SynthChallenge 2019

Ondrej Lahoda

1 Introduction

This document is an overview of MATLAB project for SynthChallenge 2019 organized by Signal Analysis, Modelling and Interpretation research group at CTU in Prague. It has 3 parts (*Obligatory composition, Major scale* and *Elective part*).

2 Sound synthesis

I used 3 main methods for sound synthesis of different instruments.

- Wavetable synthesis
- Additive synthesis
- Karplus-Strong algorithm

2.1 Wavetable synthesis

This method is sample based algorithm and it was mostly used in samplers and mobile phone devices with limited memory space. The method of wavetable synthesis is well described in articles with topic of polyphonic ringtones such as [2]. It uses one short sample of pitch period to generate periodic signal by simply repeating the basic pitch period. By reading the sample with different step than original sampling frequency, we can obtain notes of higher or lower frequencies. That leads to saving up memory space compared to true samplers.

2.2 Additive synthesis

Additive synthesis is the most basic algorithm for generating synthetic sounds. We can already obtain satisfying solutions by simply generating sine waves of desired frequency and amplitude.

2.3 Karplus-Strong algorithm

The Karplus-Strong algorithm is an extension of the original wavetable approach. It was discovered as a method of adding more interesting timbre to the wavetable synthesis. The first step is to average two successive samples of the wavetable, which gives the tone decay similar for plucked string. Later experiments showed that the original pitch period sample is not needed and can be replaced by white noise generator, which has rich spectrum. The original paper for this algorithm is [1].

3 Obligatory composition

I worked with the files of composition `'popelka.midi'`. This piece contains following instruments:

- Flute
- Piano
- Strings ensemble (bowed)
- Strings ensemble (pizzicato)
- Bassoon

The main instrument of this composition is the piano. My focus was to synthesize a precise piano sound and to distinguish between the two techniques of playing for strings.

3.1 Piano

By suggestions from lectures and articles a good choice for piano synthesis is using wavetable. Piano is a very complex instrument with large range. A sample based method is a good starting point to involve the characteristics of original piano sound. I generated sample of tone `a=220 [Hz]` in the piano software instrument of DAW Logic Pro X and then extracted one pitch period of the sample. The result using just one sample for whole range of the piano was not satisfying. I improved the method by generating different samples for each octave and loading the wavetable separately for different pitch of the note. To mimic the progress of amplitude after the hammered string a ADSR enveloping method is used. ADSR stands for Attack, Decay, Sustain and Release and shapes the amplitude with desired values for those stages of amplitude progress.

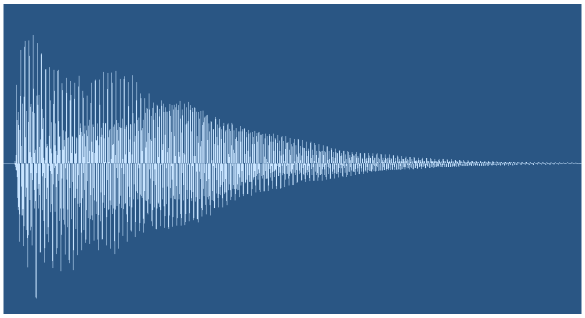

Fig. 1: sample of piano tone a = 220 [Hz]

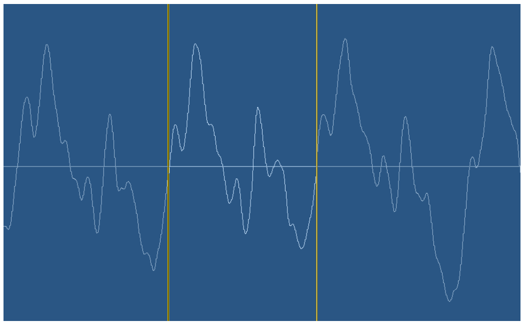

Fig. 2: one pitch period

3.2 Flute, Bassoon and Bowed Strings

The common approach for all these three instruments is additive synthesis. The main difference is the basic tone generator.

- Sinusoidal for Flute
- Sawtooth for Strings
- Square for Bassoon

Using the pure square generator MATLAB function for Bassoon leads to very sharp sound. In the implementation of my code an approximation by odd sinusoidal harmonics is used which leads to smoother sound. To improve the perception of amplitude ADSR enveloping method is used for all instruments again as for piano. This time a smoother values with longer sustain are generated.

4 Major scale

For the second part I have chosen to generate C major scale of three octaves. The previously synthesised instruments play the scale one by one in the following order:

1. Bassoon *(c-f)*
2. Piano *(g-c1)*
3. Bowed Strings *(d1-a1)*
4. Pizzicato Strings *(h1-e2)*
5. Flute *(f2-c3)*

5 Own part

For the free topic part I decided to explore an ancient acoustic problem discovered by Pythagoras called the *Pythagorean comma*.

5.1 Pythagorean scale

The musical scale called Pythagorean is based on shortening strings by exact geometrical values. The easiest is to divide the string in half which leads to tone of double frequency. That is a tone one octave higher from the previous. In the same manner a ratio of 2:3 is defined as perfect fifth. For example the frequency of `a1(440Hz)` and `e2(660Hz)` is the ratio of 2:3. Going consecutively by the ratio of 2:3 generates all of the 12 tones in the Pythagorean scale. This progression is known as the circle of fifths.

5.2 Pythagorean comma

Pythagoras soon discovered that moving up and down in larger steps (above octave) changes the tuning of the foundation string. Further exploration lead to definition of the *Pythagorean comma*. Finishing the full circle of fifths by 12 steps of 3:2 ratio does not lead to the same note as we started.

You can imagine going from the lowest note on the piano keyboard, which is `A = 27.5 [Hz]` upwards 12 fifths leading to `a4 = 3520 [Hz]`. But if you do the math for ratio of perfect fifth (3:2) it turns out to be tuned higher at `a4 = 3568 [Hz]`. That is a difference of *Pythagorean comma* and problems with solving it lead to invention of the equal temperament. More about the complications of different ways of tuning can be read in [3].

Today piano keyboards are equal tempered, but the MATLAB implementation let us do the exact ratios and hear the difference of Pythagorean comma.

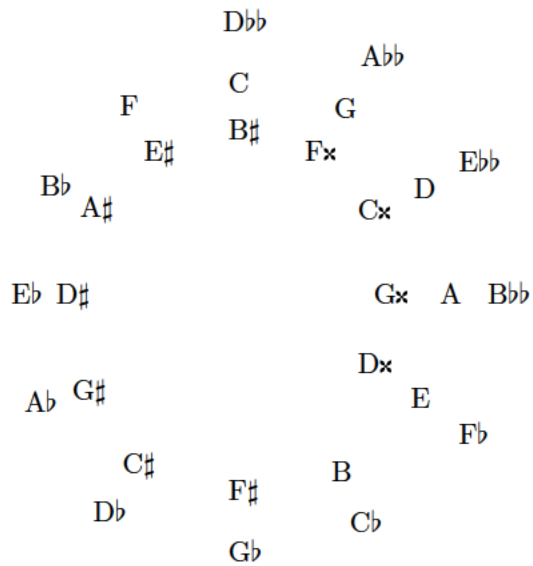

Fig. 3: Circle (spiral) of Fifths, [3]

References

[1] Kevin Karplus and Alex Strong. “Digital Synthesis of Plucked-String and Drum Timbres”. In: *Computer Music Journal* 7.2 (1983), pp. 43–55. ISSN: 01489267, 15315169. URL: http://www.jstor.org/stable/3680062.

[2] Robert Maher. “Wavetable Synthesis Strategies for Mobile Devices”. In: *Journal of the Audio Engineering Society* 53 (Mar. 2005).

[3] Dave Benson. “Scales and temperaments: the fivefold way”. In: *Music: A Mathematical Offering*. Cambridge University Press, 2006, pp. 161–209. DOI: 10.1017/CBO9780511811722.007.