Diffusion Models

Flow Matching Perspective

CS 280 2025

Angjoo Kanazawa, co-designed with
Songwei Ge, David McAllister

Thanks to Yaron Limpan and co + Steve Seitz for great slides!

Logistics

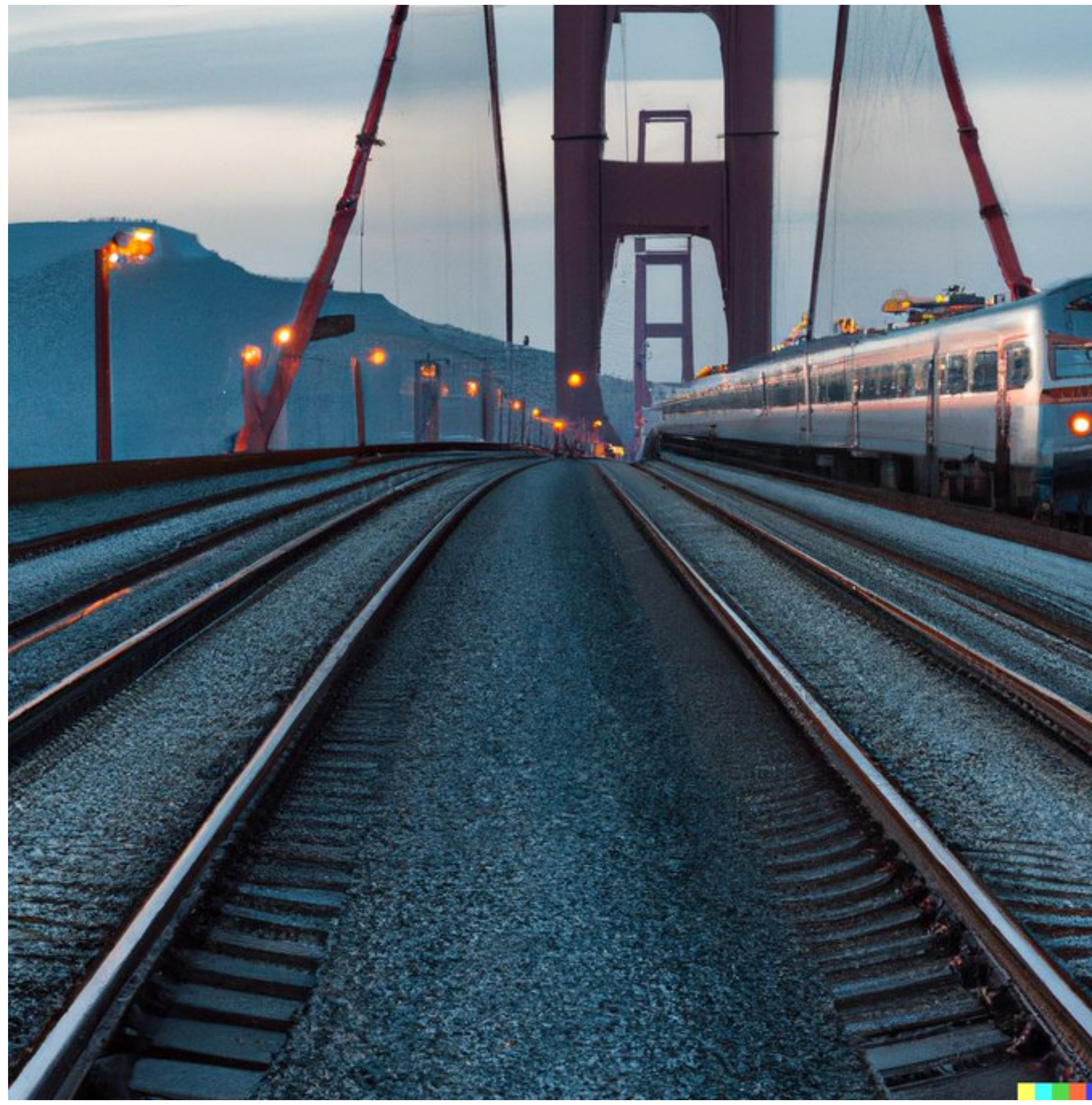
- Homework to be released today
- Today is by me
- Wednesday guest lecture by Songwei Ge and David McAllister!

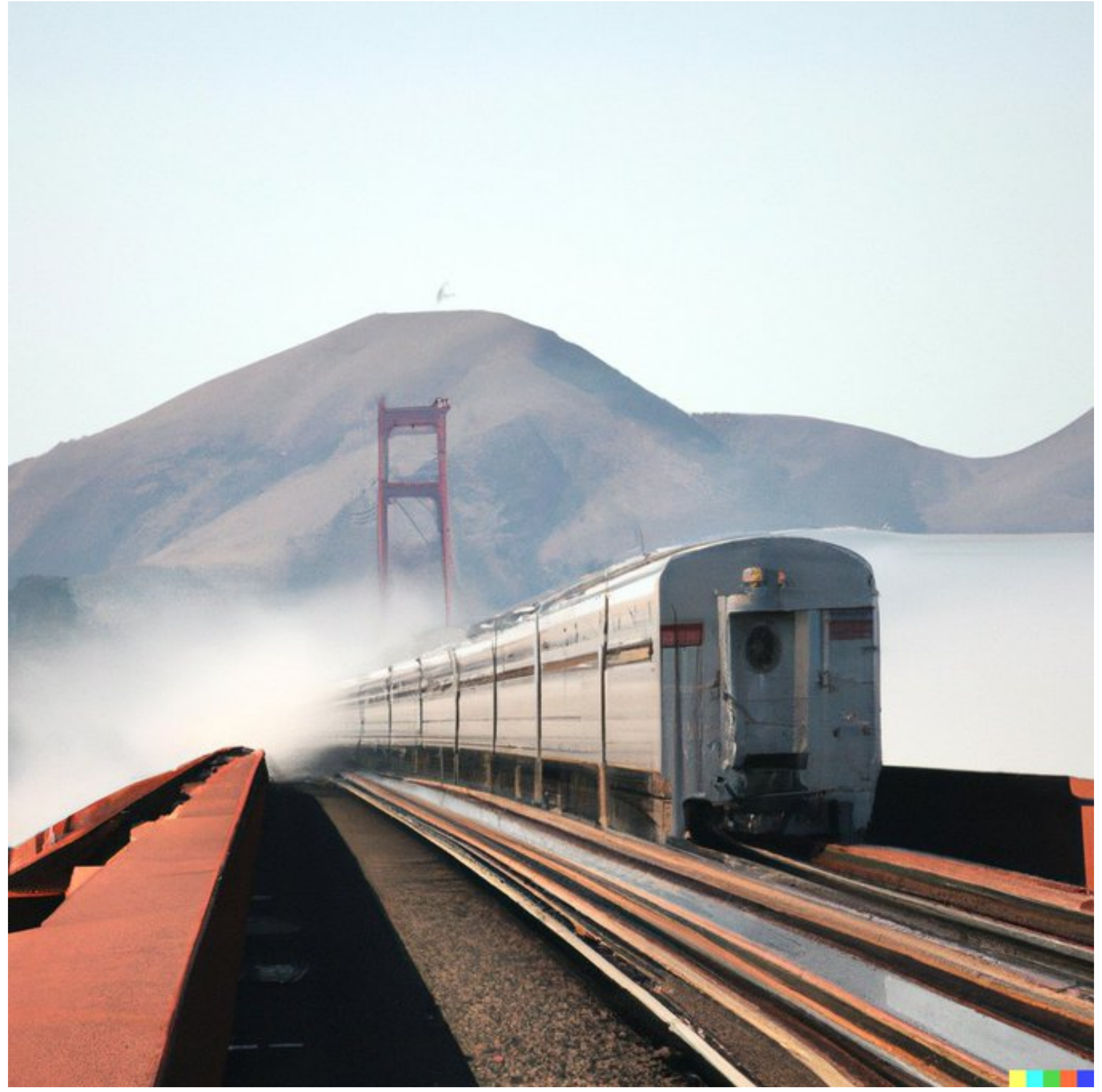
Sora, Open AI February 15, 2024

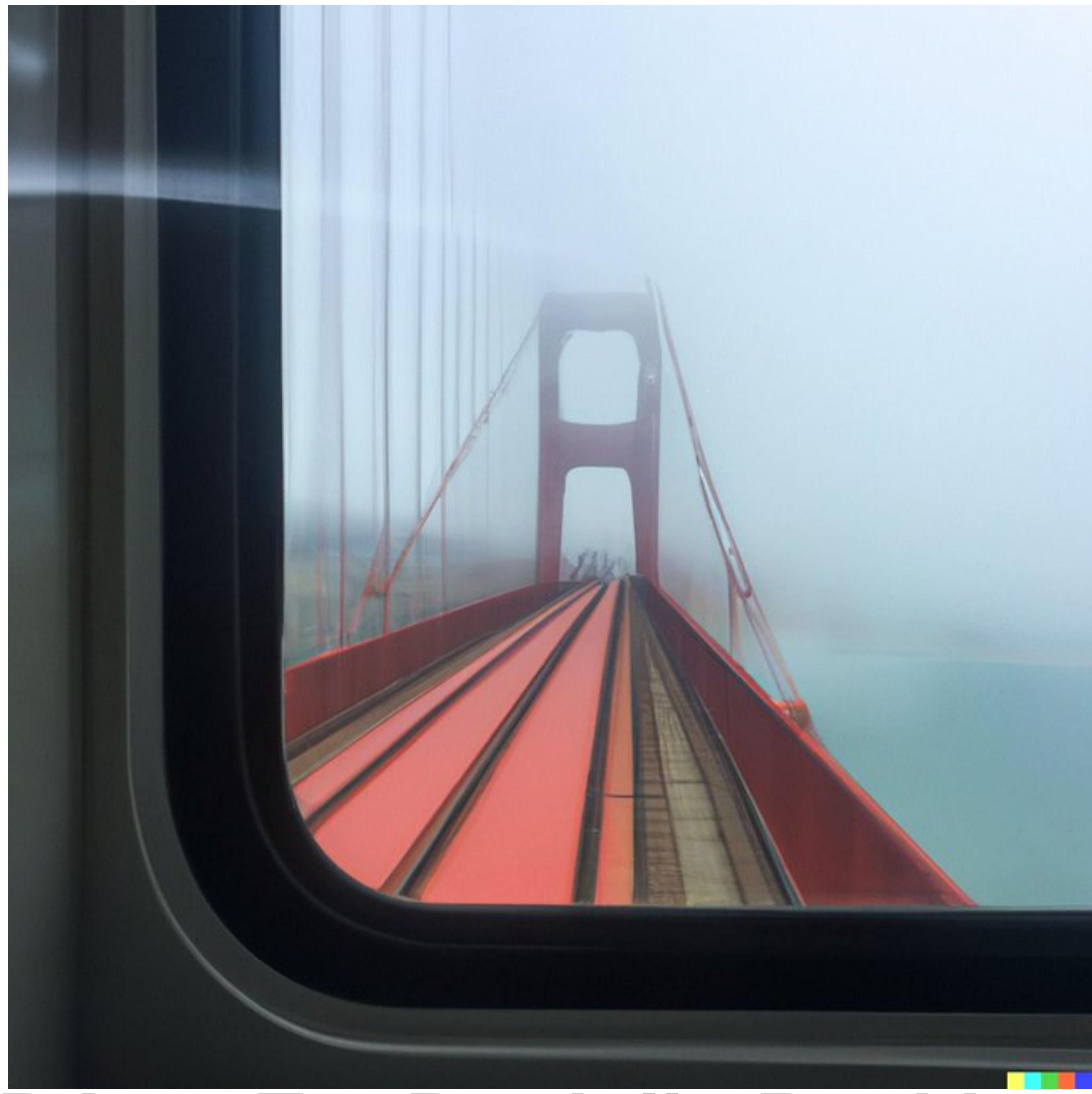
An astronaut riding a horse in a photorealistic style (Dall-E 2)

slide from Steve Seitz's video

Impressive compositionality:







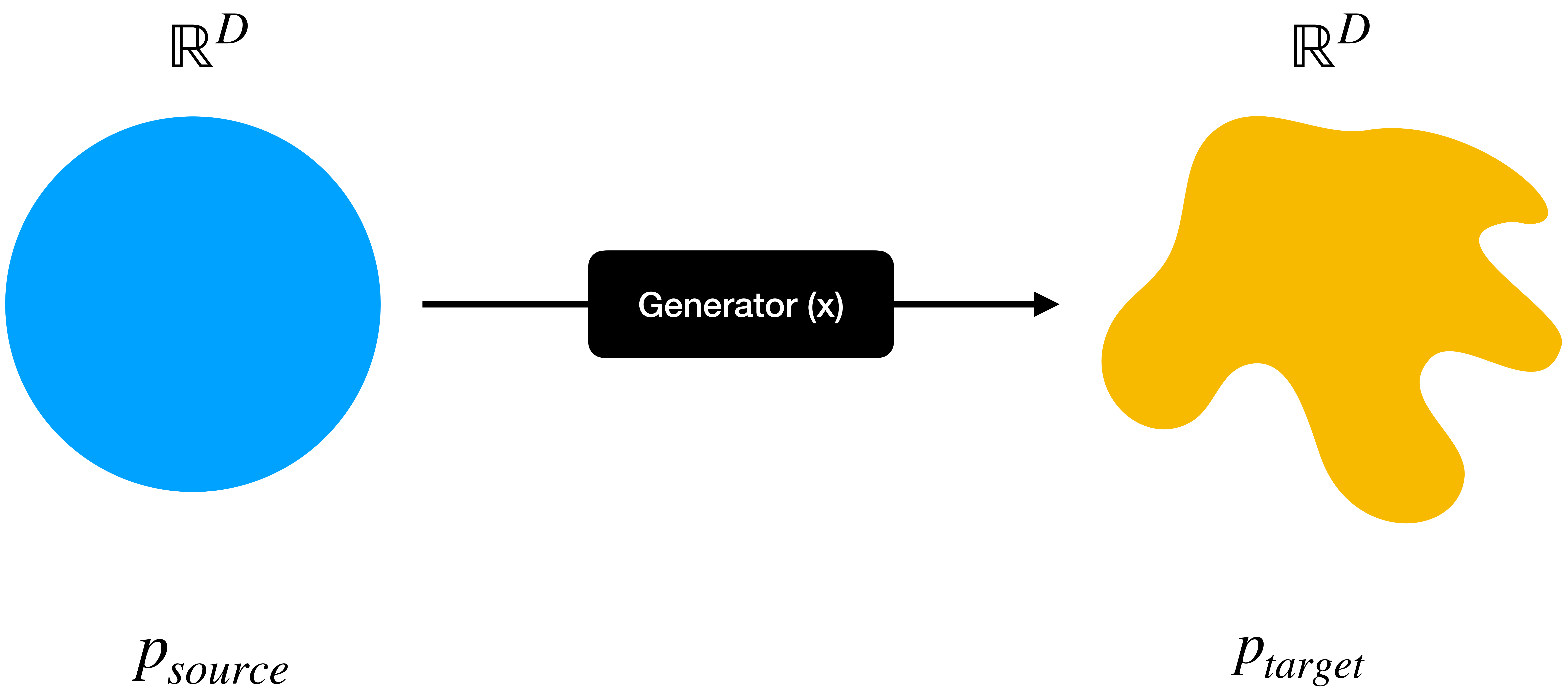
DALL-E + Danielle Baskin

Generative Models

Goal: Modeling the space of Natural Images

- Want to estimate P(x) the probability distribution of natural images
- Why? Many reasons

The generative story



Generative Story

- Any Generative Model can be described with the process of sampling an image
- For ex, here's the generative story for PCA in its probabilistic interpretation:

1. Sample from a Gaussian Distribution

$$z \sim N(0,I)$$

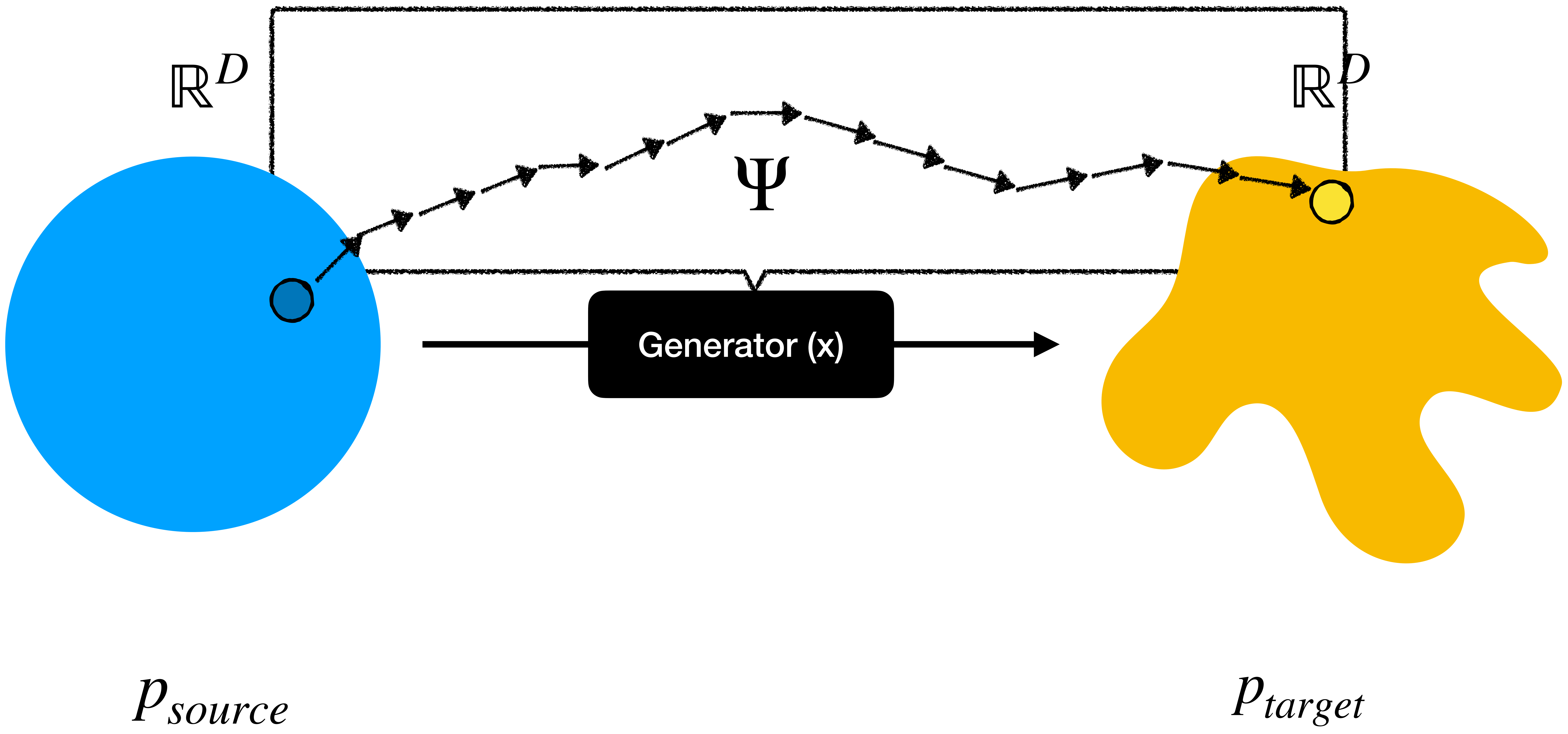
2. Project to Images (W = Eigenvectors, Mu = avg datapoint)

$$x = Wz + \mu$$

Generative Models

- Many methods:
  - Parametric Distribution Estimation (e.g. GMM, PCA)
  - Autoregressive models (e.g. PixelCNN, GPT)
  - Latent space mapping (e.g. VAE, GANs)
  - **Flow based models (e.g. Diffusion, Normalized Flow, Flow Matching)**

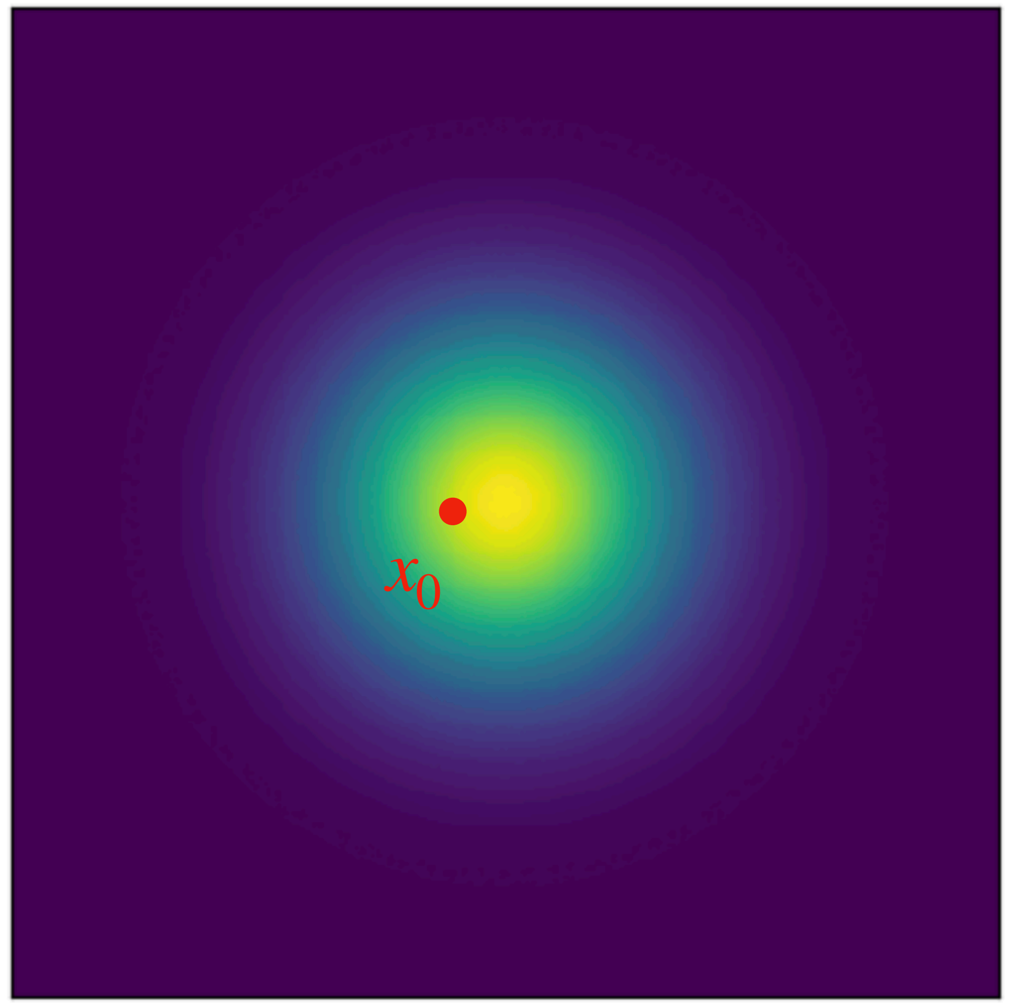
Flow based Generative Models



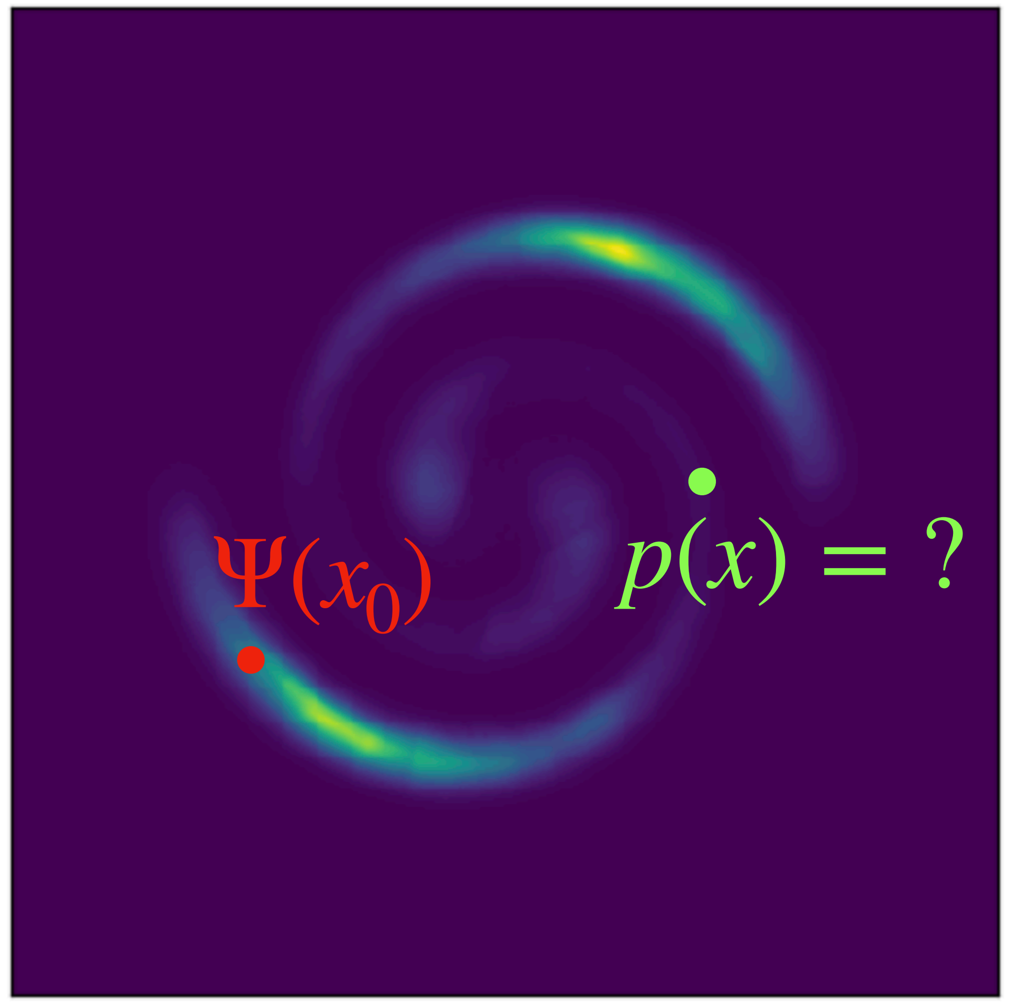
Generative models

$p(x)$ $q(x)$

$\mathbb{R}^d$



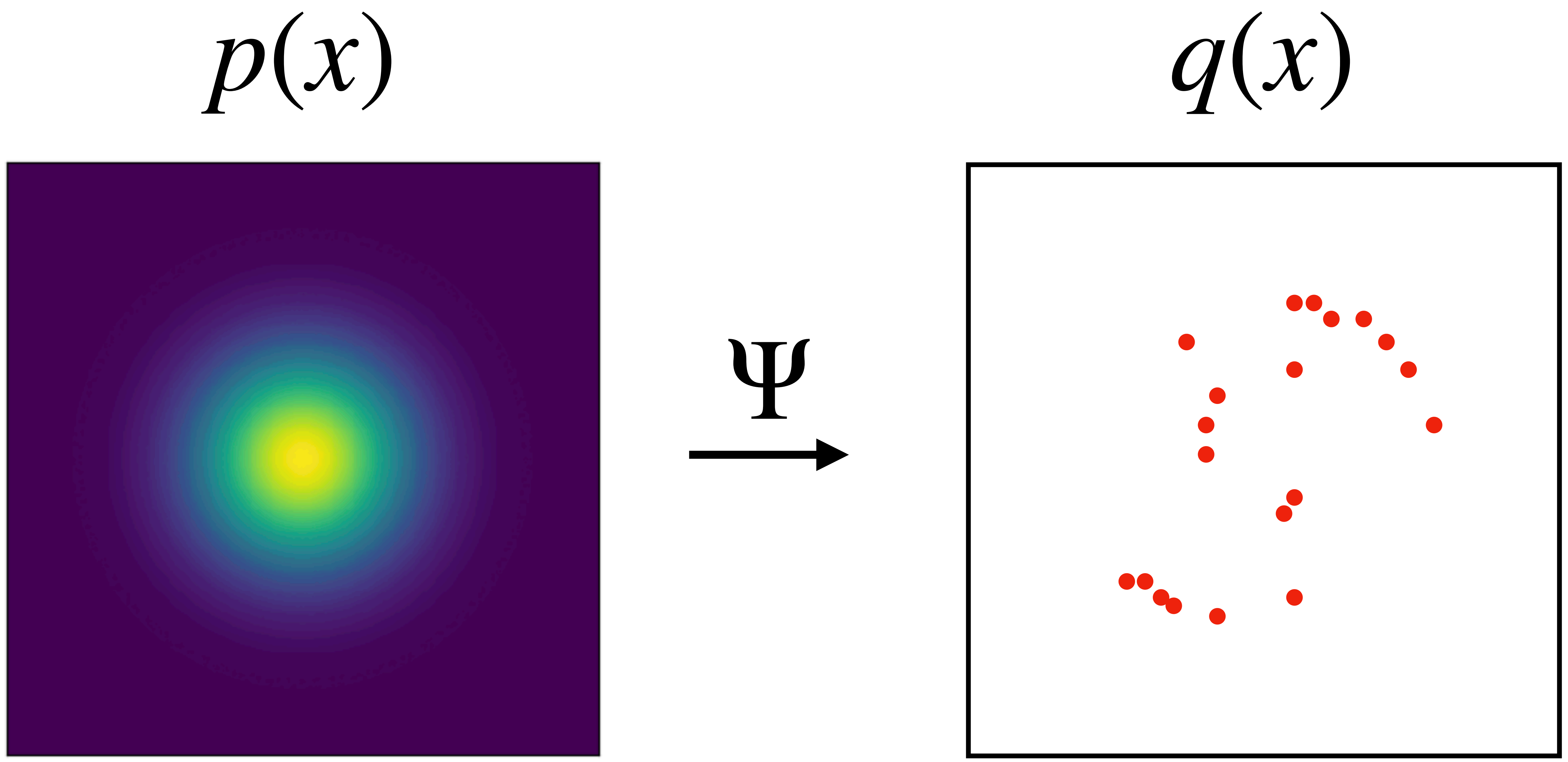
$\xrightarrow{\Psi}$



$x_0 \sim p$ $\Psi(x_0) \sim q$

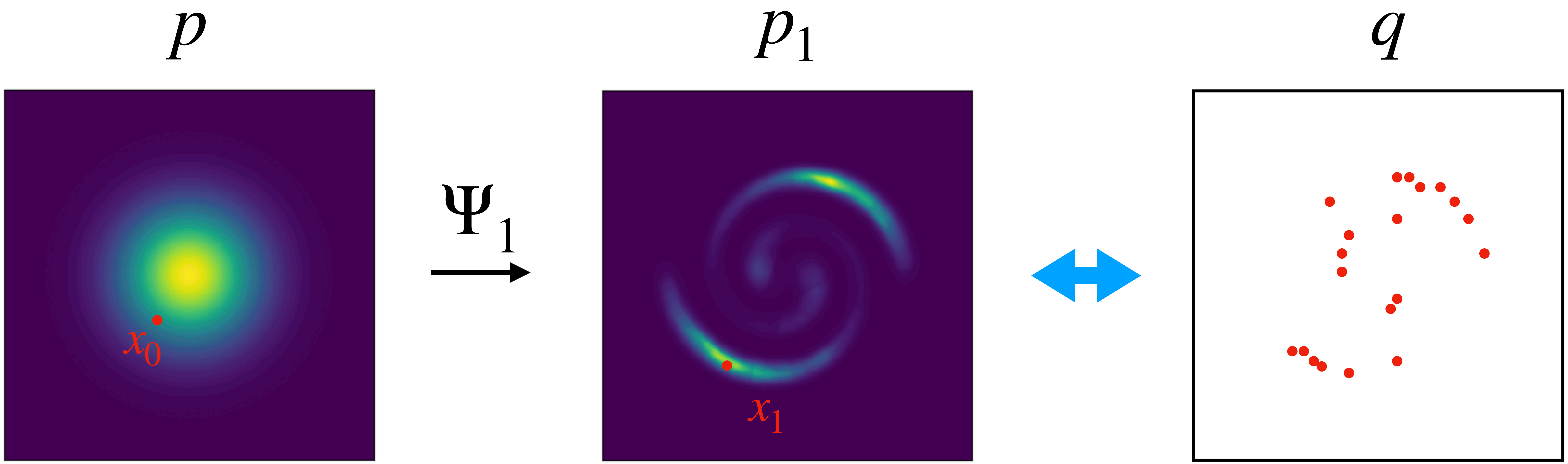
Slides from Yaron Lipman

Generative models



Slides from Yaron Lipman

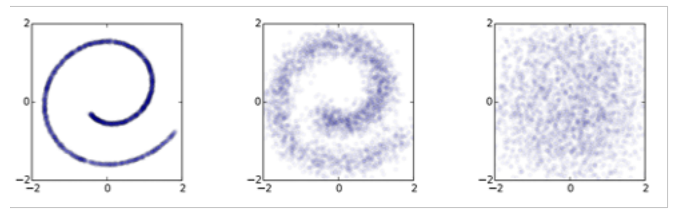
Flows as Generative Models



Slides from Yaron Lipman

[Chen et al. 2018]

History



Sohl-Dickstein et al. 2015
Deep unsupervised learning using non equilibrium thermodynamics

NICE Dinh et al.
Normalizing Flows 2015

RealNVP,
Dinh et al.
2017

Glow,
Kingma &
Dhaliwal
2018

Neural ODE
Chen et al.
2018…

DALL-E1 Open AI 2020

DDPM, Ho et al. 2020

Song et al. Score-based
Generative Models, DDIM
2021

Rectified Flow, Liu et al. 2022

**Flow Matching,
Lipman et al. 2022**

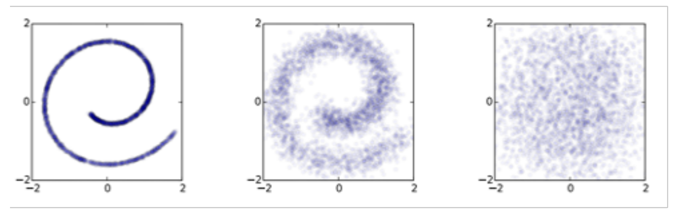
DALL-E2 Open AI 2023
StableDiffusion, Stability 2023

Flow Matching Tutorial
NeurIPS 2024

MovieGen late
2024~

History

Diffusion Arc



Sohl-Dickstein et al. 2015
Deep unsupervised learning using non equilibrium thermodynamics

DALL-E1 Open AI 2020

DDPM, Ho et al. 2020

Song et al. Score-based Generative Models, DDIM 2021

DALL-E2 Open AI 2023
StableDiffusion, Stability 2023

Rectified Flow, Liu et al. 2022

NICE Dinh et al. Normalizing Flows 2015

RealNVP, Dinh et al. 2017

Glow, Kingma & Dhaliwal 2018

Neural ODE Chen et al. 2018…

Normalizing Flow Arc

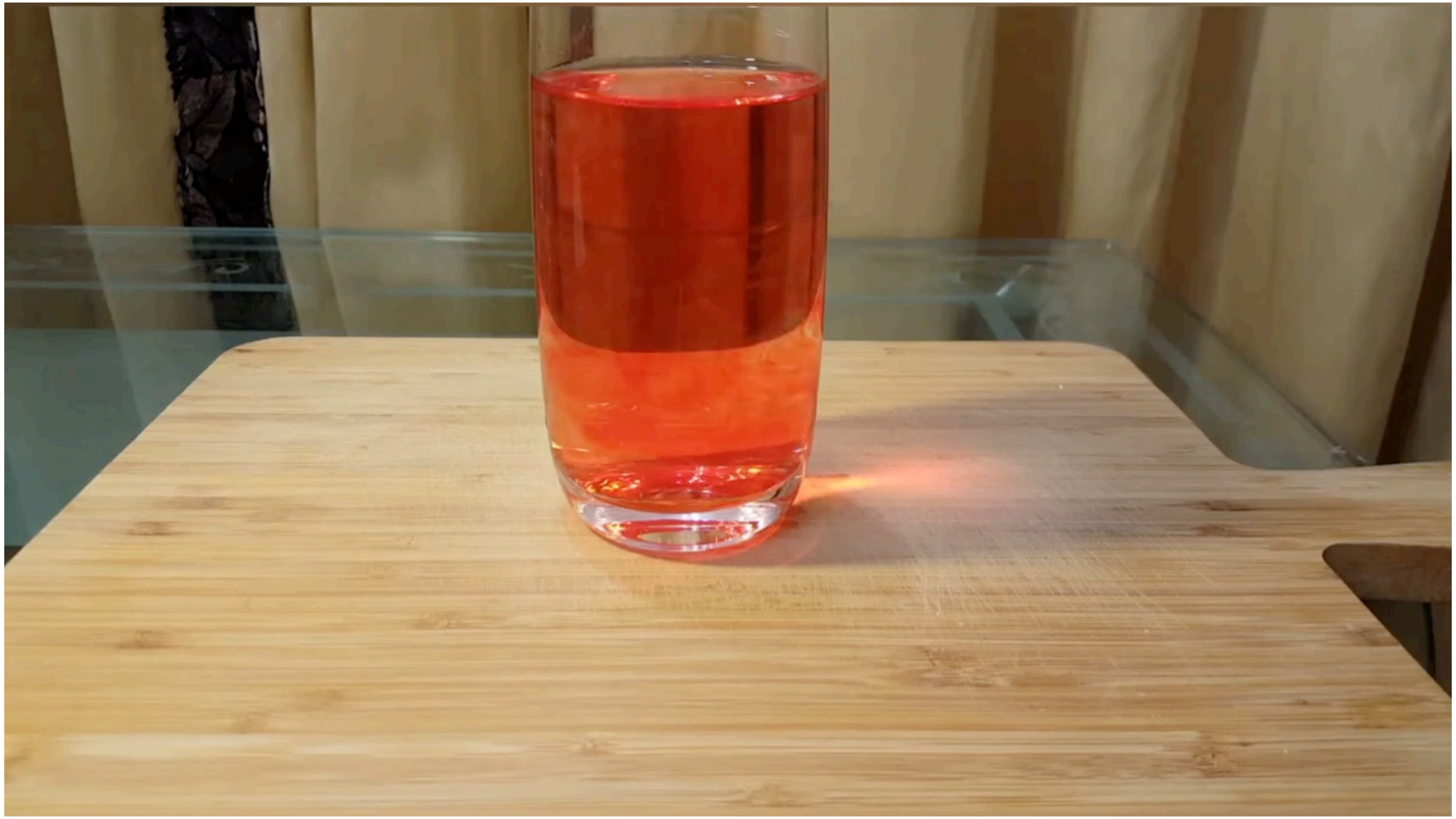
Flow Matching, Lipman et al. 2022

Unification / Simplification

Flow Matching Tutorial NeurIPS 2024

MovieGen late 2024~

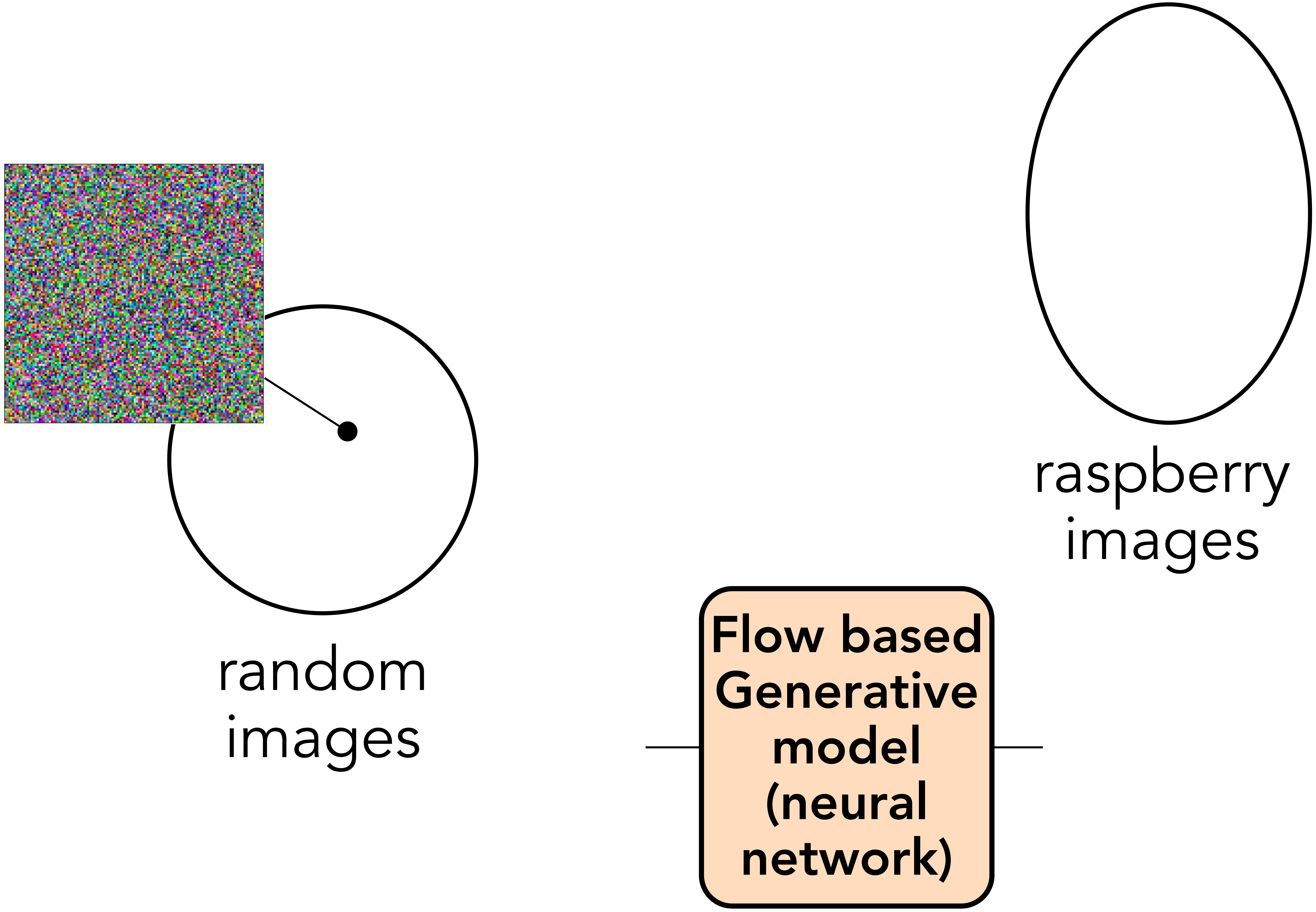
Diffusion: Physics Interpretation



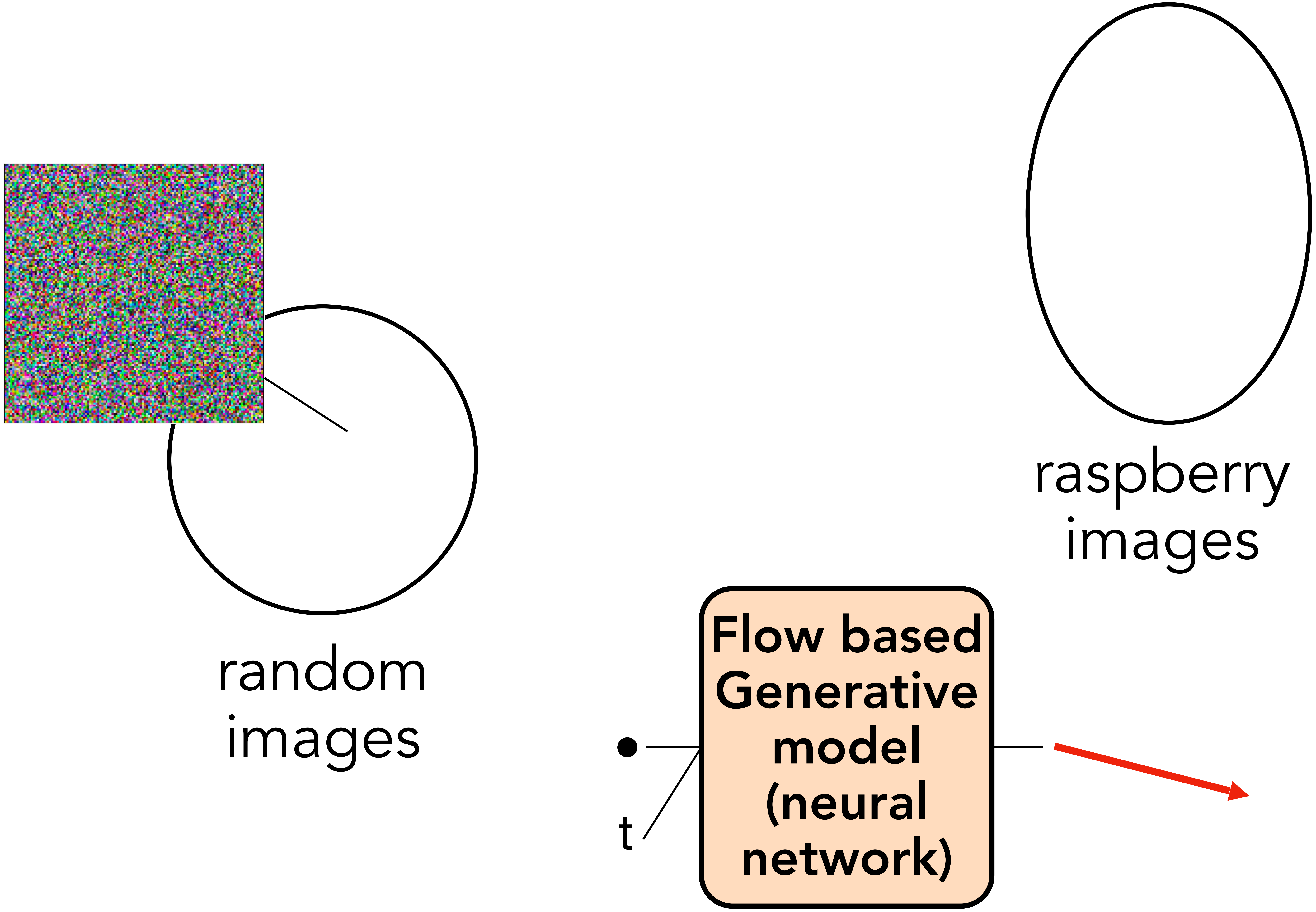
Idea in Song et al. Score-Based Generative Modeling through Stochastic Differential Equations 2021

First, the intuition

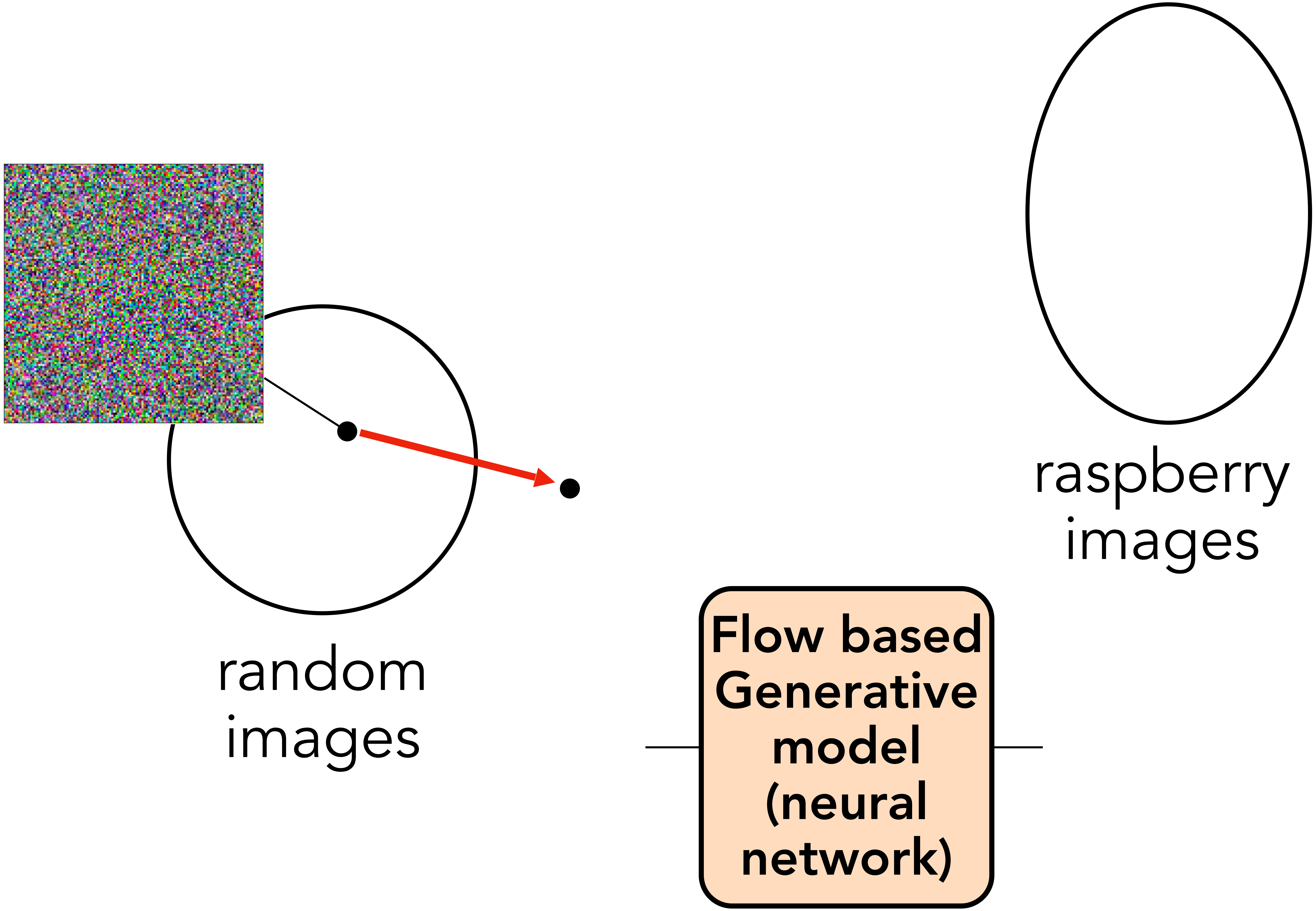
Inference



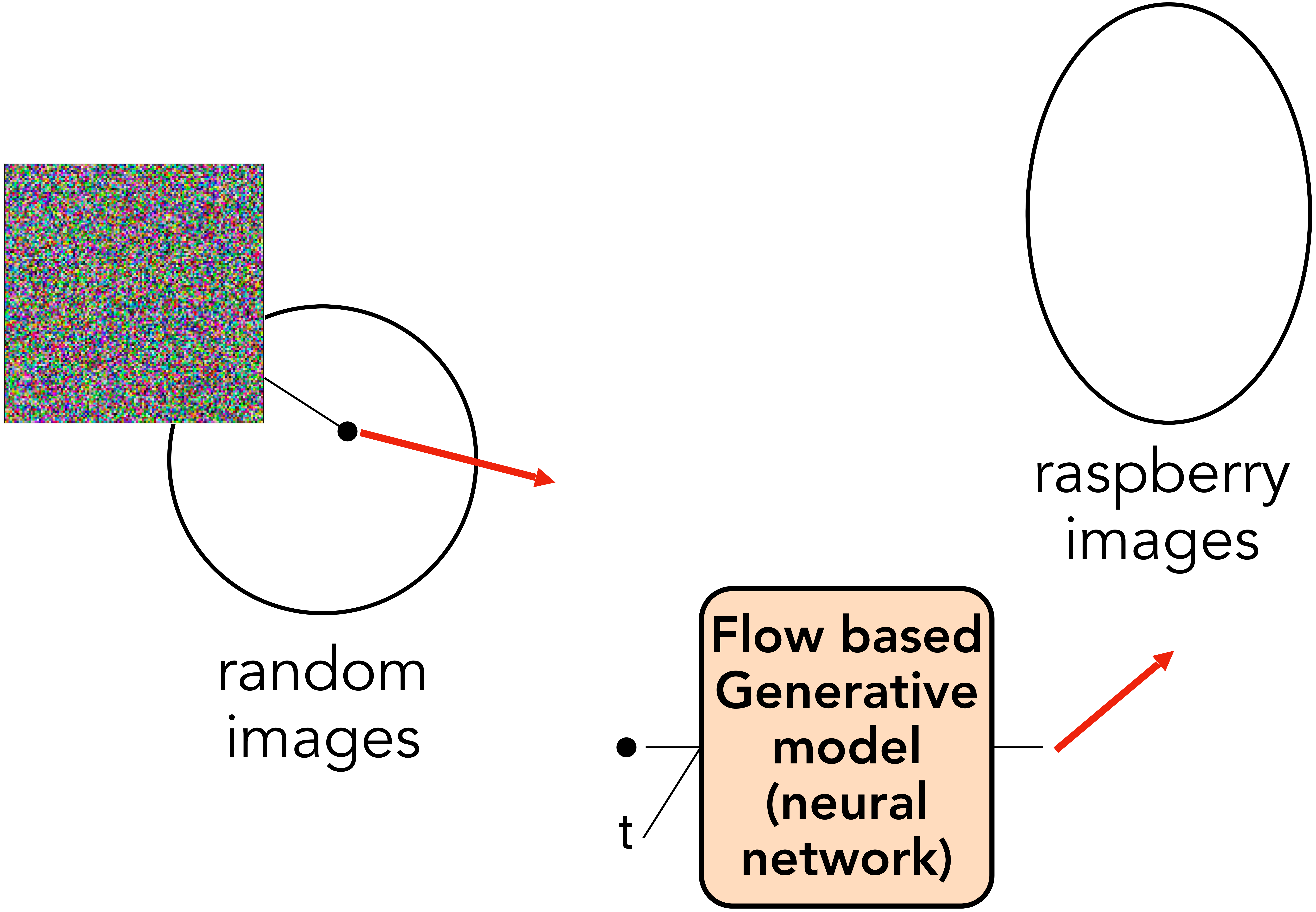
slide from Steve Seitz's video



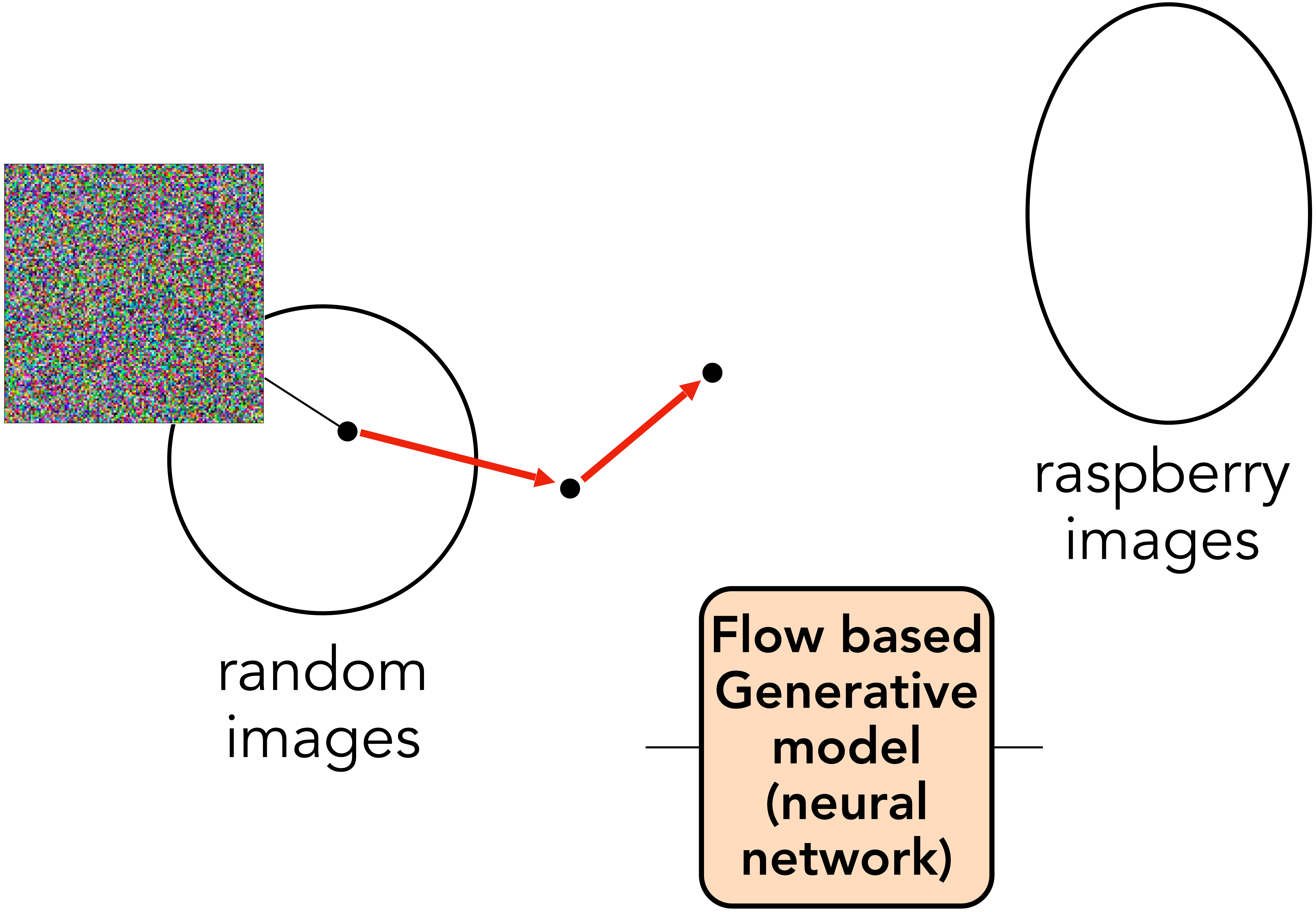
slide from Steve Seitz's video



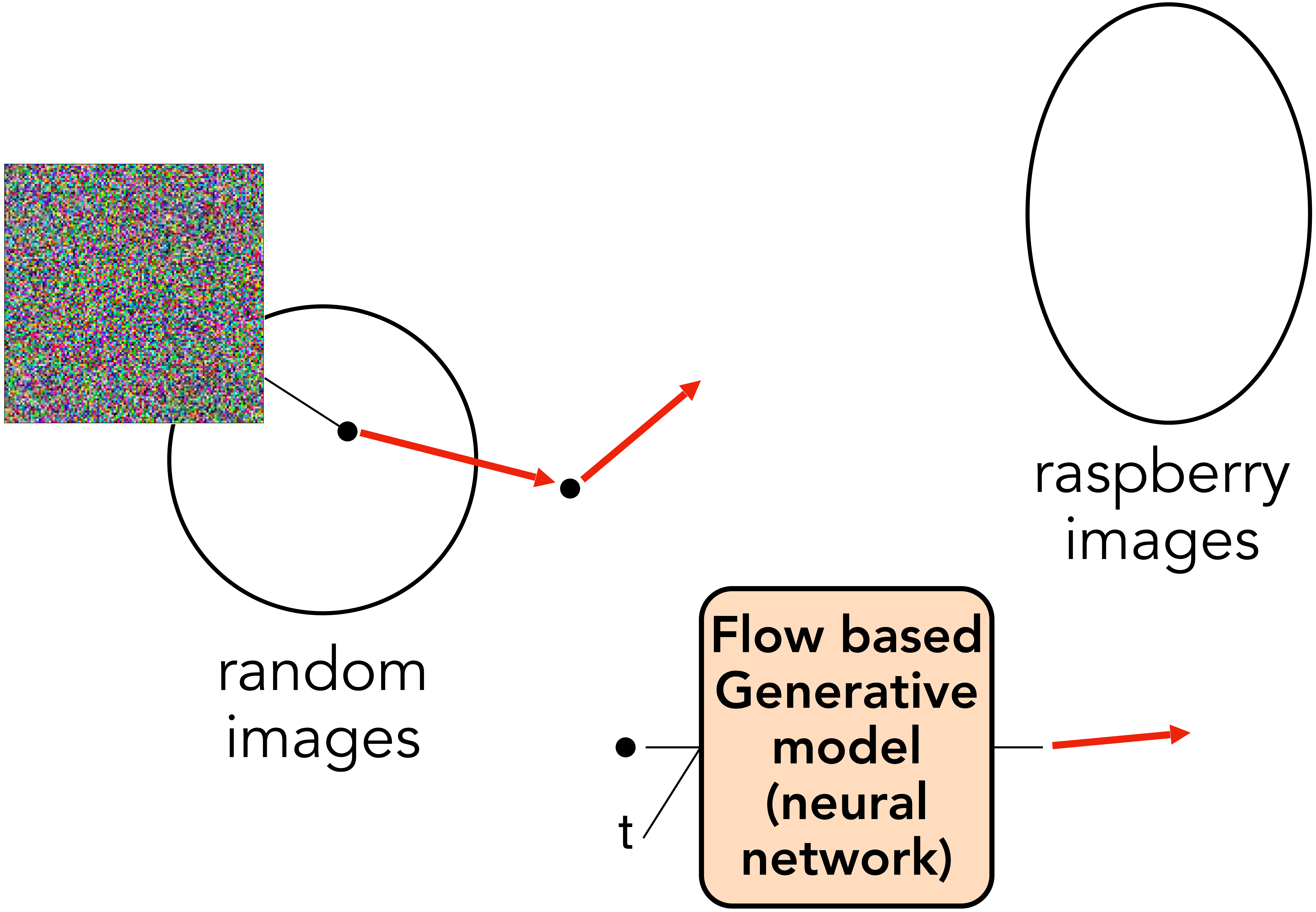
slide from Steve Seitz's video



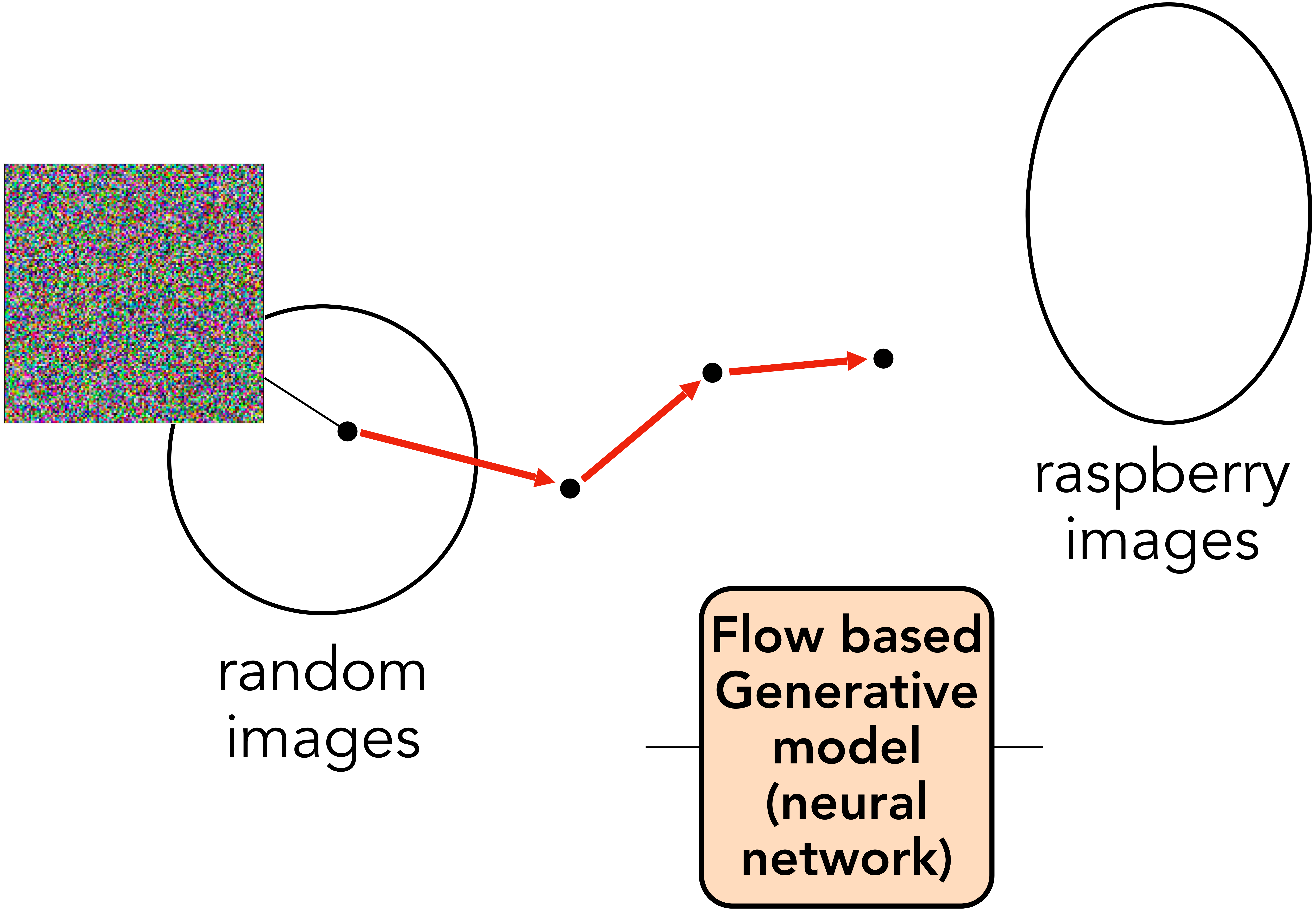
slide from Steve Seitz's video



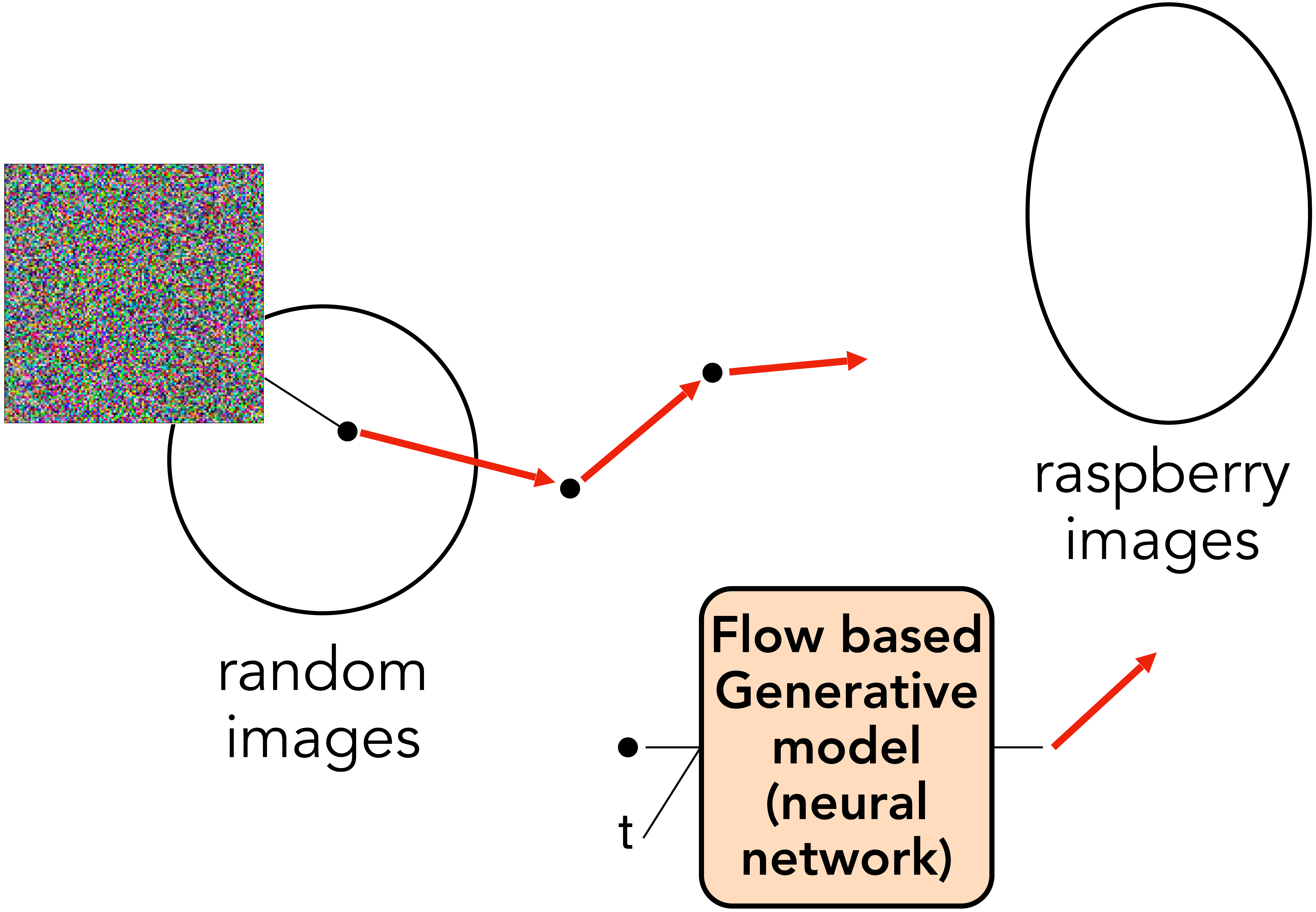
slide from Steve Seitz's video



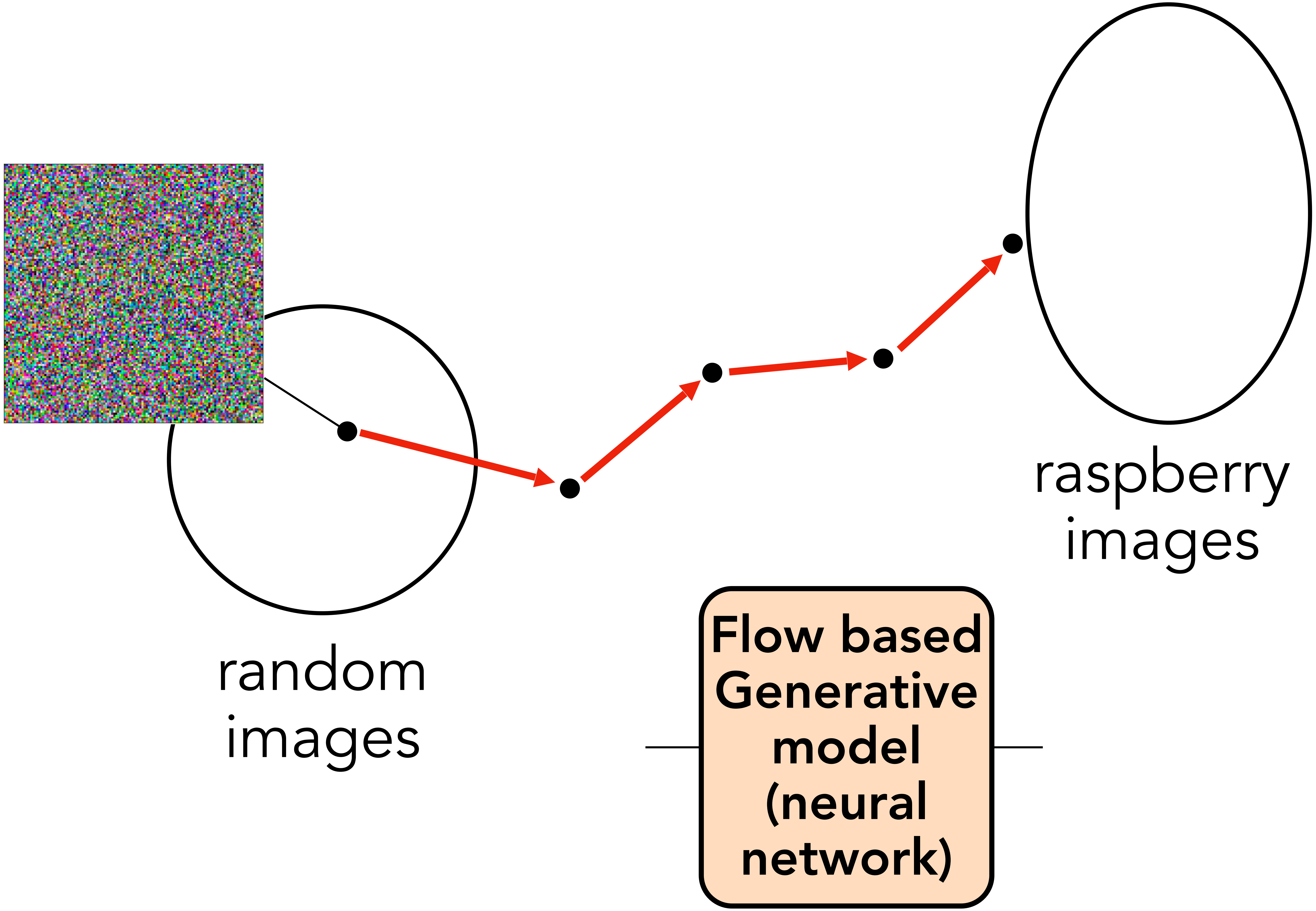
slide from Steve Seitz's video



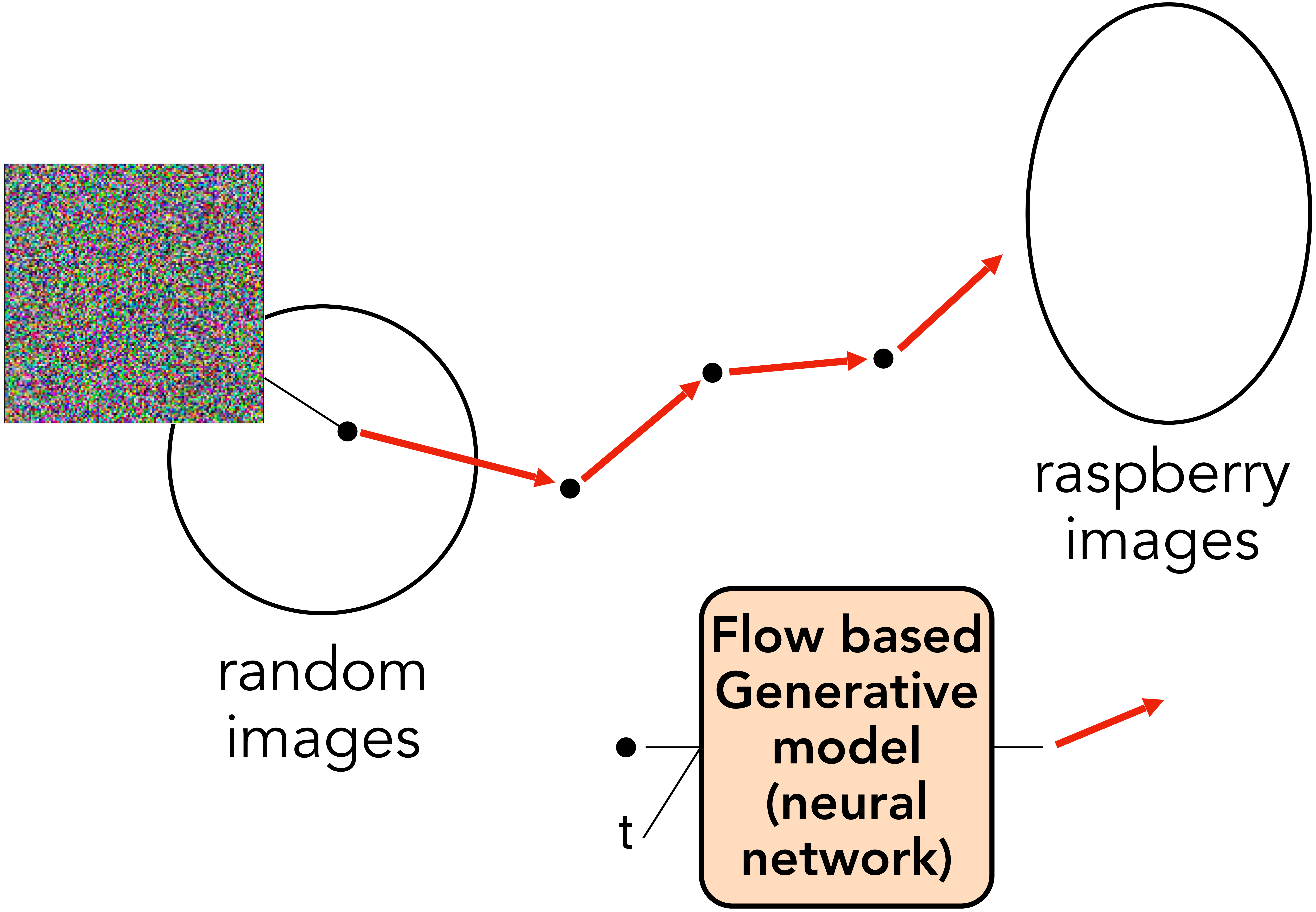
slide from Steve Seitz's video



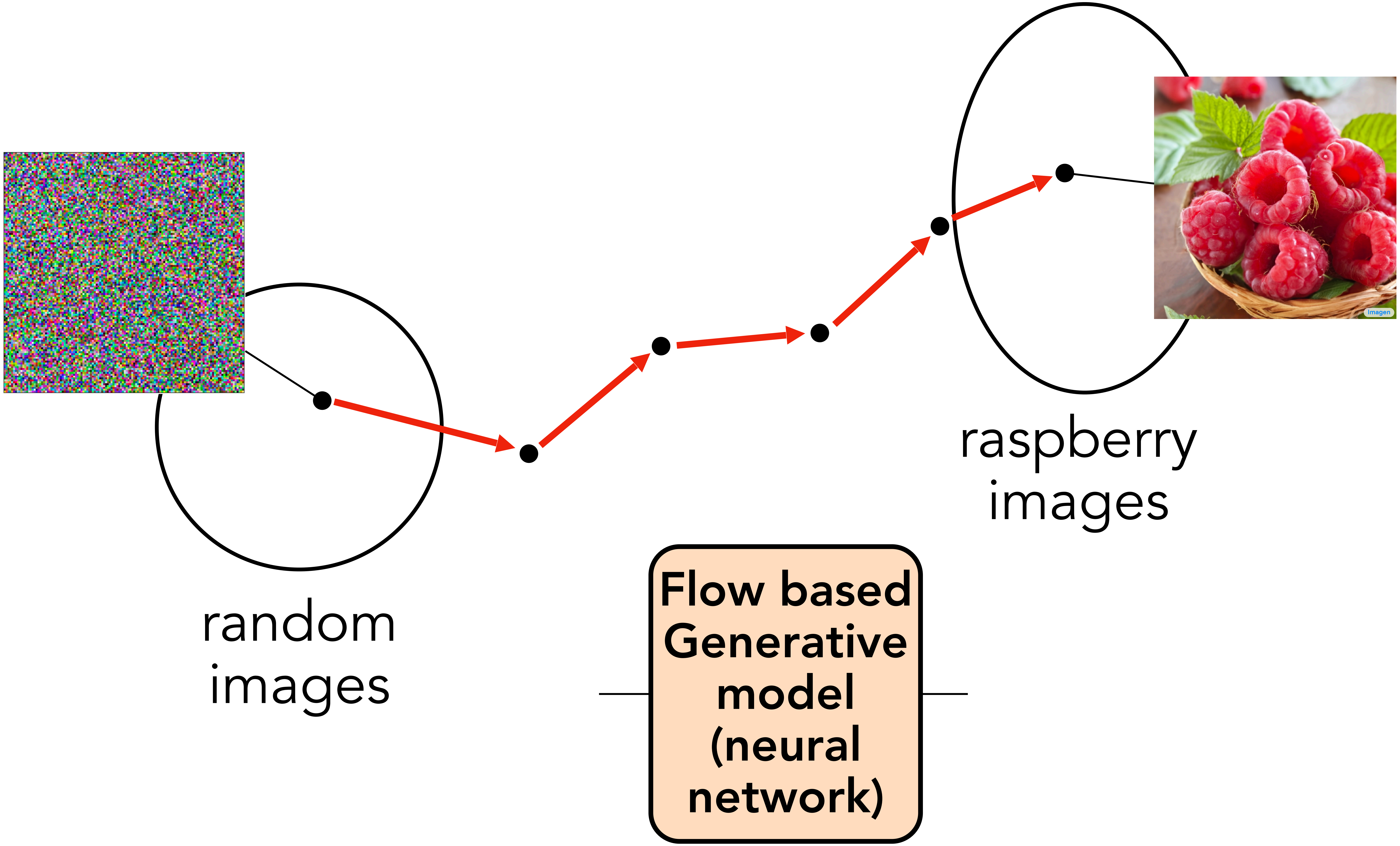
slide from Steve Seitz's [video]



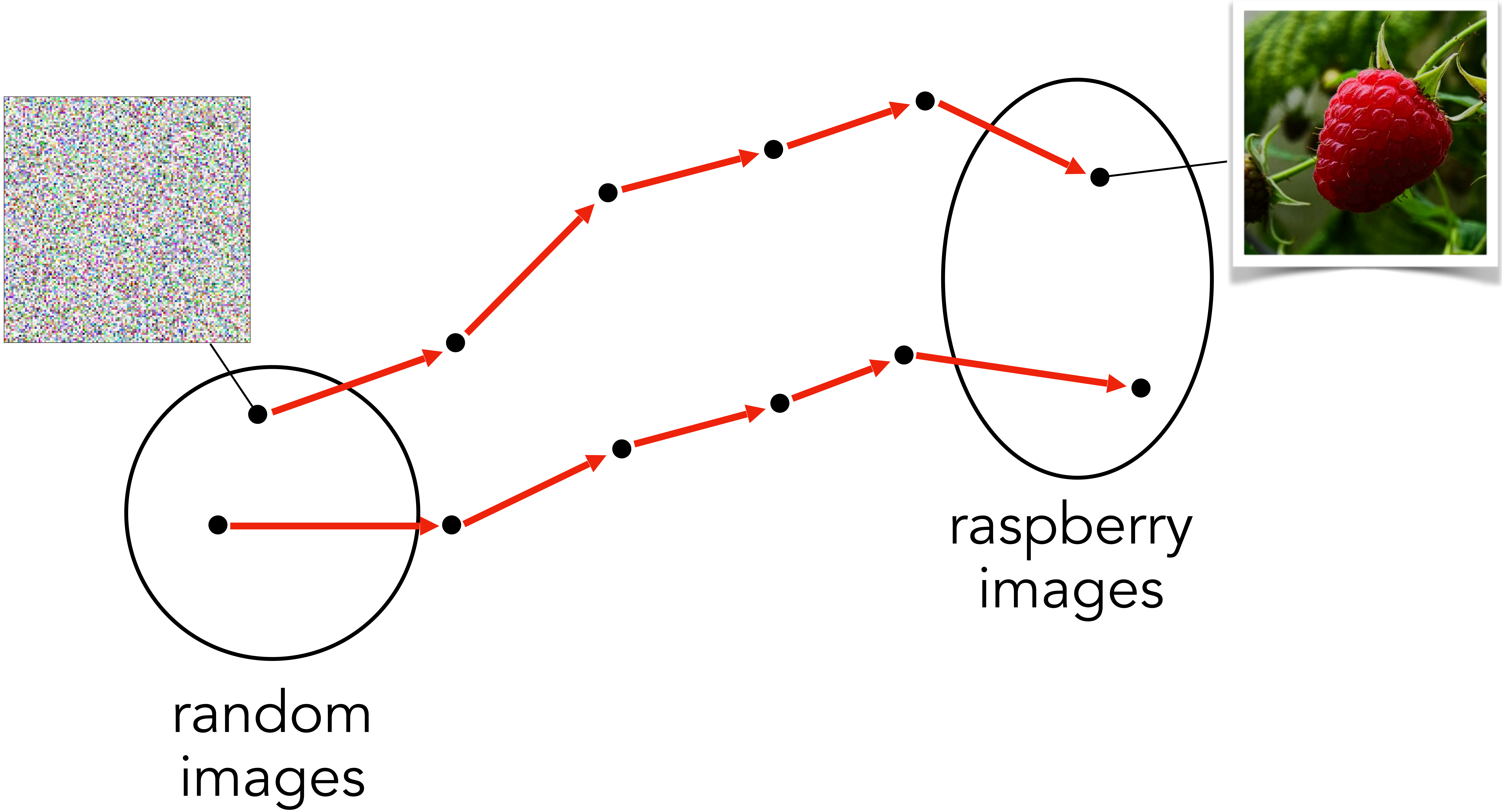
slide from Steve Seitz's video



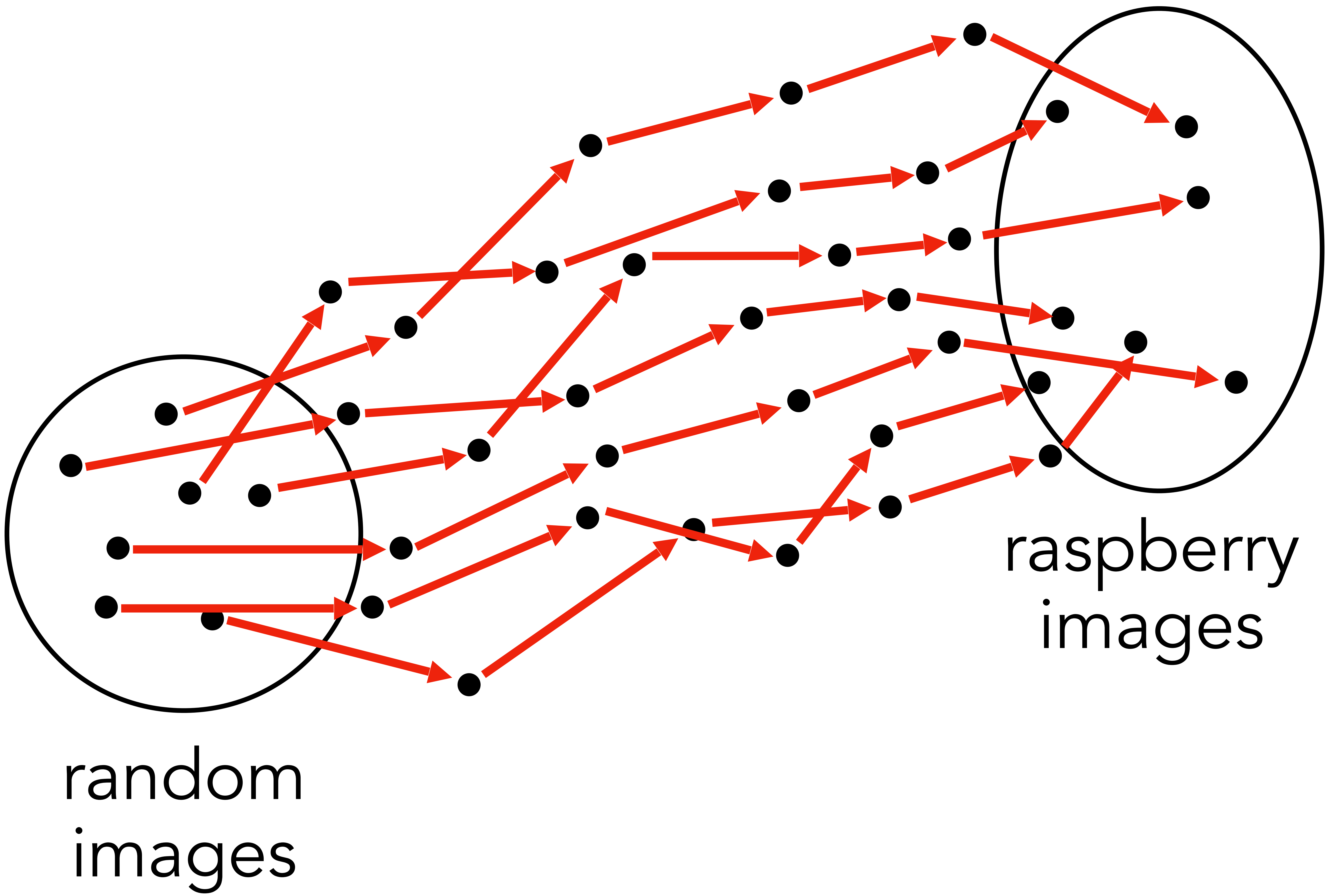
slide from Steve Seitz's video



slide from Steve Seitz's [video]



slide from Steve Seitz's video



slide from Steve Seitz's video

First, the intuition

Training

Training

1. Take real data, corrupt it to left the distribution somehow

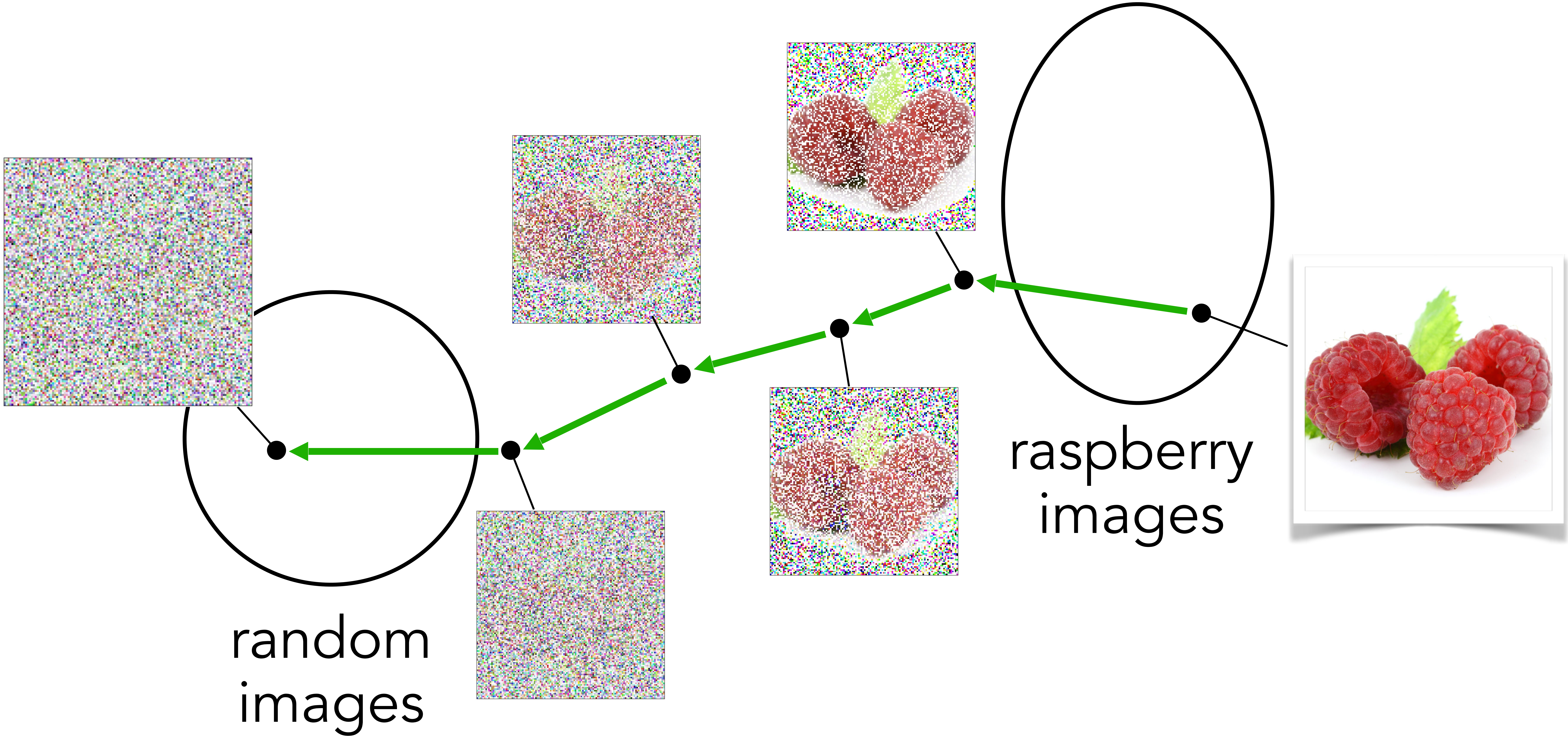


Figure from Steve Seitz's video

Training

1. Take real data, corrupt it to left distribution somehow
2. Learn to undo the process!

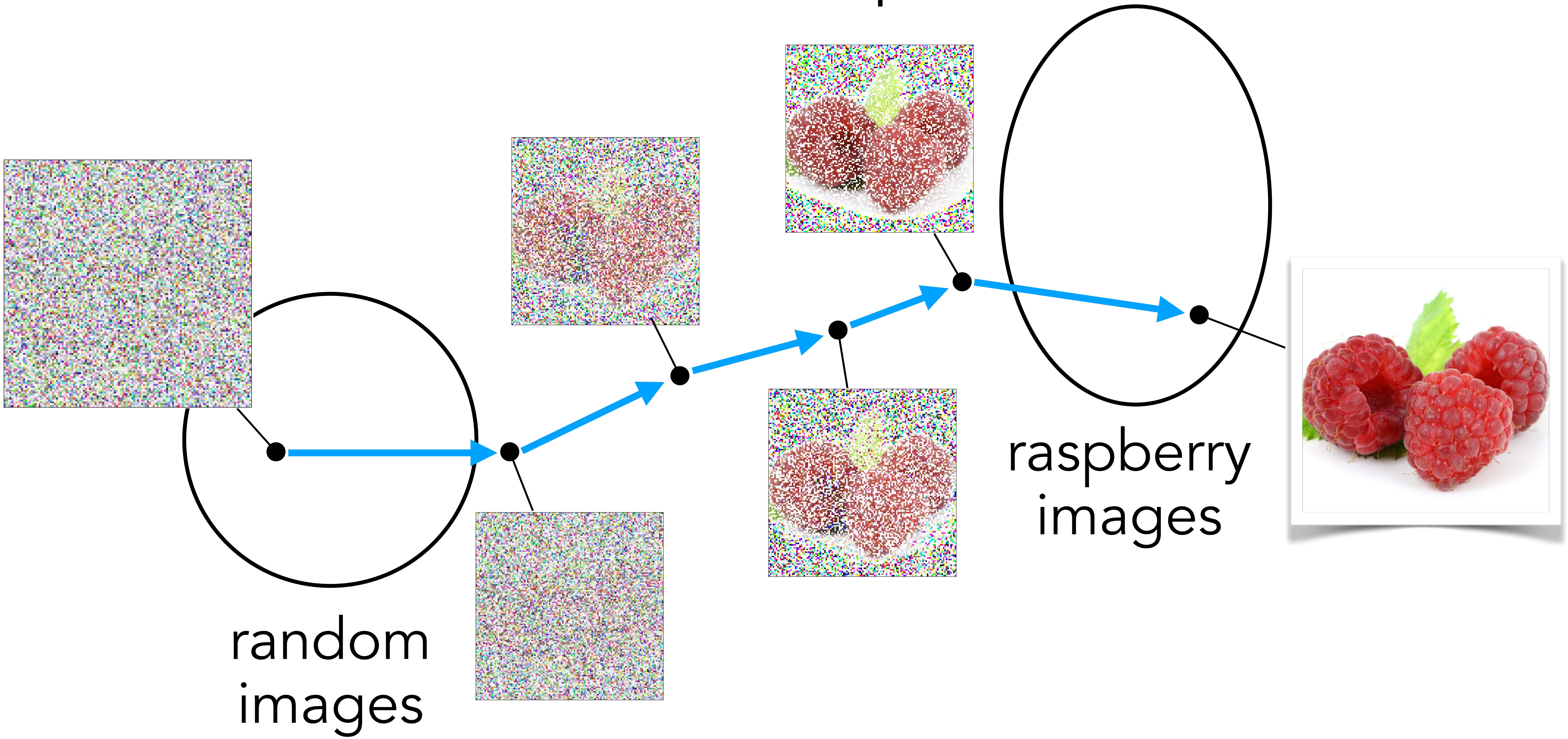
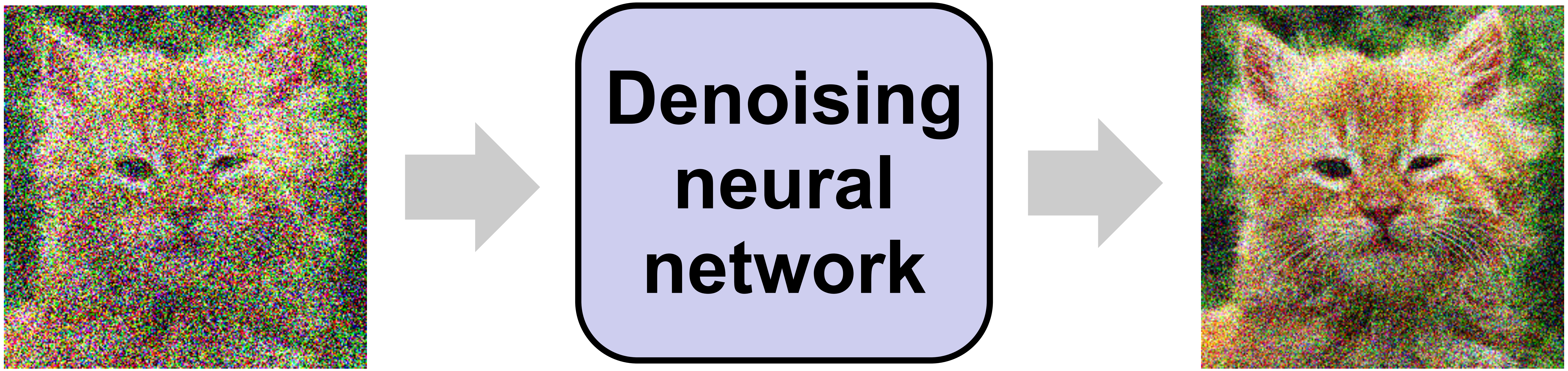


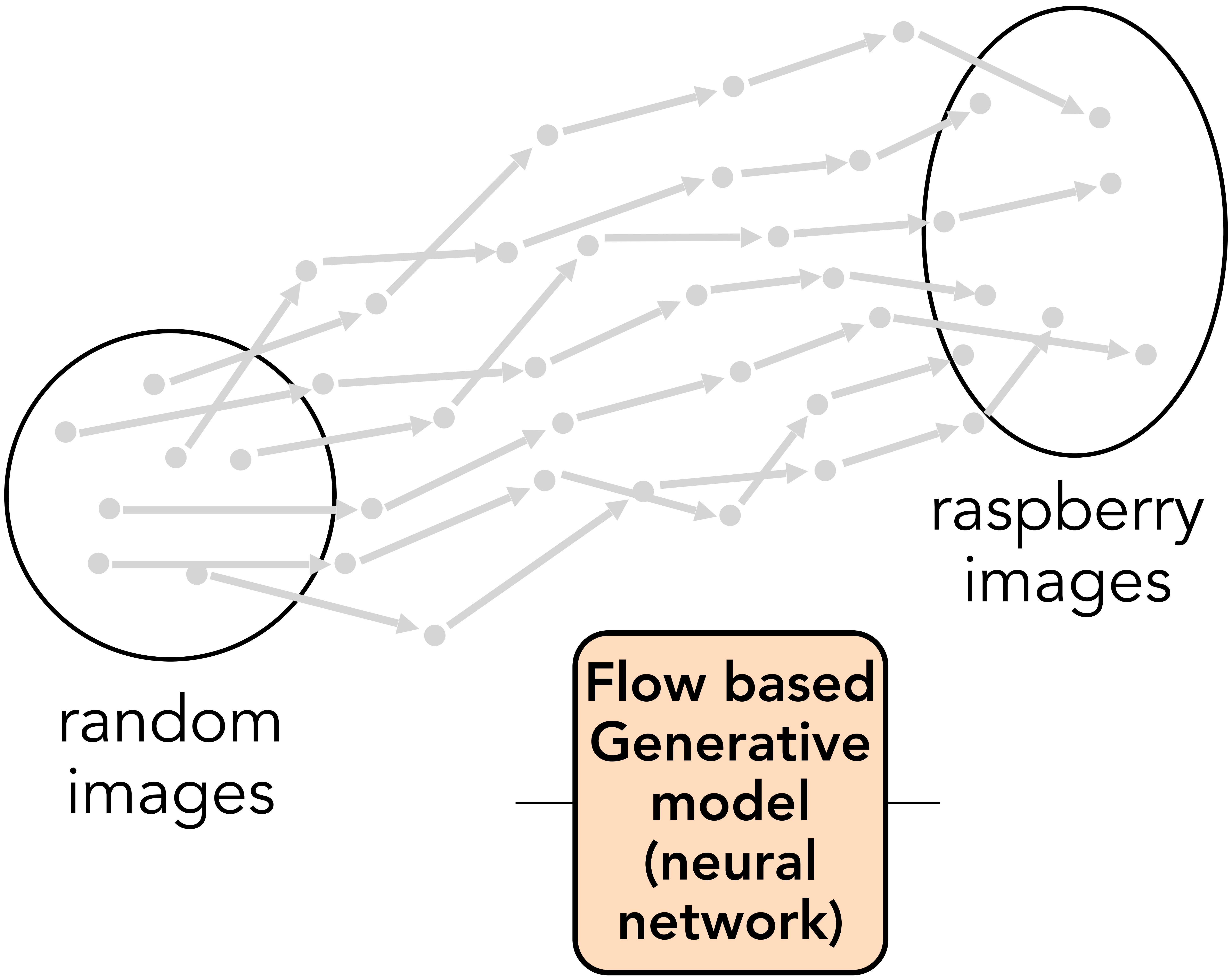
Figure from Steve Seitz's [video]

Denoising with a neural network

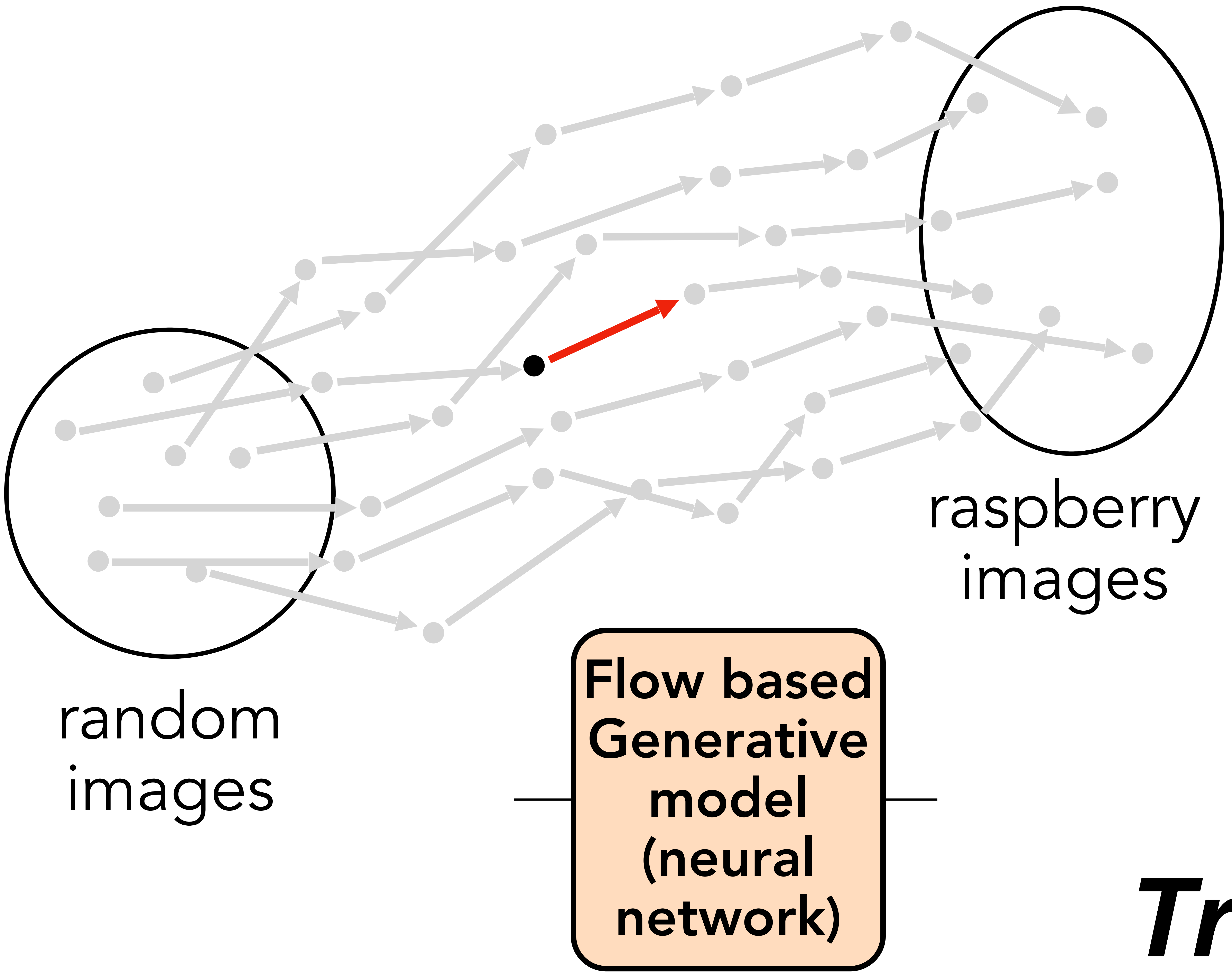


This network can be a U-Net or other suitable image-to-image network

Slide source: Steve Seitz

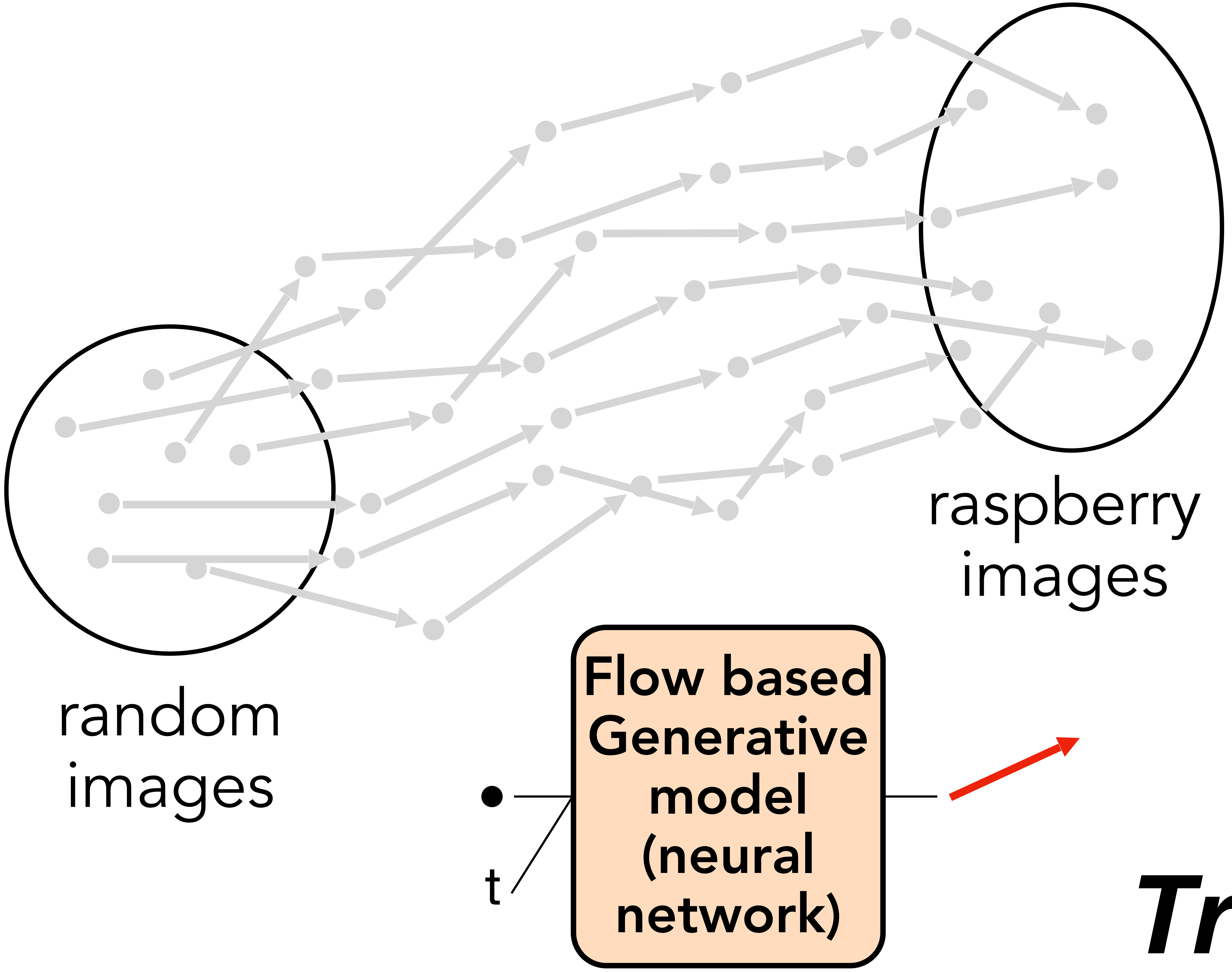


slide from Steve Seitz's [video]



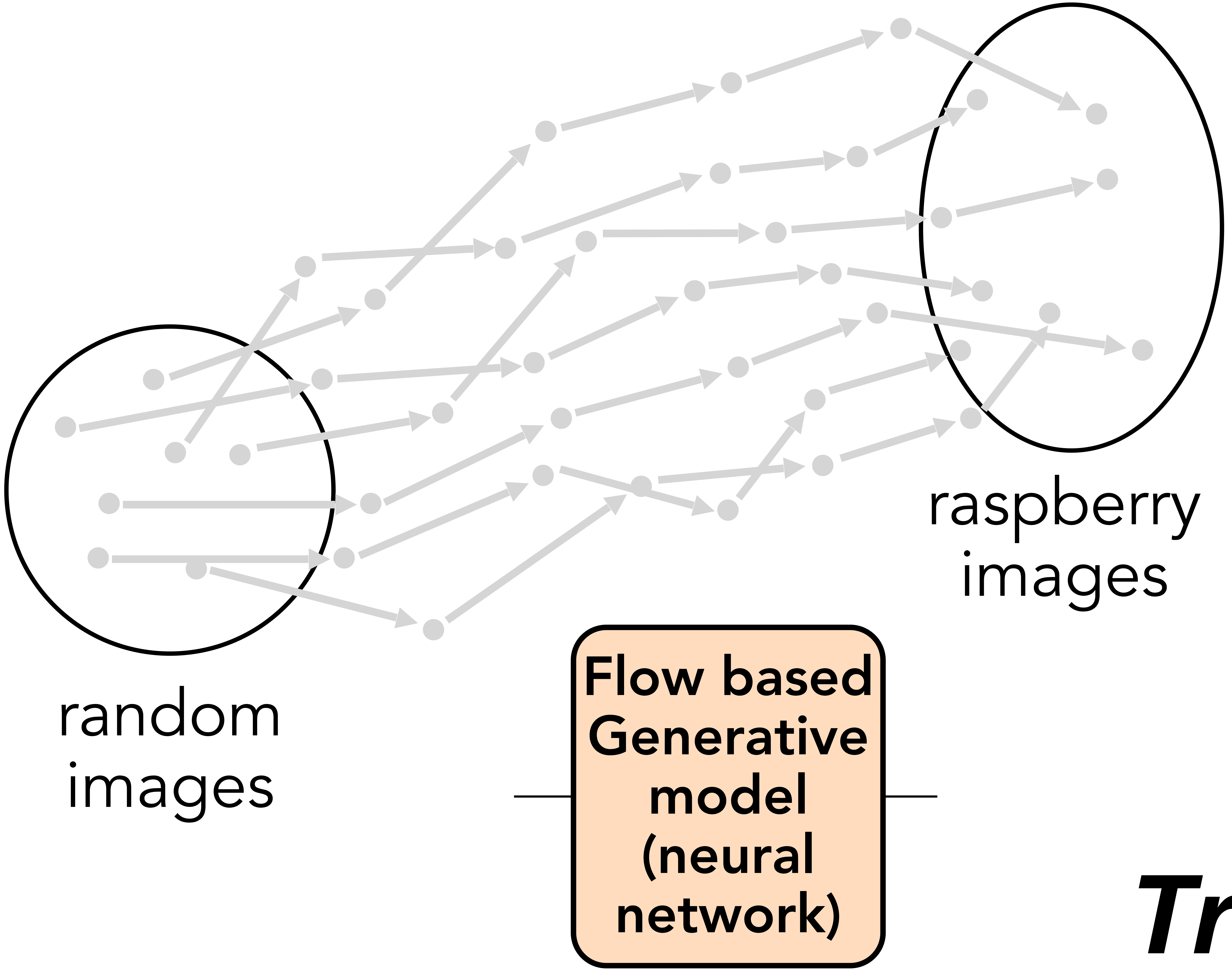
Training

slide from Steve Seitz's video



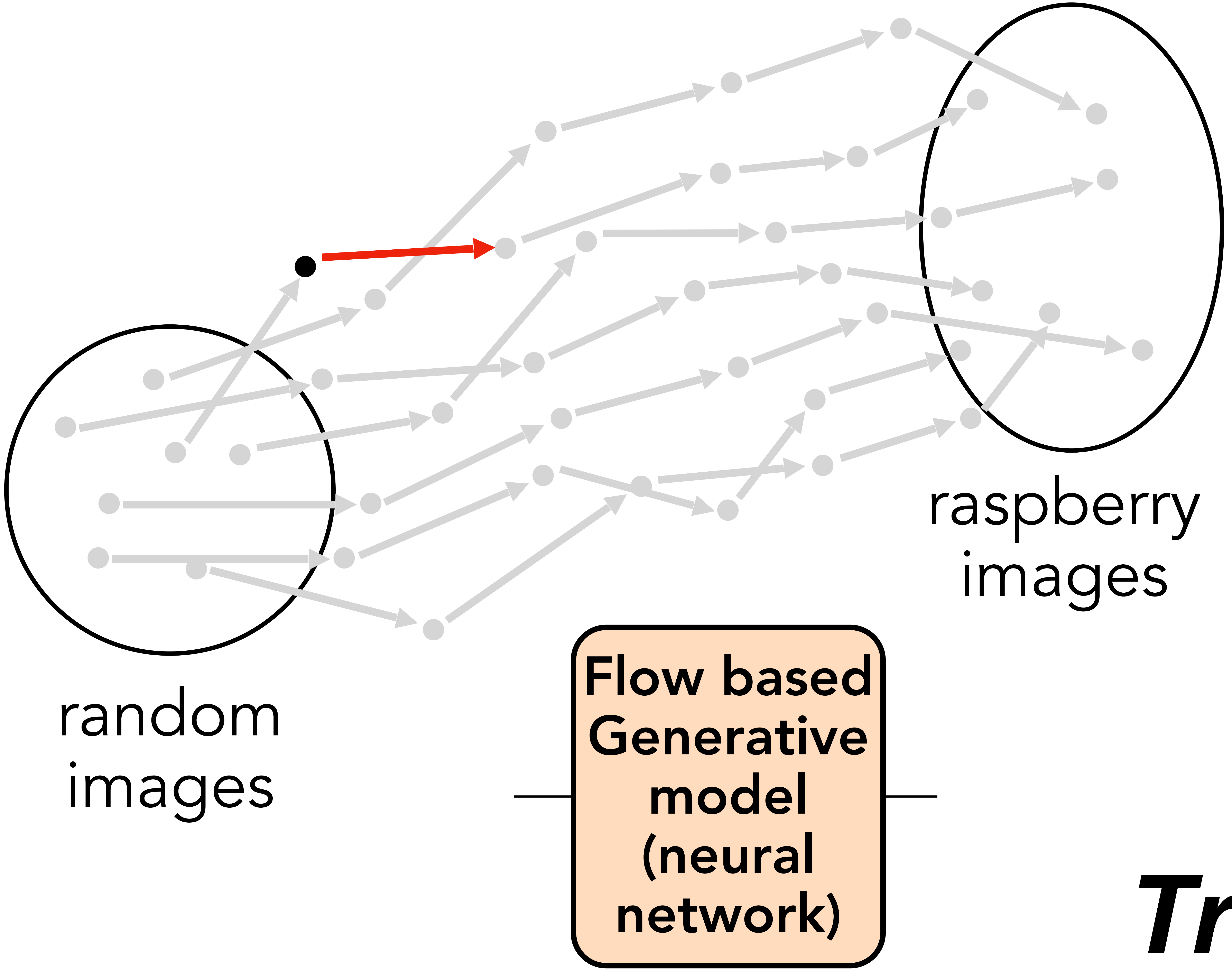
Training

slide from Steve Seitz's video



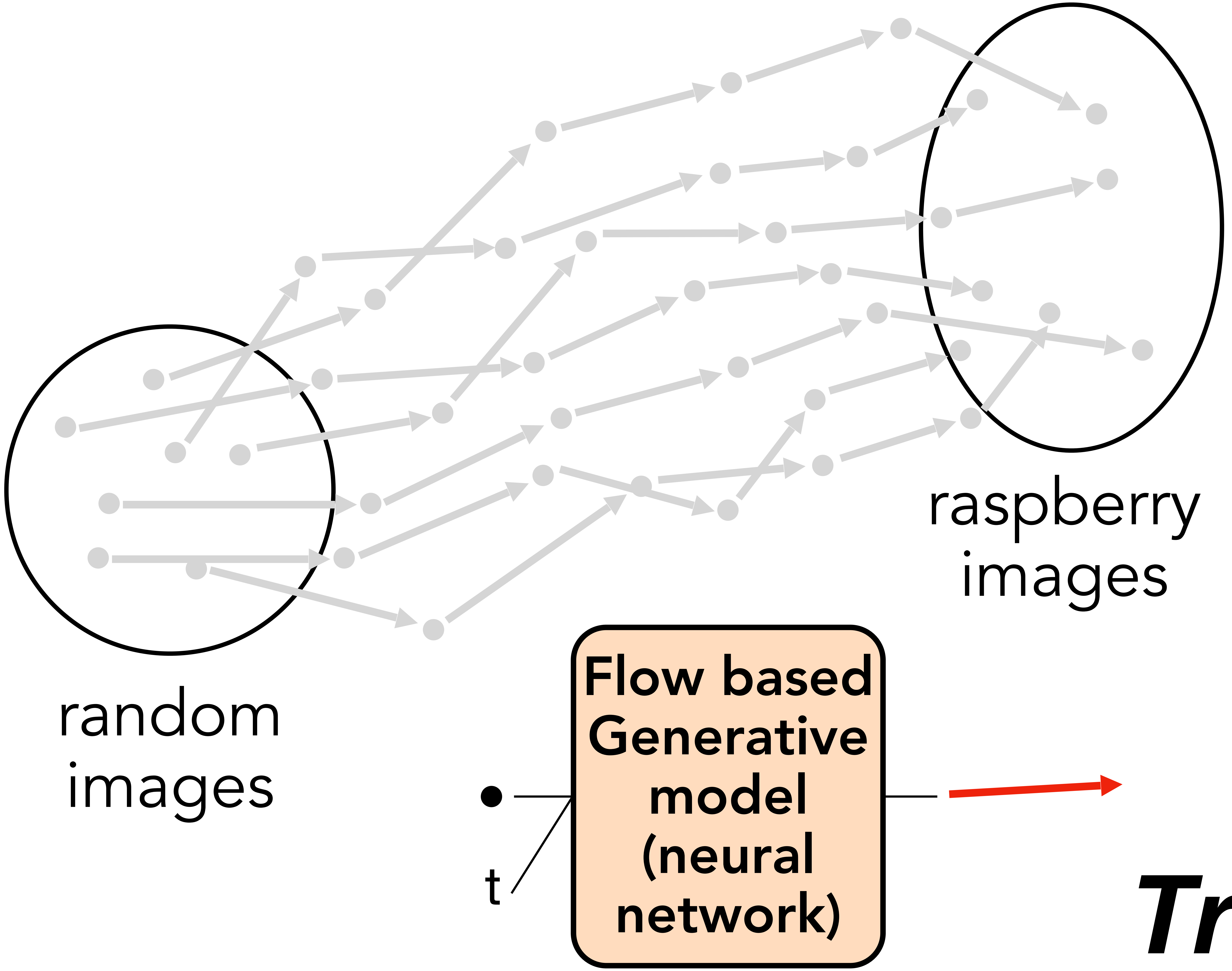
Training

slide from Steve Seitz's video



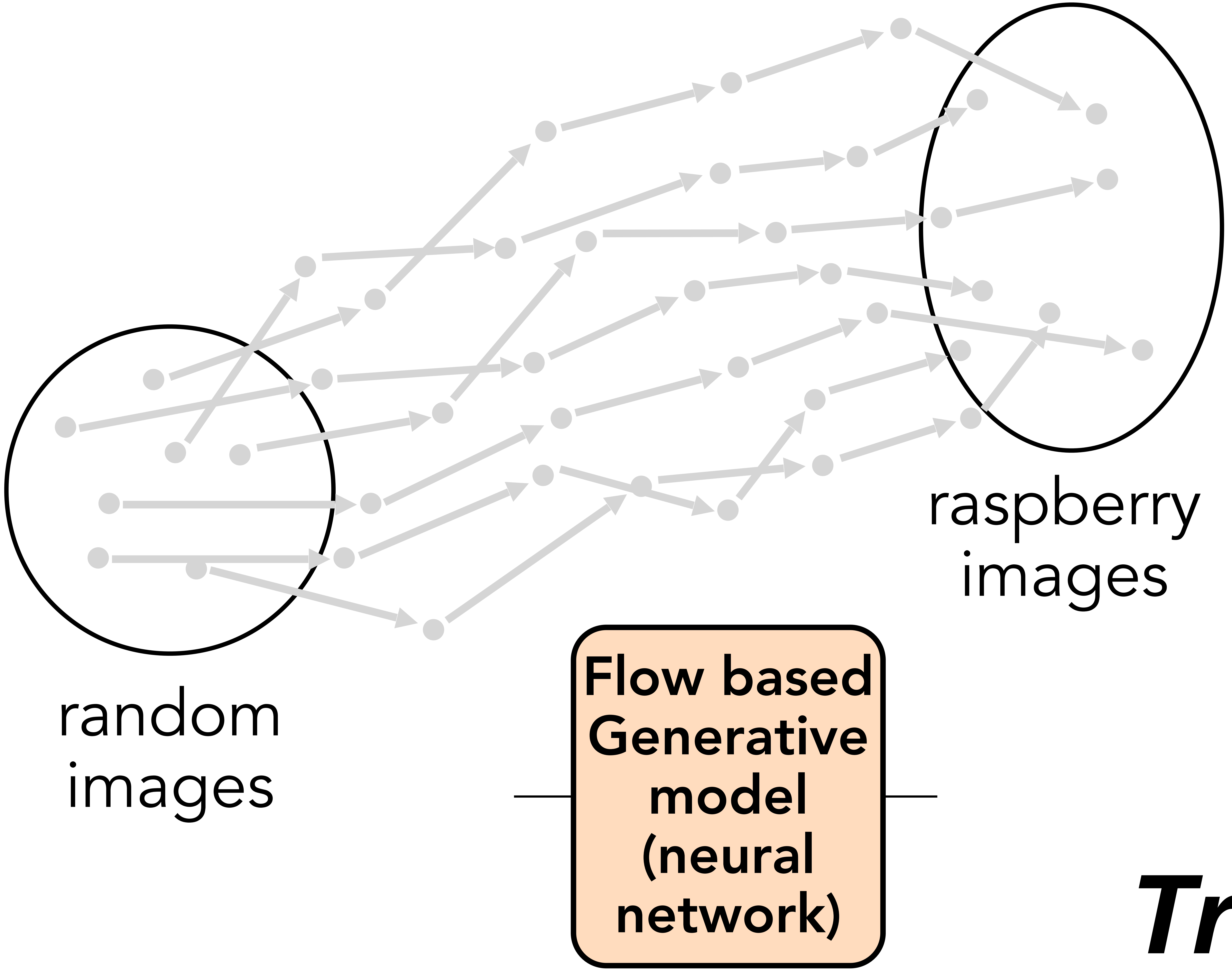
Training

slide from Steve Seitz's video



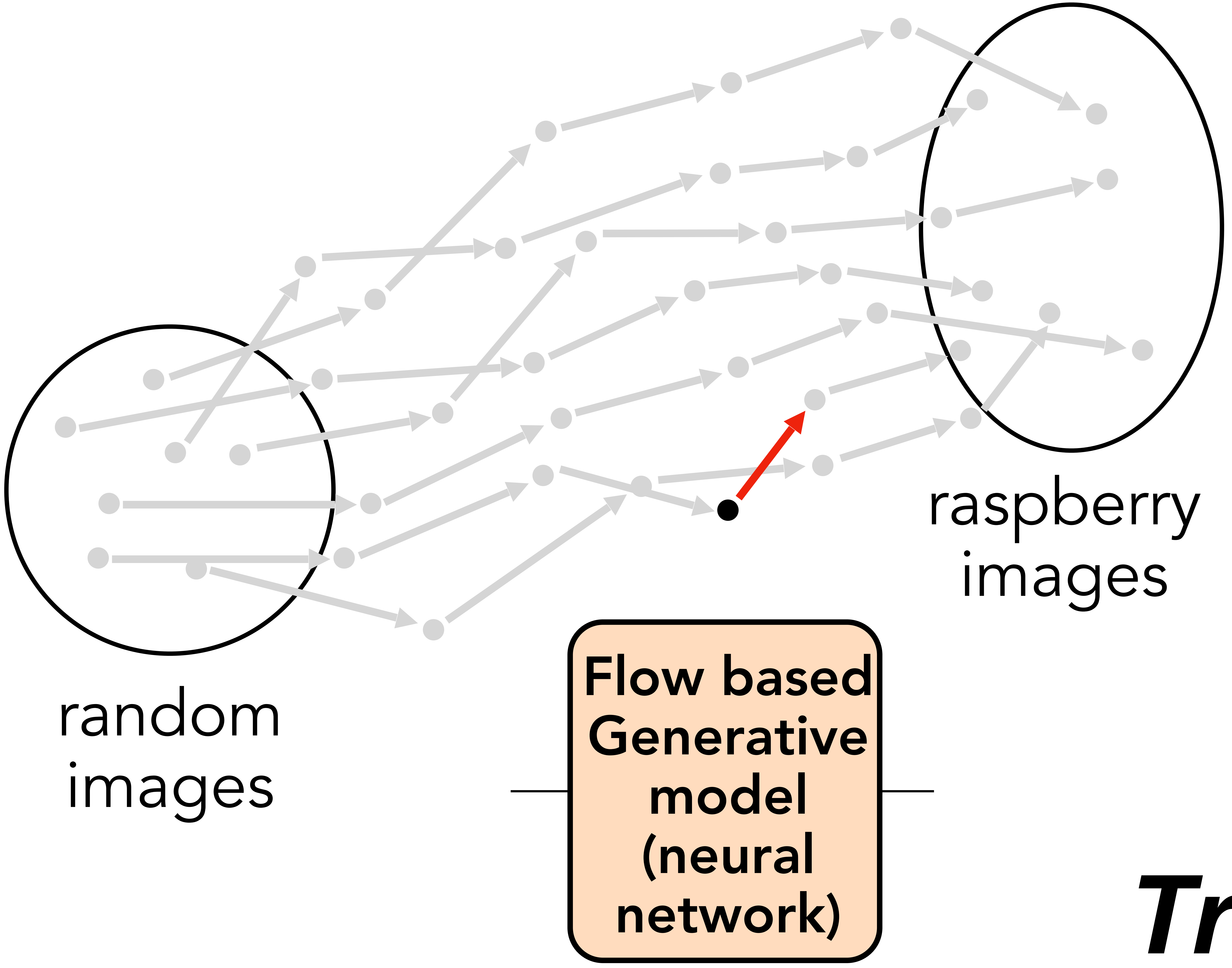
Training

slide from Steve Seitz's video



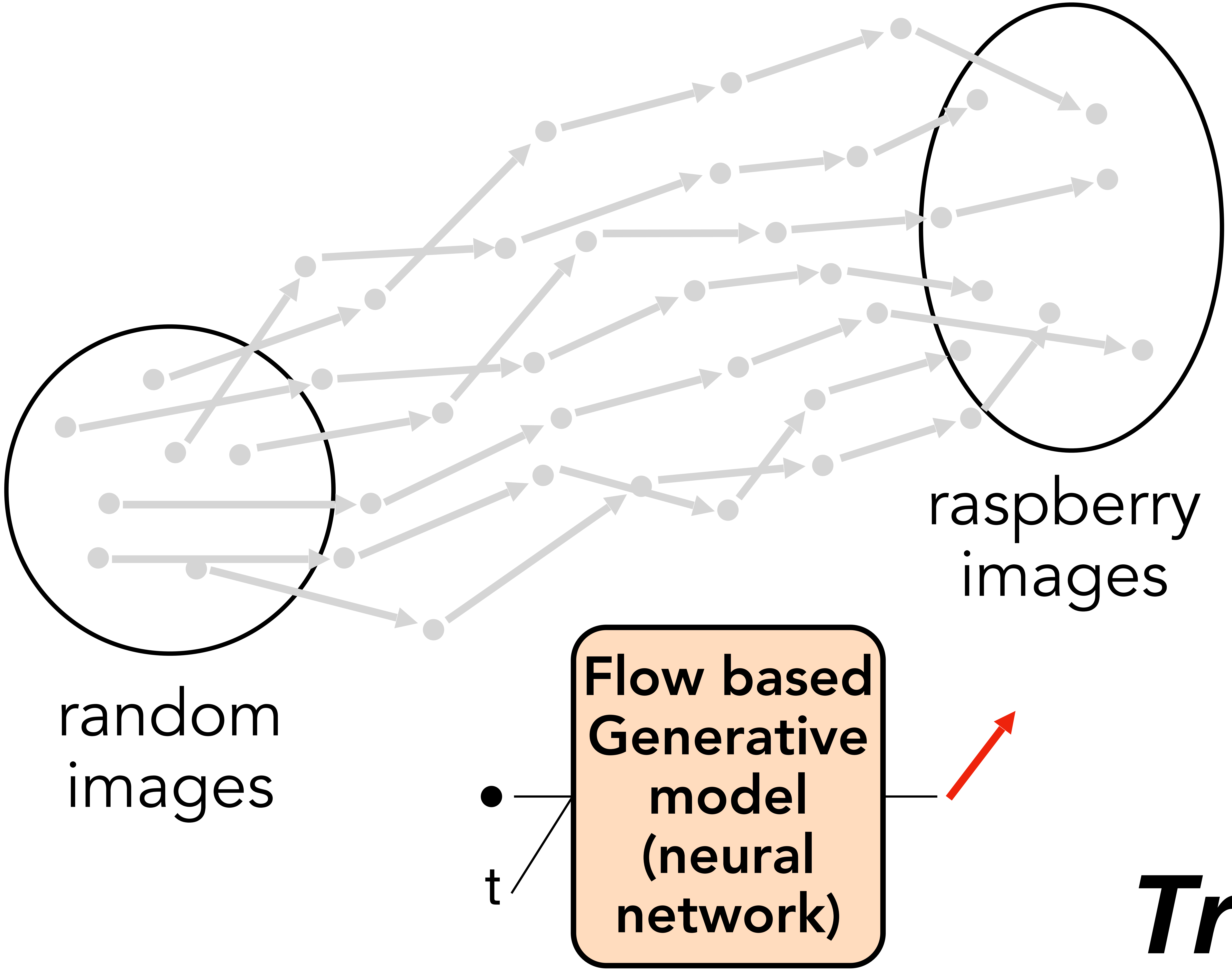
Training

slide from Steve Seitz's video



Training

slide from Steve Seitz's video



Training

slide from Steve Seitz's video

$$$ question, how to pick the intermediate path?

How to generate this Green path?

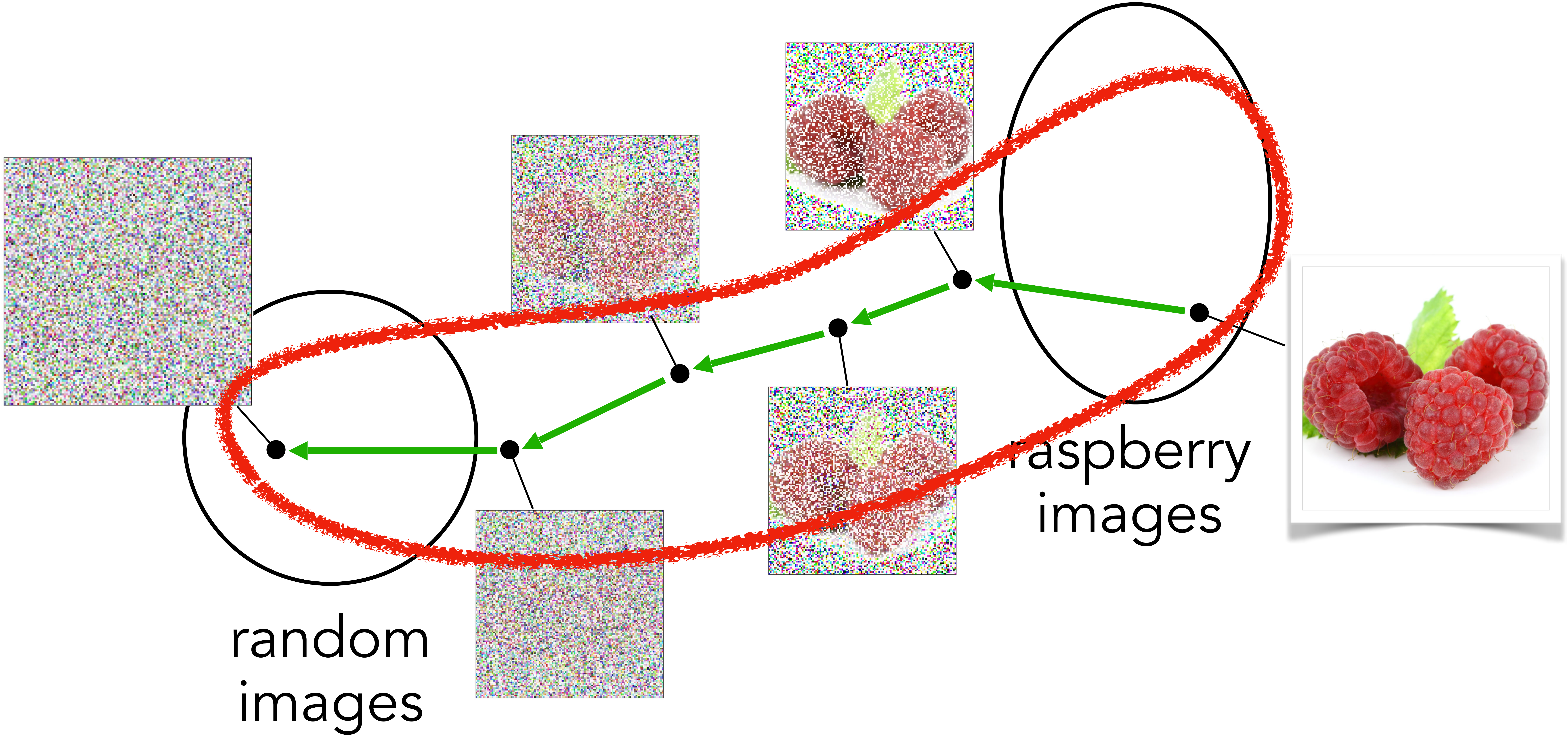


Figure from Steve Seitz's [video]

What is the path?

- How to add noise? What kind of noise?? What schedule to add them???
- Lots of math here in the diffusion literature! Can we keep it simple?

Flow Matching [Lipman et al. 2022] !

Flow matching basically says, you can add noise however you like!

Training

TLDR: Sample noise, add it, then reconstruct the data

Flow matching says you can **pick any combination**, as long as it starts from a sample in the source distribution and ends with a sample in the target distribution (image)

$$x_t = \alpha_t x_0 + \sigma_t x_1$$

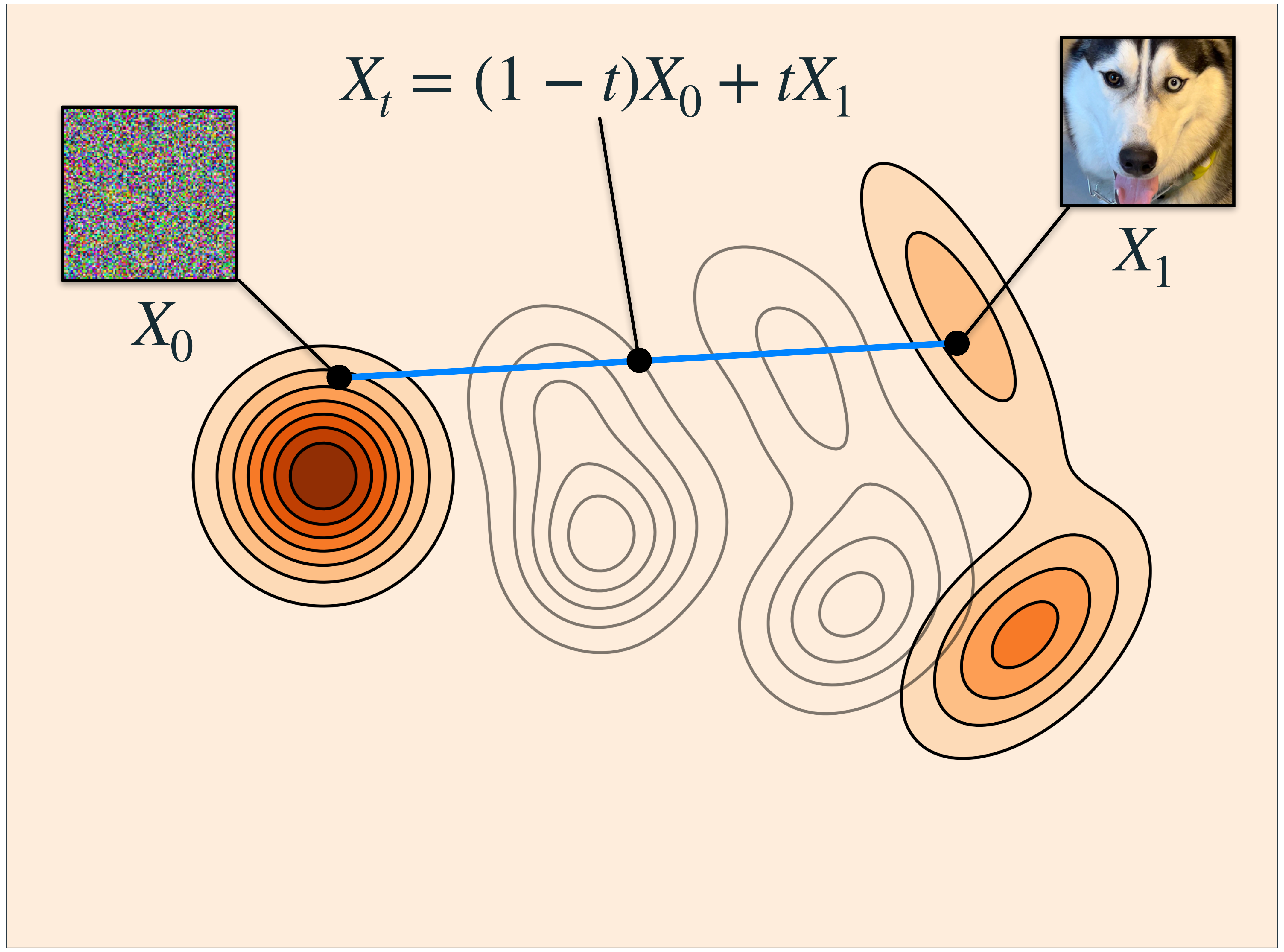
$$x_0 \sim p_0(x) \qquad\qquad x_1 \sim p_1(x)$$

A Very Simple Way

Linear interpolation!

$$x_t = \alpha_t x_0 + \sigma_t x_1$$

$$x_t = (1 - t)x_0 + tx_1$$



What is the supervision?

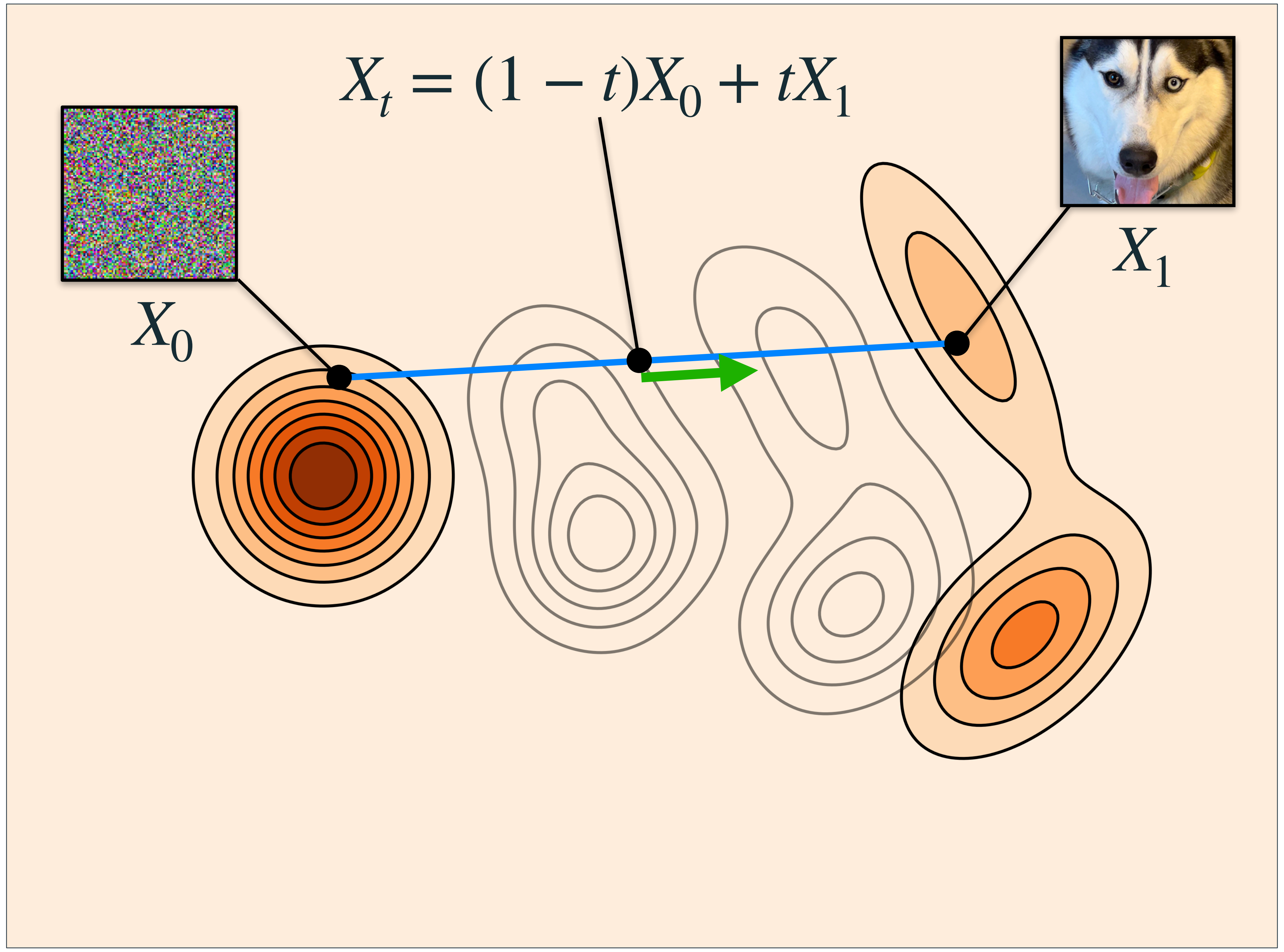
$$x_t = \alpha_t x_0 + \sigma_t x_1$$

$$x_t = (1 - t)x_0 + t x_1$$

$$\frac{dx_t}{dt} = -x_0 + x_1$$

$$= x_1 - x_0$$

$$\mathbb{E}_{t, X_0, X_1} \left\| u_t^\theta(X_t) - (X_1 - X_0) \right\|^2$$

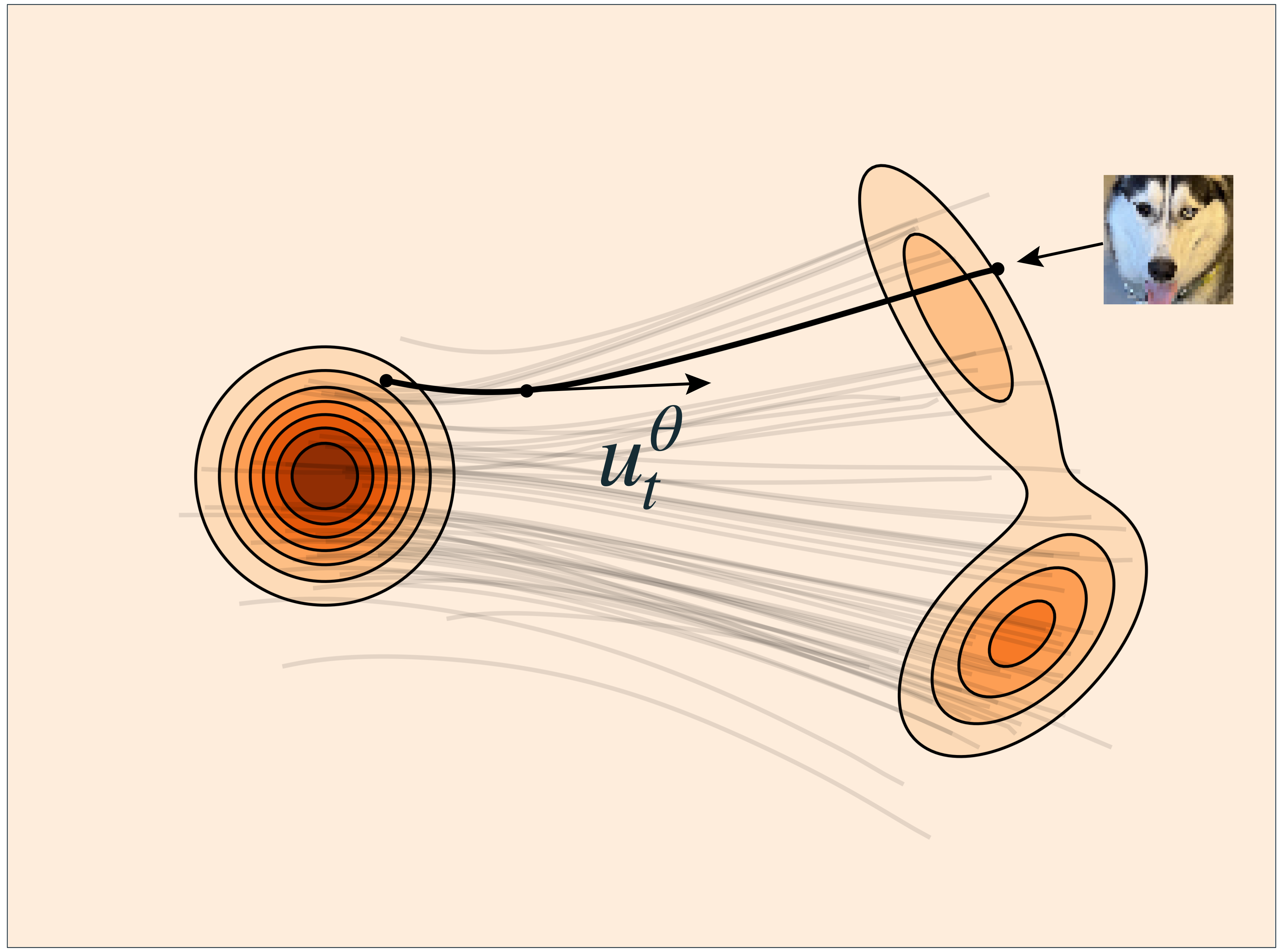


*Conditioned on a single sample

Test-time sampling

- Just take a small step in the velocity
- Use any ODE Solver, i.e. integration you like, like Euler integration:

$$x_{t+\Delta t} = x_t + \Delta t \cdot \frac{dx}{dt}\Big|_{x_t,t}$$



Sample
from $X_0 \sim p$

Model Parametrization

- Simplest — Just make your NN predict the velocity, which with the simple linear interpolation is always just x1 - x0
- Other options: Make it output the noise added or the clean image. Possible with some arithmetics
- But will have some 1/t or 1/(1-t) terms, which is annoying at the edges

Inside a Training Loop

Flow Matching

```
x = next(dataset)
t = torch.rand(1) # Sample timestep (0,1)
noise = torch.randn_like(x) # Sample noise
x_t = (1-t) * x + (t) * noise # Get noisy x_t

flow_pred = model(x_t, t) # Predict noise in x_t
flow_gt = x - noise # ground truth flow (w/ linear sched)
loss = F.mse_loss(flow_pred, flow_gt) # Update model
loss.backward()
optimizer.step()
```

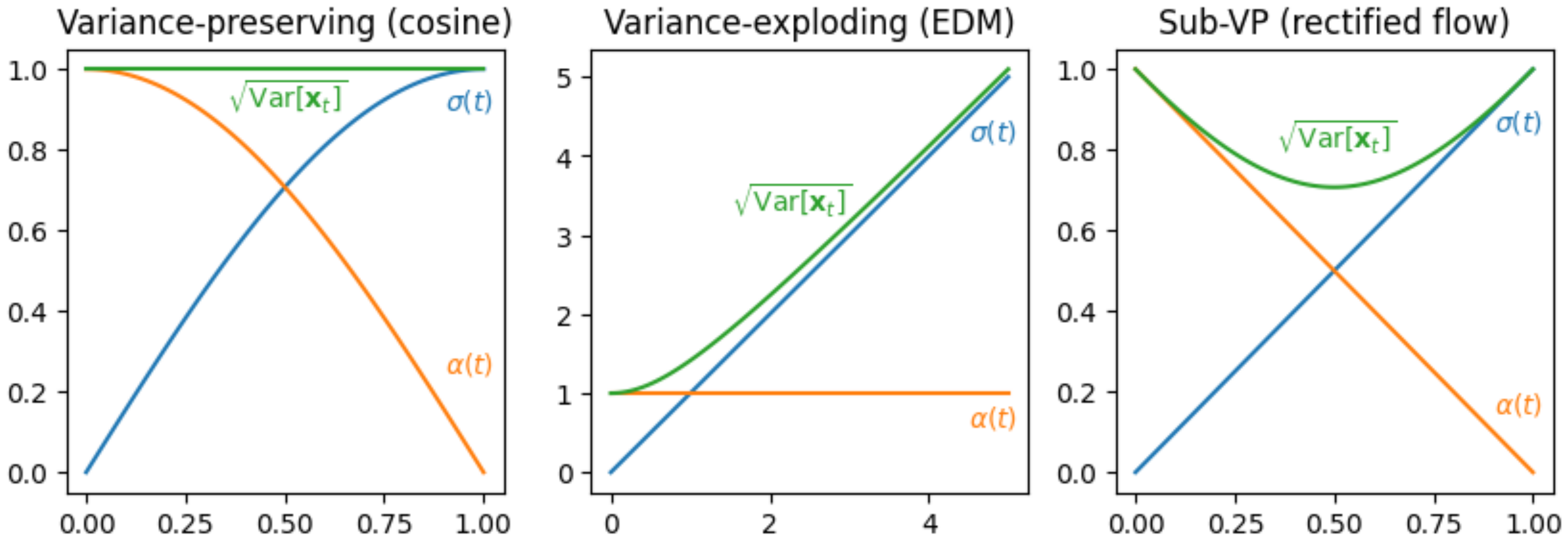
Inside Sampling Loop

```
velocity = model(x_t, t) # Predict noise in x_t
x_t = x_t + dt * velocity # Step in velocity
```

Other options lead to prior works

$$x_t = \alpha_t x_0 + \sigma_t x_1$$

- Other choices:
  - Preserve variance (VP-ODE) - DDPM
  - Exploding variance (VE-ODE) - Score Matching/DDIM
  - Linear interpolation (Flow Matching, Rectified Flow)



Why does Flow Matching work?

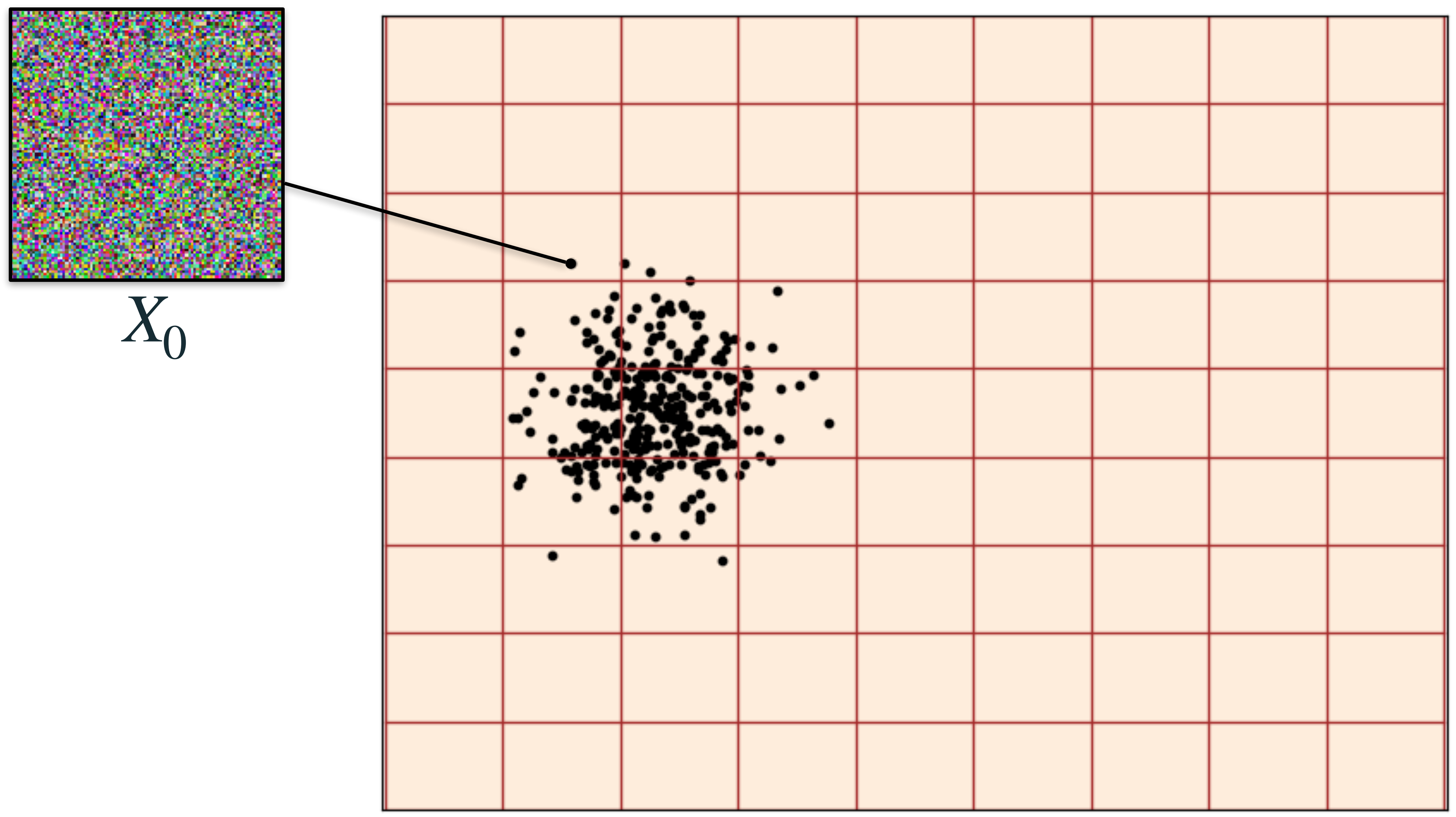
FM —> predict the velocity conditioned on a single sample

Flow as a generative model

$$X_t = \psi_t(X_0)\ , \quad t \in [0,1]$$

Warping

Source $X_0 \sim p$



- Markov: $X_{t+h} = \psi_{t+h|t}(X_t)$

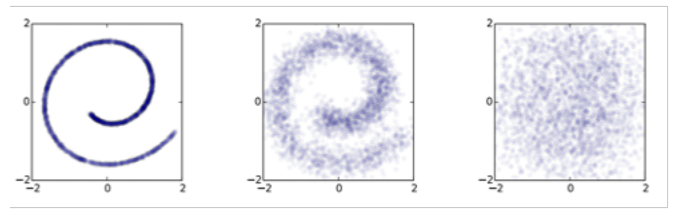
Slides from Yaron Lipman

History

Diffusion Arc

DALL-E1 Open AI 2020

DALL-E2 Open AI 2023
StableDiffusion, Stability 2023



Sohl-Dickstein et al. 2015
Deep unsupervised learning using non equilibrium thermodynamics

DDPM, Ho et al. 2020

Song et al. Score-based Generative Models, DDIM 2021

Rectified Flow, Liu et al. 2022

NICE Dinh et al.
Normalizing Flows 2015

RealNVP, Dinh et al. 2017

Glow, Kingma & Dhaliwal 2018

Flow Matching, Lipman et al. 2022

Neural ODE Chen et al. 2018…

Unification / Simplification

Flow Matching Tutorial NeurIPS 2024

MovieGen late 2024~

Normalizing Flow Arc

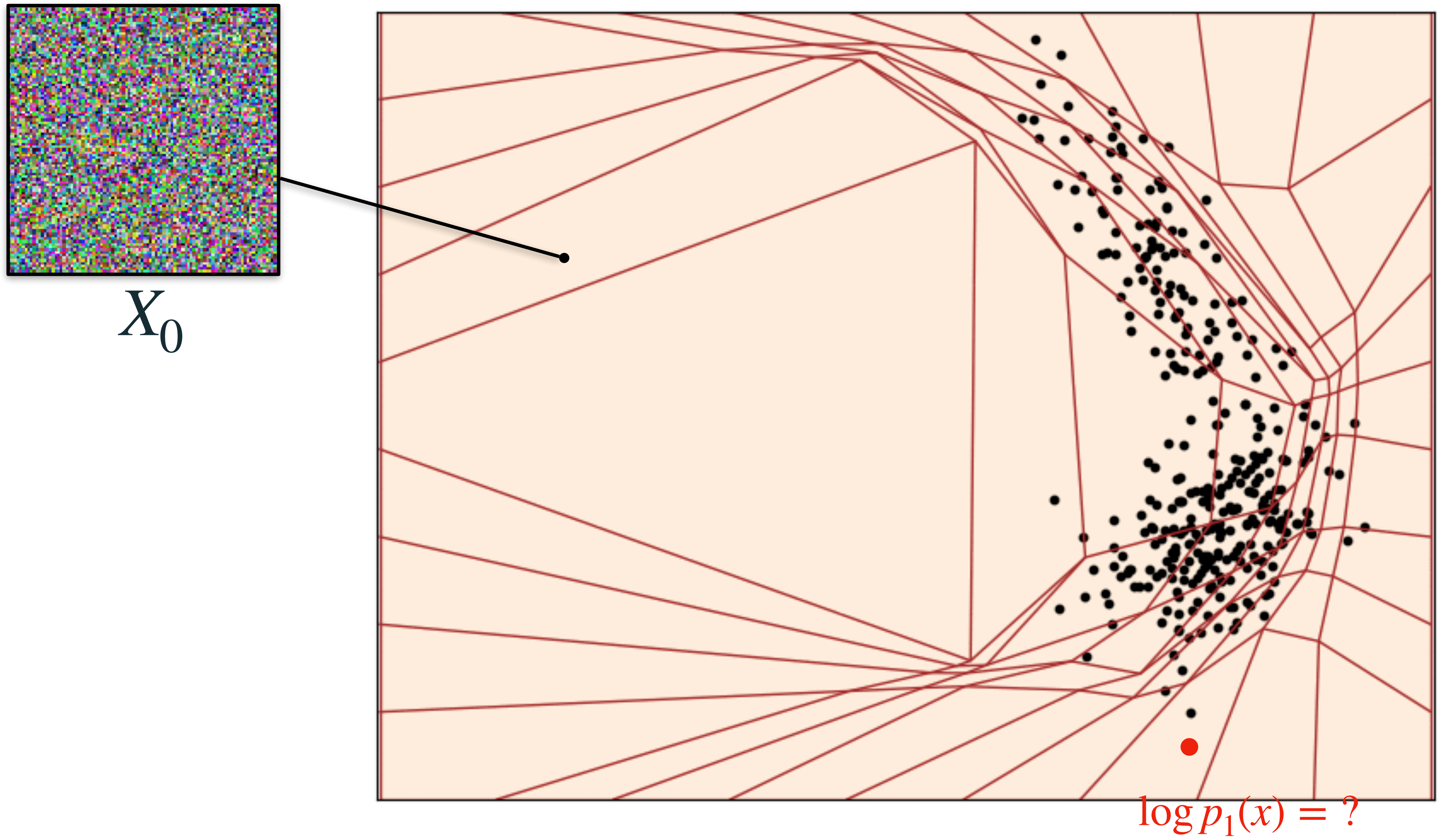
Initial approach trained flow with Maximum Likelihood

$$D_{\text{KL}}(q \,\|\, p_1) = -\mathbb{E}_{x \sim q} \log p_1(x) + c$$

$$X_t = \psi_t(X_0), \quad t \in [0,1]$$

Warping

Source $X_0 \sim p$



- Normalizing Flow, Continuous Normalizing Flow
- Requires ODE simulation DURING training with invertible neural networks

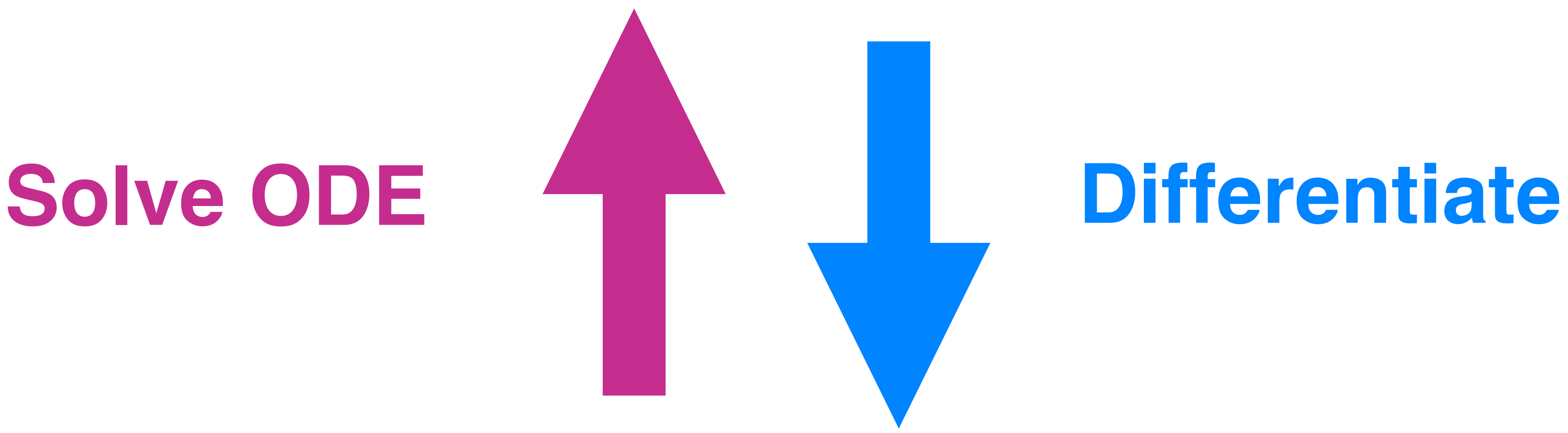
Slide adapted from Yaron Lipman

Instead, model Flow with Velocity

$$\frac{\mathrm{d}}{\mathrm{d}t}\psi_t(x) = u_t(\psi_t(x))$$

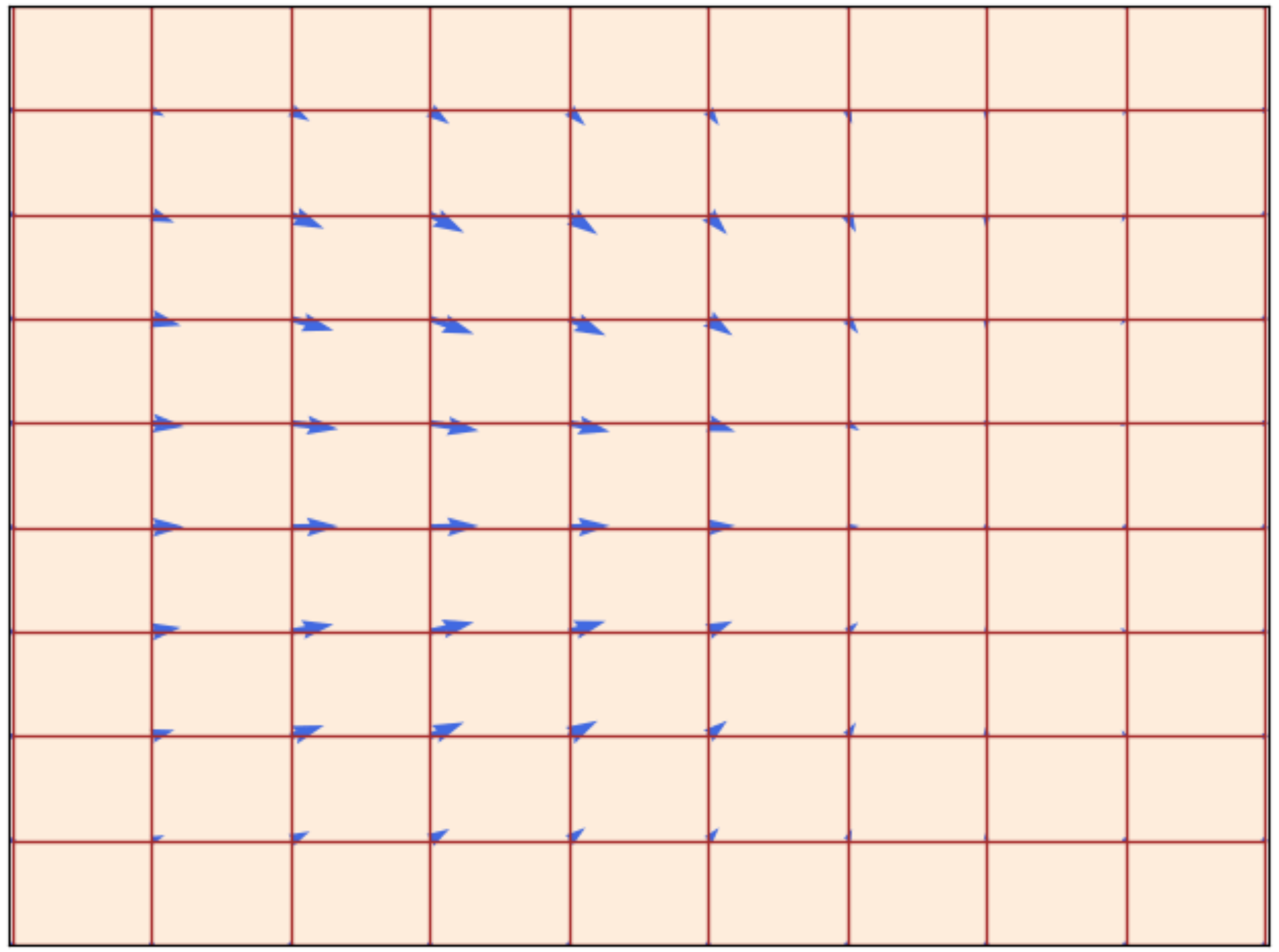
Flow

$$\psi_t(x)$$



$$u_t(x)$$

Velocity



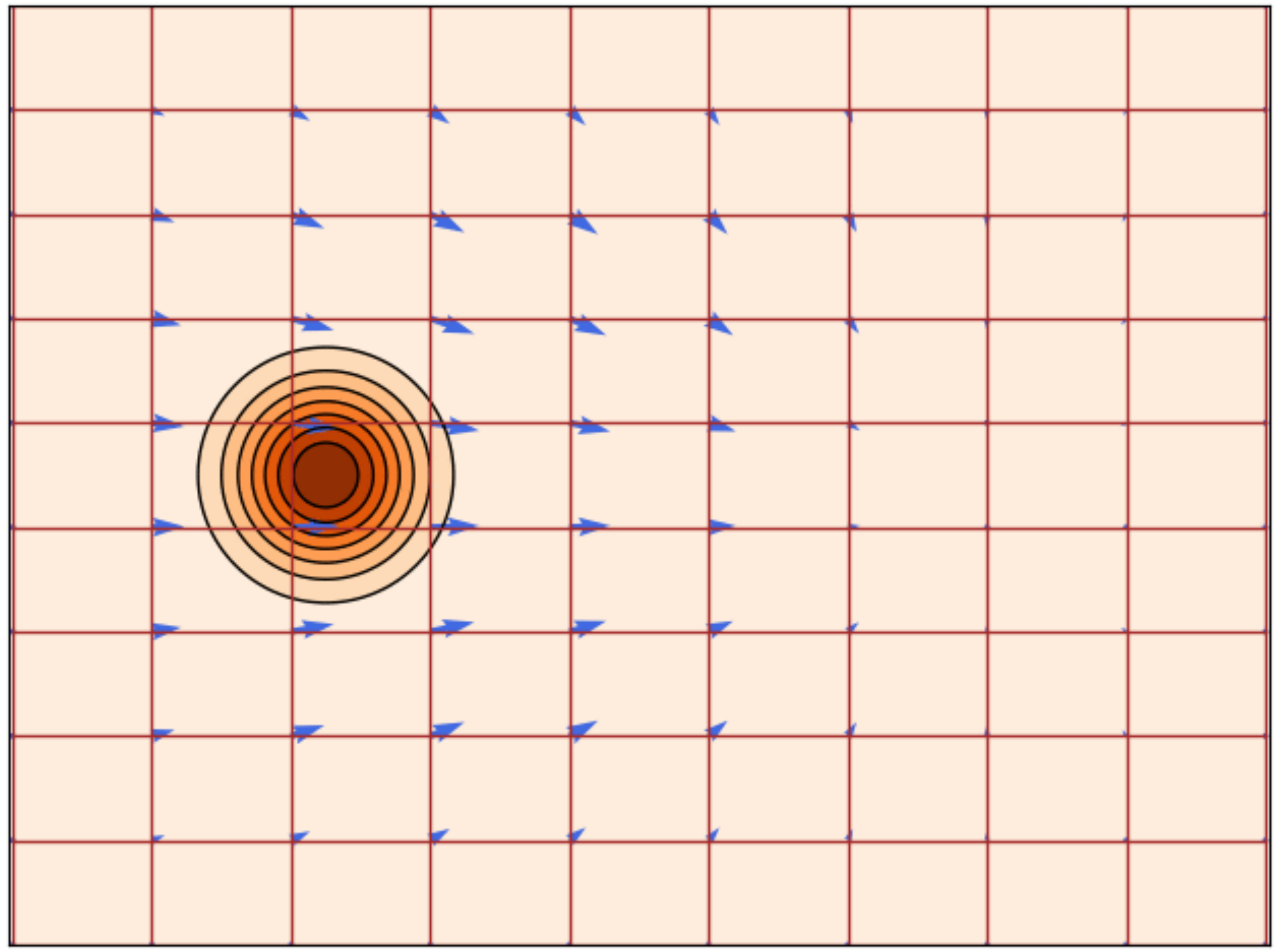
- **Pros**: velocities are ***linear***
- **Cons**: simulate to sample

Slides from Yaron Lipman

The flow gives you the marginal probability path

Velocity u_t **generates** p_t if

$$X_t = \psi_t(X_0) \sim p_t$$



u_t Marginal Flow

p_t Marginal Probability Path

We want u_t which is given by p_t .

Great! But what is the actual marginal flow?? We don't have this!

Slides from Yaron Lipman

The Marginalization Trick

Theorem*: The marginal velocity generates the marginal probability path.

$$u_t(x) = \mathbb{E}\left[u_t(X_t \mid X_1) \mid X_t = x\right] \qquad p_t(x) = \mathbb{E}_{X_1} p_{t|1}(x \mid X_1)$$

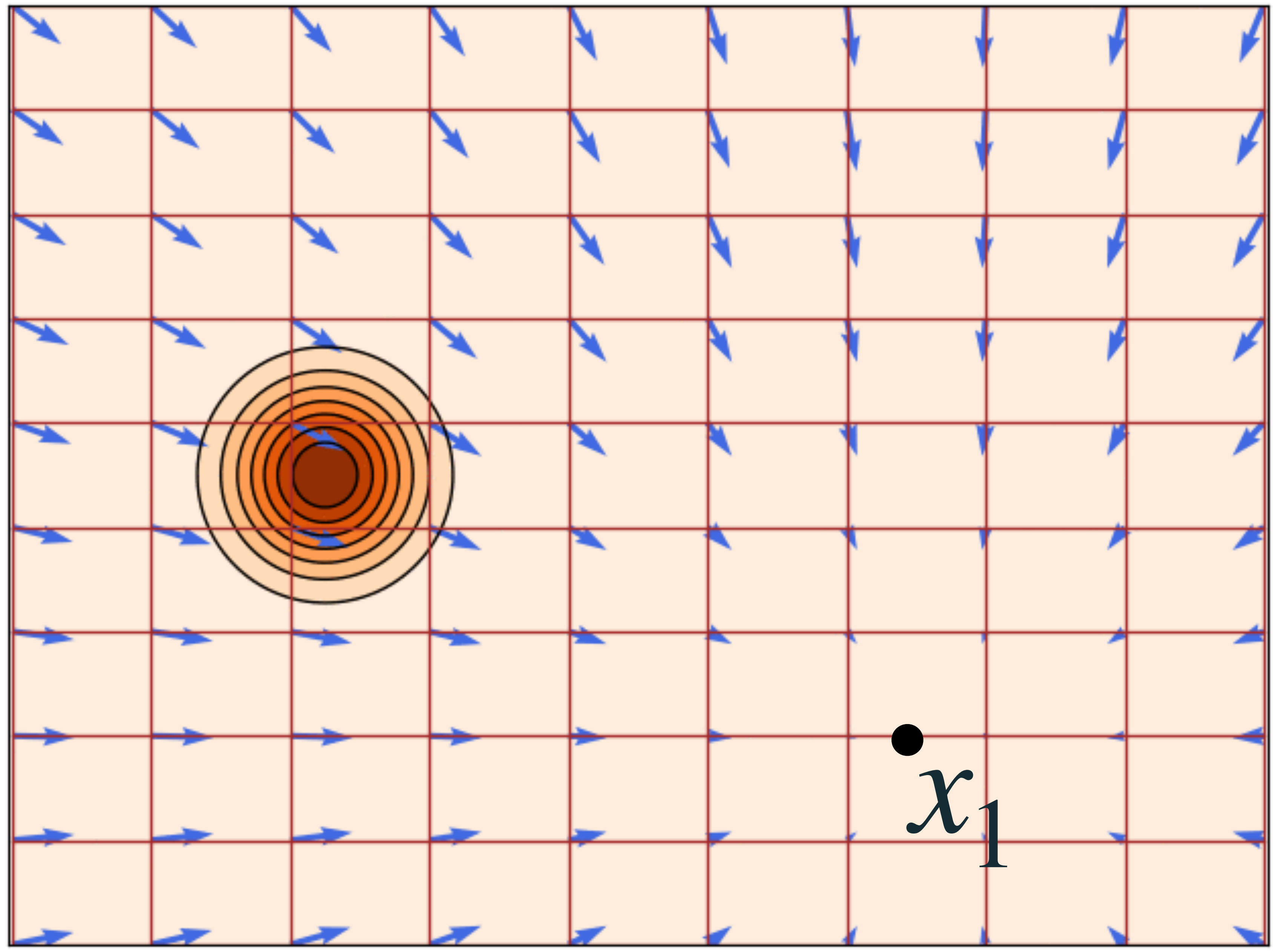
"Flow Matching for Generative Modeling" Lipman el al. (2022)

"Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow" Liu et al. (2022)

"Building Normalizing Flows with Stochastic Interpolants" Albergo et al. (2022)

Build flow from conditional flows

Generate a single target point

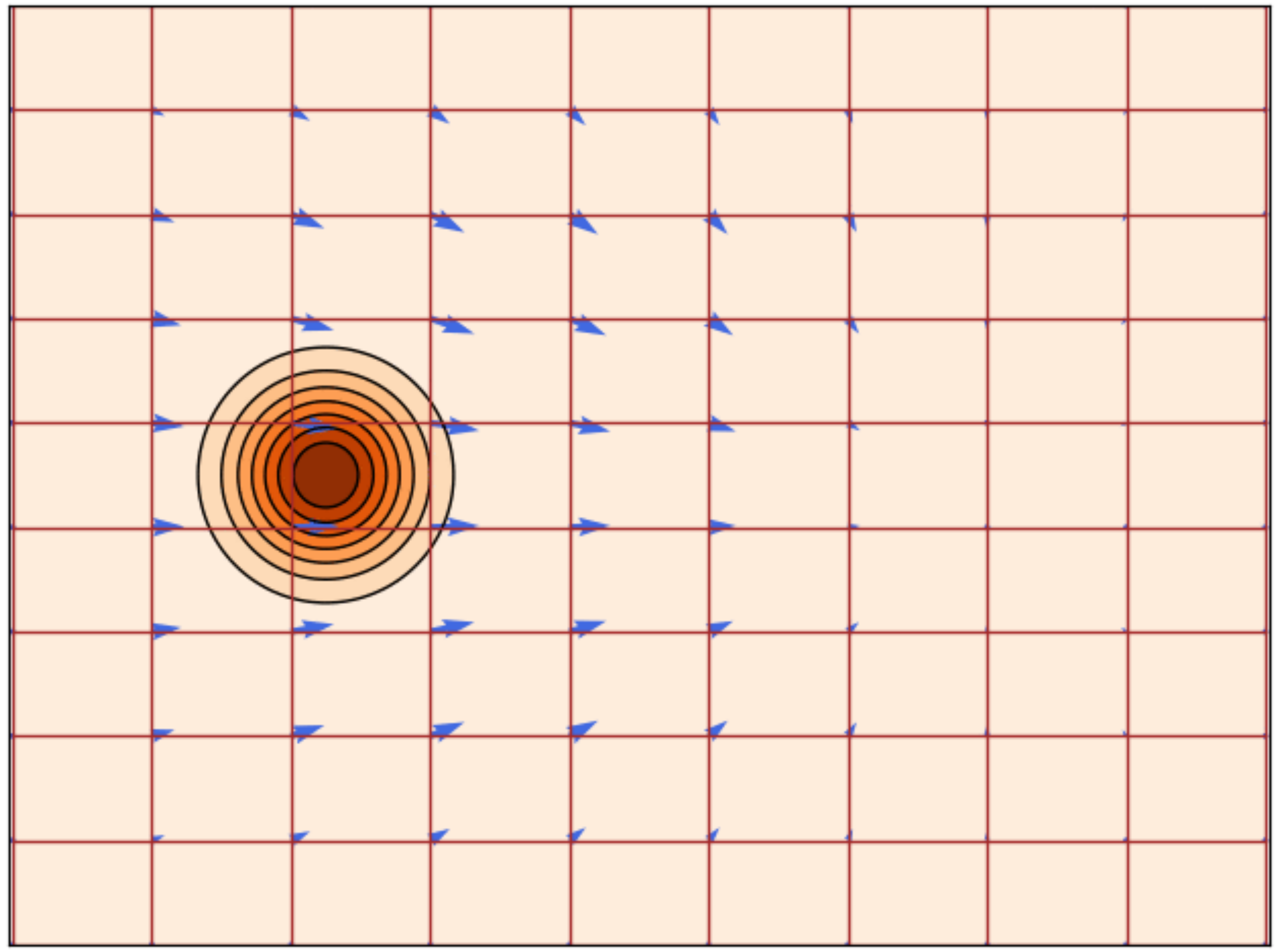


$$X_t = \psi_t(X_0 \,|\, x_1) = (1-t)X_0 + tx_1$$

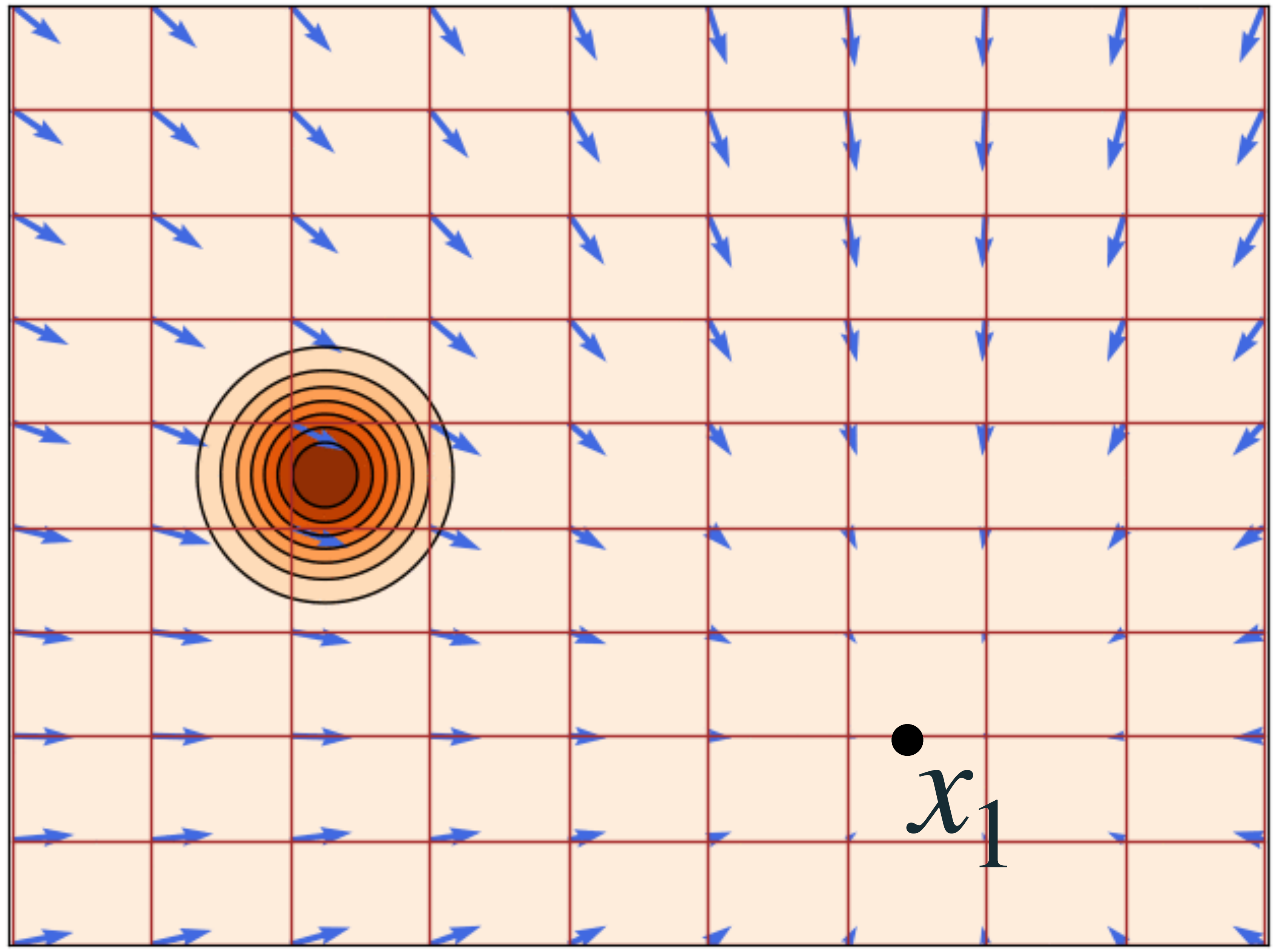
$p_{t|1}(x \,|\, x_1)$ conditional probability

$u_t(x \,|\, x_1)$ conditional velocity

Build flow from conditional flows



Generate a single target point



$$X_t = \psi_t(X_0 \mid x_1) = (1-t)X_0 + t x_1$$

$$p_t(x) = \mathbb{E}_{X_1} p_{t|1}(x \mid X_1) \longleftarrow p_{t|1}(x \mid x_1)$$ conditional probability

average

$$u_t(x) = \mathbb{E}\left[u_t(X_t \mid X_1) \,\middle|\, X_t = x\right] \longleftarrow u_t(x \mid x_1)$$ conditional velocity

Flow Matching Loss

- **Flow Matching loss:**

$$\mathscr{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t,X_t} \left\| u_t^\theta(X_t) - u_t(X_t) \right\|^2$$

- **Conditional Flow Matching loss**:

$$\mathscr{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,X_1,X_t} \left\| u_t^\theta(X_t) - u_t(X_t \mid X_1) \right\|^2$$

Theorem: Losses are equivalent,

$$\nabla_\theta \mathscr{L}_{\mathrm{FM}}(\theta) = \nabla_\theta \mathscr{L}_{\mathrm{CFM}}(\theta)$$

Training: Flow Matching vs. Diffusion

Algorithm 1: Flow Matching training.

Input : dataset q, noise p
Initialize v^θ
while *not converged* **do**
- $t \sim \mathcal{U}([0,1])$ ▷ sample time
- $x_1 \sim q(x_1)$ ▷ sample data
- $x_0 \sim p(x_0)$ ▷ sample noise
- $x_t = \Psi_t(x_0|x_1)$ ▷ conditional flow
- Gradient step with $\nabla_\theta \|v_t^\theta(x_t) - \dot{x}_t\|^2$

Output: v^θ

Algorithm 2: Diffusion training.

Input : dataset q, noise p
Initialize s^θ
while *not converged* **do**
- $t \sim \mathcal{U}([0,1])$ ▷ sample time
- $x_1 \sim q(x_1)$ ▷ sample data
- $x_t = p_t(x_t|x_1)$ ▷ sample conditional prob
- Gradient step with $\nabla_\theta \|s_t^\theta(x_t) - \nabla_{x_t} \log p_t(x_t|x_1)\|^2$

Output: v^θ

$p_t(x_t|x_1)$ general
$p(x_0)$ is general

$p_t(x_t|x_1)$ closed-form from of SDE $dx_t = f_t dt + g_t dw$

- *Variance Exploding*: $p_t(x|x_1) = \mathcal{N}(x|x_1, \sigma_{1-t}^2 I)$
- *Variance Preserving*: $p_t(x|x_1) = \mathcal{N}(x|\alpha_{1-t}x_1, (1-\alpha_{1-t}^2)I)$

$$\alpha_t = e^{-\frac{1}{2}T(t)}$$

$p(x_0)$ is Gaussian
$p_0(\cdot|x_1) \approx p$

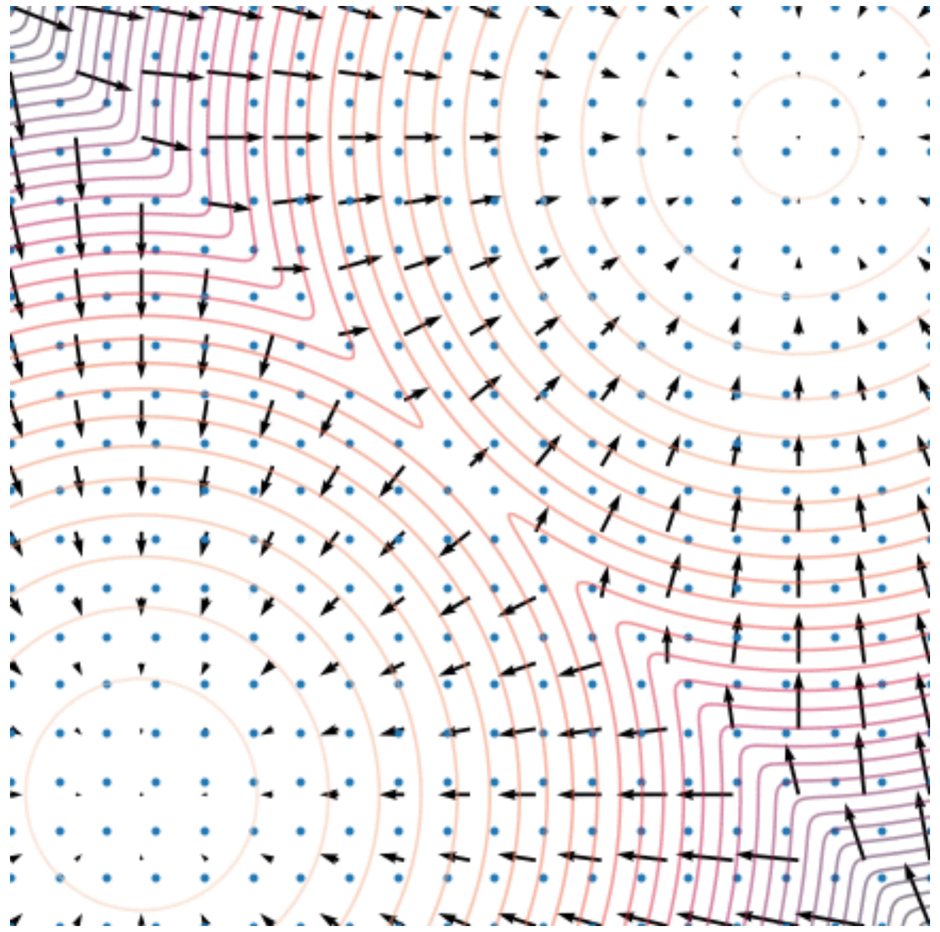
Marginal Flow ~ Score Function

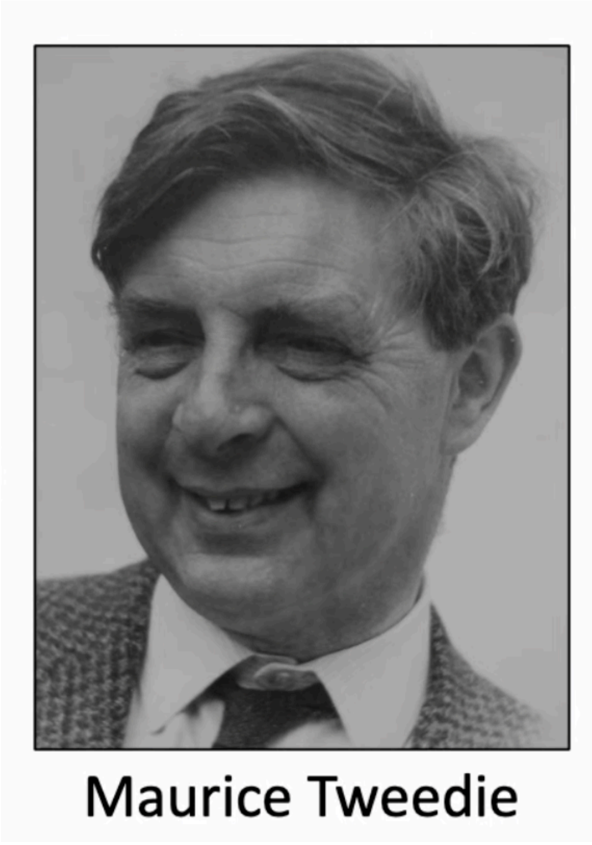
- Both Flow Matching and Diffusion Models aim to predict the expectation of denoised data given some noisy sample $\mathbb{E}[x_1 \mid x_t]$

- Tweedie's Formula says this recovers the score function, the gradient of the log likelihood. We can climb this gradient SGD-style to arrive at a sample.

$$\mathbb{E}[x_1 \mid x_t] = x_t + \sigma_t^2 \nabla_{x_t} \log p_t(x_t)$$

- Flow Matching essentially generalizes the score matching concept





Maurice Tweedie

Lipman et al. 2022

Flow Matching Perspective

Advantages

- Generalization of Diffusion Models and Continuous Normalizing Flow
- The noise process can be anything as long as boundaries are set
- Any source distribution can be used
- The steps are continuous
- The training method is simulation free (as opposed to CNF variants)

The Marginalization Trick

Theorem*: The marginal velocity generates the marginal probability path.

$$u_t(x) = \mathbb{E}\left[u_t(X_t | X_1) \mid X_t = x\right] \qquad p_t(x) = \mathbb{E}_{X_1} p_{t|1}(x | X_1)$$

$$u_t(x) = \int u_t(x | x_1) \frac{p_t(x | x_1) q(x_1)}{p_t(x)} dx_1$$

$$p_t(x) = \int p_t(x | x_1) q(x_1) dx_1$$

$$p_t(x) = \Sigma_{x_1} p_t(x | x_1) q(x_1)$$

"Flow Matching for Generative Modeling" Lipman el al. (2022)

"Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow" Liu et al. (2022)

"Building Normalizing Flows with Stochastic Interpolants" Albergo et al. (2022)

Geometric Intuition

Path from x_t to x_0

$$u_t(x_t) = \sum_{x_0} u_t(x_t \mid x_0) \frac{p_t(x_t \mid x_0)}{p_t(x_t)} q(x_0)$$

Marginal Flow

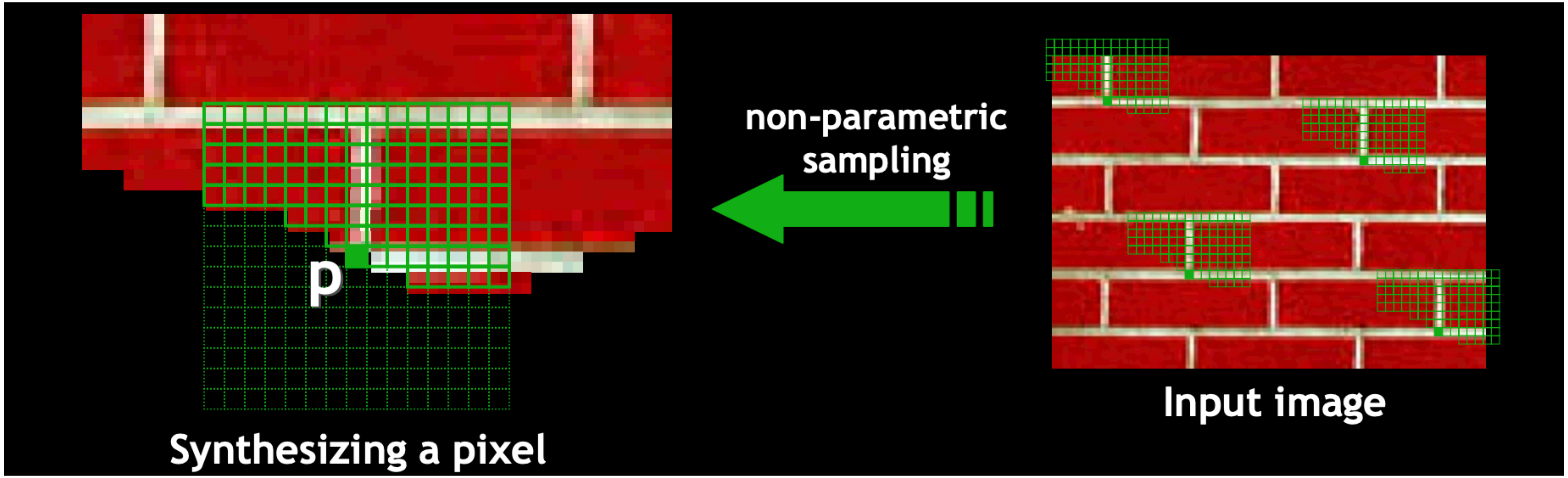
Path Weight

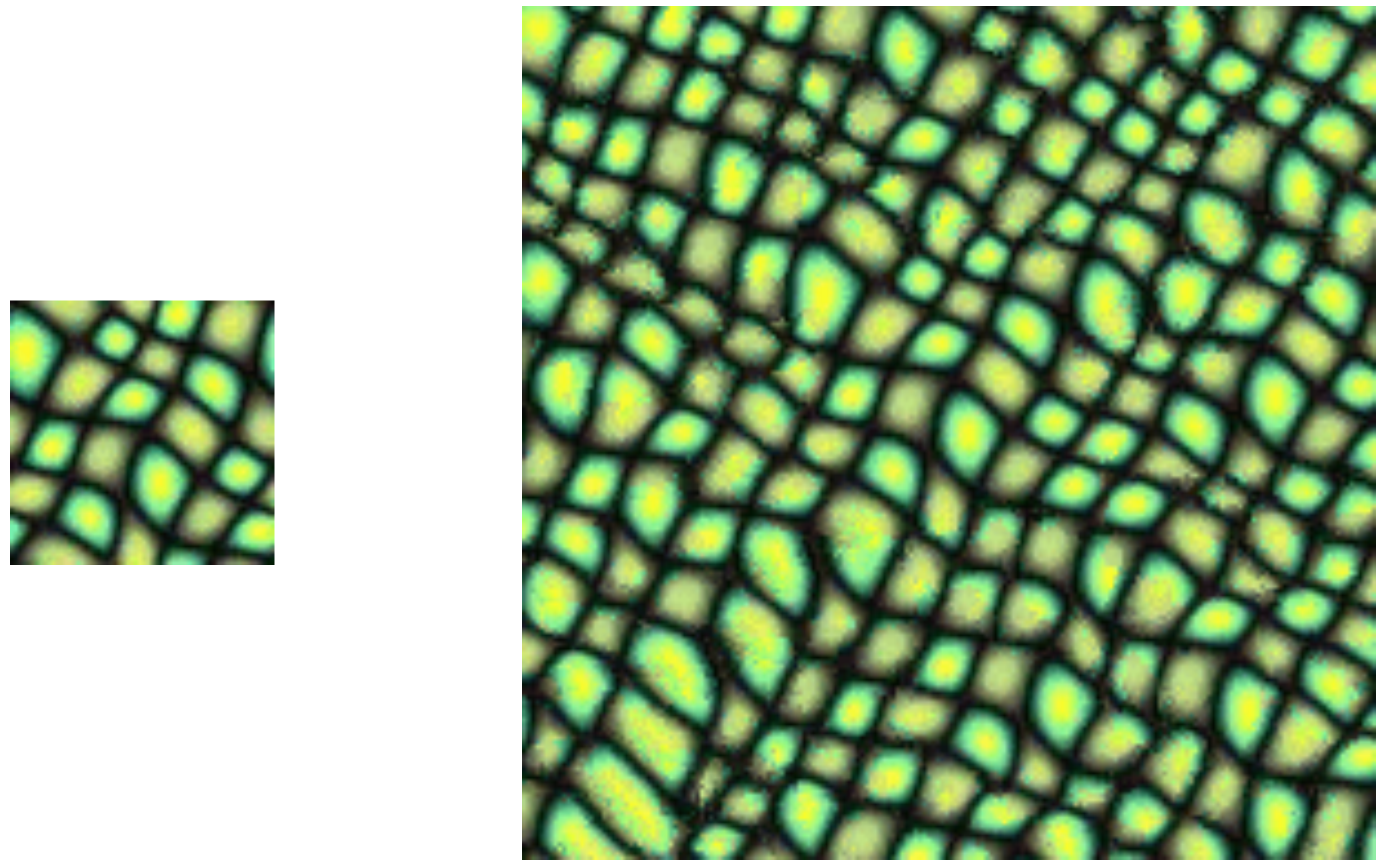
Just a weighted average of the flow to each data sample!!!!!

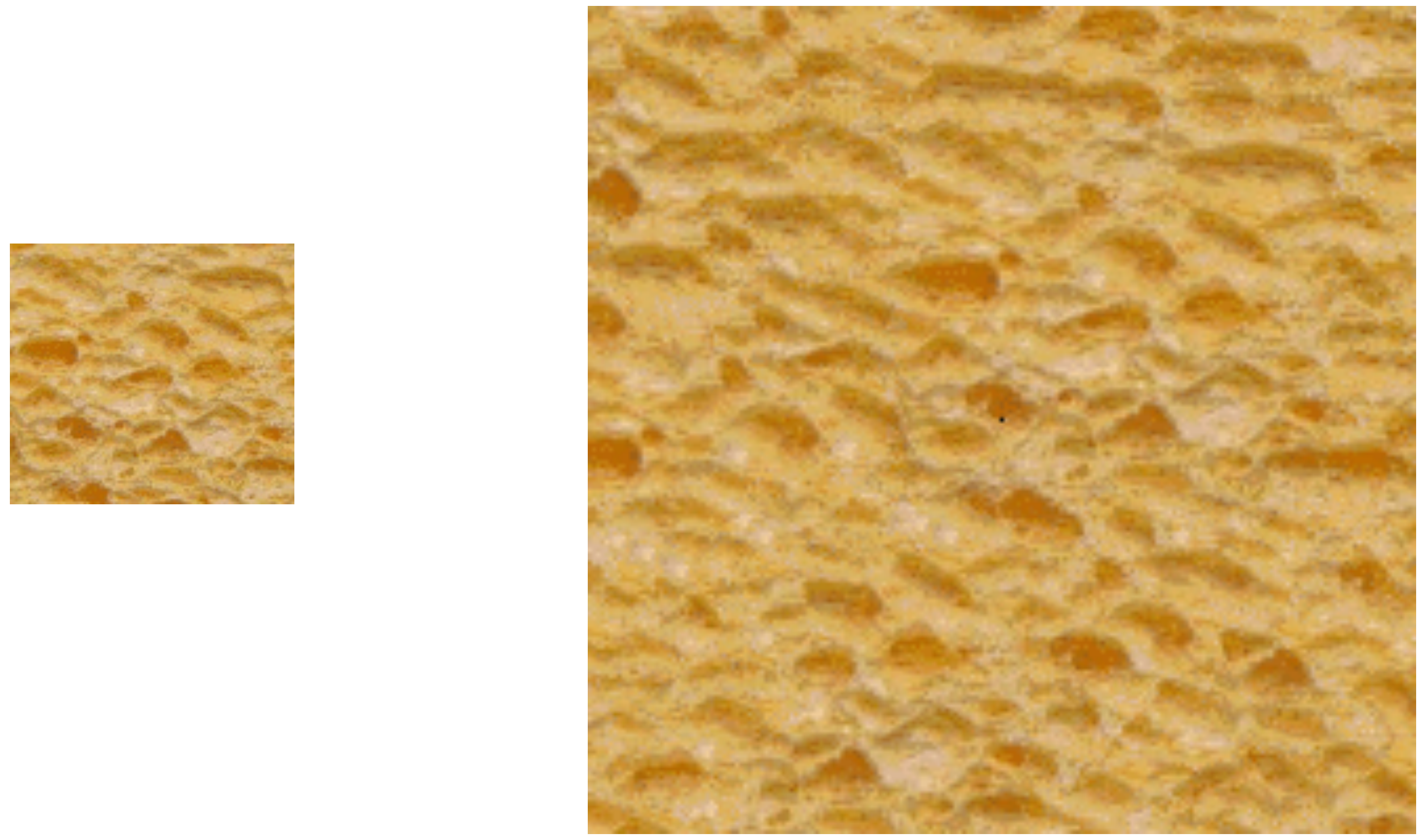
You can actually do this non-parametrically.

See interactive visualization at https://decentralizeddiffusion.github.io/

Efros & Leung ICCV'99







- Non-parametric patch-based NN sampling to fill in missing details & generate textures

SIGGRAPH2007 Computer Graphics Proceedings, Annual Conference Series, 2007

Scene Completion Using Millions of Photographs

James Hays Alexei A. Efros
Carnegie Mellon University

Original Image Input Scene Matches Output

Figure 1: Given an input image with a missing region, we use matching scenes from a large collection of photographs to complete the image.

- Non-parametric patch-based NN approach to fill in missing details with **lots of Data!**

Key message

- One can minimize the diffusion objective (marginal flow) non-parametrically and perfectly minimize the loss.
- But there is no learning! No ability to generate new images!
- i.e. Exactly minimizing this objective does not guarantee interpolation/compositionally, learning of the image manifold!
- Parametrizing it with neural networks results in magic smoothing to generate new images and interpolate between them. Exactly what makes this possible still active area of research