



Raytracing Voxels

In Teardown and beyond

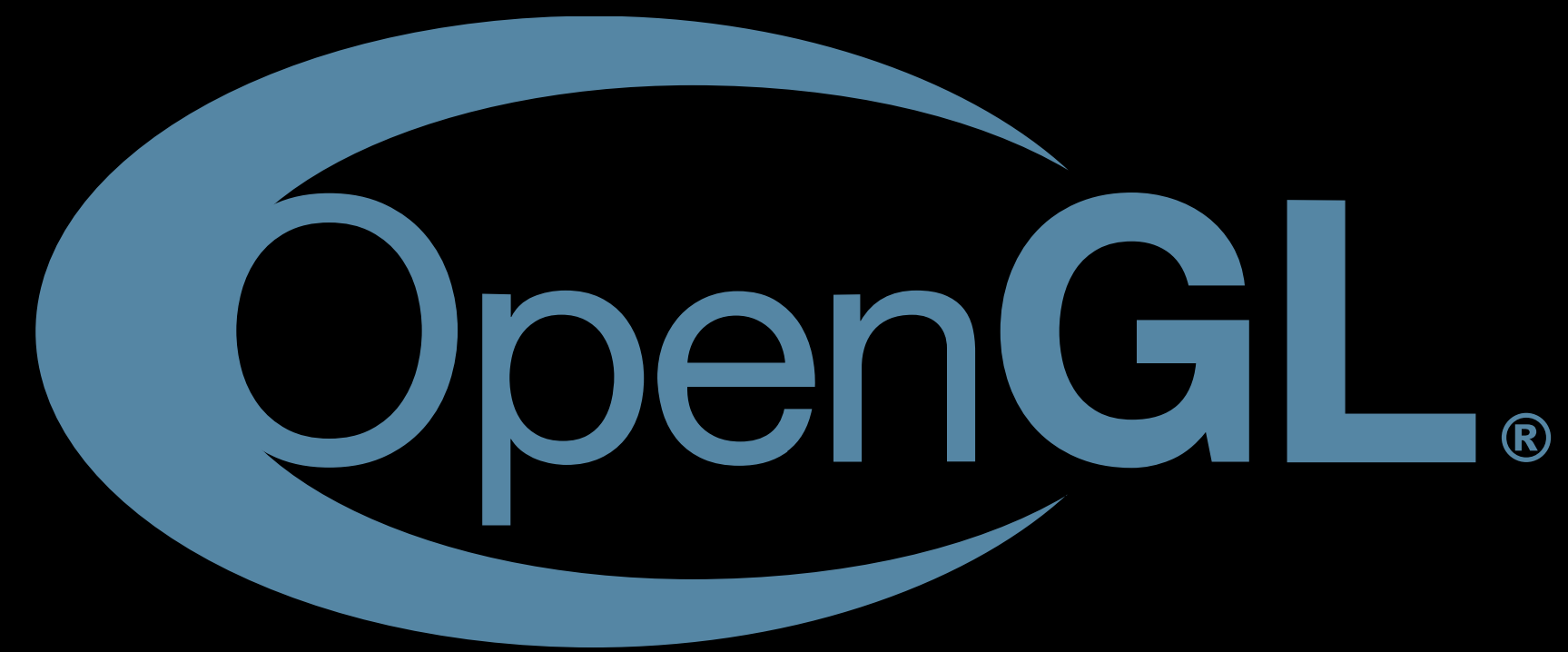
Dennis Gustafsson & Gabe Rundlett – Tuxedo Labs

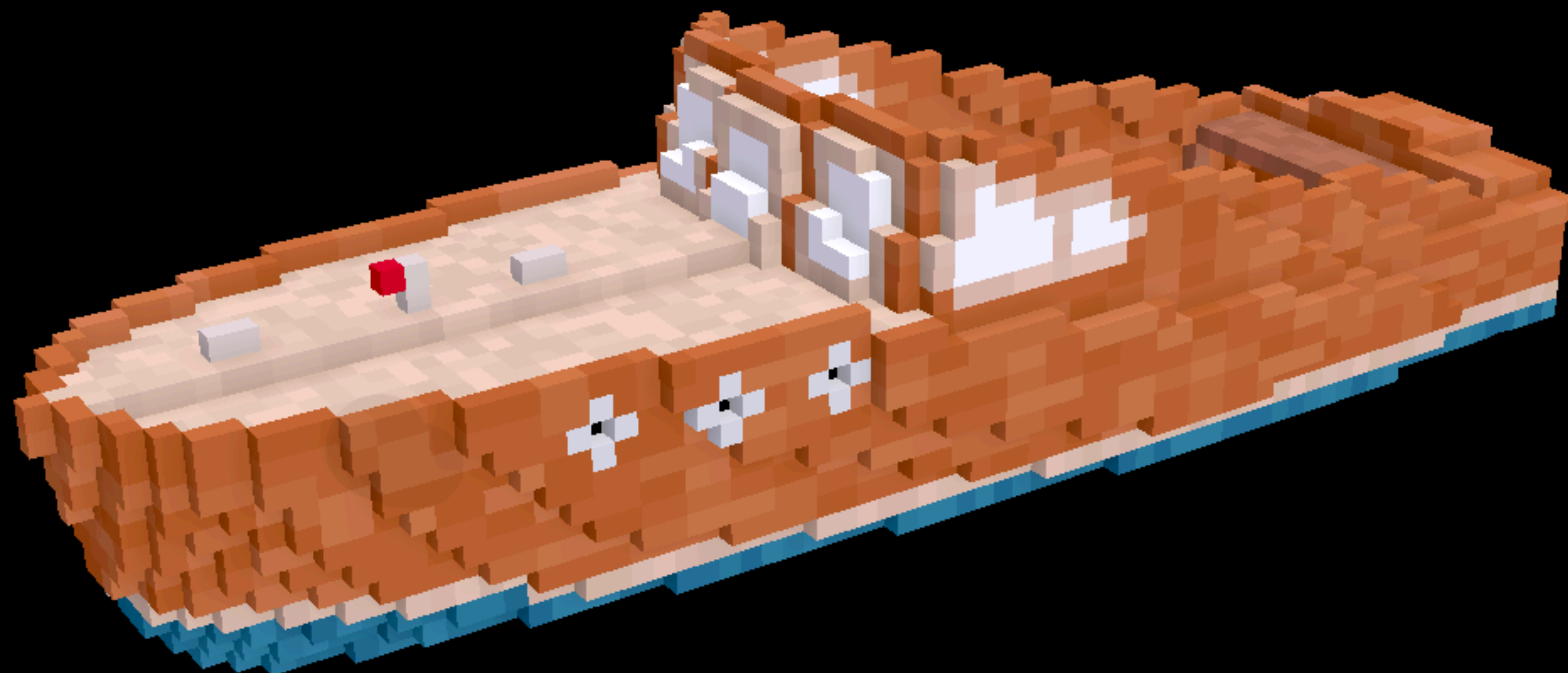


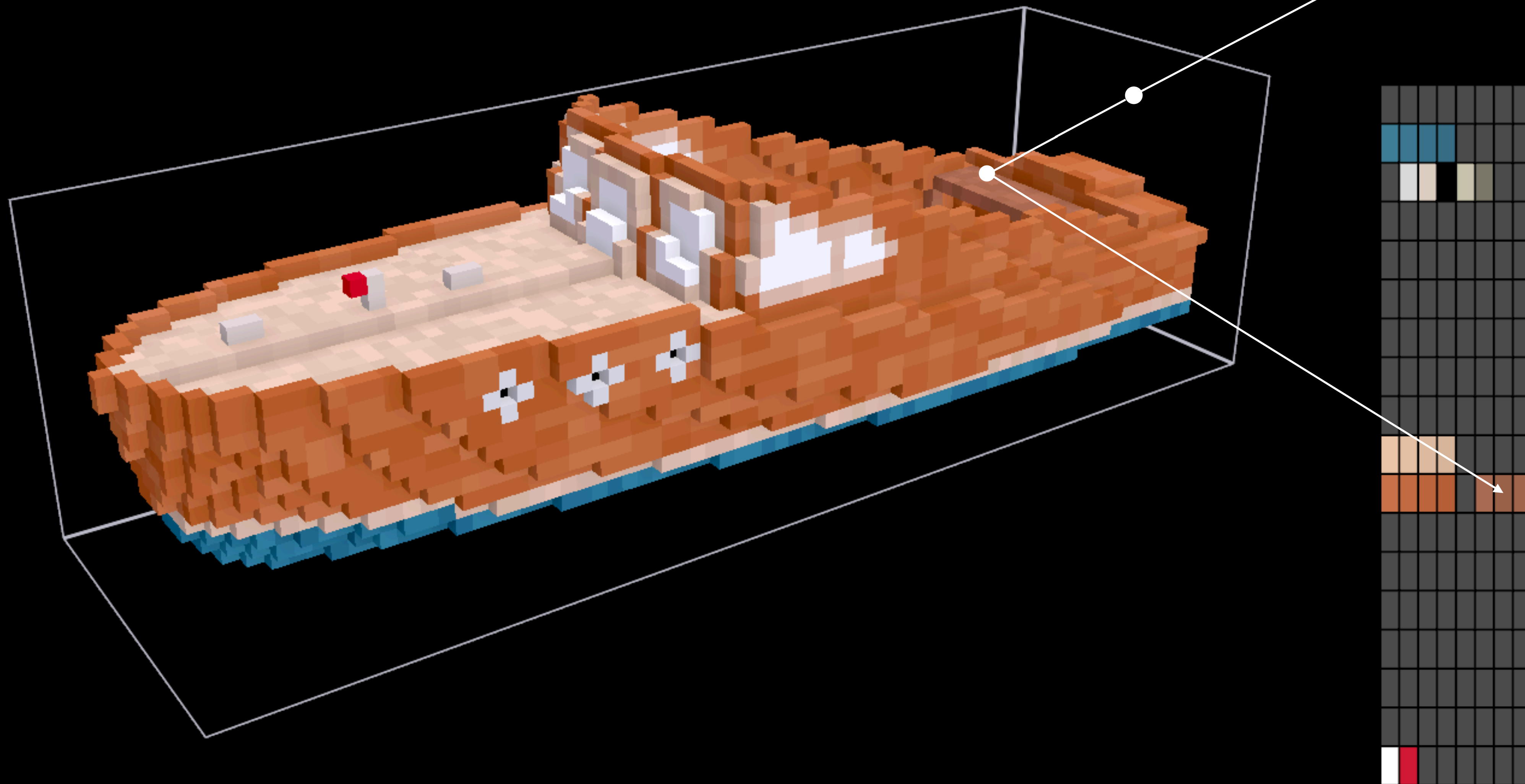
Graphics Programming Conference, November 18-20, Breda

2025



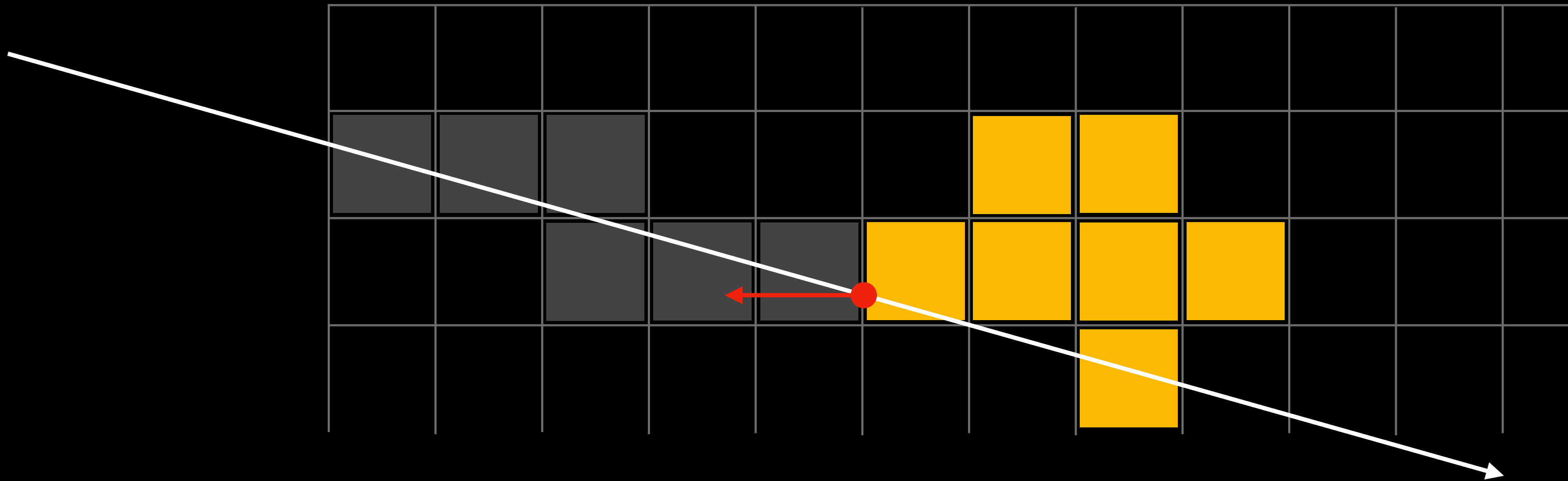






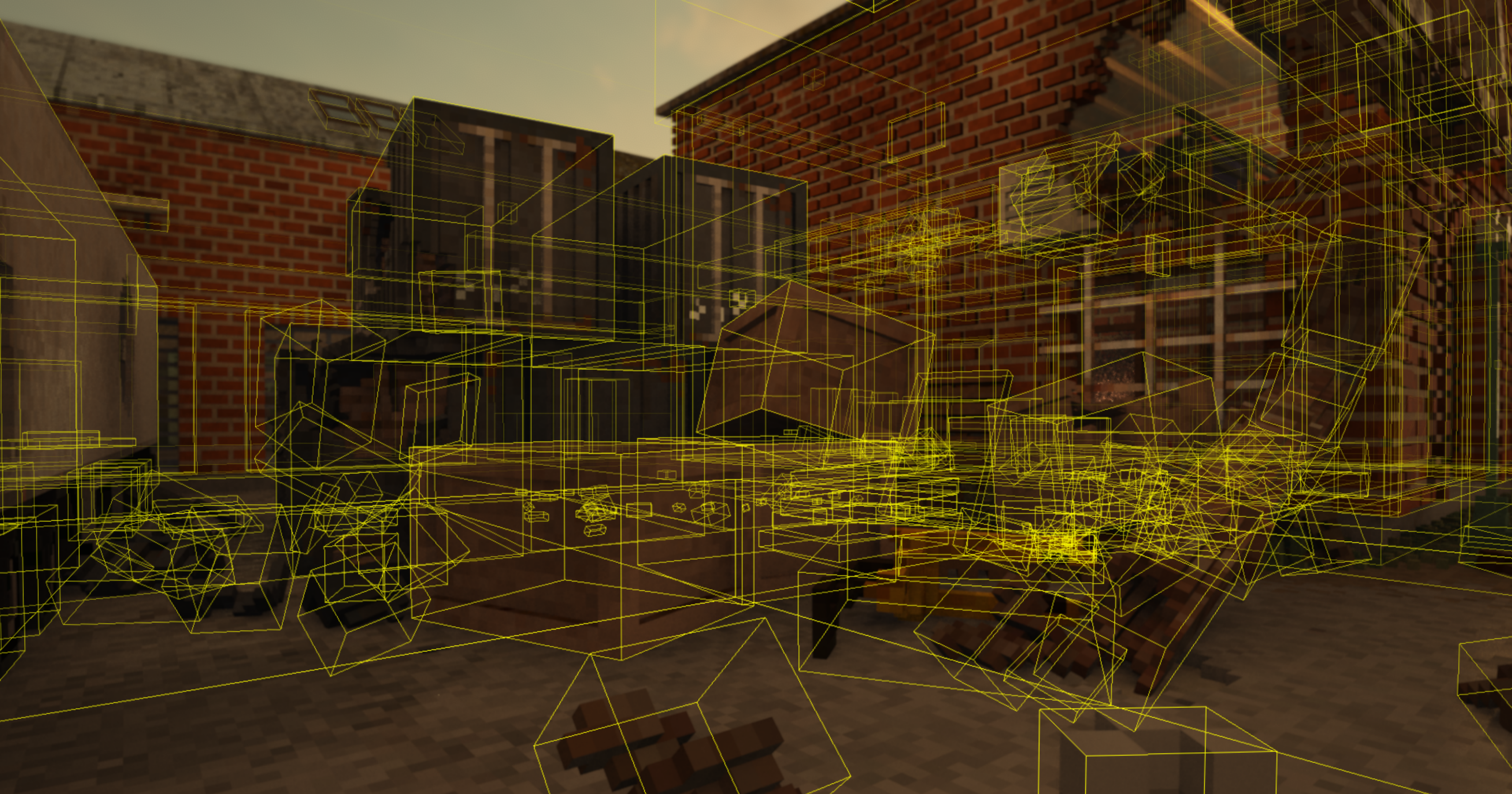
Fast Voxel Traversal Algorithm

Amanatides & Woo (1987)



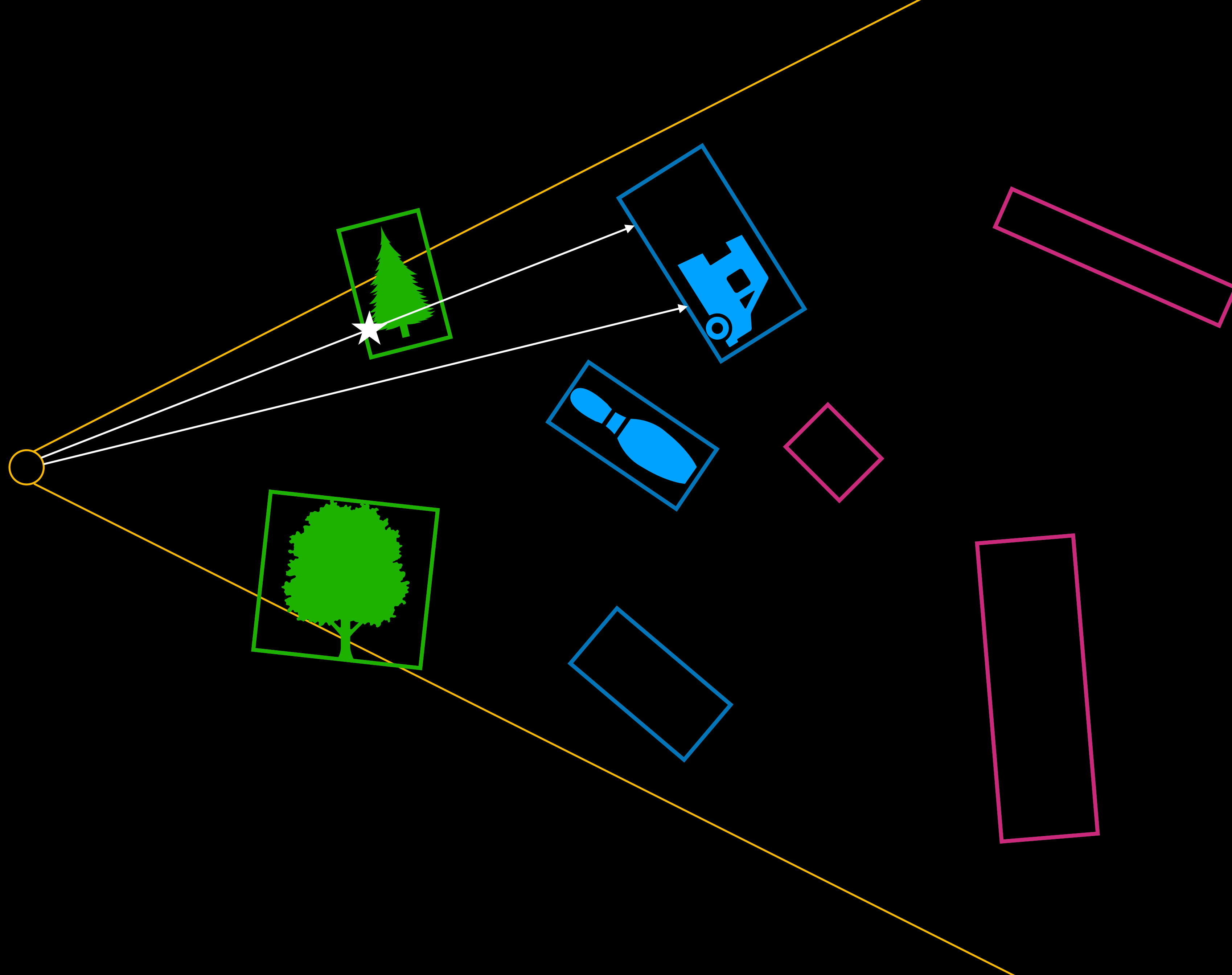






Graphics Programming Conference, November 18-20, Breda

2025

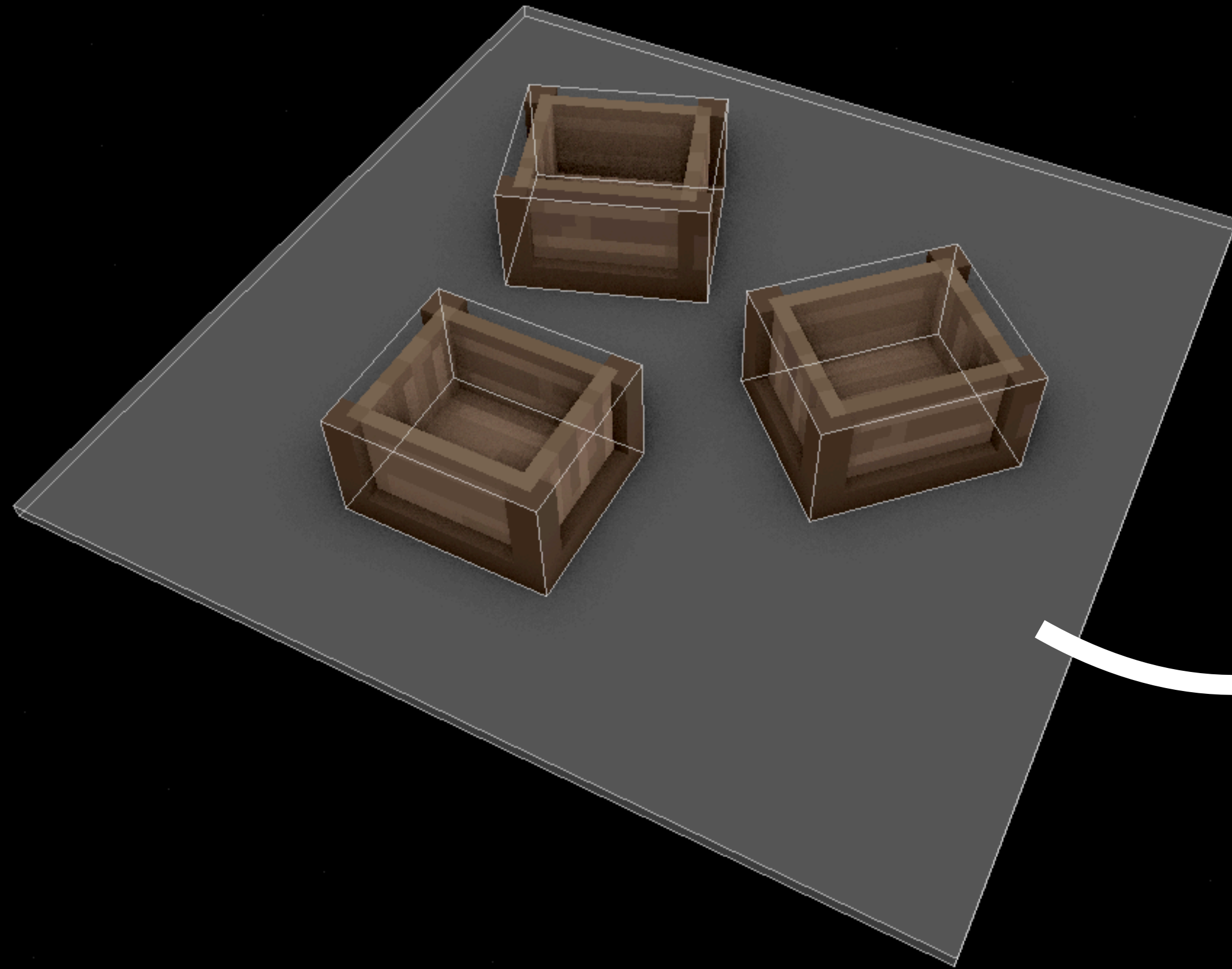




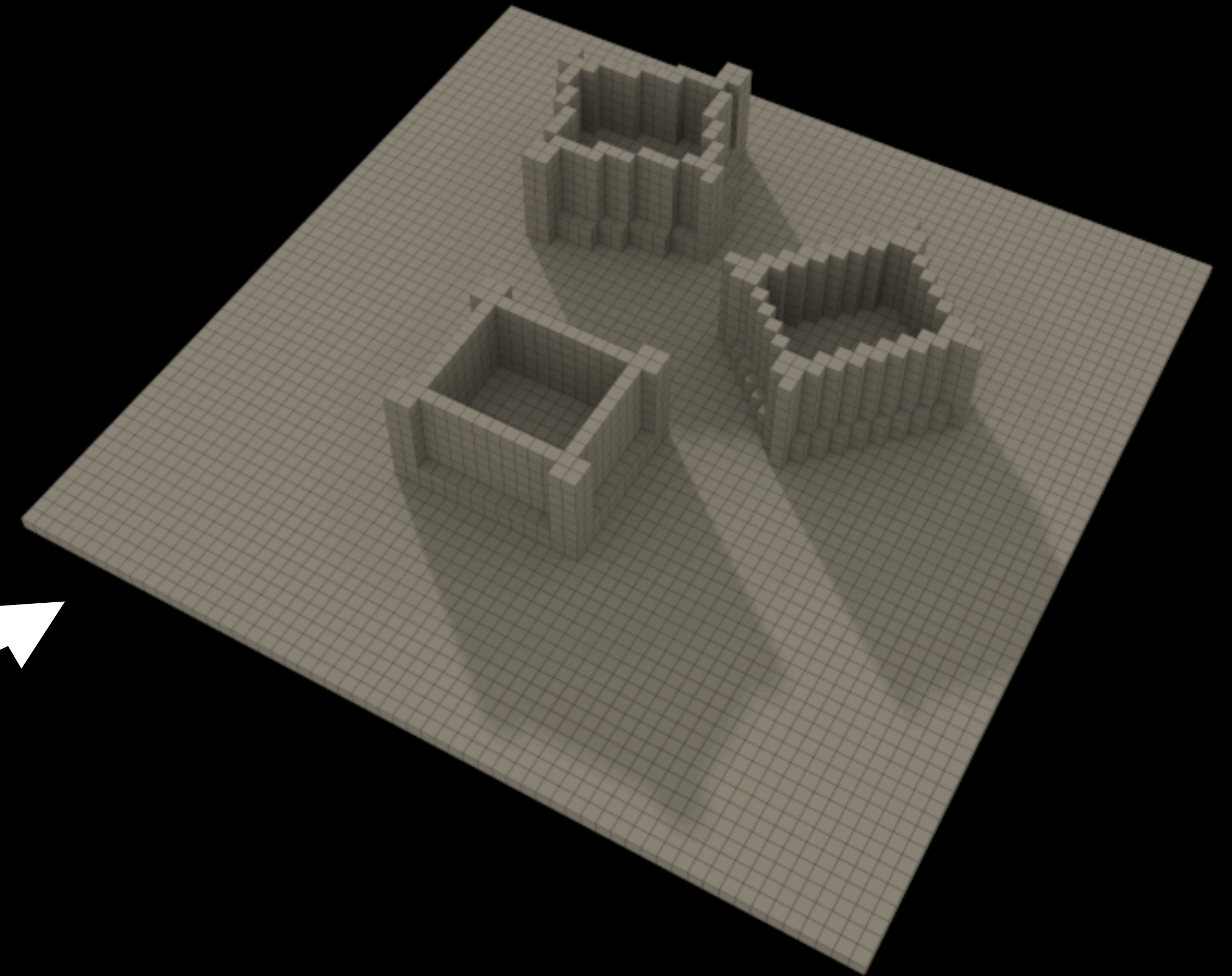
Lighting



Shadow volume

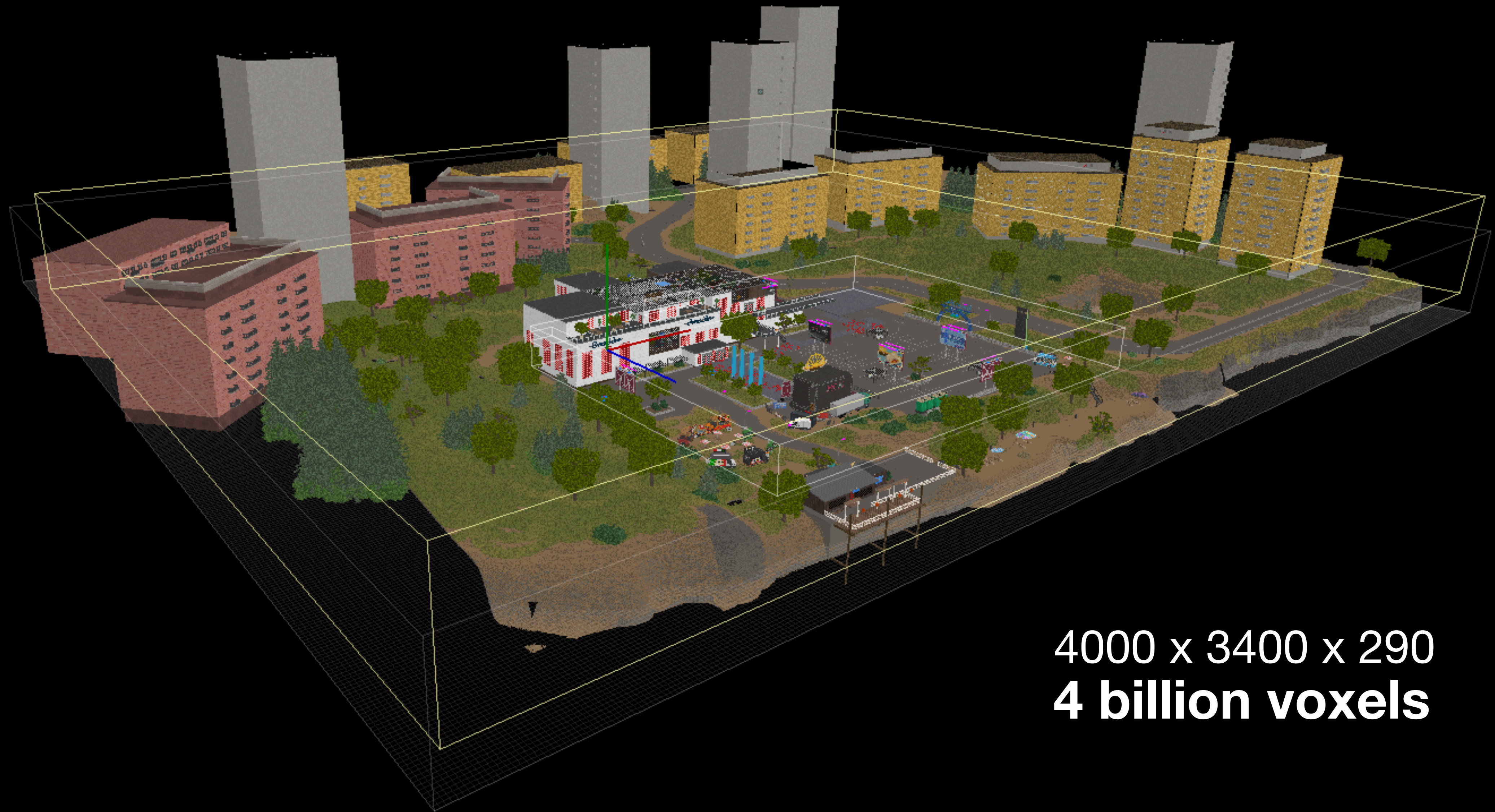


4 objects, individually transformed



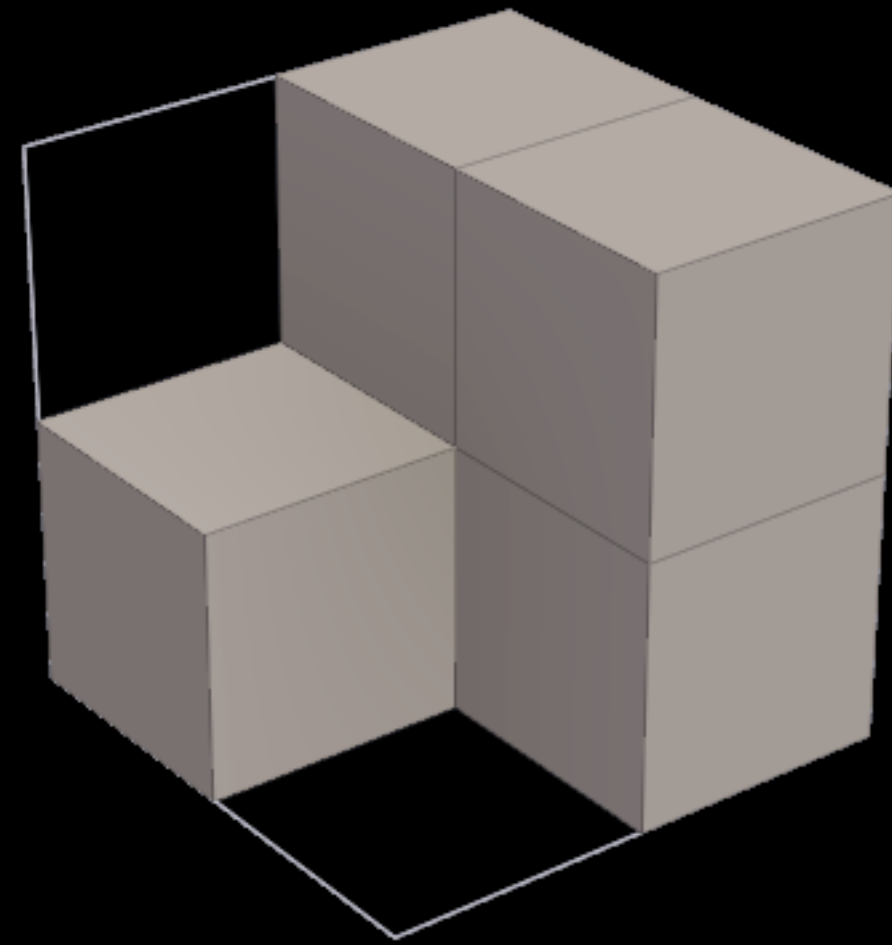
Single large axis-aligned volume

Shadow volume



4000 x 3400 x 290
4 billion voxels

Shadow volume

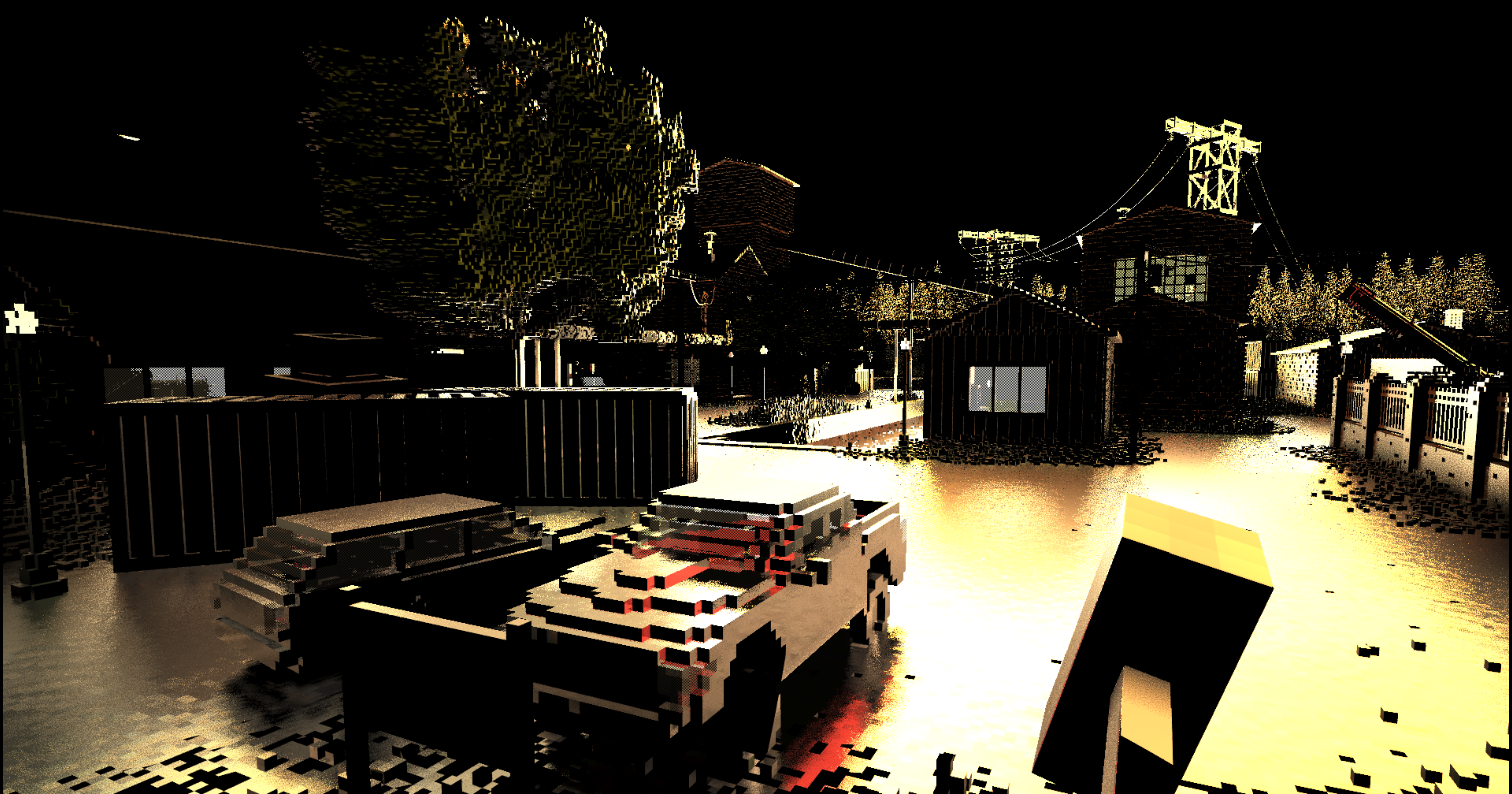


$2 \times 2 \times 2 = 8$ voxels per byte

$4000 \times 3400 \times 290$
4 billion voxels = $2000 \times 1700 \times 145$
500 Mb VRAM



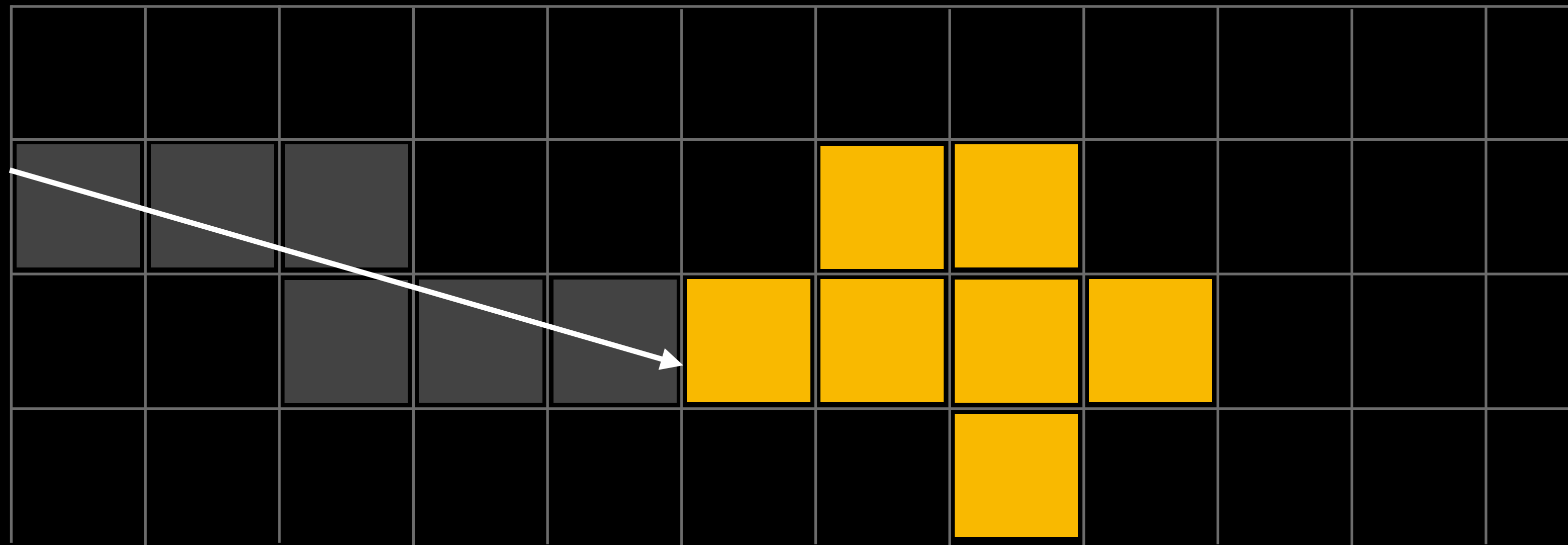




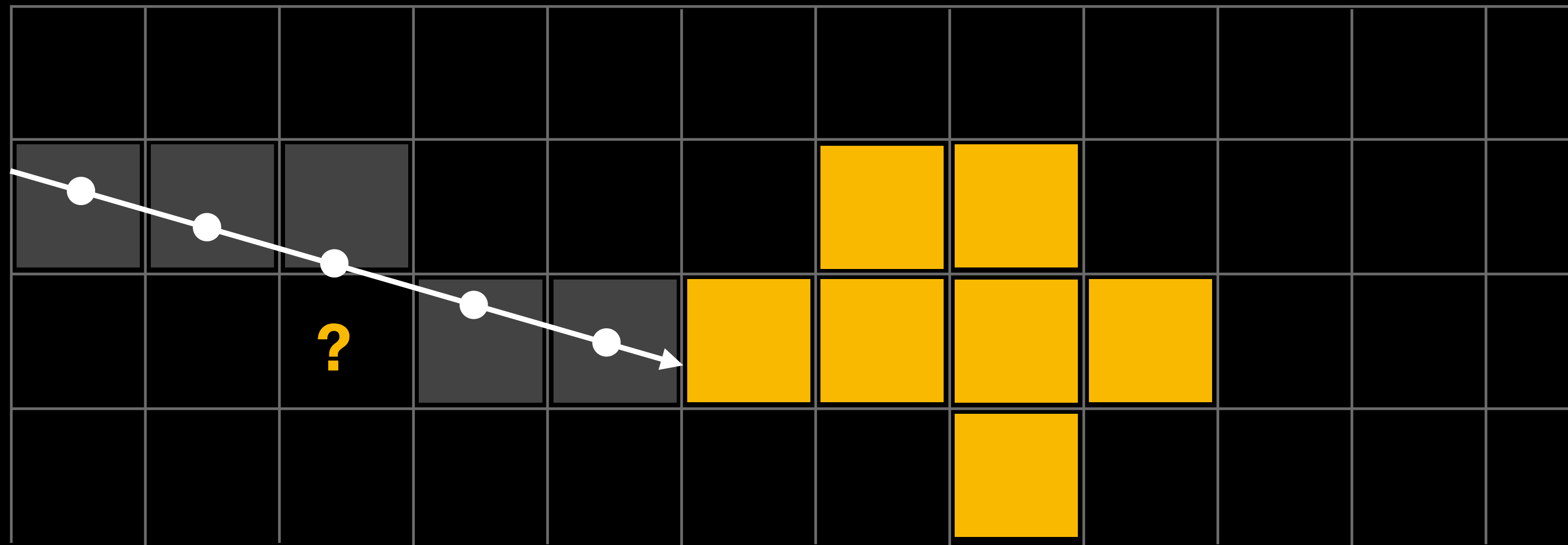




High Quality Trace



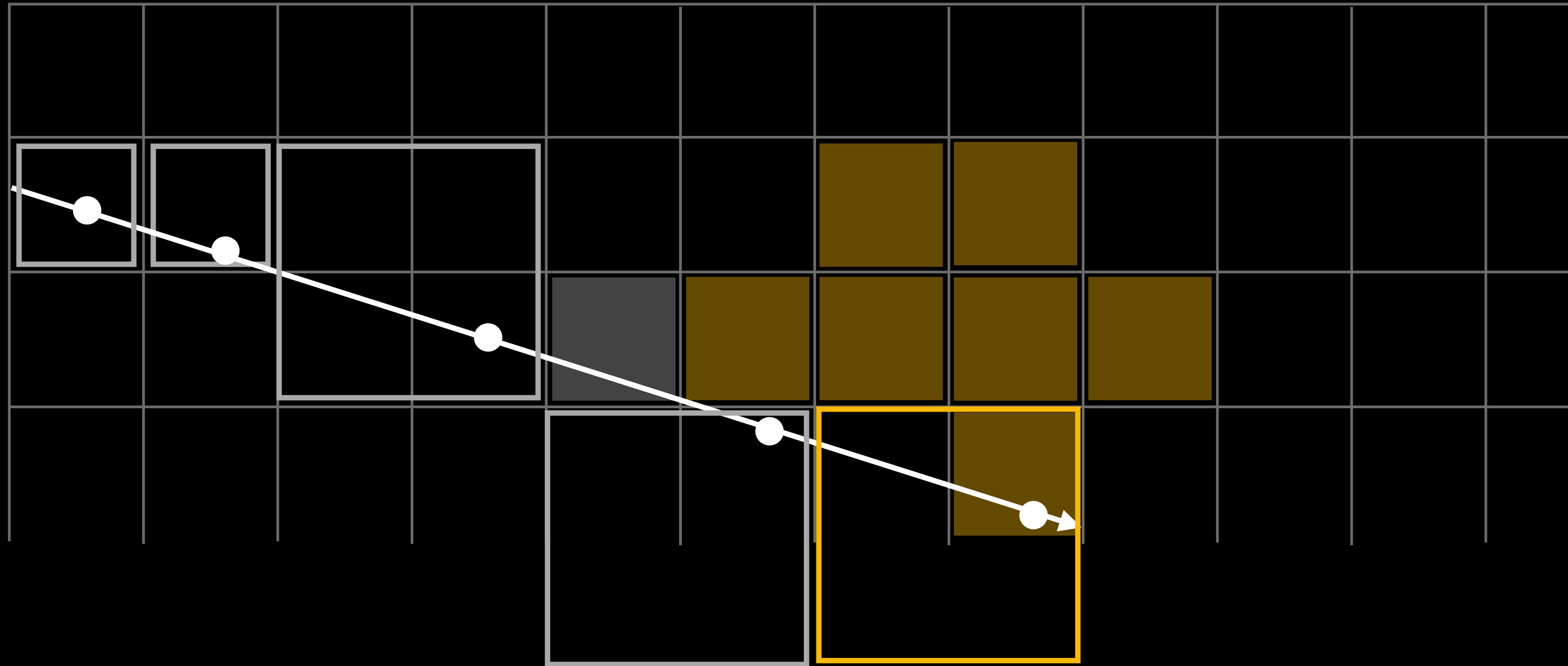
Medium Quality Trace



Medium Quality Trace



Low Quality Trace



Frame breakdown

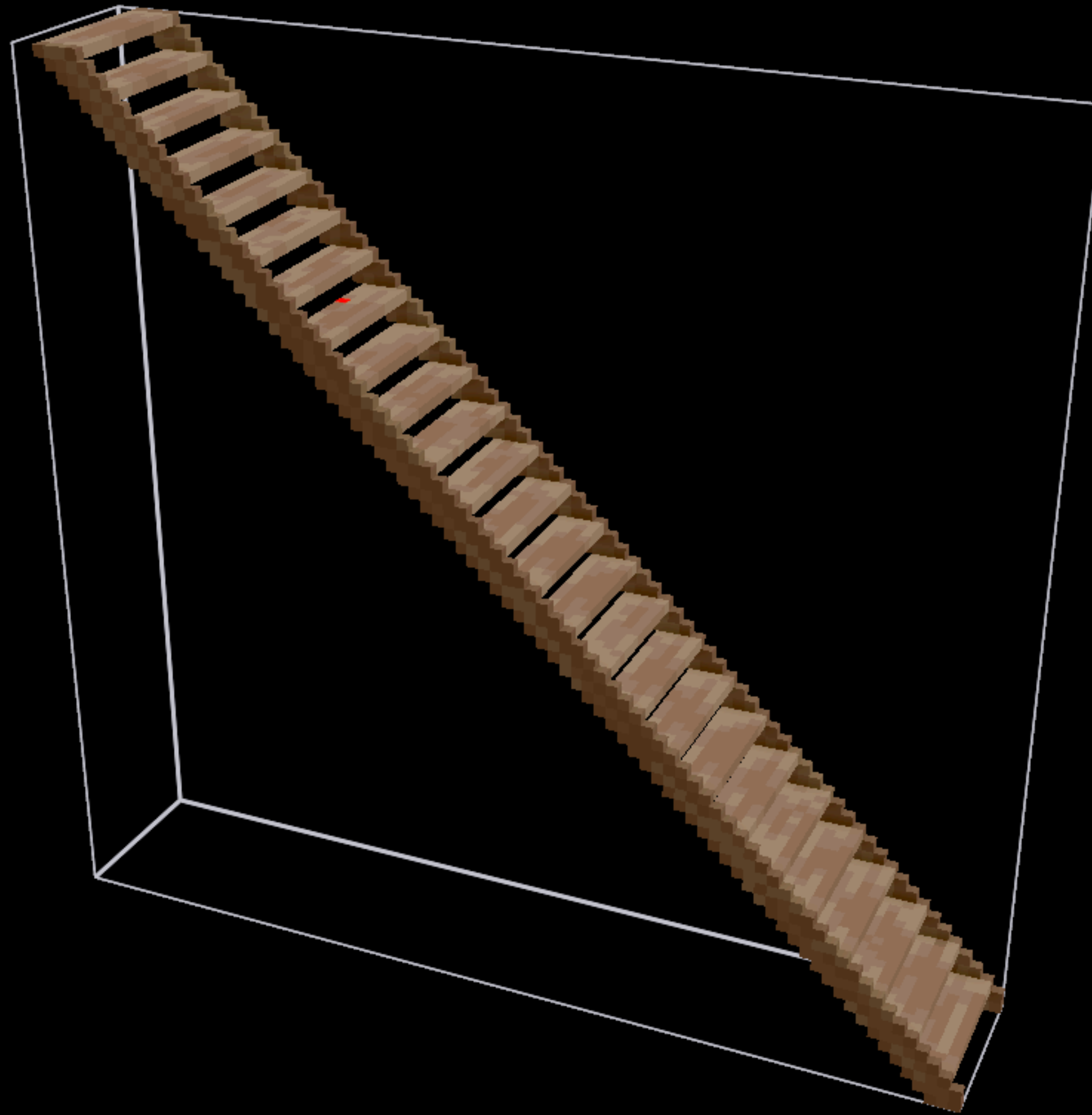
acko.net/blog/teardown-frame-teardown

juandiegomontoya.github.io/teardown_breakdown.html

Limitations

Limitations

Sparse shapes are very inefficient



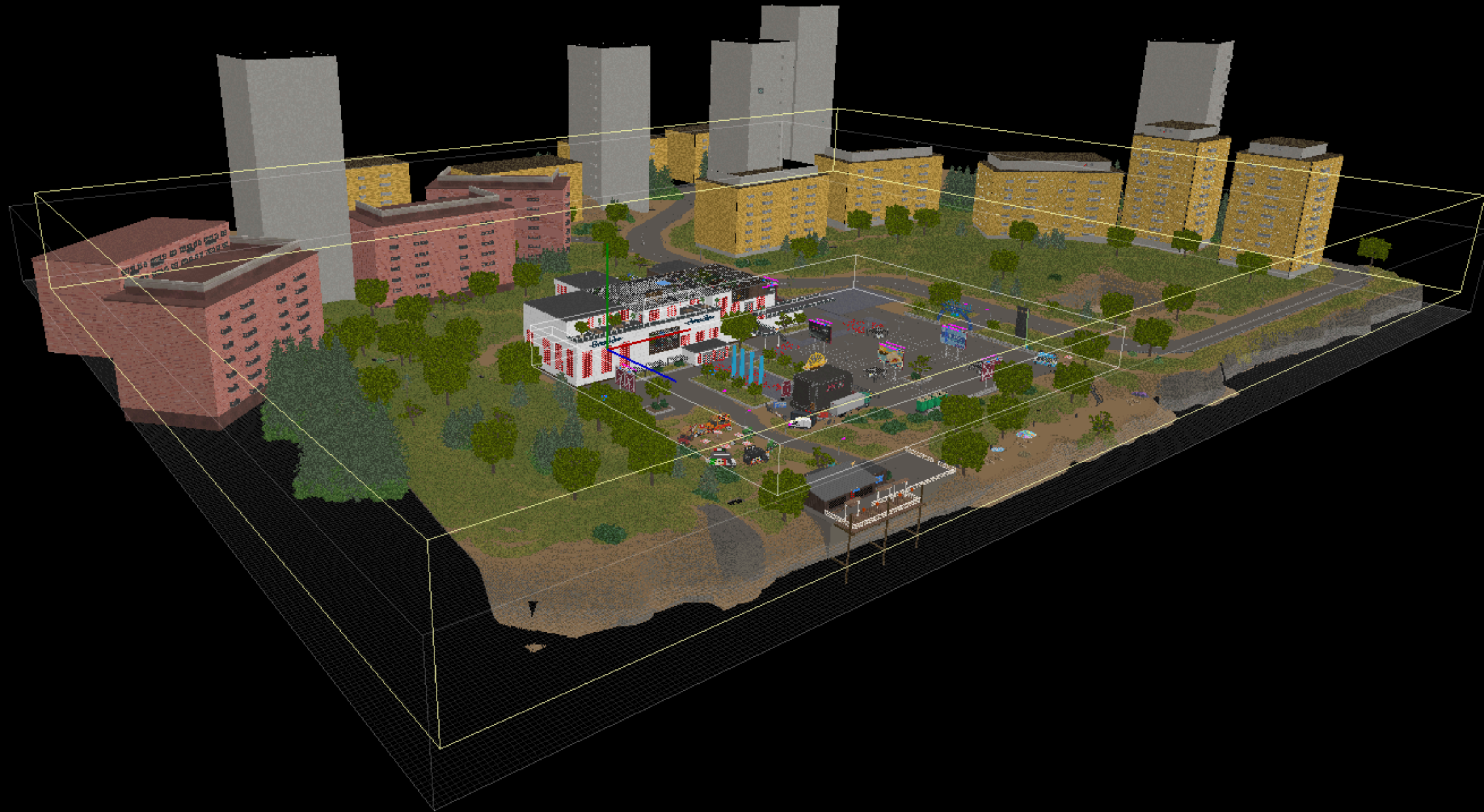
Limitations

Restricted by the palette



Limitations

Shadow volume implies a maximum level size



Limitations

No global illumination



Limitations

Dynamic objects require shadow volume updates



Limitations

Shadow volume quantization artifacts



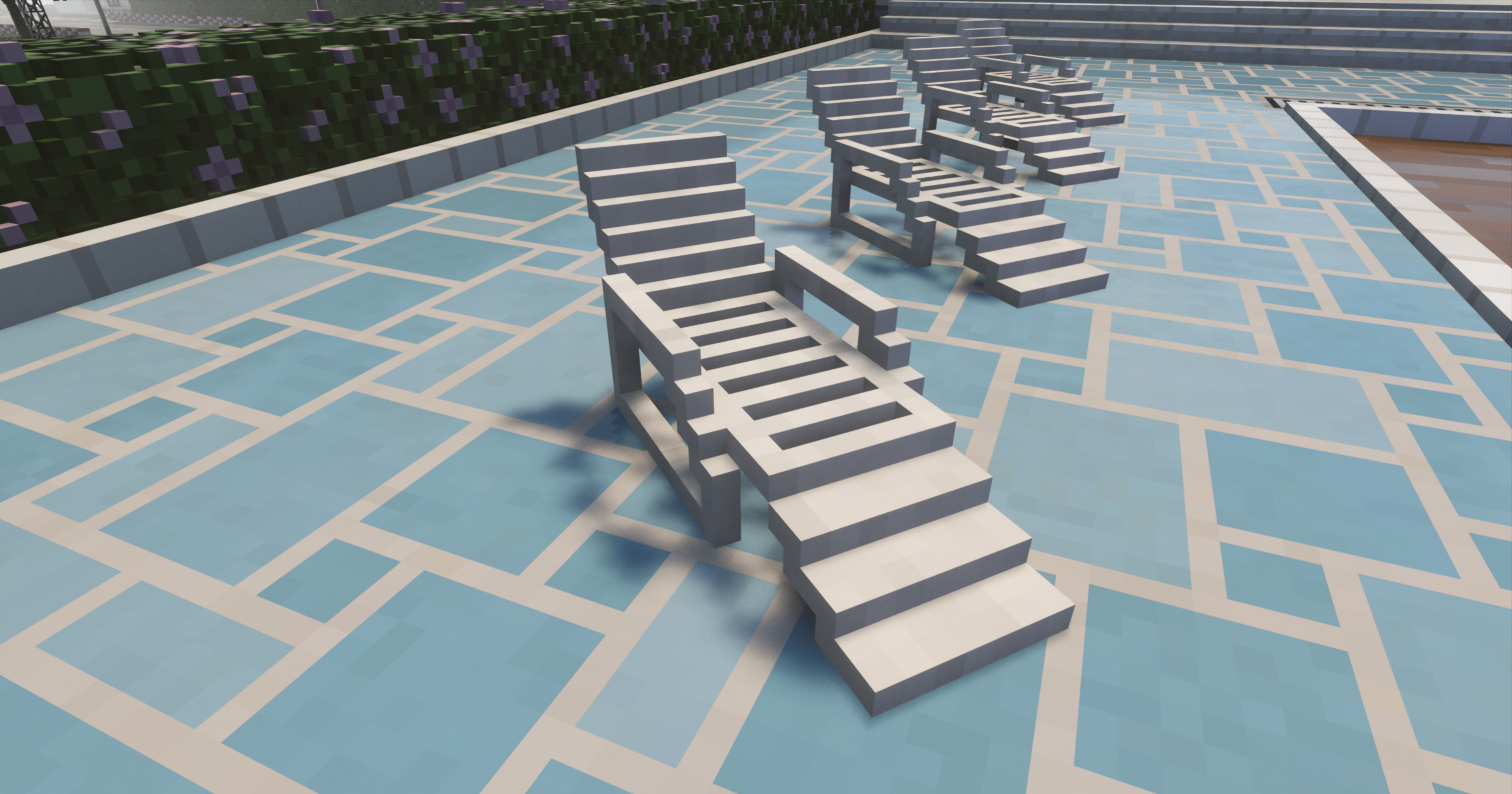
Limitations

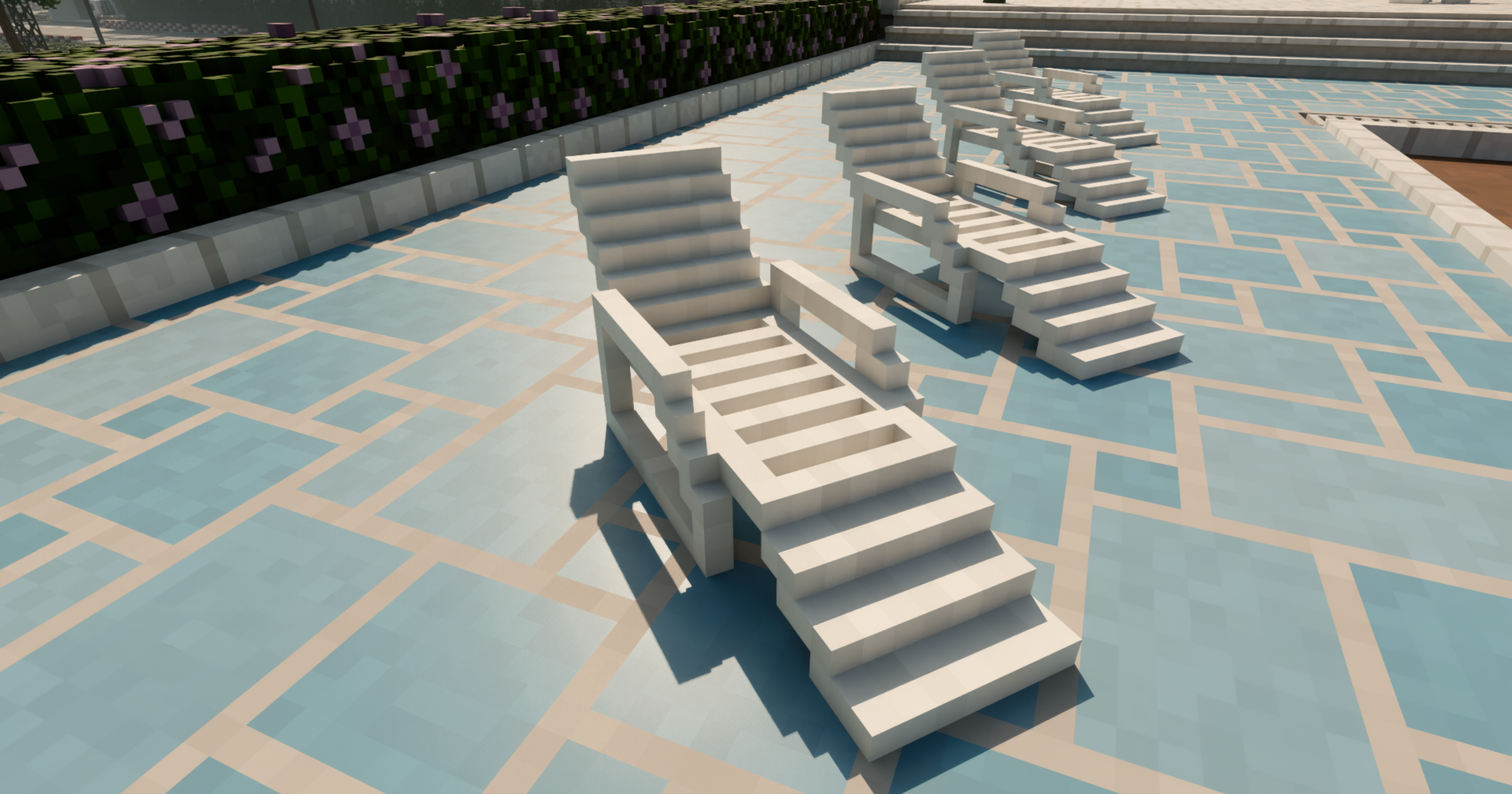
Shadow volume quantization artifacts



Limitations

- Sparse shapes
- Indexed palette
- Maximum level size
- Global illumination
- Dynamic objects
- Quantization artifacts







Daxa

- Created by Patrick Ahrens. Co-developed by Matěj Sakmary and me
- Faster iteration
- Powerful rendergraph
- Bindless resources
- Reasonable defaults
- Syntax niceties

The screenshot shows the GitHub repository for Daxa, a GPU abstraction built on Vulkan. The repository is owned by 'Daxa' and is public. It has 8 watchers, 41 forks, and 473 stars. The repository is currently on the 'master' branch, with 14 other branches and 13 tags. The commit history shows a merge of branch 'master' of https://github.com/lpotrick/Daxa, committed 4 days ago. The repository contains several files and folders, including .vscode, cmake, include/daxa, misc, src, tests, .clang-format, .clang-tidy, .gitattributes, .gitignore, CMakeLists.txt, CMakePresets.json, LICENSE, README.md, portfile.cmake, and vcpkg.json. The README.md file is open, showing a logo for Daxa. The right sidebar contains information about the repository, including the description 'Daxa is a convenient, simple and modern gpu abstraction built on vulkan', tags for 'gpu' and 'vulkan', a list of links (Readme, MIT license, Activity, 473 stars, 8 watching, 41 forks, Report repository), releases (3.2 Latest on Jul 16), packages (No packages published), contributors (21 contributors), and languages (C++ 86.6%, C 5.3%, GLSL 4.8%, CMake 2.0%, Slang 1.2%, HLSL 0.1%).

About

Daxa is a convenient, simple and modern gpu abstraction built on vulkan

Releases 9

3.2 Latest on Jul 16

+ 8 releases

Packages

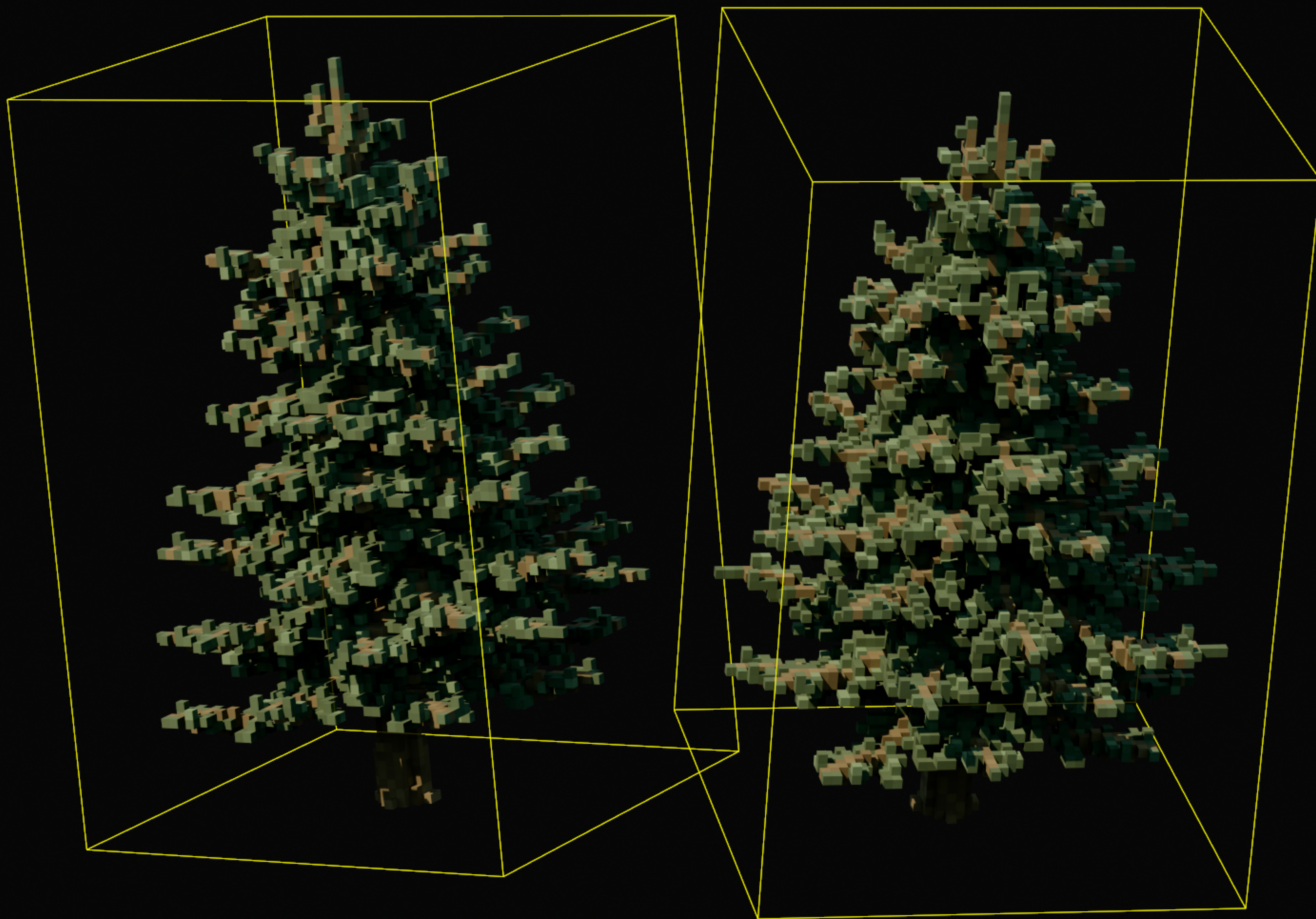
No packages published

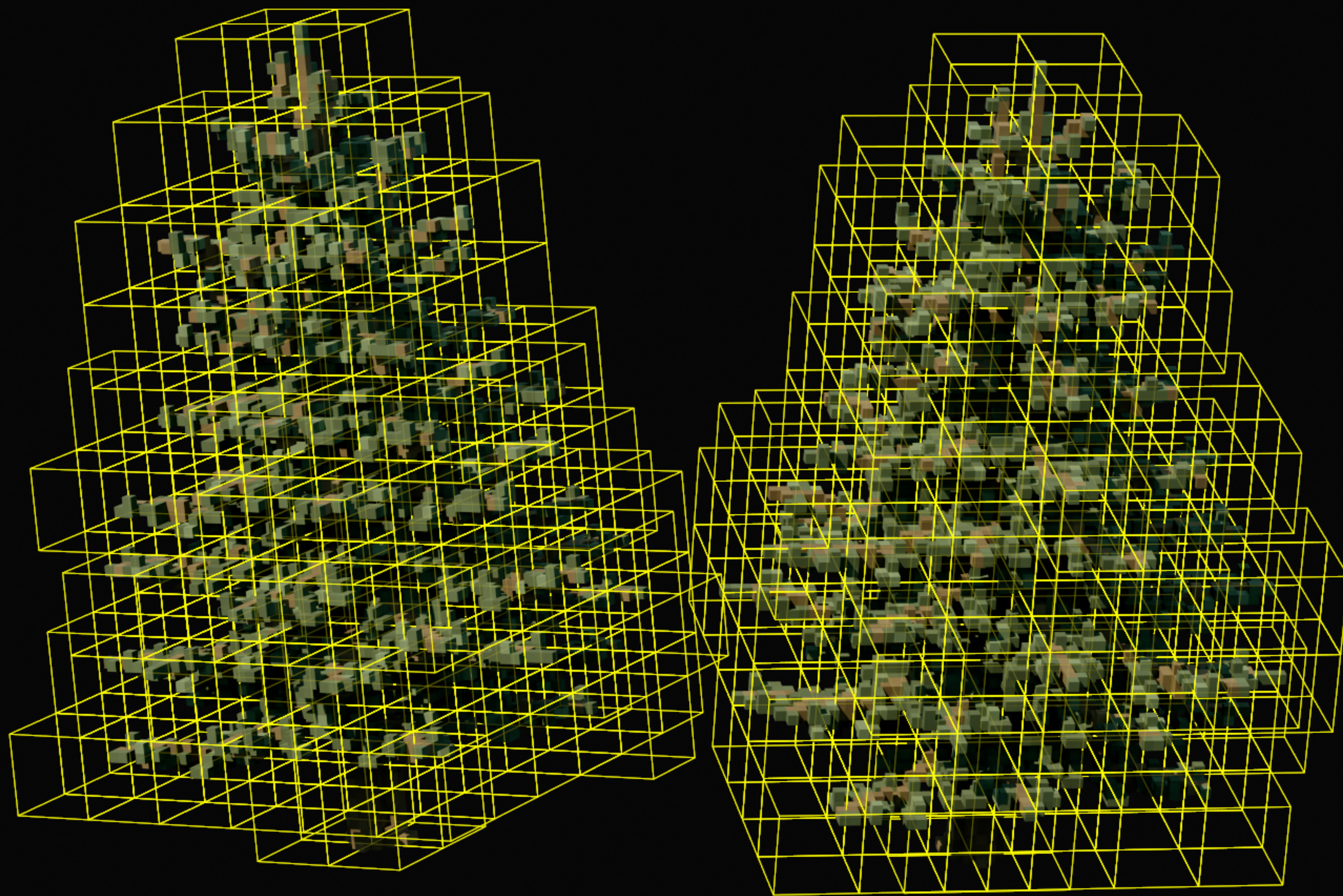
Contributors 21

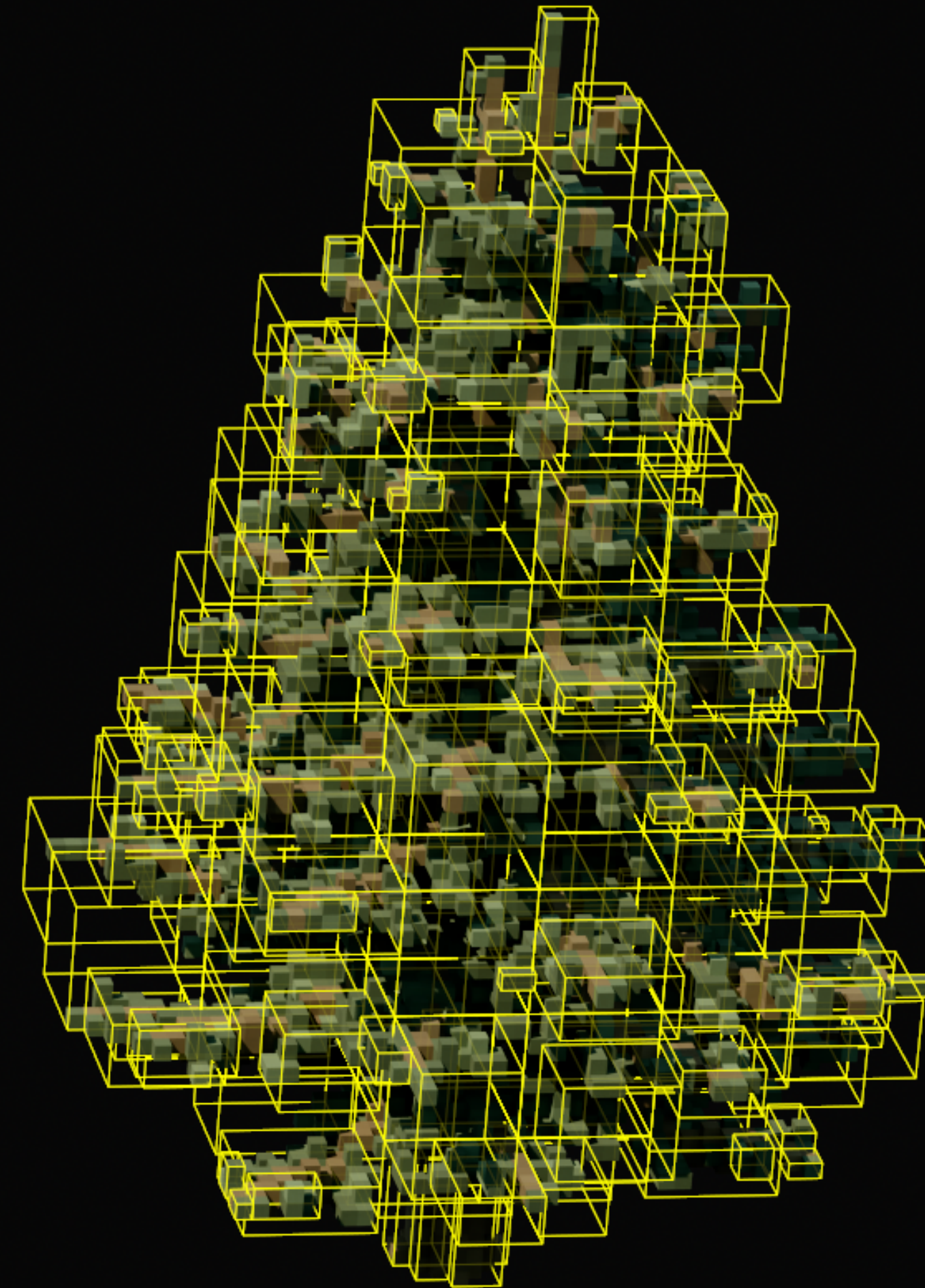
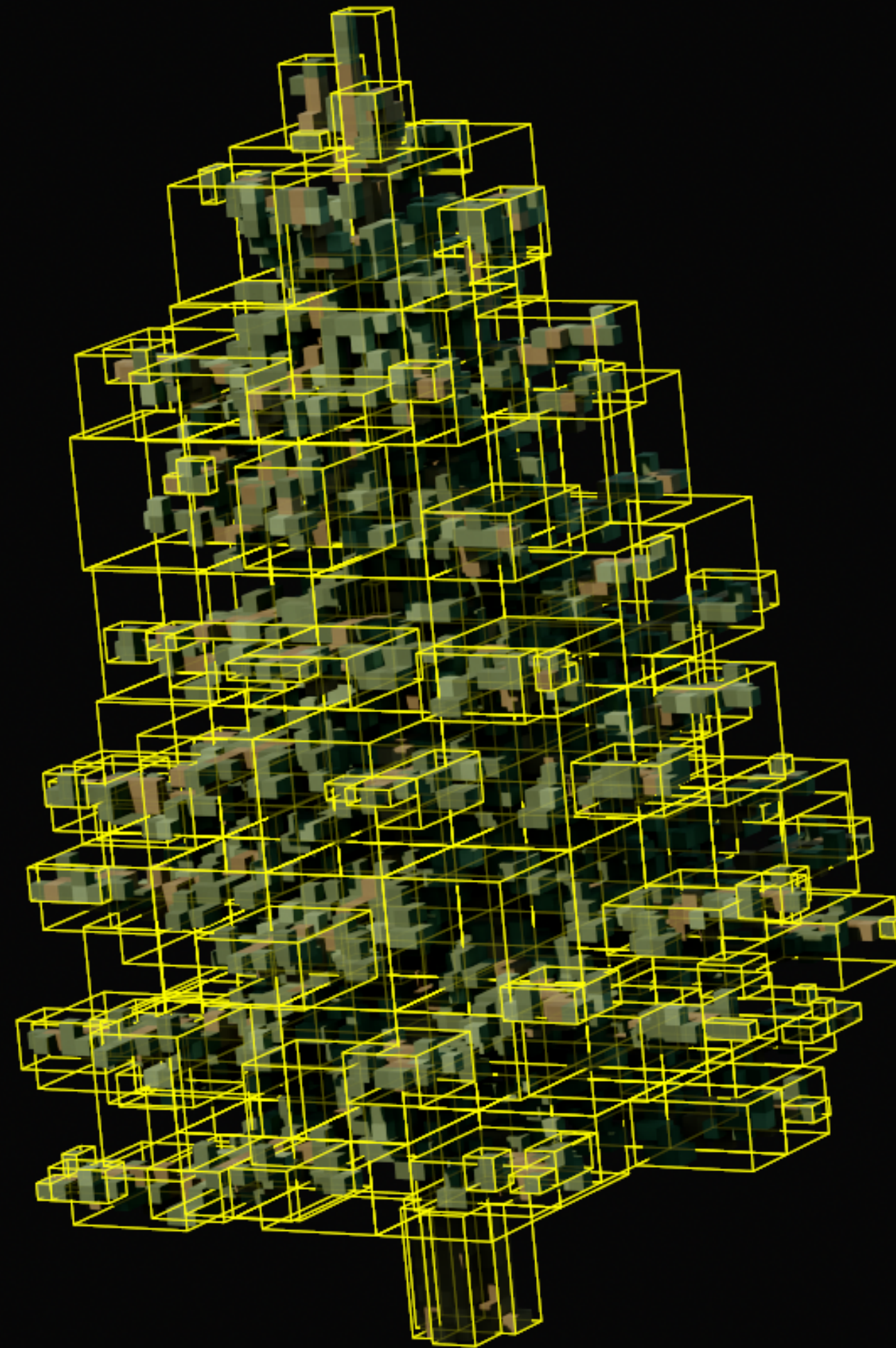
+ 7 contributors

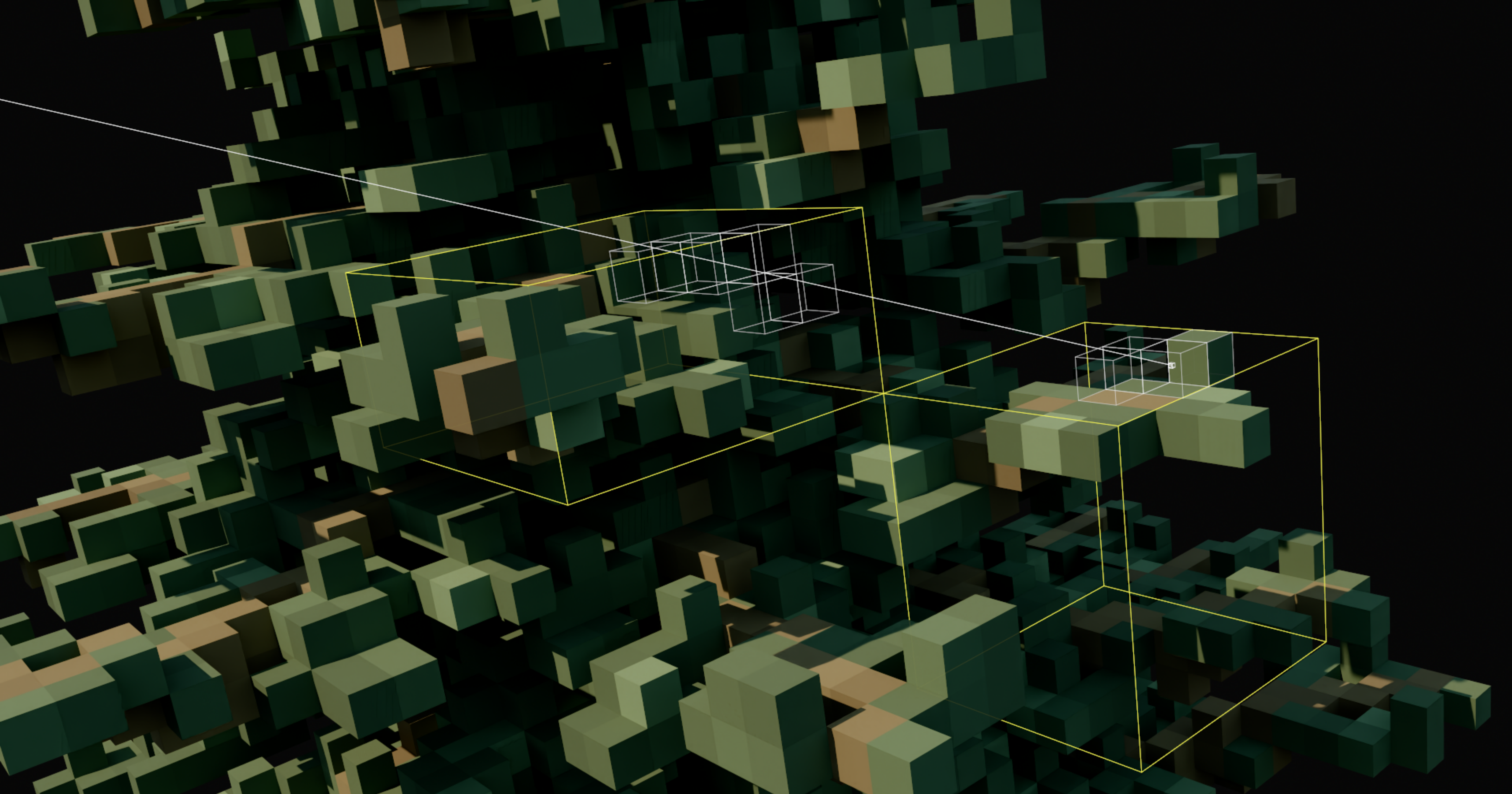
Languages

C++ 86.6% C 5.3% GLSL 4.8% CMake 2.0% Slang 1.2% HLSL 0.1%









Ray Tracing Inspector - Scene Tlas

Acceleration Structure

Search...

Name	# Prims.	Surface Ar	Size
instance[1695]	354	506.912	-
instance[1696]	48	127.971	-
instance[1697]	339	385.923	-
VoxelObject Bias (2515)	339	385.923	14.7 KB
aabb[0]	339	385.923	-
instance[1698]	46	101.119	-
instance[1699]	339	262.374	-
instance[1700]	40	149.603	-
instance[1701]	339	296.749	-
instance[1702]	43	274.785	-
instance[1703]	335	534.752	-
instance[1704]	49	208.79	-
instance[1705]	354	454.159	-
instance[1706]	49	315.416	-
instance[1707]	351	408.957	-
instance[1708]	42	241.092	-
instance[1709]	348	399.365	-
instance[1710]	393	893.828	-
instance[1711]	87	158.367	-
instance[1712]	68	203.845	-
instance[1713]	81	125.706	-
instance[1714]	77	135.724	-
instance[1715]	85	189.328	-
instance[1716]	89	214.81	-
instance[1717]	123	258.568	-
instance[1718]	84	268.682	-
instance[1719]	93	336.419	-
instance[1720]	48	180.083	-
instance[1721]	354	395.99	-
instance[1722]	48	129.81	-
instance[1723]	339	385.923	-
instance[1724]	40	182.122	-
instance[1725]	339	385.923	-
instance[1726]	44	277.56	-
instance[1727]	335	518.013	-
instance[1728]	52	187.85	-
instance[1729]	354	395.99	-

AABB: aabb[0]

FlagsOPAQUE

AABB Count339

BufferAABB Buffer

Stride24

Visualization

Viewport Size: 1577 x 1333

Ray Count: 2102141

Trace Rays: 2.361 ms

Camera Controls

Mode: Fly Camera

Position (X): 89.203

Position (Y): 58.447

Position (Z): 90.175

Direction (X): -0.663

Direction (Y): -0.233

Direction (Z): -0.712

Near Plane: 0.100

Far Plane: 24400.641

FOV: 45.000

Speed: 3.895

Up Direction: Y Axis

Coordinates: RHS

Reset Camera

Highlight

OpaqueNon-Opaque

Cull DisabledFront CCW

Fast BuildFast Trace

Low MemoryAllow Update

Allow CompactionCompacted

No Duplicate Any Hit InvocationOpacity Micromap

Heatmap Overlay Options

Heatmap Overlay Mode: Traversal Timing

Threshold (clocks): 100000

Performance HUD: ✓

Opacity Override: None

Opacity Micromap Override: None

Reset Heatmap Overlay

Rendering Options

Color By: Instance

Anti-aliasing Mode: MSAA (4x)

Transparency Effect: Enable

Opacity Micromap Effect: Enable

Disable Back Face CullShow Mesh Wireframes

Show GeometryShow AABB Wireframes

Show Axes

Reset Rendering

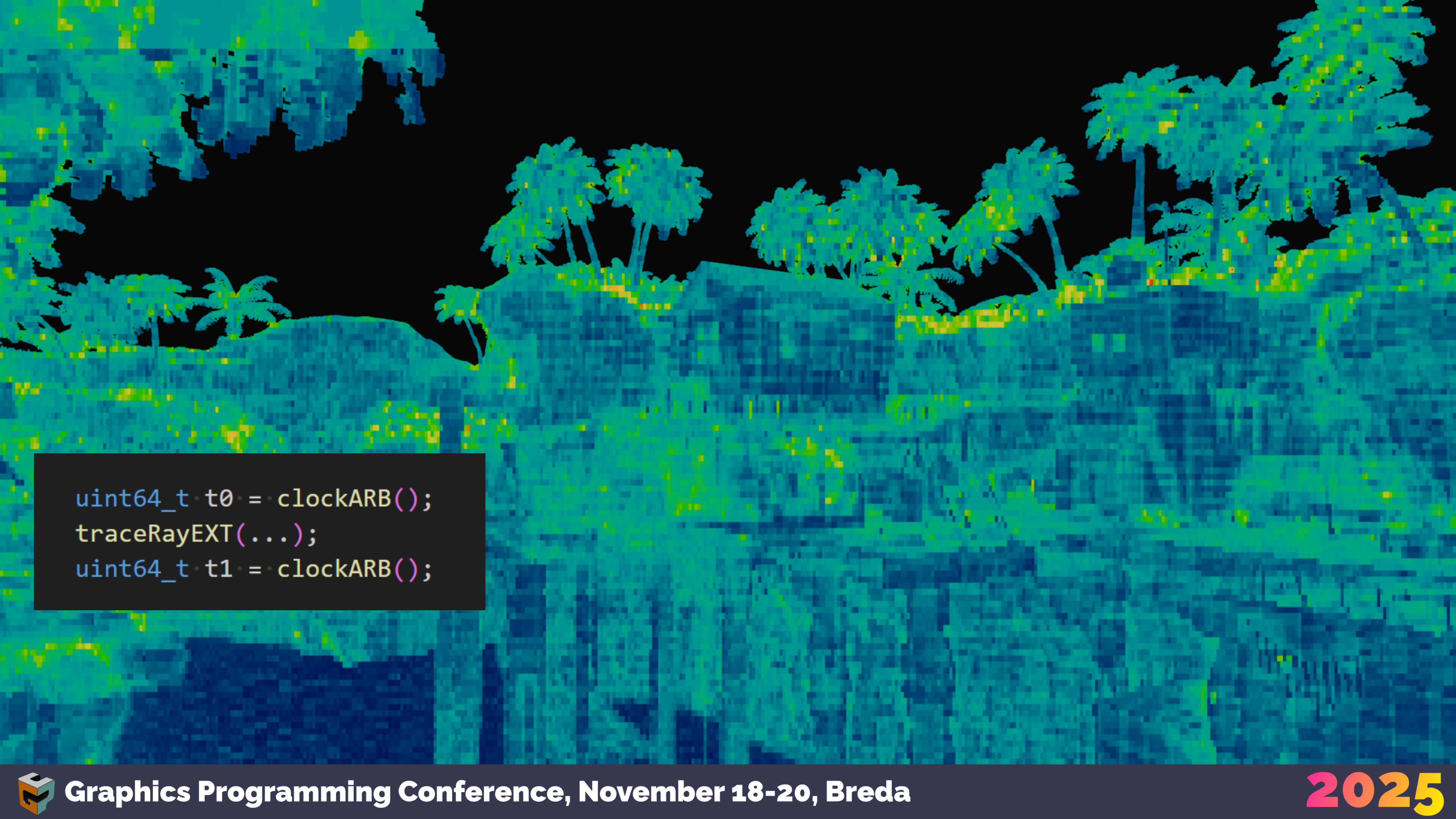
Bookmarks

Performance Analysis

Instance AABB Overlap Heatmap (world space)...

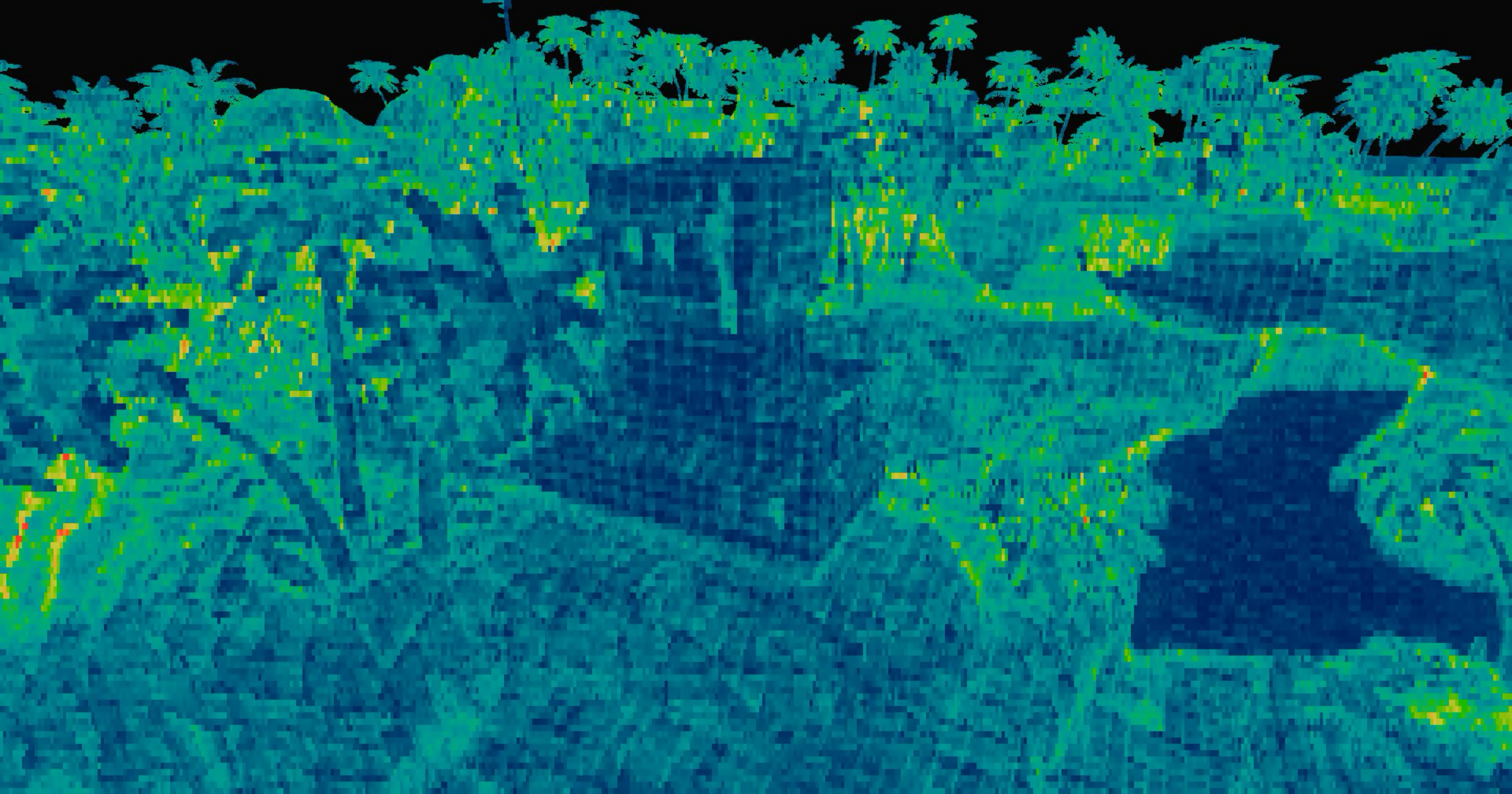
Graphics Programming Conference, November 18-20, Breda

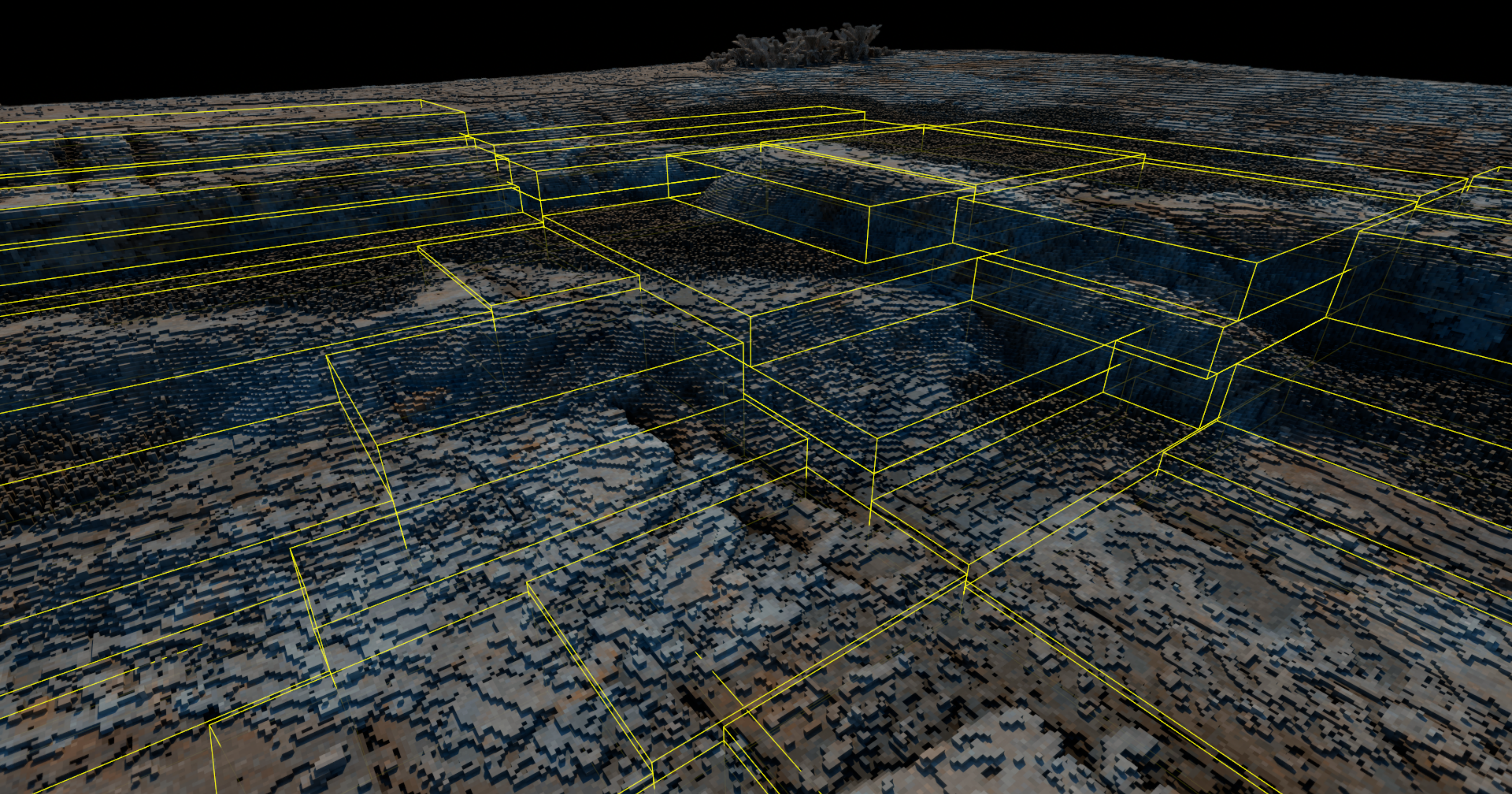
2025

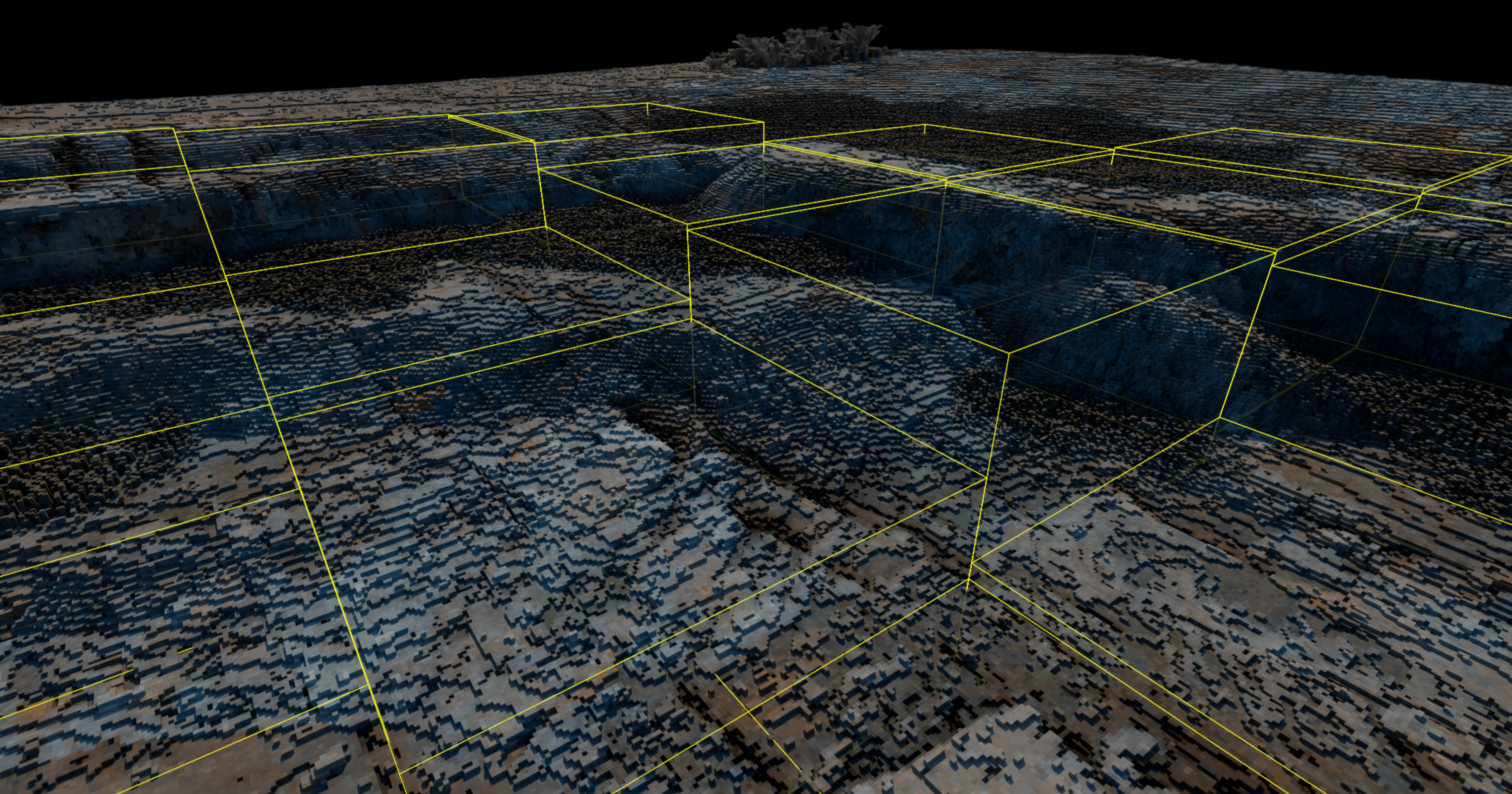


```
uint64_t t0 = clockARB();  
traceRayEXT(...);  
uint64_t t1 = clockARB();
```

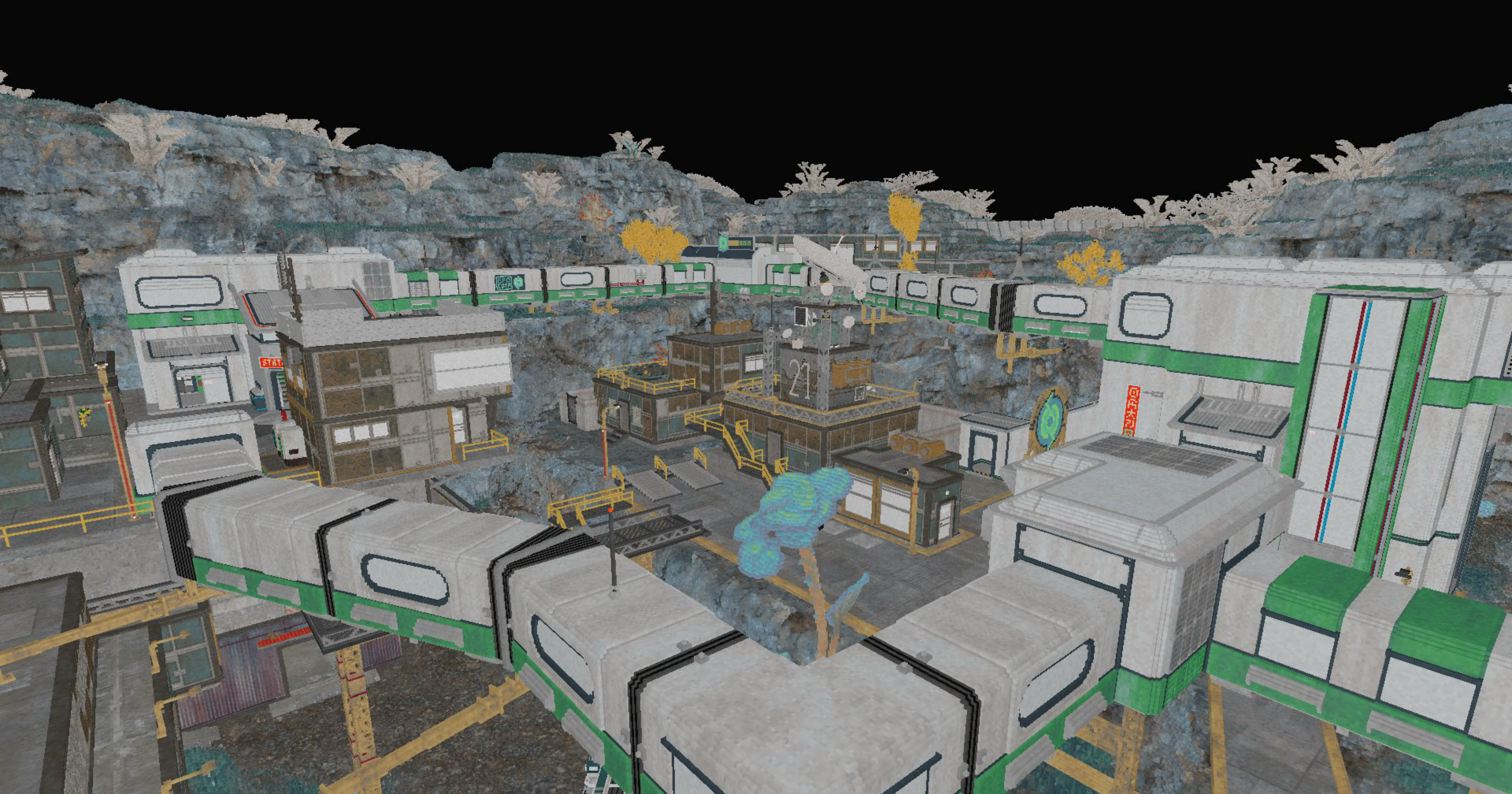








What about primary view?



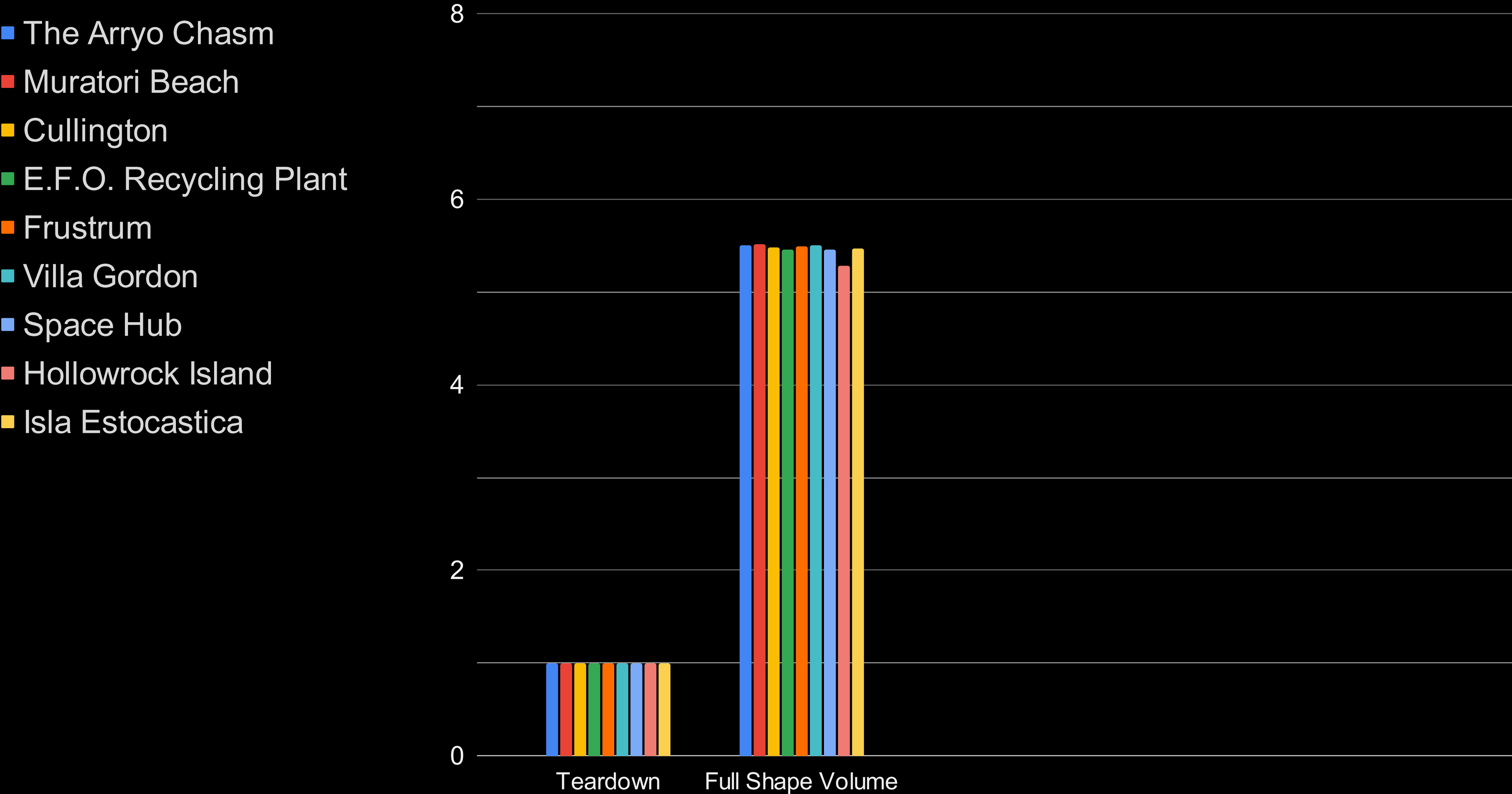
Limitations

- ~~• Sparse shapes~~
- Indexed palette
- ~~• Maximum level size~~
- ~~• Global illumination~~
- ~~• Dynamic objects~~
- ~~• Quantization artifacts~~



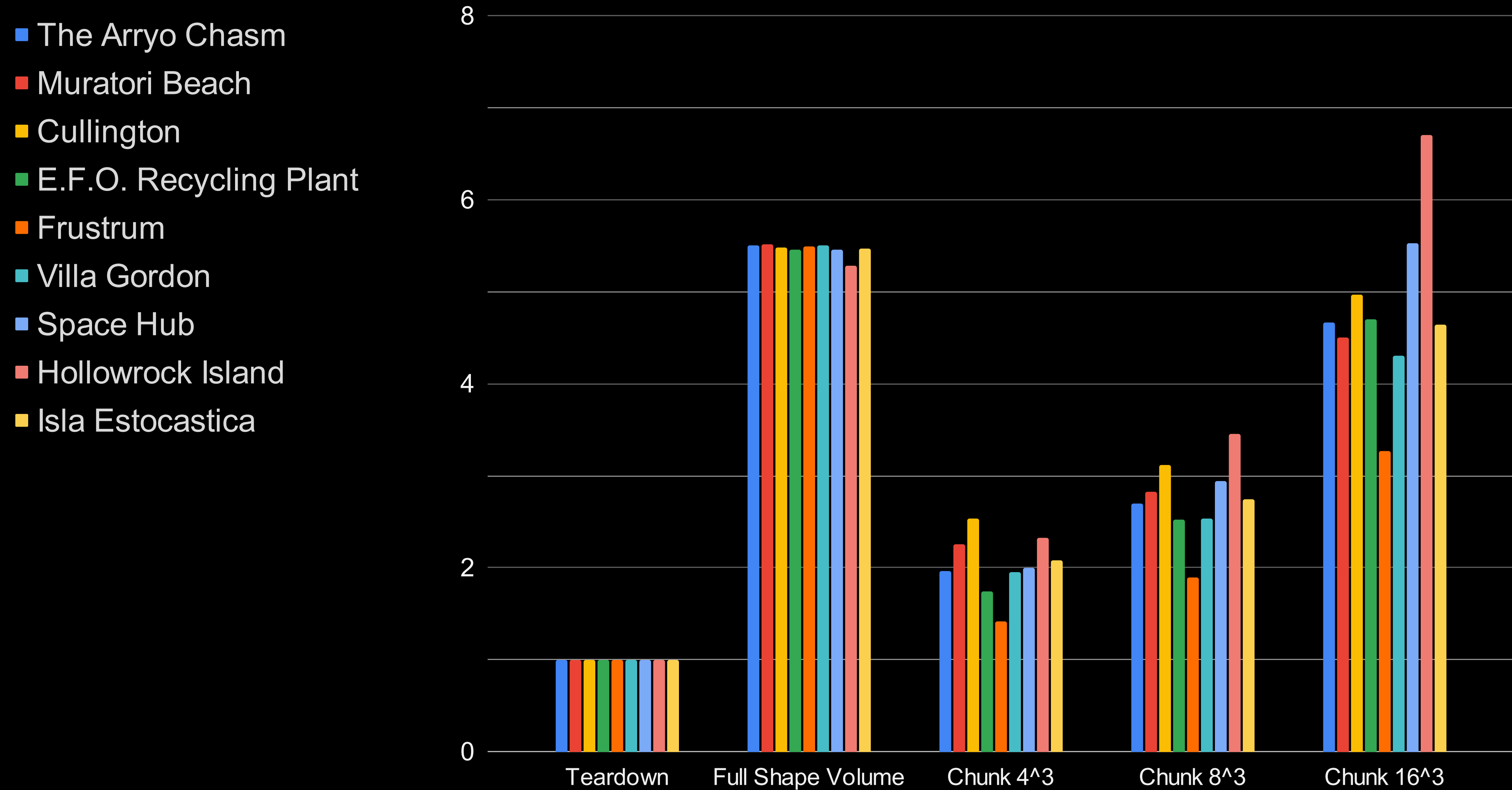
Level data VRAM, normalized by Level data VRAM in Teardown

Teardown includes the bitmap, palette indices, and palette data. New engine uses 6.125 bytes per voxel and includes ASes and AABBs



Level data VRAM, normalized by Level data VRAM in Teardown

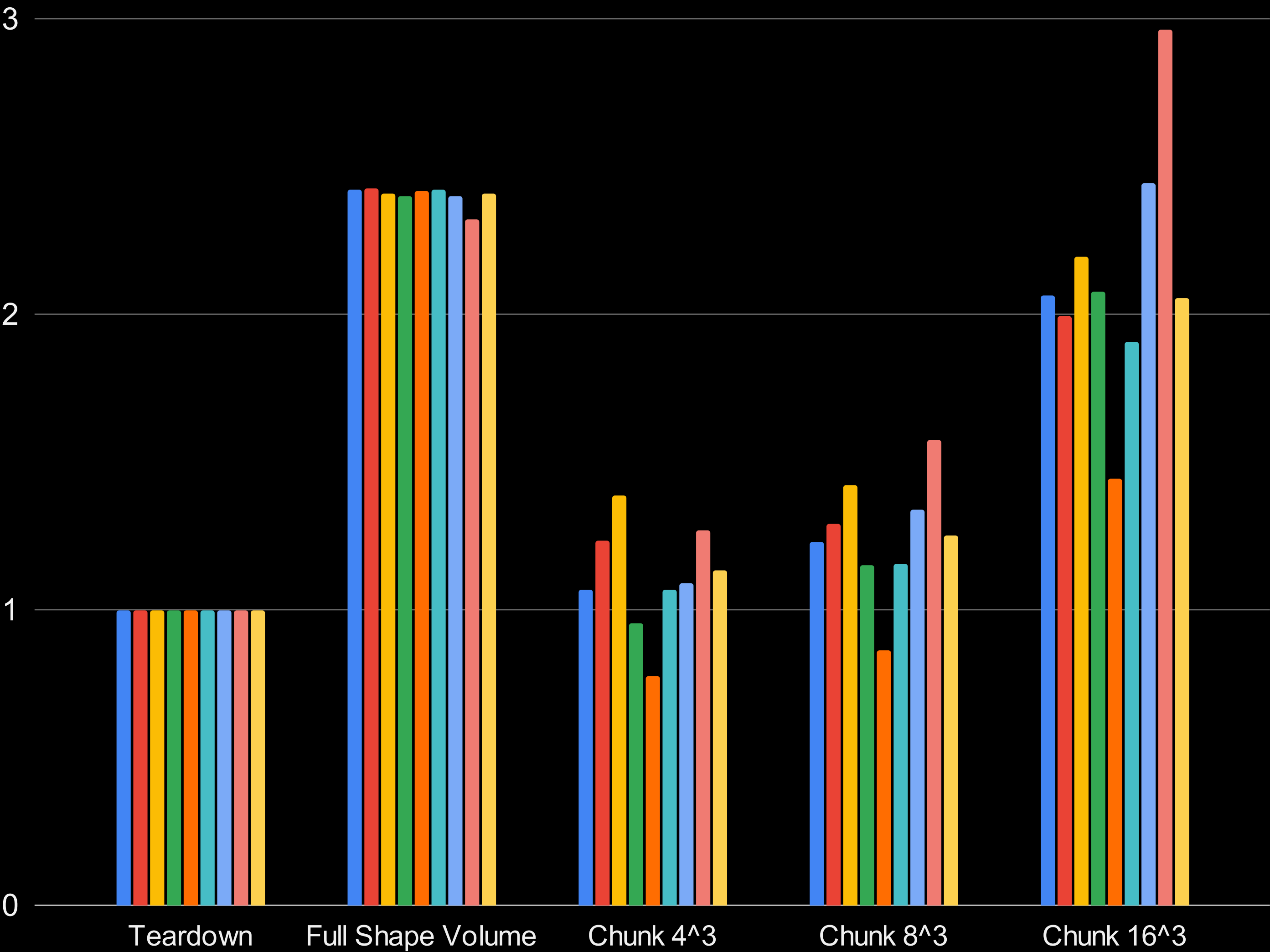
Teardown includes the bitmap, palette indices, and palette data. New engine uses 6.125 bytes per voxel and includes ASes and AABBs

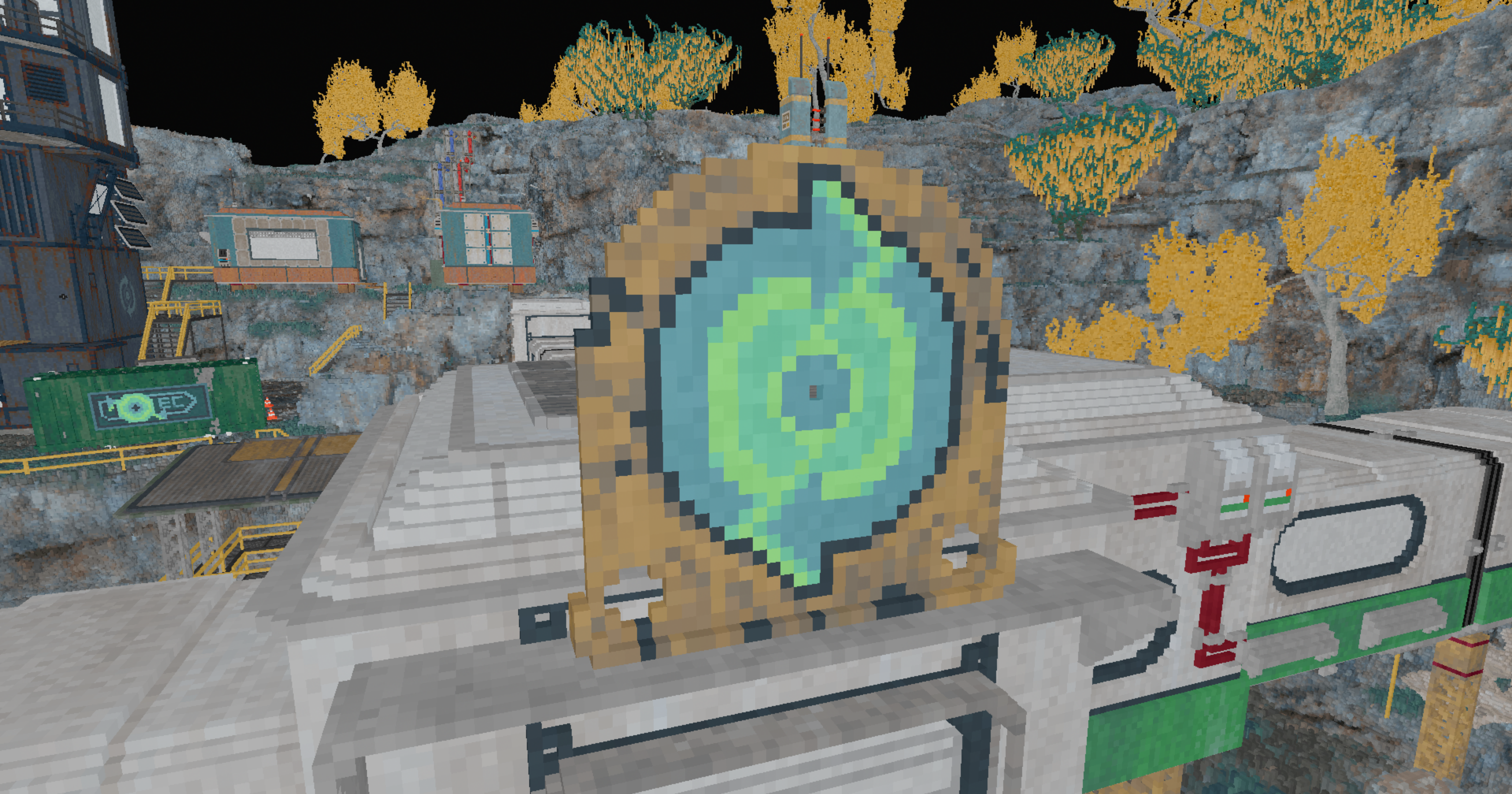


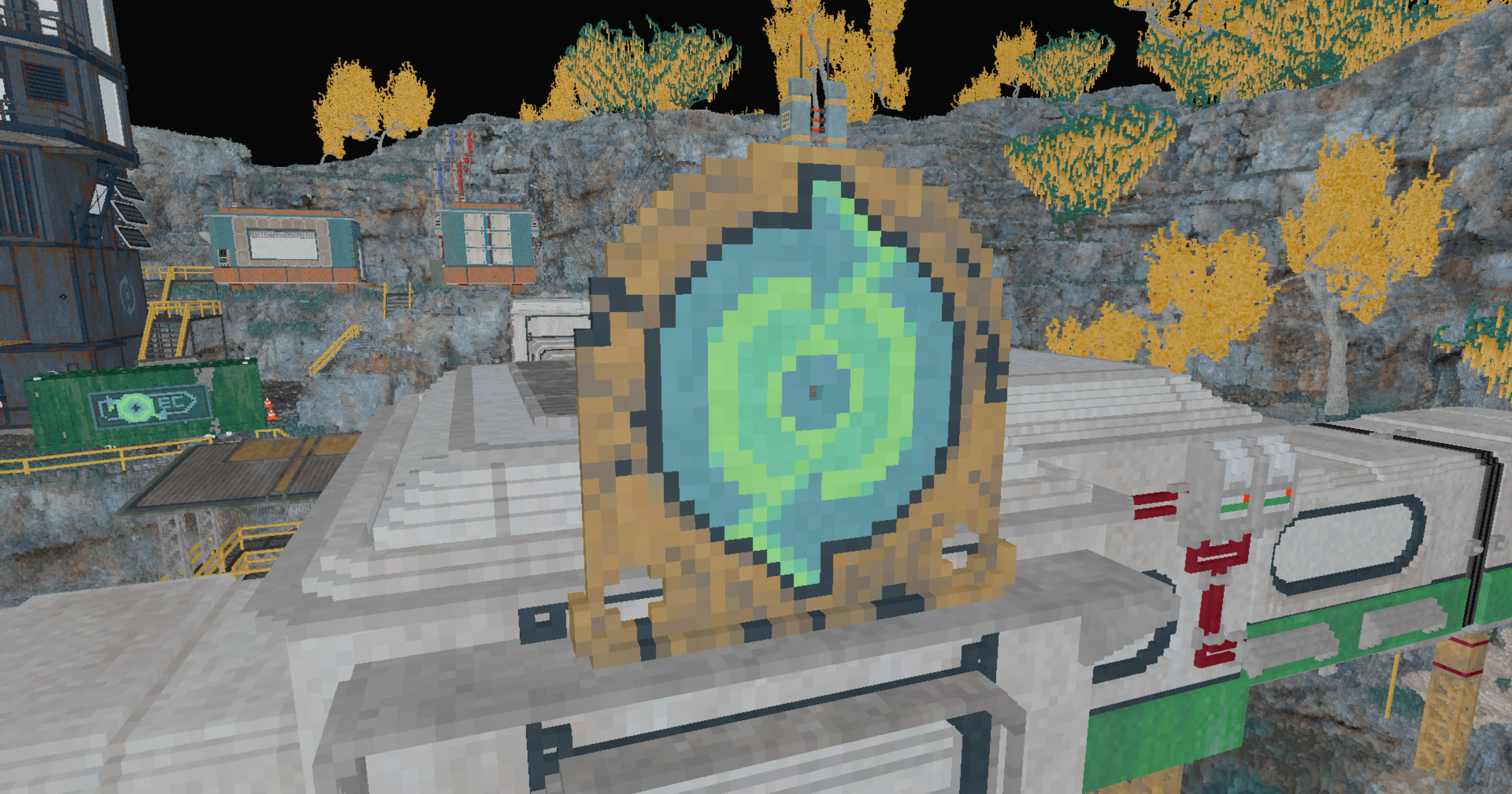
Level data VRAM, normalized by Level data VRAM in Teardown

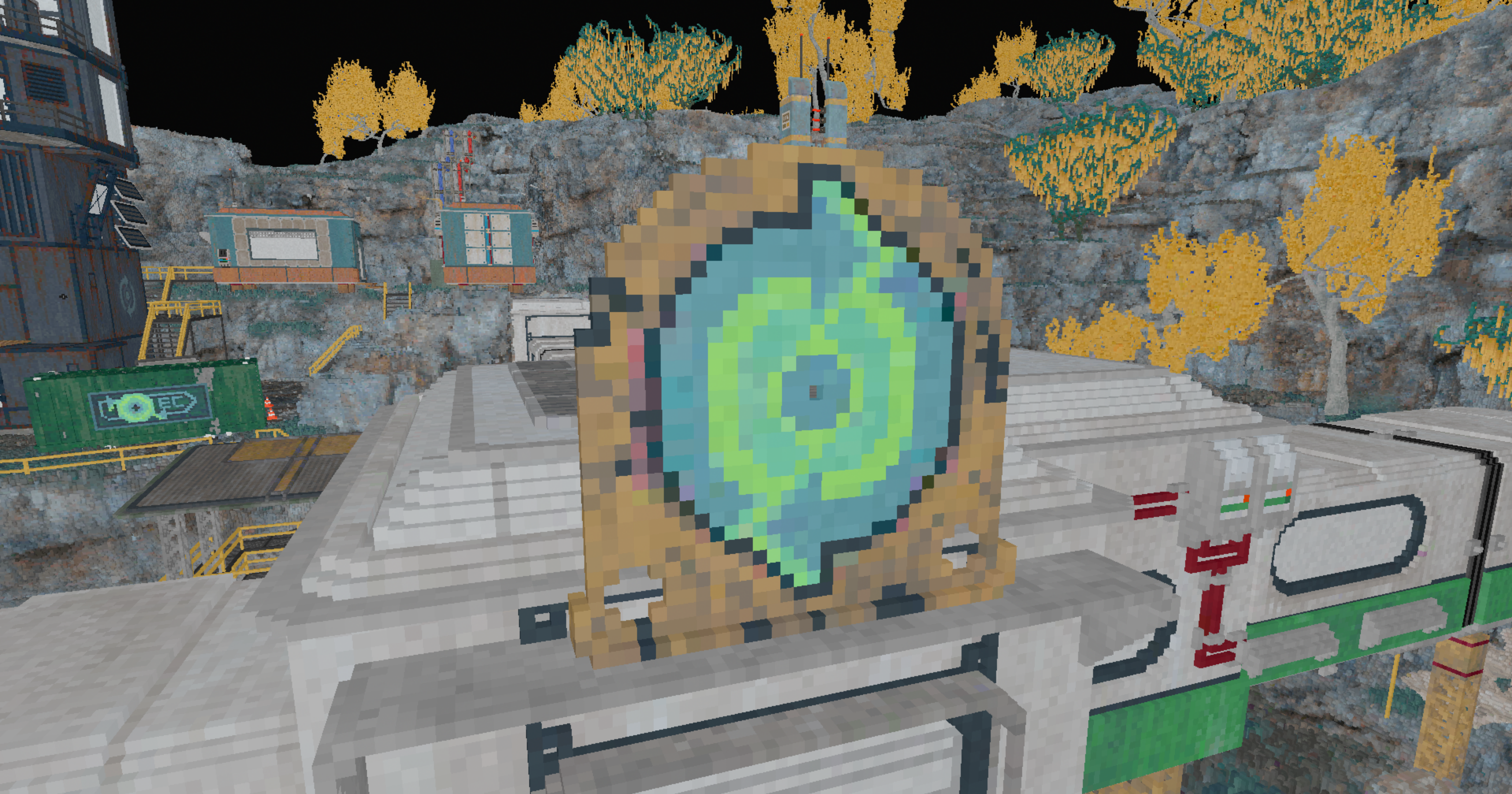
Teardown includes the bitmap, palette indices, and palette data. New engine uses 2.625 bytes per voxel and includes ASes and AABBs

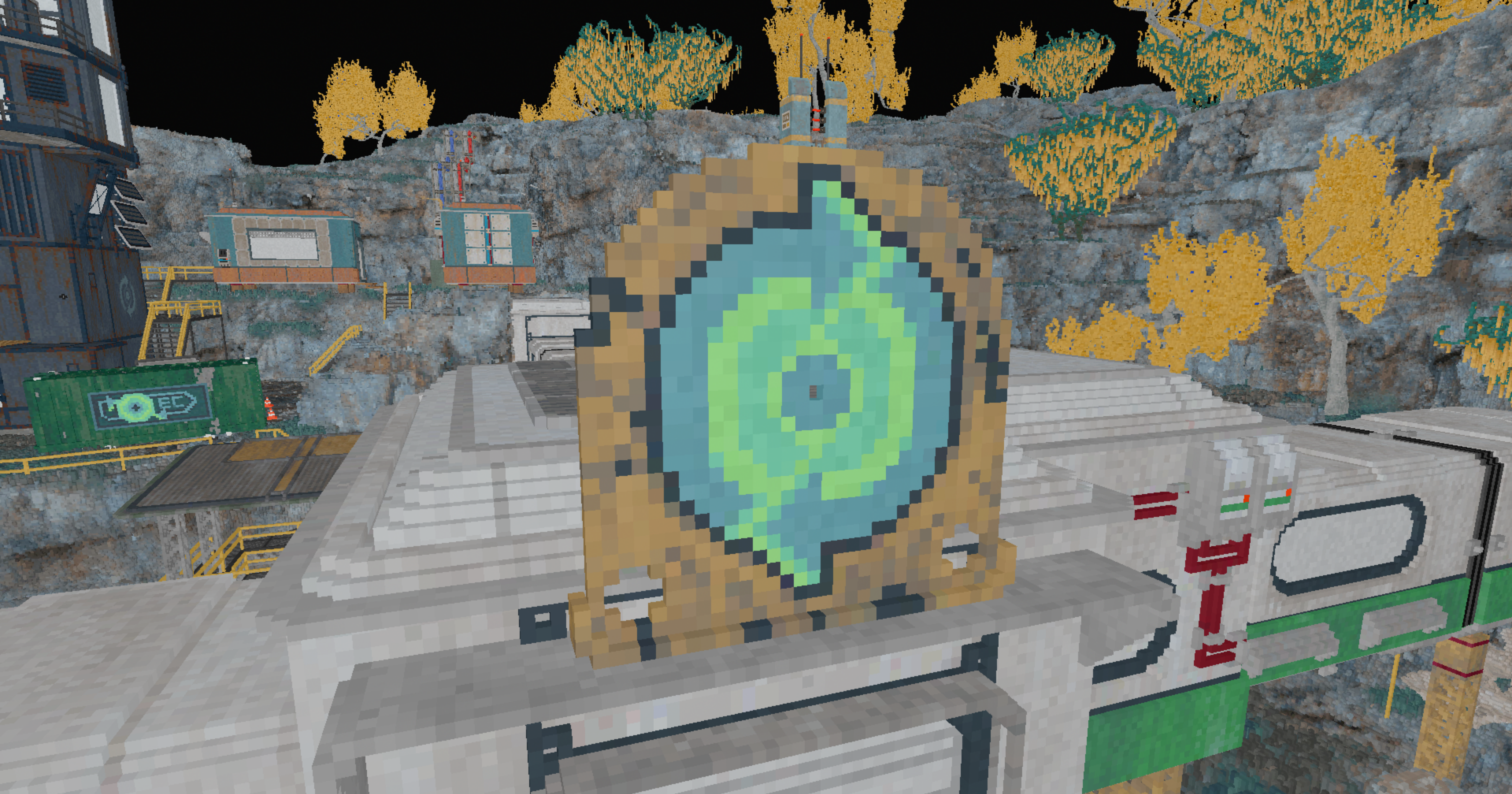
- The Arryo Chasm
- Muratori Beach
- Cullington
- E.F.O. Recycling Plant
- Frustrum
- Villa Gordon
- Space Hub
- Hollowrock Island
- Isla Estocastica







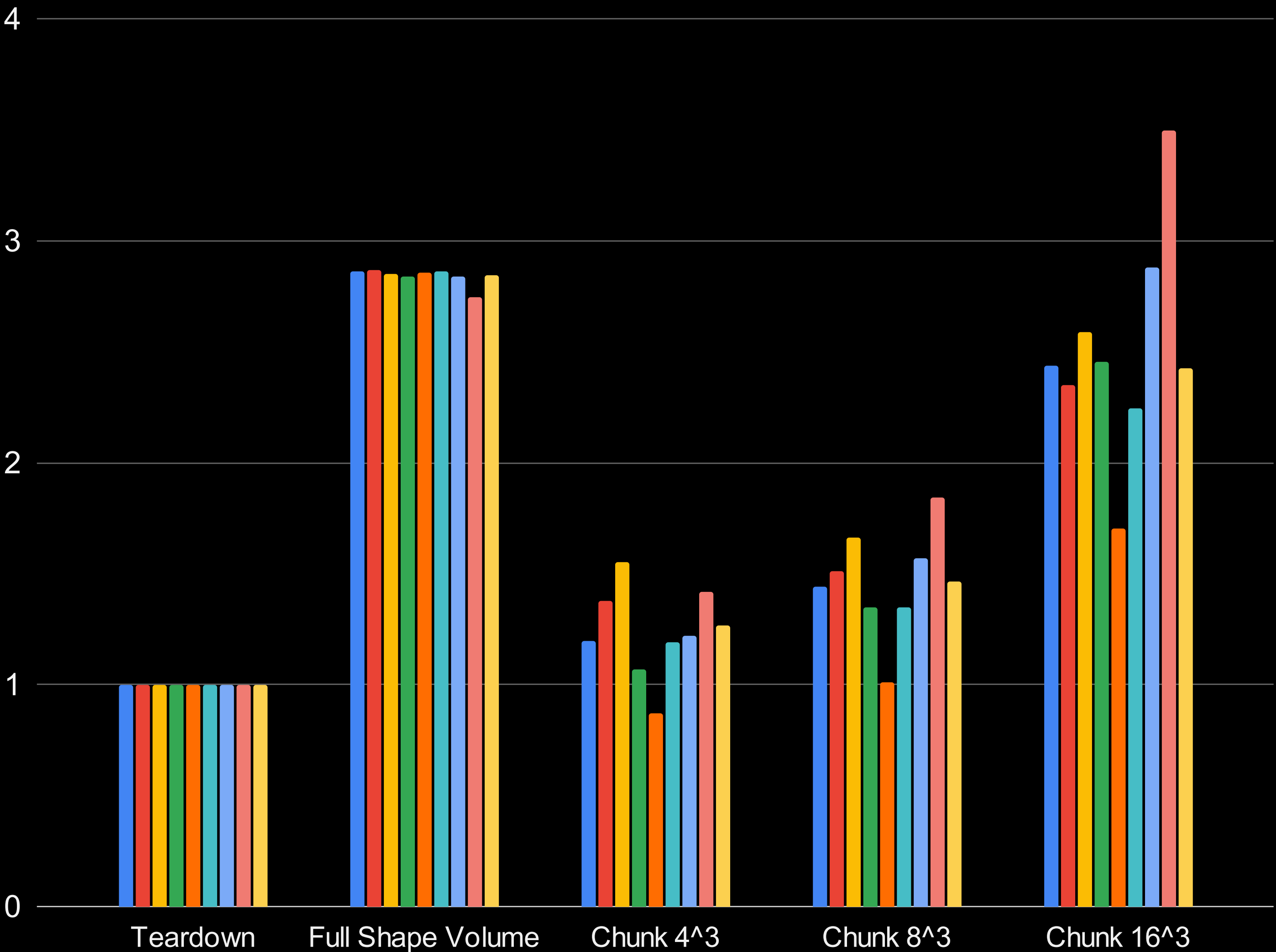




Level data VRAM, normalized by Level data VRAM in Teardown

Teardown includes the bitmap, palette indices, and palette data. New engine uses 3.125 bytes per voxel and includes ASes and AABBs

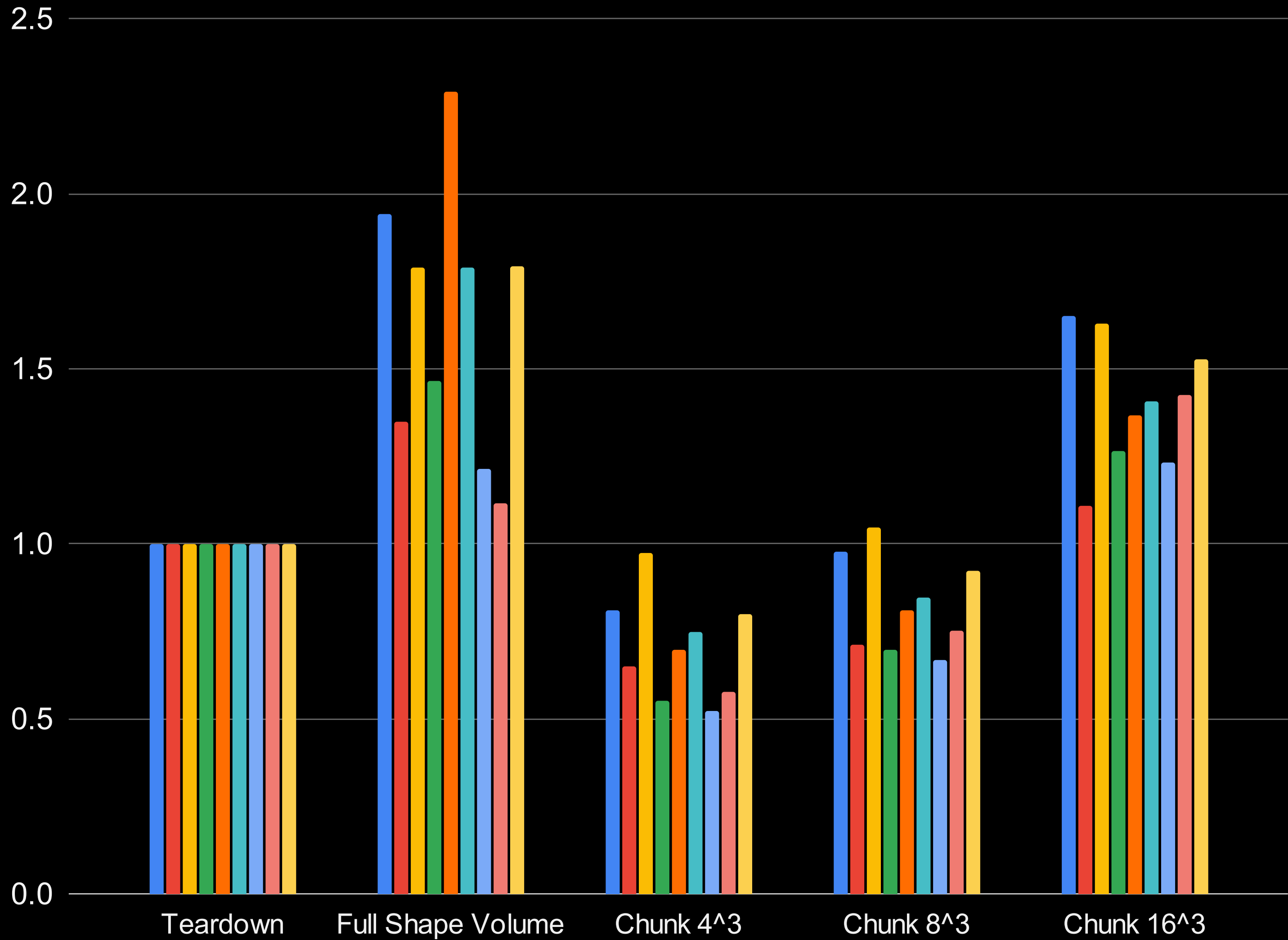
- The Arryo Chasm
- Muratori Beach
- Cullington
- E.F.O. Recycling Plant
- Frustrum
- Villa Gordon
- Space Hub
- Hollowrock Island
- Isla Estocastica



Level data VRAM, normalized by Level data VRAM in Teardown

Teardown includes the bitmap, palette indices, palette data, and Shadow Volume. New engine uses 3.125 bytes per voxel and includes ASes and AABBs

- The Arryo Chasm
- Muratori Beach
- Cullington
- E.F.O. Recycling Plant
- Frustrum
- Villa Gordon
- Space Hub
- Hollowrock Island
- Isla Estocastica



Limitations

- ~~• Sparse shapes~~
- ~~• Indexed palette~~
- ~~• Maximum level size~~
- ~~• Global illumination~~
- ~~• Dynamic objects~~
- ~~• Quantization artifacts~~





