**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

Description automatically generated

**BÁO CÁO TIỂU LUẬN**

***Đề tài:***

**NHẬN DIỆN CỬ CHỈ BÀN TAY ĐỂ TĂNG GIẢM ÂM LƯỢNG,**

**ĐỘ SÁNG, ZOOM TO NHỎ CAMERA SỬ DỤNG OPEN CV**

| **Sinh viên thực hiện:** | **Đỗ Hải Đăng** |
| --- | --- |
|  | **Nguyễn Văn Công** |
|  | **Phạm Văn Đức** |
|  | **Bùi Ngọc Lâm** |
| **Lớp:** | **CNTT K18G** |
| **Giáo viên hướng dẫn:** | **TS. Trần Quang Quý** |

***Thái Nguyên, tháng 10 năm 2023***

MỤC LỤC

# 

[CHƯƠNG 1: TỔNG QUAN VỀ HỌC MÁY VÀ OPEN CV 1](#_heading=h.tyjcwt)

[1.1. Tổng quan về học máy 1](#_heading=h.3dy6vkm)

[1.1.1. Khái niệm về học máy 1](#_heading=h.1t3h5sf)

[1.1.2. Cách hoạt động của học máy 1](#_heading=h.4d34og8)

[1.1.3. Các kiểu hệ thống học máy 2](#_heading=h.2s8eyo1)

[1.1.4. Một số khái niệm trong học máy 5](#_heading=h.35nkun2)

[1.1.5. Ứng dụng của học máy 6](#_heading=h.1ksv4uv)

[1.2. Tổng quan về thư viện Open CV 7](#_heading=h.44sinio)

[1.2.1. Lịch sử thư viện Open CV 7](#_heading=h.2jxsxqh)

[CHƯƠNG 2: GIẢI PHÁP VÀ KỸ THUẬT NHẬN DIỆN CỬ CHỈ BÀN TAY TRONG THƯ VIỆN OPEN CV 9](#_heading=h.3j2qqm3)

[2.1. Đặc trưng của thư viện Open CV 9](#_heading=h.1y810tw)

[2.1.1. Khái niệm về thư viện Open CV 10](#_heading=h.1ci93xb)

[2.1.2. Kiến trúc chung của thư viện Open CV 11](#_heading=h.3whwml4)

[2.1.3. Tính năng và các module phổ biến của OpenCV 12](#_heading=h.qsh70q)

[2.1.4. Hàm kích hoạt trong thư viện Open CV 14](#_heading=h.1pxezwc)

[2.1.5. Ứng dụng thư viện Open CV 14](#_heading=h.49x2ik5)

[2.2. Cách dùng OpenCV trong Python để xử lý ảnh 15](#_heading=h.2p2csry)

[2.2.1. Đọc hình ảnh trong OpenCV 15](#_heading=h.147n2zr)

[2.2.2. Hiển thị hình ảnh trong OpenCV 16](#_heading=h.32hioqz)

[2.2.3. Lưu hình ảnh trong OpenCV 17](#_heading=h.41mghml)

[2.3. Nhận diện cử chỉ tay trong Open CV 17](#_heading=h.vx1227)

[2.3.1. Giới thiệu 17](#_heading=h.3fwokq0)

[2.3.2. Mediapipe Hands 17](#_heading=h.1v1yuxt)

[2.3.3. Các bước để nhận diện cử chỉ tay sử dụng Mediapipe Hands trong OpenCV 18](#_heading=h.4f1mdlm)

[2.3.4. Ưu nhược điểm của Mediapipe Hands 20](#_heading=h.111kx3o)

[CHƯƠNG 3: ỨNG DỤNG THƯ VIỆN OPEN CV XÂY DỰNG MÔ](#_heading=h.3l18frh) [HÌNH NHẬN DIỆN BÀN TAY ĐỂ TĂNG GIẢM ÂM LƯỢNG ĐỘ SÁNG ZOOM CAMERA 21](#_heading=h.206ipza)

[3.1 Công cụ, cấu trúc thư mục xây dựng và huấn luyện mô hình 21](#_heading=h.4k668n3)

[3.1.1. Công cụ 21](#_heading=h.2zbgiuw)

[3.1.2. Cấu trúc thư mục 23](#_heading=h.3cqmetx)

[3.2 Thu thập và trực quan hóa dữ liệu 24](#_heading=h.4bvk7pj)

[3.2.1 Tọa độ khung xương 24](#_heading=h.2r0uhxc)

[3.2.2 Các cử chỉ tay để điều chỉnh 25](#_heading=h.1664s55)

[3.2.3 Trực quan hóa dữ liệu 26](#_heading=h.25b2l0r)

[3.3 Các bước để xây dựng một dự án bằng Python – Nhận diện cử chỉ bàn tay 27](#_heading=h.34g0dwd)

[3.3.1 Xử lý dữ liệu cơ bản 27](#_heading=h.1jlao46)

[3.4 Xây dựng và huấn luyện mô hình 27](#_heading=h.2iq8gzs)

[3.4.1 Cấu hình và khai báo đối tượng 27](#_heading=h.xvir7l)

[3.4.2 Chuyển thành ảnh màu 28](#_heading=h.3hv69ve)

[3.4.3 Từ các tọa độ => tăng giảm âm lượng , độ sáng , zoom camera 28](#_heading=h.1x0gk37)

[3.5. Thử nghiệm mô hình 28](#_heading=h.4h042r0)

[3.5.1 Code Demo 28](#_heading=h.2w5ecyt)

[3.5.2 Kết quả 31](#_heading=h.1opuj5n)

[KẾT LUẬN 32](#_heading=h.2nusc19)

[TÀI LIỆU THAM KHẢO 33](#_heading=h.3mzq4wv)

[DANH MỤC THUẬT NGỮ 34](#_heading=h.2250f4o)

[PHỤ LỤC CODE 35](#_heading=h.haapch)

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 44](#_heading=h.319y80a)

DANH MỤC HÌNH ẢNH

# 

[Hình 1.1: Hình ảnh học máy có giám sát 2](#_heading=h.17dp8vu)

[Hình 1.2: Hình ảnh học máy không giám sát 3](#_heading=h.3rdcrjn)

[Hình 1.3: Hình ảnh học máy bán giám sát 4](#_heading=h.26in1rg)

[Hình 1.4: Hình ảnh học tăng cường 5](#_heading=h.lnxbz9)

[Hình 2.1: Thư viện Open Cv 9](#_heading=h.4i7ojhp)

[Hình 2.2: Cấu trúc của Open CV cung cấp 12](#_heading=h.2bn6wsx)

[Hình 2.3: Tính năng của Open CV 13](#_heading=h.3as4poj)

[Hình 2.4: Đọc hình ảnh từ tệp tin 15](#_heading=h.3o7alnk)

[Hình 2.5: Đọc hình ảnh từ URL 15](#_heading=h.23ckvvd)

[Hình 2.6: Đọc hình ảnh từ webcam 16](#_heading=h.ihv636)

[Hình 2.7: Hiển thị hình ảnh 16](#_heading=h.1hmsyys)

[Hình 2.8: Lưu hình ảnh 17](#_heading=h.2grqrue)

[Hình 2.9: Import các thư viện 18](#_heading=h.2u6wntf)

[Hình 2.10: Cấu hình và khởi tạo Mediapipe Hands 18](#_heading=h.19c6y18)

[Hình 2.11: Mở kết nối với webcam và đọc hình ảnh 18](#_heading=h.3tbugp1)

[Hình 2.12: Chuyển đổi màu ảnh từ BGR sang RGB 18](#_heading=h.28h4qwu)

[Hình 2.13: Xử lý ảnh và nhận diện cử chỉ tay 19](#_heading=h.nmf14n)

[Hình 2.14: Hiển thị kết quả nhận diện cử chỉ bàn tay 19](#_heading=h.37m2jsg)

[Hình 2.15: Hiển thị hình ảnh lên màn hình 19](#_heading=h.1mrcu09)

[Hình 2.16: Thoát khỏi vàng lặp 19](#_heading=h.46r0co2)

[Hình 2.17: Đóng kết nối với webcam 19](#_heading=h.2lwamvv)

[Hình 3.1: Truy cập Anaconda từ trình duyệt web 21](#_heading=h.3ygebqi)

[Hình 3.2: Khởi chạy PyCharm từ Anaconda 22](#_heading=h.2dlolyb)

[Hình 3.3: Cài đặt PyCharm bằng pip 22](#_heading=h.sqyw64)

[Hình 3.4: Cấu trúc thư mục 24](#_heading=h.1rvwp1q)

[Hình 3.6: Cử chỉ tay để điều chỉnh 25](#_heading=h.3q5sasy)

[Hình 3.7: Biểu đồ đường hiển thị sự thay đổi của các cử chỉ tay qua thời gian. 26](#_heading=h.kgcv8k)

[Hình 3.8: Biểu đồ phân bố cử chỉ sau khi xử lý và loại bỏ dữ liệu dày và thưa 27](#_heading=h.43ky6rz)

[Hình 3.9: Import Thư Viện 28](#_heading=h.1baon6m)

[Hình 3.11: Chuẩn bị camera và chức năng vẽ 29](#_heading=h.3vac5uf)

[Hình 3.12: Tăng giảm âm lượng 29](#_heading=h.2afmg28)

[Hình 3.13: Tăng giảm độ sáng 30](#_heading=h.pkwqa1)

[Hình 3.14: Zoom to nhỏ camera 30](#_heading=h.39kk8xu)

[Hình 1.35: Kết quả 31](#_heading=h.48pi1tg)

**BẢNG PHÂN CÔNG CÔNG VIỆC**

| **STT** | **Họ và Tên** | **Công Việc** |
| --- | --- | --- |
| 1 | Đỗ Hải Đăng | * Tìm hiểu về thư viện Open CV * Xây dựng demo * Tổng hợp báo cáo làm word , pp |
| 2 | Nguyễn Văn Công | * Tìm hiểu về Machine Learning * Xây dựng demo |
| 3 | Phạm Văn Đức | * Tìm hiểu về thư viện Open CV * Xây dựng demo |
| 4 | Bùi Ngọc Lâm | * Tìm hiểu về Machine Learning * Xây dựng demo |

# CHƯƠNG 1: TỔNG QUAN VỀ HỌC MÁY VÀ OPEN CV

# 1.1. Tổng quan về học máy

## 1.1.1. Khái niệm về học máy

Khái niệm học có nghĩa rộng giống như sự thông minh, bao gồm cả qúa trình và khó có một định nghĩa chính xác. Theo nghĩa tự điển, học là quá trình thu nhận kiến thức, kỹ năng do người khác truyền lại hoặc đọc đi, đọc lại, nghiền ngẫm ghi nhớ (học thuộc lòng). Rộng hơn, học bao gồm cả quá trình đúc rút tri thức từ các quan sát, trải nghiệm thực tiễn.

Học máy (machine learning) mang hai nghĩa thông dụng: 1) sử dụng máy tính để khám phá tri thức từ dữ liệu, 2) sự học trong máy (tác tử: agent). Về phương diện công nghệ, học máy là một lĩnh vực của trí tuệ nhân tạo, trong đó nghiên cứu các kỹ thuật xây dựng và phát triển các chương trình máy tính có thể thích nghi và "học" từ các dữ liệu mẫu hoặc kinh nghiệm. Đến nay, đã có nhiều định nghĩa cho khái niệm này, tuy nhiên khó có một định nghĩa thỏa đáng được mọi người thừa nhận. Định nghĩa sau phát triển từ định nghĩa của T. Mitchell cho ta cách nhìn toán học của một chương trình học khi nghiên cứu, thiết kế.

**Định nghĩa:** Một chương trình máy tính được gọi là học từ dữ liệu/ kinh nghiệm E đối với lớp nhiệm vụ T và độ đo mức thực hiện P nếu việc thực hiện các nhiệm vụ T  
của nó khi đo bằng P được cải tiến nhờ dữ liệu hoặc kinh nghiệm E.

Theo định nghĩa này, người ta cần tối ưu hóa độ đo thực hiện P dựa trên phân  
tích dữ liệu/ kinh nghiệm E để tìm cách thực hiện nhiệm vụ T tốt nhất. Để hiểu rõ  
hơn, ta hãy làm quen với một số ví dụ và một số vấn đề liên quan.

## 1.1.2. Cách hoạt động của học máy

Học máy chắc chắn là một trong những tập hợp con thú vị nhất của Trí tuệ nhân tạo. Nó hoàn thành nhiệm vụ học từ dữ liệu với đầu vào cụ thể cho máy. Điều quan trọng là phải hiểu điều gì làm cho Machine Learning hoạt động và do đó, nó có thể được sử dụng như thế nào trong tương lai.

Quá trình Machine Learning bắt đầu bằng việc nhập dữ liệu đào tạo vào thuật toán đã chọn. Dữ liệu đào tạo là dữ liệu đã biết hoặc chưa biết để phát triển thuật toán Machine Learning cuối cùng. Loại đầu vào dữ liệu đào tạo có ảnh hưởng đến thuật toán và khái niệm đó sẽ được đề cập thêm trong giây lát. Dữ liệu đầu vào mới được đưa vào thuật toán máy học để kiểm tra xem thuật toán có hoạt động chính xác hay không.

Dự đoán và kết quả sau đó được kiểm tra với nhau. Nếu dự đoán và kết quả không khớp, thuật toán sẽ được đào tạo lại nhiều lần cho đến khi nhà khoa học dữ liệu nhận được kết quả mong muốn. Điều này cho phép thuật toán máy học liên tục tự học và đưa ra câu trả lời tối ưu, độ chính xác tăng dần theo thời gian.

Thuật toán Machine Learning được hướng dẫn để sử dụng một bộ dữ liệu đào tạo, từ đó tạo ra một mô hình nguyên mẫu. Khi thuật toán này tiếp nhận dữ liệu mới, nó sẽ đưa ra những dự đoán phân tích dựa trên nguyên mẫu căn bản.

Những phân tích nói trên sẽ được đánh giá về độ chính xác. Nếu độ chính xác này được công nhận, thuật toán Machine Learning sẽ tiến hành triển khai. Ngược lại, nếu độ chính xác không được công nhận, thuật toán sẽ được hướng dẫn lại nhiều lần với một bộ dữ liệu hướng dẫn tăng dần.

## 1.1.3. Các kiểu hệ thống học máy

Học máy có giám sát

Machine Learning được giám sát, hay còn được gọi là học máy có giám sát, được định nghĩa bằng cách sử dụng các tập dữ liệu được gắn nhãn để tạo các thuật toán nhằm phân loại dữ liệu hoặc dự đoán kết quả một cách chính xác.

A diagram of a diagram

Description automatically generated

*Hình 1.1: Hình ảnh học máy có giám sát*

Học máy có giám sát giúp các tổ chức giải quyết nhiều vấn đề trong thế giới thực trên quy mô lớn, chẳng hạn như phân loại thư rác trong một thư mục riêng biệt từ hộp thư đến. Một số phương pháp được sử dụng trong học máy có giám sát bao gồm mạng nơ-ron, Navie Bayes, hồi quy tuyến tính, hồi quy logistic, Random Forest, thuật toán SVM,…

**Học máy không giám sát**

Machine Learning không giám sát, còn được gọi là học máy không giám sát, sử dụng các thuật toán học máy để phân tích và phân cụm các tập dữ liệu không được gắn nhãn. Các thuật toán này phát hiện ra các mẫu hoặc nhóm dữ liệu ẩn mà không cần sự can thiệp của con người.

A diagram of a process

Description automatically generated

*Hình 1.2: Hình ảnh học máy không giám sát*

Khả năng phát hiện ra những điểm tương đồng và khác biệt trong thông tin khiến nó trở thành giải pháp lý tưởng để phân tích dữ liệu khám phá, chiến lược bán chéo, phân khúc khách hàng, nhận dạng hình ảnh và mẫu.

Học máy không có giám sát cũng được sử dụng để giảm số lượng tính năng trong một mô hình thông qua quá trình giảm kích thước. Phân tích thành phần chính (PCA) và phân tích giá trị đơn lẻ (SVD) là hai cách tiếp cận phổ biến cho việc này. Các thuật toán khác được sử dụng trong machine learning không giám sát bao gồm neural network, phân cụm K-mean, phương pháp phân nhóm xác suất,…

**Học máy bán giám sát**

Học máy bán giám sát cung cấp một phương tiện kết nối giữa Machine Learning có giám sát và không giám sát. Trong quá trình đào tạo, nó sử dụng một tập dữ liệu có nhãn nhỏ hơn để hướng dẫn phân loại và trích xuất tính năng từ một tập dữ liệu lớn hơn mà không được gắn nhãn. Semi-supervised Learning có thể giải quyết vấn đề không có đủ dữ liệu được gắn nhãn để đào tạo thuật toán học có giám sát.

A diagram of different types of data

Description automatically generated

*Hình 1.3: Hình ảnh học máy bán giám sát*

**Học tăng cường**

Học tăng cường là các bài toán giúp cho một hệ thống tự động xác định hành vi dựa trên hoàn cảnh để đạt được lợi ích cao nhất (maximizing the performance). Hiện tại, học tăng cường chủ yếu được áp dụng vào Lý Thuyết Trò Chơi (Game Theory), các thuật toán cần xác định nước đi tiếp theo để đạt được điểm số cao nhất.

Huấn luyện cho máy tính chơi game Mario. Đây là một chương trình thú vị dạy máy tính chơi game Mario. Đầu vào của thuật toán là sơ đồ của màn hình tại thời điểm hiện tại, nhiệm vụ của thuật toán là với đầu vào đó, tổ hợp phím nào nên được bấm. Việc huấn luyện này được dựa trên điểm số cho việc di chuyển được bao xa trong thời gian bao lâu trong game, càng xa và càng nhanh thì được điểm thưởng càng cao (điểm thưởng này không phải là điểm của trò chơi mà là điểm do chính người lập trình tạo ra). Thông qua huấn luyện, thuật toán sẽ tìm ra một cách tối ưu để tối đa số điểm trên, qua đó đạt được mục đích cuối cùng là cứu công chúa.

Học tăng cường (RL) là phương pháp có cấu trúc rất khác. Hệ thống học trong RL được gọi là tác nhân (agent). Nó có thể quan sát môi trường xung quanh và chọn việc hành động, sau đó nhận về điểm thưởng (reward) hoặc lượng phạt (penalty) dưới dạng điểm thưởng âm, hệ thống sau đó cần tự học chiến lược tốt nhất để nhận về nhiều điểm thưởng nhất qua thời gian, và chiến lược này được gọi là chính sách (policy).

A diagram of a fire extinguisher

Description automatically generated

*Hình 1.4: Hình ảnh học tăng cường*

## 1.1.4. Một số khái niệm trong học máy

**Dataset:** Là tập dữ liệu ở dạng nguyên thủy chưa được xử lý mà bạn đã thu thập được ở bước data collection. Một dataset có thể chứa nhiều data point.

**Data point:** Là một đơn vị thông tin độc lập trong tập dữ liệu của bạn, chẳng hạn như bạn có các tập dữ liệu bao gồm giá nhà, ngày xây, diện tích,… thì một điểm dữ liệu sẽ gồm các thông tin đó. Một tập hợp các điểm dữ liệu được gọi là tập dữ liệu.

**Training data và test data:** Dataset thường được chia làm 2 tập này, training data có nhiệm vụ để huấn luyện trong mô hình Machine Learning Workflow, test data để dự đoán kết quả và đánh giá mô hình.

**Features vector:** Là vector đặc trưng, mỗi vector mang nhiệm vụ biểu diễn cho một điểm dữ liệu trong dataset. Mỗi vector đều có n chiều biểu diễn các đặc trưng của điểm dữ liệu, mỗi đặc trưng là một chiều và phải là dữ liệu số. Các mô hình chỉ có thể huấn luyện được từ các vector đặc trưng này, do đó dataset cần phải chuyển về dạng một tập các vector đặc trưng (features vectors).

**Model:** Là những mô hình thường được dùng training trên các training data dựa theo thuật toán của mô hình. Và sau đó mô hình sẽ dự đoán hoặc đưa các quyết định dựa trên những gì đã được học.

## 1.1.5. Ứng dụng của học máy

**Nhận diện hình ảnh (Image Recognition):** Nhận dạng hình ảnh cũng là một trong những ứng dụng phổ biến của Machine Learning. Machine Learning cũng được ứng dụng để phát hiện khuôn mặt trong ảnh của nhiều người. Có một danh mục riêng cho mỗi người trong cơ sở dữ liệu hình ảnh của nhiều người.

**Nhận dạng giọng nói:**Hay còn được gọi là nhận dạng giọng nói tự động (ASR – Automatic Speech Recognition), sử dụng Natural Language Proccessing (NLP – xử lý ngôn ngữ tự nhiên) để xử lý giọng nói của con người thành định dạng viết. Nhiều thiết bị di động tích hợp tính năng nhận dạng giọng nói vào hệ thống để thực hiện tìm kiếm bằng giọng nói.

**Dịch vụ khách hàng:** Chatbot trực tuyến đang thay thế các tác nhân con người trong quá trình giao tiếp với khách hàng. Chatbot trả lời các câu hỏi thường gặp (FAQ) xung quanh các chủ đề, như vận chuyển hoặc cung cấp lời khuyên được cá nhân hóa, bán sản phẩm chéo hoặc đề xuất kích thước cho người dùng, thay đổi cách chúng ta nghĩ về mức độ tương tác của khách hàng trên các trang web và nền tảng truyền thông xã hội.

**Computer vision:** Công nghệ AI này cho phép máy tính và hệ thống lấy thông tin có ý nghĩa từ hình ảnh kỹ thuật số, video và các đầu vào trực quan khác. Dựa trên các đầu vào đó có thể thực hiện hành động. Khả năng cung cấp các khuyến nghị này phân biệt chúng với các nhiệm vụ nhận dạng hình ảnh. Được hỗ trợ bởi mạng nơ-ron phức hợp, computer vision có các ứng dụng trong việc gắn thẻ ảnh trên mạng xã hội, chụp ảnh X quang trong chăm sóc sức khỏe và xe hơi tự lái trong ngành công nghiệp ô tô.

**Công cụ đề xuất:** Sử dụng dữ liệu hành vi tiêu dùng trong quá khứ, các thuật toán AI có thể giúp khám phá các xu hướng dữ liệu có thể được sử dụng để phát triển các chiến lược bán chéo hiệu quả hơn. Điều này được sử dụng để đưa ra các đề xuất bổ sung có liên quan cho khách hàng trong quá trình thanh toán cho các nhà bán lẻ trực tuyến.

**Giao dịch chứng khoán tự động:** Được thiết kế để tối ưu hóa danh mục đầu tư chứng khoán, các nền tảng giao dịch tần suất cao do AI điều khiển thực hiện hàng nghìn hoặc thậm chí hàng triệu giao dịch mỗi ngày mà không cần sự can thiệp của con người.

# 1.2. Tổng quan về thư viện Open CV

OpenCV (Open Source Computer Vision) là một thư viện mã nguồn mở được thiết kế để xử lý hình ảnh và thị giác máy tính. Được phát triển bởi Intel, OpenCV đã trở thành một dự án mã nguồn mở phổ biến và mạnh mẽ trong lĩnh vực thị giác máy tính và xử lý ảnh. OpenCV là một công cụ quan trọng trong lĩnh vực thị giác máy tính, và sử dụng nó đơn giản và linh hoạt cho nhiều ứng dụng khác nhau từ xử lý ảnh cơ bản đến các ứng dụng phức tạp hơn.

## 1.2.1. Lịch sử thư viện Open CV

Lịch sử của thư viện OpenCV (Open Source Computer Vision) bắt đầu từ những năm đầu của thập kỷ 2000 và đã trải qua nhiều phát triển và mở rộng từ đó. Dưới đây là một cái nhìn tổng quan về lịch sử của OpenCV:

Năm 2000, OpenCV bắt đầu dưới dạng một dự án nghiên cứu tại Intel, được bắt đầu bởi Gary Bradski. Mục tiêu chính của dự án là cung cấp một thư viện mã nguồn mở cho cộng đồng nghiên cứu và phát triển trong lĩnh vực thị giác máy tính.

Năm 2006, Phiên bản 1.0 của OpenCV được phát hành. Đây là bước quan trọng đánh dấu sự ổn định và chính xác của thư viện, giúp nó trở thành một công cụ phổ biến trong cả cộng đồng nghiên cứu và ngành công nghiệp.

Năm 2011, OpenCV chính thức trở thành một dự án mã nguồn mở độc lập và không còn thuộc quản lý của Intel. Điều này mở cửa cho sự đóng góp rộng rãi từ cộng đồng phát triển.

Năm 2012, OpenCV 2.4 được phát hành với nhiều cải tiến và bổ sung, bao gồm cả việc hỗ trợ CUDA cho tăng tố hiệu suất trên GPU NVIDIA.

Năm 2015, OpenCV 3.0 được phát hành với nhiều tính năng mới và cải tiến, bao gồm hỗ trợ C++11, module mới như xử lý thị giác máy tính toàn diện (computational photography), và sự đa dạng trong việc hỗ trợ các ngôn ngữ lập trình.

Năm 2018, OpenCV 4.0 được giới thiệu, với các cải tiến lớn trong hiệu suất và chất lượng. Bản phát hành này bao gồm sự tích hợp với dlib và TensorFlow, cũng như nhiều cập nhật khác nhau.

Hiện nay 2023, OpenCV tiếp tục phát triển và cập nhật. Nó trở thành một trong những thư viện thị giác máy tính hàng đầu, được sử dụng rộng rãi trong cả nghiên cứu và các dự án ứng dụng thực tế trên toàn thế giới.

OpenCV không chỉ là một công cụ quan trọng trong nghiên cứu và phát triển, mà còn là một nguồn lực toàn diện và mở rộng cho cộng đồng toàn cầu đang làm việc trong lĩnh vực thị giác máy tính và xử lý ảnh.

# CHƯƠNG 2: GIẢI PHÁP VÀ KỸ THUẬT NHẬN DIỆN CỬ CHỈ BÀN TAY TRONG THƯ VIỆN OPEN CV

# 2.1. Đặc trưng của thư viện Open CV

A blue and black logo

Description automatically generated

*Hình 2.1: Thư viện Open Cv*

Đặc trưng của thư viện OpenCV (Open Source Computer Vision) bao gồm một loạt các tính năng mạnh mẽ và linh hoạt cho xử lý ảnh và thị giác máy tính. Dưới đây là một số đặc điểm chính của OpenCV:

- Hỗ trợ Đa Ngôn Ngữ và Đa Nền Tảng: OpenCV được viết bằng C++ nhưng cung cấp các giao diện API cho nhiều ngôn ngữ lập trình như Python, Java, và C#. Nó cũng hỗ trợ nhiều hệ điều hành như Windows, Linux, macOS, Android và iOS.

- Module Chuyên Biệt: OpenCV chia thành các module chuyên biệt, giúp người phát triển dễ dàng chọn lựa chức năng cần thiết. Các module bao gồm xử lý ảnh cơ bản, thị giác máy tính, machine learning, video, tracking, và nhiều hơn nữa.

- Hiệu Suất Cao: OpenCV được tối ưu hóa để đảm bảo hiệu suất cao. Nó sử dụng các thư viện tối ưu hóa như Intel IPP (Integrated Performance Primitives) để thực hiện các phép toán nhanh chóng trên CPU.

- Thực Hiện Trên GPU: OpenCV hỗ trợ tính toán trên GPU thông qua CUDA (Compute Unified Device Architecture) của NVIDIA. Điều này cho phép tận dụng hiệu suất của GPU để xử lý ảnh nhanh chóng.

- Algorithms và Data Structures: OpenCV cung cấp nhiều thuật toán và cấu trúc dữ liệu để xử lý ảnh và thị giác máy tính. Bạn có thể sử dụng chúng để thực hiện các nhiệm vụ như lọc ảnh, nhận diện đối tượng, phát hiện cạnh, và nhiều chức năng khác.Mạng Nơ-ron tích chập (Convulational Neural Network).

- Hỗ Trợ Camera và Video: OpenCV cung cấp các công cụ để làm việc với video và camera, bao gồm đọc và ghi video, xử lý từng khung hình, và các chức năng khác liên quan đến video.

- Machine Learning và Deep Learning: OpenCV tích hợp các tính năng học máy cơ bản và có khả năng tích hợp với các thư viện học máy như scikit-learn, TensorFlow, và PyTorch. Nó cũng hỗ trợ các mô hình học máy sâu thông qua module cv2.dnn.

- Cộng Đồng Rộng Lớn: OpenCV có một cộng đồng lớn và tích cực. Người dùng có thể tìm thấy tài liệu, diễn đàn, và nguồn lực trực tuyến rất phong phú để hỗ trợ trong quá trình sử dụng và phát triển OpenCV.

- Ứng Dụng Rộng Rãi: OpenCV được sử dụng rộng rãi trong nhiều lĩnh vực như xử lý ảnh y tế, ô tô tự lái, thị giác máy tính, thực tế ảo, và nhiều ứng dụng khác.

Những đặc trưng này khiến cho OpenCV trở thành một công cụ quan trọng và mạnh mẽ trong lĩnh vực thị giác máy tính và xử lý ảnh.

## 2.1.1. Khái niệm về thư viện Open CV

- OpenCV (Open Source Computer Vision) là một thư viện mã nguồn mở chuyên về xử lý ảnh và thị giác máy tính. Nó được thiết kế để cung cấp một loạt các công cụ và thuật toán cho việc xử lý và phân tích dữ liệu hình ảnh.

- OpenCV là thư viện nguồn mở hàng đầu cho Computer Vision và Machine Learning, và hiện có thêm tính năng tăng tốc GPU cho các hoạt động theo real-time. OpenCV được phát hành theo giấy phép BSD (\*), do đó nó miễn phí cho cả học tập và sử dụng với mục đích thương mại.

- Nó có trên các giao diện C++, C, Python và Java và hỗ trợ Windows, Linux, Mac OS, iOS và Android. OpenCV được thiết kế để hỗ trợ hiệu quả về tính toán và chuyên dùng cho các ứng dụng real-time (thời gian thực). Nếu được viết trên C/C++ tối ưu, thư viện này có thể tận dụng được bộ xử lý đa lõi (multi-core processing).

- OpenCV có một cộng đồng người dùng khá hùng hậu hoạt động trên khắp thế giới bởi nhu cầu cần đến nó ngày càng tăng theo xu hướng chạy đua về sử dụng computer vision của các công ty công nghệ. OpenCV hiện được ứng dụng rộng rãi toàn cầu, với cộng đồng hơn 47.000 người, với nhiều mục đích và tính năng khác nhau từ interactive art, đến khai thác mỏ, khai thác web map hoặc qua robotic cao cấp.

## 2.1.2. Kiến trúc chung của thư viện Open CV

- Kiến trúc của thư viện OpenCV có sự phức tạp, với sự tổ chức và phân chia chức năng thành các module và thành phần khác nhau để dễ dàng quản lý và mở rộng. Dưới đây là một cái nhìn tổng quan về kiến trúc chung của OpenCV:

* Module Chính:
  + OpenCV được chia thành các module chính, mỗi module tập trung vào một lĩnh vực cụ thể của xử lý ảnh và thị giác máy tính. Một số module quan trọng bao gồm:
    - cv2.core: Chứa các cấu trúc dữ liệu cơ bản như mảng và ma trận.
    - cv2.imgproc: Cung cấp các hàm xử lý ảnh cơ bản như làm mịn, làm nét, biến đổi hình ảnh, và các phép toán trên pixel.
    - cv2.video: Chứa các chức năng liên quan đến xử lý video như đọc và ghi video, tracking, và optical flow.
    - cv2.highgui: Cung cấp các công cụ tương tác với người dùng như hiển thị ảnh và video, và giao diện người dùng đơn giản.
* Thư Viện Nền Tảng và Ngôn Ngữ:
  + OpenCV hỗ trợ nhiều nền tảng như Windows, Linux, macOS, Android, và iOS. Nó cung cấp các giao diện API cho nhiều ngôn ngữ lập trình như C++, Python, Java, và C#.
* Hỗ Trợ GPU:
  + OpenCV có tích hợp sẵn hỗ trợ GPU thông qua CUDA của NVIDIA. Điều này cho phép tính toán trên GPU, tăng cường hiệu suất xử lý ảnh đặc biệt là trong các ứng dụng đòi hỏi nhiều tính toán.
* Mô Hình Học Máy và Deep Learning:
  + OpenCV có các module cung cấp tính năng học máy cơ bản và có khả năng tích hợp với các thư viện học máy như scikit-learn. Nó cũng hỗ trợ tích hợp với các mô hình học máy sâu thông qua module cv2.dnn.
* Hiệu Suất và Tối Ưu Hóa:
  + OpenCV được tối ưu hóa để đảm bảo hiệu suất cao. Các thư viện tối ưu hóa như Intel IPP được sử dụng để thực hiện các phép toán nhanh chóng trên CPU.
* Cộng Đồng và Tài Nguyên:
  + OpenCV có một cộng đồng lớn và đa dạng. Tài liệu chi tiết, diễn đàn hỗ trợ, và nguồn lực trực tuyến phong phú là những nguồn hỗ trợ mạnh mẽ cho người sử dụng và nhà phát triển.
* Module Chuyên Biệt:
  + Ngoài các module chính, OpenCV còn có các module chuyên biệt khác như cv2.face cho nhận diện khuôn mặt và cv2.text cho nhận diện văn bản.

Kiến trúc của OpenCV được thiết kế sao cho người sử dụng có thể dễ dàng tận dụng các chức năng cơ bản và mở rộng chúng để giải quyết các vấn đề cụ thể trong lĩnh vực thị giác máy tính và xử lý ảnh.

A screenshot of a computer

Description automatically generated

*Hình 2.2: Cấu trúc của Open CV cung cấp*

## 2.1.3. Tính năng và các module phổ biến của OpenCV

* Xử lý và hiển thị Hình ảnh/ Video/ I/O (core, imgproc, highgui)
* Phát hiện các vật thể (objdetect, features2d, nonfree)
* Geometry-based monocular hoặc stereo computer vision (calib3d, stitching, videostab)
* Computational photography (photo, video, superres)
* Machine learning & clustering (ml, flann)
* CUDA acceleration (gpu)

A group of people in a car

Description automatically generated

*Hình 2.3: Tính năng của Open CV*

* OpenCV có cấu trúc module, nghĩa là gói bao gồm một số thư viện liên kết tĩnh (static libraries) hoặc thư viện liên kết động (shared libraries). Xin phép liệt kê một số định nghĩa chi tiết các module phổ biến có sẵn.
* Core functionality (core) – module nhỏ gọn để xác định cấu trúc dữ liệu cơ bản, bao gồm mảng đa chiều dày đặc và nhiều chức năng cơ bản được sử dụng bởi tất cả các module khác.
* Image Processing (imgproc) – module xử lý hình ảnh gồm cả lọc hình ảnh tuyến tính và phi tuyến (linear and non-linear image filtering), phép biến đổi hình học (chỉnh size, afin và warp phối cảnh, ánh xạ lại dựa trên bảng chung), chuyển đổi không gian màu, biểu đồ, và nhiều cái khác.
* Video Analysis (video) – module phân tích video bao gồm các tính năng ước tính chuyển động, tách nền, và các thuật toán theo dõi vật thể.
* Camera Calibration and 3D Reconstruction (calib3d) – thuật toán hình học đa chiều cơ bản, hiệu chuẩn máy ảnh single và stereo (single and stereo camera calibration), dự đoán kiểu dáng của đối tượng (object pose estimation), thuật toán thư tín âm thanh nổi (stereo correspondence algorithms) và các yếu tố tái tạo 3D.
* 2D Features Framework (features2d) – phát hiện các đặc tính nổi bật của bộ nhận diện, bộ truy xuất thông số, thông số đối chọi.
* Object Detection (objdetect) – phát hiện các đối tượng và mô phỏng của các hàm được định nghĩa sẵn – predefined classes (vd: khuôn mặt, mắt, cốc, con người, xe hơi,…).
* High-level GUI (highgui) – giao diện dễ dùng để thực hiện việc giao tiếp UI đơn giản.
* Video I/O (videoio) – giao diện dễ dùng để thu và mã hóa video.
* GPU – Các thuật toán tăng tốc GPU từ các modun OpenCV khác.
* … và một số module hỗ trợ khác, ví dụ như FLANN và Google test wrapper, Python binding, v.v.

## 2.1.4. Hàm kích hoạt trong thư viện Open CV

Trong thư viện OpenCV, hàm kích hoạt (activation function) thường được đề cập đến khi làm việc với mô hình học máy sâu (deep learning) hoặc mô hình thị giác máy tính. Tuy nhiên, OpenCV không thường xuyên sử dụng thuật ngữ "hàm kích hoạt" như trong các thư viện học máy sâu chính.

Thay vào đó, khi sử dụng OpenCV cho mục đích học máy sâu, bạn thường làm việc với các framework như TensorFlow hay PyTorch, chúng sẽ quản lý các hàm kích hoạt trong quá trình xây dựng và đào tạo mô hình.

Dưới đây là một số hàm kích hoạt phổ biến mà bạn có thể gặp trong các framework học máy sâu khi sử dụng OpenCV:

- Hàm ReLU được sử dụng rộng rãi với công thức f(x)=max(0,x) Nó giữ giá trị dương không thay đổi và chuyển đổi các giá trị âm thành 0.

- Hàm Sigmoid: Hàm Sigmoid chuyển đổi giá trị của một biến thành khoảng (0, 1) với công thức *f*(*x*)=1/(1+*e^*−*x*)​.

- Hàm Tanh: Hàm Tanh tương tự như Sigmoid, nhưng chuyển đổi giá trị về khoảng (-1, 1) với công thức (*x*)=(*e^*2*x-1)*/(*e^*2*x*+1)​.

- Hàm Softmax: Hàm Softmax được sử dụng cho bài toán phân loại đa lớp. Nó chuyển đổi một vector số thành một vector xác suất sao cho tổng các phần tử trong vector bằng 1.

Khi làm việc với các mô hình học máy sâu trong OpenCV, bạn thường chỉ cần sử dụng các hàm kích hoạt được tích hợp trong framework mà bạn đang sử dụng.

## 2.1.5. Ứng dụng thư viện Open CV

OpenCV được sử dụng cho đa dạng nhiều mục đích và ứng dụng khác nhau bao gồm [1] :

* Hình ảnh street view
* Kiểm tra và giám sát tự động
* Robot và xe hơi tự lái
* Phân tích hình ảnh y học
* Tìm kiếm và phục hồi hình ảnh/video
* Phim – cấu trúc 3D từ chuyển động
* Nghệ thuật sắp đặt tương tác

# 2.2. Cách dùng OpenCV trong Python để xử lý ảnh

## 2.2.1. Đọc hình ảnh trong OpenCV

* Đọc hình ảnh từ tệp tin:

A screenshot of a computer program

Description automatically generated

*Hình 2.4: Đọc hình ảnh từ tệp tin*

* Đọc hình ảnh từ URL:

A screenshot of a computer program

Description automatically generated

*Hình 2.5: Đọc hình ảnh từ URL*

* Đọc hình ảnh từ webcam:

A screenshot of a computer program

Description automatically generated

*Hình 2.6: Đọc hình ảnh từ webcam*

## 2.2.2. Hiển thị hình ảnh trong OpenCV

Để hiển thị hình ảnh trong OpenCV, bạn có thể sử dụng phương thức imshow() để tạo một cửa sổ hiển thị và hiển thị hình ảnh lên đó. Dưới đây là ví dụ về cách hiển thị hình ảnh trong OpenCV:

A screenshot of a computer program

Description automatically generated

*Hình 2.7: Hiển thị hình ảnh*

## 2.2.3. Lưu hình ảnh trong OpenCV

Để lưu hình ảnh trong OpenCV, bạn có thể sử dụng phương thức imwrite(). Phương thức này cho phép bạn lưu hình ảnh từ một mảng pixel hoặc từ một đối tượng hình ảnh vào một tệp tin trên đĩa.

Dưới đây là ví dụ về cách lưu hình ảnh trong OpenCV:

A screenshot of a computer program

Description automatically generated

*Hình 2.8: Lưu hình ảnh*

# 2.3. Nhận diện cử chỉ tay trong Open CV

## 2.3.1. Giới thiệu

Nhận diện cử chỉ tay là một lĩnh vực quan trọng trong thị giác máy tính và trí tuệ nhân tạo. Nó cho phép máy tính nhận biết và hiểu các cử chỉ và động tác của tay con người, mở ra những ứng dụng đa dạng từ điều khiển máy tính bằng cử chỉ, thị giác máy tính đến thực tế tăng cường và giao tiếp người-máy tiện lợi hơn. Trong báo cáo này, chúng ta sẽ tìm hiểu về giải pháp và kỹ thuật nhận diện cử chỉ tay trong thư viện OpenCV và Mediapipe.

## 2.3.2. Mediapipe Hands

Mediapipe Hands là một module trong thư viện Mediapipe, được phát triển bởi Google. Nó cung cấp khả năng nhận diện và theo dõi các điểm đặc trưng trên tay trong thời gian thực. Mediapipe Hands sử dụng một mô hình máy học để xác định vị trí các điểm đặc trưng như ngón tay, mắt cá chân và cổ tay. Bằng cách sử dụng Mediapipe Hands, chúng ta có thể nhận diện và theo dõi các cử chỉ tay một cách đáng tin cậy.

## 2.3.3. Các bước để nhận diện cử chỉ tay sử dụng Mediapipe Hands trong OpenCV

Dưới đây là các bước cơ bản để nhận diện cử chỉ tay sử dụng Mediapipe Hands trong OpenCV:

* Bước 1: Import các thư viện

A black and grey rectangular object

Description automatically generated

*Hình 2.9: Import các thư viện*

* Bước 2: Cấu hình và khởi tạo đối tượng Mediapipe Hands

A black and grey background with white text

Description automatically generated

*Hình 2.10: Cấu hình và khởi tạo Mediapipe Hands*

* Bước 3: Mở kết nối với webcam và đọc hình ảnh

A black and grey background with white text

Description automatically generated

*Hình 2.11: Mở kết nối với webcam và đọc hình ảnh*

* Bước 4: Chuyển đổi màu ảnh từ BGR sang RGB

A grey and black background with white text

Description automatically generated

*Hình 2.12: Chuyển đổi màu ảnh từ BGR sang RGB*

Bước 5: Sử dụng Mediapipe Hands để xử lý ảnh và nhận diện cử chỉ tay

A grey and black background with white text

Description automatically generated

*Hình 2.13: Xử lý ảnh và nhận diện cử chỉ tay*

* Bước 6: Hiển thị kết quả nhận diện cử chỉ trên hình ảnh

A screen shot of a computer

Description automatically generated

*Hình 2.14: Hiển thị kết quả nhận diện cử chỉ bàn tay*

* Bước 7: Hiển thị hình ảnh lên màn hình

A grey and black background with yellow text

Description automatically generated

*Hình 2.15: Hiển thị hình ảnh lên màn hình*

* Bước 8: Thoát khỏi vòng lặp khi nhấn phím Esc

A black and grey screen

Description automatically generated

*Hình 2.16: Thoát khỏi vàng lặp*

* Bước 9: Đóng kết nối với webcam

A black and grey flag

Description automatically generated

*Hình 2.17: Đóng kết nối với webcam*

## 2.3.4. Ưu nhược điểm của Mediapipe Hands

Ưu điểm của Mediapipe Hands:

* Hiệu suất tốt: Mediapipe Hands được tối ưu hóa để hoạt động với tốc độ cao và hiệu suất đáng kể. Nó có thể xử lý và nhận diện các cử chỉ tay trong thời gian thực, phù hợp với các ứng dụng yêu cầu độ trễ thấp.
* Độ chính xác cao: Mediapipe Hands sử dụng mô hình máy học được huấn luyện trên một lượng lớn dữ liệu. Điều này giúp nó đạt được độ chính xác cao trong việc nhận diện và theo dõi các điểm đặc trưng trên tay.
* Đa dạng cử chỉ tay: Mediapipe Hands có khả năng nhận diện và theo dõi các điểm đặc trưng trên tay như ngón tay, mắt cá chân, cổ tay. Điều này cho phép nó nhận diện và phân loại các cử chỉ tay phức tạp và đa dạng.
* Dễ sử dụng: Mediapipe Hands cung cấp API đơn giản và dễ sử dụng cho việc tích hợp vào các ứng dụng. Nó cung cấp các hàm và phương thức để xử lý và hiển thị kết quả nhận diện cử chỉ tay một cách thuận tiện.

Nhược điểm của Mediapipe Hands:

* Phụ thuộc vào điều kiện ánh sáng: Mediapipe Hands có thể ảnh hưởng bởi điều kiện ánh sáng trong môi trường. Nếu ánh sáng yếu hoặc không đồng đều, độ chính xác của việc nhận diện cử chỉ tay có thể giảm đi.
* Giới hạn về góc nhìn: Mediapipe Hands có thể gặp khó khăn trong việc nhận diện và theo dõi các điểm đặc trưng trên tay khi góc nhìn của hình ảnh thay đổi quá nhiều. Điều này có thể làm giảm độ chính xác của quá trình nhận diện.
* Khả năng nhận diện không hoàn hảo: Mặc dù Mediapipe Hands có độ chính xác cao, nhưng vẫn có khả năng nhận diện không hoàn hảo trong một số trường hợp đặc biệt. Các trường hợp này có thể bao gồm các cử chỉ tay phức tạp và sự chồng chéo giữa các ngón tay.
* Hạn chế về đa tay: Hiện tại, Mediapipe Hands hỗ trợ nhận diện và theo dõi tối đa hai tay trong một hình ảnh. Điều này có nghĩa là nó có hạn chế trong việc nhận diện đa tay hoặc cử chỉ tay của nhiều người cùng một lúc.

Điều quan trọng là nhận thức về những ưu điểm và nhược điểm của Mediapipe Hands giúp lựa chọn phù hợp khi áp dụng trong các ứng dụng cụ thể và đồng thời có thể tìm cách giải quyết nhược điểm để cải thiện hiệu suất và độ chính xác của hệ thống nhận diện cử chỉ tay.

# CHƯƠNG 3: ỨNG DỤNG THƯ VIỆN OPEN CV XÂY DỰNG MÔ

# HÌNH NHẬN DIỆN BÀN TAY ĐỂ TĂNG GIẢM ÂM LƯỢNG ĐỘ SÁNG ZOOM CAMERA

# 3.1 Công cụ, cấu trúc thư mục xây dựng và huấn luyện mô hình

## 3.1.1. Công cụ

Sử dụng PyCharm, công cụ này có lợi thế là thực hiện các thay đổi hiệu quả và nhanh chóng đối với cả biến cục bộ và biến toàn cục. Tái cấu trúc trong Pycharm cho phép các nhà phát triển cải thiện cấu trúc bên trong mà không thay đổi hiệu suất bên ngoài của mã.Nó cũng cho phép phân chia các lớp với các chức năng mở rộng hơn Cài đặt jupyter-notebook

Có hai cách để cài đặt PyCharm về máy:

+ Cách 1: Cài PyCharm thông qua Anaconda.

- B1: Truy cập vào Anaconda trên trình duyệt wed sau đó download và cài đặt.

A screenshot of a computer

Description automatically generated

*Hình 3.1: Truy cập Anaconda từ trình duyệt web*

- B2: Mở Anaconda sau khi cài đặt hoàn tất.

- B3: Nhấn chọn PyCharm để khởi chạy.

A screenshot of a computer

Description automatically generated

*Hình 3.2: Khởi chạy PyCharm từ Anaconda*

+ Cách 2: Cài PyCharm trực tiếp bằng Command Prompt.

- B1: Mở command Prompt từ thanh tìm kiếm trên Desktop.

- B2: Nhập lệnh “pip install PyCharm” nhấn enter để cài đặt.

A screenshot of a computer

Description automatically generated

*Hình 3.3: Cài đặt PyCharm bằng pip*

## 3.1.2. Cấu trúc thư mục

Trong PyCharm, cấu trúc thư mục của dự án được tổ chức dưới dạng một cây thư mục. Cấu trúc mặc định của PyCharm bao gồm các thư mục và tệp quan trọng sau:

- Thư mục gốc (Root folder):

Thư mục gốc của dự án, nơi tất cả các tệp và thư mục khác được tổ chức.

- Thư mục dự án (Project folder):

Thư mục dự án chứa tệp tin cấu hình dự án và các tệp khác liên quan đến dự án.

Thư mục này chứa tệp .idea (nếu tồn tại) và các tệp cấu hình khác như .gitignore, requirements.txt,...

- Thư mục mã nguồn (Source folder):

Thư mục mã nguồn chứa mã Python của dự án.

Thư mục này thường có tên là src hoặc source.

- Thư mục gói (Package folder):

Thư mục gói chứa các gói (packages) Python của dự án.

Mỗi gói có thể được tổ chức trong một thư mục con riêng.

- Thư mục tài nguyên (Resource folder):

Thư mục tài nguyên chứa các tệp không phải mã nguồn, chẳng hạn như hình ảnh, âm thanh, tệp cấu hình, v.v.

Thư mục này thường có tên là resources hoặc assets.

- Thư mục thử nghiệm (Test folder):

Thư mục thử nghiệm chứa các tệp liên quan đến việc kiểm tra và thử nghiệm mã nguồn.

Thư mục này thường có tên là tests.

- Thư mục tài liệu (Documentation folder):

Thư mục tài liệu chứa tài liệu liên quan đến dự án, chẳng hạn như tệp README, tệp hướng dẫn, v.v.

Thư mục này thường có tên là docs.

- Thư mục ảo (Virtual folder):

Thư mục ảo chứa các môi trường ảo Python và các gói được cài đặt trong môi trường ảo.

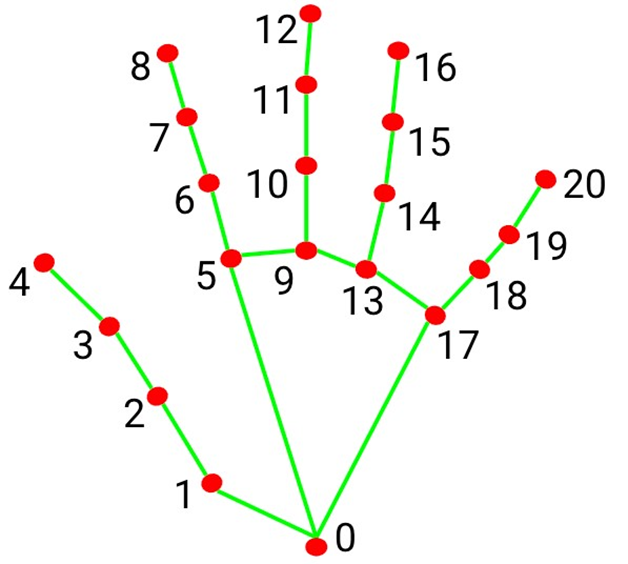
A black background with white text

Description automatically generated

*Hình 3.4: Cấu trúc thư mục*

# 3.2 Thu thập và trực quan hóa dữ liệu

## 3.2.1 Tọa độ khung xương



*Hình 3.5: Tọa độ khung xương*

## 3.2.2 Các cử chỉ tay để điều chỉnh

### A hand with a green and red dot pointing to the top Description automatically generated

*Hình 3.6: Cử chỉ tay để điều chỉnh*

## 3.2.3 Trực quan hóa dữ liệu

Dữ liệu sau khi được thu thập phải được trực quan bằng cách thống kê, tìm những mối liên hệ giữa các đặc trưng và quan sát sự phân phối của đặc trưng trong tập dữ liệu.A graph with blue and orange lines

Description automatically generated

*Hình 3.7: Biểu đồ đường hiển thị sự thay đổi của các cử chỉ tay qua thời gian.*

Trực quan hóa dữ liệu

- Biểu Đồ Thống Kê: Sử dụng thư viện như Matplotlib để vẽ biểu đồ thống kê về các giá trị âm lượng, độ sáng, và zoom theo thời gian hoặc theo cử chỉ bàn tay.

- Hiển Thị Video và Dữ Liệu Nhãn: Tạo một ứng dụng hoặc sử dụng notebook Jupyter để hiển thị video và dữ liệu nhãn tương ứng cùng một lúc. Bạn có thể sử dụng OpenCV để đọc video và Matplotlib để hiển thị dữ liệu.

- Heatmap và Grid Visualization: Nếu có thể, vẽ các heatmap hoặc lưới để trực quan hóa sự thay đổi của các thông số dựa trên cử chỉ bàn tay. Các điểm cử chỉ có thể được biểu thị bằng màu sắc trên heatmap hoặc lưới.

# 3.3 Các bước để xây dựng một dự án bằng Python – Nhận diện cử chỉ bàn tay

## 3.3.1 Xử lý dữ liệu cơ bản

Lọc và xử lý những dữ liệu phân bố không đều để giảm bớt sự mất cân bằng dữ liệu, dẫn tới mô hình không học được và không khái quát được dữ liệu ở những độ tuổi này.

A graph of hand gestures

Description automatically generated

*Hình 3.8: Biểu đồ phân bố cử chỉ sau khi xử lý và loại bỏ dữ liệu dày và thưa*

# 3.4 Xây dựng và huấn luyện mô hình

## 3.4.1 Cấu hình và khai báo đối tượng

mp\_drawing\_util = mp.solutions.drawing\_utils

mp\_drawing\_style = mp.solutions.drawing\_styles

mp\_hand = mp.solutions.hands

hands = mp\_hand.Hands model\_complexity=0,min\_detection\_confidence=0.5,

min\_tracking\_confidence=0.5 )

## 3.4.2 Chuyển thành ảnh màu

img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

result = hands.process(img)

img = cv2.cvtColor(img, cv2.COLOR\_RGB2BGR)

alphabet = ''

## 3.4.3 Từ các tọa độ => tăng giảm âm lượng , độ sáng , zoom camera

# 3.5. Thử nghiệm mô hình

## 3.5.1 Code Demo

A screen shot of a computer

Description automatically generated

*Hình 3.9: Import Thư Viện*

A screen shot of a computer program

Description automatically generated

*Hình 3.10: Menu*

*A screen shot of a computer program

Description automatically generated*

*Hình 3.11: Chuẩn bị camera và chức năng vẽ*

*A computer screen shot of a program

Description automatically generated*

*Hình 3.12: Tăng giảm âm lượng*

*A computer screen shot of a black screen

Description automatically generated*

*Hình 3.13: Tăng giảm độ sáng*

*A computer screen shot of a program

Description automatically generated*

*Hình 3.14: Zoom to nhỏ camera*

## 3.5.2 Kết quả

Kết quả thu được ở hình 3.15





*Hình 1.35: Kết quả*

# KẾT LUẬN

Báo cáo này nhóm chúng em đã nghiên cứu và tìm hiểu về Machine Learning, thư viện Open CV và các kỹ thuật xử lý, cách xây dựng mô hình. Với sự nỗ lực của nhóm và đặc biệt là sự giúp đỡ nhiệt tình của thầy giáo TS. Trần Quang Quý đã giúp nhóm em hoàn thành được báo cáo “Nhận diện cử chỉ bàn tay để tăng giảm âm lượng , độ sáng, zoom to nhỏ camera sử dụng Open CV”. Trong quá trình thực hiện nhóm em đã thu được những kết quả sau:

- Nắm được các khái niệm cơ bản trong lĩnh vực MachineLearning cũng như thư viện Open CV.

- Hiểu, nắm bắt được các phương pháp thu thập, trực quan và tiền xử lý dữ liệu phục vụ cho mô hình học sâu.

- Tầm nhìn máy tính có thể được loại bỏ để giải quyết những vấn đề hấp dẫn nhất với độ phức tạp cao nhất. Tất cả các nguyên tắc cơ bản liên quan đến kỹ thuật phát hiện theo những cách khác nhau để đạt được nó đã được có chủ ý sâu sắc.

Mặc dú đã rất nỗ lực, cố gắng nhưng do thời gian và kiến thức có hạn nên đề tài còn nhiều sai sót, hạn chế khiến cho mô hình xây dựng chưa đạt được kết quả cao. Chúng em xin nhận được sự thông cảm và góp ý của thầy cô trong ban cố vấn và các thầy cô trong hội đồng. Em xin chân thành cảm ơn.

# TÀI LIỆU THAM KHẢO

1. Vũ Hữu Tiệp và cộng sự (2018). Đắm mình vào học sâu .
2. Vũ Hữu Tiệp (2018). Machine Learning cơ bản, Nhà xuất bản Hà Nội.
3. Tìm hiểu về OpenCV : <https://viblo.asia/p/tim-hieu-ve-opencv-Do754NrXZM6>
4. Các tính năng của OpenCV mà bạn không thể bỏ qua : <https://websitehcm.com/cac-tinh-nang-cua-opencv-ma-ban-khong-the-bo-qua-vao-nam-2022/>
5. Source code tham khảo : <https://hanam88.com/kho-tai-lieu/61/120/review-bai-tap-lon-mon-python-ai-nhan-dien-cu-chi-ban-tay-va-hien-chi-chu-cai-c2009gi1-nguyen-duc-vinh-bach-khoa-aptech.html>

# DANH MỤC THUẬT NGỮ

| STT | Từ chuyên môn | Nghĩa |
| --- | --- | --- |
| 1 | OpenCV | OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở chuyên về xử lý ảnh và thị giác máy tính. Nó cung cấp các công cụ và thuật toán để xử lý, phân tích và nhận dạng ảnh và video. |
| 2 | Mediapipe | Mediapipe là một nền tảng mã nguồn mở của Google được sử dụng để xây dựng ứng dụng thị giác máy tính và xử lý dữ liệu đa phương tiện thời gian thực. Nó cung cấp các khung công cụ và thuật toán cho việc phát hiện, theo dõi và phân tích các thành phần như cử chỉ tay, khuôn mặt, điểm đặt trên khuôn mặt, v.v. |
| 3 | Supervised learning | Học có giám sát |
| 4 | Unsupervised Learning | Học không giám sát |
| 5 | Semi-Supervised Learning | Học bán giám sát |
| 6 | Reinforcement Learning | Học tăng cường |

# PHỤ LỤC CODE

# Cài đặt và sử dụng các thư viện

# OpenCV, Mediapipe, math, ctypes, comtypes, pycaw, numpy, screen-brightness-control

import cv2

import mediapipe as mp

from math import hypot

from ctypes import cast, POINTER

from comtypes import CLSCTX\_ALL

from pycaw.pycaw import AudioUtilities, IAudioEndpointVolume

import numpy as np

import screen\_brightness\_control as sbc

# MENU

def main():

while True:

print("-------- MENU --------")

print("1. Chỉnh âm lượng")

print("2. Chỉnh độ sáng")

print("3. Phóng to video")

print("0. Thoát")

a = int(input("Chọn chức năng:"))

if a == 0:

break

if a == 1:

print('Chạy chức năng chỉnh âm lượng...')

volumeControl()

if a == 2:

print('Chạy chức năng chỉnh độ sáng...')

brightnessControl()

if a == 3:

print('Chạy chức năng phóng to video...')

zoomVideo()

# Chuẩn bị camera, bàn tay và chức năng vẽ

def cameraAndHand():

# Camera

cap = cv2.VideoCapture(0)

cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH, 1080)

cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 720)

# Bàn tay

mpHands = mp.solutions.hands

hands = mpHands.Hands()

# Vẽ

mpDraw = mp.solutions.drawing\_utils

return cap, mpHands, hands, mpDraw

# Chỉnh âm lượng

def volumeControl():

cap, mpHands, hands, mpDraw = cameraAndHand()

# Truy cập vào loa của thiết bị thông qua pycaw

devices = AudioUtilities.GetSpeakers()

interface = devices.Activate(IAudioEndpointVolume.\_iid\_, CLSCTX\_ALL, None)

volume = cast(interface, POINTER(IAudioEndpointVolume))

# Vị trí 0% của thanh âm lượng

volbar = 400

# Phần trăm âm lượng

volper = 0

# Lấy ra giá trị thấp và cao nhất của âm lượng máy

volMin, volMax = volume.GetVolumeRange()[:2]

while True:

# Đọc ảnh chụp được từ camera

\_, img = cap.read()

img = cv2.flip(img, 1)

imgRGB = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

key = cv2.waitKey(1)

# Kiểm tra nút được ấn

mode = select\_mode(key, 0)

if mode == -1:

print('Thoát chỉnh âm lượng')

break

# Xử lý ảnh để nhận diện bàn tay

results = hands.process(imgRGB)

cv2.putText(img, f"Nhan ESC de thoat", (500, 30), cv2.FONT\_ITALIC, 1, (0, 255, 98), 3)

# Lưu trữ các vị trí trên bàn tay và vẽ đường nối

h, w, \_ = img.shape

lmList = []

if results.multi\_hand\_landmarks:

for handlandmark in results.multi\_hand\_landmarks:

for id, lm in enumerate(handlandmark.landmark):

cx, cy = int(lm.x \* w), int(lm.y \* h)

lmList.append([id, cx, cy]) # adding to the empty list 'lmList'

mpDraw.draw\_landmarks(img, handlandmark, mpHands.HAND\_CONNECTIONS)

# Xử lý chuyển đổi khoảng cách giữ 2 ngón tay ra âm lượng

if lmList != []:

x1, y1 = lmList[4][1], lmList[4][2] # ngón cái

x2, y2 = lmList[8][1], lmList[8][2] # ngón trỏ

cv2.circle(img, (x1, y1), 13, (255, 0, 0), cv2.FILLED) # Các thông số lần lượt (ảnh, vị trí x y, bán kính, rgb)

cv2.circle(img, (x2, y2), 13, (255, 0, 0), cv2.FILLED)

cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 3)

# Tính khoảng cách giữa 2 ngón tay bằng công thức tính cạnh huyền

length = hypot(x2 - x1, y2 - y1)

# Chuyển đổi từ khoảng cách giữa 2 ngón tay ra âm lượng, thanh âm lượng, phần trăm âm lượng

# bằng phương pháp nội suy tuyến tính

vol = np.interp(length, [30, 220], [volMin, volMax])

volbar = np.interp(length, [30, 220], [400, 150])

volper = np.interp(length, [30, 220], [0, 100])

print(vol, int(length))

# Đặt giá trị âm lượng vừa tính được cho thiết bị

volume.SetMasterVolumeLevel(vol, None)

# Vẽ thanh âm lượng

cv2.rectangle(img, (50, 150), (85, 400), (0, 0, 255),

4) # ảnh, vị trí bắt đầu, vị trí kết thúc, rgb, độ dày

cv2.rectangle(img, (50, int(volbar)), (85, 400), (0, 0, 255), cv2.FILLED)

cv2.putText(img, f"{int(volper)}%", (45, 130), cv2.FONT\_ITALIC, 1, (0, 255, 98), 3)

cv2.imshow('Chỉnh âm lượng', img)

cap.release()

cv2.destroyAllWindows()

# Kiểm tra nút được ấn

def select\_mode(key, mode):

if key == 27: # ESC

mode = -1

return mode

# Chỉnh độ sáng

def brightnessControl():

cap, mpHands, hands, Draw = cameraAndHand()

while True:

# Đọc ảnh chụp được từ camera

\_, img = cap.read()

img = cv2.flip(img, 1)

imgRGB = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

# Kiểm tra nút được ấn

key = cv2.waitKey(1)

mode = select\_mode(key, 0)

if mode == -1:

print('Thoát chỉnh độ sáng')

break

# Xử lý ảnh để nhận diện bàn tay

Process = hands.process(imgRGB)

cv2.putText(img, f"Nhan ESC de thoat", (500, 30), cv2.FONT\_ITALIC, 1, (0, 255, 98), 3)

# Lưu trữ các vị trí trên bàn tay và vẽ đường nối

height, width, \_ = img.shape

landmarkList = []

if Process.multi\_hand\_landmarks:

for handlm in Process.multi\_hand\_landmarks:

for \_id, landmarks in enumerate(handlm.landmark):

x, y = int(landmarks.x \* width), int(landmarks.y \* height)

landmarkList.append([\_id, x, y])

Draw.draw\_landmarks(img, handlm, mpHands.HAND\_CONNECTIONS)

# Xử lý chuyển đổi khoảng cách giữ 2 ngón tay ra độ sáng

if landmarkList != []:

# ngón cái

x1, y1 = landmarkList[4][1], landmarkList[4][2]

# ngón trỏ

x2, y2 = landmarkList[8][1], landmarkList[8][2]

cv2.circle(img, (x1, y1), 7, (0, 255, 0), cv2.FILLED)

cv2.circle(img, (x2, y2), 7, (0, 255, 0), cv2.FILLED)

cv2.line(img, (x1, y1), (x2, y2), (0, 255, 0), 3)

# Tính khoảng cách giữa 2 ngón tay bằng công thức tính cạnh huyền

L = hypot(x2 - x1, y2 - y1)

# Chuyển đổi từ khoảng cách giữa 2 ngón tay ra độ sáng, thanh độ sáng

# bằng phương pháp nội suy tuyến tính

b\_level = np.interp(L, [30, 220], [0, 100])

b\_level\_bar = np.interp(L, [30, 220], [400, 150])

print(b\_level)

sbc.set\_brightness(int(b\_level), display=0)

cv2.rectangle(img, (50, 150), (85, 400), (0, 0, 255),

4) # ảnh, vị trí bắt đầu, vị trí kết thúc, rgb, độ dày

cv2.rectangle(img, (50, int(b\_level\_bar)), (85, 400), (0, 0, 255), cv2.FILLED)

cv2.putText(img, f"{int(b\_level)}%", (45, 130), cv2.FONT\_ITALIC, 1, (0, 255, 98), 3)

cv2.imshow('Chỉnh độ sáng', img)

cap.release()

cv2.destroyAllWindows()

# Phóng to video

def zoomVideo():

cap, mpHands, hands, mpDraw = cameraAndHand()

scale = 1

while True:

# Đọc ảnh chụp được từ camera

\_, img = cap.read()

img = cv2.flip(img, 1)

imgRGB = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

# Đọc ảnh chụp được từ camera

key = cv2.waitKey(1)

mode = select\_mode(key, 0)

if mode == -1:

print('Thoát phóng to video')

break

# Xử lý ảnh để nhận diện bàn tay

results = hands.process(imgRGB)

cv2.putText(img, f"Nhan ESC de thoat", (500, 30), cv2.FONT\_ITALIC, 1, (0, 255, 98), 3)

# Lưu trữ các vị trí trên bàn tay và vẽ đường nối

h, w, \_ = img.shape

lmList = []

if results.multi\_hand\_landmarks:

for handlandmark in results.multi\_hand\_landmarks:

for id, lm in enumerate(handlandmark.landmark):

h, w, \_ = img.shape

cx, cy = int(lm.x \* w), int(lm.y \* h)

lmList.append([id, cx, cy])

mpDraw.draw\_landmarks(img, handlandmark, mpHands.HAND\_CONNECTIONS)

# Xử lý chuyển đổi khoảng cách giữ 2 ngón tay ra tỉ lệ phóng to

# Tỉ lệ phóng to ảnh từ 0.01 -> 1

if lmList != []:

x1, y1 = lmList[4][1], lmList[4][2] # Ngón cái

x2, y2 = lmList[8][1], lmList[8][2] # Ngón trỏ

cv2.circle(img, (x1, y1), 13, (255, 0, 0), cv2.FILLED)

cv2.circle(img, (x2, y2), 13, (255, 0, 0), cv2.FILLED)

cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 3)

# Tính cạnh huyền

length = np.hypot(x2 - x1, y2 - y1)

# Nội suy tuyến tính

scale = np.interp(length, [30, 220], [1, 0.01]) # Tỉ lệ

scaleBar = np.interp(length, [30, 220], [400, 150]) # Thanh tỉ lệ

scalePer = np.interp(length, [30, 220], [0, 100]) # Phần trăm

print(scale)

cv2.rectangle(img, (50, 150), (85, 400), (0, 0, 255), 4)

cv2.rectangle(img, (50, int(scaleBar)), (85, 400), (0, 0, 255), cv2.FILLED)

cv2.putText(img, f"{int(scalePer)}%", (45, 130), cv2.FONT\_ITALIC, 1, (0, 255, 98), 3)

# Thuật toán xử lý phóng to hình ảnh

centerX, centerY = int(h / 2), int(w / 2)

radiusX, radiusY = int(scale \* centerX), int(scale \* centerY)

minX, maxX = centerX - radiusX, centerX + radiusX

minY, maxY = centerY - radiusY, centerY + radiusY

cropped = img[minX:maxX, minY:maxY]

rsCropped = cv2.resize(cropped, (w, h))

cv2.imshow('Phóng to video', rsCropped)

main()

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

...........................................................................................................................................

*Thái Nguyên, ngày tháng năm 2023*

**Giáo viên hướng dẫn**