Sound Synthesis

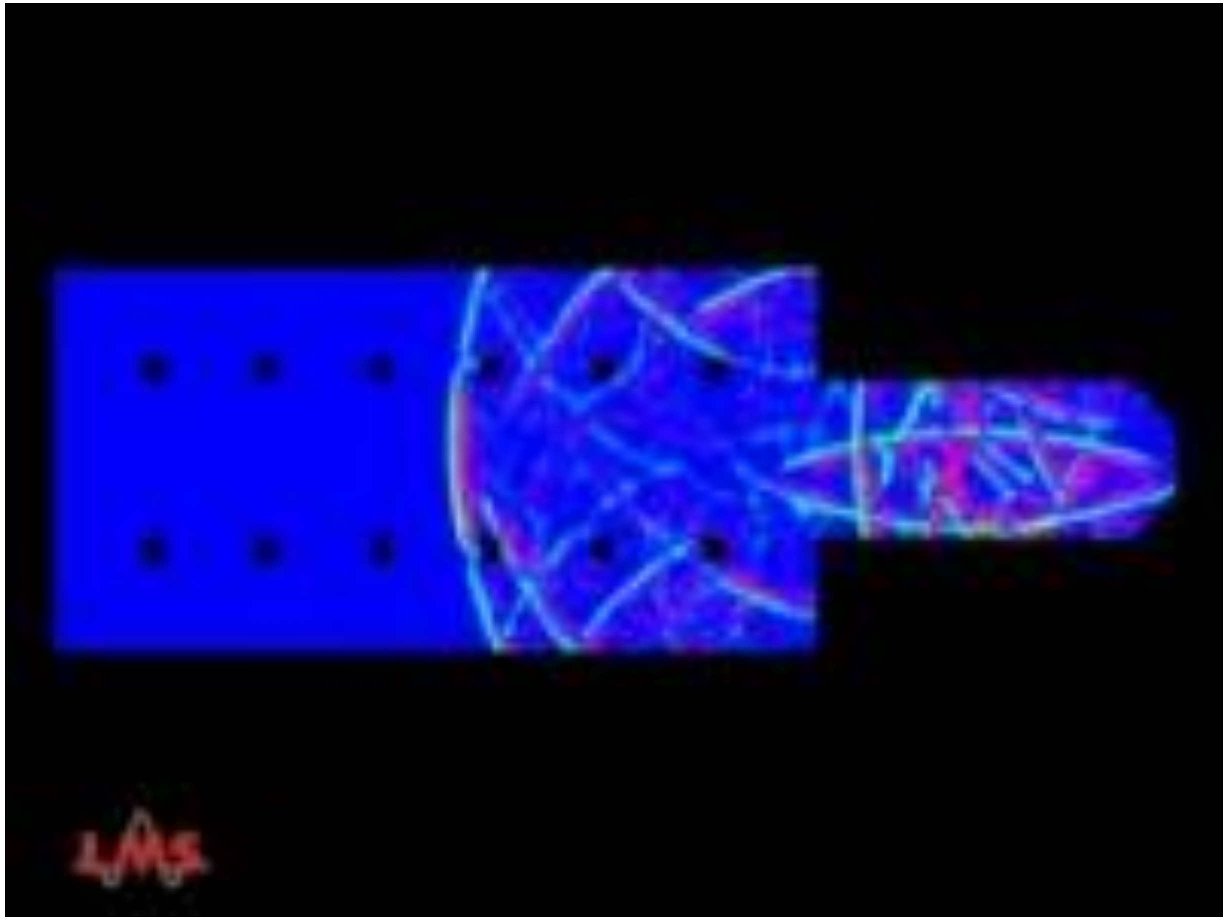
안인규 (Inkyu An)

Speech And Audio Recognition
(오디오 음성인식)

https://mairlab-km.github.io/

This lecture material is based on the Lecture Notes (EE837) of Prof. Jung-Woo Choi at KAIST
and the Ph.D. dissertation of Dr. Taeyoung Kim at Samsung Electronics

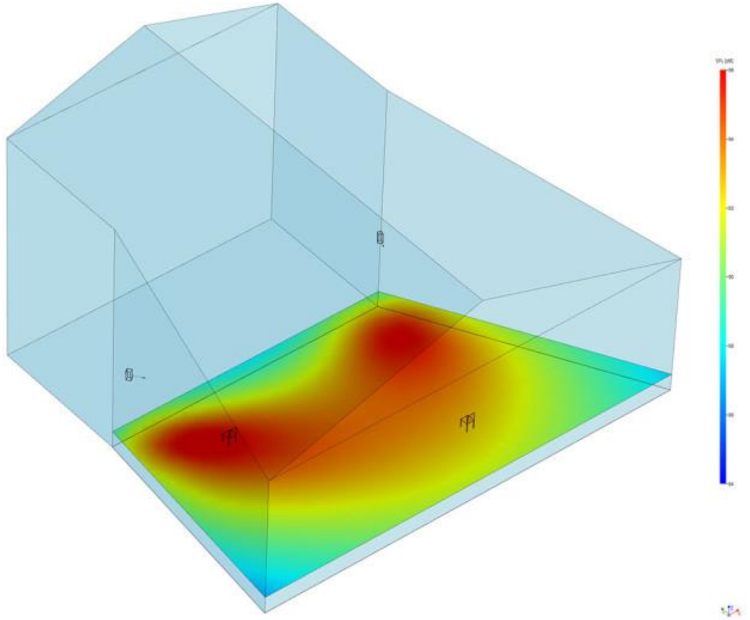
How does sound propagate?



https://www.youtube.com/watch?v=Xsx4VBEKciA

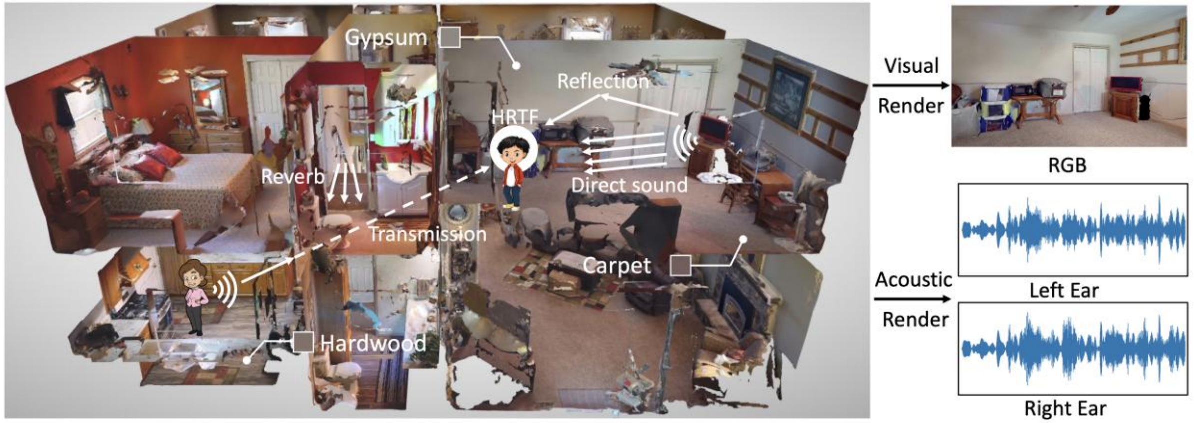
https://www.youtube.com/shorts/GnAvquGBmnY

To create realistic sound!



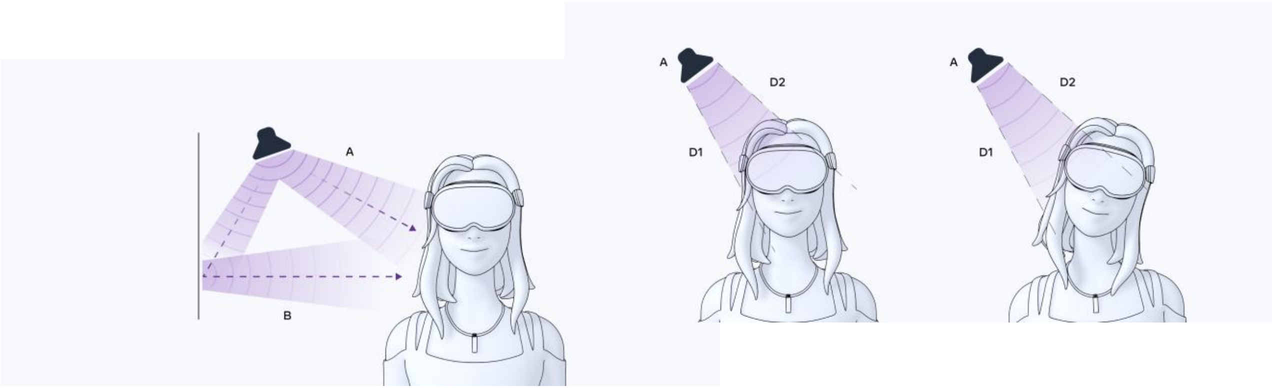
Acoustic simulation (designing Auditorium)

https://audioxpress.com/article/acoustic-simulation-ease-5-s-acousteer-engine-redefines-design-workflows



Sound simulation (generating synthetic data)

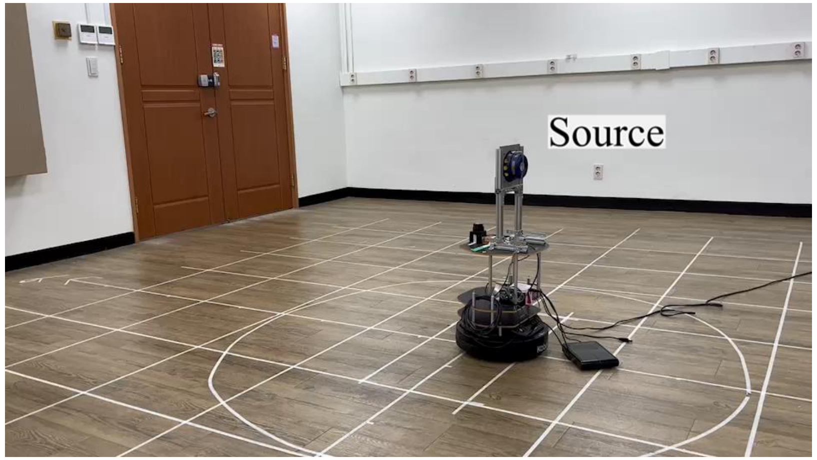
https://vision.cs.utexas.edu/projects/soundspaces2/



Sound simulation (Spatial audio in VR)

https://developers.meta.com/horizon/design/spatial_audio/

To create realistic sound!



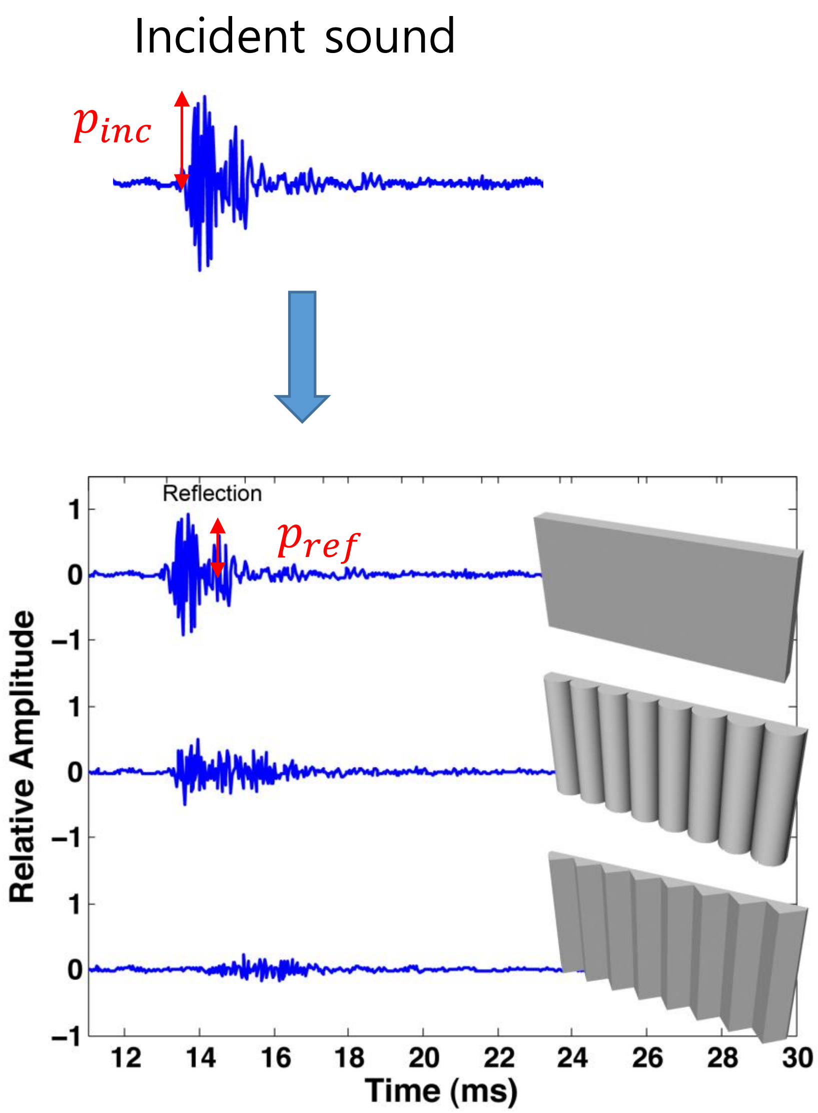
Recorded sound in the real environment

Generated sound in the virtual environment

Reflection

- If reflection occurs, what happens?
  - If the wall is <u>a rigid body</u>, It should be a fixed boundary condition

  - How will the sound intensity change?
    - Reflection coefficient: $R_p = \frac{p_{ref}}{p_{inc}}$
    - Reflection coefficient는 material마다 다르다!



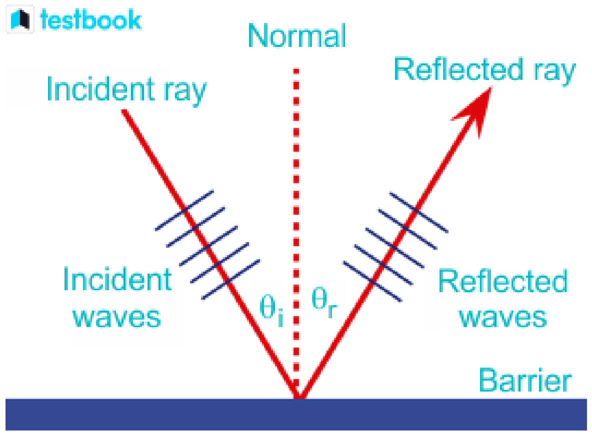
Reflection

- If reflection occurs, what happens?
  - If the wall is <u>a rigid body</u>, It should be a fixed boundary condition
  - How will the sound intensity change?
    - Reflection coefficient: $R_p = \frac{p_{ref}}{p_{inc}}$
    - Reflection coefficient는 material마다 다르다!

$$p_{inc}(x,t) = Ae^{i(kx-\omega t)}$$

$$\Downarrow$$

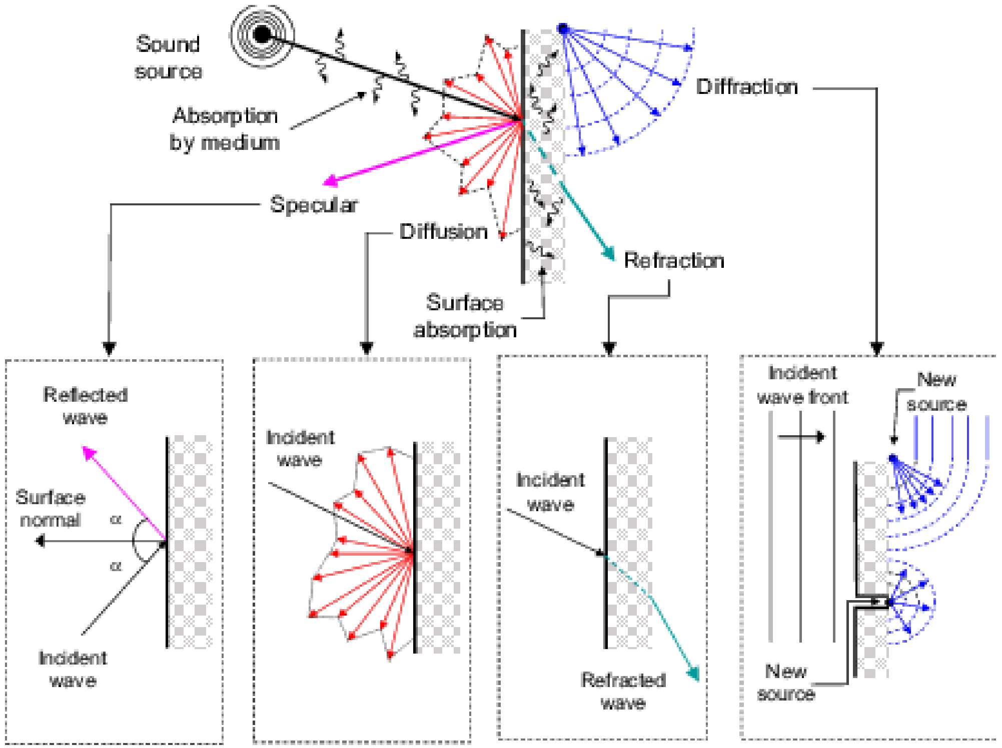
$$p_{ref}(x,t) = R_p \cdot p_{inc}(-x,t) \cdot e^{i\pi}$$



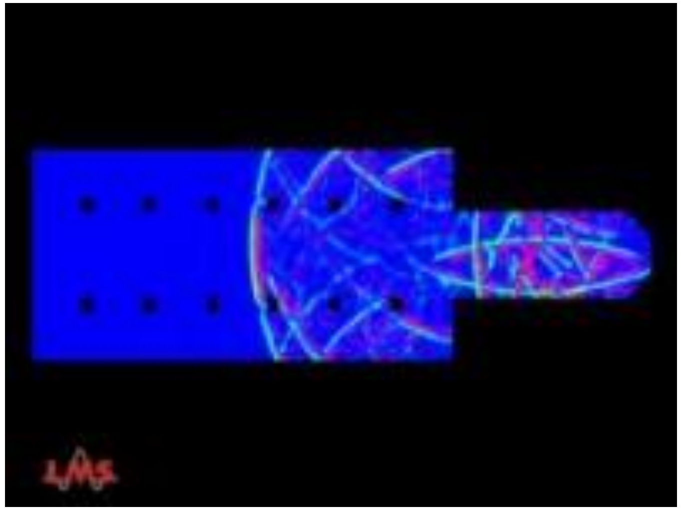
Reflection law

Reflection

- If reflection occurs, what happens?
  - If the wall is <u>a rigid body</u>, It should be a fixed boundary condition



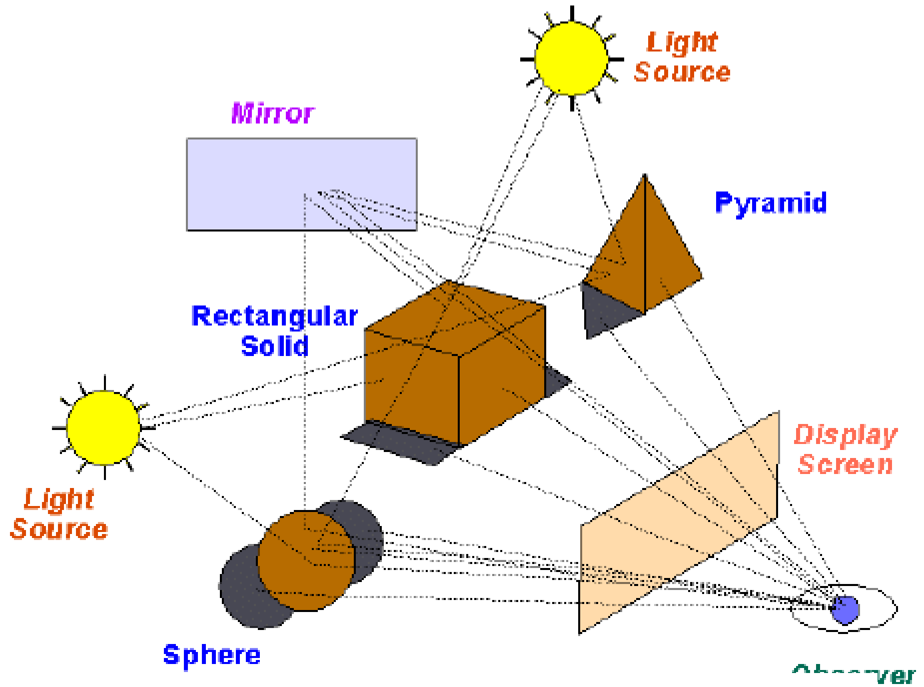
소리가 물체에 충돌할 때, specular reflection (정반사) 뿐만 아니라, **Diffuse, refraction, diffraction** 등이 함께 발생한다.



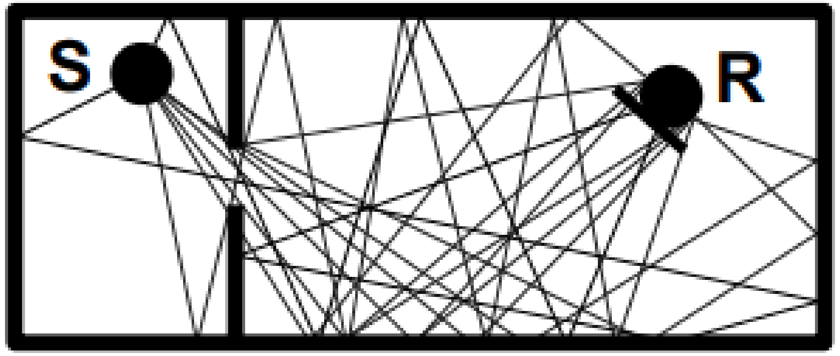
이를 정확하게 Simulation하는 것은 매우 시간이 많이 걸린다.

Geometrical Acoustics

- **Geometrical Acoustics**: 음파를 직선으로 진행하는 광선처럼 모델링하여 반사, 흡수, 산란을 다루는 음향 모델링 방법
  - 파동 방정식을 직접 풀지 않고, 고주파 근사(high-frequency approximation)를 사용하여 음파를 빛처럼 다루는 방법



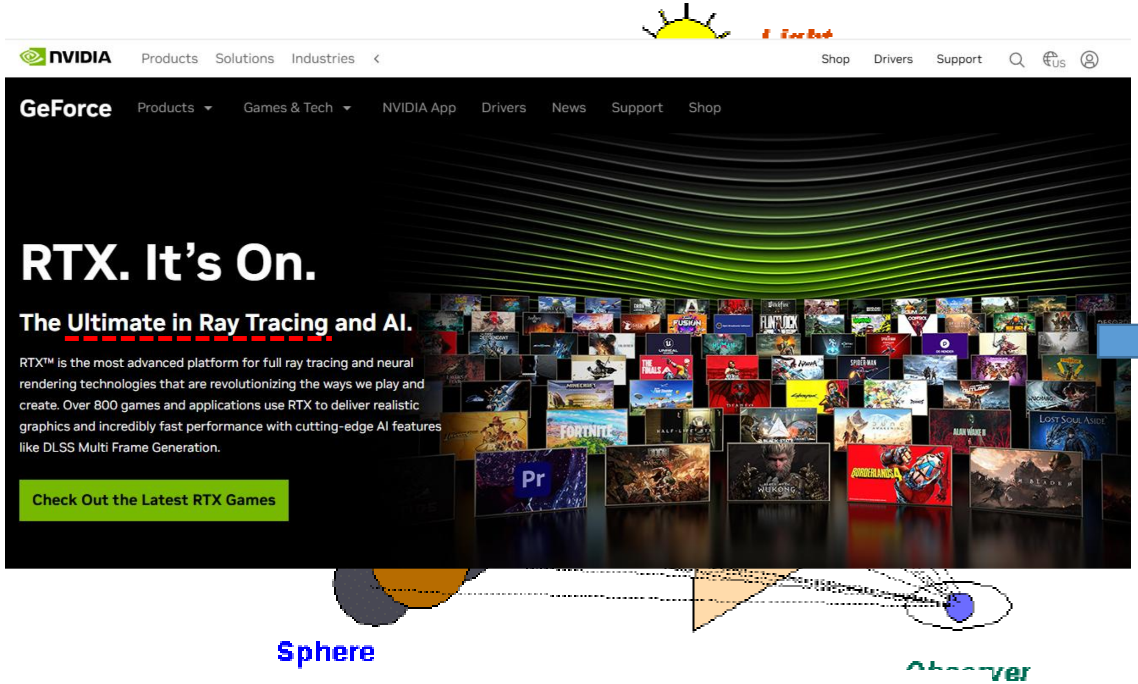
Ray tracing (Computer Graphics)



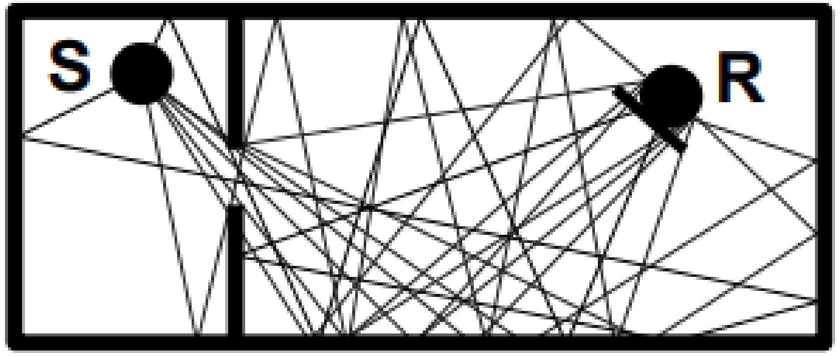
Sound propagation paths

Geometrical Acoustics

- **Geometrical Acoustics**: 음파를 직선으로 진행하는 광선처럼 모델링하여 반사, 흡수, 산란을 다루는 음향 모델링 방법
  - 파동 방정식을 직접 풀지 않고, <u>고주파 근사(high-frequency approximation)를 사용</u>하여 음파를 빛처럼 다루는 방법



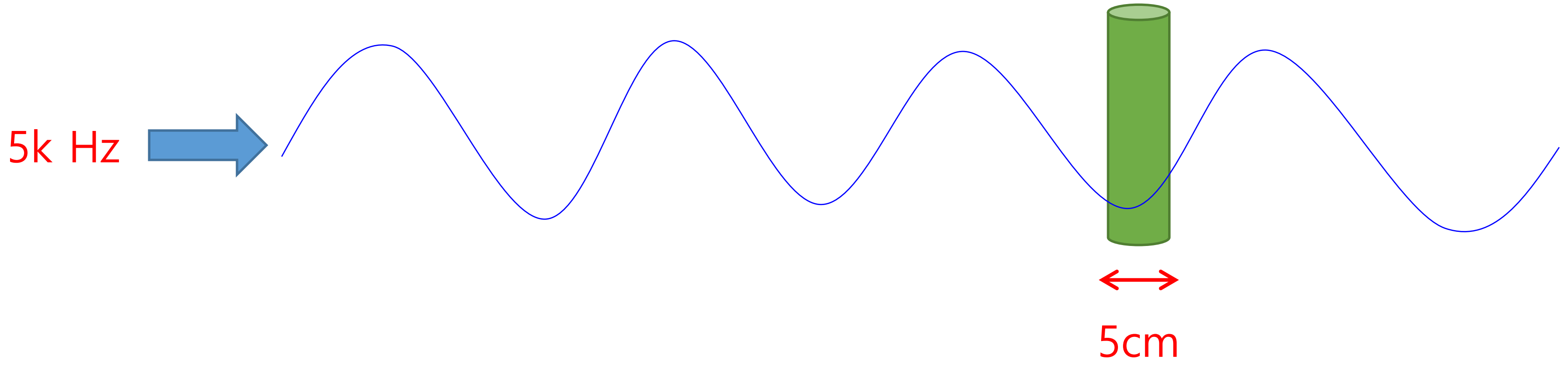
Ray tracing (Computer Graphics)



Sound propagation paths

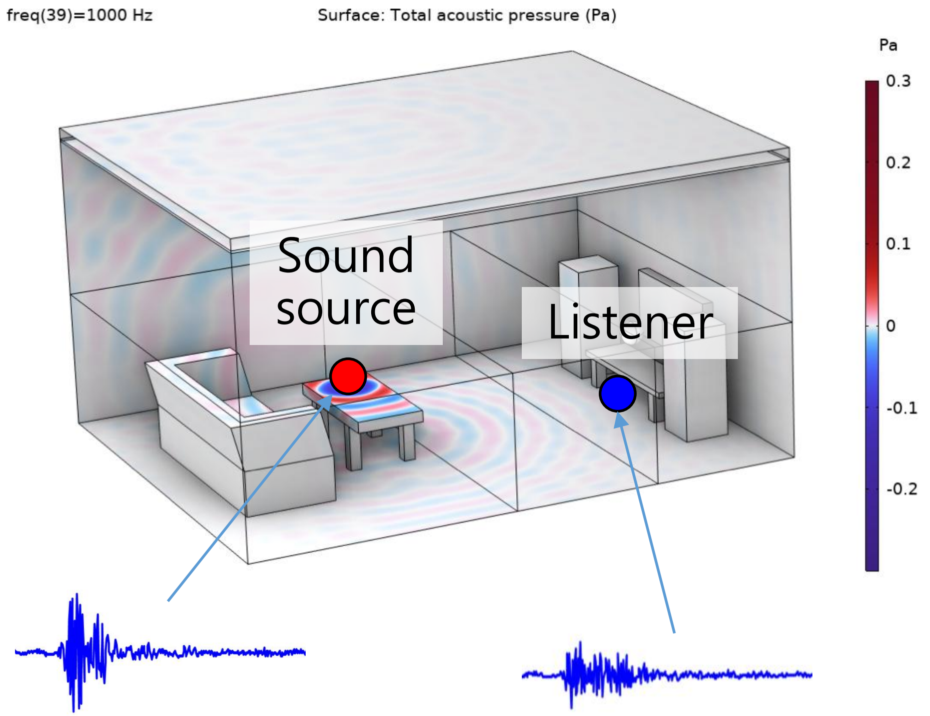
Geometrical Acoustics

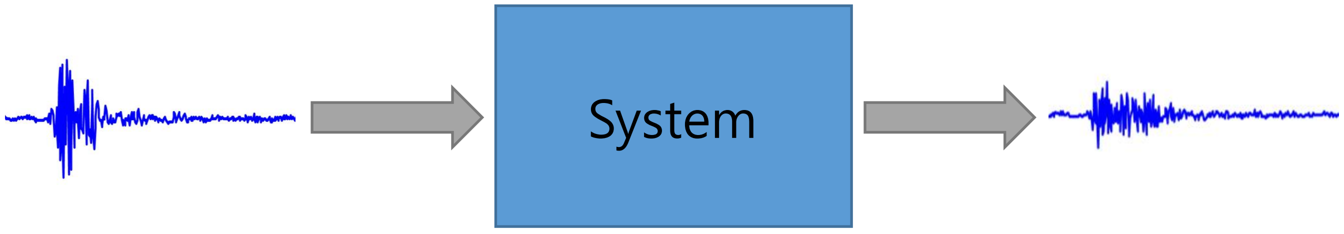
- **Geometrical Acoustics**: 음파를 직선으로 진행하는 광선처럼 모델링하여 반사, 흡수, 산란을 다루는 음향 모델링 방법
  - 파동 방정식을 직접 풀지 않고, 고주파 근사(high-frequency approximation)를 사용하여 음파를 빛처럼 다루는 방법
  - **High-frequency assumption**: 파장이 장애물 크기보다 훨씬 작을 때 $\lambda \ll D$ (예: 5kHz 의 파장은 대략 7cm, $f = \frac{c}{\lambda}$)
  - 만약 High-frequency assumption을 만족하지 않는다면, 회절이 dominant해짐



Sound Propagation System

- Sound Propagation System in Geometrical Acoustics
  - **Input**: sound source에서 발생한 소리
  - **Output**: Listener가 들을 소리

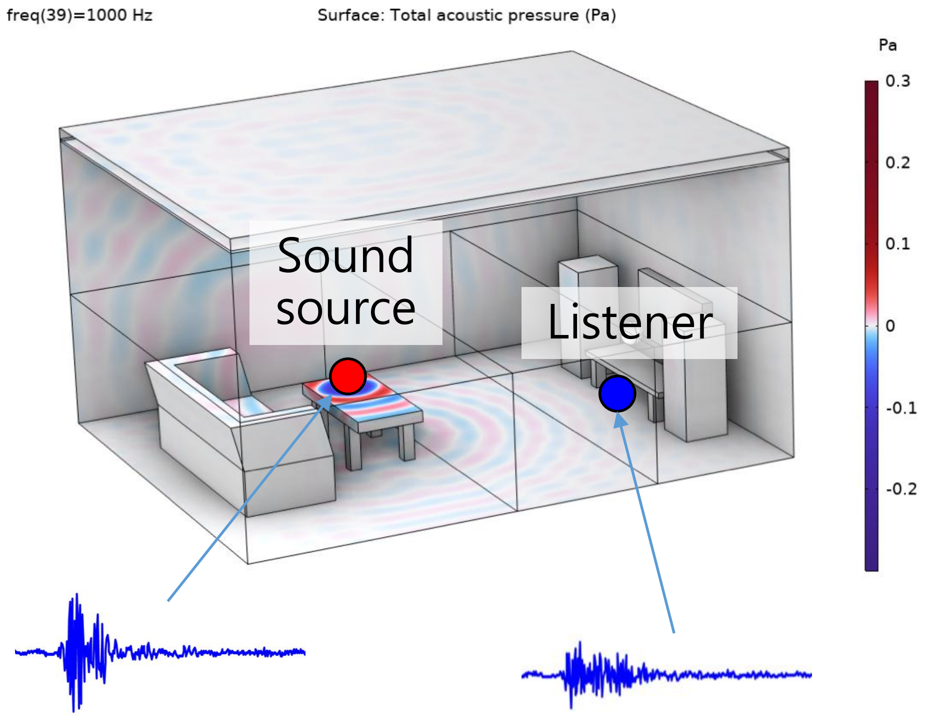


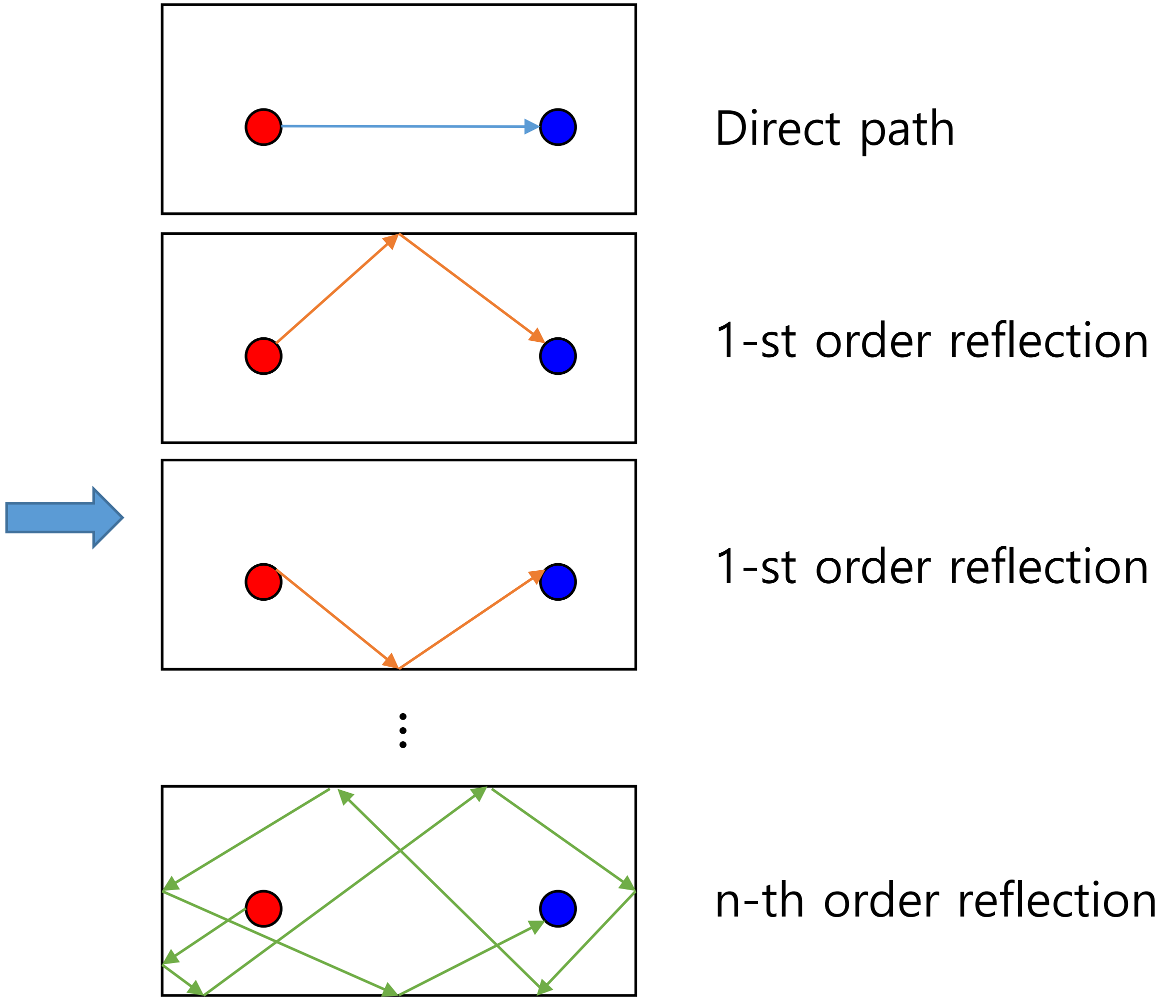


System은 어떻게 정의될까?

Sound Propagation System

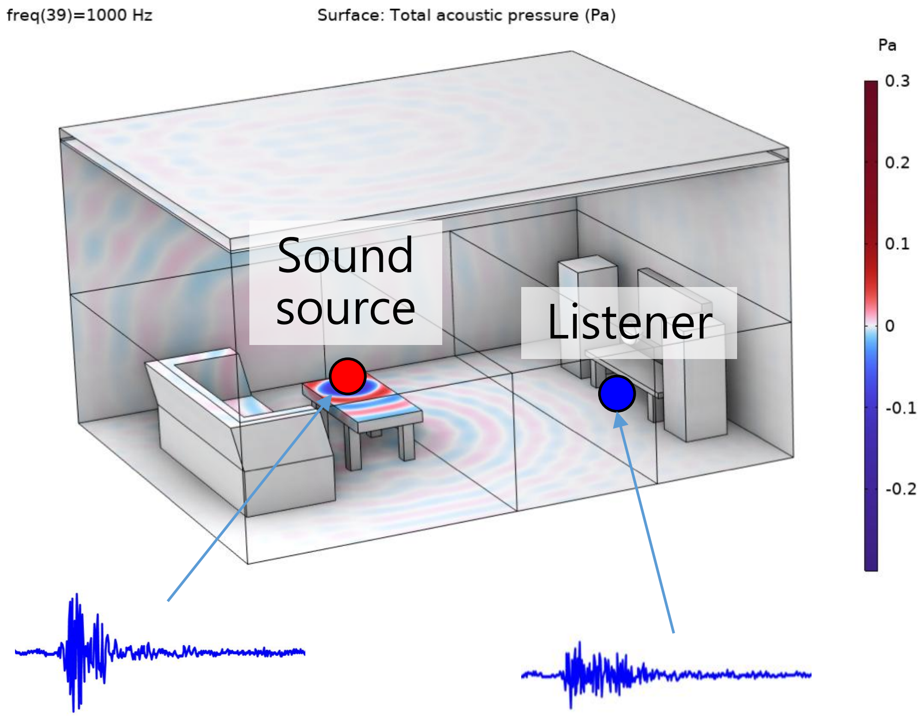
- Sound Propagation System

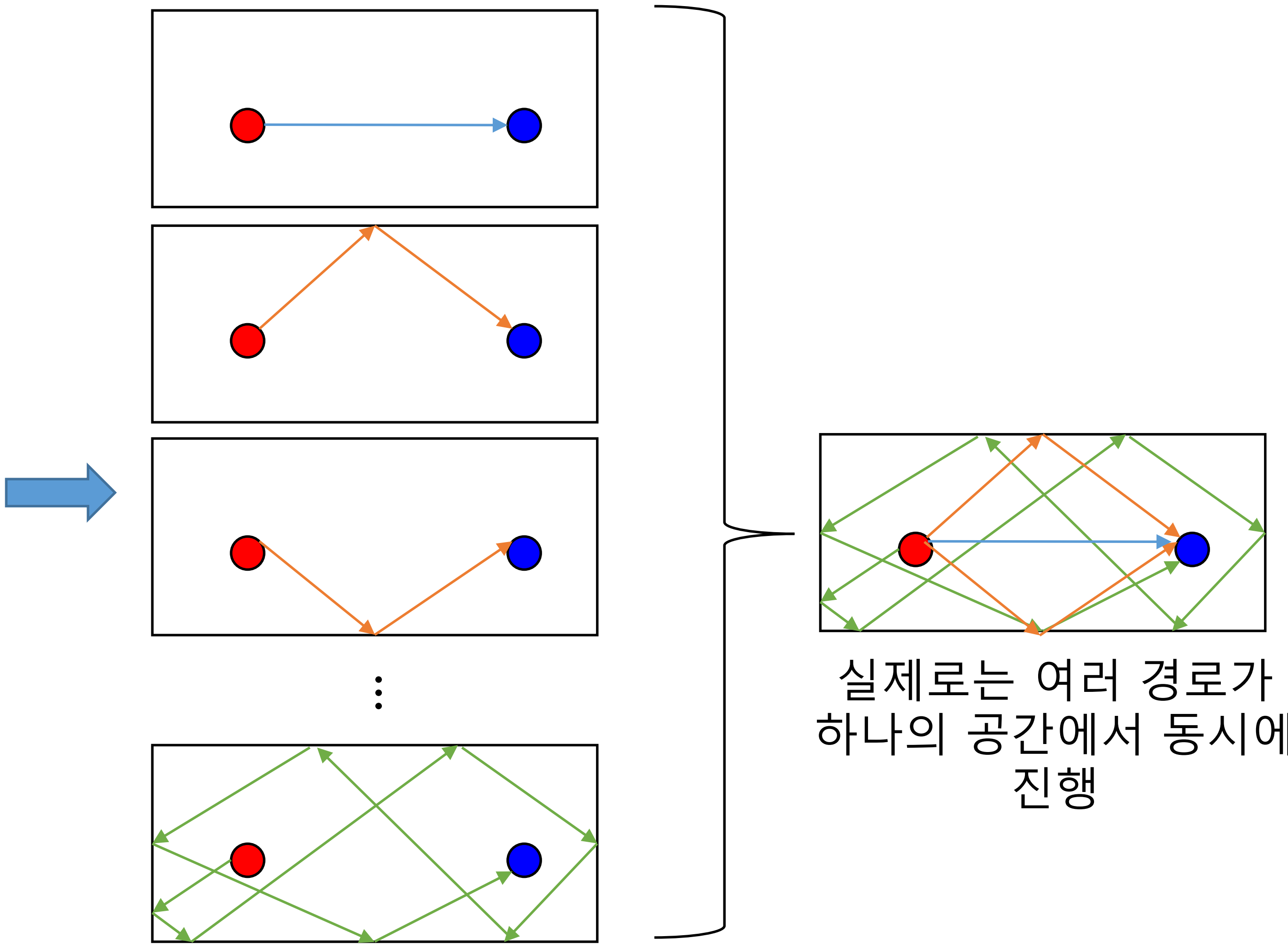




Sound Propagation System

- Sound Propagation System

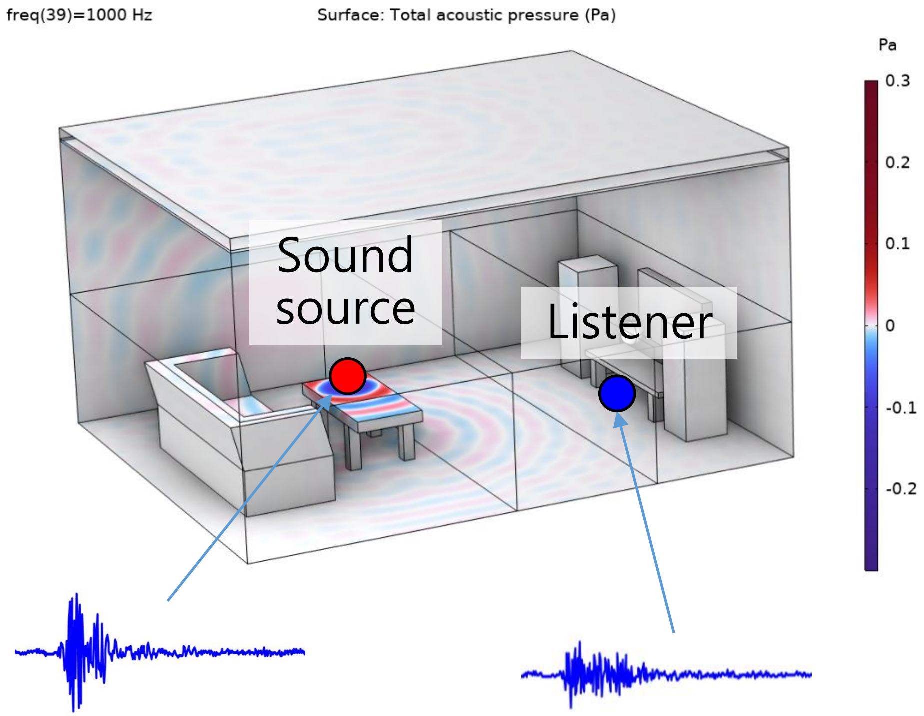


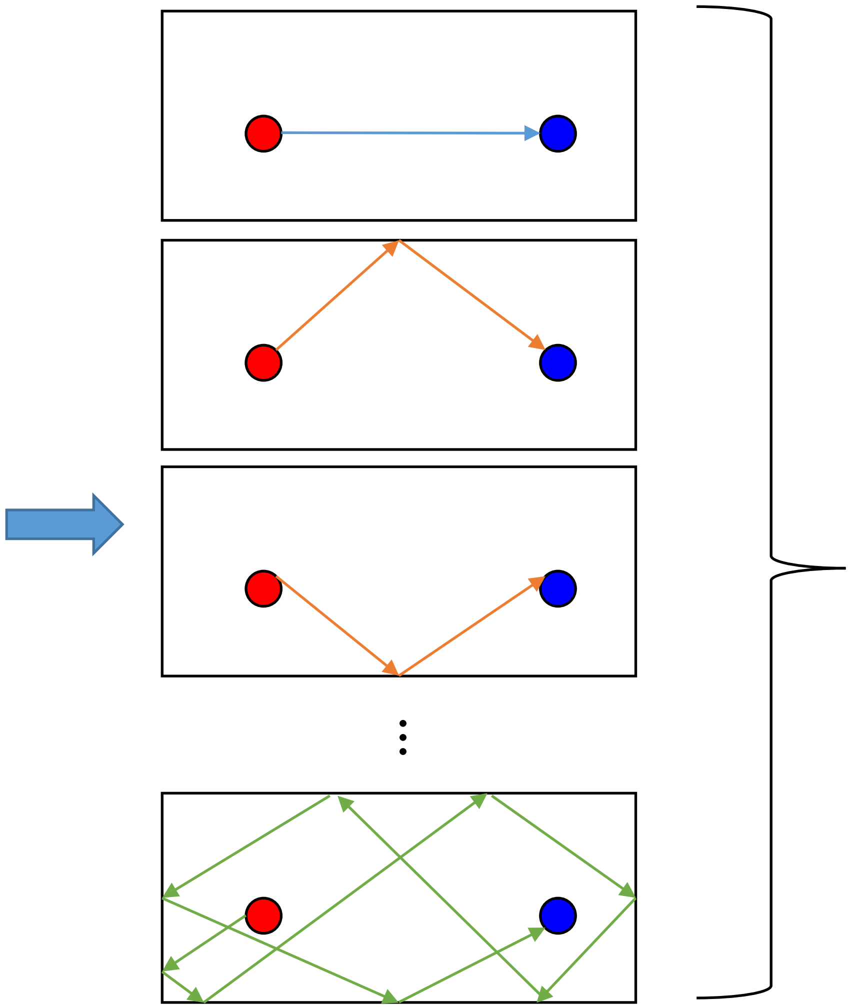


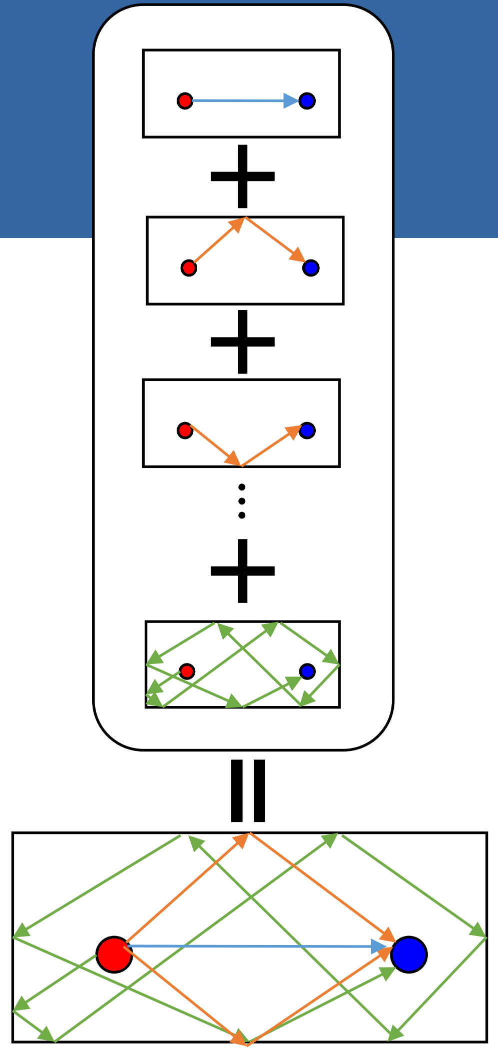
실제로는 여러 경로가 하나의 공간에서 동시에 진행

Sound Propagation System

- Sound Propagation System





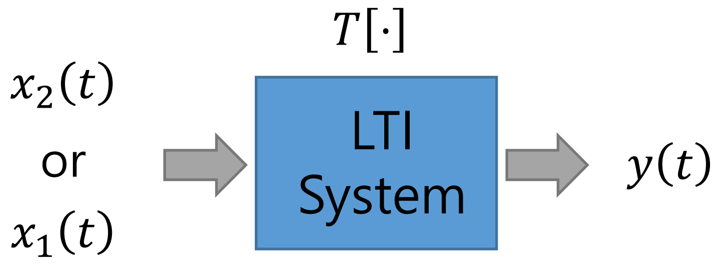


실제로는 여러 경로가 하나의 공간에서 동시에 진행

Sound Propagation System: LTI system

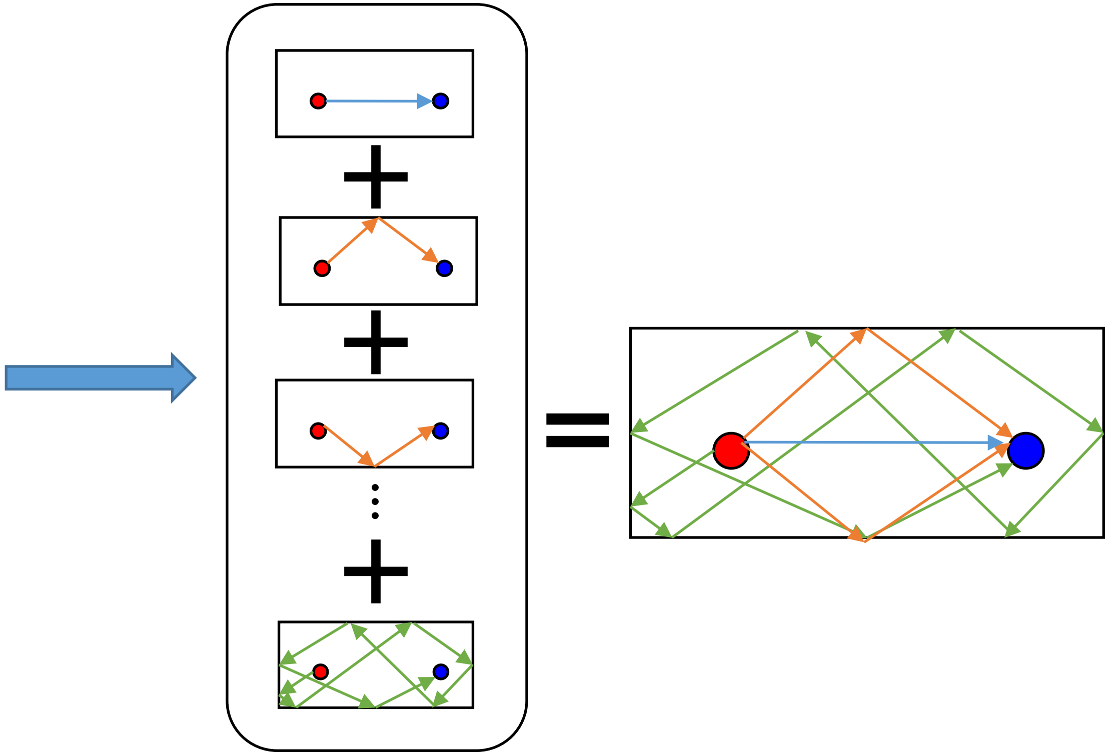
- LTI (Linear Time-Invariant) system
  : Linearity와 Time invariance를 만족

  - Linearity
    - Additivity: $T[x_1(t) + x_2(t)] = T[x_1(t)] + T[x_2(t)]$
    - scaling: $T[ax(t)] = aT[x(t)]$
  - Time invariance: $x(t - t_0) \rightarrow y(t - t_0)$



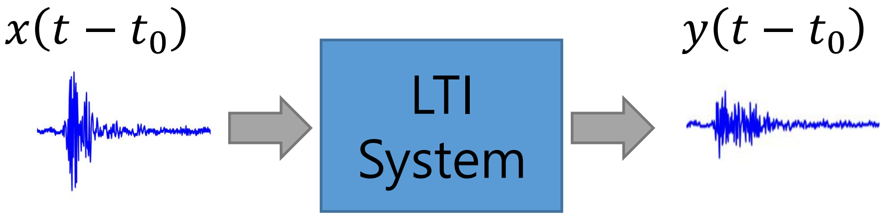
Sound Propagation System: LTI system

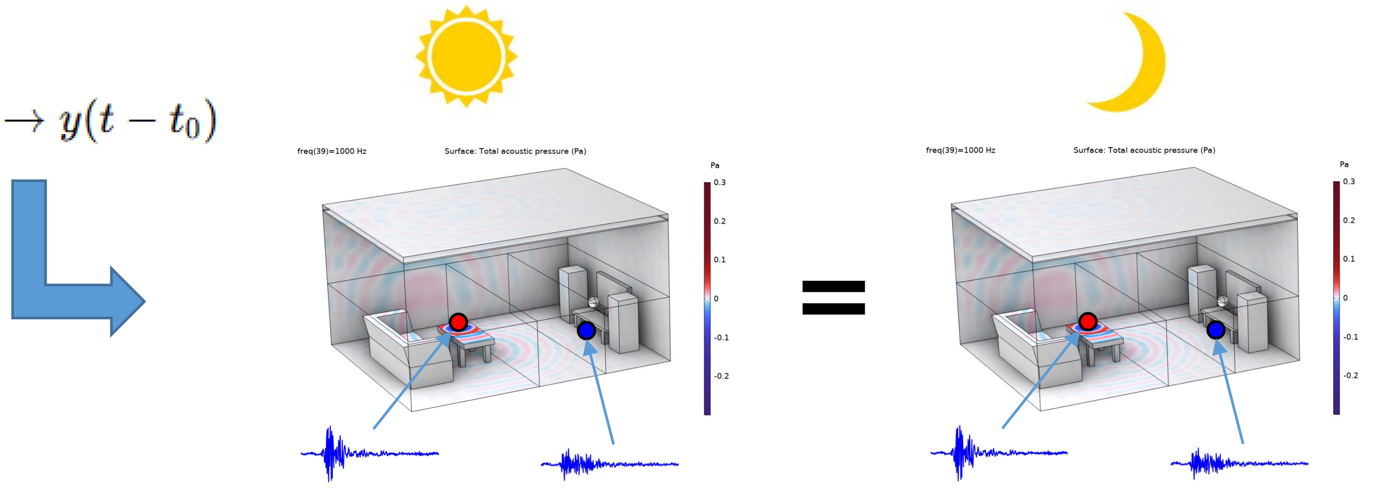
- LTI (Linear Time-Invariant) system
  : Linearity와 Time invariance를 만족
  - Linearity
    - Additivity: $T[x_1(t)+x_2(t)] = T[x_1(t)] + T[x_2(t)]$
    - scaling: $T[ax(t)] = aT[x(t)]$
  - Time invariance: $x(t-t_0) \rightarrow y(t-t_0)$



Sound Propagation System: LTI system

- LTI (Linear Time-Invariant) system
  : Linearity와 Time invariance를 만족
  - Linearity
    - Additivity: $T[x_1(t) + x_2(t)] = T[x_1(t)] + T[x_2(t)]$
    - scaling: $T[ax(t)] = aT[x(t)]$
  - Time invariance: $x(t - t_0) \rightarrow y(t - t_0)$



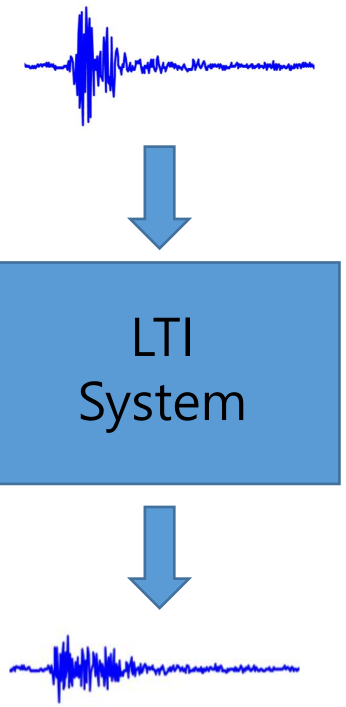


Sound Propagation System: LTI system

- LTI (Linear Time-Invariant) system
  : Linearity와 Time invariance를 만족

  - Linearity
    - Additivity: $T[x_1(t) + x_2(t)] = T[x_1(t)] + T[x_2(t)]$
    - scaling: $T[ax(t)] = aT[x(t)]$
  - Time invariance: $x(t - t_0) \rightarrow y(t - t_0)$

Impulse Response



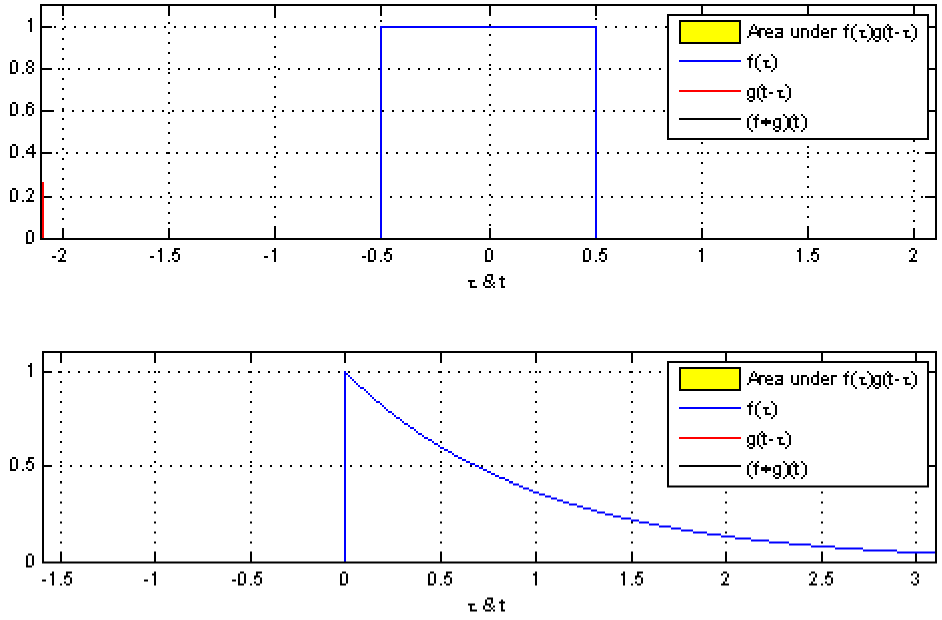
Output은 Input과 Impulse Response
의 Convolution으로 정의

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau = x(t) * h(t)$$

Sound Propagation System: LTI system

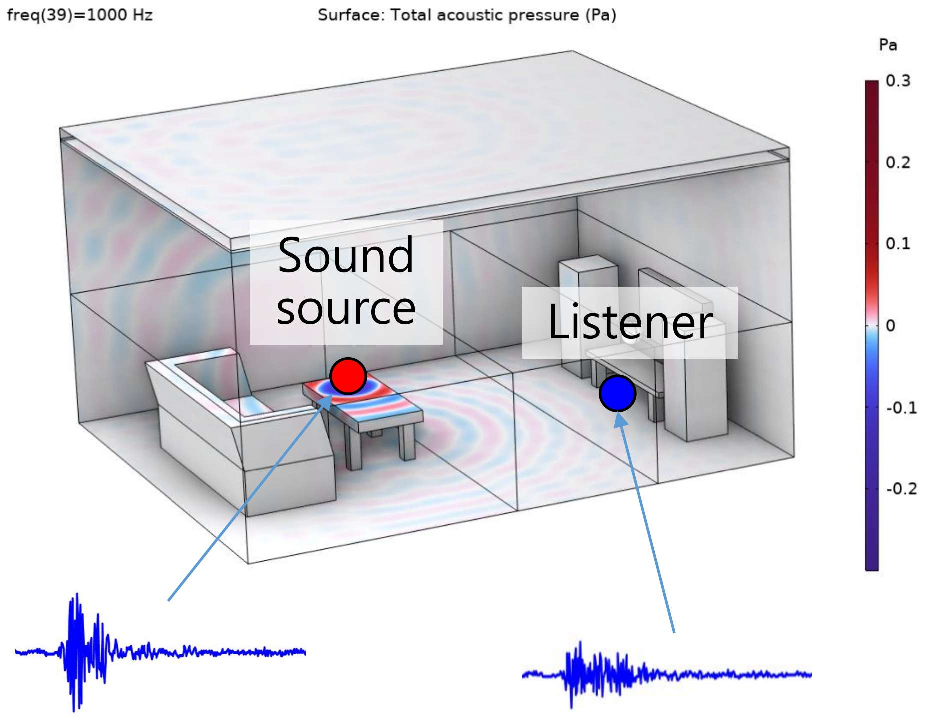
- Continuous-Time Convolution:
  - $x * h = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$

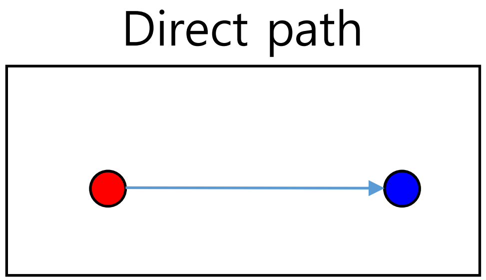
Example of Convolution

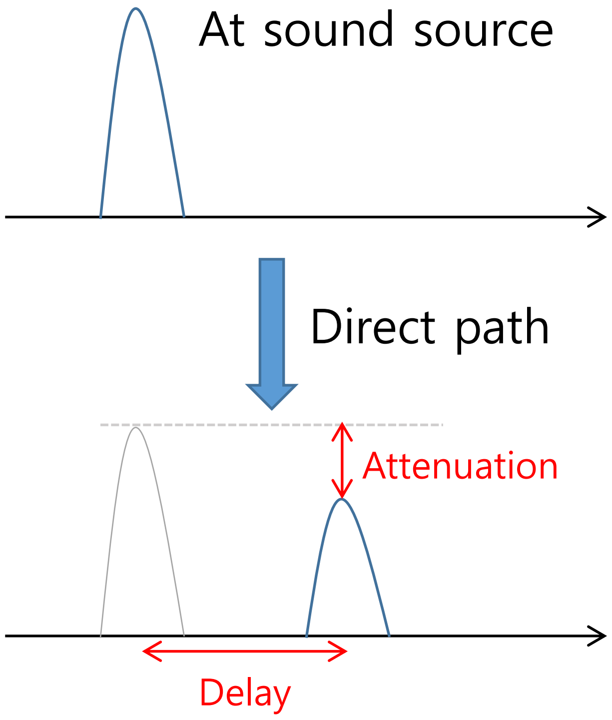


Sound Propagation System: Impulse Response

- Sound Propagation System
  - Estimating Impulse response

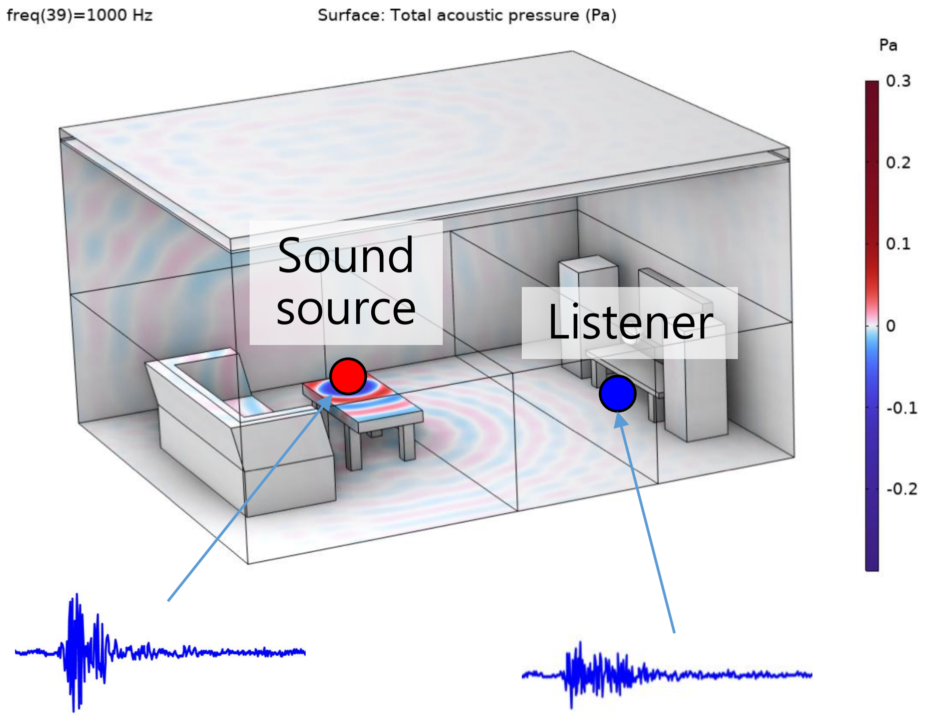


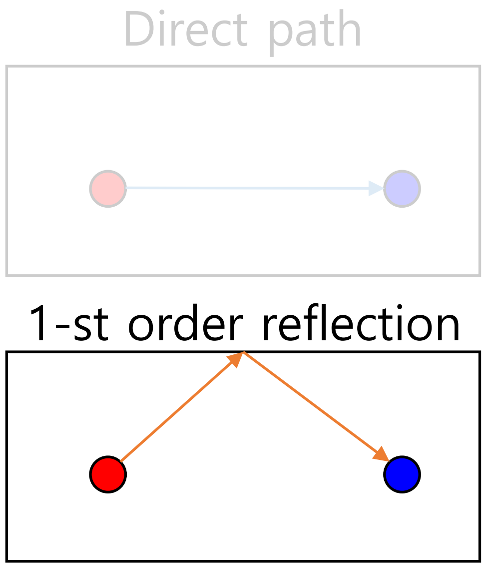


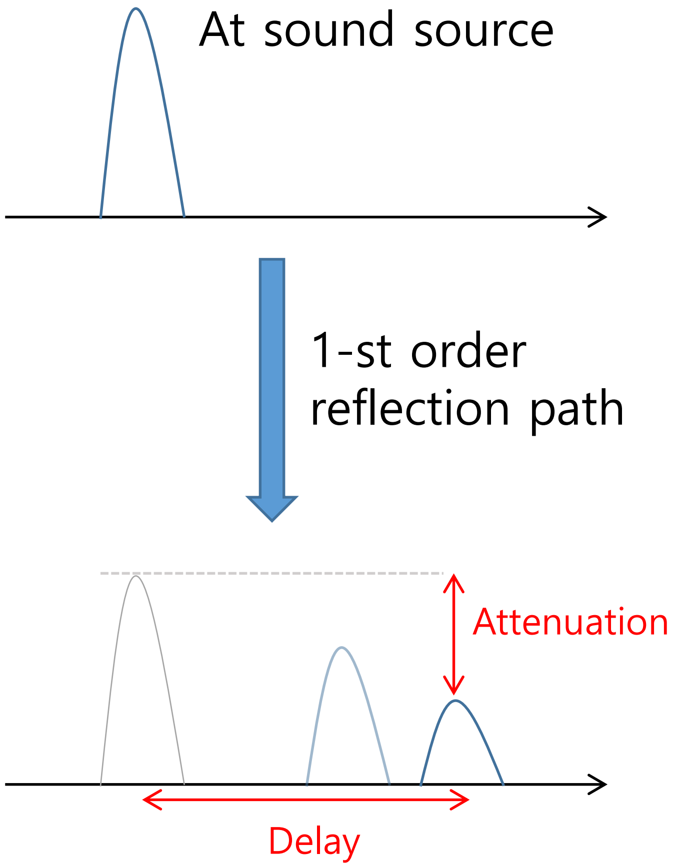


Sound Propagation System: Impulse Response

- Sound Propagation System
  - Estimating Impulse response



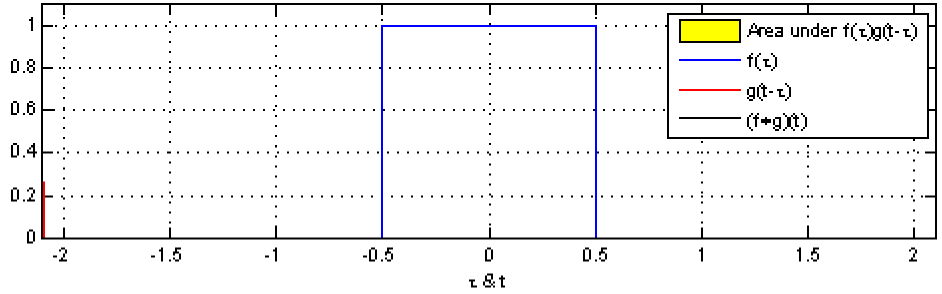


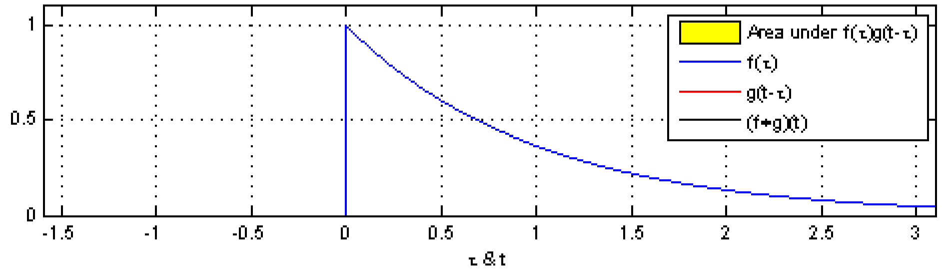


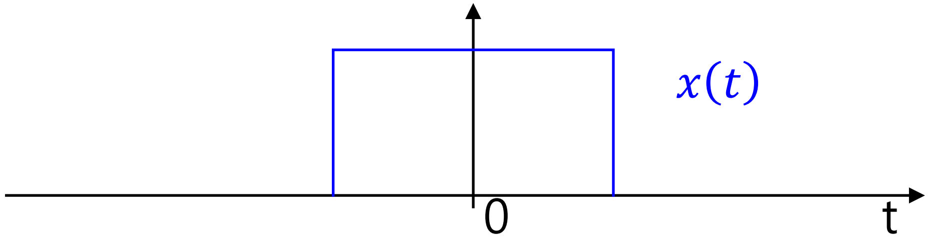
Sound Propagation System: Impulse Response

- Continuous-Time Convolution:
  - $x * h = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$

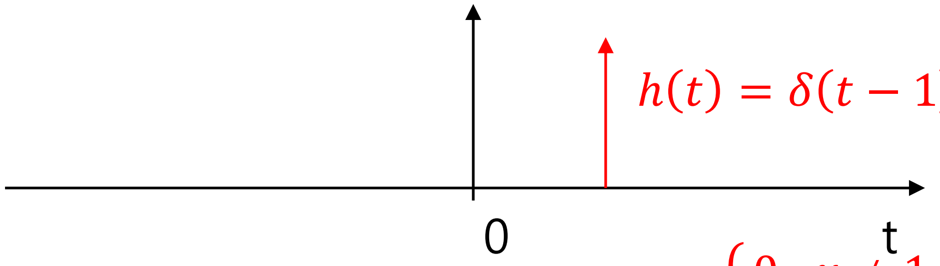
Example of Convolution







$x(t)$

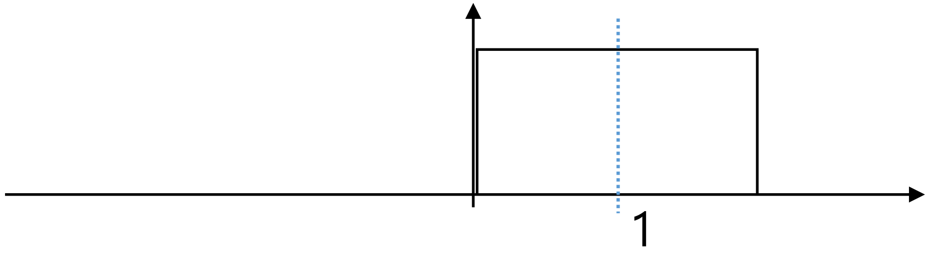


$h(t) = \delta(t-1)$

Dirac delta function $\delta(t-1) = \begin{cases} 0, & x \neq 1 \\ \infty, & x = 1 \end{cases}$

$$\int_{-\infty}^{\infty} \delta(t-1)dt = 1$$

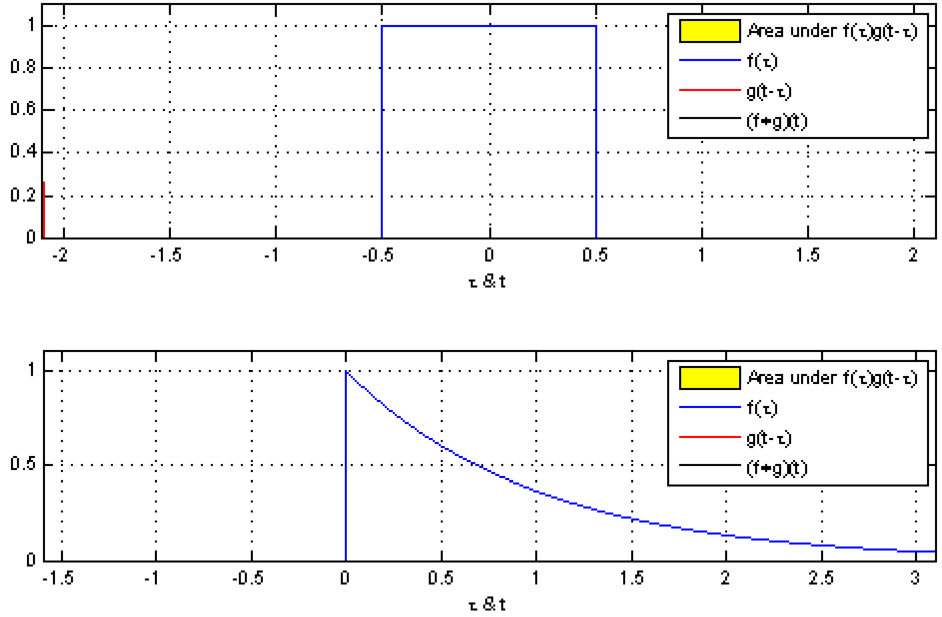
$(f * g)(t)?$

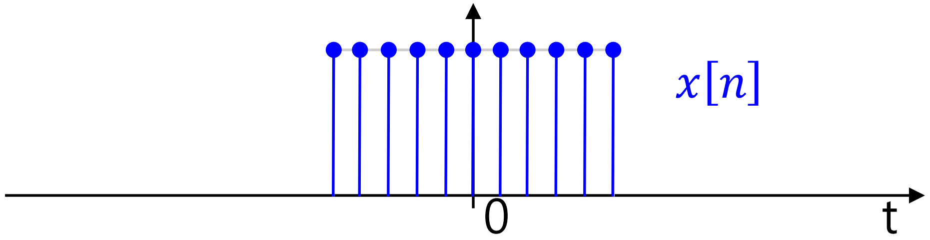


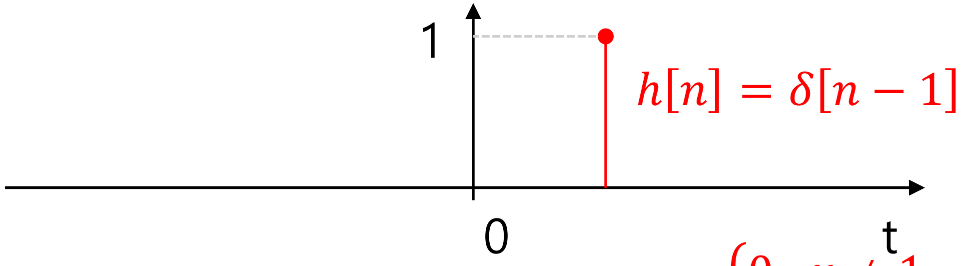
Sound Propagation System: Impulse Response

- Discrete-Time Convolution:
  - $(x * h)[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$

Example of Convolution

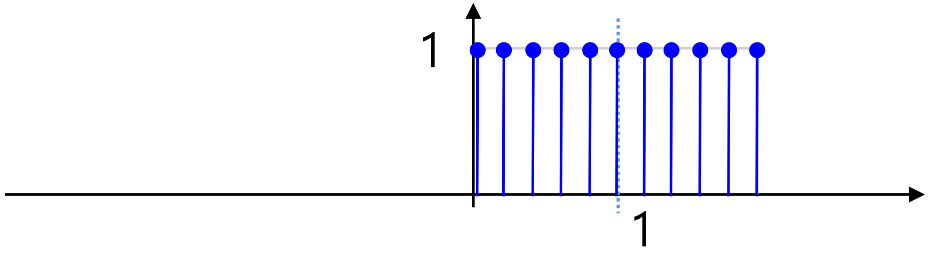






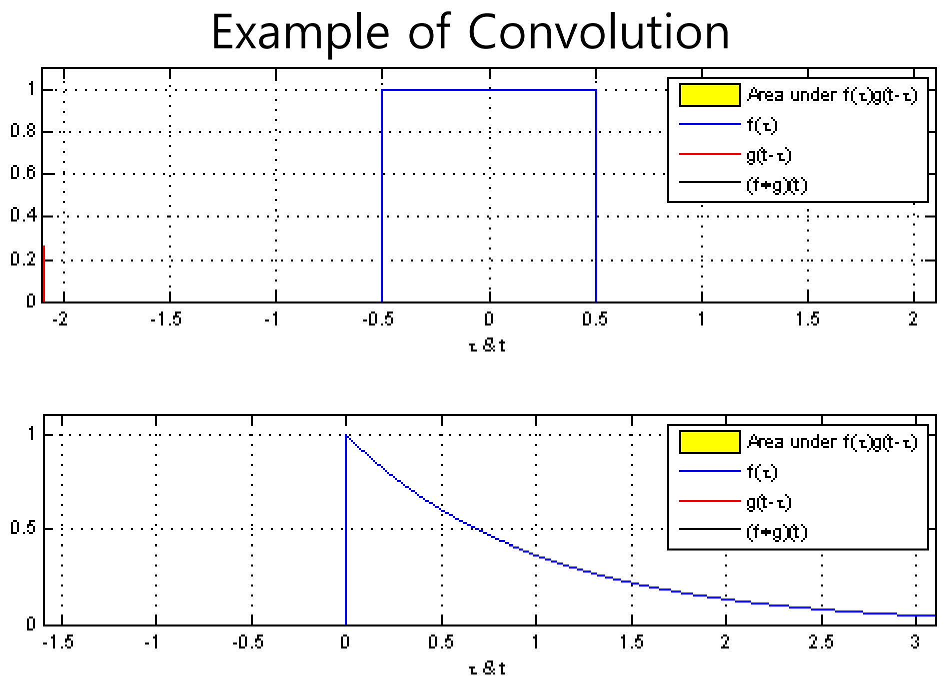
Dirac delta function $\delta[n-1] = \begin{cases} 0, & x \neq 1 \\ 1, & x = 1 \end{cases}$

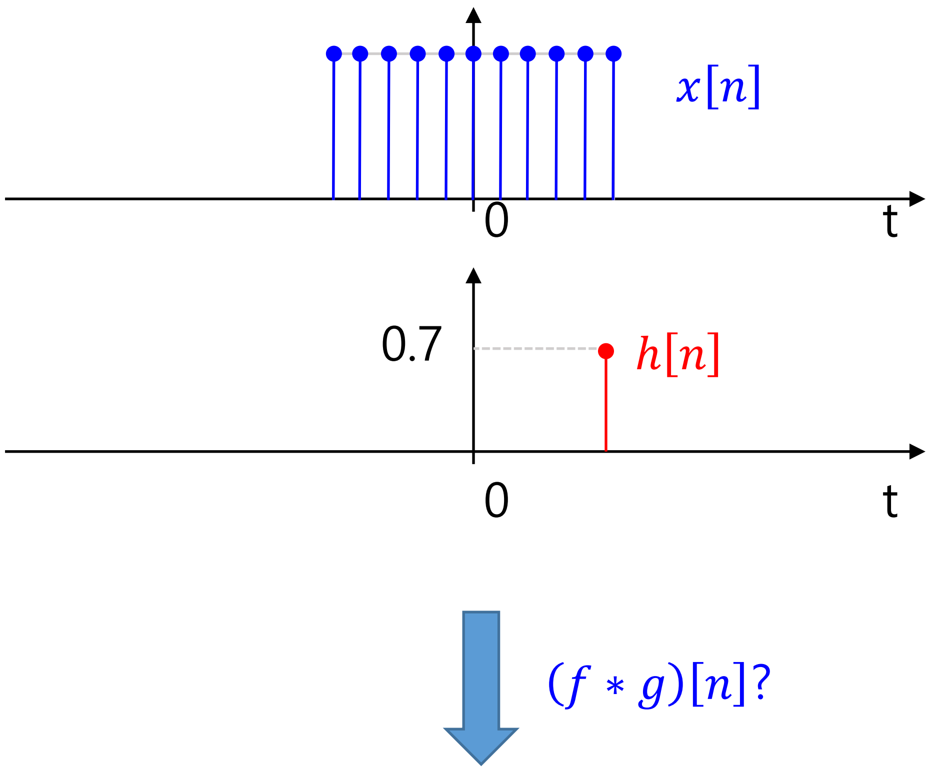
$(f * g)[n]?$



Sound Propagation System: Impulse Response

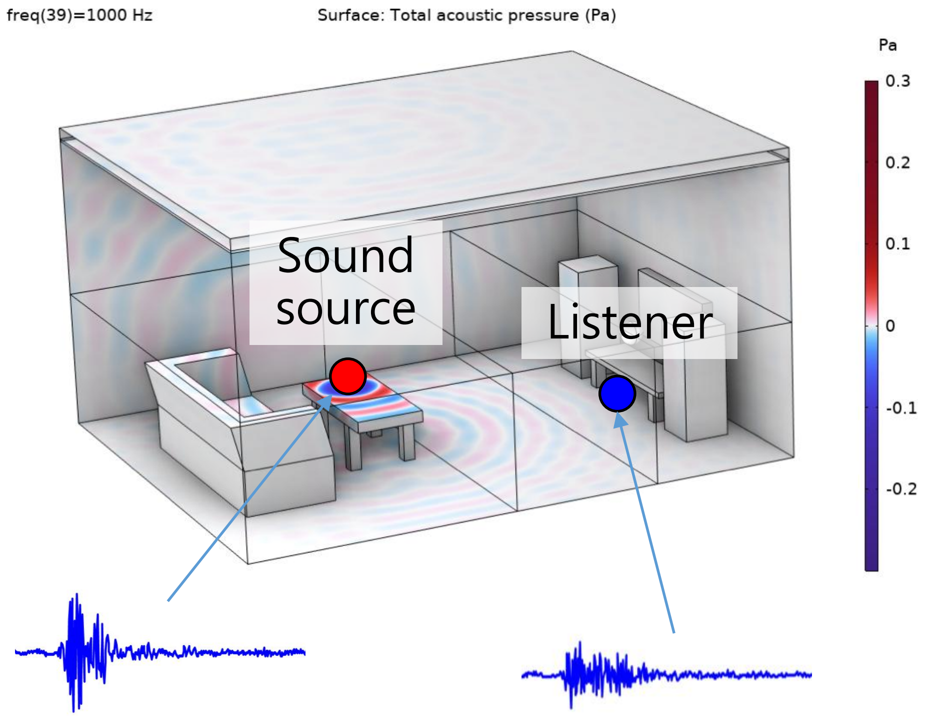
- Discrete-Time Convolution:
  - $(x * h)[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$

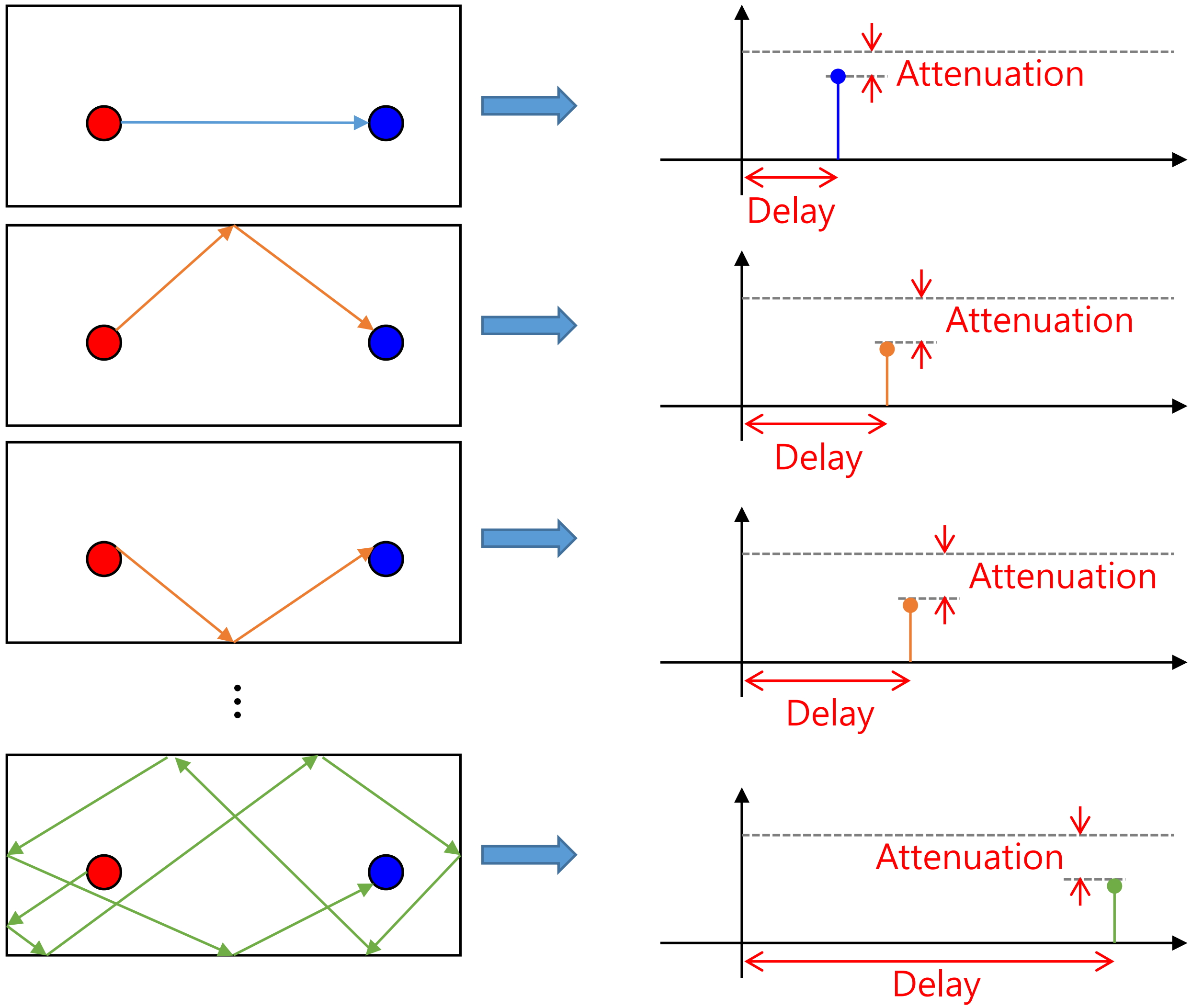




Sound Propagation System: Impulse Response

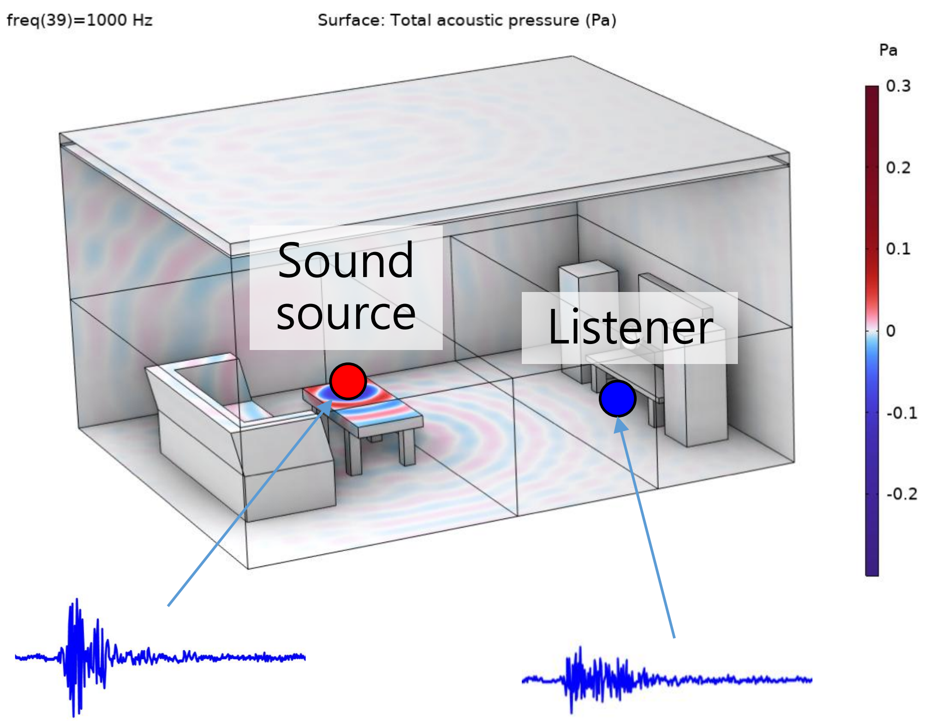
- Sound Propagation System
  - Estimating Impulse response

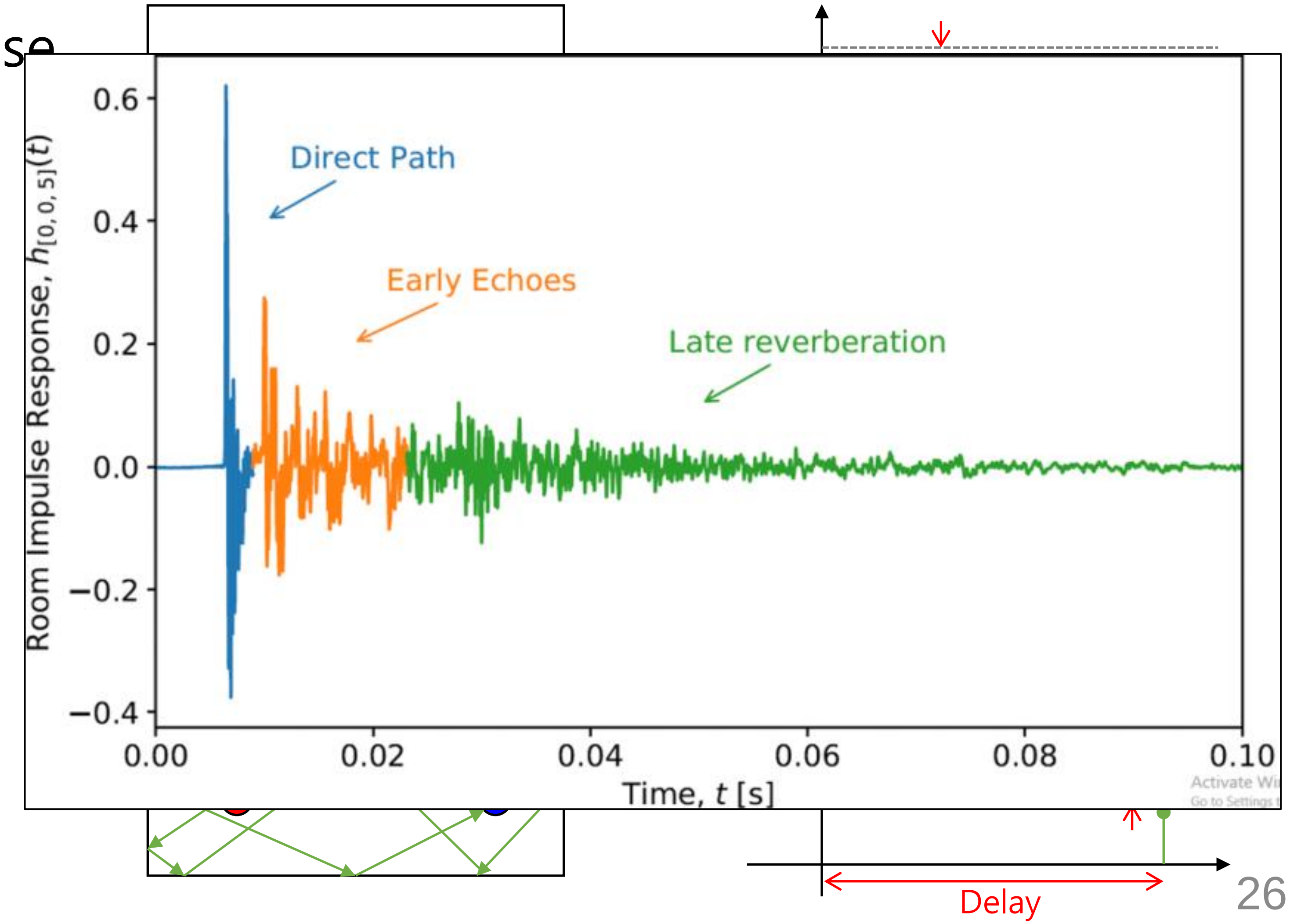




Sound Propagation System: Impulse Response

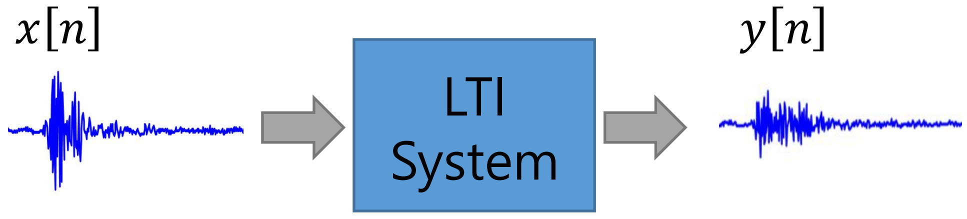
- Sound Propagation System
  - Estimating Impulse response





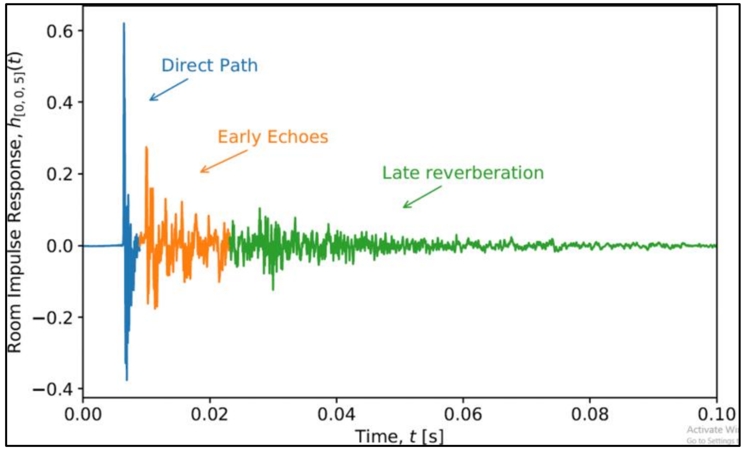
Sound Propagation System: Impulse Response

- Sound Propagation System
  - Estimating Impulse response

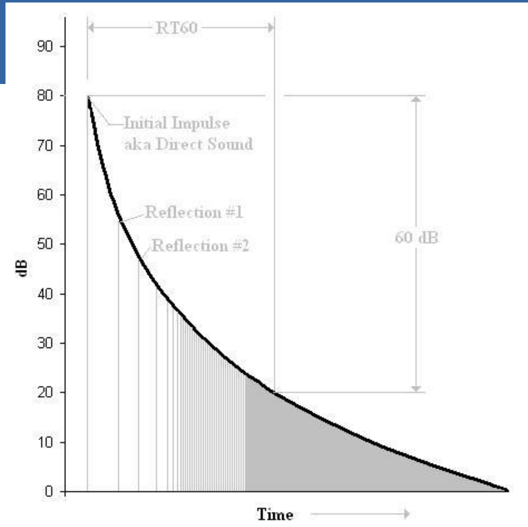


$$y[n] = (x * h)[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

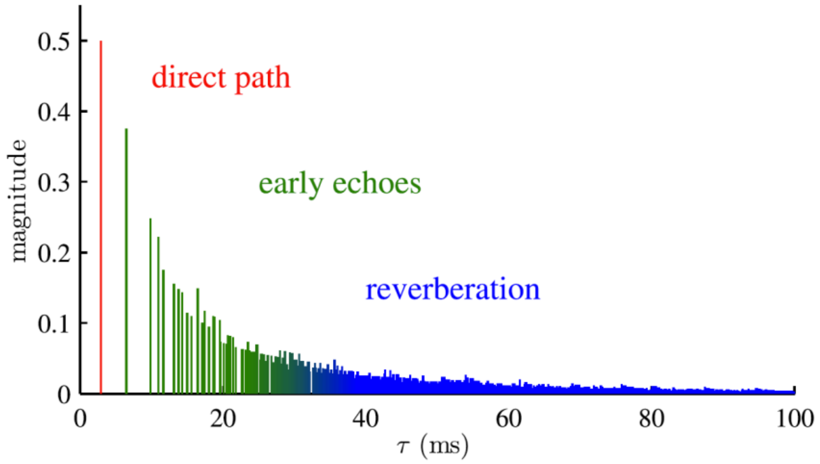
$h[n]$:

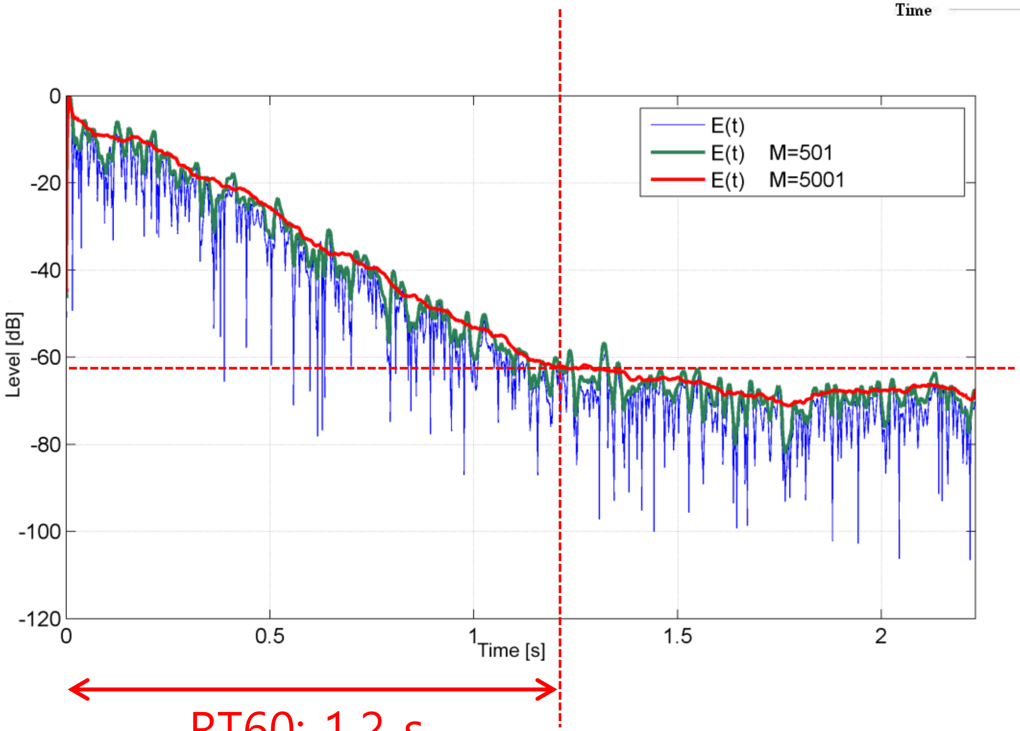


Sound Propagation System: Reverberation



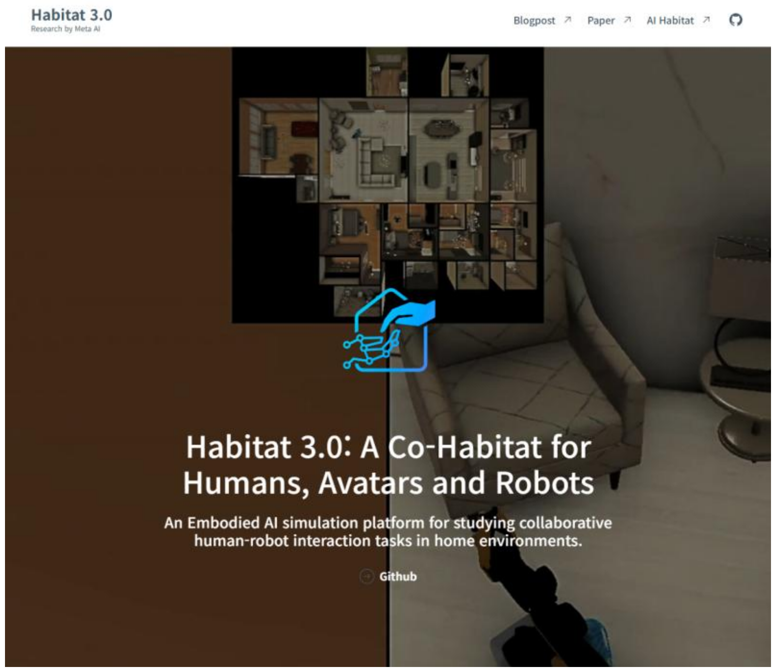
- Reverberation Time (RT60)
    - 소리의 울림이 큰 공간에서는 Reverberation Time이 증가
    - 소리의 울림이 작은 공간에서는 Reverberation Time이 감소
    - **RT60: Sound Pressure Level (SPL)이 60dB 만큼 감쇄되는 데 걸리는 시간**



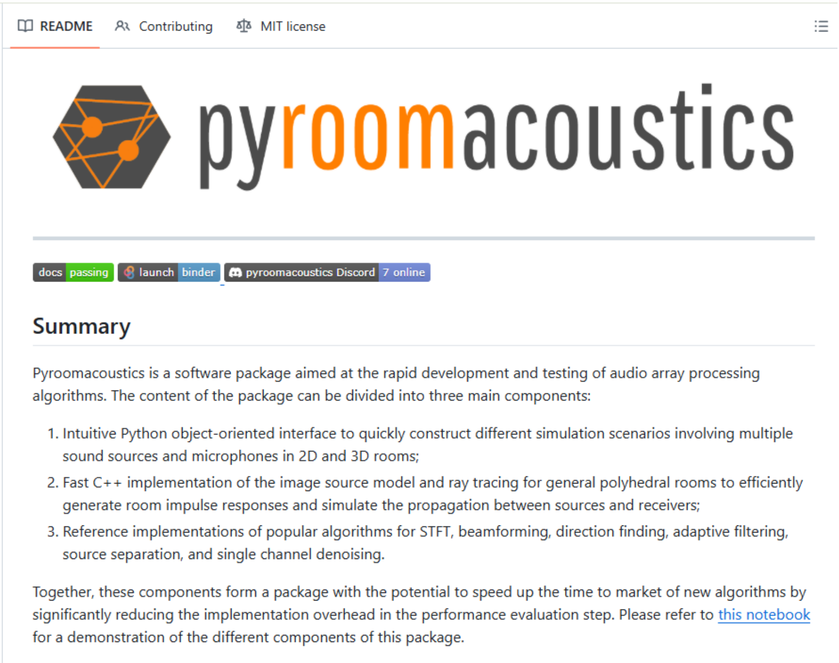


Sound Propagation System: Impulse Response

- Sound Simulation (Python)
  - Habitat2.0 – SoundSpace2.0 (META)
  - Pyroomacoustics



Habitat2.0



Summary

Pyroomacoustics is a software package aimed at the rapid development and testing of audio array processing algorithms. The content of the package can be divided into three main components:

1. Intuitive Python object-oriented interface to quickly construct different simulation scenarios involving multiple sound sources and microphones in 2D and 3D rooms;
2. Fast C++ implementation of the image source model and ray tracing for general polyhedral rooms to efficiently generate room impulse responses and simulate the propagation between sources and receivers;
3. Reference implementations of popular algorithms for STFT, beamforming, direction finding, adaptive filtering, source separation, and single channel denoising.

Together, these components form a package with the potential to speed up the time to market of new algorithms by significantly reducing the implementation overhead in the performance evaluation step. Please refer to this notebook for a demonstration of the different components of this package.

Pyroomacoustics