**hoàn toàn có thể áp dụng bộ lọc Gaussian cho ảnh màu** mà **không cần chuyển sang ảnh xám.**

Tuy nhiên, cần lưu ý một số điểm quan trọng.

**Cách hoạt động:**

* Ảnh màu thông thường có 3 kênh: **Red, Green, Blue** (RGB hoặc BGR trong OpenCV).
* Bộ lọc Gaussian hoạt động trên **ma trận 2 chiều**, vì thế:
  + Với ảnh màu (3 chiều), bạn cần xử lý **mỗi kênh riêng biệt** hoặc dùng hàm hỗ trợ nhiều kênh.

**2 cách thực hiện:**

**Cách 1: Dùng cv2.GaussianBlur trực tiếp**

img = cv2.imread("your\_image.jpg")

blurred = cv2.GaussianBlur(img, (5, 5), sigmaX=2)

* OpenCV sẽ **tự động áp dụng bộ lọc Gaussian lên từng kênh màu riêng biệt**, **bảo toàn màu sắc**

**Cách 2: Tự xử lý từng kênh (nếu dùng kernel tùy chỉnh)**

# Tạo kernel trước: gaussian\_kernel(size, sigma)

b, g, r = cv2.split(img)

b\_blur = cv2.filter2D(b, -1, kernel)

g\_blur = cv2.filter2D(g, -1, kernel)

r\_blur = cv2.filter2D(r, -1, kernel)

blurred\_img = cv2.merge([b\_blur, g\_blur, r\_blur])

* Cách này dùng khi bạn muốn kiểm soát kernel hoặc áp dụng bộ lọc khác nhau cho từng kênh.

A screenshot of a phone

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

Trường hợp có padding > 0:

Bổ sung code sau:

if padding > 0:

input = np.pad(input, ((padding, padding), (padding, padding)), mode='constant')

mode='constant': thêm padding với **giá trị hằng số** (mặc định là 0) (Thêm viền giá trị **0 xung quanh ma trận 2D)**

tuple này quy định số lượng padding cho mỗi chiều:

* (padding, padding) ở chiều 0: hàng (trên, dưới)
* (padding, padding) ở chiều 1: cột (trái, phải)

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a phone

AI-generated content may be incorrect.A screenshot of a phone

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

Để hiểu tại sao trục Y lại đi từ **trên xuống dưới** thay vì **dưới lên** như trong hệ tọa độ Descartes, chúng ta cần hiểu một số đặc điểm của cách mà **máy tính** và **hình ảnh số** được lưu trữ và xử lý.

**📌 Hệ tọa độ Descartes (toán học truyền thống):**

Trong hệ tọa độ Descartes thông thường:

* **Trục Y** tăng từ **dưới lên trên**.
* **Trục X** tăng từ **trái sang phải**.

Điều này có ý nghĩa khi chúng ta vẽ đồ thị trong mặt phẳng 2D, nơi gốc tọa độ (0,0) ở **góc dưới trái**.

**📌 Hệ tọa độ trong ảnh số (computer vision):**

Khi làm việc với ảnh số và dữ liệu ảnh trong **máy tính**, các hệ thống như **NumPy**, **OpenCV**, và các thư viện xử lý ảnh khác đều lưu trữ ảnh dưới dạng ma trận 2D, trong đó mỗi phần tử trong ma trận (pixel) đại diện cho một điểm ảnh trong không gian ảnh.

**Lý do trục Y tăng từ trên xuống trong ảnh số:**

1. **Cách lưu trữ ảnh trong bộ nhớ**:
   * **Ảnh được lưu trong bộ nhớ theo dạng ma trận 2D**, với các dòng (rows) và cột (columns).
   * Trong khi vẽ ảnh lên màn hình, **dòng đầu tiên của ảnh** sẽ được hiển thị ở **góc trên cùng**, vì màn hình máy tính thường hiển thị từ **trên xuống dưới**.
   * Điều này đồng nghĩa với việc **y = 0** (hàng đầu tiên) sẽ là dòng ở trên cùng, và các giá trị của y sẽ tăng dần khi di chuyển xuống dưới.
2. **Màn hình máy tính**:
   * Màn hình máy tính hiển thị hình ảnh theo cách mà **dòng đầu tiên** sẽ xuất hiện ở phần trên của màn hình. Điều này là do **màn hình được quét theo chiều dọc**.
   * Do đó, **dòng đầu tiên (y = 0)** luôn được hiển thị ở **góc trên cùng** của màn hình, và các dòng sau đó (y > 0) được hiển thị dần dần xuống dưới.
3. **Tính nhất quán với các phương thức vẽ đồ họa khác**:
   * Các công cụ vẽ đồ họa như **matplotlib**, **OpenCV**, hay **PIL** sử dụng cách tiếp cận này để duy trì tính nhất quán trong cách hiển thị ảnh, và thuận tiện cho việc **lập trình ảnh số**.

**4. Hệ thống quét pixel:**

* Khi hiển thị một ảnh, hệ thống quét qua ảnh **từ trên xuống dưới**. Điều này phù hợp với cách hiển thị các điểm ảnh trên màn hình, vì vậy pixel ở y=0 nằm ở trên cùng, và pixel ở y=height-1 nằm ở dưới cùng.

**📚 Tóm lại:**

* **Trục Y** đi từ **trên xuống dưới** trong ảnh số vì cách ảnh được lưu trữ trong bộ nhớ và hiển thị trên màn hình máy tính, đồng thời phù hợp với cách quét hình ảnh từ trên xuống dưới.
* Điều này là khác với hệ tọa độ Descartes truyền thống, nơi trục Y tăng từ **dưới lên**.