

Introduction and Overview

CS425: Computer Graphics I

Fabio Miranda

<https://fmiranda.me>

Overview

- Intro to computer graphics
- Course goals
- Logistics
- Requirements
- Content overview
- Grading
- Assignments
- Policy
- Workflow pattern
- Resources
- Communication

What is Computer Graphics?

- Broad sense: use of computer to create and manipulate images.
- Combination of hardware (input, processing, output) and software.
- 2D or 3D.

Example

Cyberpunk 2077

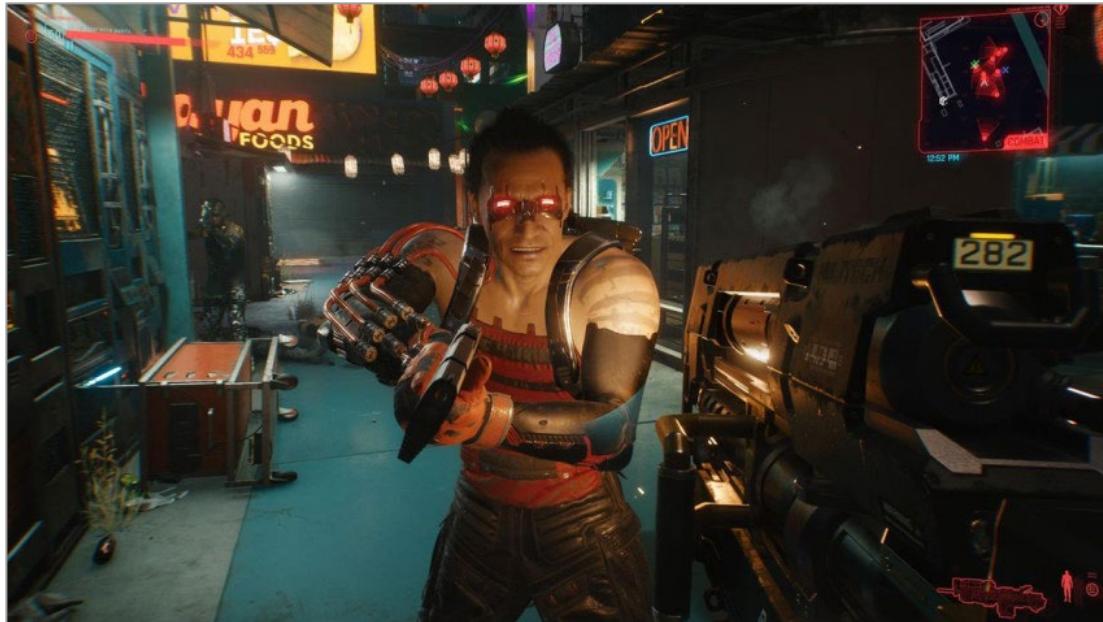


Google Earth



Example

How was this image created?



Graphics system

Urbane,
OpenSpace,
ArcGIS,
Google Earth, ...

Cyberpunk, Doom, FIFA, ...

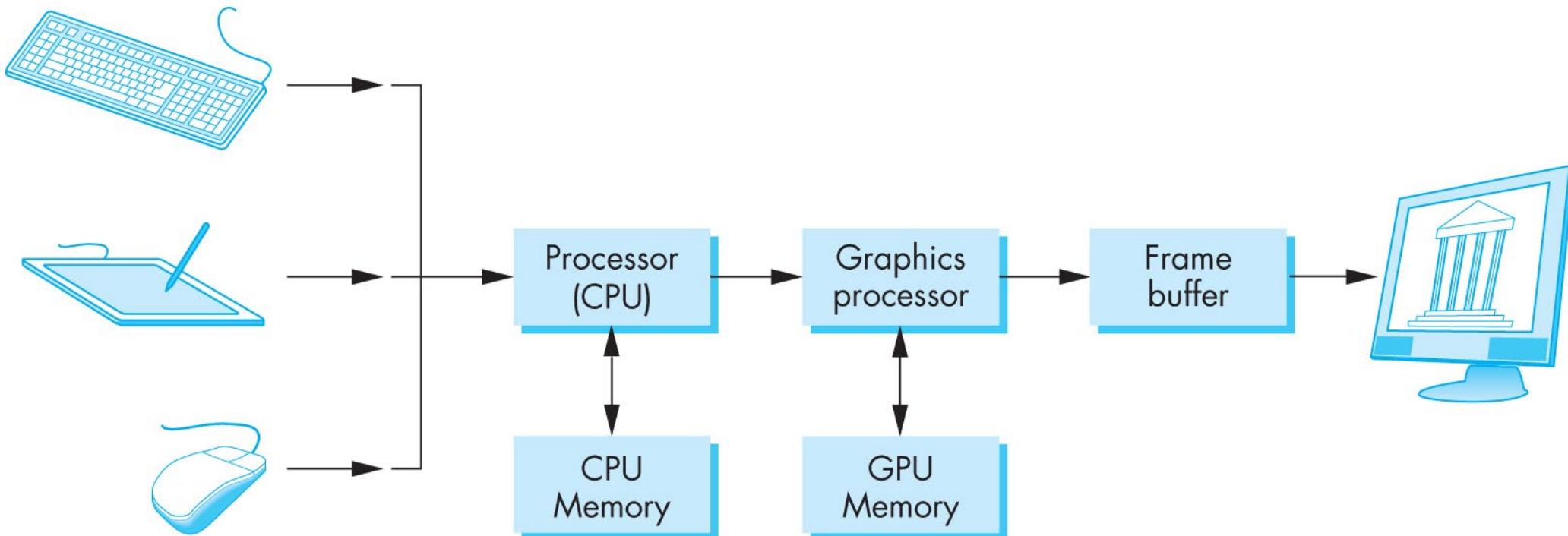
Unreal, Unity, idTech,
Frostbite, REDengine, ...

OpenGL, DirectX, Vulkan, Metal, Glide, ...

Windows, Linux, iOS, ...

CPU, RAM, GPU, VRAM, ...

Basic graphics system



From: Interactive Computer Graphics 7th Ed by
Professor Ed Angel and Dave Shreiner

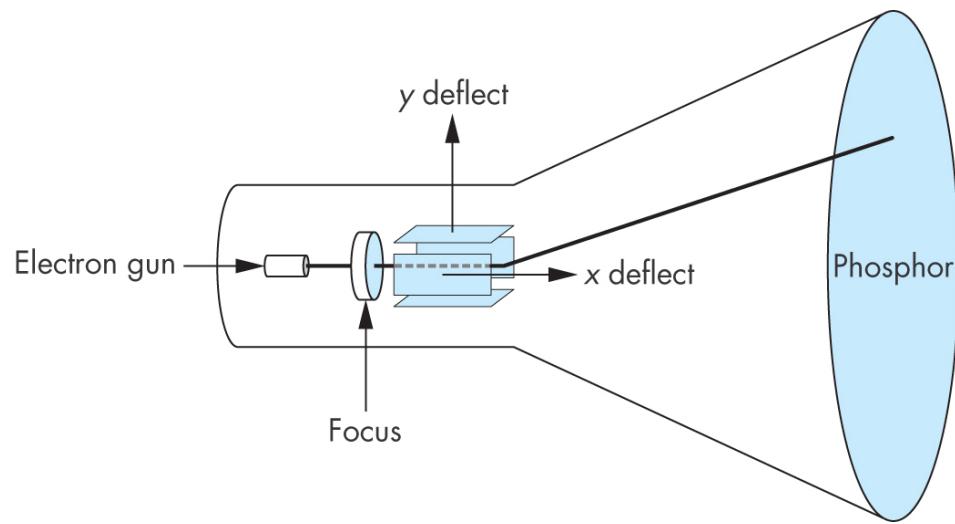
Basic graphics system



Frame buffer: a 2D array with color values

Computer Graphics: 1950s

- Introduction of cathode-ray tube (CRT) as a viable display.
- Light pen as input.



Computer Graphics: 1960s

- The phrase “computer graphics” is coined by Verne L. Hudson and William Fetter from Boeing.
- Wireframe graphics.
- Sutherland introduces Sketchpad, the first computer graphical user interface:
 - Precise drawing
 - Erase, move
 - Zoom in and out
- First ray casting algorithm by Arthur Appel.



Computer Graphics: 1960s

“The Sketchpad system uses drawing as a novel communication medium for a computer. The system contains input, output, and computation programs which enable it to interpret information drawn directly on a computer display. It has been used to draw electrical, mechanical, scientific, mathematical, and animated drawings; it is a general purpose system. Sketchpad has shown the most usefulness as an aid to the understanding of processes, such as the notion of linkages, which can be described with pictures. Sketchpad also makes it easy to draw highly repetitive or highly accurate drawings and to change drawings previously drawn with it.”

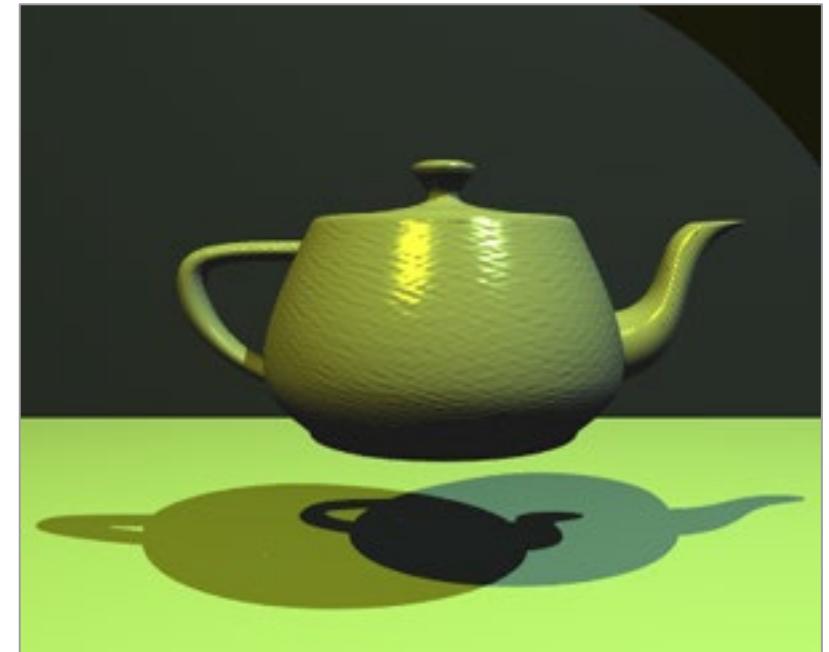
Sketchpad: A man-machine graphical communication system

By Ivan Sutherland

1963

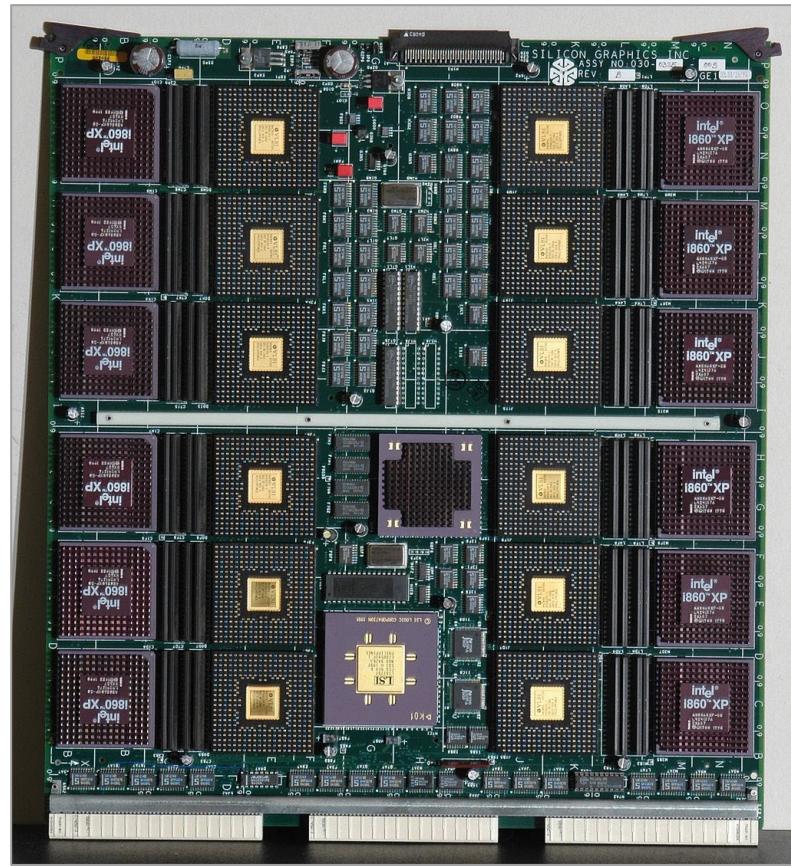
Computer Graphics: 1970s

- Beginning of graphics standards:
 - GKS: ISO 2D standard.
 - Core: 3D, not ISO standard.
- Pong, the first commercially successful video game.
- First commercial framebuffer.
- Utah teapot.
- Gouraud and Blinn-Phong shading models.
- Workstations and PCs.



Computer Graphics: 1980s

- Special purpose hardware:
 - Silicon Graphics geometry engine
 - SGI: 3D raster graphics hardware
- Shaded solids, no textures.



Computer Graphics: 1990s

- OpenGL API.
- Toy Story: First computer-generated feature-length movie .
- GeForce 256: first consumer-level with hardware-accelerated transforming and lighting.
- Texture mapping.
- Doom, Quake.



Computer Graphics: 2000s

- GeForce 3: first to support programmable shaders.
- Proliferation of CG movies.
- GPU being used in other area: computer vision, machine learning, bioinformatics, etc.
- CUDA, OpenCL.



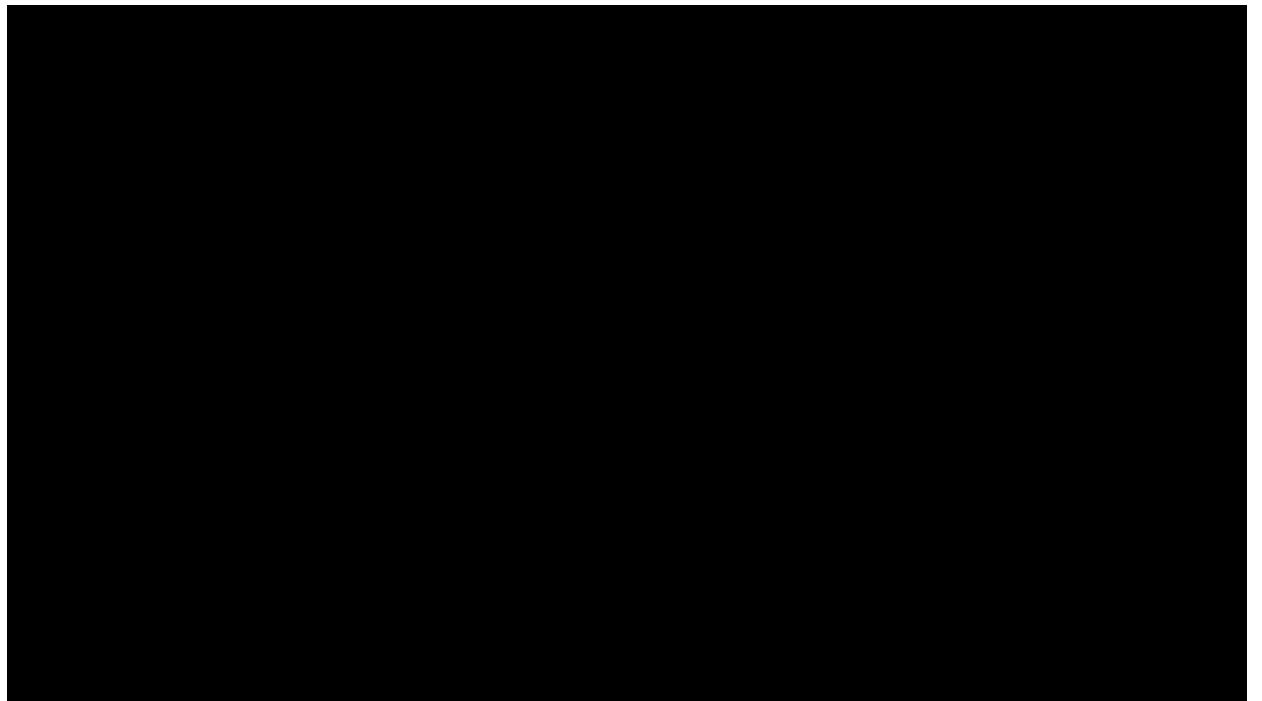
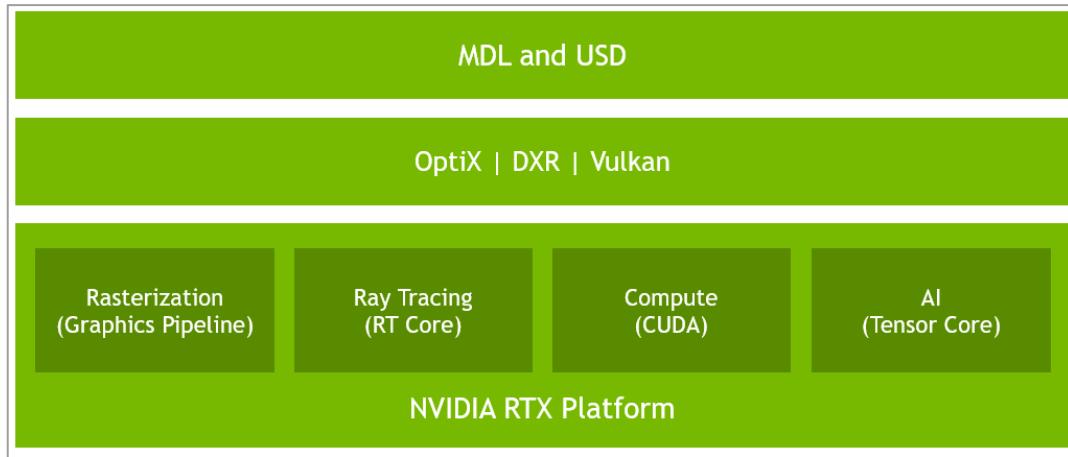
Computer Graphics: 2010s

- Graphics is now ubiquitous:
 - Cell phones.
 - Embedded.
- OpenGL ES and WebGL.
- Virtual Reality.
- 3D movies and TV.



Computer Graphics: 2020s

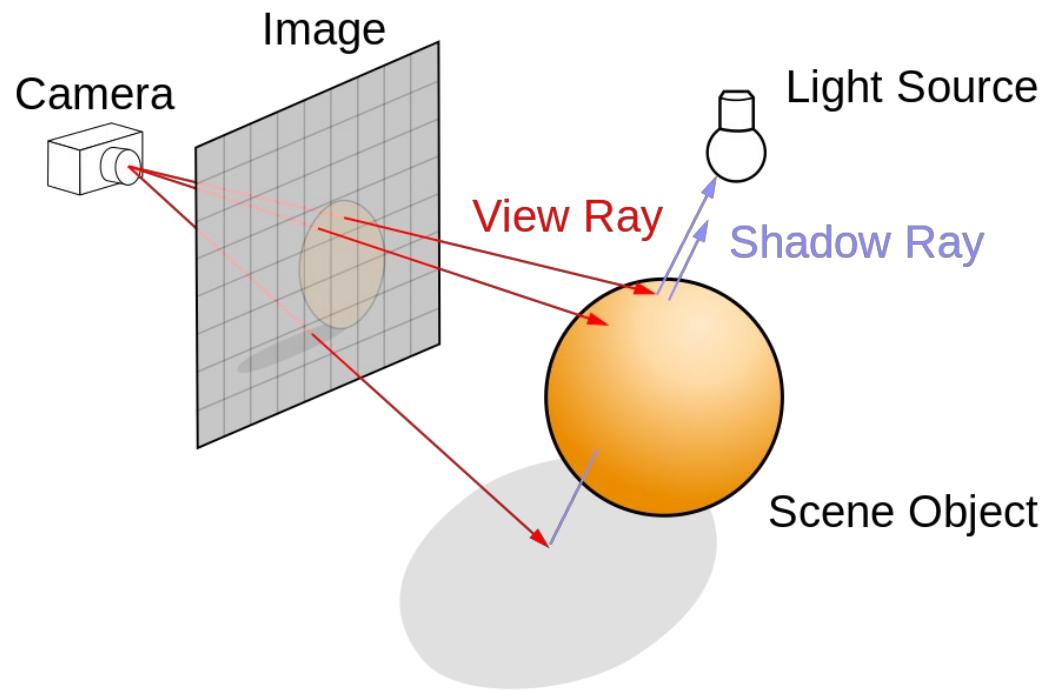
- Real-time ray tracing.
- 4K resolution.
- AI-based anti-aliasing.



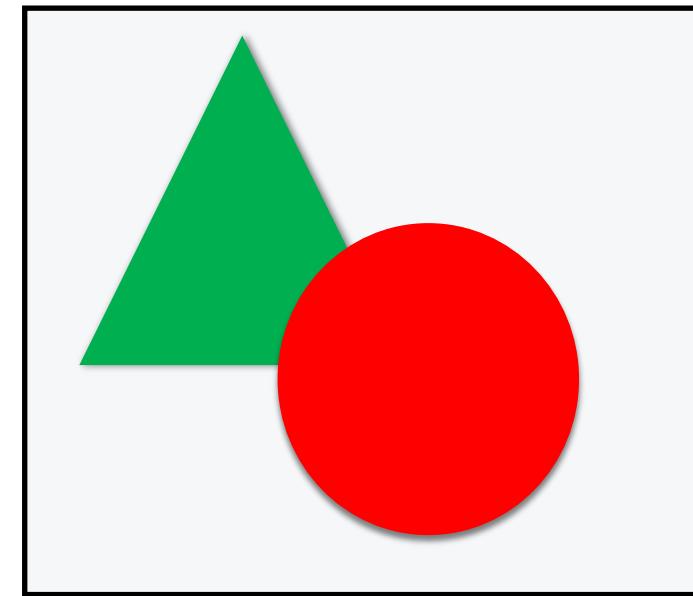
Nvidia - Reflections RTX Tech Demo

Creating images

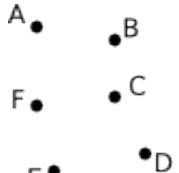
Per pixel: ray tracing



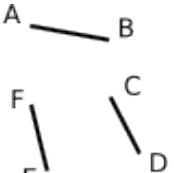
Per object: rasterization



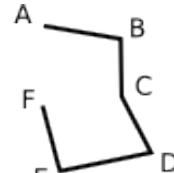
Creating images: Objects description



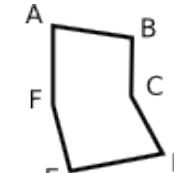
GL_POINTS



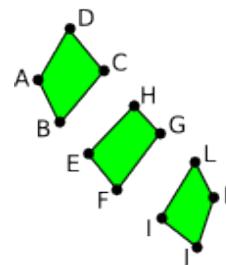
GL_LINES



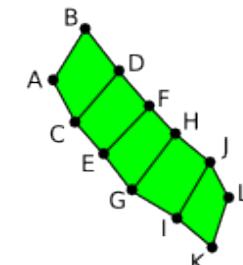
GL_LINE_STRIP



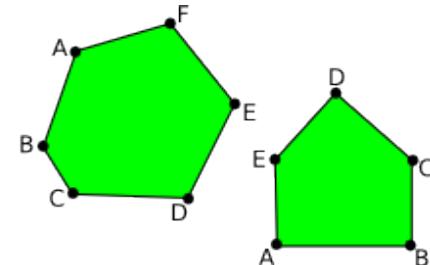
GL_LINE_LOOP



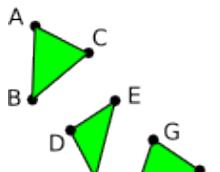
GL_QUADS



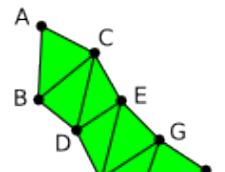
GL_QUAD_STRIP



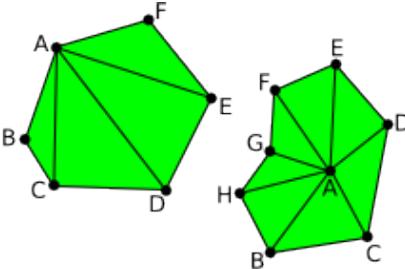
GL_POLYGON



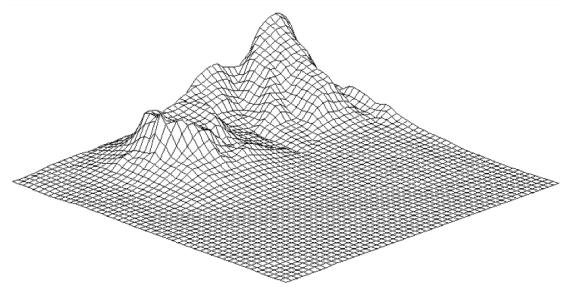
GL_TRIANGLES



GL_TRIANGLE_STRIP

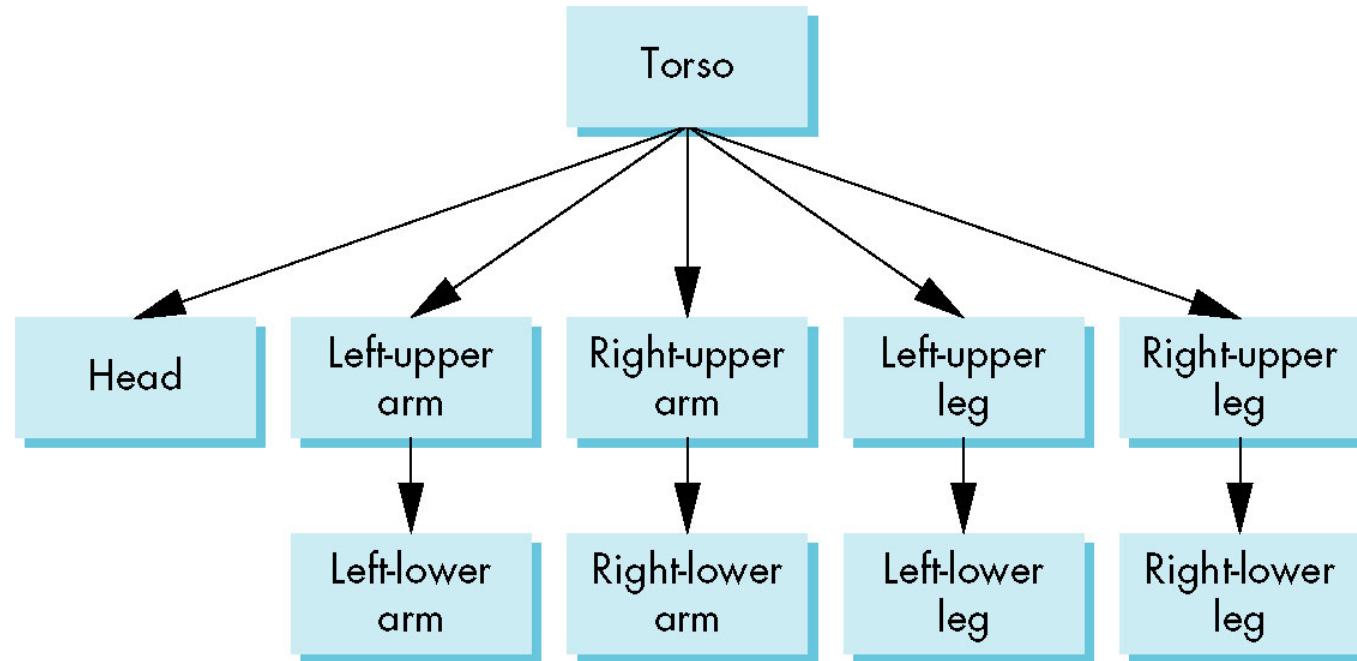
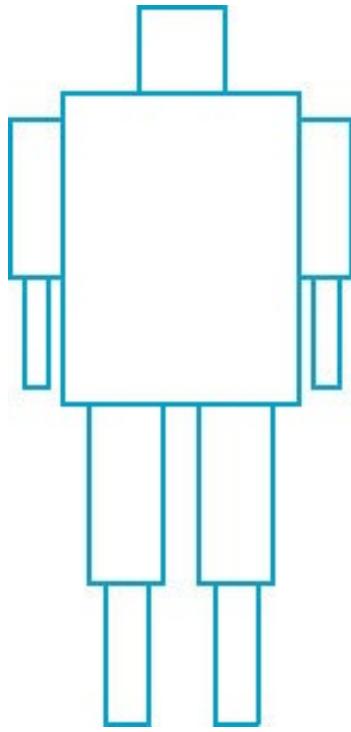


GL_TRIANGLE_FAN

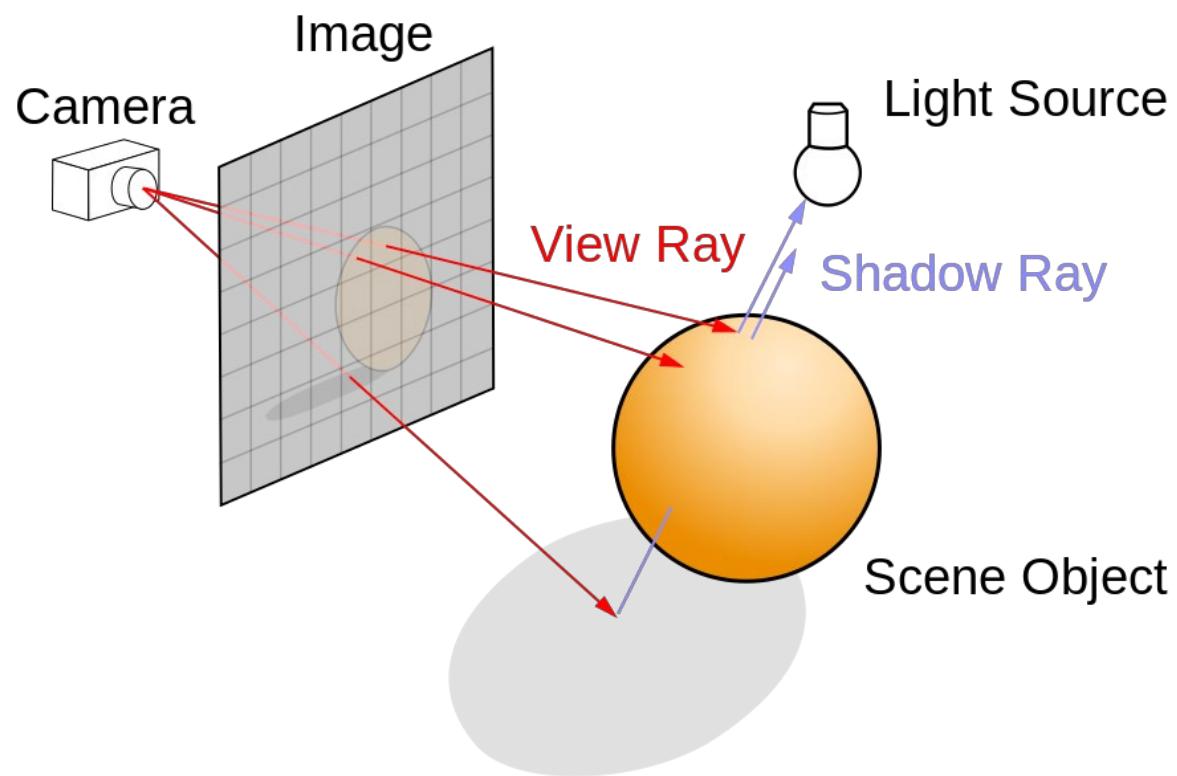


From: Introduction to Computer Graphics,
David J. Eck

Creating images: Scene description

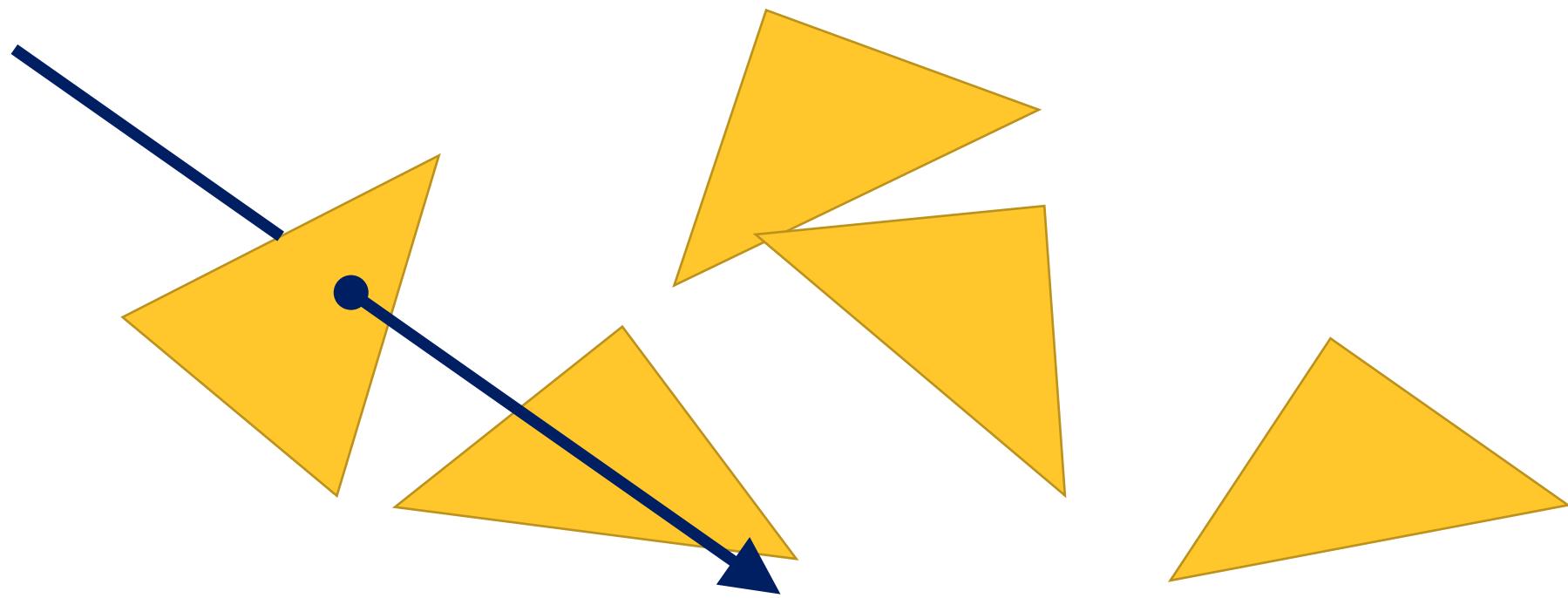


Creating images: Ray tracing

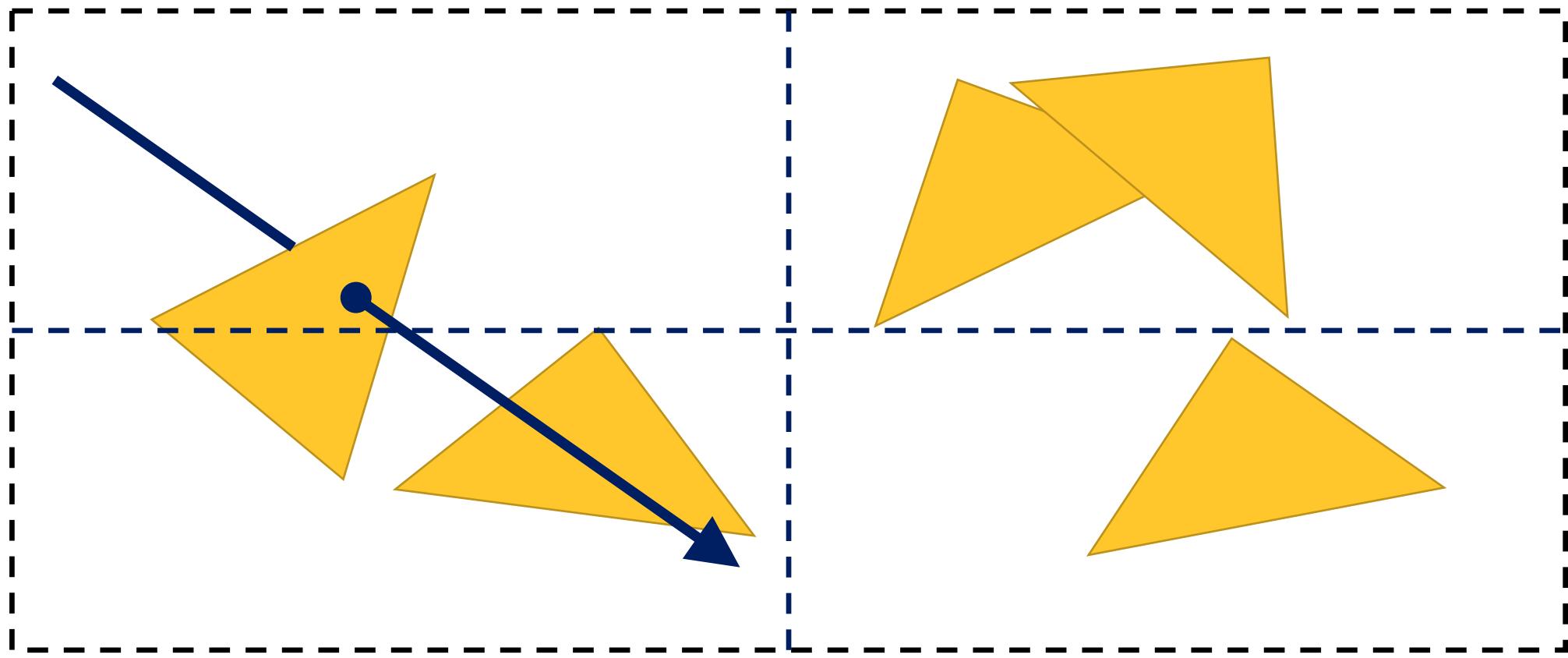


- Easy to parallelize but hard to map to hardware (up until recently)
- Expensive!
- It can be extended to model many physical phenomena (internal scattering, diffraction, reflections, etc)
- Used to obtain high-quality images

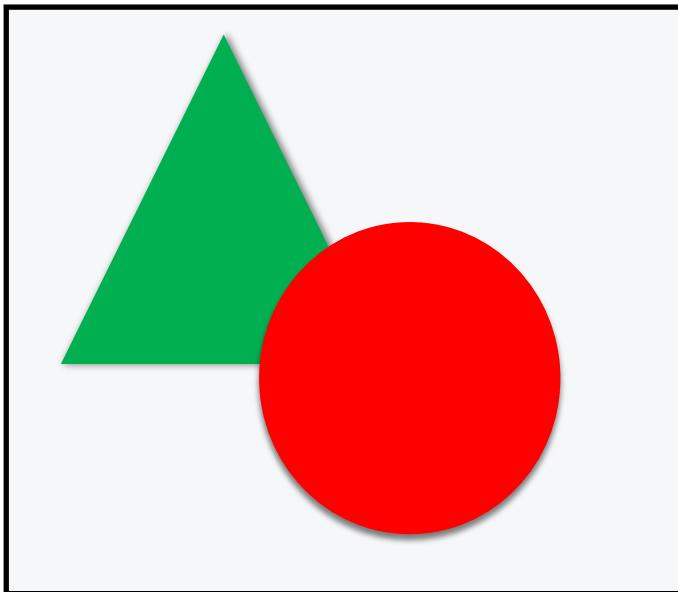
Creating images: Ray tracing



Creating images: Ray tracing

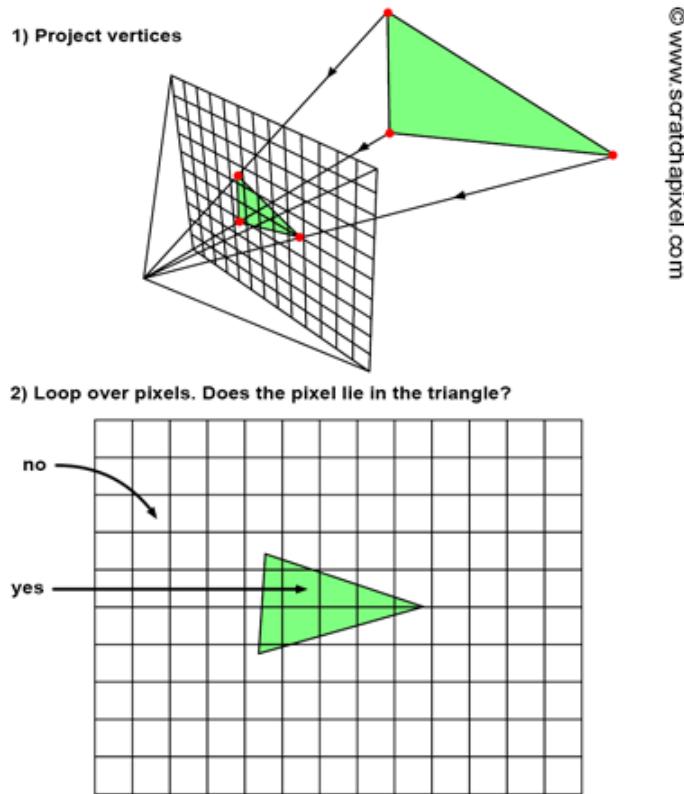


Creating images: Rasterization



- Easy to map to hardware
- While it cannot model directly complex effects, we can approximate them
- Used in interactive applications (mostly)

Creating images: Rasterization



1. Project 3D vertices onto the screen.
2. Loop through pixels in the image and test if they lie within the resulting 2D triangles.

Real-time rendering vs off-line rendering

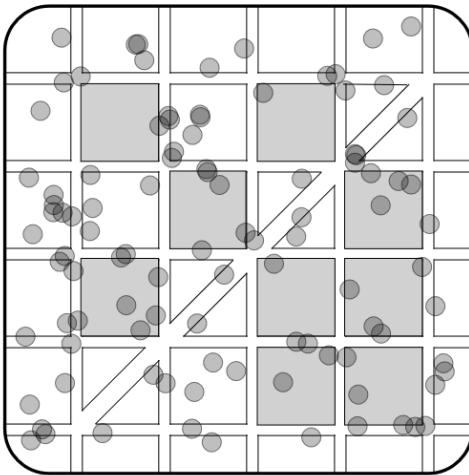


Doom (2016)



Soul (2020)

Computer Graphics & Visualization

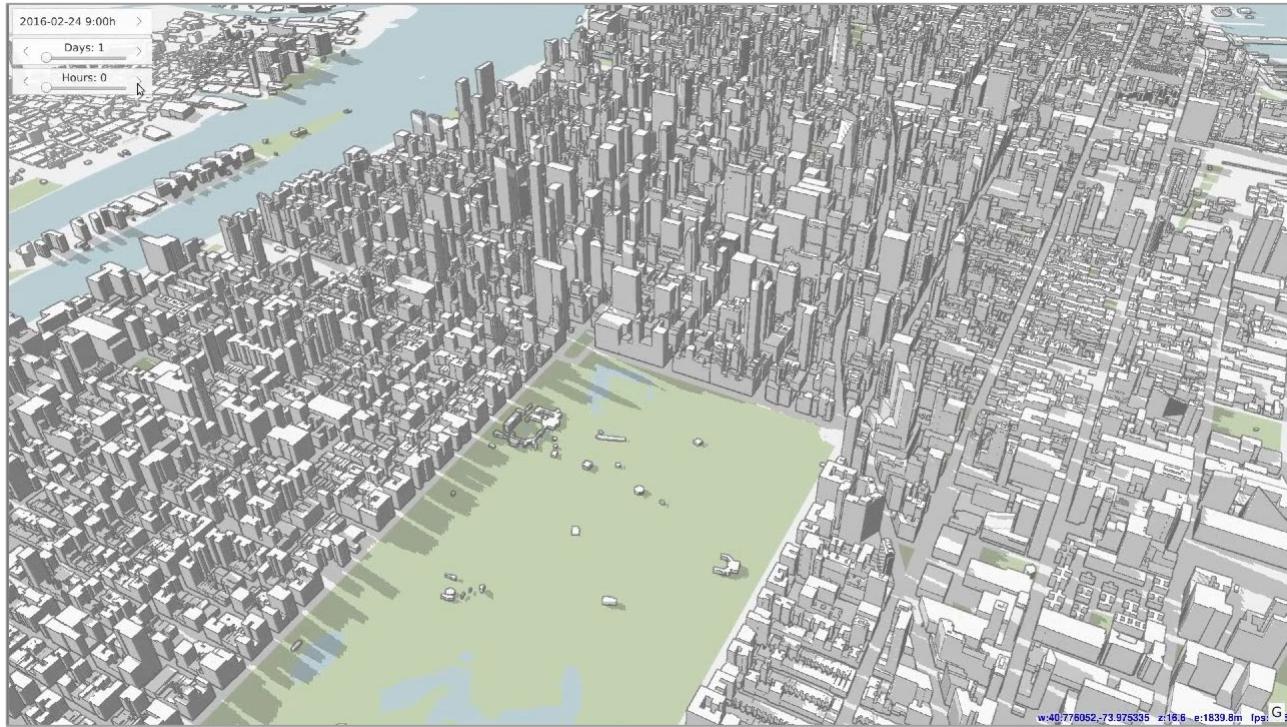


Open Street Maps

The screenshot shows the NYC 3-D Building Model website. At the top, there's a navigation bar with links for Home, About, Initiatives, Residents, Business, For Agencies, Contact, and Search. The main title is "NYC 3-D Building Model". Below the title, there's a section with social sharing icons (Facebook, Twitter, Print) and a brief description of the model. The description mentions the use of CityGML specification and highlights the inclusion of every NYC building present in the 2014 aerial survey. It also notes the hybrid specification combining elements from Level of Detail (LOD) 1 and 2. A small thumbnail image of the city is visible on the right.

DoITT

Computer Graphics & Visualization



Shadow Accrual Maps: Efficient Accumulation of City-Scale Shadows Over Time

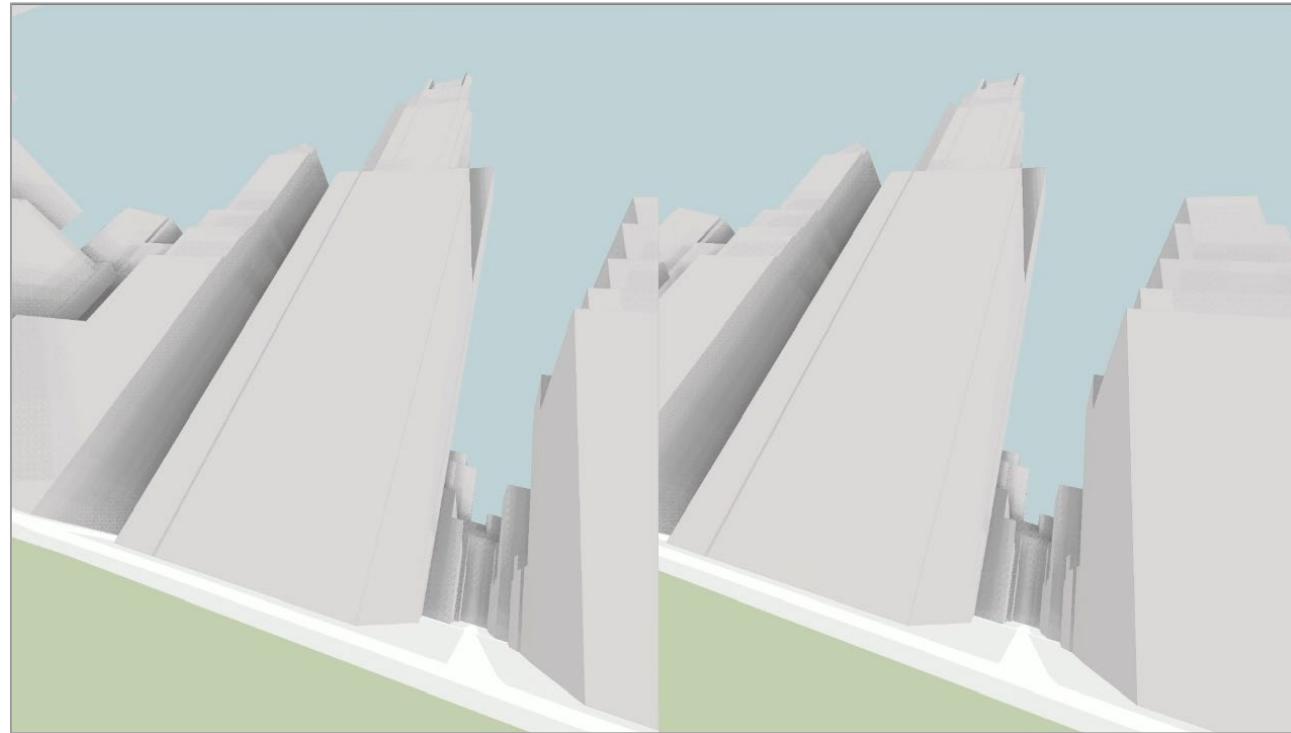


<https://nyti.ms/2k0bF0G>



COMPUTER SCIENCE

Computer Graphics & Virtual Reality



Urbanrama



Computer Graphics & Database



Interactive Visual Exploration of Spatio-Temporal Urban Data Sets using Urbane

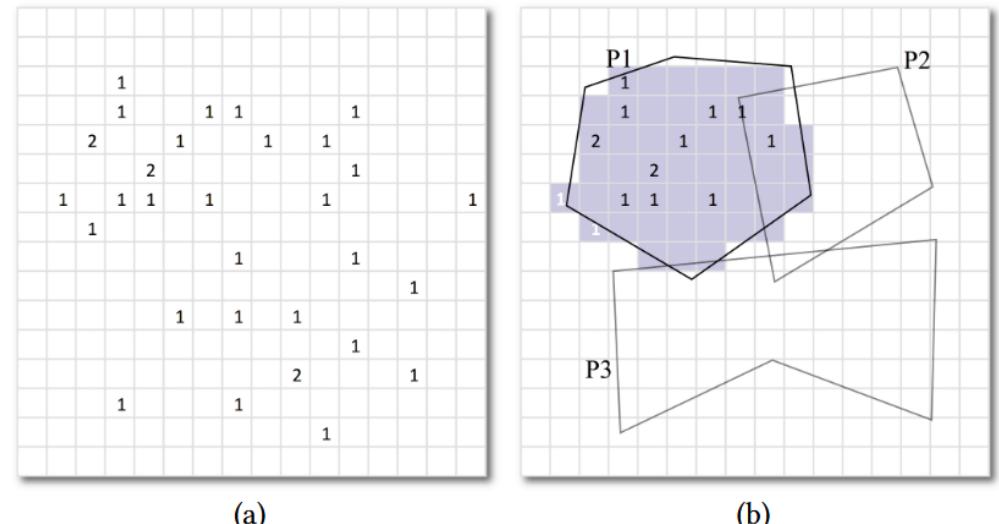
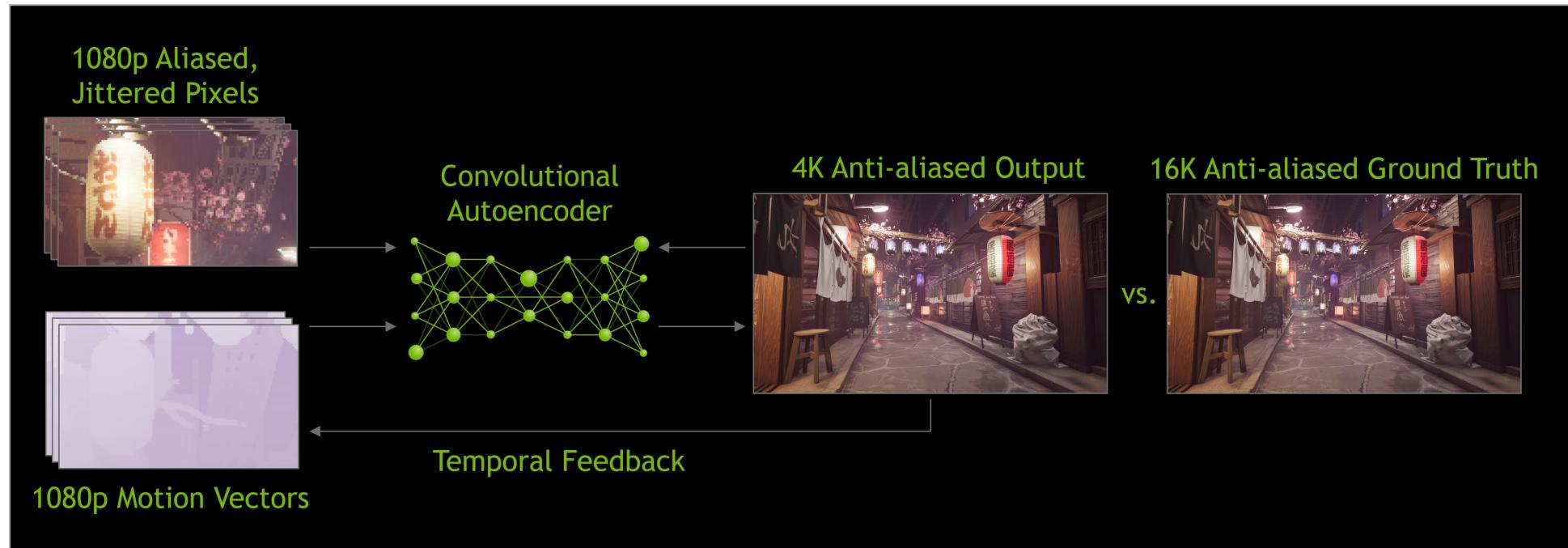


Figure 2: Raster Join first renders all points onto an FBO storing the count of points in each pixel (a). It then aggregates the pixel values corresponding to polygon fragments (b).

Computer Graphics & Machine Learning



NVIDIA: Deep learning super sampling

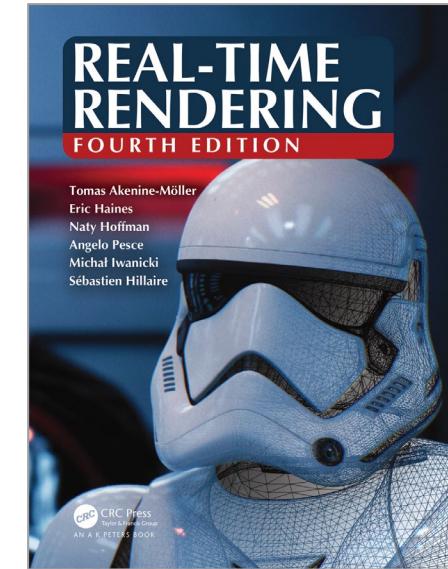
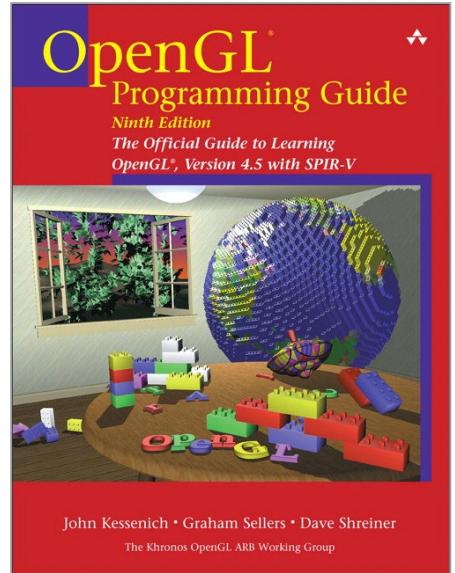
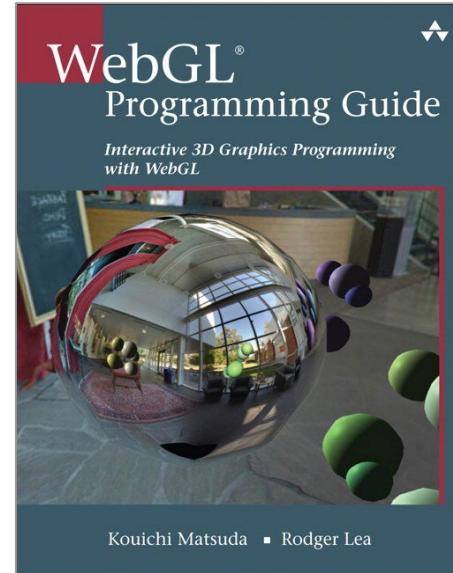
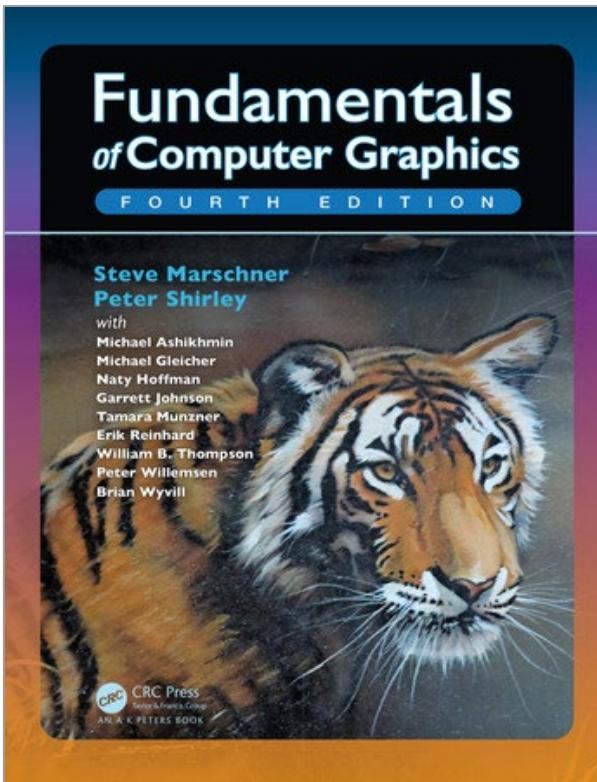
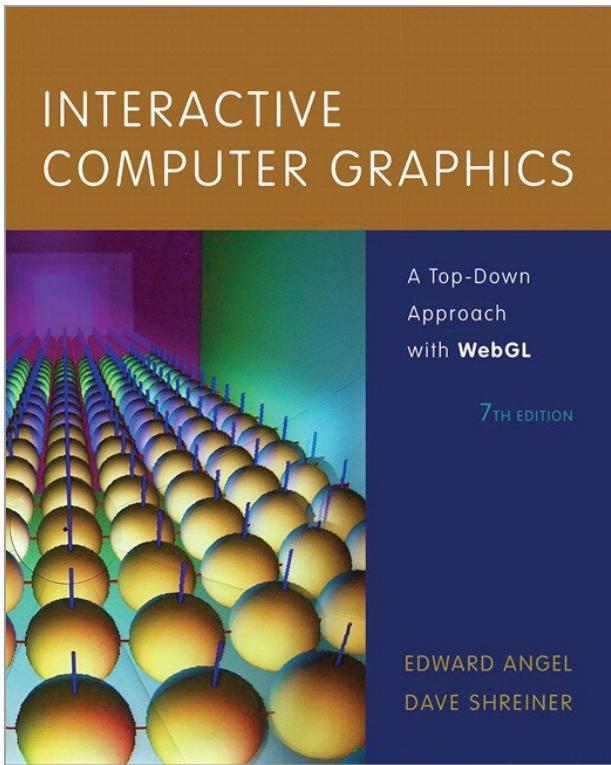
CS425: Course goals

- Explain and apply the core concepts of computer graphics.
- Understand and use the different components of a programmable graphics pipeline.
- Understand and use different spatial data structures.
- Implement visual effects such as shadows.
- Implement interactive computer graphics programs using WebGL.

CS425: Course goals

- Theory and systems behind applications
- Mathematics:
 - Physics of light, color, ...
 - Geometry, perspective, ...
- Systems:
 - Graphics APIs
 - Interaction devices

Material



Other resources

<https://open.gl>

<https://webgl2fundamentals.org/>

<https://songho.ca/opengl/>

<https://learnopengl.com/>

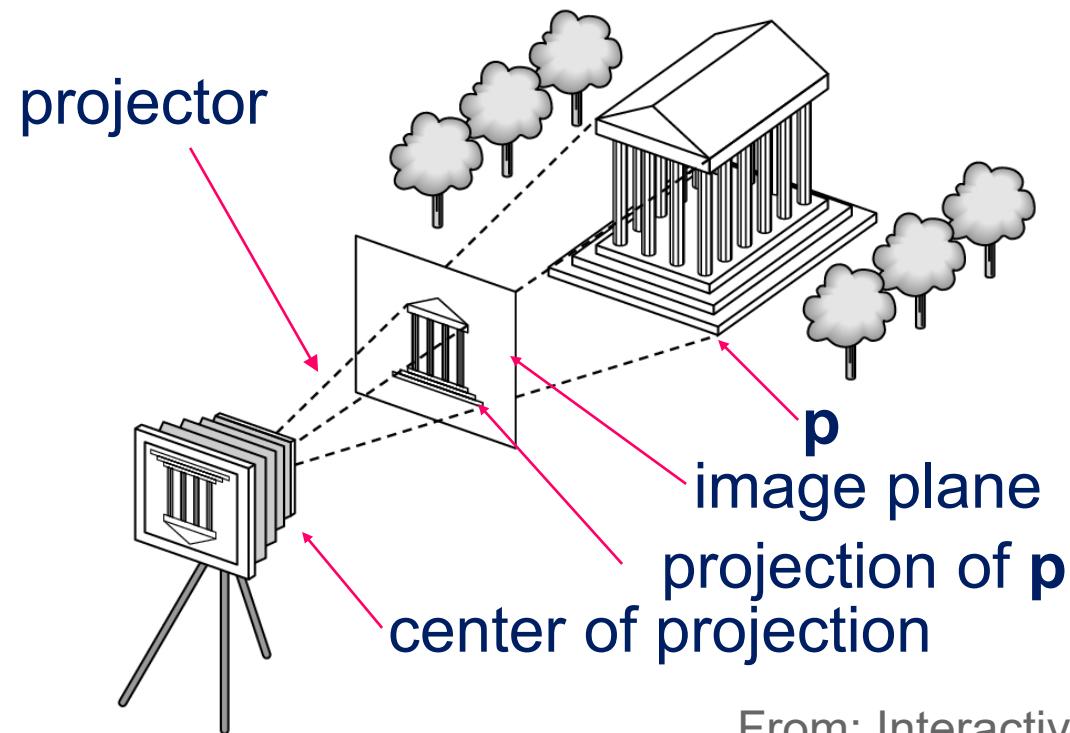
Requirements

- CS251 Data structures.
- We will use WebGL and JavaScript.
 - Portability
 - No need to install dependencies or plugins to run code: just open a browser.
 - Polyglot programming: JavaScript, GLSL, HTML, CSS, ...
 - Easy to produce interactive examples.

Content overview

- Images and color
- WebGL
- Linear algebra and transformation
- Viewer transformations and rasterization
- Graphics pipeline
- Lighting and shading
- Texture mapping
- Shadows
- Ray tracing
- Antialiasing
- Curves and surfaces
- Spatial data structures
- Modern rendering techniques

Images and color



From: Interactive Computer Graphics 7th Ed by
Professor Ed Angel and Dave Shreiner

WebGL & web programming



Click and drag to rotate teapot.

Original demo from [OSD](#).

Quisque posuere malesuada elementum. Etiam tortor purus, eleifend sit amet mattis vitae, ultrices vitae lorem. Nullam sodales risus eu nisi mollis ullamcorper. Morbi at nisl mauris. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Donec et eros lobortis sem lobortis luctus eget at arcu. In feugiat mollis purus in lobortis. Suspendisse sed nulla id augue dictum vulputate. Fusce lectus eros, dictum ac hendrerit sit amet, laoreet ultrices leo.

Morbi pellentesque, metus sit amet auctor convallis, massa lectus interdum dui, vitae auctor diam sapien vel metus. Suspendisse faucibus, erat pellentesque tristique egestas, mi justo porta velit, vitae ornare mauris metus sit amet diam. Praesent posuere dapibus eleifend. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean lorem neque, adipiscing eget egestas eget, semper rutrum velit. Praesent blandit tempor convallis. Ut scelerisque, magna pharetra consequat molestie, est arcu convallis justo, sed volutpat sem nisl vel ipsum. Donec euismod risus ut nibh accumsan eget rhoncus tellus vehicula. Etiam vulputate lorem id odio tempus quis suscipit sapien sollicitudin. Duis fringilla hendrerit elit, eget fringilla lectus consequat quis. Nunc suscipit sapien id lorem vel interdum. Sed lectus diam, mattis sed feugiat eget, ullamcorper et velit. Fusce placerat, nisi eget feugiat volutpat, leo diam porta velit, eget volutpat risus mauris sed velit.

Vestibulum quam augue, vehicula ut congue nec, pulvinar in risus. Morbi eget odio potenti. Phasellus venenatis, leo et accumsan ullamcorper, orci tortor hendrerit magna turpis purus. Phasellus accumsan odio lacina nisl auctor eget tempor nunc euismod, ullamcorper massa ultricies eget fermentum quam consequat. Proin augue nibh, dignissim nisi. Nulla convallis aliquam est, eu interdum metus malesuada non. Sed in hac elementum tellus, et mollis nibh sapien luctus dolor. Praesent egestas purus

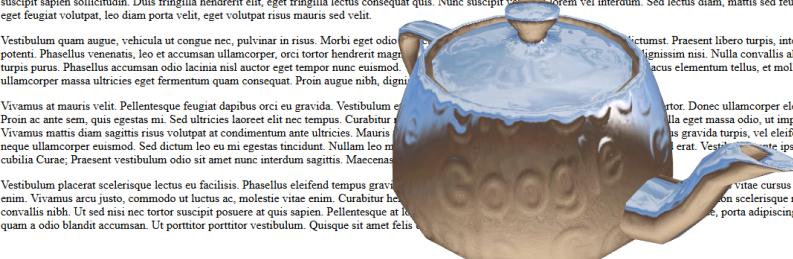
tempor. Donec ullamcorper eleifend magna, sit amet ultricies quam placerat ut. Nulla eget massa odio, ut imperdiet lacus. Sed venenatis bibendum faucibus. Vivamus gravida turpis, vel eleifend risus malesuada consectetur. Ut non massa non dolor. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere

Vivamus at mauris velit. Pellentesque feugiat dapibus orci eu gravida. Vestibulum enim. Vivamus ac ante sem, quis egestas mi. Sed ultricies laoreet elit nec tempus. Curabitur viverra, nisl euismod, nisl tincidunt turpis. Vivamus mattis diam sagittis risus volutpat at condimentum ante ultricies. Mauris neque ullamcorper euismod. Sed dictum leo eu mi egestas tincidunt. Nullam leo in cubilia Curae; Praesent vestibulum odio sit amet nunc interdum sagittis. Maecenas

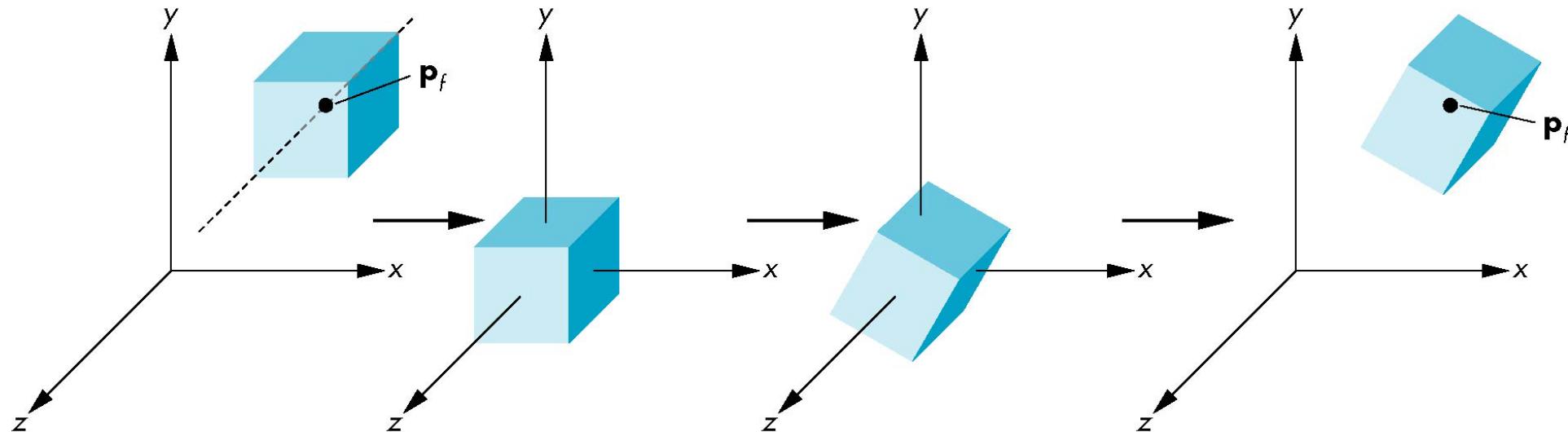
Vestibulum placerat scelerisque lectus eu facilisis. Phasellus eleifend tempus gravida, nisl euismod, nisl tincidunt turpis. Vivamus ac ante sem, quis egestas mi. Sed ultricies laoreet elit nec tempus. Curabitur viverra, nisl euismod, nisl tincidunt turpis. Vivamus mattis diam sagittis risus volutpat at condimentum ante ultricies. Mauris neque ullamcorper euismod. Sed dictum leo eu mi egestas tincidunt. Nullam leo in cubilia Curae; Praesent vestibulum odio sit amet nunc interdum sagittis. Maecenas

Vestibulum placerat scelerisque lectus eu facilisis. Phasellus eleifend tempus gravida, nisl euismod, nisl tincidunt turpis. Vivamus ac ante sem, quis egestas mi. Sed ultricies laoreet elit nec tempus. Curabitur viverra, nisl euismod, nisl tincidunt turpis. Vivamus mattis diam sagittis risus volutpat at condimentum ante ultricies. Mauris neque ullamcorper euismod. Sed dictum leo eu mi egestas tincidunt. Nullam leo in cubilia Curae; Praesent vestibulum odio sit amet nunc interdum sagittis. Maecenas

Vestibulum placerat scelerisque lectus eu facilisis. Phasellus eleifend tempus gravida, nisl euismod, nisl tincidunt turpis. Vivamus ac ante sem, quis egestas mi. Sed ultricies laoreet elit nec tempus. Curabitur viverra, nisl euismod, nisl tincidunt turpis. Vivamus mattis diam sagittis risus volutpat at condimentum ante ultricies. Mauris neque ullamcorper euismod. Sed dictum leo eu mi egestas tincidunt. Nullam leo in cubilia Curae; Praesent vestibulum odio sit amet nunc interdum sagittis. Maecenas

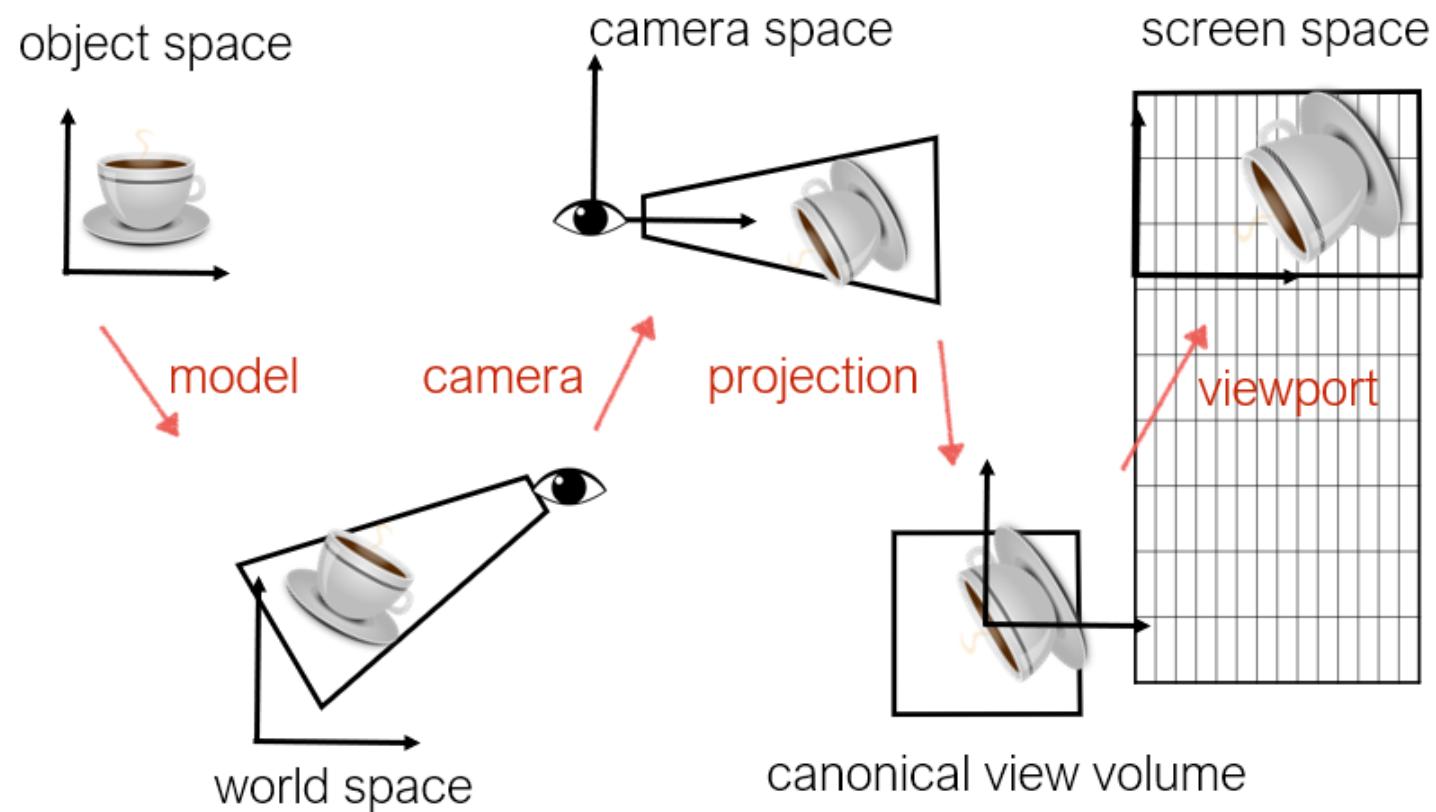


Linear algebra and transformations



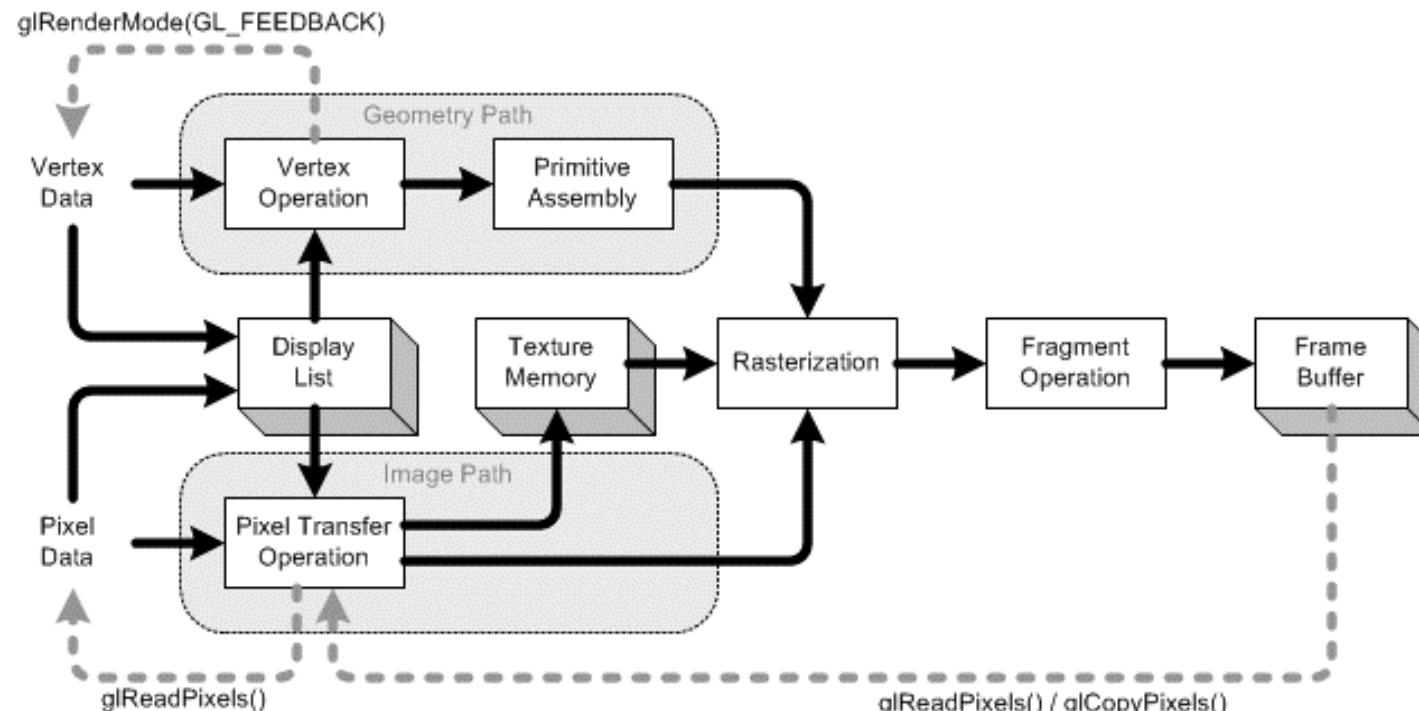
From: Computer Graphics by Professor Daniele Panozzo - NYU

Viewer transformations and rasterization



From: Computer Graphics by Professor Daniele Panozzo - NYU

Graphics pipeline



From: songho.ca

Lighting and shading

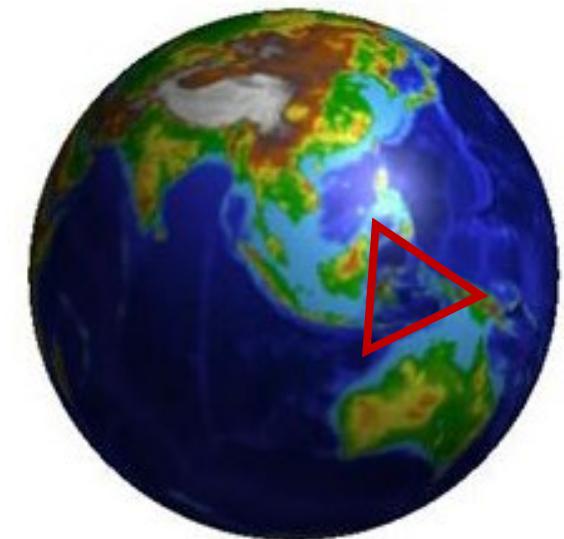
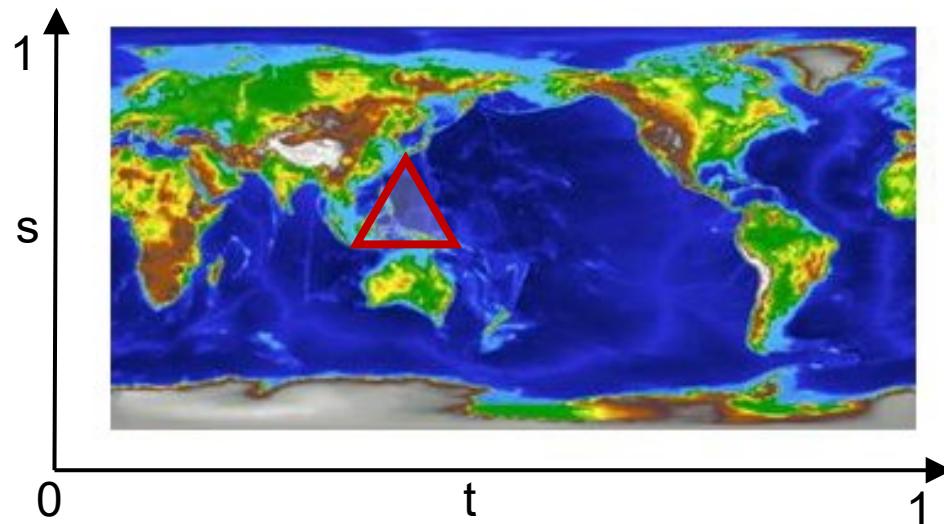
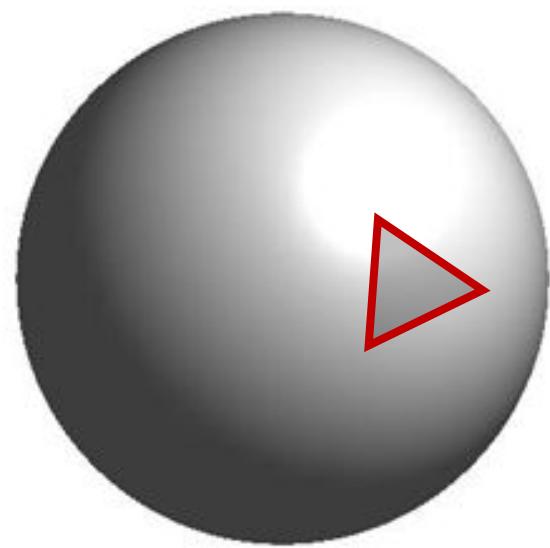


Diffuse

Ambient

Specular

Texture mapping



Build a mapping between the
texture and the object

Shadows

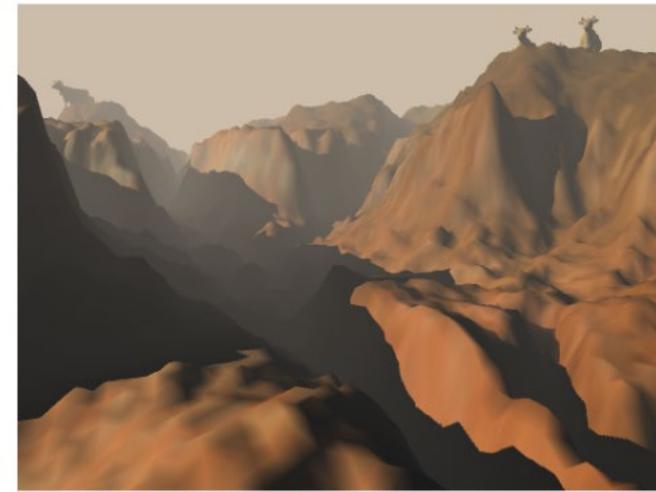
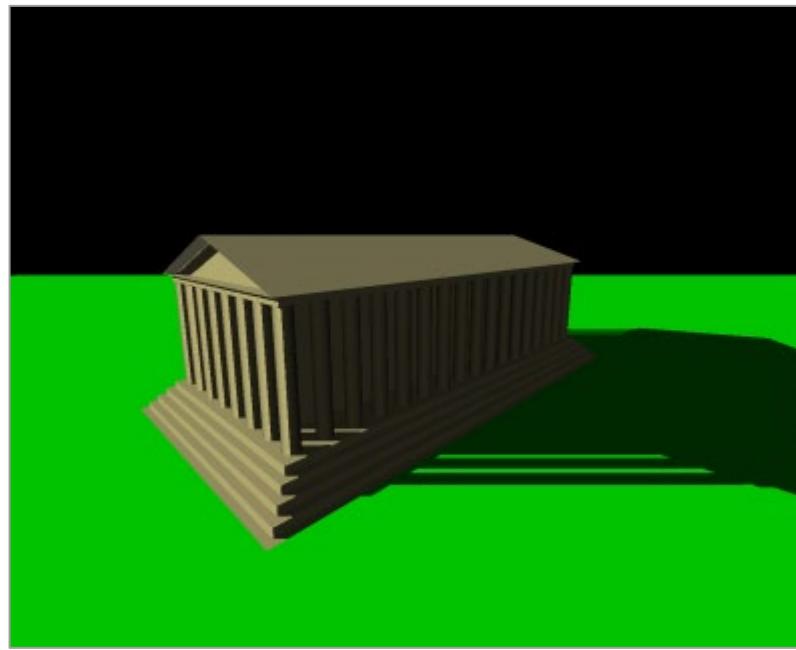
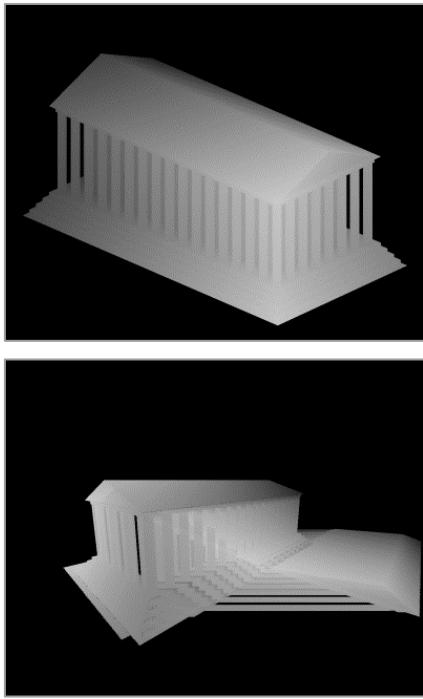


Figure 3-1. Large scale terrain rendering with 4-splits CSM

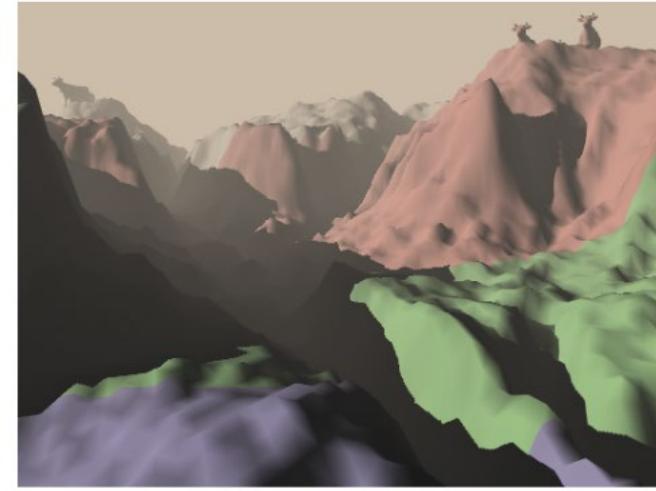
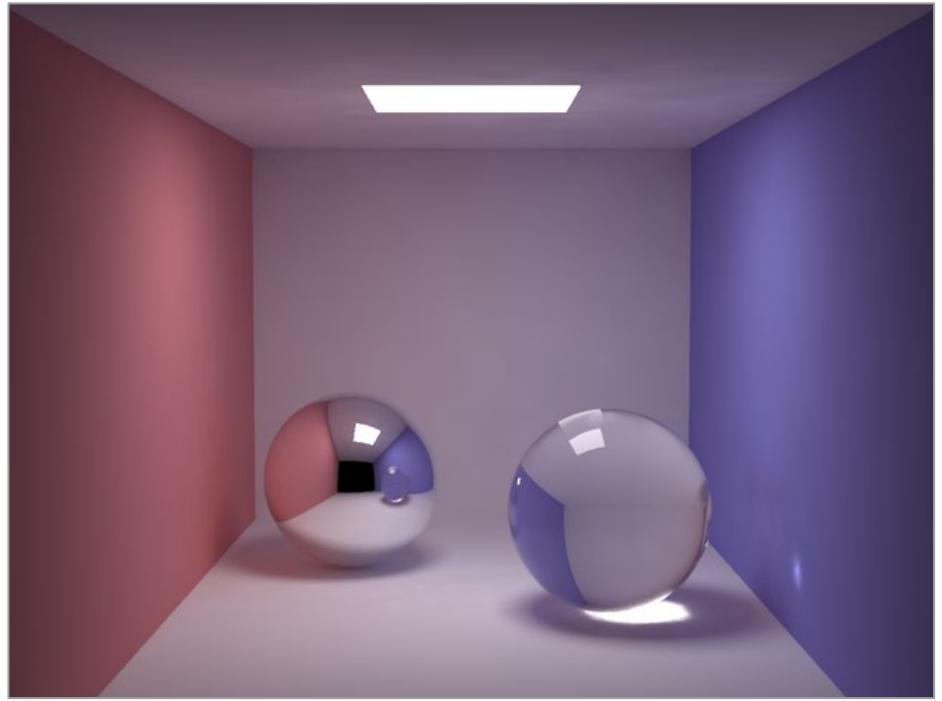
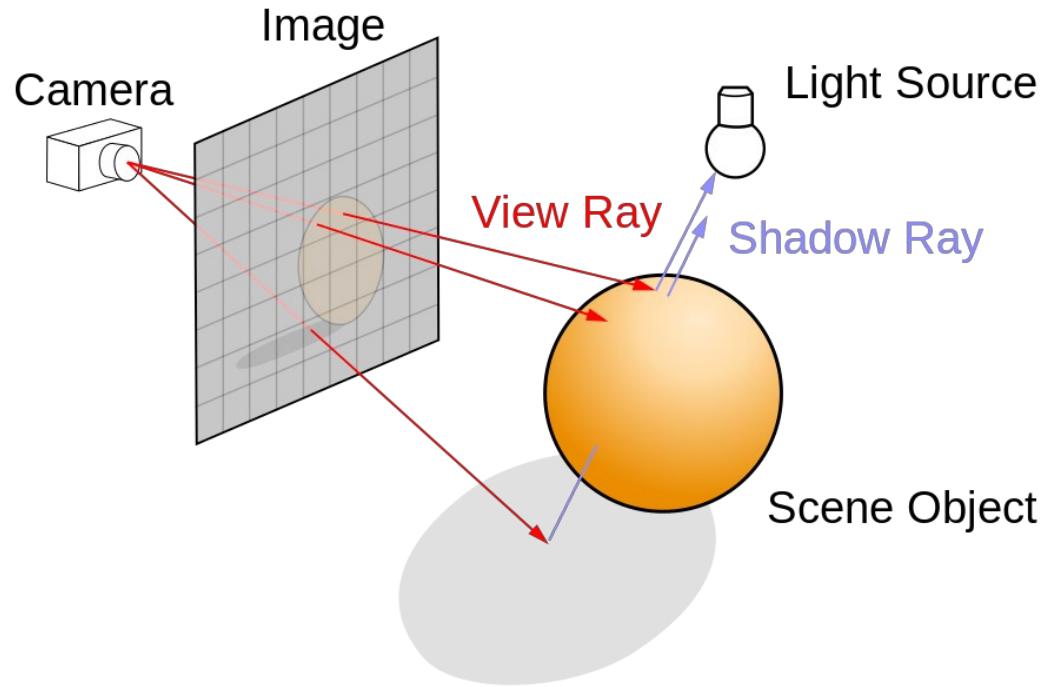


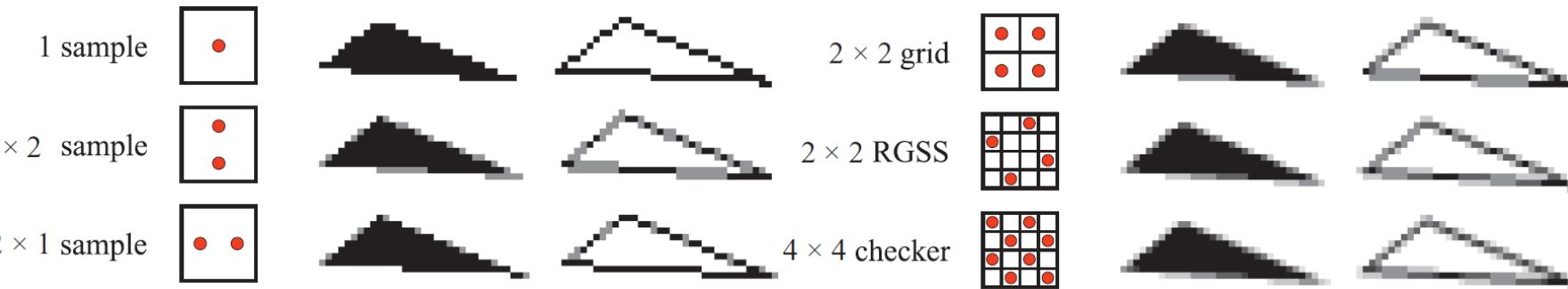
Figure 3-2. Texture look ups from different shadow maps are highlighted

Ray tracing

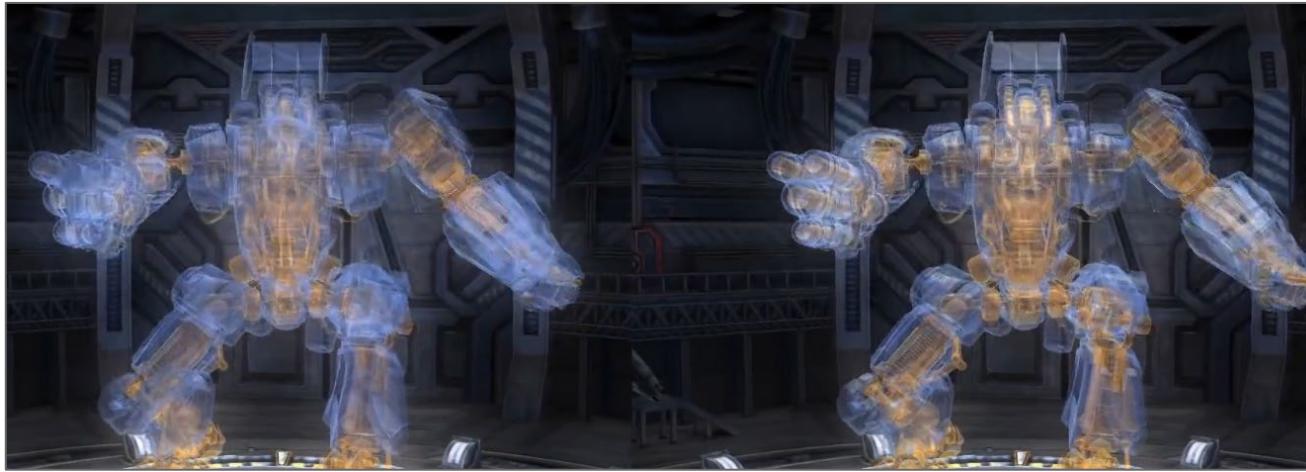


Visual appearance

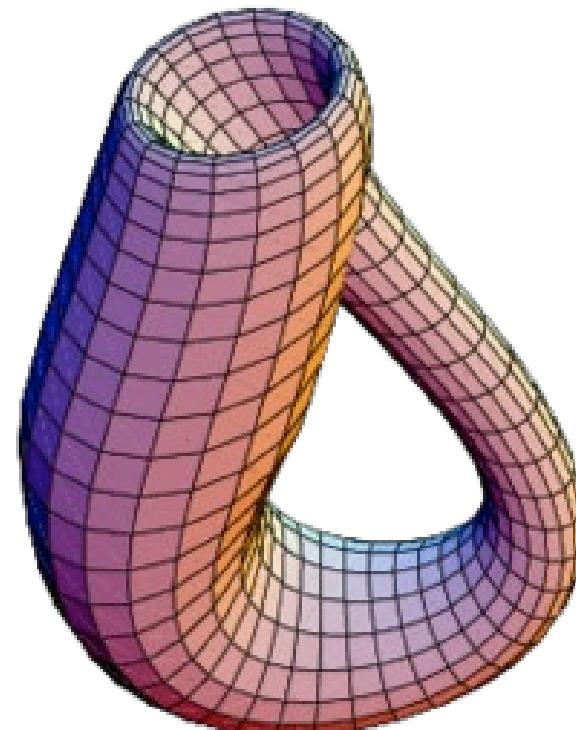
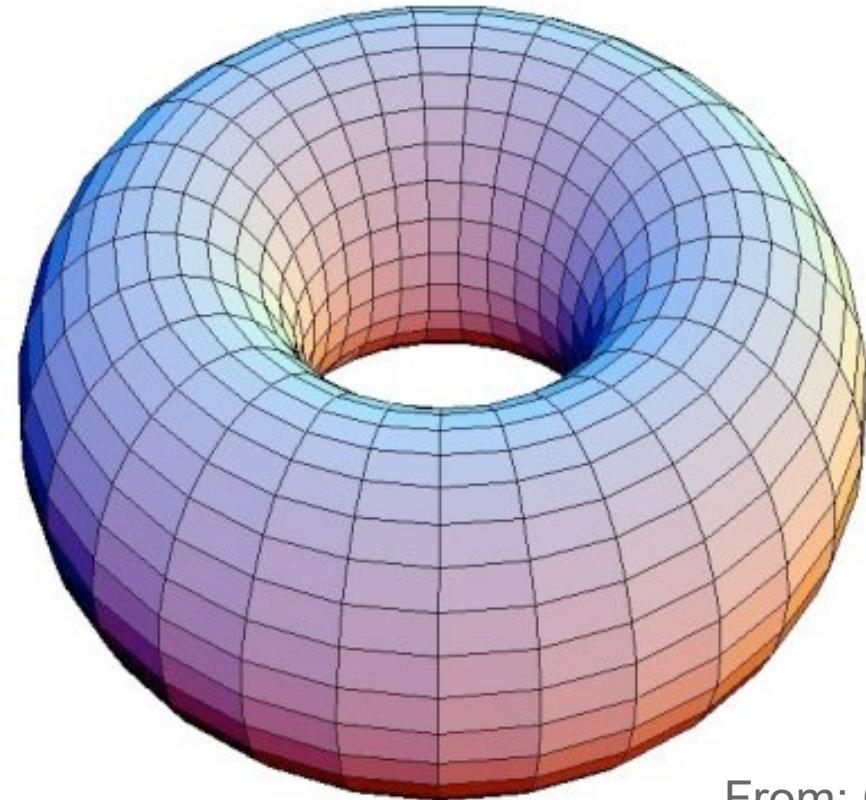
Antialiasing



Transparency

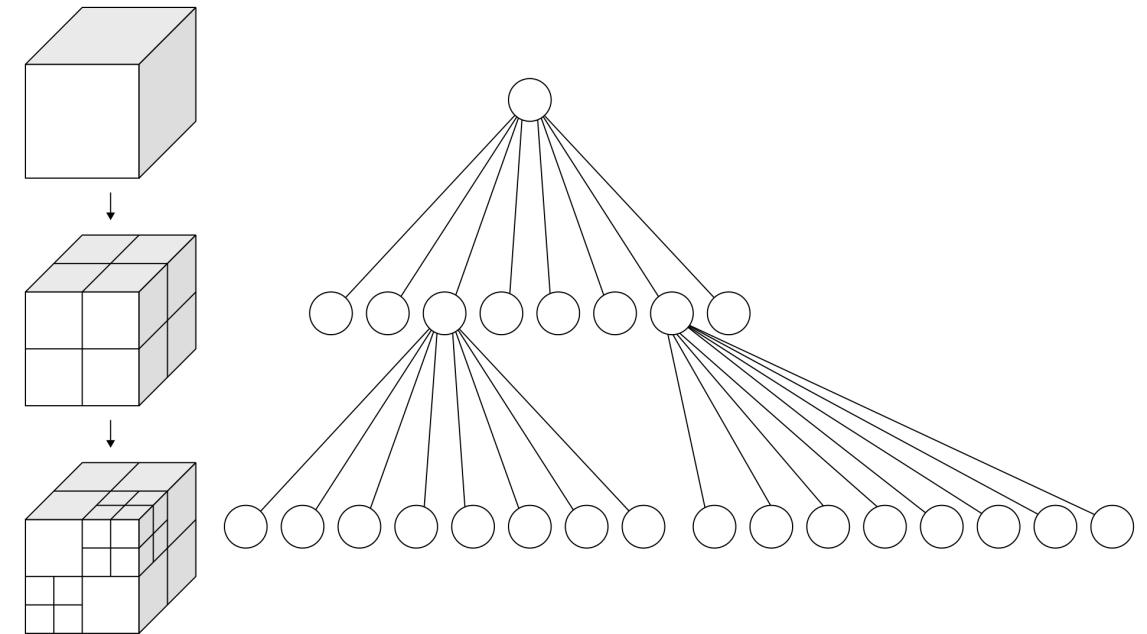
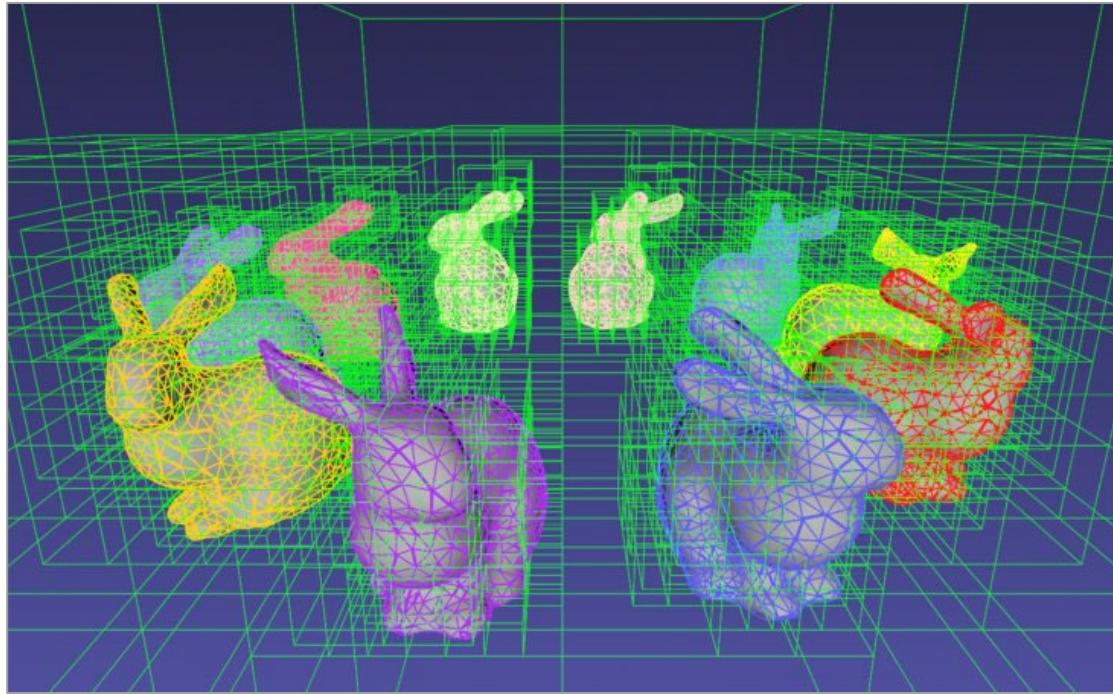


Curves and surfaces



From: Computer Graphics by Professor Daniele Panozzo - NYU

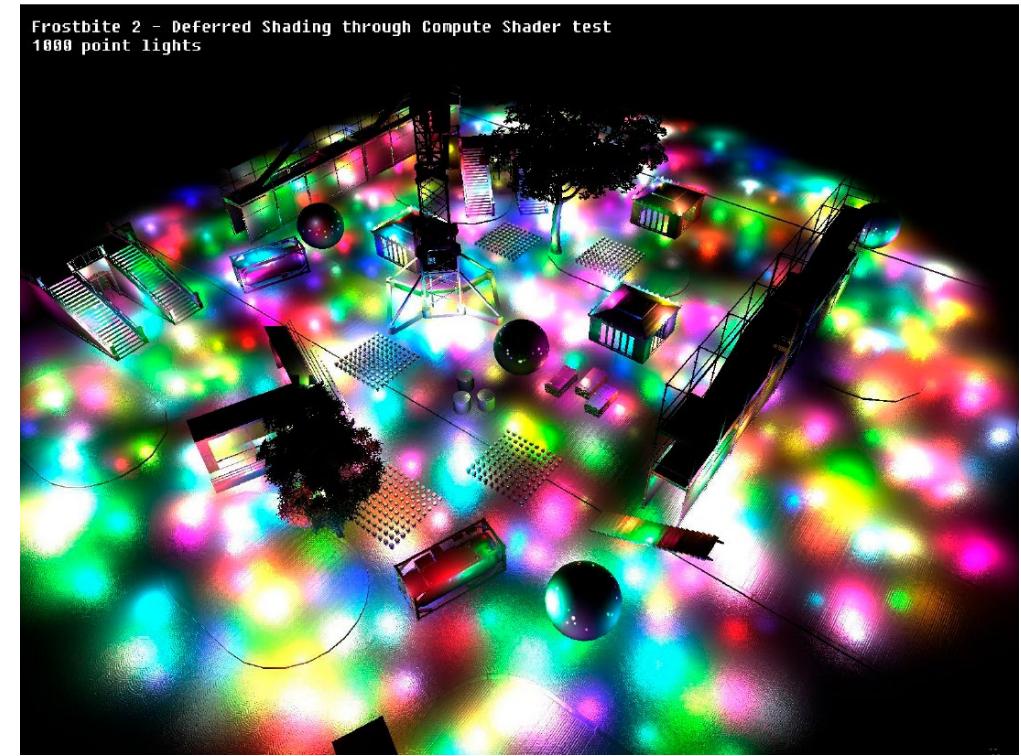
Spatial data structures



Modern rendering techniques



Ambient occlusion

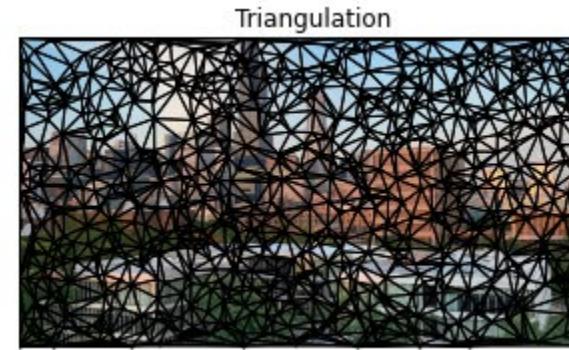
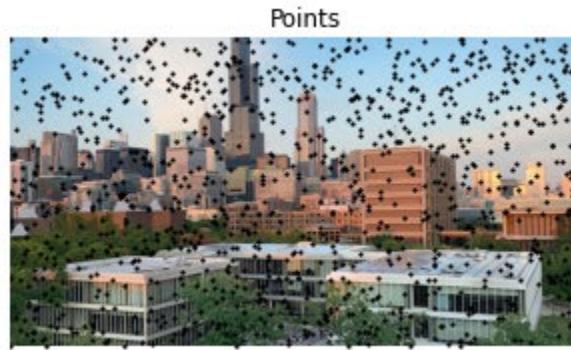


Deferred shading

Assignments

- Assignment 0: WebGL + Web environment
- Assignment 1: Triangle meshes rendering
- Assignment 2: Shading, texture and shadows
- Assignment 3: Ray tracing
- Assignment 4: Procedural meshes

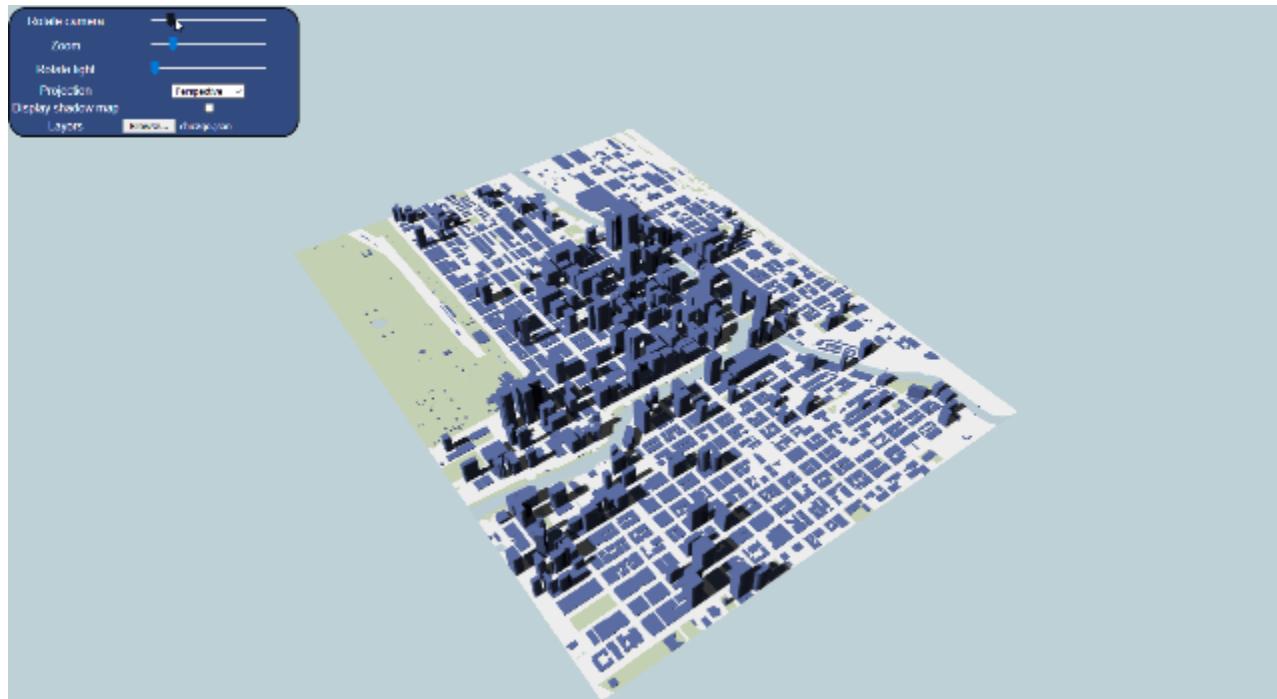
Assignment 0: Intro to JavaScript and WebGL



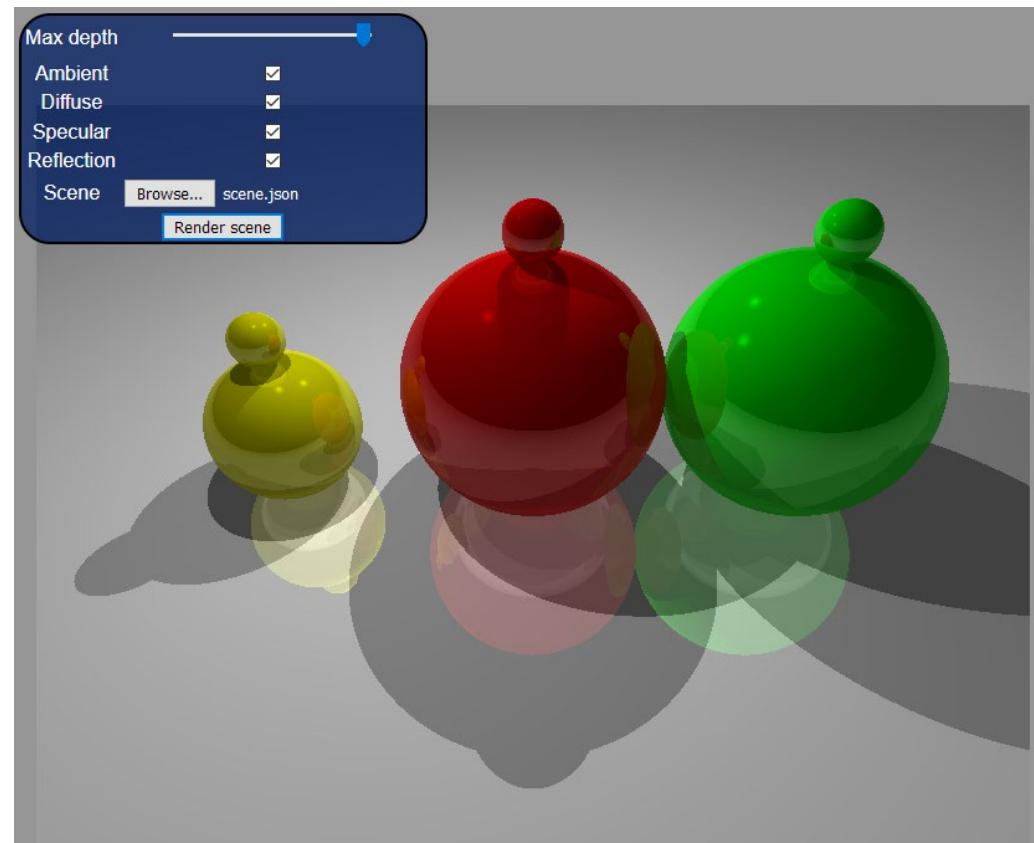
Assignment 1: Triangle meshes rendering



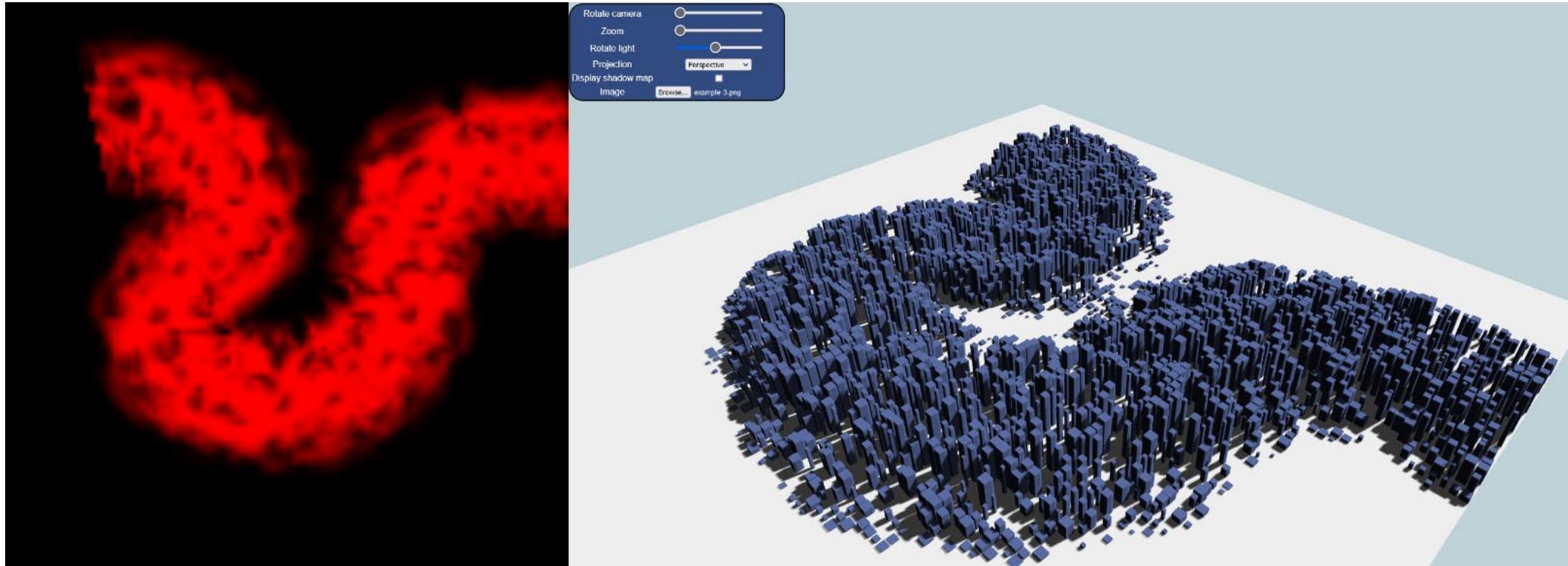
Assignment 2: Shadow maps



Assignment 3: Ray tracing



Assignment 4: Procedural meshes



Assignment considerations

- Create a GitHub project named **cs425-[term]-[year]**
- Inside this project create a folder named **assignment[X]** for each assignment.
- For each assignment you will hand in the following files:
 - index.html: main html file.
 - assignment[x].js: assignment main source code.

Grading

- Check syllabus.

Recommended workflow

Day 0: Read the assignment

Day 1-2: Familiarize yourself with the main topics of the assignment

Day 3-10: Code the different components of the assignment

Day 8-10: Start writing README.md file, make sure code runs on different computers

Day 11-12: Fix problems, double check README.md

Day 13: Submit

Assignment grading

- Example:

Grading

The code will be evaluated on Firefox. Your submission will be graded according to the quality of the image results, interactions, and correctness of the implemented algorithms. Your README.me file will also be graded.

To get a D on the assignment, your application should be able to load a JSON file in the format specified above, and visualize all layers using a perspective projection. To get a C on the assignment, you should implement the shadow map technique for a given light direction. To get a B, you must implement all interactions specified in the configuration panel (camera rotation, light rotation, perspective and orthographic projections). To get an A on the assignment, the application must be able to render the shadow map depth, and have a detailed readme file.

Assignment quizzes

- Assignments will be followed by short in-class quizzes; These quizzes will provide an opportunity for you to demonstrate your understanding of the basic concepts covered in the assignments.
- Assignment grades will be weighted by the scores of these short quizzes.

Assignment policies

- You are encouraged to discuss the assignments with your classmates, but collaboration in the assignment is not allowed.
- You are not allowed to copy code from online sources or use external libraries for any of the assignments (unless clearly specified in the assignment). Do not share implementation details or anything solution-oriented.

Final exam policies

- In-person exam.
- You can bring a one-page two-sided handwritten sheet, summarizing the content of the course.