Bài giảng môn học:

Thị giác máy tính (7080518)

CHƯƠNG 2:
XỬ LÝ VÀ NÂNG CAO CHẤT LƯỢNG ẢNH
(Phần 1)

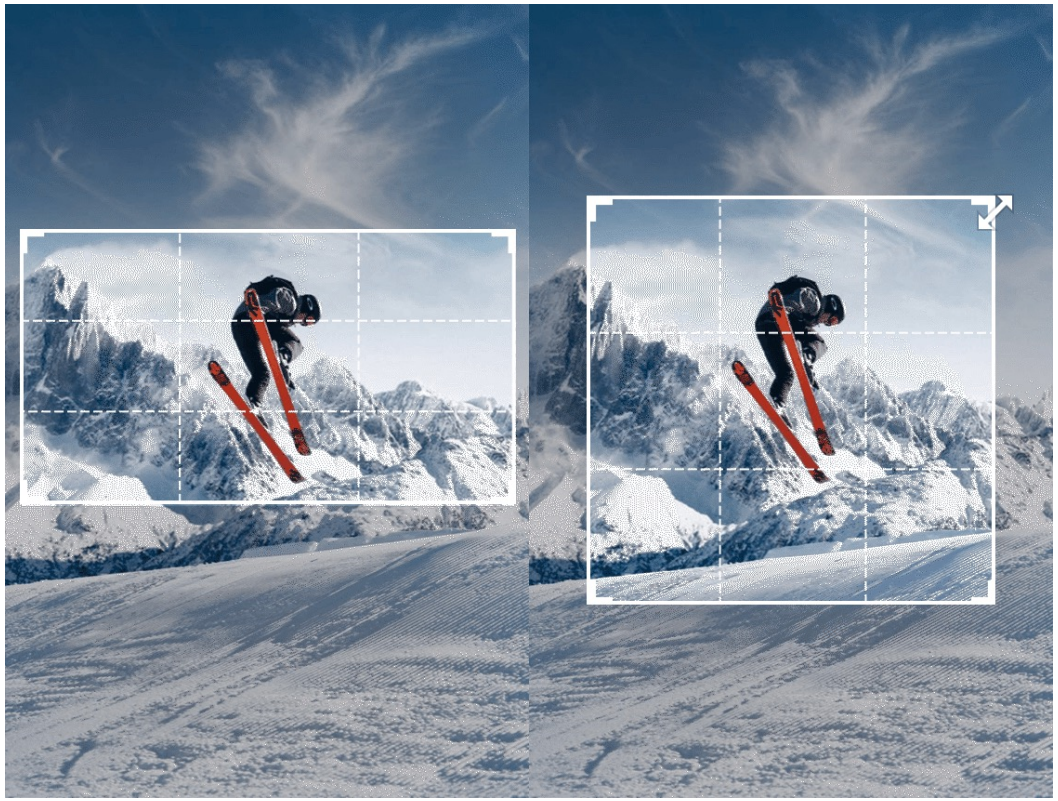
Đặng Văn Nam

dangvannam@humg.edu.vn

Nội dung chương 2 – phần 1

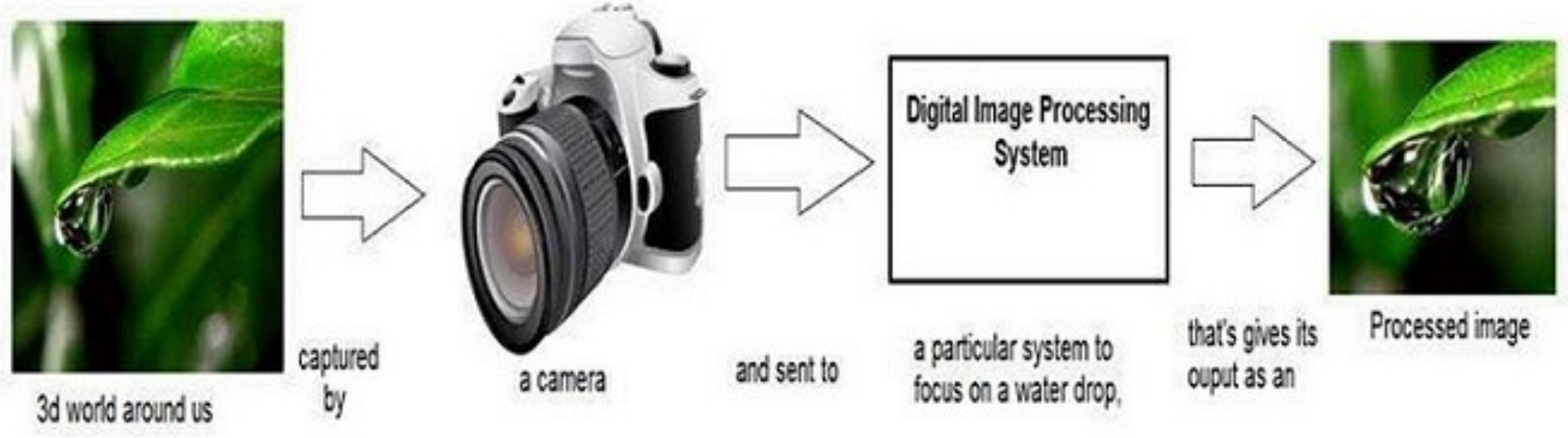
I/ Một số kỹ thuật xử lý ảnh cơ bản

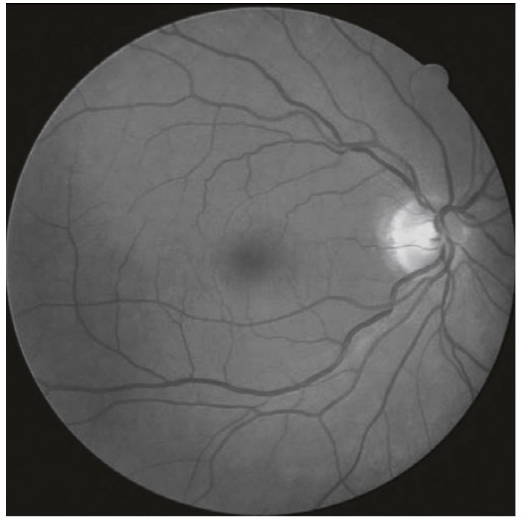
1. Thay đổi kích thước ảnh (Resizing images)
2. Cắt ảnh (Cropping images)
3. Xoay ảnh (Rotating images)
4. Lật ảnh (Flipping images)



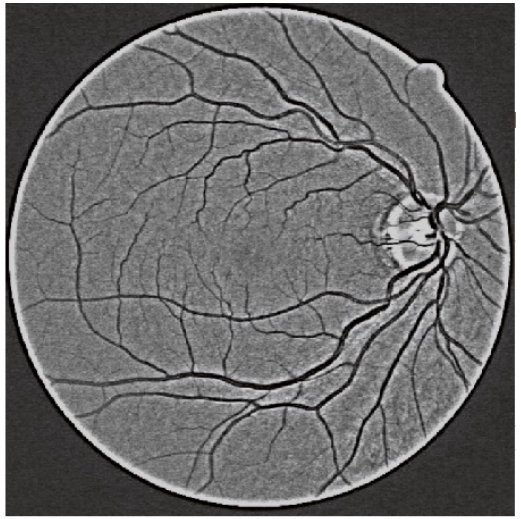
I. Một số kỹ thuật xử lý ảnh cơ bản

Một số kỹ thuật Xử lý ảnh cơ bản



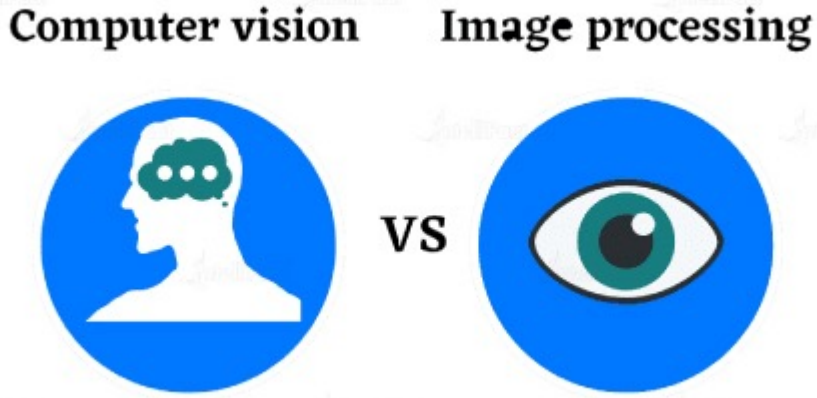


Original



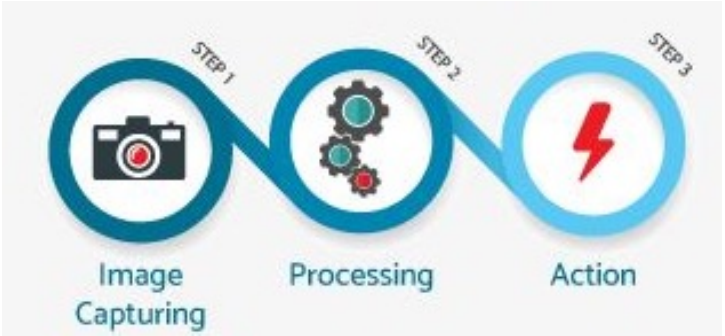
Enhanced with SUACE

Một số kỹ thuật Xử lý ảnh cơ bản

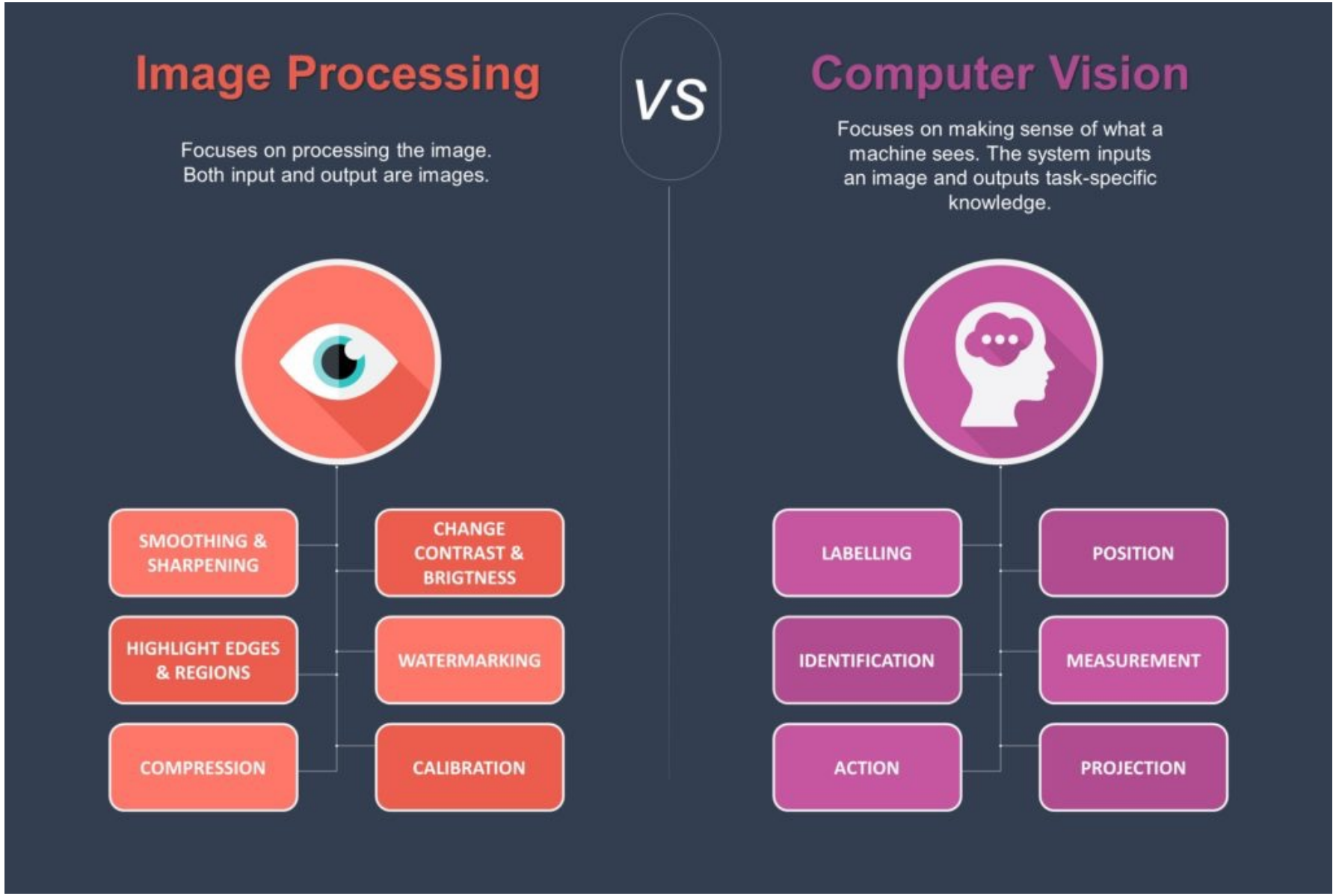


- **Xử lý ảnh (Image processing):** là quá trình tạo ra ảnh mới từ ảnh có sẵn, VD: tăng độ sáng, cắt ảnh, khử nhiễu,… không tập trung vào việc hiểu nội dung của bức ảnh.

- **Thị giác máy tính (Computer vision):** tập trung vào việc hiểu những gì máy móc nhìn thấy, sử dụng xử lý ảnh để xử lý dữ liệu thô (tiền xử lý dữ liệu ảnh).



Một số kỹ thuật Xử lý ảnh cơ bản

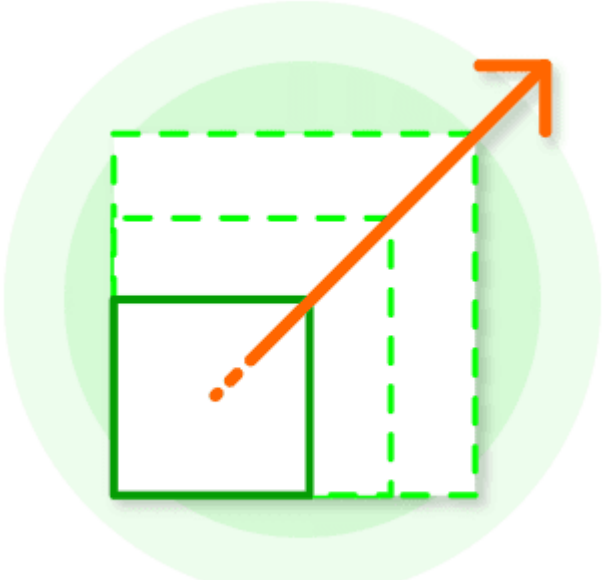


1. Thay đổi kích thước ảnh (Resize images)

1. Thay đổi kích thước ảnh

- Thay đổi kích thước (Zoom in, Zoom out) là một trong những kỹ thuật xử lý ảnh thông dụng.
- Ảnh có kích thước lớn sẽ chiếm nhiều bộ nhớ, số lượng tập dữ liệu ảnh thường rất lớn. Do đó cần giảm kích thước ảnh. Resize là một kỹ thuật phổ biến. => Trong OpenCV, Sử dụng hàm cv2.resize

Một số kích thước ảnh ML thường dùng: 32 × 32, 64 × 64, 96 × 96, 256 × 256



1. Thay đổi kích thước ảnh

Phương thức: cv2.resize(src, dsize, interpolation)

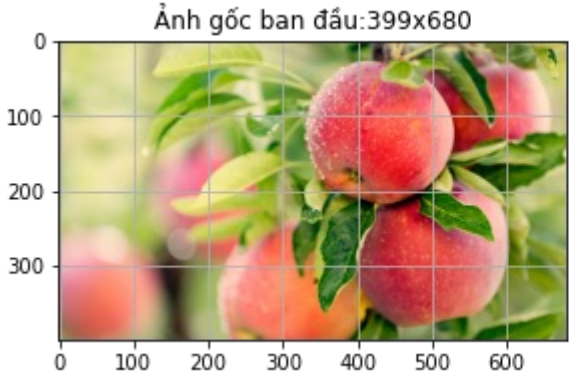
Ý nghĩa các tham số:

1.src: ảnh gốc cần thay đổi kích thước.

2.dsize (width, height): Kích thước ảnh muốn thay đổi

3.interpolation: chỉ định thuật toán thực hiện resize:

```
#Đọc ảnh màu:
img_orignal = cv2.imread('images/apples.jpg',1)

#Thay đổi kích thước ảnh về độ phân giải: 100 x 200
dsize = (200,100) #thứ tự (Width, height)

#Sử dụng phương thức resize():
img_100x200 = cv2.resize(img_orignal,dsize, interpolation=cv2.INTER_LINEAR)
```



1. Thay đổi kích thước ảnh

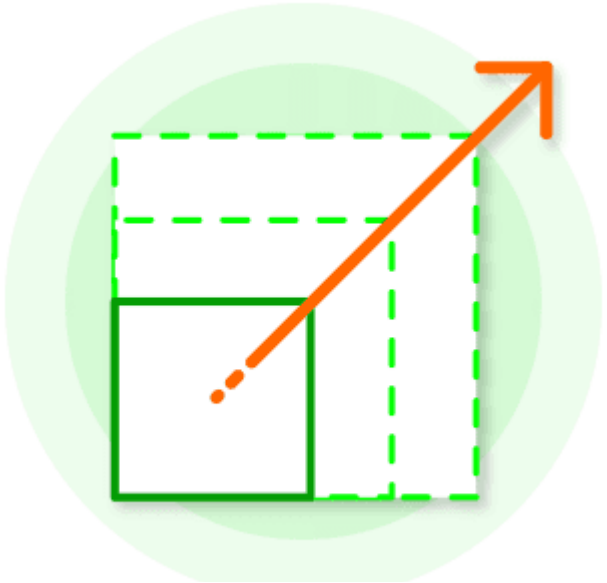
Một số giải thuật thay đổi kích thước ảnh trong OpenCV:

Các Phương pháp nội suy (interpolation) thực hiện resize ảnh:

- INTER_NEAREST – nội suy láng giềng gần nhất
- INTER_LINEAR – nội suy song tuyến tính (mặc định)
- INTER_AREA – resampling using pixel area relation. It may be a preferred method for image decimation, as it gives moire'-free results. But when the image is zoomed, it is similar to the INTER_NEAREST method.
- INTER_CUBIC – nội suy xoắn bậc 3, trên 4×4 pixel láng giềng
- INTER_LANCZOS4 – nội suy Lanczos trên 8×8 pixel láng giềng

Thông thường:

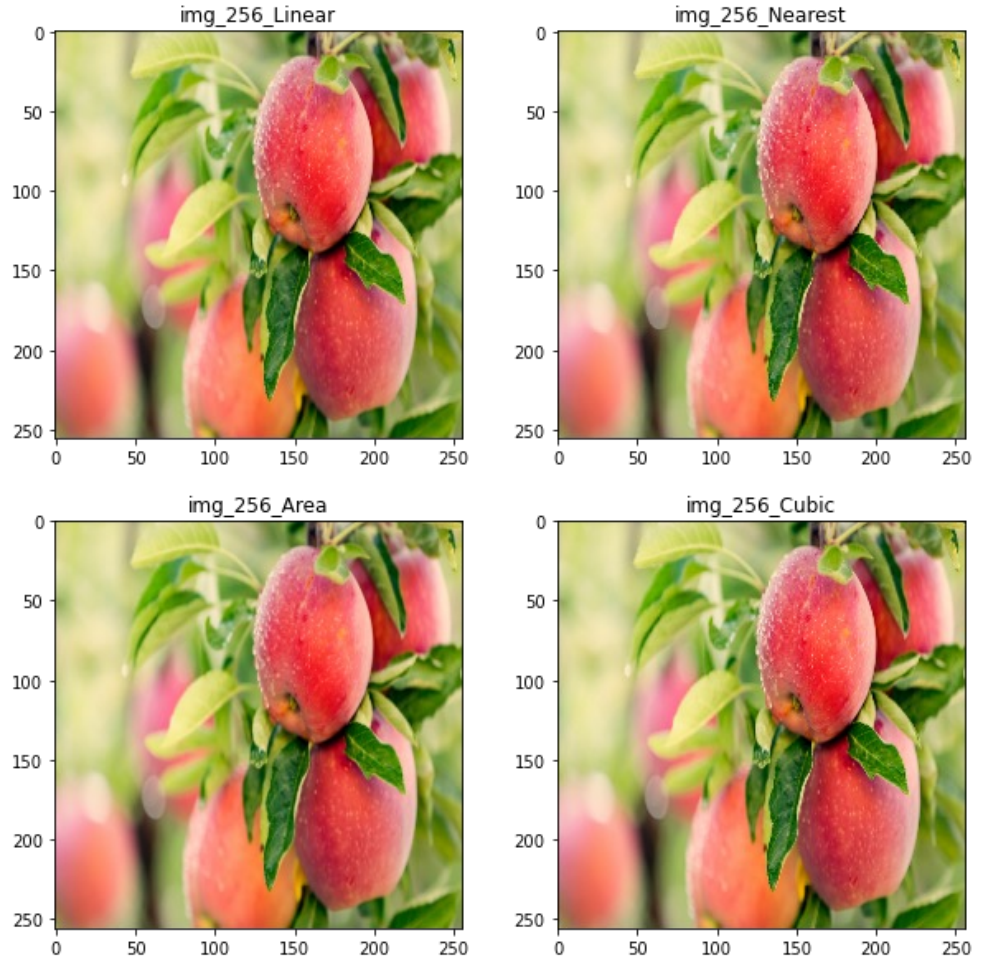
- cv.INTER_AREA để thu nhỏ
- cv.INTER_CUBIC & cv.INTER_LINEAR để phóng to.



1. Thay đổi kích thước ảnh

Một số giải thuật thay đổi kích thước ảnh trong OpenCV:

```python
#Resize ảnh với các thuật toán khác nhau:
dsize = (256,256)

#INTER_LINEAR (thuật toán mặc định):
img_256_Linear = cv2.resize(img_orignal,dsize,
                            interpolation=cv2.INTER_LINEAR)

#INTER_NEAREST:
img_256_Nearest = cv2.resize(img_orignal,dsize,
                             interpolation=cv2.INTER_NEAREST)

#INTER_AREA:
img_256_Area = cv2.resize(img_orignal, dsize,
                          interpolation=cv2.INTER_AREA)

#INTER_CUBIC
img_256_Cubic = cv2.resize(img_orignal,dsize,
                           interpolation=cv2.INTER_CUBIC)
```



1. Thay đổi kích thước ảnh

Thay đổi kích thước ảnh theo tỷ lệ phần trăm:

```
#Thay đổi kích thước ảnh theo tỷ lệ %
scale = 50 #Tỷ lệ thay đổi ảnh theo %:
#scale <100 thu nhỏ <---> scale>100 phóng to

#Tính toán chiều rộng và chiều cao ảnh theo tỷ lệ thiết lập
w_scale = int(w*scale/100)
h_scale = int(h*scale/100)

dsize = (w_scale,h_scale)

#Thực hiện resize ảnh theo scale:
img_scale = cv2.resize(img_orignal,dsize, interpolation=cv2.INTER_AREA)

print('Độ phân giải ảnh ban đầu:',h,'x',w)
print('Độ phân giải ảnh sau khi scale:',h_scale,'x',w_scale)
#Hiển thị ảnh:
plt.imshow(img_scale)
plt.title('Scale: ' + str(scale) + '%')
plt.show()
```

```
Độ phân giải ảnh ban đầu: 399 x 680
Độ phân giải ảnh sau khi scale: 199 x 340
```

Thực hành số 2.1

Thực hành 2.1

Yêu cầu 1: Sinh viên đọc ảnh màu images/Thuchanh_2_1.jpg và chuyển đổi sang hệ RGB, hiển thị ảnh

Yêu cầu 2: Cho biết chiều cao, chiều rộng và số pixel điểm ảnh.

Yêu cầu 3: Thực hiện thay đổi kích thước theo yêu cầu sau:

3.1. Thay đổi kích thước ảnh về 32x32, sử dụng tham số interpolation mặc định. Hiển thị kết quả và Lưu lại ảnh vào thư mục images/Saves/MSV_img_1.jpg theo hệ màu BGR

3.2. Giữ nguyên chiều cao của ảnh gốc, giảm chiều rộng ảnh đi một nửa, Sử dụng phương pháp INTER_NEAREST.Hiển thị kết quả và Lưu lại ảnh vào thư mục images/Saves/MSV_img_2.jpg theo hệ màu BGR

3.3. Tăng chiều cao của ảnh lên gấp đôi, giữ nguyên chiều rộng của ảnh gốc; Sử dụng phương pháp INTER_LANCZOS4. Hiển thị kết quả và Lưu lại ảnh vào thư mục images/Saves/MSV_img_3.jpg theo hệ màu BGR

Thực hành 2.1

Yêu cầu 4: Sinh viên sử dụng thư mục chứa các ảnh đã đổi tên thực hiện trong bài thực hành 1.3 chương 1.
Thực hiện thay đổi kích thước toàn bộ ảnh về độ phân giải 96x96 pixel. Lưu lại ảnh đã thay đổi vào thư mục mới có tên Pic_96 và đặt lại tên ảnh như sau: Pic_96_i.jpg với i là các số thứ tự của ảnh.

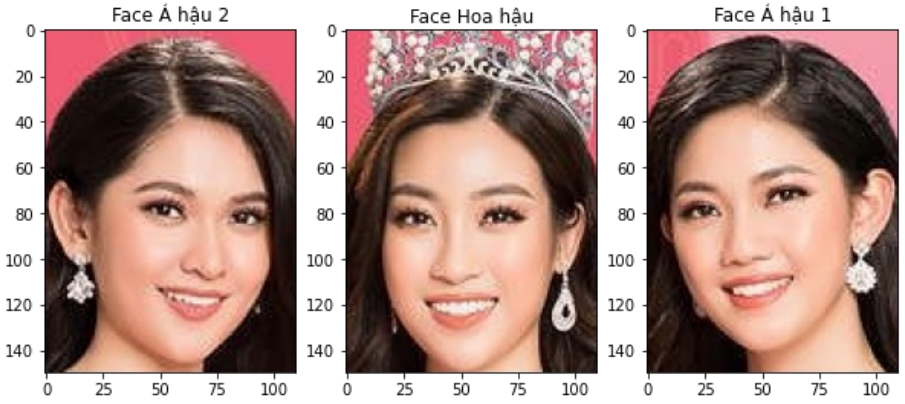
2. Cắt ảnh (Crop images)

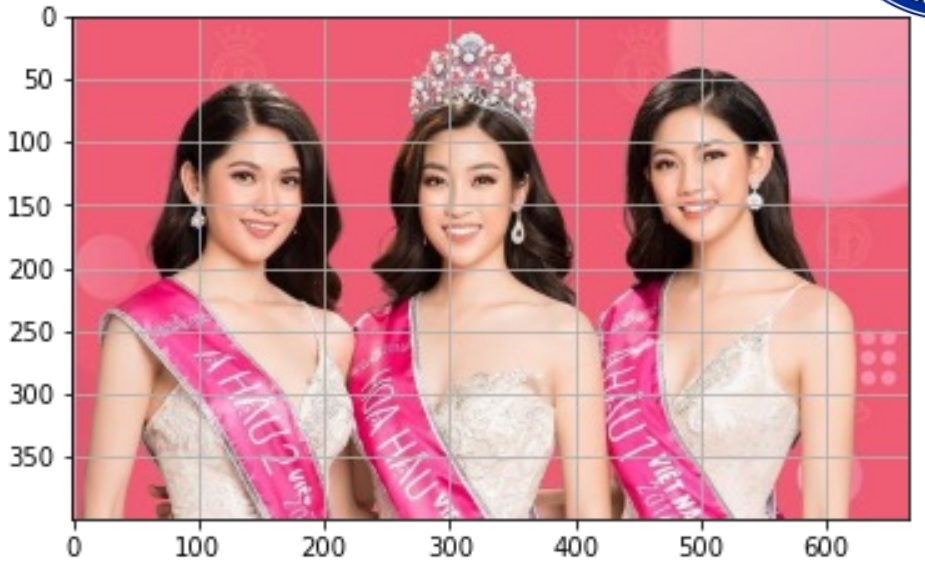
2. Cắt ảnh

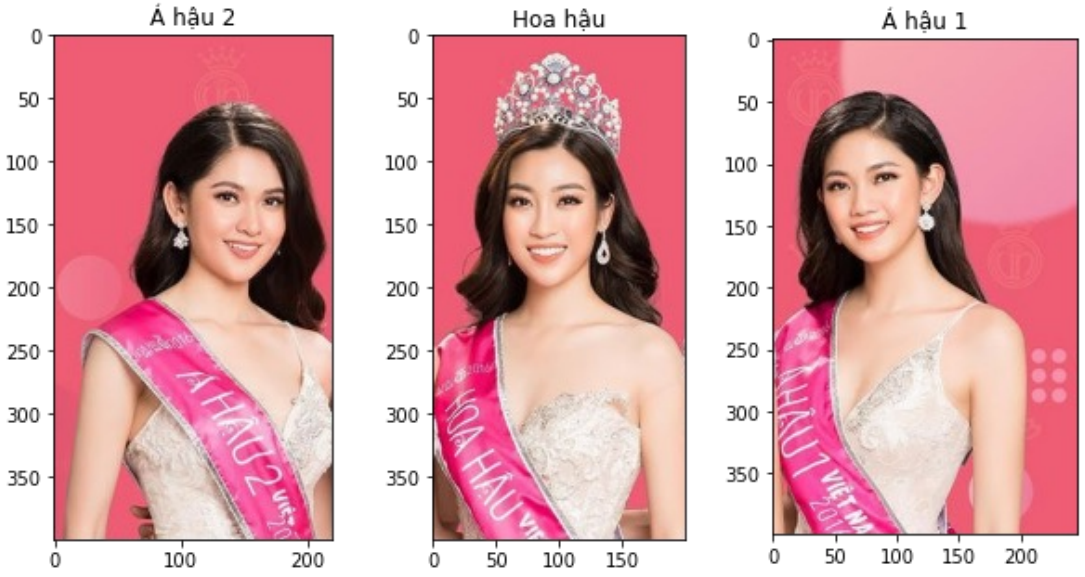
Thực hiện cắt một phần của ảnh, sử dụng phương pháp tách các phần tử trong ma trận:

img_cropped = img[start_row:end_row, start_col:end_col]

```
#Đọc ảnh màu:
img_miss = cv2.imread('images/MissVN.jpg')

#Cắt lấy từng người trong ảnh:
img_ahau2 = img_miss[:,0:220]
img_hoahau = img_miss[:,220:420]
img_ahau1 = img_miss[:,420:]

#Chỉ Cắt lấy khuôn mặt trong ảnh:
face_ahau2 = img_miss[50:200,85:195]
face_hoahau = img_miss[50:200,260:370]
face_ahau1 = img_miss[40:190,450:560]
```





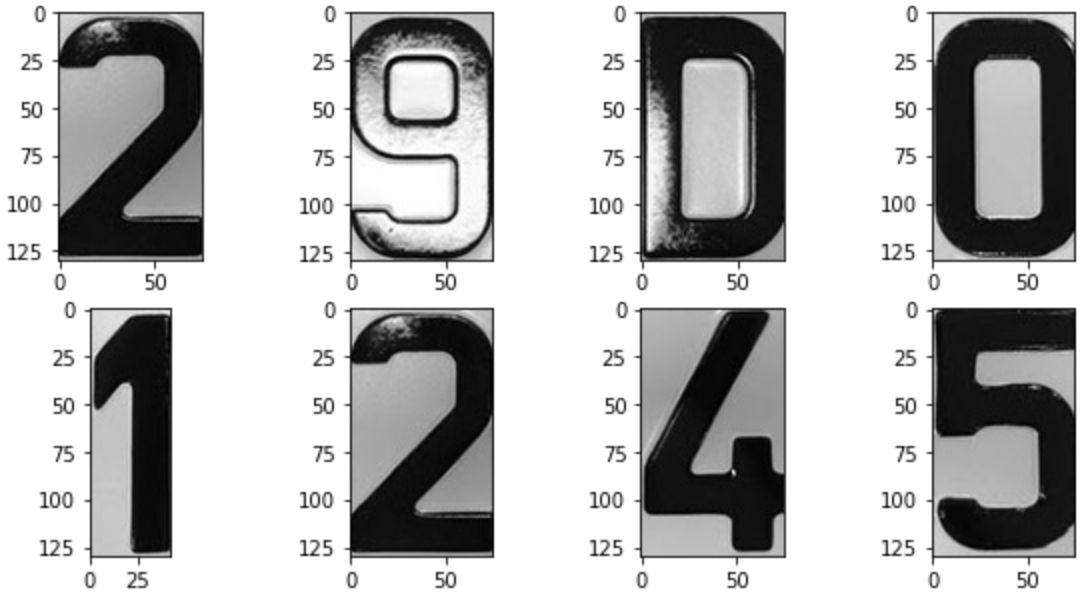


Thực hành số 2.2

Thực hành 2.2

Yêu cầu:

- Sinh viên đọc và hiển thị ảnh images/Thuchanh_2_2.jpg ở dạng ảnh xám.
- Cắt lấy từng con số và ký tự có trong ảnh, lưu ảnh đã cắt và thư mục images/Saves/ với tên tương ứng là MSV_kytutuongung.jpg



3. Xoay ảnh (Rotate Images)

3. Xoay ảnh

OpenCV hỗ trợ phương thức xoay ảnh:

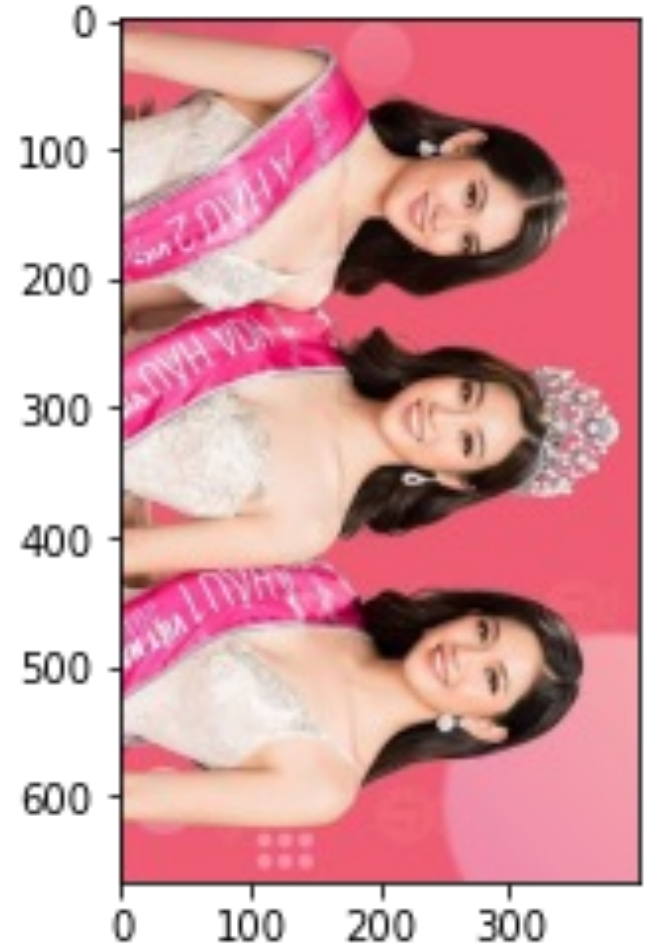
cv2.rotate(img, rotateCode)

Trong đó:

1. img: ảnh gốc muốn xoay
2. rotateCode: các chế độ xoay ảnh:
   * cv2.ROTATE_90_CLOCKWISE: Xoay 90 độ theo chiều kim đồng hồ
   * cv2.ROTATE_90_COUNTERCLOCKWISE: Xoay 90 độ ngược chiều kim đồng hồ
   * cv2.ROTATE_180: Xoay ảnh 180 (lật ảnh)

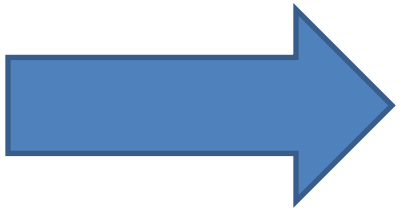
3. Xoay ảnh

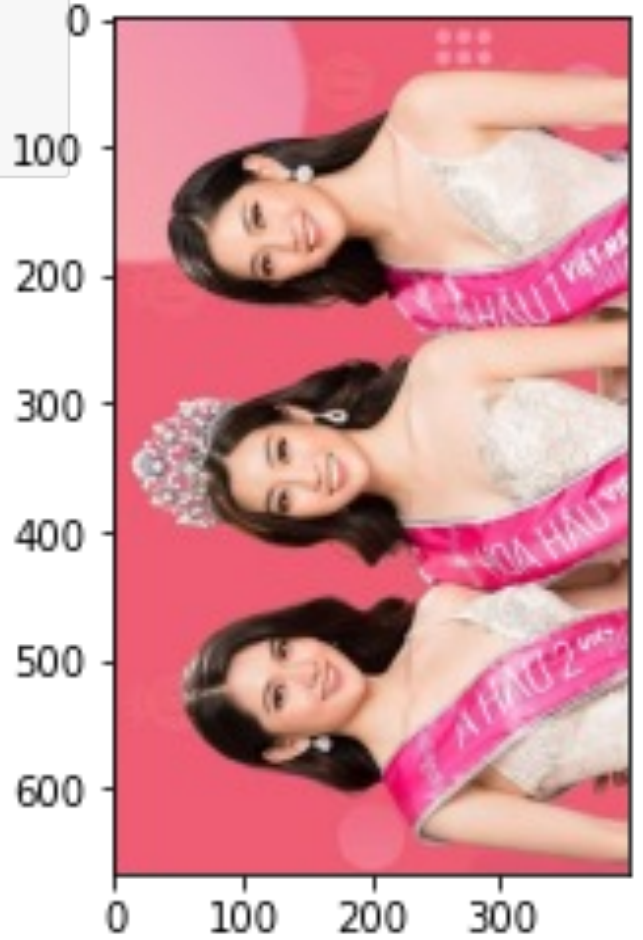
```
#Xoay ảnh 90 độ theo chiều kim đồng hồ:
img_rotate_90_clockwise = cv2.rotate(img_miss,
                                     cv2.ROTATE_90_CLOCKWISE)
#Hiển thị ảnh đã xoay:
plt.imshow(img_rotate_90_clockwise)
plt.show()
```



3. Xoay ảnh

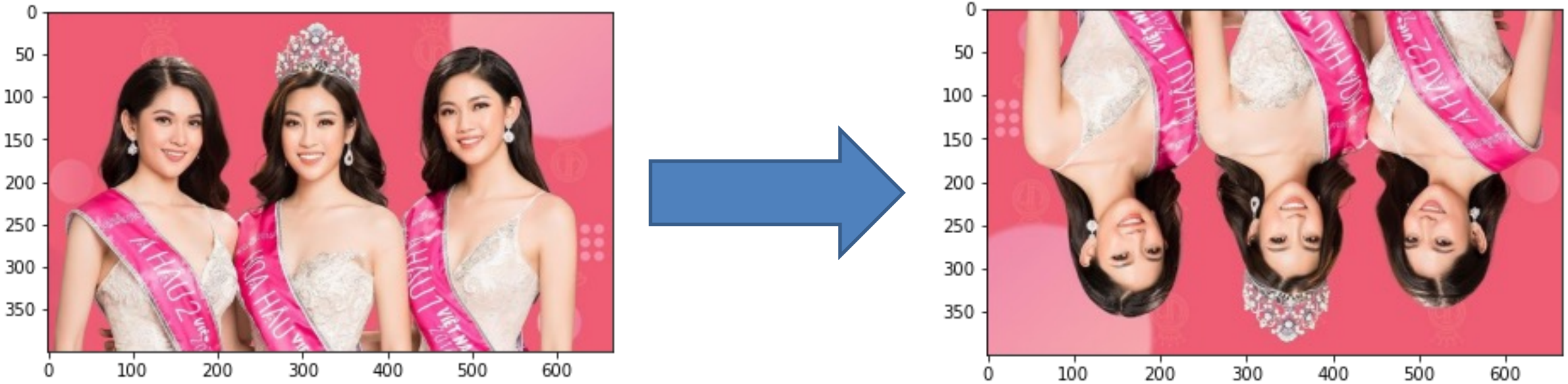
```
#Xoay ảnh 90 độ ngược chiều kim đồng hồ:
img_rotate_90_counterclockwise = cv2.rotate(img_miss,
                                            cv2.ROTATE_90_COUNTERCLOCKWISE)

#Hiển thị ảnh đã xoay:
plt.imshow(img_rotate_90_counterclockwise)
plt.show()
```



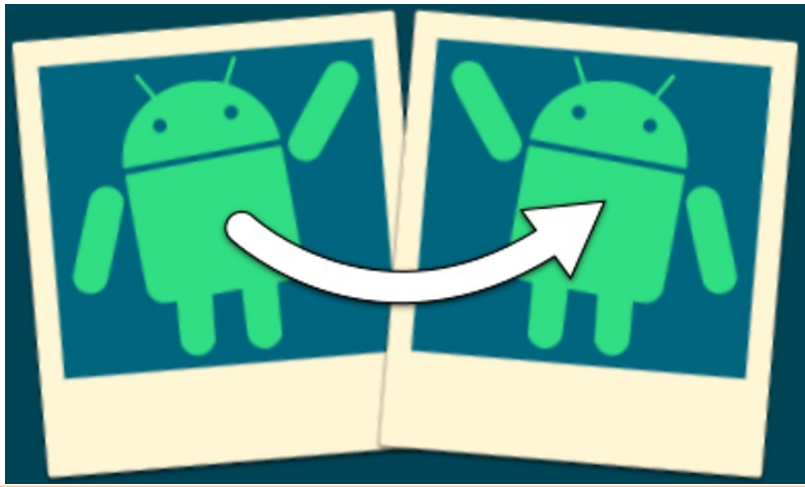


3. Xoay ảnh

```
1  #Xoay ảnh 180 độ:
2  img_rotate_180 = cv2.rotate(img_miss,
3                              cv2.ROTATE_180)
4
5  #Hiển thị đã xoay:
6  plt.imshow(img_rotate_180)
7  plt.show()
```

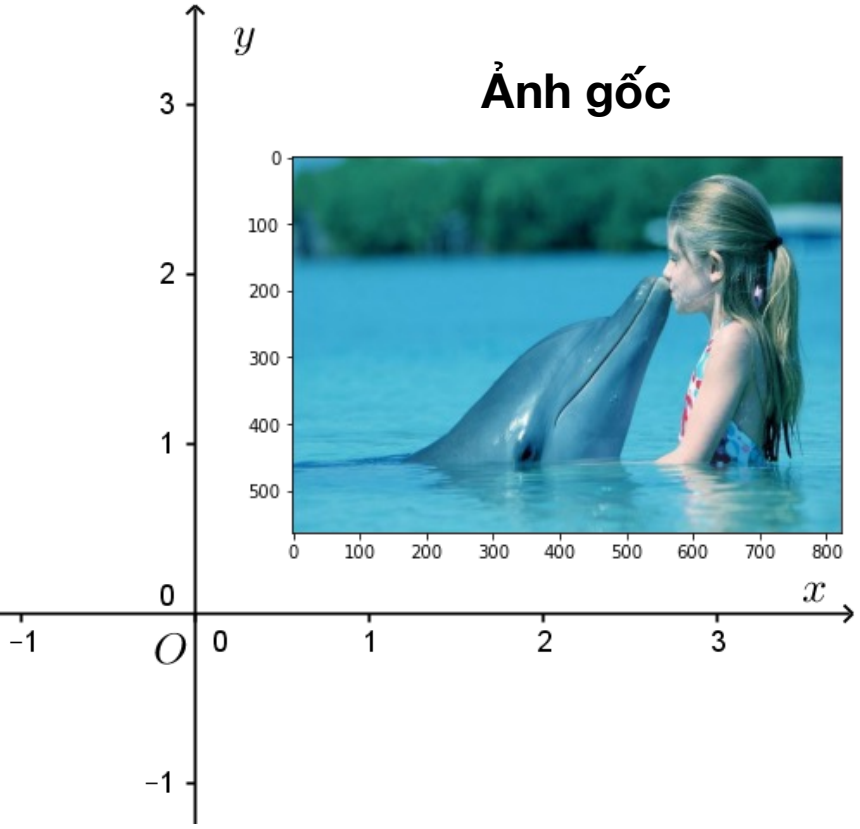


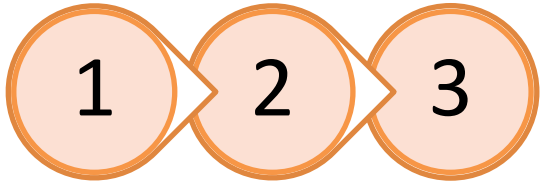
4. Lật ảnh (Flip Images)

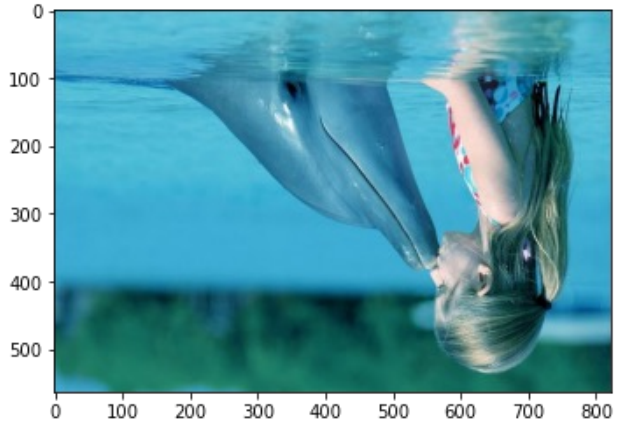


4. Lật ảnh

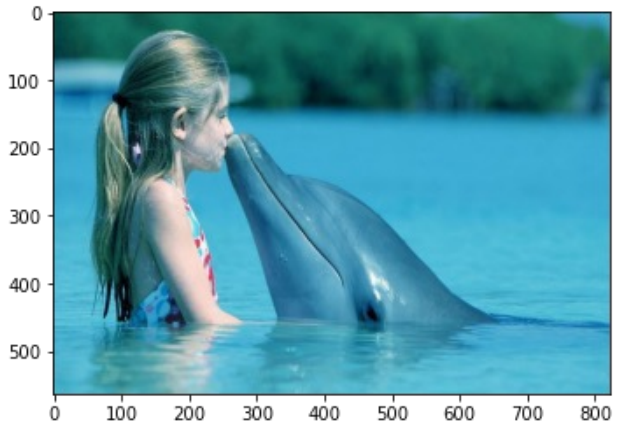
Lật ảnh cũng là một trong những kỹ thuật xử lý hình ảnh thông dụng. Có 3 kiểu lật bao gồm:



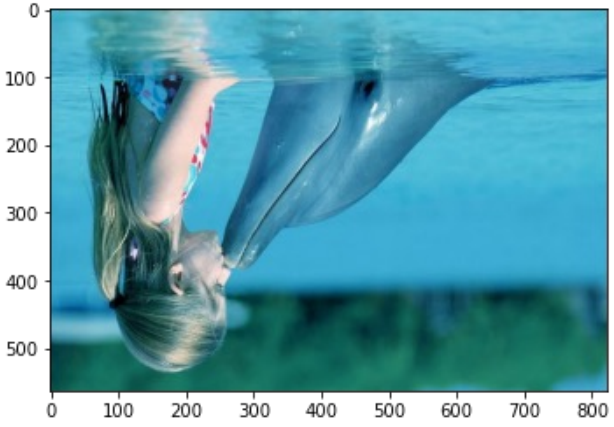




Lật theo chiều ngang (trục X)



Lật theo chiều dọc (trục Y)



Lật theo chiều dọc (Y) – ngang (X)

4. Lật ảnh

OpenCV hỗ trợ phương thức lật ảnh:

cv2.flip(img, flipCode)

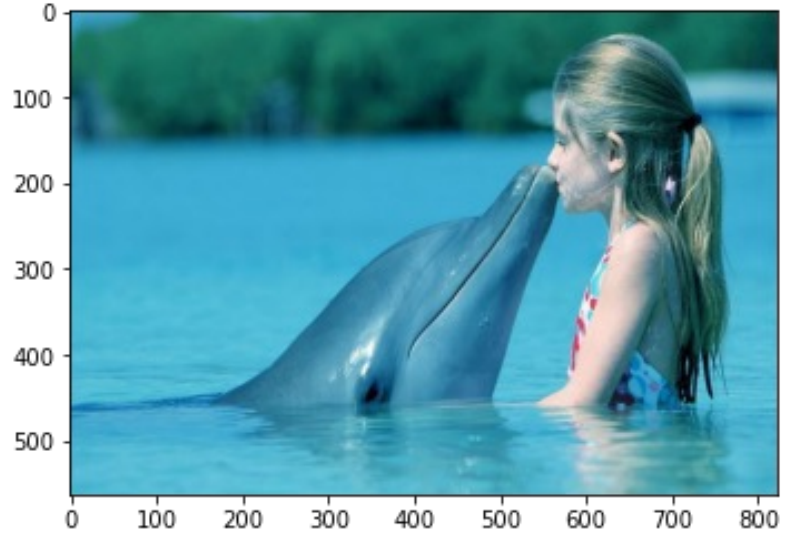
Trong đó:

1.img: ảnh gốc muốn lật.

2.flipCode: các chế độ lật ảnh:

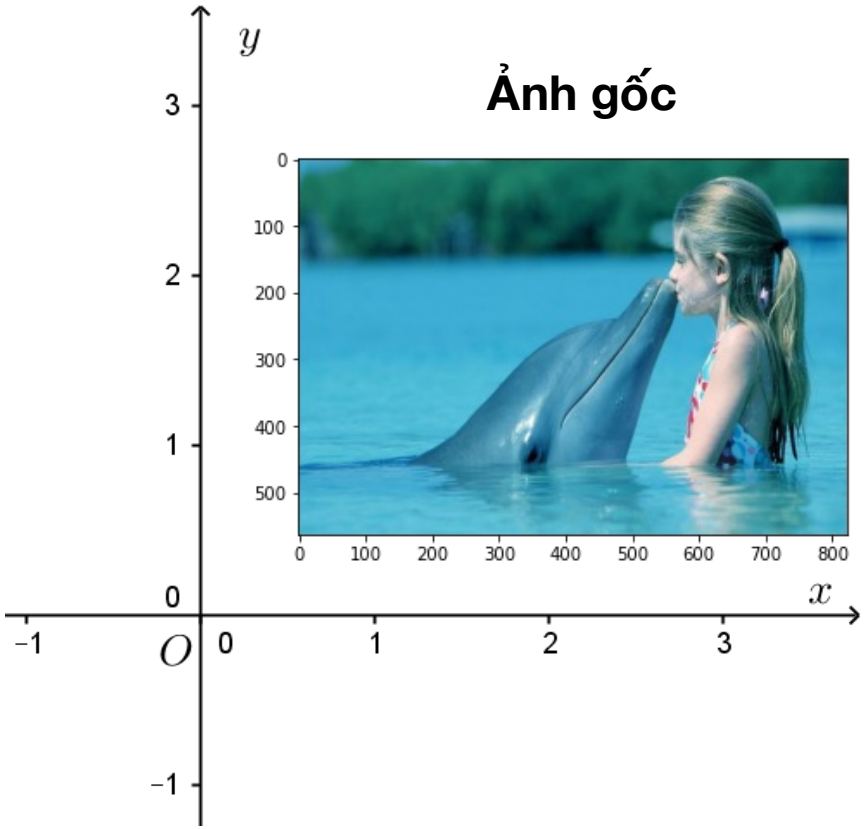
1. flipCode = 0: Lật ảnh theo chiều ngang (trục X)
2. flipCode > 0: Lật ảnh theo chiều dọc (trục Y)
3. flipCode < 0: Lật ảnh theo chiều dọc (Y) và ngang (X)

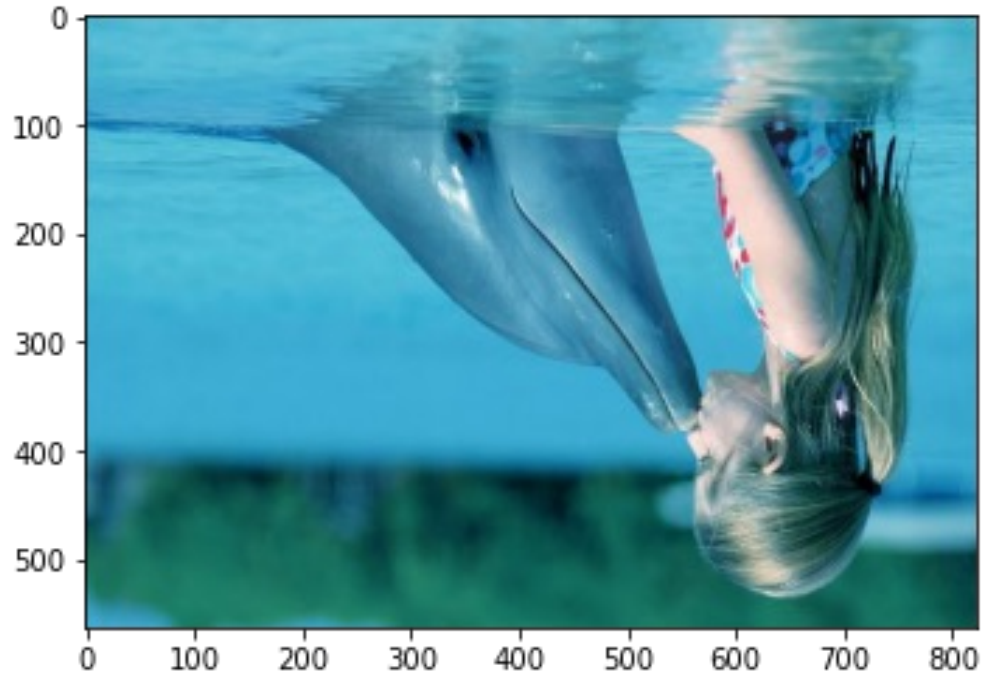
```
#Đọc ảnh màu:
img = cv2.imread('images/fish.jpg')

#Chuyển sang hệ màu RGB để hiện thị:
img_fish = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)

#Hiển thị ảnh:
plt.imshow(img_fish)
plt.show()
```



4. Lật ảnh

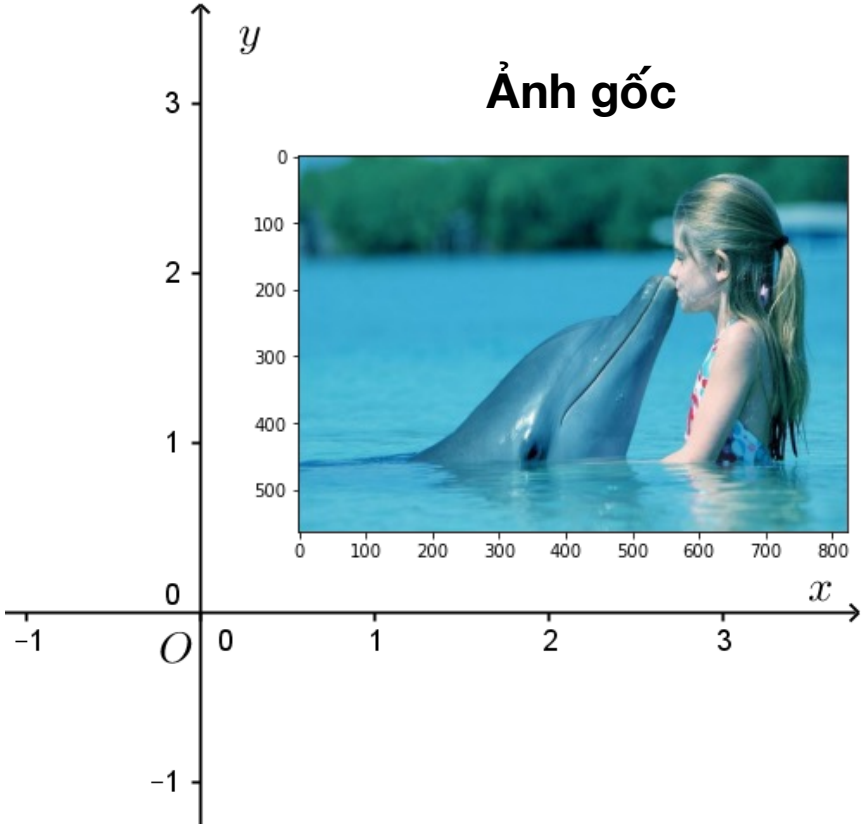
```
1  #Lật ảnh theo chiều ngang (trục X): flipCode = 0
2  img_flip_ud = cv2.flip(img_fish, 0)
3
4  #Hiển thị ảnh đã lật:
5  plt.imshow(img_flip_ud)
6  plt.show()
```

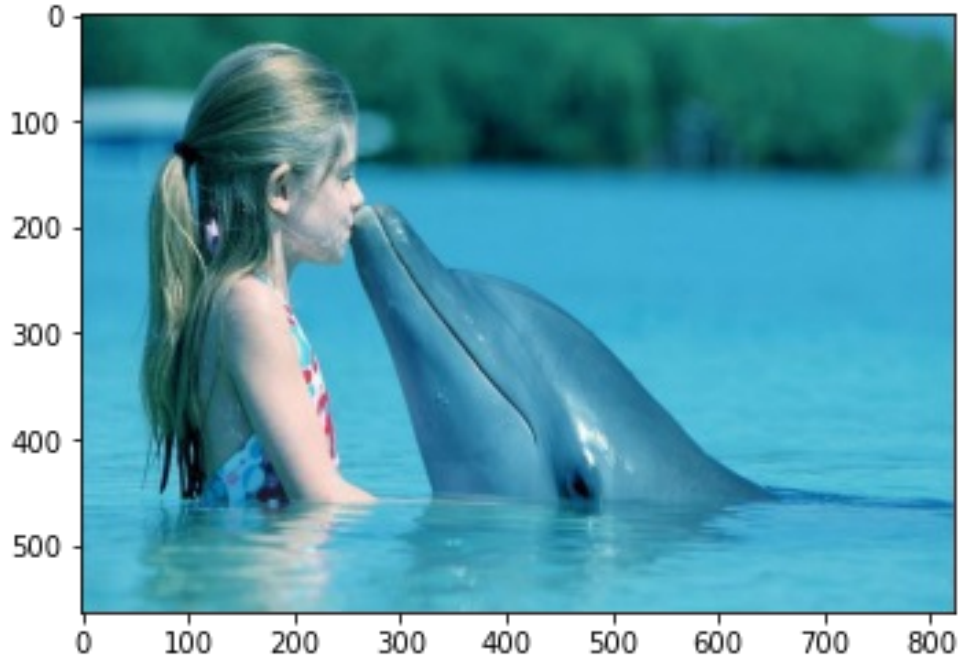




4. Lật ảnh

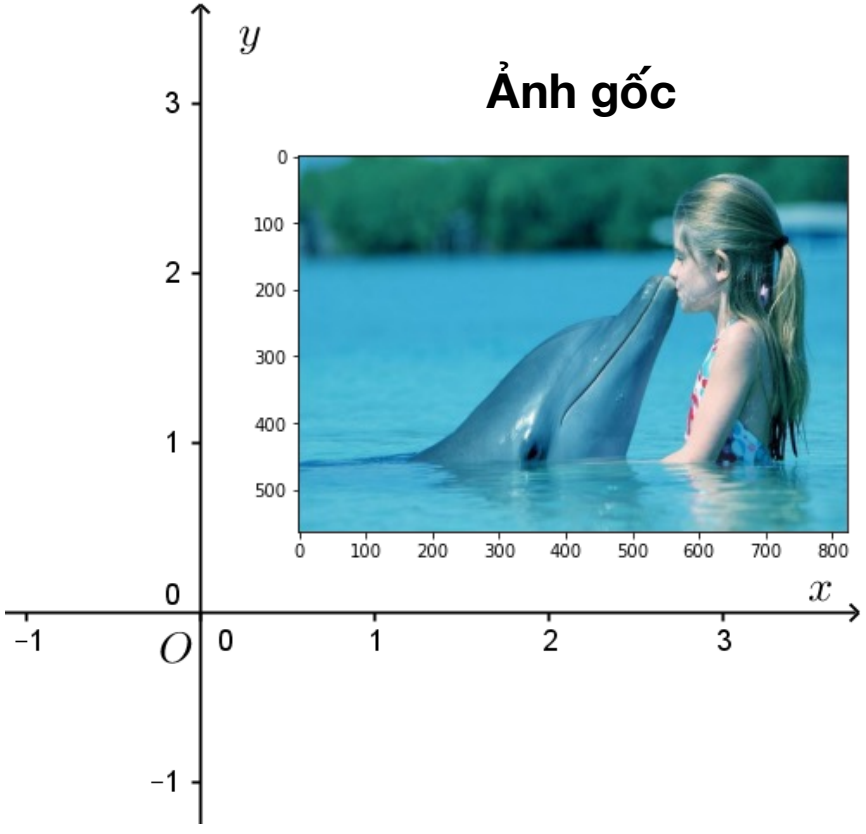
```
1 #Lật ảnh theo chiều dọc (trục Y): flipCode > 0
2 img_flip_lr = cv2.flip(img_fish, 1)
3
4 #Hiển thị ảnh đã lật:
5 plt.imshow(img_flip_lr)
6 plt.show()
```

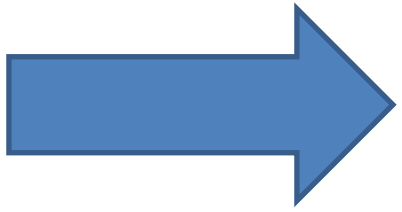


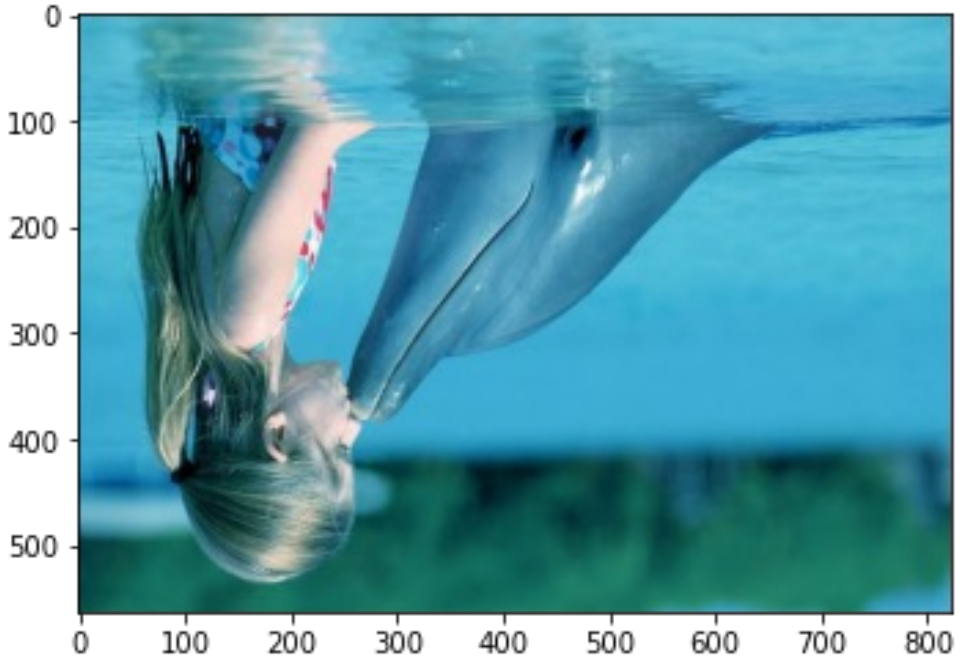


4. Lật ảnh

```
1 #Lật ảnh theo cả 2 chiều ngang, dọc (theo trục Y, X): flipCode < 0
2 img_flip_up_lr = cv2.flip(img_fish, -1)
3
4 #Hiển thị ảnh đã lật:
5 plt.imshow(img_flip_up_lr)
6 plt.show()
```





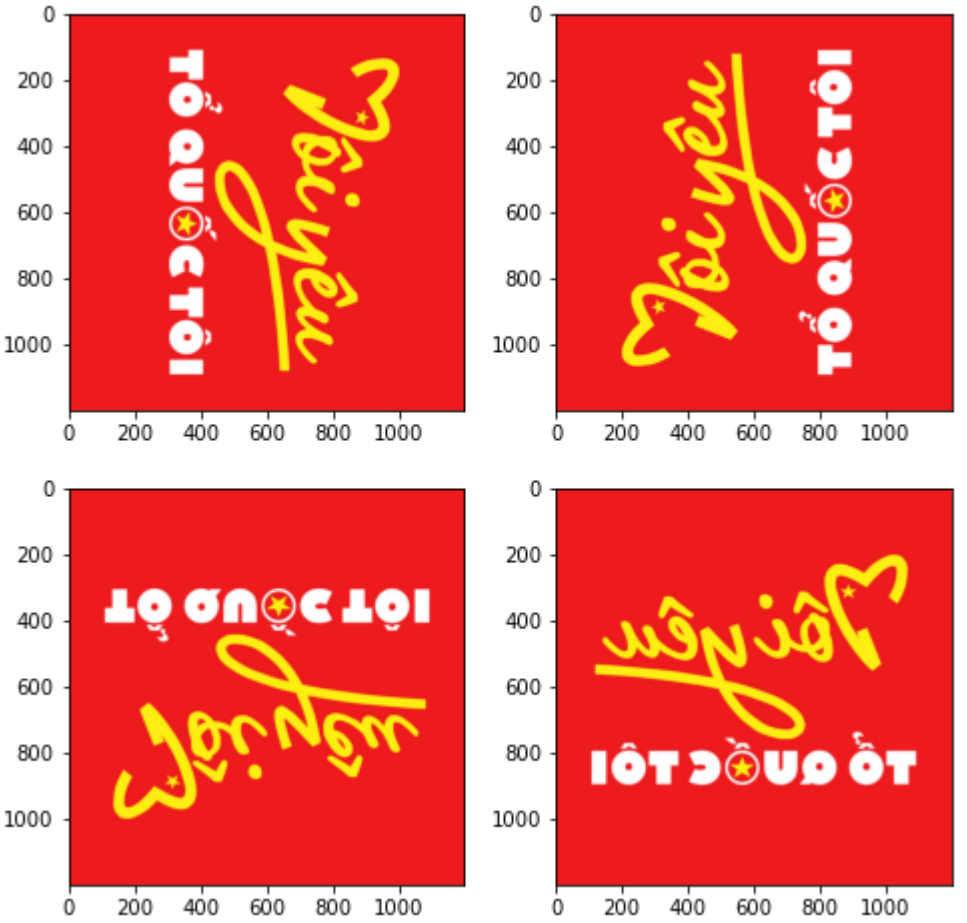


Thực hành số 2.3

Thực hành 2.3

Yêu cầu :

1. Sinh viên đọc và hiển thị ảnh images/Thuchanh_2_3.jpg ở chế độ mặc định.
2. Thực hiện xoay ảnh 90 độ thuận và ngược chiều kim đồng hồ.
3. Thực hiện lật ảnh theo chiều ngang và dọc
4. Hiển thị và lưu các ảnh đã xoay, lật vào thư mục images/Saves/




QUESTIONS
Q&A
ANSWERS