NIVEAU À BRUIT: SOUND-BASED LEVELING APP

Alexandre D'Hooge

Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France
alexandre.dhooge@univ-lille.fr

ABSTRACT

Niveau À Bruit (Noise Level) is derived from the French term *Niveau À Bulle* (bubble level/spirit level). My sonification choices were mostly driven by personal taste and experience and use both amplitude and frequency signals to help the user level their phone. When performing 1D-leveling, a *tremolo* effect is applied to a cosine signal, and the frequency of the effect is modulated by the angle of the smartphone. When a second dimension is added, an additional cosine signal can be heard and must be tuned towards a perfect octave with respect to the first signal to level the phone.

1. LINK TO APK FILE

The APK file of my implementation is available on Zenodo: `https://doi.org/10.5281/zenodo.8027582`.
My fork of the repository is also made publicly available on GitHub:
`https://github.com/adhooge/sonic-tilt`.

2. INTRODUCTION

My main motivation when developing a new sonification solution was to obtain a sound that is both pleasant to hear and rather simple, to make the leveling task as smooth as possible. When considering 2-dimensional leveling with sound, the first consideration that came to my mind was what independent parameters can be used to alter a sound on two different aspects. Considering the signal processing techniques that are Amplitude Modulation (AM) and Frequency Modulation (FM) lead me to choose *Amplitude* and *Frequency* as the two dimensions for my sonification solution. For this reason, the main technique used for 1D-leveling is a *tremolo* effect applied to a sinusoidal signal, which applies amplitude modulation with a cosine signal of controlled frequency with a unit modulation index. When the leveling is 2-dimensional, an additional signal is added, which frequency must be tuned towards the double of the other signal. Those two signals thus vary in ways that are expected to be sufficiently independent for a user to level both dimensions simultaneously with little to no training.

3. YOUR SONIFICATION

The first sound is a single cosine signal with a frequency of $440\,\mathrm{Hz}$. Its frequency was chosen to be in the working range of a smartphone tweeter, avoiding low frequencies that are harder to hear on such a device while still permitting adding a signal with higher frequency. The signal is a pure sinusoid to avoid additional harmonics that could make the sound too harsh and unpleasant to hear continuously. This sound is applied a *tremolo* effect to periodically mute it according to the angle of the phone: the further away from $0°$, the faster the tremolo effect. This effect is independent of the direction of the tilt, preliminary experiments suggested that it was natural and understandable to tilt the phone towards the slower tremolo. If the tremolo effect slows down then speeds up again, it is sufficient feedback for the user to go back in the other direction. An example of how the waveshape is affected by the tilt angle is shown Figure 1. The advantage of using amplitude modulation is that it can be detected fairly easily by any user because the modulation speed is kept way under $50\,\mathrm{Hz}$ [1, 2]. While it is true that it takes longer to detect the modulation frequency at lower speeds, it seems acceptable because the app usage calls for slow movements that give enough time to hear several modulation cycles.

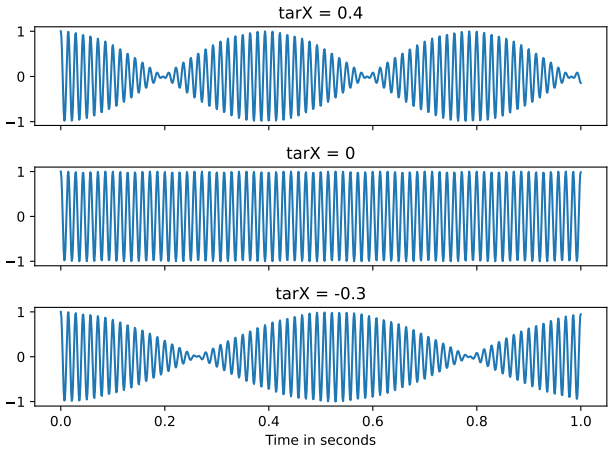

Figure 1: Waveshape of the sinusoidal signal altered by a tremolo effect for different values of `tarX`. The variable `tarX` is obtained by mapping the tilt angle from $[-45°, 45°]$ to $[-0.5, 0.5]$.

To allow the user to perform 2D-leveling, a second sound is added which default frequency is, on the neutral position, the double of the first signal *i.e.* $880\,\mathrm{Hz}$. The frequency of this second sinusoidal signal is modified by the tilt angle. If the tilt angle is below $0°$, the frequency is lower than $880\,\mathrm{Hz}$. Conversely, it is higher if the tilt angle is positive. The fact that positive and negative angles yield different sound modifications seemed necessary in that situation because the complexity of 2D leveling based solely on sound called for additional feedback provided to the user, ob-

tained by this asymetric design. The choice of tuning another signal towards a perfect octave was driven by musical considerations. While absolute pitch perception requires training, relative pitch is usually more easily perceived. In particular, octave perception in based on physical events and *octave equivalence* is a phenomenon observed across cultures [3, 4]. The task of matching the *chroma* of the two signals should thus be natural to most users with little to no training.

Finally, to avoid an overflow of information during 1D-leveling, this second signal is turned off when `tarY` is null. I was not able to find a way to keep the signal playing even when `tarY` is null for 2D-leveling so another way for the user to know that their phone is leveled is to find the spot that mutes the octave sound.

4. IMPLEMENTATION

The implemented solution is minimalistic since the chosen approaches did not call for complex PureData patches. My patch is publicly available on the accompanying code repository. The *tremolo* effect is obtained by generating a cosine wave with a simple `osc~` block and the frequency of that wave is $f_{\text{tremolo}} = 20.\texttt{tarX}$. This signal is used to modulate the amplitude of the base cosine signal y_{base}. The additional octave signal has a frequency of $f_{\text{add}} = 2 \times f_{\text{base}} + \texttt{tarY}$ and is turned off if `tarY` is null. Its amplitude is also divided by half before summing it to the base signal to reduce its perceived loudness on a phone tweeter. Overall, the final output signal is therefore defined as:

$$y_{\text{out}}(t) = \cos(2\pi f_{\text{tremolo}} t).\cos(2\pi f_{\text{base}} t) + \frac{\delta_{\texttt{tarY},0}}{2}\cos(2\pi f_{\text{add}} t)$$

where t is the time variable and δ is the *Kronecker delta*.

5. ACKNOWLEDGMENT

This work is built upon the open source project *Sonic Tilt* that was first introduced as Tiltification [5].

6. REFERENCES

[1] P. Menell, K. I. McAnally, and J. F. Stein, "Psychophysical Sensitivity and Physiological Response to Amplitude Modulation in Adult Dyslexic Listeners. (cover story)," *Journal of Speech, Language & Hearing Research*, vol. 42, no. 4, pp. 797–803, Aug. 1999, publisher: American Speech-Language-Hearing Association. [Online]. Available: https://doi.org/10.1044/jslhr.4204.797

[2] N. F. Viemeister, "Temporal modulation transfer functions based upon modulation thresholds," *The Journal of the Acoustical Society of America*, vol. 66, no. 5, pp. 1364–1380, Nov. 1979. [Online]. Available: https://doi.org/10.1121/1.383531

[3] E. M. Burns, "Intervals, Scales, and Tuning," in *The Psychology of Music (Second Edition)*, ser. Cognition and Perception, D. Deutsch, Ed. San Diego: Academic Press, Jan. 1999, pp. 215–264. [Online]. Available: https://doi.org/10.1016/B978-012213564-4/50008-1

[4] L. Crickmore, "A Re-Valuation of the Ancient Science of Harmonics," *Psychology of Music*, vol. 31, no. 4, pp. 391–403, Oct. 2003, publisher: SAGE Publications Ltd. [Online]. Available: https://doi.org/10.1177/03057356030314004

[5] M. Asendorf, M. Kienzle, R. Ringe, F. Ahmadi, D. Bhowmik, J. Chen, K. Huynh, S. Kleinert, J. Kruesilp, Y. Lee, X. Wang, W. Luo, N. Jadid, A. Awadin, V. Raval, E. Schade, H. Jaman, K. Sharma, C. Weber, H. Winkler, and T. Ziemer, "Tiltification - An Accessible App to Popularize Sonification," *Proc. 26th International Conference on Auditory Display (ICAD2021), Virtual Conference*, pp. 184–191, June 2021. [Online]. Available: https://doi.org/10.21785/icad2021.025