**Trường Đại Học Công Nghệ Thông Tin**

**Khoa Khoa Học Máy Tính**

**ĐỒ ÁN CUỐI KÌ**

Môn Học: Nhập Môn Thị Giác Máy Tính

GVHD: Nguyễn Vinh Tiệp.

SVTH: Nguyễn Ngọc Thiện – 16521811.

Đề Tài: Nhận Diện Cử Chỉ Tay.

**MỤC LỤC**

[**I. GIỚI THIỆU ĐỀ TÀI 1**](#_Toc28167238)

[I.1. Lý do chọn đề tài 1](#_Toc28167239)

[I.2. Mục đích yêu cầu của đề tài 2](#_Toc28167240)

[**II. TIẾN HÀNH PHÁT TRIỂN 2**](#_Toc28167241)

[II.1. Không gian màu HSV 2](#_Toc28167242)

[II.2. Toán tử contour 3](#_Toc28167243)

[**III. THỰC NGHIỆM VÀ ĐÁNH GIÁ 4**](#_Toc28167244)

[III.1. Thực nghiệm: 4](#_Toc28167245)

[III.2. Đánh giá: 6](#_Toc28167246)

# GIỚI THIỆU ĐỀ TÀI

## Lý do chọn đề tài

Trong các lĩnh vực đời sống hiện nay, các thành tựu khoa học công nghệ đang được áp dụng mạnh mẽ và không ngừng phát triển. Nhận dạng là một trong những bài toán giúp máy tính hiểu được con người thông qua cử chỉ, hành động, lời nói…Phát hiện cử chỉ bàn tay là 1 đề tài khá thông dụng, có thể ứng dụng vào điều khiển robot, smarthome,…Có nhiều thuật toán khác nhau để phát hiện cử chỉ bàn tay nhưng trong đề tài lần này nhóm chỉ sử dụng màu sắc để nhận diện cử chỉ bàn tay.

Thị giác máy tính (computer vision) đề cập đến toàn bộ quá trình mô phỏng tầm nhìn của con người trong một bộ máy phi sinh học. Điều này bao gồm việc chụp ảnh ban đầu, phát hiện và nhận dạng đối tượng, nhận biết bối cảnh tạm thời giữa các cảnh và phát triển sự hiểu biết ở mức độ cao về những gì đang xảy ra trong khoảng thời gian thích hợp.

Nhận diện cử chỉ của con người là một vấn đề quan trọng được đạt ra cho thị giác máy tính. Việc thực hiện lệch theo cử chỉ của con người (cụ thể ở đây là cử chỉ tay) đặt ra những thử thách lớn trong quá trình phát triển ứng dụng. Đó là làm sao để nhận diện đúng bàn tay con người trong những bối cảnh phức tạp (nhiều điểm nhiễu làm ứng dụng nhận diện nhầm)… Kiến trúc cần thiết để phát hiện đối tượng khác nhau theo một cách quan trọng. Cụ thể, kích thước của vectơ đầu ra không cố định. Ví dụ, nếu có một đối tượng trong hình, thì sẽ có bốn tọa độ xác định hộp giới hạn. Giá trị tĩnh và được xác định trước này hoạt động bằng cách sử dụng các kiến ​​trúc đã đề cập trước đó. Tuy nhiên, khi số lượng đối tượng tăng lên, số lượng tọa độ cũng tăng theo. Đặc biệt là số lượng đối tượng không được biết trước, điều này đòi hỏi phải điều chỉnh trong trang điểm của mạng lưới thần kinh.

## Mục đích yêu cầu của đề tài

Đề tài thực hiện thiết kế ứng dụng nhận diện cử chỉ bàn tay thông qua Python và OpenCV.

Đề tài hướng đến mục tiêu nhận diện chính xác cử chỉ bàn tay trong nhiều môi trường khác nhau.

Trên tất cả, vấn đề lớn nhất mà nhóm đang tập trung giải quyết chính là độ ổn định và độ chính xác của ứng dụng phải được tối ưu nhất có thể và phải đặt lên hàng đầu.

# TIẾN HÀNH PHÁT TRIỂN

## Không gian màu HSV

Không gian màu HSV (còn gọi là HSB) là một cách tự nhiên hơn để mô tả màu sắc, dựa trên 3 số liệu:

* H: (Hue) Vùng màu
* S: (Saturation) Độ bão hòa màu
* B (hay V): (Bright hay Value) Độ sáng

Giả sử ta có một điểm màu có giá trị trong hệ RGB là (R, G, B). ta chuyển sang không gian HSV như sau: Đặt M = Max(R, G, B), m = Min(R, G, B) và C = M - m. Nếu M = R, H' = (G - B)/C mod 6. Nếu M = G, H' = (B - R)/C + 2. Nếu M = B, H' = (R - G)/C + 4. Và H = H'x60. Trong trường hợp C = 0, H = 00 V = M. S = C/V. Trong trường hợp V hoặc C bằng 0, S = 0. Để chuyển từ HSV sang RGB ta làm như sau: Giả sử ta có không gian màu HSV với H = [0, 360], S = [0, 1], V = [0, 1]. Khi đó, ta tính C = VxS. H' = H/60. X = C(1 - |H' mod2 -1|). Ta biểu diễn hệ (R1, G1, B1) như sau:

* ( 1, 1, 1) ={
* (0, 0, 0) nếu H chưa được xác định
* ( C, X , 0) nếu 0<H'<1
* ( X, C, 0) nếu 1<H'<2
* ( 0, C ,X) nếu 2<H'<3
* ( 0, X , C) nếu 3<H'<4
* ( X, 0 , C) nếu 4<H'<5
* ( C, 0 , X) nếu 5<H'<6}

Sử dụng hàm HSV = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV). Với hàm cv2.cvtColor chúng ta dễ dàng chuyển màu BGR sang HSV trong khung hình màu cho video khi được hiển thị.

## Toán tử contour

Toán tử contour là “tập các điểm-liên-tục tạo thành một đường cong (curve) (boundary), và không có khoảng hở trong đường cong đó, đặc điểm chung trong một contour là các các điểm có cùng /gần xấu xỉ một giá trị màu, hoặc cùng mật độ. Contour là một công cụ hữu ích được dùng để phân tích hình dạng đối tượng, phát hiện đối tượng và nhận dạng đối tượng”. Để tìm contour chính xác, chúng ta cần phải nhị phân hóa bức ảnh ( là ảnh nhị phân, không phải ảnh grayscale ).

Cấu trúc Contour trong OpenCV:

modifiedImage, contours, hierarchy = cv2.findContours(binaryImage, typeofContour, methodofContour)

Trong đó:

* contours: Danh sách các contour có trong bức ảnh nhị phân. Mỗi một contour được lưu trữ dưới dạng vector các điểm
* hierarchy: Danh sách các vector, chứa mối quan hệ giữa các contour.
* modifiedImage: Ảnh sau khi sử dụng contour, thường chúng ta không xài đối số này
* binaryImage: Ảnh nhị phân gốc. Một chú ý quan trọng ở đây là sau khi sử dụng hàm findContours thì giá trị của binaryImage cũng thay đổi theo, nên khi sử dụng bạn có thể áp dụng binaryImage.copy() để không làm thay đổi giá trị của binaryImage
* typeofContour: có các dạng sau: RETR\_EXTERNAL, RETR\_LIST, RETR\_CCOMP, RETR\_TREE, RETR\_FLOODFILL.
* methodofContour: Có các phương thức sau: CHAIN\_APPROX\_NONE, CHAIN\_APPROX\_SIMPLE,CHAIN\_APPROX\_TC89\_L1,HAIN\_APPROX\_TC89\_KCOS.

# THỰC NGHIỆM VÀ ĐÁNH GIÁ

1. **Thực nghiệm:**

A screenshot of a cell phone

Description automatically generated

*Hình 3.1: Chuyển ảnh BGR sang ảnh nhị phân*

A picture containing drawing

Description automatically generated

*Hình 3.2: Lọc nhiễu cho đối tướng*

A picture containing indoor, green, sitting, table

Description automatically generated

A picture containing using, holding, cutting

Description automatically generated*Hình 3.2: Xác định Contour*

*Hình 3.2: Xác định CONVEX HULL*

1. **Đánh giá:**

**Trong điều kiện lý tưởng:**

* Nhận diện được các số từ 0-5, OK, Cool (THUMBSUP)
* Dễ nhầm lẫn giữa số 3 và OK

**Ưu điểm:**

* Xử lý tốc độ nhanh, khoảng 0.004s
* Đơn giản dễ thực hiện

**Nhược điểm:**

* Lọc nhiễu kém, chỉ nhận được 1 màu tay đã lấy mẫu trước đó
* Dễ bị ảnh hưởng bởi môi trường
* Việc tính toán dựa trên diện tích để nhận dạng không tối ưu.

**Cải tiến so với lần báo cáo**:

* Thêm cử chỉ **OK** và **Cool.**