Volumetric Global Illumination At Treyarch

JT Hooker
Treyarch Senior Graphics Engineer
Volumetric Global Illumination

- GI in volume texture
- Lean texture data
- IBL baked from probes
- Convex blend shapes
Presentation Order

Where we started

Evolution along the way

Where we ended up

Advances in Real-Time Rendering course, SIGGRAPH 2016
Traditional Approach: Lightmaps

Could be ok, but…
Lightmap Downsides

- Works poorly on detailed or intersecting geometry
Lightmap Downsides

- Doesn’t work at all on dynamic geometry
Lightmap Downsides

- Software ray-tracing and shading takes forever
Lightmap Downsides

- Results not visible in world editor
Process of Invention

- Deferred Renderer
- Reflections already present
- So how do we apply deferred GI?
Reflection Probes as Diffuse Data

• Higher Mips: convolved specular

[DROBOT13]

• Lowest Mip: diffuse irradiance

• Real time IBL

Advances in Real-Time Rendering course, SIGGRAPH 2016
Occlusion Is A Problem
Visibility Is A Problem

• Where the probe doesn’t see
• Looks like shadows
Irradiance Volume [TATARCHUK05]
Render a Reflection Probe Per Voxel?

$$138 \text{ Volumes} \times 40^3 \text{ Voxels} \times 6 \text{ Faces}$$

$$\div 60 \text{ FPS} \div 60 \text{ Seconds}$$

$$= 14,720 \text{ Minutes} (\approx 10 \text{ Days})$$
Collect Colors From Reflection Probes

- Re-project cube maps
- Combine to fill holes

[BUEHLER01]
In Practice

• 4096 rays per voxel
• 15 neighbors considered
• Missed rays are in-painted
Re-Project From Existing Probes
Reprojection

- Neighbor candidates sorted based on distance
- What about spec?
Reprojection

- Angle and distance to surface defines a solid angle in the cube map
Reprojection

- Sample area validated against depth pyramid
- If visible appropriate mip sampled
Reprojection Calculation

Advances in Real-Time Rendering course, SIGGRAPH 2016
Reprojection Calculation

distFromUnitCube = \sqrt(1 + u^2 + v^2); // Compensation for cube-map shape.
angleOfVoxel = 4 * PI / numSamples; // Solid angle from voxel.
inSqrt = 1 + distFromVoxel^2 * angleOfVoxel * (angleOfVoxel - 4PI) / (4 * PI^2 * distFromProbe^2);
angleOfProbe = 2PI * (1 - \sqrt{inSqrt}); // Solid angle from reflection probe.
cubeRes = 1.0f / \sqrt(\text{angleOfProbe} * \text{distFromUnitCube}^3); // Resolution needed for sample.
mipLevel = \text{clamp}(\text{mipCount} - \log2(\text{cubeRes}), 0, \text{mipCount}); // Mip level to use.

return mipLevel;
Biggest Benefit

- Hardware rendering
- Re-render to get bounces
- Only have to ray-trace and re-project once
Texture Encoding

- Flat Color?
Texture Encoding

- Ambient / Highlight / Direction?
Texture Encoding

- Second Order Spherical Harmonic?
Texture Encoding

• Ambient Cube!
  [MCTAGGART04]
  • BC6H Compressed
Volume Texture Layout
Performance Benefits

- Only 3 samples
- Hardware trilinear filtering

Evaluation:
\[
\text{color}[n] = \text{normal}^2 \cdot \text{float3}(X_{\text{sample}[n]}, Y_{\text{sample}[n]}, Z_{\text{sample}[n]})
\]

\[
\text{color} = \text{xVolume}.\text{SampleLevel}(\text{coord}) \cdot \text{normal.x} \cdot \text{normal.x} + \\
\text{yVolume}.\text{SampleLevel}(\text{coord}) \cdot \text{normal.y} \cdot \text{normal.y} + \\
\text{zVolume}.\text{SampleLevel}(\text{coord}) \cdot \text{normal.z} \cdot \text{normal.z};
\]
Light Leaking Is A Problem
Common Approach

- Adjust trilinear based on normal
  [SILVENNOINEN15]

- Our approach needs to be more reliable
More Voxel Data

- Planes
- Signed distance field
- Bad artifacts
Solve With Shaping

Advances in Real-Time Rendering course, SIGGRAPH 2016
Click To Size Boxes
Click To Add Boxes
Auto-parent on creation
Voxels Near Walls
Consider Backfaces
Complex Room Shapes

Advances in Real-Time Rendering course, SIGGRAPH 2016
Solution: Convex Shapes
Multiface Volumes

Click to add and remove faces.
Multiface Volume Editing

Drag / Cut / Slice / Rotate
Subtract Shapes

CSG add
Then subtract
Override Volumes
Like priority
Only two levels.

Advances in Real-Time Rendering course, SIGGRAPH 2016
Runtime Implementation

1. Cull against volume AABB’s to build a list of volumes

2. Per pixel calculate attenuation on visible volumes
   - Convex hull CSG
   - Groups of six planes either extended, combined or subtracted
Example GI Volume

```c
struct PlaneGroup {
    float4 planes[6]; // Groups of six planes.
    bool subtractive; // Per group, specifies whether it adds or subtracts.
    bool finished; // Per group, whether it should be combined with the previous.
}

struct GIvolume {
    PlaneGroup *groups;
}

// Blends, or “feather”, are pre-multiplied into the plane definition.
planes[i].xyz = planeNormal;
planes[i].w = planeOffset;
planes[i] /= blendWidth; // Blend width is a scalar for how wide the blend is.
```
Group Size?

[6]+[6+6+…?]  
[6]+[4+4+…?]  
[4]+[4+4+…?]  
[8]+[2+2+…?]
Shader Example

```
attenuation = 0;
groupAtten = 1;
for ( int group = 0; group < numGroups; group++)
{
    groupAtten *= saturate( dot( planes[group][0].xyz, pos ) + planes[group][0].w );
groupAtten *= saturate( dot( planes[group][1].xyz, pos ) + planes[group][1].w );
groupAtten *= saturate( dot( planes[group][2].xyz, pos ) + planes[group][2].w );
groupAtten *= saturate( dot( planes[group][3].xyz, pos ) + planes[group][3].w );
groupAtten *= saturate( dot( planes[group][4].xyz, pos ) + planes[group][4].w );
groupAtten *= saturate( dot( planes[group][5].xyz, pos ) + planes[group][5].w );
    if( finished[group] )
    {
        if( subtractive[group] )
            attenuation = max( attenuation, groupAtten );
        else
            attenuation *= 1.0f - groupAtten;
    }
    groupAtten = 1;
}
return saturate( attenuation );
```
Why Not K-DOPs?

KDOP – $k$-sided Discrete Oriented Polytope

Pairs of planes or slabs
Instead of individual planes
Runtime Implementation

3. Sample three ambient cube values depending on normal

4. Blend results between all volumes
Challenges
Problem: Geo Within Voxels

Advances in Real-Time Rendering course, SIGGRAPH 2016
Solution: Smart Centers

Advances in Real-Time Rendering course, SIGGRAPH 2016
Empty Space Skip

Advances in Real-Time Rendering course, SIGGRAPH 2016
Problem: Seams
Solution: Volume Smoothing
Careful Lighting Artistry

Advances in Real-Time Rendering course, SIGGRAPH 2016
Auto Volumes?

“Do-Everything Button”
Debug Tools

Volume Blending And Density

Volume Overdraw Per Tile
Reflections

Advances in Real-Time Rendering course, SIGGRAPH 2016
Reflection Planes

Advances in Real-Time Rendering course, SIGGRAPH 2016

[LAGARDE12]
Clever Artistry

Advances in Real-Time Rendering course, SIGGRAPH 2016
Reflection Plane Parallax

float reflectionMip = (1 - gloss) * numMips;

// as things get rougher “fade off” parallax correction
// by pushing out intersection planes
float minDist = saturate((reflectionMip - 2.5) / (numMips - 2.5)) * 100;
distanceToPlane = max(abs(distanceToPlane), minDist);

float intersectionDist = abs(distanceToPlane / -dot(direction, plane.xyz));
Parallax Fade Out
Reflection Brightness Correction

[LAZAROV13]
Reflection Brightness Correction
Brightness Correction

```c
float maximumSpecValue = max3( 1.26816,
    9.13681 * exp2( 6.85741 - 2 * mip ) * nDotV,
    9.70809 * exp2( 7.085 - mip - 0.403181 * mip^2) * nDotV );

float adjustedMaxSpec = diffuseGILum * maximumSpecValue;
float3 specLum = luminance( cubeMapSample );
float3 reflection = cubeMapSample *
    adjustedMaxSpec / ( adjustedMaxSpec + speculum );
```
Pros:

1. As good or better quality than light maps
Pros:

2. Less than 2ms for reflections and GI
Pros:

3. Works on all geometry
Pros:

4. Less baking time with incremental baking
Pros:

5. Baking is done in editor
Pros:

6. Moving and changing GI
Pros:

7. Loose connection between light and geo
Cons:

1. Takes set up time
Cons:

2. Training is hard
Cons:

3. Either lower resolution or more memory use in game
Cons:

4. Need beefy dev machines

   (48Gb RAM and 12Gb VRAM)
Cons:

5. Development challenges
Special Thanks

*Treyarch:*
Dimitar Lazarov – Original Idea
Kevin Myers – Baking Code
Everyone Else at Treyarch

*Activision Central Tech:*
Peter-Pike Sloan – Lots of Math
Josiah Manson – Light Bake Features
Angelo Pesce – Reflection Solutions

Advances in Real-Time Rendering course, SIGGRAPH 2016
References

- [TATARCHUCK05] TATARCHUK, N., 2005. *Irradiance Volumes for Games*, GDC Europe

Advances in Real-Time Rendering course, SIGGRAPH 2016