

Gama Network Presents:

Gamasutra.com

Building a Million-Particle System

()

By Lutz Latta

Gamasutra

July 28, 2004

URL: http://www.gamasutra.com/features/20040728/latta_01.shtml

1

(GPU)

GPU

1

가

GPU

(height field)

가?

(Particle systems, PS)

가

1960

2D

//

([Reeves1983]).

([Sims1990]).

([McAllister2000]).

CPU

가 가

[Burg2000].

(Fill rate) CPU GPU
가 GPU 가 가
, ,
,
,
CPU GPU
가
CPU
1
가 GPU
GPU
가 가
,
(
가 가
가
1 PC GPU
" ([NVIDIA2001]),
"
")
")

가 ").

가 GPU

([[Harris2003](#)]).

([[Green2003](#)]).

“ ”

([[Purcell2003](#)]).

가 :

GPU

)

([[NVIDIA2001](#)]).

가

가

가

가

가 (“

”).

가

가

가

“ ”

[[GPGPU2003](#)]

가

“

”

([[Buck2003](#)]):

가

가

(

).

GPU

가

(“

”)

가

가

가

- ,

GPU
가

PC

PC

“
가

”

Xbox

8

가 ,

가 . PC

2

가 (“ ”).

가

가 .

16

가 (“ ”) .

가

3

가

,

.

(, , , ,)

,

가

.

(“ ”)

,

.

:

.

.

.

.

가

가

:

,

,

.

,
.
,
.

가 :

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.

()

1.

.
가
가 .
, , .

가

GPU

가

가

CPU

가

가

가
가
가
CPU , ([McAllister2000])
).

CPU GPU . CPU
가 . GPU
가 :
가 ,
가 가 ,
가 가
.

2.

가 .
“
”
가 .
가 .
가 .

가
 ([Sims1990] [McAllister2000]): (,),
 (,), , . GPU

.

가 .

가 .

가 ([McAllister2000]) . 가
 가

GPU ,
 2D 3D ,
 . v_{fl} F_d
 :

$$F_d = \frac{6\pi\eta r}{c}(\bar{v} - v_{fl})$$

\bar{v} n , r (),
 c .

. 가

:

$$a = \frac{F}{m}$$

a 가 , F , m , 가 .

가

:

$$v = \bar{v} + a \cdot \Delta t$$

v , \bar{v} , Δt .
 , 1 , 가 0 가 .

GPU
가 . GPU

([Sims1990])

n

:

$$v_n = (v_{bc} \cdot n) v_{bc}$$

$$v_t = v_{bc} - v_n$$

v_{bc}

v_n

v_t

μ

ϵ

$$v = (1 - \mu) v_t - \epsilon v_n$$

가

가 0

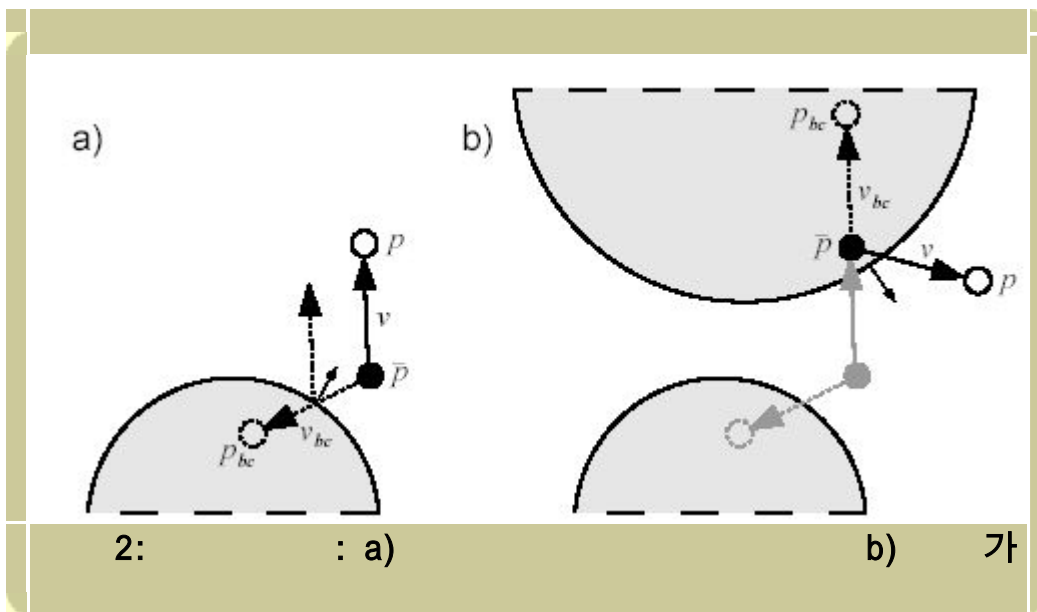
0

가

가

가

가



2: : a) b) 가

가 . 가

,
가 3
가 .

가 가 ,
(2 a) .

p_{bc} :

$$p_{bc} = \bar{p} + v_{bc} \cdot \Delta t$$

\bar{p}
가 가

(2b) .

가

:

$$v = \begin{cases} v_{bc} & | \ v_{bc} \cdot n \geq 0 \\ v_t - v_n & | \ v_{bc} \cdot n < 0 \end{cases}$$

3.

(“ ”). GPU

가

$$p = \bar{p} + v \cdot \Delta t$$

p

\bar{p}

([\[Verlet1967\]](#))

).

([\[Jakobsen2001\]](#)).

가

:

$$v = \bar{v} + a\Delta t \wedge p = \bar{p} + v\Delta t \Rightarrow v = \frac{p - \bar{p}}{\Delta t} \text{ and } \bar{v} = \frac{\bar{p} - \bar{\bar{p}}}{\Delta t}$$

$$\Rightarrow p = \bar{p} + (\bar{v} + a\Delta t)\Delta t = \bar{p} + \left(\frac{\bar{p} - \bar{\bar{p}}}{\Delta t} + a\Delta t\right)\Delta t$$

$$\Rightarrow p = 2\bar{p} - \bar{\bar{p}} + a\Delta t^2$$

(\bar{p}) 가

가 .

가 .

([\[Jakobsen2001\]](#)).

4.

가 .

GPU

“ ”

([\[Batcher1968\]](#)).

가

가 가 가 가 .
 가 가 가 20 - 50 가
 가 가 가 .

3: 8
 Y- : . X- : .

[Lang2003])

0 1 가 가 가 가
 3 8
 가 가 가

```

float4 mergeSort1DEnd(float _Current : TEXCOORD0,
    uniform int _Step) : COLOR
{
    float currentSample = (float)texRECT(_SortData,
        (float2)_Current);
    float direction = (fmod(_Current / _Step, 2.0) < 1.0 ? 1.0 :
        -1.0);
    float otherSample = (float)texRECT(_SortData,
        (float2)(_Current + direction * _Step));
    if (direction >= 0)
        return max(currentSample, otherSample);
    else
        return min(currentSample, otherSample);
}

float4 mergeSort1DRecursion(float _Current : TEXCOORD0,
    uniform int _Step, uniform int _Count) : COLOR
{
    float currentSample = (float)texRECT(_SortData, (float2)_Current);
    int modulus = fmod(_Current / _Step, (float)_Count);
    if (modulus >= 1 && modulus < _Count - 1)
    {
        if (fmod((float)modulus, 2.0) > 1.0)
            return max(currentSample,
                (float)texRECT(_SortData, (float2)(_Current + _Step)));
        else
            return min(currentSample,
                (float)texRECT(_SortData, (float2)(_Current - _Step)));
    }
    else
        return currentSample;
}

```

4: 1

Cg

4 Cg ([Mark2003]).

texRECT tex2D

HLSL ([Microsoft2002]).

1 .

2

u v . “ ”

“ ” 가 . 5 CPU

```
MergeSort(int _Count) :
    if (_Count > 1)
        MergeSort(_Count / 2)
MergeSort(int _Count) :
    if (_Count > 1)
        MergeSort(_Count / 2)
        Merge(_Count, 1)
Merge(int _Count, int _Step) :
    if (_Count > 2)
        Merge(_Count / 2, _Step * 2)
        Render with mergeSortRecursion shader
    else
        Render with mergeSortEnd shader
```

5:

$$\frac{1}{2} \log_2^2 n + \frac{1}{2} \log_2 n$$

. 1024x1024

210

가 .

가 .

210

가 .

50 ,

1-2

4

:

()

5.

PC GPU
가

:

1. DirectX OpenGL (VS) 3.0([Microsoft2002]
) ARB_vertex_shader ([OpenGL2003])
가

2. “ - ” (; [Percy2003]
) ,

GPU 가 OpenGL API
OpenGL NV_pixel_data_range
([NVIDIA2002]) , GPU
가

().

가

“ ”

:

VS3.0

가 , 가

가 ;

6.

$\frac{1}{3}$ $\frac{1}{4}$ z

2

2

가

(,)

(“ ”) .

-

:

(2)

GPU

2

가

[0..1]²

가

2

2x2

3x2

1

GPU

1024x1024

GPU

512x512

PC

GPU

([Imonen2003]).

2

)

(

가 가

CPU

가 가

가

.

가

.

NVIDIA

Batcher1968: Batcher, Kenneth E.; Sorting Networks and their Applications. In *Spring Joint Computer Conference, AFIPS Proceedings 1968*

Buck2003: Buck, Ian; Data Parallel Computing on Graphics Hardware, 2003,
<http://graphics.stanford.edu/~ianbuck/GH03-Brook.ppt>

Burg2000: van der Burg, John; Building an Advanced Particle System, *Game Developer Magazine*, 03/2000

GPGPU2003: Harris, Mark et al.; GPGPU Website, 2003-2004, <http://www.gpgpu.org/>

Green2003: Green, Simon; Stupid OpenGL Shader Tricks, 2003,
http://developer.nvidia.com/docs/IO/8230/GDC2003_OpenGLShaderTricks.pdf

Harris2003: Harris, Mark, Real-Time Cloud Simulation and Rendering, Department of Computer Science, University of North Carolina at Chapel Hill, 2003

Ilmonen2003: Ilmonen, Tommi; Kontkanen, Janne; The Second Order Particle System. In *WSCG Proceedings 2003*

Jakobsen2001: Jakobsen, Thomas; Advanced Character Physics. In *GDC Proceedings 2001*

Lang2003: Lang, Hans W.; Odd-Even Merge Sort, 2003,
<http://www.iti.fhflensburg.de/lang/algorithmen/sortieren/oemen.htm>

Mark2003: Mark, William R.; Glanville, R. Steven; Akeley, Kurt; Kilgard, Mark J.; Cg: A System for Programming Graphics Hardware in a C-like Language. In *SIGGRAPH Proceedings 2003*

McAllister2000: McAllister, David K.; The Design of an API for Particle Systems, Technical Report, Department of Computer Science, University of North Carolina at Chapel Hill, 2000

Microsoft2002: Microsoft Corporation; DirectX9 SDK, 2002,
<http://msdn.microsoft.com/directx/>

NVIDIA2001: NVIDIA Corporation; NVIDIA SDK, 2001-2003, <http://developer.nvidia.com/>

NVIDIA2002: NVIDIA Corporation; OpenGL Extension NV_pixel_data_range, 2002,
http://oss.sgi.com/projects/ogl-sample/registry/NV/pixel_data_range.txt

OpenGL2003: OpenGL ARB; OpenGL Extension ARB_vertex_shader, 2003,
http://oss.sgi.com/projects/ogl-sample/registry/ARB/vertex_shader.txt

Percy2003: Percy, James; OpenGL Extensions, 2003,

http://www.ati.com/developer/SIGGRAPH03/Percy_OpenGL_Extensions_SIG03.pdf

Purcell2003: Purcell, Timothy J.; Donner, Craig; Cammarano, Mike; Jensen, Henrik W.; Hanrahan, Pat; Photon Mapping on Programmable Graphics Hardware. In *Graphics Hardware Proceedings 2003*

Reeves1983: Reeves, William T.; Particle Systems – Technique for Modeling a Class of Fuzzy Objects. In *SIGGRAPH Proceedings 1983*

Sims1990: Sims, Karl; Particle Animation and Rendering Using Data Parallel Computation. In *SIGGRAPH Proceedings 1990*

Verlet1967: Verlet, Loup; Computer Experiments on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules, *Physical Review*, 159/1967

Copyright 2003 CMP Media Inc. All rights reserved.