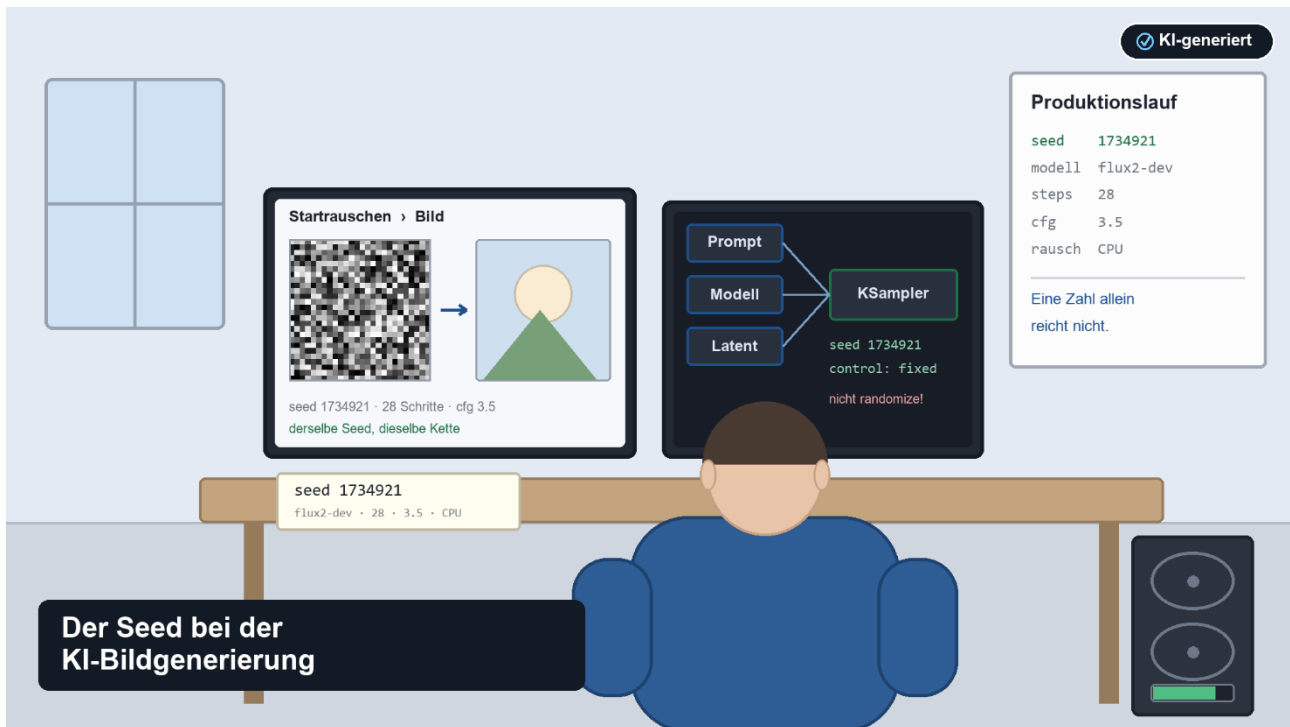


Der Seed bei der KI-Bildgenerierung



Wissen	Beratung	Schulung
Alle Grundlagen zur KI-Bildgenerierung an einer Stelle – vom Diffusionsprinzip bis zur Governance.	Von der Testinstallation zur belastbaren Bildproduktion mit Nachweis und klaren Zuständigkeiten.	Dein Team lernt die Regler zu bedienen – und zu protokollieren, was es eingestellt hat.
KI-Bildgenerierung	KI-Bildgenerierung Beratung	KI-Bildgenerierung Schulung

Es gibt einen Moment, den jeder kennt, der länger als eine Woche mit Bildmodellen arbeitet. Ein Bild sitzt. Nicht „ganz nett“, sondern genau richtig: die Stimmung, der Bildausschnitt, das Licht. Du machst zufrieden weiter, erzeugst noch zwanzig andere Motive, schließt abends den Rechner. Zwei Tage später kommt die Rückmeldung: genau dieses Bild, bitte, nur im Hochformat und mit etwas mehr Platz oben rechts für den Störer. Du öffnest das Werkzeug, trägst denselben Prompt ein, drückst auf Start – und bekommst etwas völlig anderes.

Willkommen bei der Zahl, um die es in diesem Beitrag geht. Der Seed ist bei der KI-Bildgenerierung die unscheinbarste Einstellung der gesamten Oberfläche und gleichzeitig diejenige, deren Verlust am meisten Geld kostet. Sie entscheidet nicht darüber, wie gut ein Bild wird. Sie entscheidet darüber, ob du es je wiederfindest.

Ich habe das bei einem Handelsunternehmen erlebt, das seine Katalogbilder auf eigener Hardware erzeugt hat. Das Team arbeitete sauber, nur eben mit der Standardeinstellung: In ComfyUI steht der Seed-Knoten nach der Installation auf „randomize“. Nach jedem Lauf steht im Feld also schon die nächste Zufallszahl, nicht die, aus der das gerade sichtbare Bild entstanden ist. Das gute Motiv war erzeugt, gespeichert, für den Druck nach JPEG konvertiert, und die PNG-Datei mit den Metadaten war beim Aufräumen der Arbeitsplatte mitgegangen. Ergebnis: drei Tage Nacharbeit für

ein Bild, das schon einmal fertig war. Der Kollege, der es erzeugt hatte, war zu dem Zeitpunkt im Urlaub. Der Seed auch.

Dieser Beitrag klärt, was der Seed technisch ist, warum derselbe Seed bei identischer Kette dasselbe Bild ergibt – und warum die Betonung auf „identische Kette“ liegt. Er zeigt, wo die Zahl im Alltag verloren geht, welche Anbieter sie 2026 überhaupt noch herausrücken, und welche Rolle sie im Nachweis spielt, seit Bilder aus generativen Systemen gekennzeichnet werden müssen. Wenn dir das Fundament dahinter fehlt – wie Diffusion überhaupt funktioniert – ist [Basiswissen KI-Bildgenerierung: Wie Diffusion funktioniert](#) der passende Einstieg; den großen Überblick über das gesamte Themenfeld findest du unter [KI-Bildgenerierung](#).

Was du aus diesem Beitrag mitnimmst

Der Seed ist der Startwert eines Pseudozufallsgenerators. Aus ihm wird das Rauschfeld gezogen, mit dem jede Generierung beginnt. Er ist nicht das Bild, sondern der Ausgangspunkt.

Derselbe Seed ergibt dasselbe Bild – aber nur bei identischer Kette. Dazu gehören Modell, Sampler, Schrittzahl, Guidance, Auflösung, Stapelgröße, Rauschquelle, Zahlenpräzision, Bibliotheksversion und Hardware.

Es gibt keine guten und schlechten Seeds. Seed 7 ist nicht schlechter als Seed 3.948.221.007, und Seed plus eins liefert kein ähnliches Bild, sondern ein völlig anderes.

Die großen Cloud-Anbieter geben teilweise gar keinen Seed mehr heraus. Bei OpenAI, Google und xAI existiert an den Bild-Endpunkten im September 2026 kein Seed-Parameter. Dort tritt die festgepinnte Modellversion an seine Stelle.

Der Seed allein ist kein Nachweis. Er wird erst brauchbar, wenn er zusammen mit Modellversion, Prüfsumme, Einstellungen und Software-Stand in einem Katalog außerhalb der Bilddatei steht.

Was ein Seed wirklich ist

Der Name ist ausnahmsweise ehrlich. „Seed“ heißt Saatkorn, und genau so verhält sich die Zahl: Aus demselben Korn wächst unter gleichen Bedingungen dieselbe Pflanze. Was die meisten Erklärungen unterschlagen, ist der zweite Teil des Satzes – die gleichen Bedingungen. Darum geht es im übernächsten Kapitel ausführlich.

Vom Startwert zum Rauschfeld

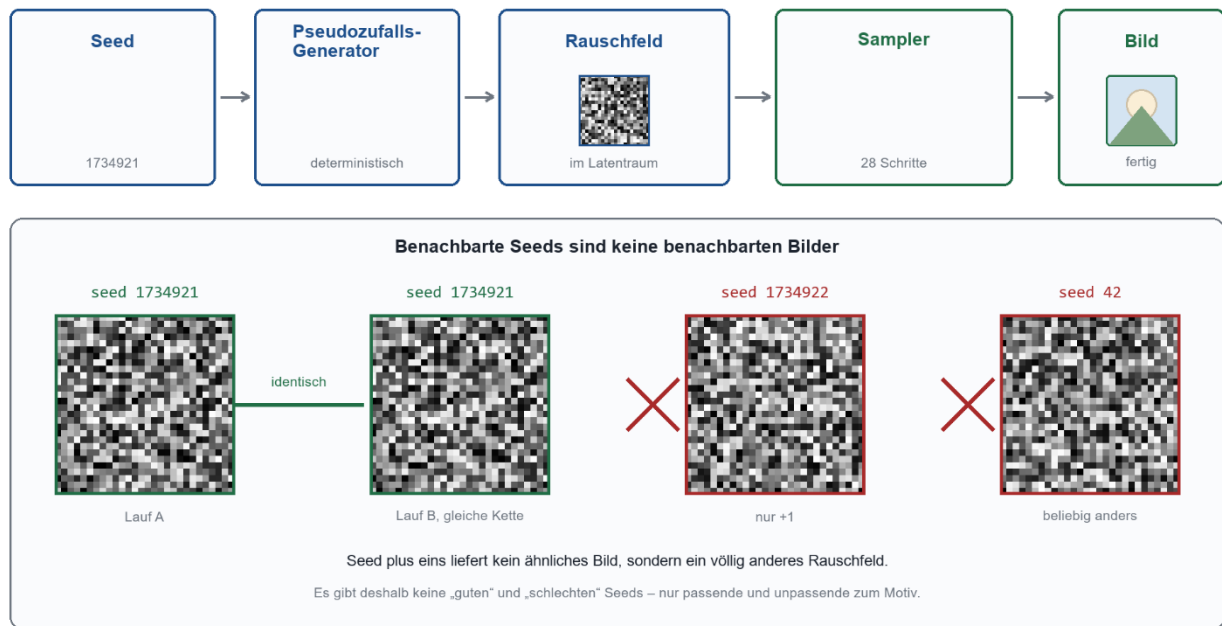
Eine Bildgenerierung fängt nicht mit einem leeren Blatt an, sondern mit Rauschen. Nicht mit Rauschen im Pixelbild, wie man es vom analogen Fernseher ohne Empfang kennt, sondern mit einem Rauschfeld im Latentraum – in der komprimierten Darstellung, in der das Modell rechnet. Ein Bild mit 1024 mal 1024 Bildpunkten entspricht dort einem deutlich kleineren Raster mit mehreren Kanälen. Aus diesem Feld entfernt das Modell Schritt für Schritt Rauschen, bis am Ende ein Motiv übrig bleibt, und ganz zum Schluss übersetzt der Decoder das Ergebnis zurück in Bildpunkte.

Dieses Rauschfeld muss irgendwo herkommen. Computer können nicht würfeln; sie rechnen. Was sie stattdessen benutzen, ist ein Pseudozufallsgenerator: ein Verfahren, das aus einem Startwert eine Zahlenfolge erzeugt, die statistisch wie Zufall aussieht, in Wahrheit aber vollständig festgelegt ist. Gibst du demselben Generator denselben Startwert, bekommst du dieselbe Folge. Jedes Mal, bis in die letzte Stelle. Dieser Startwert ist der Seed.

Der Seed erzeugt also nicht das Bild, sondern das Startrauschen. Was daraus wird, rechnet das Modell in einer festgelegten Zahl von Schritten aus – nachzulesen in [Sampling-Steps erklärt](#) – entlang eines Verfahrens, das der Sampler bestimmt, beschrieben in [Sampler & Scheduler erklärt](#). Der Seed ist der Anfang der Rechnung, nicht ihr Ergebnis.

Vom Seed zum Startrauschen zum Bild

Der Seed ist nicht das Bild – er ist der Startwert, aus dem das Rauschen gezogen wird.



Gleiche Zahl, gleiches Rauschen. Ob daraus auch dasselbe Bild wird, entscheidet der Rest der Kette.

Skizze 1: Aus der Zahl wird über den Generator ein Rauschfeld, daraus über den Sampler ein Bild. Benachbarte Seeds liefern keine benachbarten Bilder, sondern völlig verschiedene Rauschfelder.

Wie groß darf die Zahl sein?

Technisch ist der Seed eine ganze Zahl ohne Vorzeichen, und der zulässige Bereich hängt am Werkzeug. Die klassische Grenze ist der 32-Bit-Wert: null bis 4.294.967.295, also gut vier Milliarden mögliche Startpunkte. Midjourney arbeitet bis heute genau in diesem Bereich, die Stable-Image-Schnittstelle von Stability endet eine Zahl darunter, und Ideogram hat in der Version 3.0 bei 2.147.483.647 gedeckelt – dem größten Wert, den eine vorzeichenbehaftete 32-Bit-Zahl fassen kann.

ComfyUI geht deutlich weiter und nimmt den vollen vorzeichenlosen 64-Bit-Bereich an, also bis 18.446.744.073.709.551.615. Das ist keine Angeberei, sondern Bequemlichkeit: Der Wert wird ohnehin nur als Startwert durchgereicht, und ein größerer Raum macht zufällige Kollisionen noch unwahrscheinlicher als sie ohnehin schon sind. Praktisch spielt das keine Rolle. Schon die vier Milliarden reichen, dass du in einem Berufsleben nie zufällig zweimal denselben Seed ziehst.

Wichtiger als die Obergrenze ist die Konvention für den Zufall, und die ist leider nicht einheitlich. In der WebUI-Familie steht eine Minus-Eins im Feld für „würfle mir bei jedem Lauf einen neuen Wert“. Bei Stability bedeutet eine Null dasselbe, und das Weglassen des Parameters ebenfalls. In ComfyUI gibt es gar kein Sonderzeichen, sondern einen eigenen Schalter neben dem Feld. Wer die Konventionen verwechselt, erzeugt ungewollt mit einem festen Seed – oder ungewollt ohne.

Umgebung	Wertebereich	Zufall bedeutet dort	Wo die benutzte Zahl steht
ComfyUI	0 bis 18.446.744.073.709.551.615	Schalter „control_after_generate“ auf randomize	im eingebetteten Workflow der PNG-Datei und in der Lauf-Historie
WebUI-Familie (Automatic1111, reForge)	0 bis 4.294.967.295	eine -1 im Seed-Feld	im Parameterblock der PNG-Datei und unter dem Bild
Midjourney	0 bis 4.294.967.295	kein Seed angegeben	auf Nachfrage über die

			Oberfläche abrufbar
Stability-Schnittstelle	0 bis 4.294.967.294	Null oder Parameter weglassen	wird in der Antwort zurückgeliefert
Ideogram 3.0	0 bis 2.147.483.647	kein Seed angegeben	wird in der Antwort zurückgeliefert
FLUX-Schnittstelle von Black Forest Labs	beliebige ganze Zahl	kein Seed angegeben	kein eigenes Antwortfeld – du musst ihn selbst mitschreiben

Warum es keine guten und schlechten Seeds gibt

Es hält sich hartnäckig die Vorstellung, manche Seeds seien besser als andere – Lieblingszahlen, die angeblich schärfere Gesichter oder bessere Kompositionen liefern. Das ist Kaffeesatzleserei. Der Zahlenwert selbst trägt keine Information über Bildqualität; er ist nur der Einsprungpunkt in eine Zahlenfolge. Seed 7 ist nicht „kleiner“ oder „schwächer“ als Seed 3.948.221.007, und ein Seed, der für ein Produktfoto großartig funktioniert, kann beim nächsten Motiv völlig unbrauchbar sein.

Was es dagegen wirklich gibt: Rauschfelder, deren grobe Struktur zufällig gut zu einem bestimmten Bildaufbau passt. Ein Feld mit heller Masse in der oberen Bildhälfte begünstigt Himmel, eines mit starker Struktur in der Mitte begünstigt ein zentrales Objekt. Das ist aber keine Eigenschaft der Zahl, sondern ein Zusammenspiel aus Rauschfeld, Prompt und Modell. Wenn ein Motiv systematisch nicht funktioniert, liegt das fast nie am Seed – die häufigeren Ursachen stehen in [Warum die KI den Prompt ignoriert](#).

Der teuerste Denkfehler: Seed plus eins

Benachbarte Seeds sind keine benachbarten Bilder. Wer glaubt, mit Seed 1734922 eine leichte Variante von Seed 1734921 zu bekommen, irrt: Ein um eins veränderter Startwert erzeugt eine vollständig andere Zahlenfolge und damit ein vollständig anderes Rauschfeld. Das Ergebnis hat mit dem Ausgangsbild nichts zu tun.

Seed-Reihen wie 100, 101, 102 sind deshalb ein bequemer Weg, viele unabhängige Varianten zu erzeugen – aber kein Weg, ein bestimmtes Bild behutsam zu verändern. Dafür brauchst du andere Werkzeuge, dazu später mehr.

Derselbe Seed, dasselbe Bild – aber nur bei identischer Kette

Jetzt kommt der Satz, den alle kennen, und die Einschränkung, die kaum jemand mitspricht. Gleicher Seed plus gleicher Prompt plus gleiche Einstellungen ergibt dasselbe Bild. Das stimmt, und es ist kein netter Nebeneffekt, sondern schlichte Mathematik: Die Rechnung ist deterministisch. Sie enthält an keiner Stelle echten Zufall, nur die Pseudofolge aus dem Seed.

Der Haken steckt in „gleiche Einstellungen“. Damit sind nicht die fünf Regler gemeint, die du in der Oberfläche siehst, sondern jedes Glied einer ziemlich langen Kette. Die sichtbaren Glieder sind der harmlose Teil. Die unsichtbaren sind es, die den Wiederherstellungsversuch scheitern lassen.

Die sichtbaren Glieder

Was in der Oberfläche steht, gehört vollständig dazu: Prompt und Negativ-Prompt, beschrieben in [Prompt und Negativ-Prompt in der KI-Bildgenerierung](#), das Modell samt Decoder aus [Checkpoints, Modelle und VAE](#), jedes geladene Zusatzmodell mitsamt seiner Stärke aus [Was ist ein LoRA?](#), Sampler und Scheduler, die Schrittzahl, der Guidance-Wert aus [CFG-Scale: Guidance in KI-Bildgeneratoren](#) und die Bildmaße aus [Auflösung & Seitenverhältnis bei KI-Bildern](#). Ändert sich eines davon, ändert sich das Ergebnis – trotz identischem Seed.

Ein Glied gehört dazu, das viele übersehen: die Stapelgröße. Ob ein Bild allein oder als Teil eines Stapels gerechnet wird, verändert die Reihenfolge der Rechenoperationen und damit unter Umständen das Ergebnis. Zusätzlich vergeben die Werkzeuge den Seed im Stapel unterschiedlich: Die WebUI-Familie zählt hoch, ComfyUI zieht weiter aus demselben Rauschstrom und kennt nur Indexpositionen. Das ist wichtig genug für einen eigenen Beitrag, nämlich [batch size und Reproduzierbarkeit](#).

Die stillen Glieder – hier scheitern die Rekonstruktionen

Das interessanteste Glied steht in keiner Oberfläche prominent herum: die Frage, wo das Rauschen erzeugt wird. ComfyUI zieht es grundsätzlich auf dem Hauptprozessor, auch wenn anschließend die Grafikkarte rechnet. Der Grund ist genau dieser: Ein auf der CPU erzeugtes Rauschfeld ist über unterschiedliche Hardware hinweg viel besser reproduzierbar. Die WebUI-Familie erzeugt es standardmäßig auf der Grafikkarte und bietet in den Einstellungen die Wahl zwischen CPU, GPU und einer Variante, die das Verhalten von NVIDIA-Karten nachbildet.

Daraus folgt eine Regel, die viele Stunden Fehlersuche erspart: Derselbe Seed liefert in ComfyUI und in der WebUI-Familie niemals dasselbe Bild. Nicht ähnlich, nicht fast, sondern verschieden – weil das Startrauschen aus zwei verschiedenen Generatoren stammt. Wer eine Bildserie von einem Werkzeug ins andere umzieht, nimmt die Seeds nicht mit. Das ist keine Fehlfunktion, sondern eine Architekturentscheidung, und sie ist auf beiden Seiten dokumentiert.

Eine Ebene tiefer wird es noch unangenehmer. Ein Diffusionsmodell rechnet mit Fließkommazahlen begrenzter Genauigkeit, und Fließkomma-Addition ist nicht assoziativ: Die Reihenfolge, in der Teilergebnisse zusammengefasst werden, beeinflusst die letzten Stellen. Diese Reihenfolge hängt davon ab, welchen Rechenkern die Bibliothek für die konkrete Tensorform auswählt – und das hängt an Treiber, Bibliotheksversion und Kartengeneration. Die Dokumentation der gängigen Bibliotheken formuliert das erfrischend offen: Vollständig reproduzierbare Ergebnisse sind weder über verschiedene Versionen hinweg garantiert noch zwischen Prozessor und Grafikkarte, selbst bei identischem Seed.

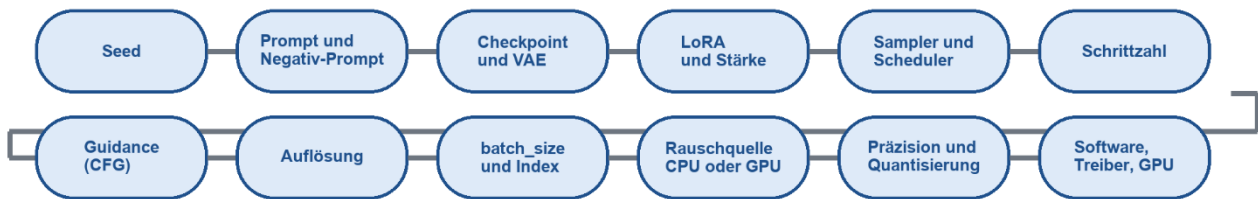
Diese Abweichung beginnt in der zwölften Nachkommastelle und bleibt meistens auch dort. Meistens. Ein Diffusionsprozess füttert das Ergebnis jedes Schritts in den nächsten, und dabei können sich winzige Unterschiede aufschaukeln. In der Praxis erlebst du beide Extreme: Oft sind zwei Läufe nicht unterscheidbar, gelegentlich dreht sich ein Kopf, ändert sich ein Schriftzug, oder ein Detail kippt. Dass ausgerechnet Hände dabei besonders gern auseinanderfallen, hat eigene Gründe, nachzulesen in [Warum KI keine Hände kann](#).

Der dritte stille Störer ist die Zahlenpräzision. Wer ein Modell quantisiert lädt – in reduzierter Genauigkeit, um es überhaupt in den Grafikspeicher zu bekommen – rechnet nicht mehr dasselbe wie mit der vollen Gewichtsdatei. Bei den FLUX-Modellen von Black Forest Labs ist das sogar ausdrücklich dokumentiert: Der Schnellmodus arbeitet mit 8-Bit-Gleitkommazahlen und bricht damit die Reproduzierbarkeit, auch bei festem Seed. Wer exakte Wiederholbarkeit braucht, schaltet ihn ab und bezahlt mit Rechenzeit.

Die Kette der Reproduzierbarkeit

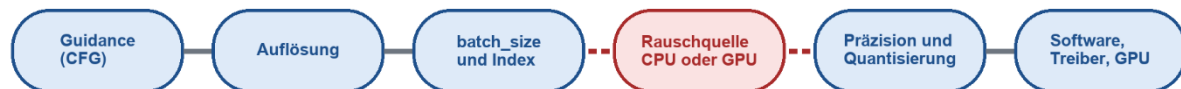
Der Seed ist ein Glied von vielen. Bricht eines, ist das Versprechen weg.

Lauf A – alles identisch



Kette geschlossen > derselbe Seed ergibt bitgenau dasselbe Bild

Lauf B – ein Glied verändert



Kette gebrochen > derselbe Seed ergibt ein anderes Bild – ohne Fehlermeldung

Die stillen Glieder sind die gefährlichen

Rauschquelle, Präzision, Treiber und Bibliotheksversion stehen in keinem Prompt – und ändern sich beim nächsten Update von allein.

Reproduzierbarkeit ist keine Einstellung, sondern der Zustand einer vollständigen Kette.

Skizze 2: Der Seed ist ein Glied von zwölf. Solange die Kette geschlossen ist, hält das Versprechen; ein einziges verändertes Glied genügt, und das Ergebnis wandert – ohne Fehlermeldung.

Glied der Kette	Was passiert, wenn es sich ändert	Wie du es festnagelst
Prompt, auch nur ein Komma	anderes Bild, oft deutlich anders	Prompt als Textdatei neben dem Bild ablegen, nicht nur in der Oberfläche
Modell oder Decoder	anderes Bild, selbst bei identischem Seed	Dateiname, Version und Prüfsumme notieren, Dateien nie überschreiben
Zusatzmodell (LoRA) und Stärke	verschobener Stil, veränderte Komposition	Name, Version und Gewichtung mitschreiben
Sampler, Scheduler, Schrittzahl	anderer Rechenweg, anderes Ergebnis	als festen Satz je Bildsorte definieren
Guidance und Auflösung	anderes Bild bis hin zu anderem Bildaufbau	zum Parametersatz gehören lassen, nicht ad hoc drehen
Stapelgröße und Indexposition	veränderte Reihenfolge, andere Zuordnung	in der Produktion auf 1 stellen, Index protokollieren
Rauschquelle (CPU, GPU, NV)	völlig anderes Startrauschen	Einstellung dokumentieren, zwischen Werkzeugen nie übertragen
Quantisierung und Schnellmodus	abweichende Rechnung, Reproduzierbarkeit fällt weg	in der Produktion abschalten oder Variante festschreiben
Bibliotheks- und Treiberversion	andere Rechenkerne, abweichende Rundung	Produktionsumgebung einfrieren, Versionsstand mitschreiben
Grafikkartenmodell	andere Kernel, gelegentlich sichtbare Unterschiede	Produktionsrechner benennen, nicht quer über den Fuhrpark rechnen

Eine Variable nach der anderen

Der Seed ist das beste Werkzeug für saubere Experimente, das die Bildgenerierung zu bieten hat. Setz ihn fest, ändere genau eine Einstellung, sieh dir das Ergebnis an. Dann weißt du, was die Änderung bewirkt hat – und nicht, was der Zufall dazwischengefunkt hat.

Wer zwei Regler gleichzeitig dreht und den Seed dabei würfeln lässt, lernt über sein Modell genau nichts. Er sammelt nur Bilder.

Der verlorene Seed – ein Praxisdesaster mit Ansage

Bis hierher war alles Theorie, und die Theorie ist freundlich. Der Alltag ist es nicht. In der Praxis geht der Seed nicht verloren, weil jemand ihn wegwirft, sondern weil eine Kette von völlig vernünftigen Einzelschritten ihn Stück für Stück aus dem Material herausschiebt.

Wie die Werkzeuge ihn selbst überschreiben

Die häufigste Falle ist eine Standardeinstellung. In ComfyUI hat der Seed-Wert einen zweiten Schalter direkt darunter, der bestimmt, was nach dem Absenden des Auftrags mit dem Feld passiert: unverändert lassen, hochzählen, herunterzählen oder neu würfeln. Werkseitig steht er auf Würfeln. Das ist für die Suchphase genau richtig und in dem Moment fatal, in dem ein Treffer dabei ist: Das Feld zeigt dann bereits den Wert für den nächsten Lauf, nicht den, aus dem das Bild vor dir entstanden ist. Wer diese Zahl abschreibt, schreibt die falsche ab.

In der WebUI-Familie ist die Falle die Minus-Eins, die nach jedem Lauf stehen bleibt. Immerhin zeigt die Oberfläche den tatsächlich verwendeten Wert unter dem Bild an, und ein kleiner Knopf neben dem Feld übernimmt ihn mit einem Klick. Dieser Knopf ist die billigste Versicherung der ganzen Bildproduktion, und er wird trotzdem seltener gedrückt als er sollte.

Was in den Metadaten steckt – und wie lange

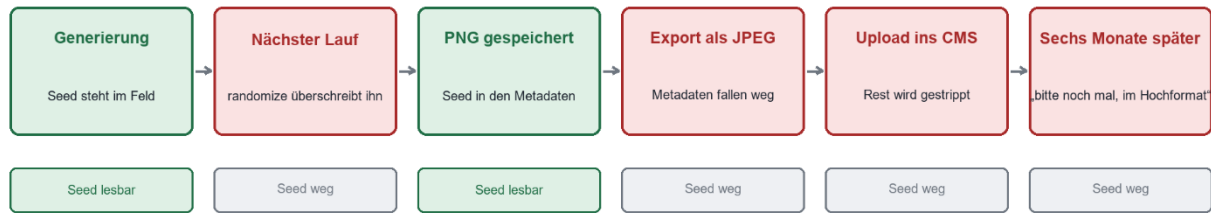
Die gute Nachricht: Beide Werkzeugfamilien schreiben ihre Parameter in die PNG-Datei. Die WebUI-Familie legt einen Textblock mit Prompt, Seed, Sampler, Schrittzahl, Guidance und Modell-Prüfsumme ab. ComfyUI geht weiter und bettet den kompletten Workflow als Datenstruktur ein – du kannst eine erzeugte PNG-Datei in die Oberfläche ziehen und bekommst den gesamten Arbeitsablauf zurück, Seed inklusive. Das ist eine der besten Eigenschaften des Werkzeugs und rettet regelmäßig Projekte.

Die schlechte Nachricht steht im Wort „PNG“. Diese Zusatzinformationen leben in Textblöcken des Dateiformats, und sie überleben fast nichts. Ein Export nach JPEG für den Druck: weg. Eine Bearbeitung in einem Bildprogramm ohne ausdrückliches Beibehalten: weg. Ein Upload in ein Redaktionssystem, das automatisch skaliert und in ein Web-Format konvertiert: weg. Ein Weg über eine soziale Plattform: weg, dort wird ohnehin fast alles entfernt. Und selbst wenn die Datei unangetastet bleibt, reicht ein beherzter Aufräumtag auf dem Netzlaufwerk.

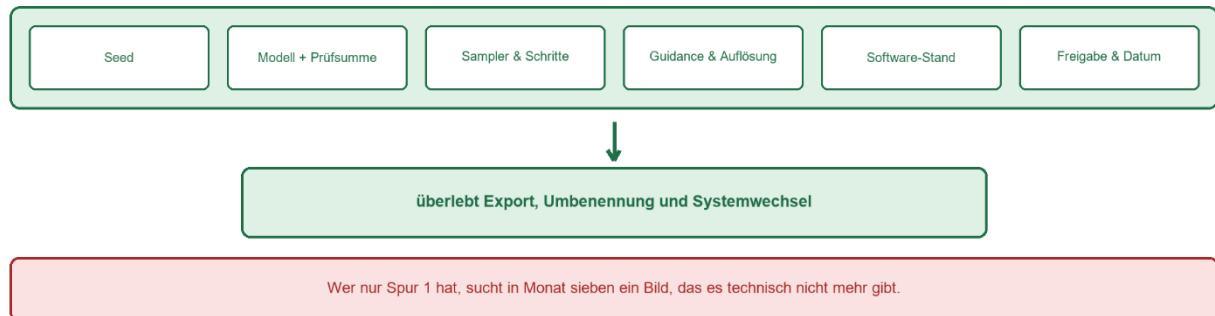
Wo der Seed unterwegs verloren geht

Oben die Bilddatei, unten das Protokoll daneben. Nur eine der beiden Spuren kommt an.

Spur 1: die Bilddatei



Spur 2: das Protokoll außerhalb des Bildes



Skizze 3: Der Weg eines Bildes durch die Produktion. Die Metadaten überleben ihn nicht – ein Protokoll neben der Datei schon.

Den Seed eines fremden Bildes herausfinden – geht das?

Nein, und zwar aus zwei unabhängigen Gründen. Erstens steckt die Zahl nicht im Bild, sondern nur in Zusatzinformationen, die jemand aktiv mitgegeben haben muss. Manche Plattformen zeigen sie bei eigenen Bildern an, die meisten nicht. Zweitens ist die Rückrechnung mathematisch aussichtslos: Selbst wenn du das Modell und sämtliche Einstellungen exakt kennen würdest, müsstest du den Startwert durch Ausprobieren finden – bei über vier Milliarden Möglichkeiten im kleinen und über achtzehn Trillionen im großen Zahlenraum, jede davon mit einer vollständigen Bildgenerierung als Prüfkosten.

Es gibt Werkzeuge, die behaupten, so etwas zu können. Sie erkennen in Wahrheit nur, ob in der Datei noch ein Parameterblock steht. Steht er nicht drin, ist der Seed weg – endgültig. Was sich einem fremden Bild tatsächlich ansehen lässt und was nicht, behandelt [KI-generierte Bilder erkennen](#) ausführlich. Die Kurzfassung: Kennzeichnungen und Signaturen lassen sich prüfen, Erzeugungsparameter nicht rekonstruieren.

Die drei Handgriffe, die den Treffer sichern

Erstens: Sobald ein Bild sitzt, den Seed festsetzen – in ComfyUI den Schalter auf „fixed“, in der WebUI-Familie den Übernahme-Knopf drücken. Vor dem nächsten Lauf, nicht danach.

Zweitens: Die erzeugende PNG-Datei unverändert archivieren, bevor irgendein Export passiert. Das Original ist das Sicherungsmedium, die Ableitung nicht.

Drittens: Seed, Modellversion und Parametersatz in eine Zeile außerhalb der Bilddatei schreiben. Das dauert zwanzig Sekunden und ist das einzige, was Systemwechsel überlebt.

Warum das keine Frage der Sorgfalt ist

Es ist verlockend, den verlorenen Seed als Schlamperei abzutun. Das trifft es nicht. Die Person, die das Bild erzeugt hat, hat in diesem Moment ein Bild gesucht und kein Archiv aufgebaut – und die

Oberflächen unterstützen sie dabei nach Kräften. Kein gängiges Werkzeug fragt beim Speichern: „Soll ich den Lauf protokollieren?“ Wer Wiederauffindbarkeit will, muss sie organisieren, und zwar außerhalb der Kreativarbeit. Genau das ist der Punkt, an dem aus einer Bastelinstallation eine Produktion wird – der Aufbau dafür steht in [Aufbau einer skalierbaren Bild-Maschine mit ComfyUI und Flux](#). Und weil es dabei weniger um Technik als um eingeübte Handgriffe geht, ist [KI-Bildgenerierung Schulung](#) für Teams meist der wirksamere Hebel als die nächste Grafikkarte.

Der Seed im Cloud-Zeitalter: nicht überall gibt es ihn noch

Wer in den letzten Jahren nur lokal gearbeitet hat, hält den Seed für eine Selbstverständlichkeit. Er ist keine. Der Stand im September 2026 ist bemerkenswert uneinheitlich, und die Richtung geht bei den großen Anbietern eher weg vom offenen Startwert.

Wer einen Seed anbietet – und wer nicht

Bei OpenAI führt keiner der Bild-Endpunkte einen Seed-Parameter; weder die Erzeugung noch die Bearbeitung noch die Variantenbildung nimmt einen solchen Wert an. Bei Google gilt dasselbe für die Bildmodelle der Nano-Banana-Reihe: Einen Seed gibt es nur in der allgemeinen Generierungskonfiguration für Textmodelle, nicht an den Bildfunktionen. Auch xAI verzichtet darauf. Drei der größten Anbieter geben dir also schlicht keine Möglichkeit, den Startpunkt zu fixieren.

Das ist kein Versehen, sondern folgt aus der Bauweise. Diese Systeme sind keine reinen Diffusionspipelines mehr, sondern multimodale Modelle mit einem Verarbeitungsschritt davor, der den Auftrag interpretiert. Was dort passiert, ist selbst bei identischer Eingabe nicht deterministisch. Ein Seed würde eine Zusage suggerieren, die das System nicht halten kann – und dann hätte man ein Feld, das Erwartungen weckt und Support kostet.

Auf der anderen Seite stehen die Anbieter, die den Seed ernst nehmen. Midjourney führt ihn seit der 8er-Reihe wieder verlässlich und beschreibt das Ergebnis selbst als „zu 99 Prozent identisch“ – eine ungewöhnlich ehrliche Formulierung, die genau das ausdrückt, worum es in diesem Beitrag geht. Stability liefert den verwendeten Seed in der Antwort zurück, verspricht aber ausdrücklich keine Determiniertheit, sondern „ähnliche Ergebnisse“. Und Ideogram hat in der vierten Generation den Seed als Eingabe fallen lassen, gibt in der Antwort aber weiterhin einen zurück – ein Wert, mit dem sich nichts anfangen lässt.

Was an die Stelle des Seeds tritt

Black Forest Labs hat für dieses Problem die aus meiner Sicht sauberste Lösung gefunden, und sie verschiebt den Bezugspunkt. Der reguläre FLUX.2-Endpunkt bleibt unverändert – ein festgeschriebener Stand für Arbeitsabläufe, die Reproduzierbarkeit brauchen. Daneben steht ein Vorschau-Endpunkt, auf dem Verbesserungen zuerst landen und der sich laufend ändert. Die Einheit der Reproduzierbarkeit ist damit nicht mehr der Seed, sondern die gepinnte Modellversion. Der Seed regelt nur noch die Streuung innerhalb dieser Version.

Das ist der Denkfehler, den viele Projekte teuer bezahlen: Ein Seed über ein Modell-Update hinweg ist wertlos. Kein Anbieter gibt eine Zusage, dass eine Zahl nach einer neuen Modellgeneration dasselbe Bild liefert – und in der Praxis tut sie es auch nicht. Seit FLUX.2 im November 2025 erschienen ist, hat allein diese Familie mehrere Varianten und im Juli 2026 mit FLUX 3 eine neue Generation bekommen. Wer Bildserien über Monate konsistent halten muss, plant das über Versionsstände, Referenzbilder und feste Parametersätze – wie das für eine Marke aussieht, steht in [Markenkonsistente KI-generierte Bilder](#). Welche Werkzeuge derzeit überhaupt zur Auswahl stehen, ordnet [Übersicht der wichtigsten KI-Bildgeneratoren](#) ein; die Modelle für den eigenen Rechner listet [Die 15 wichtigsten Modelle zur lokalen Bildgenerierung](#).

Anbieter und Modellfamilie	Seed als Eingabe	Zusage zur Wiederholbarkeit
OpenAI, GPT-Image-Reihe	nein, an keinem Bild-Endpunkt	keine; Wiederholbarkeit ist nicht vorgesehen
Google, Nano-Banana-Reihe	nein, nur bei Textmodellen	keine; alle Bilder tragen ein unsichtbares Wasserzeichen
xAI	nein	keine
Midjourney (8er-Reihe)	ja, 0 bis 4.294.967.295	ausdrücklich „zu 99 Prozent identisch“, nicht bitgenau
Stability	ja, wird zurückgeliefert	„ähnliche Ergebnisse“, ausdrücklich keine Garantie
Black Forest Labs, FLUX-Familie	ja, beliebige ganze Zahl	über den festgeschriebenen Endpunkt, nicht über den Vorschau-Endpunkt
Lokale Installation (ComfyUI und Verwandte)	ja, voller 64-Bit-Bereich	bitgenau, solange die gesamte Umgebung eingefroren bleibt

Reproduzierbarkeit ist eine Architekturfrage

Die Zeile ganz unten in der Tabelle ist der eigentliche Grund, warum Unternehmen mit Nachweispflichten lokal rechnen. Nicht wegen der Kosten und nicht wegen der Datenhoheit allein, sondern weil nur dort die gesamte Kette in eigener Hand liegt – bis hinunter zur Bibliotheksversion. Wer in der Cloud arbeitet, verlagert die Frage nicht, sondern gibt sie ab. Das kann völlig richtig sein. Man sollte es nur wissen, bevor jemand nach dem Originalbild fragt.

Den Vergleich beider Betriebsmodelle mit allen Konsequenzen findest du in [Cloud vs. lokale KI-Bildgenerierung](#); wie sich eine moderne Modellfamilie in die vorhandene Microsoft-Landschaft einfügt, zeigt [Black Forest FLUX.2 in PowerPoint und Copilot](#).

Der Seed als Teil des Nachweises

Bisher ging es um Bequemlichkeit: Ein Bild wiederfinden ist angenehm. Seit dem Sommer 2026 ist daraus in Teilen eine Pflichtübung geworden, und der Seed bekommt eine zweite Rolle – nicht als Kennzeichnung nach außen, sondern als Bestandteil der internen Beweisführung.

Was seit dem 2. August 2026 gilt

Die Transparenzpflichten der europäischen KI-Verordnung sind an diesem Tag wirksam geworden. Wer ein generatives System betreibt, muss dafür sorgen, dass synthetisch erzeugte Bilder maschinenlesbar als solche erkennbar sind. Das Gesetz ist technologieneutral formuliert und nennt kein konkretes Verfahren; in den Materialien zur Umsetzung taucht der Herkunftsnachweis nach dem C2PA-Standard – die signierten „Content Credentials“ – als Beispiel auf. Bei Verstößen sind Bußgelder bis zu 15 Millionen Euro oder drei Prozent des weltweiten Jahresumsatzes vorgesehen, je nachdem, welcher Betrag höher ist.

Wichtig ist die Abgrenzung: Der Seed ist kein Kennzeichnungsmerkmal. Er steht in keinem Standard, er wird von keiner Prüfstelle abgefragt, und er macht ein Bild nicht erkennbar. Die Kennzeichnung ist eine eigene Baustelle, beschrieben in [KI-generierte Bilder erkennen](#); die rechtliche Einordnung mit Fristen und Zuständigkeiten steht in [EU AI Act: Was seit dem 2. August 2026 gilt](#).

Wofür der Seed dagegen sehr wohl gebraucht wird, ist die Frage dahinter: Kannst du belegen, wie ein Bild entstanden ist? Diese Frage kommt nicht von der Aufsicht, sondern aus dem eigenen Haus – von der Rechtsabteilung, wenn jemand eine Ähnlichkeit zu einem geschützten Motiv behauptet; vom Einkauf, wenn geklärt werden muss, unter welcher Lizenz das verwendete Modell stand; vom

Kunden, der wissen will, ob ein bestimmtes Gesicht einer realen Person nachempfunden ist. In allen drei Fällen ist die Antwort nur so gut wie das Protokoll.

Was ins Protokoll gehört

Ein brauchbarer Nachweis besteht aus drei Blöcken. Der erste beschreibt, wie das Bild entstanden ist: Seed, Prompt und Negativ-Prompt, Sampler und Scheduler, Schrittzahl, Guidance, Auflösung sowie Stapelgröße und Indexposition. Der zweite beschreibt, womit es entstanden ist: Modellname mit Version und Prüfsumme, Decoder, Zusatzmodelle mit Stärke, Quantisierungsvariante, Rauschquelle, Software-Stand und die verwendete Grafikkarte. Der dritte beschreibt, was rechtlich zählt: Kennzeichnung, Signatur, Modell-Lizenz, Rechte am Motiv, Freigabe, Datum und verantwortliche Person.

Die Nachweiskarte eines Bildes

Der Seed ist ein Feld davon – und ohne die anderen wertlos.

Wie es entstanden ist	Womit es entstanden ist	Was rechtlich zählt
seed ☐	modell + version ☐	kennzeichnung maschinenlesbar ☐
prompt / negativ-prompt ☐	prüfsumme der gewichte ☐	c2pa-signatur ☐
sampler + scheduler ☐	vae ☐	modell-lizenz ☐
schritte ☐	lora + stärke ☐	nutzungsrechte am motiv ☐
guidance (cfg) ☐	quantisierung (fp8, gguf) ☐	freigabe durch ☐
auflösung ☐	rauschquelle (cpu/gpu) ☐	datum des laufs ☐
batch_size + index ☐	software-version ☐	verantwortliche person ☐
denoise / stärke ☐	grafikkarte + treiber ☐	ablageort des originals ☐

Spalte 1 steckt oft in der Bilddatei. Spalte 2 fast nie. Spalte 3 nirgends.

Deshalb gehört die Karte in einen Katalog neben dem Bild – nicht in das Bild hinein.

Skizze 4: Drei Blöcke gehören zu jedem Bild. Der erste steckt oft in der Datei, der zweite fast nie – und der dritte nirgends, wenn ihn niemand hinschreibt.

Handgeführt hält das keine drei Wochen durch. In meinen eigenen Aufbauten liegt der erste und zweite Block in UB.Forge: Die Workflows liegen dort, jeder Lauf wird mit allen Werten mitgeschrieben, und vor dem Produktionslauf greift ein Plausibilitäts-Check – unter anderem auf die Frage, ob der Seed überhaupt festgesetzt ist oder noch auf Würfeln steht. Der dritte Block steht in UB.Dossier, zusammen mit Modellversion und Lizenzstand: Wer einen Wert ändert, ändert ihn dort, und was dort steht, ist gleichzeitig Katalog und Nachweis. Wie so ein Katalog auf Microsoft-Technik aussehen kann, beschreibt [Digital Asset Management für KI-Bilder mit SharePoint Online und Power Platform](#).

Ein Seed allein belegt gar nichts

Die Zahl 1734921 in einer Tabelle ist ohne Kontext wertlos. Sie sagt nicht, welches Modell lief, in welcher Genauigkeit, auf welcher Karte und mit welchem Software-Stand. Ein Wiederherstellungsversuch auf dieser Grundlage scheitert – und schlimmer: Er scheitert lautlos, weil ein Bild herauskommt. Nur eben nicht das gesuchte.

Wer den Seed protokolliert, aber nicht die Modellversion, hat sich ein Gefühl von Ordnung gekauft und sonst nichts.

Varianten statt Neuwürfe

Ein letzter praktischer Punkt, weil er ständig gefragt wird. Wenn ein Bild fast passt und nur eine Kleinigkeit anders sein soll, ist der Seed dein Anker – aber nicht dein Regler. Du hältst ihn fest und veränderst genau eine andere Stelle: eine Formulierung im Prompt, den Guidance-Wert um einen halben Punkt, die Schrittzahl. Das Ergebnis bleibt in der Nähe des Ausgangsbildes, weil das Rauschfeld dasselbe ist.

Für feinere Abstufungen bietet die WebUI-Familie einen zweiten Seed mit einer Mischungsstärke an: Das Startrauschen wird dann zwischen zwei Zahlen interpoliert, und bei geringer Stärke bekommst du echte Geschwister desselben Bildes. In ComfyUI erreichst du dasselbe über Knoten, die Rauschfelder mischen, oder über eine Bild-zu-Bild-Verarbeitung mit niedriger Veränderungsstärke. Was in beiden Fällen nicht funktioniert, ist die naheliegende Idee, einfach den Seed um eins zu erhöhen. Das ist kein Feintuning, sondern ein Neuwurf.

Häufige Fragen zum Seed

Was ist ein Seed bei der KI-Bildgenerierung?

Der Seed ist eine ganze Zahl, die als Startwert eines Pseudozufallsgenerators dient. Aus dieser Zahl wird das Rauschfeld erzeugt, mit dem jede Generierung beginnt und aus dem das Modell Schritt für Schritt das Bild herausrechnet. Derselbe Seed liefert dasselbe Startrauschen – und bei sonst identischer Einstellungskette dasselbe Bild. Er ist damit der Schlüssel zur Reproduzierbarkeit, aber eben nur der Schlüssel und nicht die ganze Tür.

Was bedeutet Seed –1?

In der Automatic1111-Familie und ihren Ablegern ist eine –1 keine echte Seed-Zahl, sondern die Anweisung, bei jedem Lauf einen neuen Zufallswert zu ziehen. Andere Umgebungen benutzen andere Konventionen: Bei der Stability-Schnittstelle bedeutet eine Null dasselbe, ebenso das Weglassen des Parameters, und ComfyUI hat dafür gar kein Sonderzeichen, sondern einen eigenen Schalter neben dem Feld. Für die Suchphase ist das richtig, für den Treffer nicht.

Wie finde ich den Seed eines erzeugten Bildes?

In der WebUI-Familie steht er unter dem Bild und lässt sich mit einem Knopf ins Feld übernehmen; zusätzlich landet er im Parameterblock der PNG-Datei. In ComfyUI steckt der gesamte Workflow in der PNG-Datei – du kannst sie in die Oberfläche ziehen und bekommst alles zurück, Seed inklusive. Vorsicht bei der Zahl, die nach dem Lauf im Feld steht: Wenn der Schalter auf Würfeln oder Hochzählen steht, ist das bereits der Wert für den nächsten Lauf, nicht der des sichtbaren Bildes.

Kann ich den Seed eines fremden Bildes herausfinden?

In aller Regel nein. Die Zahl steckt nicht im Bildinhalt, sondern nur in Zusatzinformationen, die jemand mitgegeben haben muss – und die überstehen weder einen Export noch einen Upload auf den meisten Plattformen. Eine Rückrechnung aus dem fertigen Bild ist ausgeschlossen: Der Zahlenraum ist zu groß, und man bräuchte zusätzlich das exakte Modell samt aller Einstellungen. Werkzeuge, die etwas anderes versprechen, lesen in Wahrheit nur Metadaten aus.

Gibt es gute und schlechte Seeds?

Nein. Die Höhe der Zahl sagt nichts über Bildqualität aus, und es gibt keine Lieblingszahlen, die grundsätzlich bessere Ergebnisse liefern. Was es gibt, sind Rauschfelder, deren grobe Struktur zufällig gut zu einem bestimmten Bildaufbau passt – das ist aber eine Eigenschaft des Zusammenspiels aus Rauschen, Prompt und Modell, nicht der Zahl. Ein Seed, der für ein Motiv hervorragend funktioniert, kann beim nächsten völlig danebenliegen.

Bekomme ich mit demselben Seed bei einem anderen Modell dasselbe Bild?

Nein. Der Seed legt nur das Startrauschen fest; was daraus wird, rechnet das jeweilige Modell. Anderer Checkpoint, anderes Bild. Das gilt schon für eine neue Version derselben Modellfamilie und erst recht für den Sprung zu einem anderen Hersteller. Ein Seed ist immer nur im Kontext genau der Umgebung gültig, in der er entstanden ist.

Warum liefert derselbe Seed in ComfyUI ein anderes Bild als in der WebUI?

Weil das Rauschen an verschiedenen Stellen erzeugt wird. ComfyUI zieht es grundsätzlich auf dem Hauptprozessor, was es über unterschiedliche Hardware hinweg gut reproduzierbar macht. Die WebUI-Familie erzeugt es standardmäßig auf der Grafikkarte und bietet die Wahl zwischen CPU, GPU und einer NVIDIA-Nachbildung. Zwei verschiedene Generatoren, zwei verschiedene Rauschfelder, zwei verschiedene Bilder – auch bei identischer Zahl. Seeds lassen sich zwischen diesen Welten nicht übertragen.

Wie groß darf ein Seed sein?

Das hängt am Werkzeug. Der klassische Bereich ist 0 bis 4.294.967.295, also der volle 32-Bit-Raum; dort arbeiten unter anderem Midjourney und die WebUI-Familie. ComfyUI nimmt den vollen 64-Bit-Bereich bis 18.446.744.073.709.551.615 an, Ideogram deckelte in der dritten Generation bei 2.147.483.647. Praktisch ist die Obergrenze bedeutungslos – wichtig ist nur, dass du innerhalb eines Werkzeugs bleibst, weil ein zu großer Wert sonst abgeschnitten oder abgewiesen wird.

Warum bieten ChatGPT und Gemini keinen Seed an?

Weil ihre Bildmodelle keine reinen Diffusionspipelines mehr sind, sondern multimodale Systeme mit einem interpretierenden Schritt davor. Dieser Schritt ist selbst bei identischer Eingabe nicht deterministisch, ein fester Startwert würde also eine Zusage suggerieren, die das System nicht einhalten kann. Im September 2026 führt weder OpenAI noch Google noch xAI einen Seed-Parameter an den Bild-Funktionen. Wer dort Konsistenz braucht, arbeitet mit Referenzbildern und festen Prompt-Bausteinen statt mit einer Zahl.

Bleibt ein Seed über Modell-Updates hinweg gültig?

Nein, und kein Anbieter verspricht das. Sobald sich die Modellversion ändert, ist die Zahl nur noch ein Startwert ohne Bezug zum alten Ergebnis. Black Forest Labs zieht daraus die sauberste Konsequenz und trennt einen festgeschriebenen Endpunkt von einem Vorschau-Endpunkt, der sich laufend weiterentwickelt. Die Einheit der Reproduzierbarkeit ist damit die gepinnte Modellversion, nicht der Seed. Wer lokal arbeitet, erreicht dasselbe, indem er Gewichtsdateien niemals überschreibt.

Reicht der Seed als Nachweis, wie ein Bild entstanden ist?

Nein. Er ist ein Feld von etwa zwanzig. Ohne Modellname, Version, Prüfsumme, Sampler, Schrittzahl, Guidance, Auflösung, Rauschquelle, Quantisierung und Software-Stand lässt sich aus der Zahl nichts rekonstruieren. Und für die rechtliche Seite kommen Kennzeichnung, Modell-

Lizenz, Nutzungsrechte und Freigabe hinzu – Angaben, die in keiner Bilddatei stehen. Der Nachweis gehört in einen Katalog neben das Bild, nicht in das Bild hinein.

Wie erzeuge ich mit dem Seed gezielt Varianten?

Indem du ihn festhältst und genau eine andere Stelle veränderst: eine Formulierung im Prompt, einen halben Punkt Guidance, ein paar Schritte mehr. Für feine Abstufungen gibt es in der WebUI-Familie einen zweiten Seed mit Mischungsstärke, in ComfyUI Knoten zum Mischen von Rauschfeldern oder eine Bild-zu-Bild-Verarbeitung mit niedriger Veränderungsstärke. Was nicht funktioniert: den Seed um eins zu erhöhen. Das ist ein kompletter Neuwurf.

Soll ich mit festem oder zufälligem Seed arbeiten?

Mit beidem, nur nicht gleichzeitig. In der Suchphase lässt du würfeln, weil du Streuung sehen willst und dich die einzelne Zahl nicht interessiert. Sobald ein Bild sitzt, setzt du den Seed fest und arbeitest kontrolliert weiter. Der Fehler liegt nie beim Würfeln, sondern immer im fehlenden Wechsel – und der passiert genau in dem Moment, in dem man sich über ein gelungenes Bild freut und deshalb nicht daran denkt.

Fazit

Der Seed ist die einzige Einstellung der Bildgenerierung, die nichts über die Bildqualität aussagt und trotzdem über den Wert deiner Arbeit entscheidet. Er macht aus Glückstreffern wiederholbare Ergebnisse, aus Herumprobieren ein Experiment und aus einem Bild ein Ergebnis, über das man Auskunft geben kann.

Die Merksätze sind kurz. Der Seed ist der Startwert für das Rauschen, nicht das Bild. Derselbe Seed ergibt dasselbe Bild, aber nur bei identischer Kette – und diese Kette reicht von Prompt und Modell bis hinunter zu Rauschquelle, Zahlenpräzision und Treiberversion. Benachbarte Seeds sind keine benachbarten Bilder. Die Zahl eines fremden Bildes bekommst du nicht heraus. Und über ein Modell-Update hinweg ist ein Seed ungefähr so wertvoll wie ein Schlüssel zu einem ausgetauschten Schloss.

In der Cloud ist die Lage 2026 zusätzlich unübersichtlich geworden: Drei der größten Anbieter bieten überhaupt keinen Seed mehr an, andere liefern ihn zurück, ohne eine Zusage zu machen. Wer verbindliche Wiederholbarkeit braucht, kommt an der Frage nicht vorbei, wo die Kette liegt und wem sie gehört.

Praktisch läuft alles auf drei Handgriffe hinaus: nach dem Treffer den Seed festsetzen, das Original unverändert sichern und den vollständigen Parametersatz außerhalb der Bilddatei ablegen. Das kostet eine halbe Minute pro Bild und erspart die Tage, an denen ein Team ein Motiv sucht, das es technisch nicht mehr gibt. Wenn du das für ein Team aufsetzen willst – von der Umgebung über den Arbeitsablauf bis zum Nachweis – ist [KI-Bildgenerierung Beratung](#) der passende Einstieg; die strategische Einordnung dahinter steht in [KI-Bildgenerierung im Unternehmen – Die strategische Alternative zur Cloud](#).

Weiterlesen

- › Alles Übrige zum Thema im Überblick: [KI-Bildgenerierung](#)
- › Was nach dem Startrauschen passiert: [Sampling-Steps erklärt](#) und [Sampler & Scheduler erklärt](#)
- › Wie stark der Prompt das Ergebnis steuert: [CFG-Scale: Guidance in KI-Bildgeneratoren](#) und [Prompt und Negativ-Prompt in der KI-Bildgenerierung](#)
- › Der Seed im Stapel und was dabei schiefgeht: [batch size und Reproduzierbarkeit](#)
- › Woran das Bild sonst noch hängt: [Checkpoints, Modelle und VAE](#), [Was ist ein LoRA?](#) und [Auflösung & Seitenverhältnis bei KI-Bildern](#)

- › Kennzeichnung und Herkunftsnachweis: [KI-generierte Bilder erkennen](#) und [EU AI Act: Was seit dem 2. August 2026 gilt](#)
- › Vom Einzelbild zur Produktion: [Aufbau einer skalierbaren Bild-Maschine mit ComfyUI und Flux](#) und [Digital Asset Management für KI-Bilder mit SharePoint Online und Power Platform](#)
- › Warum die Bilder hier selbst erzeugt sind: [Dogfooding: Ich setze ein, was ich berate](#)