**ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**LẬP TRÌNH HỆ THỐNG VÀ VI ĐIỀU KHIỂN**

**ĐỀ TÀI: HỆ THỐNG CẢNH BÁO TÀI XẾ NGỦ GẬT VÀ KIỂM SOÁT NỒNG ĐỘ CỒN**

GIẢNG VIÊN HƯỚNG DẪN: TS.PHẠM CÔNG THẮNG

GIẢNG VIÊN CHẤM: TS. NINH KHÁNH DUY

|  |  |  |  |
| --- | --- | --- | --- |
| STT NHÓM:  HỌ VÀ TÊN SINH VIÊN | LỚP HỌC PHẦN ĐỒ ÁN | ĐIỂM QUÁ TRÌNH (do GVHD ghi) | ĐIỂM BẢO VỆ (do GV chấm ghi) |
| Nguyễn Xuân Tuấn | **18N10B** |  |  |
| Đinh Trọng Hùng | **18N10B** |  |  |
| Nguyễn Văn Hữu Phước | **18N10A** |  |  |
| Lê Đức Thắng | **18N10A** |  |  |

**ĐÀ NẴNG, 07/2021**

**TÓM TẮT ĐỒ ÁN**

“Hệ thống cảnh báo tài xế ngủ gật và kiểm soát nồng độ cồn” nhằm giải quyết vấn đề : Giúp lái xe đảm bảo an toàn cho bản thân, phương tiện và những người cùng tham gia giao thông. Phương pháp giải quyết là sử dụng công nghệ nhận diện khuôn mặt với thư viện OpenCV để nhận dạng khuôn mặt, tính tỷ lệ độ mở mắt để cảnh báo người lái xe buồn ngủ khi đang tham gia giao thông, đồng thời dùng cảm biến đo nồng độ cồn để kiếm soát nồng độ cồn của lái xe, nếu vượt quá quy định thì sẽ cảnh báo và dùng module sim để nhắn tin về cho người thân.

**LỜI NÓI ĐẦU**

Ngày nay, ngành khoa học kĩ thuật ngày càng phát triển, cùng với đó là nhu cầu đòi hỏi của con người ngày một nâng cao trong nhiều lĩnh vực của đời sống xã hội. Trong đó có nhu cầu về một cuộc sống tiện nghi, thông minh. Hiện tại, công nghệ thông tin được người ta quan tâm và nhắc đến nhiều hơn bao giờ hết vì nó là một phần không thể thiếu trong cuộc sống văn minh, tri thức. Máy vi tính, các thiết bị di động cùng với những phần mềm là công cụ đắc lực giúp ta quản lý, tổ chức, sắp xếp, điều khiển và xử lý công việc một cách nhanh chóng và chính xác. Là một sinh viên ngành Công nghệ thông tin, chúng em có cơ hội được nghiên cứu sâu hơn và đi vào các bước tạo ra sản phẩm thông qua **Đồ án Lập trình hệ thống và Vi điều khiển.**

Thông qua quá trình thực hiện đồ án, chúng em đã nắm bắt được những kỹ thuật quan trọng của việc xây dựng mã nguồn, lắp ráp và hoàn thiện sản phẩm. Chúng em xin chân thành cảm ơn thầy Phạm Công Thắng đã bổ sung ý tưởng và tận tình giúp đỡ chúng em thực hiện đồ án này. Trong quá trình thực hiện, không tránh khỏi sai sót. Chúng em rất mong nhận được sự góp ý từ phía thầy cô để sản phẩm của chúng em được hoàn thiện hơn.

**MỤC LỤC**

[I. GIỚI THIỆU: 5](#_Toc42704237)

[II. GIẢI PHÁP : 7](#_Toc42704238)

[1. Mô tả hệ thống : 7](#_Toc42704239)

[2. Phát hiện khuôn mặt sử dụng đặc trưng Haar-like : 8](#_Toc42820226)

2.1. Các đặc trưng Haar-like : 8

2.2. Áp dụng đặc trưng Haar-like để phát hiện khuôn mặt trong ảnh: 9

2.3. Ảnh tích hợp (Integral image) : 10

3. Thuật toán Adaboost (Adaptive Boost): 11

[4. Mô hình phân lớp phân tầng (Cascade classifier) :](#_Toc42704241) 12

[5. Phát hiện khuôn mặt sử dụng thư viện OpenCV :](#_Toc42704241) 13

[6. Facial landmark :](#_Toc42704241) 14

[7. Hệ thống cảnh báo nồng độ cồn :](#_Toc42704241) 18

[8. Sơ đồ mạch của hệ thống :](#_Toc42704241) 19

[9. Giới thiệu một số phần cứng, linh kiện sử dụng :](#_Toc42704241) 19

[III. KẾT QUẢ :](#_Toc42704250) 22

[1. Mô tả dữ liệu kiểm thử :](#_Toc42704241) 22

[2. Quy trình kiểm thử :](#_Toc42704241) 22

IV. KẾT LUẬN : 23

1. Ưu và nhược điểm : 23

2. Kết luận : 23

3. Hướng phát triển : 23

[TÀI LIỆU THAM KHẢO 24](#_Toc42704251)

1. GIỚI THIỆU:

Tên đề tài : Hệ thống cảnh báo tài xế ngủ gật và kiểm soát nồng độ cồn

* Đặt vấn đề :

Hiện nay, ở nước ta các tệ nạn xã hội ngày càng gia tăng mà một trong những nguyên nhân chủ yếu đó là do uống nhiều rượu bia. Rượu là nguyên nhân chủ yếu làm giảm năng suất lao động, gây ra các tệ nạn xã hội như bạo lực, gia đình tan vỡ, con cái hư hỏng và đặc biệt gây ra các vụ tai nạn giao thông nghiêm trọng và đặc biệt nghiêm trọng.

Việt Nam là quốc gia thuộc top đầu các nước sử dụng rượu, bia khi tham gia giao thông. Tình trạng sử dụng rượu, bia tràn lan ở nhiều nơi đã khiến trật tự an toàn giao thông trở thành vấn đề báo động đây chính là nguyên nhân hàng đầu gây tai nạn giao thông, khi điều khiển phương tiện bởi người lái xe thường phản ứng chậm, buồn ngủ, thiếu tập trung, việc nhìn thấy các biển báo, tín hiệu hoặc quan sát trên đường không còn rõ ràng nên có khi xử lý không đúng và có thể gây ra tai nạn Người say cũng "bốc đồng", không còn khả năng kiểm soát tốc độ cho nên thường phóng nhanh, vượt ẩu, lấn đường rất dễ gây tai nạn.

Lái xe khi buồn ngủ là một vấn đề nghiêm trọng. Cứ 3 người thì có 1 người ngủ gật khi lái xe. Tai nạn giao thông do ngủ gật nhiều khả năng là nguyên nhân chính và có thể gây chết người. Khoảng 20% số vụ tai nạn giao thông chết người là lái xe khi buồn ngủ.

* Các vấn đề cần giải quyết :

Hệ thống cảnh báo tài xế ngủ gật giúp cảnh báo tài xế tránh được tai nạn ngoài mong muốn nên đòi hỏi tốc độ xử lý và độ chính xác cao. Thuật toán nhận dạng khuôn mặt cần được tối ưu nhất để cảnh báo cho tài xế một cách nhanh nhất, hạn chế tối đa sự nhầm lẫn

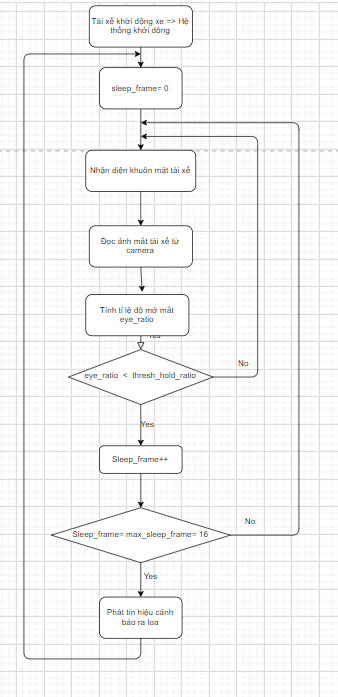
Hệ thống cảnh báo nồng độ cồn giúp phát hiện nồng độ cồn giúp cảnh báo cho người thân của tài xế biết được để có những biện pháp hỗ trợ kịp thời.

**BẢNG PHÂN CÔNG NHIỆM VỤ**

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **Nhiệm vụ** | Tự đánh giá theo 3 mức  (Đã hoàn thành/Chưa hoàn thành/Không triển khai) |
| Nguyễn Xuân Tuấn | Tìm hiểu về thuật toán nhận dạng khuôn mặt sử dụng thư viện OpenCV, lập trình về nhận diện khuôn mặt và hàm tính tỷ lệ khung hình của mắt, viết báo cáo | dfdddfff  Đã hoàn thành  f |
| Đinh Trọng Hùng | Tìm hiểu về cảm biến nồng độ cồn, về module sim 800A, lập trình về thuật toán phát hiện nồng độ cồn, dùng module sim để gửi tin nhắn | Đã hoàn thành |
| Nguyễn Văn Hữu Phước | Tìm hiểu về thuật toán Adaboost, mô hình phân lớp phân tầng, làm slide thuyết trình | Đã hoàn thành |
| Lê Đức Thắng | Tìm hiểu lý thuyết về Raspberry Pi 3, các đặc trưng Haar-like  Tìm hiểu về thuật toán phát hiện khuôn mặt sử dụng thư viện OpenCV | Đã hoàn thành |

1. GIẢI PHÁP :
2. Mô tả hệ thống :

Hệ thống gồm Raspberry Pi 3 kết nối với loa và chạy phần mềm cảnh báo ngủ gật trên Pi. Phần mềm trên Pi sẽ theo dõi đôi mắt của tài xế thường xuyên. Nếu như tài xế nhắm mắt ngủ trong 1 khoảng thời gian nhất định do chúng ta quy định thì âm thanh chói tai sẽ phát ra từ hệ thống loa để cảnh báo làm giật mình tài xế hoặc khuyên tài xế nên dừng xe để nghỉ ngơi tránh các tai nạn đáng tiếc. Cùng với đó là hệ thống cảnh báo nồng độ cồn, khi phát hiện nống độ cồn của tài xế quá mức quy định thì sẽ phát ra cảnh báo đồng thời nhắn tin về cho người thân để có những biện pháp xử lý kịp thời.



Hình 1: Sơ đồ hệ thống cảnh báo tài xế ngủ gật

1. **Phát hiện khuôn mặt sử dụng đặc trưng Haar-like :**

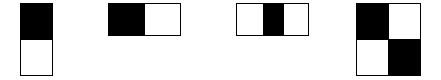
Vùng khuôn mặt trên ảnh là tập hợp các điểm ảnh có khác biệt so với các vùng ảnh khác, tạo nên các đặc trưng riêng của khuôn mặt. Tất cả khuôn mặt người đều có chung những đặc điểm sau khi đã chuyển qua ảnh xám, ví dụ:

* Vùng hai mắt sẽ tối hơn vùng má và vùng trán
* Vùng giữa sống mũi cũng tối hơn vùng hai bên mũi.

Và còn rất nhiều những đặc điểm khác của khuôn mặt và các đặc trưng Haar like dựa vào các đặc điểm này để nhận dạng.

* 1. **Các đặc trưng Haar-like:**

Mỗi đặc trưng Haar-like là một miền hình chữ nhật được chia thành 2,3 hoặc 4 hình chữ nhật nhỏ phân biệt quy ước bằng màu trắng và đen.



Hình 2: Các đặc trưng Haar-like [8]

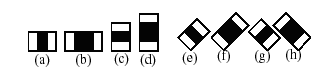
Như hình vẽ trên [8] là 3 loại đặc trưng Haar-like cơ bản gồm 2,3 và 4 hình chữ nhật hợp thành. Từ những đặc trưng cơ bản mở rộng ra thành tập các đặc trưng:

* + Đặc trưng cạnh [8]:



Hình 3: Đặc trưng cạnh [8]

* + Đặc trưng đường [8]:



Hình 4: Đặc trưng đường [8]

* + Đặc trưng tâm – xung quanh [8]:



Hình 5: Đặc trưng tâm - xung quanh [8]

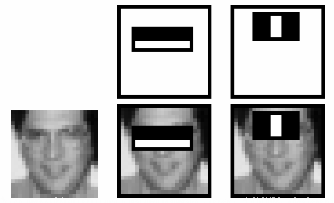
Giá trị của một đặc trưng Haar-like là sự khác biệt giữa tổng các giá trị xám của các pixel trong vùng “đen” với tổng các giá trị xám của các pixel trong vùng “trắng”:

*f(x) = Tổngvùng đen(pixel) - Tổngvùng trắng(pixel)*

Vậy khi được đặt lên một vùng ảnh, đặc trưng Haar-like sẽ tính toán và đưa ra giá trị đặc trưng *h(x)* của vùng ảnh đó.

* 1. **Áp dụng đặc trưng Haar-like để phát hiện khuôn mặt trong ảnh :**

Để phát hiện khuôn mặt, hệ thống sẽ cho một cửa sổ con quét lên toàn bộ ảnh đầu vào. Như vậy sẽ có rất nhiều ảnh con ứng với từng cửa sổ con, các đặc trưng Haar-like sẽ được đặc lên các cửa sổ con này để từ đó tính ra giá trị của đặc trưng. Sau đó các giá trị này được bộ phân loại xác nhận xem khung hình đó có phải khuôn mặt hay không.



Hình 6. Áp dụng đặc trưng Haar-like lên từng khung của ảnh [9]

Ứng với mỗi một đặc trưng như trên, một bộ phân lớp yếu xác định khi giá trị của đặc trưng Haar-like tại cửa sổ con vượt qua một ngưỡng đã xác định thì bộ phân lớp sẽ kết luận cửa sổ con là khuôn mặt, cònkhông vượt qua ngưỡng đó thì không là khuôn mặt.

* 1. **Ảnh tích hợp (Integral Image) :**

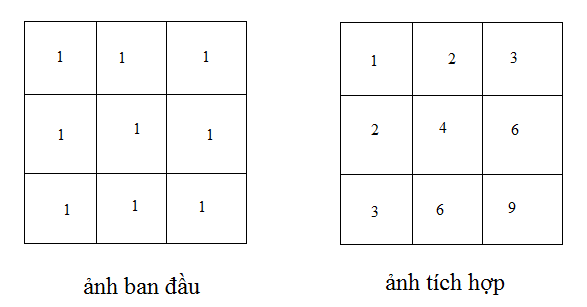
Giá trị của mỗi một đặc trưng Haar-like được tính bằng tổng giá trị các điểm ảnh vùng đen trừ đi tổng các điểm ảnh vùng trắng, như vậy là với số lượng lớn các đặc trưng sẽ tạo ra một khối lượng tính toán khổng lồ. Điều này là không phù hợp để đáp ứng thời gian thực vì thời gian xử lý rất lâu. Vì vậy ảnh tích hợp được đưa ra nhằm tính toán nhanh chóng các đặc trưng, giảm thời gian xử lý.

Ảnh tích hợp được định nghĩa theo công thức:



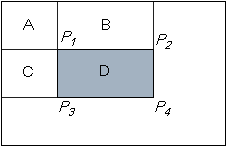
Giá trị của ảnh tích hợp tại vị trí (x,y) là tổng các điểm ảnh thuộc hình chữ nhật xác định bởi góc trái trên là (0,0) và góc phải dưới (x,y). Sau khi chuyển ảnh cần nhận dạng thành ảnh tích hợp, viêc tính toán giá trị các đặc trưng Haar-like sẽ rất đơn giản.

Ví dụ chuyển một ảnh 33 có giá trị xám như bên dưới thành ảnh tích hợp:



Hình 7. Chuyển ảnh ban đầu về ảnh tích hợp [2]

Sau khi chuyển về ảnh tích hợp, ta tiếp tục tính giá trị đặc trưng Haar-like, ta phải tính được tổng giá trị điểm ảnh trong một vùng hình chữ nhật trên ảnh. Ví dụ như vùng D trong hình vẽ:



Hình 8. Tính giá trị các điểm ảnh tại vùng D [10]

Với : A,B,C,D là tổng giá trị các điểm ảnh trong từng vùng.

P1,P2,P3,P4 là giá trị của ảnh tích hợp tại 4 đỉnh của D.

Nếu như là ảnh xám bình thường thì để tính D ta phải tính tổng tất cả các giá trị điểm ảnh trong D, miền D càng lớn thì số phép cộng càng nhiều. Nhưng với ảnh tích hợp dù miền D có kích thước như thế nào thì D cũng chỉ cần tính thông qua 4 giá trị tại 4 đỉnh.

Ta có: P1 = A, P2 = A + B, P3 = A + C, P4 = A + B + C + D

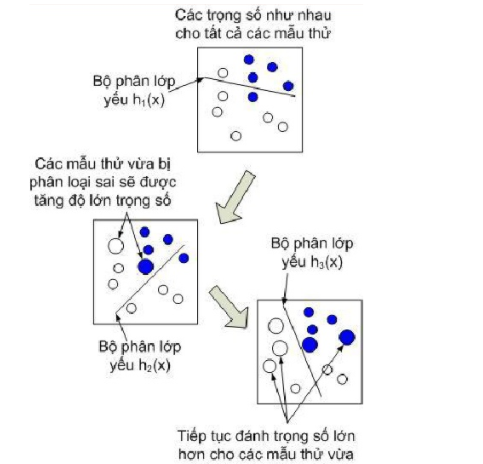
P1+ P4–P2–P3= A + (A + B + C + D) – (A + B) – (A + C) = D

Vậy ta có: **D = P1 + P4 – P2 – P3.**

1. **Thuật toán Adaboost (Adaptive Boost):**

Kỹ thuật Boosting: nguyên tắc cơ bản của Boosting là kết hợp các bộ phân lớp yếu (hay các bộ phân lớp cơ sở) để tạo nên một bộ phân lớp mạnh.

Adaboost là một thuật toán sử dụng kỹ thuật Boosting được đưa ra bởi Freund và Schapire vào 1996. Thuật toán này được sử dụng rất phổ biến cho bài toán phát hiện khuôn mặt. Điểm cải tiến của Adaboost là ta sẽ gán cho mỗi mẫu một trọng số. Ý nghĩa của việc gán trọng số như sau: Ở mỗi vòng lặp của quá trình huấn luyện, khi một bộ phân lớp yếu đã được xây dựng, ta sẽ tiến hành cập nhật trọng số cho các mẫu. Việc cập nhật này được tiến hành như sau: ta sẽ tăng trọng số của các mẫu bị phân lớp sai bởi bộ phân lớp yếu và giảm trọng số của các mẫu được phân lớp đúng. Bằng cách này, ở vòng lặp kế, ta sẽ xây dựng bộ phân lớp yếu tiếp theo theo hướng: tập trung vào các mẫu bị phân lớp sai bởi bộ phân lớp yếu trước đó. Cuối cùng, để có được bộ phân lớp mạnh, ta sẽ kết hợp tuyến tính các bộ phân lớp yếu đã tìm được lại với nhau.

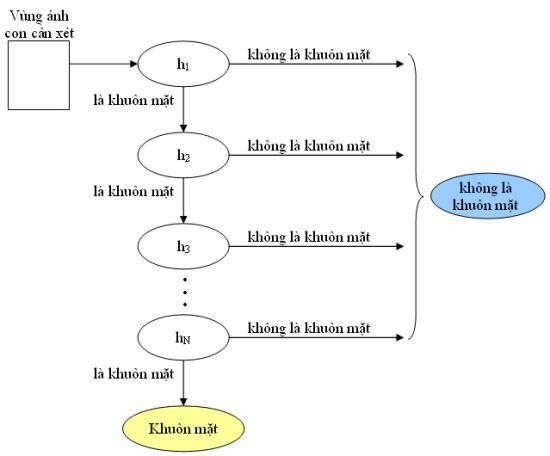


Hình 9: Áp dụng thuật toán Adaboost [8]

1. **Mô hình phân lớp phân tầng (Cascade classifier):**

Cascade classifier là một chuỗi phân lớp trong đó mỗi bộ phân lớp được xây dựng bằng thuật toán Adaboost.

Bộ phân lớp đầu tiên sẽ loại bỏ phần lớn các cửa sổ không phải khuôn mặt và cho đi qua các cửa sổ được cho là khuôn mặt. Ở đây, bộ phân lớp này rất đơn giản nên độ phức tạp tính toán cũng rất thấp. Bởi vì bộ phân lớp đầu tiên khá đơn giản nên trong số các cửa sổ được nhận dạng là khuôn mặt sẽ có một số khung lớn cửa sổ bị nhận dạng sai (không phải là khuôn mặt).



Hình 10: Thuật toán phát hiện khuôn mặt [10]

Những cửa chỗ được cho đi qua bởi bộ phân lớp đầu sẽ được xem xét bởi bộ phân lớp sau đó, nếu bộ phân lớp cho rằng đó không phải là khuôn mặt thì ta loại bỏ, nếu bộ phân lớp cho rằng đó là khuôn mặt thì ta lại cho đi qua và chuyển đến bộ phân lớp phía sau. Chỉ những cửa sổ đi qua được tất cả các bộ phân lớp thì ta mới quyết định đó là khuôn mặt. Sau khi đã xác định tất cả các khung có thể là khuôn mặt thì bước cuối cùng sẽ gộp các khung hình lân cận đó thành một khung mặt chứa khuôn mặt người duy nhất.

1. **Phát hiện khuôn mặt sử dụng thư viện OpenCV :**

Môhình phân tầng nhóm em sử dụng là một bộ phân lớp được xây dựng sẵn trên các bộ cơ sở dữ liệu chuẩn của thư viện OpenCV. Bộ phân lớp này được lưu ở file xml có tên haarcascade\_frontalface\_default.xml

Bên cạnh bộ phân lớp ở trên, một hàm quan trọng phục vụ công việc phát hiện khuôn mặt là hàm **detectMultiScale(image,scale\_factor,min\_neighbors)** thuộc lớp CascadeClassifier (lớp phục vụ tìm kiếm đối tượng của Opencv).

Các tham số đầu vào:

* **image** là ảnh cần phát hiện khuôn mặt.
* **scale\_factor** là tỉ lệ tăng kích thước của khung cửa sổ tìm kiếm. Ví dụ nếu scale\_factor=1.3 thì sau khi quét hết bức ảnh 1 lần, khung cửa sổ sẽ tăng kich thước lên 30% và thực hiện lần quét tiếp theo. Tham số này ảnh hưởng đến tốc độ xử lý và độ tin cậy của chương trình. Nếu để nó quá lớn thì tốc độ chương trình sẽ tăng lên do số lần quét giảm đi, tuy nhiên có thể chương trình có thể bỏ qua không phát hiện được một số khuôn mặt có kích thước nằm giữa 2 khung cửa sổ liên tiếp do độ tăng kích thước của khung là quá lớn. Nếu để nó quá thấp thì ta có thể không bỏ sót bất kì khuôn mặt nào nhưng chương trình sẽ tốn thời gian hơn vì tăng số lần quét lên.
* **min\_neighbors:** Trong quá trình tìm kiếm khuôn mặt chương trình sẽ tìm được nhiều những khung hình chữ nhật chứa khuôn mặt cho dù đó chỉ là một khuôn mặt và có nhưng trường hợp nhận diện nhầm. Do đó **min\_neighbors** giá trị tối thiểusố hình chữ nhật lân cận được gộp lại sau khi quá trình quét đã xong. Ví dụ min\_neighbors=5 thì sau khi quét xong bức ảnh, tối thiểu 5 hình chữ nhật lân cận sẽ được gộp lại thành một khung có chứa khuôn mặt.ads

1. **Facial landmark:**

Facial landmark là xác định các vị trí như eye(mắt), nose(mũi),mounth(miệng) trên khuôn mặt(face). Nó có rất nhiều ứng dụng vui mà ta thường thây trên các app điện thoại chẳng hạn ( swap face,draw in face or tạo hiệu ứng trên khuôn mặt).

Trong đồ án này, chúng em sẽ tìm hiểu về thư viện Dlib trong python để xác định Facial landmark.Facical landmark có cấu trúc như hình bên dưới. Dlib sẽ detection 62 point trên khuôn mặt:

Mouth point = 48-61

Right\_brow\_point = 17-21

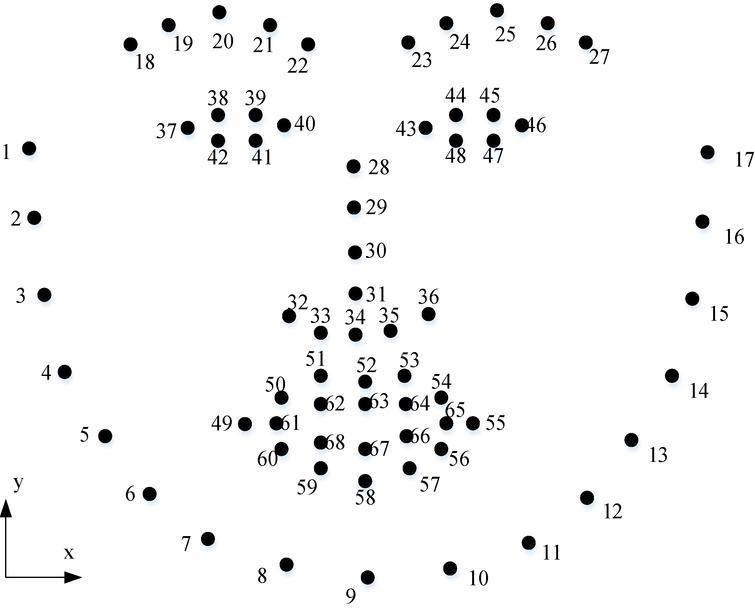
Left\_brow\_point = 22-26

Right\_eye\_point = 36-42

Left\_eye\_pint = 42-48

Nose\_point = 27-35

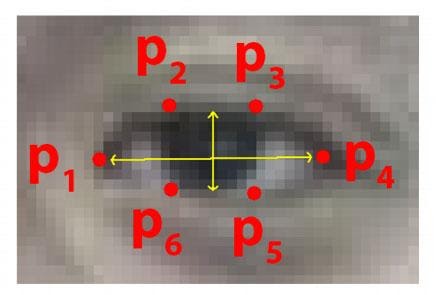
Jaw\_point = 0-17



**Hình 11. 68 điểm nhận dạng trên khuôn mặt**

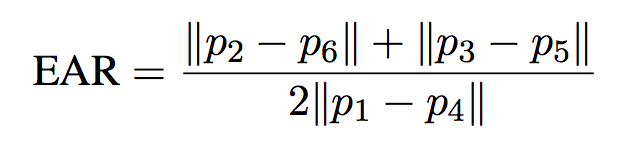
* Tính lệ khung hình của mắt:

Mỗi mắt được biểu thị bằng 6 (x, y) -tọa độ, bắt đầu từ góc trái của mắt (như thể bạn đang nhìn vào người), sau đó làm việc theo chiều kim đồng hồ xung quanh phần còn lại của vùng:



Hình 12: Các vị trí toạ độ của mắt trong facial landmark

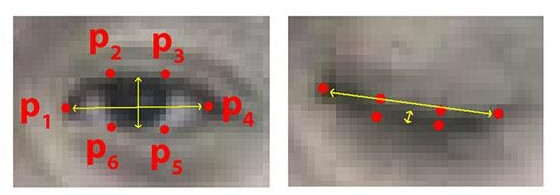
Công thức tính tỉ lệ khung hình mắt:



Trong đó p1,…, p6 là các vị trí mốc trên khuôn mặt 2D.

Tử số của phương trình này tính khoảng cách giữa các mốc mắt dọc trong khi mẫu số tính khoảng cách giữa các mốc mắt ngang, cân mẫu số một cách thích hợp vì chỉ có một tập hợp các điểm nằm ngang nhưng có hai tập hợp các điểm thẳng đứng.Tỷ lệ khung hình của mắt gần như không đổi khi mắt đang mở, nhưng sẽ nhanh chóng giảm xuống 0 khi diễn ra chớp mắt.

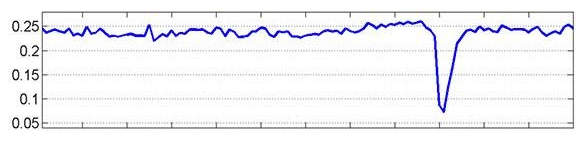
Sử dụng phương trình đơn giản này, chúng ta có thể tránh các kỹ thuật xử lý hình ảnh và chỉ cần dựa vào tỷ lệ khoảng cách mốc của mắt để xác định xem một người có đang chớp mắt hay không.



Hình 13: Các vị trí mốc trước và sau khi nhắm mắt

Ở trên cùng bên trái, chúng ta có một mắt mở hoàn toàn - tỷ lệ khung hình của mắt ở đây sẽ lớn (r) và tương đối không đổi theo thời gian. Tuy nhiên, khi người đó chớp mắt ( trên cùng bên phải ), tỷ lệ khung hình của mắt giảm đột ngột, gần bằng không.

Hình dưới vẽ biểu đồ tỷ lệ khung hình của mắt theo thời gian cho một video clip. Như chúng ta có thể thấy, tỷ lệ khung hình của mắt là không đổi, sau đó nhanh chóng giảm xuống gần bằng không, sau đó lại tăng lên, cho thấy một lần chớp mắt đã diễn ra.



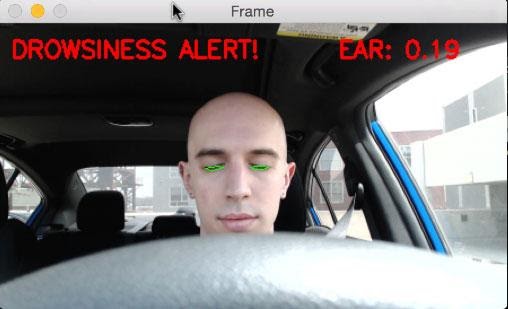
**Hình 14: Biểu đồ tỷ lệ khung hình của mẳt**



**Hình 15. Áp dụng khoanh vùng các mốc trên khuôn mặt để tách vùng mắt ra khỏi khuôn mặt**

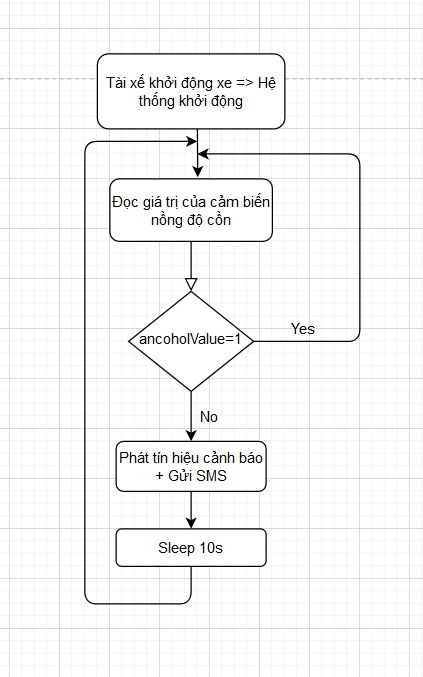


**Hình 16. Tính toán tỷ lệ khung hình của mắt để xác định xem mắt có nhắm được không**



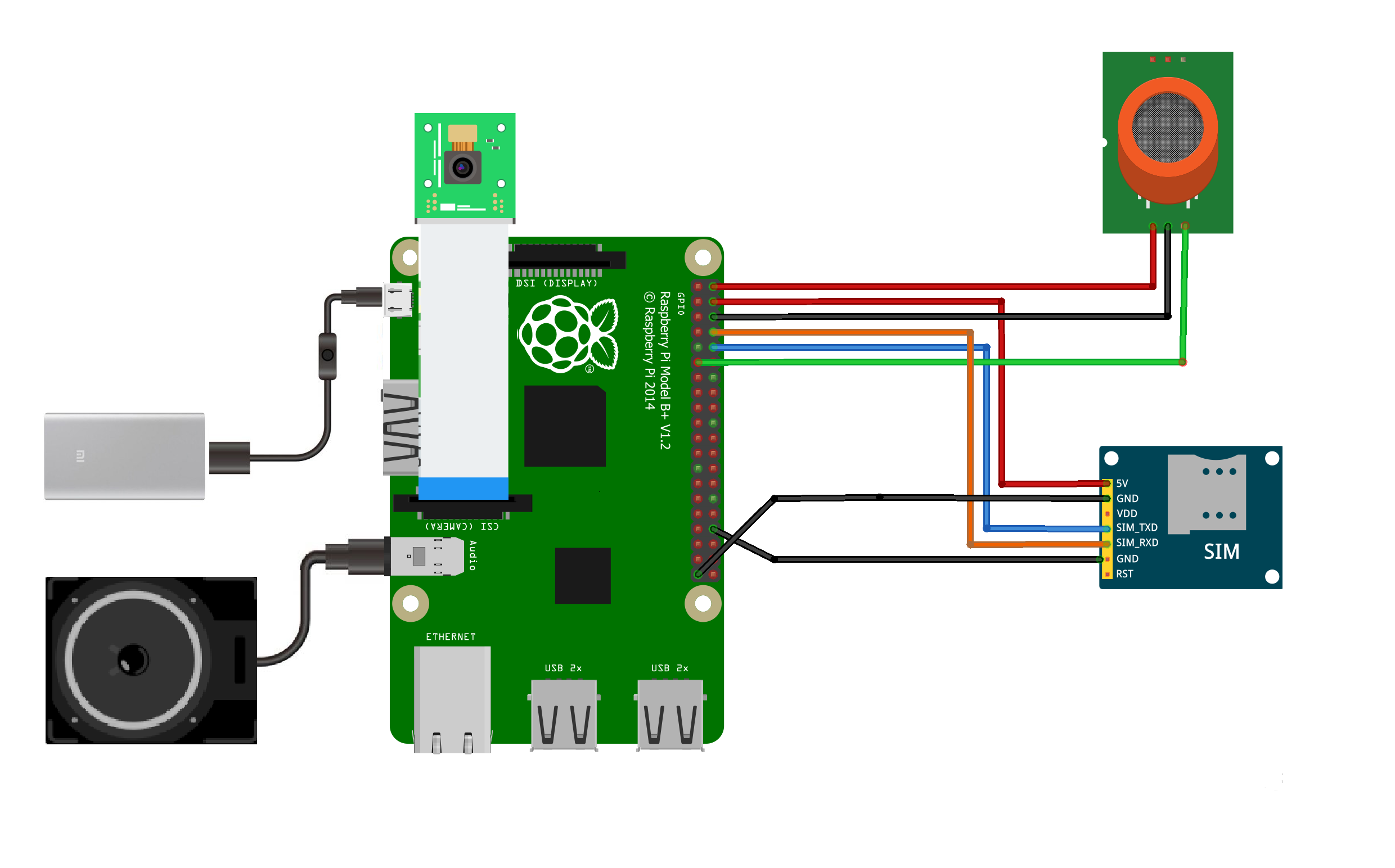
**Hình 17. Báo động nếu mắt nhắm đủ lâu**

1. Hệ thống cảnh báo nồng độ cồn :



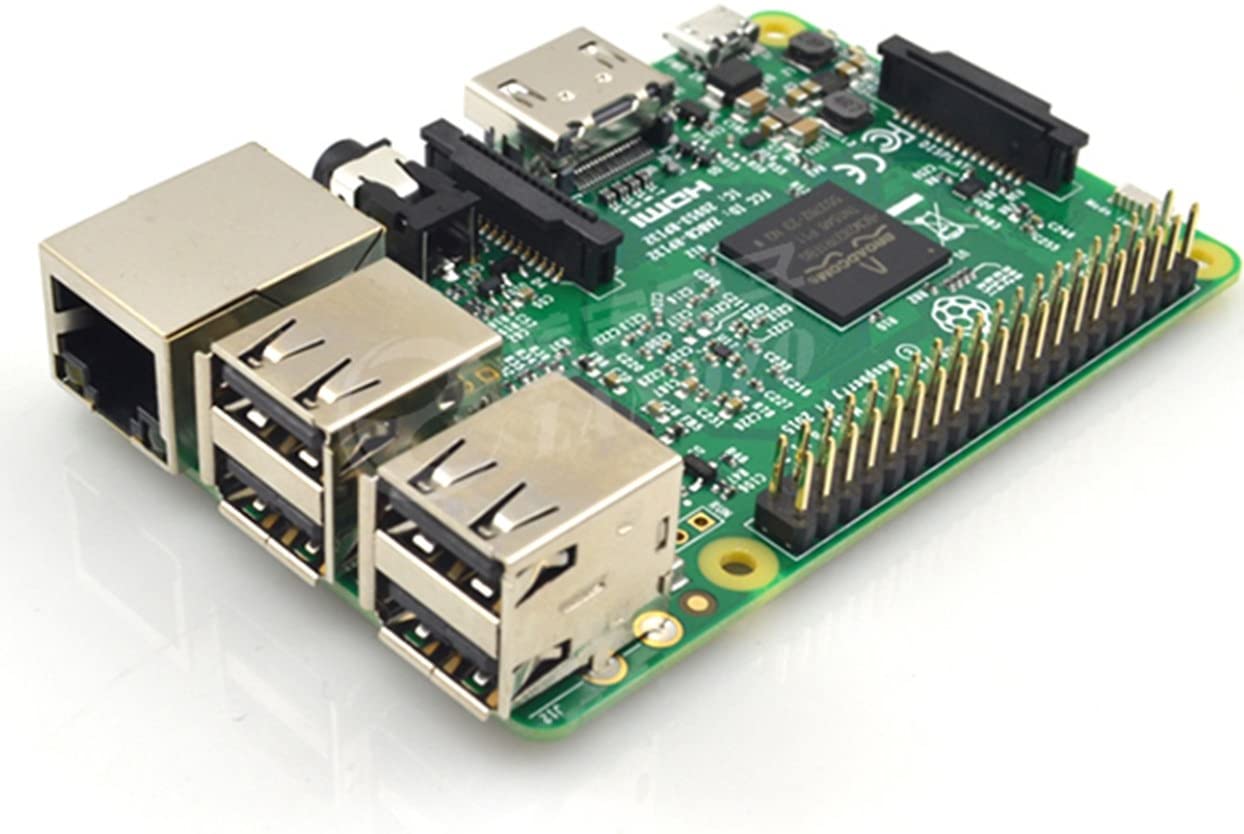
Hình 18. Sơ đồ hệ thống cảnh báo nồng độ cồn

1. Sơ đồ mạch của hệ thổng :



Hình 19. Sơ đồ mạch của hệ thống

1. Giới thiệu một số phần cứng, linh kiện sử dụng :



* 1. Raspberry Pi 3B:

Thông số kĩ thuật :

Quad Core 1.2GHz Broadcom BCM2837 64bit CPU1 1GB RAM

BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board

GPIO mở rộng 40 chân

Hình 20. Raspberry Pi 3B

4 cổng USB

* 1. Camera Raspberry



Độ phân giải: HD1080P/ 5 megapixel 30fps.

* 1. Cảm biến nồng độ cồn MQ3:

Hình 21. Camera Raspberry



**Hình 22. Cảm biến nồng độ cồn MQ3**

Các chân của cảm biến nồng độ cồn MQ3:

* Chân Vcc: Cung cấp nguồn điện cho cảm biến
* Chân GND: Chân nối đất
* Chân D0: Đọc giá trị digital của cảm biến. 0 (low) tương ứng với có cồn trong không khí và 1 (high) tương ứng với không có cồn.
* Chân A0: Đọc giá trị Analog của cảm biến.

Trong đồ án này chỉ cần xử dụng chân Do của cảm biến để xác định có cồn trong không khí hay không (Do pháp luật Việt Nam quy định tài xế lái xe 4 bánh trở lên nồng độ cồn phải <= 0). Khi phát hiện cồn trong không khí đèn led trên cảm biến sẽ sáng lên và gửi giá trị digital =0 về cho rasberry xử lí.

* 1. Module sim 800A :



**Hình 23. Module Sim 800A**

- Các chân của module Sim 800A: + Chân Vcc: Cấp nguồn cho module

+ Chân GND: Chân nối đất

+ Chân Rx: Chân nhận tín hiệu TTL

+ Chân Tx: Chân truyền tín hiệu TTL

- Hoạt động: Chân Rx của module nối với chân Tx của ras để nhận tín hiệu của Ras. Chân Tx của module nối với chân Rx của Ras để truyền tín hiệu ngược lại cho Ras.

- Tập lệnh AT dùng để điều khiển nhắn tin:

+  AT+CMGF=1 : Đưa SMS về chế độ text, phải có lệnh này mới có thể nhắn tin dạng text được.

+ AT+CMGS=”Số\_điện\_thoại” : Gửi SMS đến số điện thoại chỉ định trước

+ AT+CNMI=2,1,0,0,0 : Tạo SMS mới

+ port.write(“Content”) : Tạo nội dung của SMS

+ port.write(‘\x1A’) Lệnh gửi SMS



* 1. Loa:

Dùng để thông báo đến người dùng kết quả nhận dạng thành công hay không thể nhận dạng được.

Thông số kĩ thuật:

Công suất: 5W ; Điện áp hoạt động: 5V/1A

Hình 24. Loa

* 1. Nguồn :



Trong đồ án này nhóm em sử dụng nguồn là pin sạc dự phòng Xiaomi Mi Gen 2 có khả năng điều chỉnh cường độ dòng điện phù hợp với Raspberry Pi 3B+

Thông số kĩ thuật:

Dòng điện ra: 5.1V-2.4A / 9V-1.6A (MAX) và 12V-1.2A (MAX) - MAX 15W

Hình 25. Pin sạc

|  |  |  |
| --- | --- | --- |
| STT | Tên linh kiện | Giá |
| 1 | Raspberry Pi 3B | 1190000 VNĐ |
| 2 | Camera 5MP cho Raspberry Pi | 110000 VNĐ |
| 3 | Module sim 800A | 200000 VNĐ |
| 4 | Cảm biến nồng độ cồn | 40000 VNĐ |
| 5 | Loa | 200000 VNĐ |
| 6 | Cáp HDMI | 40000 VNĐ |
| 7 | Pin sạc dự phòng | 200000 VNĐ |
| 8 | Breadboard, dây dẫn | 40000 VNĐ |
|  | Tổng cộng: | 2015000 VNĐ |

# Bảng 2. Bảng kê chi phí các linh kiện

# KẾT QUẢ:

1. **Mô tả dữ liệu kiểm thử :**

* 1 sinh viên sẽ vào vai người tài xế để hệ thống nhận diện.
* Dùng cồn trong lon bia để thử nghiệm hệ thống cảnh báo tài xế uống rượu bia.

1. **Quy trình kiểm thử :**

- Đầu tiên là thử khả năng nhận diện khuôn mặt của hệ thống, kiểm tra kết quả trên màn hình LCD.

- Tiếp theo nhắm mắt trong khoảng thời gian < thời gian đặt ra để kiểm tra hệ thống có báo động không. Có => Hệ thống tính toán sai.

- Nhắm mắt trong khoảng thời gian >= thời gian đặt ra để kiểm tra hệ thống nhận dạng tài xế ngủ gật và báo động.

- Cuối cùng là kiểm tra hoạt động của hệ thống cảnh báo tài xế uống rượu bia bằng cách đưa cảm biến lại gần lon bia. Có hơi cồn => hệ thống đưa ra cảnh báo và gửi tin nhắn cho người thân của tài xế.

1. **KẾT LUẬN :**
2. **Ưu điểm:**

* Sản phẩm đáp ứng các yêu cầu đề ra, triển khai và vận hành tương đối tốt.
* Có cảnh báo cho người sử dụng bằng loa.
* Hệ thống dễ sử dụng.
* Theo dõi người dùng theo thời gian thực.
* Hoạt động không cần internet.

1. **Nhược điểm:**

* Linh kiện còn rườm rà, hình thức bên ngoài chưa gọn nhẹ.
* Camera chưa nhận dạng tốt trong môi trường thiếu sáng.
* Quá trình nhận dạng chưa thực sự nhanh và chính xác tuyệt đối.

1. **Kết luận:**

* Hệ thống đã cơ bản đáp ứng được các yêu cầu đề ra ban đầu.
* Quá trình nhận dạng tương đối chính xác.

1. **Hướng phát triển:**

* Cải thiện quá trình nhận dạng nhanh và chính xác hơn.
* Thêm phần định vị GPS cho thiết bị để gửi về cho người thân.

# 

# TÀI LIỆU THAM KHẢO

[1] OpenCV Dev team, “Face Recognition with OpenCV”, OpenCV, Dec 31, 2019. [Online].Available:https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec\_tutorial.html

[2] [Zuzana Képešiová](https://www.researchgate.net/profile/Zuzana_Kepesiova), “Integral Image”, [Online]. Availble: <https://www.researchgate.net/figure/mage-conversion-to-summed-area-table-also-known-as-Integral-image_fig1_324554906>

[3] Le Van Lap, “Nhận diện khuôn mặt OpenCV Python,”, Apr 24, 2019. [Online]. Availble: <https://github.com/leevawns/opencv-haar-gui>. [Accessed: Feb 12, 2020].

[4] Adrian Rosebrock, PhD, “Eye blink detection with OpenCV, Python, and dlib” <https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/?fbclid=IwAR1Ua3jkype9Y2-lb6yPIx_PcWQQTcovx85Qw-yASmQasXLZ9GT9XpO-Fkc>

[5] Adrian Rosebrock, PhD, “Drowsiness detection with OpenCV” <https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/?fbclid=IwAR0RhIdR8B7BWZAzJ6Cj70hl_LDs_4m1MIZUpNR3bGC45khDhxAIq_jaCFs>

[6] Thorpham Blog, “Face landmark với Dlib” <https://thorpham.github.io/blog/2018/04/20/face-landmark/>

[7] Ngoc N Tran, “Haar Cascade là gì? Luận về một kỹ thuật chuyên dùng để nhận biết các khuôn mặt trong ảnh.” https://viblo.asia/p/haar-cascade-la-gi-luan-ve-mot-ky-thuat-chuyen-dung-de-nhan-biet-cac-khuon-mat-trong-anh-E375zamdlGW