2023
3
ISSN 1433-2620 > 27. Jahrgang >> www.digitalproduction.com
Publiziert von Busch Glatz Germany GmbH

Deutschland € 17,90
Österreich € 19,–
Schweiz sfr 23,–

DIGITAL PRODUCTION

DIGITAL PRODUCTION

MAGAZIN FÜR DIGITALE MEDIENPRODUKTION

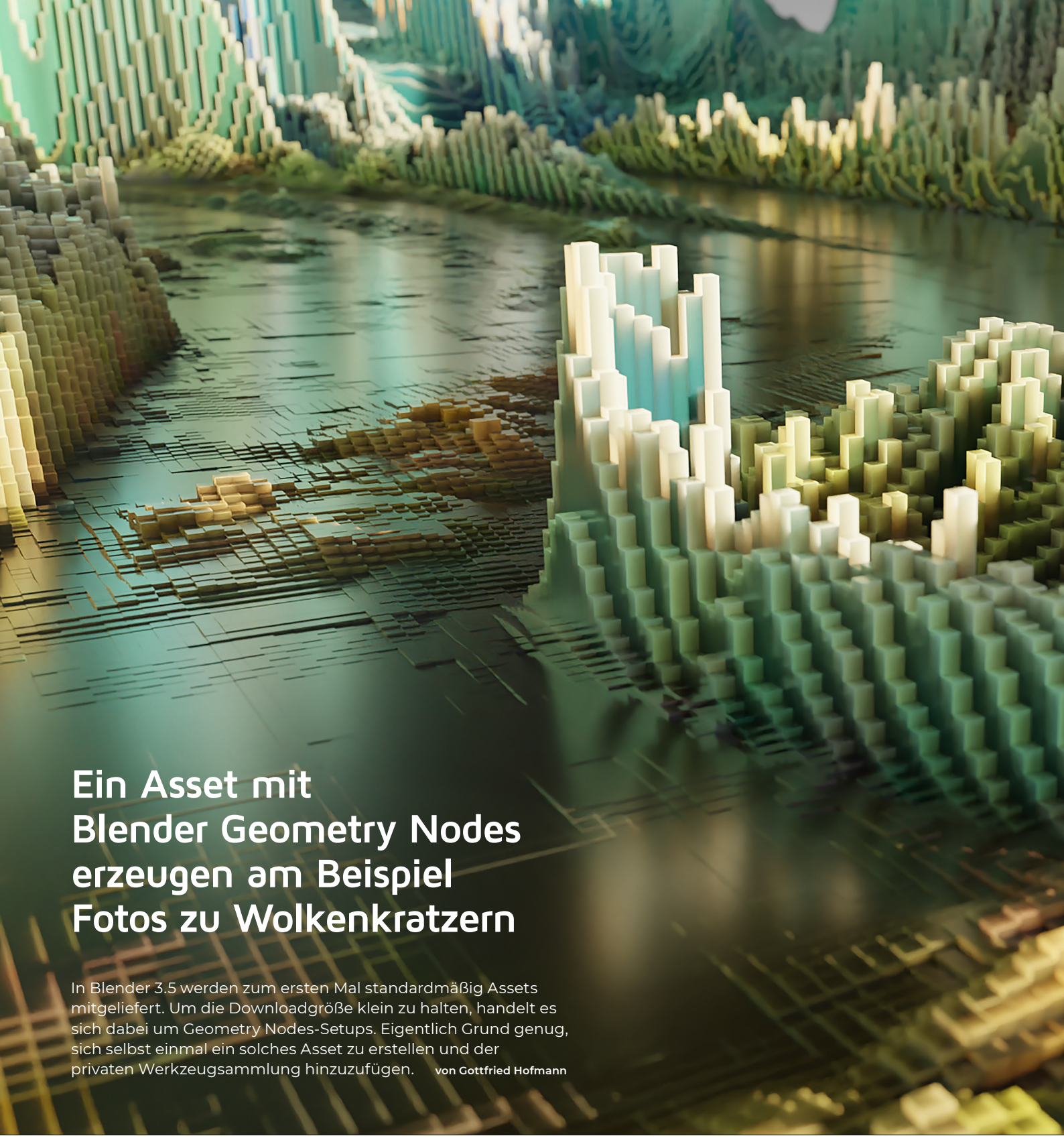
MAI | JUNI 03:2023

FMX!
The Beauty, Present, Interviews und Emil XR

Projekte
Firefly, Unreal Animatics, Blender Image Info Node

Tools
3ds Max, Blender 3.5, Nuke Studio und Braw

Tests
Resolve Mobile, RTX 4090, TyFlow Terrain & Rebelle 6



Ein Asset mit Blender Geometry Nodes erzeugen am Beispiel Fotos zu Wolkenkratzern

In Blender 3.5 werden zum ersten Mal standardmäßig Assets mitgeliefert. Um die Downloadgröße klein zu halten, handelt es sich dabei um Geometry Nodes-Setups. Eigentlich Grund genug, sich selbst einmal ein solches Asset zu erstellen und der privaten Werkzeugsammlung hinzuzufügen. **von Gottfried Hofmann**

Am besten wir verwenden dafür auch gleich noch neu hinzugekommene Nodes, um Blender 3.5 noch etwas besser kennenzulernen.

Neu in Blender 3.5 – Die Image Info-Node

Der Effekt, der in diesem Workshop erzeugt werden soll, ist auch mit älteren Blender-Versionen möglich. In Blender 3.5 ist aber eine Node hinzugekommen, mit der sich Anpassungen leichter durchführen lassen, nämlich die Node „Image Info". Mit dieser Node wird unser Ergebnis wirklich rund und etwas, das sich gerne Asset nennen darf.

Denn mit dieser Node kann die Größe eines Bildes in Pixeln ausgelesen werden. Mit dieser Information lässt sich nicht nur exakt ein Block auf einem Pixel des Bildes abstellen und einfärben, man kann auch gleich das korrekte Seitenverhältnis berechnen und garantieren, dass die Blöcke immer exakt gleich groß sind, egal wie viele oder wenig Pixel das Ausgangsbild hat.

Cat Content

Besagtes Ausgangsbild soll in unserem Fall ein Katzenporträt sein, das von **is.gd/blender_cat** heruntergeladen werden kann. Man kann aber ein beliebiges Bild dafür verwenden. Allgemein eignen sich kontrastreiche Bilder besonders gut, neben Cat Content werden für diesen Effekt auch gerne Satel-

In diesem Workshop soll ein Geometry Nodes-Setup erzeugt werden, mit dem man aus Bildern automatisch Landschaften wie diese generieren kann, die abstrakt an Wolkenkratzer erinnern.

Das Demobild für diesen Workshop, zu finden auf https://is.gd/blender_cat

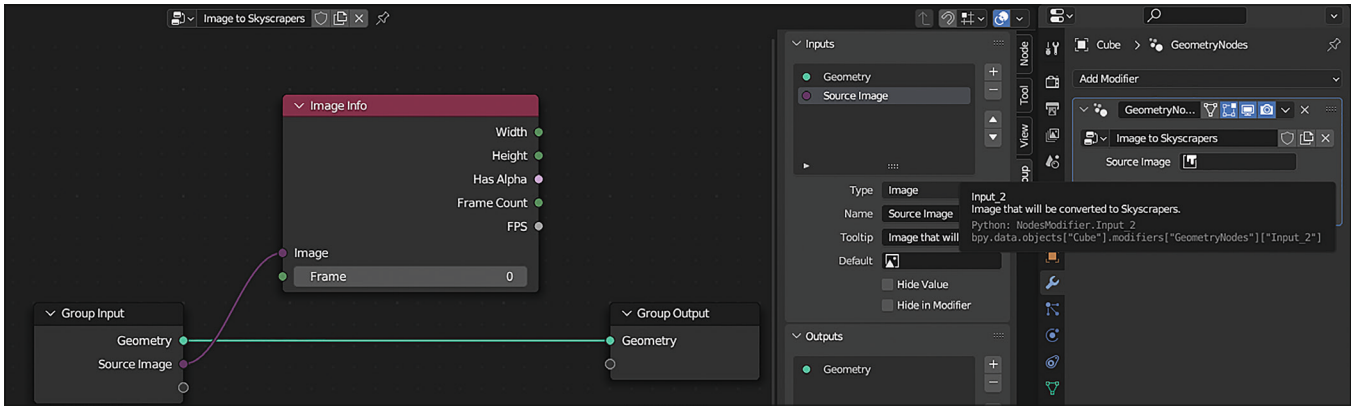
litenbilder genutzt, wahrscheinlich wegen der visuellen Nähe des Ergebnisses zu Stadtaufnahmen aus dem Flugzeug.

Unser Asset

Das fertige Asset wird von der Anwendung her wie ein Objekt per Drag-and-drop eingefügt werden. Als Eingabe wird es ein Bild und diverse Parameter erhalten. Damit der Anwender sofort etwas vor Augen hat, kommt es mit einem Beispielbild und ist auch gleich mit einem Material versehen. An jeden Pixel des Ausgangsbildes wird ein Quader gesetzt, der die Farbe jenes Bildpunktes annimmt.

Variable Größe oder variable Pixeldichte?

Man könnte das Asset so bauen, dass die Größe des Objektes entlang einer Achse immer gleich bleibt, egal wie hoch oder niedrig die Auflösung des Bildes ist. Da es



Das Interface für unser Asset Geometry Nodes ist vollständig in den Modifier Stack von Blender integriert und wird darüber auch gesteuert. Die Reihenfolge der Eingabefelder, ihren Datentyp, Name, Tooltip und Startwert lässt sich im Group-Tab der Sidebar des Node Editors definieren.

aber Bilder mit unterschiedlichen Seitenverhältnissen gibt, hätten wir dadurch nicht viel gewonnen, da ohne Verzerrungen die Größe entlang einer Achse variabel bleiben muss. Wenn man hingegen die Länge und Breite der Quader, die an jedem Pixel erscheinen, als festen Parameter nimmt, kann sich die Größe des Gesamtobjekts je nach Bild zwar entlang zweier Achsen ändern, dafür werden aber die Stäbchen nicht dünner oder dicker.

Der Default Cube darf bleiben

Auch wenn das Asset später quasi wie ein eigenes Objekt in die Szene eingefügt werden kann, braucht es für die erste Iteration quasi ein Startobjekt. Dafür wird der Default Cube herhalten, der ausnahmsweise mal nicht sofort gelöscht wird. Begib dich stattdessen zum Geometry Nodes Workspace, indem du auf den entsprechenden Tab im Header klickst. Der 3D Viewport ist jetzt merklich kleiner und der meiste Platz wird von einem Node Editor im unteren Bereich ausgefüllt, wo du mittels New eine neuen Node Tree erzeugst. Gib ihm am besten gleich einen sprechenden Namen wie „Image to Skyscrapers".

Alles bleibt beim Alten

Noch führt der Node Tree keine Änderungen am Würfel durch, es gibt nur eine Node Group Input und eine Group Output. Diese beiden Nodes regeln unsere Eingaben, also mit welchen Daten das Node Setup gefüttert wird, und unsere Ausgaben, was neben reiner Geometry auch Zusatzdaten sein können, auf die wir zum Beispiel in einem Shader zugreifen können.

Neu in Blender 3.5

Füge nun mittels Umschalt+A eine Image Info-Node hinzu. Dieser ist in Blender 3.5 neu hinzugekommen und findet sich unter Input – Scene. Über diese Node kann man Informationen zu einem in Blender geöffneten Bild erfragen, darunter die Breite und Höhe, was wir später noch benötigen werden. Um ein Bild darin zu laden, kann man auf Open klicken oder ein Bild in den lila Eingangssocket stecken. In unserem Beispiel werden wir später noch eine weitere Node verwenden, die ein Bild aus Eingabe bekommt. Daher macht es Sinn, besagtes Bild nicht in beiden Nodes separat zu laden, sondern weiter vorne im Node Tree und die beiden Nodes dann damit zu füttern. Noch besser wäre von Nutzerseite, wenn man das Bild außerhalb des Node Trees auswählen könnte. Ziehe dazu eine Verbindung aus dem lila Eingangssocket der Image Info-Node in den leeren Kreis-Ausgangssocket der Group Input-Node.

Unser eigener Modifier

Jetzt ist rechts im Properties Editor im Modifier Properties-Tab mit dem blauen Schraubenschlüssel ein neues Feld Image im Geometry Nodes-Modifier aufgetaucht. Mit Geometry Nodes erzeugen wir uns unsere eigenen Modifier, die vollständig im Stack integriert sind und von dort aus auch gesteuert werden. Konkret definieren wird dort, was aus den Ausgängen der Group Input-Node herauskommt, in unserem Fall bisher das Bild, das wir für den Effekt einsetzen. Ein übersichtliche Anordnung sowie passende Namen und eine optionale Beschreibung sind hier Pflicht, damit das Asset später von anderen Benutzern intuitiv eingesetzt werden kann. Man selbst profitiert natürlich auch, wenn man es nach Monaten mal wieder herauskramt und sich erst einmal zurechtfinden muss.

Best Practices

Klicke auf die Group Input-Node und öffne die Sidebar über die Taste N oder indem du auf das winzige < in der rechten oberen Ecke des Node Editors klickst. Dort im Tab Group kannst du die Reihenfolge der Sockets der Group Input und Group Output-Node ändern, neue Sockets hinzufügen und entfernen, ihren Datentyp ändern und einen Namen sowie einen Tooltip und einen Standardwert festlegen.

Zweckentfremdung

Einen kleinen Nachteil hat das Interface noch, und zwar kann man im Auswahlfeld für das Bild nur solche Auswählen, die bereits als Datablock in Blender geladen sind. Das bei anderen Feldern für die Auswahl von Bildern übliche Symbol zum Öffnen des Dateibrowsers von Blender fehlt. Das bedeutet, dass du dein Wunschbild an anderer Stelle laden musst, zum Beispiel in einem Blender Bildeditor. Es bietet sich an dieser Stelle an, den Spreadsheet Editor links oben im Geometry Nodes-Workspace zu einem Bildeditor zu machen, das gewünschte Bild zu laden und anschließend im Feld im Geometry Nodes-Modifier auszuwählen.

Ansichtssache

Jetzt kommen wir bereits an einen Punkt, an dem wir im Node Editor die ersten Daten einsehen können, und zwar konkret die Breite und Höhe des Eingangsbildes. Dieser werden aber nicht direkt am Socket angezeigt, dafür muss dieser zuerst evaluiert werden oder in anderen Worten: Wir müssen etwas reinstecken, dann können wir den Wert erfahren. Dafür gibt es die Viewer Node. Halte die Tasten STRG+Umschalt gedrückt und klicke auf die Image Info-Node. Es erscheint eine Viewer Node, die direkt mit dem obersten Socket verbunden ist. Zusätzlich ist im 3D Viewport der Würfel verschwunden, denn dort wird jetzt das angezeigt, was in die Viewer Node gefüttert wird. Und dort ist der Geometry-Socket

gerade leer, dafür befindet sich direkt darunter ein Socket Value. Fahre mit der Maus darüber und es erscheint der Wert mit dem Datentyp in Klammern in einem Tooltip, im Falle des Katzenbildes wäre das 1536 Integer. Damit lässt sich doch schon arbeiten.

Gitter

Wenn wir später an alle Pixel des Bildes ein Objekt setzen wollen, brauchen wir dafür Punkte. Und diese lassen sich mit Geometry Nodes sehr einfach aus einem Mesh nehmen. Füge über Mesh – Primitives eine Grid-Node hinzu und verbinde Vertices X mit Width und Vertices Y mit Height. Verbinde weiterhin den Ausgang Mesh mit dem Geometry-Eingang der Viewer Node und im 3D Viewport wird eine Ebene sichtbar. Wir sehen dort jetzt den Eingang der Viewer Node und nicht mehr das normale Ergebnis der Geometry Nodes. Dass es sich nur um eine Ansicht und nicht die tatsächliche Szene handelt, erkennst du, wenn du bei den Overlays die Statistiken einschaltest. Dort werden immer noch die acht Vertices vom Würfel angezeigt während es beim Grid eigentlich viel mehr sein müssten. Wenn du bei der Viewer Node auf das Augensymbol rechts oben klickst, wird diese deaktiviert und es erscheint wieder der Würfel. Wenn du den Mesh-Ausgang der Grid Node mit dem Geometry-Eingang der Group Output-Node verbindest, erscheint wieder die Ebene im Viewport, die Statistik zeigt diesmal aber die richtige Anzahl an Vertices an. Die Viewer Node ist sehr nützlich, um Zwischenergebnisse anzuzeigen. An dieser Stelle wird sie nicht mehr gebraucht, lösche sie mit der Taste X.

Seitenverhältnis

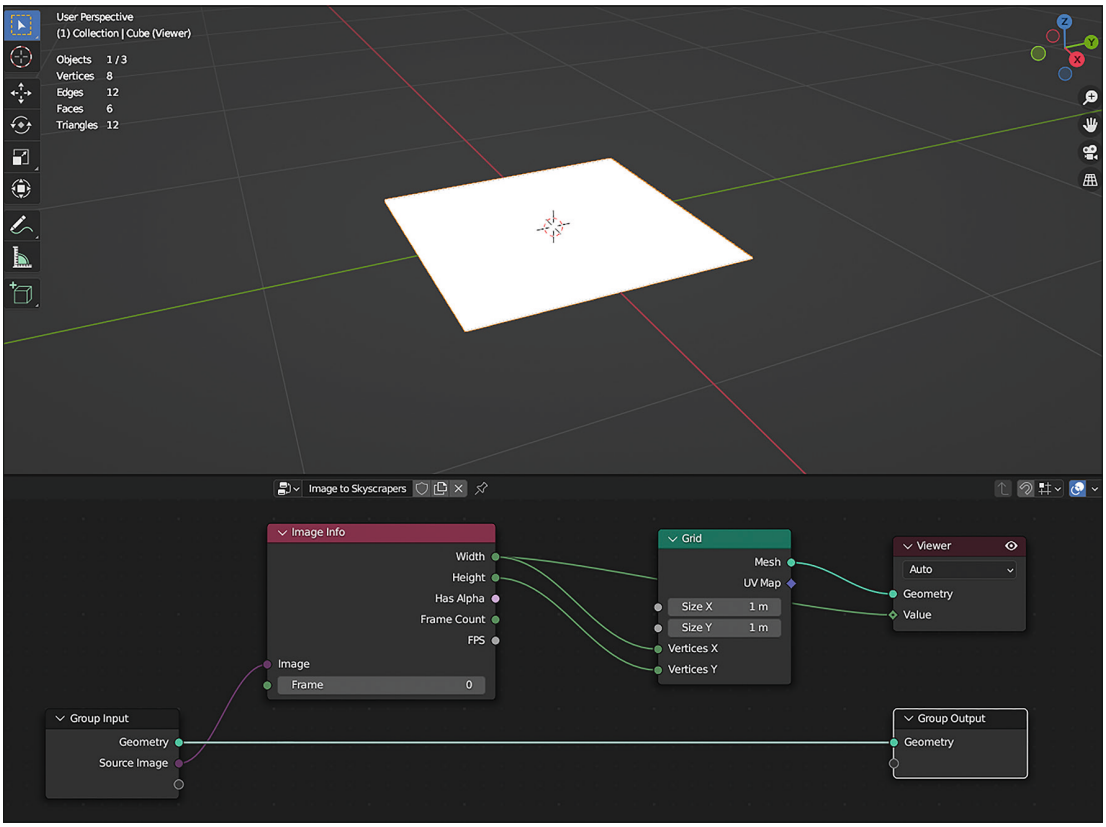
Das Gitter ist quadratisch, obwohl die Anzahl der Vertices in X und Y unterschiedlich sind. Denn die Abmessungen werden über Size X und Size Y definiert, und dieser stehen momentan beide auf einem Meter. Hier bietet es sich wieder an, mit der Width und Height des Bildes darauf Einfluss zu nehmen, damit das Seitenverhältnis stimmt. Hier gibt es verschiedene Arten, vorzugehen. Man könnte das Objekt z.B. so definieren, dass die Länge entlang einer Achse fest ist (z.B. immer ein Meter entlang X) und die andere Achse angepasst wird. Oder die Größe eines Pixels ist fest und das Objekt kann entlang beider Achsen wachsen und schrumpfen. Für dieses Projekt ist letztere Option besser geeignet, da dadurch die Dicke der einzelnen Stifte gleich bleibt. Dadurch können wir später im Shader Subsurface Scattering einsetzen und der Look bleibt gleich, egal welches Bild.

Mathematik

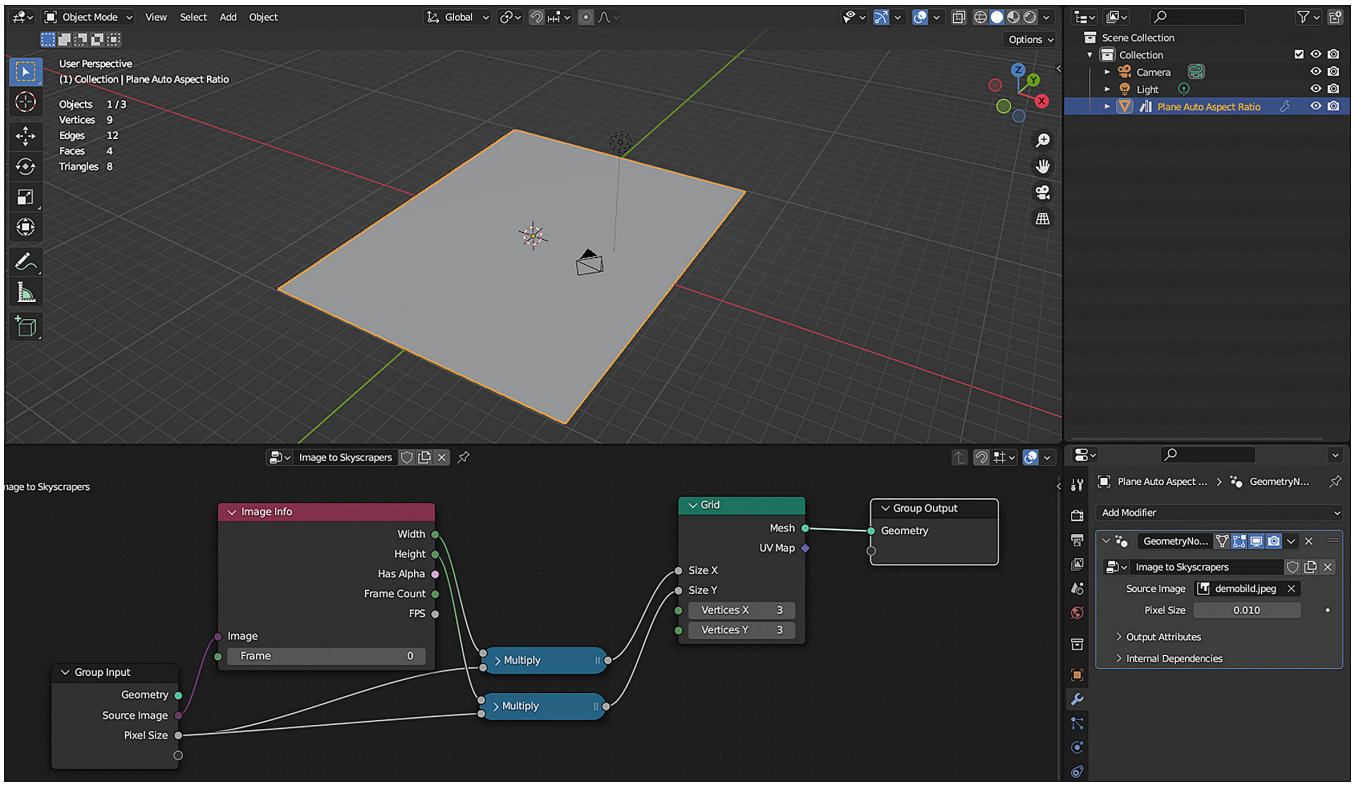
Füge eine Utilities > Math > Math-Node hinzu und wählen Sie im Dropdown-Menü Multiply. Verbinde den oberen Value-Eingangssocket mit dem Width-Socket der Image Info-Node und den Value-Ausgangssocket mit Size X bei Grid. Jetzt ist der Streifen extrem breit. Setze den unteren Wert von Value auf 0.01 und ziehe eine Verbindung zum leeren Kreis bei Group Input. Jetzt ist daraus ebenfalls ein Eingabefeld im Modifier geworden. Nenne es „Pixel Size“. Dadurch, dass du zuerst den Wert im Feld gesetzt und danach erst die Verbindung zur Group Input-Node aufgebaut hast, ist dieser Wert jetzt auch gleich der Defaultwert. Die Math Node heißt jetzt Multiply im Header, da dort immer die aktuell gewählte Operation angezeigt wird. Dupliziere sie mit Umschalt+D und verbinde Pixel Size von Group Input jetzt auch mit dem unteren Eingangssocket und Height der Image Info-Node mit dem oberen. Das Ergebnis verbindest du mit Size Y der Grid-Node.

Unser erstes Asset

Im Grund genommen hast du jetzt schon ein praktisches Asset. Nämlich eine Plane, deren Seitenverhältnis sich automatisch an das ausgewählte Bild anpasst. Wenn man das gleiche Bild als Textur verwendet, hat man dann keine Verzerrungen. Dafür würde es natürlich Sinn machen, die Vertices in X und Y auf einen festen Wert wie den Standard von drei zu setzen. Klicke im Outliner mit Rechtsklick auf den Cube und wähle Mark as Asset und gib ihm einen sprechenden Namen. Speichere dann eine Kopie der Datei mit File – Save Copy in ihrem persönlichen Asset-Ordner.



Die Viewer Node ist sehr praktisch, um Zwischenergebnisse anzuzeigen. Sie überschreibt die Viewport-Ausgabe, weshalb die Statistik hier die acht Vertices des Cubes anzeigt, obwohl es doch ein Gitter ist und es viel mehr sein sollten.

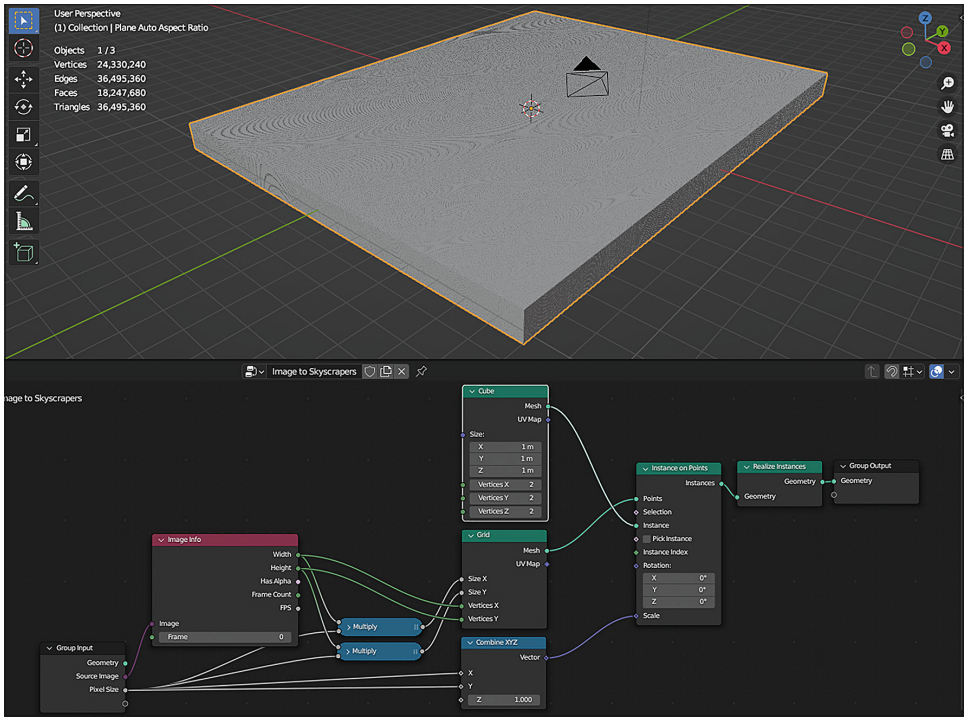


Bereits das Zwischenergebnis lässt sich gut als Asset einsetzen.

Verbinde danach wieder Width und Height mit Vertices X und Vertices Y.

Quader Ahoi

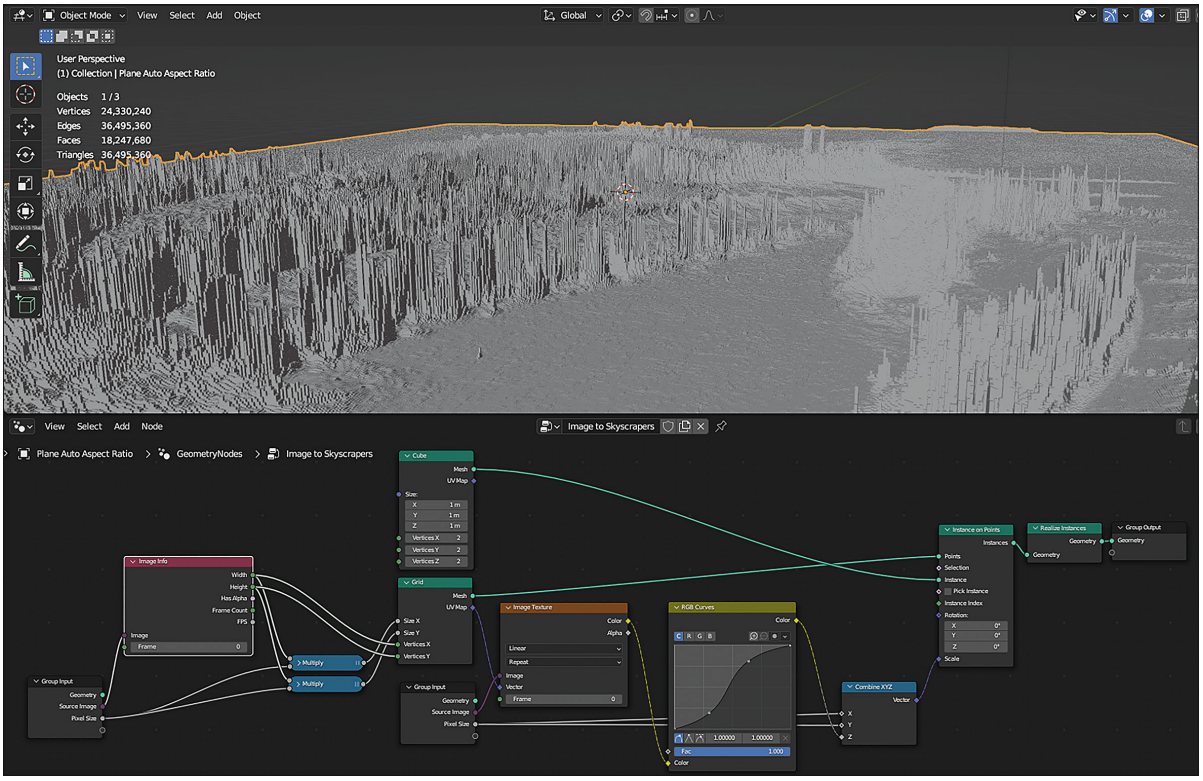
Im nächsten Schritt sollen die kleinen Quader gebaut werden, die später die abstrakte Landschaft erzeugen. Dafür wird an jeden Vertex bzw. Punkt des Gitters ein Würfel platziert und langgezogen. Füge eine Node Instances > Instance on Points hinzu und setze diese auf die Verbindung zwischen Grid und Group Output. Die Verbindung wird dabei automatisch erzeugt, im Viewport ist aber die Plane verschwunden. Das liegt daran, dass noch nichts instanziiert wird. Klicke auf den Instance-Socket und ziehe eine Verbindung heraus. Am Ende siehst du ein +-Symbol. Das bedeutet, dass, wenn du die Maus über einer freien Stelle loslässt, ein Suchfeld erscheint. Gib dort „Cube" ein und nach einer kurzen Wartezeit erscheinen Millionen von Würfeln im Viewport. Obwohl es sich um Instanzen handelt, dürfte Blender jetzt kaum noch bedienbar sein. Das liegt daran, dass Blender mit Millionen von Objekten so seine Probleme hat, selbst wenn es sich um Instanzen handelt. Füge eine Instances > Realize Instances-Node zwischen Instance on Points und Group Output ein. Jetzt ist das Ergebnis ein ganzes Objekt, das zwar aus Millionen von Vertices besteht, aber weder den Viewport, noch den Node Editor in die Knie zwingt.



Blender kann besser mit einem großen Objekt umgehen als mit Millionen von Kleinen.

Dicht an dicht

Damit sich die Würfel nicht überlappen, sollte die Scale der Instanzen bei X und Y jeweils der Pixel Size entsprechen. Die Instance on Points-Node hat unten einen Eingang Scale der momentan als ein Karo mit einem Punkt in der Mitte dargestellt wird. Das deutet an, dass die Skalierung für jede Instanz einzeln angewendet wird und man unterschiedliche Instanzen auch unterschiedlich behandeln könnte. Letzteres werden wir uns für die Z-Achse vorbehalten, deren Wert wir aus dem Bild extrahieren. Die X- und Y-Achsen sollen aber direkt die Pixel Size übernehmen. Ziehe mit der Maus eine Verbindung aus dem Scale Socket und suche nach Combine XYZ. Mit dieser Node kannst du dir Vektoren frei zusammenkonfigurieren und die Scale ist ein solcher Vektor. Verbinde X und Y mit der



Stäbe: Die Höhe der Quader wird über die Helligkeit der Pixel des Quellbildes definiert. Über eine RGB Curves-Node justieren wir den Kontrast.

Pixel Size aus der Group Input-Node und setze Z zunächst auf 1.0. Jetzt besteht das Objekt quasi aus Millionen von dicht gepackten Streichhölzern, die alle noch die gleiche Höhe haben. Aber nicht mehr lange.

Bildhoch

Die Höhe der einzelnen Stäbe soll aus dem Bild extrahiert werden. Praktischerweise hat unser Grid einen Ausgang für die UV-Map. Füge eine Node Texture > Image Texture hinzu und verbinde den Vector-Eingang mit dem UV Map-Socket der Grid-Node. Füge eine weitere Group Input-Node über Input > Group hinzu. Es handelt sich dabei um eine rein kosmetische Angelegenheit, du kannst so viele Group Input-Nodes hinzufügen, wie du willst. Dadurch werden überlange Node-Verbindungen beziehungsweise Nudeln verhindert. Das kannst du dir zu Nutze machen, indem du auch die Pixel Size neu mit dem X- und Y-Eingang der Combine XYZ-Node verbindest.

Justieren

Es bleibt die Frage, wie wir die Farbinformationen der Pixel am besten in die Höhe der Quader umrechnen. Verbinde den Color-Ausgang der Image Texture Node mit dem Z-Eingang der Combine XYZ-Node. Blender rechnet die Farbwerte der Textur jetzt automatisch in Graustufen um, die wiederum die Höhe der Quader bestimmen. Das schöne daran ist, dass uns jetzt die üblichen Werkzeuge zur Farbmanipulation für den Feinschliff zur Verfügung stehen. Diese findest du unter Utilities > Color. Füge eine RGB Curves-Node hinzu und justiere den Kontrast, bis dir das Ergebnis gefällt. Wenn du die typische S-Kurve zeichnest, werden die glatten Bereiche im Bild noch glatter und die Spitzen noch spitzer.

Die Farbe muss zum Shader

Die Landschaft sieht schon beeindruckend aus, aber es fehlt noch die Farbe. Dieser wird erst im Shader evaluiert und angezeigt. Der einfach Weg wäre, dort nochmals die gleiche Textur zu verwenden und korrekt auf die Blöcke zu mappen. Dadurch wird allerdings die Bedienung des fertigen Assets verkompliziert, da der Nutzer an zwei Stellen das Bild austauschen muss. Alternativ können die Farbinformationen auch von den Geometry Nodes an Cycles übermittelt werden, z.B. indem sie in den Vertices des Ausgabeobjekts gespeichert werden. Dann muss der Nutzer nur noch an einer Stelle das Bild auswählen und es wird einfacher, später die Quader durch andere Objekte auszutauschen, zum Beispiel Stecknadeln.

Materialfrage

Damit die Farbe von Cycles ausgelesen werden kann, werden zwei Dinge benötigt. Zum Einen müssen wir dem Mesh ein Material zuweisen. Dieses wird später auch Teil des Assets. Zum Anderen müssen wir die Farbinformationen im Mesh speichern, um sie nach Außen führen zu können.

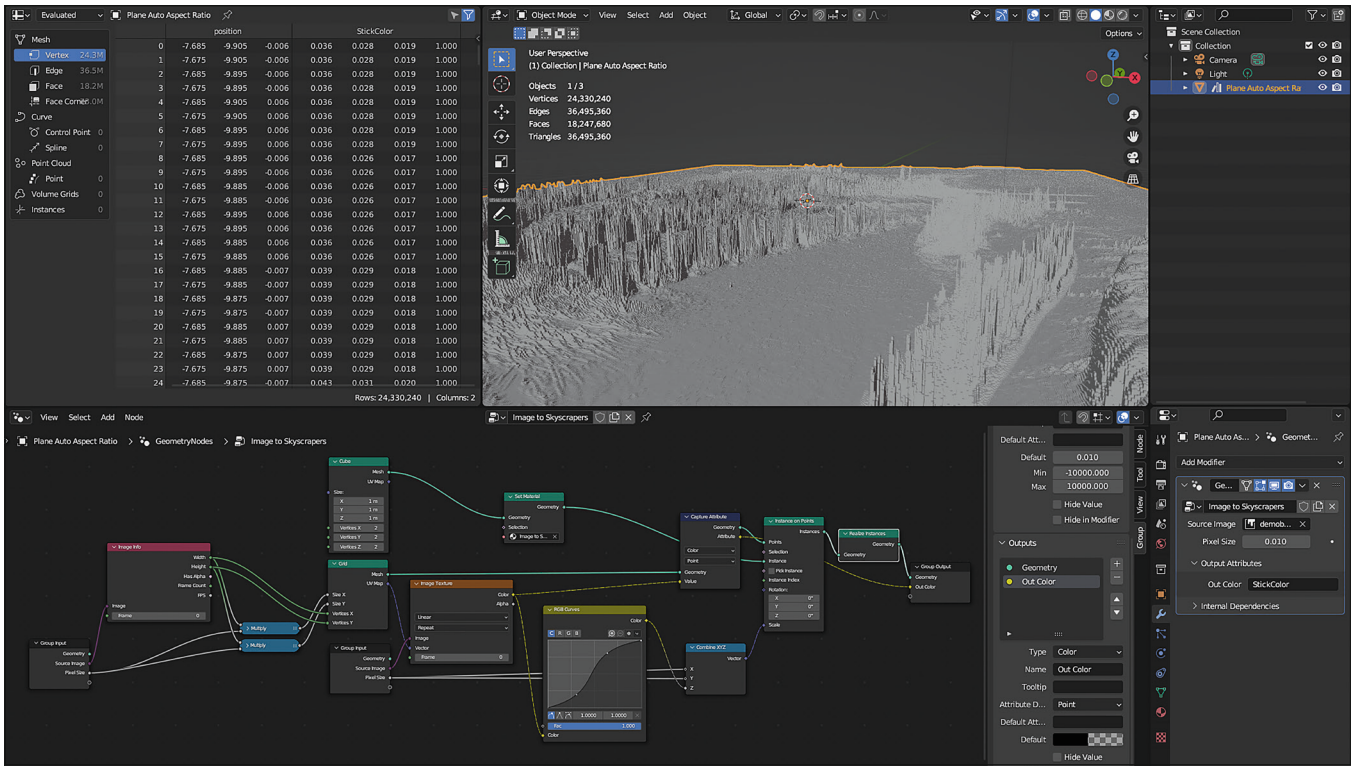
Erzeuge ein neues Material und füge ein Material > Set Material-Node hinzu. Hänge diese zwischen die Cube-Node und die Instance on Points-Node und wähle dort das neu erzeugte Material aus.

Abgreifen

Die Farbe für jeden einzelnen Punkt auf dem Gitter können wir über die Image Texture-Node erhalten, da wir für diese die UV Map des Gitters verwenden. Später verschwinden diese Punkte aber wieder und werden durch Instanzen des Würfels ersetzt. Wir müssen sie also vorher noch abgreifen.

Füge eine Attribute > Capture Attribute-Node hinzu und platziere diese auf der Verbindung zwischen Grid und Instance on Points. Wähle im oberen Drop-Down Color aus. Jetzt erscheint ein gelber Farbsocket, der für jeden Punkt einzeln evaluiert wird. Verbinde ihn mit dem Color-Ausgang der Image Texture-Node. Damit haben wir die Farbe abgegriffen, aber wie kommt sie raus aus den Geometry Nodes und rein in den Shader? Verbinde dafür den Attribute-Ausgang der Capture Attribute-Node mit dem leeren Kreis der Group Output-Node. Jetzt ist im Modifier unter Output Properties ein neues Feld erschienen.

Den Namen kannst du wie bei den Group Inputs über die Sidebar ändern. Der Name, den wir später im Material benötigen, ist



Der (fast finale) Node Tree auf Geometry Nodes-Seite. Obwohl die Farben noch nicht sichtbar sind, kann man ihre Werte schon im Spreadsheet Editor sehen.

derjenige, den du in das Feld schreibst, z.B. StickColor. Sobald du dort einen Namen eingibst, wird automatisch das Attribut erzeugt und erscheint auch sofort im Spreadsheet Editor.

Reinholen

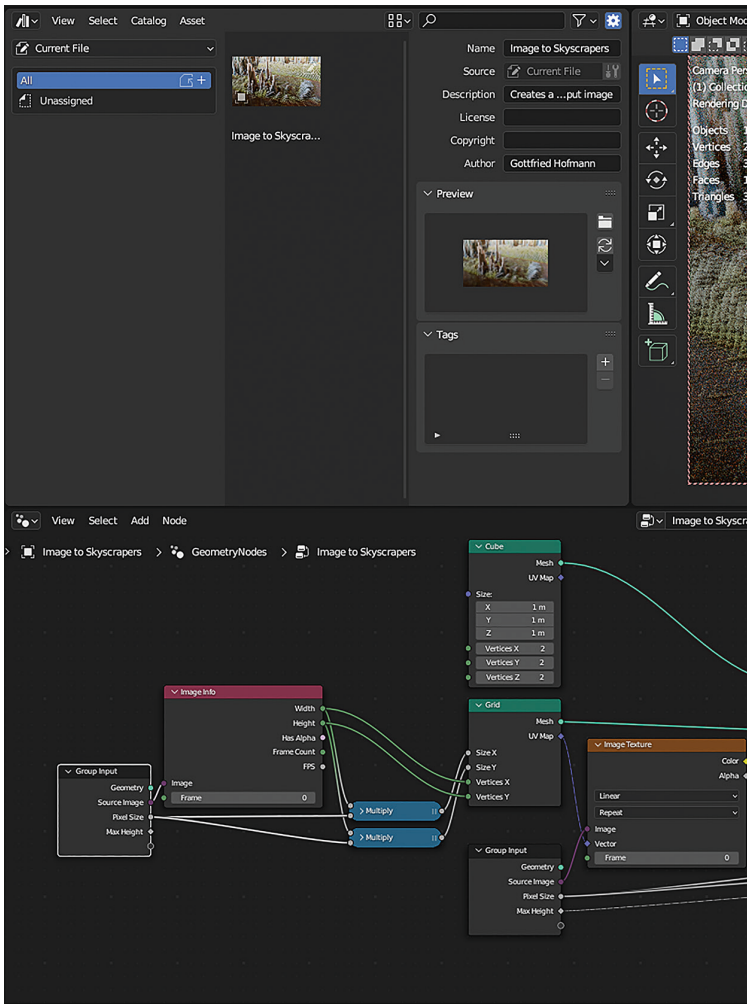
Jetzt müssen wir die Farbe nur noch in den Shader beziehungsweise ins Material holen. Wechsle dazu in den Shading Workspace und lade das Material in den Shader Editor. Füge eine Input > Attribute-Node hinzu und verbinde deren Color-Ausgang mit Base Color und Subsurface Color der Principled BSDF-Node. Als letzten Schritt musst du noch den Namen aus dem Feld im Modifier kopieren und bei der Attribute Node eintragen. Jetzt sollten die Stäbchen bunt und das Ziel erreicht sein.

Look

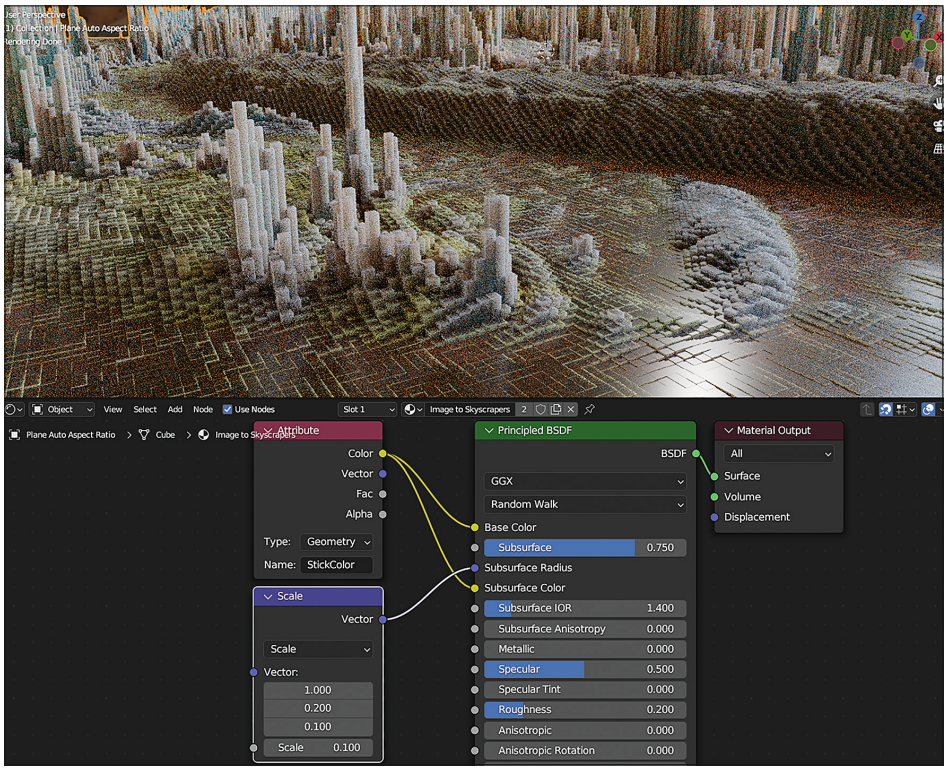
Um den Look wie im Aufhängerbild zu erreichen, wechsle die Renderengine zu Cycles und setzen Sie Subsurface auf 0.75. Um den Subsurface Radius einfach bearbeiten zu können, ziehe eine Verbindung aus dem Socket und suche nach Vector Math Scale. Gib bei Vector die Werte 1.0 0.2 und 0.1 ein. Da die Stäbchen aber nur eine Breite von 0.01 haben, setze die Scale herunter. Hier kannst du ein wenig experimentieren, wir haben uns für den Wert 0.075 entschieden, kleinere Werte lassen das Ergebnis rötlicher und damit fleischiger wirken. In der Principled BSDF haben wir uns für einen Wert von 0.2 entschieden. Beleuchtet wird die Szene von einem HDRI von Polyhaven.

Tweaking

Eigentlich ist unser Asset damit vom Effekt her fertig. Du kannst nun damit Arbeiten und dir Gedanken machen, welche Eigenschaften des Assets du sonst noch von



Das Ergebnis: Unten der vollständige Nodetree, rechts unten das Interface im Modifier Stack, oben links das Asset im Browser mit Metadaten und Vorschaubild.

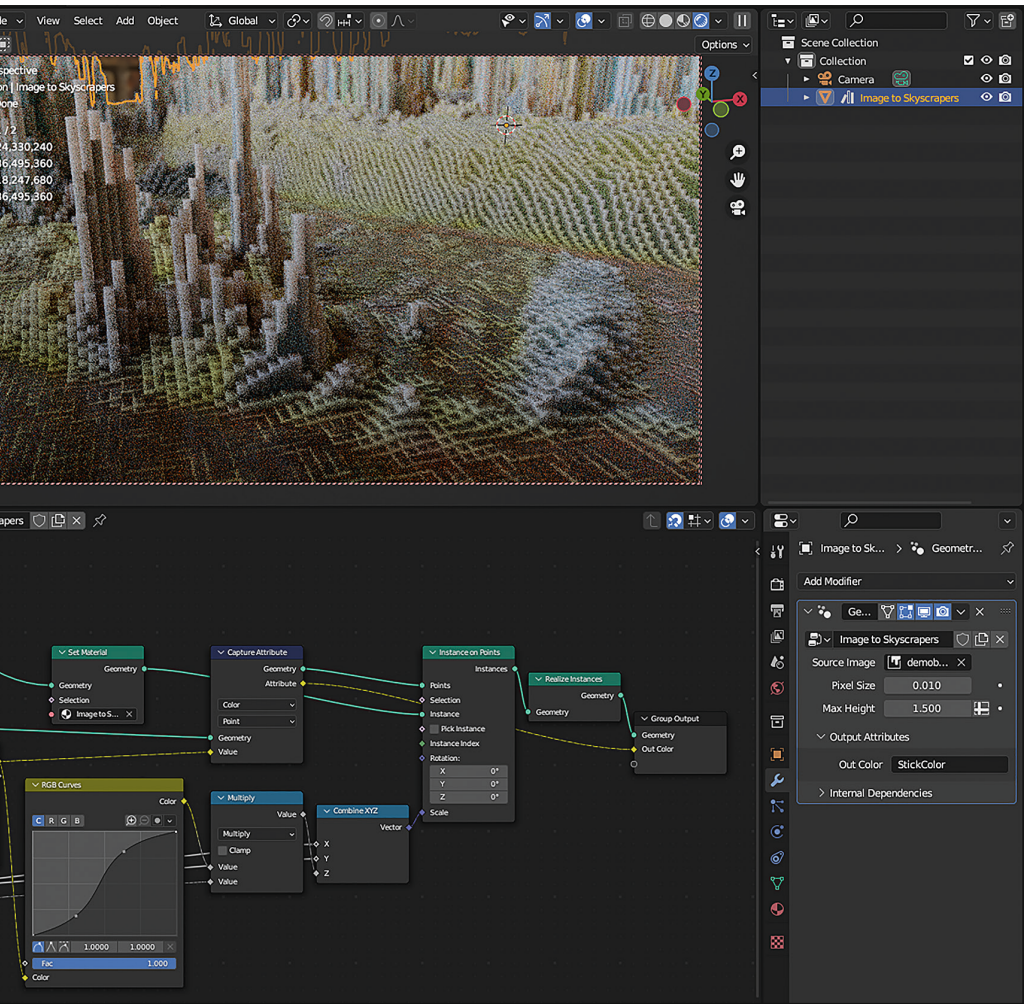


Die Farbe wird aus der Attribute-Node geholt. Der Name muss leider eingegeben werden. Wir empfehlen, ihn direkt aus dem Modifier zu kopieren.
Die Beleuchtung stammt hier aus einem HDRI von Polyhaven.

Außen steuern willst. Ein Multiplikator für die Höhe der Gebäude beziehungsweise Stäbchen wäre zum Beispiel praktisch. Füge dafür im Geometry Nodes-Setup zwischen RGB Curves und Combine XYZ eine Utilities > Math > Math-Node ein und setze diese auf Multiply. Mit dem unteren Wert bei Value kannst du jetzt die maximale Höhe der Türme bestimmen. Setze diese zunächst auf 1.0, damit dieser Wert später der Defaultwert wird, und verbinde den Socket mit dem leeren Kreis bei Group Input. Gib dem Socket in der Sidebar die Bezeichnung Max Height. Für das Demobild fanden wir einen Wert von 1.5 sehr ästethisch.

Finalisierung des Assets

Wenn noch nicht geschehen, markiere das Objekt mit dem Geometry Nodes-Modifier als Asset per Rechtsklick – Mark as Asset. Ändere ein verfügbares Editorfenster zu einem Asset Browser und wähle im Dropdown links oben Current File. Jetzt sollte ein Asset erschienen, das eine graue Ebene als Vorschau hat. Diese Vorschau kannst du durch eine Datei deiner Wahl austauschen, wenn du die Sidebar öffnest und unter Preview auf das Ordnersymbol klickst. Rendere also an dieser Stelle, speichere das Ergebnis und wähle es als Vorschaubild. Jetzt hast du ein vollständiges Asset. Wenn du die Datei in ihrem persönlichen Asset-Ordner speicherst, kannst du von nun an immer über den Asset Browser darauf zugreifen als Teil deiner persönlichen Bibliothek.



Fazit

Mit Geometry Nodes und dem Asset Manager ist Blender jetzt endlich so weit, dass man seine eigenen Werkzeuge ohne Programmierkenntnisse erstellen und paketieren kann, so dass man sie nicht nur wiederverwenden, sondern auch an andere verteilen kann. Dieser Workshop hat den Workflow einer Asseterzeugung mit Geometry Nodes gezeigt. Das Ergebnis kann über den Asset Browser einfach in bestehende Blenderszenen gezogen werden. Je nachdem, welches Bild du darin lädst, sieht das Ergebnis anders aus, aber die Dicke der Klötze und damit die Charakteristik des Subsurface Scattering-Effekts bleiben konstant. Die entstehenden Welten laden zum Verweilen und Entdecken ein. **›ei**

Gottfried Hofmann ist Diplom-Informatiker und bietet seit mehr als einem Jahrzehnt professionellen Support sowie Schulungen für die freie 3D-Software Blender an. Als freischaffender Autor schreibt er für Fach- und Computerzeitschriften und hat zahlreiche Blender-Tutorials für verschiedene Anbieter verfasst. Weiterhin betreibt er die Webseite **www.BlenderDiplom.com**, auf der Blender-Tutorials in deutscher und englischer Sprache zur Verfügung stehen und Schulungen gebucht werden können, und hilft bei der Organisation von BlenderDay und Blender Summer School in Mannheim.