Control

A detailed summary of each method can be found here.

The key of diffusion: how to get the score function.

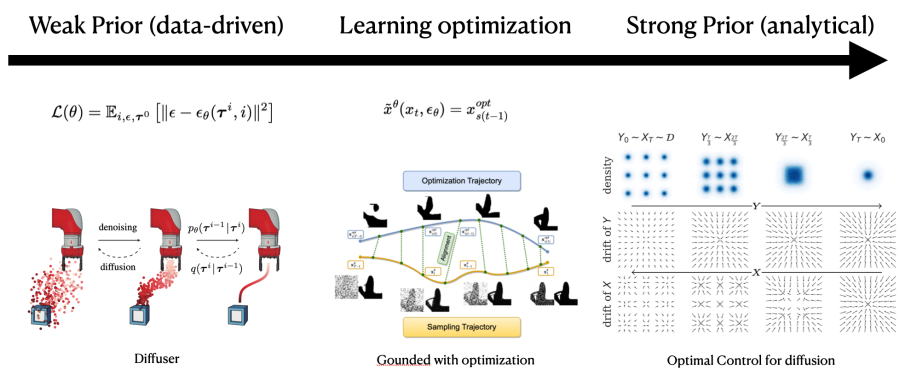
$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \left(\underbrace{x}_{\text{what to diffuse}} \mid \underbrace{y}_{\text{how to impose constraints/objectives}} \right)$$

- How to get score function: data-driven v.s. analytical.
- What to diffuse: sequential v.s. non-sequential.
- How to impose constraints/objectives: hard v.s. soft.

📚 Literatures: Recent Research Progress in Diffusion for RL/Control

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \left(\underbrace{x}_{\text{what to diffuse}} \middle| \underbrace{\quad y \quad}_{\text{how to impose constraints/objectives}} \right)$$

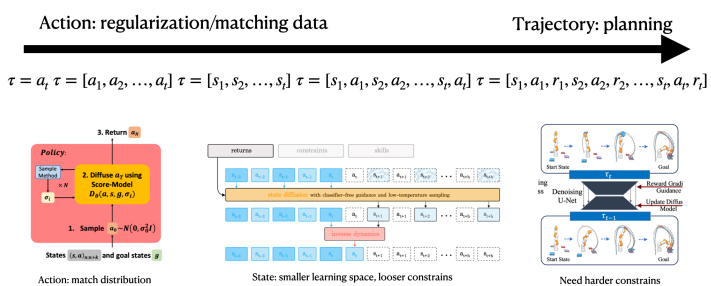
- How to get score function: data-driven v.s. analytical.
  - Data-driven: learn the score function from data.
  - Hybrid: learning from optimization intermediate results.
  - Analytical: use the analytical score function.



📚 Literatures: Recent Research Progress in Diffusion for RL/Control

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \quad (\quad \underbrace{x}_{\text{what to diffuse}} \quad | \quad \underbrace{y}_{\text{how to impose constraints/objectives}} \quad)$$

- How to get score function: data-driven v.s. analytical.
- What to diffuse: sequential v.s. non-sequential.
  - Action/Value: learn a model to match action/value distribution, serve as regularizer and policy.
  - Transition: learn a model to match transition distribution, serve as a world mode. ▶ MPC
  - Trajectory: learn a model to match trajectory distribution, serve as a TO solver. (planning state v.s. state-action v.s. action)



📚 Literatures: Recent Research Progress in Diffusion for RL/Control

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \quad (\quad \underbrace{x}_{\text{what to diffuse}} \quad | \quad \underbrace{y}_{\text{how to impose constraints/objectives}} \quad)$$

- How to get score function: data-driven v.s. analytical.
- What to diffuse: sequential v.s. non-sequential.
- How to impose constraints/objectives: hard v.s. soft.
  - Guidance function: Predefined or learned
  - Classifier-free: Use the unconditional score and conditional score (most common)
  - Inpainting: Fix the state and fill in the missing parts of the distribution (complimentary to the other two)

📚 Literatures: Recent Research Progress in Diffusion for RL/Control

$$\underbrace{\nabla_x \log P}_{\text{how to get score function}} \quad (\quad \underbrace{x}_{\text{what to diffuse}} \quad | \quad \underbrace{y}_{\text{how to impose constraints/objectives}} \quad)$$

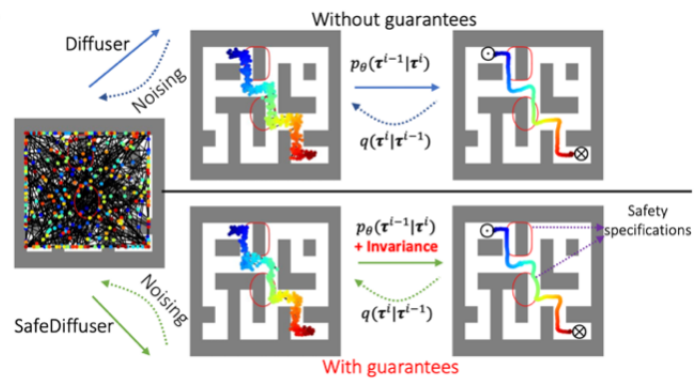
- How to get score function: data-driven v.s. analytical.
- What to diffuse: sequential v.s. non-sequential.
- How to impose constraints/objectives: hard v.s. soft.

SafeDiffuser: Safe Planning with Diffusion Probabilistic Models

Wei Xiao*, **Tsun-Hsuan Wang, Chuang Gan, Daniela Rus**

Algorithm 1 Enforcing invariance in diffusion models

Input: the last trajectory of diffusion τ^{j+1} at diffusion step $j \in \{0, \ldots, N\}$
Output: safe diffusion state τ^{j*}.
(a) Run diffusion procedure (4) and sample (5) as usual at step j and get τ^j.
(b) Find diffusion dynamics as in (6) - (7).
if *Robust-safe diffuser* **then**
  Formulate the QP (13), solve it and get u^{j*}.
else if Relaxed-safe diffuser **then**
  Define the time-varying weight $w_k(j)$ in (9), formulate the QP (14), solve it and get u^{j*}, r^{j*}.
else
  Design the time-varying function $\gamma_k(j)$ in (11), formulate the QP (13), solve it and get u^{j*}.
end if
(c) Update dynamics (7) with $u^j = u^{j*}$ and get τ^{j*}. Finally, $\tau^j \leftarrow \tau^{j*}$.



Summary & Challenges in Diffusion Models

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches demostration distribution from data.

📝 Summary & Challenges in Diffusion Models

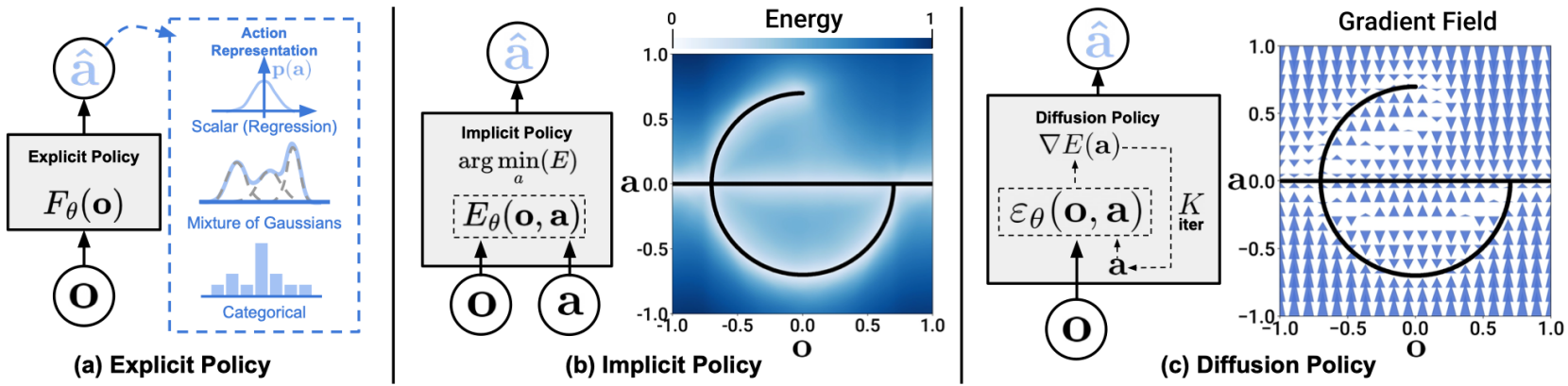
- Diffusion in robotics: matches demostration distribution from data.
- Use cases: imitation learning, offline RL, model-based RL.

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches demostration distribution from data.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, trajectory, or model as a regularizer/world model/planner.

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches dataset distribution in control and planning.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, planner, or model as a distribution matching problem.
- Advantages: high-dimensional matching, stability, scalability.



📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches dataset distribution in control and planning.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, planner, or model as a distribution matching problem.
- Challenges:

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches dataset distribution in control and planning.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, planner, or model as a distribution matching problem.
- Challenges:
  - 🕒 Computational cost: longer training and inference time.

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches dataset distribution in control and planning.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, planner, or model as a distribution matching problem.
- Challenges:
  - 🕒 Computational cost: longer training and inference time.
  - 🔀 Shifting distribution: difficulties in adapting to dynamic datasets.

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches dataset distribution in control and planning.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, planner, or model as a distribution matching problem.
- Challenges:
  - 🕒 Computational cost: longer training and inference time.
  - 🔀 Shifting distribution: difficulties in adapting to dynamic datasets.
  - 📊 High variance: inconsistent performance in precision tasks.

📝 Summary & Challenges in Diffusion Models

- Diffusion in robotics: matches dataset distribution in control and planning.
- Use cases: imitation learning, offline RL, model-based RL.
- Role: Learns policy, planner, or model as a distribution matching problem.
- Challenges:
  - 🕒 Computational cost: longer training and inference time.
  - 🔀 Shifting distribution: difficulties in adapting to dynamic datasets.
  - 📊 High variance: inconsistent performance in precision tasks.
  - ⛔ Constraint satisfaction: limited adaptability to new constraints.

🎉 Thank you!

Useful Resources

Paper list with labels

Diffusion for RL survey paper

Diffusion for RL repo

Awesome Diffusion RL repo