

Basiswissen KI-Bildgenerierung: Wie Diffusion funktioniert



Links die Zwischenstände eines Laufs, rechts die Kette, die sie erzeugt. Bild selbst erzeugt und als KI-generiert gekennzeichnet.

Wissen	Beratung	Schulung
Alle Grundlagen zur KI-Bildgenerierung an einer Stelle – vom Prinzip bis zur Governance.	Von der Testinstallation zur belastbaren Bildproduktion mit Nachweis und klaren Zuständigkeiten.	Dein Team versteht, was hinter den Reglern passiert – und warum ein Bild wiederkommt oder nicht.
KI-Bildgenerierung	KI-Bildgenerierung Beratung	KI-Bildgenerierung Schulung

In einer Werkhalle in Südwestfalen habe ich vor einiger Zeit eine Frage gestellt bekommen, die ich seitdem als Eingangstest benutze. Der Marketingleiter zeigte auf den Bildschirm, auf dem gerade ein Produktfoto entstand, und sagte: „Das Ding sucht sich das doch irgendwo raus, oder? Aus dem Internet?“ Die Kollegin daneben widersprach: Nein, das Modell habe alle Bilder gespeichert und setze sie neu zusammen, wie eine Collage. Beide lagen falsch, und beide lagen falsch auf eine Weise, die teuer wird – nicht technisch, sondern rechtlich und organisatorisch. Wer glaubt, das Modell suche Bilder heraus, stellt die falschen Fragen an die Rechtsabteilung. Wer glaubt, es sei eine Collagemaschine, sucht in den Ergebnissen nach Bildfetzen, die dort nicht sind, und übersieht die Fehler, die tatsächlich drin stecken.

Die Wahrheit ist unspektakulärer und deutlich nützlicher: Ein Bildmodell ist ein sehr gut trainierter Aufräumdienst. Es bekommt ein Feld voller Zufallszahlen vorgesetzt und die Aufgabe, daraus das Rauschen zu entfernen. Was übrig bleibt, wenn man das oft genug macht, ist ein Bild. Kein gefundenes, kein zusammengeklebtes, sondern eines, das aus der Statistik von Millionen Trainingsbildern rekonstruiert wurde. Dieser Beitrag erklärt genau das: wie Diffusion funktioniert,

welche vier Bauteile beteiligt sind, an welcher Stelle jeder Regler der Oberfläche eingreift – und warum du dasselbe Bild morgen nur dann wiederbekommst, wenn eine ganze Kette unverändert bleibt.

Es lohnt sich, das einmal sauber zu verstehen. Nicht, weil Grundlagenwissen hübsch ist, sondern weil praktisch jedes Alltagsproblem der Bildproduktion an einer bestimmten Stelle dieses Ablaufs entsteht. Ein Prompt wird ignoriert? Text-Encoder. Das Bild ist matschig? Schrittzahl oder Sampler. Die Farben sind ausgewaschen? Falscher VAE. Das Bild von vorgestern kommt nicht wieder? Irgendwo in der Kette hat sich etwas bewegt. Wenn du weißt, wie die Maschine läuft, wird die Fehlersuche vom Ratespiel zur Diagnose. Den großen Überblick über das gesamte Themenfeld findest du unter [KI-Bildgenerierung](#).

Was du aus diesem Beitrag mitnimmst

Diffusion heißt: Ein Modell lernt, Rauschen von einem Bild zu trennen – und wendet dieses Wissen rückwärts an, um aus reinem Rauschen ein Bild zu gewinnen.

Das Modell speichert keine Bilder. Ein Checkpoint mit mehreren Gigabyte hat Milliarden Trainingsbilder gesehen; gespeichert sind Gewichte, keine Bilddateien.

Vier Bauteile sind beteiligt: Text-Encoder, Latentraum mit Startrauschen, Denoiser und VAE-Decoder. Jeder Regler in der Oberfläche greift an genau einer dieser Stellen an.

Die aktuelle Modellgeneration rechnet streng genommen nicht mehr mit klassischer Diffusion, sondern mit Flow Matching entlang gerader Bahnen. Der Begriff Diffusion ist als Sammelbezeichnung geblieben.

Nicht jeder Bildgenerator ist ein Diffusionsmodell. OpenAIs GPT-Image-Reihe erzeugt Bilder autoregressiv, Bildstück für Bildstück – dort greift das Diffusionsmodell im Kopf nicht.

Reproduzierbarkeit ist keine Eigenschaft des Seeds, sondern einer Kette aus rund fünfzehn Gliedern. Jedes davon kann dir dasselbe Bild kaputtmachen.

Vom Rauschen zum Bild: das Grundprinzip

Die meisten Erklärungen zur KI-Bildgenerierung fangen mit dem Prompt an. Das ist pädagogisch verständlich und technisch irreführend, denn der Prompt ist nicht der Anfang. Der Anfang ist das Rauschen. Der Prompt ist nur die Richtungsangabe, die dem Aufräumdienst sagt, worauf er beim Aufräumen achten soll.

Der Vorwärtsprozess: Bilder kaputtmachen als Lehrplan

Ein Bildmodell wird nicht darauf trainiert, Bilder zu malen. Es wird darauf trainiert, eine sehr eng gefasste Frage zu beantworten. Der Trick dabei ist, dass man sich die Trainingsaufgaben selbst bauen kann, ohne dass ein Mensch etwas beschriften müsste.

Man nimmt ein Trainingsbild und schüttet Rauschen darüber. Nicht alles auf einmal, sondern in definierten Dosen: ein bisschen, mehr, viel, bis am Ende nichts mehr vom Motiv zu erkennen ist und nur noch Zufall übrig bleibt. Weil man die Dosis selbst festgelegt hat, kennt man die Lösung: Man weiß exakt, welches Rauschen man draufgekippt hat. Das ist der Vorwärtsprozess, und er ist reine Buchhaltung – dabei wird nichts gelernt.

Gelernt wird beim Aufräumen. Das Modell bekommt ein verrauschtes Bild und die Angabe, wie stark es verrauscht ist, und soll schätzen, welches Rauschen darauf liegt. Anschließend vergleicht man die Schätzung mit der bekannten Wahrheit und korrigiert die Gewichte. Milliardenfach, über Monate, auf Rechenclustern, deren Stromrechnung man lieber nicht sehen will. Das Ergebnis ist ein Modell, das bei jedem beliebigen Rauschpegel eine erstaunlich gute Schätzung abgibt.

Der Rückwärtsprozess: aus Zufall wird Motiv

Jetzt dreht man die Sache um. Man gibt dem Modell kein verrauschtes Bild, sondern reines Rauschen – ein Feld voller Zufallszahlen, gezogen aus einer einzigen Zahl, dem Seed. Das Modell schätzt, welches Rauschen darauf liegt. Man zieht einen Teil davon ab. Das Ergebnis ist immer noch fast nur Rauschen, aber eben ein kleines bisschen weniger. Dann fragt man erneut. Und wieder. Und nach zwanzig, dreißig Runden steht dort ein Bild.

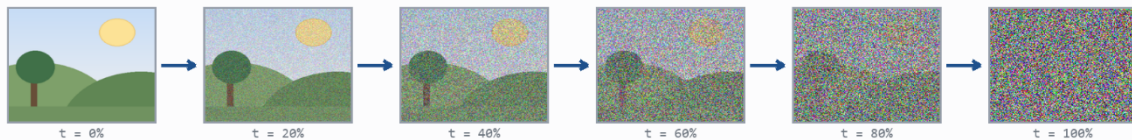
Das klingt nach einem Taschenspielertrick, und in gewisser Weise ist es einer. Es gab nie ein Bild unter dem Rauschen. Das Modell hat lediglich bei jedem Schritt die wahrscheinlichste Interpretation dessen geliefert, was da liegen könnte – und weil seine Vorstellung davon, wie Bilder aussehen, aus Milliarden Beispielen stammt, landet es verlässlich bei etwas, das wie ein Foto oder eine Illustration aussieht.

Zwei Richtungen, ein Modell

Oben wird beim Training kaputtgemacht. Unten wird beim Erzeugen repariert.

Vorwärtsprozess · Training

Bekanntes Bild, definiert verrauscht. Die Antwort ist bekannt.

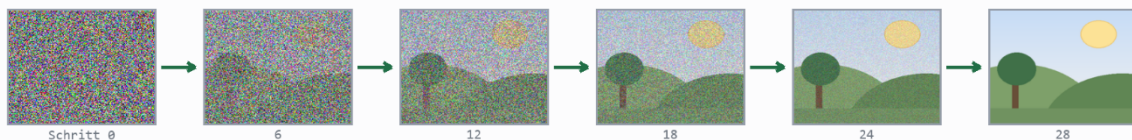


Gelernt wird nur eine einzige Frage:

Wie viel Rauschen liegt auf diesem Feld – und in welche Richtung liegt das saubere Bild?

Rückwärtsprozess · Erzeugen

Unbekanntes Rauschen aus dem Seed. Die Antwort wird geschätzt – Schritt für Schritt.



Achtung: Jeder Schritt schätzt das ganze Rauschen. Der Sampler zieht davon nur einen Teil ab.

Oben der Vorwärtsprozess im Training, unten der Rückwärtsprozess beim Erzeugen. Die Rauschfelder stammen aus echten Zufallszahlen mit festem Startwert.

Der häufigste Denkfehler beim Entrauschen

Fast jede Erklärung behauptet, das Modell entferne pro Schritt „ein bisschen Rauschen“. Das ist nicht, was passiert. Das Modell schätzt bei jedem einzelnen Schritt das gesamte Rauschen – es sagt also implizit: „So würde das fertige Bild aussehen.“

Dass trotzdem nicht nach einem Schritt Schluss ist, liegt am Sampler. Der nimmt die Vorhersage als Richtungsangabe und geht nur ein kurzes Stück in diese Richtung. Danach wird neu gefragt, und weil das Feld jetzt weniger verrauscht ist, fällt die Schätzung besser aus.

Das erklärt zwei Beobachtungen aus dem Alltag: Erstens sieht das Bild schon nach wenigen Schritten grob richtig aus und wird danach nur noch schärfer. Zweitens ist eine hohe Schrittzahl kein Qualitätsgarant, sondern ab einem Punkt schlicht Wartezeit.

Was heute unter dem Wort Diffusion läuft

Jetzt kommt der Teil, den die meisten Grundlagentexte verschlafen haben. Das eben beschriebene Verfahren mit tausend kleinen Rauschschritten und einer Rauschvorhersage war der Stand von

2022 bis etwa 2024 – die Zeit von Stable Diffusion 1.5 und SDXL. Die aktuelle Modellgeneration arbeitet anders.

Seit Stable Diffusion 3 und FLUX setzt die Branche auf Flow Matching, meist in der Variante Rectified Flow. Die Idee: Statt eines verschlungenen Rauschpfads zieht man eine möglichst gerade Linie zwischen reinem Rauschen und dem fertigen Bild. Das Modell sagt nicht mehr voraus, welches Rauschen auf dem Feld liegt, sondern in welche Richtung und mit welcher Geschwindigkeit man sich entlang dieser Linie bewegen muss. Eine gerade Linie lässt sich mit wenigen großen Schritten abfahren, eine krumme nicht – deshalb kommen moderne Modelle mit deutlich weniger Schritten aus als ihre Vorgänger.

Für die Bedienung ändert das wenig: Es gibt weiterhin Seed, Schritte, Sampler und einen Guidance-Regler. Für das Verständnis ändert es einiges, weil einige Faustregeln aus der alten Welt nicht mehr gelten. Der Sammelbegriff „Diffusion“ ist geblieben, die Firmen heißen weiter so, die Werkzeuge auch. Wer es genau nimmt, sagt: Diffusionsmodelle und Flow-Matching-Modelle sind zwei Formulierungen derselben Grundidee – aus Zufall wird über eine gelernte Bahn ein Bild.

Aspekt	Klassische Diffusion (bis SDXL)	Flow Matching (ab SD 3 / FLUX)
Vorhersage	Das Rauschen auf dem Feld	Geschwindigkeit entlang der Bahn
Bahn Rauschen zu Bild	Krumm, viele kleine Schritte nötig	Möglichst gerade, große Schritte möglich
Typische Schrittzahl	25 bis 50	20 bis 30, destilliert 4 bis 8
Rechenkern	U-Net mit Faltungsschichten	Transformer, Text und Bild in gemeinsamer Aufmerksamkeit
Guidance	Klassisches CFG mit zwei Vorhersagen	Oft eingebacken, Regler wirkt als Kennwert
Auflösungsverhalten	Fest an die Trainingsauflösung gebunden	Zeitachse wird je nach Auflösung verschoben

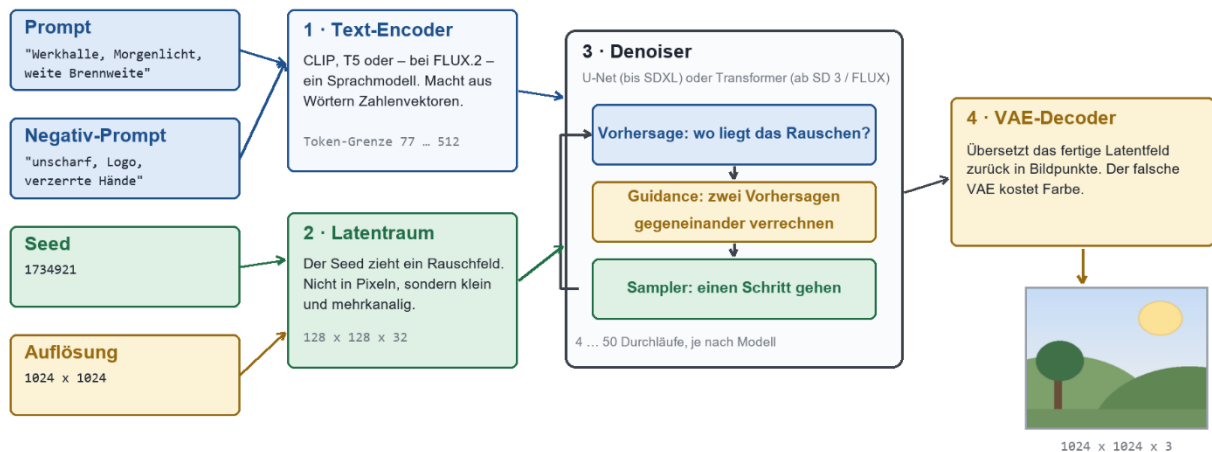
Wie viele Schritte dein Modell tatsächlich braucht und woran du die Sättigung erkennst, steht ausführlich in [Sampling-Steps erklärt](#); welcher Algorithmus die Schritte abfährt, klärt [Sampler & Scheduler erklärt](#).

Die vier Bauteile einer Bildmaschine

Was in der Oberfläche wie ein Kasten aussieht, in den oben Text hineingeht und unten ein Bild herausfällt, sind in Wirklichkeit vier eigenständige Modelle, die hintereinandergeschaltet arbeiten. Sie kommen teils aus unterschiedlichen Häusern, sie werden getrennt trainiert, und man kann sie einzeln austauschen. Genau das ist der Grund, warum so viel schiefgehen kann.

Die vier Bauteile und der Weg dazwischen

Vier Eingaben links, ein Bild rechts. In der Mitte die Schleife, die alle Rechenzeit frisst.



Warum überhaupt ein Latentraum?

Ein Bild mit 1024 x 1024 Punkten hat 3.145.728 Farbwerte.

Als Latentfeld mit 8-facher Kantenkompression und 4 Kanälen (SD 1.5, SDXL): 128 x 128 x 4 = 65.536 Werte – Faktor 48 weniger.

Mit 16 Kanälen (SD 3.5, FLUX.1): 128 x 128 x 16 = 262.144 Werte – Faktor 12, dafür deutlich mehr Feindetail.

Mit 32 Kanälen (FLUX.2): noch mehr Detailreserve, entsprechend mehr Rechenlast pro Schritt.

Die Kette von der Eingabe bis zum Bildpunkt. Die Rechenzeit steckt fast vollständig in der Schleife in der Mitte.

Bauteil 1: Der Text-Encoder

Das Modell versteht keine Sprache. Es versteht Zahlen. Der Text-Encoder ist das Bauteil, das aus deinem Prompt einen Satz von Vektoren macht – Zahlenreihen, die die Bedeutung der Wörter in einem Raum abbilden, in dem „Hund“ und „Wolf“ nah beieinanderliegen und „Hund“ und „Gabelstapler“ weit auseinander.

Welcher Encoder zum Einsatz kommt, hat sich über die Modellgenerationen massiv verändert, und daran hängt direkt, wie gut ein Modell längere Prompts versteht. Stable Diffusion 1.5 benutzte einen einzelnen CLIP-Encoder mit 77 Token Fassungsvermögen – etwa fünfzig deutsche Wörter, danach wird abgeschnitten. SDXL kam mit zwei CLIP-Encodern, blieb aber bei 77 Token. Stable Diffusion 3.5 benutzt drei Encoder parallel: zwei CLIP-Varianten für die grobe Bildsemantik und zusätzlich T5-XXL, ein Sprachmodell, das mit ganzen Sätzen umgehen kann. FLUX.1 kombinierte CLIP mit T5-XXL bei 512 Token. FLUX.2 ist den Schritt konsequent zu Ende gegangen und benutzt nur noch einen einzigen Encoder – das Sprachmodell Mistral Small 3.1, ebenfalls mit 512 Token und mit der Besonderheit, dass nicht nur dessen letzte Schicht ausgewertet wird, sondern mehrere Zwischenschichten.

Diese Entwicklung erklärt eine Alltagsbeobachtung: Ältere Modelle reagieren auf Prompts eher wie auf eine Stichwortliste, neuere tatsächlich wie auf einen Auftrag. Wenn ein Modell deine Anweisung trotzdem ignoriert, liegt die Ursache fast immer hier – im Encoder oder in dessen Token-Grenze. Was du dagegen tun kannst, steht in [Warum die KI den Prompt ignoriert](#) und in [Prompt und Negativ-Prompt in der KI-Bildgenerierung](#).

Bauteil 2: Der latente Raum

Niemand rechnet in Bildpunkten. Ein Bild mit 1024 mal 1024 Punkten hat 3.145.728 Farbwerte, und dreißig Durchläufe über diese Datenmenge wären selbst auf einer teuren Grafikkarte eine Zumutung. Stattdessen findet die gesamte Diffusion in einer komprimierten Darstellung statt, dem latenten Raum.

Man kann sich das wie eine Landkarte vorstellen. Die Karte ist viel kleiner als das Gelände, sie zeigt trotzdem alles Wesentliche, und sie zeigt es in einer Sprache, die für die Aufgabe passender ist – Höhenlinien statt Erdreich. Der latente Raum eines Bildmodells hat typischerweise ein Achtel der Kantenlänge, dafür aber mehrere Kanäle statt der drei Farbkanäle. Bei Stable Diffusion 1.5 und SDXL sind es vier Kanäle, macht 128 mal 128 mal 4 gleich 65.536 Werte – Faktor 48 weniger als im Pixelbild. Bei Stable Diffusion 3.5 und FLUX.1 sind es sechzehn Kanäle, bei FLUX.2 zweiunddreißig. Mehr Kanäle heißen mehr Detailreserve, insbesondere für feine Strukturen und lesbare Schrift, und gleichzeitig mehr Rechenlast pro Schritt.

Woran du den Latentraum im Alltag merkst

Die Achterkompression ist der Grund, warum Auflösungen in Vielfachen von 8, in der Praxis meist von 64 angegeben werden. Krumme Werte werden intern gerundet, und dabei verschiebt sich der Bildausschnitt gegenüber dem, was du eingestellt hast.

Sie ist auch der Grund, warum Bilder jenseits der Trainingsauflösung in doppelte Köpfe und gespiegelte Gliedmaßen zerfallen: Das Latentgitter wird größer, als das Modell es je gesehen hat, und die Bildkomposition verliert den Zusammenhalt.

Welche Auflösung dein Modell wirklich beherrscht und wie du größere Formate in zwei Stufen erzeugt, steht in [Auflösung & Seitenverhältnis bei KI-Bildern](#).

Bauteil 3: Der Denoiser

Das ist das eigentliche Modell – der Teil, der die Schätzung abgibt und in dem praktisch alle Parameter stecken. Auch hier hat es einen Architekturwechsel gegeben, der sich in der Bildqualität deutlich niederschlägt.

Bis SDXL war der Denoiser ein U-Net: ein Faltungsnetz, das das Latentfeld stufenweise verkleinert, im Zentrum die grobe Bildidee verarbeitet und dann wieder vergrößert, mit Querverbindungen zwischen den Stufen. Der Prompt kam über sogenannte Cross-Attention-Schichten hinein, an denen sich jeder Bildbereich fragen konnte, welche Wörter für ihn relevant sind. Seit Stable Diffusion 3 und FLUX ist der Denoiser ein Transformer – dieselbe Bauart, die auch in Sprachmodellen steckt. Bild und Text laufen dabei nicht mehr getrennt mit gelegentlichem Kontakt, sondern gemeinsam durch die Aufmerksamkeitsschichten. FLUX.2 in der dev-Variante besteht aus acht Doppelstrom- und achtundvierzig Einzelstromblöcken und bringt rund 32 Milliarden Parameter mit.

Diese Zahl ist der Ort, an dem alte Grundlagentexte am sichtbarsten veraltet sind. Ein Stable-Diffusion-1.5-Checkpoint war rund vier Gigabyte groß und hatte etwa 860 Millionen Parameter im Denoiser. Wir reden heute über Modelle, die mehr als das Dreißigfache an Parametern mitbringen und die auch nach Quantisierung ab etwa 13 Gigabyte Videospeicher verlangen. Der Sprung in der Bildqualität zwischen 2023 und 2026 ist zu einem guten Teil schlicht ein Größensprung.

Was in einem Modell nicht drin ist

Ein Checkpoint speichert keine Bilder. Die 5,85 Milliarden Bild-Text-Paare des LAION-5B-Datensatzes, mit denen die ersten Stable-Diffusion-Versionen trainiert wurden, passen nicht in ein paar Gigabyte Gewichte – schon rein rechnerisch nicht.

Das heißt aber nicht, dass nichts durchschlägt. Motive, die im Training extrem häufig und extrem gleichförmig vorkamen – bekannte Logos, Filmfiguren, ikonische Fotos – kann ein Modell so genau reproduzieren, dass die Ähnlichkeit rechtlich relevant wird.

Für die Praxis heißt das: Die Frage „hat die KI hier abgeschrieben?“ ist technisch falsch gestellt, aber die Prüfung des Ergebnisses auf fremde Marken und erkennbare Personen bleibt trotzdem Pflicht.

Bauteil 4: Der VAE-Decoder

Am Ende steht das Latentfeld fertig da, und irgendjemand muss daraus Bildpunkte machen. Das erledigt der Decoder eines Variational Autoencoders. Sein Gegenstück, der Encoder, wird gebraucht, sobald ein vorhandenes Bild in den Prozess eingespeist wird – beim Bild-zu-Bild-Verfahren, beim Inpainting, beim Hochskalieren.

Der VAE ist das unauffälligste Bauteil und dasjenige, das am häufigsten für rätselhafte Ergebnisse sorgt. Ein VAE, der nicht zum Modell gehört, produziert ausgewaschene Farben, einen Grauschleier oder – der Klassiker bei falscher Zuordnung – einen violetten Stich. Der Bildaufbau stimmt dabei vollkommen, nur die Übersetzung am Schluss ist falsch. Wer dann am Prompt schraubt, sucht an der falschen Stelle.

Welche VAE zu welchem Modell gehört, wann er separat geladen werden muss und wann er schon im Checkpoint steckt, ist Thema von [Checkpoints, Modelle und VAE](#).

Modellgeneration	Text-Encoder	Latentkanäle	Denoiser
Stable Diffusion 1.5	CLIP ViT-L, 77 Token	4	U-Net, ca. 860 Mio. Parameter
SDXL	CLIP ViT-L + OpenCLIP bigG, 77 Token	4	U-Net, ca. 2,6 Mrd. Parameter
Stable Diffusion 3.5	2x CLIP + T5-XXL	16	MMDiT-Transformer
FLUX.1	CLIP + T5-XXL, 512 Token	16	Transformer, 12 Mrd. Parameter
FLUX.2 dev	Mistral Small 3.1, 512 Token	32	Transformer, ca. 32 Mrd. Parameter
Qwen-Image	Eigener Sprach-Encoder	16	Transformer, Apache-2.0-Lizenz

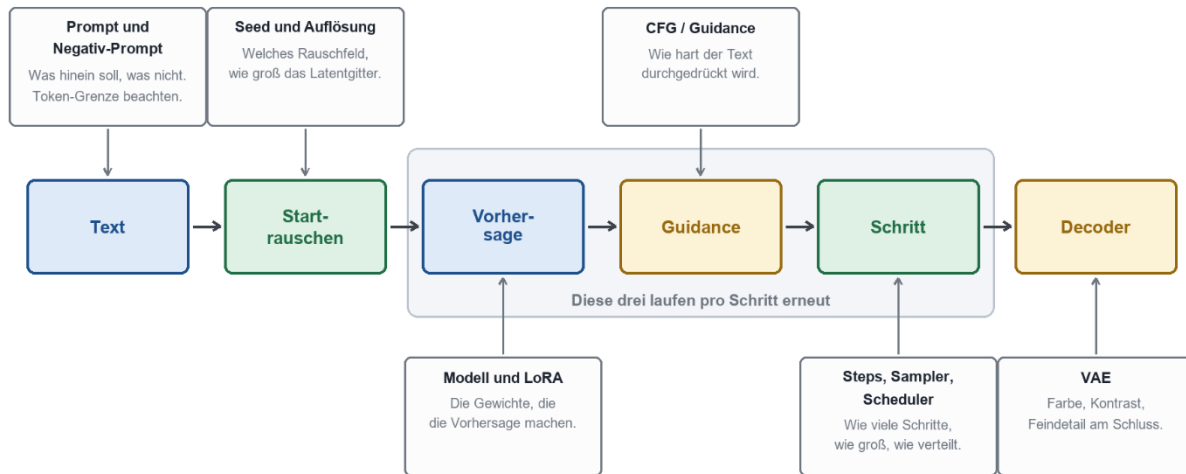
Welches dieser Modelle für welchen Zweck taugt, wie die Lizenzen aussehen und was die Hardware kostet, vergleicht [Die 15 wichtigsten Modelle zur lokalen Bildgenerierung](#); die Anbieterseite dazu findest du in [Übersicht der wichtigsten KI-Bildgeneratoren](#).

Die Regler – und wo sie wirklich eingreifen

Jetzt wird das Ganze praktisch. Wenn du den Ablauf einmal im Kopf hast, ist jeder Regler der Oberfläche einer bestimmten Station zugeordnet, und das Verhalten wird vorhersagbar. Der folgende Abschnitt geht die Stellschrauben in der Reihenfolge durch, in der sie wirken.

Wo greift welcher Regler ein?

Dieselbe Kette wie zuvor, diesmal mit den Stellschrauben der Oberfläche.



Destillierte Modelle haben die Guidance eingebacken: Dort bleibt CFG auf 1, sonst kippt das Bild.

Jede Einstellung greift an genau einer Station an. Drei davon laufen in der Schleife und werden pro Schritt neu ausgewertet.

Seed: welches Rauschen am Anfang steht

Der Seed ist der Startwert eines Pseudozufallsgenerators. Aus ihm wird das Rauschfeld gezogen, mit dem alles beginnt. Er entscheidet nicht, ob ein Bild gut wird, sondern nur, welches von unendlich vielen möglichen Bildern du bekommst. Es gibt keine guten und keine schlechten Seeds, und benachbarte Zahlen ergeben keine ähnlichen Bilder – Seed 41 und Seed 42 haben nichts miteinander zu tun.

Der Seed ist die Voraussetzung für Reproduzierbarkeit, aber allein reicht er nicht. Warum, steht im nächsten Kapitel und ausführlich in [Der Seed bei der KI-Bildgenerierung](#).

Auflösung: wie groß das Latentgitter ist

Die eingestellte Auflösung bestimmt die Größe des Rauschfelds im latenten Raum. Bei Flow-Matching-Modellen kommt eine Feinheit hinzu: Weil ein größeres Feld pro Schritt mehr Unordnung enthält, wird die Zeitachse des Prozesses je nach Auflösung verschoben. In der Fachsprache heißt das Timestep Shift. In manchen Oberflächen taucht dafür ein Regler namens „shift“ oder „mu“ auf. Wer ihn verstellt, ohne zu wissen, was er tut, bekommt entweder ein matschiges oder ein überzeichnetes Bild.

Prompt und Negativ-Prompt: die Richtungsangabe

Der Prompt geht durch den Text-Encoder und liegt danach als Vektorsatz vor. Der Negativ-Prompt ebenfalls – er wird nicht etwa „abgezogen“, sondern als zweite Bedingung parallel geführt. Bei klassischer Guidance macht das Modell pro Schritt eine Vorhersage für den positiven und eine für den negativen Prompt und entfernt sich dann von der negativen. Bei destillierten Modellen, bei denen die Guidance eingebacken ist, hat der Negativ-Prompt schlicht keine Wirkung mehr. Das erklärt, warum Anwender, die von SDXL auf ein schnelles FLUX-Modell wechseln, ihre sorgsam gepflegte Negativliste plötzlich wirkungslos vorfinden.

Guidance und CFG: wie hart der Text durchgedrückt wird

Classifier-Free Guidance ist der Regler mit dem schlechtesten Ruf, weil er am häufigsten falsch bedient wird. Technisch passiert Folgendes: Das Modell macht pro Schritt zwei Vorhersagen, eine

mit Prompt und eine ohne. Die Differenz zwischen beiden ist die Richtung, in die der Prompt zieht. Dieser Vektor wird mit dem CFG-Wert multipliziert und aufaddiert. Ein Wert von 1 bedeutet: keine Verstärkung. Werte um 7 waren bei SDXL üblich. Sehr hohe Werte treiben das Bild in Übersättigung, harte Kanten und Doppelkonturen.

Der Preis ist Rechenzeit: Zwei Vorhersagen pro Schritt bedeuten doppelten Aufwand. Deshalb gibt es destillierte Modelle, bei denen die Guidance ins Modell hineintrainiert wurde. FLUX.1 schnell arbeitet so, ebenso die destillierten Varianten von FLUX.2 klein, die seit Januar 2026 in Größen von 4 und 9 Milliarden Parametern verfügbar sind und mit vier Schritten auskommen. Bei diesen Modellen muss CFG auf 1 stehen. Der Regler, der in der Oberfläche trotzdem „guidance“ heißt und etwa auf 3,5 steht, ist etwas anderes: ein Kennwert, der dem Modell mitgeteilt wird, kein Verstärkungsfaktor.

Die zwei häufigsten Fehlbedienungen bei destillierten Modellen

CFG hochdrehen: Bei einem destillierten Modell führt das nicht zu mehr Prompt-Treue, sondern zu verbrannten Farben und Artefakten. Der Wert bleibt auf 1.

Schrittzahl hochdrehen: Ein auf vier Schritte destilliertes Modell wird bei zwölf Schritten nicht besser, sondern schlechter. Es wurde auf einen bestimmten Pfad optimiert und verlässt ihn bei zu vielen Schritten.

Wie du den passenden Wert für dein Modell findest und woran du erkennst, dass er zu hoch ist, steht in [CFG-Scale: Guidance in KI-Bildgeneratoren](#).

Steps, Sampler und Scheduler: wie der Weg abgefahren wird

Diese drei gehören zusammen und werden regelmäßig durcheinandergeworfen. Der Scheduler legt fest, wie die Schritte über den Rauschpegel verteilt werden – vorne grob, hinten fein, oder gleichmäßig. Der Sampler ist das Rechenverfahren, das aus der Modellvorhersage den nächsten Zustand ermittelt; Euler ist das einfachste, DPM++-Varianten kommen mit weniger Schritten zum Ziel, indem sie die Krümmung des Wegs berücksichtigen. Die Schrittzahl bestimmt, wie fein der Weg abgetastet wird.

Zwei Dinge sind wichtig zu wissen. Erstens: Sampler und Scheduler verändern das Bild, nicht nur die Geschwindigkeit. Derselbe Seed mit Euler und mit DPM++ 2M ergibt zwei verschiedene Bilder. Zweitens: Manche Sampler sind stochastisch, fügen also während des Laufs zusätzlichen Zufall hinzu. Solche Verfahren konvergieren nicht – mehr Schritte ergeben nicht ein genaueres, sondern ein anderes Bild.

Modell, LoRA und VAE: was rechnet und was übersetzt

Das geladene Modell bestimmt, welche Gewichte die Vorhersage machen. Ein LoRA hängt sich als kleiner Zusatz an einen Teil dieser Gewichte und verschiebt die Vorhersage in eine bestimmte Richtung – hin zu einem Stil, einer Person, einem Produkt. Es ersetzt das Modell nicht, es verstimmt es kontrolliert. Deshalb wirkt ein LoRA-Gewicht wie ein Mischregler, und deshalb kippt das Bild, wenn man mehrere LoRAs gleichzeitig zu stark aufdreht.

Der VAE greift ganz am Schluss ein und hat auf den Bildinhalt keinen Einfluss mehr, wohl aber auf Farbe, Kontrast und Feinzeichnung. Wie ein LoRA technisch funktioniert, erklärt [Was ist ein LoRA?](#).

Einstellung	Greift ein bei	Verändert	Wirkt pro Schritt neu
Seed	Startrauschen	Welches Bild überhaupt entsteht	nein
Auflösung	Größe des Latentgitters	Bildaufbau, Komposition	nein

Prompt	Text-Encoder	Inhalt und Stil	nein
Negativ-Prompt	Text-Encoder, zweite Bedingung	Was vermieden wird	ja
CFG / Guidance	Zwischen Vorhersage und Schritt	Bindung an den Text	ja
Steps	Anzahl der Durchläufe	Feinheit der Lösung	—
Sampler	Schrittberechnung	Pfad und Bildcharakter	ja
Scheduler	Verteilung der Rauschpegel	Wo Detail entsteht	ja
Modell / LoRA	Gewichte des Denoisers	Stil, Wissen, Motivreue	ja
VAE	Decoder am Ende	Farbe, Kontrast, Feindetail	nein
batch_size	Parallele Berechnung	Nichts – im Idealfall	—

Die letzte Zeile hat einen Haken, den viele erst merken, wenn ein Nachlauf nicht dasselbe Bild liefert: Je nach Werkzeug wird das Startrauschen für einen Stapel anders gezogen als für ein Einzelbild. Der Fall ist in [batch_size und Reproduzierbarkeit](#) auseinandergenommen.

Dieselbe Maschine, anderer Startpunkt

Der schönste Nebeneffekt dieses Verständnisses: Fast alle Verfahren, die in Werkzeugen als eigene Funktionen auftauchen, sind kein anderes Verfahren. Es ist derselbe Ablauf, nur an einer anderen Stelle angestochen oder mit einer zusätzlichen Bedingung versehen. Wer das einmal gesehen hat, muss nicht mehr für jede Funktion eine eigene Erklärung lernen.

Bild zu Bild, Inpainting und Outpainting

Beim Bild-zu-Bild-Verfahren startet der Lauf nicht bei reinem Rauschen, sondern in der Mitte. Das vorhandene Bild wird vom VAE-Encoder in den latenten Raum gebracht, dort wird Rauschen bis zu einem bestimmten Pegel hinzugefügt, und ab dort läuft der normale Rückwärtsprozess. Der Regler, der in der Oberfläche „Denoising Strength“ heißt, ist nichts anderes als die Angabe, wie weit hinten im Prozess du einsteigst. Ein Wert von 0,3 bedeutet: Es bleiben nur die letzten dreißig Prozent des Wegs, also ändert sich wenig. Ein Wert von 0,9 bedeutet: fast von vorn, also bleibt vom Original nur eine grobe Erinnerung.

Inpainting ist dasselbe, nur auf einen Bereich begrenzt: Nach jedem Schritt wird das Latentfeld außerhalb der Maske auf den Originalzustand zurückgesetzt. Deshalb bleibt der Rest des Bildes erhalten, und deshalb kann es an den Maskenrändern zu sichtbaren Übergängen kommen – dort treffen zwei unterschiedlich entstandene Bereiche aufeinander. Outpainting ist Inpainting auf einer vergrößerten Leinwand, bei der alles Neue als Maske gilt.

Zusätzliche Bedingungen: ControlNet und Referenzbilder

Ein ControlNet gibt dem Denoiser bei jedem Schritt eine zweite Information neben dem Text: eine Kantenzeichnung, ein Posenskelett, eine Tiefenkarte. Der Prompt sagt, was zu sehen ist, das ControlNet sagt, wo es steht. Technisch läuft neben dem eigentlichen Denoiser ein zweites, kleineres Netz mit, dessen Ausgaben in die Zwischenschichten eingespeist werden.

Hier hat sich seit 2024 einiges verschoben. Die aktuelle Modellgeneration nimmt Referenzbilder direkt als Teil der Eingabe entgegen, ohne separates Zusatznetz. Man legt ein Bild dazu und beschreibt in normaler Sprache, was daran geändert werden soll. Für Layoutvorgaben und exakte Posen bleibt ControlNet weiterhin die genauere Methode, für den Alltag der meisten Unternehmen – Produkt freistellen, Hintergrund tauschen, Farbvariante erzeugen – ist der Referenzbild-Weg deutlich einfacher geworden.

Angelerntes Wissen: LoRA, Fine-Tuning, Embeddings

Diese drei greifen nicht in den Ablauf ein, sondern verändern die Bauteile selbst. Ein LoRA setzt kleine Zusatzmatrizen auf die Gewichte des Denoisers – wenige Megabyte, in Stunden trainiert, jederzeit zu- und abschaltbar. Ein vollständiges Fine-Tuning verändert die Gewichte dauerhaft und erzeugt einen neuen Checkpoint; bei Modellen mit dreißig Milliarden Parametern ist das ein Projekt mit ernsthaftem Hardwarebedarf und kein Nachmittagsausflug mehr. Textuelle Inversion – ein antrainierter Vektor für ein künstliches Wort im Text-Encoder – war zu SD-1.5-Zeiten populär und spielt bei den heutigen Architekturen kaum noch eine Rolle, weil die Encoder zu groß und zu sprachfähig geworden sind.

Hochskalieren

Ein KI-Upscaler ist in den meisten Fällen ein weiterer Diffusionslauf mit niedriger Denoising-Stärke auf einem vergrößerten Bild. Das Modell füllt die durch die Vergrößerung entstandene Unschärfe mit plausiblen Details. Wichtig ist das Wort plausibel: Diese Details standen nicht im Original. Für Druck und Marketing ist das unerheblich, für Dokumentation und Beweisführung ist es disqualifizierend.

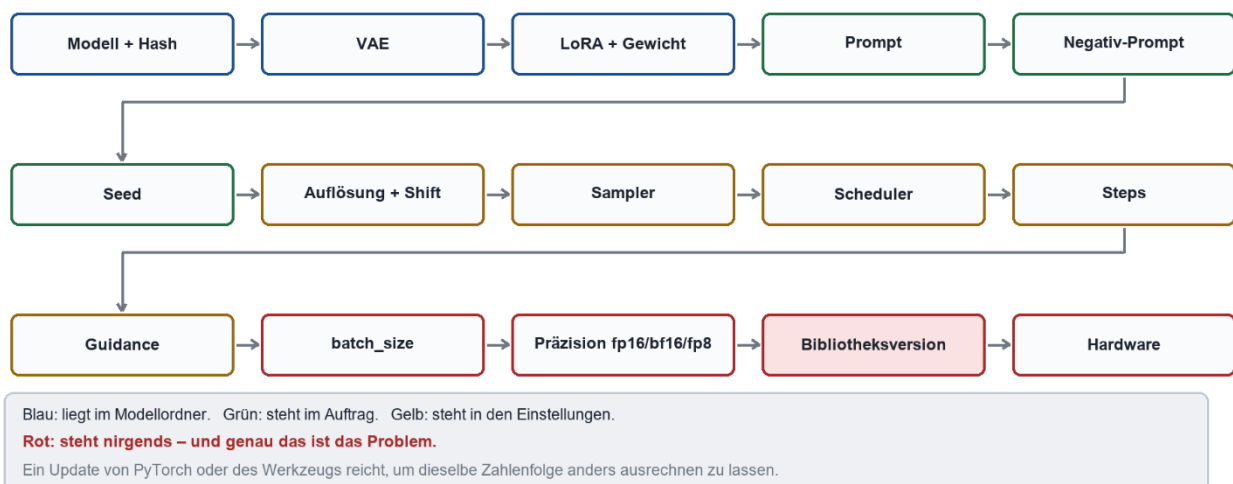
Verfahren	Was am Ablauf anders ist	Typischer Einsatz
Text zu Bild	Start bei reinem Rauschen	Neues Motiv ohne Vorlage
Bild zu Bild	Start bei teilweisem Rauschen	Stilwechsel, Variante, Verfeinerung
Inpainting	Maske setzt Außenbereich zurück	Objekt entfernen oder ersetzen
Outpainting	Leinwand vergrößert, Neues ist Maske	Format ändern, Szene erweitern
ControlNet	Zweite Bedingung pro Schritt	Pose, Kante, Tiefe, Layout
Referenzbild-Editing	Bild als Teil der Eingabe	Produkt tauschen, Szene anpassen
LoRA	Gewichte verschoben	Hausstil, Produkt, Person
Fine-Tuning	Gewichte dauerhaft geändert	Eigener Checkpoint für ein Thema
Hochskalieren	Zweiter Lauf, niedrige Stärke	Druckauflösung, Großformat

Die Kette der Reproduzierbarkeit

Bis hierher war das Grundlagenwissen. Jetzt kommt der Teil, der in Unternehmen über Erfolg und Ärger entscheidet. Denn aus allem, was bisher beschrieben wurde, folgt eine unangenehme Konsequenz: Ein Bild ist nicht das Ergebnis eines Prompts. Es ist das Ergebnis einer Kette, und jedes Glied dieser Kette geht in das Ergebnis ein.

Die Kette der Reproduzierbarkeit

Ändert sich ein Glied, ändert sich das Bild. Die Reihenfolge ist die des Ablaufs.



Fünfzehn Glieder zwischen Auftrag und Bild. Die roten stehen in keiner Oberfläche und in keiner Bilddatei.

Ich habe das bei einem Maschinenbauer erlebt, der seine Katalogbilder auf eigener Hardware erzeugte. Alles war sauber dokumentiert: Prompt, Seed, Schritte, Sampler, Guidance, alles in einer Tabelle. Nach einem Wartungsfenster, in dem die Arbeitsstationen turnusmäßig aktualisiert wurden, kam bei identischen Eingaben ein sichtbar anderes Bild heraus. Die Suche dauerte anderthalb Tage. Am Ende war es eine neue Version der zugrundeliegenden Rechenbibliothek, die dieselbe Fließkommaoperation minimal anders ausführte. Im ersten Schritt ein Unterschied in der fünften Nachkommastelle, nach achtundzwanzig Schritten ein anderes Gesicht.

Das ist kein Bug. Das ist die Natur eines iterativen Prozesses, bei dem jeder Schritt auf dem Ergebnis des vorherigen aufsetzt. Winzige Abweichungen wachsen sich aus. Die Konsequenz für die Praxis ist unbequem, aber klar: Wer Bilder reproduzieren können muss, muss die gesamte Umgebung festhalten – nicht nur die Einstellungen.

Kettenglied	Wo es steht	Was passiert, wenn es sich ändert
Modelldatei	Modellordner	Völlig anderes Bild
Modell-Prüfsumme	Nirgends, muss erhoben werden	Gleicher Dateiname, anderer Inhalt – der stille Killer
VAE	Modellordner oder Checkpoint	Farbe und Feindetail kippen
LoRA und Gewicht	Workflow	Stil verschiebt sich
Prompt und Negativ-Prompt	Auftrag	Anderer Inhalt
Seed	Auftrag	Anderes Bild, gleiche Qualität
Auflösung und Shift	Einstellungen	Anderer Bildaufbau
Sampler und Scheduler	Einstellungen	Anderer Pfad, anderes Bild
Schrittzahl	Einstellungen	Mehr oder weniger Feinzeichnung
Guidance	Einstellungen	Andere Bindung an den Text
batch_size	Einstellungen	Kann das Startrauschen verschieben
Zahlenpräzision	Startparameter des Werkzeugs	Abweichungen, die sich aufschaukeln
Bibliotheksversion	Nirgends	Dasselbe anders gerechnet
Hardware	Nirgends	Andere Kerne, andere Rundung

Was daraus organisatorisch folgt

Die Erkenntnis ist nicht, dass Reproduzierbarkeit unmöglich wäre. Sie ist herstellbar – aber nicht dadurch, dass man sich Zahlen notiert, sondern dadurch, dass man die Umgebung einfriert und sie zusammen mit dem Bild dokumentiert. In der Praxis heißt das: feste Modellversionen mit Prüfsumme, ein definierter Softwarestand, versionierte Workflows und ein Katalog außerhalb der Bilddateien. Metadaten in der Datei sind schön, überleben aber weder eine Konvertierung nach JPEG noch einen Upload in ein Social-Media-Netz.

In meinen eigenen Projekten liegt dieser Katalog in UB.Dossier: Wer eine Modellversion, einen Prompt-Baustein oder einen Standardwert ändert, ändert einen Datensatz – mit Nachweis, wer wann was geändert hat. Die Produktionsseite läuft über UB.Forge: Dort liegen die Workflows, dort wird ein Ablauf geändert, und dort greift vor dem Lauf ein Plausibilitäts-Check, der die üblichen Verdächtigen abfängt – destilliertes Modell mit CFG über 1, LoRA ohne passendes Basismodell, Auflösung jenseits der Trainingsgröße. Warum ich die Werkzeuge, über die ich berate, selbst im Tagesgeschäft benutze, steht in [Dogfooding: Ich setze ein, was ich berate](#).

Das Mindestmaß an Protokoll

Halte je erzeugtem Bild fest: Modellname und Prüfsumme, VAE, LoRAs mit Gewicht, Prompt und Negativ-Prompt im Wortlaut, Seed, Sampler, Scheduler, Schrittzahl, Guidance, Auflösung,

Stapelgröße, Werkzeug samt Version.

Speichere das außerhalb der Bilddatei. Eine Bilddatei verliert ihre Metadaten beim ersten Umwandeln, beim Beschneiden und bei jedem Upload.

Wenn du nur eine Sache umsetzen kannst: die Prüfsumme des Modells. Dateinamen sind Wunschdenken – dieselbe Datei kursiert in Varianten, und der Unterschied fällt erst auf, wenn ein Nachlauf nicht passt.

Cloud oder eigene Hardware – die Reproduzierbarkeit entscheidet mit

Bei Cloud-Diensten liegt ein Teil dieser Kette außerhalb deiner Reichweite. Du kennst die Modellversion nicht immer, du kennst den Softwarestand nicht, und bei mehreren großen Anbietern gibt es an den Bild-Endpunkten schlicht keinen Seed-Parameter mehr. Wenn der Anbieter sein Modell aktualisiert, ändert sich dein Bildstil – im laufenden Betrieb, ohne Ankündigung. Für schnelle Entwürfe ist das egal, für eine Katalogserie, die über drei Jahre einheitlich aussehen soll, ist es das Gegenteil von egal.

Der eigene Betrieb dreht das um: volle Kontrolle über jedes Kettenglied, dafür die Verantwortung für Hardware, Updates und Betrieb. Die Abwägung im Detail steht in [Cloud vs. lokale KI-Bildgenerierung](#); die strategische Sicht auf die Frage entwickelt [KI-Bildgenerierung im Unternehmen – Die strategische Alternative zur Cloud](#). Wie eine belastbare Produktionsumgebung technisch aussieht, zeigt [Aufbau einer skalierbaren Bild-Maschine mit ComfyUI und Flux](#), und wie die fertigen Bilder samt Nachweis abgelegt werden, beschreibt [Digital Asset Management für KI-Bilder mit SharePoint Online und Power Platform](#).

Und die rechtliche Seite?

Das Protokoll, das du für die Reproduzierbarkeit ohnehin brauchst, ist gleichzeitig die halbe Antwort auf die Compliance-Frage. Seit dem 2. August 2026 gelten die Transparenzpflichten des EU AI Act: Wer künstlich erzeugte Bilder in Umlauf bringt, die reale Szenen oder Personen vortäuschen, muss sie erkennbar machen. Was das konkret bedeutet, steht in [EU AI Act: Was seit dem 2. August 2026 gilt](#). Die Fragen nach Urheberrecht, Modelllizenzen, Marken im Bild und Persönlichkeitsrechten gehören nicht in einen Grundlagentext über Diffusion, sondern sind im Überblick unter [KI-Bildgenerierung](#) eingeordnet. Wie man ein Bild als KI-Erzeugnis erkennt und welche technischen Marker es dafür gibt, klärt [KI-generierte Bilder erkennen](#).

Wenn es darum geht, das Ganze im Unternehmen aufzusetzen – von der Modellauswahl über die Kette bis zum Freigabeprozess – ist das genau der Punkt, an dem ich einsteige: [KI-Bildgenerierung Beratung](#). Wenn dein Team die Regler selbst sicher bedienen soll, passt eher [KI-Bildgenerierung Schulung](#). Für den Fall, dass die Bilder markenkonform sein müssen, ist [Markenkonsistente KI-generierte Bilder](#) der richtige Einstieg.

Wo das Modell an Grenzen stößt

Ein Grundlagentext, der nur erklärt, wie gut alles funktioniert, ist Werbung. Deshalb zum Schluss der ehrliche Teil: Aus dem beschriebenen Funktionsprinzip folgen Schwächen, die keine Softwareversion wegpacht.

Das Modell versteht die Szene nicht

Es gibt in einem Bildmodell kein Modell der Welt. Kein Konzept von „eine Hand hat fünf Finger“, kein Konzept von „Schatten fallen in eine Richtung“, kein Konzept von „dieses Kabel muss irgendwo hinführen“. Was es gibt, ist eine sehr gute Statistik darüber, wie Bildbereiche typischerweise aussehen. Das reicht erstaunlich weit und bricht genau dort zusammen, wo lokale Plausibilität und globale Konsistenz auseinanderfallen.

Hände sind das bekannteste Beispiel: lokal sieht jeder Finger richtig aus, global sind es sechs. Warum das so hartnäckig ist und was tatsächlich hilft, steht in [Warum KI keine Hände kann](#).

Zählen und Anordnen bleiben schwierig

„Drei Personen“ heißt für das Modell nicht drei, sondern „so viele Personen, wie auf Bildern mit dieser Bildunterschrift üblicherweise zu sehen sind“. Bei kleinen Zahlen klappt das inzwischen recht gut, bei sieben oder zwölf wird es zum Glücksspiel. Räumliche Beziehungen – links, rechts, hinter, darauf – haben sich mit den sprachfähigeren Encodern deutlich verbessert, bleiben aber die erste Stelle, an der ein komplexer Prompt auseinanderfällt.

Nicht jeder Bildgenerator ist ein Diffusionsmodell

Der letzte Punkt ist der wichtigste, wenn du zwischen Werkzeugen wechselst. Alles in diesem Beitrag beschreibt Diffusions- und Flow-Matching-Modelle. Das trifft auf FLUX, Stable Diffusion, Qwen-Image und die meisten offenen Modelle zu. Es trifft nicht auf OpenAIs GPT-Image-Reihe zu, die die alten DALL·E-Modelle abgelöst hat: Diese erzeugt Bilder autoregressiv, also Bildstück für Bildstück, so wie ein Sprachmodell Wort für Wort schreibt. Daher stammt die gute Textwiedergabe im Bild – Schrift wird als Zeichenfolge vorhergesagt und nicht als Textur gemalt. Daher stammt aber auch, dass es dort keinen Seed, keine Schrittzahl und kein CFG gibt. Die mentalen Modelle aus der Diffusionswelt lassen sich auf diese Systeme nicht übertragen.

Das ist ein gutes Beispiel für den Hausgebrauch: Man kann keine Einstellung übernehmen, bevor man geprüft hat, ob das Zielsystem überhaupt dieselbe Bauart hat. Wie die Anbieterlandschaft derzeit aussieht, steht in [Übersicht der wichtigsten KI-Bildgeneratoren](#); wie FLUX.2 in eine Microsoft-Umgebung kommt, zeigt [Black Forest FLUX.2 in PowerPoint und Copilot](#). Wenn Sprachmodelle Teil der Werkzeugkette werden – etwa für Prompt-Aufbereitung oder Prüfschritte – ist [Claude im Unternehmen](#) der passende Einstieg.

Häufige Fragen

Speichert ein KI-Modell die Trainingsbilder?

Nein. Ein Checkpoint enthält Gewichte, keine Bilddateien. Die frühen Stable-Diffusion-Versionen wurden auf dem LAION-5B-Datensatz mit rund 5,85 Milliarden Bild-Text-Paaren trainiert – die passen schon rein rechnerisch nicht in ein paar Gigabyte. Was das Modell speichert, ist die Statistik darüber, wie Bilder aussehen. Bei extrem häufigen und gleichförmigen Motiven kann diese Statistik allerdings so scharf sein, dass das Ergebnis dem Original zum Verwechseln ähnelt. Deshalb bleibt die Prüfung auf fremde Marken und erkennbare Personen Pflicht.

Was ist der Unterschied zwischen Diffusion und Flow Matching?

Beide bringen Rauschen und Bild in Verbindung, nur auf unterschiedlichen Wegen. Klassische Diffusion lernt, das Rauschen auf einem Feld zu schätzen, und braucht dafür eine krumme Bahn mit vielen kleinen Schritten. Flow Matching spannt eine möglichst gerade Linie zwischen Rauschen und Bild und lernt die Bewegungsrichtung entlang dieser Linie. Das Ergebnis: weniger Schritte bei gleicher Qualität. Alle aktuellen großen Modelle arbeiten so, der Sammelbegriff Diffusion ist trotzdem geblieben.

Wie viele Schritte braucht ein Bild wirklich?

Das hängt vom Modell ab, und die alte Faustregel von 20 bis 50 Schritten gilt nur noch für die ältere Generation. Moderne Flow-Matching-Modelle liegen typischerweise bei 20 bis 30. Destillierte Varianten sind auf vier bis acht Schritte optimiert und werden bei mehr Schritten nicht besser, sondern schlechter. Die Sättigungsgrenze für dein Modell findest du empirisch: gleicher Seed, Schrittzahl schrittweise erhöhen, vergleichen. Details in [Sampling-Steps erklärt](#).

Warum sieht dasselbe Bild bei mir anders aus als beim Kollegen?

Weil mindestens ein Glied der Kette abweicht. Die häufigsten Kandidaten in dieser Reihenfolge: ein anderer VAE, eine andere Modelldatei bei gleichem Dateinamen, ein anderer Sampler, eine andere Version des Werkzeugs. Wenn Prompt, Seed und alle Einstellungen nachweislich identisch sind und das Bild trotzdem abweicht, liegt es an Software oder Hardware – und dann hilft nur, die Umgebung anzugleichen.

Was genau ist der latente Raum – in einem Satz?

Eine kompakte Zwischendarstellung des Bildes, in der das Modell rechnet: bei den gängigen Modellen ein Achtel der Kantenlänge, dafür mit 4, 16 oder 32 Kanälen statt der drei Farbkanäle – klein genug, um bezahlbar zu bleiben, reich genug, um das Wesentliche zu tragen.

Wofür ist der VAE zuständig und woran merke ich, dass er falsch ist?

Der VAE übersetzt zwischen Bildpunkten und Latentraum: hinein beim Bild-zu-Bild, hinaus am Ende jedes Laufs. Ein falscher VAE erkennt man daran, dass der Bildaufbau vollkommen stimmt, aber die Farben nicht: ausgewaschen, grauschleierig oder mit violettem Stich. Wenn der Bildinhalt stimmt und nur die Anmutung falsch ist, schau zuerst hier nach, nicht beim Prompt. Mehr dazu in [Checkpoints, Modelle und VAE](#).

Braucht man eine teure Grafikkarte?

Für die ältere Generation reichen 8 bis 12 Gigabyte Videospeicher problemlos. Für die aktuellen großen Modelle wird es enger: FLUX.2 in der dev-Variante läuft quantisiert ab etwa 13 Gigabyte, ohne Quantisierung deutlich darüber. Wer knapp bei Kasse ist, nimmt die kleineren Varianten – FLUX.2 klein gibt es seit Januar 2026 mit 4 und 9 Milliarden Parametern, jeweils als Basis- und als Vier-Schritt-Version. Wichtiger als die Rechenleistung ist fast immer der Videospeicher: Was nicht hineinpasst, läuft gar nicht oder quälend langsam.

Wie lange dauert ein Bild?

Auf aktueller Hardware und mit einem destillierten Modell: wenige Sekunden. Mit einem großen Modell bei dreißig Schritten und 1024 mal 1024 Punkten: je nach Karte zwischen zehn Sekunden und gut einer Minute. Cloud-Dienste liegen meist im einstelligen Sekundenbereich, weil dort Hardware steht, die kein Mittelständler unter den Schreibtisch stellt. Beim Hochskalieren und bei Video vervielfacht sich die Zeit entsprechend.

Arbeiten alle Bildgeneratoren nach diesem Prinzip?

Nein. FLUX, Stable Diffusion, Qwen-Image und die meisten offenen Modelle ja. OpenAIs GPT-Image-Modelle, die seit 2026 die DALL·E-Reihe vollständig abgelöst haben, arbeiten autoregressiv: Sie erzeugen das Bild stückweise wie ein Sprachmodell einen Text. Das erklärt die gute Schriftdarstellung und das Fehlen von Seed, Schrittzahl und CFG. Wer zwischen den Welten wechselt, kann Einstellungen nicht übernehmen.

Lernt das Modell aus den Bildern, die ich erzeuge?

Der Generierungsvorgang selbst verändert das Modell nicht – ein Checkpoint ist während der Nutzung unveränderlich. Die relevante Frage ist eine andere: Was macht der Anbieter mit deinen Prompts und Bildern? Bei Cloud-Diensten steht die Antwort in den Nutzungsbedingungen, und sie fällt je nach Tarif unterschiedlich aus. Bei lokalem Betrieb stellt sich die Frage nicht, weil nichts das Haus verlässt.

Warum kann die KI keine Hände und keinen lesbaren Text?

Weil das Modell lokal plausibel arbeitet und keine globale Vorstellung von der Szene hat. Bei Text ist das Problem inzwischen weitgehend gelöst – die sprachfähigen Encoder und die Transformer-

Architektur haben da einen Qualitätssprung gebracht, und autoregressive Modelle sind darin ohnehin stark. Bei Händen hält sich das Problem hartnäckiger. Die Ursachen und die Gegenmaßnahmen stehen in [Warum KI keine Hände kann](#).

Ersetzt die KI unsere Grafiker?

Nach allem, was in diesem Beitrag steht: Was du bekommst, ist eine Maschine, die statistisch plausible Bilder erzeugt und dabei weder weiß, was ein Kunde will, noch was eine Marke ausmacht, noch warum ein Bild in einem bestimmten Kontext falsch wirkt. Was sich verändert, ist die Arbeitsteilung: Varianten, Hintergründe, Formatanpassungen und Bildserien entstehen um Größenordnungen schneller. Die Entscheidung, welches Bild richtig ist, und die Verantwortung dafür, dass es keine fremden Marken und keine erkennbaren Personen enthält, bleibt bei Menschen. In der Praxis verschiebt sich die Rolle vom Erzeugen zum Kuratieren und Prüfen – und der Prüfteil wird eher mehr als weniger.

Was bringt mir das Verständnis der Diffusion praktisch?

Es verwandelt Fehlersuche in Diagnose. Wenn du weißt, dass der Prompt am Text-Encoder hängt, der Bildaufbau am Startrauschen, die Feinzeichnung an Schritten und Sampler und die Farbe am VAE, dann weißt du bei jedem Problem, an welchem Regler du drehen musst – statt zwanzig Varianten zu erzeugen und zu hoffen. Und du weißt, welche Angaben du aufschreiben musst, damit ein gelungenes Bild in einem halben Jahr noch einmal in derselben Form entsteht.

Fazit

KI-Bildgenerierung ist keine Magie und keine Bildersuche. Sie ist ein trainierter Aufräumvorgang: Aus einem Rauschfeld, gezogen aus einer einzigen Zahl, arbeitet sich ein Modell in mehreren Durchgängen zu einem Bild vor, geleitet von der Bedeutung, die ein Text-Encoder aus deinem Prompt gemacht hat, gerechnet in einem komprimierten Zwischenraum, übersetzt von einem Decoder in Bildpunkte. Vier Bauteile, ein Kreislauf, ein Ergebnis.

Wer das verstanden hat, hat zwei Dinge gewonnen. Erstens einen Diagnosebaum: Jedes Problem lässt sich einer Station zuordnen, und die Zahl der sinnlosen Versuche sinkt dramatisch. Zweitens ein realistisches Bild von den Grenzen – das Modell hat kein Weltwissen, es kann nicht zählen, und es arbeitet lokal plausibel statt global konsistent.

Und dann ist da noch die Sache mit der Kette. Reproduzierbarkeit ist keine Eigenschaft des Seeds, sondern das Ergebnis einer Umgebung, in der fünfzehn Glieder festgehalten sind. Drei davon stehen in keiner Oberfläche: Zahlenpräzision, Bibliotheksversion, Hardware. Wer Bilder in Serie produziert und sie in einem Jahr noch einmal braucht, muss diese drei mit dokumentieren. Das ist der unspektakulärste Teil der ganzen Technologie – und derjenige, der am Ende über die Betriebskosten entscheidet.

Weiterlesen

- › Alles Übrige zum Thema im Überblick: [KI-Bildgenerierung](#)
- › Die Regler im Einzelnen: [Der Seed bei der KI-Bildgenerierung](#), [Sampling-Steps erklärt](#), [Sampler & Scheduler erklärt](#) und [CFG-Scale: Guidance in KI-Bildgeneratoren](#)
- › Was in den Prompt gehört und was nicht: [Prompt und Negativ-Prompt in der KI-Bildgenerierung](#) sowie [Warum die KI den Prompt ignoriert](#)
- › Modelle, VAE und angelerntes Wissen: [Checkpoints, Modelle und VAE](#) und [Was ist ein LoRA?](#)
- › Format und Größe: [Auflösung & Seitenverhältnis bei KI-Bildern](#), Stapelverarbeitung in [batch_size](#) und [Reproduzierbarkeit](#)
- › Wo die Technik an Grenzen stößt: [Warum KI keine Hände kann](#) und [KI-generierte Bilder erkennen](#)

- › Betriebsmodell und Werkzeugauswahl: [Cloud vs. lokale KI-Bildgenerierung](#), [Die 15 wichtigsten Modelle zur lokalen Bildgenerierung](#) und [Übersicht der wichtigsten KI-Bildgeneratoren](#)
- › Von der Einzelgenerierung zur Produktion: [Aufbau einer skalierbaren Bild-Maschine mit ComfyUI und Flux](#), [KI-Bildgenerierung im Unternehmen – Die strategische Alternative zur Cloud](#) und [Digital Asset Management für KI-Bilder mit SharePoint Online und Power Platform](#)
- › Marke, Kennzeichnung und Recht: [Markenkonsistente KI-generierte Bilder](#), [EU AI Act: Was seit dem 2. August 2026 gilt](#)
- › Unterstützung im Haus: [KI-Bildgenerierung Beratung](#) und [KI-Bildgenerierung Schulung](#)