Geometric Sound Propagation

Micah Taylor

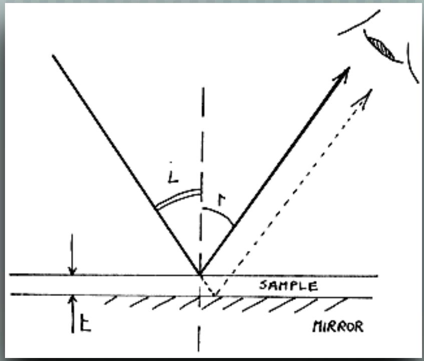
Sound propagation

- Given a sound source in a scene, what does a listener hear?
  - Source emits waves
  - Travel out, interacting with the surroundings
  - Some waves arrive at the listener

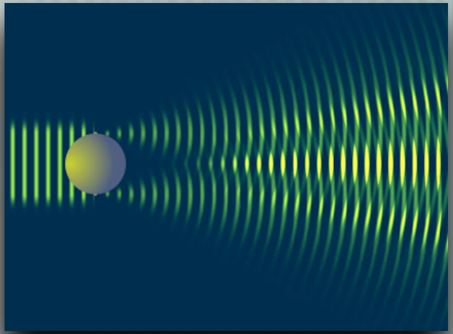
Sound propagation

- Sound travels slow
  - 344 m/s

- Specular reflections
  - Perfect reflection

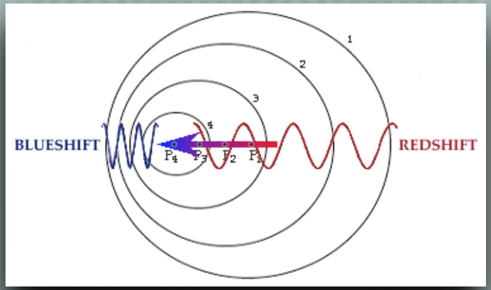


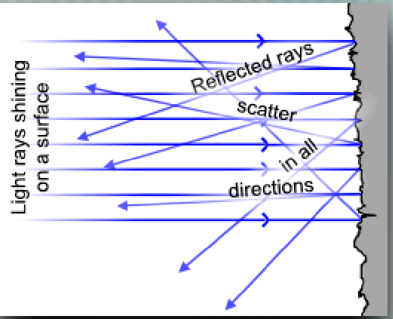
- Diffraction
  - Sounds ‘bends’ around corners



Sound propagation

- Doppler effect
  - Change in frequency due to motion
- Diffuse reflection
  - Surfaces can scatter reflection
- Wave properties
  - Interference





Adorable and Cute

Adorable and Cute

Horrible and Ugly

$$\frac{\partial^2 p}{\partial x^2} - \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} = 0$$

fig 1. Acoustic wave equation. Looks simple, but is full of poison and very sharp pointy teeth

Horrible and Ugly

$$\frac{\partial^2 p}{\partial x^2} - \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} = 0$$

fig 1. Acoustic wave equation. Looks simple, but is full of poison and very sharp pointy teeth

Numerical methods [4]

- Finite Element Method / Boundary Element Method
  - Divide space into elements
  - Solved with discrete linear equations
  - Model wave equation well
  - **Extremely** computationally intensive

Geometric methods [4]

- Can be very fast
  - Heavily explored field
  - Many optimization techniques
- Not entirely physically accurate
  - Ignores some wavelength properties
  - Some effects are expensive

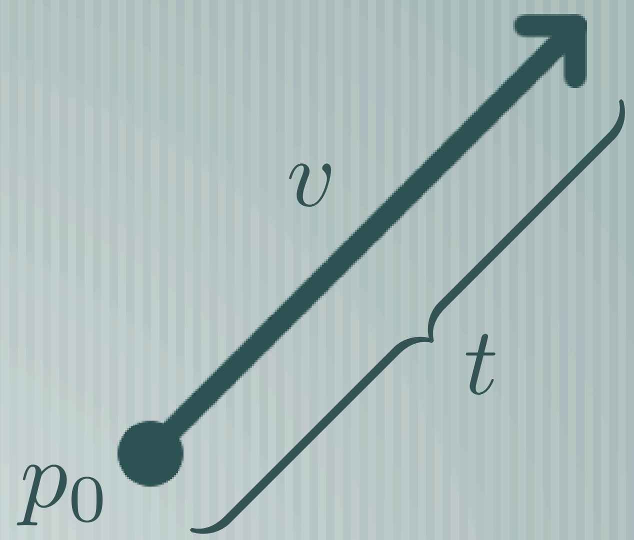
Geometric methods [4]

- General pipeline is often variant of ray tracing
  - Create many sound waves from source
  - Propagate through scene
  - Interact with scene objects
  - Collect at listener

Ray generation[6]

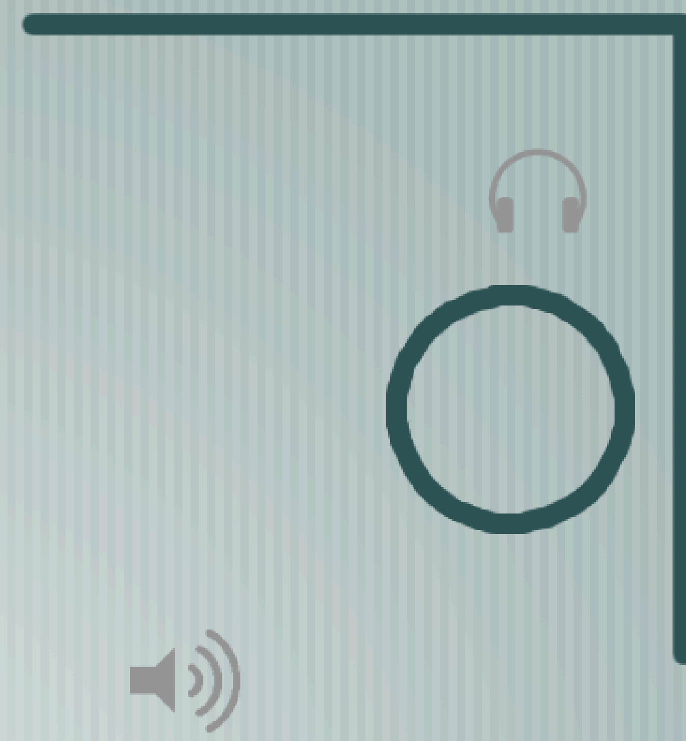
- Ray is a vector from a point

$$p = p_0 + tv$$



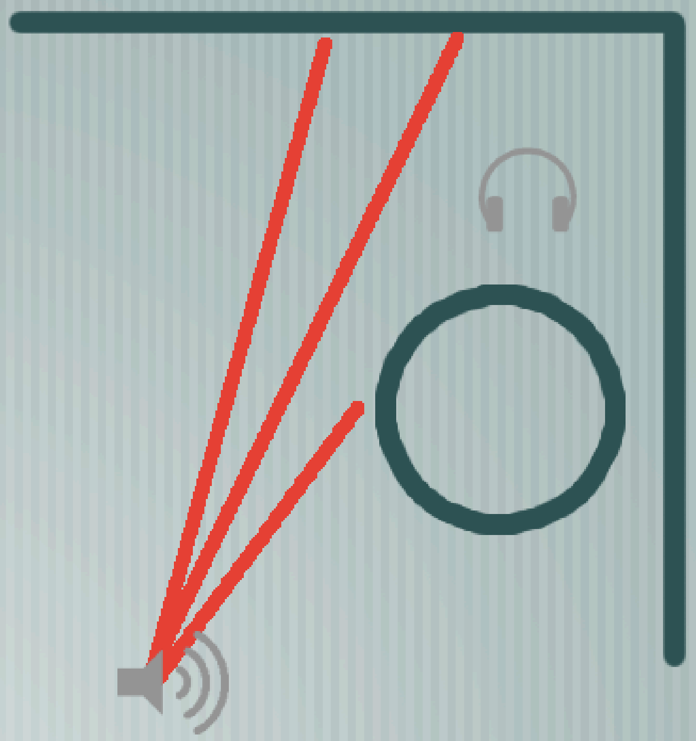
Ray generation [6]

- Send many rays into scene
- Calculate interacting objects
  - Intersect ray with object



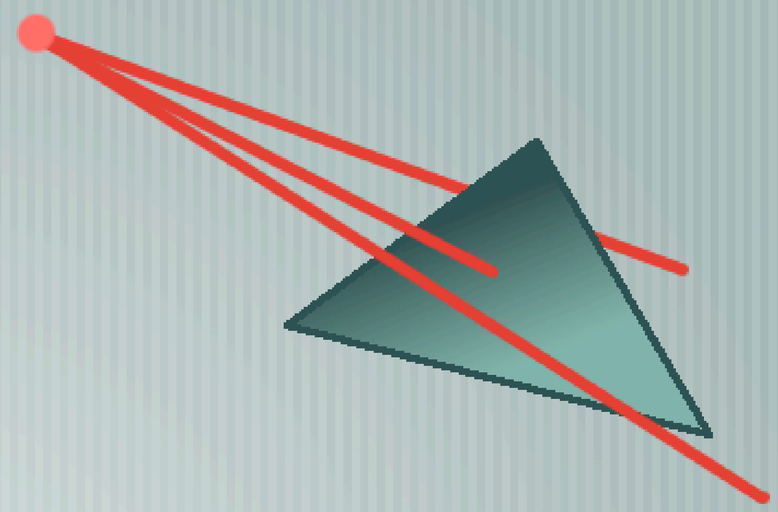
Ray generation [6]

- Send many rays into scene
- Calculate interacting objects
  - Intersect ray with object



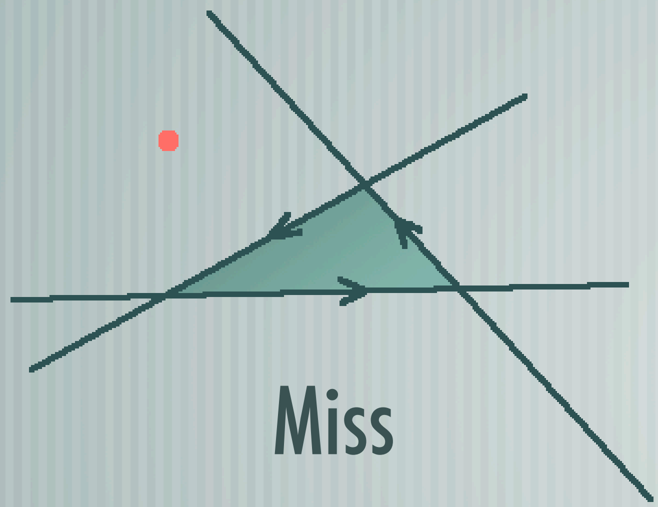
Object intersection

- Many methods
  - Transformation
  - Barycentric coordinates
  - Projection to 2d
  - Plücker coordinates

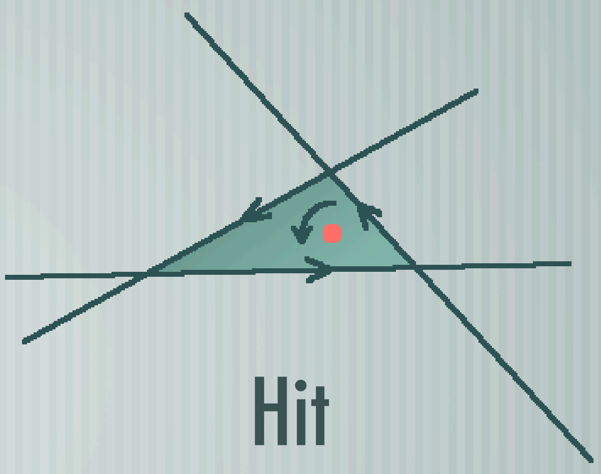


Plücker intersection [5]

- Transform triangle and ray into 6 dimension coordinates
- Determine direction triangle edges 'wrap' around ray (clockwise vs anti-clockwise)
- Test ray against all three triangle edges to intersect



Miss



Hit

Plücker intersection [5]

- Conversion to Plücker coordinates

$$p = (p_x, p_y, p_z)$$
$$q = (q_x, q_y, q_z)$$

$$\begin{aligned}
\pi_{l0} &= p_x q_y - q_x p_y \\
\pi_{l1} &= p_x q_z - q_x p_z \\
\pi_{l2} &= p_x - q_x \\
\pi_{l3} &= p_y q_z - q_y p_z \\
\pi_{l4} &= p_z - q_z \\
\pi_{l5} &= q_y - p_y
\end{aligned}$$

Plücker intersection [5]

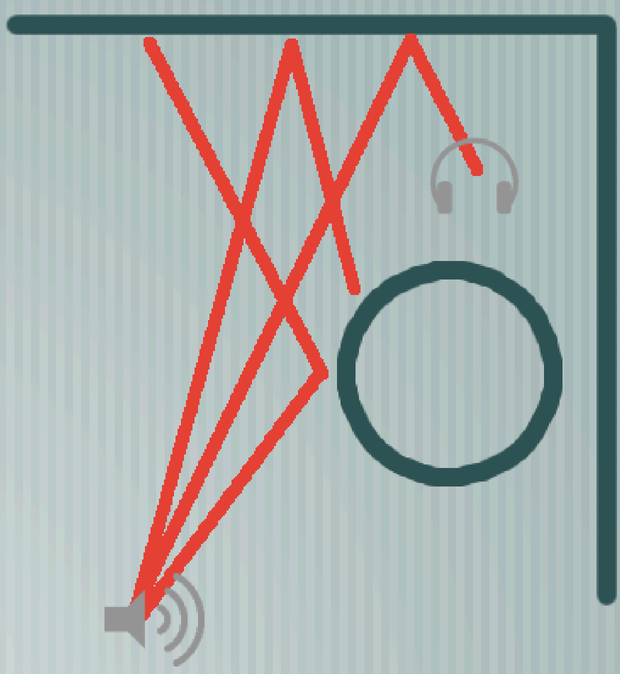
- Plücker inner product, given lines a and b

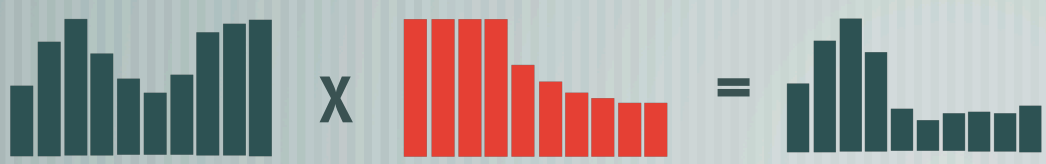
$$\pi_{a0}\pi_{b4} + \pi_{a1}\pi_{b5} + \pi_{a2}\pi_{b3} + \pi_{a3}\pi_{b0} + \pi_{a4}\pi_{b1} + \pi_{a5}\pi_{b2}$$

- Evaluate ray against 3 triangle lines
  - If all signs are the same, ray hits triangle
  - If any sign is different, ray misses triangle

Reflection

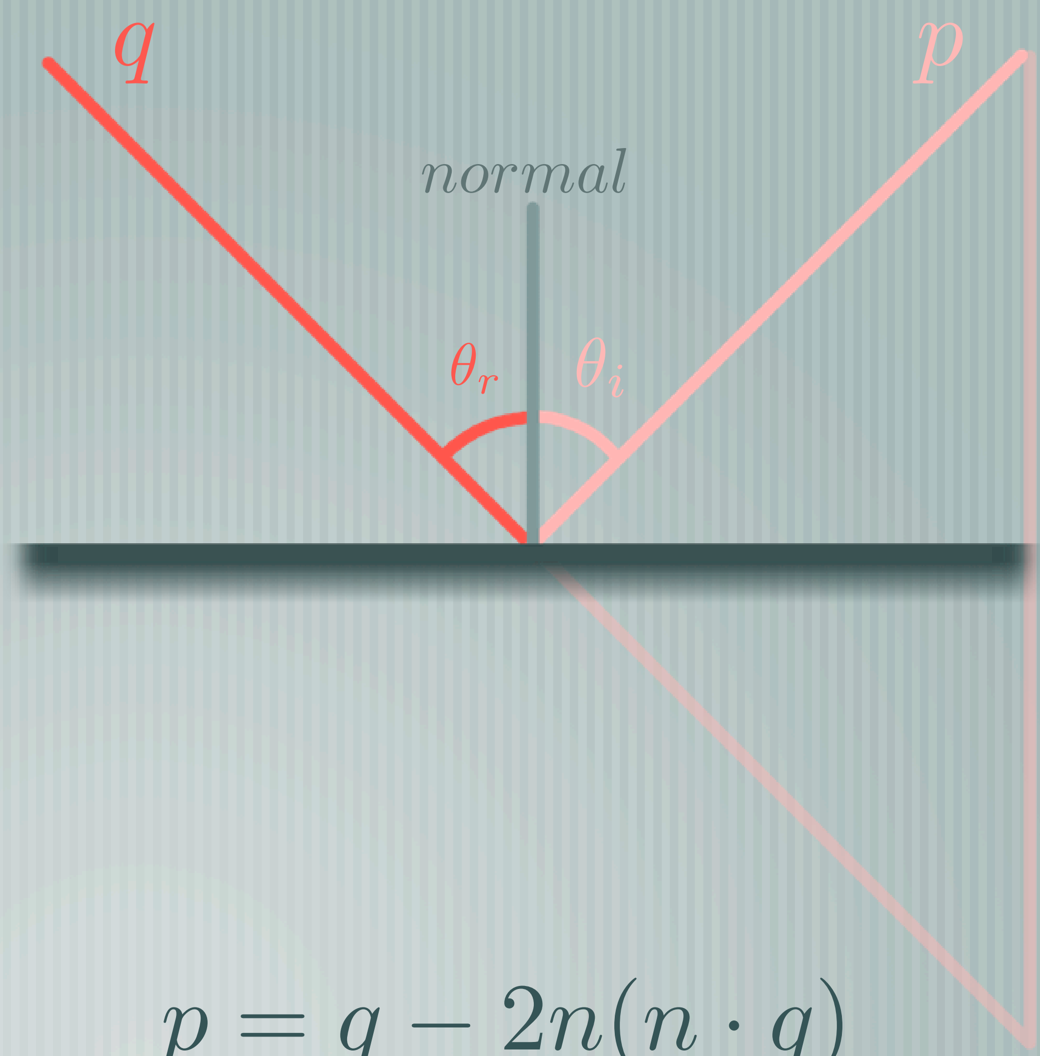
- Surfaces reflect sound
- Some sound is absorbed
  - Multiply bands by some absorption coefficient A[2]
  - Coefficient is based on material





Reflection [6]

- Very easy with rays
- Assumes flat surface



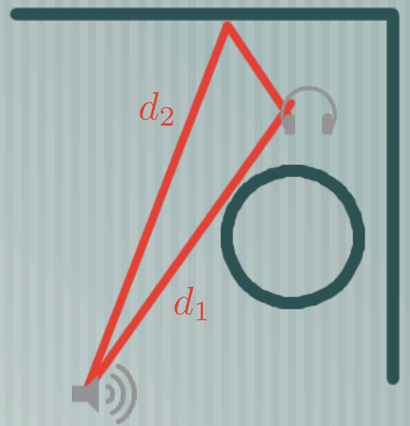
q: incoming ray
$normal$: surface normal
p: reflected ray

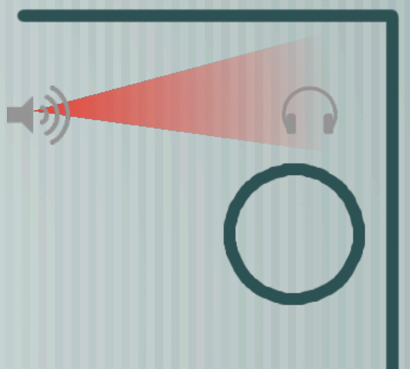
$$p = q - 2n(n \cdot q)$$

Sound output [4]

- Collect all contributing paths
- Delay of sound $\dfrac{d}{c}$
  - Uses distance of path
- Attenuation from distance $\dfrac{1}{d}$
  - Inverse distance

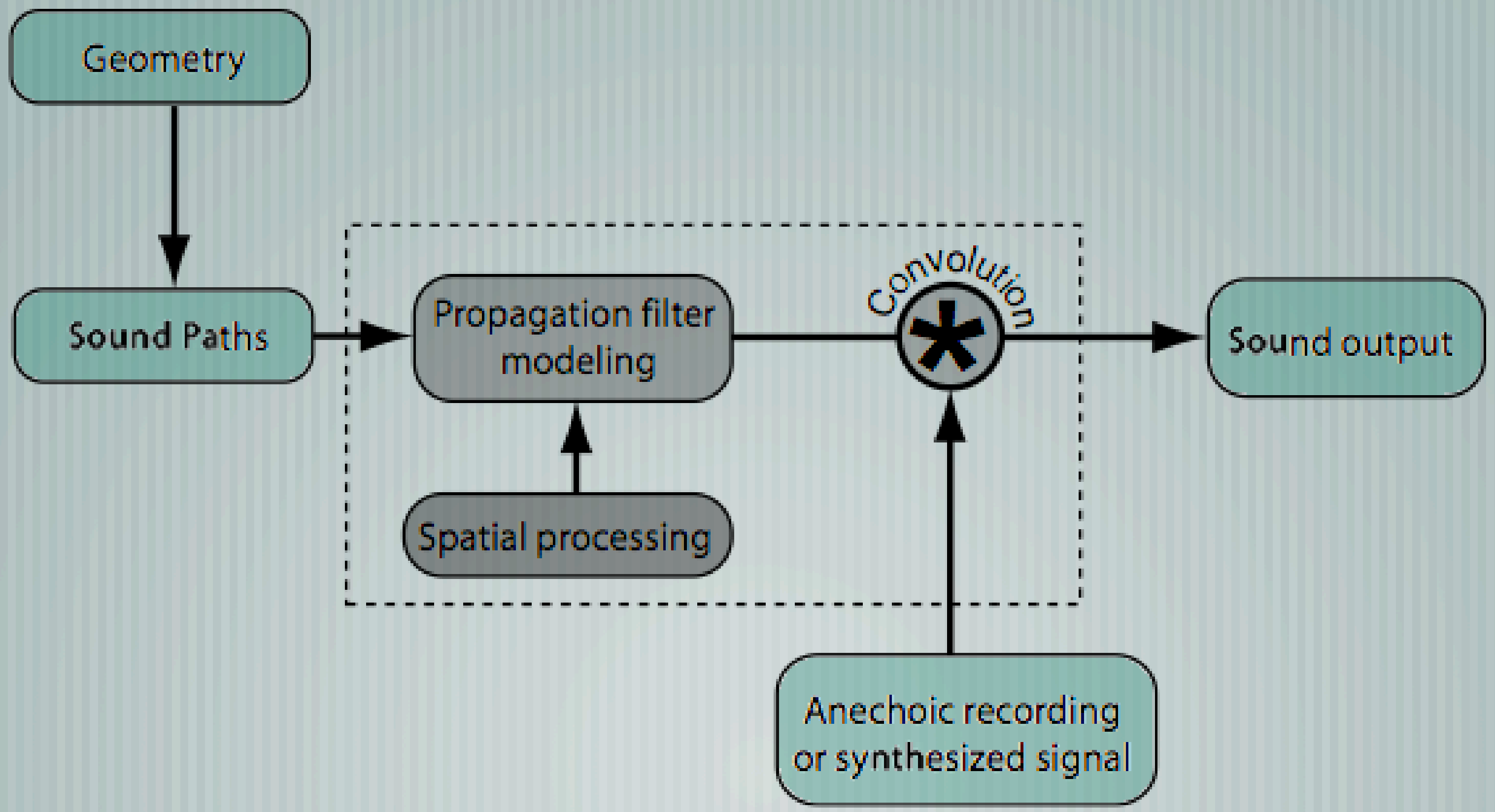
d: path distance
c: speed of sound



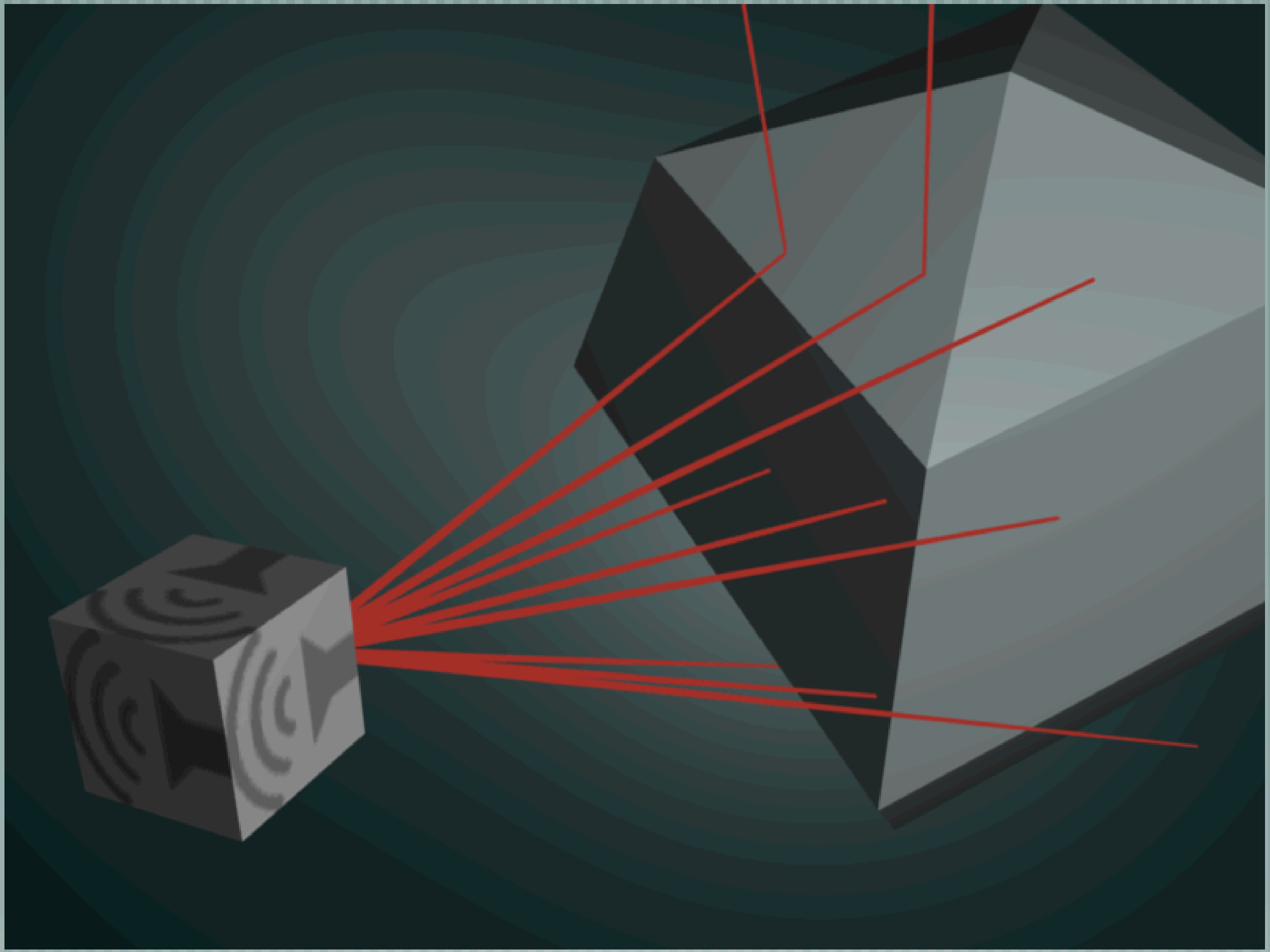


Sound output [4]

Sound pipeline



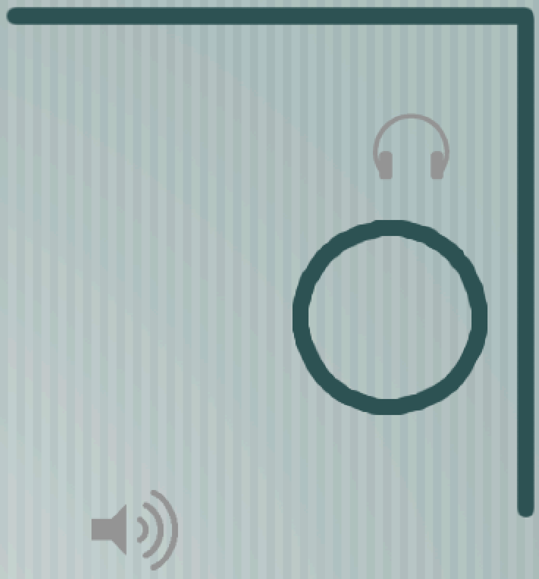
Demo



Ray tracing[3]

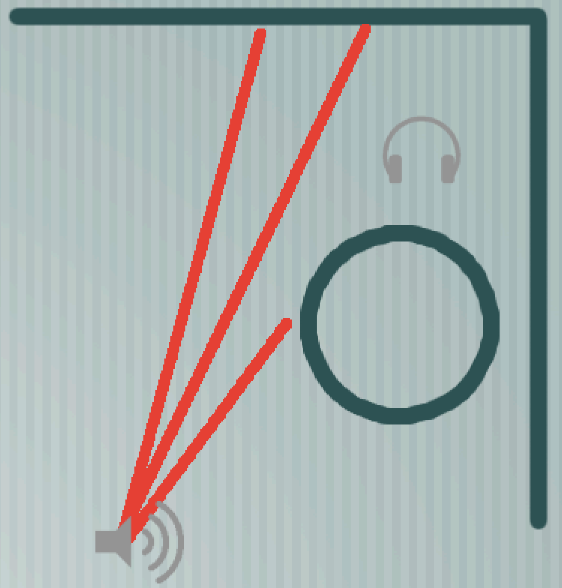
Ray tracing

- Create many rays from sound source
- Propagate through scene
- Collect rays at listener



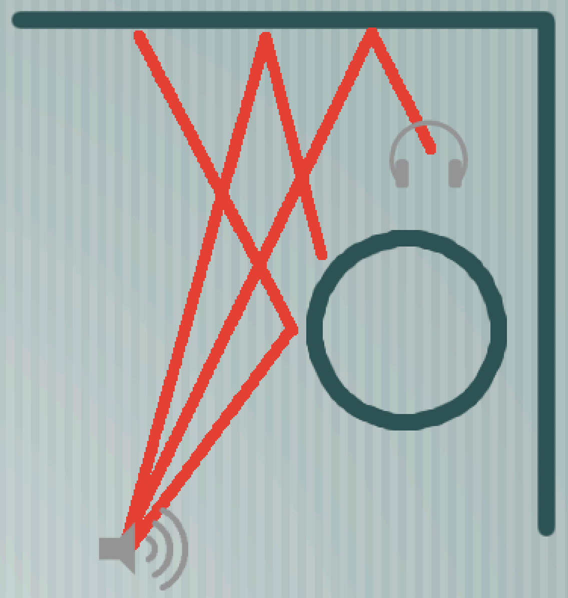
Ray tracing

- Create many rays from sound source
- Propagate through scene
- Collect rays at listener



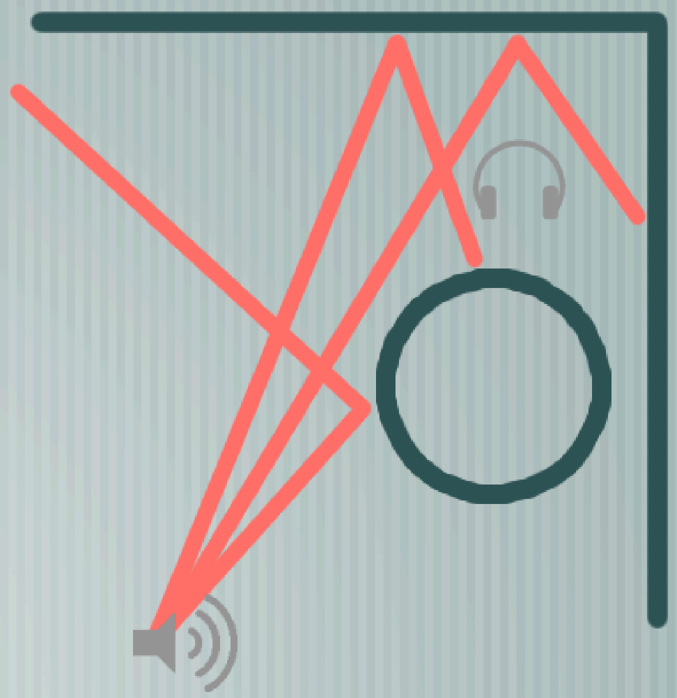
Ray tracing

- Create many rays from sound source
- Propagate through scene
- Collect rays at listener



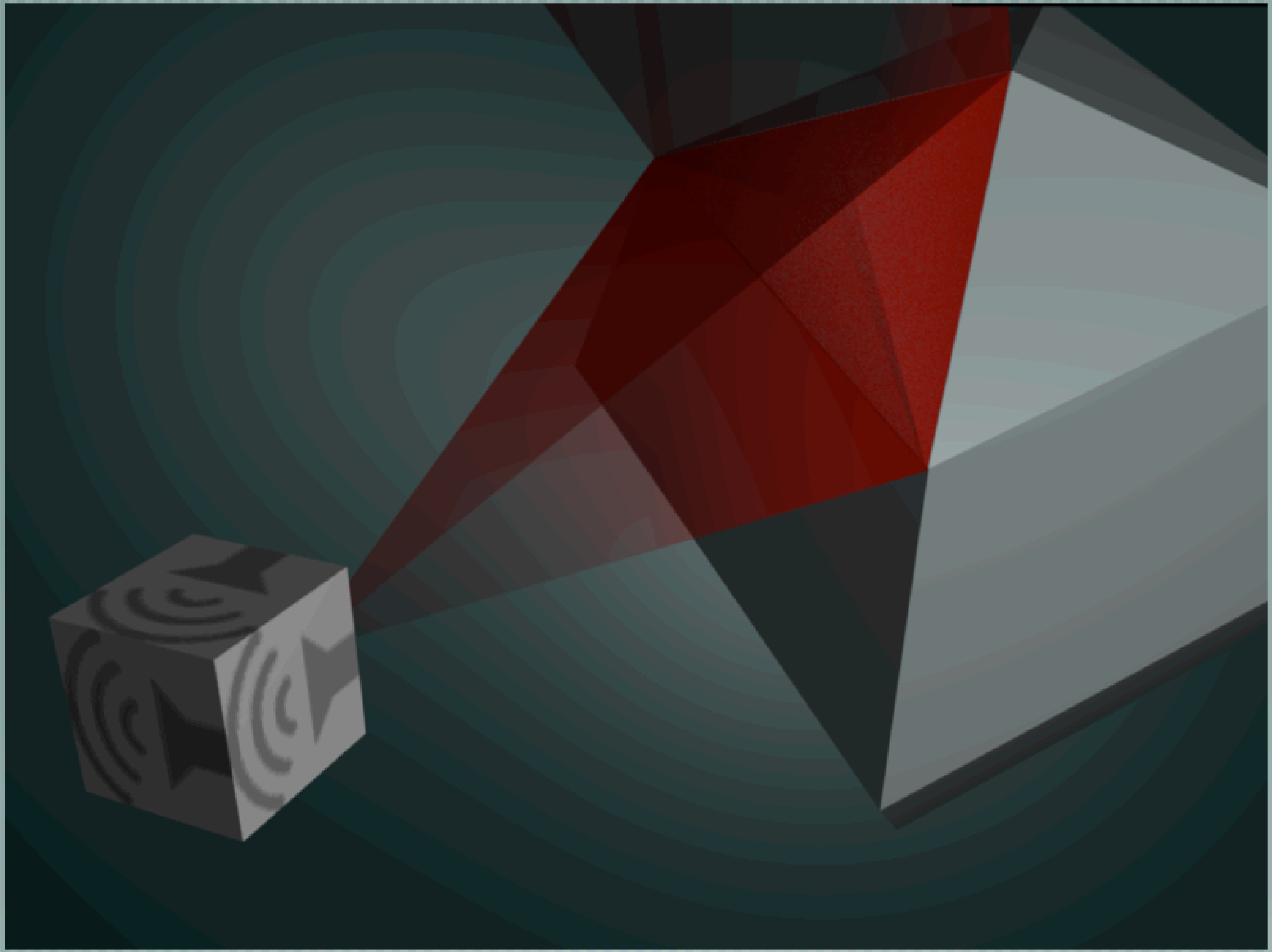
Ray tracing

- Since rays are discrete, listener may be between rays
  - Rays spread out over distance
  - Acute reflections can spread rays
- Use more rays to correct
  - Decreases performance



Ray tracing

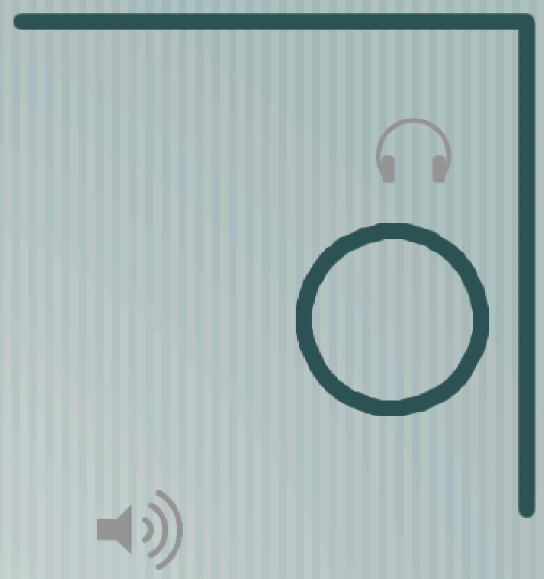
- Advantages
  - Simple to implement
- Disadvantages
  - Suffers from aliasing
  - Slow due to aliasing



Beam tracing[2]

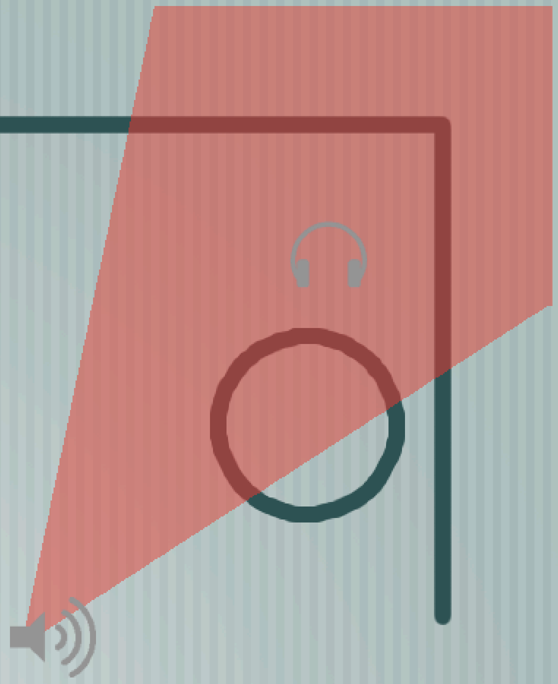
Beam tracing

- Compute BSP structure
- Create beam from sound source
  - Clip with surrounding geometry
- Continue propagation through scene
- Collect beams at listener



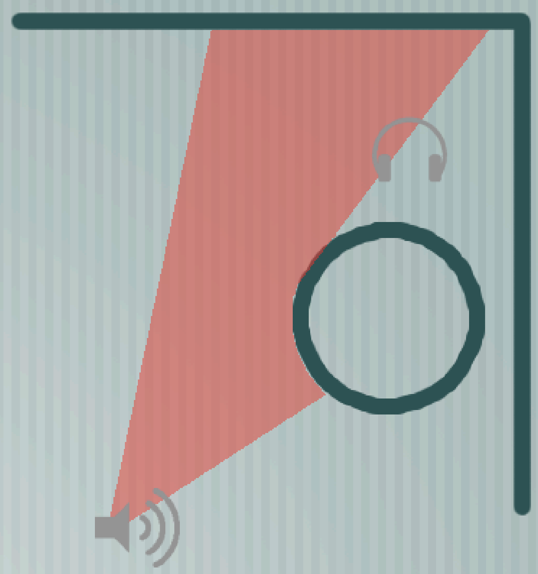
Beam tracing

- Compute BSP structure
- Create beam from sound source
  - Clip with surrounding geometry
- Continue propagation through scene
- Collect beams at listener



Beam tracing

- Compute BSP structure
- Create beam from sound source
    - Clip with surrounding geometry
- Continue propagation through scene
- Collect beams at listener

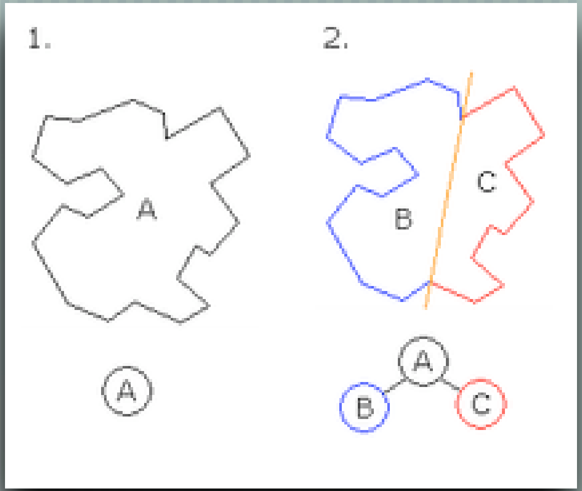


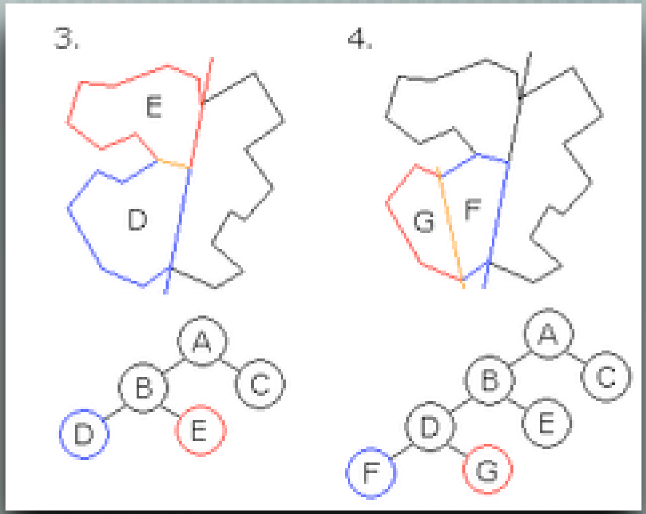
Beam tracing

- Compute BSP structure
- Create beam from sound source
    - Clip with surrounding geometry
- Continue propagation through scene
- Collect beams at listener

Beam tracing

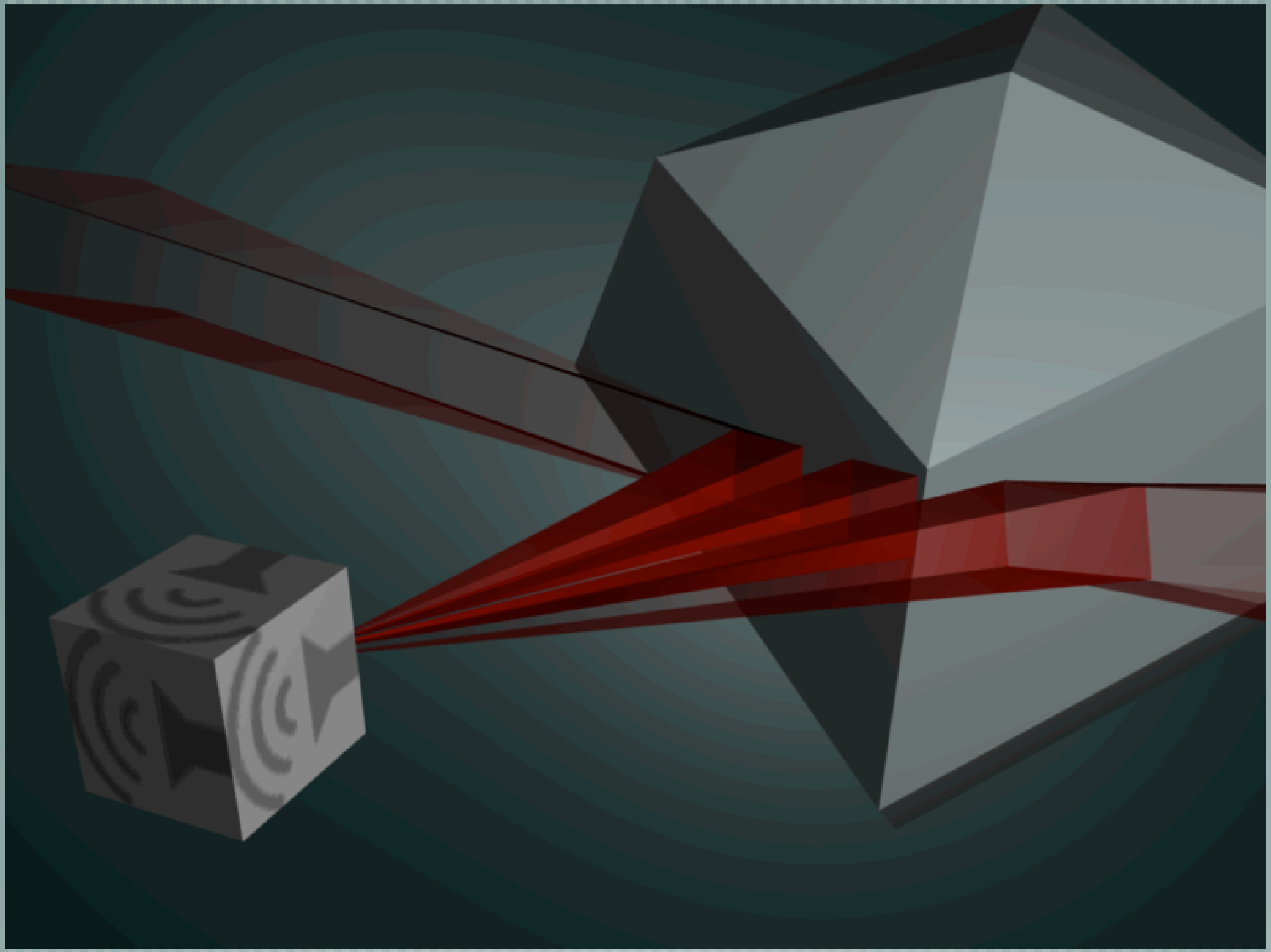
- Relies on BSP tree
  - Made famous in *Doom*
  - Divides scene into cells
  - Slow to generate
  - Cannot be dynamically updated





Beam tracing

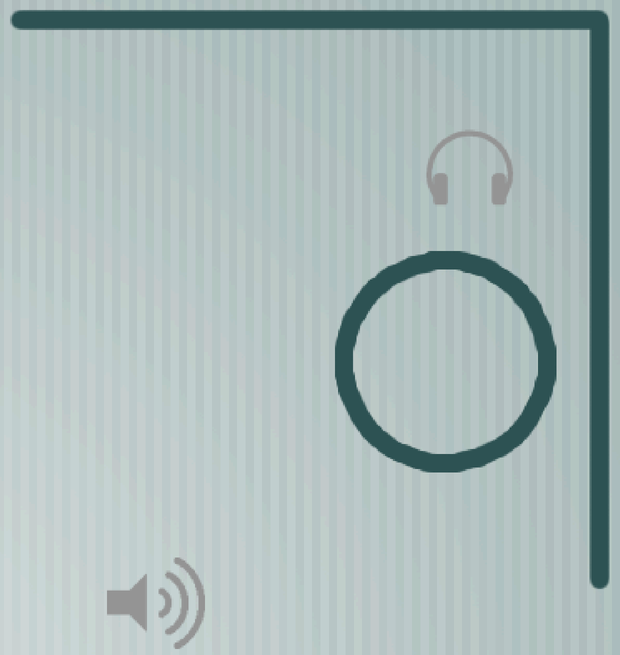
- Advantages
  - Good accuracy
  - Fast for static scenes
  - Volume method
- Disadvantages
  - Complex to implement



Frustum tracing[1]

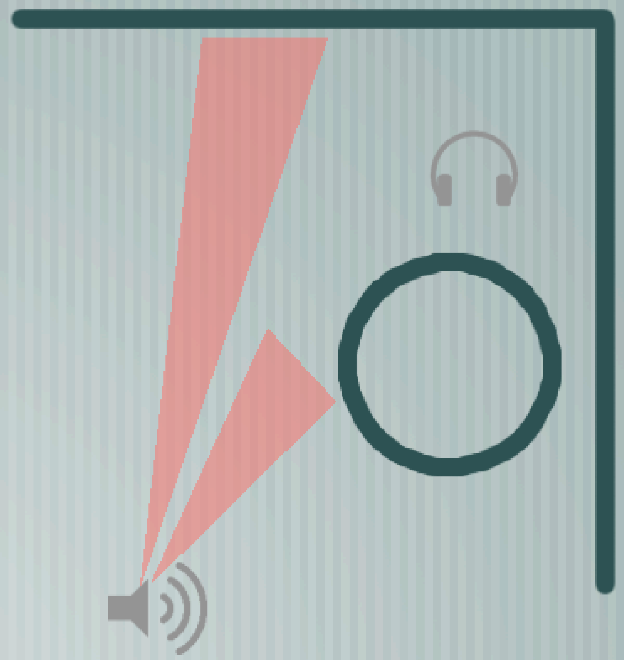
Frustum tracing

- Create many frusta from sound source
- Propagate through scene
  - Subdivide if needed
- Collect frusta at listener



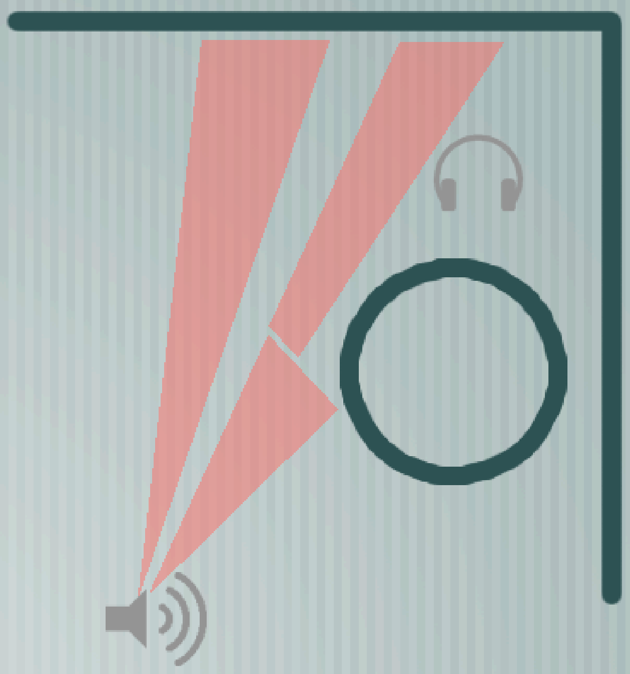
Frustum tracing

- Create many frusta from sound source
- Propagate through scene
  - Subdivide if needed
- Collect frusta at listener



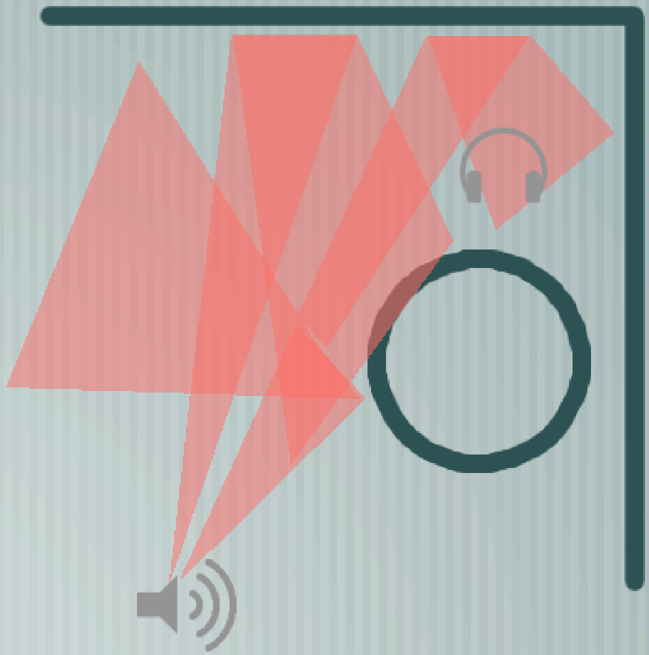
Frustum tracing

- Create many frusta from sound source
- Propagate through scene
  - Subdivide if needed
- Collect frusta at listener



Frustum tracing

- Create many frusta from sound source
- Propagate through scene
  - Subdivide if needed
- Collect frusta at listener



Frustum tracing

- Relies on some form of sub-division to minimize error

Triangle coverage is not perfectly accurate

Frustum tracing

- Advantages
  - Very fast performance
  - Supports dynamism
  - Mostly volume method
- Disadvantages
  - Complex to implement
  - Small inaccuracies in rendering

Summary

Ray tracing

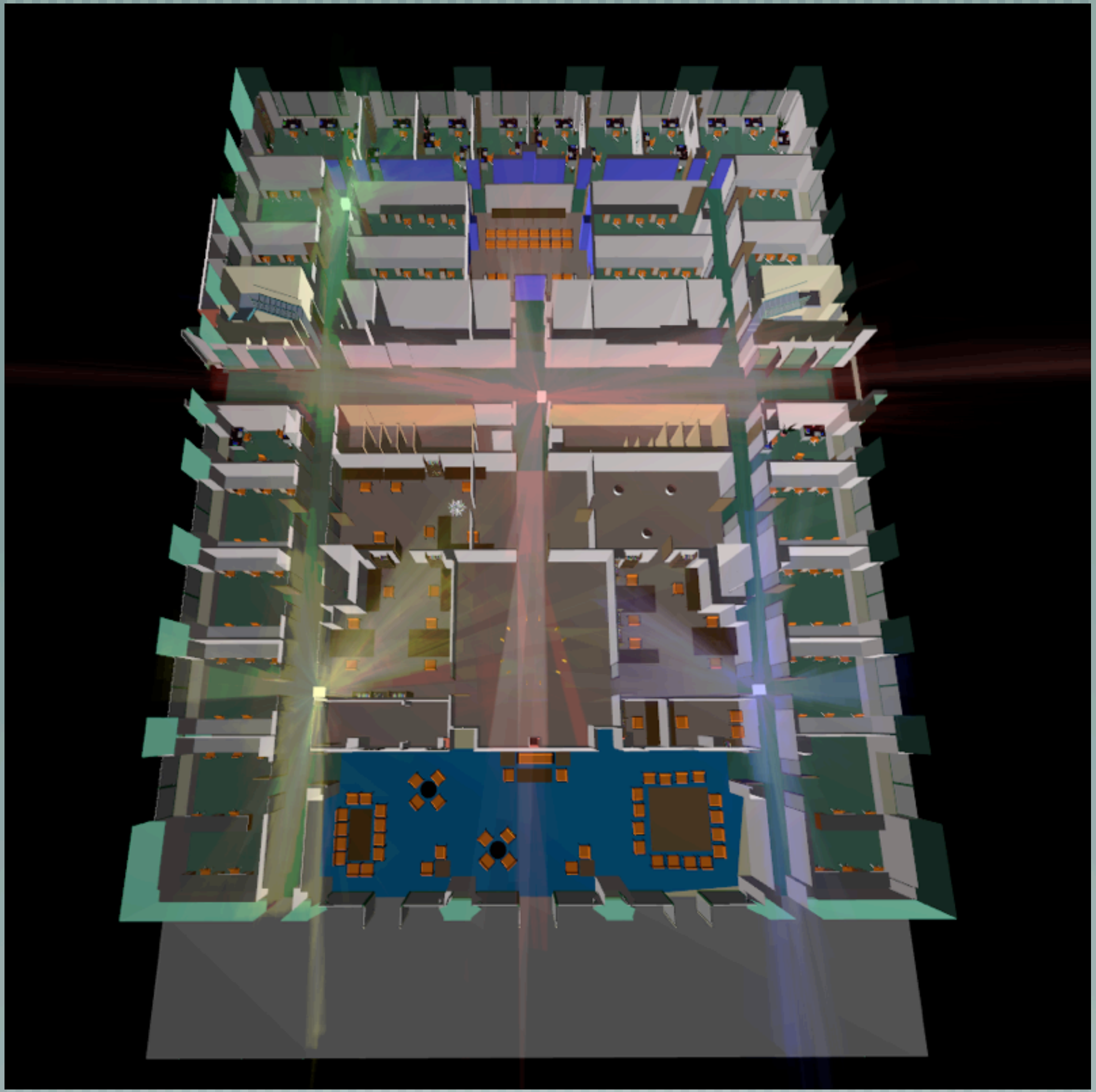
- Classical slow technique
- Still in use by industry

Beam tracing

- Fast with full coverage
- No dynamism

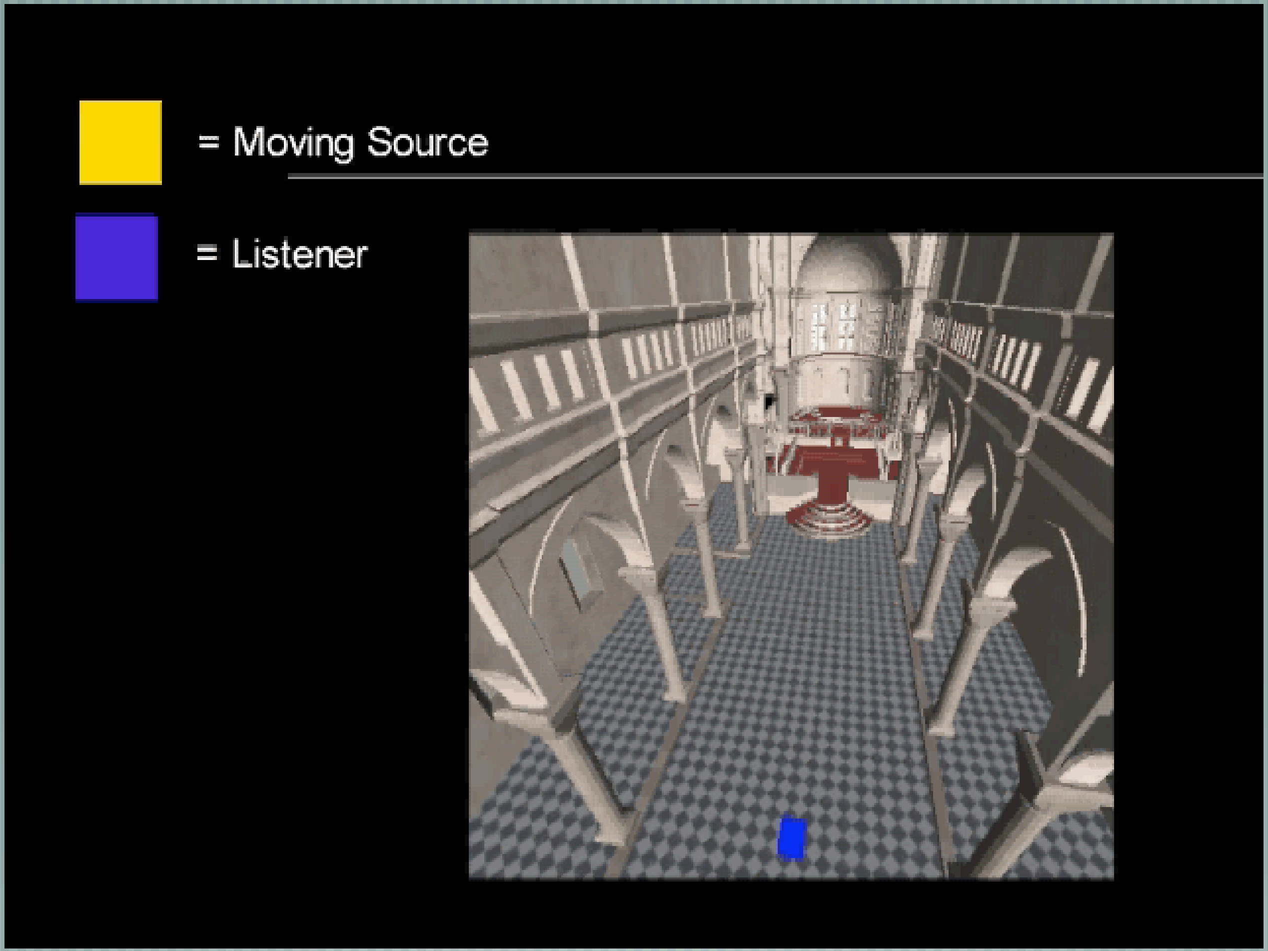
Frustum tracing

- Fast with dynamism
- Small inaccuracies in coverage



Results

Results



Results

Sound seminar at UNC

- A sound seminar may be offered Spring 2008
- Contact Dinesh Manocha for details
  - dm@cs.unc.edu

References

[1] Interactive Sound Propagation in Dynamic Scenes Using Frustum Tracing (2007), Lauterbach at al.

[2] A Beam Tracing Approach to Acoustic Modeling for Interactive Virtual Environments (1998), Funkhouser et al.

[3] Calculating the Acoustical Room Response by the Use of a Ray Tracing Technique (1968), Krokstad et al.

[4] Survey of Methods for Modeling Sound Propagation in Interactive Virtual Environment Systems (2003), Funkhouser et al.

[5] Ray Tracing Triangular Meshes (1997), Amanatides et al.

[6] An Introduction to Ray Tracing (1989), Glassner et al.

Questions?