Youtube Video

Generation of Corrective Impulse Response for Spatial Reconstruction and Real-time Application in Performance

Taebin Yoo 2025. 09. 30

1. Problem Statement

An **Impulse Response (IR)** is a system response that represents how a space reacts to sound. In performance environments, convolution reverb is typically applied by using the IR of a target venue as it is. However, when the same IR is applied to venues with different acoustic characteristics, listeners do not perceive them as the same space. For instance, convolving a "Concert Hall A" IR in a small jazz club versus a large gymnasium results in vastly different auditory experiences. This is because the inherent reverberation, early reflections, and frequency response of the current space are not accounted for.

This project aims to quantitatively analyze the differences between the **current space (My Hall)** and the **target space (Target Hall)** and generate a **Corrective IR** to achieve consistent spatial reconstruction in any performance environment.

2. Concept

Traditional Method
Input Signal * Target Hall IR

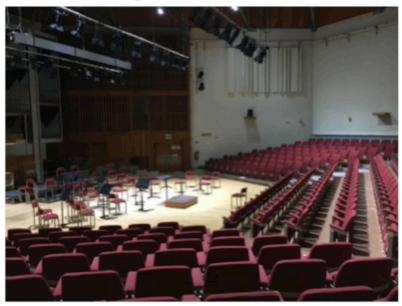
Proposed Method
Input Signal * **Corrective IR (Generated via Python)** ≈ Target Hall IR

Formula
Corrective IR (Generated via Python) = Target Hall IR / My Hall IR

My Hall Venue

Location : DIMA Concert Hall, South Korea
Purpose : Live Band Concerts
Seating Capacity : 200
T30 (500Hz) : 0.39s

Target Hall Venue



Location : Jack Lyons Concert Hall, University of York (UK)
Purpose : Classic Concerts
Seating Capacity : 350
T30 (500Hz) : 1.89s

First, align and tune the speaker system accordingly.
Then, capture the impulse responses (IR) from each speaker using a stage rig microphone.
Compare the impulse responses of the two halls and generate correction filters using a Python script.
Analyze the acoustic coherence across frequencies — the coherence was verified to reach 1.0.
Finally, convolve and apply the correction filters to each speaker channel in real time using Max/MSP.

Youtube Video

3. Experimental Environment and Data Collection

IR Capture Process:

- Surround speaker system simulation, alignment, and tuning.
- Rigging of AB Stereo microphones above the stage.
- Capturing IR for each individual surround speaker channel.
- Standardization of sample rate: 48 kHz.

4. Corrective IR Generation Algorithm (Python Script Summary)

1. Alignment and Pre-processing
- Time-alignment based on direct sound (peak/ cross-correlation), polarity verification, and gating.

2. Band-limiting
- Applying weight to the 125 Hz – 4 kHz range, focusing on the perceptually significant frequency bands.

3. Frequency Domain Ratio Calculation
- Ideal relationship defined as F(ω)=B(ω)/A(ω); applied regularization and smoothing to prevent divergence.

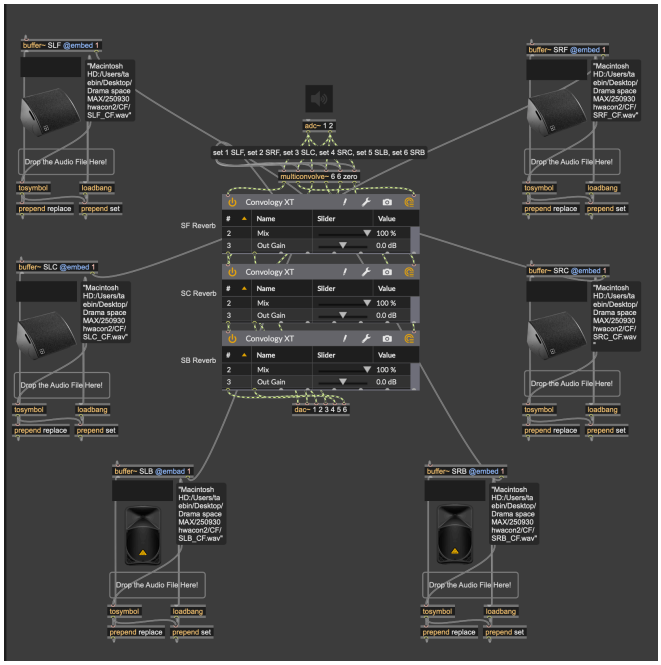
4. Separation of Early/Late Reflections
- Early (0–40 ms) to preserve imaging; Late (60–200 ms) to prevent over-correction.

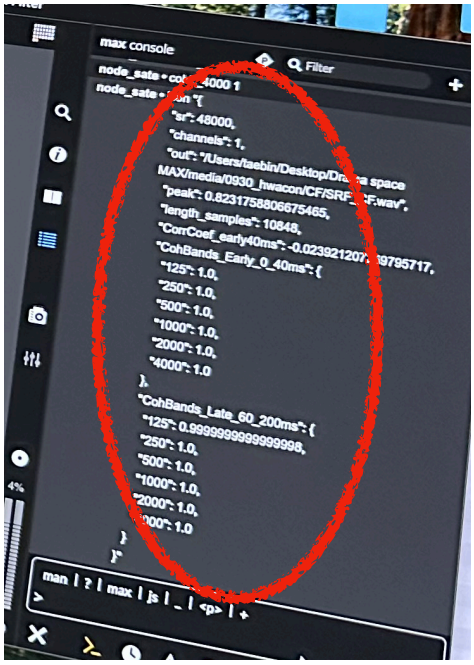
5. Time Domain FIR Generation
- IFFT → Windowing → Length and peak control.

```
const args = [
  String(script),
  "--club", String(club),
  "--hall", String(hall),
  "--out",  String(outPath),
  "--mode","mono",
  "--band_low","125","--band_high","4000",
  "--transition_hz","900",
  "--reg_db","-32","--smooth_bins","121",
  "--gate_db","-33","--gate_pre_ms","42","--gate_tail_ms","16","--gate_fade_ms","5",
  "--early_ms","40","--xover_ms","16","--late_ms","200",
  "--reg_db_late","-28","--smooth_bins_late","129",
  "--late_phase","keep",
  "--final_align","1",
  "--peak_limit_ir","0.95",
  "--export_full","1"
];

const child = spawn(String(py), args, { shell:false });
let out = "", err = "";

child.stdout.on("data", d => out += d.toString());
child.stderr.on("data", d => err += d.toString());

child.on("close", code => {
  const text = (out + "\n" + err).trim();
  if (code !== 0 || /^usage:/mi.test(text)) {
    Max.outlet(["error", text || `python exited ${code}`]);
    return;
  }
  try {
    const j = JSON.parse(text);
    if (j && j.error){ Max.outlet(["error", text]); return; }

    Max.outlet(["done", outPath]);  // 최종 파일 경

    if (j.CorrCoef_early40ms !== undefined)
      Max.outlet(["corrcoef", j.CorrCoef_early40ms]);

    const bands = ["125","250","500","1000","2000","4000"];
    if (j.CohBands_Early_0_40ms)
      for (const k of bands)
        if (j.CohBands_Early_0_40ms[k] !== undefined)
          Max.outlet([`cohE_${k}`, j.CohBands_Early_0_40ms[k]]);
    if (j.CohBands_Late_60_200ms)
      for (const k of bands)
        if (j.CohBands_Late_60_200ms[k] !== undefined)
          Max.outlet([`cohL_${k}`, j.CohBands_Late_60_200ms[k]]);

    Max.outlet(["json", text]);
  } catch(e){
    Max.outlet(["error", "JSON parse failed\n"+text]);
  }
```

5. Real-time Application System (Max/MSP)

The generated Corrective IR was implemented into a **Max/MSP-based real-time convolution processor**. By applying individual Corrective IRs independently to each surround speaker channel, the system was designed to enable consistent spatial reconstruction in real-time during a performance.





6. Results and Verification

By analyzing the frequency and phase response in the early reflection period (0–40 ms)—which determines the acoustic color of a space—the Coherence (correlation coefficient) reached 1.0, verifying a near-perfect match with the target hall.

This project approaches performance sound not merely as the application of new effects, but as an engineering system capable of measuring, analyzing, and correcting the architectural acoustic characteristics of different spaces. It is significant as a case where field audio experience was successfully expanded into a structured research problem.