2013
7
DIGITAL PRODUCTION
ISSN 1433-2620 > B 43362 >> 17. Jahrgang >>> www.digitalproduction.com
Published by ATEC

Deutschland € 14,95
Österreich € 17,–
Schweiz sfr 23,–

DIGITAL PRODUCTION

MAGAZIN FÜR DIGITALE MEDIENPRODUKTION

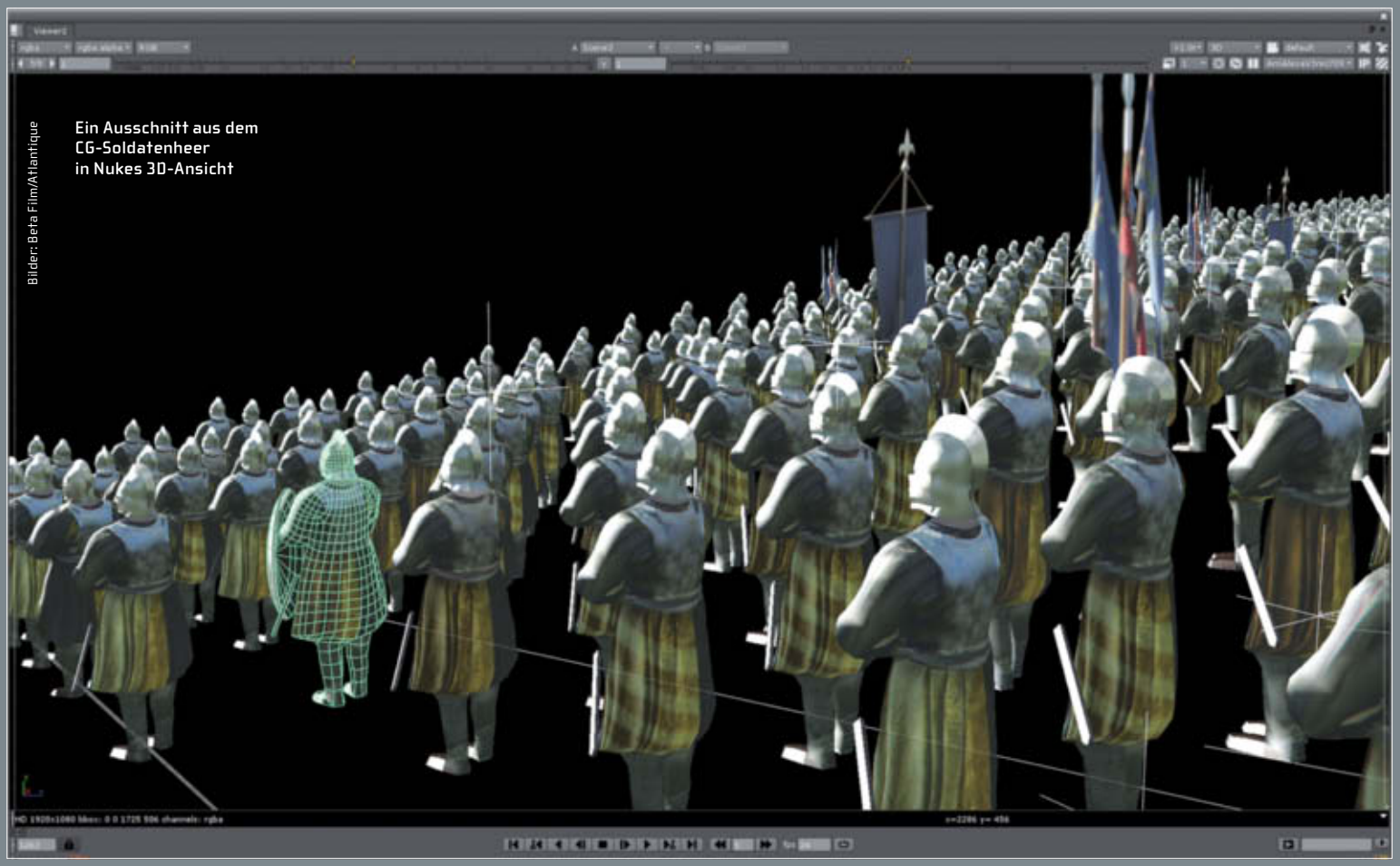
NOVEMBER | DEZEMBER 07|13

Games
Cryengine, Ingame Cinematics, Maya LT 2014 und mehr ...

FMX 2014
„Rugbybugs“ vorneweg im FMX-Trailer

4K-Grundlagen
Gerüstet für den nächsten Hype

4 194336 214951 07



Bilder: Beta Film/Atlantique

Ein Ausschnitt aus dem CG-Soldatenheer in Nukes 3D-Ansicht

Aus 1 mach 600

Die TV-Serie „Borgia" läuft in Deutschland im ZDF und ist mit rund 25 Millionen Euro Budget eine der teuersten europäischen TV-Produktionen. Für die zweite Staffel erstellte Rise | Visual Effects Studios neben vielen weiteren VFX-Arbeiten eine Crowd-Replication mit CG-Soldaten. Lead Compositor und DP-Autor Rayk Schroeder zeigt, wie diese Aufgabe mit Python-Skripten für Nuke im Produktionsalltag vereinfacht werden konnte und welche Verbesserungen für diesen Prozess darüber hinaus möglich wären.

von Rayk Schroeder

Nachdem 2011 die erste Staffel der TV-Serie „Borgia" ein voller Erfolg bei den Zuschauern war, wurde eine zweite Staffel produziert. Mit dabei waren wieder Joachim Grüninger als VFX-Supervisor von Atlantique Productions und Rise | Visual Effects Studios als Main Vendor für die Visual Effects. Der VFX-Umfang erstreckte sich wie bereits in der ersten Staffel über ein weites Spektrum an Aufgaben. Im folgenden Artikel soll eine Crowd-Replication mit CG-Soldaten genauer erklärt und anhand einiger einfacher Scripting-Beispiele sollen die Möglichkeiten von Nuke als Compositing-System im Zusammenhang mit Python gezeigt werden.

Die Serie und ihre VFX

Den Auftakt der zweiten Staffel bildet eine aufwendige Explosionssequenz einer Statue, in die ein Blitz einschlägt. Weiterhin wurden verschiedene Establisher Shots von einzelnen Städten mit Mattepaintings gelöst und die Drehorte in das alte Italien verwandelt. Dabei war die Hauptaufgabe die Retusche moderner Elemente, die Erweiterung einzelner Gebäude und Landschaftszüge. Die Innenansichten der Sixtinischen Kapelle und des sogenannten „Throne Rooms" konnten dank vorhandener Setups aus der ersten Staffel leicht umgesetzt werden. Von Pointcloud9 erstellte LIDAR-Scans vereinfachten die Matchmoves und Set Extensions.

Die Serie wurde auf der Arri-Alexa-Kamera gedreht und Rise erhielt die Alexa-Quicktime-Files zusammen mit einer EDL. René Grasser war als VFX-Production-Manager für den korrekten Export und Import der Daten aus und in Hiero von The Foundry verantwortlich und fütterte die hausinterne Projektdatenbank Rise | Base mit entsprechenden Informationen und Änderungswünschen zu den einzelnen Shots. Zu seinen weiteren Aufgaben gehörten Deliveries und die zeitliche Projektplanung. Da die Bearbeitung der ersten Episoden bereits begann, während

noch für weitere Episoden gedreht beziehungsweise geschnitten wurde, lief die Shot-Abnahme über regelmäßige RV-Sessions und Skype-Konferenzschaltungen mit Joachim Grüninger.

Die Crowd-Replication

Einer der ausgefalleneren Shots war die Erweiterung einer kleinen, 40 Mann starken Soldatengruppe, die auf einem Schlachtfeld steht, zu einem riesigen Heer – und das Ganze mit bewegter Kamera ohne Motion-Control-System. Der Plan war, dies mithilfe von CG-Soldaten zu lösen.

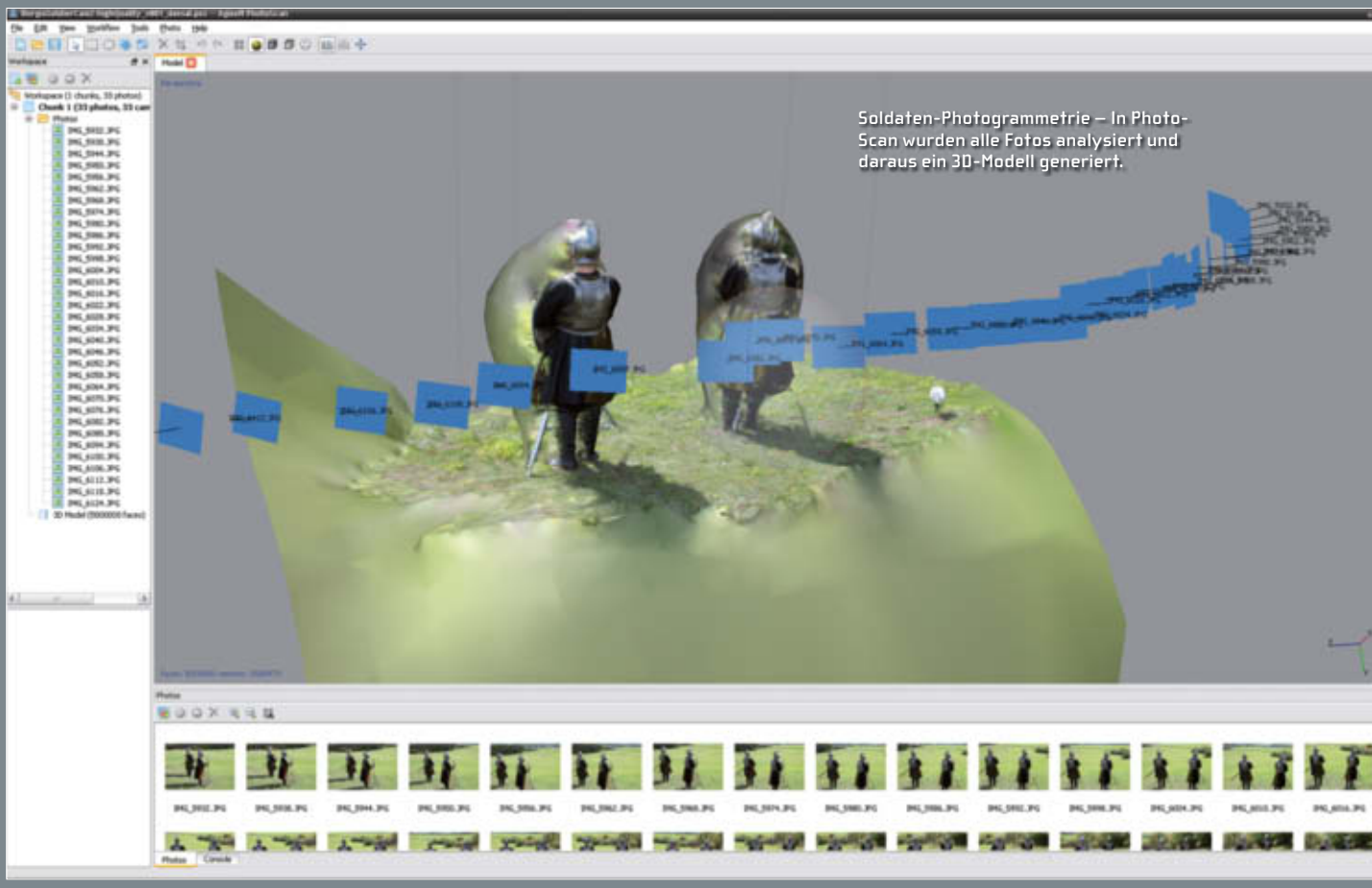
Eine große Menge von Statisten ist bei einem Filmdreh sehr teuer und aufwendig zu koordinieren, deshalb sind Crowd-Replications ein beliebtes Einsatzgebiet von Visual Effects. Dadurch entfällt das zeitaufwendige Umstellen und neue Einrichten der Statisten und der Drehplan bietet mehr Raum für die restlichen zu drehenden Einstellungen. CG-Crowds kann man am Computer hin- und herschieben. Somit bieten sie Flexibilität, sodass sogar im Nachhinein Anzahl, Position, Aussehen und Verhalten der virtuellen Statisten verändert werden können.

Am Set wurde Joachim Grüninger bei Bedarf von VFX-Supervisor Markus Degen und weiteren Mitarbeitern von Rise und Pointcloud9 unterstützt. Für diesen Shot fertigte das Team einen LIDAR-Scan des Schlachtfelds an. Für die späteren CG-Soldaten wurden zwei Statisten aus verschiedenen Kameraperspektiven fotografiert.

Das Team drehte mehrere Plates: als Erstes die bewegte Action-Plate mit beiden Hauptdarstellern, der Soldatengruppe an einer Position und einigen Pferden. Danach wurden die Soldaten und Pferde umpositioniert und mehrere Plates mit ihnen an unterschiedlichen Positionen gedreht. Die Kamera war dabei locked und wurde bei jeder Plate unterschiedlich eingerichtet, um die Soldatengruppe aus einer ungefähr passenden Perspektive zu drehen.

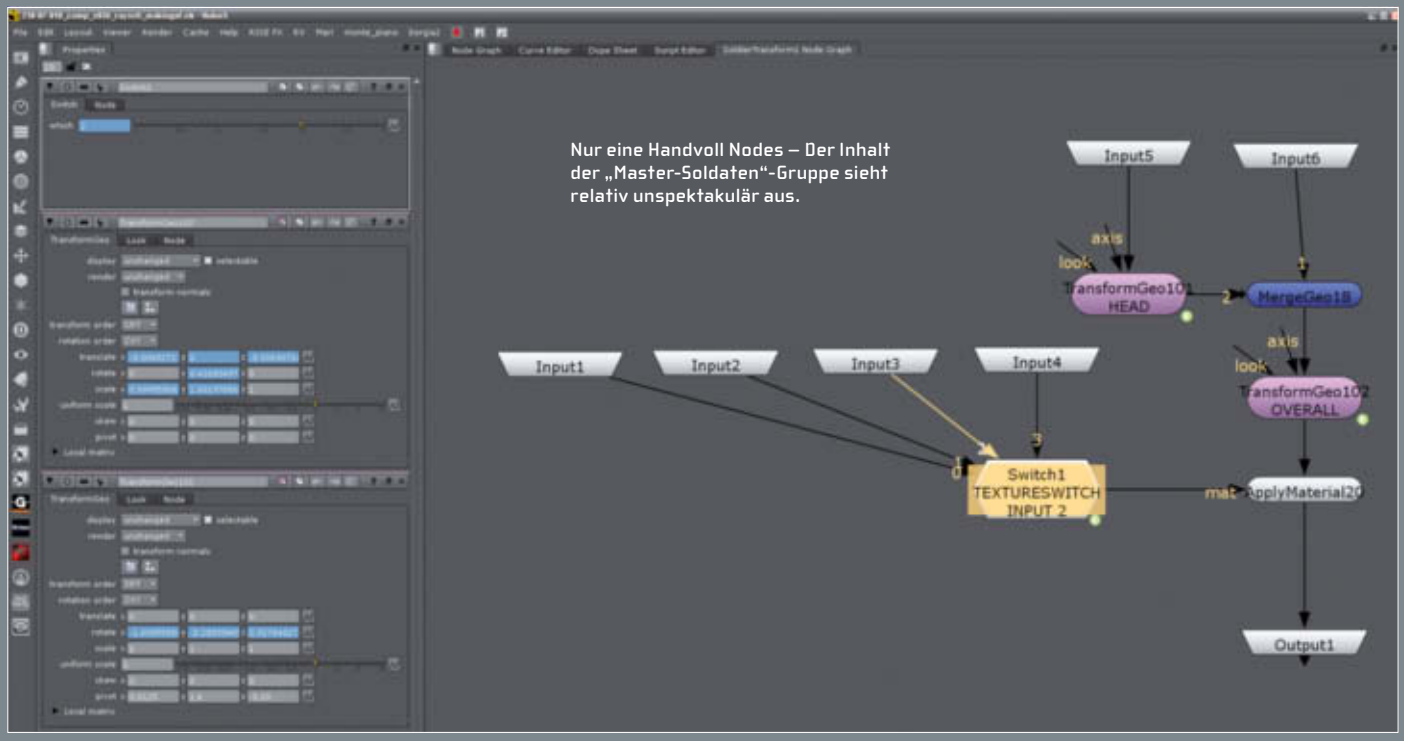
Ursprünglich sollte der Kameraschwenk als Nodal-Schwenk gedreht werden, um keine Parallaxe zu sehen und so die Bearbeitung zu

Zwei Soldaten stehen Pose – Am Set wurden zwei Soldaten aus unterschiedlichen Perspektiven fotografiert, um daraus später ein 3D-Modell zu erstellen.



Soldaten-Photogrammetrie – In PhotoScan wurden alle Fotos analysiert und daraus ein 3D-Modell generiert.

Die Grundlage für die CG-Soldaten – Alle drei Nodes zur Erstellung und Manipulation der CG-Soldaten und ihre Properties



Nur eine Handvoll Nodes – Der Inhalt der „Master-Soldaten"-Gruppe sieht relativ unspektakulär aus.

vereinfachen. Allerdings war der Nodal-Kopf für die Kamera nicht ganz richtig eingestellt, sodass doch ein wenig Parallaxe auftrat. Das machte den Matchmove in Syntheyes etwas aufwendiger.

Bevor das Feld mit Soldaten gefüllt werden konnte, musste die Action-Plate von verschiedenen Setbauteilen bereinigt werden: Die Registrierungskugeln für den LIDAR-Scanner waren zu sehen und quer durch das Bild liefen die Dolly-Schienen, die schon für die nächste Kameraeinstellung aufgebaut waren. Mittels auf Cards projizierter Patches war das aber einfach und schnell erledigt. Auch wenn auf dem Feld Gras wuchs, reichte das nicht aus, um die einzelnen Soldaten und Pferde keyen zu können. Somit war für die Soldaten und Pferde mit all den wehenden Fahnen und Pferdeschwänzen ein aufwendiges Rotoscoping notwendig. Die Soldatenreihen sollten hinter einem Baum weitergehen und dort langsam verschwinden. Dazu wurde am Set ein Bluescreen hinter den Baum gestellt, damit eine Trennung des Baums vom Hintergrund möglich war.

Die gelockten Soldaten- und Pferde-Plates konnten auf Spheres projiziert werden, dabei boten die LIDAR-Kugeln und Dolly-Schienen hervorragende Referenzpunkte zur Einrichtung der Projektionskameras. Deren Position und Rotation wurden so lange korrigiert, bis die Schienen und Kugeln deckungsgleich mit denen aus der Action-Plate waren (siehe rechte Bildabfolge).

CG-Soldaten in Nuke

In einem Heer sehen die Soldaten trotz ihrer gleichen Uniform immer ein wenig unterschiedlich aus: Der eine ist etwas größer, ein anderer hat den Kopf etwas mehr zur Seite geneigt, beim Nächsten ist der Helm nicht ganz so auf Hochglanz poliert wie bei allen anderen usw. Diese kleinen Unterschiede sollten umgesetzt werden, ohne dass jeder Soldat einzeln bearbeitet werden muss.

Aus den Fotos der Statisten wurde mittels Photogrammetrie (Anm. d. Red.: Mess-

Die Source-Plate – Der Kameraschwenk als Panorama

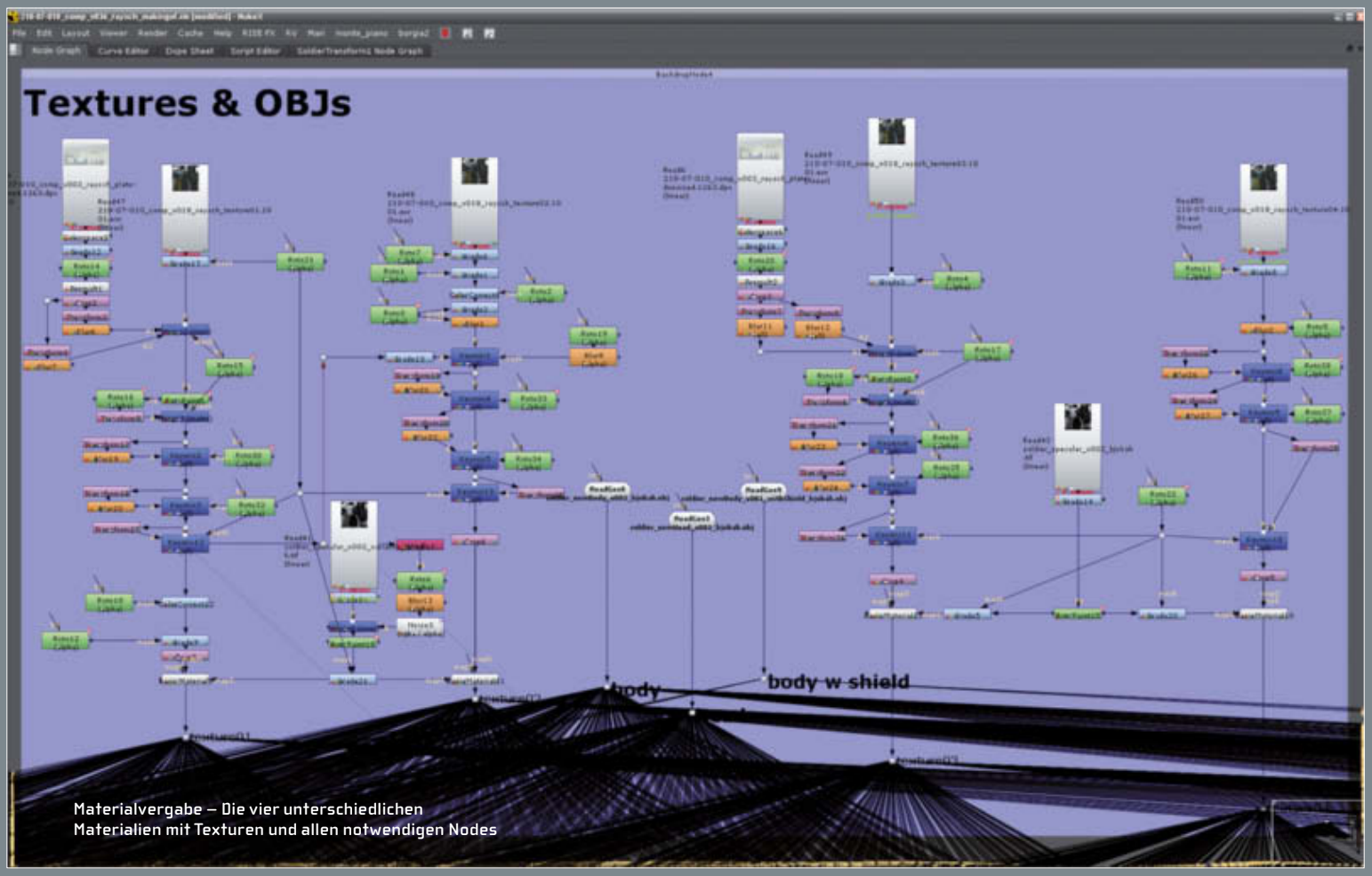
Retuschen – Alle nicht benötigten Setelemente wurden entfernt.

Mehr Pferde – Die Pferde aus den Crowd-Plates wurden mehrfach verwendet.

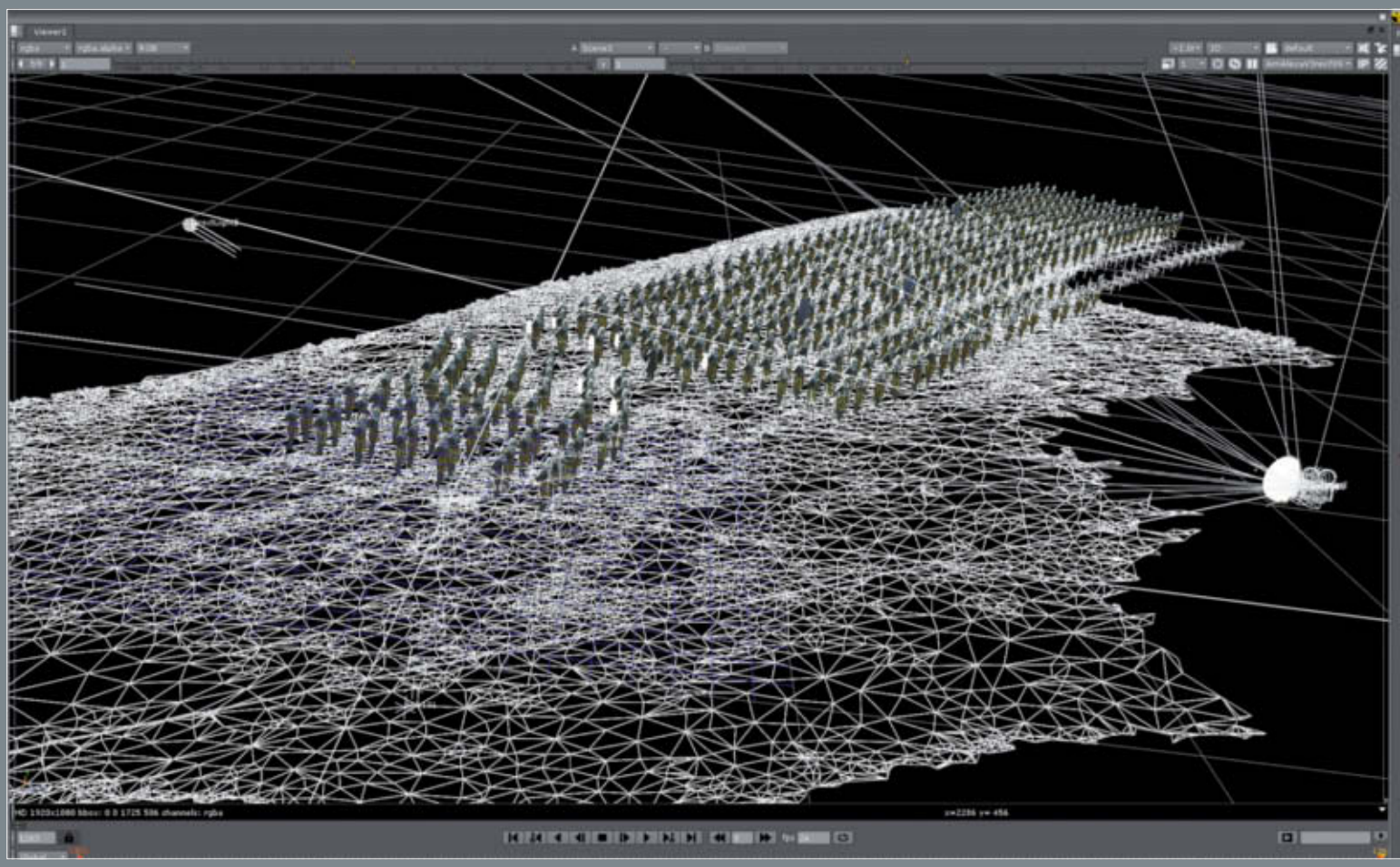
CG-Soldaten – Das Feld wurde mit den CG-Soldaten aufgefüllt.

Crowd-Plates – Die gedrehten Crowd-Plates wurden auf Spheres projiziert und im 3D-Raum positioniert.

Finales Compositing – Der finale Shot mit den Originalsoldaten im Vordergrund



Materialvergabe – Die vier unterschiedlichen Materialien mit Texturen und allen notwendigen Nodes



methode, um aus Fotografien eines Objekts seine dreidimensionale Form zu bestimmen) in dem Programm PhotoScan ein 3D-Modell erstellt, das zur weiteren Bearbeitung nach Maya exportiert wurde.

In Maya glättete ein 3D-Artist das importierte 3D-Objekt und reduzierte die Polygone, indem er unter anderem alle Polygone auf der Vorderseite löschte, da die Soldaten nur von hinten zu sehen sein sollten. Danach trennte er Kopf und Körper voneinander, um sie als Einzelobjekte bearbeiten zu können. Weiterhin wurde noch jeweils eine Variante mit und ohne Schutzschild erstellt.

Schließlich wurden aus Maya drei OBJs exportiert: der Kopf, der Körper mit Schild und der Körper ohne Schild. Für noch mehr Variationsmöglichkeiten wurden vier unterschiedliche Texturen mit dazu passenden Specular Maps erstellt.

Als Nächstes ging es daran, in Nuke eine „Master-Soldaten"-Gruppe für die Soldaten zu erstellen, diese sollte die Materialzuweisung und Transformationsvariationen für alle zukünftigen Soldaten beinhalten. Zur leichteren Weiterverarbeitung wurden alle notwendigen Nodes in einer Gruppe zusammengefasst, die Geometrien sollten für alle Soldaten nur einmal geladen werden. Deshalb erhielt die „Master"-Gruppe insgesamt sechs Inputs: vier für die verschiedenen Materialien, einen für die Kopf-Geometrie und einen für die Körper-Geometrie. Um den Kopf unabhängig vom Körper bewegen zu können, bekam er eine eigene TransformGeo-Node. Danach wurden die beiden Input-Geometrien miteinander gemerged und noch mit einer gemeinsamen TransformGeo-Node versehen.

Da die Soldaten etwas unterschiedlich aussehen sollten, brauchten sie natürlich auch unterschiedliche Materialien. Bei mehreren Hundert Soldaten ist es schwierig, in der Node-Ansicht den Überblick zu behalten. Deshalb wurden die vier Material-Inputs mit einem Switch verbunden. Mithilfe einer Expression schaltete der Switch auf einen der vier Material-Inputs per Zufall um, sodass jeder erstellte Soldat eines der vier vorhandenen Materialien zugeordnet bekam.

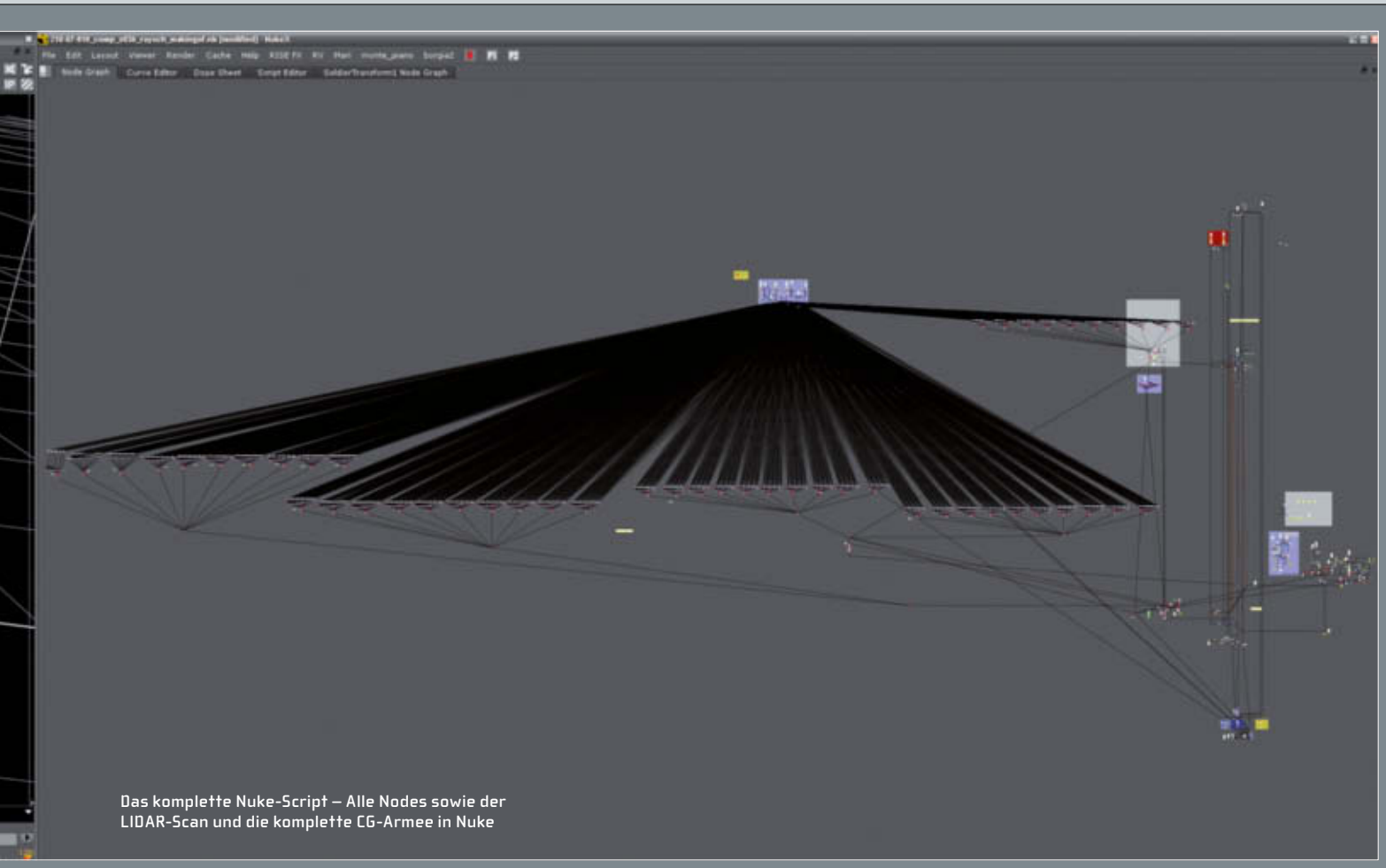
Die Expression war eine einfache random()-Funktion. Diese Funktion benötigte aber einen Zahlenwert als Basis, um einen Zufallswert zu generieren. In der Regel verwendet man für bewegte Zufallsfunktionen den aktuellen Frame. Da sich die Materialien natürlich nicht ändern sollten, brauchte man einen Zahlenwert, der sich nicht ändert. Wenn in Nuke eine Node erstellt wird, hat diese immer eine Zahl im Namen. Wird diese Node kopiert, ändert sich auch die Zahl – genau dies brauchte man für den Zufallswert der random()-Funktion. Um die Zahl aus dem Node-Namen zu extrahieren, wurde die random()-Funktion mit einer Python-Abfrage kombiniert. Am Ende hatte der Switch die folgende Expression:

```
fmod(int(random([python filter
(lambda x: x.isdigit(), nuke.thisPa-
rent().name())])*3),4)
```

Die Python-Abfrage extrahierte die Zahl aus dem Node-Namen der „Master"-Gruppe und die random()-Funktion erzeugte daraus einen Zufallswert zwischen 0 und 1. Da der Switch zum Umschalten aber Werte zwischen 0 und 3 benötigte, ist der Zufallswert mit 3 multipliziert worden.

Der vorliegende Float-Wert wurde mit der int()-Funktion in einen Integer-Wert umgewandelt, indem der Zufallswert zur nächsten vollen Zahl entweder auf- oder abgerundet wurde. Als Letztes rundete die fmod()-Funktion die komplette Expression ab und führte eine Division mit Rest durch. Sie lieferte einen Wert zwischen 0 und 3 und schaltete den Switch zu dem entsprechenden Input.

Die Transform-, Rotate- und Scale-Knobs der allgemeinen TransformGeo-Node sowie die Rotate-Knobs für die Kopf-Rotation wurden ebenfalls mit einer ähnlichen Expression versehen, sodass durch eine zufällige Abweichung die Soldaten minimal anders positioniert, rotiert und skaliert sind. Diese



Das komplette Nuke-Script – Alle Nodes sowie der LIDAR-Scan und die komplette CG-Armee in Nuke

Expression erhielt zusätzlich zum Zufallswert einen Faktor, der später noch verändert werden konnte. Dazu wurde eine NoOp-Node mit entsprechenden User-Knobs für die notwendigen Zufallswerte angefertigt, alle erstellten Soldaten waren mit dieser Node über die Expression verbunden. So konnte man bei allen Soldaten die Transformationswerte für den Kopf und den gesamten Körper mit einem Zufallswert versehen und so noch mehr Varianz in das Heer bringen.

Die Produktion der Soldaten sah so aus, dass die „Master-Soldaten"-Gruppe vervielfältigt und automatisch mit den Materialien und ReadGeo-Nodes der Soldaten-Geometrie verbunden werden sollte. Dazu war ein etwas längeres Python-Script notwendig. Es wurde wieder eine NoOp-Node erzeugt, diesmal mit einem Eingabefeld für die Anzahl der zu generierenden Soldaten und einem Python-Button. Der Python-Button führte ein Script aus, das die selektierten Nodes so oft duplizierte, wie in dem Textfeld angegeben war, das heißt vor dem Drücken des Buttons musste die „Master-Soldaten"-Gruppe selektiert werden.

Um die Programmierung etwas zu vereinfachen, wurde nach den ganzen notwendigen Nodes für die Materialbearbeitung (Color Correction et cetera) und ReadGeo-Nodes jeweils ein Dot mit eindeutigem Namen eingefügt: „Texture01", „Body", usw.

Dadurch konnte in dem Python-Script einfach darauf zugegriffen werden, auch falls später noch andere Nodes notwendig gewesen wären. Sie hätten einfach nur vor dem Dot hängen müssen.

```
bodyObj = nuke.toNode(‚Body‘)
headObj = nuke.toNode(‚Head‘)
texture01 = nuke.toNode(‚Texture01‘)
texture02 = nuke.toNode(‚Texture02‘)
texture03 = nuke.toNode(‚Texture03‘)
texture04 = nuke.toNode(‚Texture04‘)
this = nuke.thisNode()

for n in
range(int(this[‚soldierAmount‘].va-
lue())):
 nuke.nodeCopy(„R:\\temp\\myclip-
board.nk“)
 for i in nuke.selectedNodes():
 i.knob(‚selected‘).setValue(False)

 nuke.nodePaste(„R:\\temp\\myclip-
board.nk“)
 for i in nuke.selectedNodes():
 nuke.autoplace(i)
 i.setInput(0,texture01)
 i.setInput(1,texture02)
 i.setInput(2,texture03)
 i.setInput(3,texture04)
 i.setInput(4,headObj)
 i.setInput(5,bodyObj)
```

Um nun die benötigte Anzahl von Soldaten zu duplizieren, sind wir immer reihenweise vorgegangen: Es wurde die „Master-Soldaten"-Gruppe selektiert und über das Python-Script dupliziert. Anschließend haben wir an jeden neu erstellten Soldaten TransformGeo-Nodes zur eigentlichen Positionierung der Soldaten auf dem Feld gehängt.

Diese TransformGeo-Nodes waren alle mit der gleichen Axis vebunden, sodass darüber die komplette Soldatenreihe verschoben werden konnte. Jede Reihe ist noch zu einer Scene zusammengefasst worden – war ein Block vollständig, wurden dessen Reihen auch zu einer Scene zusammengefasst, bevor sie mit dem Scanline-Renderer verbunden wurden. So hat Rise schließlich fast 600 Soldaten erstellt und auf dem Feld positioniert.

Der LIDAR-Scan vom Feld diente dabei als Positionierungs- und Orientierungshilfe und war sehr hilfreich, um noch einige Cards mit rotoskopierten Fahnen und Bannern zwischen die Soldaten zu stellen.

Für die richtigen Highlights auf den Rüstungen wurde ein Directional Light gesetzt, damit realisierten wir auch die Schatten der am Rand eines Blocks stehenden Soldaten. Allerdings reichten hier als Projektionsfläche zwei Cards, die an der ungefähren Stelle des Feld-Scans lagen – dadurch wurde Nuke nicht unnötig belastet.

Verbesserungsmöglichkeiten

Geübte Python-Programmierer werden bei den hier vorgestellten Scripts ziemlich schnell Verbesserungsmöglichkeiten entdecken und einfachere Wege kennen. Denn gerade der Umweg, die Zwischenablage in eine temporäre Nuke-Datei zu schreiben, ist eher quick and dirty und nicht sehr flexibel, da das im Computer vorhandene Laufwerk bekannt sein muss. Doch in diesem Artikel sollte gezeigt werden, wie man sich als Compositor auch mit wenigen Python-Kenntnissen das Leben vereinfachen kann. Und da kann auf solche einfachen und schnellen Mittel zurückgegriffen werden, da diese Scripts auf der eigenen Workstation ausgeführt werden.

Im Produktionsalltag mit mehreren zu bearbeitenden Shots hat man als Artist meist keine Zeit, ein vollständig ausgereiftes Tool nebenbei zu erstellen. Dazu bedarf es mehr Zeit und einiger Testläufe. Deshalb ist verständlich, dass auch bei uns nach Projektabschluss noch viele Verbesserungsideen kamen.

So könnte zum Beispiel der Umweg über das temporäre Nuke-Script komfortabler gelöst werden oder die Erstellung der nachfolgenden Transform- und Scene-Nodes. Auch der Switch mit den vier Material-Inputs wäre nicht unbedingt notwendig: Stattdessen könnte direkt im Python-Script eine zufällige Auswahl des zu verwendenden Materials stattfinden und so nur ein Material-Input gebraucht werden. Ebenso könnte man noch eine Funktion einbauen, die es ermöglicht, eine bestimmte Anzahl an Soldaten mit Schutzschild zu definieren – dies wurde in der vorgestellten Version noch per Hand erledigt. Auch dass die „Master-Soldaten"-Gruppe erst selektiert werden musste, ist verbesserungswürdig.

Von einer automatischen Lösung ähnlich Nukes Partikelsystem ist die vorgestellte Lösung also noch weit entfernt.

Fazit

Als Compositor muss man oft viel Zeit in die Entwicklung eines funktionierenden Scripts stecken, da man meist mehrere Shots bearbeitet und sich auf die grundlegenden Arbeiten konzentrieren muss. Letztendlich macht sich die investierte Zeit jedoch dadurch bezahlt, dass man, wie in diesem Beispiel, mehrere Tage Positionierungs- und Soldaten-Erstellungsarbeit spart.

Während aktuell die zweite Staffel ausgestrahlt wird, sitzt das Team von Rise | Visual Effects Studios bereits an der Bearbeitung der dritten Staffel. Und auch hier konnte das Soldaten-Script für die schnelle Erstellung erster Layouts schon eingesetzt werden.

› mf

Rayk Schroeder ist VFX-Supervisor und Lead Compositor bei Rise | Visual Effects Studios. www.risefx.com
Für „Borgia Season 2" war er als Lead Compositor tätig und ist auch bei Season 3 wieder dabei.

animago 2013

DAS SPECIAL

published by DIGITAL PRODUCTION

- **animago AWARD 2013:** Die Making-ofs der Sieger und die weiteren Projekte der Nominierten
- **animago CONFERENCE 2013:** Highlights & Key-Speaker und viele Hintergrundinformationen
- **Große Still-Galerie:** Eine Vielzahl eingereichter Arbeiten aus aller Welt

Jetzt bestellen: www.digitalproduction.com/animago

Veranstaltet von: DIGITAL PRODUCTION

Gefördert durch: medienboard Berlin-Brandenburg

www.animago.com
animagoAWARD

Bilder: animago AWARD & CONFERENCE 2013

Gesponsert von:

4Dgraphic · AUFS MAUL · AUTODESK · CADnetwork · DOSCH DESIGN · D-E · JOHN ROBIE · MAXON · mStore · PLAZZ · PNY · POMMERY · pond5 · wacom