goto; conference aarhus

MASSIVE ACCELERATION THROUGH THE MANY-CORE PROCESSOR THAT YOU CALL A GRAPHICS CARD

Jesper Mosegaard

Head of Computer Graphics Lab
Alexandra Institute

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

gotocon.com

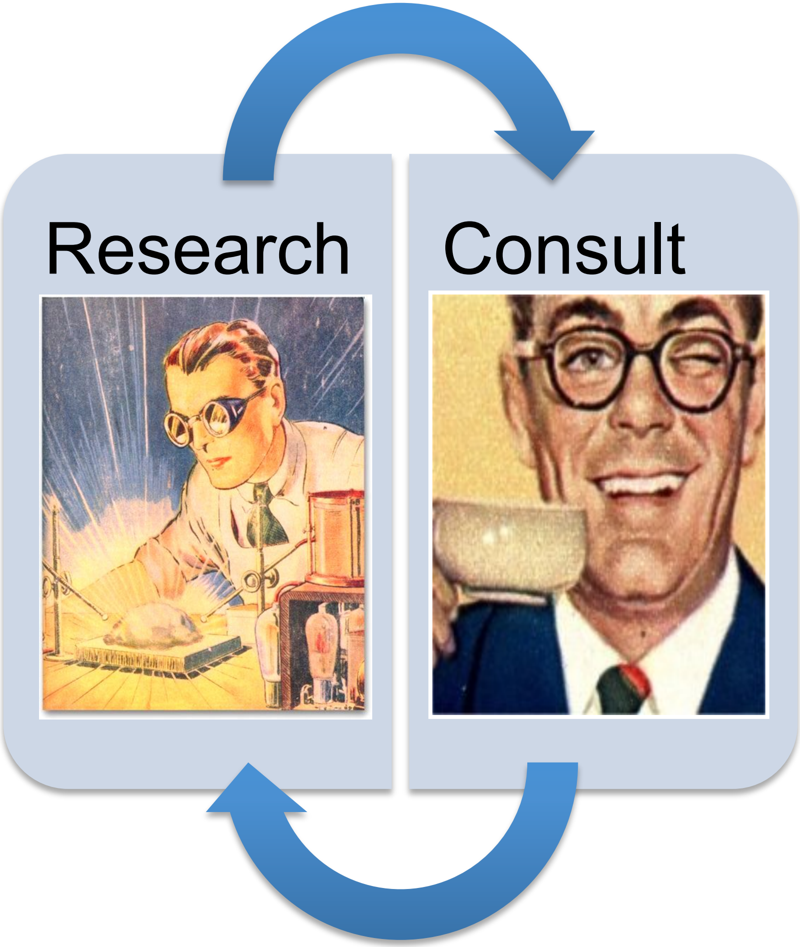
Plan

- Historical review
- Cases
- Future - and when is it for you ?

GTS - Advanced Technology Group

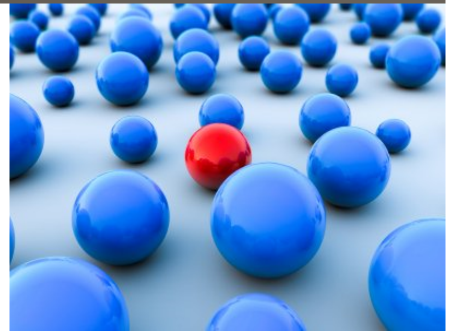
- The Alexandra Institute is one of Denmark's nine GTS Institutes
  - Approved by the Danish Ministry of Science, Technology and Innovation
  - Independent and not-for-profit companies
  - The core of technological infrastructure in Denmark
  - Develop technological services based on latest research
  - Sell state-of-the-art technological services to private enterprises and public authorities

Research based user driven innovation



ALEXANDRA
INSTITUTTET

What do we do ?



- Cutting-edge knowledge and competencies
- Research strategy, and active in research

- Software development
- Teaching and training
- Partner in research projects
- Independent partner in choice of technology, method etc.
- Idea-generating

ALEXANDRA INSTITUTTET

Computer Graphics Lab

Jesper Mosegaard, head of research
Ph.d. Computer Science

Peter Trier Mikkelsen
Masters Computer Science

Karsten Noe
Ph.d. Computer Science

Jens Rimestad
Masters Computer Science

Brian Christensen
Ph.d. Computer Science

Jesper Børlum
Masters in Civil Engineering

Thomas Kim Kjeldsen
Ph.d. In Physics

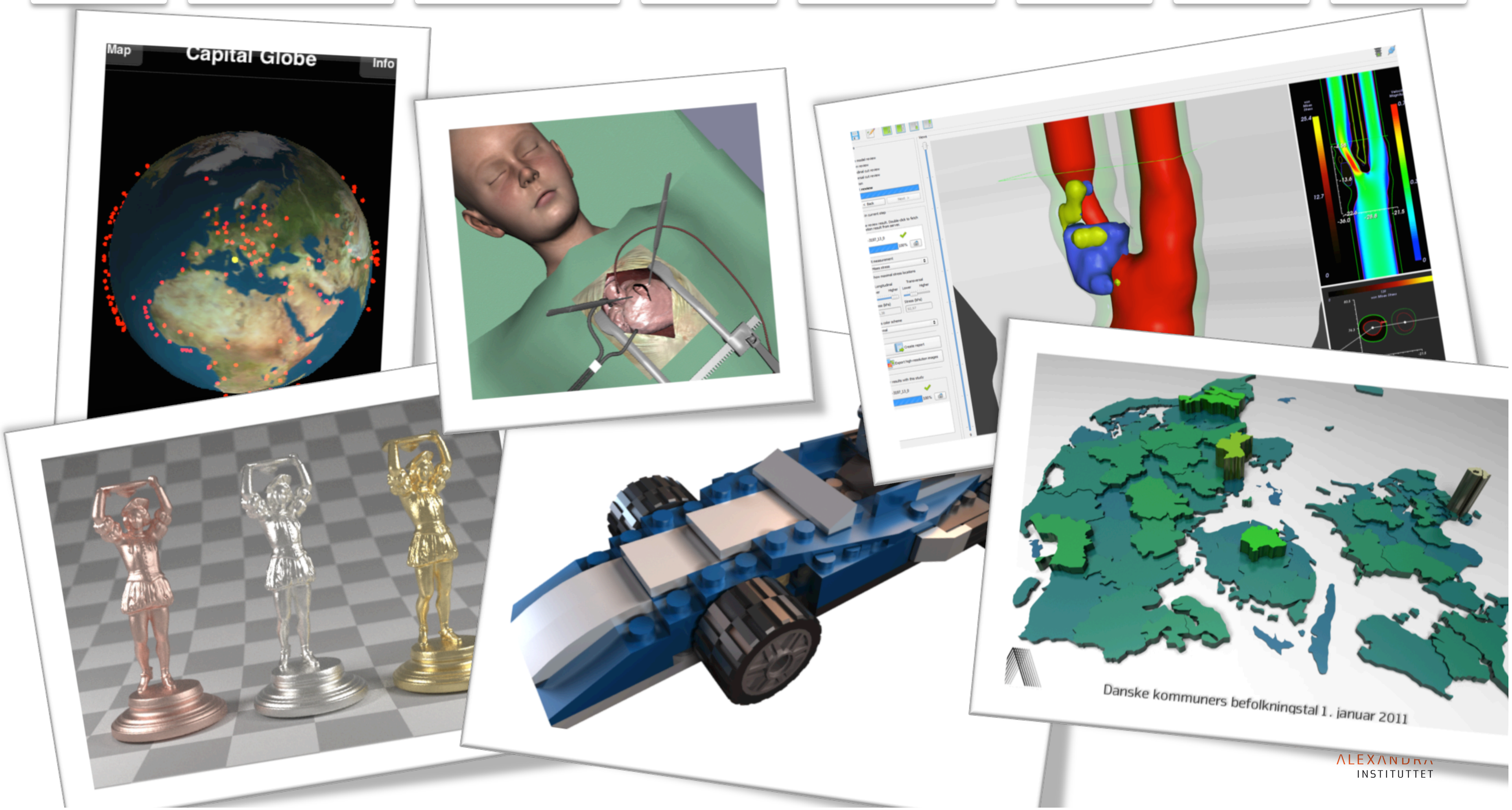
Lee Lassen
Masters in Computer Science

Nikolaj Andersen
3D graphics Artist

ALEXANDRA
INSTITUTTET

An overview

3D | Photorealistic | Visualization | Materials | Fast calculation | GPGPU | Big Data | Medical



ALEXANDRA INSTITUTTET

Computer Graphics in many areas

CG cooperation

molegro bioinformatics solutions

Quantum Wise

LUXION

Danfoss

brainreader ApS

DIGITAL URBAN LIVING

DUL

Apex

LEGO

BANG & OLUFSEN

B&O

Animation Hub

unity studios

MORPHOLOGICA
virtual surgery

rediaals

intertisement

DI ITEK

3XN

CAVI

novo nordisk®

DTU Informatik
Institut for Informatik og Matematisk Modellering

3D SUPPLY

UCS
Utopian City Scape

Historical Review

ALEXANDRA INSTITUTTET

Software rasterization

- Creative freedom

Outcast, 1999

Comanche, 1992

Fixed Function pipeline

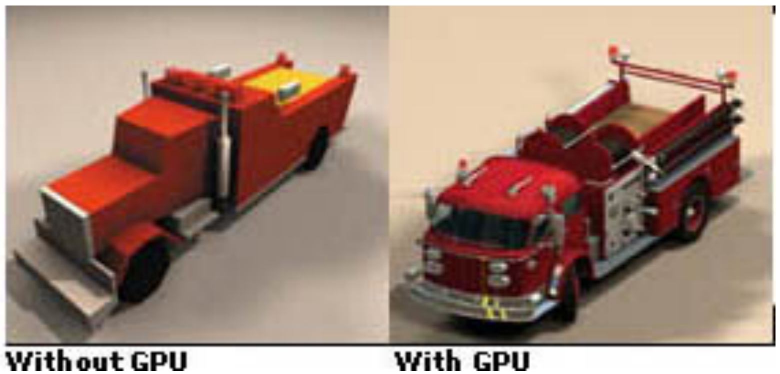
Ridge Racer

Battlefield 1942

Quake 2

The GPU

- GeForce 256 ”The worlds first GPU” (1999)
  - Integrated T&L
  - Texture/Environment Mapping




Viewer
Drawstyle
nVIDIA

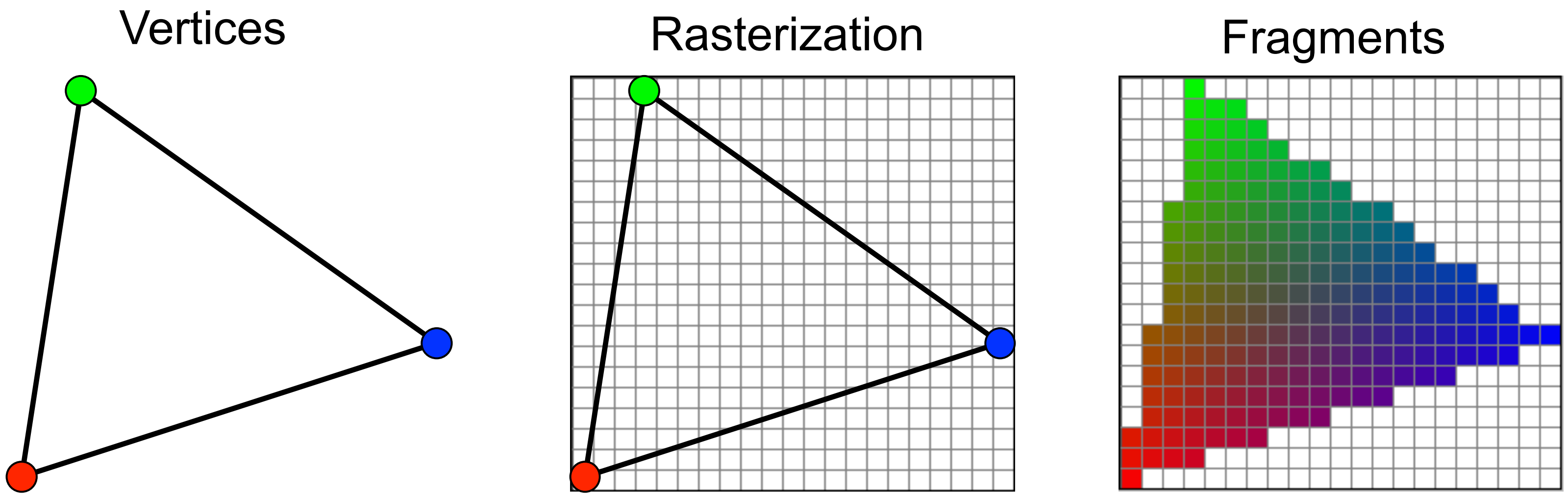

Viewer
Drawstyle
nVIDIA

First programmable cards

- NV_Vertex_program (Geforce3) - 2000
- NV_Fragment_program (GeForce FX) - 2001

- In 2002
  - ARB_Fragment_program
  - ARB_Vertex_program

Programmable vertices and fragments



ARB Vertex program 1.0

```
!!ARBvp1.0
TEMP R0, R1;
DP3   R0, program.local[32], vertex.normal;
MUL   result.color.primary.xyz, R0, program.local[35];
MAX   R0, program.local[64].x, R0;
MUL   R0, R0, vertex.normal;
MUL   R0, R0, program.local[64].z;
ADD   R1, vertex.position, -R0;
DP4   result.position.x, state.matrix.mvp.row[3], R1;
DP4   result.position.y, state.matrix.mvp.row[1], R1;
DP4   result.position.z, state.matrix.mvp.row[2], R1;
DP4   result.position.w, state.matrix.mvp.row[3], R1;
```

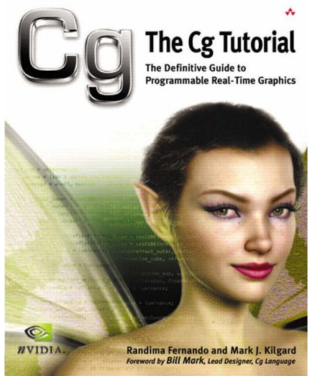
ALEXANDRA INSTITUTTET

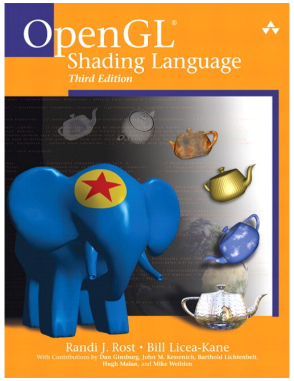
nVidia Dawn demo

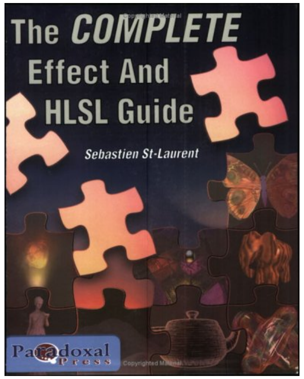
- GeForce FX, 2002

High level shader languages

- nVidia Cg, 2002
- Microsoft HLSL, 2002
- OpenGL GLSL, 2004







ALEXANDRA INSTITUTTET

GLSL example

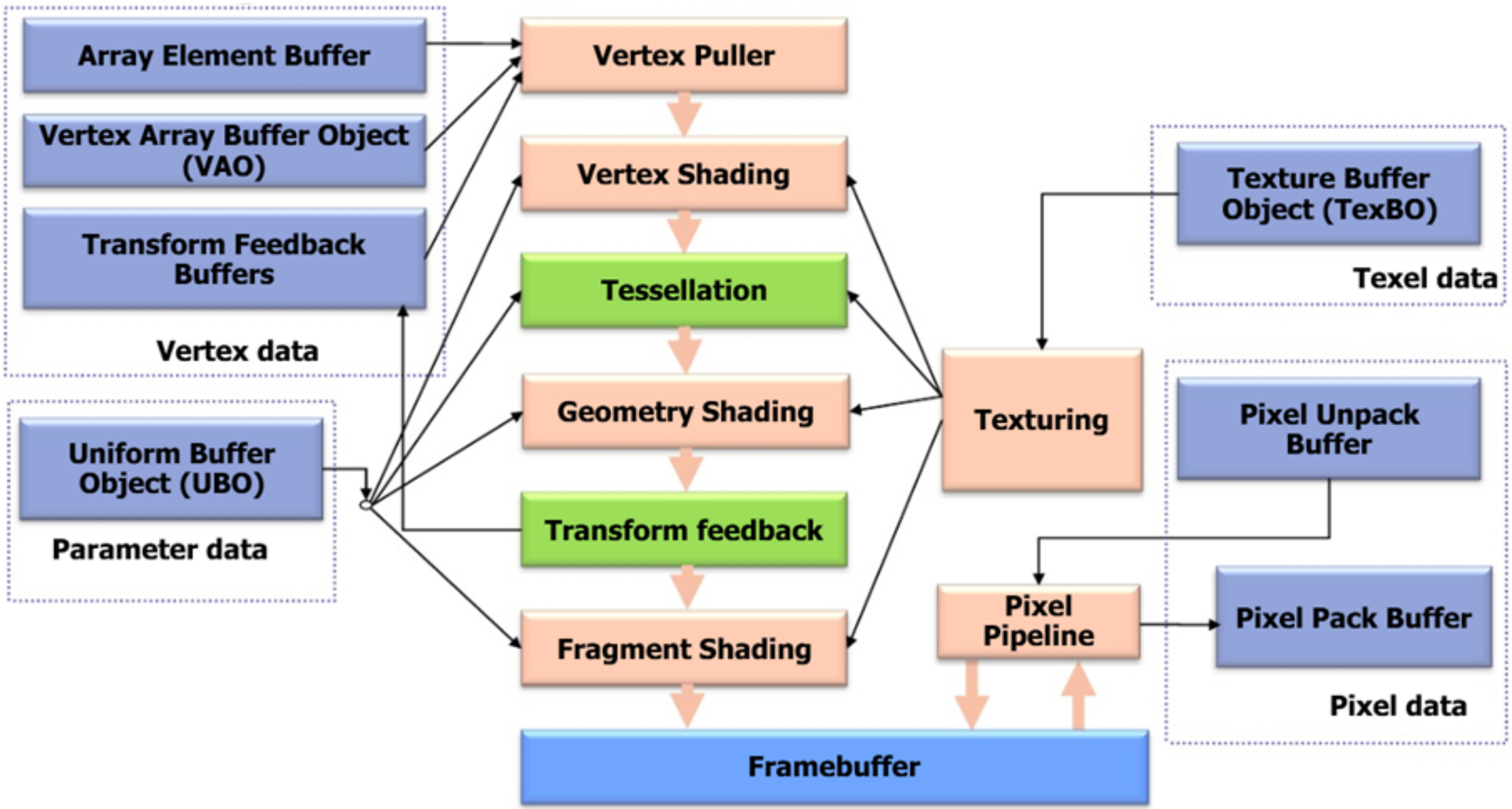
```
#version 140

uniform Transformation {
    mat4 projection_matrix;
    mat4 modelview_matrix;
};

in vec3 vertex;

void main() {
    gl_Position = projection_matrix * modelview_matrix * vec4(vertex, 1.0);
}
```

OpenGL 4.x pipeline



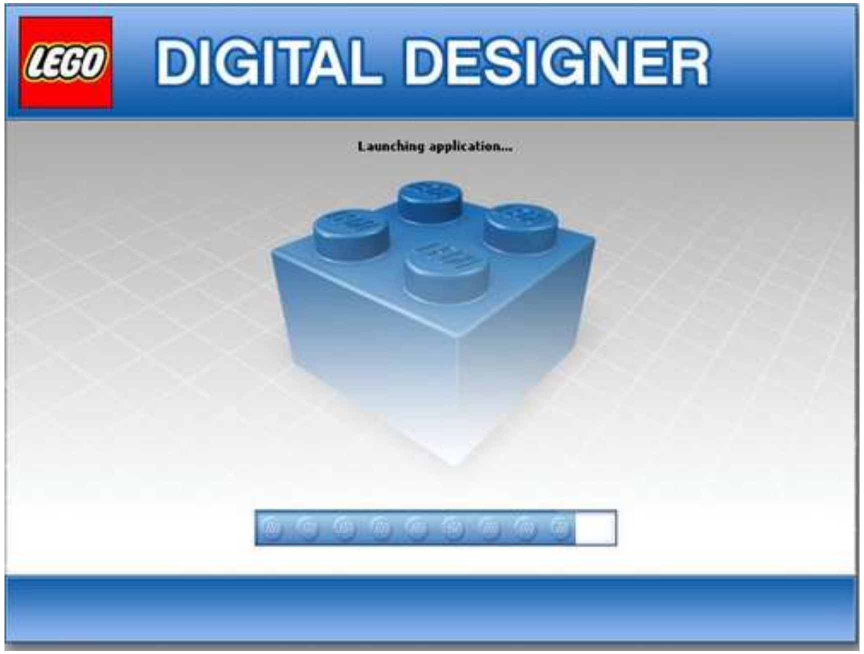
From http://www.khronos.org/developers/library/overview/opengl_overview.pdf

Examples of programmable graphics

- Lego Digital Designer
- Subsurface scattering
- Molecular visualization

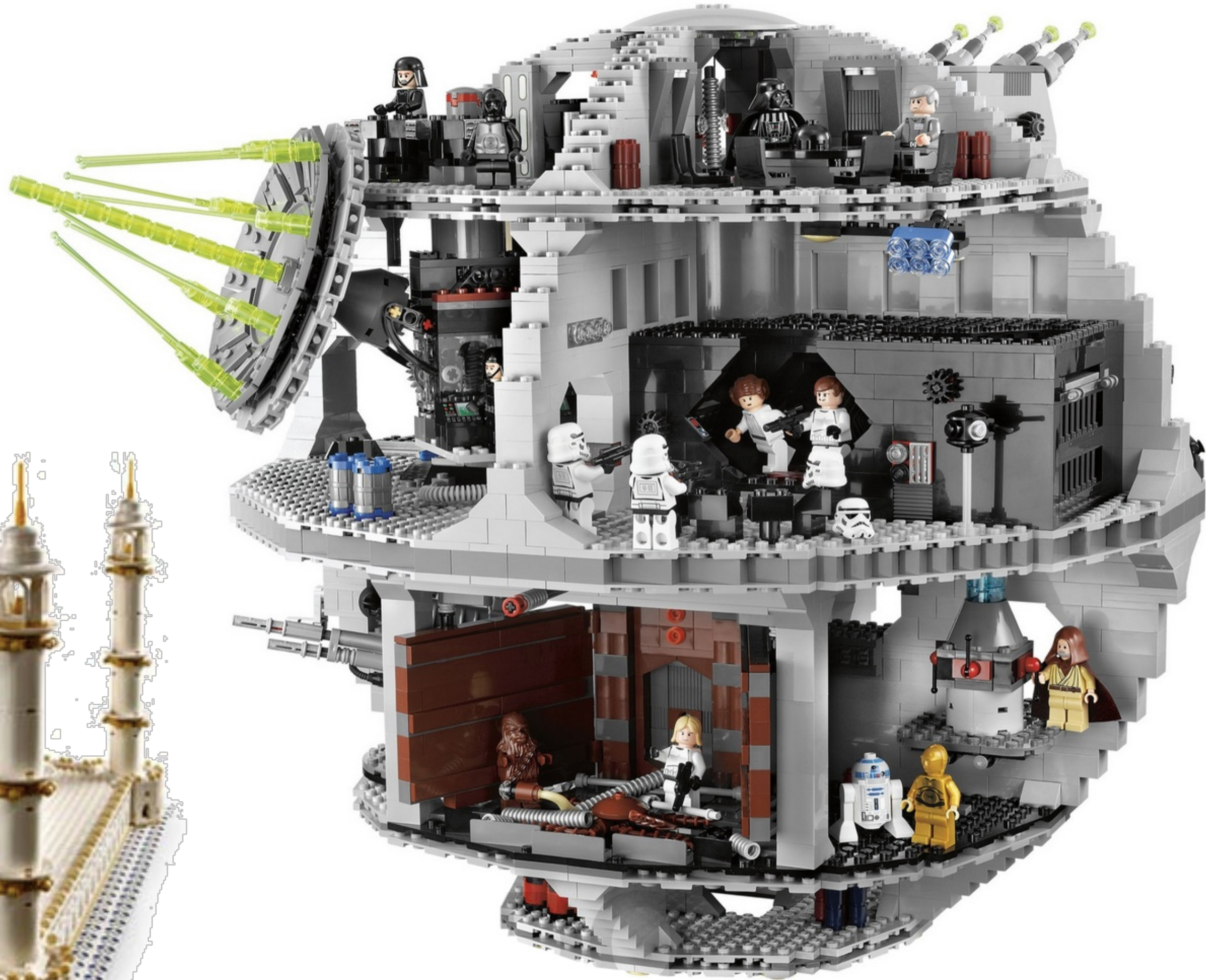
ALEXANDRA
INSTITUTTET

Lego Digital Designer 3 → 4



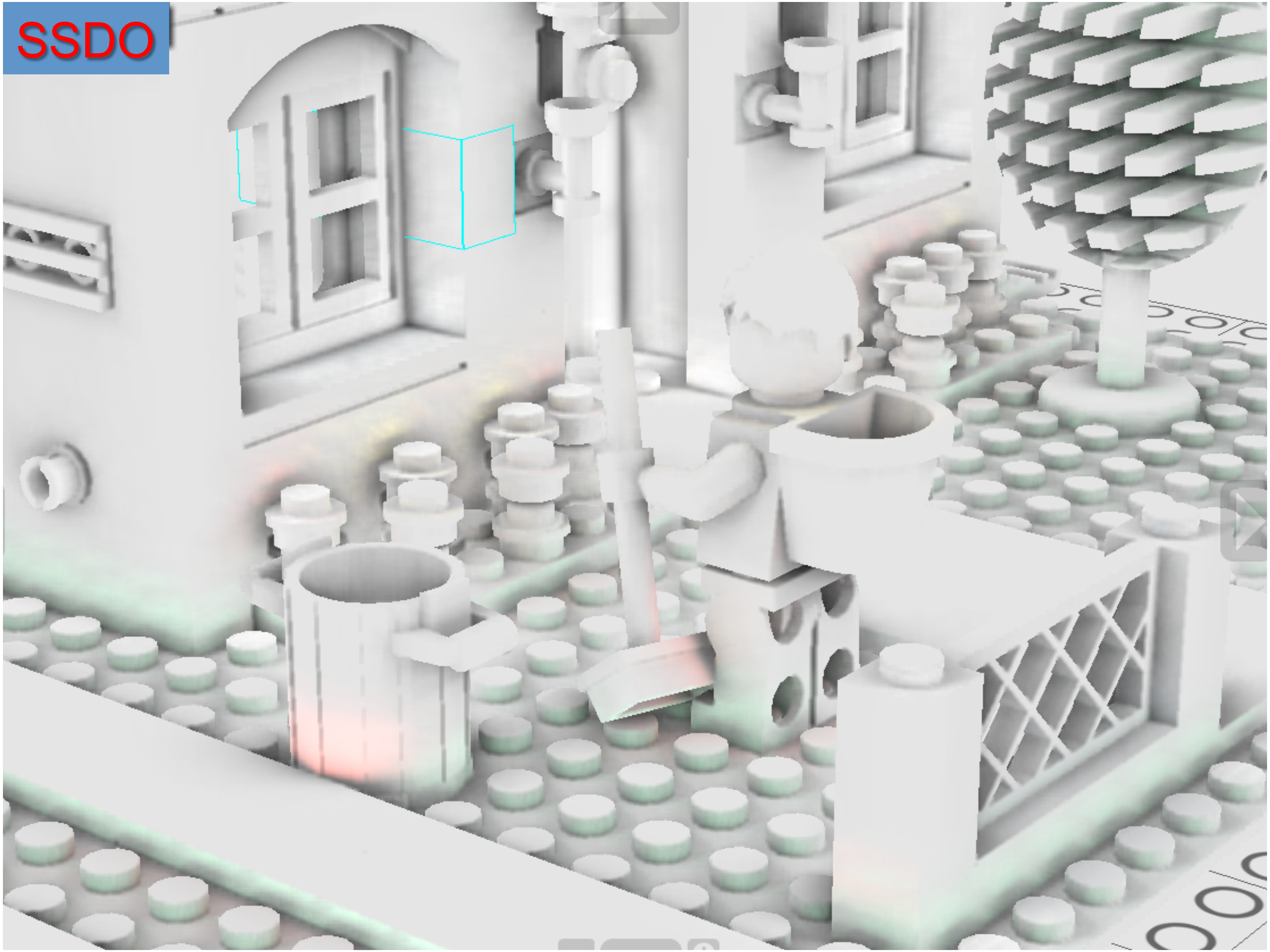
YES... Playing with LEGO at work

- 5.922 Taj Mahal
- 3.803 Death Star



Without SSDO (3.0)

With SSDO (4.0)


SSDO

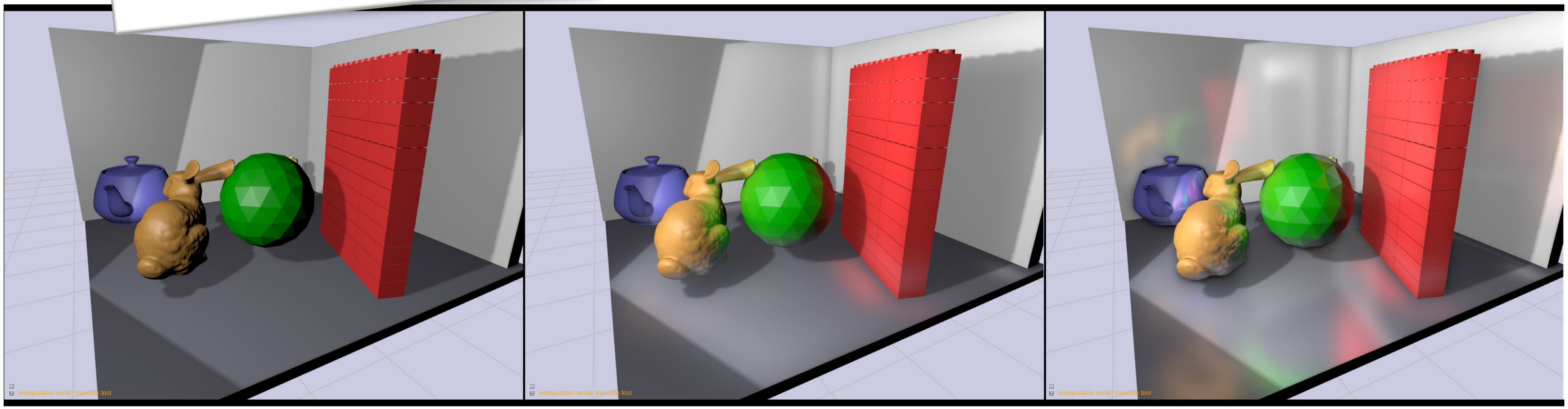
Light Probagation Volumes


Cascaded Light Propagation Volumes for Real-Time Indirect Illumination

Anton Kaplanyan* Crytek GmbH
Carsten Dachsbacher† VISUS / University Stuttgart

- Crytek's realtime Global Illumination

Kaplanyan, A. and Dachsbacher, Cascaded light propagation volumes for real-time indirect illumination. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on interactive 3D Graphics and Games*

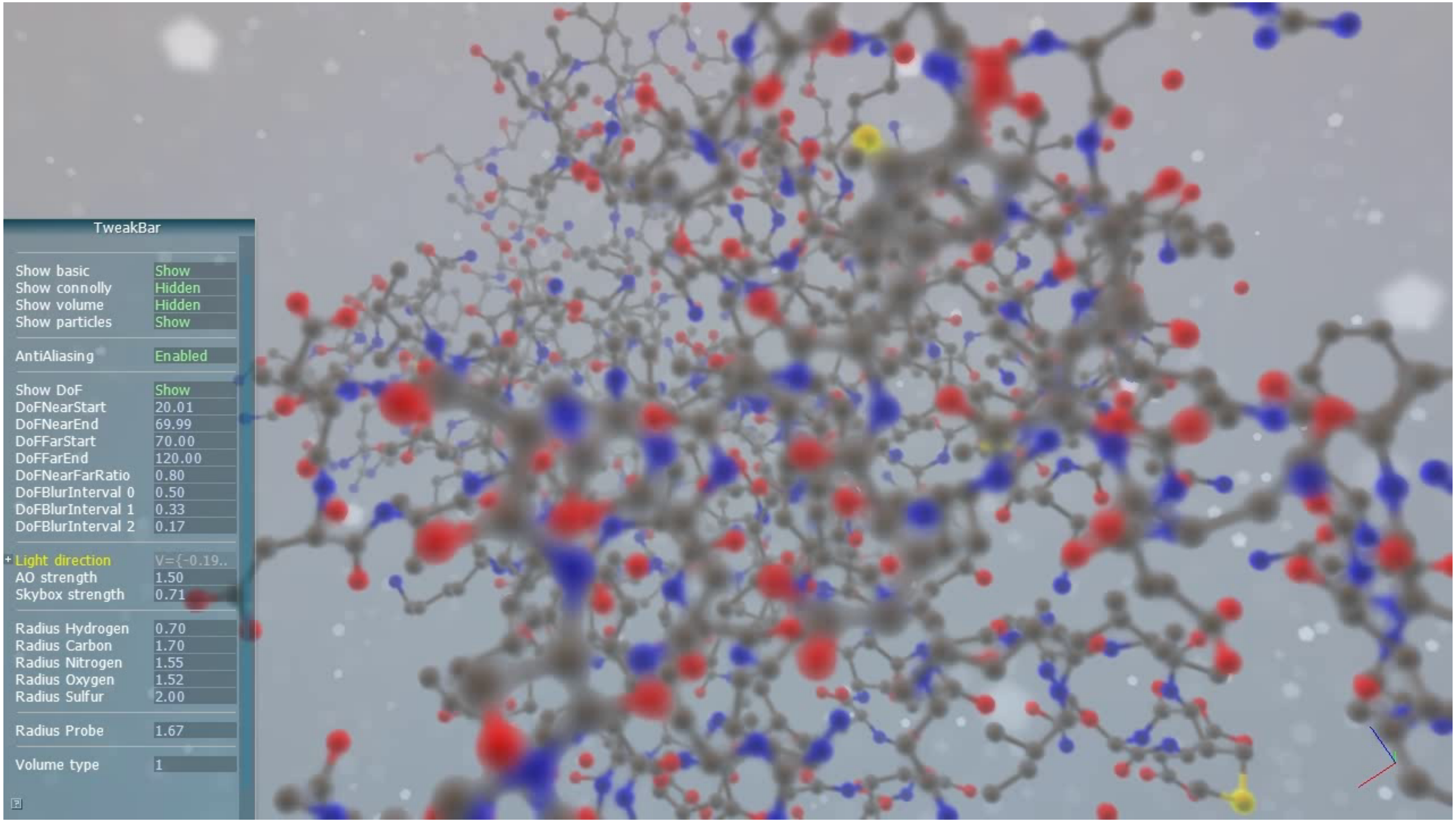


Realtime Subsurface scattering

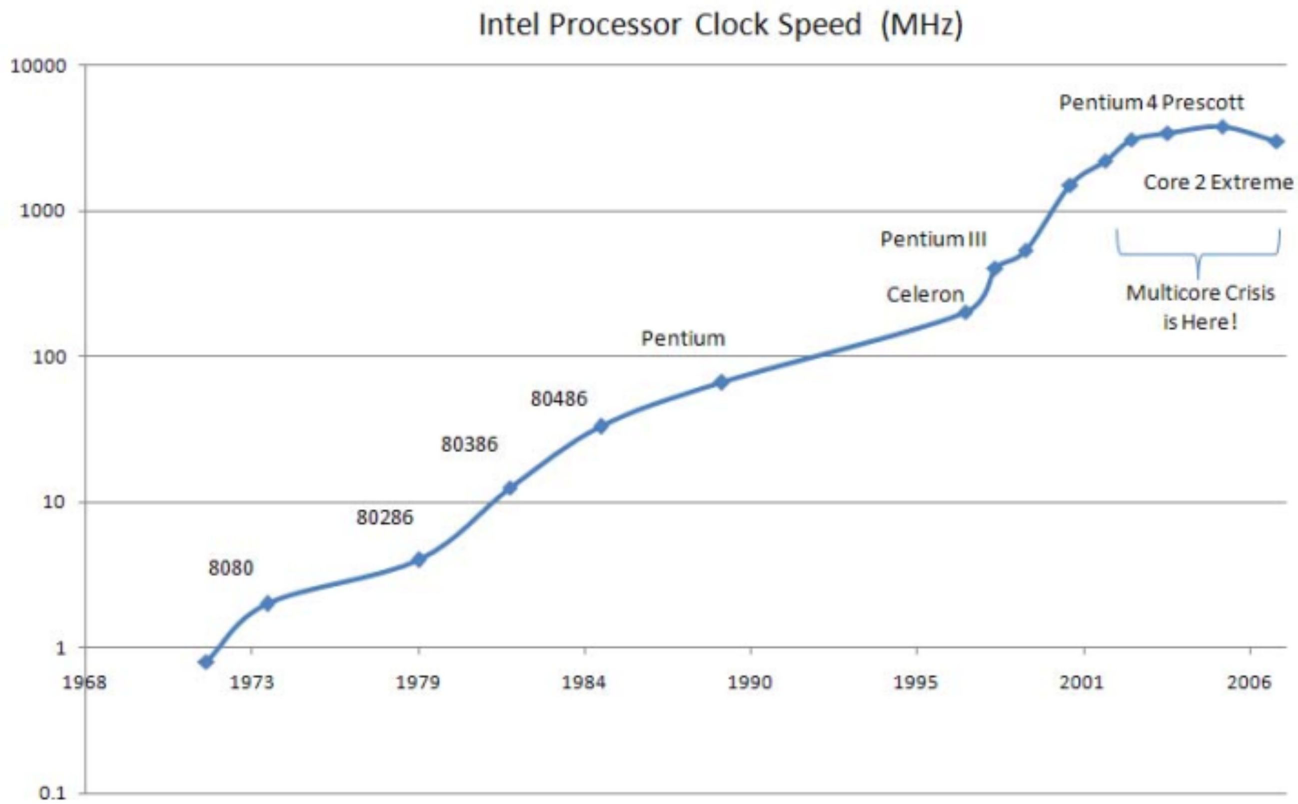
SSLPV: subsurface light propagation volumes. In *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics* (HPG '11)

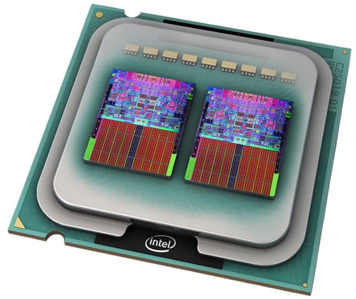
ALEXANDRA INSTITUTTET

Molecular visualization



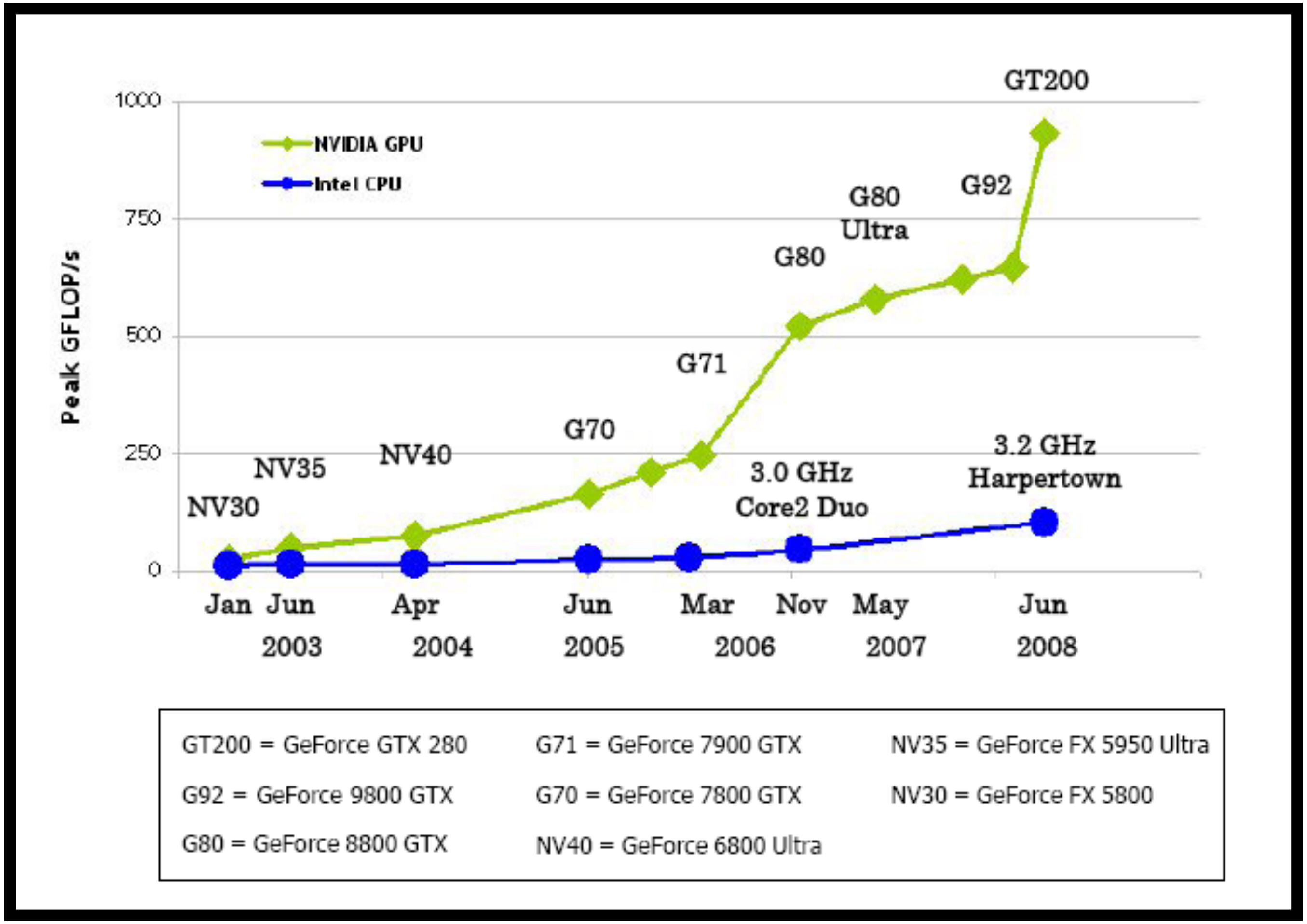
Multicore crisis

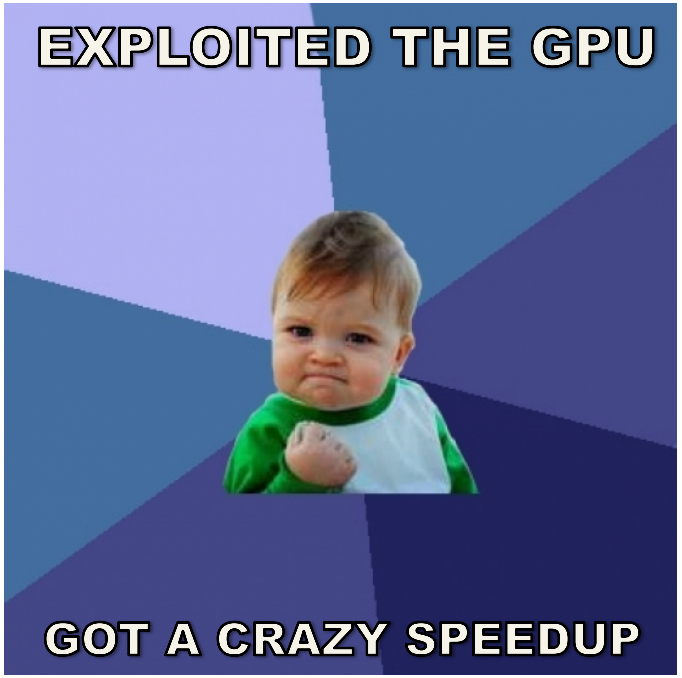




ALEXANDRA INSTITUTTET

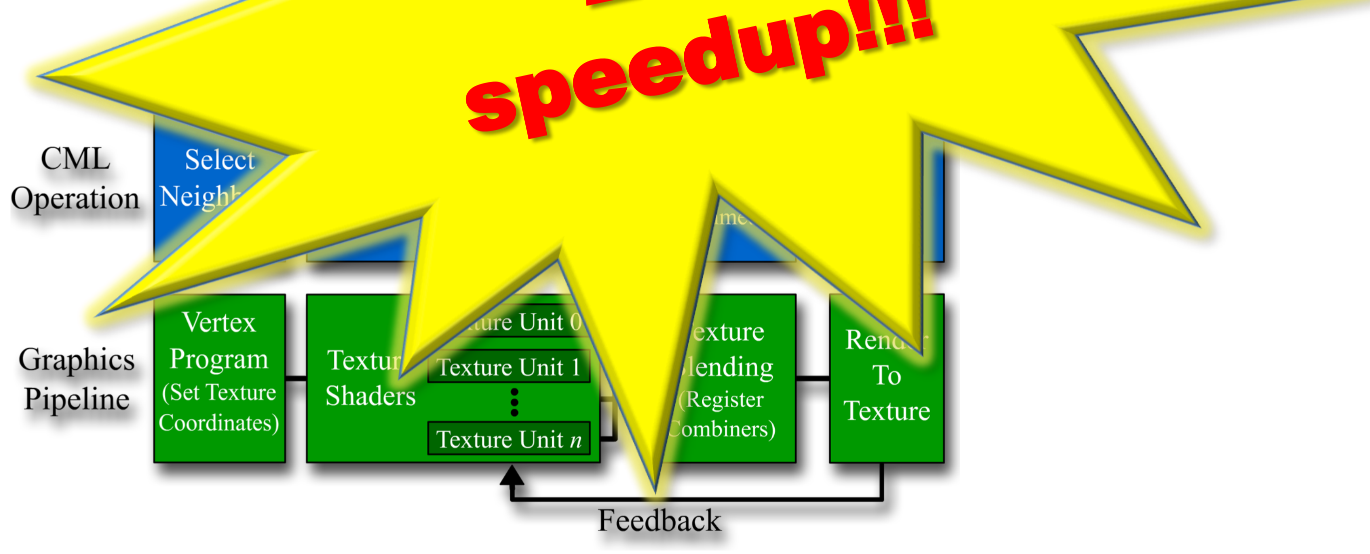
Computing power of the GPU




EXPLOITED THE GPU
GOT A CRAZY SPEEDUP

CMLLab

- **Physically-Based Visual S ulation on Graphics Hardware.** Mark J. Harris, g C be, Thorsten Scheuermann, an Proc. 2002 *SIGG Graphics Hardware 2*



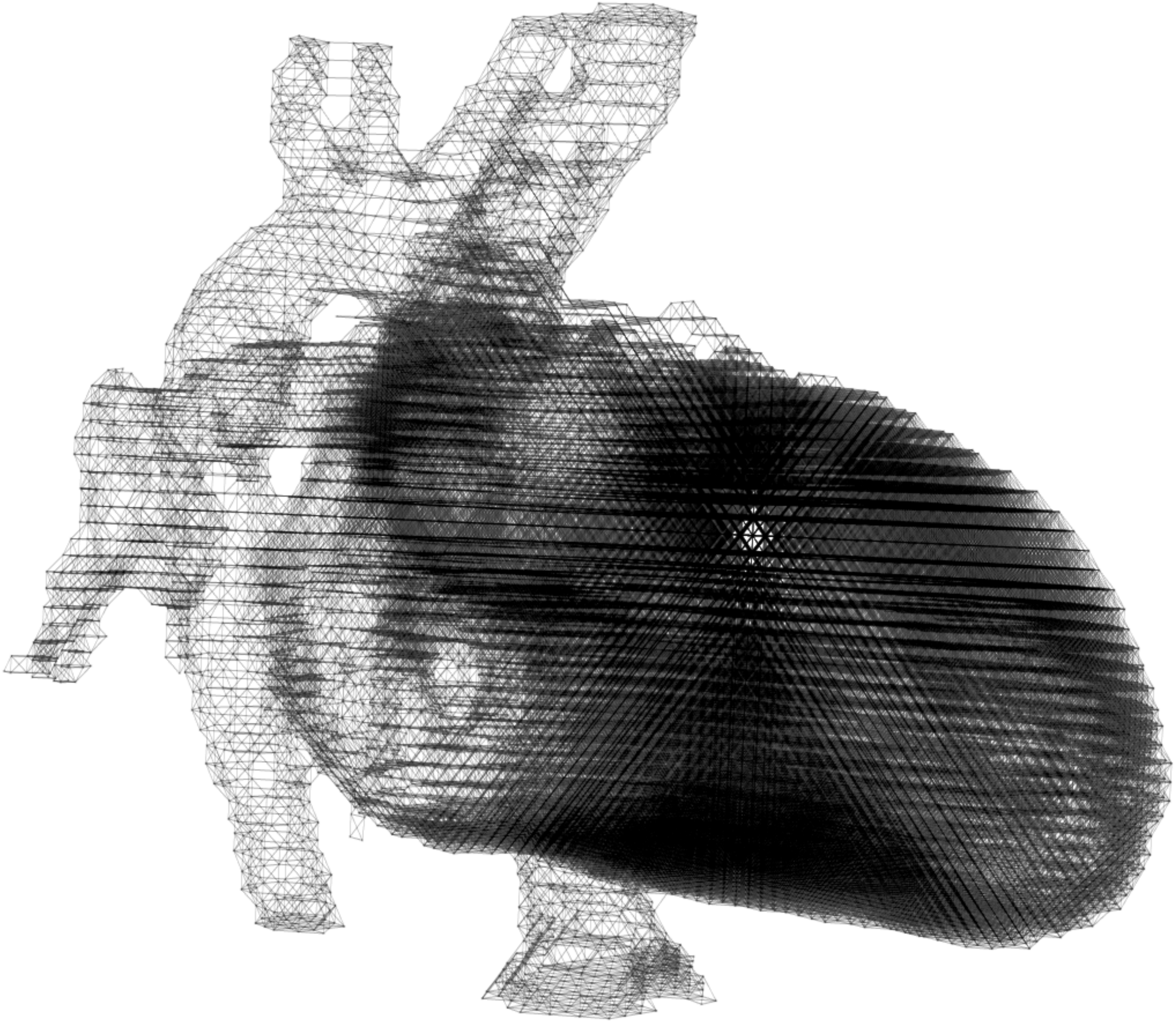
Ignoring early work in the Ikonas (1978), the Pixel Machine (1989) and Pixel Planes 5 (1992)

ALEXANDRA INSTITUTTET

My adventure in gpgpu land

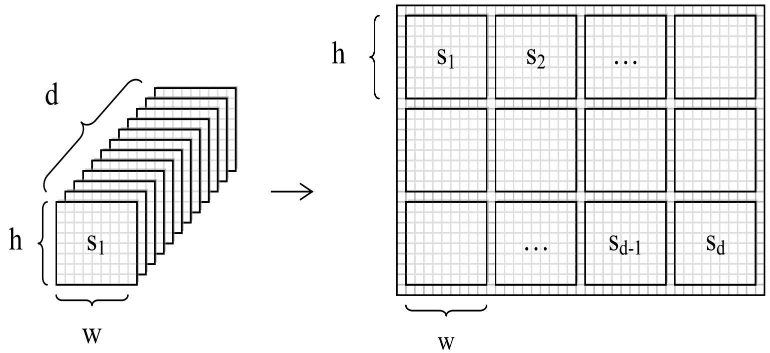
- ... a PhD on surgical simulators for procedures on children with malformed hearts

Physics systems

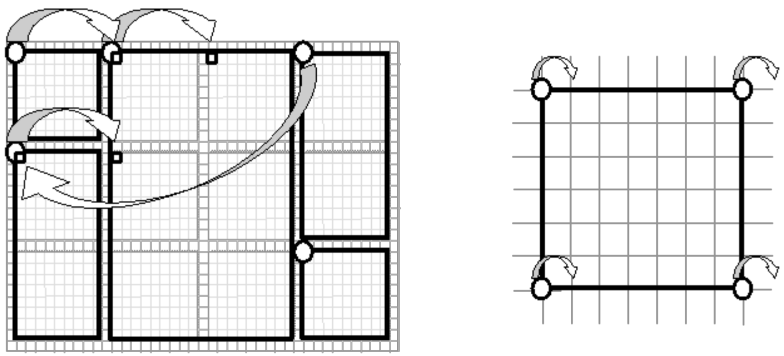


Mapping to 2D render-target

- 3D grid → 2D texture
  – Flat 3d-texture



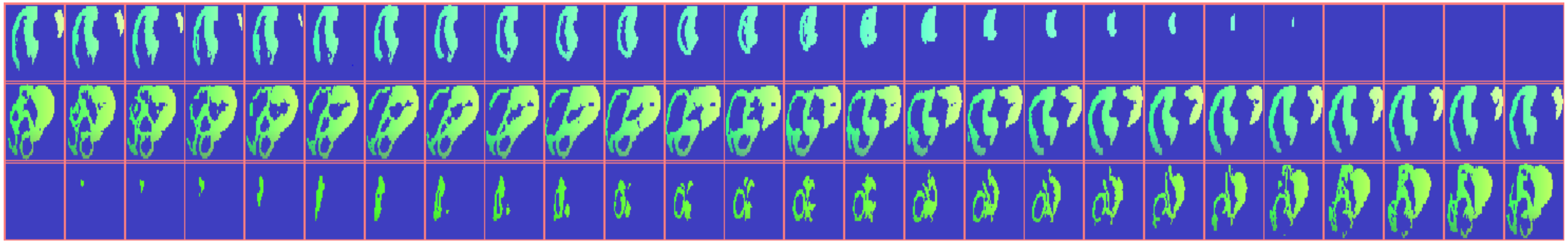
- Per vertex texture coordinates for neighbors



a)   b)

Approximation of arbitrary shapes

- That is, some fragments are not valid particles
  - Exclude calculations with a depth-test based cull as well as fragment based conditional kill



I don’t like graphics

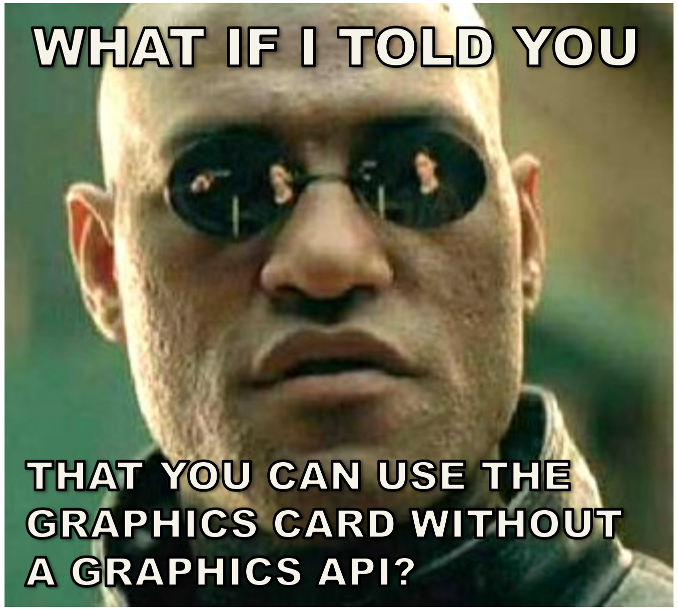
- Graphics API is about graphics
- Limitied memory model by textures
- Limited shader capabilities
- Lack of integer and bit operations
- Communication limit between pixels
- No scatter operation

ALEXANDRA INSTITUTTET

Away with the graphics

- Early academic work
  - BrookGPU (2004)

- CTM (ati) - 2006
- Cuda (nvidia) - 2007
- OpenCL - 2008

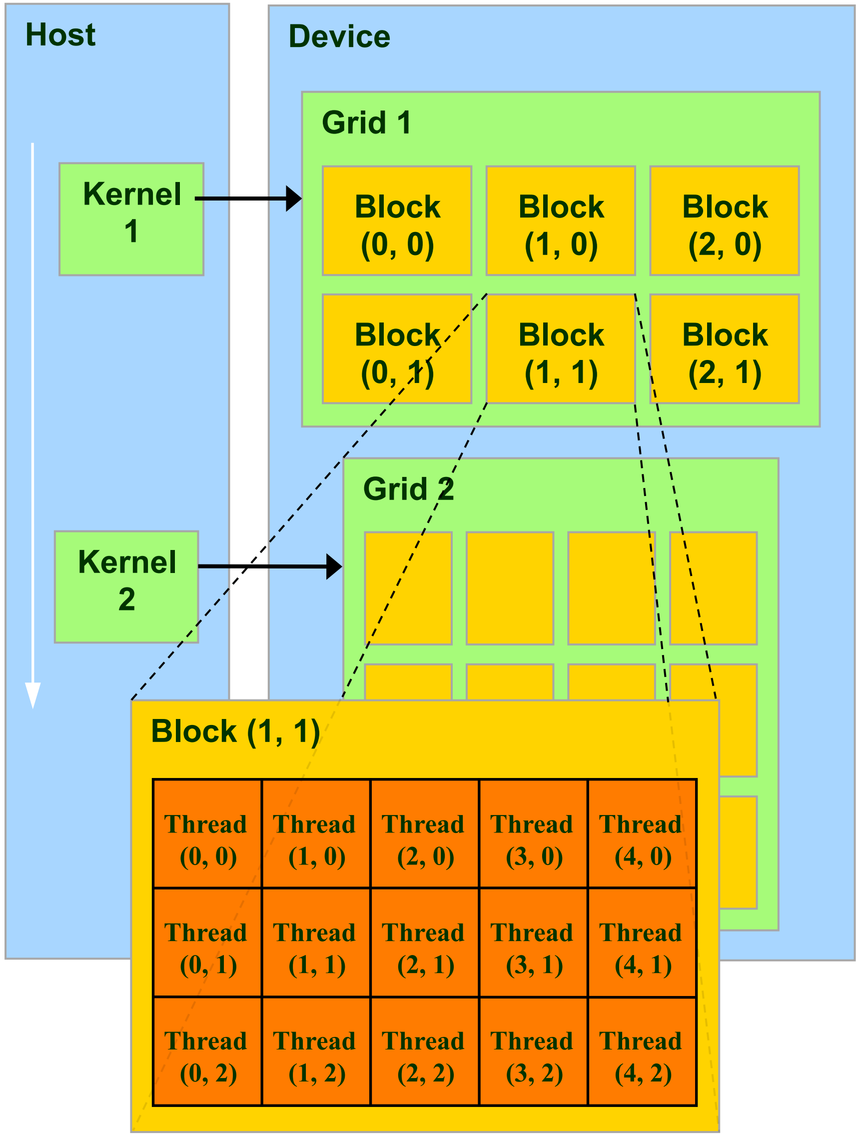


CUDA

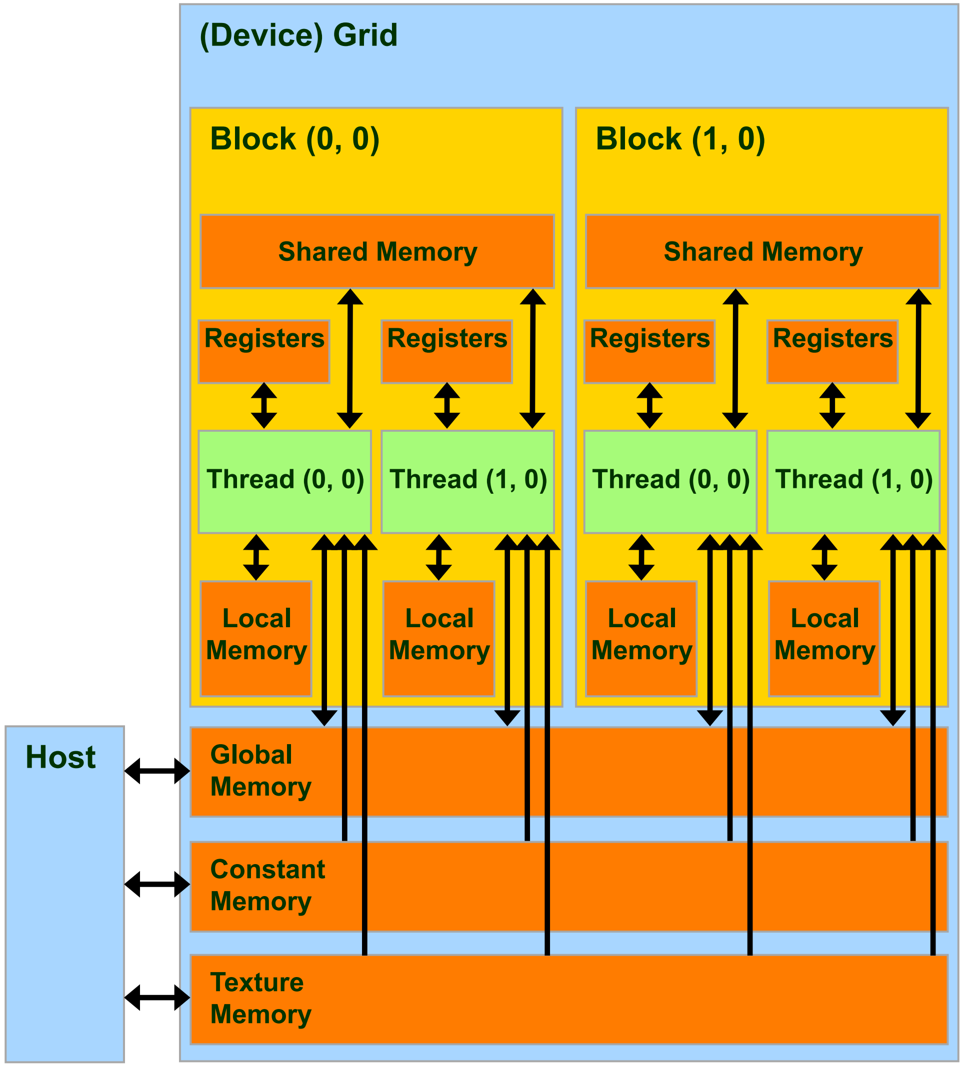
- Compute Unified Device Architecture
  - Compute oriented language
  - Extension of C
  - A kernel is executed as a number of threads in parallel
    - Lightweight
    - 1000s of threads for full efficiency
    - SIMD (mostly)

- Heterogenous computing
  - Host and device

Grids, blocks, threads



CUDA memory space



ALEXANDRA INSTITUTTET

OpenCL, Khronos group

- Much the same as CUDA

CUDA term	OpenCL term
GPU	Device
Multiprocessor	Compute Unit
Scalar core	Processing element
Global memory	Global memory
Shared (per-block) memory	Local memory
Local memory (automatic, or local)	Private memory
kernel	program
block	work-group
thread	work item

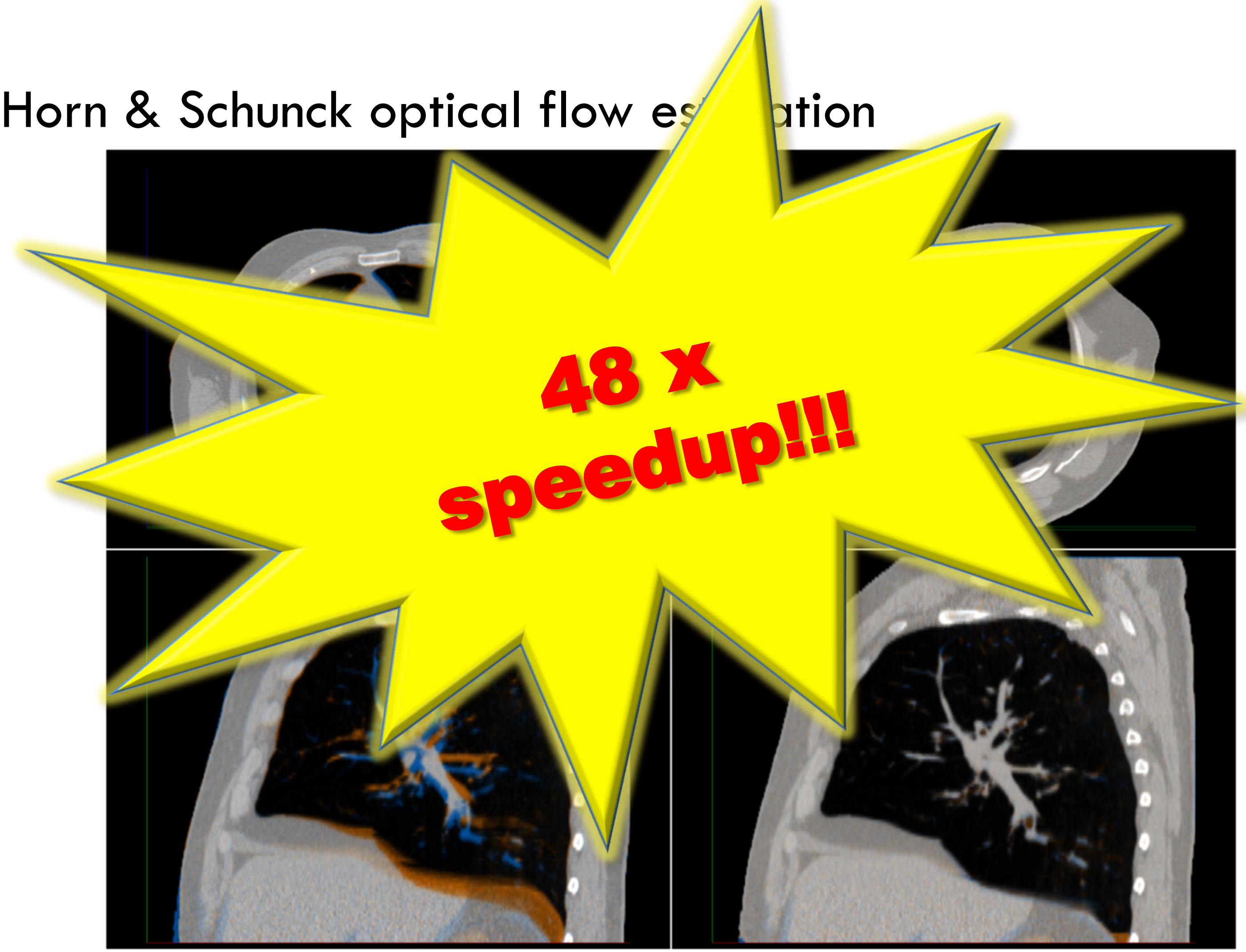
GPGPU work at the Alexandra Institute

- LEGO, 3D services
- Luxion, spatial acceleration structures
- BrainReader, Optical flow registration

ALEXANDRA
INSTITUTTET

POPI 4D Thorax registration

- Horn & Schunck optical flow es ation



Acceleration and validation of optical flow based deformable registration for image-guided radiotherapy. **K.Ø. Noe**, B.D. de Senneville, U.V. Elstrøm, K. Tanderup, T.S. Sørensen. Acta Oncologica 2008; 47(7):1286-1293.

Optical flow registration

- 3D grid of displacement vectors
  – From one dataset to another

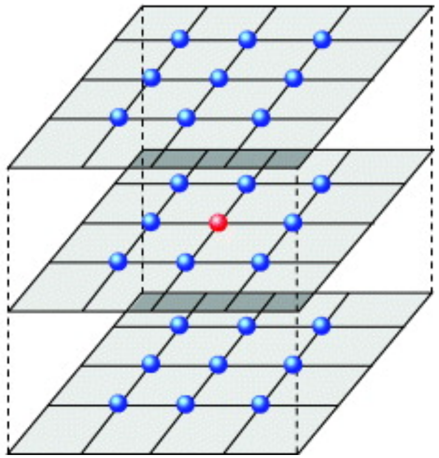
- Find optimum of the following;

$$E = \iiint \left(\mathscr{E}_b^2 + \alpha^2 \mathscr{E}_c^2\right) \,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$$

$$\mathscr{E}_b = I_x u + I_y v + I_z w + I_t.$$

$$\mathscr{E}_c^2 = ||\nabla u||^2 + ||\nabla v||^2 + ||\nabla w||^2$$

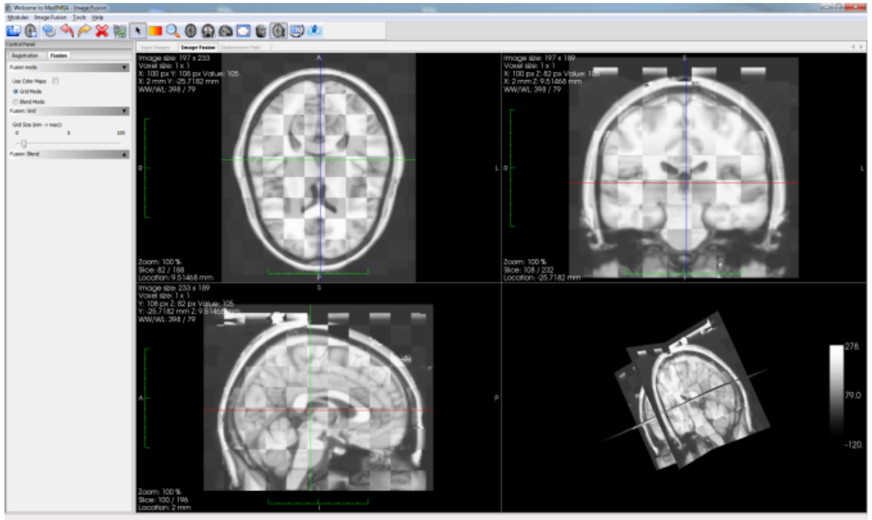
- Euler-Lagrange
  - Integral to differential equation
- Finite difference
  - discretized
  - → iterative local update scheme
- Multiresolution
  - Global solution

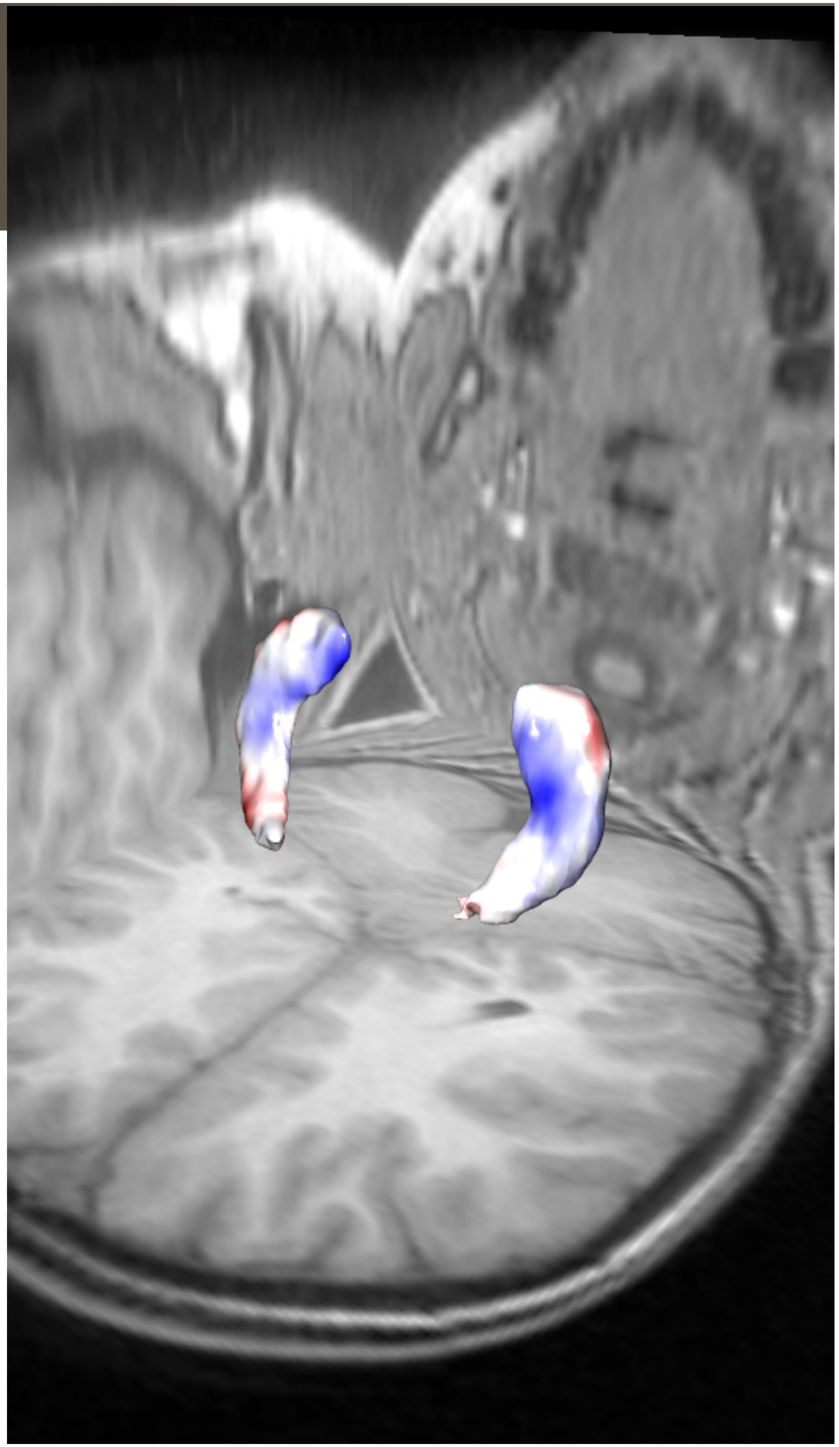


ALEXANDRA INSTITUTTET

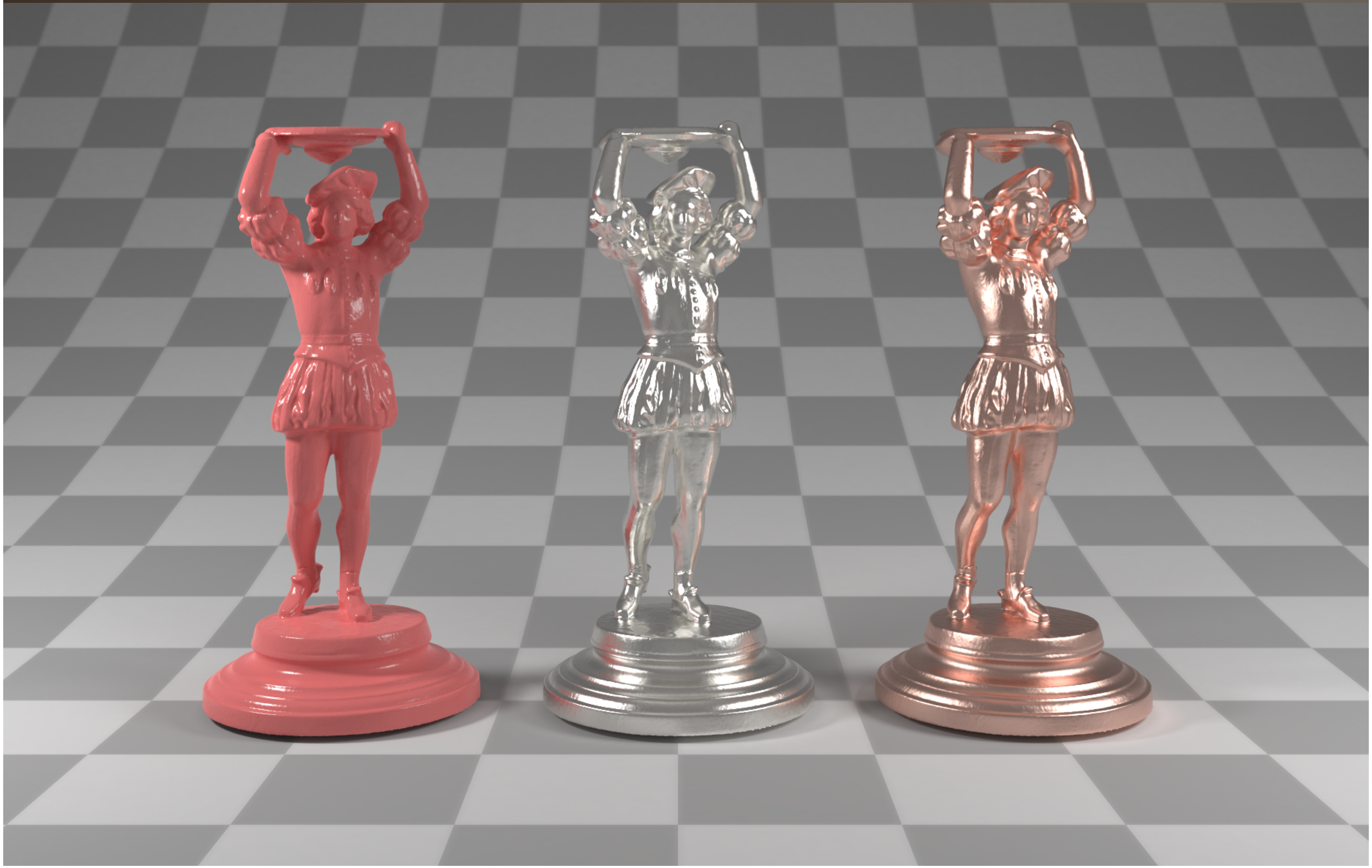
BrainReader ApS

- Registration of the hipocampus



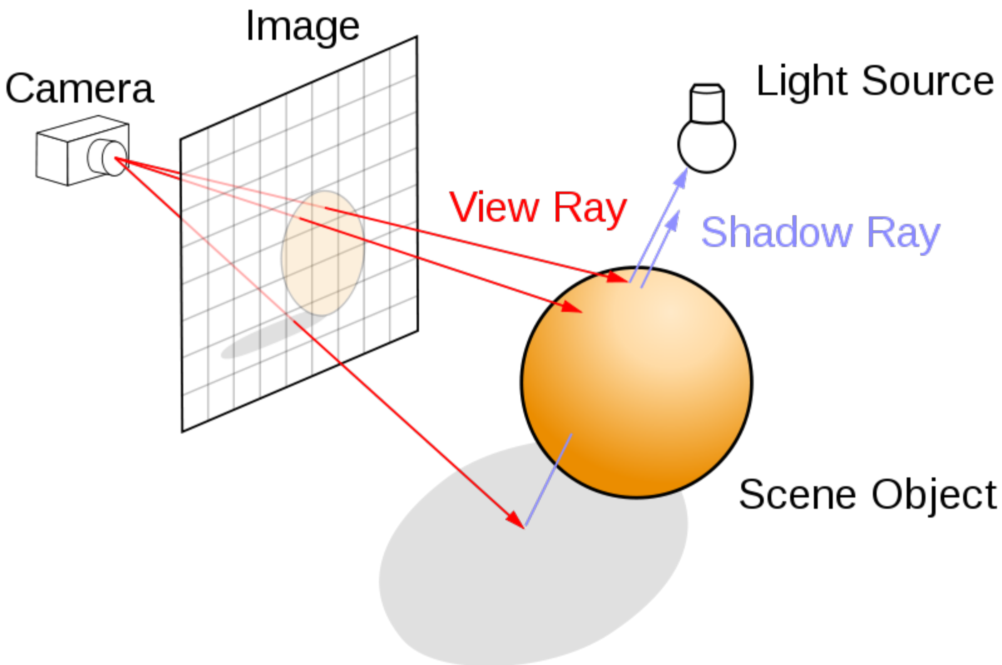


Photorealistic... ”Easy” enough



Photorealistic interactive images

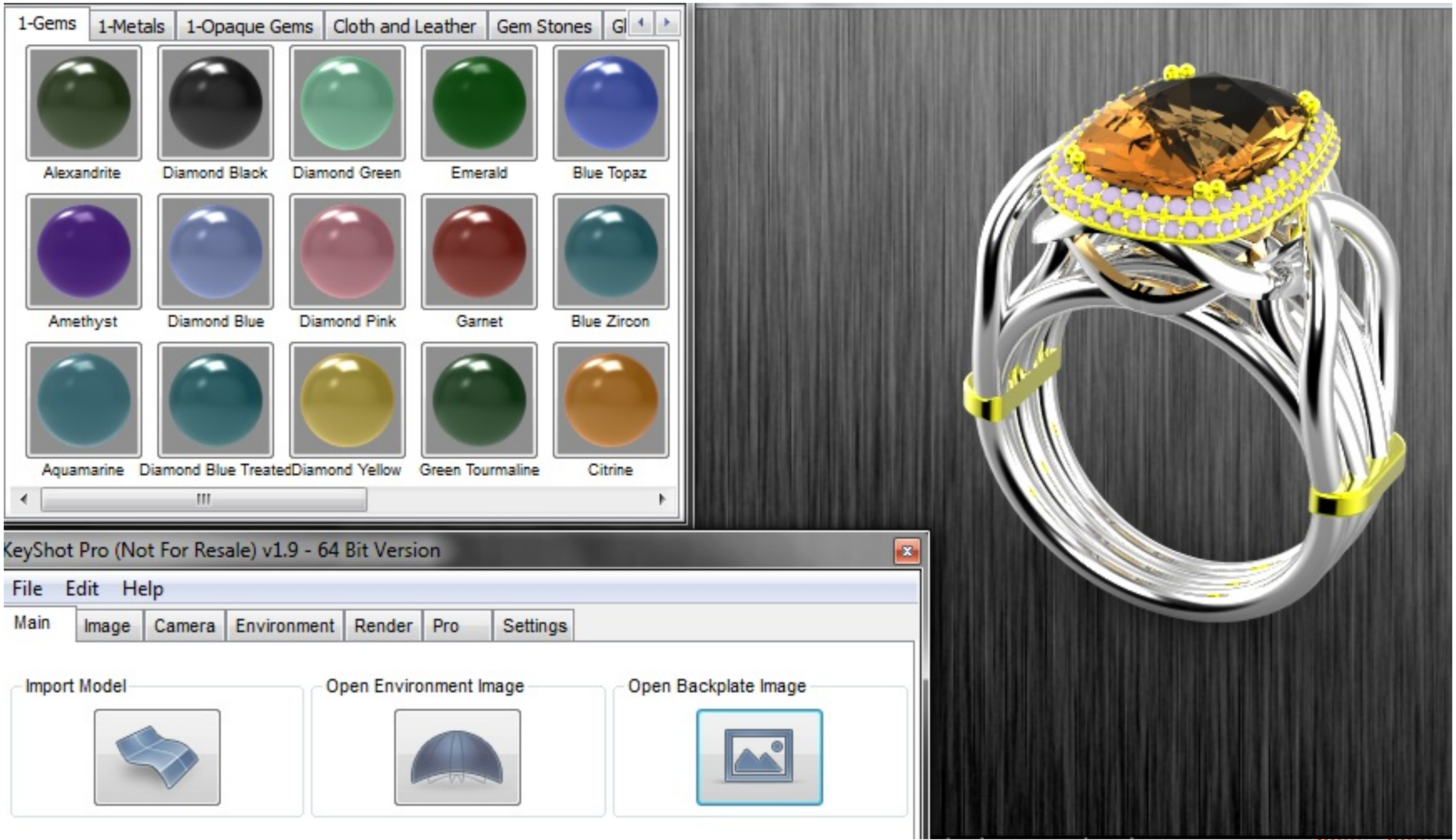
- Fast raytracing



Luxion: GPU/CPU raytacing

- Professor Henrik Wann Jensen

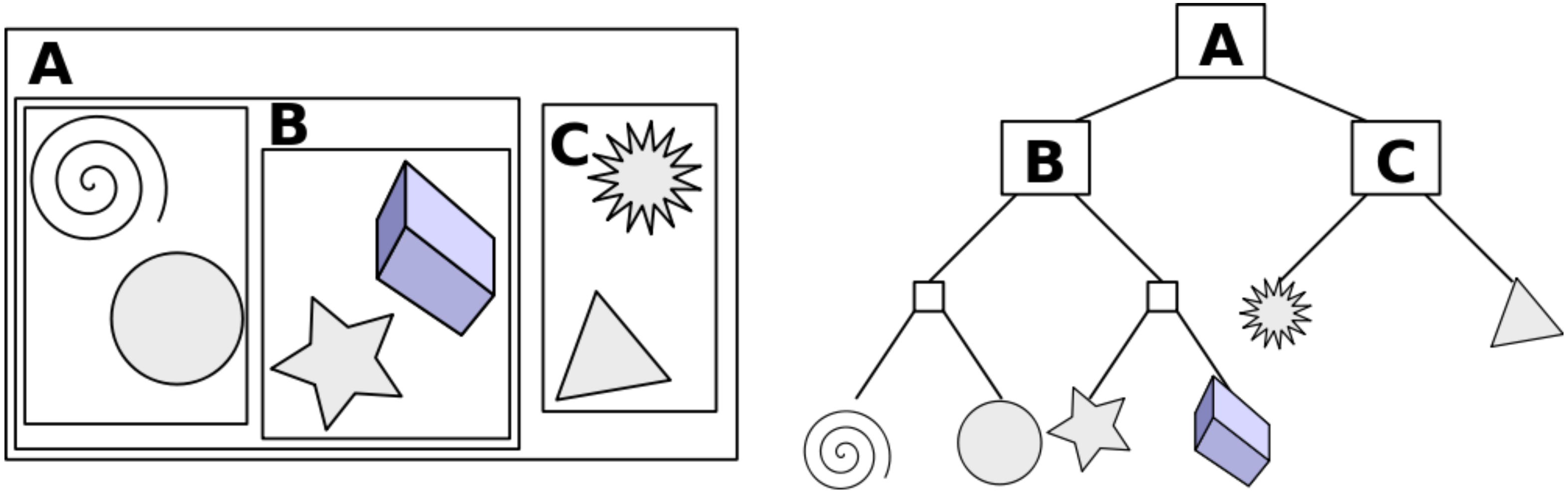
Keyshot



1-Gems
1-Metals
1-Opaque Gems
Cloth and Leather
Gem Stones
Alexandrite
Diamond Black
Diamond Green
Emerald
Blue Topaz
Amethyst
Diamond Blue
Diamond Pink
Garnet
Blue Zircon
Aquamarine
Diamond Blue Treated
Diamond Yellow
Green Tourmaline
Citrine
KeyShot Pro (Not For Resale) v1.9 - 64 Bit Version
File
Edit
Help
Main
Image
Camera
Environment
Render
Pro
Settings
Import Model
Open Environment Image
Open Backplate Image

Spatial Data Structures

- E.g. Bounding Volume Hierarchy

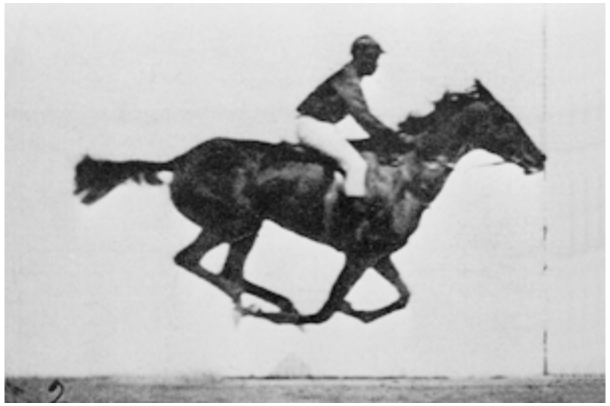


- GPS location, GIS systems, BIM systems
- ... And ray tracing (through ray-triangle query)

ALEXANDRA INSTITUTTET

Dynamic spatial objects

- Rebuild many times, queries many times
  - Could refit or do partial rebuilds
- We focus on FAST and COMPLETE rebuild
  - Based on a series of papers at ”High Performance Graphics” 2010-2012

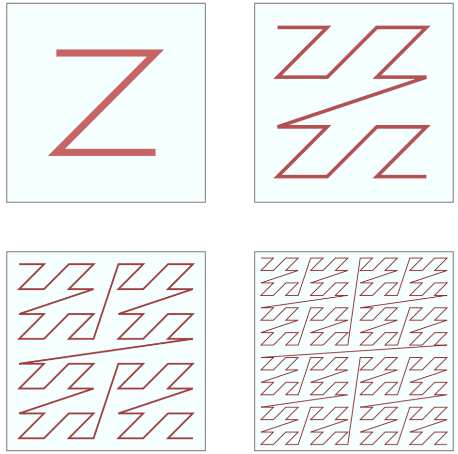


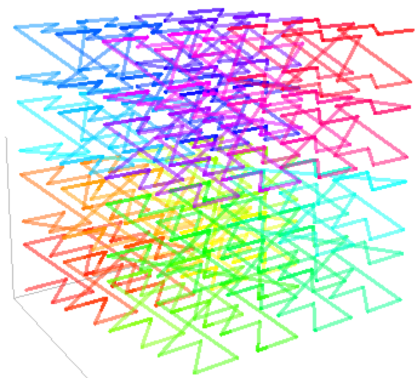
HLBVH

- **HLBVH: Hierarchical LBVH Construction for Real Time Ray Tracing of Dynamic Geometry (2010)**

Computing Morton number

y \ x:	0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
y: 0 000	000000	000001	000100	000101	010000	010001	010100	010101
1 001	000010	000011	000110	000111	010010	010011	010110	010111
2 010	001000	001001	001100	001101	011000	011001	011100	011101
3 011	001010	001011	001110	001111	011010	011011	011110	011111
4 100	100000	100001	100100	100101	110000	110001	110100	110101
5 101	100010	100011	100110	100111	110010	110011	110110	110111
6 110	101000	101001	101100	101101	111000	111001	111100	111101
7 111	101010	101011	101110	101111	111010	111011	111110	111111





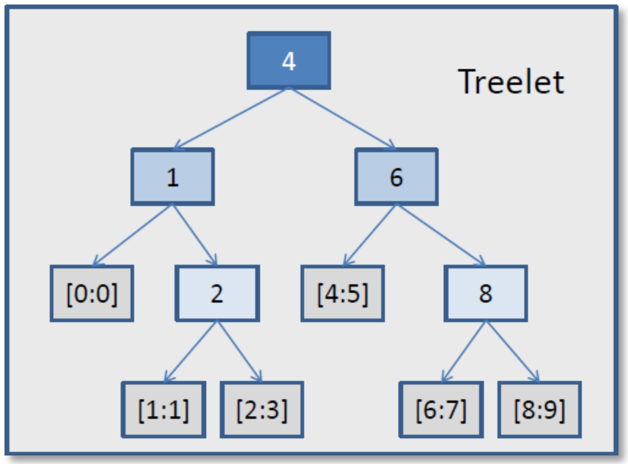
ALEXANDRA INSTITUTTET

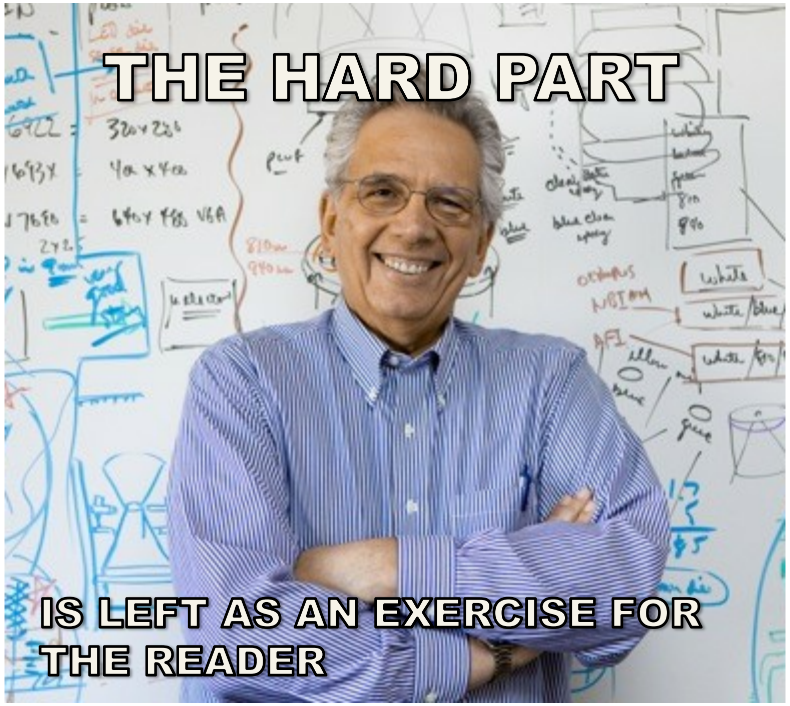
From sorted prims to tree

0	1	2	3	4	5	6	7	8	9
0	0	0	0	1	1	1	1	1	1
0	1	1	1	0	0	1	1	1	1
1	0	1	1	1	1	0	0	1	1
...	...	...	...	...	...	...	...	...	...

Block descriptor

4			
1	6		
-	2	-	8




THE HARD PART
IS LEFT AS AN EXERCISE FOR THE READER

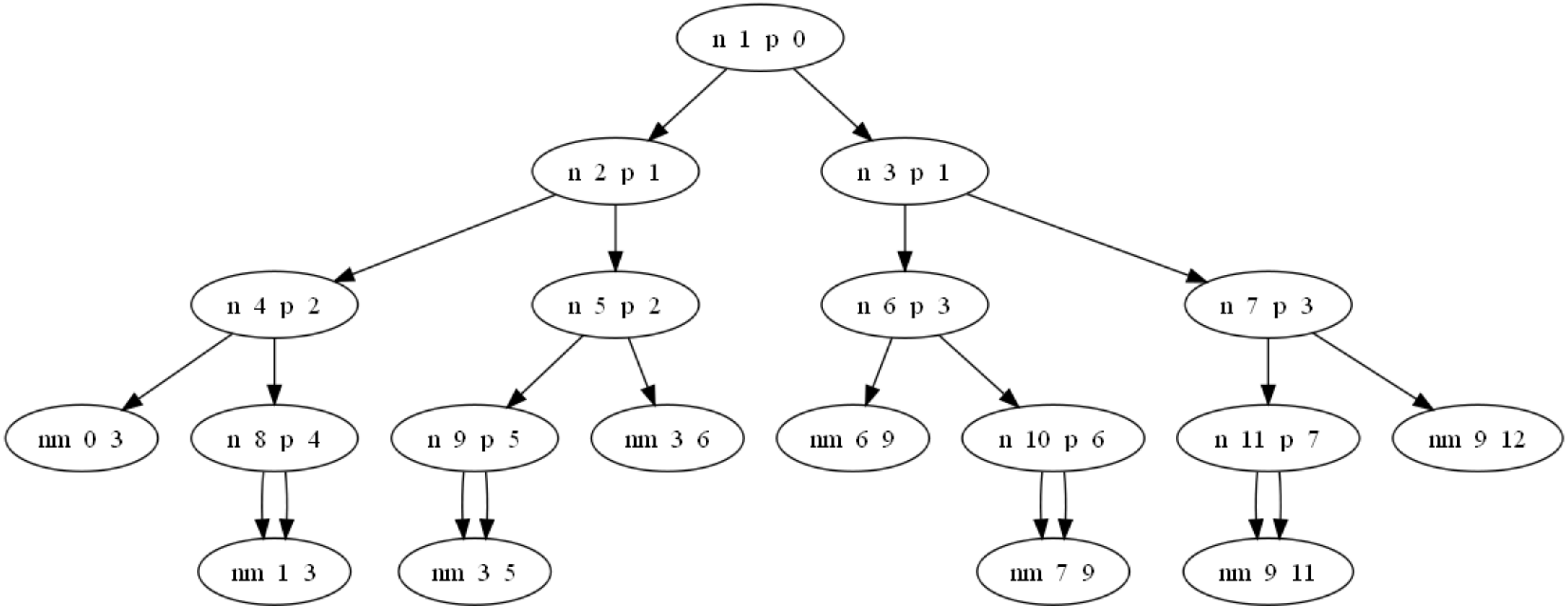
Devil is in the detail

- From paper to working was difficult
  - *Array pointing to array of arrays pointing to the head of an index of another heads index to an array in a segmentes part of the treelet*

  - No debugger (at the time)
  - No source code from author
  - Segmentation fault → full reboot

> *"You really implemented Jacopo's paper? That's really cool. [snip] When I was asked to implement Jacopo's paper I failed (or was lazy) and that's why I developed HLBVH2 which was simpler. That's why a new paper appeared."* Kirill Garanzha

Debug output til dot graph



Prefixsum is pure magic

- The size of each treelet varied based on the subdivisions according to morton code
  - How do you find the write position, and how do you know how much memory to allocate ?

Prefixsum

Segment	1	2	3	4	5	6	7
Emit size	0	3	2	1	2	4	0
prefixsum	0	3	5	6	8	12	12

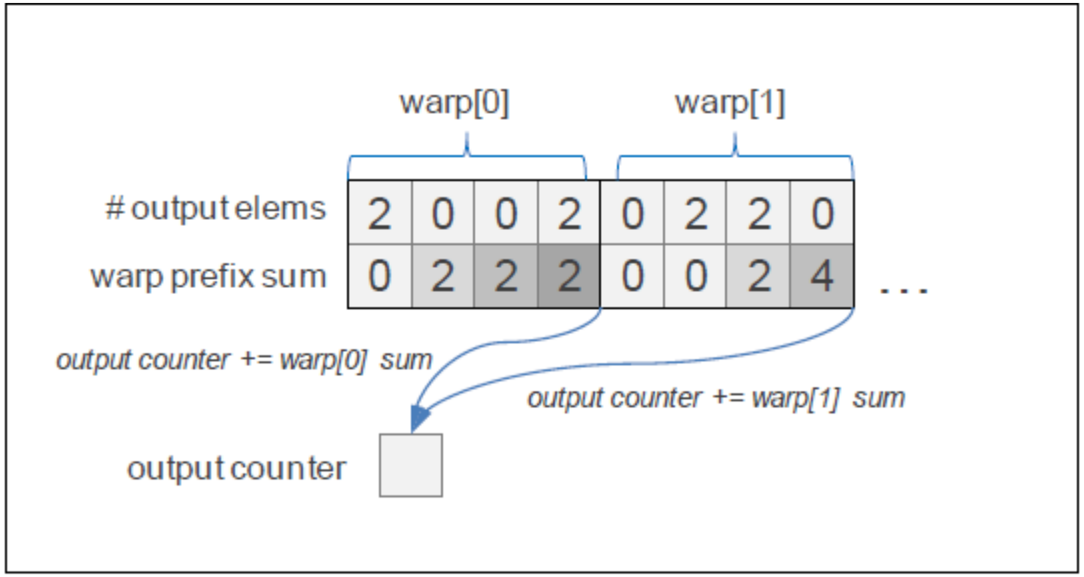
HLBVH 2

- Kirill Garanzha et. al. 2011. Simpler and faster HLBVH with work queues. In *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*

- 5-10 times faster than HLBVH 1

Basic Ideas of HLBVH 2

- Task queue
- Each task is a node (of the finished tree)
- Each task is processed by one thread (in a warp)



Per warp prefix sum

```
static __device__ int scanWarpPopc(int *start_write_offset, // shared memory for this warp
   int active, // 0 if one or zero nodes are generated, 1 if two nodes are generated
   int *output_counter)
{
   uint active_mask = __ballot(active); // bitmask with 1 for each threads in current warp that
    will output two queue jobs
   uint thread_write_offset = __popc(active_mask << (WARP_SIZE - threadIdx.x)) * scale; // sum of
    1-bits in the mask, before the thread itself
   uint warp_write_offset = __popc(active_mask) * scale; // total number of threads in current
    warp that will output.

   if(threadIdx.x == 0 && warp_write_offset > 0)
   {
      start_write_offset[threadIdx.y] = atomicAdd(output_counter, warp_write_offset);    // global
    warp offset
   }

   return start_write_offset[threadIdx.y] + thread_write_offset; // return the position where the
    current thread can write
}
```

Workqueue loop

```
while(number_of_queue_elements > 0)
   {
      host_counter[0] = 0;
      cudaMemset(device_counter, 0, sizeof(int));

      int number_of_threads_needed = number_of_queue_elements;
      int number_of_blocks = ceil(number_of_threads_needed / ((float)WARP_SIZE) );

      dim3 grid(number_of_blocks,1,1);
      dim3 block(WARP_SIZE,1,1);

           mortonSplit_KERNEL<<<grid, block>>>(bit_level,
            thrust::raw_pointer_cast(&dev_morton_codes[0]),
            bottom_work_queues[qin].getQueue(),
            bottom_work_queues[1-qin].getQueue(),
            number_of_queue_elements,
            thrust::raw_pointer_cast(&bvh_build_nodes[0]),
            device_counter,
            total_number_of_nodes,
            max_number_of_prims_in_leaf);


      cudaMemcpy(host_counter, device_counter, sizeof(int), cudaMemcpyDeviceToHost);
      number_of_queue_elements = host_counter[0];
      total_number_of_nodes += host_counter[0];
      qin = 1 - qin; // swap the pointer
      bit_level--;
      number_of_bvh_levels++;
      bvh_level_offsets[number_of_bvh_levels] = total_number_of_nodes;
      level_counter++;
   }
```

HLBVH 3

- Tero Karras. **Maximizing Parallelism in the Construction of BVHs, Octrees, and k-d Trees.** *Proceedings of the EUROGRAPHICS Conference on High Performance Graphics 2012, Paris, France, June 25-27, 2012 2012*

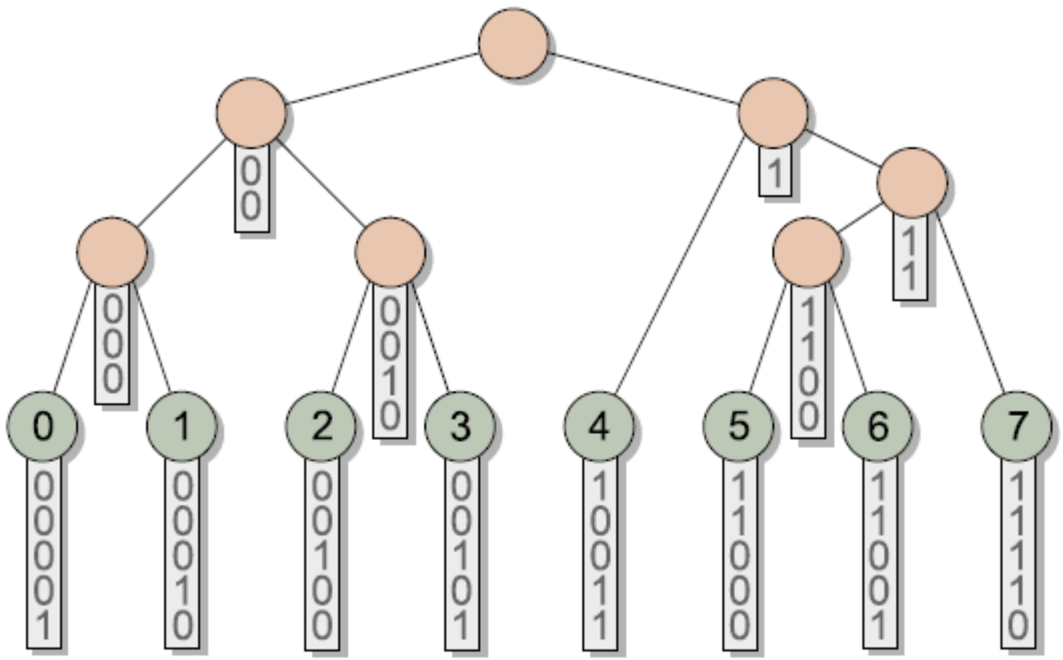
Basic Idea of HLBVH 3

- A limiting factor is that the node hierarchy is generated in a sequential fashion
  - In the first levels there might be very few elements, i.e. starving the highly parallel many core processor
  - Sublinear scaling with cores

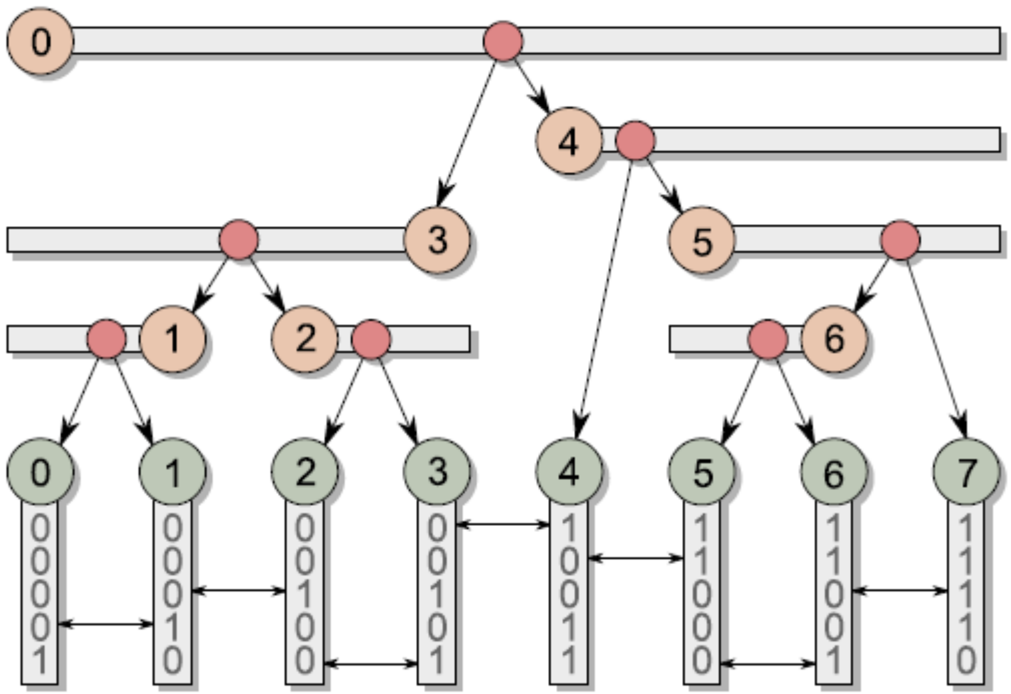
- So parallelize over all (internal) nodes of the tree

Binary Radix tree

- For n primitives there are n-1 internal nodes
- An internal node is the longest common prefix of the children



- Each internal node is stored at an index corresponding to its start range (if right child) or end range (if left child)



Clz - GPU

```
// returns the length of the longest common prefix of the two input morton
   bitstrings
__device__ int _deltaFunc(uint m1, uint m2)
{
   uint tmp = m1 ^ m2; // xor

   /*int len = 0;
   // count the leading zero
   for(int k = 31; k >= 0; k--)
   {
      uint mask = 1U;
      mask <<= k;
      if((tmp&mask) == 0)// (i & mask) == (j & mask))
      {
         len++;
      }
      else
      {
         break;
      }
   }

   return len;*/
   return __clz(tmp);
}
```

Clz - CPU

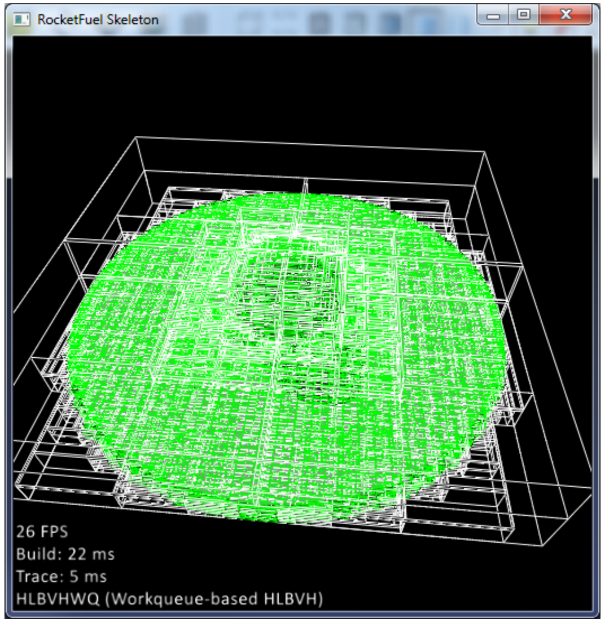
```
inline int clz( uint bit_string )
{
    __asm
    {
        MOV EAX, bit_string;
        BSR EAX, EAX;
        SUB EAX, 31;
        IMUL EAX, -1;
    }
    // Return with result in EAX
}
```

Build time

	CPU - Karras
Asm	5.5 ms
Loop	18.4 ms

SAH ?!

- Surface area heuristic taking into account the size and distribution of triangles to find split
- Right now experimenting with an iterative scheme to improve fast trees... Tree rotations



ALEXANDRA INSTITUTTET

OpenCL experience

- Optimizing for each platform
  - i.e. taking a working intel OpenCL and compiling for nvidia GPU gave a bad performance
- Difficult to make an implementation that works on all platforms
  - i.e. taking a working (optimized) nvidia OpenCL and compiling for intel gave wrong results (problem in barriers)
- We need standard algorithms for sort, prefixsum etc.

Fast ray tracing

ALEXANDRA
INSTITUTTET

Editing environment

Photonmap_realtime.mp4

ALEXANDRA
INSTITUTTET

LEGO Universe

- February 2010
- Lego Universe was in Development

Lego Universe (Oct. 2010)

ALEXANDRA INSTITUTTET

- http://www.youtube.com/watch?v=rYAuzslBg0w
- http://www.youtube.com/watch?v=rl0Xr1nscH4

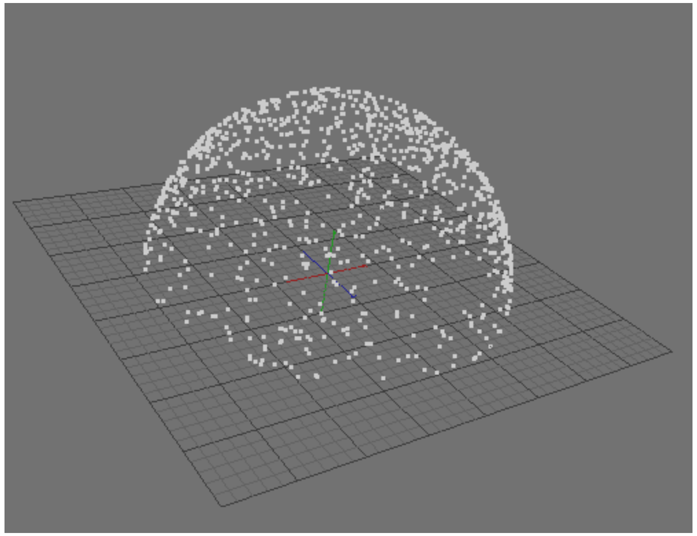
GPU Supercomputing med LEGO

- Rack-mounted Quadro Plex servers (17 in Miami)
- Model processing
  - Geometri simplifikation *(Optix)*
  - Per-vertex ambient occlusion *(Optix)*
- In game icons
  - *(OpenGL + CUDA)*
- Images for moderation
  - *(OpenGL + CUDA)*

Optix

- CUDA kernel - generate ray program (per triangle)
  - Generate samples on hemisphere sampled on triangle
- CUDA kernel - Material program
  - Write if occluded

- Max_unoccluded_for_keeping_face
  - If exceeded keep vertex
- Ambient occlusion per vertex
  - Sampler hemisphere af face-normal



Lego server geometry optimization

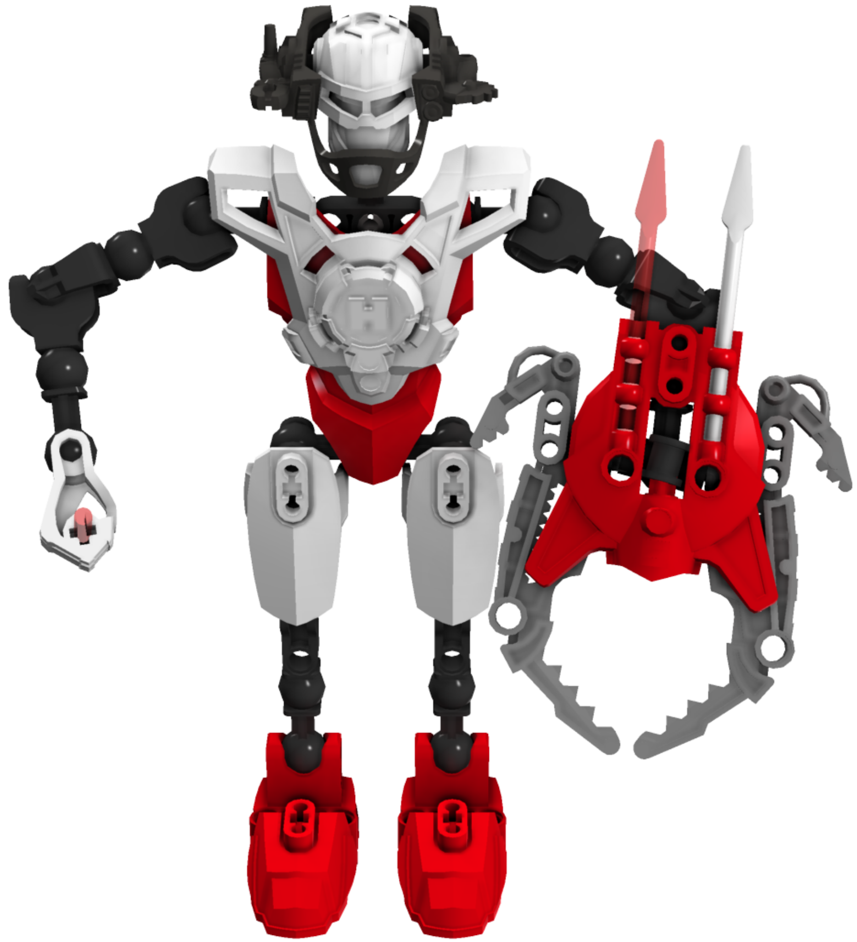
538.000 vertices

444.924 vertices

Lego Universe, Moderation

Numbers (okt. 2010 – apr. 2011)

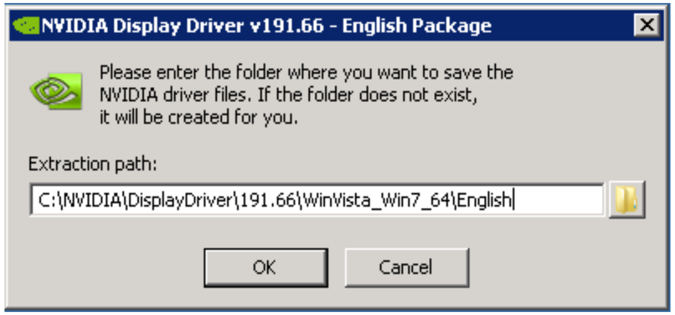
- 6.3 million dds renderinger (icons)
  - 128x128
- 11.9 million png (moderation)
  - 1024x1024
- 12.6 million geo. optimizations

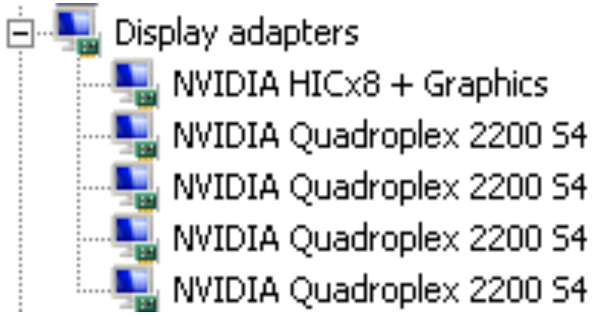


LEGO model optimized and rendered

Lego rendering

Affinity





GHIC adapter driver

```
if (!wglEnumGpusNV || !wglCreateAffinityDCNV ||
    !wglDeleteDCNV || !wglEnumGpuDevicesNV ||
    !wglEnumGpusFromAffinityDCNV)
{
    errorStrings.PushBack("Affinity not supported by graphics hardware");
    return false;
}
```

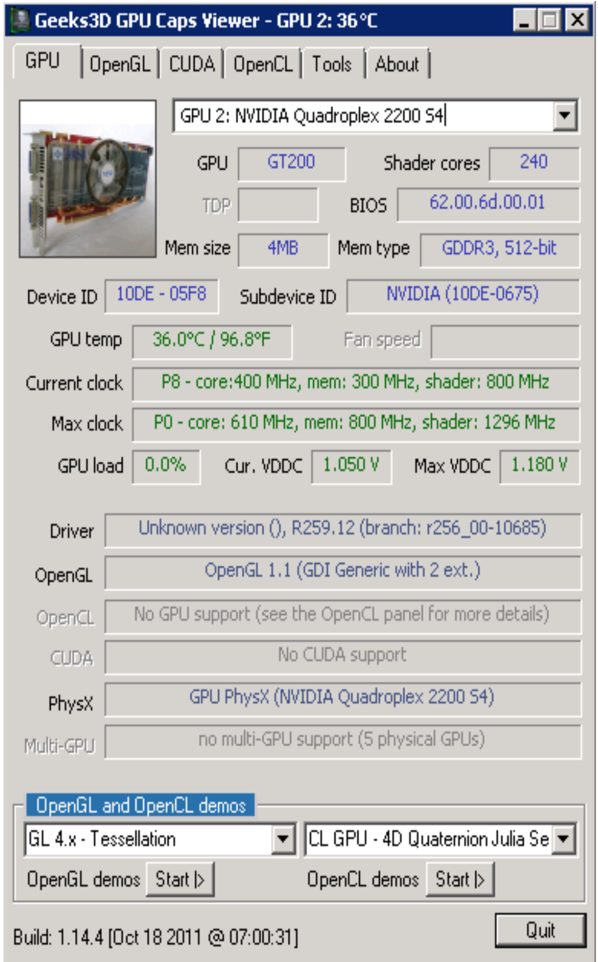
OpenGL Extension for Affinity selection was unavailable on the G-HICx8 frontend card

ALEXANDRA INSTITUTTET

Virtual Adapter and Session-0 isolation

Windows Service

Remote Desktop (Hosting / Terremark)



4 high-end GPUs OpenGL 1.1 with 2 extensions

ALEXANDRA INSTITUTTET

Porting, performance and maintanability

- Multi-core / many-core is here to stay

- Porting of code
- Performance (and target)
- Maintenance

Graphics API ?

- Still good at graphics
- Still features that are not in Cuda/OpenCL

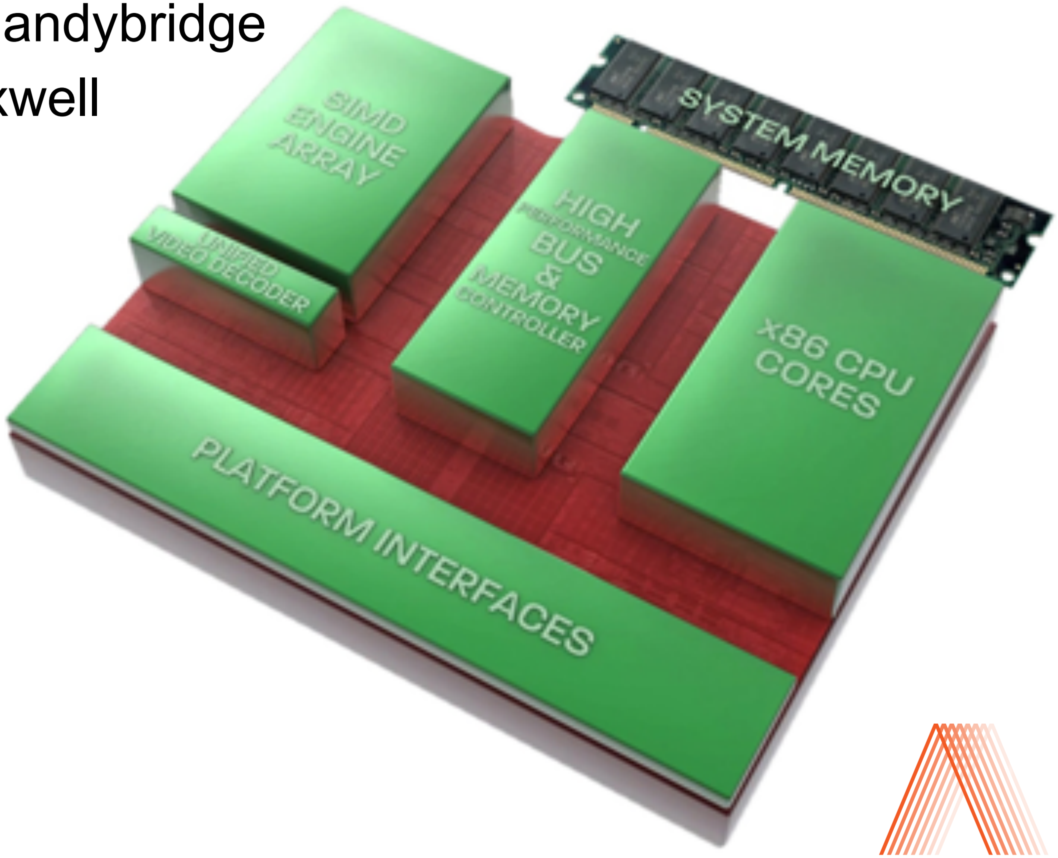
- Portable / standardised
- Compute capability
  - Direct Compute (Direct X)
  - Compute Shaders (OpenGL)

- Web
  - WebGL (OpenGL for web)
  - WebCL (OpenCL for web)

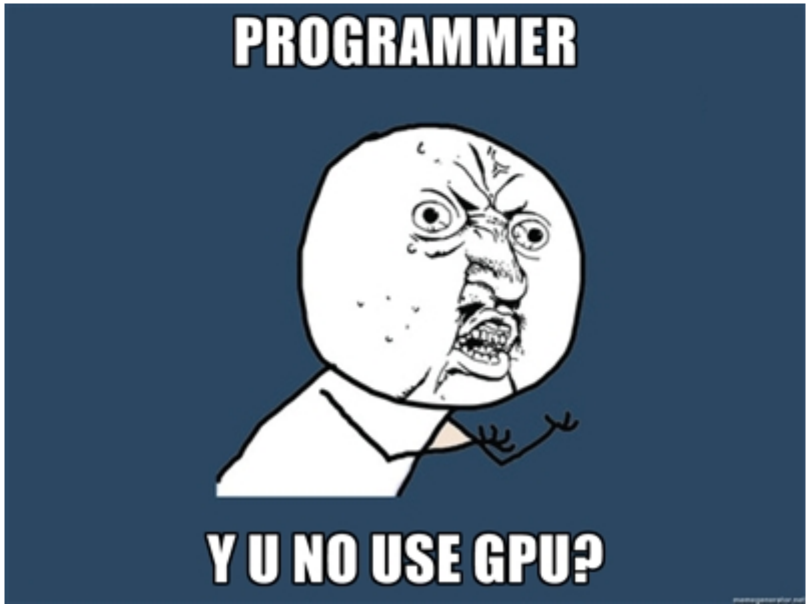
ALEXANDRA INSTITUTTET

Heterogen processering

- CPU/GPU hybrid processors
  - AMD Fusion / Llano
  - intel Larrabee / Sandybridge
  - NVIDIA Kepler / Maxwell



ALEXANDRA INSTITUTTET



ALEXANDRA
INSTITUTTET

Jesper.mosegaard@alexandra.dk

twitter.com/mosegaard

cg.alexandra.dk