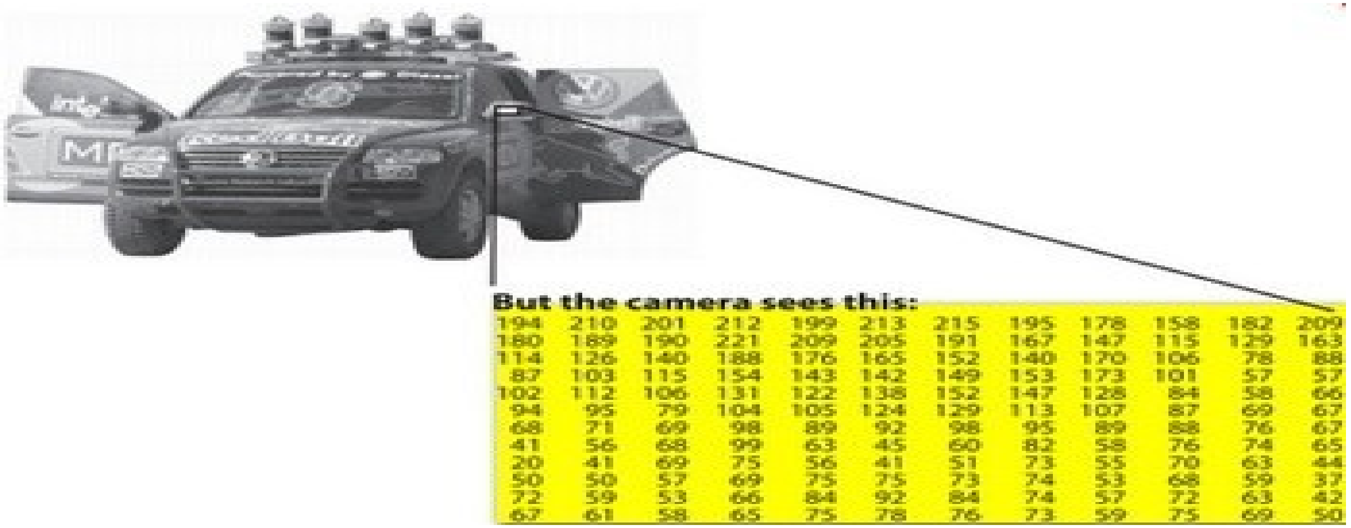
Introduction to Image Processing in OpenCV

The *Mat* thingy

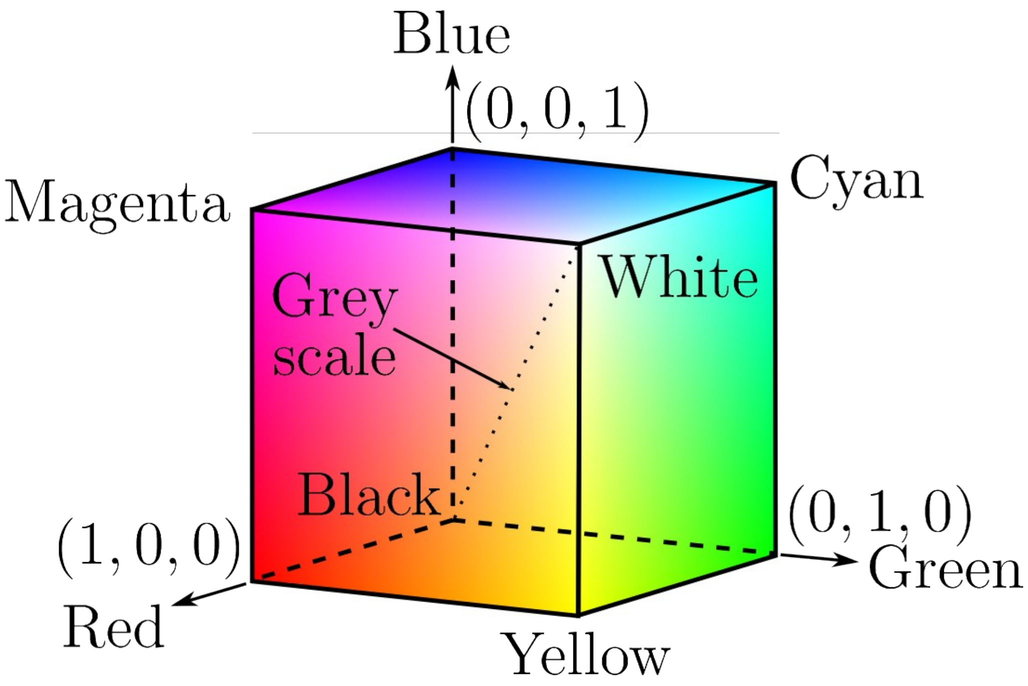
- It is a class to store image in C++, which replaced the *Iplimage* structure in C where we needed to manually allocate memory.
- Basically you can assume Mat to be a 2D array of integers. Where each matrix element is called a pixel of the ima
- Each pix is from [0,255].

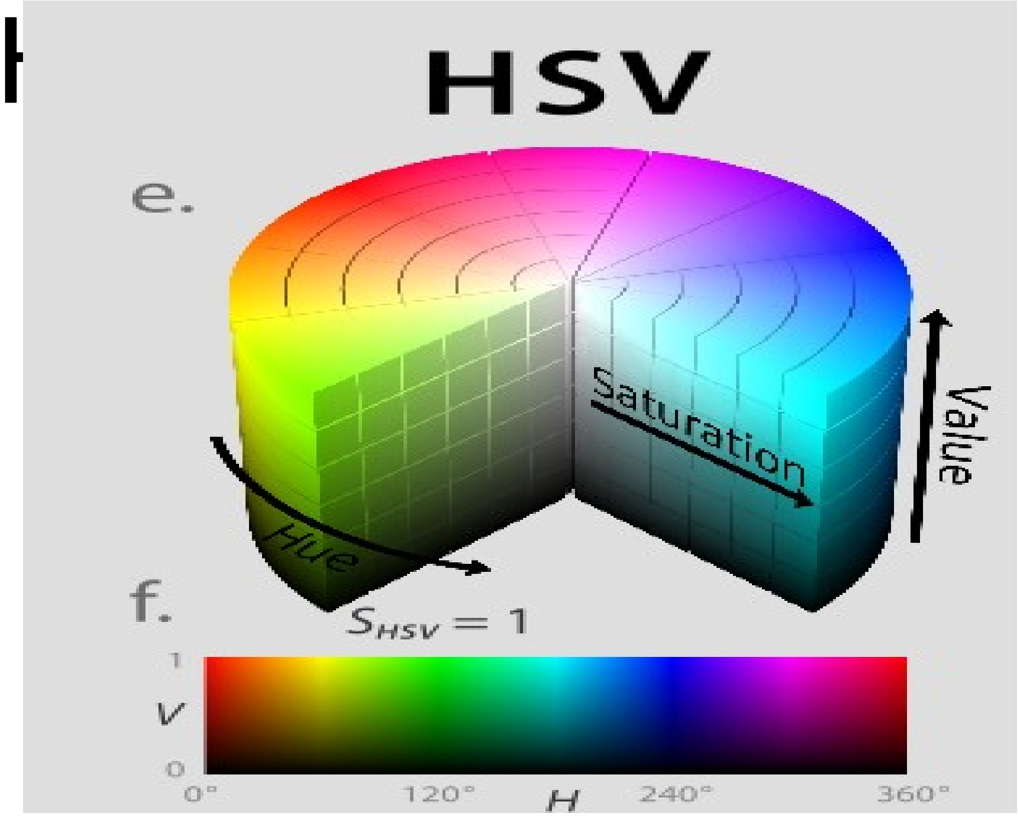


But the camera sees this:

194	210	201	212	199	213	215	195	178	158	182	209
180	189	190	221	209	205	191	167	147	115	129	163
114	126	140	188	176	165	152	140	170	106	78	88
87	103	115	154	143	142	149	153	173	101	57	57
102	112	106	131	122	138	152	147	128	84	58	66
94	95	79	104	105	124	129	113	107	87	69	67
68	71	69	98	89	92	98	95	89	88	76	67
41	56	68	99	63	45	60	82	58	76	74	65
20	41	69	75	56	41	51	73	55	70	63	44
50	50	57	69	75	75	73	74	53	68	59	37
72	59	53	66	84	92	84	74	57	72	63	42
67	61	58	65	75	78	76	73	59	75	69	50

Grayscale, RGB and H



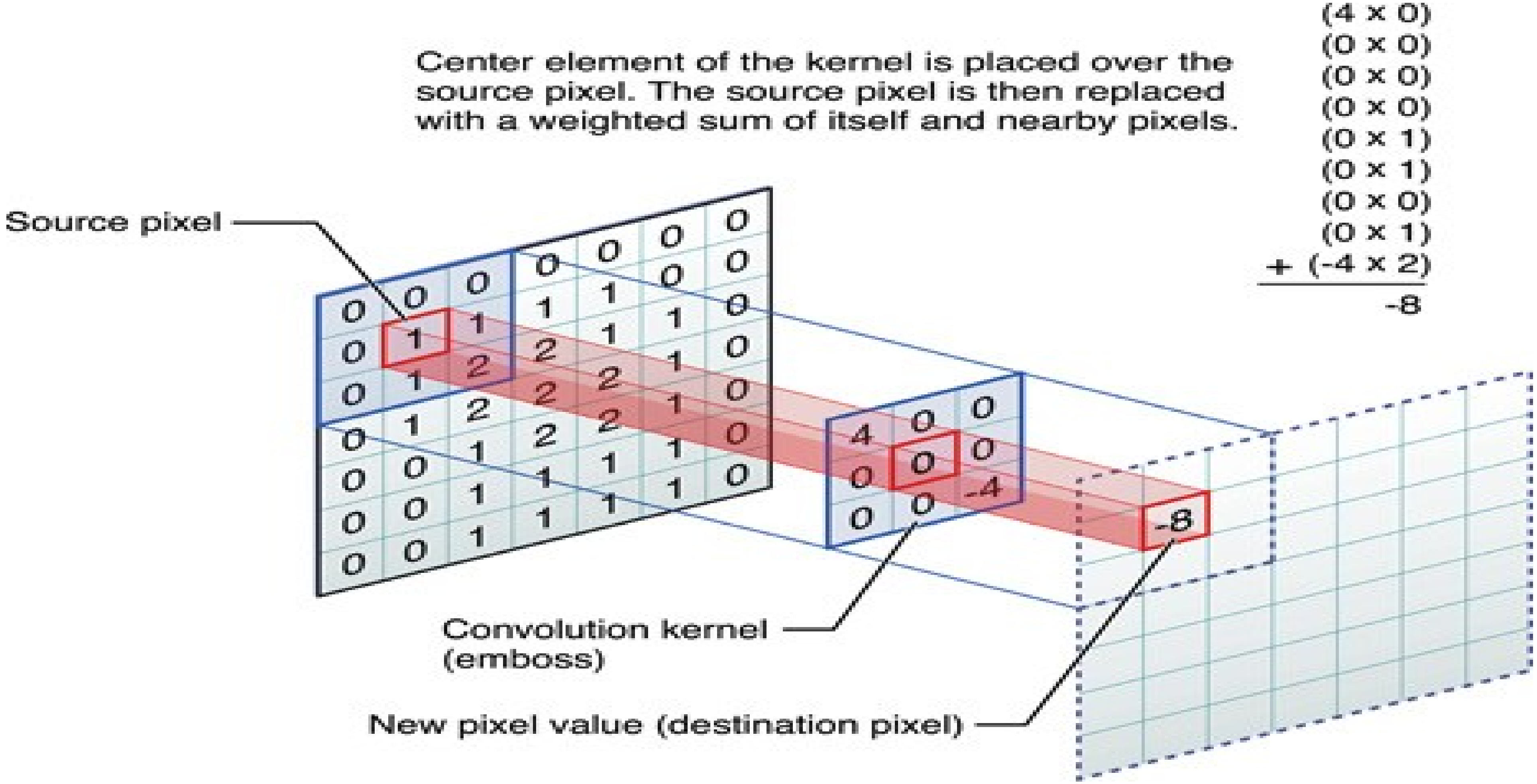


Most often in tracking applications we prefer HSV as the color component and the intensity component in HSV are independent of each other. Baffled?

Contrast and Brightness

- Two commonly used point processes are *multiplication* and *addition* with a constant
- $G(x,y) = AF(x,y) + B$
- The parameters $A>0$ and B control the *contrast* and *brightness* parameters.

Kernel Operations: Convolution



Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8

Source pixel

0	0	0	0	0	0	0
0	1	1	1	1	0	0
0	1	2	2	1	1	0
0	1	2	2	2	1	0
0	0	1	2	2	1	0
0	0	1	1	1	1	0
0	0	1	1	1	1	0

Convolution kernel (emboss)

4	0	0
0	0	0
0	0	-4

New pixel value (destination pixel)

Basic Operations: Smoothing

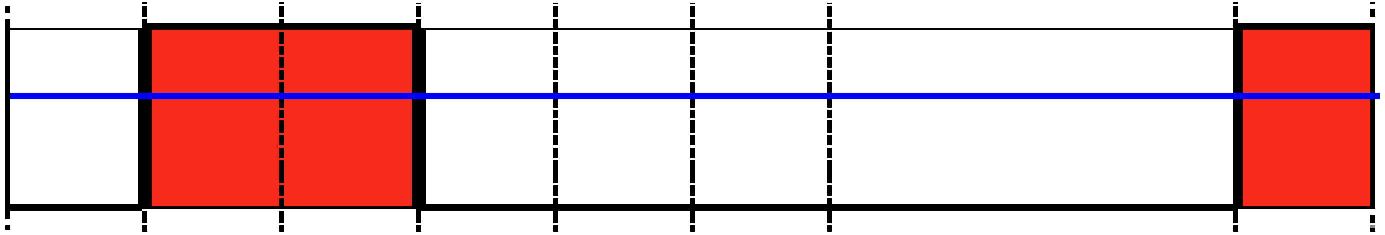
- Used frequently in IP operations.
- Typically used blurs are, Blur, Gaussian Blur, Median Blur.
- A median blur example (code)

$$K = \frac{1}{K_{width} \cdot K_{height}} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \\ . & . & . & \dots & 1 \\ . & . & . & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \end{bmatrix}$$

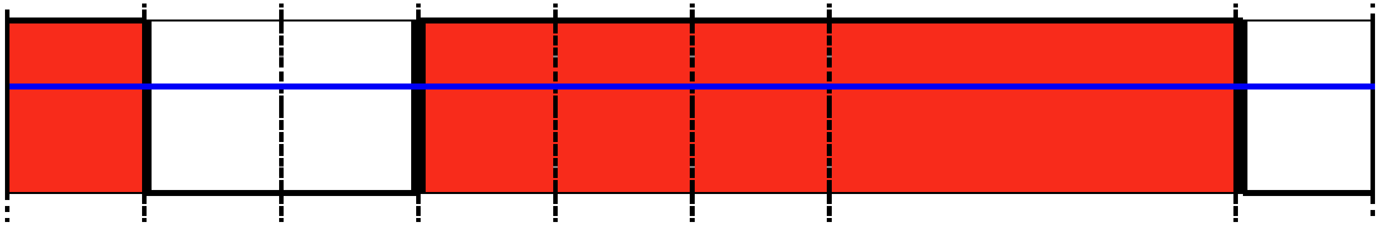
Basic Operations: Thresholding

- Many kind of thresholding, binary , inverted binary are the most common ones. For more reference you can look at OpenCV tutorial on thresholding for more details.

$$\mathtt{dst}(x,y) = \begin{cases} \mathtt{maxVal} & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ 0 & \text{otherwise} \end{cases}$$



$$\mathtt{dst}(x,y) = \begin{cases} 0 & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ \mathtt{maxVal} & \text{otherwise} \end{cases}$$



Basic Operations: Edge Detection

- There are many edge detection algorithms, most popular are Sobel, Laplace and Canny, these three have their functions predefined in OpenCV. The basic concept is the same.
- When we take the derivative of the image we can find spikes at edge et a sharp change in intensity.

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

$$G = |G_x| + |G_y|$$

Basic Operations: Image Moments

- Basic definition is given by the function $M_{ij} = \sum_x \sum_y x^i y^j I(x,y)$
- Area (for binary images) or sum of grey level (for greytone images): M_{00}
- Centroid: { x, y } = {M10/M00, M01/M00 }
- Useful in tracking applications, how to use moment functions will be explained a bit in the following example which I will cover for tracking a red ball.

Using Webcam

- Basically, using a direct video feed is getting images at a certain frame rate, and you need to process each frame for you desired output.

Will show it to you in the sample program.

References: The most important Slide

- Background Subtraction: http://docs.opencv.org/trunk/doc/tutorials/video/background_subtraction/background_subtraction.html
- Hand Gesture recog.(tutorial explains using IplImage usage): http://anikettatipamula.blogspot.ro/2012/02/hand-gesture-using-opencv.html
- Patch of OpenCV(any C++ code) with Arduino. http://salilkapur.wordpress.com/2013/03/08/communicating-with-arduino-using-c/
- Nice Shit http://www.cse.unr.edu/~bebis/CS485/Lectures/Intro_OpenCV.pdf