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 reasoning, tracing architectural developments from first principles through frontier systems. We begin with foundational theory—variational autoencoders, vector quantization, score functions, and the U-Net and Transformer architectures that underpin modern generators. We then analyze the three dominant paradigms: diffusion models (DDPM, latent diffusion, classifier-free guidance), rectified flow (Stable Diffusion 3, FLUX), and autoregressive approaches (LlamaGen, VAR, MaskGIT). The survey examines critical components including visual tokenizers, text encoders (CLIP, T5, native multimodal), and conditioning mechanisms (cross-attention, ControlNet, IP-Adapter), alongside landmark commercial systems (Imagen, DALL-E, GPT-Image, Gemini/Nano Banana, Qwen-Image). We explore the unification of image understanding and generation through architectures like Chameleon, Emu3, Show-o, and Transfusion, as well as structured generation via scene graphs, layout conditioning, neuro-symbolic methods, and programmatic synthesis. Practical considerations of training—datasets, scaling laws, distributed infrastructure, and distillation—are covered alongside evaluation frameworks spanning FID, compositional benchmarks (T2I-CompBench, GENEVAL), learned preference metrics, and human evaluation protocols. We document persistent weaknesses including compositional reasoning failures, visual quality limitations, and control gaps, then project future trends: unified multimodal models, efficiency advances, video and 3D generation, and evolving ecosystem dynamics. Throughout, we emphasize the fundamental tradeoffs—expressiveness versus tractability, quality versus speed, flexibility versus control—that shape this rapidly advancing field.

Contents

Contents

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 [1]. 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 [2]. 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 [3]. 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) [4, 5] 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. The architecture is illustrated in Figure 1.

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:

$$\begin{aligned}
\log p_\theta(x) &= \log \int p_\theta(x|z)p(z)\,dz && (2)\\
&= \log \int q_\phi(z|x)\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz && (3)\\
&\geq \int q_\phi(z|x)\log\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz && (4)\\
&= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\mathrm{KL}}(q_\phi(z|x)\|p(z)) && (5)
\end{aligned}$$

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 [6]—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* [7]: instead of comparing pixels directly, compare features extracted by a pretrained network (typically VGG [8] 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.

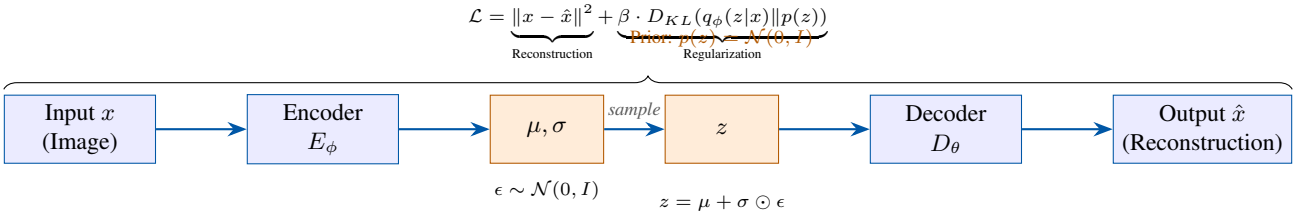

Figure 1: Variational Autoencoder architecture. The encoder maps input x to distribution parameters (μ, σ), from which latent code z is sampled via the reparameterization trick. The decoder reconstructs $\hat{x}$ from z. Training optimizes the ELBO, balancing reconstruction fidelity against latent space regularization.

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) [9] replaces the continuous latent space with a discrete codebook $\mathcal{C} = \{e_1, e_2, \ldots, e_K\}$ of K learnable embedding vectors (Figure 2). 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 [10] 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× or higher, compared to VQ-VAE which becomes visibly degraded at such ratios.

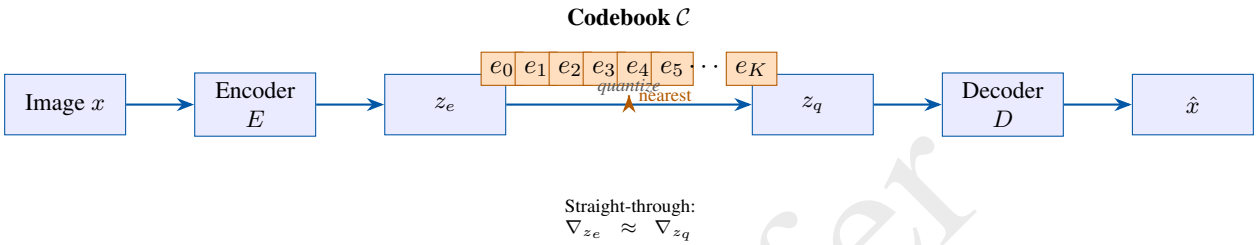

Figure 2: VQ-VAE/VQ-GAN architecture. The encoder produces continuous features z_e, which are quantized to nearest codebook entries to produce discrete tokens z_q. The decoder reconstructs from quantized codes. The straight-through estimator copies gradients through the non-differentiable quantization step.

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) [11]: 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) [12]: 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) [13] 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 [14], originally developed for image segmentation, has become the backbone of diffusion models (Figure 3). 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.

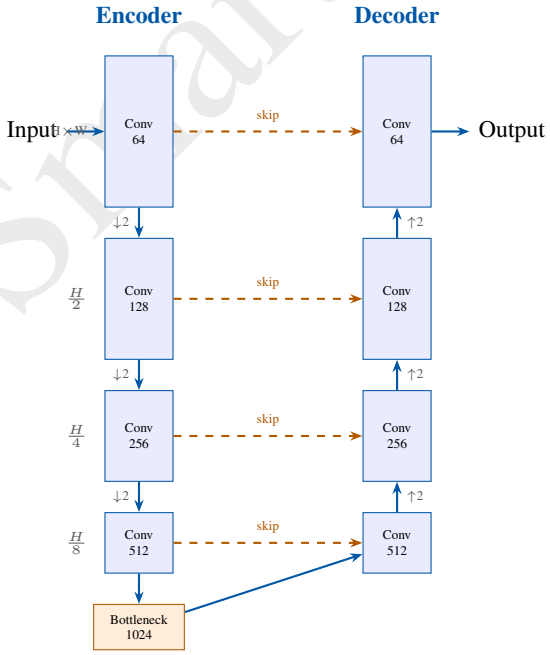

Figure 3: U-Net architecture for diffusion models. The encoder progressively downsamples the input, increasing channels while reducing spatial resolution. The bottleneck processes the most compressed representation. The decoder upsamples while receiving skip connections (dashed orange) from corresponding encoder layers, preserving fine-grained spatial details.

4.3 Transformers: Attention as Learned Connectivity

The Transformer architecture [15], 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 [16].

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) [17] directly challenges the necessity of U-Net's inductive biases. It replaces the U-Net with a standard transformer operating on latent patches (Figure 4), 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 [18], tensor parallelism) developed for LLMs, potentially enabling image models to scale alongside language models.

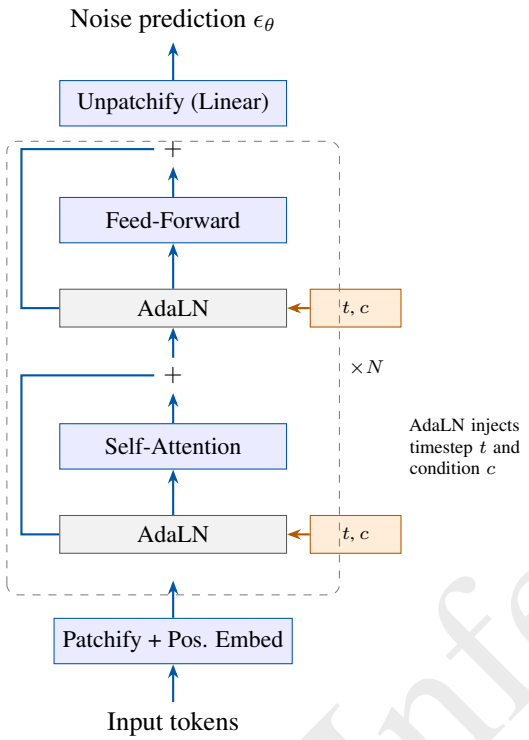

Figure 4: Diffusion Transformer (DiT) block architecture. Input latents are patchified into tokens. Each transformer block uses Adaptive Layer Normalization (AdaLN) to inject timestep t and conditioning c, followed by self-attention and feed-forward layers with residual connections. N blocks are stacked, then unpatchified to predict noise.

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 [19] 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 [20] 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 [21, 22]: 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.

7. Diffusion Models: From Theory to Practice

Diffusion models have emerged as the dominant paradigm for high-quality image generation. Building on the score function perspective established in Part I, we now examine how these theoretical foundations translate into practical architectures.

7.1 Denoising Diffusion Probabilistic Models

The Denoising Diffusion Probabilistic Model (DDPM) [22] frames generation as the reversal of a gradual noising process (Figure 5). The key insight is that while the forward noising process destroys information in a simple, tractable way, learning to reverse this process is equivalent to learning the data distribution.

7.1.1 The Forward Process

Given a data point $x_0 \sim p_{\text{data}}(x)$, the forward process produces a sequence of increasingly noisy versions $x_1, x_2, \ldots, x_T$ according to:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{18}$$

where $\{\beta_t\}_{t=1}^T$ is a variance schedule. Each step adds a small amount of Gaussian noise while slightly shrinking the signal. After sufficiently many steps, x_T is approximately standard Gaussian regardless of x_0.

A crucial property enables efficient training: we can sample x_t directly from x_0 without iterating through intermediate steps:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I) \tag{19}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. This closed form allows training on arbitrary timesteps without sequential computation.

7.1.2 The Reverse Process

The generative model learns to reverse the forward process:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{20}$$

Starting from $x_T \sim \mathcal{N}(0, I)$, we iteratively sample $x_{t-1} \sim p_\theta(x_{t-1}|x_t)$ until reaching x_0.

The mean μ_θ can be parameterized in several equivalent ways. DDPM uses noise prediction: the network $\epsilon_\theta(x_t, t)$ predicts the noise that was added, and the mean is computed as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) \tag{21}$$

The training objective simplifies to:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2\right] \tag{22}$$

This is precisely denoising score matching: predicting the noise ϵ is equivalent to estimating the score $\nabla_{x_t} \log q(x_t|x_0)$.

7.1.3 The Connection to Score Functions

The relationship between noise prediction and score estimation is:

$$\nabla_{x_t} \log q(x_t|x_0) = -\frac{\epsilon}{\sqrt{1-\bar{\alpha}_t}} \tag{23}$$

Thus $\epsilon_\theta(x_t, t) \approx -\sqrt{1-\bar{\alpha}_t} \cdot s_\theta(x_t, t)$ where s_θ is the learned score function. The DDPM sampling process is a discretization of Langevin dynamics guided by the learned score.

This connection explains why diffusion models work: they learn the score function at multiple noise levels, then use annealed Langevin dynamics to sample. The forward process defines a path from data to noise; the reverse process follows the score back.

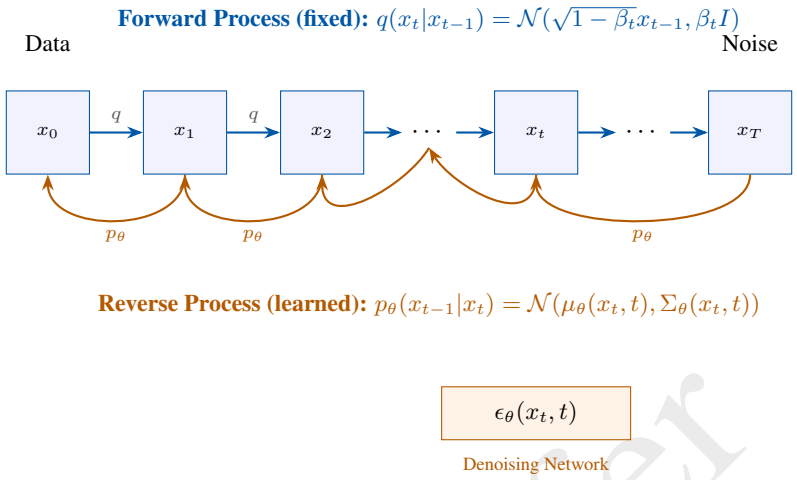

Figure 5: Diffusion model forward and reverse processes. The forward process q gradually adds Gaussian noise over T timesteps, transforming data into pure noise. The reverse process p_θ learns to denoise, iteratively recovering the data distribution. The network ϵ_θ predicts the noise to be removed at each step.

7.2 Accelerating Sampling: From DDPM to DDIM

A practical limitation of DDPM is sampling speed: generating one image requires hundreds to thousands of sequential denoising steps. Denoising Diffusion Implicit Models (DDIM) [23] address this by reinterpreting the generative process.

7.2.1 The Key Insight

DDPM's reverse process is stochastic—each step adds noise. DDIM shows that the same training objective supports a family of generative processes indexed by a parameter η:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}} \right)}_{\text{predicted } x_0} + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \cdot \epsilon_\theta(x_t, t) + \sigma_t \epsilon_t \tag{24}$$

When $\eta = 1$, this recovers DDPM. When $\eta = 0$, the process becomes deterministic—no noise is added during sampling. The deterministic case defines an ordinary differential equation (ODE) whose trajectories connect noise to data.

7.2.2 Accelerated Sampling

The ODE formulation enables larger step sizes. Instead of stepping through all T timesteps, DDIM can use a subsequence $\tau_1 < \tau_2 < \cdots < \tau_S$ with $S \ll T$. The same trained model supports this without retraining.

In practice, DDIM can produce high-quality samples in 20-50 steps compared to DDPM's 1000, a 20-50$\times$ speedup. This makes diffusion models practical for interactive applications.

7.2.3 Determinism and Interpolation

The deterministic ($\eta = 0$) case has a remarkable property: the mapping from noise to image is deterministic. The same initial noise x_T always produces the same image x_0. This enables:

- **Consistent generation**: Reproducible outputs for a given seed
- **Latent interpolation**: Interpolating between noise vectors produces smooth transitions between images
- **Inversion**: Given an image, find the noise that generates it by running the ODE backward

These properties are essential for image editing applications where consistency and controllability matter.

7.3 Latent Diffusion Models

Operating diffusion in pixel space is computationally prohibitive for high-resolution images. A 512×512 RGB image has $\sim$800K dimensions; the U-Net must process this at every denoising step. Latent Diffusion Models (LDM) [24] address this by separating perceptual compression from generative modeling (Figure 6).

7.3.1 Two-Stage Architecture

LDM consists of two components trained separately:

Stage 1: Autoencoder. A VQ-GAN or similar model learns to compress images into a lower-dimensional latent space:

$$z = E(x) \quad \text{(encode)} \tag{25}$$
$$\tilde{x} = D(z) \quad \text{(decode)} \tag{26}$$

The encoder E reduces spatial dimensions (typically by $4\times$ or $8\times$) while the decoder D reconstructs the image. This autoencoder is trained once with reconstruction and perceptual losses.

Stage 2: Diffusion in latent space. The diffusion model operates on latents z rather than pixels x:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{z,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(z_t, t, c)\|^2\right] \tag{27}$$

where c represents conditioning information (text, class label, etc.) and z_t is the noised latent.

7.3.2 Computational Benefits

For a compression ratio of f (e.g., $f = 8$ means $8\times$ spatial downsampling), the latent has $f^2 = 64$ times fewer spatial elements. Since attention complexity scales quadratically with sequence length, this yields massive computational savings:

Table 2: Computational comparison: Pixel vs. Latent Diffusion (512×512 image)

Metric	Pixel Space	Latent (f=8)
Spatial dimensions	512×512	64×64
Sequence length (patches)	1024	16
Relative attention cost	$1\times$	$\sim$0.00025$\times$

This efficiency enabled Stable Diffusion to run on consumer GPUs, democratizing high-quality image generation.

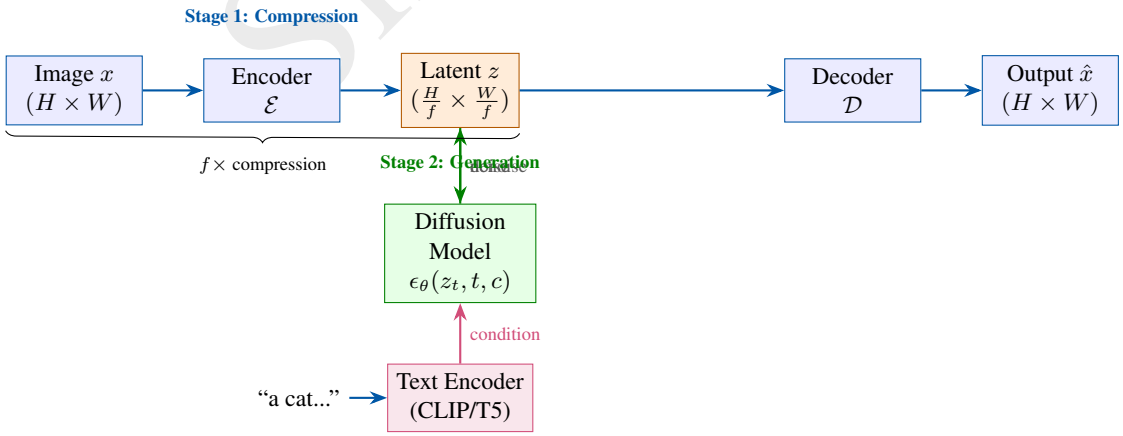

Figure 6: Latent Diffusion Model (LDM) architecture. Stage 1: A pretrained autoencoder compresses images to a lower-dimensional latent space (typically $8\times$ spatial reduction). Stage 2: Diffusion operates in this compressed space, conditioned on text embeddings from CLIP or T5. The decoder reconstructs the final image from the denoised latent.

7.3.3 Conditioning Mechanisms

LDM introduced cross-attention for flexible conditioning. Text prompts are encoded by a frozen language model (originally CLIP [25], later T5 [26]), producing a sequence of embeddings. These condition the U-Net via cross-attention layers:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{28}$$

where Q comes from the image features and K, V come from the text embeddings. This allows spatial features to selectively attend to relevant parts of the prompt.

7.4 Classifier-Free Guidance

A critical technique for improving prompt adherence is classifier-free guidance (CFG) [27]. The key observation is that conditional and unconditional models can be combined at inference time to amplify the effect of conditioning.

7.4.1 The Guidance Formula

During training, the model learns both conditional $\epsilon_\theta(x_t, t, c)$ and unconditional $\epsilon_\theta(x_t, t, \varnothing)$ predictions by randomly dropping the conditioning with some probability (typically 10-20%).

At inference, the predictions are combined:

$$\tilde{\epsilon}_\theta(x_t, t, c) = \epsilon_\theta(x_t, t, \varnothing) + w \cdot (\epsilon_\theta(x_t, t, c) - \epsilon_\theta(x_t, t, \varnothing)) \tag{29}$$

where $w > 1$ is the guidance scale. This extrapolates in the direction of the conditioning, making the output more strongly reflect the prompt.

7.4.2 The Tradeoff

Higher guidance scales produce images that more closely match the prompt but reduce diversity. At $w = 1$, we recover the unguided model. As w increases, images become more "stereotypical" representations of the prompt. Typical values range from $w = 5$ to $w = 15$ depending on the application.

Mathematically, CFG modifies the score function:

$$\tilde{s}(x_t, c) = s(x_t) + w \cdot (s(x_t|c) - s(x_t)) \tag{30}$$

This is equivalent to sampling from a tempered distribution $p(x|c)^w/Z$, which concentrates probability mass on high-likelihood regions under the conditional distribution.

7.5 Architecture Evolution: U-Net to Transformer

The architectural backbone of diffusion models has evolved significantly. Understanding this evolution illuminates broader trends in deep learning.

7.5.1 The Original U-Net Design

Early diffusion models used convolutional U-Nets with:

- ResNet blocks for local feature extraction
- Self-attention at low resolutions (e.g., 16×16, 8×8) for global context
- Cross-attention for text conditioning
- Timestep embedding via adaptive normalization (AdaGN)

This hybrid design balances efficiency (convolutions for high-resolution features) with expressiveness (attention for global reasoning).

7.5.2 The DiT Architecture

The Diffusion Transformer (DiT) [17] replaces the U-Net entirely with a vision transformer. Key design choices:

Patchification: The noisy latent is divided into non-overlapping patches, linearly projected, and combined with positional embeddings.

Conditioning via AdaLN-Zero: Timestep and class embeddings modulate layer normalization parameters. The "Zero" refers to initializing certain parameters to zero, ensuring the model behaves like an identity function at initialization.

Standard transformer blocks: Self-attention and MLP layers, following the ViT recipe.

DiT demonstrated that the U-Net's inductive biases are not necessary—pure transformers can match or exceed U-Net performance given sufficient scale. This finding is significant because it allows diffusion models to leverage transformer scaling infrastructure.

7.5.3 Scaling Properties

DiT established scaling laws for diffusion transformers: FID improves log-linearly with compute across model size, training tokens, and sampling steps. Larger models also require fewer sampling steps for equivalent quality, a favorable property for deployment.

Table 3: DiT model configurations and ImageNet 256×256 FID

Model	Parameters	GFLOPs	FID-50K
DiT-S/2	33M	6	68.4
DiT-B/2	130M	23	43.5
DiT-L/2	458M	80	23.3
DiT-XL/2	675M	119	9.62

8. Rectified Flow: Straightening the Path

While diffusion models achieve excellent quality, their iterative sampling remains computationally expensive. Rectified flow offers a principled approach to faster generation by learning straighter paths from noise to data.

8.1 The Problem with Curved Trajectories

Diffusion models learn to reverse a noising process, but the resulting trajectories from noise to data are typically curved. Curved paths require many small steps to follow accurately; taking large steps introduces discretization error.

The insight of rectified flow is that *straighter paths require fewer steps*. If we could learn a direct, linear path from noise to data, we could traverse it in very few (ideally one) steps.

8.2 Flow Matching Formulation

Flow matching [28] provides a general framework for learning continuous normalizing flows. Instead of defining a forward noising process, we directly specify the desired interpolation between noise and data.

8.2.1 Conditional Flow Matching

Given paired samples (x_0, x_1) where $x_0 \sim \mathcal{N}(0, I)$ (noise) and $x_1 \sim p_{\text{data}}$ (data), define a conditional path:

$$x_t = (1-t)x_0 + tx_1 \tag{31}$$

This is a straight line from noise to data. The velocity along this path is constant:

$$v_t = \frac{dx_t}{dt} = x_1 - x_0 \tag{32}$$

We train a neural network $v_\theta(x_t, t)$ to predict this velocity:

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t,x_0,x_1}\left[\|v_\theta(x_t,t) - (x_1 - x_0)\|^2\right] \tag{33}$$

8.2.2 Connection to Diffusion

Remarkably, flow matching with this objective trains models that can generate samples by solving an ODE:

$$\frac{dx_t}{dt} = v_\theta(x_t, t), \quad x_0 \sim \mathcal{N}(0, I) \tag{34}$$

The trained velocity field transports the noise distribution to the data distribution. Diffusion models can be viewed as a special case where the path is defined implicitly by the noising process.

8.3 Rectified Flow: Learning Straighter Paths

Rectified flow [29] extends flow matching with a procedure to iteratively straighten trajectories.

8.3.1 The Reflow Procedure

Starting with a trained flow model, we can generate paired data (x_0, x_1) by:

1. Sample noise $x_0 \sim \mathcal{N}(0, I)$
2. Solve the ODE to get $x_1 = \text{ODE}(x_0)$

These pairs define straighter paths than the original data-noise pairs because they follow the learned flow. Training a new model on these pairs produces straighter trajectories. This process can be iterated ("reflow") to progressively straighten paths.

8.3.2 Why Straighter is Better

Consider traversing a path from $t = 0$ to $t = 1$ in N steps. For a straight path, each step moves directly toward the goal regardless of step size. For a curved path, large steps overshoot, requiring correction.

The approximation error for Euler integration scales with path curvature. Straighter paths enable:

- Fewer sampling steps for equivalent quality
- Larger step sizes without quality degradation
- More predictable generation behavior

8.4 Stable Diffusion 3 and the MMDiT Architecture

Stable Diffusion 3 [30] combines rectified flow with architectural innovations in the Multimodal Diffusion Transformer (MMDiT).

8.4.1 Multimodal Architecture

Unlike previous models that use cross-attention for text conditioning, MMDiT processes text and image tokens in a unified transformer:

- Text tokens from multiple encoders (CLIP, T5) are concatenated with image latent tokens
- Self-attention operates over the combined sequence
- Separate MLPs for text and image modalities preserve their distinct characteristics

This "two-way flow of information" allows text representations to adapt based on image content, improving prompt adherence.

8.4.2 Scaling Results

SD3 demonstrated that rectified flow transformers scale effectively:

- Models from 800M to 8B parameters
- Larger models achieve comparable quality with fewer sampling steps
- Improved text rendering and compositional understanding

The combination of rectified flow (straighter paths) and transformers (scalable architecture) represents the current frontier of diffusion-based generation.

8.5 FLUX: Efficiency at Scale

FLUX models from Black Forest Labs [31] push the rectified flow paradigm further with architectural refinements optimized for efficiency.

8.5.1 Key Innovations

Parallel attention layers: Instead of sequential self-attention and cross-attention, FLUX computes them in parallel and combines the outputs. This reduces sequential dependencies and enables better hardware utilization.

Rotary positional embeddings (RoPE): Unlike absolute positional encodings, RoPE encodes relative positions through rotation matrices applied to query and key vectors. This improves generalization to different resolutions.

Flow matching training: Like SD3, FLUX uses rectified flow, enabling high-quality generation in fewer steps.

8.5.2 Model Variants

FLUX offers multiple variants targeting different use cases:

Table 4: FLUX model variants

Variant	Parameters	Steps	Use Case
FLUX.1 [schnell]	12B	4	Fast generation
FLUX.1 [dev]	12B	20-50	Quality/speed balance
FLUX.1 [pro]	12B	25+	Maximum quality

The [schnell] variant achieves remarkable quality in just 4 steps through distillation [32], making it 10× faster than standard sampling while maintaining competitive quality.

8.5.3 Why FLUX is More Efficient

Several factors contribute to FLUX's efficiency advantage:

1. **Rectified flow**: Straighter paths require fewer steps
2. **Distillation**: Student models trained to match teacher outputs in fewer steps [33]
3. **Architecture**: Parallel attention reduces memory bandwidth bottlenecks
4. **Training scale**: 12B parameters with extensive training capture more structure, reducing the burden on sampling

9. Autoregressive Image Generation

While diffusion models generate all pixels in parallel through iterative refinement, autoregressive models generate images sequentially, one token at a time. This paradigm, borrowed from language modeling, offers distinct advantages in scalability and unification with text.

9.1 The Autoregressive Factorization

Autoregressive models factorize the joint distribution as a product of conditionals:

$$p(x) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1}) \tag{35}$$

For images tokenized into a sequence $(z_1, z_2, \ldots, z_n)$, this becomes:

$$p(z) = \prod_{i=1}^{n} p(z_i | z_1, \ldots, z_{i-1}, c) \tag{36}$$

where c represents any conditioning (text prompt, class label).

9.1.1 Why This Factorization?

The autoregressive factorization has several appealing properties:

Tractable likelihood: Unlike diffusion models where likelihood computation requires expensive integration, autoregressive models provide exact log-likelihoods as a sum of log-conditionals.

No mode collapse: Training maximizes likelihood directly, avoiding the adversarial dynamics that cause GAN mode collapse.

Scalability: The same architecture (transformer decoder) that scales for language also scales for images, inheriting extensive engineering investment.

Unification: Text and image tokens can share the same vocabulary and model, enabling seamless multimodal generation.

9.1.2 The Ordering Problem

Unlike text, images have no natural sequential order. A sentence has grammatical structure that makes left-to-right generation sensible. Images have spatial structure that doesn't privilege any direction.

Common orderings include:

- **Raster scan**: Left-to-right, top-to-bottom (like reading)
- **Spiral**: From center outward or outside inward
- **Hierarchical**: Coarse-to-fine (VAR approach)
- **Random**: Different order each training step (order-agnostic training)

The choice of ordering affects what context is available when predicting each token. Raster scan means each token sees all tokens above and to the left—a particular inductive bias.

9.2 LlamaGen: Language Model Architecture for Images

LlamaGen [34] directly applies the Llama architecture to image token sequences, demonstrating that vanilla language models can achieve state-of-the-art image generation.

9.2.1 Architecture

LlamaGen uses standard Llama components:

- Decoder-only transformer with causal attention
- RMSNorm for layer normalization
- SwiGLU activation functions
- Rotary positional embeddings

The only modifications are:

- Vocabulary: Image token vocabulary (from VQ tokenizer) instead of text
- Class conditioning: Prepend class token or use AdaLN

9.2.2 Scaling Results

LlamaGen demonstrated clear scaling laws for autoregressive image generation:

Table 5: LlamaGen scaling on ImageNet 256×256

Model	Parameters	Tokens	FID
LlamaGen-L	343M	256	3.80
LlamaGen-XL	775M	256	3.39
LlamaGen-XXL	1.4B	256	3.10
LlamaGen-3B	3.1B	256	2.18

The 3.1B model achieves FID 2.18, outperforming DiT-XL/2 (FID 2.27) and demonstrating that autoregressive models can match diffusion quality.

9.2.3 Key Insights

LlamaGen's success reveals several insights:

1. **Tokenizer quality matters**: Using a high-quality VQ tokenizer (0.94 rFID reconstruction) is essential
2. **No vision-specific inductive bias needed**: Standard language model architectures suffice
3. **Scaling works**: Performance improves predictably with model size
4. **Training data quality**: Synthetic captions and data filtering significantly impact results

9.3 VAR: Next-Scale Prediction

Visual AutoRegressive modeling (VAR) [35] challenges the assumption that autoregressive generation must proceed token-by-token. Instead, it generates images scale-by-scale, from coarse to fine.

9.3.1 Multi-Scale Tokenization

VAR uses a hierarchical tokenizer that represents images at multiple resolutions. A single image produces token maps at scales 1×1, 2×2, 4×4, ..., 16×16 (for example).

The generation proceeds:

$$p(r_1, r_2, \ldots, r_K) = p(r_1) \prod_{k=2}^{K} p(r_k | r_1, \ldots, r_{k-1}) \tag{37}$$

where r_k is the token map at scale k. Each scale conditions on all coarser scales.

9.3.2 Why Next-Scale Works

This formulation has several advantages over raster-scan:

Natural hierarchy: Images have inherent multi-scale structure. Coarse features (overall composition) logically precede fine details (textures).

Global before local: The 1×1 token captures the entire image's gist. Subsequent scales refine this globally-informed representation.

Parallelism within scales: All tokens at scale k can be predicted in parallel given scales $1, \ldots, k-1$. Only K sequential steps are needed regardless of total token count.

Faster generation: With $K \approx 10$ scales, VAR is $\sim$20$\times$ faster than token-by-token generation of equivalent resolution.

9.3.3 Results

VAR achieved state-of-the-art results on ImageNet, winning the NeurIPS 2024 Best Paper Award:

Table 6: VAR results on ImageNet 256×256

Model	Parameters	FID	Steps
VAR-d16	310M	3.30	10
VAR-d20	600M	2.57	10
VAR-d24	1.0B	2.09	10
VAR-d30	2.0B	1.73	10

VAR-d30 achieves FID 1.73, surpassing both DiT and LlamaGen while requiring only 10 forward passes.

9.4 Masked Prediction: Bidirectional Generation

Between fully autoregressive (one token at a time) and fully parallel (diffusion), masked prediction models occupy a middle ground.

9.4.1 MaskGIT

MaskGIT [36] generates images through iterative refinement of masked tokens:

1. Start with all tokens masked
2. Predict all tokens in parallel
3. Keep the most confident predictions, remask the rest
4. Repeat until all tokens are unmasked

This bidirectional approach allows each prediction to condition on context from all directions, unlike causal autoregressive models.

9.4.2 Training

Training uses a BERT-style objective: randomly mask a fraction of tokens and predict the masked tokens given the unmasked ones:

$$\mathcal{L} = \mathbb{E}\left[-\sum_{i \in \text{masked}} \log p_\theta(z_i | z_{\text{unmasked}})\right] \tag{38}$$

The masking ratio is varied during training to expose the model to different levels of context.

9.4.3 Advantages and Limitations

Advantages:

- Bidirectional context improves prediction accuracy
- Parallel prediction within each iteration enables speedup
- 8-16 iterations typically suffice for high quality

Limitations:

- Training-inference mismatch: model sees ground truth during training but its own predictions during inference
- Requires confidence-based scheduling for unmasking

- Less natural fit for text-image unification than pure autoregressive

9.5 Paradigm Comparison

We summarize the three paradigms:

Table 7: Comparison of generative paradigms

Aspect	Diffusion	Autoregressive	Masked
Generation order	Parallel refinement	Sequential (token/scale)	Iterative unmasking
Steps required	20-1000	n tokens or K scales	8-16 iterations
Training objective	Denoising	Next-token prediction	Masked prediction
Likelihood	Intractable (ELBO)	Exact	Approximate
Text unification	Cross-attention	Shared vocabulary	Separate
Quality (FID)	2-3	2-3	3-5

No paradigm dominates across all criteria. Diffusion excels at quality but is slow. Autoregressive enables text unification but has ordering constraints. Masked prediction balances speed and bidirectional context but lacks exact likelihood.

The field is converging toward hybrid approaches: Show-o combines autoregressive and diffusion in one transformer; DART unifies autoregressive and diffusion training objectives; VAR uses autoregressive structure with parallel token prediction within scales.

10. Part III: Critical Components & Commercial Systems

The generative paradigms in Part II require several supporting components to work in practice. This section examines these components in depth and surveys landmark commercial systems that demonstrate their integration.

10.1 Image Tokenizers: Deep Dive

We introduced VQ-VAE and VQ-GAN in Part I. Here we examine the training dynamics, scaling challenges, and recent innovations that have made tokenizers critical to modern image generation.

10.1.1 Training Dynamics and Codebook Learning

The fundamental challenge in training VQ-based tokenizers is the interplay between encoder, decoder, and codebook. Consider the VQ-VAE loss:

$$\mathcal{L} = \underbrace{\|x - D(z_q)\|_2^2}_{\text{reconstruction}} + \underbrace{\|\text{sg}[z_e] - e\|_2^2}_{\text{codebook}} + \underbrace{\beta\|z_e - \text{sg}[e]\|_2^2}_{\text{commitment}} \tag{39}$$

where $\text{sg}[\cdot]$ denotes stop-gradient. The codebook loss pulls embeddings toward encoder outputs; the commitment loss keeps encoder outputs close to codebook entries.

Codebook collapse remains a central problem. With K codebook entries, we often observe only a small fraction ($< 10\%$) being used. Several mechanisms cause this:

- **Rich-get-richer**: Popular codes receive more gradient updates, becoming better representations, attracting more assignments
- **Encoder drift**: The encoder may move faster than codebook updates can track
- **Local optima**: Once a code becomes unused, it receives no gradients and cannot recover

Solutions include:

- **Exponential moving average (EMA)** updates for codebook entries, smoothing the learning dynamics
- **Codebook reset**: Periodically reinitialize unused codes to encoder outputs
- **Entropy regularization**: Add auxiliary loss encouraging uniform code usage
- **Factorized codes**: LFQ and FSQ eliminate learned codebooks entirely

10.1.2 Scaling Codebook Vocabulary

Larger vocabularies enable finer-grained representation but exacerbate collapse. Recent approaches address this through factorization:

Product Quantization: Decompose the latent into G groups, each with its own codebook of size K. Total vocabulary $= K^G$ but only $G \times K$ parameters. For example, 8 groups of 256 codes yields $256^8 \approx 10^{19}$ effective vocabulary with only 2048 codebook vectors.

Residual Quantization: Apply VQ iteratively to the residual error:

$$z_q^{(1)} = \text{VQ}(z_e) \tag{40}$$

$$z_q^{(2)} = \text{VQ}(z_e - z_q^{(1)}) \tag{41}$$

$$z_q = z_q^{(1)} + z_q^{(2)} + \cdots \tag{42}$$

Each level captures finer details. Used in SoundStream for audio and extended to images.

Grouped Spherical Quantization (GSQ) [37]: Projects codes onto a hypersphere before quantization, normalizing the latent space geometry. Combined with group decomposition, GSQ achieves vocabulary sizes exceeding 10^{12} with stable training.

10.1.3 Tokenizer Quality Metrics

Tokenizer quality directly impacts downstream generation. Key metrics:

- **rFID (reconstruction FID)**: FID between original and reconstructed images. State-of-the-art: < 1.0 for 256×256 images
- **Codebook utilization**: Fraction of codes used over a dataset. Target: $> 90\%$
- **PSNR/SSIM**: Pixel-level reconstruction metrics (less correlated with generation quality)
- **Downstream FID**: Generation quality when paired with a prior model

Key insight: A tokenizer with rFID 0.5 but poor codebook structure may yield worse generation than one with rFID 1.5 but well-distributed codes. The prior model must learn the code distribution; irregular distributions are harder to model.

10.1.4 Unified Tokenizers for Understanding and Generation

A frontier challenge is building tokenizers that serve both image understanding (discriminative) and generation (reconstructive). Current approaches:

Separate tokenizers: Use CLIP/DINOv2 features for understanding, VQ-GAN for generation. Simple but doubles parameters and loses potential synergy.

Dual-objective training: Train VQ-GAN with auxiliary contrastive loss. Reconstruction quality often suffers.

Semantic tokenizers: TiTok [38] compresses images to just 32 tokens by learning semantically meaningful codes. Achieves reasonable reconstruction and competitive understanding benchmarks.

The Qwen-Image approach feeds images to both a VL encoder (Qwen2.5-VL for semantics) and VAE encoder (for reconstruction), fusing representations for generation. This dual-encoding enables semantic-aware editing while maintaining visual fidelity.

10.2 Text Encoders: From CLIP to Native Multimodal

Text conditioning is critical for controllable generation. The choice of text encoder profoundly affects both prompt adherence and image quality.

10.2.1 CLIP: Contrastive Language-Image Pretraining

CLIP [25] trains image and text encoders jointly with a contrastive objective: matching image-text pairs should have higher similarity than non-matching pairs. The text encoder (a Transformer) produces embeddings that live in a shared space with images.

Strengths for generation:

- Embeddings capture visual semantics ("a dog" is close to dog images)
- Zero-shot alignment with arbitrary text
- Relatively compact representations (77 tokens max, 768/1024 dimensions)

Limitations:

- Bag-of-concepts behavior: "A dog chasing a cat" ≈ "A cat chasing a dog"
- Limited understanding of negation, counting, spatial relations
- Short context window prevents detailed descriptions

10.2.2 T5: Text-to-Text Transfer Transformer

Google's Imagen [39] demonstrated that large frozen language models outperform CLIP for text-to-image generation. T5-XXL (11B parameters) as the text encoder achieved state-of-the-art FID while CLIP-based models lagged.

Key finding: Scaling the text encoder improves generation quality more than scaling the image model. This suggests the bottleneck is often text understanding, not image synthesis.

T5 provides:

- Richer semantic representations from language model pretraining
- Better compositional understanding
- Longer context (512+ tokens)

However, T5 embeddings are not aligned to images, requiring the diffusion model to learn the mapping.

10.2.3 Multi-Encoder Fusion

Modern systems combine multiple text encoders:

SDXL [40]: Uses CLIP-L (language) + CLIP-G (giant) + optional refiner. The two CLIP models capture different aspects of the prompt.

SD3/FLUX: Uses CLIP-L + CLIP-G + T5-XXL. CLIP provides image-aligned semantics; T5 provides deep language understanding. Embeddings are concatenated and fed to the transformer.

Integration approaches:

- **Concatenation**: Stack encoder outputs along sequence dimension
- **Cross-attention**: Image model attends to text embeddings
- **Pooled conditioning**: Global text embedding added to timestep embedding

10.2.4 Native Multimodal Encoders

The latest systems (GPT-Image, Nano Banana) use natively multimodal models where text and image understanding are unified:

- No separate text encoder—the base LLM understands both modalities
- Text conditioning is implicit through the model's language understanding
- Enables conversational refinement: "make the sky more blue"
- Real-world knowledge from language pretraining transfers to images

This represents a paradigm shift from "text encoder + image generator" to "multimodal model with generation capability."

10.3 Conditioning Mechanisms

Beyond text, images can be conditioned on layouts, depth maps, poses, and reference images. The conditioning mechanism determines how control signals influence generation.

10.3.1 Cross-Attention Conditioning

The dominant approach in diffusion transformers (Figure 7). Given image features x and conditioning c:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{43}$$

where $Q = W_Q x$, $K = W_K c$, $V = W_V c$.

Cross-attention allows each image token to attend to all conditioning tokens, enabling fine-grained text-image correspondence. The attention maps reveal which image regions correspond to which words.

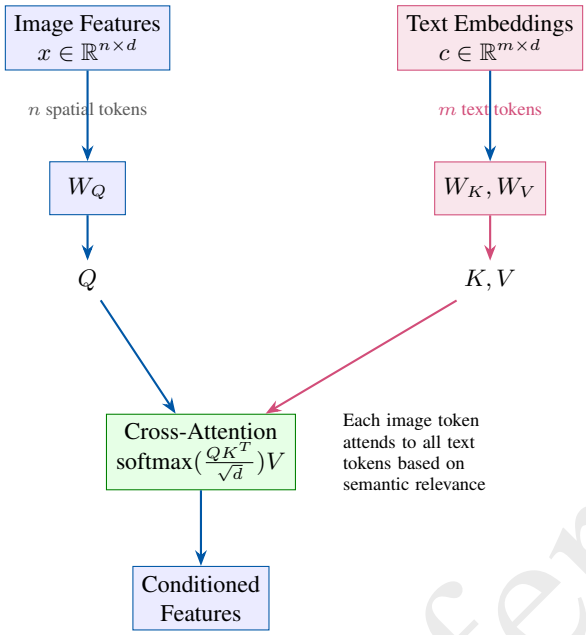

Figure 7: Cross-attention conditioning mechanism. Image features provide queries (Q), while text embeddings provide keys (K) and values (V). The attention operation allows each spatial location in the image to selectively attend to relevant words in the prompt, enabling fine-grained text-to-image correspondence.

10.3.2 Classifier-Free Guidance (CFG)

CFG [27] improves prompt adherence without a separate classifier. During training, randomly drop the conditioning (e.g., 10% of the time). During inference:

$$\tilde{\epsilon}_\theta = \epsilon_\theta(x_t, \varnothing) + w \cdot (\epsilon_\theta(x_t, c) - \epsilon_\theta(x_t, \varnothing)) \tag{44}$$

The guidance scale w controls the strength:

- $w = 1$: Standard conditional generation
- $w > 1$: Amplified conditioning (more prompt adherence, less diversity)
- $w = 7.5$: Common default for Stable Diffusion
- $w > 15$: Often produces artifacts

CFG requires two forward passes per step (conditional and unconditional), doubling compute. Distilled models (SDXL-Turbo, FLUX-schnell) bake guidance into the weights.

10.3.3 ControlNet: Structural Conditioning

ControlNet [41] adds spatial control (edges, depth, pose) to pretrained diffusion models. The key insight: copy the encoder of a pretrained U-Net, train only the copy on control signals, and add outputs to the original model via zero-initialized convolutions.

Architecture:

1. Frozen pretrained diffusion model
2. Trainable copy of encoder blocks
3. Control signal (e.g., Canny edges) input to trainable copy
4. Zero-conv layers connect trainable copy to frozen decoder

Zero convolutions (1×1 convs initialized to zero) ensure the ControlNet initially has no effect, enabling stable fine-tuning without destroying pretrained knowledge.

ControlNet enables diverse applications: pose-guided generation, depth-conditioned inpainting, edge-to-image translation, all while preserving the base model's text understanding.

10.3.4 IP-Adapter: Image Prompt Conditioning

IP-Adapter [42] enables conditioning on reference images through a lightweight adapter. A CLIP image encoder extracts features from the reference image, which are projected and added to the cross-attention layers.

This enables:

- Style transfer from reference images
- Character consistency across generations
- Composition of multiple reference images

Combined with ControlNet, IP-Adapter allows controlling both structure (via edges/depth) and style (via reference).

10.4 Landmark Commercial Systems

We survey major commercial image generation systems, examining their architectural innovations and practical contributions.

10.4.1 Imagen (Google, 2022)

Imagen [39] established several principles that influenced subsequent work:

Architecture: Cascaded diffusion with three stages:

1. Base model: 64×64 generation
2. Super-resolution 1: $64\times64 \rightarrow 256\times256$
3. Super-resolution 2: $256\times256 \rightarrow 1024\times1024$

Key innovations:

- **Frozen T5-XXL**: Demonstrated that large language models beat CLIP for text encoding
- **Dynamic thresholding**: Clipping technique for high CFG scales without saturation
- **Efficient U-Net**: Simplified architecture with faster convergence
- **DrawBench**: Comprehensive evaluation benchmark for compositional prompts

Finding: "Scaling the text encoder is more important than scaling the diffusion model." This redirected field attention toward text understanding.

10.4.2 DALL-E Series (OpenAI, 2021-2025)

OpenAI's DALL-E series evolved through distinct architectural phases:

DALL-E (2021) [43]: Autoregressive transformer generating image tokens. Used dVAE (discrete VAE) tokenizer. 12B parameters. Demonstrated zero-shot text-to-image at scale.

DALL-E 2 (2022): Diffusion-based with "unCLIP" architecture:

1. CLIP text encoder produces text embedding
2. Prior model maps text embedding to CLIP image embedding
3. Decoder (diffusion) generates image from CLIP image embedding

This two-stage approach separated semantic planning (prior) from pixel generation (decoder).

DALL-E 3 (2023) [44]: Key insight was **caption quality**. Trained a captioning model to generate detailed descriptions, then retrained on (image, detailed caption) pairs. This dramatically improved prompt adherence without architectural changes.

Also introduced GPT-4 integration: users describe intent, GPT-4 generates optimized prompts.

GPT-Image-1 (April 2025): Native multimodal generation within GPT-4o:

- No separate text encoder—uses GPT-4o's language understanding directly
- Conversational editing: "make her smile," "add a hat"
- Up to 4096×4096 resolution
- Integrated with GPT's world knowledge and reasoning

GPT-Image-1.5 (December 2025): Improved version with $4\times$ faster generation, better instruction following, and facial consistency across edits.

The evolution from DALL-E to GPT-Image represents the shift from "diffusion model with text encoder" to "language model with image generation capability."

10.4.3 Nano Banana / Gemini Image (Google, 2025)

Google's Nano Banana series represents natively multimodal image generation built on the Gemini foundation models:

Nano Banana (August 2025) = Gemini 2.5 Flash Image:

- Built on Gemini 2.5 Flash architecture
- Multi-turn conversational editing
- Subject consistency across edits
- Multi-image fusion (combine elements from multiple photos)
- 1024×1024 maximum resolution
- SynthID watermarking for provenance

Nano Banana Pro (November 2025) = Gemini 3 Pro Image:

- Built on Gemini 3 Pro
- **Search grounding**: Can query Google Search for factual accuracy in infographics
- Up to 4K resolution
- Superior text rendering across multiple languages
- Advanced controls: camera angle, lighting, depth of field, color grading
- 14 input reference images for composition

Key architectural insight: By building on a frontier LLM (Gemini), the image generator inherits:

- World knowledge (accurate infographics, historical scenes)
- Multilingual understanding (text rendering in any language)
- Reasoning capabilities (complex prompt interpretation)
- Long context (detailed multi-part instructions)

The "search grounding" capability is unique: when generating an infographic about "France's GDP," the model can retrieve current data before generation.

10.4.4 Qwen-Image (Alibaba, 2025)

Qwen-Image represents the state-of-the-art in open-source image generation:

Architecture: 20B parameter MMDiT (Multimodal Diffusion Transformer):

- Text input: Qwen2.5-VL (vision-language model) for semantic understanding
- Image input: VAE encoder for reconstruction features
- Generation: MMDiT with joint text-image attention
- Dual encoding: Separate semantic and reconstructive representations

Training innovations:

- Progressive curriculum: non-text → simple text → complex text → paragraphs
- Multi-task training: T2I (text-to-image) + TI2I (text-image-to-image editing) + I2I (reconstruction)
- Extensive recaptioning for training data quality

Key capabilities:

- **Native Chinese text rendering**: First model to accurately render complex Chinese characters, paragraphs, and mixed Chinese-English text
- **High-fidelity editing**: Change hairstyles, backgrounds, poses while preserving identity
- **Understanding tasks**: Depth estimation, edge detection, segmentation

Qwen-Image-2512 (December 2025):

- Dramatically reduced "AI look" in human portraits
- Finer natural textures (fur, water, landscapes)
- Ranked as strongest open-source model on AI Arena blind evaluation
- Apache 2.0 license (fully open for commercial use)

10.4.5 Comparative Analysis

Table 8: Comparison of major commercial image generation systems

System	Architecture	Text Encoder	Key Innovation
Imagen (2022)	Cascaded U-Net	T5-XXL (frozen)	LLM text encoder matters most
DALL-E 3 (2023)	Diffusion	CLIP + GPT-4	Caption quality is crucial
SD3 (2024)	MMDiT + RF	CLIP + T5	Rectified flow, joint attention
FLUX (2024)	DiT + RF	CLIP + T5	12B efficiency, parallel attention
GPT-Image (2025)	Native multimodal	GPT-4o native	LLM is the generator
Nano Banana Pro	Native multimodal	Gemini 3 native	Search grounding for facts
Qwen-Image (2025)	MMDiT 20B	Qwen2.5-VL	Chinese text, open source

Trend: The field is converging on natively multimodal architectures where understanding and generation share the same foundation model. Separate text encoders and diffusion models are giving way to unified systems where the LLM itself generates images.

Open vs. Closed: Qwen-Image and FLUX represent the strongest open models, while GPT-Image and Nano Banana Pro demonstrate capabilities (search grounding, conversational refinement) not yet available in open systems.

10.5 Summary: The Component Stack

Modern image generation systems combine:

1. **Tokenizer**: Converts images to/from discrete or continuous latents. VQ-GAN variants dominate; scaling requires factorization (LFQ, FSQ, GSQ).
2. **Text encoder**: CLIP for alignment, T5 for semantics, or native multimodal understanding. Multi-encoder fusion common in diffusion; unified in LLM-based systems.
3. **Conditioning**: Cross-attention for text; ControlNet for structure; IP-Adapter for style. CFG amplifies prompt adherence at diversity cost.
4. **Generator**: Diffusion/flow for quality; autoregressive for unification; masked for speed. DiT/MMDiT architectures dominate at scale.

The architectural choices interact: native multimodal systems eliminate the text encoder but require massive scale; separate components allow modular improvement but miss synergies.

11. Part IV: Unified Understanding & Generation

A fundamental question in multimodal AI is whether understanding and generation should share the same model. Humans seamlessly perceive and create images using the same visual system. Can AI achieve similar unification, and what are the architectural paths to get there?

11.1 The Unification Challenge

11.1.1 Why Unify?

Separate models for understanding (VLMs) and generation (diffusion/AR) have achieved impressive results independently. Why pursue unification?

Shared representations: Understanding and generation may benefit from the same underlying visual representations. A model that can generate realistic dogs likely understands dog anatomy; a model that understands spatial relationships should generate them correctly.

Emergent capabilities: Unified models may exhibit emergent behaviors impossible for specialists. Reasoning about hypotheticals ("what would this room look like with blue walls?"), visual analogies, and in-context visual learning all require tight coupling of understanding and generation.

Efficiency: A single model serving both tasks requires less total compute and memory than two separate models. Shared parameters amortize training cost.

Consistency: Separate models may have inconsistent internal representations. A unified model maintains coherent beliefs about visual content across tasks.

11.1.2 The Core Tension

Understanding and generation optimize different objectives:

Understanding (discriminative): Map images to semantic representations. Typically trained with contrastive or classification losses. Representations should be *invariant* to irrelevant details (lighting, viewpoint).

Generation (reconstructive): Map representations back to pixels. Trained with reconstruction or adversarial losses. Representations must preserve *all* visual details for faithful reconstruction.

This creates a fundamental tension: understanding benefits from discarding information; generation requires preserving it. Unified architectures must navigate this tradeoff.

11.1.3 Architectural Approaches

Four main approaches to unification have emerged:

1. **Separate encoders, shared decoder**: Use different encoders for understanding (CLIP/DINOv2) and generation (VAE), but share the language model backbone. (NExT-GPT, SEED)
2. **Discrete tokens in LLM vocabulary**: Quantize images to discrete tokens, add to LLM vocabulary, train on interleaved text-image sequences. (Chameleon, Emu3)
3. **Hybrid continuous-discrete**: Use continuous features for understanding, discrete tokens for generation, with adapters connecting them. (Show-o, VILA-U)
4. **Native multimodal**: Train a single model from scratch on all modalities jointly, with shared architecture throughout. (Gemini, GPT-4o)

11.2 Vision-Language Models for Understanding

Before examining unified models, we review the VLM architectures that established the understanding side of the equation.

11.2.1 The LLaVA Paradigm

LLaVA (Large Language and Vision Assistant) [45] established the dominant VLM architecture:

1. **Vision encoder**: Pretrained CLIP ViT extracts image features
2. **Projection layer**: MLP maps vision features to LLM embedding space
3. **Language model**: Pretrained LLM (Vicuna/LLaMA) processes interleaved text and projected image tokens

Training stages:

- Stage 1: Freeze vision encoder and LLM, train only projector on image-caption pairs
- Stage 2: Unfreeze LLM, fine-tune on visual instruction data

This simple recipe—frozen vision encoder, lightweight adapter, instruction-tuned LLM—proved remarkably effective. LLaVA-1.5 with a 2-layer MLP projector achieved strong results across VQA, captioning, and reasoning benchmarks.

11.2.2 Scaling Vision Encoders

Subsequent work explored larger and more sophisticated vision encoders:

InternVL [46]: Scales the vision encoder to 6B parameters (InternViT-6B), trained with contrastive learning on large image-text datasets. The larger vision encoder captures finer visual details.

Qwen-VL: Uses a custom vision encoder with dynamic resolution support. Images are processed at their native resolution (up to 4K) rather than fixed 336×336, preserving fine-grained details.

Key insight: VLM performance scales with vision encoder quality. The LLM's language capabilities are necessary but not sufficient; visual understanding bottlenecks often dominate.

11.2.3 Resolution and Efficiency

Processing high-resolution images is computationally expensive. A 1024×1024 image at 14×14 patch size produces 5,329 tokens—more than most LLM context windows.

Solutions:

- **Dynamic resolution**: Process images at variable sizes based on content (Qwen-VL, InternVL-1.5)
- **Token compression**: Reduce visual tokens via pooling or learned compression (LLaVA-Next)
- **Multi-scale features**: Extract features at multiple resolutions, concatenate (SPHINX)
- **Tile-based processing**: Split large images into tiles, process independently, aggregate (LLaVA-NeXT)

11.2.4 Limitations of Understanding-Only VLMs

Current VLMs excel at perception but cannot generate:

- Can describe an image but not create one from description
- Can answer "what color is the car?" but not "show me the car in red"
- Cannot perform visual reasoning that requires imagining alternatives

This asymmetry motivates unified architectures.

11.3 Unified Generation Systems

Several architectures attempt to add generation capability to the VLM paradigm.

11.3.1 Chameleon: Unified Token Sequences

Meta's Chameleon [47] represents the most direct approach to unification: treat images as tokens in the LLM vocabulary.

Architecture:

- Image tokenizer: VQ-VAE with 8,192 codebook entries, producing 1,024 tokens per 512×512 image
- Unified vocabulary: 65,536 text tokens + 8,192 image tokens = 73,728 total
- Single transformer: Standard decoder-only architecture, trained on interleaved text-image sequences

Training: Mixed-modal sequences where text and image tokens are interleaved:

```
[TEXT] A photo of a cat [IMG_1] [IMG_2] ... [IMG_1024] [TEXT] sitting on a mat
```

The model learns to predict text given images (understanding) and images given text (generation) with the same next-token objective.

Results: Chameleon-34B achieves competitive performance on both image understanding (VQA, captioning) and generation (FID, CLIP score) benchmarks, though not state-of-the-art on either.

Key insight: Unification is possible with minimal architectural changes—just expand the vocabulary. However, the discrete tokenizer becomes a bottleneck; VQ-VAE reconstruction quality limits generation fidelity.

11.3.2 Emu3: Next-Token Prediction for Everything

Emu3 [48] extends the Chameleon approach to video:

Tokenization: SBER-MoVQ tokenizer converts images/video to discrete tokens. A 512×512 image becomes 64×64 = 4,096 tokens (more aggressive compression than Chameleon).

Training data: Interleaved sequences of text, images, and video. The model learns temporal dynamics from video prediction.

Key finding: A single next-token prediction model can achieve strong performance across text, image understanding, image generation, and video generation. Specialization is not required.

11.3.3 Show-o: Unifying Autoregressive and Diffusion

Show-o [49] observes that autoregressive and diffusion serve different purposes:

- Autoregressive: Good for sequential reasoning, text generation
- Diffusion: Good for parallel refinement, image generation

Rather than forcing both into one paradigm, Show-o uses both within a single transformer:

Architecture:

- Shared transformer backbone processes all tokens
- Text tokens: Causal attention mask, autoregressive generation
- Image tokens: Bidirectional attention within image, diffusion-style denoising

Training: Joint objective combining:

- Next-token prediction loss for text
- Denoising loss for images

Inference:

- Understanding: Encode image tokens, autoregressively generate text response
- Generation: Autoregressively generate text plan, then denoise image tokens in parallel

Show-o achieves strong performance on both understanding and generation while respecting the different inductive biases each modality prefers.

11.3.4 Transfusion: Hybrid Training Objectives

Transfusion [50] takes a similar hybrid approach but at the loss level rather than attention level:

Key idea: Train on both discrete (text) and continuous (image) representations simultaneously:

- Text: Standard cross-entropy loss on discrete tokens
- Images: Diffusion loss on continuous latent patches

Both losses backpropagate through the same transformer. The model learns shared representations that support both modalities.

Advantage: Avoids the VQ tokenizer bottleneck by keeping images in continuous space. Generation quality approaches specialized diffusion models.

11.3.5 SEED and NExT-GPT: Modular Unification

An alternative to end-to-end training is modular composition:

SEED [51]:

- Discrete visual tokens compatible with LLM training
- Separate diffusion decoder (SDXL) for high-quality generation
- LLM generates semantic tokens; decoder renders to pixels

NExT-GPT [52]:

- Encoders: ImageBind for multimodal understanding
- LLM: Vicuna as reasoning backbone
- Decoders: Separate specialist decoders for image/audio/video
- Adapters: Lightweight projectors connecting components

Tradeoff: Modular approaches leverage pretrained specialists (better individual quality) but may lack tight integration (weaker emergent capabilities).

11.4 Native Multimodal LLMs

The frontier approach trains multimodal models from scratch rather than adapting unimodal components.

11.4.1 Gemini Architecture

Google's Gemini [53] was trained natively on text, images, audio, and video:

Key design choices:

- **Unified tokenization**: All modalities converted to tokens in shared space
- **Interleaved training**: Mixed sequences of text, image patches, audio features
- **Sparse MoE**: Mixture-of-Experts for efficient scaling (Gemini 1.5+)
- **Long context**: Up to 10M tokens (Gemini 1.5 Pro)

Capabilities enabled by native multimodality:

- Cross-modal reasoning: "Describe the sound this image would make"
- In-context multimodal learning: Few-shot image classification from examples
- Interleaved generation: Documents with text and images

Gemini's image generation (Nano Banana/Gemini Image) builds on this foundation, adding generation-specific training on top of the unified base.

11.4.2 GPT-4o and Native Image Generation

OpenAI's GPT-4o represents the tightest integration of understanding and generation:

Architecture (inferred from behavior):

- Single transformer processes text, images, and audio natively
- Image generation integrated as a capability, not a separate model
- Conversational context maintained across modality switches

Distinctive capabilities:

- **Conversational editing**: "Make her smile" operates on the image in context
- **Consistent characters**: Generate the same person across multiple images
- **World knowledge**: Accurately depicts known entities, locations, styles
- **Instruction following**: Complex multi-step image modification

GPT-Image-1.5 (December 2025) demonstrated that native multimodal training can match or exceed specialized diffusion models on generation quality while providing superior controllability.

11.4.3 The Quality Gap

A persistent question: do unified models match specialized ones?

Table 9: Unified vs. specialized model quality (approximate)

Task	Specialist	Unified	Gap
Image understanding (VQA)	90+	85-90	Small
Image generation (FID)	2-3	4-6	Moderate
Text generation (perplexity)	Best	Comparable	Minimal
Cross-modal reasoning	N/A	Strong	Unique

Unified models typically lag specialists by 10-20% on individual benchmarks but enable capabilities (cross-modal reasoning, consistent editing) that specialists cannot perform.

The gap is closing. Chameleon-34B underperformed SD-XL significantly; GPT-Image-1.5 and Nano Banana Pro are competitive with FLUX on many prompts. Native multimodal training at sufficient scale may eliminate the specialist advantage.

11.5 Open Problems in Unification

11.5.1 The Tokenizer Bottleneck

Discrete tokenizers limit generation quality. VQ-VAE reconstruction errors compound through generation:

- Tokenizer rFID: 1.0 (excellent)
- Generation from perfect tokens: FID 3.0
- Generation from predicted tokens: FID 5.0+

Potential solutions:

- Continuous representations (Transfusion approach)
- Higher-quality tokenizers (MAGVIT-v2, TiTok)
- Hybrid discrete-continuous (coarse discrete, continuous refinement)

11.5.2 Understanding vs. Generation Features

What makes a good understanding representation differs from generation:

- **Understanding**: Semantic, invariant, compressed
- **Generation**: Detailed, equivariant, high-dimensional

Current unified models either compromise on both or use separate pathways. A principled solution—representations that are simultaneously semantic and detailed—remains elusive.

11.5.3 Training Data and Objectives

Unified training requires balancing multiple objectives:

- Text understanding and generation
- Image understanding (classification, VQA, captioning)
- Image generation (reconstruction, text-to-image)
- Cross-modal tasks (image-text matching, visual reasoning)

How to weight these objectives, what data mixtures to use, and how to prevent catastrophic forgetting across tasks are active research questions.

11.5.4 Evaluation

No comprehensive benchmark evaluates unified capabilities:

- Understanding benchmarks (VQA, GQA) don't test generation
- Generation benchmarks (FID, CLIP score) don't test understanding
- No standard benchmark for "edit this image" or "imagine what happens next"

The field needs evaluation frameworks that capture the unique value of unification.

11.6 Summary: The Path to Unification

The evolution toward unified models follows a clear trajectory:

1. **Separate specialists** (2020-2022): CLIP for understanding, DALL-E/SD for generation. Best individual quality but no interaction.
2. **Modular composition** (2023): LLaVA + Stable Diffusion via adapters. Understanding informs generation but not vice versa.
3. **Shared vocabulary** (2024): Chameleon, Emu3. True unification via discrete tokens, but quality limited by tokenizer.
4. **Hybrid objectives** (2024-2025): Show-o, Transfusion. Respect modality differences while sharing representations.
5. **Native multimodal** (2025+): GPT-4o, Gemini 3. Full unification at scale, closing quality gap.

The field is converging on native multimodal training at scale. The remaining questions are practical: how much data, what architectures, which training recipes yield the best unified models?

12. Part V: Programmatic & Structured Generation

Text-to-image models frequently fail on compositional prompts—"two red apples on a blue plate" might yield three apples or miscolored objects. The core problem is that natural language conflates multiple constraints into a single embedding, creating ambiguities that diffusion models cannot reliably resolve. Scene graphs, layouts, programs, and neuro-symbolic approaches address this by decomposing generation into structured representations where objects, attributes, and relationships are explicitly specified.

12.1 Why Structured Representations Help

The failure modes of text conditioning are well-documented: *catastrophic neglect* (omitting prompted objects), *attribute leakage* (swapping colors between objects), and *relationship ambiguity* (misinterpreting spatial predicates). These arise because CLIP-style text encoders compress compositional semantics into fixed-dimensional vectors, losing the discrete structure of language.

Structured representations solve this through **explicit decomposition**. A scene graph $G = (O, E)$ separates nodes O (objects with categorical labels and attributes) from edges E (relationships with semantic predicates). The prompt "a dog chasing a cat on grass" becomes three nodes (dog, cat, grass) with explicit edges (dog→chasing→cat, cat→on→grass). Each element receives independent encoding, preventing the entanglement that causes neural networks to conflate "red book, yellow table" into ambiguous color assignments.

The architectural implication is significant: instead of a single text embedding controlling the entire generation, structured methods inject **per-object conditioning** through specialized attention mechanisms. This architectural decomposition mirrors the semantic decomposition of the input representation.

12.2 Scene Graph to Image Generation

12.2.1 Formal Representation

A scene graph represents an image as a directed graph where nodes are objects and edges are relationships:

- **Nodes** $O = \{o_i\}_{i=1}^{n}$ with semantic labels $c_i \in \mathcal{C}^O$ (category vocabulary)
- **Edges** $E = \{e_{ij}\}$ with relationship labels $r_{ij} \in \mathcal{C}^R$ (predicate vocabulary)
- **Triplets** $T = \{(o_i, e_{ij}, o_j)\}$ representing $\langle$subject, predicate, object$\rangle$

Modern systems enrich this with bounding boxes $b_i = (x, y, w, h)$ and CLIP embeddings for open-vocabulary generalization.

12.2.2 SG2Im: The Foundational Pipeline

SG2Im [54] established the canonical two-stage approach: **graph→layout→image** (Figure 8). Graph convolutional networks process scene graphs to compute object embeddings, which are decoded into bounding boxes and segmentation masks forming a scene layout. A cascaded refinement network then synthesizes pixels from this layout.

This separation—explicit layout prediction before pixel generation—remains influential. It enables interpretable intermediate representations and modular improvement of each stage.

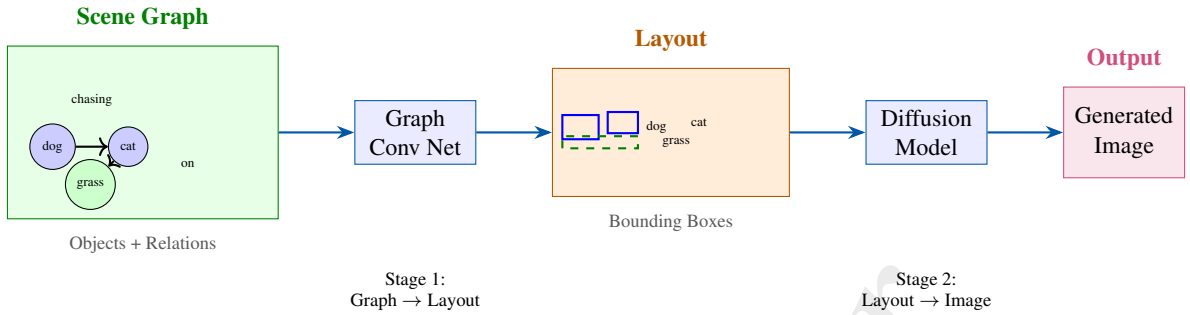

Figure 8: Scene graph to image generation pipeline. A scene graph encodes objects as nodes and relationships as edges. Graph convolutional networks process the graph to predict a spatial layout (bounding boxes). A diffusion model then generates the final image conditioned on the layout, ensuring objects appear in specified locations with correct relationships.

12.2.3 DisCo: Disentangled Scene Composition

DisCo [55] (NeurIPS 2024) represents current state-of-the-art by jointly modeling **disentangled layout and semantics**. The Semantics-Layout VAE (SL-VAE) uses a graph union encoder to parameterize both spatial positions and semantic embeddings in a shared Gaussian latent space.

The key innovation is **Compositional Masked Attention (CMA)**, inserted between self-attention and cross-attention in diffusion U-Nets:

$$C = \{s_i \otimes F(b_i)\}, \quad M_{i,j} = \begin{cases} 1 & \text{if pixels } i,j \text{ belong to same object} \\ -\infty & \text{otherwise} \end{cases} \tag{45}$$

The attention mask M prevents information mixing between objects, addressing attribute leakage. On COCO-Stuff, DisCo achieves FID 30.8 and IS 23.1, substantially outperforming prior methods.

12.2.4 Diffusion-Based Scene Graph Methods

Several approaches integrate scene graphs with diffusion models:

- **SGDiff**: Uses contrastive pre-training to learn scene graph embeddings aligned with images, bypassing explicit layout prediction
- **SceneGenie**: Provides training-free guidance by computing classifier gradients from scene graph constraints
- **R3CD**: Adds relation-aware contrastive control during denoising

12.2.5 Datasets and Evaluation

Visual Genome provides 108,077 image-scene graph pairs, typically filtered to 178 object categories and 45 relationship types. **COCO-Stuff** synthesizes scene graphs geometrically from 80 thing + 91 stuff categories.

Evaluation uses FID/IS for quality, plus specialized metrics: G2I-ACC (graph-to-image retrieval), ACCattr (CLIP-based attribute accuracy), and SGScore for object presence and relationship fidelity.

12.3 Layout-Based Conditioning

12.3.1 GLIGEN: Grounded Language-to-Image Generation

GLIGEN [56] extends Stable Diffusion by inserting **gated self-attention layers** that inject grounding information while preserving pretrained knowledge. Each grounding entity has two components: a semantic representation (encoded text/image describing the object) and a spatial location (Fourier-encoded bounding box or keypoints).

The gating mechanism computes:

$$h' = h + \beta \cdot \tanh(\gamma) \cdot \text{Attn}(h, [h; \text{grounding\_tokens}]) \tag{46}$$

where β is learnable and γ controls information flow. Original model weights remain **frozen**, enabling plug-and-play spatial control. GLIGEN achieves zero-shot performance on COCO and LVIS outperforming supervised layout-to-image baselines by large margins.

12.3.2 Training-Free Attention Manipulation

Several methods improve compositionality without retraining:

Attend-and-Excite [57] addresses catastrophic neglect through *Generative Semantic Nursing*—intervening during inference to strengthen cross-attention to all subject tokens. The excitation loss:

$$\mathcal{L} = 1 - \min_j(\max(A_j)) \tag{47}$$

encourages all subject tokens to have high activation somewhere in the image. At each denoising timestep, gradients backpropagate through the U-Net to update latent representations.

BoxDiff [58] applies three spatial constraints on cross-attention maps: inner-box (high attention inside bounding box), outer-box (suppressed attention outside), and corner (boundaries align with box corners). Combined with GLIGEN, BoxDiff improves AP from 29.7 to 40.2 on open-world layouts.

RealCompo [59] introduces a **dynamic balancer** combining text-to-image and layout-to-image model predictions during inference. The insight is that T2I models excel at realism while L2I models achieve compositionality—early-stage denoising determines positions, requiring careful fusion throughout generation.

12.3.3 LLM-Based Layout Generation

LLM-grounded Diffusion (LMD) [60] uses two stages: an LLM (GPT-3.5/4 or Mixtral) generates captioned bounding boxes with background and negative prompts via in-context learning; then layout-to-image generation applies training-free cross-attention control or GLIGEN adapters.

LLM Blueprint [61] handles complex prompts through two-phase generation: global scene generation from LLM-extracted blueprints, then iterative box-level refinement using diffusion conditioned on masks and reference images.

12.4 Neuro-Symbolic Image Generation

The core appeal of neuro-symbolic approaches is **verifiable control**: ensuring generated images satisfy explicit constraints rather than merely correlating with them.

12.4.1 Constraint Satisfaction for Spatial Layouts

SPRING [62] provides the strongest guarantees through a three-module architecture: an RNN learns implicit aesthetic preferences, **SampleSearch** (a SAT-inspired algorithm) ensures explicit spatial constraints receive non-zero probability only if satisfied, and Stable Diffusion renders compliant layouts. SampleSearch **zeros out** any layout violating explicit spatial rules, providing hard guarantees for specified constraints.

Factor Graph Diffusion Models [63] decompose the joint distribution of images and conditioning variables into modular factors. Each factor (semantic map, depth map, sketch) contributes independently, addressing the low recall problem in multi-object generation through structured probabilistic modeling.

12.4.2 Object Counting

Object counting remains challenging for neural generators:

CountGen [64] achieves 54% human-evaluated accuracy on object counting—a major improvement over SDXL's 26% but far from reliable. The method identifies self-attention features carrying object identity, uses a ReLayout U-Net to predict missing object locations, and applies dual-optimization during denoising.

Make It Count [65] and **YOLO-Count** [66] introduce differentiable counting objectives enabling gradient-based guidance during generation.

The gap between 54% and 100% reveals a fundamental limitation: these methods provide **soft guidance** through loss functions, not hard constraints through logical satisfaction.

12.4.3 Program Synthesis for Images

PS-GM [67] synthesizes structural programs (nested for-loops) from training images, then learns their distribution via variational encoder-decoder. Programs execute to produce structure renderings that neural networks complete into final images. This hybrid approach maintains interpretable generative procedures.

VisReP [68] trains LLMs to synthesize visual programs using reinforced self-training with REINFORCE. Self-trained 7B models outperform 10× larger frozen LLMs on detection and VQA tasks.

12.5 Programmatic Image Synthesis

The programmatic paradigm—LLM generates code, code executes to render image—offers fundamentally different tradeoffs: **exact specifications** through code parameters, **full editability** by modifying code, **deterministic outputs** from identical code, and **resolution independence** for vector formats.

12.5.1 Scientific Figures and TikZ

DeTikZify [69] synthesizes TikZ graphics programs from images or text using multimodal LLMs (CodeLLaMA + SigLIP vision encoder). Trained on DaTikZv2 (360K+ TikZ drawings), it outperforms Claude 3 and GPT-4V on compilation success rate and human preference. The key innovation is **MCTS-based inference** for iterative refinement without additional training.

AutomaTikZ/CLiMA [70] fine-tunes LLaMA on 120K TikZ drawings augmented with CLIP embeddings. Fine-tuned 7B/13B models outperform GPT-4 and Claude 2 on text-to-TikZ synthesis.

MagicGeo [71] generates geometric diagrams through an analytical geometry solver that computes precise point locations before TikZ generation.

12.5.2 SVG and Vector Graphics

VectorFusion [72] requires no captioned SVG dataset by using **Score Distillation Sampling (SDS)** loss from pretrained diffusion models. The pipeline samples rasters from Stable Diffusion, auto-traces with LIVE, and optimizes with differentiable vector rasterizer DiffVG.

Chat2SVG [73] combines LLM template generation with diffusion-based optimization. The LLM generates SVG structures with basic primitives, SDEdit adds visual details, and dual-stage optimization refines latent space then coordinate positions. Each element has unique IDs enabling semantic annotation and natural language editing.

OmniSVG [74] builds on Qwen-VL with specialized SVG tokenizers and the MMSVG-2M dataset (2 million annotated SVG assets), enabling text-to-SVG, image-to-SVG, and character-reference SVG generation at scale.

12.5.3 Chart and Data Visualization

MatPlotAgent [75] implements agentic scientific visualization: a query expansion module converts user requests to detailed instructions, a code agent generates Matplotlib with iterative debugging, and a visual agent (GPT-4V) provides feedback.

ChartLlama and **SynChart** [76] scale chart generation with 4 million chart images and 75 million dense annotations across Matplotlib, Seaborn, Plotly, and Bokeh.

12.5.4 Tradeoffs versus Neural Generation

Table 10: Programmatic versus neural image generation

Criterion	Programmatic	Neural
Precision	Exact via code parameters	Statistical approximation
Editability	High—modify code	Low—requires regeneration
Resolution	Independent (vector)	Fixed pixels
Photorealism	Limited by DSL	High for natural images
Complexity ceiling	DSL vocabulary bounds	Arbitrary visual content

Programmatic approaches excel for **technical content** (figures, diagrams, charts, icons) where precision and editability matter. Neural generation remains superior for **photorealistic scenes** requiring continuous texture and lighting variation.

12.6 Compositional Evaluation Benchmarks

12.6.1 Limitations of Traditional Metrics

FID measures distributional similarity between generated and real images but fails to capture text-image alignment. A 2024 analysis found FID has poor sample complexity and contradicts human ratings on compositional tasks. **CLIPScore** similarly fails on compositional prompts—it cannot distinguish "horse eating grass" from "grass eating horse."

12.6.2 T2I-CompBench

T2I-CompBench [77] provides 6,000 compositional prompts across three categories: **attribute binding** (color, shape, texture), **object relationships** (spatial, non-spatial), and **complex compositions**. T2I-CompBench++ extends this to 8,000 prompts with 3D-spatial relationships and numeracy.

Proposed metrics include:

- **Disentangled BLIP-VQA**: Decomposes prompts into independent VQA questions per attribute-object pair
- **UniDet-based metric**: Uses unified object detection to verify spatial positions and object counts
- **MLLM-based scoring**: GPT-4V with chain-of-thought reasoning

Key finding: **spatial relationships are the hardest subcategory**, with best models achieving only 15-35% accuracy.

12.6.3 GENEVAL

GENEVAL [78] uses OWL-ViT to automatically verify object presence, count, positioning, and color classification. Best models achieve only 15% on relative positioning and 35% on attribute binding—demonstrating substantial headroom.

12.7 The Flexibility-Control Tradeoff

Structured representations fundamentally trade **expressiveness for precision**:

- Scene graphs require predefined object and relationship vocabularies
- Layout conditioning constrains spatial arrangement but not fine-grained appearance
- Programmatic synthesis achieves exact specification within DSL bounds but cannot match neural generation for photorealistic variation

The practical implication is **complementary deployment**:

- Scene graphs for compositional scenes with known objects and relationships
- Layouts for spatial control of open-vocabulary content
- Programmatic methods for technical figures requiring precision and editability
- Free-form text for unconstrained creative generation

Current research trends toward **hybrid architectures** combining multiple conditioning modalities. RealCompo's dynamic balancer exemplifies this—fusing text-to-image and layout-to-image predictions during inference to achieve both realism and compositionality.

12.8 Summary: Structure Enables Control

Part V has examined how explicit structure improves image generation:

1. **Scene graphs** decompose prompts into objects and relationships, enabling DisCo's Compositional Masked Attention to achieve state-of-the-art compositional accuracy
2. **Layout conditioning** through GLIGEN's gated attention and training-free methods (Attend-and-Excite, BoxDiff) provides spatial control without retraining
3. **Neuro-symbolic methods** attempt hard guarantees through constraint satisfaction, though counting accuracy plateaus around 54%
4. **Programmatic synthesis** achieves exact, editable specifications for technical content through code generation (DeTikZify, VectorFusion, MatPlotAgent)
5. **Evaluation** requires compositional benchmarks (T2I-CompBench, GENEVAL) beyond FID and CLIPScore

The fundamental insight is that **structure is not opposed to neural generation but complementary**. The most capable systems will likely combine free-form neural generation with structured conditioning, automatically selecting appropriate structure levels based on prompt complexity and user intent.

13. Part VI: Training & Scaling

Architectural innovations mean little without the training infrastructure to realize them. This section examines the practical considerations—datasets, scaling laws, compute efficiency, and training stability—that determine whether image generation models achieve their potential.

13.1 Training Data: The Foundation

13.1.1 Major Datasets

The scale and quality of training data fundamentally bounds model capability:

LAION-5B [79]: 5.85 billion image-text pairs scraped from Common Crawl. Enabled open-source replication of Stable Diffusion. However, contains noise, duplicates, and problematic content. The 2023 CSAM discovery led to dataset withdrawal and highlighted the need for rigorous filtering.

LAION-Aesthetics: Subset of LAION filtered by aesthetic score (CLIP-based predictor trained on human ratings). The "6+" and "6.5+" subsets (approximately 600M and 120M images) dramatically improve visual quality at the cost of diversity.

DataComp [80]: Systematic study of dataset curation. Key finding: filtering by CLIP score and image-text alignment outperforms raw scale. A well-curated 1B dataset can match or exceed poorly curated 10B datasets.

Internal datasets: Google (Imagen), OpenAI (DALL-E), and others train on proprietary datasets of unknown composition. Estimated scale: hundreds of millions to billions of high-quality, licensed or generated images.

13.1.2 Data Filtering Pipeline

Modern training pipelines apply multiple filtering stages:

1. **Deduplication**: Remove near-duplicate images via perceptual hashing or embedding similarity. Reduces memorization and improves diversity.
2. **CLIP score filtering**: Retain only image-text pairs with high CLIP similarity (typically > 0.28). Removes misaligned captions.
3. **Aesthetic scoring**: Predict human aesthetic ratings; filter to top percentiles. Improves visual quality but may reduce diversity.
4. **NSFW filtering**: Remove explicit content via classifiers. Required for safe deployment but imperfect.
5. **Text quality filtering**: Remove captions that are too short, repetitive, or contain boilerplate ("click here", "stock photo").
6. **Resolution filtering**: Retain only images above minimum resolution (typically 256×256 or 512×512).

13.1.3 Synthetic Recaptioning

DALL-E 3's key insight [44]: **caption quality matters more than caption authenticity**. Training a captioning model to generate detailed descriptions, then retraining the generator on (image, synthetic caption) pairs dramatically improves prompt adherence.

The recaptioning pipeline:

1. Train a high-quality image captioner on human-annotated data
2. Generate detailed captions for all training images
3. Optionally blend original and synthetic captions
4. Train the generator on enriched captions

This addresses the fundamental mismatch: web-scraped alt-text is often brief, generic, or incorrect; user prompts are detailed and specific. Synthetic captions bridge this distribution gap.

CogView3 [81] and other recent systems report 2-3× improvements in prompt adherence from recaptioning alone, without architectural changes.

13.1.4 Data Augmentation

Standard augmentations for image generation training:

- **Random crop**: Especially important for aspect ratio variation
- **Horizontal flip**: With probability 0.5 (disabled for text-heavy images)
- **Multi-resolution training**: Train on 256→512→1024 progressively
- **Aspect ratio bucketing**: Group images by aspect ratio to minimize padding

NovelAI's bucketing demonstrated that aspect-ratio-aware batching significantly improves generation at non-square resolutions without architectural changes.

13.2 Scaling Laws

13.2.1 Compute-Optimal Training

Following Hoffmann et al.'s Chinchilla analysis for language models, image generation exhibits similar scaling relationships:

$$L(N, D) = \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}} + E \tag{48}$$

where N is parameters, D is data (images seen), and L is loss. Empirically, $\alpha \approx 0.5$ and $\beta \approx 0.5$ for diffusion models, suggesting balanced scaling of model and data.

Key finding: Unlike language models where data scaling is often bottlenecked, image generation benefits from both more parameters and more training iterations. The "compute-optimal" frontier favors larger models trained longer than typical practice.

13.2.2 DiT Scaling Experiments

Peebles and Xie [17] systematically studied DiT scaling:

Table 11: DiT scaling results on ImageNet 256×256

Model	Parameters	GFLOPs	FID-50K
DiT-S/2	33M	6	68.4
DiT-B/2	130M	23	43.5
DiT-L/2	458M	80	23.3
DiT-XL/2	675M	119	9.62
DiT-XL/2 (long)	675M	119	2.27

The results show clear power-law scaling: FID improves predictably with compute. The "long" variant demonstrates that extended training (7M steps vs 400K) yields substantial gains even at fixed parameter count.

13.2.3 Text Encoder Scaling

Imagen's surprising finding: **scaling the text encoder improves image quality more than scaling the diffusion model**. Specifically:

- T5-Small (60M) → T5-XXL (11B): Large FID improvements
- U-Net scaling over same compute range: Smaller gains

This suggests the generation bottleneck is often *understanding the prompt*, not *synthesizing pixels*. It motivated the shift toward larger language model backbones (T5-XXL in SD3, native LLM integration in GPT-Image).

13.2.4 Resolution Scaling

Higher resolution requires disproportionately more compute:

- Pixel count scales as $O(r^2)$ for resolution r
- Self-attention scales as $O(n^2)$ for n tokens, yielding $O(r^4)$ for full attention
- Latent diffusion reduces this to $O((r/f)^4)$ for compression factor f

Practical implication: Training at 1024×1024 costs 16× more than 256×256 in pixel space, or 16× more even in latent space (since latent resolution also quadruples). This motivates cascaded approaches (base + super-resolution) and efficient attention mechanisms.

13.3 Compute Requirements

13.3.1 Training Costs

Approximate training costs for major systems (estimated):

Table 12: Estimated training compute for image generation models

Model	Parameters	GPU-Hours	Hardware
Stable Diffusion 1.x	860M	150K	A100-40GB
Stable Diffusion 2.x	860M	200K	A100-80GB
SDXL	2.6B	500K+	A100-80GB
FLUX.1	12B	Unknown	H100
Imagen	2B (cascade)	Unknown	TPUv4

At current cloud prices ($2-3/GPU-hour), training a competitive open model costs $300K-$1M+. Frontier models from large labs likely cost $10M+ including failed experiments.

13.3.2 Memory Requirements

Training memory scales with:

- **Model parameters**: $4N$ bytes for fp32, $2N$ for bf16/fp16
- **Optimizer state**: $8N$ bytes for Adam (momentum + variance in fp32)
- **Gradients**: $2N$-$4N$ bytes depending on precision
- **Activations**: Dominant for large batch sizes; reduced via checkpointing

A 2B parameter model requires approximately:

- Parameters: 4 GB (bf16)
- Optimizer: 16 GB (Adam fp32)
- Gradients: 4 GB (bf16)
- Activations: 20-40 GB (batch-dependent)
- **Total**: 44-64 GB per GPU (before parallelism)

This motivates distributed training strategies.

13.3.3 Inference Costs

Generation costs depend heavily on sampling steps:

Table 13: Inference compute per image (approximate)

Model	Steps	Time (A100)	TFLOPs
SD 1.5 (512×512)	50	2.5s	25
SDXL (1024×1024)	30	8s	200
FLUX.1-dev	20	12s	400
FLUX.1-schnell	4	2.5s	80

Distilled models (FLUX-schnell, SDXL-Turbo) achieve 4-10× speedups by reducing required steps from 20-50 to 1-4.

13.4 Training Stability

13.4.1 Noise Schedules

The noise schedule $\bar{\alpha}_t$ determines the signal-to-noise ratio at each timestep:

Linear schedule (DDPM): β_t increases linearly from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. Simple but suboptimal—too much noise added early, too little late.

Cosine schedule (Improved DDPM) [82]:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)^2 \tag{49}$$

Provides more uniform SNR distribution across timesteps. Standard for most modern systems.

Shifted schedules (SD3): Shift the schedule toward higher noise levels for high-resolution training. Prevents the model from "coasting" on easy low-noise steps.

Zero terminal SNR: Ensure $\bar{\alpha}_T = 0$ exactly, so the model must generate from pure noise. Important for consistency between training and inference.

13.4.2 Loss Weighting

The standard diffusion loss weights all timesteps equally:

$$\mathcal{L} = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right] \tag{50}$$

However, different timesteps contribute differently to generation quality:

- **High noise** (large t): Determines global structure
- **Low noise** (small t): Determines fine details

Min-SNR weighting [83]: Weight loss by $\min(\text{SNR}(t), \gamma)$ for threshold γ. Prevents high-SNR (low noise) timesteps from dominating training.

v-prediction: Predict $v = \sqrt{\bar{\alpha}_t}\epsilon - \sqrt{1-\bar{\alpha}_t}x_0$ instead of ϵ. Provides more stable gradients, especially at high noise levels.

13.4.3 Exponential Moving Average (EMA)

EMA maintains a smoothed copy of model weights:

$$\theta_{\text{EMA}} \leftarrow \beta\theta_{\text{EMA}} + (1-\beta)\theta \tag{51}$$

Typical $\beta = 0.9999$ (update every step) or 0.999 (faster tracking). The EMA model is used for inference while the base model continues training.

Why EMA helps:

- Reduces variance from mini-batch gradients
- Smooths over learning rate fluctuations
- Acts as implicit ensembling across training trajectory

EMA is essentially universal in image generation training.

13.4.4 Learning Rate and Batch Size

Linear scaling rule: When increasing batch size by factor k, scale learning rate by k (or $\sqrt{k}$ for more conservative scaling).

Warmup: Gradually increase learning rate over first 1-10K steps. Prevents early training instability from large initial gradients.

Typical values:

- Learning rate: 10^{-4} to 10^{-5} (AdamW)
- Batch size: 256 to 2048 (effective, after gradient accumulation)
- Warmup: 1K-10K steps
- Weight decay: 0.01 to 0.1

13.5 Distributed Training

Training billion-parameter models requires distributing computation across many GPUs.

13.5.1 Data Parallelism

The simplest strategy: replicate the model on each GPU, split batches across GPUs, synchronize gradients.

Distributed Data Parallel (DDP): Synchronous gradient averaging via all-reduce. Standard for models that fit in single-GPU memory.

Limitations: Each GPU must hold full model, optimizer state, and gradients. Fails for models >10B parameters on current hardware.

13.5.2 ZeRO and FSDP

ZeRO (Zero Redundancy Optimizer) [84] partitions optimizer state, gradients, and optionally parameters across GPUs:

- **ZeRO-1**: Partition optimizer state. Memory: $O(N) + O(N/P)$ for P GPUs.
- **ZeRO-2**: Partition optimizer + gradients. Memory: $O(N/P)$ optimizer, $O(N/P)$ gradients.
- **ZeRO-3**: Partition everything including parameters. Memory: $O(N/P)$ total.

FSDP (Fully Sharded Data Parallel): PyTorch's implementation of ZeRO-3. Shards parameters, gathers for forward/backward, reshards after.

Trade-off: ZeRO-3/FSDP reduces memory by $P\times$ but increases communication. Effective up to hundreds of GPUs with fast interconnects.

13.5.3 Tensor and Pipeline Parallelism

For very large models, split computation within each layer:

Tensor parallelism: Split weight matrices across GPUs. For a linear layer $Y = XW$, split W column-wise, compute partial results, all-reduce. Requires high-bandwidth interconnect (NVLink).

Pipeline parallelism: Split layers across GPUs. GPU 1 runs layers 1-10, GPU 2 runs layers 11-20, etc. Requires micro-batching to maintain utilization.

Typical configuration for 10B+ models:

- Tensor parallelism: 4-8 GPUs within a node (fast NVLink)
- Pipeline parallelism: 2-4 stages across nodes
- Data parallelism: Remaining GPUs via FSDP

13.5.4 Mixed Precision Training

fp16: Half precision reduces memory $2\times$, enables tensor cores. Requires loss scaling to prevent gradient underflow.

bf16: Brain float format with larger exponent range. No loss scaling needed. Standard for modern training.

fp8: Quarter precision emerging in H100 GPUs. Requires careful quantization but provides $2\times$ throughput over bf16.

Typical setup: Model weights and activations in bf16, optimizer state in fp32, gradient accumulation in fp32.

13.5.5 Gradient Checkpointing

Trade compute for memory by recomputing activations during backward pass instead of storing them:

- **Full checkpointing**: Recompute all activations. Memory: $O(\sqrt{L})$ for L layers.
- **Selective checkpointing**: Checkpoint only expensive layers (attention). Balances memory and compute.

Impact: Reduces activation memory by 5-10$\times$ at cost of $\sim$30% more compute. Essential for training large models with limited memory.

13.6 Progressive and Curriculum Training

13.6.1 Resolution Progression

Train at low resolution first, then fine-tune at higher resolutions:

1. Train at 256$\times$256 until convergence (fast, cheap)
2. Fine-tune at 512$\times$512 (initialize from 256 checkpoint)
3. Fine-tune at 1024$\times$1024 (initialize from 512 checkpoint)

Benefits:

- Faster initial training (16$\times$ cheaper at 256 vs 1024)
- Curriculum effect: model learns coarse structure before fine details
- Enables early evaluation and iteration

SDXL uses this approach, with base model at 1024 and optional refiner for additional detail.

13.6.2 Caption Complexity Curriculum

Qwen-Image's training progression:

1. Non-text images (simple objects, scenes)
2. Simple text rendering (short words, common fonts)
3. Complex text (sentences, multiple fonts)
4. Paragraph-level text (documents, signs with paragraphs)

This curriculum prevents the model from being overwhelmed by hard examples early in training.

13.6.3 Noise Schedule Curriculum

Start with easier noise levels (lower maximum noise), gradually increase:

- Early training: $t_{\max} = 0.5T$
- Mid training: $t_{\max} = 0.8T$
- Late training: $t_{\max} = T$ (full schedule)

This stabilizes early training when the model is weakest at high-noise prediction.

13.7 Distillation for Efficient Inference

Training large models for quality, then distilling for speed, is increasingly standard.

13.7.1 Progressive Distillation

Salimans and Ho [33] reduce sampling steps by training a student to match two teacher steps in one:

1. Teacher: 1024 steps (DDPM baseline)
2. Student 1: 512 steps (learns to match 2 teacher steps)
3. Student 2: 256 steps (learns to match 2 student-1 steps)
4. Continue halving until 4-8 steps

Result: 64× speedup with minimal quality loss.

13.7.2 Adversarial Distillation

ADD (Adversarial Diffusion Distillation) [32] trains a discriminator to distinguish student outputs from real images and teacher outputs. The student learns to fool the discriminator in 1-4 steps.

SDXL-Turbo achieves single-step generation via ADD, producing 512×512 images in under 100ms on consumer GPUs.

13.7.3 Guidance Distillation

CFG requires two forward passes (conditional and unconditional). Guidance distillation trains a student to produce CFG-equivalent outputs in one pass:

$$\mathcal{L} = \|\epsilon_{\text{student}}(x_t, c) - \tilde{\epsilon}_{\text{teacher}}(x_t, c, w)\|^2 \tag{52}$$

where $\tilde{\epsilon}_{\text{teacher}}$ is the CFG-combined prediction. This halves inference cost with no quality loss.

13.8 Summary: The Training Stack

Training competitive image generation models requires:

1. **Data**: Billions of filtered, recaptioned image-text pairs. Caption quality matters as much as quantity.
2. **Scaling**: Follow compute-optimal frontier. Text encoder scaling is underrated; resolution scaling is expensive.
3. **Stability**: Cosine noise schedule, v-prediction or Min-SNR weighting, EMA, careful warmup.
4. **Infrastructure**: FSDP/ZeRO for memory efficiency, mixed precision (bf16), gradient checkpointing for large batches.
5. **Curriculum**: Progressive resolution, caption complexity, noise schedule progression.
6. **Distillation**: Progressive or adversarial distillation for efficient inference.

The training recipe matters as much as the architecture. Many architectural innovations fail without corresponding training improvements; many training improvements succeed across architectures.

14. Part VII: Evaluation & Benchmarks

Evaluating image generation is fundamentally harder than evaluating discriminative models. There is no single ground truth; quality is subjective and multi-dimensional. This section examines the metrics, benchmarks, and protocols that enable meaningful comparison.

14.1 Classic Metrics and Their Limitations

14.1.1 Fréchet Inception Distance (FID)

FID [85] remains the most widely reported metric. It measures the distance between real and generated image distributions in Inception-v3 feature space:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\right) \tag{53}$$

where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean and covariance of real and generated features.

Strengths:

- Captures both quality (mean distance) and diversity (covariance)
- Correlates reasonably with human judgment on unconditional generation
- Standard benchmark enables cross-paper comparison

Limitations:

- **Sample complexity**: Requires 10K-50K samples for stable estimates. Small sample FID is highly variable.
- **Reference set dependence**: FID depends on which real images are used. Different splits yield different scores.
- **Inception bias**: Features from ImageNet-trained Inception may not capture relevant dimensions for other domains.
- **Ignores text alignment**: FID measures distributional similarity, not whether images match their prompts.
- **Mode coverage blindness**: A model generating only one perfect image per class scores well despite zero diversity.

Practical issues: FID computed with different libraries (TensorFlow vs PyTorch), image preprocessing, or sample sizes cannot be compared directly. The field lacks standardized FID computation.

14.1.2 Inception Score (IS)

Inception Score [86] measures both quality and diversity:

$$\text{IS} = \exp\left(\mathbb{E}_x\left[D_{KL}(p(y|x)\|p(y))\right]\right) \tag{54}$$

where $p(y|x)$ is the Inception classifier's prediction for generated image x, and $p(y)$ is the marginal across all generated images.

Interpretation: High IS means the classifier is confident about each image (quality) but the overall distribution is diverse (variety).

Limitations:

- Only measures ImageNet class diversity—useless for faces, scenes, or non-object domains
- Ignores intra-class diversity (many similar dogs score as well as varied dogs)
- Easily gamed by generating one perfect example per class
- Does not consider text conditioning at all

IS is largely deprecated for text-to-image evaluation but remains reported for ImageNet class-conditional generation.

14.1.3 CLIP Score

CLIP Score measures text-image alignment using CLIP embeddings:

$$\text{CLIP Score} = \mathbb{E}\left[\cos(\text{CLIP}_{\text{image}}(x), \text{CLIP}_{\text{text}}(c))\right] \tag{55}$$

Strengths:

- Directly measures prompt adherence
- Works across arbitrary text prompts
- Cheap to compute (single forward pass)

Limitations:

- **Bag-of-concepts**: CLIP cannot distinguish "dog bites man" from "man bites dog"
- **Coarse semantics**: High-level concept matching, misses fine details
- **Conflates quality and alignment**: A blurry correct image may score higher than a sharp wrong one
- **Score clustering**: Most images score 0.25-0.35, providing little discriminative power

CLIP Score is necessary but insufficient for evaluating text-to-image generation.

14.1.4 Precision and Recall

Kynkäänniemi et al. [87] define precision and recall for generative models:

- **Precision**: Fraction of generated images that fall within the real data manifold (quality)
- **Recall**: Fraction of real data manifold covered by generated images (diversity)

Computed via k-nearest neighbor density estimation in feature space.

Value: Disentangles quality and diversity, unlike FID which conflates them. A model with perfect precision but low recall generates only a subset of possible images; one with high recall but low precision generates diverse but low-quality outputs.

14.2 Human Evaluation

Automated metrics are proxies for human judgment. For high-stakes comparisons, human evaluation remains essential.

14.2.1 Preference Testing

The most common protocol: show two images (from different models) for the same prompt, ask which is better.

Variants:

- **Overall preference**: "Which image is better overall?"
- **Aspect-specific**: "Which better matches the prompt?" / "Which has higher quality?"
- **Forced choice**: Must pick one (no ties)
- **Best-of-N**: Rank or select best from multiple options

Analysis: Report win rate, and use Bradley-Terry or Elo models for transitive ranking when comparing multiple systems.

14.2.2 Likert Scales

Rate individual images on a scale (e.g., 1-5):

- **Quality**: "Rate the visual quality of this image"
- **Alignment**: "Rate how well this image matches the prompt"
- **Realism**: "Rate how realistic this image appears"

Advantages: Provides absolute scores, not just relative comparisons. Enables within-model analysis.
Disadvantages: Calibration varies across annotators. Requires more annotations for statistical power.

14.2.3 Inter-Annotator Agreement

Human evaluation is noisy. Measure agreement to assess reliability:

Cohen's Kappa: Agreement between two annotators, corrected for chance:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{56}$$

where p_o is observed agreement and p_e is expected chance agreement.

Krippendorff's Alpha: Generalizes to multiple annotators and various scales.

Typical values: $\kappa > 0.6$ indicates substantial agreement; $\kappa > 0.8$ indicates near-perfect agreement. Image quality judgment typically achieves $\kappa \approx 0.5$-0.7.

14.2.4 Crowdsourcing Best Practices

- **Qualification tests**: Filter annotators via known-answer questions
- **Attention checks**: Include obvious cases to detect random clicking
- **Redundancy**: 3-5 annotators per item, aggregate via majority vote
- **Balanced presentation**: Randomize left/right position to avoid bias
- **Clear instructions**: Provide examples of good/bad ratings

14.3 Compositional Benchmarks

Standard metrics fail on compositional prompts. Specialized benchmarks address this gap.

14.3.1 DrawBench

Introduced with Imagen [39], DrawBench contains 200 prompts across 11 categories:

- Colors, counting, spatial relationships
- Text rendering, rare words, misspellings
- Conflicting descriptions, unusual interactions

Evaluation is human-only: annotators compare model outputs pairwise. DrawBench revealed that models struggle with counting, negation, and spatial relationships.

14.3.2 T2I-CompBench

As discussed in Part V, T2I-CompBench [77] provides 6,000 prompts with automated evaluation:

Table 14: T2I-CompBench categories and metrics

Category	Example	Metric
Color binding	"red apple, yellow banana"	BLIP-VQA per object
Shape binding	"triangular sign, circular clock"	BLIP-VQA per object
Texture binding	"wooden table, metal chair"	BLIP-VQA per object
Spatial relationships	"cat left of dog"	UniDet position verification
Non-spatial relations	"dog chasing cat"	CLIP-based scoring
Complex	Multiple constraints	Combined metrics

Key findings: Spatial relationships are hardest (15-35% accuracy); attribute binding remains challenging; models have improved but significant gaps persist.

14.3.3 GENEVAL

GENEVAL [78] uses object detection (OWL-ViT) to verify:

- **Object presence**: Are all mentioned objects generated?
- **Counting**: Is the count correct?
- **Position**: Are objects in specified locations?
- **Color**: Do detected objects have correct colors?

Detection-based evaluation is more robust than VQA-based approaches but limited by detector accuracy.

14.3.4 TIFA

Text-to-Image Faithfulness with Question Answering (TIFA) [88] generates yes/no questions from prompts, then answers them about generated images:

1. Parse prompt: "A red car on a beach" $\rightarrow$ questions
2. Generate questions: "Is there a car?", "Is the car red?", "Is there a beach?"
3. Answer questions about generated image via VQA
4. Score = fraction of correct answers

TIFA provides interpretable, fine-grained evaluation but depends on question generation and VQA quality.

14.4 Learned Preference Metrics

Human preferences can be learned and applied at scale.

14.4.1 ImageReward

ImageReward [89] trains a reward model on human preference data:

- Collect 137K expert comparisons on image-text pairs
- Train BLIP-based model to predict preference
- Score correlates with human judgment better than CLIP

Can be used for evaluation or as a training signal (RLHF for image generation).

14.4.2 Human Preference Score v2 (HPSv2)

HPSv2 [90] improves on ImageReward with larger training data:

- 798K preference annotations
- CLIP backbone with preference head
- Separate models for different aspects (aesthetics, alignment)

HPSv2 is widely used for model comparison and as a reward signal for preference optimization.

14.4.3 PickScore

PickScore [91] focuses on the Pick-a-Pic dataset:

- 500K+ user preferences from web application

- Real user choices, not crowdworker annotations
- CLIP-H backbone fine-tuned on preferences

14.4.4 Aesthetic Score

LAION's aesthetic predictor estimates visual appeal:

- Trained on 5M images rated by humans for aesthetics
- CLIP embedding + MLP regressor
- Used for filtering training data (LAION-Aesthetics)

Aesthetic score captures visual appeal but not text alignment or correctness.

14.4.5 Limitations of Learned Metrics

- **Training data bias**: Preferences reflect annotator demographics and instructions
- **Overfitting risk**: Models may overfit to artifacts of the training procedure
- **Goodhart's Law**: Optimizing for the metric may diverge from true quality
- **Lack of interpretability**: Scores don't explain *why* one image is preferred

14.5 Safety and Bias Evaluation

Responsible deployment requires evaluating potential harms.

14.5.1 NSFW Detection

Evaluate generation of explicit content:

- Generate from benign prompts, measure false positive rate
- Generate from adversarial prompts, measure bypass rate
- Use classifier ensemble (NudeNet, CLIP-based detectors)

No model is perfectly safe; the goal is minimizing harmful outputs while preserving legitimate uses.

14.5.2 Bias Audits

Measure demographic biases in generated images:

Occupation bias: Generate "a photo of a CEO" / "a nurse" / "a criminal" and measure demographic distribution of generated faces.

Attribute correlation: Measure whether certain attributes (gender, race) correlate with quality, realism, or prompt adherence.

Representation gaps: Compare diversity of generated images to real-world demographics or training data.

14.5.3 Memorization Detection

Test whether models reproduce training data:

- Generate with training captions, compare to training images
- Use perceptual similarity (SSIM, LPIPS) to detect near-copies
- Search generated images against training set via embeddings

Memorization rates vary: rare images with distinctive captions are most vulnerable. Deduplication during training reduces but doesn't eliminate memorization.

14.5.4 Adversarial Robustness

Test resistance to prompt injection and jailbreaking:

- Prompt perturbations that bypass safety filters
- Unicode/encoding tricks
- Indirect prompting via image conditioning

14.6 Benchmark Suites

14.6.1 Holistic Evaluation

No single metric suffices. A comprehensive evaluation includes:

Table 15: Recommended evaluation suite for text-to-image models

Aspect	Metric	Notes
Quality	FID (50K samples)	Compare to same reference set
Diversity	Precision/Recall	Disentangles quality vs coverage
Text alignment	CLIP Score	Coarse semantic match
Compositionality	T2I-CompBench	Attribute binding, spatial
Object fidelity	GENEVAL	Detection-based verification
Human preference	HPSv2 or ImageReward	Learned from annotations
Human evaluation	Pairwise comparison	Gold standard, expensive
Safety	NSFW rate, bias audit	Required for deployment

14.6.2 Leaderboards and Arenas

Open leaderboards (Hugging Face, Papers with Code) track FID and other metrics but suffer from inconsistent evaluation protocols.

AI Arena (blind evaluation): Users compare anonymous model outputs; Elo ratings computed from preferences. Provides robust relative ranking but no absolute quality measure.

Limitations: Leaderboards encourage metric optimization over genuine improvement; gaming is common.

14.7 Evaluation Best Practices

1. **Report multiple metrics**: No single metric captures all dimensions of quality.
2. **Use consistent protocols**: Specify sample size, reference set, preprocessing, and random seeds.
3. **Include human evaluation**: For significant claims, automated metrics are insufficient.
4. **Evaluate on held-out prompts**: Models may overfit to common benchmarks.
5. **Report confidence intervals**: FID and other metrics have high variance; point estimates are misleading.
6. **Compare fairly**: Same compute budget, same training data scale, same inference steps.
7. **Acknowledge limitations**: Every metric has failure cases; be transparent about what isn't measured.

14.8 Summary: Evaluation Remains Unsolved

The evaluation landscape has improved but fundamental challenges persist:

- **FID** remains standard despite known limitations; no consensus replacement exists
- **Compositional benchmarks** (T2I-CompBench, GENEVAL) reveal failures that FID/CLIP miss
- **Learned preference metrics** (HPSv2, ImageReward) provide scalable human-like judgment but may overfit
- **Human evaluation** is the gold standard but expensive and slow
- **Safety evaluation** is necessary but methods are immature

The ideal evaluation framework would be cheap, interpretable, comprehensive, and aligned with human values. Current methods achieve at most two of these properties.

15. Part VIII: Weaknesses & Open Problems

Despite remarkable progress, image generation models exhibit systematic failures that reveal fundamental limitations. Understanding these weaknesses is essential for both users (knowing when to trust outputs) and researchers (identifying high-value problems).

15.1 Compositional Reasoning Failures

15.1.1 The Binding Problem

Models struggle to bind attributes to the correct objects. The prompt "a red cube and a blue sphere" frequently yields a blue cube or red sphere. This *attribute leakage* stems from how text embeddings conflate multiple object-attribute pairs into a single vector.

Root cause: Cross-attention distributes information globally. When generating the cube region, the model attends to both "red cube" and "blue sphere" tokens, mixing their attributes.

Partial solutions: Compositional Masked Attention (DisCo), layout conditioning (GLIGEN), attention manipulation (Attend-and-Excite). These improve but don't eliminate the problem.

Fundamental tension: The same global attention that enables coherent scene composition causes attribute mixing. Local attention solves binding but loses global coherence.

15.1.2 Counting

"Three cats" reliably produces two or four cats. As documented in Part V, even specialized methods (CountGen) achieve only 54% accuracy. The failure is worse for:

- Larger numbers (>5)
- Zero ("a room with no windows")
- Approximate quantities ("several", "a few")

Root cause: Diffusion models don't have explicit object representations. They generate images holistically, without discrete counting mechanisms.

Why this is hard: Counting requires segmenting the scene into objects, then enumerating—operations diffusion models perform implicitly at best.

15.1.3 Spatial Relationships

"A cat to the left of a dog" produces spatially random arrangements. T2I-CompBench shows 15-35% accuracy on spatial tasks. Models understand individual objects but not their relative positions.

Root cause: Text encoders represent "left of" and "right of" similarly (both indicate spatial relationship). The diffusion model receives weak spatial signal.

Layout conditioning helps: GLIGEN with explicit bounding boxes achieves much higher spatial accuracy. But this requires users to specify layouts manually.

15.1.4 Negation

"A beach with no people" produces people on beaches. Models treat negation as weak negative signal rather than hard constraint.

Root cause: Training data rarely contains (image, negation caption) pairs. The model never learns that "no X" means "absence of X."

Negative prompts partially work: Classifier-free guidance with negative prompts reduces unwanted content but doesn't guarantee absence.

15.2 Visual Quality Failures

15.2.1 Human Anatomy

Hands remain notoriously difficult:

- Wrong number of fingers (four or six)
- Impossible joint configurations
- Merged or bifurcated fingers
- Inconsistent hand size

Faces have improved dramatically but still fail on:

- Unusual angles or expressions
- Multiple people interacting
- Consistency across generations

Root cause: Hands are small in most training images, providing weak supervision. Their articulated structure with many degrees of freedom is hard to learn from limited data.

Recent progress: GPT-Image-1.5 and Nano Banana Pro show improved hands, likely from curated training data and larger scale.

15.2.2 Text Rendering

Generating readable text in images remains challenging:

- Misspellings and character substitutions
- Inconsistent font within a word

- Gibberish that resembles text
- Failure on non-Latin scripts (except Qwen-Image for Chinese)

Root cause: Text is discrete and exact; generation is continuous and approximate. The model must learn character-level correspondence from image-level supervision.

Progress: DALL-E 3, Ideogram, and recent models show substantial improvement, likely from text-focused training data and recaptioning that emphasizes text content.

15.2.3 Fine Details and Textures

At high resolution, models often produce:

- Repetitive textures (tiling artifacts)
- Inconsistent fine details (different leaf patterns on same tree)
- "Plastic" or "AI look" on skin and surfaces

Root cause: Latent diffusion compresses images $8\times$, losing fine detail. Super-resolution stages recover some detail but can't invent consistent novel detail.

15.3 Semantic and Factual Failures

15.3.1 Hallucination

Models generate plausible-looking but incorrect content:

- Nonexistent landmarks or buildings
- Incorrect animal anatomy (horses with six legs)
- Physically impossible scenes (floating objects, impossible shadows)

Root cause: Models learn correlations, not causation or physics. A horse "usually" has four legs but the model has no mechanism to enforce this.

15.3.2 Factual Errors

When generating known entities:

- Wrong details on famous buildings
- Incorrect flags, logos, or symbols
- Anachronistic combinations (modern cars in historical scenes)

Nano Banana Pro's approach: Search grounding retrieves factual information before generation. This helps for infographics but doesn't solve general factual consistency.

15.3.3 Physical Plausibility

Generated scenes often violate physics:

- Objects floating without support
- Impossible reflections or shadows
- Inconsistent lighting across scene
- Materials behaving incorrectly (water, glass, metal)

Root cause: Models learn appearance, not physics. They've seen many shadows but don't understand that shadows require light sources and occluders.

15.4 Control and Consistency Failures

15.4.1 Fine-Grained Control

Users cannot precisely specify:

- Exact poses or expressions
- Specific camera angles
- Precise object sizes or positions
- Detailed material properties

Current solutions: ControlNet (requires control images), layout conditioning (requires bounding boxes), extensive prompt engineering (unreliable).

Gap: No natural language interface achieves the control that professional tools (3D software, photo editing) provide.

15.4.2 Cross-Generation Consistency

Generating the same character or scene multiple times yields different results:

- Different facial features, clothing, body proportions
- Inconsistent style even with same prompt
- No persistent identity across generations

Partial solutions: Seed fixing (deterministic but inflexible), IP-Adapter (requires reference image), fine-tuning (expensive, overfits).

GPT-Image-1.5 progress: Demonstrated improved character consistency through native multimodal context, but mechanism unclear.

15.4.3 Style Control

Achieving specific artistic styles is unreliable:

- "In the style of [artist]" produces inconsistent results
- Mixing styles often fails
- Fine style distinctions (early vs late Picasso) not captured

15.5 Efficiency and Scalability

15.5.1 Inference Cost

Despite distillation advances, generation remains expensive:

- FLUX.1-dev: 12 seconds per image on A100
- Real-time generation requires significant hardware
- Mobile deployment limited to small models or cloud

Fundamental tradeoff: Quality correlates with compute. Few-step models sacrifice quality for speed.

15.5.2 Training Cost

State-of-the-art models require massive resources:

- $1M+ for competitive open models
- $10M+ for frontier closed models (estimated)
- Limits who can train new architectures

15.5.3 Memory Requirements

Large models challenge deployment:

- FLUX.1 12B: Requires 24GB+ VRAM
- Consumer GPUs (8-12GB) can't run full models
- Quantization helps but degrades quality

15.6 Safety and Ethics

15.6.1 Deepfakes and Misinformation

High-quality generation enables:

- Fake images of real people in compromising situations
- Fabricated evidence for misinformation
- Identity theft and fraud

Mitigations: Watermarking (SynthID), detection models, platform policies. None are fully effective.

15.6.2 Copyright and Training Data

Unresolved legal and ethical questions:

- Training on copyrighted images without consent
- Models reproducing training data (memorization)
- Generated images resembling copyrighted works

- Artist style replication without attribution

15.6.3 Bias Amplification

Models amplify training data biases:

- Demographic skew in generated faces
- Stereotyped associations (occupations, activities)
- Underrepresentation of minority groups

15.7 Open Research Problems

We highlight the most important unsolved problems:

15.7.1 Compositional Generation

Problem: Generate images with reliable object-attribute binding, correct counts, and accurate spatial relationships.

Why hard: Requires discrete reasoning (counting, binding) within continuous generation. Current architectures lack explicit object representations.

Promising directions: Neuro-symbolic integration, scene graph conditioning, layout-based generation.

15.7.2 World Models

Problem: Generate images consistent with physical laws and common sense.

Why hard: Physics is not in the training signal. Models see correlations (shadows appear near objects) not causation (shadows require light blockage).

Promising directions: Physics-informed losses, simulation-guided training, neuro-symbolic reasoning.

15.7.3 Controllable Generation

Problem: Enable precise user control without requiring technical expertise (bounding boxes, control images).

Why hard: Natural language is ambiguous; precise specification requires structured input.

Promising directions: LLM-based layout generation, conversational refinement, learned user intent models.

15.7.4 Efficient High-Quality Generation

Problem: Achieve frontier quality with consumer hardware and real-time latency.

Why hard: Quality scales with compute; efficiency gains often trade quality.

Promising directions: Better distillation, architecture innovations (Mamba, linear attention), specialized hardware.

15.7.5 Consistent Character Generation

Problem: Generate the same character across multiple images without fine-tuning or reference images.

Why hard: Models lack persistent representations; each generation starts fresh.

Promising directions: Learned identity embeddings, memory-augmented generation, native multimodal context.

15.7.6 Evaluation

Problem: Develop metrics that reliably predict human preference across all dimensions of quality.

Why hard: Human judgment is multidimensional, subjective, and context-dependent.

Promising directions: Learned preference models, decomposed evaluation, human-in-the-loop benchmarking.

15.8 Summary: The Gap Between Demo and Deployment

Image generation models produce stunning demos but exhibit systematic failures in deployment:

- **Compositional reasoning** remains brittle—counting, binding, spatial relationships fail frequently
- **Visual quality** has improved but hands, text, and fine details remain challenging
- **Semantic correctness** is not guaranteed—hallucination and factual errors are common
- **Control** requires technical expertise or accepts unreliable natural language
- **Efficiency** limits deployment to powerful hardware or cloud services
- **Safety** concerns are mitigated but not solved

The gap between cherry-picked results and reliable generation defines the current frontier. Closing this gap requires not just scaling but fundamental advances in compositional reasoning, world modeling, and controllable generation.

16. Part IX: Trends & Future Directions

The trajectory of image generation is shaped by converging technical trends, evolving applications, and shifting competitive dynamics. This section synthesizes the directions emerging from the developments surveyed in Parts I-VIII.

16.1 Architectural Convergence

16.1.1 Unified Multimodal Models

The clearest trend is convergence toward unified architectures that handle text, images, audio, and video within a single model:

- **2022-2023**: Separate models for each modality (CLIP + Stable Diffusion + Whisper)
- **2024**: Adapter-based integration (LLaVA + SD via projectors)
- **2025**: Native multimodal (GPT-4o, Gemini 3, unified generation)

Prediction: By 2027, the distinction between "language model" and "image model" will be obsolete. Frontier models will be natively multimodal, with generation emerging from the same representations used for understanding.

Implications: Specialized image models may persist for efficiency or specific applications, but general-purpose AI will be inherently multimodal.

16.1.2 Transformers Everywhere

The U-Net's dominance is ending:

- DiT demonstrated pure transformer diffusion
- FLUX and SD3 use transformer-based architectures
- Autoregressive image models are transformers by construction

Prediction: U-Nets will be relegated to specialized applications (super-resolution, inpainting). Main generation will be transformer-based, benefiting from LLM infrastructure and scaling insights.

16.1.3 Rectified Flow Ascendant

Rectified flow is displacing traditional diffusion:

- Straighter trajectories enable fewer steps
- Simpler training (no noise schedule design)
- Better distillation properties

Prediction: DDPM-style diffusion will be viewed as a historical stepping stone. Rectified flow (or successors) will be the default continuous generation paradigm.

16.2 Efficiency Revolution

16.2.1 Step Reduction

The trend from 1000 steps (DDPM) to 1-4 steps (SDXL-Turbo, FLUX-schnell) will continue:

- Progressive and adversarial distillation now standard
- Consistency models promise one-step generation
- Architecture-distillation co-design emerging

Prediction: Production models will commonly run in 1-4 steps. Research will focus on closing the quality gap between single-step and multi-step generation.

16.2.2 Mobile and Edge Deployment

Consumer deployment is becoming viable:

- On-device models (Apple, Qualcomm) generating 512×512 in seconds
- Quantization (INT8, INT4) reducing memory requirements
- Architecture innovations targeting mobile constraints

Prediction: By 2027, flagship phones will run competitive image generation locally. Cloud will remain necessary for frontier quality but not for basic generation.

16.2.3 Real-Time Generation

Interactive generation is emerging:

- Sketch-to-image in real-time
- Live video stylization
- Interactive 3D scene editing

Prediction: Real-time generation will enable new creative workflows—painting with AI as a live collaborator rather than a batch processor.

16.3 Beyond Images

16.3.1 Video Generation

Video is the natural extension:

- Sora (OpenAI), Veo (Google), and others demonstrate long-form video
- Temporal consistency remains challenging
- Compute costs are order of magnitude higher

Key challenges: Temporal coherence, motion realism, long-range consistency, efficient generation of many frames.

Prediction: 2025-2026 will see rapid video generation progress. By 2028, short video generation may be as reliable as image generation today.

16.3.2 3D Generation

Multiple approaches are competing:

- Multi-view diffusion (generate views, reconstruct 3D)
- Direct 3D generation (3D-aware architectures)
- Text-to-3D via SDS optimization

Key challenges: Consistency across views, geometric accuracy, texture quality, generation speed.

Prediction: 3D generation will mature more slowly than video. High-quality text-to-3D comparable to current text-to-image may require 3-5 more years.

16.3.3 Interactive Worlds

The frontier is generating explorable environments:

- Persistent, navigable 3D scenes
- Physics-aware object interactions
- Real-time response to user actions

Prediction: This represents the "world model" vision—AI that understands physical reality well enough to simulate it. Significant research challenge with unclear timeline.

16.4 Structured and Controllable Generation

16.4.1 Scene Graph and Layout Integration

Part V documented structured generation methods. The trend is toward seamless integration:

- LLMs generating layouts from natural language
- Hybrid systems combining text, layout, and direct control
- Automatic structure inference from complex prompts

Prediction: Users won't manually create scene graphs. LLMs will parse complex prompts into structured representations, giving users natural language control with structured reliability.

16.4.2 Neuro-Symbolic Advancement

Hard guarantees remain elusive but promising:

- Differentiable constraint satisfaction improving
- Verification models checking outputs against specifications
- Hybrid pipelines: neural generation + symbolic verification + refinement

Prediction: Within 5 years, systems will offer "verified generation" for specific constraints (counting, spatial relationships). General compositional guarantees may require fundamental breakthroughs.

16.4.3 Conversational Control

GPT-Image and Nano Banana demonstrate conversational refinement:

- "Make the sky more blue"
- "Remove the person on the left"
- "Same scene but at sunset"

Prediction: This interaction pattern will become standard. Generation will be iterative dialogue rather than one-shot prompting.

16.5 Training and Data Evolution

16.5.1 Synthetic Data Dominance

The role of synthetic data is expanding:

- Synthetic captions (DALL-E 3's recaptioning) now standard
- Synthetic images for training (data augmentation via generation)
- Model-generated evaluation data

Prediction: Future models may be trained primarily on synthetic data, with real images providing grounding. The quality ceiling then depends on synthetic data quality, creating recursive improvement potential.

16.5.2 Data Quality Over Quantity

DataComp and similar studies show diminishing returns from raw scale:

- Curated 1B datasets matching uncurated 10B
- Quality filtering, deduplication, and alignment more important than size
- Curriculum and data mixing becoming sophisticated

Prediction: The arms race for dataset size will plateau. Competition will shift to curation quality, diversity, and specialized domain coverage.

16.5.3 Preference Optimization

Training on human preferences is expanding:

- RLHF and DPO for image generation
- Learned reward models guiding training
- Preference data as valuable asset

Prediction: Preference optimization will be as important for image models as it is for language models. Collecting and curating preference data will become a major activity.

16.6 Ecosystem Dynamics

16.6.1 Open vs Closed

The competitive landscape is bifurcating:

Open source:

- FLUX, Stable Diffusion, Qwen-Image leading
- Rapidly improving, closing gap to closed models
- Enables research, customization, local deployment

Closed source:

- GPT-Image, Nano Banana Pro, Midjourney
- Unique capabilities (search grounding, conversational refinement)
- Tighter safety controls, clearer liability

Prediction: Both ecosystems will thrive. Open models will dominate research and customization; closed models will dominate consumer applications requiring safety and support.

16.6.2 Specialization vs Generalization

Two strategies are emerging:

General-purpose: Single model handles all image tasks (generation, editing, understanding).

Specialized: Fine-tuned models for specific domains (medical imaging, product photography, game assets).

Prediction: General-purpose models will improve, but specialized models will persist where domain expertise and reliability matter. The market will support both.

16.6.3 Regulation and Standards

Regulatory pressure is increasing:

- Watermarking mandates (C2PA, SynthID)
- Disclosure requirements for AI-generated content
- Liability frameworks for harmful outputs

Prediction: Regulation will shape deployment more than technology. Compliance requirements may favor large players with resources to implement safety measures.

16.7 Research Priorities

Based on the gaps identified in Part VIII, we identify high-value research directions:

16.7.1 Near-Term (1-2 years)

- Better distillation for single-step high-quality generation
- Improved text rendering across languages
- Robust counting and binding through architectural innovation
- Efficient transformers for high-resolution generation

16.7.2 Medium-Term (3-5 years)

- Reliable compositional generation with soft guarantees
- Native video generation matching image quality
- Controllable generation through natural dialogue
- World models incorporating physical reasoning

16.7.3 Long-Term (5+ years)

- Verifiable generation with hard compositional guarantees
- Interactive, explorable 3D worlds from text
- Generation that understands and respects physics
- Creative AI as genuine collaborator, not tool

16.8 Conclusion: The Path Forward

Image generation has progressed from research curiosity to transformative technology in under five years. The trajectory suggests continued rapid advancement:

What's working:

- Scaling continues to improve quality
- Unified multimodal architectures show clear benefits
- Distillation enables practical deployment
- Open-source ecosystem accelerates research

What needs breakthroughs:

- Compositional reasoning requires new approaches, not just scaling
- World models need integration of physics and common sense
- Evaluation metrics remain inadequate
- Safety solutions are mitigations, not guarantees

The big picture: We are moving from "impressive demos" to "reliable tools." The remaining gaps—compositional reasoning, world knowledge, precise control—are not merely engineering challenges. They probe fundamental questions about how neural networks represent and reason about the visual world.

The next generation of advances will likely come from:

1. **Deeper integration** of generation with understanding and reasoning
2. **Structured representations** that enable compositional control
3. **World models** that capture physical and common-sense constraints
4. **Human-AI collaboration** patterns beyond prompting

Image generation is not approaching saturation—it is approaching a phase transition from pattern matching to visual reasoning. The models that achieve this transition will define the next era of AI capabilities.

17. Summary

This survey has traced the evolution of image generation and reasoning from first principles through frontier systems:

Part I established foundations: VAEs, VQ-VAE, score functions, and the architectural building blocks (CNNs, U-Nets, Transformers).

Part II examined core paradigms: diffusion models (DDPM, LDM, CFG), rectified flow (SD3, FLUX), and autoregressive approaches (LlamaGen, VAR, MaskGIT).

Part III analyzed critical components: tokenizers, text encoders (CLIP, T5, native multimodal), conditioning mechanisms (cross-attention, ControlNet, IP-Adapter), and landmark commercial systems (Imagen, DALL-E, GPT-Image, Nano Banana, Qwen-Image).

Part IV explored unified understanding and generation: VLM architectures, Chameleon, Emu3, Show-o, Transfusion, and native multimodal LLMs.

Part V surveyed structured generation: scene graphs (DisCo), layout conditioning (GLIGEN), neuro-symbolic methods, programmatic synthesis (DeTikZify, VectorFusion), and compositional benchmarks.

Part VI covered training and scaling: data pipelines, scaling laws, distributed training, and distillation.

Part VII examined evaluation: metrics (FID, CLIP, learned preferences), human evaluation protocols, and benchmark suites.

Part VIII documented weaknesses: compositional failures, visual quality gaps, control limitations, and open problems.

Part IX projected trends: unified multimodal models, efficiency advances, video and 3D, structured generation, and ecosystem dynamics.

The field stands at an inflection point. The paradigmatic questions—how to represent images, how to generate them, how to condition generation—have provisional answers. The remaining questions—how to compose reliably, how to reason visually, how to control precisely—point toward the next frontier.

References

[1] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. *International Conference on Machine Learning*, pages 1530–1538, 2015.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27, 2014.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[4] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2014.

[5] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286. PMLR, 2014.

[6] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations*, 2017.

[7] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[9] Aaron Van Den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[10] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[11] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2024.

[12] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations*, 2024.

[13] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[14] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[17] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[18] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[19] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[20] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7): 1661–1674, 2011.

[21] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

[23] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[24] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *International Conference on Machine Learning*, pages 8748–8763, 2021.

[26] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[27] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[28] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[29] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[30] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorber, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

[31] Black Forest Labs. FLUX: A new era of text-to-image generation. *Technical Report*, 2024. https://blackforestlabs.ai.

[32] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*, 2023.

[33] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[34] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

[35] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in Neural Information Processing Systems*, 37, 2024.

[36] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.

[37] Jianfeng Zhao et al. Image tokenization with grouped spherical quantization. *arXiv preprint arXiv:2412.02632*, 2024.

[38] Qihang Yu, Jose Lezama, Nitesh B Gundavarapu, Luca Golber, Kate Saenko, and Irfan Essa. TiTok: An image is worth 32 tokens for reconstruction and generation. *arXiv preprint arXiv:2406.07550*, 2024.

[39] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[40] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[41] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[42] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. IP-Adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023.

[43] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *International Conference on Machine Learning*, pages 8821–8831, 2021.

[44] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science*, 2(3):8, 2023.

[45] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in Neural Information Processing Systems*, 36, 2024.

[46] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024.

[47] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

[48] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[49] Jinheng Xie, Weijia Ye, Kai Chen, Yining Lou, Yanhong Zang, and Yu Li. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[50] Chunting Zhou, Lili Yu, Ram Pasunuru, Kushal Ghosh, Vladimir Karpukhin, et al. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.

[51] Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. SEED-X: Multimodal models with unified multi-granularity comprehension and generation. *arXiv preprint arXiv:2404.14396*, 2024.

[52] Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. NExT-GPT: Any-to-any multimodal LLM. *arXiv preprint arXiv:2309.05519*, 2023.

[53] Gemini Team. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[54] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1219–1228, 2018.

[55] Yikai Wang et al. Scene graph disentanglement and composition for generalizable complex image generation. *Advances in Neural Information Processing Systems*, 37, 2024.

[56] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. GLIGEN: Open-set grounded text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22511–22521, 2023.

[57] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023.

[58] Jinheng Xie, Yuexiang Li, Yawen Huang, Haozhe Liu, Wentian Zhang, Yefeng Zheng, and Mike Zheng Shou. BoxDiff: Text-to-image synthesis with training-free box-constrained diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7452–7461, 2023.

[59] Xinchen Zhang et al. RealCompo: Balancing realism and compositionality improves text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 37, 2024.

[60] Long Lian, Boyi Li, Adam Yala, and Trevor Darrell. LLM-grounded diffusion: Enhancing prompt understanding of text-to-image diffusion models with large language models. *Transactions on Machine Learning Research*, 2024.

[61] Hanan Gani, Shariq Farooq Basu, Matthias Segu, Alexander Bergman, et al. LLM blueprint: Enabling text-to-image generation with complex and detailed prompts. In *International Conference on Learning Representations*, 2024.

[62] Tianyi Wang et al. SPRING: Structured probabilistic image generation with neuro-symbolic constraints. *Artificial Intelligence*, 336, 2024.

[63] Yiran Chen et al. Factor graph diffusion models for compositional image generation. *Advances in Neural Information Processing Systems*, 37, 2024.

[64] Seonjae Kim et al. CountGen: Counting-aware text-to-image generation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[65] Lital Binyamin et al. Make it count: Text-to-image generation with an accurate number of objects. *arXiv preprint arXiv:2406.10210*, 2024.

[66] Wei Zhang et al. YOLO-Count: Differentiable object counting for text-to-image generation. *arXiv preprint arXiv:2508.00728*, 2025.

[67] Khimya Halbert, Ziyuan Jia, and Mayur Naik. Learning neurosymbolic generative models via program synthesis. *International Conference on Machine Learning*, pages 2499–2508, 2019.

[68] Zhan Li et al. Self-training large language models for improved visual program synthesis with visual reinforcement. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[69] Jonas Belouadi, Anne Biermann, and Christin Seifert. DeTikZify: Synthesizing graphics programs for scientific figures and sketches with TikZ. *Advances in Neural Information Processing Systems*, 37, 2024.

[70] Jonas Belouadi and Christin Seifert. AutomaTikZ: Text-guided synthesis of scientific vector graphics with TikZ. *International Conference on Learning Representations*, 2024.

[71] Chen Liu et al. MagicGeo: Coordinate-aware geometric figure generation with LLMs. *arXiv preprint*, 2025.

[72] Ajay Jain, Amber Xie, and Pieter Abbeel. VectorFusion: Text-to-SVG by abstracting pixel-based diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1911–1920, 2023.

[73] Yixing Wu et al. Chat2SVG: Vector graphics generation with large language models and text-guided diffusion. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[74] Yiying Zhang et al. OmniSVG: A unified scalable vector graphics generation model. *arXiv preprint*, 2024.

[75] Zhiyu Yang, Zihan Zhou, Shuo Wang, Fangyin Cong, Pinghui Xu, Yao Wang, and Yanfeng Lan. MatPlotAgent: Method and evaluation for LLM-based agentic scientific data visualization. *arXiv preprint arXiv:2402.11453*, 2024.

[76] Ahmed Masry et al. SynChart: Synthesizing charts from language using large language models. *arXiv preprint*, 2024.

[77] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2I-CompBench: A comprehensive benchmark for open-world compositional text-to-image generation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[78] Dhruba Ghosh et al. GENEVAL: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36, 2023.

[79] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. LAION-5B: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

[80] Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smber, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghber, Jenia Zhang, et al. DataComp: In search of the next generation of multimodal datasets. *Advances in Neural Information Processing Systems*, 36, 2023.

[81] Wendi Zheng, Jiayan Teng, Zhuoyi Yang, Weihan Wang, Jidong Chen, Xiaotao Gu, Yuxiao Dong, Ming Ding, and Jie Tang. CogView3: Finer and faster text-to-image generation via relay diffusion. *arXiv preprint arXiv:2403.05121*, 2024.

[82] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *International Conference on Machine Learning*, pages 8162–8171, 2021.

[83] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-SNR weighting strategy. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7441–7451, 2023.

[84] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. ZeRO: Memory optimizations toward training trillion parameter models. *International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16, 2020.

[85] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in Neural Information Processing Systems*, 30, 2017.

[86] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, 29, 2016.

[87] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019.

[88] Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A Smith. TIFA: Accurate and interpretable text-to-image faithfulness evaluation with question answering. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20406–20417, 2023.

[89] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2023.

[90] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

[91] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2023.