Overview of Ray Tracing "Pipeline"

Updated 15:33, 3 December 2021

Start With

The usual transformation and projection matrices.

Light locations.

Geometry:

Triangles.

Shader-computable shapes. (E.g., true-sphere shader.)

Preparation

Prepare Acceleration Structures

These describe where geometry is.

Assemble a ray tracing "pipeline".

Start

Host records:

```
CommandBuffer::traceRaysNV( shader_table, ...  width, height ..  );
```

Invokes the pipeline's Ray-Generation Shader width * height times.

A Ray

Like mathematical rays, has an origin (starting point) and a direction.

But NV Ray Tracing Rays also have:

A starting and ending point: `t_min`, `t_max`.

For a mathematical ray `t_min = 0`, and `t_max =` ∞.

But we are free to set `t_min = 0.1, t_max = 0.9`.

A payload. (A variable holding data shared by shaders.)

E.g., a color determined by what the ray intersected.

Flags. Specifies which shaders to use, etc.

E.g., `gl_RayFlagsOpaqueNV`. Don't bother with transparent geometry.

Cull Mask

An indicator of which geometry to consider.

Shader Stages – Quick Overview

Ray Generation

- The starting point.
- Casts rays at pixels.

Intersection

- Called during ray traversal.
- Determines if ray intersects geometry.

Closest Hit

- Called on closest geometry intersected by a ray.

Any Hit

- Called on any geometry intersected by a ray.

Miss

Called when a ray does not intersect anything.

Shader Data Access

Can access Uniform and Storage Buffers.

Ray Casting

Done by shaders. (Not host code.)

Cast by `traceNV( .., ray_origin, ..., ray_direction, ..  )`

Can be called by:

Ray Generation Shaders

Sets `ray_origin` to the eye location.

Aims `ray_direction` at a pixel.

Closest Hit Shaders, Any Hit Shaders

Sets `ray_origin` to hit location (position on geometry).

Might aim `ray_direction` at a light.

Might set `ray_direction` to reflected direction of incoming ray.

Calling traceNV starts Ray Traversal.

As a result of ray traversal shaders may be invoked. . .
. . . and those shaders may update the payload.

When traceNV returns the payload will have been updated by the invoked

shaders.

Ray Traversal

Calling traceNV starts Ray Traversal

Ray traversal is performed by the fixed functionality.

Ray traversal consists of finding geometry intersected by the ray.

Ray traversal uses acceleration structures to guide the search.

Intersection shaders are invoked during traversal.

Typically:

The geometry-specific closest-hit shader is called for the geometry closest to the ray origin.

Or the miss shader is called if no geometry is intersected.

Variations

Cull and ray flags can be used to limit geometry.

Ray Payload

Data carried by a ray, such as a color.

Sample declarations:

```
layout ( location = 0 ) rayPayloadNV vec3 rp_color;
layout ( location = 0 ) rayPayloadInNV vec3 rp_color;
```

Qualifiers

`rayPayloadNV`: Used by shader that casts the ray.

`rayPayloadInNV`: Used by shader invoked due to the ray.

Just one payload variable per ray.

Locations

The location is used in the traceNV call.

Typical Ray Generation Shader

Invoked for each pixel.

Casts a ray from eye to its assigned pixel.

Declares a ray payload that can carry a color.

Determines eye location and its assigned pixel.

Casts a ray at that pixel. (Calls traceNV)

Shaders invoked by traceNV should set payload to a lighted color.

After traceNV returns, the ray generation shader writes payload...
... to frame buffer.