Projekt zespołowy

**System weryfikacji jakości strumienia danych audio wykorzystujący technikę fingerprintingu.**

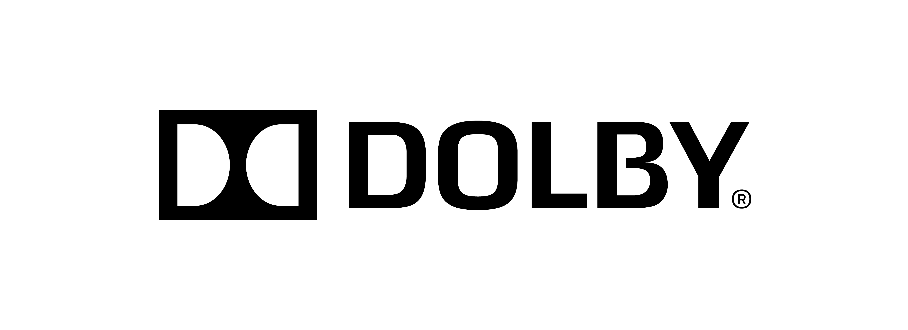
Prowadzący: Paweł Jaroch, Maciej Walczyński

Skład grupy projektowej:

* Anna Gralak
* Karolina Pondel
* Patryk Grzybała
* Monika Góral
* Marta Hubiak

Kierunek: Elektronika, studia I stopnia

Specjalność: Inżynieria Akustyczna





Spis treści.

Spis treści

**Nie znaleziono żadnych pozycji spisu treści.**

1. **OPIS PROJEKTU**

Krótki opis projektu:[[1]](#footnote-1)

Audio fingerprint (ang. odcisk palca) to skondensowana reprezentacja sygnału dźwiękowego – jedno- lub wielowskaźnikowa macierz wyznaczona w oparciu o właściwości czasowe i/lub częstotliwościowe sygnału. Projekt ma na celu skonstruowanie fingerprint’u, który będzie stanowił referencję dla oceny jakości strumienia danych audio poddanego dalszej obróbce (kodowanie/dekodowanie, postprocessing, itd.) w sytuacji kiedy nie jest możliwe odwołanie się do oryginalnego sygnału, np. podczas streamingu, transmisji ‘na żywo’ lub dowolnej innej sytuacji, w której chcemy zredukować złożoność obliczeniową oraz zasoby potrzebne do przechowania sygnału referencyjnego. Odległość między dwoma fingerprint’ami (macierzami) będzie proporcjonalna do stopnia degradacji strumienia audio.

Bazując na przykładowym rozwiązaniu, opracowanym przez pracowników firmy Dolby, uczestnicy projektu będą mieli za zadanie m. in.:

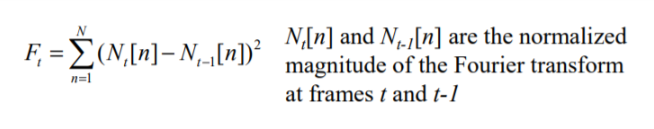
* wyselekcjonować charakterystyki sygnału adekwatne do oszacowania jego jakości,
* zaimplementować metody normalizacji danych oraz wyznaczania odległości między macierzami,
* zbadać możliwości wykorzystania charakterystyk percepcyjnych sygnału dźwiękowego,
* opracować sposób na dodanie do fingerprint’u metadanych audio.

Projekt w całości realizowany będzie w języku Python oraz z wykorzystaniem zewnętrznych bibliotek pythonowych.

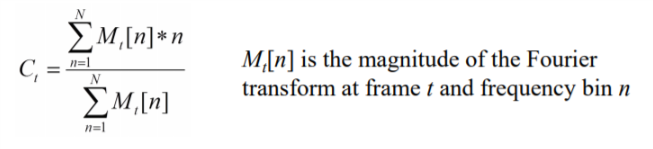
Dokładny opis projektu:

Spis parametrów: (opis liter we wzorach, można dodać jakie info o jakości daje nam każdy parametr, np. zc mówi o zaszumieniu)

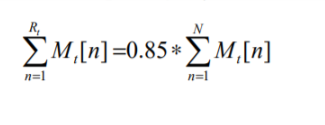
1. **RMS** – pierwiastek kwadratowy ze średniej kwadratu wartości chwilowych sygnału okresowego.
2. **ZC** (zero crossing) -Ilość przejść przez zero jest sumą wszystkich przejść amplitudy sygnału.
3. **SPECTRAL FLUX** - Kwadratowa różnica między znormalizowanymi wielkościami kolejnych rozkładów widmowych. Miara wielkości lokalnej zmiany widmowej. (wzory lepiej wyglądają napisane w wordzie)



1. **SPECTRAL CENTROID** - miara używana w DSP to opisu widma. Wyznacza "środek ciężkości" widma. Percepcyjnie ma związek z wrażeniem 'jasności' dźwięku.



1. **SPECTRAL FLATNESS** - jest definiowany jako stosunek średniej geometrycznej do średniej arytmetycznej widma mocy. Płaskość widmowa jest obliczana przez podzielenie średniej geometrycznej widma mocy przez średnią arytmetyczną widma mocy.
2. **SPECTRAL ROLL OFF** - miara „skośności” kształtu widmowego.



Służy do odróżnienia mowy i muzyki dźwięcznej od bezdźwięcznej (mowa bezdźwięczna ma wysoki udział energii zawartej w zakresie wysokiej częstotliwości widma).

1. **MEL-FREQUENCY CEPSTRUM** (parametry mel-cepstralne) - to parametry szeroko stosowane w akustyce mowy oraz w kompresji sygnałów fonicznych. Powstają z cepstrum sygnału przedstawionego w skali melowej. Skalę melową uzyskuje się oprzez filtrację sygnału bankiem filtrów o charakterystyce trójkątnej. (opis modułu z biometrii)
2. **ENERGY SPECTRAL DENSITY** (Widmowa gęstość energii) - opisuje jaki jest rozkład częstotliwościowy energii (wariancji) sygnału lub szeregu czasowego. Jeśli f(t) jest sygnałem o skończonej energii, całkowalnym z kwadratem to widmowa gęstość Φ(ω) sygnału jest kwadratem modułu ciągłej transformaty Fouriera tego sygnału:



Zaimplementowane nieużywane funkcje:

Uzasadnienie czemu nie były używane

1. OCT (split fft into bandwidth)
2. WAVE TO ARRAY – funkcja jest realizowana w algorytmie głównym.
3. **IMPLEMENTACJA**
4. Kod projektu w całości został napisany w języku programowania **Python 3**. (czemu python jest dobry do przetwarzania sygnałów?)
5. Implementacja parametrów w programie: (czego brakuje?)
6. Używane biblioteki:
   1. Numpy - Moduł Numpy jest podstawowym zestawem narzędzi dla języka Python umożliwiającym zaawansowane obliczenia matematyczne, w szczególności do zastosowań naukowych (tzw. obliczenia numeryczne, jak mnożenie i dodawanie macierzy, diagonalizacja czy odwrócenie, całkowanie, rozwiązywanie równań, itd.)
   2. Matplotlib - [biblioteka](https://pl.m.wikipedia.org/wiki/Biblioteka_programistyczna) do tworzenia [wykresów](https://pl.m.wikipedia.org/wiki/Wykres) dla języka programowania [Python](https://pl.m.wikipedia.org/wiki/Python) i jego rozszerzenia numerycznego [NumPy](https://pl.m.wikipedia.org/w/index.php?title=NumPy&action=edit&redlink=1).
   3. Scipy - pozwala na dostęp do bardziej złożonych i różnorodnych algorytmów wykorzystujących bazę zdefiniowaną w Numpy.
   4. Scikit-image (skimage) – biblioteka zawiera zbiór parametrów do przetwarzania obrazu.
   5. Argparse – zawiera narzędzia do przekazywania parametrów wiersza poleceń.
   6. Librosa
   7. Pandas

1. Użyte funkcje pomocnicze:
   1. STFT function - procedura obliczania STFT polega na podzieleniu dłuższego sygnału czasowego na krótsze segmenty o równej długości, a następnie obliczeniu transformat Fouriera na każdym krótszym segmencie. Jest to widmo Fouriera na każdym krótszym segmencie- zaimplementowane z biblioteki scipy.
   2. Sine gen (sinus generator) – funkcja generuje sygnał sinusoidalny o wybranej częstotliwości i częstotliwości próbkowania, który może być wykorzystany do testowania programu.
   3. Wave open – służy do otwierania sygnału wave i przekazywania próbek do analizy.
   4. Windowing (okienkowanie) – funkcja opisująca sposób pobierania próbek sygnału – to nie jest dobry opis okienkowania
2. Baza dźwięków
3. **WKORZYSTYWANE PLATFORMY ONLINE**
4. **EFEKT KOŃCOWY PROGRAMU**
5. **PODSUMOWANIE PROJEKTU I WNIOSKI**

1. https://kpz.pwr.edu.pl/projekty-zespolowe/ [↑](#footnote-ref-1)