GPU Architectures for Next Generation Rendering

Karthik Vaidyanathan

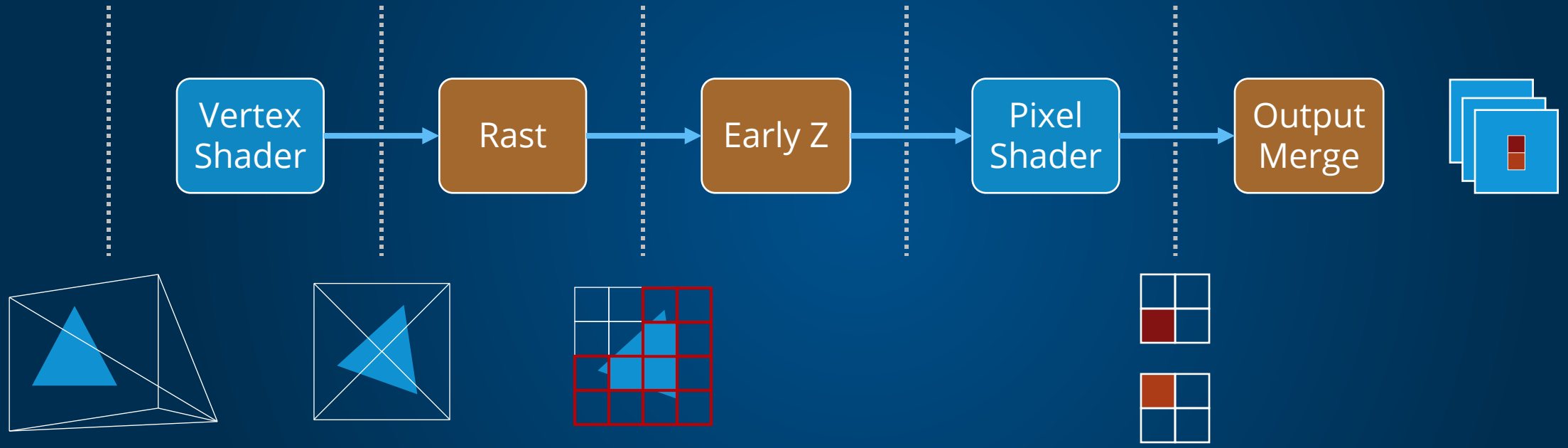
Advanced Rendering Technologies (ART)

Karthik.vaidyanathan@intel.com

Introduction to ART

- Part of Intel’s GPU architecture team
- Two distinct roles
  - Publish research to advance the field of real time graphics
  - Influence Intel’s GPU roadmap to achieve graphics leadership

The GPU Rendering Pipeline



What are Quad Fragments ?

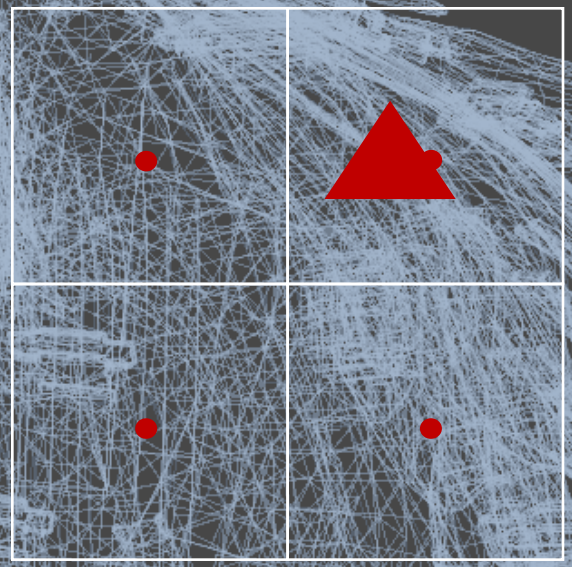
- Group of 2x2 pixels rasterized and shaded together
- Used to approximate derivatives for texture sampling ($\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial v}{\partial x}, \frac{\partial v}{\partial y}$)
- Using finite differences over adjacent pixels

Cinematic Quality Rendering

- Detailed geometry
- Realistic cameras
  - Depth of field and motion blur
- Realistic lighting
  - Area lights and soft shadows
  - Indirect lighting

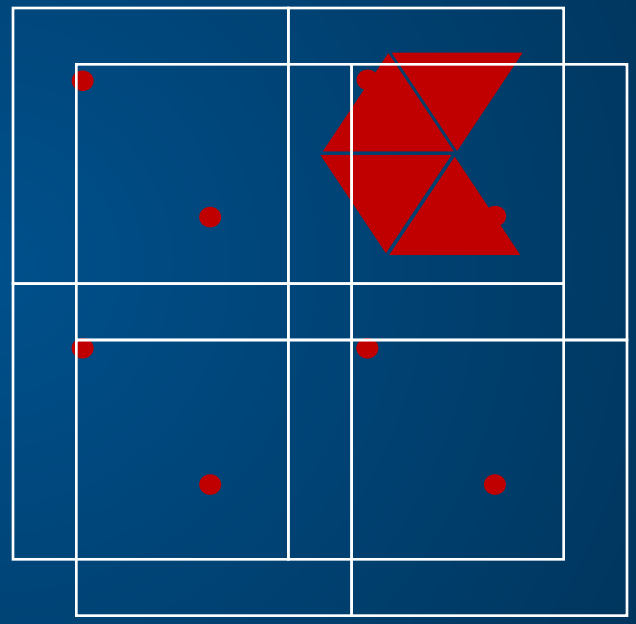

Detailed Geometry

Detailed Geometry



Detailed Geometry

- Want to shade once per pixel
- 4X shading with small triangles
- 8X shading with 2X MSAA



Shading for Detailed Geometry

- Reducing shading on GPUs using quad-fragment merging [Fatahalian ‘10]
- Object (Texture / Patch) space shading
  - The Reyes Image Rendering Architecture [Cook ‘87]
  - **AMFS: Adaptive Multi-Frequency Shading for Future Graphics Processors [Clarberg ‘14]**

AMFS: Adaptive multi-frequency shading

- Shade in patch space instead of screen space
- Reuse shading across triangles in a patch
- Shade at multiple rates in patch space

MSAA

Pixel shading: **6556** ops/px

Triangle area: 8.3 px

6556 ops x 3 Mpixels x 30 fps = **576 Gops/second**

AMFS

Pixel shading: **457** ops/px

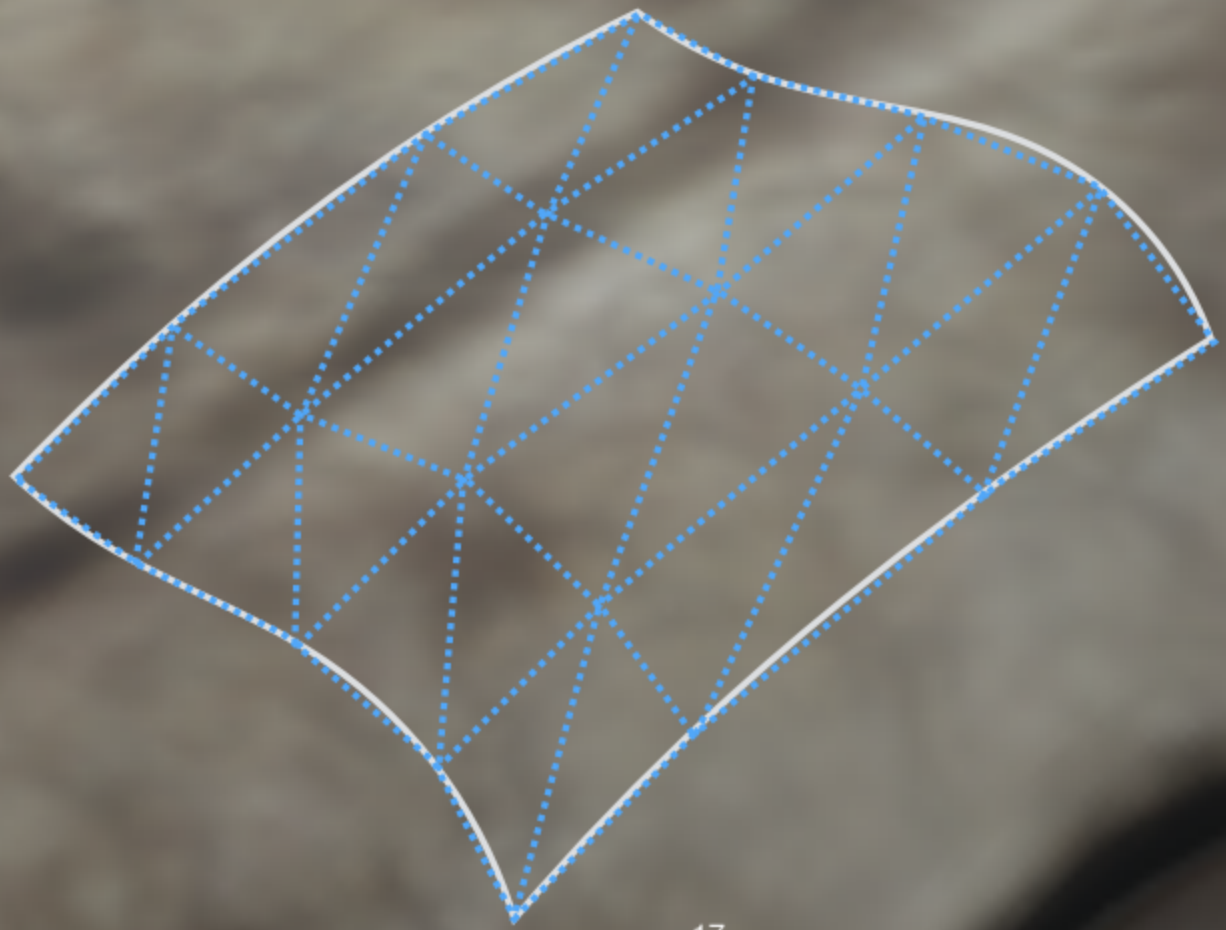
Triangle area: 8.3 px

14x reduction!

457 ops x 3 Mpixels x 30 fps = **40 Gops/second**

Patch-space Shading

- Use tessellation to render patches…



Patch-space Shading

- Use tessellation to render patches…

Patch-space Shading

- Lazily shade 2x2 grids in patch space

Patch-space Shading

- Lazily shade 2x2 grids in patch space

Patch-space Shading

- Lazily shade 2x2 grids in patch space
- Grids are cached & reused within patch

Patch-space Shading

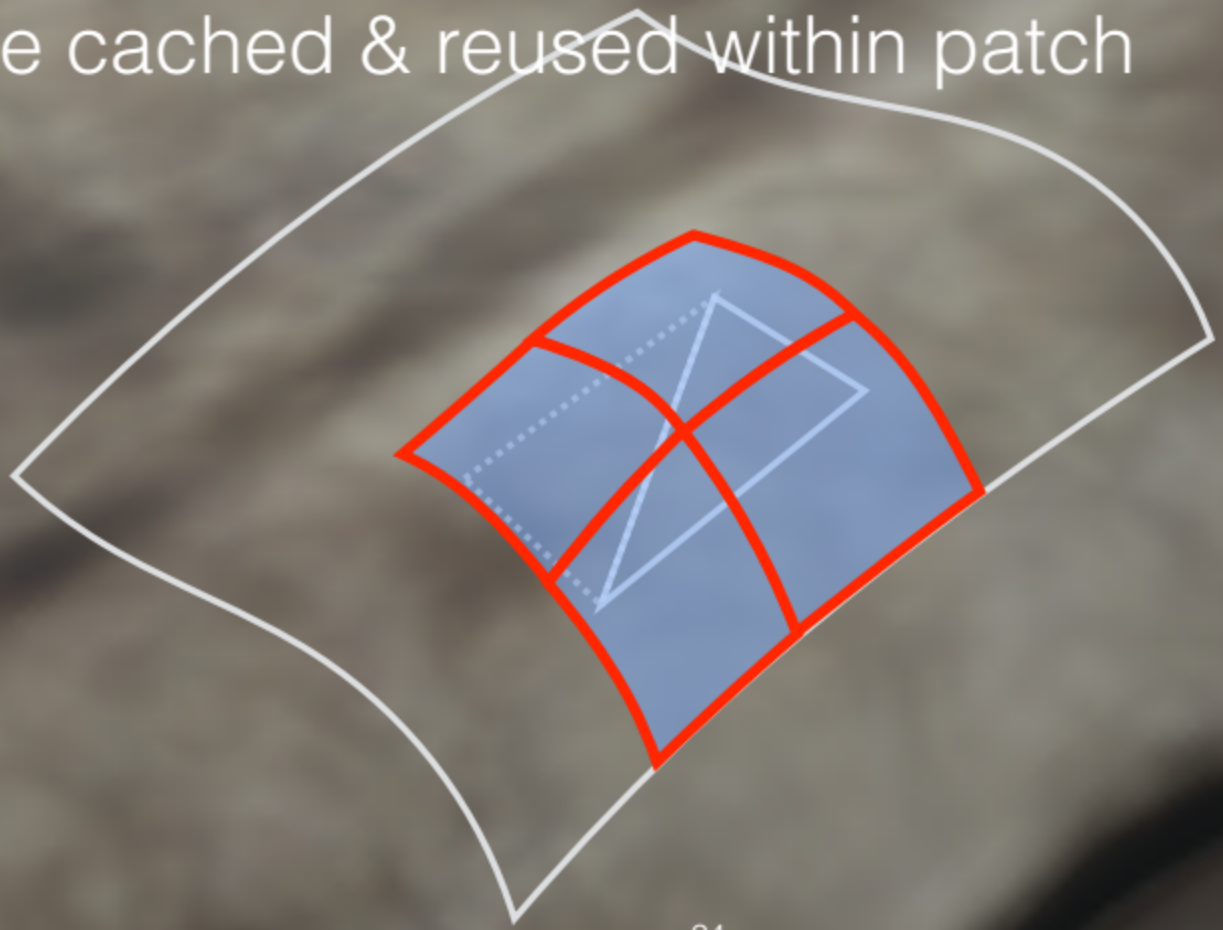
- Lazily shade 2x2 grids in patch space
- Grids are cached & reused within patch

Patch-space Shading

- Lazily shade 2x2 grids in patch space
- Grids are cached & reused within patch

Patch-space Shading

- Lazily shade 2x2 grids in patch space
- Grids are cached & reused within patch

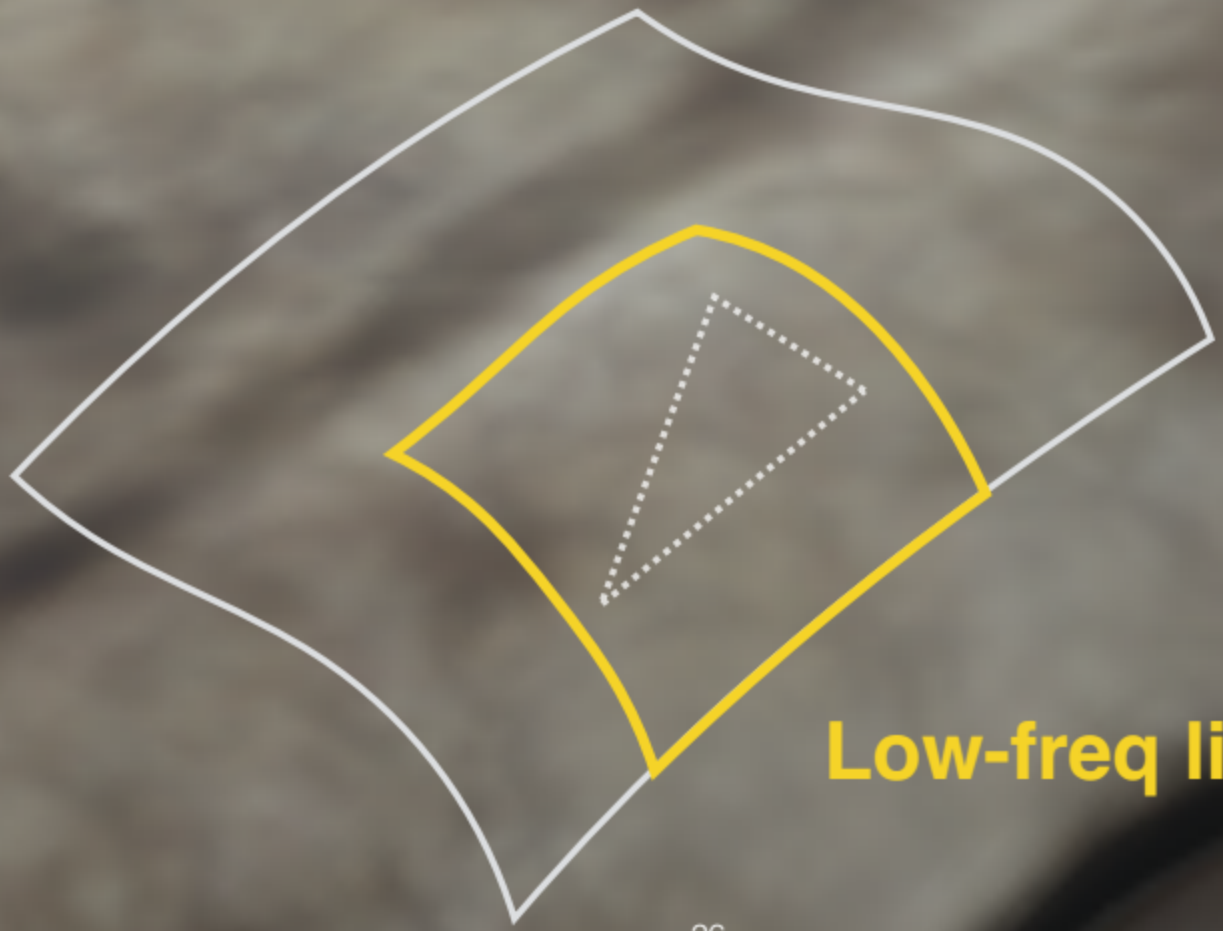


Adaptive Rate

- Grid size chosen based on **gradients** and/or **user control**

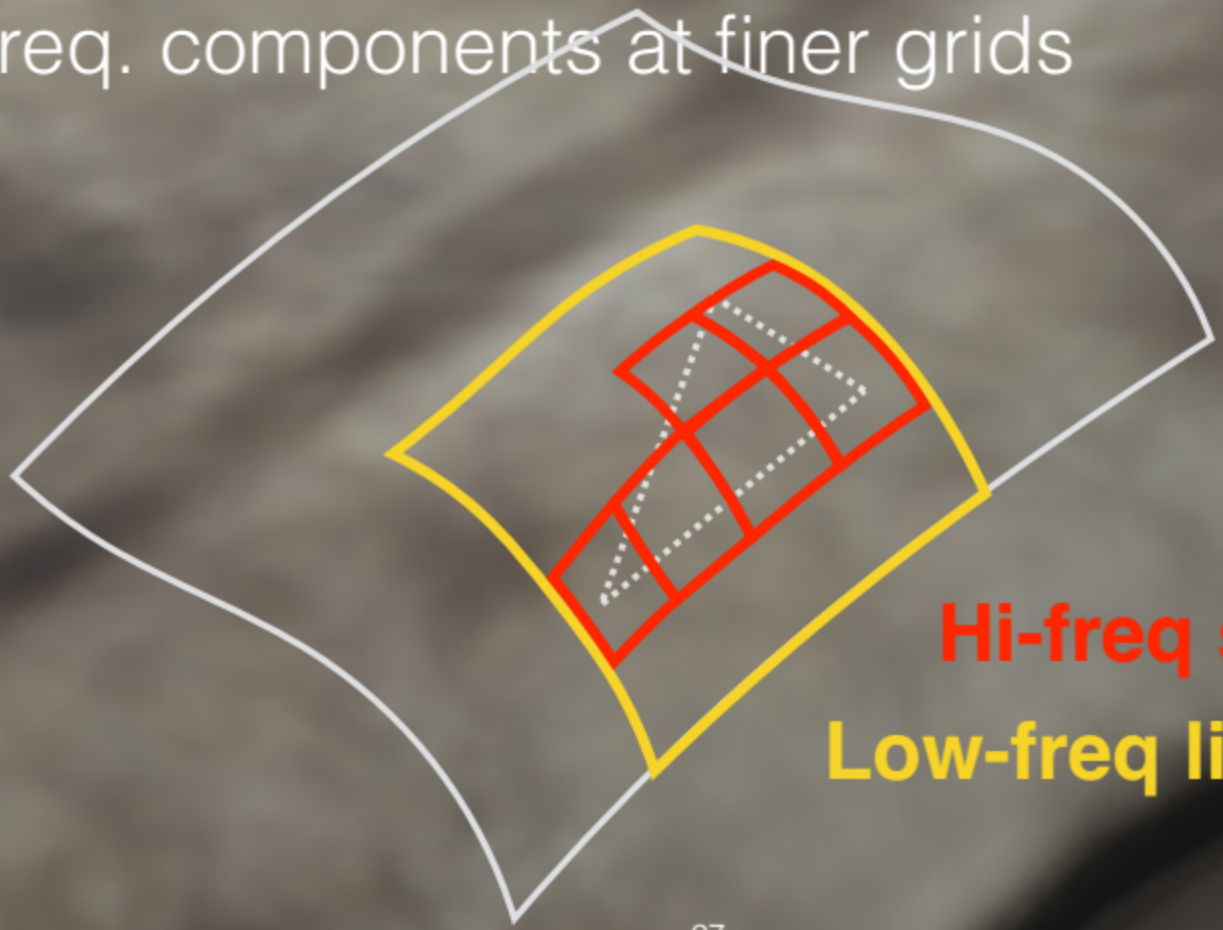
Multi-Frequency Shading

- Low-frequency components evaluated over larger grids



Multi-Frequency Shading

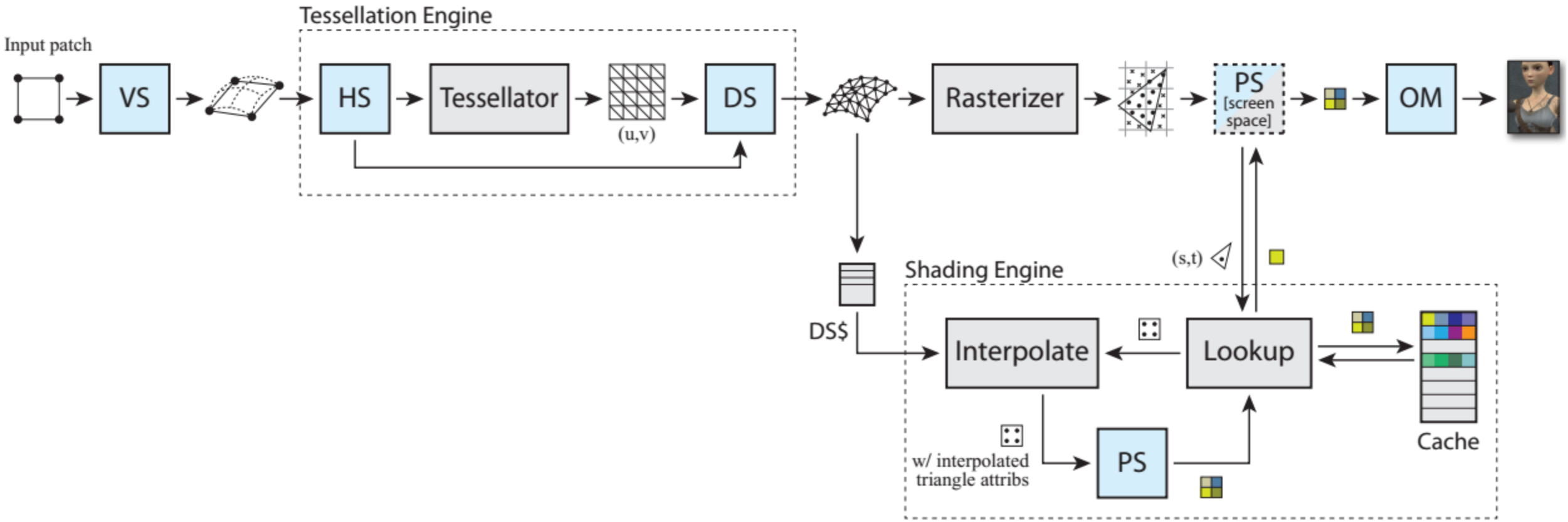
- Low-frequency components evaluated over larger grids
- Higher freq. components at finer grids



Hi-freq surface detail

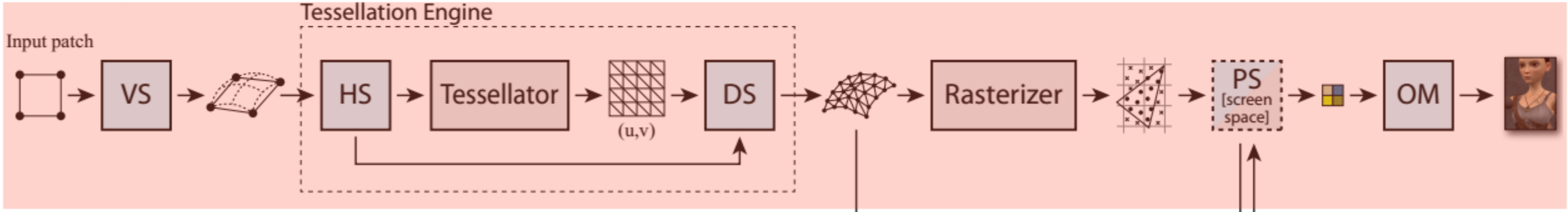
Low-freq lighting

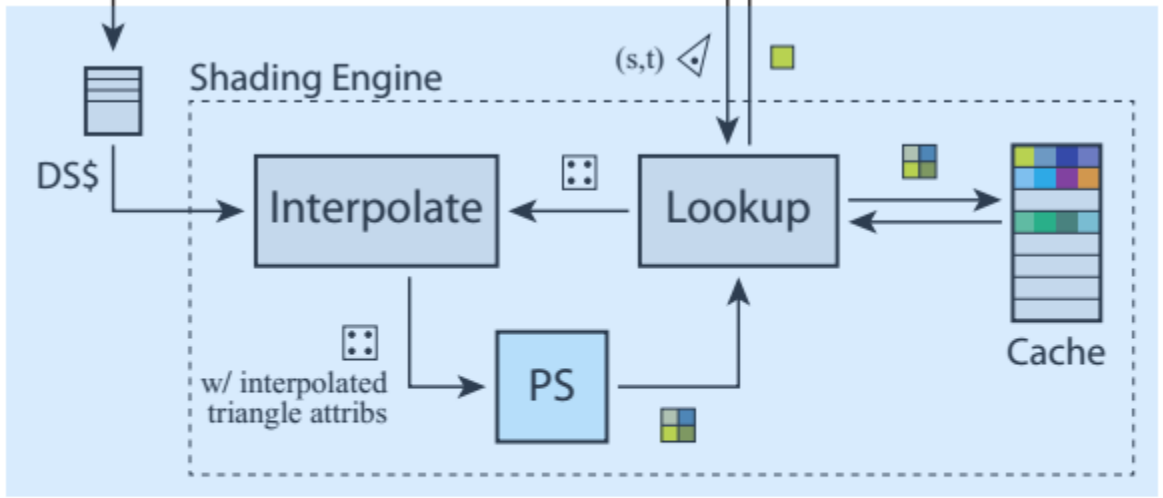
The AMFS Pipeline



The AMFS Pipeline

Today's GPUs (DX11-level)





New shading system

Image Quality Degradation

1/1 resolution
2.1x cost reduction

1/4 resolution
5.4x cost reduction

1/16 resolution
11.0x cost reduction

Cinematic Quality Rendering

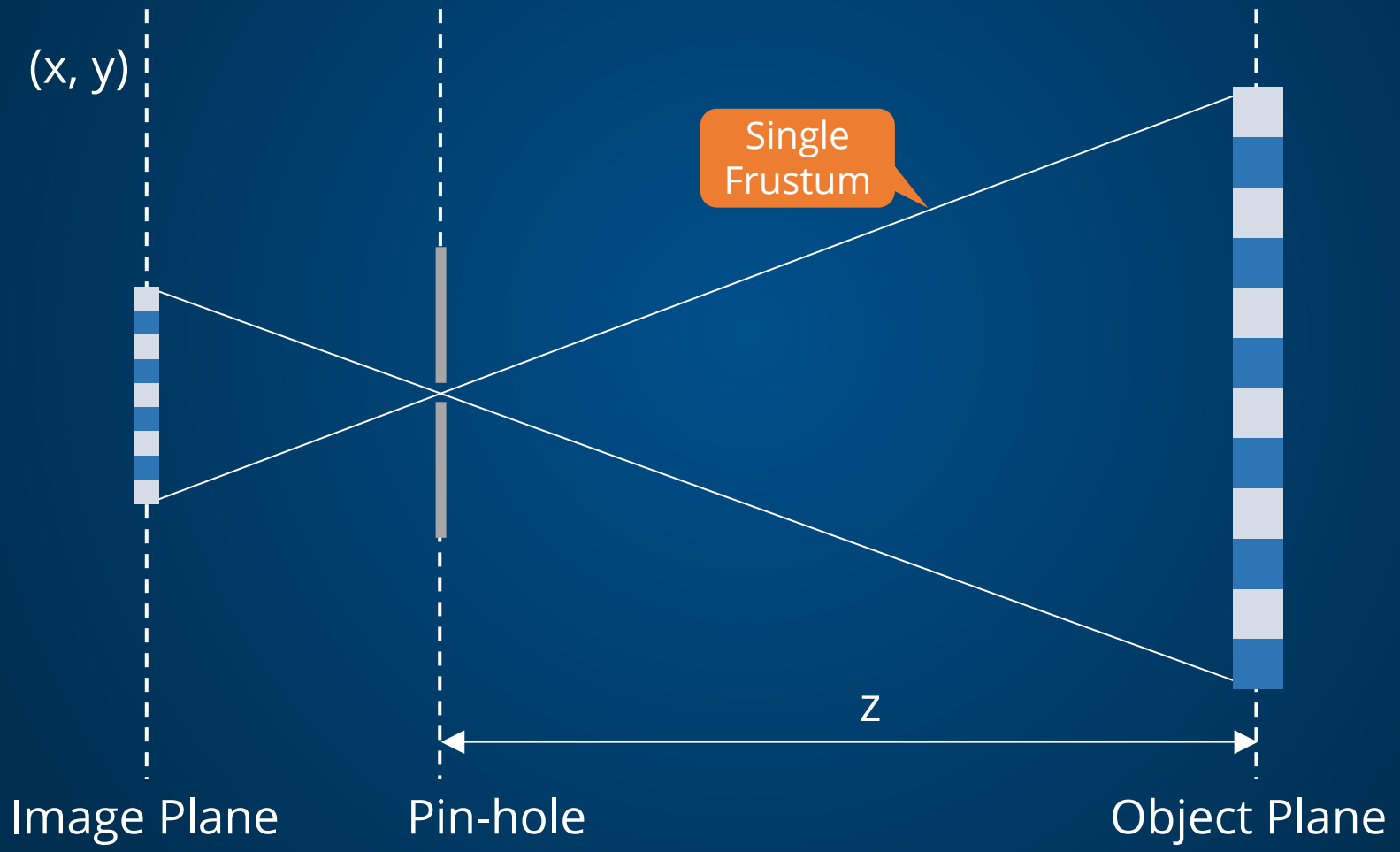
- Detailed geometry
- **Realistic cameras**
  - **Depth of field and motion blur**
- Realistic lighting
  - Area lights and soft shadows
  - Indirect lighting

Depth of Field and Motion Blur

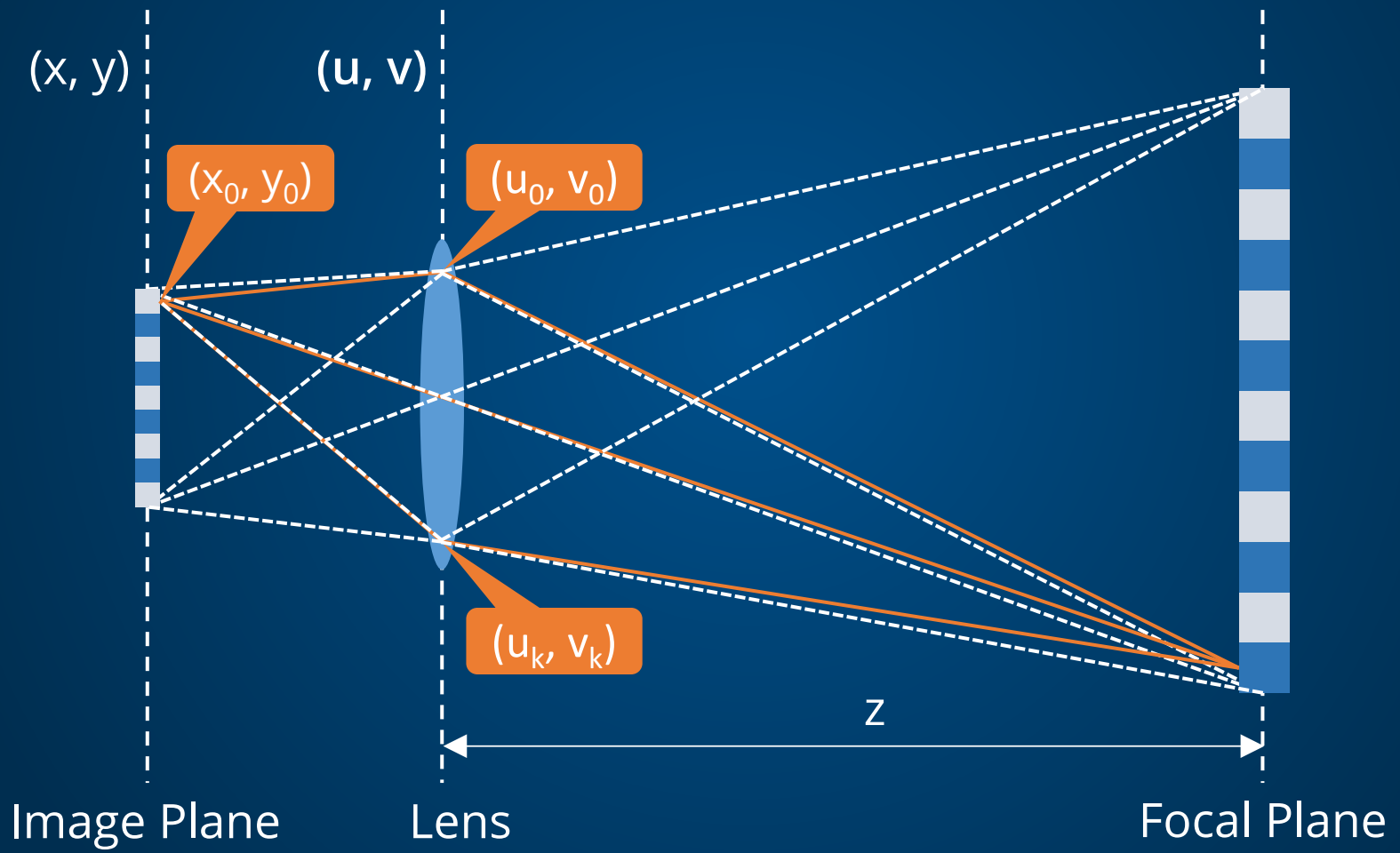
Stochastic Rasterization

- The Reyes Image Rendering Architecture
  [Cook ‘87]
- Stochastic Rasterization using Time-Continuous Triangles
  [Akenine-Möller ‘07]
- Data-Parallel Rasterization of Micropolygons with Defocus and Motion Blur
  [Fatahalian ‘09]

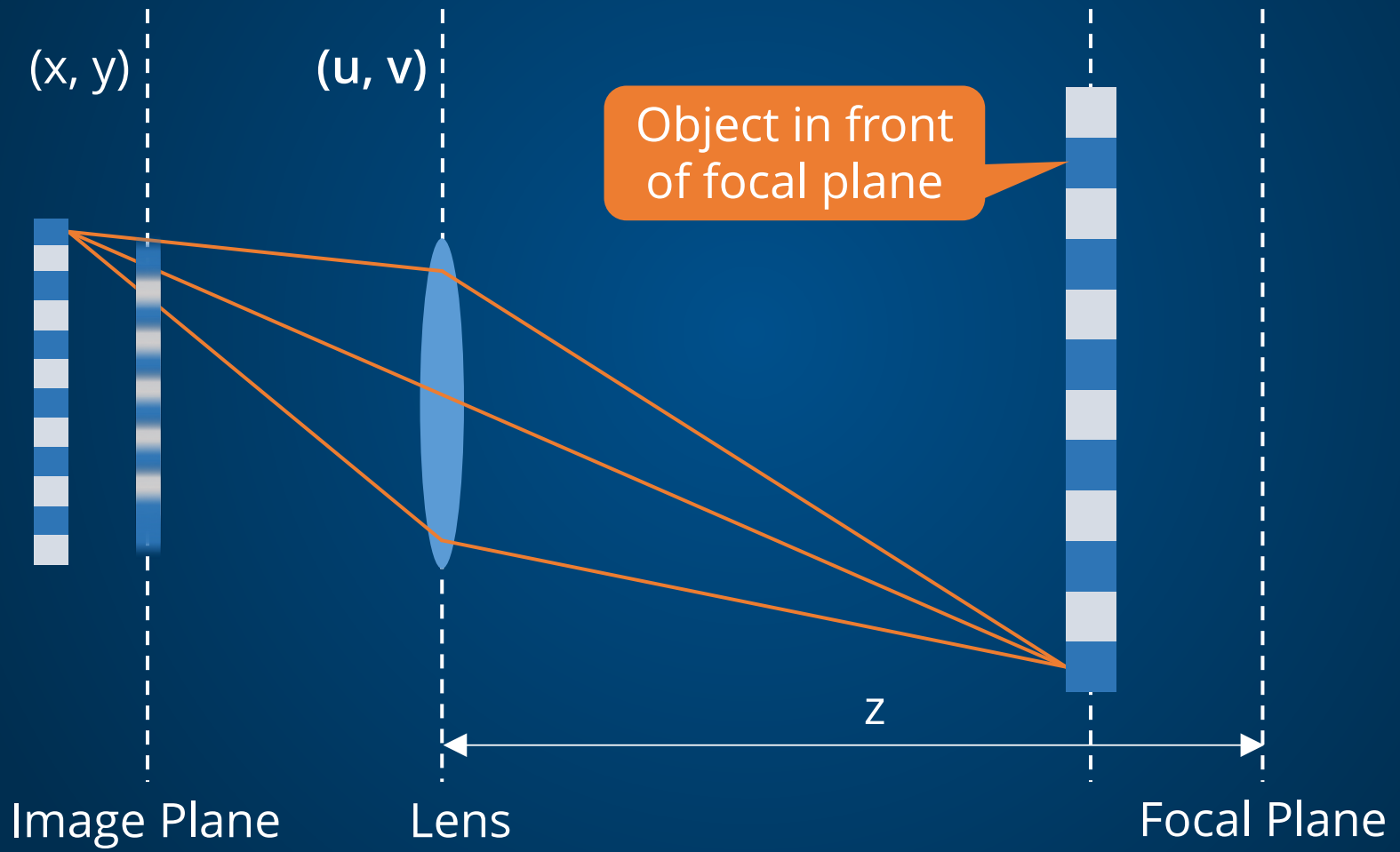
Rasterization with a Pinhole Camera



Stochastic Rasterization with a Lens



Stochastic Rasterization with a Lens



5D Stochastic Rasterization

- Stochastically sample five dimensions
  - Image plane (x, y) + Lens (u, v) + Time (t)
  - Needs a very large number of frustums
- Interleave N rasterizers, each with a unique (u, v, t) tuple
  - N ≥ 16 for reasonable image quality
  - Data-Parallel Rasterization of Micropolygons with Defocus and Motion Blur [Fatahalian ‘09]

Shading with Stochastic Rasterization

- Need ≥ 64 samples per pixel for decent quality
- Shading all these samples is expensive
- Shade once in object space and reuse across samples
  - Always shade at a fixed (u, v, t) (Lambertian source assumption)
  - AMFS: Adaptive multi-frequency shading for future graphics processors [Clarberg ‘14]

Reducing the Number of Samples

- 64 samples per pixel is still has significant rasterization cost and Z bandwidth
- Using fewer samples results in noisy images
- Use better reconstruction filters to suppress the noise
  - Layered Reconstruction for Defocus Blur [Vaidyanathan ‘14]
  - Layered Reconstruction for Defocus and Motion Blur [Munkberg ‘14]

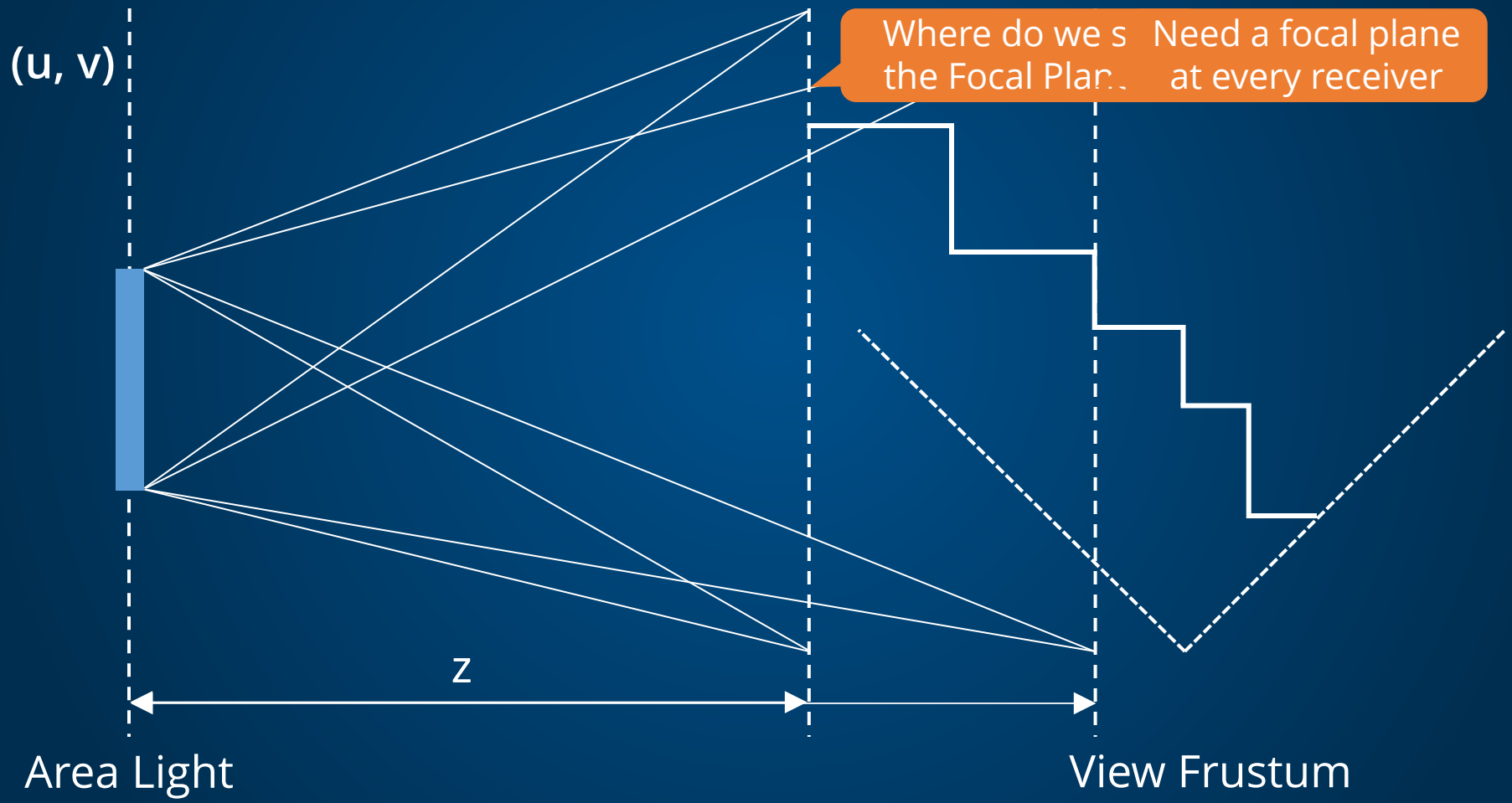

8 samples / pixel


Reconstructed

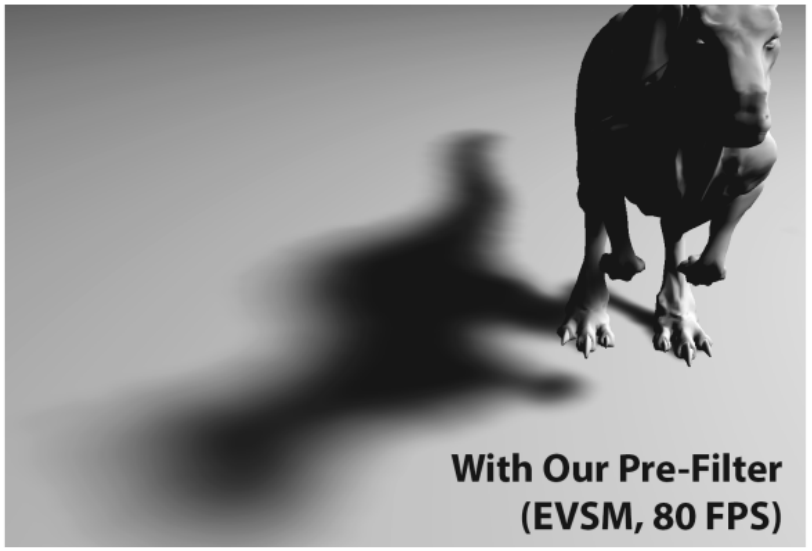
Cinematic Quality Rendering

- Detailed geometry
- Realistic cameras
  - Depth of field and motion blur
- **Realistic Lighting**
  - **Area lights and soft shadows**
  - Indirect lighting

The Area Lights Problem



Area Lights with Stochastic Rasterization



- Stochastic Soft Shadow Mapping [Liktor ‘15]
  - Reproject samples to multiple planes
  - Pre-filter by computing distributions (Exponential Variance)
  - Sample from distribution

Issues with Stochastic Rasterization

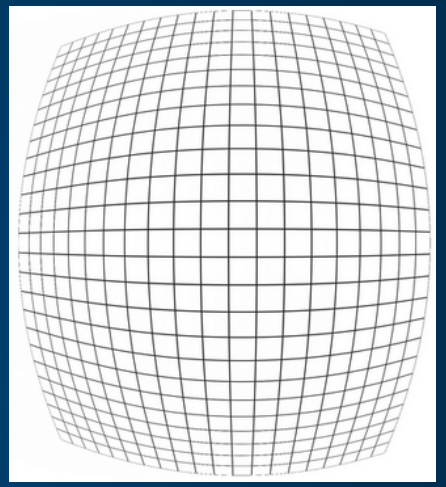
- Is accurate depth of field and motion blur really important ?
  - Not as much for games
- Can we do other distribution effects like soft shadows ?
  - Not very well
- 16 rasterizer setups is expensive
- **Poor Z locality and significantly higher bandwidth**

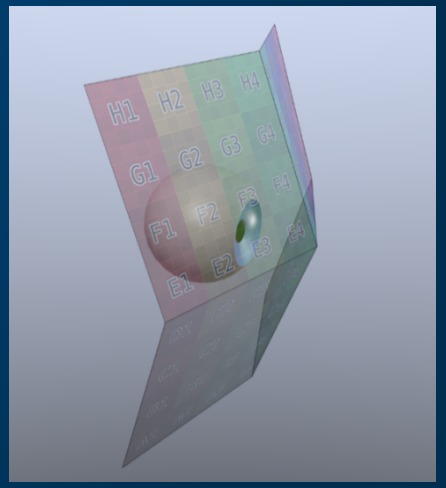
VR Rendering (2015)

- Several ideas from stochastic rasterization research can be applied to VR rendering
- Render multiple views simultaneously in the pipeline
  - Left / Right eye and lens matching (more on that soon)
- Reuse shading across multiple views
  - AMFS is a perfect fit!

Multi-View Rendering for VR

- VR HMDs use wide angle lenses that distort the image
- Lens distortion compensated with barrel pre-distortion
- Pre-distortion results in non-uniform sample density
- Better sampling with multiple projections
  - Comparison of Projection Methods for Rendering Virtual Reality [Toth ’16]





The Transition to Ray Tracing (2014)

- REYES was the standard for offline rendering and was based on rasterization until …

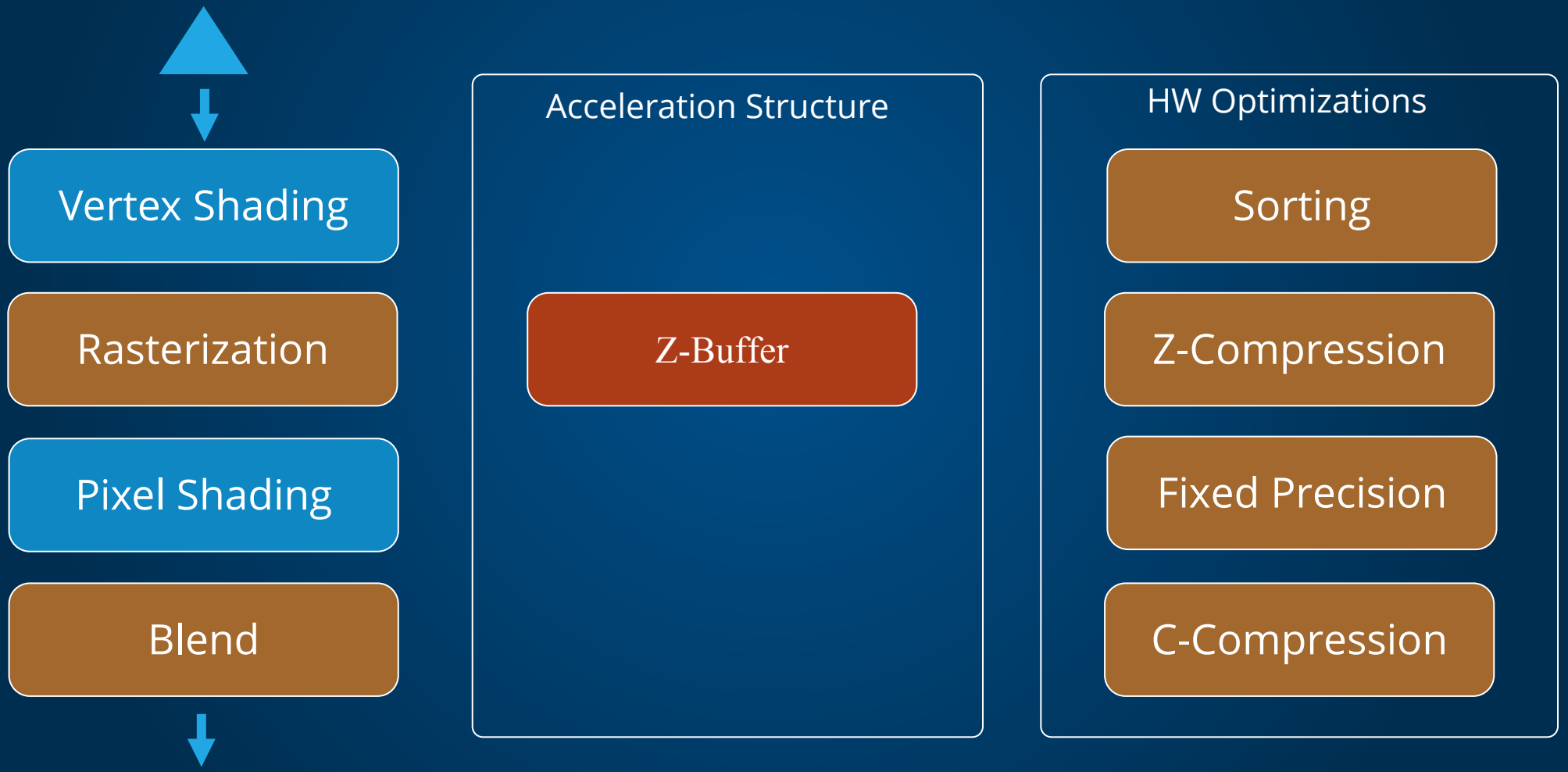
- The transition to raytracing at Sony Pictures Imageworks [‘14]
  - Brute force ray tracing greatly simplified content creation
  - Eliminated artist cycles for positioning shadow and reflection maps

- By 2014 ray tracing dominated the movie and VFX industry

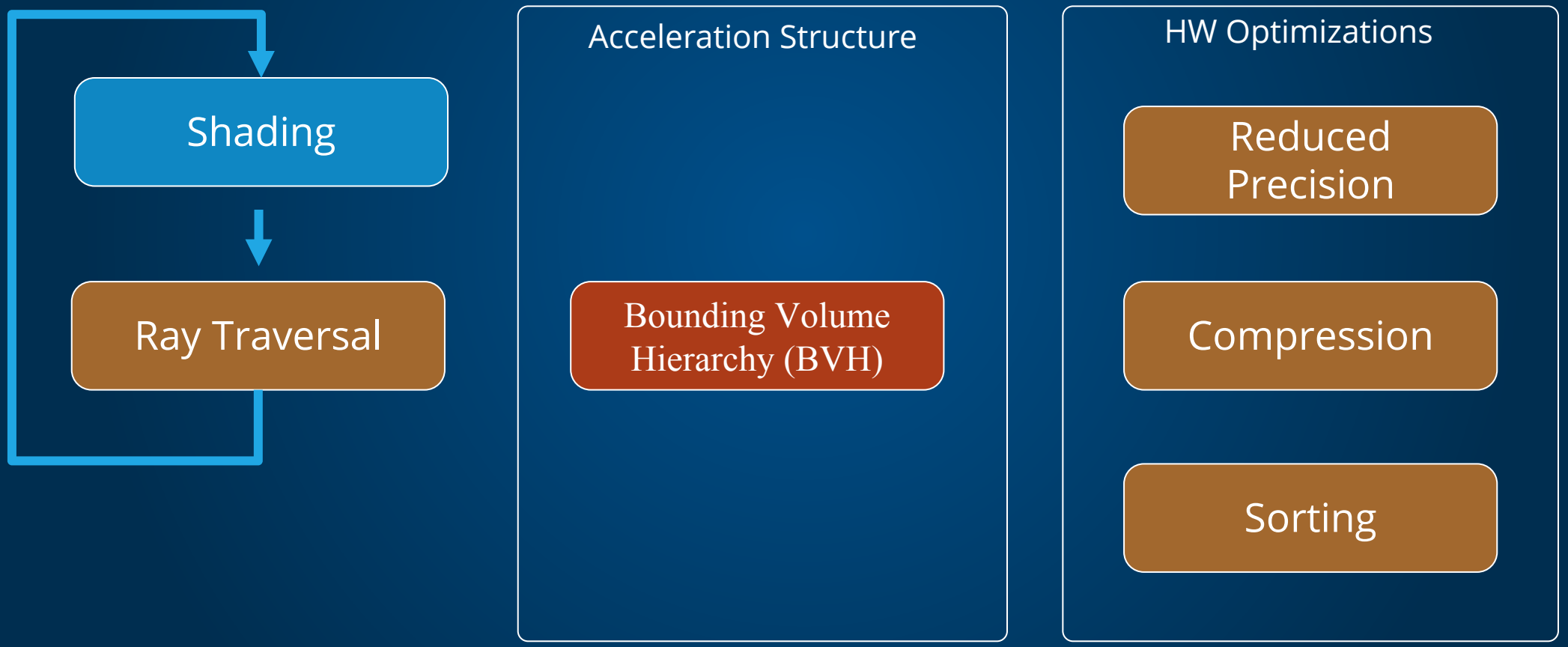
Real-Time Ray Tracing

- Approximate methods based on rasterization are less efficient and produce visible artifacts for e.g. shadow maps, cube maps
- Simplified ray tracing is already used in games today, e.g. ambient occlusion, screen space reflections
- Ray tracing is ideally suited for VR rendering
- However SIMD processors are a poor fit for ray tracing
  - Dedicated hardware traversal can be significantly more efficient
  - T&I engine: traversal and intersection engine for hardware accelerated ray tracing [Nah ‘11]

Hardware Rasterization

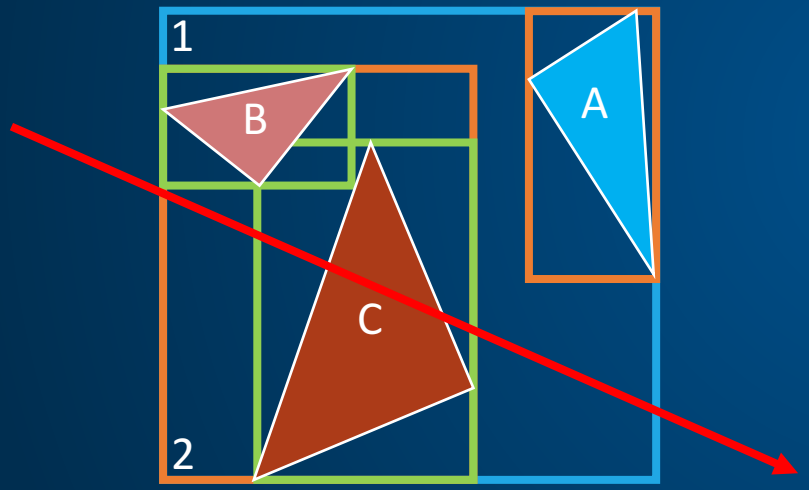


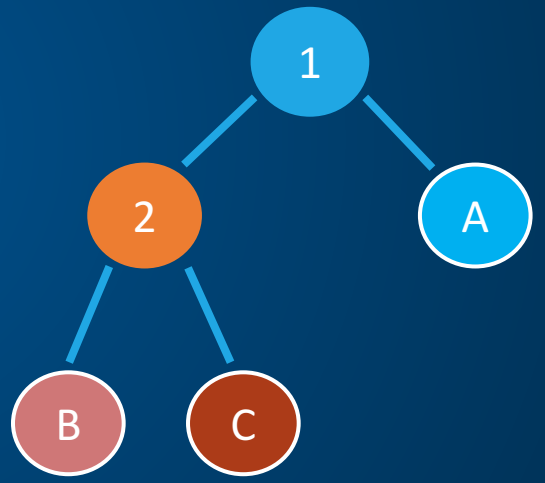
State of the Art in Hardware Ray Tracing



Ray Traversal

- Process of finding a ray scene intersection
- State of the art: Build a Bounding Volume Hierarchy (BVH)

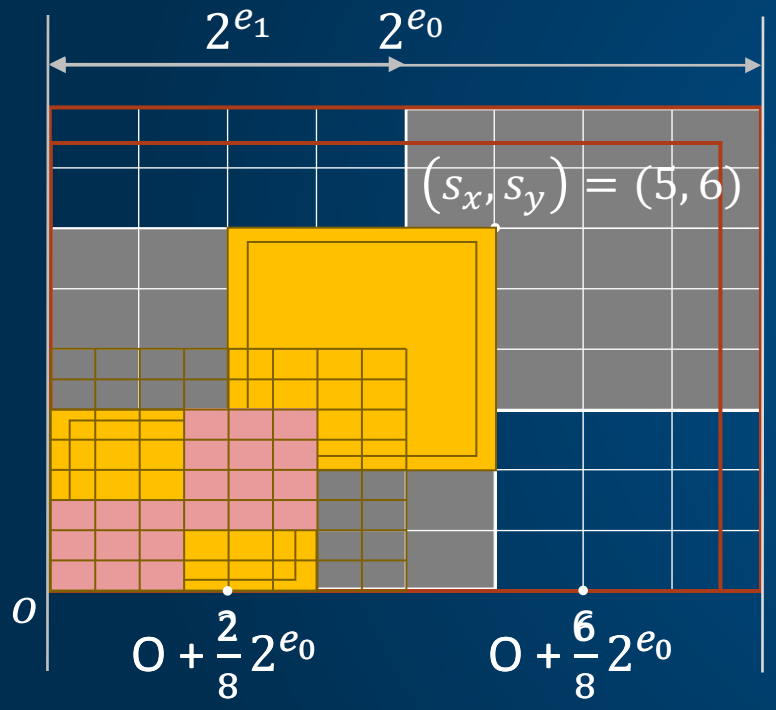




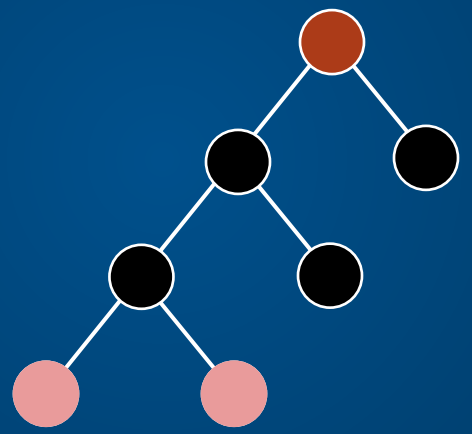
- Intersect ray with BVH to find triangles
  - Intersect ray with triangles to find hit point

BVH Compression with Incremental Encoding

[Mahovsky 06, Keely 14]



Bounds

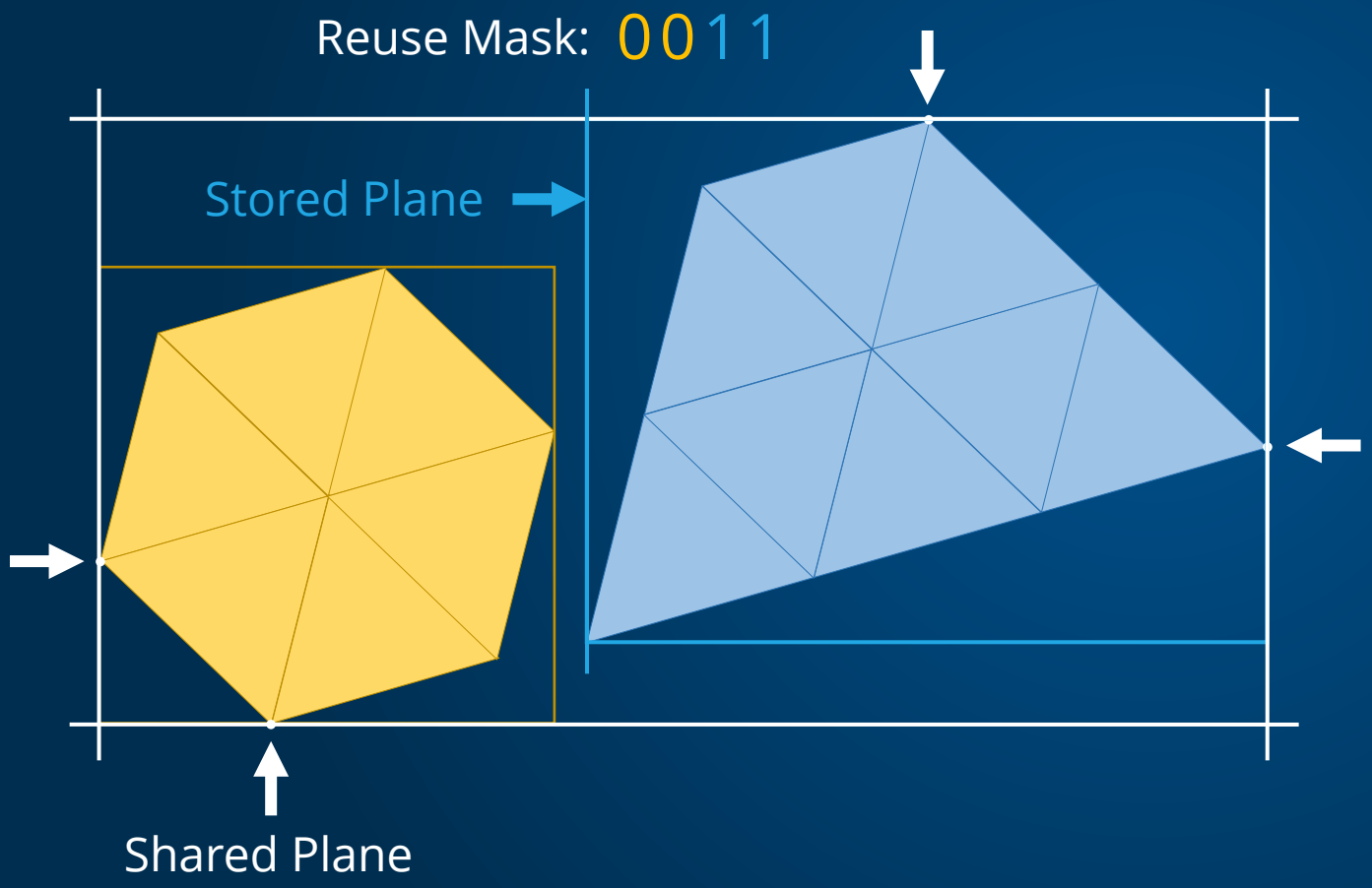


BVH Topology

- Derive low-res grid
- Quantize Box
- Compute 3-bit offsets w.r.t parent origin
- Derive new grid
- ….

Compact BVH with Shared Parent Bounds

[Fabianowsky 09]



- Each parent plane shared with one child
- 1 bit to indicate the child
- Only store new child planes
- Reduces planes by half

BVH Compression

[Incremental Encoding + Parent Sharing]

Field	Bits
Reuse Mask	6
6-bit Planes X 6	36
Child Index	21
Leaf Indicator	1
Total	64 (8 bytes for 2 nodes)

Reduced Precision Ray-BVH Traversal

- Desirable to have consistent watertight intersections (no false misses)
  - Especially for reduced precision traversal
- Well known approaches for full precision traversal
  - Robust BVH Ray Traversal [Ize ‘13]
  - Watertight Ray/Triangle Intersection [Woop ‘13]
- Watertight Ray Traversal with Reduced Precision [Vaidyanathan ‘16]
  - Leverages incremental encoding for reduced precision traversal
  - Significant savings in hardware complexity

Conclusion

- Research continues to drive the evolution of the GPU architecture to enable new applications and cinematic quality rendering
- Exciting times ahead and exciting opportunities for new research and architecture innovations

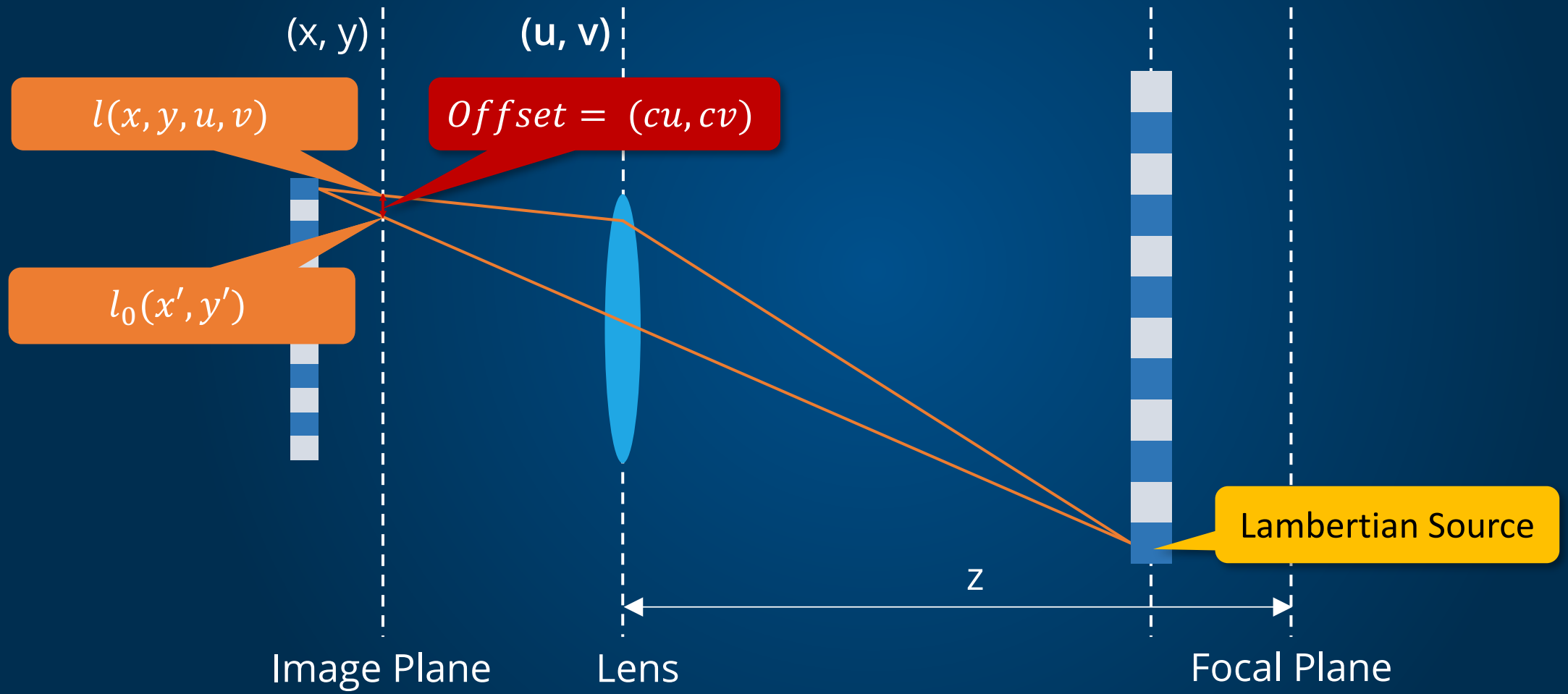
Acknowledgements

- The Advanced Rendering Technologies team
- Petrik Clarberg, Jacob Munkberg, Marco Salvi
- David Kanter

BACKUP

Fourier Analysis based Reconstruction Filters

Stochastic Rasterization with a Lens

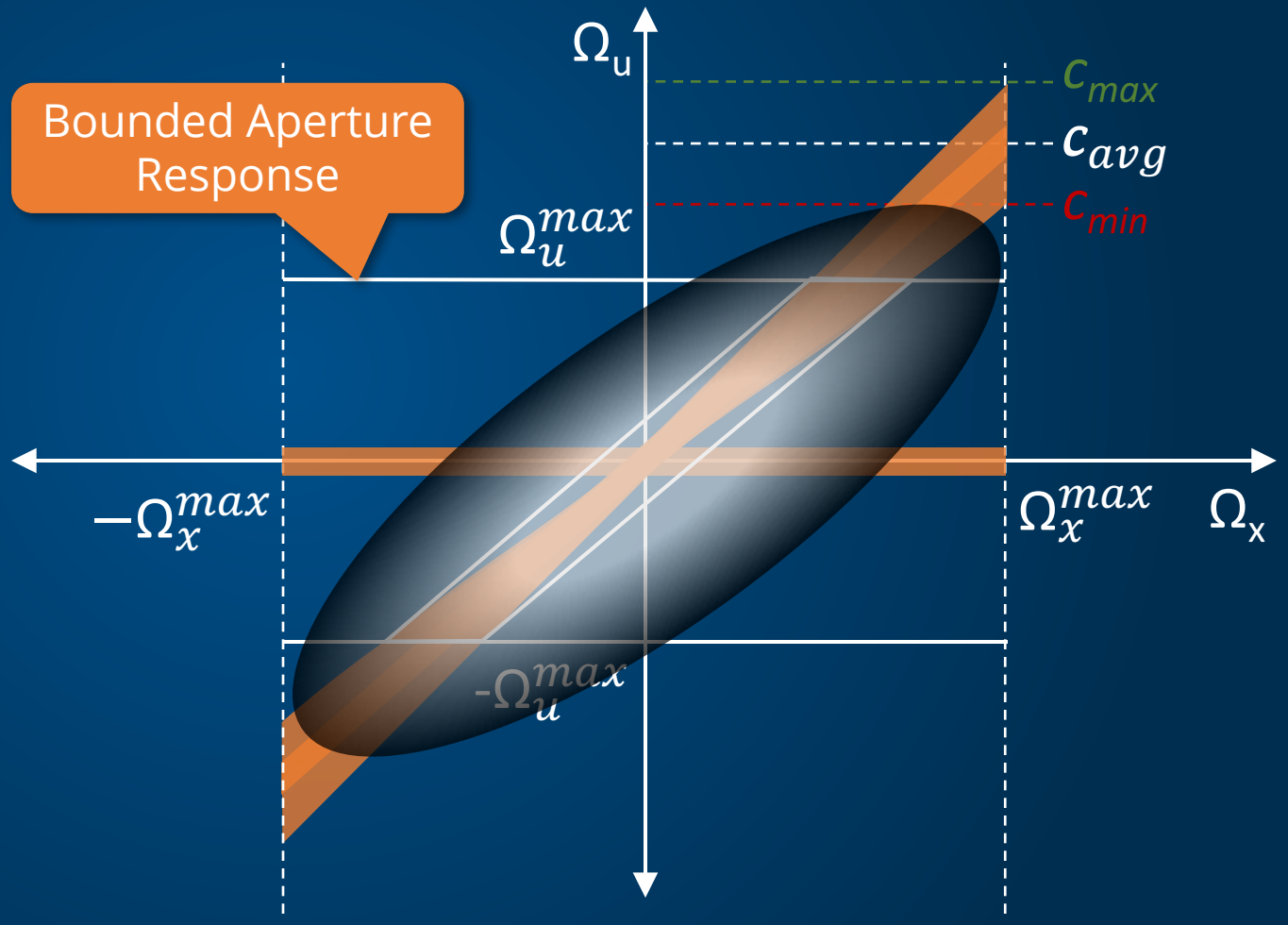


Radiance Frequency Response

- Sheared Radiance
  - $l(x, y, u, v) = l_0(x + cu, y + cv)$
- Frequency Domain
  - $L(\Omega_x, \Omega_y, \Omega_u, \Omega_v) = L_0(\Omega_u - c\Omega_x, \Omega_v - c\Omega_y)$

Radiance Frequency Bounds
[Egan et al. 2011]

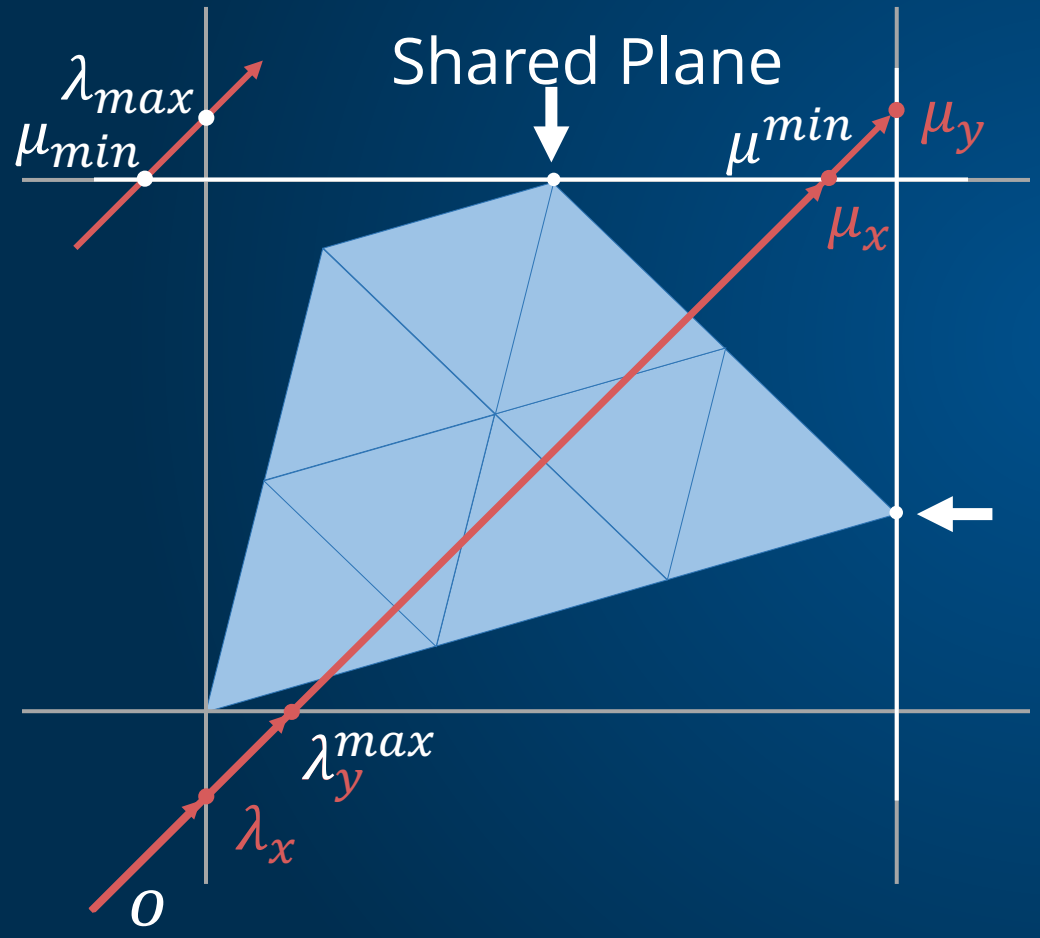
- Response bounded in (Ω_x, Ω_y)
- Shear depends on Circle of Confusion
- Shear range with Min and Max shear
- Clipped by Aperture frequency limits



Watertight Reduced Precision Traversal

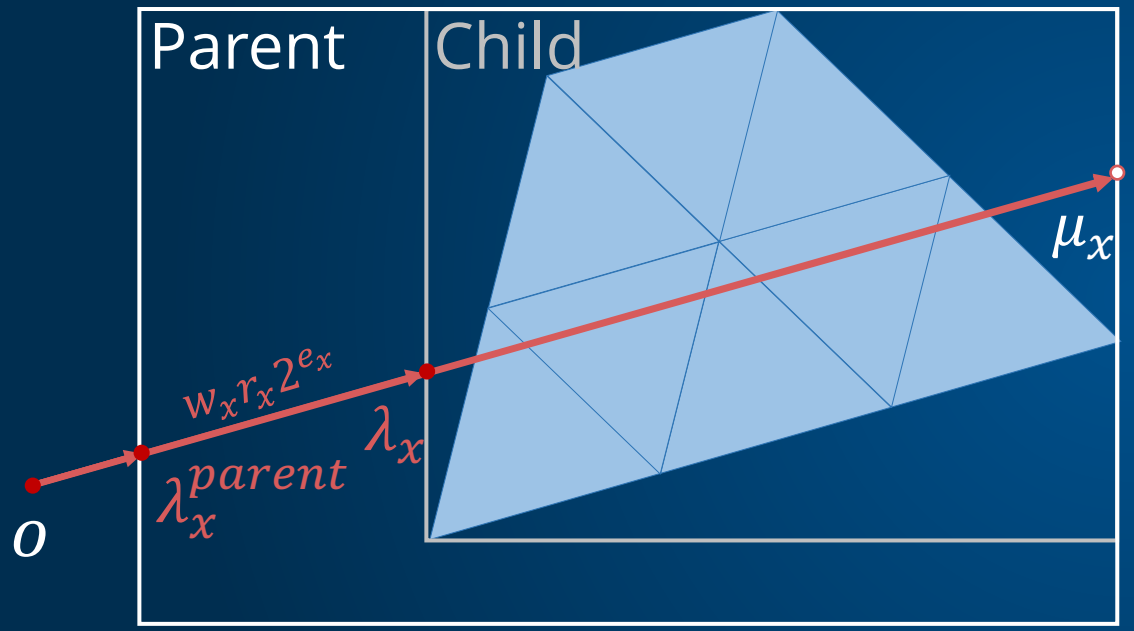
Ray-Box Intersections

[Kay, Kajia 86]



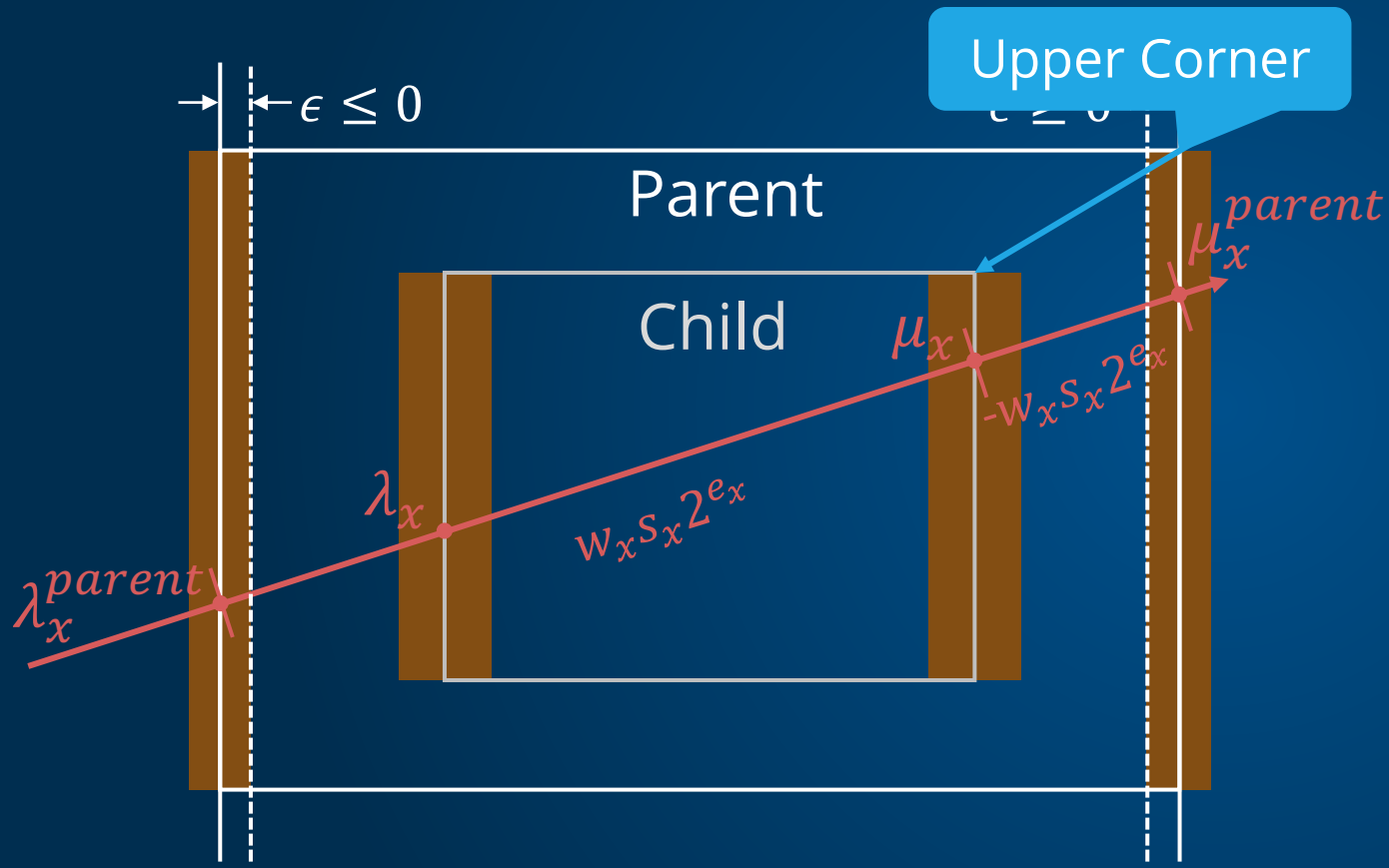
- Compute distance to each plane
- Find max distance to lower planes
- Find min distance to upper planes
- Test $\lambda^{max} \leq \mu^{min}$
- Reuse distances for shared planes
- Saves half the computations

Reduced Precision & Parent Sharing



- Leverage incremental encoding
- Incremental computation
  - $\lambda_x = \lambda_x^{parent} + w_x r_x 2^{e_x}$
  - w_x is the ray slope
  - r_x is a quantized offset
  - Reduced precision MUL
- Can reuse parent distance
  - $\mu_x = \mu_x^{Parent}$

Watertight Intersections



- Numerical errors in λ, μ
- Errors accumulate
- For watertight intersections
  - No +ve error in λ
  - No -ve error in μ
- Errors in λ, μ are coupled
- Encode s w.r.t upper corner
- Independently round λ, μ