2016
4
ISSN 1433-2620 > B 43362 >> 20. Jahrgang >>> www.digitalproduction.com
Published by ATEC

Deutschland	€ 15,20
Österreich	€ 17,–
Schweiz	sfr 23,–

DIGITAL PRODUCTION

DIGITAL PRODUCTION

MAGAZIN FÜR DIGITALE MEDIENPRODUKTION

JUNI | JULI 04:2016

Zeichnen!

TVPaint, Grease Pencil, After Effects, Clip Studio & mehr ...

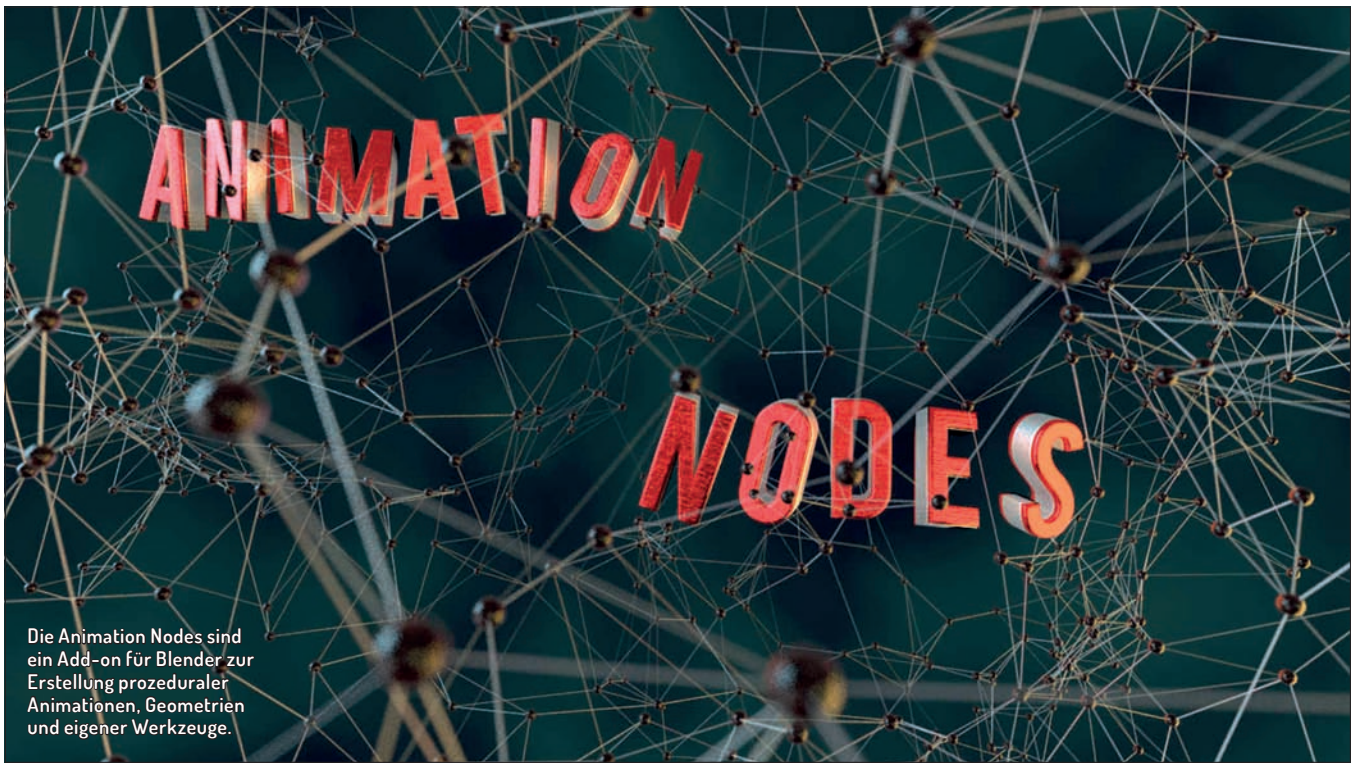
Rendern!

3ds Max 2017, Modo Fur, Laubwerk, Plug-ins & Blender

Filmen!

PTGui, NC3D-Scanner, Shootout, Tracking-Tool

4 194336 215200 04



Die Animation Nodes sind ein Add-on für Blender zur Erstellung prozeduraler Animationen, Geometrien und eigener Werkzeuge.

Blender Animation Nodes

Das Add-on „Animation Nodes" von Jacques Lucke erlaubt die Erzeugung prozeduraler Animationen und Geometrien über Node-Netzwerke, kann als Ersatz für Driver genutzt werden und lässt sich über Python-Skripte erweitern. Blender wurde damit nicht nur ein großartiges Werkzeug für Motion Graphics spendiert, sondern auch die Möglichkeit, eigene Erweiterungen über Nodes zu erstellen. Man muss also nicht mehr Python beherrschen, um Abläufe automatisieren zu können. Digital Production zeigt die Grundlagen anhand von Beispielen aus der Praxis und sprach mit dem Entwickler.

von Gottfried Hofmann

Eines vorweg: Die Animation Nodes sind zwar mächtig, aber Low Level. Mit Ausnahme der Templates gibt es dort keine Ein-Klick-Lösungen, sondern viele kleine Bausteine, aus denen man mit etwas technischem Verständnis ein großes Ganzes schaffen kann. Somit ist die Bedienung noch schwieriger als Mograph in Cinema 4D oder das neue Motion Graphics-System „Mash" in Maya.

Was mit Animation Nodes ebenfalls nicht machbar ist, sind Simulationen, bei denen der Zustand des Systems vom Zustand im vorherigen Frame abhängig ist wie Rigid Body-Simulationen, Partikel-Simulationen mit Newtonscher Physik oder Fluid-Simulationen. Denn die Animation Nodes arbeiten deterministisch. Kein Frame ist von einem anderen Frame abhängig.

Los geht's – die Installation

Herunterladen können Sie das Animation Nodes-Add-on von **https://github.com/JacquesLucke/animation_nodes/releases**. Als dieser Artikel entstand, war Version 1.5 aktuell. Aufgrund der aktiven und schnellen Entwicklung lohnt es sich, auch einen Blick auf die bleeding-edge-Version unter **https://github.com/JacquesLucke/animation_nodes** zu werfen. Dort kann sie über „Download ZIP" bezogen werden.

Installiert werden die Animation Nodes wie andere Blender-Add-ons über den Add-on-Manager, den Sie unter File > User Preferences > Add-ons finden. Wählen Sie dort „Install from File" und klicken Sie doppelt auf die .ZIP-Datei, die Sie heruntergeladen haben. Jetzt erscheinen die Animation Nodes als neuer Punk in der Liste. Ein Klick auf das Kästchen daneben aktiviert sie und mittels „Save User Settings" bleiben sie auch dann aktiviert, wenn Sie Blender neu starten.

Wenn Sie jetzt das Screen-Layout zu „Compositing" wechseln, sehen Sie ein neues Symbol für Node-Typen: Die Animation Nodes. Wenn Sie diese auswählen und über „New" einen neuen Node Tree erzeugen, bleibt das Node-Editor-Fenster leer. Anders als bei Compositing- oder Material-Nodes gibt es keine Standard-Eingänge und -Ausgänge. Die Anwendungsmöglichkeiten sind so vielseitig, dass es einfach keinen Sinn ergäbe, bereits mit Nodes zu starten. Sie würden diese wahrscheinlich sowieso gleich wieder löschen.

Mehr als 200 Nodes?

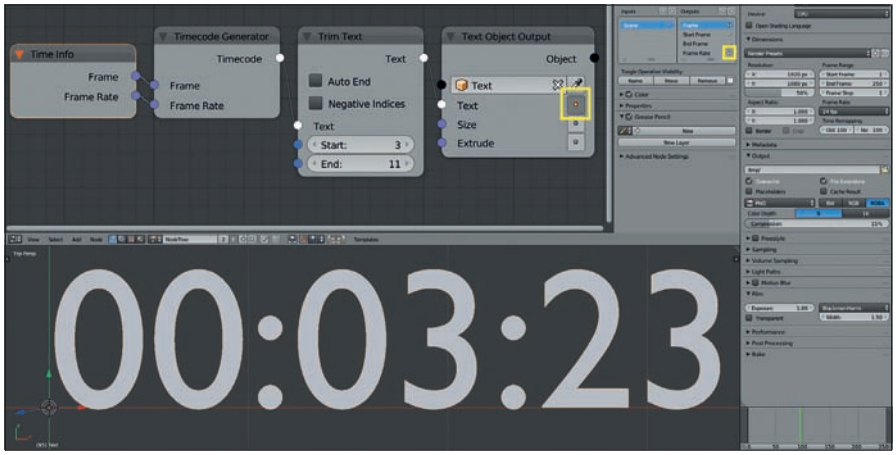
Wenn Sie jetzt mit dem bekannten Shortcut Shift+A eine neue Node hinzufügen wollen, sollten Sie sich nicht von der Länge der Liste und der Tiefe der Unterkategorien schrecken lassen. In den Menüs und Untermenüs tummeln sich bereits mehr als 200 Nodes.

Die große Anzahl rührt daher, dass die Nodes eine Art Interface sind, um auf Blender-Funktionen zuzugreifen und sie ins das Node-Netzwerk integrieren zu können. In diesem Artikel wird eine kleine Auswahl davon vorgestellt zusammen mit grundsätzlichen Bedienkonzepten, um als eine Art Anker zu dienen. Von dort aus können Sie dann selbständig einen Sprung in den Ozean der Animation Nodes wagen.

Klein anfangen

Die allerersten Funktionen, die für die Animation Nodes entwickelt wurden, drehten sich um die Animation von Text. Denn in Blender ist es nur sehr umständlich möglich, einzelne Buchstaben eines Textes zu animieren, sei es um sie auszutauschen oder sie

zu bewegen. Fügen Sie als erstes ein Text-Objekt hinzu und lassen Sie es angewählt. Wenn Sie den Text dieses Objekts jetzt per Animation Nodes verändern wollen, benötigen Sie eine „Text Object Output"-Node. Diese finden Sie unter Shift+A / Text / Object Output oder indem Sie mittels Strg+A direkt danach suchen. In der Node befindet sich ein Dropdown-Feld, wo Sie das gerade hinzugefügte Text-Objekt auswählen können. Für Szenen mit sehr vielen Objekten ist diese Methode aber recht umständlich. Alternativ können Sie auch auf das Pipetten-Symbol daneben klicken. Dann wird das aktuell aktive Objekt eingetragen.



Time Code Display – Der fertige Node Tree des ersten Beispiels. Bei der Text Object Output-Node wurde der Text-Socket durch einen Klick auf den Knopf mit dem grauen Punkt aktiviert (erste Hervorhebung). Bei der Time Info-Node wurde der Frame Rate-Socket durch einen Klick auf das Auge im Properties Panel der Node sichtbar gemacht (zweite Hervorhebung).

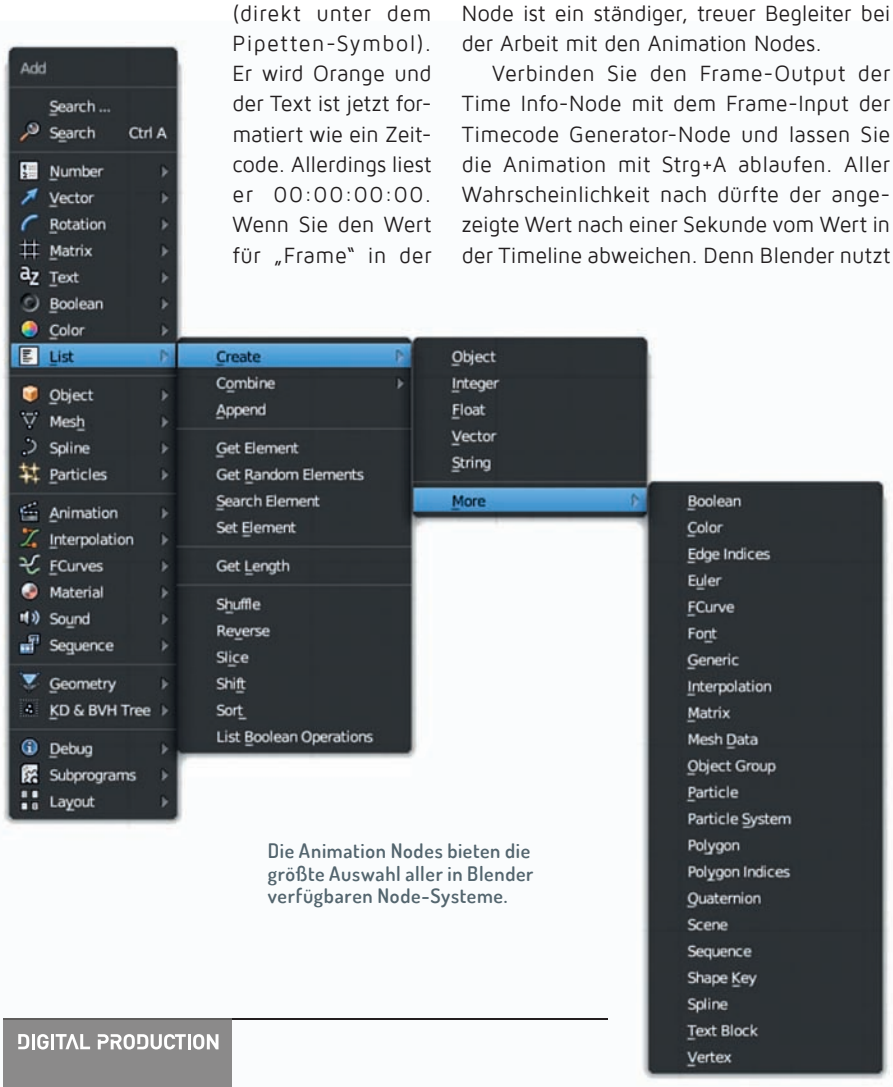
High Level Nodes – Es gibt sie doch!

Die Node hat einen weißen Eingang, an dem „Text" steht. Darüber können Sie die Buchstaben des Objektes durch andere austauschen. Als Quelle soll für den Anfang eine der wenigen High Level-Nodes dienen: Timecode Generator (Shift+A/ Text/ Timecode Generator). Fügen Sie die Node hinzu und verbinden Sie den weißen Ausgang mit dem weißen Eingang der Text Object Output-Node. Der Text hat sich nicht verändert. Der Grund dafür ist, dass der Eingang noch nicht aktiviert wurde. Klicken Sie auf den grauen Punkt rechts vom Text-Eingang (direkt unter dem Pipetten-Symbol). Er wird Orange und der Text ist jetzt formatiert wie ein Zeitcode. Allerdings liest er 00:00:00:00. Wenn Sie den Wert für „Frame" in der Timecode Generator-Node ändern, ändert sich auch der Text.

Animation über Nodes statt über Keyframes

Sie könnten diesen Wert jetzt über Keyframes animieren, eleganter ist bei Animation Nodes aber der Weg über eine „Time Info"-Node (Shift+A/ Animation/ Time Info), da in Animation Nodes gesetzte Keyframes z.B. nicht im Graph Editor auftauchen und man daher wenig Kontrolle über sie hat. Die Time Info-Node ist ein ständiger, treuer Begleiter bei der Arbeit mit den Animation Nodes.

Verbinden Sie den Frame-Output der Time Info-Node mit dem Frame-Input der Timecode Generator-Node und lassen Sie die Animation mit Strg+A ablaufen. Aller Wahrscheinlichkeit nach dürfte der angezeigte Wert nach einer Sekunde vom Wert in der Timeline abweichen. Denn Blender nutzt standardmäßig 24 Bilder pro Sekunde. Sie könnten diesen Wert direkt bei „Frame Rate" eintragen. Nehmen wir jetzt aber mal den hypothetischen Fall an, dass Sie sich noch nicht endgültig für eine Framerate ihres Projekts entschieden haben oder das Projekt in verschiedenen Framerates ausgeben wollen und dann natürlich für jede Bildwiederholrate ein anderes Overlay bräuchten.



Die Animation Nodes bieten die größte Auswahl aller in Blender verfügbaren Node-Systeme.

Versteckte Bonus-Sockets

Hier hilft ein geheimer Socket der „Frame Info"-Node weiter. Wenn Sie diese anklicken und mit der Taste „N" das Properties Panel im Node Editor öffnen, sehen Sie eine Liste von Eingängen und Ausgängen. Das ist natürlich erst einmal merkwürdig, denn die Node hat ja gar keinen Eingang und nur einen Ausgang (und nicht vier, wie in der Liste).

Des Rätsels Lösung: Neben jedem Ein- und Ausgang ist ein Augen-Symbol, das die Sichtbarkeit anzeigt. Von den insgesamt fünf Sockets ist nur einer aktiviert. Klicken Sie auf das Auge neben „Frame Rate". Jetzt erscheint ein neuer Output Socket an der Time Info Node. Verbinden Sie diesen mit dem Frame Rate-Eingang der Timecode Generator-Node. Ab jetzt wird jede Änderung der Bildwiederholrate in ihrem Blender-Projekt im generierten Timecode abgebildet. Bei vielen sind selten verwendete Sockets versteckt. Es lohnt sich also, immer mal wieder einen Blick ins Properties Panel zu werfen.

Texte bearbeiten

Die Timecode-Anzeige könnte je nach Projekt ein wenig lang sein. Schließlich werden nicht nur Einzelbilder, Sekunden, Minuten und Stunden, sondern sogar Tage angezeigt. Wir wollen die Anzeige auf Stunden, Minuten, Sekunden und Frames reduzieren. Bei mehr als 200 Nodes ist zu erwarten, dass für diese Aufgabe ein Werkzeug zur Verfü-

Interview mit dem Animation Nodes Entwickler

Jacques Lucke studiert Informatik an der TU Berlin und ist der Erfinder und Hauptentwickler des Add-ons Animation Nodes für Blender. Digital Production sprach mit ihm über die neuen Entwicklungen und welche Pläne er für die Zukunft des Add-ons hat.

DP: Wie bist du auf Blender gestoßen?
Jacques Lucke: Das erste Mal, dass ich Blender geöffnet hatte, war, als ich 3D-Modelle für meine ersten Gehversuche in der 3D-Spieleprogrammierung brauchte. Ich bin damit jedoch nie wirklich weit gekommen, da ich mich seit dem Moment sehr viel mehr für 3D, Animation und VFX interessiert habe. Diese Leidenschaft wurde von meinem kurzen Praktikum im Jahr 2011 bei Rise FX in Berlin noch gestärkt. Selbst aktiv als Entwickler von Blender-Add-ons wurde ich Mitte 2014.

DP: An welchen Projekten hast du vor den Animation Nodes gearbeitet?
Jacques Lucke: Ich habe, wie schon erwähnt, vorher schon außerhalb von Blender programmiert. Nachdem ich in die Blender-Python-Entwicklung eingestiegen war, ging daher eigentlich alles ziemlich schnell. Mein erstes Add-on war „Sniper", welches vergleichbar zur dem „Sure Target"-Plug-in für After Effects von Video Copilot ist. Die Reaktionen auf das Add-on waren alle sehr positiv und haben mich motiviert, weiter zu machen. Neben mehreren kleinen Tests war Animation Nodes schon das nächste große Projekt von mir, an dem ich jetzt seit nunmehr eineinhalb Jahren arbeite. In der Zwischenzeit habe ich noch das „Code Autocomplete" Add-on entwickelt, welches die Python-Entwicklung in Blenders Text Editor durch die Bereitstellung von Templates und einer Autovervollständigung deutlich vereinfacht.

DP: Wie lief bisher die Zusammenarbeit mit anderen Entwicklern?
Jacques Lucke: Da ich vorher noch nie mit anderen Entwicklern zusammen gearbeitet hatte, musste ich das auch erst lernen. Mittlerweile funktioniert das jedoch sehr gut. Wir diskutieren viel über verschiedene Themen auf Github und es gab mittlerweile ca. 100 Pull-Requests von 13 anderen Programmierern.

DP: Planst du AN als offizielles Blender-Add-on einzureichen?
Jacques Lucke: Nein. Zumindest noch nicht in naher Zukunft, aus mehreren Gründen: Es würde sehr viel mehr Verantwortung für mich bedeuten als es jetzt schon ist. Zudem wäre es schwieriger, verschiedene Ideen auszuprobieren und auch wieder zu verwerfen. Ein weiterer Grund ist, dass ich den Usern erlauben will, das Animation Nodes-Add-on unabhängig von Blender zu updaten. Zu guter Letzt müsste ich noch viel Dokumentation schreiben, bevor es zugelassen werden würde. Für diese Mammutaufgabe kann ich zurzeit leider nicht die Zeit aufbringen. Ich sehe für niemanden einen echten Vorteil, wenn AN ein offizielles Blender Add-on wird.

DP: Welcher Ansatz wäre das?
Jacques Lucke: Animation Nodes ist, genau wie Blender, ein Open Source Projekt. Für mich bedeutet das, dass ich nicht nur für Artists entwickle sondern auf für andere Programmierer oder solche, die es werden wollen. Dementsprechend sehe ich in AN nicht nur ein Tool zum Erstellen von Animationen, sondern auch ein Tool, mit dem andere Tools entwickelt werden können, ohne sich jedes Mal auf unwichtige Einzelheiten konzentrieren zu müssen, die die Add-on-Entwicklung mit sich bringt. Dafür braucht es größtmögliche Flexibilät, was der Grund dafür ist, dass AN einer Programmiersprache sehr ähnlich ist. Zudem ist es für mich einfacher, zuerst sehr viele Low-Level-Nodes zu entwickeln und mich später auf High-Level-Nodes zu konzentrieren als anders herum.

DP: Was war der interessanteste Einsatzzweck für AN, den du bisher mitbekommen hast?
Jacques Lucke: Das ist eine schwierige Frage. Besonders interessant sind eigentlich immer die Sachen, für die AN noch nicht konzipiert ist. Es hat zum Beispiel mal jemand eine Musikanimation gemacht, wo die Musik die Größe und Farbe von anderen Nodes gesteuert hat. Das fand ich eine lustige Idee.

DP: Was sind deine Pläne für die Zukunft von Animation Nodes?
Jacques Lucke: Neben etlichen kleinen habe ich noch zwei große Ideen, die ich in die Tat umsetzen will. Das erste ist, dass ich Cython Code in den jetzigen Python Code integrieren möchte. Dieser erlaubt es, die performancelastigen Nodes mit einer deutlich höheren Geschwindigkeit auszuführen und öffnet neue Türen. Beispielsweise kann man ernsthafter in prozeduale Meshes und Image Processing einsteigen. Der zweite Plan ist die Implementierung einer Plug-in-Schnittstelle innerhalb des Animation Nodes-Add-ons. Diese soll es Entwicklern erlauben, ihre Nodes sehr einfach zu verbreiten, ohne sie in die offizielle AN-Version einzubauen. Das bringt Vorteile für alle. Ich muss weniger Nodes entwickeln und bin auch nicht für den Code anderer verantwortlich. Andere Entwickler können das AN-Framework unabhängig verwenden. Die User dürfen sich über weitere Nodes freuen, die man leicht installieren kann. Meine Hoffnung ist, dass es dadurch mehr High-Level-Nodes geben wird, die es Usern erlauben, bestimmte Effekte schneller zu erreichen.

gung steht. Fügen Sie eine „Trim Text"-Node (Shift+A / Text/ Trim) zwischen der Timecode Generator- und der Text Object Output-Node ein. Setzen Sie „Start" auf 3 und „End" auf 11. Die Anzeige der Tage ist verschwunden und das erste Beispiel damit fertig.

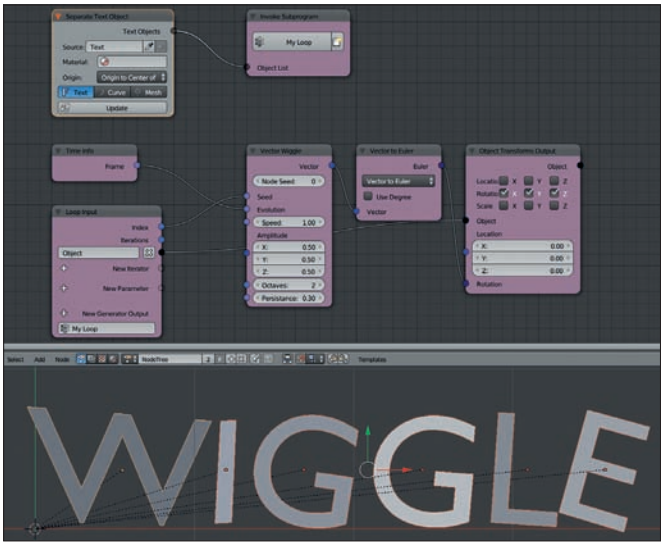
Natürlich hat man mit den Animation Nodes noch viele andere Möglichkeiten, Text zu erzeugen und zu bearbeiten. Ein Beispiel dafür sind Zufallszahlen, worüber auf BlenderDiplom ein Tutorial veröffentlicht wurde (**http://www.blenderdiplom.com/de/tutorials/605-animation-nodes-zufallszahlen-erklaert.html**).

Wiggle wiggle wiggle

Bleiben wir bei Text-Animationen. Erstellen Sie eine neue Szene und fügen Sie ein Text-Objekt hinzu. Erstellen Sie einen neuen Node Tree in AN. Fügen Sie eine „Object Transforms Output"-Node hinzu (Shift+A/ Object/ Transforms Output). Diese Node wird immer dann benötigt, wenn über Animation Nodes ein Objekt bewegt werden soll. Setzen Sie über die Pipette das Text-Objekt in das Drop-Down-Feld und setzen Sie Häkchen bei den drei Auswahl-Feldern hinter „Location". Die drei Felder stehen für die X-, Y- und Z-Achse. Nur ausgewählte Achsen werden von der Node beeinflusst. Jetzt ist auch ein neuer Socket erschienen und Sie können die Position des Objektes jetzt verändern, indem Sie die Werte in den Feldern beim Eingang ändern.

Was Sie jetzt benötigen, ist eine Quelle für den Eingang. Nutzen Sie dafür eine „Vector Wiggle"-Node (Shift+A/ Vector/ Wiggle) und verbinden Sie diese mit dem Location-Eingang. Wenn Sie den Wert für „Evolution" in der Vector Wiggle-Node ändern, bewegt sich jetzt entsprechend das Text-Objekt. Als Eingang für den Evolution-Socket bietet sich

Das Node-Setup für eine einfache Wiggle-Bewegung eines Objektes



Wiggle individuell – Das Node-Setup, um die einzelnen Buchstaben eines Text-Objekts wiggeln zu lassen.

wieder der Frame-Ausgang der Time Info-Node an. Über „Amplitude" können Sie die Stärke des Ausschlags kontrollieren und über „Speed" die Geschwindigkeit.

Nutzern von After Effects dürfte die Bezeichnung Wiggle geläufig sein. Damit wird dort eine Funktion bezeichnet, die eine zufällig hin- und herschwingende Animation erzeugt. Das gleiche gibt es auch in den Bordmitteln von Blender in Form des F-Curve-Modifiers „Noise". Dort arbeiten die gleichen Algorithmen wie in der Wiggle-Funktion in After Effects und in der Wiggle-Node, nämlich Perlin Noise.

Die Wiggle-Nodes der Animation Nodes haben einige Vorteile. So muss beim Noise-Modifier für jeden Kanal ein eigenständiger Modifier erstellt werden. Und diese müssen auch jeweils einzeln bearbeitet werden, wenn man etwas ändern möchte. Bei den Animation Nodes gibt es hingegen neben dem normalen Wiggle auch ein Wiggle für Vektoren – was drei Kanälen gleichzeitig entspricht – und sogar für Quaternionen. Und wichtiger noch: Die Bewegung lässt sich sehr gut steuern, während beim Noise-Modifier gerade mal ein Fade-In und Fade-Out möglich ist.

Buchstaben individuell animieren

Motion Graphics beinhalten häufig eine Vielzahl von Elementen, die sich einzeln bewegen und dabei einer übergeordneten Kontrolle unterliegen. Ein Beispiel wären Buchstaben, die der Reihe nach einfliegen und einen Schriftzug bilden. Die Grundlagen hierfür werden wir im nächsten Schritt erstellen, wobei die Bewegung der Buchstaben der Einfachheit halber auf ein Hin- und Herwackeln beschränkt bleiben. Im vorhergehenden Beispiel hat der gesamte Text gewackelt, jetzt wenden wir den Effekt auf einzelne Buchstaben an, und zwar auf die Rotation. Dafür müssen wir den Text zunächst in die besagten einzelnen Buchstaben aufbrechen. Fügen Sie dem Wiggle-Beispiel eine „Separate Text Object"-Node hinzu (Shift+A/ Text/ Object Separate). Wählen Sie bei „Source" das Text-Objekt und wählen Sie bei „Origin" die Einstellung „Origin to Center of Mass", damit sich die Buchstaben später um ihren jeweiligen Schwerpunkt drehen.

Text aufgebrochen, Original versteckt, was nun?

Wenn Sie jetzt auf „Update" klicken, ist der Text aufgebrochen und jeder Buchstabe ein eigenes Text-Objekt mit Ursprung in seinem Schwerpunkt. Der Text wackelt auch nicht mehr, wenn Sie die Animation abspielen, denn das ursprüngliche Text-Objekt wurde versteckt. Um die einzelnen Buchstaben wackeln zu lassen, benötigen Sie eine Schleife oder einen Loop, der sämtliche Objekte abarbeitet. Schleifen werden bei den Animation Nodes über sogenannte „Subprograms" zur Verfügung gestellt. Diese werden dann wiederum von einer eigenen Node, die sich „Invoke Subprogram" nennt, aufgerufen. Fügen Sie diese Node hinzu (Shift+A/ Subprograms/ Invoke Subprogram). Klicken Sie nun auf „New Subprogram" und wählen Sie in der erscheinenden Liste „Loop" aus.

Es erscheint eine neue Node „Loop Input". Diese hat bereits zwei Ausgänge „Index" und „Iterations". Wir werden später den „Index" benötigen, denn der bezeichnet die aktuelle Position in der Schleife – in unserem Fall, bei welchem Buchstaben wir gerade sind. Darunter befinden sich leere Sockets. Diese müssen erst noch verbunden werden, bevor wir mit der Schleife arbeiten können. Verbinden Sie den Object-Eingang der Object Transforms Output-Node mit dem „New Iterator"-Eingang der Loop Input-Node. Nun ist bei der Invoke Subprogram-Node ein neuer Eingang entstanden: „Object List". Verbinden Sie diesen mit dem „Text Objects"-

Lorenz-Attraktor Robert Schütze hat in Animation Nodes einen Lorenz-Attraktor implementiert. Mit dessen Hilfe entstand dieses Bild eines aufgerollten Netzwerkkabels.

Dass man die Animation Nodes auch für statische Bilder nutzen kann, zeigt dieser Gefäß-Kopf von Benny Govaerts.

Ausgang der Seperate Text Object-Node. Wenn Sie jetzt die Animation abspielen, so befinden sich alle Buchstaben an derselben Stelle und bewegen sich hin- und her.

Die Liste abarbeiten

Wie funktioniert das gerade erstellte Setup? Die Separate Text Object-Node erstellt eine Liste von einzelnen Text-Objekten, eines für jeden Buchstaben. Indem wir den Object-Eingang der Object Transforms Output-Node als Iterator für die Schleife definiert haben und dort diese Liste hinschicken (über die Invoke Subprogram-Node), arbeitet die Schleife alle diese Objekte ab. Sie wackeln zwar, landen aber auch alle an der gleichen Position, denn der Vector Wiggle wird jedes mal mit den gleichen Parametern aufgerufen.

Um das zu ändern, verbinden Sie den Index-Ausgang der Loop Input-Node mit dem Seed-Eingang der Vector Wiggle-Node. Jetzt bewegen sich die Buchstaben alle unterschiedlich, denn der erste bekommt den Seed 1, der zweite den Seed 2 usw.

In der Mitte ein Haufen

Sie bewegen sich aber alle noch um den Ursprung herum. Das liegt daran, dass die Object Transforms Output-Node die ursprüngliche Position eines Objekts komplett überschreibt. Wir könnten jetzt die ursprüngliche Position vorher auslesen und dann hinzufügen, das würde aber den Rahmen hier sprengen. Stattdessen werden wir einfach die Rotation statt der Position überschreiben.

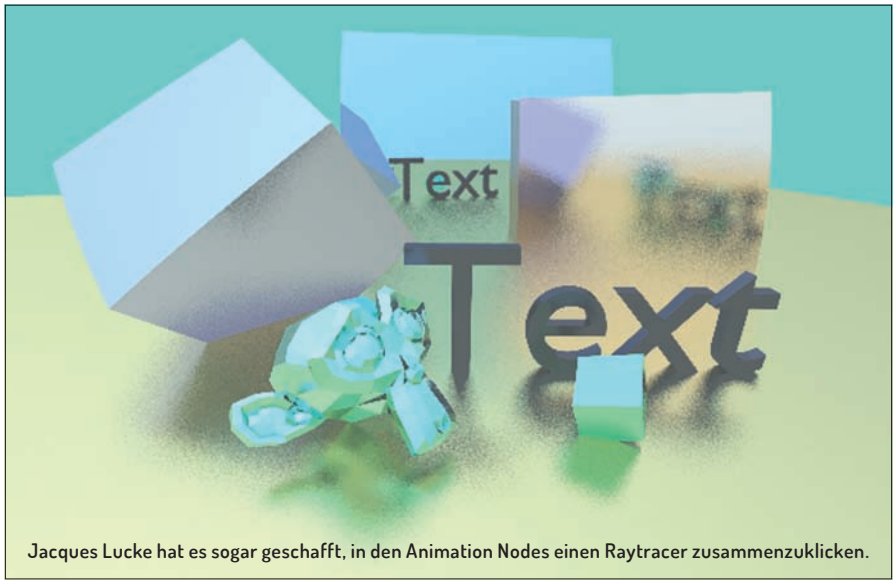
Wählen Sie in der Object Transforms Output-Node alle Häkchen bei Location ab, und dafür alle bei Rotation an. Verbinden Sie jetzt den Vector-Ausgang der Vector Wiggle-Node mit dem Rotation-Eingang. Das Animation Nodes-Add-on sollte jetzt automatisch eine Node zwischengeschaltet haben, die von Vector zu Euler konvertiert. Diese Konversionen werden von den Animation Nodes automatisch erstellt, was einiges an Arbeit spart. Die Buchstaben rotieren jetzt und wackeln dabei um ihren Ursprung, befinden sich aber immer noch auf einem Haufen.

Der Grund dafür liegt darin, dass die Separate Text Object-Node aus Performance-Gründen nicht automatisch updated (Text-Objekte in Blender sind leider eine aufwändige Sache). Ein Klick auf „Update" und die Buchstaben befinden sich da, wo sie hingehören und wackeln vor sich hin.

Eine Frage bleibt allerdings: Wie kann man den Text jetzt bearbeiten? Hierfür hilft ein Blick in den Outliner. Das ursprüngliche Text-Objekt wurde von den Animation Nodes nämlich versteckt. Sie können es auch ohne Outliner wieder sichtbar machen, indem Sie auf das Augen-Symbol der Separate Text Object-Node klicken. Dann können Sie es bearbeiten und auf „Update" klicken. Und schon wackelt Ihr neuer Text.

Fazit

Das Animation Nodes-Add-on erweitert den Spielraum von Blender deutlich. Sie liefern fehlende Funktionen nach, erlauben die prozedurale Erstellung von Animationen und Geometrien und können als Baukasten für eigene Werkzeuge genutzt werden. Beim Arbeiten mit Animation Nodes sind ein paar Eigenheiten zu beachten, die in diesem Workshop vorgestellt wurden. Wer das volle Potential der Animation Nodes erfahren möchte, sollte einen Blick auf das offizielle Animation Nodes Showreel 2015 werfen: **https://www.youtube.com/watch?v=hnlUhlkxQZU** **› ei**



Jacques Lucke hat es sogar geschafft, in den Animation Nodes einen Raytracer zusammenzuklicken.

Gottfried Hofmann ist Diplom-Informatiker. Er arbeitet als Freelancer in den Bereichen Visualisierung und VFX sowie als Trainer und Consultant für die freie 3D-Software Blender. Als freischaffender Autor schreibt er für Fach- und Computerzeitschriften. Er hat zahlreiche Blender-Tutorials verfasst, u.a. für CG Tuts+ und CG Cookie. Weiterhin betreibt er die Webseite **www.BlenderDiplom.com**, auf der Blender-Tutorials in deutscher und englischer Sprache zur Verfügung stehen.

HÖCHSTLEISTUNG HAT EIN NEUES GESICHT…

SHUTTLE® XPC CUBE SZ170R8

Mini-PC-Barebone mit Intel Z170 Chipsatz.

■ Bis zu 64 GB DDR4-Speicher ■ Unterstützt neue LGA1151 CPUs ■ M.2-Steckplatz für NVMe-SSDs ■ Platz für vier Festplatten
■ Verbessertes Kühlsystem ■ Starkes 500-Watt-Netzteil ■ Nur 33,2 x 21,6 x 19,8 cm (TBH) ■ Intel Gigabit Ethernet
■ 2x DisplayPort, 1x HDMI

Vier Befestigungen für 3,5"-Laufwerke erlauben den Einbau großer Festplatten (zusammen bis aktuell 40 TB).

Dieses Modell ist auch für moderne, große Dual-Slot-Grafikkarten geeignet (z.B. NVIDIA® Quadro® M5000).

www.shuttle.eu

Shuttle®