CS5625 Interactive Computer Graphics

Steve Marschner

Spring 2019

01 Introduction

CD Projekt RED—*The Witcher 3* (2015)

Naughty Dog—*The Last of Us* (Remastered, 2014)

Rockstar Games—*Red Dead Redemption 2* (2018)

Valve—*Portal* (2007)

Ubisoft—*Child of Light* (2014)

Oculus VR

Oculus VR

Oculus VR

NASA

[John C. Stone, UIUC]

University of Calgary

Autodesk Revit Architecture 2015

How To Draw a Triangle, c. 1985

Transform vertices to screen coordinates

Find all the pixels covered by the triangle

Fill all the pixels with the triangle's color

How To Draw a Triangle, c. 1988

Perform lighting calculations to find vertex colors

Transform vertices to screen coordinates

Find all the pixels covered by the triangle

Fill all unoccluded pixels with the interpolated vertex colors and depth

How To Draw a Triangle, c. 1992

Perform lighting calculations to find vertex colors

Transform vertices to screen coordinates

Find all the pixels covered by the triangle

Look up a texture map value

Fill all unoccluded pixels with a function of the texture and the interpolated vertex colors, as well as the depth

How To Draw a Triangle, c. 1999

Perform elaborate lighting calculations to find vertex colors

Transform vertices to screen coordinates

Find all the pixels covered by the triangle

Look up a value from one or more 1D, 2D, or 3D texture maps

Fill all unoccluded pixels with a complicated, adjustable function of the textures and the interpolated vertex colors, as well as the depth

Pixar—*Ratatouille* (2007)

How To Draw a Triangle in 2001

Execute a vertex program over all the vertices

Find all the pixels covered by the triangle

Execute a fragment program over all those pixels

Fill all unoccluded pixels with the resulting color and depth

How To Draw a Triangle in 2007

Execute a vertex program over all the vertices

Execute a geometry program over all primitives

Find all the pixels covered by the triangle

Execute a fragment program over all those pixels

Fill all unoccluded pixels with the resulting color and depth

How To Draw a Triangle in 2010

Execute a vertex program over all the vertices

Execute tesselation programs to refine primitives

Execute a geometry program over refined primitives

Find all the pixels covered by the triangle

Execute a fragment program over all those pixels

Fill all unoccluded pixels with the resulting color and depth

How To Draw a Triangle in 2018

Execute a vertex program over all the vertices

Execute tesselation programs to refine primitives

Execute a geometry program over refined primitives

Find all the pixels covered by the triangle

Execute a fragment program over all those pixels

Fill all unoccluded pixels with the resulting color and depth

Integrate rasterized results with GPU ray tracing as needed

Project Sol: A Real-Time Ray-Tracing Cinematic Scene Powered by NVIDIA RTX
Watch later
Share
MORE VIDEOS
Play (k)
2:17 / 2:25
YouTube

Development of Hardware Capabilities

Workstation era

- ’85–’87: transform and render flat-shaded points, lines, polygons (no z buffer)
- ’88–’91: transform, light, and render smooth shaded polygons
- ’92–: transform, light, and render texture-mapped, antialiased polygons

PC era

- ’95–’98: render texture-mapped polygons
- ’99–’00: transform, light, and render texture-mapped, antialiased polygons
- ’01–’06: execute vertex and fragment shaders over antialiased polygons
- ’07–’09: execute vertex, geometry, and fragment shaders over antialiased polygons
- ’10–: execute vertex, geometry, tesselation, and fragment shaders over antialiased polygons

[Stewart & Lempel JPCS 2017]

SGI RealityEngine Architecture (1992)

Figure 1. Board-level block diagram of an intermediate configuration with 8 Geometry Engines on the geometry board, 2 raster memory boards, and a display generator board.

SGI InfiniteReality Architecture (1996)

NVIDIA G70 Architecture (2005)

[NVIDIA Corporation]

NVIDIA G80 “Tesla” Architecture (2007)

[NVIDIA Corporation]

AMD Radeon HD 2900 “TeraScale” Architecture (2007)

NVIDIA GK104 “Kepler” Architecture (2012)

[NVIDIA Corporation]

NVIDIA GM204 “Maxwell” Architecture (2014)

[NVIDIA Corporation]

NVIDIA GP100 “Pascal” Architecture (2016)

More general-purpose design

- good performance for more datatypes
- architecture driven by ML and sci comp as well as graphics

[NVIDIA Corporation]

NVIDIA GU102 “Turing” Architecture (2018)

[NVIDIA Corporation]

Compute cores becoming more heterogeneous, ray tracing acceleration support

What is Graphics?

- Generating images!

- Create them
  - Modeling
  - Animation
  - Rendering
- Manipulate them
  - Imaging

slide courtesy of Kavita Bala, Cornell University

What is in 5625?

- Generating images
- Modern Graphics Pipeline
- Create them
  - Modeling
  - Animation
  - Rendering
- Manipulate them
  - Imaging
- Focus on **performance**

slide courtesy of Kavita Bala, Cornell University

Modeling

Subdivision Surfaces

LOD: Level-of-Detail

LOD: Level-of-Detail

Animation

Morph Targets

José Alves da Silva—*Corlyorn Family* (Vodafone campaign)

Game Programming Wiki (gpwiki.org)

Panozzo & Jacobson, libigl tutorial (libigl.github.io/libigl)

Rendering

Rob Cook's vases

Source: Cook, Torrance 1981

slide courtesy of Kavita Bala, Cornell University

Texture Mapping

- Bump Maps
- Normal Maps
- Environment Maps
- Irradiance Maps
- ....

Displacement Maps

slide courtesy of Kavita Bala, Cornell University

Filtered Environment Mapping

- Environment map → radiance
- Filter this map → irradiance (diffuse lighting)
- Fast diffuse and ambient (just a lookup, or eqn)

slide courtesy of Kavita Bala, Cornell University

Shadow Algorithms

- Crucial for spatial and depth perception

slide courtesy of Kavita Bala, Cornell University

Shadow Maps

- Introduced by Lance Williams (SIGGRAPH 1978)
- Render scene from light's view
  - black is far, white is close

slide courtesy of Kavita Bala, Cornell University

Shadow Volumes

- Clever counting method using stencil buffer
- Can cast shadows onto curved surfaces

Mark Kilgard, NVIDIA Inc.

slide courtesy of Kavita Bala, Cornell University

Soft Shadows

- Soft shadows appear more natural
  - Hard to get soft shadows in hardware

slide courtesy of Kavita Bala, Cornell University

Ambient Occlusion

slide courtesy of Kavita Bala, Cornell University

Imaging

Modeling flare in a camera lens

Hullin et al. SIGGRAPH 2011

Anti Aliasing of textures

slide courtesy of Kavita Bala, Cornell University

slide courtesy of Kavita Bala, Cornell University

Tone reproduction operator

Modeling flare in the eye

Greger et al. SIGGRAPH 2005

Projects

CS 5625 Coursework

5 projects (optionally in pairs)

- mostly implementation, some written
- Primarily C++ and OpenGL
- anticipated topics: shading, texturing, shadows, antialiasing/filtering, physics simulation

Midterm exam

Final project (groups of 2–4)

- project proposal
- milestone presentations and evaluation
- final project demos, presentations, writeup

Sampling and filtering

Texture antialiasing

Shading

CS 5625 Programming Assignment 1
Scene:
2. Material test scene
Renderer:
Forward renderer
Buffer:
Uber shader output

Deferred shading

Mesh animation

Shadow Mapping

Shadow Mapping

Soft shadows

Soft shadows

Physics

Final project examples

Spring 2015

Natalie Diebold, Hani Altwaijry | *Portal-ish*

Natalie Diebold, Hani Altwaijry | *Portal-ish*

Natalie Diebold, Hani Altwaijry | *Portal-ish*

Eridanus Collazo, John Carlos, Zeqiang Zhao | *Solarcraft*

- MOTION BLUR
- LENS FLARE

- MOTION BLUR
- LENS FLARE

Victoria Dye, Joshua Reichler | *White-Out 2k17: The Snowening*

Fight your way through the zombie fairies to rescue Orin!

Fight your way through the zombie fairies to rescue Orin!

Ari Karo, Christopher Yu, Jonathan Behrens, Jeremy Cytryn | *Subterranean Arsonism*

About CS5625

Prereqs

- introductory graphics course (e.g. 4620) or instructor permission

Dissemination

- website www.cs.cornell.edu/Courses/cs5625
  - schedule (very much subject to change!)
  - announcements, updates
  - lecture slides, notes, readings
- CMS
  - homeworks, lecture notes
- Piazza
  - discussion, questions

Academic Integrity

Don’t copy code from Web without careful attribution

- small snippets of, e.g. OpenGL boilerplate OK **with attribution**

Collaboration only when projects/homeworks are with groups

Always cite sources of code and ideas

- think carefully about who and what contributed to your work
- if you tell me what is going on, there is never any AI problem

Recommended texts

Real-time Rendering

- Akenine-Moller, Haines, Hoffman
- available via library

Other books

- many listed on website

Online resources

- lots of them!

Other Policies

Late policy

- 5 free late days over the semester
  - in total over all assignments
  - not for the final project
- Otherwise, 10% penalty per day for up to 5 days
  - after that, 0

Final project

An interactive 3D game with fancy graphics

Open ended, needs to have technically impressive results

Ways to impress

- rendering: shading, shadows, global illumination, …
- modeling: splines, subdivision surfaces, procedural generation, …
- animation: particle systems, character motion, collision detection, …
- imaging: flare, antialiasing, …

Focus is on graphics, not gameplay