Jimmy Johansson

Advanced Computer Graphics Programming (TNCG14)

Course home page:
http://staffwww.itn.liu.se/~jimjo/courses/TNCG14-2009/
E-mail: jimmy.johansson@itn.liu.se

Linköpings universitet

People

- Jimmy Johansson (JJ)
  - PhD in Information Visualization
  - Research/teaching: Information visualization, data mining, OpenGL, Shader programming, ...
- Matt Cooper (MC)
  - PhD in computational chemistry
  - Research/teaching: VR-technology, parallel programming, data mining, computer graphics, ...
- Karljohan Palmerius (KP)
  - PhD in Scientific visualization
  - Research/teaching: Haptics, realtime rendering, VR-technology, ...

Aim of the course

- “...is to develop an understanding of and ability to work with the conditions associated with high performance and real-time computer graphics”
- “..and the wide array of methods used to reach their goals”

Aim of the course

- demonstrate an understanding of the goals and requirements of high performance and real time rendering
- demonstrate the ability to decompose graphics algorithms into suitable parallel and serial components for implementation on parallel computer systems and graphics hardware
- Develop high performance graphics applications using:
  - advanced algorithms for high performance graphics
  - parallel and multi-threaded computing systems
  - programmable graphics processing units

Organization

- Lectures, labs and project work
- Strongly oriented towards self-study
- Significant amount of distributed litterature
- Course Literature:
  - Real-Time Rendering
  - The OpenGL Programming Guide - The Redbook
  - OpenGL Shading Language – The Orange book

Organization

- 6 Lectures

1. Introduction/OpenGL (JJ)
2. OpenGL and GPU programming (JJ)
3. GPU programming (JJ)
4. Parallel Architectures for CG (MC)
5. Parallel Programming for CG (MC)
6. Geometric and illumination “tricks” (KJ)

Organization

- 3 Labs
  - OpenGL
  - Shader programming using GLSL
  - Parallel Programming

- Available on the course webpage
- Individual or in pairs
- SP5117 (linux lab) is available
- Sign up on lists outside the lab
- Lab assistant is Ruman Zakaria

Organization

- Project work
  - Individually or in pairs
  - Suggested topics on the course homepage
    - Many ideas from the course book
    - Send email with suggestion/s to Jimmy no later than Thursday, April 2
  - Course grade will be based on the project work
  - Implementation
  - Public presentation by the group
    - May 28, 13-17 or May 29, 08-12
  - Written report. June 5

Why this course?

- Increase in CPU speed is not as large anymore
- Size of data still rapidly increasing
  - Increasing complexity in calculations
- Multi-core
- Powerful graphics processors

Real-time rendering

- Computer graphics with a time constraint
  - “Realtime” is relative
  - Computer games, interactive visualization
  - Limited resources, time limits and other demands
- Concepts:
  - do as much as you can!
  - don't do more than you can!

Issue 1

- Do as much as you can!

Do as much as you can

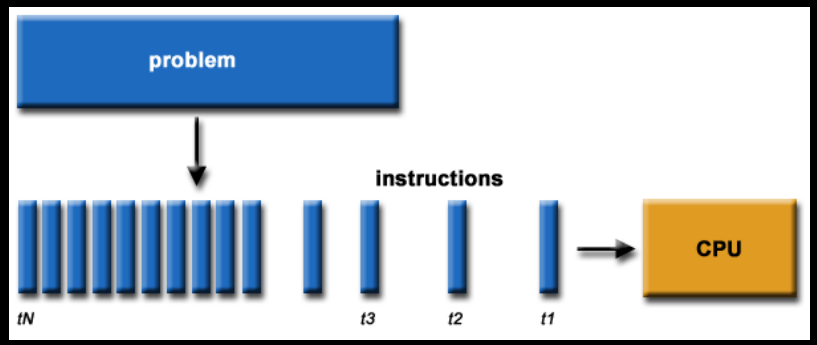
- Acceleration
  - Software acceleration
    - Code optimization
    - Parallelization
  - Hardware acceleration
    - Dedicated hardware
    - Hardware optimized graphics

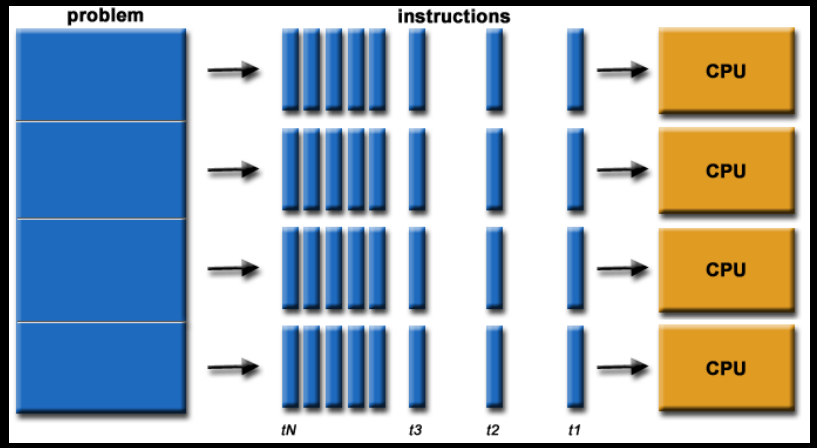
Software acceleration

- Code optimization
  - Keep the cost low
  - Low-level optimization
    - LUT or replace intensive functions
      - Pow, arctan, sincos
- Parallelization
  - Independent operations in parallel

Software acceleration

- Parallel computing
  - run using multiple CPUs
  - problem is broken into discrete parts that can be solved concurrently
  - Each part is further broken down to a series of instructions



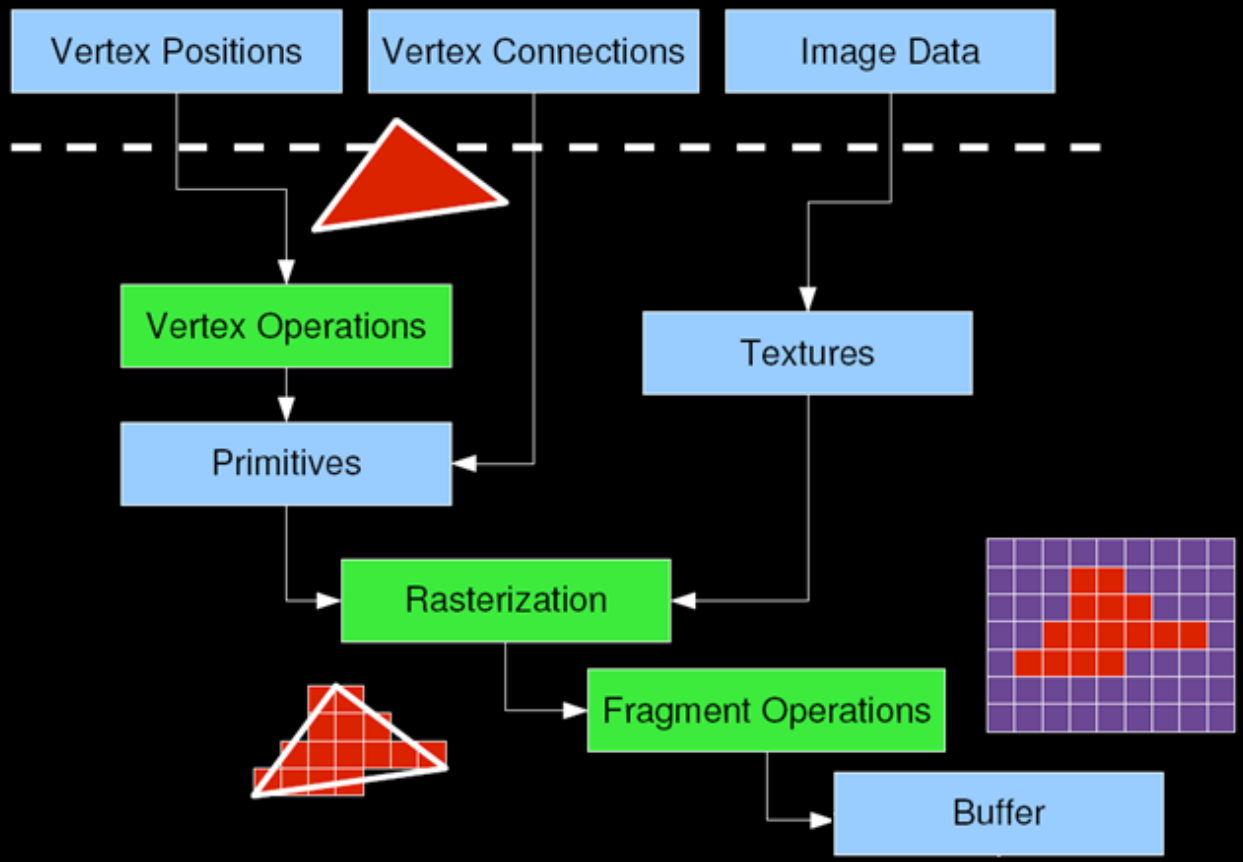


Hardware acceleration

- Hardware optimized algorithms
  - simple operations in parallel pipelines

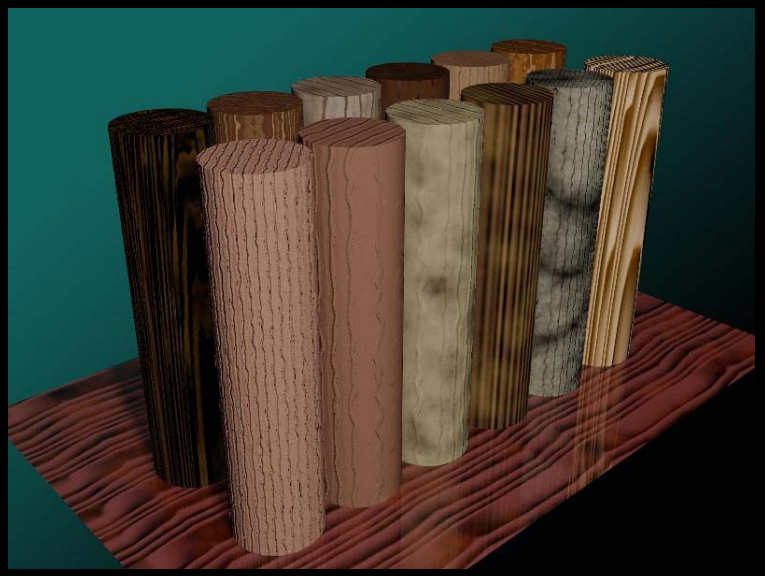
- Specialized hardware
  - GPU, PPU
  - Hardware interface
    - API: OpenGL, Direct3D, PhysX, …

Graphics Hardware



Shader Programming

- Replacing Fixed Functionality
  - Vertex shaders
  - Geometry shaders
  - Fragment shaders
- Examples include
  - Realistic materials
    - Stone, wood
  - Realistic lighting effects
    - Soft shadowing, area lights
  - Image processing (GPGPU)
    - Convolution, complex blending



Issue 2

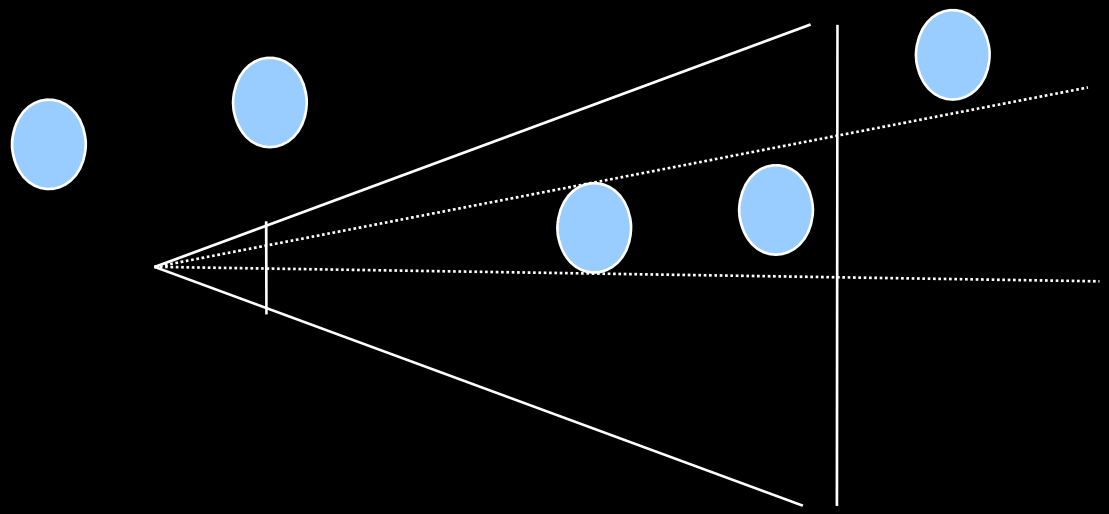
- Don’t do more than you can!

Don’t do more than you can

- Frame-rate maintenance
  - Remove unnecessary work
  - Flexibility
    - Adjust rendering to fit conditions
    - Graphics hardware, memory, CPU
  - Degrading
    - Use graceful degradation

Culling

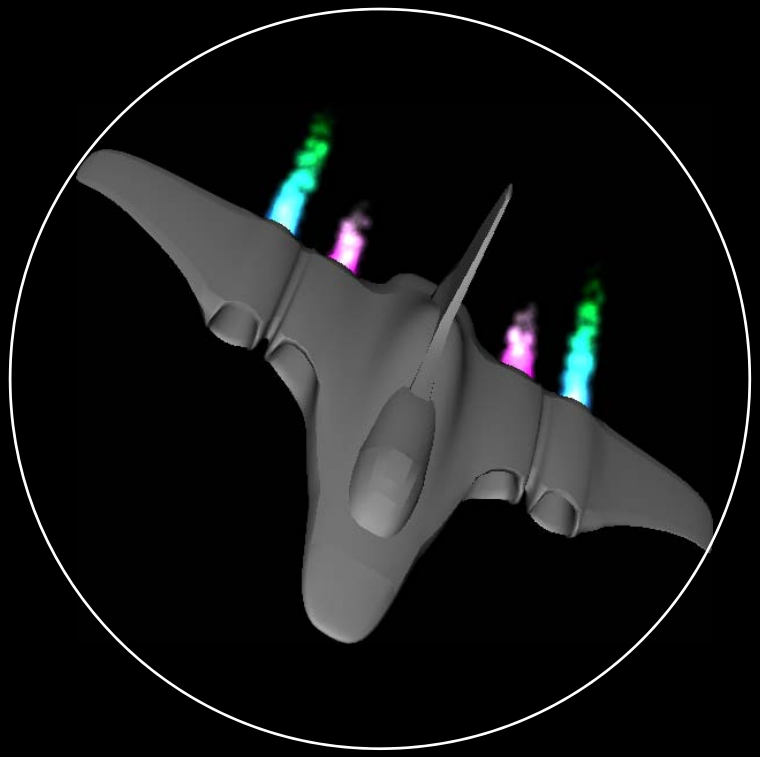
- Remove objects that will not affect the scene
  - Behind the viewer (near/far plane)
  - Outside view frustum
  - Behind another object

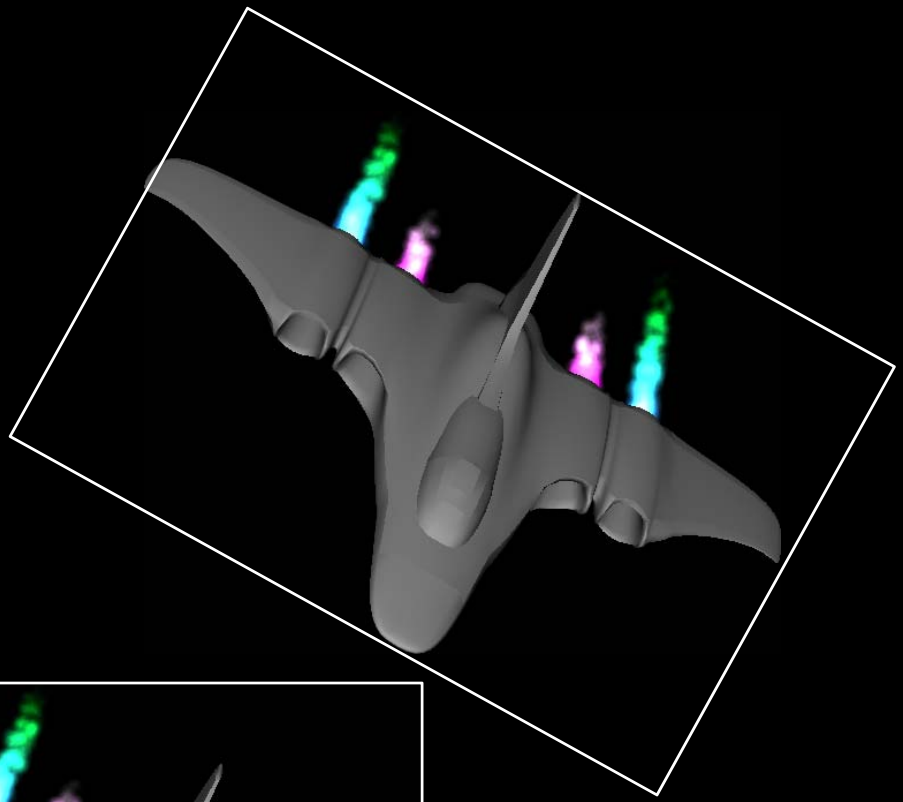


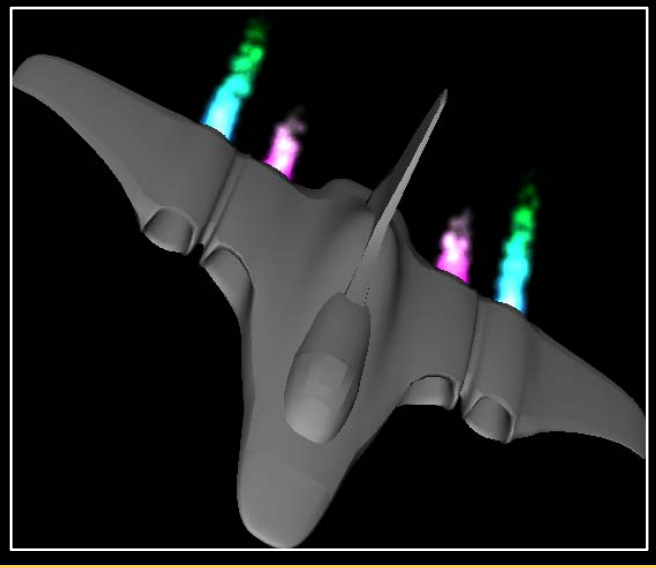
Culling

- Bounding volumes
  - Cull groups with bounding spaces outside the view
  - Bounding
    - Sphere
    - axis-aligned bounding box (AABB)
    - oriented bounding box (OBB)
  - should be easy to work with
  - should have small void space

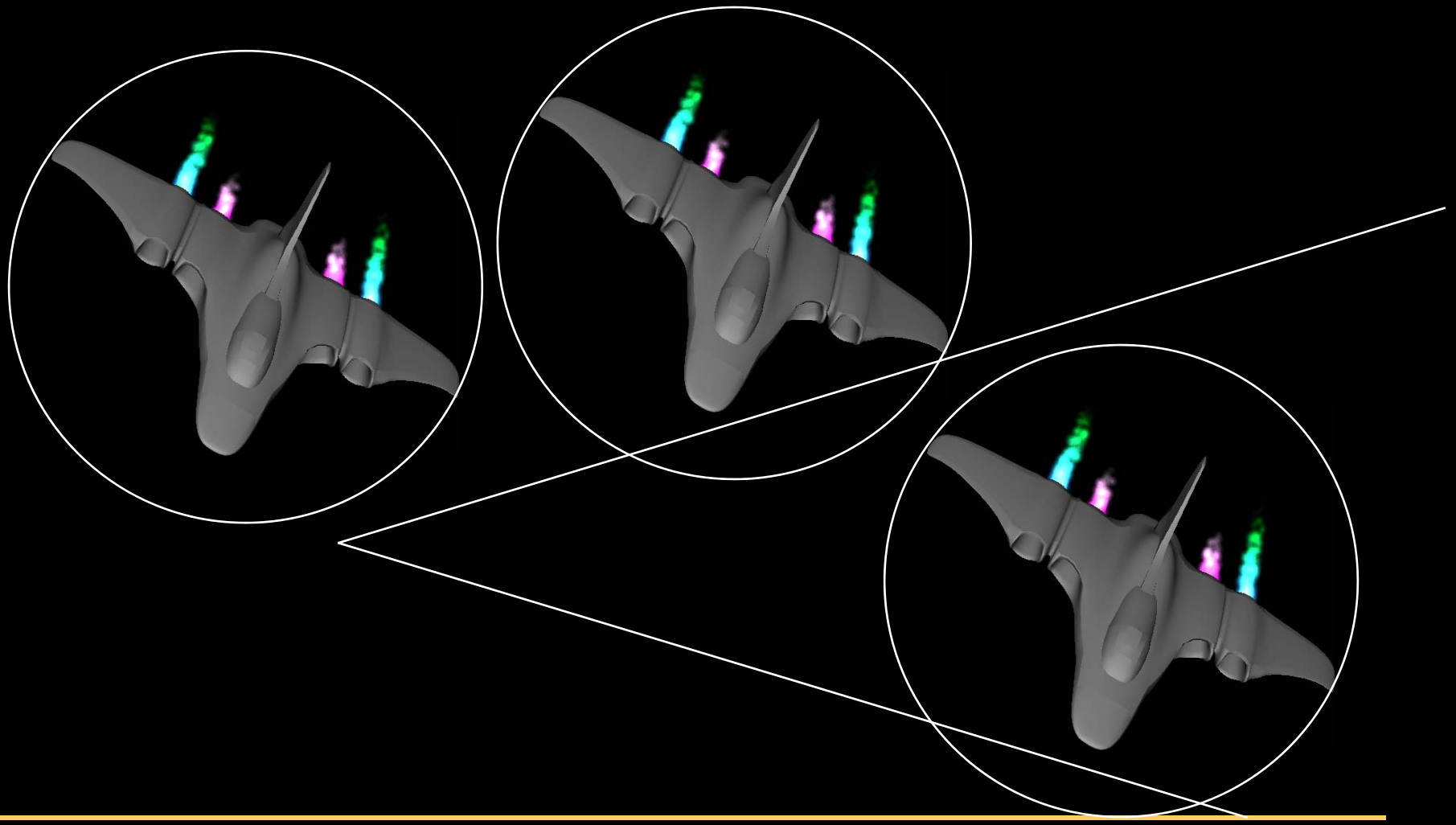
Bounding volumes





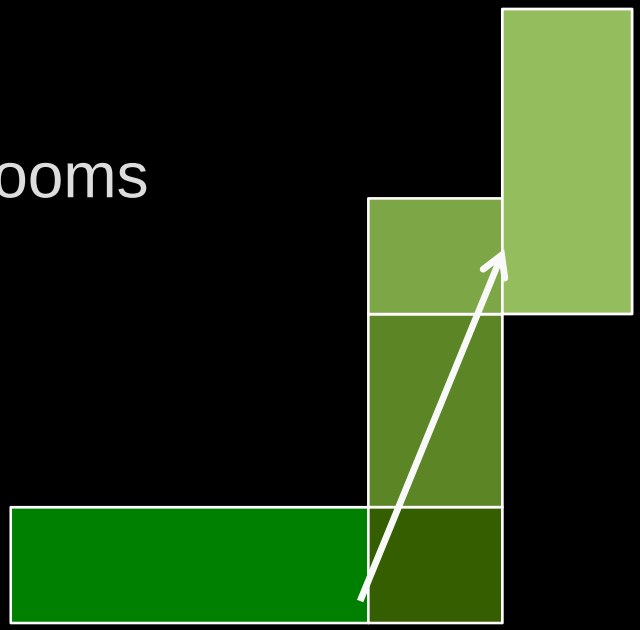


Bounding volumes



Scene Partitioning

- Example: rooms
  - Limited content
  - Occluding walls
  - Static relation to other rooms
- Intelligent door placement
  - Automatic closing



Degrading

- Graceful degradation
  - In the event of a reduction of available resources (time, CPU, memory,...) good choices are made to make the best possible use of what is available without failing to complete the originally planned task

Degrading

- Graceful degradation

Degrading

- Graceful degradation

Degrading

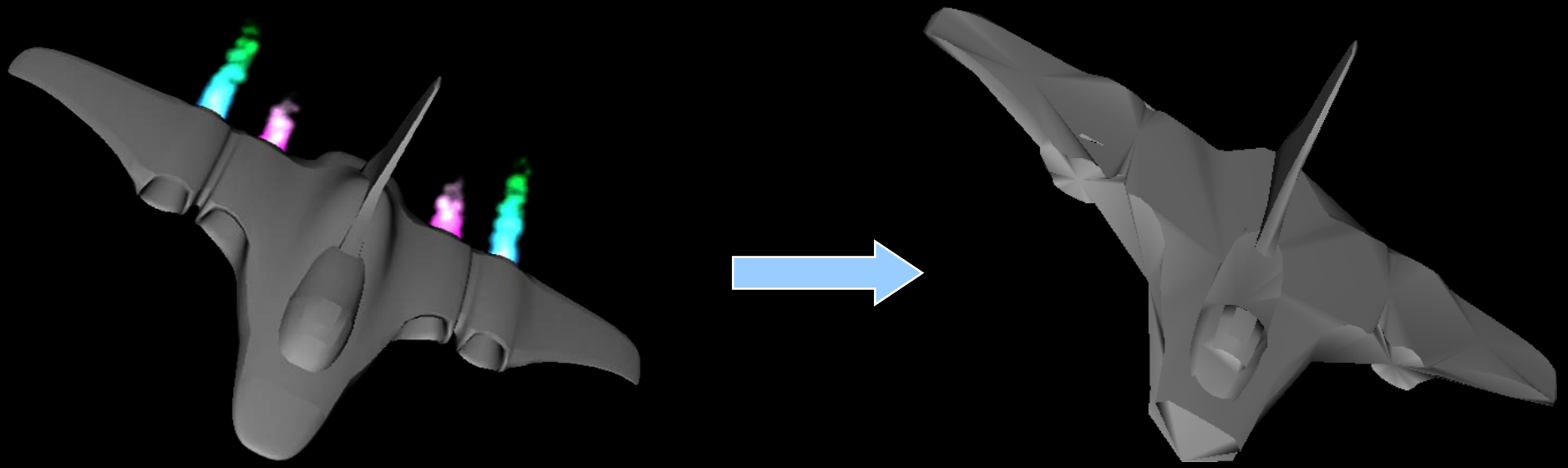
- Graceful degradation

Cheating

- Don't use more effort than necessary
  - Psychophysical aspects (perception)
    - speed, contrast, (eccentricity)
    - fog, (depth-of-field)

Object Resolution

- Level-of-detail (LoD)
  - Use as low resolution as possible
  - Select resolution at render-time



Textures

- Replace geometrical details with texture (Image)
  - Clothes, faces and ground

Textures

- Use photographs of things
  - Lighting
  - Material and colour
  - Microstructure

Fragment Shaders

- Hardware accelerated details
  - Each fragment individually estimated
  - Real-time updated texturing
  - Bumpiness, details, etc.
  - Relief texturing

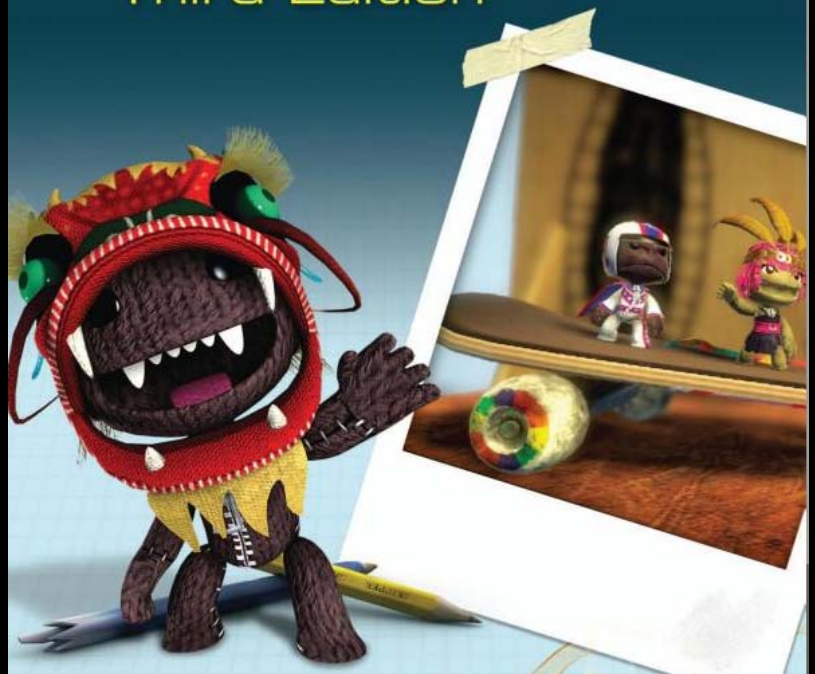
Summary Introduction

- Do as much as you can!
- Don't do more than you can!

- Many techniques and “tricks” to achieve this

Real-Time Rendering

Third Edition



Tomas Akenine-Möller
Eric Haines
Naty Hoffman

Contents

Copyrighted Material