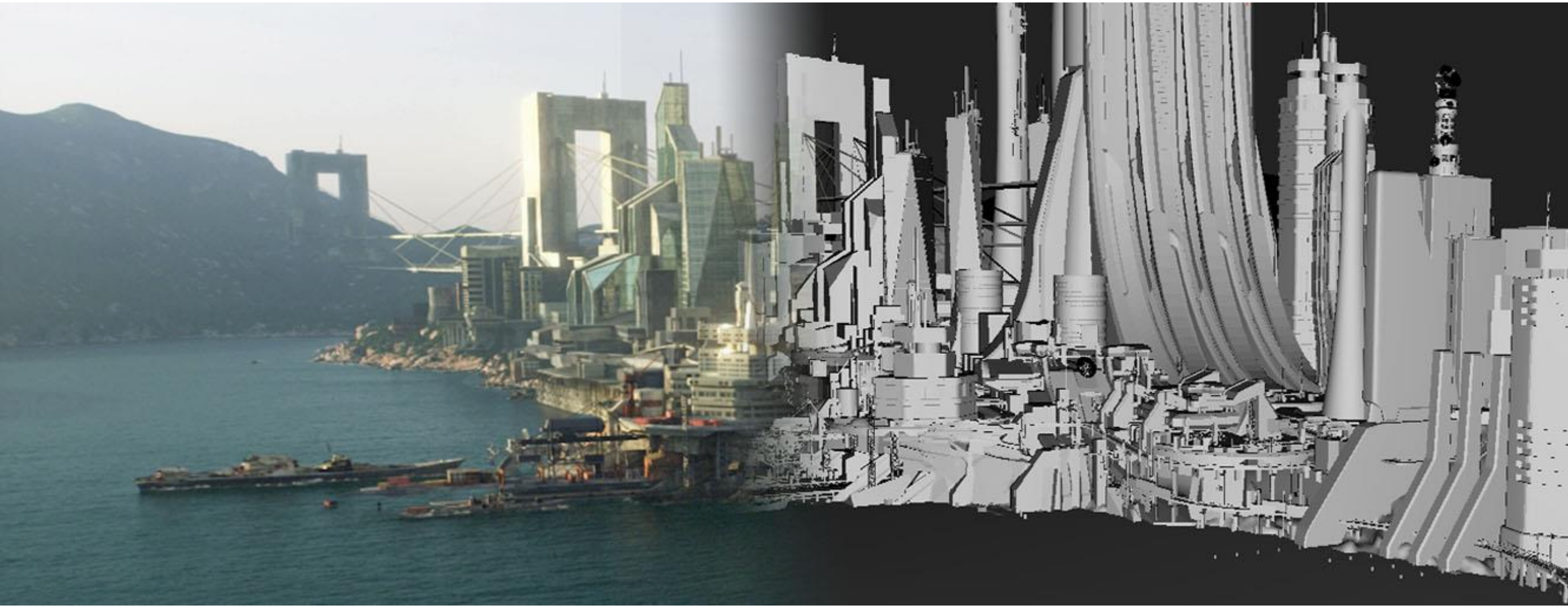
COMPUTER GRAPHICS COURSE

Introduction

Georgios Papaioannou - 2014

Image Synthesis

Image Synthesis



Educational Goals

- Learn about image synthesis techniques and algorithms
- Find out how image synthesis is applied to common tasks and applications
- Learn how to develop applications using computer graphics (BSc)
- Find out how complex imagery is created (MSc)

AUEB COMPUTER GRAPHICS GROUP

Course Structure

BSc

- Lectures (theory + Unity primer)
- Labs (OpenGL)

- Written exams
- Final project

- Grading system:
  - Exams: 8 points
  - Project: 3 points
  - Max 10 points

MSc

- Lectures
- Paper reading

- Written exams
- Surveys
- Final project (optional)

- Grading system:
  - Exams: 8 points
  - Oral presentation / project: 3 points
  - Max 10 points

Prerequisites

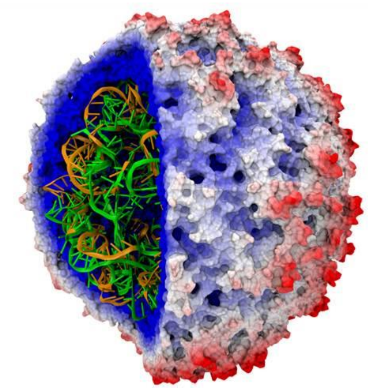
- Basic linear algebra and calculus
- Basic understanding of computing architectures
- Basic programming skills (preferably in C/C++)

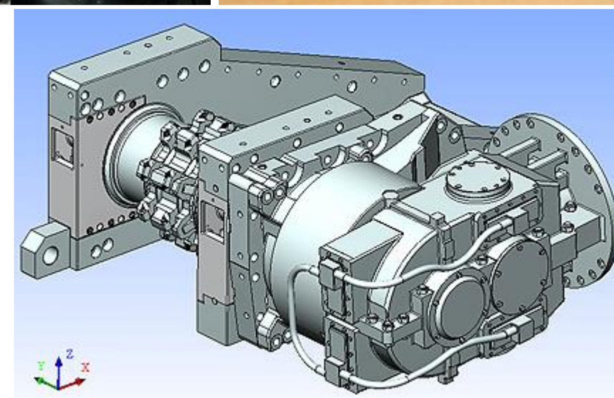
MSc:

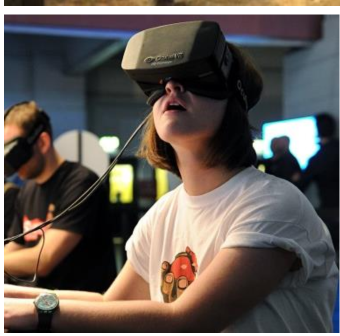
- Elements of probability theory
- Basic linear algebra and calculus
- A plus, but not mandatory:
  - Undergraduate course in CG

AUEB COMPUTER GRAPHICS GROUP

What do we use CG for?







CG Applications

- Computer games and interactive applications
  - Most high-efficiency algorithms come from and target this application domain!
  - Big industry, from indy and casual games to AAA productions
- Computer-Aided Design / Manufacturing / Engineering
  - Physical product design using surface and solid modeling and geometric tools
- GUIs
  - 2D GUI implementation and acceleration
  - VR / AR and human-friendly interactive systems

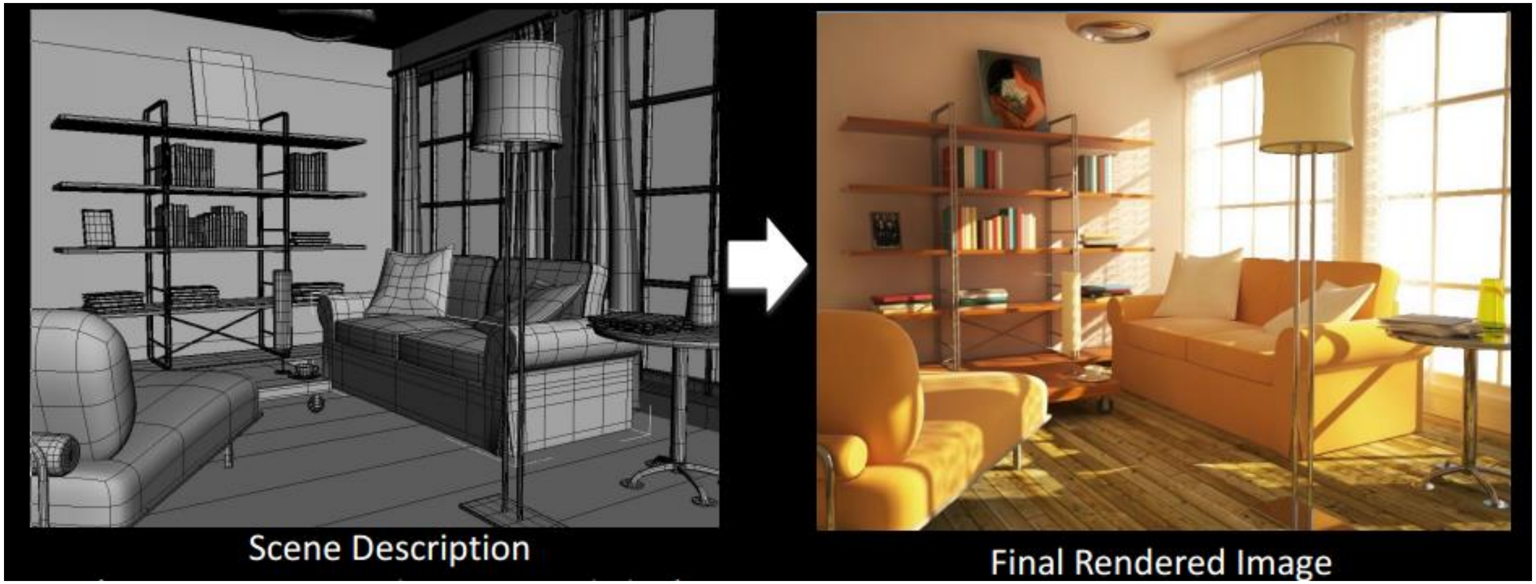
CG Applications

- Special effects for feature films
  - Ability to create the impossible or non-existent
- Animated films
- Scientific and medical visualization
- 2D and 3D printing technology

DEFINITIONS

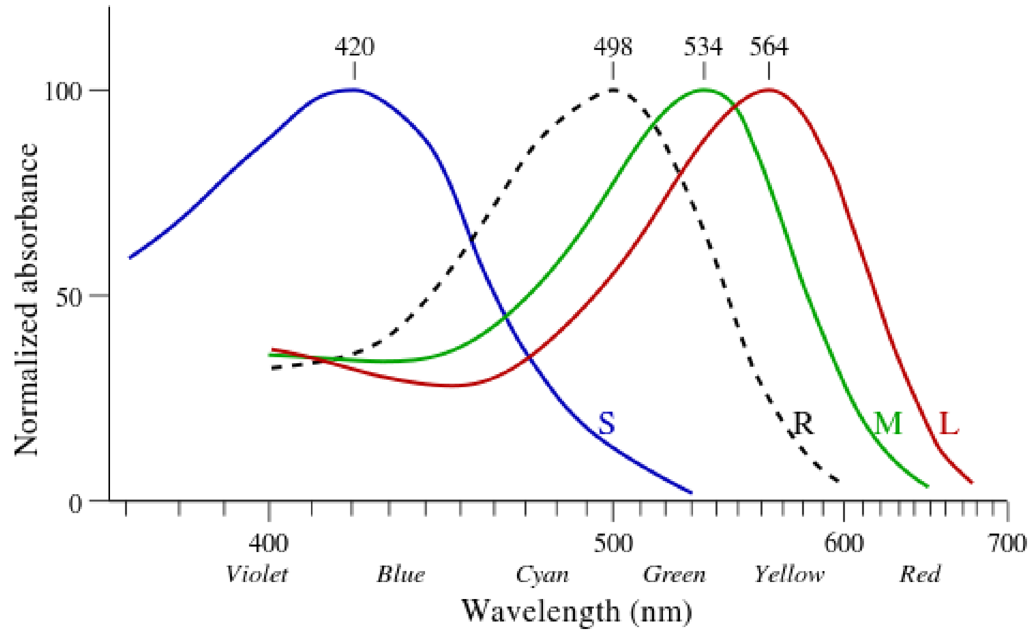
Rendering

- Rendering is the process of generating an image from a set of models (geometric representation)
  - It is the final product of a general image synthesis task using a rendering pipeline



The Human Visual System

- Sensitive photoreceptors: Rodes and Cones
- The human visual system adapts to the level of illumination incident to the photoreceptors
  - Rods (scotoptic light): $10^{-6}cd/m^2 – 10cd/m^2$
  - Cones (photoptic light): $10^{-2}cd/m^2 – 10^8\ cd/m^2$
- Total luminance range: $10^8:10^{-6}$

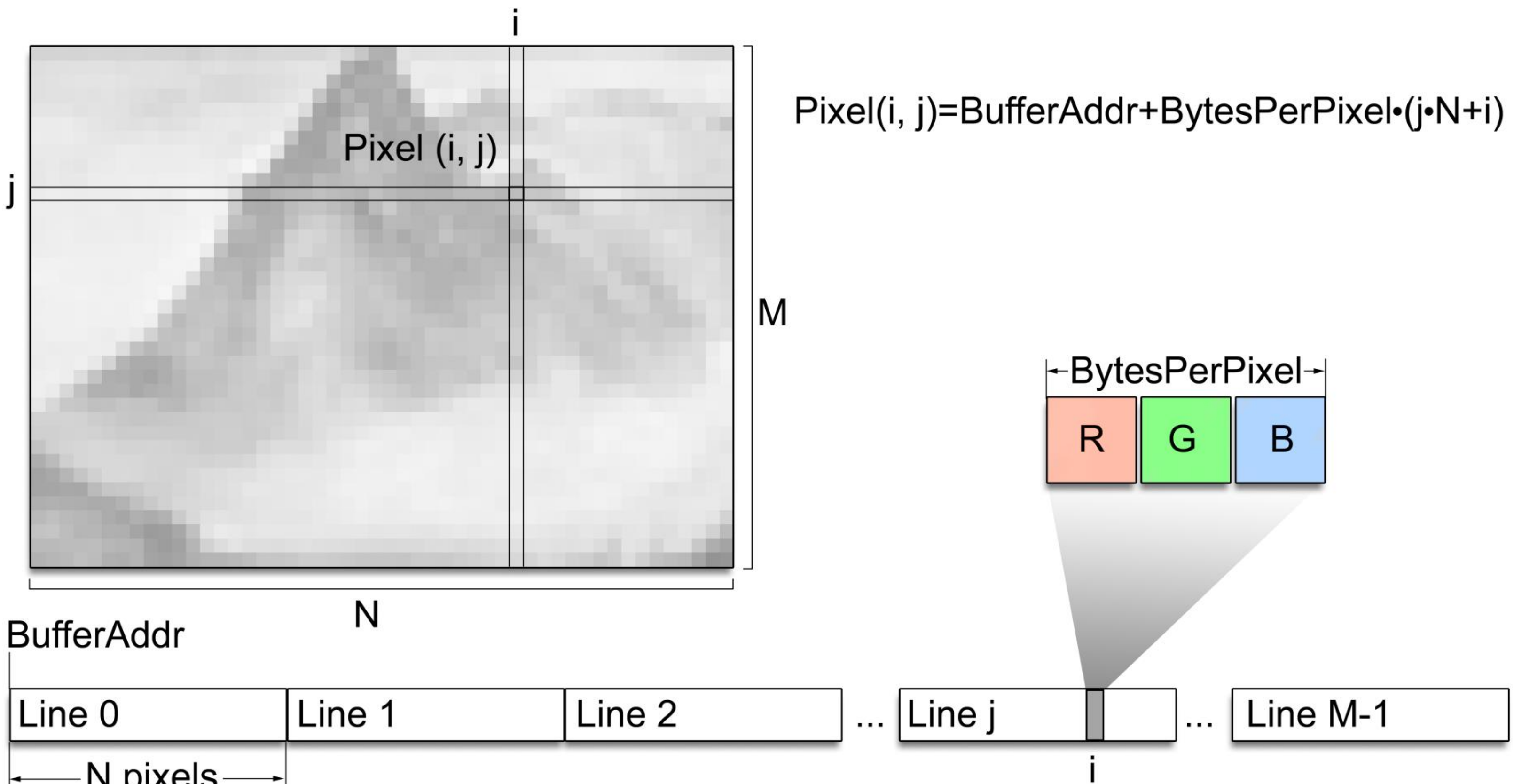


The Digital Image

- A discretized configuration of intensity samples
- Usually an array of pixels (a *raster*)
  - Discrete approximation of a 2D continuous signal
  - Luminance is sampled using a fixed or variable rate and attributed to the neighborhood of each pixel
- Image color data are represented using a color model
  - RGB (compatible with HVS)
  - Other (see Color chapter)
- Storage:
  - Separate color channels per pixel

Image – Storage

- Typically stored as an array in memory
- Interleaved color channels
- Line/column configuration, but also in blocks



Dynamic Range

- Dynamic range: the minimum to maximum luminance level achieved by a system
- HVS range: 10^8:10^{-6}
- H/W cannot achieve these levels simultaneously!
  - We use *tone mapping* to adjust the "useful" range to match the output range of a device
- Physically measured or simulated radiance (therefore luminance) in a natural environment matches the HVS levels
- Typical displays can achieve a **dynamic contrast ratio** of 6000:1 and an **actual luminance level** of 1-300cd/m^2

High Dynamic Range Images

- Use floating point arithmetic representation to store a wide range of luminance values
- Used both in CG and photography
- Typical integer buffers and image formats (8 bits per color channel) are not enough
- Precision depends on storage limitations and application: unsigned bytes: 24bit color, floating point (half/full): 48/96bit images
- HDR screens use a combination of 8bpp panels and temporal dithering to increase the perceived levels.

The Frame Buffer

- The area of memory that stores the resulting pixels/fragments during rendering.
- Can either represent:
  - The final displayed frame
  - Intermediate results, later to be used as textures

Intermediate Frame Buffer (render-to-texture)

Final Frame Buffer with Color Grading

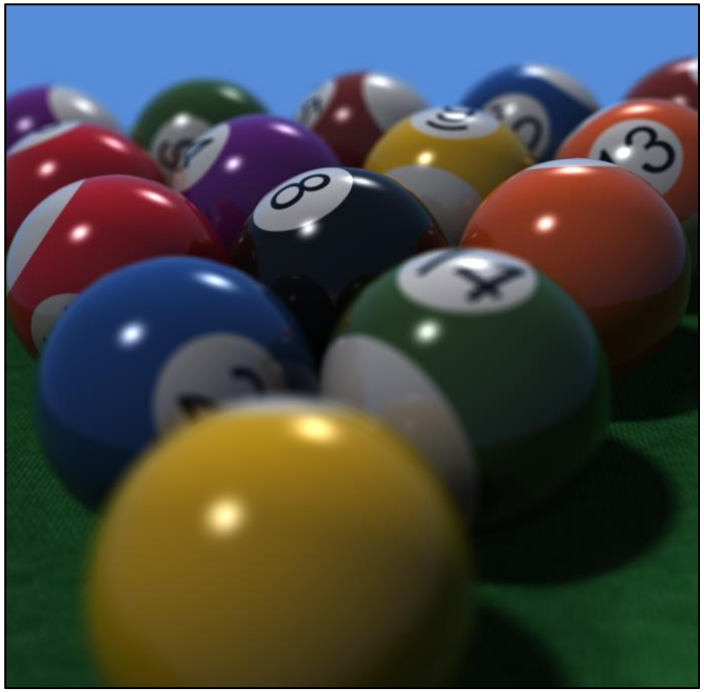
Offline and Real-time Rendering

- Offline Rendering
  - Quality is fixed, time is negotiable
  - No Artifacts (AA, motion blur, smooth surfaces etc)
  - Want < 1min per frame, can accept 10-12 hours
  - Typical machine: *render farm* (computing cluster)
- Real-time Rendering
  - Time is fixed, quality is negotiable
  - Many artifacts (aliasing, poor lighting)
  - Max bound: ~16 ms (60 fps),
  - Can accept ~50 ms (20 fps)
  - Typical machine: commodity hardware (GPUs), game consoles, mobile devices

Where is the Borderline?

- Increasingly fast GPUs and the many-core implementation of "traditionally" offline algorithms blur the border
- Still, physically correct and high quality rendering of complex environments are offline

Computer game (Unity 5 demo - 2016)



Real-time ray tracing of a simple scene

What About 2D Rendering?

- Today, most hardware-accelerated 2D drawing is handled via the 3D h/w pipeline!
  - Textured polygons and framebuffers for views and bitmap storage: fast access and redraw, easy transitions and effects
  - Blending and widget ordering handled by the hidden surface removal and blending of the GPU
  - Shape rasterization and fills via GPU rasterization

Graphics Pipelines

- "Graphics Pipeline" is the sequence of steps that we use to create the final image
- Many graphics/rendering pipelines have been proposed

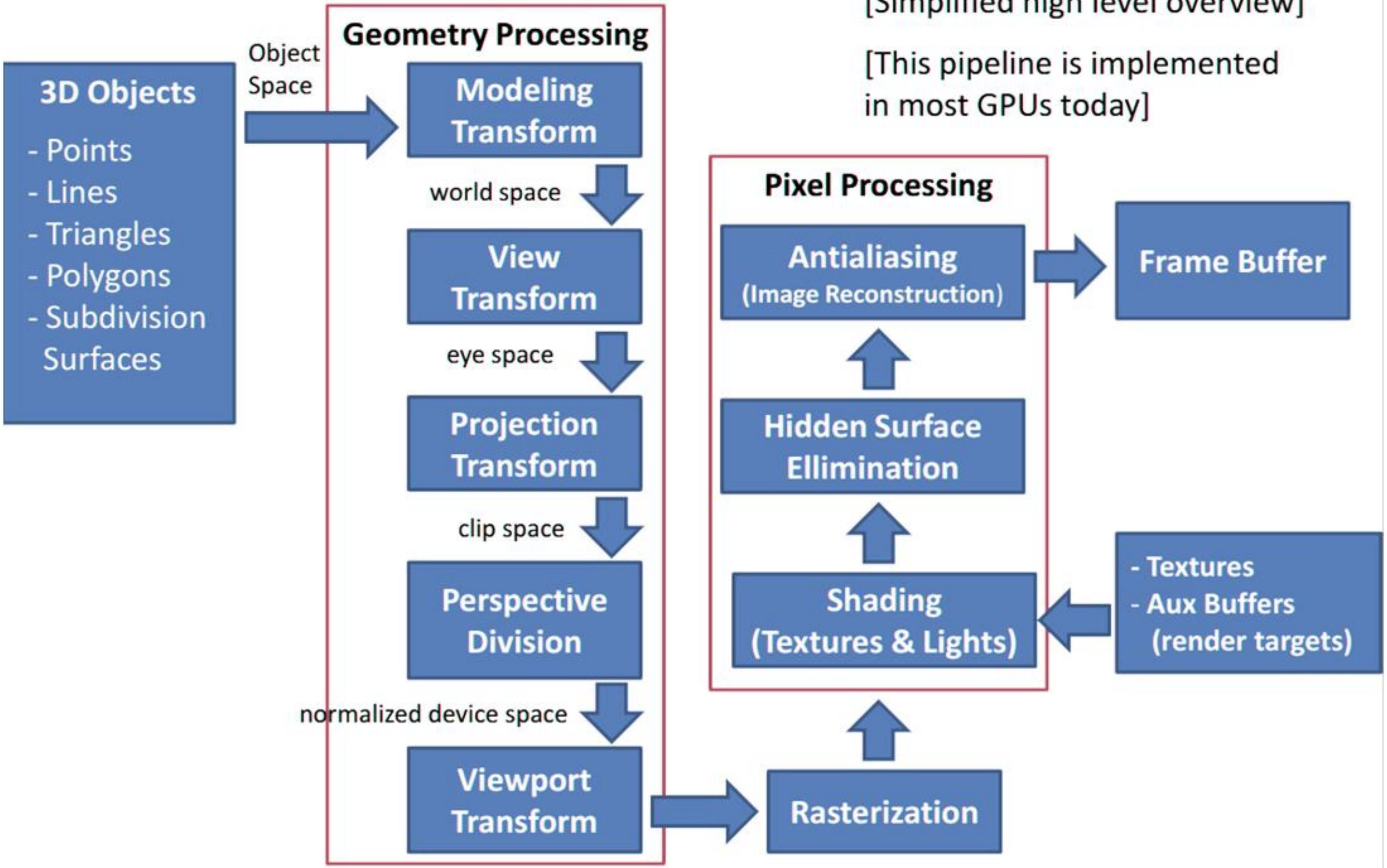
Graphics Pipelines (2)

- Scanline- / Rasterization-based
  - Immediate Direct rendering, Tile-Based, Deferred Rendering, 2D shape rasterization (windowing systems, GUIs)
  - Used mainly for real-time rendering (GPUs)
- Micropolygon-based Reyes (e.g. old Pixar's Renderman)
- Ray Tracing-based
  - Path tracing, photon mapping, bidirectional path tracing etc.
  - Used for advanced lighting simulations

GPU Rasterization Pipeline

[Simplified high level overview]

[This pipeline is implemented in most GPUs today]



Contributors

- Georgios Papaioannou
- Pavlos Mavridis