

A GPU-Based Particle Engine

Seminar Computergrafik: »Procedural Content Generation«

Sebastian Rockel

Department für Informatik
Universität Hamburg

23. Juni 2010

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② ÜberFlow-Partikelsystem

Überblick

Anwendungen

Alles mit Texturen

③ Zusammenfassung

Zusammenfassung und Demo

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② ÜberFlow-Partikelsystem

Überblick

Anwendungen

Alles mit Texturen

③ Zusammenfassung

Zusammenfassung und Demo

Einführung I

Partikelsystem

Wikipedia

»Ein Partikelsystem ist eine *Funktion* von *Animationsprogrammen*, mit der sich auf einfache Weise eine große Anzahl von Objekten animieren lässt.«

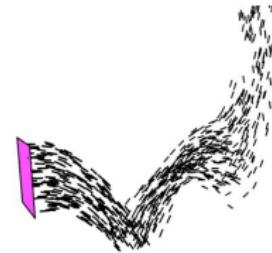
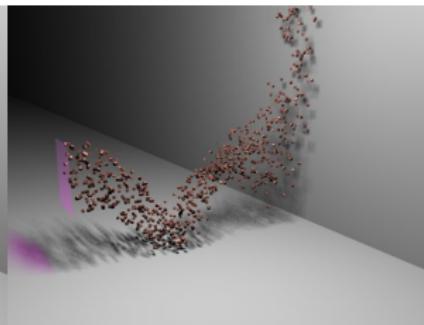
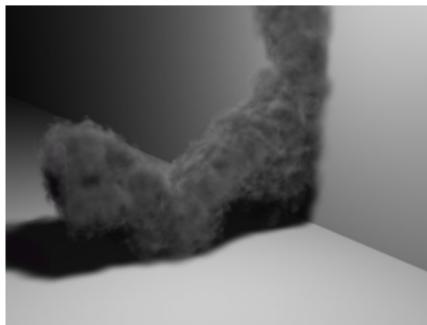


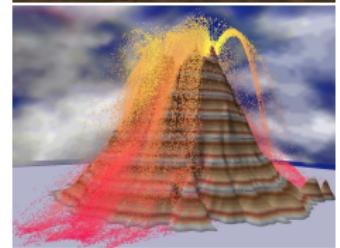
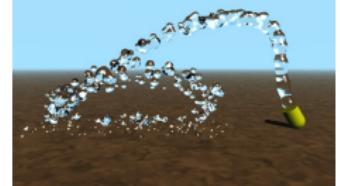
Abbildung: Wikipedia

Einführung II

Partikelsystem

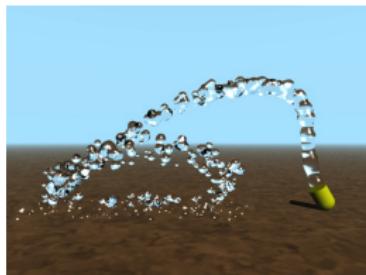
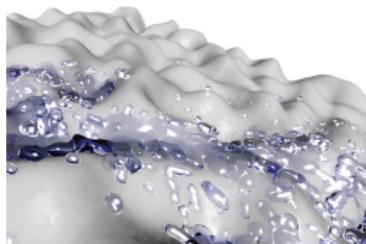
Wofür?

- Volumetrische Effekte
 - Feuer, Explosionen, Rauch, Flüssigkeiten
 - basierend auf einfachen physikalischen Modellen
- Dynamik-Simulationen
 - Partikel-Partikel-Interaktion
 - komplexere physikalische Modelle



Demo

POV-Ray [Joh02]



Fluss

Feuer

Fontäne

Rauch

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② ÜberFlow-Partikelsystem

Überblick

Anwendungen

Alles mit Texturen

③ Zusammenfassung

Zusammenfassung und Demo

Partikelparameter

- Emission
 - Geschwindigkeit
 - Lebensdauer
 - Dämpfung
 - Anzahl aller Partikel

Partikelparameter

- Emission
 - Geschwindigkeit
 - Lebensdauer
 - Dämpfung
 - Anzahl aller Partikel
- Bewegung

Partikelparameter

- Emission
 - Geschwindigkeit
 - Lebensdauer
 - Dämpfung
 - Anzahl aller Partikel
- Bewegung
- Kollision
 - Partikel
 - Umgebung

Partikelparameter

- Emission
 - Geschwindigkeit
 - Lebensdauer
 - Dämpfung
 - Anzahl aller Partikel
- Bewegung
- Kollision
 - Partikel
 - Umgebung
- Kollisionsreaktion

Partikelparameter

- Emission
 - Geschwindigkeit
 - Lebensdauer
 - Dämpfung
 - Anzahl aller Partikel
- Bewegung
- Kollision
 - Partikel
 - Umgebung
- Kollisionsreaktion
- Externe Kräfte
 - Gravitation
 - Wind

Zwischenstand

Partikelsystem

- bisher: keine,
 - Echtzeit
 - Partikel-zu-Partikel-Interaktion

Zwischenstand

Partikelsystem

- bisher: keine,
 - Echtzeit
 - Partikel-zu-Partikel-Interaktion

Anforderungen an »unser« Partikelsystem

- es soll echtzeitfähig sein, sowie Partikel untereinander reagieren lassen
- sehr viele Partikel behandeln können
 - GPU-Berechnung

GPU-Realisierungen

Populäre

- *CUDA (NVIDIA)*
- *OpenGL*
- OpenCL
- DirectX (Microsoft)

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② ÜberFlow-Partikelsystem

Überblick

Anwendungen

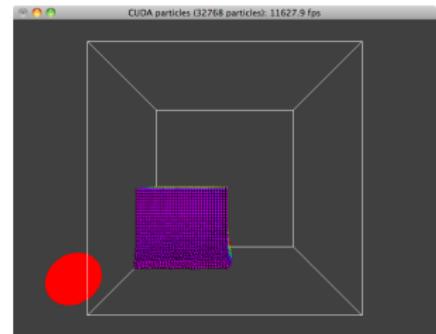
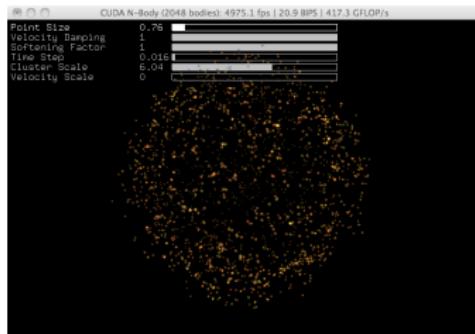
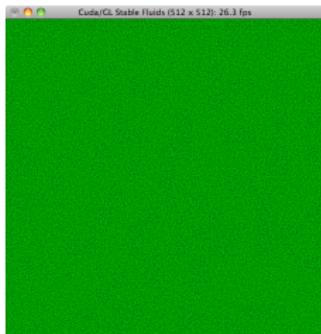
Alles mit Texturen

③ Zusammenfassung

Zusammenfassung und Demo

Demo

CUDA [NVI10]



- 2D-Wassersimulation: fluidsGL
- 3D-Galaxysimulation: nbody
- 3D-Partikelsimulation: particles

Zwischenstand

Partikelsystem

- echtzeitfähig
- > 30.000 Partikel
- GPU-Beschleunigung

Zwischenstand

Partikelsystem

- echtzeitfähig
- > 30.000 Partikel
- GPU-Beschleunigung

Flaschenhals

CPU-zu-GPU-Datentransfer

Zwischenstand

Partikelsystem

- echtzeitfähig
- > 30.000 Partikel
- GPU-Beschleunigung

Flaschenhals

CPU-zu-GPU-Datentransfer

Anforderung

Berechnung und Rendering auf GPU

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② UberFlow-Partikelsystem

Überblick

Anwendungen

Alles mit Texturen

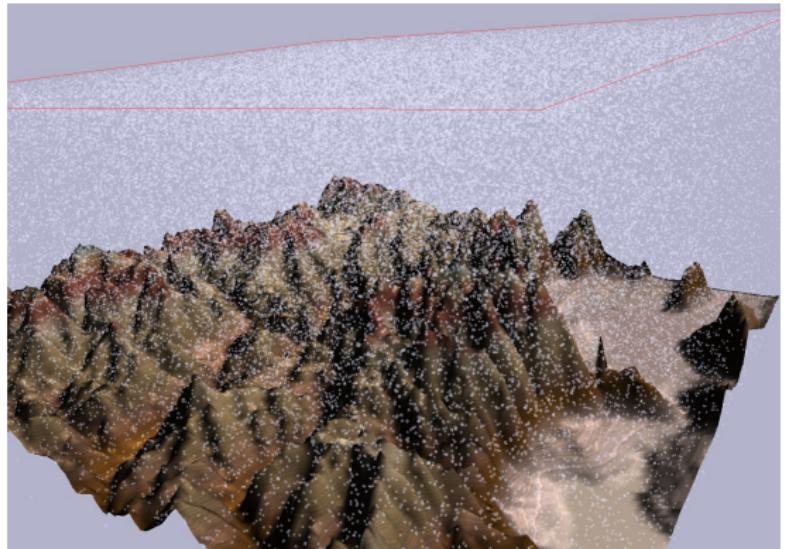
③ Zusammenfassung

Zusammenfassung und Demo

UberFlow I

Partikelsystem [KSW04]

- Echtzeit-Darstellung
- > 100.000 Partikel
- GPU-Berechnung



UberFlow II

Partikelsystem [KSW04]

- OpenGL
 - Pixelshader 3.0
- Positions- und Rendering-Berechnung auf GPU
 - Vermeidung CPU-GPU-Flaschenhals
- 3D Position als RGBA-Textur
- Geschwindigkeit als Textur
- Kollisionserkennung
- Heightmaps

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② UberFlow-Partikelsystem

Überblick

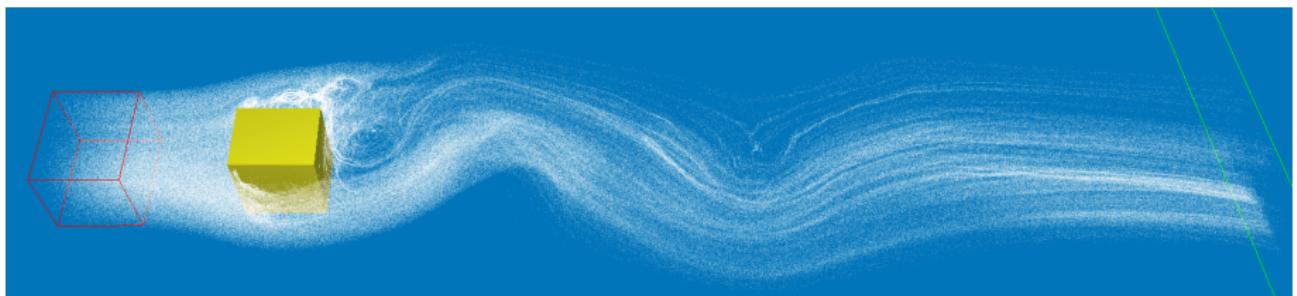
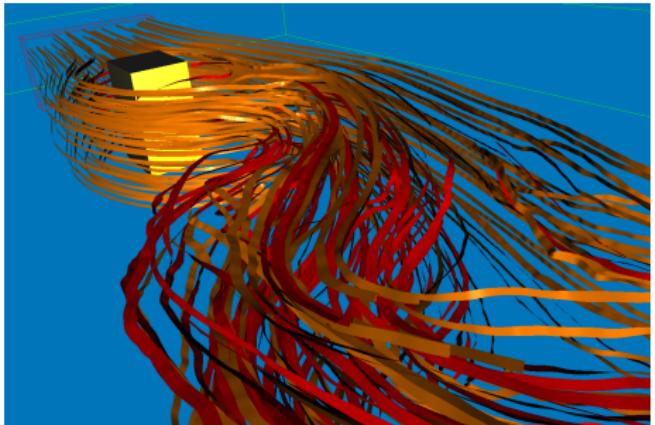
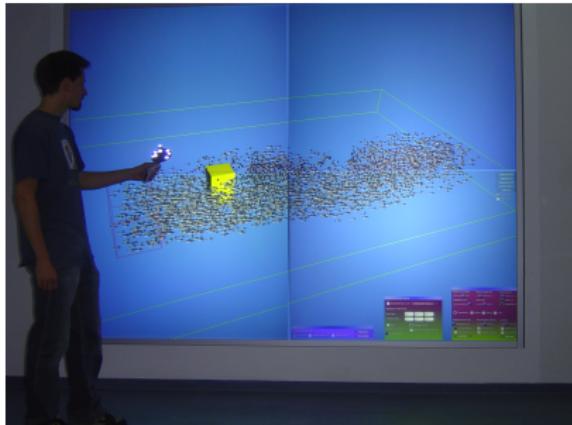
Anwendungen

Alles mit Texturen

③ Zusammenfassung

Zusammenfassung und Demo

Anwendung



[KSW04]

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② UberFlow-Partikelsystem

Überblick

Anwendungen

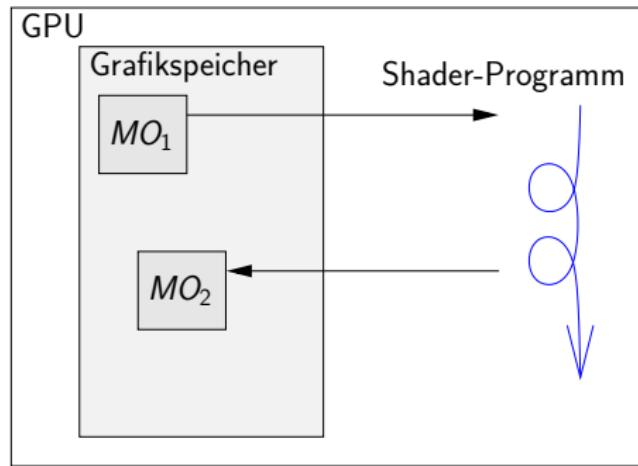
Alles mit Texturen

③ Zusammenfassung

Zusammenfassung und Demo

OpenGL Memory Objects

- Daten und Rendertargets im Grafikspeicher



Emission

UberFlow

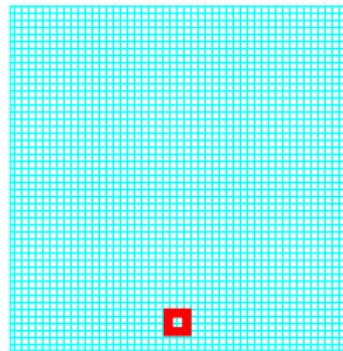
- Partikelposition und Zeitstempel

Emission

UberFlow

- Partikelposition und Zeitstempel

RGBA-Textur ($n \times n$)

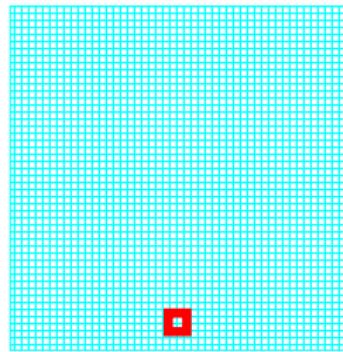


Emission

UberFlow

- Partikelposition und Zeitstempel

RGBA-Textur ($n \times n$)



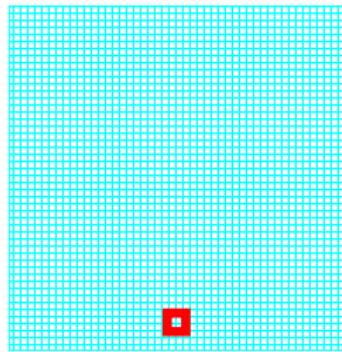
$$\overbrace{\begin{pmatrix} R \\ G \\ B \\ A \end{pmatrix}^{n \times n}} = r(t)^{n \times n} = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix}^{n \times n}$$

Emission

UberFlow

- Partikelposition und Zeitstempel

RGBA-Textur ($n \times n$)



$$\overbrace{\begin{pmatrix} R \\ G \\ B \\ A \end{pmatrix}^{n \times n}} = r(t)^{n \times n} = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix}^{n \times n}$$

- ähnlich für Geschwindigkeit und Windfeld

Sortierung

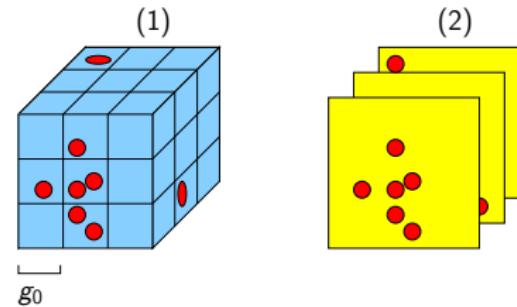
UberFlow

- ① nach Abstand zwischen Partikeln (Kollision)
- ② nach Abstand zum Betrachter (Transparenz)

Sortierung

UberFlow

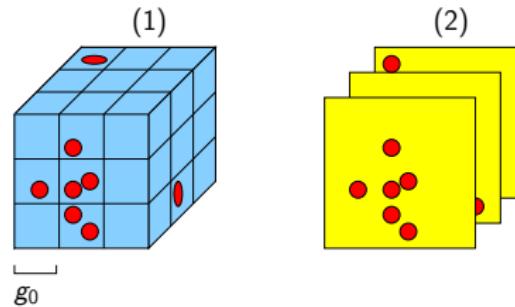
- ① nach Abstand zwischen Partikeln (Kollision)
- ② nach Abstand zum Betrachter (Transparenz)



Sortierung

UberFlow

- ① nach Abstand zwischen Partikeln (Kollision)
- ② nach Abstand zum Betrachter (Transparenz)

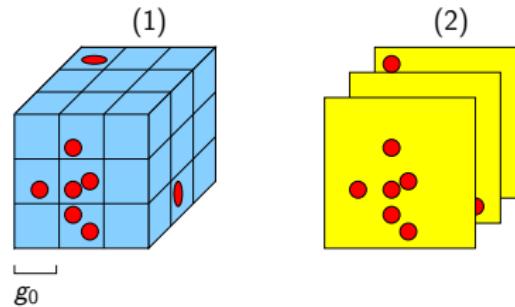


- Sortierschlüssel:
 - ① $x/g_0^2 + y/g_0 + z$
 - ② Distanz zum Betrachter

Sortierung

UberFlow

- ① nach Abstand zwischen Partikeln (Kollision)
- ② nach Abstand zum Betrachter (Transparenz)



- Sortierschlüssel:
 - ① $x/g_0^2 + y/g_0 + z$
 - ② Distanz zum Betrachter
- Speicherung in 2D-(RG)-Textur (Index + Schlüssel)
- Sortierung (optimiert) und Anpassung der Partikel-Position und Geschwindigkeit (Texturen)

Kollisionserkennung

UberFlow

- suche den nächsten Partikel in der (sortierten) 2D-Textur

Kollisionserkennung

ÜberFlow

- suche den nächsten Partikel in der (sortierten) 2D-Textur
- 2. Sortierung mit anderem Schlüssel

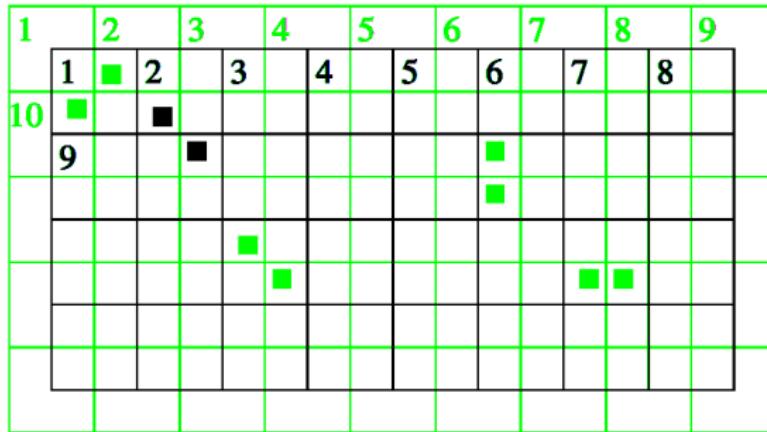


Abbildung: [KSW04]

Gliederung

① Einführung

Partikelsystem

Details

GPU-Partikelsysteme

② ÜberFlow-Partikelsystem

Überblick

Anwendungen

Alles mit Texturen

③ Zusammenfassung

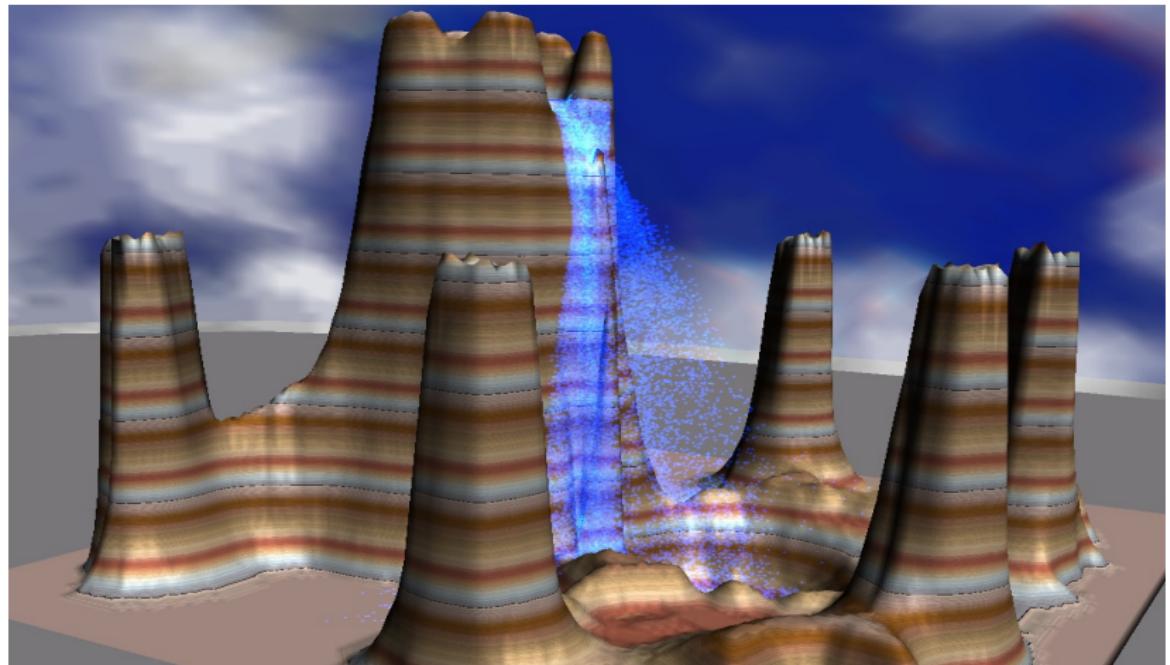
Zusammenfassung und Demo

Zusammenfassung

- Vermeidung des Bus-Flaschenhals
 - »erstes« Partikelsystem komplett auf GPU
- Inter-Partikel-Kollisionserkennung
- Sichtbarkeitssortierung
- echtzeitfähig
- GPU-Sortier-Algorithmen
- andere Physikmodelle implementierbar

Demo

UberFlow



Uberflow Demo

Weiterführende Literatur I



Rune Skovbo Johansen.

rune|vision - rune's particle system (pov-ray) [online].
2002.

Available from: <http://runevision.com/3d/include/particles/>
[cited Donnerstag, 10. Juni 2010].



Peter Kipfer, Mark Segal, and Rüdiger Westermann.

Overflow: a gpu-based particle engine.

In *HWWS '04: Proceedings of the ACM*

SIGGRAPH/EUROGRAPHICS conference on Graphics hardware,
pages 115–122. ACM Press, New York, NY, USA, 2004.

doi:10.1145/1058129.1058146.

Weiterführende Literatur II



NVIDIA.

Cuda release archive [online].

2010.

Available from:

http://developer.nvidia.com/object/cuda_archive.html [cited Donnerstag, 10. Juni 2010].