Training-free approaches for image inversion and editing using latent diffusion and flow models

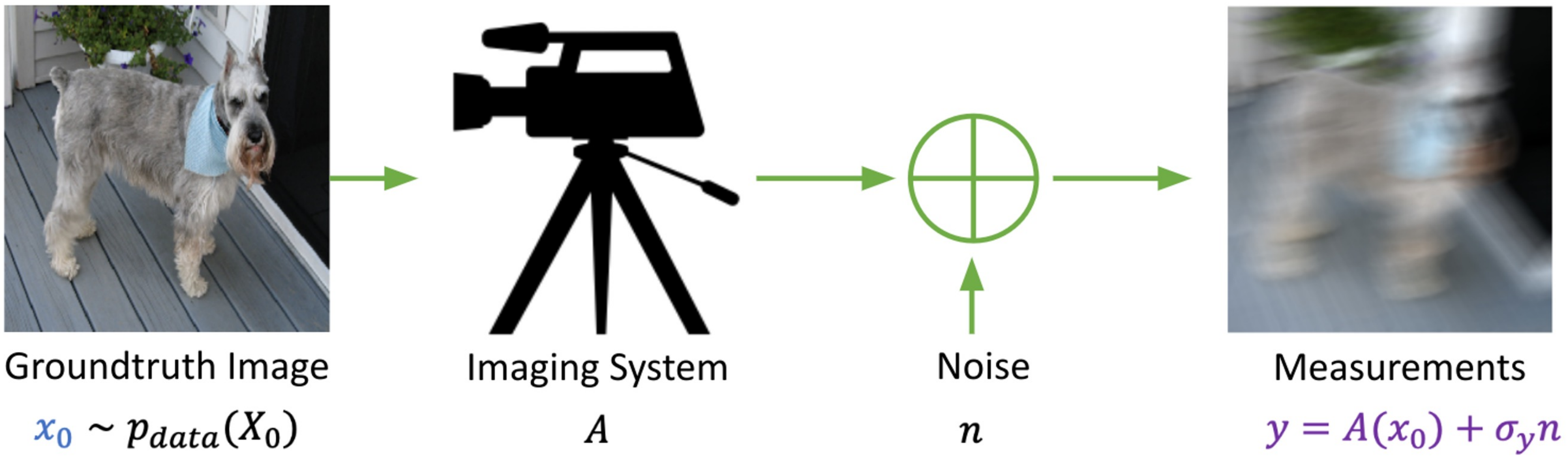
Sanjay Shakkottai

Based on joint work with: Litu Rout, Yujia Chen, Nataniel Ruiz, Abhishek Kumar*, Constantine Caramanis, and Wen-Sheng Chu

The University of Texas at Austin, Google Research, Google DeepMind

The University of Texas at Austin Machine Learning Laboratory
The University of Texas at Austin Chandra Department of Electrical and Computer Engineering
Cockrell School of Engineering
Google
Google DeepMind
IFML
AI EDGE Institute
WNCG

Inverse Problems Setting

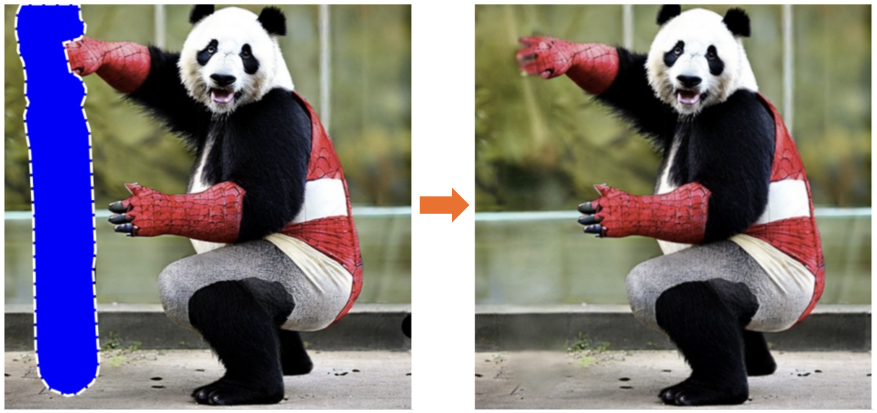


Problem: Reconstruct ground truth image x_0 from noisy measurements y

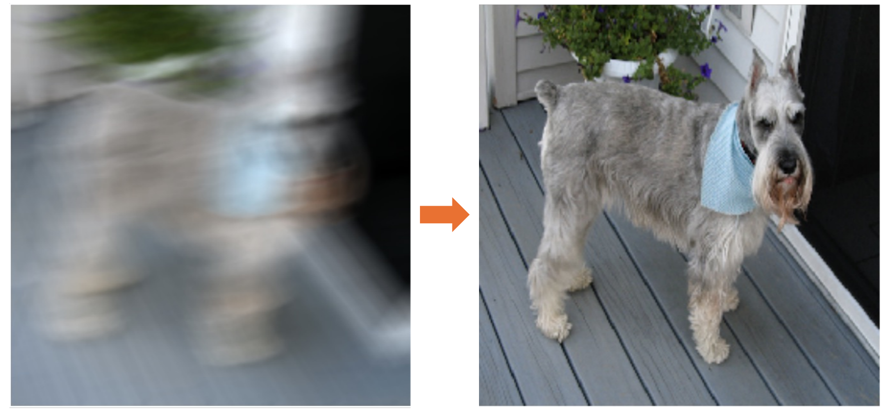
Challenge: Problem is ill-posed, that is infinitely many solutions x_0 exist

Approach: Use prior knowledge $p(x_0)$ of how the image should look like

Examples of Inverse Problems

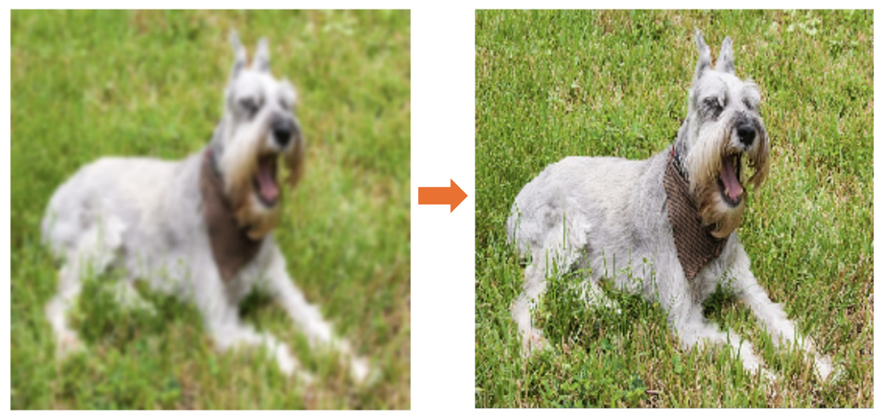


Free-form inpainting



Motion Deblur

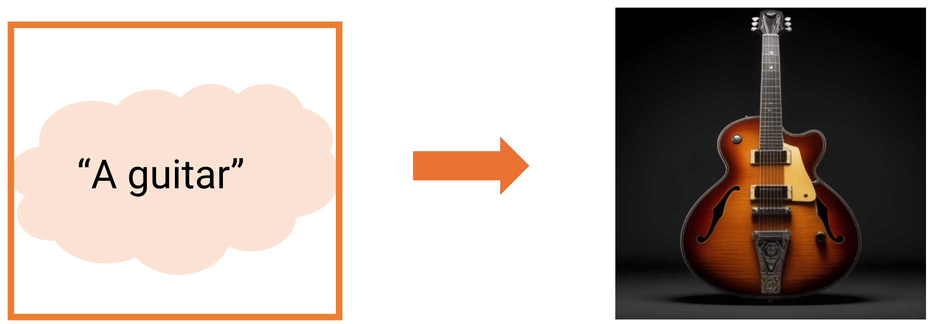
Super-resolution (4X)



Gaussian Deblur

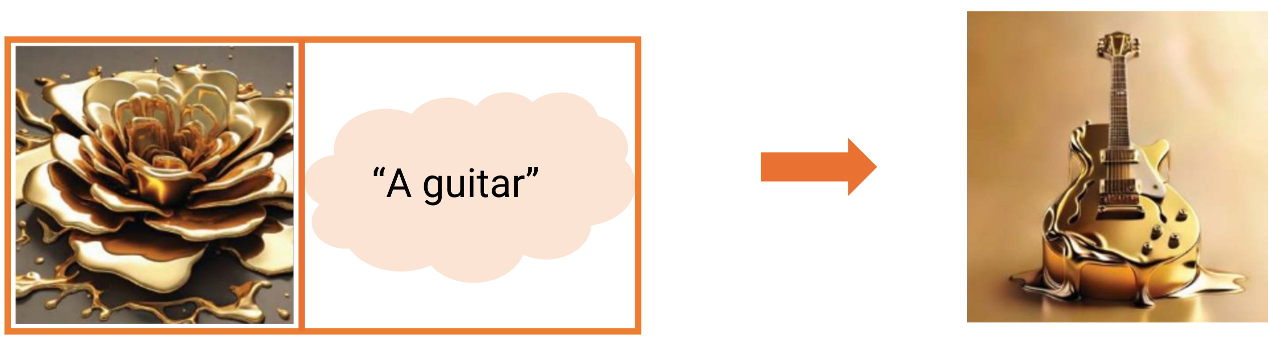
Stylization using Text and Image Prompts

Text-to-image generation



Input Output

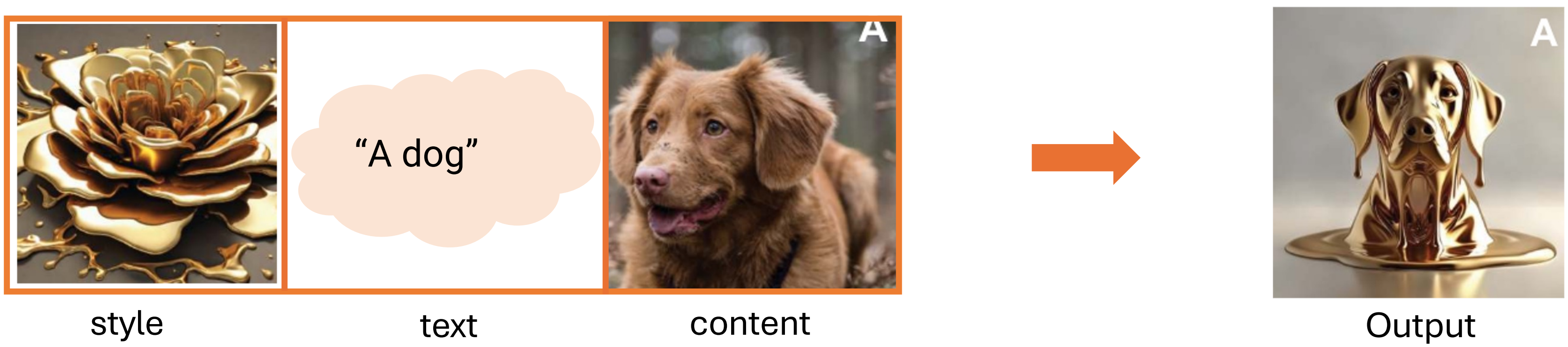
Personalized text-to-image generation: stylization



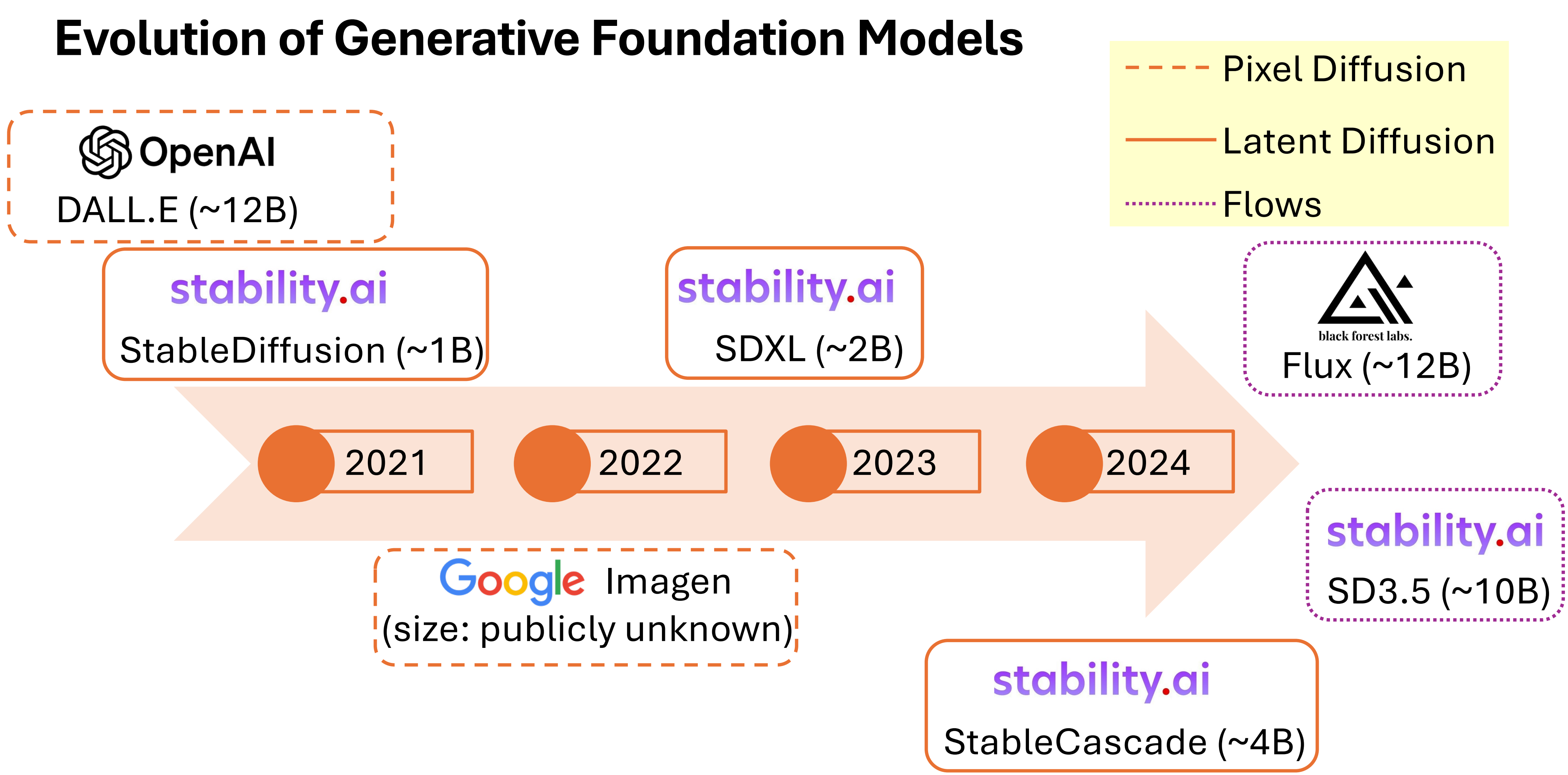
style text Output

Content-Style Composition Using Text and Image Prompts

Personalized text-to-image generation: content-style composition



Diffusion models have recently emerged as powerful foundation models for solving such generalized inverse / composition problems

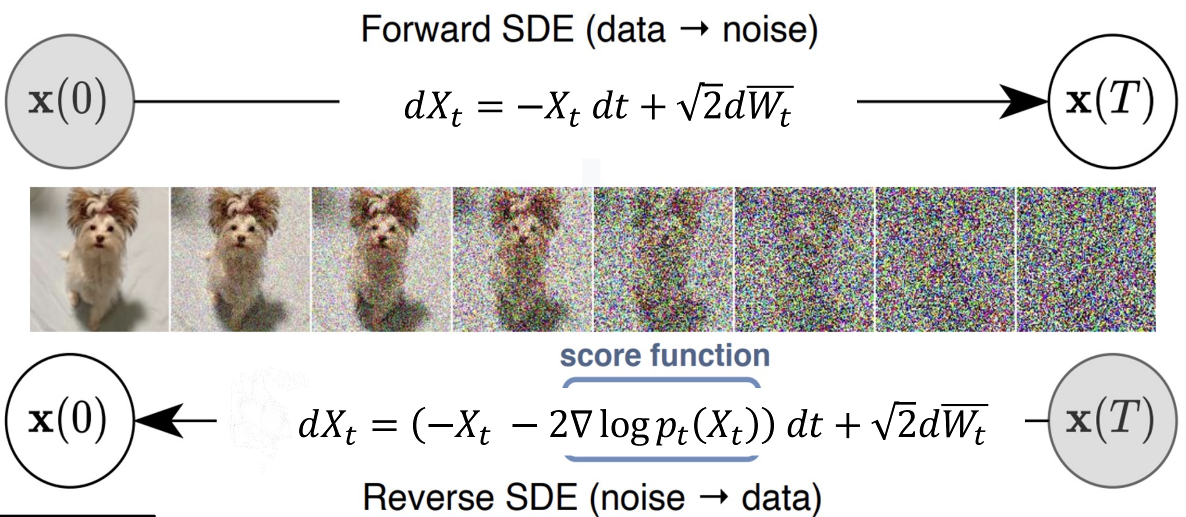

Evolution of Generative Foundation Models
Pixel Diffusion
Latent Diffusion
Flows
OpenAI
DALL.E (~12B)
stability.ai
StableDiffusion (~1B)
stability.ai
SDXL (~2B)
black forest labs.
Flux (~12B)
2021
2022
2023
2024
Google Imagen
(size: publicly unknown)
stability.ai
SD3.5 (~10B)
stability.ai
StableCascade (~4B)

Our Work on Inverse Problems and Editing using Latent Diffusion Models and Rectified Flows

- PSLD – First algorithm for solving inverse problems in latent space of diffusion models (NeurIPS 2023)
- STSL – Algorithm for inverse problems and image editing through efficient second-order methods (CVPR 2024)
- RB-Modulation – Algorithm for stylization and editing via Test-time Optimization using proximal methods (ICLR 2025, Oral)
  - Avoids backpropagation through score network
- RF-Inversion – First Algorithm for Inversion and Editing with Rectified Flow (ICLR 2025)

Focus of today's talk

Background on Diffusion Models



512 ×512 image ≈ 800K dim. vector

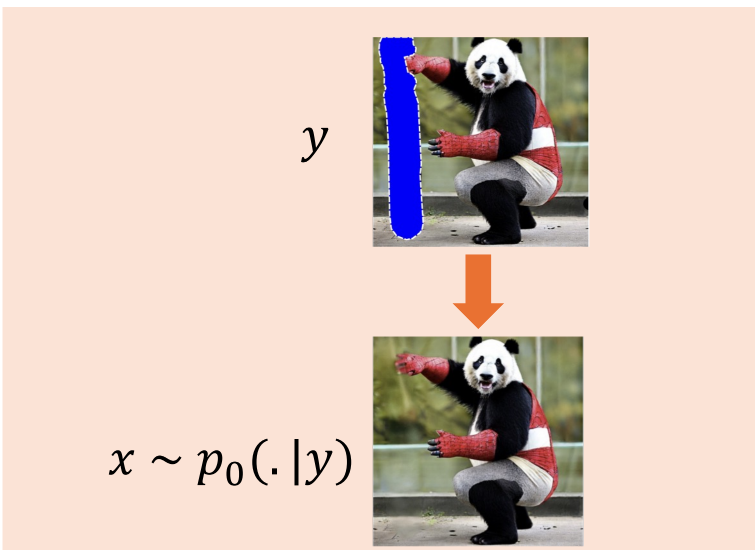
Image Credit: Song *et al.*, https://arxiv.org/pdf/2011.13456.pdf, ICLR'21

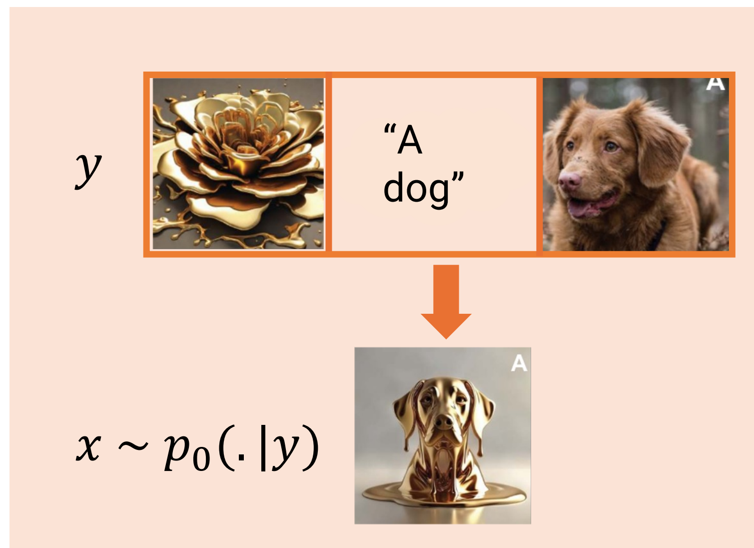
- Goal: Design a Markov process-based sampler (a transition kernel) such that stationary distribution samples images
- Approach: Learn annealed score that is affine in the conditional expectation of $X(0)$ (clean image) given $X(t)$ (noisy image) by Tweedie's Formula

References: Deep Unsupervised Learning using Diffusion (Sohl-Dickstein et al.' 2015); Score-based Generative Models (Song & Ermon' 2019); Diffusion Probabilistic Models (Ho et al.'2020); Score-based Generative Models through SDEs (Song et al.' 2021)

Posterior Sampling with Diffusion

- Inverse problems such as infilling, super resolution, denoising, editing, stylization, etc. are all examples of posterior sampling
- Goal: Given “measurement / context” y, generate a sample x, where $x \sim p_0(.\,|y)$





Posterior Sampling with Diffusion

Problem: Sample from $p_0(x_0|y)$ instead of $p(x_0)$

$$dX_t = (-X_t - 2\nabla \log p_t(X_t|y))\, dt + \sqrt{2}d\bar{W}_t, \qquad t = T, \cdots, 0$$

Unknown

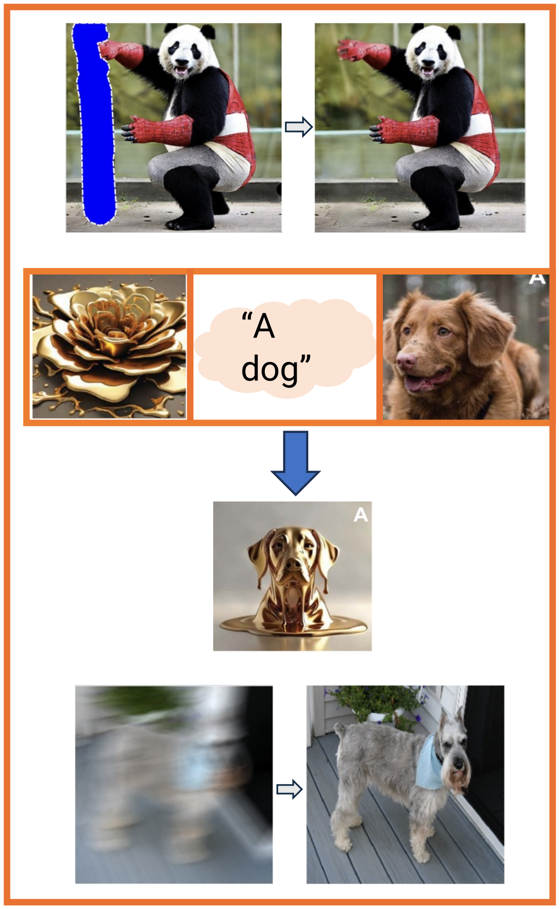
Bayes rule: $\log p_t(x_t|y) = \log p_t(y|x_t) + \log p_t(x_t) - \log p_t(y)$

$$dX_t = (-X_t - 2\nabla \log p_t(y|X_t) - 2\nabla \log p_t(X_t))\, dt + \sqrt{2}d\bar{W}_t, \qquad t = T, \cdots, 0$$

Unknown

Known: $\nabla \log p_t(X_t) \approx s_\theta(X_t, t)$

How well can we approximate $\nabla \log p_t(y|x_t)$?



DDRM: Kawar et al. https://arxiv.org/pdf/2201.11793.pdf , NeurIPS'21; DPS: Chung *et al.*, https://arxiv.org/pdf/2209.14687.pdf, ICLR'23
See also Delbracio & Milanfar, https://openreview.net/pdf?id=VmyFF5lL3F, TMLR'23 for an alternate formulation of supervised inverse problems.

PSLD: Posterior Sampling using Latent Diffusion

- First algorithm for solving inverse problems in latent diffusion (NeurIPS 2023)

- Generalizes prior work DPS (Chung et' al, ICLR 2023) that holds for pixel space diffusion

- $\nabla \log p_{T-t}(y|Z_t)$ ensures consistency w.r.t. the measurement y
  - Approximated using a Test Time Optimization (aka training-free) step
  - Requires a gradient computation with respect to the input to score $s_\theta(.,t)$ at each denoising step

PSLD (Rout et al.,NeurIPS'23):

$$\nabla \log p_{T-t}(y|Z_t) \approx \nabla \log p_0(y|Dec(\bar{Z}_T)) \;+\; \gamma_t \nabla \left|\left|\bar{Z}_T - Enc\left(A^T y + (I - A^T A)Dec(\bar{Z}_T)\right)\right|\right|^2$$

where $\bar{Z}_T = E_{Z_T \sim p_{T-t}(Z_T|Z_t)}[Z_T] = c_1(t) + c_2(t)s_\theta(Z_t, t)$

"Solving Linear Inverse Problems Provably using Posterior Sampling with Latent Diffusion Models", Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alexandros G. Dimakis, and Sanjay Shakkottai, *NeurIPS 2023*

Experimental Results with PSLD

(a) Input (b) Groundtruth (c) Comm. Serv. 1 (d) Comm. Serv. 2 (e) PSLD (Ours)

(a) Input (b) Groundtruth (c) DPS [11] (d) PSLD (Ours)

- Scalable to higher resolution images
    - Gradients computed in latent space
- One foundation model (SD1.5) many tasks
    - FFHQ (human faces) and ImageNet

- Convenient for real-world deployment
    - Images from the web, OOD samples
- No additional training or finetuning needed
    - Faster than pixel space diffusion

STSL: Second-order Tweedie from Surrogate Loss

- Algorithm for inverse problems and image editing through efficient second-order methods (CVPR 2024)
- Decreases bias of DPS or PSLD through a second order drift correction
- Requires only trace (a scalar quantity), leading to lighter computations
  - Estimate using an inner-loop stochastic approximation

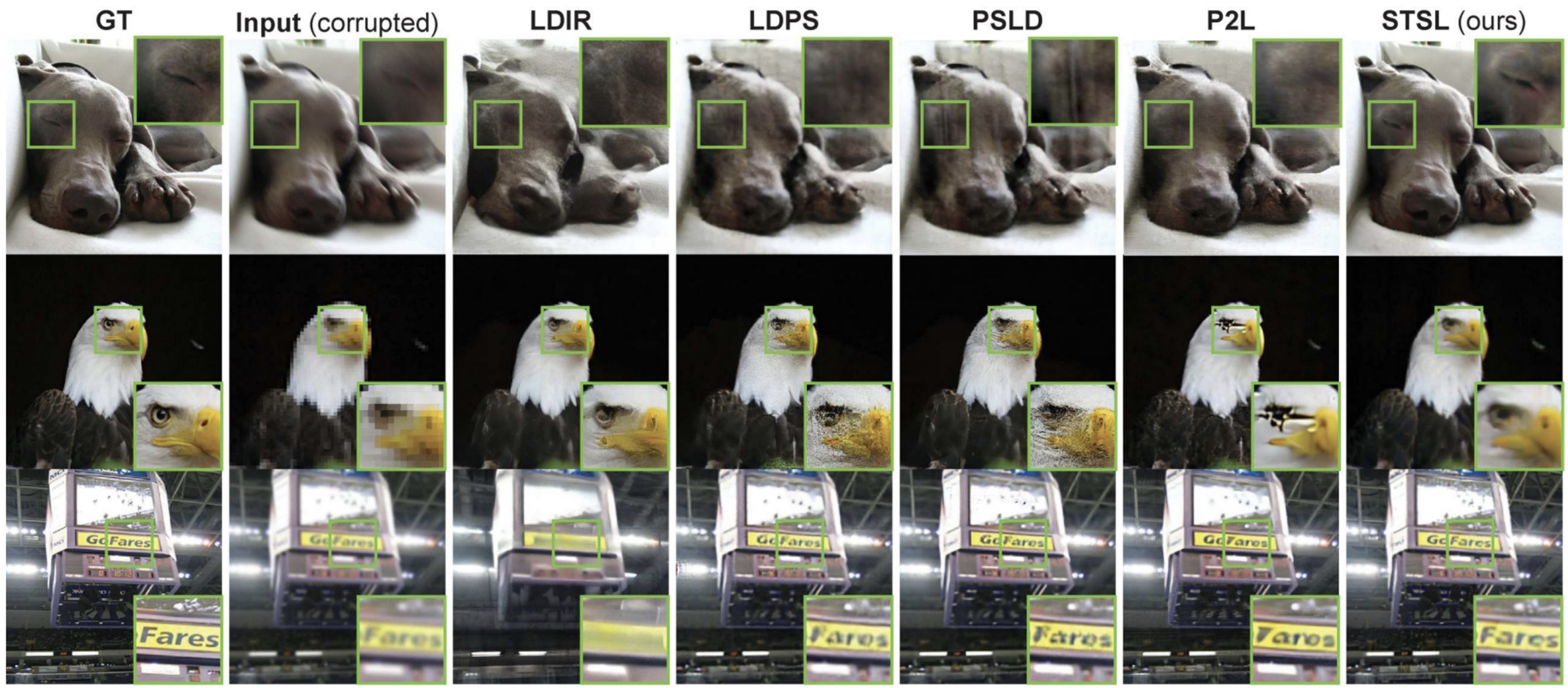
STSL (Rout et al., CVPR'2024):

$$\nabla \hat{L}(y, Z_t) = \nabla \log p_{T-t}(y|Dec(\bar{Z}_T)) - \gamma \nabla \, \mathrm{Trace}(\nabla^2 \log p_{T-t} \, (Z_t))$$

where $\bar{Z}_T = E_{Z_T \sim p_{T-t}(Z_T|Z_t)}[Z_T]$

"Beyond First-Order Tweedie: Solving Inverse Problems using Latent Diffusion", Litu Rout, Yujia Chen, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, Wen-Sheng Chu, CVPR 2024

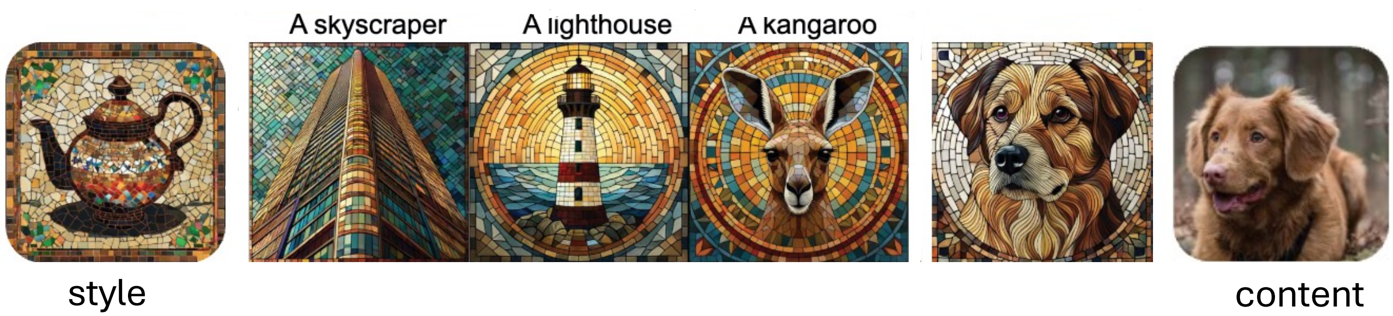
Experimental Results with STSL: ImageNet



First row: Motion Deblur, **Second row:** Super-resolution, **Third row:** Gaussian Deblur.

RB-Modulation for Content-Style Composition

- Algorithm for stylization and editing via Test-time Optimization using proximal methods (ICLR 2025, Oral)
  - Training-free approach
  - Avoids backpropagation through score network
  - All previous training-free algorithms (e.g., DPS, PSLD, STSL) require backpropagating through the score network to address inverse problems



"RB-Modulation: Training-Free Stylization using Reference-Based Modulation", Litu Rout, Yujia Chen, Nataniel Ruiz, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, Wen-Sheng Chu, *ICLR 2025 (Oral)*

Training vs Test-time Optimization

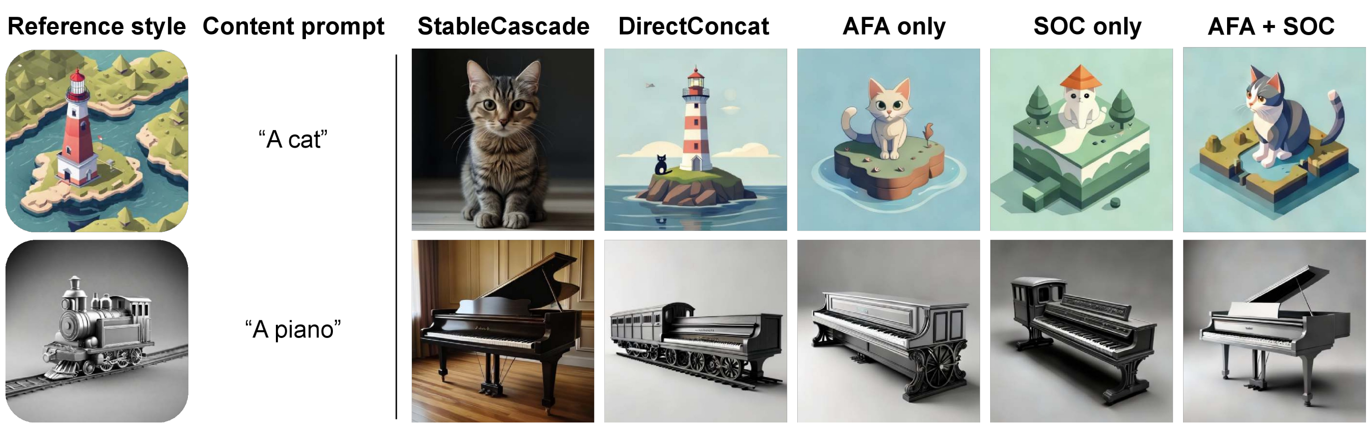
- Training-time optimization (DreamBooth, LoRA, IP-Adapter)
  - Approximately 10s of samples per conditioning (style/content)
    - Single sample leads to catastrophic forgetting
  - Gradient computed with respect to weights of base model
  - LoRA finetuning takes ~20 min per style (40 min for content-style)
  - Full finetuning takes hours

- Test-time optimization (DPS, PSLD, P2L, STSL)
  - Single sample suffices (no catastrophic forgetting)
  - Gradient computed with respect to input to base model
  - Takes ~10 min for PSLD (1B), ~20 min for P2L(1B) (longer for Flux-12B)

- Proximal test-time optimization (RB-Modulation)
  - Takes 40 sec using StableCascade (4B)

RB-Modulation: SOC and AFA

- RB Modulation has two key elements
  - Stochastic Optimal Controller (SOC) and Attention Feature Aggregation (AFA)
  - SOC: An optimal control formulation-based sampler, implemented as a test-time proximal optimizer
  - SOC: Incorporate desired attributes (e.g., style) in controller's terminal cost
  - AFA: Personalize the score and disentangle content-style from the reference images through an alternate cross-attention processor



Posterior Sampling using Diffusion and Optimal Control

Goal: Interpret posterior sampling as a control problem

Recall: Sample $p_X(\cdot\,|y)$ instead of $p_X(\cdot)$ using conditional reverse SDE

$$dX_t = (-X_t - 2\nabla \log p_t(X_t|y))\, dt + \sqrt{2} d\overline{W}_t, \qquad t = T, \cdots, 0$$

Approach: (i) Using Bayes rule, $\log p_t(x_t|y) = \log p_t(y|x_t) + \log p_t(x_t) - \log p_t(y)$

$$dX_t = (-X_t - 2\nabla \log p_t(y|X_t) - 2\nabla \log p_t(X_t))\, dt + \sqrt{2} d\overline{W}_t\,, \qquad t = T, \cdots, 0$$

(ii) Stochastic optimal control problem with terminal cost

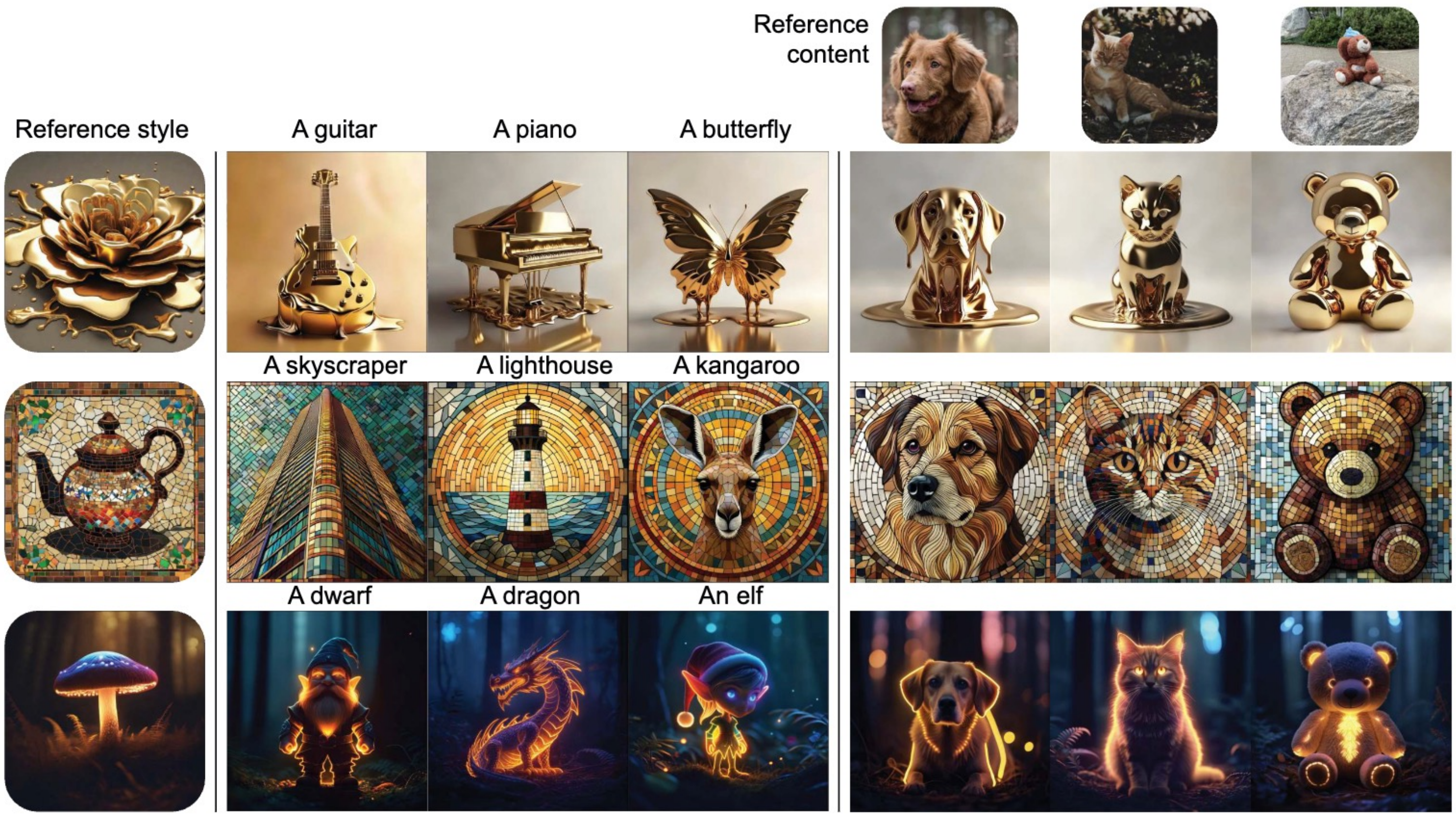
Terminal Cost

$$\min_{u \in U} E\left[\int_0^1 [\|u(X_t^u, t)\|^2]\, dt + g(X_1^u, y)\right]$$

Style Features

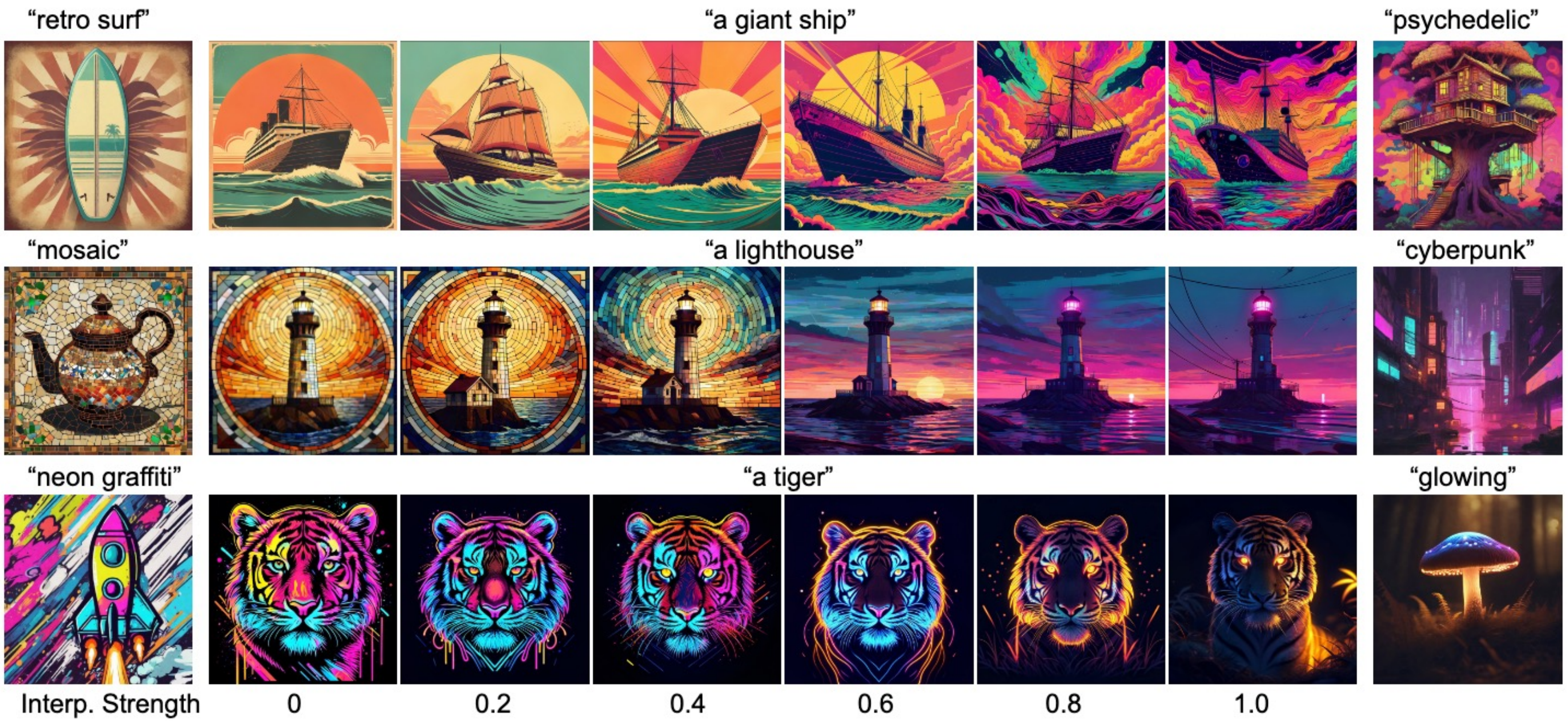
$$dX_t^u = (-X_t + u(X_t^u, t) - 2\nabla \log p_t(X_t))dt + \sigma(t) dW_t\,, \qquad X_0^u \sim p_0.$$

Personalization using RB-Modulation



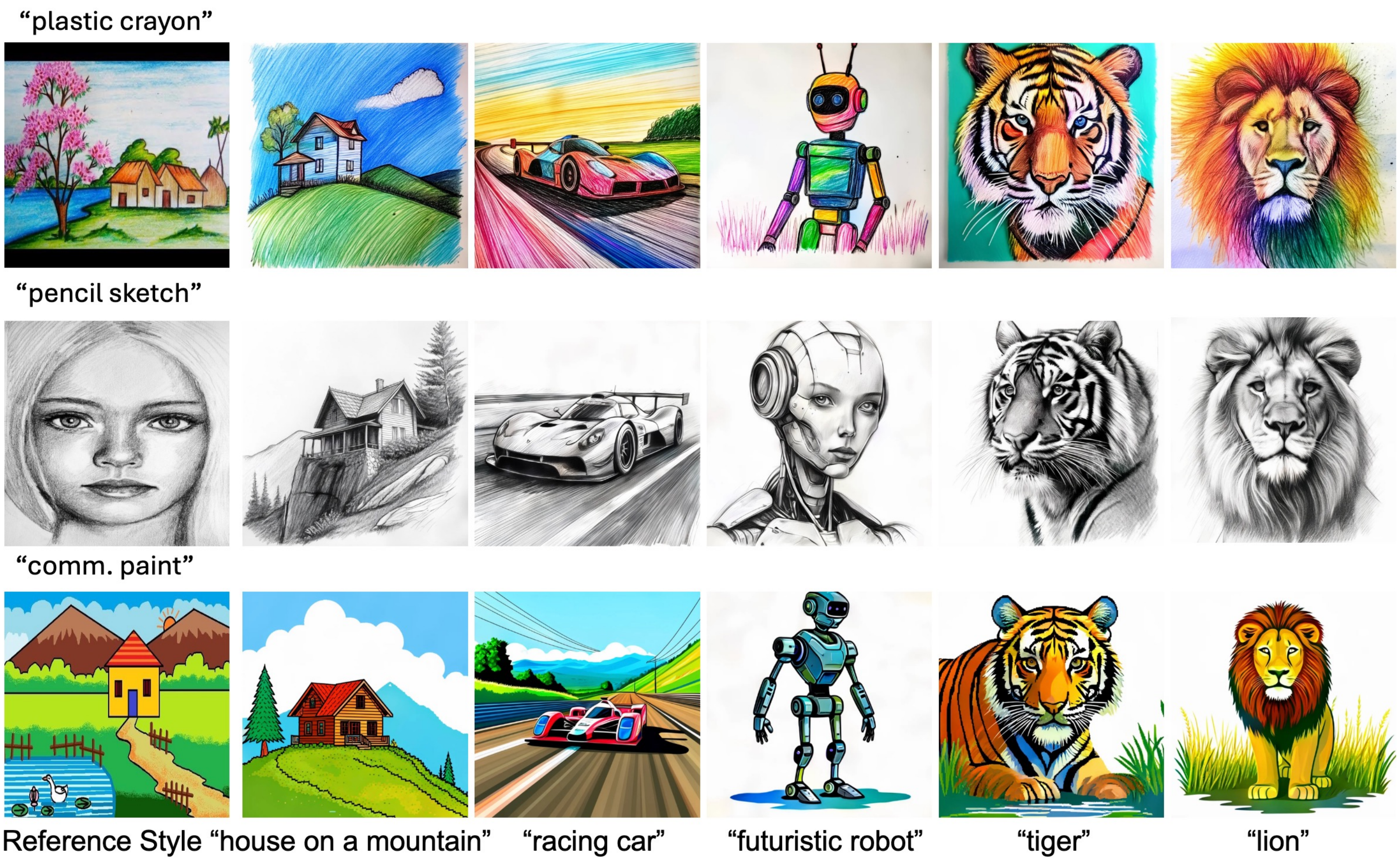
RB-Modulation as a plug-and-play solution for (a) stylization (b) content-style composition

Novel Style Synthesis: Interpolating Reference Styles



Training based methods cannot interpolate novel styles due to lack of prior examples

Stylization: Hand Drawn Reference Images



Please see: https://openreview.net/forum?id=bnINPG5A32 for reference image credits

RB-Modulation: Production Status

- Collaboration with Google and Google DeepMind researchers
  - Code available on github: https://rb-modulation.github.io
- Teams at Google are currently productionizing RB Modulation into several devices and production pipelines
  - Pixel, Chromebook, Tablet, and YouTube
  - Several application settings (e.g., on device personalization)
- Demo became #1 on HuggingFace in the week of its release

Reference Style "mountain" "pillow" "building" "bottle" "turtle"

Emerging Foundation Models: Rectified Flows

RF-Inversion

- First Algorithm for Inversion and Editing with Rectified Flow (ICLR 2025)
  - Rectified Flow models are current SOTA (Flux, SD3.5)
  - RF-Inversion avoids any test-time optimization
  - Can implement on edge device such as Pixel

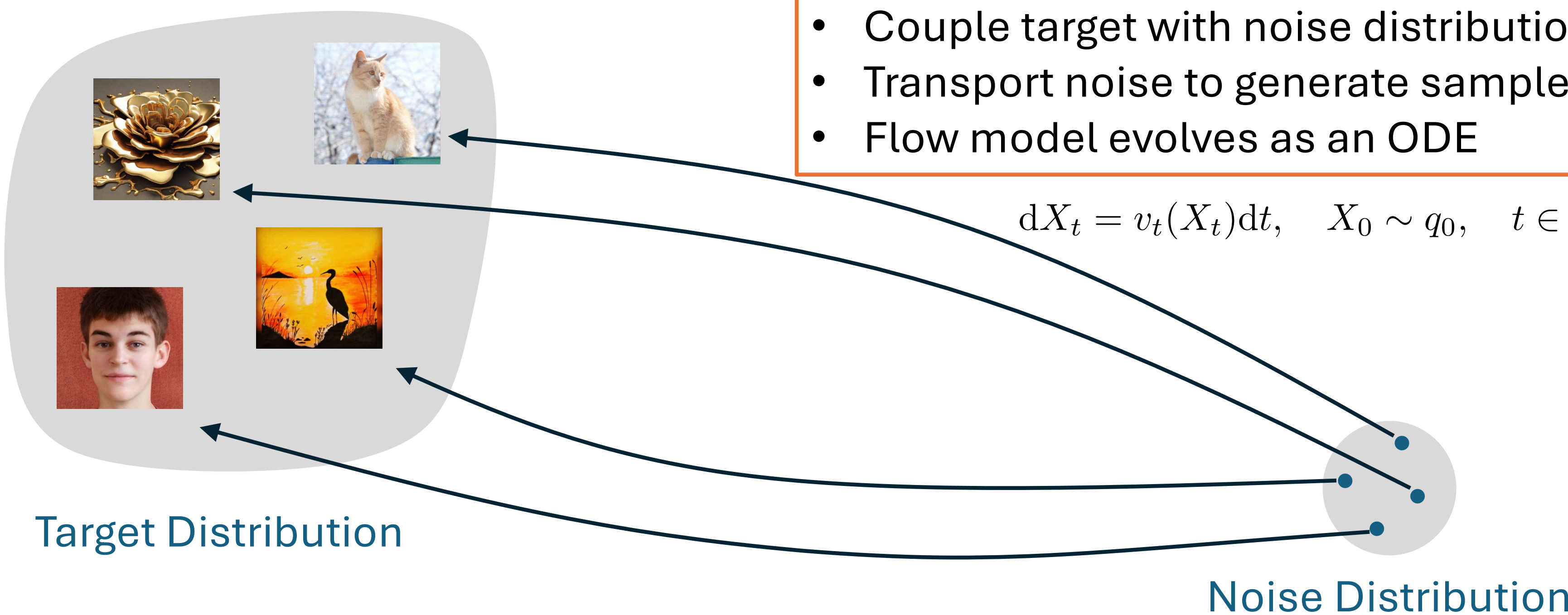
"Semantic Image Inversion and Editing using Stochastic Rectified Differential Equations", Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, Wen-Sheng Chu, *ICLR 2025*

Goal of Rectified Flows

Generate samples from a target distribution given a (large) finite number of samples from that distribution

- Couple target with noise distribution
- Transport noise to generate samples
- Flow model evolves as an ODE

$$\mathrm{d}X_t = v_t(X_t)\mathrm{d}t, \quad X_0 \sim q_0, \quad t \in [0,1]$$

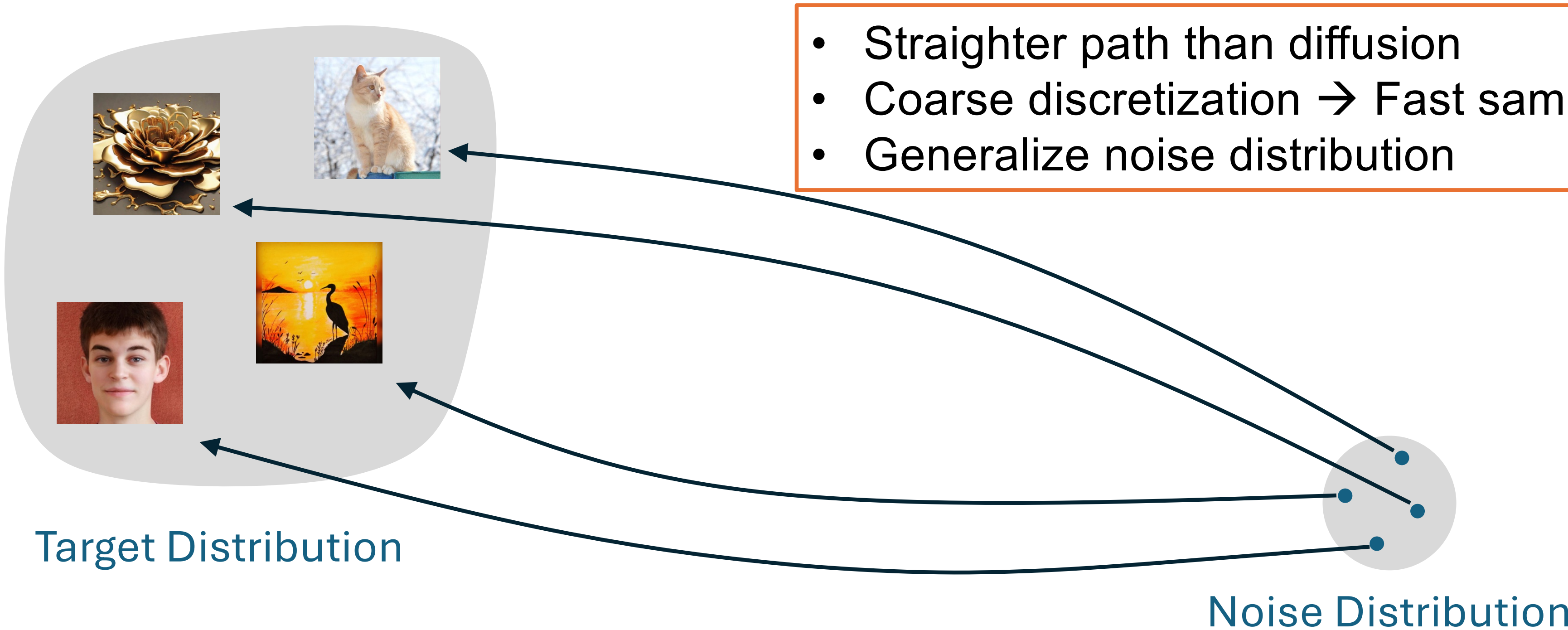


RF: Liu et. al., Flow straight and fast: Learning to generate and transfer data with rectified flow, ICLR 2022; Lipman et. al. Flow matching for generative modeling, 2022; Albergo and Vanden-Eijnden, Building normalizing flows with stochastic interpolants; ICLR 2023.

Benefit of Rectified Flows

Generate samples from a target distribution given a (large) finite number of samples from that distribution

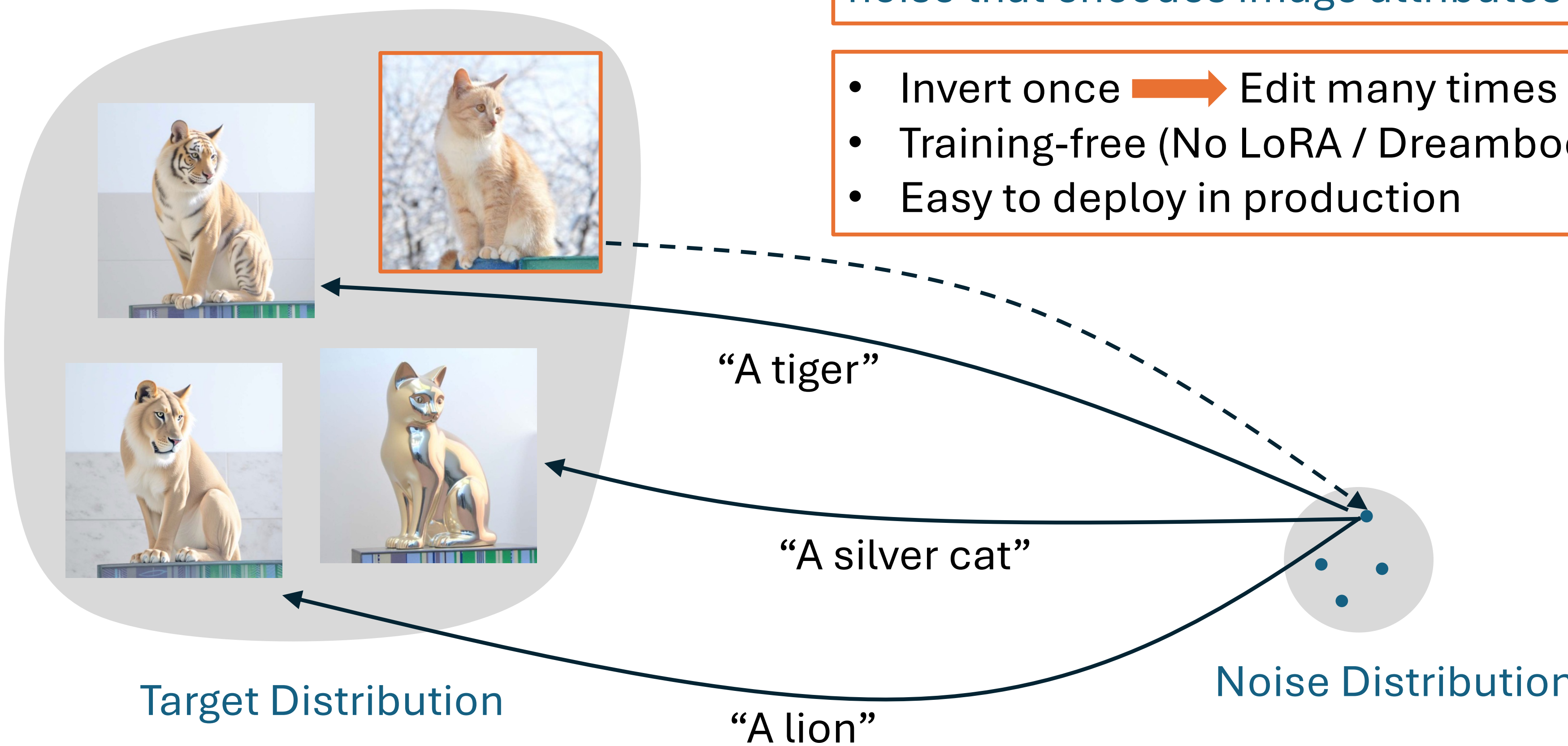
- Straighter path than diffusion
- Coarse discretization → Fast sampling
- Generalize noise distribution



Inversion with RF (1/2)

Inversion: Transform image into structured noise that encodes image attributes

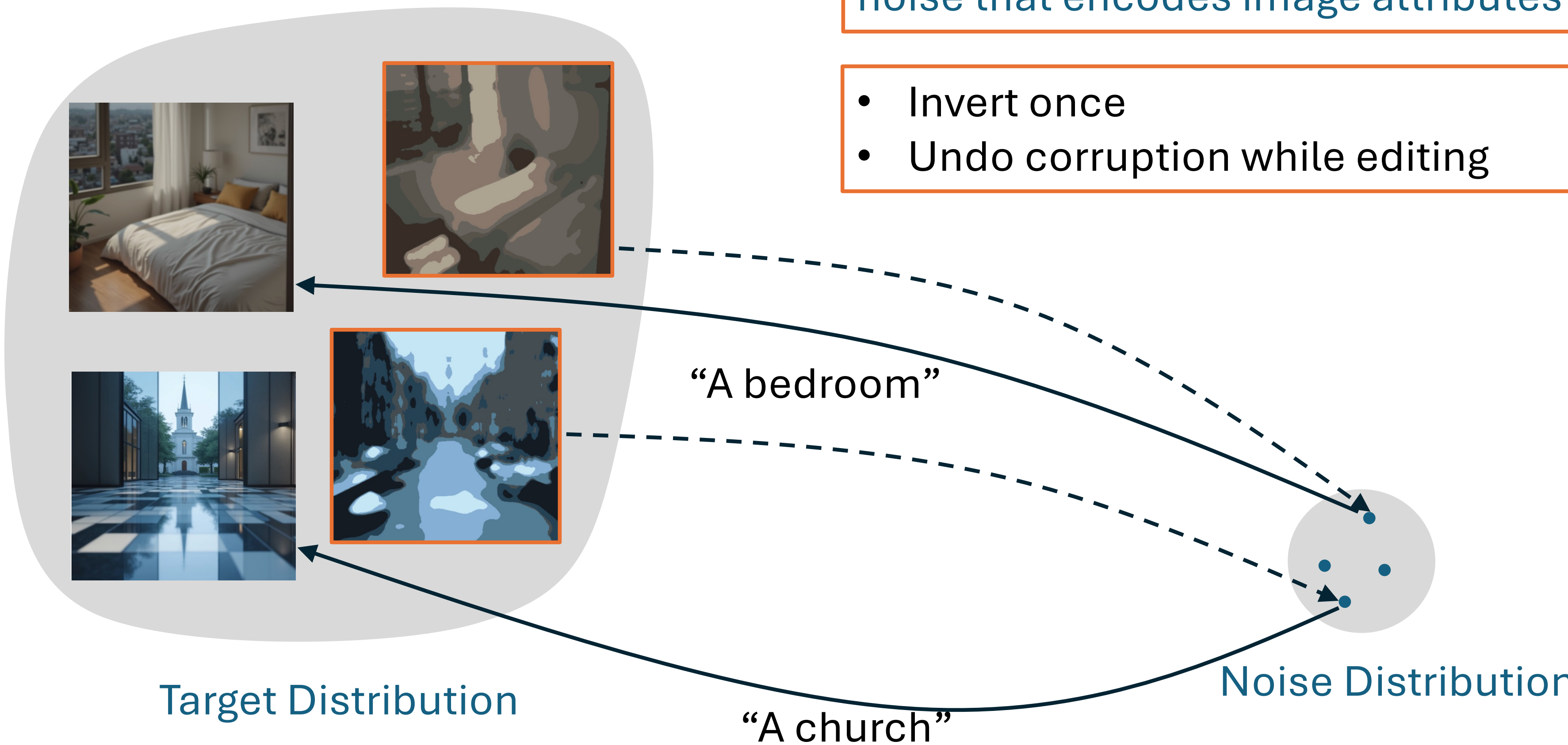
- Invert once ➡ Edit many times
- Training-free (No LoRA / Dreambooth)
- Easy to deploy in production



Inversion with RF (2/2)

Inversion: Transform image into structured noise that encodes image attributes

- Invert once
- Undo corruption while editing



State-of-the-art Inversion

- No algorithm to directly invert and edit using rectified flows
- Other approaches available for diffusion models
  - Inversion possible through SDEdit and DDIM inversion (for diffusions) but ...
    - They lead to inconsistencies (preservation of conditioning structure/layout) due to highly non-linear sample paths
  - Alternate methods maintain consistency through expensive training (e.g., DreamBooth, LoRA), test-time optimization (RB Modulation), or complex attention processors (NTI, P2P)

Related Works: Inversion and Editing using Diffusion Models

Method	**Training**	**Optimization**	**Attention Manipulation**
SDEdit [MHS+22]	✗	✗	✗
DDIM [SME21]	✗	✗	✗
NTI [MHA+23]	✗	✓	✗
NTI+P2P [HMT+22]	✗	✓	✓
LEDIT++ [BFK+24]	✗	✗	✓
InstructPix2Pix [BHE23]	✓	✗	✗
Ours	✗	✗	✗

- Diffusion models are the mainstream approach for inversion and editing
- SoTA methods require training, optimization, or attention manipulation
- SDEdit, DDIM, NTI, NTI+P2P are leading training-free methods
- NTI and P2P require test-time optimization or complex attention processors

Goal: Inversion and Editing using Rectified Flows

- Diffusion models (DMs) traditionally outperformed Rectified Flows (RFs)
- SD3.5 and Flux show RFs can beat DMs

- RF-Inversion or editing remain unexplored

- DM inversion techniques face challenges in RFs
    - Training of additional parameters (DreamBooth, StyleDrop)
    - Optimization of latent variables (RB-Modulation)
    - No null conditioning in distilled Flux (NTI)
    - Complex cross-attention processors (P2P)

First efficient inversion and editing using rectified flows without training, optimization, complex attention processors

Introduction to Rectified Flows

Goal: Generate samples from a target distribution given a finite number of samples from that distribution

Approach: Simulate an ODE to generate samples

$$\mathrm{d}X_t = v_t(X_t)\mathrm{d}t, \quad X_0 \sim q_0, \quad t \in [0,1]$$

vector field / drift

initialization

Normalized time

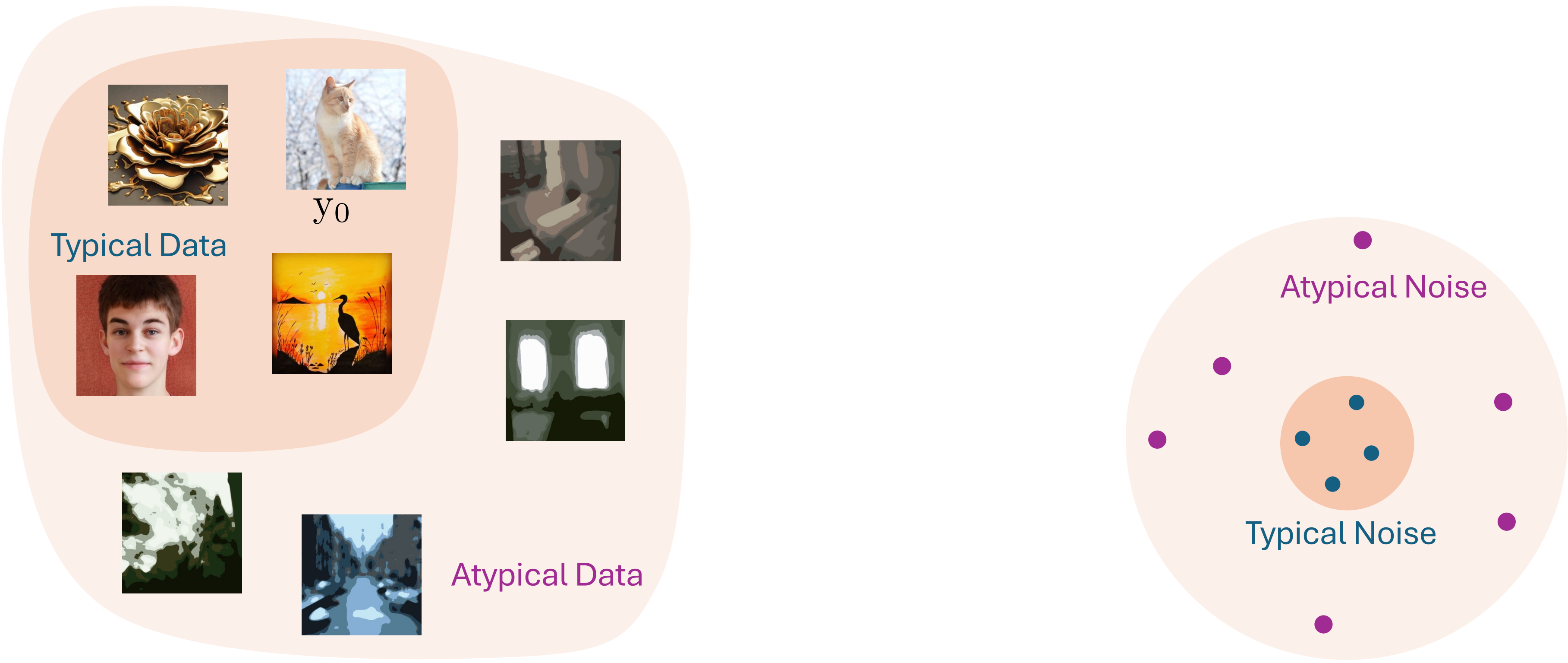
A common choice: $X_0 \sim \mathcal{N}(0, I)$ and $v_t(\cdot) = -u(\cdot, 1-t; \phi)$

$u(\cdot, \cdot; \phi)$ is a Neural Network (NN) trained using Conditional Flow Matching (CFM)

RF: Liu et. al., Flow straight and fast: Learning to generate and transfer data with rectified flow, ICLR 2022; Lipman et. al. Flow matching for generative modeling, 2022; Albergo and Vanden-Eijnden, Building normalizing flows with stochastic interpolants; ICLR 2023.

Our Approach: RF-Inversion

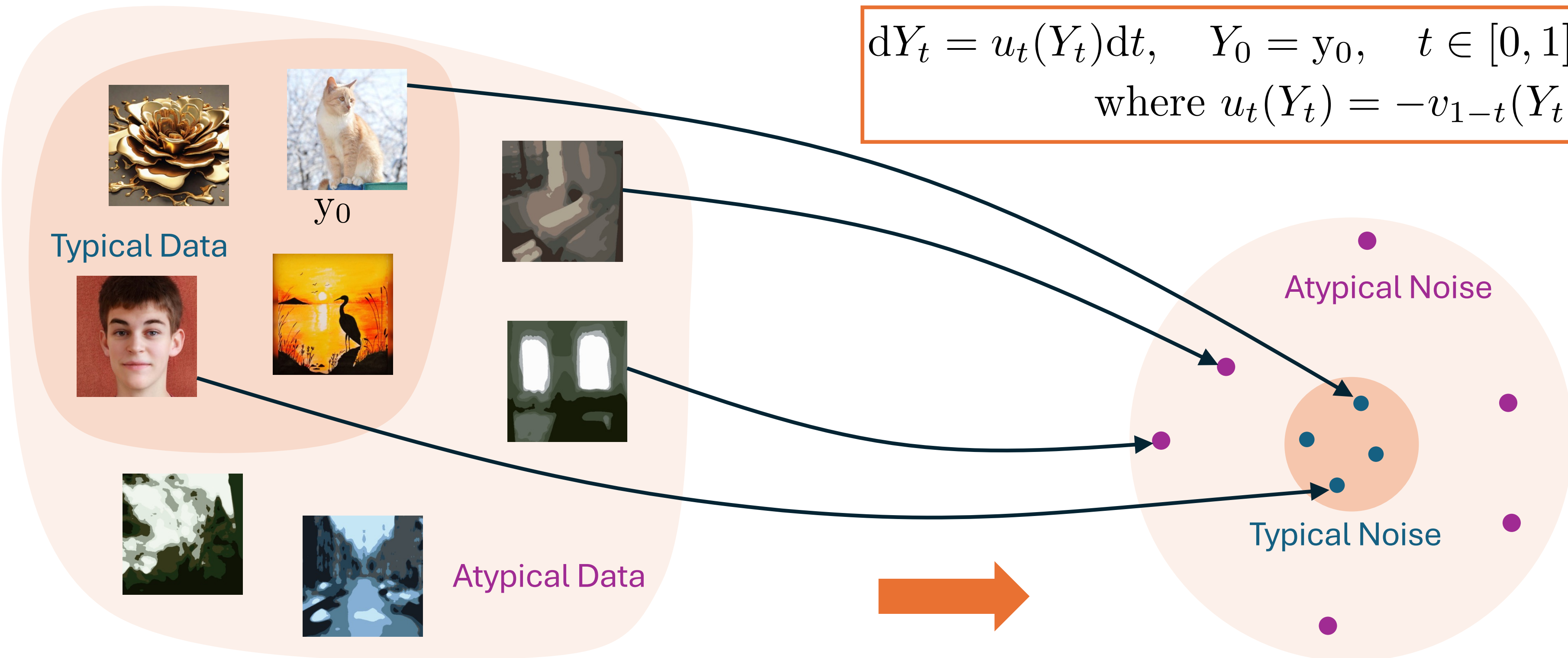
Inversion using Rectified Flows



Distributions can be (roughly) grouped into two types: typical and atypical

Inversion using Rectified Flows

$$\mathrm{d}Y_t = u_t(Y_t)\mathrm{d}t, \quad Y_0 = \mathrm{y}_0, \quad t \in [0,1],$$
$$\text{where } u_t(Y_t) = -v_{1-t}(Y_t)$$

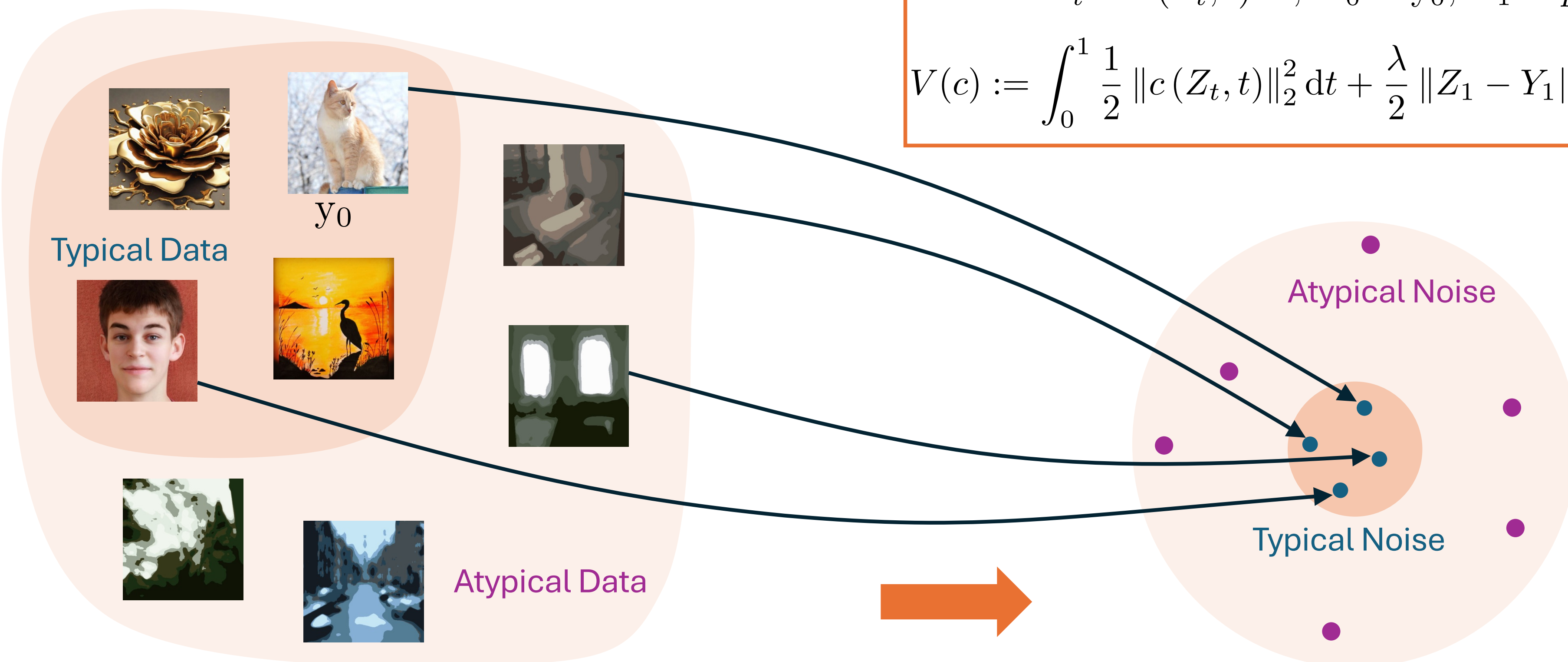


RF transforms typical image to typical noise; atypical image to atypical noise

Inversion using Optimal Control

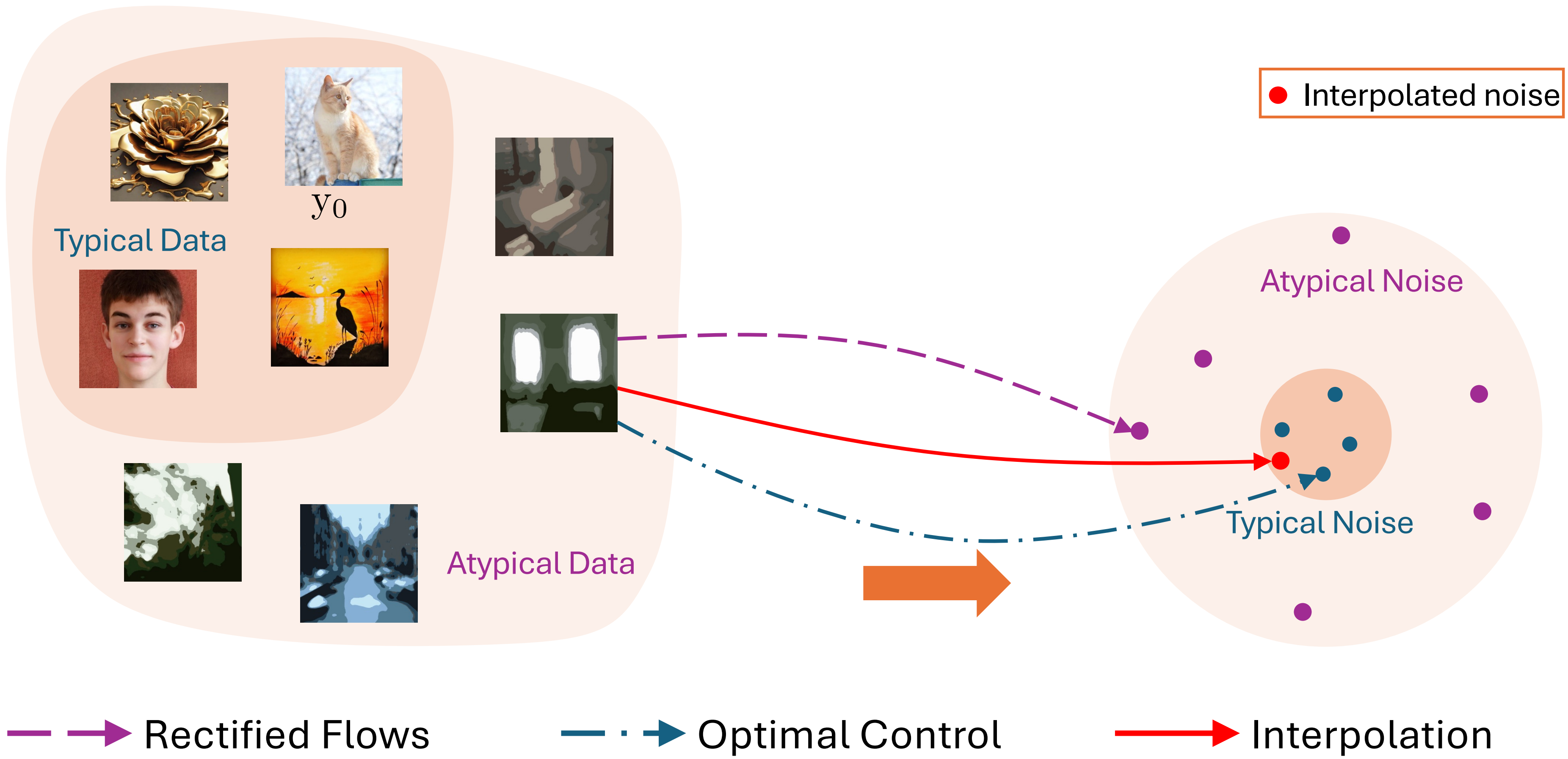
$$\mathrm{d}Z_t = c\,(Z_t, t)\,\mathrm{d}t,\;\; Z_0 = \mathrm{y}_0,\;\; Y_1 \sim p_1$$

$$V(c) := \int_0^1 \frac{1}{2}\,\|c\,(Z_t, t)\|_2^2\,\mathrm{d}t + \frac{\lambda}{2}\,\|Z_1 - Y_1\|_2^2$$

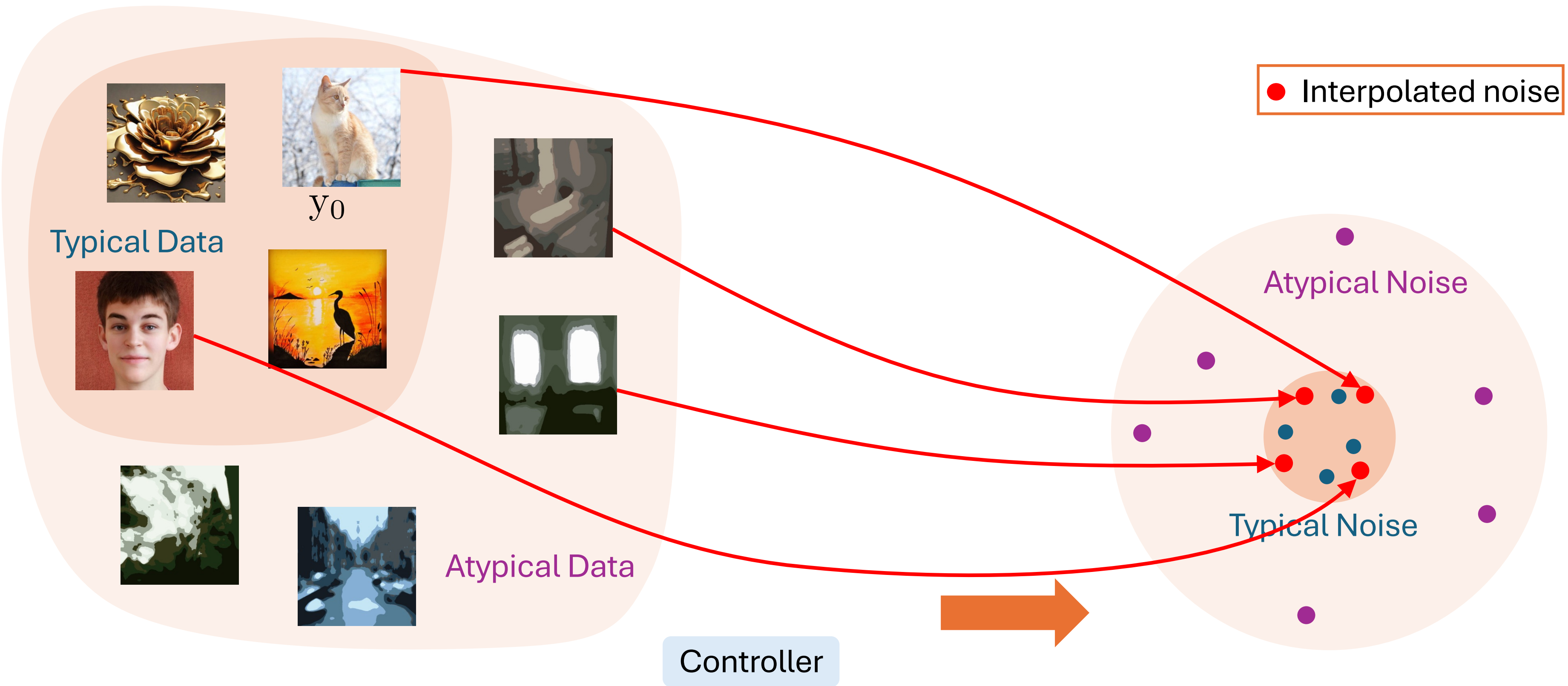


Optimal controller transforms any image to typical noise

Interpolation of the Two Fields

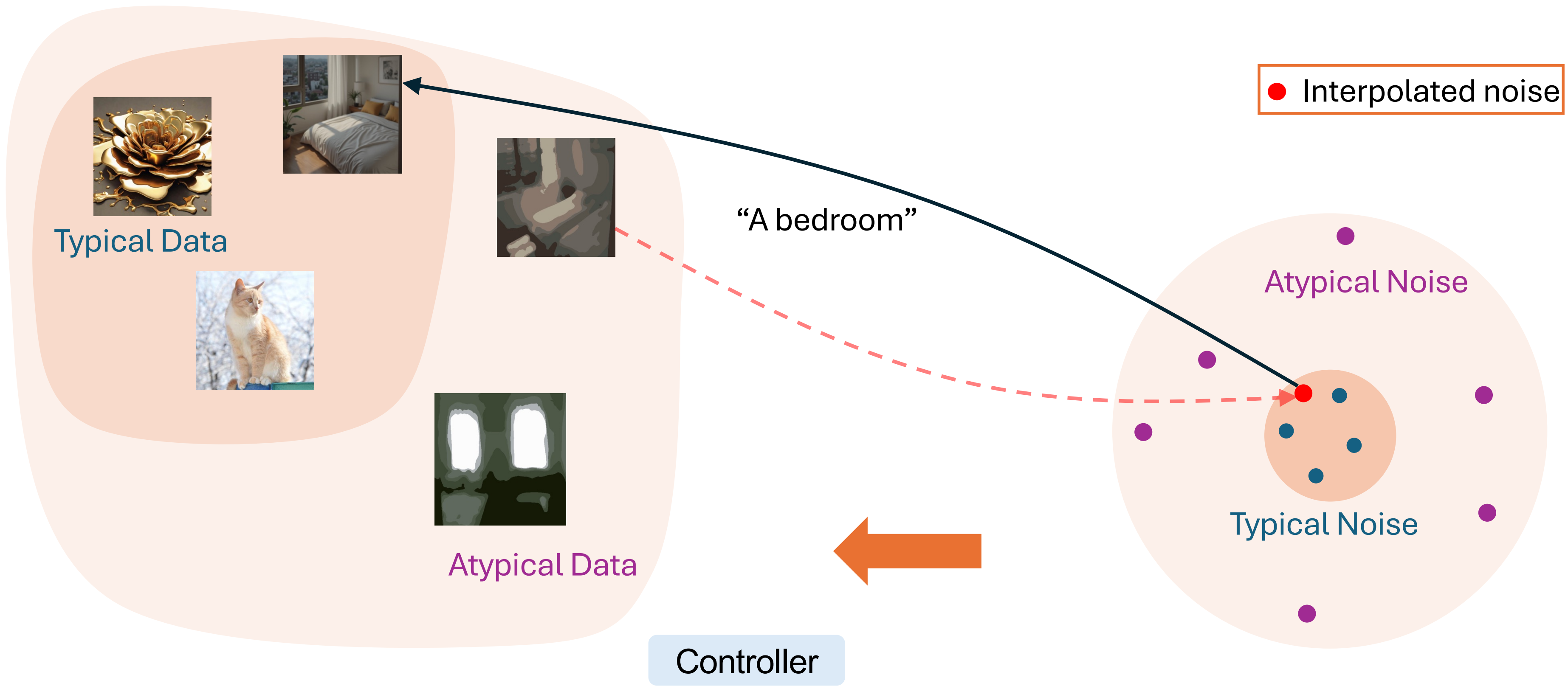


Inversion using Optimally Controlled Rectified Flow



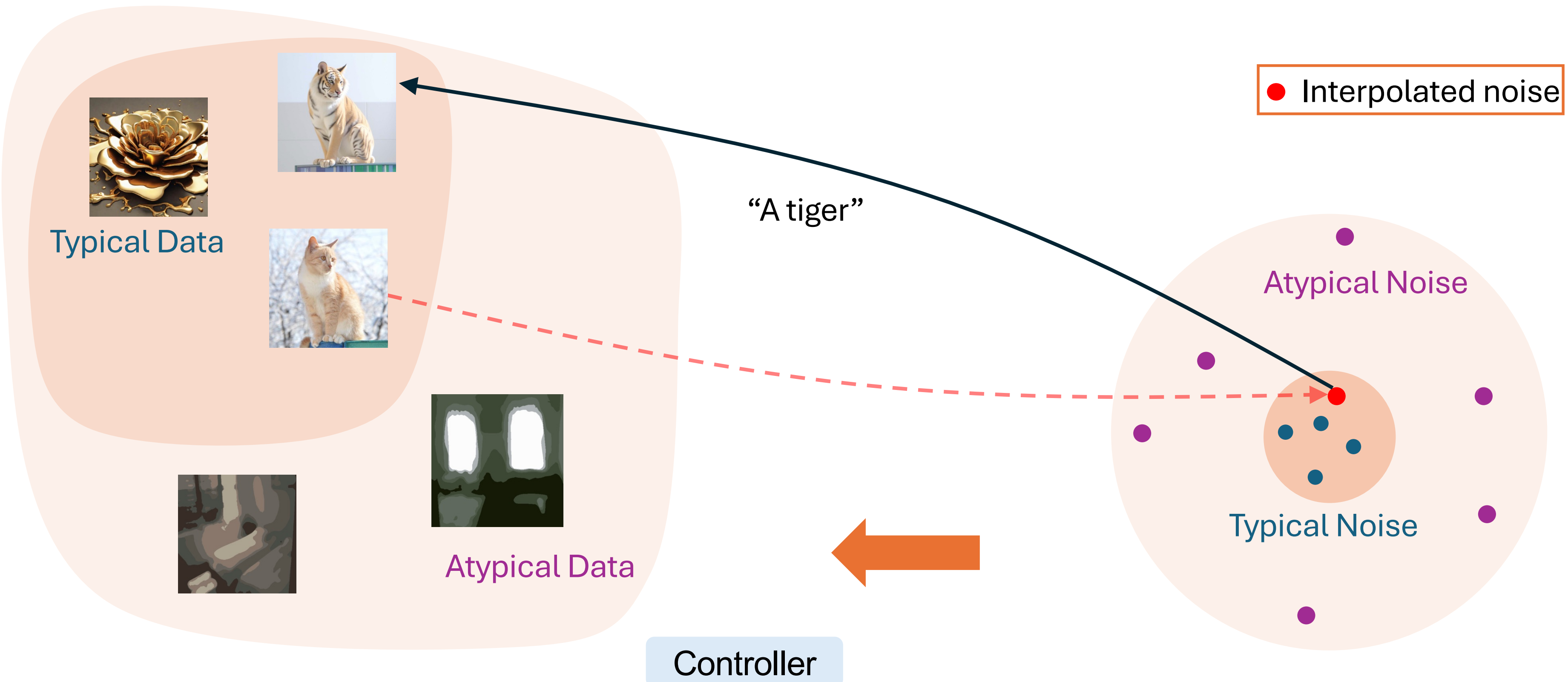
$$\mathrm{d}Y_t = \Big[u_t(Y_t) + \gamma\left(u_t(Y_t|\mathrm{y}_1) - u_t(Y_t)\right)\Big]\mathrm{d}t, \quad Y_0 = \mathrm{y}_0$$

Generation using Optimally Controlled Rectified Flows



$$\mathrm{d}X_t = \Big[v_t(X_t, \mathrm{prompt}) + \eta\,\big(v_t(X_t|\mathrm{y}_0) - v_t(X_t, \mathrm{prompt})\big)\Big]\mathrm{d}t, \quad X_0 = \mathrm{y}_1$$

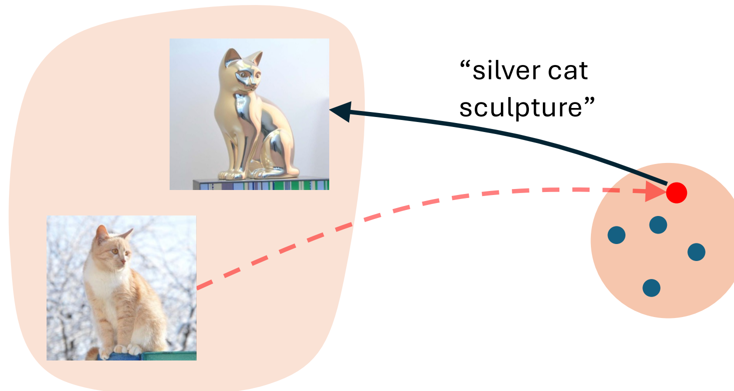
Generation using Optimally Controlled Rectified Flows



$$\mathrm{d}X_t = \Big[v_t(X_t, \mathrm{prompt}) + \eta \left(v_t(X_t | \mathrm{y}_0) - v_t(X_t, \mathrm{prompt}) \right) \Big] \mathrm{d}t, \quad X_0 = \mathrm{y}_1$$

Counterfactual Sampling (1/2)

- The counterfactual question: "Imagine if this cat was a silver sculpture"
- Three step approach for counterfactual reasoning with an SCM (Pearl et. al. 2016)
  - Noise abduction
  - Action ('do')
  - Prediction
- RF-Inversion intuition
  - Going back to noise through Inversion ⇔ Noise abduction
  - Doing through text conditioning ⇔ Action
  - Generating through reverse controlled flow ⇔ Prediction



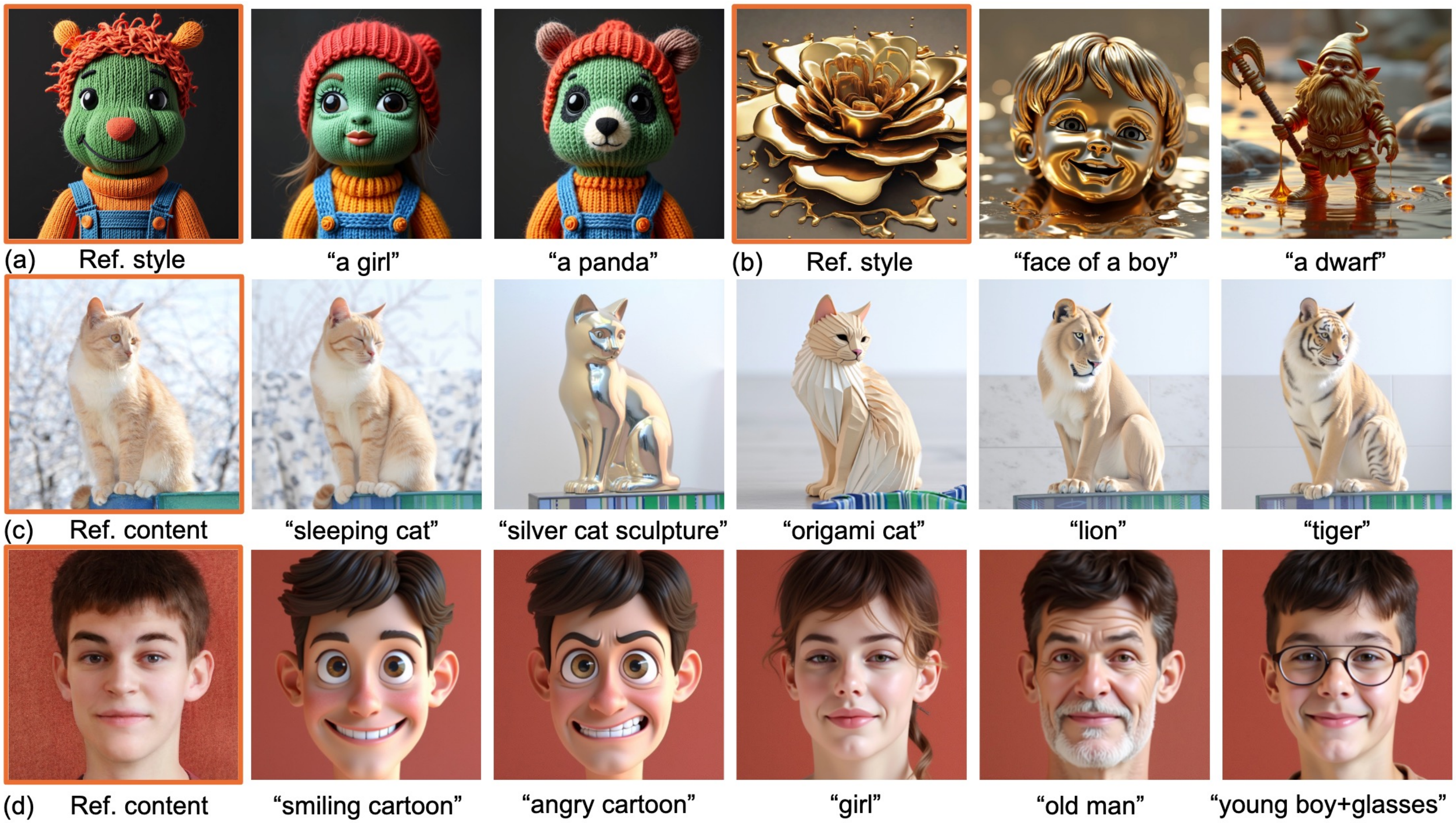
RF-Inversion interpreted as a prototype of a counterfactual sampler

Counterfactual Sampling (2/2)

Ref. Image | w/o controller + null-text (“”) | w/ controller + null-text (“”) | w/ controller + “silver sculpture” | w/ controller + “tiger”

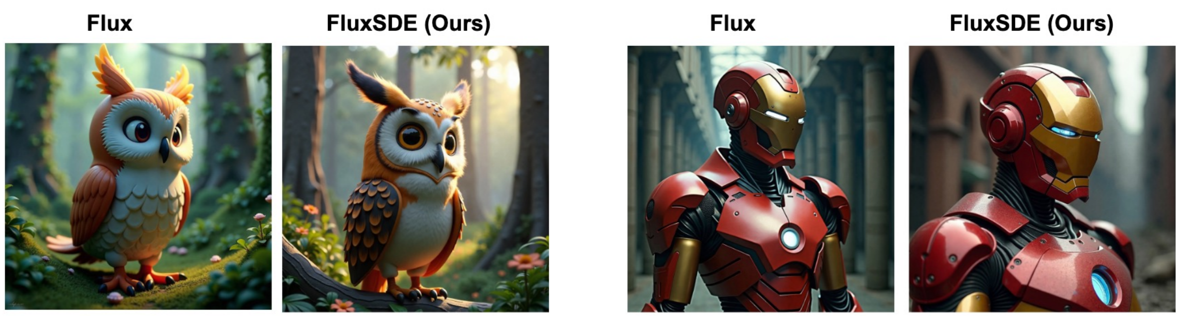
- Reference image
- (Reference image ➞ noise ➞ generated image) without our controller
- (Reference image ➞ noise ➞ generated image) with our controller
  - Conditional vector field is grounded to the reference image
- Using text prompts of: ‘A silver cat sculpture’ and ‘A tiger’

Image Inversion and Editing using Rectified Flows



A Stochastic Sampler for RF

- Benefits of a Stochastic Sampler for Rectified Flows
  - Many diffusion-based inversion and editing approaches rely on stochastic nature of the diffusion sampler
  - Higher-order solvers benefit from SDE interpretation of diffusion samplers
  - With finer discretization, SDE samplers outperform deterministic samplers in generative modeling, measured by Frechet Inception Distance (FID)
  - SDE samplers show robustness to corruption in the initial distribution, i.e., their invariant measure remains the same



Our Approach: Deterministic to Stochastic Sampler

- Closed-form expression for vector field with RF (using Tweedie's Formula):

$$u_t(\mathrm{y}_t) = \mathbb{E}_{(Y_0,Y_1)\sim p_1 \times p_0}\left[Y_1 - Y_0 | Y_t = \mathrm{y}_t\right] \; = \left[-\frac{1}{1-t}\mathrm{y}_t - \frac{t}{1-t}\nabla \log p_t(\mathrm{y}_t)\right]$$

- Closed form expression for optimal controller (using minimum principle):

$$u_t(\mathrm{y}_t | Y_1) = \frac{Y_1 - \mathrm{y}_t}{1-t}$$

- Interpolate between these drift fields to get structured noise:

$$\boxed{\mathrm{d}Y_t = u_t(Y_t) + \gamma(u_t(Y_1|\mathrm{y}_1) - u_t(Y_t))\mathrm{d}t, \quad Y_0 = \mathrm{y}_0, \quad t \in [0,1]}$$

Our Approach: Deterministic to Stochastic Sampler

- Controlled Rectified Flow ODE:

$$\mathrm{d}Y_t = u_t(Y_t) + \gamma(u_t(Y_1|\mathrm{y}_1) - u_t(Y_t))\mathrm{d}t, \quad Y_0 = \mathrm{y}_0, \quad t \in [0,1]$$

- Density evolution by continuity equation:

$$\frac{\partial p_t(Y_t)}{\partial t} = \nabla \cdot \left[\left(\frac{1}{1-t} \left(Y_t - \gamma \mathrm{y}_1 \right) + \frac{(1-\gamma)t}{1-t} \nabla \log p_t(Y_t) \right) p_t(Y_t) \right]$$

- Controlled SDE using Fokker-Planck equation:

$$\mathrm{d}Y_t = -\frac{1}{1-t} \left(Y_t - \gamma \mathrm{y}_1 \right) \mathrm{d}t + \sqrt{\frac{2(1-\gamma)t}{1-t}} \mathrm{d}W_t, \quad Y_0 \sim p_0$$

Analogous approach for deriving SDE for Generation

Experiments

Experiments: Identity Preservation in Face Editing

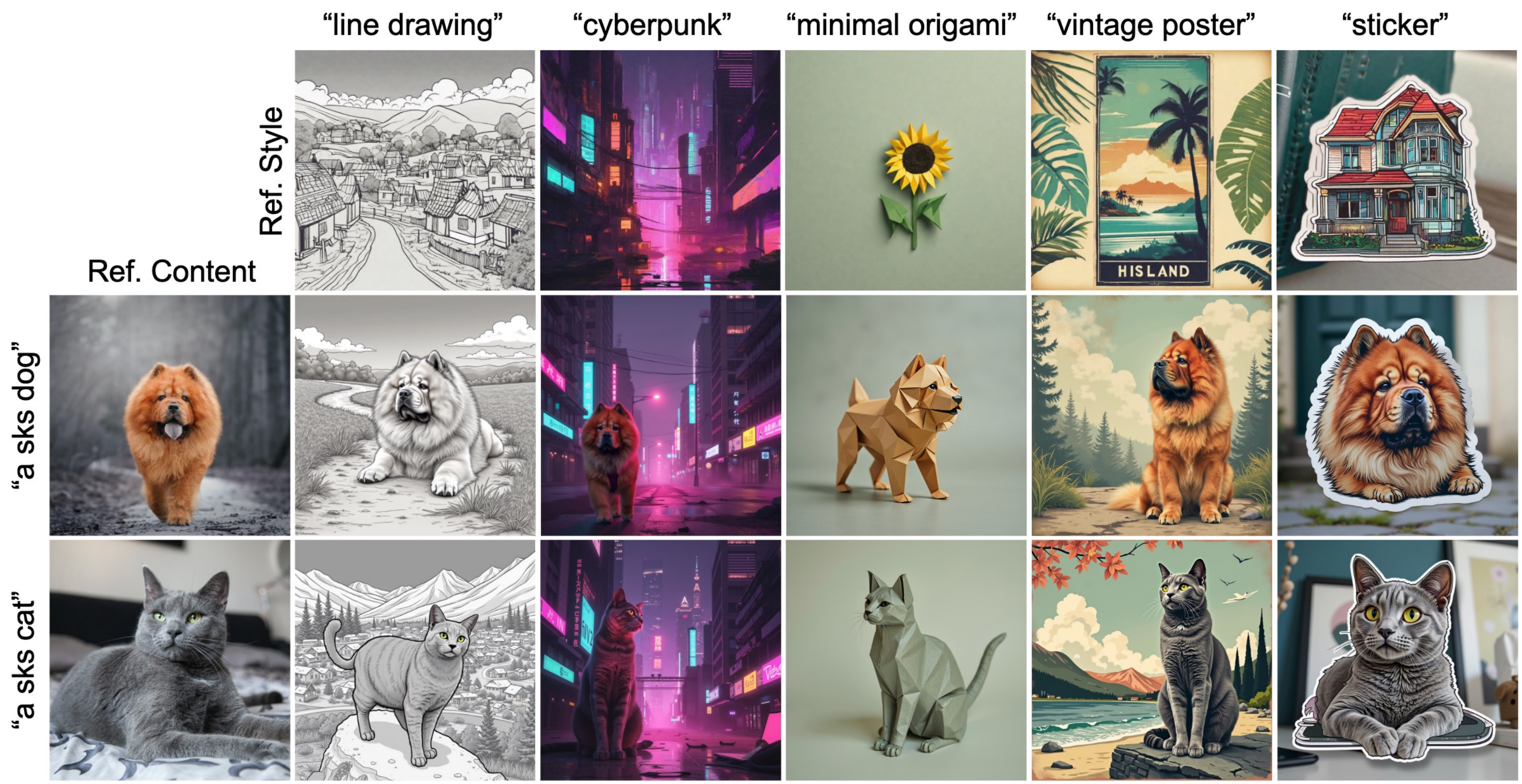
Original | SDEdit | DDIM Inversion | NTI | NTI+P2P | Ours

Prompt: “… + wearing glasses”

Experiments: Semantic Image Editing

Editing (a) stylized expression, (b) age, (c) gender, and (d) object insert

Experiments: Content-style composition



Experiments: Generalization to another flow model SD3.5

(a,b) Generated reference style (c,d) Hand drawn reference style

Please see: https://openreview.net/forum?id=bnINPG5A32 for reference image credits

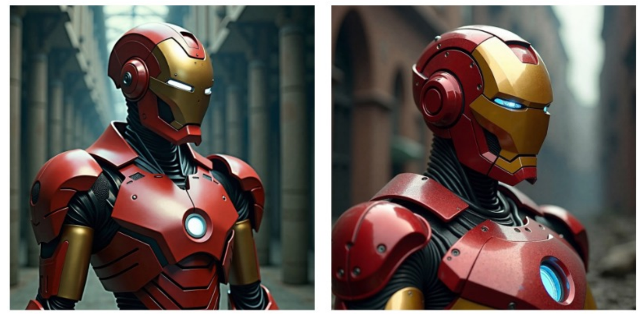
Experiments: Generative modeling using rectified flow SDE

Flux **FluxSDE (Ours)**

Prompt: “portrait, looking to one side of frame, lucid dream-like 3d model of an owl, video game character, forest, wonderland, photorealism, cinematic artistic style.”

Prompt: “a dragon soaring through the sky, battle ground, people fighting on the ground.”

Flux **FluxSDE (Ours)**



Prompt: “a robot with a reflective helmet, iron armor, photorealistic, in shades of red and golden brown, dark gloomy environment, epic scene.”

Prompt: “a genius scientist, in his 60s, standing, writing on the black board, white hair, white beard, round spectacles.”

Community Developments

Day 1
(Oct 14)

- Paper released on ArXiv: https://arxiv.org/pdf/2410.10792
- Project page: https://rf-inversion.github.io/
- ComfyUI code reproduced results from RF-Inversion (<24hrs)
  https://tinyurl.com/xwv24wbp

Week 1-2

- 8 Steps Style and Face Transfer with Unsampling and RF Inversion
- YouTube Tutorial: https://www.youtube.com/watch?v=H_G2AaLWN2o
- Endless creative possibilities: https://tinyurl.com/57b72ks4
- Test of RF-Inversion on style transfer: https://tinyurl.com/bdhs4vy3
- Podcasts: https://tinyurl.com/3x496jkv, https://tinyurl.com/3djmevef

Week 3-4

- Animate movies using RF-Inversion: https://tinyurl.com/xwv24wbp
- Mochi Video Editing with RF-Inversion: https://tinyurl.com/yeyej7x8
- Integration in diffusers from HuggingFace: https://tinyurl.com/2avrfzh5
- Follow up works: ReCapture (Zhang et al.), RF-Edit (Wang et al.), AnimateAnything (Lei et al.), EditAway (wang et al.), MyTimeMachine (Qi et al.), HeadRouter (Xu et al.)

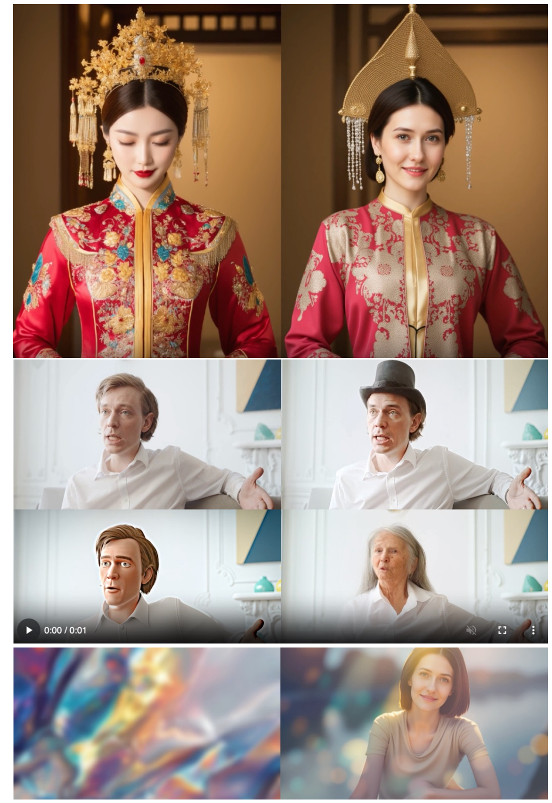


Image Credit: https://tinyurl.com/57b72ks4, https://tinyurl.com/xwv24wbp

Summary

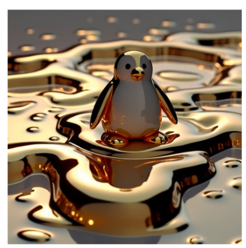
- First efficient inversion and editing for rectified flows
  - Interpolates two vector fields

- Stochastic equivalence between rectified flow ODE and SDE

- State-of-the-art zero-shot performance w/o training, optimization, prompt tuning and complex attention processors

- Effectiveness in stroke-to-image synthesis, face editing, stylization, content-style composition, w/ large-scale human evaluations

“a butterfly”



“a baby penguin”

“a boat”

“a piano”