SpeedTree in der Spieleproduktion

POWERED BY SAE INSTITUTE

Die Erstellung von Vegetationen kann ein kritisches Element in der Produktion eines Games darstellen. Und in der Regel ist schon am Anfang eines Projektes klar, welchen Stellenwert die Vegetation in dem Spiel einnehmen wird. In vielen kleineren Spielen, 2D-Produktionen oder Sci-Fi-Titeln sind die Vegetation Assets rar gesät und schnell durchproduziert. Entwickelt man aber ein technisch vielschichtiges Produkt beispielsweise mit einem Open-World-Szenario, hat man schnell einen erweiterten Feature-Bedarf, den man abdecken möchte.

von Sascha Henrichs

Warum brauchen wir SpeedTree? Nehmen wir an, wir haben ein realistisches Naturszenario in einer mit Schrittgeschwindigkeit begehbaren 3D-Welt in einer First-Person-Ansicht. Zielplattform ist die Xbox360. Hierfür wollen wir eine üppige Anzahl von verschiedenen Vegetation Assets produzieren. Einige Büsche und ein paar Bäume verschiedener Art, jeweils in verschiedenen Größen und Formen, diverse Wurzeln und Hängepflanzen, Bodenbewuchs und zudem einige "Hero"-Bäume, die speziell für Schlüsselszenen produziert werden.

Die Assets müssen alle sehr detailliert sein, da man sie in der First-Person-Ansicht betrachtet. Gleichzeitig müssen sie performant dargestellt werden, also auch ein "Level of Detail"-System (LOD) unterstützen, und ressourcenschonende Dateigrößen und -anzahlen haben. Zusätzlich soll sich die Vegetation noch im Wind wiegen. Diese Fülle an Attributen, die die Vegetationslösung nun bearbeiten soll, lässt auf Anhieb einen immensen Arbeits- und Pflegeaufwand erkennen. Somit muss man sich auch gut überlegen, wie man diese Aufgabe plant und schließlich löst.

Früher wurden diese Assets ausnahmslos von Hand modelliert und mit eigenen Texturen versehen. So wird fast jeder 3D-Grafiker schon einmal einen Baum in seiner 3D-Software von Hand modelliert haben. Es ist ein sehr aufwendiger Prozess, und wenn man später noch Veränderungen an den Bäumen vornehmen muss, so sind diese ebenfalls sehr zeitaufwendig. Also allein die Herstellung der Modelle wäre ein großes Unterfangen und für jede Grafikabteilung eine langwierige und demotivierende Arbeit.

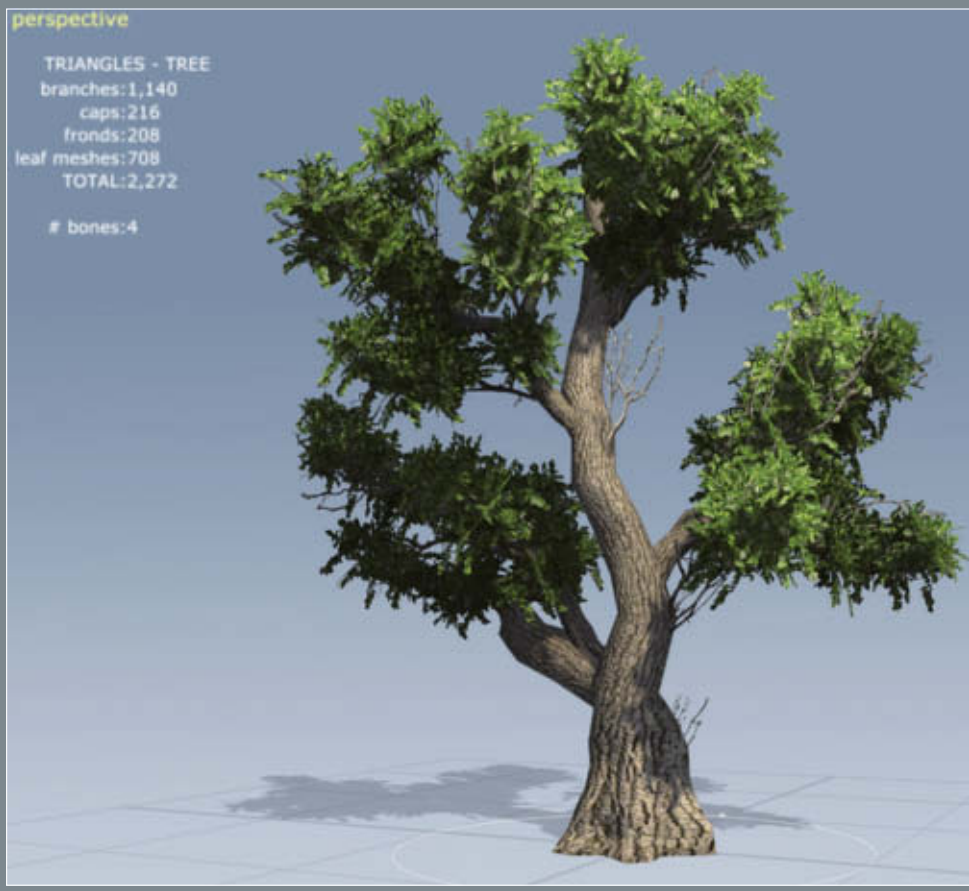
Gleichzeitig müsste man die unterschiedlichen LOD-Stufen jederzeit nachpflegen und ein entsprechendes System für die Darstellung eben jener in die Engine integrieren. Man müsste ein Programmmodul programmieren welches den Wind erstellt, und eventuell sieht die Planung auch Interaktion mit der Vegetation vor, womit dann in der Regel auch ein Skelettsystem für die Bäume programmiert werden muss.

Allein bei der Planung und dem Prototyping eines solchen Systems mit so vielen Kompo-

nenten, gehen Wochen ins Land. Kurz gesagt, wird ein Programmierer – als Vollzeitkraft – für diese Arbeit eine sehr lange Zeit abgestellt sein. Gleichzeitig wird dieses System in der Anfangszeit nur so vor Programmfehlern strotzen, so dass die Datenersteller zu Alpha- und Beta-Testern werden und kaum ungehindert Daten erstellen können.

Spätestens jetzt muss man sich fragen, ob die Ressourcen und das Budget für eine solche Inhouse-Lösung mit eigener Engine-Implementation zur Verfügung stehen oder ob man auf eine Middleware zurückgreifen sollte. Also auf ein fertig programmiertes Engine-Modul, welches sich eben nur um diese Aufgaben kümmert.

Eine sehr beliebte Middleware in diesem Zusammenhang ist natürlich SpeedTree von der Firma IDV (Interactive Data Visualization, Inc). In vielen aktuellen Spieletiteln wird SpeedTree auch mittlerweile eingesetzt, um all der oben genannten Probleme mit einem Schlag Herr zu werden: „Battlefield 3", „Batman: Arkham Asylum", „Witcher 2" oder zum Beispiel unser neuestes Spiel „Risen 2", um nur einige zu nennen.



Der fertige Baum

Das SpeedTree-Paket

Im Folgenden möchte ich nun die Arbeit mit SpeedTree erläutern und einen typischen Workflow vorstellen. Doch zunächst eine kurze Einleitung, was SpeedTree überhaupt ist. SpeedTree gibt es in verschiedenen Produktversionen: SpeedTree Studio, SpeedTree Cinema und SpeedTree for Games. Studio ist die günstigste Variante mit derzeit etwa 900 Dollar. Diese Version kommt zum Beispiel ohne die Baumbibliothek, LOD-Mechanismus und der Floating License aus, womit man den Modeler auf beliebig vielen Rechnern installieren kann.

Die Version SpeedTree Cinema, hat die Bibliothek und zum Beispiel auch Floating Licences und kostet derzeit circa 5.000 Dollar. Hierbei sei kurz darauf hingewiesen, dass James Cameron SpeedTree als Vegetationslösung für seinen Film "Avatar" ausgesucht hat, noch bevor es SpeedTree Cinema gab. Infolge der Entwicklung des Films wurde SpeedTree dann letztlich auf diesen Produktbereich angepasst und vertrieben.

Mittlerweile bietet sogar die Gnomon School of Visual Effects in Hollywood eine eigene Schulung für SpeedTree an, was noch einmal den Stellenwert unterstreicht, den SpeedTree derzeit auch im Film-Business genießt.

Das letzte und in der Regel teuerste Produkt ist SpeedTree for Games. Es hat alle oben genannten Features und noch einige mehr. Um einen Preis dafür zu erfahren, muss man sich mit IDV in Verbindung setzten, dann werden die Konditionen verhandelt. In der Regel bezieht sich der zu zahlende Betrag auf die Entwicklung von genau einem Spieletitel und kann in der Höhe variieren. Hat man letztlich eine Version käuflich erworben, bekommt man einen Zugang zu einem Download-Bereich auf den Servern von IDV und kann sich sein Produkt herunterladen.

SpeedTree for Games

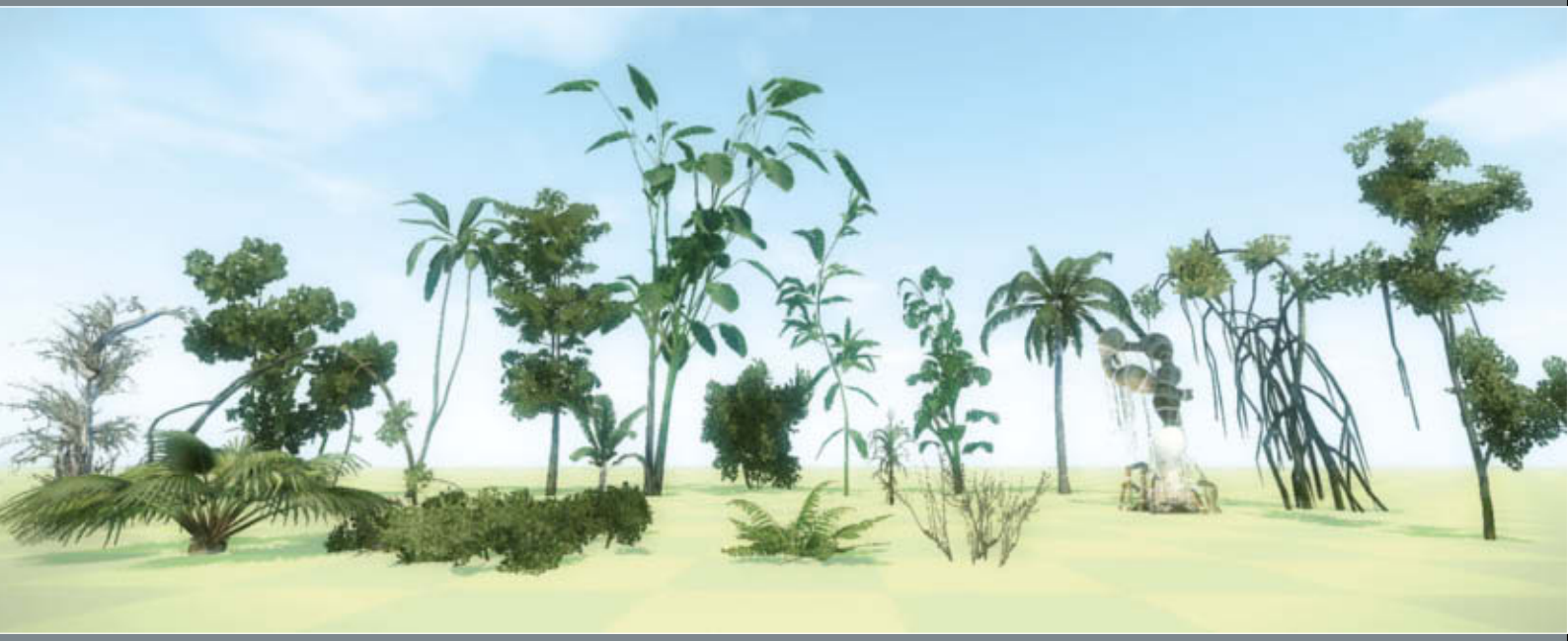
Das Produkt SpeedTree for Games wird mit einem prozeduralen Tree Modeler, einer umfassenden Baumsammlung, dem SpeedTree Compiler und dem SDK (Software Development Kit) ausgeliefert. Für den 3D-Artist ist die Installation des Modelers mit circa 250 MB relativ schlank, die Baumbibliothek ist der weitaus größere Brocken mit etwa 2,5 GB. Das SDK benötigt nur der Programmierer und es muss bei den Grafikern nicht installiert werden.

Die Aktivierung einer sogenannten Floating Licence für den Modeler muss für jeden Rechner, auf dem das Produkt installiert wird, neu erfolgen. Die Anzahl ist hier unbegrenzt. Die Lizenzanfrage erfolgt online im SpeedTree Modeler mittels einer Host ID, die versandt wird. Und der benötigte Lizenzcode wird derzeit noch persönlich durch einen IDV-Mitarbeiter erstellt und via E-Mail verschickt.Bis zur letztendlichen Freischaltung einer Version können also einige Stunden vergehen. Möchte man schnell loslegen, sollte man dies unter Berücksichtigung der Zeitverschiebung bedenken. Man kann diesen Vorgang etwas beschleunigen, wenn man die Host-ID (wird beim Start eines unlizensierten SpeedTree Modelers immer wieder in einem Pop-up dargestellt) selbst noch einmal via E-Mail an seinen Sachbearbeiter verschickt.

Die Aufgaben der verschiedenen Programmteile sind soweit selbsterklärend. In dem Modeler modelliert man die Bäume auf Basis einer prozeduralen Technik, und die Baumsammlung bietet zur Vereinfachung der Baumerstellung schon einen riesigen Fundus an voreingestellten (i.d.R. botanisch korrekten) Bäumen und Texturen.

Der SpeedTree Compiler ist interessant zu erwähnen. Die in dem Modeler erstellte Vegetation wird mit dem Compiler in ein für die Engine optimiertes Format umgeschrieben, und die verwendeten Einzeltexturen werden in "Atlas Sheets" umarrangiert, damit nicht etliche von Einzeltexturen im Spiel abgerufen werden müssen. Dies spart sogenannte "Draw Calls". Die Texturkoordinaten der Bäume werden in diesem Prozess auch automatisch auf die neuen Atlanten angepasst.

Des Weiteren schreibt der SpeedTree Compiler auch Billboards für die Vegetation heraus, damit sehr weit entfernte Bäume im Spiel nur noch als niedrig-polygonale Planes gerendert werden. Zu guter Letzt haben die



SpeedTree-Vegetation aus dem aktuellen Spiel „Risen 2" von Piranha Bytes

Programmierer im Team auch noch zu tun, indem sie den SpeedTree-Programmcode in die Spiele-Engine integrieren, und zwar mit dem SDK. Mithilfe dieser Programmteile wird ein SpeedTree korrekt im Spiel angezeigt und vom SpeedTree-Wind bewegt.

Der Wind selbst wird auch im SpeedTree Editor erstellt und in jedem SpeedTree File gespeichert. In den einzelnen Baumkomponenten werden verschiedene Attribute abgelegt, die angeben, wie die jeweilige Komponente auf Wind reagiert. Ein Farn beispielsweise ist sehr biegsam im Gegensatz zu einem abgebrannten dicken Geäst. All dies stellt man individuell ein, und mithilfe des SDK können diese Daten ausgelesen und interpretiert werden.

Der SpeedTree Modeler

Es soll nicht unerwähnt bleiben, dass die Entwicklung von SpeedTree nun seit vielen Jahren vorangetrieben wird. IDV hat sich in dieser Zeit durchaus bemüht, die Feature-Requests der Kunden in ihr Produkt einfließen zu lassen. So ist es heute zum Beispiel aufgrund der hohen Nachfrage möglich, die Bäume als Mesh-Geometrie mitsamt Texturen zu exportieren als auch eigene Meshes ganz einfach im OBJ-Format in SpeedTree zu importieren. Installiert man sich heute also eine aktuelle Version des Modelers, hat man es mit einer durchaus ausgereiften und fast bugfreien Software zu tun.

Die Programmoberfläche

Schauen wir uns erst einmal das Layout der Programmoberfläche an, welche grob in vier Teile unterteilt ist. Von links nach rechts haben wir zuerst das "Properties"-Feld. Hier werden alle Einstellungen, wie zum Beispiel Astlänge und -dicke aber beispielsweise auch das UV-Mapping eines Astes festgelegt. Auch Einstellungen für den Wind können hier vorgenommen und abgespeichert werden. Danach kommt der Viewport, welcher oben noch ein paar Icons besitzt, mit denen man Baumteile schnell ein- oder ausblenden kann. Dort oben kann man auch zusätzliche Darstellungen, wie zum Beispiel Collision-Geometrie sichtbar machen.

Rechts oben ist der Asset-Bereich. Hier werden Texturen oder Custom Meshes eingeladen. Denn man kann in SpeedTree auch eigene Geometrie einladen und diese entweder als Baumgeometrie verwenden oder einen Baum um eingeladene Meshes herumwachsen lassen. Schließlich ist unten rechts noch eine schematische Ansicht unserer Node-Hierarchie, auf die ich kurz eingehen möchte.

Die Nodes

Denn bevor wir nun den ersten Ast erstellen, sei zunächst erwähnt, dass ein SpeedTree-Baum ein hierarchisch aufgebautes Konstrukt ist, welches mithilfe von Generatoren zusammengesteckt wird. Ich werde diese Generatoren der Einfachheit halber im Folgenden mit "Nodes" oder "Objektknoten" bezeichnen.

Im Grunde gibt es in SpeedTree nur zwei Hauptkategorien von Objektknoten. Zum einen gibt es die "Branch Nodes" und zum anderen die "Leaf Nodes". Aus diesen zwei Objektarten lässt sich jeder Baum herstellen. Wobei der Branch Node hauptsächlich zum Erstellen von Stämmen, Ästen, Zweigen, Lianen oder Palmblättern benutzt wird, und der Leaf Node zum Erstellen von kleinerem Blattwerk. Diese beiden Objektarten haben unterschiedliche Einstellungsmöglichkeiten im Properties-Bereich.

Tree Modeling

Um nun auf das Modeling einzugehen, wollen wir den Vorgang nicht anhand eines fertigen Baumes aus der Library besprechen, sondern selbst einen Baum von Grund auf neu erstellen. Auf der vorangegangenen Seite sehen wir das Endergebnis, welches wir besprechen wollen. Zunächst erstellt man sich ein neues und leeres Dokument unter dem Menüpunkt "File"->"New".

Ein Node, der Hauptknoten, wird dann automatisch im Node-Fenster erstellt und im Viewport wird ein weißer Ring am Boden dargestellt. Der Hauptknoten ist weder ein Branch Node noch Leaf Node, er ist ein spezieller Node, ohne den kein Baum auskommt. Er ist stets Ankerpunkt für darauf folgende Nodes. Gleichzeitig kann dieser Node auch keine Geometrie erstellen. Dafür braucht man immer einen der oben beschriebenen Objekt-Nodes.

Dieser Ring stellt zunächst einmal nur den Radius dar, in dem der erste Ast potenziell erzeugt werden kann beziehungsweise der erste Geometrie-Node stattfinden kann. Man kann diesen Radius stets im Properties-Fenster verändern als auch diverse andere Settings, wie zum Beispiel einen Multiplizierer auf die Gesamtgröße des Baumes und diverse Darstellungsoptionen. Auch das LOD'ing kann hier innerhalb des SpeedTree Modelers angeschaltet werden, damit man die verschiedenen LOD-Stufen vorab auf Funktionalität und Aussehen testen kann. Im Node-Fenster, rechts unten, führt man

nun zum Beispiel einen Rechtsklick auf den Haupt-Node aus und kann mit "Add template to selected"->"Trunks"->"Standard RT" einen Baumstamm, in dem vom Haupt-Shader-Knoten vorgegebenen Radius erzeugen.

RT steht für Real Time und ist letztlich nur ein Indiz für eine polygonreduzierte Voreinstellung für dieses Template. Die Nodes werden hier als Templates bezeichnet, da diese nach wie vor alle noch dieselben Branch Nodes sind, diese aber je nach Nutzungsart, mit unterschiedlichen Werten befüllt sind. In den Properties dieses ersten Objekt-Nodes stehen nun etliche Parameter zur Verfügung, mit denen man die Geometrie des Baumstammes definieren kann.

Die Fülle an Einstellungsmöglichkeiten ist immens und kann hier nur angerissen werden. Man hat zunächst einmal Zugriff darauf, wo der Baumstamm innerhalb des Radius vom Haupt-Shader-Knoten erstellt wird – ob mittig oder eher am Rand. Dies macht in dem Fall vor allen Dingen Sinn, wenn man mehrere Baumstämme oder Äste direkt zu Anfang erstellen möchte. Beispielsweise für einen Busch, denn man kann hier außerdem die Anzahl der erstellten Stämme vorgeben. Und da ein Busch in der Regel aus einigen kleineren Bäumen besteht, bietet es sich an, hier direkt beim ersten Geometrie-Node mehrere kleine Baumstämme in einigem Abstand zueinander zu generieren.

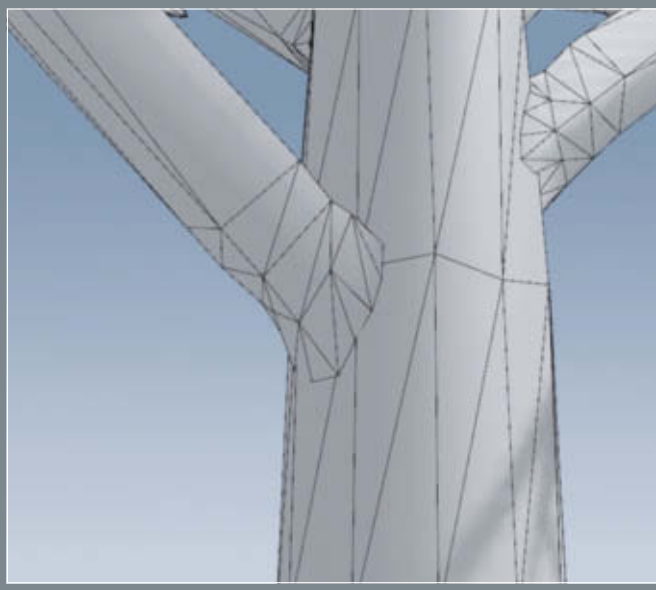
Etwas weiter unten kommen die Sektionen "Spine" und "Branch", in denen die Form des Astes vorgegeben wird: Länge, Dicke, Grad der Verformung, Astbruch, Astteilung, Anzahl und Ausprägung von Wurzelansätzen et cetera. All diese Werte werden in weiteren, später addierten Geometrie-Nodes ebenfalls genutzt, um die Äste an dem Baumstamm zu generieren. Nur die Werte, mit denen die Parameter dann gefüllt werden, sind jeweils andere.

Polycounts

Eine weitere wichtige Sektion sind die "Segments". In dieser Sektion gibt man die Polygonauflösung des jeweiligen Objektes an. Gerade bei der Spieleproduktion kommt es erheblich auf den Polycount der verwendeten Assets an. Die Polygonanzahl folgt im Idealfall dem Motto: "Stets so wenig wie möglich, doch so viel wie nötig." Manchmal auch weniger als nötig. Polygoncounts sind heutzutage nach wie vor nebst Füllrate, Draw Calls und Ressourcenverbrauch kritische Elemente in der Spieleproduktion.

Je nachdem wie es das Leveldesign erlaubt, kann man mehr oder weniger Polygone für ein einzelnes Asset verwenden. Und hat man eine offene Welt zu bestücken, so kann man nicht immer auf Sichtschutz hoffen, der zum Beispiel einen großen Teil des Spielgebietes ausblenden kann. Stattdessen geht man in dem Fall besser vorsichtig vor und hält die Polycounts der einzelnen Assets möglichst gering.

In unserem aktuellen Titel „Risen 2" haben große Bäume einen Polycount von 2.000 bis 6.000 Triangles – je nach Form und Art des Baumes. Büsche haben zwischen 300 bis 1.000 Triangles. Die Geometrie-LOD-Stufen (wir haben immer nur eine, die noch relevante Geometrie-Features besitzt, bis nach dieser dann die Billboard-LOD-Stufe folgt) haben bei uns nur noch ungefähr ein Viertel bis ein Fünftel des ursprünglichen Polygoncounts.



Man sieht, wie die Radialsegmente schon kurz nach dem Anfang des Astes in der Anzahl reduziert werden.

Äste und Zweige

Um nun unseren Baumstamm mit einigen Ästen auszustatten, fügen wir an den Trunk Node mit der rechten Maustaste ein "Branches->Standard RT" Template an, wodurch zwei weitere Nodes in der Hierarchie erzeugt werden, die sich sofort mit dem Baumstamm-Node verbinden. Im Viewport sieht man entsprechend die neue Geometrie der Äste, welche sich in den oberen zwei Dritteln des Baumstammes erzeugt hat.

Das "Branches-Standart RT"-Template, besteht aus zwei Ebenen von Branch Nodes, die nacheinander geschaltet sind. Man sieht hier, dass Templates auch aus ganzen Node-Konstrukten bestehen können. Es ist auch jederzeit möglich, selbst erzeugte Node-Verschachtelungen als eigene Templates abzuspeichern. Dafür muss man nur mittels Auswahl-Rechteck im Node Editor oder durch Strg-Linksklick mehrere Nodes auswählen und dann mit Rechtsklick "Save selected as template" ausführen.

Schauen wir uns nun den Polycount im OnScreenDisplay (OSD) des Viewports an, so sehen wir einen kombinierten Polycount von allen Ästen plus Stamm von circa 7.500 Polygonen. Damit bewegt man sich also schon über dem Limit von dem, was wir in unseren Spielen maximal an Geometrieaufwand für einen großen Baum einsetzen. Da uns hier aber noch Blätter fehlen, und je nachdem vielleicht auch noch extra Wurzelgeometrie, würden wir das Limit schnell übersteigen.

Folglich müsste man jetzt schon damit beginnen, die Segmente der einzelnen Nodes zu optimieren. Speziell bei sehr großen Bäumen kann dies bald zu einem künstlerischen Problem werden, da man stets zwischen guter Optik und niedrigem Polycount abwägen muss. Äste mit wenigen Segmenten sehen schnell sehr eckig aus. Um die Äste zu optimieren, hat man aber verschiedene effektive Möglichkeiten. Zum einen kann man schon zur Hälfte der Baumlänge anfangen, die Radialsegmente der Äste auf nur vier oder gar drei Seiten zu reduzieren. Dies fällt am letztendlich texturierten Baum kaum auf und spart sehr viel Geometrie.

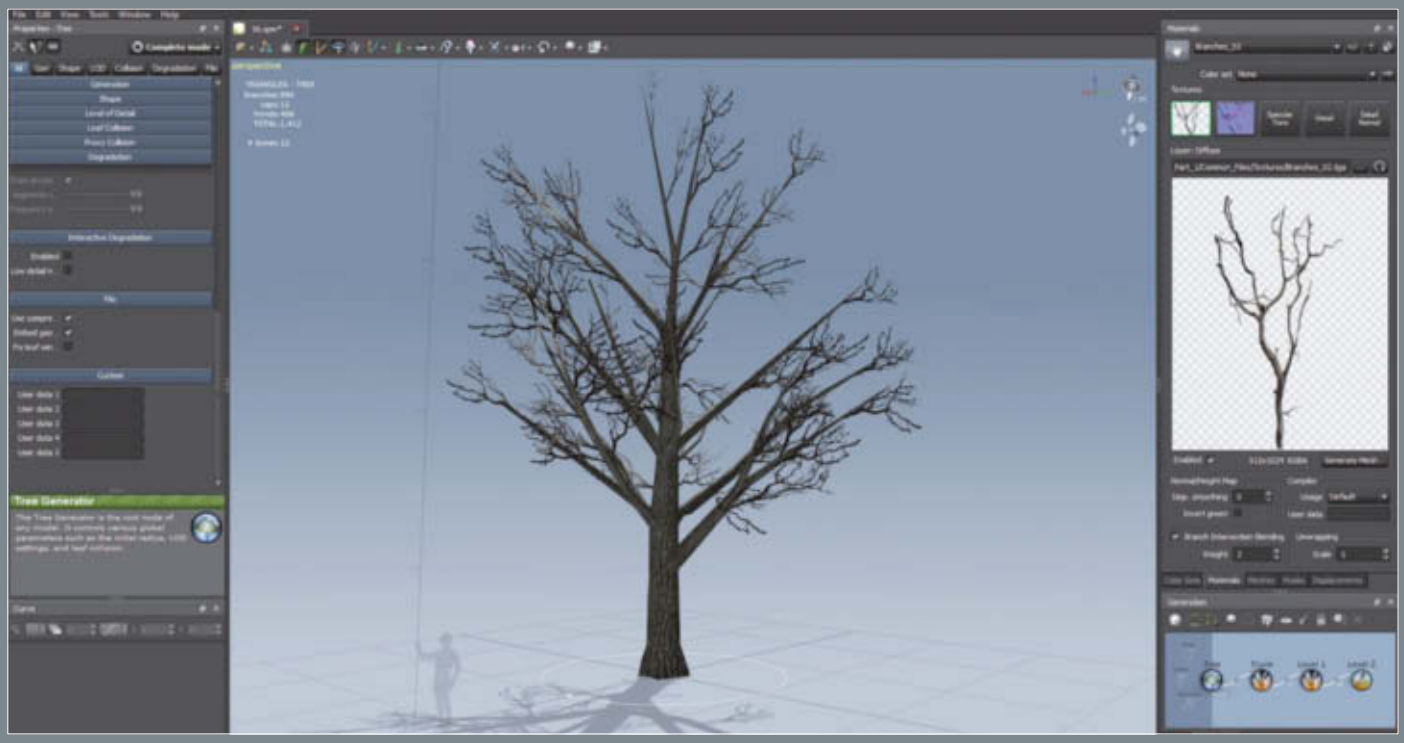
Man kann sogar mittels des Kurven-Editors, welcher sich im unteren Bereich des Properties-Fensters befindet, angeben, dass die Äste direkt am Baumstammübergang noch eine hohe Segmentauflösung aufweisen, dann aber schnell die Segmentreihen im Verlauf des Astes auf ein Minimum reduziert werden.

Des Weiteren fällt auf, dass die meisten Polygone in unserem Standard-RT-Template von dem Level-2-Node erzeugt werden. Also von den Ästen, die ihrerseits schon an der ers-ten Lage Äste erzeugt werden. Dies liegt zum einen an der großen Anzahl der Level-2-Äste und zum anderen an deren hoher Geometrieauflösung.

Man kann nun versuchen, die Polygondichte dieses Nodes mittels der Segmente und Reduktion der Anzahl der Äste zu verringern, wird aber feststellen, dass die Optik bald unter den Optimierungen leidet. Die Äste werden zu kantig und mit einer reduzierten Anzahl dieser Äste verliert der Baum auch an Dichte. Um also eine hohe Dichte von kleinem Geäst zu erlangen, ohne die Polygonanzahl zu strapazieren, kann man sogenannte "Fronds" einsetzen.

Von „Fronds" und Materialien

"Fronds" sind im Prinzip nichts anderes als veränderte Branch Nodes, welche nur noch mit zweiseitigen Polygon Planes dargestellt



Asttexturen als auch Blatt-Texturen liegen in der Regel als TGA-Format vor und haben einen Alpha-Kanal mit abgespeichert, der den transparenten Texturbereich maskiert.

werden. Auf diesen Planes werden letztendlich Ast-Texturen mit Alpha Kanal dargestellt. Um unseren Level-2-Branch-Node entsprechend umzustellen, stellt man den sogenannten "Geometry Type" auf den Wert "Frond" und die Bifurcation Chance stellt man auf "0". Mit der Bifurcation kann man Astgabeln an Ästen erstellen. Wird jedoch eine 2D-Plane auf diese Weise geteilt, werden die Texturteile der Asttextur nicht mehr aufeinanderpassen. Daher sollte bei Fronds auf diese Funktion verzichtet werden. Auf der Fronds-Textur sind in der Regel mehrere Astunterteilungen dargestellt, damit man den Eindruck von dichtem Geäst erhält. Der Vorteil von Fronds gegenüber echten 3D-Ästen ist natürlich, dass sie Polygone betreffend wesentlich effizienter sind.

Ein 3D-Ast mit Volumen besitzt mindestens drei Seitenflächen pro Längensegment, um nur einen einzelnen Astbereich darzustellen, wohingegen ein Frond nichts anderes als eine doppelseitige Polygon Plane ist. Verwendung finden die Fronds hauptsächlich bei kleineren Zweigen und dünnem Geäst. Dies fällt kaum auf, ja ist sogar in diesem Bereich optisch ansprechender als tatsächliche 3D-Äste, da diese sehr niedrig-polygonal ausfallen müssten, und dementsprechend kantig aussähen.

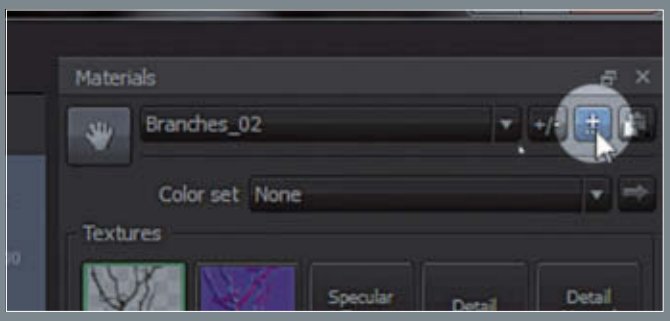
Möchte man nun wie in dem Beispielbild sowohl die Baumstämme als auch die Ast-Polygone mit Materialien versehen, so wenden wir uns der Asset-Verwaltung zu, welche sich oben rechts direkt über dem Nodes-Bereich befindet. Es kann hier mithilfe von Reitern, auch bekannt als Tabs, zwischen verschiedenen Asset-Bereichen gewählt werden. Im Bereich "Materials" kann man nun sehr einfach neue Materialien anlegen, indem man einen oben dargestellten Shortcut anklickt.

Der Shortcut ist mit einem kleinen Icon visualisiert, welches ein Pluszeichen darstellt. Hier wird man nun direkt in ein Explorer-Fenster verlinkt, in dem man eine Diffuse Textur auswählen kann. Hat man sich für eine entschieden, so wird automatisch ein Material mit dem Namen der Diffuse Textur angelegt. Zudem sucht SpeedTree auch automatisch nach Texturen mit dem Suffix "*_Normal.*" und "*_Spec* und trägt diese, sofern welche gefunden werden, automatisch in die entsprechenden Material-Slots ein.

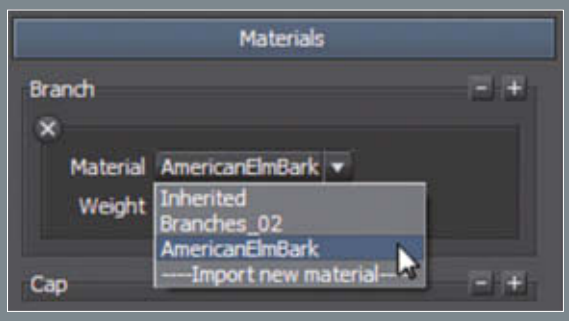
Hat man nun die beiden benötigten Materialien erstellt, möchte man diese den einzelnen Baumteilen zuordnen. Dieser Vorgang wird in den jeweiligen "Properties" der einzelnen Tree Nodes vorgenommen. Ein Klick auf den Trunk Node im Nodes-Bereich öffnet dessen Eigenschaften im "Properties"-Bereich. Dort findet man das "Materials"-Rollout, in dem man letztlich das Material zuweisen kann.

Das war also nun das Grundgerüst, in der nächsten DP werden wir uns dann mit den Blättern, Mesh-Importen, LOD und dem Compiler beschäftigen – sowie den Problemen von SpeedTree, zum Beispiel bei Kollisionen.

› mf



Man kann einfach neue Materialien erstellen, indem man das kleine Pluszeichen anklickt.



Die erzeugten Materialien erscheinen automatisch in den Node-Eigenschaften.

animago 2013

1. März 2013

Startschuss
animago AWARD

30. Juni 2013

Einreichfrist
animago AWARD

24./25. Oktober 2013

animago AWARD & CONFERENCE

SAVE THE DATE!

Rund 950 Beiträge, davon die Hälfte von internationalen Digital Artists, Produktionen aus Moldawien, Argentinien, Makedonien, Sri Lanka, Peru und vielen anderen Ländern rund um den Globus – das ist die Bilanz des animago AWARD 2012. Seit 1997 zeichnet der vom Fachmagazin DIGITAL PRODUCTION ins Leben gerufene Wettbewerb Spitzenleistungen im Spektrum digitaler Medienproduktion mit Schwerpunkt Animation/VFX, Visualisierung und interaktive Medien aus. Die animago AWARD & CONFERENCE wird seit 2009 vom Medienboard Berlin-Brandenburg gefördert.

Veranstaltet von:
DIGITAL PRODUCTION

Gefördert durch:
medienboard Berlin-Brandenburg

Bilder: animago AWARD & CONFERENCE 2012

www.animago.com
animagoAWARD