Survival Kit

EUGENE MARKIN, New York University

1. PROGRAM NOTES

Survival Kit is a live electroacoustic piece that explores the connection between textual and musical meanings. It is a revised take on choral music in the digital era. The author experiments with ways to interpret natural language in computer music and suggests a novel approach to performing text/sound compositions. The foundation of the piece is a poetic text that lists all the things that may come to mind amidst a futile preparation for a global disaster.
The piece is performed by a single performer in the *live coding* manner. The author enters the text in his original computer music software, which triggers sections of pre-recorded music and corresponding processing algorithms. All vocals were performed by a collaborator vocalist (tenor) using a recording score for individual lines, and then edited and programmed into the software by the author.

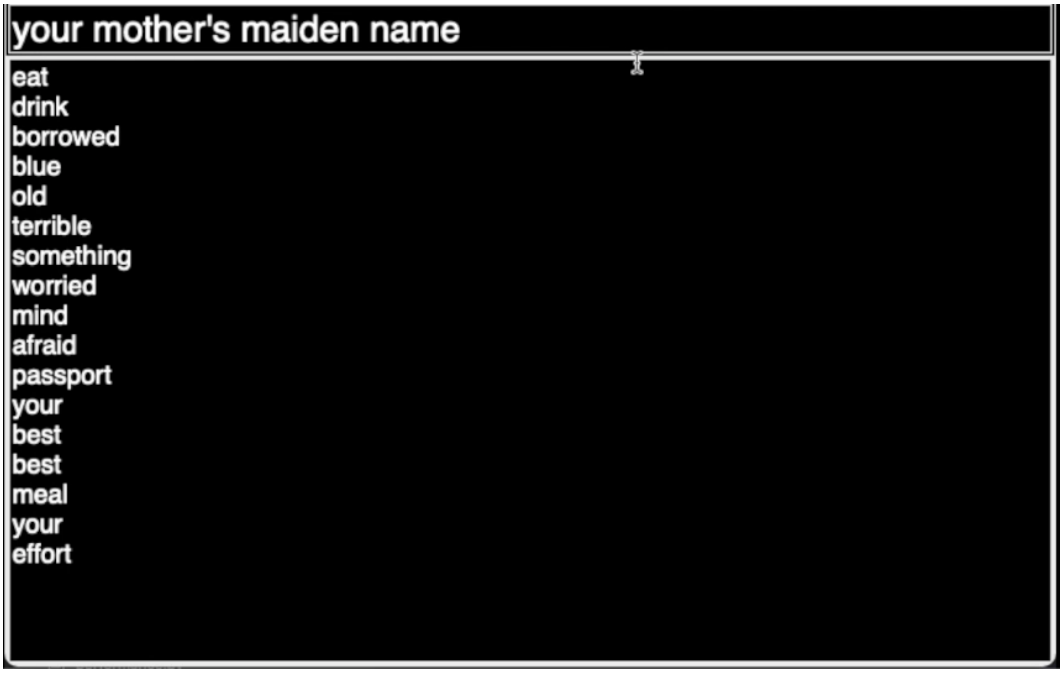

Fig. 1. The software's user interface.

DOI:http://dx.doi.org/10.1145/0000000.0000000
Music Proceedings of the International Conference on New Interfaces for Musical Expression
NIME'23, May 31–June 2, 2023, Mexico City, Mexico

2. PROJECT DESCRIPTION

The project started with writing of the poem and composing a recording score for the vocalist to perform. Selected words and phrases from the text were sung as simple pre-composed motives, others were improvised during the recording session. In addition, the vocalist has recorded the entire poem as a spoken word. The recording was subsequently sliced into individual samples corresponding to each line of the text and uploaded into the software.

The software is written in Python and adds functionality to FoxDot [1] live coding language allowing the user to control computer music operations with plain text. Recorded samples are placed into directories titled with words that trigger their playback. Whenever a word is entered into the input console, the software randomly selects one of the samples in the corresponding directory.

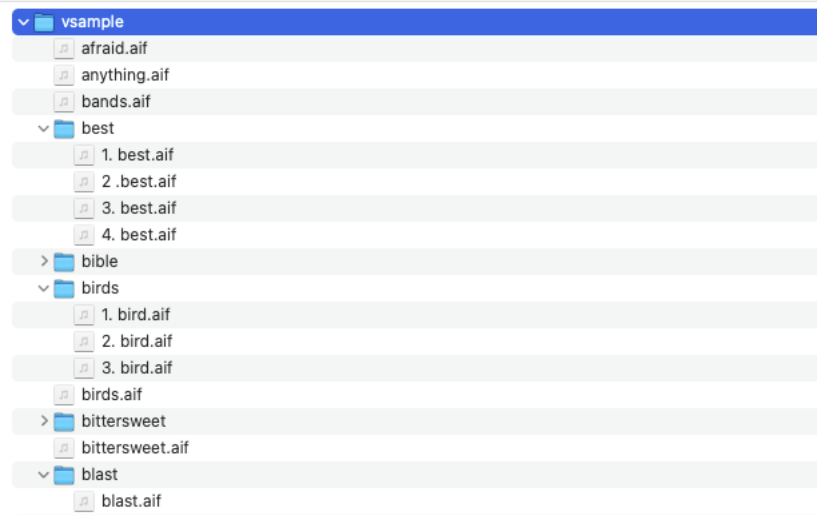

Fig. 2. Sample directory.

The samples are played by a Supercollider [2] synthesizer which performs an FFT, applies selected spectral domain operations and resynthesizes the signal with the inverse FFT. The synthesizer function takes multiple arguments that control the sample duration, envelope, LFO and parameters of the spectral domain operations such as freeze and enhance. Upon launch, the software executes a Python script that collects all samples from a given top-level directory and assigns their individual parameters as in the following example:

```
best = sk["best"]
best.enhance = Pattern([8, 16, 32, 64])
best.freeze = 0.4
best.dur = 16
best.lfohz = Pattern([0.25, 0.5, 2, 3])
best.lfodepth = 0.7
```

During the performance the text of the poem is entered into the input console line-by-line. Each line is interpreted so that the words separated by a whitespace are played in sequence. If a line ends with a period, the sequence is played once and stops, otherwise the last sample in the sequence keeps looping until it is removed by the performer from the output window of the user interface. Certain words and characters are reserved for special control operations. For example, the word "three" tells the program that three samples marked with the word following "three" should be played at the same time.

3. PERFORMANCE NOTES

The screen of the performer's laptop is projected in the venue for the audience to see. Each line of the entered text immediately triggers the sample playback, allowing the listeners to follow the action/reaction process of the performance. The performer is able to control the timing of starting and stopping the sounds, as well which sounds should be playing simultaneously and which of them should be sequenced. In addition, the code contains randomization of the selected samples and their playback parameters. As a result, each performance of the piece is unique and always differs from the previous one.

4. MEDIA LINK(S)

- Video:https://drive.google.com/file/d/1T57PFOl9oOIpze-f88bkcRfuJNj8MW1F/view?usp=share_link

ACKNOWLEDGMENTS

The author would like to thank vocalist Peter Traver for the recording of the source material that served as a base of this piece.

REFERENCES

[1] R. Kirkbride. Foxdot: Live coding with python and supercollider. *Proceedings of the International Conference on Live Interfaces*, 2016.

[2] J. McCartney. SuperCollider, a new real time synthesis language. *Proceedings of the 1996 International Computer Music Conference*. The International Computer Music Association, 1996.