**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

🙞∙∙∙☼∙∙∙🙜



**NHẬP MÔN THỊ GIÁC MÁY TÍNH**

**EXERCISE1**

**Giảng viên hướng dẫn: PGS.TS. Lê Đức Hạnh**

**Danh sách thành viên:**

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | **Mã số sinh viên** |
| Trần Quang Đạo | 2210647 |
| Đào Trọng Chân | 2210350 |
| Võ Hữu Dư | 2210604 |
| Dương Quang Duy | 2210497 |

**Thành phố Hồ Chí Minh – 2024**

**MỤC LỤC**

[**I. Bài 1 3**](#_Toc209389918)

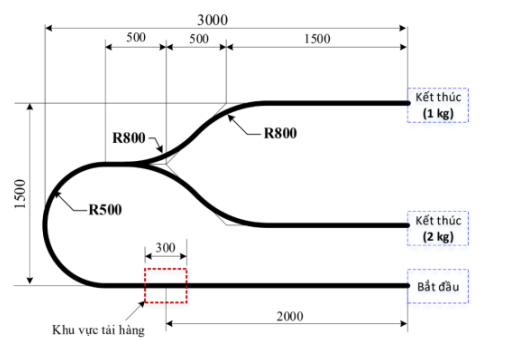
[**II. Bài 2 7**](#_Toc209389923)

[**III. Bài 3 13**](#_Toc209389924)

[**IV. Bài 4 17**](#_Toc209389925)

[**V. Bài 5 21**](#_Toc209389926)

# **I. Bài 1**



**- Từ yêu cầu đề bài ta có:**

+ Smallest feature: 1mm

+ FOV = 3000 x 1500

**- Tính độ phân giải**

Độ phân giải : resolution =

**- Khoảng cách làm việc**

**- FOV height mới khi theo tỷ lệ**

**- Lựa chọn Basler**

Khi vào trang web basler (<https://www.baslerweb.com/en-sg/tools/vision-system-configurator/?component=lens_selector#option-camera> ) nhập các thông số FOV và smallest feature web hiện ra 9 kết quả:

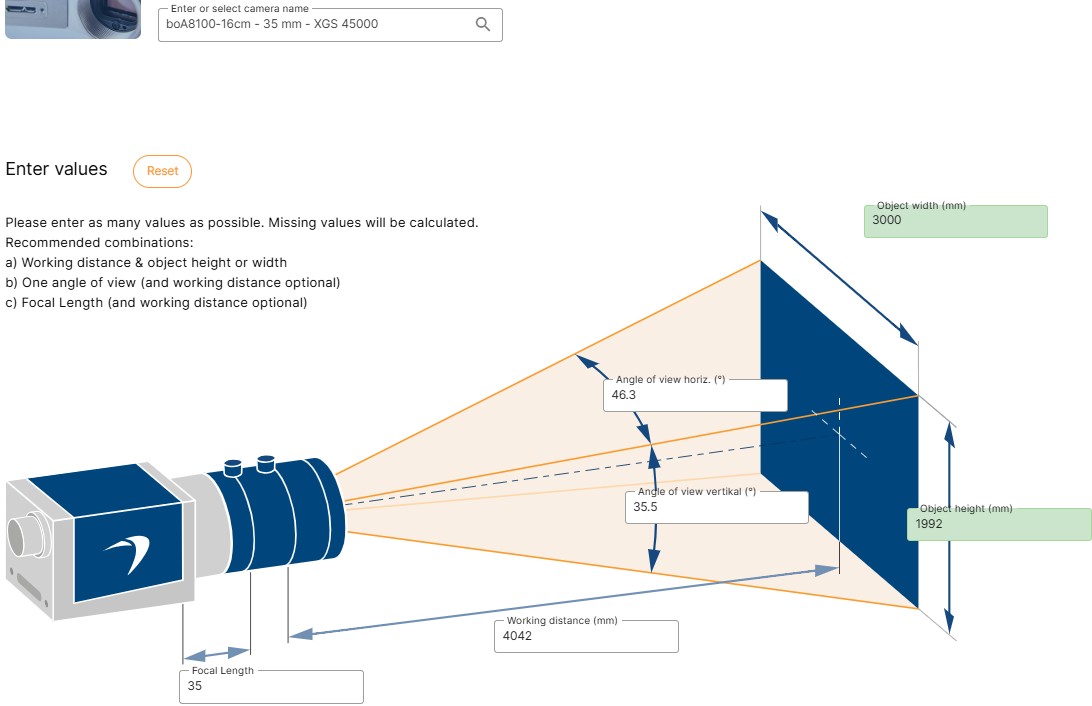
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Basler boost**  boA8100-16cm/16cc | **Basler boost**  boA6500-36cm/36cc | **Basler boost**  boA9344-30cm/30cc/70cm/70cc | **Basler boost**  boA13440-17cm |
| **Resolution(px)** | 8192x5460 | 6580x4935 | 9344x7000 | 13376x9528 |
| **Sensor size(mm)** | 26.2 × 17.4 | 21 × 15.8 | 29.9 × 22.4 | 46.14 × 32.87 |
| **Sensor diagonal(mm)** | 31.45 | 26.28 | 37.36 | 56.65 |
| **Pixel Size (** | 3.2 × 3.2 | 3.2 × 3.2 | 3.2 × 3.2 | 3.45 × 3.45 |
| **Focal length(mm)** | 35 | | | |
| **FOV width (mm)** | 3000 | | | |
| **New FOV height (mm)** |  |  |  |  |
| **Working distance(mm)** |  |  |  |  |

**Bảng 1.1:** Tính toán thông số camera Basler

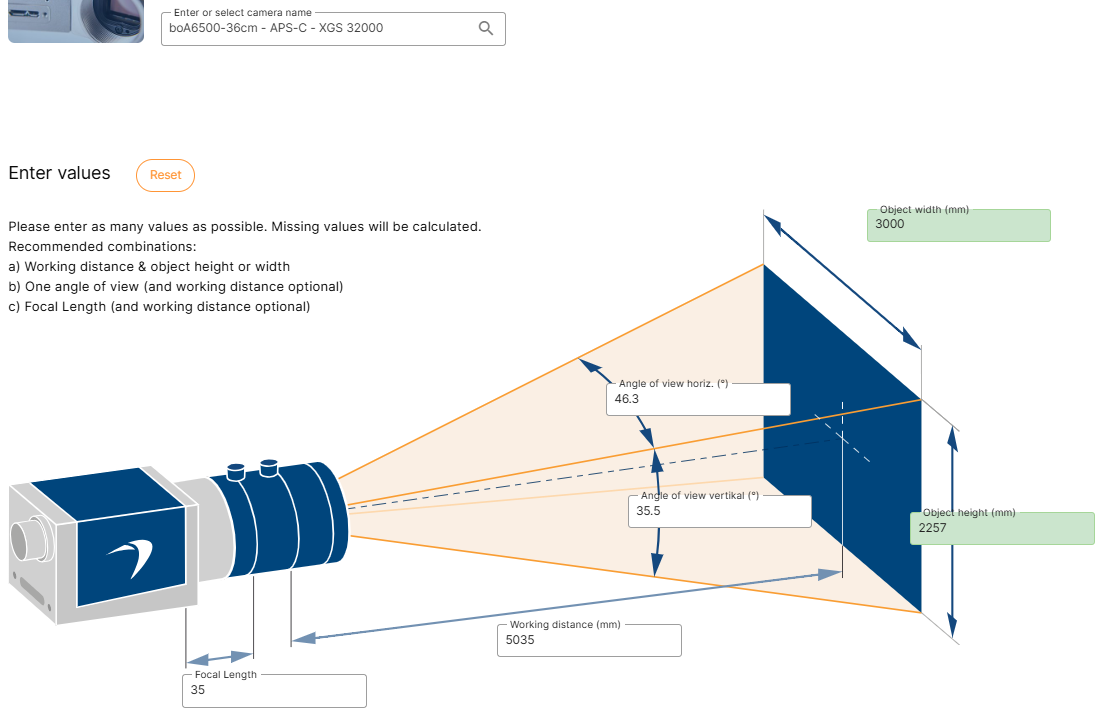
Kiểm tra bằng trang web, chọn model sau đó nhập Focal length (35mm) và FOV width (3000mm):

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Tính tay** | **Kết quả web** |  |
| boA8100-16cm/16cc | 4013 | 4042 | <https://www.baslerweb.com/en-sg/tools/lens-selector/#option-camera;model=boA8100-16cm;focallength=35;workingdistance=4042> |
| boA6500-36cm/36cc | 4495 | 5035 | <https://www.baslerweb.com/en-sg/tools/lens-selector/#option-camera;model=boA6500-36cm;focallength=35;workingdistance=5035> |
| boA9344-30cm/30cc/70cm/70cc | 3512 | 3546 | <https://www.baslerweb.com/en-sg/tools/lens-selector/#option-camera;model=boA9344-70cm;focallength=35;workingdistance=3546> |

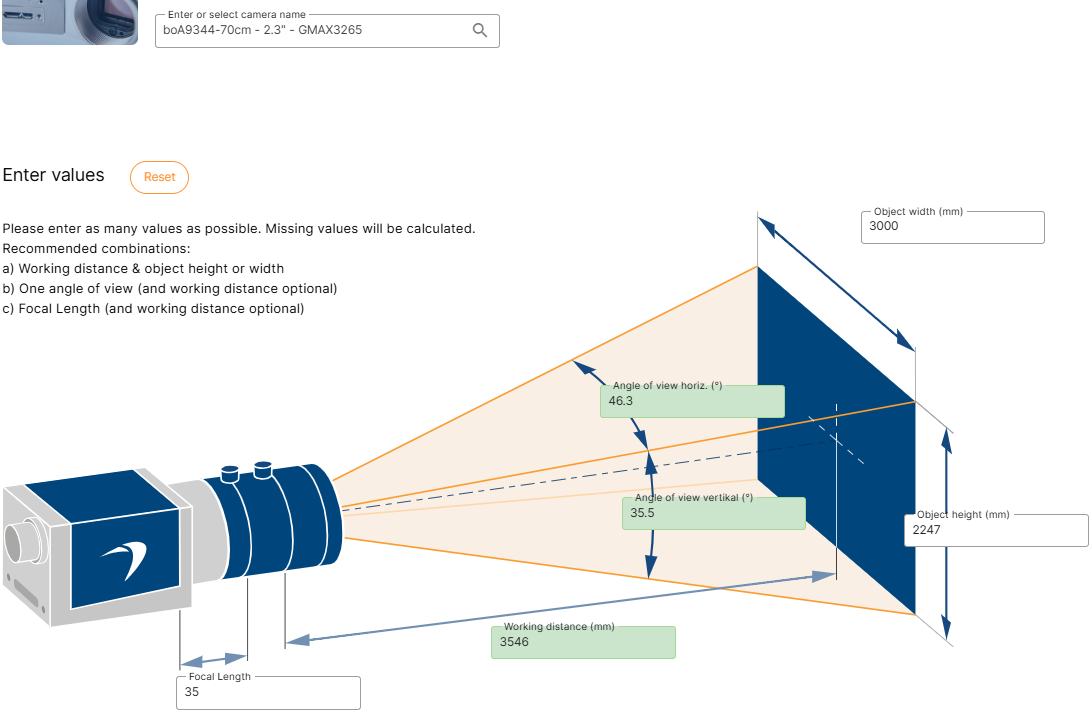
**Bảng 1.2:** So sánh kết quả



**Hình 1.1:** Kết quả trên web camera boA8100-16cm/16cc



**Hình 1.2:** Kết quả trên web camera boA6500-36cm/36cc



**Hình 1.3:** Kết quả trên web camera boA9344-30cm/30cc/70cm/70cc

* Hiện tại Basler boost boA13440-17cm chưa có model trên web

# **II. Bài 2**

1. **Equalize Histogram this picture in range 0-255, using 256 bin**

Cân bằng histogram :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mức xám** | **Pixel** | **Phân bố**  **(Probability)** | **Tổng tích lũy (Cumulative Probability)** | **Trải đều 255 (CP\*255)** | **Làm tròn (Scaled Intensity)** | **Phân bổ lại (New pixel)** |
| 20 | 1 | 1/16 | 1/16 | 15.9375 | 15 | 0 |
| 30 | 1 | 1/16 | 2/16 | 31.875 | 32 | 17 |
| 50 | 4 | 4/16 | 6/16 | 95.625 | 96 | 85 |
| 80 | 1 | 1/16 | 7/16 | 111.5625 | 112 | 102 |
| 100 | 1 | 1/16 | 8/16 | 127.5 | 128 | 119 |
| 110 | 1 | 1/16 | 9/16 | 143.4375 | 144 | 136 |
| 120 | 1 | 1/16 | 10/16 | 159.375 | 160 | 153 |
| 150 | 1 | 1/16 | 11/16 | 175.3125 | 176 | 170 |
| 160 | 1 | 1/16 | 12/16 | 191.25 | 192 | 187 |
| 220 | 1 | 1/16 | 13/16 | 207.1875 | 208 | 204 |
| 230 | 1 | 1/16 | 14/16 | 223.125 | 224 | 221 |
| 240 | 1 | 1/16 | 15/16 | 239.0625 | 240 | 238 |
| 250 | 1 | 1/16 | 1 | 255 | 255 | 255 |
| MxN = 16 | | Cumulative probability min (CPM)=1 | | | L(Bin)=256 | |

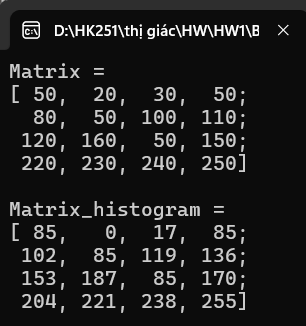
**Bảng 1.3:** Bảng cân bằng histogram

**Kết luận:**

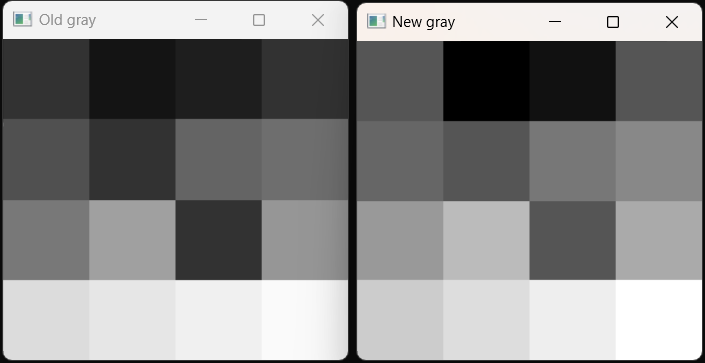
1. **Program using opencv lib to equalize histogram this picture and display histogram before and after equalize**

|  |
| --- |
| #include "opencv2/opencv.hpp"  using namespace cv;  using namespace std;  void histogram(string const& name, Mat const& Image)  {  int bin = 255;  int histsize[] = { bin };  float range[] = { 0,255 };  const float\* ranges[] = { range };  Mat hist;  int chanel[] = { 0 };  int hist\_heigt = 256;  Mat hist\_image = Mat::zeros(hist\_heigt, bin, CV\_8SC3);  calcHist(&Image, 1, chanel, Mat(), hist, 1, histsize, ranges, true, false);  double max\_val = 0;  minMaxLoc(hist, 0, &max\_val);  for (int i = 0; i < bin; i++)  {  float binV = hist.at<float>(i);  int height = cvRound(binV \* hist\_heigt / max\_val);  line(hist\_image, Point(i, hist\_heigt - height), Point(i, hist\_heigt), Scalar::all(255));  }  imshow(name, hist\_image);  }  int main(int argv, char\*\* argc)  {  float img[16] = { 50,20,30,50,  80,50,100,110,  120,160,50,150,  220,230,240,250 };  Mat gray\_img = Mat(4, 4, CV\_32F, img);  Mat gray\_img\_his;  namedWindow("Old gray", WINDOW\_FREERATIO);  namedWindow("New gray", WINDOW\_FREERATIO);  gray\_img.convertTo(gray\_img, CV\_8UC1);  equalizeHist(gray\_img, gray\_img\_his);  cout << "Matrix = " << endl << "" << gray\_img << endl << endl;  cout << "Matrix\_histogram = " << endl << "" << gray\_img\_his << endl << endl;  imshow("Old gray", gray\_img);  imshow("New gray", gray\_img\_his);  histogram("Old histogram", gray\_img);  histogram("New histogram ", gray\_img\_his);  waitKey();  } |

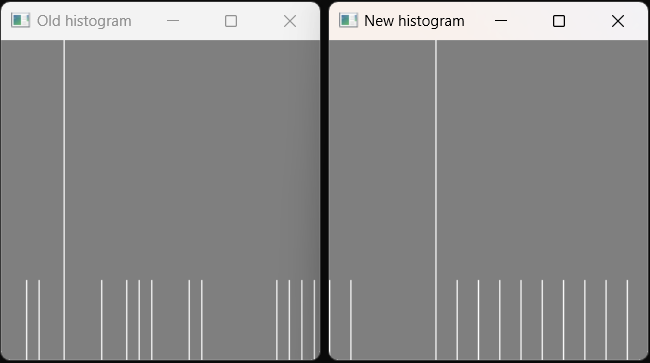
**Kết quả:**



**Hình 1.4:** Kết quả trước và sau khi cân bằng histogram



**Hình 1.5:** Ảnh trước và sau khi cân bằng



**Hình 1.6:** Histogram trước và sau khi cân bằng

1. **Threshold this picture using Otsu method with random initial guessing, Check again using opencv lib.**

Chọn

* **Phân 2 nhóm điểm ảnh G1 < T; G2>T:**

+ G1: 20; 30; 50; 80; 100; 110;120

+ G2: 150; 160; 220; 230; 240; 250

* **Phân 2 nhóm điểm ảnh G1 < ; G2 >**

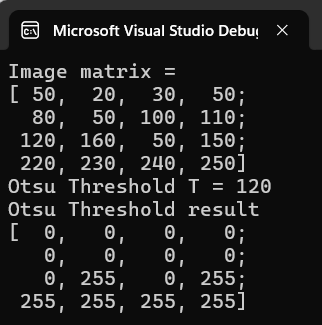
+ G1: 20; 30; 50; 80; 100; 110;120

+G2: 150; 160; 220; 230; 240; 250

Do đó, Otsu thresholding có giá trị là T = 134.165

**Kết quả sau khi cân bằng là:**

|  |
| --- |
| #include <opencv2/opencv.hpp>  using namespace cv;  using namespace std;  int main() {  Mat source = (Mat\_<uint8\_t>(4, 4) << 50, 20, 30, 50,  80, 50, 100, 110,  120, 160, 50, 150,  220, 230, 240, 250);  Mat dst;  double thresh = 0, maxValue = 255;  long double thres = threshold(source, dst, thresh, maxValue, THRESH\_OTSU);  cout << "Image matrix = " << endl << "" << source << endl;  cout << "Otsu Threshold T = " << thres << endl;  cout << "Otsu Threshold result" << endl << "" << dst << endl;  waitKey(0);  return 0;  } |



**Hình 1.7**: Kết quả phương pháp Otsu Threshold

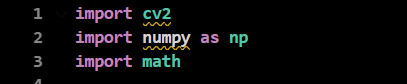
# **III. Bài 3**

**Two different pictures are shown in following figures (2 marks)**

**a) Calculate Euclidean distance between two (R,G,B) of pictures (remember to scale the pictures to the same size)**

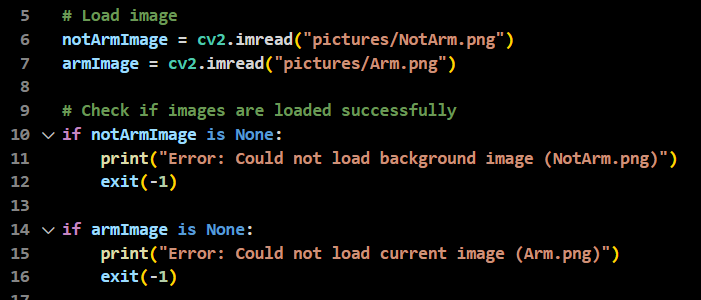
**b) Set threshold of Euclidean distance to see the hand.**

**1. Import thư viện**

****

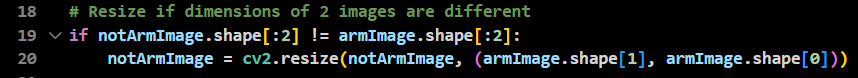
* **cv2**: xử lý ảnh với OpenCV.
* **numpy**: xử lý ma trận, ảnh được biểu diễn dạng mảng.
* **math**: dùng để tính toán căn bậc hai trong công thức Euclidean distance.

**2. Load ảnh và kiểm tra:**

****

* **notArmImage**: ảnh nền (không có bàn tay).
* **armImage**: ảnh hiện tại (có bàn tay).

**3. Resize ảnh**



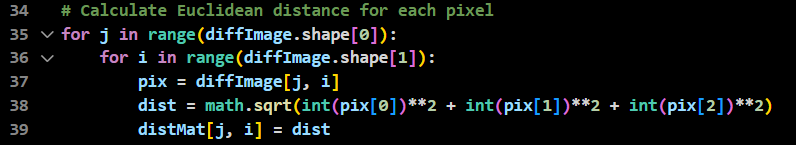
Cần cùng kích thước thì mới so sánh pixel theo pixel.

**4. Xác định ảnh chênh lệch tuyệt đối từng kênh màu:**

****

Với

**4. Tính khoảng cách Euclidean cho từng pixel**

****

**5. Tách bàn tay ra khỏi nền bằng threshold**

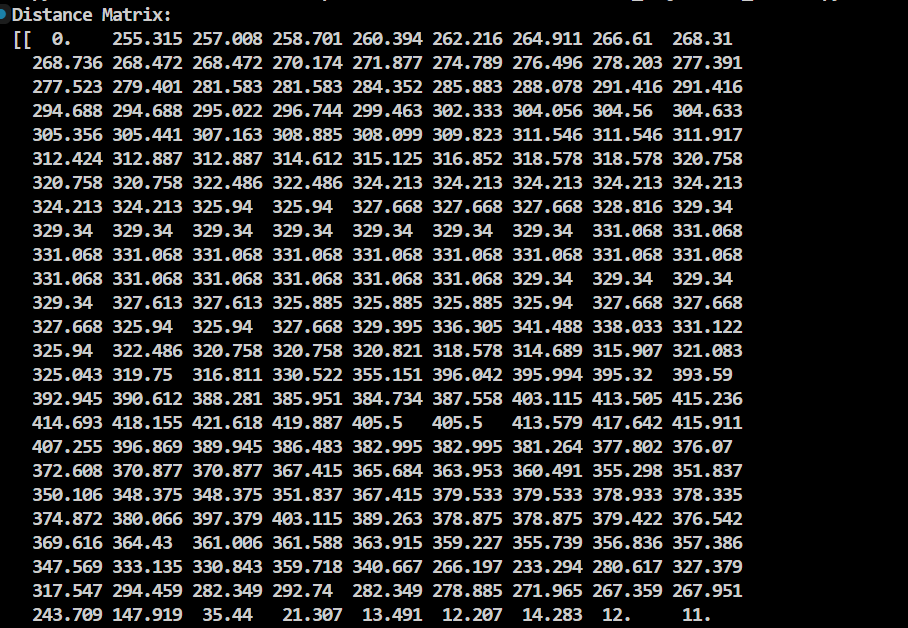
****

****

**6. Hiển thị kết quả:**

**6.1. Ma trận khoảng cách Euclidean:**

****

****

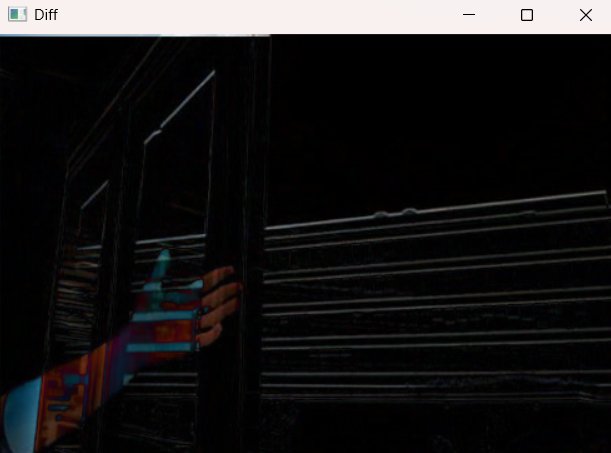
**Nhận xét:**

- Ma trận Euclidean biểu diễn mức độ khác biệt màu sắc giữa hai ảnh tại từng pixel.

- Giá trị trong ma trận càng **nhỏ (≈0)** → pixel của hai ảnh gần như giống nhau (vùng nền không thay đổi).

- Giá trị càng **lớn** → mức độ khác biệt càng cao, thể hiện có sự xuất hiện hoặc biến đổi của đối tượng (ở đây là bàn tay).

**6.2. Hiển thị ảnh**

****

**Nhận xét:**

**-** Đây là ảnh biểu diễn sự khác biệt tuyệt đối giữa ảnh nền (NotArm) và ảnh có bàn tay (Arm).

- Những vùng **giống nhau** giữa hai ảnh (tường, cửa, song cửa, nền) → màu rất tối hoặc đen gần như 0.

- Những vùng **có sự thay đổi** (bàn tay, một số viền cạnh do ánh sáng thay đổi) → hiện rõ dưới dạng vùng sáng hoặc có màu.



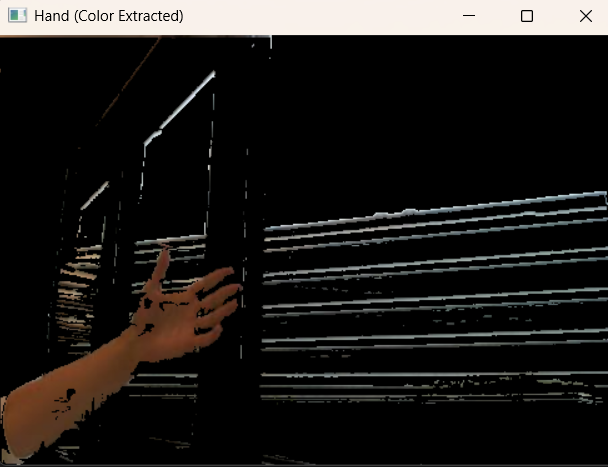
**Nhận xét:**

- Đây là ảnh biểu diễn trực quan của ma trận khoảng cách Euclidean.

- Mỗi điểm ảnh (pixel) có cường độ sáng tỉ lệ với độ khác biệt màu giữa hai ảnh tại vị trí đó.

+ **Màu sáng** (xám/trắng) → độ khác biệt lớn → chính là các vùng có bàn tay hoặc biên cạnh cửa

+ **Màu tối (đen)** → độ khác biệt nhỏ → các vùng nền giống nhau (tường, cửa kính không thay đổi).



**Nhận xét:** Với ngưỡng Threshold = 30 ta có thể phân tích được những điểm ảnh có sự khác biệt trong 2 bức ảnh có bàn tay và không có bàn tay

# **IV. Bài 4**

**Different pictures are shown in following figures (2 marks).**



**a) Program to store the number of white pixel on the circle with different Radius (1,5,10,15,20,25,30,35,40) and center of the circle is at the center of the picture to the TXT file (remember to scale the pictures to the same size).**

**b) Compare to identify the chessman.**

**Giải:**

a) Tiến hành resize tất cả các ảnh về kích thước 100x100, vì vậy tâm ảnh nằm ở tọa độ O(50, 50). Để đếm được số pixel trắng trên các vòng tròn bán kính 1, 5, 10, 15, 20, 25, 30, 35, 40. Thực hiện quét trên từng vòng tròn bán kính r và góc quét . Mỗi bước quét cách nhau 1 góc rad. Tọa độ của các pixel trên đường tròn được xác định bằng công thức

Toàn bộ code thực hiện nhiệm vụ a) và b):

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

Kết quả số pixel trắng trên từng đường tròn được ghi vào trong file result.txt:

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

b) Đặc trưng của các hình được xây dựng từ câu a), chính là vector chứa thông tin các pixel trắng trên từng đường tròn bán kính khác nhau. Từ đó dùng Normalized Cross-Correlation để phân biệt giữa các ảnh. Với 2 vector x, y có độ dài n

Xét ngưỡng nếu thì kết luận 2 ảnh giống, ngược lại thì kết luận 2 ảnh khác nhau.

A black background with white text

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

# **V. Bài 5**

**Extract color of lemon and determine the good, bad and acceptable lemon (2 marks)**

Nhận thấy giữa các quả chanh có sự khác nhau về độ chín, cụ thể là quả chín sẽ có màu ngã vàng, quả non sẽ ngã xanh lá nhiều hơn. Nên nhóm quyết định sử dụng **RGB – HSV** để nhận biết và phân loại quả chanh

**Tính sơ bộ HSV trên quả A**

Chọn 3 điểm trên quả A, ta được các giá trị RGB như sau:



Vì R có giá trị lớn nhất nên ta dùng công thức:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **R** | **G** | **B** | **H** |
| **1** | 234 | 203 | 45 | 51,245 |
| **2** | 232 | 201 | 43 | 50,159 |
| **3** | 233 | 201 | 41 | 50 |

Trung bình giá trị H sơ bộ ở quả A là: 50,468

**Tính HSV trên quả C**

Chọn 3 điểm trên quả A, ta được các giá trị RGB như sau:



Vì G có giá trị lớn nhất nên ta dùng công thức:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **R** | **G** | **B** | **H** |
| **1** | 193 | 200 | 45 | 62,71 |
| **2** | 175 | 190 | 46 | 66,25 |
| **3** | 179 | 189 | 47 | 64,225 |

Trung bình giá trị H sơ bộ ở quả B là: 64,4

**Tính sơ bộ HSV trên quả C**

Chọn 3 điểm trên quả C, ta được các giá trị RGB như sau:



Vì R có giá trị lớn nhất nên ta dùng công thức:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **R** | **G** | **B** | **H** |
| **1** | 148 | 172 | 40 | 70,9 |
| **2** | 162 | 187 | 47 | 70,71 |
| **3** | 122 | 142 | 31 | 70,81 |

Trung bình giá trị H sơ bộ ở quả C là: 70,8

Do đó ta đặt ngưỡng:

* Good đối với quả chín vàng trong khoảng 40-55
* Acceptable đối với quả chín vừa trong khoảng 56-65
* Bad đối với quả non trong khoảng 66 trở lên

A computer screen with text and numbers

AI-generated content may be incorrect.

Sau đó tạo mask để loại bỏ phần background màu tối.

Tính giá trị mean RBG của từng ảnh.

Dùng giá trị mean RBG để tính được Hue của từng ảnh.

Cuối cùng dùng giá trị Hue để phân loại quả chanh như đã nêu ở trên.

**Kết quả đạt được:**

A screenshot of a computer

AI-generated content may be incorrect.