The Audio Technology of

Guy Somberg

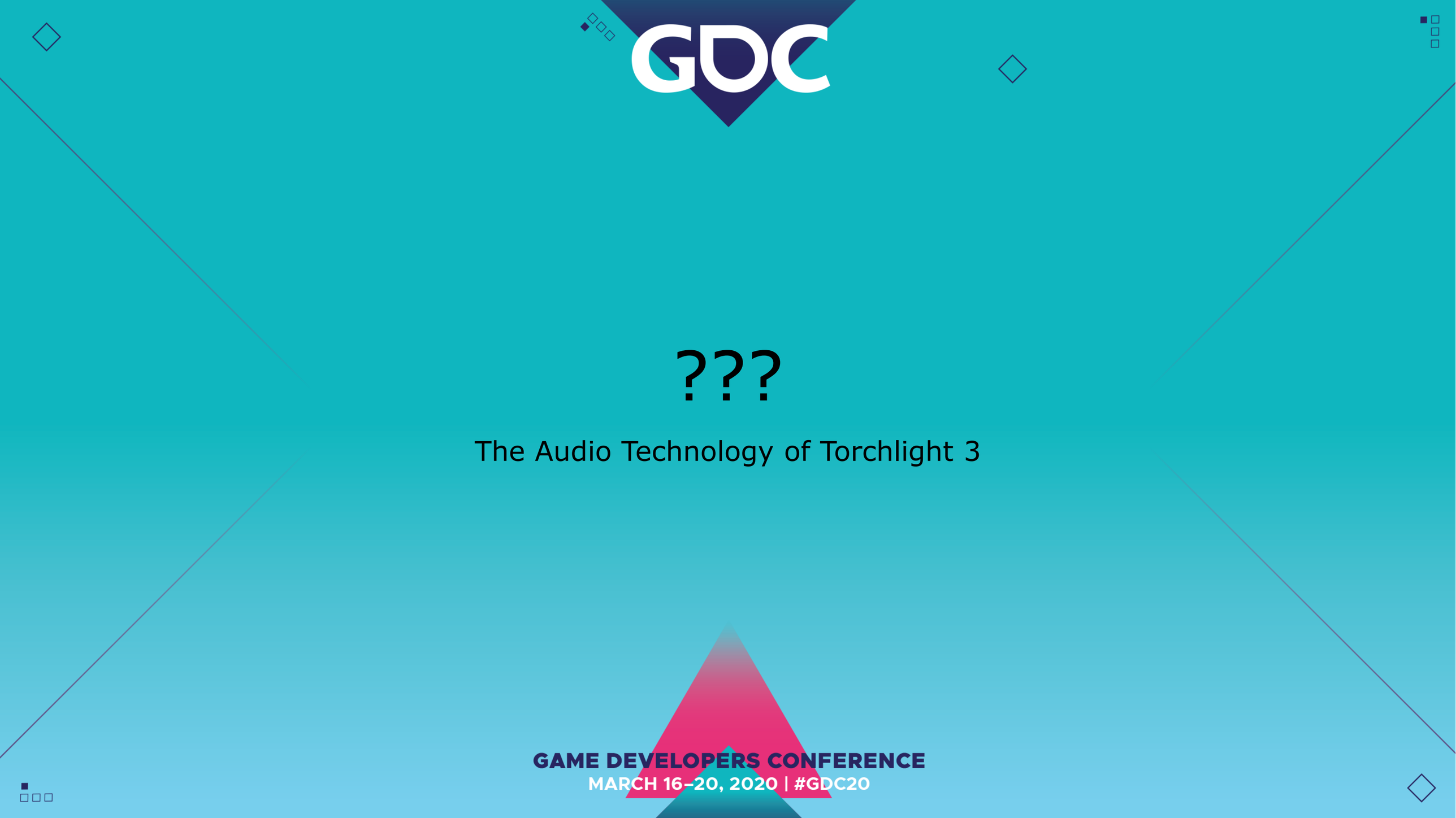
GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20

The Audio Technology of

Guy Somberg

GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20

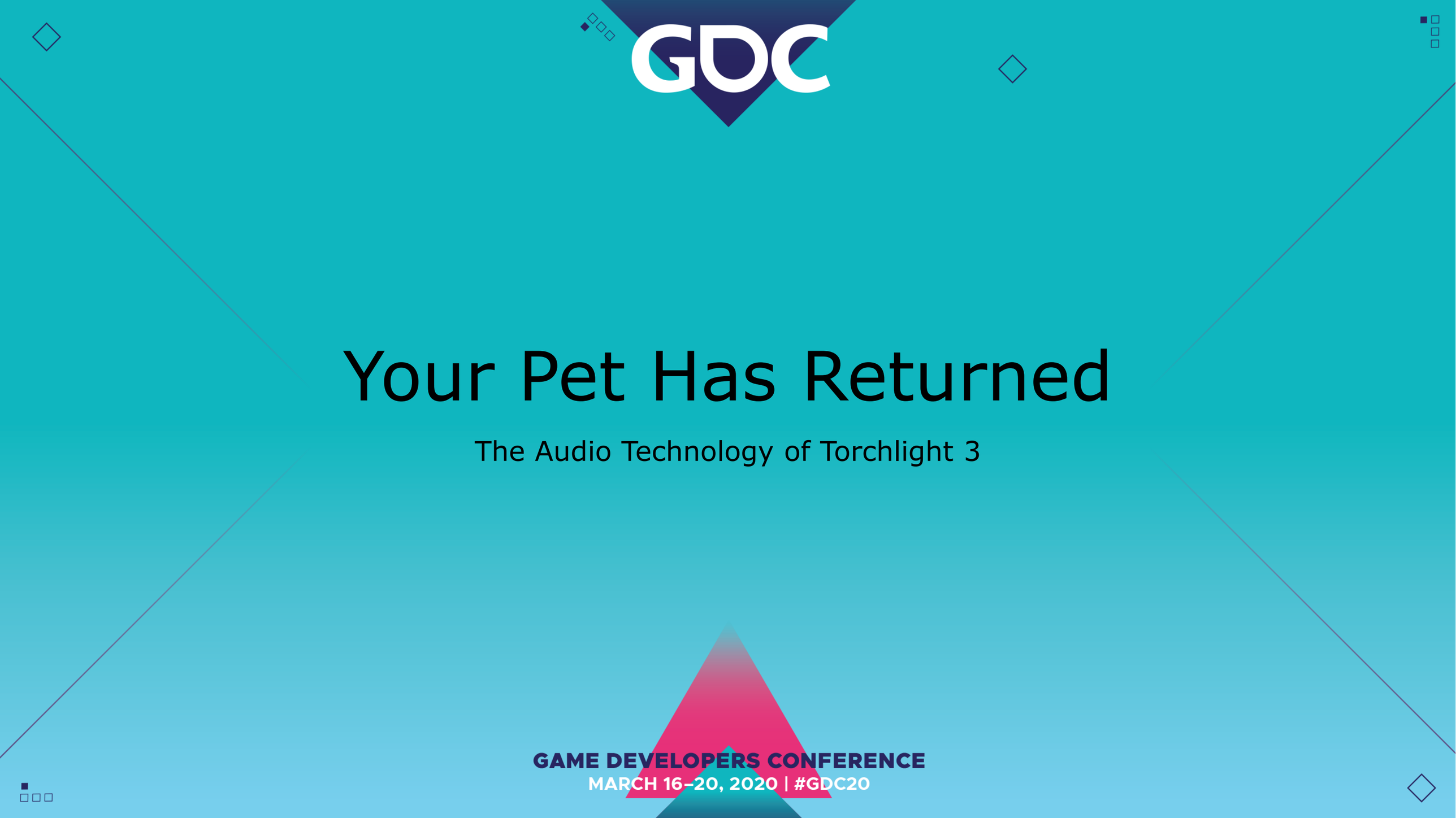

GDC
Short and Snappy Title
Long and Boring Subtitle
GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20



???

The Audio Technology of Torchlight 3

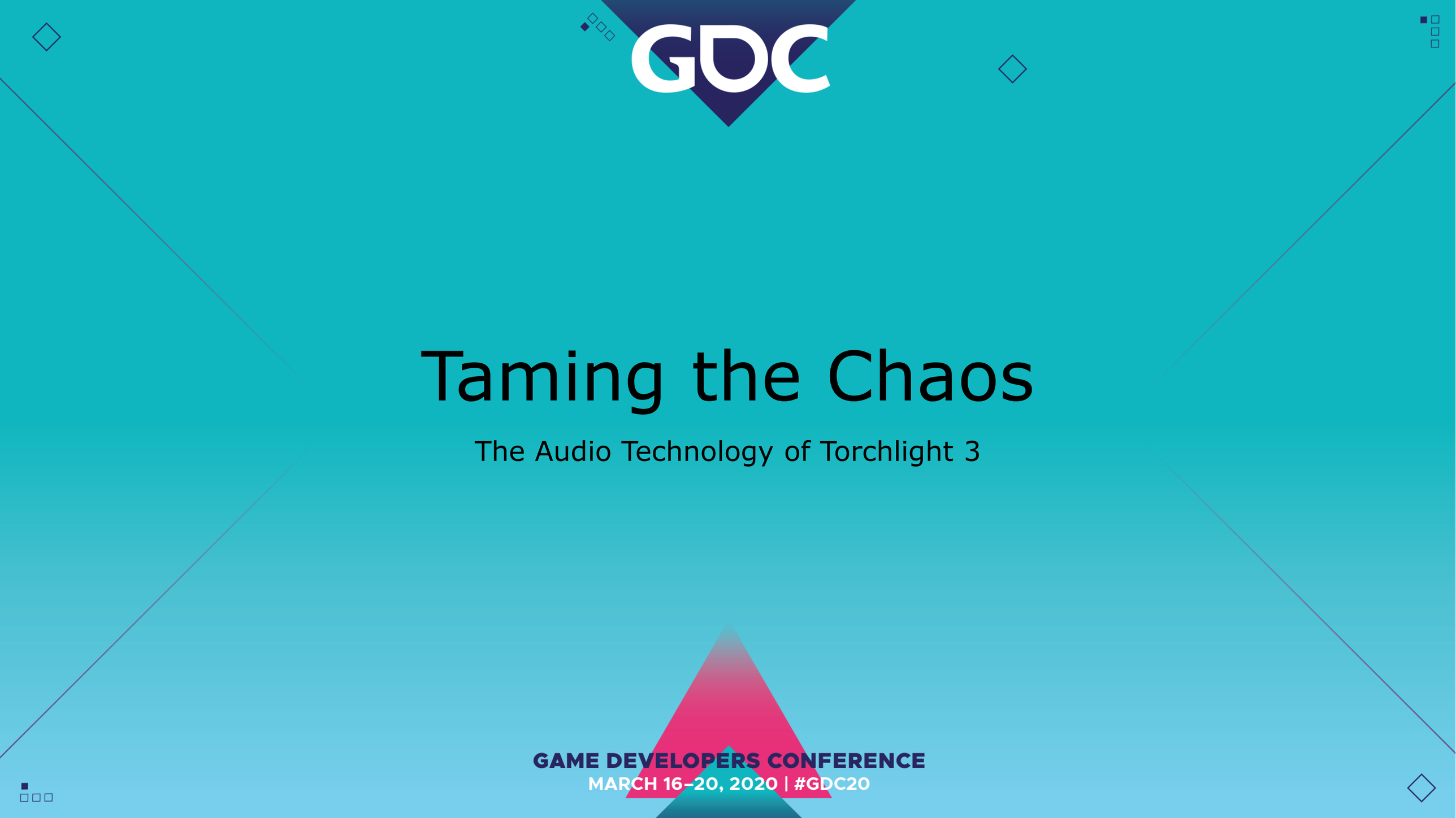
GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20



Your Pet Has Returned

The Audio Technology of Torchlight 3

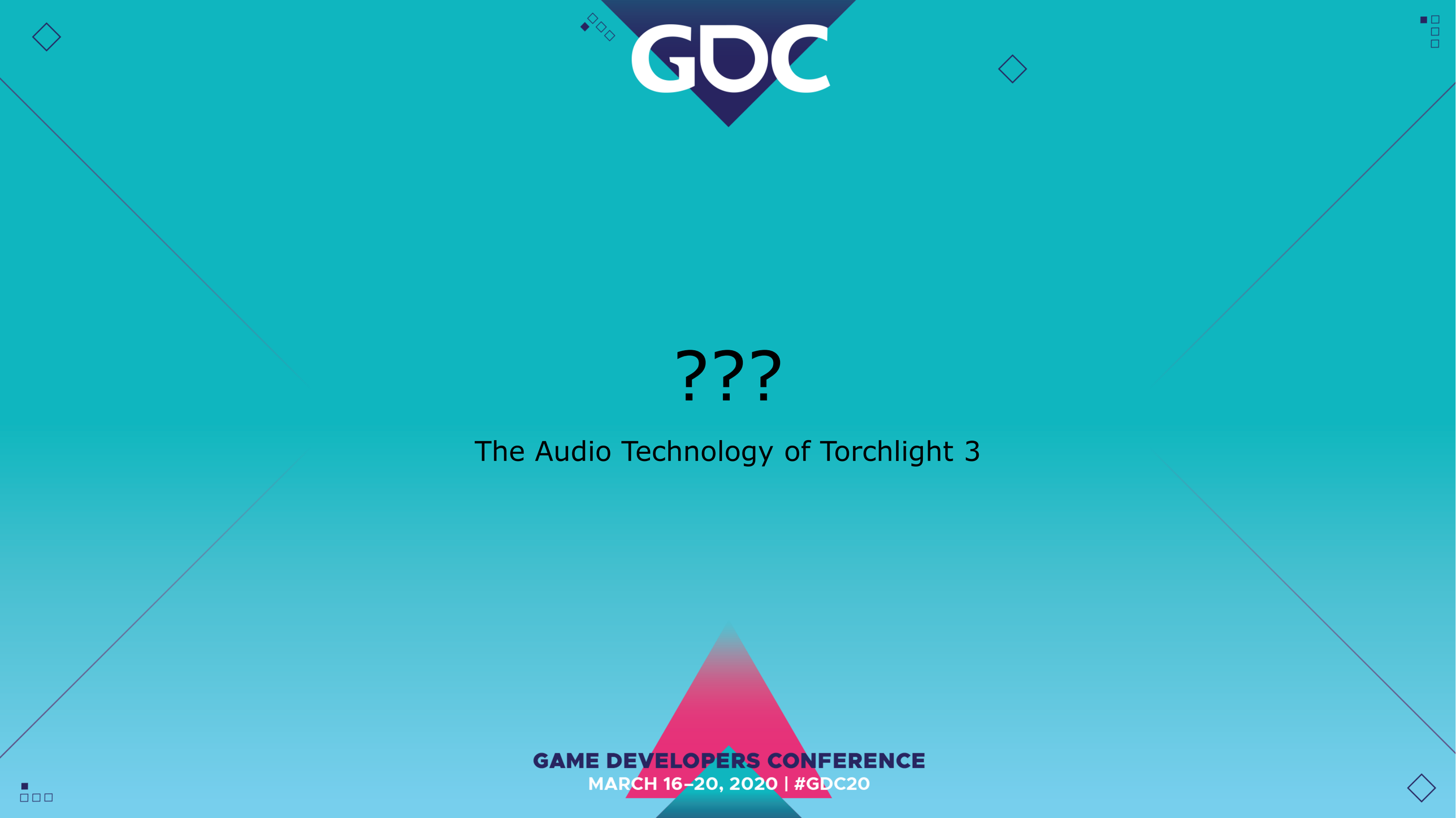
GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20



Taming the Chaos

The Audio Technology of Torchlight 3

GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20



???

The Audio Technology of Torchlight 3

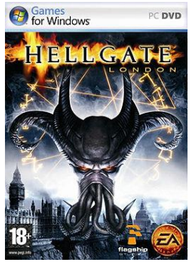
GAME DEVELOPERS CONFERENCE
MARCH 16–20, 2020 | #GDC20

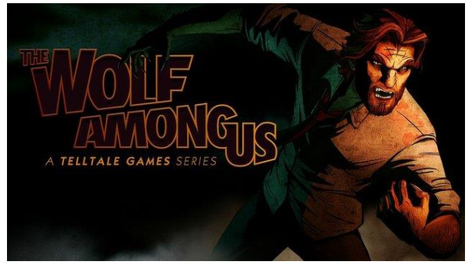
About Guy

- In games since 2002
- Owned the audio engine at (nearly) every company

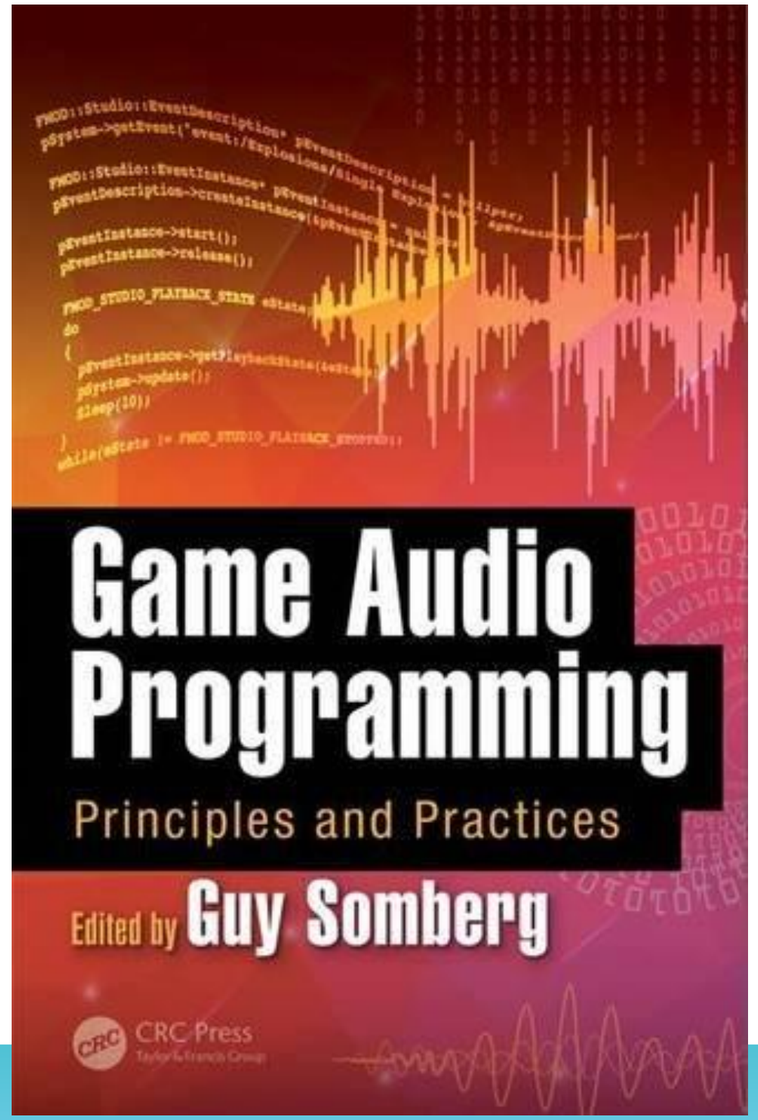
About Guy

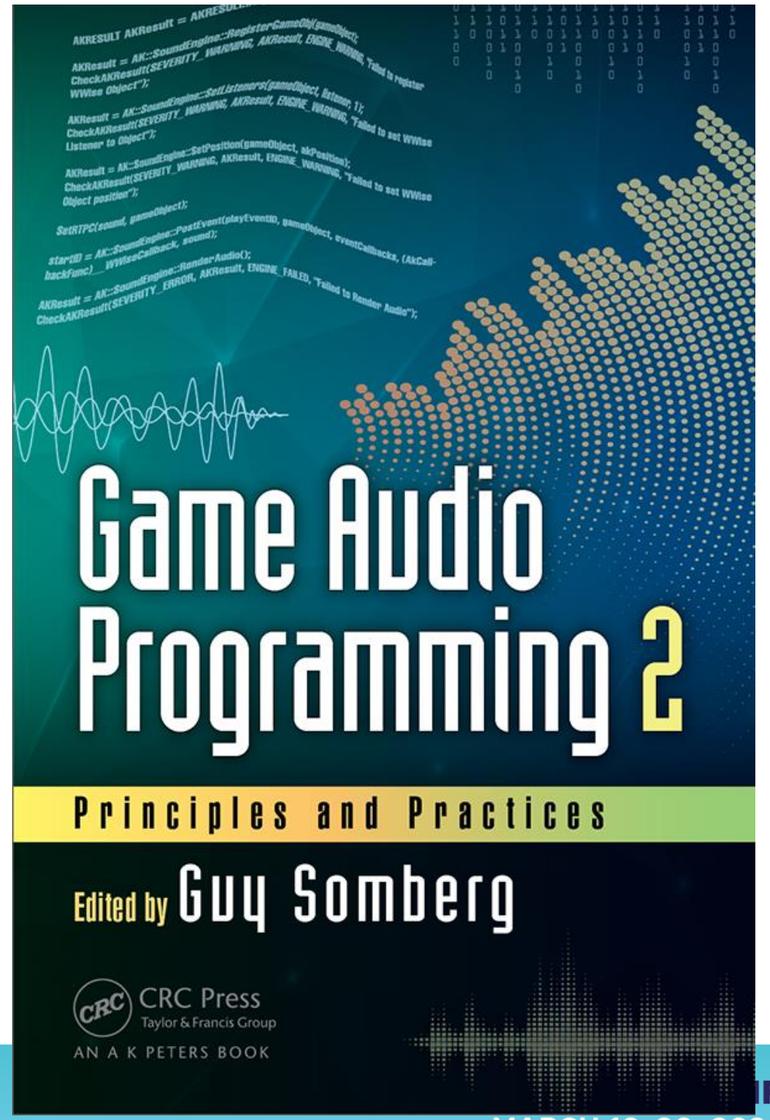
- …and shipped lots of games

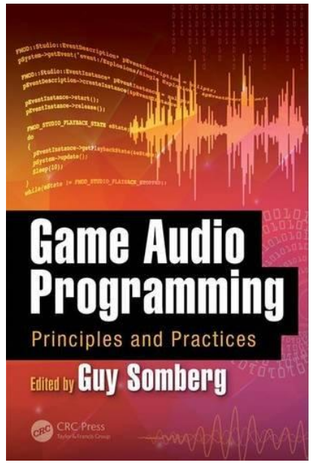



Shameless Plug

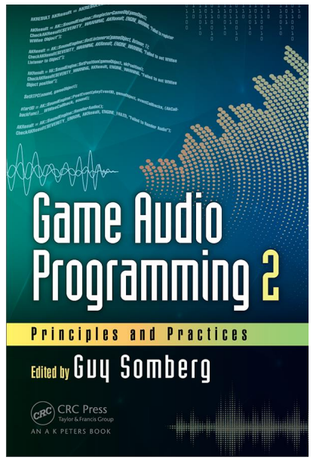




Shameless Plug



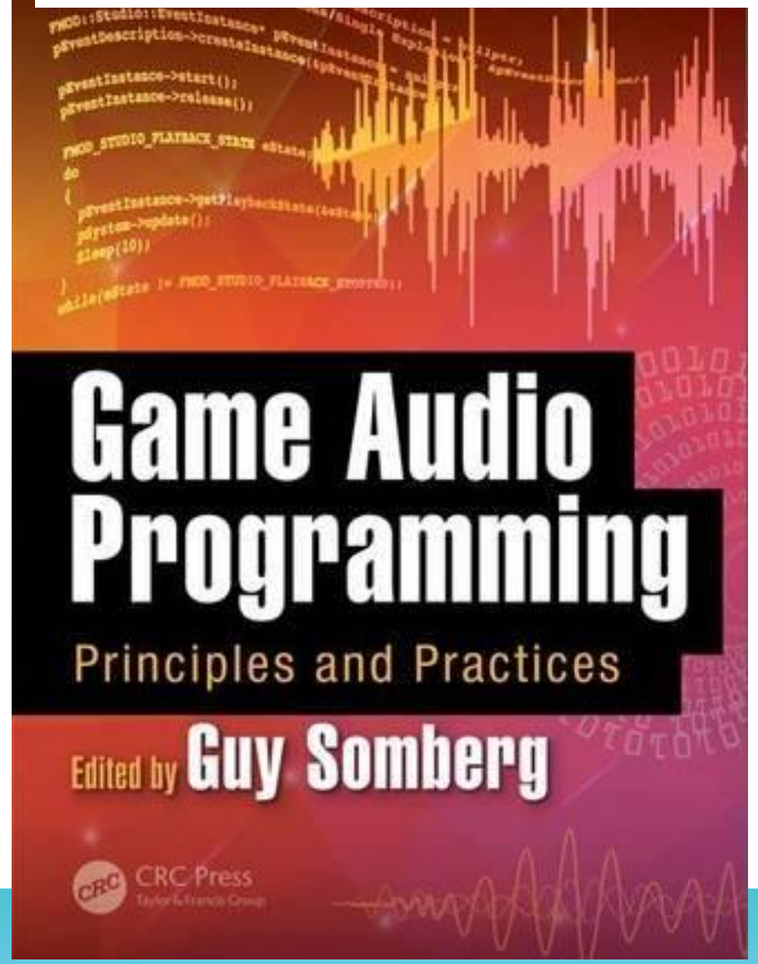
https://www.crcpress.com/Game-Audio-Programming-Principles-and-Practices/Somberg/p/book/9781498746731

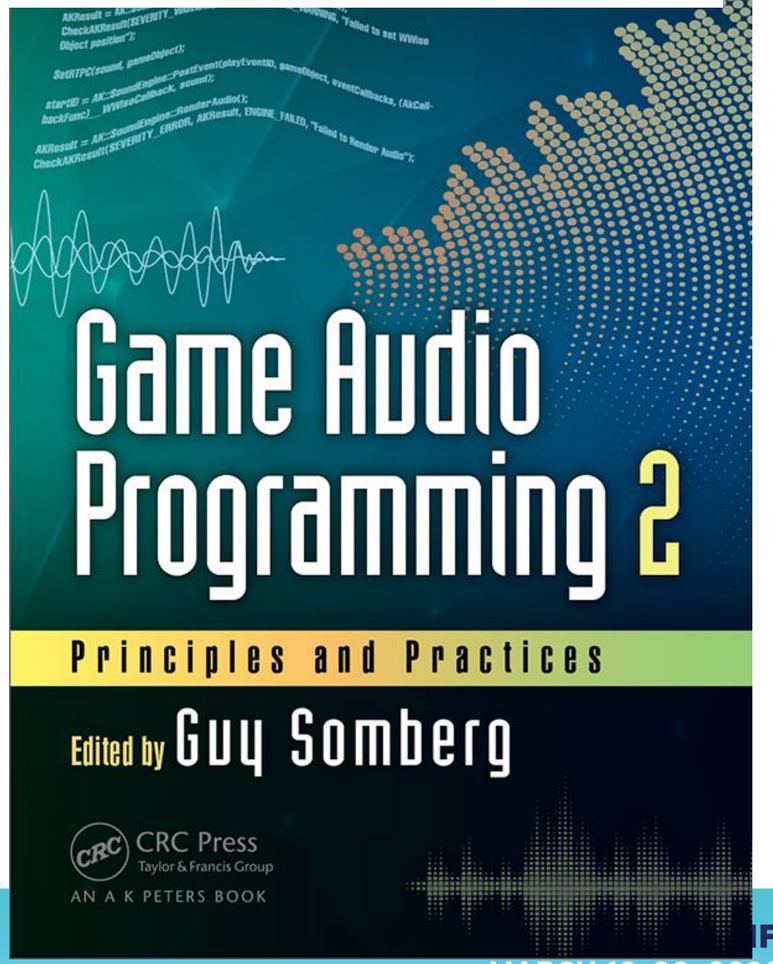


https://www.crcpress.com/Game-Audio-Programming-2-Principles-and-Practices/Somberg/p/book/9781138068919

Shameless Plug

Coming Soon: Volume 3!






TORCHLIGHT
FRONTIERS

ARPGs are Hard

- Everything is happening on the screen in front of you
  - See also: RTS, Adventure Games, etc.
- Chaotic action
  - See also: FPS, MOBA, etc.
- Randomized level layout
  - See also: Roguelikes, Strategy, etc.

Today’s Topics

- Importance-Based Mixing
- Volumetric Sounds
- Screen-Space Distance Attenuation

Not talking about:

- Narrator
- Music
- Modified FMOD Studio Unreal Plugin
- Tool-time Bank Building
    - Other cool stuff that we’ve done

Today’s Topics

- Importance-Based Mixing
- Volumetric Sounds
- Screen-Space Distance Attenuation

Mixing Woes

- Fundamental Problem:
  - Chaotic mix: dozens of events playing all at once
  - Traditional mixing techniques: Snapshots, Prioritization, Culling, HDR
    - All useful, and we use most of them
  - All insufficient when faced with our game

Early Attempts

- Special case: “Nearby monster count”
- Offline: Categorization
- Didn’t really solve the problem


The
Chasm
Level: 5
RedOctober_NOLOGIN0


The
Chasm
Level: 5
RedOctober_NOLOGIN0

Solution: Importance

- In Torchlight 3, Importance is the most important mixing technique that we use
  - Thanks to Tomas Neumann and Paul Lackey from the Overwatch team!

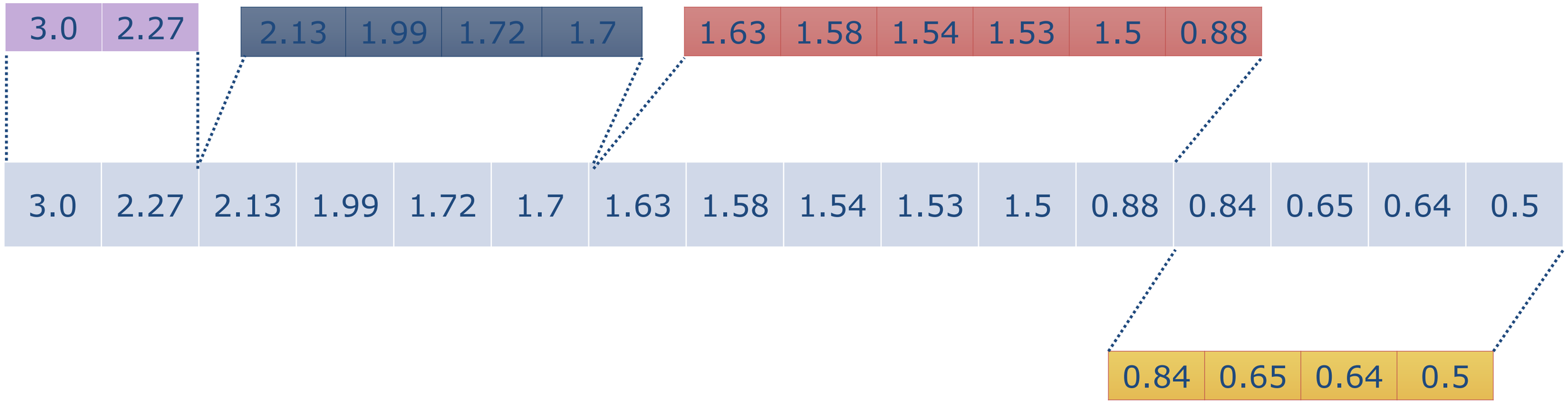
Importance

- Assign each object an importance score
- Sort all objects by score
- Place sorted objects into buckets
- Apply effect to each sound in the bucket

Importance Scores

- Extremely game-specific
- In Torchlight 3:
  - Identity
  - Distance to player
  - Relative “drama” score
  - Skill target
- Total score = weighted sum of individual scores


The Chasm
Level: 5
0.84
0.5
0.65
RedOcto
0.88
OGIN0
2.13
0.64
2.27
1.99
1.5
1.7
3.0
1.53
1.72
1.58
1.63
1.54


3.0
2.27
2.13
1.99
1.72
1.7
1.63
1.58
1.54
1.53
1.5
0.88
3.0
2.27
2.13
1.99
1.72
1.7
1.63
1.58
1.54
1.53
1.5
0.88
0.84
0.65
0.64
0.5
0.84
0.65
0.64
0.5


The Chasm
Level: 5
0.84
0.5
0.65
RedOcto
0.88
OGIN0
2.13
0.64
2.27
1.99
1.5
1.7
3.0
1.53
1.72
1.58
1.63
1.54

Importance Bucket Effects

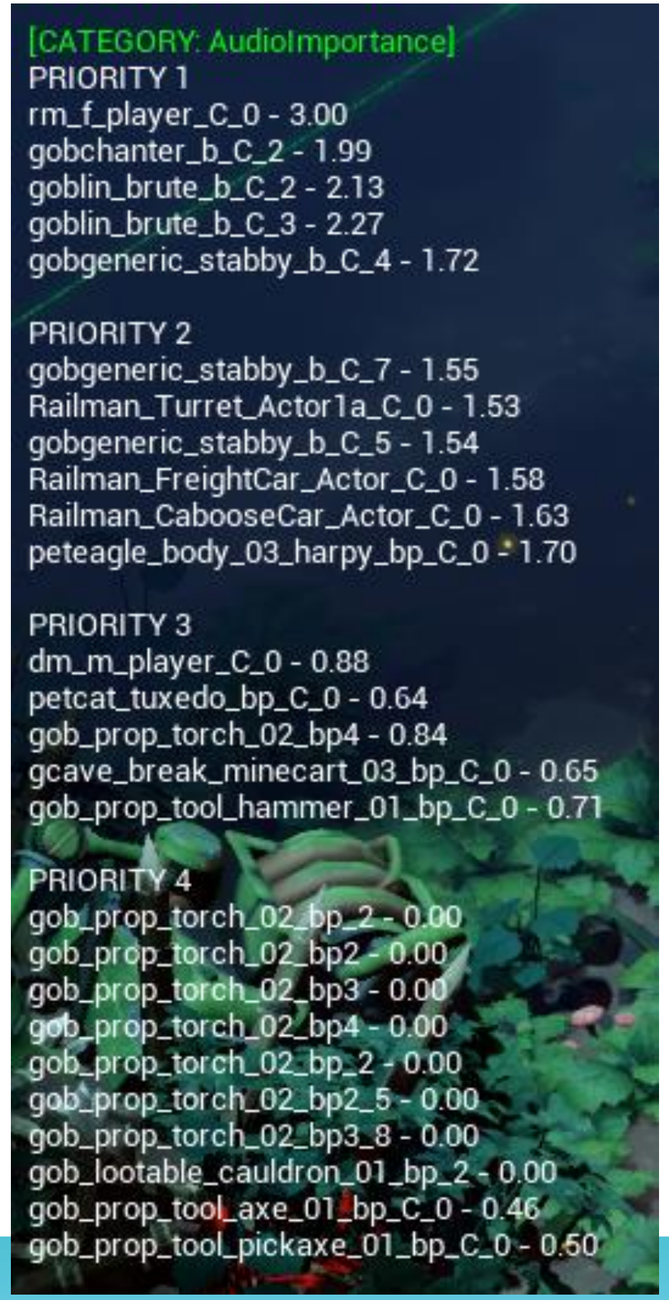
- Volume, Peaking Filter, High Shelf Filter
  - Implementation detail: Peaking and High Shelf Filters implemented using Multiband EQ

Priority	Effect
1	Peaking Filter
2	No change
3	Volume Reduction
4	Volume Reduction, High Shelf
5	Volume Reduction, High Shelf

Importance in Action

- Live Demo

Debug Display



Debug Display

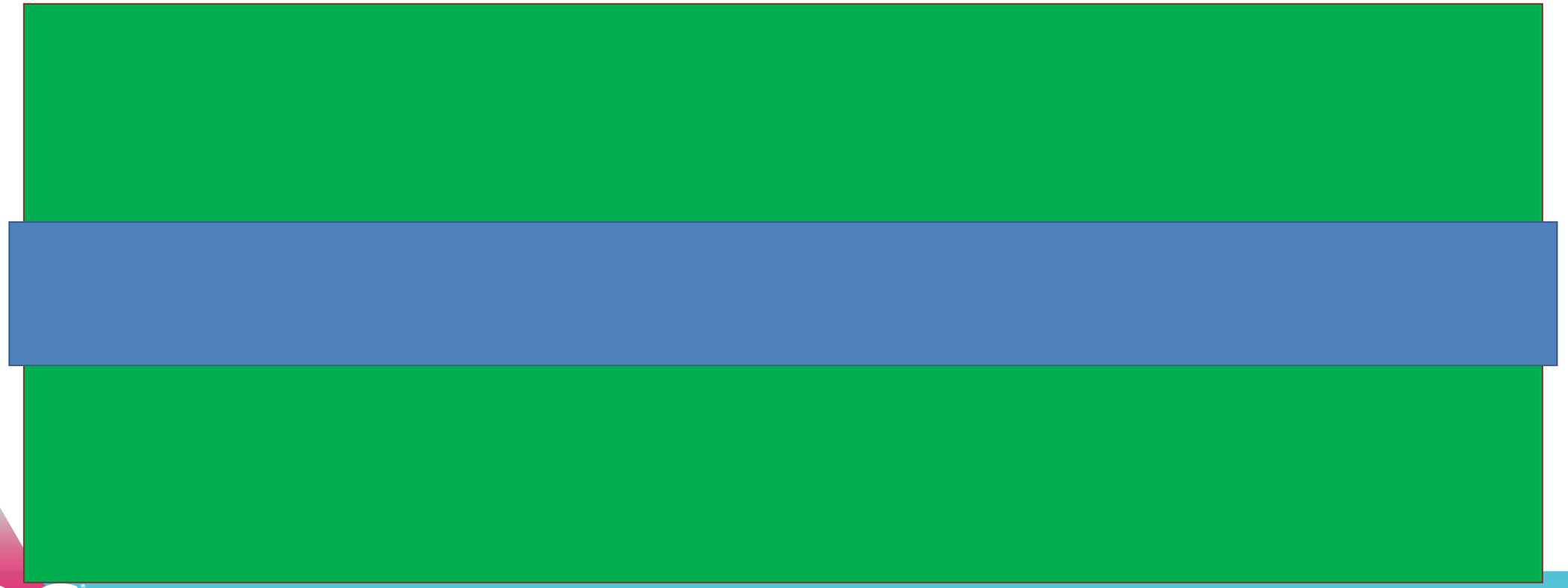
Debug Display

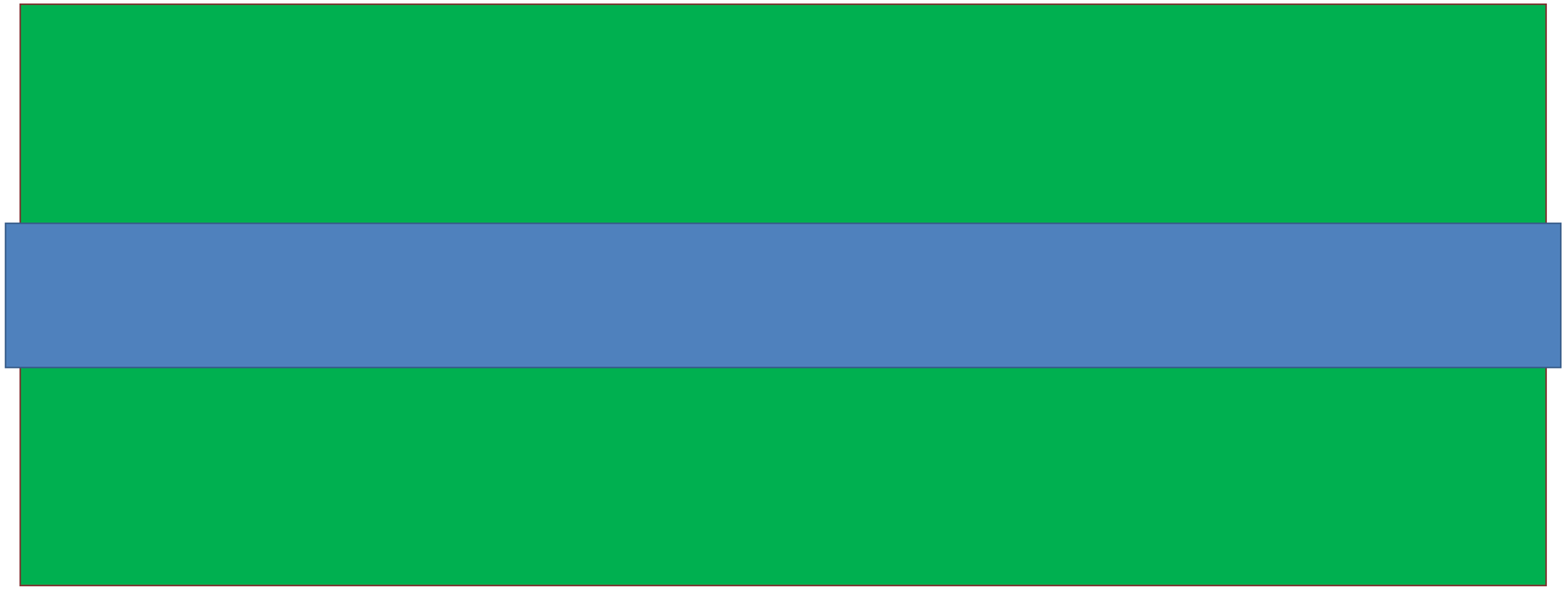
Today’s Topics

- Importance-Based Mixing
- Volumetric Sounds
- Screen-Space Distance Attenuation

The Problem

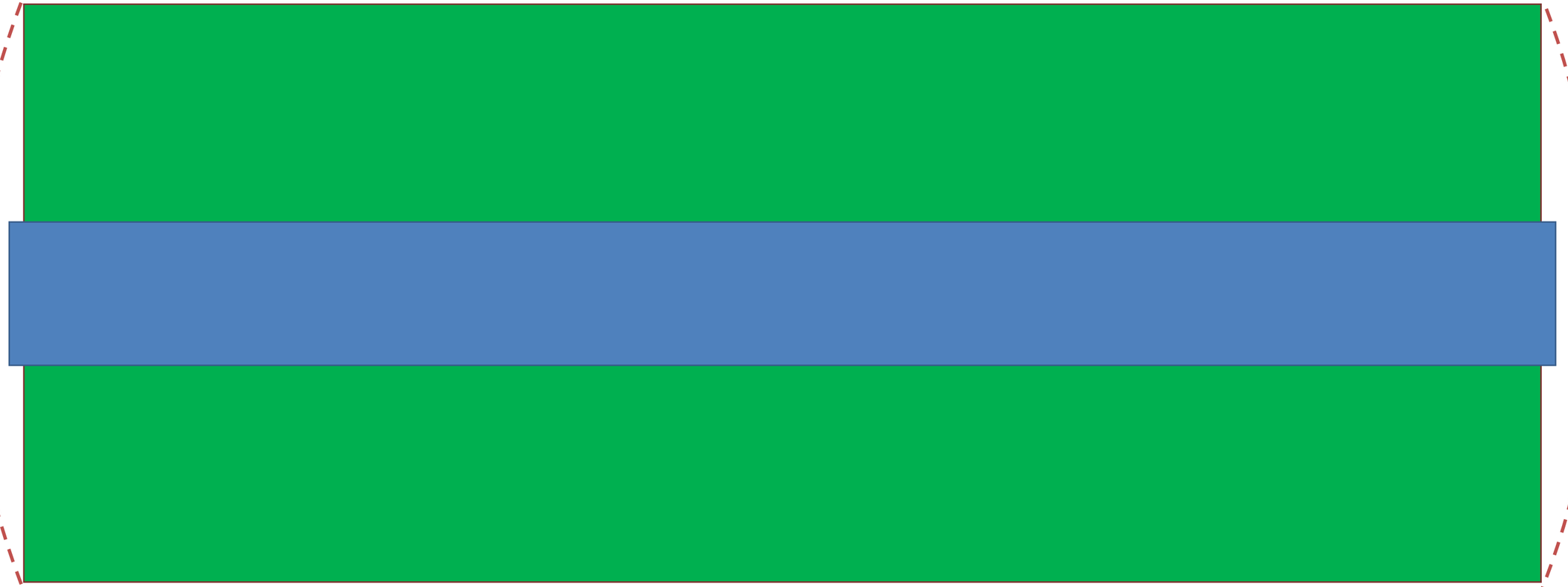
- In one word: rivers





Failed Idea #1

- Single sound source at the river center

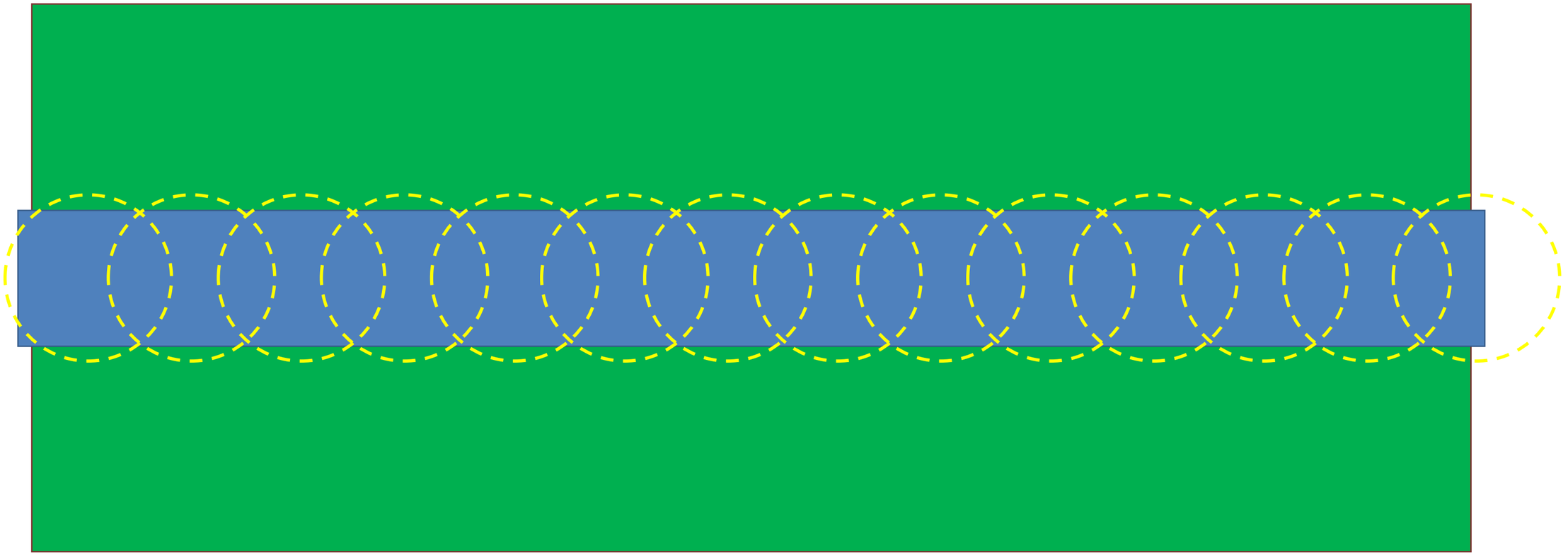


Nope!

- No control over attenuation
- No control over shape
- Doesn’t account for randomized levels

Failed Idea #2

- Many smaller sound sources



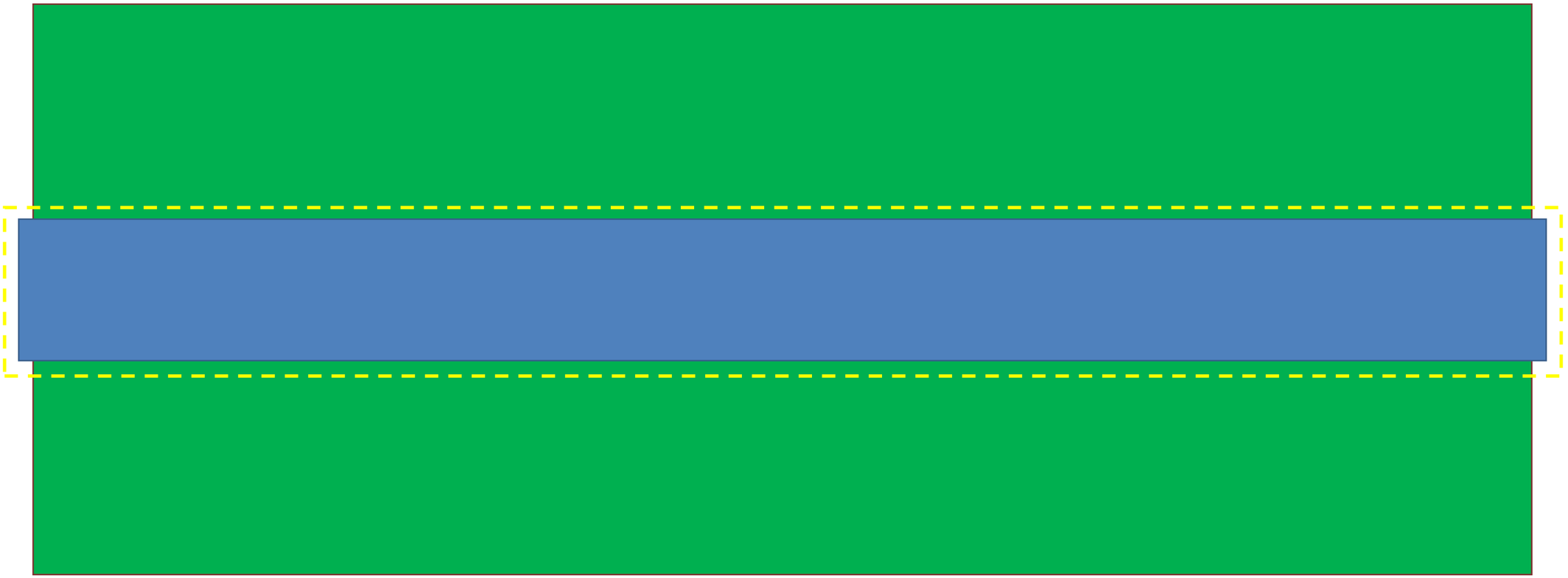
Nope!

- Control over attenuation
- Control over shape
- Deals with random levels

- But:
  - Too many sound sources
  - Phasing
    - Mind-numbing setup

Failed Idea #3

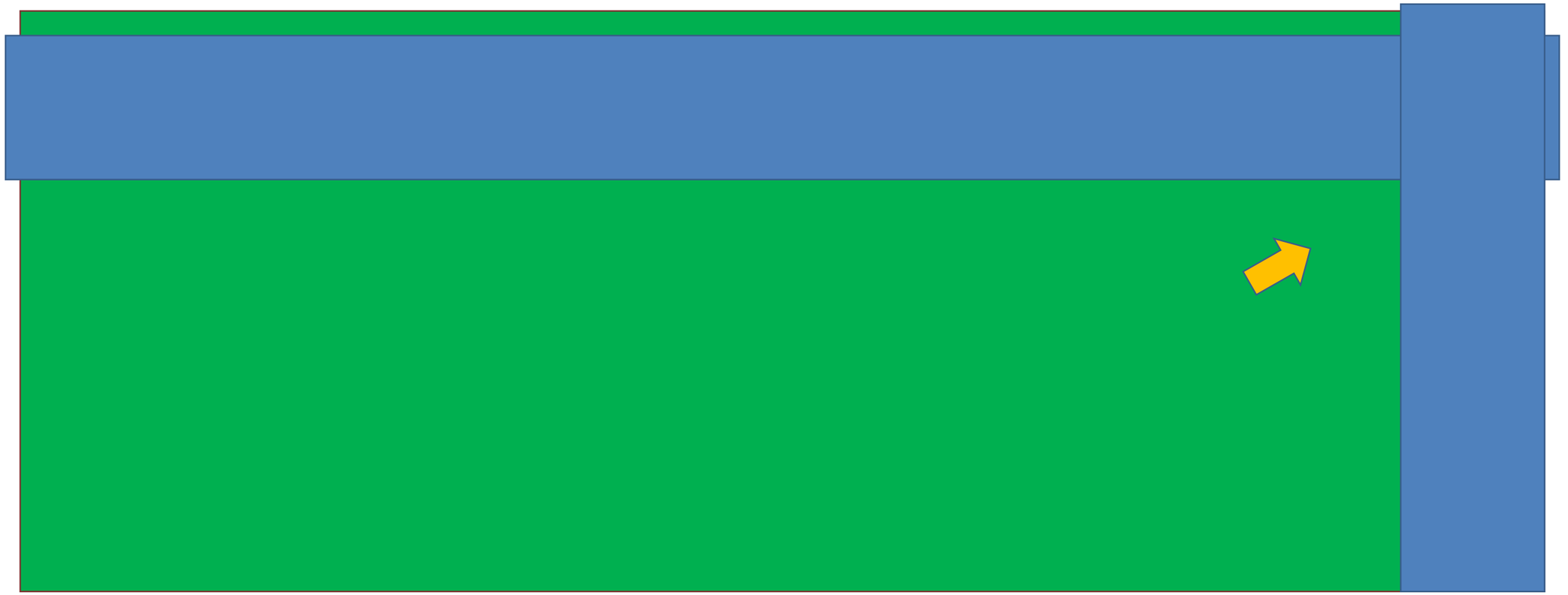
- Nearest point on box

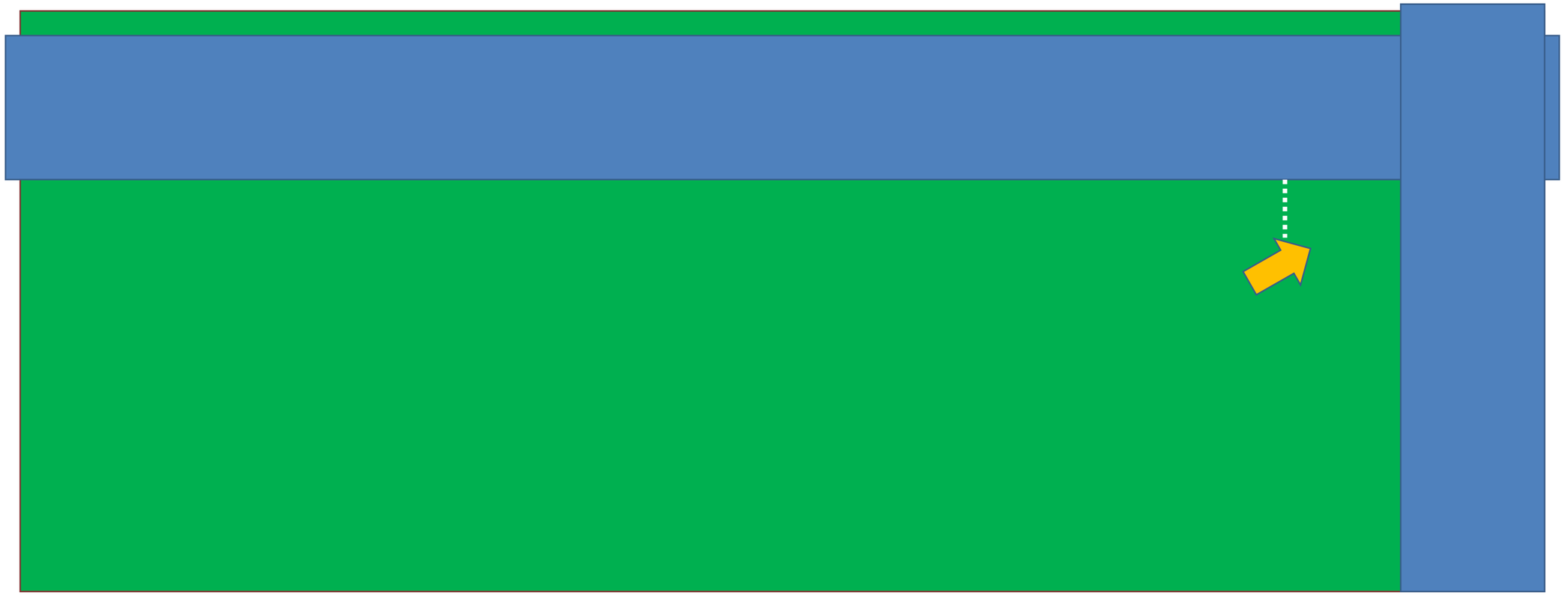


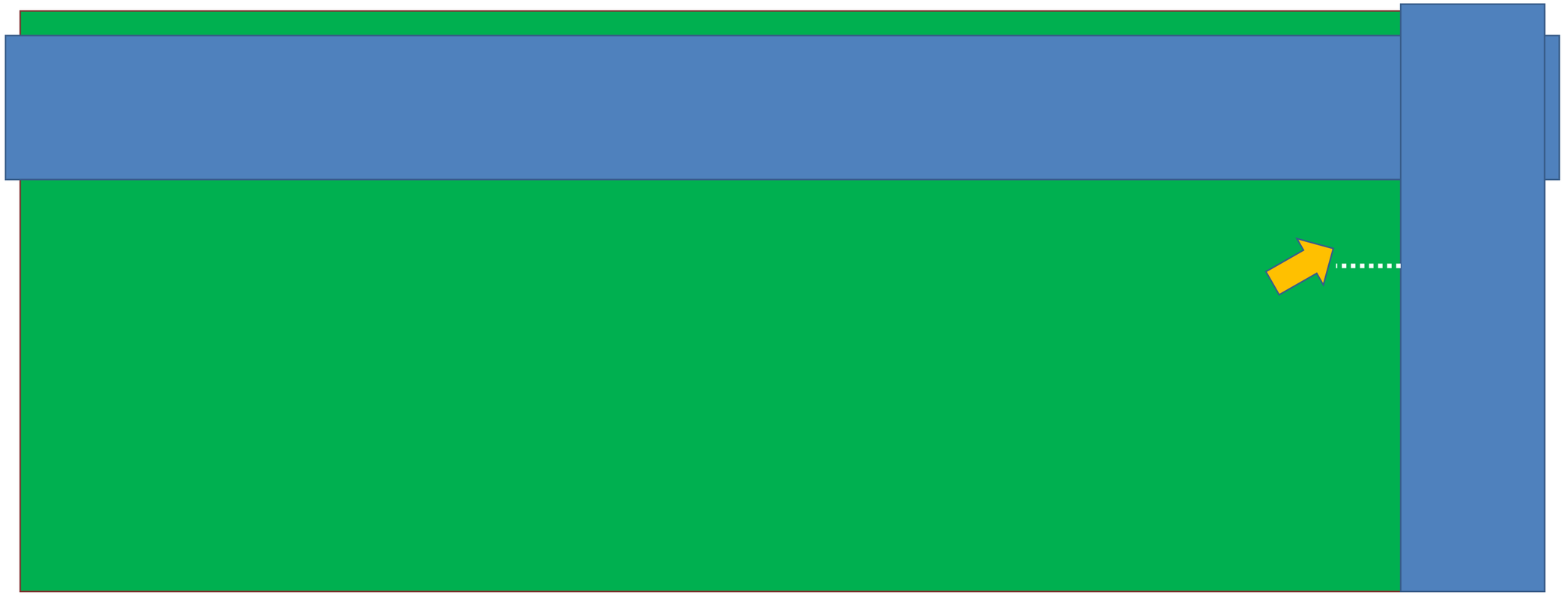
Promising!

- Control over attenuation
- Control over shape
- Deals with random levels
- Only one sound source
- Easy setup

- But…







Darn :(

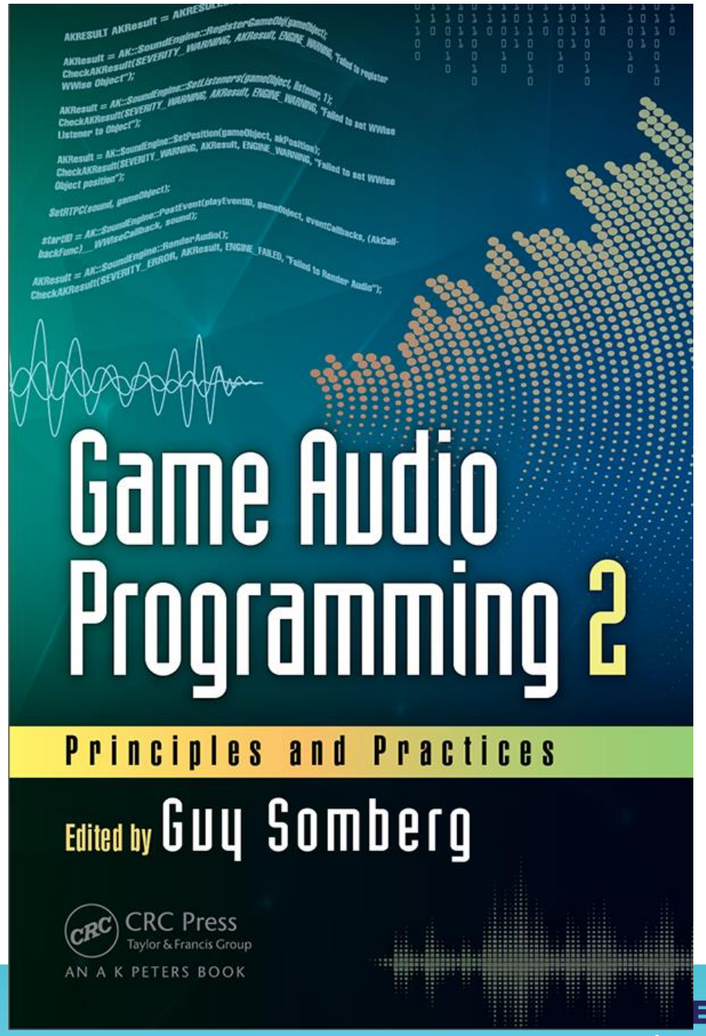
- So close!

Take a Step Back

- We want one single sound source
- Position + Direction is insufficient to describe our sound source
- So what do we actually want to describe?

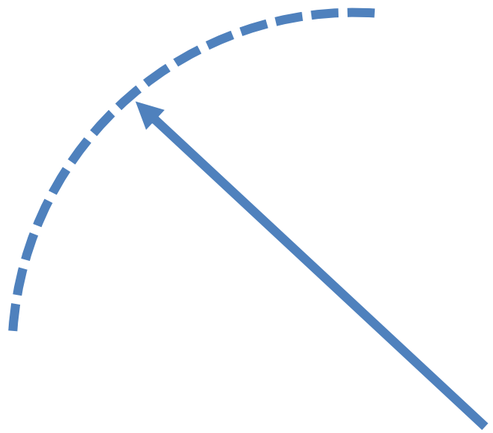
Working Solution

- Game Audio Programming Principles and Practices Volume 2
- Chapter 12: “Approximate Position of Ambient Sounds of Multiple Sources” by Nic Taylor
- Thanks, Nic!



Working Solution

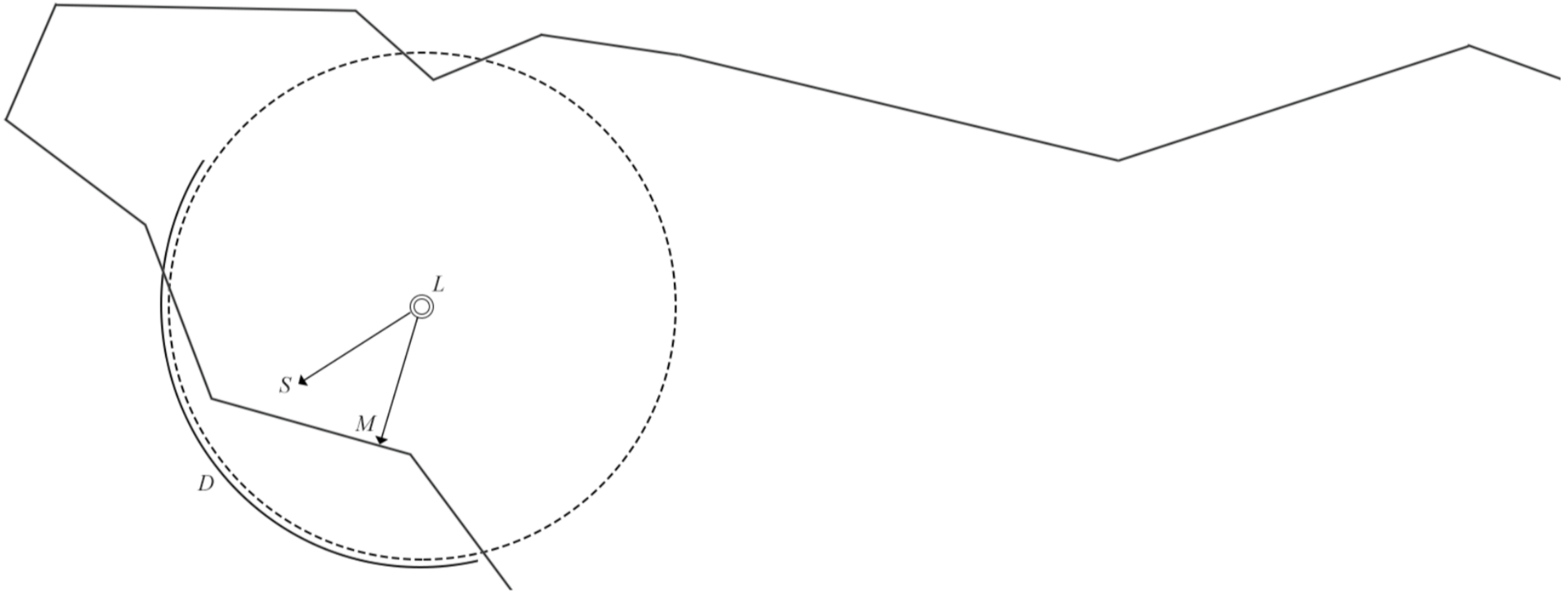
- Direction
- Magnitude
- Spread



Analytical Solution

- Direction and Spread should be impacted less the farther a point is from the listener
- Spread of two parallel lines with listener equidistant should be 1 (full spread)
- Spread of a line segment that passes through the listener should be 0
- Small changes in listener position should result in small changes to Direction, Magnitude, and Spread
  - Subdividing a line segment should not alter Magnitude, Direction, or Spread

Image Credit: Nic Taylor



Math!

$$\hat{\sigma} = \lim_{\Delta s_{i \to 0}} \sum_{i=1}^{n} \frac{\hat{v}}{\|\hat{v}\|} W(\hat{v}) \Delta s_i = \int_C \frac{\hat{v}}{\|\hat{v}\|} W(\hat{v}) \Delta s$$

But…

- All of that is just for line segments
- Upgrading to two dimensions gets us both more features and more flexibility

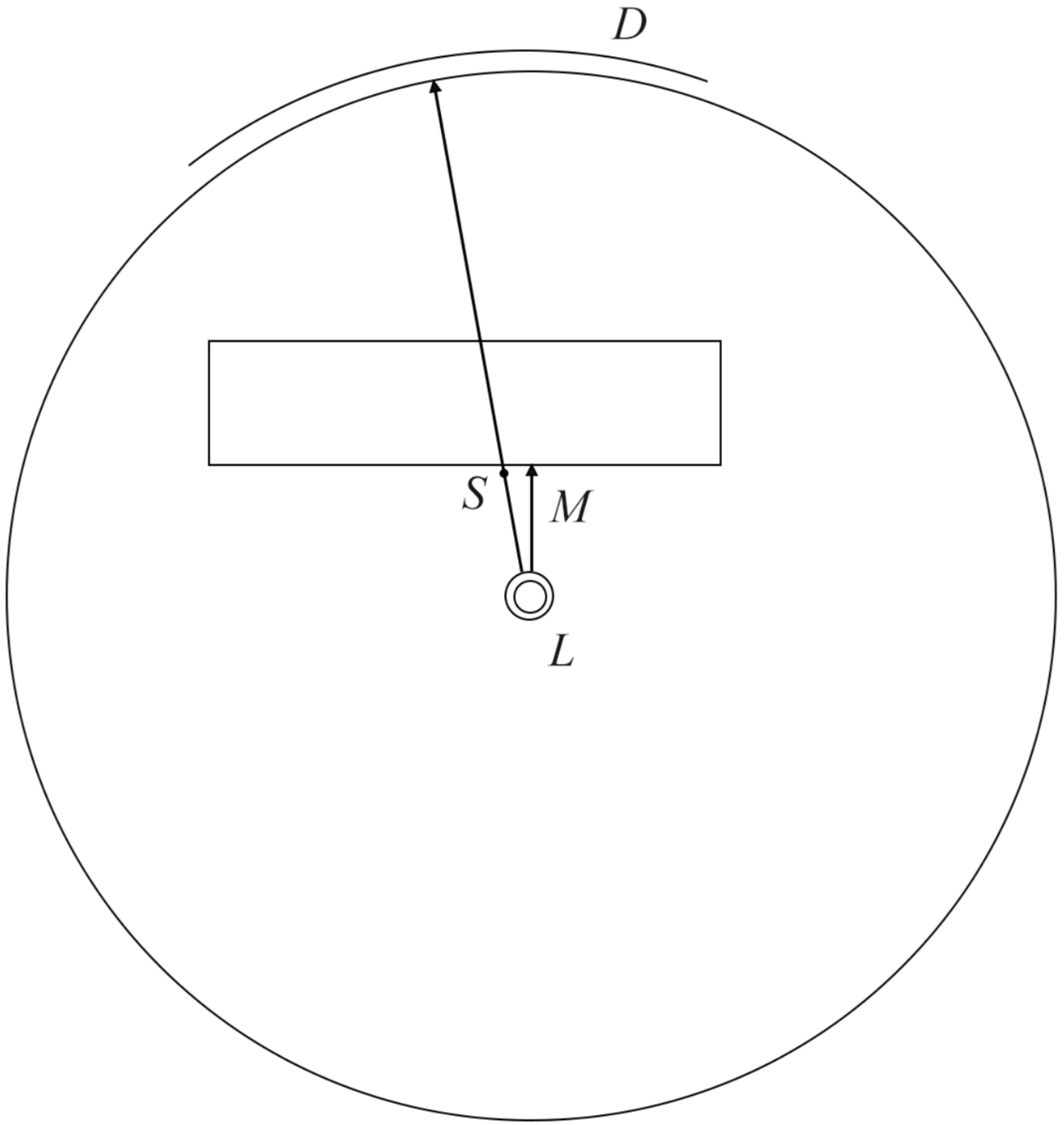
More Math!

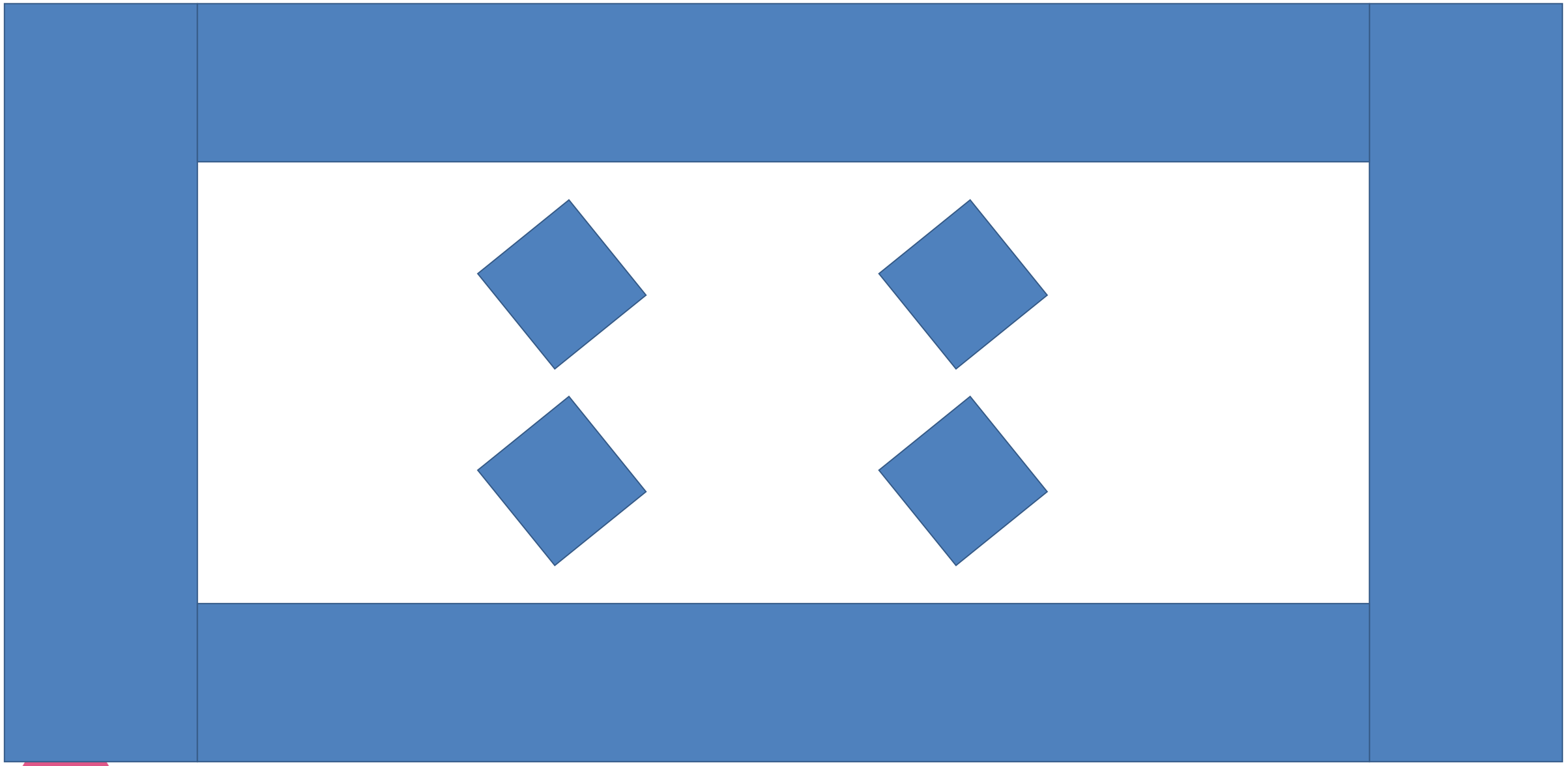
$$\hat{\sigma} = \lim_{\Delta s_{i \to 0}} \sum_{i=1}^{n} \frac{\hat{v}}{\|\hat{v}\|} W(\hat{v}) \Delta s_i = \int_C \frac{\hat{v}}{\|\hat{v}\|} W(\hat{v}) \Delta s$$

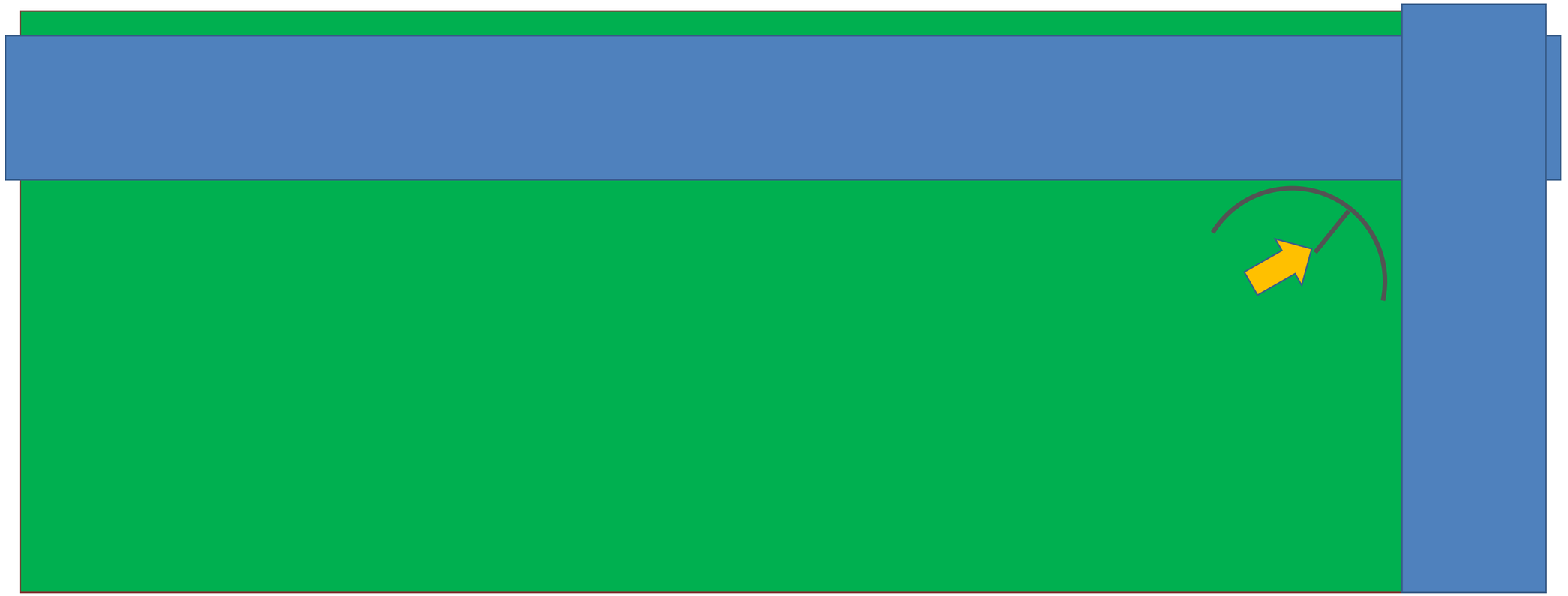
Becomes:

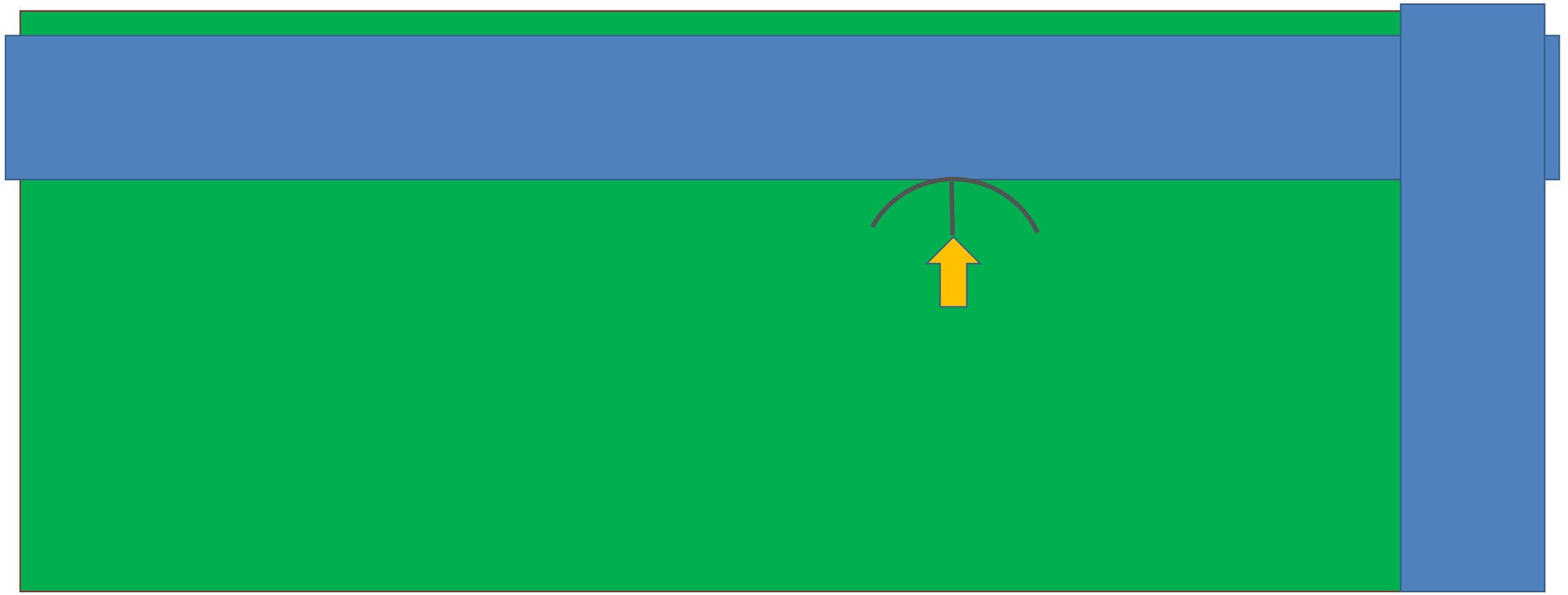
$$\hat{\sigma} = \iint_R \frac{\langle x, y \rangle}{\sqrt{x^2 + y^2}} \left(1 - \sqrt{x^2 + y^2}\right) \Delta x \Delta y$$

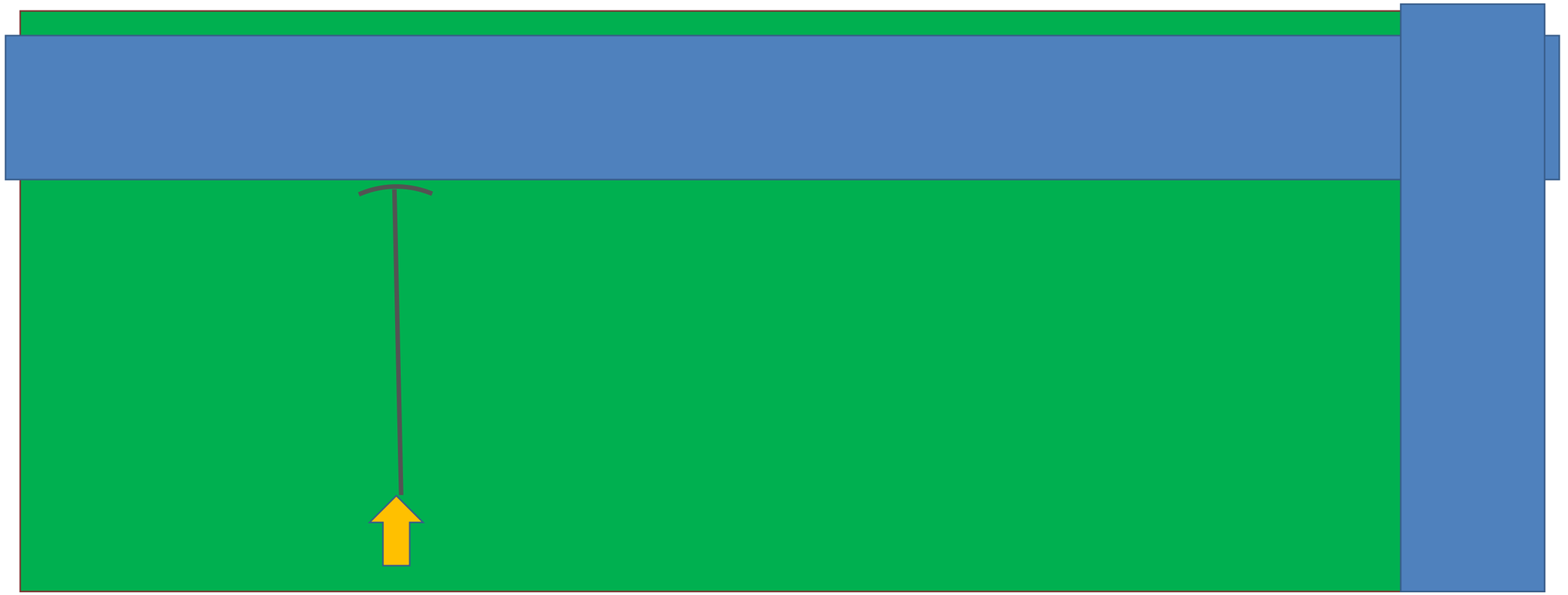
Image Credit: Nic Taylor





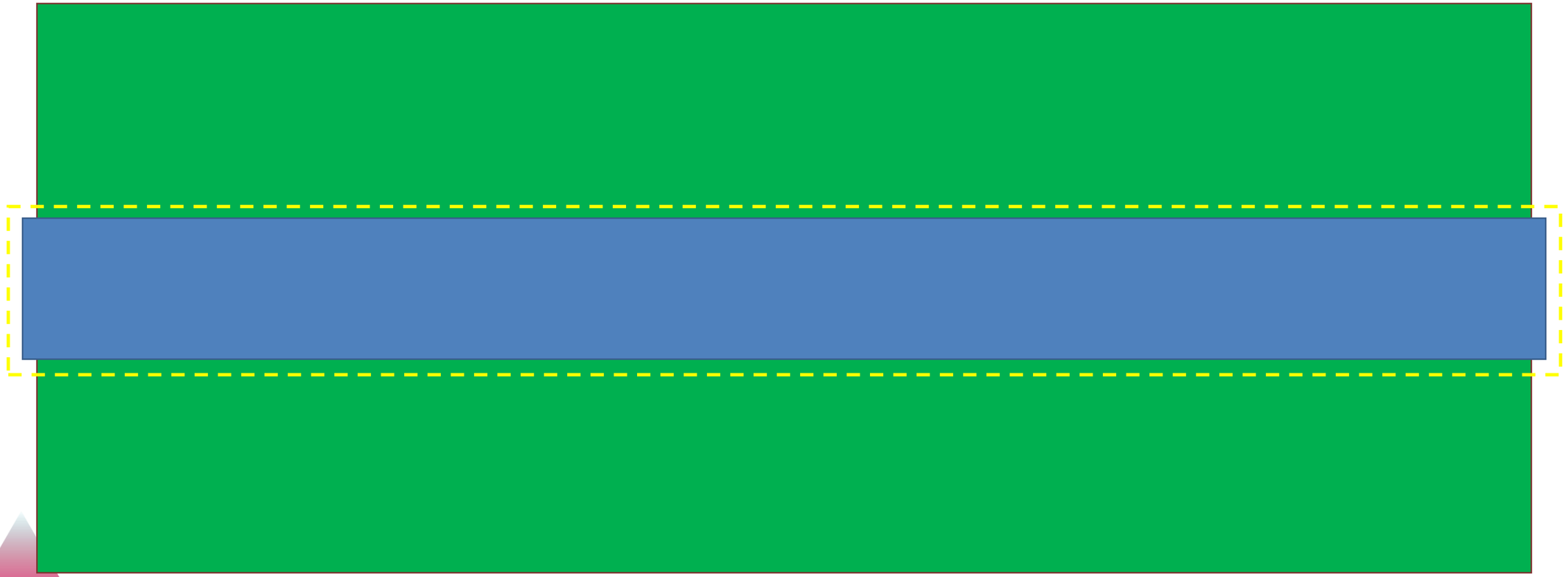






Details: Setup

- Setup is easy:



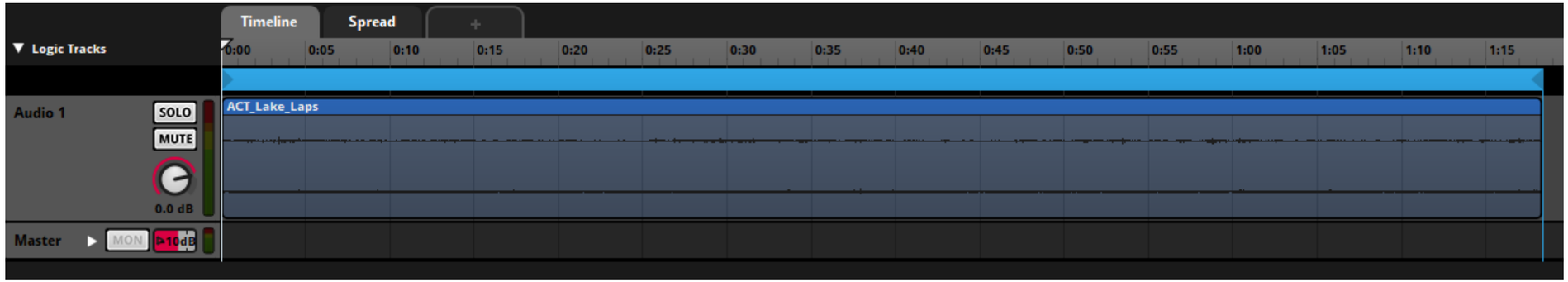
Details: Box Subdivision

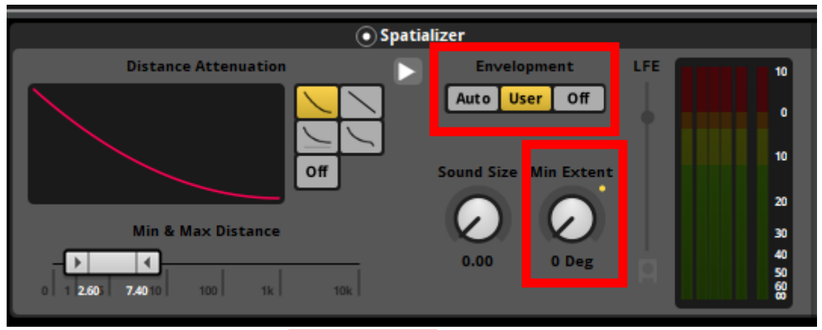
- Two solutions:
  - Subdivide boxes down
  - Build up the shape using voxels
- Both are covered in Game Audio Programming 2
- TL3 subdivides
  - (Minimum axis length – 0.5 meters)

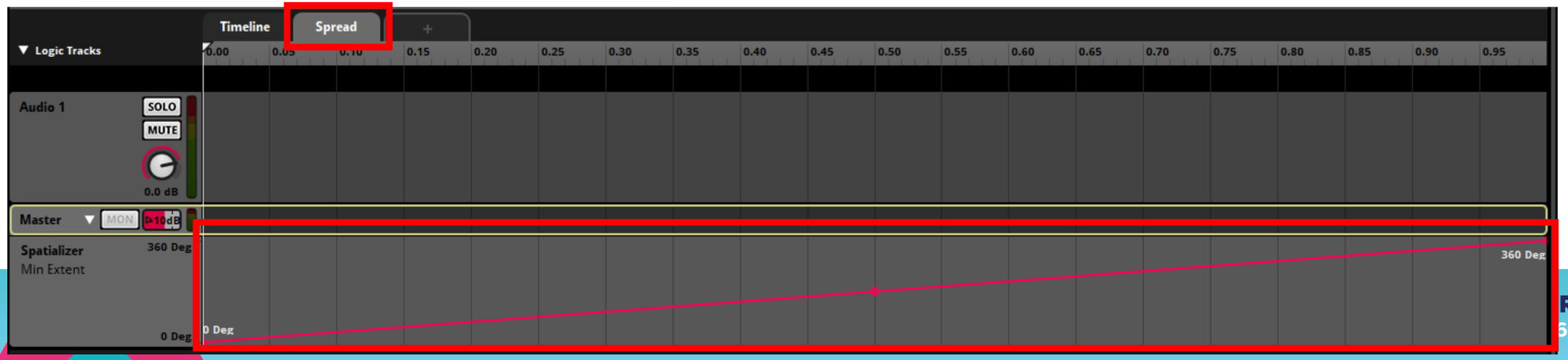
Details: Edge Cases

- Near field
  - As you get near to the volume, the sound needs to transition to full spread
- Popping
  - Add a seek speed to the direction and spread so that sudden changes don’t pop
- Close to zero
  - Certain edge cases are close to zero, and can cause strange values to pop up.
    - TL3 still has a few of these unsolved

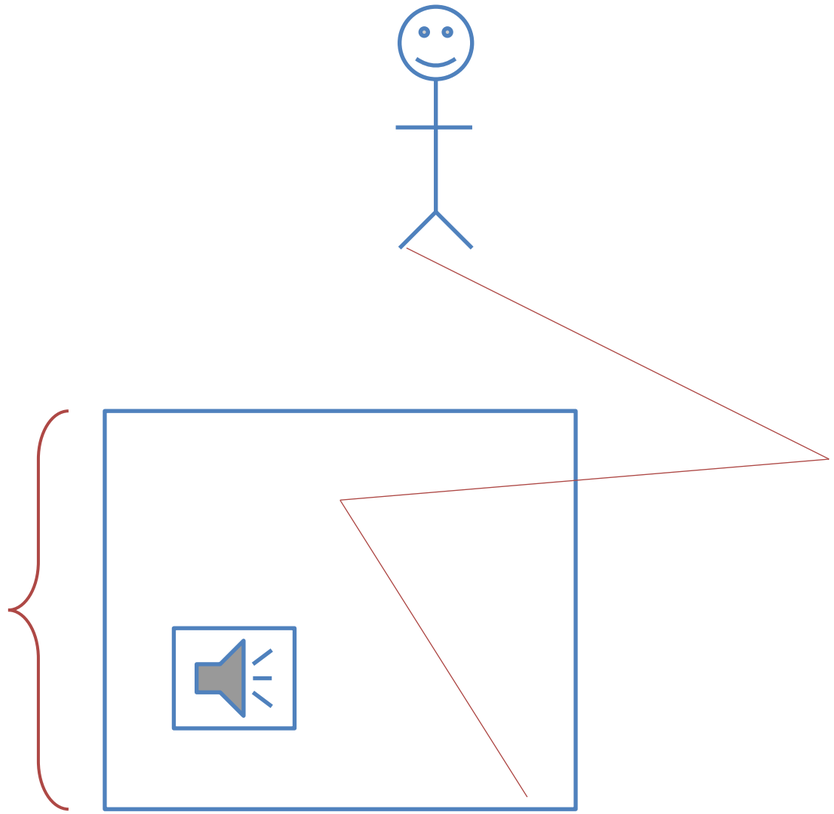
Details: FMOD Studio Setup







Details: Elevation



Details: Debug Display

- Show as much detail as possible:
  - Subdivided Boxes
  - Direction
  - Spread arc

Volumetric Sounds in Action

- Live Demo

Today’s Topics

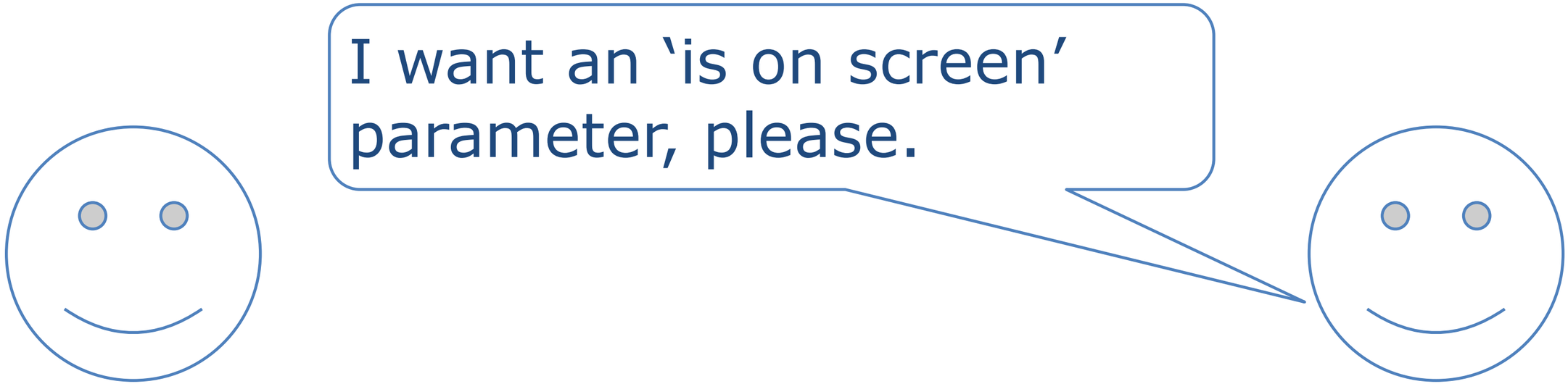
- Importance-Based Mixing
- Volumetric Sounds
- Screen-Space Distance Attenuation

A True Story

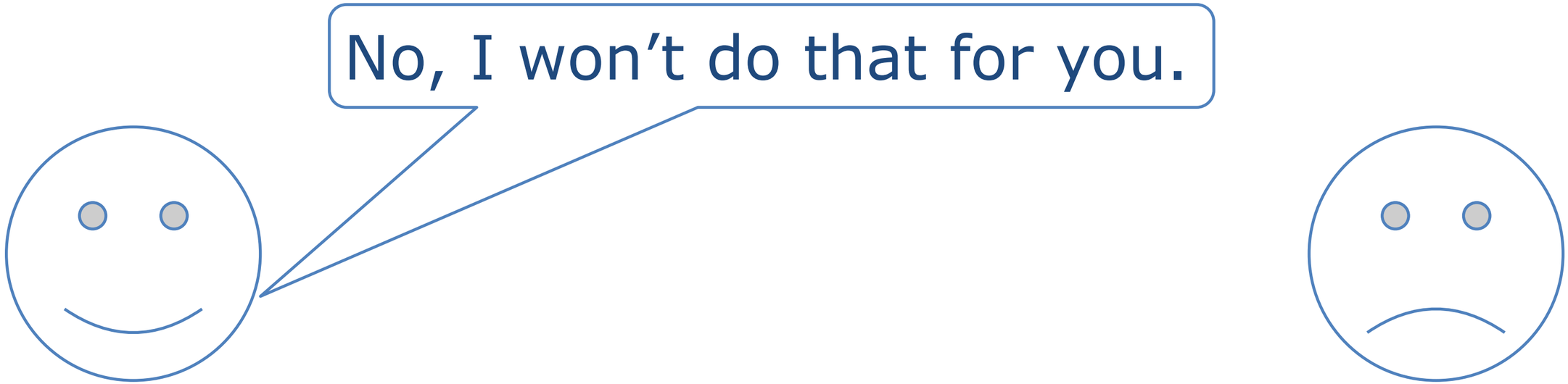
Me

Sound Designer

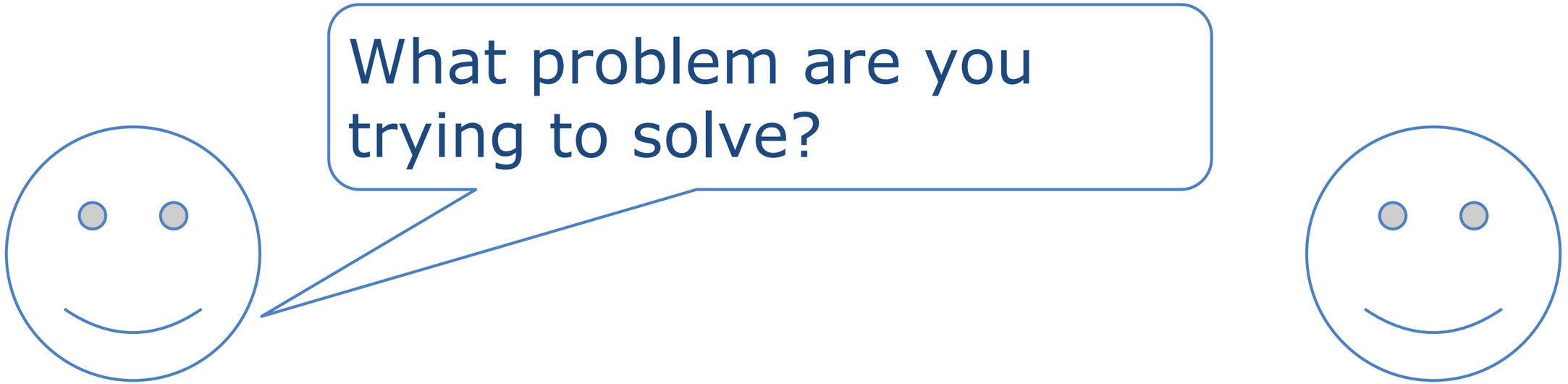
A True Story



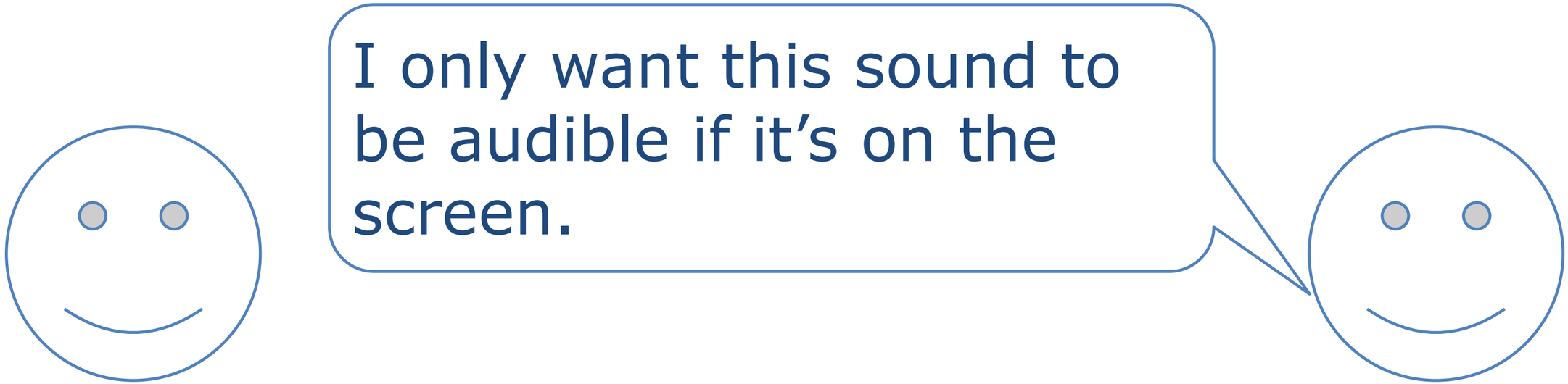
A True Story



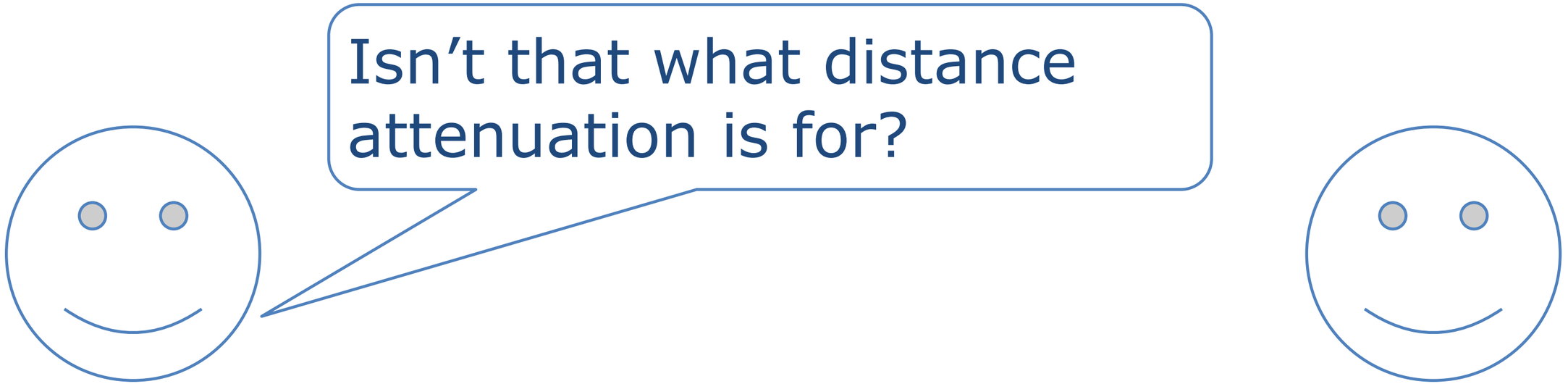
A True Story



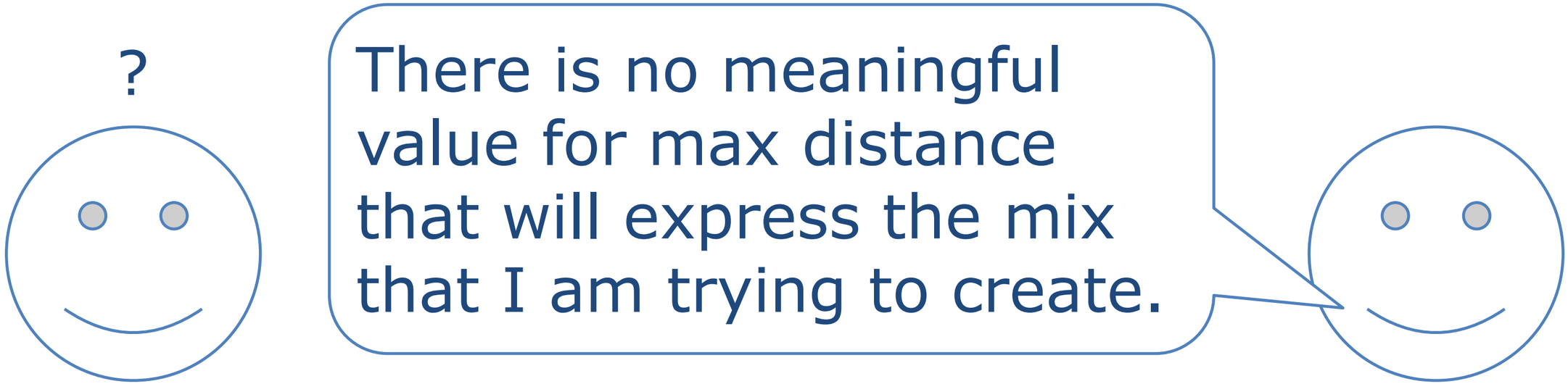
A True Story


I only want this sound to be audible if it’s on the screen.
Me
Sound Designer

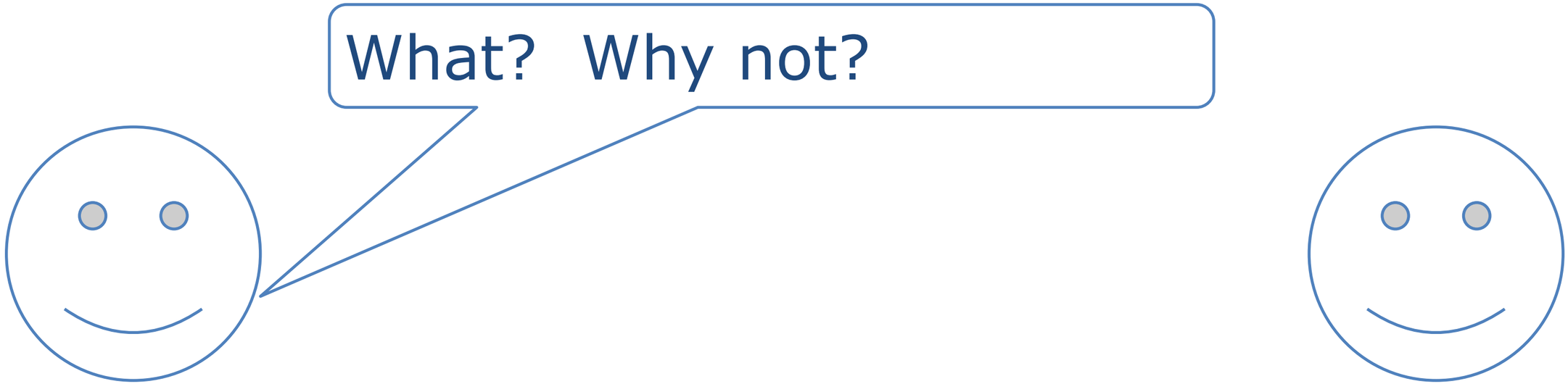
A True Story


Isn’t that what distance attenuation is for?
Me
Sound Designer

A True Story

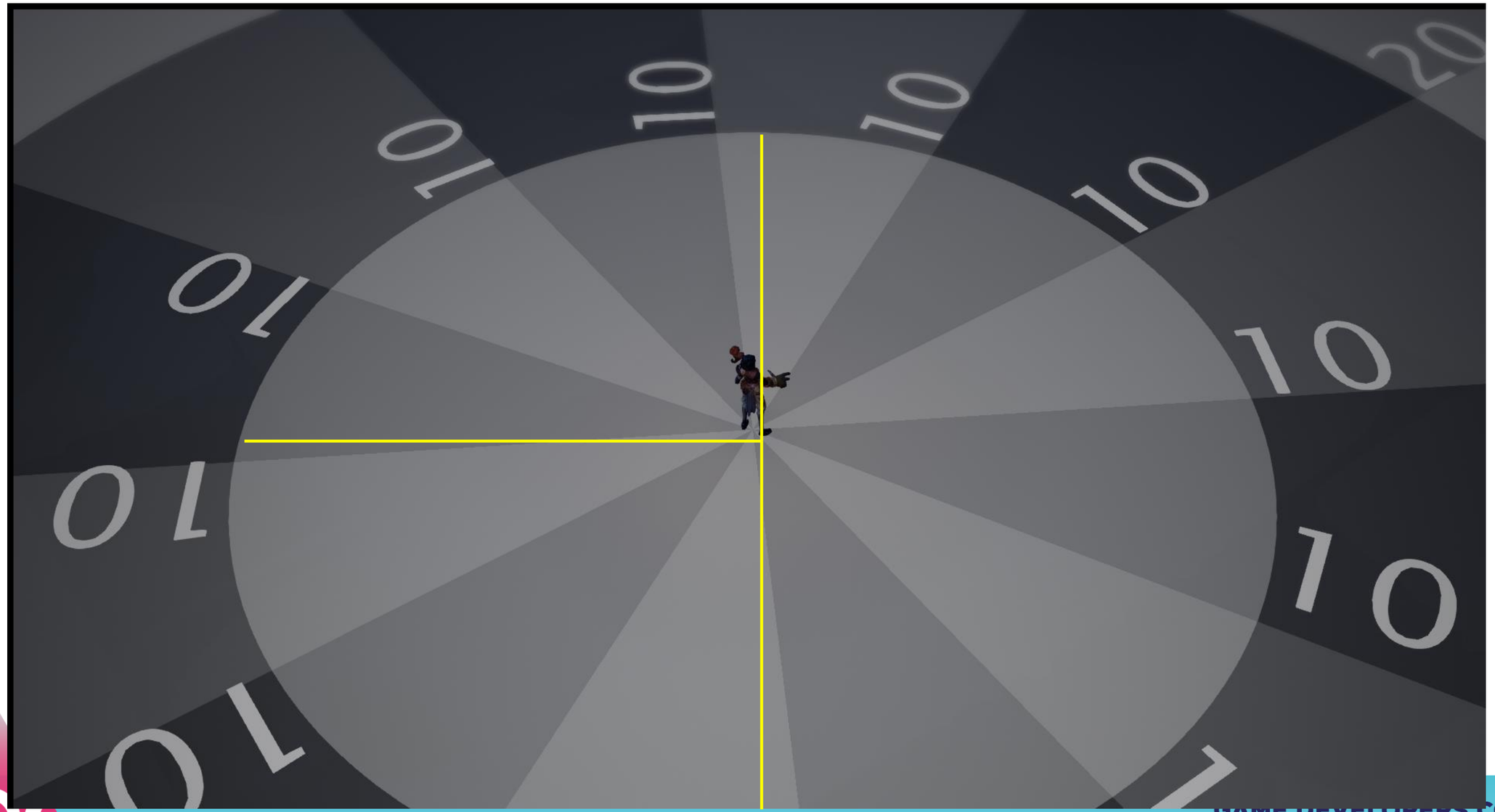

?
There is no meaningful value for max distance that will express the mix that I am trying to create.
Me
Sound Designer

A True Story

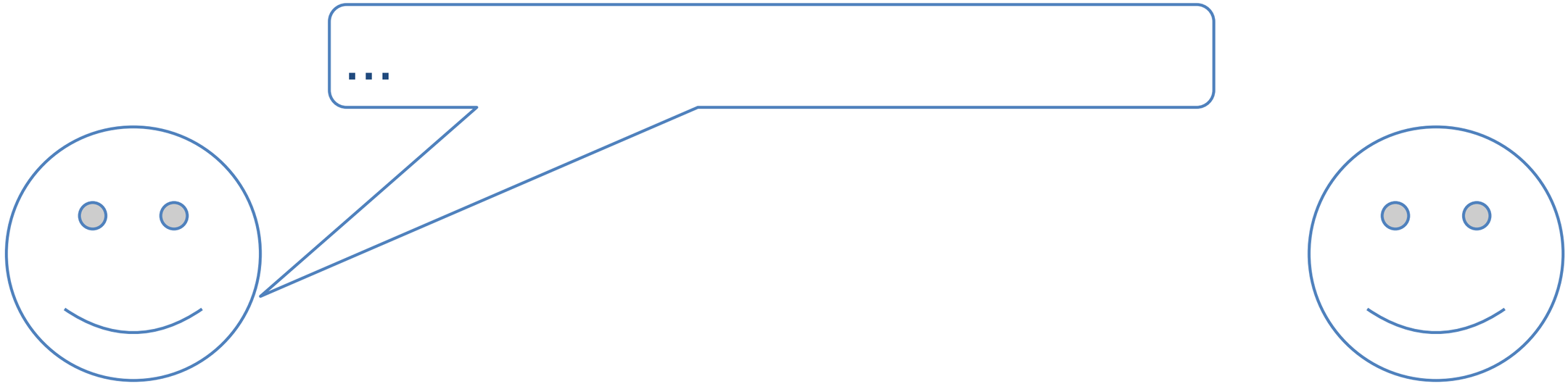


The Problem

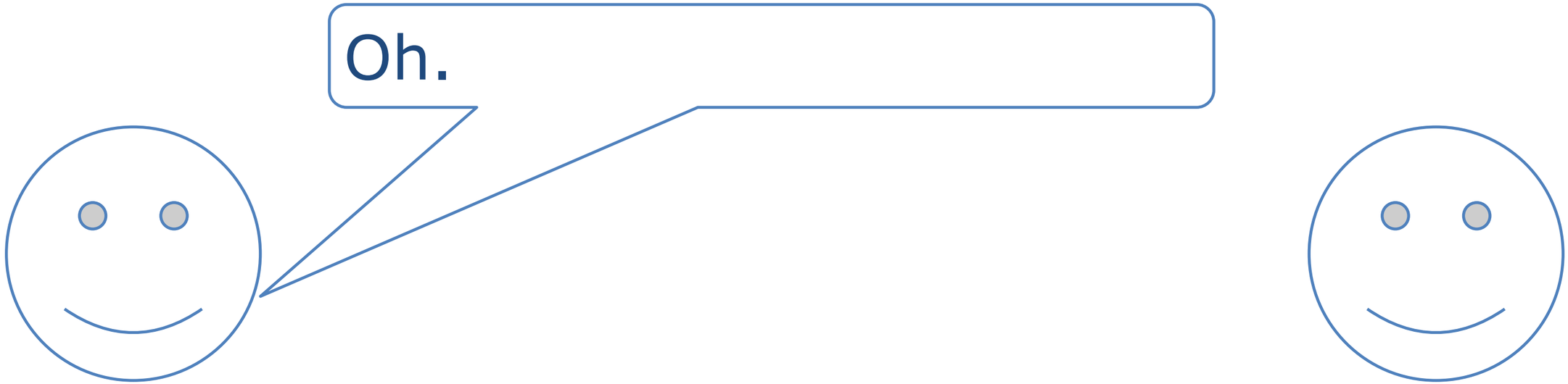
The Problem



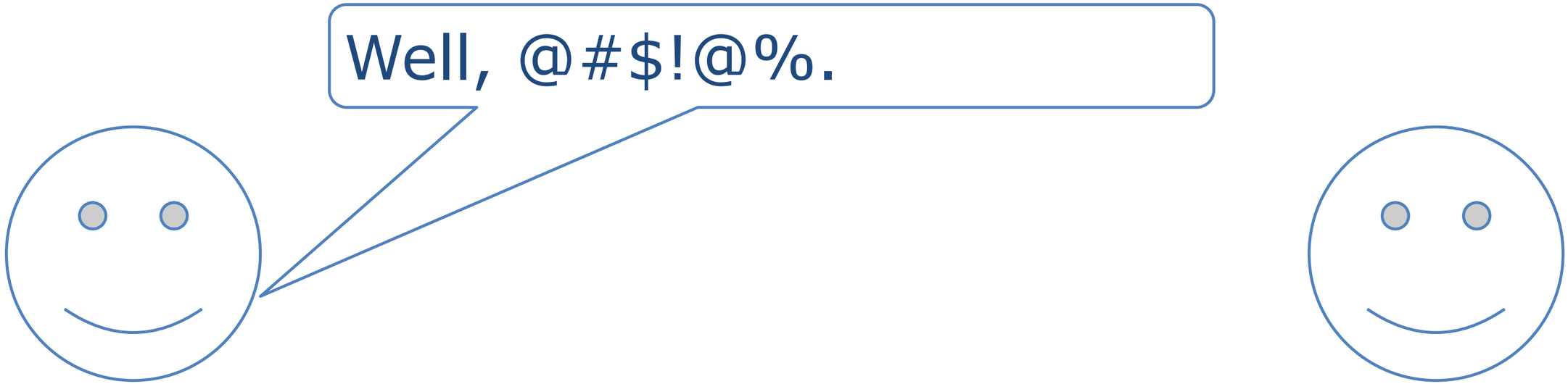
A True Story



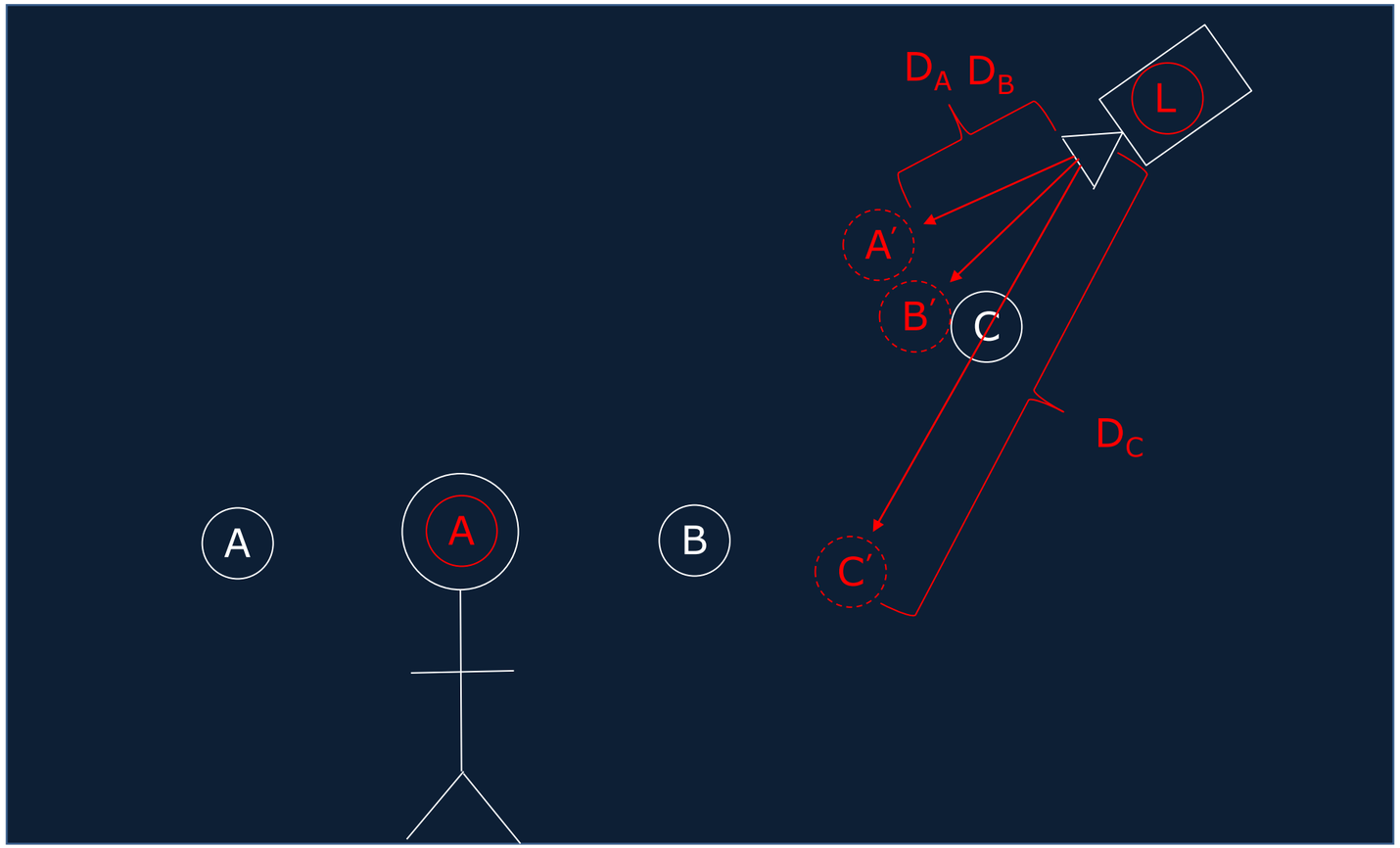
A True Story



A True Story



This is a given:



Original Listener Algorithm

- For each Channel
  - Calculate distance (d) from Channel Position to Attenuation Position
  - Find normalized vector ($\vec{N}$) from Listener Position (L) to Channel position
  - Place actual playing Channel at $L + d\vec{N}$

Screen-Space Distance Attenuation

- Project points onto unclipped screen space.
- Make distance equal to the screen-space distance to attenuation position

Adjusted Listener Algorithm

- For each Channel
  - Project Channel Position and Attenuation Position to screen space, and calculate 2D distance (d)
  - Find normalized vector ($\vec{N}$) from Listener Position (L) to Channel position
  - Place actual playing Channel at $L + d\vec{N}$

Implementation Details

- Scale is challenging
- Zooming in shouldn’t affect attenuation
- Debug Display
- Every sound’s distance attenuation must be re-authored

Scale Issues

- By default, screen-space projections give you either 0..1 or -1..+1
- Sound designer tools don’t like to work in such small scales
  - Also: Sound designers don’t like to work in such small scales

- Multiply all positions/distances by an agreed-upon factor
  - Torchlight 3 uses a scale of 20

Camera Zoom

- We can’t just call `ProjectWorldToScreen()`

Camera Zoom

- We can’t just call `ProjectWorldToScreen()`

Camera Zoom

- Project to where the camera would be if it were fully zoomed-out
- Need a function that will do the projection at a different location than the camera
  - (This is particularly useful for debug info)

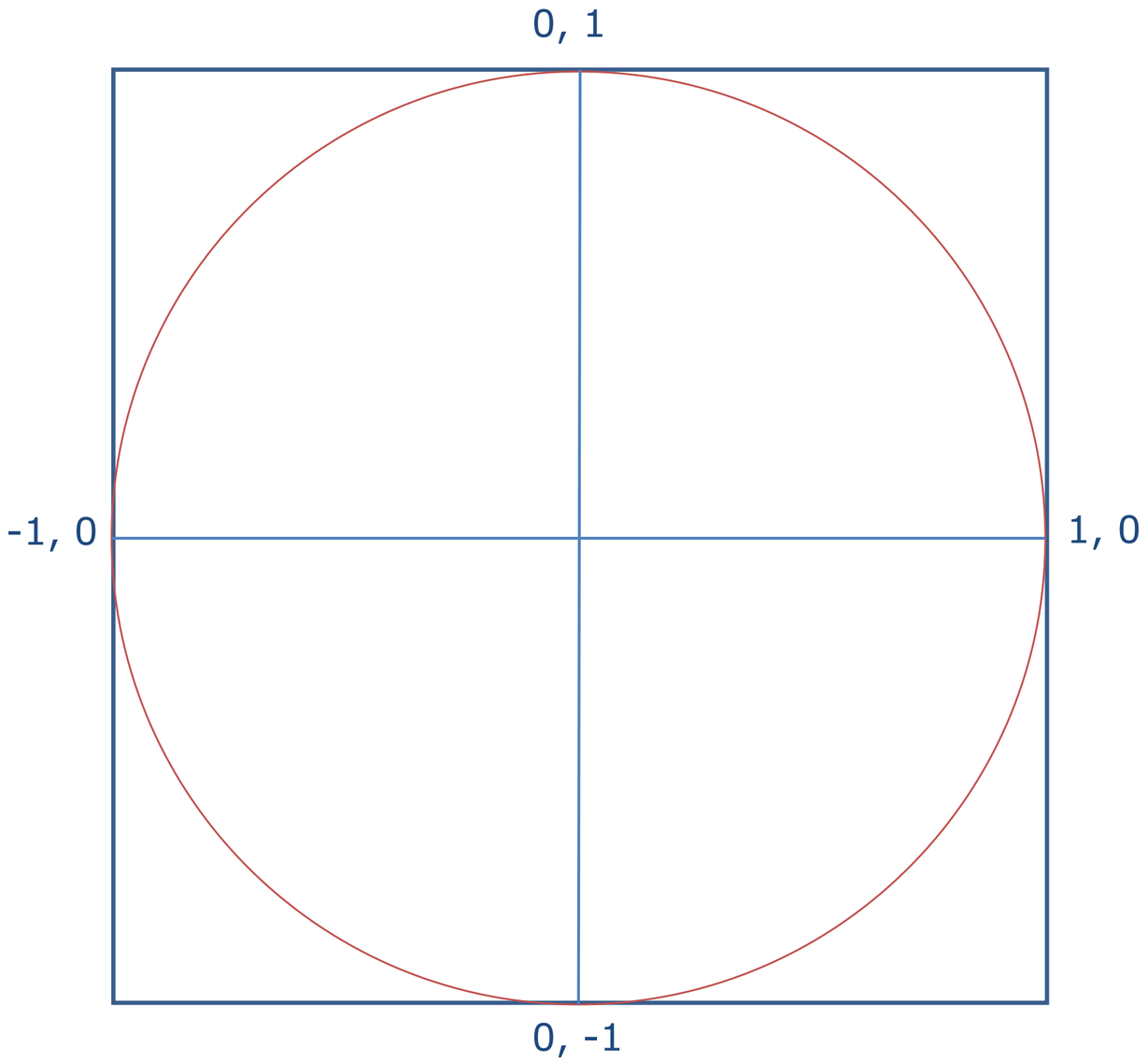
Debug Display

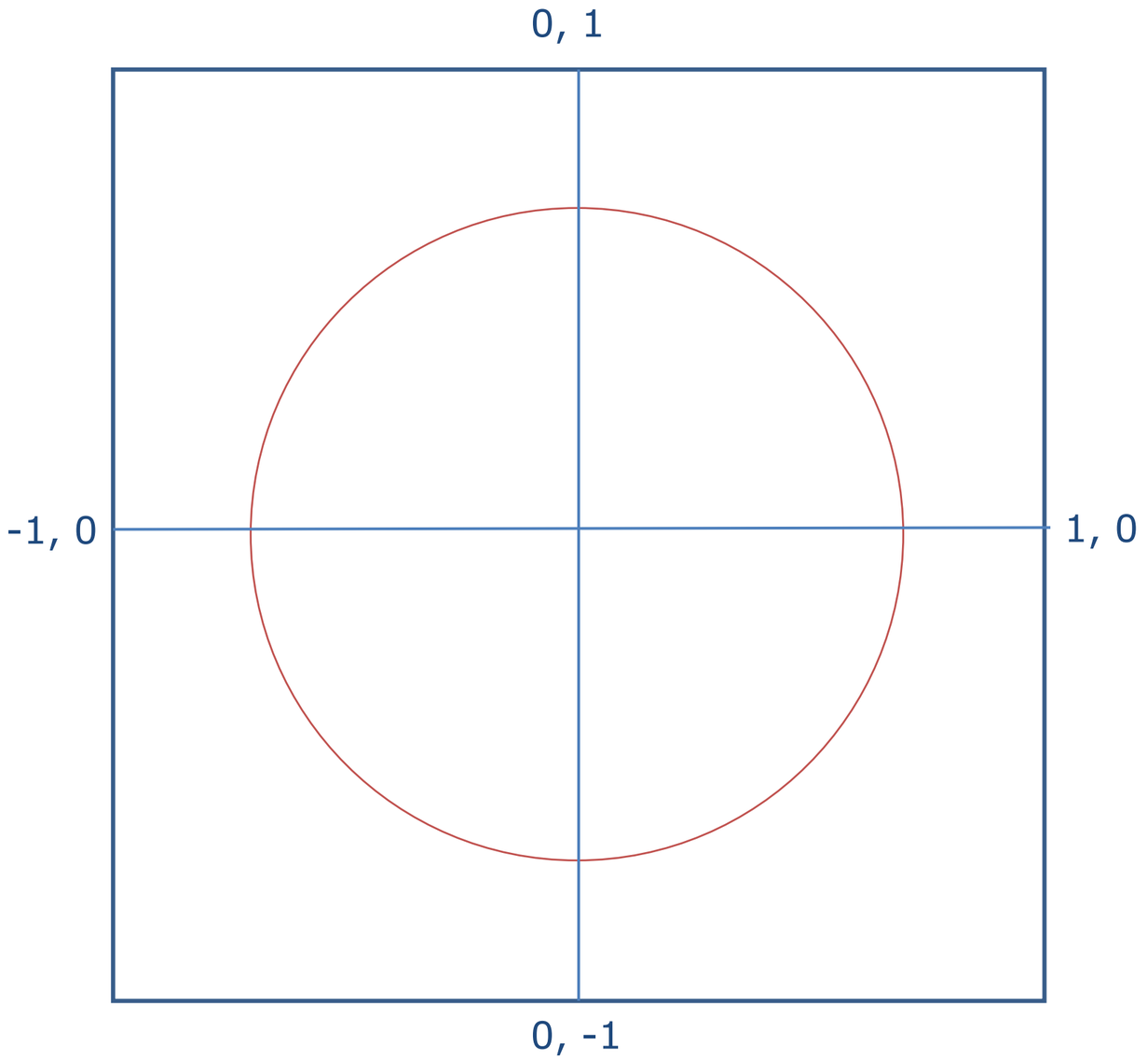
- Traditionally, we display a sphere of the appropriate radius.
- Not anymore: distance attenuation isn’t shaped like a sphere (or a circle) anymore
- How do we help the sound designers understand where in world space the sound will be audible?

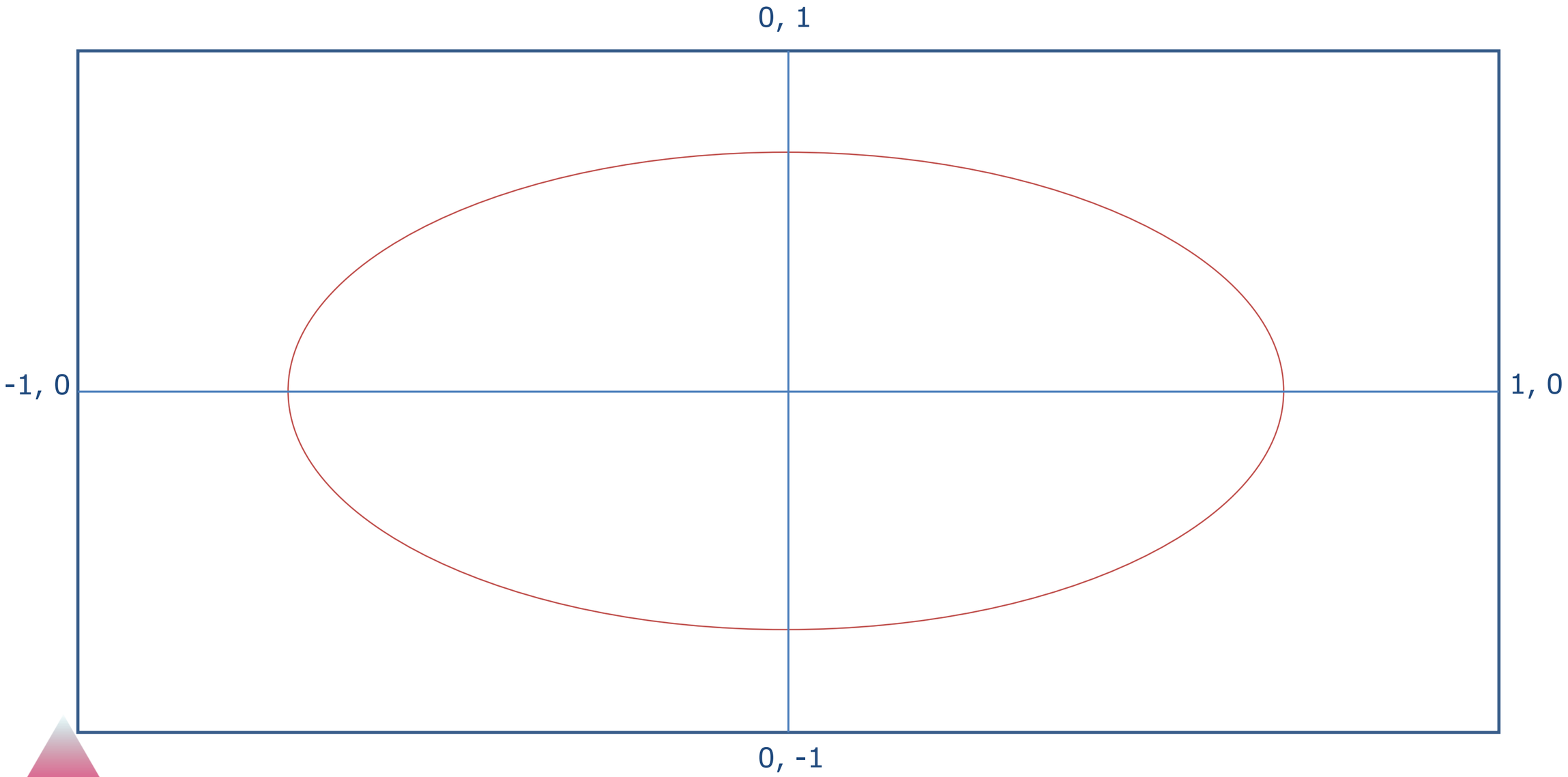
  - We need to invert the process of projection

Algorithm:

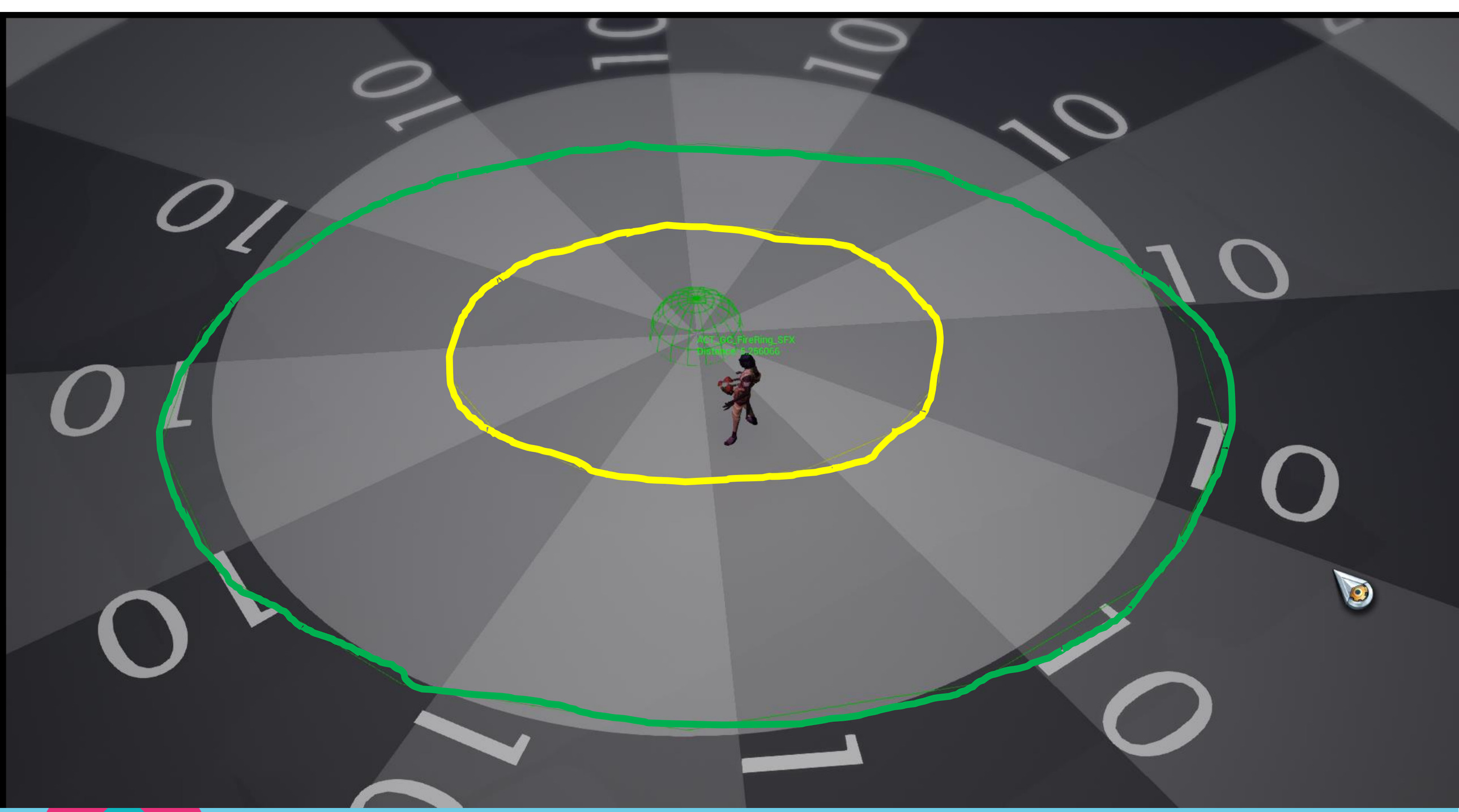
- Take the vector (S_x, S_y, A_z) and project onto screen space
- Make a circle in screen-space at a radius of Distance/20 (or whatever your screen-space scale is), centered on the projected point
  - Convert to pixel coordinates if necessary
    - In Torchlight 3, we must first convert from -1..+1 to 0..1, then convert to pixel coordinates
  - Deproject screen to world position + ray
  - Intersect ray with the attenuation position's plane
    - Flip intersected X and Y around original sound location

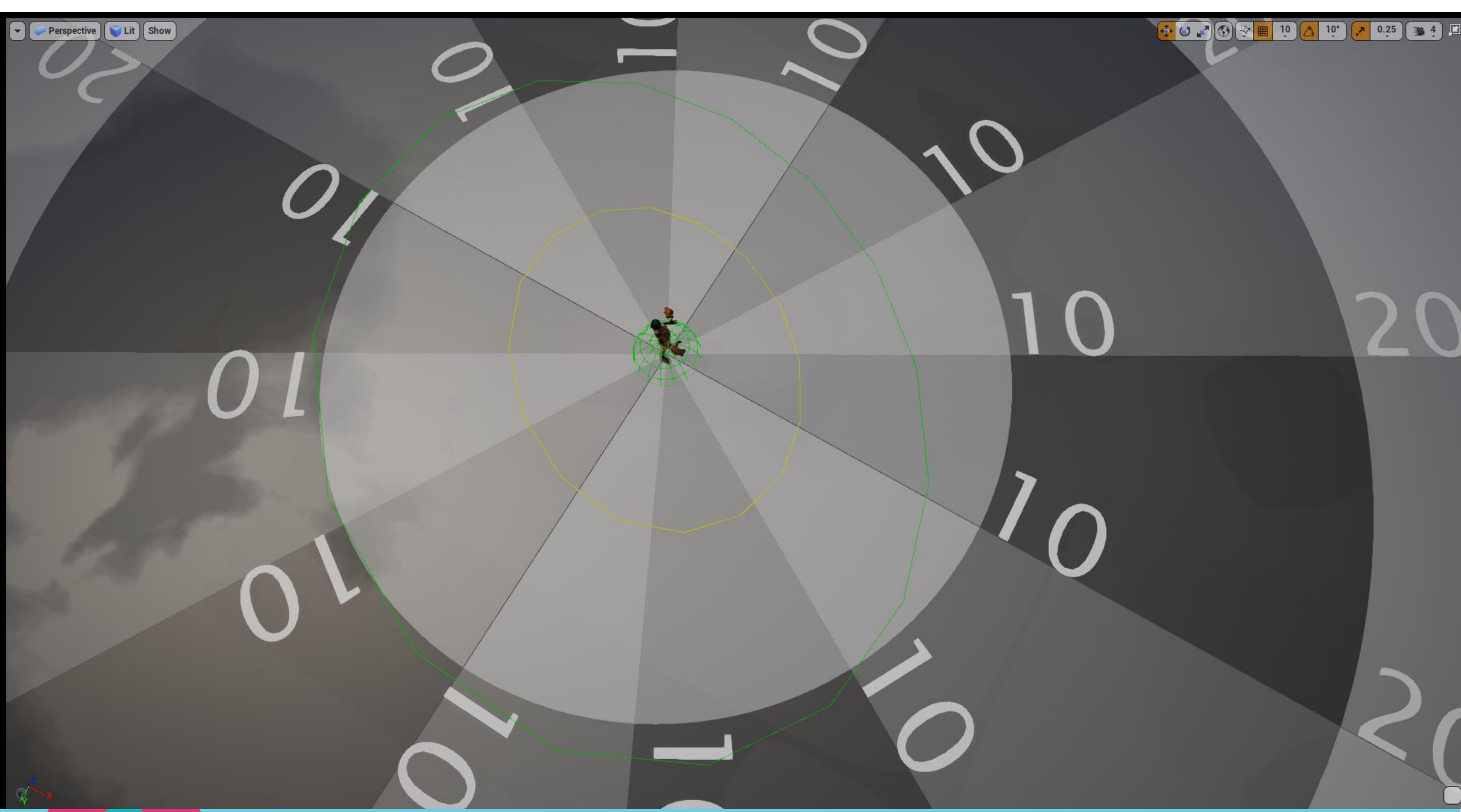

0, 1
-1, 0
1, 0
0, -1

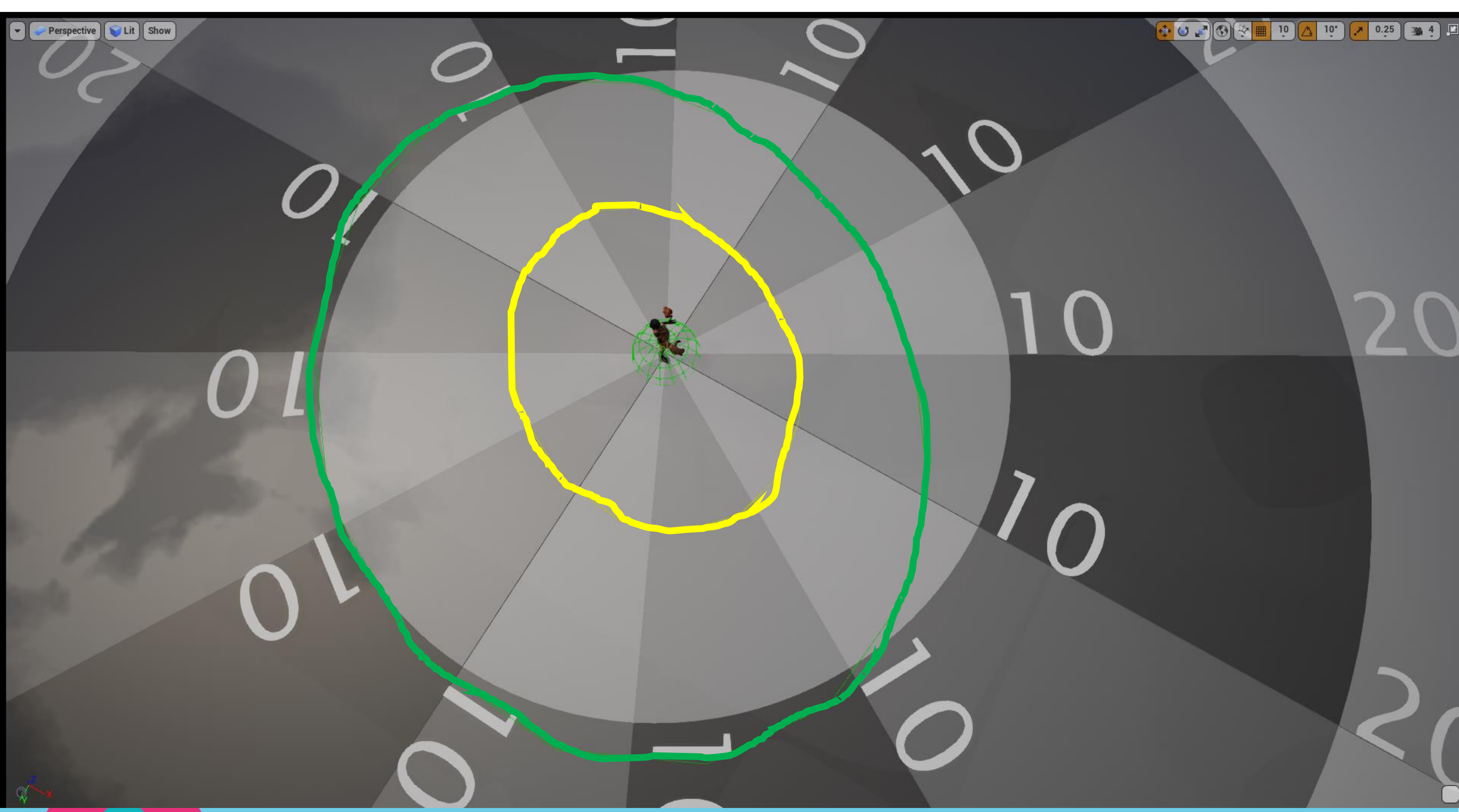

0, 1
-1, 0
1, 0
0, -1


0, 1
-1, 0
1, 0
0, -1


ACT_GC_FireRing_SFX
Distance: 5.256066

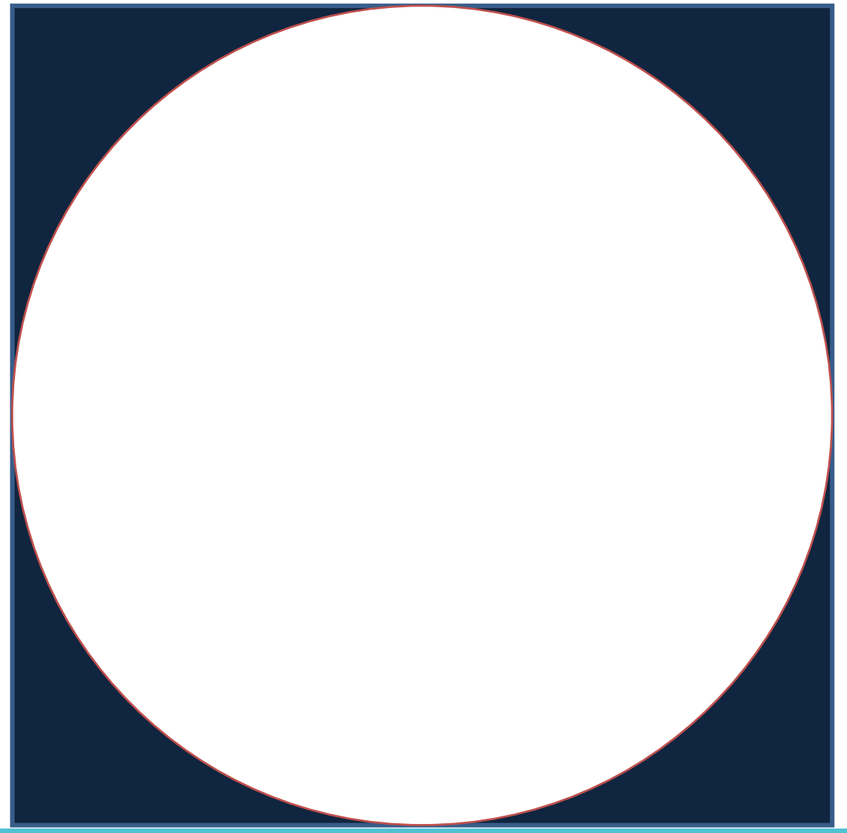

ACT_GC_FireRing_SFX

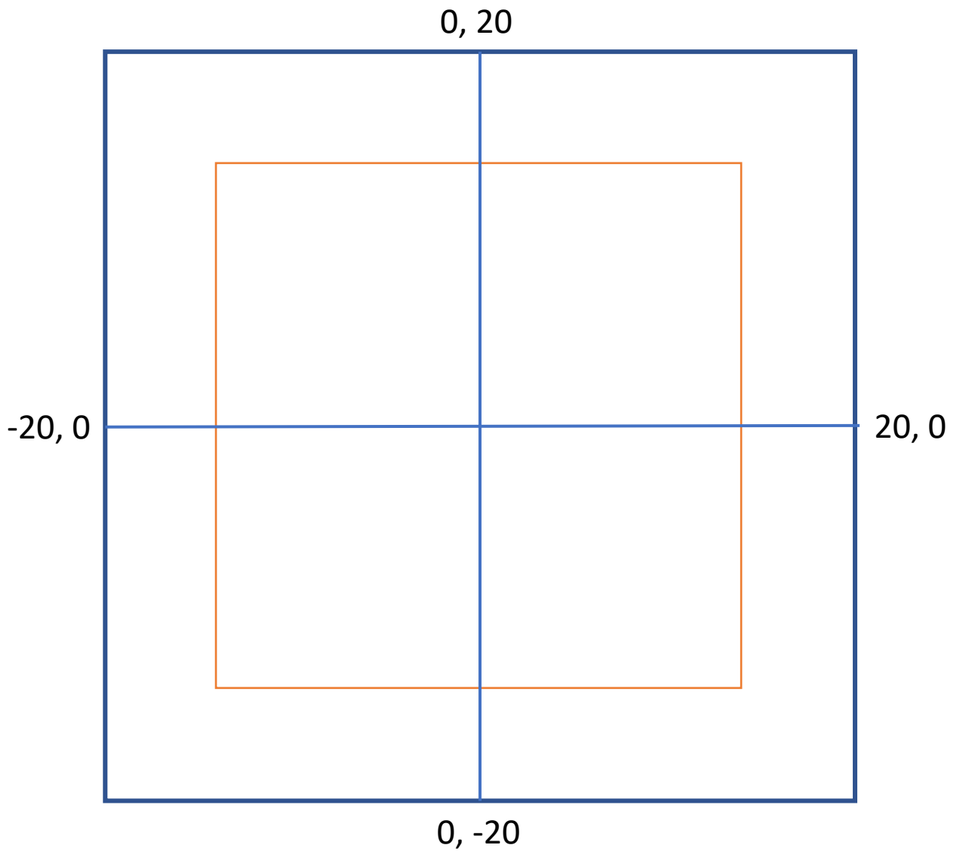

Perspective
Lit
Show
10
10°
0.25
4
10
20

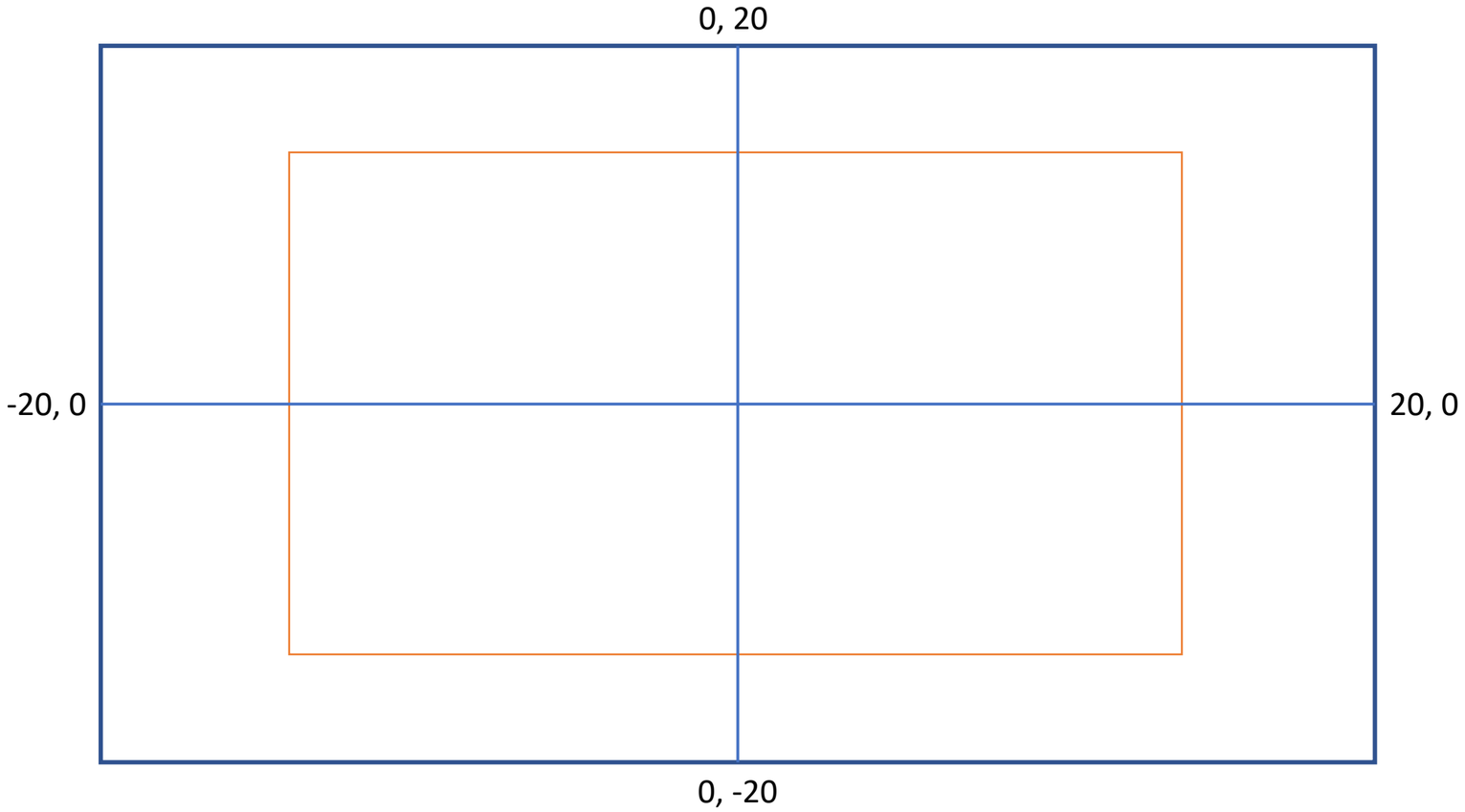

Perspective
Lit
Show
10
10°
0.25
4
10
20

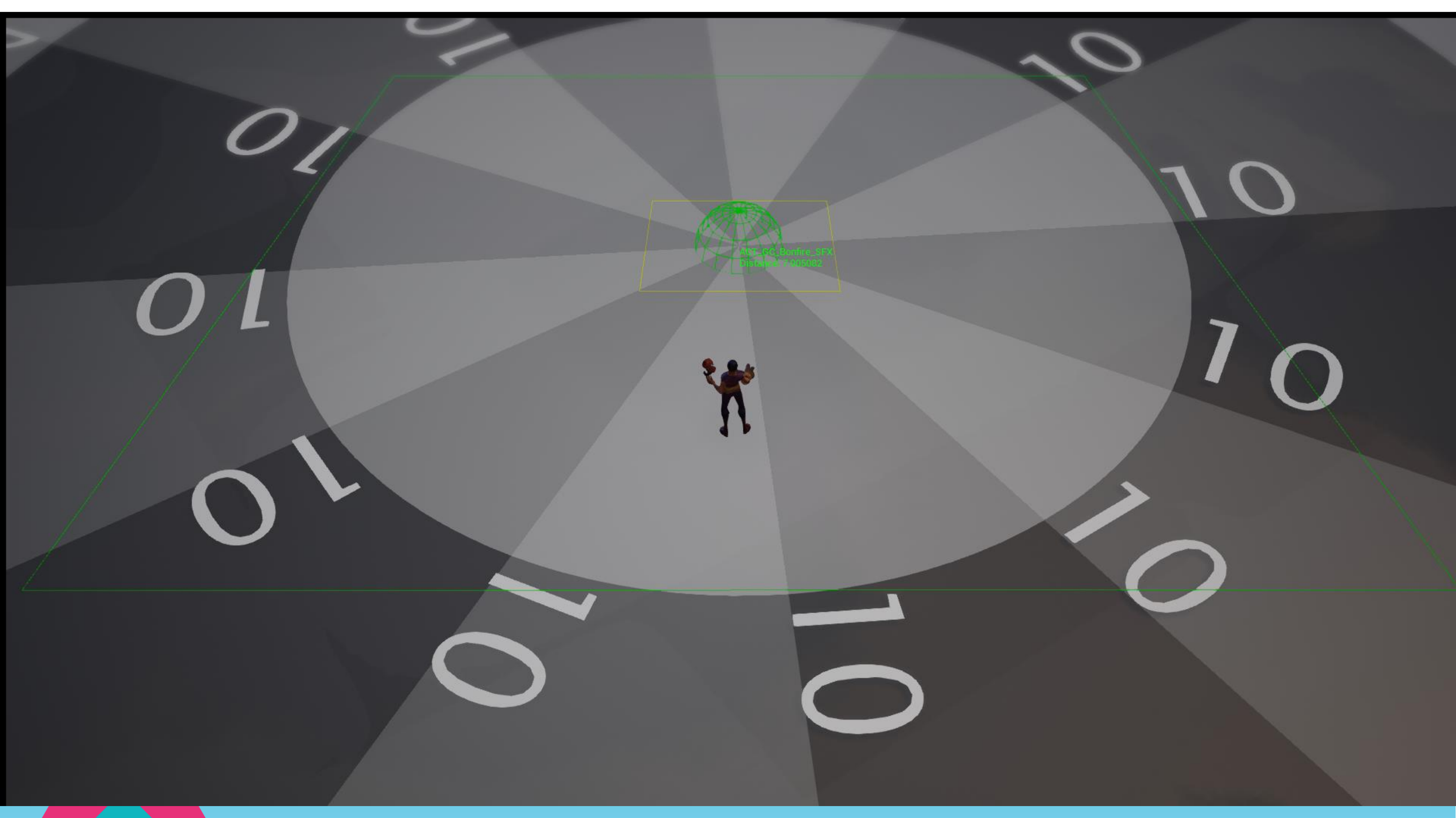
But wait…

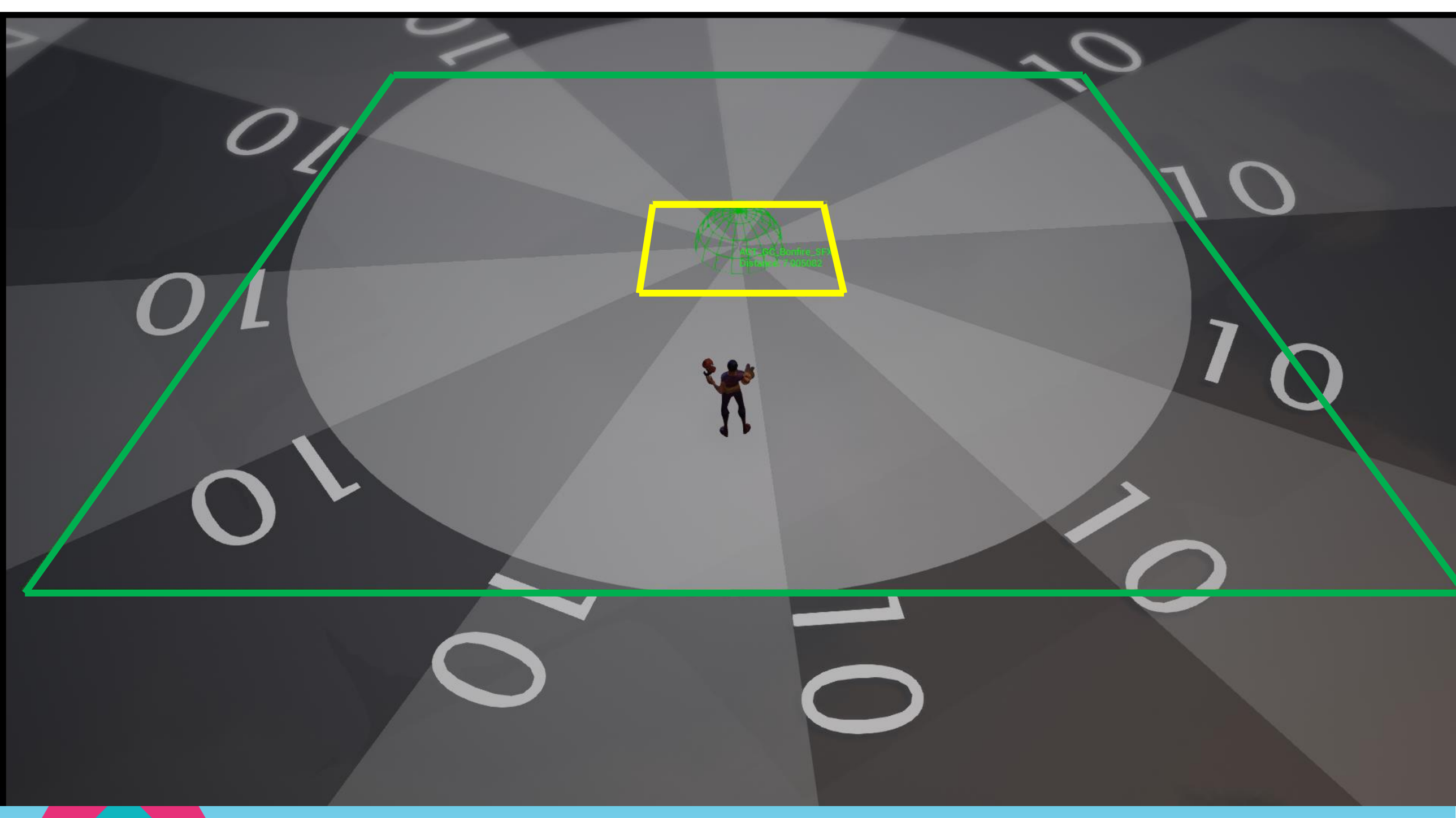
- We’ve been doing all of this work in screen-space
- What about the corners?

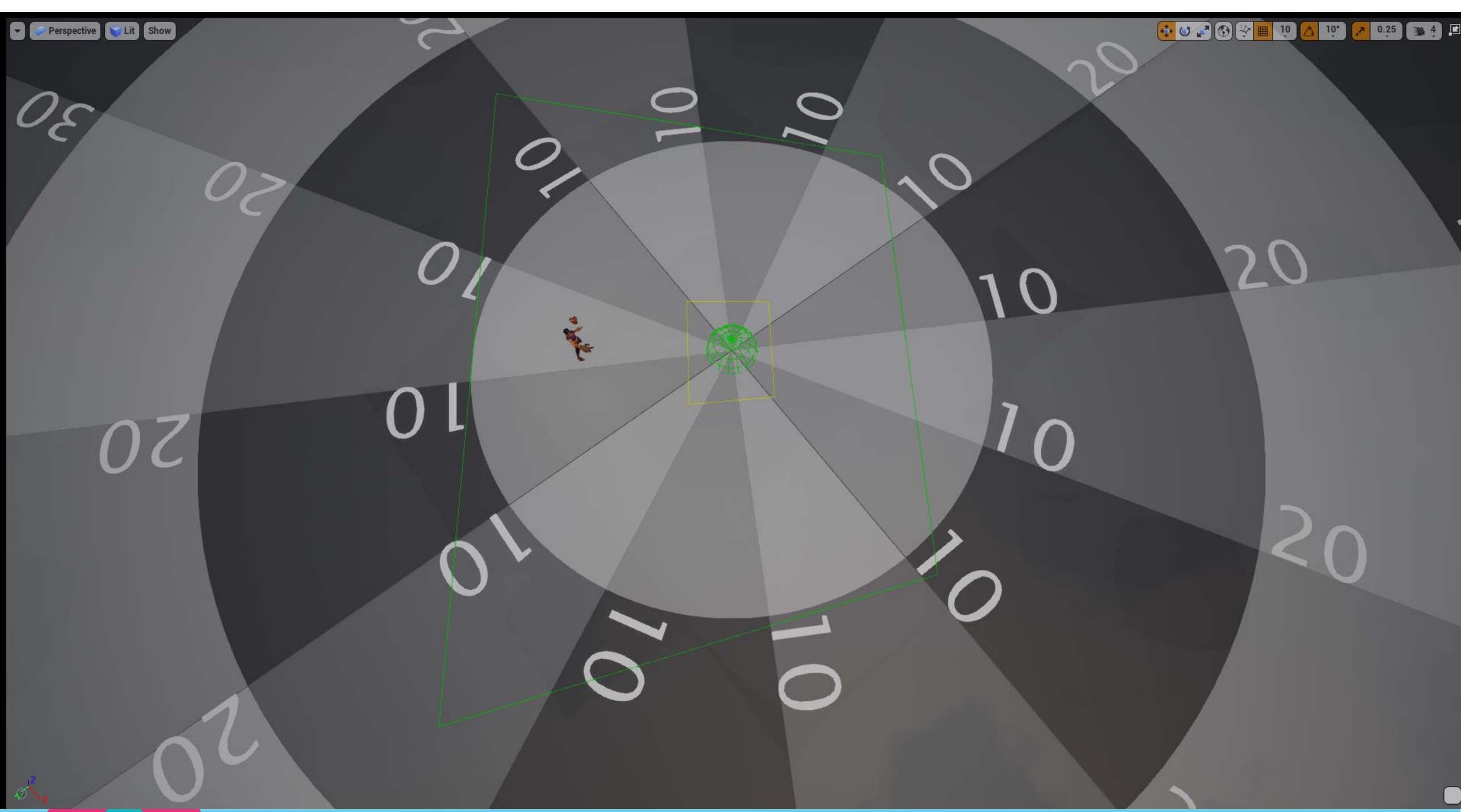


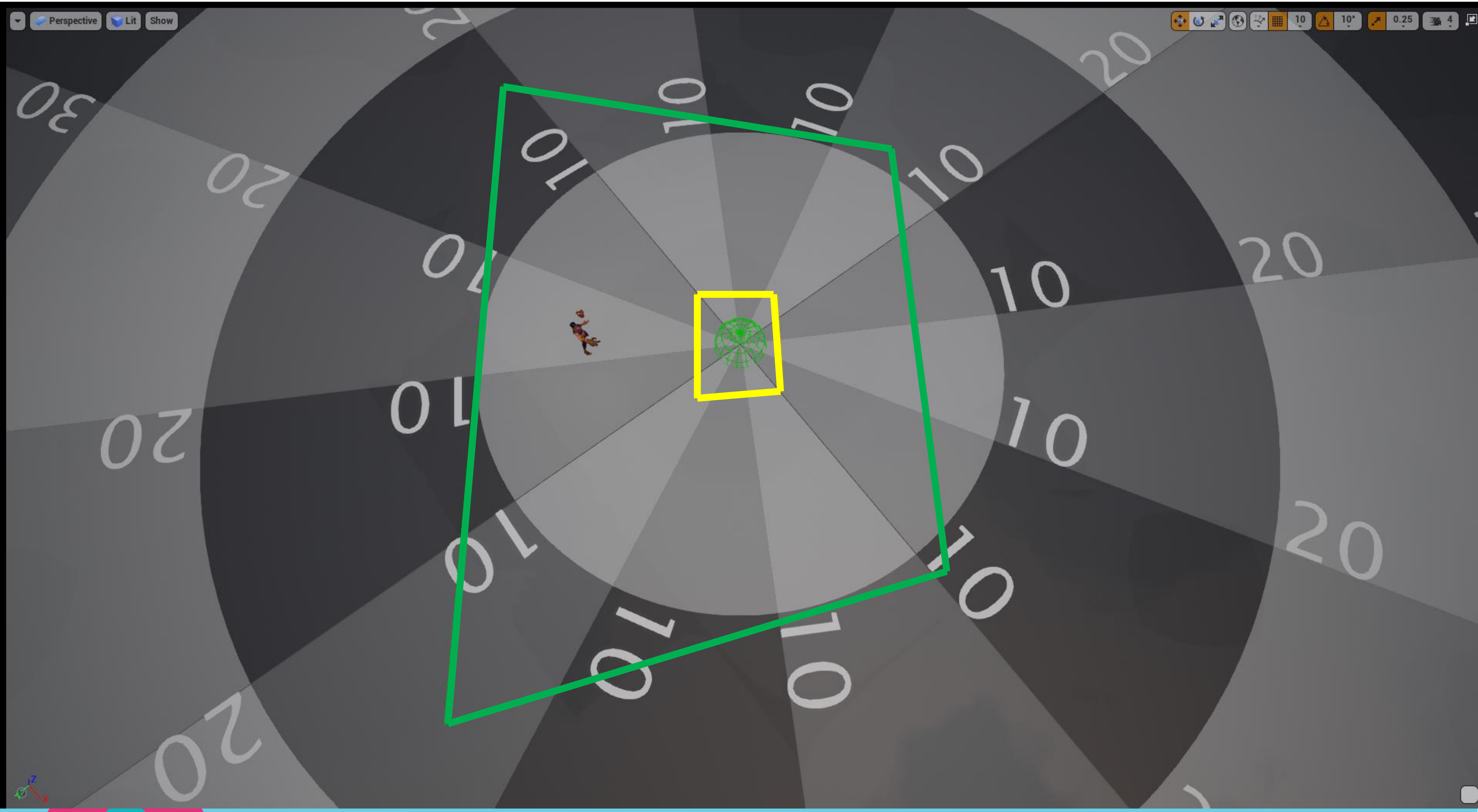

0, 20
-20, 0
20, 0
0, -20


0, 20
-20, 0
20, 0
0, -20


AGT_GC_Bonfire_SFX
Distance: 7.005082


AGT_GC_Bonfire_SFX
Distance: 7.005082


Perspective
Lit
Show
10
10°
0.25
4
10
20
30


Perspective
Lit
Show
10
10°
0.25
4

Content Tendrils

- We have replaced a 3D world-space value with a 2D screen-space value
- Distance no longer means the same thing that it used to

- Every 3D sound event must be reauthored to account for this new idea of distance
  - Every distance calculation must make a decision whether to use 3D or 2D distance

Screen-Space Distance Attenuation in Action

- Live Demo

Conclusion

- ARPGs have distinctive (but not unique) challenges

- Importance-Based Mixing is a fundamental feature
  - Part of every audio engine from day 1
- Volumetric Sounds solve the river problem elegantly
  - You don’t necessarily have to understand the math
- Screen-space distance attenuation is also a fundamental feature
    - (If you have a fixed-camera game like an ARPG or RTS)

Conclusion

- Complex features are complex
- Debugging and visualization routines are *critical*
- Good visualization can be hard, but it’s always worthwhile

Questions

- Comments
- Compliments
- Complaints
- Queries
- Inquiries
- Inquests
- Observations
- Opinions

- Remarks
- Commendations
- Objections
- Impressions
- Thoughts
- Commentary
- Assertions
  - (Just kidding! Everybody knows that game devs ignore asserts.)