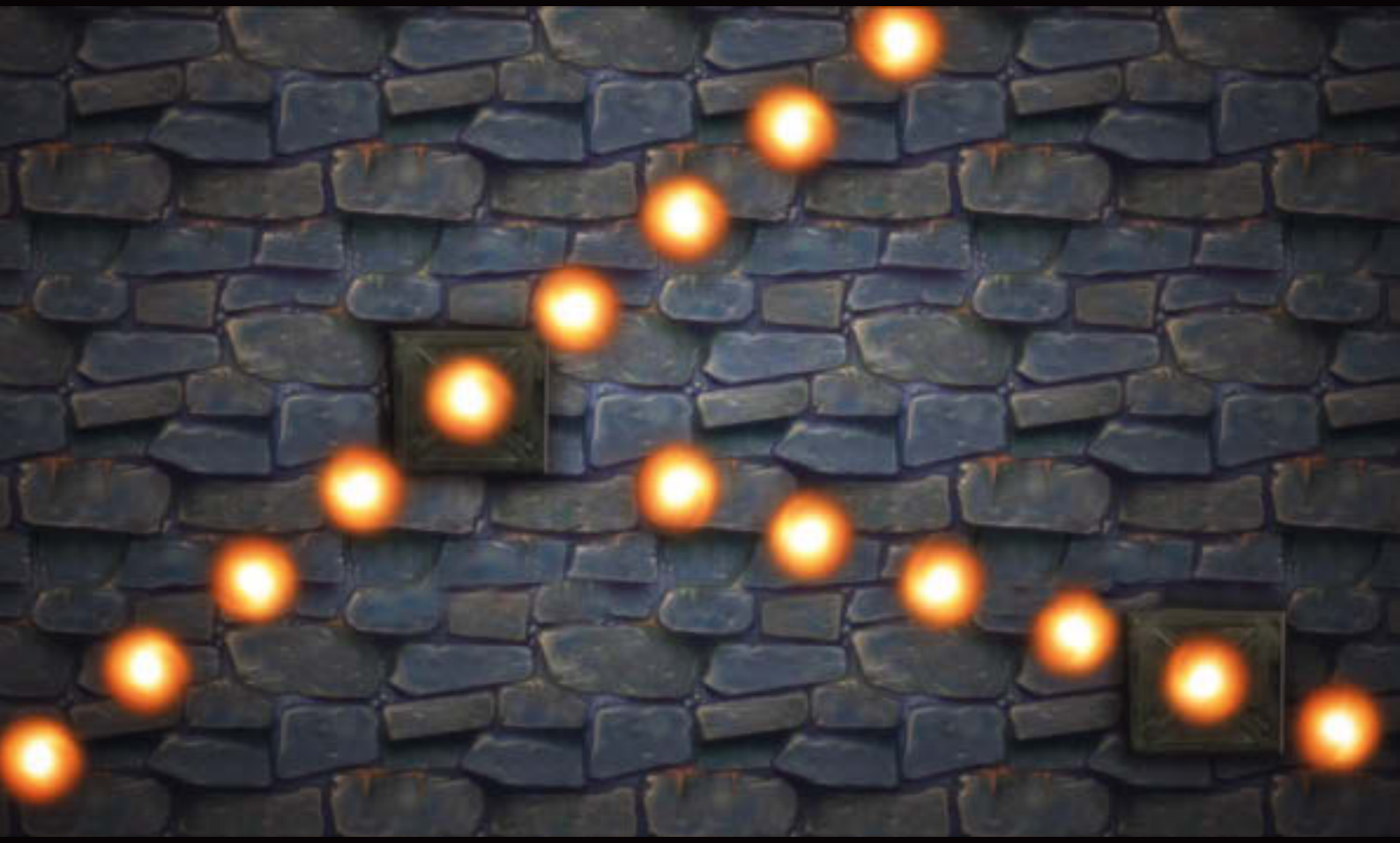


Teil 3

Fusion 9: Partikel durch Expressions bewegen

Seitdem Blackmagic Design die Software Compositing Fusion 9 Studio für kleines Geld veräußert – und damit ist eine kommerzielle Einzelplatzlizenz für 300 Euro gemeint – wird das Interesse, auf den erschwinglichen Compositing-Giganten umzusteigen, immer größer. Dem Anwender wird ein großes Repertoire an Werkzeugen geboten, mit denen sich Partikel auf alle erdenklichen Weisen steuern lassen, angefangen von einfachen, kreisrunden Bewegungen für kleine Motion-Graphics-Projekte bis hin zu wilden Animationen basierend auf mathematischen Fundamenten. Für beide Fälle lassen sich für unterschiedlichste Zwecke aufbereitete Nodes verwenden, doch das wahre Potenzial liegt bei Fusion 9 in den sogenannten Custom Nodes. von Rainer Duda

Einer der berüchtigten Custom Nodes, die in diesem Workshop näher unter die Lupe genommen werden, ist der pcustom Node. Das kleine P am Anfang des Namens gibt dem Anwender zu verstehen, dass der Node im Partikel-Bereich beheimatet ist und demnach nur in Verbindung mit Partikeln und deren gesamtem Repertoire an Nodes genutzt werden kann. Wie man den Node sinnvoll in ein Projekt einbindet und wie der Anwender selbst Expressions aufsetzen kann, wird in diesem Teil 3 der Einführungsreihe zu Fusion 9 aufgezeigt.

Particles before Partikel!

Partikel lassen sich durch den Einsatz unterschiedlicher Particle Force Nodes wunderbar bewegen. Doch wie stellt es der Anwender an, Partikel zu bewegen, bevor überhaupt ein Particle Force Node ins Spiel kommt? Die Antwort liegt nahe: durch den Einsatz von mathematischen Grundkonzepten, die Partikel für Motion Graphics in vordefinierten Laufbahnen bewegen. Es gibt in professionellen Medienproduktionen Projekte, bei denen die standardmäßigen Nodes nicht mehr ausreichen und Partikelpositionen durch Parametrisierung aufbereitet werden müssen. Des Rätsels Lösung ist der Einsatz des pcustom Nodes, der die Eigenschaften der einzelnen Partikel durch gezielte Expressions beeinflusst.

Wenn man den pcustom Node das erste Mal erblickt, ist man wahrscheinlich erstaunt und auch verwundert, teilweise auch verloren. Das liegt am Aufbau des Nodes mit den unterschiedlichen Tabs und deren Unterteilungen. Viele Informationen wirken verdeckt. Als erstes ist es wichtig zu verste-

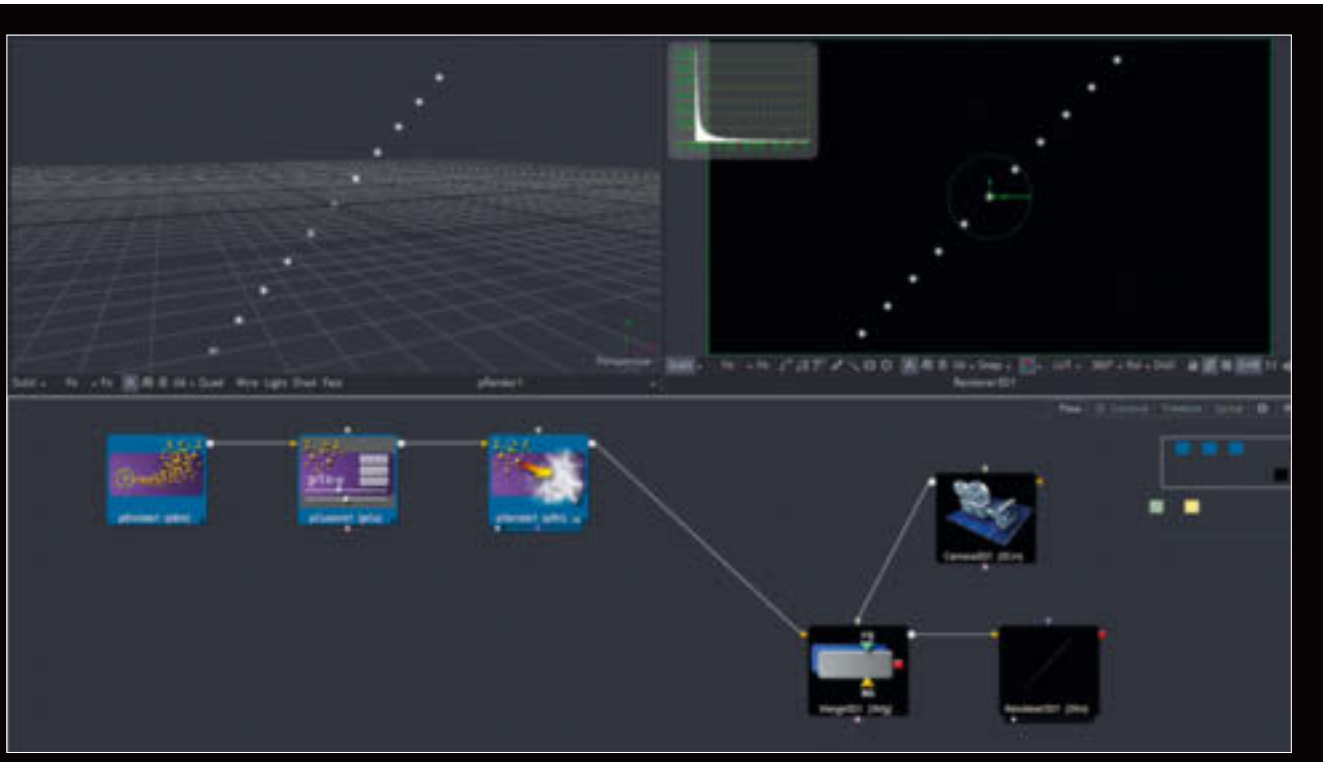
hen, wo genau der pcustom Node eingesetzt wird – vom Node Graph her gesehen. Der pcustom Node wird zwischen Partikel-Emitter (pemitter) und Particle Renderer (prender) Node eingefügt. Setzt man den Node ein, dann wird keinerlei Veränderung wahrgenommen – es wurden ja noch keine Expressions definiert.

Aufbau des pcustom Nodes

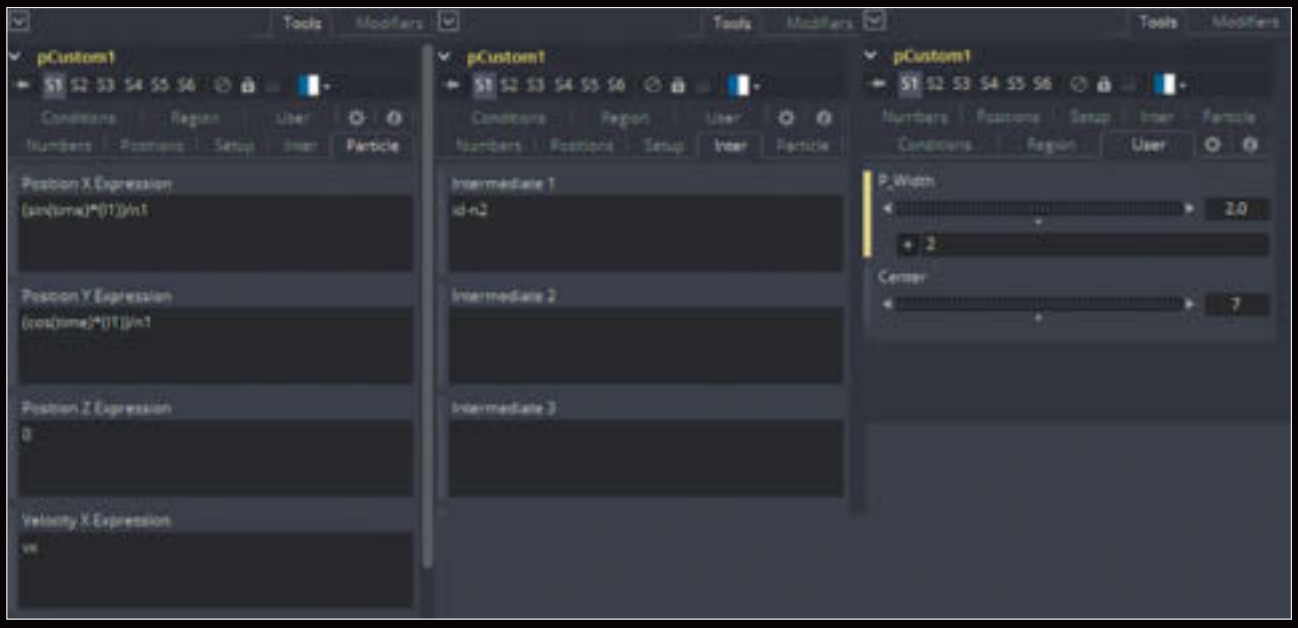
Der erste Tab, den der Anwender zu Gesicht bekommt, nennt sich „Numbers". Darin befinden Osind standardmäßig acht Schieberegler – alle auf den Wert 0 gestellt. Die Namensgebung ist rudimentär und benennt jeden Schieberegler mit „Numbers" und der jeweiligen Reihennummer von 1 bis 8. Hinter diesen Schiebereglern verbergen sich Blanko-Variablen, die der Anwender beliebig nutzen kann, und zwar für unterschiedlichste Kontrollmechanismen. Das Verhalten der Partikel kann so parametrisiert werden, ohne fixe Zahlenwerte direkt in die Expressions einzubinden – Stichwort Prozeduralität. Des Weiteren können die Variablen animiert werden, wenn die Motion Graphics dirigiert werden sollen. Möchte man nun mit den Variablen (Numbers) arbeiten, dann lassen sie sich in den Expressions sehr einfach adressieren, nämlich durch die Eingabe des Buchstaben n gefolgt von der jeweiligen Zahl – n1, n2, nX.

Setup

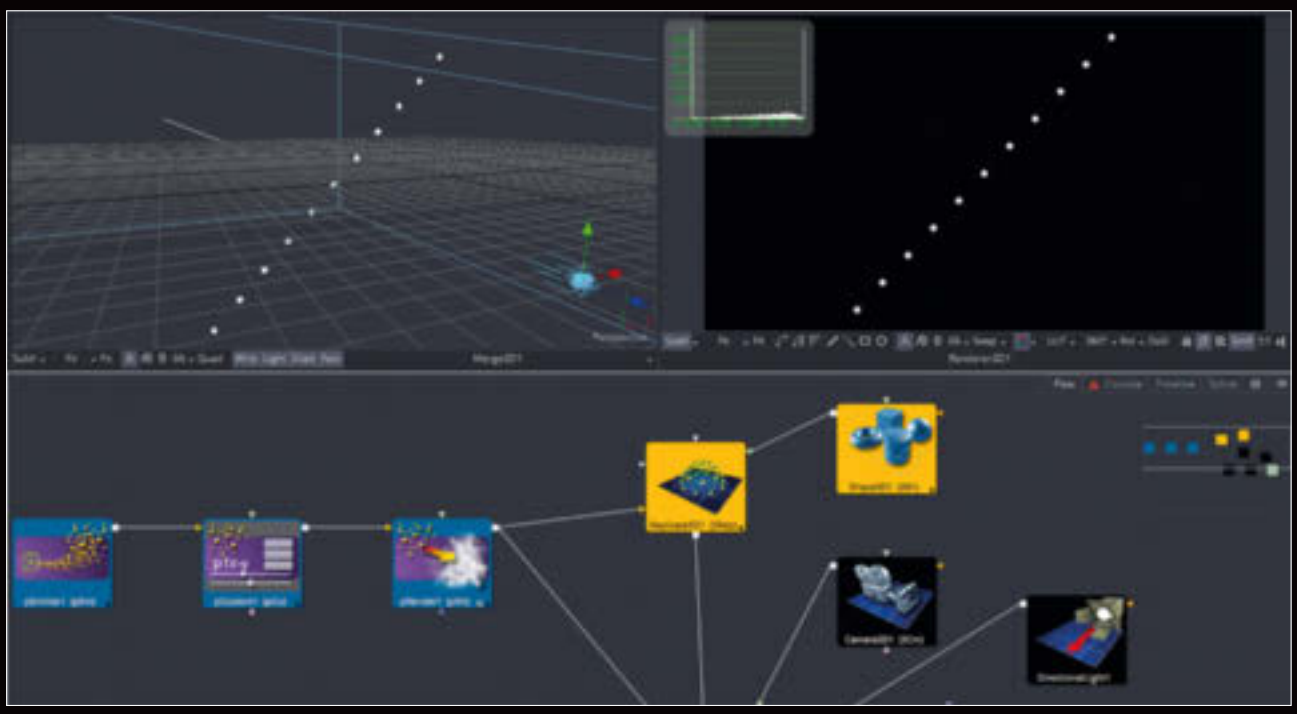
Ein weiterer wichtiger Tab nennt sich „Setup". Darin enthalten sind standardmäßig acht leere Felder. Trägt der Anwender Expressions innerhalb des Setups ein, dann muss er sich bewusst machen, dass die Setups einmal pro Frame ausgeführt werden und vor



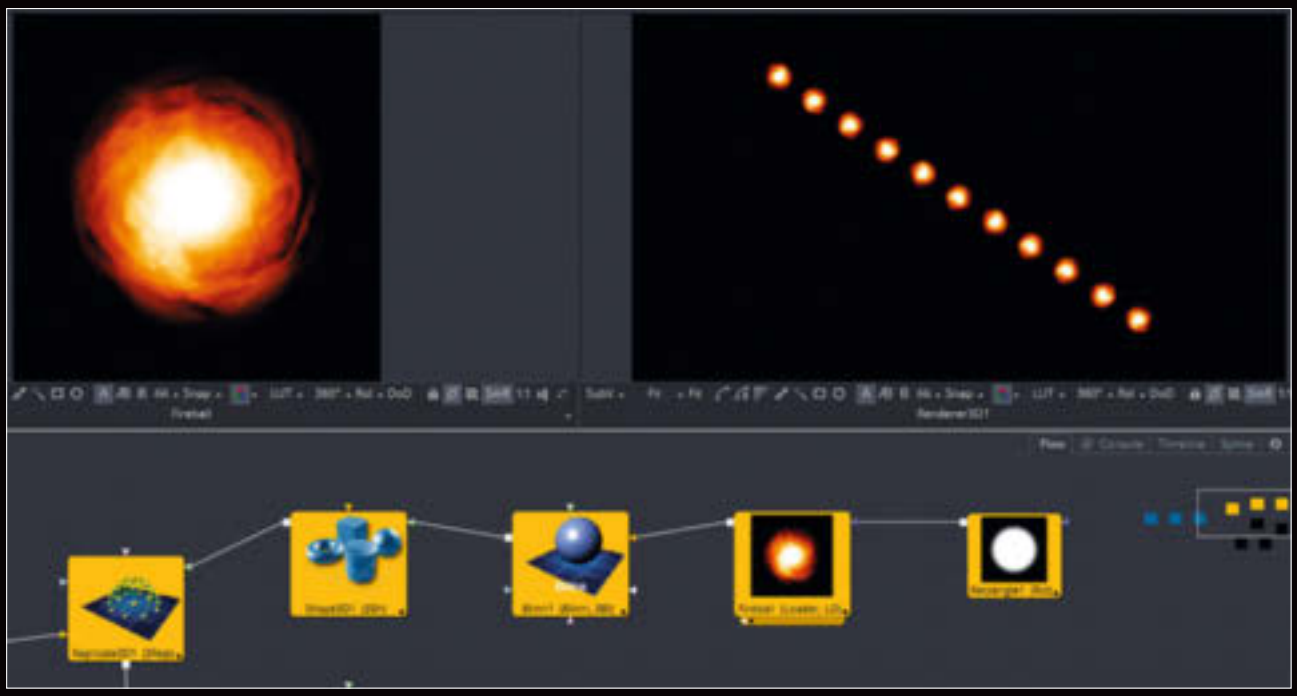
Um die aus „Super Mario" bekannte Feuerschranke nachzubilden, kommen Partikel zum Einsatz.



Die Partikel sind an einer Linie entlang ausgerichtet, wobei die Rotationsachse nach Wunsch verändert werden kann.



Die Darstellungsmethode des Blobs wird nun durch Geometrie ersetzt und so jedem Partikel ein eigenständiges Mesh zugewiesen.



Der Feuerball wird als Textur hinzugefügt und mit einer eigens in Fusion erstellten Maske zurechtgeschnitten.

allem als Erstes, bevor andere Expressions evaluiert werden. Daher der Name Setup. Darauf aufbauend gibt es den Tab „Inter" – kurz für Intermediate. Die Besonderheit der zu erstellenden Expressions im Intermediate-Bereich ist der Zeitpunkt der Ausführung. Während Setups als Erstes bei jedem Frame evaluiert werden, sieht es bei den Intermediate Expressions so aus, dass sie pro Frame pro Partikel evaluiert werden. Die acht Blöcke im Intermediate-Bereich können ähnlich einfach in Expressions adressiert werden wie die Numbers, nur dass statt dem n ein i an erster Stelle steht – i1, i2, i3 und iX.

Particle

Der wichtigste Tab trägt den Namen „Particle". Die dort aufzufindenden Expressions werden ebenfalls pro Frame für jeden Partikel evaluiert. Innerhalb dieses Tabs wird in professionellen Produktionen am meisten Zeit verbracht, da darin die am meisten genutzten Expressions liegen. Dazu zählen zum Beispiel die Position von XYZ, Velocity für XYZ, Rotation für XYZ, Masse, Größe, Farbkanäle für RGB und Spinning für XYZ. Der erste Schritt in Richtung Arbeit mit dem pcustom Node beruht darauf zu wissen, wie man die darin befindlichen Informationen und Variablen verständlich deklariert bzw. bezeichnet und zuweist. Um das zu bewerkstelligen, reicht ein Klick mit der rechten Maustaste auf den Namen pcustom (den Namen des jeweiligen pcustom Nodes) in der Tool-Ansicht (rechts am Bildschirm oben) – die Ansicht, in der sämtliche Einstellungen und Attribute/Variablen vorhanden sind. In dem kleinen sich öffnenden Optionsmenü findet sich die Zeile „Edit Controls".

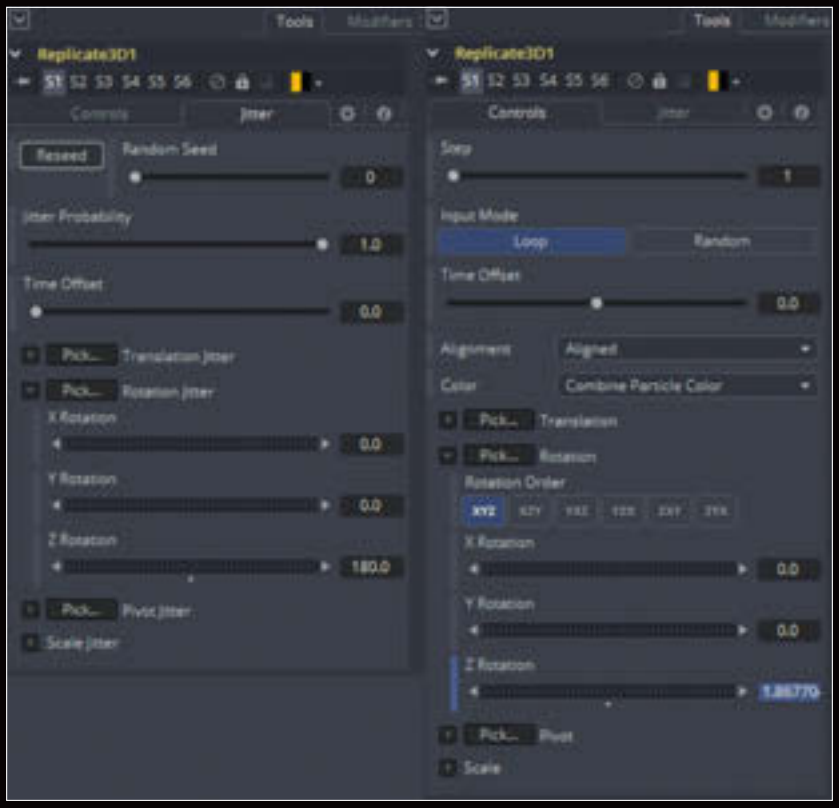
Betätigt man die Funktion, dann wird ein weiteres Fenster geöffnet, das es dem

Anwender erlaubt, über eine ID die jeweilige Variable auszuwählen und mit den im Fenster vorrätigen Einstellungsmöglichkeiten an die eigenen Bedürfnisse anzupassen. Bei Bedarf kann ebenfalls ein neuer Tab im pcustom Node erstellt werden, der als Container für die eigene Expression dient. Der Vollständigkeit halber sollte an dieser Stelle erwähnt werden, dass das Menü bei jedem Node vorhanden ist und eine Anpassung des jeweiligen Nodes vornimmt. Es lassen sich auch komplett neue Variablen erstellen, ohne dass man auf die bereits verfügbaren, nicht genutzten Variablen zurückgreift.

Die ersten Gehversuche

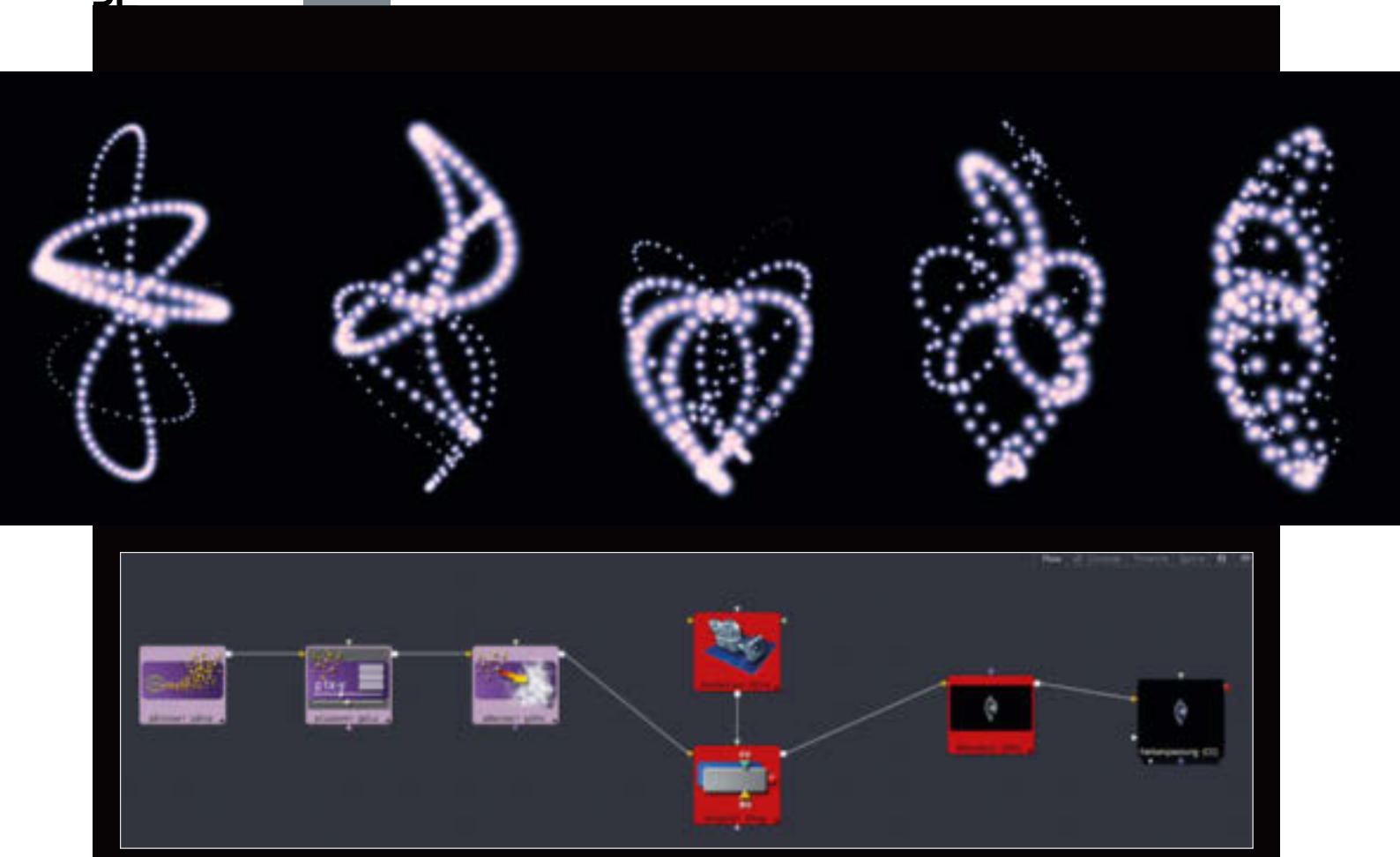
Das erste Beispiel, das mit dem pcustom Node realisiert wird, kennt man von berühmten Computerspielen wie „Super Mario". Ein mutiger Klempner, der die Prinzessin befreien möchte und auf dem Weg zu ihr allerhand Hindernisse überwinden muss. Eine Kategorie der Hindernisse sind bewegliche Schranken aus Feuerbällen, die Mario zwingen, auf den richtigen Zeitpunkt zu warten, um sie zu umgehen. Wie würde der Weg wohl aussehen, solche Feuerschranken selbst zu erstellen? Natürlich durch den Einsatz des pcustom Nodes. Die grundlegenden Einstellungen wie Frame-Format bleiben dem Anwender überlassen. Innerhalb einer neuen Komposition wird zunächst ein Particle Emitter Node erstellt nebst Particle Render Node. Beide müssen der Reihe nach miteinander verbunden werden. Dazwischen wird nun der erwähnte pcustom Node erstellt und eingefügt. Als erstes geht es darum, dem Emitter Node mitzuteilen, dass nur eine kleine Anzahl an Partikeln erzeugt werden soll, und das nur innerhalb des ersten Frames. Hierzu wird der Partikel-Emitter selektiert, in den Einstellungen innerhalb des Tabs „Controls" die Lifetime auf das Maximum gesetzt und die Expression-Zeile im Feld „Number" aktiviert. Letzteres geschieht über einen Klick mit der rechten Maustaste auf das Feld „Number" und eine Bestätigung auf dem Feld „Expression". Darin wird nun folgende Expression eingetragen: iif(time > 0, 0,10)

Die Expression bewirkt, dass am Frame 0 insgesamt 10 Partikel erstellt werden sollen, und sobald Frame 0 verlassen wird, werden keine weiteren Partikel erstellt. Für eine bessere erste Darstellung der Partikel kann im Tab „Style" der Stil mit dem Namen „Blob" ausgewählt werden. Zusätzlich sollte die Größe der Blobs erhöht werden, bis man innerhalb des Viewers/Monitors die Partikel gut erkennen kann. Es kann allerdings vorkommen, dass es Komplikationen bei der Transparenz gibt und das visuelle Ergebnis nicht zufriedenstellend ist.



Wie man es von herkömmlichen Scattering-Systemen gewohnt ist, wird auch in Fusion ein variables Offset im Bereich der Rotation vorgenommen.

Nachdem die Einstellungen seitens des Renderings vollzogen wurden, können Post-Effekte wie in diesem Fall ein Softglow hinzugefügt werden, damit das visuelle Ergebnis weicher dargestellt wird.



Mithilfe trigonometrischer Funktionen werden Partikel Akteure in einer futuristischen Benutzeroberfläche.

Hierzu kann innerhalb des Viewers/Monitors und am Render Node unter den Transparency-Einstellungen „Quick Sort" eingestellt werden.

Nun sollten die unterschiedlichen Blob-Objekte mit deren Transparenz keine Artefakte mehr hervorrufen. Dem prender Node sollte noch mitgeteilt werden, dass ausschließlich im High-Quality-Modus gerendert werden soll, und die gleiche Aktivierung sollte in der Zeitleiste am entsprechenden Symbol stattfinden.

Pcustom Node unter der Lupe

Nun geht es daran, den pcustom Node genauer unter die Lupe zu nehmen. Die Feuerschranke in den „Super Mario"-Spielen besitzt einen festen Ankerpunkt, um den sich eine bestimmte Anzahl an Feuerbällen im oder gegen den Uhrzeigersinn dreht. Es wird in diesem Workshop angenommen, dass die Z-Achse vorerst nicht verändert wird. Daher wird im Tab „Particle" im Feld „Position Z Expression" der feste Wert 0 eingetragen. Nun sollten die Partikel im Viewport fest auf der Mittellinie der Z-Achse liegen.

Um die Partikel im oder gegen den Uhrzeigersinn wandern zu lassen, gibt es die Mathematik mit dem Feld „Trigonometry". Eine weitere Erleichterung für den Anwender ist die Möglichkeit, parametrische Funktionen zu nutzen. Als erstes muss im Feld „Position X Expression" eine Sinusfunktion eingetragen werden, der als Wert die Variable für die Zeit zugeordnet wird – sin(time). Dementsprechend muss im Feld „Position Y Expression" eine Cosinusfunktion eingetragen werden, ebenfalls mit der Zeit als Variable – cos(time). Betätigt man daraufhin den Play-Button innerhalb der Zeitleiste, dann bewegen sich alle zuvor durch den Emitter erstellten Partikel im Uhrzeigersinn, jedoch sitzen quasi alle Partikel auf derselben Stelle und wandern im Kreis.

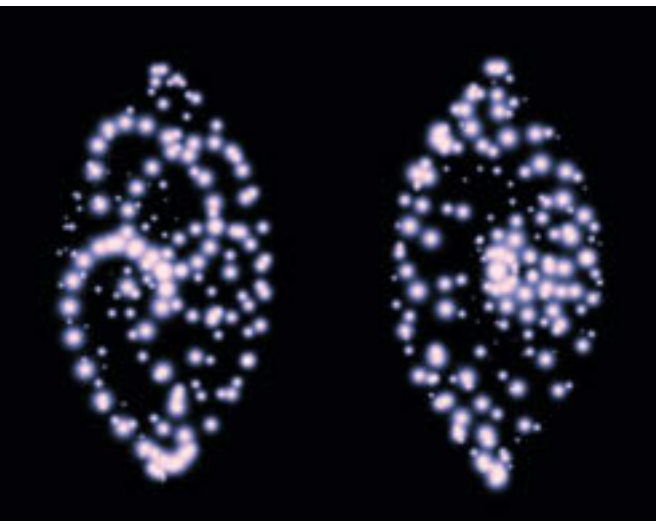
Beheben lässt sich die Positionierung der individuellen Partikel (Blob-Objekte), indem beide Expressions – für X- und Y-Position – mit der Partikel-ID multipliziert werden. Die ID kann über das Kürzel id angesprochen werden. Die Expressions sehen wie folgt aus: sin(time)*id sowie cos(time)*id. Da es sich bei den IDs um eine iterative Aufzählung handelt, sind die Partikel mit einem festen Abstand zueinander in einer Reihe angeordnet.

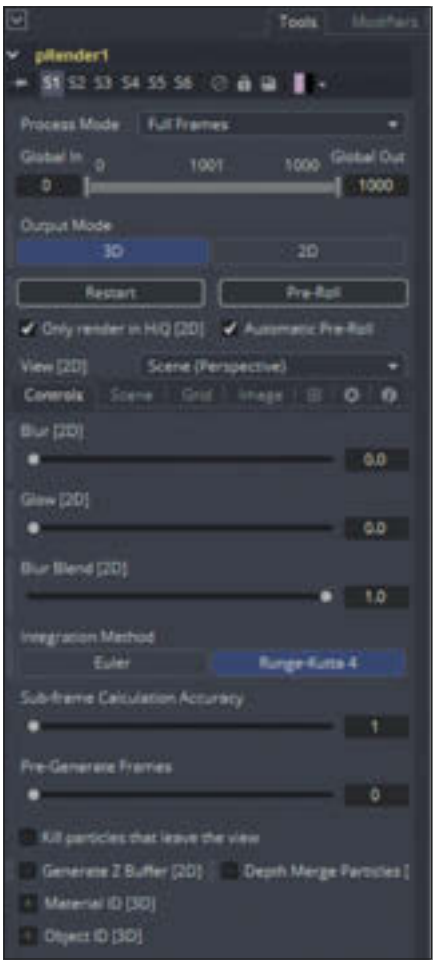
Die Anordnung der einzelnen Partikel in Blob-Form wirkt jedoch noch immer zu starr. Was noch fehlt, ist eine Variable, die die Abstände der einzelnen Partikel zueinander verändert. Diese Variable soll von Hand eingestellt werden oder per Animation die Positionen verändern. Der einfachste Weg besteht in der Aufbereitung einer Variablen für diesen Zweck. Durch einen Klick mit der rechten Maustaste auf den Namen des pcustom Nodes im Tool-Widget – standardmäßig rechts am Bildschirmrand – erscheint der Eintrag „Edit Controls".

Bestätigt man die Funktion, dann landet man in den Konfigurationseinstellungen des Nodes. Im Feld „ID" wird aus dem Numbers-Tab der erste Schieberegler ausgewählt – NumberIn1. Als nächstes wird ein neuer Name vergeben, wie zum Beispiel P-Width oder P-Distance.

Im Feld „Tab" wird der Tab ausgewählt, indem die eigene Variable abgelegt wird. Der Tab User bietet sich an, da er leer ist. Es sollten noch eine Range und ein Default-Wert definiert werden. Die Range kann von 0,1 bis 10 eingestellt sein mit einem Standardwert von 8. Über eine Bestätigung durch „OK" werden die Änderungen übernommen.

Ab jetzt ist die Variable bereit für ihren Einsatz. Zurück im Tab „Particle" muss nun jede Expression, also die Expressions in Position X und Position Y, durch die Variable n1 dividiert werden. Ab diesem Zeitpunkt lassen sich die Abstände der Partikel zueinander variabel anpassen.





Für die korrekte Darstellung der Benutzeroberfläche müssen die Einstellungen für das Rendering zielgerichtet angepasst werden.

Zentrumsprobleme

Doch ein großes Problem ist noch immer vorhanden. Betrachtet man die Animation als Render-Output auf dem Viewer/Monitor oder in der 3D-Szene direkt, dann merkt man, dass die Partikel nicht wie erhofft am Zentrum beginnen und um das Zentrum der 3D-Szene rotieren.

Das liegt an der Nummerierung der Partikel-ID und an der Art, wie Fusion die Partikel-ID vergibt. Es scheint, als würde bei den Partikel-Standardeinstellungen die Zählweise der Partikel beim Wert 2 beginnen. Als Lösung bietet es sich an, die Sinus- und Cosinus-Expressions – bevor sie dividiert werden – mit (id-2) zu multiplizieren und die Ergebnisse daraufhin durch n1 zu dividieren. Wahlweise kann hier im Tab „Inter" ein Feld mit der Expression id-2 belegt werden und die Expression id-2 in Position X und Position Y durch i1 ersetzt werden.

Mit der kleinen Anpassung liegt der erste Partikel auf dem Zentrum der 3D-Szene und die anderen liegen in dem definierten Abstand (ID-Wert) zum ersten Partikel. Nun hat man ein rudimentäres Setup für eine Feuerschranke. Betätigt man den Play-Button, so drehen sich die Blobs um das Zentrum der 3D-Szene, wobei der erste Blob ebenfalls auf dem Zentrum liegt und kein leerer Platz entsteht.

An dieser Stelle kann aus dem Blob ein Feuerball kreiert und über die Spin-Expressions rotiert werden. Arbeitet man gezielt im 3D-Raum, dann lässt sich Geometrie auf die Partikel verteilen, indem man einen Replicate 3D Node erstellt und die Partikel als Input für den Node nutzt. Im zweidimensionalen Bereich kann auf die altbekannten Compositing-Methoden gebaut werden, um aus dem Blob ein Feuer-Sprite zu erzeugen.

Wenn an dieser Stelle eine weitere Feuerschranke erstellt werden soll, die die bereits vorhandene Feuerschranke im Grunde nur verlängert, und das Zentrum, um das rotiert wird, um eine bestimmte Position versetzt werden muss, braucht man dafür kein Voodoo.

Anstatt auf eine große If-Verschachtelung zu bauen, sollte sich der Anwender bewusst machen, dass die Partikel-ID eine wichtige Rolle einnimmt und bestimmt, welcher Partikel im Zentrum der 3D-Szene liegt. Mit diesem Konzept im Hinterkopf lässt sich eine zweite Schranke einfach realisieren.

Daher ist es empfehlenswert, die manuelle Position, die durch die Expression id-2 festgelegt ist, variabler zu gestalten. Erneut wird eine Variable im Edit-Controls-Fenster angepasst – diesmal der zweite Schieberegler namens „NumberIn2". Eine Empfehlung an dieser Stelle ist, dass nur Integer-Werte benutzt werden, um zu gewährleisten, dass das Zentrum immer auf einem Blob liegt. Benannt wird die Variable mit etwas intuitivem wie „RotCenter". Im Intermediate-Tab wird im ersten Feld die Expression id-2 – falls die Optimierung zuvor durchgeführt wurde – angepasst und in id-n2 abgeändert.

Nun muss man nur noch die Expressions im Tab „Particle" anpassen. Die Rede ist von Position X und Position Y. Sie würden dann wie folgt aussehen: X=(sin(time)*(I1))/n1 sowie (cos(time)*(I1))/n1.

Über die neue Variable (n2, RotCenter) lässt sich das Zentrum der Feuerschranke nun variabel anpassen. Gegebenenfalls muss man an dieser Stelle am Emitter die Zahl der zu erstellenden Partikel am ersten Frame anpassen, um genügend fixe Partikel zu erhalten.

Ring of Fire

Feuerschranken sind schön und gut, doch wie würde es mit einem Feuerring aussehen? Auch hier erlaubt der pcustom Node den Einsatz von Trigonometrie. Bei den Feuerschranken werden die vom Emitter erstellten Partikel in eine Reihe gebracht, in jeweils dem gleichen Abstand zueinander. Wie würde es wohl aussehen, wenn eine bestimmte Anzahl an Partikeln mit dem Stil Blob entlang einer runden Bahn angeordnet ist?

Der Einfachheit halber wird der prender Node mit den Einstellungen belassen, wie er ist. Im pemitter Node wird eine Anzahl von 20 Partikeln an Frame 0 erzeugt und der pcustom Node wird gegen einen leeren pcustom Node getauscht. Der Ort des Geschehens am pcustom Node ist der Tab „Particle".

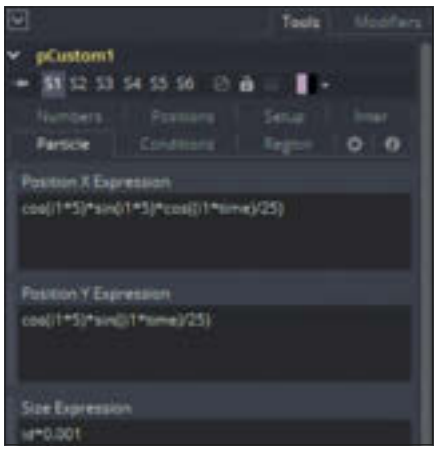
Dort sind die bereits bekannten Felder für die Positionen XYZ, die durch Expressions modifiziert werden können. Als erstes wird das Feld für die X-Position eingestellt. Hier wird wie zuvor eine Sinusfunktion benötigt. Innerhalb der Sinusfunktion wird zunächst von Hand definiert, wie viele Punkte an der Kreisfläche überhaupt entstehen sollen. Das geschieht über die Mini-Expression 360 dividiert durch die Anzahl der Punkte – zum Beispiel 360/16. Das allein reicht jedoch nicht. Es muss noch gesagt werden, wer sich daran halten soll – wer sind die ominösen 16 Punkte?

Die Mini-Expression wird somit mit der ID multipliziert. Die gesamte Expression lautet: sin(id*(360/16)). Die Expression für das Y-Feld sieht ähnlich aus, nur dass die

Sinusfunktion gegen eine Cosinusfunktion ersetzt wird: cos(id*(360/16)). Setzt man im Feld für die Z-Position vorerst eine 0, dann sollte nach betätigter Play-Taste eine runde Form aus 16 Partikeln erscheinen. Wenn man an dieser Stelle noch eine passende Animation einstellen möchte, wie zum Beispiel die Positionsveränderung vom äußeren Ring zum Mittelpunkt hin, muss man nur im Position-X-Feld die Expression mit cos(time) multiplizieren. Im Position-Y-Feld lautet die Expression sin(time).

Abgesehen von den sehr rudimentären Animationen von zuvor besitzt diese Art von Vorgehen enormes Potenzial, um exzellente visuelle Effekte zu erzeugen. So zum Beispiel für ein Hologramm auf einem futuristischen Display, das jedoch seitens der Energiequelle nicht die volle Auflösung der Applikation liefert und man innerhalb des Bildschirms Energiefluktuationen wahrnimmt. Dafür ist die Arbeit mit dem pcustom Node prädestiniert. Angenommen wird, dass eine futuristische Display-Lösung mit einer Touch-Screen-Funktion ausgestattet ist, der Fingerabdruckanalysen erlaubt. Durch die kaum noch vorhandene Energie des Betriebs ergeben sich Energiefluktuationen, die sich im visuellen Stil zeigen.

Der vorige pcustom Node wird entfernt und ein neuer, leerer pcustom Node wird zwischen Partikel-Emitter und Partikel-Renderer eingefügt. Der Partikel-Renderer verbleibt mit seinen vorigen Einstellungen. Der Emitter Node bekommt etwas andere Werte. Es sollen lediglich 10 Partikel am Frame 0 erzeugt werden. Nicht vergessen: Die Lebensdauer etwas erhöhen und im Tab „Stil" den unterschiedlichen Blobs zufällige Größenunterschiede zuweisen. Die Initialgröße bleibt auf 0,05, einem sehr kleinen Wert, während der Größenunterschied „Size Variance" auf einen Wert von 0,125



Die Animation der Partikel erfolgt rein über Funktionen.

Die Menge und Darstellung der Partikel sowie der Raum, in dem die Partikel erzeugt werden, müssen vorab angepasst werden.

Vor dem Rendering noch die passende Kamera einstellen und die Transparenzeinstellungen anpassen.

geändert wird. Innerhalb des neuen pcustom Nodes wird zunächst unter dem Tab „Inter" im ersten Feld die bekannte Expression id-2 hinterlegt – sollte es vorkommen, dass ein Partikel stillsteht, muss die Expression auf id-1 abgewandelt werden. Wahlweise kann der fixe Wert 2 durch eine Variable ersetzt werden.

Zunächst werden innerhalb des Tabs „Particle" nur die Felder für die X- und Y-Position betrachtet. Im Feld für X wird als Expression sin(i1*time) eingetragen, im Feld für Y cos(i1*time). Man nimmt Bewegungen innerhalb einer runden Laufbahn wahr. Das liegt daran, dass die Expressions die Beschleunigungen der jeweiligen Partikel basierend auf der Nummer ihrer ID festlegen. Die zuvor eingestellten Größenunterschiede geben nochmals ein wenig mehr Detail. Sollte auf eine zufällige Verteilung der Größe kein Wert gelegt werden und stattdessen die Größe der ID entsprechend angepasst werden, dann geht das im pcustom Node im Feld namens „Size".

Cosinus, Sinus, Feuerball?

Dort muss die ID mit einem passenden Wert belegt werden. So kann zum Beispiel den weiter außen liegenden Partikeln mehr Größe gegeben werden. Das nun entstehende Partikelsystem basiert aber derzeit nur auf den vom Emitter erstellten 10 Partikeln. Da der Effekt für ein Display sein soll, darf unter dem Feld „Position Z" anstelle der Variablen pz eine Null eingetragen werden. Somit sind alle Partikel fest auf der Z-Achse verankert. Im Feld für die Y-Position ist noch immer die Expression cos(i1*time) hinterlegt. Diese wird mit einer Expression multipliziert, die folgendermaßen aufgebaut ist: cos(i1*20). Komplettiert sollte die Expression wie folgt aussehen: cos(i1*20)*cos(i1*time). Die Zahl 20 ist für einen ersten Versuch fest eingetragen, da der Wert aber im Position-X-Feld ebenfalls zum Einsatz kommt, bietet es sich an, eine neue Variable hierfür zu erzeugen. Betrachtet man das Ergebnis bis jetzt, dann nimmt man im Viewer/Monitor quasi einen Kreis von der Seite wahr. In die Richtung soll der gesamte Aufbau gehen. Es soll kein dreidimensionaler Partikelkreis entstehen, dafür wurde der Position in Z bereits der Wert Null gegeben. Was aber gemacht wird, ist, dass die Kreislaufbahn in der X-Achse so verändert wird, dass ein vertikal angeordnetes Auge aus Partikeln entsteht.

Hierzu wird als nächstes im Feld für die Position X die dort bereits stehende Expression sin(i1*time) – die mit dem Pendant aus dem Position-Y-Feld für die runde Laufbahn sorgt – mit einer weiteren Expression multipliziert. Diese neue Expression lautet sin(i1*20).

Die Zeile muss also folgende vollständige Expression beinhalten: sin(i1*20)*sin(i1*time).

Betätigt man nun den Play-Button, dann zeigt der Viewer/Monitor quasi eine Raute an, die in Wirklichkeit aus einer Vielzahl von Partikeln besteht, die ellipsenförmigen Laufbahnen folgen. Der Weg ist bis dato fast fertig für das Display. Damit nun das zuvor erwähnte, vertikal angeordnete Auge entstehen kann, in dessen Mitte man sich mit einem Fingerabdruck registrieren kann, muss jedoch eine weitere Expression vorne angefügt werden. Die letzte Expression sieht wie folgt aus: cos(i1*20) und wird mit der restlichen Expression multipliziert. Das Resultat im Feld „Position X Expression" sieht wie folgt aus: cos(i1*20)*sin(i1*20)*sin(i1*time). Die komplette Expression aus dem Feld „Position Y Expression" sieht so aus: cos(i1*20)*cos(i1*time).

Betätigt man nun den Play-Button in der Zeitleiste, dann nimmt man das vertikal angeordnete Auge wahr. Es empfiehlt sich, an dieser Stelle noch eine symbolische Stellschraube für die Geschwindigkeit der Partikelbewegungen einzubringen, um die störende Unruhe zu beseitigen. In den Teilen der Expressions, in denen die Variable namens „time" untergebracht ist, muss das Ergebnis der jeweiligen Multiplikation (i1*time) durch eine neue Variable namens „speed" dividiert werden. Die Variable „speed" kann bei den anderen Variablen im User-Tab abgelegt werden und sollte nicht mit Integer-Werten ausgestattet sein. Für die Geschwindigkeit oder besser gesagt das Feintuning der Geschwindigkeit bietet sich eine Float-Variable bestens an.

Als Quasi-Aufräumaktion könnte man nun den fixen Wert 20 durch eine Variable ersetzen, die Integer-Werte bereithält. So lassen sich unterschiedliche Effekte in einem weitaus schnelleren Zeitraum evaluieren, als den fixen Wert von Hand zu ersetzen. So erlebt man nicht einfach nur einen ansehnlichen Effekt, sondern kann sich der mathematischen Schönheit erfreuen.

Eine Vielfalt an Effekten mit Trigonometrie

Die in diesem Workshop aufgeführten Vorgehensmaßnahmen dienen rein dazu, dem neuen Anwender einen ersten Einblick zu geben und zu zeigen, wie kraftvoll der pcustom Node ist und wie man mit diesem Node eine Vielzahl an unterschiedlichen Effekten generieren kann, ohne auch nur eine Variable über Keyframes zu animieren.

Jeder Anwender und jede Anwenderin, die den Workshop verfolgt, ist dazu aufgerufen, jegliche Werte zu verändern und selbst imposante Effekte zu erstellen. Ein weiterer wichtiger Punkt ist, das Große und Ganze zu sehen. Der Fokus in diesem Workshop lag stark auf dem pcustom Node, der innerhalb der Partikel-Menüs aufzufinden ist. Aber gerade wenn die Partikelanimation beziehungsweise -simulation ihren Zweck erfüllt hat, geht es direkt weiter mit dem Ausbau der Effekte. So könnte man direkt nach dem Particle Render Node mittels Displacement Nodes Warping hinzufügen oder per Trail Node jedem Partikel beziehungsweise Blob einen Schweif gegen im Bezug zu Sternenschweif. Spaß und Spannung ist mit Sicherheit garantiert.

Ein kleiner Tipp am Schluss. Unabhängig vom Effekt des vertikal angeordneten Auges: Wenn man innerhalb des Tabs namens „Particle" im Feld für die Größe eine Expression wie zum Beispiel id*0.001 eingibt – wobei letzterer Wert ebenfalls durch eine Variable ersetzt werden kann – erhält man ein spannendes Ergebnis, das die Laufbahnen der unterschiedlichen Partikel besser nachverfolgen lässt. **› ei**