Final Exam Review

When / Where

Thursday, 12 December 2024 **7:30 (AM)** CST.

Plan to have breakfast with a dependable person who has your phone number.

Conditions

Closed Book, Closed Notes

Bring one sheet of notes (both sides), 216 mm × 280 mm.

No use of communication devices.

Format

Several problems, short-answer questions.

fr-2

Resources

Lecture "slides" used in class: `https://www.ece.lsu.edu/koppel/gpup/ln.html`

Solved tests and homework:. `https://www.ece.lsu.edu/koppel/gpup/prev.html`

It's important to study the solutions.

fr-2

fr-3

Study Recommendations

Study this semester's homework assignments. Similar problems may appear on the exam.

Solve Old Problems—memorizing solutions **is not the same** as solving.

Following and understanding solutions **is not the same as** solving.

Use the solutions for brief hints and to check your own solutions.

fr-3

fr-4

Mathematics

See `https://www.ece.lsu.edu/koppel/gpup/slides/set-1-math.pdf`.

Coordinates, Points, Vectors, Homogeneous Coordinates

Dot and Cross Products

Line / Plane Intercept

Transformations

Projections

fr-4

fr-5

Coordinate and Vector Classes

pVect, pCoor, pNorm, pMatrix

Use these for basic computations.

fr-5

fr-6

Simple Physical Simulation.

Understand how world modeled.

Point masses, ideal springs, gravity field.

Time Step

Updating velocity and position.

Forces

Gravity.

Ideal spring.

Simple Collisions.

fr-6

fr-7

Coordinate Spaces

OpenGL and OpenGL Shading Language: World, Object, Eye, Clip, Window

Vulkan Rasterization: Clip, Window

Vulkan Ray Tracing: Object, World, Clip, Window

fr-7

fr-8

CPU-Only Rendering Pipelines

Purpose of the CPU-Only Pipelines

Rendering without OpenGL, Vulkan, or any other library!

Intended to illustrate what graphics acceleration libraries such as OpenGL and Vulkan do.

The CPU-only pipelines show all the code starting from object-space triangles . . .
. . . to the writing of the frame buffer.

Code in the `cpu-only` directory in the course repo.

fr-8

fr-9

CPU-Only Rasterization

Very simple CPU-only rasterization:
`https://www.ece.lsu.edu/koppel/gpup/code/cpu-only/demo-02-raster.cc.html`

CPU-only code using multiple coordinate spaces:
`https://www.ece.lsu.edu/koppel/gpup/code/cpu-only/demo-03-coord-space.cc.html`

Depth-buffering (z-buffering) and Lighting:
`https://www.ece.lsu.edu/koppel/gpup/code/cpu-only/demo-04-z-light.cc.html`

Rendering-Pipeline-Like API:
`https://www.ece.lsu.edu/koppel/gpup/code/cpu-only/demo-06-rend-pipe.cc.html`

CPU-Only Ray Tracing

CPU-Only ray tracing:
`https://www.ece.lsu.edu/koppel/gpup/code/cpu-only/demo-05-ray-tracing.cc.html`.

fr-9

fr-10

Vulkan Rasterization Rendering Pipeline

See `https://www.ece.lsu.edu/koppel/gpup/slides/set-3-rend-pipe.pdf`

See `https://www.ece.lsu.edu/koppel/gpup/slides/nset-4-vulkan-rp.pdf`

The Stages: Vertex, Geometry, Fragment

Types of data accessible by shaders.

Fixed Functionality v. Programmable Stage.

fr-10

fr-11

Shader Data Access

See "Shader Data Access" in `https://www.ece.lsu.edu/koppel/gpup/slides/set-3-rend-pipe.pdf`

Types of Data:

Shader Input. (`in`) – Data for a particular vertex, primitive, fragment.

Shader Output (`out`) – Data for a particular vertex, primitive, fragment.

Uniform (`uniform`) – Small amount of read-only data. Same for every vertex, primitive, fragment.

Buffer (`buffer`) – Large amount of data.

fr-11

fr-12

Shader Programming

Reference: recent editions of the "OpenGL Programming Guide," Shreiner, *et al.*

Programmable Shaders

Vertex, Geometry, Fragment.

For Each One:

Inputs, Outputs.

Conventional functionality.

EE 4702-1 Lecture Transparency. Formatted 15:10, 9 December 2024 from lslifr.
fr-12

fr-13

Vulkan Primitives and Vertex Specification

Primitives (Primitive Topology)

Vulkan 1.3 Section 21.1 (`https://www.khronos.org/registry/vulkan/specs/1.3-extensions/html/chap21.html`)

- `vk::PrimitiveTopology::eTriangleList`
- `vk::PrimitiveTopology::eTriangleStrip`
- `vk::PrimitiveTopology::eLineList`
- `vk::PrimitiveTopology::eLineStrip`
- `vk::PrimitiveTopology::ePointList`

Vertex Shader Inputs.

Vertex (coordinate), color, normal, etc.

Estimate amount of data needed.

fr-13

fr-14

Vulkan Rasterization Pipelines; Storage Buffers

Pipeline Setup (Using Course Helpers)

See `https://www.ece.lsu.edu/koppel/gpup/slides/nset-4-vulkan-rp.pdf`

For each: Where do vertex shader inputs come from?

Difference between array on CPU (client) and Vulkan buffer.

Difference between buffers used as uniforms and storage.

Estimate amount of data sent between CPU and GPU.

fr-14

fr-15

Vulkan and OpenGL Textures

See `https://www.ece.lsu.edu/koppel/gpup/code/vulkan/demo-08-texture.cc.html`

Basic Idea

Texture Filtering: Minification, Magnification, mipmap levels.

Linear/Nearest

Texture application.

fr-15

fr-16

Ray Tracing

References

See `https://www.ece.lsu.edu/koppel/gpup/slides/set-5-ray-tracing.pdf`.

See `https://www.ece.lsu.edu/koppel/gpup/2022/nv-rt.txt`.

fr-16