NHÀ THÔNG MINH ĐIỀU KHIỂN THIẾT BỊ BẰNG WEB SERVER VÀ GIỌNG NÓI TRÊN NỀN TẢNG PYTHON

SMART HOME CONTROL DEVICES BY WEB SERVER AND VOICE USING PYTHON

Phan Anh Tuấn[1,*], Nguyễn Đức Nam[1], Trần Thị Thảo[1], Đinh Thị Kim Phượng[2]

TÓM TẮT

Bài báo trình bày phương pháp nhận dạng tiếng nói cho việc điều khiển thiết bị trong ngôi nhà thông minh. Thực hiện trên máy tính nhỏ nhằm tăng khả năng triển khai của hệ thống. Bằng việc nhận dạng sử dụng CMU Sphinx, huấn luyện các từ để điều khiển các thiết bị. Hệ thống sử dụng Raspberry Pi bộ xử lý trung tâm để truyền, nhận tín hiệu và điều khiển thiết bị thông qua giọng nói hoặc web server.

Từ khóa: *Nhà thông minh, CMU Sphinx, Raspberry Pi.*

ABSTRACT

The paper presents method of speech recognition for controlling devices in smart homes. Perform on small computers to increase the implementation of the system. By identification uses CMU Sphinx train words to control devices. The system uses the Raspberry Pi central processor to transmit, receive signals and control devices via voice or web server.

Keywords: *Smart home, CMU Sphinx, Raspberry Pi.*

[1]Lớp TT&MMT1, Khoa Điện tử, Trường Đại học Công nghiệp Hà Nội
[2]Khoa Điện tử, Trường Đại học Công nghiệp Hà Nội
[*]Email: tuanlion1998@agrexvn.com

1. GIỚI THIỆU

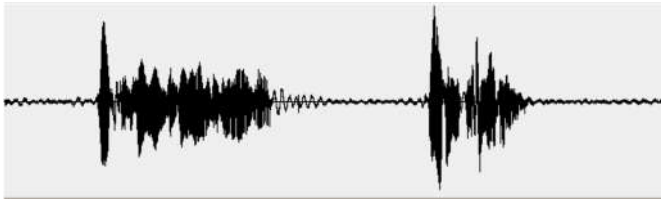
Với sự phát triển trong thời đại công nghệ thông tin, các chương trình giao tiếp người dùng ngày càng đòi hỏi sự thân thiện và hiệu năng mạnh mẽ. Nhu cầu giao tiếp với thiết bị máy bằng tiếng nói trở nên cần thiết, đó là phương thức giao tiếp thông minh và tự nhiên nhất. Trong những năm gần đây, công nghệ này đã có mặt trên rất nhiều thiết bị và chủng loại, từ máy tính đến điện thoại di động, và các thiết bị nhúng khác, các thiết bị ngày càng trở nên nhỏ hơn về kích thước. Tuy nhiên các thiết bị càng nhỏ cũng làm giới hạn về chức năng. Tiếng nói có khả năng điều khiển và tương tác phức tạp với hệ thống nhúng. Nhận diện tiếng nói được phân loại như nhận diện các từ đã được nối với với nhau và nhận biết từng từ một cách độc lập. Đối với hệ thống nhúng thì sử dụng nhận diện từng từ độc lập với nhau có hiệu quả hơn cả. Tất nhiên, hiện nay công nghệ giọng nói vẫn chỉ mới ở giai đoạn đầu chứ chưa thể nào thay thế hoàn toàn bàn phím ảo/vật lý hoặc các nút trên màn hình. Tuy nhiên, chúng ta đang dần tiến đến một kỷ nguyên hiện đại hơn, các ứng dụng giọng nói cũng dần dần được hoàn thiện. Càng ngày những phần mềm nhận dạng giọng nói càng thông minh hơn, khả năng nhận dạng chính xác hơn, thông tin trả về cũng hữu ích và đa dạng hơn. Người ta còn áp dụng cả những kỹ thuật như data mining (khai thác dữ liệu theo chiều sâu), machine learning (cho phép máy móc tự học hỏi thói quen, hành vi của người dùng) nhằm cải thiện hiệu suất làm việc của công nghệ nhận dạng giọng nói nữa.

Thông thường, nhận dạng tiếng nói là một loại mẫu nhận dạng dựa trên huấn luyện và nhận dạng. Phương pháp sử dụng là HMM (Hidden Markov Model) và thư viện sử dụng là bộ nhận dạng HTK. Tín hiệu tiếng nói người dùng được lấy bởi micro USB đã được kết nối với hệ thống. Để xử lý tín hiệu nhận dạng bài báo sử dụng bộ thư viện nhận dạng tiếng nói CMU Sphinx nhận dạng các câu lệnh điều khiển thiết bị bằng tiếng Việt. Chương trình điều khiển được viết bằng ngôn ngữ Python và được biên dịch trên máy tính nhúng Raspberry Pi để điều khiển các thiết bị đèn, quạt, cửa, điều hòa,... trực tiếp qua hàng chân GPIO (General Purpose Input/ Output) trên Raspberry Pi bằng giọng nói hoặc thông qua Web server.

2. NHẬN DẠNG GIỌNG NÓI VÀ FRAMEWORK

2.1. Cấu trúc của giọng nói

Giọng nói là một xử lý động lực học mà không có những phần đặc biệt rõ ràng. Nó rất hữu ích trong việc biên tập âm thanh và nhìn vào bản ghi âm của giọng nói và lắng nghe nó.



Hình 1. Ví dụ bản ghi âm giọng nói

Tất cả các mô tả hiện đại của giọng nói là một số mức độ xác suất. Điều đó có nghĩa là không có ranh giới nhất định giữa các đơn vị, hoặc giữa các từ. Việc chuyển đổi giọng nói sang văn bản và một vài ứng dụng khác của giọng nói là không bao giờ đúng 100%. Đó là một ý tưởng khá tuyệt vời dành cho những nhà phát triển phần mềm, người mà thường làm việc với những hệ thống định luật.

Và nó tạo ra rất nhiều các vấn đề riêng chỉ có trong công nghệ xử lý giọng nói [1].

2.2. Cách thức nhận dạng giọng nói

Nhận dạng tiếng nói là một quá trình phức tạp bao gồm nhiều khâu biến đổi. Tín hiệu tiếng nói phát ra là tương tự. Từ quá trình lấy mẫu, lượng tử hóa và mã hóa để thu được tín hiệu số. Các mẫu tín hiệu này được trích chọn đặc trưng. Những đặc trưng này sẽ là đầu vào của quá trình nhận dạng. Hệ thống nhận dạng sẽ đưa ra kết quả nhận dạng. Tín hiệu tiếng nói đầu tiên được tiền xử lý và rút trích đặc trưng. Kết quả thu được sau quá trình này là tập các đặc trưng âm học (acoustic features). Để có thể thực hiện việc so sánh với các tham số đầu vào của hệ thống nhận dạng, trước hết hệ thống phải được huấn luyện và xây dựng các đặc trưng [2].

Trong quá trình huấn luyện, hệ thống dùng các vector đặc trưng được đưa vào để ước lượng, tính toán các tham số cho các mẫu tham khảo. Một mẫu tham khảo chính là bản mẫu dùng để so sánh và nhận dạng, các mẫu tham khảo này mô phỏng cho một từ, một âm tiết, hoặc một âm vị. Trong quá trình nhận dạng, dãy các vector đặc trưng được so sánh với các mẫu tham khảo. Sau đó, hệ thống tính toán độ tương đồng của dãy vector đặc trưng và mẫu tham khảo. Việc tính toán độ tương đồng được được thực hiện bằng cách áp dụng các thuật toán đã được chứng minh hiệu quả như thuật toán Vitertbi (trong Markov ẩn). Mẫu có độ tương đồng cao nhất là kết quả của quá trình nhận dạng. Có 3 phương pháp phổ biến được sử dụng trong nhận dạng tiếng nói hiện nay: phương pháp âm học - ngữ âm học, phương pháp nhận dạng mẫu, phương pháp ứng dụng trí tuệ nhân tạo [3].

2.3. Bộ Framework CMU Sphinx nhận dạng giọng nói

Sphinx [4] là một bộ thư viện nhận dạng tiếng nói mạnh mẽ và được sử dụng rất nhiều trong các ứng dụng trong cuộc sống. Đó là nhờ các đặc điểm:

- Hỗ trợ nhận dạng tiếng nói ở chế độ trực tiếp hoặc chia lô, có khả năng nhận dạng tiếng nói rời rạc và liên tục.

- Là một hệ thống nhận dạng đồ sộ nhưng có khả năng tháo lắp rất linh động. Hỗ trợ sẵn đầy đủ các tính năng đáp ứng nhu cầu nhận dạng như xây dựng các bộ lọc, các hàm cửa sổ, các phép biến đổi…

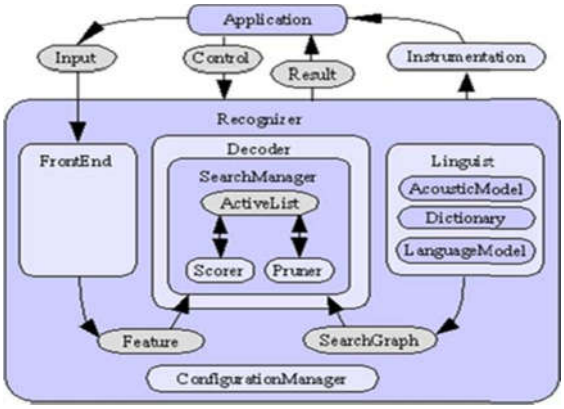
- Hỗ trợ nhiều mô hình ngôn ngữ dạng ASCII và các phiên bản nhị phân của unigram, bigram, trigram, Java Speech API Grammar Format (JSGF) và ARPAformat FST grammars.

- Xây dựng sẵn các thuật toán tìm kiếm tối ưu (breath first, word pruning), dễ dàng tinh chỉnh cho phù hợp với nhu cầu nhận dạng.

Hai tiến trình nòng cốt của Sphinx là: Huấn luyện (Training) và Nhận dạng (Decoding). Trong đó:

+) Huấn luyện: Tiến trình huấn luyện xây dựng cơ sở dữ liệu ngôn ngữ linguist, thành phần quan trọng nhất của hệ thống nhận dạng tiếng nói Sphinx, cho hệ thống (gồm 2 phần: âm học và ngôn ngữ) Bước khởi tạo của quá trình huấn luyện chính là định nghĩa danh sách các âm thanh và từ điển cho mô hình. Sau đó chúng ta sẽ ghi âm lại giọng nói của chúng ta để xây dựng một tập hợp các âm thanh (corpora) làm nguyên liệu cho quá trình Training. Trong quá trình training, lời nói của chúng ta sẽ được chuyển thành dạng Cepstral (dưới dạng vector đặc trưng). Dựa trên các vector này, Hidden Markov Model (HMM) được xây dựng cho mỗi âm trong danh sách âm. Sau khi training, chúng ta sẽ có một bộ ngôn ngữ linguist được xây dựng trên HMM mà có thể giúp ta nhận dạng lời nói của bất cứ người nào để nhận dạng giọng nói được mã hóa từ các âm thanh trong text form.

+) Nhận dạng: Nhiệm vụ chính của tiến trình này là tìm ra từ thích hợp nhất với giọng nói đầu vào. Bước đầu của tiến trình nhận dạng cũng giống với huấn luyện, lời nói sẽ được ghi âm, chuyển thành dạng vector đặc trưng bởi thiết bị đầu cuối. Mô hình ngôn ngữ Linguist sẽ sinh SearchGraph mà dùng để xác định từ bởi bộ giải mã decoder. Bộ giải mã này, sẽ dựa vào SearchGraph và vector đặc trưng, nó sẽ áp dụng thuật toán tìm kiếm để tìm ra kết quả tốt nhất. Tiến trình nhận dạng được đặc tả rất chi tiết trong biểu đồ hình 2.



Hình 2. Kiến trúc bộ thư viện nhận dạng [4]

Kiến trúc bộ thư viện nhận dạng bao gồm các phần sau:

Bộ ngoại vi (FrontEnd): Xử lý tín hiệu từ bên ngoài, thực hiện qua một số bộ lọc và xử lý dữ liệu cho ra kết quả là một tập các vector đặc trưng.

Bộ ngôn ngữ (Linguist): bằng các công cụ và phương pháp ngôn ngữ, đọc vào các tập tin cấu trúc của một ngôn ngữ rồi mô hình hóa chúng vào đồ thị. Ở bộ này cấu tạo khá phức tạp vì nó quy định hầu như toàn bộ phạm vi ngôn ngữ mà chúng ta cần nhận dạng, nó gồm các thành phần nhỏ sau: Mô hình ngôn ngữ: Đọc vào tập tin cấu trúc ngôn ngữ ở cấp độ là các từ. Thành phần này có vai trò quan trọng xác định những thứ hệ thống cần nhận dạng. Cấu trúc ngôn ngữ sẽ được mô hình hóa ở đây theo hai mô hình: ***Mô hình graph-driven grammar:*** Biểu diễn một đồ thị từ có hướng. Mỗi nút biểu diễn một từ đơn và mỗi cung là xác suất dịch chuyển sang một từ. ***Mô hình stochastic N-Gram:*** Mô hình này cung cấp các xác suất cho các từ được có dựa vào việc quan sát N-1 từ đứng trước.

SimpleWordListGrammar: Định nghĩa một từ dựa trên danh sách các từ. Một tham số tùy chọn chỉ ra ngữ pháp có lặp hay không. Nếu không lặp, ngữ pháp sẽ dùng được dùng cho một nhận dạng từ tách biệt. Nếu lặp, nó sẽ được dùng để hỗ trợ liên kết nhận dạng từ tầm thường.

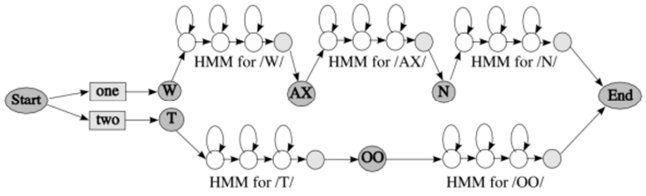
JSGFGrammar: Hỗ trợ JavaTM Speech API Grammar Format (JSGF), định nghĩa một biểu diễn theo BNF, độc lập nền tảng, Unicode của các ngữ pháp.

LMGrammar: Định nghĩa một ngữ pháp dựa trên một mô hình ngôn ngữ thống kê. LMGrammar phát sinh một nút ngữ pháp mỗi từ và làm việc tốt với các unigram và bigram, xấp xỉ 1000 từ. FSTGrammar: Hỗ trợ một bộ chuyển đổi trạng thái giới hạn (finite-state tranducer) trong định dạng ngữ pháp ARPA FST. SimpleNGramModel Cung cấp hỗ trợ cho các mô hình ASCII N-Gram trong định dạng ARPA. SimpleNGramModel không cố làm tối ưu việc sử dụng bộ nhớ, do đó nó làm việc tốt với các mô hình ngôn ngữ nhỏ.

Bộ từ điển: Thành phần này cung cấp cách phát âm cho các từ ta đã xây dựng trong mô hình ngôn ngữ.

Mô hình âm học: Cung cấp một ánh xạ giữa một đơn vị tiếng nói và một HMM có thể được đánh giá dựa vào các đặc trưng được cung cấp bởi bộ ngoại vi. Các ánh xạ có thể đưa thông tin vị trí của từ và ngữ cảnh từ thành phần mô hình ngôn ngữ. Định nghĩa ngữ cảnh này được xây dựng từ cấu trúc ngữ pháp của mô hình ngôn ngữ.

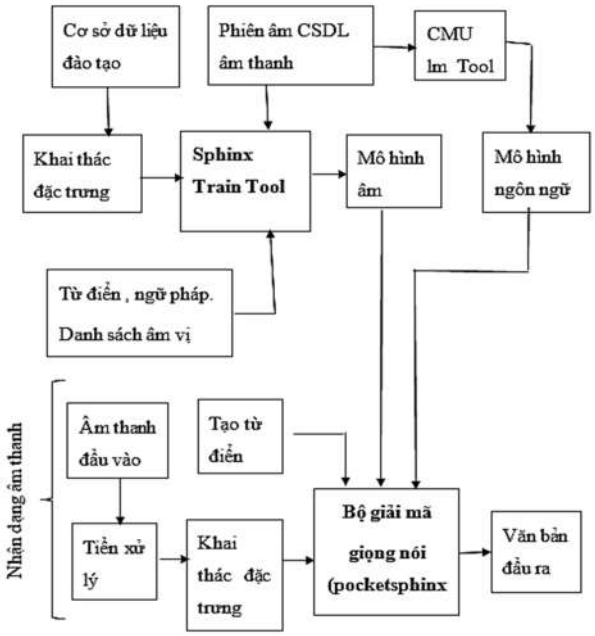
Đồ thị tìm kiếm (Search Graph): Là kết quả mà bộ ngôn ngữ phát sinh được cuối cùng để đưa vào sử dụng trong bộ giải mã. Đồ thị tìm kiếm này là một đồ thị có hướng, trong đó mỗi nút được gọi là một trạng thái tìm kiếm (SearchState), biểu diễn một trong hai trạng thái: phát hoặc không phát (emitting state hay non-emitting state). Và các đường cung biểu diễn các trạng thái biến đổi có thể, trên các cung này có các giá trị xác suất được tính toán từ mô hình âm học: biểu diễn khả năng chuyển từ trạng thái này đến trạng thái kia.



Hình 3. Ví dụ của đồ thị tìm kiếm

Bộ giải mã (Decoder): Sử dụng các đặc trưng (Features) từ bộ ngoại vi kết hợp với đồ thị tìm kiếm được phát sinh từ bộ ngôn ngữ để tiến hành giải mã và áp dụng các thuật toán suy ra kết quả nhận dạng. Nhiệm vụ của thành phần quản lý tìm kiếm là nhận dạng các tập các vector đặc trưng để tìm ra ánh xạ tương ứng của nó trong đồ thị tìm kiếm. Để đáp ứng tìm ra kết quả chính xác trong đồ thị tìm kiếm khi xử lý kết quả, Sphinx cung cấp các tiện ích có khả năng phát sinh lưới và các đánh giá độ tin cậy từ kết quả. Và thêm đặc điểm nữa khác các hệ thống khác là không gian tìm kiếm trong Sphinx có thể được tinh chỉnh thay đổi trong quá tình tìm kiếm để tăng hiệu suất tìm kiếm. Ngoài ra để nâng cao hiệu suất của kết quả nhận dạng, Sphinx còn bổ sung thêm các công cụ hỗ trợ cho việc đánh giá kết quả nhận được, đó là thành phần đánh giá (Scorer) và thành phần cắt tỉa (Pruner). Nói về thành phần Scorer thì nó là một module dùng để ước lượng xác suất của trạng thái khi cung cấp các giá trị mật độ trạng thái xuất hiện. Khi thành phần quản lý tìm kiếm yêu cầu đánh giá điểm số cho một trạng thái, nó sẽ gọi đến thành phần Scorer, nó sẽ phân tích các thông tin đặc trưng của trạng thái đó rồi áp dụng các phép toán để tính điểm số [5].

Nhận dạng giọng nói với Pocketsphinx: PocketSphinx [4] là một thư viện con của CMU Sphinx. PocketSphinx là trình nhận dạng nhẹ, tuy hơi không chính xác nhưng nó có thể giải mã các cụm từ nhanh hơn.



Hình 4. Sơ đồ khối nhận dạng giọng nói với pocketsphinx [4]

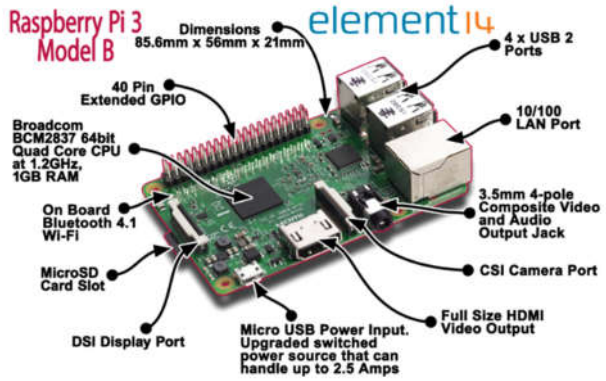
Khi các mô hình được phát triển từ cơ sở dữ liệu đào tạo, chúng có thể được sử dụng trong nhận dạng giọng nói. Khởi tạo mô hình thực hiện bằng cách sử dụng các công cụ huấn luyện sphinx. Nhận dạng được thực hiện bởi bộ giải mã giọng nói pocketsphinx. Cơ sở dữ liệu giọng nói, phiên âm, từ điển ngữ âm và danh sách âm vị với từng ngôn ngữ cụ thể được sử dụng để đào tạo hệ thống nhận diện giọng nói.

Các mô hình âm thanh thu được bằng cách sử dụng dữ liệu giọng nói với từ điển phiên âm và ngữ âm. Mô hình âm thanh và mô hình ngôn ngữ là rất quan trọng trong hệ thống. Tuy nhiên, các mô hình ngôn ngữ có thể thu được bằng cách sử dụng số lượng lớn dữ liệu sao chép. Tức là, dữ liệu giọng nói không bắt buộc đối với các mô hình ngôn ngữ. Các phần sau đây sẽ mô tả chi tiết về quy trình được thông qua để xây dựng cơ sở dữ liệu, mô hình âm thanh, mô hình ngôn ngữ và nhận dạng.

3. MÁY TÍNH NHÚNG RASBPERRY VÀ HỆ THỐNG NHẬN DẠNG GIỌNG NÓI ĐIỀU KHIỂN THIẾT BỊ

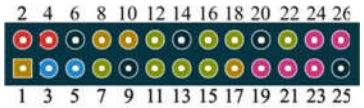
3.1. Máy tính nhúng Raspberry

Raspberry Pi là một máy tính nhúng kích cỡ nhỏ và chạy hệ điều hành Linux. Được phát triển bởi tổ chức phi lợi nhuận Raspberry Pi Foundation với tiêu chí xây dựng hệ thống mà nhiều người có thể sử dụng được trong những công việc tùy biến khác nhau. Đặc tính của Raspberry Pi xây dựng xoay quanh bộ xử lí SoC Broadcom BCM2835 (là chip xử lí mobile có kích thước nhỏ hay được dùng trong điện thoại di động) bao gồm CPU, GPU, bộ xử lý âm thanh video, và các tính năng khác. Tất cả được tích hợp bên trong con chip nảy.



Hình 5. Cấu hình máy tính nhúng Raspberry Pi 3

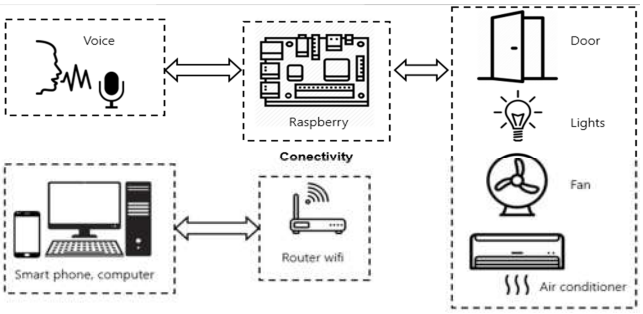
Raspberry Pi [5] không thể chạy hệ điều hành Windows vì bộ xử lí BCM2835 dựa trên cấu trúc ARM nên không hỗ trợ mã x86/x64, nhưng vẫn có thể chạy bằng Linux với các tiện ích như lướt web, môi trường desktop và các nhiệm vụ khác. Raspberry Pi hỗ trợ lập trình C/C++, Java, Python,… việc hỗ trợ nhiều ngôn ngữ lập trình nhằm tùy biến theo sở thích của người dùng. Khả năng lập trình ngay trên Raspberry Pi vẫn có thể được, tuy nhiên với một chiếc máy tính chuyên dụng chỉ xử lý một hay vài công việc thì việc lập trình trở nên khó khăn. Do đó bài báo đã dùng máy tính để lập trình cho việc nhận dạng tiếng nói thử nghiệm với độ ổn định cao. Bước tiếp theo đưa đoạn code vào hệ thống nhúng Raspberry Pi. Nhằm mục đích tối ưu trong quá trình xử lý tiếng nói. Máy tính nhúng này còn hỗ trợ điều khiển thiết bị ngoại vi qua hàng chân GPIO. Với tính năng này Raspberry Pi có thể được áp dụng vào nhiều công trình nghiên cứu và ứng dụng như: điều khiển robot, điều khiển thiết bị điện trong nhà,…



Hình 6. Sơ đồ chân GPIO

Trong sơ đồ chân GPIO có các chân được đánh thứ tự hàng trên là số chẵn hàng dưới là số lẻ. Tính từ trái qua phải khi đặt Raspberry Pi, hàng chân GPIO sẽ nằm cạnh bên cổng tín hiệu video trên bo Raspberry Pi. Trong đó: 1, 17: chân nguồn 3.3v, 2, 4: chân nguồn 5v, 6, 9, 14, 20, 25: chân nối đất Ground 0v, 8, 10: GPIO truyền nhận tín hiệu theo chuẩn UART, 7, 11, 12, 13, 15, 16, 18 và 22: chân GPIO, 19, 21, 23, 24, 26: GPIO giao tiếp chuẩn SPI, 3, 5: GPIO giao tiếp chuẩn I2C, 12: GPIO điều khiển PWM, tăng giảm cường độ [5].

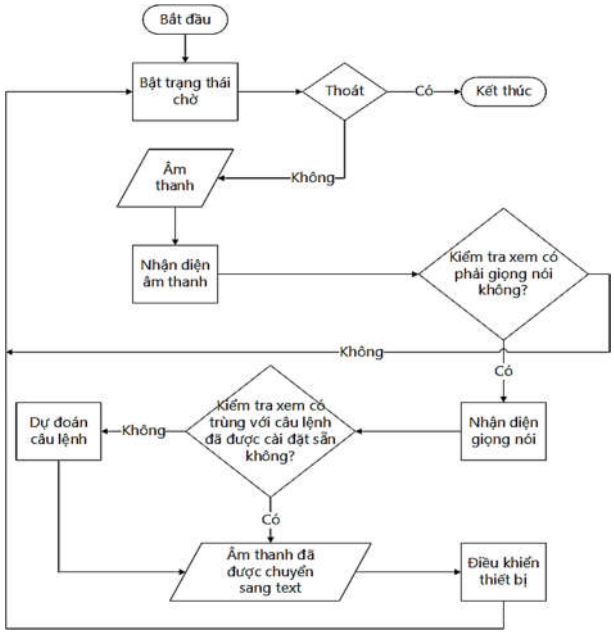
3.2. Sơ đồ khối hệ thống nhận dạng giọng nói và điều khiển thiết bị



Hình 7. Sơ đồ khối toàn hệ thống

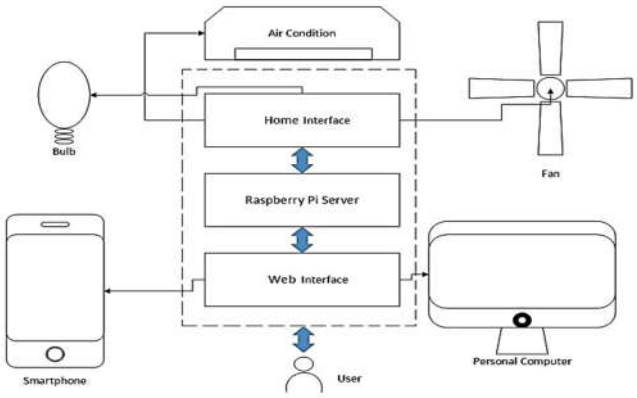
Tín hiệu tiếng nói người dùng được lấy bởi micro USB đã được kết nối với hệ thống. Chương trình điều khiển được viết bằng ngôn ngữ Python và được biên dịch trên máy tính nhúng Raspberry Pi để điều khiển các thiết bị trực tiếp qua hàng chân GPIO (General Purpose Input/ Output) trên Raspberry Pi. Có hai cách để điều khiển thiết bị: điều khiển bằng giọng nói, điều khiển bảng web server.

Lưu đồ thuật toán của hệ thống như hình 8.



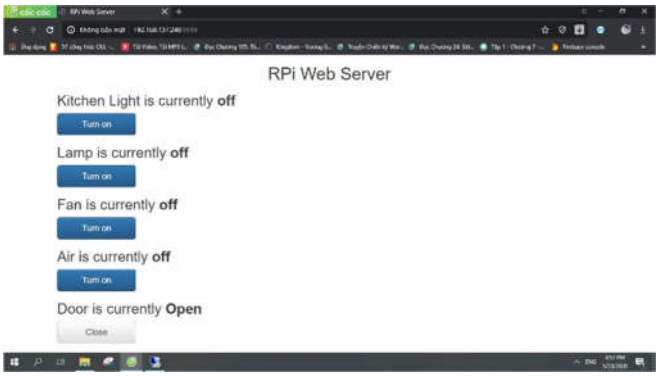
Hình 8. Lưu đồ thuật toán của hệ thống

Khi được khởi động hệ thống sẽ tiến hành kiểm tra mic. Nếu mic không có vấn đề gì thì sẽ chuyển sang trạng thái waiting và có đèn báo hiệu là hệ thống đã sẵn sàng tiếp nhận giọng nói. Âm thanh được truyền vào mic, hệ thống sẽ chuyển sang trạng thái nhận diện âm thanh. Nếu âm thanh là giọng nói hệ thống sẽ chuyển sang quá trình nhận diện giọng nói. Giọng nói được xử lý và chuyển sang dạng text. Sau khi chuyển sang text, giọng nói sẽ được đối chiếu với command đã được định nghĩa sẵn. Nếu đã đúng câu lệnh hệ thống sẽ thực thi và quay trở lại trạng thái waiting.



Hình 9. Cấu trúc điều khiển qua web server

Bộ thư viện để kiểm tra trạng thái thiết bị theo thời gian thực là Flask. Điều khiển bằng giọng nói có thể thực hiện khi đang ngoại tuyến tức là không cần mạng, thì khi điều khiển thông qua server thì chúng ta cần kết nối điện thoại hoặc máy tính và raspberry chung một mạng.



Hình 10. Giao diện web server

4. KẾT QUẢ, KẾT LUẬN VÀ ĐÁNH GIÁ

4.1. Kết quả kiểm thử

Kiểm thử 10 lần cho kết quả như bảng 1.

Bảng 1. Kết quả kiểm thử

Lần thử	Điều khiển bằng giọng nói		Điều khiển bằng web server	
	Nhận dạng	Điều khiển	Hiển thị web	Điều khiển
1	YES	YES	YES	YES
2	YES	YES	YES	YES
3	YES	YES	YES	YES
4	NO	NO	YES	YES
5	YES	YES	YES	YES
6	YES	YES	YES	YES
7	YES	YES	YES	YES
8	YES	YES	YES	YES
9	NO	NO	YES	YES
10	YES	YES	YES	YES
Tỷ lệ	80%	80%	100%	100%

Kết quả nhận dạng khoảng 80% để hệ thống nhận dạng và điều khiển được bằng giọng nói. Nguyên nhân là chủ yếu do khả năng phát âm tiếng Anh, môi trường và micro chưa được tốt.

Khi kiểm thử với số lượng lớn hơn với khoảng 200 lần , kết quả nhận đúng xấp xỉ khoảng 88%. Đây là một kết quả khá khả quan. (Thực nghiệm trong môi trường không nhiễu)

4.2. Đánh giá, ưu nhược điểm

Ưu điểm

- Thiết bị nhỏ gọn, linh động, hiểu quả.

- Đã nhận diện được giọng nói mà không phải thông bất cứ API trung gian nào. Quá trình sử dụng không cần thiết phải sử dụng kết nối internet.

- Ngoài bật tắt thì còn có thể điều khiển các thiết bị theo mức bằng giọng nói như độ sáng đèn, tốc độ quạt, các chế độ điệu điều hòa.

- Độ trễ khi thực hiện điều khiển thiết bị rất nhỏ khi điều khiển bằng giọng nói lẫn web server.

Nhược điểm

- Chưa xử lý triệt để được vấn đề tạp âm gây khó khăn trong quá trình nhận diện để điều khiển.

- Nếu liên tục bị nhiễu âm, micro phải reset liên tục gây hiện tượng treo micro.

5. KẾT LUẬN

Xây dựng thành công hệ thống nhận dạng và điều khiển thiết bị trên hệ thống máy tính Raspberry Pi. Bài báo đã đạt được một số kết quả như: ứng dụng và áp dụng được công nghệ nhận dạng tiếng nói thông qua bộ thư viện nhận dạng CMU Sphinx xây dựng hệ thống nhận dạng tiếng nói trên Raspberry Pi và điều khiển thiết bị bằng giọng nói. Lập trình bằng ngôn ngữ Python kết hợp thêm Flask để nhận được trạng thái thiết bị và điều khiển trên web server, thực thi trên hệ thống nhúng Raspberry Pi, nhỏ gọn nhưng vẫn linh động và hiệu quả về tốc độ xử lý và nâng cấp về sau. Vì hệ thống còn gặp rất nhiều hạn chế khi nhận diện giọng nói ở môi trường có nhiễu nên cần nghiên cứu và ứng dụng các phương pháp lọc nhiễu âm thanh và xử lý khi gặp âm thanh lỗi.

TÀI LIỆU THAM KHẢO

[1]. Mai Ngọc Chừ, Vũ Đức Nghiệu, Hoàng Trọng Phiến, 2000. *Cơ sở ngôn ngữ học và tiếng Việt*. Nhà xuất bản Giáo dục.

[2]. H. Kang, 2003. *Speech Signal Processing*. Yonsei University.

[3]. Z. Ling, 2019. *An Acoustic Model for English Speech Recognition Based on Deep Learning*. International Conference on Measuring Technology and Mechatronics Automation.

[4]. https://cmusphinx.github.io/wiki/tutorial/

[5]. Imteaj Ahmed, Tanveer Rahman, Muhammad Kamrul Hossain, Saika Zaman, 2016. *IoT based autonomous percipient irrigation system using raspberry Pi.* In Computer and Information Technology.

[6]. Jurafsky Daniel, Martin James H, 1996. *Speech and Language Processing: An Introduction to Natural Language Processing.* Computational Linguistics, and Speech Recognition.

[7]. Preeti Saini, Parneet Kaur, 2013. *Automatic Speech Recognition: A Review.* International Journal of Engineering Trends and Technology.

[8]. Lakkhanawannakun, P. Noyunsan, 2019. *Speech Recognition using Deep Learning.* International Technical Conference on Circuits/Systems.