BEST OF GTC

PRACTICAL REAL-TIME VOXEL-BASED GLOBAL ILLUMINATION FOR CURRENT GPUS

Alexey Panteleev
NVIDIA

BEST OF GTC

OUTLINE

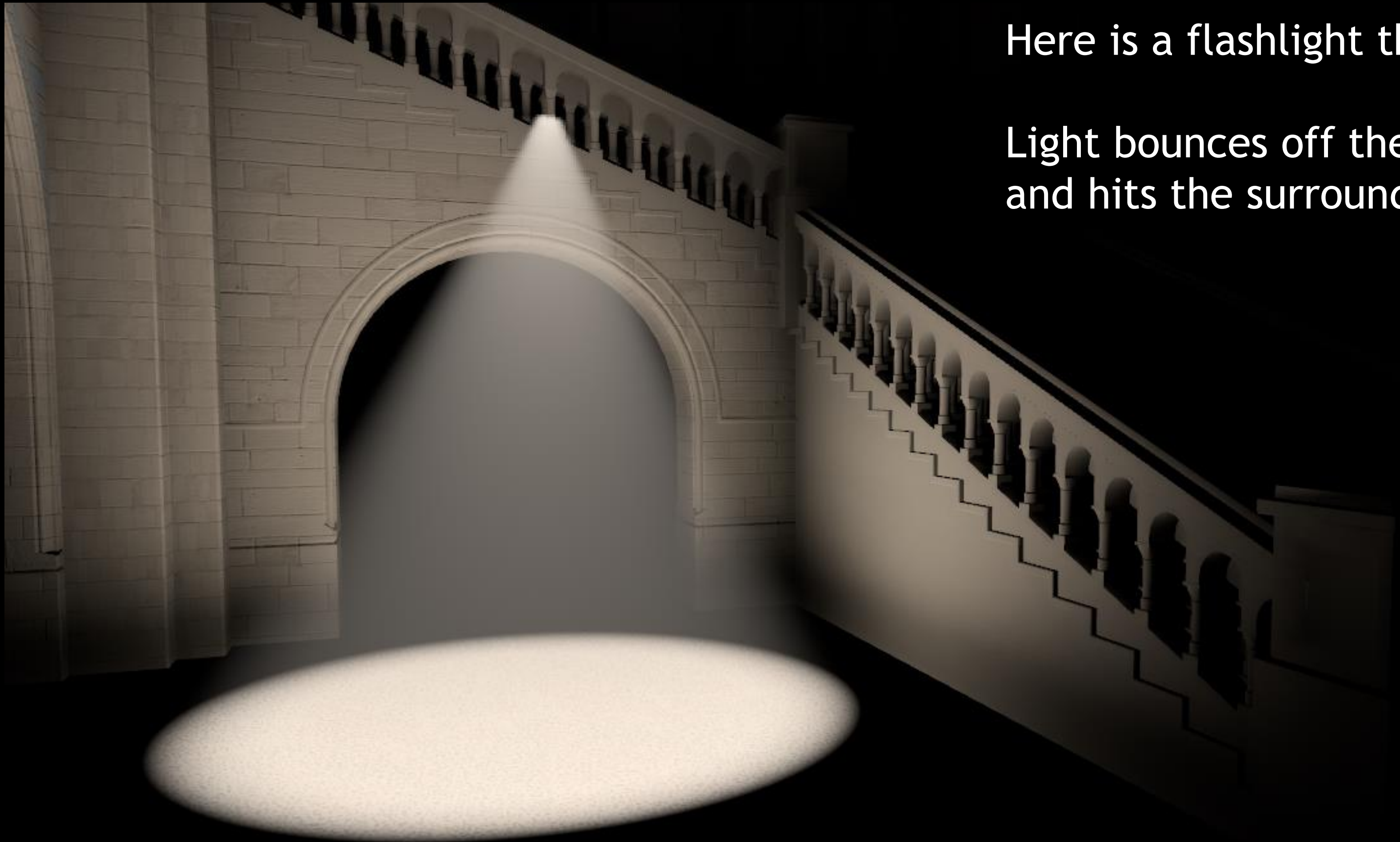
- Introduction: what is Global Illumination?
- Screenshots
- Overview of Voxel Cone Tracing
- Implementation details
  - Voxel clipmaps and incremental updates
  - Voxelization algorithms
  - Light injection algorithms
  - Cone tracing
- Performance

WHAT IS GLOBAL ILLUMINATION?

Here is a flashlight that lights the floor.

BEST OF GTC

WHAT IS GLOBAL ILLUMINATION?



Here is a flashlight that lights the floor.

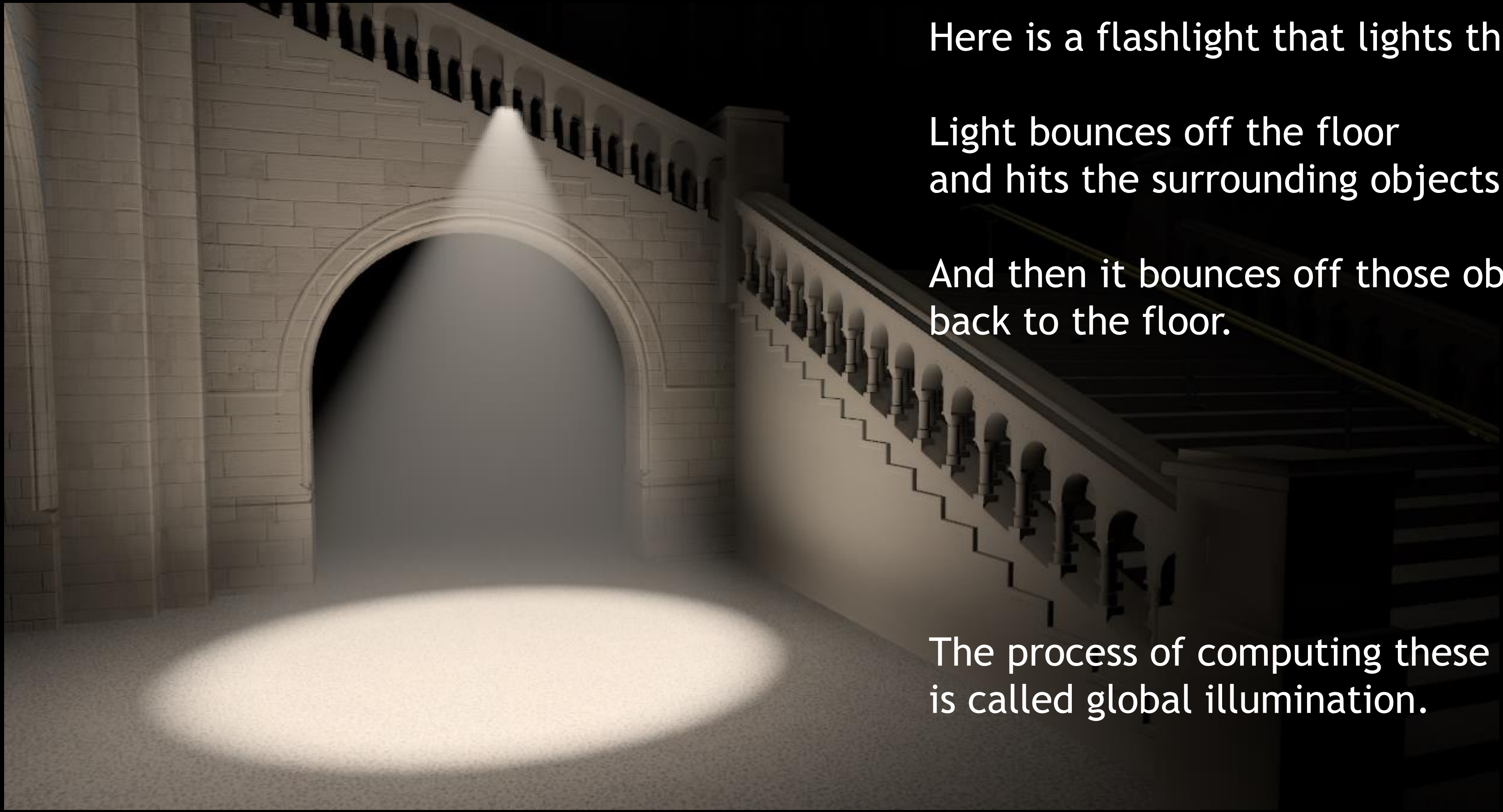
Light bounces off the floor
and hits the surrounding objects.

BEST OF GTC

NVIDIA

BEST OF GTC

WHAT IS GLOBAL ILLUMINATION?



Here is a flashlight that lights the floor.

Light bounces off the floor
and hits the surrounding objects.

And then it bounces off those objects
back to the floor.

The process of computing these bounces
is called global illumination.

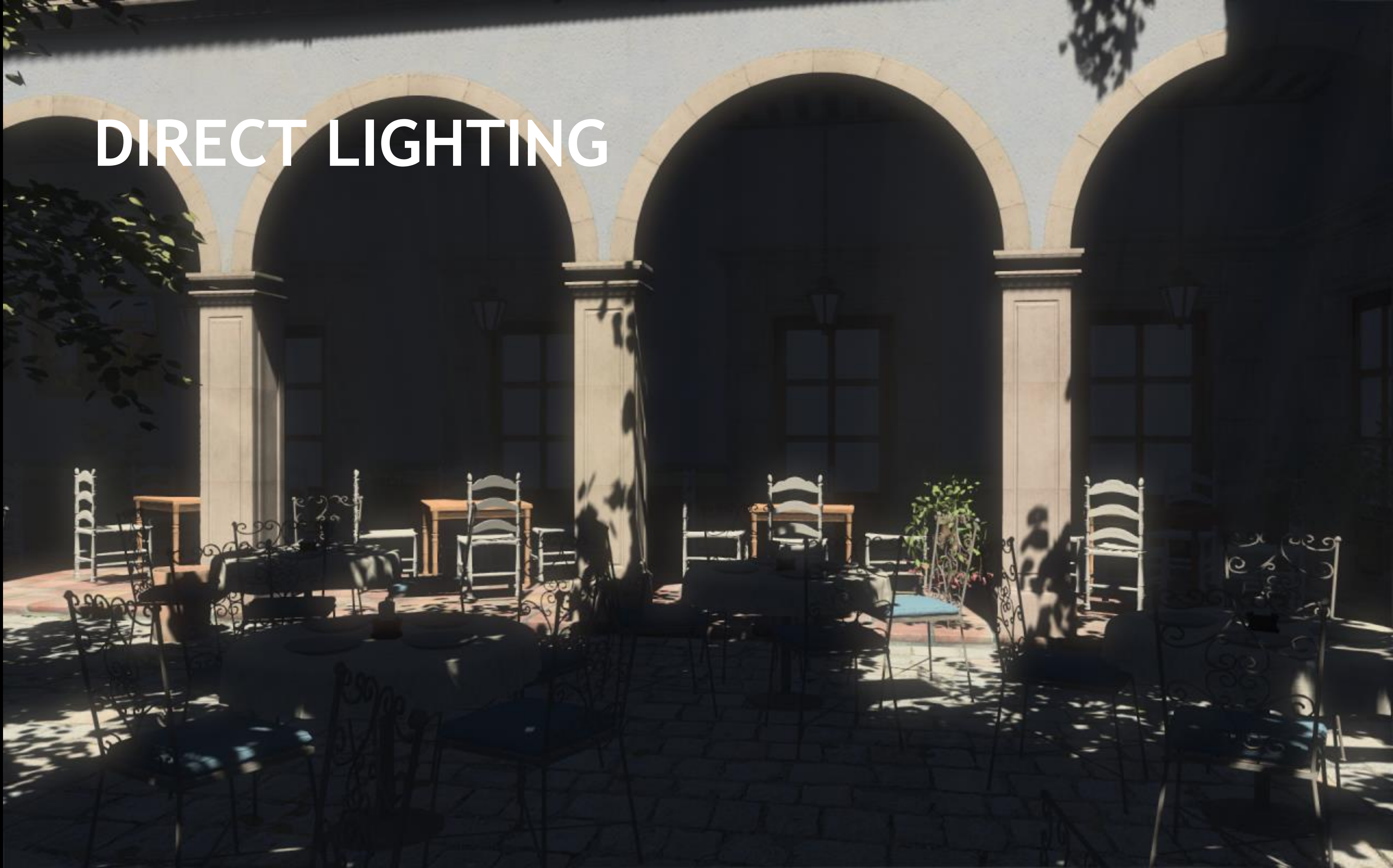
HOW IT IS USUALLY SOLVED

- Accurate physics-based GI computation is extremely expensive
- Static approximations
  - Flat ambient
  - Light maps
- Dynamic approximations
  - Manually placed lights simulating indirect illumination
  - Virtual Point Lights – expensive, no occlusion
  - SH irradiance volumes, Light propagation volumes – no specular
  - Image-space approaches – incomplete scene information
  - Sparse Voxel Octree Global Illumination (SVOGI) – doesn’t handle dynamic or large scenes well

OUR SOLUTION

- Dynamic approximation
  - No offline pre-computations
  - Handles dynamic scenes easily
- Voxel Cone Tracing
  - "Interactive Indirect Illumination Using Voxel Cone Tracing" by Cyril Crassin, Fabrice Neyret, Miguel Sainz, Simon Green, Elmar Eisemann
- Gathering information from a multi-resolution voxel representation of the scene

BEST OF GTC

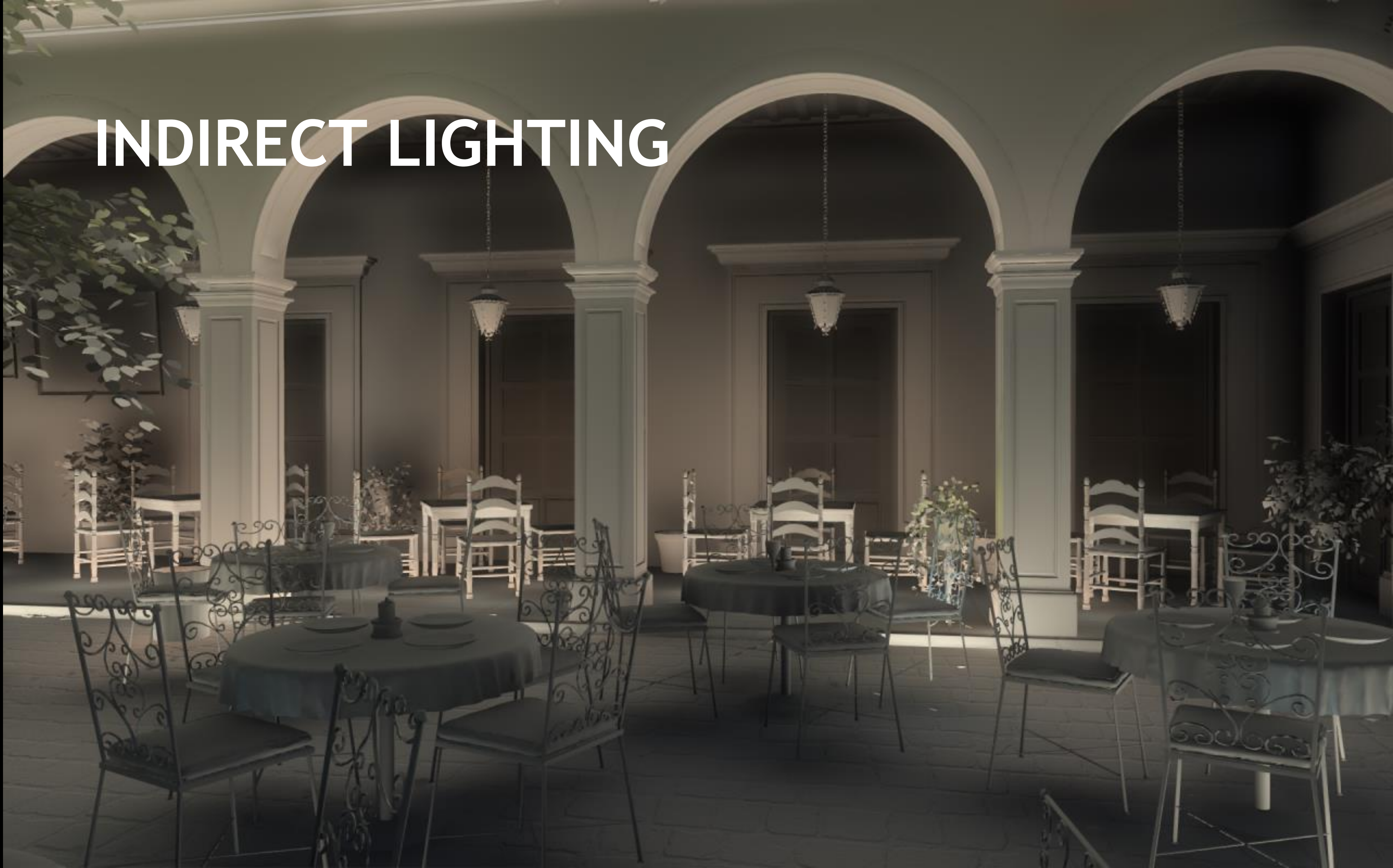


DIRECT LIGHTING

BEST OF GTC

NVIDIA.

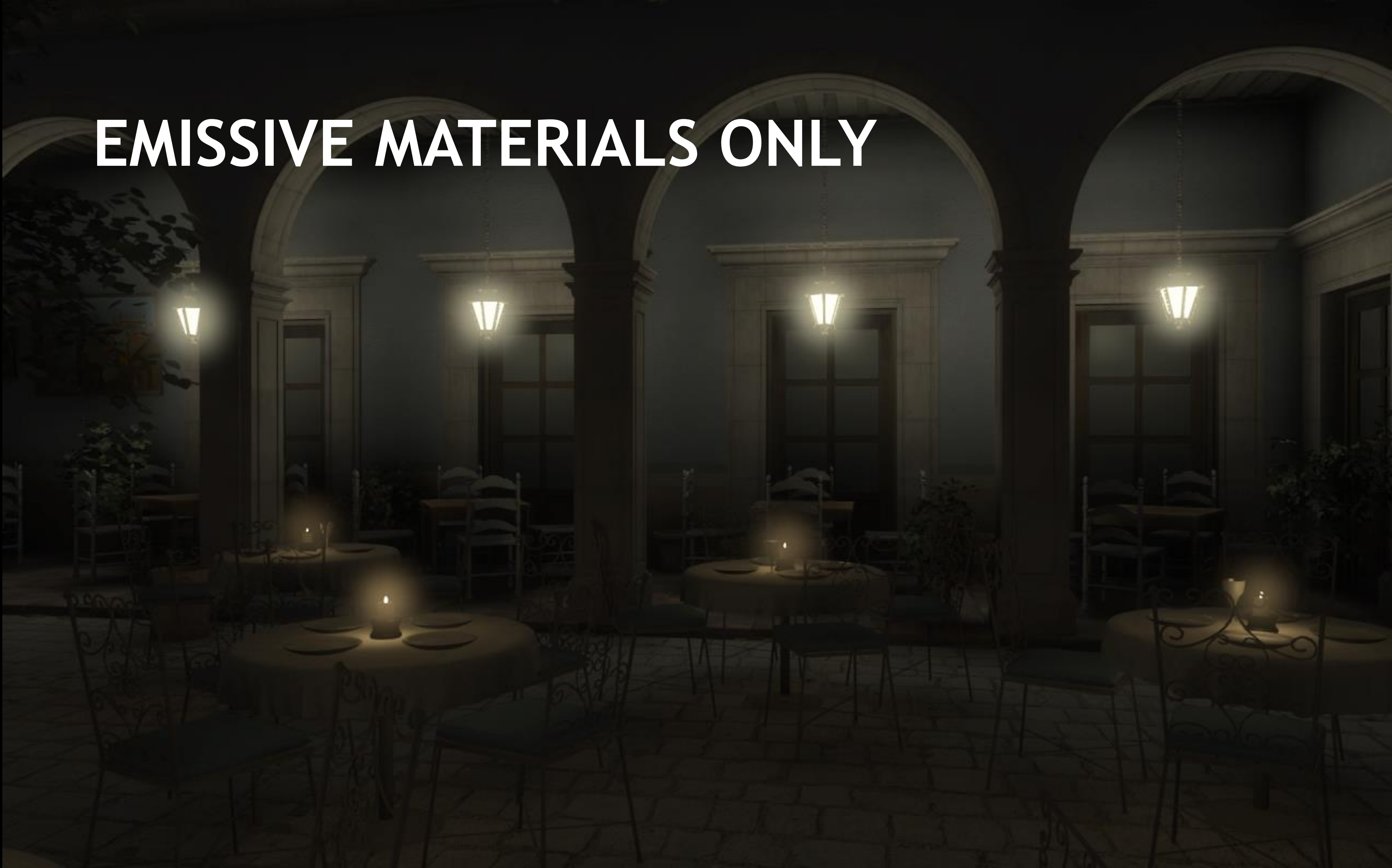
INDIRECT LIGHTING



COMBINED

BEST OF GTC

NVIDIA



EMISSIVE MATERIALS ONLY

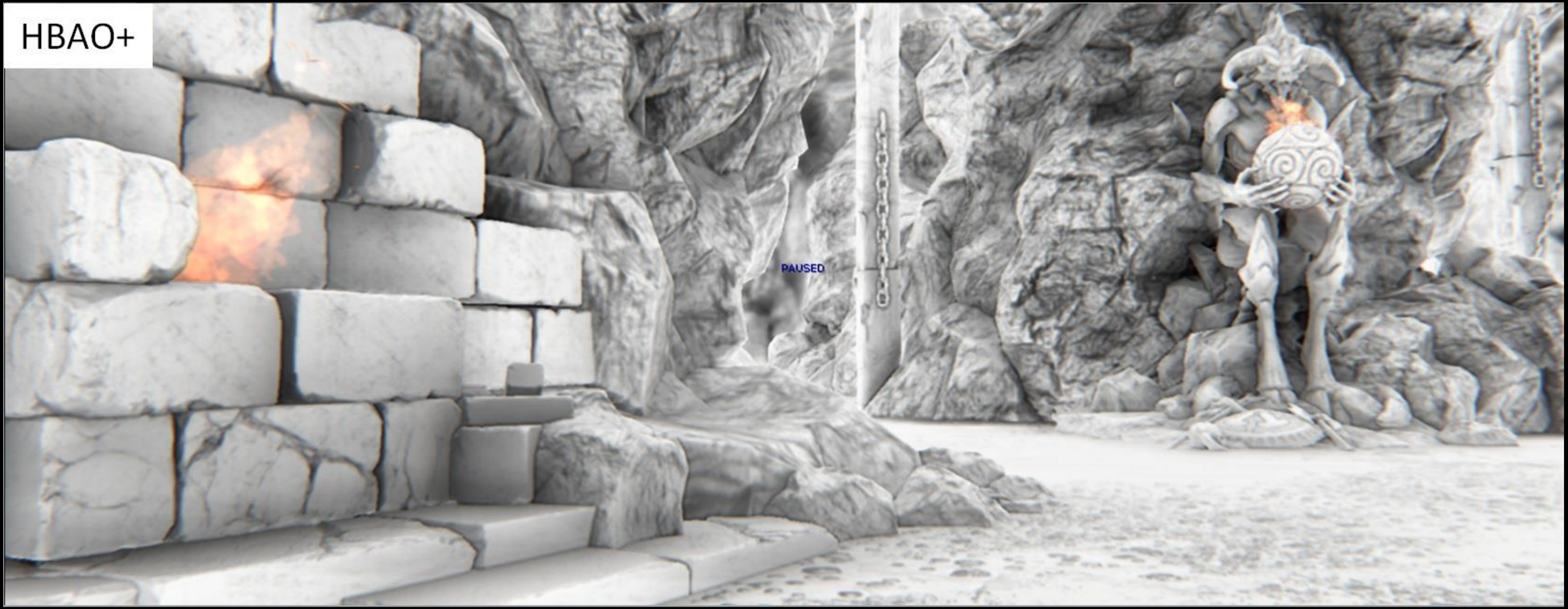
BEST OF GTC

NVIDIA.

MORE EMISSIVE MATERIALS

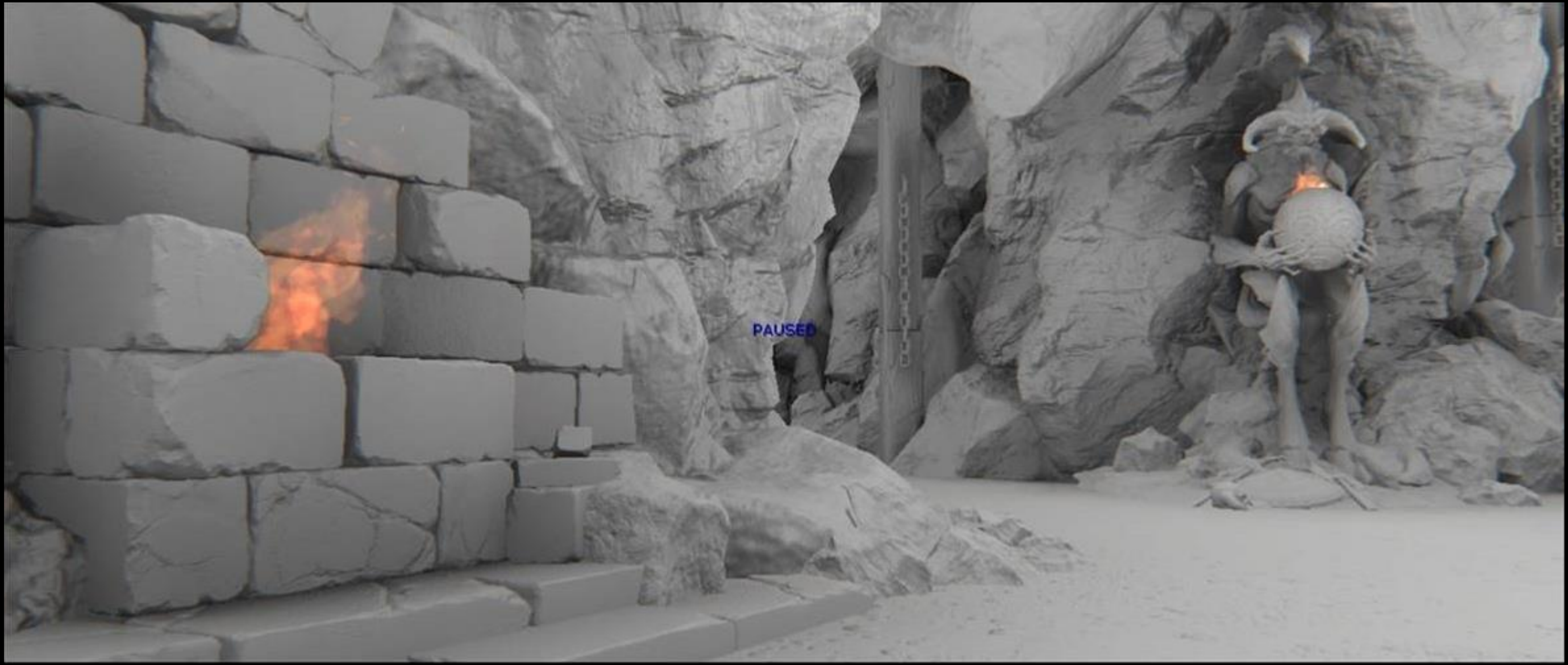
BEST OF GTC

SCREEN-SPACE AMBIENT OCCLUSION



Unreal Engine 4 Effects Cave demo

VOXEL-BASED AMBIENT OCCLUSION



Highlights the volumetric structure of the scene.
3.5x more expensive than HBAO+, including full scene voxelization.

OVERVIEW OF VOXEL CONE TRACING

- Transform the scene into voxels that encode opacity
  - Then downsample the opacity map
- Inject light into voxels that encode emittance or radiosity
  - This includes both emissive materials and light reflected by objects
  - Then downsample the emittance map
- Gather light by tracing cones through the opacity and emittance maps

VOXEL TEXTURE CONTENTS

- Opacity textures
  - 3 or 6 opacity directions for each voxel
  - “How opaque is the voxel when viewed from a certain direction”
  - 6 directions work better for wide cones: less self-shadowing

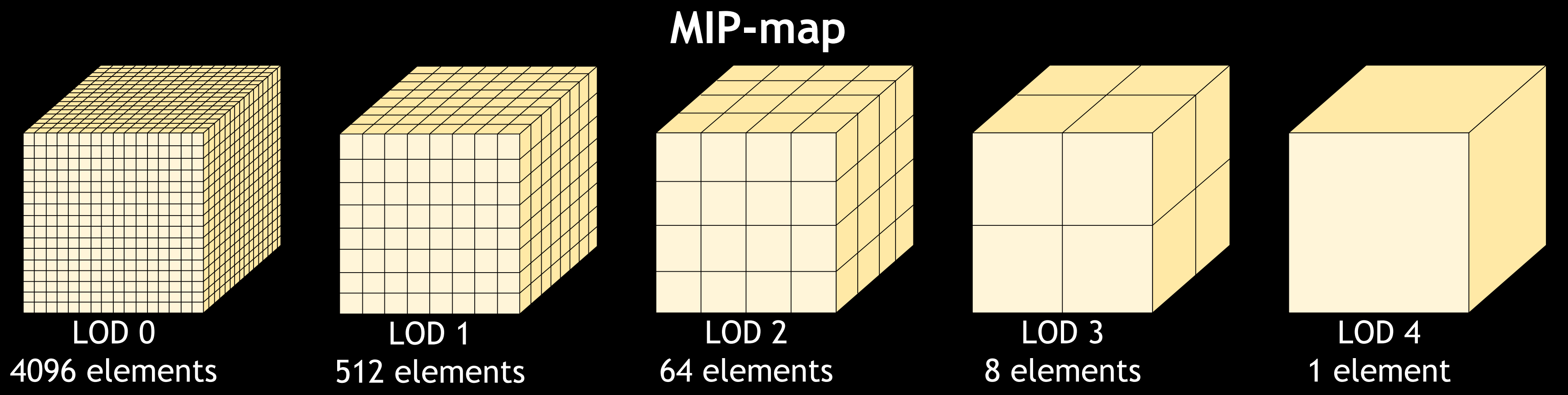
- Emittance textures
  - 3 or 6 emittance directions for each voxel
  - “How much light does the voxel emit to a certain direction”
  - 6 for HQ and second-bounce tracing, 3 for LQ tracing

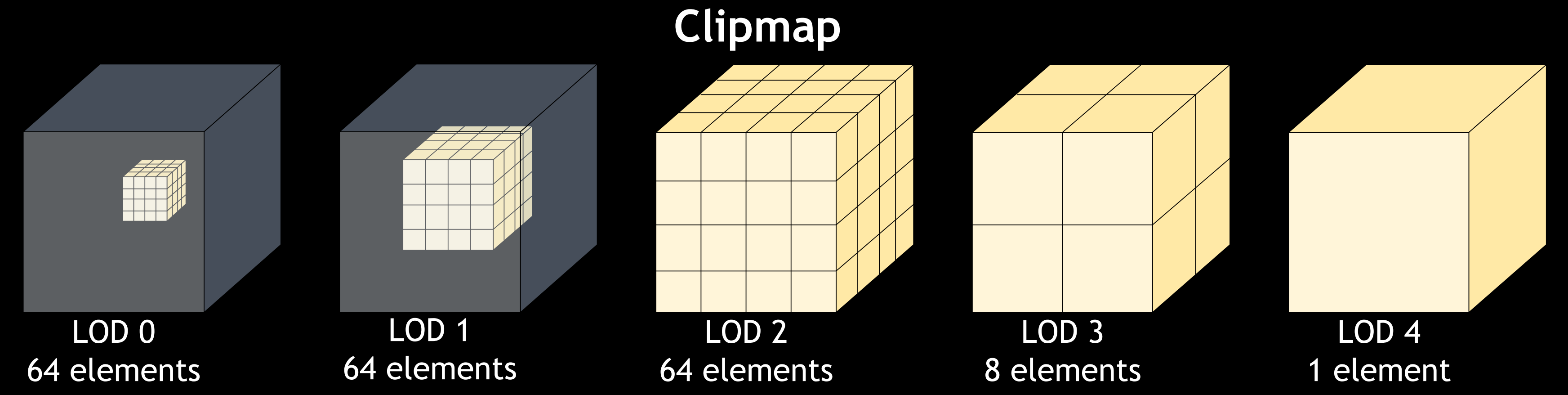
OUR INNOVATION: 3D CLIPMAP

- We store the voxel data in clipmaps
  - Multi-resolution texture
  - Regions near the center have higher spatial resolution
  - Seems to map naturally to cone tracing needs
- A clipmap is easier to build than SVO
  - No nodes, pointers etc., handled by hardware
- A clipmap is easier to read from
  - Same reasons

- Clipmap size is (64…256)^3 with 3…5 levels of detail
  - 16…32 bytes per voxel => 12 MB … 2.5 GB of video memory required

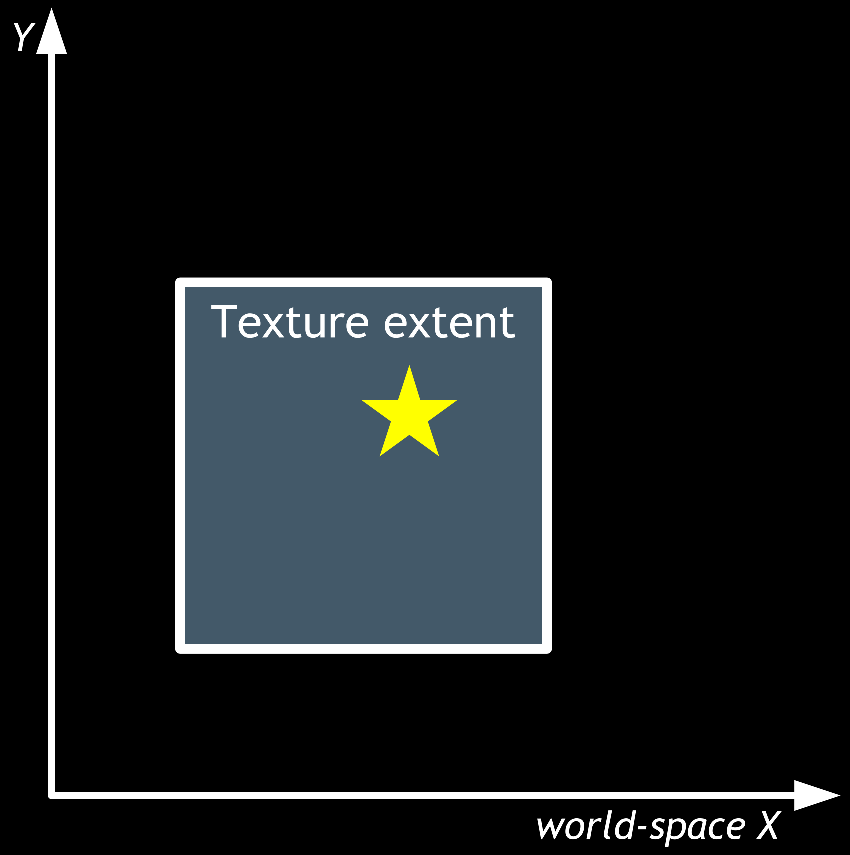
CLIPMAP VS. MIPMAP

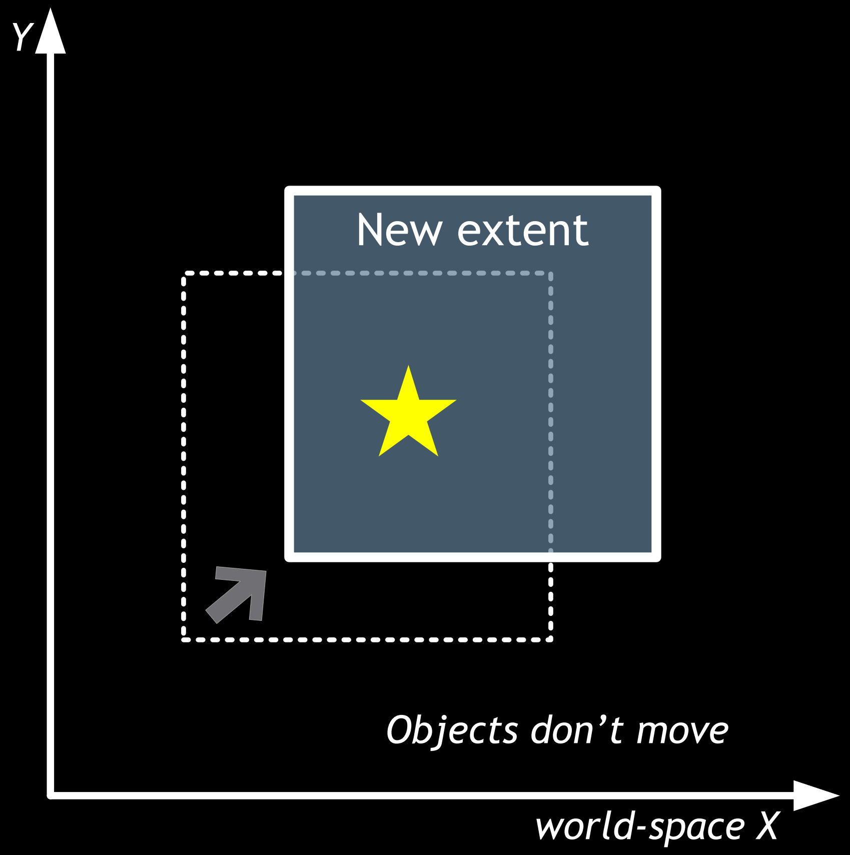




BEST OF GTC

UPDATING THE CLIPMAP DATA



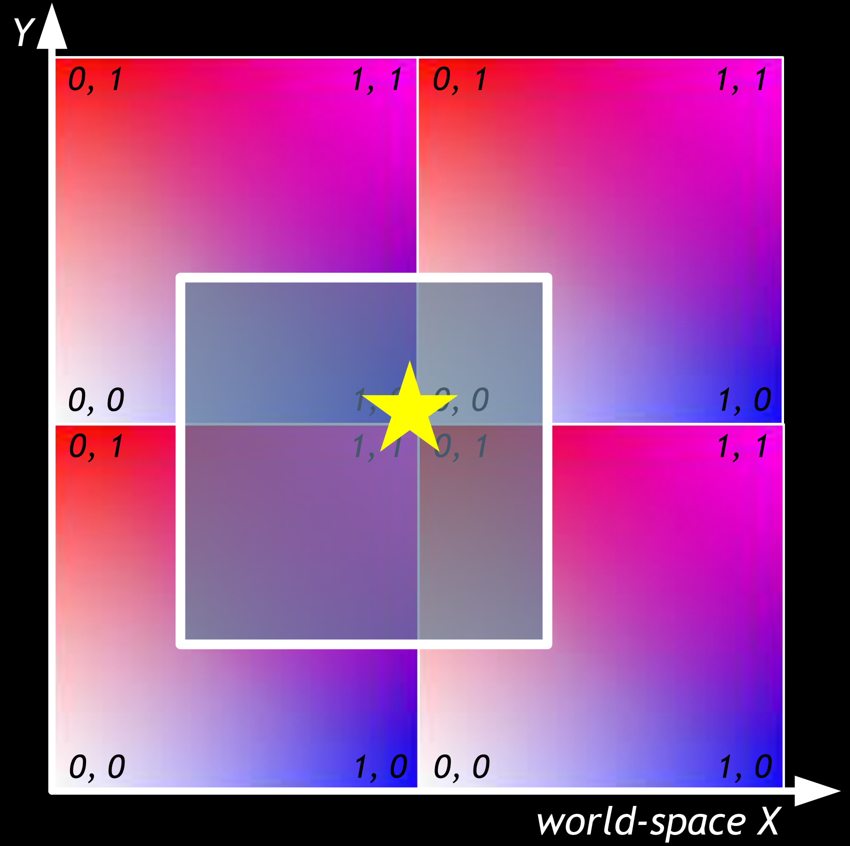


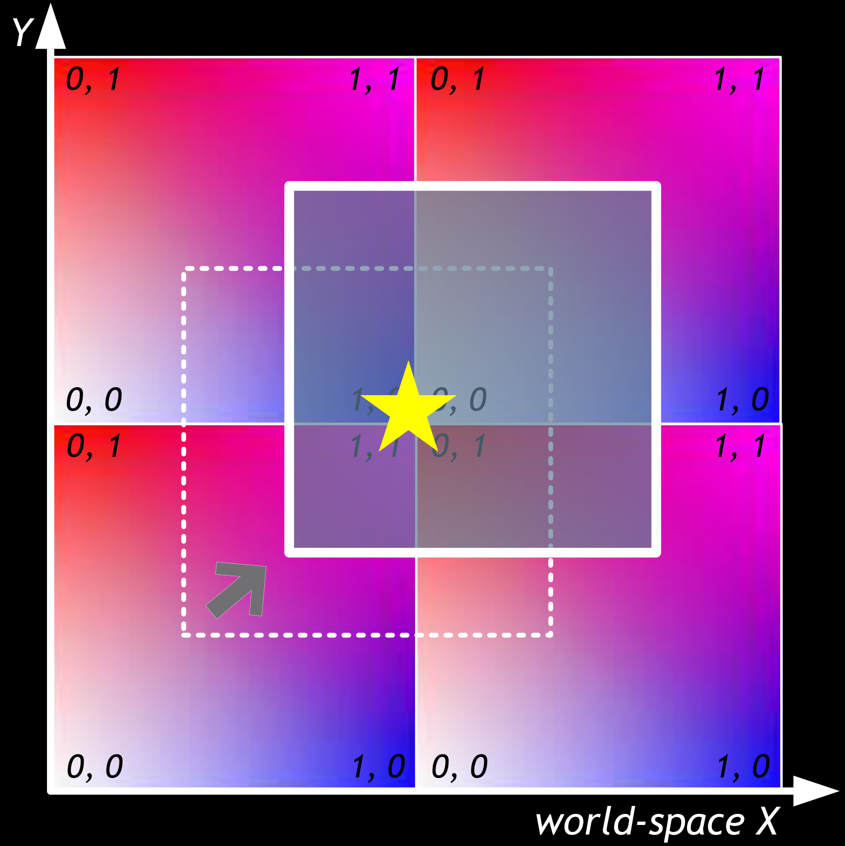
BEST OF GTC

NVIDIA

TOROIDAL ADDRESSING

A fixed point in space always maps to the same address in the clipmap.

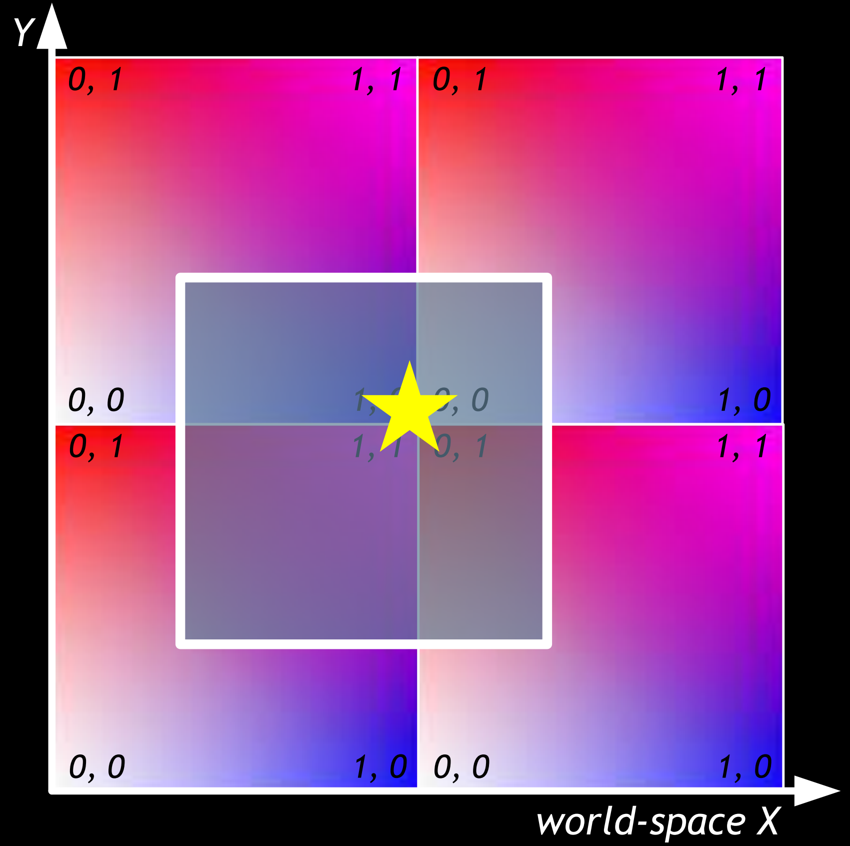


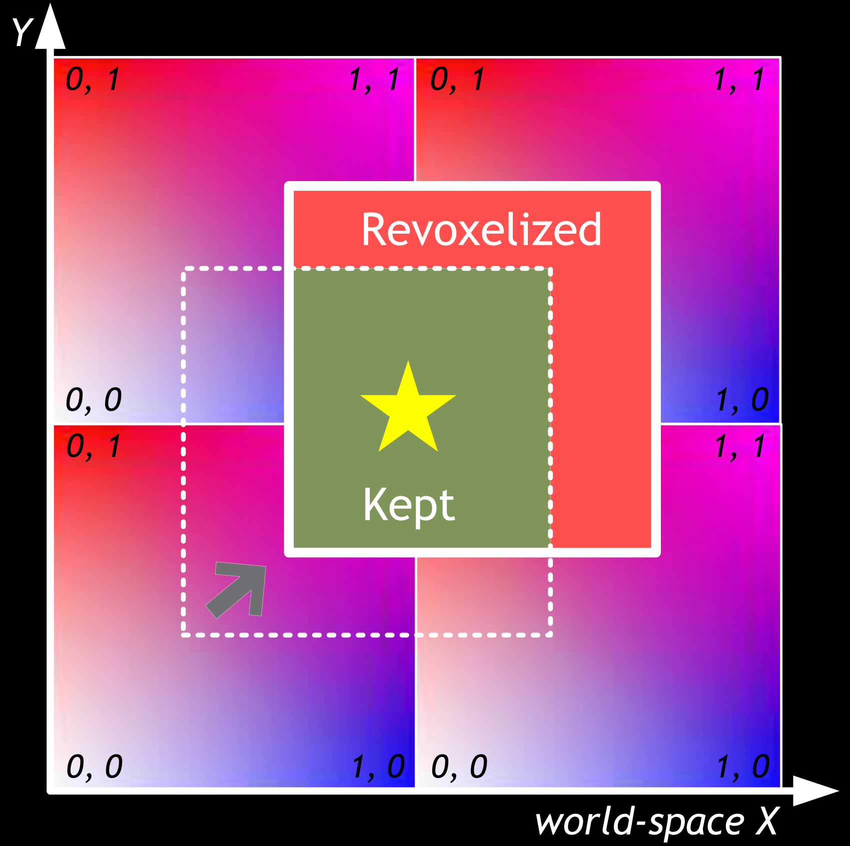


The background shows texture addresses: *frac(worldPos.xy / levelSize.xy)*

INCREMENTAL UPDATES: CLIPMAP MOVES

When the clipmap moves slightly, most of the data remains valid.



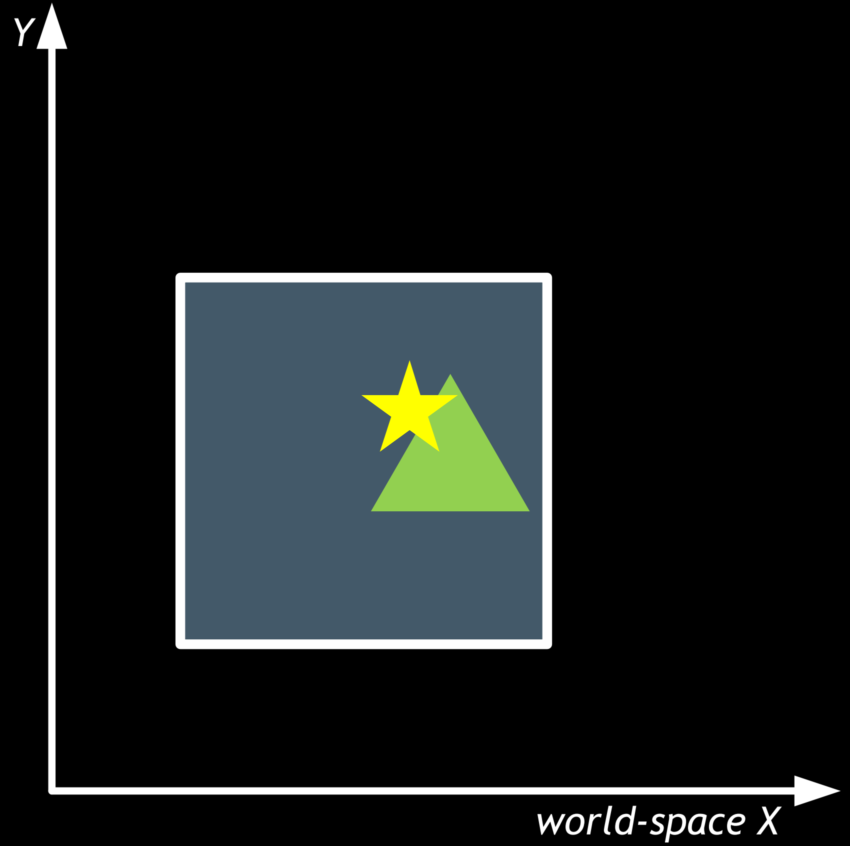


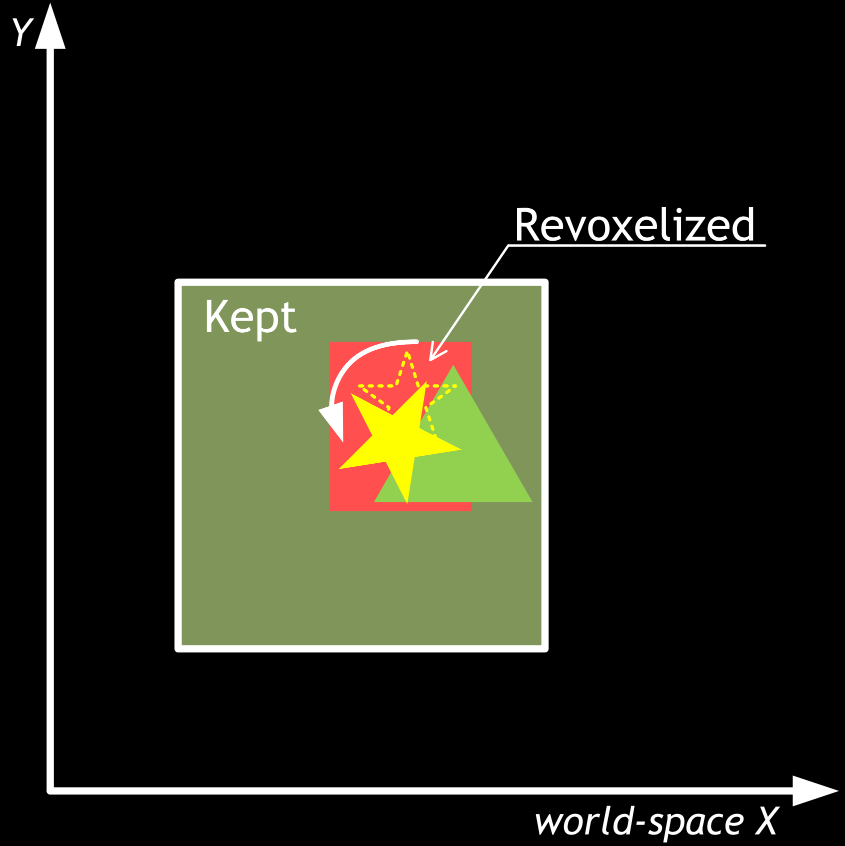
BEST OF GTC

NVIDIA

INCREMENTAL UPDATES: OBJECTS MOVE

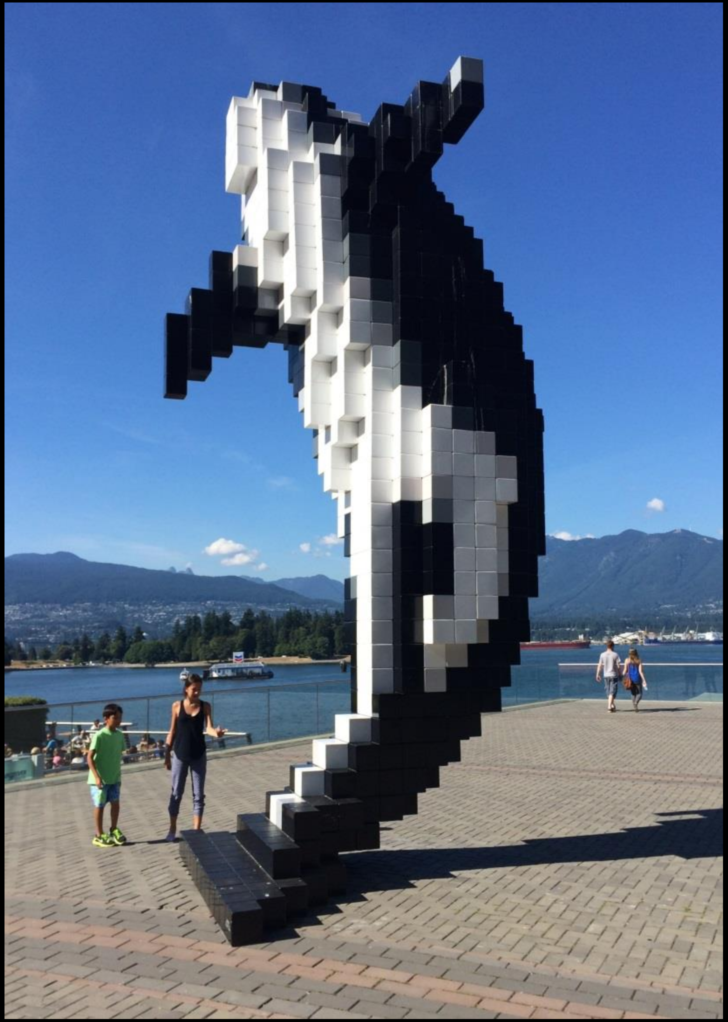
If some objects change, only the corresponding regions need to be revoxelized.





VOXELIZATION

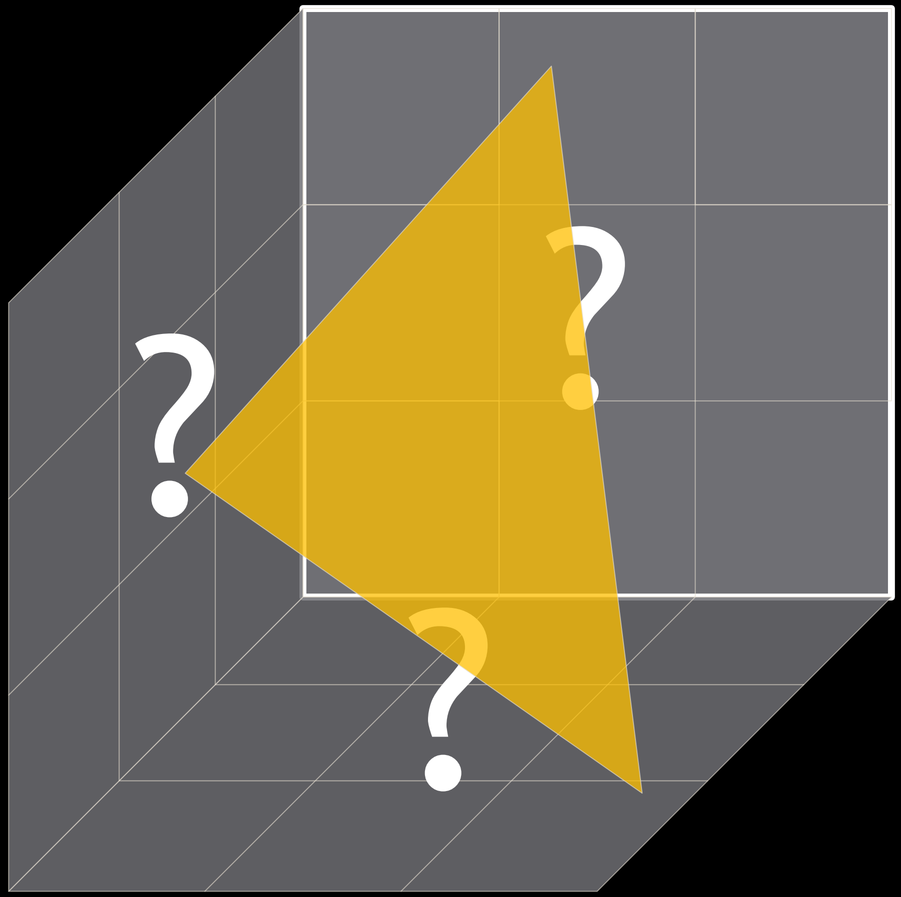
- The process of converting a mesh into a voxel representation
- Different kinds of voxelization:
  - Solid or surface
  - 6 or 26-separating
  - Binary, antialiased or more complex (e.g. surface parameters stored in voxels)
- We use antialiased 6-separating surface voxelization + thickening



A binary voxel representation of an object with color information

BEST OF GTC

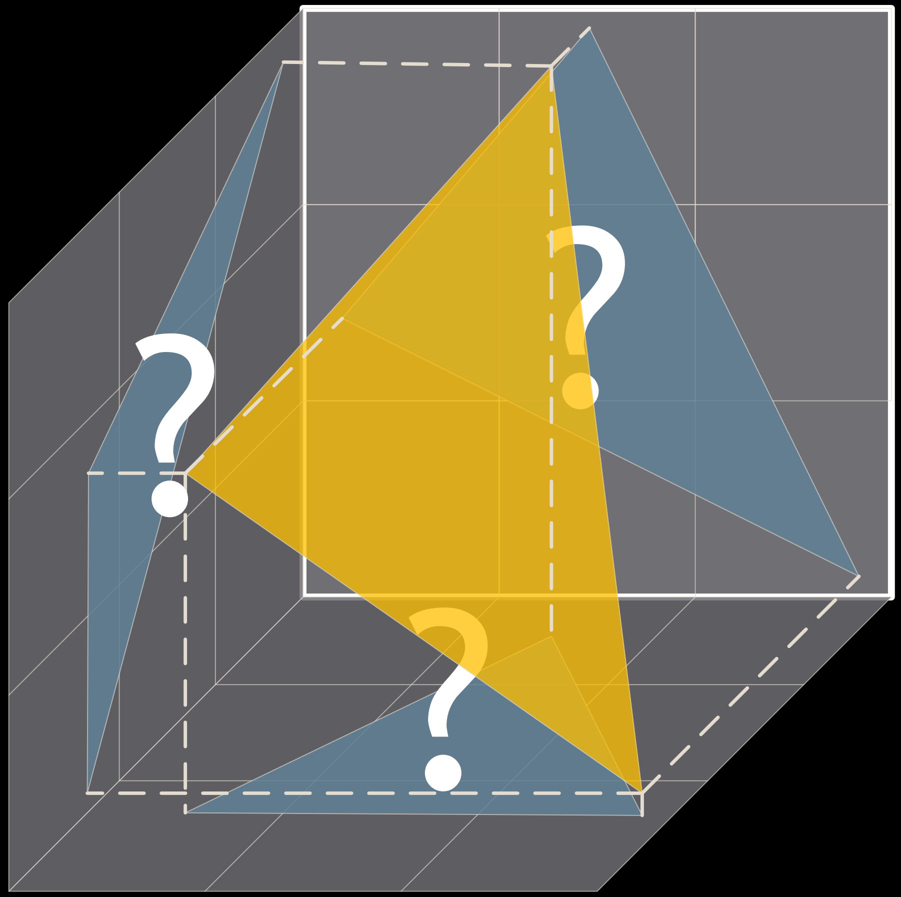
VOXELIZATION FOR OPACITY



This is one voxel.

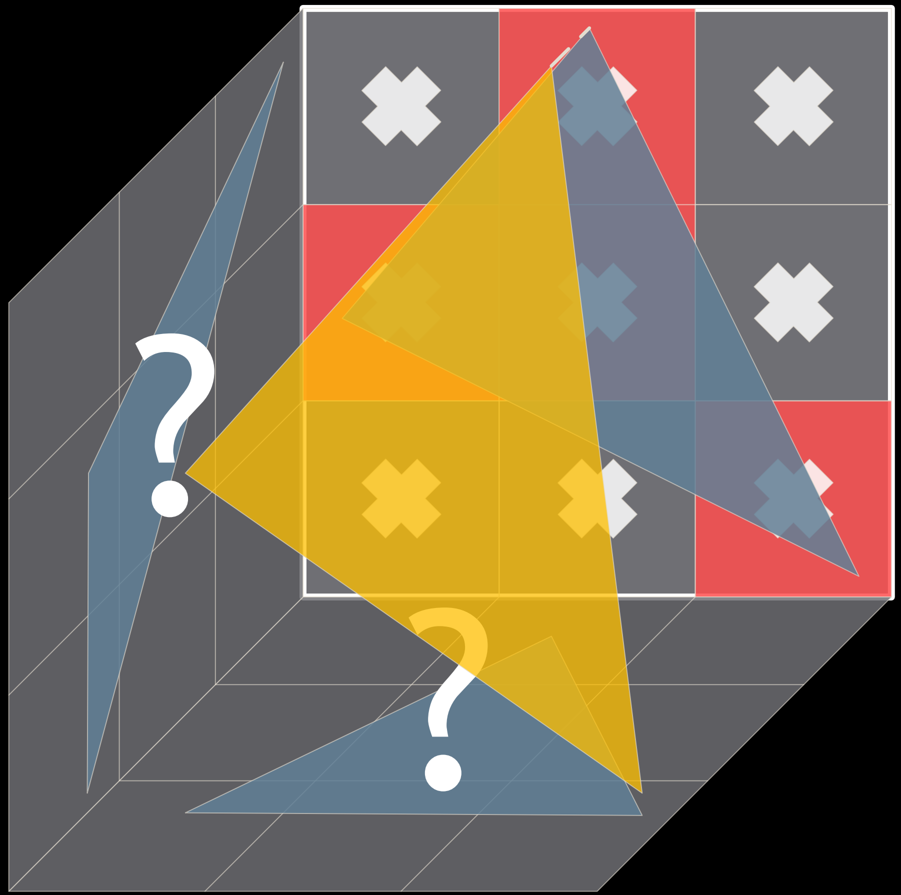
1. We have a triangle and a voxel.

VOXELIZATION FOR OPACITY



2. Select the projection plane that yields the biggest projection area (the back face in this case).

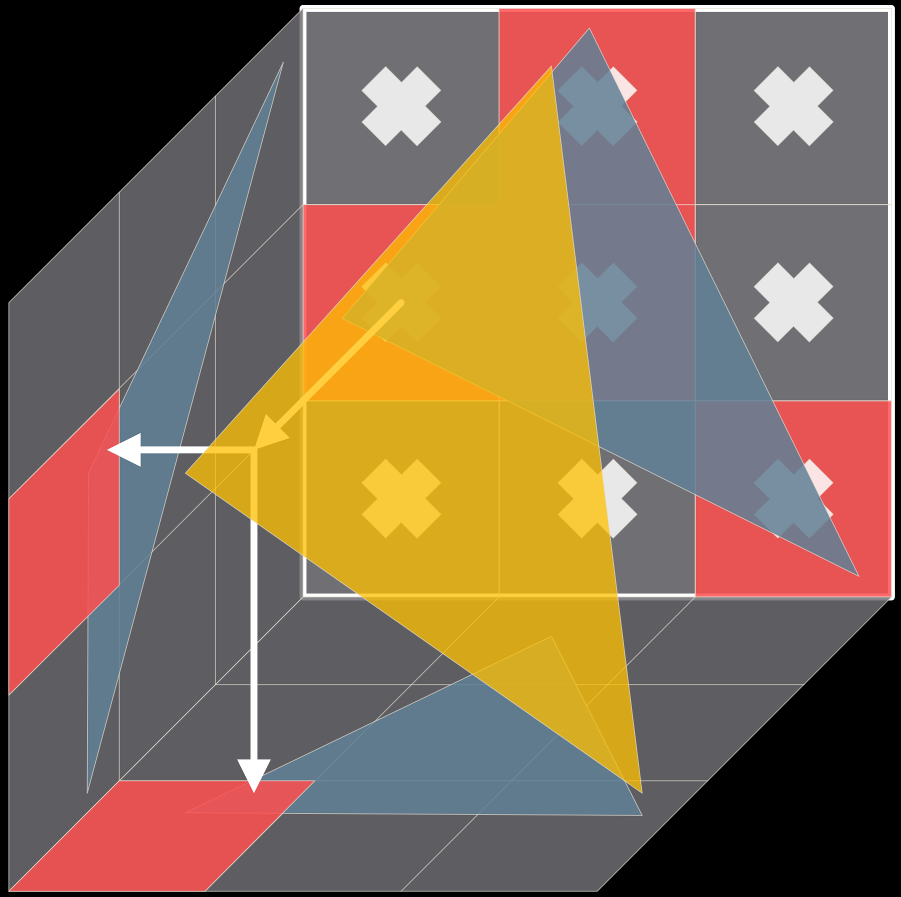
VOXELIZATION FOR OPACITY



3. Rasterize the triangle using MSAA to compute one coverage mask per pixel.

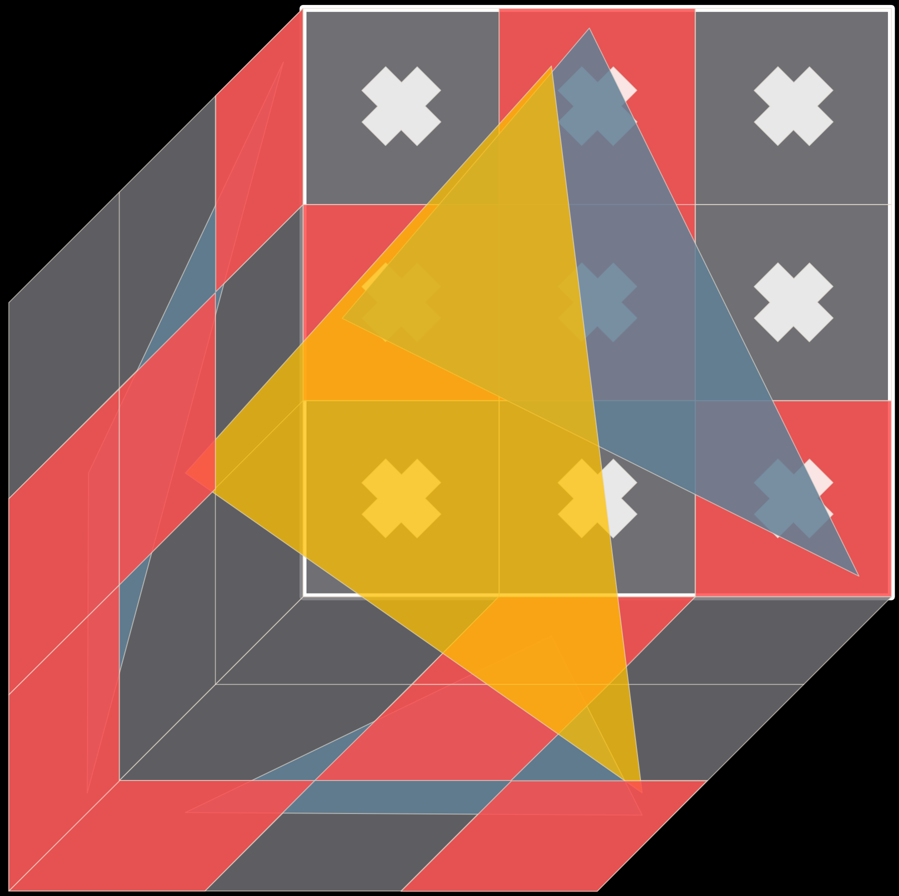
Actual MSAA pattern is different, but we translate those samples onto a regular grid.

VOXELIZATION FOR OPACITY



4. Now take the MSAA samples and reproject them onto other planes using the triangle plane equation.

VOXELIZATION FOR OPACITY

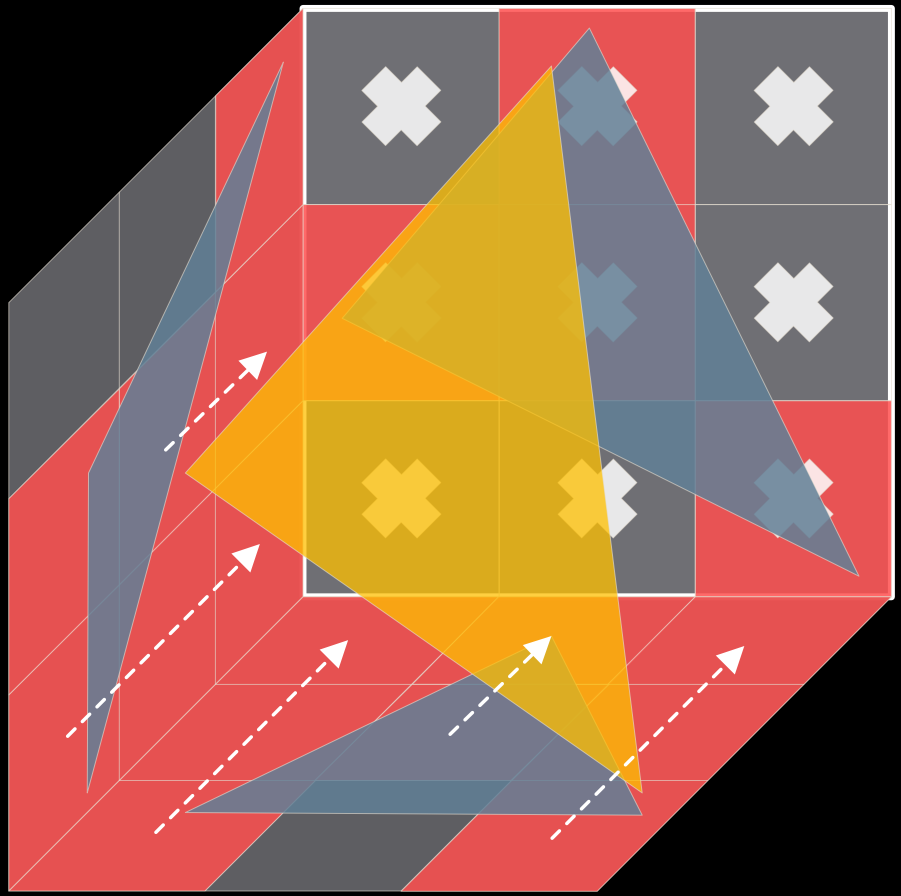


5. Repeat that process for all covered samples.

BEST OF GTC

NVIDIA.

VOXELIZATION FOR OPACITY



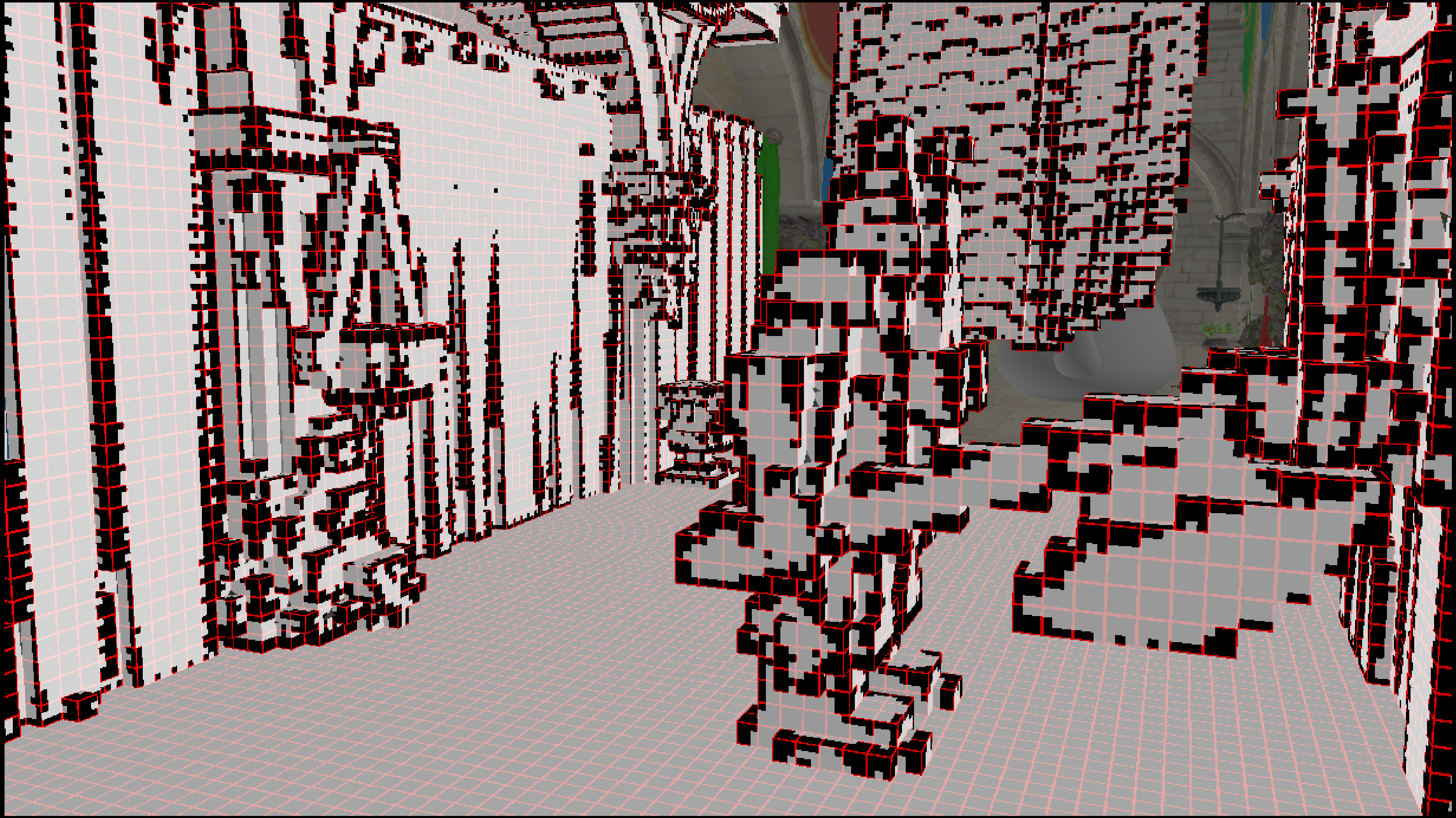
6. Thicken the result by blurring all the reprojected samples.

Some samples may go into the closer or further voxels depending on the Z-slope of the triangle.

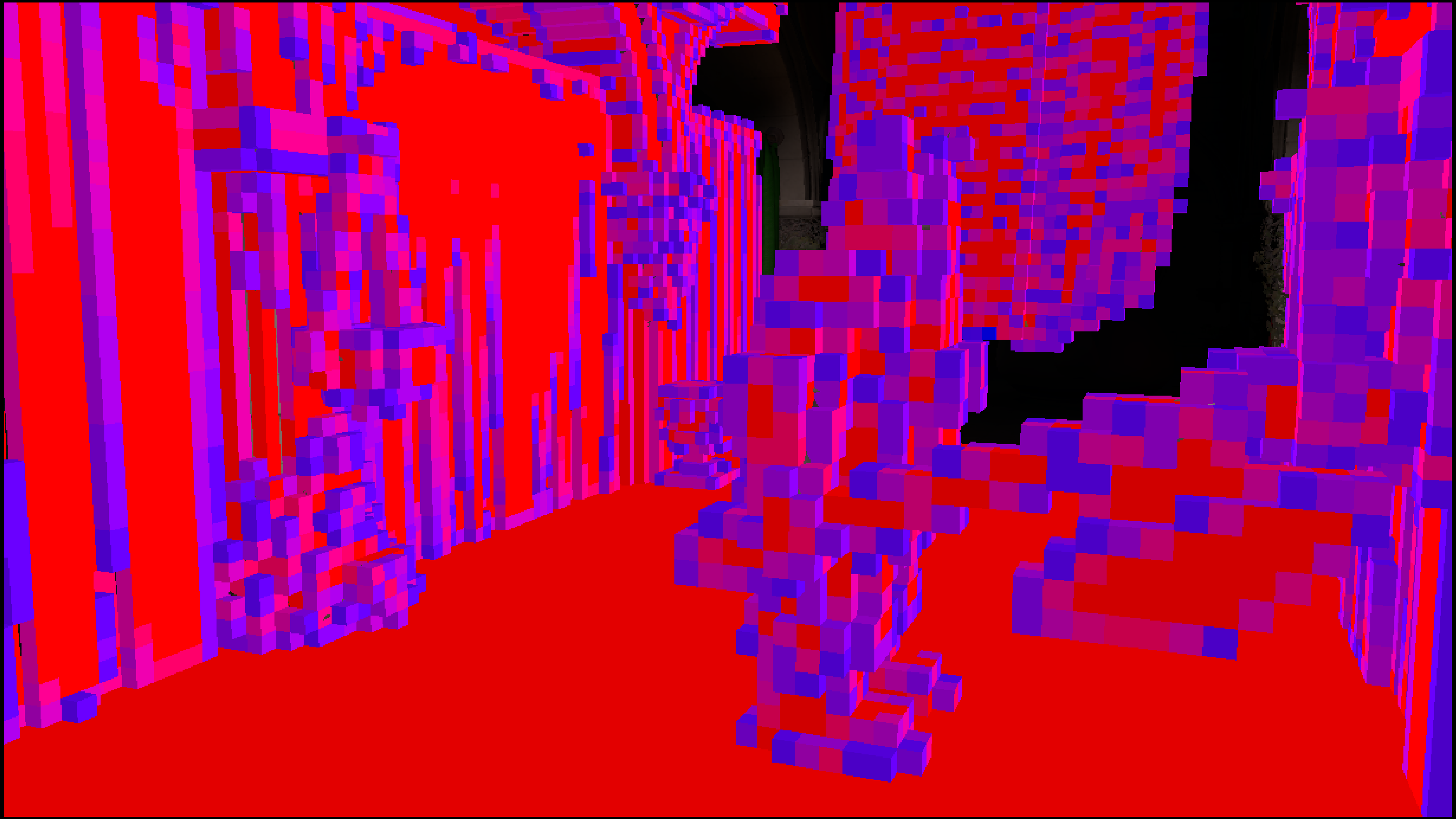
We can add uniform noise (dither) to Z positions of samples to reduce aliasing.

VOXELIZATION: SCENE GEOMETRY

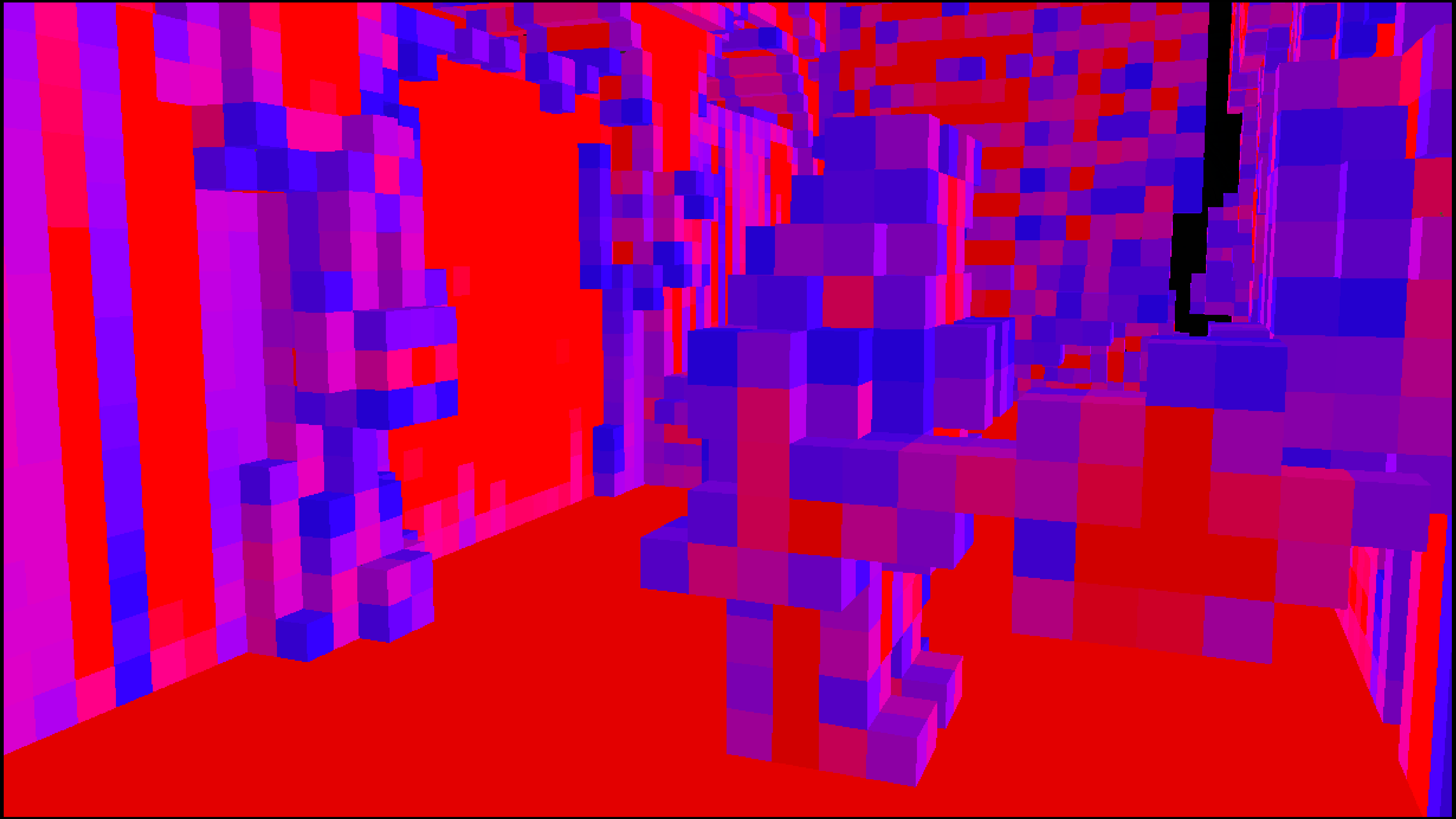
VOXELIZATION: DIRECTIONAL COVERAGE



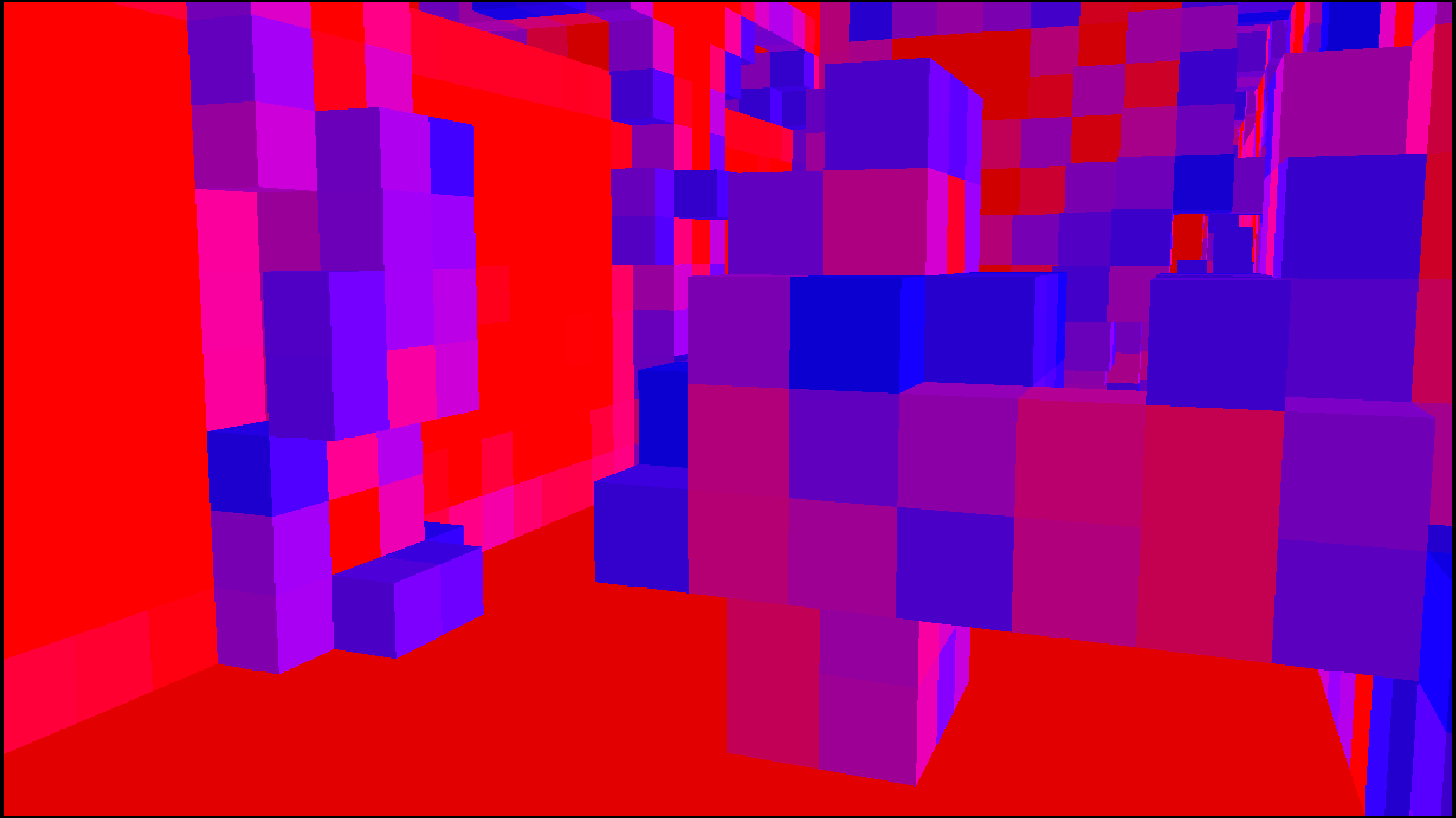
VOXELIZATION: OPACITY



VOXELIZATION: DOWNSAMPLING 1



VOXELIZATION: DOWNSAMPLING 2



VOXELIZATION FOR EMITTANCE

- Step 1: select the projection plane and rasterize the triangle

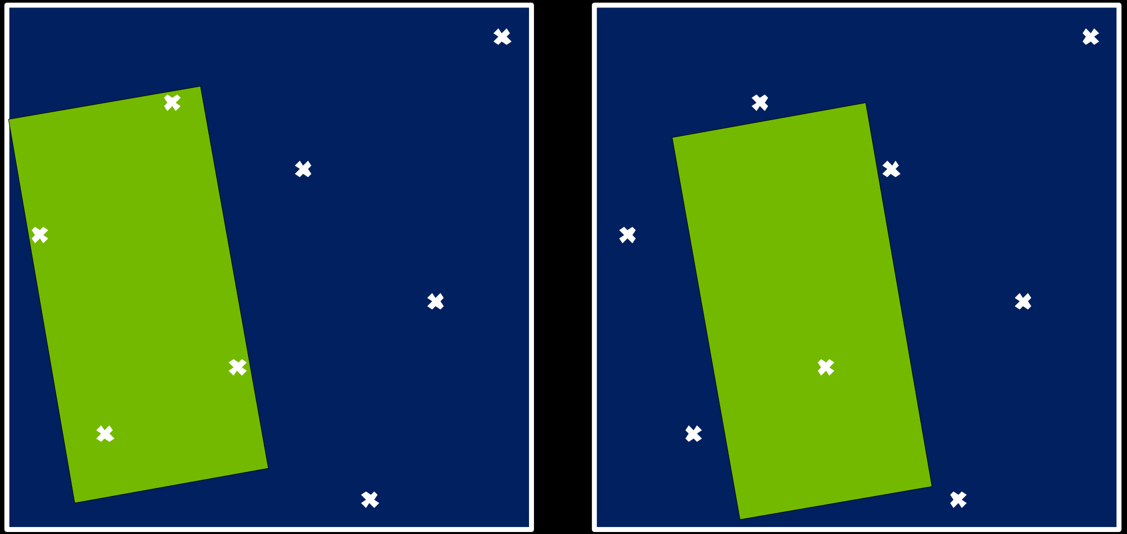
- Step 2: compute the approximate light intensity for each voxel
  - Can use the coverage mask to weigh the emittance texture/color
  - Project the intensity to 3 or 6 directions

- Step 3: accumulate the directional intensities for all rasterized triangles

EMISSIVE VOXELIZATION ALIASING

- Small objects change apparent brightness abruptly
- Mostly appears in the remote regions of the clipmap
- Possible solutions are adaptive supersampling and analytical coverage computation

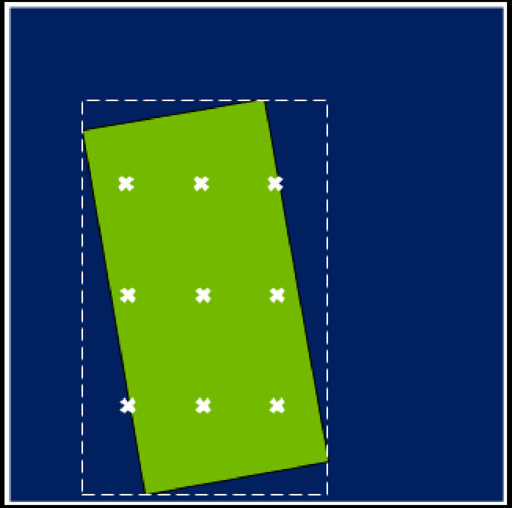
- 8x MSAA pattern
- The object covers:
  - Left: 4 samples
  - Right: 1 sample
  - Flickers when moves



ADAPTIVE SUPERSAMPLING

- Compute the triangle AABB and edge equations in the GS
- Rasterize the triangle conservatively
- Sample the edge equations on a regular grid within the bounding box in the PS
- Number of samples depends on the clip level / voxel size

- The result: no flickering at all.

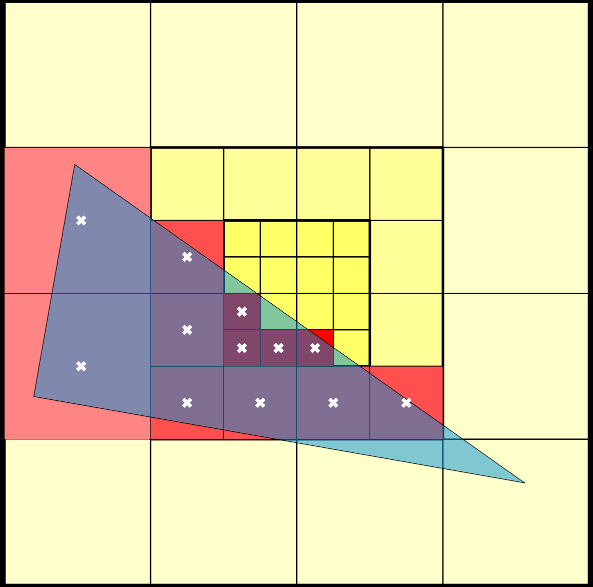


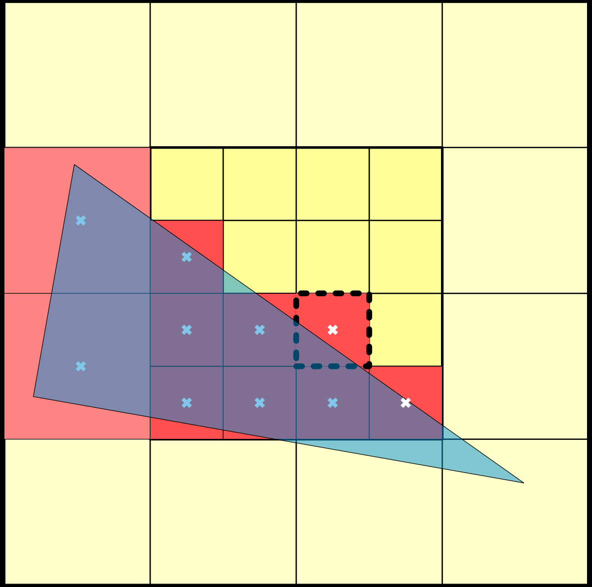
MULTI-RESOLUTION VOXELIZATION

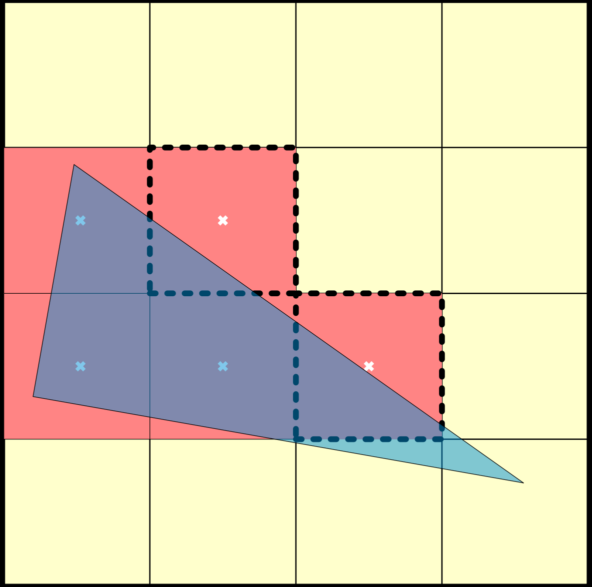
- MIP-map: downsample finer levels to get coarser levels
- Clipmap: there are no finer levels for most of coarser levels
- Rasterize every triangle at several resolutions
  - Obtain center regions of coarser levels by downsampling finer levels
  - Use GS instancing to rasterize one triangle several times

MULTI-RESOLUTION VOXELIZATION

Voxelization with downsampling yields higher quality results than multi-res voxelization.

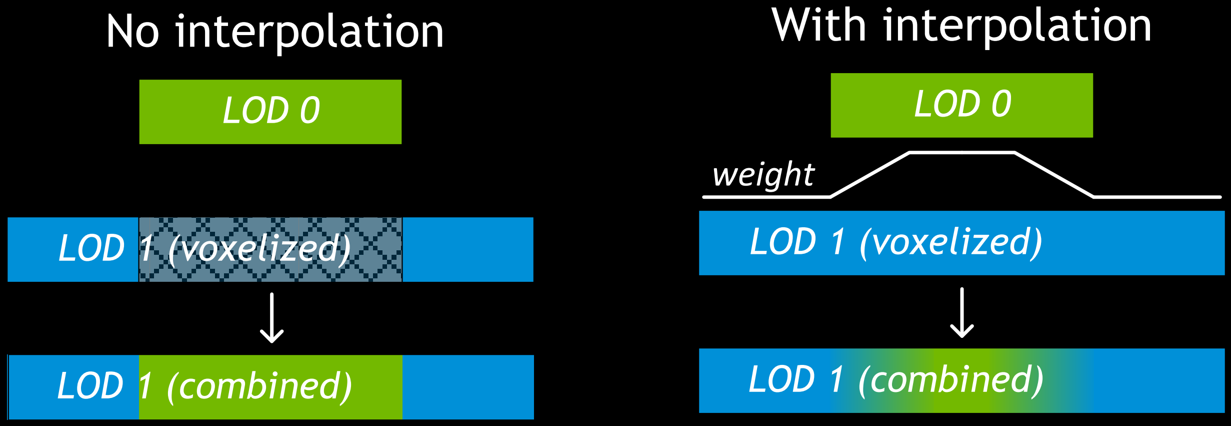

Rasterize…


Downsample once


Downsample twice

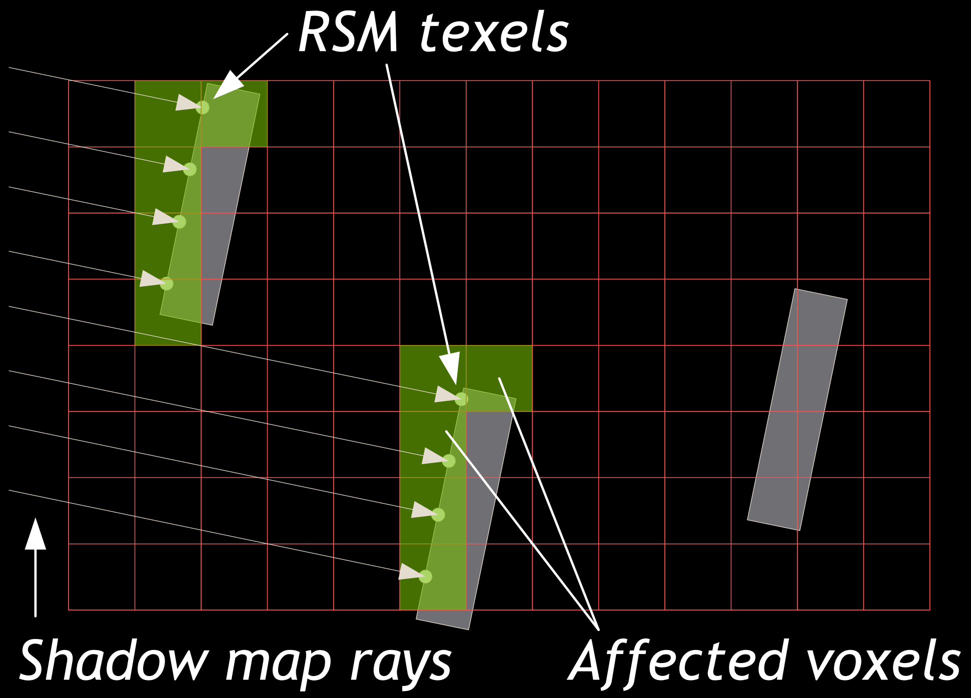
OPACITY INTERPOLATION

- Issue: When an object moves from one clip level to another, its coarse representation changes
- Solution: interpolate between downsampled and directly voxelized representations
  - The weights are derived from the distance to the clipmap anchor
  - Smooth changes in AO following the camera



LIGHT INJECTION

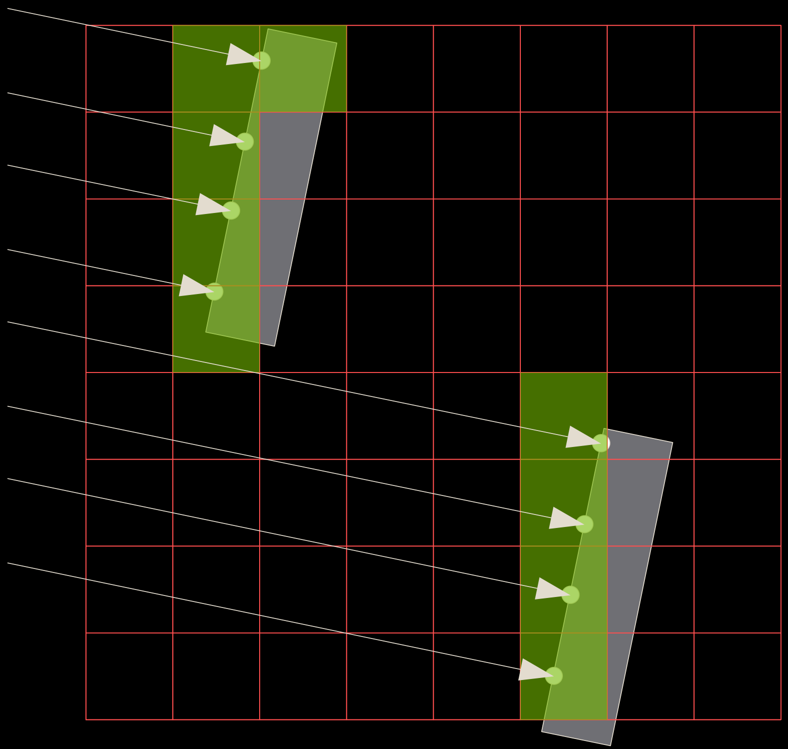
- A process of calculating emittance of voxels that contain surfaces lit by direct lights
- We can take information from reflective shadow maps (RSMs)

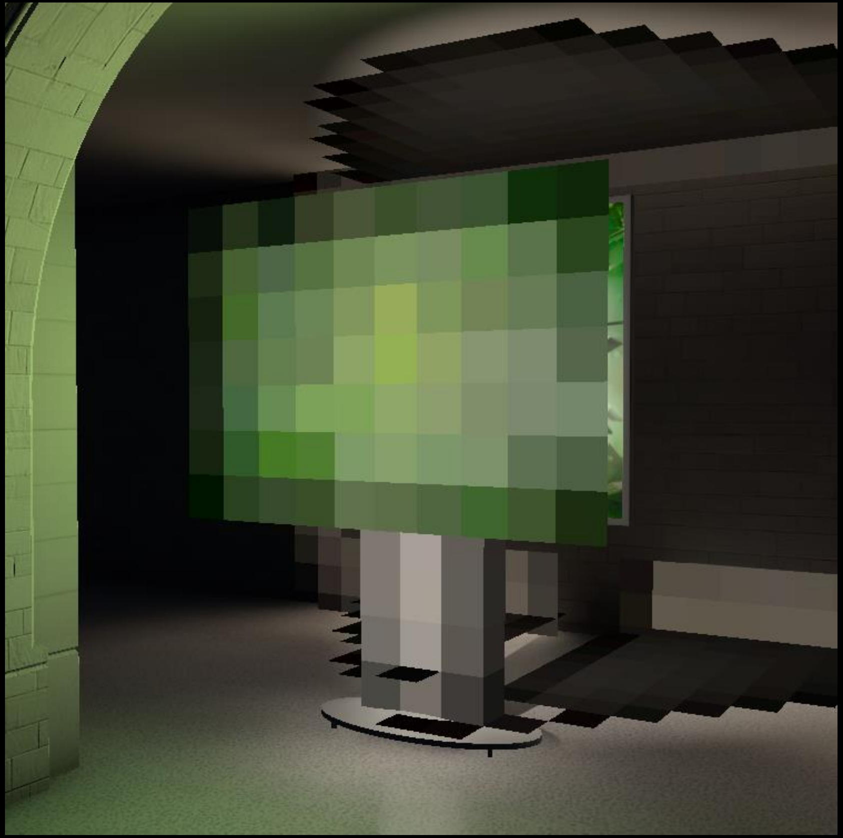


RSM LIGHT INJECTION ALGORITHMS

- Simplest option: test every voxel center against the RSM
  - Consider only voxels with nonzero opacity
  - If a voxel is lit, take the color and normal from the RSM
  - Problems: aliasing, false lighting on object boundaries
- Better option: gather all RSM texels that belong to the voxel
  - Many texture fetches per voxel, most of them are useless
- Even better option: scatter RSM texels into voxels using atomic operations
  - Lots of atomic collisions if there are many texels per voxel
  - Not very stable if there are few texels per voxel

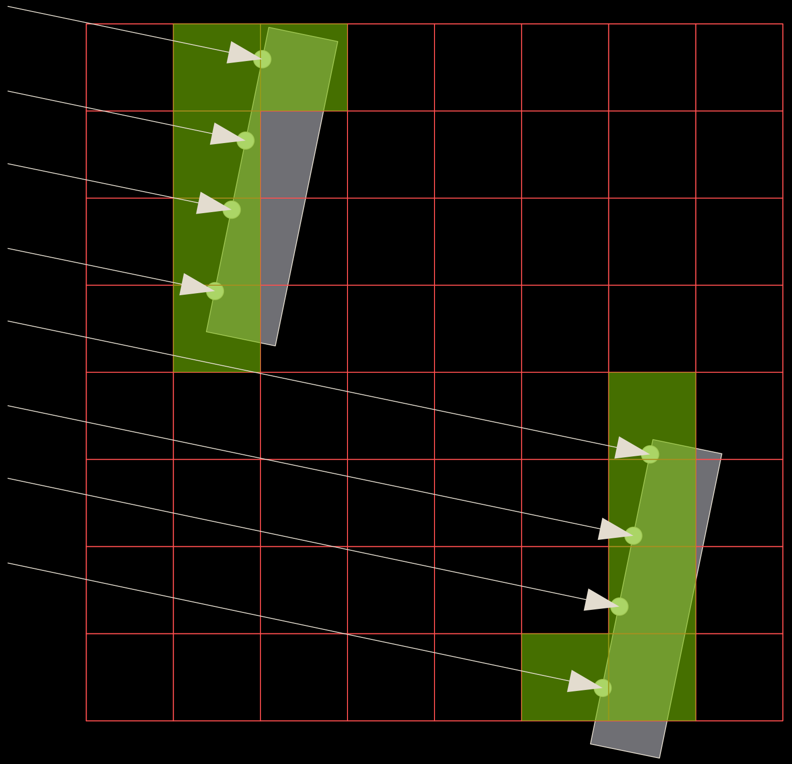
LIGHT INJECTION ALIASING

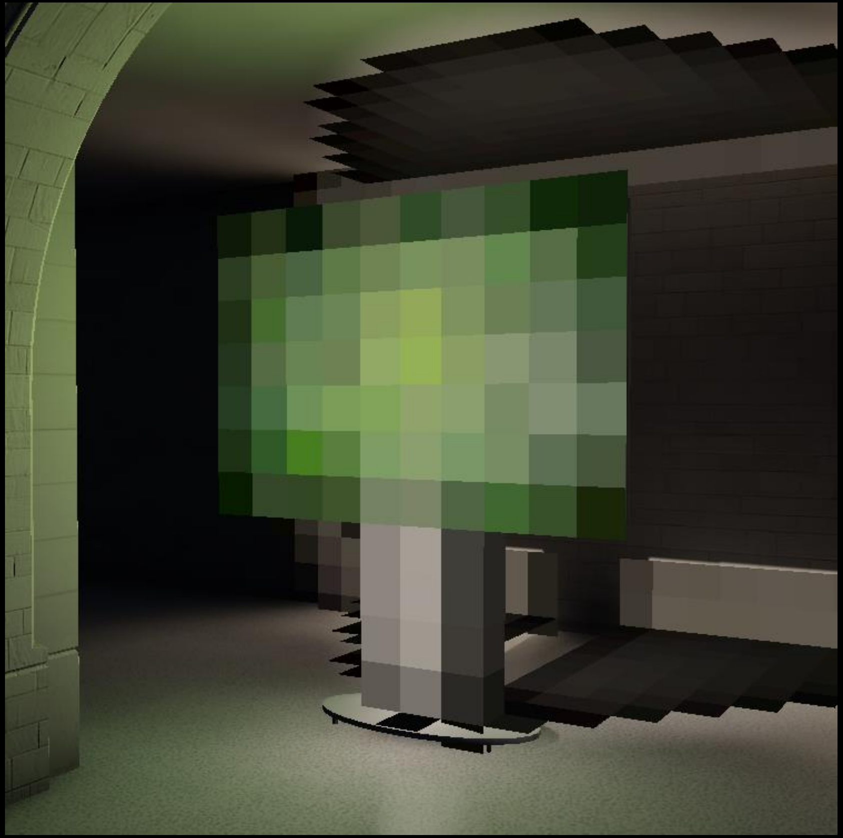




Slight changes in object or light positions sometimes change the lighting significantly.

LIGHT INJECTION ALIASING





Slight changes in object or light positions sometimes change the lighting significantly.

LIGHT INJECTION PRE-FILTERING

- Need to pre-filter the RSM point cloud before injecting
  - Every texel will affect more than 1 voxel (3^3 or even 5^3)
  - Expensive to inject with scattering: atomics will be a bottleneck

- Solution: add noise to point positions and hope there are enough points to filter out that noise
  - Dithering once again
  - The offset is computed as (normal * voxelSize * [-0.5…0.5])
  - Using different noise scales for different clip levels results in a mismatch between injected and downsampled light…

VOXELIZATION BASED LIGHT INJECTION

- Alternative approach to light injection
  - Voxelize all potentially lit scene geometry as emissive objects
  - Compute the reflected light amount for every rasterized fragment using the material shader and shadow maps

- Compared to scatter light injection...
  - No need for the expensive-to-render RSMs
  - Higher quality with low-res shadow maps
  - Better performance with simple enough geometry and high-res shadow maps

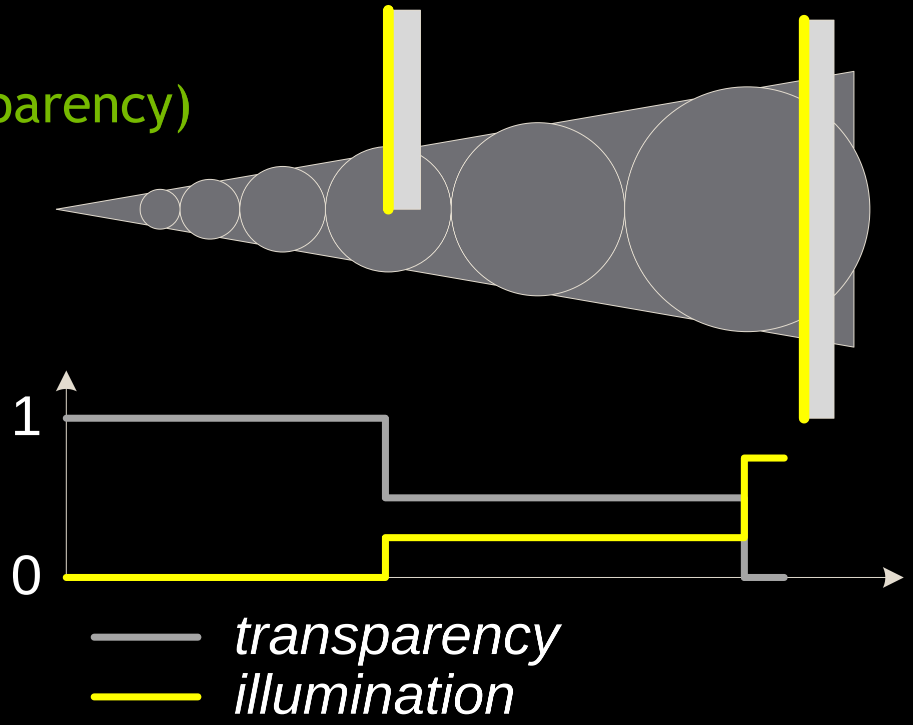
BEST OF GTC

NVIDIA.

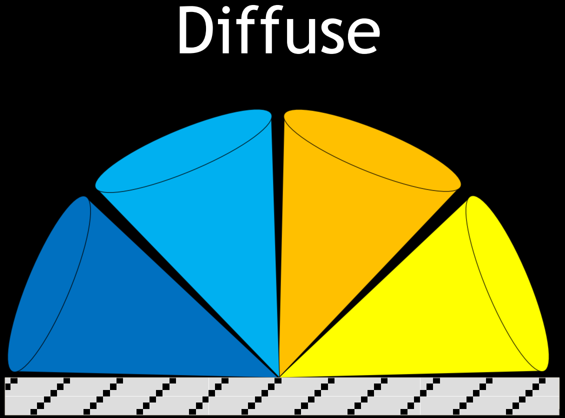
CONE TRACING BASICS

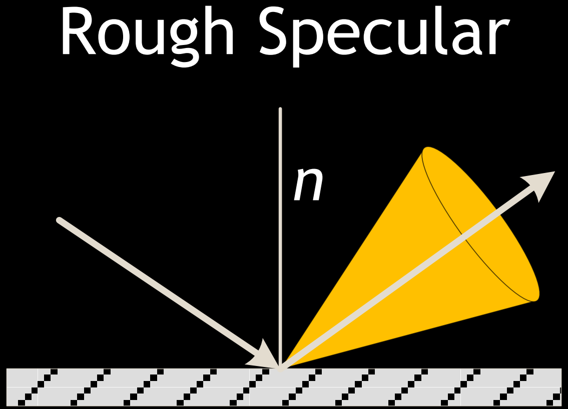
- Several cones are traced from every visible surface
- A cone marches through the clipmap accumulating:
  - transparency (1-opacity)
  - illumination (emittance * transparency)

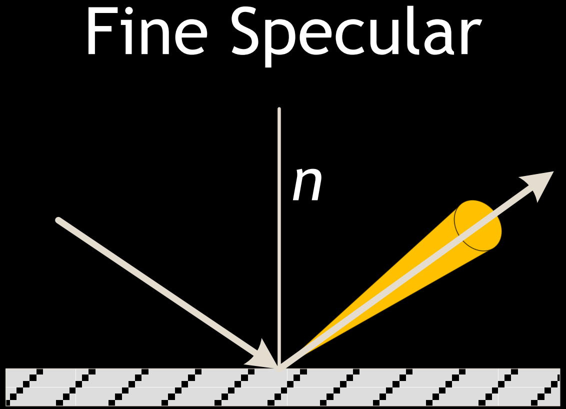
Each sample is taken from a coarser LOD than the previous one.

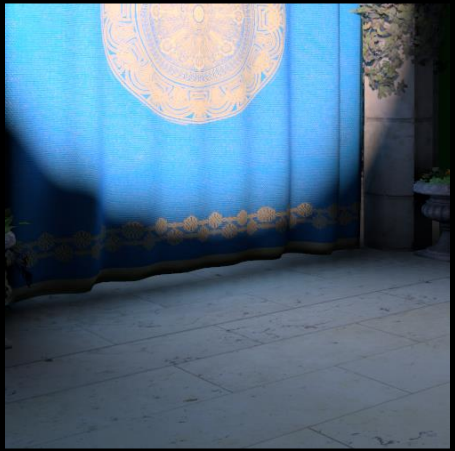


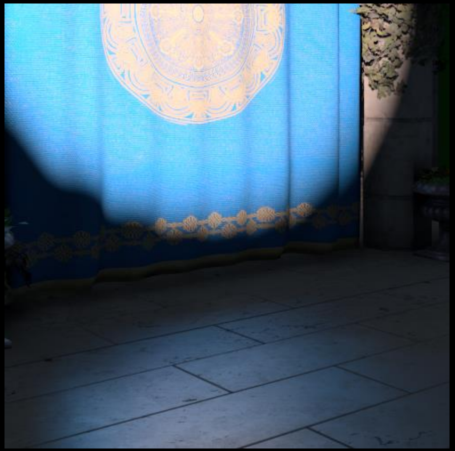
DIFFUSE AND SPECULAR CONE TRACING

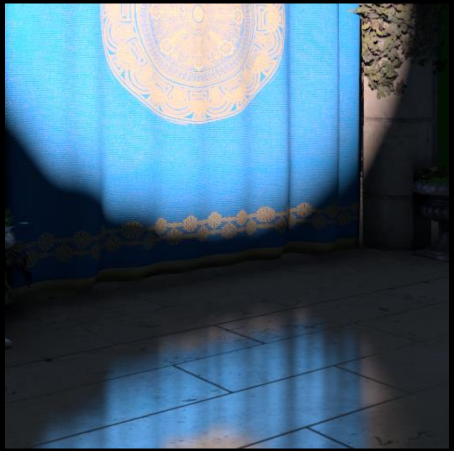










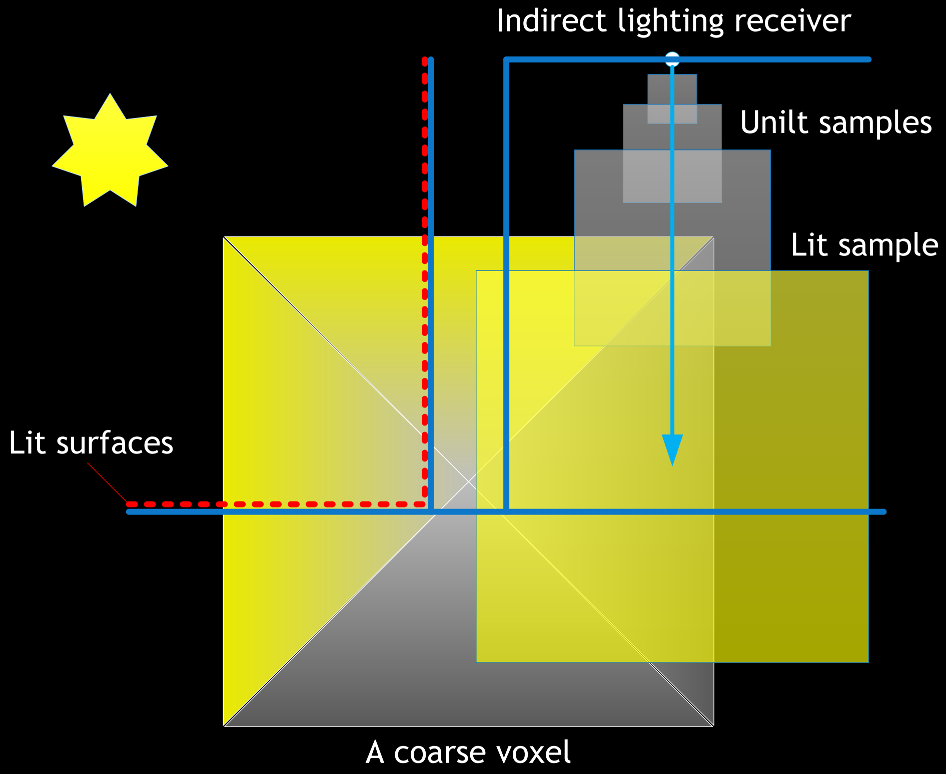


SPARSE DIFFUSE CONE TRACING

- Diffuse lighting is usually low-frequency
  - HQ cone tracing for every pixel is redundant
- We trace every Nth pixel on the screen
  - N = {1, 4, 9, 16}
  - Rotated grid pattern to reduce aliasing
- Interpolate using a bilateral filter
  - MSAA resolve fits naturally into the interpolation pass

BEST OF GTC

LIGHT LEAKING ISSUE



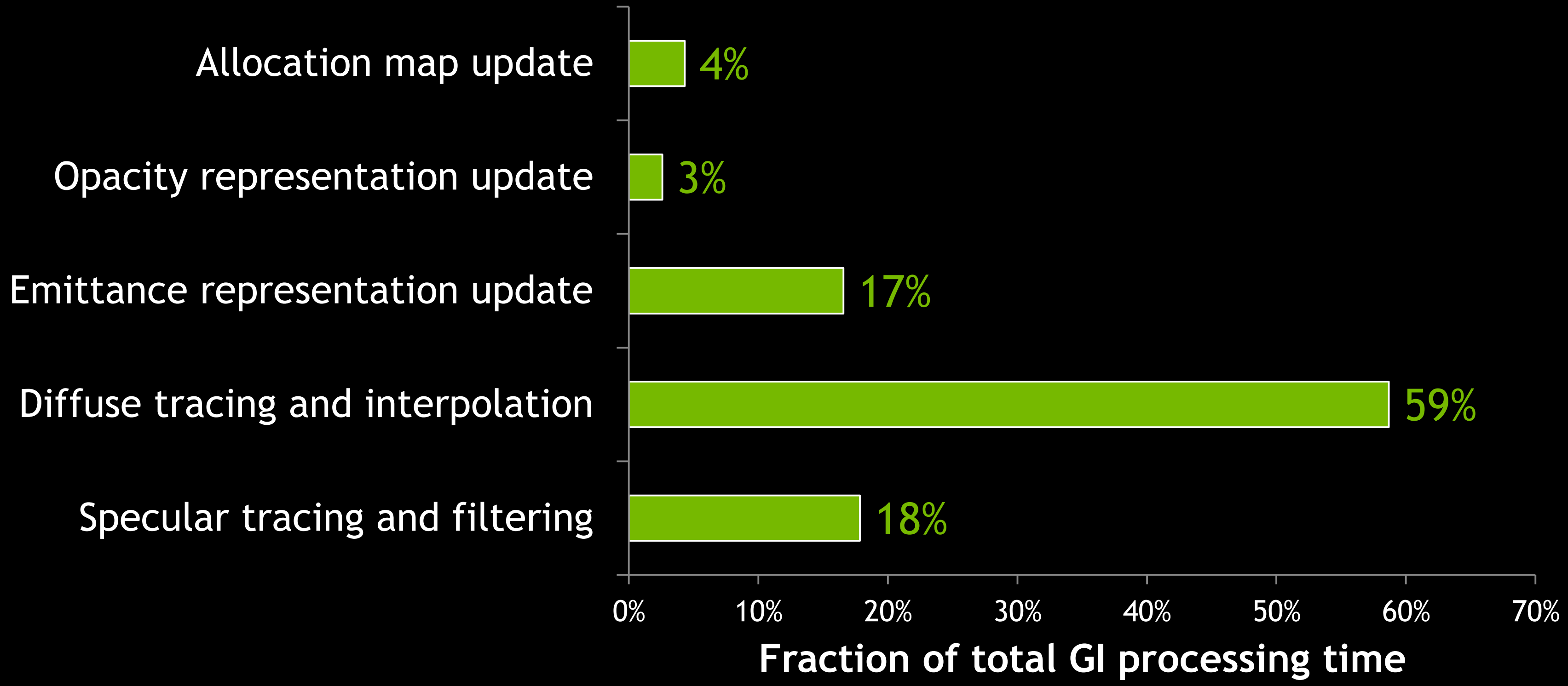
PERFORMANCE

- Full GI is practical on current mainstream GPUs
  - e.g. GeForce GTX 770
- Voxel-based AO works well on low-end GPUs, too
  - Looks much better than SSAO
- GI processing time per frame, in ms:

	AO only	Med	High	Ultra
GTX 650 (GK107)	14.3	28.1		
GTX 770 (GK104)	3.8	7.4	12.9	
GTX TITAN (GK110)	3.1	6.6	9.6	25.4

@ 1920x1080

RENDERING TIME BREAKDOWN



High quality settings: 1920x1080, 17 cones, trace every 9th pixel. GTX TITAN.

CONCLUSION

- Fully dynamic GI becomes practical on mainstream GPUs
  - Low end GPUs can benefit from much higher-quality AO

- Future work
  - Further reduction of aliasing issues
  - Performance optimizations
  - Reduce the amount of content changes necessary

- Planned to be released as NVIDIA global illumination library

BEST OF GTC

QUESTIONS?

- Send them to me: alpanteleev@nvidia.com

- Thanks:
  - Cyril Crassin
  - Evgeny Makarov
  - Khariton Kantiev
  - Monier Maher
  - Holger Gruen
  - Lucas Magder
  - Chris Cowan
  - Yury Uralsky
  - Sergey Bolotov
  - Alexey Barkovoy
  - Miguel Sainz
  - Louis Bavoil
  - Zohirul Sharif
  - Edward Liu

NVIDIA