Ciaran McKay

Computational Sound: COMS 3430

Blog Post: Farnell synthesis - Brownian Noise and Rainfall

In this assignment, I tackled the task of simulating two distinct sounds from nature: the gentle flow of a babbling brook and the soothing pattern of rainfall. My tool of choice was the WebAudio API, which is essentially a Swiss Army knife for web-based audio processing and synthesis. The challenge was not just about technical execution but also about capturing the essence and tranquility of these sounds that many of us find calming.

The Goal

The primary goal was to create a continuous sound that mimics a babbling brook. This sound has a constant, soothing presence that I wanted to replicate digitally. The secondary goal was to add the sound of rainfall. Rain sounds are more sporadic and softer compared to the brook, adding a layer of complexity to the implementation. The task required a mix of creativity, to imagine how these sounds interact in nature and experimentation with sound outputs. The process was a deep dive into the nuances of natural sounds, experimenting with different techniques to see what worked best in recreating these auditory experiences.

Breakdown

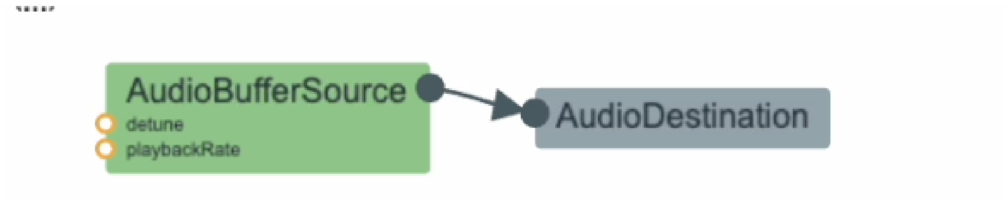
Babbling Brook

For the brook, I used brown noise generation. Brown noise, or Brownian noise, has a strong emphasis on lower frequencies, providing a deep, rich texture reminiscent of flowing water. The code snippet below illustrates the initial setup:

```
var bufferSize = 10 * audioCtx.sampleRate,
    noiseBuffer = audioCtx.createBuffer(1, bufferSize, audioCtx.sampleRate),
    output = noiseBuffer.getChannelData(0);
var lastOut = 0.0;
for (var i = 0; i < bufferSize; i++) {
    var brown = Math.random() * 2 - 1;
    output[i] = (lastOut + (0.02 * brown)) / 1.02;
    lastOut = output[i];
    output[i] *= 3.5; }
```

This code generates a buffer filled with brown noise, achieving the brook's continuous flow sound.

Diagram from WebAudio visualizer:

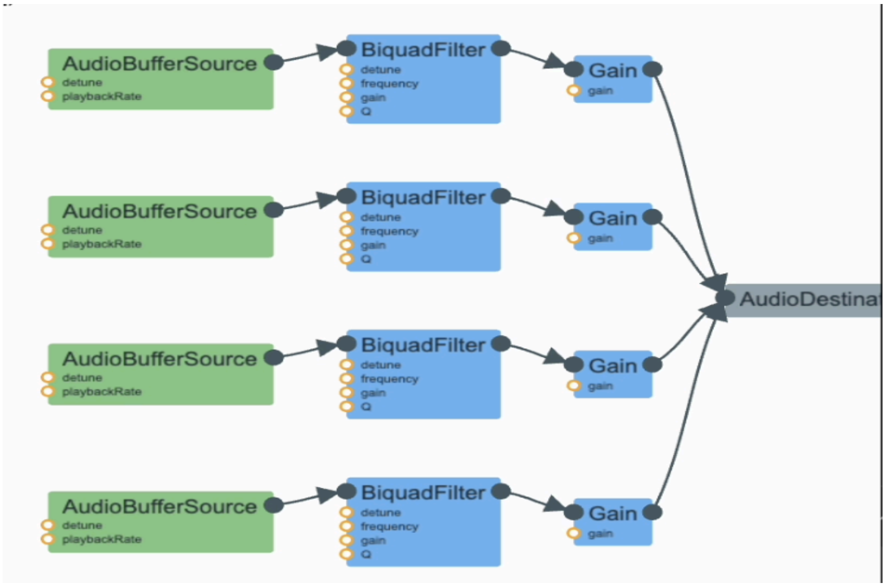


Rainfall

The rainfall effect – inspired from the Farnell book – was created using filtered white noise bursts to simulate raindrops. Each "drop" is a brief burst of noise passed through a bandpass filter, shaping the sound to mimic the impact of a raindrop. This approach allowed for a realistic variance in the raindrop sounds. The key part of the code for generating raindrops is shown below:

```
function createRaindrop() {
    var bufferSize = audioCtx.sampleRate * 0.2;
    var noiseBuffer = audioCtx.createBuffer(1, bufferSize, audioCtx.sampleRate);
    var output = noiseBuffer.getChannelData(0);
    for (var i = 0; i < bufferSize; i++) {
        output[i] = Math.random() * 2 - 1;   }}
```

Diagram from WebAudio visualizer:



The synthesis techniques employed for this project included noise generation and filtering. For the brook, brown noise was the foundation, due to its low-frequency profile. For the rainfall, white noise through a bandpass filter created varied raindrop sounds. These choices were driven by the need to simulate natural sound characteristics accurately.

Overlaying the rain on the brook required careful consideration of volume and density to maintain a balance that reflects a natural setting. Experimenting with different filter frequencies, noise types, and gain levels was crucial in achieving a result that felt both organic and immersive.

This project was a good learning experience in sound synthesis and the capabilities of the WebAudio API. It challenged me to think critically about how to decompose complex natural sounds into simpler, synthesizable components. The choice of synthesis techniques was guided by the inherent qualities of the sounds I aimed to replicate, leading to a deeper understanding of sound properties and how they can be manipulated to recreate the essence of natural environments digitally.