

Real-Time Computer Graphics Paradigm Shift?

Dr. Bedřich Beneš
Department of Computer Graphics Technology
Purdue University, USA
bbenes@purdue.edu

Overview

- o What is Computer Graphics?
- o What are the traditional approaches?
- o What are the new trends?
- o Conclusions (puros, tequilas...)

How the CG was born

- o **the late 50s**,
computers could produce graphics,
but software is missing, software crisis
- o **in the beginning of 60s**
first Computer Aided Design (CAD)
1963 Sketchpad by Ivan Sutherland
Douglas Englebart – mouse
- o **the late 60s and beginning of the 70s**
raster graphics = revolution (Ivan Sutherland)

How the CG was born (contd.)

- o **the late 70s**
boom of computer graphics,
beginning of interaction,
first Graphical User Interfaces (GUI)
1970 Bézier curves
1971 Gouraud shading
1973 Westworld – first film with CG
1974 Ed Catmull – Z-buffer
1975 Phong shading
1976 Blinn invented bump mapping
1979 Whitted – ray tracing
- o **12th August 1981** IBM PC has been introduced

How the CG was born (contd.)

- o **the 80s**
boom of computer graphics
1986 Luxo Jr. nominated for an Academy Award
1989 Tin Toy (Pixar) wins Oscar



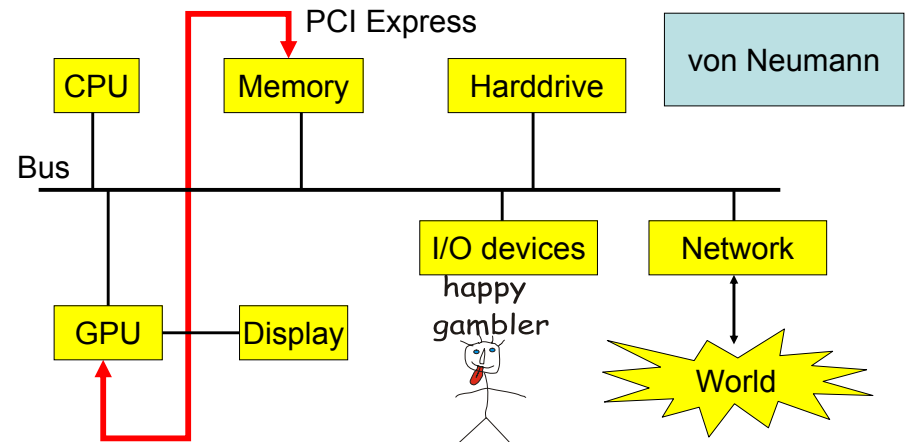
How the CG was born (contd.)

- o **1999** programmable GPU
vertex and pixel shaders
- o **2000** every computer is a parallel computer
CPU vs GPU
- o **2003** Nv30 is more complex than CPU
- o **2004** RT cinematic effects (real time Toy Story)
- o **2005** GeForce 7800 GTX
the most complex chip ever
(302M transistors)

Overview

- o What is Computer Graphics?
- o *What are the traditional approaches?*
- o What are the new trends?
- o Conclusions (puros, tequilas...)

3D CG principles



3D CG Image Creation

Virtual objects “live” inside memory of computer

How did they get there?

They have a texture

How is it done?

They are illuminated by light

How is this simulated?

and are displayed on the screen

How is it achieved?

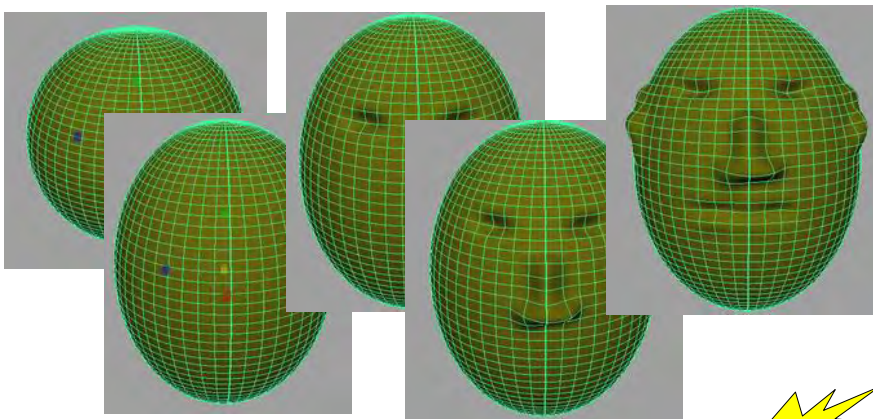
Objects

How can we get objects into the computer memory?

- 1) by hand
- 2) by capturing (3D scanners)
- 3) by a procedure

geometric modeling

Objects (contd.)



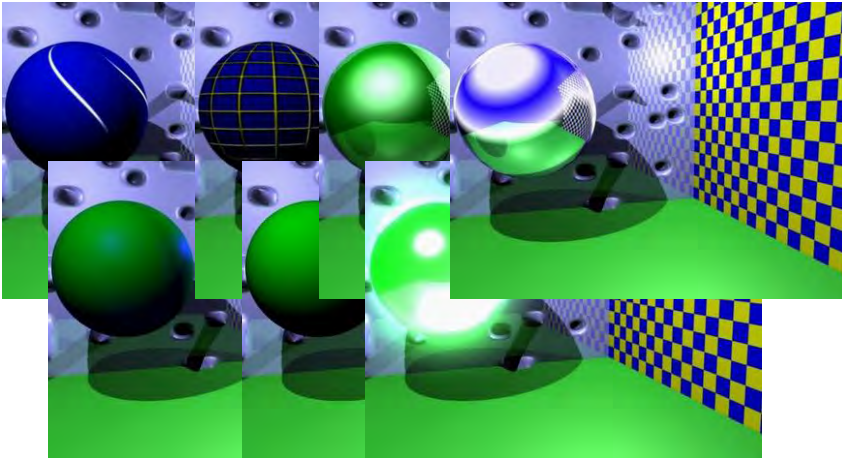
live example

Textures

- o How does the surface look like?
- o What are the properties of material?
- o How does a surface react to the light?
- o In what direction and what part of the spectra is it reflecting?
- o Is it fuzzy?
- o Is the surface bumped like metal?

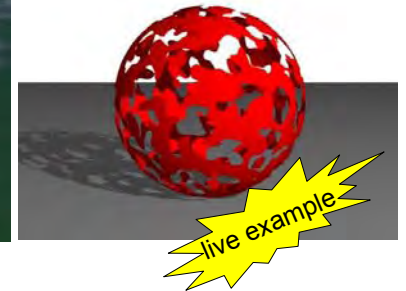
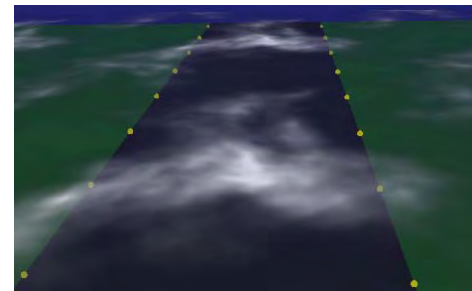
This is called texturing

Textures (contd.)



Textures (contd.)

- o Texturing is a very strong tool
- o ...especially transparency



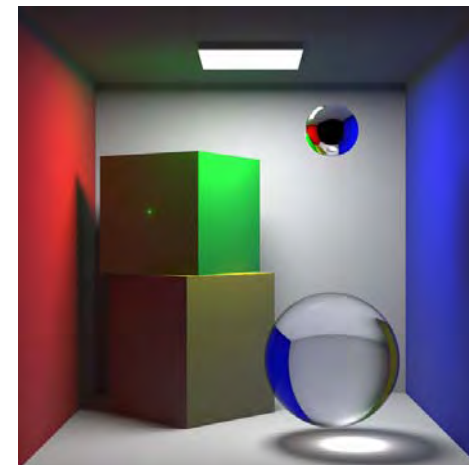
Light

Textures and light properties are the most important for seeing the objects

Lights cast shadows, causes reflection of objects, caustics, reacts with particles of dust and drops of water in the air etc.

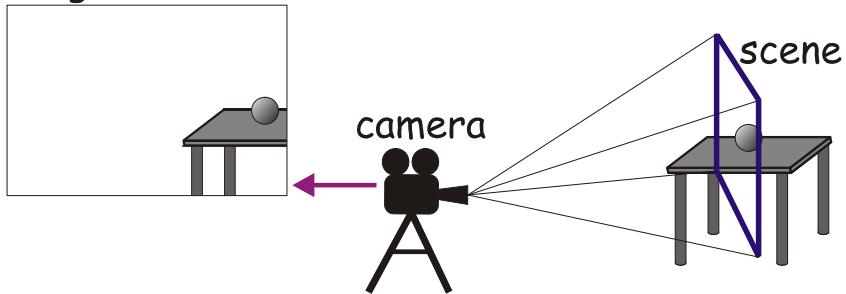
illumination model or lighting

Light (contd.)



Camera

A very special object is the camera
image



through the camera we see inside the scene

Summary

How the image is created?

- 1) Put objects into the memory
- 2) Put textures on their surfaces
- 3) Assign lights
- 4) Position camera(s)
- 5) Run the illumination algorithm

Overview

- o What is Computer Graphics?
- o What are the traditional approaches?
- o *What are the new trends?*
- o Conclusions (puros, tequilas...)

The New Trends

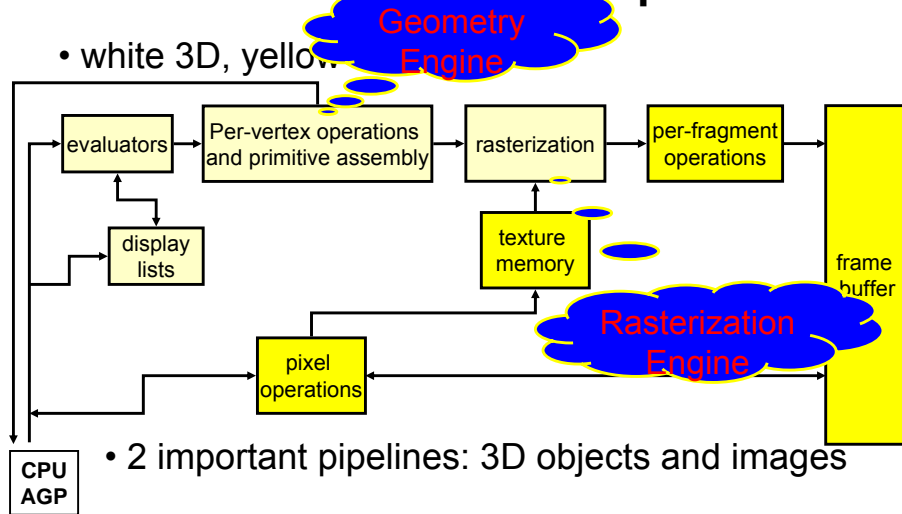
Each Computer is a Parallel Machine

CPU is (traditionally) driven by programs

GPU can be *programmed too!*

Let's have a look inside the GPU

The traditional CPU pipeline



The traditional GPU pipeline

Why pipeline?

It is the pipelining architecture known from CPU

Each operational unit can process chunk of data *in parallel* with another one

The *slowest* and most *demanding* is the rasterization unit

In some GPUs this unit is more times

Traditional GPU architecture

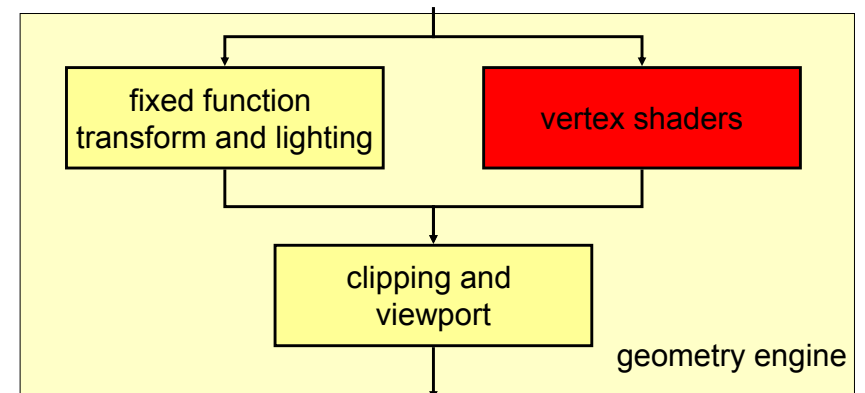
the fixed function pipeline

- o data is *sent* into the OpenGL
- o OpenGL has internal states (“turn on the light 2”)
- o The data is processed according to the states
- o Very low variability
- o Good, but limited

GPU architecture

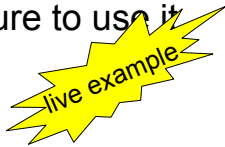
- the fixed function pipeline is obsolete since 1999
- in these days

Per vertex primitive and assembly

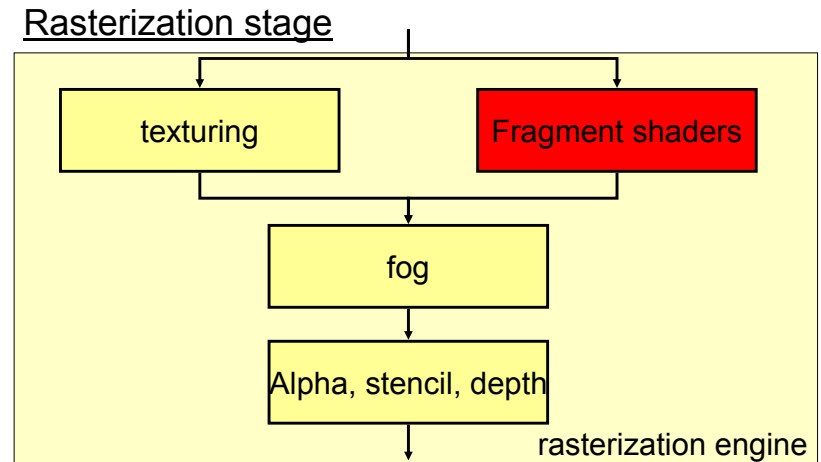


Vertex Shader

- o vertex shader is a small program
- o it is *first sent* to the GPU
- o *vertices* are processed according to the program
- o it is similar to the assembly language
- o but we *must know* the GPU architecture to use it



GPU architecture



Fragment Shader

- o fragment shader is a small program
- o it is *first sent* to the GPU
- o *pixels* are processed according to the program
- o we can do morphing, warping, etc.

Fragment Shader

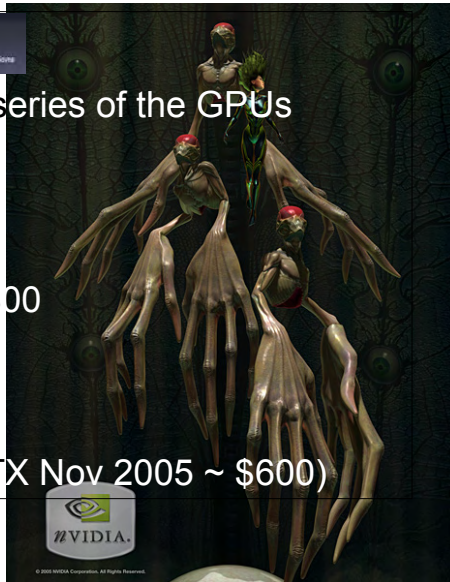
Many traditional approaches can be converted to graphics problems

- o Matrix algebra can be done on the GPU!
- o String searching (DNA cracking)
- o Cellular automata
- o New scientific libraries use the GPU (!)



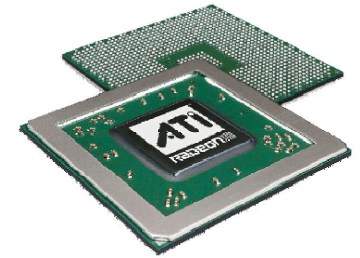
The Leaders

- o NVIDIA
- o NVidia has the GeForce series of the GPUs
- o GeForce, GeForce 2,
- o GeForce 3, GeForce 4,
- o GeForce FX, GeForce 6800
- o GeForce 7800 GTX
- o the last one
(ASUS GeForce 7800 GTX Nov 2005 ~ \$600)



The Leaders

- o ATI Technologies
- o Radeon Series
- o Radeon X1800 Pro



Overview

- o What is Computer Graphics?
- o What are the traditional approaches?
- o What are the new trends?
- o **Conclusions (puros, tequilas...)**

Conclusions

- o There is a *silent revolution* in the computer hardware
- o Many demanding things are moved on the GPU
- o Can we use it for our purposes?
- o What should we do?
 - o learn CG (new languages to program GPUs)
 - o learn OpenGL Shading Language, D3D...
 - o visit www.nvidia.com and www.ati.com

Future?

- o Predictions are difficult, especially of the future
- o GHz hunt – the time is over...
- o 8GHz CPU should have temperature of a core of a nuclear reactor
- o There is no way to cool it!
- o We need to go parallel
- o Multi-core CPUs

Conclusions

They say...

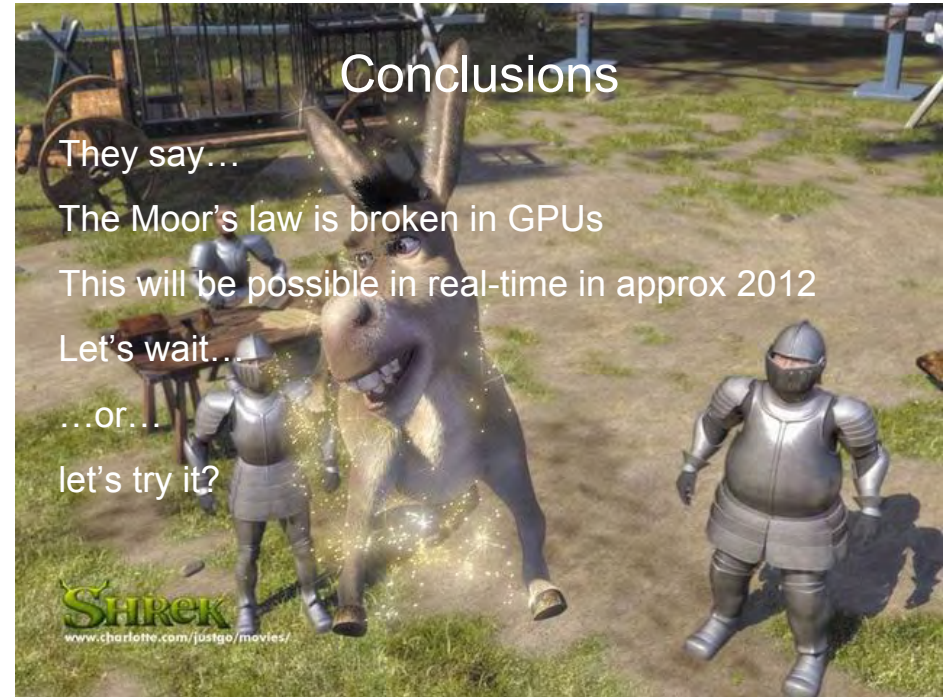
The Moor's law is broken in GPUs

This will be possible in real-time in approx 2012

Let's wait...

...or...

let's try it?



Real-Time Computer Graphics Paradigm Shift?

Dr. Bedřich Beneš
Department of Computer Graphics Technology
Purdue University, USA
bbenes@purdue.edu