

Tytuł projektu: Program umożliwiający dodawanie efektów gitarowych do pliku w formacie WAVE
Autor: Mateusz Danilczuk
Data: 9.06.2019

1. Założenia projektowe i opis projektu

Celem projektu było stworzenie programu pozwalającego na otwieranie plików w formacie 16-bit WAVE, a następnie dodawanie do nich efektów gitarowych z parametrami ustalonymi przez użytkownika.

Program został napisany w języku Python 3.7 z użyciem następujących modułów:

- NumPy – zoptymalizowane pod względem szybkości i pamięci obliczenia na macierzach
- wavio – odczytywanie i zapisywanie danych w formacie WAVE
- TkInter – otwieranie standardowych okien do wybierania plików
- Simpleaudio – prosty moduł do odtwarzania dźwięku
- wxPython – interfejs graficzny użytkownika

Oferuje następującą funkcjonalność:

- Intuicyjne otwieranie i zapisywanie plików
- Wyświetlanie nazwy i długości pliku
- Wybór czterech efektów wraz z możliwością dobrania parametrów
- Możliwość odtworzenia oryginalnego i zmodyfikowanego utworu
- Resetowanie dodanych efektów
- Wygodny interfejs graficzny

2. Opis efektów

1. Echo (Delay)

Efekt polega na zwyczajnym dodaniu do sygnału wejściowego jego opóźnionej kopii pomnożonej przez współczynnik zanikania (decay factor) określający amplitudę opóźnionej kopii.

$y[n] = x[n] + gx[n - N]$, gdzie: N – opóźnienie w próbkach

2. Distortion (Overdrive)

Nieliniowy efekt charakterystyczny m.in. dla muzyki rockowej. Polega na zniekształceniu sygnału przez funkcję o określonym nachyleniu. Im funkcja jest bardziej stroma tym efekt jest mocniejszy. Do najczęściej stosowanych funkcji należą wszelkie modyfikacje funkcji arcus tangens, tangensa hiperbolicznego, czy funkcji eksponencjalnej.

W swoim projekcie zastosowałem następującą funkcję, która moim zdaniem generowała najbardziej przyjemne dla ucha zniekształcenie:

$$y[n] = \text{sgn}(x[n]) * (1 - e^{-|Gx[n]|}) , \text{gdzie: } G - \text{input gain}$$

Większa wartość input gain oznacza bardziej stromą charakterystykę a tym samym bardziej zniekształcony sygnał.

3. Tremolo

Efekt ten powoduje szybką, regularną zmianę w amplitudzie sygnału, w efekcie dźwięk przypomina szybkie pociągnięcia struny. Implementowany jest jako dodanie do sygnału wejściowego kopii sygnału pomnożonej przez oscylator o niskiej częstotliwości (do 20 Hz).

$$y[n] = x[n](1 + d\cos(2\pi \frac{f_{LFO}}{f_s} n)) , \text{gdzie: } d \text{ (depth) - określa to jak słyszalny jest efekt}$$

f_{LFO} – częstotliwość oscylatora

f_s – częstotliwość próbkowania sygnału

4. Flanging

Efekt wytwarzający dźwięk kojarzący się z samolotem odrzutowym lub statkiem kosmicznym. Implementacja flangu jest podobna do implementacji echa z tą różnicą, że oprócz stałego, małego opóźnienia (do kilku milisekund) występuje opóźnienie określone przez oscylator o niskiej częstotliwości (do 2 Hz) oraz amplitudę oscylatora.

$$y[n] = x[n] + x[n + N + \left\lfloor R\sin\left(2\pi \frac{f_{Sweep}}{f_s} n\right) \right\rfloor] , \text{gdzie:}$$

R (range) – określa rozrzut opóźnienia

f_{Sweep} – częstotliwość oscylatora

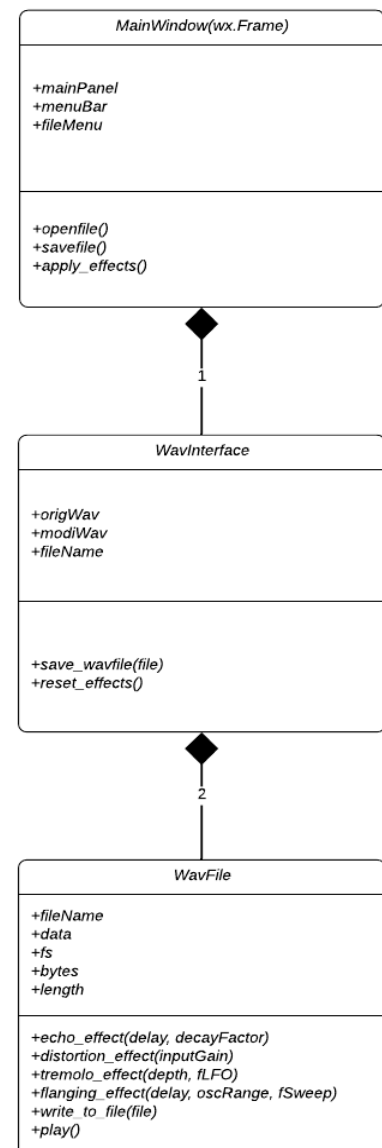
f_s – częstotliwość próbkowania sygnału

Struktura programu

Na program składają się trzy klasy:

1. WavFile – klasa reprezentująca dane pochodzące z pliku .wav, zawierająca oprócz samego sygnału, wszystkie informacje takie jak częstotliwość próbkowania, ilość bajtów na próbkę, długość utworu. Zawiera metody służące do dodawania efektów, odtwarzania dźwięku i zapisywania danych do pliku.
2. WavInterface – klasa pełniąca rolę łącznika między WavFile a resztą programu. Zawiera dwa obiekty klasy WavFile, które reprezentują oryginalny i zmodyfikowany utwór. Oprócz metod służących do dodawania efektów, pozwala także na ich resetowanie.
3. MainWindow – klasa dziedzicząca z wx.Frame, która stanowi całe GUI i odpowiada za zarządzanie eventami takimi jak kliknięcie przycisku, czy wpisanie wartości parametru. Do jej pola należą wszystkie obiekty reprezentujące elementy interfejsu graficznego oraz WavInterface.

Rys. 1 Diagram klas



Rys. 2 Interfejs graficzny

