

Real-Time Volumetric Cloudscapes for Games (2016)
NUBIS EVOLVED / Background

“Nubification”

“Nubis”

Luke Howard, 1802
NUBIS EVOLVED / Skies / Background

- Structure
- Movement
- Color
“Tropospheric Clouds”

- Cirrostratus
- Cirrus
- Altostratus
- Altocumulus
- Stratus
- Stratocumulus
- Cumulus
- Cumulonimbus
NUBIS EVOLVED / Skies / Structure

Stratus Sub-Layer

Cirrus Sub-Layer
NUBIS EVOLVED / Skies / Structure

Nubis Data Fields

“NDFs”
NUBIS EVOLVED / Skies / Structure

2D NDF Mapping

16km

W
N
E
S

16km
Vertical Profile Model
“Vertical Profile Model NDFs”

- Cloud Min Height
- Cloud Max Height
- Cloud Coverage
- Cloud Bottom Type
- Cloud Top Type

The diagram shows a vertical profile model with the following dimensions:

- Cloud Height: 2048m
- Cloud Bottom: 256m
- Cloud Top: 2048m
- Cloud Coverage: 2048m - 256m

This model is used to simulate atmospheric effects in games.
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Density
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Density

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
In-Engine Render
Stratocumulus
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Density

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
Bottom Type = 0.0
“Vertical Profile”
float dimensional_profile = vertical_profile * cloud_coverage;
float cloud_density = saturate(cloud_noise_composite - (1.0 - dimensional_profile));
// Deform sample coordinates
float3 noise_sample_position = sample_position - wind_direction * scroll_offset;
NUBIS EVOLVED / Skies / Vertical Profile Model / Rendering

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
// Define step size constants
float near_step_size = 3.0;
float far_step_size_offset = 60.0;
float step_adjustment_distance = 16384.0;

// Calculate distance-based step size
float step_size = near_step_size + ((far_step_size_offset * distance_from_camera) / step_adjustment_distance);
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Light
Light Energy = **Direct Scattering** + **Ambient Scattering**
Direct Scattering = (Transmittance \times \text{Primary Scattering Phase}) + (\text{Multiple Scattering} \times \text{Secondary Scattering Phase})
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Light
Transmittance \( T \) = \( e^{-d} \)

Augustus Beer (1852)  
J.H. Lambert (1760)

**NUBIS EVOLVED** / Skies / Vertical Profile Model / Modeling Light

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Light

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));
}
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Light

Photograph
float ms_volume = Remap(dimensional_profile* step_size, 0.1, 1.0, 0.0, 1.0) * pow(cloud_coverage * cloud_type, 0.25);
ms_volume *= pow(attenuated_light, cMultipleScatteringDepthPower);
ms_volume *= pow(height_fraction, cMultipleScatteringHeightPower);
NUBIS EVOLVED / Skies / Vertical Profile Model / Modeling Light

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course

In-Engine Render
float ambient_scattering = pow(1.0 - dimensional_profile, 0.5);
// 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;
NUBIS EVOLVED / Skies / 2.5-D Model

2.5-D Model
“2.5D NDF Definition”

Cloud Coverage

Cloud Type

NUBIS EVOLVED / Skies / 2.5-D Model / Structure

2048m

Height

256m
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);
NUBIS EVOLVED / Skies / 2.5-D Model / Modeling Density

In-Engine Render
NUBIS EVOLVED / Skies / 2.5-D Model / Modeling Light

Cirrus Sub-Layer
NUBIS EVOLVED / Skies / Authoring

- NDF Generator
- Influence NDFs
- NDF Editor

"Authored NDFs"
NUBIS EVOLVED / Skies / Authoring / NDF Editor
NUBIS EVOLVED / Skies / Authoring

NDF Generator

NDF Editor

In-Engine Renders

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / Skies / Authoring / Influence NDFs

In-Engine Render
NUBIS EVOLVED / Skies / Authoring / Influence NDFs

"Influence NDF Definition"

"Vertical Profile Model NDF Definition"

"2.5D NDF Definition"

Influence Mask

Influence Mask
San Francisco

Cumulus Procedural NDFs
+ Regional Influence NDFs

Stratocumulus Procedural NDFs
+ Regional Influence NDFs
NUBIS EVOLVED / Skies / Authoring / Influence NDFs

Current Weather

- 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
NUBIS EVOLVED / Skies / Authoring / Influence NDFs
NUBIS EVOLVED / Environments
NUBIS EVOLVED / Environments / Goals

- Open World
- Performant
- Detailed
NUBIS EVOLVED / Environments / Goals
NUBIS EVOLVED / Environments / Goals

Paintings by Albert Bierstadt

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
heart of the clouds and then fall back, some distance away, in the unusual form of a shower of hatchets.

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 summits 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...

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 altocumulus 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...
// 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;
NUBIS EVOLVED / Environments / Problems
Orographic clouds
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);
NUBIS EVOLVED / Environments / The Envelope Model / Modeling Density

Type = 1.0
In-Engine Render
float3 noise_sample_pos = inSamplePosition + float3(0.0, 0.0, (1.0 - saturate((max_height – min_height) * 0.0125)) * 40.0 );
NUBIS EVOLVED / Environments / The Envelope Model / Modeling Density
// 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;
NUBIS EVOLVED / Environments / The Envelope Model / Modeling Density

"Envelope Model NDF Definition"

- Cloud Min Height
- Cloud Max Height
- Cloud Type
- Cloud Density

2048m

256m

Height
NUBIS EVOLVED / Environments / The Envelope Model / Authoring

Houdini Viewport

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / Environments / The Envelope Model / Modeling Lighting
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;
NUBIS EVOLVED / Environments / The Envelope Model / Modeling Lighting
NUBIS EVOLVED / Environments / The Envelope Model / Ray-Marching
Sphere Tracing

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

"Cone Step Mapping: An Iterative Ray-Heightfield Intersection Algorithm."
NUBIS EVOLVED / Environments / The Envelope Model / Ray-Marching
NUBIS EVOLVED / Environments / The Envelope Model / Ray-Marching
4.2 Milliseconds
In-Engine Render

**1.3 Milliseconds**

Scalenel GPU Profiler

1,800 Events

Extra fine
Profile precision

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
"Envelope Model NDF Definition"

- Cloud Min Height
- Cloud Max Height
- Cloud Type
- Cloud Density
- Cloud Distance
- Upper Angle
- Lower Angle
NUBIS EVOLVED / Environments / The Envelope Model / Locations

In-Engine Render

Zion
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
©Board of Trustees, University of Illinois
NUBIS EVOLVED / VFX / Of Supercells...

Photographs
"Superstorm Influence NDFs"

Cloud Min Height
Cloud Max Height
Cloud Coverage
Cloud Bottom Type
Cloud Top Type
Superstorm Mask
NUBIS EVOLVED / VFX / Modeling Superstorm Density

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / VFX / Modeling Superstorm Density / The Mesocyclone

In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
Bottom Type = 0.0
NUBIS EVOLVED / VFX / Modeling Superstorm Density / The Mesocyclone

In-Engine Render
NUBIS EVOLVED / VFX / Modeling Superstorm Density / The Mesocyclone

NDF

In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / VFX / Modeling Superstorm Density / The Mesocyclone

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]));
float noise = SampleNoise(GetRotatedPosition(sample_position, superstorm_center, time_offset * ring_rotation_speed[n] + ring_skew[n]));
NUBIS EVOLVED / VFX / Animating Superstorm Density / The Mesocyclone
NUBIS EVOLVED / VFX / Animating Superstorm Density / The Mesocyclone
NUBIS EVOLVED / VFX / Animating Superstorm Density / The Mesocyclone
// 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);
NUBIS EVOLVED / VFX / Animating Superstorm Density / The Mesocyclone
// 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);
Anvil

NUBIS EVOLVED / VFX / Modeling Superstorm Density / The Anvil

Photograph
NUBIS EVOLVED / VFX / Modeling Superstorm Density / The Anvil

Vertical Gradient

In-Engine Render
In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / VFX / Modeling Superstorm Density

In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
Optical Thickness ($d$)

Transmittance ($T$)

$T = e^{-5d}$

$T = e^{-10d}$

$T = e^{-20d}$

NUBIS EVOLVED / VFX / Superstorm Lighting / Attenuation

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
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));
NUBIS EVOLVED / VFX / Superstorm Lighting / Attenuation

In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
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);
NUBIS EVOLVED / VFX / Superstorm Lighting / Ambient

In-Engine Render
NUBIS EVOLVED / VFX / Superstorm Lighting / Internal Glow
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;
Ground Discharge

Intra-Cloud

Ground Discharge
NUBIS EVOLVED / VFX / Superstorm Lightning Effects

In-Engine Render
NUBIS EVOLVED / VFX / Superstorm Lightning Effects
Intra-Cloud Intra-Cloud + Ground Discharge

Repeat, build intensity

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

NUBIS EVOLVED / VFX / Superstorm Lightning Effects
NUBIS EVOLVED / VFX / Superstorm Lightning Effects

No Solution

Our Solution

In-Engine Renders

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / VFX / Superstorm Lightning Effects

Lightning System

Generate Position → Trigger Flash Animation

Flash Position

Animated Flash Intensity

Nubis Renderer

Generate Stable Mask → Multiply Flash Intensity by Mask

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / VFX / Superstorm Lightning Effects

In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
<table>
<thead>
<tr>
<th>Feature</th>
<th>PlayStation 4</th>
<th>PlayStation 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Resolution</td>
<td>960 x 540</td>
<td>1920 x 1080</td>
</tr>
<tr>
<td>Light Ray Samples</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>View Ray Samples</td>
<td>60 - 90</td>
<td>96 - 180</td>
</tr>
<tr>
<td>Blur Scale (Pixels)</td>
<td>2x</td>
<td>1x</td>
</tr>
<tr>
<td>Noise Texture MIP Level</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

NUBIS EVOLVED / VFX / Scaling PS4 & PS5

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
<= 4 milliseconds

<= 2-3 milliseconds

In-Engine Renders
In-Engine Render
Tropospheric / Superstorm

Near Orographic

Far Orographic
while (...) {
    distance_sum += d[n] * sample_opacity;
}

float weighted_sum = distance_sum / opacity_sum;
cloud_color = lerp(cloud_color, atmospherics_color, weighted_sum);
Clouds in the skies of *Rio*

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

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.

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 parallel, and could not be in properly...
In-Engine Renders

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
NUBIS EVOLVED / Volume Data Fields

“NVDF Definition”

Cloud Density

Cloud Distance

Vertical Slices
Source-Agnostic Distance Step Mapping
NUBIS EVOLVED / Volume Data Fields

In-Engine Render

SIGGRAPH 2022 Advances in Real-Time Rendering in Games course
In-Engine Render
NUBIS EVOLVED / Volume Data Fields

In-Engine Render
In-Engine Render

NuBias Evolved / Volume Data Fields
In-Engine Render
In-Engine Render

NUBIS EVOLVED / Volume Data Fields
NUBIS EVOLVED / Volume Data Fields

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


/Previous Talks


