Hi There

I'm Chris

About me:

EA SEED (2019-)

Improbable (2019)

EA Frostbite Physics (2010-2019)

Vaporwave is a product of the SEED Dynamic Worlds team, particularly these folks ------->

Chris Lewin

Will Donnelly

Henrik Halen

Marin Moran

Martin Mittring

Jon Greenberg

SEED

Vaporwave Why?

Want to represent high resolution volumetric effects in games and simulate all air near the player.

Standard fluid simulations are not really efficient enough to do this.

Standard volume rendering approaches are not well suited to this kind of content.

So we need to push forward in both directions.

Search for Extraordinary Experiences Division

Fluid Simulation Basics

Fluid Dynamics Sims

Incompressible Euler equations (constant density):

$$\frac{D\boldsymbol{u}}{Dt} = -\frac{1}{\rho_0}\nabla \mathrm{p} + \mathbf{g}$$

$$\nabla \cdot \boldsymbol{u} = 0$$

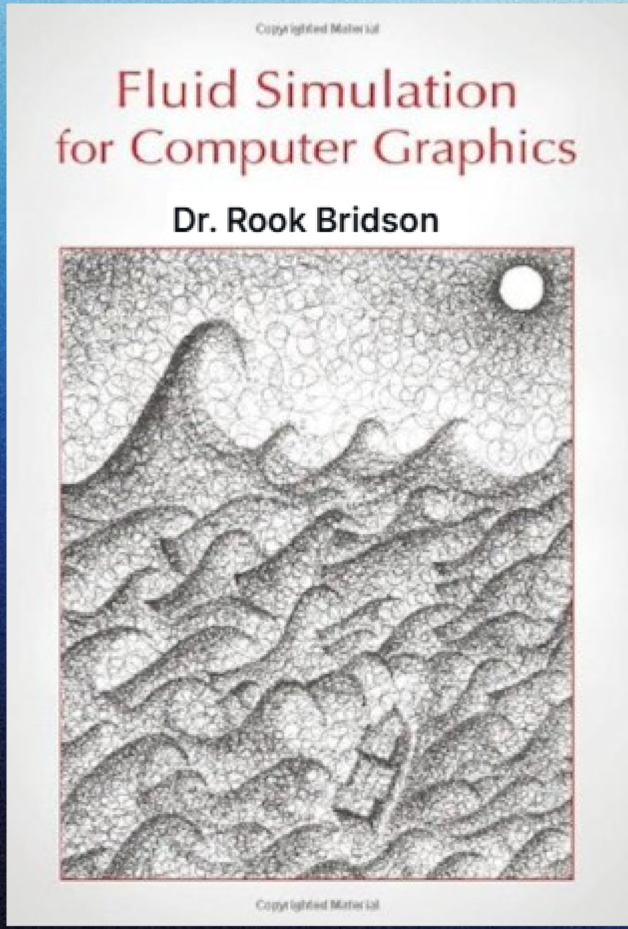
Usually simulate using **Stable Fluids** algorithm:

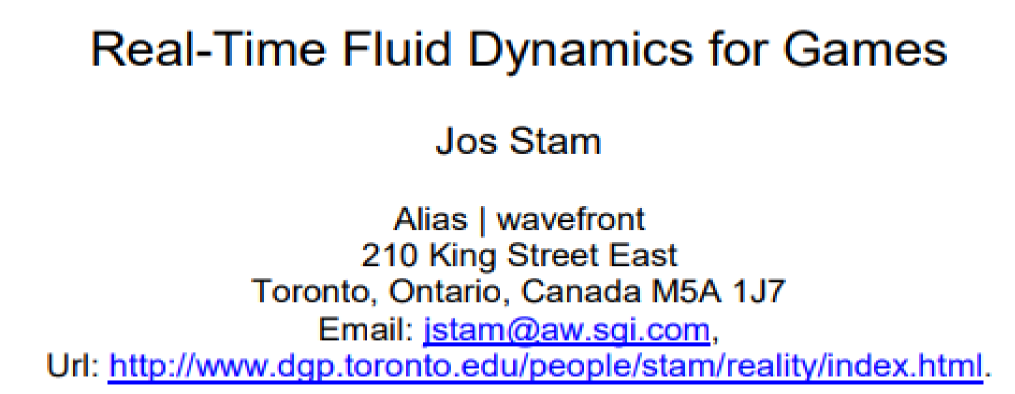
1. $\boldsymbol{u} \leftarrow project(\boldsymbol{u})$
2. $\boldsymbol{u} \leftarrow advect(\boldsymbol{u})$
3. $\boldsymbol{u} \leftarrow external\_forces(\boldsymbol{u})$

Simulating all the air is expensive so we try to use GPU and good algorithms.

[Sebastien Lague]

[Matthias Mueller]




Real-Time Fluid Dynamics for Games

Jos Stam

Alias | wavefront
210 King Street East
Toronto, Ontario, Canada M5A 1J7
Email: jstam@aw.sgi.com,
Url: http://www.dgp.toronto.edu/people/stam/reality/index.html.

Semi-Lagrangian Advection

Move the velocity field in the same way as resampling an image.

Conserves mass as long as the velocity field is divergence-free.

Get current velocity at each sample point and trace backwards

Sample velocity at arbitrary location and bring it back to the start position.

Semi-Lagrangian Advection

Move the velocity field in the same way as resampling an image.

Conserves mass as long as the velocity field is divergence-free.

Get current velocity at each sample point and trace backwards

Sample velocity at arbitrary location and bring it back to the start position.

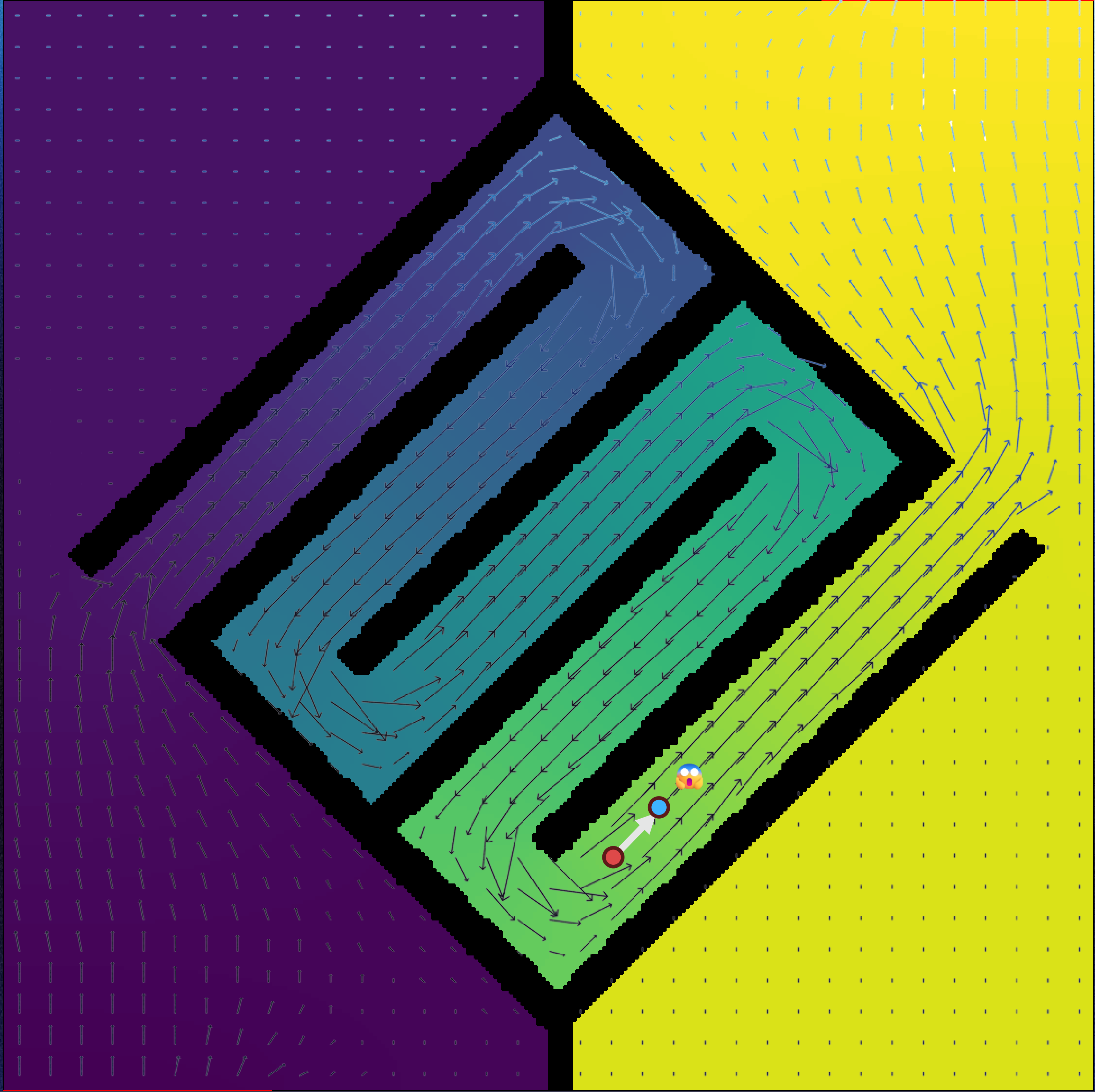
Semi-Lagrangian Advection

Move the velocity field in the same way as resampling an image.

Conserves mass as long as the velocity field is divergence-free.

Get current velocity at each sample point and trace backwards

Sample velocity at arbitrary location and bring it back to the start position.



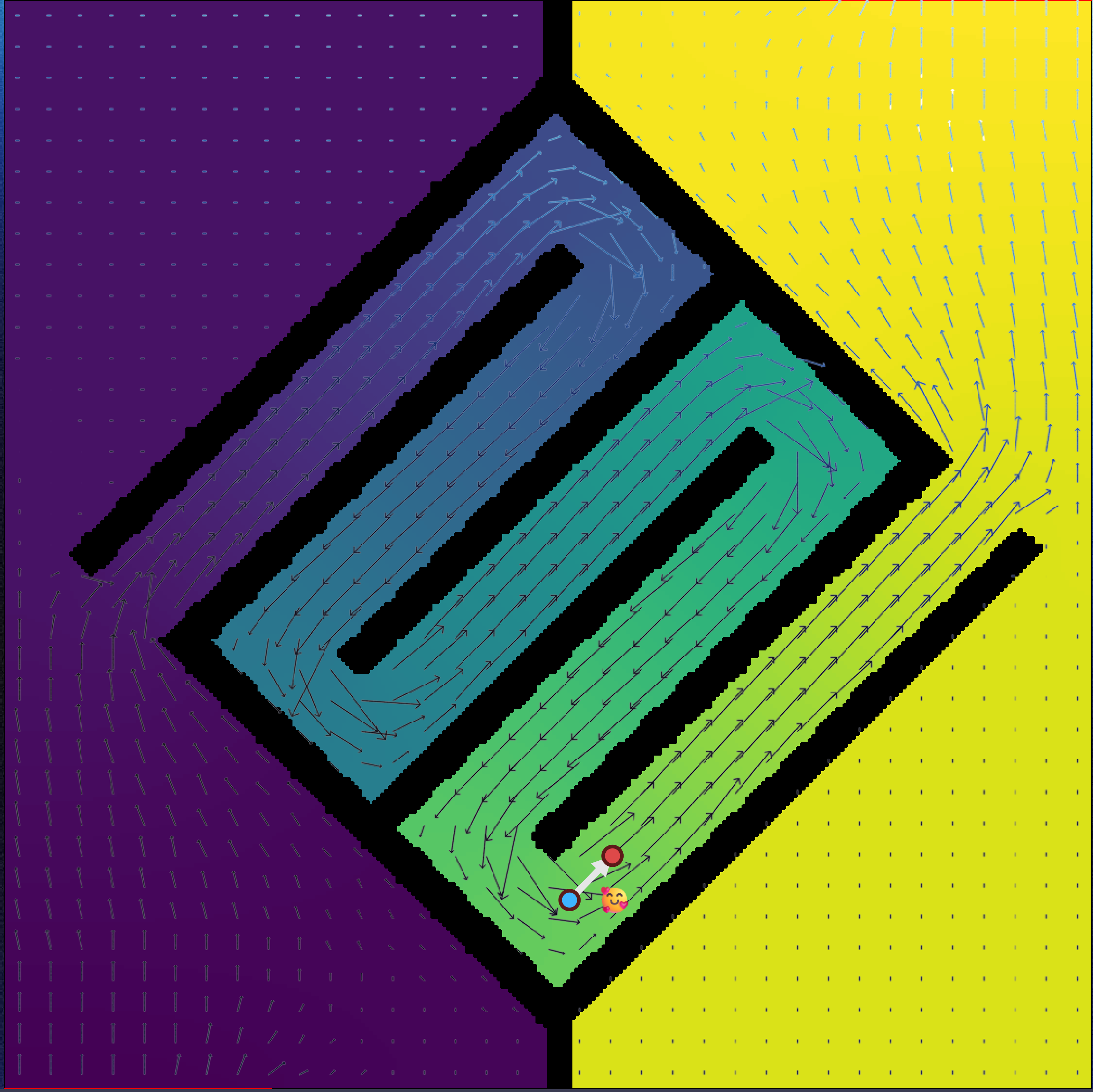
Semi-Lagrangian Advection

Move the velocity field in the same way as resampling an image.

Conserves mass as long as the velocity field is divergence-free.

Get current velocity at each sample point and trace backwards

Sample velocity at arbitrary location and bring it back to the start position.



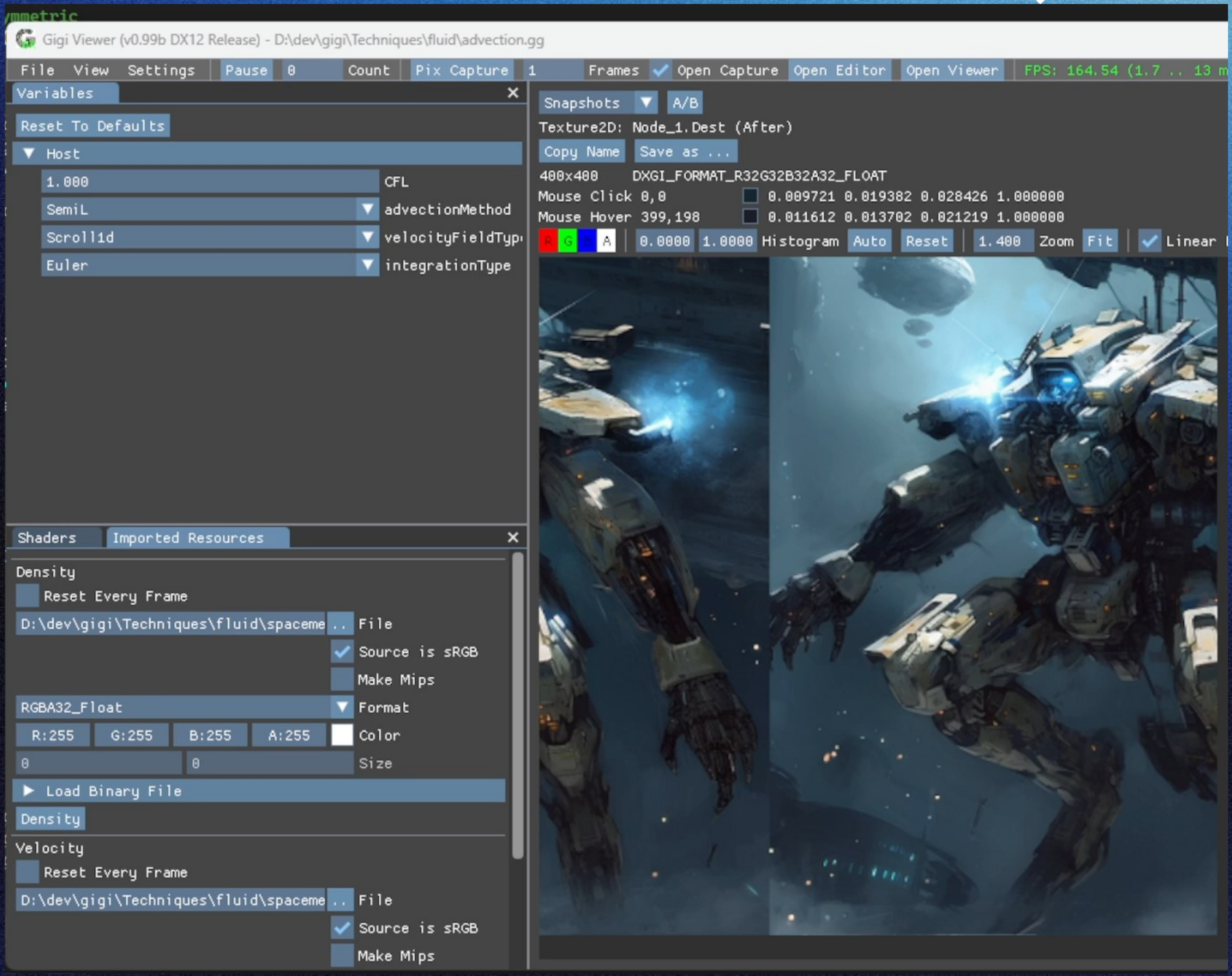
CFL Numbers

We can classify velocities by CFL number

CFL=1 means moving one grid cell per time step.

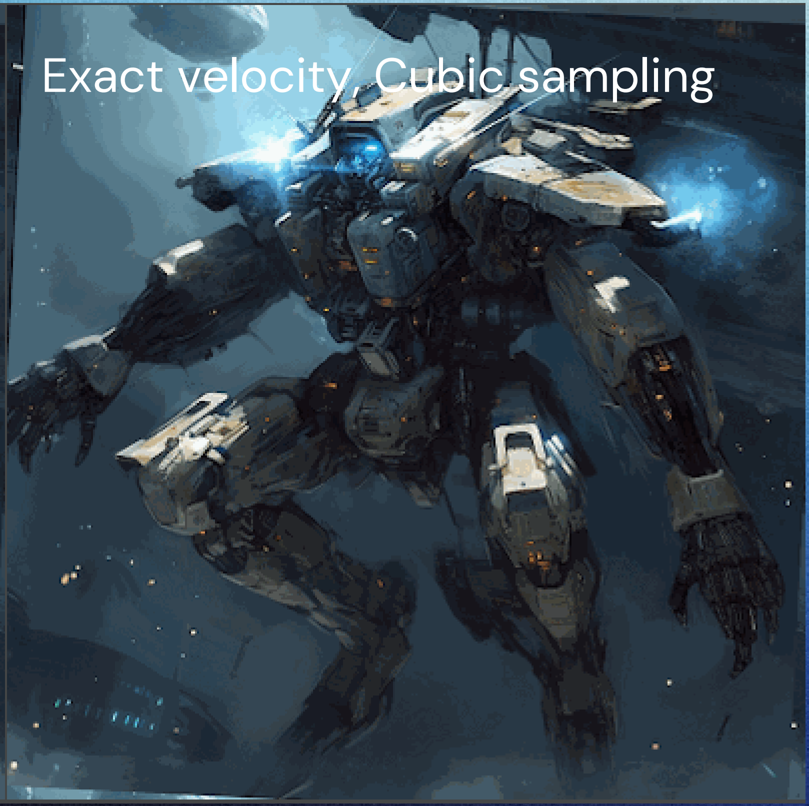
Axis-aligned flows of $CFL = n \in \mathbb{Z}$ are **exact**

All other flows accumulate resampling error



Advection Quality

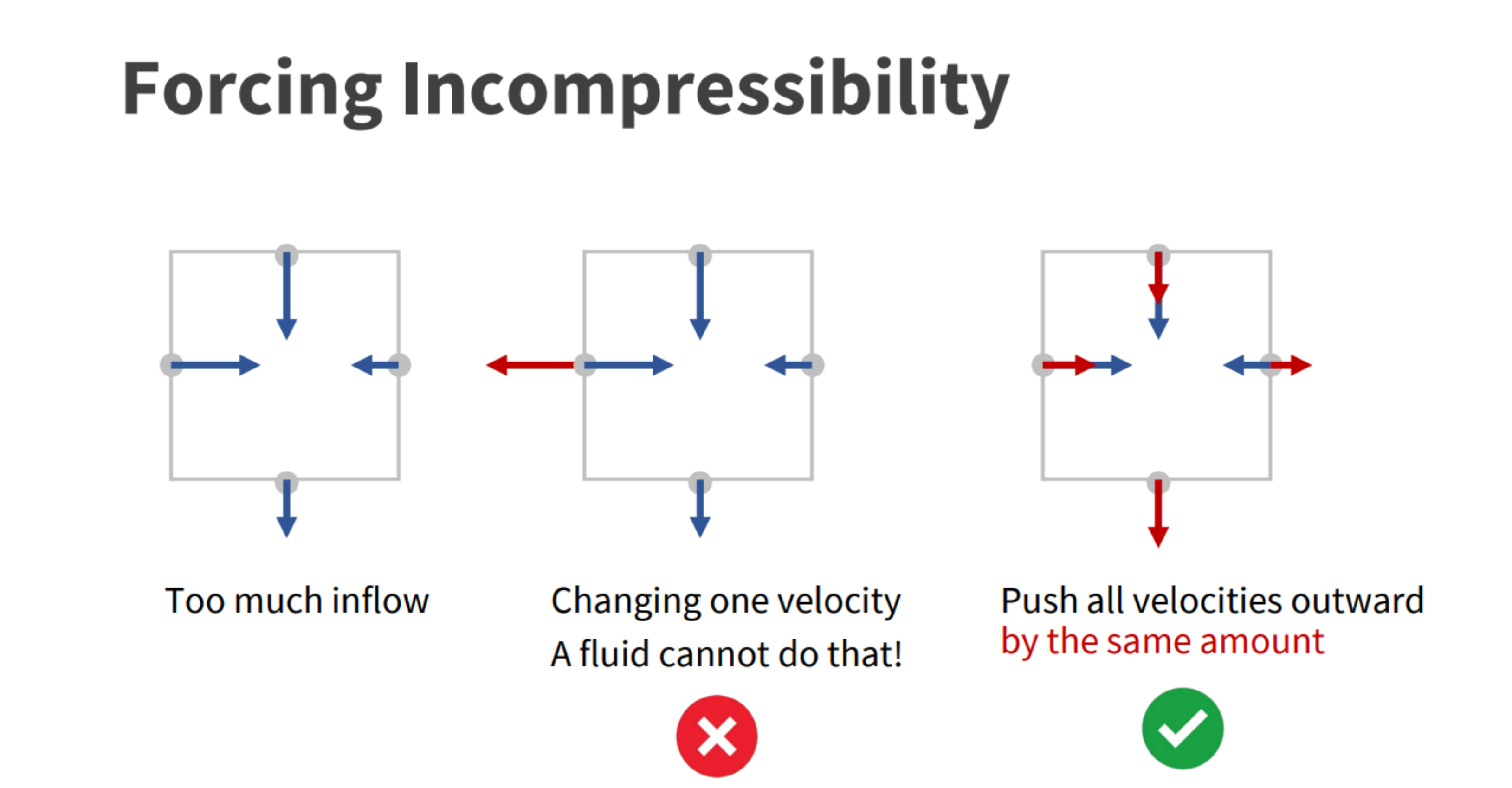
Use higher order sampling (e.g. tricubic) to reduce resampling error



Use better tracing (e.g. RK4) to better resolve curved trajectories in the flow

Projection

Makes the velocity field divergence free by generating hydrostatic forces.



[Matthias Mueller – 10 Minute Physics]

Projection

This can be done by solving a single large linear system of equations $A\boldsymbol{x} = b$

The matrix A operates on **x** like a convolution with a small kernel.

Solver choices:
Preconditioned Conjugate Gradient
Jacobi/Gauss-Seidel/SOR
Compact Poisson Filters
Fourier Transform
Multigrid

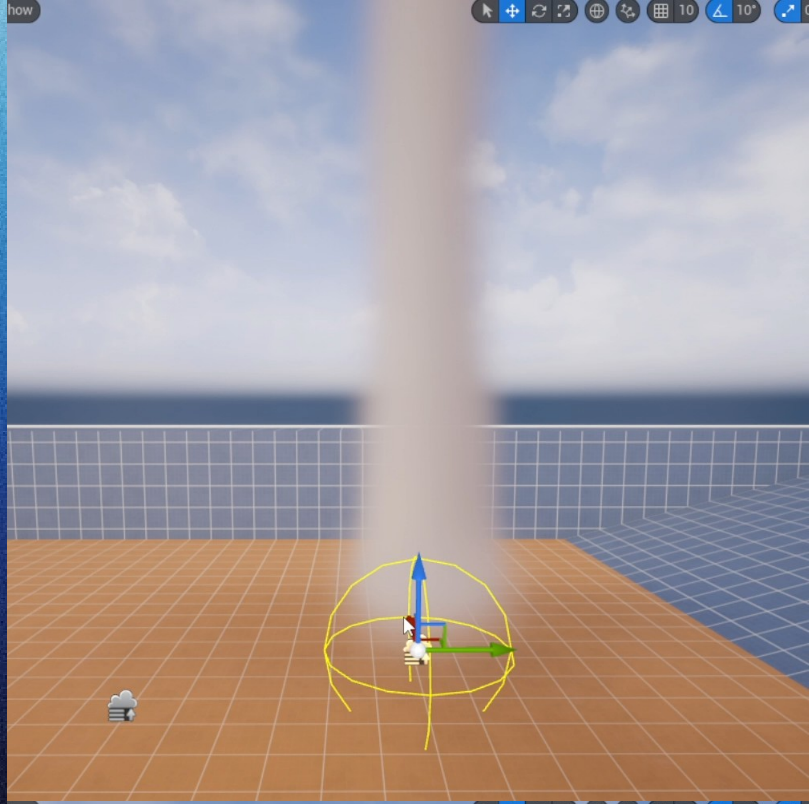
Fluid Sim Drawbacks

Large voxel count – n^3 in 3d

Need a good solver to get O(n^3) performance

Looks bad with low resolution

This limits range of the sim.



Alternative approaches

Low resolution invisible sim pushing opaques around – Returnal

Small visible sim located around a character - Returnal

Advection-diffusion sim – God of War

Our Simulation

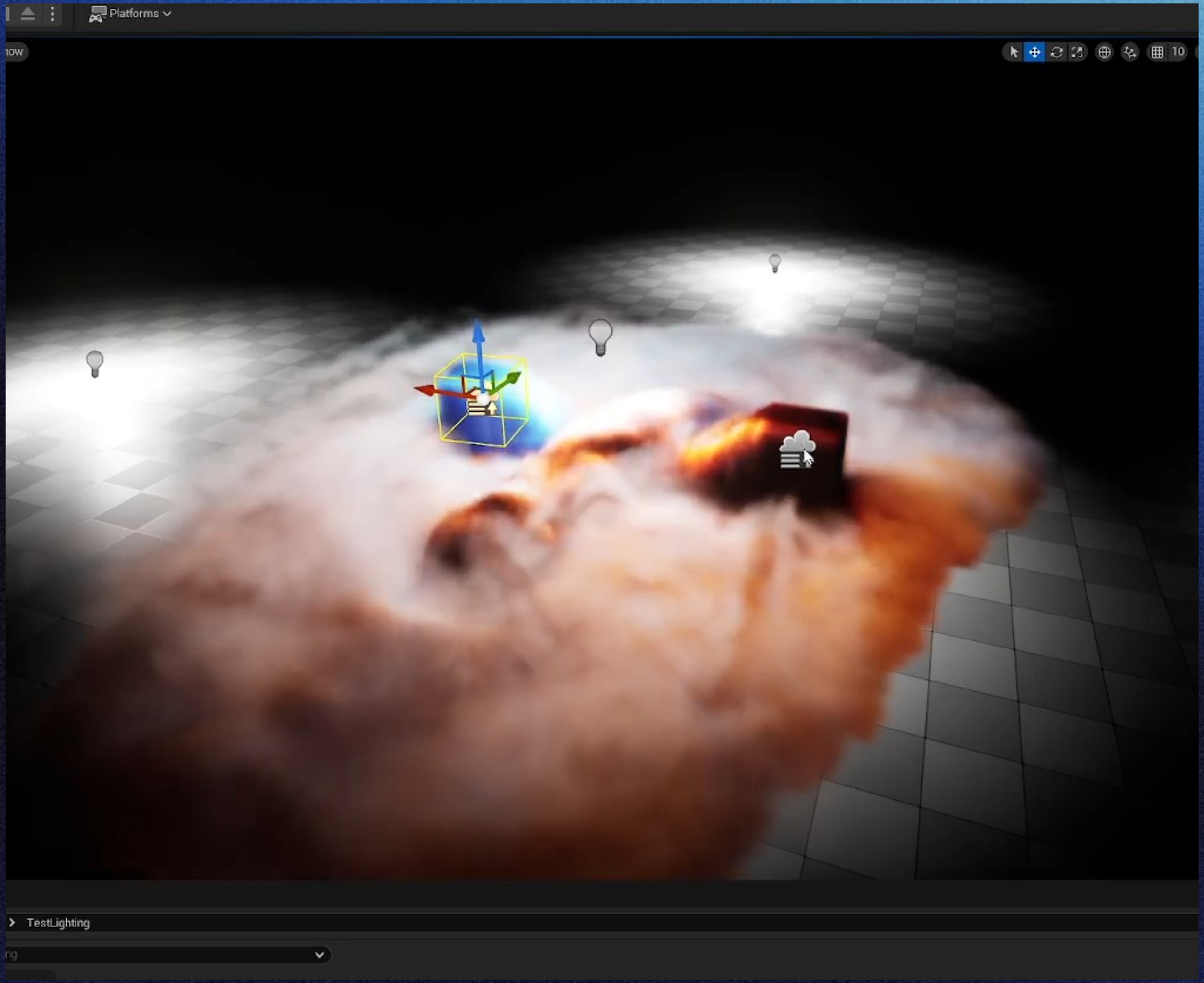
What we propose

Extend effective range of sim using **mixed resolution simulation**

Allows for sim and render lod

Locate simulation around the camera and regularly rebase

All fluid effects around a player can interact



Mixed Resolution Simulation

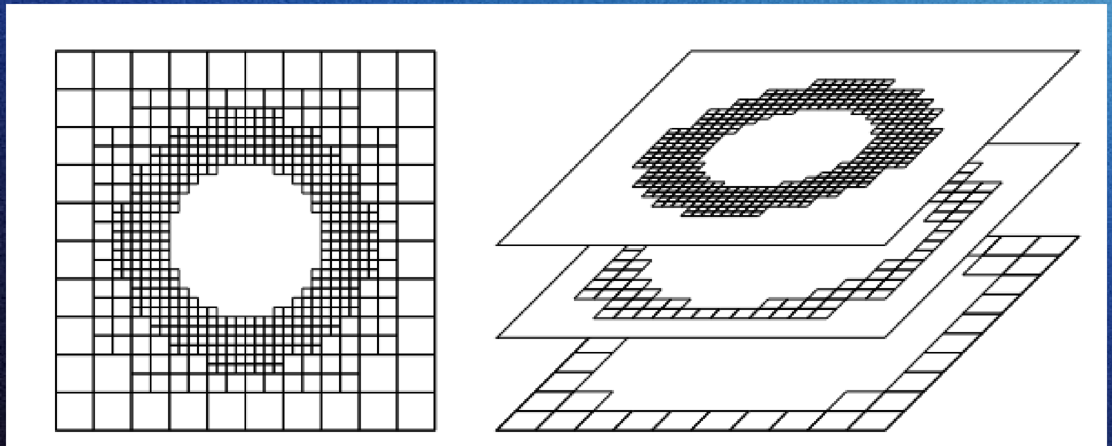
Multiple grid resolutions in one simulation.

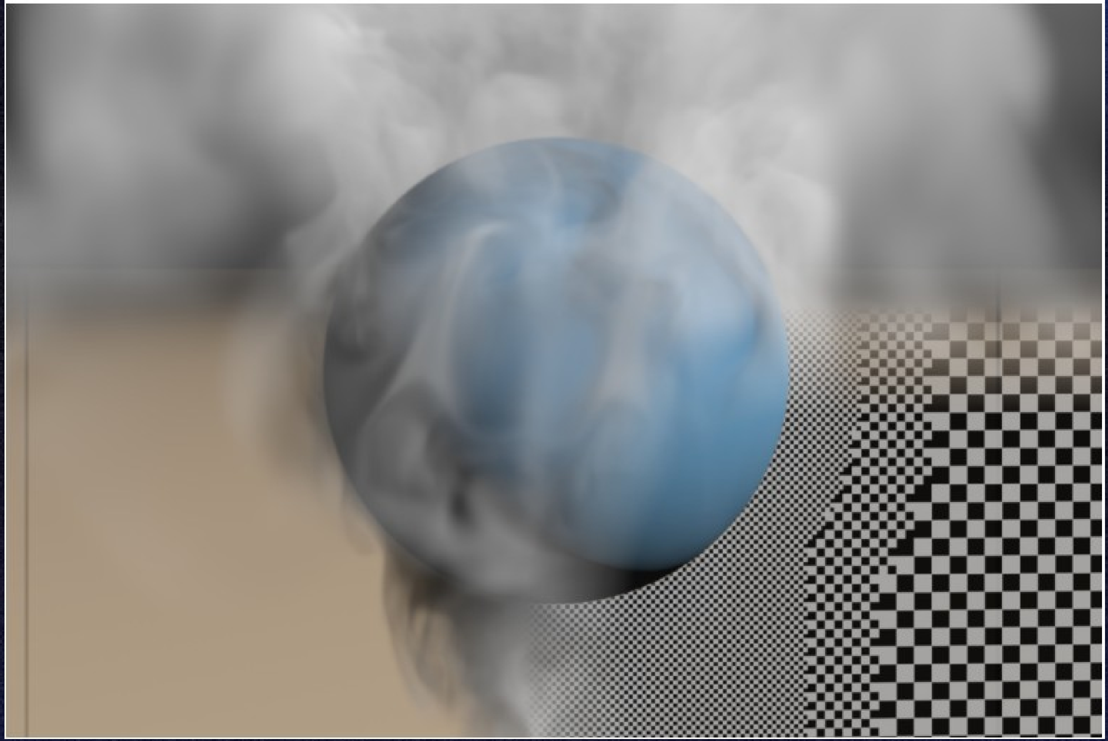
Not to be confused with Multigrid solver!!

Classically tries to concentrate resolution near fluid detail.

Difficult to get big performance boost this way.

We propose a much simpler alternative suited for real-time use.





Nested Grids

Advantages:

Constant-time lookup, no indirection

No adverse cache effects

No remeshing

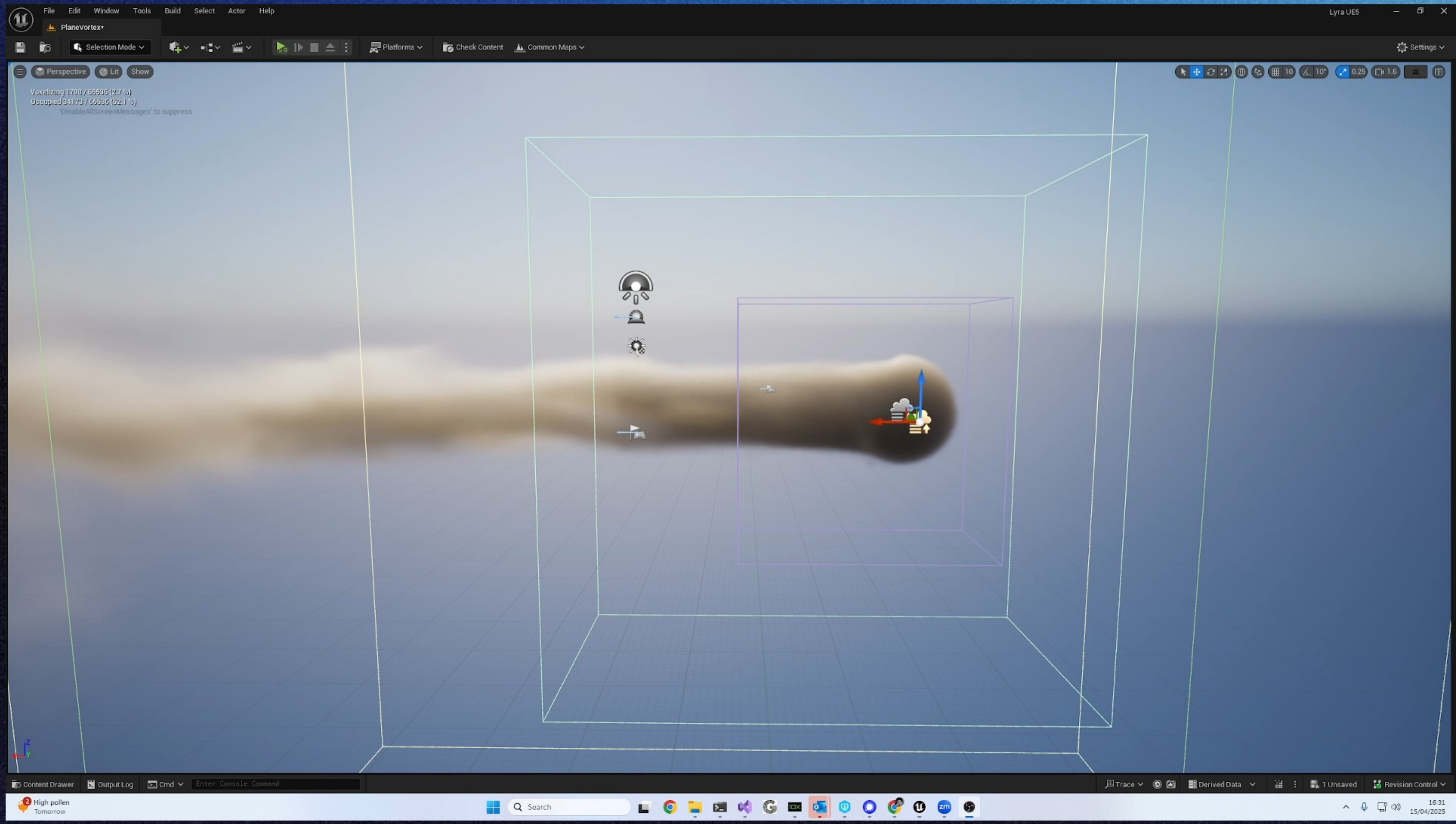
Mathematically more convenient

Worst-case performance is better

Disadvantages:

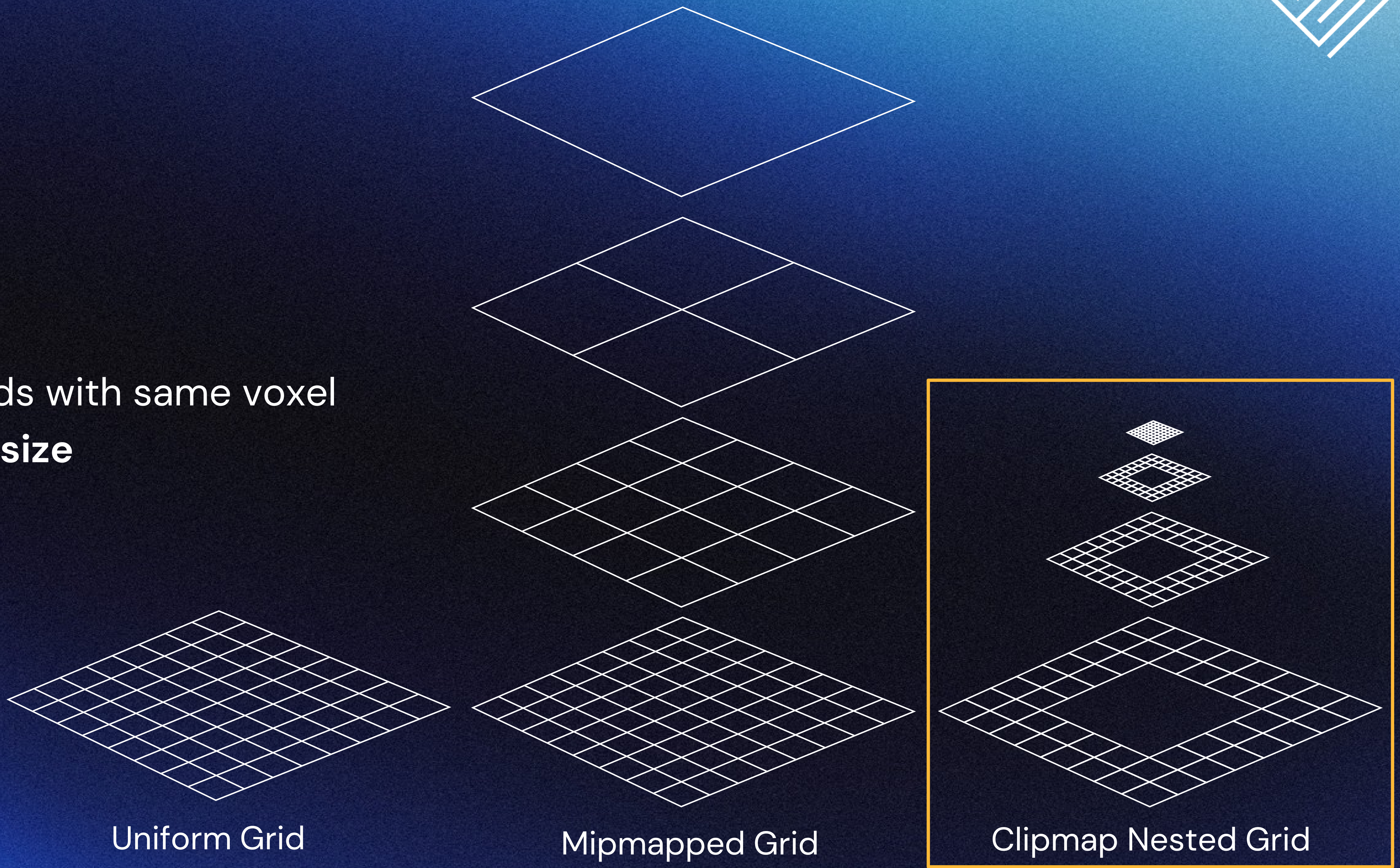
Can only have high detail in one location

Best-case performance is worse



Clipmap Nested Grids

Clipmaps!
= Sequence of grids with same voxel **count**, increasing **size**



Clipmap Nested Grids

Similar to Geometry Clipmaps

[Asirvatham and Hoppe, GPU Gems 2]

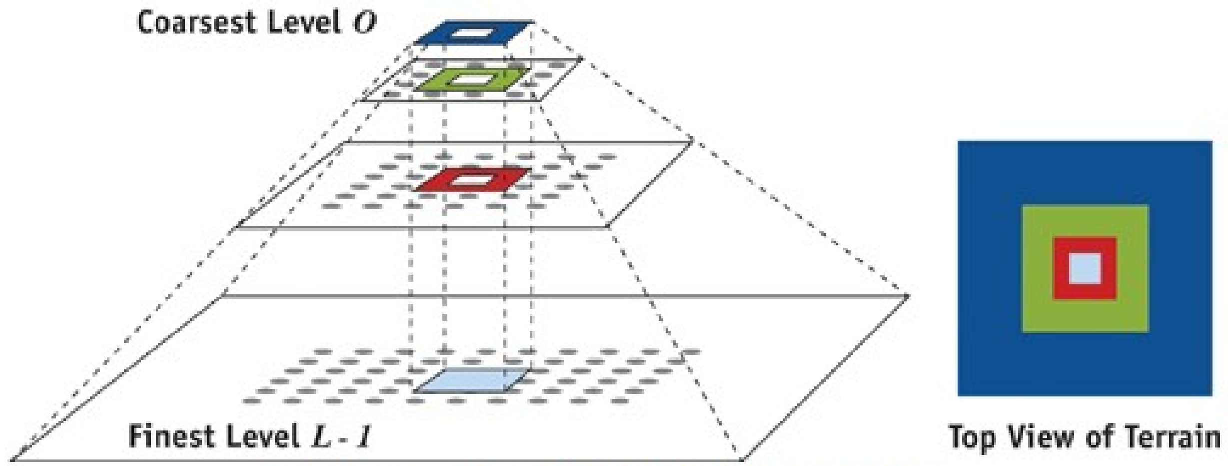


Figure 2-1 How Geometry Clipmaps Work

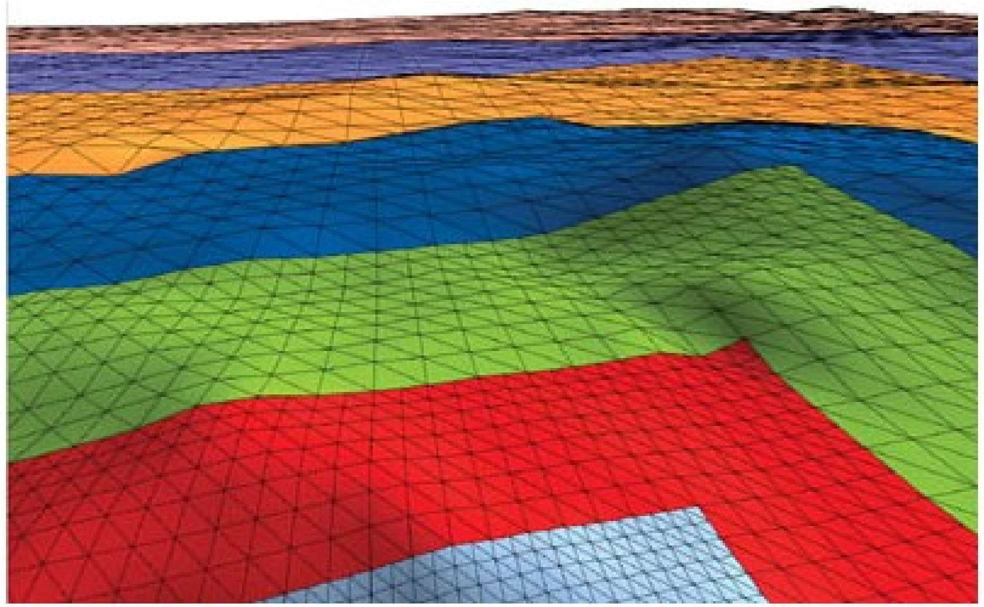


Figure 2-2 Terrain Rendering Using a Coarse Geometry Clipmap

Nested Grid Implementation

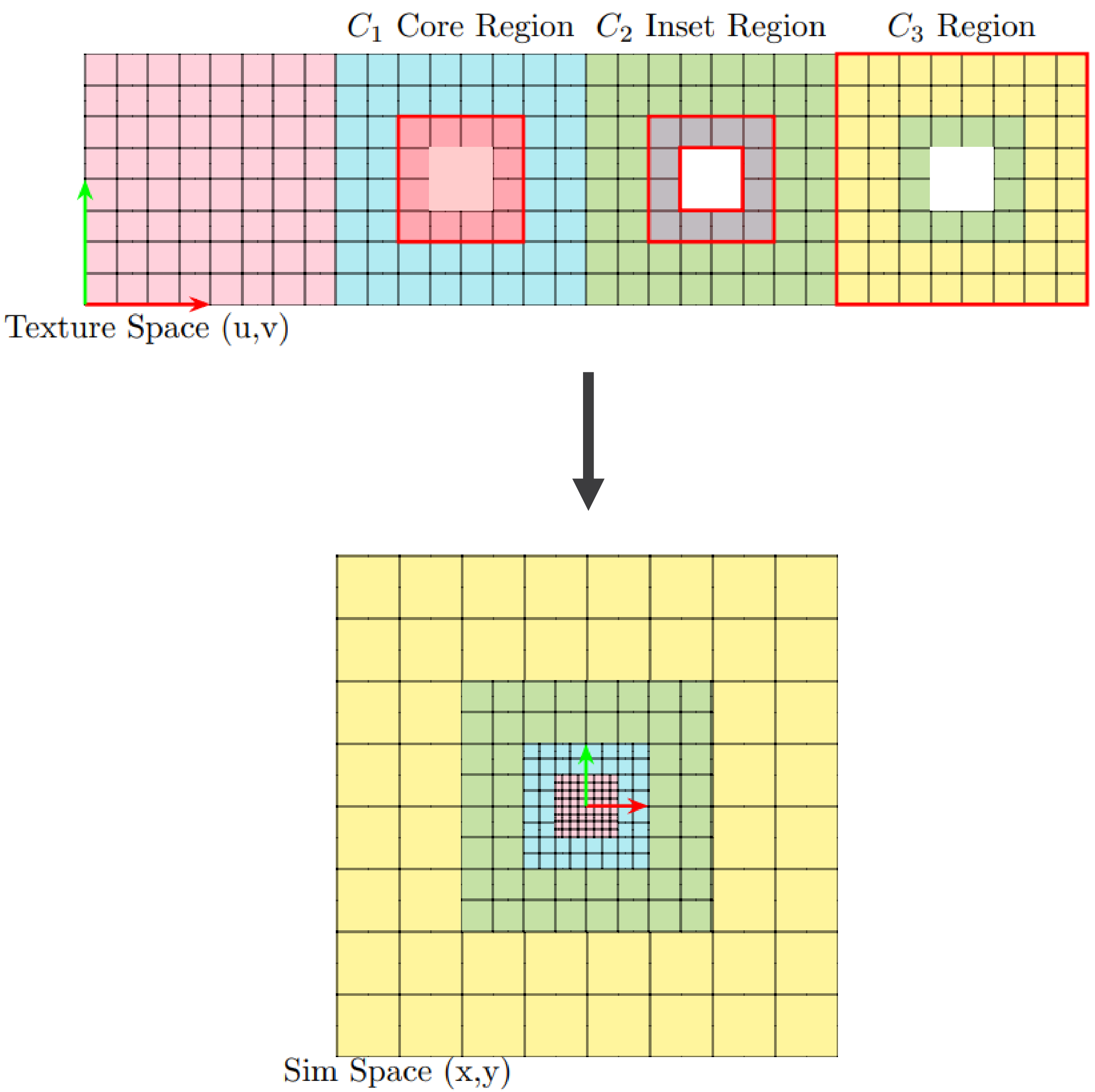
Want all data in one GPU resource.

Clipmaps all have same voxel count.

Voxels are same size in texture space but 2x larger at each level in sim space.

Core Region: covered by the previous clip. Occupies 1/8 volume in 3D.

Inset Region: n voxel wide halo used for sampling.



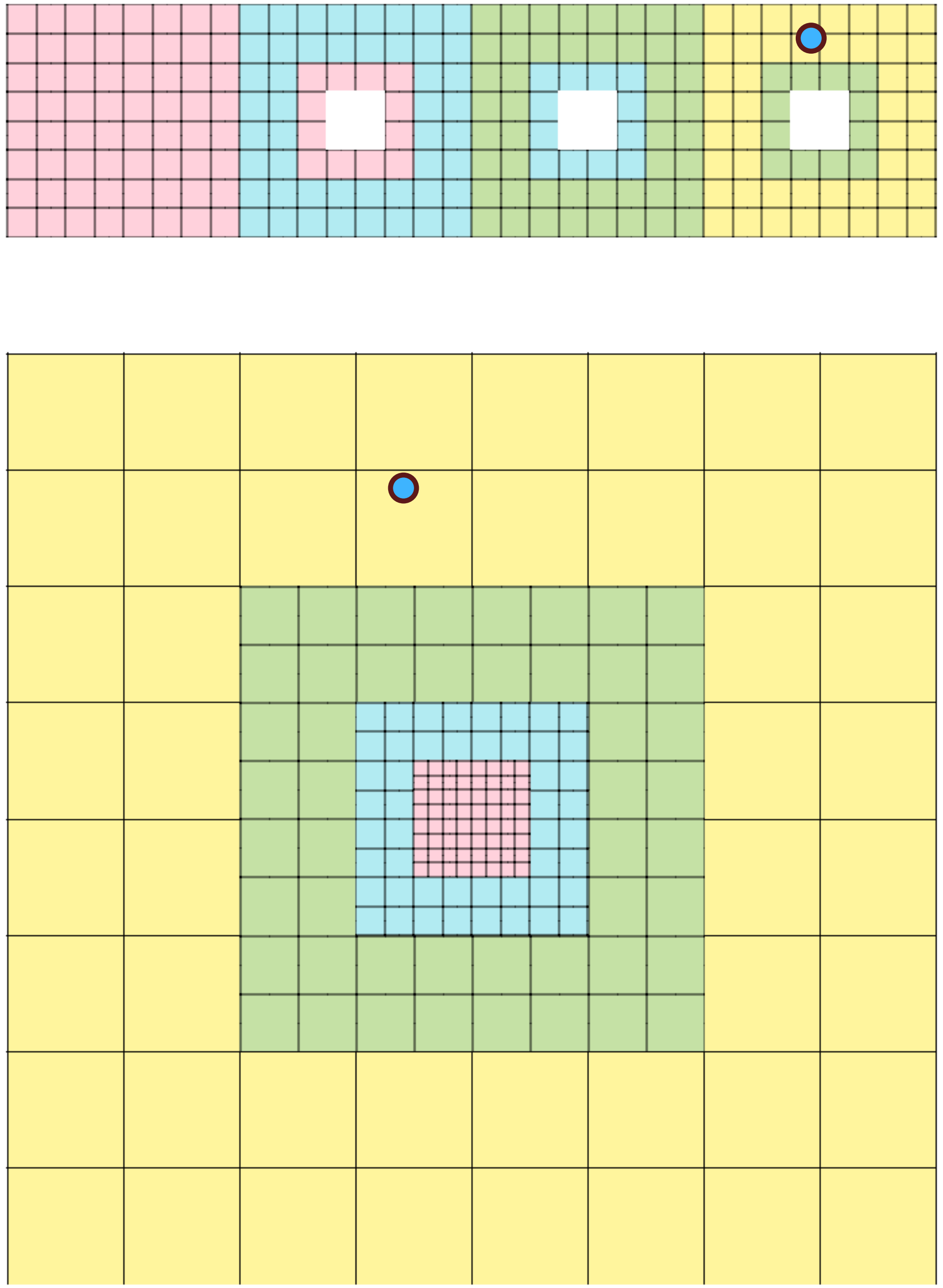
Advection

Just need to figure out how to sample anywhere in sim space.

Rays that hit in the interior of a clip are same as dense case.

Rays that hit near the **Core Region** can be handled using the inset values.

Rays that hit near the exterior of a clip are demoted to the next LOD.



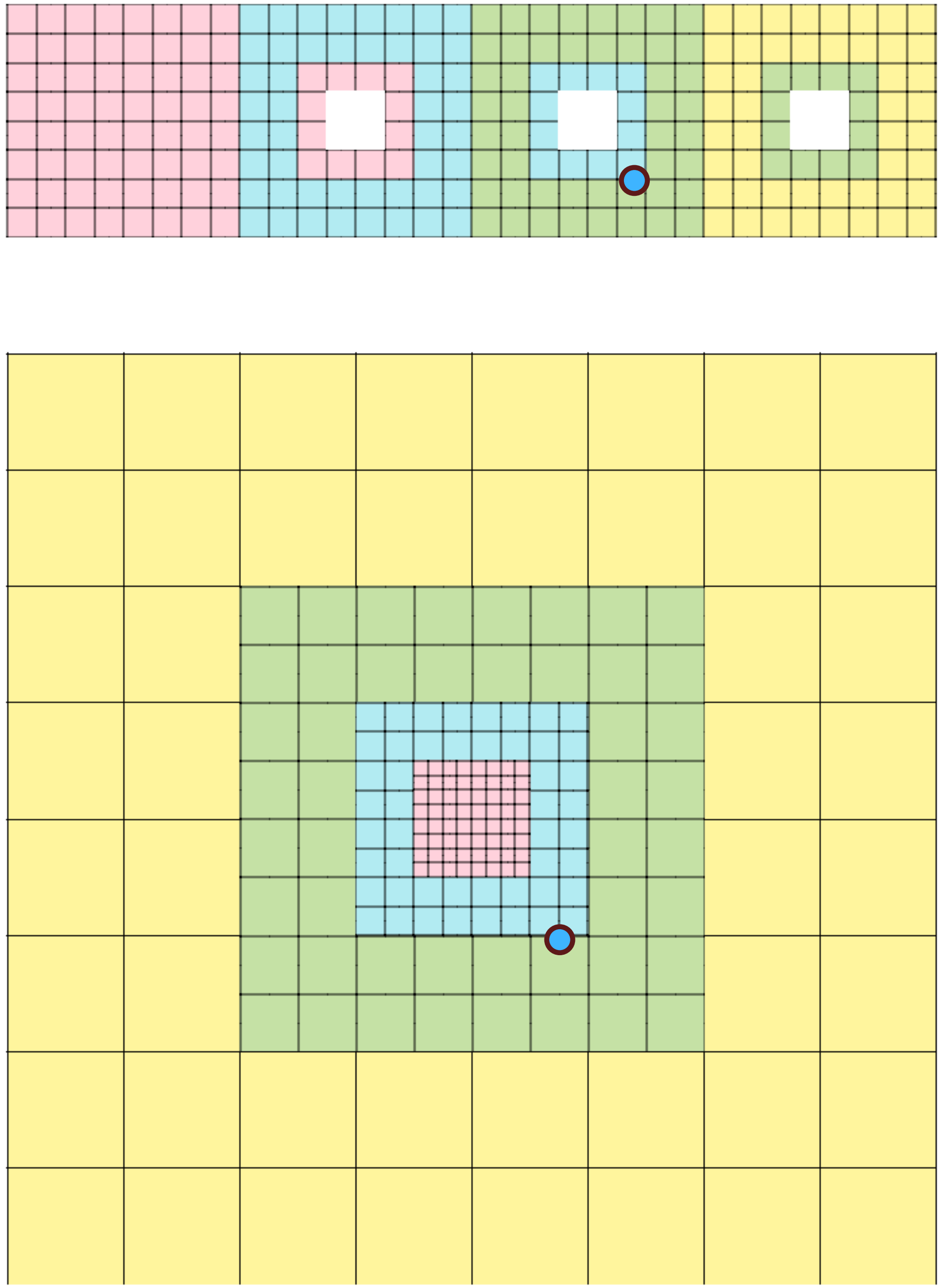
Advection

Just need to figure out how to sample anywhere in sim space.

Rays that hit in the interior of a clip are same as dense case.

Rays that hit near the **Core Region** can be handled using the inset values.

Rays that hit near the exterior of a clip are demoted to the next LOD.



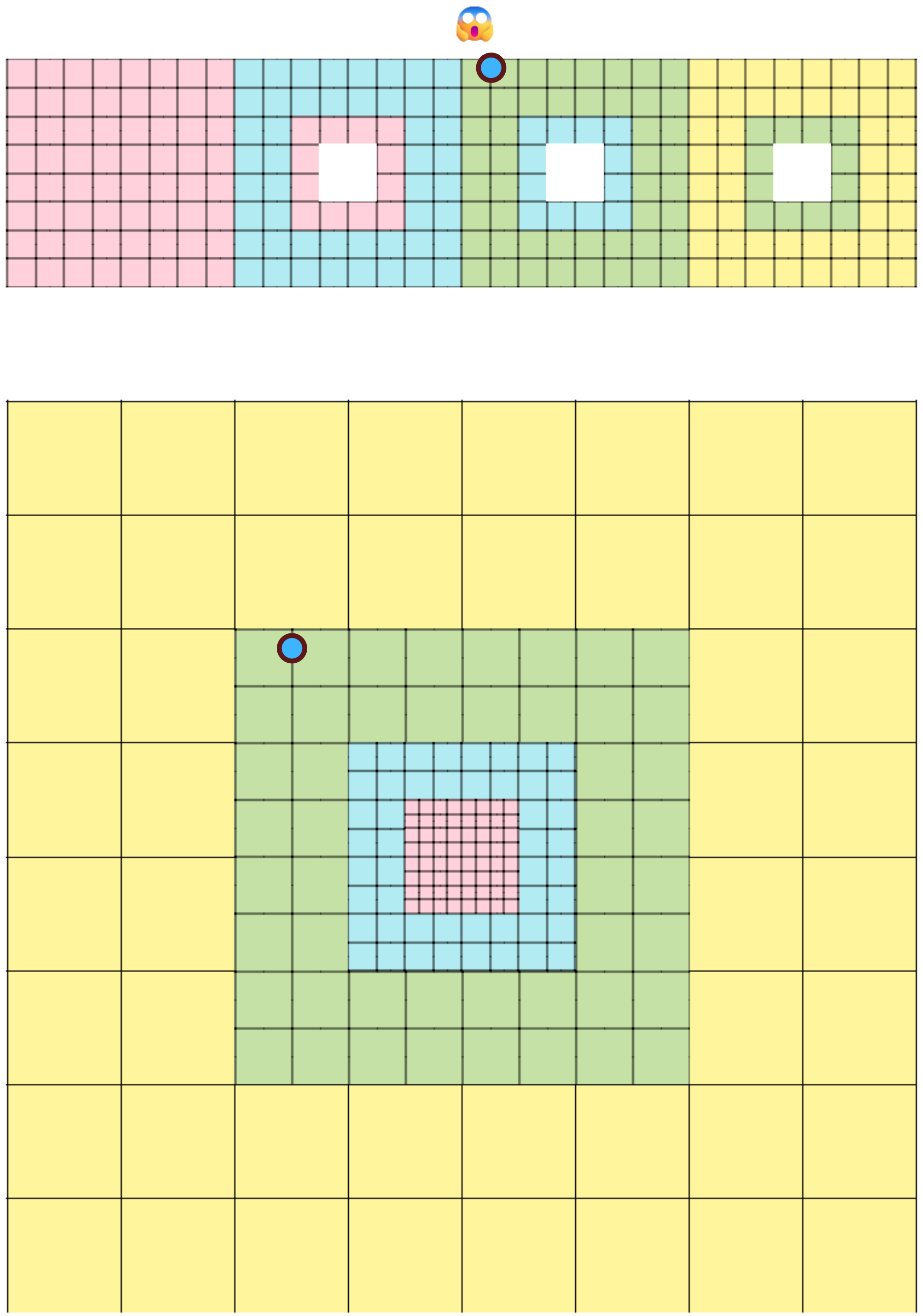
Advection

Just need to figure out how to sample anywhere in sim space.

Rays that hit in the interior of a clip are same as dense case.

Rays that hit near the **Core Region** can be handled using the inset values.

Rays that hit near the exterior of a clip are demoted to the next LOD.



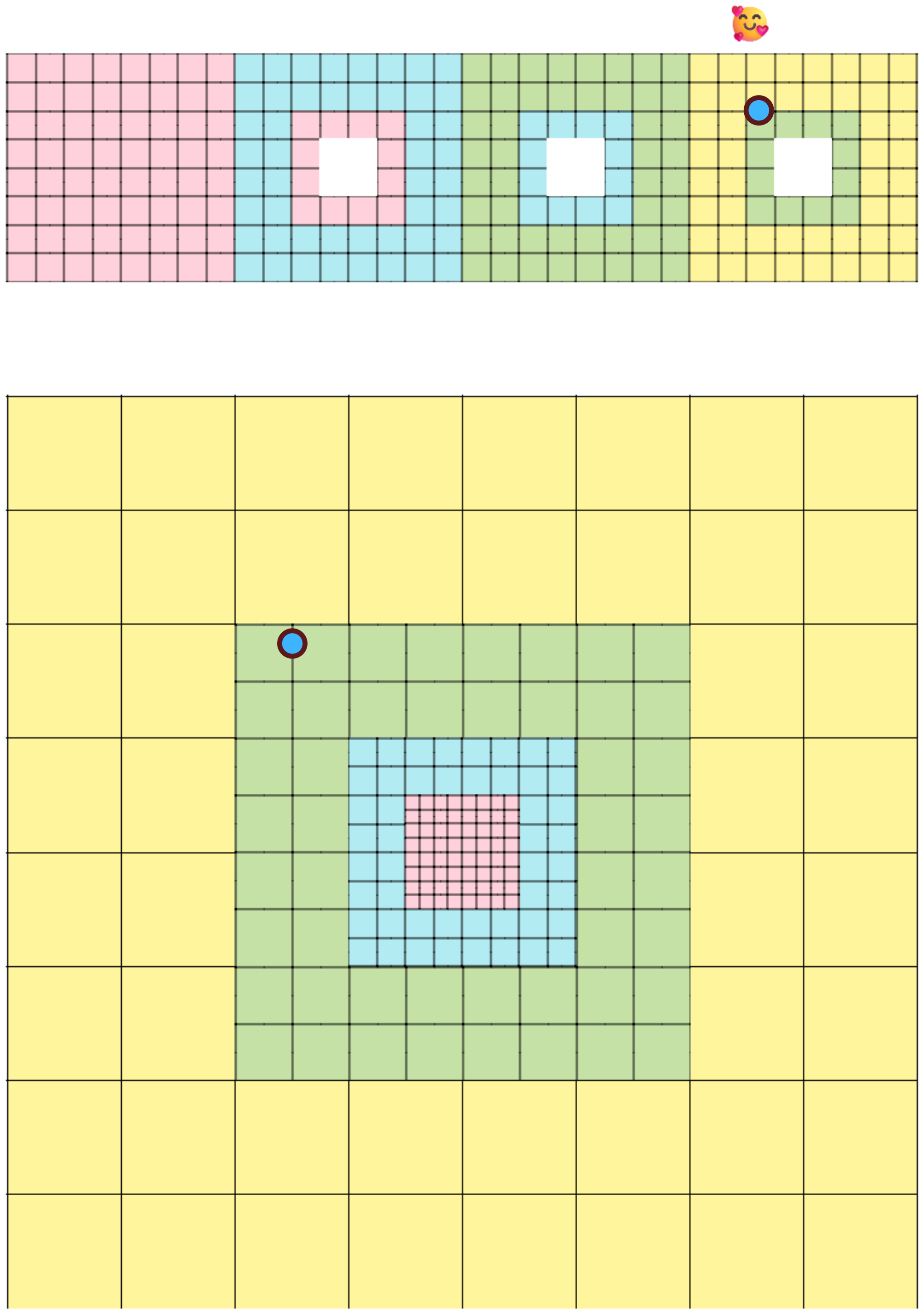
Advection

Just need to figure out how to sample anywhere in sim space.

Rays that hit in the interior of a clip are same as dense case.

Rays that hit near the **Core Region** can be handled using the inset values.

Rays that hit near the exterior of a clip are demoted to the next LOD.



Rebasing

Natural placement of grid is around camera

Have to move the grid without artefacts.

Naïve counter-advection causes smearing.

Use integer CFL trick!

Exact within one clip but pops at the boundary.

Naïve counter-advection

Snapped to integer CFL

Projection

Solving projection is analogous to applying a very wide convolution

Very slow when only using neighbours!

Can apply known popular techniques:

- Fourier Transform
- Separable blur -> Compact Poisson Filters
- **Multi-resolution blur -> Multigrid**

Jacobi iterative solver

Projection

Solving projection is analogous to applying a very wide convolution.

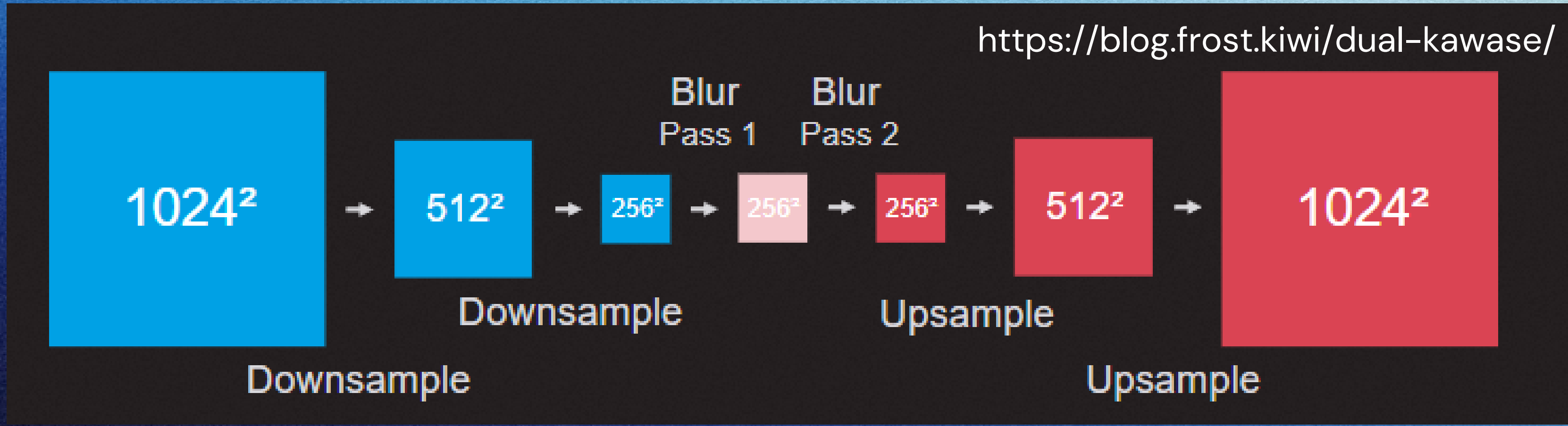
Very slow when only using neighbours!

Can apply known popular techniques:

Fourier Transform

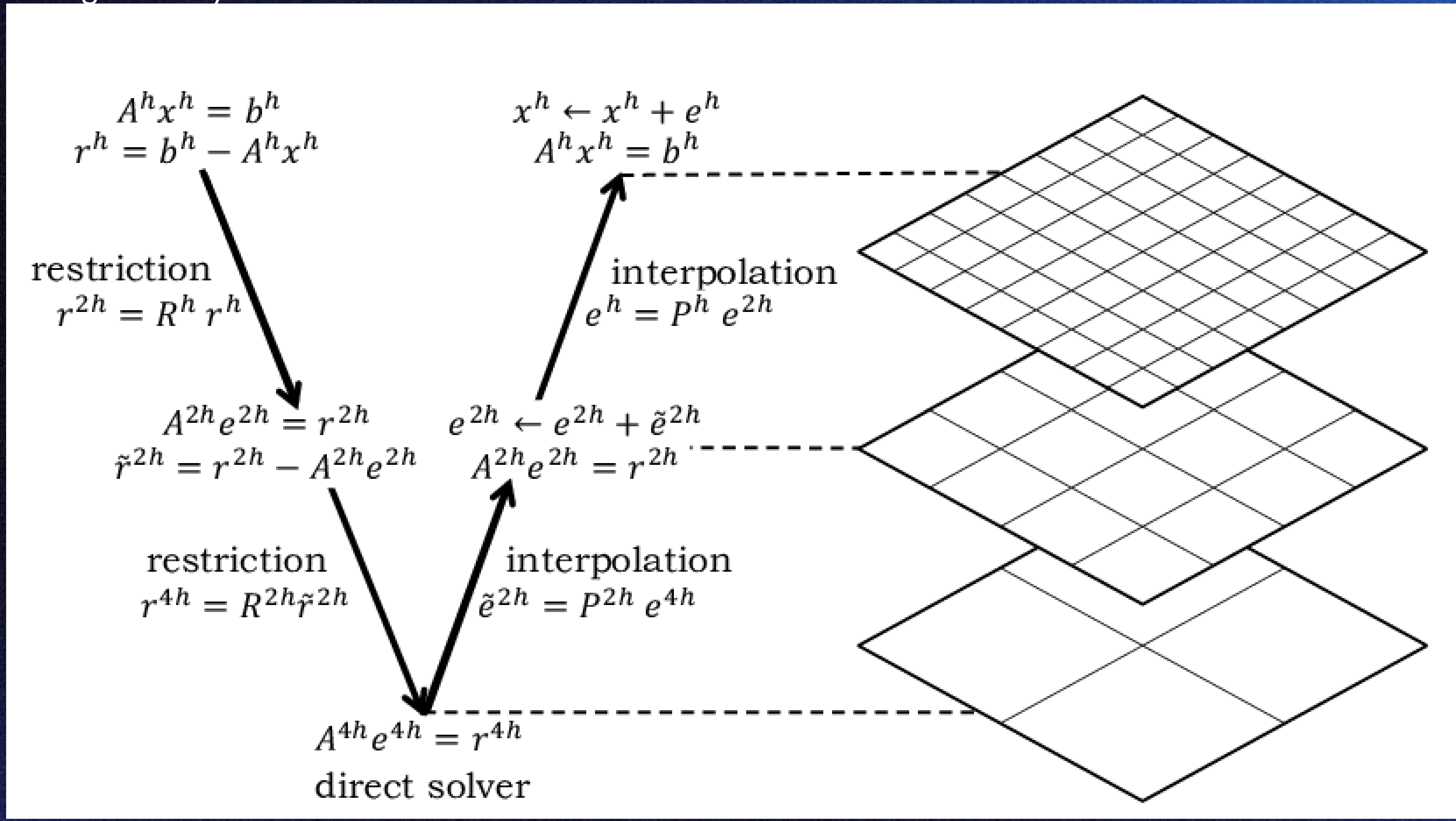
Separable blur -> Compact Poisson Filters

Multi-resolution blur -> Multigrid



Multi-resolution blur

Multigrid V-Cycle



Clipmap Multigrid Projection

Solving Poisson's Equation using Adaptive Mesh Refinement

D. F. Martin* and K. L. Cartwright†

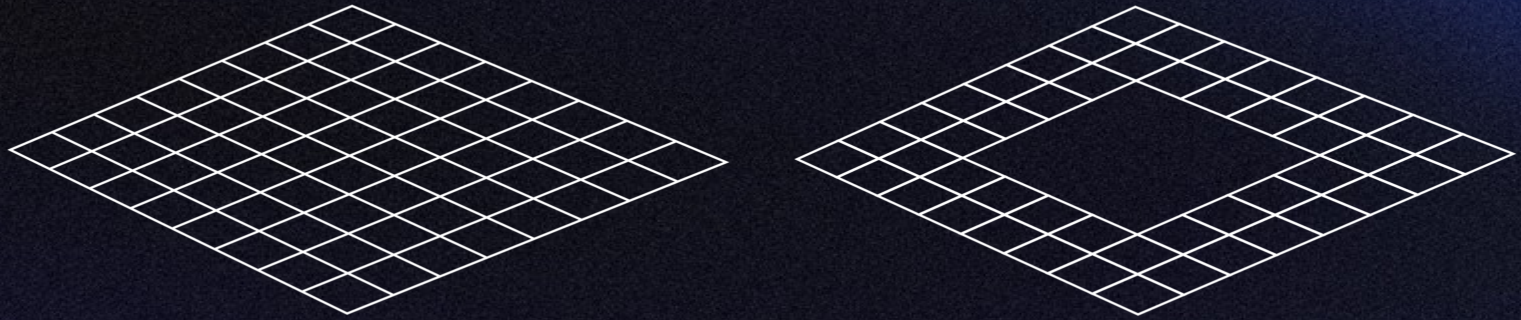
October 24, 1996

Based on [Martin and Cartwright 96]

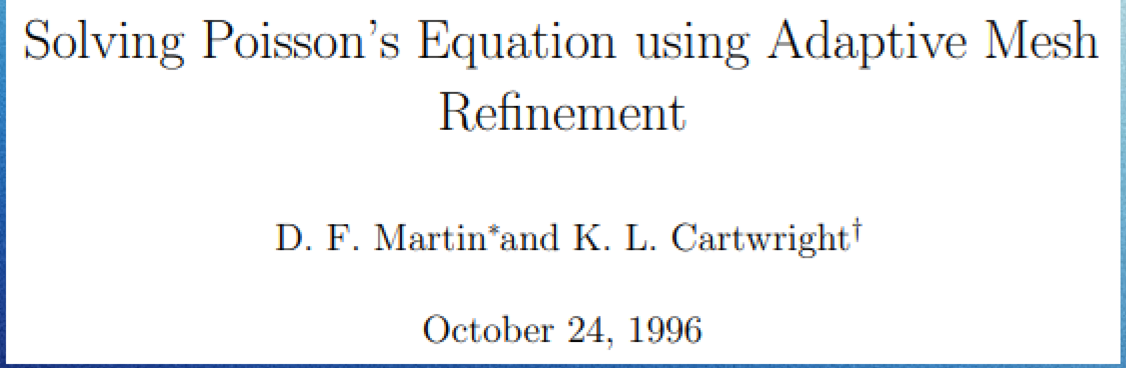
Replaces the mip sequence with a clip sequence

Downsample clip into core region of next

1. Solve on k



Clipmap Multigrid Projection



Solving Poisson's Equation using Adaptive Mesh Refinement

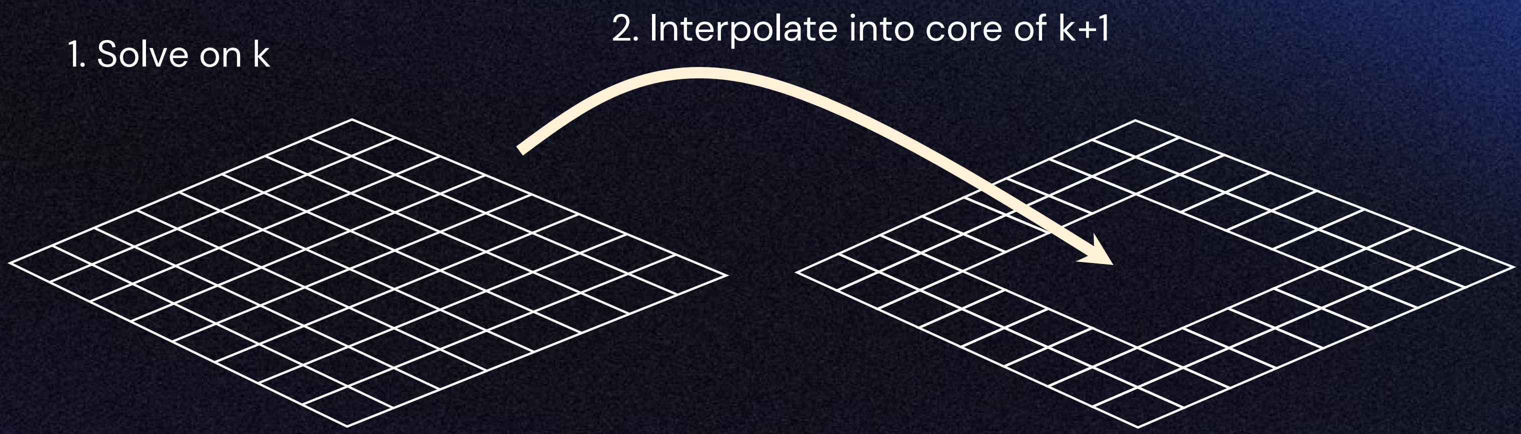
D. F. Martin* and K. L. Cartwright†

October 24, 1996

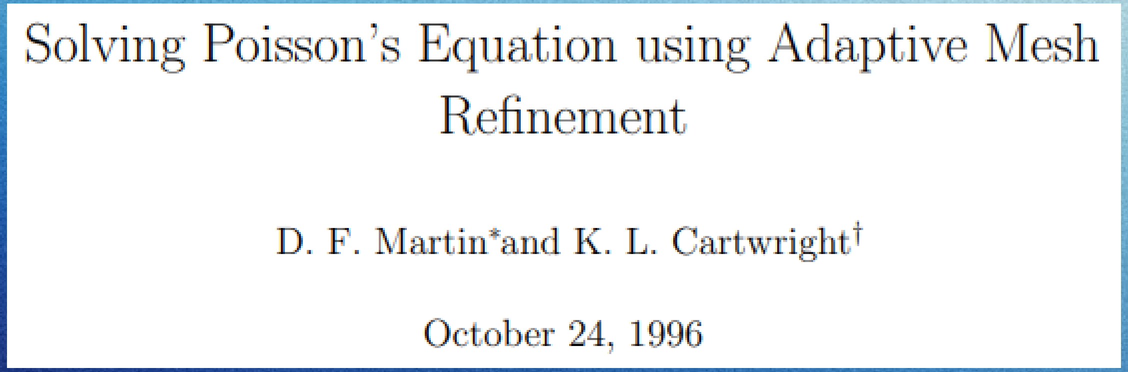
Based on [Martin and Cartwright 96]

Replaces the mip sequence with a clip sequence

Downsample clip into core region of next



Clipmap Multigrid Projection


Solving Poisson's Equation using Adaptive Mesh Refinement

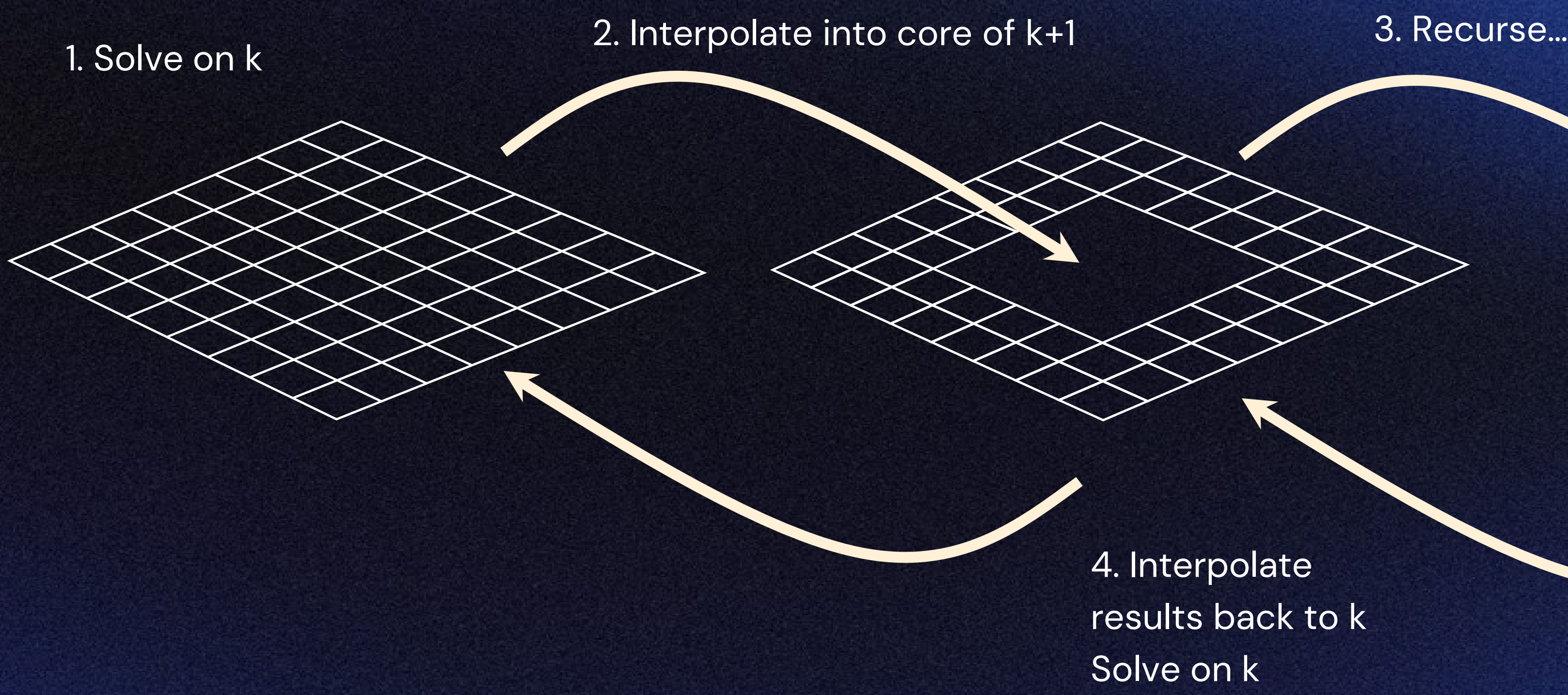
D. F. Martin*and K. L. Cartwright†

October 24, 1996

Based on [Martin and Cartwright 96]

Replaces the mip sequence with a clip sequence

Downsample clip into core region of next



Clipmap Multigrid Projection

Solving Poisson's Equation using Adaptive Mesh Refinement

D. F. Martin* and K. L. Cartwright†

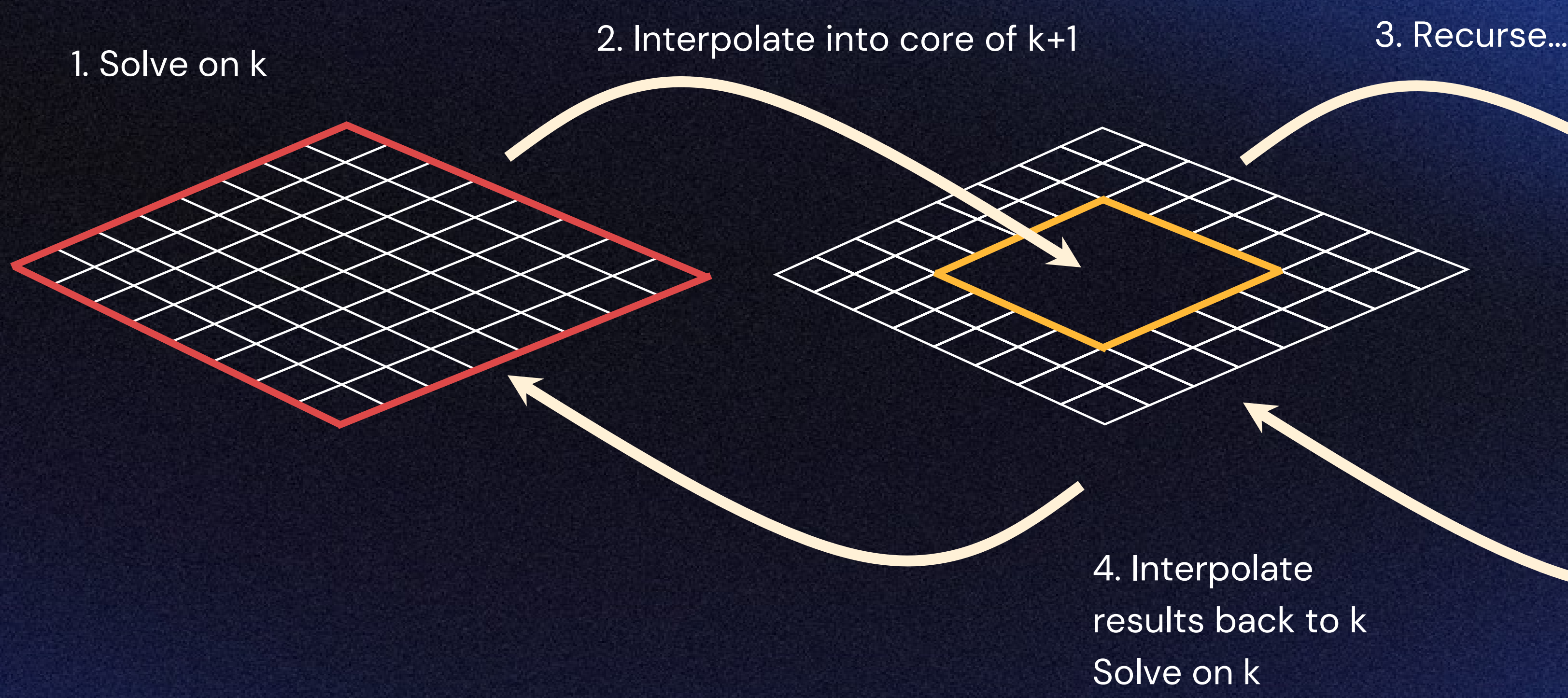
October 24, 1996

Based on [Martin and Cartwright 96]

Replaces the mip sequence with a clip sequence

Downsample clip into core region of next

Paper shows how to handle outer boundary

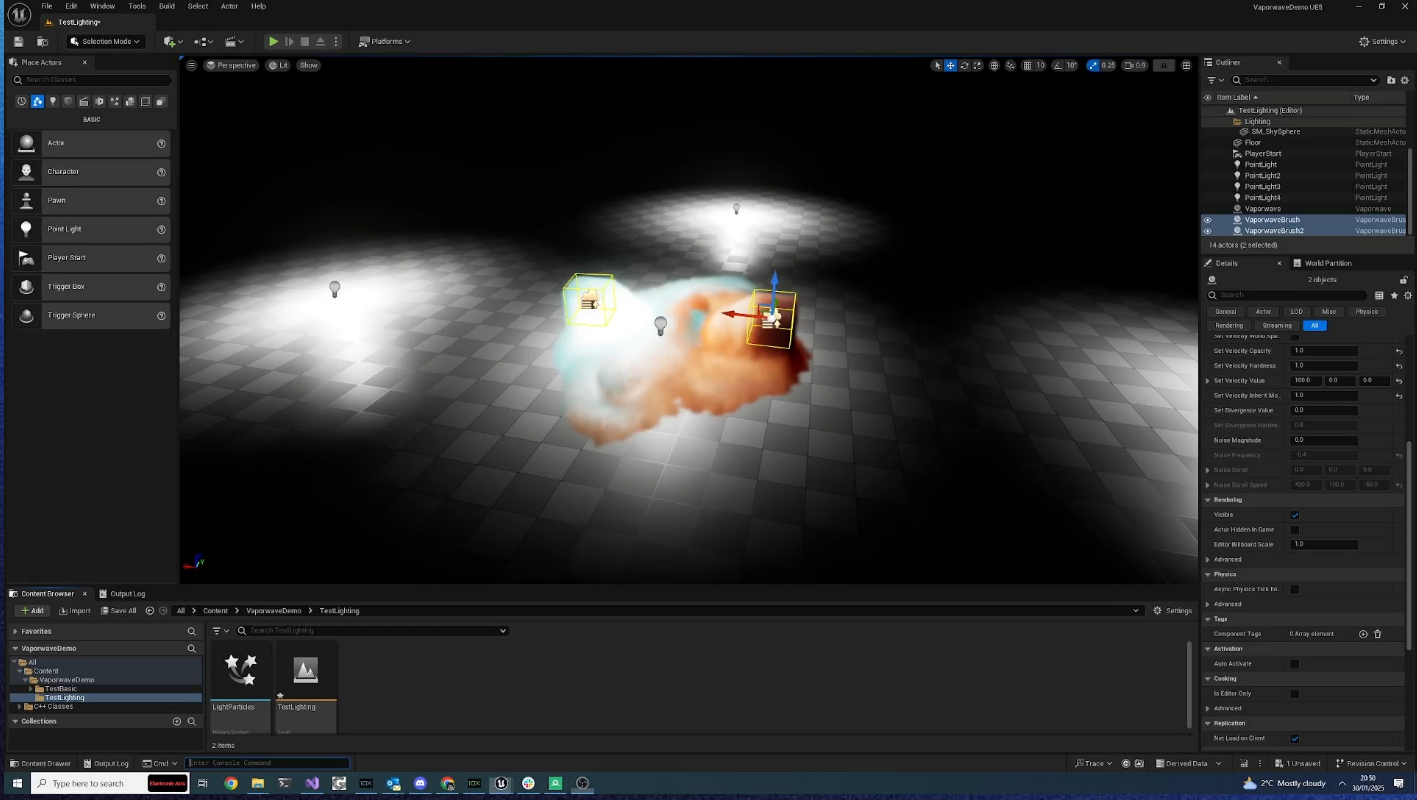


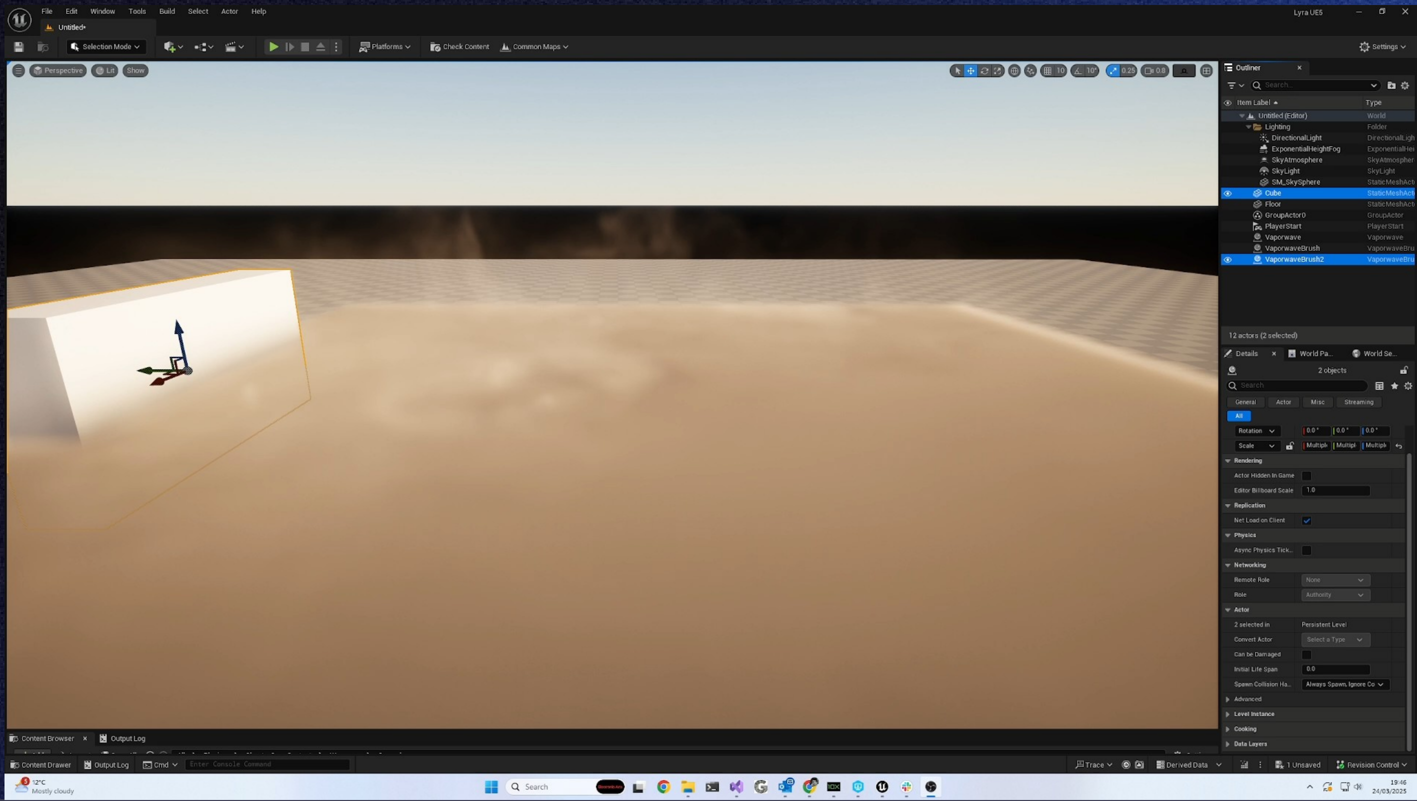
Control

Add material, velocity, collision using 3d Brushes.

Use UE global distance field to get world collision.

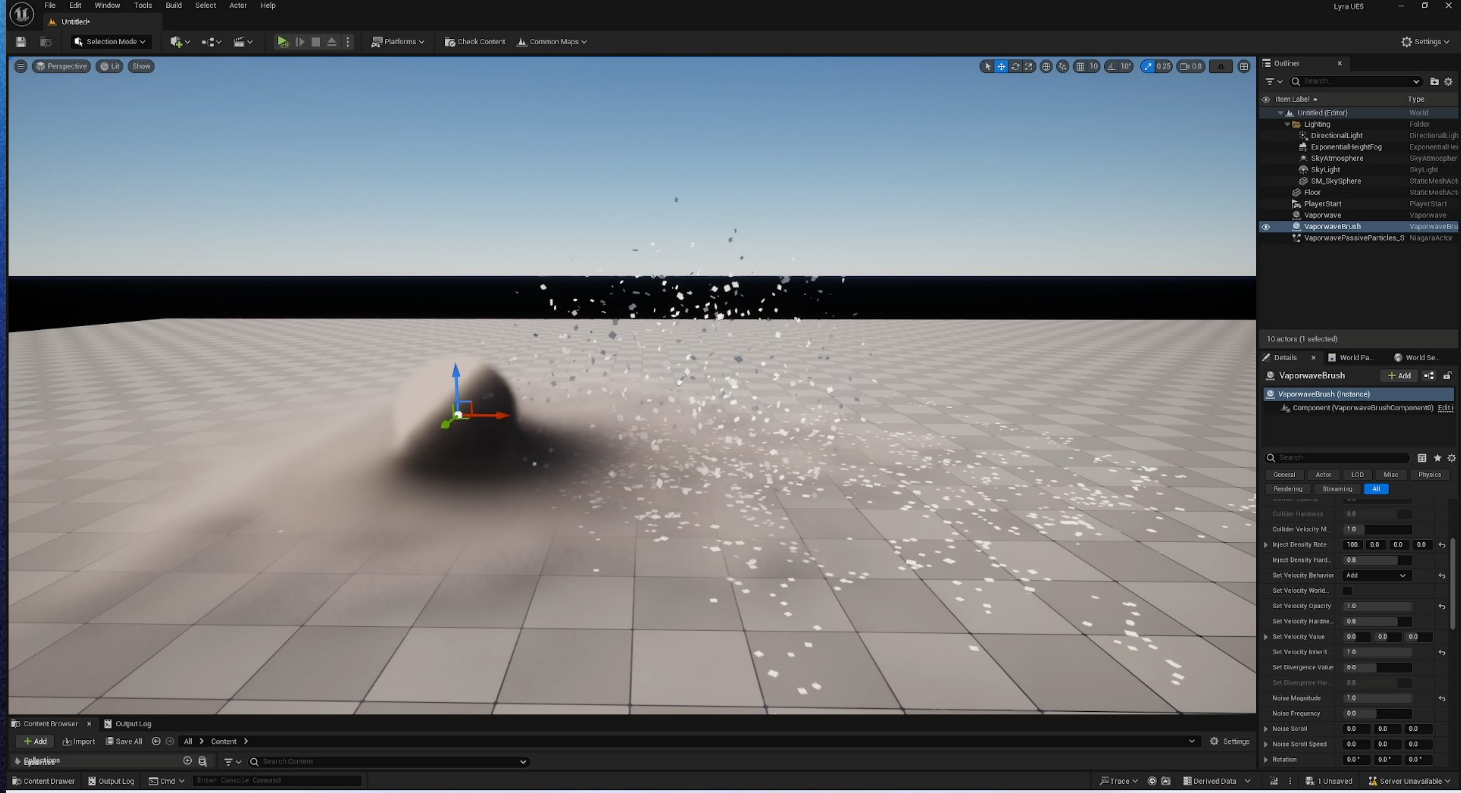
One velocity field and four visual-physical materials allowed.

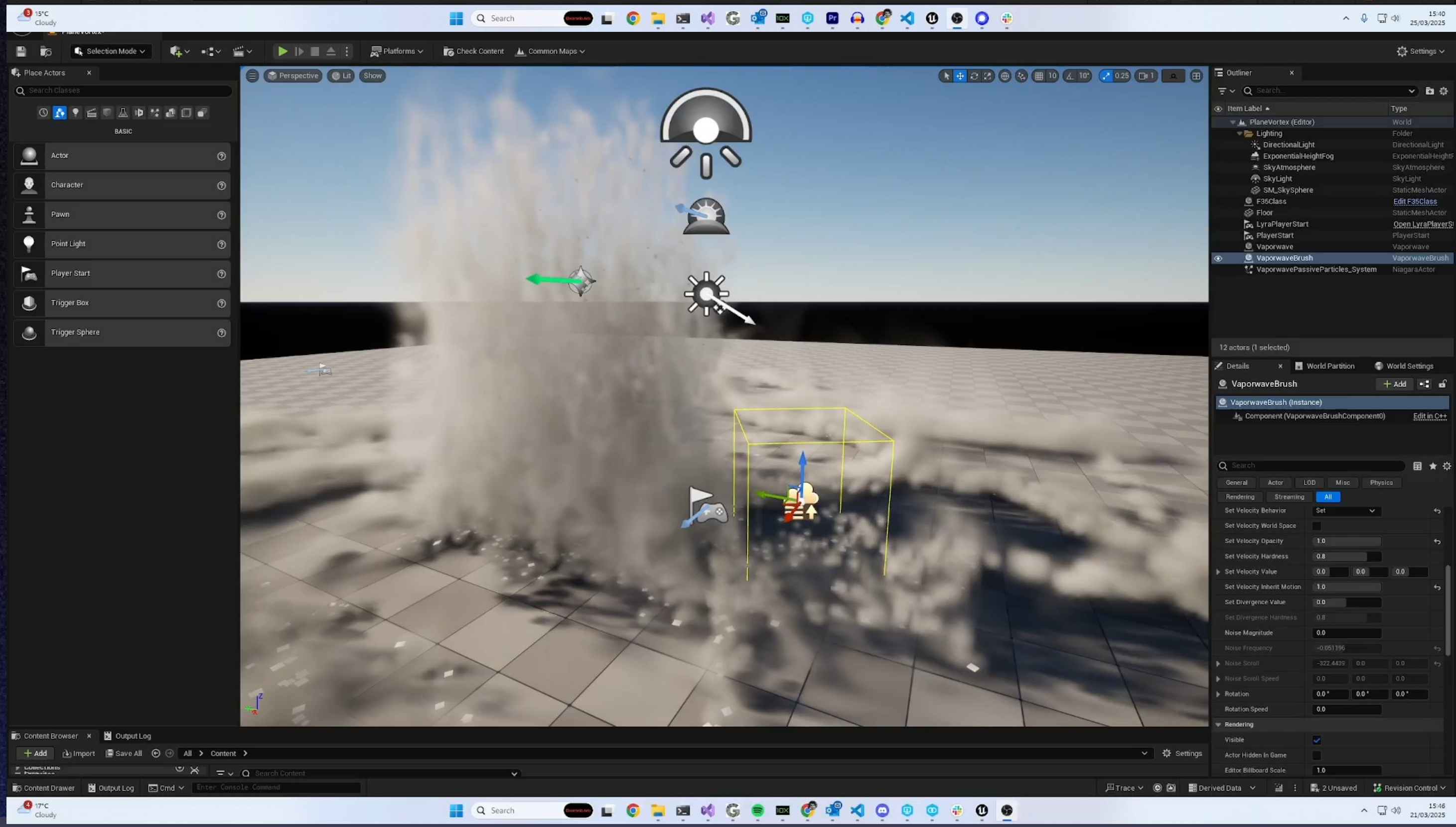




Niagara Interaction

Provide a Niagara Data Interface to allow read-write access to sim from GPU particle effects.





Search for Extraordinary Experiences Division

Rendering

Unreal's Renderer

Unreal's volumetric rendering is based on froxels, and designed around static, thin fog.

Vaporwave content is:

- Thick,
- Inhomogeneous,
- Fast-moving.

We don't want to rely on temporal reprojection.

Occlusion of indirect lighting is important.

Sébastien Hillaire, Advances 2015
Physically-based and Unified Volumetric Rendering in Frostbite

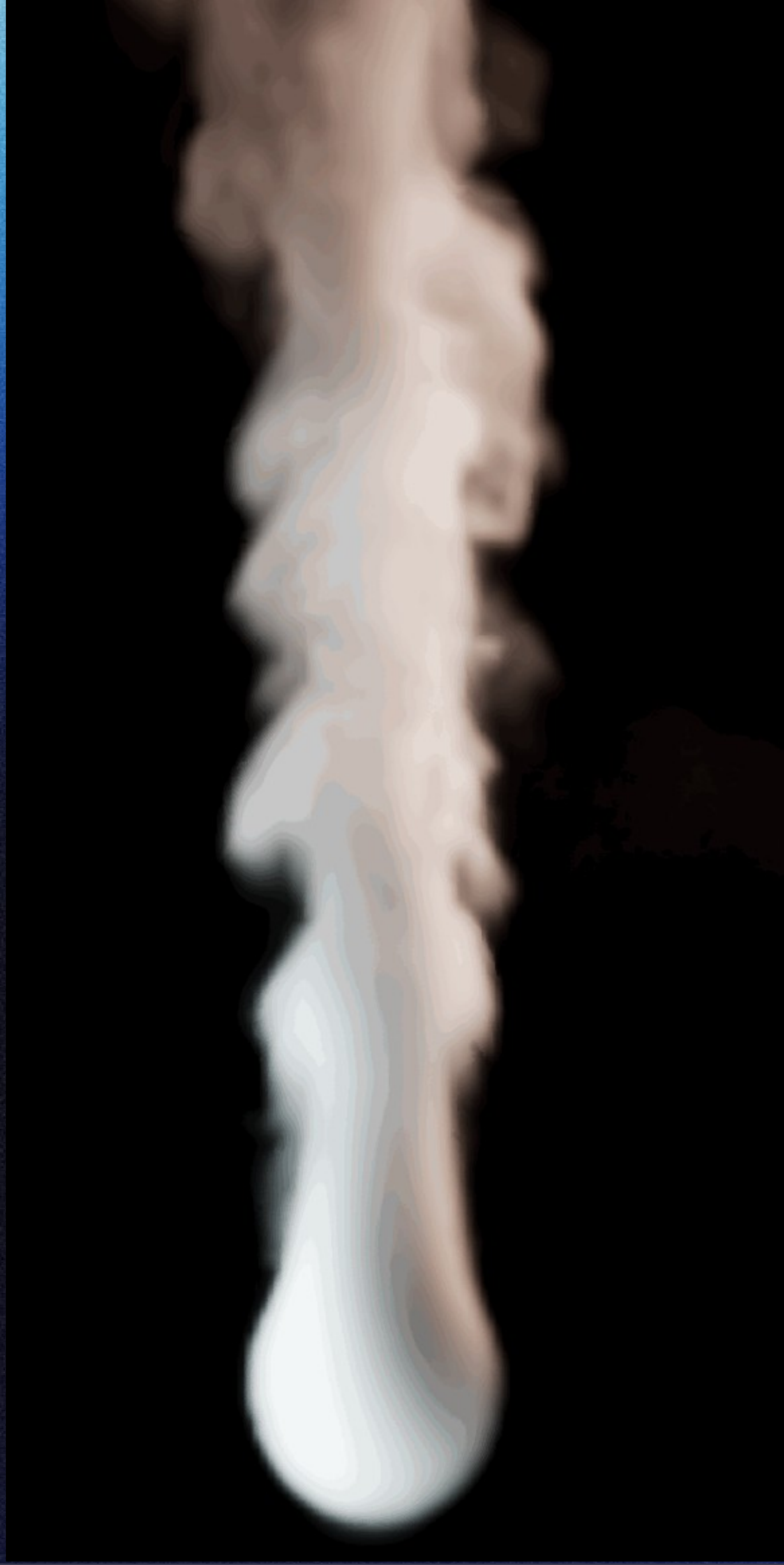
Primary Traversal

We just use ray-marching to the depth buffer

Can sample a single volume

Do light culling+sampling, shadowing etc during traversal.

No need to stash results in a froxel grid.



Lighting

Will Donnelly

Self-occlusion/shadowing is important for look of thick smoke

Without these features, only a limited look is achievable

Full scattering in the realm of an expensive path-tracer

We propose Volumetric Occlusion (VO)

Lighting

Self-occlusion/shadowing is important for look of thick smoke.

Without these features, only a limited look is achievable.

Full scattering in the realm of an expensive path-tracer.

We propose Volumetric Occlusion (VO).

Lighting

Self-occlusion/shadowing is important for look of thick smoke.

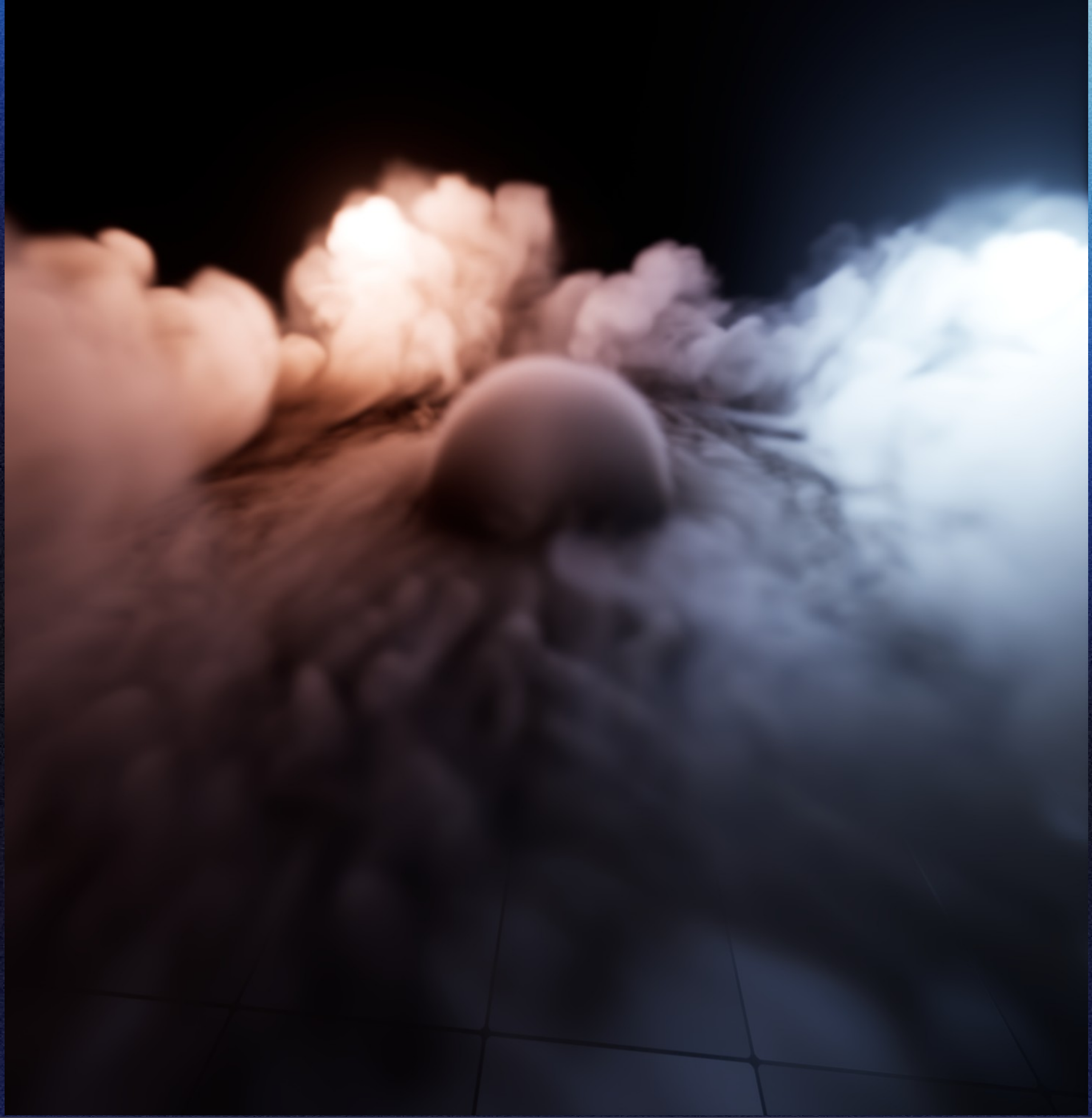
Without these features, only a limited look is achievable.

Full scattering in the realm of an expensive path-tracer.

We propose Volumetric Occlusion (VO).

(It's not just AO for volumes).

Volumetric Occlusion (no directional component)



Volumetric Occlusion

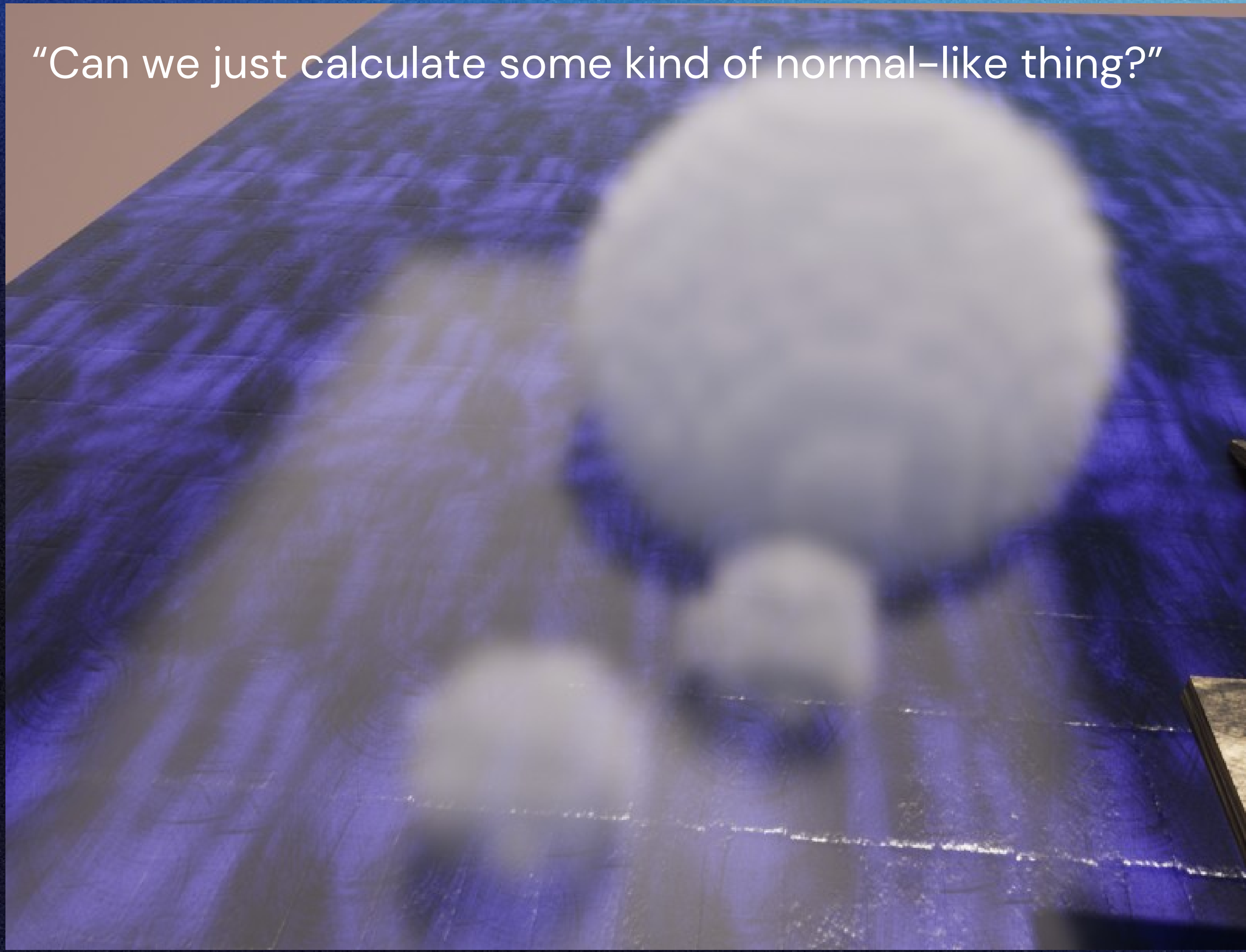
Punctual lighting: can raymarch but expensive.

Environmental light sources: no option.

Only want visual shaping to break up bulk – exact self-shadowing not important.

The normal does this job in surface rendering, can we do something similar?

VO is a principled elaboration of this idea.



Single scattering

Single scattering from environment is an integral:

$$L_s(\omega) = \alpha \int d\omega_i \; p(\omega, \omega_i) \; T(x, \omega_i) \; L_\infty(\omega_i)$$

We multiply together:

- Albedo
- Phase function
- Beam transmittance i.e. visibility 0…1
- Environment lighting

Integrate over all incoming light directions.

VO Single scattering from environment

Visibility function is the negative exponential of the *optical thickness*

$$T(x,\omega) = e^{-\tau(x,\omega)} \qquad \tau(x,\omega) = \int_0^\infty \sigma_t(x + t\omega)dt$$

Approximate optical thickness with a linear function: $\tau(x,\omega) \approx v_0(x) + \vec{v}_1(x) \cdot \omega$

$$v_0(x) = \int d^3y \frac{\sigma_t(y)}{4\pi r^2} \qquad \vec{v}_1(x) = \int d^3y \frac{\sigma_t(y)}{4\pi r^2} 3\hat{r}$$

Convolution is implemented as a sum over the density Clip/Mip hierarchy.

Visibility is exponential of a linear function – a spherical gaussian: $T(x,\omega) = e^{-v_0(x)+v_1(x)\cdot\omega}$

Integral of Spherical Harmonic lighting times Spherical Gaussian visibility is solved analytically.

Single scattering from environment

No Occlusion

Volumetric Occlusion

Path Traced Reference

Phase function

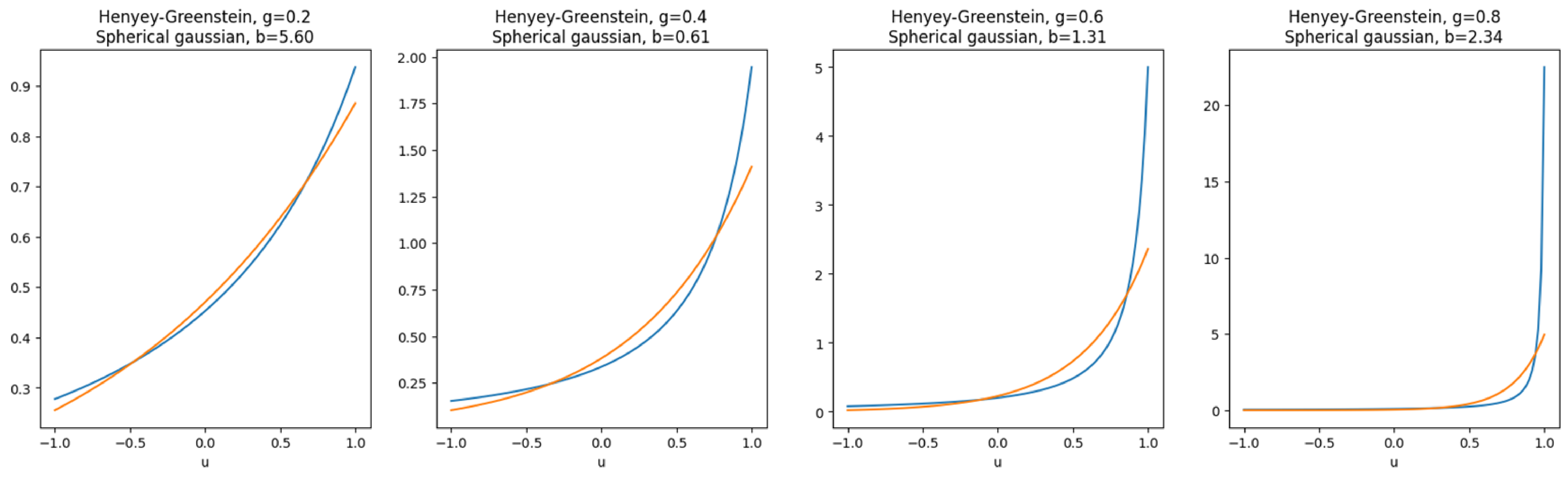
Phase function is typically Henyey-Greenstein:

$$P_{\mathrm{HG}}(u; g) = \frac{1 - g^2}{4\pi(1 - 2gu + g^2)^{3/2}}$$

We fit a normalized spherical gaussian:

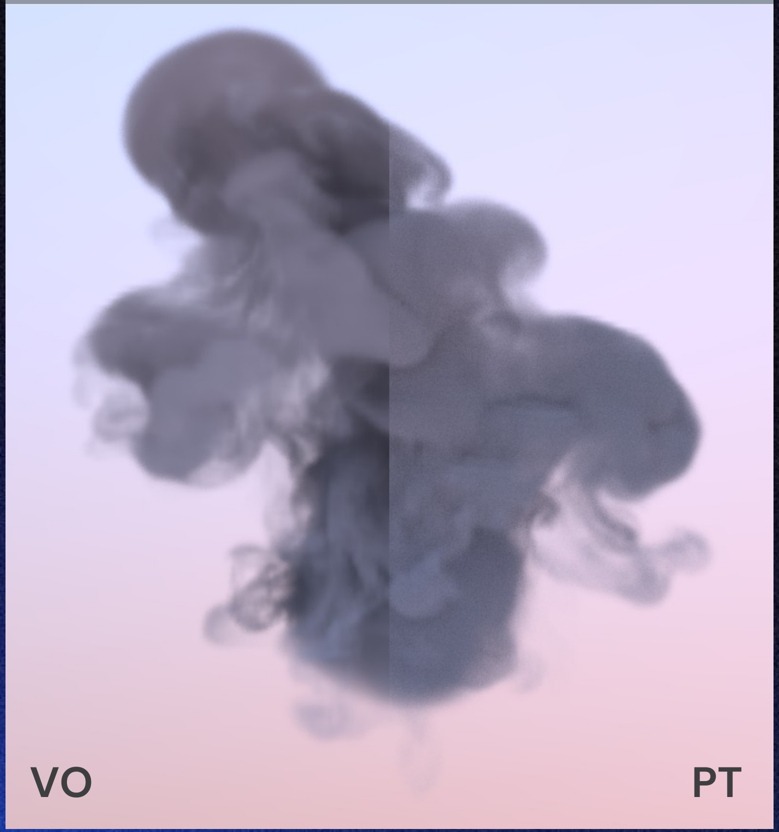
$$P_{\mathrm{SG}}(u; k) = \frac{ke^{ku}}{4\pi \sinh(k)}.$$

Multiplication with visibility is cheap! Product of spherical gaussians is spherical gaussian.

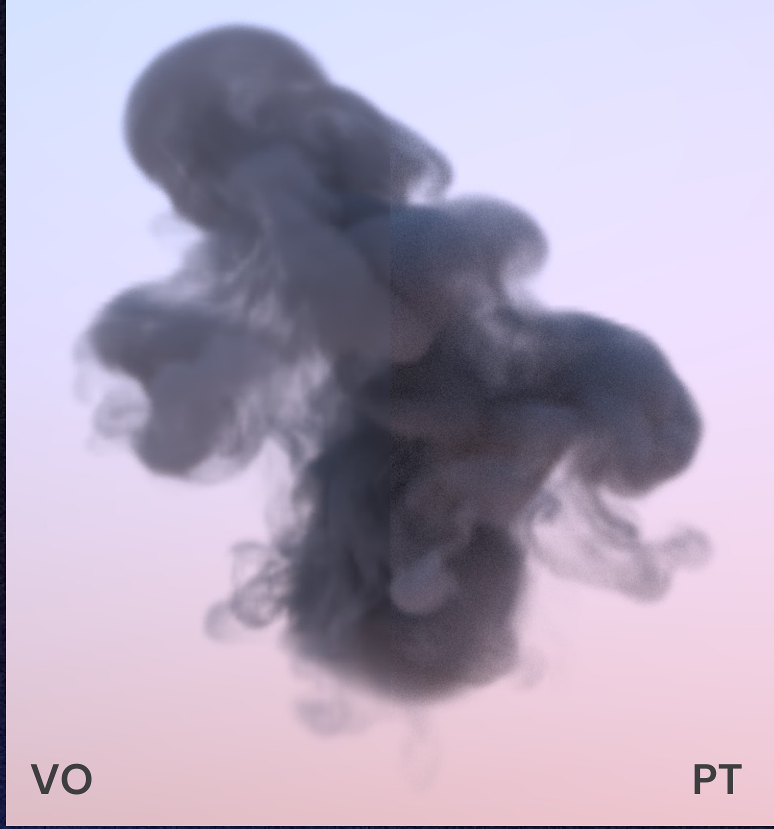


Phase Function

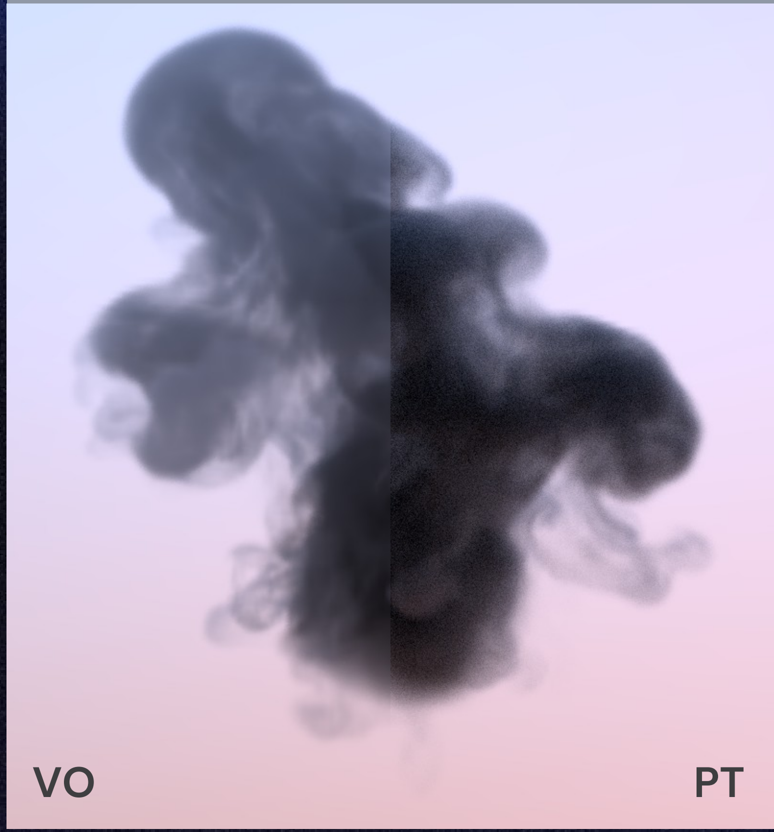
Backscattering g = -0.8



Isotropic g = 0



Forward scattering g = 0.8



Multiple scattering

Reduce to 1D: Assume fraction f of paths scatter perfectly forward, and b perfectly backward:

$$f = \frac{1+g}{2g}\left(1 - \frac{1-g}{\sqrt{1+g^2}}\right), \qquad b = 1 - f$$

Volume rendering reduces to Kubelka-Munk theory.

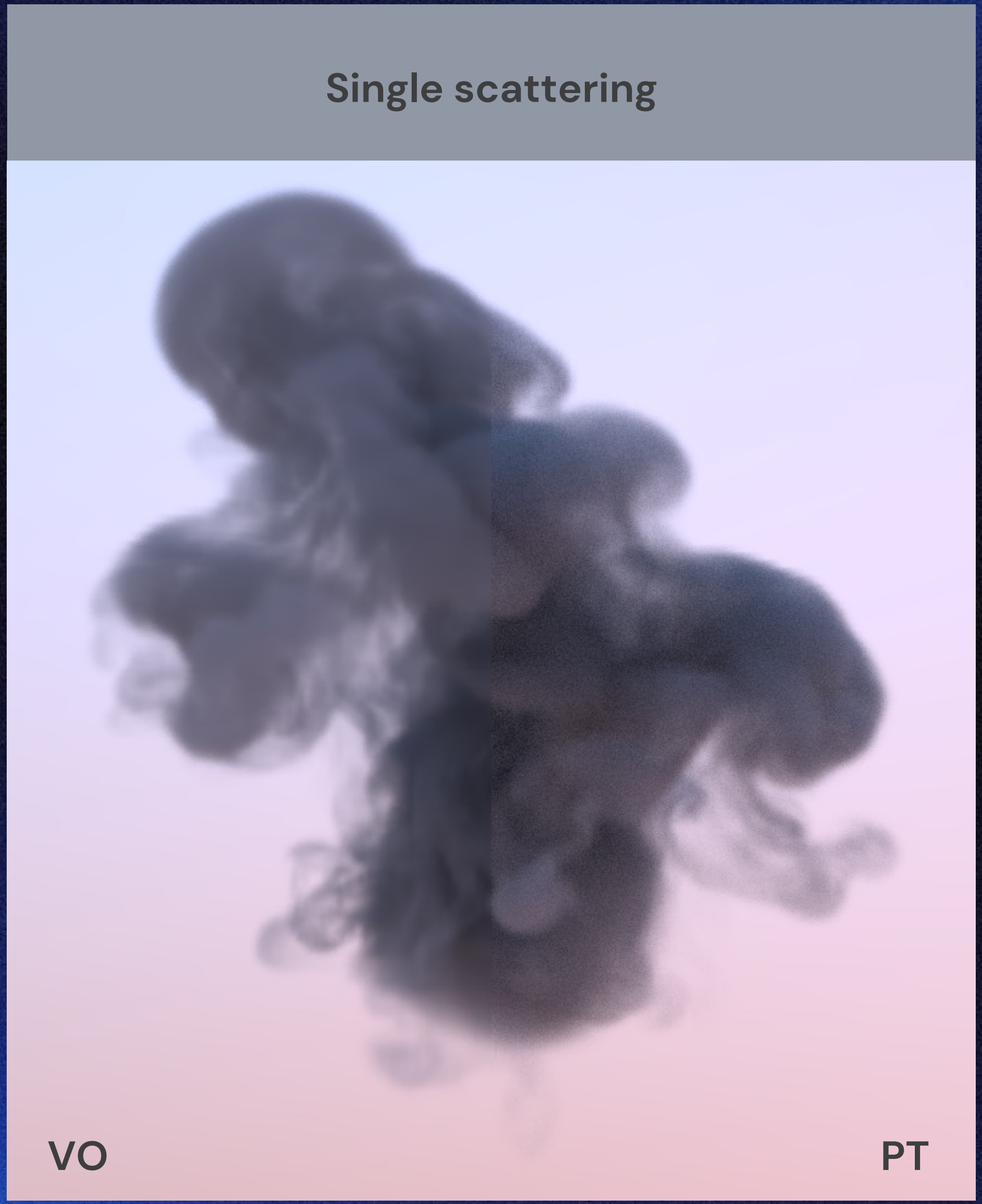
Extinction coefficient is replaced with a new *effective* extinction coefficient:

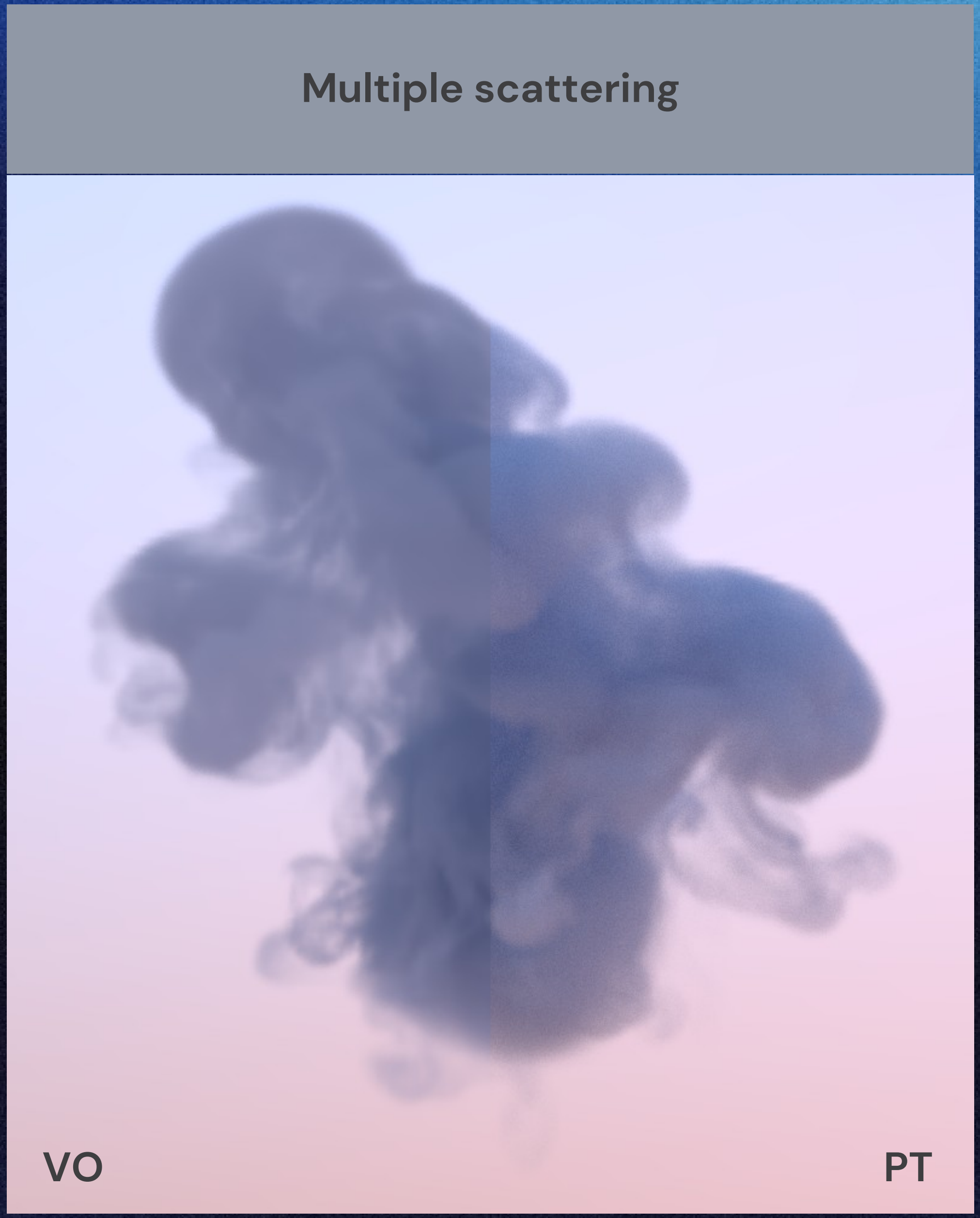
$$\sigma_e = \sigma_t \sqrt{(1-\alpha f)^2 - (\alpha b)^2}.$$

Note: effective extinction depends on albedo - it now has a color.

Extinction coefficient can be precomputed per material; all the rest of the math is the same.

Multiple Scattering



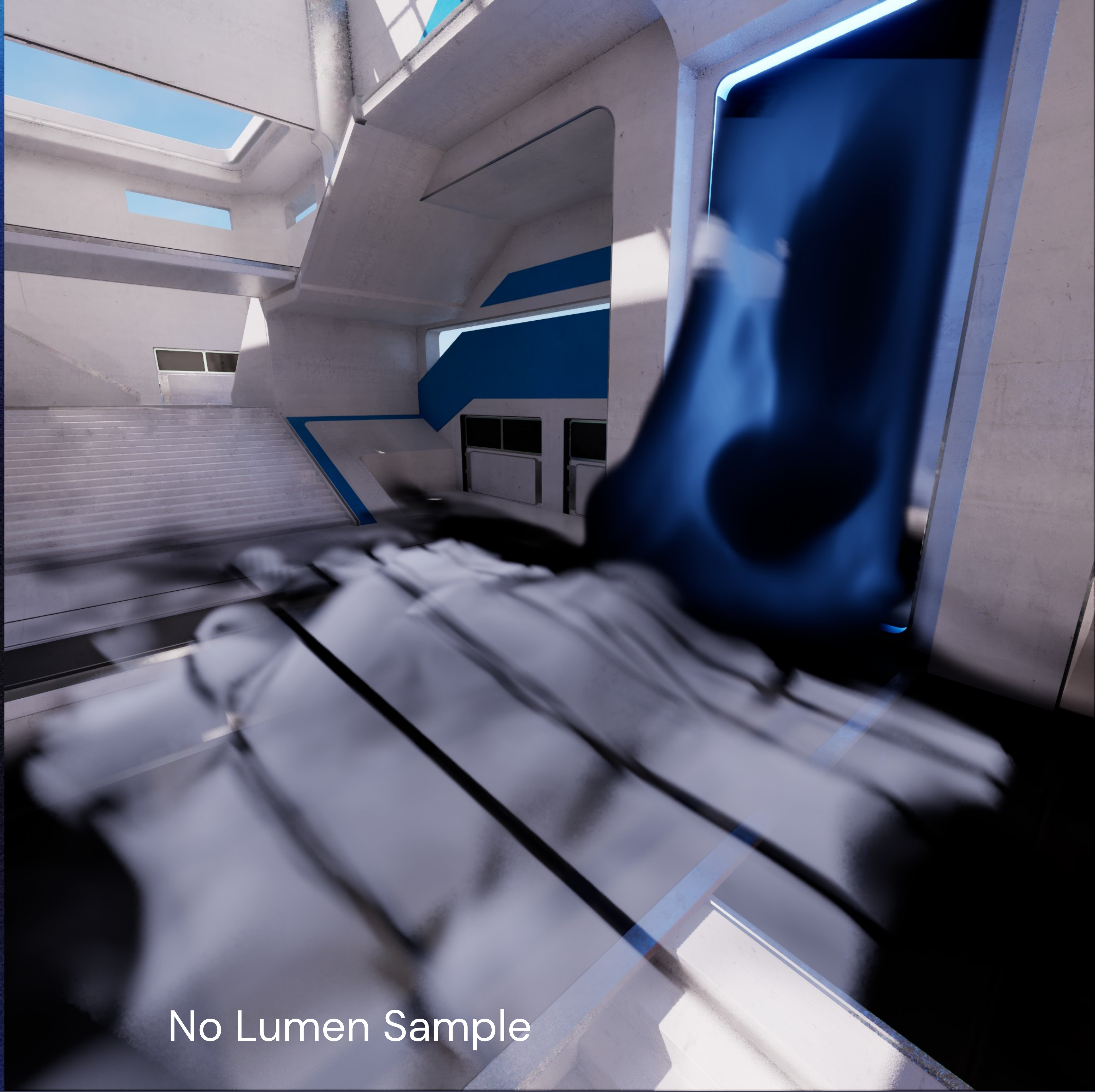


Use in Unreal

Use to attenuate both direct and environment lighting

Can easily integrate against Lumen or Translucency Lighting Volume

Can use for simple shadowing

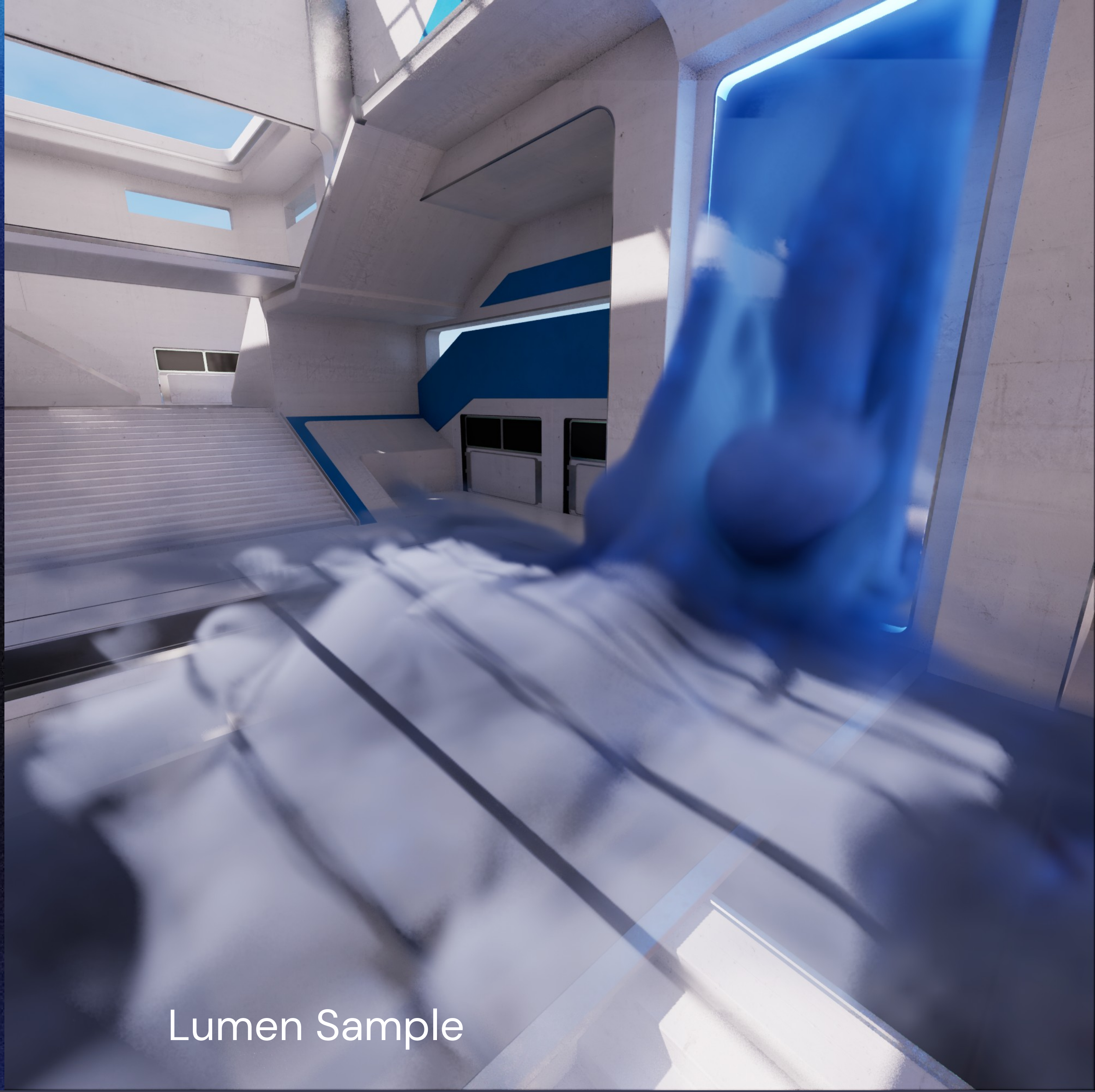


Use in Unreal

Use to attenuate both direct and environment lighting

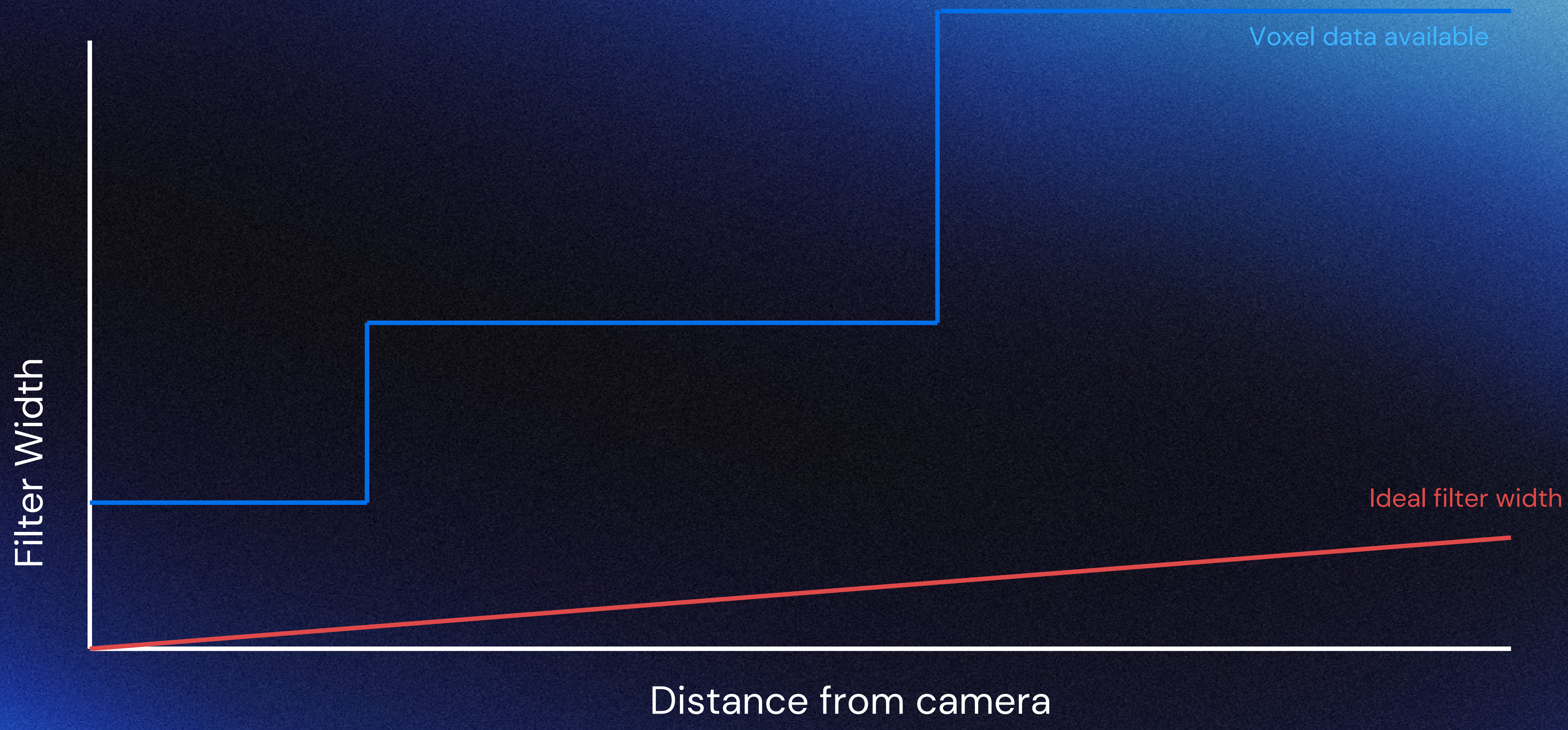
Can easily integrate against Lumen or Translucency Lighting Volume

Can use for simple shadowing

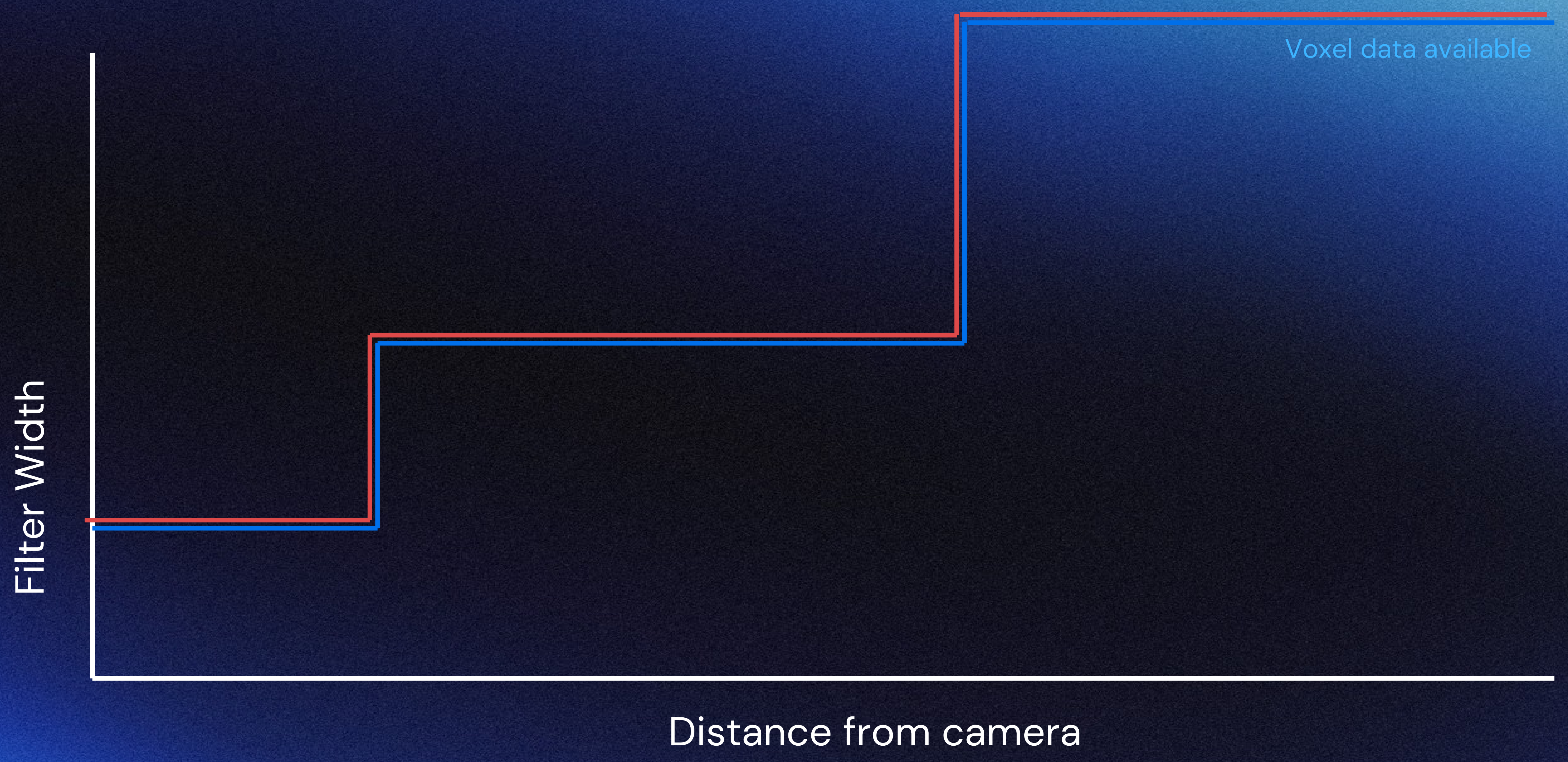


Lumen Sample

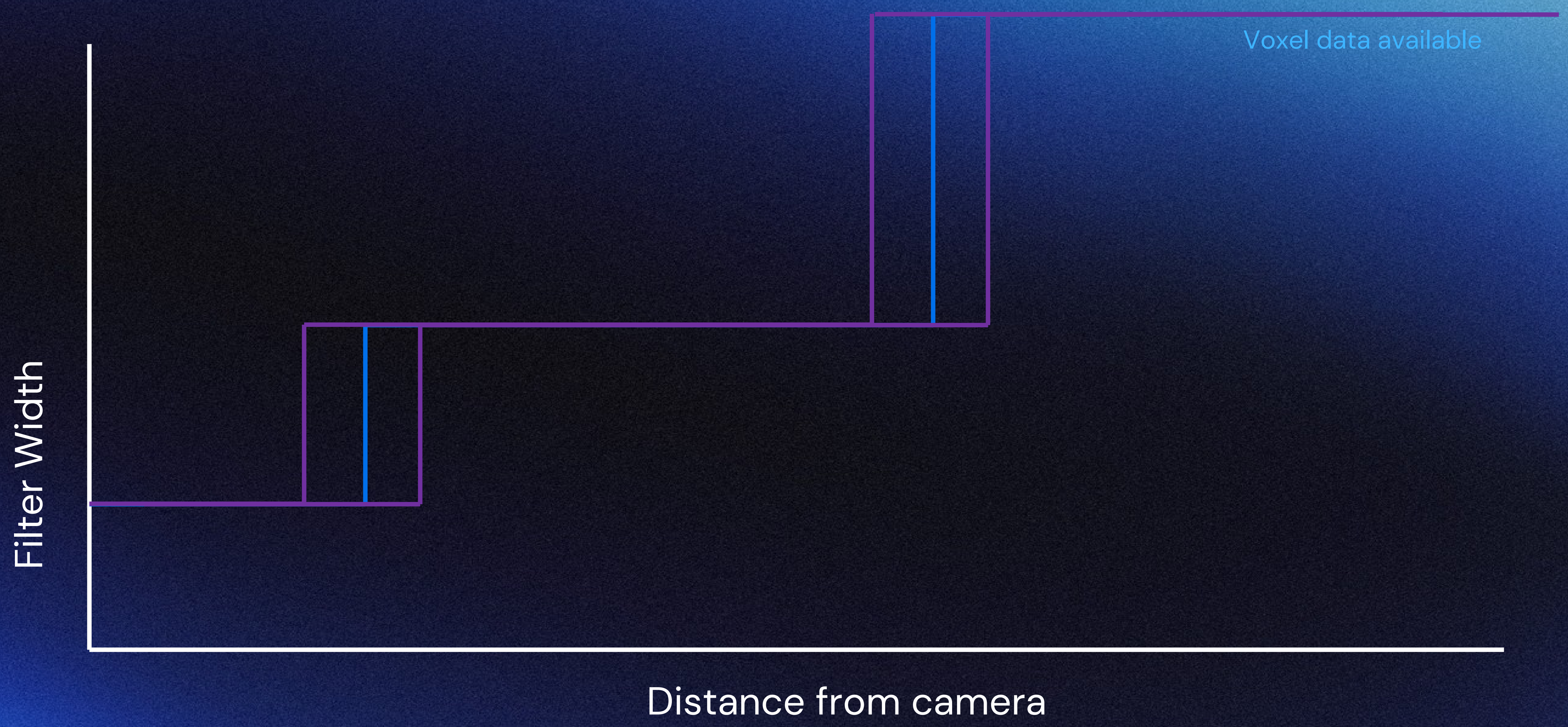
Pop Suppression



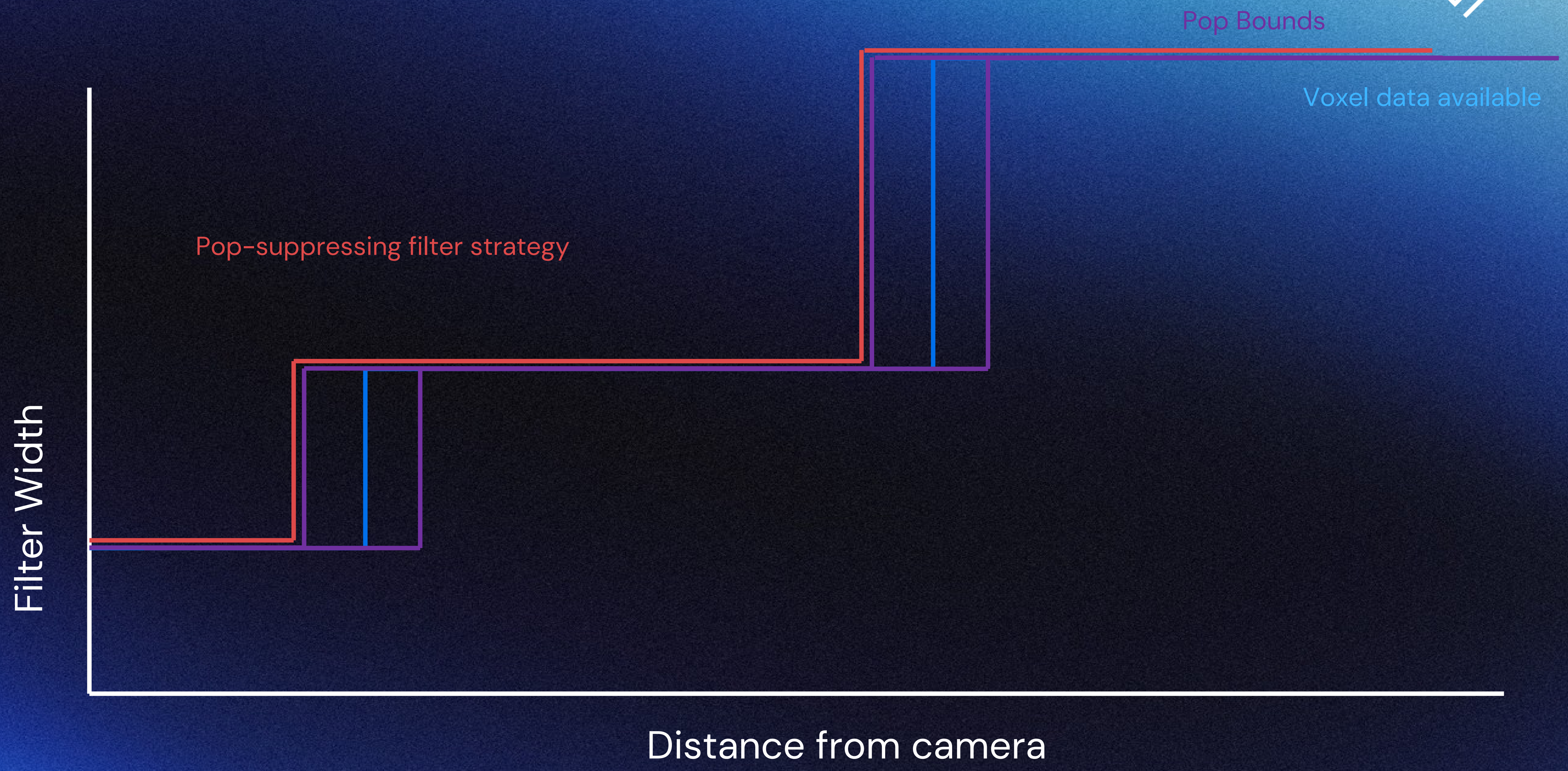
Pop Suppression



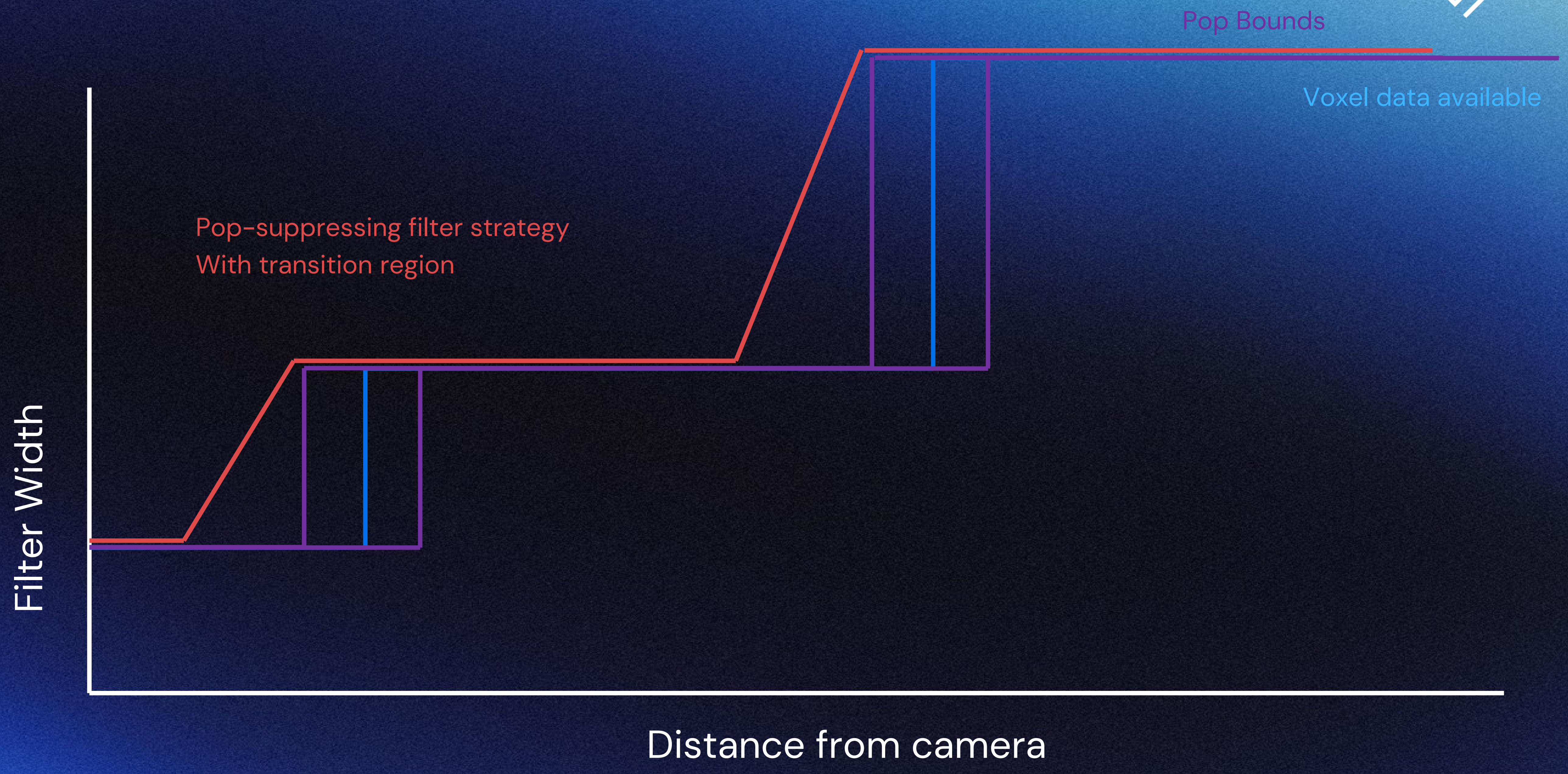
Pop Suppression



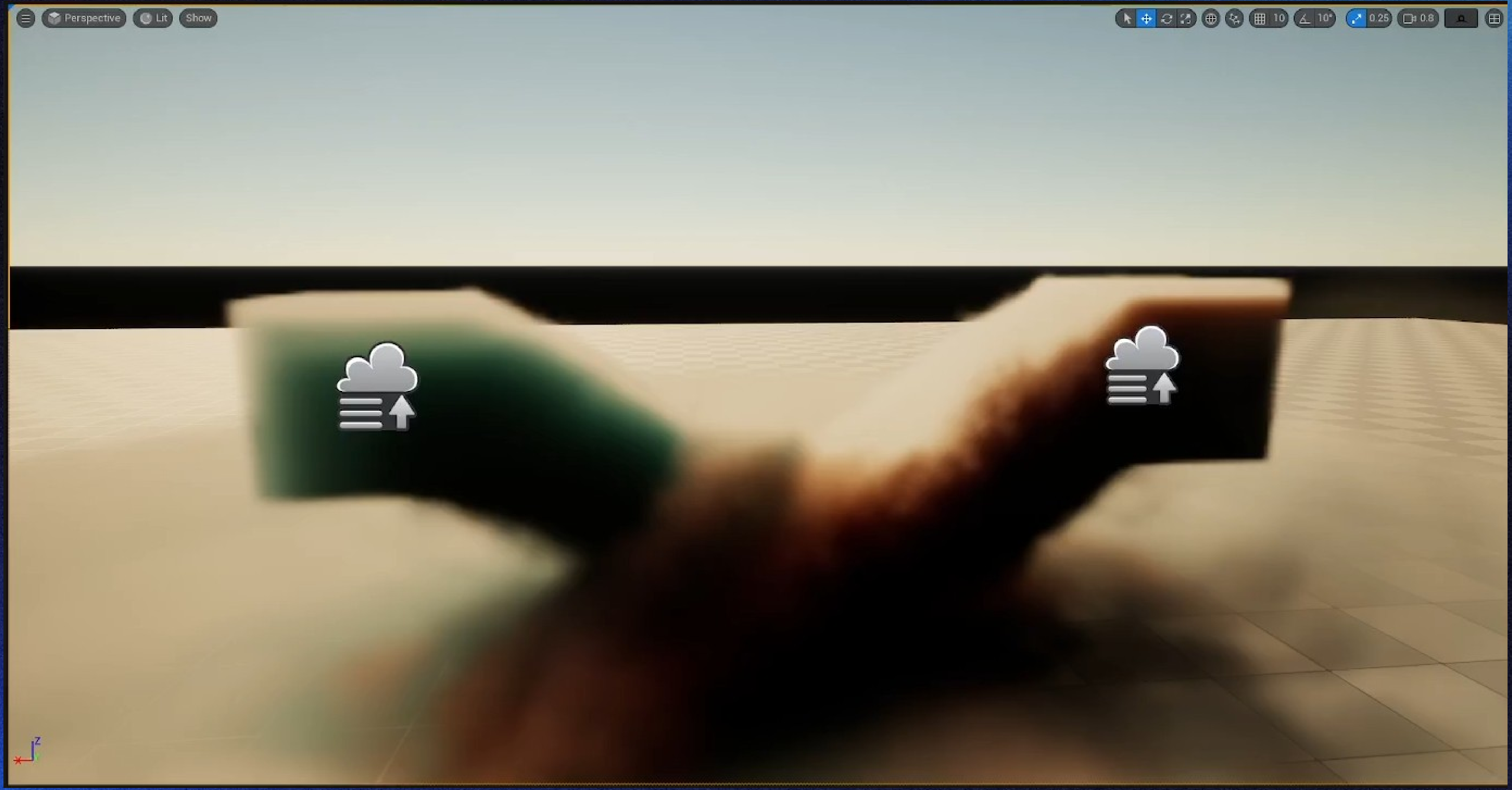
Pop Suppression



Pop Suppression



Pop Suppression



Performance Results

(gpu, on a console platform)

Limitations

Memory use with dense allocation

Sniper rifles

IOP collision is ‘eh’

Gameplay / Multiplayer

Future Direction

Ship a game!

Perfecting VO

Better explosive effects / compressible simulation

How do we do this in a fully path traced world?

Thank you!

Special thanks:

Gigi – Alan Wolfe https://github.com/electronicarts/gigi

Partner Teams

Morten Vassvik

Learning Resources:

Rook Bridson – Fluid Simulation for Computer Graphics

WL Briggs et al. – A Multigrid Tutorial

Martin and Cartwright – Solving Poisson's Equation using Adaptive Mesh Refinement