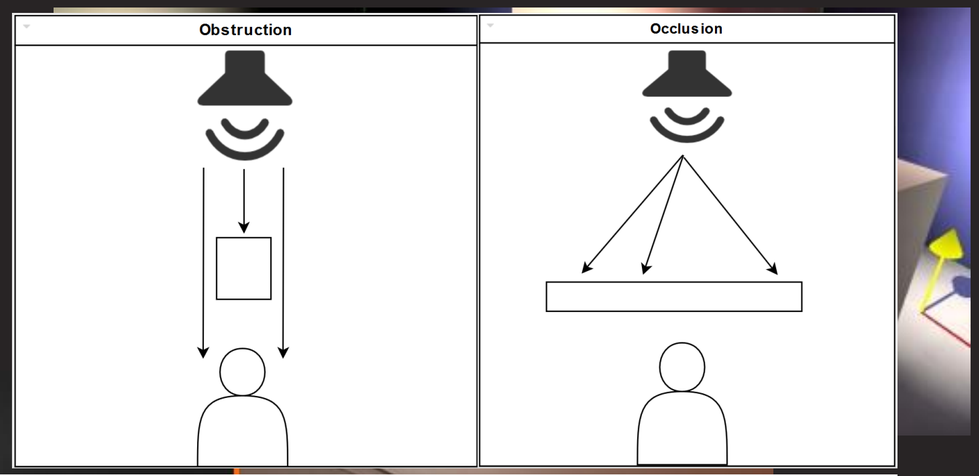
Audio Propagation Through the Ears of VERA

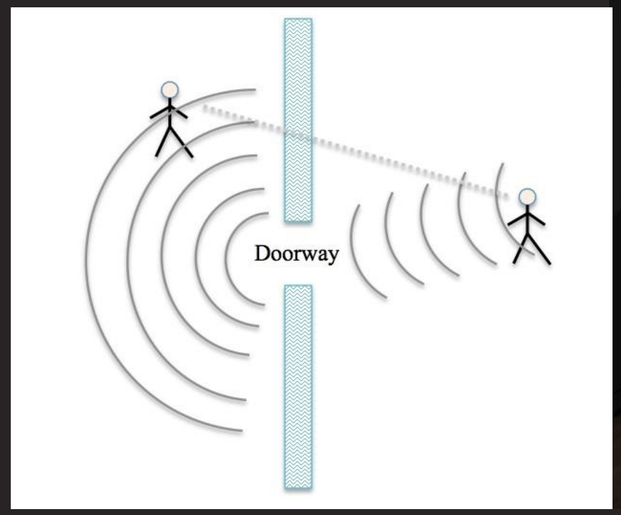
Jeff Ballard
Software Engineer – Microsoft ATG

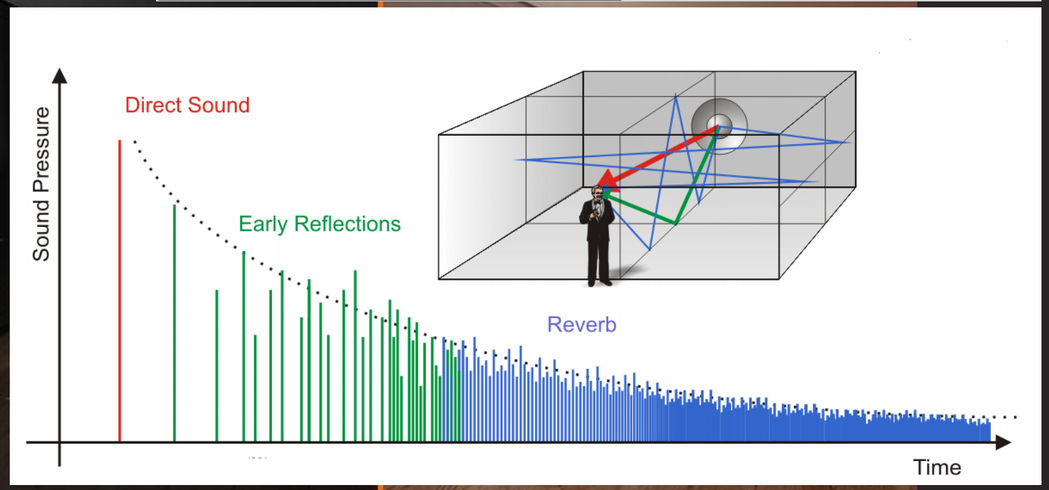
Robert Ridihalgh
Senior Technical Audio Specialist – Microsoft ATG

Lighting Acoustic Properties

- Occlusion and Obstruction
- Early Reflections
- Portaling
- Late Reverb





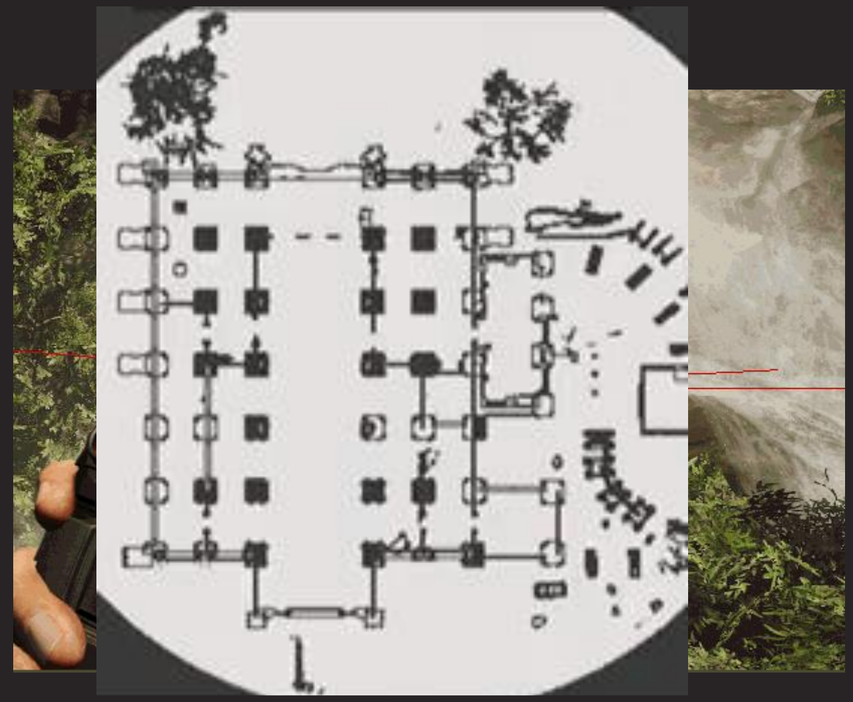


Acoustic Simulation Challenges

- Static world
  - Manual acoustic zones or trigger boxes
- Procedurally generated world
  - Complicated and imperfect
  - A lot of manual work still required
- Dynamic and destructible world
  - Lots of code to modify hand created zones

Existing Solutions

- Brute force raycasting
  - Per sound path finding
  - Can be expensive and lack information
- Wave simulation
  - Usually too expensive for real time
- Game specific implementations



Setting Goals

- Provide a variety of valuable data for audio propagation in real time
- Drastically reduce manual iteration and markup
- Heavily scalable
- Portable and universal

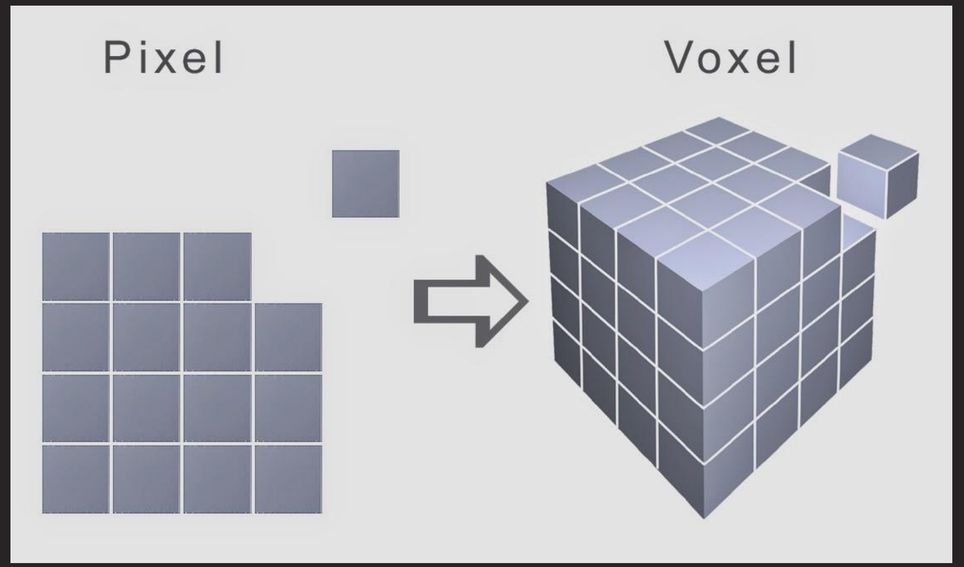
Voxel Engine for Real-time Acoustics

VERA can automatically generate:

- Obstruction and occlusion data
- Environmental data including reflection surfaces
- Locations for audio portaling

VERA features:

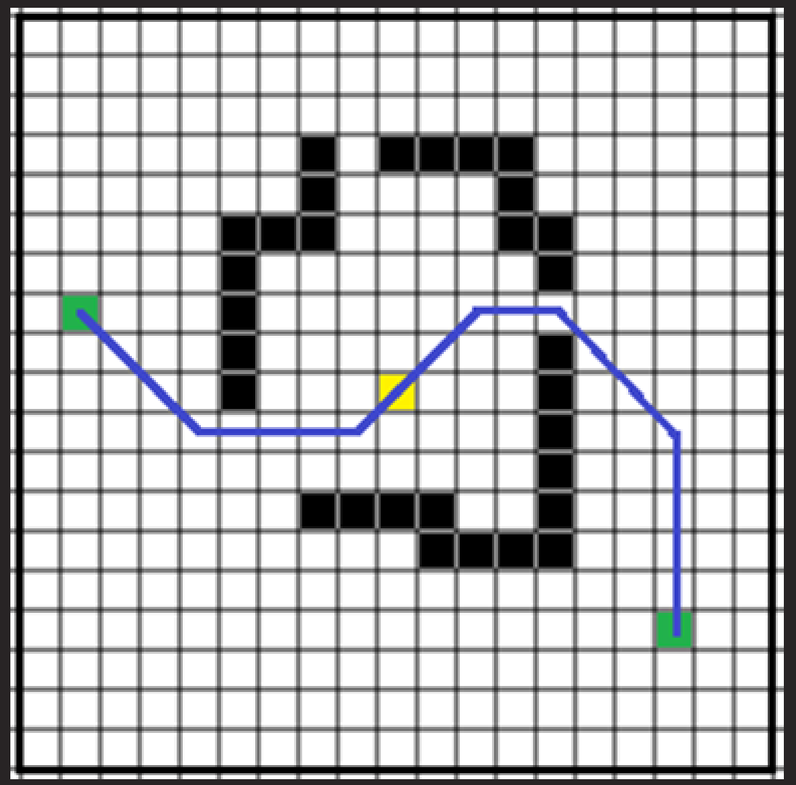
- Support for any platform
- Standalone library and UE4 plugin
- Audio engine agnostic
- 100% linear scalability
- Support for unlimited emitters



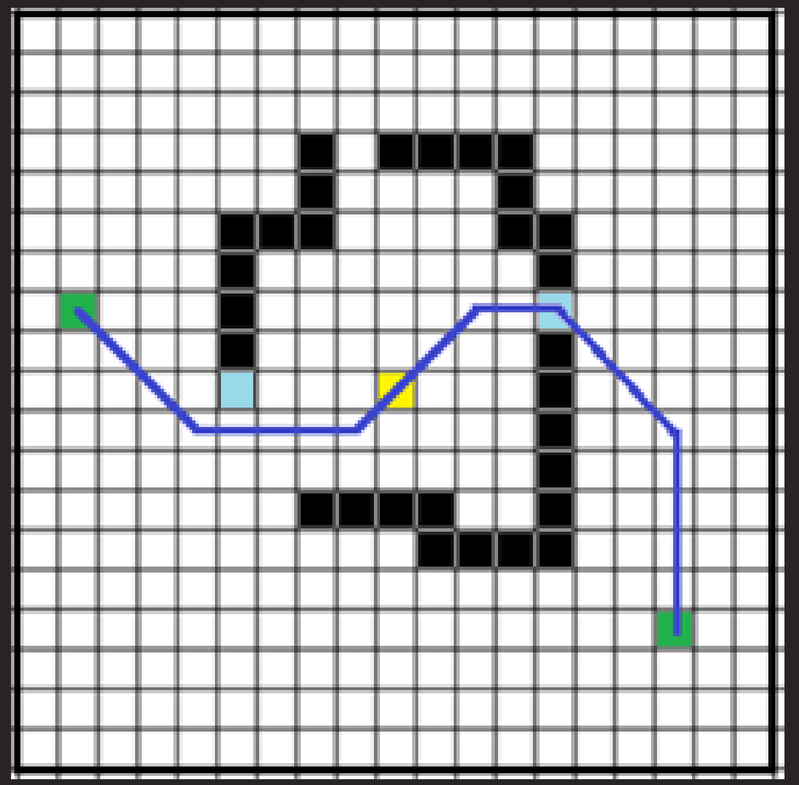
VERA's Core

- Step 1: Voxelize
  - Optimized system for converting 3D geometry into voxels
  - Voxels have "density"
- Step 2: Set a reference point
  - Usually the point where the listener is in relation to geometry
- Step 3: Floodfill
  - Propagate "movement" data throughout voxel space
- Results
  - A 3D map of the cost for audio to propagate from the reference point

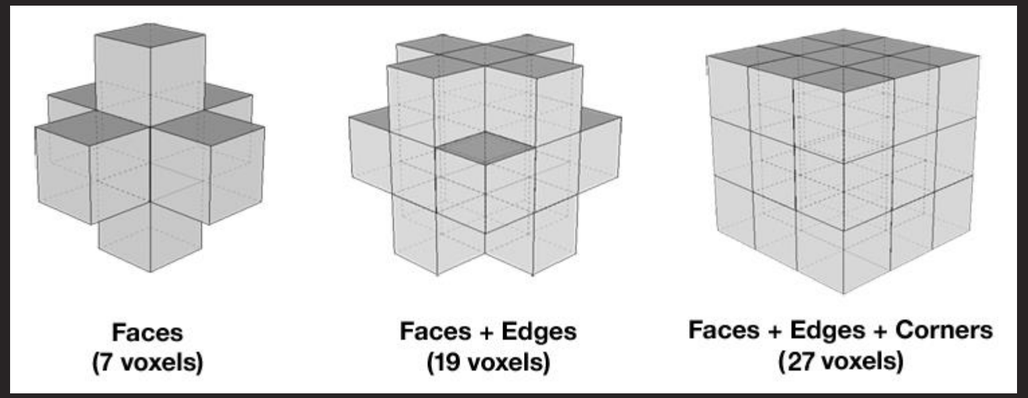
Obstruction



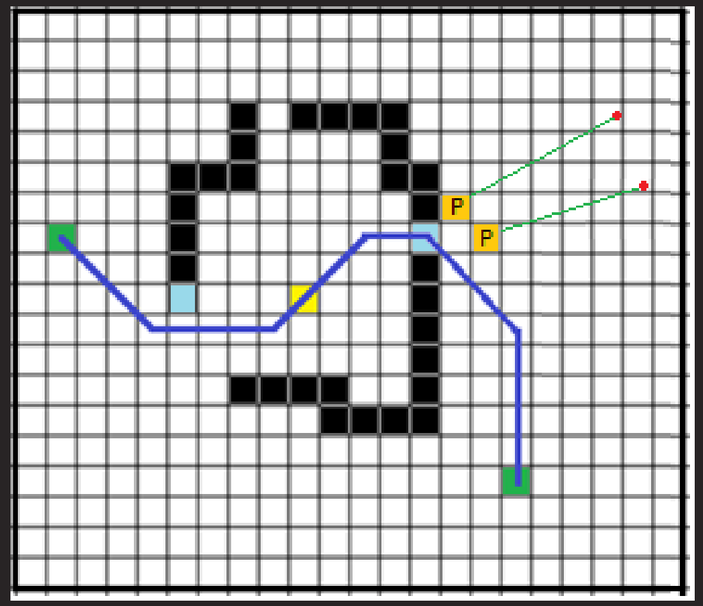
Occlusion



Environment

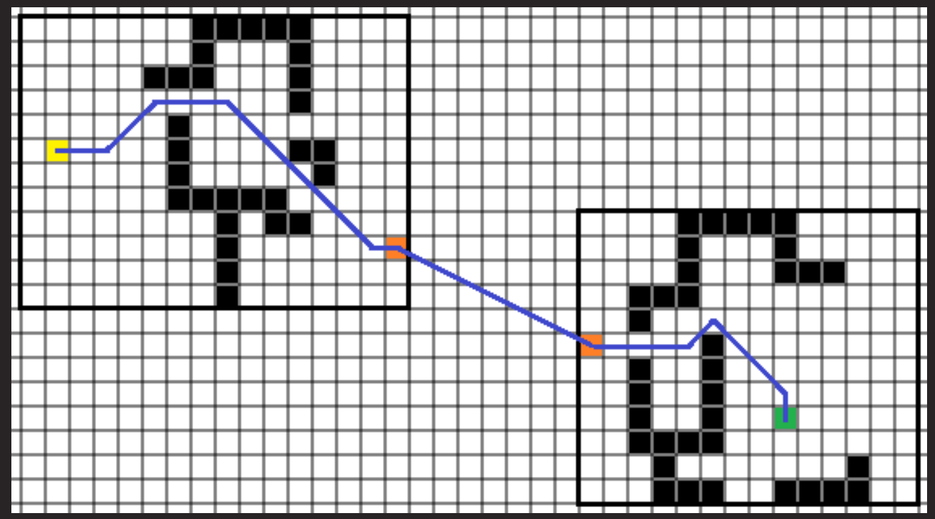


Portaling

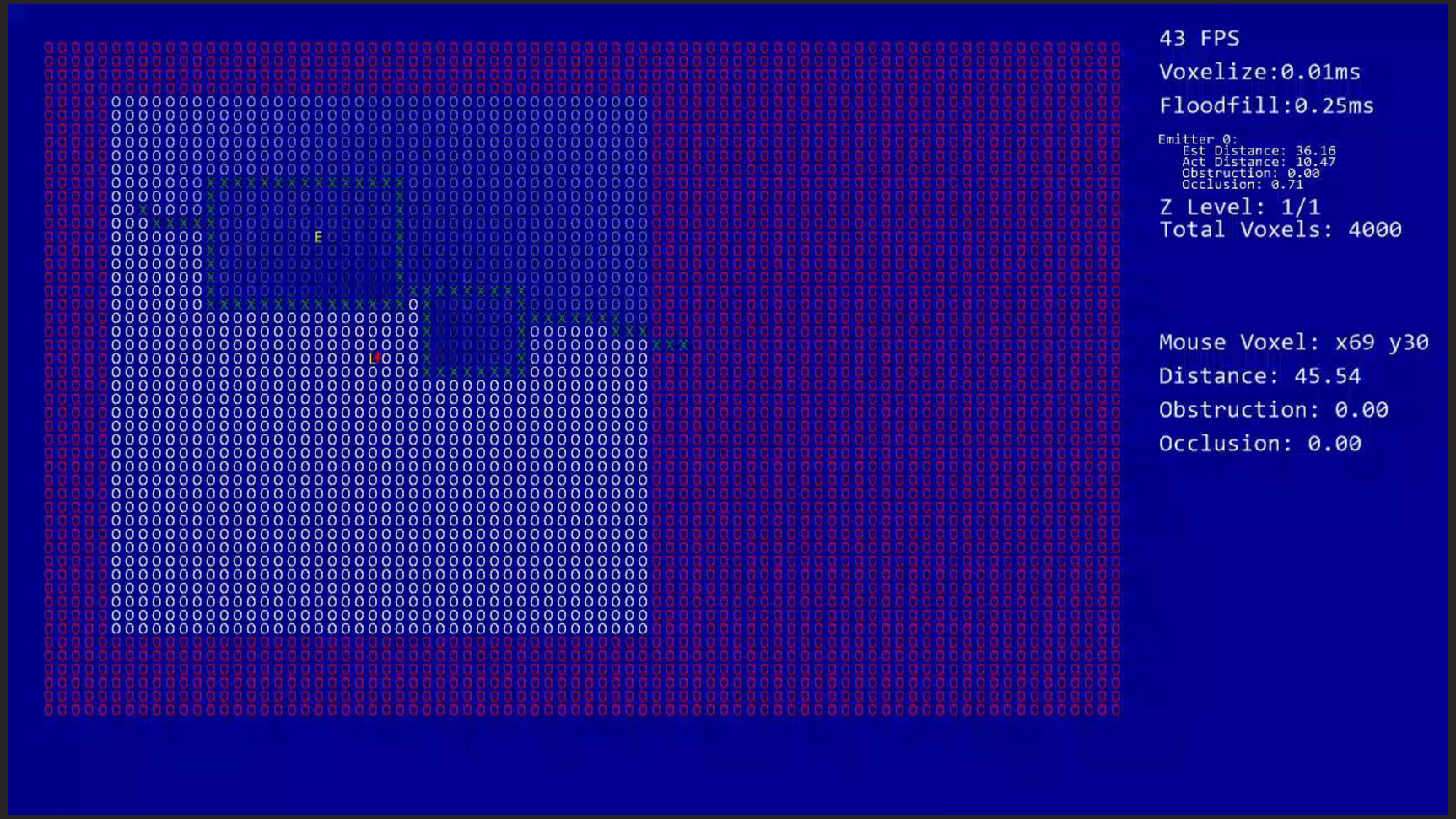


Handling Large Environments

- Multiple voxel spaces
- Variable level of detail (LOD)



Video: 2D Sample



Case Studies

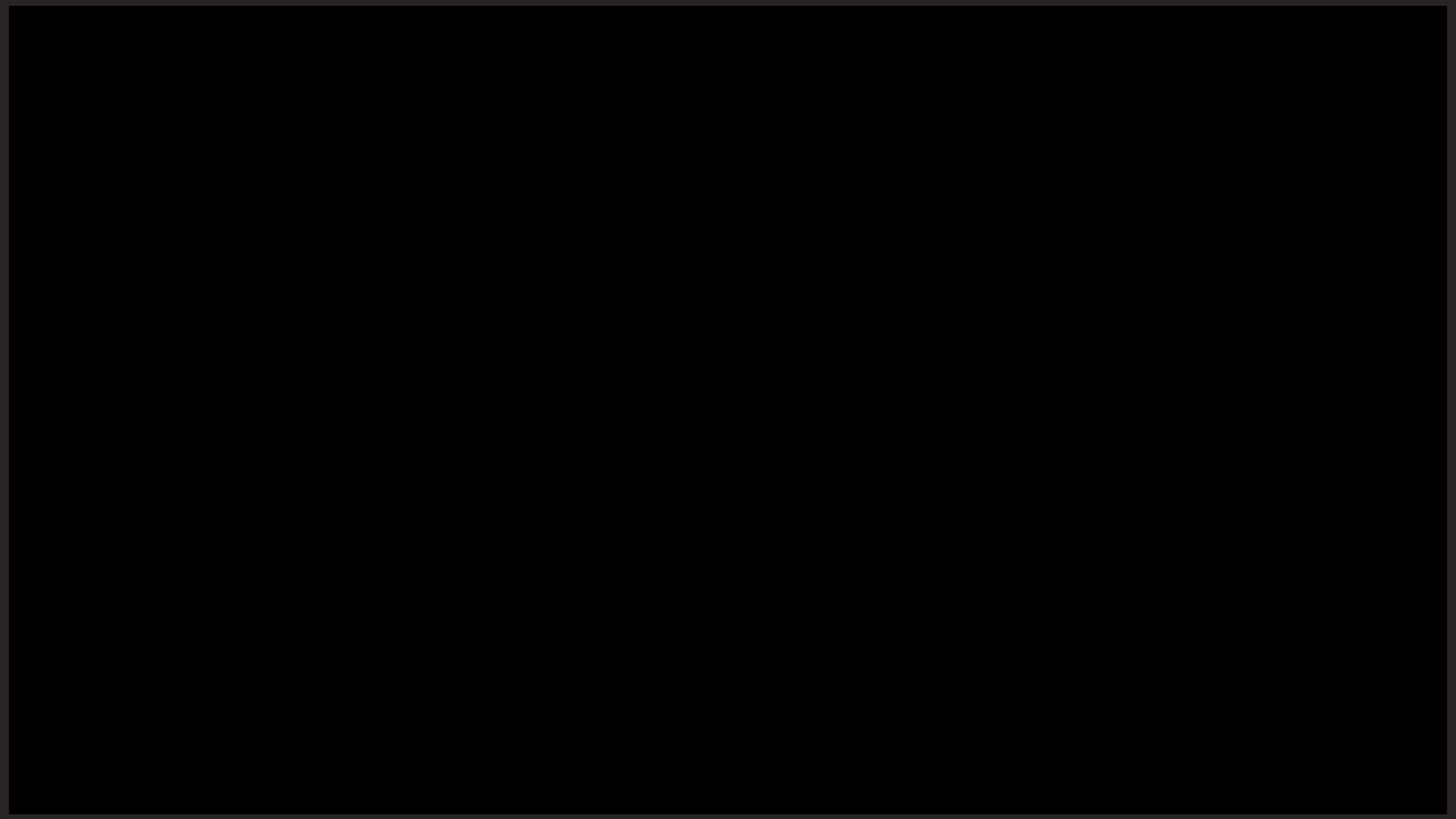
Case Study: Minecraft

- Perfect match for voxel engine
- Obstruction/Occlusion in a fully dynamic world
- Scale to target slower PCs
- Small audio space, faster updates

Case Study: Minecraft Reverb

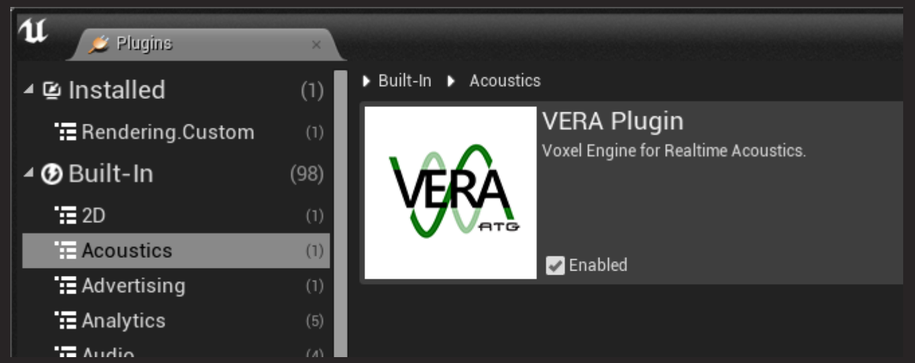
- Only uses +/- on 3 cardinal axis
- Finds which are “open”
- Get the area around the player
- Treat data differently based on if there is geometry overhead
- Adjust decay times based on area around the player

Video: Minecraft



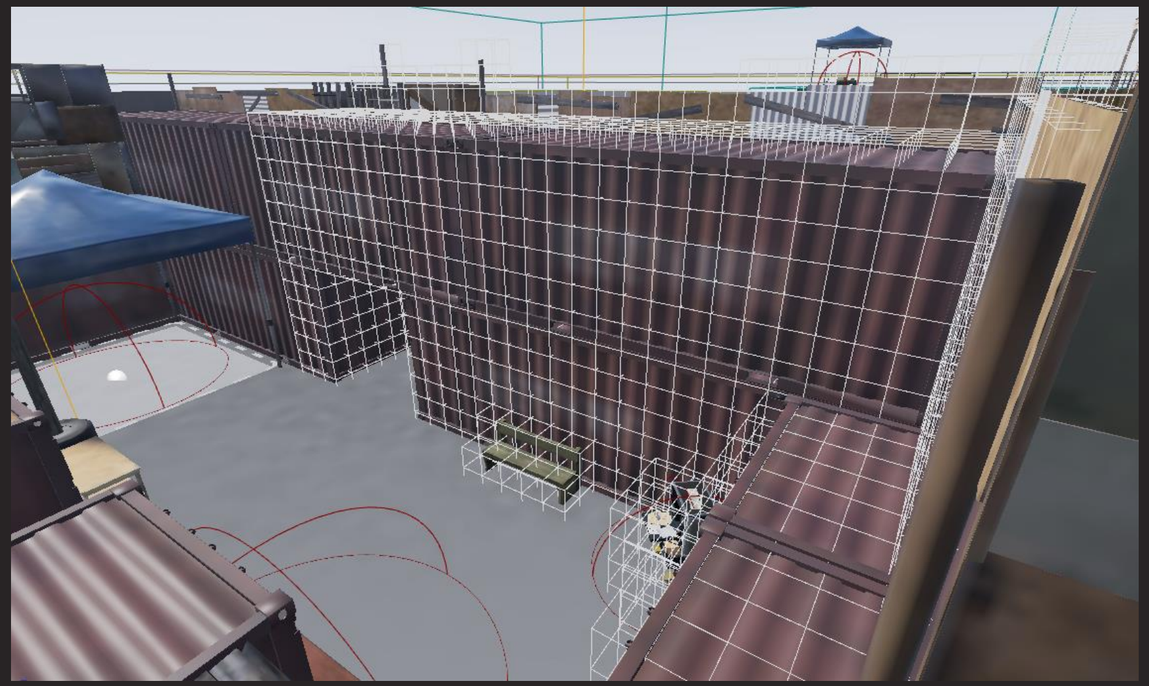
Case Study: UE4 Plugin

- Drop in integration
- Matches UE4 paradigms (actors, components, etc)
- Fully scriptable in blueprint
- Debug tools for visualization and performance
- Numerous feature toggles
  - Density
  - Portaling
  - Threading
  - Memory limits



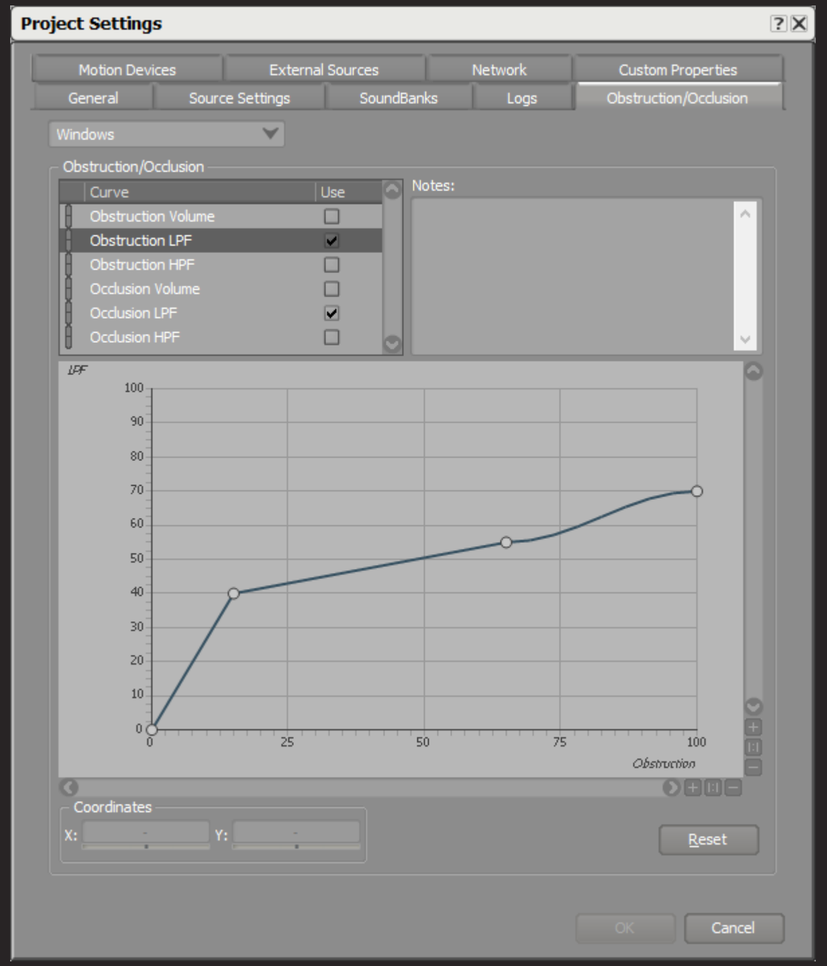
Case Study: State of Decay 2

- First large scale integration
- Improve dev tools for optimizing usage
- First UE4 exercise with diverse assets



Case Study: State of Decay 2

- Lerping to account for update delay
- Tweaking obstruction and occlusion curves



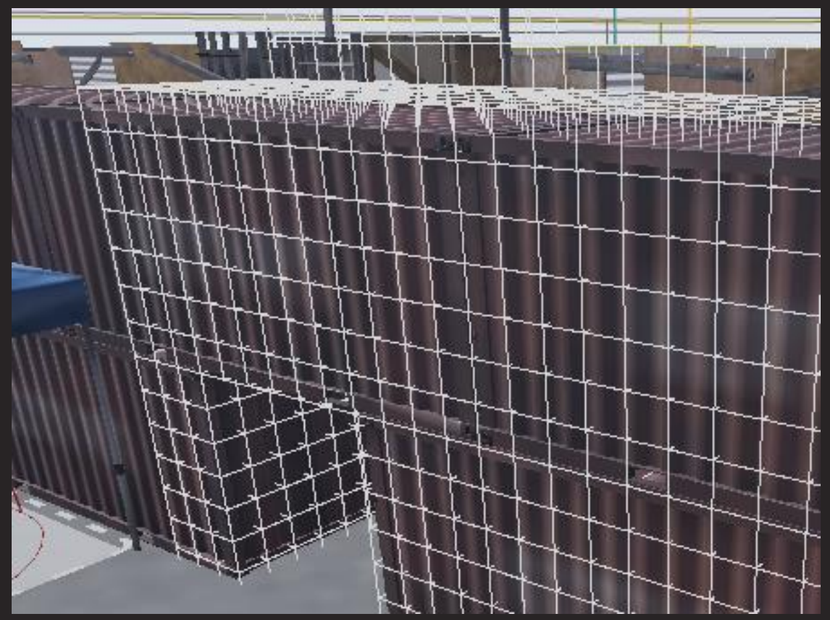
Video: State of Decay 2

Implementation Challenges

- Geometry collection
  - Potential impact on physics engine
  - Filtering
- Fitting in to memory and CPU requirements
  - Linear scaling
- Title specific integration

Implementation Challenges

- Audio listeners inside voxels
- Audio emitted from inside of voxels



Wrap Up

- Audio propagation is essential for immersion
- Implementing audio propagation is challenging
- Early implementation pays off
- Research and development is ongoing for VERA and more!

Thank You!

Jeff Ballard
@lifespan
Software Engineer – Microsoft ATG

Robert Ridihalgh
Senior Technical Audio Specialist – Microsoft ATG