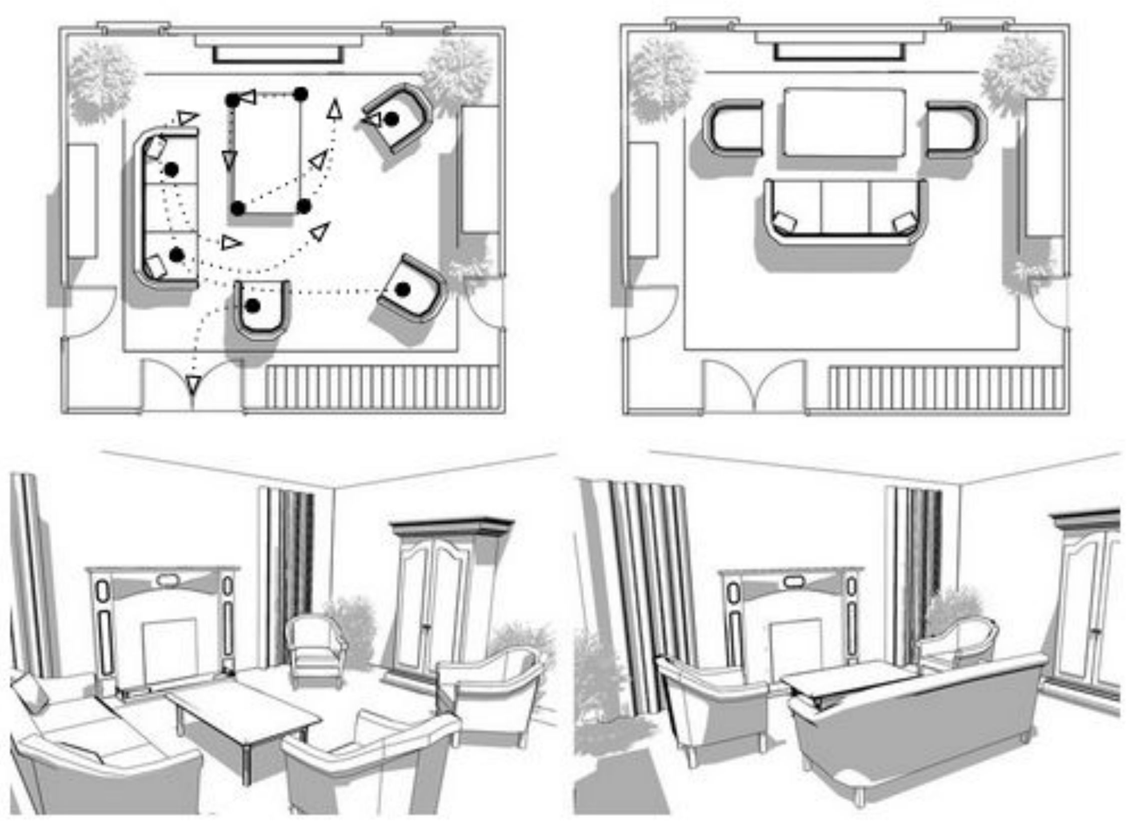
Physically Plausible Reverse Diffusion

Alexander Rosen
University of Toronto

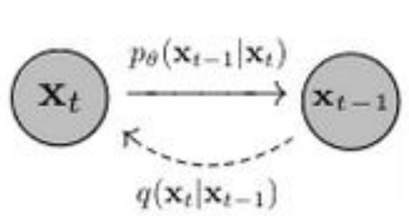
Motivation

Diffusion models have revolutionized computer vision by achieving unprecedented quality, fidelity, and diversity across various applications. To extend this success to real-world tasks, the principles and heuristics of diffusion models must be adapted to incorporate physical constraints into their formulation. For example, suppose we place furniture randomly in a room and want to sample a neat configuration of the given objects along with a process to reach this configuration:

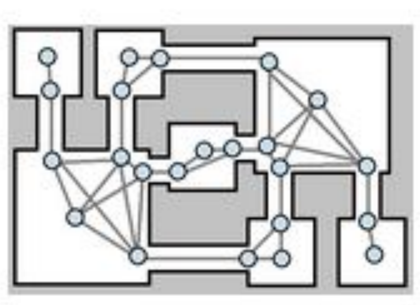


Related Work

- Simplest approach is to use any diffusion-based algorithm, say DDPM [1], to sample a final configuration



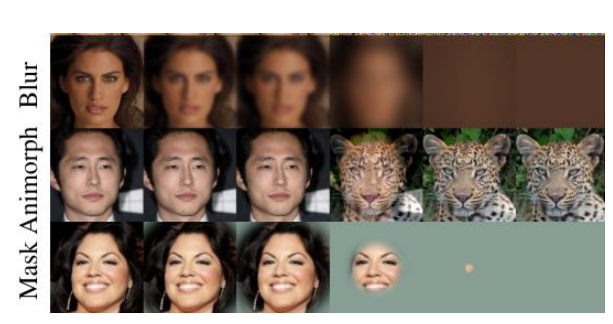
- Use a path finding algorithm to move objects through space to such configuration
- Approach may be intractable for complex objects and/or scenes



- Another approach is to setup a Markov decision process and learn a diffusion policy [2]
- Requires a dataset beyond final configurations, a robot to move the objects, and a problem specific model with no out of distribution guarantees

- A more elegant approach may be to assume we have some remote control over these objects and reasonable differentiable simulations for how these objects interact with each other and the scene
- We could combine the process of denoising to a final configuration with the process of moving the objects through space.

We may also wish for our diffusion model to support non-Gaussian distributed equilibriums (cold diffusion [3]).

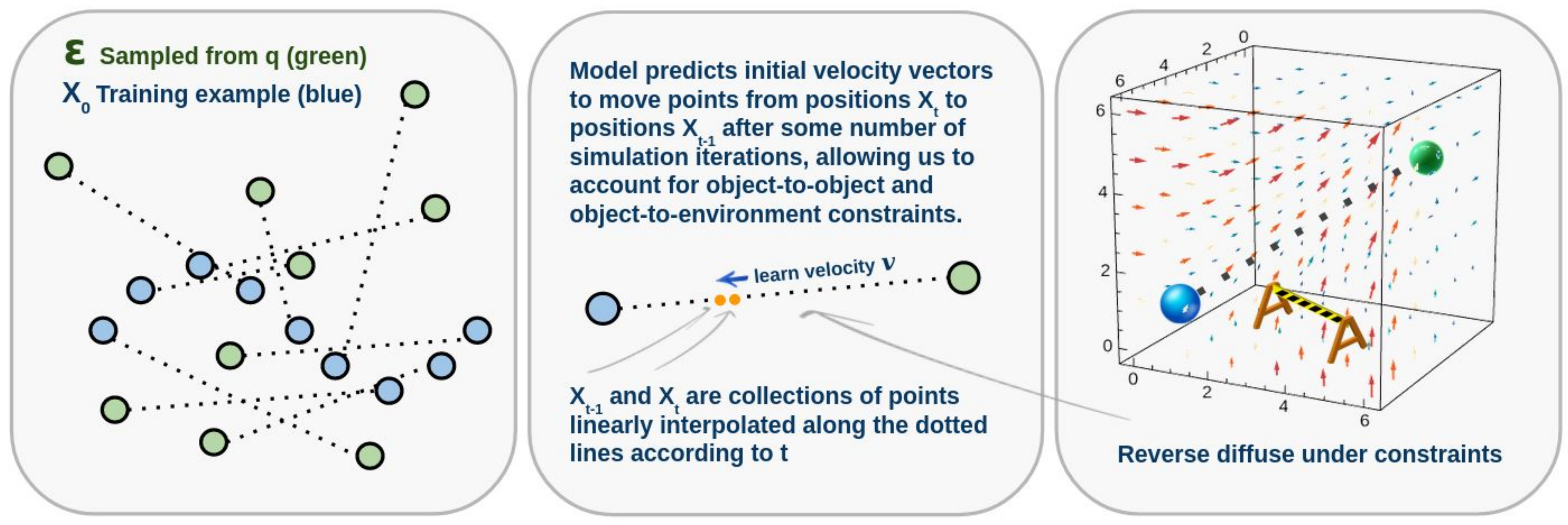


References

[1] Ho, Jain, Abbeel, *Denoising Diffusion Probabilistic Models*, NeurIPS, 2020.
[2] Chi, Xu, Feng, Cousineau, Du, Burchfiel, Tedrake, Song, *Diffusion Policy: Visuomotor Policy Learning via Action Diffusion*, The International Journal of Robotics Research, 2024.
[3] Bansal, Borgnia, Chu, Li, Kazemi, Huang, Goldblum, Geiping, Goldstein, *Cold Diffusion: Inverting Arbitrary Image Transforms Without Noise*, NeurIPS, 2024.

A New Generative Model

- Remove Markov chain "forward process" seen in most diffusion algorithms
- Make use of differentiable physics engine D for each timestep update
- Sample from any distribution q you want (can even be deterministic)



- Uniformly sample a timestep between 1 and the number of timesteps T (inclusive)
- Stochastically (linearly) interpolate between training examples and sampled noise
- Learn velocity vectors to move backward accounting for physical constraints

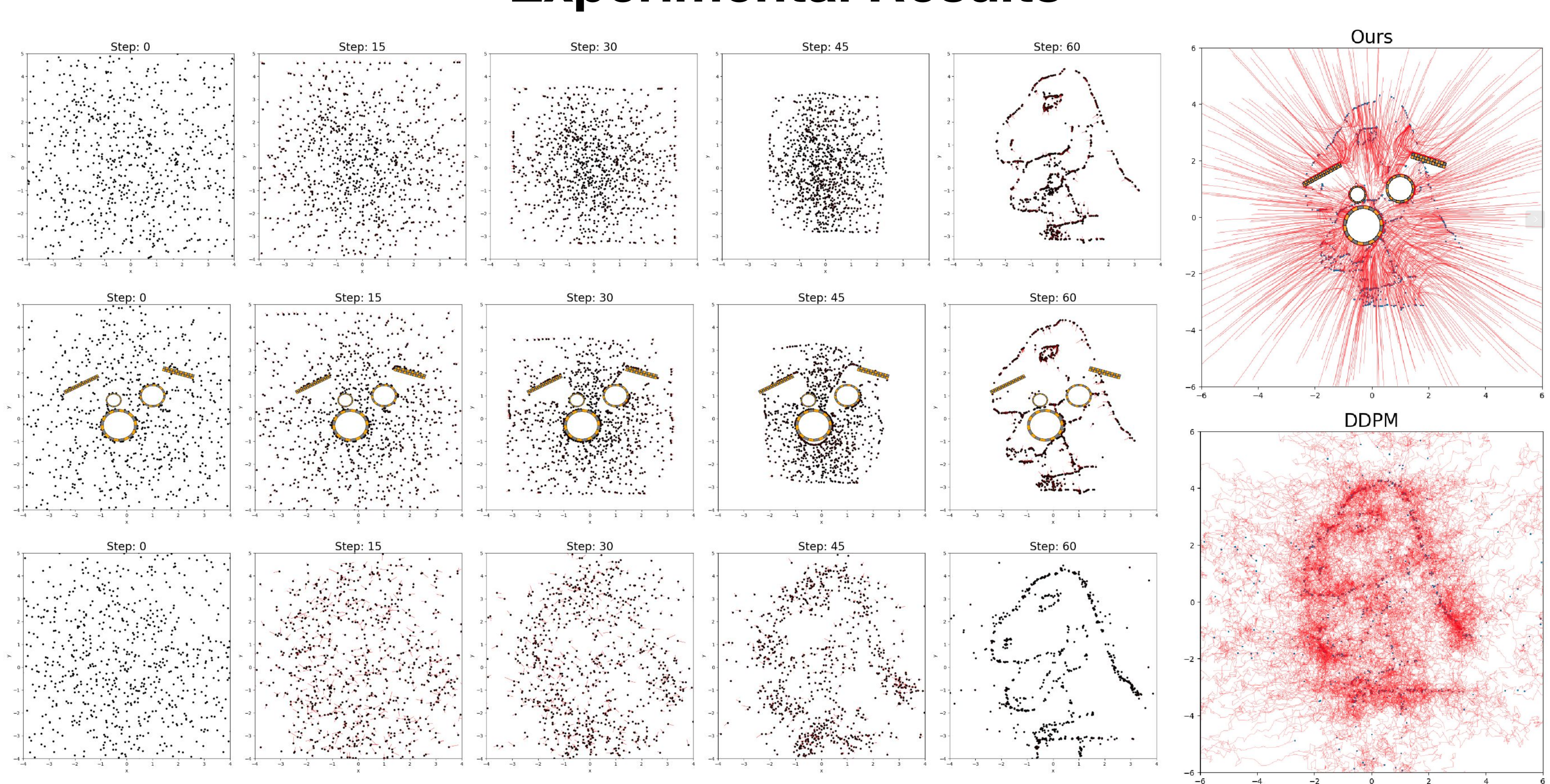
Algorithm 1 Train(x_0)

1: $t \sim \text{Unif}(1, T), \epsilon \sim q$
2: $x_t \leftarrow (1 - \frac{t}{T})x_0 + \frac{t}{T}\epsilon$
3: $x_{t-1} \leftarrow (1 - \frac{t-1}{T})x_0 + \frac{t-1}{T}\epsilon$
4: $v \leftarrow M(x_t, t)$
5: $\hat{x}_{t-1} \leftarrow D(x_t, v)$ ▷ Positions from simulation
6: Optimize with $\mathcal{L}(x_{t-1}, \hat{x}_{t-1})$

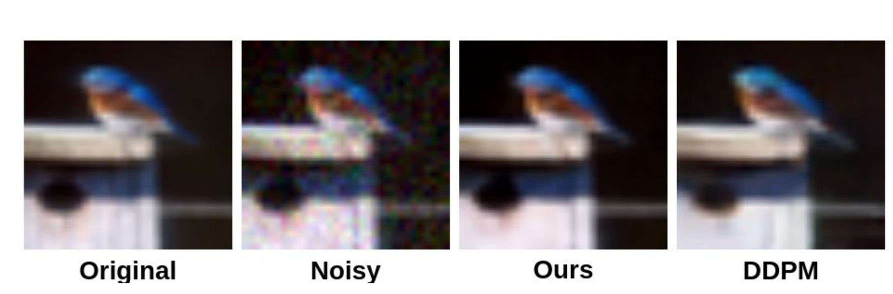
Algorithm 2 Sampling

1: $x_t \sim q$
2: **for** $t \leftarrow T$ to 1 **do**
3: $v \leftarrow M(x_t, t)$
4: $x_t \leftarrow D(x_t, v)$
5: **end for**
6: **return** x_t ▷ Sampled x_0

Experimental Results



	PSNR (denoising)	SSIM (denoising)	FID (generation)
DDPM	22.84	0.907	29.8
Ours	21.12	0.953	31.1



<u>Advantages</u>

- Matches generation quality and diversity of DDPM and similar methods
- Ability to sample initial positions from any distribution you want
- Only need a dataset of final positions
- Paths through space appear, quantitatively and qualitatively, efficient
- New positions at inference time come from the real world and not simulation
- Generalizes to objects and/or environments with complicated structure

<u>Disadvantages</u>

- Large training time overhead from physics engine
- Lack of experiments and testing compared to other approaches
- Linear interpolation can fail to produce optimal paths for scenes with several obstacles