ĐIỀU KHIỂN CON TRỎ CHUỘT BẰNG CỬ CHỈ BÀN TAY DỰA TRÊN TRÍ TUỆ NHÂN TẠO

Nguyễn Đức Nhật Quang[1*], Phan Thị Huỳnh Ngân[1], Phan Văn Cường[1], Trần Thị Thu Huyền[2]

[1] Khoa Điện, Điện tử và Công nghệ vật liệu, Trường Đại học Khoa học, Đại học Huế

[2] Khoa Công nghệ thông tin, Trường Đại học Khoa học, Đại học Huế

*Email: ndnquang@hueuni.edu.vn

Ngày nhận bài: 30/9/2023; ngày hoàn thành phản biện: 8/10/2023; ngày duyệt đăng: 4/12/2023

TÓM TẮT

Nghiên cứu này trình bày về việc sử dụng cử chỉ tay để điều khiển con trỏ chuột, thay thế các phương pháp truyền thống như chuột máy tính, touchpad và màn hình cảm ứng. Nhóm tác giả sử dụng thư viện MediaPipe để nhận diện và theo dõi cử chỉ tay thông qua webcam kết hợp với thư viện PyAutoGUI trong Python để điều khiển con trỏ chuột. Ngoài ra, trợ lý giọng nói được sử dụng để tương tác với chương trình nhận diện cử chỉ và thực hiện điều khiển máy tính. Kết quả cho thấy độ chính xác của hệ thống đạt trên 95% trong điều kiện ánh sáng tốt, nền đơn giản và khoảng cách gần, và có độ chính xác cao hơn so với các phương pháp truyền thống. Tóm lại, hệ thống chuột ảo này mang lại tiện ích và cải thiện trải nghiệm tương tác với máy tính.

Từ khóa: virtual mouse, hand recognition, hand detection, mediapipe.

1. MỞ ĐẦU

Các phương pháp điều khiển con trỏ chuột hiện nay bao gồm chuột máy tính truyền thống, bàn di chuột (touchpad) và màn hình cảm ứng. Mỗi phương pháp này đều có nhược điểm riêng như hạn chế về không gian, độ chính xác, tính di động và tương tác không như ý muốn của người dùng. Việc sử dụng cử chỉ tay để điều khiển con trỏ chuột giúp triển khai ứng dụng nhanh chóng, dễ dàng hơn và cũng khắc phục nhược điểm của các phương pháp đang được sử dụng hiện nay.

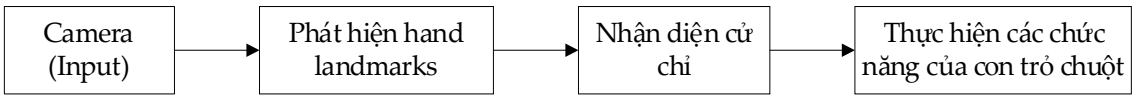
Bên cạnh đó, trí tuệ nhân tạo (Artificial Intelligence – AI) và thị giác máy tính (Computer Vision – CV) đang phát triển mạnh mẽ, đặc biệt trong lĩnh vực nhận diện khuôn mặt, đối tượng, chữ viết tay và cử chỉ tay. Công nghệ này cải thiện tương tác giữa người và máy (Human-Computer Interaction – HCI), đặc biệt là điều khiển chuột ảo. Có

nhiều phương pháp nhận diện cử chỉ tay, từ cảm biến chuyển động đến xử lý hình ảnh. Tuy nhiên, chúng thường gặp các vấn đề như tốc độ xử lý chậm và độ chính xác thấp. MediaPipe là thư viện mã nguồn mở cho thị giác máy tính của Google, chủ yếu về nhận diện cử chỉ tay. Thư viện này sử dụng mô hình học sâu để theo dõi bộ phận tay trong không gian 3D, đơn giản hóa việc phát triển ứng dụng, thuật toán và mô hình.

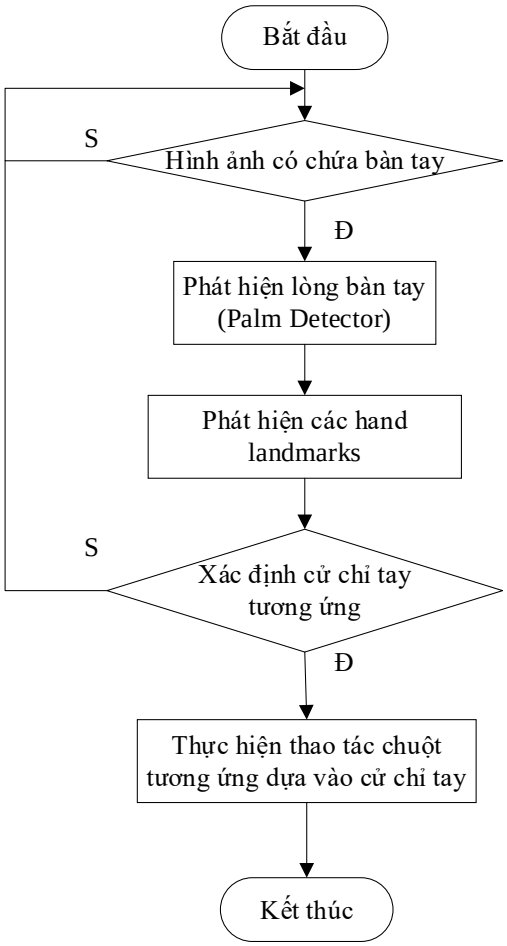
Sử dụng cử chỉ tay để điều khiển con trỏ chuột mang lại nhiều lợi ích như tính di động, tiết kiệm thời gian, hỗ trợ người khuyết tật và cải thiện trải nghiệm người dùng. Nghiên cứu này sử dụng webcam và giao diện lập trình ứng dụng (Application Programming Interface - API) của MediaPipe để nhận diện và theo dõi tay, loại bỏ sự cần thiết phải sử dụng các thiết bị phụ trợ. Để điều khiển con trỏ chuột, nhóm tác giả sử dụng thư viện PyAutoGUI trong Python. PyAutoGUI cung cấp các hàm để điều khiển chuột, bàn phím và ghi lại thao tác chuột cũng như bàn phím. Ngoài ra, nghiên cứu này còn sử dụng trợ lý giọng nói để tương tác với chương trình nhận diện cử chỉ và thực hiện điều khiển máy tính.

2. PHƯƠNG PHÁP NGHIÊN CỨU

Quá trình hoạt động của hệ thống điều khiển con trỏ chuột bằng cử chỉ bàn tay dựa trên AI được chia thành ba giai đoạn, được thể hiện trong ***Hình 1***. Giai đoạn đầu tiên, hệ thống phát hiện và theo dõi các điểm đặc trưng (Landmark), sau đó xác định vị trí và tọa độ của các điểm đặc trưng và theo dõi chuyển động theo thời gian. Giai đoạn tiếp theo, hệ thống nhận diện, chuyển đổi các đặc điểm điểm đặc trưng thành các cử chỉ tay có thể nhận diện được dưới dạng các số nhị phân. Giai đoạn cuối cùng, hệ thống tùy chỉnh và thực hiện các chức năng dựa trên các cử chỉ tay đã nhận diện.



Hình 1. Sơ đồ khối quá trình hoạt động của hệ thống.



Hình 2. Quá trình nhận diện và điều khiển cử chỉ tay.

Quá trình nhận diện và điều khiển cử chỉ tay (***Hình 2***) bao gồm: thu thập dữ liệu về tay, phát hiện lòng bàn tay và xác định điểm đặc trưng. Dựa trên vị trí của các điểm đặc trưng, hệ thống nhận diện cử chỉ tay và thực hiện các thao tác chuột tương ứng. Sau đó, quá trình kết thúc và hệ thống sẵn sàng cho yêu cầu tiếp theo.

2.1. Thư viện MediaPipe Hands

Thư viện MediaPipe Hands cung cấp theo dõi bàn tay thời gian thực chỉ với một webcam thông thường. Quy trình này sử dụng học máy để suy luận 21 điểm đặc trưng 3D từ một hình ảnh duy nhất. Quy trình bao gồm hai bước chính: mô hình phát hiện lòng bàn tay và mô hình điểm đặc trưng tay có khả năng theo dõi nhiều bàn tay cùng lúc [1].

2.1.1. Mô hình phát hiện bàn tay

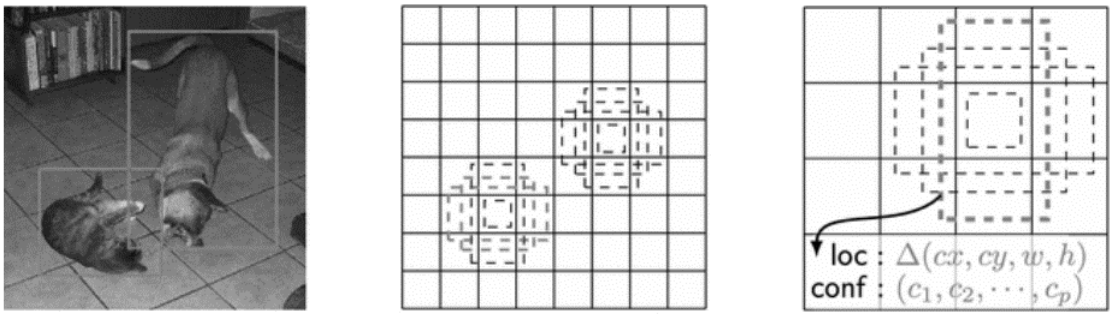
Phát hiện bàn tay là nhiệm vụ phức tạp, đặc biệt khi xem xét nhiều kích cỡ tay và khả năng che khuất. Mô hình phát hiện lòng bàn tay ước tính hộp giới hạn xung quanh các đối tượng cứng như lòng bàn tay và nắm. Quy trình này áp dụng tính năng

mã hóa giải mã và giảm mất mát tập trung để cải thiện nhận biết bối cảnh quang cảnh lớn. Module nhận diện đối tượng lấy mẫu đặc trưng cho mỗi hộp và sử dụng bộ phân loại chất lượng cao. Có nhiều thuật toán nhận diện đối tượng như Faster RCNN, SSD và YOLO, mỗi thuật toán đánh đổi giữa tốc độ và độ chính xác tùy theo yêu cầu [2].

2.1.2. Bộ phát hiện chụp một lần

Quy trình cho bộ phát hiện chụp một lần (Single Shot Detector – SSD) bao gồm tạo ra các thông tin cơ bản (Ground Truth – GT) là các hộp giới hạn xung quanh đối tượng trong hình ảnh. Hình ảnh được chia thành các bản đồ đặc trưng với các kích thước khác nhau. Các hộp mặc định (default boxes) được đánh giá tại mỗi ô vuông trên bản đồ đặc trưng. Các hộp mặc định chứa thông tin về vị trí (center_x, center_y, width, height) và độ tin cậy cho từng loại đối tượng. Mục tiêu là dự đoán các giá trị thay đổi hình dạng (shape offsets) và độ tin cậy cho mỗi hộp mặc định, sao cho chúng khớp với các hộp GT trong quá trình huấn luyện.

Các hộp khớp với hộp mặc định được xem là tích cực (positive), còn lại là tiêu cực (negative). Mô hình SSD tính mất mát dựa trên tổng có trọng số giữa mất mát về vị trí (localization loss – Smooth L1) và mất mát về độ tin cậy (ví dụ: softmax). ***Hình 3*** minh họa ví dụ về quá trình này trên một hình ảnh với mèo và chó, sử dụng các bản đồ đặc trưng kích thước 8x8 và 4x4, cùng với các hộp mặc định và hộp thực tế. Trong đó, loc là vị trí của hộp giới hạn, conf là độ tin cậy của tất cả các loại đối tượng.

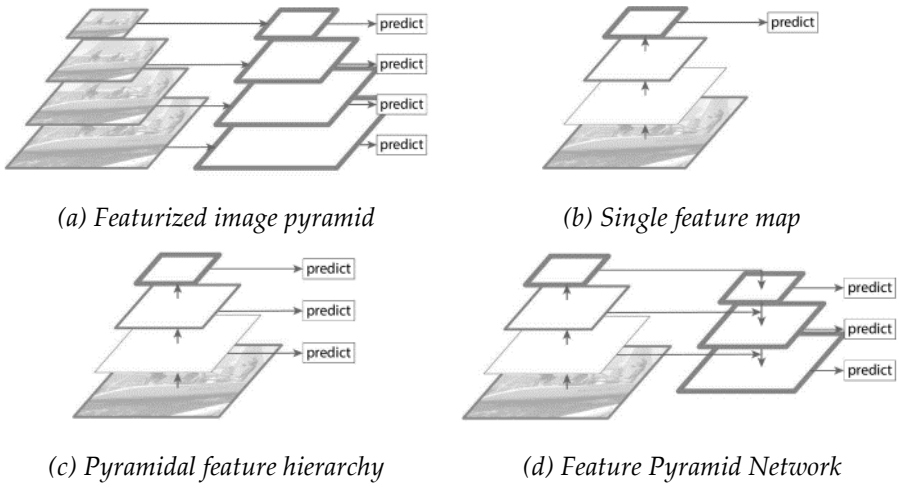


(a) Hình ảnh với các hộp GT *(b) Bản đồ đặc trưng 8x8* *(c) Bản đồ đặc trưng 4x4*

Hình 3. Single Shot Detector framework [2].

2.1.2.1. Mạng kim tự tháp đặc trưng

Mạng kim tự tháp đặc trưng (Feature Pyramid Network – FPN) được sử dụng như một "cổ" (neck) nối vào "xương sống" (backbone) của SSD để tính toán biểu diễn đa đặc trưng của hình ảnh. Cấu trúc kim tự tháp đặc trưng xây dựng trên cơ sở kim tự tháp hình ảnh và là không gian tỉ lệ, giúp bắt kích thước đa dạng của các đối tượng. So với các kiến trúc khác, FPN kết hợp các đặc trưng từ độ phân giải cao nhất đến thấp nhất, làm cho nó chính xác hơn và có nhiều thông tin hơn.



(a) Featurized image pyramid *(b) Single feature map*

(c) Pyramidal feature hierarchy *(d) Feature Pyramid Network*

Hình 4. Feature Pyramid Network (FPN) [3].

2.1.2.2. Giảm thiểu mất tiêu điểm

Hàm mất mát Focal (Focal Loss) giải quyết vấn đề khi mô hình phát hiện đối tượng bị mất cân bằng giữa việc nhìn nhận những vật thể chính (foreground) và nền. Với cross-entropy loss, ngay cả các vật thể dễ nhìn nhận cũng gây mất mát lớn hơn mức dự kiến. Và khi tổng các mất mát nhỏ này trên một số lượng lớn các vật thể dễ, thì tổng mất mát này có thể áp đảo lên các vật thể khó hơn. Cross-entropy loss bị chi phối bởi sự mất cân bằng lớn về lớp khi gặp các mô hình đối tượng dày đặc. Phần lớn mất mát tạo nên từ các vật thể dễ nhận diện, làm áp đảo lên độ dốc. "Balanced cross-entropy" giải quyết vấn đề mất cân bằng lớp bằng cách sử dụng hệ số trọng số alpha. Tuy nhiên, nó không phân biệt giữa các vật thể dễ và khó.

$$CE(p_t) = -\alpha \log(p_t) \quad (1)$$

Mất tiêu điểm (Focal Loss) thay đổi hình dạng của hàm mất mát để giảm trọng số của các ví dụ dễ dàng và tập trung vào việc huấn luyện các ví dụ âm khó khăn. Công thức cho Focal Loss (FL) được cho trong (2), trong đó y là tham số làm mịn [4].

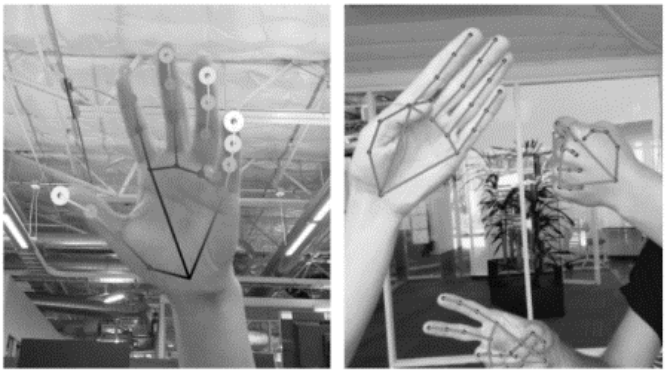
$$FL(p_t) = -(1 - p_t)^y \log(p_t) \quad (2)$$

Công thức (2) giải quyết vấn đề mất cân bằng giữa các mặt tích cực và tiêu cực trong mất mát entropy chéo. Các ví dụ dễ có xác suất cao ($p_t$tiến gần đến 1) và được giảm trọng số để không ảnh hưởng quá mạnh đến mất mát. Tham số tập trung y được sử dụng để điều chỉnh mượt mà tốc độ giảm trọng số đối với các ví dụ dễ. Trong thực tế, biến thể alpha ở công thức (3) của Focal Loss được ưa chuộng để đạt độ chính xác tốt hơn.

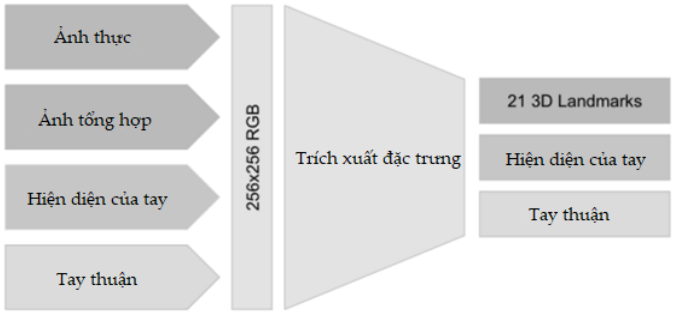
$$FL(p_t) = -\alpha(1 - p_t)^y \log(p_t) \quad (3)$$

2.1.3. Mô hình điểm đặc trưng tay

Mô hình điểm đặc trưng tay (Hand Landmark Model) sử dụng phương pháp hồi quy để xác định vị trí chính xác của 21 tọa độ điểm đặc trưng 2.5D trong lòng bàn tay. Ngay cả khi tay bị che khuất hoặc chỉ một phần hiển thị, mô hình tạo biểu diễn liên tục về dạng tay trong không gian 2.5D. Đầu ra của mô hình bao gồm các tọa độ 3D của 21 điểm đặc trưng, xác suất hiện diện của tay và thông tin về tay trái/phải. Một kiến trúc dựa trên khởi động đa chế độ xem (Multiview Bootstrapping) được áp dụng để cải thiện hiệu suất phát hiện điểm đặc trưng, bao gồm cả việc ước tính điểm đặc trưng khi tay bị che khuất. Đồng thời, một mô hình được tạo ra để điều chỉnh căn chỉnh tay một cách chính xác dựa trên thông tin ảnh đầu vào.



Hình 5. Ví dụ về ước tính tư thế bàn tay bằng MediaPipe [5].



Hình 6. Kiến trúc mô hình điểm đặc trưng cho MediaPipe [5].

2.1.3.1. Khởi động đa chế độ xem (Multiview Bootstrapping)

Kỹ thuật Multiview Bootstrapping giúp tạo ra một bộ phát hiện tay hiệu quả có khả năng xác định vị trí các điểm chính trong các góc nhìn tốt và loại bỏ các phát hiện sai. Điều này đặc biệt hữu ích khi bàn tay bị che khuất trong một bức ảnh. Bộ phát hiện này được huấn luyện trên một tập dữ liệu nhỏ và có khả năng tổng quát hóa vượt ra ngoài các tình huống chụp cụ thể.

2.1.3.2. Khôi phục lỗi theo dõi (Recover Tracking Failure)

MediaPipe dùng bộ phát hiện tay nhẹ để tạo hộp giới hạn và điểm đặc trưng cho bàn tay, cổ tay, ngón tay. Điểm đặc trưng giúp xoay hộp giới hạn của bàn tay, đảm bảo trung tâm cổ tay căn chỉnh với trục ngang. Ảnh gốc sau đó được cắt và điều chỉnh kích thước để làm đầu vào cho mạng dự đoán lưới khuôn mặt. Mô hình dự đoán vector tọa độ điểm đặc trưng 3D và xác suất có bàn tay trong phần cắt.

2.2. Nhận diện cử chỉ

Giai đoạn này xác định và chuyển đổi các điểm đặc trưng từ MediaPipe thành các cử chỉ tay có thể nhận biết. Các khoảng cách và sự khác biệt giữa các điểm đặc trưng được tính toán để xác định và ánh xạ thành các cử chỉ tay tương ứng. Các cử chỉ được biểu diễn dưới dạng số nhị phân và được quy định bằng lớp IntEnum, với các giá trị tương ứng cho mỗi cử chỉ.

Bảng 1. Ánh xạ giá trị nhị phân cho các cử chỉ bằng lớp IntEnum.

FIST	PINKY	RING	MID	LAST3	INDEX	FIRST2	LAST4	THUMB
0	1	2	4	7	8	12	15	16

PALM	V_GEST	TWO_FINGER_CLOSED	PINCH_MAJOR	PINCH_MINOR
31	33	34	35	36

Giai đoạn này mã hóa thông tin về sự thuận tay (handedness) khi sử dụng nhiều tay và cung cấp các phương thức để cập nhật dữ liệu điểm đặc trưng, tính toán khoảng cách giữa các điểm đánh dấu và xác định cử chỉ tương ứng.

2.3. Thực hiện các chức năng của con trỏ chuột

Lớp này đóng vai trò quan trọng trong việc thực thi các lệnh tương ứng với cử chỉ tay đã được nhận diện. Nó cho phép tương tác với màn hình, di chuyển con trỏ chuột, nhấp chuột, kéo thả và cuộn trang dựa trên thông tin vị trí và các cử chỉ đã được phân tích. Lớp này lưu trữ thông tin về vị trí chuột trước đó, các cờ xác định việc nhận diện cử chỉ, cũng như khoảng cách và hướng di chuyển của cử chỉ PINCH, quan trọng để xử lý chính xác các hành động điều khiển hệ thống.

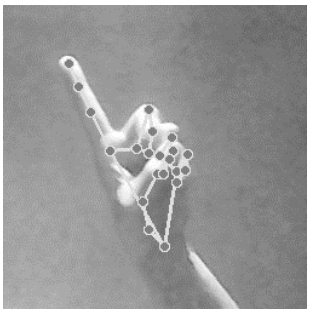
2.4. Trợ lý giọng nói

Ứng dụng kết hợp giọng nói và cử chỉ tay để tương tác với máy tính, sử dụng thư viện như *pyttsx3* và *speech_recognition*. Có thể chào hỏi người dùng, cung cấp thông tin, tìm kiếm trên web và bản đồ, sao chép, dán văn bản và điều hướng tệp tin. Sử dụng module nhận diện cử chỉ tay để mở ứng dụng và dừng nhận diện cử chỉ.

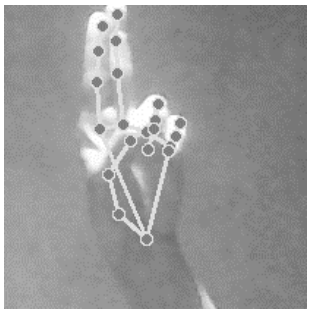
3. KẾT QUẢ VÀ THẢO LUẬN

3.1. Nhận diện cử chỉ

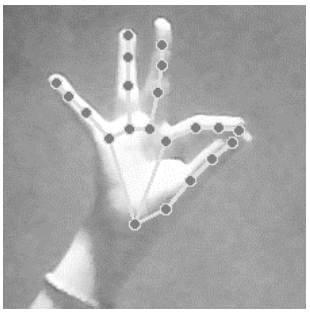
Dưới đây là một số cử chỉ và mô tả của chúng, cung cấp khả năng dừng, di chuyển con trỏ chuột, nhấp chuột trái/phải, nhấp đúp chuột, cuộn và kéo và thả.



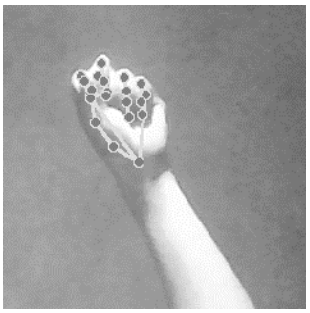
(d) Nhấp chuột phải (INDEX)



(e) Nhấp đúp chuột TWO_FINGER_CLOSED



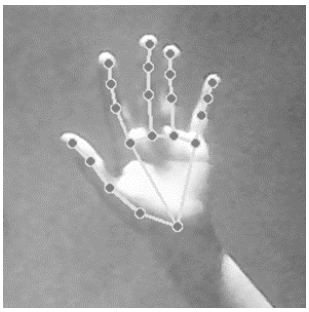
(f) Cuộn (PINCH_MINOR)



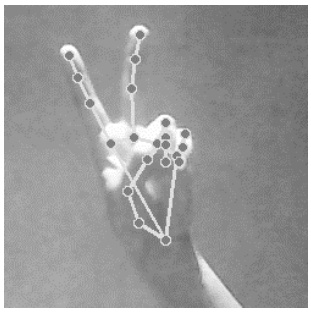
(g) Kéo và thả (FIST)

Hình 7 minh họa các cử chỉ tương ứng bằng hình ảnh:

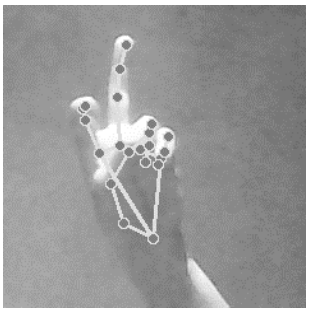
- Cử chỉ trung lập (PALM): được sử dụng để dừng cử chỉ hiện tại.
- Di chuyển con trỏ chuột (V_GEST): Con trỏ được gán cho điểm giữa của ngón trỏ và ngón giữa. Cử chỉ này di chuyển con trỏ đến vị trí mong muốn. Tốc độ di chuyển của con trỏ tỷ lệ thuận với tốc độ di chuyển của tay.
- Nhấp chuột trái (MID).
- Nhấp chuột phải (INDEX).
- Nhấp đúp chuột (TWO_FINGER_CLOSED).
- Cuộn (PINCH_MINOR): Cử chỉ động để cuộn ngang và dọc. Tốc độ cuộn tỷ lệ thuận với khoảng cách di chuyển bằng cử chỉ pinch từ điểm bắt đầu. Cuộn dọc và ngang được điều khiển bằng các chuyển động của cử chỉ pinch theo hướng dọc và ngang tương ứng.
- Kéo và thả (FIST): có thể được sử dụng để di chuyển tập tin từ thư mục này sang thư mục khác hoặc chọn nhiều mục cùng lúc.



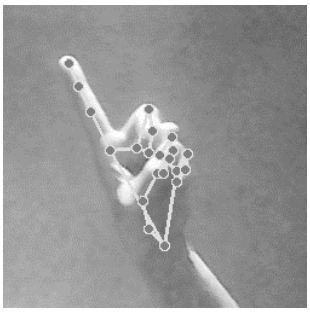
(a) Cử chỉ trung lập (PALM)



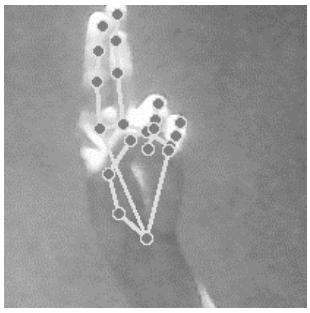
(b) Di chuyển con trỏ chuột (V_GEST)



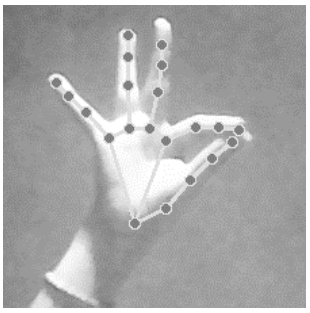
(c) Nhấp chuột trái (MID)



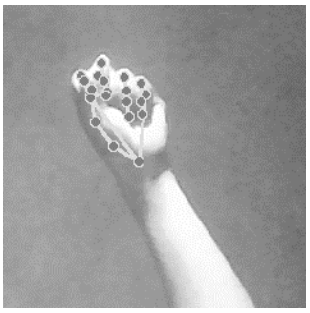
(d) Nhấp chuột phải (INDEX)



(e) Nhấp đúp chuột TWO_FINGER_CLOSED



(f) Cuộn (PINCH_MINOR)



(g) Kéo và thả (FIST)

Hình 7. Hình ảnh minh họa các cử chỉ và mô tả đi kèm.

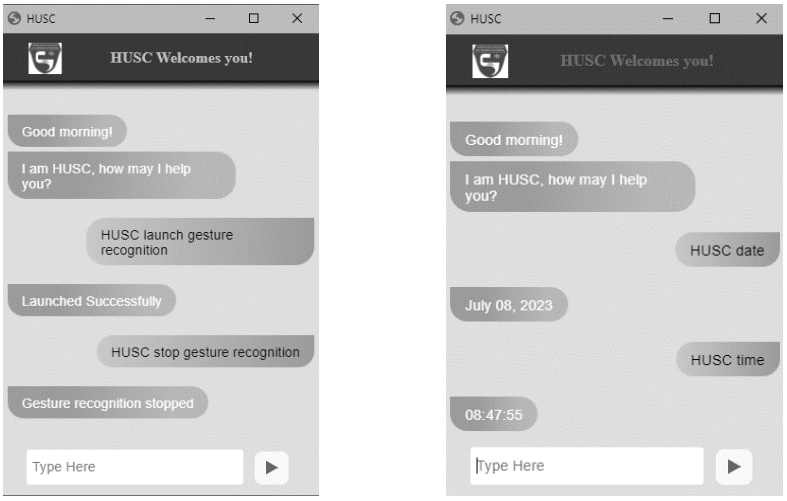
3.2. Trợ lý giọng nói

Dưới đây là một số lệnh và chức năng mà ta có thể sử dụng với trợ lý giọng nói. ***Hình 8*** mô tả 2 trong số những chức năng này:

- HUSC Launch Gesture Recognition: Bật webcam để nhận diện cử chỉ tay.
- HUSC Stop Gesture Recognition: Tắt webcam và dừng nhận diện cử chỉ.
- HUSC Search <nội dung muốn tìm kiếm> : Mở một tab mới trên trình duyệt mặc định nếu nó đang chạy, nếu không nó sẽ mở một cửa sổ mới. Tìm kiếm văn bản đã cho trên trình duyệt.
- HUSC Find Location: Ứng dụng sẽ hỏi người dùng vị trí cần tìm. Sau đó nhập địa điểm muốn tìm kiếm, ứng dụng sẽ tìm thấy vị trí cần thiết trên Google Maps trong tab mới.
- Điều hướng file:
    - HUSC list files/HUSC list: Liệt kê các tệp và số thứ tự tương ứng trong thư mục hiện tại (theo mặc định là D:\).
    - HUSC open <số thứ tự file>: Mở tệp/thư mục tương ứng với số thứ tự.

- HUSC go back/HUSC back: Thay đổi thư mục hiện tại thành thư mục mẹ và liệt kê các file.

- HUSC date/HUSC time: trả về ngày/giờ hiện tại.
- HUSC bye: Tạm dừng thực thi lệnh thoại cho đến khi trợ lý được đánh thức.
- HUSC wake up: Tiếp tục thực thi lệnh thoại.
- HUSC exit: Chấm dứt chuỗi trợ lý giọng nói. Cửa sổ GUI cần đóng thủ công.



(a) Chạy và dừng chương trình *(b) Trả về ngày/giờ hiện tại*

Hình 8. Hình ảnh minh họa hai chức năng của trợ lý giọng nói.

Bảng 2 cho thấy độ chính xác của điểm đặc trưng dưới các điều kiện khác nhau. Mô hình đạt độ chính xác trên 95% trong điều kiện ánh sáng tốt, nền đơn giản và khoảng cách gần. Tuy nhiên, độ chính xác giảm trong điều kiện ánh sáng yếu, nền phức tạp và khoảng cách tay xa. Để đạt độ chính xác tốt, cần tạo môi trường thuận lợi với ánh sáng đủ mạnh, nền đơn giản và khoảng cách tay hợp lý.

Bảng 2. Độ chính xác của các điểm đặc trưng trong các điều kiện khác nhau.

Điều kiện	Độ chính xác (%)
Ánh sáng tốt	98
Ánh sáng thấp	92
Nền đơn giản	98
Nền phức tạp	92
Khoảng cách gần	99
Khoảng cách xa	95

Bảng *3* so sánh độ chính xác của mô hình sử dụng MediaPipe với các phương pháp truyền thống. Kết quả cho thấy mô hình này có độ chính xác cao hơn, nhờ việc sử dụng điểm đặc trưng và mạng nơ-ron trong MediaPipe để học các đặc trưng phức tạp của tay. Tuy nhiên, độ chính xác và tốc độ xử lý vẫn phụ thuộc vào môi trường và yếu tố khác nhau như ánh sáng, nền và kích thước hình ảnh. Để tăng hiệu suất, cần tạo môi trường thuận lợi và áp dụng các kỹ thuật tối ưu hóa như giảm độ phân giải hình ảnh hoặc cải thiện phần cứng và thuật toán.

Bảng 3. So sánh với các hệ thống hiện có.

Các phương pháp đã thực hiện	Độ chính xác (%)
Đầu ngón tay có gắn màu [6]	78
Phát hiện lòng bàn tay và trích xuất tâm [7]	93
Nhận diện màu da, phát hiện đường biên, tạo lồi convex [8]	90
Nhận diện và theo dõi tay MediaPipe	96

4. KẾT LUẬN

Hệ thống nhằm thay thế chuột máy tính bằng cử chỉ tay, sử dụng webcam để nhận diện cử chỉ và thực hiện chức năng tương tự như chuột. Độ chính xác của hệ thống vượt trội, đạt tỷ lệ chính xác lên đến 96%, mở ra nhiều ứng dụng tiềm năng. Mặc dù tốc độ xử lý còn chậm và còn hạn chế, nhóm tác giả đề xuất cải thiện thuật toán để giải quyết vấn đề này và mở rộng tính năng ứng dụng. Tóm lại, chuột ảo tối ưu hóa trải nghiệm tương tác với máy tính, mang lại tiện ích và sự thoải mái cho người dùng.

LỜI CẢM ƠN

Nghiên cứu này được thực hiện trong khuôn khổ đề tài nghiên cứu khoa học sinh viên cấp cở sở ĐHKH2023B-03.

TÀI LIỆU THAM KHẢO

[1] Google LLC. Hands. https://google.github.io/mediapipe/solutions/hands.html. accessed: 12/05/2023.

[2] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, ChengYang Fu, and Alexander Berg (2016). Ssd: Single shot multibox detector, *SSD: Single Shot MultiBox Detector*, volume 9905, pages 21–37.

[3] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie (2017). Feature Pyramid Networks for Object Detection, *Feature Pyramid Networks for Object Detection*, pages 936–944.

[4] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar (2017). Focal Loss for dense object detection, *Focal Loss for Dense Object Detection*, pages 2999–3007.

[5] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann (2020). Mediapipe hands: On-device real-time hand tracking.

[6] K. H. Shibly, S. Kumar Dey, M. A. Islam and S. Iftekhar Showrav (2019). Design and Development of Hand Gesture Based Virtual Mouse, *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, Dhaka, Bangladesh, 2019, pp. 1-5.

[7] S. Hussain, R. Saxena, X. Han, J. A. Khan and H. Shin (2017). Hand gesture recognition using deep learning, *2017 International SoC Design Conference (ISOCC), Seoul, Korea (South),* pp. 48-49.

[8] V. V. Reddy, T. Dhyanchand, G. V. Krishna and S. Maheshwaram (2020). Virtual Mouse Control Using Colored Finger Tips and Hand Gesture Recognition, *2020 IEEE-HYDCON*, Hyderabad, India, 2020, pp. 1-5.

MOUSE CURSOR CONTROL USING HAND GESTURES BASED ON ARTIFICIAL INTELLIGENCE (AI)

Nguyen Duc Nhat Quang[1*], Phan Thi Huynh Ngan[1], Phan Van Cuong[1], Tran Thi Thu Huyen[2]

[1] Faculty of Electronics, Electrical Engineering and Material Technology, University of Sciences, Hue University

[2]Faculty of Information Technology, University of Sciences, Hue University

*Email: ndnquang@hueuni.edu.vn

ABSTRACT

This study investigates the use of hand gestures to control the mouse cursor, replacing traditional methods such as computer mice, touchpads, and touchscreen displays. The authors utilized the MediaPipe library to detect and track hand gestures through a webcam, coupled with the PyAutoGUI library in Python to control the mouse cursor. Additionally, a voice assistant was employed to interact with the gesture recognition program and perform computer control tasks. The results demonstrate that the system achieves an accuracy rate of over 95% under good lighting conditions, with a simple background and close proximity, surpassing the accuracy of traditional methods. In summary, this virtual mouse system provides utility and enhances the computer interaction experience.

Keywords: virtual mouse, hand recognition, hand detection, mediapipe.

Nguyễn Đức Nhật Quang sinh ngày 08/10/1992 tại Thừa Thiên Huế. Năm 2015, ông tốt nghiệp kỹ sư chuyên ngành Điện tử - Viễn thông, Trường Đại học Khoa học, Đại học Huế. Năm 2020, ông nhận bằng thạc sĩ chuyên ngành Khoa học máy tính và Kỹ thuật thông tin (CSIE) tại Trường Đại học Quốc gia Thành Công (NCKU), Đài Loan. Hiện nay, ông đang công tác tại Khoa Điện, Điện tử và Công nghệ vật liệu, Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Thiết kế vi mạch số, Trí thông minh nhân tạo (AI), Internet vạn vật kết nối (IoT), Hệ thống nhúng.

Phan Thị Huỳnh Ngân sinh ngày 28/01/2001 tại Thừa Thiên Huế. Hiện nay, bà đang là sinh viên năm 5 chuyên ngành Điện tử - Viễn thông, Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Trí thông minh nhân tạo (AI).

Phan Văn Cường sinh ngày 19/10/2002 tại Hà Tĩnh. Hiện nay, ông đang là sinh viên năm 4 chuyên ngành Điện tử - Viễn thông, Trường Đại học Khoa học, Đại học Huế

Lĩnh vực nghiên cứu: Quản trị mạng, Kỹ thuật đảo ngược (Reverse Engineering)

Trần Thị Thu Huyền sinh ngày 23/08/1986 tại Tỉnh Hưng Yên. Năm 2008, bà tốt nghiệp cử nhân chuyên ngành Cử nhân Tin học, Trường Đại học Tây Nguyên. Hiện nay, bà đang công tác tại Trường THPT Lê Duẩn, Đắk Lắk.

Lĩnh vực nghiên cứu: Quản lý Công nghệ thông tin, Internet vạn vật kết nối (IoT)