ACCELERATING YOUR VR APPLICATIONS WITH VRWORKS

Cem Cebenoyan
Edward Liu

ACCELERATING YOUR VR APPLICATIONS WITH VRWORKS

Talk Overview

- VRWorks Overview

  Graphics, audio, video stitching, physics

- Lens Matched Shading Deep Dive

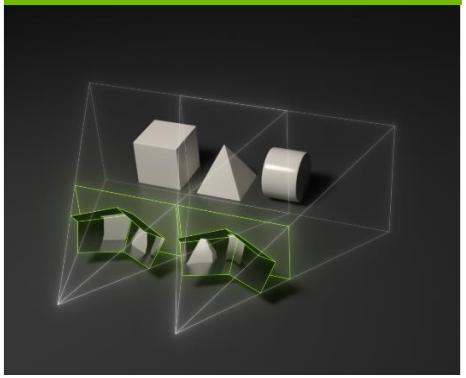
  Fundamental technology

  Engine integration details

NVIDIA VRWORKS

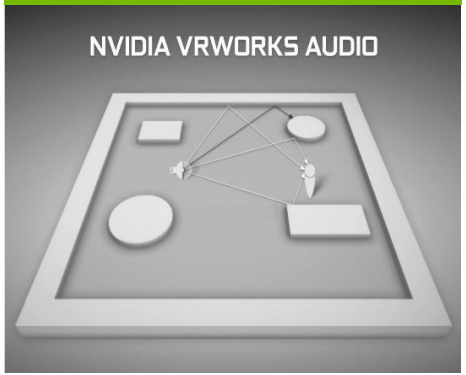
Bringing Reality to VR

SIGHT



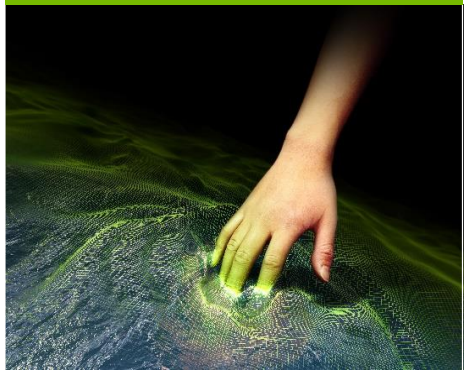
Simultaneous
Multi-projection

SOUND



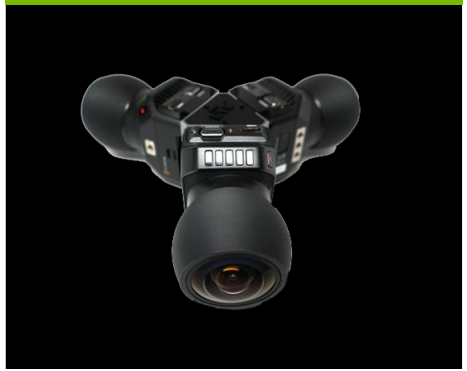
Ray Traced
Audio

PHYSICS & TOUCH



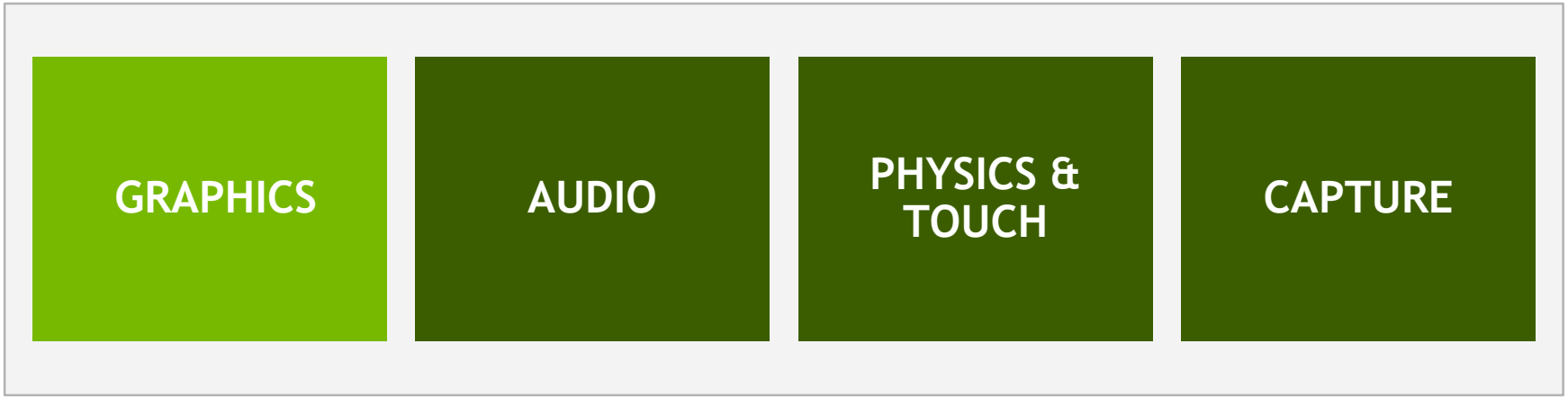
PhysX

CAPTURE



VRWorks 360 Video
& CUDA

COMPUTING CHALLENGES IN REPRODUCING REALITY

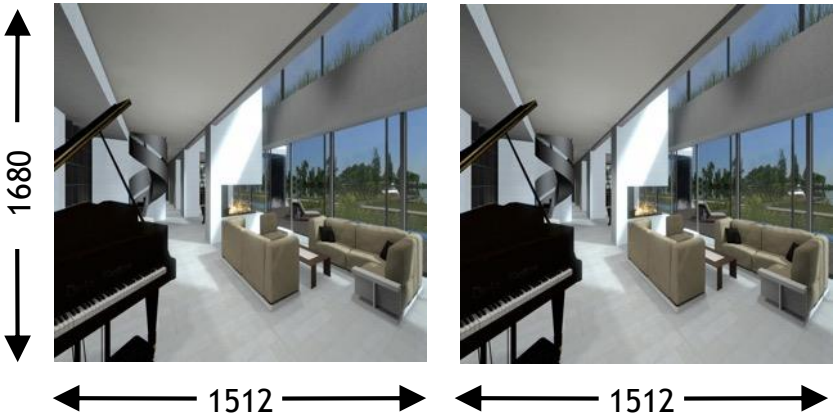


VR RENDERING

Ultra-High Resolution and Frame Rate

TRADITIONAL = 60 MP/S
(1920 X 1080 @ 30 FPS)

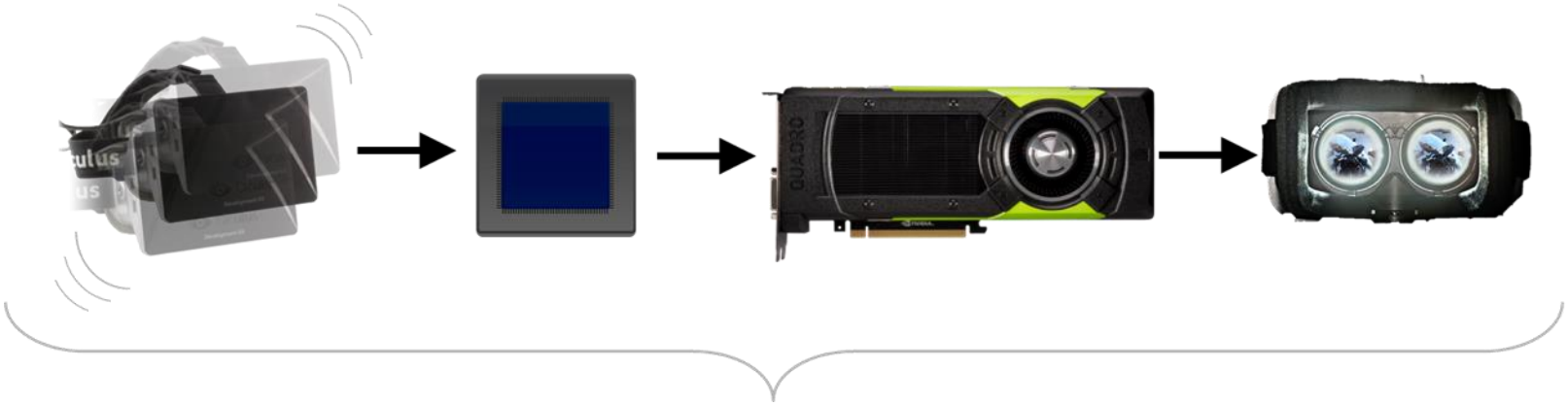
VIRTUAL REALITY = 450 MP/S
(3024 X 1680 @ 90 FPS)*



**VR render resolution*

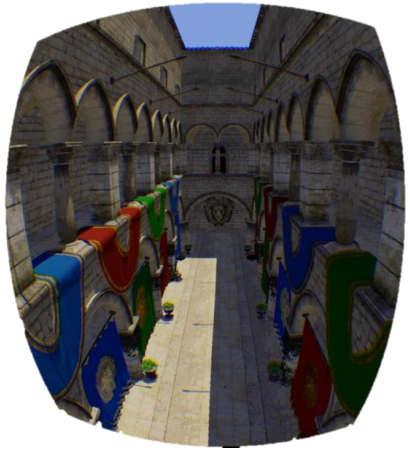
VR PERFORMANCE DEMANDS

Ultra-Low Latency

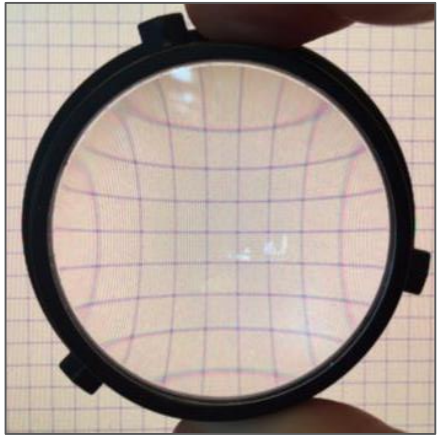


Motion to Photon: ≤ 20 ms

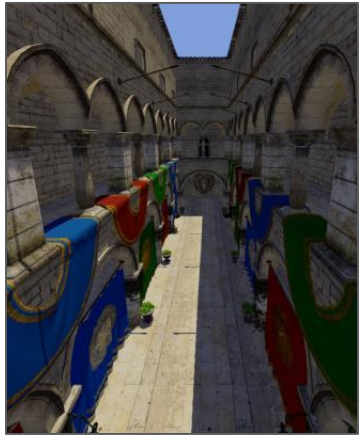
VR OPTICS



LCD display

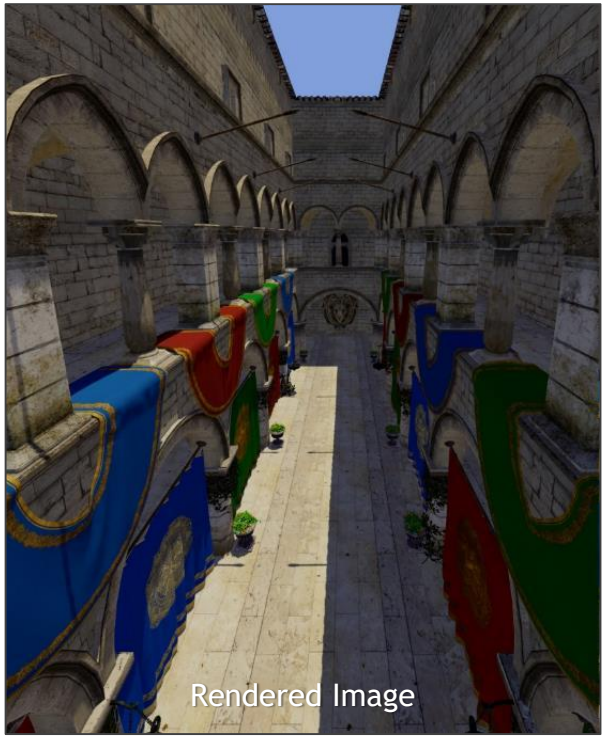


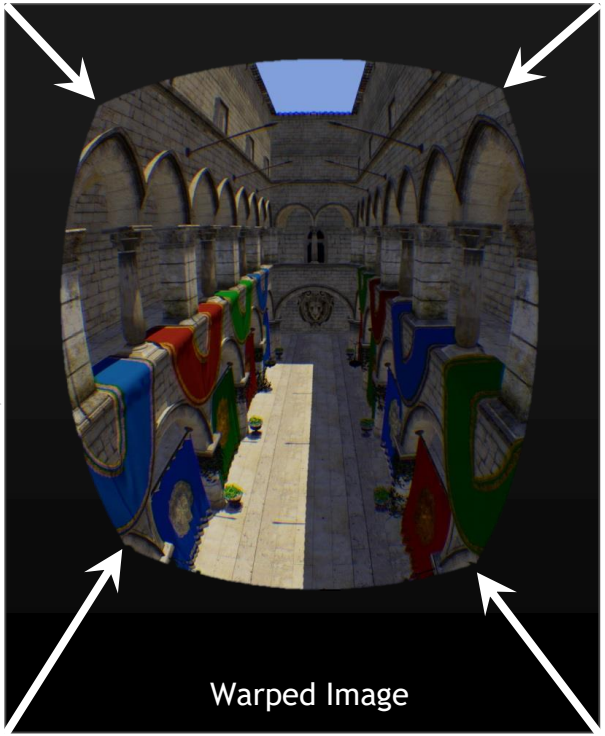
Optics



User’s view

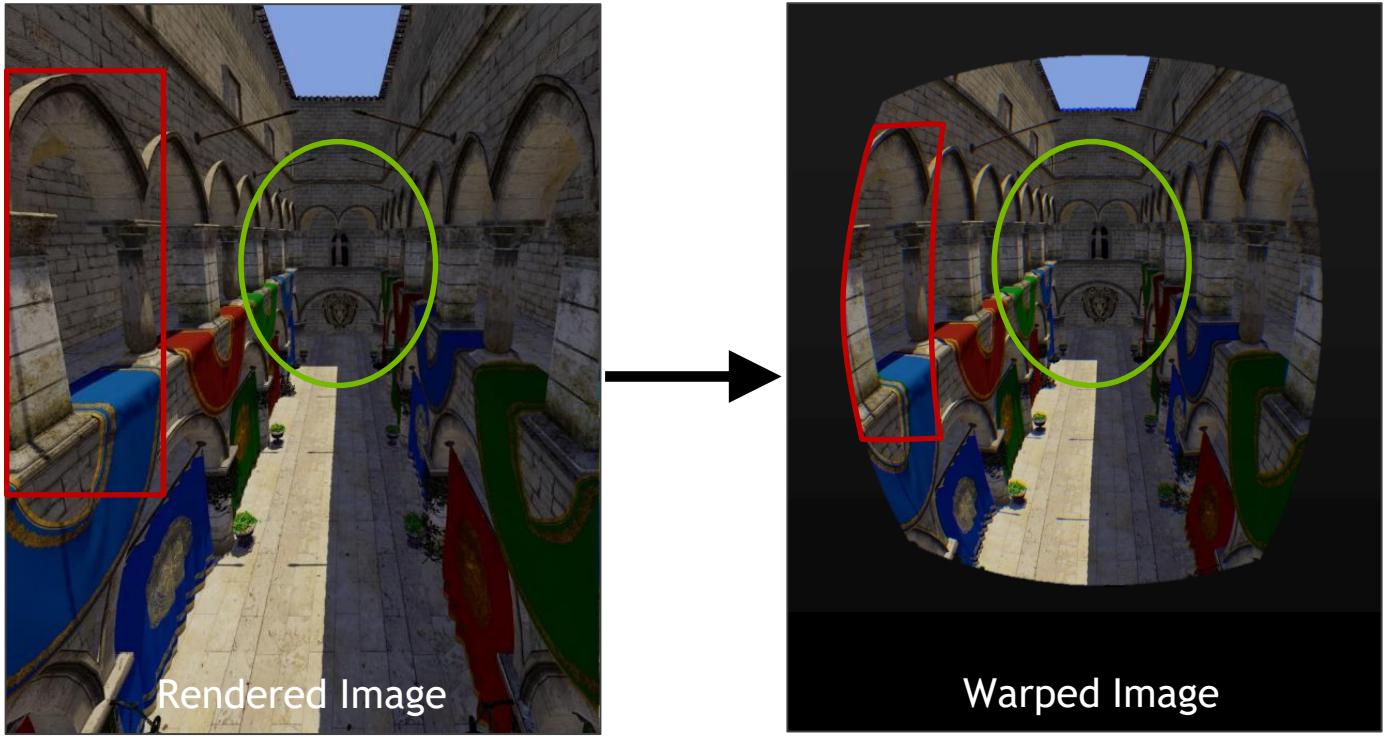
VR RENDERING



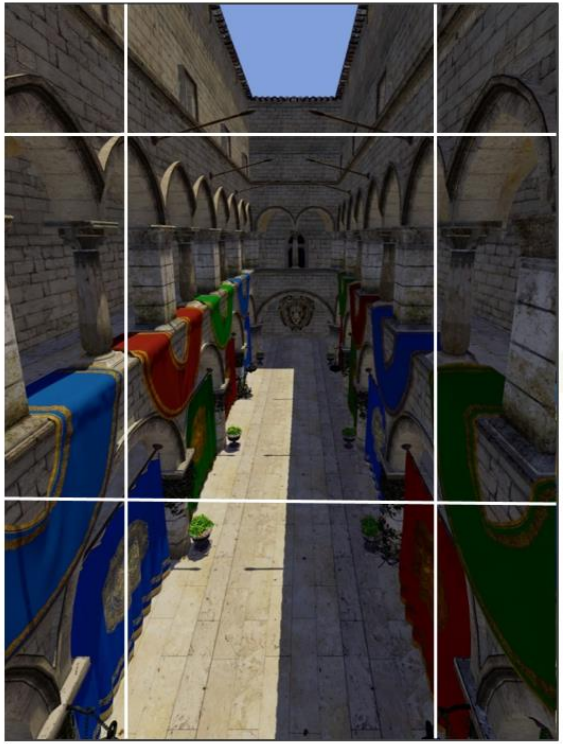


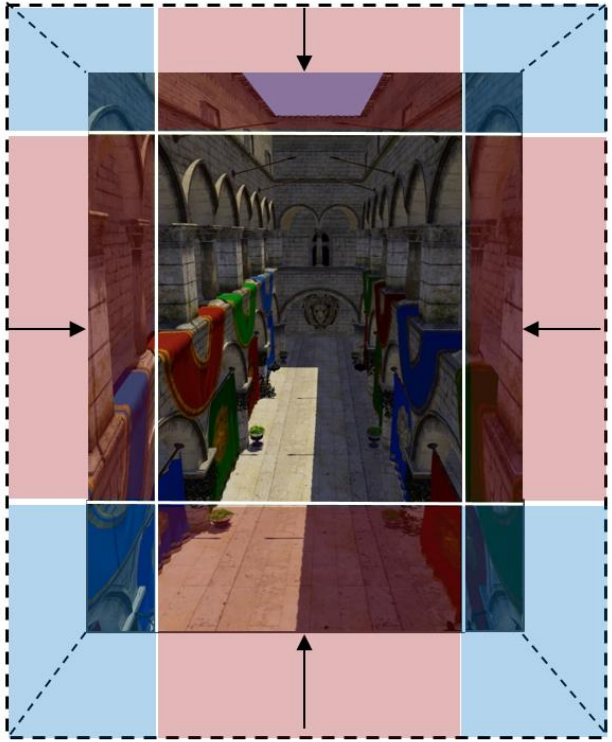
VR RENDERING

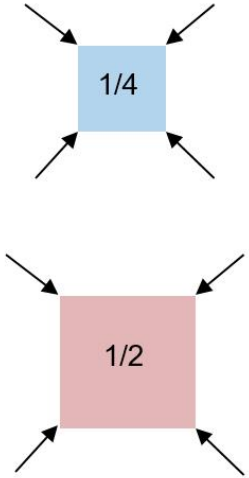
GPU renders many pixels that never make it to the screen



VRWORKS MULTI-RES SHADING

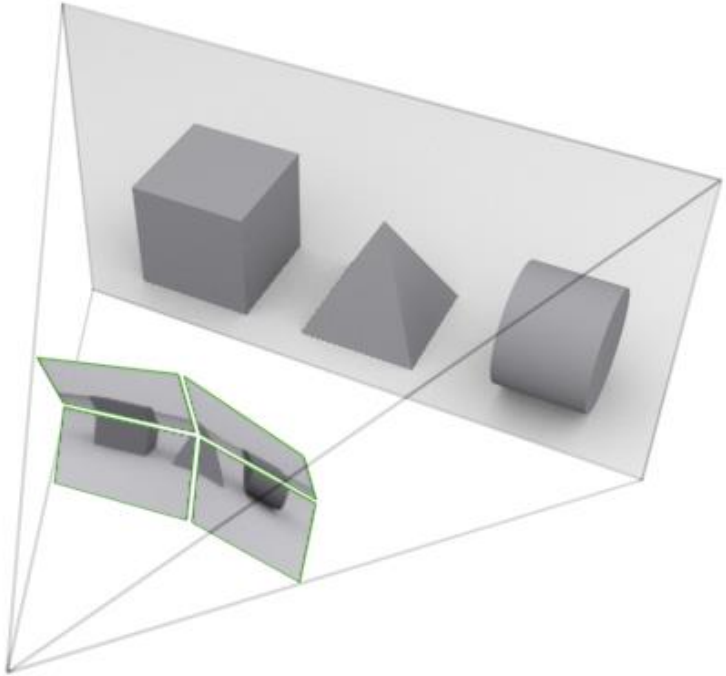






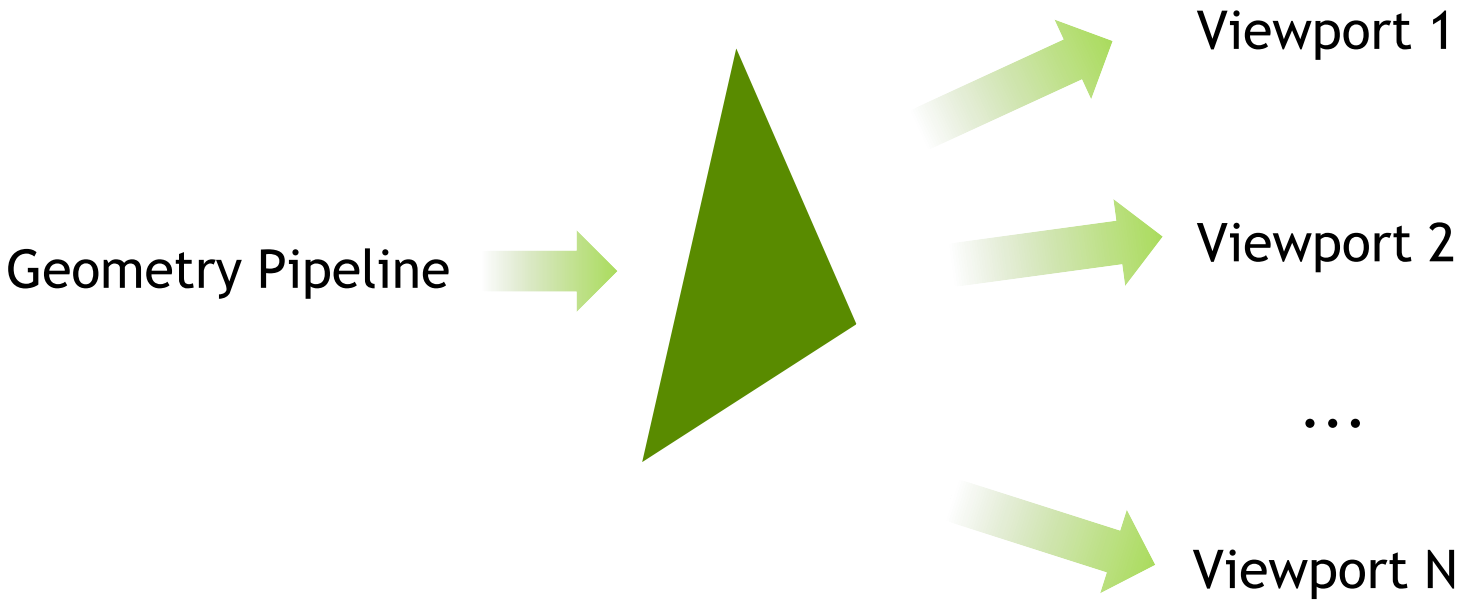
VRWORKS LENS MATCHED SHADING

Renders to a lens corrected surface



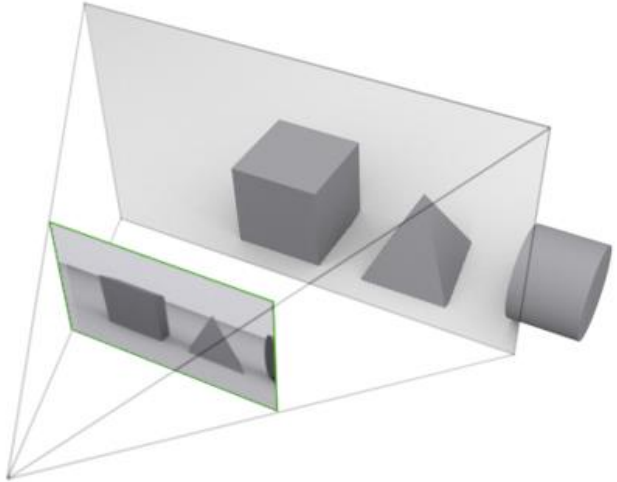
VRWORKS MRS AND LMS

Fast viewport broadcast hardware on NVIDIA Maxwell and beyond GPUs

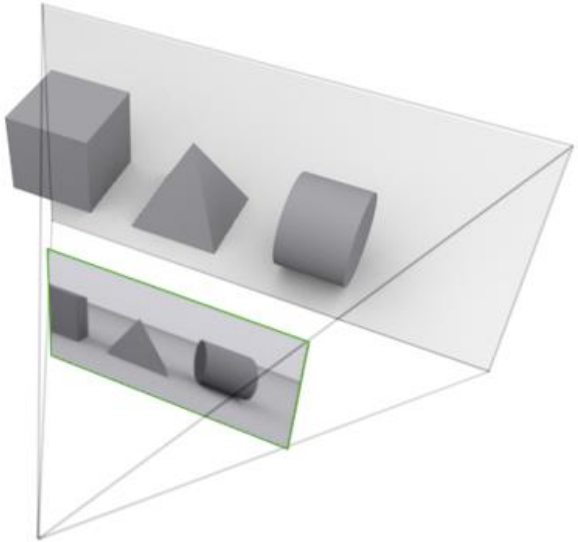


TRADITIONAL STEREO RENDERING

Requires 2 geometry passes



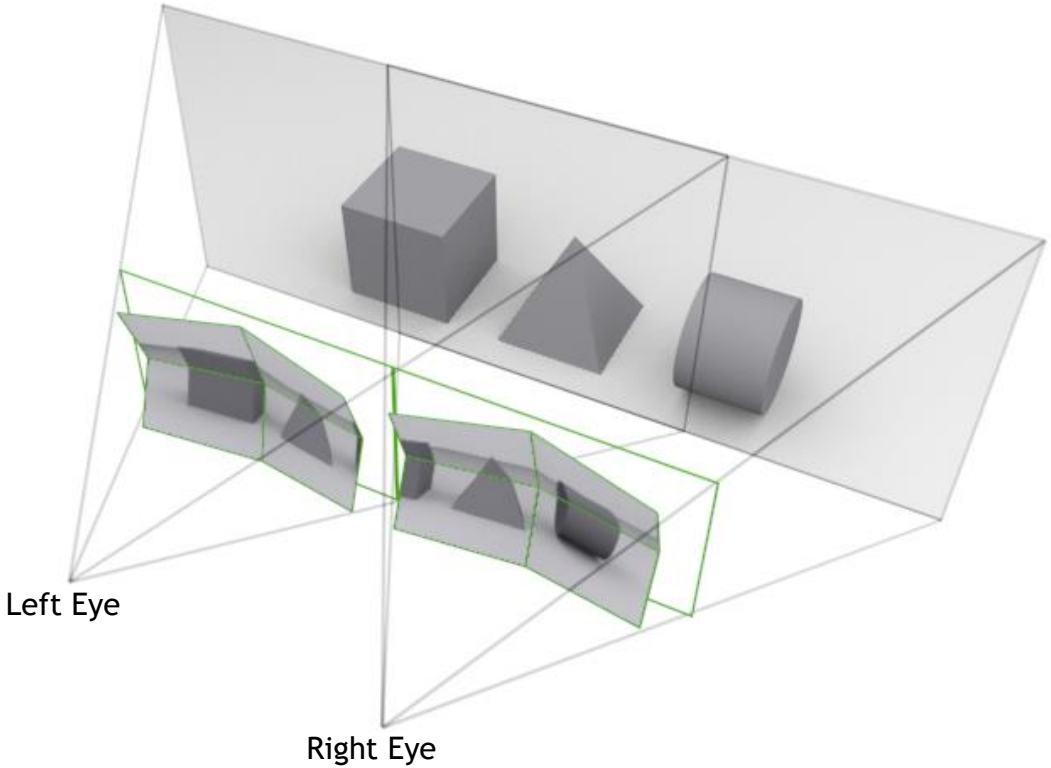
Left Eye (Pass 1)



Right Eye (Pass 2)

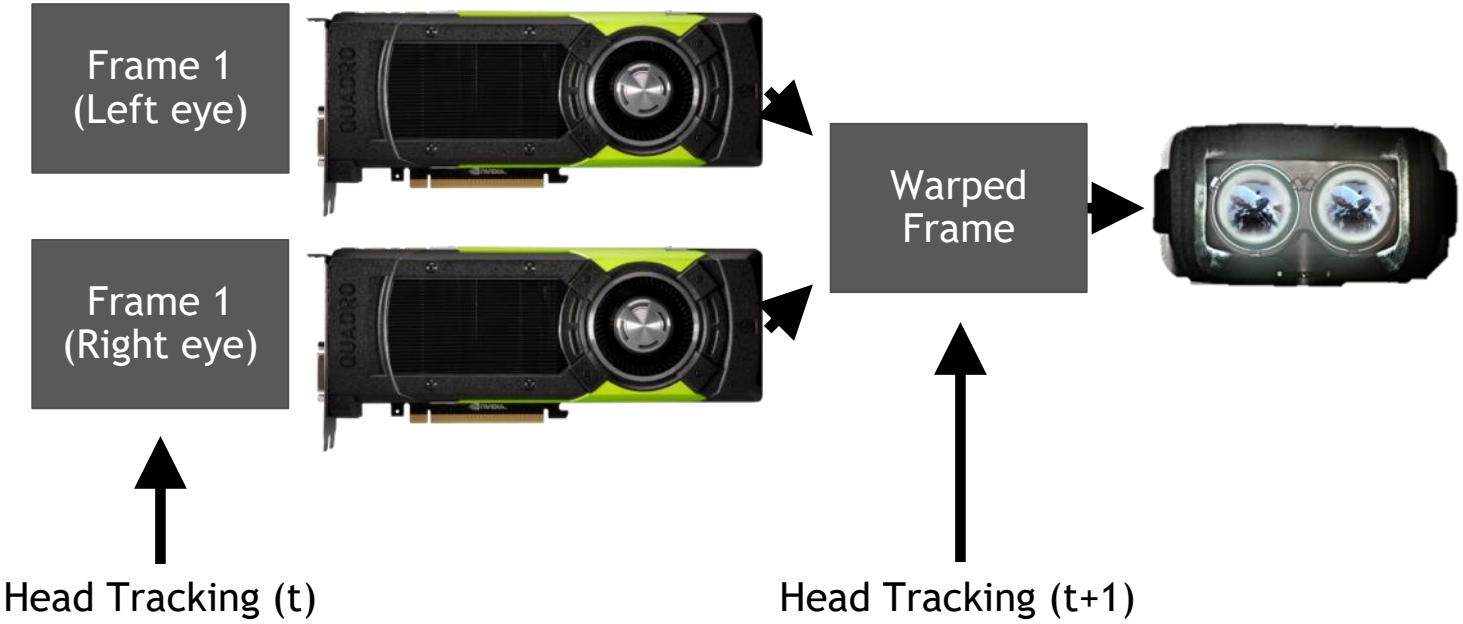
VRWORKS SINGLE PASS STEREO

Renders left & right eye in one geometry pass



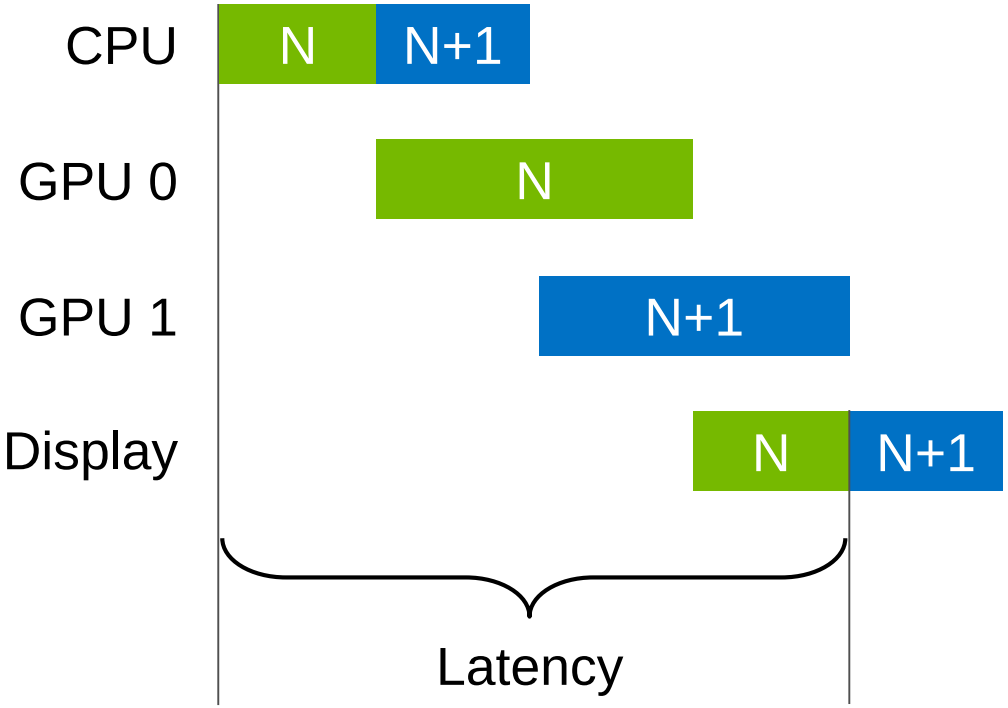
VRWORKS VR SLI

Scales performance across multiple GPUs



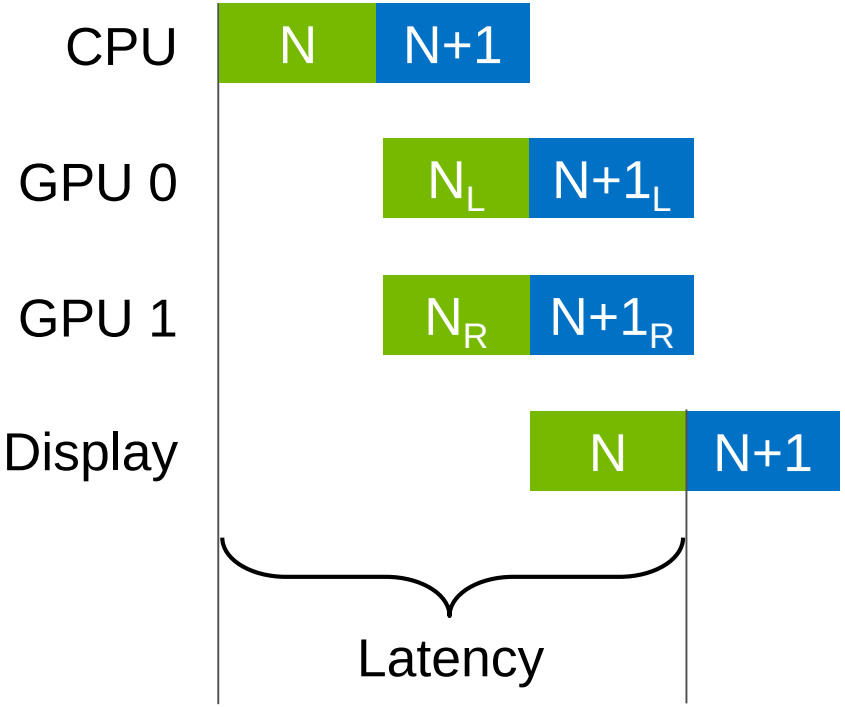
“Normal” SLI

GPUs render alternate frames



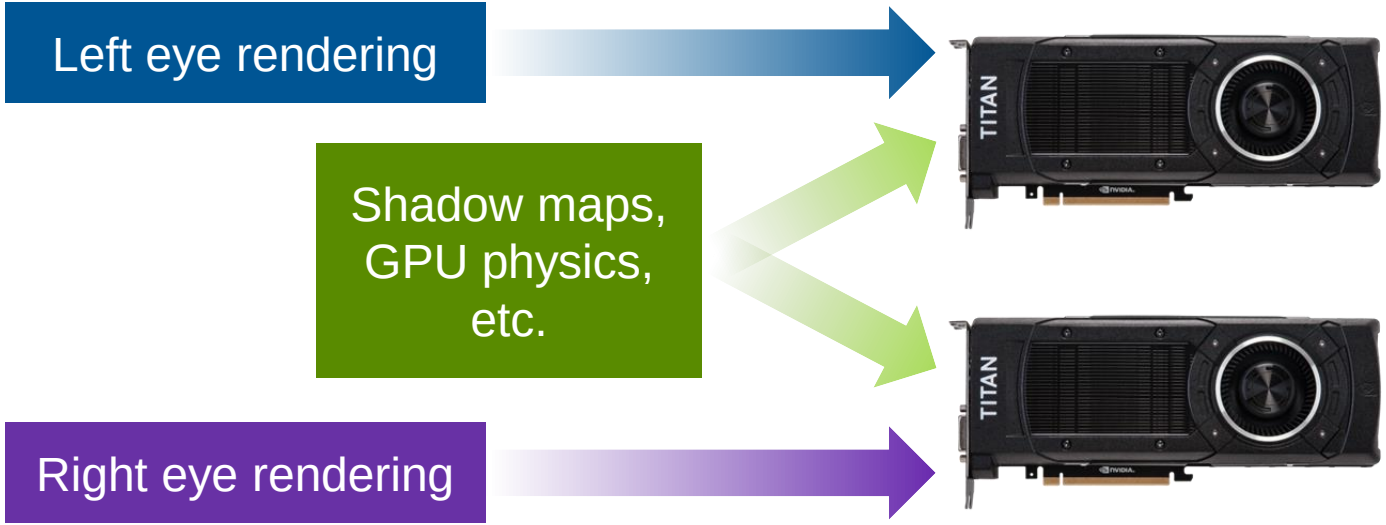
VR SLI

Each GPU renders one eye—lower latency



VR SLI

GPU affinity masking: full control

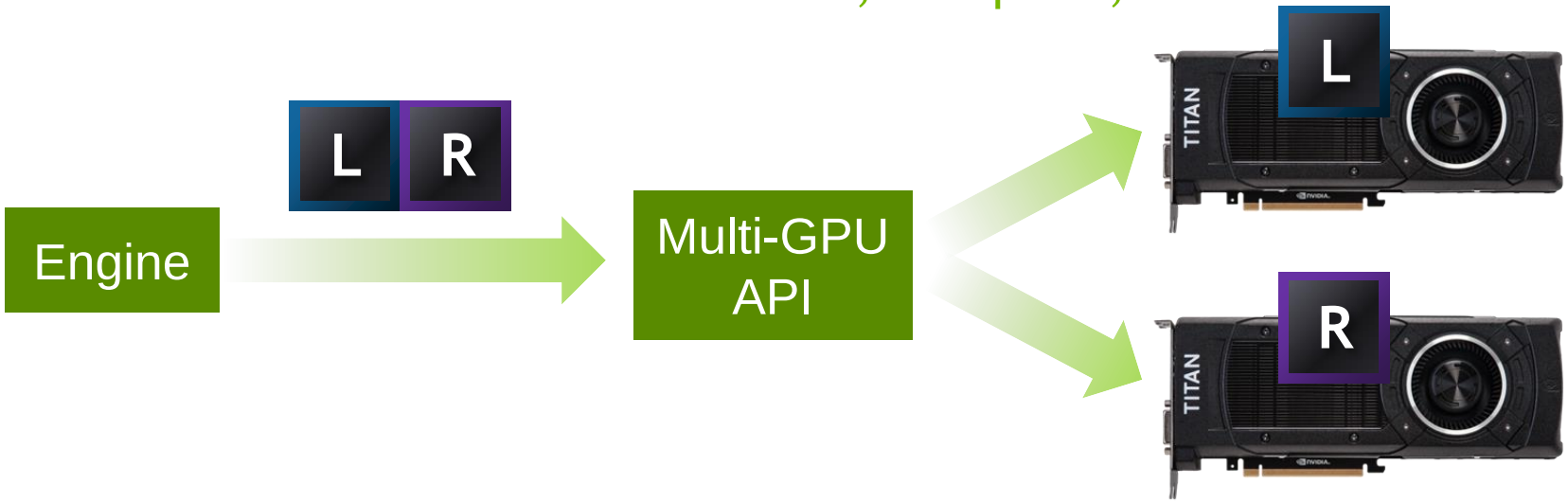



```
UINT SetGPUMask(
     [in]  UINT GPUMask
);
```

```
void RenderGPUMaskNV(
     [in]  bitfield mask
);
```

VR SLI

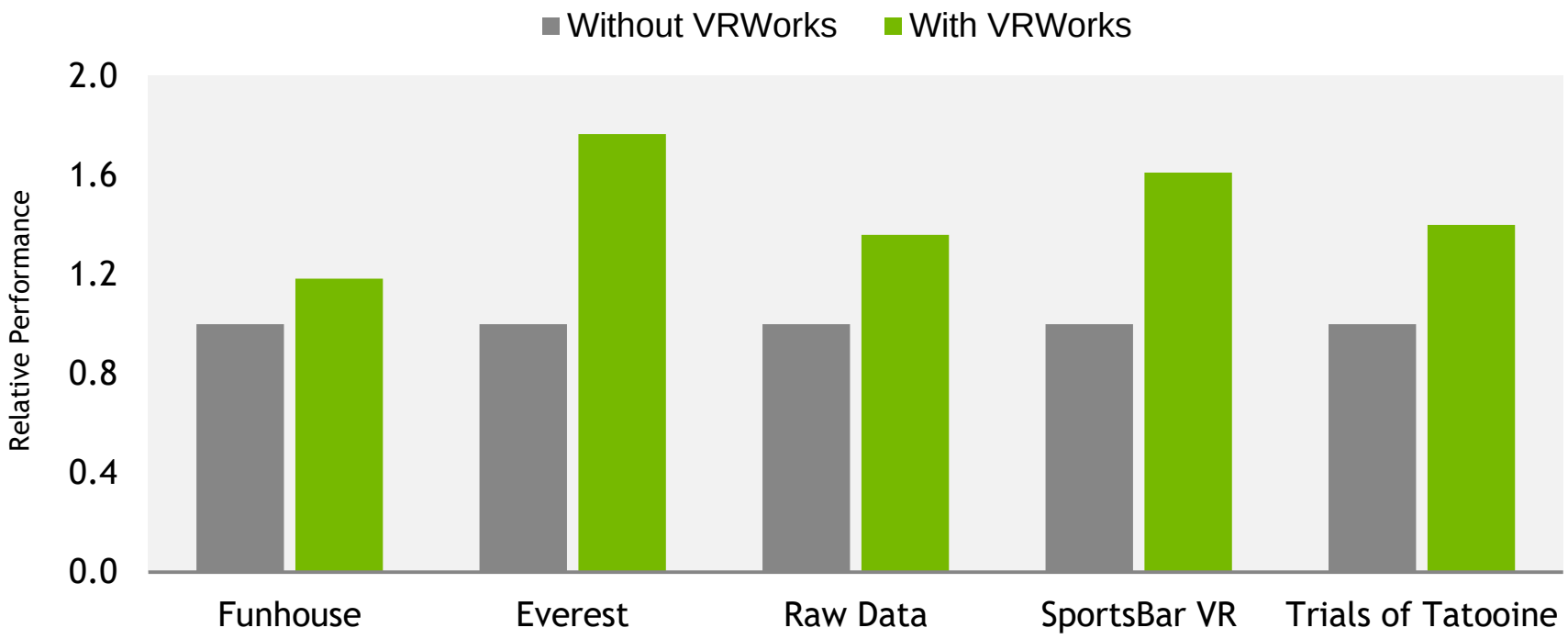
Per-GPU constant buffers, viewports, scissors




```
NvAPI_Status VSSetConstantBuffers(
    [in]   ID3D11DeviceContext *pContext,
    [in]   UINT GPUMask,
    [in]   UINT StartSlot,
    [in]   UINT NumBuffers,
);
```

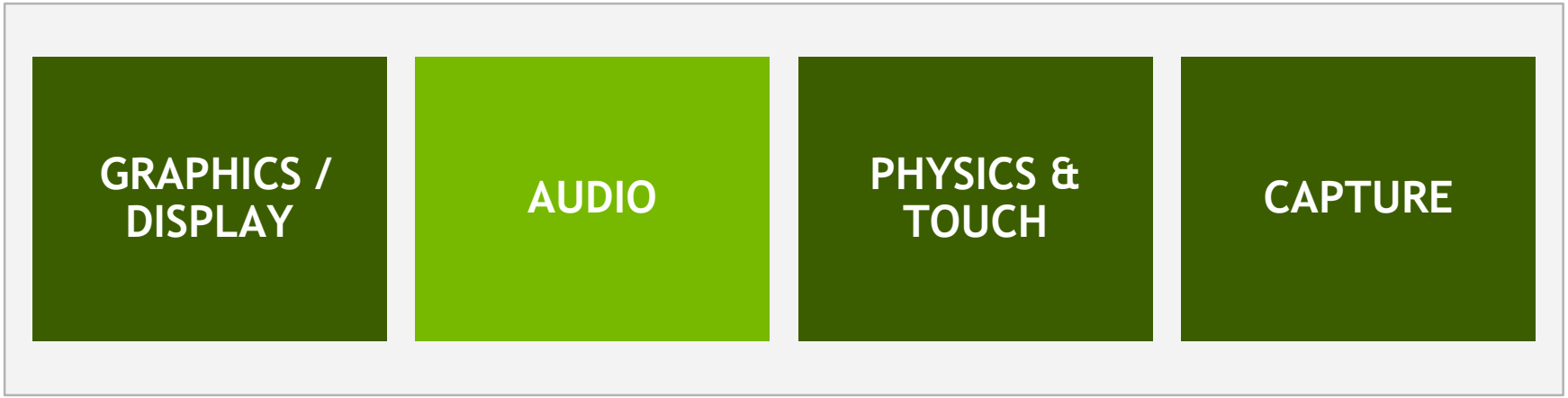
```
void MulticastBufferSubDataNV(
    bitfield gpuMask,
    uint buffer,
    intptr offset,
    sizeiptr size,
    const void *data );
```

VRWORKS BRINGS MAJOR SPEEDUPS TO TOP VR APPS



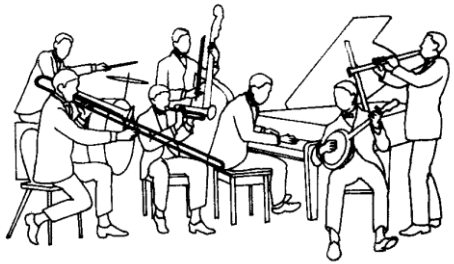
**Performance measured on GeForce GTX 1080 using VRWorks MRS, LMS, or VR SLI*

COMPUTING CHALLENGES IN REPRODUCING REALITY



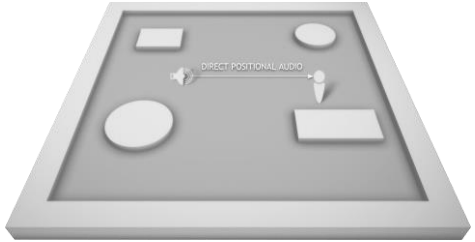
SIMULATING AUDIO IN VR

SYNTHESIS



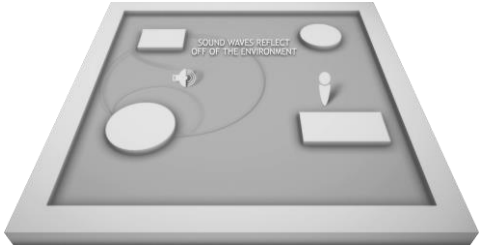
Creation of Source Sounds

DIRECTION



Location of Incoming Sound

PROPAGATION



How Sound Moves in Space

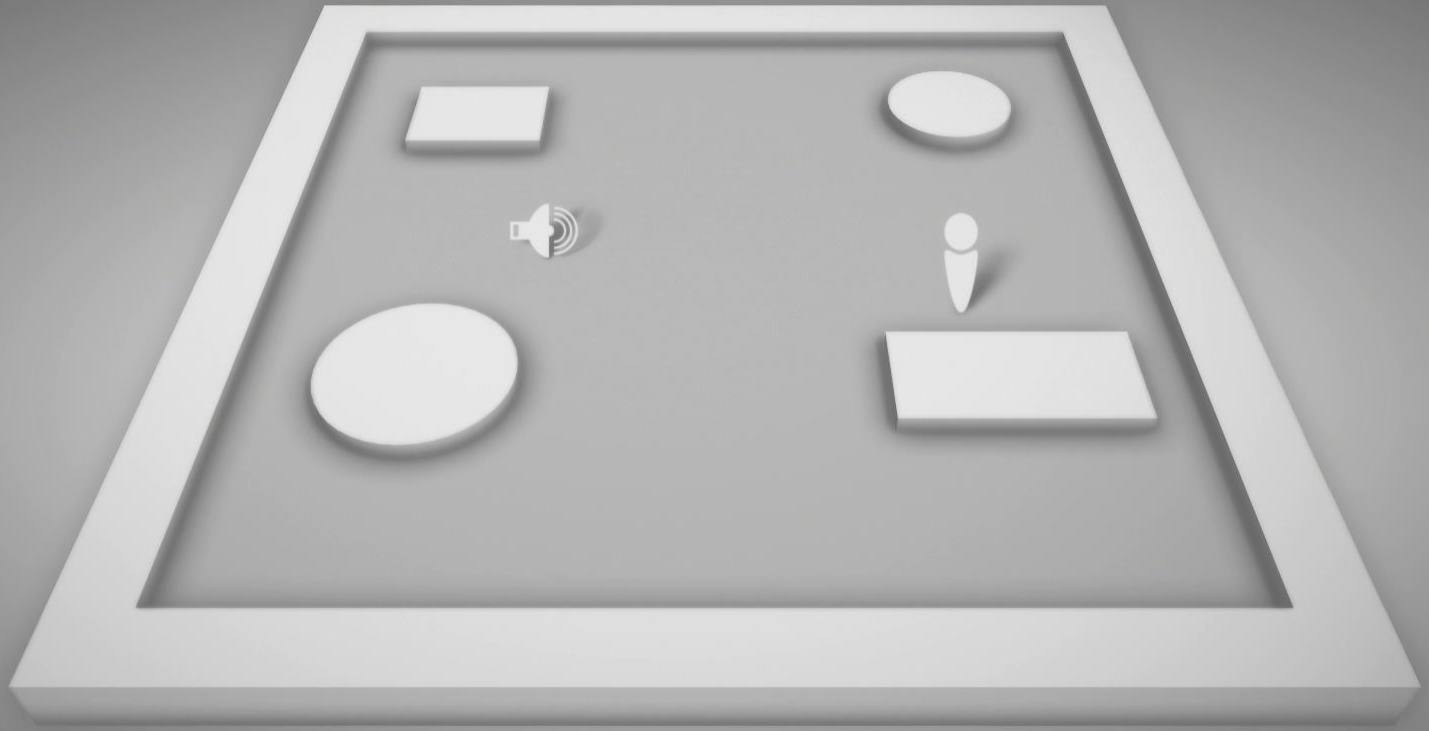
AUDIO REFLECTS YOUR ENVIRONMENT

Elevator

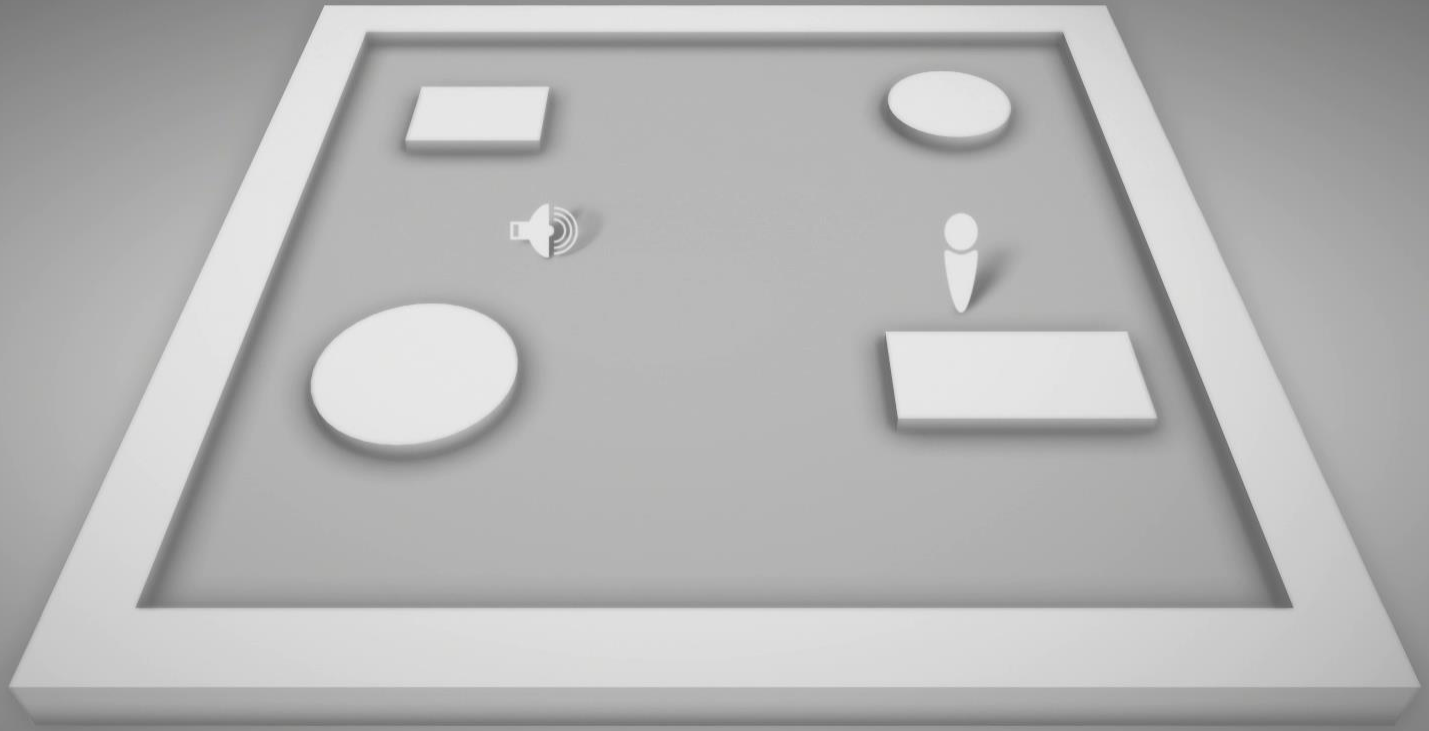
Symphony Hall

Meadow

NVIDIA VRWORKS AUDIO

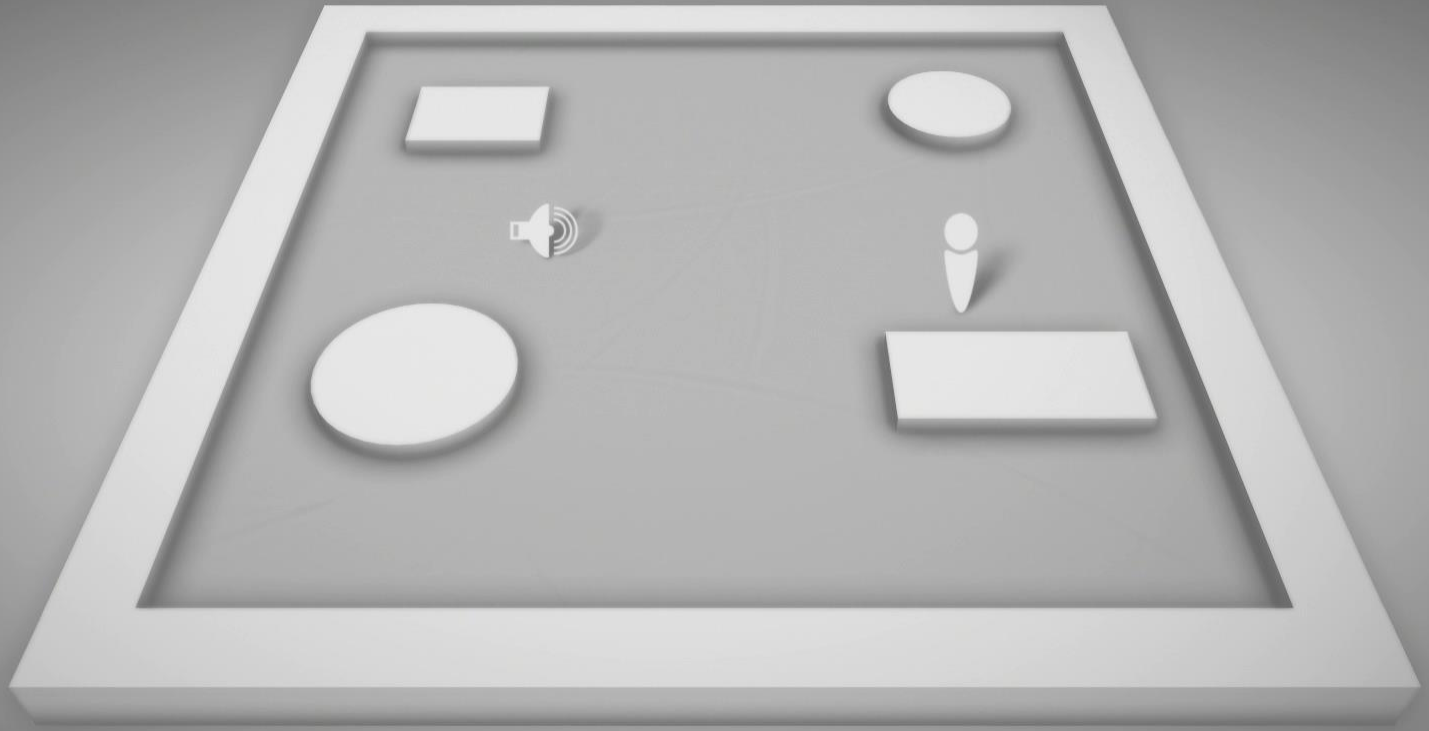


NVIDIA VRWORKS AUDIO



NVIDIA VRWORKS AUDIO

Models direction and propagation using Ray tracing



VRWORKS AUDIO SDK NOW AVAILABLE!

Features:

- Directionality and Reverbs
- Sound Propagation with Attenuation, Reflection, Diffraction, Transmission
- Realistic Occlusion
- Dynamic Geometry

Easy To Use Presets:

- Effect Strength
- Materials
- Compute

Available as SDK and Plugin for UE 4.15+

COMPUTING CHALLENGES IN REPRODUCING REALITY

GRAPHICS / DISPLAY

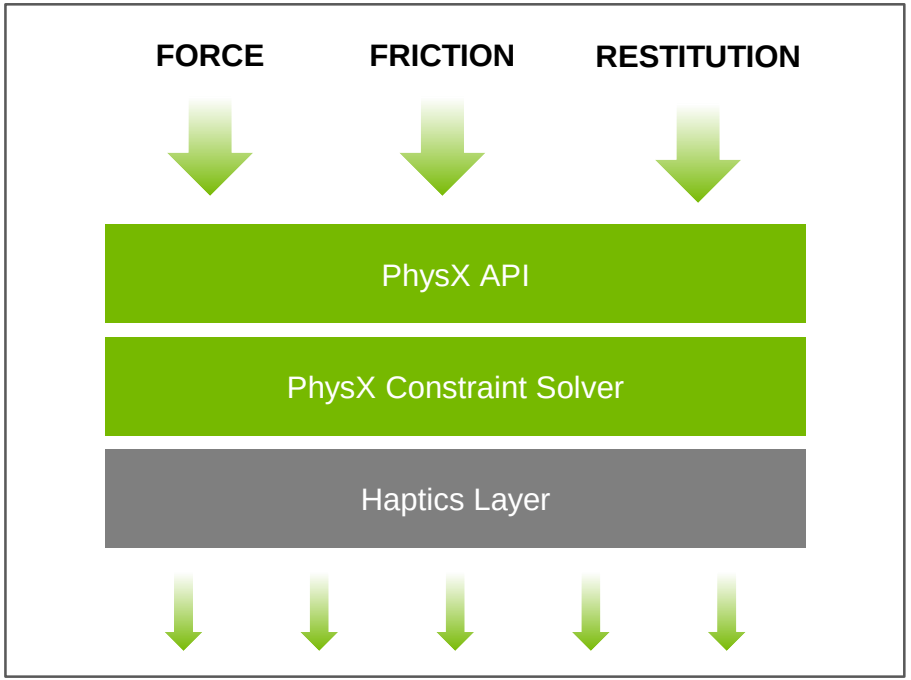
AUDIO

PHYSICS & TOUCH

CAPTURE

HAPTICS

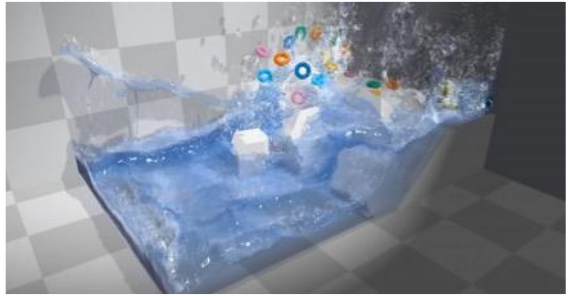
Collision detection & deformation modeling



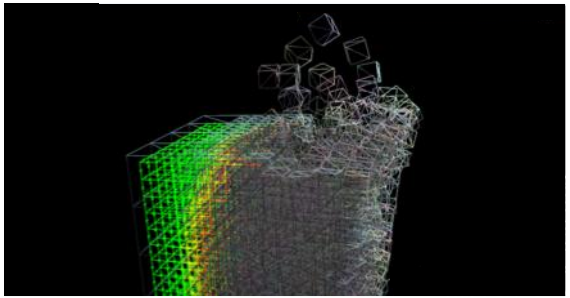
REALISTIC PHYSICS

Simulating behavior in VR

PHYSX

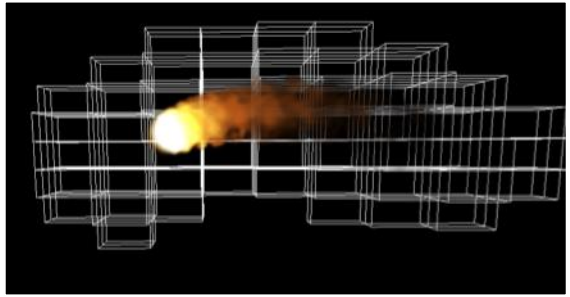


FLEX



DESTRUCTION

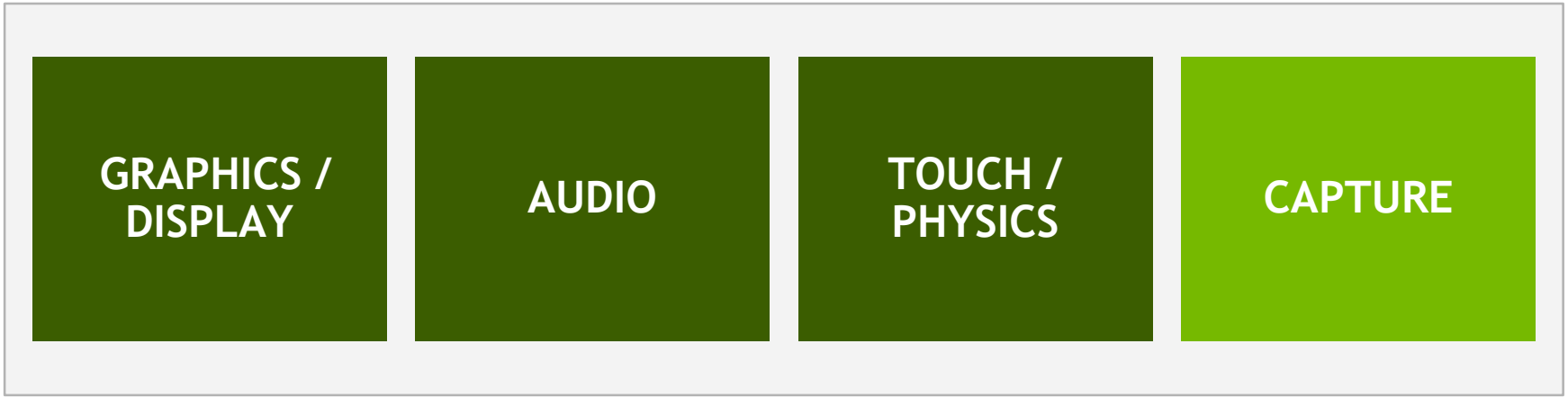
HAIRWORKS



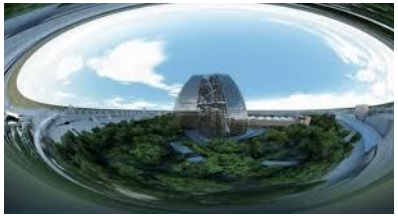
FLOW

CLOTH

COMPUTING CHALLENGES IN REPRODUCING REALITY



SIGNIFICANT COMPUTATION REQUIRED TO DELIVER 360 VIDEO



Capture

4k cameras

Stitch

Decode → Calibrate → Equalize → Stitch → Encode

Display

Single 360 video

VRWORKS 360 VIDEO SDK NOW AVAILABLE!

"The fact that NVIDIA manages to stitch 4K 360 stereoscopic video in real time making live streaming possible changes the production pipeline and enables entirely new use cases in VR"
Kinson Loo, CEO of Z CAM.

Features

- Real-Time & Offline Stitching
- Up to 32 x 4k Camera Rigs

GPU Accelerated Pipeline

Decode -> Calibration -> Equalization -> Stitching -> Encode

Mono SDK Available in Beta Now!

Stereo SDK Available Soon

VRWORKS FOR UE4 AVAILABLE NOW

Features:

- Multi-res Shading
- Lens Matched Shading
- Single Pass Stereo
- VR SLI

Available for 4.12 through 4.15

Download at:
developer.nvidia.com/nvidia-vrworks-and-ue4

VRWORKS FOR UNITY AVAILABLE NOW

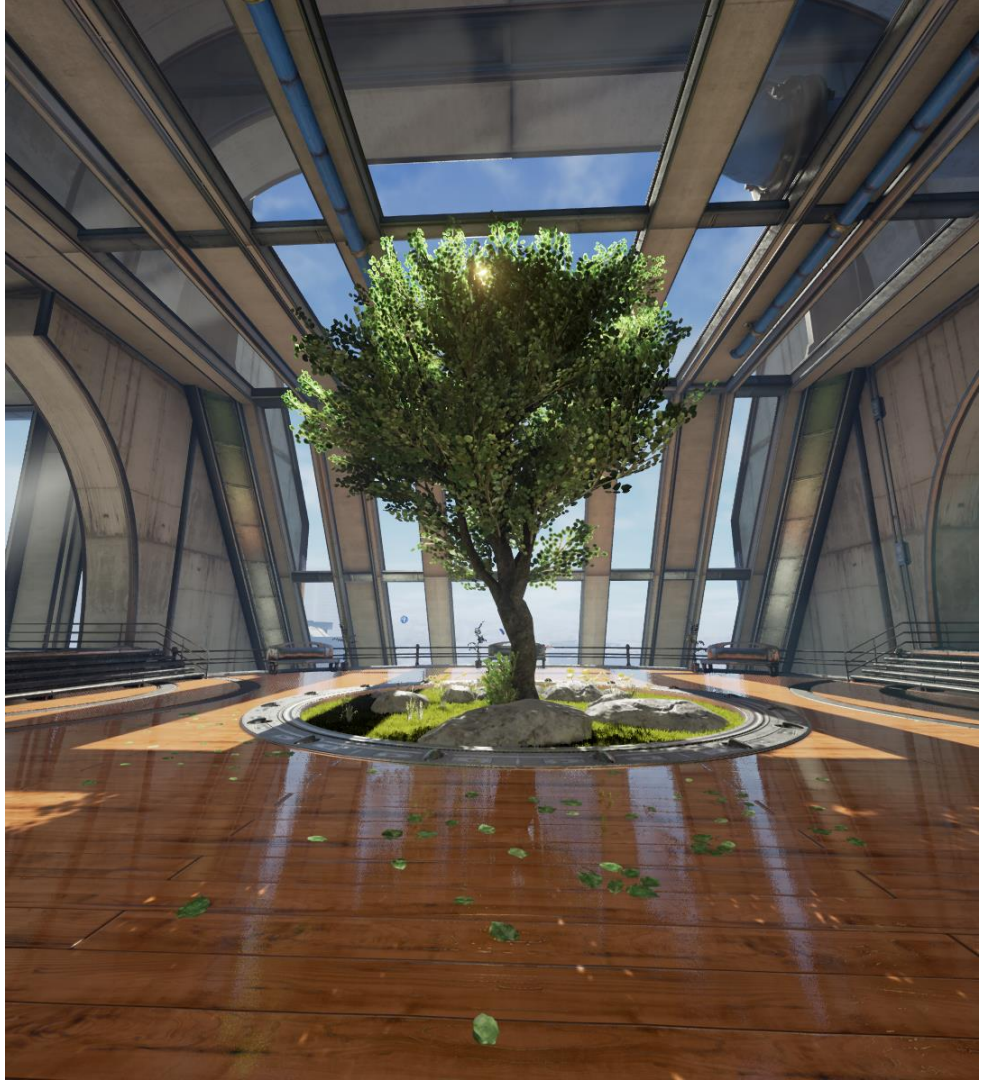
Features:

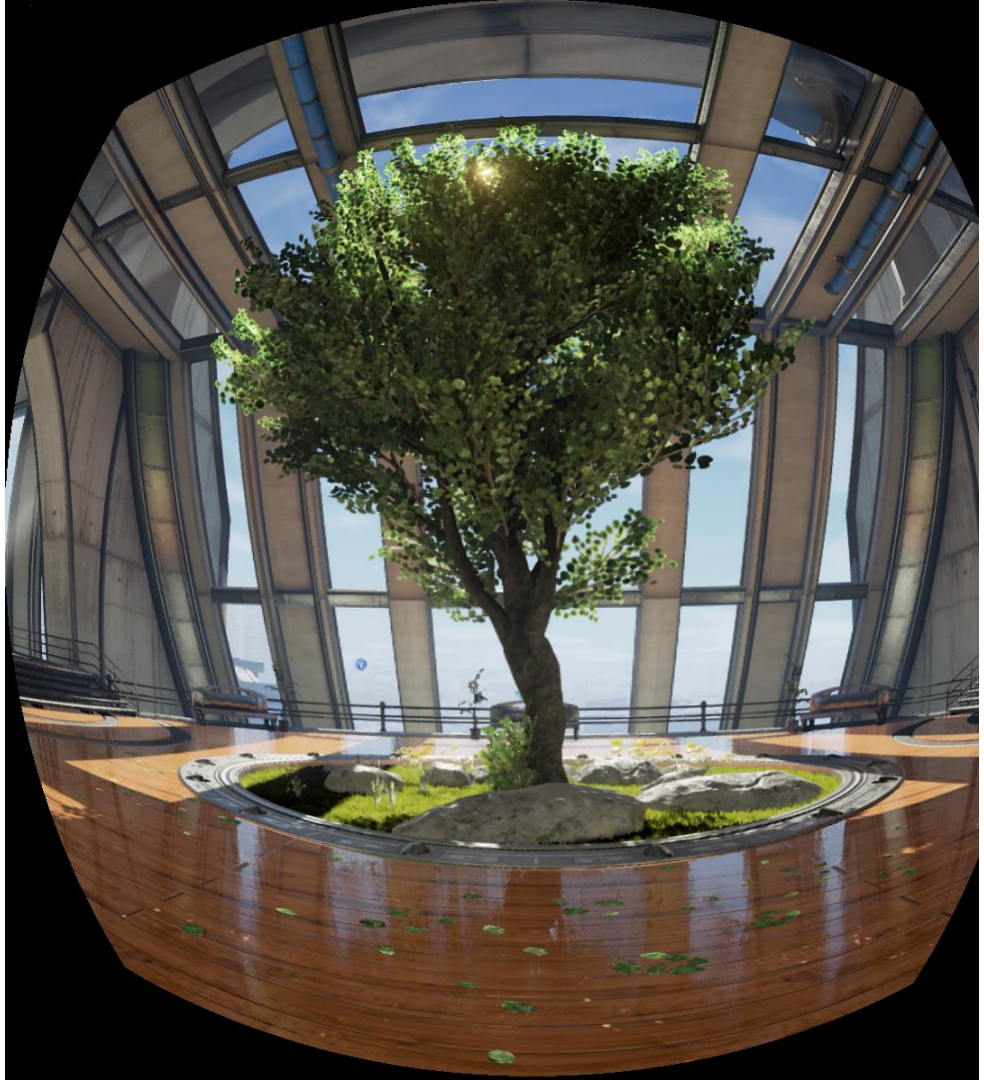
- Multi-res Shading
- Lens Matched Shading
- Single Pass Stereo
- VR SLI

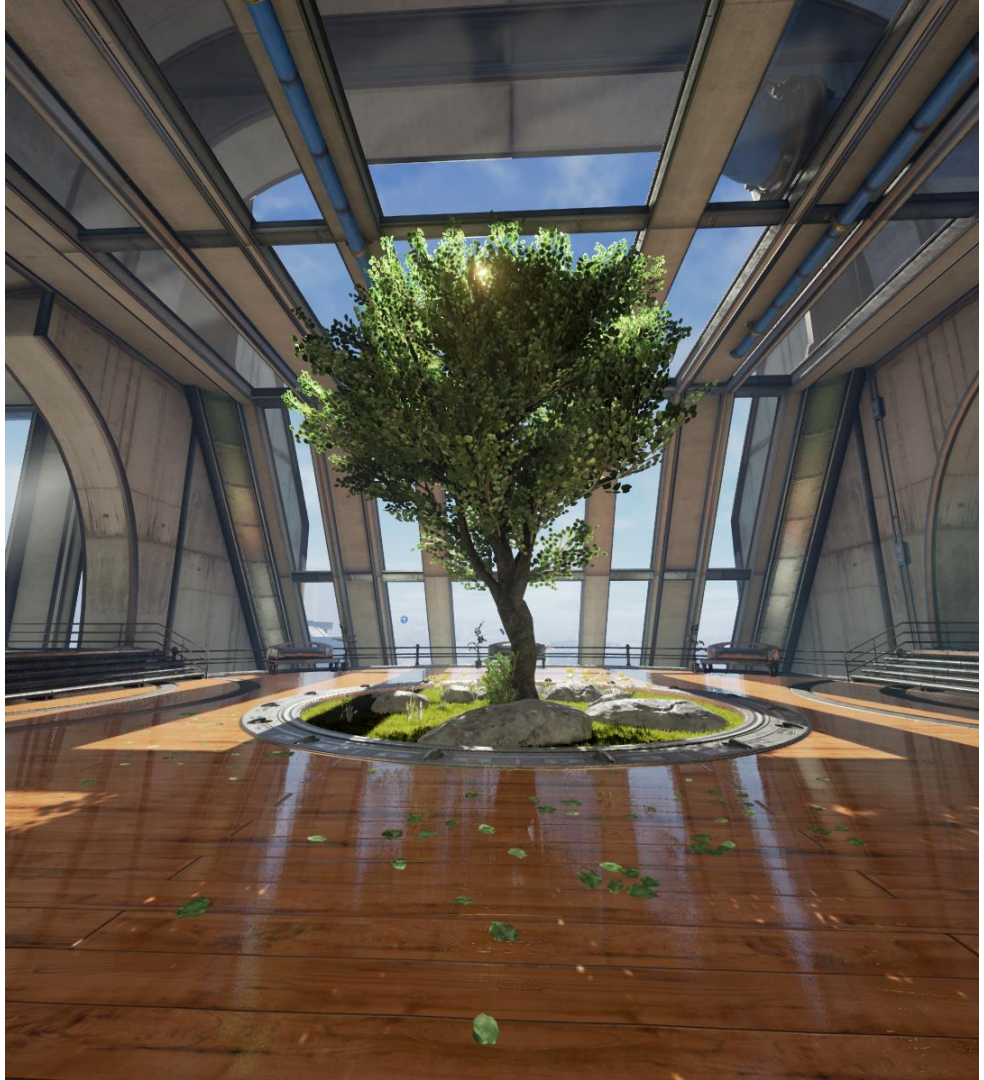
Supported in Unity 2017.1 Beta 2

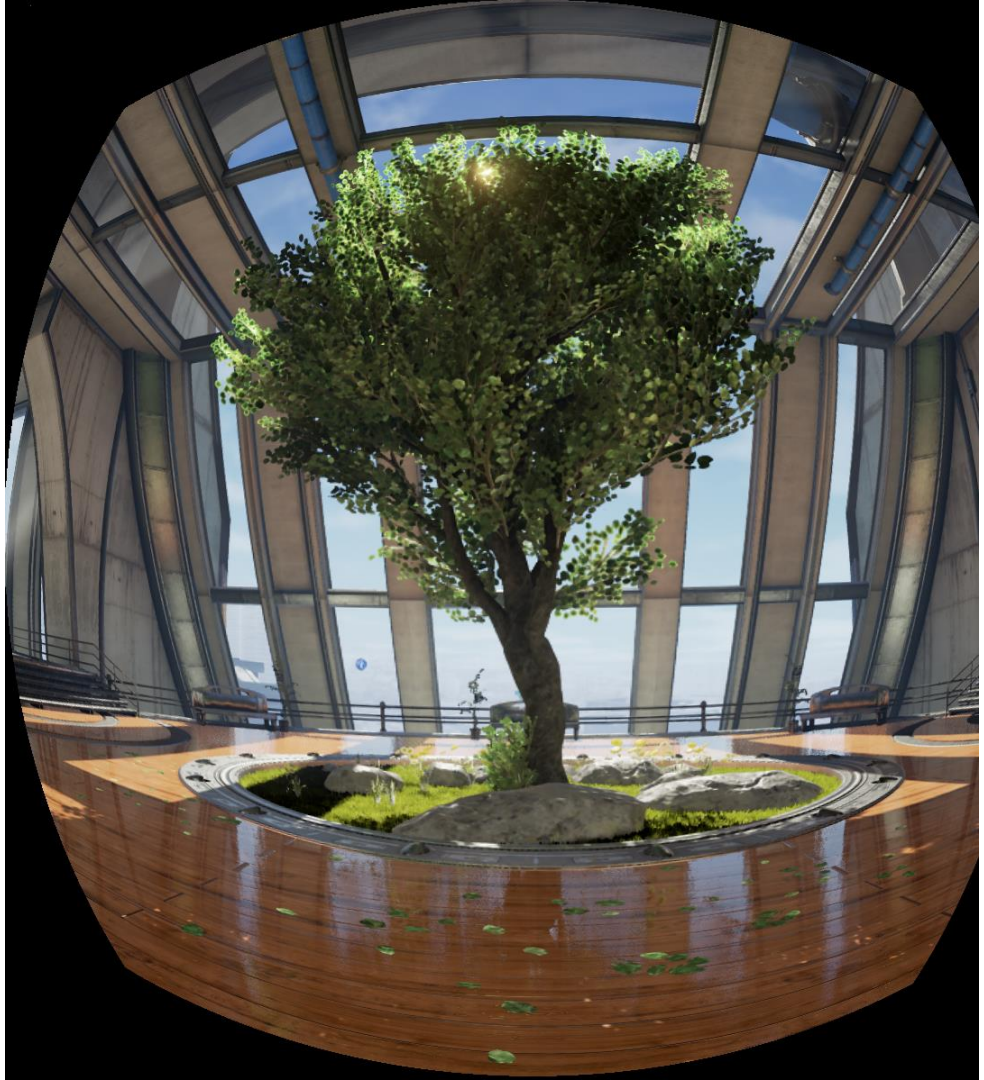
Download VRWorks plugin at www.assetstore.unity3d.com

LENS MATCHED SHADING DEEP DIVE







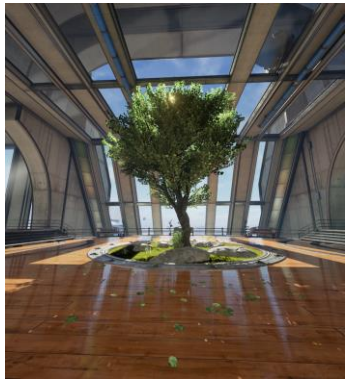


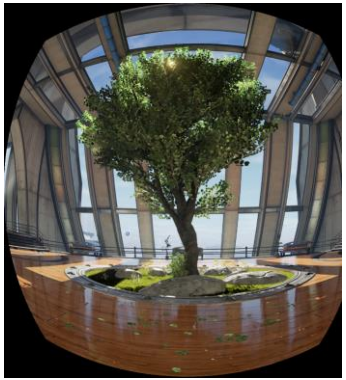
SHADING RATE AND LENS WARP

Two effects of applying lens warp to an image:

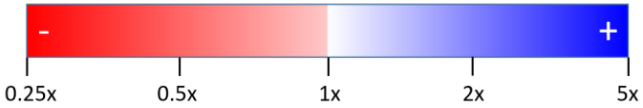
1. Periphery squashed
2. Central region magnified

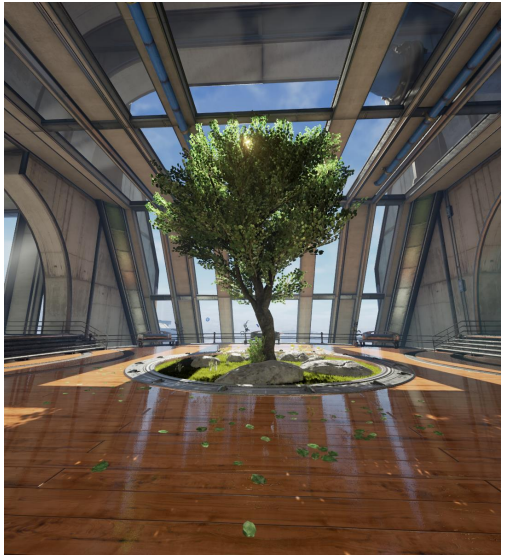
This distortion makes image center undersampled and periphery supersampled

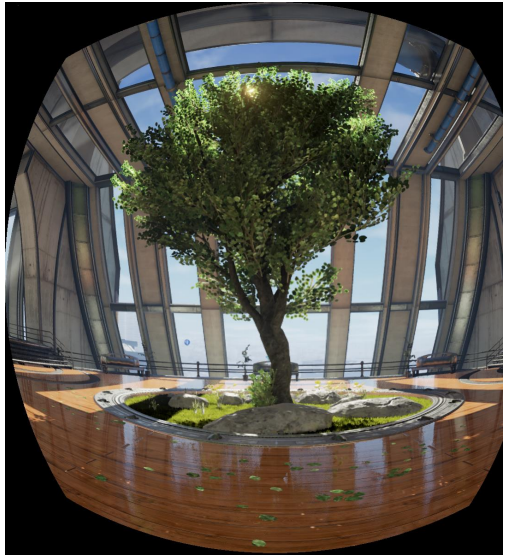


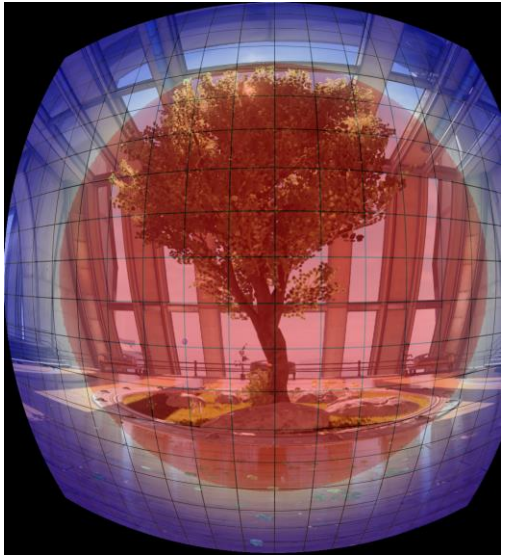


SHADING RATE AND LENS WARP









SHADING RATE AND LENS WARP

Therefore VR applications usually render at a higher resolution than display resolution.

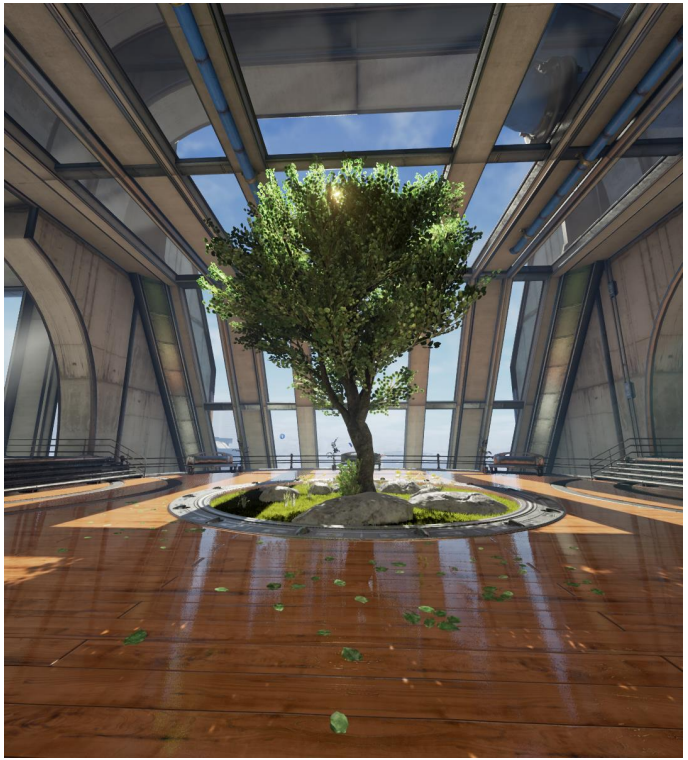
- HTC Vive

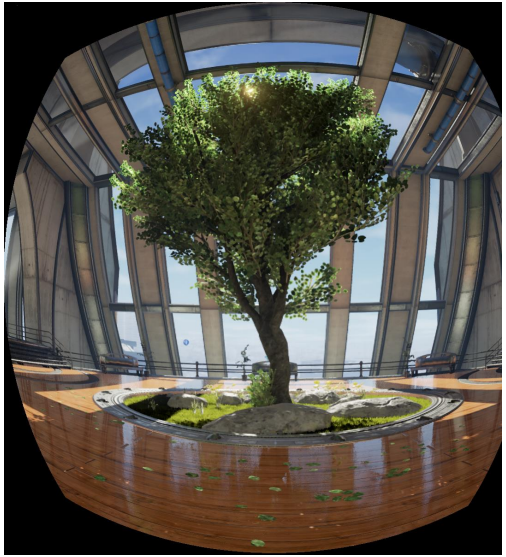
  Display Resolution 2160x1200, Render Resolution 3024x1680

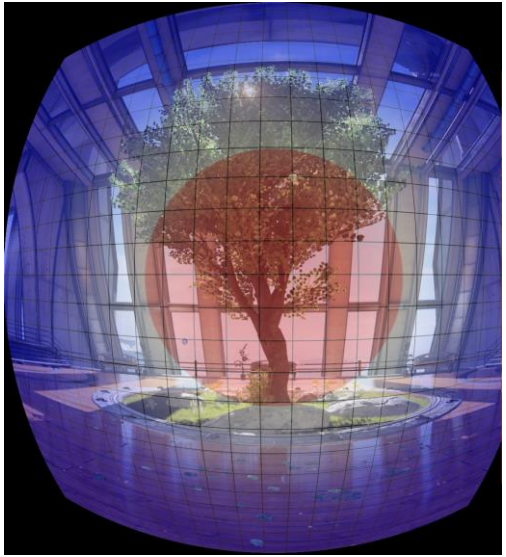
- Oculus Rift

  Display Resolution 2160x1200, Render Resolution 2664x1586

SHADING RATE AND LENS WARP







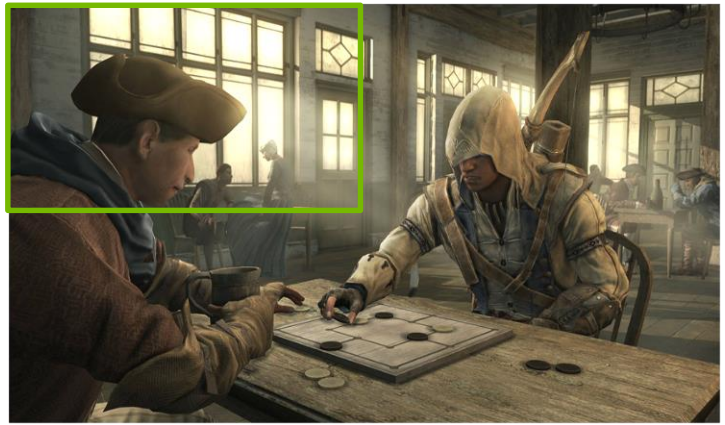
LENS MATCHED SHADING BREAKDOWN

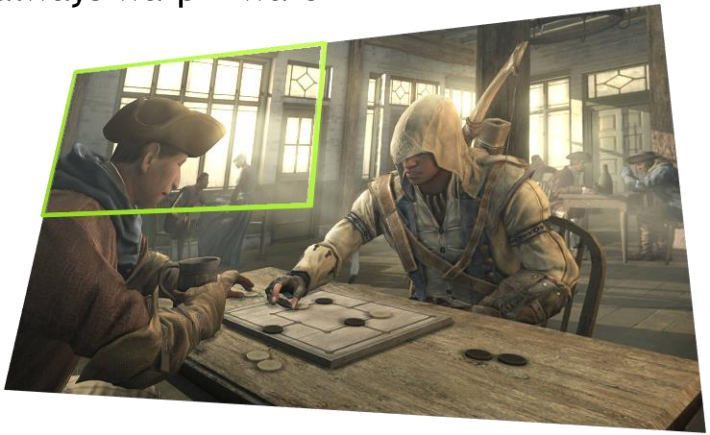
LMS approximate post lens warp shading rate by:

1. Enlarge the entire viewport to increase overall shading rate
2. Modify clip space w to reduce periphery shading while maintaining the center

   $w' = Ax + By + w$

3. Apply different coefficient A, B per quadrant to always warp inward





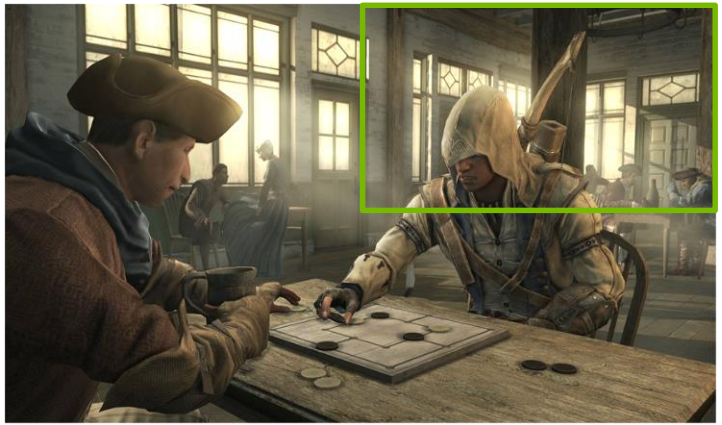
LENS MATCHED SHADING BREAKDOWN

LMS approximate post lens warp shading rate by:

1. Enlarge the entire viewport to increase overall shading rate
2. Modify clip space w to reduce periphery shading while maintaining the center

   $w' = Ax + By + w$

3. Apply different coefficient A, B per quadrant to always warp inward



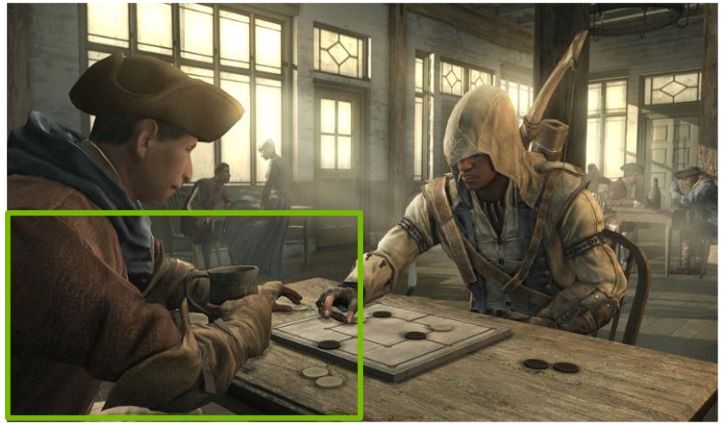
LENS MATCHED SHADING BREAKDOWN

LMS approximate post lens warp shading rate by:

1. Enlarge the entire viewport to increase overall shading rate
2. Modify clip space w to reduce periphery shading while maintaining the center

   $w' = Ax + By + w$

3. Apply different coefficient A, B per quadrant to always warp inward



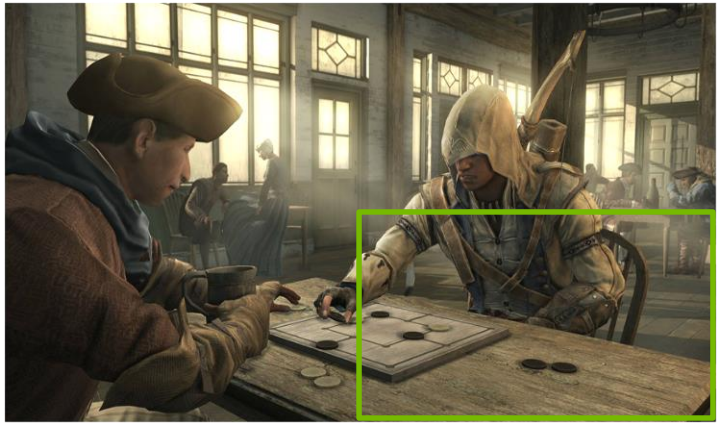
LENS MATCHED SHADING BREAKDOWN

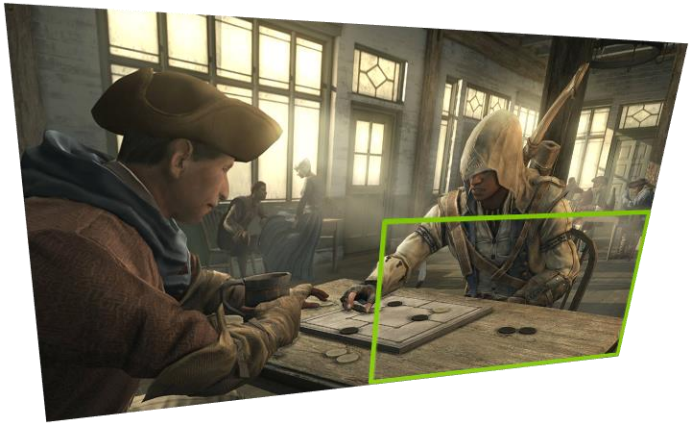
LMS approximate post lens warp shading rate by:

1. Enlarge the entire viewport to increase overall shading rate
2. Modify clip space w to reduce periphery shading while maintaining the center

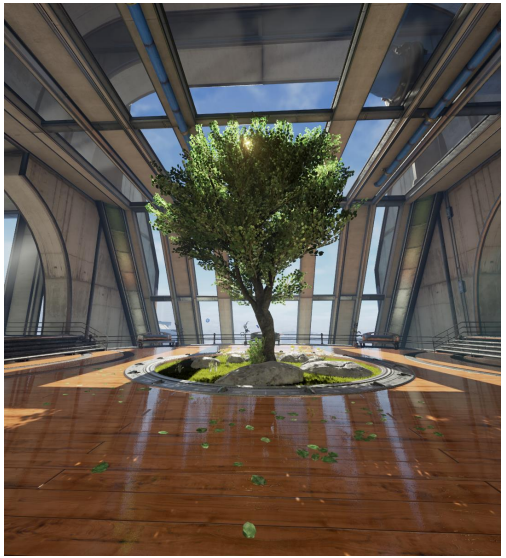
   $w' = Ax + By + w$

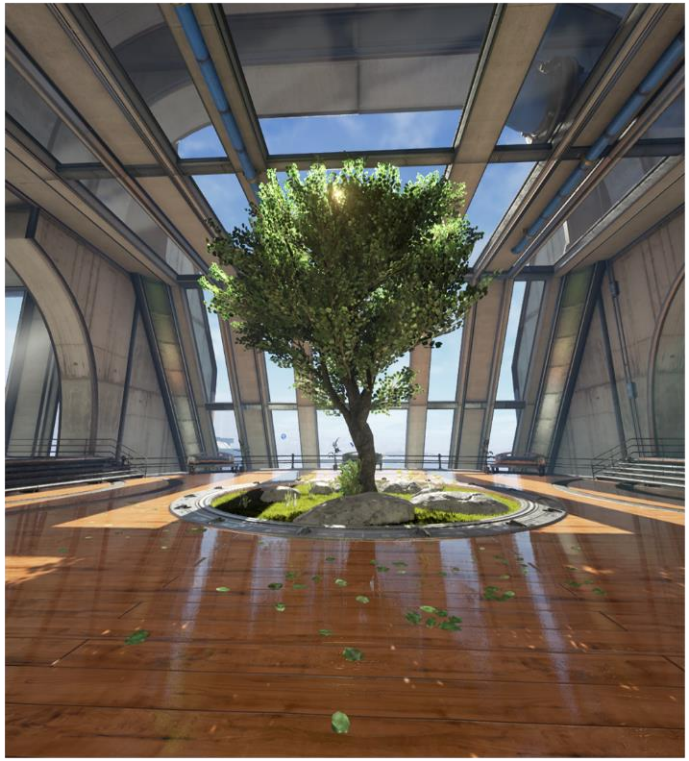
3. Apply different coefficient A, B per quadrant to always warp inward

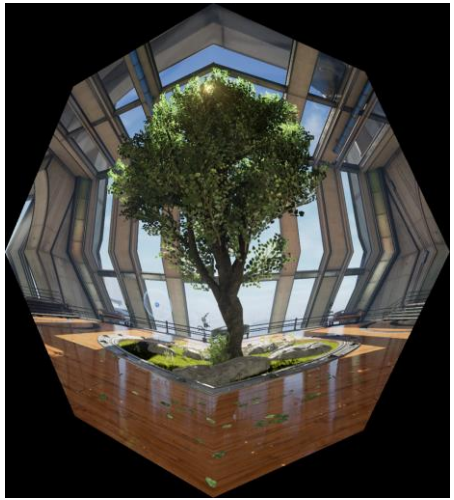




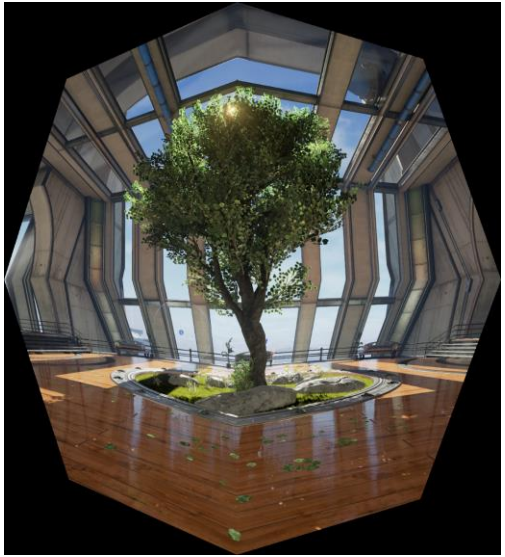
LENS MATCHED SHADING BREAKDOWN

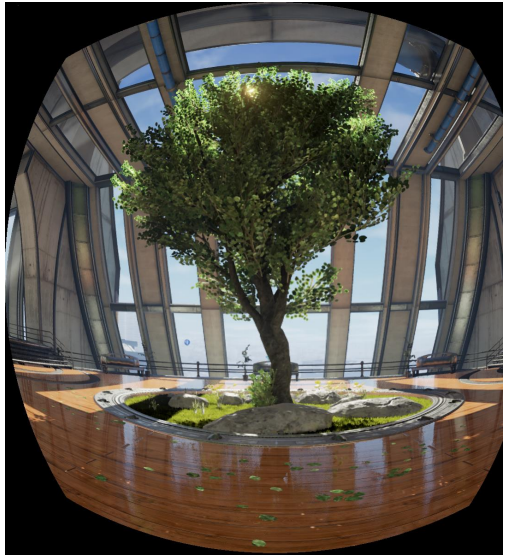


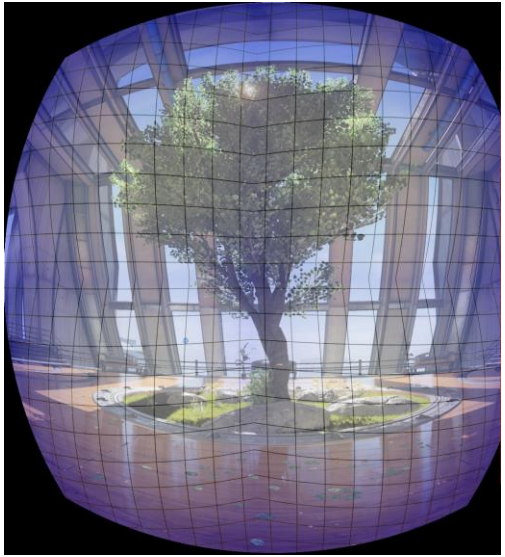


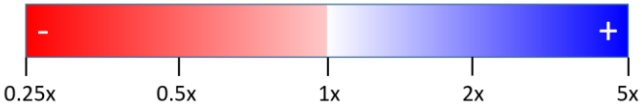


LENS MATCHED SHADING BREAKDOWN









PRE-SET CONFIGURATIONS

We provide 3 sets of configurations for both HTC Vive and Oculus Rift

1. Quality:

   No undersampling accross the image (while reducing total # of pixels)

2. Conservative:

   Undersampling no worse than baseline

3. Aggressive:

   ¾ Resolution of the Conservative config (keeping center shading rate high)

PRE-SET CONFIGURATIONS

We also provide a scalar variable that smoothly change the shading rate

It keeps the center shading rate constant

Allows finer grain tuning between image quality and performance

LMS VS. MRS

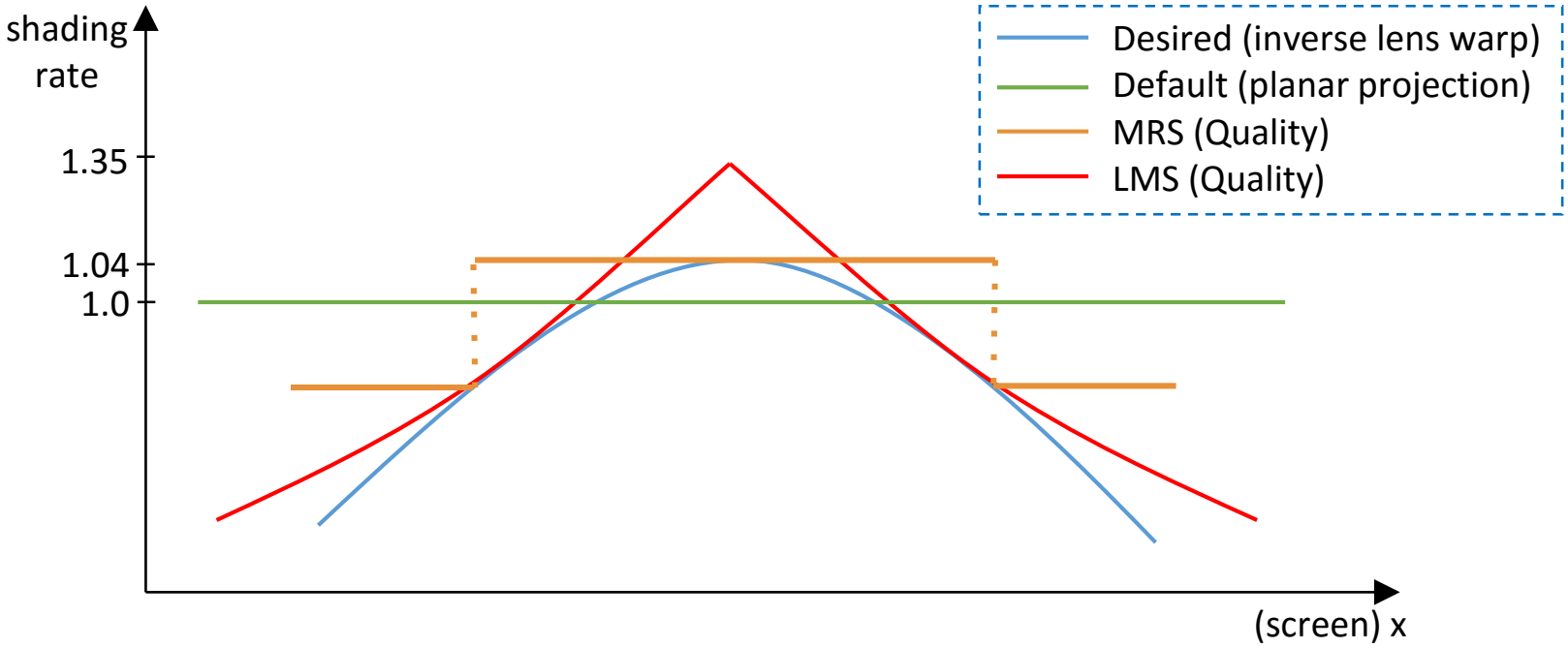
The 1/x profile of LMS can closer approximate the lens profile than the piecewise constant of MRS

- LMS needs fewer shading to achieve the same image quality with MRS
- LMS has a smoother shading rate transition across the image

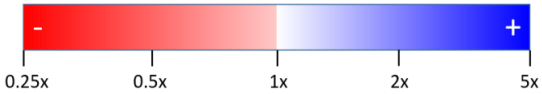
LMS uses 4 viewports per eye, while MRS uses 9. This makes LMS easier to work along with Instanced Stereo and Single Pass Stereo

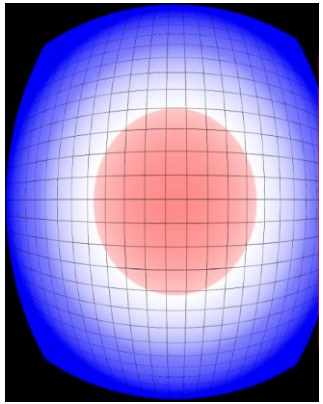
- Fewer viewports also benefits performance

LMS VS. MRS

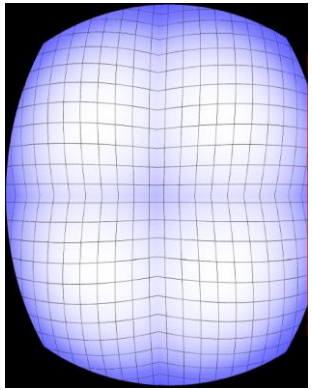
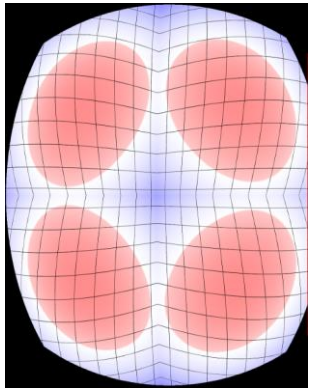
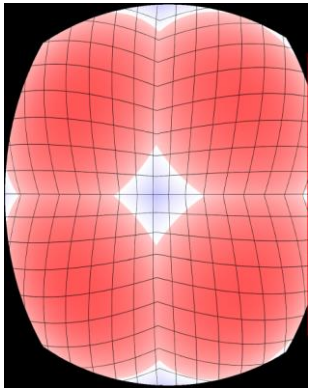
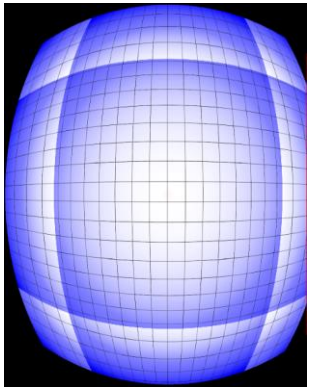
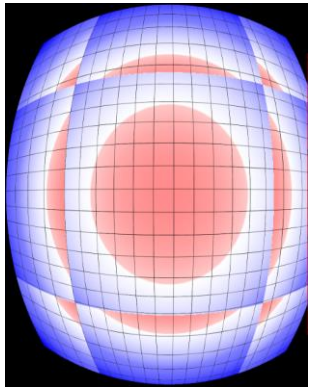
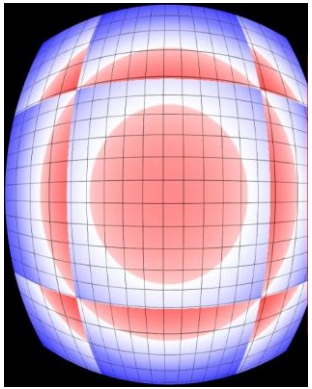

shading rate
1.35
1.04
1.0
(screen) x
Desired (inverse lens warp)
Default (planar projection)
MRS (Quality)
LMS (Quality)

LMS vs. MRS

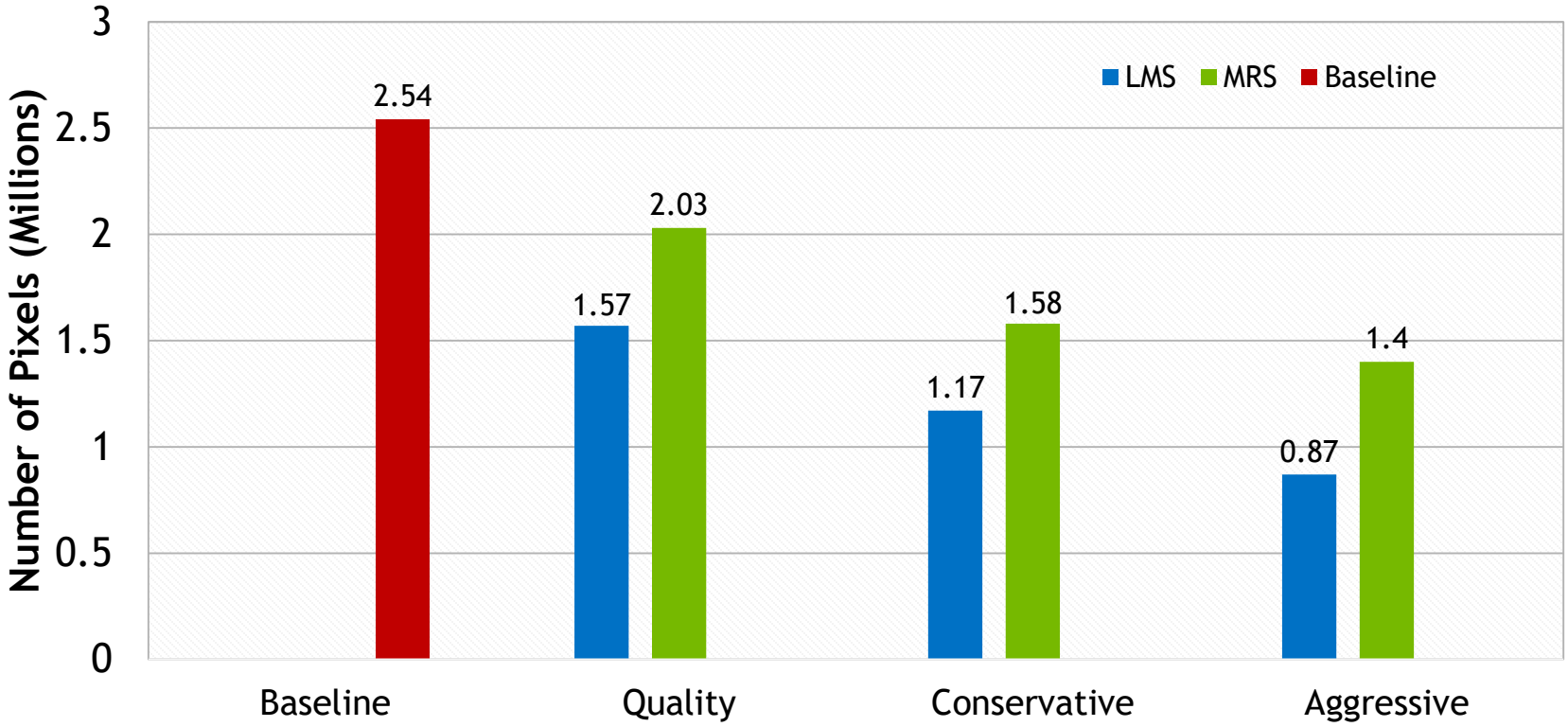




Baseline
(no warp)
2.54 MPix / eye

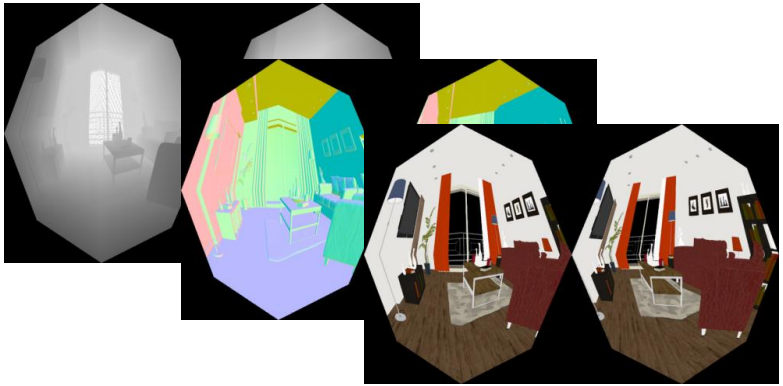
	Quality (no undersampling)	Conservative (no worse than baseline)	Aggressive (3/4 Reso. of conservative)
LMS	 1.57 MPix / eye	 1.17 MPix / eye	 0.87 MPix / eye
MRS	 2.03 MPix / eye	 1.58 MPix / eye	 1.40 MPix / eye

LMS/MRS Shading Rate Comparison



ENGINE INTEGRATION OVERVIEW

1. Fill Gbuffer in LMS Space

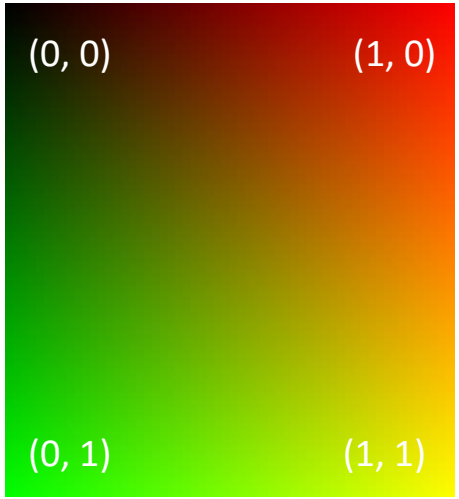


2. Shading in LMS Space

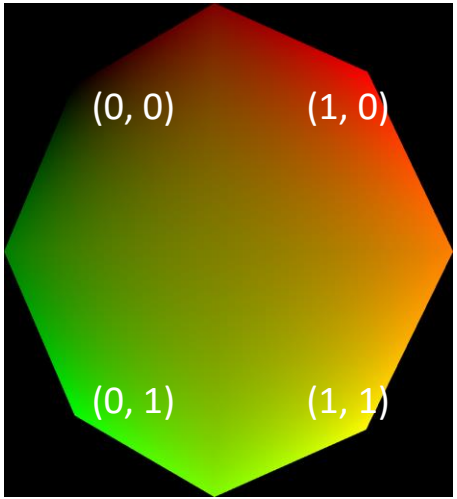
3. Post Processing in LMS Space

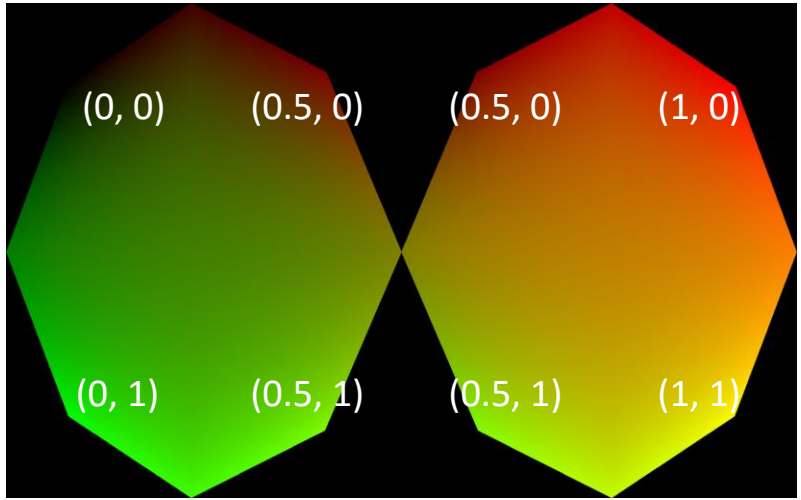
4. Resample to Linear Space

LMS COORDINATES SYSTEM



Linear Space UV





LMS Space Stereo UV

RENDER PASSES

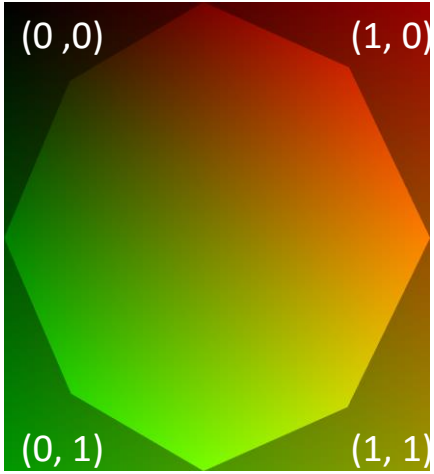
- Passes that renders geometries
  - Need to enable w modification, bind fast geometry shaders and set up multiple viewports & scissors before draw call submission, for which we have provided helper functions.

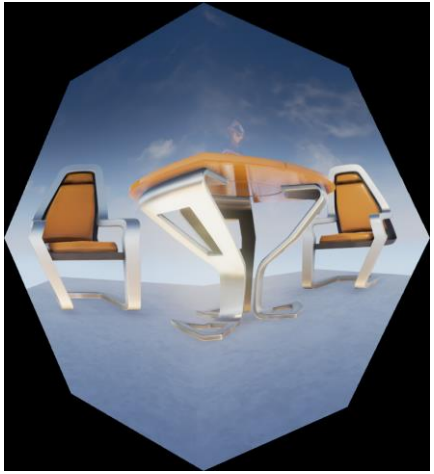
```
View.BeginVRProjectionStates(RHICmdList);
// Invoke draw calls..
View.EndVRProjectionStates(RHICmdList);
```

- Screen space passes, eg. SSR, SSAO and other post processing
  - Invoke full view rendering by rendering an octagon

FULL SCREEN OCTAGON

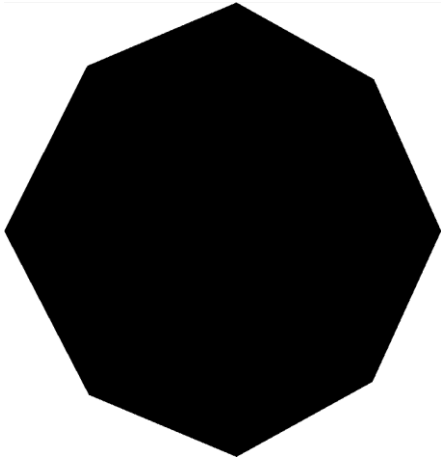
- Normally, full screen quads are used to invoke shading for most screen space passes
- With LMS, full screen octagon is used to avoid drawing to pixels outside of the valid region
- The UV in the octagon should span linearly across the view.

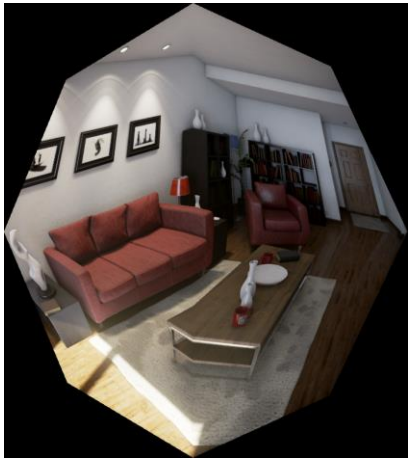




BOUNDARY MASK

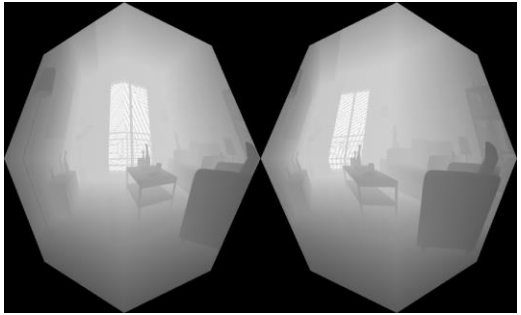
- A boundary mask is rendered during Z pre-pass to avoid rendering to pixels outside of the octagon region
- Need to carefully bind depth buffer for passes that did not have it bound

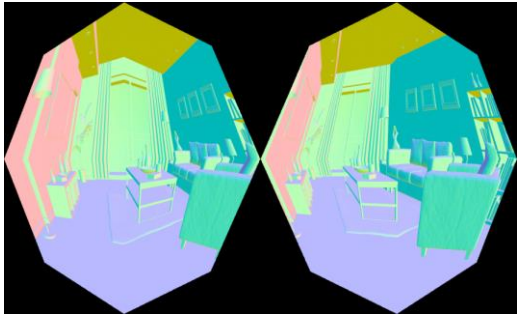


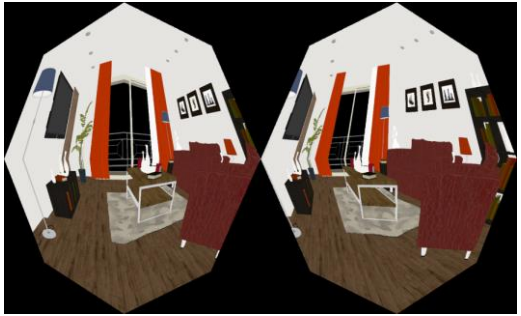


BASE PASS

- Fast GS, multiple viewport & scissors, w modification
  - Z values used in base pass PS needed to be remapped from LMS space to linear
    - Modified `FHLSLMaterialTranslator::PixelDepth()` so the generated code produce Z value in linear space
- Instanced Stereo requires more changes

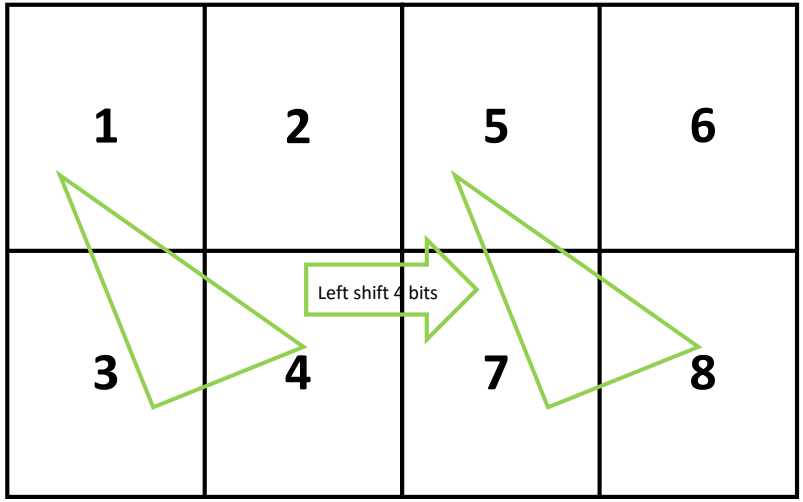






INSTANCED STEREO

- Using a single viewport that encompasses both view breaks W modification
- We set up 8 viewports and rely on multiprojection to send primitives to the correct viewports.

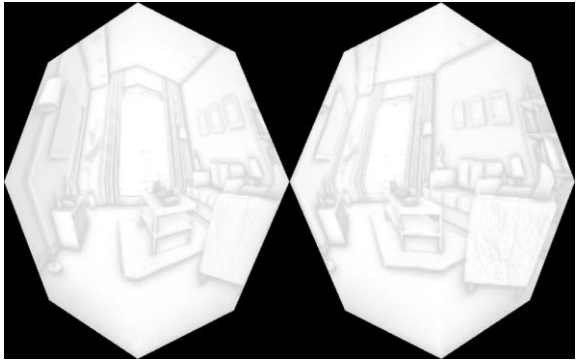


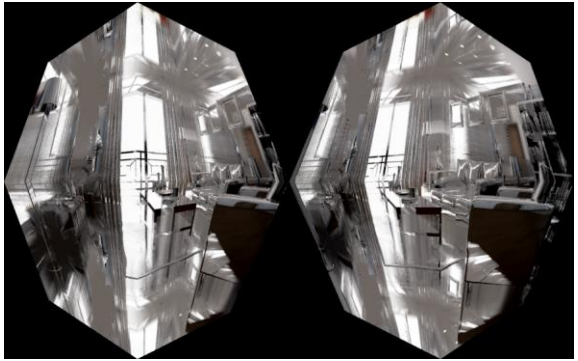
DEFERRED LIGHTING

- Directional light
  - Use full screen octagon to invoke shading
- Point and spot light
  - Apply w modification when render light volume geometries
  - Boundary mask does not work when camera is inside light volume, Z test disabled
  - Set boundary mark in the stencil buffer instead
- Dynamic shadows
  - Shadow map generation not affected. Apply w modification to Shadow projection.
- Tiled based lighting
  - Kill thread group if all threads are outside of the octagon covered region

SCREEN SPACE REFLECTIONS AND AO

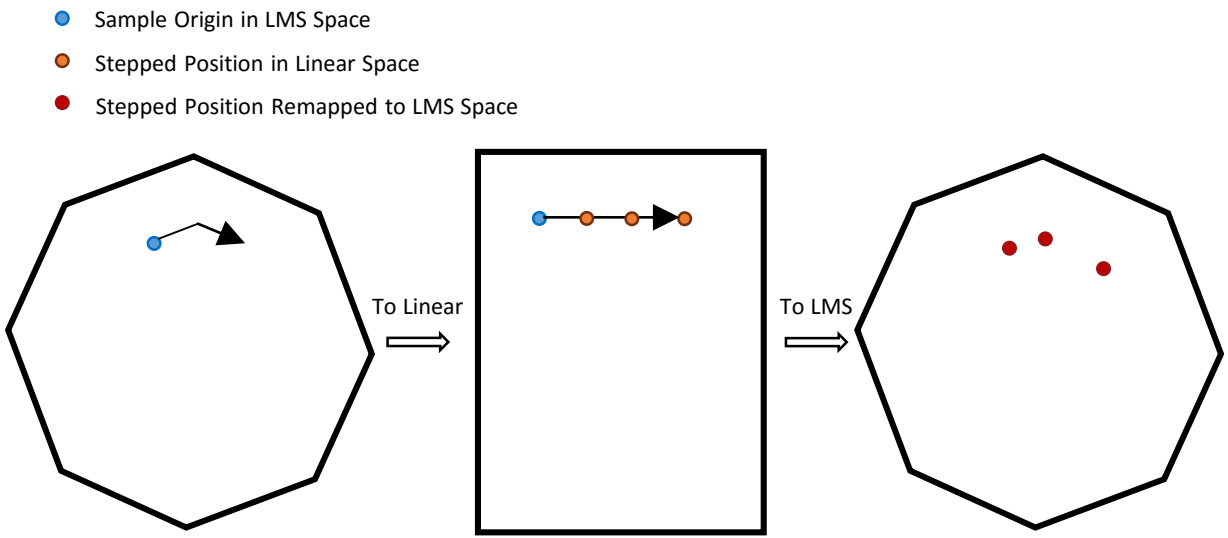
- Invoke shading with full screen octagon
- SSR and SSAO both samples the HZB
  - The Z values are remapped to linear space at HZB creation time to avoid sampling time overhead





SCREEN SPACE REFLECTIONS AND AO

- It's difficult to apply marching offset in LMS space, so we transform sample coordinates to linear space before applying offset in linear space. Remap it back to LMS space for sampling



COORDINATE REMAP IN SHADERS

- All PS input data are in LMS space (eg. SvPosition, ScreenPos, UV)
- Fetching from GBuffer should use LMS space coordinate
  - Gbuffer indexed in LMS space, so we can directly use UV passed in from Octagon VS to fetch Gbuffer
- Use linear space coordinates to do any world space compupation
  - Therefore PS input needs to be remapped to linear space before doing computation with GBuffer data
  - GBuffer content are in linear space, or world space

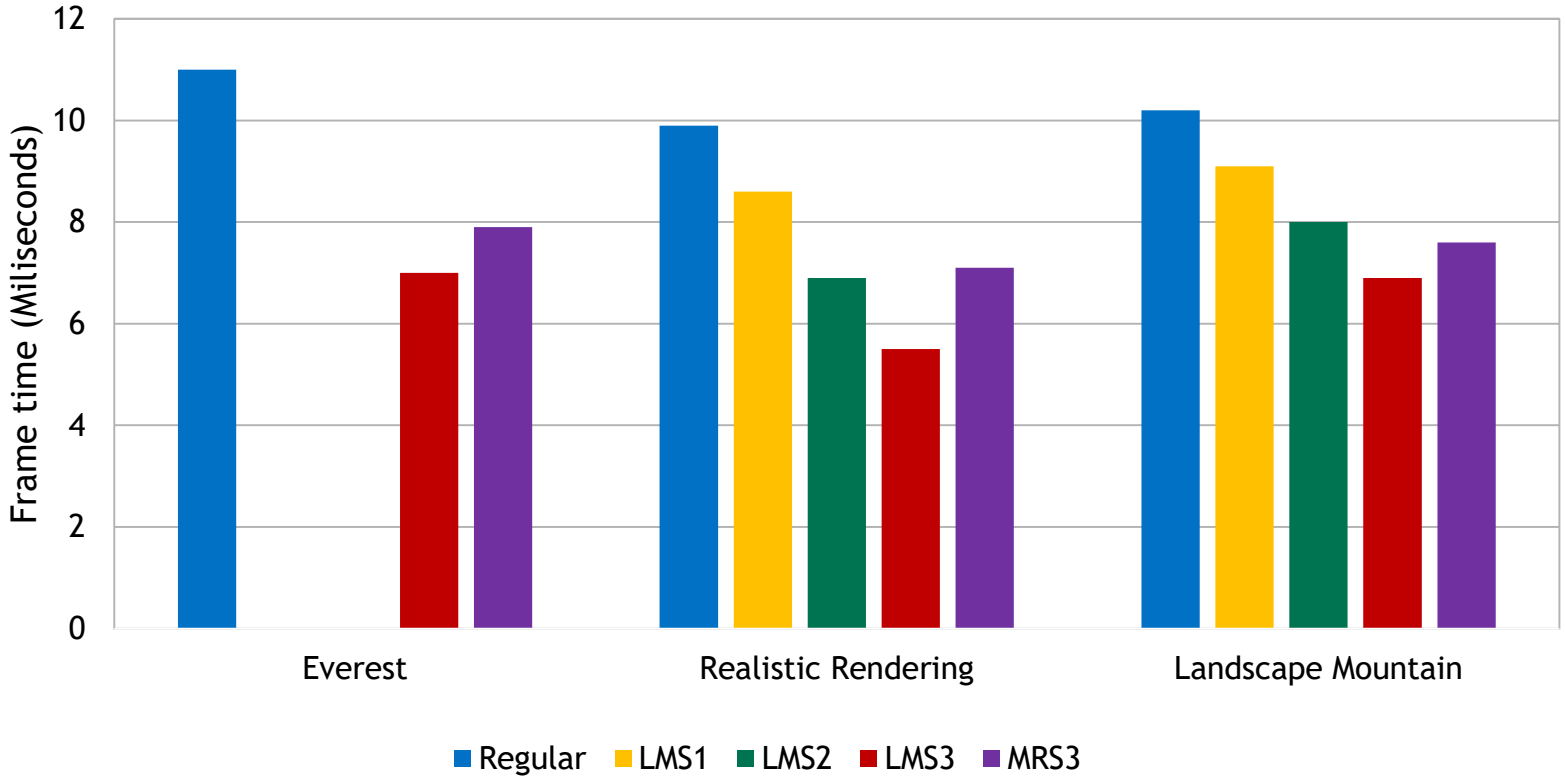
SHADER MODIFICATION EXAMPLE

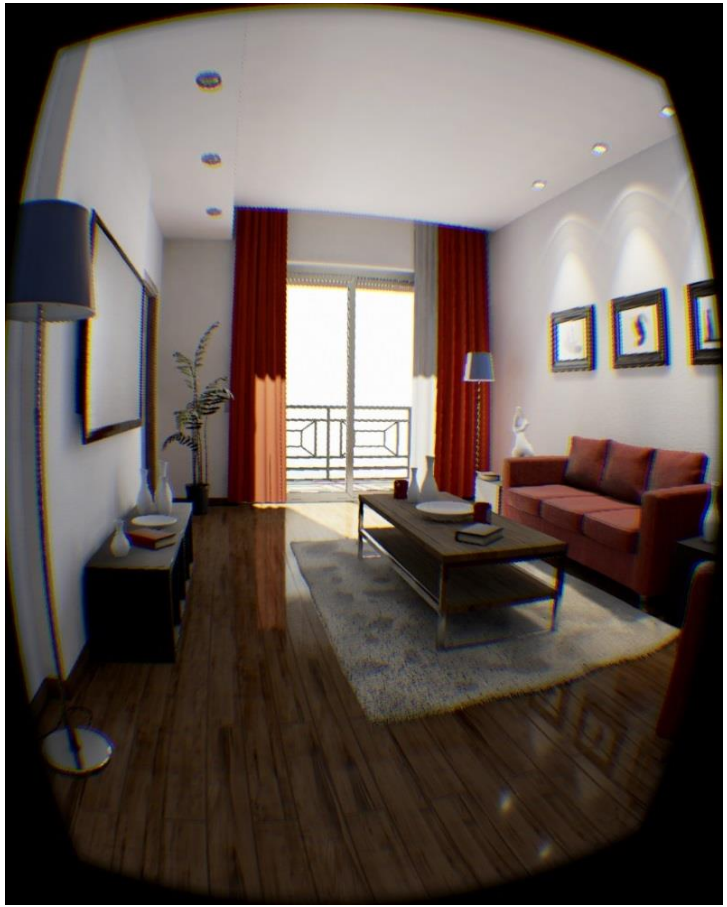
```
/**
 * Pixel shader for rendering a directional light using a full screen quad.
 */
void DirectionalPixelMain(
    float2 InUV : TEXCOORD0,
    float3 ScreenVector : TEXCOORD1,
    float4 SVPos : SV_POSITION,
    out float4 OutColor : SV_Target0
    )
{
    OutColor = 0;

    if (VRProjectionIsActive())
    {
        float4 LinearSvPos = SvPositionToLinearSvPosition(SVPos);
        ScreenVector = mul(float4(LinearSvPos.xy, 1, 1), View.SVPositionToTranslatedWorld).xyz;
    }

    float3 CameraVector = normalize(ScreenVector);

    FScreenSpaceData ScreenSpaceData = GetScreenSpaceData(InUV);

    // Only light pixels marked as using deferred shading
    BRANCH if( ScreenSpaceData.GBuffer.ShadingModelID > 0
#if USE_LIGHTING_CHANNELS
        && (GetLightingChannelMask(InUV) & DeferredLightUniforms.LightingChannelMask)
#endif
        )
    {
        float SceneDepth = CalcSceneDepth(InUV);
        float3 WorldPosition = ScreenVector * SceneDepth + View.WorldCameraOrigin;

        FDeferredLightData LightData = SetupLightDataForStandardDeferred();

        uint2 Random = ScrambleTEA( uint2( SVPos.xy ) );
        Random.x ^= View.Random;
        Random.y ^= View.Random;

        OutColor = GetDynamicLighting(WorldPosition, CameraVector, ScreenSpaceData.GBuffer,
            ScreenSpaceData.AmbientOcclusion, ScreenSpaceData.GBuffer.ShadingModelID,
            aLightData, GetPerPixelLightAttenuation(InUV), Random);
    }
}
```

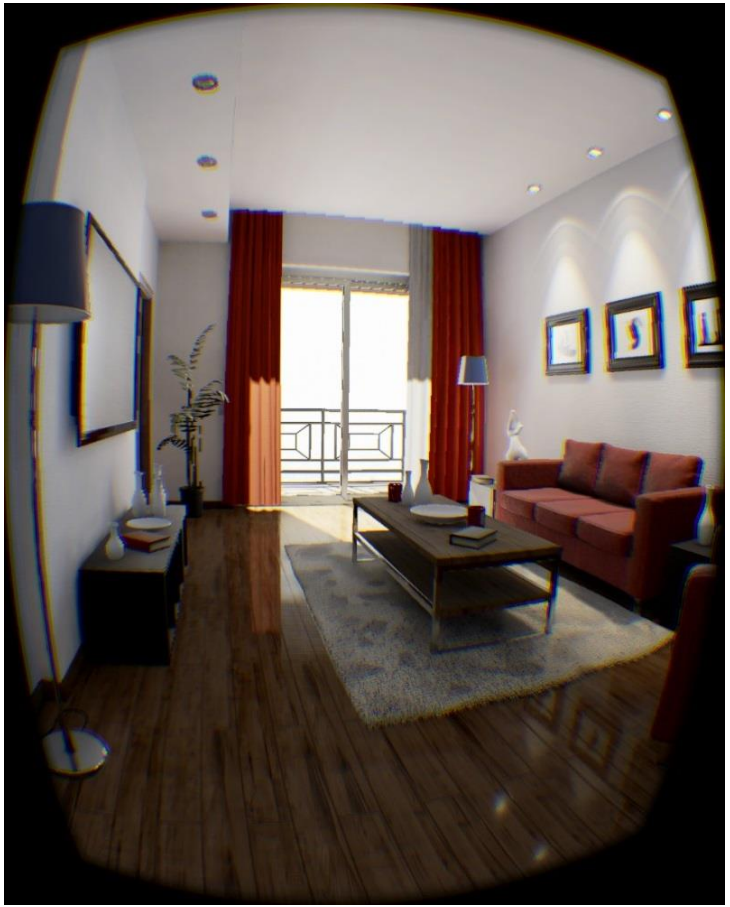
The input SVPos is in LMS space.
So convert it to linear space, since CameraVector is used to calculate lighting with GBuffer data, which is also in linear space.

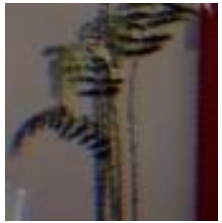
InUV is LMS space.
When fetching data from GBuffers, use LMS space coordinates directly
Since GBuffer is indexed in LMS space.

LMS/MRS Performance Benchmark





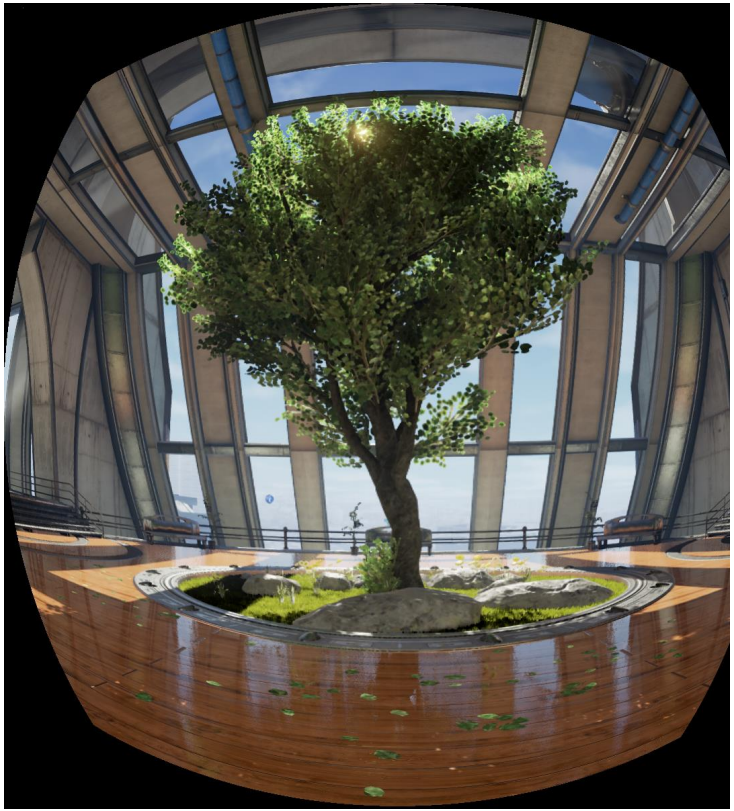


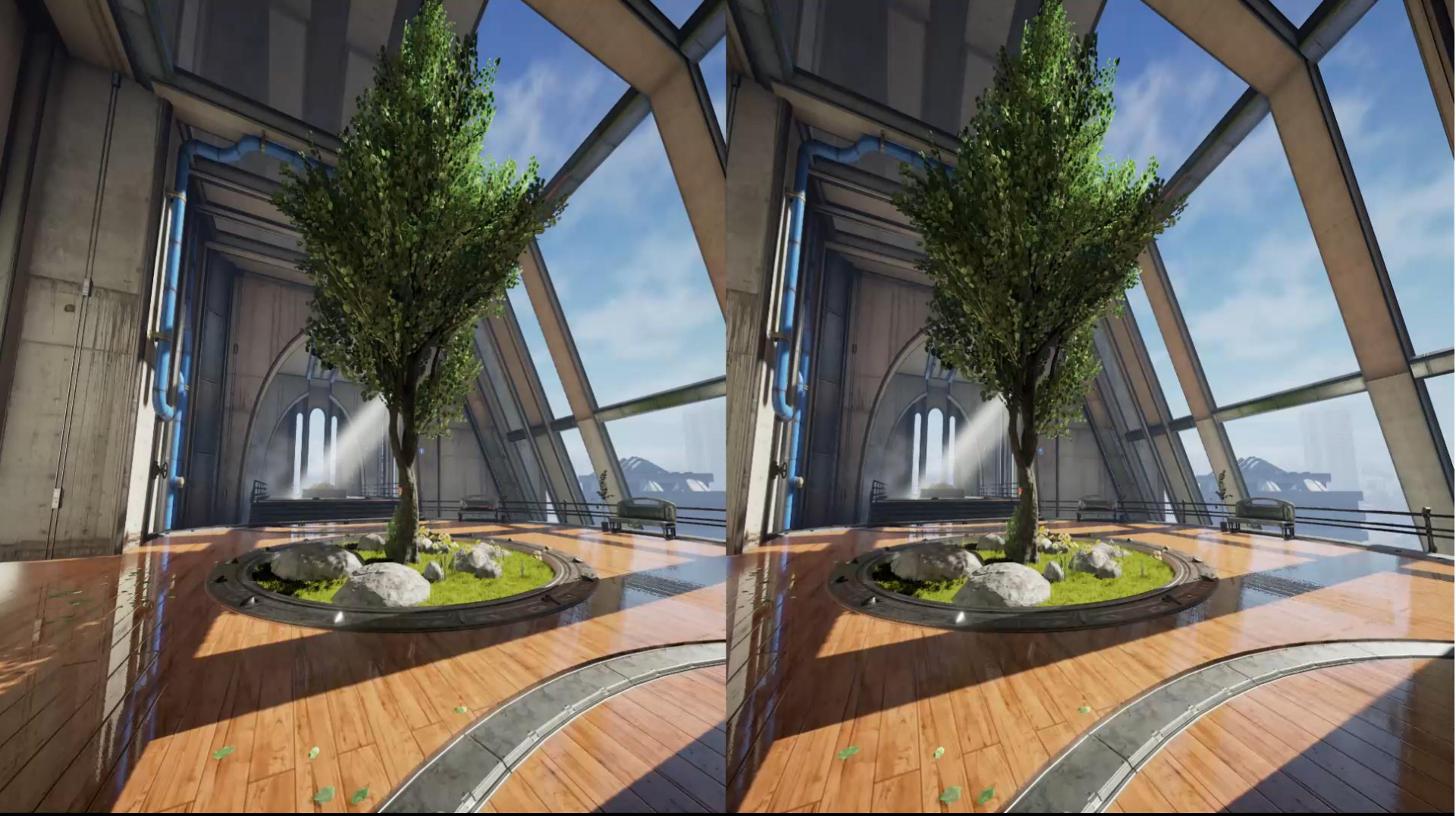


NOTE ON PERFORMANCE

- Pixel shading is only part of the frame
  - LMS won't help with geometry or CPU work
- Linear and LMS coordinate conversion isn't free
  - Nonideal in passes like SSAO, SSR
- Linear resampling at the end isn't free
  - We do this at high resolution to keep the center sharpness

DEMO





Thank You!

cem@nvidia.com

edliu@nvidia.com

developer.nvidia.com/vr