Hybrid Ray-Traced Shadows

Jon Story
Senior Developer Technology Engineer
NVIDIA
Problems with shadow mapping?

- Acne
- Peter-panning
- Aliasing

- Endless tuning to alleviate the above... 😞
Peter-panning
Aliasing
Ray tracing to the rescue...
Traditional Bounding Volume Hierarchy

- Can skip many ray-triangle hit tests
- Need to rebuild hierarchy on the GPU
  - For dynamic objects
- Tree traversal is inherently slow
Storing Primitives for Ray-Tracing
Without building a bounding volume hierarchy!

- For shadow maps, store depth from light
- Simple and coherent lookups
- Store primitives similarly
  - A “Deep Primitive Map“
  - Store an array of front facing triangles per texel

Nearest triangle normal
Deep Primitive Map Rendering #1

N在一个维度上的深的原始映射

- 这由3个资源组成：
  - Prim Count Map - 计数每个texel中的三角形数量，使用原子计数相交三角形
  - Prim Indices Map - 三角形在prim缓冲器中的索引
  - Prim Buffer - 三角形的后置变换

Tune N & d per model
Deep Primitive Map Rendering #2

- Is $d$ large enough?
- Visualize occupancy
  - Black: Empty
  - White: Full
  - Red: Limit exceeded
- Easy to get this right for a known model
Deep Primitive Map Rendering #3

- GS outputs 3 vertices & SV_PrimitiveID to PS

```c
[maxvertexcount(3)]
void Primitive_Map_GS( triangle GS_Input IN[3], uint uPrimID : SV_PrimitiveID, inout TriangleStream<PS_Input> Triangles )
{
    PS_Input O;
    [unroll]
    for( int i = 0; i < 3; ++i )
    {
        O.f3PositionWS0 = IN[0].f3PositionWS; // 3 WS Vertices of Primitive
        O.f3PositionWS1 = IN[1].f3PositionWS;
        O.f3PositionWS2 = IN[2].f3PositionWS;
        O.f4PositionCS = IN[i].f4PositionCS;  // SV_Position
        O.uPrimID = uPrimID;                 // SV_PrimitiveID
        Triangles.Append( O );
    }
    Triangles.RestartStrip();
}
```

Tip: Use DX11.1 to output WS vertices directly to UAV
Deep Primitive Map Rendering #4

- PS hashes draw call ID (shader constant) with SV_PrimitiveID to produce prim index/address

```c
float Primitive_Map_PS( PS_Input IN ) : SV_TARGET
{
    // Hash draw call ID with primitive ID
    uint PrimIndex = g_DrawCallOffset + IN.uPrimID;

    // Write out the WS positions to prim buffer
    g_PrimBuffer[PrimIndex].f3PositionWS0 = IN.f3PositionWS0;
    g_PrimBuffer[PrimIndex].f3PositionWS1 = IN.f3PositionWS1;
    g_PrimBuffer[PrimIndex].f3PositionWS2 = IN.f3PositionWS2;

    // Increment current primitive counter
    uint CurrentIndexCounter;
    InterlockedAdd( g_IndexCounterMap[uint2( IN.f4PositionCS.xy )], 1, CurrentIndexCounter );

    // Write out the primitive index
    g_IndexMap[uint3( IN.f4PositionCS.xy, CurrentIndexCounter)] = PrimIndex;

    return 0;
}
```
Deep Primitive Map Rendering #5

- Conservative raster is needed to capture *all* prims touching a texel

- Can be done in SW or HW...
HW Conservative Raster #1

- Rasterize every pixel touched by a triangle
- Enabled in DirectX 12 & 11.3
  - D3D12_RASTERIZER_DESC
  - D3D11_RASTERIZER_DESC2
3 Tiers of functionality
- See DirectX documentation

Be aware that a tier 1 CR can cull degenerate triangles post sub-pixel snapping
- Solution: Ensure you snap triangles in a consistent way for ray tracing
SW Conservative Raster

- Use the GS to dilate a triangle in clip space
- Generate AABB to clip the triangle in the PS
- See GPU Gems 2 - Chapter 42
Ray-Tracing #1
For each screen pixel

- Calc prim map coords (as for shadow mapping)
- Iterate over prim index array
- For each index fetch a triangle for ray testing
```c
float Ray_Test( float2 MapCoord, float3 f3Origin, float3 f3Dir, out float BlockerDistance )
{
    uint uCounter = tIndexCounterMap.Load( int3( MapCoord, 0 ), int2( 0, 0 ) ).x;

    [branch]
    if( uCounter > 0 )
    {
        for( uint i = 0; i < uCounter; i++ )
        {
            uint uPrimIndex = tIndexMap.Load( int4( MapCoord, i, 0 ), int2( 0, 0 ) ).x;

            float3 v0, v1, v2;
            Load_Prim( uPrimIndex, v0, v1, v2 );

            // See "Fast, Minimum Storage Ray / Triangle Intersection"
            // by Tomas Müller & Ben Trumbore
            [branch]
            if( Ray_Hit_Triangle( f3Origin, f3Dir, v0, v1, v2, BlockerDistance ) != 0.0f )
            {
                return 1.0f;
            }
        }
    }

    return 0.0f;
}
```
Shadow Map

SM = 3K x 3K (36 MB)
Shadow Map

SM = 8K x 8K (256 MB)
Ray Traced

SM = 3K x 3K (36 MB)
PM = 1K x 1K x 64 (256 MB)
Shadow Map

SM = 3K x 3K (36 MB)
Ray Traced

SM = 3K x 3K (36 MB)
PM = 1K x 1K x 64 (256 MB)
Shadow Map

SM = 3K x 3K (36 MB)
Ray Traced

SM = 3K x 3K (36 MB)
PM = 1K x 1K x 32 (128 MB)
Anti-Aliasing

- Shoot additional rays to achieve this?
  - This is very expensive!
- Simple trick – apply a screen space AA technique (FXAA, MLAA, etc.)

If you’re not cheating, you’re just not trying
No AA
FXAA
Are hard shadows that useful in games?
Hybrid Approach

- Combine ray-traced shadow with conventional soft shadows
- Use an advanced filtering technique such as CHS or PCSS
- Use blocker distance to compute a lerp factor
- As blocker distance -> 0 ray-traced result is prevalent
L = saturate( BD / WSS * PHS )

L: Lerp factor
BD: Blocker distance (from ray origin)
WSS: World space scale - chosen based upon model
PHS: Desired percentage of hard shadow

FS = lerp( RTS, PCSS, L )

FS: Final shadow result
RTS: Ray traced shadow result (0 or 1)
PCSS: PCSS+ shadow result (0 to 1)
Use a Shrinking Penumbra Filter

- Otherwise the soft shadow result will not be fully contained by the ray traced result
- This would cause problems when performing a lerp between the two
Hybrid Ray Traced + PCSS
Standard Filter
Hybrid Ray Traced + PCSS
Shrinking Penumbra Filter
PCSS
SM = 3K x 3K (36 MB)
Hybrid Ray Traced + PCSS

SM = 3K x 3K (36 MB)
PM = 1K x 1K x 64 (256 MB)
PCSS
SM = 3K x 3K (36 MB)
Hybrid Ray Traced + PCSS
SM = 3K x 3K (36 MB)
PM = 1K x 1K x 64 (256 MB)
PCSS
SM = 3K x 3K (36 MB)
Hybrid Ray Traced + PCSS

SM = 3K x 3K (36 MB)
PM = 1K x 1K x 32 (128 MB)
Prims: ~10K
Shadow Map: 3K x 3K (36 MB)
Primitive Map: 1K x 1K x 32 (128 MB)
Primitive Buffer: ~360K
Shadow Buffer: 1920 x 1080

<table>
<thead>
<tr>
<th></th>
<th>AMD R9 290X</th>
<th>NV GTX 980</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primitive Map + HW CR</td>
<td>----</td>
<td>0.4</td>
</tr>
<tr>
<td>Primitive Map + SW CR</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>Ray Trace</td>
<td>0.5</td>
<td>0.4</td>
</tr>
<tr>
<td>PCSS</td>
<td>1.4</td>
<td>1.3</td>
</tr>
<tr>
<td>PCSS + Ray Trace</td>
<td>1.9</td>
<td>1.8</td>
</tr>
<tr>
<td>FXAA</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Quoted times in milliseconds
Prims: ~65K
Shadow Map: 3K x 3K (36 MB)
Primitive Map: 1K x 1K x 64 (256 MB)
Primitive Buffer: ~2.2 MB
Shadow Buffer: 1920 x 1080

<table>
<thead>
<tr>
<th></th>
<th>AMD R9 290X</th>
<th>NV GTX 980</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primitive Map + HW CR</td>
<td>----</td>
<td>0.5</td>
</tr>
<tr>
<td>Primitive Map + SW CR</td>
<td>1.4</td>
<td>0.7</td>
</tr>
<tr>
<td>Ray Trace</td>
<td>1.0</td>
<td>0.7</td>
</tr>
<tr>
<td>PCSS</td>
<td>1.4</td>
<td>1.3</td>
</tr>
<tr>
<td>PCSS + Ray Trace</td>
<td>3.0</td>
<td>2.8</td>
</tr>
<tr>
<td>FXAA</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Quoted times in milliseconds
Prims: ~240K
Shadow Map: 3K x 3K (36 MB)
Primitive Map: 1K x 1K x 64 (256 MB)
Primitive Buffer: ~8.2 MB
Shadow Buffer: 1920 x 1080

<table>
<thead>
<tr>
<th></th>
<th>AMD R9 290X</th>
<th>NV GTX 980</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primitive Map +</td>
<td>----</td>
<td>3.4</td>
</tr>
<tr>
<td>HW CR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Primitive Map +</td>
<td>----</td>
<td>4.1</td>
</tr>
<tr>
<td>SW CR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ray Trace</td>
<td>1.2</td>
<td>1.0</td>
</tr>
<tr>
<td>PCSS</td>
<td>1.4</td>
<td>1.3</td>
</tr>
<tr>
<td>PCSS + Ray Trace</td>
<td>3.7</td>
<td>3.4</td>
</tr>
<tr>
<td>FXAA</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Quoted times in milliseconds
Limitations

- Currently limited to a single light source
- This would not scale up to work for a whole scene
  - Storage would become the limiter
- But is ideal for closest models:
  - Current model of focus
  - Contents of nearest cascade
Summary

- Addresses conventional shadow map problems
- AA ray-traced hard shadows are highly performant
- Hybrid shadows combine best of both worlds
- No need to re-write your engine
- Fast enough for games today!
ShaderX 5: Alias-free Hard Shadows with Geometry Maps
Laszlo Szecsi

Sub-Pixel Shadow Mapping - ACM i3D 2014
Pascal Lecocq, Jean-Eudes Marvie, Gael Sourimant, Pascal Gautron

Fast, Minimum Storage Ray / Triangle Intersection
Tomas Möller, Ben Trumbore

GPU Gems 2: Conservative Rasterization
Jon Hasselgren, Tomas Akenine-Möller, Lennart Ohlsson
Questions?

jons@nvidia.com