1. Grundlagen

2. Projektidee

Ziel des Projekts ist es, einen Landschaftspinsel bereitzustellen, welcher prozedural verschiedene Biome erzeugt, darunter Wüsten, Wälder und Schneelandschaften. Der Nutzer soll die Möglichkeit haben, bestimmte Bereiche seines Mesh‘ anzumalen; je nach Farbe wird ein anderes Biom inklusive passender Objekte sowie einer passenden Textur erzeugt. Besonderer Fokus liegt auf der Verteilung der Objekte, vor allem an den Übergängen zwischen zwei Biomen. Der Landschaftspinsel soll als Blender Add-On bereitgestellt werden.

3. Architektur

Zum Verteilen der Objekte benutzen wir das Partikelsystem von Blender. Dieses generiert je nach Einstellungen eine unterschiedliche Menge an Haarpartikeln, welche wir als Ursprung für unsere Objekte benutzen. Da das Partikelsystem von Blender nur auf ganze Objekte und nicht auf selektierte Teile anwenden lässt, gestaltete sich die Umsetzung als Pinsel als sehr schwierig. Letztendlich haben wir uns dazu entschlossen, das Gebiet auf welches die Objekte verteilt werden nicht durch einen Pinsel zu definieren, sondern stattdessen nur ganze Flächen als Grundlage für ein Biom zu verwenden. Die selektierte Fläche wird dazu vom restlichen Mesh getrennt, anschließend wird dem neu entstandenen Objekt eine Textur hinzugefügt. Dazu benutzen wir eine von und bereitgestellte Textur sowie das unwrapping-Werkzeug von Blender.

/\*\*

Hier was zum Verteilen der Objekte

\*/

Schlussendlich werden die generierten Objekte als Kind-Objekte angehängt und die zuvor abgetrennte Fläche wieder dem Ursprungsobjekt hinzugefügt.

4. Implementation

Zuerst werden die von uns bereitgestellten Objekte in Blender geladen. Wir benutzen dazu für jedes Biom eine andere vorgefertigte Bibliothek, die aus den Objekten sowie einer Bodentextur besteht. Die Bibliotheken folgen alle der gleiche Namenskonvention, sodass nach Selektion des Bioms automatisch die richtigen Objekte ohne weitere Abfragen hinzugefügt werden. Anschließend wird das selektierte Mesh vom restlichen Objekt getrennt. Dies ist leider nötig, weil das Partikelsystem von Blender nur auf ganze Objekte und nicht auf Teile davon (wie zum Beispiel einzelne Flächen) angewendet werden kann.

Leider gibt das Blender Werkzeuge zur Separation von Mesh nicht das separierte, neu entstandene Objekt zurück, weswegen wir Alternativlösungen in Anspruch nehmen mussten. Vor der Anwendung der Operationen werden alle Objekte in der Szene in eine Liste kopiert, welche anschließend verglichen wird mit den Objekten, die sich nach der Operation in der Szene befinden – das Objekt welches nur in der neuen Liste vorhanden ist muss entsprechend das neue, separierte Objekt sein. Leider sind derlei Operationen in Blender oftmals nötig, weil die Operationen alle auf die graphische Nutzeroberfläche auslegt sind, womit eine Selektion einfach durch Anklicken des entsprechenden Objekts erfolgen kann. Im Skript ist das allerdings natürlich nicht möglich.

Anschließend wird dem neu erzeugten Objekt ein Material mit Textur hinzugefügt. Wir haben uns hierbei dafür entschieden, das Node-basierte Materialsystem des Cycle-Renderers zu benutzen. Wir laden hierzu die zum Biom passende Textur und fügen dem Material eine Input-, eine Output- und eine Principled BSDF-Node hinzu. Zuletzt benutzen wir die unwrap-Funktion von Blender und fügen dem Objekt das konfigurierte Material hinzu.

Als nächstes werden dem Objekt die Haarpartikel hinzugefügt. Dazu haben wir Standard-Einstellungen definieren, wobei sich die Werte hauptsächlich durch Tests ergeben haben.

/\*\*

Hier Tommy was über die Verteilung

\*/

Anschließend werden die generierten Objekte einer Gruppe hinzugefügt, damit diese vom Nutzer leichter verwendet oder auch wieder gelöscht werden können. Zuletzt werden die importierten Objekte sowie das Partikelsystem wieder aus der Szene entfernt, außerdem wird das vorher getrennte Objekte wieder mit dem Ursprungsobjekt zusammengefügt.

/\*\*

Hier Niko was über AddOn

\*/

5. Reflektion

Durch die intensive Projektarbeit konnten wir einige Erfahrungen sammeln und gleichzeitig Inspirationen für zukünftige Objekte gewinnen.

Bisher beschränkte sich unsere Erfahrung mit Blender nur auf den graphischen Modellierungsbereich, weswegen sich zuerst Adaptionsprozess vollziehen musste. Wir haben Blender damals schon als Programm mit sehr guter und umfänglicher Funktionalität kennenlernt, welches allerdings wie einige Open Source-Projekte etwas unter einer nicht einheitlichen Oberfläche leidet. Dieser Eindruck wurde beim Verwenden der Python-API bestätigt. Alle wichtigen Funktionen sind vorhanden, diese funktionieren auch meistens wie erwartet, allerdings ist mitunter schwierig zu erahnen, was erwartet ist. Es ist zum Beispiel von enormer Wichtigkeit, vor unterschiedlichen Operationen den Kontext anzupassen; allerdings wird nirgendwo verwiesen, was der richtige Kontext ist. Die Fehlermeldungen sowie Debugmöglichkeiten sind leider häufig etwas spärlich und nur wenig hilfreich. Entsprechend spärlich ist auch die API-Dokumentation, ähnliche Einträge sind oft ohne Verlinkungen an verschiedenen Orten verteilt, wodurch sich das Einarbeiten in die Aufgaben als sehr zeitintensiv herausgestellt hat.

Einige unserer Ideen waren leider auch nicht so umsetzbar wie anfangs geplant, zum Beispiel die Umsetzung als Pinsel ist leider nur schwerlich möglich.

Nichtsdestotrotz konnten wir einen tieferen Einblick in die 3D-Modellierung gewinnen

Blender python ist bisschen dumm

Wir sind glücklich mit ergebnissen

Texture smart hinzufügen