ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

ĐỒ ÁN MÔN HỌC XỬ LÝ ẢNH

Đề tài: Nhận diện tiền mặt Việt Nam Đồng

(VND Currency Recognition)

NHÓM 3

Thành viên tham gia:	MSSV
1. Nguyễn Tiến Dũng	19020531
2. Lê Minh Đức	21020305
3. Phạm Minh Hiếu	21020320
4. Cao Việt Hoàng	19020819
5. Phan Mạnh Thắng	21020405

Hà Nội, năm 2023.

I. Phương pháp thực hiện:

Sử dụng OPENCV + Machine Learning

Các bước chính:

1) Chuẩn bị dữ liệu: Hình ảnh các tờ tiền

2) Tạo mẫu tiền (template)

3) Tạo mô hình nhận diện

II. Phương pháp đánh giá:

Sử dụng các hình ảnh của các tờ tiền mà chưa được sử dụng để training để nhận diện, có thể là:

+ Các tờ tiền với chất lượng khác (nhàu hơn, mờ hơn, rõ nét hơn, …)

+ Chụp ở góc độ khác

+ Độ sáng khác nhau

→ Nhận diện được càng nhiều ảnh mới thì độ chính xác càng cao

III. Các bước thực hiện

Các thư viện được sử dụng: opencv, math, numpy, glob, pyplot (từ matplotlib)

```
import cv2
import numpy as np
import math
import glob
from matplotlib import pyplot as plt
```

1. Chuẩn bị dữ liệu

Chuẩn bị hình ảnh các mẫu tiền hiện hành của Việt Nam. Chụp ảnh các tờ tiền ở các góc khác nhau, các độ sáng khác nhau.

2. Tạo mẫu tiền (template) (+ training)

Tạo các mẫu tiền **cơ bản** bằng cách tải ảnh các mẫu tiền hiện hành (gồm mặt trước và mặt sau, phẳng, đặt thẳng góc) (500, 1000, 2000, 5000, 10000, 20000, 50000, 100000, 200000, 500000 Đồng) dưới dạng **ảnh xám** (có thể resize lại khích thước phù hợp).

→ Các mẫu tiền tăng lên trong quá trình nhận diện (bước training): Với mỗi hình ảnh tiền mà mô hình chưa nhận diện được sẽ được chuyển sang ảnh xám (có thể được resize) và đưa vào tập template.

Tên của mẫu tiền trùng với giá trị đồng tiền (Các đồng tiền có giá trị giống nhau thì phân biệt bằng cách thêm các dấu cách " " phía sau).

Ví dụ:

20000 đồng:

(20000.png)

(20000 .png)

(20000 .png)

(20000 .png)

3. Thực hiện tạo mô hình nhận diện

Bước 1: Load các hình ảnh mẫu, và ghi giá trị vào một mảng (tên là money_value)

Giá trị của các biến mẫu trong mảng được lấy bằng cách tách ra từ tên của các hình ảnh mẫu (sử dụng split)

```
files = glob.glob("template\\*.png")
print(files)
for item in files:
    img = cv2.imread(item, 0)
    money_template.append(img)
    val = item.split("\\")[-1] # sau dấu \\
    val = val.split(".")[0] # trước dấu .
    money_value.append(val)
```

Xem giá trị của các biến mẫu và ảnh mẫu:

```
for i in range(len(money_template)):
    print(money_value[i])
    plt.subplot(1,len(money_template),i+1),plt.imshow(money_template[i],'gray')
    plt.xticks([]),plt.yticks([])
    plt.show()
```

Kết quả:

1000

1000 10000

1000 100000

1000 100000

10000 100000

, …

Bước 2: Tải ảnh cần kiểm tra vào dưới dạng ảnh xám

```
img = cv2.imread("VND Image\\20000(2).jpg", 0)
plt.subplot(1,1,1),plt.imshow(img,'gray')
plt.title("input image")
plt.xticks([]),plt.yticks([])
plt.show()
```

Bước 3: Tìm hình ảnh tờ tiền trong ảnh đưa vào và lấy ra hình ảnh tờ tiền

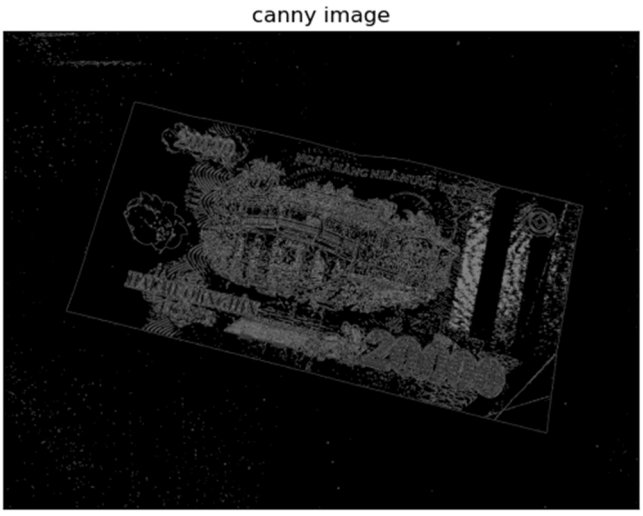
1) Làm mờ ảnh đưa vào bằng bộ lọc trung bình

```
img_blur = cv2.blur(img, (5,5))
plt.subplot(1,1 , 1),plt.imshow(img_blur,'gray')
plt.title("blur image")
plt.xticks([]),plt.yticks([])
plt.show()
```

2) Thực hiện tìm cạnh (canny)

Với threshold1=30, threshold2=40.

```
img_canny = cv2.Canny(img_blur,30, 40)
plt.subplot(1,1 , 1),plt.imshow(img_canny,'gray')
plt.title("canny image")
plt.xticks([]),plt.yticks([])
plt.show()
```



3) Close hình ảnh với kernel (30,30)

```
kernel = np.ones((30,30),np.uint8)
img_closed = cv2.morphologyEx(img_canny, cv2.MORPH_CLOSE, kernel)
plt.subplot(1,1 , 1),plt.imshow(img_closed,'gray')
plt.title("closed image")
plt.xticks([]),plt.yticks([])
plt.show()
```

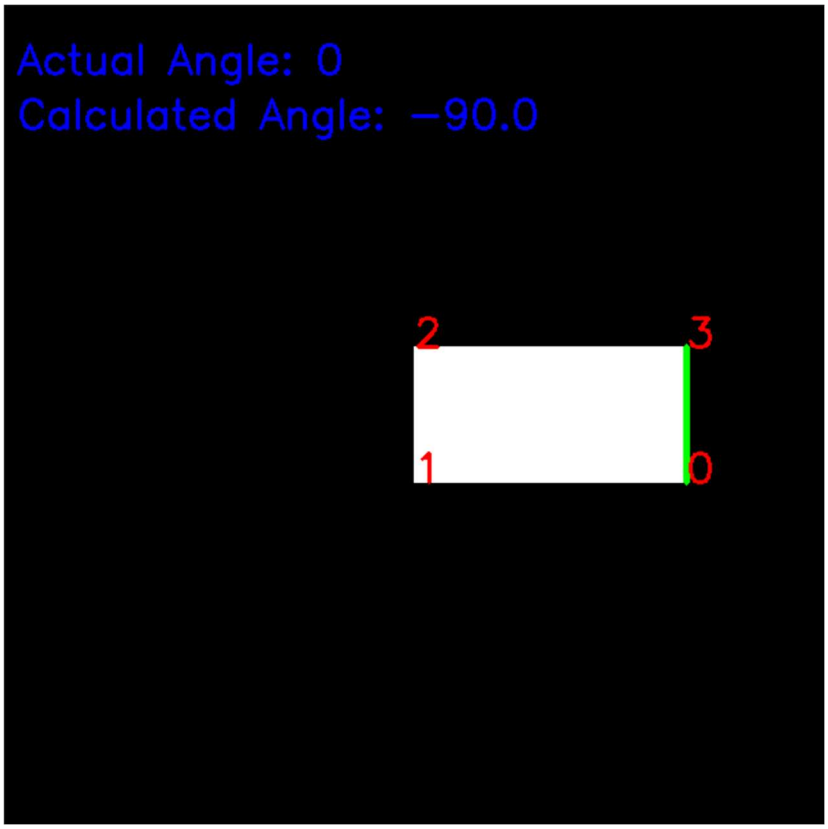
4) Tìm contours có kích thước lớn nhất là đồng tiền

```
contours, _ = cv2.findContours(img_closed,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
cnt = None
s_max_large = 0
for item in contours:
    s = cv2.contourArea(item)
    if(s > s_max_large):
        s_max_large = s
        cnt = item
```

5) Tìm hình chữ nhật nhỏ nhất bound lấy đồng tiền

```
rect = cv2.minAreaRect(cnt)
box = cv2.boxPoints(rect)
box = np.int0(box)
imgcp = img.copy()
im = cv2.drawContours(imgcp,[box],0,(127),10)
plt.subplot(1,1 , 1),plt.imshow(im,'gray')
plt.title("detected image")
plt.xticks([]),plt.yticks([])
plt.show()
```

6) Thực hiện tìm góc lệch và xoay ảnh lại cho đồng tiền nằm ngang




```
#xác định tâm xoay
M = cv2.moments(cnt)
cx = int(M['m10']/M['m00'])
cy = int(M['m01']/M['m00'])
box = cv2.boxPoints(rect)
box = np.int0(box)
#xoay ảnh
angle = rect[2]
RM = cv2.getRotationMatrix2D((cx,cy),angle,1)
# tính độ dài chiều dọc (dist1) và chiều ngang(dist2).
# nếu chiều dọc > chiều ngang đồng tiền nằm dọc khi đó cần cộng thêm 90 độ xoay
dist1 = int(math.sqrt((box[1][0]  - box[0][0])**2 + (box[1][1]  - box[0][1])**2))
dist2 = int(math.sqrt((box[3][0]  - box[0][0])**2 + (box[3][1]  - box[0][1])**2))
if(dist1 > dist2):
    angle -= 90
rows, cols = img.shape
# thực hiện xoay lại ảnh
RM = cv2.getRotationMatrix2D((cx,cy),angle,1)
img_rotated = cv2.warpAffine(img,RM,(cols,rows))
plt.subplot(1,1 , 1),plt.imshow(img_rotated,'gray')
plt.title("rotation image")
plt.xticks([]),plt.yticks([])
plt.show()
```

→ Thực hiện lại từ bươc 1 → bước 5 để tìm đồng tiền với bức ảnh đồng tiền nằm ngang

```
img_blur2 = cv2.blur(img_rotated, (5,5))
img_canny2 = cv2.Canny(img_blur2,30, 40)
kernel2 = np.ones((30,30),np.uint8)
img_closed2 = cv2.morphologyEx(img_canny2, cv2.MORPH_CLOSE, kernel2)
contours2, _ = cv2.findContours(img_closed2,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
cnt2 = None
s_max_large2 = 0
for item in contours2:
    s = cv2.contourArea(item)
    if(s > s_max_large2):
        s_max_large2 = s
        cnt2 = item
```

7) Tìm hình chữ nhật bound contours lớn nhất và cắt hình ảnh với hình chữ nhật này

```
x,y,w,h = cv2.boundingRect(cnt2)
img_money = img_rotated[y:y+h, x:x+w]
plt.subplot(1,1 , 1),plt.imshow(img_money,'gray')
plt.title("Money Image")
plt.xticks([]),plt.yticks([])
plt.show()
```

Bước 4: Thực hiện so khớp (bằng phép trừ ảnh) với từng ảnh mẫu. Nếu ảnh nào khớp nhất thì đồng tiền hiện tại có giá trị cùng với ảnh mẫu đó

1) Đặt 1 giá trị đếm pixel lớn hơn diện tích của hình ảnh đồng tiền

2) Làm mờ ảnh đồng tiền

3) Duyệt lần lượt các ảnh mẫu tiền đã load vào mảng để so khớp với ảnh đồng tiền cần nhận diện:

+ Thực hiện resize ảnh đồng tiền về cùng kích thước với ảnh mẫu

+ Làm mờ ảnh mẫu

+ Cân bằng sáng ảnh mẫu và ảnh đồng tiền (Histogram Equalization) (không in ra histogram)

+ Thực hiện phép trừ ảnh

+ Threshold hình ảnh kết quả của phép trừ ảnh, thực hiện đếm các điểm sáng (là các điểm không khớp)

→ Giá trị này càng nhỏ thì càng khớp

→ Giá trị của đồng tiền là giá trị ứng với hình ảnh mẫu có độ khớp lớn nhất

```
val = ""
# đặt 1 giá trị đếm pixel vô cùng lớn(lớn hơn diện tích của hình ảnh đồng tiền)
min_count = 500*500
# thực hiện làm mờ ảnh kiểm tra
img_det_blur = cv2.blur(img_money, (5,5))
for i , im in enumerate(money_template):
    # thực hiện resize kích thước ảnh kiểm tra về cùng với ảnh mẫu để có thể trừ
    h,w = im.shape # lấy kích thước của ảnh mẫu
    im_rz = cv2.resize(img_det_blur, (w,h))
    # làm mờ ảnh mẫu
    img_ref_blur = cv2.blur(im, (5,5))
    # cân bằng sáng ảnh mẫu và ảnh kiểm tra để tránh ánh sáng quá sáng hoặc quá tối
    eq_ref = cv2.equalizeHist(img_ref_blur)
    eq_def = cv2.equalizeHist(im_rz)
    # thực hiện trừ ảnh
    sub_img = cv2.subtract(eq_ref, eq_def)
    # phân ngưỡng và thực hiện đếm điểm không khớp (điểm sáng là điểm không khớp)
    _, img_threshold = cv2.threshold(sub_img, 0, 255, cv2.THRESH_OTSU)
    count = cv2.countNonZero(img_threshold)
    # giá trị này càng nhỏ thì độ khớp càng lớn
    if(count  < min_count):
        min_count = count
        val = money_value[i]
```

Bước 5: Đồng tiền có thể nằm ngược nên xoay 180 độ và thực hiện so khớp lại (Bước 4) 1 lần nữa

```
# xoay ảnh 180 độ
rows, cols = img_det_blur.shape
M = cv2.getRotationMatrix2D((int(cols/2),int(rows/2)),180,1)
img_det_blur = cv2.warpAffine(img_det_blur,M,(cols,rows))
 #thực hiện so khớp lại lần 2
for i , im in enumerate(money_template):
    h,w = im.shape
    im_rz = cv2.resize(img_det_blur, (w,h))
    img_ref_blur = cv2.blur(im, (5,5))
    eq_ref = cv2.equalizeHist(img_ref_blur)
    eq_def = cv2.equalizeHist(im_rz)
    sub_img = cv2.subtract(eq_ref, eq_def)
    _, img_threshold = cv2.threshold(sub_img, 0, 255, cv2.THRESH_OTSU)
    count = cv2.countNonZero(img_threshold)
    if(count  < min_count):
        min_count = count
        val = money_value[i]
    #cv2.imwrite("out_jupyter2//" + "lan2 - " + money_value[i] + ".png", sub_img)
```

Bước 6: In ra giá trị đồng tiền

```
print(val)
```

Kết quả: 20000