

Title:

“Esotropio: A Dynamic Sample Based Sequencer for Artificial Soundscapes in Pure Data”

Download Link (*esotropio.pd*):

https://drive.google.com/drive/folders/1M4xUhJ7leEWYc1_6JhbnwZxqndlzdpjJ?usp=sharing

Requires: *Else* and *Cyclone* Pure Data Libraries (included in the above folder)

Video Presentation:

https://youtu.be/S_UnQl_WlcU?si=FNYxUxyEBPqMTo9x

Abstract

This paper introduces Esotropio, a sample sequencer developed in Pure Data as a patch, designed to facilitate music synthesis with soundscapes. The patch utilizes a multi-stage processing chain, allowing the user to load and manipulate eight distinct sound banks. It autonomously sequences and transforms these samples based on frequency analysis and various algorithms. This paper discusses the main features of the patch, the technical challenges encountered during its development and proposes future enhancements to expand its capabilities.

1. Introduction

The creation of Esotropio arises from a long-standing interest in developing automated music synthesis systems that embody the aesthetic sensibilities of their creators while minimizing direct human intervention. This concept aligns with the artistic pursuits of figures such as John Cage, who sought to transcend personal ego in the creative process through the use of aleatory techniques. After extensive experience in live electronic music, I revisited Pure Data recently in order to create a public sound installation in, Crete, an endeavor that reawakened my interest in Pd and led to the development of Esotropio.

2. Patch Overview

Esotropio is a comprehensive Pure Data patch that functions as an automated sample sequencer with the ability to evolve over time. The patch is capable of loading eight sound banks from a designated folder on a computer, which it sequences and manipulates using a combination of algorithmic processes and frequency analysis. The resulting output is an experimental soundscape that transforms familiar samples into new, abstract auditory forms.

2.1. Signal Flow Architecture

The core functionality of Esotropio is distributed across six primary stages, each contributing to the modulation and processing of the sound output.

2.1.1. Stage A: Sample Playback and Triggering

Stage A constitutes the foundation of the patch, comprising eight sample players. These players are triggered by a system of counters that operate under the influence of either a Low Frequency Oscillator (LFO) based on $1/f$ noise, wirelessly with OSC (Open Sound Control) protocol or manually from the user. The counters track the frequency of excitations and initiate sample playback when a predefined threshold is met. This stage also triggers an optional internal phase modulation synthesizer, which modulates an oscillator based on the frequency characteristics of the samples.

2.1.2. Stage B: Stereo Reverb and Signal Splitting

The output from the sample engines in *Stage A* is fed into a stereo reverb and a set of crossfaders, splitting the audio signal into left and right channels. This stage introduces spatial dynamics to the sound, enhancing the immersive quality of the auditory experience. Additionally, an external line or mic input can be introduced at this point in the signal chain.

2.1.3. Stage C: Looping Mechanism

Stage C employs loopers to further manipulate the signal. These loopers can be controlled manually or triggered automatically based on the patch's internal logic. The loop points can be adjusted in real-time or left to be selected randomly, allowing for both precise and indeterminate sound manipulation.

2.1.4. Stage D: Advanced Modulation and Phase Synthesis

In *Stage D*, the processed channels undergo a pitch-shifter and are optionally routed through the phase modulation synthesizer. This synthesizer is triggered by the playback frequencies of the samples and incorporates a Freeze-Unfreeze function which allows specific frequencies to be held and manipulated. This stage significantly contributes to the textural complexity of the final output.

2.1.5. Stage E: Feedback, Sampling, and Secret Recipe Effects

Stage E further processes the split channels, with portions of the signal being routed directly to the output while others pass through a feedback delay and a sampler with time-stretching properties. An additional layer of processing is applied through the "secret recipe" effects, which include spectral crossfading, frequency shifting and echo. These effects are either manually controlled or triggered randomly, introducing an element of unpredictability into the final sound.

2.1.6. Stage F: Final Signal Mixing and Output

The final stage, *Stage F*, involves the mixing of all processed signals from the previous stages, which are then routed to the stereo output. This stage marks the completion of the signal's journey through the patch, producing the final auditory output.

3. Key Features

3.1. Phase Modulation Synthesizer

The phase modulation synthesizer is a critical component of Esotropio. This synthesizer modulates an oscillator using the frequencies derived from the playback samples. It includes a Freeze-Unfreeze function, which can be triggered manually or automatically, allowing for the selective manipulation of specific frequencies. This feature adds a significant layer of depth and variability to the soundscapes produced by the patch.

3.2. Secret Recipe Effects

The "secret recipe" effects are a collection of processing techniques applied during *Stage E*. These effects include spectral crossfading, frequency shifting, and echo processing, which can be routed through the signal chain in a semi-randomized manner. This feature is designed to introduce unique, unpredictable elements to the sound, further enhancing the experimental nature of Esotropio.

4. Technical Challenges

4.1. Sample Triggering

One of the primary challenges in developing Esotropio was achieving a balanced and coherent method for triggering sample playback. The initial implementation relied solely on self-generated triggers, which led to uneven and unpredictable playback. To address this, a low-frequency noise oscillator was incorporated to control sample triggering, allowing for a more nuanced balance between randomization and user control.

4.2. Frequency Representation

Accurately determining the frequency of each sample proved to be a significant challenge, particularly given the dynamic nature of the sounds involved. Various methods were tested, including real-time analysis and averaging techniques. Ultimately, the decision was made to capture the closing frequency of each sample, as this provided the most reliable and consistent results for further processing.

4.3. Temporal Control

The implementation of temporal control was another area of difficulty. To enhance the precision and flexibility of the patch, an additional layer of time-triggering was introduced in *Stage A*. This involved the use of triggers and counters to manage the timing of sample playback, resulting in a more controlled and deliberate sequencing process.

4.4. Freeze-Unfreeze Functionality

The development of the Freeze-Unfreeze function within the phase modulation synthesizer required careful consideration of timing and frequency detection algorithms. Ensuring that this feature operated seamlessly within the broader patch architecture was a complex task, necessitating extensive testing and refinement.

5. Future Directions

Several enhancements are planned for future iterations of Esotropio, aimed at expanding its capabilities and improving its functionality. These include:

1. Spectral modulation techniques for blending between sounds.
2. Enhanced looper functionality, incorporating zero-crossing detection for smoother transitions.
3. A search function based on harmonic context and amplitude for more precise sample selection.
4. Support for up to eight-channel output, allowing for more complex spatialization.
5. Improved algorithms for the Freeze-Unfreeze function, enabling more dynamic control.
6. Exploration of alternative synthesizer engines to diversify the sound palette.
7. Refined methods for frequency calculation to enhance accuracy.
8. Development of a graphical user interface (GUI) for more intuitive control over user-defined parameters.
9. Integration of MIDI output, linked to the frequencies of the samples for expanded musical possibilities.
10. Porting the patch to other platforms or developing it as a standalone application for broader accessibility.

6. Interface and User Interaction

The user interface of Esotropio is designed to facilitate ease of use while providing access to the patch's extensive functionality. The process for operating the patch is as follows:

1. **Loading Samples:** Begin by loading sample banks into the patch.
2. **Selecting Tempo Source:** Then choose a tempo source, which can be the internal LFO, an OSC input or manual control.
3. **Playback Initiation:** Playback is initiated and the volume can be adjusted as needed.
4. **Exploration of Features:** Free to experiment with the various controls, including tempo adjustments, sample manipulation, loopers, and the phase modulation synthesizer. Additionally, the secret recipe effects can be activated to further shape the sound output.

7. Conclusion

Esotropio represents my first exploration into automated and *adaptive* sound synthesis based on audio samples within the Pure Data environment. Although many refinements are needed, the patch offers a unique tool for creating artificial soundscapes. While the current implementation demonstrates substantial functionality, ongoing development will continue to expand its capabilities, providing users with even more creative possibilities.

Acknowledgments

The development of Esotropio would not have been possible without the contributions of several individuals and resources. Special thanks are extended to Alexandre Torres Porres from ELSE library, Pierre Massat, oeyxsaue at Patchstorage and Martin Brinkmann for their invaluable input.

References

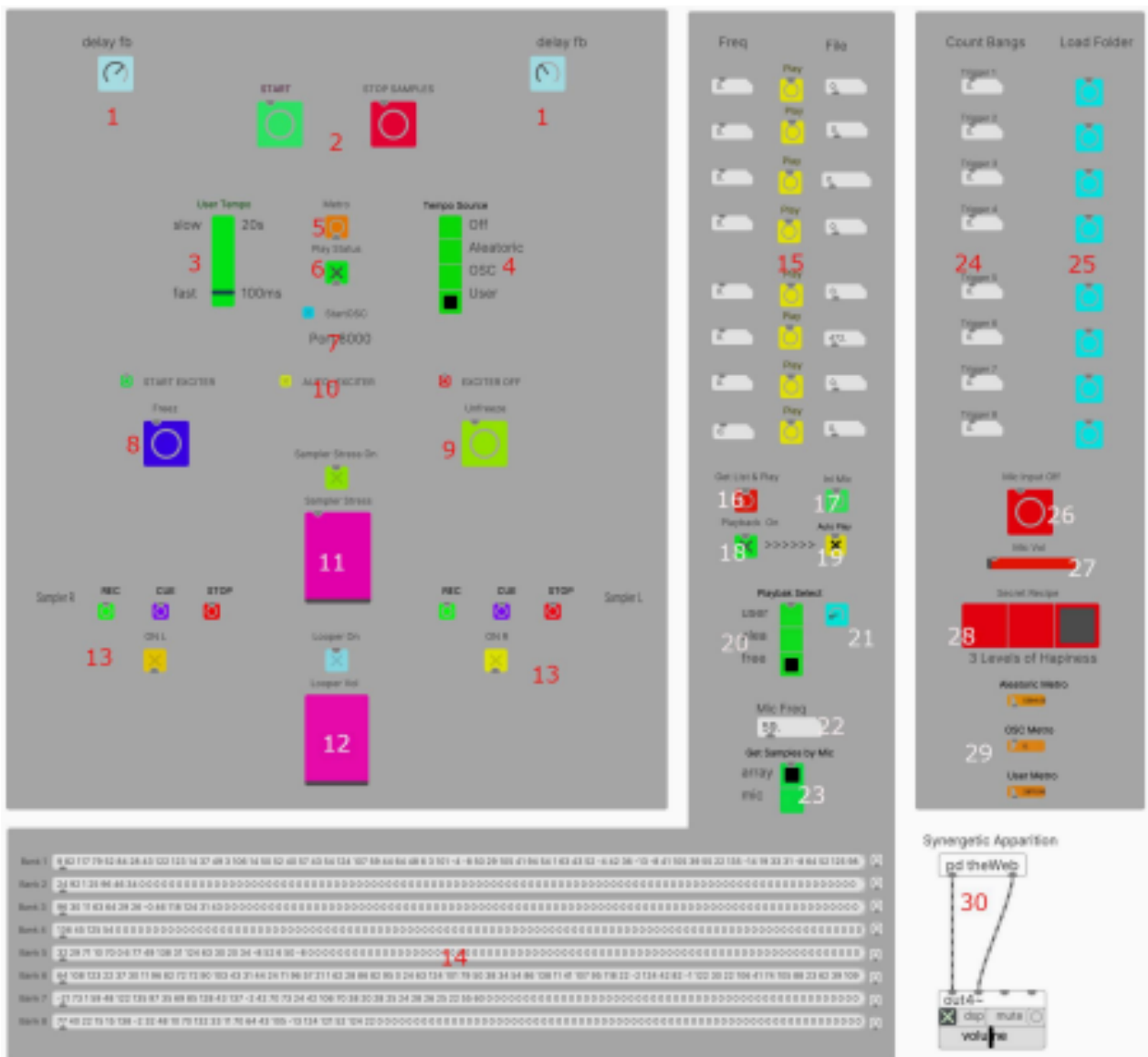
1. Brinkmann, M. (2008). *On the Threshold of Silence: Sound Installation and Sound Art*. In R. Laner (Ed.), *Sound Art: Between Avant-Garde and Popular Culture* (pp. 153-170). Springer.
2. Cage, J. (1961). *Silence: Lectures and Writings*. Wesleyan University Press.
3. Collins, N. (2006). *Handmade Electronic Music: The Art of Hardware Hacking*. Routledge.
4. Puckette, M. (1996). *Pure Data*. Proceedings of the International Computer Music Conference, Hong Kong, 269-272.
5. Roads, C. (2001). *Microsound*. MIT Press.
6. Truax, B. (1988). *Real-Time Granular Synthesis with a Digital Signal Processor*. *Computer Music Journal*, 12(2), 14-26.

Dimitris Barnias

PhD Candidate, Hellenic Mediterranean University

email: dbarnias@gmail.com

Interface Explanation:



1. Load Samples Banks [25].
2. Select Tempo Source [4].
3. Press Start [2].
4. Raise the Volume Slider [25.]
5. Wait a few seconds for the first counter to trigger a bang.
6. Play with Tempo Source.
7. Activate Sampler and Looper [11] & [12].
8. Start the exciter (Phase Mod synth) [10].
9. Initialize the external input and mix your voice with the patch [17].
10. Play with the settings of the Secret Recipe [28].

Controls Explained

1. Visual Delay Feedback (not user defined).
2. Start and Stop Playback (Stops sample playback - not the firing of the samples).
3. User defined Tempo (From 100ms to 10 sec)
4. Tempo control:
 - a. Aleatoric (based on $1/f$ noise)
 - b. OSC communication via port 8000
 - c. User (200ms - 10 sec)
5. Blinks on the current tempo.
6. Time triggering Indicator. Some internal actions may stop sample triggering.

Press again if necessary.
7. Initialize OSC communication
8. User option to Freeze internal PM synth.
9. User option to Unfreeze internal synth.
10. Phase Modulation Synth States: On / Off / Automated.
11. Timestress sampler volume and On / Off.
12. Looper Volume and On / Off.
13. Looper Cue Points and Stop Left / Right. Default: Automatic.
14. Display of frequencies per sample engine. Requested by [Get List & Play].
15. Row Freq: Search for Frequency. Row File: The result of the search. Play the file.
16. Search for selected frequencies within the past played samples.
17. Start capturing frequency values from external input.
18. Playback the result of search by [get list & play].
19. Automatic search & playback of samples based on external input.
20. Set playback method. Aleatoric & User based on the main patch. Free is independent.
21. If the user method is selected, adjust independently the time of playback.

- 22.** Display of the external input frequency.
- 23.** Select the source of the file search. External input or internal temporary memory.
- 24.** User defined playback weight of each sample engine. Higher values = less frequent.
- 25.** Load folders of samples.
- 26.** Stop external input.
- 27.** External input volume.
- 28.** Route the signal through various processors (Frequency Shifter / Echo).
- 29.** Display the main time triggering in milliseconds based on the selected source.
- 30.** Main volume (Best start with medium setting).