Introduction to Real-time Ray Tracing Part 2

Going fast: Parallelizing Your Ray Tracer

Slides: Chris Wyman
Principal Research Scientist
NVIDIA

Narration: Eric Shaffer, University of Illinois

SOME PRELIMINARIES

Ideas needed before GPU ray tracing

YOU JUST GOT THE BASICS

http://rtintro.realtimerendering.com/

INTRODUCTION TO REAL-TIME RAY TRACING

SIGGRAPH 2019 Course
Peter Shirley, Chris Wyman, and Morgan McGuire
NVIDIA

Ray tracing is a fundamental topic in computer graphics, and often the main subject of an introductory university course.

This course takes the audience from zero prior computer rendering knowledge to an understanding of modern Monte Carlo path tracing by focusing on a real-time, parallel approach. We use concrete examples in a variety of APIs and programming languages, and provide source code.

Examples of images that attendees will understand how to render after each third of the course.

1 Syllabus

1.1 Slides

Section	Duration	Presenter	Download
1. Course Overview	65 min	Shirley	PDF • PPT
2. Going fast: Parallelizing your ray tracer	50 min	Wyman	PDF • PPT
3. Production-scale real-time ray tracing	60 min	McGuire	PDF • PPT

YOU JUST GOT THE BASICS

- But additional features ***expected*** for GPU rendering

YOU JUST GOT THE BASICS

- But additional features ***expected*** for GPU rendering
  - Typically, increased complexity; not just a few primitives

YOU JUST GOT THE BASICS

- But additional features ***expected*** for GPU rendering
  - Typically, increased complexity; not just a few primitives
  - Render triangle meshes
  - Just collections of triangles approximating 3D shapes
  - Easy enough; intersect each triangle in turn

YOU JUST GOT THE BASICS

- But additional features ***expected*** for GPU rendering
  - Typically, increased complexity; not just a few primitives
  - Render triangle meshes
  - Just collections of triangles approximating 3D shapes
  - Easy enough; intersect each triangle in turn
  - Mesh files usually contain material information
  - Often small-scale detail stored in textures

HOW TO HANDLE MATERIALS AND TEXTURES?

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color
    - Texture coordinates
    - Material parameters
    - Et cetera

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color
    - Texture coordinates
    - Material parameters
    - Et cetera

Our texture:

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color
    - Texture coordinates
    - Material parameters
    - Et cetera

Triangle vertices have:
texture coordinates

(0,0) (1,0)

(0,1)

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color
    - Texture coordinates
    - Material parameters
    - Et cetera

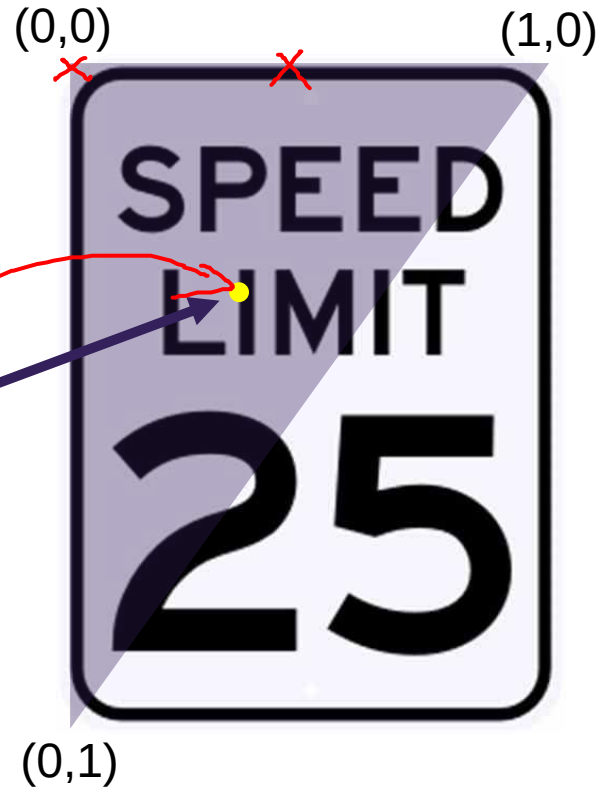
Triangle vertices have:
texture coordinates

Coordinate here:
Interpolates coordinates at vertices

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color
    - Texture coordinates
    - Material parameters
    - Et cetera

Triangle vertices have:
texture coordinates



Coordinate here:
Interpolates coordinates at vertices

Same interpolation as position, normal, color, etc.

Use coord to index in the image array

HOW TO HANDLE MATERIALS AND TEXTURES?

- Ray-primitive intersection
  - Not just binary: Did we hit? Yes / No
  - Also need to store ***attributes*** at the hit point, e.g.:
    - Positions
    - Normal
    - Color
    - Texture coordinates
    - Material parameters
    - Et cetera
  - All attribute interpolation work the same way