**Chương 2 btl**

————-

**1.Giới thiệu về bài toán dò tìm khuôn mặt**

Bài toán dò tìm khuôn mặt (face detection) là một trong những bài toán quan trọng trong lĩnh vực thị giác máy tính và trí tuệ nhân tạo. Nó đặt ra mục tiêu nhận diện và phát hiện khuôn mặt trong một hình ảnh hoặc một đoạn video.

Bài toán này thường được giải quyết thông qua việc sử dụng các thuật toán học máy và deep learning để học và tìm kiếm các đặc trưng của khuôn mặt, ví dụ như hình dạng, màu sắc, kích thước và đặc điểm hình học khác.

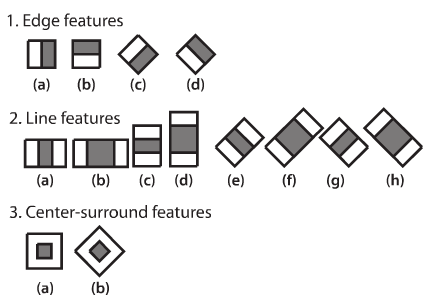
Các ứng dụng của bài toán dò tìm khuôn mặt rất đa dạng, từ nhận dạng khuôn mặt để mở khóa điện thoại, nhận diện khuôn mặt để đăng nhập vào các ứng dụng, đến giám sát an ninh và phát hiện tội phạm. Tuy nhiên, bài toán này còn đối mặt với nhiều thách thức như sự biến đổi về ánh sáng, góc nhìn và vật liệu, cũng như sự đa dạng của các khuôn mặt trong thực tế.

——————

**2.đặc trưng Haar-Like Features**

Đặc trưng Haar-Like Features là một trong những phương pháp được sử dụng trong bài toán dò tìm khuôn mặt. Phương pháp này được đề xuất bởi Viola và Jones vào năm 2001 và đã trở thành một trong những phương pháp phổ biến trong lĩnh vực nhận dạng khuôn mặt.

Trong bài báo của tác giả *Paul Viola và Michael Jones* đã trình bày một phương pháp mới và nhanh hơn để xử lý hình ảnh và phát hiện khuôn mặt bằng cách sử dụng các đặc điểm hình chữ nhật như hình dưới đây. Các **đặc điểm hình chữ nhật** tương tự như nhân được sử dụng để phát hiện các đặc điểm khác nhau của khuôn mặt như mắt và các nốt như trong hình minh họa.



Hình minh họa đặc trưng

Các tính năng hình chữ nhật được chạy lần lượt trên hình ảnh và tổng số pixel nằm trong phần màu trắng được trừ cho tổng số pixel nằm trong phần màu đen.

Trong đó:

Edge features: Đặc trưng cạnh

Line features: Đặc trưng đường

Center-suround features: Đặc trưng trung tâm

Trong hình minh họa bên dưới, đặc điểm hình chữ nhật đầu tiên đang tính toán cho sự khác biệt về cường độ giữa vùng mắt và vùng má trên khuôn mặt. Và đặc điểm hình chữ nhật thứ hai là đo sự chênh lệchvề cường độ giữa hai vùng mắt và sống mũi. Bộ lọc Haar chỉ có thể nhìn cụ thể vào một vùng trong cửa sổ để nhận diện.



Ví dụ

————-

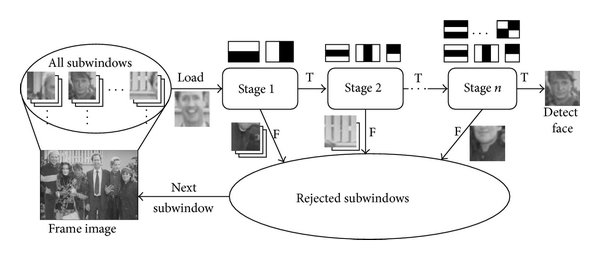
**3.Thuật toán AdaBoost và bộ phân tầng Cascade of Boosted Classifier**

Thuật toán AdaBoost (Adaptive Boosting) là một trong những thuật toán quan trọng và phổ biến nhất được sử dụng trong bài toán dò tìm khuôn mặt. AdaBoost kết hợp nhiều bộ phân loại yếu thành một bộ phân loại mạnh. Mỗi bộ phân loại yếu được huấn luyện để phân loại một tập con của dữ liệu và nhận được một trọng số ứng với độ chính xác của nó. AdaBoost sẽ tăng trọng số của các dữ liệu bị phân loại sai và giảm trọng số của các dữ liệu được phân loại đúng. Sau đó, các bộ phân loại yếu sẽ được sử dụng để tạo ra một bộ phân loại mạnh bằng cách kết hợp dự đoán của chúng theo một quy tắc đa số.

Bộ phân tầng Cascade of Boosted Classifier là một phương pháp dò tìm khuôn mặt cải tiến dựa trên AdaBoost. Nó bao gồm nhiều giai đoạn, mỗi giai đoạn sử dụng một bộ phân loại được huấn luyện để loại bỏ các vùng không phải là khuôn mặt. Ở mỗi giai đoạn, hình ảnh được chia thành nhiều vùng nhỏ hơn, các vùng này được xem xét bởi các bộ phân loại với độ chính xác khác nhau. Các vùng bị loại bỏ không phải là khuôn mặt sẽ không được xem xét trong các giai đoạn tiếp theo. Các khuôn mặt được phát hiện trong các giai đoạn sau sẽ được xác định chính xác hơn vì chỉ cần xem xét các vùng nhỏ hơn và đặt ưu tiên cho các vùng có khả năng cao hơn chứ không cần phân tích toàn bộ hình ảnh.

Bộ phân tầng Cascade of Boosted Classifier có nhiều ưu điểm, bao gồm tăng tốc độ dò tìm, cải thiện độ chính xác và giảm độ phức tạp tính toán. Tuy nhiên, cần phải cân nhắc kỹ lưỡng trong việc lựa chọn các tham số và bộ phân loại để đạt được hiệu quả cao nhất.

Để được nhận dạng là khuôn mặt, các cửa sổ ảnh nhỏ sẽ phải đi qua hết các stage của bộ phân loại. Các bộ lọc ở mỗi stage đằng sau sẽ được huấn luyện bằng các mẫu không có đối tượng mà các stage trước nó nhận định sai, do đó sự kết hợp các lớp lọc này sẽ giúp hệ thống giảm thiếu khả năng nhận định sai.



Bộ phân tầng Haar Cascade

* **Bước 1:** Hình ảnh (đã được gửi đến bộ phân loại) được **chia thành các phần nhỏ**(hoặc các cửa sổ con như trong hình minh họa).
* **Bước 2:** Chúng tôi đặt N bộ dò theo cách xếp tầng trong đó mỗi bộ phát hiện sự kết hợp của các loại **đặc trưng khác nhau** từ các hình ảnh (ví dụ: đường thẳng, cạnh, hình tròn, hình vuông) được truyền qua. Giả sử khi việc trích xuất đối tượng địa lý được thực hiện, mỗi phần phụ được gán một **giá trị tin cậy**.
* **Bước 3:** Hình ảnh (hoặc hình ảnh phụ) có độ tin cậy cao nhất được phát hiện dưới dạng khuôn mặt và được gửi đến **bộ tích lũy** trong khi phần còn lại bị từ chối. Do đó, Cascade tìm nạp khung hình / hình ảnh tiếp theo nếu còn lại và bắt đầu lại quá trình.

——————-

**4 Tổng quan về gói thư viện OpenCV để nghiên cứu về nhận dạng mặt người,**

OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở cung cấp nhiều công cụ và thuật toán để xử lý ảnh và thị giác máy tính. OpenCV được viết bằng C++, nhưng cũng có thể sử dụng các ngôn ngữ khác như Python và Java thông qua các giao diện lập trình ứng dụng (API).

OpenCV cung cấp nhiều tính năng để hỗ trợ nhận dạng mặt người, bao gồm:

1. Xử lý ảnh: OpenCV cung cấp các tính năng để xử lý ảnh như đọc, ghi, chuyển đổi, cắt, phóng to, thu nhỏ ảnh, làm mịn ảnh, tăng độ tương phản và cân bằng sáng tối.
2. Dò tìm khuôn mặt: OpenCV cung cấp các thuật toán để dò tìm khuôn mặt như Haar Cascades, LBP (Local Binary Pattern), và HOG (Histogram of Oriented Gradients).
3. Nhận dạng khuôn mặt: OpenCV cung cấp các thuật toán để nhận dạng khuôn mặt như Eigenfaces, Fisherfaces, và Local Binary Patterns Histograms (LBPH).
4. Ghi chú và vẽ hình trên ảnh: OpenCV cung cấp các công cụ để ghi chú và vẽ hình trên ảnh để đánh dấu khuôn mặt đã được tìm thấy hoặc hiển thị kết quả của thuật toán.

OpenCV cũng có thể được sử dụng để kết hợp nhiều thuật toán nhận dạng khuôn mặt và tạo ra một hệ thống hoàn chỉnh để nhận dạng khuôn mặt trong thời gian thực.

Việc sử dụng OpenCV để nghiên cứu về nhận dạng khuôn mặt là rất phổ biến vì thư viện này cung cấp nhiều tính năng và thuật toán cho bài toán này và có thể được sử dụng trên nhiều nền tảng khác nhau như Windows, Linux, MacOS, Android và iOS.

————

**thư viện Emgu nền tảng của OpenCV trên nền ngôn ngữ C# và hệ thống gửi tin nhắn GSMComm.**

Thư viện Emgu CV là một thư viện mã nguồn mở dùng để xử lý ảnh và thị giác máy tính trên nền tảng của OpenCV nhưng được phát triển trên nền tảng của ngôn ngữ lập trình C#. Thư viện này giúp cho các lập trình viên .NET có thể sử dụng các tính năng của OpenCV một cách dễ dàng hơn thông qua các API được đóng gói sẵn.

Emgu CV hỗ trợ các tính năng như:

* Xử lý ảnh cơ bản như đọc, ghi, chuyển đổi, cắt, phóng to, thu nhỏ ảnh.
* Dò tìm đối tượng trên ảnh như khuôn mặt, mắt, mũi, miệng, xe, người.
* Phân loại đối tượng trên ảnh như khuôn mặt, chữ số, ký tự.
* Trích xuất đặc trưng trên ảnh như SIFT, SURF, HOG, LBP.
* Nhận dạng khuôn mặt và đối tượng trên ảnh.

Hơn nữa, thư viện Emgu CV có tích hợp sẵn với Visual Studio và hỗ trợ nhiều phiên bản của .NET Framework, từ 2.0 đến 4.5. Nó cũng có thể được sử dụng trên các nền tảng khác nhau như Windows, Linux và MacOS.

Ngoài ra, để kết hợp với thư viện Emgu CV, bạn cũng có thể sử dụng hệ thống gửi tin nhắn GSMComm. GSMComm là một thư viện mã nguồn mở được phát triển trên nền tảng .NET Framework, cho phép gửi và nhận tin nhắn SMS bằng một modem GSM thông qua cổng COM hoặc USB.

Với việc kết hợp sử dụng Emgu CV và GSMComm, bạn có thể thiết kế và triển khai các ứng dụng nhận dạng khuôn mặt và gửi tin nhắn SMS tự động, ví dụ như hệ thống chấm công, kiểm soát ra vào.

1. **Image Kernels**

Image Kernels là 1 ma trận nhỏ dùng để sử dụng các hiệu ứng chỉnh sửa. Trong mỗi ô của ma trận sẽ chứa giá trị tờ 0 đến 255 để biểu thị giá trị sang tối của từng pixel ảnh trong ảnh chính. Từ đó, người dùng sẽ áp dụng các công thức tính toán để làm mờ, làm rõ nét bằng các phép nhân ma trận. Hơn thế nữa, chúng ta có thể dùng phép tích phân ảnh để sử dụng trong việc nhận dạng khuôn mặt.

1. **Trình tự hoạt động**

Sau khi quét ảnh từ camera, hệ thống sẽ chuyển hình ảnh quét được thành ảnh đen trắng. Sau đó chuyển hình ảnh đen trắng đã có thành dạng Image Kernels, hệ thống trích các đặc trưng haar-like từ ảnh và sử dụng các hình chữ nhật đặc trưng để phát hiện gương mặt trong khung hình.

1. Trích đặc trưng Harr-Like từ ảnhĐặc trưng Haar-like sẽ diễn đạt được tri thức về các đối tượng trong ảnh (bởi vì

nó biểu diễn mối liên hệ giữa các bộ phận của đối tượng), điều mà bản thân từng điểm

ảnh không diễn đạt được – dựa trên việc co giãn của khung cửa sổ tìm kiếm. Để tính giá

trị các đặc trưng Haar-Like, ta tính sự chênh lệch giữa tổng của các pixel của các vùng

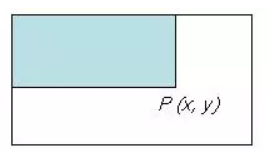
đen và các vùng trắng như trong công thức:

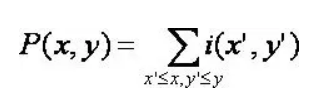
****

Vì vậy chúng ta sẽ thấy rằng, với tổng các pixel trên ảnh *(pixel vùng trắng và  
pixel vùng đen)* sẽ cho ta các giá trị của đặc trưng Haar – Like. Nhưng để tính toán những giá trị của các đặc trưng Haar-like cho tất cả các vị trí trên ảnh đòi hỏi chi phí tính toán khá lớn, dẫn đến không thể đáp ứng được cho các ứng dụng đòi hỏi tính thời gian thực*(run-time)* trực tiếp từ Camera *(hay Webcam trên Laptop)*. Do đó Viola và Jones đề xuất một khái niệm gọi là ***“tich phân ảnh”*** *(Integral Image)* để tính toán nhanh cho các đặc trưng cơ bản của Haar-like.

Tích phân ảnh được định nghĩa là 1 mảng 2 chiều có kích thước bằng kích thước ảnh cần tìm.

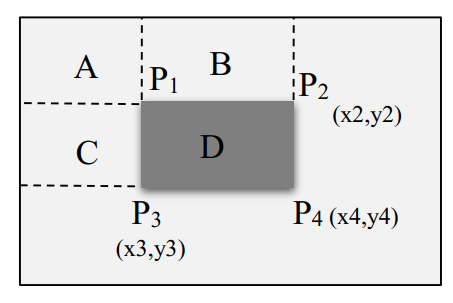
Mỗi phần tử trong mảng có giá trị bằng tổng các phần từ ở phái trên và bên trái của nó. Bắt đầu từ vị trí trên bên trái và kết thúc tại vị trí dưới bên phải của ảnh, tính lần lượt từ trái qua phải và từ trên xuống dưới.





Công thức tính tích phân ảnh

Sau khi tính tích phân ảnh, ta tính tổng điểm ảnh theo các sau:



Tính nhanh tổng điểm ảnh trong vùng D

Tính điểm D:  
▪Tổng = A + B + C + D  
▪D =Tổng – (A+B) – (A+C) + A

Với A + B + C + D chính là giá trị tại điểm P4 trên đảo hàm ảnh , tương tự như  
vậy A+B là giá trị tại điểm P2, A+C là giá trị tại điểm P3, và A là giá trị tại điểm P1.  
Vậy ta có thể viết lại biểu thức tính D ở trên như sau:

D= P4-P2-P3+P1

D= (x4,y4)-(x2,y2)-(x3,y3)+(x1,y1)