Image Reasoning and Generative Models

A First-Principles Survey of Architectures, Training, and Open Problems

Anjan Goswami

General Manager, SmartInfer.com

January 2026

Abstract: This survey provides a comprehensive examination of modern image generation and understanding models, tracing the architectural lineage from first principles. We analyze why specific design choices emerged—from the information-theoretic foundations of variational autoencoders to the score-matching basis of diffusion models, and from the sequential factorization of autoregressive approaches to the structured reasoning of scene graph methods. Rather than merely cataloging models, we emphasize the fundamental tradeoffs that motivated each paradigm: expressiveness versus tractability, quality versus speed, and flexibility versus control. The survey covers diffusion models, rectified flow, autoregressive image generation, visual tokenization, unified multimodal architectures, and programmatic image synthesis. We identify persistent open problems including compositional reasoning, efficient high-fidelity generation, and the unification of image understanding with generation.

1. Introduction: The Fundamental Challenge

Image generation is, at its core, a problem of learning and sampling from high-dimensional probability distributions. A 256×256 RGB image lives in a space of $256^3 \approx 16.7$ million possible values per pixel, yielding a joint space of dimension $256 \times 256 \times 3 = 196,608$. Yet natural images occupy a vanishingly small manifold within this space—most random pixel configurations produce noise, not coherent scenes. The central challenge of generative modeling is to characterize this manifold and sample from it efficiently.

This survey examines image generation through the lens of first principles, asking not just *what* architectures exist, but *why* they emerged and *what fundamental tradeoffs* they navigate. We trace the intellectual lineage from classical density estimation through modern diffusion and autoregressive approaches, revealing how each paradigm represents a different resolution of the core tension between expressiveness and tractability.

1.1 The Density Estimation Perspective

The most direct formulation of image generation is density estimation: given a dataset of images $\{x_1, x_2, \ldots, x_N\}$ drawn from some unknown distribution $p_{\text{data}}(x)$, learn a model $p_\theta(x)$ that approximates this distribution. Once learned, we can generate new images by sampling $x \sim p_\theta(x)$.

This framing immediately reveals the fundamental difficulty. For a model to assign high probability to natural images and low probability to noise, it must capture the intricate statistical dependencies among hundreds of thousands of pixels. These dependencies are hierarchical (edges form textures, textures form objects, objects form scenes), long-range (the sky's color constrains the lighting on a face), and semantically rich (a caption mentioning "sunset" implies specific color palettes).

Classical approaches to density estimation fall into two categories, each with characteristic limitations:

Explicit density models directly parameterize $p_\theta(x)$ and optimize the log-likelihood $\mathcal{L} = \mathbb{E}_{x \sim p_{\text{data}}}[\log p_\theta(x)]$. The challenge is ensuring that $p_\theta(x)$ integrates to 1 (normalization) while remaining expressive. Normalizing flows achieve this through invertible transformations but are constrained in architecture. Energy-based models avoid the constraint but require expensive MCMC sampling or contrastive divergence for training.

Implicit density models learn to transform simple noise $z \sim \mathcal{N}(0, I)$ into samples $x = G_\theta(z)$ without explicitly modeling $p_\theta(x)$. Generative Adversarial Networks (GANs) exemplify this approach, using a discriminator to guide the generator toward the data distribution. While capable of high-quality samples, GANs suffer from training instability and mode collapse—the generator may learn to produce only a subset of the data distribution.

The modern era of image generation can be understood as the search for methods that combine the stable training of likelihood-based models with the sample quality of implicit models.

1.2 The Manifold Hypothesis and Latent Spaces

A key insight underlying modern architectures is the *manifold hypothesis*: natural images lie on or near a low-dimensional manifold embedded in the high-dimensional pixel space. This suggests a two-stage approach:

1. Learn an encoder $E : \mathcal{X} \to \mathcal{Z}$ that maps images to a lower-dimensional latent space where the data distribution is simpler.
2. Model the distribution in latent space $p_\theta(z)$ and generate images via a decoder $D : \mathcal{Z} \to \mathcal{X}$.

This factorization is not merely a computational convenience—it reflects a deep structural assumption about images. The latent space $\mathcal{Z}$ ideally captures semantic content (what objects are present, their poses, the scene layout) while the decoder handles low-level rendering (textures, lighting, precise pixel values). This separation enables both efficient computation and meaningful manipulation of generated content.

The Variational Autoencoder (VAE) formalizes this intuition through the lens of variational inference, while VQ-VAE and its successors discretize the latent space to interface with language model architectures. The choice between continuous and discrete latent spaces represents a fundamental design decision with implications for expressiveness, training stability, and downstream applications.

2. The Information-Theoretic Foundation: Variational Autoencoders

The Variational Autoencoder provides the theoretical scaffold for understanding latent-space generative models. Its derivation from first principles illuminates why certain architectural choices are necessary rather than arbitrary.

2.1 The Evidence Lower Bound

Consider the problem of maximizing the log-likelihood of the data under a latent variable model:

$$\log p_\theta(x) = \log \int p_\theta(x|z)p(z)\,dz \tag{1}$$

This integral is generally intractable because it requires marginalizing over all possible latent codes z. The VAE addresses this through variational inference: introduce an approximate posterior $q_\phi(z|x)$ and derive a lower bound on the log-likelihood.

Applying Jensen's inequality:

$$\log p_\theta(x) = \log \int p_\theta(x|z)p(z)\,dz \tag{2}$$

$$= \log \int q_\phi(z|x)\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz \tag{3}$$

$$\geq \int q_\phi(z|x)\log\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz \tag{4}$$

$$= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\mathrm{KL}}(q_\phi(z|x)\|p(z)) \tag{5}$$

This is the *Evidence Lower Bound* (ELBO), comprising two terms with clear interpretations:

Reconstruction term $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$: The expected log-likelihood of reconstructing x from its latent code. This encourages the encoder-decoder pair to preserve information.

Regularization term $-D_{\mathrm{KL}}(q_\phi(z|x)\|p(z))$: The KL divergence between the approximate posterior and the prior. This encourages the latent space to be "smooth" and match the prior distribution, enabling sampling.

2.2 The Rate-Distortion Tradeoff

The ELBO reveals a fundamental tension at the heart of VAE training. The reconstruction term pushes for high-fidelity encoding—each image should map to a distinct, informative latent code. The KL term pushes for compression—latent codes should be close to the prior, making them indistinguishable and thus enabling generation.

This is precisely the *rate-distortion tradeoff* from information theory. The "rate" (information transmitted through the bottleneck) is upper-bounded by the KL term, while the "distortion" (reconstruction error) is minimized by the reconstruction term. No encoder can simultaneously achieve zero distortion and zero rate unless the data has no entropy.

In practice, VAEs often exhibit *posterior collapse*: the model learns to ignore the latent code entirely, setting $q_\phi(z|x) \approx p(z)$ for all x. This achieves zero KL cost but forces the decoder to model the entire data distribution without latent structure. Various techniques address this—KL annealing, free bits, β-VAE—but they represent different positions on the rate-distortion curve rather than escaping the tradeoff.

The lesson for image generation is profound: *any latent-space model must balance the informativeness of its representation against its regularity*. This tradeoff reappears in different guises across all generative paradigms.

2.3 Why Reconstruction Loss Matters: MSE vs. Perceptual

The choice of reconstruction loss $\log p_\theta(x|z)$ has dramatic effects on generated image quality. If we assume a Gaussian observation model $p_\theta(x|z) = \mathcal{N}(x; D_\theta(z), \sigma^2 I)$, the reconstruction term becomes mean squared error (MSE):

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \propto -\|x - D_\theta(z)\|^2 \tag{6}$$

MSE treats all pixel errors equally and independently. But human perception is not uniform: we are more sensitive to edges than to smooth regions, more tolerant of high-frequency noise than of structural distortions. Images that are close in MSE may look very different to humans, and vice versa.

This motivates *perceptual losses*: instead of comparing pixels directly, compare features extracted by a pretrained network (typically VGG or a similar classifier):

$$\mathcal{L}_{\text{perceptual}} = \|\phi(x) - \phi(D_\theta(z))\|^2 \tag{7}$$

where ϕ extracts intermediate feature maps. This aligns the loss with human perception, as the pretrained network has learned to extract semantically meaningful features.

The VQ-GAN architecture takes this further by adding an adversarial loss: a discriminator network learns to distinguish real images from reconstructions, and the reconstruction is penalized for being detectable as fake. This implicitly captures all aspects of "looking realistic" that the discriminator can learn, rather than relying on a fixed perceptual metric.

3. Discretization: From Continuous to Discrete Latents

The transition from continuous VAE latents to discrete tokens represents a pivotal architectural choice that enables the interface between vision and language models.

3.1 Vector Quantization: The VQ-VAE

The Vector Quantized VAE (VQ-VAE) replaces the continuous latent space with a discrete codebook $\mathcal{C} = \{e_1, e_2, \ldots, e_K\}$ of K learnable embedding vectors. The encoder produces a continuous representation $z_e = E(x)$, which is then quantized to the nearest codebook entry:

$$z_q = e_k \quad \text{where} \quad k = \arg\min_j \|z_e - e_j\|^2 \tag{8}$$

The decoder receives the quantized code z_q and reconstructs the image. This discretization has several important consequences:

Eliminates posterior collapse: Unlike VAEs where the KL term encourages $q(z|x) \approx p(z)$, VQ-VAE uses a uniform prior over codebook entries. The model cannot "cheat" by ignoring the latent code, as the discrete bottleneck forces information through.

Enables autoregressive modeling: Discrete tokens can be modeled by transformers using standard cross-entropy loss, exactly as in language modeling. This opens the door to leveraging the massive scaling that has proven effective for LLMs.

Introduces the codebook collapse problem: If the encoder consistently ignores certain codebook entries, they receive no gradient signal and become "dead codes." Techniques like exponential moving average updates, codebook reset, and commitment losses address this.

3.2 The Gradient Problem and Straight-Through Estimation

The quantization operation $z_q = e_{\arg\min_j \|z_e - e_j\|}$ is non-differentiable: small changes to z_e may not change which codebook entry is selected, and when they do, the change is discontinuous. How can we train through this operation?

VQ-VAE uses the *straight-through estimator*: during the forward pass, use the quantized code z_q; during the backward pass, copy gradients from z_q directly to z_e, as if quantization were the identity function. This is mathematically unjustified but empirically effective.

The training objective becomes:

$$\mathcal{L} = \|x - D(z_q)\|^2 + \|\text{sg}[z_e] - e_k\|^2 + \beta\|z_e - \text{sg}[e_k]\|^2 \tag{9}$$

where $\text{sg}[\cdot]$ denotes stop-gradient. The three terms are:

1. **Reconstruction loss**: drives the encoder-decoder to preserve information
2. **Codebook loss**: moves codebook entries toward encoder outputs
3. **Commitment loss**: encourages encoder outputs to stay close to their assigned codes

The commitment loss, weighted by β, prevents the encoder from "wandering" arbitrarily far from the codebook, which would make quantization increasingly disruptive.

3.3 From VQ-VAE to VQ-GAN: Perceptual Quality

VQ-VAE with MSE loss produces blurry reconstructions, particularly at high compression ratios. The VQ-GAN addresses this by incorporating:

Perceptual loss: Compare VGG features rather than raw pixels, aligning the loss with human perception.

Adversarial loss: A patch-based discriminator classifies local regions as real or reconstructed. The generator (decoder) is trained to fool the discriminator, encouraging perceptually realistic outputs even if they differ from the input in pixel space.

Transformer architecture: While the original VQ-VAE uses convolutional encoders/decoders, adding attention layers captures global dependencies that convolutions miss.

The result is dramatically improved reconstruction quality: VQ-GAN can achieve visually faithful reconstructions at compression ratios of $16\times$ or higher, compared to VQ-VAE which becomes visibly degraded at such ratios.

3.4 Scaling Codebooks: The Collapse Problem

A natural question is whether larger codebooks yield better reconstructions. In principle, more codes should enable finer-grained distinctions. In practice, naively increasing codebook size often leads to *codebook collapse*: most codes are never used, and the effective vocabulary is much smaller than the nominal size.

This occurs because the encoder-codebook-decoder system is jointly optimized. If certain codes are initially poorly positioned, they may never be selected, receive no gradients, and remain unused indefinitely. The problem compounds: as some codes die, the remaining codes must cover more of the representation space, making it harder for new codes to become active.

Several innovations address this challenge:

Lookup-Free Quantization (LFQ): Instead of learning codebook embeddings, map the continuous code to a fixed binary representation. Each dimension is independently quantized to $\{-1, +1\}$, yielding 2^d possible codes for a d-dimensional latent. This eliminates the codebook entirely, replacing it with a deterministic mapping.

Finite Scalar Quantization (FSQ): Each latent dimension is independently quantized to a small set of fixed values (e.g., $\{-2, -1, 0, 1, 2\}$). The total vocabulary size is the product of per-dimension vocabularies. Like LFQ, this removes learned codebook embeddings.

Multi-group quantization: Split the latent into groups, each with its own codebook. This factorizes the exponentially large joint space into manageable per-group vocabularies while still enabling large effective vocabulary sizes.

These methods have enabled codebook sizes of 262K or more with near-complete utilization, a crucial advance for autoregressive image generation where vocabulary size directly impacts expressiveness.

4. Architectural Foundations: Why These Network Designs?

The choice of neural network architecture for encoders, decoders, and generative models is not arbitrary. Each design reflects assumptions about the structure of images and the computational patterns needed to capture that structure.

4.1 Convolutional Networks: Translation Equivariance

Convolutional neural networks (CNNs) dominate image processing due to two key properties:

Local connectivity: Each neuron connects only to a local patch of the input, reflecting the intuition that nearby pixels are more related than distant ones. A 3×3 convolution looks at 9 pixels regardless of image size.

Weight sharing: The same filter is applied at all spatial locations, reducing parameters and enabling the network to detect a feature (e.g., an edge) regardless of where it appears. This is *translation equivariance*: shifting the input shifts the feature map by the same amount.

These properties are well-suited to images because: (1) local patterns like edges and textures are meaningful building blocks, and (2) these patterns can appear anywhere in the image with similar semantics. A vertical edge in the top-left corner should be processed the same way as one in the bottom-right.

However, CNNs have a fundamental limitation: their receptive field grows slowly with depth. A stack of 3×3 convolutions has a receptive field that grows linearly with the number of layers. To capture global structure—the relationship between a person's face and the scene's background—requires either very deep networks or large kernels, both computationally expensive.

4.2 The U-Net: Multi-Scale Processing

The U-Net architecture, originally developed for image segmentation, has become the backbone of diffusion models. Its design addresses the multi-scale nature of images:

Encoder path: Progressively downsample the image through convolutions and pooling, increasing the receptive field and capturing coarse, global structure.

Decoder path: Progressively upsample, recovering spatial resolution while combining with fine-grained features from the encoder via skip connections.

Skip connections: The key innovation. High-resolution features from the encoder are concatenated with upsampled decoder features at corresponding resolutions. This allows the network to preserve fine details that would otherwise be lost through the bottleneck.

For image generation, the U-Net's multi-scale design is natural: generating an image requires both coarse decisions (overall composition, object placement) and fine decisions (textures, edges, exact pixel values). The encoder captures context, the bottleneck integrates global information, and the decoder renders details guided by skip connections.

The U-Net's inductive biases—locality, multi-scale hierarchy, skip connections—are well-matched to images. But they are biases nonetheless, and recent work questions whether they are necessary or merely convenient.

4.3 Transformers: Attention as Learned Connectivity

The Transformer architecture, originating in NLP, replaces fixed local connectivity with learned attention patterns:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{10}$$

Each position attends to all other positions, with attention weights determined by query-key similarity. This has profound implications:

Global receptive field: Every position can directly attend to every other position, enabling long-range dependencies without deep stacking.

Learned connectivity: Unlike convolutions with fixed spatial patterns, attention learns which positions are relevant. The network can discover that a face's lighting depends on the sun's position even if they are spatially distant.

No inductive bias: Transformers impose minimal structure—they don't assume locality or translation equivariance. This is both strength and weakness: they can learn arbitrary patterns but require more data to do so.

For images, the challenge is computational: self-attention over n positions has $O(n^2)$ complexity. A 256×256 image has $65,536$ pixels; full attention would require ~ 4 billion operations per layer. This motivates various approaches:

Patch-based processing: Divide the image into patches (e.g., 16×16) and treat each patch as a token. A 256×256 image becomes 256 tokens, making attention tractable. This is the Vision Transformer (ViT) approach.

Local + global attention: Use local attention for fine details and sparse global attention for long-range structure.

Latent-space attention: Apply attention in the compressed latent space rather than pixel space, reducing token count by the compression ratio squared.

4.4 The DiT Hypothesis: Do We Need the U-Net?

The Diffusion Transformer (DiT) directly challenges the necessity of U-Net's inductive biases. It replaces the U-Net with a standard transformer operating on latent patches, using:

Patchification: Divide the latent representation into patches, flatten, and add positional embeddings.

Transformer blocks: Standard self-attention and feedforward layers, with conditioning (timestep, class label) injected via adaptive layer normalization (AdaLN).

Unpatchification: Reshape the output tokens back into spatial form for the final image.

DiT's success demonstrates that the U-Net's specific architecture is not essential—transformers can learn effective image priors given sufficient data and compute. However, DiT is more computationally expensive and requires more training, consistent with the bias-variance tradeoff: less inductive bias means more data needed.

This finding has significant implications for scaling: transformers benefit from the extensive engineering for efficient attention (FlashAttention, tensor parallelism) developed for LLMs, potentially enabling image models to scale alongside language models.

5. The Score Function Perspective

Before diving into diffusion models (in Part II), it is valuable to understand the mathematical object at their core: the *score function*.

5.1 Score Functions and Langevin Dynamics

The score function of a distribution $p(x)$ is the gradient of its log-density:

$$s(x) = \nabla_x \log p(x) \tag{11}$$

This vector field points toward regions of higher probability. At any point x, following the score moves toward more likely configurations. Unlike the density $p(x)$ itself, the score does not require normalization—we can compute $\nabla_x \log p(x)$ without knowing the normalizing constant.

Langevin dynamics uses the score to sample from $p(x)$:

$$x_{t+1} = x_t + \epsilon \nabla_x \log p(x_t) + \sqrt{2\epsilon}\, z_t, \quad z_t \sim \mathcal{N}(0, I) \tag{12}$$

The first term moves toward high-density regions (gradient ascent on log-probability); the second term adds noise to enable exploration. As $\epsilon \to 0$ and $t \to \infty$, the distribution of x_t converges to $p(x)$ under mild conditions.

This is remarkable: if we can estimate the score function, we can sample from the distribution without ever computing densities or normalizing constants.

5.2 Score Matching: Learning Without Normalization

How do we learn the score function? The naive approach—minimize $\mathbb{E}_{p(x)}[\|s_\theta(x) - \nabla_x \log p(x)\|^2]$—requires knowing the true score, which we don't have.

Score matching provides an elegant solution. Through integration by parts, the objective can be rewritten as:

$$\mathcal{L}_{\text{SM}} = \mathbb{E}_{p(x)}\left[\frac{1}{2}\|s_\theta(x)\|^2 + \text{tr}(\nabla_x s_\theta(x))\right] \tag{13}$$

This depends only on the model s_θ and samples from $p(x)$, not the true score. However, computing $\text{tr}(\nabla_x s_\theta(x))$—the trace of the Jacobian—is expensive for high-dimensional x.

Denoising score matching offers a practical alternative. Instead of matching the score of the data distribution, match the score of a noised distribution:

$$\tilde{x} = x + \sigma z, \quad z \sim \mathcal{N}(0, I) \tag{14}$$

The score of this noised distribution has a simple form:

$$\nabla_{\tilde{x}} \log p(\tilde{x}|x) = -\frac{\tilde{x} - x}{\sigma^2} = -\frac{z}{\sigma} \tag{15}$$

Thus we can train a model to predict the noise added to each sample:

$$\mathcal{L}_{\text{DSM}} = \mathbb{E}_{x,z}\left[\|s_\theta(\tilde{x}, \sigma) - (-z/\sigma)\|^2\right] \tag{16}$$

This is the foundation of diffusion models: learn to denoise, and the denoiser provides the score function needed for sampling.

5.3 The Multi-Scale Score: Noise Scheduling

A single noise level σ is insufficient for learning the full score function. At low noise, the score has fine details but is hard to learn from limited samples. At high noise, the score is smooth but loses information about the data.

The solution is to train across multiple noise levels, using a noise schedule $\sigma_1 > \sigma_2 > \cdots > \sigma_T$. The model learns to denoise at each level:

$$\mathcal{L} = \sum_{t=1}^{T} \lambda_t \mathbb{E}_{x,z}\left[\|s_\theta(x + \sigma_t z, \sigma_t) - (-z/\sigma_t)\|^2\right] \tag{17}$$

During sampling, we start from high noise (where the distribution is nearly Gaussian) and progressively denoise toward lower noise levels, following the score at each step. This annealed Langevin dynamics traverses from the prior toward the data distribution.

The weights λ_t determine the relative importance of each noise level. Optimal weighting is an active area of research, with significant impact on sample quality.

6. Summary: The Design Space of Generative Models

Part I has established the foundational concepts that underpin all modern image generation systems. Before proceeding to specific architectures, we summarize the key design axes:

Table 1: Fundamental Tradeoffs in Generative Model Design

Design Choice	Option A	Option B
Density modeling	Explicit (tractable likelihood)	Implicit (no density, sample only)
Latent space	Continuous (VAE)	Discrete (VQ-VAE)
Compression stage	Pixel-space generation	Latent-space generation
Architecture	CNN (local inductive bias)	Transformer (learned attention)
Generation order	Parallel (diffusion)	Sequential (autoregressive)
Training signal	Reconstruction	Denoising / Score matching
Loss function	MSE (pixel)	Perceptual + Adversarial

Each choice represents a tradeoff:

Explicit vs. Implicit: Explicit models enable likelihood evaluation and stable training but constrain architecture. Implicit models are more flexible but harder to train.

Continuous vs. Discrete: Continuous latents enable gradient-based optimization but require regularization. Discrete latents interface naturally with language models but introduce non-differentiability.

Pixel vs. Latent: Pixel-space models operate directly on images but are computationally expensive. Latent-space models are efficient but limited by encoder quality.

CNN vs. Transformer: CNNs are efficient and encode useful biases but have limited receptive fields. Transformers capture global structure but require more data and compute.

Parallel vs. Sequential: Parallel generation (diffusion) is fast but requires modeling the joint distribution. Sequential (autoregressive) factorizes the distribution but introduces ordering dependencies.

Modern state-of-the-art systems combine these choices strategically: latent diffusion (latent + parallel), autoregressive image transformers (discrete + sequential), and unified multimodal models (all of the above). Understanding these foundations enables informed analysis of why certain combinations succeed.

Part II will examine the specific architectures that build on these foundations: diffusion models, rectified flow, and autoregressive approaches.

References for Part I

The foundational concepts in this section draw from the following key works (to be compiled into full bibliography):

- Variational Autoencoders: Kingma & Welling (2014), Rezende et al. (2014)
- VQ-VAE: van den Oord et al. (2017)
- VQ-GAN: Esser et al. (2021)
- Score Matching: Hyvärinen (2005), Vincent (2011)
- Denoising Score Matching: Song & Ermon (2019)
- U-Net: Ronneberger et al. (2015)
- Vision Transformer: Dosovitskiy et al. (2021)
- DiT: Peebles & Xie (2023)
- MAGVIT-v2 / LFQ: Yu et al. (2024)
- FSQ: Mentzer et al. (2024)