
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



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 literature
- 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



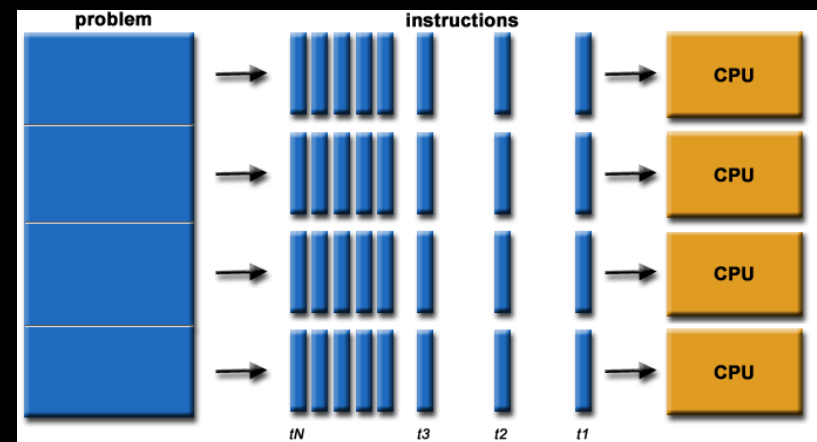
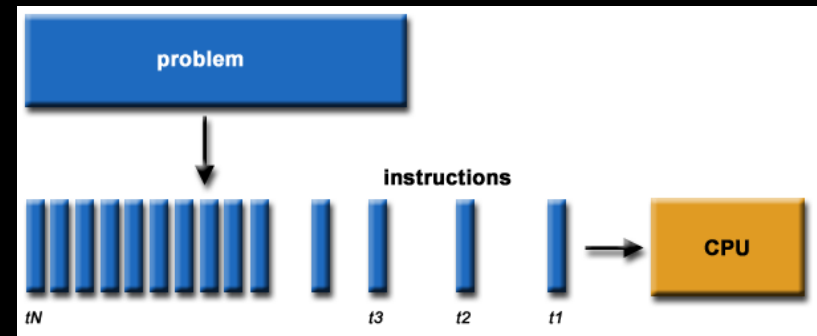
Software acceleration

- 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

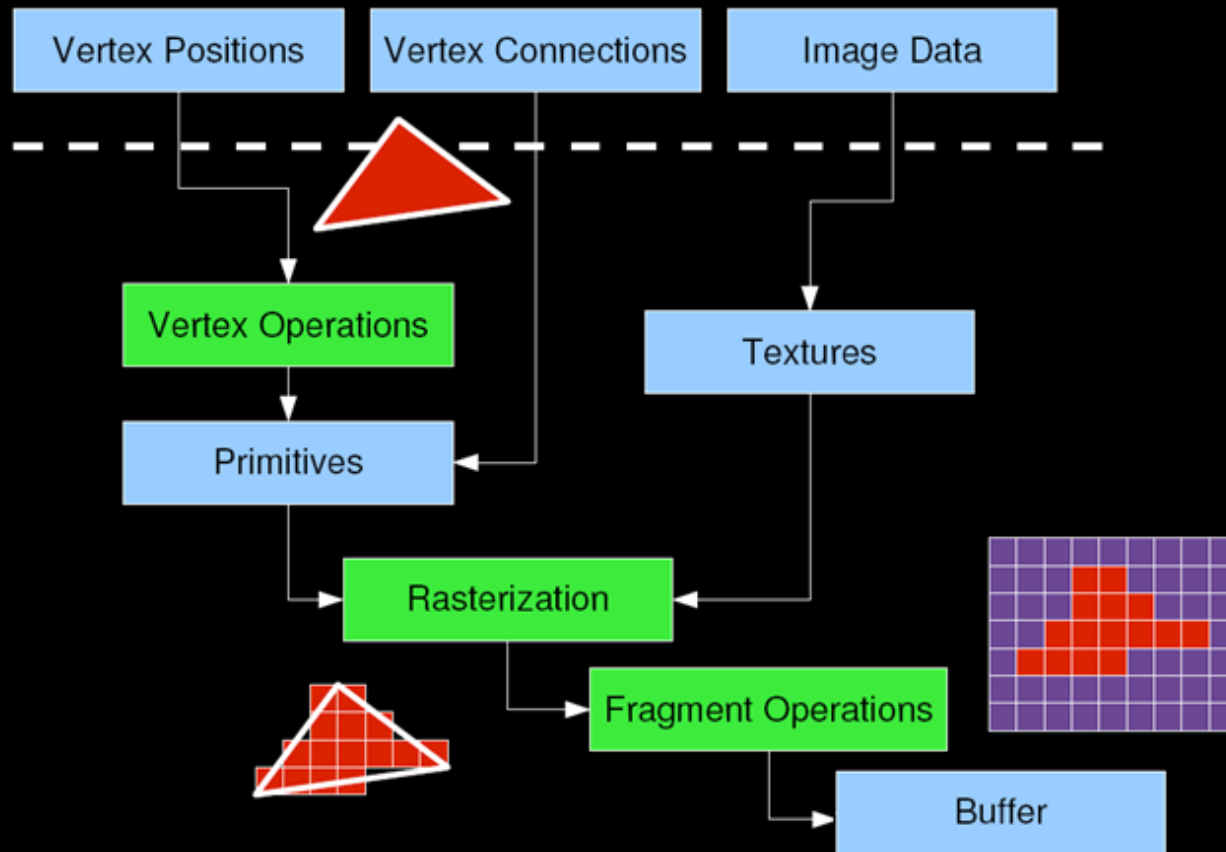


Hardware acceleration

- Hardware optimized algorithms
 - simple operations in parallel pipelines
- Specialized hardware
 - GPU, PPU
 - Hardware interface
 - API: OpenGL, Direct3D, PhysX, ...

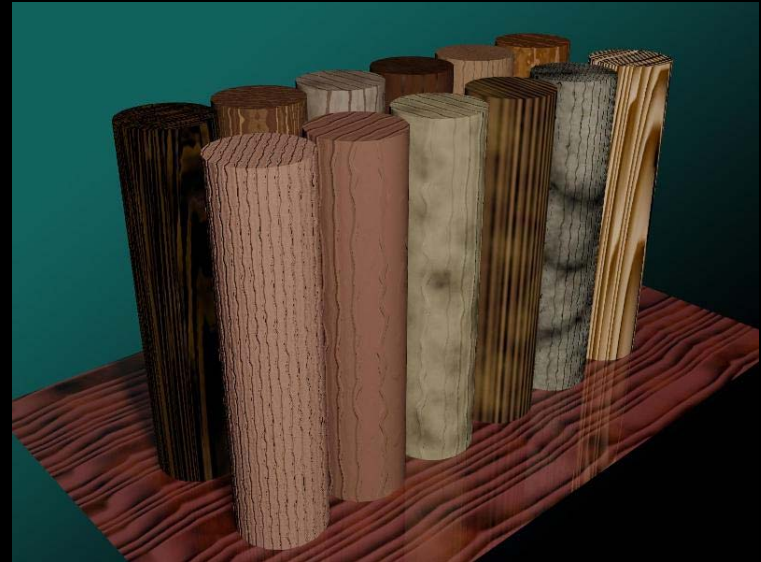


Graphics Hardware



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!



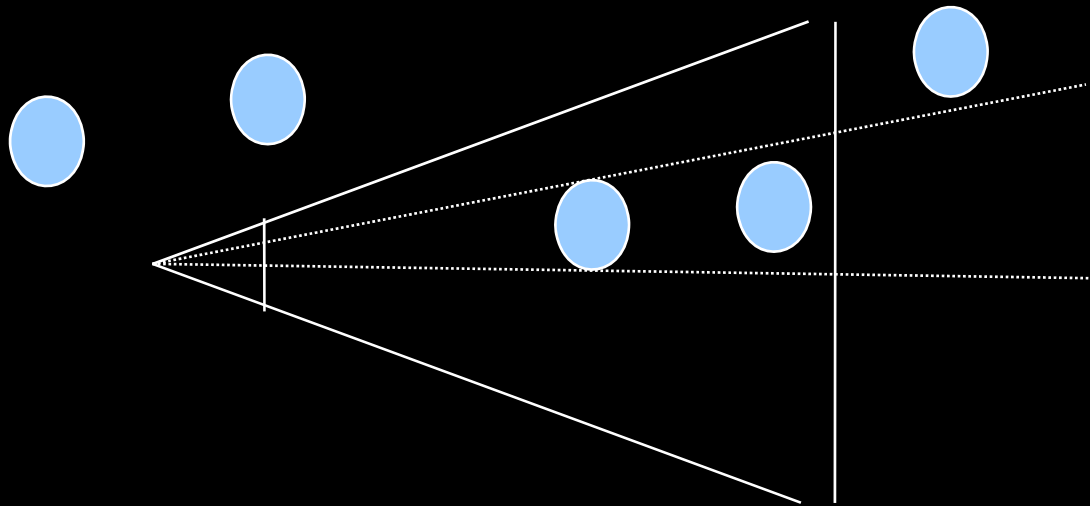
Don't do more than you can

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



Culling

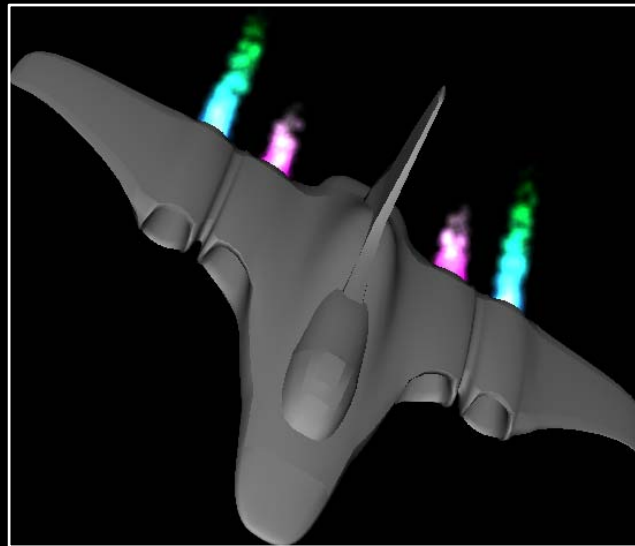
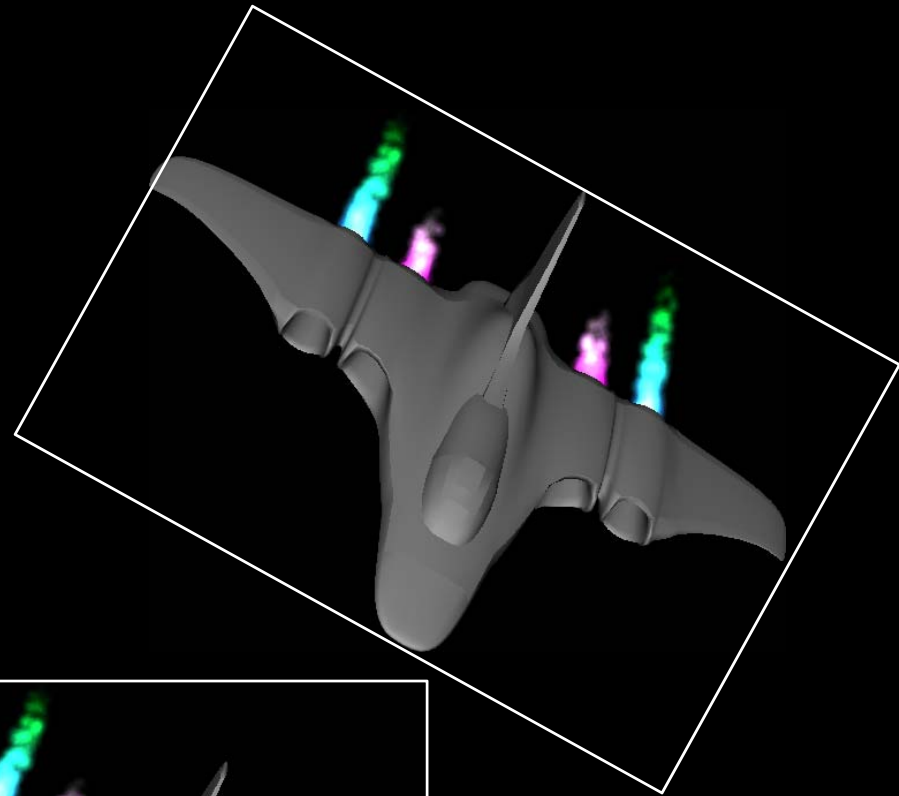
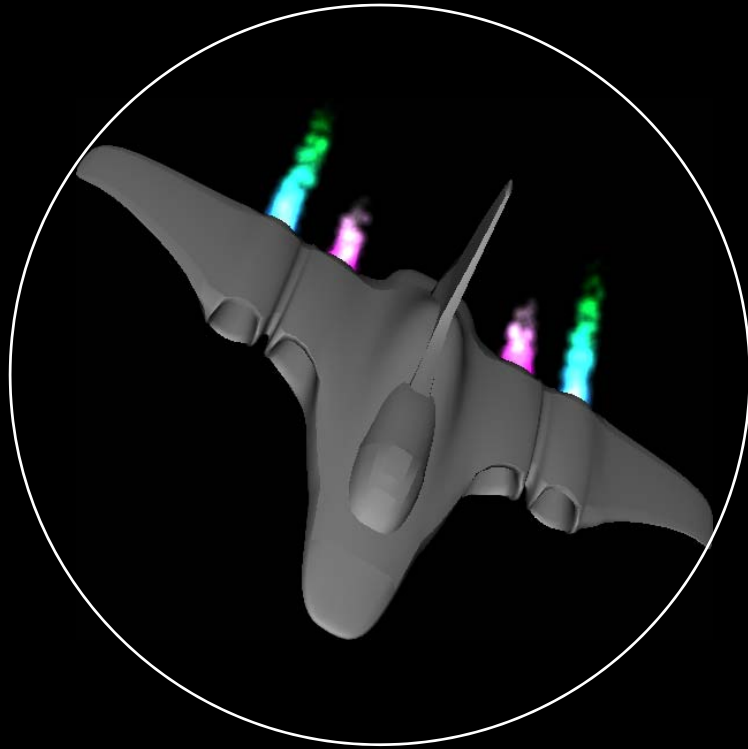
- Remove objects that will not affect the scene
 - Behind the viewer (near/far plane)
 - Outside view frustum
 - Behind another object



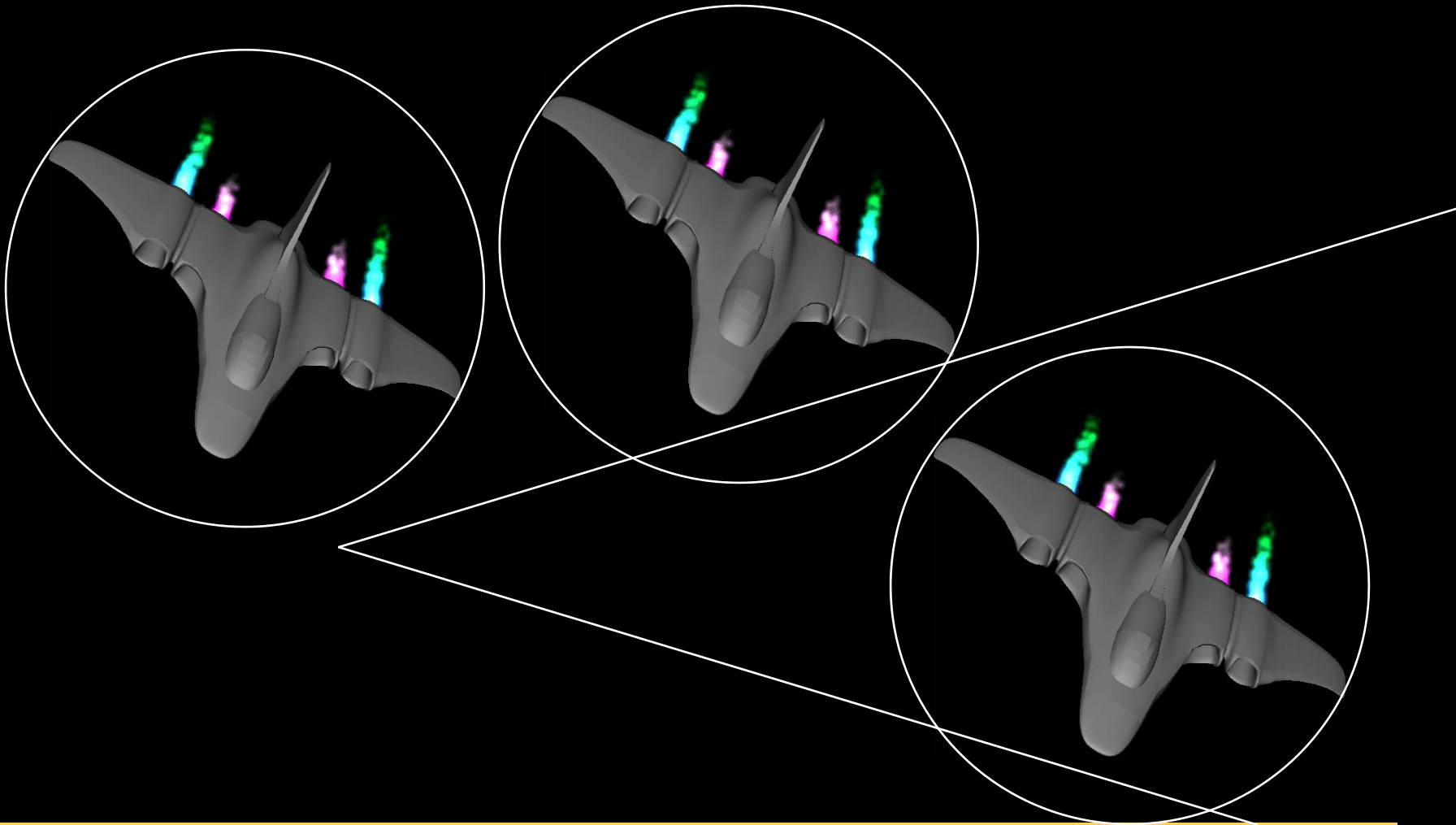
Culling

- 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

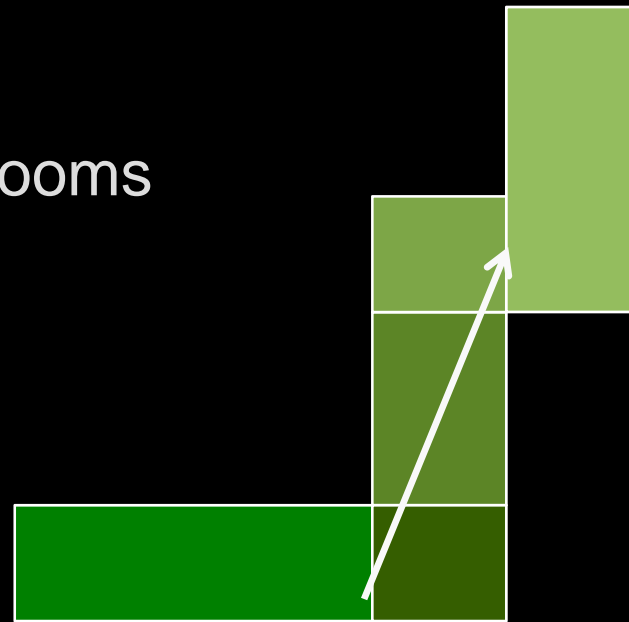


Bounding volumes



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



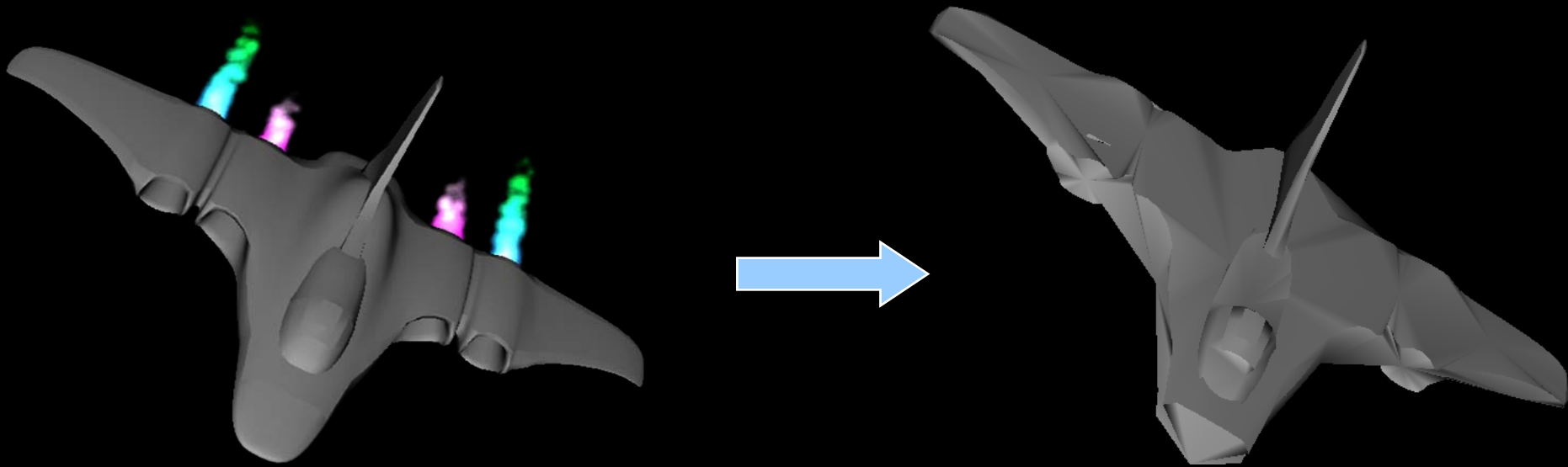
Cheating

- 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



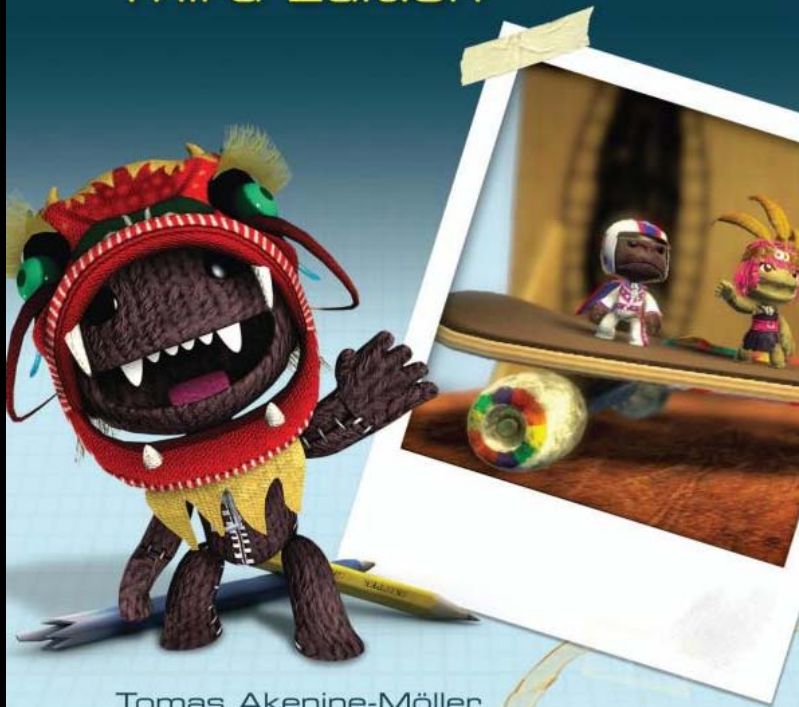
Summary Introduction

- 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

Preface	xi
1 Introduction	1
1.1 Contents Overview	2
1.2 Notation and Definitions	4
2 The Graphics Rendering Pipeline	11
2.1 Architecture	12
2.2 The Application Stage	14
2.3 The Geometry Stage	15
2.4 The Rasterizer Stage	21
2.5 Through the Pipeline	25
3 The Graphics Processing Unit	29
3.1 GPU Pipeline Overview	30
3.2 The Programmable Shader Stage	30
3.3 The Evolution of Programmable Shading	33
3.4 The Vertex Shader	38
3.5 The Geometry Shader	40
3.6 The Pixel Shader	42
3.7 The Merging Stage	44
3.8 Effects	45
4 Transforms	53
4.1 Basic Transforms	55
4.2 Special Matrix Transforms and Operations	65
4.3 Quaternions	72
4.4 Vertex Blending	80
4.5 Morphing	85
4.6 Projections	89
5 Visual Appearance	99
5.1 Visual Phenomena	99
5.2 Light Sources	100

viii	Contents	
5.3	Material	104
5.4	Sensor	107
5.5	Shading	110
5.6	Aliasing and Antialiasing	116
5.7	Transparency, Alpha, and Compositing	134
5.8	Gamma Correction	141
6	Texturing	147
6.1	The Texturing Pipeline	148
6.2	Image Texturing	156
6.3	Procedural Texturing	178
6.4	Texture Animation	180
6.5	Material Mapping	180
6.6	Alpha Mapping	181
6.7	Bump Mapping	183
7	Advanced Shading	201
7.1	Radiometry	202
7.2	Photometry	209
7.3	Colorimetry	210
7.4	Light Source Types	217
7.5	BRDF Theory	223
7.6	BRDF Models	251
7.7	BRDF Acquisition and Representation	264
7.8	Implementing BRDFs	269
7.9	Combining Lights and Materials	275
8	Area and Environmental Lighting	285
8.1	Radiometry for Arbitrary Lighting	286
8.2	Area Light Sources	289
8.3	Ambient Light	295
8.4	Environment Mapping	297
8.5	Glossy Reflections from Environment Maps	308
8.6	Irradiance Environment Mapping	314
9	Global Illumination	327
9.1	Shadows	331
9.2	Ambient Occlusion	373
9.3	Reflections	386
9.4	Transmittance	392
9.5	Refractions	396
9.6	Caustics	399
9.7	Global Subsurface Scattering	401

Contents		ix
9.8	Full Global Illumination	407
9.9	Precomputed Lighting	417
9.10	Precomputed Occlusion	425
9.11	Precomputed Radiance Transfer	430
10	Image-Based Effects	439
10.1	The Rendering Spectrum	440
10.2	Fixed-View Effects	440
10.3	Skyboxes	443
10.4	Light Field Rendering	444
10.5	Sprites and Layers	445
10.6	Billboarding	446
10.7	Particle Systems	455
10.8	Displacement Techniques	463
10.9	Image Processing	467
10.10	Color Correction	474
10.11	Tone Mapping	475
10.12	Lens Flare and Bloom	482
10.13	Depth of Field	486
10.14	Motion Blur	490
10.15	Fog	496
10.16	Volume Rendering	502
11	Non-Photorealistic Rendering	507
11.1	Toon Shading	508
11.2	Silhouette Edge Rendering	510
11.3	Other Styles	523
11.4	Lines	527
12	Polygonal Techniques	531
12.1	Sources of Three-Dimensional Data	532
12.2	Tessellation and Triangulation	534
12.3	Consolidation	541
12.4	Triangle Fans, Strips, and Meshes	547
12.5	Simplification	561
13	Curves and Curved Surfaces	575
13.1	Parametric Curves	576
13.2	Parametric Curved Surfaces	592
13.3	Implicit Surfaces	606
13.4	Subdivision Curves	608
13.5	Subdivision Surfaces	611
13.6	Efficient Tessellation	629

x	Contents	
14	Acceleration Algorithms	645
14.1	Spatial Data Structures	647
14.2	Culling Techniques	660
14.3	Hierarchical View Frustum Culling	664
14.4	Portal Culling	667
14.5	Detail Culling	670
14.6	Occlusion Culling	670
14.7	Level of Detail	680
14.8	Large Model Rendering	693
14.9	Point Rendering	693
15	Pipeline Optimization	697
15.1	Profiling Tools	698
15.2	Locating the Bottleneck	699
15.3	Performance Measurements	702
15.4	Optimization	703
15.5	Multiprocessing	716
16	Intersection Test Methods	725
16.1	Hardware-Accelerated Picking	726
16.2	Definitions and Tools	727
16.3	Bounding Volume Creation	732
16.4	Geometric Probability	735
16.5	Rules of Thumb	737
16.6	Ray/Sphere Intersection	738
16.7	Ray/Box Intersection	741
16.8	Ray/Triangle Intersection	746
16.9	Ray/Polygon Intersection	750
16.10	Plane/Box Intersection Detection	755
16.11	Triangle/Triangle Intersection	757
16.12	Triangle/Box Overlap	760
16.13	BV/BV Intersection Tests	762
16.14	View Frustum Intersection	771
16.15	Shaft/Box and Shaft/Sphere Intersection	778
16.16	Line/Line Intersection Tests	780
16.17	Intersection Between Three Planes	782
16.18	Dynamic Intersection Testing	783
17	Collision Detection	793
17.1	Collision Detection with Rays	795
17.2	Dynamic CD using BSP Trees	797
17.3	General Hierarchical Collision Detection	802
17.4	OBBTree	807

Contents	xi	
17.5	A Multiple Objects CD System	811
17.6	Miscellaneous Topics	816
17.7	Other Work	826
18	Graphics Hardware	829
18.1	Buffers and Buffering	829
18.2	Perspective-Correct Interpolation	838
18.3	Architecture	840
18.4	Case Studies	859
19	The Future	879
19.1	Everything Else	879
19.2	You	885
A	Some Linear Algebra	889
A.1	Euclidean Space	889
A.2	Geometrical Interpretation	892
A.3	Matrices	897
A.4	Homogeneous Notation	905
A.5	Geometry	906
B	Trigonometry	913
B.1	Definitions	913
B.2	Trigonometric Laws and Formulae	915
	Bibliography	921
	Index	1003