Echtzeit-Visualisierung SS09

"Volume Shadows"

Ingo Radax, 0500848

Das Program

Das Program verwendet ein Partikelsystem um Rauch zu simulieren und schließlich auch zu rendern. Hauptaugenmerk lag dabei auf dem Rendern des Rauchs, die Simulation ist daher als einfaches Partikelsystem gehalten.
Als Vorlage für das Program dient das 'Smoke Particles' Beispiel aus dem Cuda SDK. Es wurde das Beispiel aus dem SDK neu implementiert (unter teilweiser Verwendung von Code aus dem Beispiel) und Opacity Shadow Maps erweitert. Außerdem wurde noch gewöhnliches Shadow Mapping verwendet um Schatten auf den Rauch werfen zu lassen.

Dateien

Der Source Code des Programs befindet sich im wesentlichen im Ordner **src/source/RTVis**. Hier befinden sich die Klassen für die Simulation und das Rendering des Partikelsystems. Die Dateien in den anderen Ordnern beinhalten nur alle möglichen Hilfsklassen (Matrizen, Vektoren, etc) und sind für das Program zweitrangig. Einzige Ausnahme ist die ApplicationManager Klasse (Ordner **src/source/Application**) in der das Partikelsystem initialisert, jeden Frame aktualisiert und schließlich auch gerendert wird.
Der Shader Code befindet sich in einer separeten Datei im Ordner **bin/data/shaders**.

Wiederverwendeter Source Code

Lediglich das **GpuArray** und der **Radix Sort** wurden eins zu eins aus dem Cuda SDK Beispiel übernommen. Die anderen Klassen wurden neu implementiert, wobei jedoch das Cuda SDK Beispiel als Vorlage gedient hat, weswegen es in einigen Bereicen starke Ähnlichkeiten gibt.

Steuerung

Man kann mittels **ASDW** und **gedrückter linken Maustaste** die Kamera bewegen bzw drehen. Zusätzlich gibt es noch ein User Interface auf der linken Seite mit dem alle möglichen Einstellungen vorgenommen werden können. Das User Interface ist grob in 3 Teile geteilt: Simulation, Rendering und Szene. 'Simulation' beinhaltet Einstellungen welche die Simulation des Rauchs betreffen und 'Rendering' entsprechend die Einstellungen für das Rendern. In 'Szene' finden sich schließlich noch allgemeine Einstellungen für die Szene, wie zum Beispiel die Licht Farbe.

Das seitliche User Interface wird ausschließlich mittels der Maus bedient. Fährt man mit der Maus über eines der Felder erscheinen + und - Buttons mit denen man die Werte verändern kann. Man kann die drei großen Kategorien auch wegklappen. Farben und Richtungen müssen erst ausgeklappt werden bevor die Werte verändert werden können (sie sind anfangs immer eingeklappt). Das seitliche User Interface kann auch mittels Drag&Drop verschoben bzw wie ein Fenster verkleinert werden.

UI - Smoke Simulation

Simulate	Simulation ein-/ausschalten
Time Scale	Simulation schneller oder langsamer durchführen

New Particles	Neue Partikel werden dem Rauch hinzugefügt. Falls ausgeschaltet verschwindet der Rauch nach kurzer Zeit.
EmitterType	Wechseln zwischen einem fixen Box Emitter und einem sich bewegenden Sphere Emitter. Durch die Emitter wird bestimmt wie neue Partikel hinzugefügt werden.
EmitterSize	Bestimmt die Größe des Emitters
Num. Particles	Bestimmt die Anzahl der Partikel. Intern sind 100.000 Partikel angelegt, man kann hiermit jedoch die Anzahl derer die verwendet werden reduzieren.
Gravity	Gravitation die auf die Partikel wirkt (immer nur entlang der y-Achse)
Damping	Dämpfung für die Geschwindigkeit der Partikel.
Noise Amplitude	Stärke des positionsabhängigen Rauschens welches auf die Partikel gelegt wird.
Noise Frequenz	Frequenz des positionsabhängigen Rauschens (intern wird eine statische 3D Noise Textur verwendet, die Frequenz bezieht sich auf den Zugriff in dieser Textur)
Noise Speed	Zusätzliches zeitabhängiges Rauschen .
P. Lifetime	Lebenszeit eines Partikels. Partikel werden im Laufe ihres Alters immer transparenter. Außerdem wird hierdurch indirekt bestimmt wie viele Partikel pro Sekunde neu hinzugefügt werden.
P. Init Velocity Dir.	Richtung der Anfangsgeschwindigkeit welche neue Partikel bekommen.
P. Init Velocity Magn.	Stärke der Anfangsgeschwindigkeit für neue Partikel.
P. Init Velocity Spread	Zusätliche zufällige (zufälliger Punkt auf einer Kugel) Geschwindigkeit für neue Partikel.

UI - Smoke Rendering

Render as	Legt fest wie die Partikel dargestellt werden (Siehe 'Rendering')
Display Smoke	Legt fest ob der Rauch angezeigt werden soll. Auch wenn die Anzeige ausgeschaltet wird wird der Rauch intern noch berechnet (sieht man zum Beispiel am Schattenwurf auf dem Boden).
Num Slices	Anzahl der Slices in welche die Partikel aufgeteilt werden.
Num Displayed Slices	Anzahl der Slices welche angezeigt werden.
Slice Mode	Art wie die Einteilung der Partikel in Slices vorgenommen wird (siehe 'Slicing')
Non-Uniform Factor	Parameter für das non-uniforme Slicing.
Sprite Size	Größe der Point Sprites.
Particle Alpha	Alpha für Partikel (verwendet wenn von der Kamera aus gesehen).
OSM Kappa	Parameter Kappa für Opacity Shadow Maps.
Smoke Shadow Alpha	Alpha für Partikel (verwendet wenn von der Lichtquelle aus gesehen).
Hard Shadow Alpha	Zusätliches Alpha für Partikel die im Schatten eines Objekt liegen.
Hard Shadow Factor	Bestimmt wie stark Farben abgedunkelt werden die im Schatten eines Objekts liegen.
Attenuation Color	Bestimmt wie das Licht abgeschwächt wird wenn es durch den Rauch geht. Bestimmt dadurch teilweise die Rauchfarbe.
Double Lightbuffer	Bestimmt ob zwei Lichtpuffer verwendet werden sollen (siehe 'Double

	Buffering')
Display Lightbuffer	Der Lichtpuffer wird angezeigt.
Blur Lightbuffer	Der Lichtpuffer wir pro Slice geblurt so dass Scattering simuliert wird.
Blur Strength	Stärke des Blurring.

UI - Scene

Occluder 1	Ein Schatten werfendes Objekt wird in der Szene angezeigt.
Occluder 2	Mehrre Schatten werfende Objekte werden in der Szene angezeigt.
Light Color	Farbe des Lichts.
Light Position	Position des Lichts.

Rendering

Es gibt insgesamt fünf verschiedene Möglichkeiten das Partikelsystem zu rendern, jedoch sind nur zwei davon gemacht um die Beleuchtung im Rauch darzustellen.

Mit der **Points** Darstellung werden einfach alle Partikel als einfache Punkte dargestellt. Hiermit kann man leichter sehen wie das Partikelsystem aufgebaut ist.

Mit der **Slices** Darstellung soll es leichter sein die einzelnen Slices zu erkennen. Die Partikel werden dabei als einfärbige Sprites gerendert. Als Farbe der Sprites wird der Interpolationsfaktor des Double Buffering verwendet. Wenn man den Noise ausschaltet kann man mit dieser Methode recht gut erkennen wie die Slices erzeugt werden.

Bei der **Point Sprites** Darstellung werden die Partikel als kreisförmige Pointssprites dargestellt, die vom Kreiszentrum zum Rand hin immer transparenter werden. Diese Darstellung ist die Grundlage für die zwei Beleuchtungs Darstellungen

Bei der **Nvidia** Darstellung handelt es sich um die Nachimplementierung des Beispiels aus dem Cuda SDK. Hier wird die Beleuchtung und Selbstabschattung des Rauchs gerendert.

Die **OSM** Darstellung entspricht im wesentlichen der Nvidia Darstellung, doch es werden hier Opacity Shadow Maps verwendet um die Beleuchtung zu berechnen.

Slicing

Es wurden mehrer Varianten implementiert wie die Slices berechnet werden. Mittels der Slices Darstellung kann man sehen wo die Unterschiede liegen.

Bei **Uniform Particles** werden die Slices so erzeugt dass jedes Slices die selbe Anzahl an Partikeln enthält (Gesamtanzahl der Partikel / Anzahl der Slices).

Bei **Uniform Space** werden die Slices so erzeugt dass die Dicke aller Slices gleich bleibt.

Non-Uniform Particles und **Non-Uniform Space** entsprechen den uniformen Varianten, jedoch werden die Slices anhand einer Exponentialfunktion erstellt. Die ersten Slices haben weniger Partikel/sind weniger dick, die letzten Slices haben mehr Partikel/sind dicker. Diese Methode hat den Sinn dass die ersten Partikel wesentlich mehr Einfluss auf die Lichtfarbe haben und daher hier mehr Slices sein sollten.

Double Buffering

Bem Beispiel im Cuda SDK wurde lediglich ein Lichtpuffer verwendet um die Beleuchtung eines

Partikels zu bestimmen. Bei den Opacity Shadow Maps hingegen wurde davon ausgegangen dass man zwei Lichtpuffer (welche das jeweilige Slice begrenzen) verwendet um die Beleuchtung interpolieren zu können.

Im Programm sind beide Varianten implementiert, man kann auch bei der Nvidia Methode zwei Puffer verwendet bzw bei der OSM Methode nur einen. Wobei in beiden Fällen jedoch zwei Puffer ein wesentlich besseres Ergebnis liefern (sogar wenn man die Anzahl der Slices reduziert).

Shadow Mapping

Um auch Schatten auf den Rauch werfen zu können wurde Shadow Mapping verwendet. Dabei wird einfachheitshalber der Tiefenpuffer verwendet welcher beim SmokeRenderer verwendet wird. Dieser ist zwar relativ niedrig aufgelöst, doch fällt das beim Rauch selbst nicht auf (nur auf dem Boden).

Die Beleuchtung des Rauchs (von der Lichtquelle aus gesehen) wird immer ohne Tiefenpuffer berechnet, also so als wenn nichts einen Schatten werfen würde, erst wenn der Rauch von der Kamera aus gerendert wird wird auch auf den Tiefenpuffer zugegriffen. Das hat den Sinn dass der Rauch im Schatten selbst auch eine Farbe haben sollte. Wenn man den Rauch (von er Lichtquelle aus) mit Tiefenpuffer rendert würde der Rauch im Schatten oft nur einfärbig aussehen (so wie bei der Point Sprites Darstellung).

Für den Lookup der Shadow Map wird außerdem immer der Mittelpunkt der Partikel verwendet und nicht die exakten Positionen. Würde man die exakten Positionen verwenden so sieht man an den Übergängen von Schattiert zu Unschattiert scharfe Kanten (ähnlich wie wenn ein Partikel den Boden schneidet und man das Sprite erkennt). Durch die Verwendung des Sprite Mittelpunkes jedoch entstehen keine solche sichtbaren Kanten und der Schatten sieht dennnoch richtig und gut aus.