This slide is intentionally left blank.

STUDIOS™ GUERRILLA™ DECIMA

Photograph

Horizon Zero Dawn, 2017

“Cloud Coverage” “Cloud Type” “Perlin-Worley”

Visualization

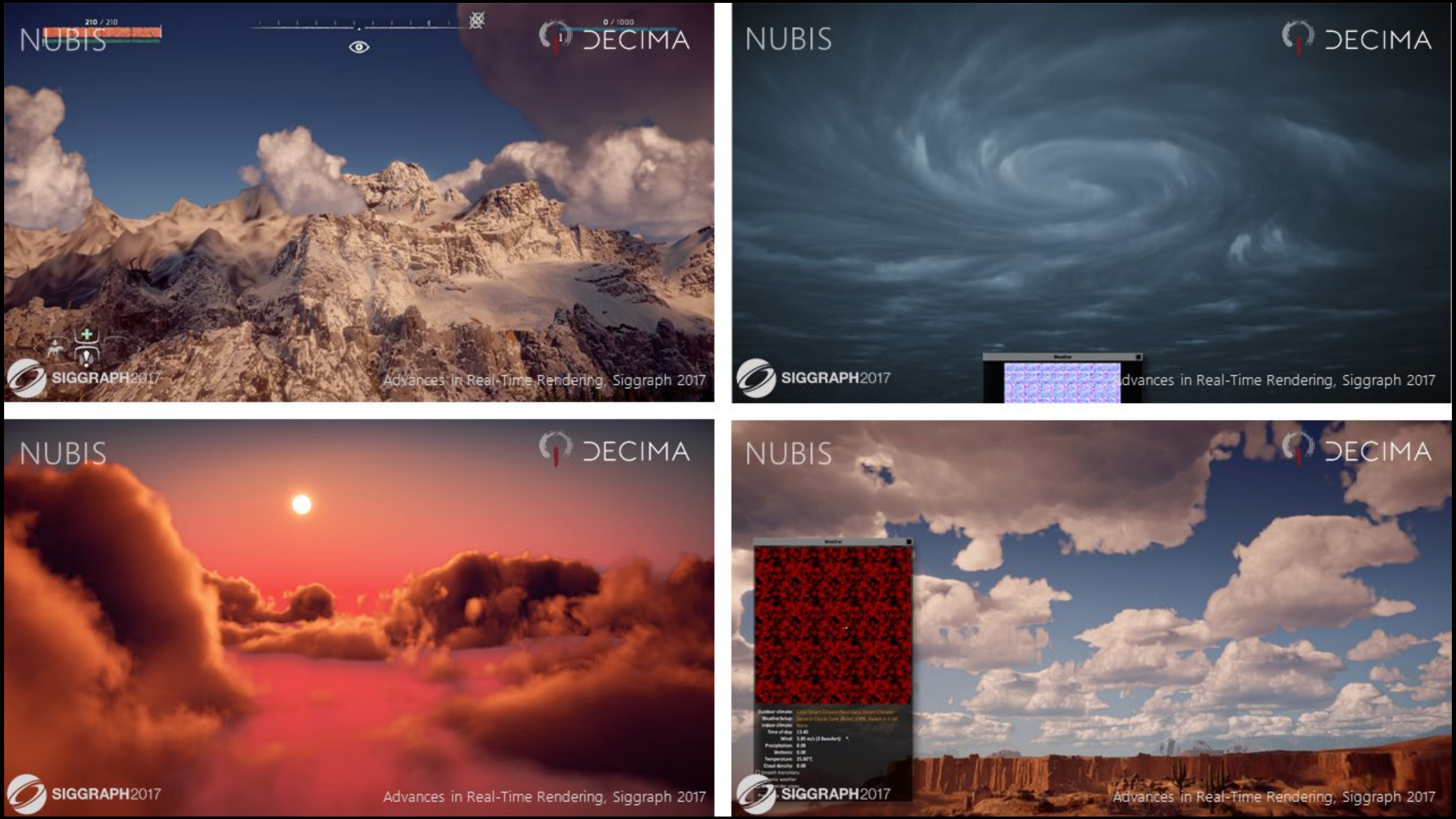
The Real-Time Volumetric Cloudscapes of Horizon Zero Dawn (2015)

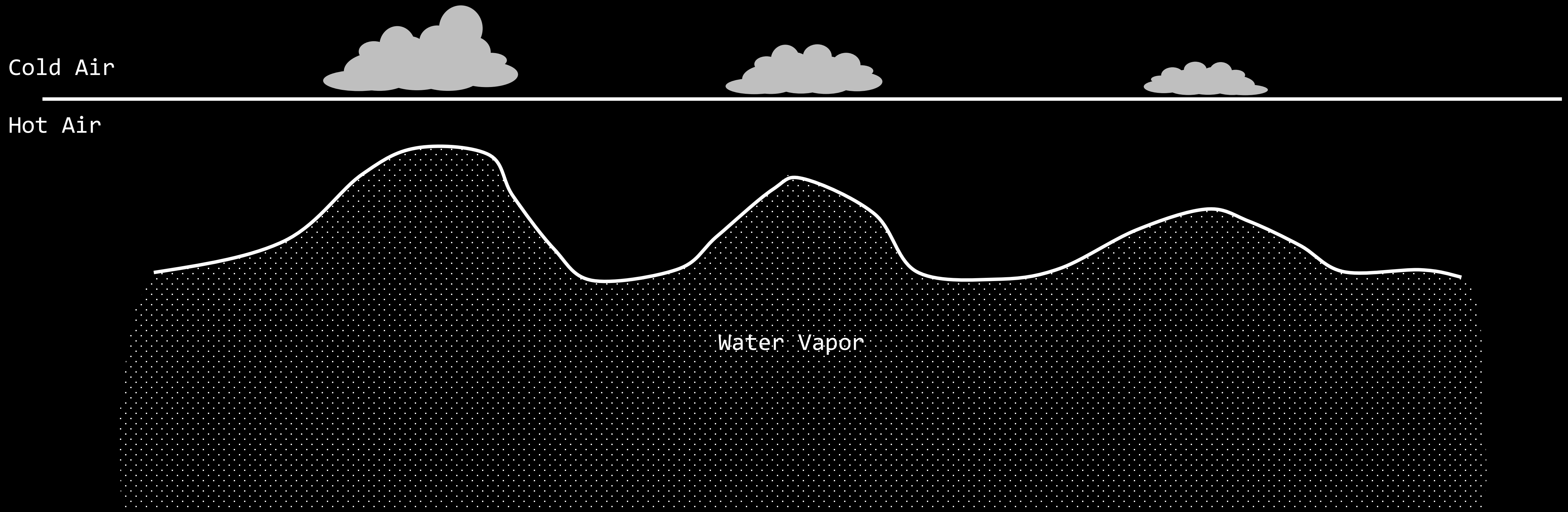
Nubis: Authoring Real-Time Volumetric Cloudscapes with the Decima Engine (2017)

Real-Time Volumetric Cloudscapes for Games (2016)

Horizon Forbidden West, 2022

NUBIS EVOLVED / Background




Cold Air
Hot Air
Water Vapor

Photograph

Luke Howard, 1802

Cirrus ▶

Cumulus ▶

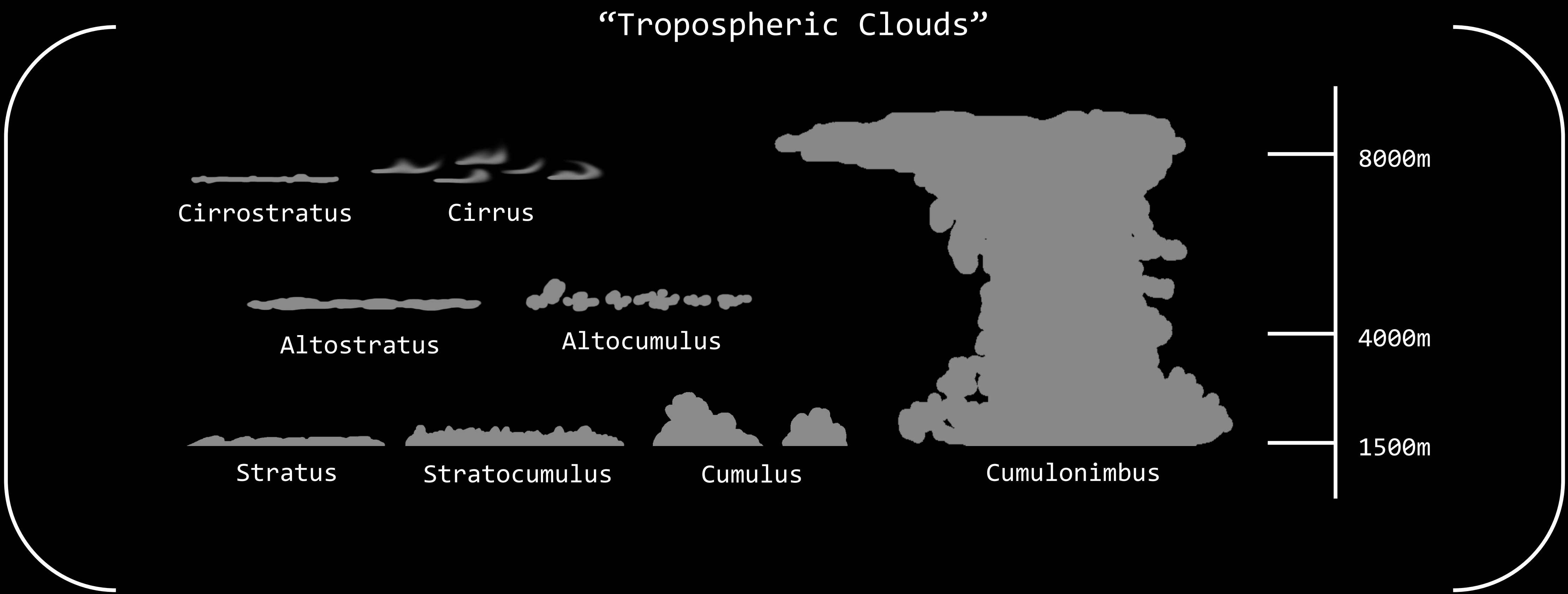
Stratus ▶

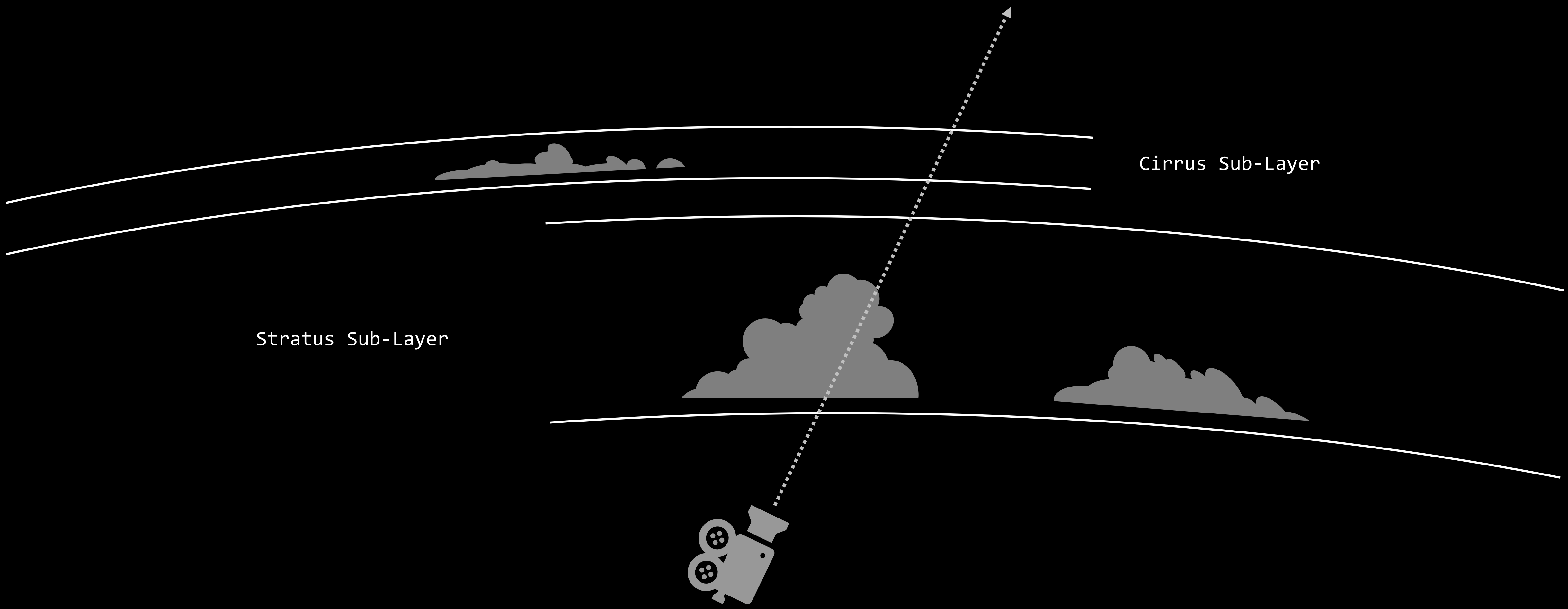
“Nubification”

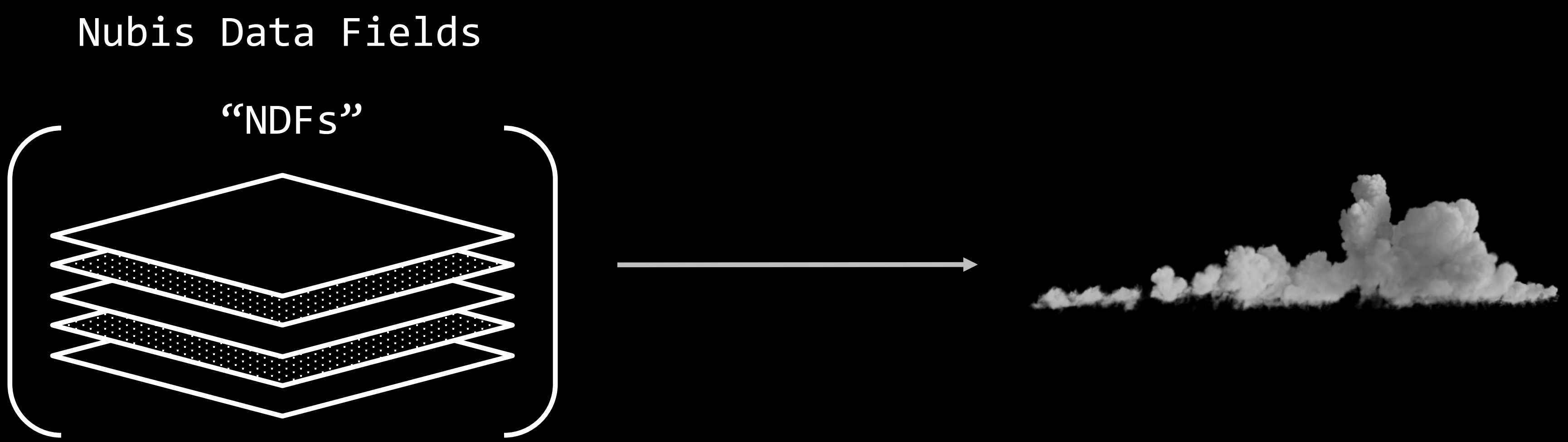
“Nubis”

- Structure
- Movement
- Color

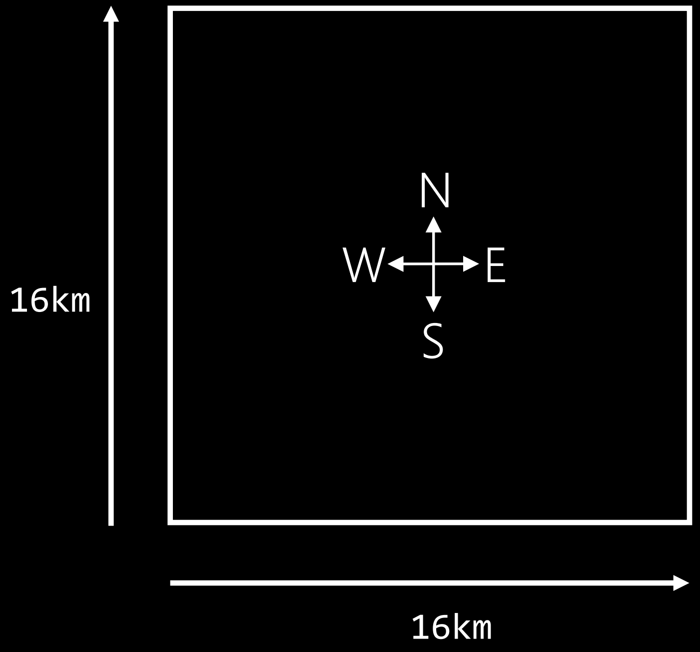
Photograph


"Tropospheric Clouds"
Cirrostratus
Cirrus
Altostratus
Altocumulus
Stratus
Stratocumulus
Cumulus
Cumulonimbus
8000m
4000m
1500m

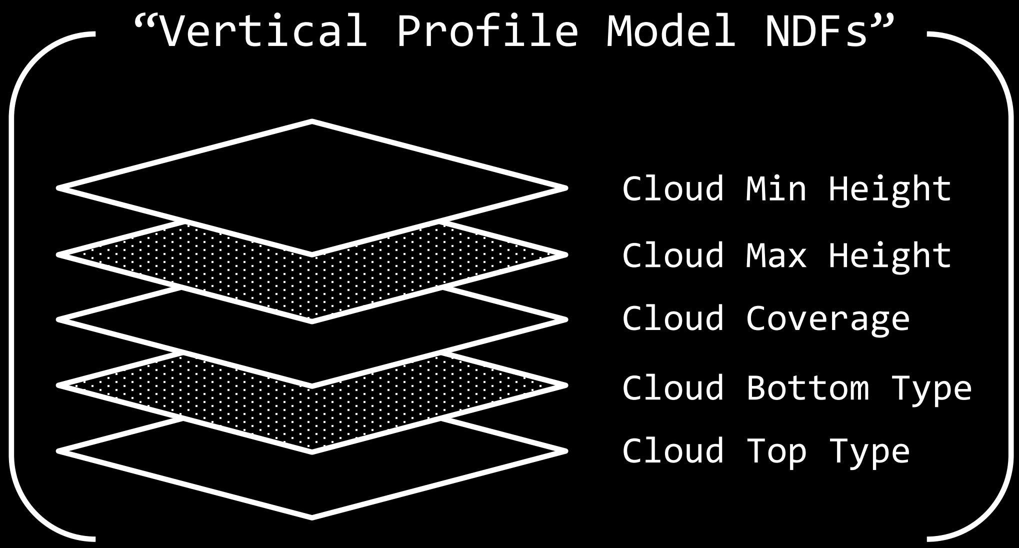

Cirrus Sub-Layer
Stratus Sub-Layer

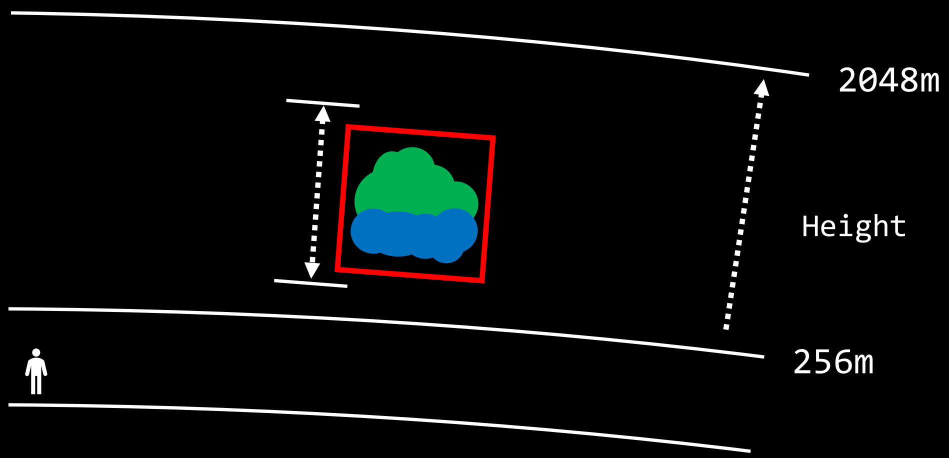

Nubis Data Fields
"NDFs"

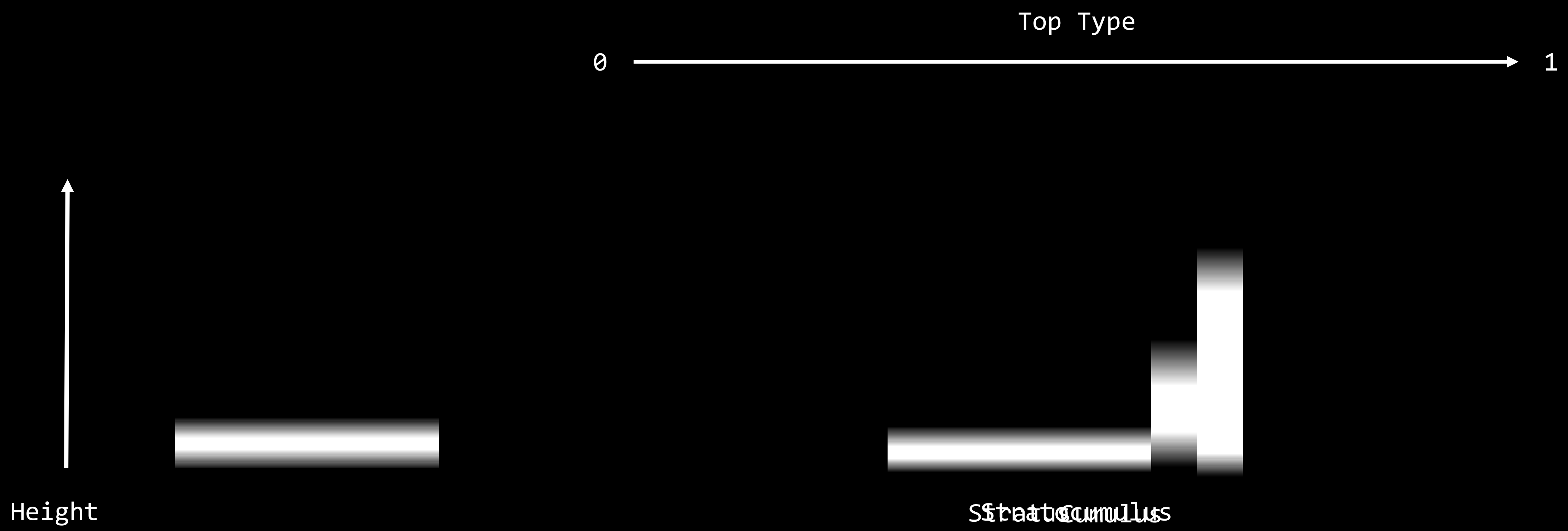
2D NDF Mapping

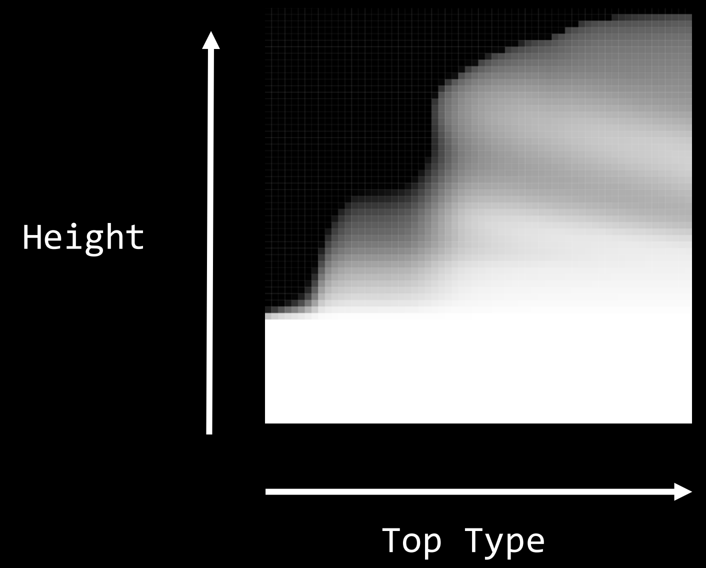


Vertical Profile Model


"Vertical Profile Model NDFs"
Cloud Min Height
Cloud Max Height
Cloud Coverage
Cloud Bottom Type
Cloud Top Type


2048m
Height
256m

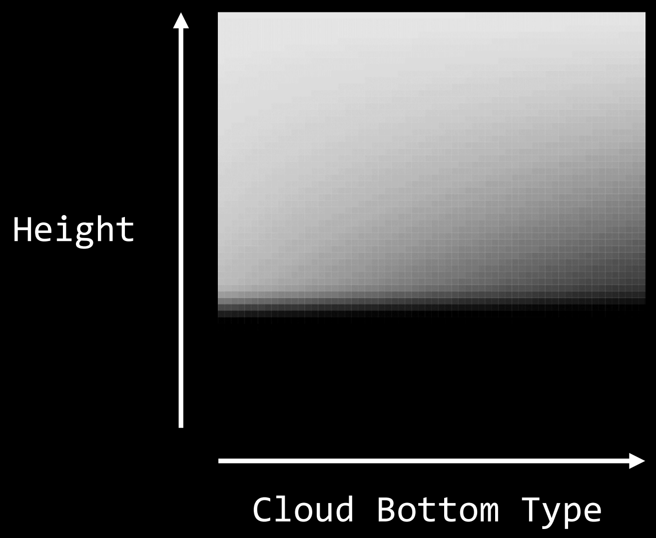

Top Type
0
1
Height


Height
Top Type


Cumulus
In-Engine Render

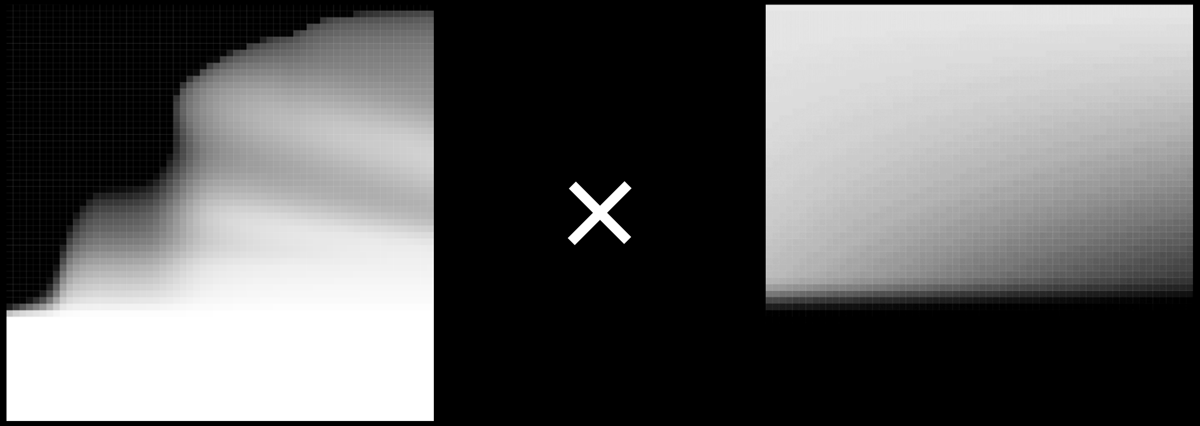
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Density


Stratus
In-Engine Render


Height
Cloud Bottom Type


Bottom Type = 0.0
In-Engine Render

“Vertical Profile”

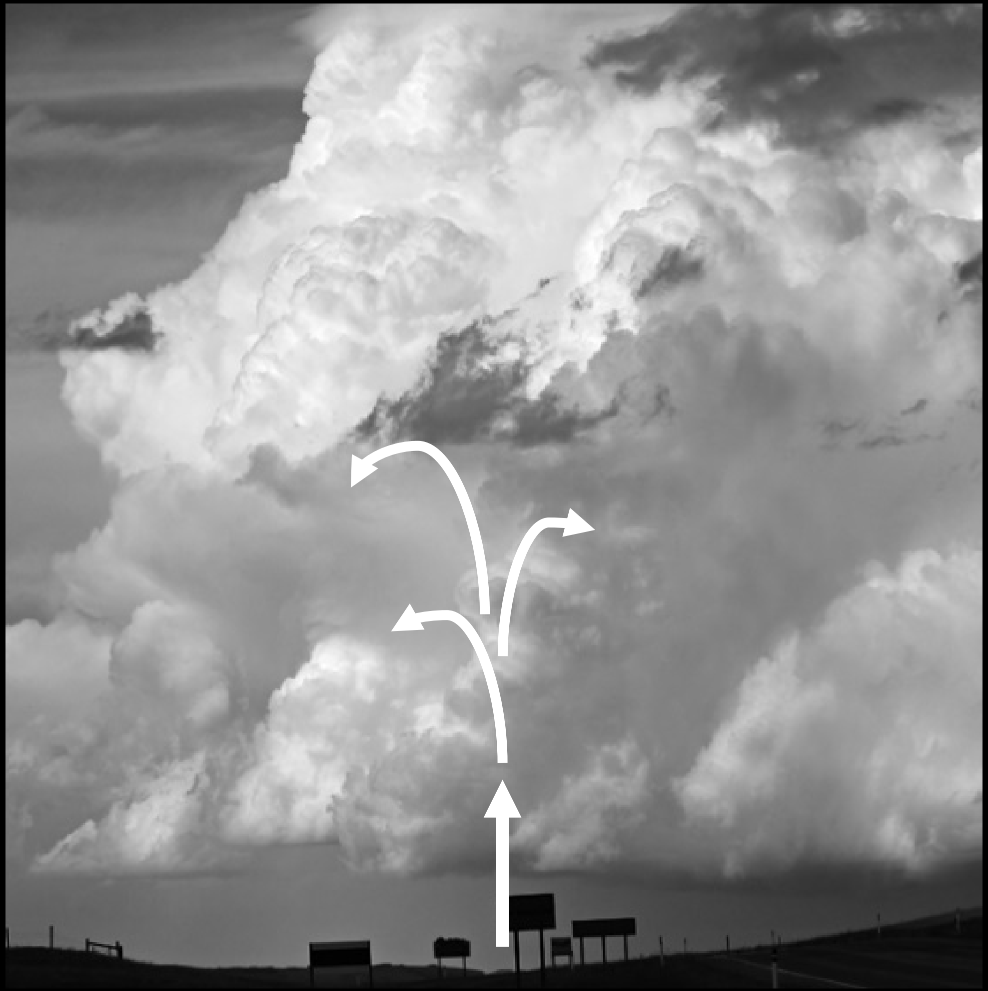


Coverage

0 → 1

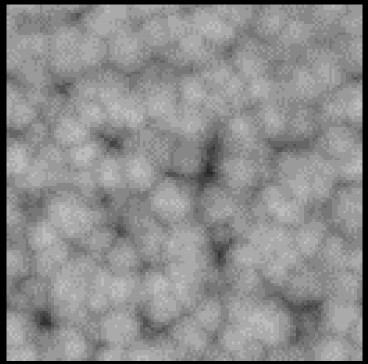
```
float dimensional_profile = vertical_profile * cloud_coverage;
```


In-Engine Render

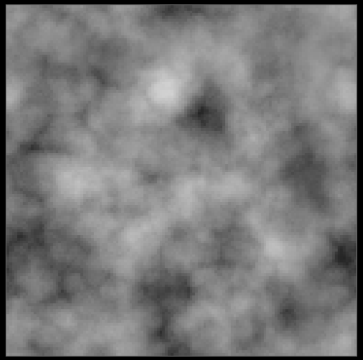


Photographs

Perlin

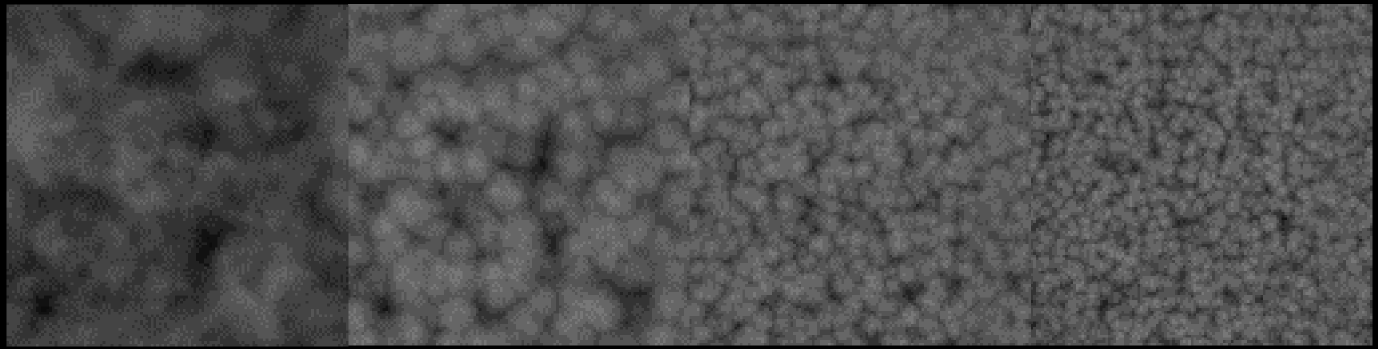


1-Worley



‘Perlin-Worley’

4 Channel [128^3]



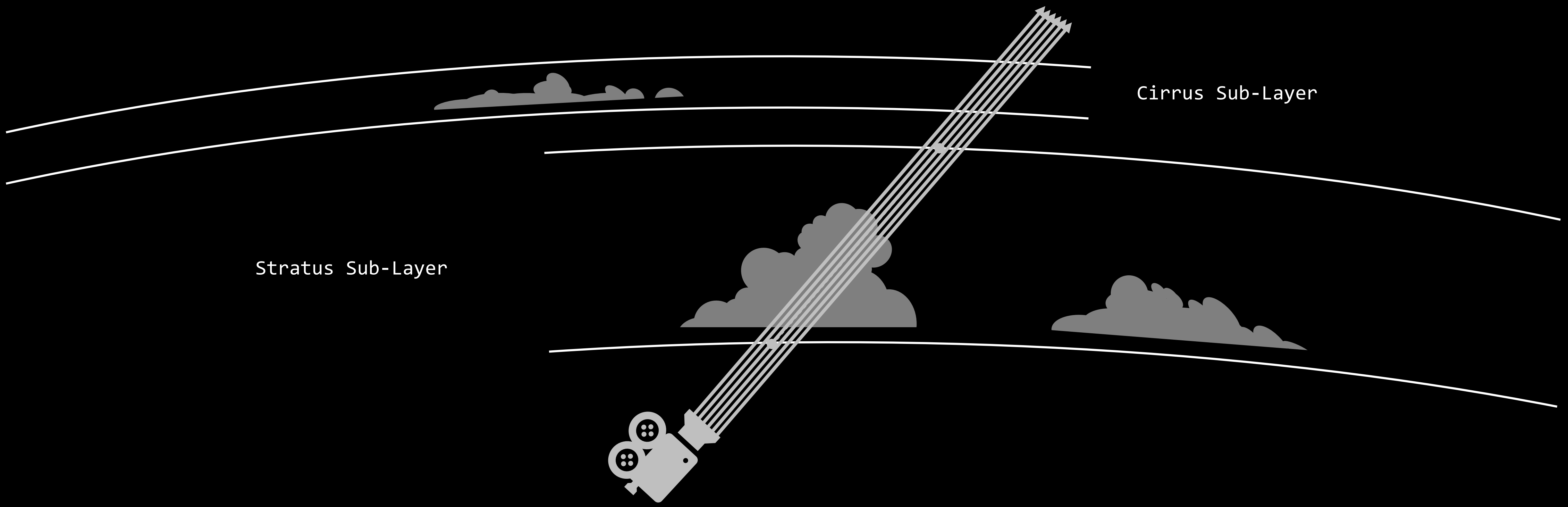
"Noise Composite"

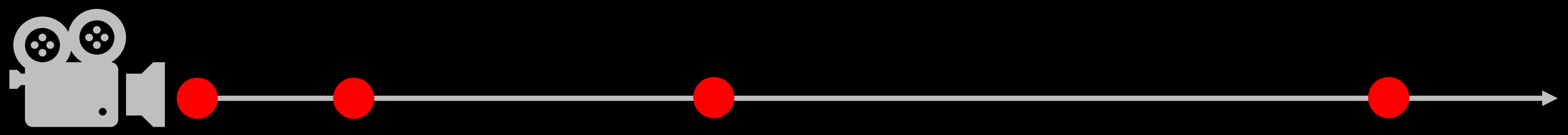
```
float cloud_density = saturate(cloud_noise_composite - (1.0 - dimensional_profile));
```


In-Engine Render

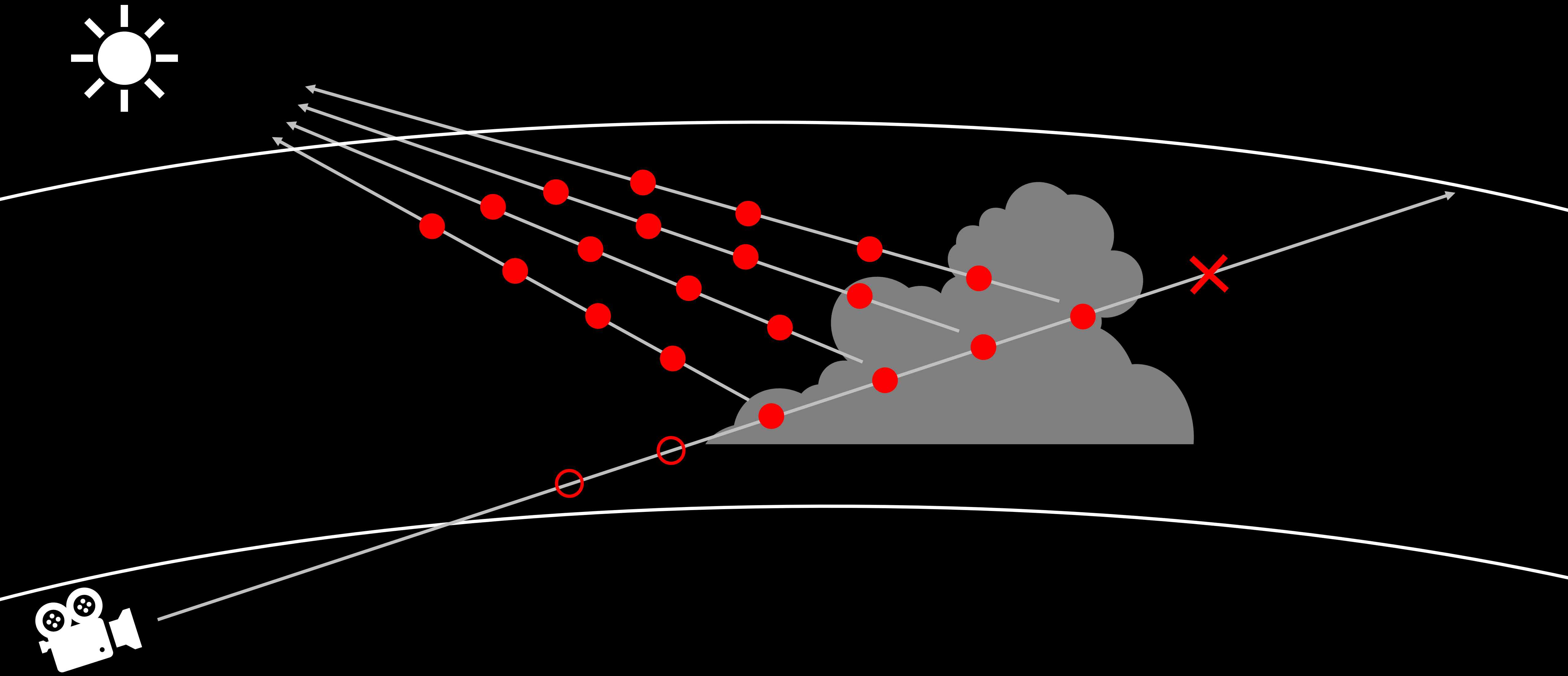

In-Engine Render

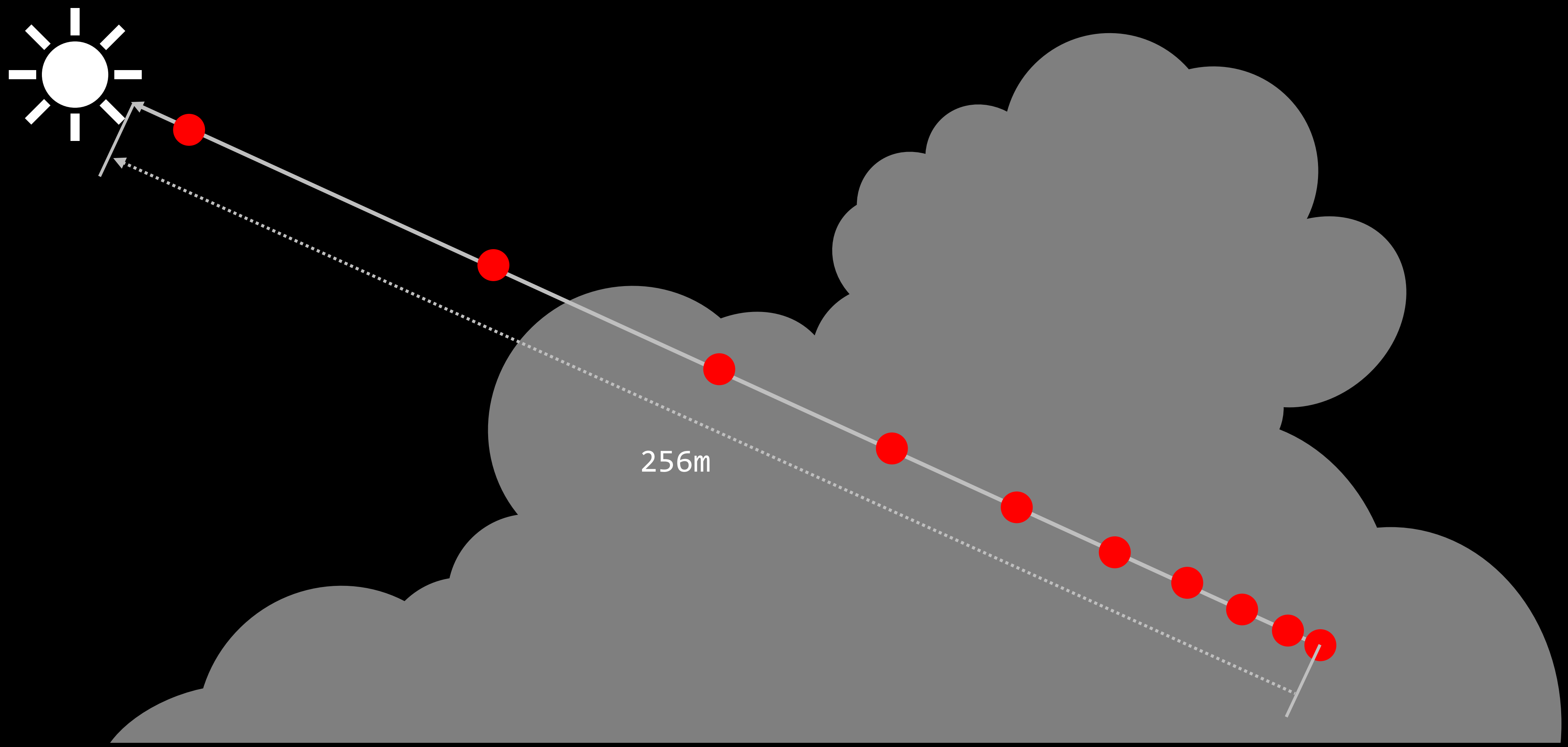

```
// Deform sample coordinates
float3 noise_sample_position = sample_position - wind_direction  * scroll_offset;
```

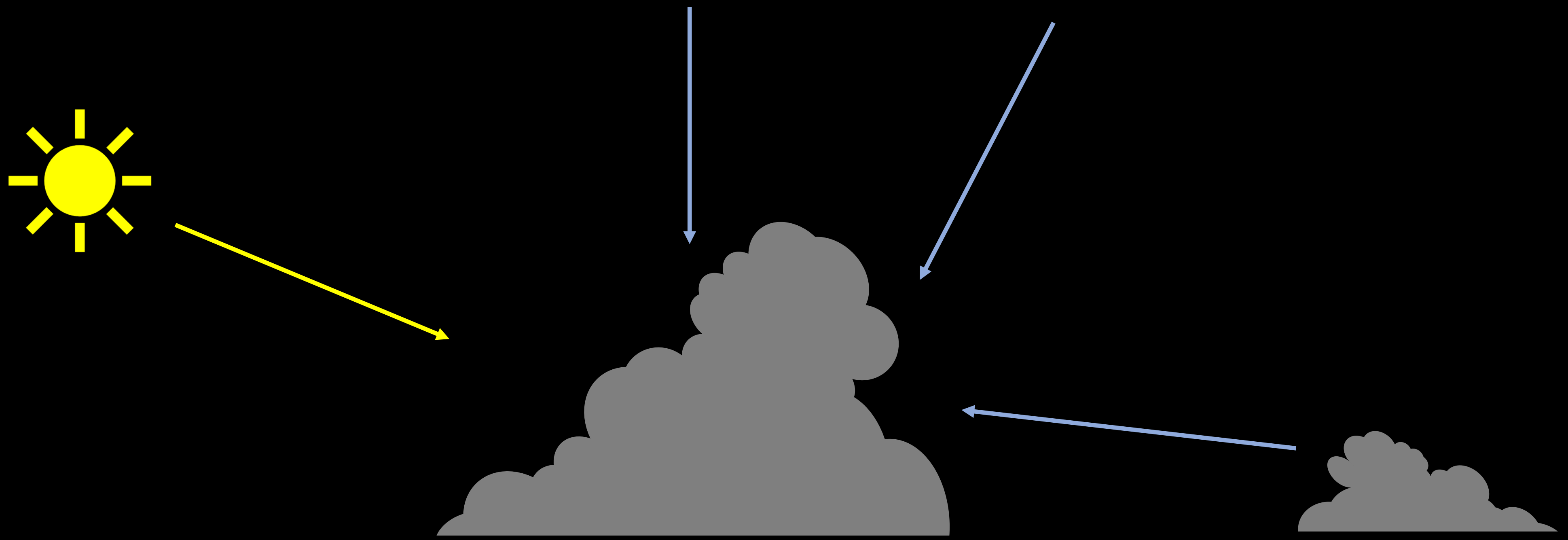

Cirrus Sub-Layer
Stratus Sub-Layer



```
// Define step size constants
float near_step_size = 3.0;
float far_step_size_offset = 60.0;
float step_adjustment_distance = 16384.0;

// Calculate distanced-based step size
float step_size = near_step_size + ((far_step_size_offset * distance_from_camera) / step_adjustment_distance);
```

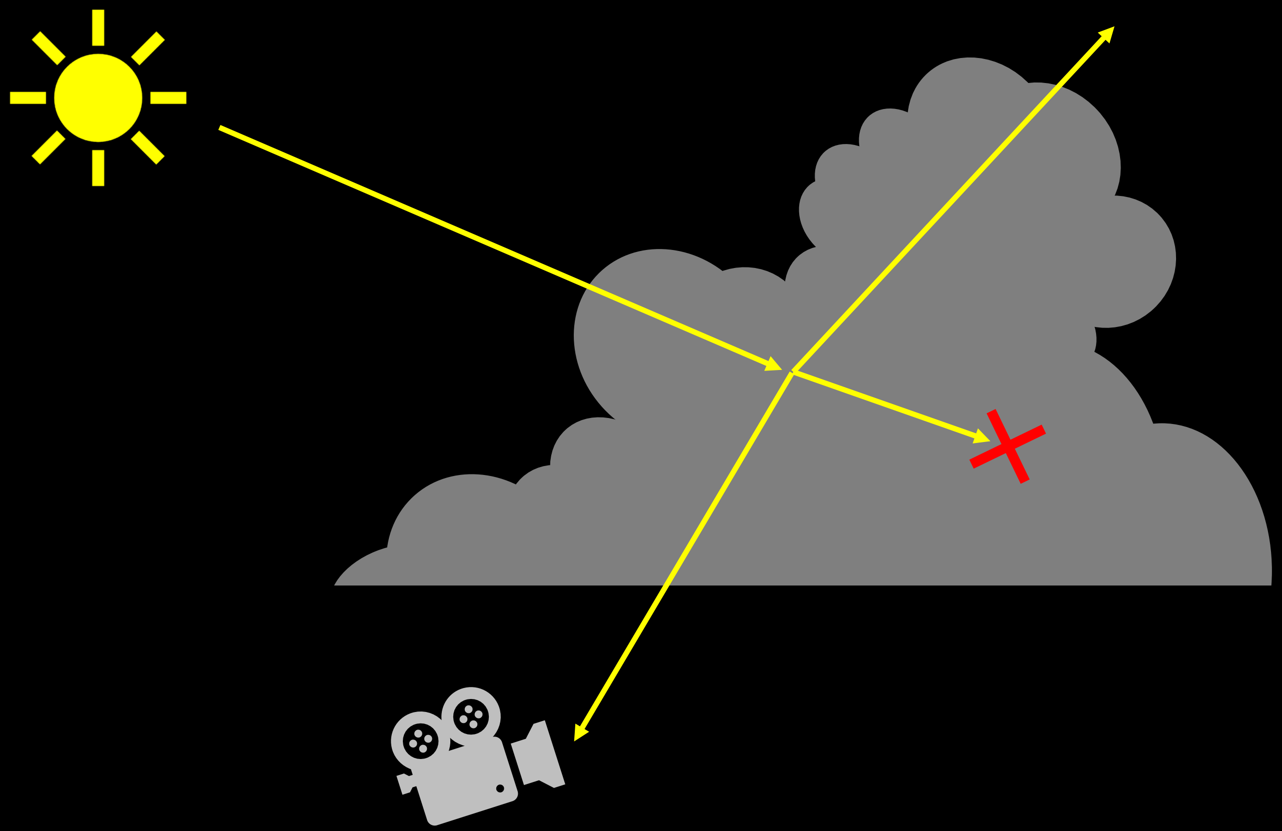


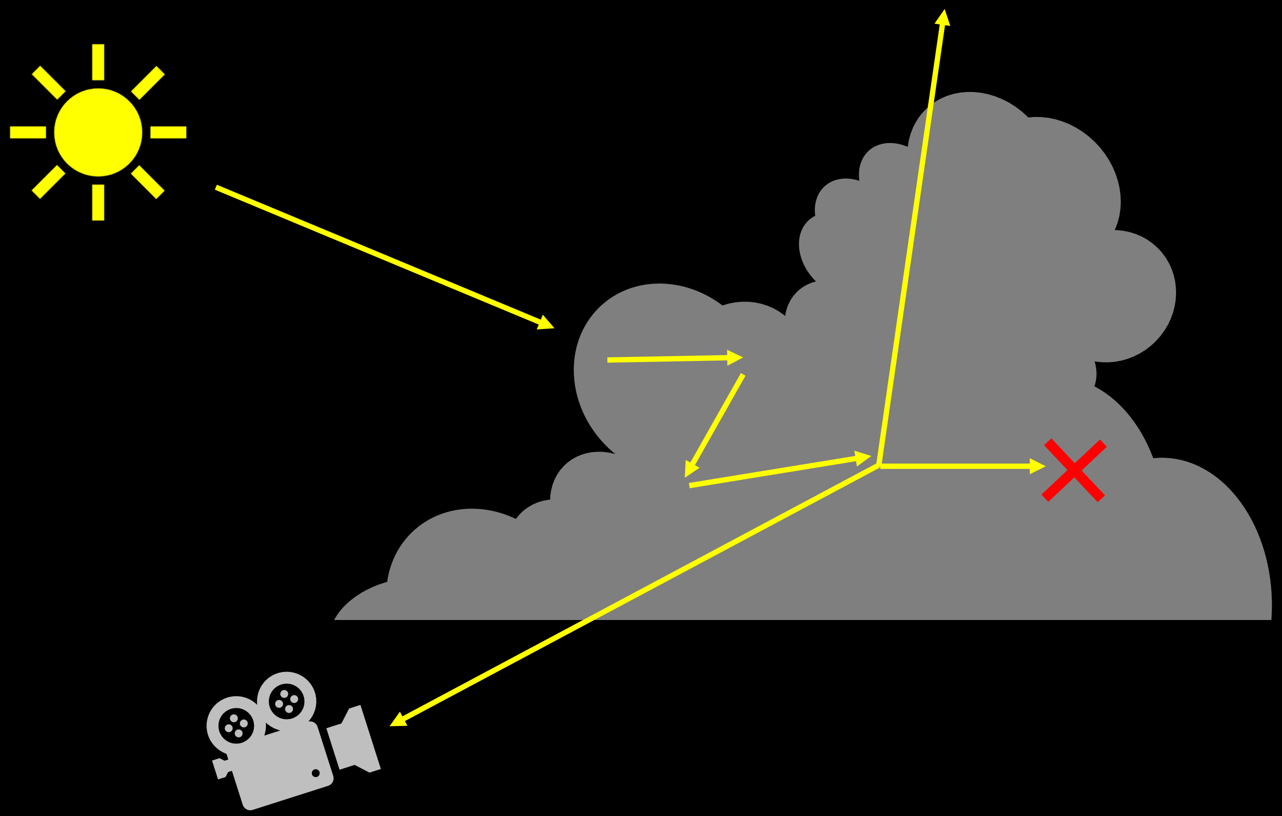

256m

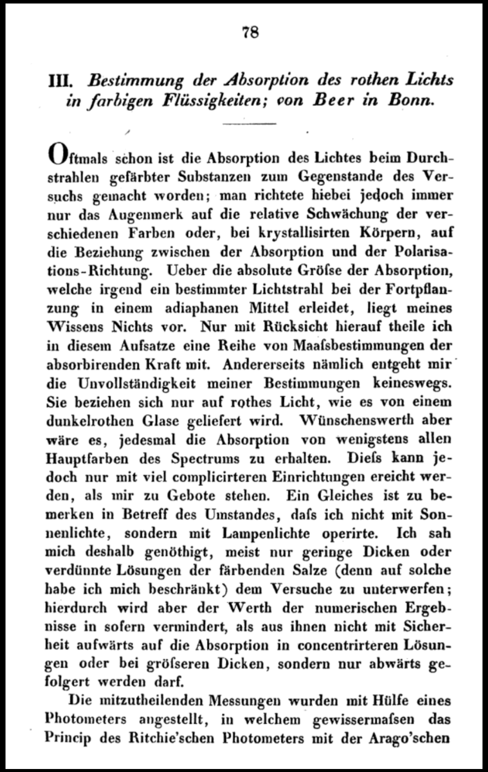


Light Energy = Direct Scattering + Ambient Scattering

```
Direct Scattering = (Transmittance * Primary Scattering Phase) + (Multiple Scattering * Secondary Scattering Phase)
```







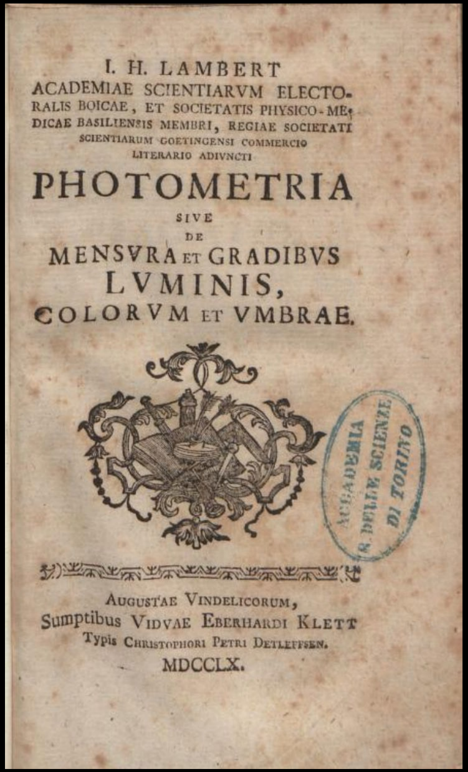
78

III. *Bestimmung der Absorption des rothen Lichts in farbigen Flüssigkeiten; von* ***Beer*** *in Bonn.*

Oftmals schon ist die Absorption des Lichtes beim Durchstrahlen gefärbter Substanzen zum Gegenstande des Versuchs gemacht worden; man richtete hiebei jedoch immer nur das Augenmerk auf die relative Schwächung der verschiedenen Farben oder, bei krystallisirten Körpern, auf die Beziehung zwischen der Absorption und der Polarisations-Richtung. Ueber die absolute Gröfse der Absorption, welche irgend ein bestimmter Lichtstrahl bei der Fortpflanzung in einem adiaphanen Mittel erleidet, liegt meines Wissens Nichts vor. Nur mit Rücksicht hierauf theile ich in diesem Aufsatze eine Reihe von Maafsbestimmungen der absorbirenden Kraft mit. Andererseits nämlich entgeht mir die Unvollständigkeit meiner Bestimmungen keineswegs. Sie beziehen sich nur auf rothes Licht, wie es von einem dunkelrothen Glase geliefert wird. Wünschenswerth aber wäre es, jedesmal die Absorption von wenigstens allen Hauptfarben des Spectrums zu erhalten. Diefs kann jedoch nur mit viel complicirteren Einrichtungen ereicht werden, als mir zu Gebote stehen. Ein Gleiches ist zu bemerken in Betreff des Umstandes, dafs ich nicht mit Sonnenlichte, sondern mit Lampenlichte operirte. Ich sah mich deshalb genöthigt, meist nur geringe Dicken oder verdünnte Lösungen der färbenden Salze (denn auf solche habe ich mich beschränkt) dem Versuche zu unterwerfen; hierdurch wird aber der Werth der numerischen Ergebnisse in sofern vermindert, als aus ihnen nicht mit Sicherheit aufwärts auf die Absorption in concentrirteren Lösungen oder bei gröfseren Dicken, sondern nur abwärts gefolgert werden darf.

Die mitzutheilenden Messungen wurden mit Hülfe eines Photometers angestellt, in welchem gewissermafsen das Princip des Ritchie'schen Photometers mit der Arago'schen

Augustus Beer
(1852)



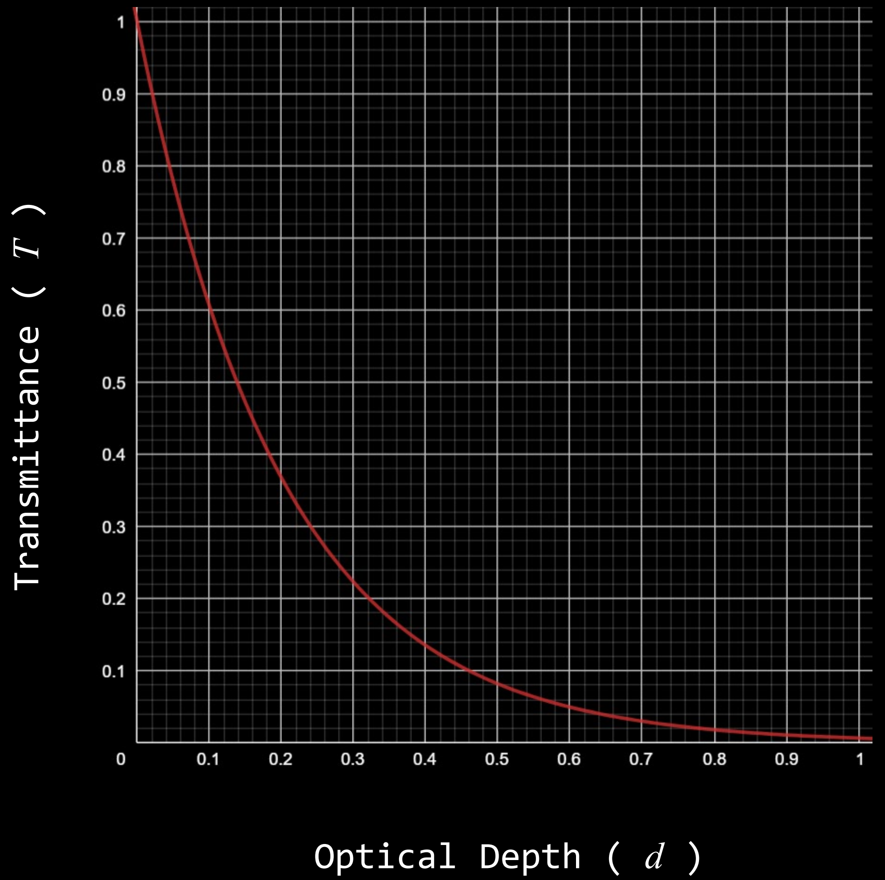
I. H. LAMBERT
ACADEMIAE SCIENTIARVM ELECTORALIS BOICAE, ET SOCIETATIS PHYSICO-MEDICAE BASILIENSIS MEMBRI, REGIAE SOCIETATI SCIENTIARUM GOETINGENSI COMMERCIO LITERARIO ADIVNCTI

PHOTOMETRIA
SIVE
DE
MENSVRA ET GRADIBVS LVMINIS, COLORVM ET VMBRAE.

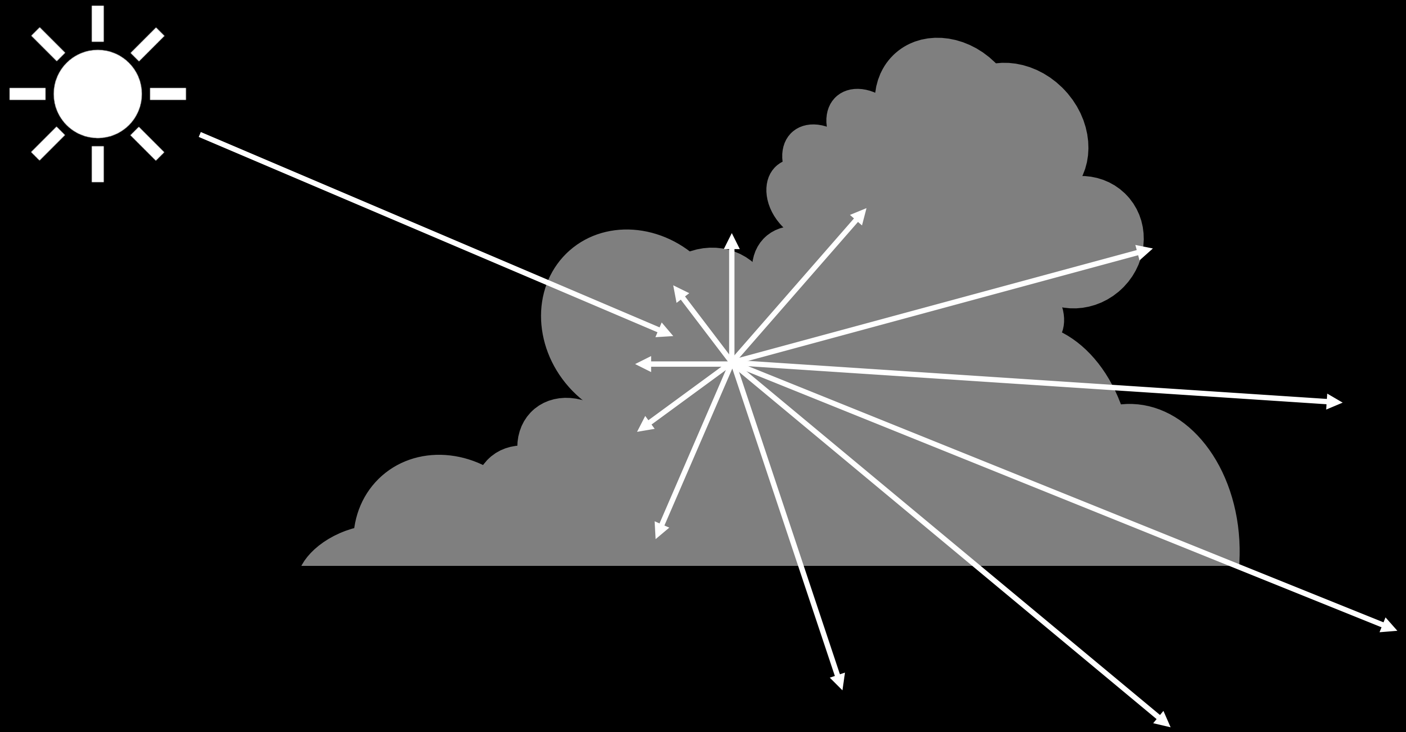
AUGUSTAE VINDELICORUM,
Sumptibus VIDVAE EBERHARDI KLETT
Typis CHRISTOPHORI PETRI DETLEFFSEN.
MDCCLX.

J.H. Lambert
(1760)

$$T = e^{-d}$$



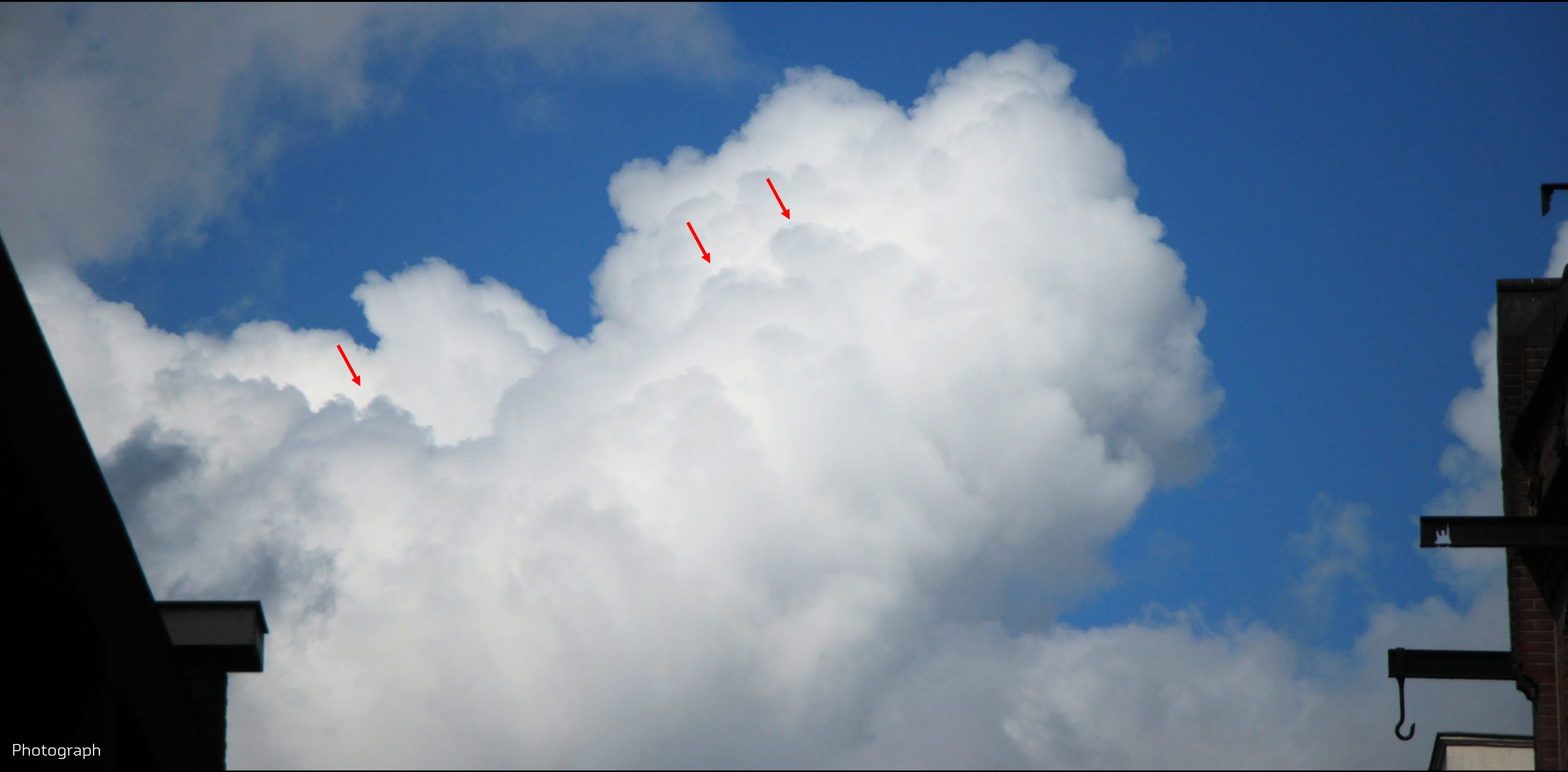

In-Engine Renders



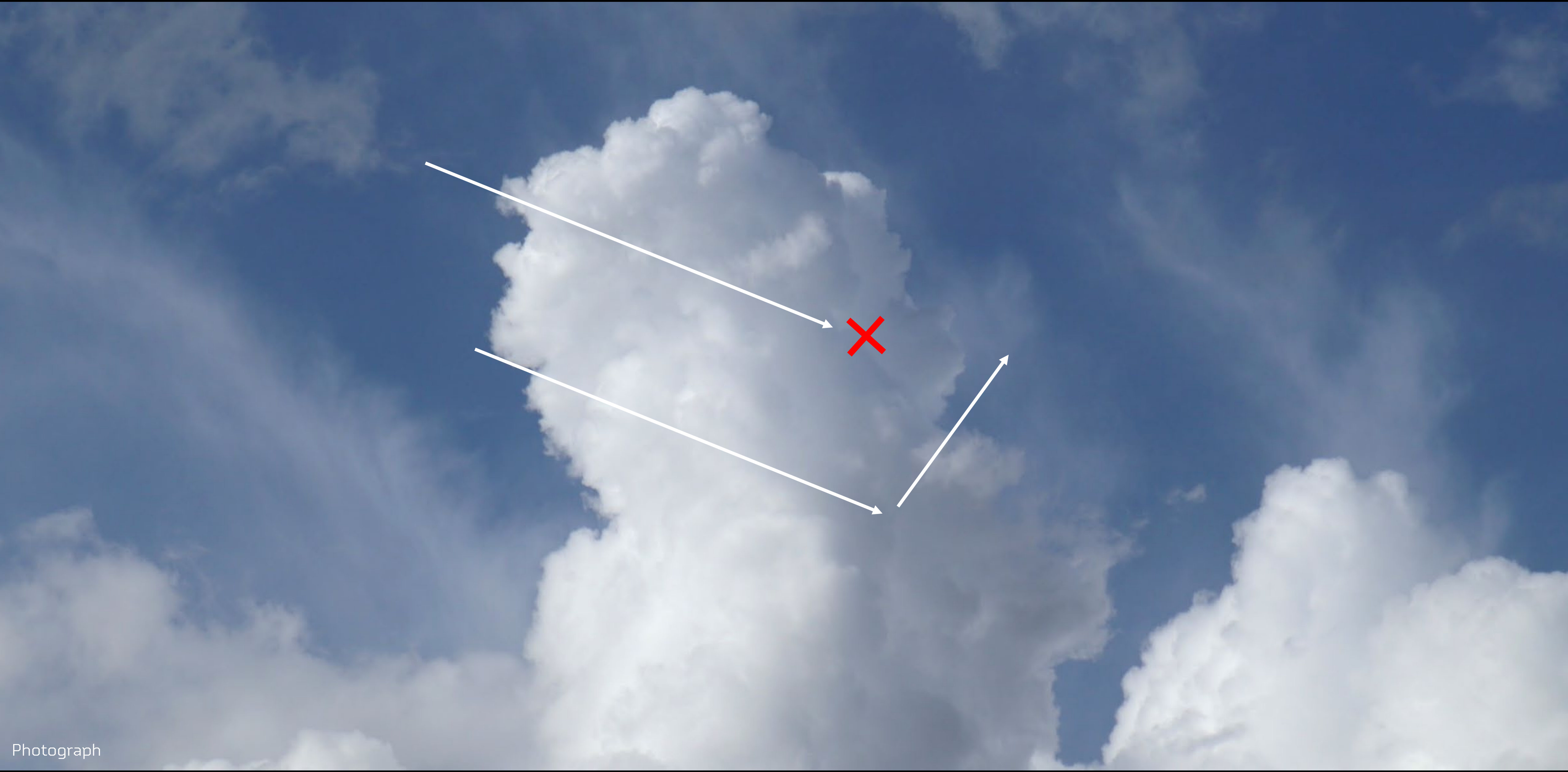
```
float HenyeyGreenstein(float inCosAngle, float inG)
{
    float num = 1.0 - inG * inG;
    float denom = 1.0 + inG * inG - 2.0 * inG * inCosAngle;
    float rsqrt_denom = rsqrt(denom);
    return num * rsqrt_denom * rsqrt_denom * rsqrt_denom * (1.0 / (4.0 * M_PI));
}
```


In-Engine Render

Photograph



Photograph



Photograph

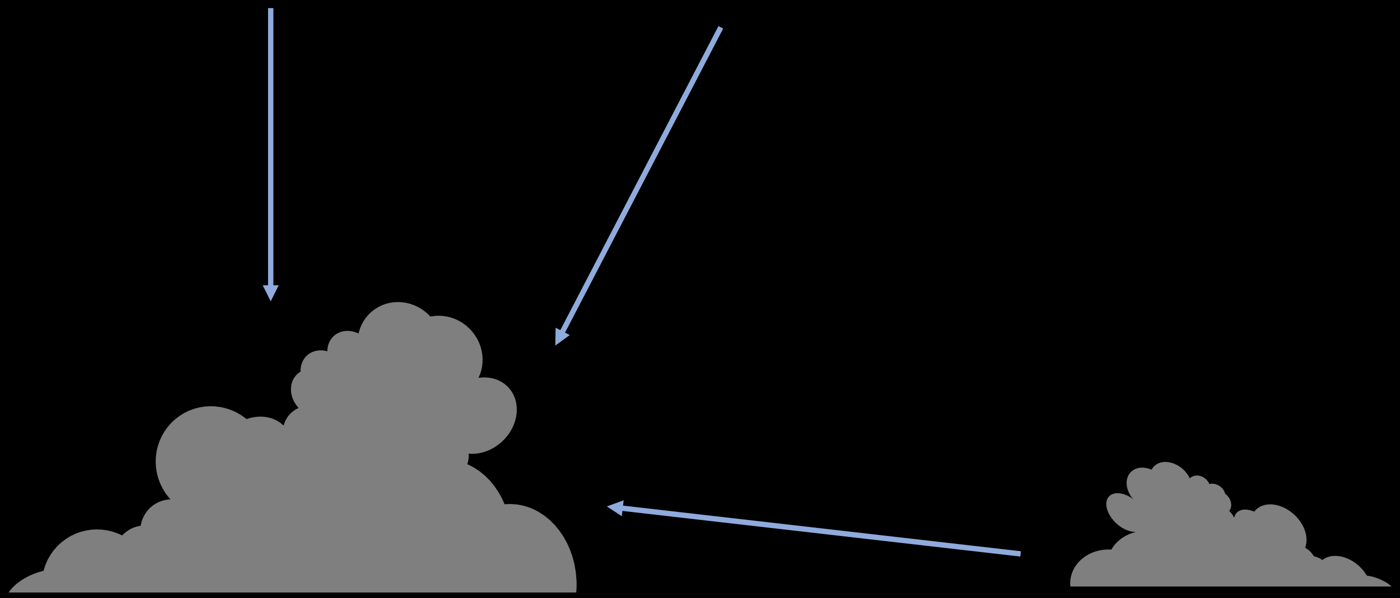

In-Engine Render

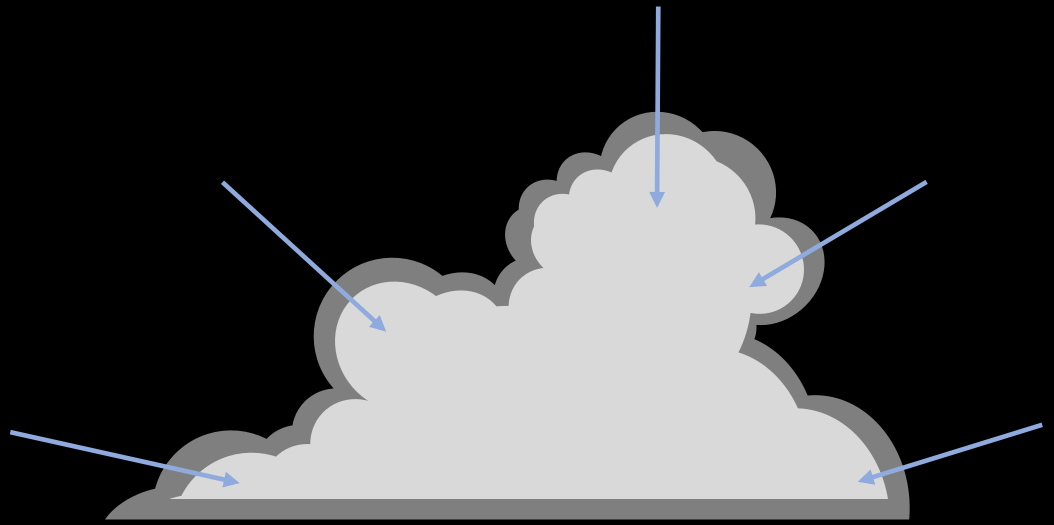
```
float ms_volume = Remap(dimensional_profile* step_size, 0.1, 1.0, 0.0, 1.0)

ms_volume *= pow(attenuated_light, cMultipleScatteringDepthPower);

ms_volume *= pow(height_fraction, cMultipleScatteringHeightPower);
```


In-Engine Render


In-Engine Render

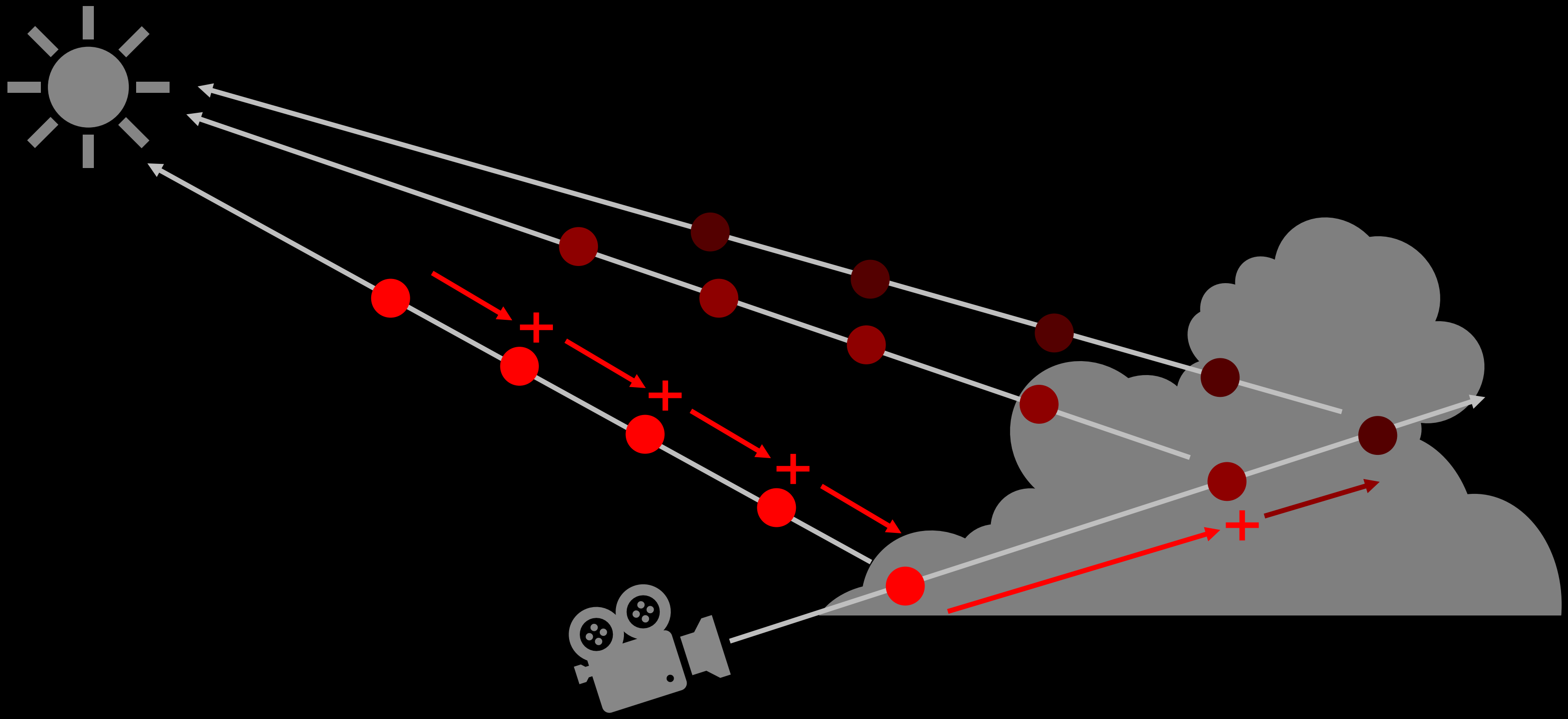

In-Engine Render



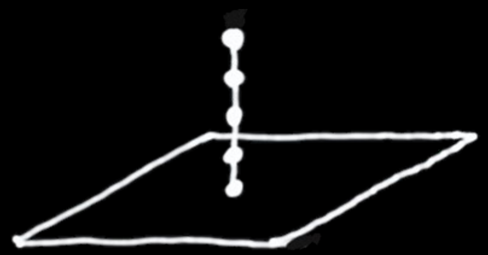


```
float ambient_scattering = pow(1.0 - dimensional_profile, 0.5);
```


In-Engine Render

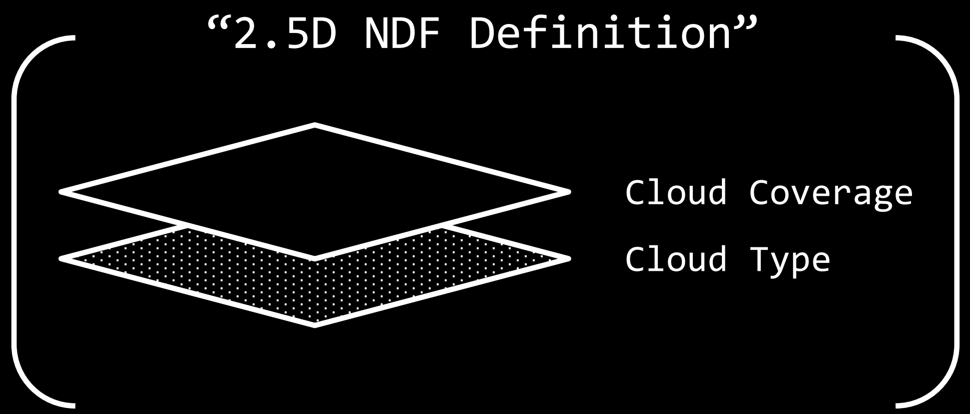


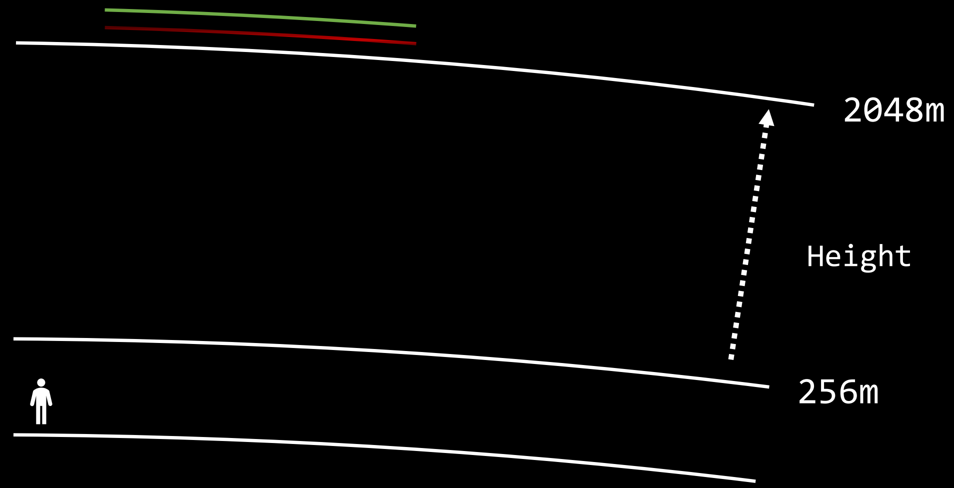
```
// Accumulate light_absorption from sampled density
light_absorption += sampled_density * (1.0 - light_absorption);

// Accumulate energy and attenuate based on depth in the cloud along the view ray
light_intensity += (light_energy * sampled_density * (1.0 - light_absorption));

// Accumulate energy and attenuate based on depth in the cloud along the view ray
float3 color = float4(direct_intensity * sun_color + amb_intensity * amb_color);
float alpha = light_absorption;
```



2.5-D Model

Photograph

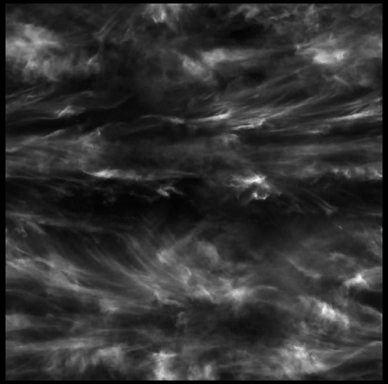

"2.5D NDF Definition"
Cloud Coverage
Cloud Type


2048m
Height
256m

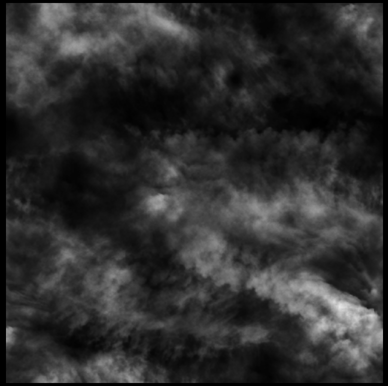
Type

0 → 1

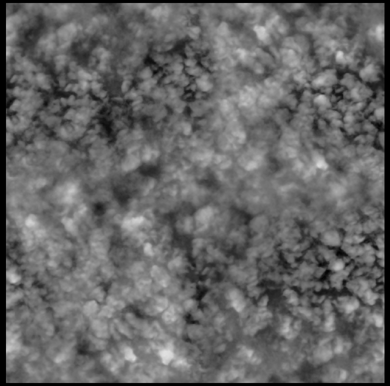
cr_streaky



cr_wispy

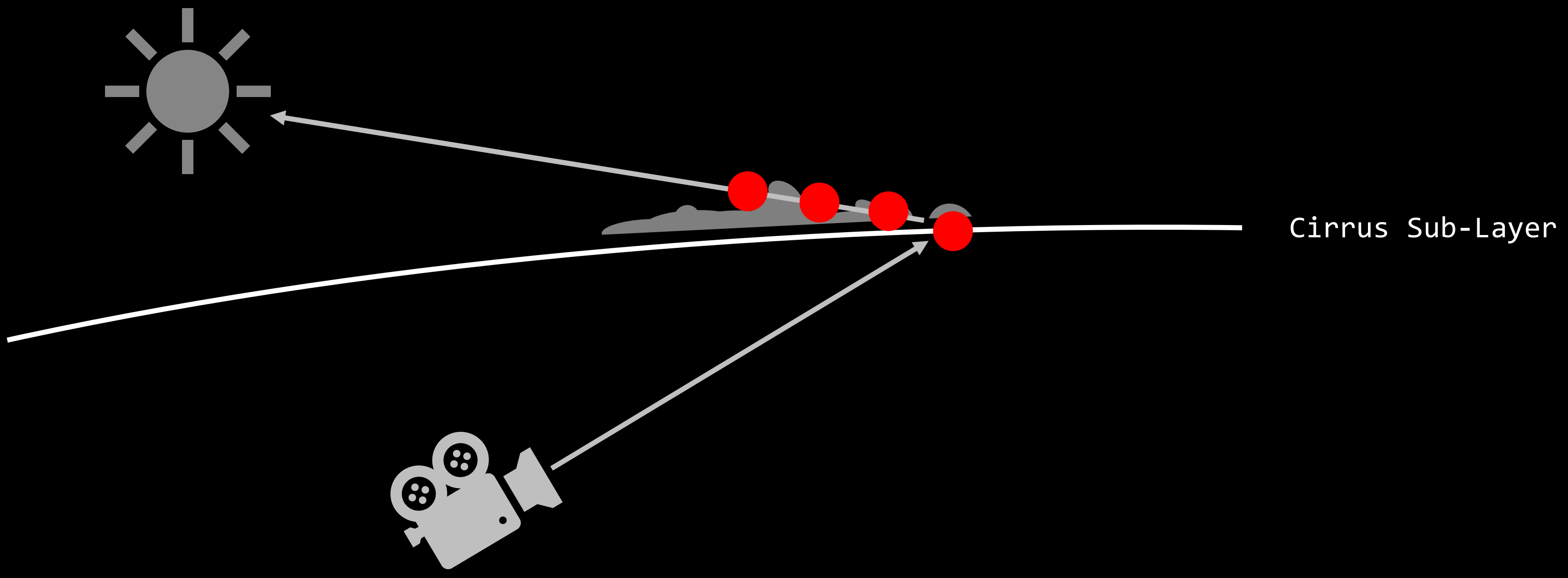


cr_round

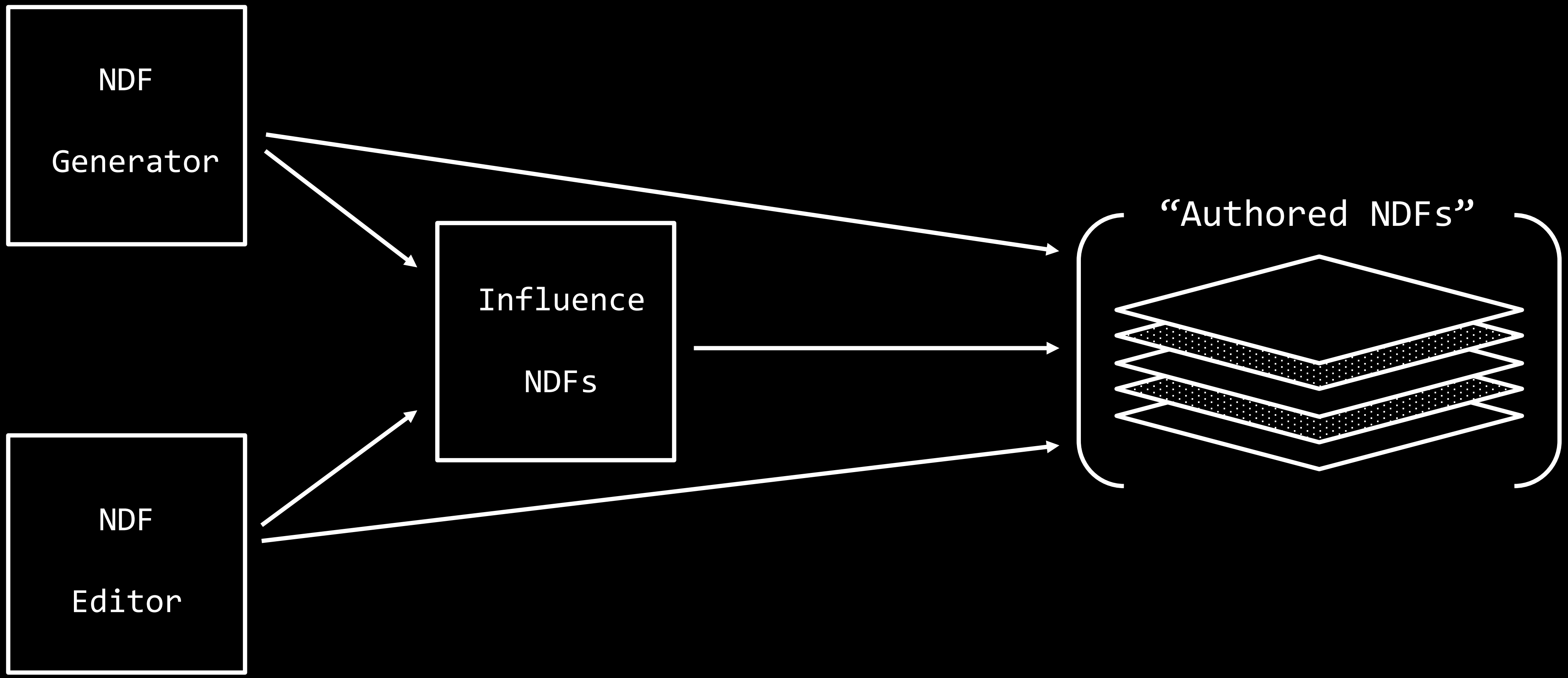


```
float density = ValueRemap(cloud_type, 0.5, 1.0, ValueRemap(cloud_type, 0.0, 0.5, cr_streaky, cr_wispy), cr_round);
density = pow(density, 1.0 - ValueRemap(cloud_coverage, 0.0, 1.0, -0.9, 0.9));
density *= ValueRemap(pow(cloud_coverage, 3.0), 0.0, 0.5, 0.0, 1.0);
```

In-Engine Render


Cirrus Sub-Layer

In-Engine Render


NDF Generator
Influence NDFs
NDF Editor
"Authored NDFs"


Weather
In-Engine Render


Weather
In-Engine Render

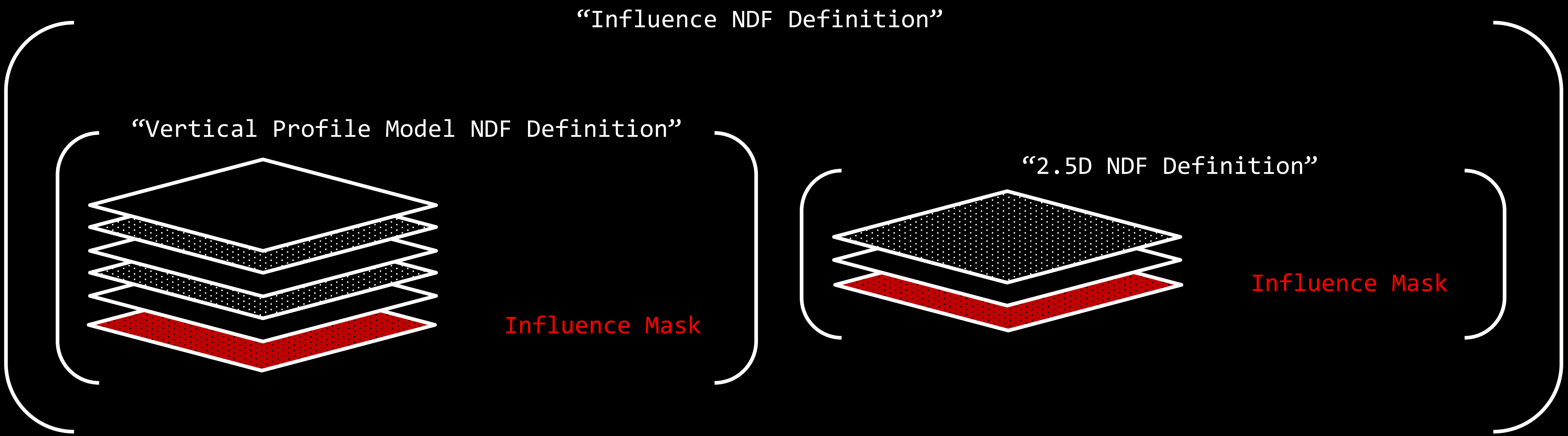
Houdini Viewport / In-Engine Render

NDF Generator

NDF Editor

In-Engine Renders


Weather
In-Engine Render

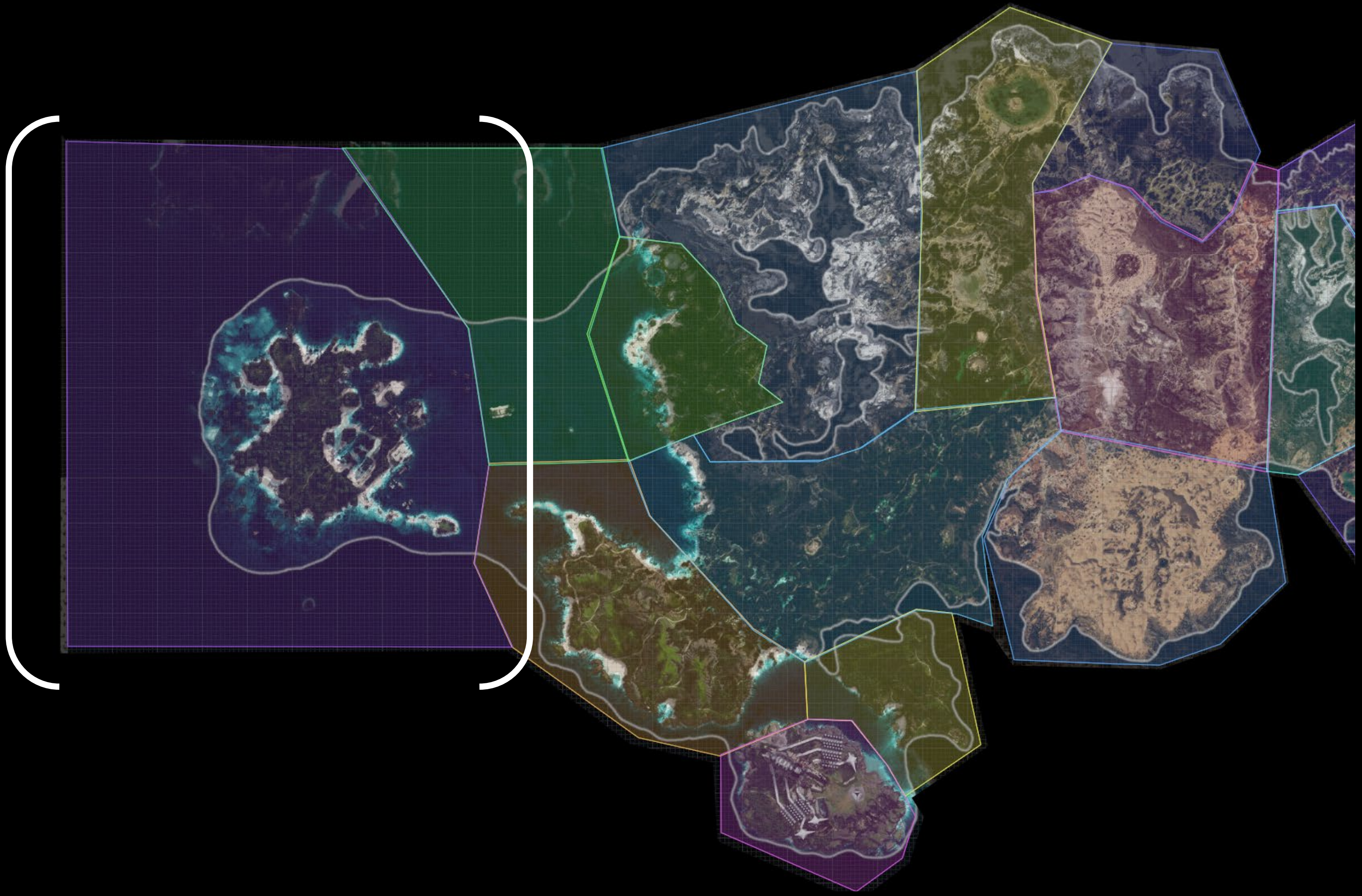

“Influence NDF Definition”
“Vertical Profile Model NDF Definition”
Influence Mask
“2.5D NDF Definition”
Influence Mask


Weather
In-Engine Render

San Francisco

Cumulus Procedural NDFs
+
Regional Influence NDFs

Stratocumulus Procedural NDFs
+
Regional Influence NDFs



NUBIS EVOLVED / Skies / Authoring / Influence NDFs

Calm Cloudy Weather State
Partly-Cloudy Cumulus Cloudscape
Regional Influence NDFs

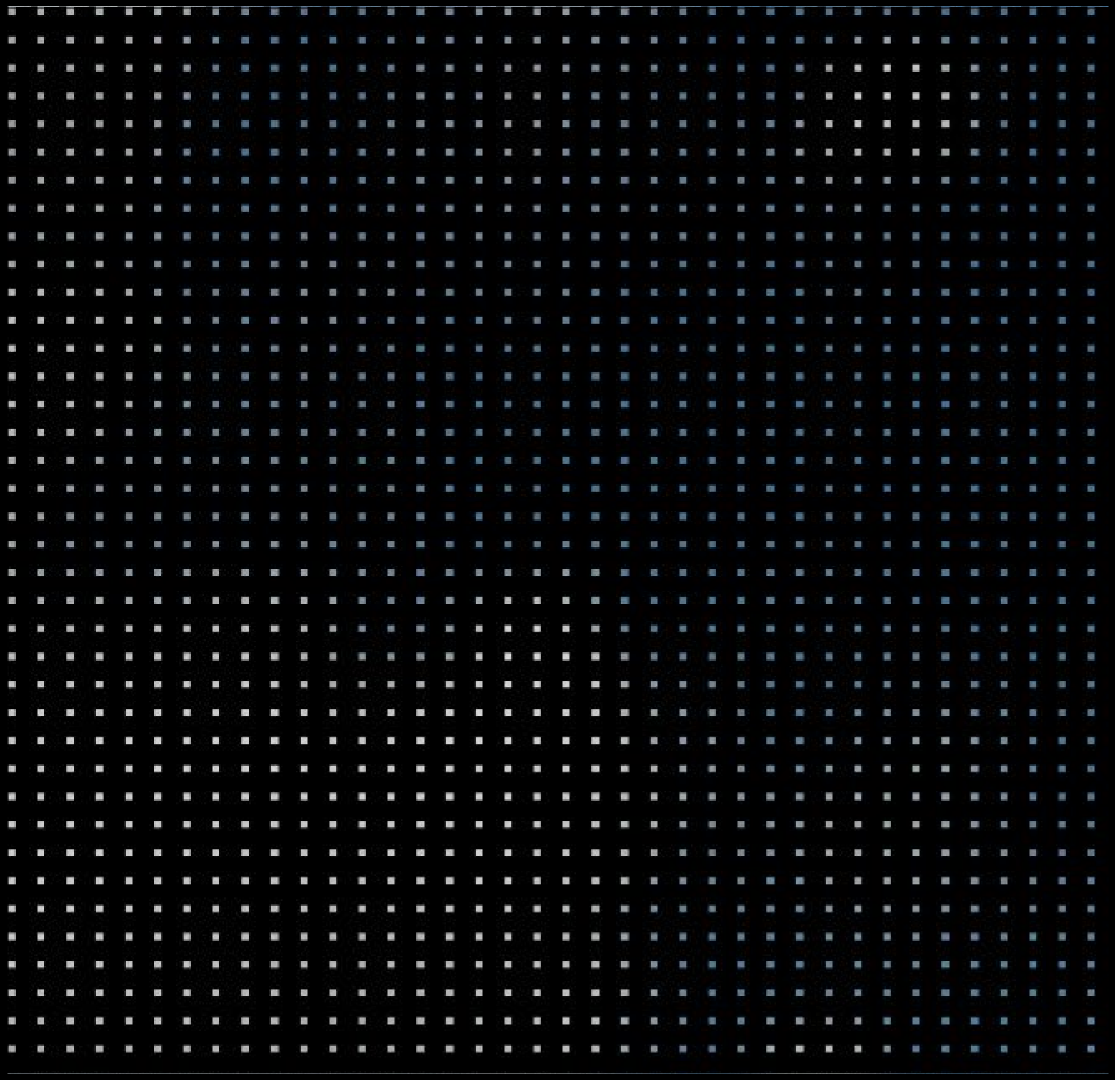
Stormy Weather State
Stormy Cumulus Cloudscape
Regional Influence NDFs

Calm Clear Weather State
Light Cirrus Cloudscape
Regional Influence NDFs

Current Weather


In-Engine Render


In-Engine Render




Time 434 µs
Scalpel GPU Profiler
Clouds
Time 434 µs
Ticks 43,412
Start Tick 562,100
End Tick 605,512
GPU Profile
2,849 Events
Filter
Frame 10 / 10
Extra fine
Profile precision
Clouds
Time Avg / Median 329 µs / 401 µs
Wait Flip
WaitSync
WaitUntilSafeForRendering 1
In-Engine Render

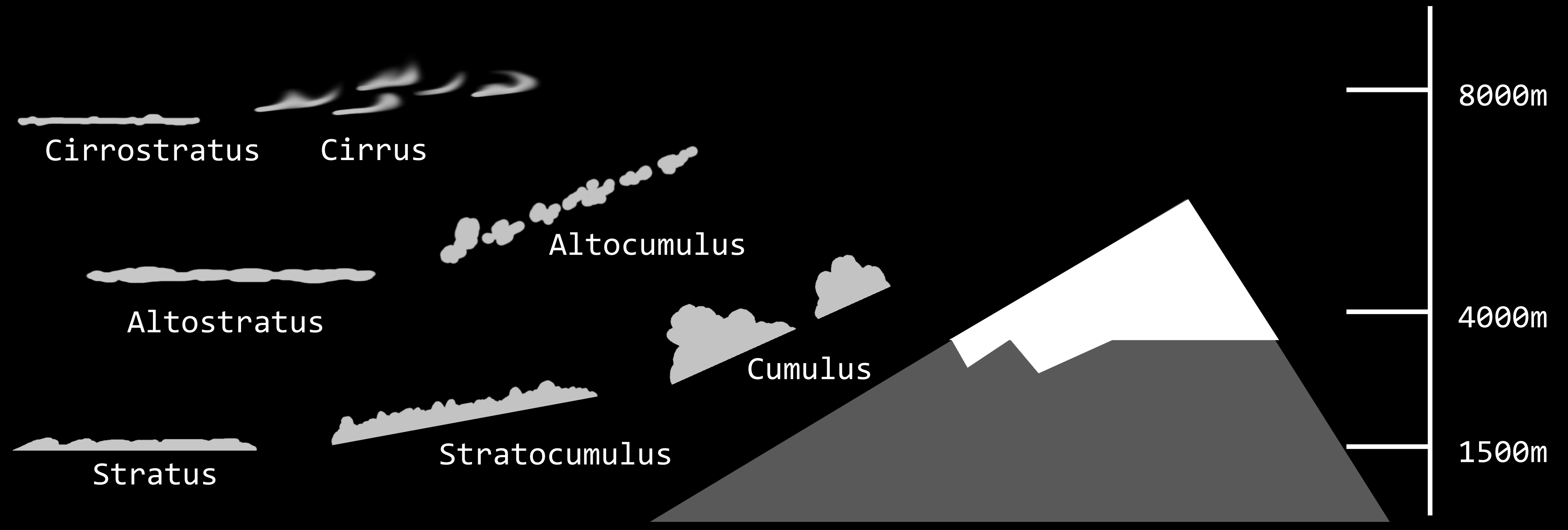
In-Engine Renders

- Open World
- Performant
- Detailed

In-Engine Render

NUBIS EVOLVED / Environments / Goals

Paintings by Albert Bierstadt


Cirrostratus
Cirrus
Altocumulus
Altostratus
Cumulus
Stratus
Stratocumulus
8000m
4000m
1500m

heart of the clouds and then fall back, some distance away, in the unusual form of a shower of batrachians.

The tornado that rages in North America, takes on a terrifying aspect, affecting regions some hundreds of yards wide by fifteen, thirty, or even sixty miles long.

Sedentary clouds

For respite from these apocalyptic spectacles, and to make peace with the cloud world, let us go to the mountains.

On calm days in the fine season, the summit of a peak or an isolated mountain may be covered by a hood of a cloud which sends its peak rather high into the clear sky. This immobile cumulus seems to keep watch as long as the rise of humid and warm air from the valley provides sufficient expansion to cause condensation; that is from mid-morning to late afternoon.

If the condensation level is below the mountain peak, the mountain may be surrounded by a collar of clouds, similar to a giant, immobile smoke-ring.

It may also happen that the mountains unfurl to the wind a banner of clouds caught at their peaks, continually reforming only to disperse some tens or hundreds of yards farther on. As a consequence of the ascending motion of the threads of air along the slopes, the formation of this cloud requires certain specific conditions: the condensation level must correspond to that at the summit of the mountain and, on the other hand, a strong wind must cause the rapid ascent of air along a slope; then, when it reaches the summit, this air abandons its surplus humidity in the form of drops. Afterward the air, hugging in its course the profile of the mountain, redescends on the other side. It is here that the opposite phenomenon intervenes: while the ascent caused

Orographic clouds

Orographic clouds are sometimes detached from the projection that causes them. The spectacle of these

107

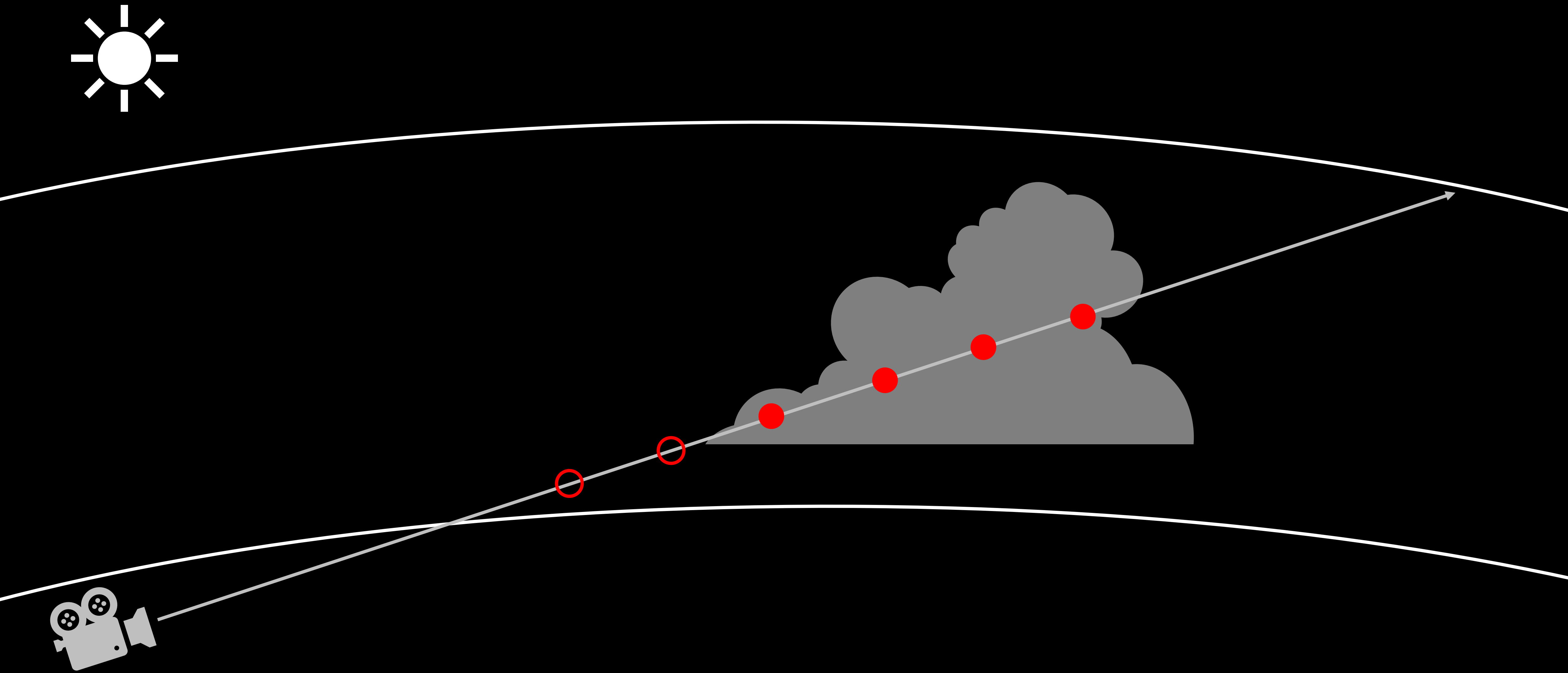
Cloud banner on the side of a mountain

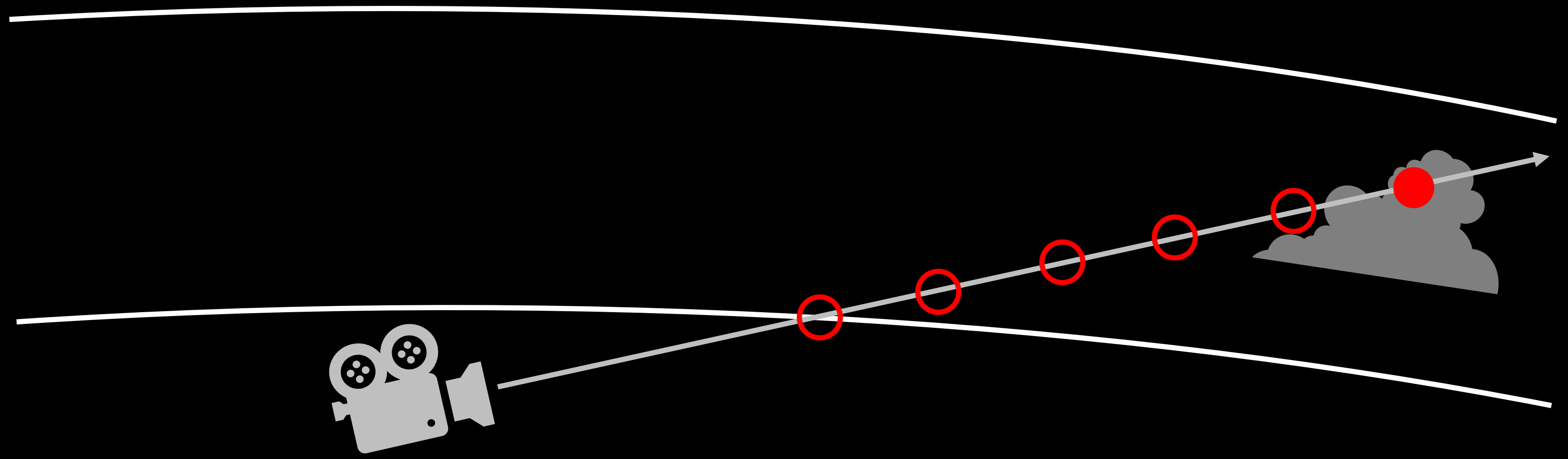
immobile clouds, anchored in the wind, is quite surprising. Elongated at the two extremities, these clouds in the form of a giant lentil (they are called *alto-cumulus lenticularis)* form at the summit of waves caused by the obstacle of the mountain, in the wind's eye, however little assistance humidity and temperature conditions may offer.

These invisible waves, on the cloud crests, can extend horizontally over several miles, materialized in hyphens or ellipses scattered through the sky. Sometimes, when the air is overlaid with alternately humid and dry layers, superposed lenticular clouds, seen from the earth, give the impression of a pile of dishes which the imagination sometimes transforms into giant rotating saucers.

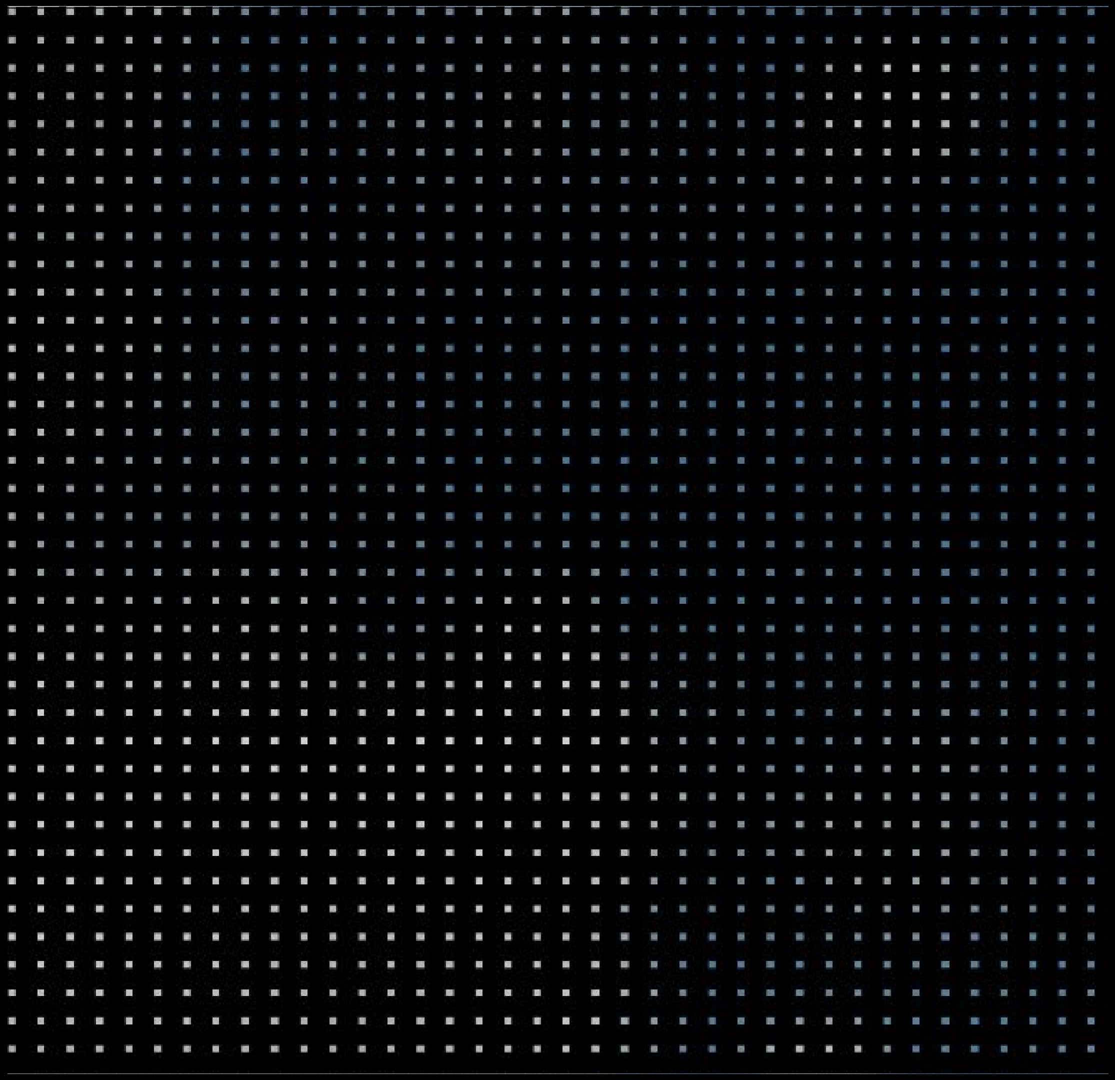
At New Amsterdam Island, lost in the South Indian Ocean, such cloud masses are often seen; they are due to simultaneous action of insular projection and contrasts of temperature existing between the hot currents of subtropical regions and cold currents of the Antarctic. The action of the latter on

108



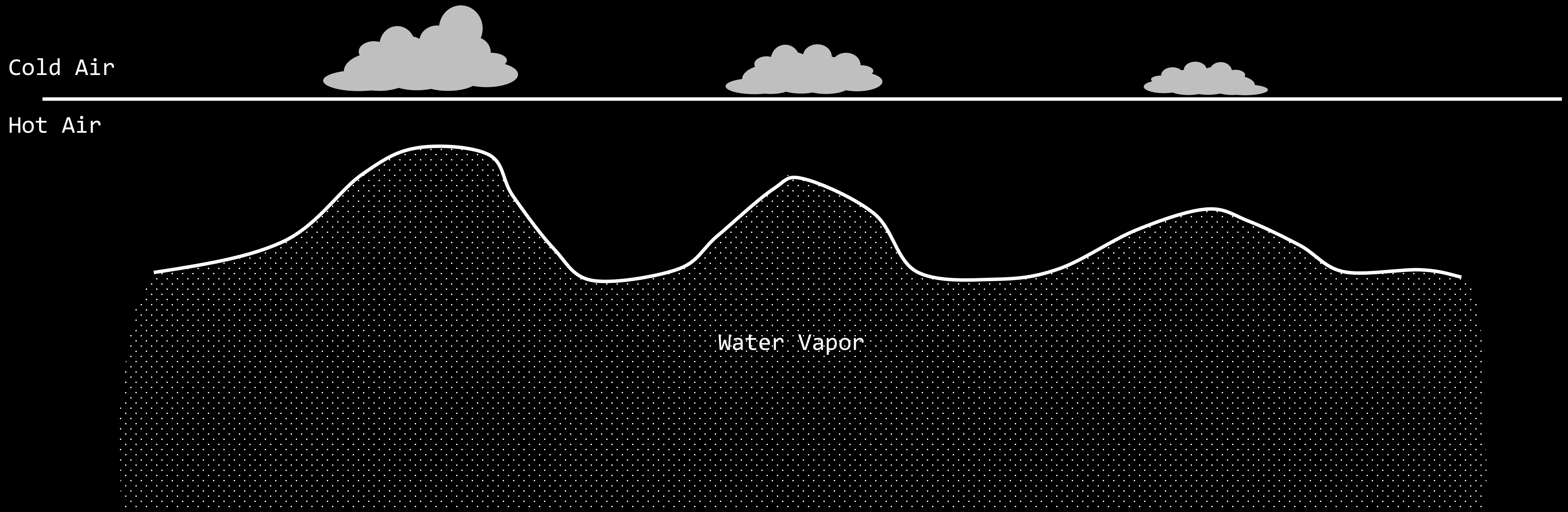


```
// Construct Dimensional Profile.
float cloud_coverage = GetCloudCoverageSample(sample_position); (1 Texture Read)
float vertical_profile = GetVerticalProfile(sample_position); (2 Texture Reads)
float dimensional_profile = vertical_profile * cloud_coverage; (1 Multiply)

// Test if this is empty space and exit.
if ( dimensional_profile < density_threshold)
        return 0.0;
```

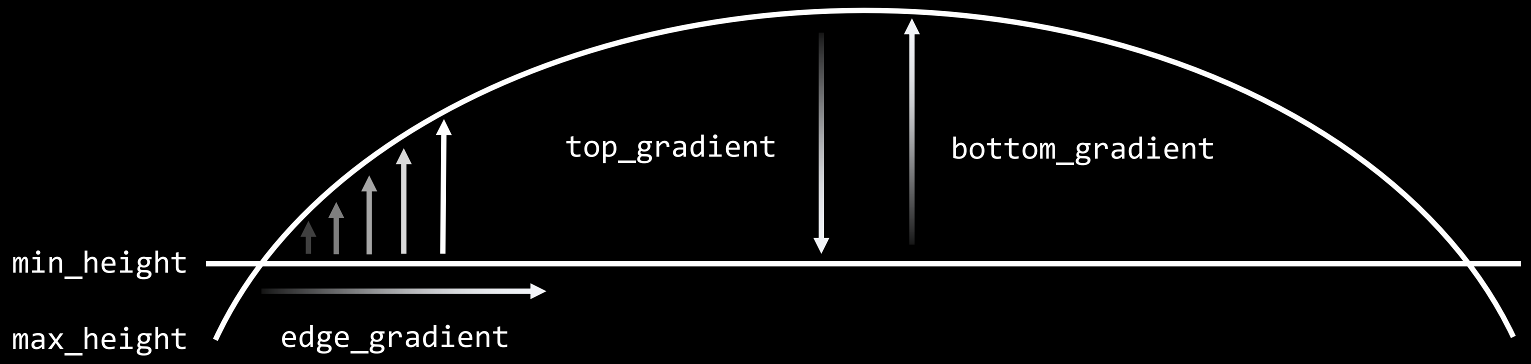


Orographic clouds

Photograph

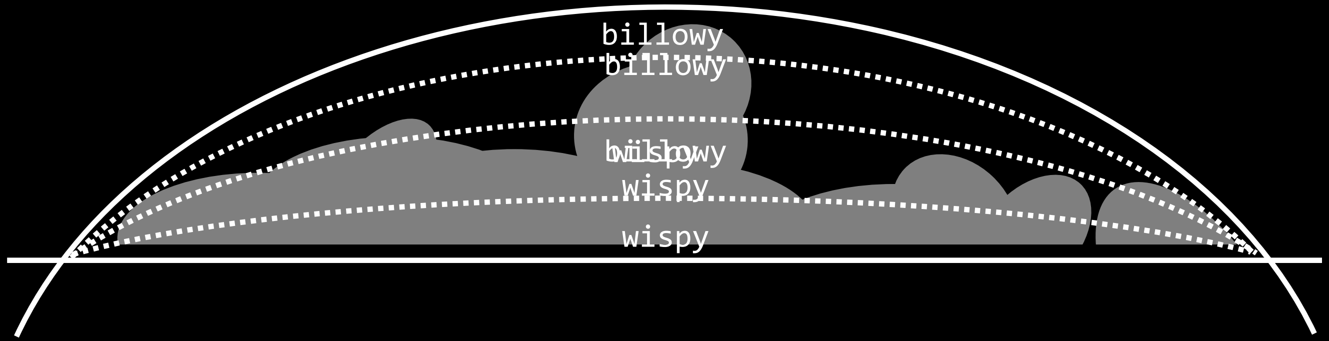

Cold Air
Hot Air
Water Vapor

Envelope Model




```
// Construct Dimensional Profile.
float height_fraction = Remap(height, min_height, max_height, 0.0, 1.0);
float top_gradient = pow( 1.0 - height_fraction, 1.5);
float bottom_gradient = pow( height_fraction, 2.0);
float edge_gradient = Remap( sample_height, 0.0, 35.0, 1.0, 0.0);
float dimensional_profile = bottom_gradient * top_gradient * edge_gradient;
```


In-Engine Render




```
float noise_height_blend = Remap(height_fraction, cloud_type + 0.1, cloud_type - 0.1);
float composite = lerp(wispy_noise, billowy_noise,  noise_height_blend);
```

In-Engine Render

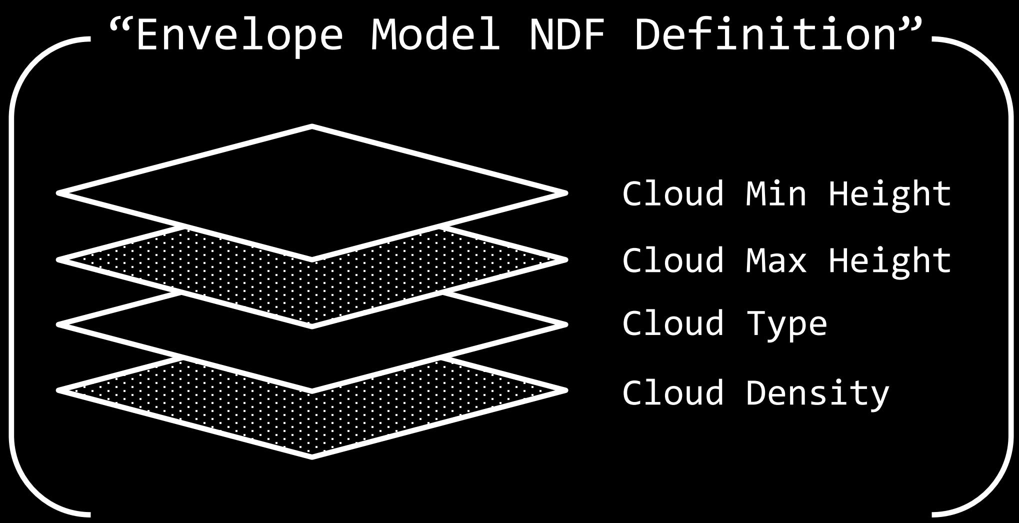
In-Engine Render

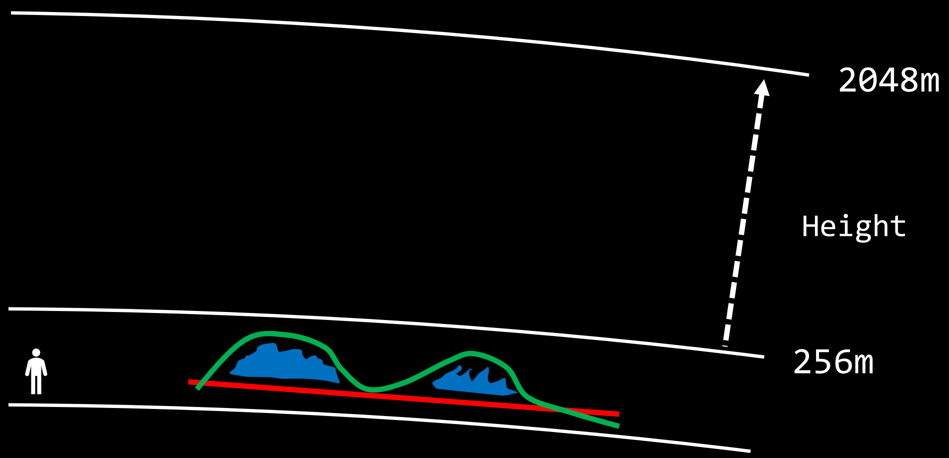
```
float3 noise_sample_pos = inSamplePosition + float3(0.0, 0.0, (1.0 - saturate((max_height – min_height) * 0.0125)) * 40.0 );
```

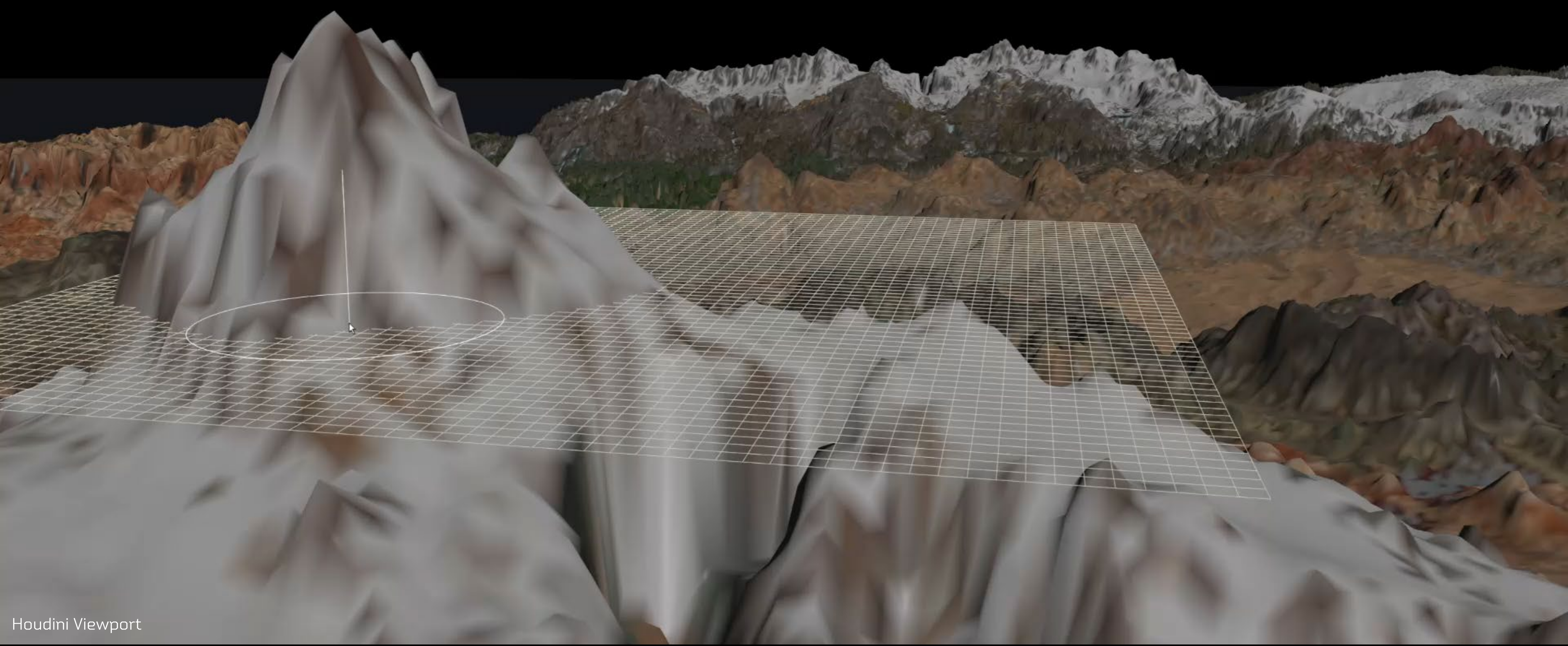
In-Engine Render


In-Engine Render

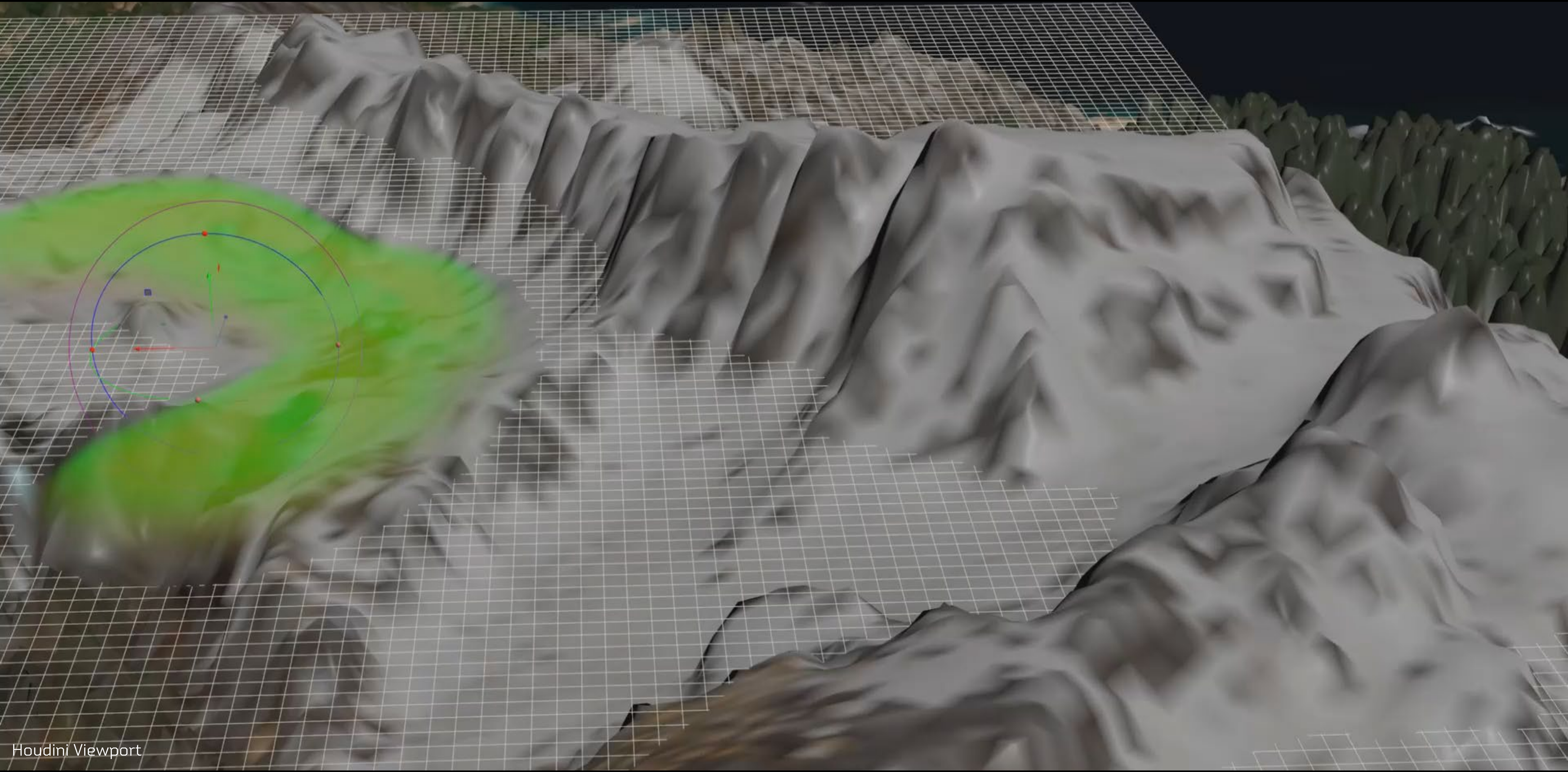

```
// Get cloud density
float cloud_density_sample = height_fraction * pow(saturate(noise_composite - (1.0 - dimensional_profile)), 0.27);

// The inverse edge signal is powered by three and used to fade off the edges of clouds in several places below this
float inv_edge_signal_pow_3 = pow(inv_edge_signal, 3.0);

// Define Samples
float cloud_density = cloud_density_sample;
float cloud_coarse_density = pow(ValueErosion(dimensional_profile, 0.04), 0.5) * inv_edge_signal_pow_3 * 5.0;
```


“Envelope Model NDF Definition”
Cloud Min Height
Cloud Max Height
Cloud Type
Cloud Density

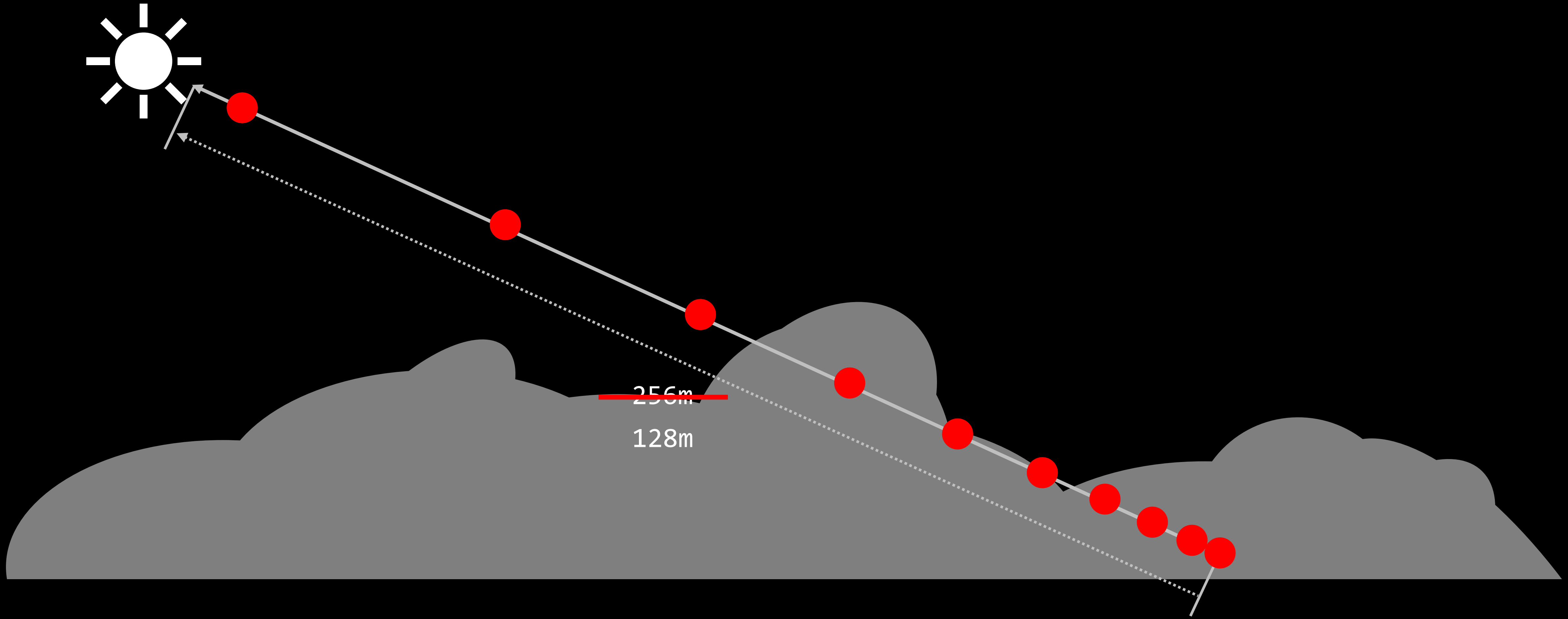

2048m
Height
256m



Houdini Viewport


Houdini Viewport


In-Engine Render

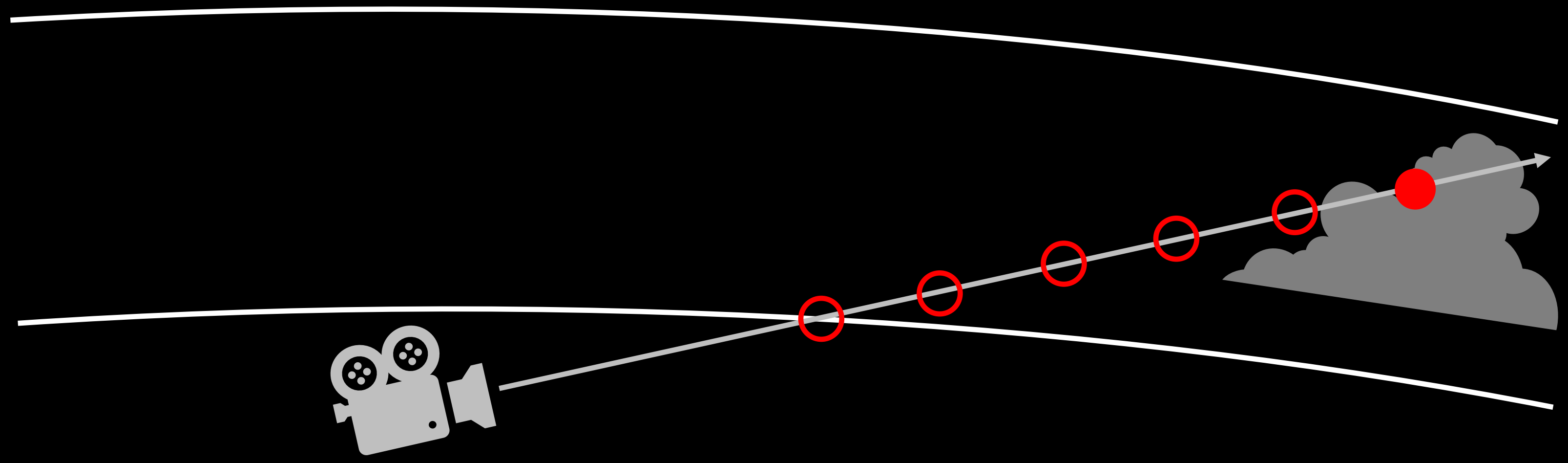

256m
128m

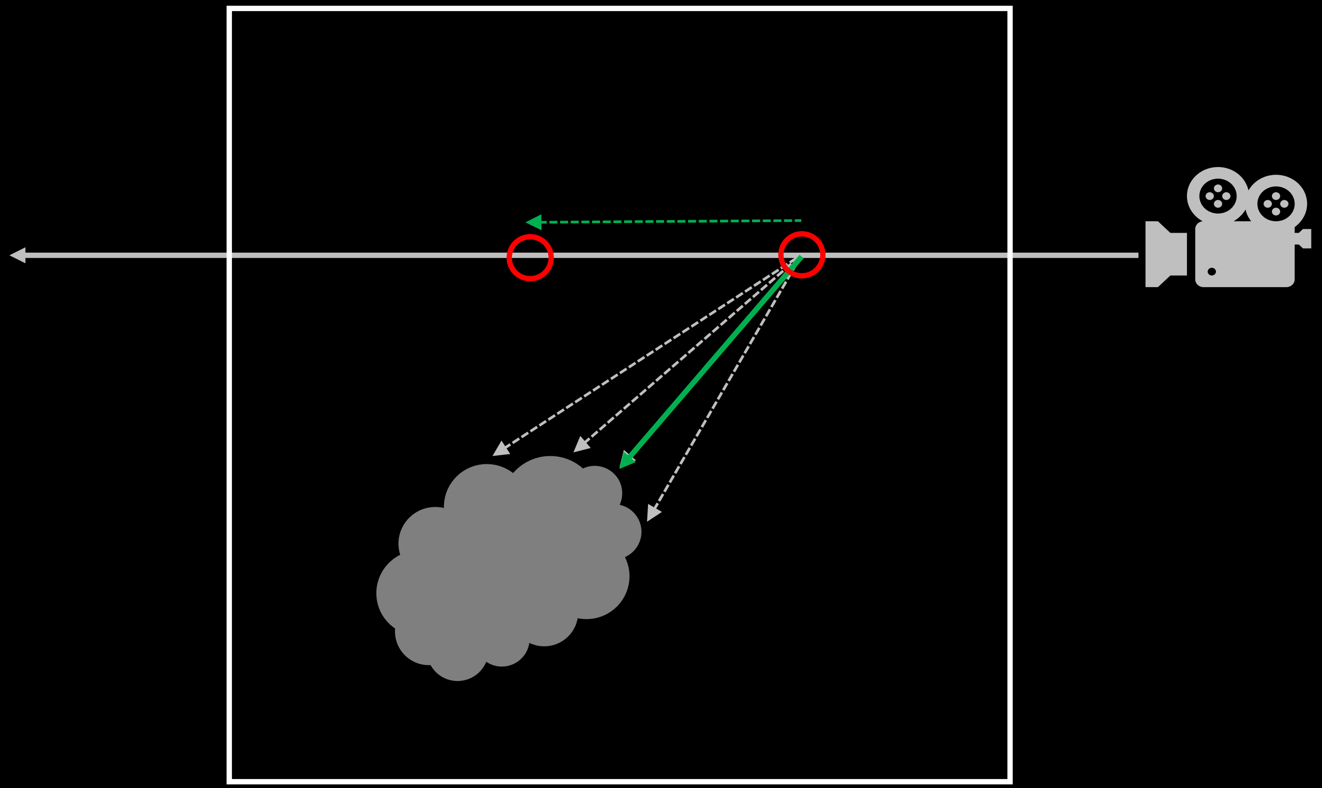
```
Light Energy = Direct Scattering + Ambient Scattering
```


```
// Calculate Transmittance
float transmittance = exp(-inSummedSamples);

// Get Long Distance Shadow Sample
float long_distance_shadow_sample = SampleLongDistanceShadowMap(inSamplePosition);

// Define Direct Scattering
float direct_scattering = transmittance * long_distance_shadow_sample;
```


```
// Get the height fraction so that we can reduce the ambient influence at the bottoms of envelope model clouds
float height_fraction = ValueRemap(inSamplePosition.z, min_height, max_height. 0.0, 1.0);

// Define Ambient Scattering
float ambient_scattering = pow(1.0 - saturate(cloud_coarse_density), 0.25) * height_fraction;
```

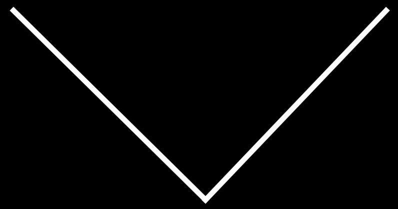

In-Engine Render





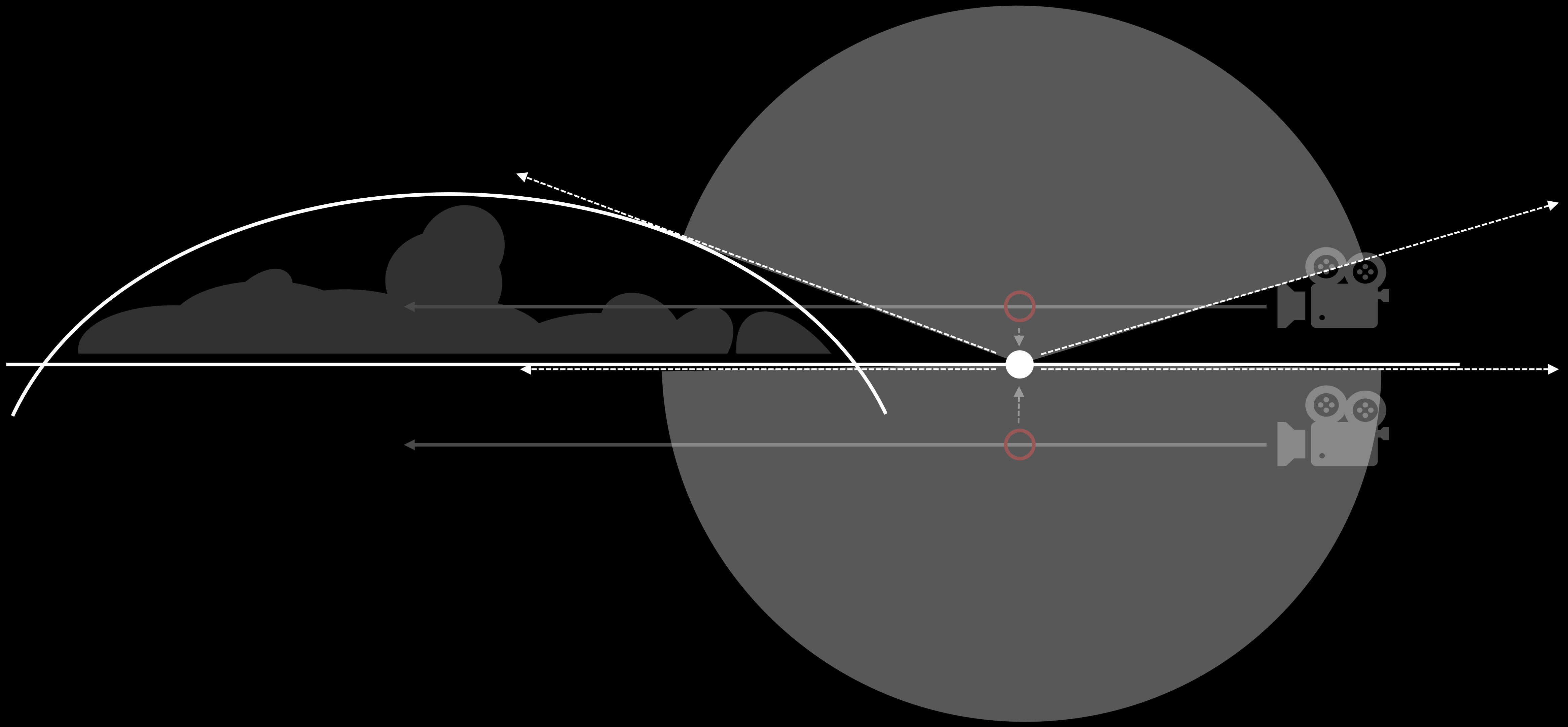
Sphere Tracing

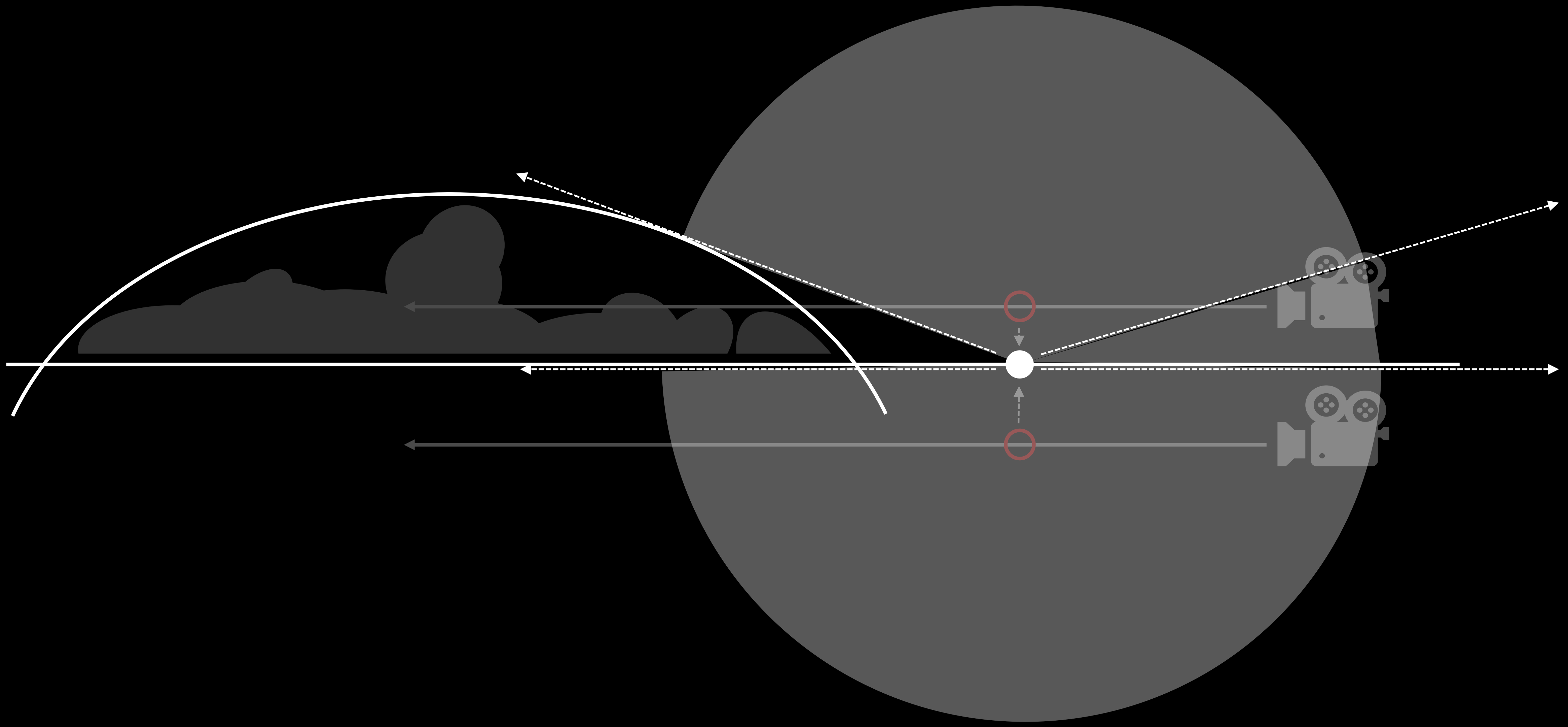
Hart, John C. 1995.
“Sphere Tracing: Simple Robust
Antialiased Rendering of Distance-Based
Implicit Surfaces”

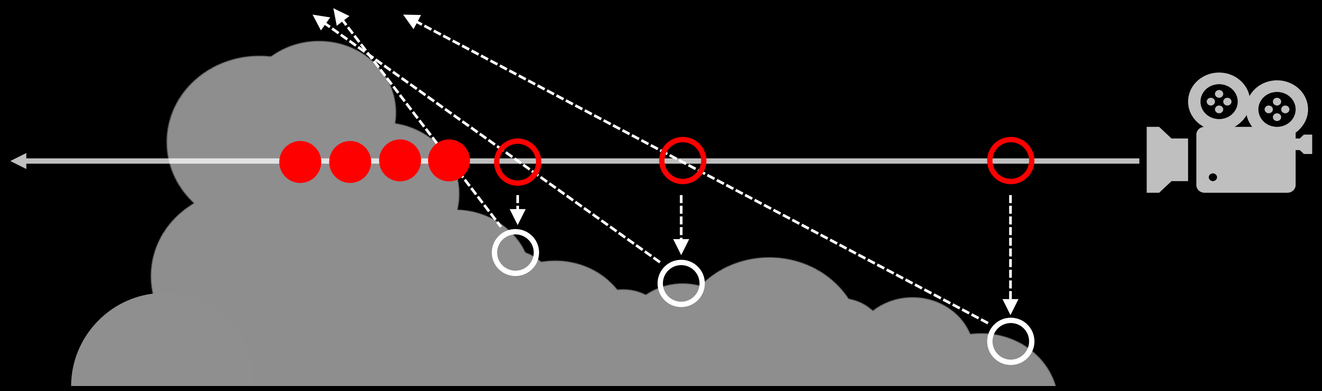


Cone Step Mapping

Dummer, Jonathan. 2006.
"Cone Step Mapping: An Iterative
Ray-Heightfield Intersection Algorithm."







In-Engine Render

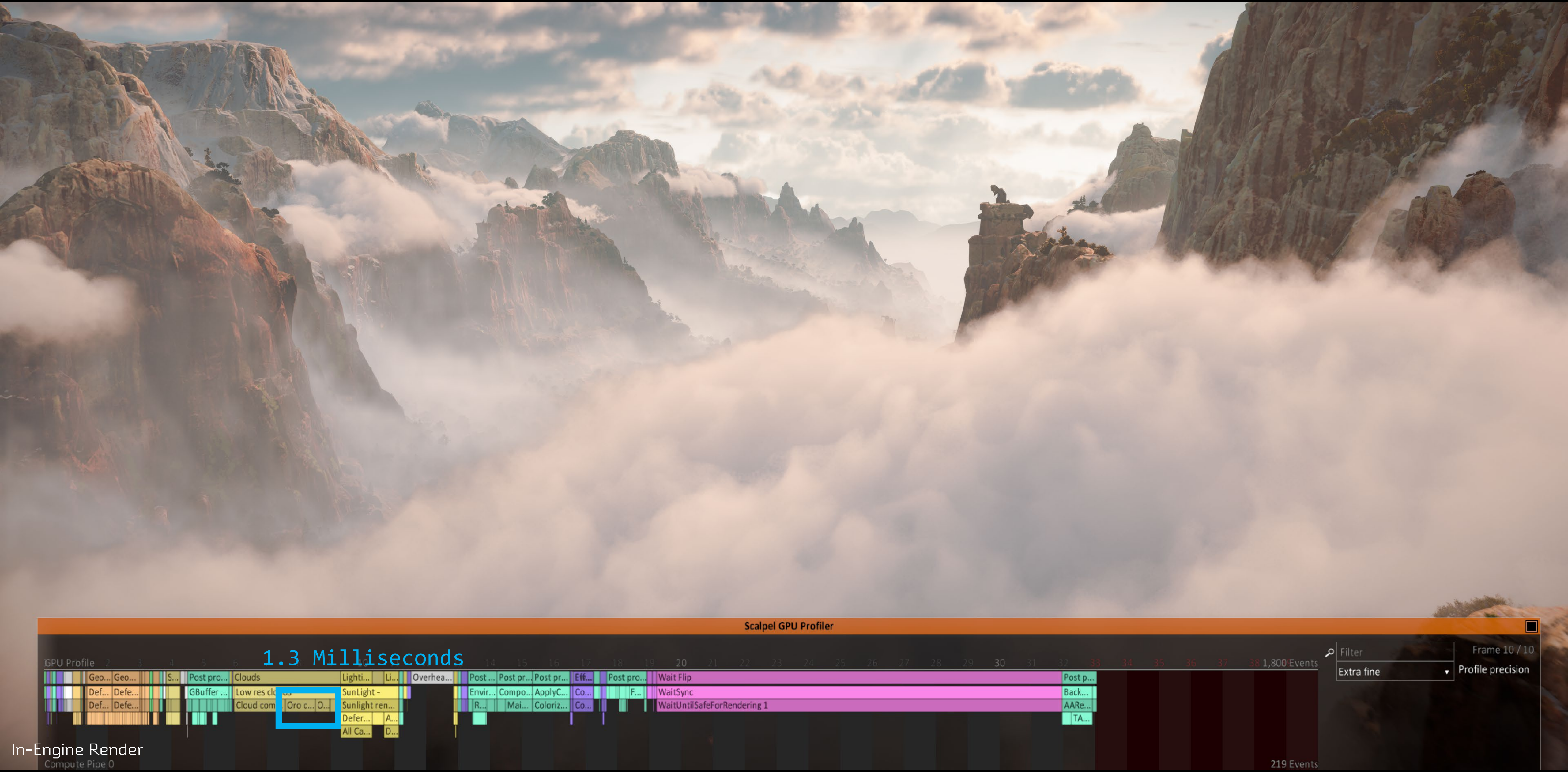

In-Engine Render


In-Engine Render


In-Engine Render


Scalpel GPU Profiler
4.2 Milliseconds
GPU Profile
Clouds
Low res clouds
Oro clo...
Oro clouds - high res
Overhead
Wait Flip
WaitSync
WaitUntilSafeForRendering 1
1,800 Events
Frame 10 / 10
Filter
Extra fine
Profile precision
219 Events
Compute Pipe 0
In-Engine Render

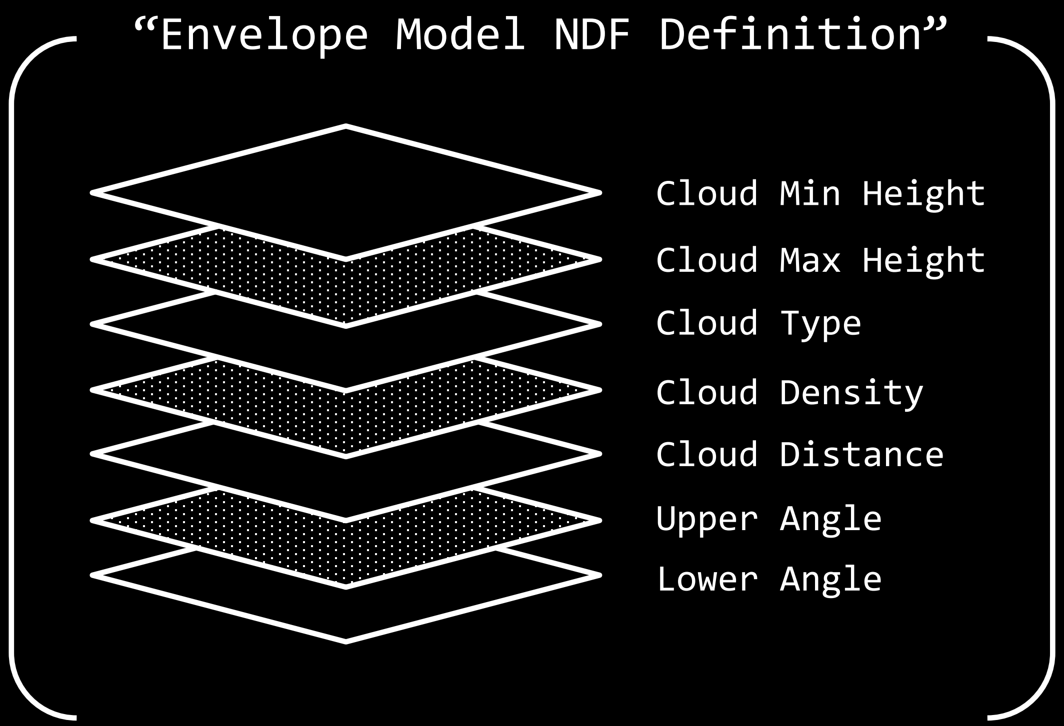
NUBIS EVOLVED / Environments / The Envelope Model / Ray-Marching



In-Engine Render

In-Engine Render

NUBIS EVOLVED / Environments / The Envelope Model / Results


"Envelope Model NDF Definition"
Cloud Min Height
Cloud Max Height
Cloud Type
Cloud Density
Cloud Distance
Upper Angle
Lower Angle

In-Engine Render

NUBIS EVOLVED / Environments / The Envelope Model / Locations


In-Engine Render

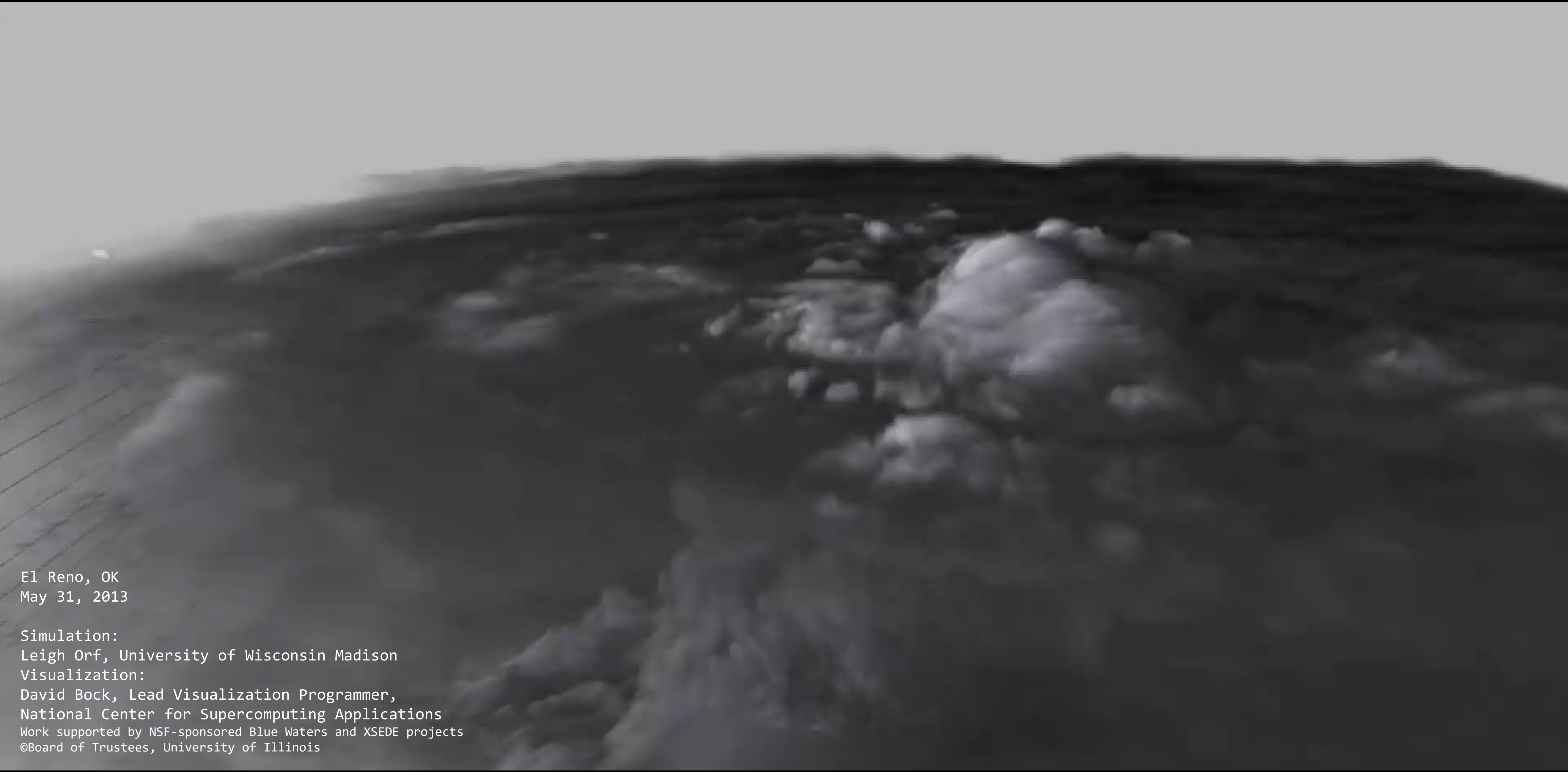
NUBIS EVOLVED / VFX

- Realistic
- Powerful
- Ominous
- Performant

In-Engine Render

Earth, TX - May 16, 2021
8:40pm-8:50pm

Photography: Isaac Schlueshe
@slushywx

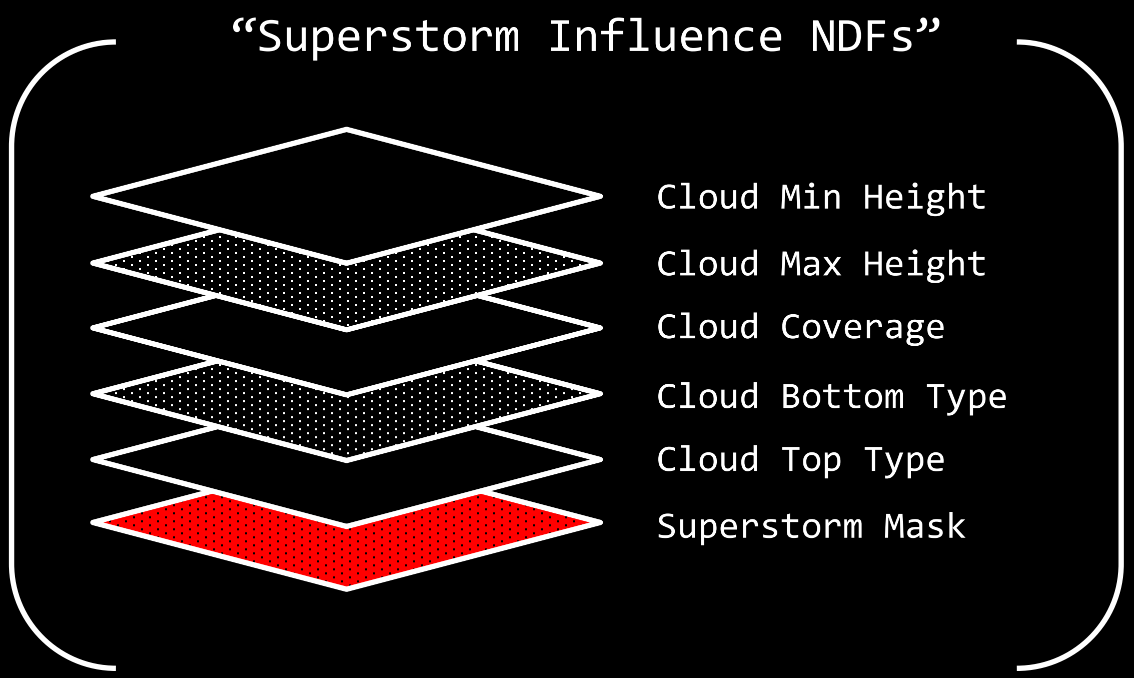


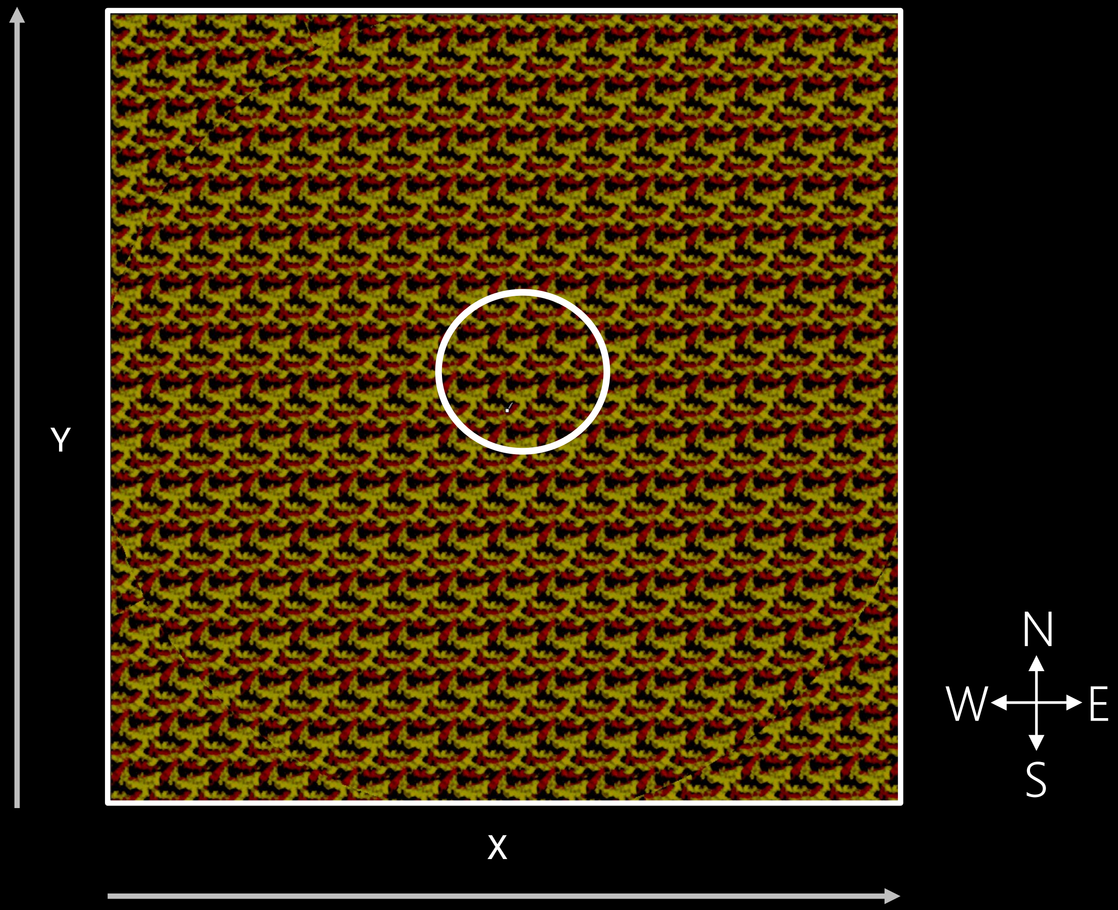
El Reno, OK
May 31, 2013

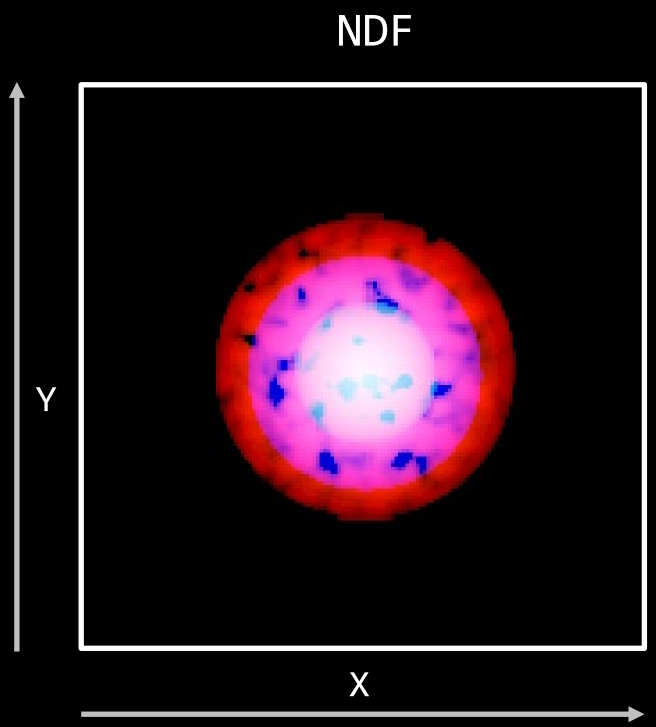
Simulation:
Leigh Orf, University of Wisconsin Madison
Visualization:
David Bock, Lead Visualization Programmer,
National Center for Supercomputing Applications
Work supported by NSF-sponsored Blue Waters and XSEDE projects

Photographs

Photograph


"Superstorm Influence NDFs"
Cloud Min Height
Cloud Max Height
Cloud Coverage
Cloud Bottom Type
Cloud Top Type
Superstorm Mask

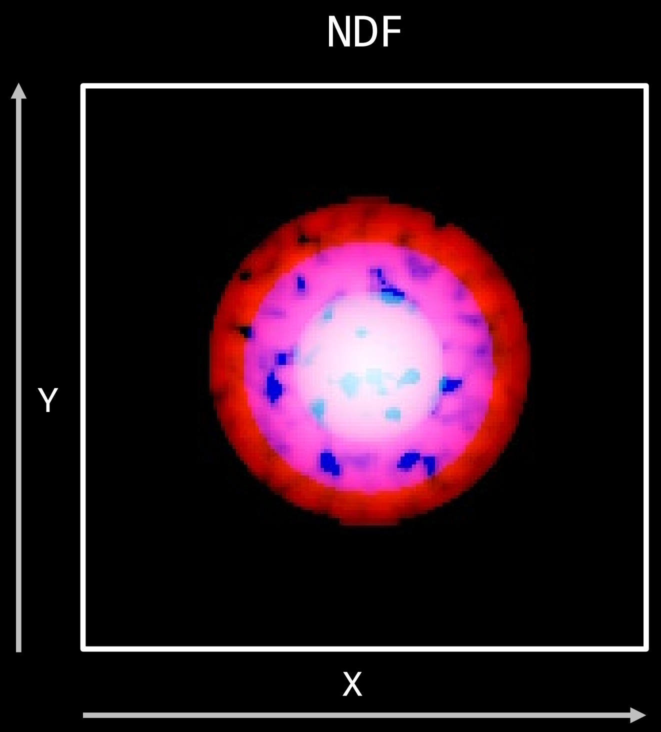

Y
X
N
W
E
S



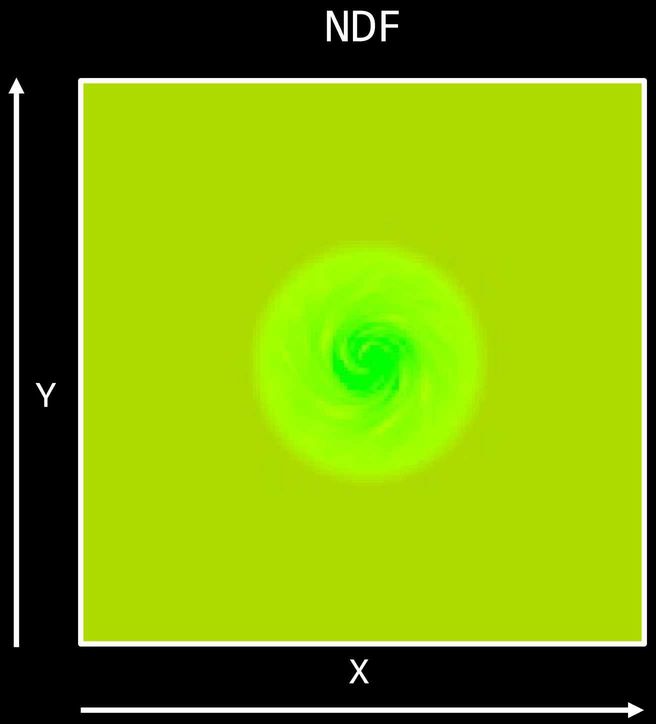
In-Engine Render


Bottom Type = 0.0
In-Engine Render


In-Engine Render

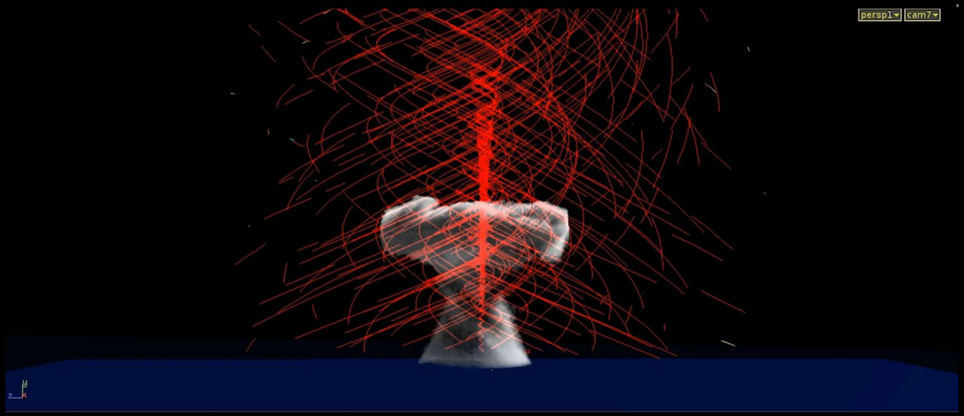


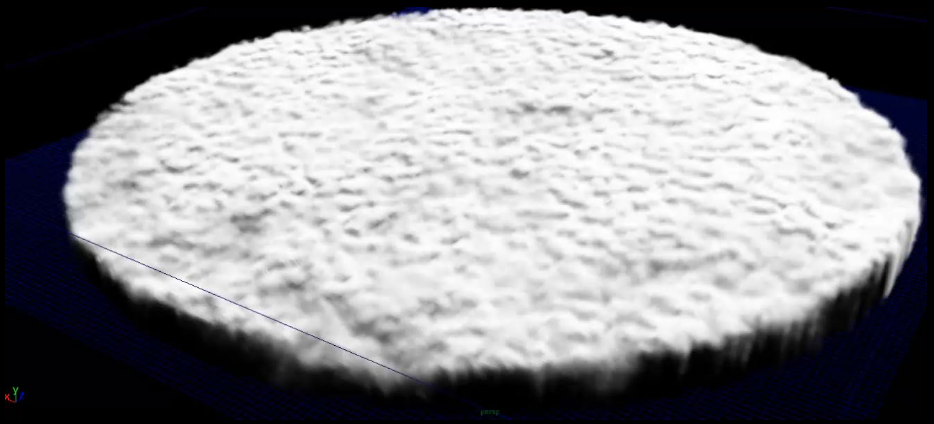
In-Engine Render

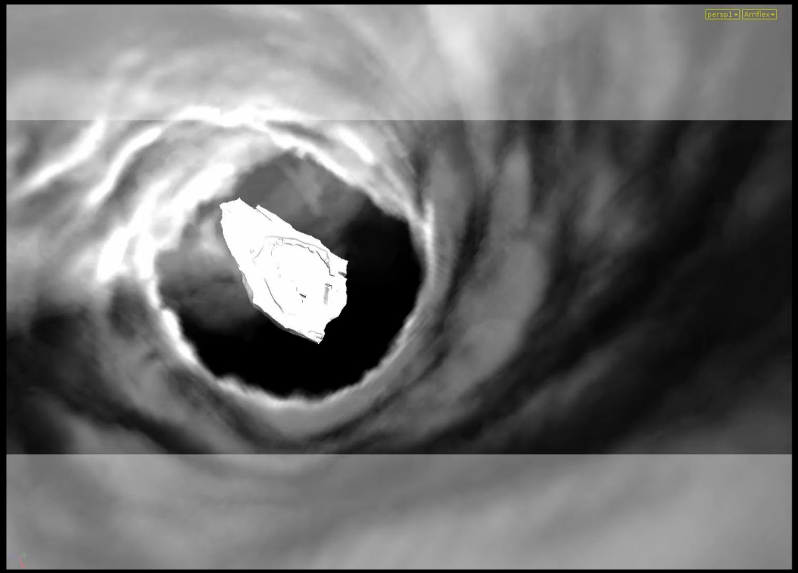


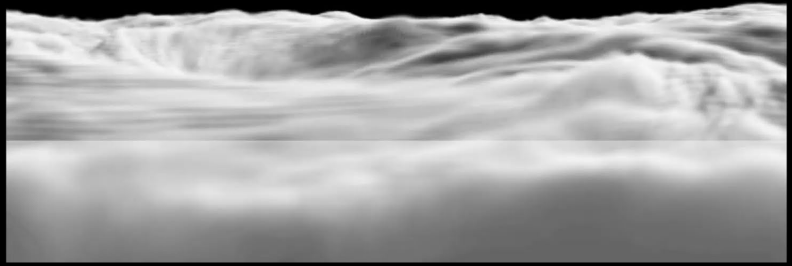
In-Engine Render


In-Engine Render



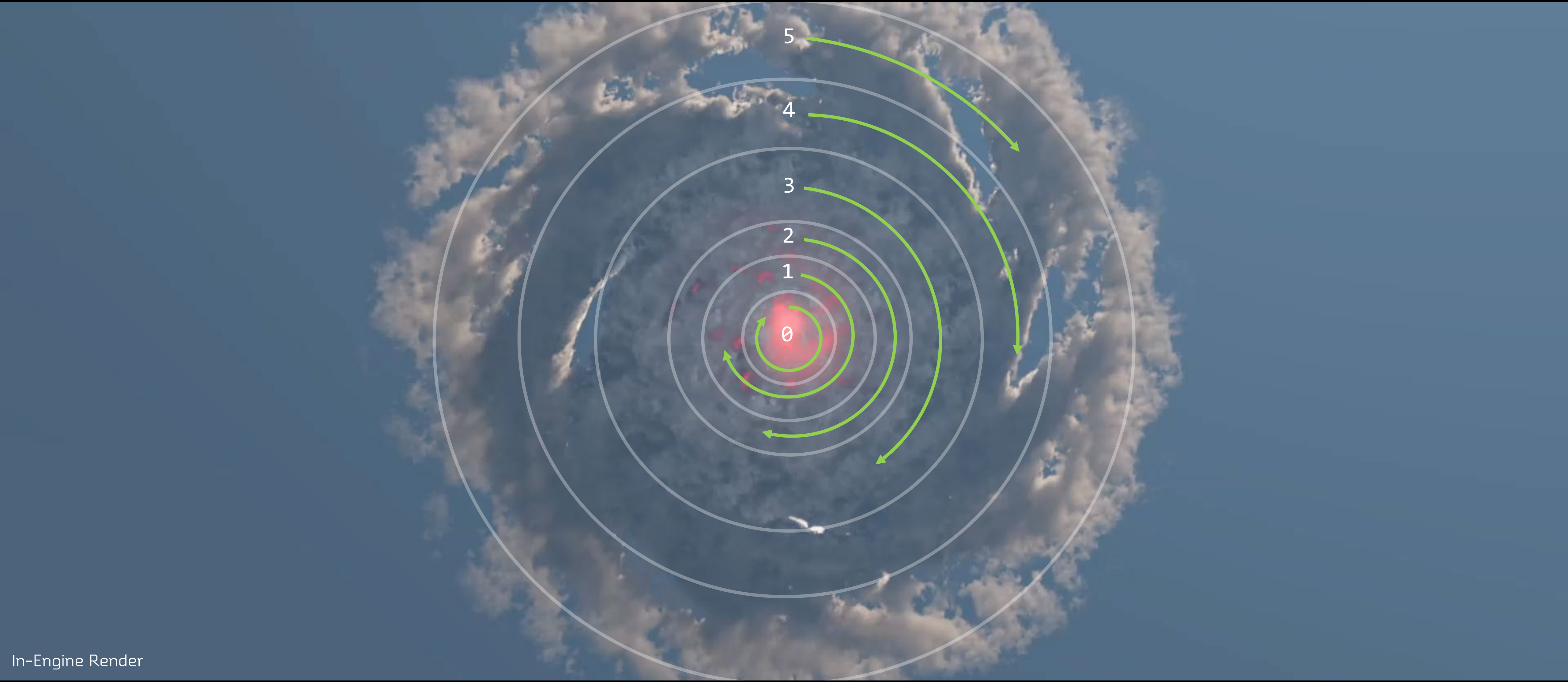






“M.D.R. Vortex Field”
Sine/Cosine rotation concept by Matthew D. Roach
Fluid Simulations by Andrew Schneider (Houdini/Maya)


```
float noise = SampleNoise(GetRotatedPosition( sample_position, superstorm_center, time_offset));
```

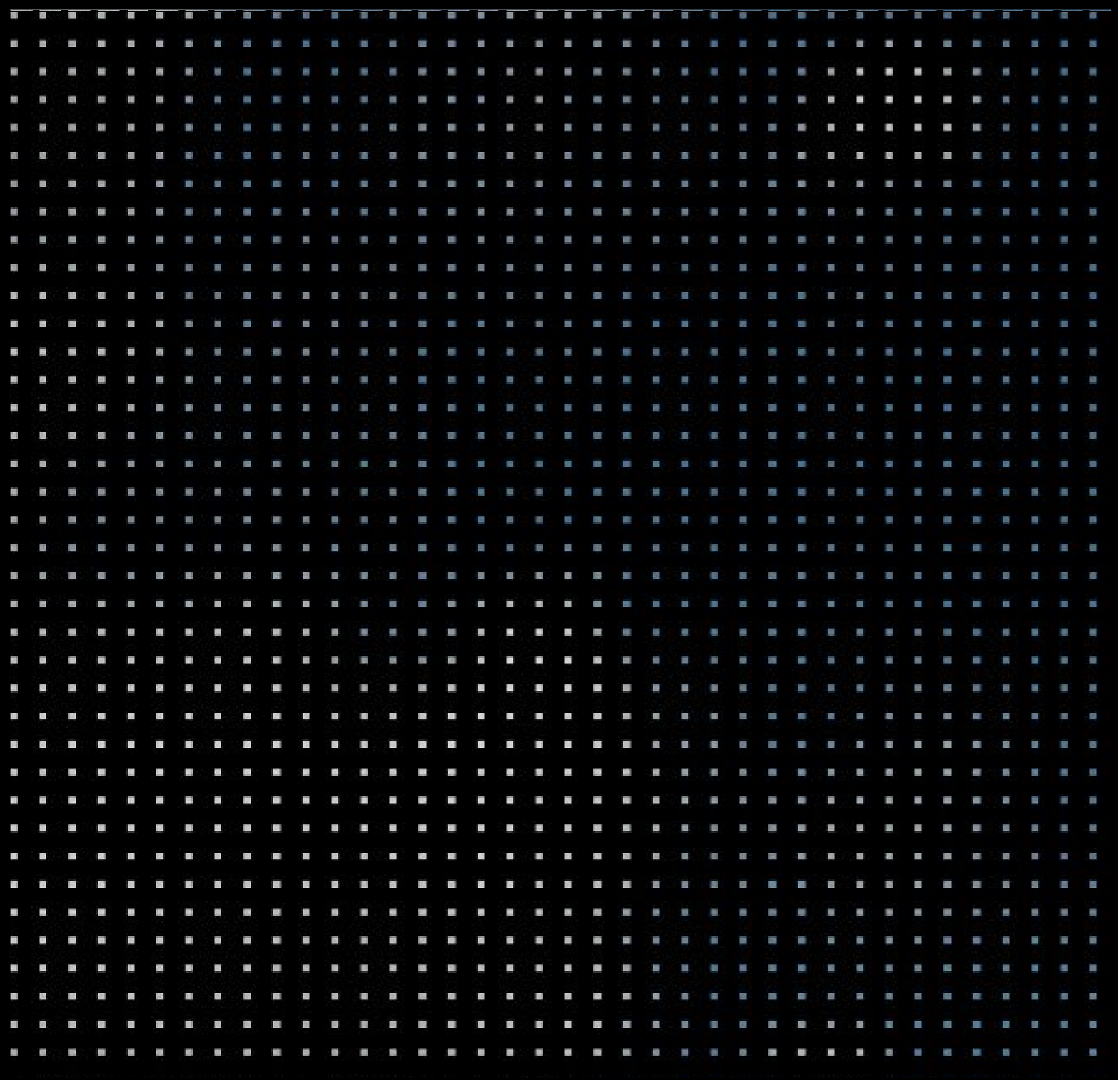



```
float noise = SampleNoise(GetRotatedPosition( sample_position, superstorm_center, time_offset * RingRotationSpeed[n]));
```

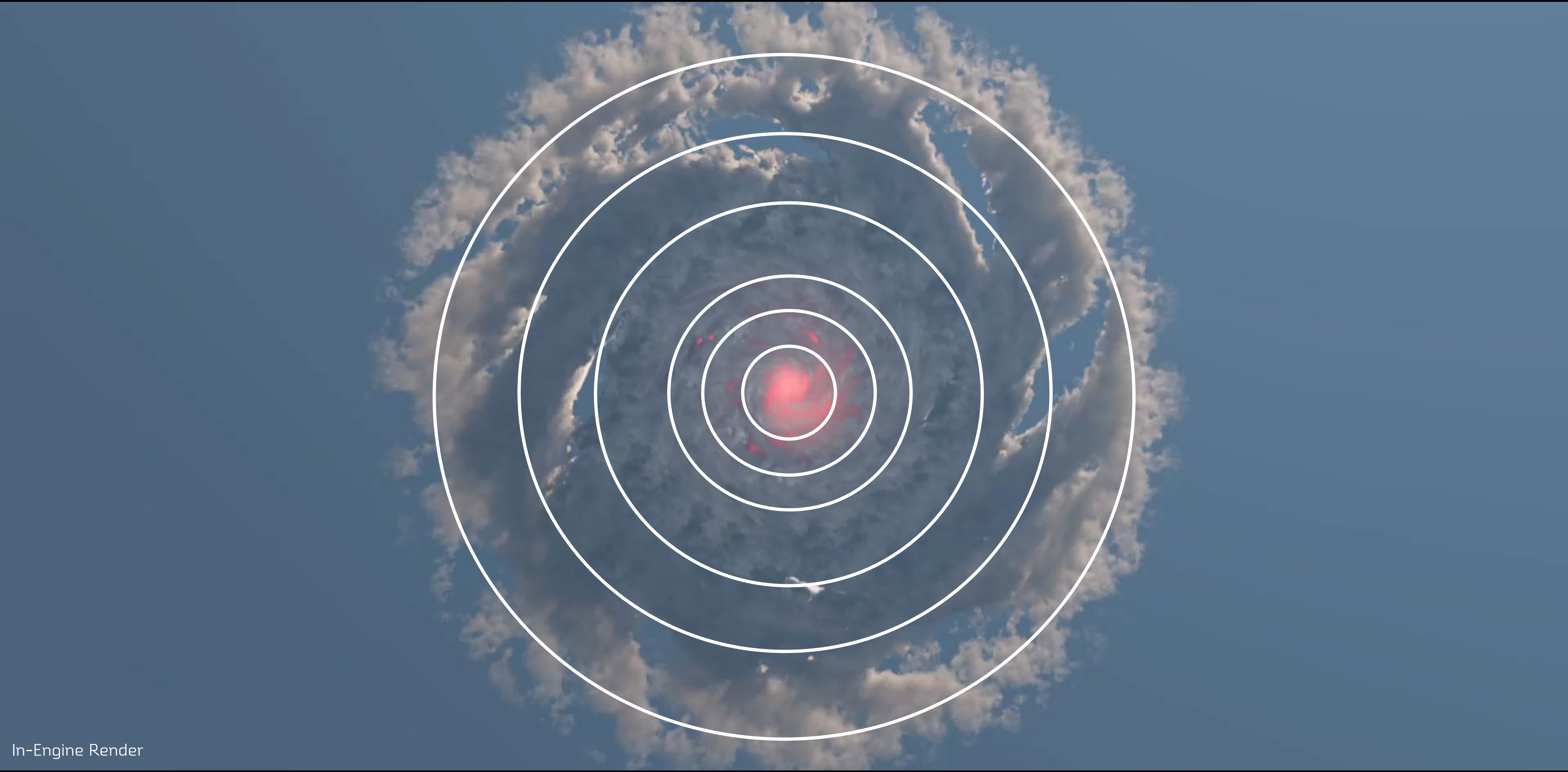
In-Engine Render

```
float noise = SampleNoise(GetRotatedPosition( sample_position, superstorm_center, time_offset * ring_rotation_speed[n] + ring_skew[n]));
```


In-Engine Render




In-Engine Render



In-Engine Render

```
// Get world space cloud position
float3 view_space_vec = CreateEyeRay(inViewportUV, inFovScale);
float cloud_distance = inCloudAttrWorkingBuffer.SampleLOD(inSampler, inUV, 0).r;
float3 cloud_world_space = mul(inInvViewMatrix, float4(view_space_vec * cloud_distance, 1.0)).xyz;

// Rotate around superstorm center
float rotation_speed = GetSuperstormRotationSpeed(cloud_world_space.xy, superstorm_center, superstorm_radius, superstorm_blend_factor);
float2 rotating_motion_offset = GetRotatedPosition(cloud_world_space.xy, superstorm_center, rotation_speed * inDeltaTime);
float3 superstorm_rotated_world_space_position = float3(0.0, 0.0, cloud_world_space.z);
superstorm_rotated_world_space_position.xy = rotating_motion_offset;

// Get superstorm mask for blending - powered linear distance from radius to center of superstorm
float superstorm_mask = pow(saturate(1.0 - length(cloud_world_space.xy - superstorm_center) / superstorm_radius), 0.1);

// Blend vectors from normal to superstorm over superstorm mask.
cloud_world_space = lerp(cloud_world_space, superstorm_rotated_world_space_position, superstorm_mask);
view_space_vec = mul(inViewMatrix, float4(cloud_world_space, 1.0)).xyz;

// Construct previous sample position from new view space vector
float4 prev_sample_pos = mul(inReprojectionMatrix, float4(view_space_vec, 1.0));
prev_sample_pos /= prev_sample_pos.w;
prev_sample_pos.xy *= float2(0.5, -0.5)
prev_sample_pos.xy += float2(0.5, 0.5);
```

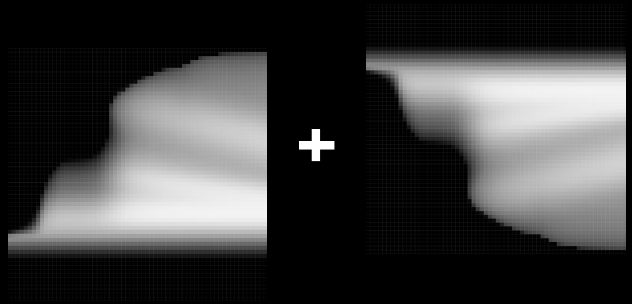
In-Engine Render

```
// Get world space cloud position
float3 view_space_vec = CreateEyeRay(inViewportUV, inFovScale);
float cloud_distance = inCloudAttrWorkingBuffer.SampleLOD(inSampler, inUV, 0).r;
float3 cloud_world_space = mul(inInvViewMatrix, float4(view_space_vec * cloud_distance, 1.0)).xyz;

// Rotate around superstorm center
float rotation_speed = GetSuperstormRotationSpeed(cloud_world_space.xy, superstorm_center, superstorm_radius, superstorm_blend_factor);
float2 rotating_motion_offset = GetRotatedPosition(cloud_world_space.xy, superstorm_center, rotation_speed * inDeltaTime);
float3 superstorm_rotated_world_space_position = float3(0.0, 0.0, cloud_world_space.z);
superstorm_rotated_world_space_position.xy = rotating_motion_offset;

// Get superstorm mask for blending - powered linear distance from radius to center of superstorm
float superstorm_mask = pow(saturate(1.0 - length(cloud_world_space.xy - superstorm_center) / superstorm_radius), 0.1);

// Blend vectors from normal to superstorm over superstorm mask.
cloud_world_space = lerp(cloud_world_space, superstorm_rotated_world_space_position, superstorm_mask);
cloud_world_space = lerp(cloud_world_space + scroll_direction_2D * inDeltaTime, superstorm_rotated_world_space_position, superstorm_mask);
view_space_vec = mul(inViewMatrix, float4(cloud_world_space, 1.0)).xyz;

// Construct previous sample position from new view space vector
float4 prev_sample_pos = mul(inReprojectionMatrix, float4(view_space_vec, 1.0));
prev_sample_pos /= prev_sample_pos.w;
prev_sample_pos.xy *= float2(0.5, -0.5)
prev_sample_pos.xy += float2(0.5, 0.5);
```

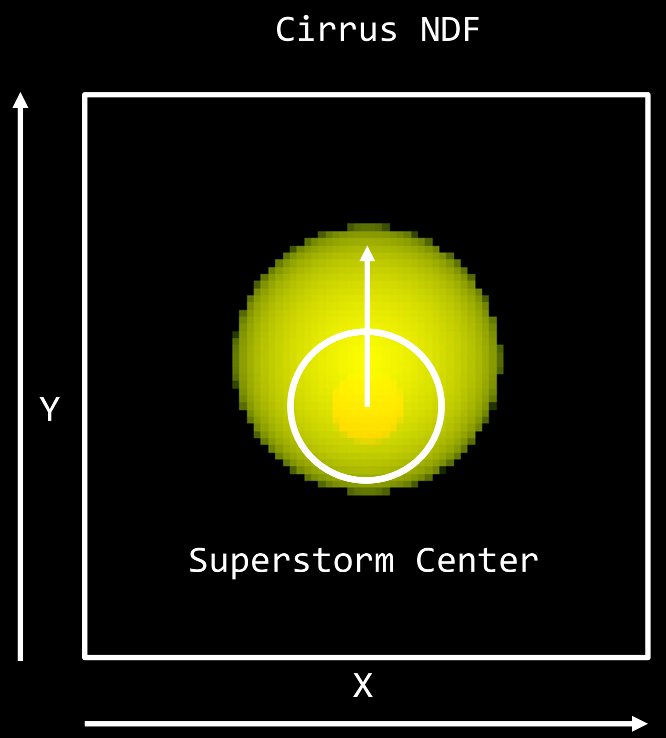

In-Engine Renders

Photograph

Vertical Gradient



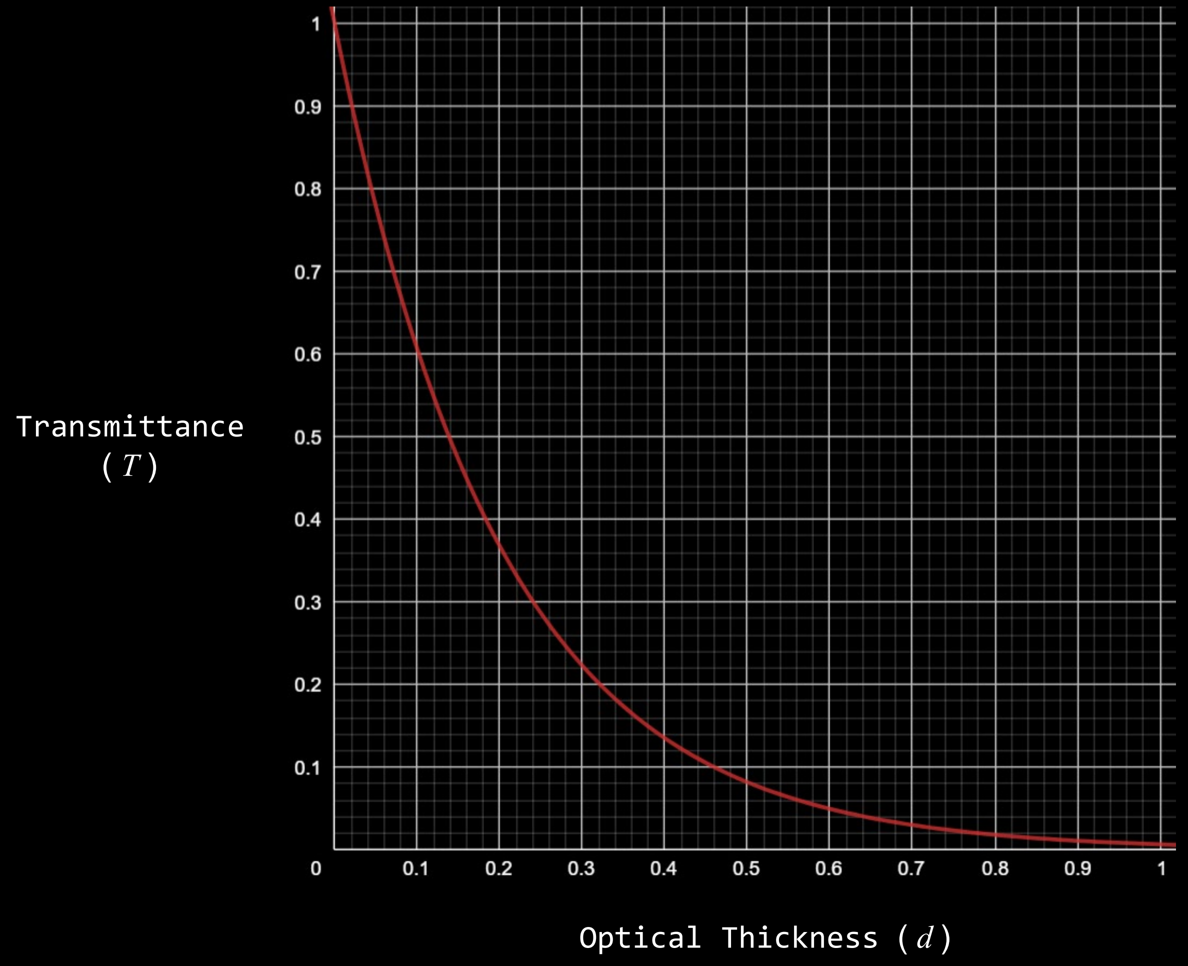
In-Engine Render

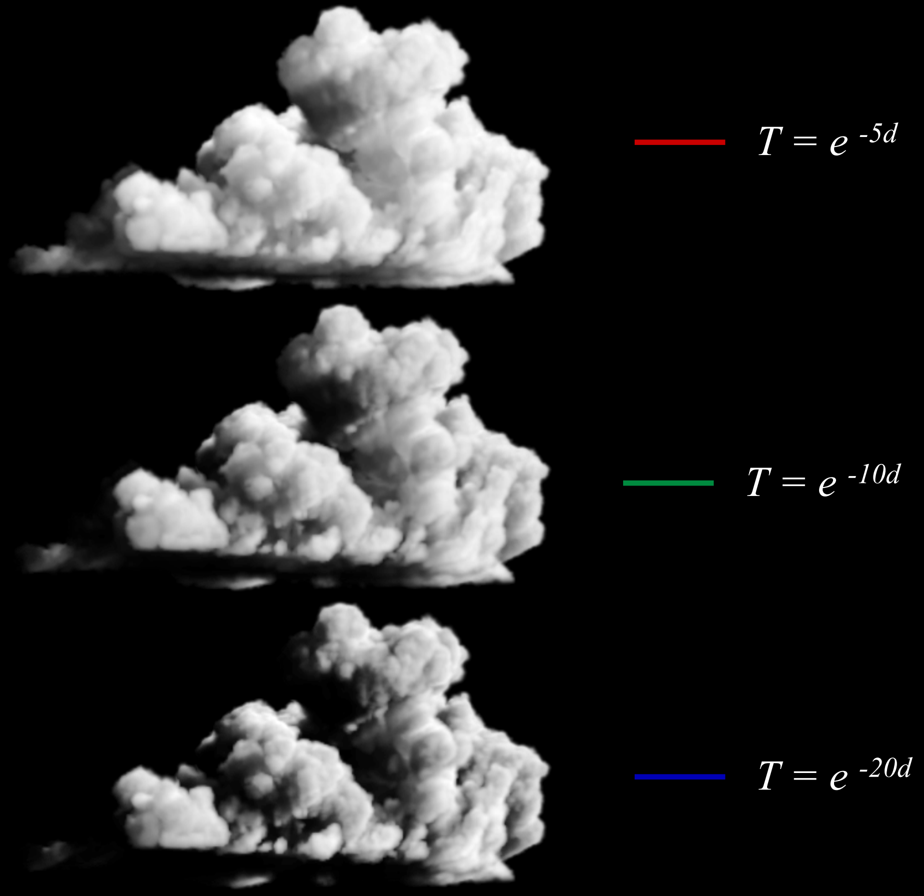


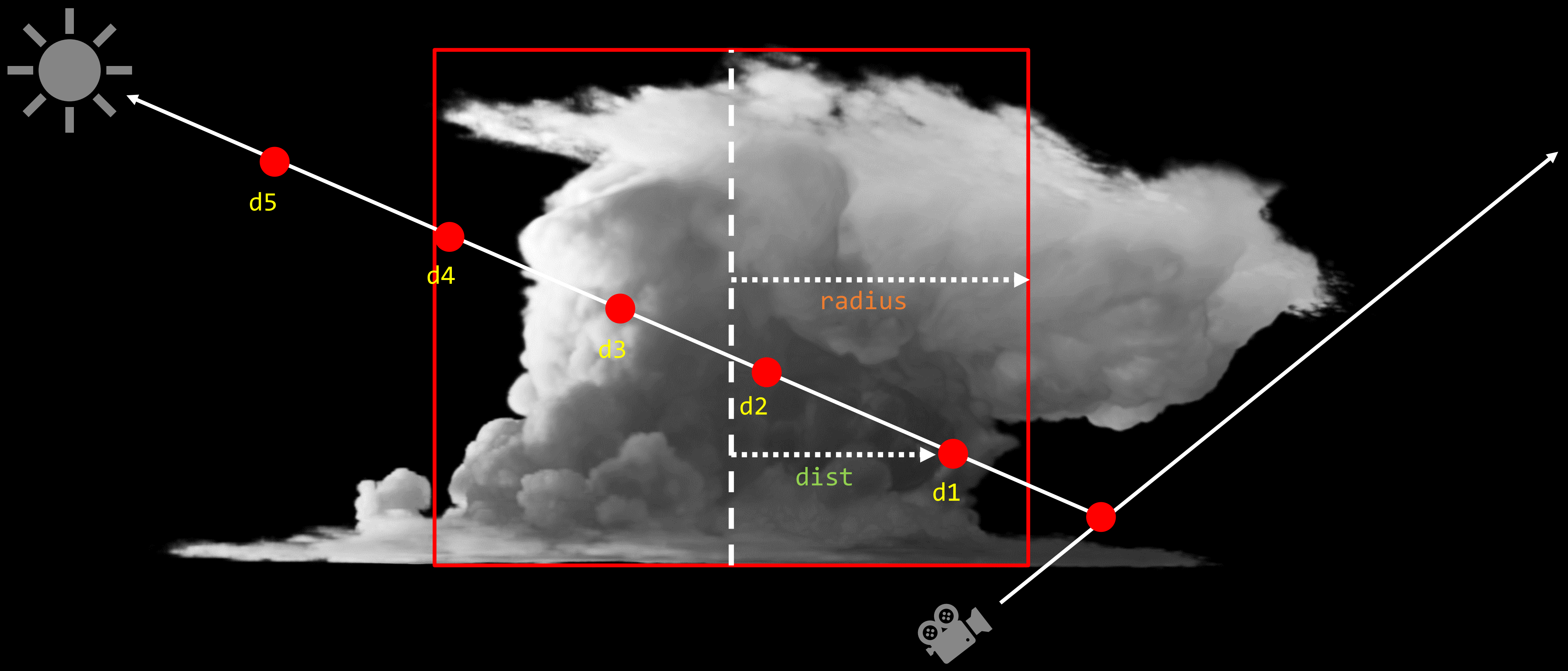
In-Engine Render


In-Engine Render

In-Engine Render

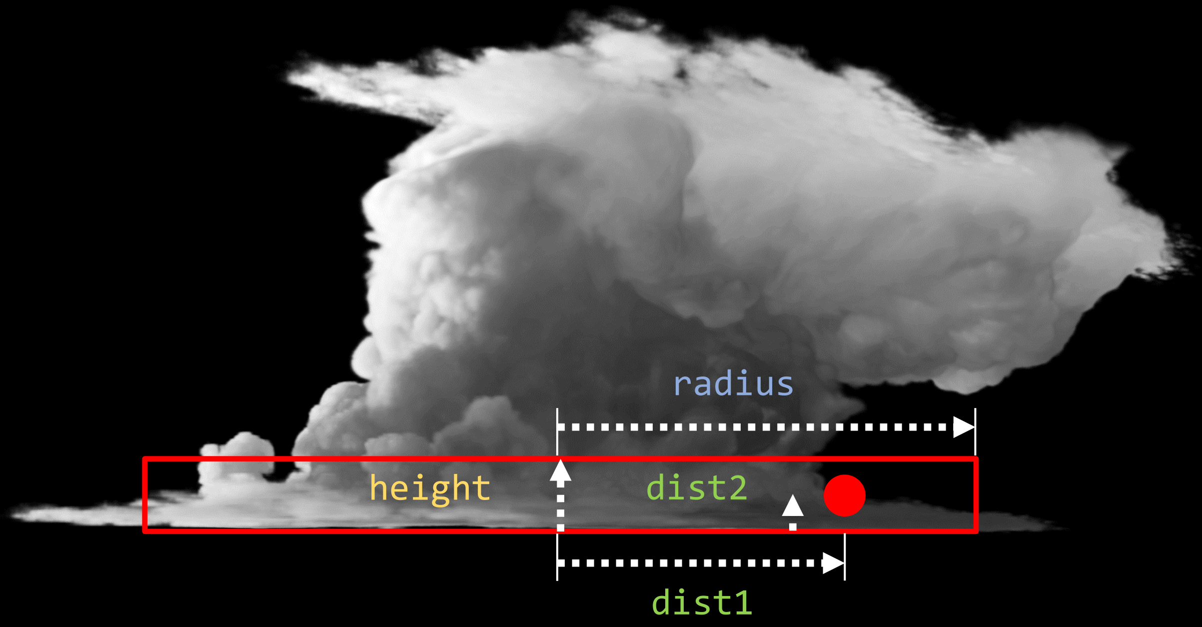

Transmittance
(T)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Optical Thickness (d)


$T = e^{-5d}$
$T = e^{-10d}$
$T = e^{-20d}$




```
density_scale = pow(min(1.0 - (dist / radius), 0.0) + 1.0, 0.5);
summed_density += d(n) * density_scale;
transmittance = exp( -1.0 * (summed_density));
```

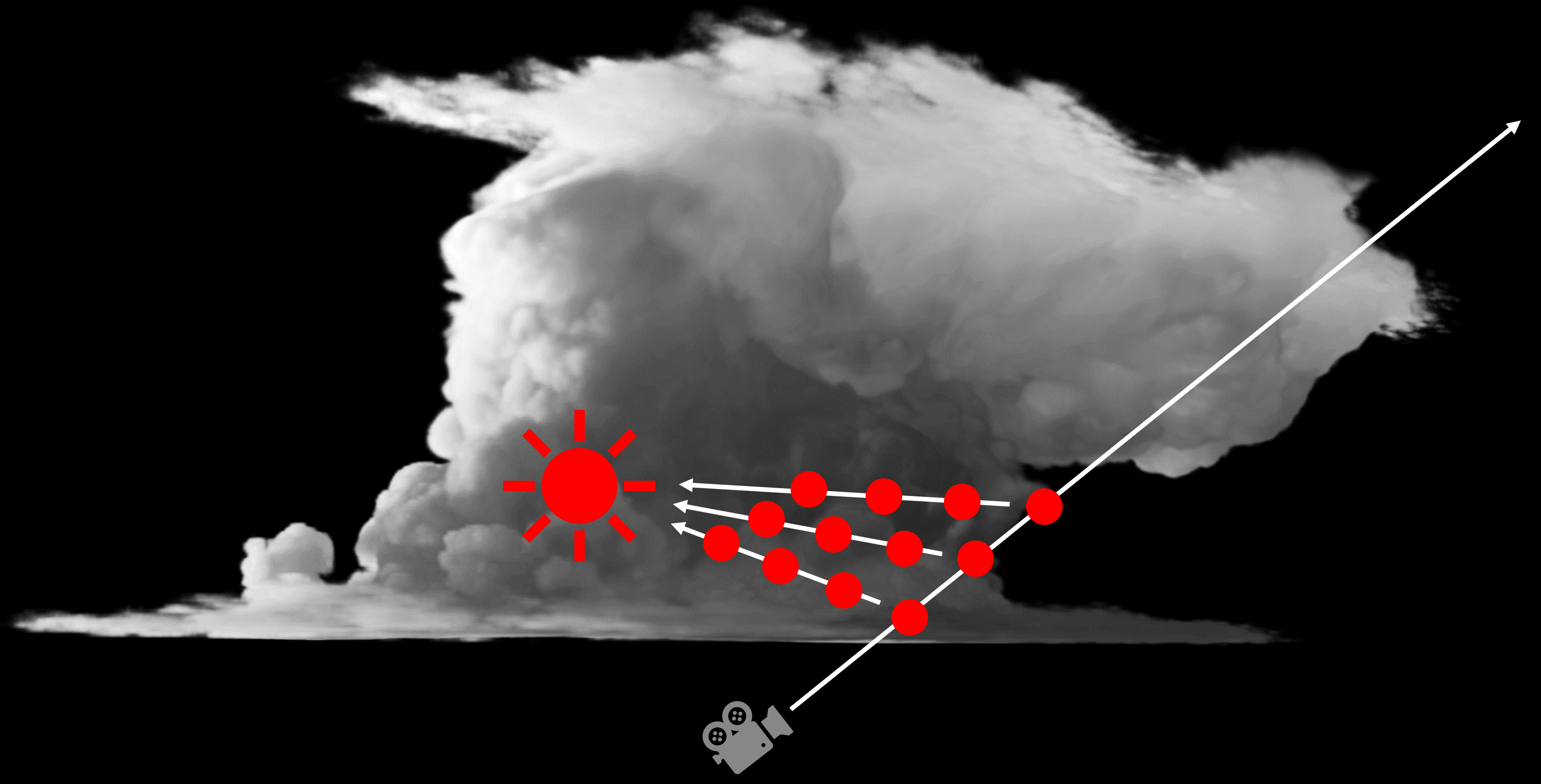
In-Engine Render

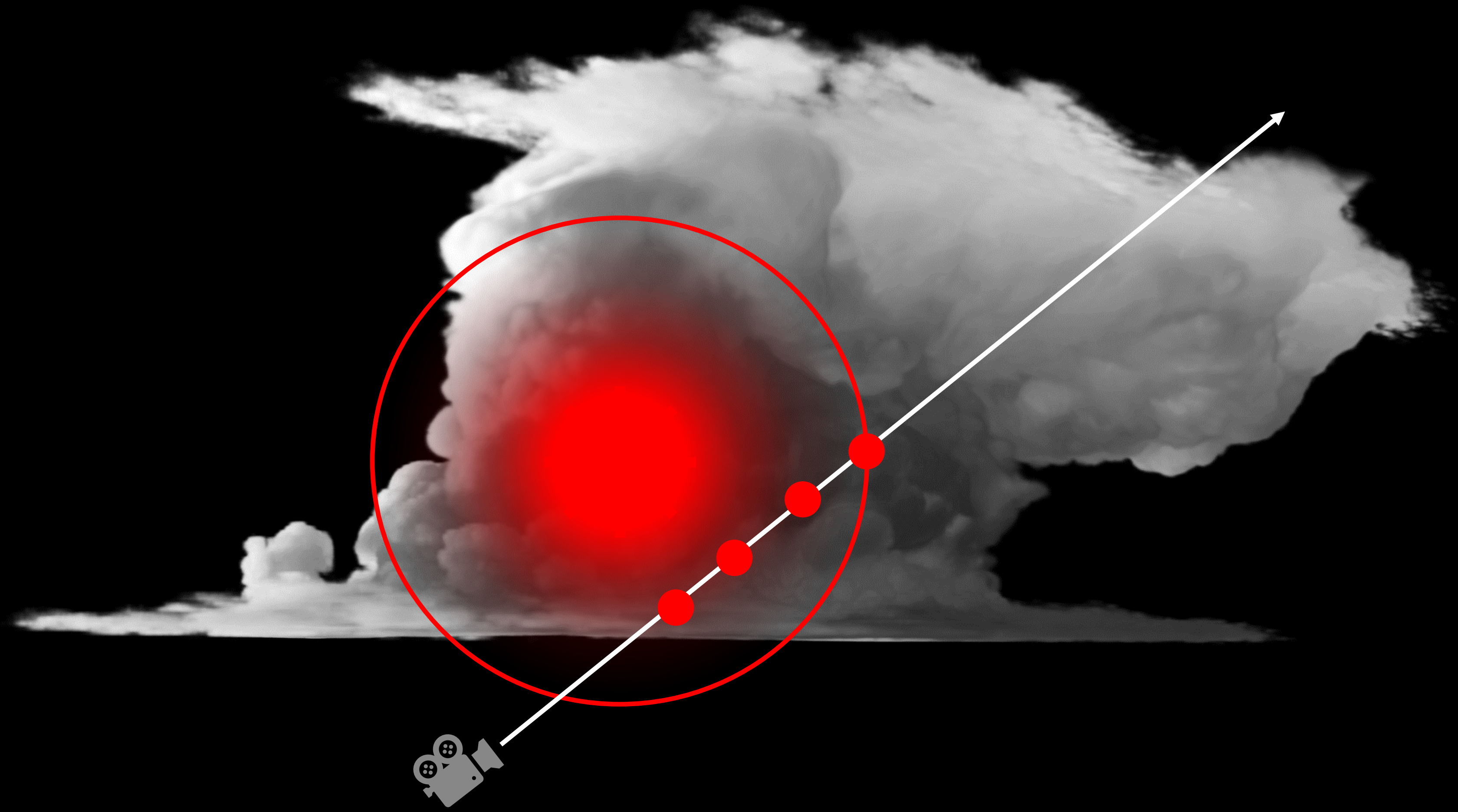


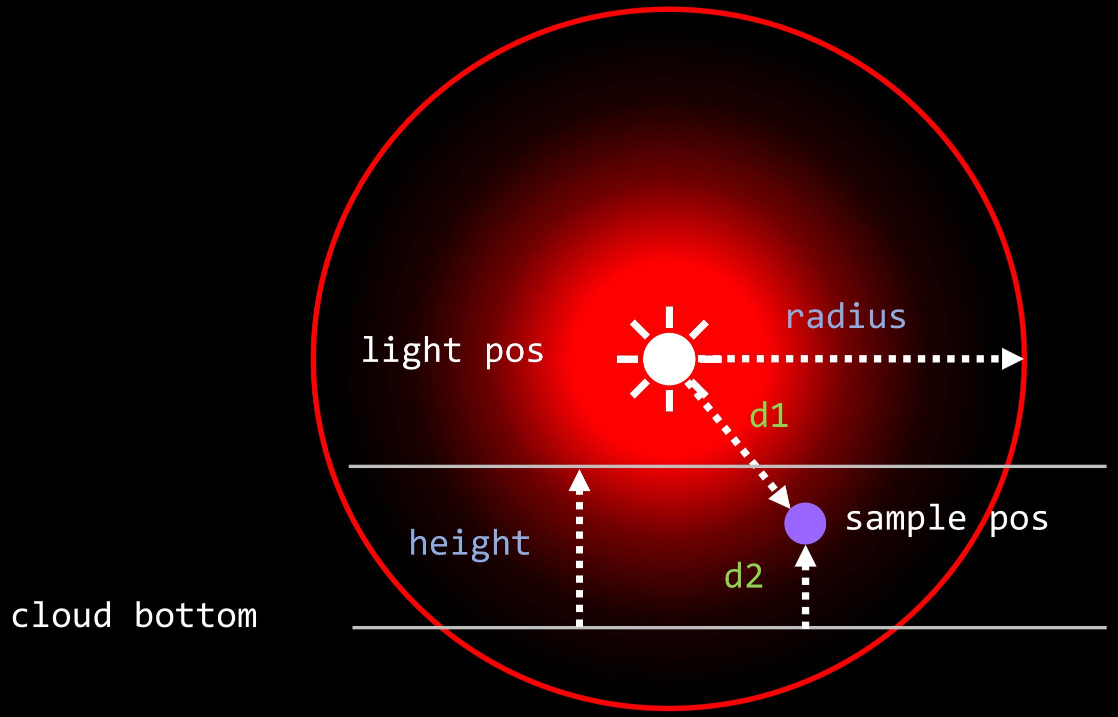

```
amb_settings_blend = min(1.0 - (dist1 / radius), 0.0) * (dist2 / height);

ambient_scattering_settings = lerp(cloud_amb_settings, superstorm_amb_settings, amb_settings_blend);
```


In-Engine Render

In-Engine Render

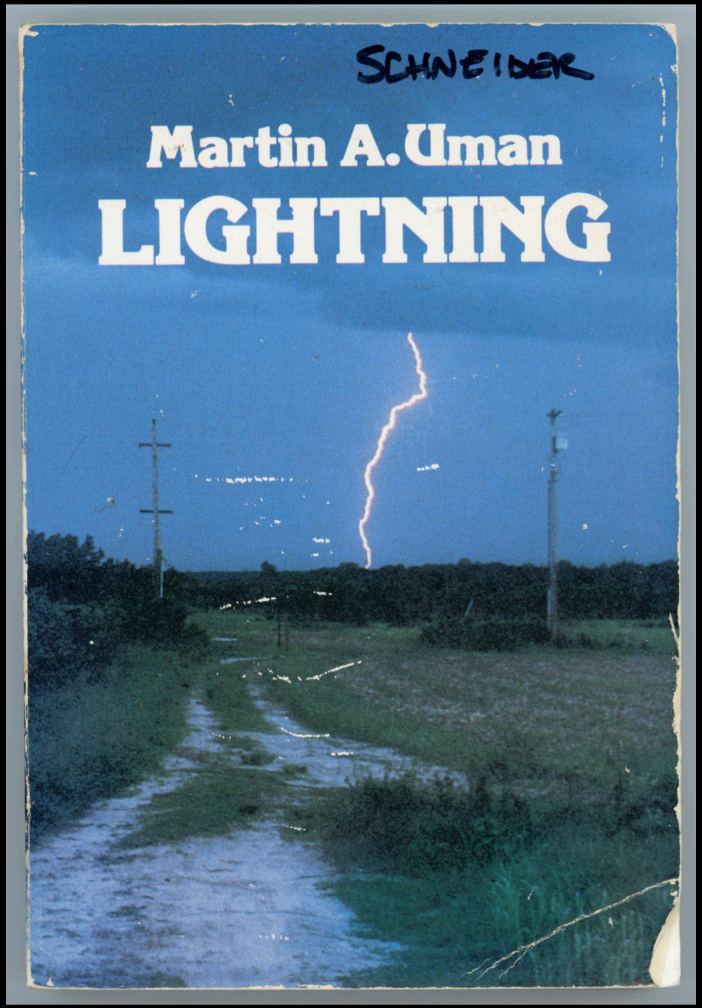


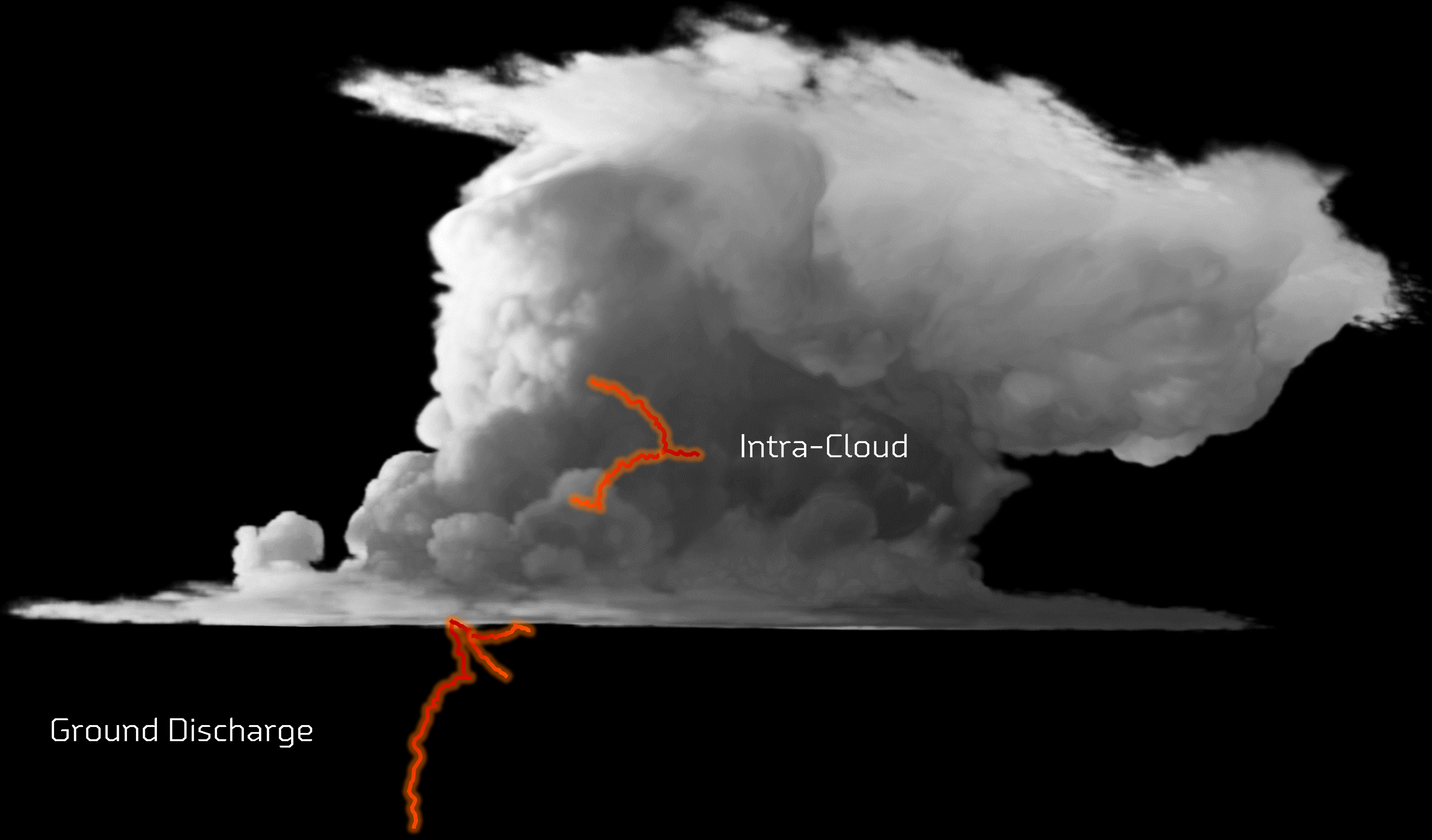




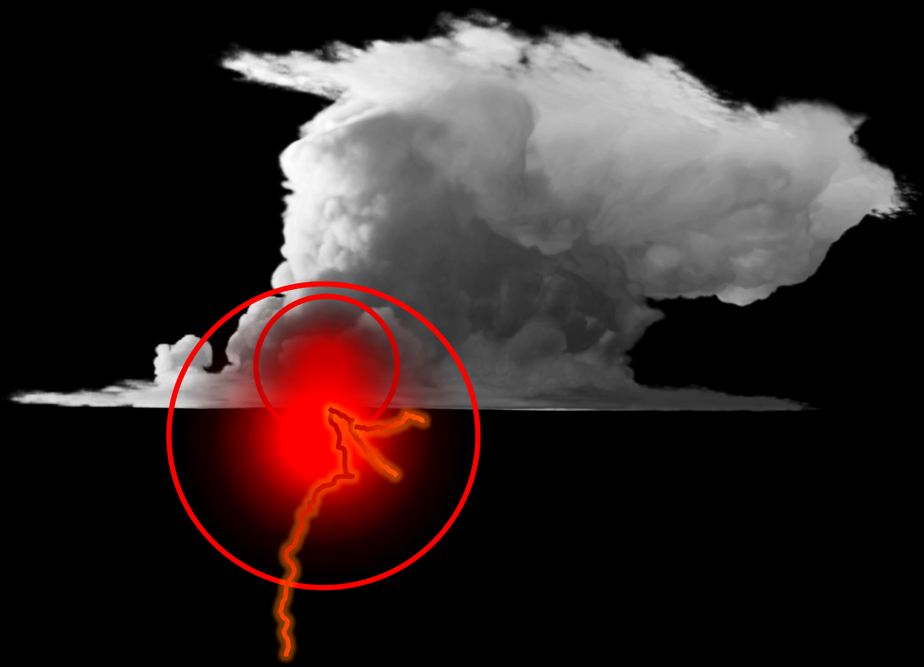

```
potential_energy = pow( 1.0 - (d1 / radius), 12.0);
height_gradient = (d2 / height);
pseudo_attenuation = (1.0 - saturate(fine_density * 5.0));
glow_energy = potential_energy * height_gradient * pseudo_attenuation;
light_energy = direct_scattering + ambient_scattering + glow_energy;
```


In-Engine Render


SCHNEIDER
Martin A. Uman
LIGHTNING


Intra-Cloud
Ground Discharge


In-Engine Render




In-Engine Render

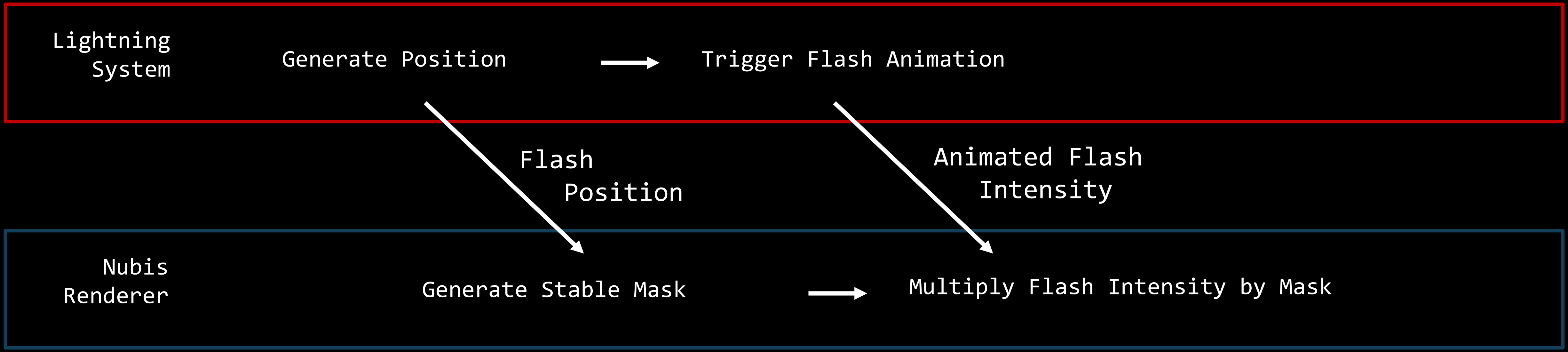
Repeat, build intensity

Intra-Cloud → Random Delay → .... → Intra-Cloud + Ground Discharge

No Solution

Our Solution

In-Engine Renders


Lightning System
Generate Position
Trigger Flash Animation
Flash Position
Animated Flash Intensity
Nubis Renderer
Generate Stable Mask
Multiply Flash Intensity by Mask


In-Engine Render


RenderBuffer XRay - Press F9 for a screenshot of the selected buffer.


In-Engine Render

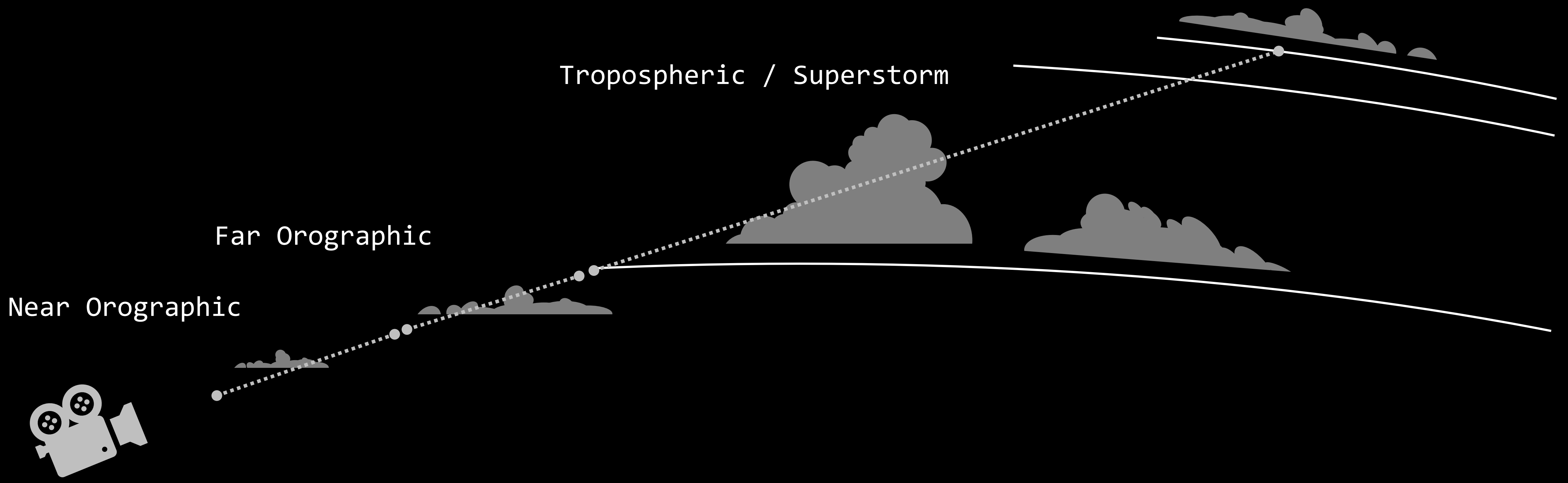
	PlayStation 4	PlayStation 5
Max Resolution	960 x 540	1920 x 1080
Light Ray Samples	6	10
View Ray Samples	60 – 90	96 – 180
Blur Scale (Pixels)	2x	1x
Noise Texture MIP Level	1	0

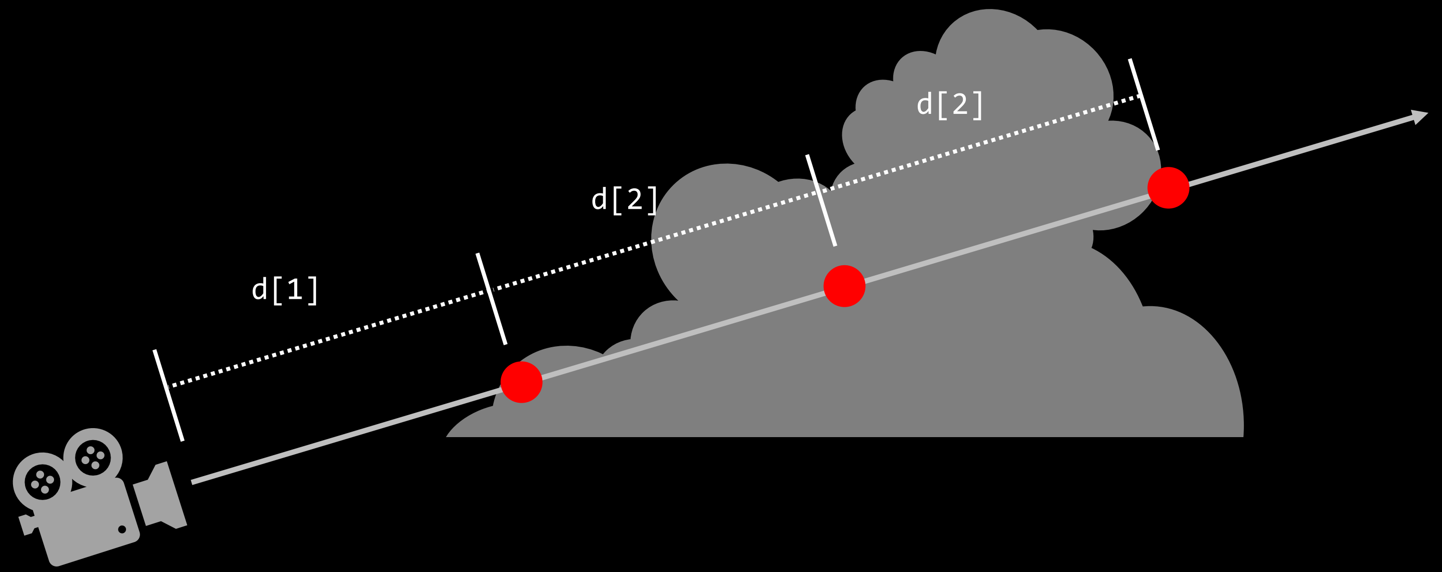
<= 4 milliseconds

<= 2-3 milliseconds

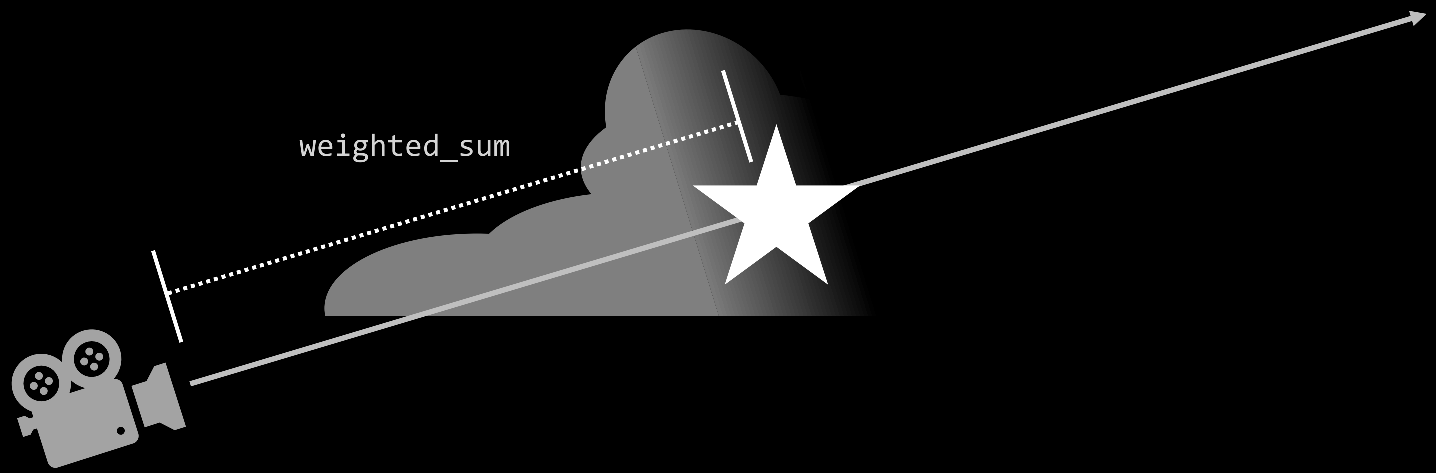
In-Engine Renders

In-Engine Render


Tropospheric / Superstorm
Far Orographic
Near Orographic




```
while (…)
{
    distance_sum += d[n] * sample_opacity;
}

float weighted_sum = distance_sum / opacity_sum;
cloud_color = lerp(cloud_color, atmospherics_color, weighted_sum);
```


weighted_sum

NUBIS EVOLVED / Integration

SIGGRAPH 2011, Vancouver

Clouds in the skies of *Rio*

Andrew P. Schneider Trevor G. Thomson Mathew S. Wilson
Blue Sky Studios*

Figure 1: *Blu and Jewel hangliding through the clouds over* Rio. Rio© *2011 Twentieth Century Fox Film Corporation. All Rights Reserved.*

1 Introduction

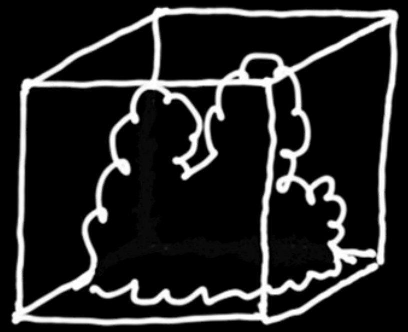
We faced four major challenges when creating the clouds and skies in *Rio*. The first challenge was making volumetric clouds that could be rendered in stereo. Previous approaches involved matte paintings and 2D cards, which lacked parallax and could not be lit correctly.

clouds into a scene, modified them, and wrote a low resolution version of each cloud to disk. The resampled resolution was arrived at interactively inside of Houdini based on each clouds distance from camera. In this case, evolution was applied at render time by skewing each voxel grid and deforming the noise coordinates according to wind direction and speed. For long sequences like the one where Blu and Jewel fly over Rio (Fig. 1), we placed all of the clouds into a master set, and then adjusted as needed per shot.

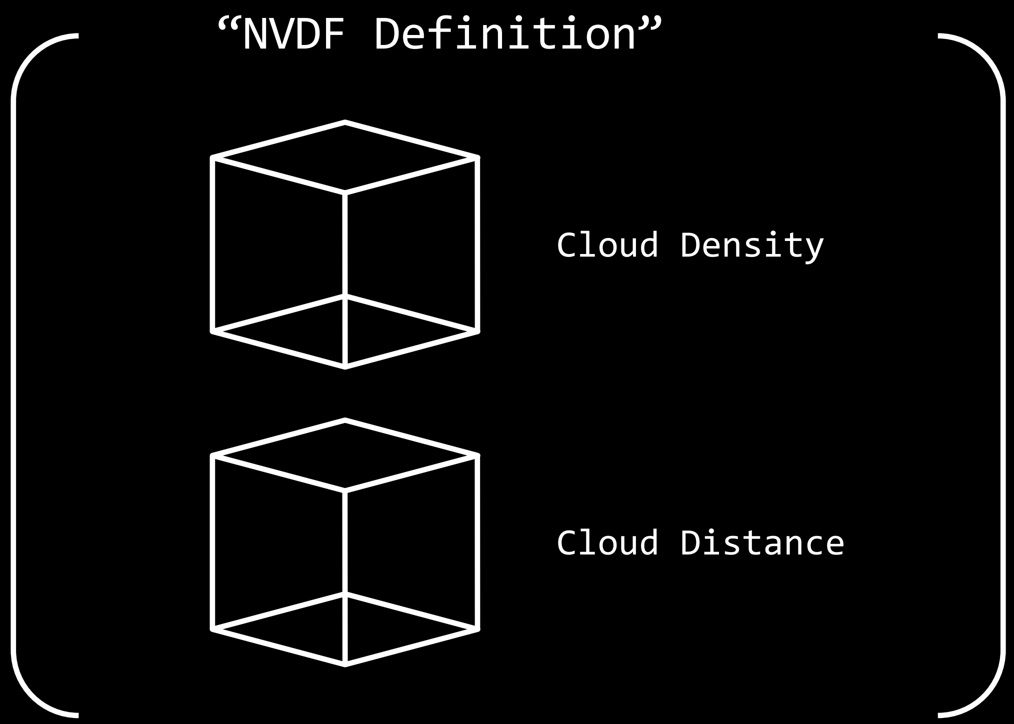
Finally, for distant shots, where clouds were so distant that parallax was of no concern, RGB renders of assets from the cloud library were placed in 3D composite space and then relit in Nuke. These cloud cards were then combined with rendered versions of other 3D clouds in the shot into a ”sky set” that could be shared across multiple shots and sequences. We developed a number of tools to color the clouds in harmony with a proprietary, pseudo-volumetric sky generator, that allowed us to maintain tight control over art direction and time of day.

4 Volumetric Rendering

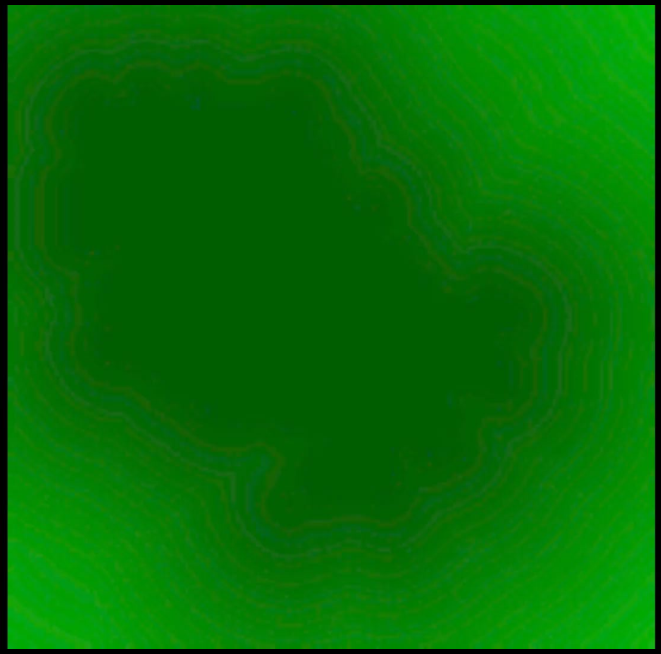
In-Engine Renders

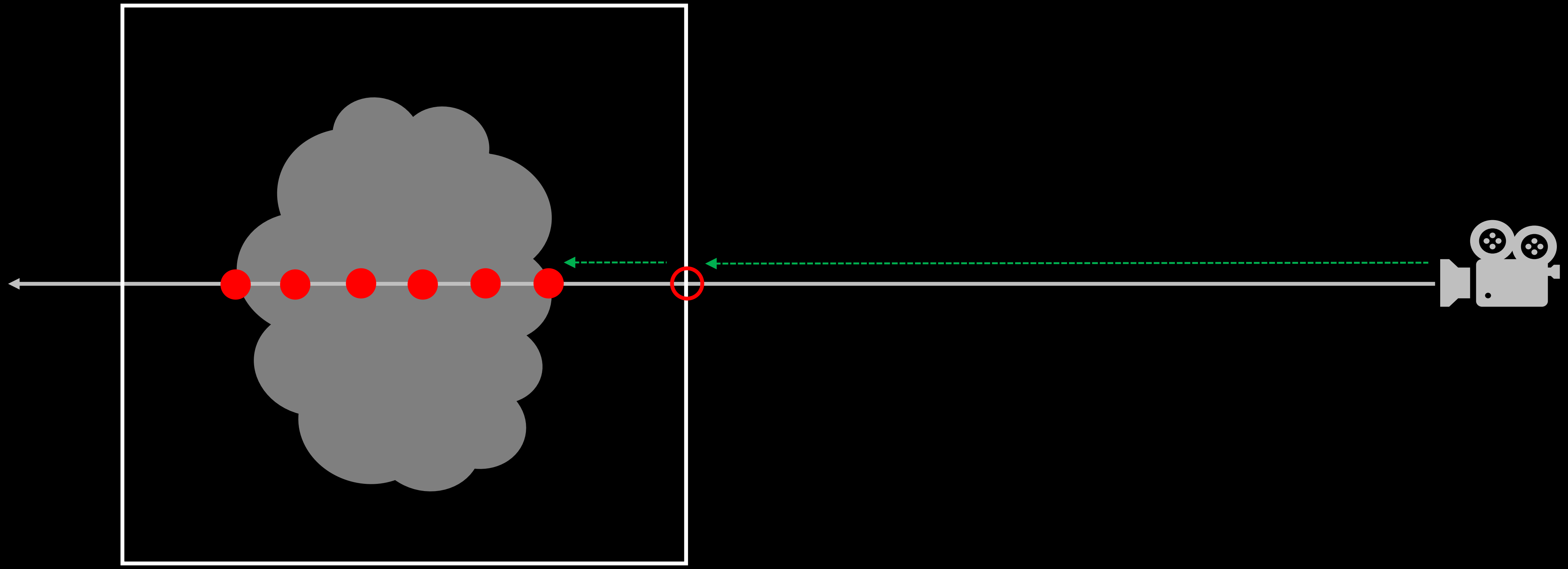


Volume Model



Vertical Slices

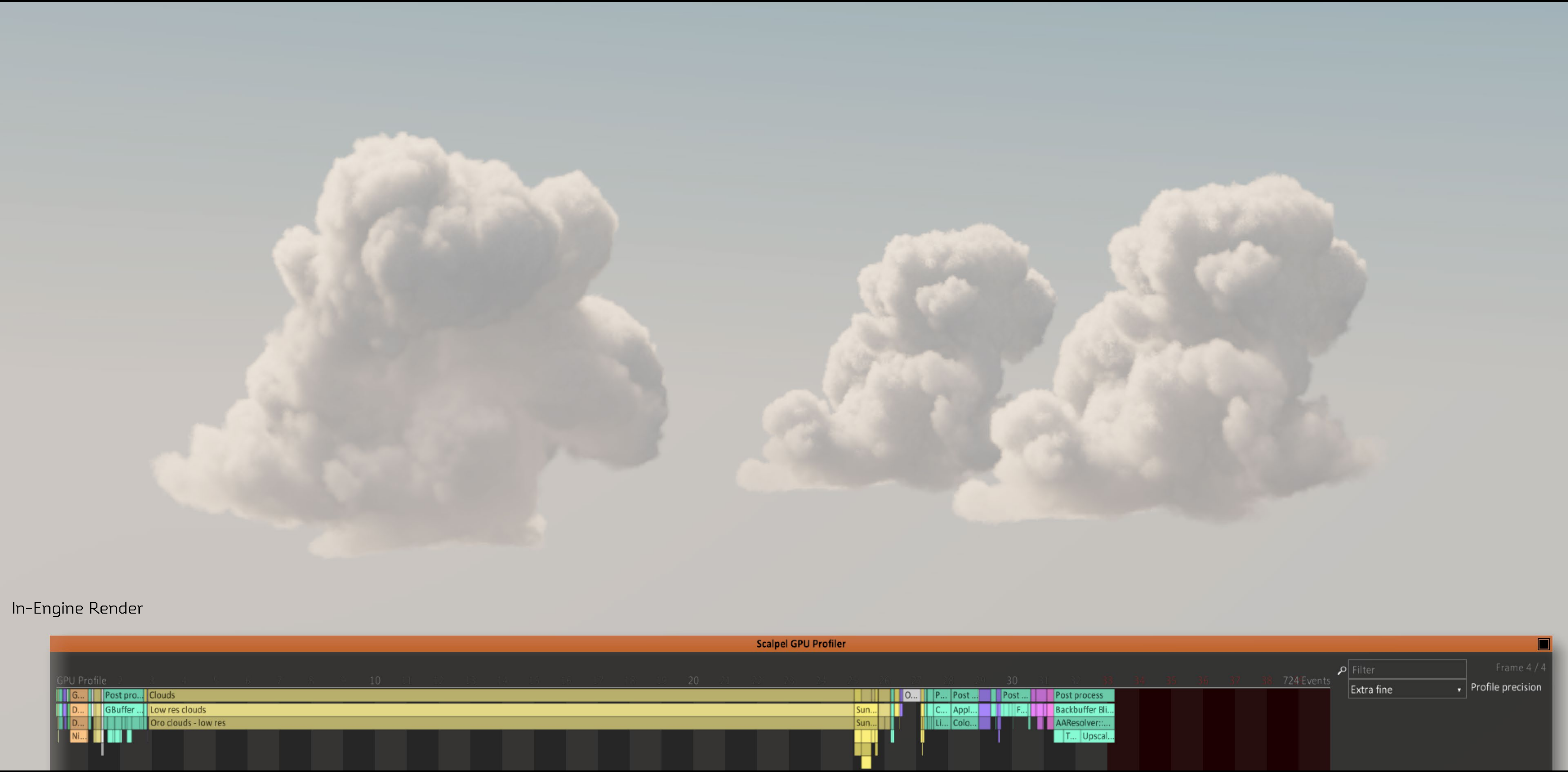




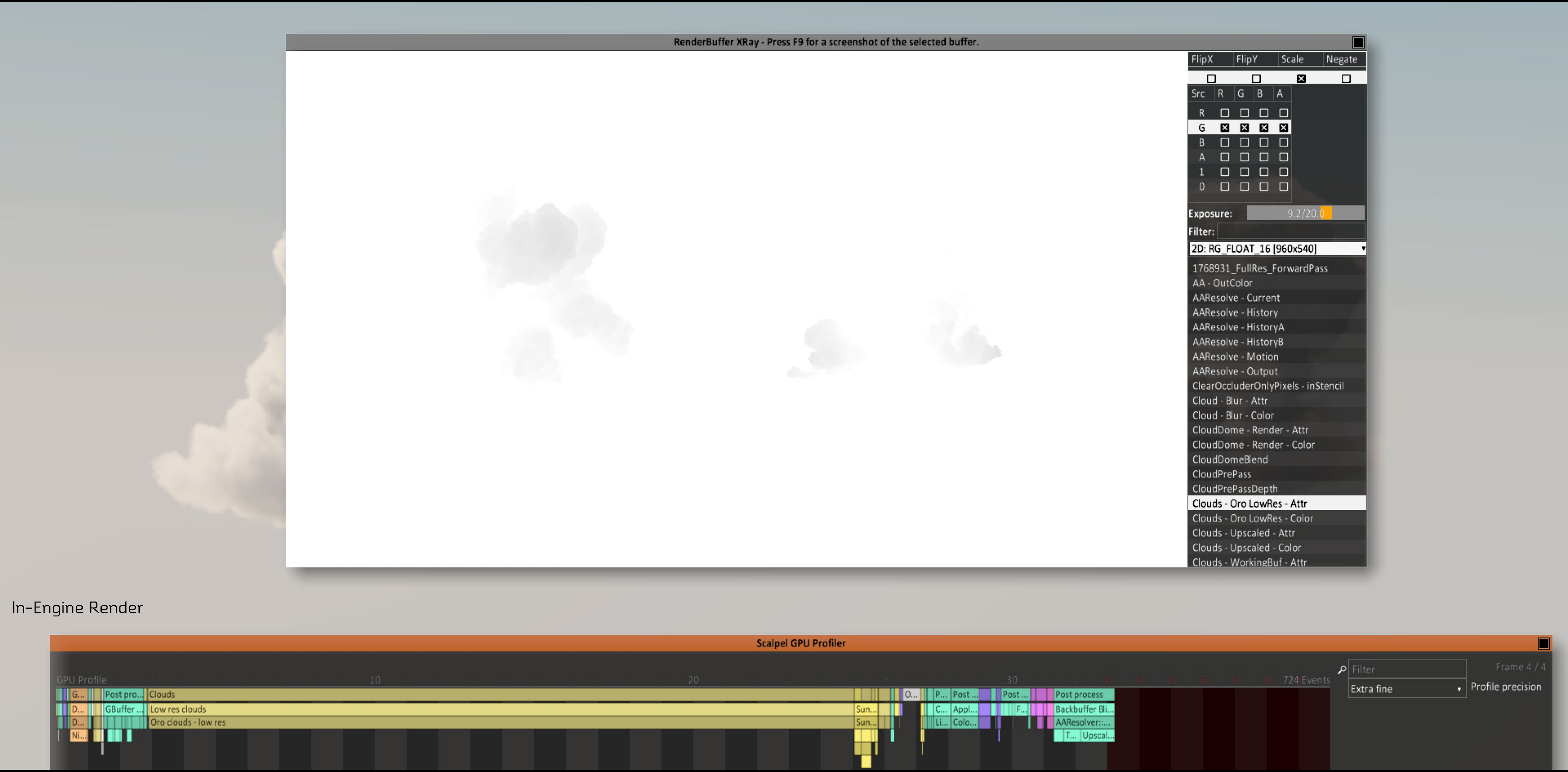
Source-Agnostic Distance Step Mapping

In-Engine Render

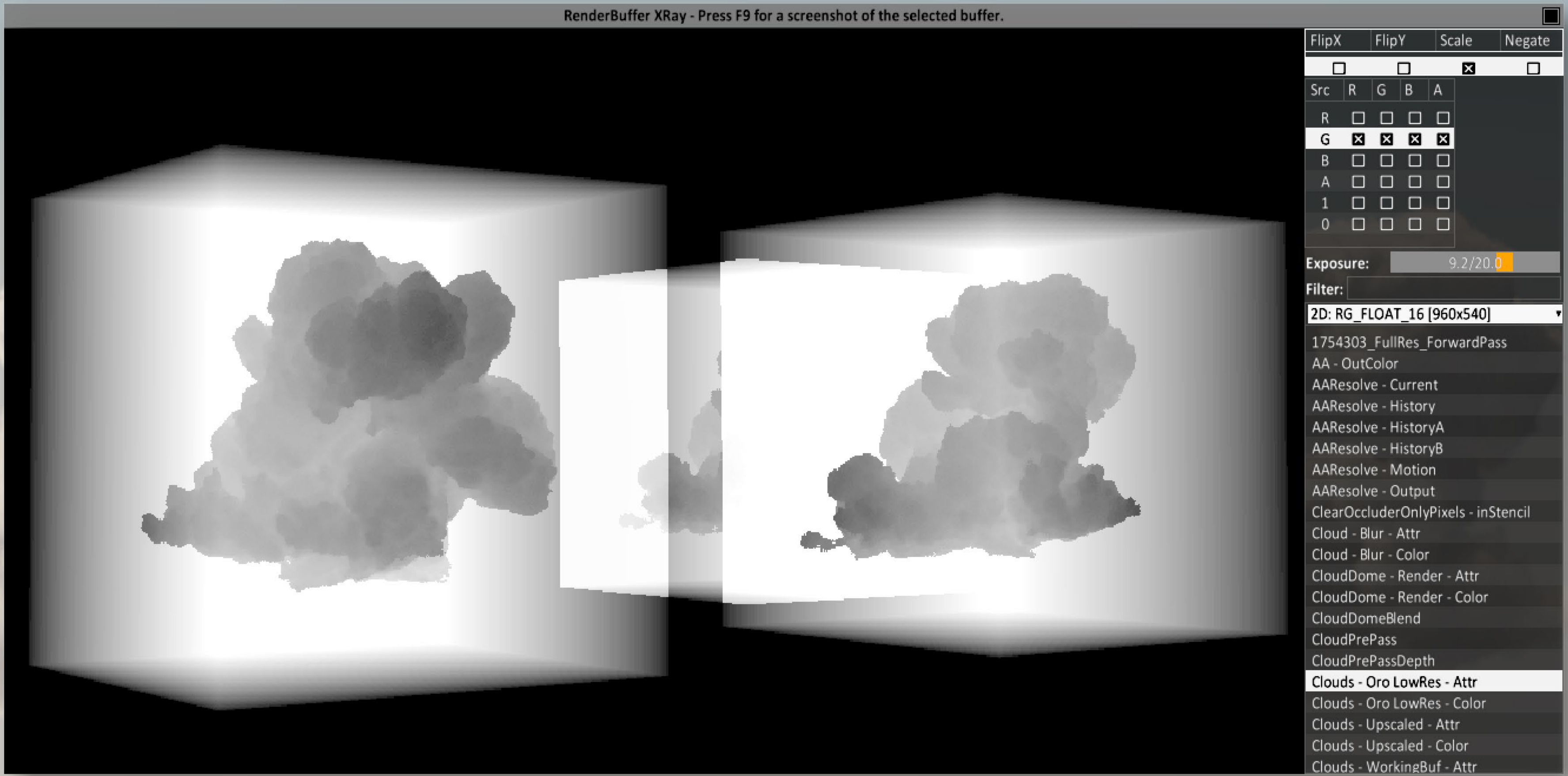
NUBIS EVOLVED / Volume Data Fields



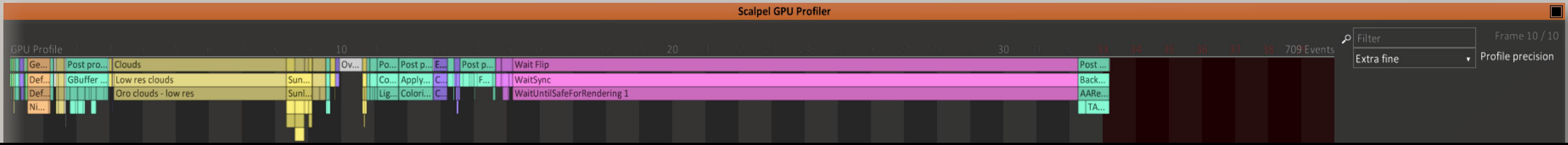
NUBIS EVOLVED / Volume Data Fields

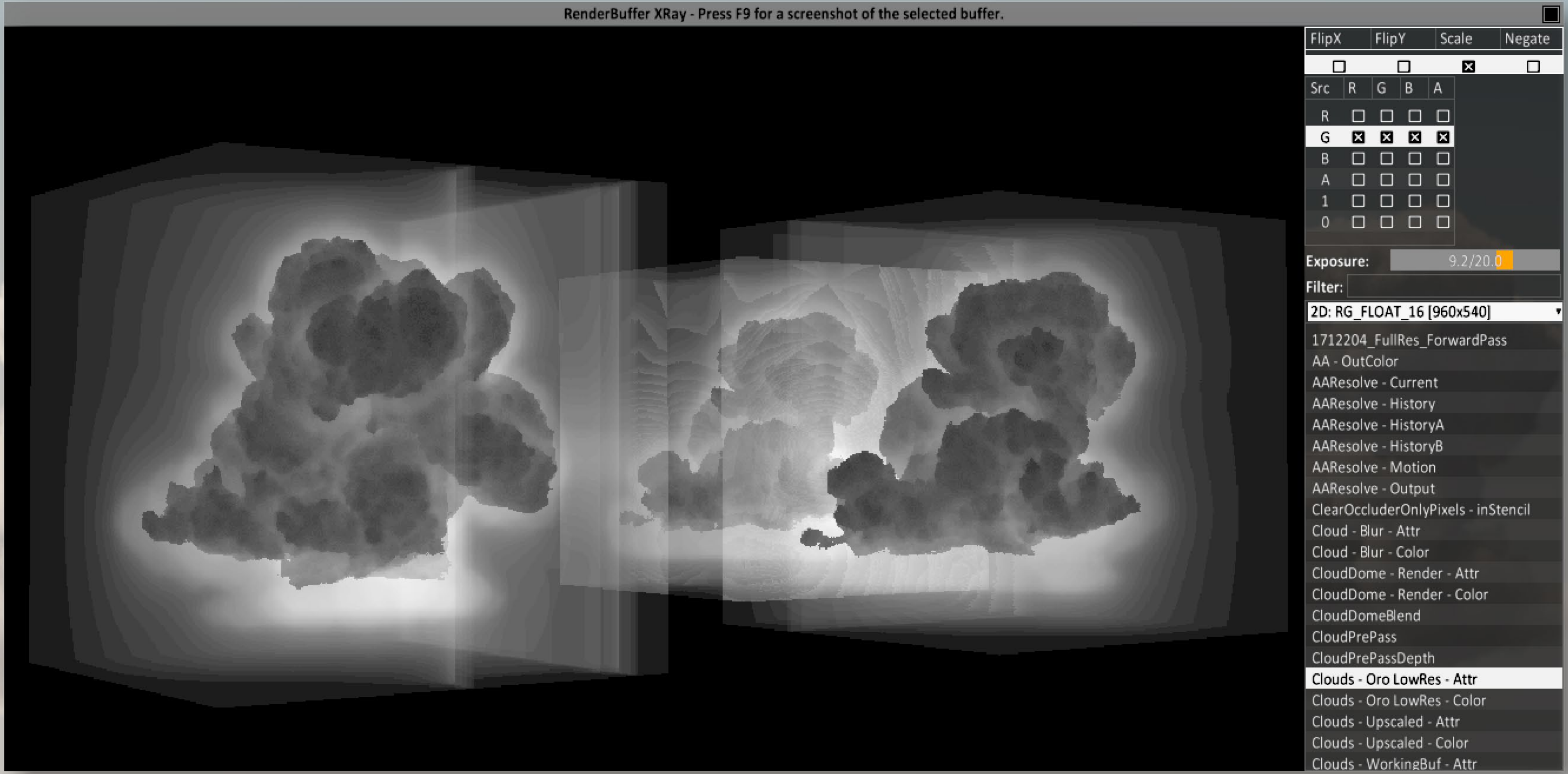


NUBIS EVOLVED / Volume Data Fields

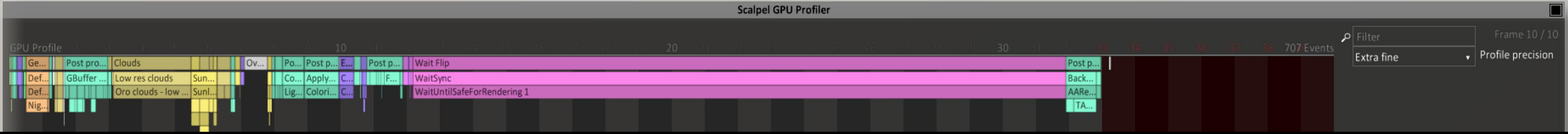


In-Engine Render

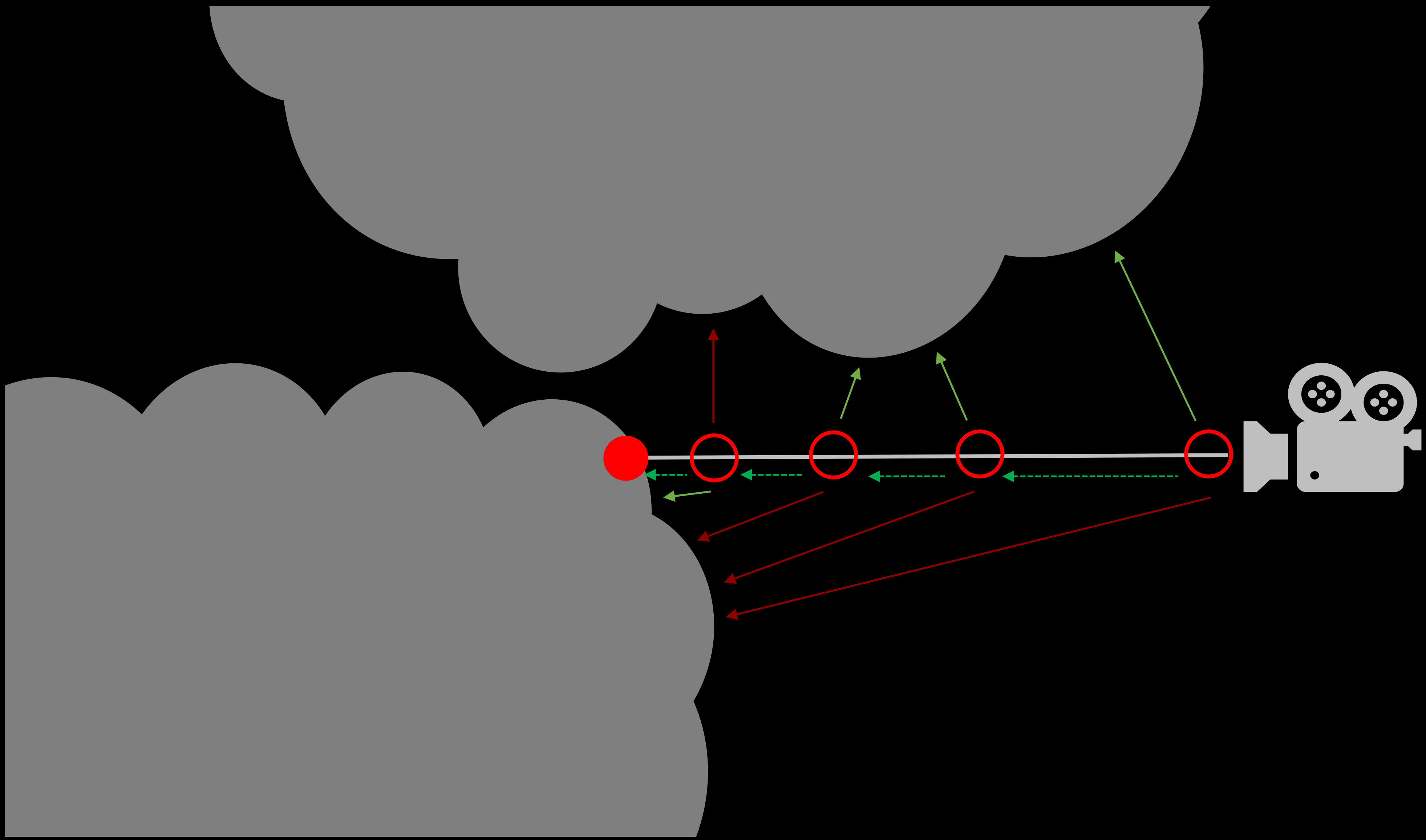




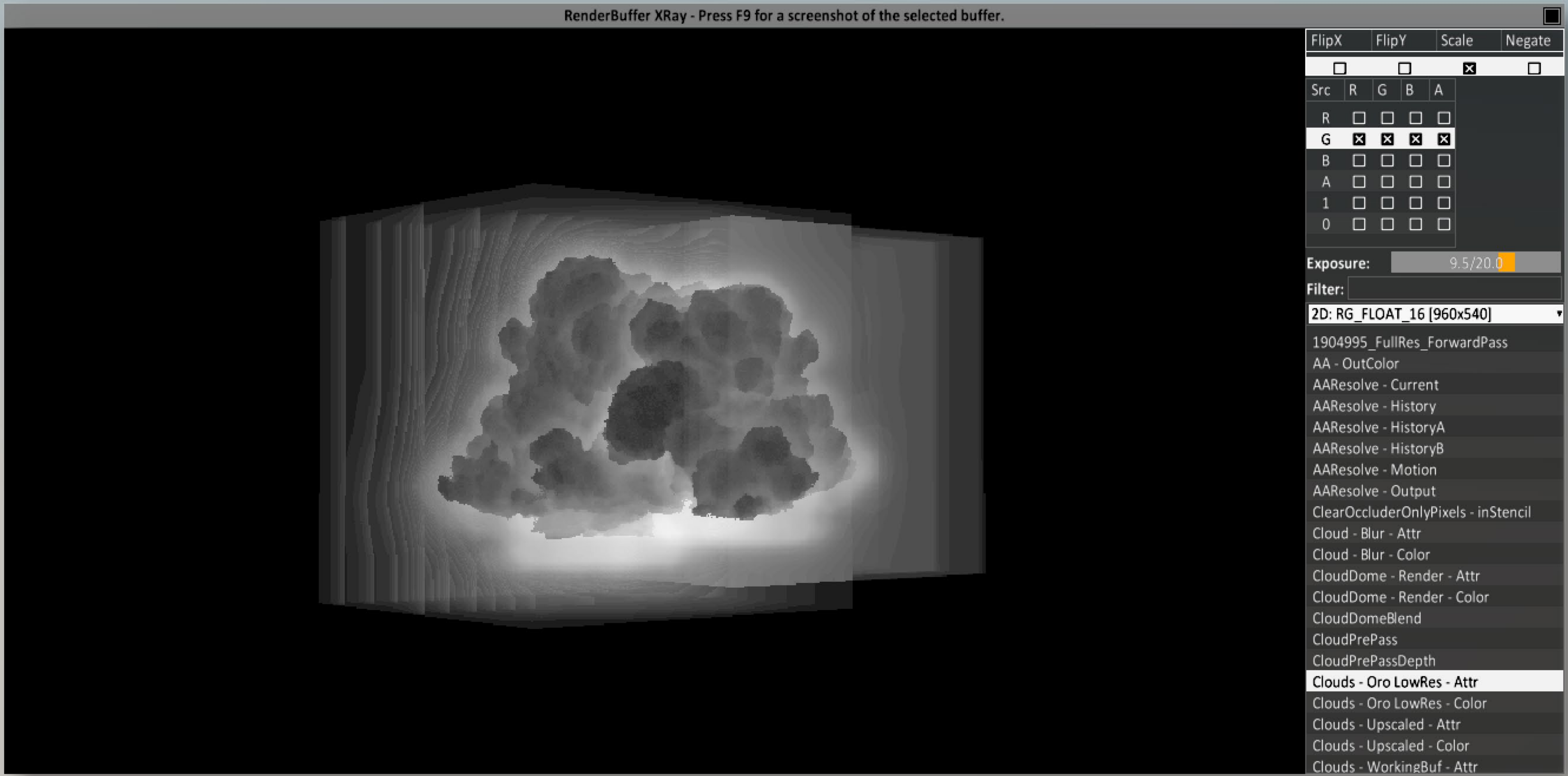
In-Engine Render



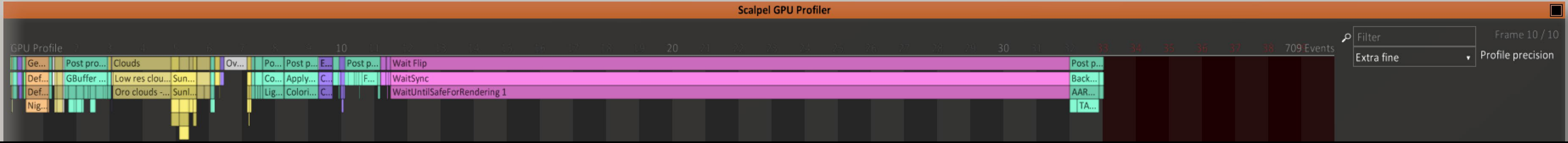
NUBIS EVOLVED / Volume Data Fields



In-Engine Render



In-Engine Render



In-Engine Render

/Thanks

Nathan Vos
James McLaren
Bryan Adams
Giliam de Carpentier

Elco Vossers
Bart van Oosten

Anton Woldhek
Nick van Kleef

Jan-Bart van Beek
Misja Baas
Marijn Giesberts
Roderick van der Steen
Jeroen Krebbers
Niklas Modrow
Alexis Russel
Mark van Berkel
Tim Stobo
Angie Smets
Michiel van der Leew

Rosa, Aidan, Liam and Amelia

/References

Richard Hamblyn, *The Invention Of Clouds*. New York: Picador Reprints, 2011.

Brantley Hargrove, *The Man Who Caught The Storm*. New York: Simon & Schuster, 2018.

Martin Uman, *Lightning*. New York: Dover Publications, 1969.

Augustus Beer, ''Bestimmung der Absorption des rothen Lichts in farbigen Flüssigkeiten'' (Determination of the absorption of red light in colored liquids), *Annalen der Physik und Chemie*, vol. 86, pp. 78–88, 1852.

L. G. Henyey and J. L. Greenstein, "Diffuse radiation in the Galaxy," *Astrophysical Journal*, vol. 93, pp. 78–83, 1941.

John Hart, "Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces", *The Visual Computer*, June, 1995.

Jonathan "Lone Sock" Dummer, Cone Step Mapping: An Iterative Ray Heightfield Intersection Algorithm. 2006.

/Previous Talks

Andrew Schneider. "The Real-Time Volumetric Superstorms of Horizon Forbidden West". *GDC 2022*. SF, USA, Web. 2022.

Andrew Schneider. "Nubis: Real-Time Volumetric Cloudscapes in a Nutshell". *Eurographics*. Delft, NL, Web. 2018.

Andrew Schneider. "Nubis: Authoring The Real-Time Volumetric Cloudscapes Of Horizon Zero Dawn". *ACM SIGGRAPH*. Los Angeles, CA: ACM SIGGRAPH, 2017. Web. 2017.

Andrew Schneider, GPU Pro 7: *Real Time Volumetric Cloudscapes*. p.p. (97-128) CRC Press, 2016.

Andrew Schneider. "The Real-Time Volumetric Cloudscapes Of Horizon Zero Dawn". *ACM SIGGRAPH*. Los Angeles, CA: ACM SIGGRAPH, 2015. Web. 26 Aug. 2015.

/ Guerrilla

/ Atmospheric FX

Twitter: @guerrilla_jobs LinkedIn: Guerrilla Online: www.guerrilla-games.com