Bounding box :

در کتابخانه‌ی **OpenCV**، اصطلاح **Bounding Box**(چارچوب) به یک مستطیل اطلاق می‌شود که یک شیء (مثل یک چهره، خودرو، یا هر ناحیه‌ای از عکس) را در یک تصویر یا فریم ویدیویی محصور می‌کند. این مستطیل معمولاً با استفاده از مختصات گوشه بالا-چپ و عرض و ارتفاع تعریف می‌شود. هدف از ایجاد این چارچوب محصور کردن اشیا درون عکس با ساده ترین هندسه ممکن است.

در OpenCV ، این مستطیل معمولاً به صورت یک تاپل با چهار مقدار تعریف می‌شود:

```
(x, y, width, height)
```

`x` مختصات افقی گوشه بالا-چپ مستطیل

`y` مختصات عمودی گوشه بالا-چپ مستطیل

`width` عرض مستطیل

`height` ارتفاع مستطیل

کاربردهای عملی Bounding Box

1. **تشخیص شیء (Object Detection)**
   - خروجی مدل‌های مثل YOLO ، SSD، Faster R-CNN، اغلب به شکل Bounding Box + کلاس + احتمال است.
2. **ردیابی شیء (Object Tracking)**
   - در الگوریتم‌هایی مثل `cv2.TrackerKCF_create()` ابتدا یک Bounding Box اولیه انتخاب می‌شود و سپس در فریم‌های بعدی دنبال می‌شود.
3. **آماده‌سازی داده برای یادگیری ماشین / شبکه‌های عصبی**
   - در فریم‌ورک‌هایی مانند TensorFlow یا PyTorch از Bounding Box برای آموزش مدل استفاده می‌شود.

برای ایجاد **Bounding Box** بر روی اشیا مختلف می توانیم مراحل زیر را دنبال کنیم.

1- در مرحله اول عکس را می خوانیم اگر عکسمان رنگی بود باید آن را به یک عکس خاکستری تبدیل کنیم.

```
img = cv2.imread("images/balloon.png")

# convert image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

2- مرحله بعدی تبدیل عکس به یک عکس باینری با یکی از روش های آستانه گذاری است.

```
# invert black and white
_, binary = cv2.threshold(gray,127,255,cv2.THRESH_BINARY_INV)
```

3- در صورتی که عکس باینری ما دارای نویز یا از دست رفتن بخش هایی از اشیا بود با یکی از انواع عملیات مورفولوژی آن را ترمیم می کنیم.

```
# Let's define our kernel size
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5), (2, 2))
closing = cv2.morphologyEx(binary, cv2.MORPH_CLOSE, kernel)
```

4- مرحله بعد تشخیص کانتورها برای چیدا کردن اشیا مختلف در عکس است

```
contours, _ = cv2.findContours(closing, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

5- در مرحله بعد بجای رسم خود کانتورها می توانیم با استفاده از دستور زیر دور هر کدام از کانتورها یک مستطیل رسم کرده و در نهایت آنها را چاپ کنیم:

```
for c in contours:
    x, y, w, h = cv2.boundingRect(c)
    cv2.rectangle(drawing, (x, y), (x + w, y + h), (0, 255, 255), 2)

plt.imshow(drawing[...,::-1])
```

برای انجام این مرحله ابتدا یک حلقه for بر روی کانتورهای پیدا شده ایجاد می کنیم، بنابراین متغیر c در هر دور از حلقه در بردارنده یک کانتور است. تابع cv2.boundingRect(c) برای هریک از کانتورهای موجود در c یک مستطیل ایجاد می کند. و خروجی آن در یک تاپل با 4 مقدار (x,y,w,h) ذخیره می شود.

- `x` مختصات افقی گوشه بالا-چپ مستطیل
- `y` مختصات عمودی گوشه بالا-چپ
- `w` عرض مستطیل از `x` تا `(x+w)`
- `h` ارتفاع مستطیل از `y` تا `(y+h)`

حال که با استفاده از متد فوق توانستیم مستطیل ها را پیدا کنیم کافی است که هر کدام از آنها را رسم کنیم. برای رسم مستطیل ها از دستور cv2.trctangle استفاده می کنیم.

به عنوان مثال اگر فرض کنیم شی ما یک دایره یا مستطیل است که در این مختصات قرار گرفته باشد:

```
x = 100, y = 50, w = 40, h = 30
```

یعنی مستطیل از نقطه `(100, 50)` شروع می‌شود و تا `(140, 80)` ادامه خواهد داشت.