**ĐẠI HỌC QUỐC GIA TP. HCM**

###### **TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

###### **KHOA ĐIỆN TỬ - VIỄN THÔNG**

###### **KỸ THUẬT ĐIỆN TỬ - VIỄN THÔNG, HỆ CHẤT LƯỢNG CAO**



##### VÕ MINH CHẤN HUỲNH THÁI TUẤN

22207110 22207098

##### NGUYỄN ANH TUẤN NGUYỄN PHÚC VĨNH

22207099 22207104

TÊN ĐỒ ÁN:

**Autocorrelation and Cross-Correlation: Speech Command recognition**

ĐỒ ÁN MÔN HỌC XỬ LÝ TÍN HIỆU SỐ

ETC10013

NGƯỜI HƯỚNG DẪN:

**Giảng viên:** ĐẶNG LÊ KHOA

**Giảng viên:** NGUYỄN THÁI CÔNG NGHĨA

**TP. Hồ Chí Minh – Năm 2024**

**ĐẠI HỌC QUỐC GIA TP. HCM**

###### **TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

###### **KHOA ĐIỆN TỬ - VIỄN THÔNG**

###### **KỸ THUẬT ĐIỆN TỬ - VIỄN THÔNG, HỆ CHẤT LƯỢNG CAO**



##### VÕ MINH CHẤN HUỲNH THÁI TUẤN

22207110 22207098

##### NGUYỄN ANH TUẤN NGUYỄN PHÚC VĨNH

22207099 22207104

TÊN ĐỒ ÁN:

**Autocorrelation and Cross-Correlation: Speech Command recognition**

ĐỒ ÁN MÔN HỌC XỬ LÝ TÍN HIỆU SỐ

ETC10013

NGƯỜI HƯỚNG DẪN:

**Giảng viên:** ĐẶNG LÊ KHOA

**Giảng viên:** NGUYỄN THÁI CÔNG NGHĨA

**TP. Hồ Chí Minh – Năm 2024**

LỜI CẢM ƠN

Bài báo cáo đồ án môn học được thực hiện tại Khoa Điện tử - Viễn Thông, Trường đại học Khoa Học Tự Nhiên, dưới sự hướng dẫn của giảng viên Đặng Lê Khoa và Nguyễn Thái Công Nghĩa.

Trước tiên nhóm chúng em xin bày tỏ lòng biết ơn sâu sắc đến Thầy Đặng Lê Khoa và Thầy Nguyễn Thái Công Nghĩa đã giúp nhóm chúng em đến với đồ án này. Thầy đã tận tình giảng dạy, hướng dẫn giúp chúng em tiếp cận và đạt được thành công trong việc hoàn thành đồ án này. Thầy luôn động tận tâm động viên, khuyến khích và chỉ dẫn giúp chúng em hoàn thành được đồ án báo cáo này.

Nhóm chúng em cũng xin cảm ơn các bạn trong lớp 22DTV - CLC3 Trường đại học Khoa Học Tự Nhiên TP. Hồ Chí Minh đã góp phần hỗ trợ nhóm chúng em thực thi đồ án.

Nhóm chúng em xin bày tỏ lòng biết ơn sâu sắc tới các Thầy/Cô thuộc khoa Điện Tử - Viễn Thông, đã tạo mọi điều kiện thuận lợi giúp đỡ nhóm chúng em trong quá trình học tập và nghiên cứu.

Nhóm chúng em rất mong nhận được sự góp ý và chỉ bảo từ quý Thầy/Cô.

*TP. Hồ Chí Minh, ngày 19 tháng 7 năm 2024*

*Tác giả*

**Võ Minh Chấn - Huỳnh Thái Tuấn**

**Nguyễn Anh Tuấn – Nguyễn Phúc Vĩnh**

MỤC LỤC

[NGƯỜI HƯỚNG DẪN: 1](#_Toc172630137)

[NGƯỜI HƯỚNG DẪN: 2](#_Toc172630138)

[LỜI CẢM ƠN i](#_Toc172630139)

[MỤC LỤC 1](#_Toc172630140)

[DANH MỤC CÁC HÌNH 2](#_Toc172630141)

[CHƯƠNG 1: GIỚI THIỆU 3](#_Toc172630142)

[**1.1.** **Đặt Vấn Đề** 3](#_Toc172630143)

[**1.2.** **Mục tiêu đề tài** 3](#_Toc172630144)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 3](#_Toc172630145)

[**2.1.** **Ngôn ngữ lập trình** 3](#_Toc172630146)

[2.1.1. ANACONDA 3](#_Toc172630147)

[2.1.2. JYPYTER NOTEBOOK 3](#_Toc172630148)

[**2.2.** **Autocorrelation và Cross-Correlation** 4](#_Toc172630149)

[2.2.1. Autocorrelation 4](#_Toc172630150)

[2.2.2. Cross-Correlation 4](#_Toc172630151)

[**2.3.** **Nguyên lý hoạt động:** 5](#_Toc172630152)

[CHƯƠNG 3: THỰC THI 6](#_Toc172630153)

[**3.1.** **Thực thi code trên ứng dụng Jupyter Notebook** 6](#_Toc172630154)

[**3.2.** **Thực thi thực tế bằng code trên ứng dụng Jupyter Notebook** 11](#_Toc172630155)

[CHƯƠNG 4: PHÂN TÍCH KẾT QUẢ THỰC THI 11](#_Toc172630156)

[**4.1.** **Mô tả kết quả thực thi và kết quả mô phỏng** 11](#_Toc172630157)

[**4.2.** **Phân tích kết quả** 12](#_Toc172630158)

[CHƯƠNG 5: KẾT QUẢ CỦA CẢ PROJECT VÀ HƯỚNG PHÁT TRIỂN 12](#_Toc172630159)

[**5.1.** **Kết quả của cả project và hướng phát triển** 12](#_Toc172630160)

[TÀI LIỆU THAM KHẢO 13](#_Toc172630161)

**DANH MỤC CÁC HÌNH**

[**Hình ảnh 1:** Logo của *Anaconda* 3](#_Toc153323703)

[**Hình ảnh 2:** Logo của *Jupyter Notebook* 4](#_Toc153323704)

[**Hình ảnh 3:** Biểu đồ kết quả mô phỏng 12](#_Toc153323704)

# GIỚI THIỆU

## **Đặt Vấn Đề**

Trong bối cảnh nhu cầu về nhà thông minh ngày càng gia tăng, các công nghệ tiên tiến được tích hợp vào nhiều lĩnh vực nhằm nâng cao hiệu quả và tiện ích cho cuộc sống con người. Một trong những ứng dụng nổi bật là hệ thống nhận dạng giọng nói, đóng vai trò quan trọng trong việc tương tác giữa con người và máy móc. Những hệ thống này không chỉ mang lại sự tiện lợi mà còn mở ra nhiều ứng dụng thực tiễn như trợ lý ảo, điều khiển thiết bị thông minh, và hỗ trợ người khuyết tật.

Nhận thấy tiềm năng của công nghệ này trong việc cải thiện cuộc sống hàng ngày, nhóm chúng em đã nghiên cứu và xây dựng một hệ thống nhận dạng lệnh giọng nói dựa trên các phương pháp Autocorrelation và Cross-Correlation. Hệ thống này không chỉ cho phép nhận diện và phân loại các lệnh giọng nói một cách chính xác mà còn đóng góp quan trọng vào việc điều khiển tự động các thiết bị điện tử trong ngôi nhà thông minh. Với hệ thống này, người dùng có thể thực hiện các thao tác như mở cửa, bật đèn, và điều khiển nhiều thiết bị khác chỉ bằng giọng nói.

Dự án này không chỉ là một minh chứng cho khả năng ứng dụng kiến thức lý thuyết vào thực tiễn mà còn là một bước đột phá trong việc phát triển các công nghệ tương tác người-máy. Chúng em tin rằng hệ thống này sẽ mang lại nhiều tiện ích và cải tiến cho cuộc sống con người, đáp ứng nhu cầu ngày càng cao về các giải pháp nhà thông minh và tự động hóa.

## **Mục tiêu đề tài**

Xây dựng mô hình nhận dạng lệnh giọng nói sử dụng các phương pháp Autocorrelation và Cross-Correlation để đáp ứng yêu cầu ứng dụng thực tiễn và phù hợp với kiến thức đã học. Mô hình này sẽ nhận diện và phân loại các lệnh giọng nói, giúp điều khiển các thiết bị thông minh một cách tự động và hiệu quả.

Phát triển mô hình mô phỏng dựa trên lý thuyết xử lý tín hiệu số, đồng thời áp dụng các kiến thức đã học để giải thích nguyên lý hoạt động của các phương pháp Autocorrelation và Cross-Correlation. Ngoài ra, dự án sẽ cung cấp một cái nhìn sâu hơn về cách thức xử lý tín hiệu giọng nói, từ đó đánh giá mô hình thực tiễn dựa trên cơ sở lý thuyết đã học.

# CƠ SỞ LÝ THUYẾT

## **Ngôn ngữ lập trình**

### ANACONDA

****

**Hình 1**: *“Logo của Anaconda”*

**Anaconda** là một bản phân phối các ngôn ngữ lập trình Python và R cho tính toán khoa học (khoa học dữ liệu, ứng dụng học máy, xử lý dữ liệu quy mô lớn, phân tích dự đoán, ...), nhằm mục đích đơn giản hóa việc quản lý và triển khai gói.

### JYPYTER NOTEBOOK

****

**Hình 2**: *“Logo của Jupyter Notebook”*

**Jupyter Notebook** (trước đây là IPython Notebook) là một môi trường tính toán tương tác dựa trên web để tạo tài liệu sổ ghi chép. Jupyter Notebook được xây dựng bằng một số thư viện mã nguồn mở, bao gồm IPython, ZeroMQ, Tornado, jQuery, Bootstrap và MathJax. Ứng dụng Jupyter Notebook là một REPL dựa trên trình duyệt chứa danh sách các ô nhập / đầu ra có thể chứa mã, văn bản (sử dụng Github Flavored Markdown), toán học, biểu đồ và đa phương tiện.

## **Autocorrelation và Cross-Correlation**

### Autocorrelation

**- Khái niệm :** Đo lường sự tương quan của tín hiệu với chính nó ở các thời điểm khác nhau. Dùng để phân tích độ tương quan, phát hiện chu kỳ của tín hiệu.

- Autocorrelation của một tín hiệu x[n] tại một độ trễ (lag) k được định nghĩa là:

- **Giải thích công thức** :

1. **Tích giá trị tín hiệu tại hai thời điểm**: Trong công thức, x[n]x[n−k] là tích của giá trị tín hiệu tại thời điểm n và n−k. Điều này cho phép đánh giá mức độ tương quan giữa hai giá trị này.

2. **Tổng hợp các tích**: Tổng hợp tích của các cặp giá trị tín hiệu này qua toàn bộ chuỗi (hoặc một khoảng hữu hạn) cho chúng ta một chỉ số thể hiện mức độ tương quan tổng thể của tín hiệu tại độ trễ k.

3. **Phạm vi tổng**: Trong thực tế, tín hiệu thường chỉ tồn tại trong một khoảng thời gian hữu hạn, do đó tổng chỉ chạy từ 0 đến N−1(với N là độ dài của tín hiệu).

**- Ví dụ Trực Quan:**

Giả sử bạn có một chuỗi tín hiệu ngắn x[n]={1,2,3,4}, bạn muốn biết mức độ tương quan của tín hiệu với chính nó khi trễ đi một bước (k=1). Theo công thức, bạn tính toán như sau:

Rx[1]=x[0]⋅x[0+1]+x[1]⋅x[1+1]+x[2]⋅x[2+1]

Rx[1]=1⋅2+2⋅3+3⋅4=2+6+12=20

Kết quả này cho thấy mức độ tương quan của tín hiệu khi trễ đi một bước là 20, một con số biểu thị mức độ tương quan mạnh hay yếu tùy vào tín hiệu.

### Cross-Correlation

**- Khái Niệm :** Đo lường mức độ tương quan giữa hai chuỗi tín hiệu khác nhau. Nó giúp tìm ra mối liên hệ và sự tương tự giữa hai tín hiệu tại các thời điểm khác nhau

Cross-correlation của hai tín hiệu x[n] và y[n] tại một độ trễ (lag) k được định nghĩa là:

- **Giải thích công thức** :

**1. Tích giá trị tín hiệu tại hai thời điểm**: Trong công thức, x[n]y[n−k] là tích của giá trị tín hiệu x tại thời điểm n và giá trị tín hiệu y tại thời điểm n−k. Điều này cho phép đánh giá mức độ tương quan giữa hai giá trị này.

**2. Tổng hợp các tích**: Tổng hợp tích của các cặp giá trị tín hiệu này qua toàn bộ chuỗi (hoặc một khoảng hữu hạn) cho chúng ta một chỉ số thể hiện mức độ tương quan tổng thể của hai tín hiệu tại độ trễ k.

**- Ví dụ Trực Quan:**

Giả sử chúng ta có hai tín hiệu x[n]={1,2,3} và y[n]={4,5,6} Ta tính hàm cross-correlation với độ trễ k từ 0 đến 2:

Rxy[0]=1∗4+2∗5+3∗6=4+10+18=32

Rxy[1]=1∗5+2∗6=5+12=17

Rxy[2]=1∗6=6

Hàm cross-correlation của tín hiệu x[n] và y[n] là Rxy[k]={32,17,6}

## **Nguyên lý hoạt động:**

**Thu thập tín hiệu âm thanh**:

Tín hiệu âm thanh từ micro được ghi lại trong 3 giây với tần số lấy mẫu là 44.1 kHz.

Tín hiệu được lưu trữ dưới dạng các mảng số thực (float64).

**Tiền xử lý tín hiệu**:

Tín hiệu được làm phẳng (flatten) để loại bỏ các chiều không cần thiết.

**Tính Cross-Correlation**:

Cross-correlation được tính giữa lệnh kiểm tra và từng lệnh mẫu ("open", "close", "light").

Giá trị lớn nhất của cross-correlation được sử dụng để đánh giá mức độ tương quan.

**Nhận dạng lệnh giọng nói:**

Lệnh với giá trị cross-correlation lớn nhất được xác định là lệnh giọng nói tương ứng với lệnh kiểm tra.

**Hiển thị tín hiệu âm thanh**:

Các tín hiệu âm thanh được vẽ trên đồ thị để kiểm tra và phân tích.

# THỰC THI

## **Thực thi code trên ứng dụng Jupyter Notebook**

***#Import các thư viện***

import numpy as np

import sounddevice as sd

#Thư viện sounddevice là một thư viện Python cho phép ghi âm và phát lại âm thanh. Nó cung cấp một API dễ sử dụng để làm việc với âm thanh.

from scipy.signal import correlate

# Hàm correlate từ thư viện scipy.signal được sử dụng để tính toán độ tương quan chéo (cross-correlation) giữa hai tín hiệu.

# Độ tương quan chéo là một thước đo về mức độ giống nhau của hai tín hiệu tại các độ dịch khác nhau.

import matplotlib.pyplot as plt

***# Thực hiện thiết lặp các thông số ghi âm***

fs = 44100 # Tần số lấy mẫu (Hz)

duration = 3 # Thời gian ghi âm (giây)

#Hàm ghi âm

def record\_sample(fs, duration):

print("Đang ghi âm...")

sample = sd.rec(int(duration \* fs), samplerate=fs, channels=1, dtype='float64') #sd.rec():

#int(duration \* fs): Số mẫu cần ghi âm. Tính bằng cách nhân thời gian ghi âm (duration) với tần số lấy mẫu (fs).

#samplerate=fs: Tần số lấy mẫu, xác định số lần mẫu được lấy mỗi giây.

#channels=1: Số lượng kênh âm thanh, ở đây là một kênh (mono).

#dtype='float64': Định dạng dữ liệu của các mẫu âm thanh

#float64 nghĩa là mỗi mẫu âm thanh sẽ được lưu trữ dưới dạng số thực 64 bit.

#Cách hoạt động: sd.rec() bắt đầu ghi âm và trả về một mảng NumPy chứa các mẫu âm thanh đã được ghi.

sd.wait() # Đợi cho đến khi ghi âm xong

#Hàm này sẽ chặn (block) chương trình cho đến khi tất cả các mẫu âm thanh đã được ghi xong.

#Điều này đảm bảo rằng không có mẫu âm thanh nào bị mất và quá trình ghi âm hoàn tất trước khi tiếp tục các bước tiếp the

return sample.flatten()

#chuyển đổi mảng 2D thành mảng 1D

#Điều này giúp đơn giản hóa việc xử lý và phân tích dữ liệu âm thanh sau khi ghi âm

***#Thực hiện ghi âm.***

print('Ghi âm lệnh "Unclock":')

open = record\_sample(fs, duration) # "lệnh muốn ghi âm" = record\_sample(fs, duration)

print('Ghi âm lệnh "Iamgoingout":')

close = record\_sample(fs, duration)

print('Ghi âm lệnh "Turn\_on\_light":')

light = record\_sample(fs,duration)

print('Ghi âm lệnh kiểm tra:')

testSample = record\_sample(fs, duration)

***# Tính tương quan chéo giữa tín hiệu kiểm tra và các tín hiệu lệnh qua hàm cross-correlation***

correlationOpen = np.max(correlate(testSample, open))

#correlate(testSample, open): Tính toán độ tương quan chéo giữa tín hiệu kiểm tra testSample và tín hiệu lệnh open

#np.max(...): Lấy giá trị lớn nhất trong mảng kết quả của hàm

correlationClose = np.max(correlate(testSample, close))

#Tương tự như trên, nhưng với tín hiệu lệnh close.

correlationLight = np.max(correlate(testSample, light))

#Tương tự như trên, nhưng với tín hiệu lệnh light.

***# So Sánh giá trị độ tương quan chéo***

if correlationOpen > correlationClose and correlationOpen > correlationLight:

print('Lệnh "Unclock" được nhận dạng.')

#Kiểm tra xem giá trị độ tương quan cao nhất với lệnh "Unclock" (correlationOpen)

#có lớn hơn các giá trị độ tương quan với lệnh "Iam\_going\_out" (correlationClose) và lệnh "Turn\_on\_light" (correlationLight) hay không.

#Nếu đúng, tức là tín hiệu kiểm tra giống lệnh "Unclock" nhất, in ra thông báo "Lệnh 'Unclock' được nhận dạng."

elif correlationClose > correlationLight and correlationClose > correlationOpen:

print('Lệnh "Iam\_going\_out" được nhận dạng.')

#Tương tự như trên, nhưng với tín hiệu lệnh close.

elif correlationLight > correlationOpen and correlationLight > correlationClose:

print('Lệnh "Turn\_on\_light" được nhận dạng.')

#Tương tự như trên, nhưng với tín hiệu lệnh light

***#Vẽ phổ âm thanh thu được***

t = np.linspace(0, duration, len(open))

#Tạo một mảng thời gian t với các giá trị từ 0 đến duration để định vị thời gian cho mỗi mẫu dữ liệu âm thanh.

plt.figure(figsize=(12, 8))

plt.subplot(4, 1, 1)

plt.plot(t, open, 'b')

plt.title('Lệnh kiểm tra Unclock ')

plt.ylabel('Biên độ')

plt.subplot(4, 1, 2)

plt.plot(t, close, 'g')

plt.title('Lệnh kiểm tra Iamgoingout ')

plt.ylabel('Biên độ')

plt.subplot(4, 1, 3)

plt.plot(t, light, 'r')

plt.title('Lệnh kiểm tra Turn\_on\_light')

plt.xlabel('Thời gian (s)')

plt.ylabel('Biên độ')

plt.subplot(4, 1, 4)

plt.plot(t, testSample, 'r')

plt.title('Lệnh kiểm tra')

plt.xlabel('Thời gian (s)')

plt.ylabel('Biên độ')

plt.tight\_layout()

plt.show()

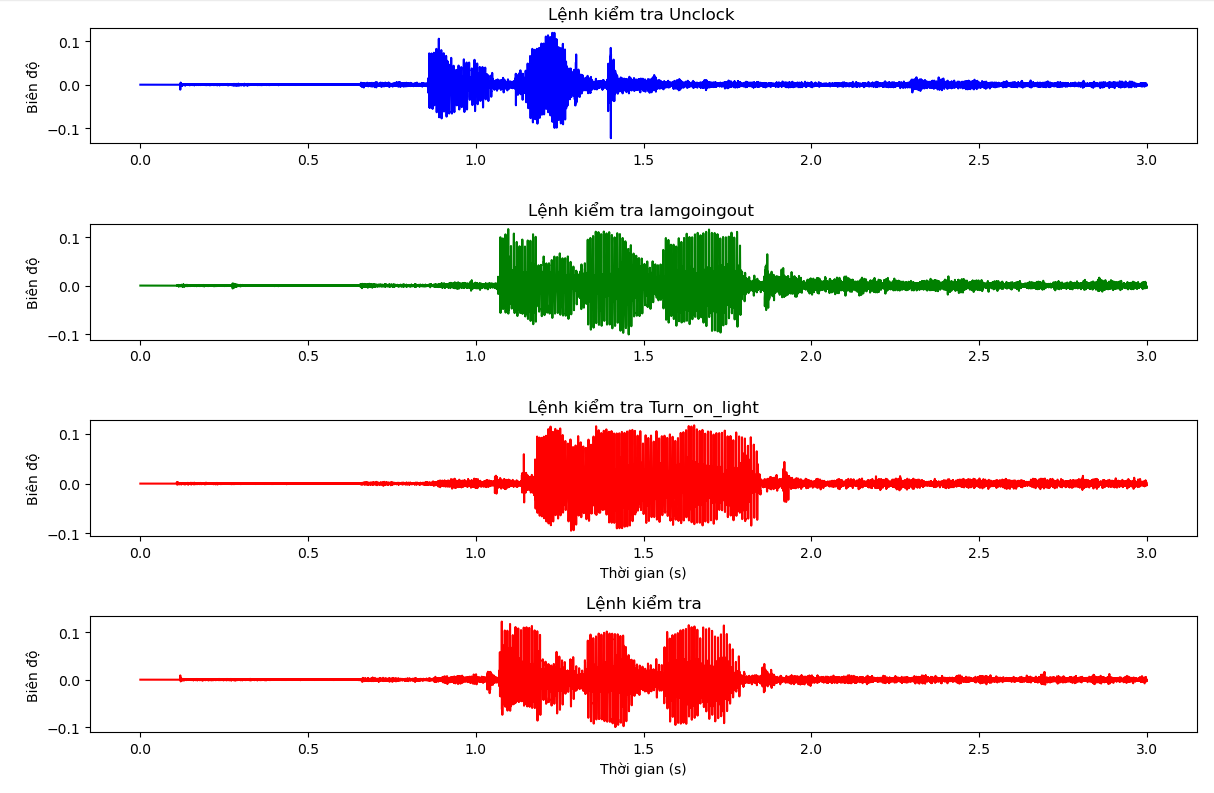
## **Thực thi thực tế bằng code trên ứng dụng Jupyter Notebook**

Video thực tế:

<https://drive.google.com/file/d/1nNcm4msWuHpGYSgcC9NUNZ9zcoqu-jHc/view?usp=sharing>

# PHÂN TÍCH KẾT QUẢ THỰC THI

## **Mô tả kết quả thực thi và kết quả mô phỏng**



**Hình 3** : “ *Biểu đồ kết quả mô phỏng* “

Đồ thị thứ nhất: Biểu đồ của tín hiệu open (lệnh "Unclock").

Đồ thị thứ hai: Biểu đồ của tín hiệu close (lệnh "Iamgoingout").

Đồ thị thứ ba: Biểu đồ của tín hiệu light (lệnh "Turn\_on\_light").

Đồ thị thứ tư: Biểu đồ của tín hiệu testSample (lệnh kiểm tra).

## **Phân tích kết quả**

- Kết quả của các đồ thị này sẽ cho thấy các đặc điểm biên độ và thời gian của các tín hiệu âm thanh đã ghi và lệnh kiểm tra, giúp bạn hiểu rõ hơn về sự tương đồng giữa chúng.

- Cụ thể như hình trên ta thấy đồ thị Lệnh kiểm tra có độ tương đồng chéo cao nhất với đồ thị Lệnh kiểm tra Iamgoingout nên hệ thống sẽ trả ra lệnh Iamgoingout.

# KẾT QUẢ CỦA CẢ PROJECT VÀ HƯỚNG PHÁT TRIỂN

## **Kết quả của cả project và hướng phát triển**

Dự án này có thể nhận dạng các lệnh giọng nói cơ bản dựa trên độ tương quan chéo của tín hiệu âm thanh. Cụ thể:

1. Nhận dạng lệnh giọng nói:

Hệ thống có thể phân biệt và nhận dạng các lệnh giọng nói đã được ghi trước đó như "Unclock", "Iamgoingout", và "Turn\_on\_light".

Tính toán độ tương quan chéo giữa tín hiệu kiểm tra và các tín hiệu lệnh đã ghi cho phép xác định lệnh nào có độ tương quan cao nhất với tín hiệu kiểm tra, từ đó xác định lệnh được nhận dạng.

1. Trực quan hóa tín hiệu âm thanh:

Hệ thống vẽ đồ thị của các tín hiệu âm thanh đã ghi và tín hiệu kiểm tra, cho phép người dùng nhìn thấy trực quan sự khác biệt và tương đồng giữa các tín hiệu.

Hướng phát của hệ thống **"Autocorrelation and Cross-Correlation: Speech Command recognition** ":

1. **Tăng số lượng lệnh nhận dạng**

- Thêm nhiều lệnh mới: Mở rộng hệ thống để nhận dạng nhiều lệnh giọng nói hơn. Ví dụ: "Turn off light", "Open window", "Close window", "Play music", "Stop music", "Lock door", "Unlock door", "Increase temperature", "Decrease temperature", v.v.

**2) Cải thiện độ chính xác bằng Autocorrelation**

**-** Áp dụng autocorrelation để phân tích tín hiệu giọng nói và trích xuất các đặc trưng đặc biệt của mỗi lệnh. Autocorrelation giúp nhận diện các mẫu lặp lại trong tín hiệu, giúp hệ thống nhận dạng tốt hơn ngay cả khi có nhiễu.

TÀI LIỆU THAM KHẢO

<https://www.investopedia.com/terms/a/autocorrelation.asp>

<https://www.youtube.com/watch?v=txmLrQ8OZuA>