Convolutions with OpenCV and Python

[convolution is simply an element-wise multiplication of two matrices followed by a sum.]

Essentially, this tiny kernel sits on top of the big image and slides from left-to-right and top-to-bottom, applying a mathematical operation (i.e., a convolution) at each (x, y)-coordinate of the original image.

Blurring kernels:

- Average smoothing
- Gaussian smoothing
- Median smoothing

Edge Detection kernels:

- Laplacian
- Sobel
- Scharr
- Prewitt

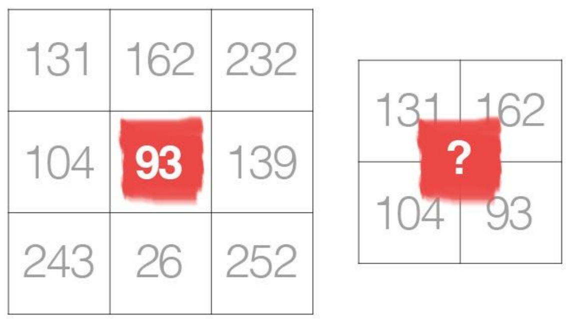
Defining a kernel:

$$K = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Kernel can be of any size (MxN), where both M and N are *odd integers.*

[Most kernels you'll typically see are actually square N x N matrices.]

We use an *odd* kernel size to ensure there is a valid integer *(x, y)*-coordinate at the center of the image:



In image processing, a convolution requires three components:

1. An input image.
2. A kernel matrix that we are going to apply to the input image.
3. An output image to store the output of the input image convolved with the kernel.

Convolution itself is actually very easy. All we need to do is:

1. Select an *(x, y)*-coordinate from the original image.
2. Place the **center** of the kernel at this *(x, y)*-coordinate.
3. Take the element-wise multiplication of the input image region and the kernel, then sum up the values of these multiplication operations into a single value. The sum of these multiplications is called the **kernel output**.
4. Use the same *(x, y)*-coordinates from **Step #1**, but this time, store the kernel output in the same *(x, y)*-location as the output image.

Example:
example of convolving (denoted mathematically as the "*" operator) a 3 x 3 region of an image with a 3 x 3 kernel used for blurring:

$$O_{i,j} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \star \begin{bmatrix} 197 & 50 & 213 \\ 3 & 181 & 203 \\ 231 & 2 & 93 \end{bmatrix} = \begin{bmatrix} 1/9 \times 197 & 1/9 \times 50 & 1/9 \times 213 \\ 1/9 \times 3 & 1/9 \times 181 & 1/9 \times 203 \\ 1/9 \times 231 & 1/9 \times 2 & 1/9 \times 93 \end{bmatrix}$$

$$O_{i,j} = \sum \begin{bmatrix} 21 & 5 & 23 \\ 0 & 20 & 22 \\ 25 & 0 & 10 \end{bmatrix} = 126$$

After applying this convolution, we would set the pixel located at the coordinate (i, j) of the output image O to O_i,j = 126.