Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
«Высшая школа интеллектуальных систем и суперкомпьютерных технологий»

КУРСОВОЙ ПРОЕКТ

РАСПОЗНАВАНИЕ ДВУХТОНАЛЬНОГО МНОГОЧАСТОТНОГО НАБОРА ТЕЛЕФОННОГО НОМЕРА

по дисциплине «Методы обработки экспериментальных данных»

Выполнил
студент гр. 3540203/10101 В.В. Сухомлинов

Руководитель
доцент ВШИСиСТ, к.ф.-м.н. И.Н. Белых

Санкт-Петербург
2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Двухтональный многочастотный набор (DTMF) — это метод представления цифр клавиатуры телефона тонами для передачи по аналоговому каналу связи. Технология DTMF представляет собой надежную альтернативу роторным телефонным системам и позволяет пользователю вводить данные во время телефонного разговора (рис.0.1). Эта функция позволила создать интерактивные системы автоматического ответа, такие как системы, используемые для телефонного банкинга, маршрутизации звонков в службу поддержки клиентов, голосовой почты и других подобных приложений.

Рис.0.1. Физическое устройство для работы с DTMF-сигналами

Частоты, выбранные для тонов DTMF, имеют некоторые отличительные характеристики и уникальные свойства:

- все тона находятся в слышимом диапазоне частот, что позволяет человеку определить, когда была нажата клавиша;
- ни одна частота не является кратной другой;
- сумма или разность любых двух частот не равна другой выбранной частоте.

Второе и третье свойства упрощают декодирование DTMF и уменьшают количество ложно распознанных тонов. Уникальные свойства позволяют приемникам DTMF определять, когда пользователь нажимает несколько клавиш одновременно.

Цель данной работы заключается в создании инструмента для генерации DTMF-сигналов и распознавания их в звуковом файле соответственно.

Для достижения поставленной цели необходимо выполнить следующие задачи:

– изучить теоретических материалов по моделированию DTMF-сигналов;
– реализовать метод генерации двухтонального многочастотного сигнала;
– изучить материалы по распознаванию DTMF-сигналов;
– выбрать и реализовать одним из методов декодирования.

В результате данной работы предполагается создание программного инструмента, который способен как моделировать DTMF-сигналы из входящего набора символов, так и декодировать звуковую дорожку в сообщение.

ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1. Формат входного сигнала

В качестве входного сигнала был выбран аудиоформат wav по следующим причинам:

- отсутствие сжатия данных;
- наличие готовых инструментов для чтения/записи данных.

1.2. Алгоритм генерации сигнала

DTMF-сигнал представляет собой аддитивную модель двух гармонических процессов:

$$x(t) = A_0 * sin(2 * \pi * n * f_1 * \triangle t) + A_0 * sin(2 * \pi * n * f_2 * \triangle t), \qquad (1.1)$$

где A_0 - амплитуда сигнала, f_1 и f_2 - частоты гармоник, $\triangle t$ - частота дискретизации.

Частоты гармоник берутся по приведённой ниже табл.1.1 из столбца и строки, соответствующих передаваемому символу. Каждая строка набора представлена частотой низкого тона, а каждый столбец - частотой высокого тона.

Таблица 1.1

Таблица соответствия частот и символов DTMF

1209 Гц	**1336 Гц**	**1477 Гц**	**1633 Гц**	
1	2	3	A	**697 Гц**
4	5	6	B	**770 Гц**
7	8	9	C	**852 Гц**
*	0	#	D	**941 Гц**

Процесс генерации сообщения двухтонального многочастотного набора заключается в последовательной генерации множества значений гармоник для каждого символа сообщения с последующей записью в файл.

1.3. Алгоритм Гёрцеля

Для решения задачи детектирования и декодирования тональных сигналов в телефонии обычно применяются две вариации дискретного преобразования Фурье:

быстрое преобразование Фурье (FFT) и алгоритм Гёрцеля. В рамках данной работы разберем далее подробнее последнюю по причине того, что нам заранее известны частотные компоненты, которые мы хотим искать, что уменьшает необходимое количество подсчетов.

Пусть x_n, $n = 0, \ldots, N-1$ — измеренные значения сигнала, которые являются входными данными для дискретного преобразования Фурье, а X_k, $k = 0, \ldots, N-1$ — частотные компоненты дискретного преобразования Фурье, по определению равные $X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn}$. Для расчёта X_k с помощью алгоритма Гёрцеля:

- Последовательно вычисляются члены последовательности s_n для $n = 0,\ldots,N-1$ по рекуррентной формуле $s_n = 2\cos\left(\frac{2\pi k}{N}\right) s_{n-1} - s_{n-2} + x_n$, где $s_{-1} = s_{-2} = 0$, $k = [0.5 + \frac{N*DTMF}{rate}]$, rate - частота дискретизации.
- Искомое значение частотного компонента получается как $X_k = e^{\frac{2\pi i}{N} k} s_{N-1} - s_{N-2}$.

Так как нам фаза сигнала не важна, на втором этапе алгоритма вместо комплексного значения частотного компонента вычислим квадрат его модуля по формуле:

$$|X_k|^2 = s_{N-1}^2 - 2\cos\left(\frac{2\pi k}{N}\right) s_{N-1} s_{N-2} + s_{N-2}^2. \tag{1.2}$$

Следующим этапом выбираются две частоты: с самой большой мощностью и самой маленькой. После происходит поиск элемента в табл.1.1, у которого совпадают значения высокой и низкой частоты с найденными ранее.

1.4. Выводы

ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1. Используемые инструменты

Для реализации был выбран язык программирования Python 3.9. Среда разработки: JetBrains PyCharm. Также, использовались такие пакеты для языка Python, как:

- math, numpy - для расчетов;
- scipy - для работы с wav-файлом.

Для работы с гармоническими процессами использовались ранее реализованные в рамках курса функции библиотеки spbstu-processing-data.

2.2. Модели и константы

Для работы с данными генерируемых сигналов был создан класс Signal, который хранит в себе значения сигнала, частоту дискретизации и интерпретируемый символ.

Значения частот, соответствующих символов приведены в программе в виде констант:

- DTMF_TABLE - словарь символов и частот;
- DTMF_FREQ - массив возможных частот набора;
- DTMF_HIGH - массив высоких частот;
- DTMF_LOW - массив низких частот.

2.3. Чтение и запись

Чтобы записывать значения сигналов в файл был создан класс Writer, который в функции *def write(filename: str, signals: [Signal])* формирует из объектов Signal весь массив значений и передает его на вход функции *write* библиотеки scipy.

Для чтения данных из wav-файла используется класс Reader и функция *def read(filename: str)*, которая обращается к *read* фреймворка scipy.

2.4. Генерация сигнала

Чтобы создать звуковой файл, был написан класс Generator, в котором реализованы две функции:

- *def generate_from(symbols: str, duration, volume, rate) → [Signal]* - принимает на вход строку символов с заданными параметрами продолжительности, громкости и частоты и возвращает массив сгенерированных элементов Signal;
- *def calculate(symbol: str, duration, volume, rate) → Signal* - для входного символа вычисляет значение двух гармоник и их аддитивную модель.

Результат функции *generate_from* передается объекту класса Writer, описанному ранее.

2.5. Распознавание символов

Алгоритм Гёрцеля реализован в рамках класса Goertzel, в котором используются следующие функции и методы:

- *init* - инициализатор класса, в котором заранее подсчитываются значения коэффициентов DTMF-частот;
- *def calc_s_n(self, sample_data)* - вычисляет значения последовательности s_n;
- *def calc_power(self) -> {float: float}* - вычисляет мощность для каждого частотного компонента;
- *def get_number(self, powers)* - на основе полученных мощностей находим необходимый нам символ по таблице DTMF_TABLE;
- *def reset(self)* - для каждого последующего пакета значений сигнала сбрасываем посчитанные значения последовательности s_n.

Чтобы определить символы, которые были закодированы в wav-файле, был определен класс Detector. Он включает в себя одну функцию:

- *def detect(rate, data) -> str* - принимает на вход частоту и значения сигнала, а возвращает строку с распознанным сообщением.

В рамках этапа распознавания разбиваем массив значений сигнала на пакеты (bins), элементы которых поочередно передаем на вход алгоритма Гёрцеля - объекту класса Goertzel. В итоге получаем строку распознанных значений.

2.6. Выводы