Euclidean Algorithm for auto-generative patterns in a Supercollider application

Project of Computer Music course

Students:
De Bari Mauro Giuseppe Matr.899371
Albertini Davide Matr.883347

October 18, 2018

Contents

1 Introduction

Euclidean rhythms are a large class of rhythms which can be generated through the Euclidean Algorithm. Making its first appearence in Euclid's Element, it's one of the oldest algorithm, and it was used to compute the greatest common divisor of two given integers. It was Godfried Toussaint to discover, in 2005, that the Euclidean algorithm's structure may be used to generate the aforementioned class of rhythms. Moreover, he found that those patterns are often present traditional and world music[1]. He noticed that the main characteristic of those rhythms is that their onset pattern are distribuited as evenly as possible. Since then, Euclidean rhythm generators started to spread in the music production world, since with very few parameters one can generate very complex patterns and polyrhythms. A lot of eurorack synthesizer modules have been developed: Mutable Instruments Yarns[2], vpme.de Euclidean Circles[3] and 2HP Euclid[4], just to name a few. This paper will present a software implementation of a drum machine capable of generating Euclidean rhythms, using Supercollider programming language.

2 The algorithm

The purpose of the researchers in the past year was to find a easy way to generate all the possible rhythms present in music, in order to extend the dictionary of Automatic Music Composition[5]. The derivation of G. Touissant[1] comes from an analyisis over Bjorklund's studies on SNS accelerators[6] in nuclear physics. In this case, time is divided into intervals and during some of these intervals a gate is to be enabled by a timing system that generates pulses that accomplish this task. The problem for a given number n of time intervals, and another given number $k < n$ of pulses, is to distribute the pulses as evenly as possible among these intervals.

In our case the algorithm to be develped take as input a tuple *(k, n)*, where n represents the length of the played sequence and k the number of onsets inside the sequence. The output is a sequence of length n, where the k onsets are equally spaced inside the sequence. For example, if we consider the tuple *(4, 16)*, the result has to be a sequence like:

[1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0]

3 SynthDefs and Gui Layout

In this section we will speak about the SynthDefs, their pattern controllers (Pdefs and Pbinds) and the Gui definitions.

4 Conclusions

In this section we will conclude the report providing short synthesis of the project and Github analysis.

References

[1] Godfried Toussaint, “The euclidean algorithm generates traditional musical rhythms,” Tech. Rep., School of Computer Science, McGill University, Montreal, Quebec, Canada, 2004.

[2] O. Gillet, “Mutable instruments. yarns, eurorack for generetion of euclidean rythms,” Available at https://mutable-instruments.net/modules/yarns/manual/.

[3] Vladimir Pantelic, “Euclidean circles eurorack,” Available at https://vpme.de/euclidean-circles/.

[4] “Modulus for generetion of euclidean rythms,” Available at http://www.twohp.com/modules/euclid.

[5] J.-P. Allouche and J. O. Shallit, Eds., *Automatic Sequences*, Cambridge University Press, 2002.

[6] E. Bjorklund, “The theory of rep-rate pattern generation in the sns timing system,” Tech. Rep., Los Alamos National Laboratory, Los Alamos, U.S.A, 2003.