Serge Modular Archive Instrument (SMAI): Bridging Skeuomorphic & Machine Learning Enabled Interfaces

Ted Moore[*]
New Haven, CT
ted@tedmooremusic.com

Jean Brazeau[†]
Simon Fraser University
jean_brazeau@sfu.ca

ABSTRACT

The *Serge Modular Archive Instrument* (SMAI) is a sample-based computer emulation of selected patches on the vintage Serge Modular instrument that is housed at Simon Fraser University. Hours of recorded audio created by specified parameter combinations have been analyzed using audio descriptors and machine learning algorithms in the FluCoMa toolkit. Sound is controlled via (1) a machine learning dimensionality reduction plot showing all the recorded samples and/or (2) a skeuomorphic graphical user interface of the patches used to record the sounds. Flexible MIDI and OSC control of the software enables custom modulation and performance of this archive from outside the software. Differing from many software synthesis-based emulations, the SMAI aims to capture and archive the idiosyncrasies of vintage hardware as digital audio samples; compare and contrast skeuomorphic and machine learning enabled modes of exploring vintage sounds; and create a flexible instrument for creatively performing this archive.

Author Keywords

analog emulation, machine learning, skeuomorph, archive

CCS Concepts

•Applied computing → Performing arts; •Human-centered computing → Visualization systems and tools; •Information systems → Multimedia and multimodal retrieval;

1. INTRODUCTION

This paper introduces the Serge Modular Archive Instrument (SMAI), a sample-based computer emulation of selected patches on the vintage Serge Modular Music System instrument that is housed at Simon Fraser University.

[*]Audio Descriptor Analysis, Machine Learning Analysis, & Application Code.

[†]Synthesis Design & Graphic Design

NIME'23, 31 May–2 June, 2023, Mexico City, Mexico.

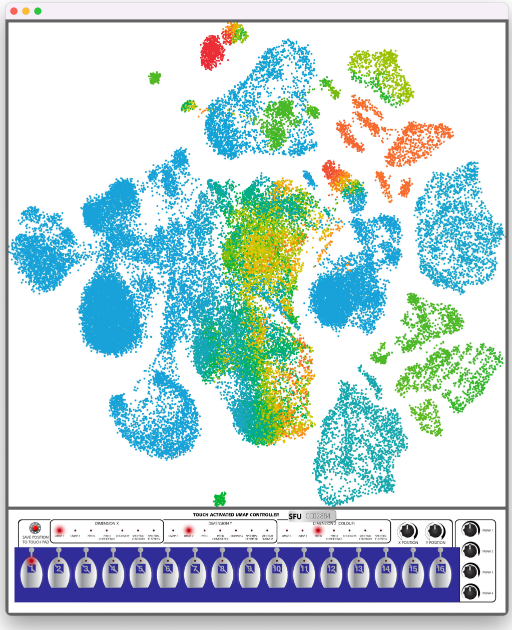

Figure 1: Two-dimensional plot for navigating samples in the Serge Modular Archive Instrument.

SMAI is designed to compare and contrast two strategies for navigating recorded audio samples in a multi-gigabyte corpus: (1) a skeuomorphic interface modeled on the original Serge Modular Music System [16] and (2) an audio analysis and machine learning based three dimensional plot of sound slices. In this paper, we describe the process of creating the dataset of audio, audio analyses, and data representations and how one uses the SMAI to explore and perform this archive in various ways. Future research will include qualitative comparisons of users' musical expressiveness with each interface [20]. for Download SMAI from the GitHub Repo at https://github.com/tedmoore/Serge-Modular-Archive-Instrument.

There are many software synthesizers that emulate analog synthesizer sounds, including the sounds of vintage Serge Modular systems [8, 4, 6, 7, 1]. Many also include skeuomorphic interfaces [2]. While some draw on recorded wavetables to approximate the idiosyncratic waveforms of analog systems [1], SMAI is unique in that it draws on many gigabytes of recorded audio, not for wavetable navigation, but as buffer playback to produce the sound of the emulated analog system.

Because of the continual evolution electronic music technology, there's a strong need for reliable and robust preservation and archiving practices for electronic music repertoire and techniques. [10] The multimodal ways we engage electronic sounds (recording, playing synthesizing, storing, teaching, learning, etc.) requires multimodal systems for accessing and making archives useful [3]. SMAI provides a strategy for both preservation and multimodal access by creating a unique archival instrument that not only archives material but offers multimodal exploration and performance of the archived instrumental sounds via skeuomorphic and data visualization interfaces.

While others have used audio descriptor analysis and dimensionality reduction to organize recorded audio in lower dimensional space for sonic navigation and performance [5, 17, 22, 13], SMAI is not created from a small, user-specified corpus of sounds, but rather a multi-gigabyte corpus that approximates the entire sonic space of a synthesizer patch. Additionally, comparing the dimensionally reduced plot with the skeuomorph allows one to contrast the two control strategies, comparing each for affordances of musical expression.

2. DATASET CREATION

Creating the datasets (audio recordings, audio analyses, and data representations) is done by the authors and therefore is not part of the users' experience. These datasets are presented to the users as "Patch Folders" (see section 2.4) that one can download and load into the SMAI software. This section describes the how the authors create these datasets.

2.1 Recording

For each patch created, four modulatable control voltage parameters were chosen on the Serge so each could be programmatically set to specific values (as voltage) via the Expert Sleepers ES-9 eurorack module [19]. A SuperCollider [11] script programmatically stepped through all possible combinations of the four parameters. For each of the four parameters a minimum and maximum voltage (potentially ranging from -10 volts to +10 volts as output by the ES-9; specified in SuperCollider from -1 to 1) and a number of steps for dividing up the voltage range (usually 10-15) is selected. We often choose a resolution of 15 steps for each parameter, which results in 50,625 (15^4) possible parameter combinations.[1] A voltage range smaller than -10 to +10 was used in cases where a certain parameter was mostly inaudible in a given range such as an oscillator frequency above the range of human hearing. After each parameter combination is sent to the Serge, a specified duration of time is waited for the recording to capture audio. Each of the 50,625 parameter combinations is recorded for one second, creating over 14 hours of recorded audio (this had to be broken up into multiple `wav` files for proper storage and transmission). Due to the need to balance audio quality with file size we chose to store these files with a sample rate of 44,100 Hz and bit depth of 16[2], which resulted in over 4 gigabytes of audio. During this process, the recording settings, number of steps, voltage ranges, and recording time were stored in a log file; the parameter combinations were stored as a CSV file; and a log of the patch cables' connections were all stored for future use.

[1]For explanation purposes, this paper describes the specifics of one dataset, or "Patch". Other datasets have slightly different settings.

[2]The recordings were originally made at a sample rate of 96,000 Hz and a bit depth of 24. While these much larger files become infeasible for easily sharing across the internet or storing in a computer's memory for random access by SMAI, we have kept these files for potential future use and/or research by ourselves or others.

2.2 Audio Analysis

Using the hours of recorded audio, audio analyses were conducted in non-real-time on each parameter combination's one second of audio. The audio analysis and machine learning analyses were achieved with the FluCoMa Toolkit [21] in SuperCollider [11]. Each one second of audio was analyzed for spectral centroid, spectral spread, spectral skewness, spectral kurtosis, spectral rolloff, spectral flatness, spectral crest, pitch, pitch confidence, loudness, true peak, and 40 mel-frequency cepstral coefficients (including coefficient zero). Each analysis used a window size and FFT size of 1024 samples, with a hop size of 512 samples. The FluCoMa analysis returns a time series of each of these audio descriptors across the duration of the one second sound slice (one descriptor value per FFT frame). In order to summarize the time series of descriptors, a statistical summary was conducted on each, returning the mean, standard deviation, skewness, kurtosis, minimum, median, and maximum values for each descriptor. These seven statistics for each of the 51 descriptors' time series was used as the raw vector (of 357 dimensions) for each sound slice analyzed.

2.3 Dimensionality Reduction with PCA and UMAP

After performing the audio analyses, the sound slices were represented as 50,625 points in 357 dimensional space. Dimensionality reduction algorithms were used to organize these sound slices into two dimensional space for plotting and navigating in the SMAI software. Because of the large variance in ranges of the different audio descriptors (e.g., spectral centroid in hertz from 20 to 20,000 while pitch confidence ranges between 0 and 1) it was important to first scale the dataset using standardization so each dimension would have a mean of 0 and standard deviation of 1. This ensures that each dimension will be weighted relatively equally in the distance computations that follow.[3] Principal Component Analysis was first used with the goal of removing any noise or redundancy in the dataset. A target of preserving 95% of the variance was set, which resulted in keeping the first 226 principal components, reducing the size of the dataset by about 37%. Next, the machine learning dimensionality reduction algorithm Uniform Manifold Approximation and Projection (UMAP) [12] was used to reduce the 226 principal components down to two dimensions for plotting in SMAI. Various UMAPs parameters were tested to find a two dimensional projection that felt musical and useful to the authors.[4] Future research should include a qualitative assessment of musicality by more users with more varied UMAP parameters.

2.4 "Patch Folder" Format

[3]Because MFCCs are 40 of the 51 audio descriptors, their timbral descriptions constitute the majority of the distance computation between points. The authors felt that this weighting was not inappropriate. Future experiments might vary weights between different kinds of analyses.

[4]The resulting parameters used were number of neighbors = 2, minimum distance = 1, iterations = 200, learning rate = 0.1.

After the audio of the parameter combinations has been recorded, the audio analysis complete, and the data reduced to two dimensions, all of these resources are placed in a "Patch Folder" that a user can select to load when booting up SMAI. So far, four "Patch Folders" have been created from four different analog Serge Modular patches. The contents of this folder include: (1) `wav` audio files, (2) a `csv` file containing audio analyses and parameter positions for each sound slice (see section 3 for more), (3) a `json` file containing some information about how the patch was created and how to display the skeuomorph, and (4) a `png` file to overlay on the skeuomorph showing a mock up of the cable connections used during recording.

3. SOUND NAVIGATION

Once the user opens the software and selects which patch to load, the audio samples can then be accessed in two ways: (1) a skeuomorphic interface of the original Serge Modular Music System or (2) a three-dimensional plot displaying sound slices as points.

3.1 Skeuomorphic Interface

In order to provide a sense of what kind of analog patch produced the sounds being heard, a skeuomorphic interface allows users to see what interconnections were used and adjust virtual knobs to control the sound. While the effectiveness and learnability of skeuomorphic interfaces have been shown to strongly rely on users' previous experiences [14, 15], the authors feel the SMAI skeuomorph strengthens its use as an archival representation of the Serge. on the SMAI, a red glow (see figure 3.1) shows which knobs are adjustable, while the rest do not respond to interaction. Each time one of these knobs is moved, a KDTree [9] is used to find the nearest neighbor of the current parameter settings within the dataset of sound slices loaded (these parameter settings are loaded through the `csv` file) and loops the corresponding one second sample of audio. The three-dimensional plot (section 3.2) shows the position of the slice being heard.

3.2 Three-Dimensional Plot

A three-dimensional plot (`x` and `y`, with color as a third dimension) allows for a more spatial and similarity-based interface for accessing the sounds slices. Each time the mouse is clicked or dragged on the plot a KDTree [9] is used to find the nearest point to the current mouse position (only the `x` and `y` dimensions are considered). The one second audio recording of the nearest point is then looped and the original parameter combination is displayed via the knobs on the skeuomorph. The three-dimensional plot can display, on any axis, various audio descriptors representing the sound slices (pitch, pitch confidence, loudness, spectral centroid, and spectral flatness, or either of the two UMAP dimensions). This allows for viewing, navigating, and performing the sound slices from multiple angles. Users may find different combinations of axes to be more musically intuitive (e.g., spectral centroid and loudness as a spectrogram), to reveal clusters of similar sounds (e.g., UMAP), or to be more performable via MIDI or OSC (see section 3.4).

3.3 Touch Activated Keyboard Sequencer

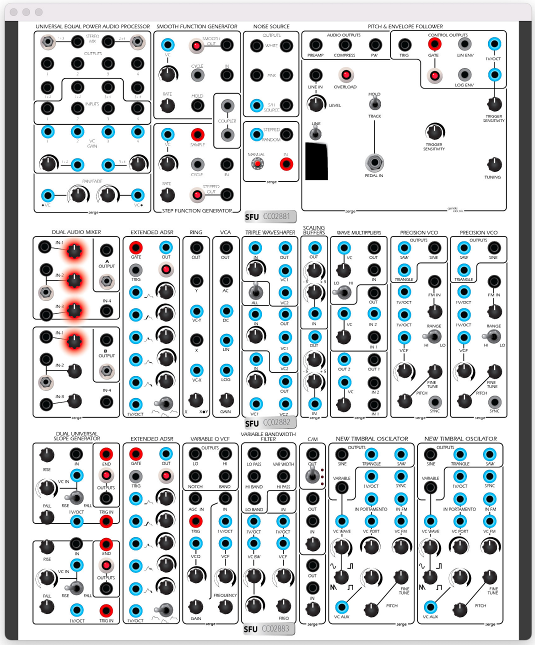

Figure 2: Skeuomorphic interface for controlling samples in the Serge Modular Archive Instrument.

Below the three-dimensional plot is a set of controls for the plot itself, four knobs that mirror the four knobs on the Serge Modular skeuomorph, and a skeuomorph of a Serge Touch Activated Keyboard Sequencer. This sequencer allows the user to store specific sound slices in one of the sixteen steps and then recall them by clicking on a given step (similar to the original analog device). To store a desired sound the user first uses the three-dimensional plot or the Serge skeuomorphic interface to navigate to a sound, then presses the digital button labeled "Save Position to Touchpad", and next clicks on which of the sixteen steps in which it should be stored. While this sequencer does not have a tempo or step advancement control, it is controllable via MIDI and OSC. The benefit of having 16 settable presets in the SMAI is that one can natively select and store 16 of the tens of thousands of possible sounds with a interactive GUI and then trigger those 16 sounds programmatically from a different software which the authors believe is preferable to asking users to specify one of the tens of thousands of sounds from outside the SMAI.

3.4 MIDI and OSC Control

Rather than add a limited set of modulation sources, such as LFOs, to the SMAI we decided to maximize the controllability of the software by opening up all the parameters for control via MIDI and OSC. Users are able to design their own LFOs, gestural mappings, sequencing strategies, and other performative techniques in exiting tools such as Max, SuperCollider, or Pure Data and send control information to the SMAI. See tables 1 and 2 for the relevant MIDI Messages and OSC commands.

Additionally, a MIDI learn functionality allows for assigning Control Change numbers (other than the reserved ones found in table 1) to control plot navigation (`x` and `y` axis) and the four skeuomorphic knobs.[5] This allows a user to

[5]The user can specify which connected MIDI controller to

quickly make and control sound with physical knobs on any attached MIDI controller.

Action	MIDI	Args
Skeuomorphic Param 1	CC 1	0-127
Skeuomorphic Param 2	CC 2	0-127
Skeuomorphic Param 3	CC 3	0-127
Skeuomorphic Param 4	CC 4	0-127
X Position on Plot	CC 5	0-127
Y Position on Plot	CC 6	0-127
Set X Axis	CC 101	0-6
Set Y Axis	CC 102	0-6
Set Color "Axis"	CC 103	0-6
Step Seq. Step "`n`"	Note On note 59 + `n`	n/a
Step Seq. Advance	Note On note 100	n/a

Table 1: Controlling the SMAI with MIDI.

Action	OSC	Args
Skeuomorphic Param 1	/param1	0-1 (float)
Skeuomorphic Param 2	/param2	0-1 (float)
Skeuomorphic Param 3	/param3	0-1 (float)
Skeuomorphic Param 4	/param4	0-1 (float)
X Position on Plot	/x	0-1 (float)
Y Position on Plot	/y	0-1 (float)
Set X Axis	/x-axis	0-6 (int)
Set Y Axis	/y-axis	0-6 (int)
Set Color "Axis"	/color-axis	0-6 (int)
Step Seq. Step "`n`"	/step-seq	`n` (int)
Step Seq. Advance	/step-seq-advance	n/a

Table 2: Controlling the SMAI with OSC.

3.5 Audio Playback

Looping the one second sound slice is executed with a 50 millisecond fade in and fade out. The next loop starts to fade in when the previous begins fading out. When a new sound slice is selected to be heard, either with the skeuomorphic controls or the two-dimensional plot, the previous slice begins fading out at the same time the chosen slice begins fading in. This sample cross-fading strategy allows for convincing and fluid movement between sound slices that approximates the sound of turning a knob on an analog system.

4. FUTURE RESEARCH

This paper introduces the design, implementation, and control strategies of SMAI. Future research should include qualitative assessments comparing users' experiences of musical expressiveness with SMAI's two control systems: the skeuomorphic control and the machine learning enabled three dimensional plot. The authors hope that the design choices of this software can lead to more understanding of the musical differences between these two control design paradigms.

Future software improvements to SMAI might include (1) using `flac` files for smaller audio file sizes, (2) a custom file type contains all the elements in a 'patch' folder would make for a simpler user experience, and (3) investigating stochastic granulation strategies may help prevent the one second sounds slices from being clearly perceived as a loop. Currently there is no way to natively record the sounds made by SMAI, however it is possible to use the "loopback" functionality found in may audio hardware devices, or with an internal software audio device, such as BlackHole [18]. Future versions may include a loudness threshold to remove very quiet slices from the dataset before performing the dimensionality reduction. This would ensure that the inaudible sound slices do not end up in the UMAP plot, where they currently take up a lot of space. These slices could still appear in the plot when a non-UMAP dimension is selected for either the `x` or `y` axis and could still be reached via the skeuomorphic interface.

The authors hope that the design choices and control paradigms of SMAI can be applied to the preservation, archiving, and performing of other vintage analog synthesizer systems as well. Aside from the Serge-specific skeuomorphic designs, the technical pipeline and system suggested here could be adapted for any large corpus of archival synthesizer recordings.

5. ETHICS STATEMENT

The authors have made efforts to make the artistic and research use of SMAI as accessible and inclusive as possible. All of the code is open source and available on GitHub under the BSD-3-Clause license. All of the sound files that are used with the software are licensed under Creative Commons 4.0 International. Additionally, all of the software dependencies are open source and can also be found on GitHub. Currently SMAI is only available for Mac. Future improvements should include a development pipeline that enables compiling for more operating systems.

All of the technical work done on SMAI was performed on MacBook Pro laptops. Recording the audio is necessarily done in real-time and therefore requires many (around 14) hours of personal computer energy usage per "Patch Folder" created (currently we have 4 "Patch Folders" available). For each "Patch Folder" the audio analysis and machine learning are completed using a personal computer (all on a CPU) using around one hour of computer energy usage. All of the SuperCollider code that does the analysis and machine learning has been created with the aim of computational efficiency. Because SMAI is a sample based emulation, while running it takes very little CPU energy to run, however does require a large amount of memory.

6. CONCLUSIONS

The SMAI is a sample-based computer emulation of selected patches on a vintage Serge Modular Music System. By including a skeuomorphic interface alongside a machine learning enabled three-dimensional plot of the sound slices, SMAI offers the comparison of two distinct control paradigms from which to approach the same corpus of audio. MIDI and OSC control further enables maximum flexibility in how the sounds are controlled. All of these ways of viewing and accessing the sounds makes SMAI a unique software program that exists both as an archive and a performable instrument for musical expression. Through further research using SMAI, the authors hope that these design choices can lead to more understanding of the musical differences between these two control design paradigms.

use with the SMAI by hitting `m` (for "MIDI").

7. ACKNOWLEDGMENTS

Thank you to Owen Green, Mauricio Pauly, Judy Radul, Pierre Alexandre Tremblay, and Simon Fraser University for support and guidance during the development of SMAI.

8. REFERENCES

[1] P. Bacon. Surge rack. Web: `https://github.com/surge-synthesizer/surge-rack`, 2019.

[2] A. Belt. Vcv rack. Web: `https://vcvrack.com/`, 2016.

[3] K. Dahan. (re) discovering sounds of ccrma-towards computer music preservation. International Computer Music Association, 2018.

[4] F. Esqueda, H. Pöntynen, J. D. Parker, and S. Bilbao. Virtual analog models of the lockhart and serge wavefolders. *Applied Sciences*, 7(12):1328, 2017.

[5] S. Fasciani and L. Wyse. Adapting general purpose interfaces to synthesis engines using unsupervised dimensionality reduction techniques and inverse mapping from features to parameters. *Proceedings of the International Computer Music Conference*, (June 2016):467–472, 2012.

[6] B. Galloway. Creeture. Web: `https://www.native-instruments.com/en/reaktor-community/reaktor-user-library/entry/show/7369/`, 2013.

[7] B. Galloway. Animal surge r5/r6. Web: `https://www.native-instruments.com/en/reaktor-community/reaktor-user-library/entry/show/9275/`, 2015.

[8] G. Gormond, F. Esqueda, H. Pöntynen, and J. D. Parker. Waveshaping with norton amplifiers: modeling the serge triple waveshaper. In *International Conference on Digital Audio Effects*, pages 288–295. University of Aveiro, 2018.

[9] G. Kogan. ofxkdtree. Web: `https://github.com/genekogan/ofxKDTree`, 2017.

[10] S. Lemouton. The electroacoustic repertoire: Is there a librarian? *array.*, pages 7–14, 2020.

[11] J. McCartney. Rethinking the computer music language: Supercollider. *Computer Music Journal*, 4(26), 2002.

[12] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[13] C. Ó Nuanáin, S. Jordà Puig, H. Boyer, et al. An interactive software instrument for real-time rhythmic concatenative synthesis. In *NIME 2016. Proceedings of the International Conference on New Interfaces for Musical Expression; 2016 Jul 11-15; Brisbane, Australia.[place unknown]: NIME; 2016. p. 383-7.* International Conference on New Interfaces for Musical Expression (NIME), 2016.

[14] D. Oswald, S. Kolb, et al. Flat design vs. skeuomorphism–effects on learnability and image attributions in digital product interfaces. In *DS 78: Proceedings of the 16th International conference on Engineering and Product Design Education (E&PDE14), Design Education and Human Technology Relations, University of Twente, The Netherlands, 04-05.09. 2014*, pages 402–407, 2014.

[15] C. Rasmussen. *Evaluating the usability of software synthesizers: An analysis and first approach.* PhD thesis, University of Guelph, 2018.

[16] D. J. Rich Gold and M. LaPalma. *An Introduction to the Serge Modular Music System.*

[17] G. Roma, O. Green, and P. A. Tremblay. Adaptive mapping of sound collections for data-driven musical interfaces. In *NIME*, pages 313–318, 2019.

[18] D. Roth. Blackhole. Web: `https://github.com/ExistentialAudio/BlackHole`, 2019.

[19] E. Sleepers. Es-9 usb audio interface. Web: `https://www.expert-sleepers.co.uk/es9.html`, 2022.

[20] D. Stowell, A. Robertson, N. Bryan-Kinns, and M. D. Plumbley. Evaluation of live human–computer music-making: Quantitative and qualitative approaches. *International journal of human-computer studies*, 67(11):960–975, 2009.

[21] P. A. Tremblay, O. Green, G. Roma, J. Bradbury, T. Moore, J. Hart, and A. Harker. Fluid corpus manipulation toolbox. Web: `https://zenodo.org/record/6834643`, 7 2022.

[22] A. Xambo Sedo, G. Roma, A. Lerch, M. Barthet, and G. Fazekas. Live repurposing of sounds: Mir explorations with personal and crowdsourced databases. In *Proceedings of the international conference on new interfaces for musical expression.* Virginia Tech Blacksburg, VA, 2018.