Deep Learning

Angshuman Paul

Assistant Professor

Department of Computer Science & Engineering

Diffusion Probabilistic Models

The Motivation

https://www.istockphoto.com/photos/stone-carving

The Motivation

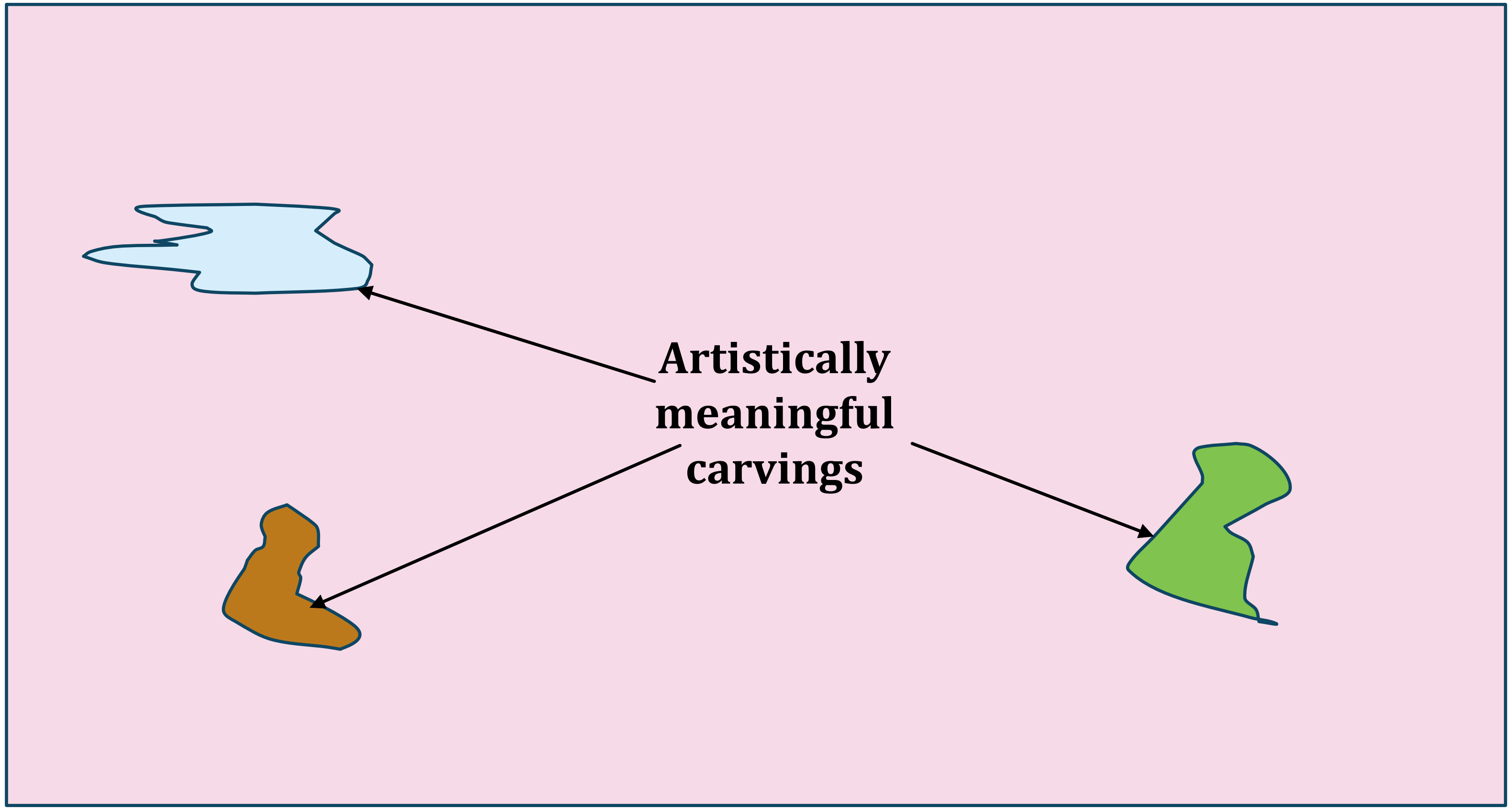
By carving the same stone differently, I will get different objects

https://www.istockphoto.com/photo/classic-antique-marble-columns-set-gm1301800814-393738579

The Motivation

Among all possible carvings, most of the carvings will produce meaningless objects

All possible carvings



All possible carvings

Stone
(Every side is similar)

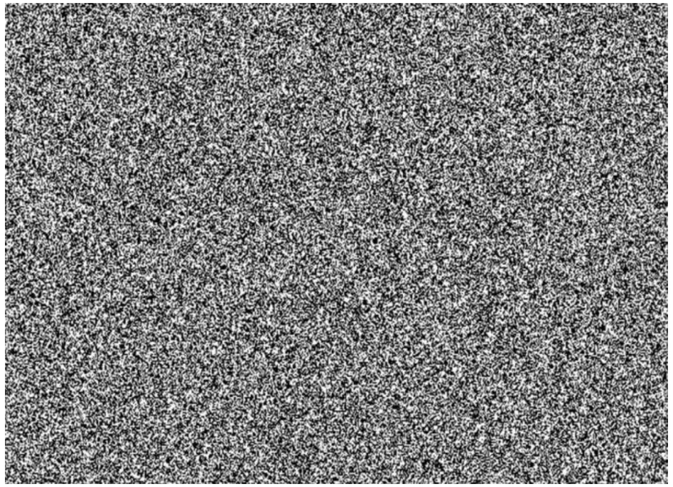
Stone
(Every side is similar)

Artist
(Carving)

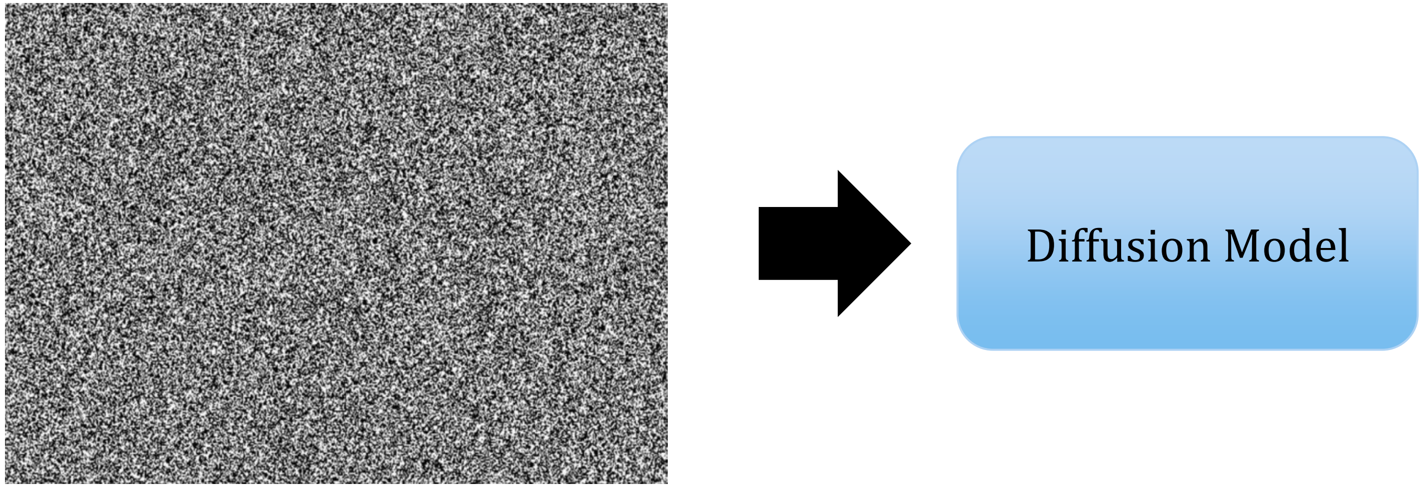
Stone
(Every side is similar)

Artist
(Carving)

Artwork



Noise
(Homogeneous)



Noise
(Homogeneous)

Artist

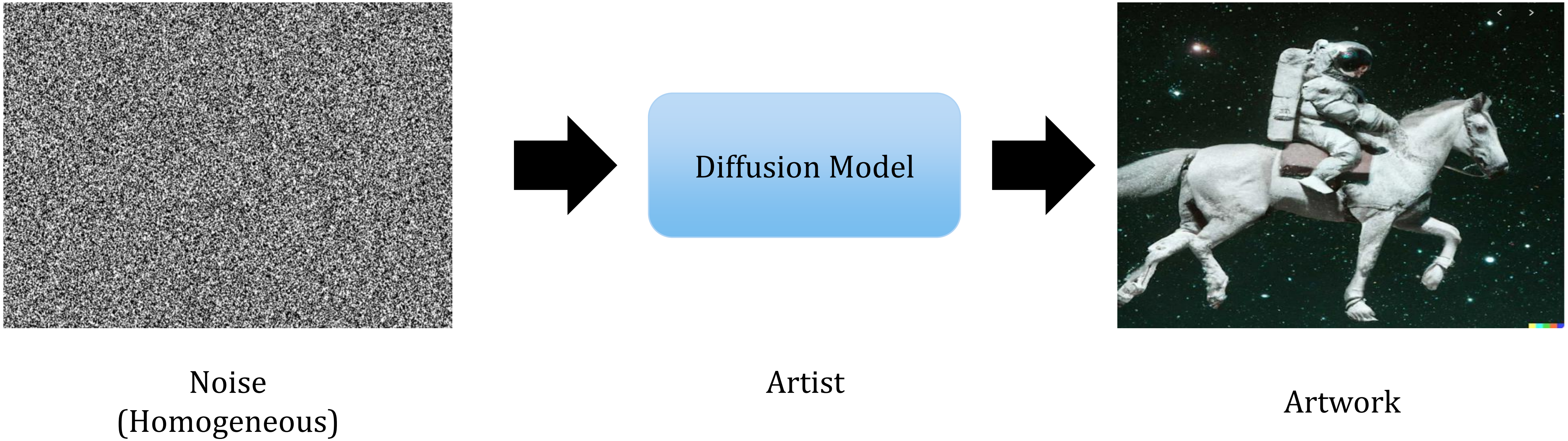

Diffusion Model
Noise
(Homogeneous)
Artist
Artwork

Image Space

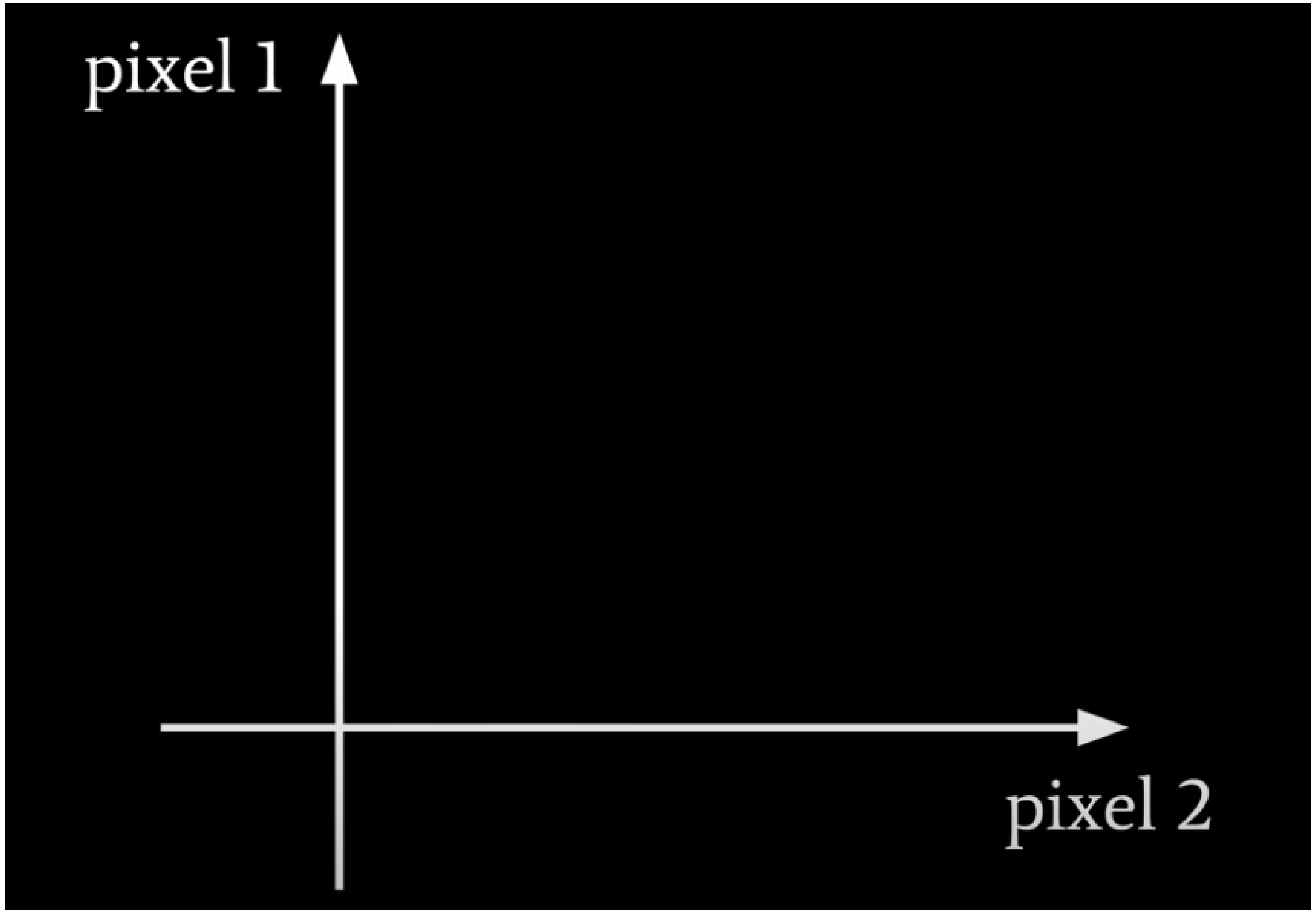
Space of all possible images

Image Space

Space of all possible images

Consider two-pixel images

Each axis represents the value of a pixel in a two-pixel image



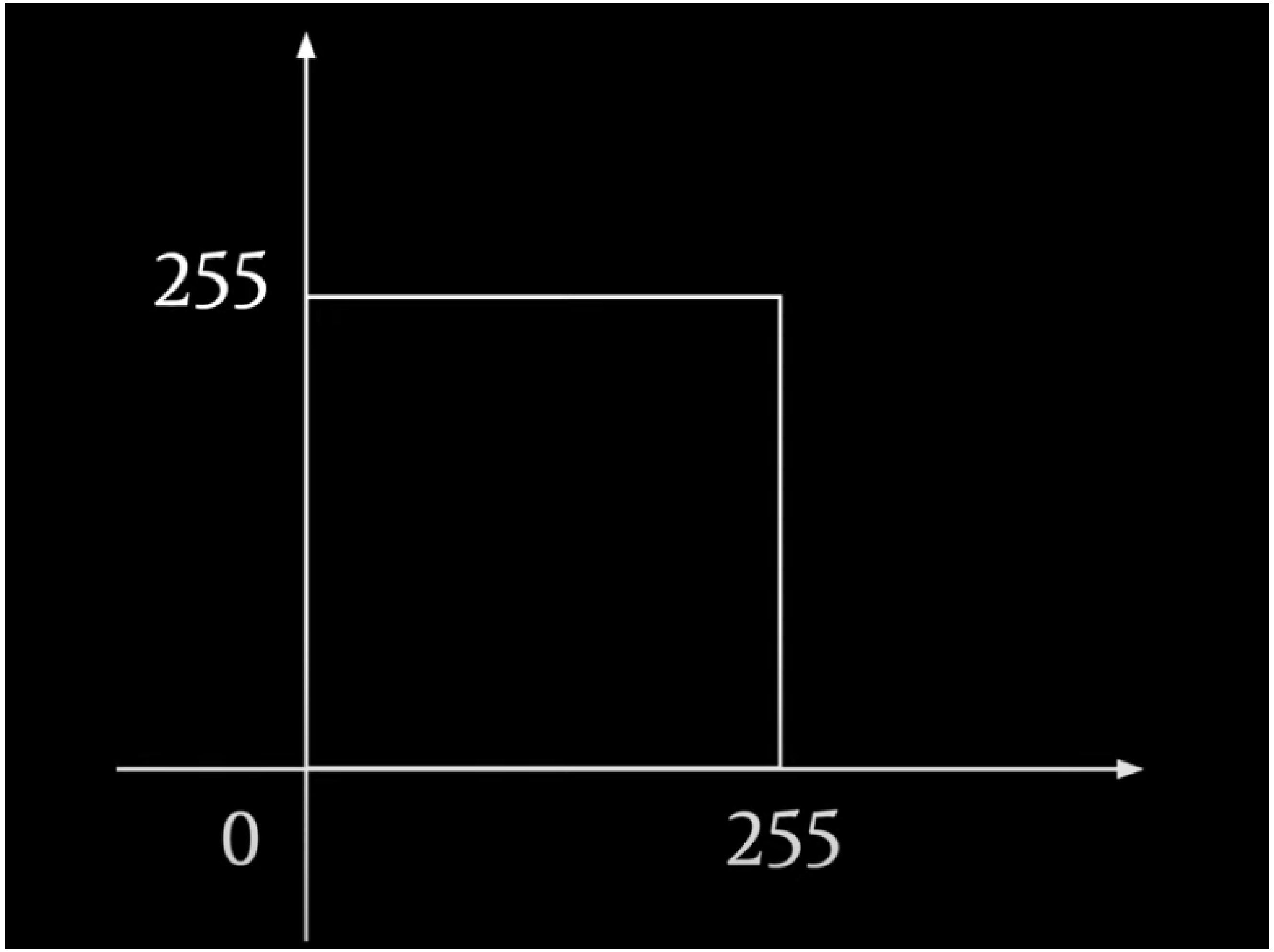
https://www.youtube.com/watch?v=1pgiu--4W3I&list=PLFs-r3e2iBRG_QOJLQ-_eSdHmlBEP_VK_&index=7

Image Space

Space of all possible images

Consider two-pixel images

Each axis represents the value of a pixel in a two-pixel image



https://www.youtube.com/watch?v=1pgiu--4W3I&list=PLFs-r3e2iBRG_QOJLQ-_eSdHmlBEP_VK_&index=7

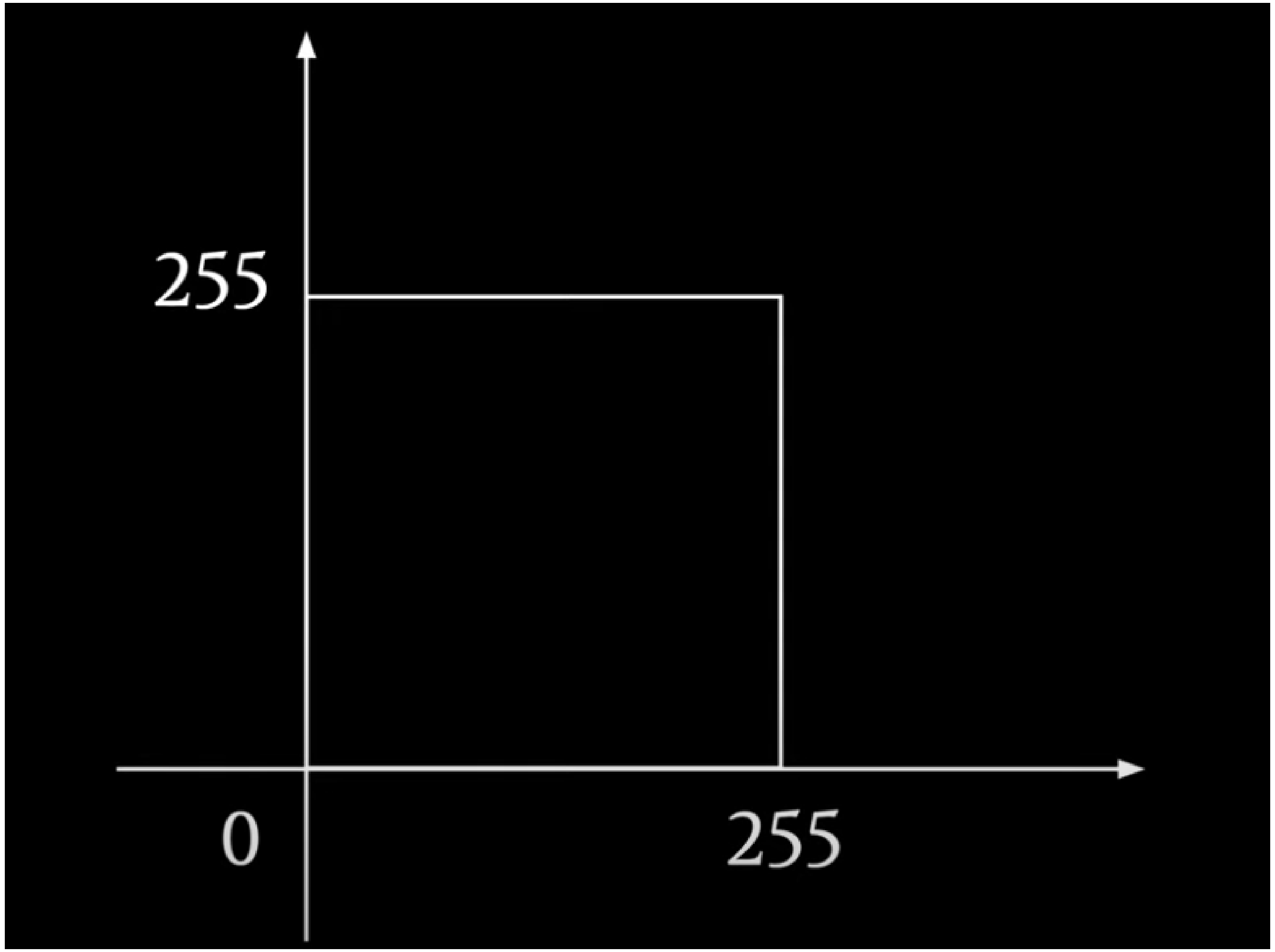
Image Space

Space of all possible images

Consider two-pixel images

Each axis represents the value of a pixel in a two-pixel image

If we talk about 10×10 images, we have to consider 100-dimensional plots



https://www.youtube.com/watch?v=1pgiu--4W3I&list=PLFs-r3e2iBRG_QOJLQ-_eSdHmlBEP_VK_&index=7

Image Space

All possible two-pixel images live in this space

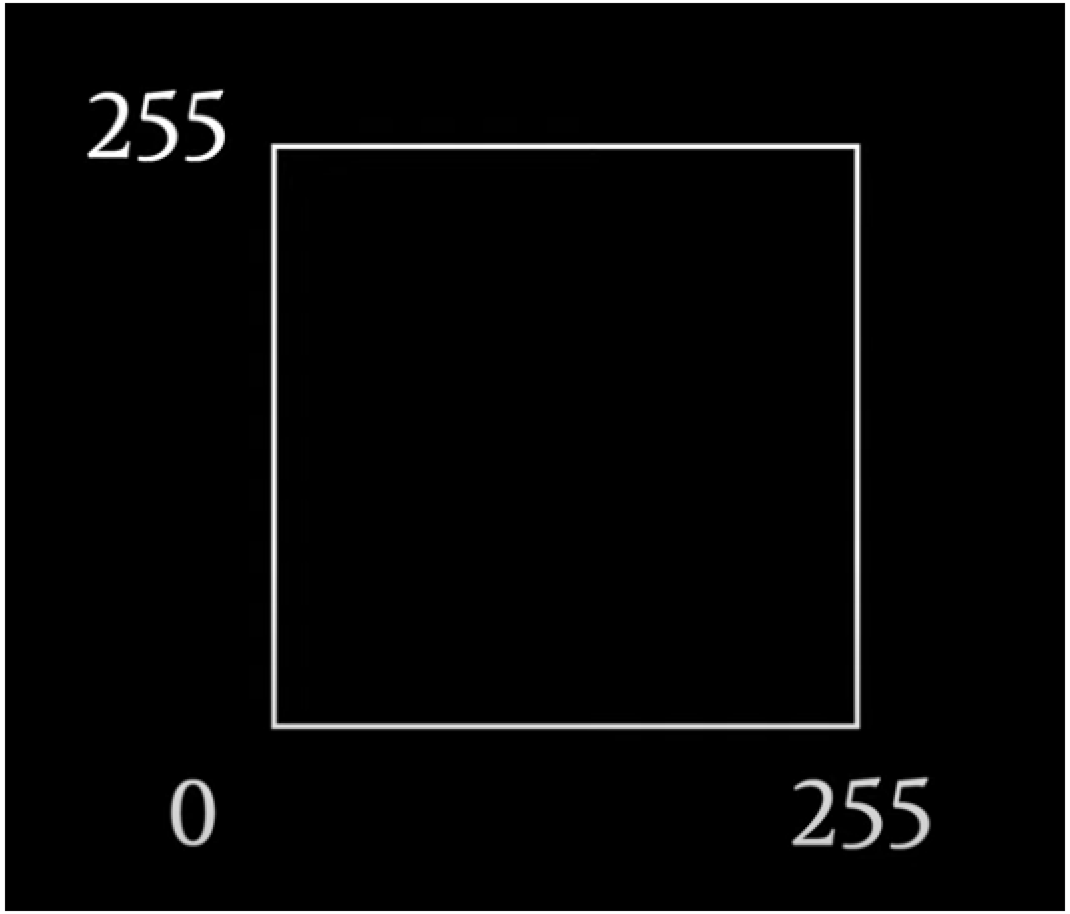


Image Space

Each point in this space will represent one image

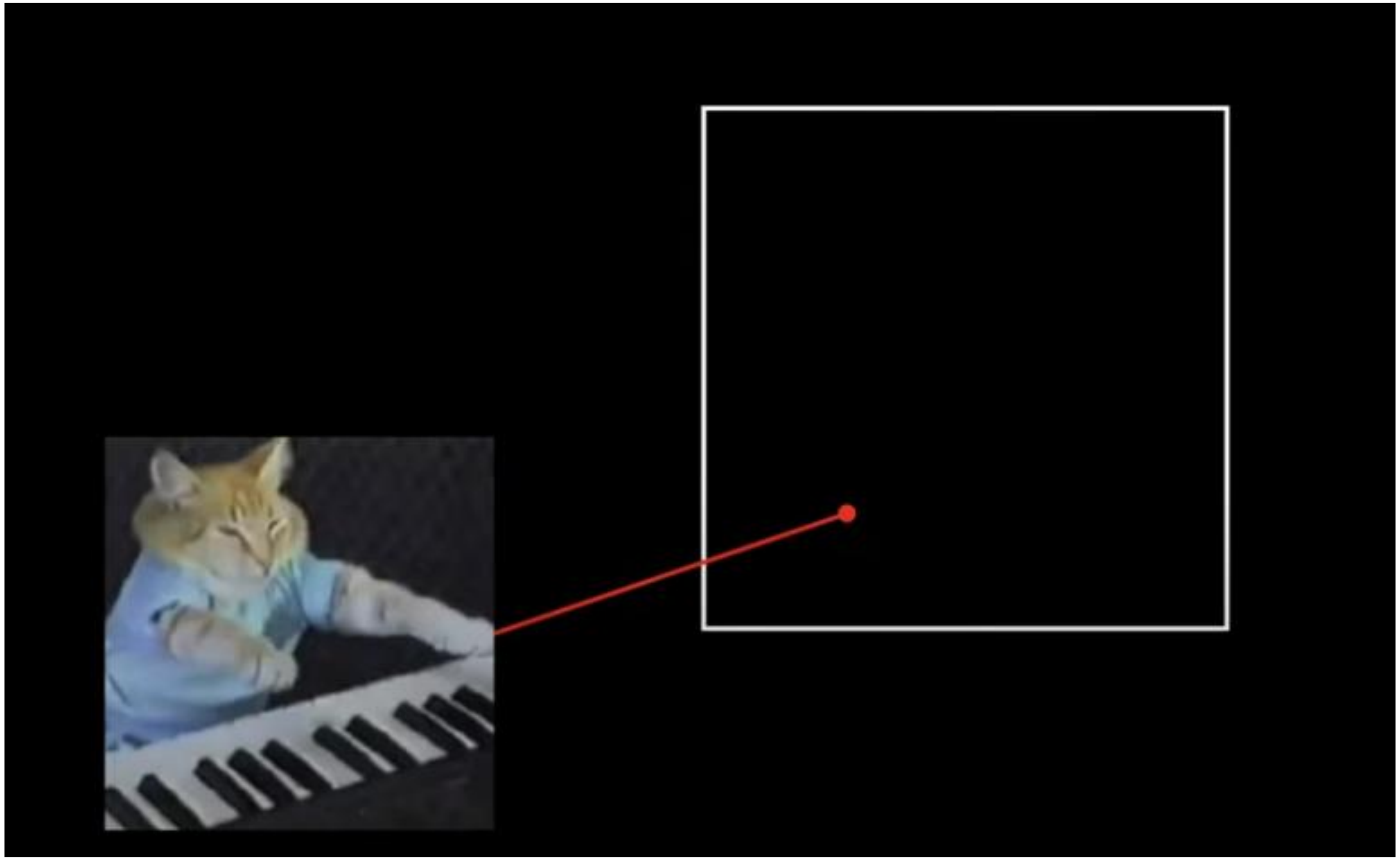


Image Space

Each point in this space will represent one image

The image may be a meaningful image or noise

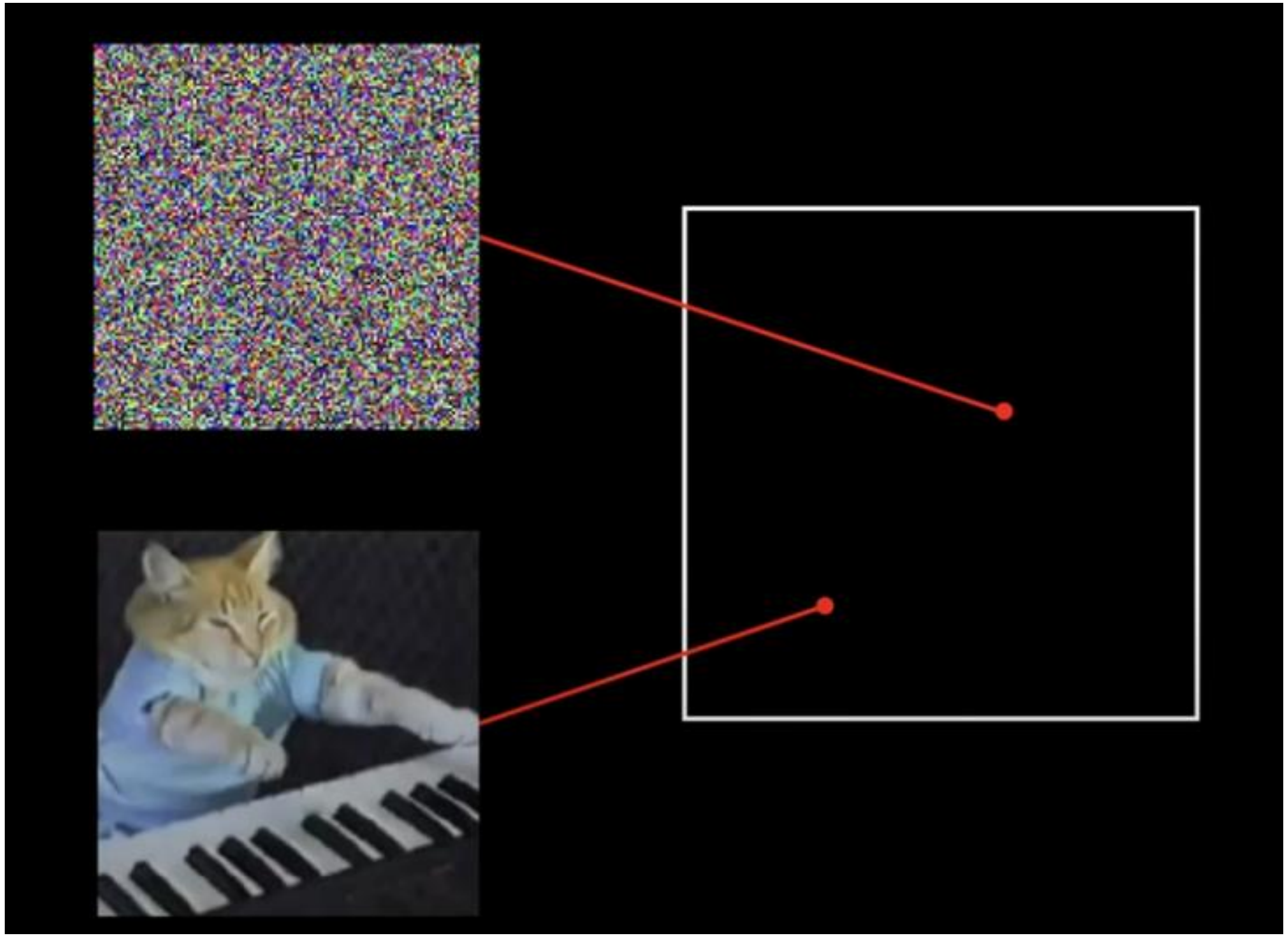
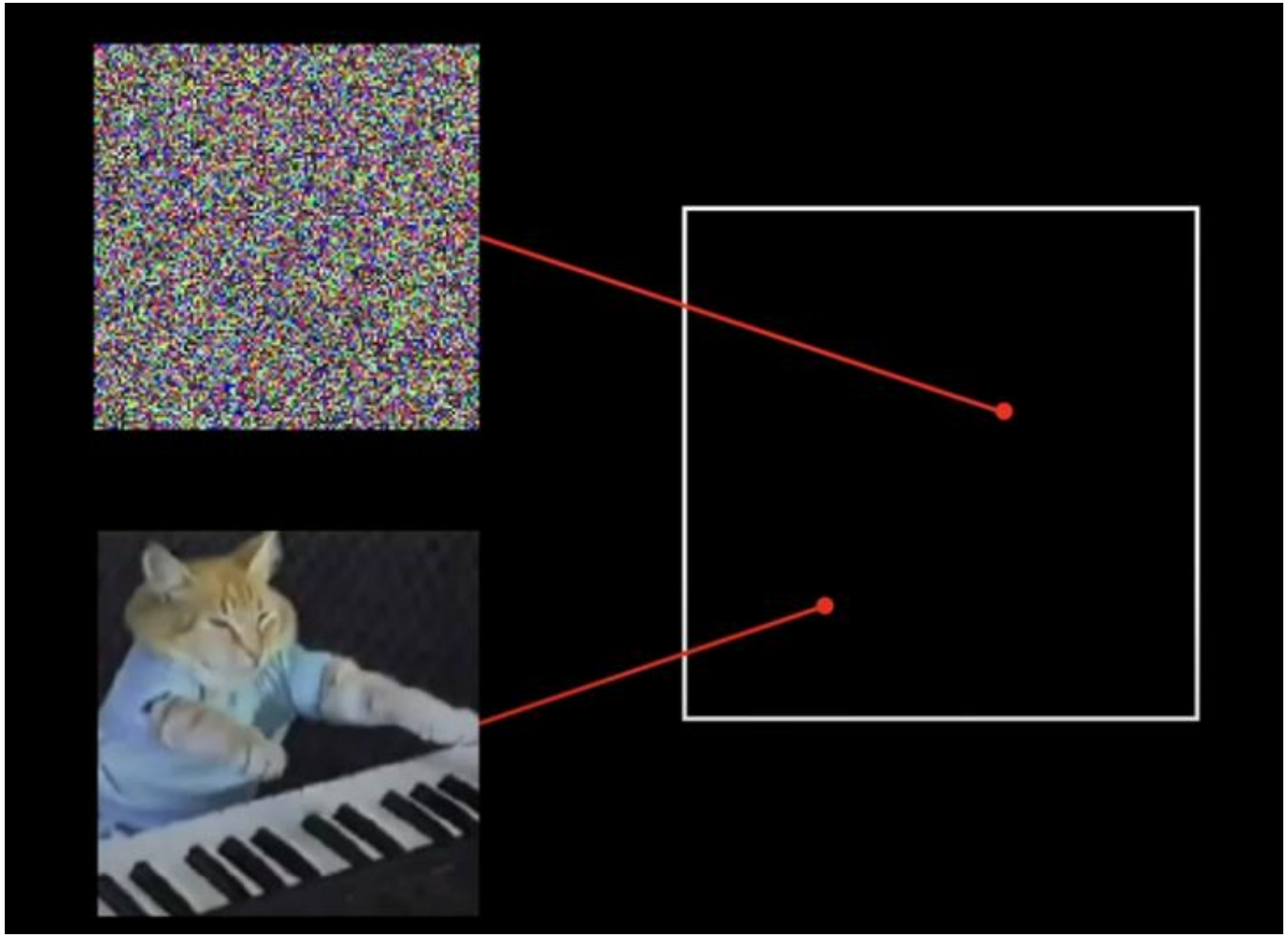


Image Space

Each point in this space will represent one image

The image may be a meaningful image or noise

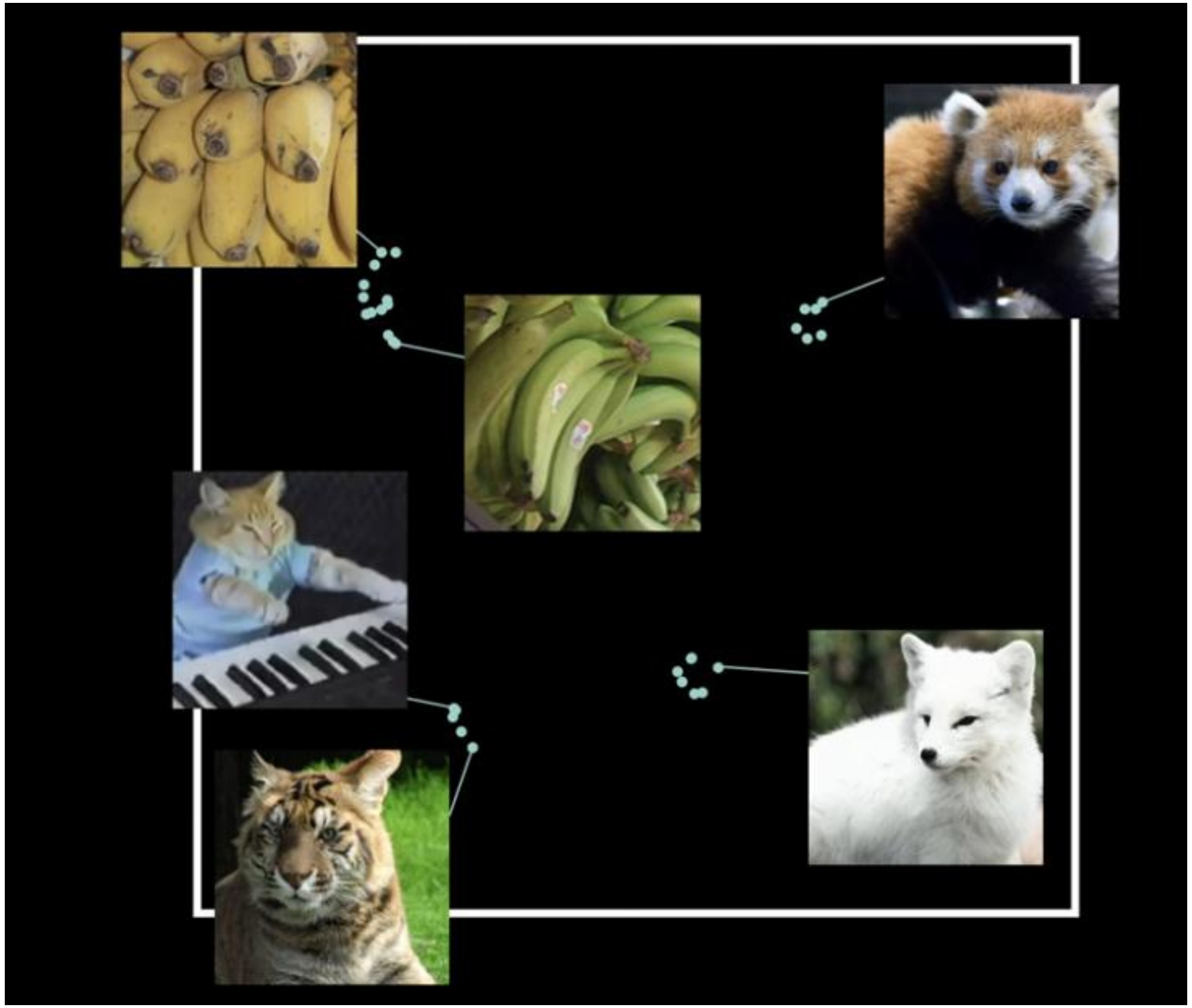


Designing an Image Generator using the Idea of Image Space

Take a dataset of good images

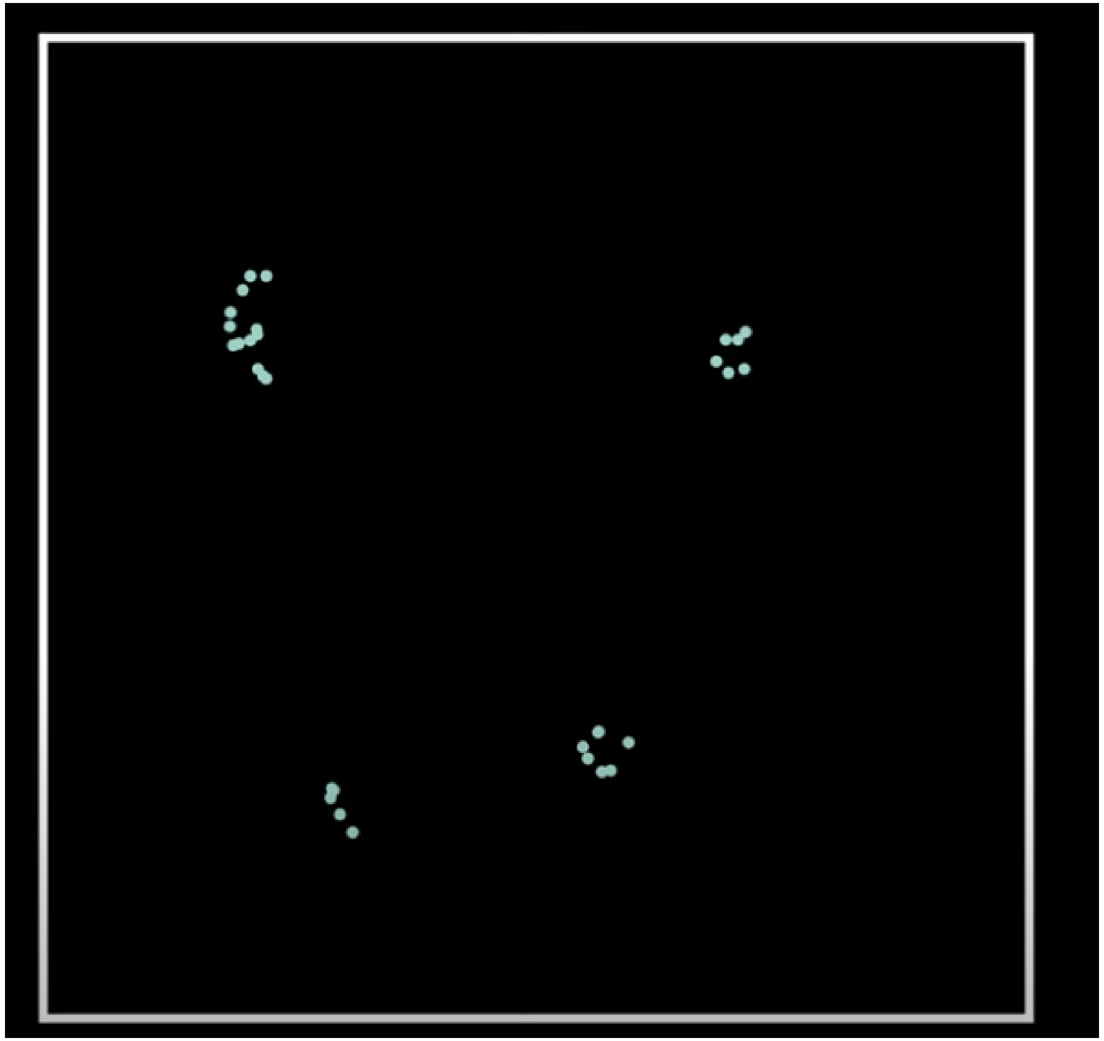
See where those good images live in image space

Designing an Image Generator using the Idea of Image Space



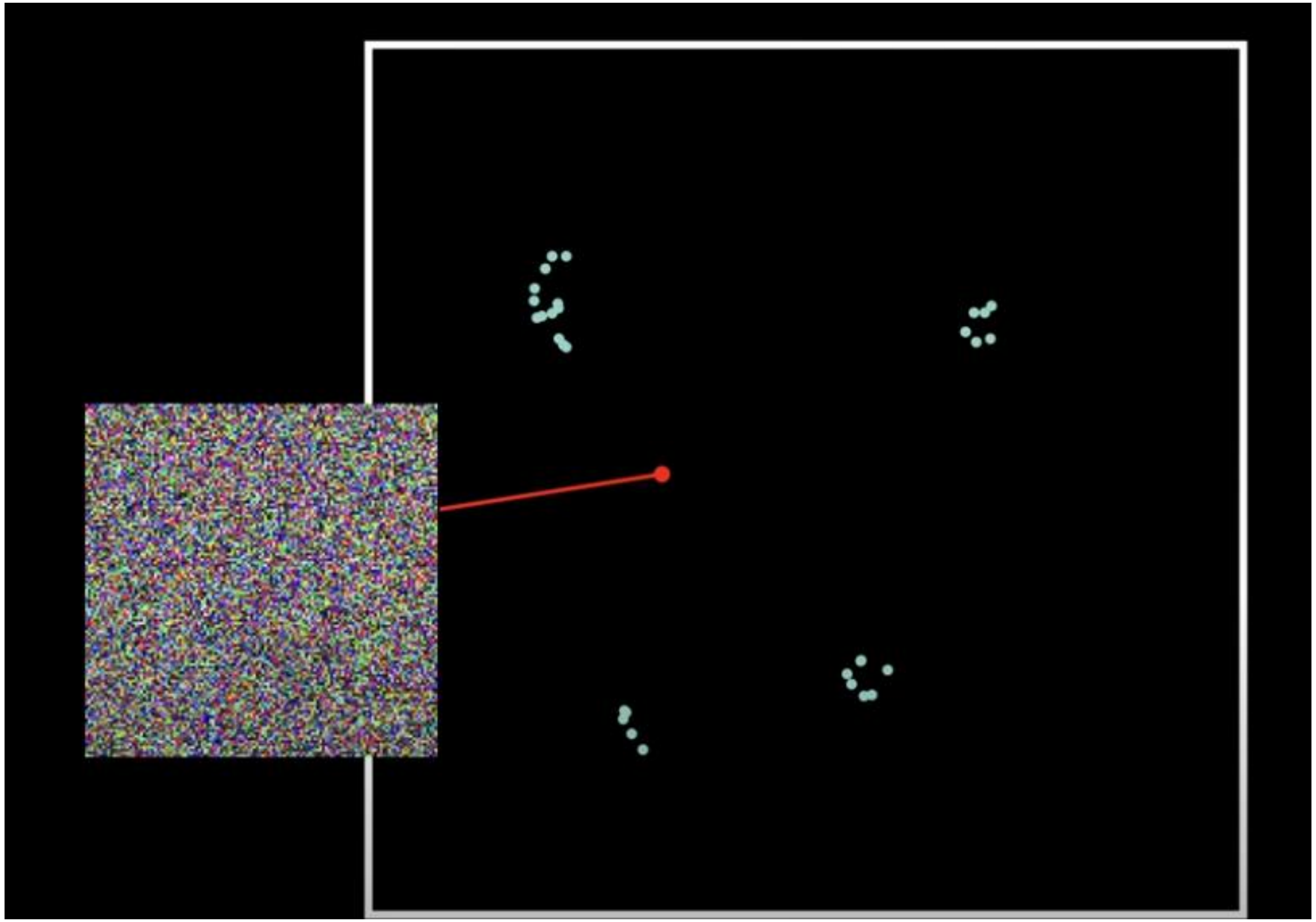
Designing an Image Generator using the Idea of Image Space

We find that a vast majority of the image space is empty, i.e., good images live only in very few places in the image space



Designing an Image Generator using the Idea of Image Space

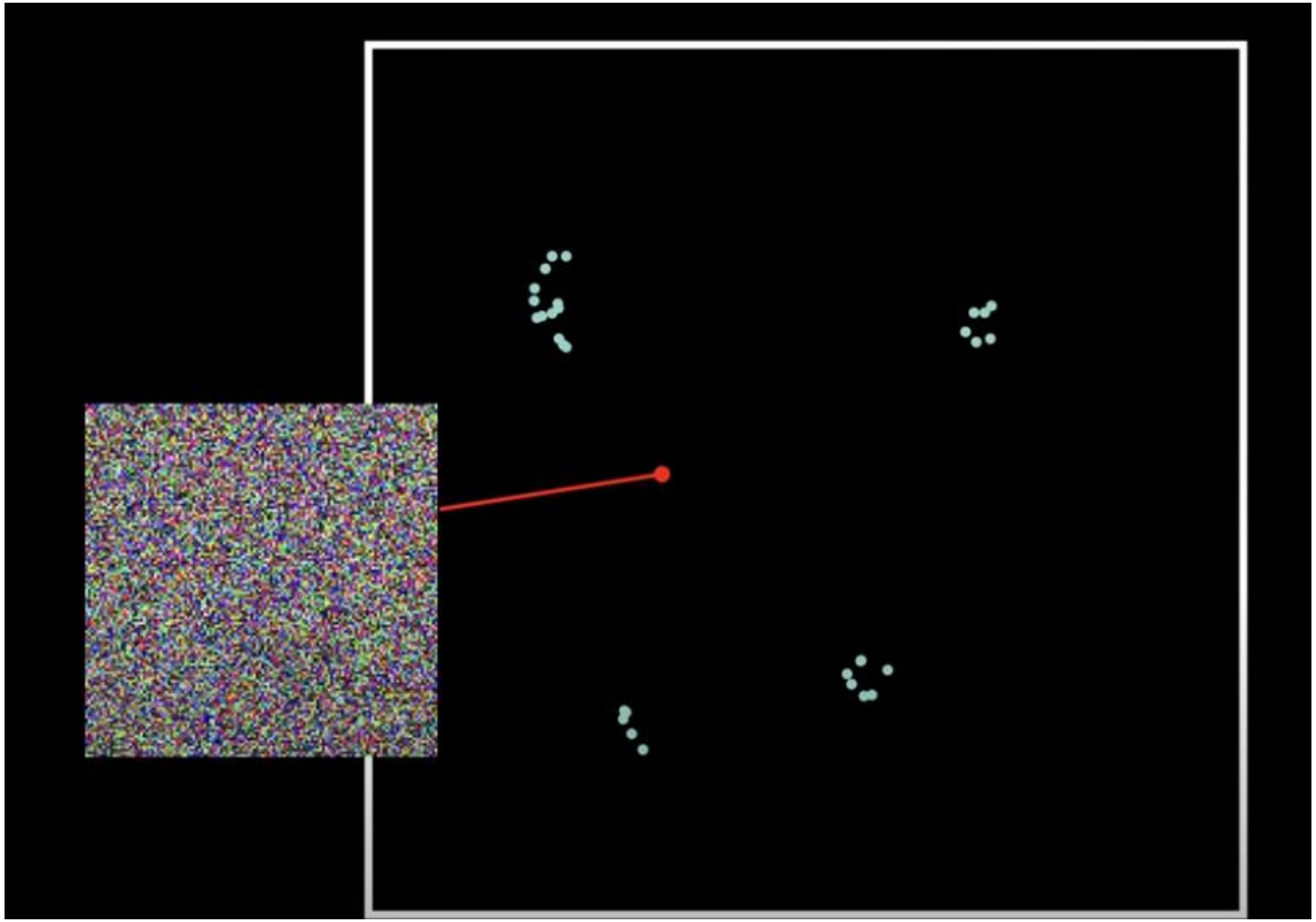
In the rest of the places,
the images are noise



Designing an Image Generator using the Idea of Image Space

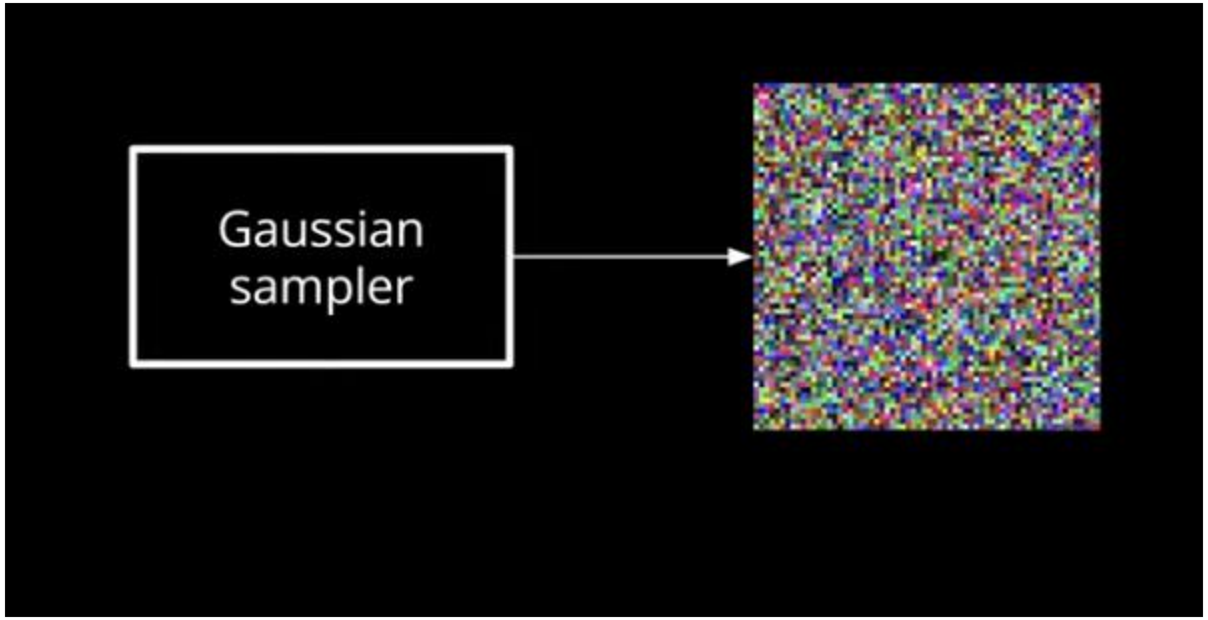
Most of the regions in the image space are empty (only noise)

Similar images reside closely in the image space



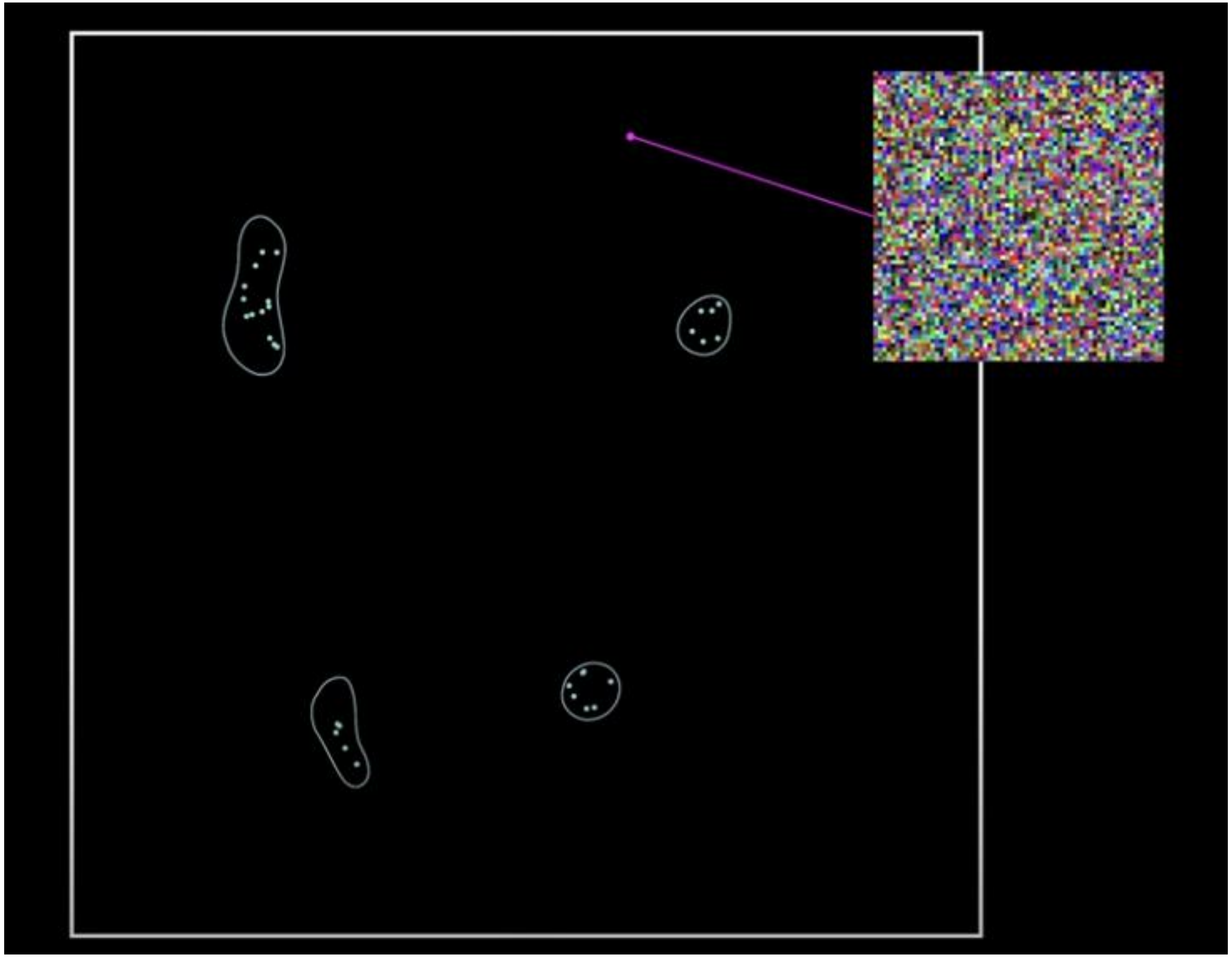
Diffusion Models: The Basic Intuition

Start with a noise sample



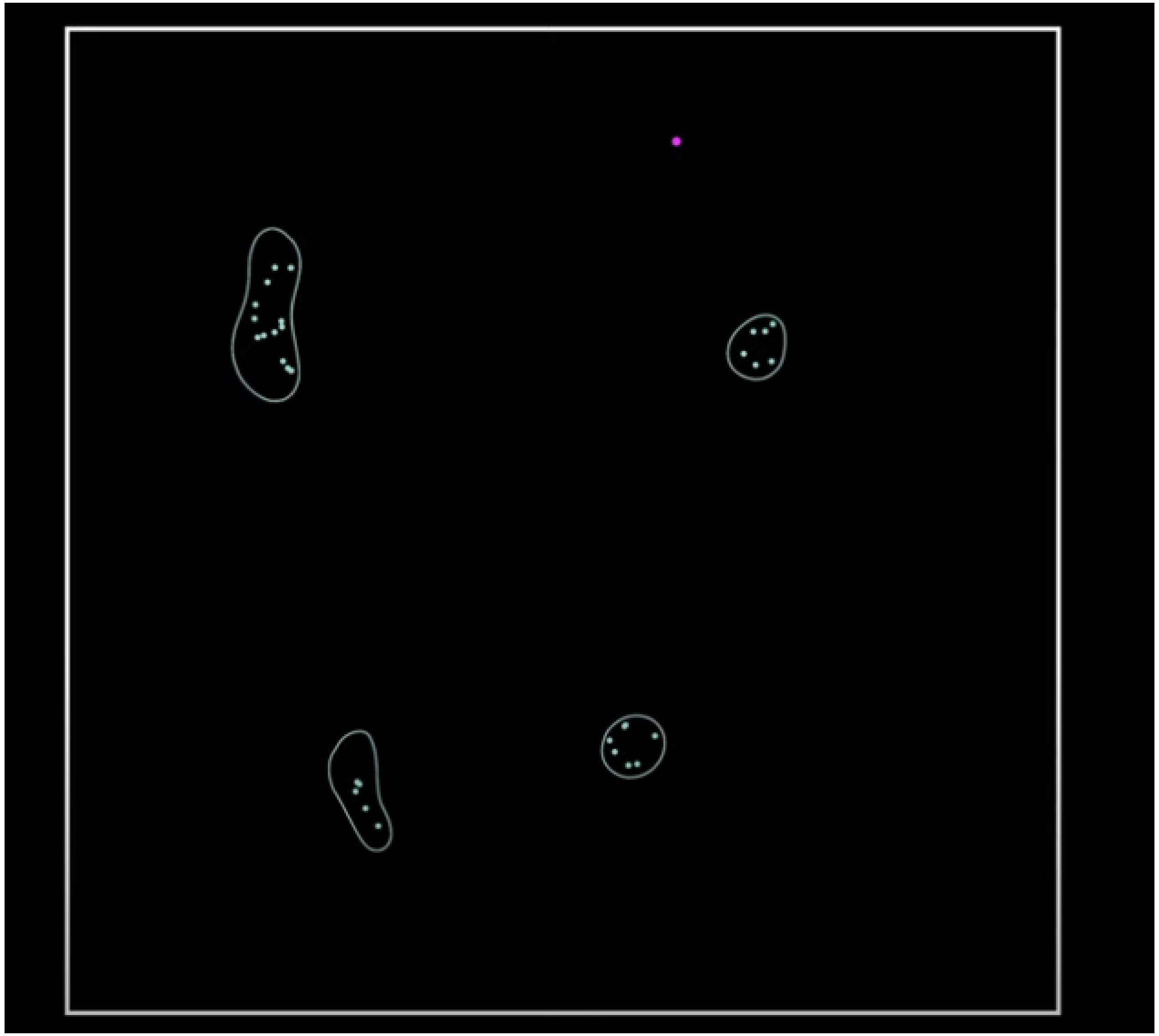
Diffusion Models: The Basic Intuition

Start with a noise sample



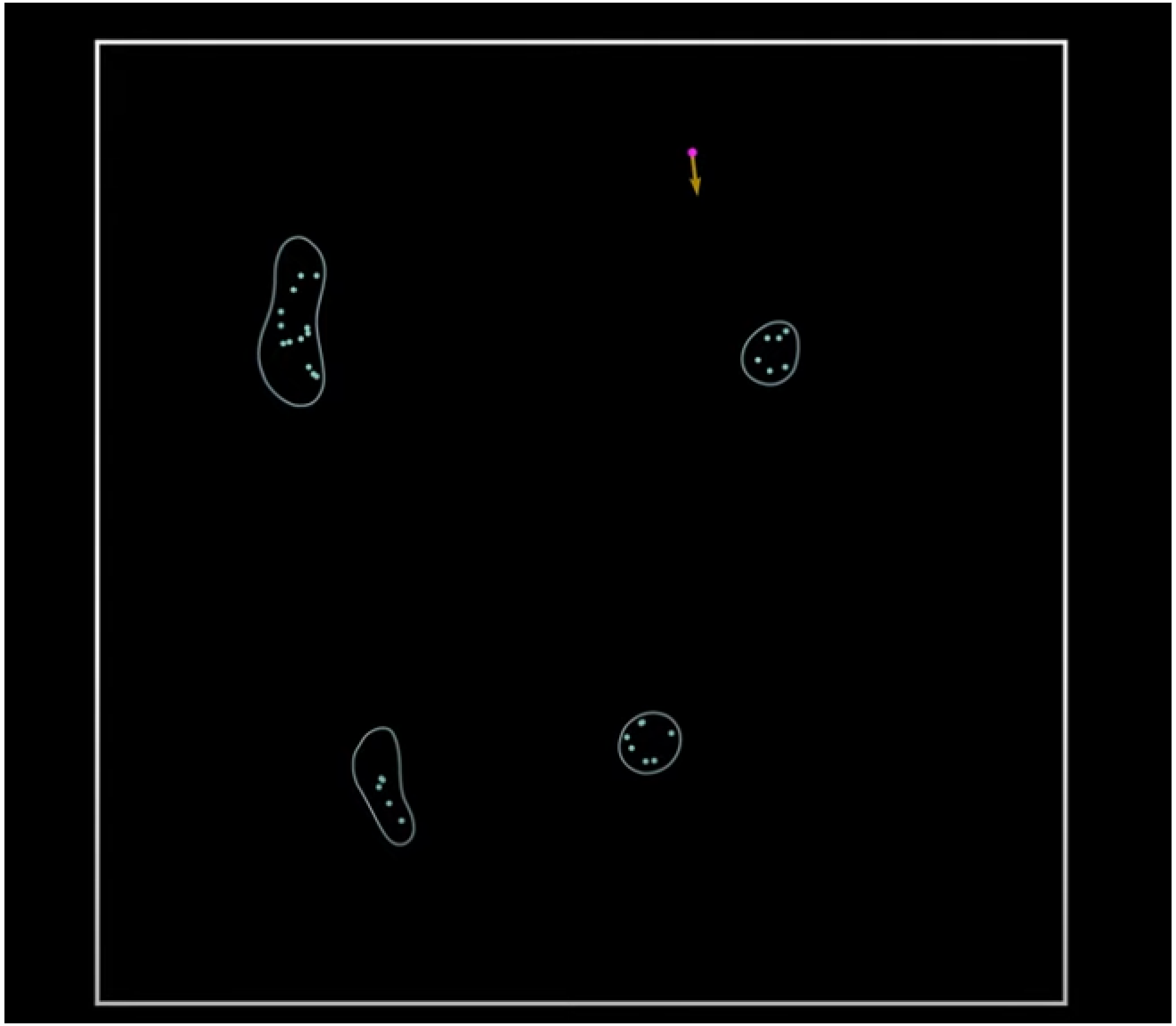
Diffusion Models: The Basic Intuition

Then move to the direction of (closest) image cluster



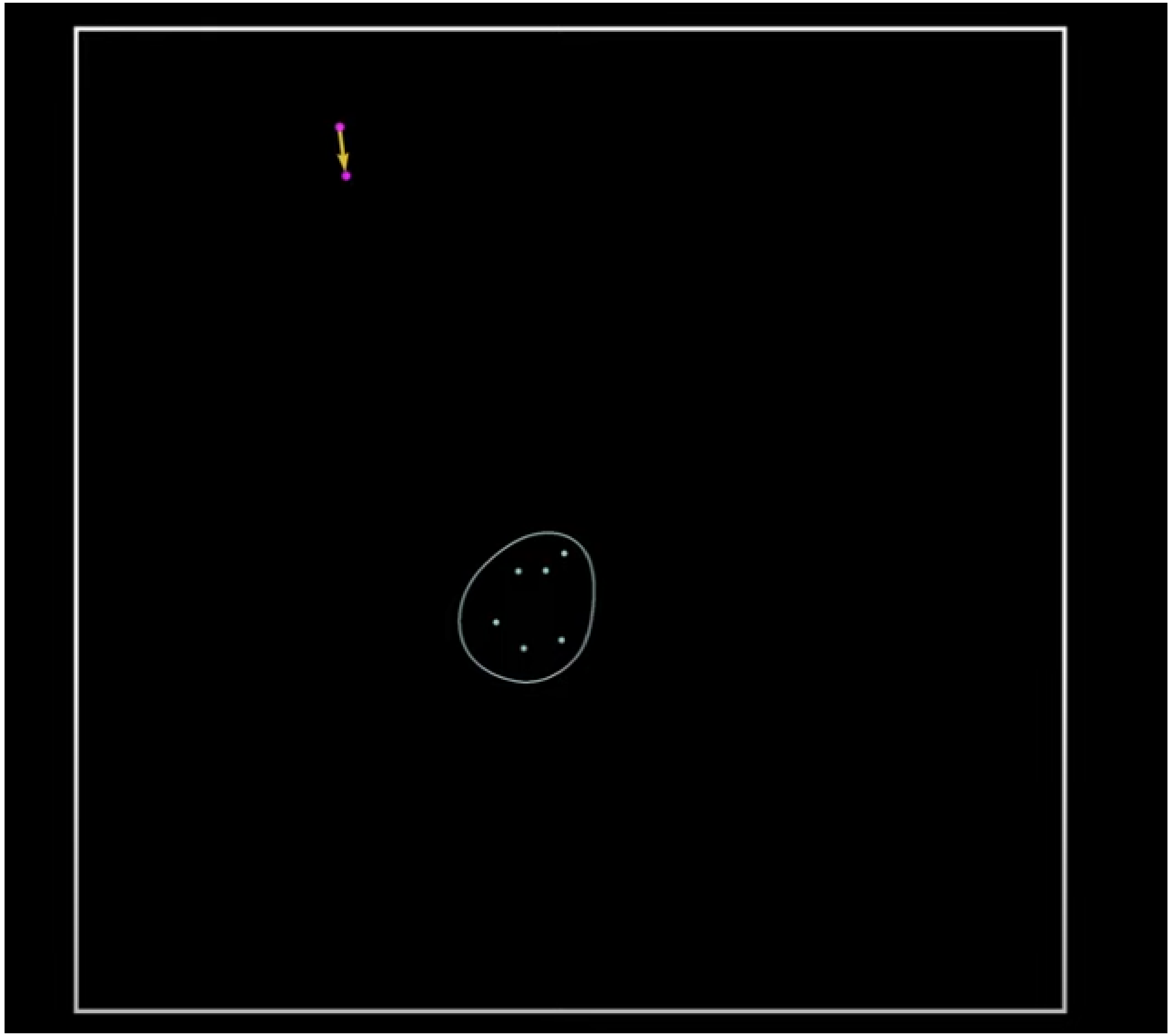
Diffusion Models: The Basic Intuition

Then move to the direction of (closest) image cluster



Diffusion Models: The Basic Intuition

Then move to the direction of (closest) image cluster

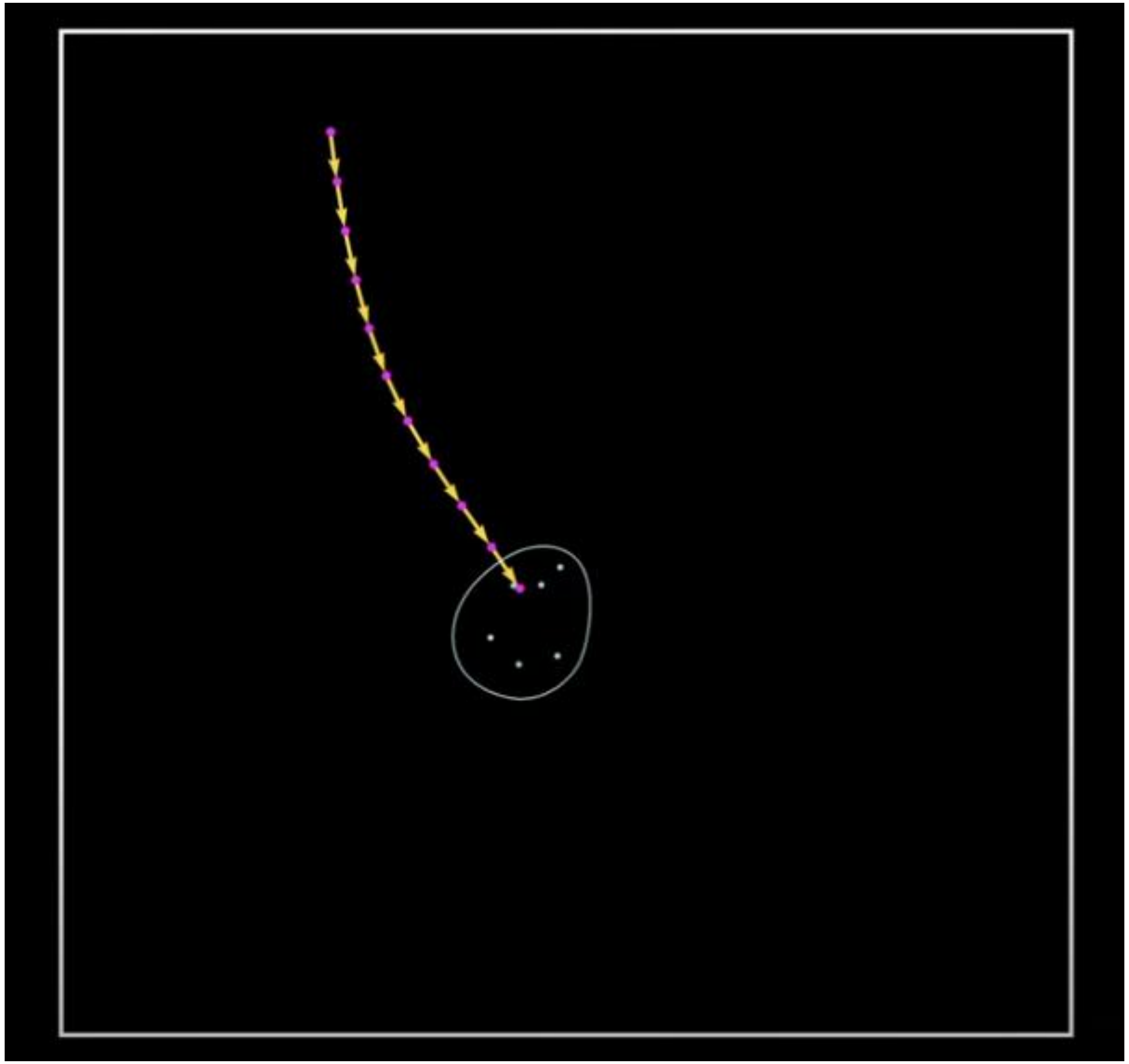


Diffusion Models: The Basic Intuition

Then move to the direction of (closest) image cluster

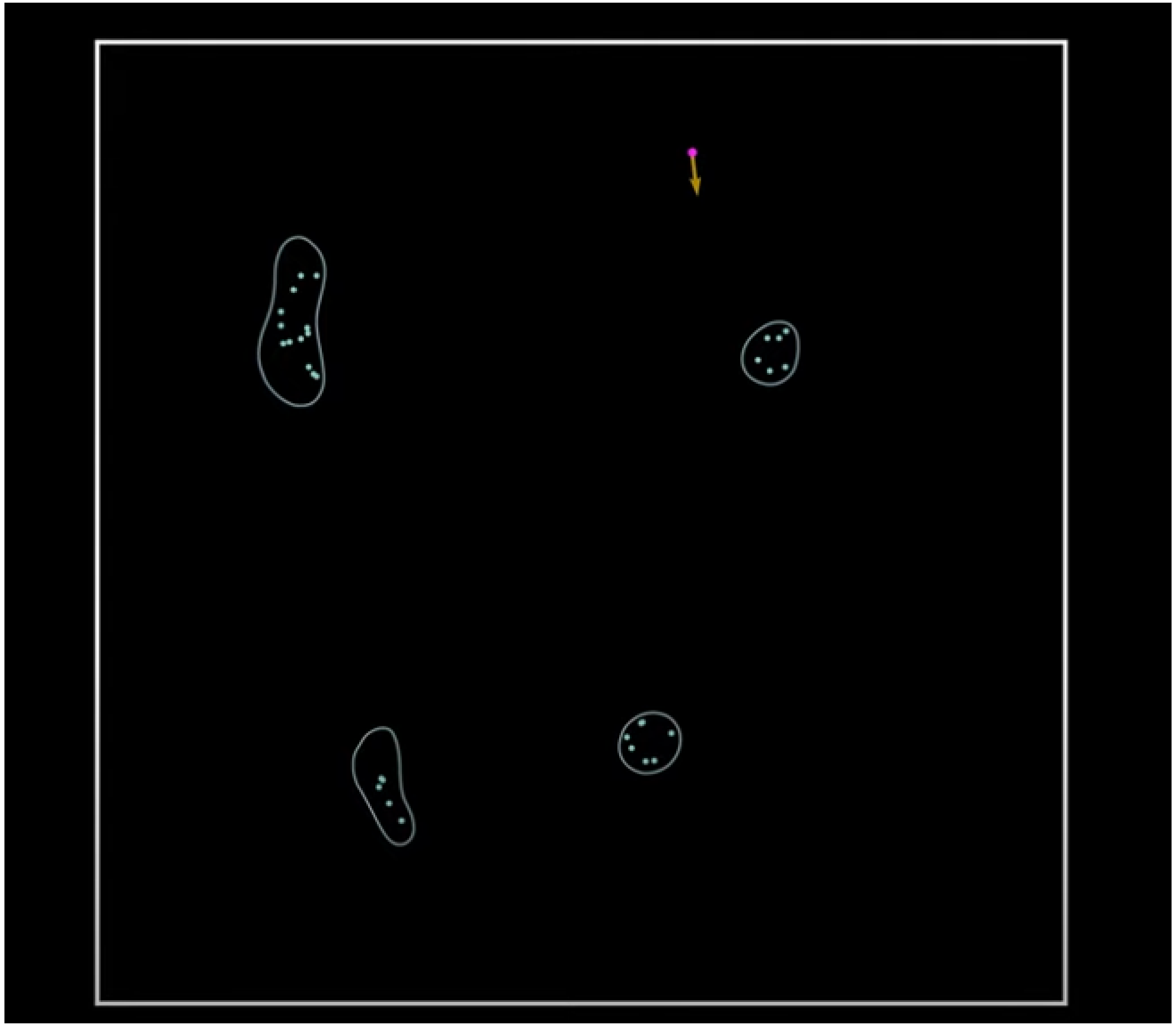
Diffusion Models: The Basic Intuition

Then move to the direction of (closest) image cluster



Diffusion Models: The Basic Intuition

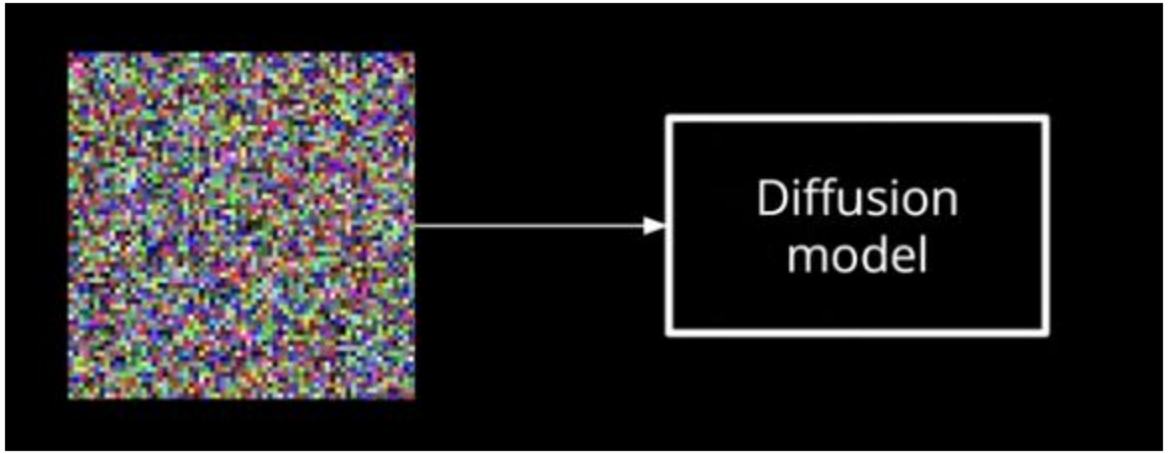
How to find out the direction of the movement from the initial position?



Diffusion Models: The Basic Intuition

How to find out the direction of the movement from the initial position?

Diffusion model does exactly this

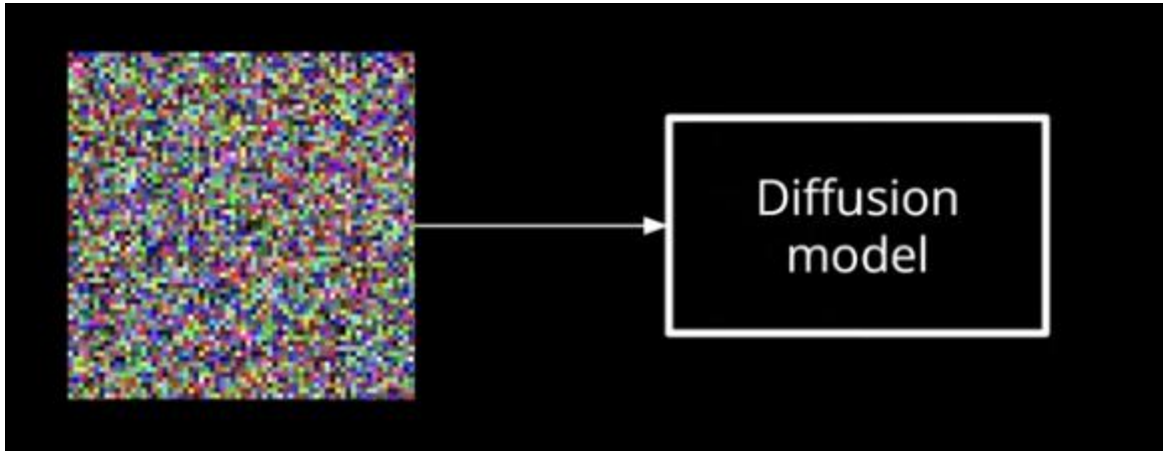


Diffusion Models: The Basic Intuition

How to find out the direction of the movement from the initial position?

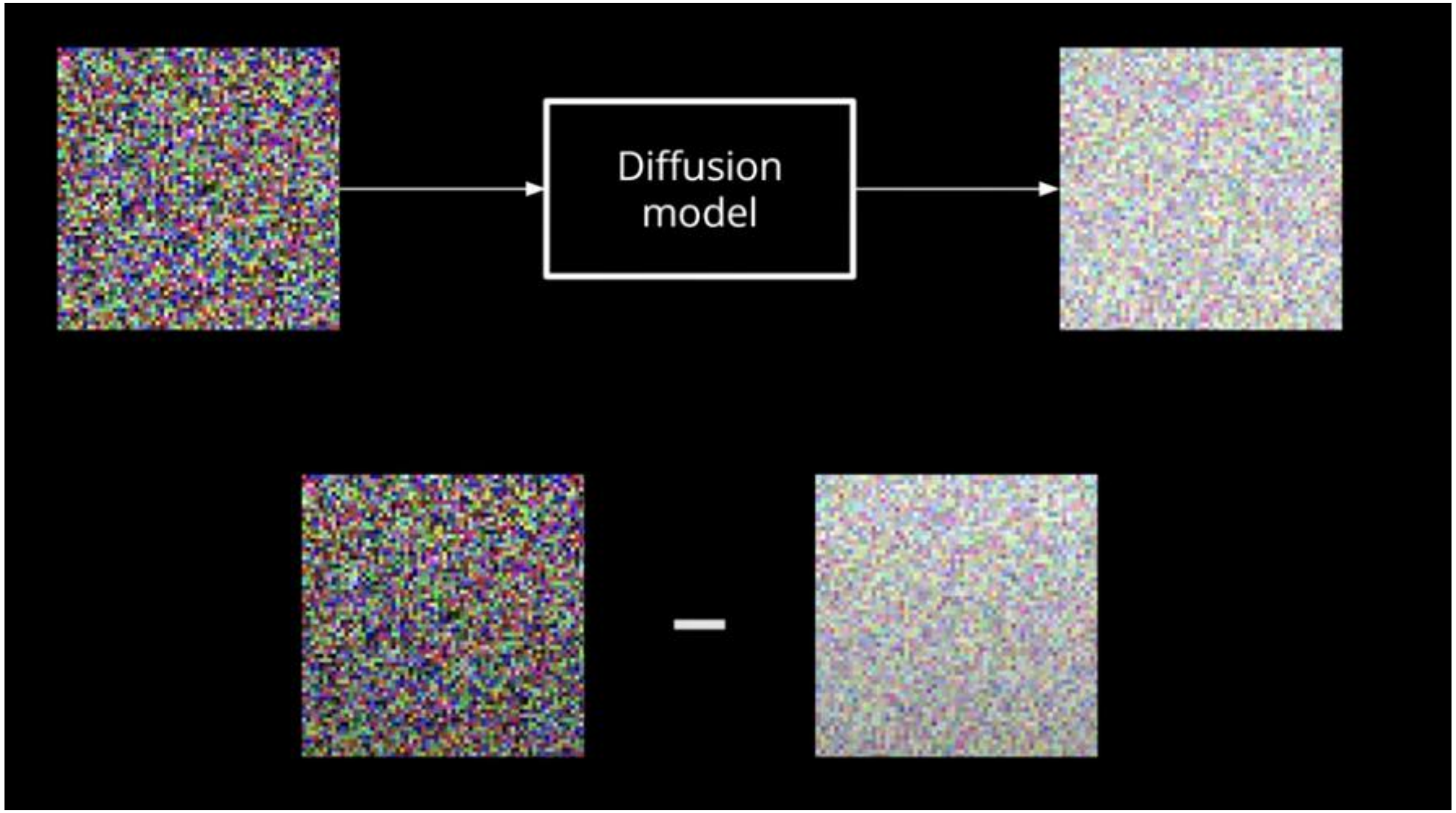
Diffusion model does exactly this

How?



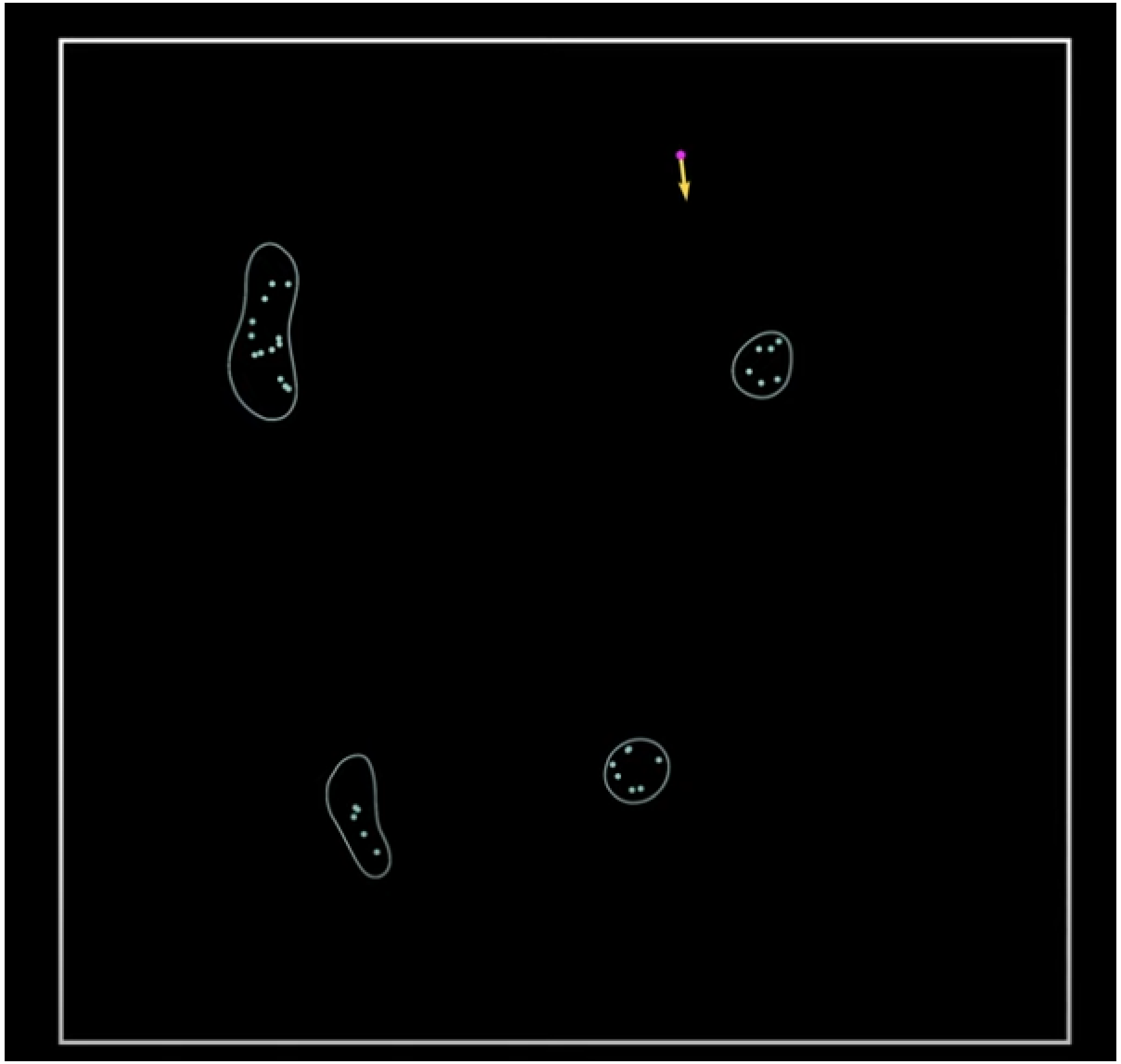
Diffusion Models: The Basic Intuition

From the input noise, the model will predict an amount that needs to be subtracted to make a movement towards image cluster



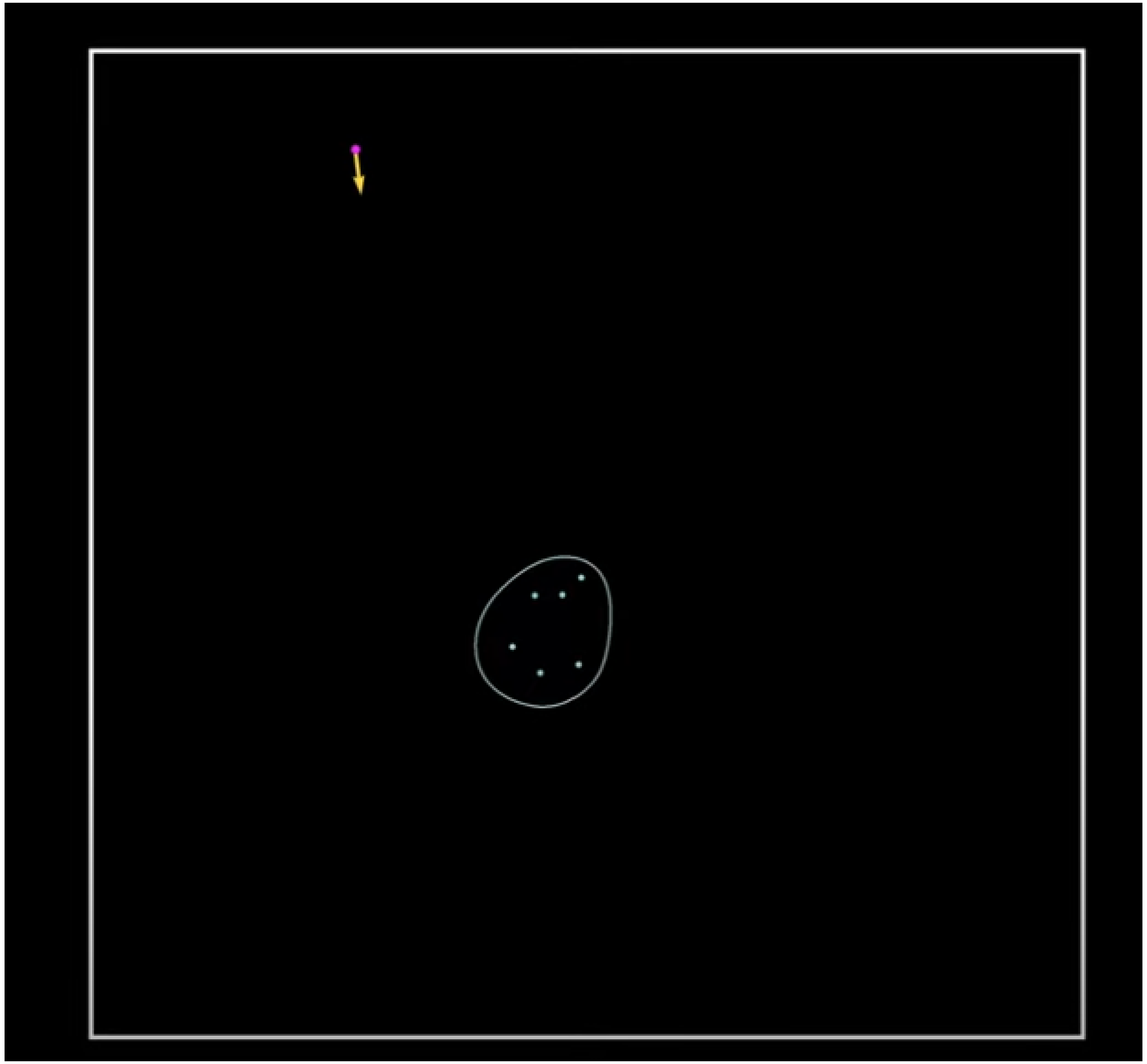
Diffusion Models: The Basic Intuition

From the input noise, the model will predict an amount that needs to be subtracted to make a movement towards image cluster



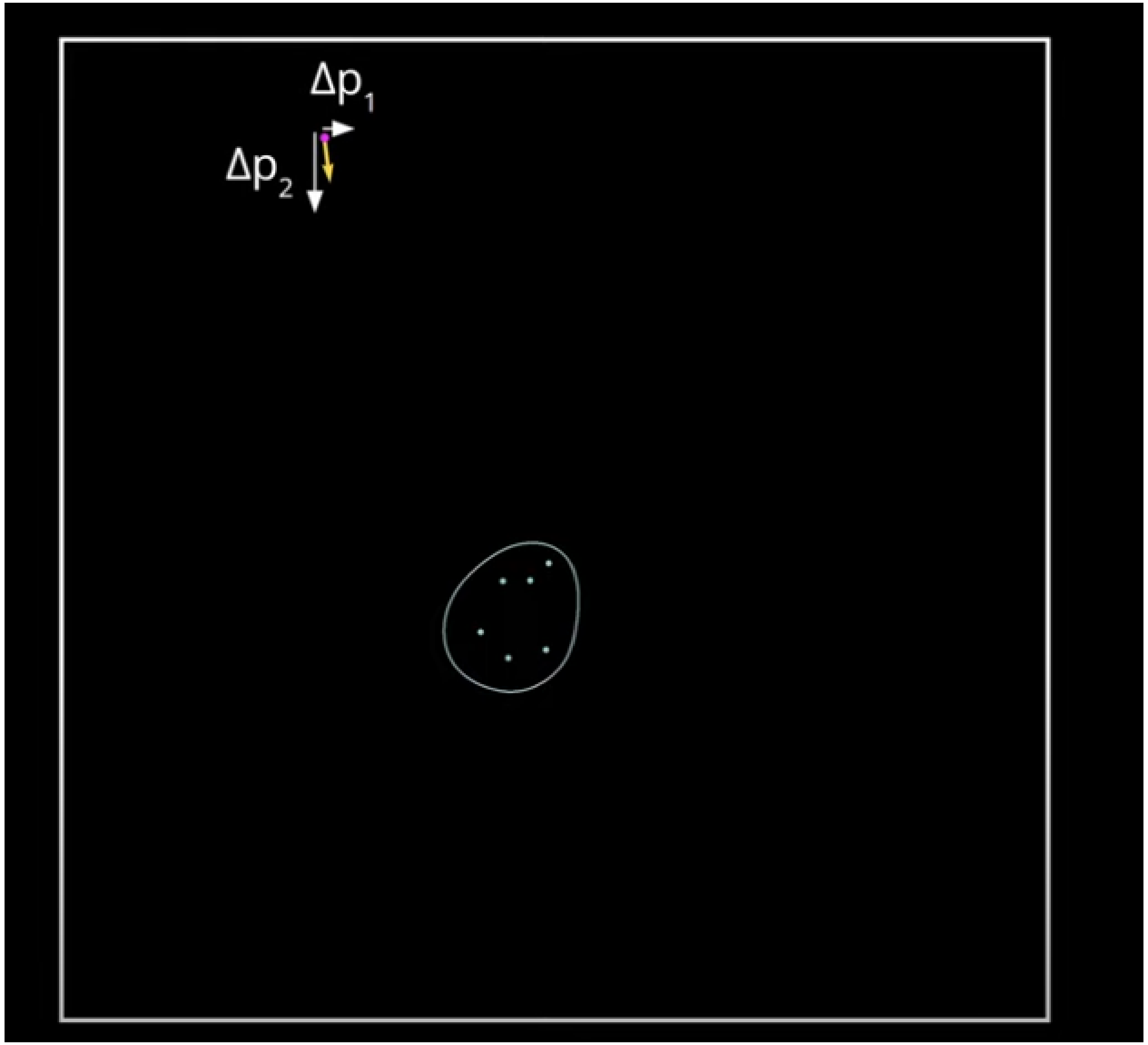
Diffusion Models: The Basic Intuition

From the input noise, the model will predict an amount that needs to be subtracted to make a movement towards image cluster



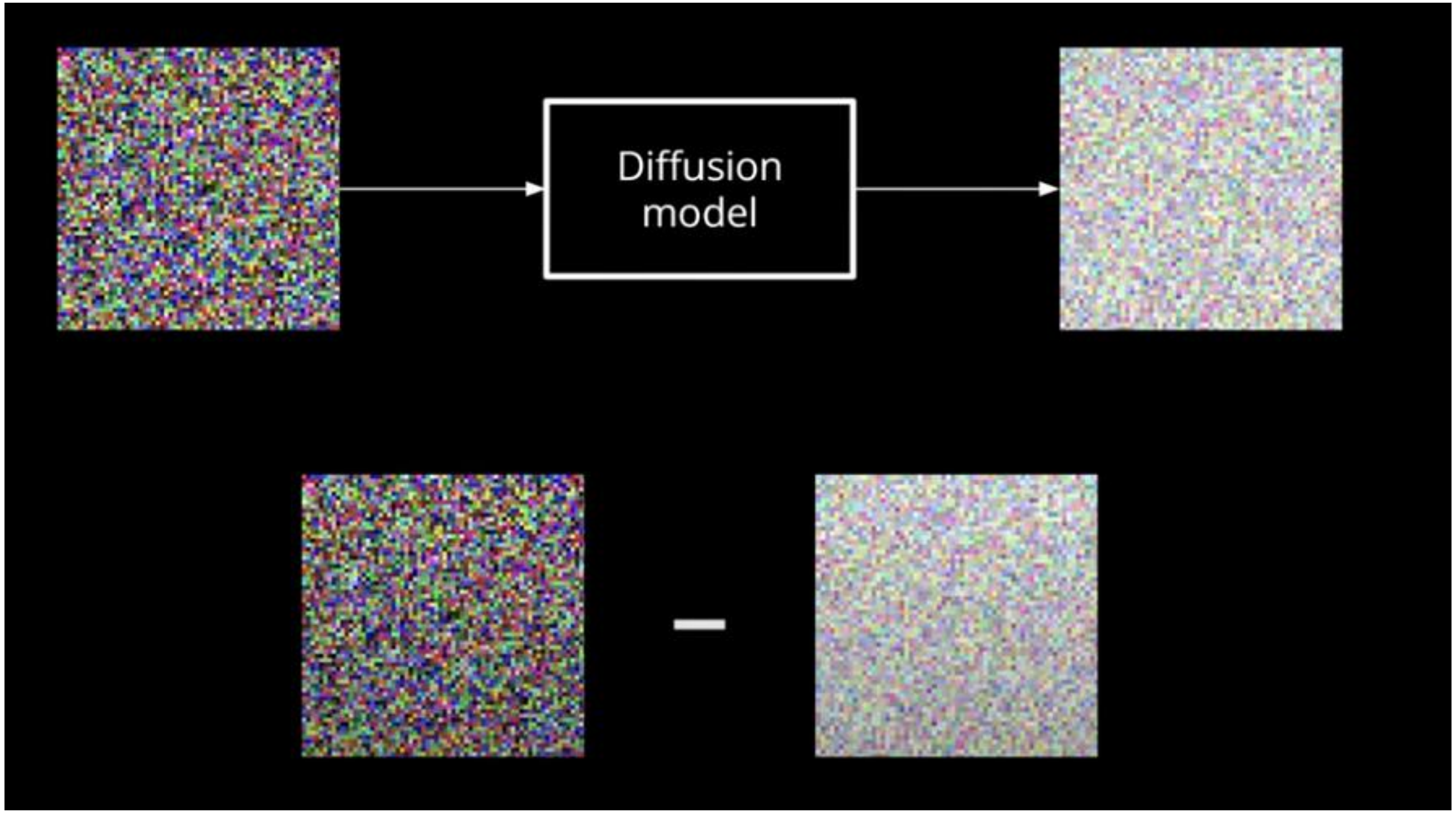
Diffusion Models: The Basic Intuition

So, diffusion model tells us on which direction we should move to get to the cluster



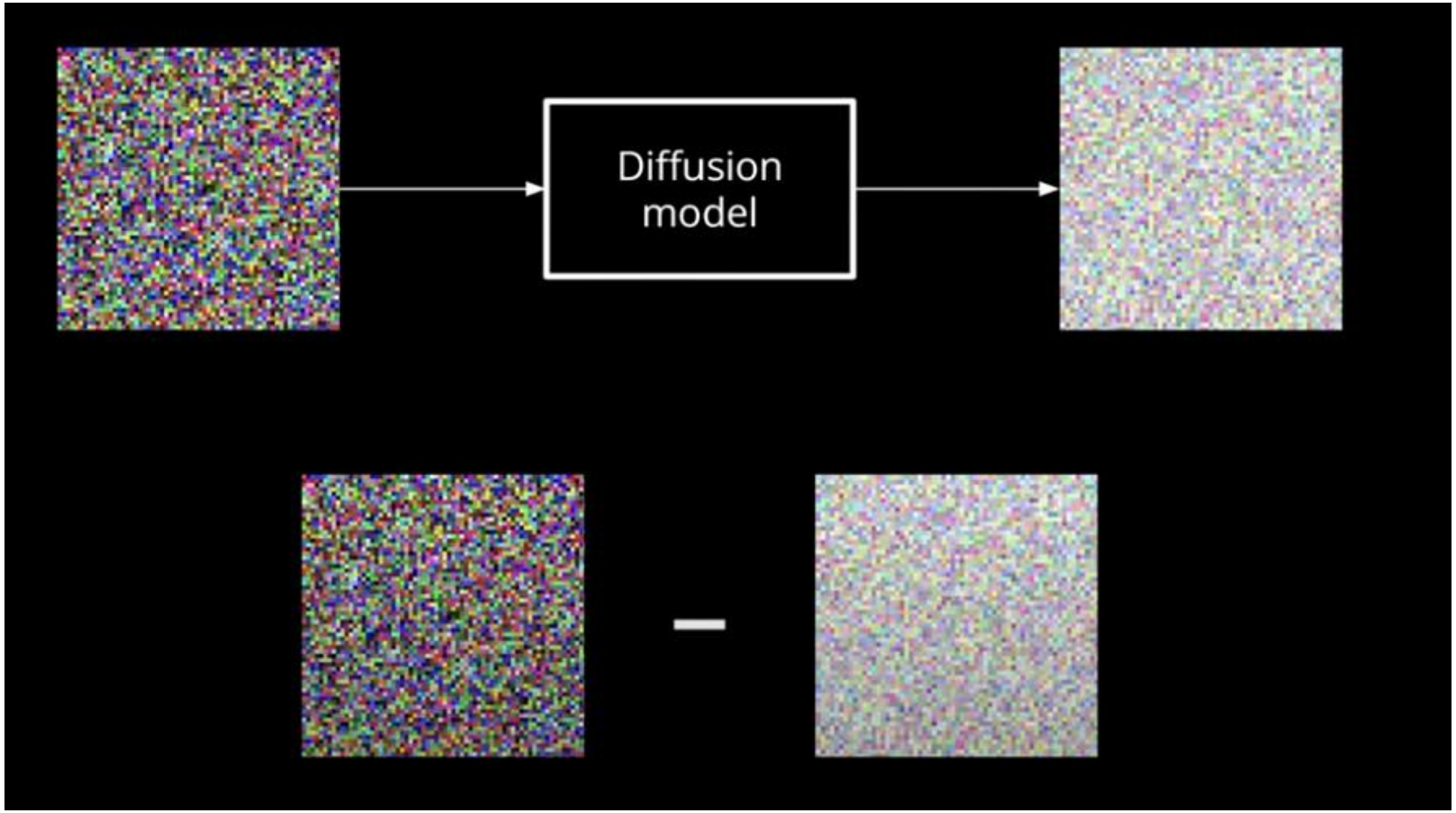
Diffusion Models: The Basic Intuition

Every subtraction is basically the movement towards the image cluster from the noise sample

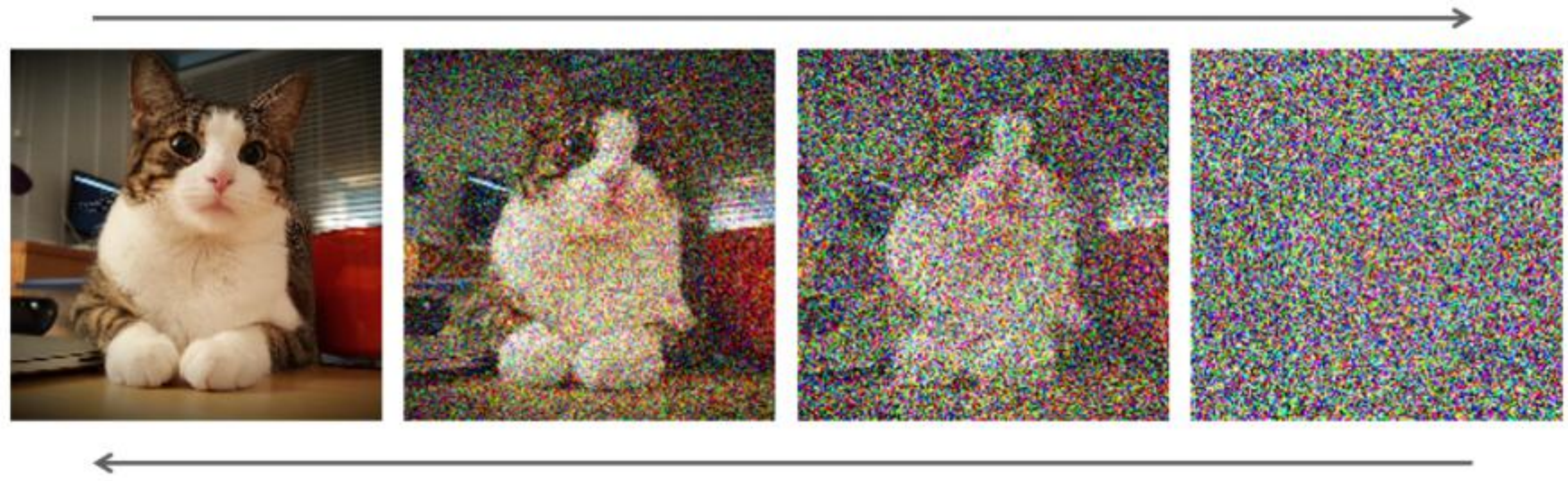


Diffusion Models: The Basic Intuition

At every subtraction (every movement), we get a slightly denoised sample compared to the previous one

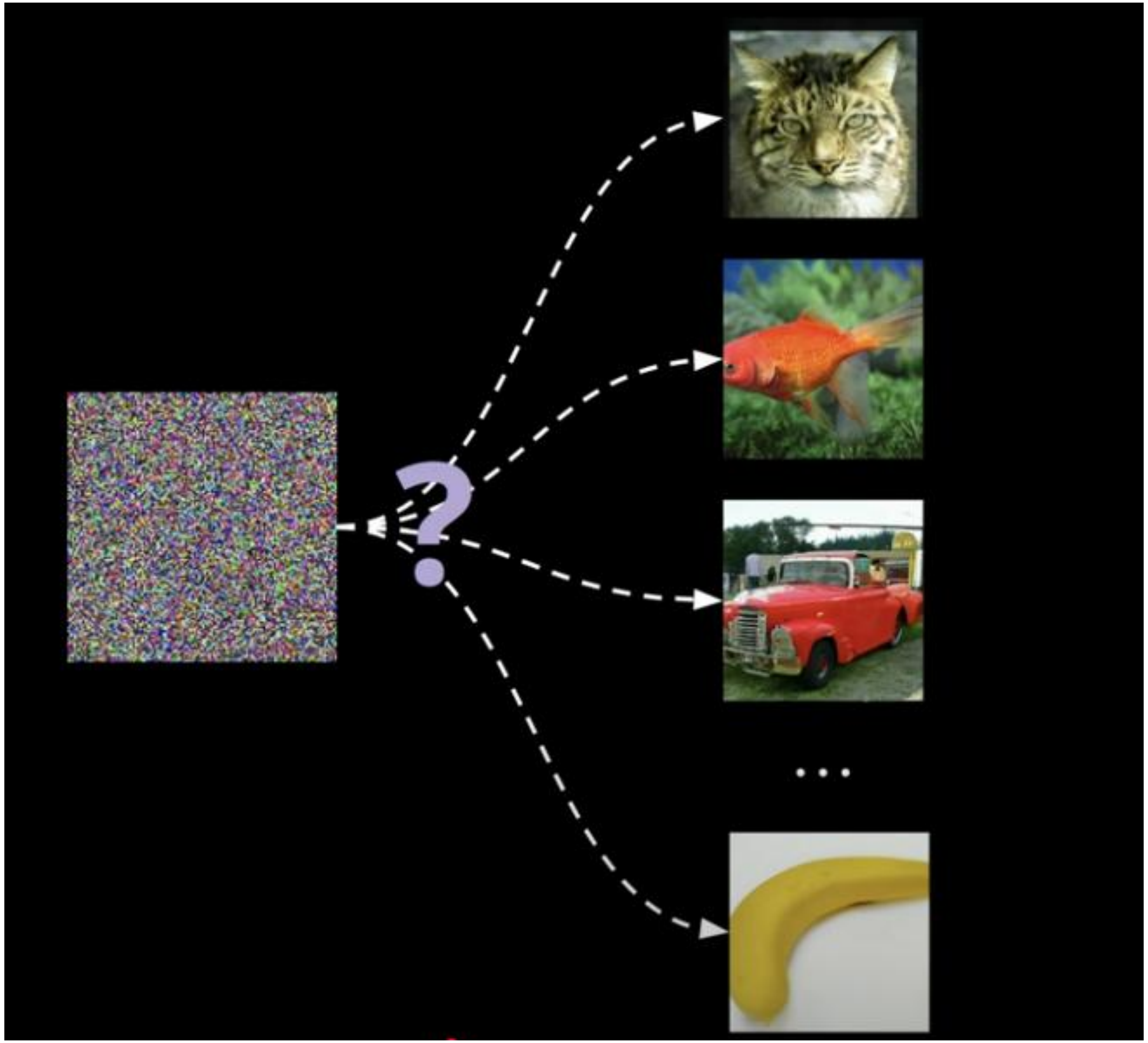


Diffusion Models in Action

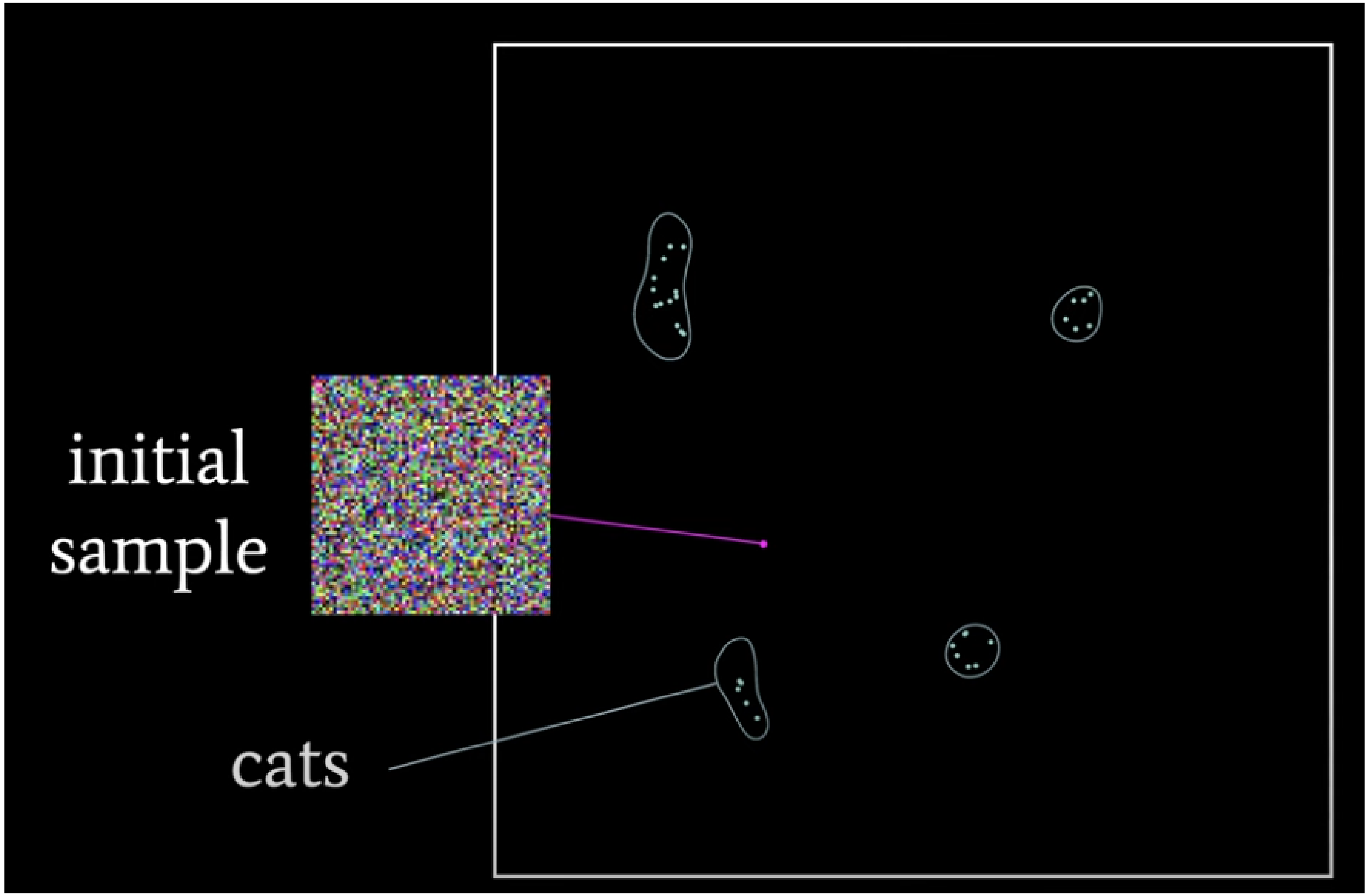


https://developer.nvidia.com/blog/improving-diffusion-models-as-an-alternative-to-gans-part-1/

How do Diffusion Models Decide which Image to Generate from the Noise?

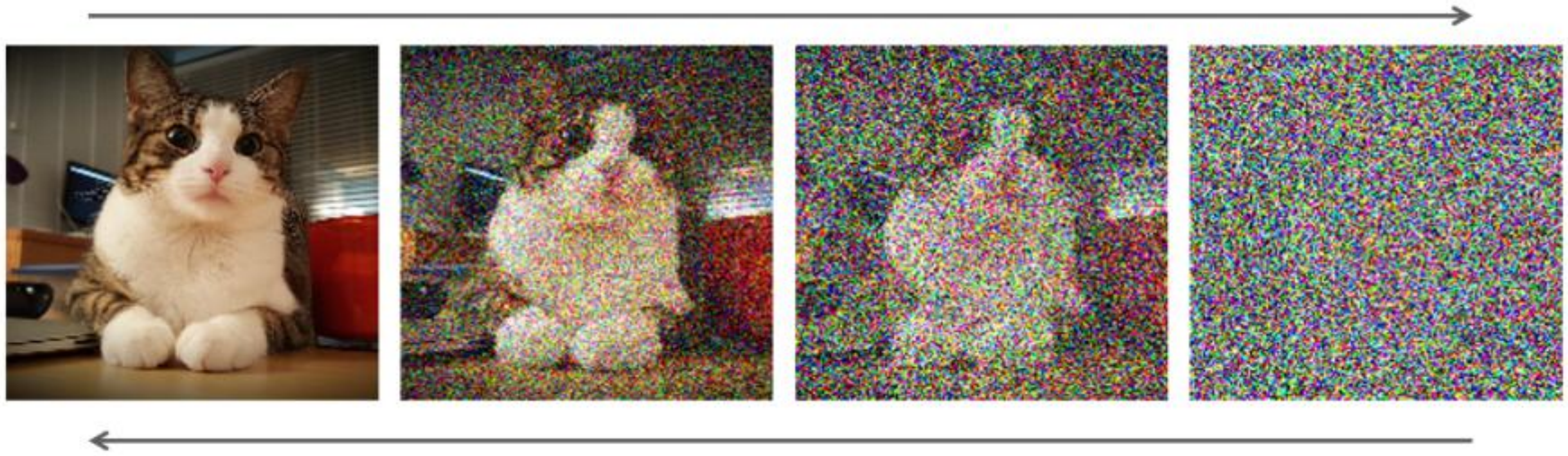


How do Diffusion Models Decide which Image to Generate from the Noise?



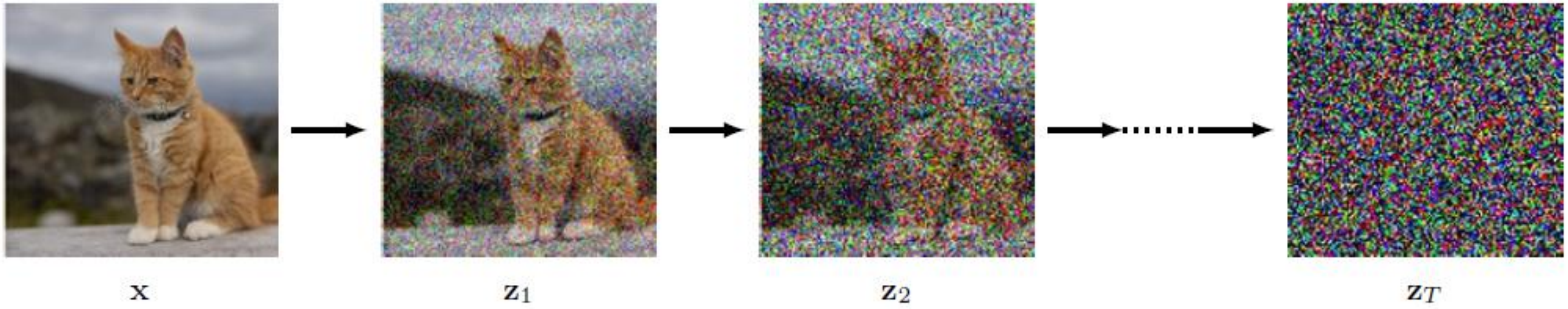
Diffusion Model

Forward Process



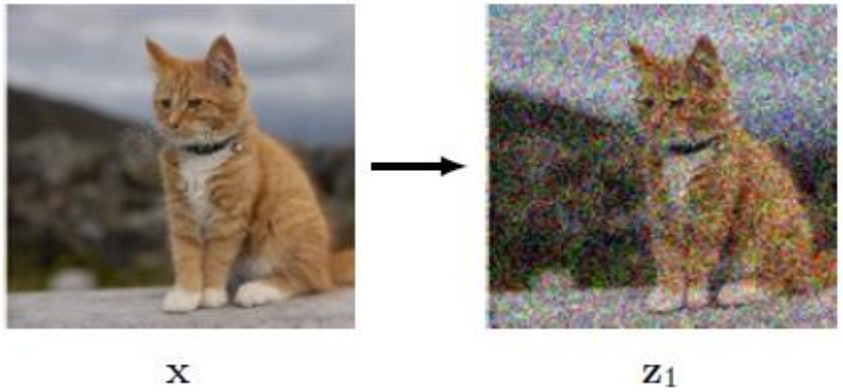
Reverse Process

Forward Process



Gradually add noise

Forward Process

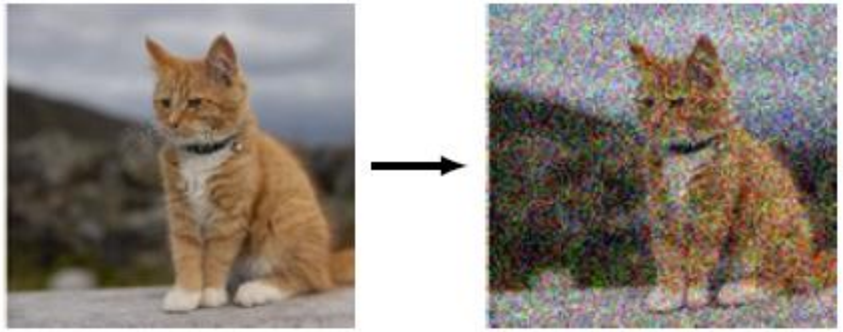


$$\mathbf{z}_1 = \sqrt{1-\beta_1}\mathbf{x} + \sqrt{\beta_1}\epsilon_1$$

$$\epsilon_1 \sim \mathcal{N}(\epsilon_1|\mathbf{0}, \mathbf{I}) \text{ and } \beta_1 < 1$$

Gradually add noise

Forward Process



$$\mathbf{z}_1 = \sqrt{1-\beta_1}\,\mathbf{x} + \sqrt{\beta_1}\,\epsilon_1$$

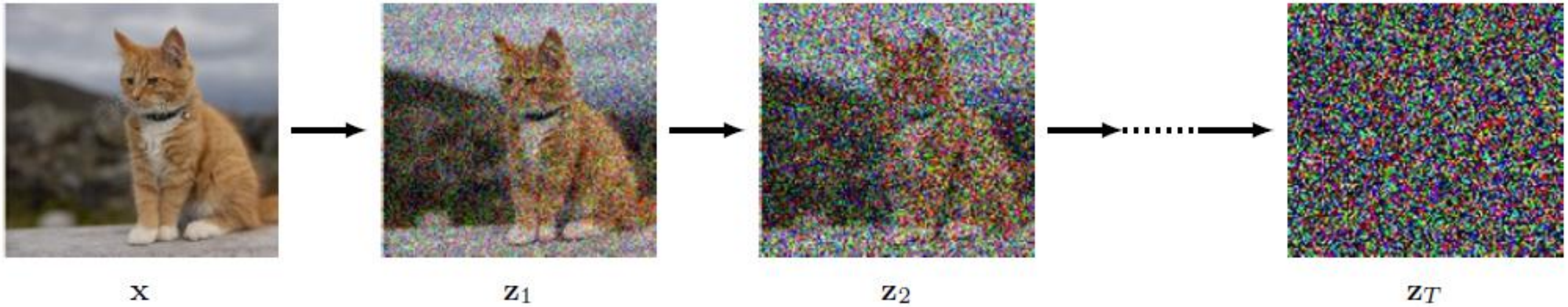
$$\epsilon_1 \sim \mathcal{N}(\epsilon_1|\mathbf{0}, \mathbf{I}) \text{ and } \beta_1 < 1$$

Variance of noise distribution

Equivalently

$$q(\mathbf{z}_1|\mathbf{x}) = \mathcal{N}(\mathbf{z}_1|\sqrt{1-\beta_1}\,\mathbf{x}, \beta_1\mathbf{I}).$$

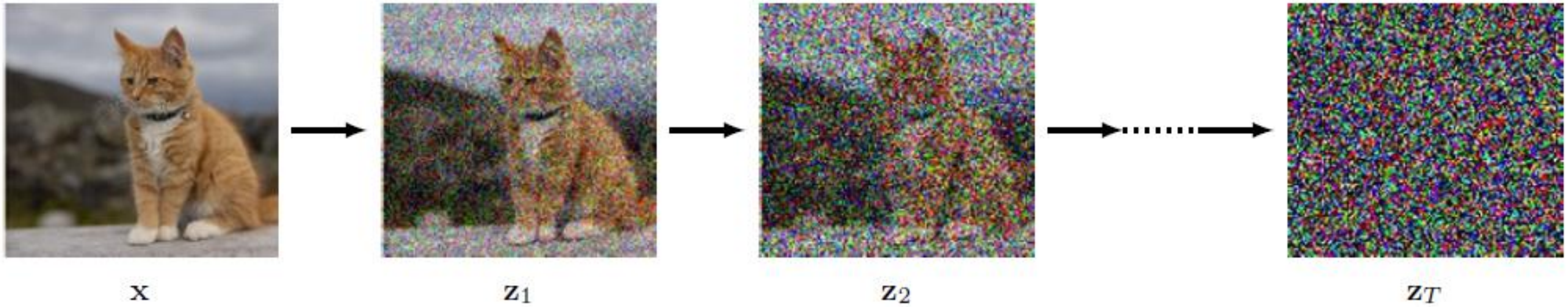
Forward Process



We then repeat this process for *T* time steps

$$\mathbf{z}_t = \sqrt{1-\beta_t}\,\mathbf{z}_{t-1} + \sqrt{\beta_t}\,\epsilon_t \qquad\qquad \epsilon_t \sim \mathcal{N}(\epsilon_t|0,\mathrm{I}).$$

Forward Process



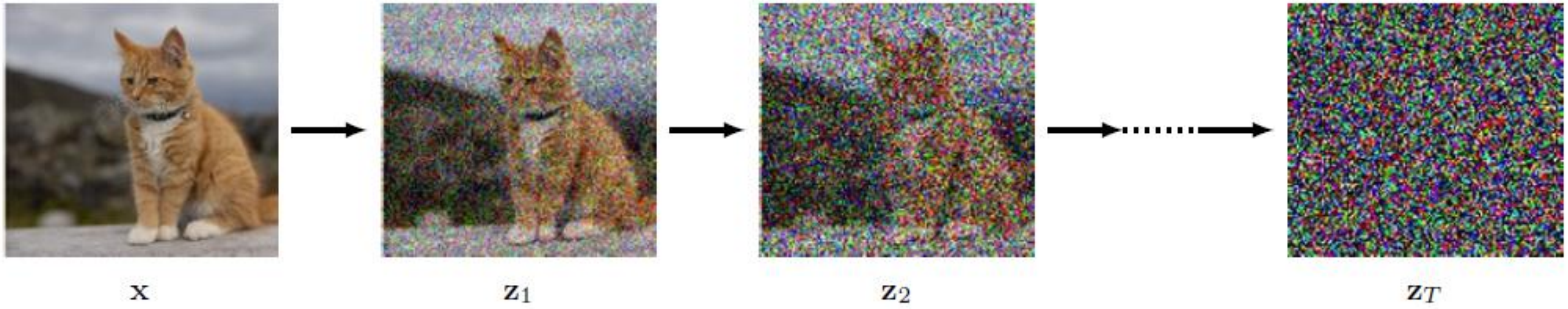
We then repeat this process for *T* time steps

$$\mathbf{z}_t = \sqrt{1-\beta_t}\mathbf{z}_{t-1} + \sqrt{\beta_t}\epsilon_t \qquad \epsilon_t \sim \mathcal{N}(\epsilon_t|0, \mathbf{I}).$$

Equivalently

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}).$$

Forward Process



We then repeat this process for *T* time steps

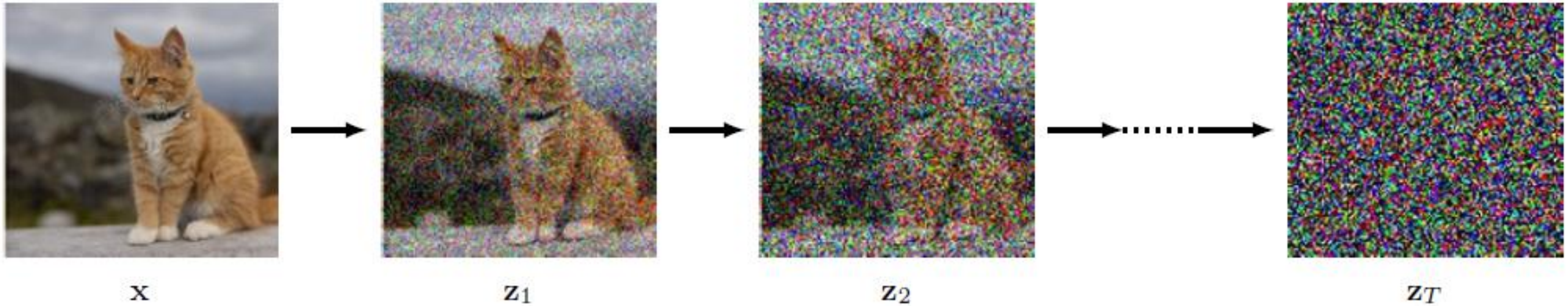
$$\mathbf{z}_t = \sqrt{1-\beta_t}\,\mathbf{z}_{t-1} + \sqrt{\beta_t}\,\epsilon_t \qquad \epsilon_t \sim \mathcal{N}(\epsilon_t|0, \mathbf{I}).$$

Equivalently

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\,\mathbf{z}_{t-1}, \beta_t \mathbf{I}).$$

Markov Chain

Forward Process



$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}).$$

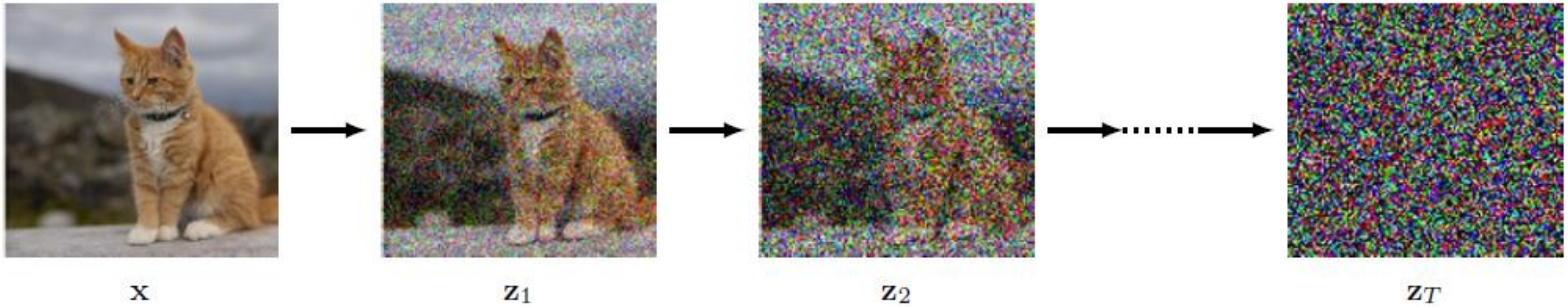
Markov Chain

$$\beta_t \in (0,1)$$

$$\beta_1 < \beta_2 < \ldots < \beta_T.$$

Schedule

Forward Process



$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t|\sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t\mathbf{I}).$$

$$\beta_t \in (0,1)$$

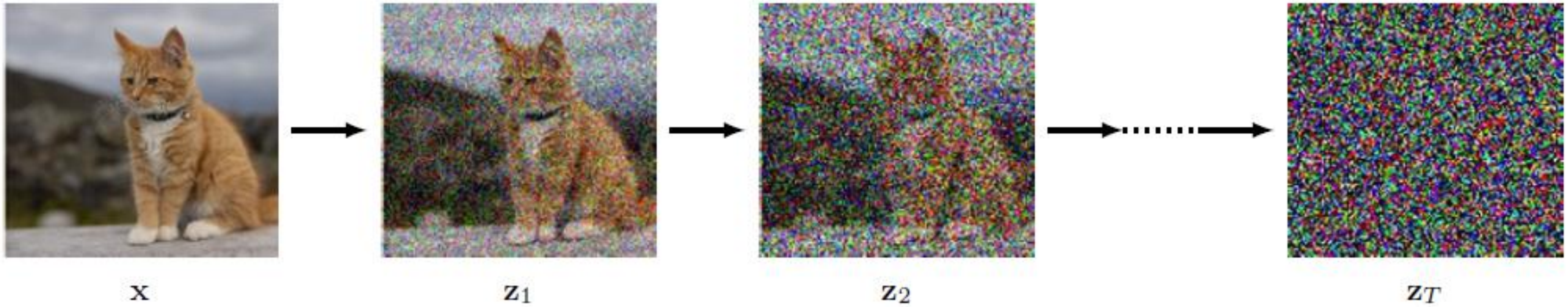
$$\beta_1 < \beta_2 < \ldots < \beta_T.$$

Schedule

Ensures that the mean of the distribution of z_t is closer to zero than the mean of z_{t-1} and that the variance of z_t is closer to the unit matrix than the variance of z_{t-1}

So, eventually z_T becomes pure Gaussian noise

Forward Process

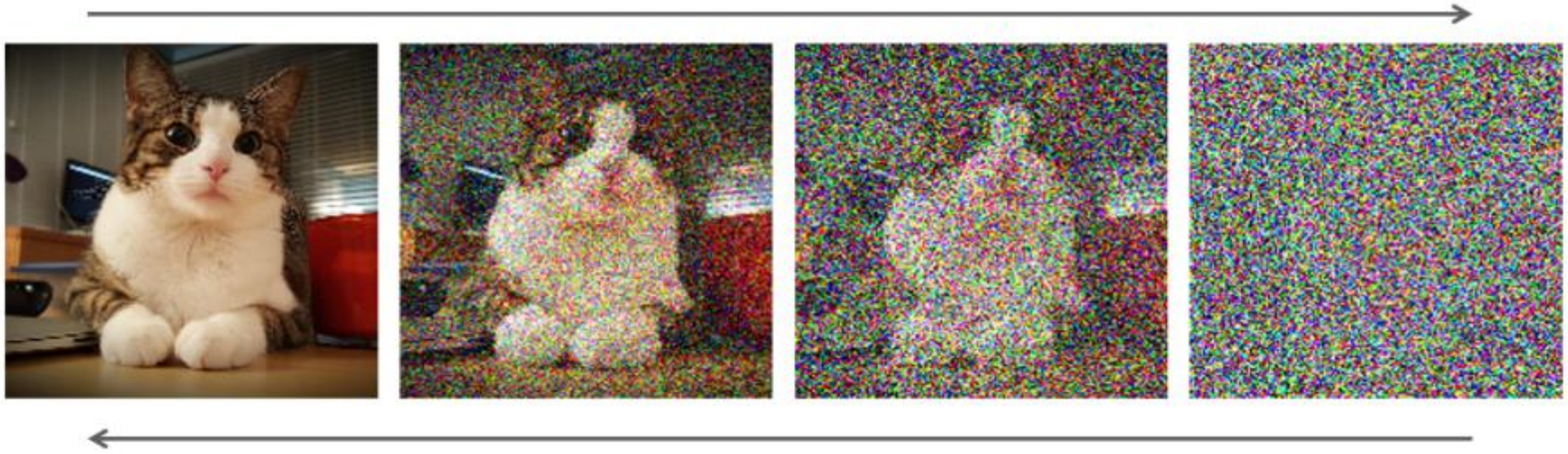


It can be shown that

$$\mathbf{z}_t = \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t \qquad \alpha_t = \prod_{\tau=1}^{t}(1-\beta_\tau).$$

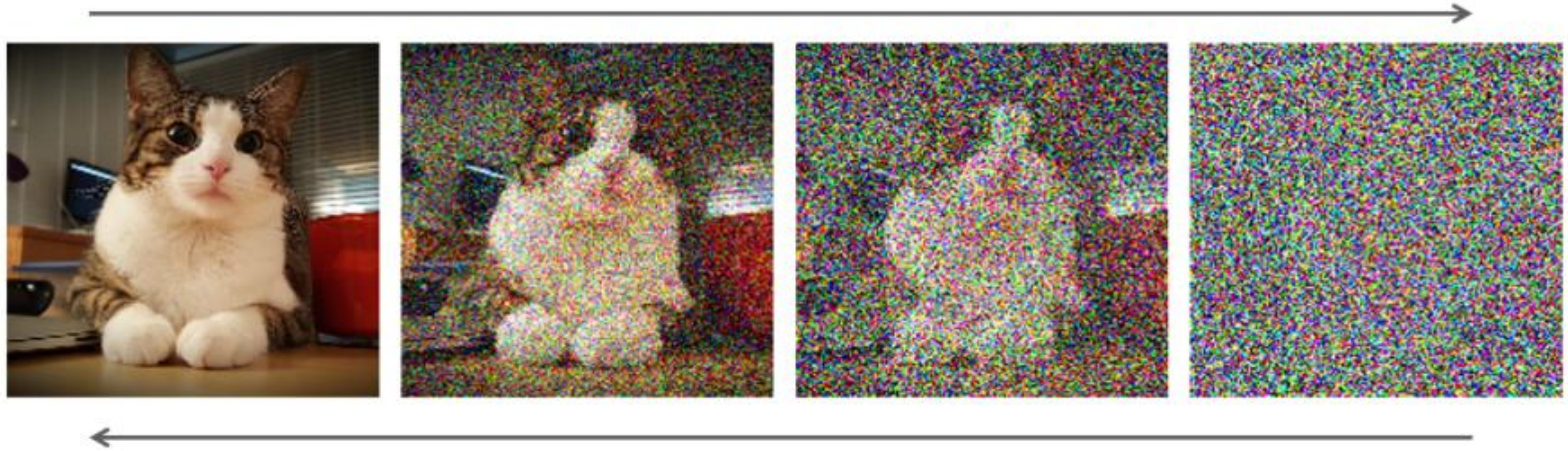
ϵ_t is the total noise added to the original image to generate z_t (not incremental noise)

Reverse Process



Reverse Process

Reverse Process

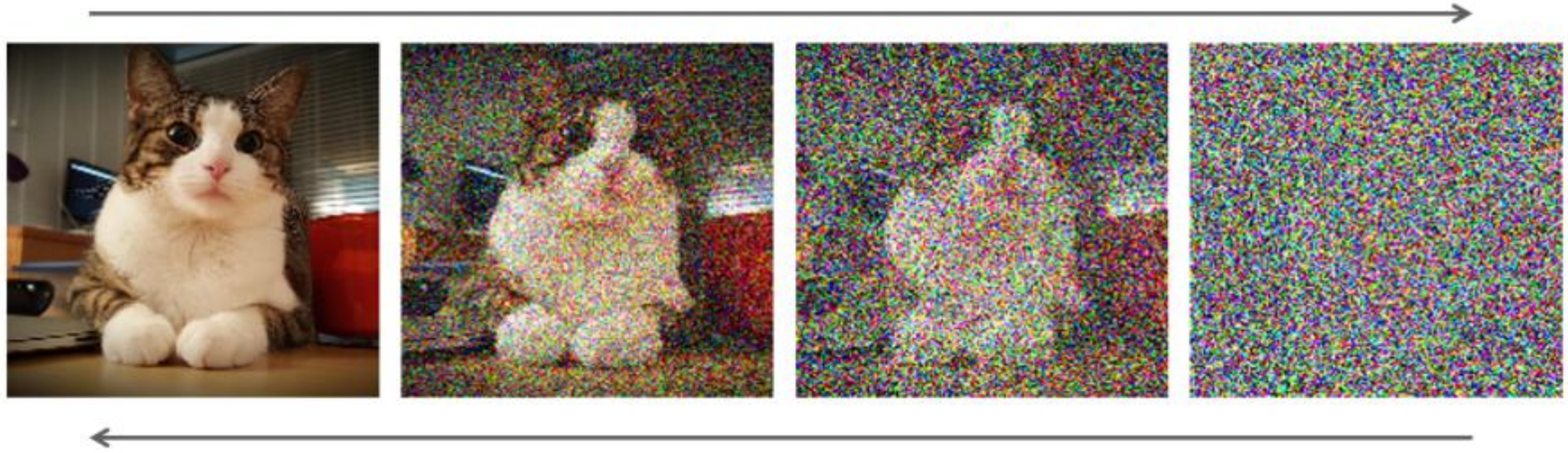


Reverse Process

Use a model to predict the noise from z_t

Subtract theta noise from z_t to get z_{t-1}

Reverse Process



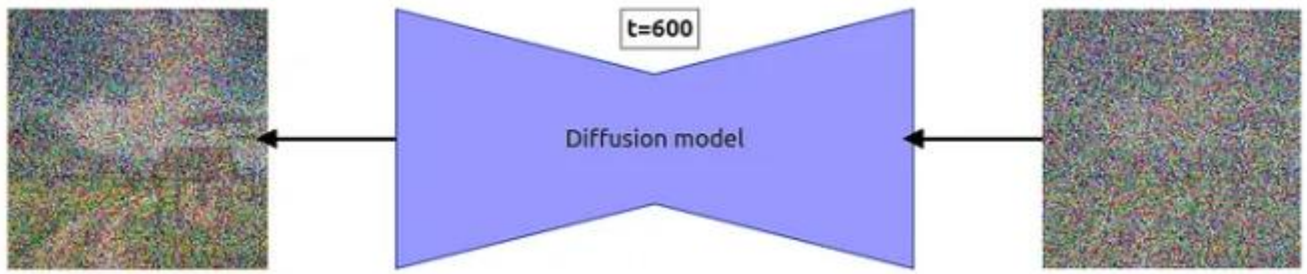
Reverse Process

Use a model to predict the noise from z_t

Subtract theta noise from z_t to get z_{t-1}

Typically, we use U-Net

Reverse Process



Reverse Process

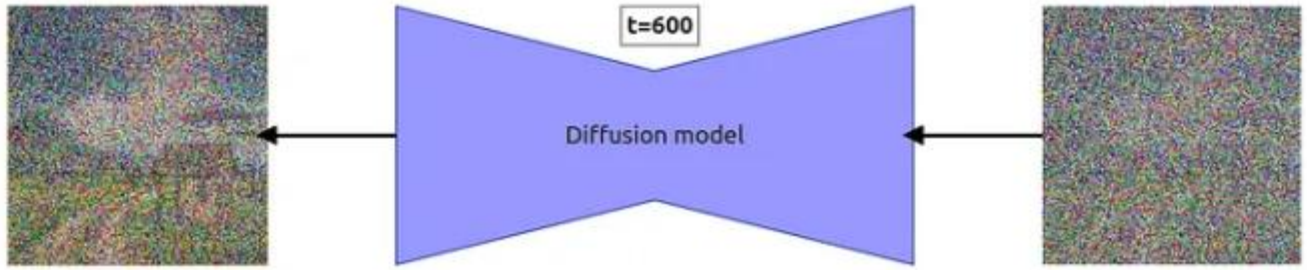
Use a model to predict the noise from z_t

Subtract theta noise from z_t to get z_{t-1}

Typically, we use U-Net

https://medium.com/@kemalpiro/step-by-step-visual-introduction-to-diffusion-models-235942d2f15c

Reverse Process



Loss function

$$\mathcal{L}(\mathbf{w}) = -\sum_{t=1}^{T} \left\| \mathrm{g}(\sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon_t, \mathbf{w}, t) - \epsilon_t \right\|^2 .$$

Reverse Process

Loss function

$$\mathcal{L}(\mathbf{w}) = -\sum_{t=1}^{T} \left\| \mathbf{g}(\sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon_t, \mathbf{w}, t) - \epsilon_t \right\|^2 .$$

Predicted total noise

Actual total noise

Training

Algorithm 20.1: Training a denoising diffusion probabilistic model

Input: Training data $\mathcal{D} = \{\mathbf{x}_n\}$
Noise schedule $\{\beta_1, \ldots, \beta_T\}$
Output: Network parameters $\mathbf{w}$

---

for $t \in \{1, \ldots, T\}$ **do**
    $\alpha_t \leftarrow \prod_{\tau=1}^{t}(1 - \beta_\tau)$ `// Calculate alphas from betas`
end for
repeat
    $\mathbf{x} \sim \mathcal{D}$ `// Sample a data point`
    $t \sim \{1, \ldots, T\}$ `// Sample a point along the Markov chain`
    $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I})$ `// Sample a noise vector`
    $\mathbf{z}_t \leftarrow \sqrt{\alpha_t}\mathbf{x} + \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}$ `// Evaluate noisy latent variable`
    $\mathcal{L}(\mathbf{w}) \leftarrow \|\mathbf{g}(\mathbf{z}_t, \mathbf{w}, t) - \boldsymbol{\epsilon}\|^2$ `// Compute loss term`
    Take optimizer step
until converged
return $\mathbf{w}$

Generating a New Synthetic Data

Algorithm 20.2: Sampling from a denoising diffusion probabilistic model

Input: Trained denoising network $\mathbf{g}(\mathbf{z}, \mathbf{w}, t)$
Noise schedule $\{\beta_1, \dots, \beta_T\}$
Output: Sample vector $\mathbf{x}$ in data space

$\mathbf{z}_T \sim \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ `// Sample from final latent space`
for $t \in T, \dots, 2$ **do**
  $\alpha_t \leftarrow \prod_{\tau=1}^{t}(1-\beta_\tau)$ `// Calculate alpha`
  `// Evaluate network output`
  $\boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) \leftarrow \frac{1}{\sqrt{1-\beta_t}}\left\{\mathbf{z}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\mathbf{g}(\mathbf{z}_t, \mathbf{w}, t)\right\}$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I})$ `// Sample a noise vector`
  $\mathbf{z}_{t-1} \leftarrow \boldsymbol{\mu}(\mathbf{z}_t, \mathbf{w}, t) + \sqrt{\beta_t}\boldsymbol{\epsilon}$ `// Add scaled noise`
end for
$\mathbf{x} = \frac{1}{\sqrt{1-\beta_1}}\left\{\mathbf{z}_1 - \frac{\beta_1}{\sqrt{1-\alpha_1}}\mathbf{g}(\mathbf{z}_1, \mathbf{w}, t)\right\}$ `// Final denoising step`
return $\mathbf{x}$

https://huggingface.co/spaces/stabilityai/stable-diffusion

All the best!