Analysis Of Synthesizing Natural Sounds Based on Nyquist

Yue Yu

Basis International School, Shenzhen 518000, China

yue.yu11987@basischina.com

Abstract. As a matter of fact, many captivating sounds emanate from the natural world, yet their inherent dynamism presents a difficult challenge for replication by existence instruments in recent years. For this reason, computers rarely manage to faithfully replicate these natural sounds. With this in mind, this paper revolves around capturing these beautiful sounds within algorithms, which are later integrated into a demonstrative meditative composition. To be specific, this study will detail demonstrate the methods as well as techniques employed to recreate sounds such as cricket chirps, bird songs, and rain. In detail, the program Nyquist will be introduced in brief as well as applied to achieve the goal. Additionally, a concise overview of the compositional process will be provided and analyzed at the same time. According to the analysis, the current drawbacks for the approaches will be analyzed. In addition, the future prospects will be discussed accordingly. Overall, these results shed light on guiding further exploration of natural sounds synthesizing.

Keywords: Synthesizing Natural Sounds, Nyquist, distinct sound profiles, computational sound generation.

1. Introduction

Since the advent of Digital Signal Processing (DSP), numerous techniques for digitally recreating sounds have emerged. A naive approach to sound replication is through the repetitive playback of a static wavetable. However, this method only performs well at mimicking relatively stable tones, such as those produced by a flute. For more complex sounds such as brass and percussion, the application of FM synthesis can introduce dynamic elements and inharmonic overtones [1]. Some complex methodologies involve dynamic models based on sound attractors, which are even more successful at reproducing complicated sounds that change over time [2]. Nevertheless, it is worth noting that the aforementioned methods are primarily purposed to recreate traditional instrumental sounds and experience difficulty emulating sounds found in nature. Even training-based models, like the dynamic model mentioned earlier, may produce undesirable results when trained with natural sounds due to their unpredictable and intricate nature. This is not to suggest that natural sounds have not been studied before, but it is true that they have rarely been studied in conjunction with digital signal processing. Notably, detailed analyses on animals' sound producing mechanisms have been conducted, as exemplified by the detailed study of frog vocalizations [3]. This study examines the workings of the frog's dilators and constrictors, as well as the dominant frequencies and harmonics in their calls. Analyses like these offer valuable insights, as an understanding of the physical mechanisms behind these sounds could guide the replication of them.

The presented work does not strive to be a one-size-fits-all model capable of handling any sound input. Instead, it is primarily motivated by previous research surrounding the mechanisms behind certain sounds found in nature. The objective is to recreate and refine these mechanisms in code, making them more customizable. Therefore, this study will only select a few sounds for re-creation. While prior work on the digital re-synthesis of natural sounds do exist, a majority of these synthesized sounds are implemented in VST synthesizers, but they are rarely written in code [4]. However, due to limitations of a synth's functionalities, generated results may sound static or inauthentic. Programming these sounds into code allows more control over parameters, opening new possibilities for variation. This study has opted to use Nyquist, a computer language designed by Dannenberg, for its versatility in manipulating sound files and its availability as a free and accessible resource. Additionally, sounds characterized by distinct features, such as bird songs, have been more frequently

researched, while subtler sounds like rain and insect noises remain comparatively under-explored. This paper's purpose is also to shed light on these lesser researched soundscapes.

This study will present three sounds in nature, each with distinct characteristics: cricket chirps, rain, and merlin bird songs. It is believed that these sounds are encountered frequently enough in nature to be recognizable to the general public, and each are distinctive enough such that the analyses would not overlap. In fact, as shown later in this paper, each sound was employed with a unique algorithm. It is hoped that the somewhat limited choice of sounds presented should not limit creativity, but, rather, inspire readers to explore new methods for approximating other sounds.

2. Methods and data

2.1. Synthesizing the Cricket Chirp

The production of a cricket chirp is attributed to the movement of a male cricket's wings. Each wing possesses a scraper and a file, and when these components rub against each other, they generate a short vibration at approximately 2 to 5 kilohertz [5]. One calls this tone a syllable of a chirp. Arising from to the free vibration of wings, there is a higher-frequency component at approximately 3 kHz in each syllable in addition to the 2.3 kHz component, which is the main frequency of the cricket song [6]. On this occasion, this study incorporated a secondary sine tone playing at 3 kHz, albeit at a reduced volume, into each syllable (seen from Fig. 1).

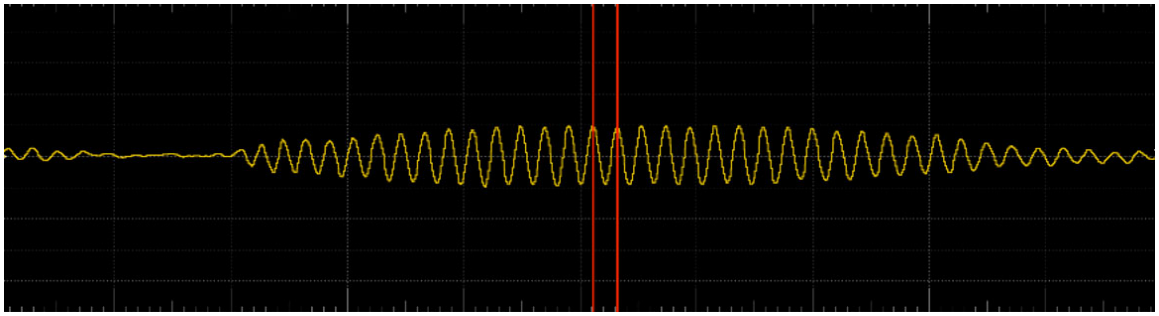

Figure 1. Zoomed in look at a single syllable (Photo/Picture credit: Original).

The waveform of a single cricket chirp reveals a discernible pattern: the repetition of multiple syllables approximately 6 to 9 times. Each syllable is a pure sine tone playing at the frequency of 2300 hertz whose volume dynamics can be approximated by a trapezoid shaped piece-wise-linear function, as outlined as pwl(smoothness, 1, 0.0017 - smoothness, 1, 0.017, 0), where smoothness is a variable controlling the extent to which the syllable fades in and out during playback. For the cricket sample in Fig. 1, for instance, one determined that its smoothness value would be around 0.0006 as shown in Fig. 2.

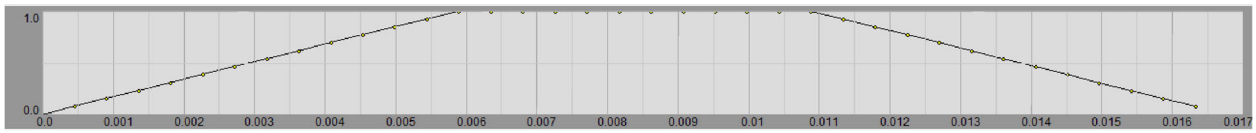

Figure 2. Amplitude envelope modulating the signal where smoothness is set to 0.0006 (Photo/Picture credit: Original).

Subsequently, each syllable was looped 8 times, separated by a short time of t seconds. This generated a single cricket chirp. Adjusting t changes the duty cycle, which is the ratio of sound to silence within a chirp [5]. The perceived "choppiness" of a cricket chirp is directly correlated with its duty cycle, with a longer duration of t resulting in a "choppier" sound. One has found $t = 0.002$ to produce the most natural sounding chirp. Together, multiple chirps separated by a short interval form the cricket sound that people are familiar with.



Figure 3. Frequency spectrum of pink noise (Photo/Picture credit: Original).

2.2. Synthesizing Rain

Conventional models of rain synthesis predominantly rely on complex computations involving real time simulations of environments such as surfaces and water droplets [7]. This study introduces an alternative approach: a model solely based on sample manipulation, rather than physical simulation. This method offers the advantage of resource efficiency while still producing dynamic, non-static results. This study began with the observation that the overall sonic profile of rain can be effectively simulated using pink noise. Nevertheless, some equalizing is applied to better match the frequency distribution of pink noise with that of rain as highshelf(highpass6(lowpass2(pink, 4800), 363), 2800, -15).

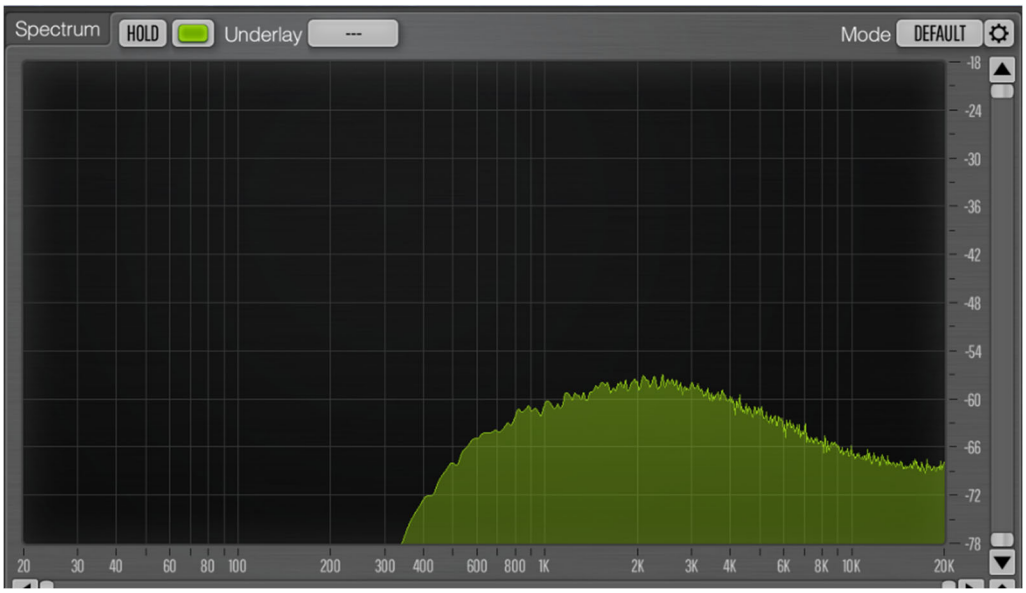

Figure 4. Frequency spectrum of pink noise after processing (Photo/Picture credit: Original).

As illustrated in Fig. 3 and Fig. 4, the upper frequencies were attenuated through a lowpass filter at 4800 hertz, while the lower frequencies below 363 hertz were rolled off through a highpass filter. Additional high end was rolled off starting from 2800 herz using a -15 db highshelf filter. An additional band equalizer was introduced to fine tune the sound and eliminate undesirable frequencies with eq-band (eq-band (pink, 1000, -3, 2), 500, -9, 1.5).

Furthermore, 10% of reverb with a 1 second decay was added, giving a spatial dimension to the rain soundscape. To create movement in the sound, one simulated the effect of wind swirling within the rain. The wind is governed by an intensity variable ranging from 0 to 1. As the intensity increases, a band equalizer boosts a higher frequency while also increasing the decibel boost as eq-band(s, 1200 + 1000 * intensity, intensity * 10, 1.8 - 0.2 * intensity.

Controlling the wind effect can be easily achieved by manually sketching an envelope, where the wind's intensity will follow the y value of the curve over time. Alternatively, the wind effect can be simulated by generating a random series of continuous numbers bounded between 0 and 1. Despite these efforts, the generated sound remains unconvincing. Thus, a secondary sound was incorporated to emulate the sound of droplets striking the ground. The foundation of this sound layer is the sound of the crumpling of a paper bag. To make it blend in with the rain ambiance, filters were applied to shape its frequency distribution. These included a lowpass filter cutting off frequencies above 8000 hertz, a highshelf filter reducing frequencies above 5000 hertz by 15 decibels, and a highpass filter to get rid of unwanted low end below 350 hertz with highpass2(eq-highshelf(lowpass4(a, 8000), 5000, -15), 350). An additional EQ was added to further shape the sound as eq-band (eq-band (s, 1400, -4, 2), 510, -3, 1.5). Subsequently, the amplitude of the signal was modulated with the following envelope, which played at randomized speed from 80~100 hertz with envelope = (1 - pwev(0.02, 0.995, 1, 1, 0.036)). This introduced transients to the sound, creating the illusion of raindrops colliding with the ground. The randomization of the LFO frequency was crucial in introducing variability to the rain sound, preventing it from sounding like a loop being played repetitively. Played together, the pink noise and the paper bag sound produced a remarkably faithful representation of a rain soundscape.

2.3. Synthesizing Bird Songs

The bird song can be split into two components, each a frequency sweep from around 500 to 4000 hz. To generate the first sound component, this study modulated both the frequency and amplitude. A sine wave of approximately 180 to 300 milliseconds duration seemed to work best. The sine wave was then subjected to frequency modulation (FM) by means of a low-frequency oscillator (LFO). The shape of the LFO was that of an inverted U curve. To attain this shape in Nyquist, one subtracted a concave downward curve from 1 as 1 - pwzv(1, (dur / 2), 0, dur, 1). The start and end frequency of the sound was set to be around 500 to 1500 hz, with its peak frequency approximately an octave higher than that the end frequency. In order to avoid unnatural clicks, the amplitude was modified with a trapezoid shaped LFO as pwlv(0, (dur / 8), 1, (dur / 8 * 7), 1, dur, 0) (seen from Fig. 5).

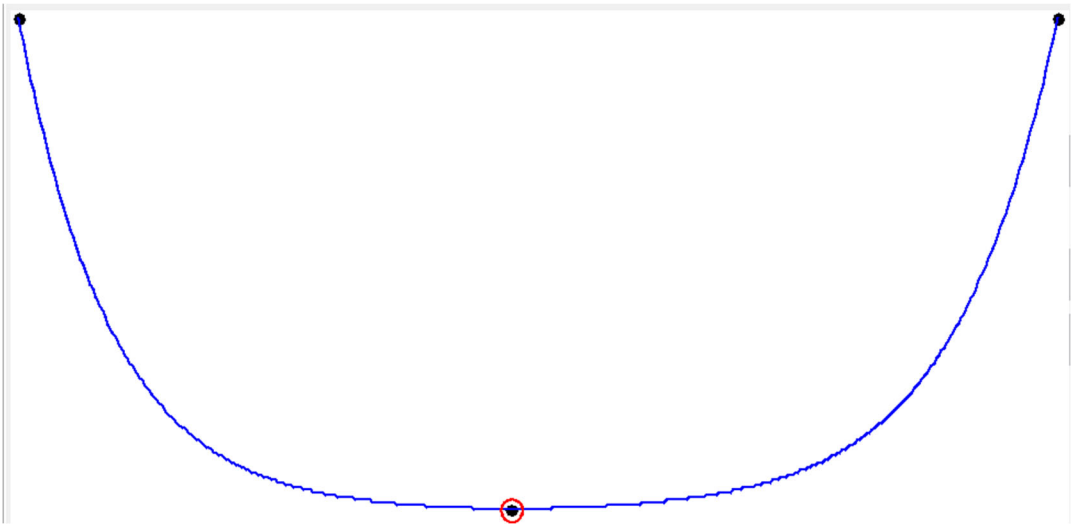

Figure 5. Frequency modulation on the sine wave (Photo/Picture credit: Original).

The generation of the second sound component followed a similar process to that of the swoop sound. The main difference was the shape of the frequency modulator LFO: in the case of the chirp sound, the LFO was an increasing exponential curve as pwzv(0, dur, 1). The start of the LFO was one

octave higher than the start of LFO in the swoop sound, while the end of the LFO was 2 octaves higher than the start. In the end, these two sounds were combined using a seq function.

3. Limitations and Future Outlooks

Although the models have generally produced convincing results, there is still significant room for improvement. The cricket sound consistently produced each syllable at a selected pitch, perfectly separated by a fixed amount of time, which contradicts natural cricket songs, where each syllable typically differs slightly in pitch and timing. Hence, introducing randomization to the LFO or pitch of the cricket song could contribute to a more organic quality. Additionally, one could explore bird songs and new algorithms to generate them. Currently, the approach is centred around a sweeping sine wave, leaving room for further innovation. Furthermore, despite employing various randomization processes in the creation of the rain sound, it may still occasionally appear quite static. This is likely an inherent limitation of the sample manipulation technique when compared to physical models, as the looping of samples cannot be entirely avoided. In the broader context, the study did not provide a universally applicable model capable of approximating any natural sound. Instead, one went for a specialized approach, selecting specific sounds and creating dedicated models for each, thus limiting the model's capacity to produce only those predefined sounds. Future studies may aim to investigate a generalized model that could effectively produce a variety of natural sounds with minimal tinkering needed to achieve a desired result [8-10].

4. Conclusion

To sum up, this paper has presented three distinct sound profiles (those of cricket, rain, and bird songs) implemented in Nyquist code. A key achievement lies in the customizable parameter controls, offering flexibility to fine-tune these sounds according to specific preferences, or to randomize these parameters for an organic quality. The final composition showcases how evoking natural environments through synthetic soundscapes opens up endless possibilities for immersive auditory experiences, underscoring the potential of computational sound generation. In the future, the fusion of creativity and technology will only promise to yield even more fascinating results.

References

[1] G. de Poli, Computer Music Journal **7**, 8–26 (1983).

[2] A. Röbel, Computer Music Journal **25**, 46–61 (2001).

[3] C. GANS, American Zoologist **13**, 1179–1194 (2015),

[4] D. L. Davids, “Recovery effects in binary aluminum alloys,” Ph.D. thesis, Harvard University, 1998.

[5] F. Huber and J. Thorson, Scientific American **253**, 60–73 (1985)

[6] R. O. Stephen and J. C. Hartley, Journal of Experimental Biology **198**, 2139–2152 (1995).

[7] S. J. Miklavcic, A. Zita, and P. Arvidsson, “Computational real-time sound synthesis of rain,” Department of Science and Technology (ITN), Campus Norrköping, Linköping University, 2004.

[8] Q. Zhou, J. Shan, W. Ding, et al, Frontiers in Robotics and AI, **8**, 580080 (2020).

[9] W. Hong, J. Wright, K. Huang and Y. Ma, IEEE Transactions on Image Processing, **15(12)**, 3655-3671 (2006).

[10] B. Mishachandar, and S. Vairamuthu, Applied Acoustics **181**, 108141 (2021).