Real-time Graphics

0. Introduction

Martin Samuelčík

Room I-4, samuelcik@fmph.uniba.sk,

http://www.sccg.sk/~samuelcik

Motivation

- Rendering – visualization of 3D scene, geometry + material + effects
- Real-time – 60 frames per second, maintain constant rate
- Close approximation of reality
- Usage: games, games, games, scientific visualizations, interactive presentations
- Inclusion in web browsers, cell phones, …

Real-time Graphics
Martin Samuelčík

Motivation

Demonstrations & project
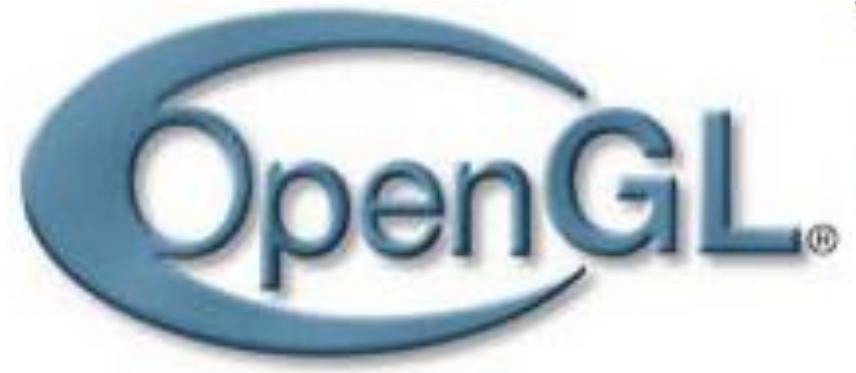
C++

GLSL (OpenGL Shading Language)

Prerequisites

- Linear algebra, geometry
- Computer graphics
- Programming language – C, C++, C#, Java, Python, …
- OpenGL course
- Willing to learn something new and exciting
- Lots of time

Lecture plan

- Graphics pipeline, VBO, FBO, GLSL
- Animation
- Shading, texturing
- Global illumination, shadows
- Reflections, refractions
- Optimalization, culling techniques, collision detection, LODs, curves, terrains
- Post-processing, image based rendering
- GPGPU, raytracing
- Volume rendering
- Non-photorealistic rendering

Real-time Graphics
Martin Samuelčík

Graphics pipeline

- Based on architecture of graphics cards
- Processing of geometry
- Input = geometry and its properties
- Output = pixels
- OpenGL = API for setting pipeline parts and inserting geometry
- Fixed parts, programmable parts

Graphics pipeline
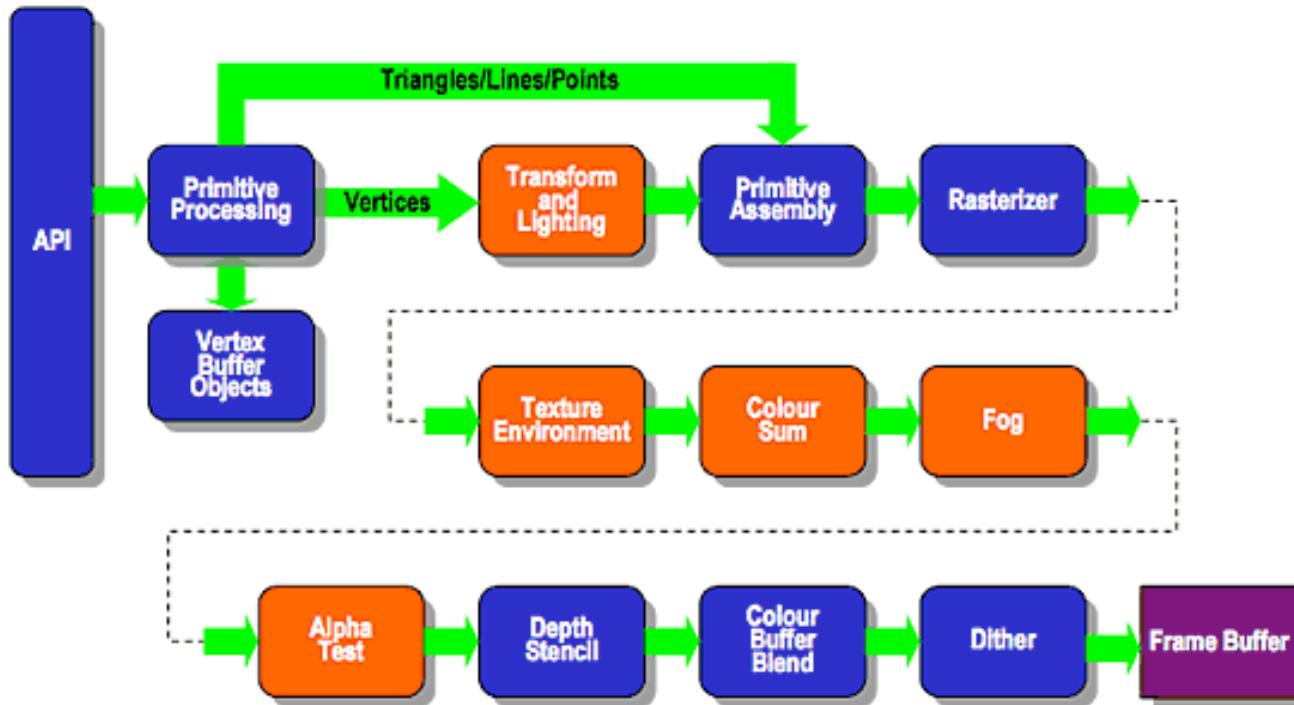
[www.khronos.org]

Graphics pipeline
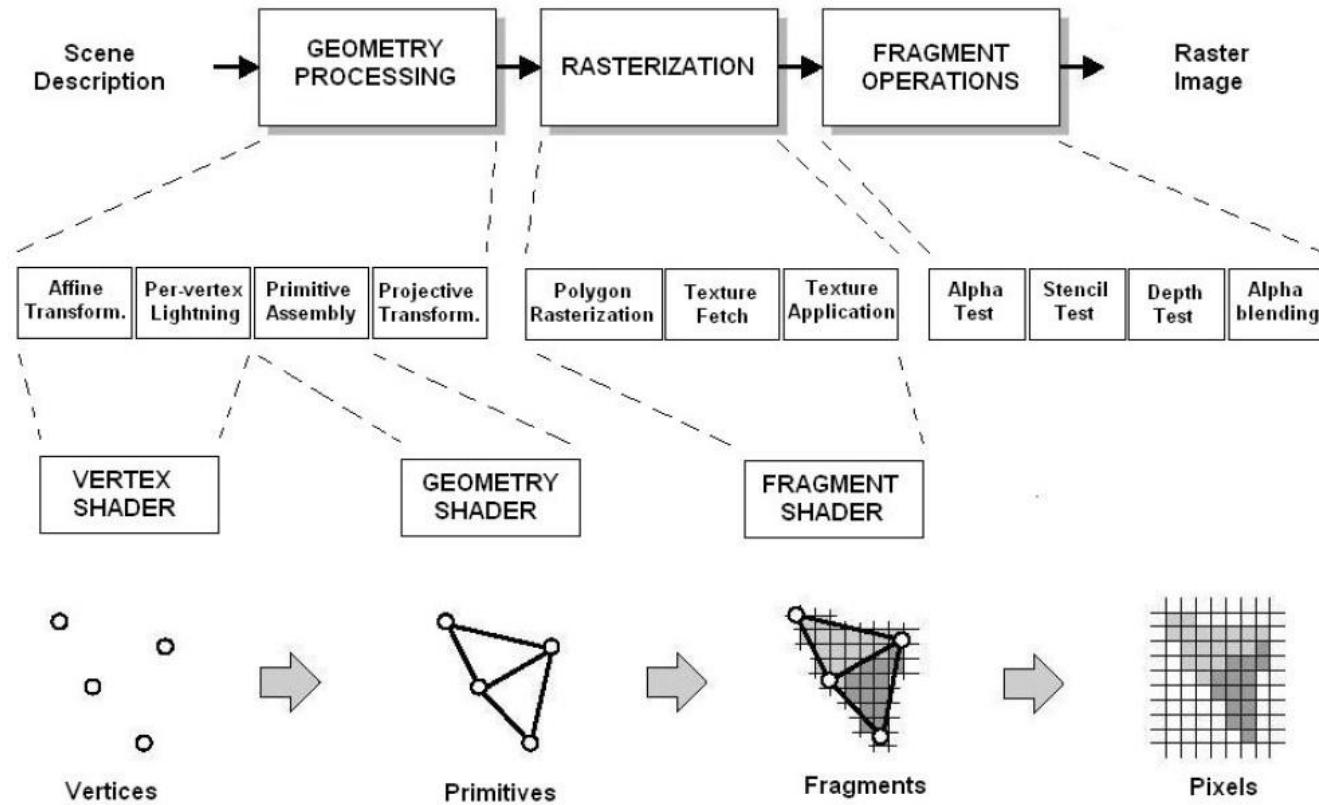
[www.wikipedia.org]

Shading, textures

- Improving visual quality

Shadows

[NVIDIA]

[ID software]

**Real-time Graphics**
Martin Samuelčík

Global Illumination

[Crytek, www.wikipedia.org]

Reflections

[Naughty Dog, Sony]
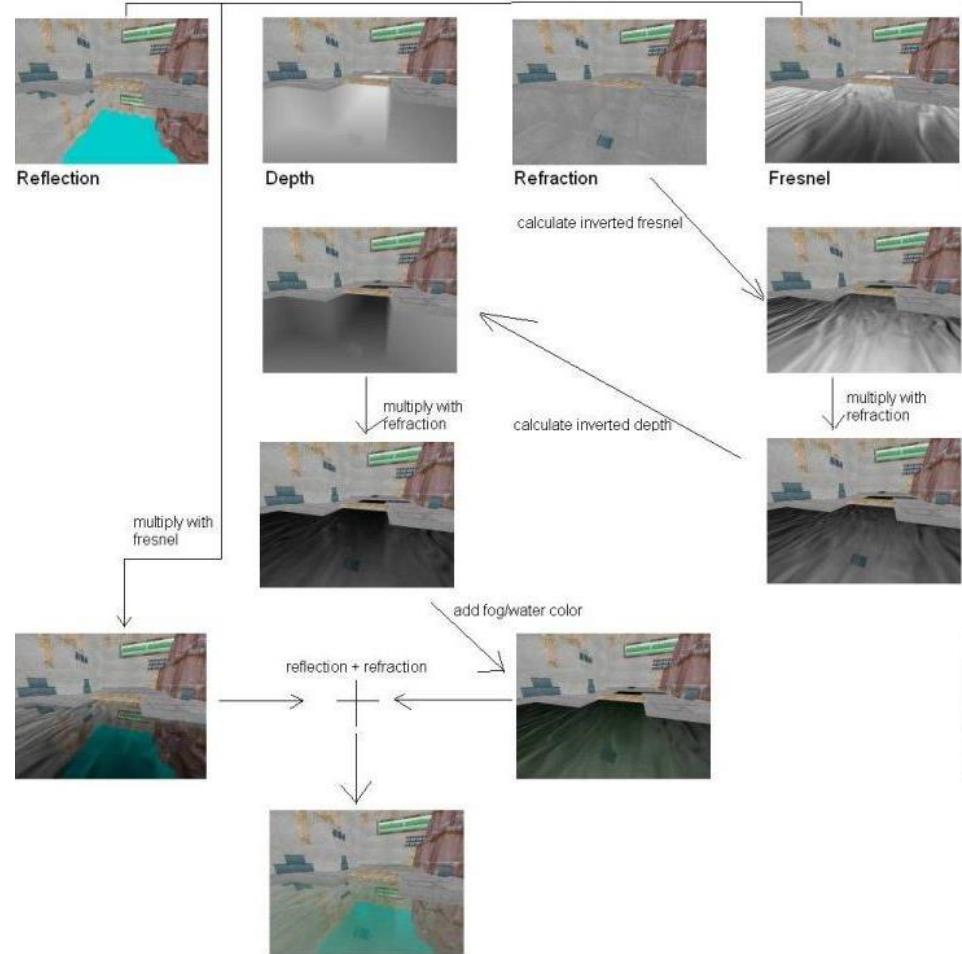
[www.bonzaisoftware.com]

Terrain
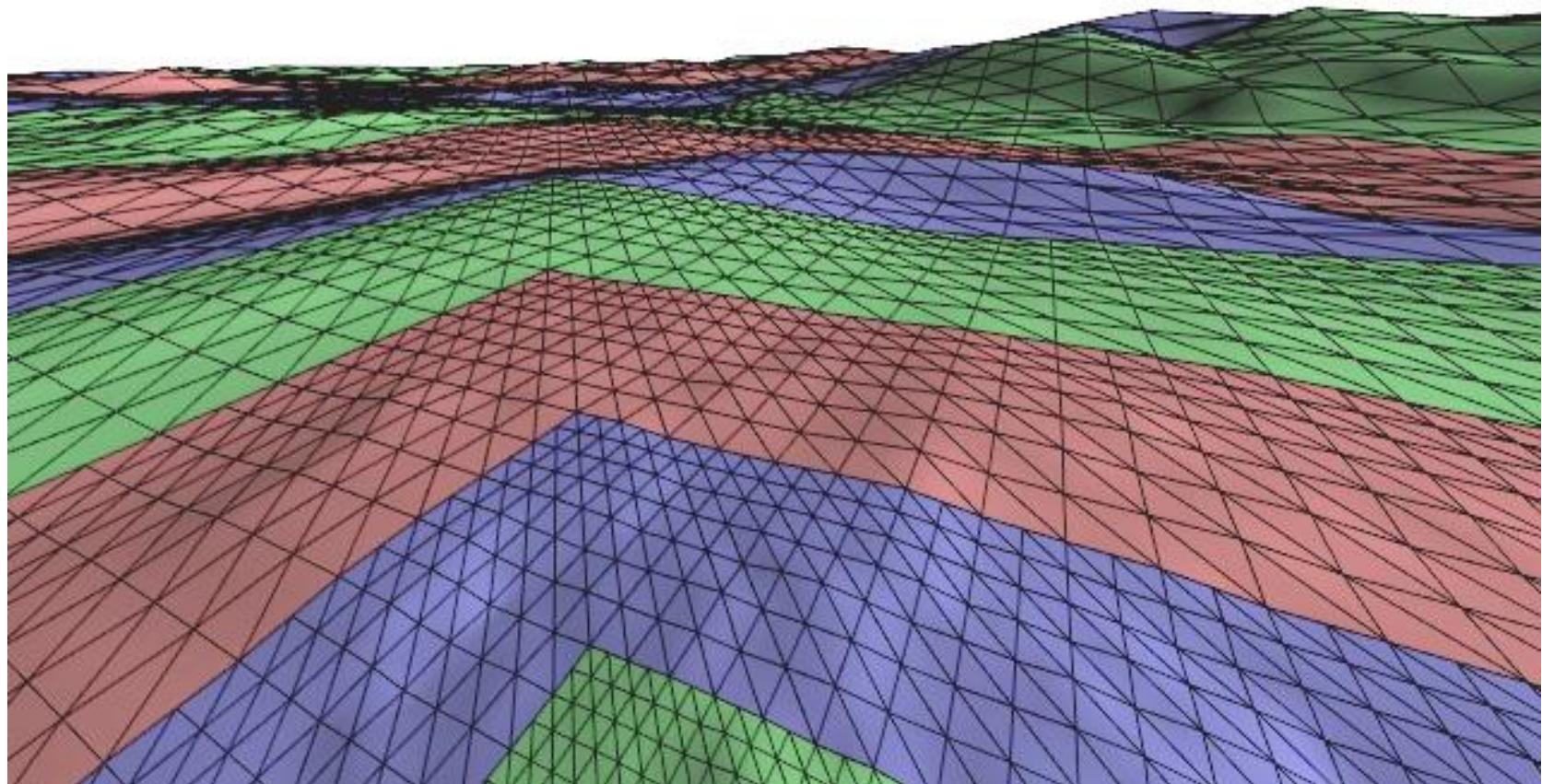
[Lossaso, Hoppe: Geometry Clipmaps]

Real-time Graphics
Martin Samuelčík

Post-processing
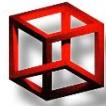
Real-time Graphics
Martin Samuelčík

Image-based rendering


[NVIDIA]

**Real-time Graphics**
Martin Samuelčík

16

GPGPU, raytracing

- Parallelization, CUDA, OpenCL
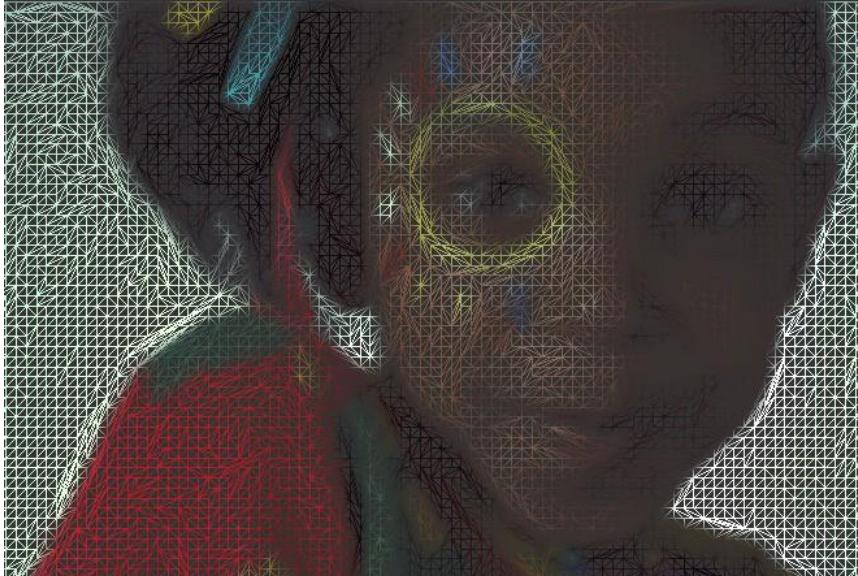
Non-photorealistic rendering
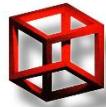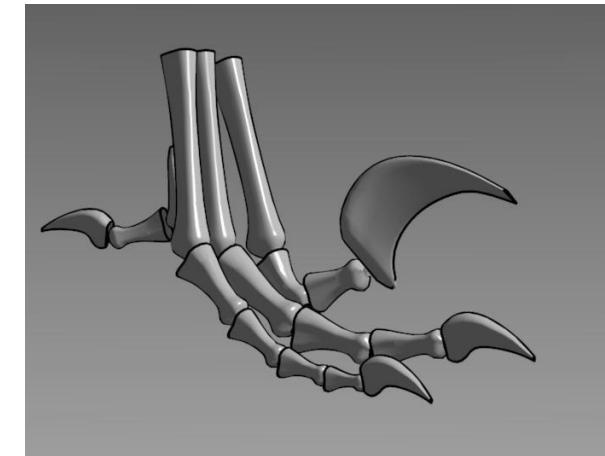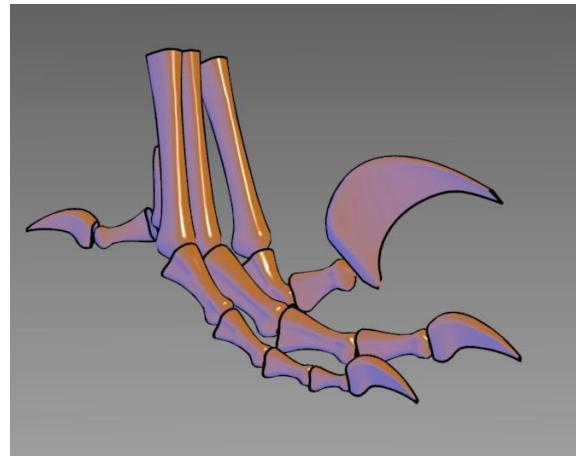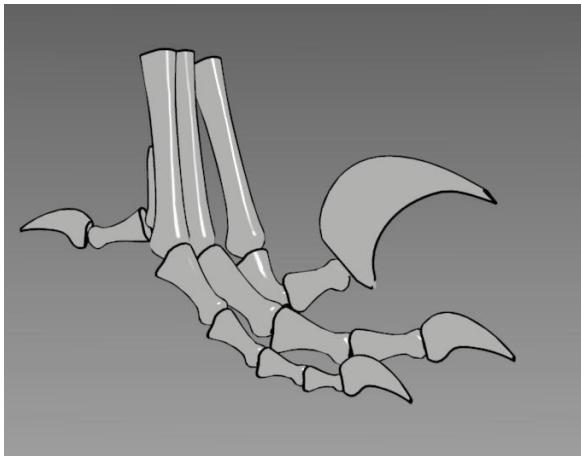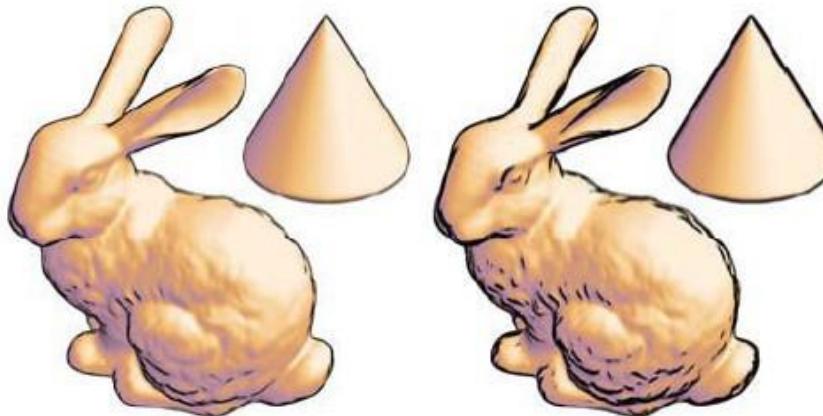
Volume rendering

- Discrete data, usually 3D grid
- Several visualization methods

Real-time Graphics
Martin Samuelčík
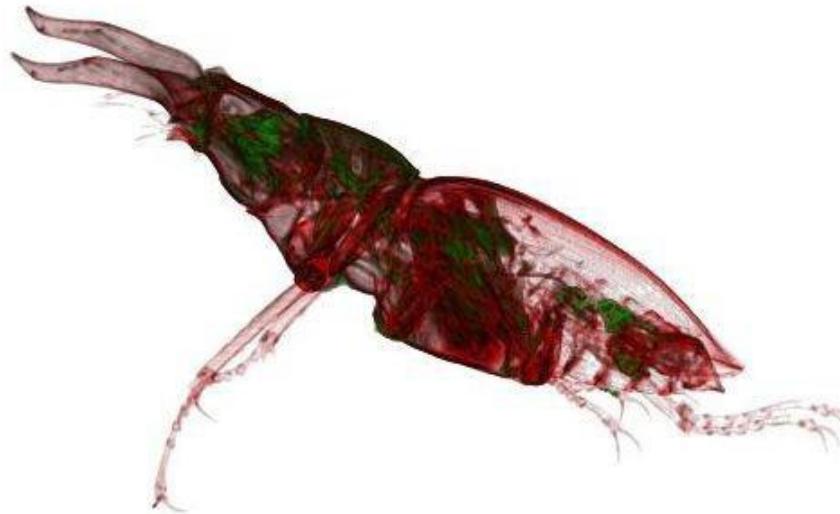
Project

- Project is demo program that uses OpenGL and GLSL for visualization of scene
- Necessary conditions:
  - Loading of scene and all objects from COLLADA or .3DS file
  - At least 4 animated objects, including necessary animated camera and 1 light, animation control: play, pause, stop, all animations loaded from external files
  - All objects should be textured and rendered using shaders
  - At least 3 light sources (point + directional)
  - 2 basic per-pixel lighting methods (phong, phong+normal mapping), switched in real-time
  - At least 3 different shader programs (vertex+fragment shader)
  - Rendering to texture, Shadows
  - Video showing most important and interesting parts and effects of demo
  - Presenting using working Release configuration

Real-time Graphics
Martin Samuelčík

Project

- Pick 3 additional packages of effects:
  - Using geometry shader for generating subdivision surfaces
  - Displacement mapping, Terrain rendering with LOD (on/off)
  - Depth of field, Motion blur (on/off)
  - Screen space ambient occlusion, 2 options (on/off)
  - HDR rendering of sun, Lens flare, Bloom effects (on/off)
  - Parallax, bump, relief mapping (on/off)
  - GPGPU (after consultation)
  - Reflection and refraction on water surface (on/off)
  - Particle system for waterfall and fire visualization, particles update and rendered using shaders
  - Volume rendering of clouds, volumetric effects (smoke, fog, light volumes)
  - Toon, cell shading, Oren-Nayar & Cook-Torrance per-pixel lighting (on/off)

Real-time Graphics
Martin Samuelčík

Project dates

- 31.3.2015: Specification of project – detailed description of platform, content, animations, effects, milestones
- 3.7.2015: Final project
- Rating:
  - Project: 70% - everything on time, complexity, fulfilled conditions
  - Exam: 30% - while presenting project, questions about functionality, algorithms, structures

Videos

Literature

- OpenGL Programming Guide
- OpenGL Shading Language
- Real-Time Rendering
- GPU Gems I,II,III
- Shader X1,X2,…
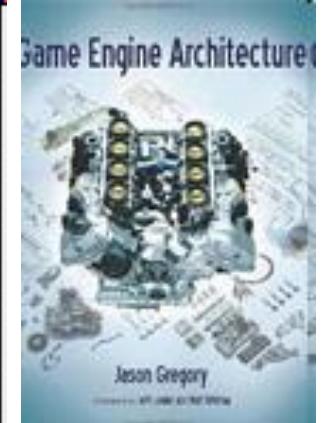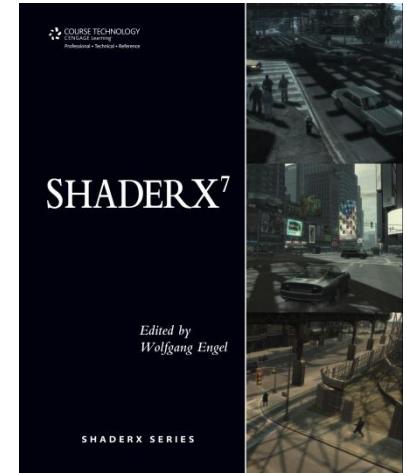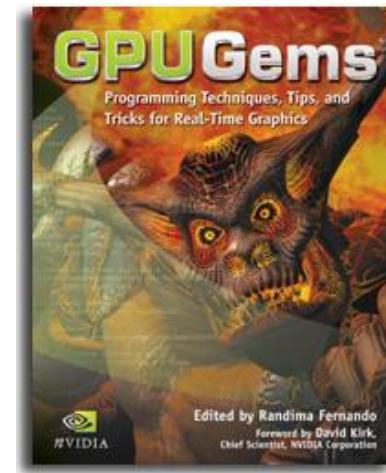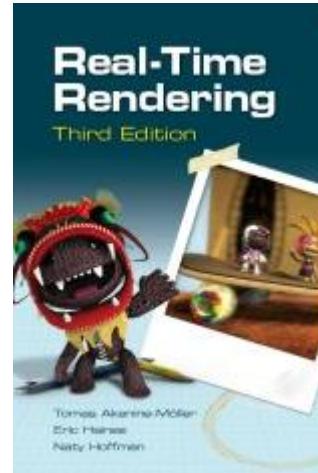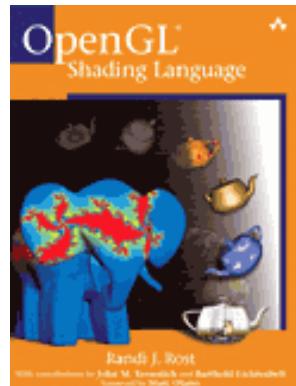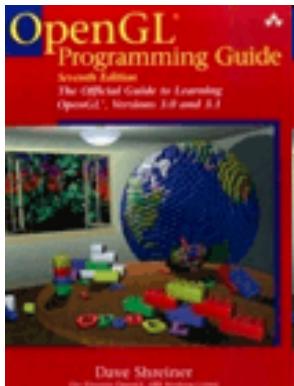
Real-time Graphics
Martin Samuelčík

Libraries

- **GLUT**
  - Managing OpenGL windows, input events
  - http://www.opengl.org/resources/libraries/glut/

- **GLEW**
  - Managing OpenGL extensions
  - http://glew.sourceforge.net/

- **Assimp**
  - Managing and loading models from external files
  - http://assimp.sourceforge.net/

Real-time Graphics
Martin Samuelčík
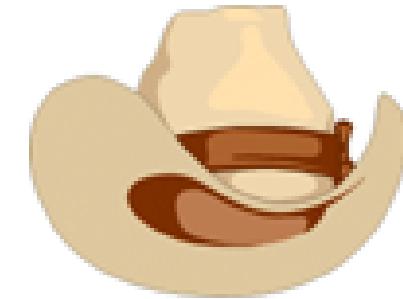
Libraries

- **DevIL**
  - Loading images from external files
  - http://openil.sourceforge.net/

- **GLM**
  - Mathematics routines for OpenGL
  - http://glm.g-truc.net/0.9.5/index.htm

- **FreeType**
  - Managing TrueType fonts
  - http://www.freetype.org/

Real-time Graphics
Martin Samuelčík

Web

- http://nehe.gamedev.net
- http://www.opengl.org, http://www.opengl.org/sdk/
- http://www.cg.tuwien.ac.at/courses/Realtime/
- http://www.cs.virginia.edu/~gfx/Courses/2004/RealTime
- http://developer.nvidia.com/object/gpu_gems_home.html
- http://developer.nvidia.com/object/sdk_home.html
- http://developer.amd.com/gpu/radeon/pages/default.aspx
- http://tog.acm.org/resources/shaderx/
- http://fly.cc.fer.hr/~unreal/theredbook/
- http://www.openscenegraph.org
- http://www.ogre3d.org/
- http://www.google.com

Questions?