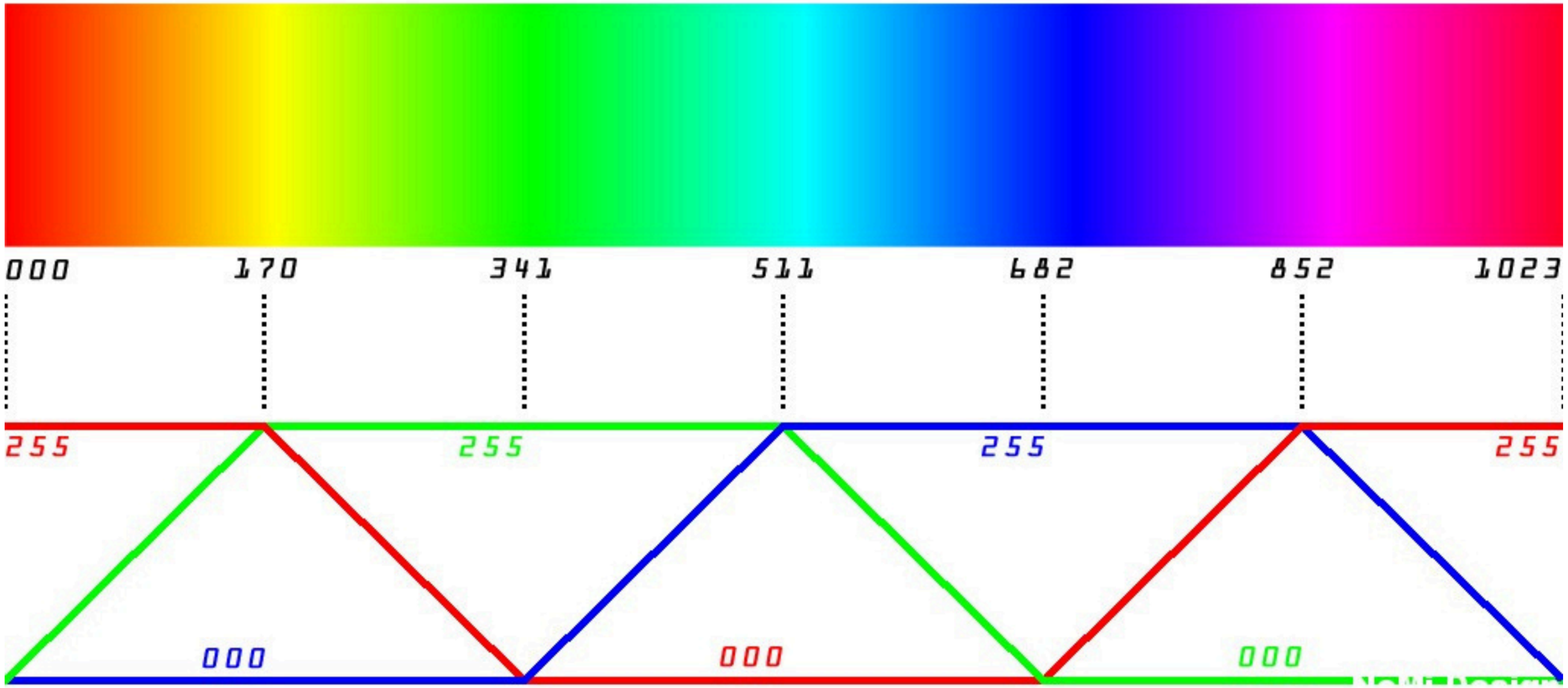
Week 2

DEEP LEARNING FOR COMPUTER VISION

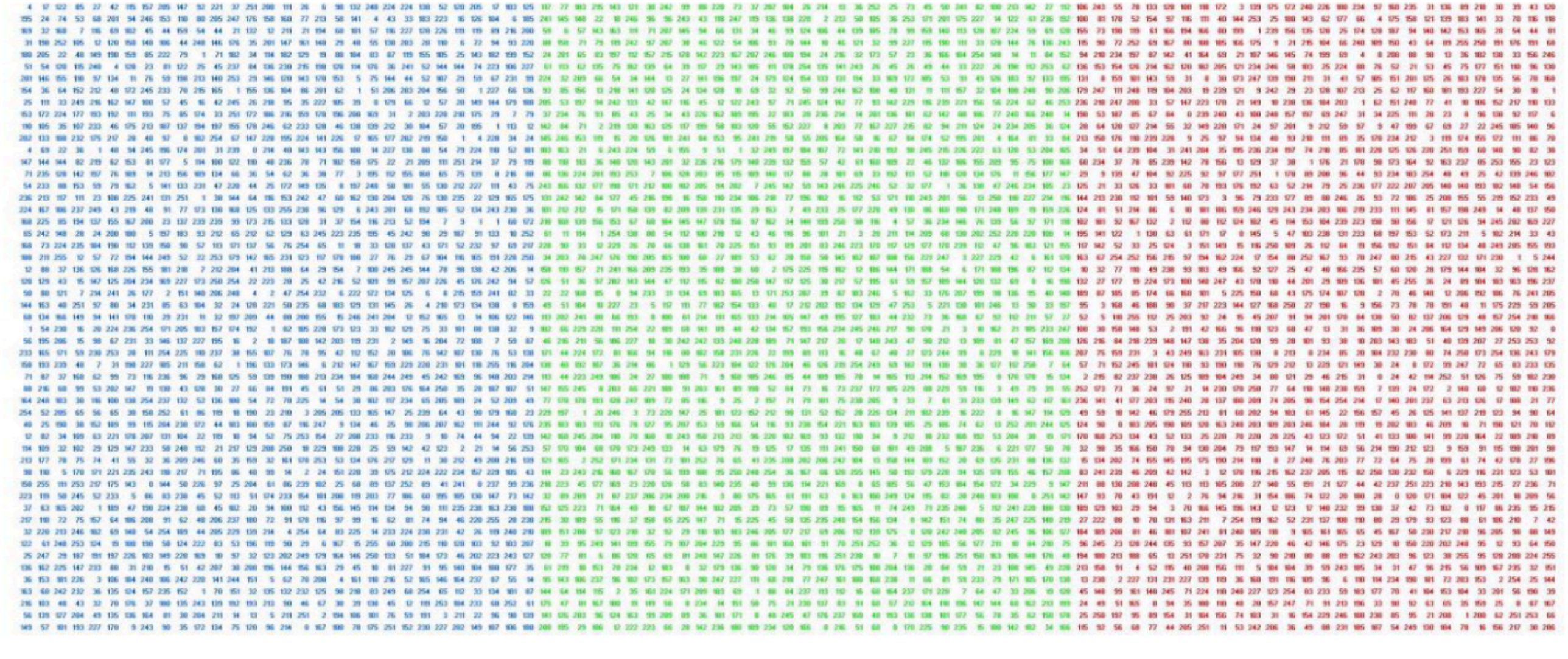
Presented by **Asst. Prof. Dr. Tuchsanai Ploysuwan**

Understanding Images

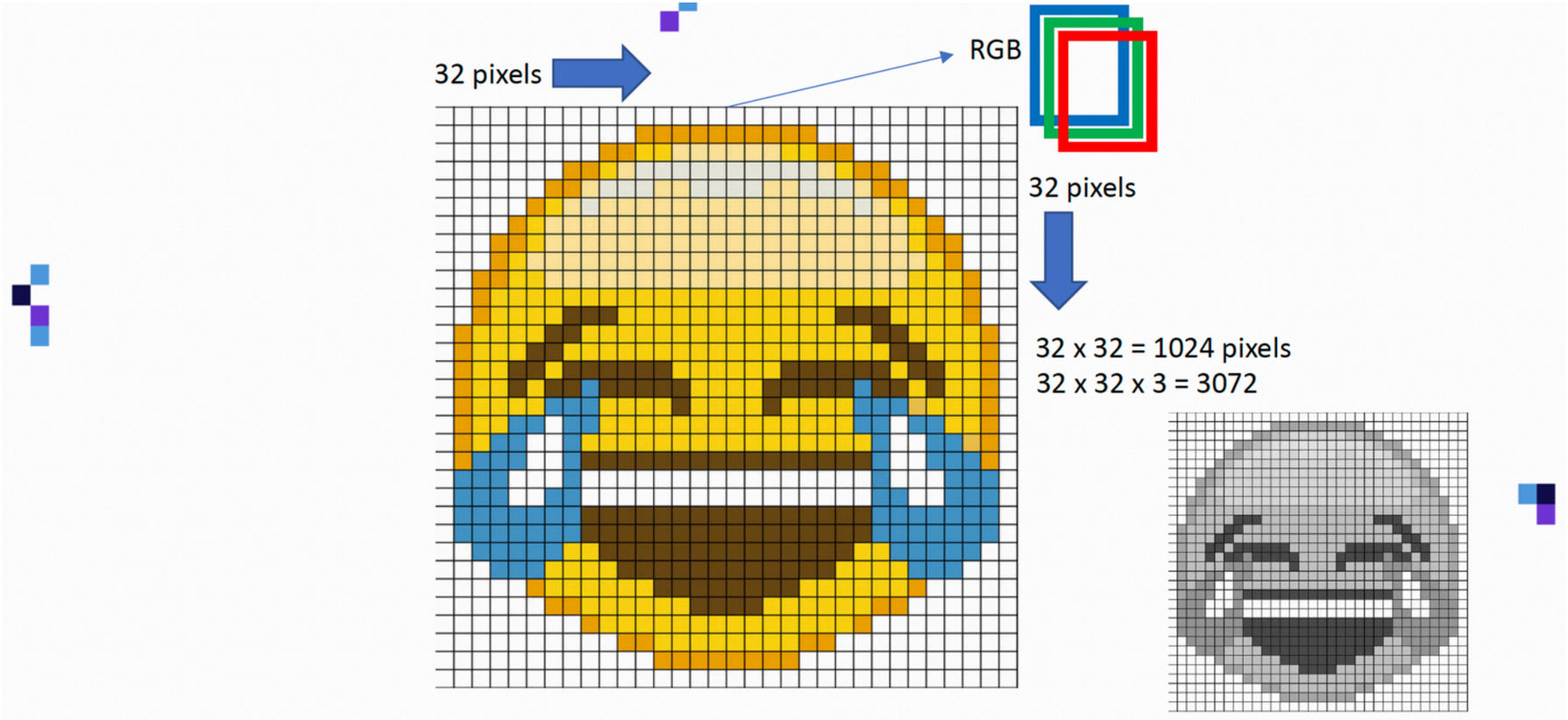
Image Basics



How do Computers ‘see’ Images?



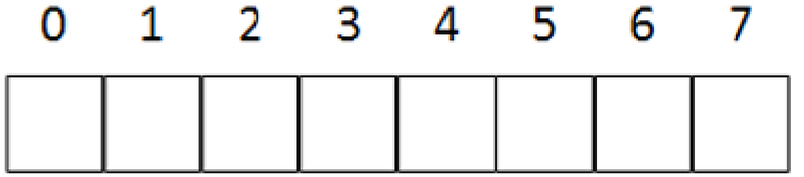
Pixels



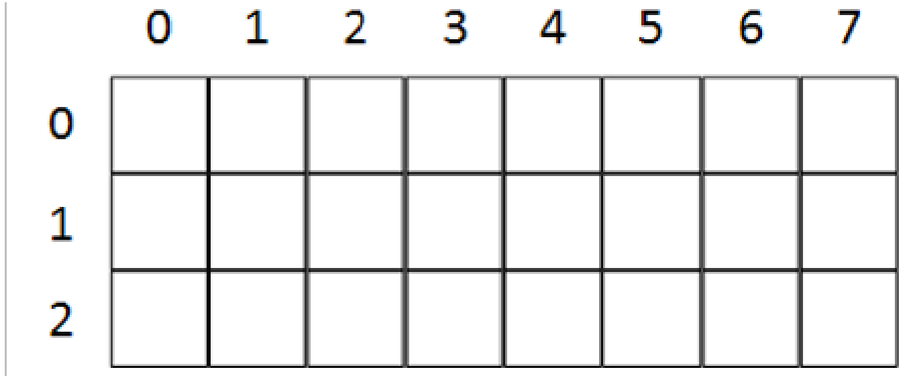
Digital Images Format

Images are stored in Multi-Dimensional Arrays

- A 1-Dimensional array looks like this



- A 2-Dimensional array looks like this



- A **3-Dimensional** array looks like this

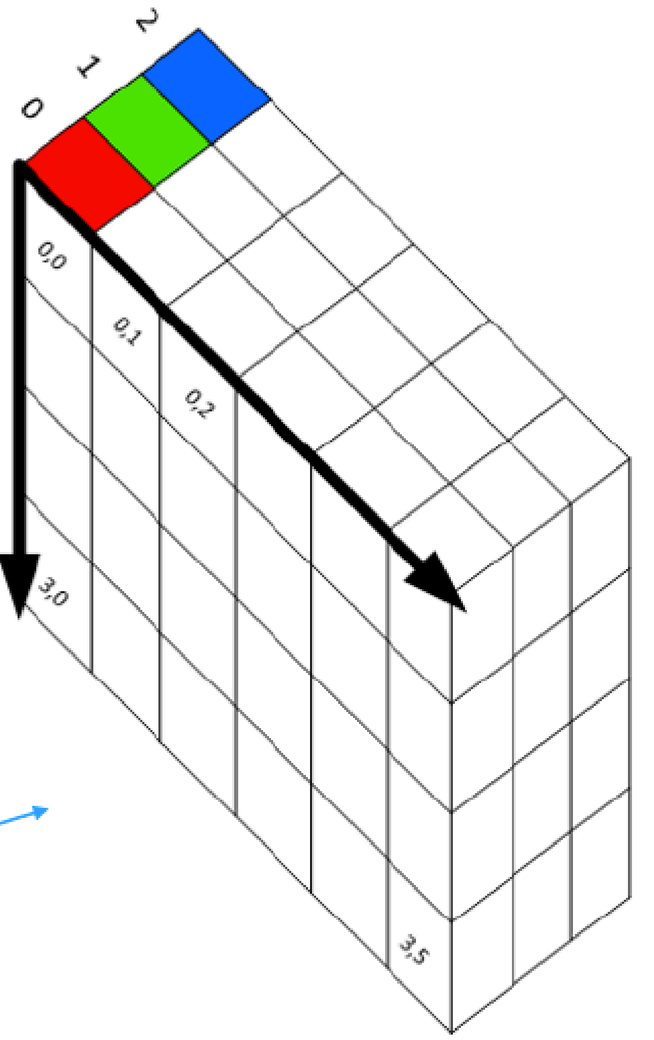


Image File Formats

OpenCV Basics: Reading & Displaying Images

Installation

```
# Install OpenCV using pip
!pip install opencv-python
```

Importing OpenCV

```
import cv2
import numpy as np
print(cv2.__version__)
```

✓ 2.8s

Loading Images

```
# Load an image using 'imread' specifying the path to image
image = cv2.imread('./images/castara.jpeg')
```

✓ 0.0s

Displaying Images

```
from matplotlib import pyplot as plt

# Show the image with matplotlib without axis
plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
plt.axis('off')
plt.show()
```

✓ 0.1s

Saving Images

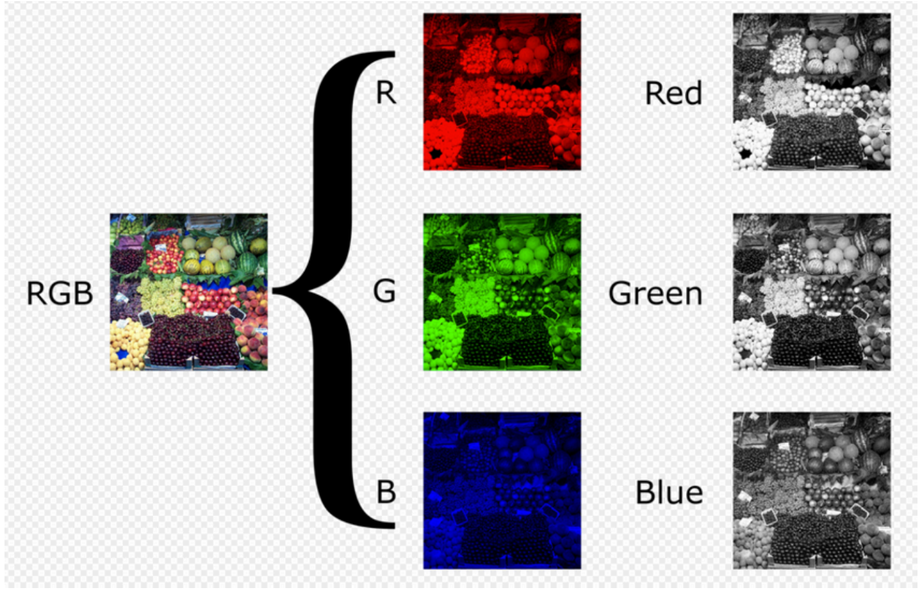
```
cv2.imwrite('output.jpg', image)
```

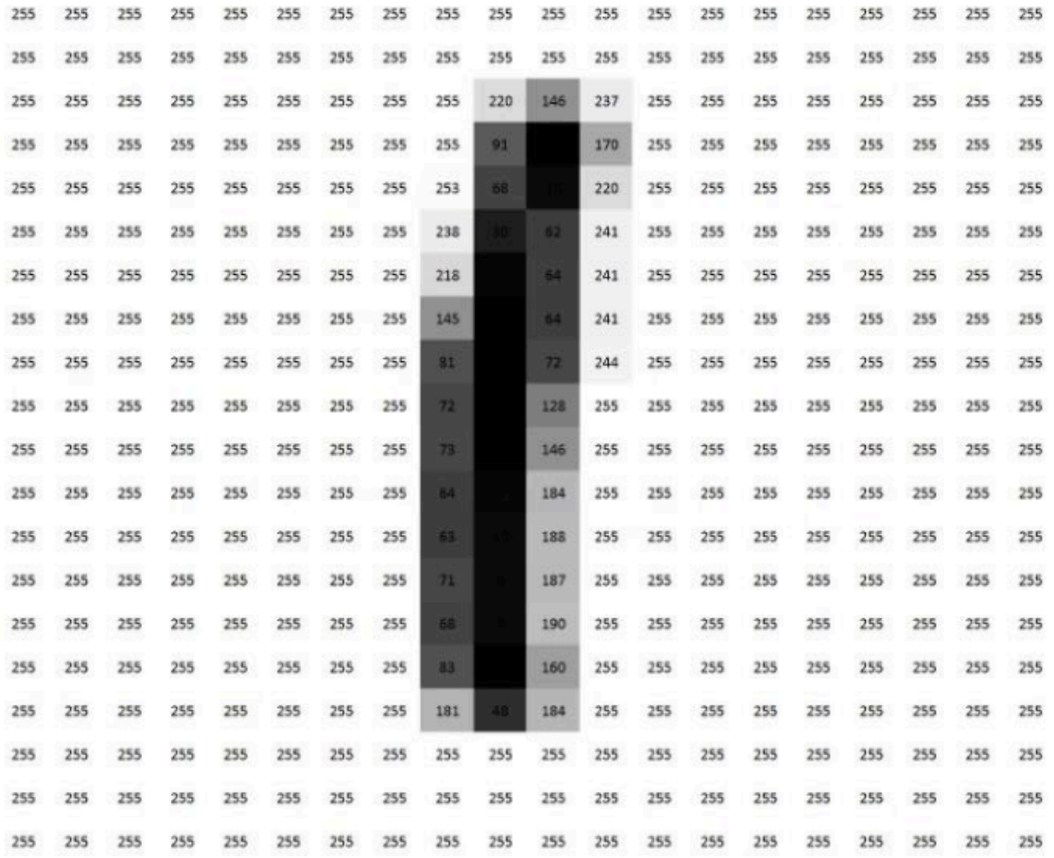
```
True
```

```
cv2.imwrite('output.png', image)
```

Grayscale Image

Sometimes referred to as black and white image





Luma coding in video systems [edit]

Main article: luma (video)

For images in color spaces such as Y'UV and its relatives, which are used in standard color TV and video systems such as PAL, SECAM, and NTSC, a nonlinear luma component (Y') is calculated directly from gamma-compressed primary intensities as a weighted sum, which, although not a perfect representation of the colorimetric luminance, can be calculated more quickly without the gamma expansion and compression used in photometric/colorimetric calculations. In the Y'UV and Y'IQ models used by PAL and NTSC, the rec601 luma (Y') component is computed as

$$Y' = 0.299R' + 0.587G' + 0.114B'$$

where we use the prime to distinguish these nonlinear values from the sRGB nonlinear values (discussed above) which use a somewhat different gamma compression formula, and from the linear RGB components. The ITU-R BT.709 standard used for HDTV developed by the ATSC uses different color coefficients, computing the luma component as

$$Y' = 0.2126R' + 0.7152G' + 0.0722B'.$$

Although these are numerically the same coefficients used in sRGB above, the effect is different because here they are being applied directly to gamma-compressed values rather than to the linearized values. The ITU-R BT.2100 standard for HDR television uses yet different coefficients, computing the luma component as

$$Y' = 0.2627R' + 0.6780G' + 0.0593B'.$$

The Formula

```
Grayscale = 0.299R + 0.587G + 0.114B
```

Where:

- R = Red channel value (0-255)
- G = Green channel value (0-255)
- B = Blue channel value (0-255)

Why These Weights?

• Green (0.587): Human eyes are most sensitive to green

• Red (0.299): Second most impactful color

• Blue (0.114): Least impact on perceived brightness

Example Calculation

```
For a pink pixel:
R = 255
G = 192
B = 203
```

Calculation:

(0.299 × 255) +

(0.587 × 192) +

(0.114 × 203)

= 76.245 + 112.704 + 23.142

= 212 (rounded)

Key Points

• Result is rounded to nearest integer

• Output range is 0 (black) to 255 (white)

• Same value applied to all RGB channels

Grayscaling Images

```
import cv2
from matplotlib import pyplot as plt
```

✓ 0.4s

```
# Load our input image
image = cv2.imread('./images/castara.jpeg')
image.shape
```

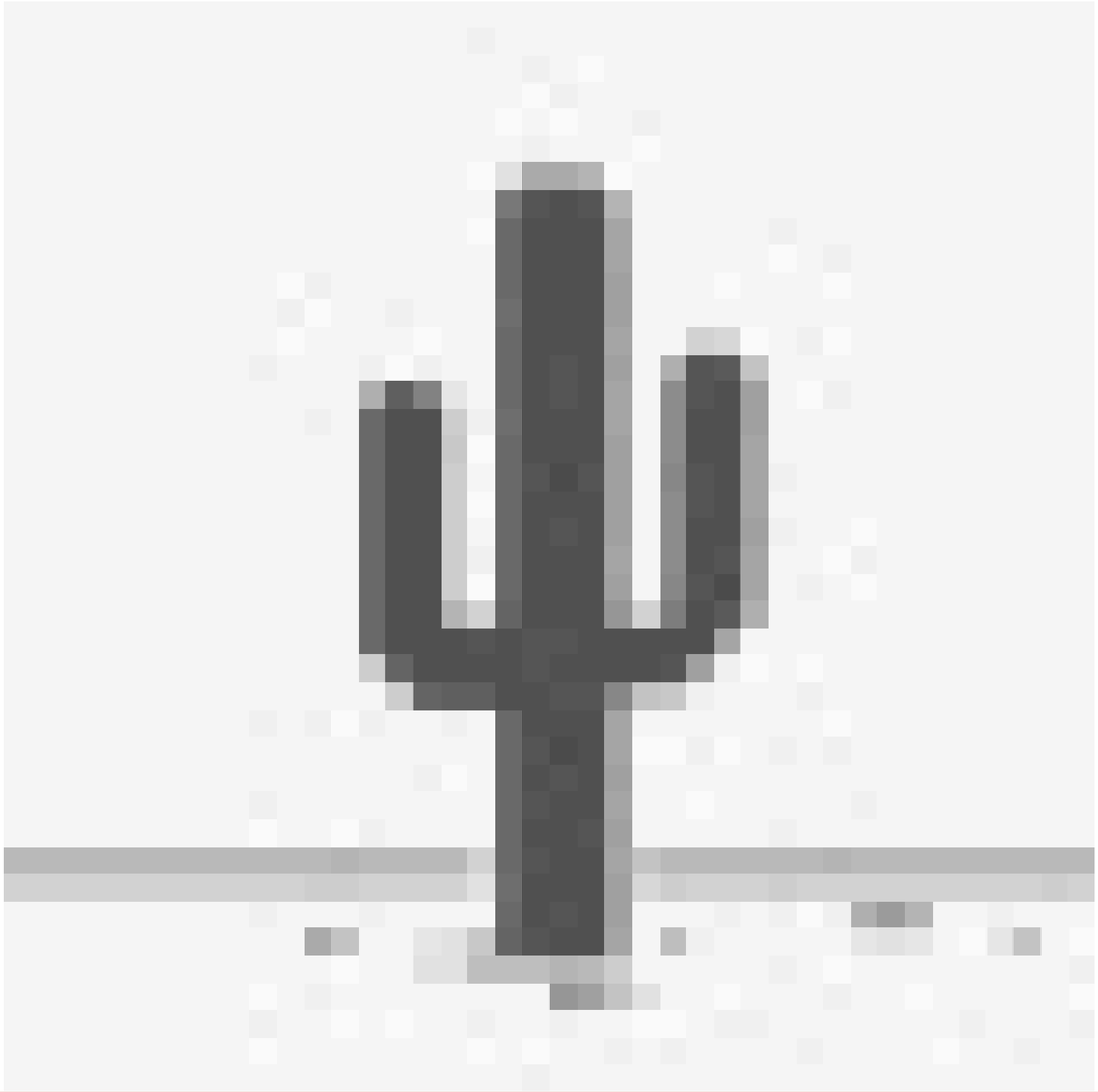
✓ 0.0s

```
(1280, 960, 3)
```

```
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
plt.imshow(cv2.cvtColor(gray_image, cv2.COLOR_GRAY2RGB))
plt.axis('off')
plt.title('Grayscale Image')
plt.show()
```

✓ 0.1s

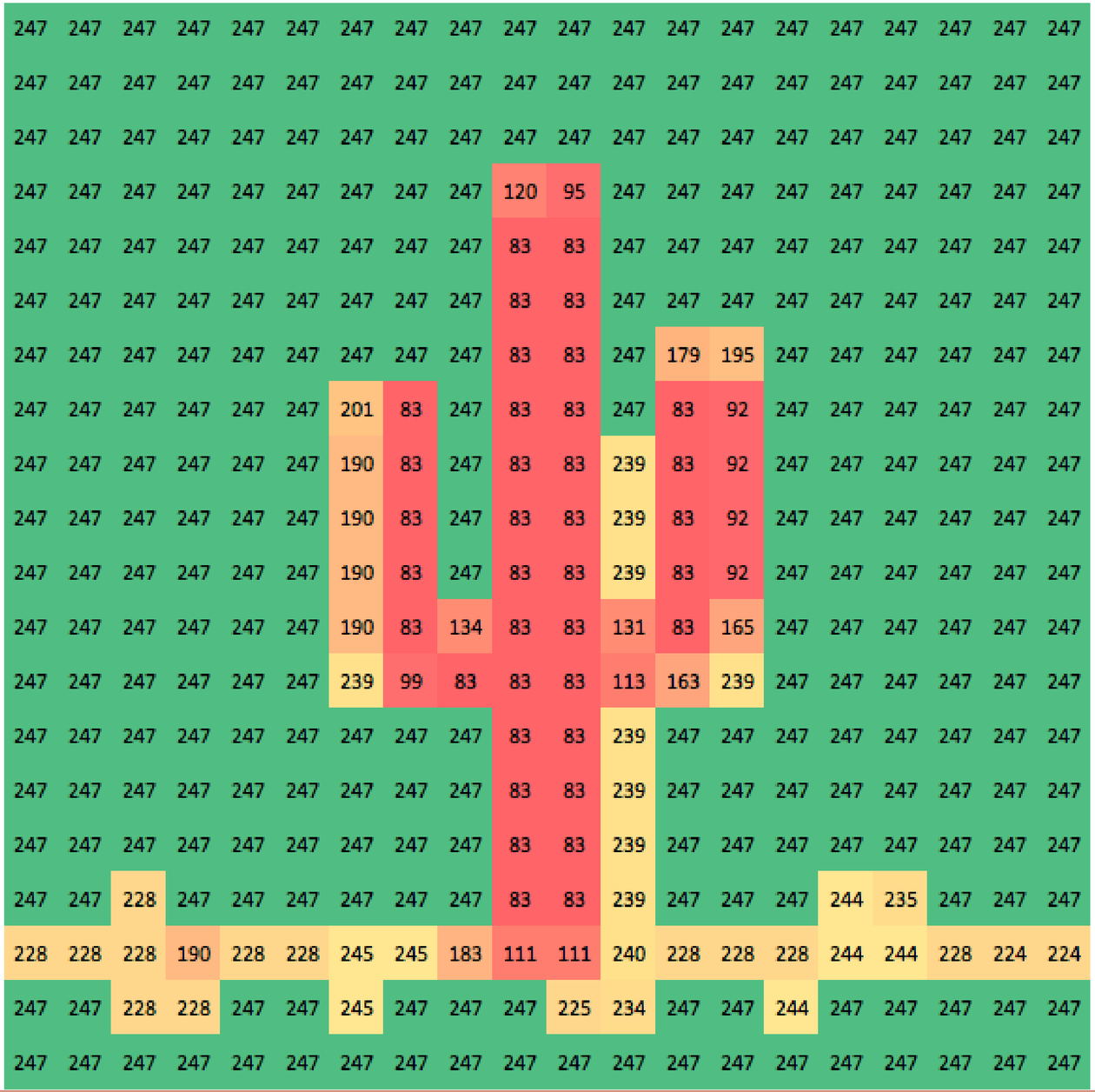
Images & Kernel Convolutions



Images & Kernel Convolutions

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Images & Kernel Convolutions



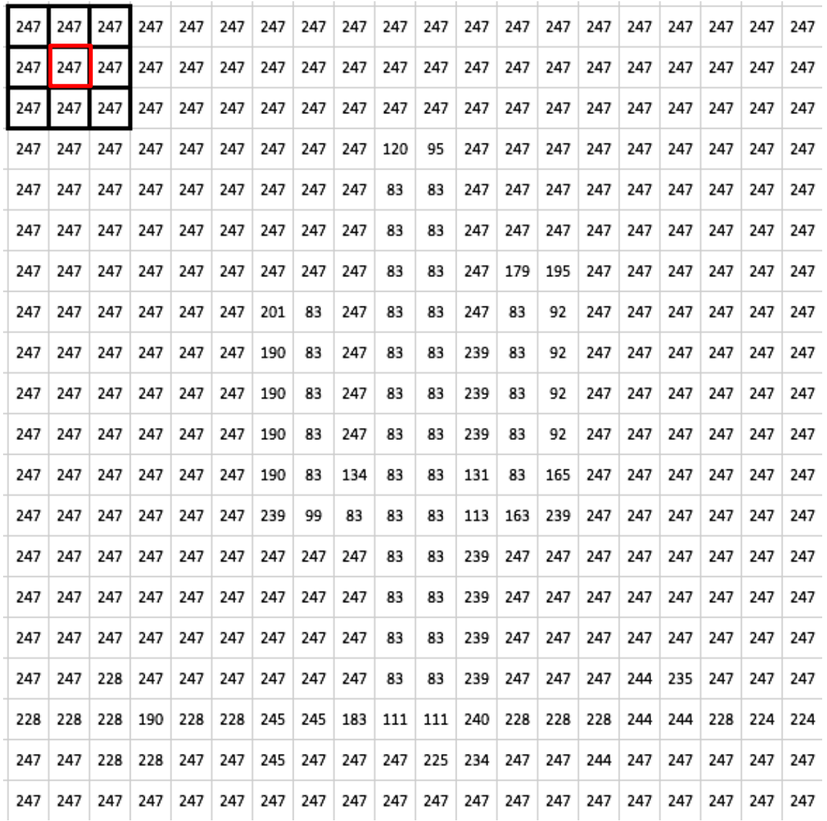
Kernel Convolutions

Kernel is an MxN matrix

3x3 Kernel Example

1	1	1
1	1	1
1	1	1

Kernel Convolutions



Window

247	247	247
247	247	247
247	247	247

x

Kernel

1	1	1
1	1	1
1	1	1

Sum = 9

=

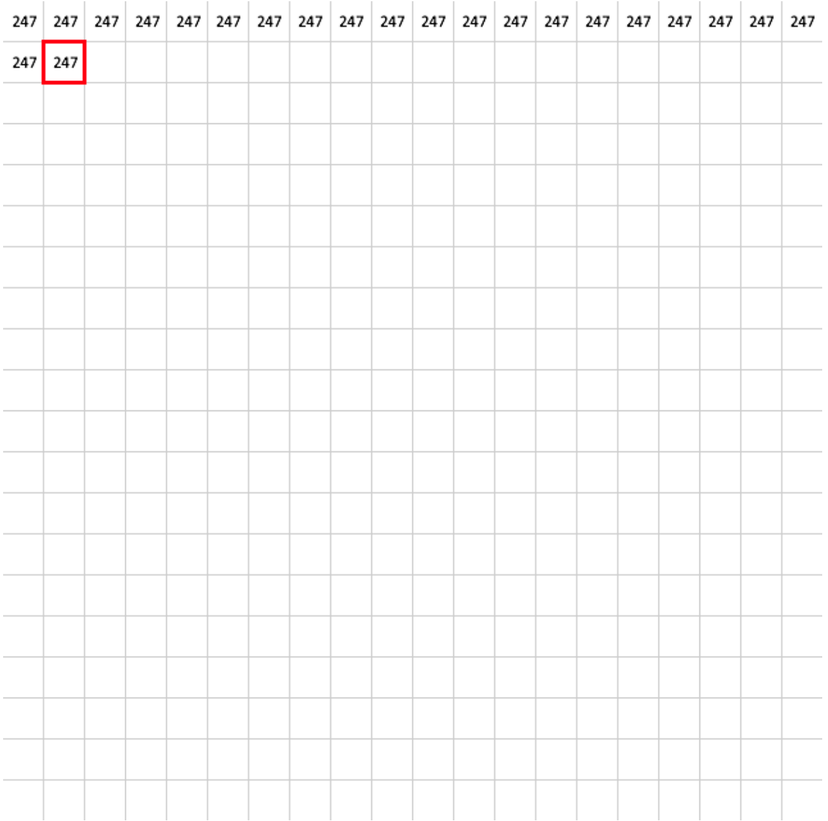
Multiplied

247	247	247
247	247	247
247	247	247

Sum = 2223

New Pixel Value

2223 / 9 = **247**



Kernel Convolutions

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Window

247	247	247
247	247	247
247	247	247

x

Kernel

1	1	1
1	1	1
1	1	1

Sum = 9

=

Multiplied

247	247	247
247	247	247
247	247	247

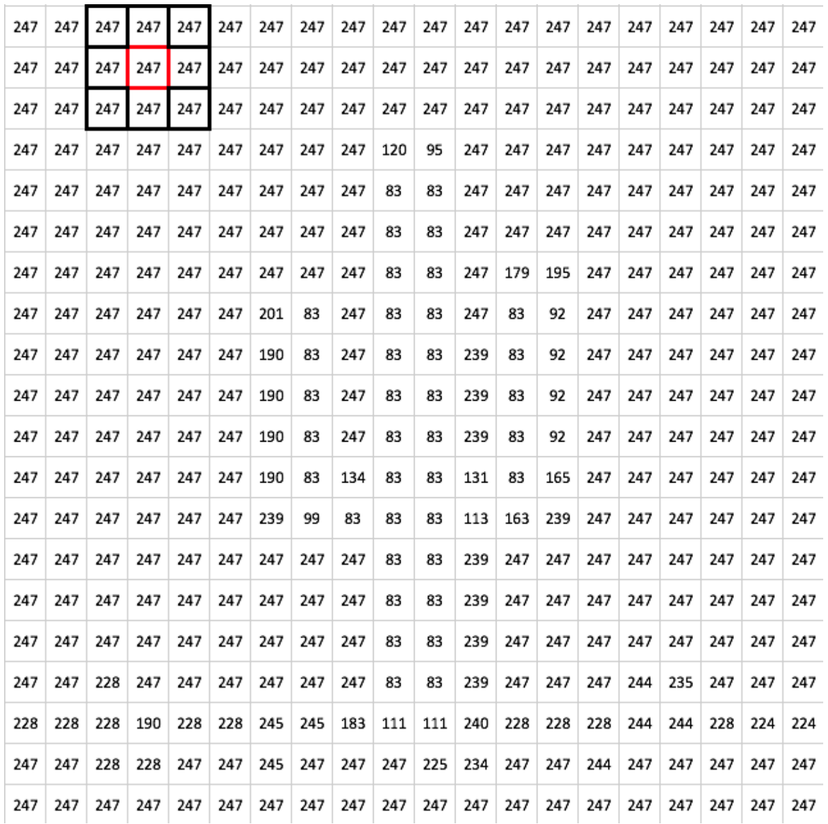
Sum = 2223

New Pixel Value

2223 / 9 = **247**

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247																

Kernel Convolutions



Window

247	247	247
247	247	247
247	247	247

x

Kernel

1	1	1
1	1	1
1	1	1

Sum = 9

=

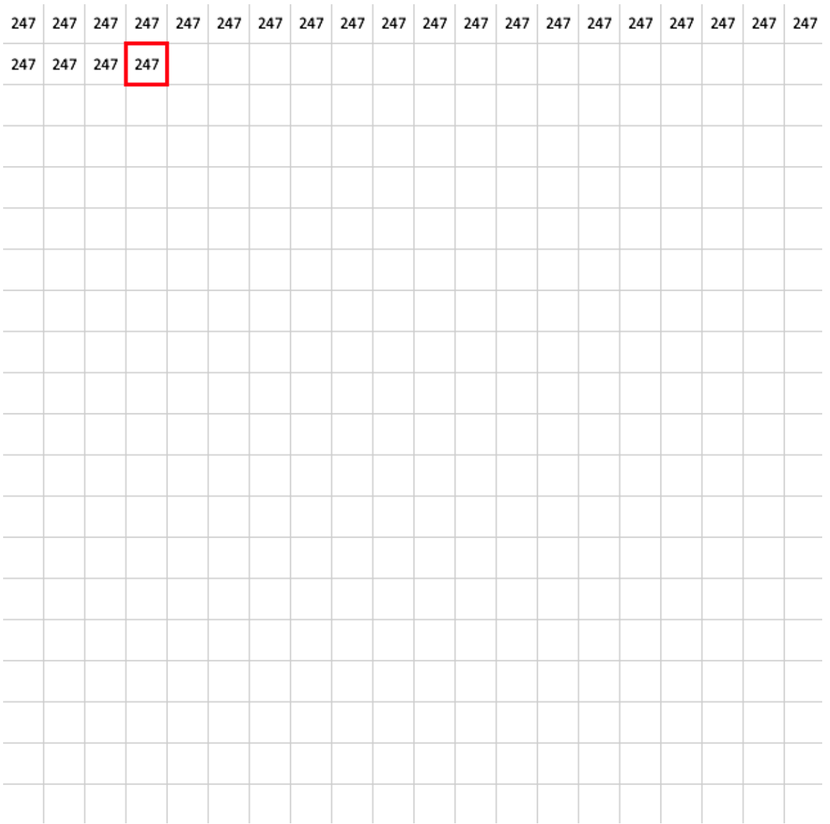
Multiplied

247	247	247
247	247	247
247	247	247

Sum = 2223

New Pixel Value

2223 / 9 = **247**



Kernel Convolutions

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Window

247	247	120
247	247	83
247	247	83

x

Kernel

1	1	1
1	1	1
1	1	1

Sum = 9

=

Multiplied

247	247	120
247	247	83
247	247	83

Sum = 1768

New Pixel Value

1768 / 9 = 196.4

196

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	230	216	216	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	214	180	180	218	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	196										

Kernel Convolutions

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Window

247	120	95
247	83	83
247	83	83

x

Kernel

1	1	1
1	1	1
1	1	1

Sum = 9

=

Multiplied

247	120	95
247	83	83
247	83	83

Sum = 1288

New Pixel Value

1288 / 9 = 143.1

143

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	230	216	216	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	214	180	180	218	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	196	143									

Kernel Convolutions

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Window

247	247	
247	247	

x

Kernel

1	1	
1	1	

Sum = 4

=

Multiplied

247	247	
247	247	

Sum = 988

New Pixel Value

998 / 4 = 247

247

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	233	216	216	230	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	215	180	178	211	246	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	196	143	141	192	245	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	182	231	234	241	247	247	247	247	247
247	247	247	247	247	242	224	224	174	138	135	164	196	198	224	247	247	247	247	247
247	247	247	247	247	236	199	199	156	138	135	146	160	163	207	247	247	247	247	247
247	247	247	247	247	229	175	175	138	138	135	135	138	141	195	247	247	247	247	247
247	247	247	247	247	228	173	173	138	138	135	135	138	141	195	247	247	247	247	247
247	247	247	247	247	228	173	161	125	125	123	123	134	149	203	247	247	247	247	247
247	247	247	247	247	234	181	151	109	107	109	118	146	175	221	247	247	247	247	247
247	247	247	247	247	241	206	175	127	107	109	136	182	210	238	247	247	247	247	247
247	247	247	247	247	247	231	212	158	119	121	166	221	238	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	190	244	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	190	244	247	247	245	245	246	247	247
247	247	247	247	246	246	246	240	188	137	141	191	242	245	247	245	245	243	244	242
247	247	247	247	246	246	246	240	207	171	175	206	242	245	247	245	245	243	244	242
247	247	247	247	246	246	246	240	225	207	212	225	243	245	247	247	247	244	244	242
247	247	247	247	247	247	247	247	247	242	239	239	244	247	247	247	247	247	247	247

Kernel Convolutions

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Kernel

1	1	1
1	1	1
1	1	1

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	233	216	216	230	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	215	180	178	211	246	247	247	247	247	247	247
247	247	247	247	247	247	247	247	196	143	141	192	245	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	182	231	234	241	247	247	247	247
247	247	247	247	247	242	224	224	174	138	135	164	196	198	224	247	247	247	247
247	247	247	247	247	236	199	199	156	138	135	146	160	163	207	247	247	247	247
247	247	247	247	247	229	175	175	138	138	135	135	138	141	195	247	247	247	247
247	247	247	247	247	228	173	173	138	138	135	135	138	141	195	247	247	247	247
247	247	247	247	247	228	173	161	125	125	123	123	134	149	203	247	247	247	247
247	247	247	247	247	234	181	151	109	107	109	118	146	175	221	247	247	247	247
247	247	247	247	247	241	206	175	127	107	109	136	182	210	238	247	247	247	247
247	247	247	247	247	247	231	212	158	119	121	166	221	238	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	190	244	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	190	244	247	247	245	245	246	247	247
247	247	247	247	246	246	246	240	188	137	141	191	242	245	247	245	245	243	244	242
247	247	247	247	246	246	246	240	207	171	175	206	242	245	247	245	245	243	244	242
247	247	247	247	246	246	246	240	225	207	212	225	243	245	247	247	247	244	244	242
247	247	247	247	247	247	247	247	247	242	239	239	244	247	247	247	247	247	247	247

Kernel Convolutions

Mean Blur

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	120	95	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	247	179	195	247	247	247	247	247
247	247	247	247	247	247	201	83	247	83	83	247	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	247	83	83	239	83	92	247	247	247	247	247
247	247	247	247	247	247	190	83	134	83	83	131	83	165	247	247	247	247	247
247	247	247	247	247	247	239	99	83	83	83	113	163	239	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	83	83	239	247	247	247	247	247	247	247
247	247	228	247	247	247	247	247	247	83	83	239	247	247	247	244	235	247	247	247
228	228	228	190	228	228	245	245	183	111	111	240	228	228	228	244	244	228	224	224
247	247	228	228	247	247	245	247	247	247	225	234	247	247	244	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247

Kernel

1	1	1
1	1	1
1	1	1

247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	233	216	216	230	247	247	247	247	247	247	247
247	247	247	247	247	247	247	247	215	180	178	211	246	247	247	247	247	247	247
247	247	247	247	247	247	247	247	196	143	141	192	245	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	182	231	234	241	247	247	247	247
247	247	247	247	247	242	224	224	174	138	135	164	196	198	224	247	247	247	247
247	247	247	247	247	236	199	199	156	138	135	146	160	163	207	247	247	247	247
247	247	247	247	247	229	175	175	138	138	135	135	138	141	195	247	247	247	247
247	247	247	247	247	228	173	173	138	138	135	135	138	141	195	247	247	247	247
247	247	247	247	247	228	173	161	125	125	123	123	134	149	203	247	247	247	247
247	247	247	247	247	234	181	151	109	107	109	118	146	175	221	247	247	247	247
247	247	247	247	247	241	206	175	127	107	109	136	182	210	238	247	247	247	247
247	247	247	247	247	247	231	212	158	119	121	166	221	238	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	190	244	247	247	247	247	247	247
247	247	247	247	247	247	247	247	192	138	135	190	244	247	247	245	245	246	247	247
247	247	247	247	246	246	246	240	188	137	141	191	242	245	247	245	245	243	244	242
247	247	247	247	246	246	246	240	207	171	175	206	242	245	247	245	245	243	244	242
247	247	247	247	246	246	246	240	225	207	212	225	243	245	247	247	247	244	244	242
247	247	247	247	247	247	247	247	247	242	239	239	244	247	247	247	247	247	247	247

Kernel Convolutions

Blur

Kernel

1	1	1
1	1	1
1	1	1

Kernel Convolutions

Sharpen

Kernel

0	-1	0
-1	5	-1
0	-1	0

Kernel Convolutions

Edge
Highlight

Kernel

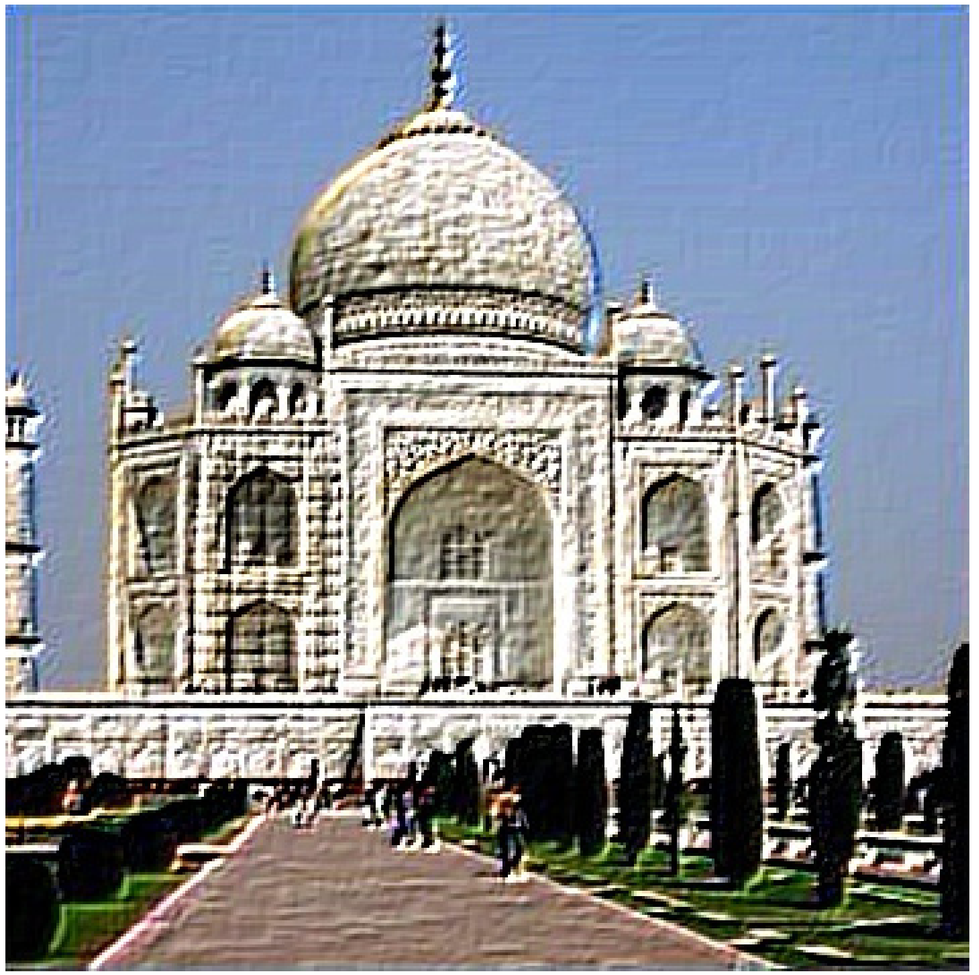
0	1	0
1	-4	1
0	1	0

Kernel Convolutions

Emboss

Kernel

-2	-1	0
-1	1	1
0	1	2



Kernel Convolutions

Horizontal “Derivative”

Kernel

1	0	-1
2	0	-2
1	0	-1

Kernel Convolutions

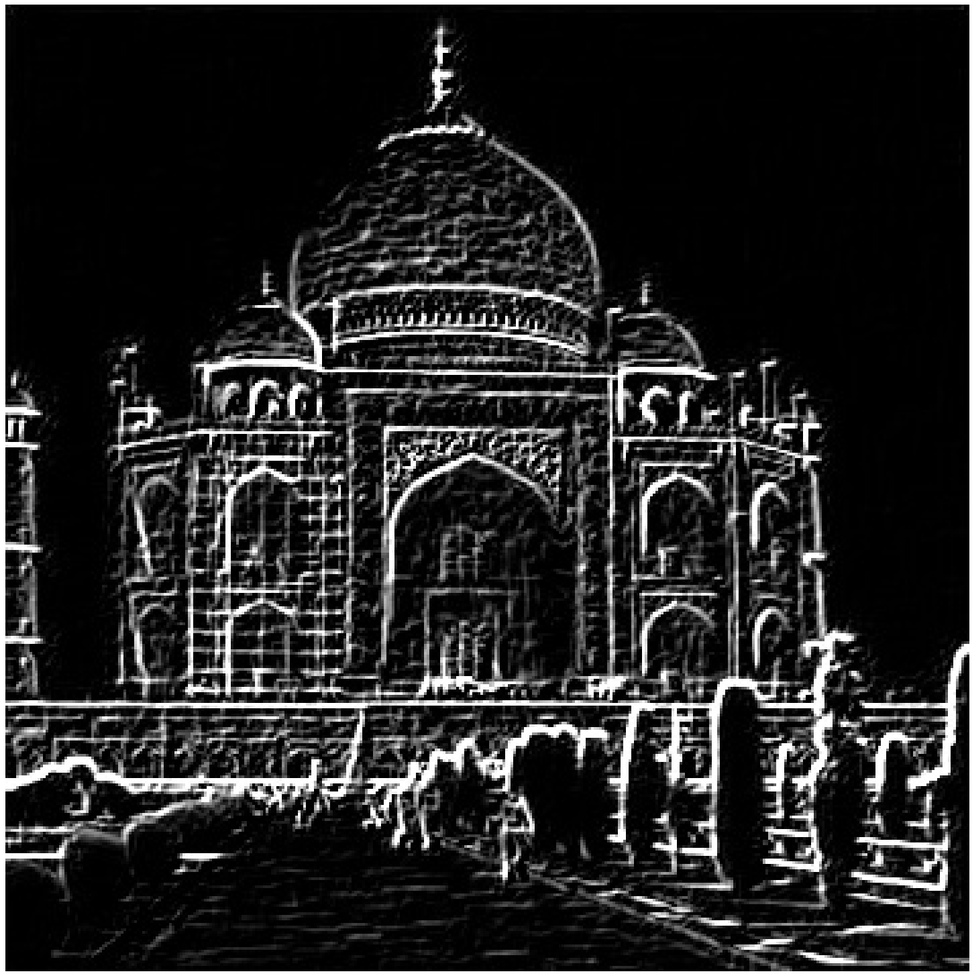
Vertical “Derivative”

Kernel

1	2	1
0	0	0
-1	-2	-1

Kernel Convolutions

Overlay them to get **Sobel Operator**

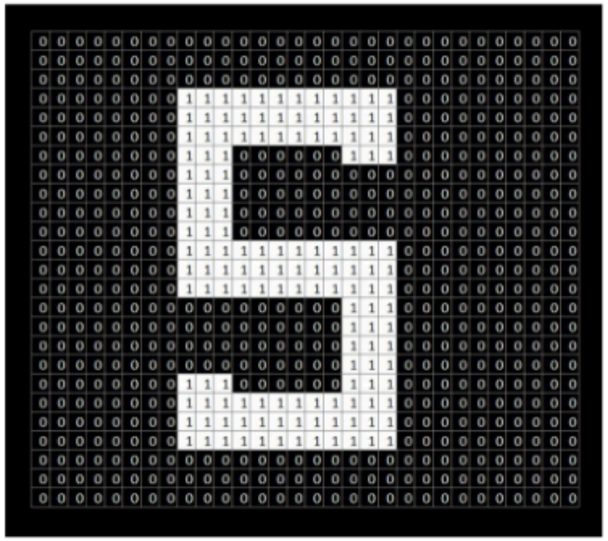


```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
import math

# Load an image
image = cv2.imread('./images/Taj_Mahal.jpg', cv2.IMREAD_GRAYSCALE)

# Define different kernels
kernels = {
    "Blur": np.ones((5, 5), np.float32) / 25,
    "Sharpen": np.array([[0, -1, 0], [-1, 5, -1], [0, -1, 0]]),
    "Edge Detection": np.array([[-1, -1, -1], [-1, 8, -1], [-1, -1, -1]]),
    "Emboss": np.array([[-2, -1, 0], [-1, 1, 1], [0, 1, 2]]),
    "Horizontal Derivative (Sobel X)": np.array([[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]]),
    "Vertical Derivative (Sobel Y)": np.array([[-1, -2, -1], [0, 0, 0], [1, 2, 1]]),
    "Gaussian Blur": np.array([[1, 2, 1], [2, 4, 2], [1, 2, 1]], np.float32) / 16,
    "Laplacian Edge Detection": np.array([[0, -1, 0], [-1, 4, -1], [0, -1, 0]])
}

# Calculate the number of subplots needed
num_kernels = len(kernels) + 1  # +1 for the original image
num_cols = 3  # Define number of columns you want in the grid
num_rows = math.ceil(num_kernels / num_cols)  # Calculate rows required

# Apply each kernel to the image
filtered_images = {name: cv2.filter2D(image, -1, kernel) for name, kernel in kernels.items()}

# Plot original and filtered images
plt.figure(figsize=(12, 10))
plt.subplot(num_rows, num_cols, 1)
plt.imshow(image, cmap='gray')
plt.title("Original Image")
plt.axis("off")

for i, (name, filtered_img) in enumerate(filtered_images.items(), 2):
    plt.subplot(num_rows, num_cols, i)
    plt.imshow(filtered_img, cmap='gray')
    plt.title(name)
    plt.axis("off")

plt.tight_layout()  # Adjust layout to prevent overlap
plt.show()
```

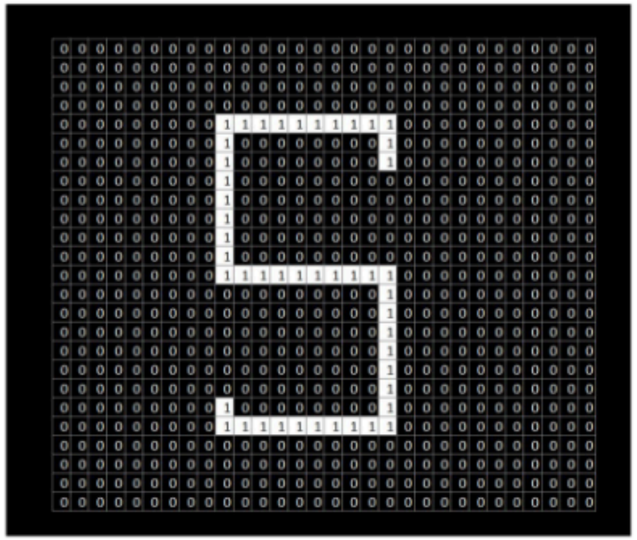
kernels in image processing

- **Dilation** – Adds pixels to the boundaries of objects in an image
- **Erosion** – Removes pixels at the boundaries of objects in an image
- **Opening** - Erosion followed by dilation
- **Closing** - Dilation followed by erosion

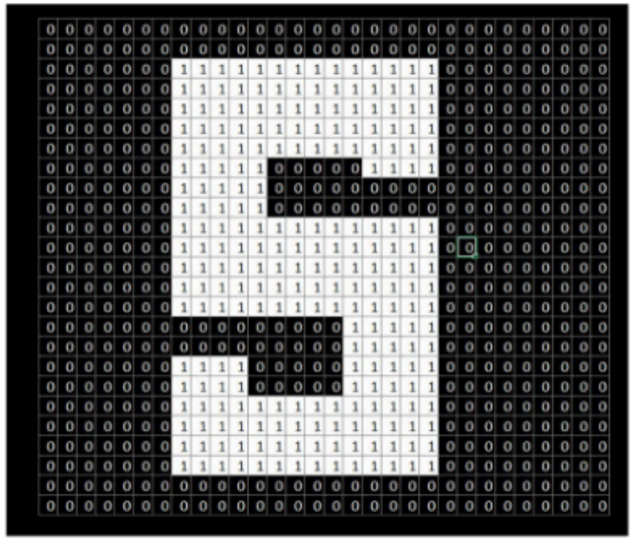
Original



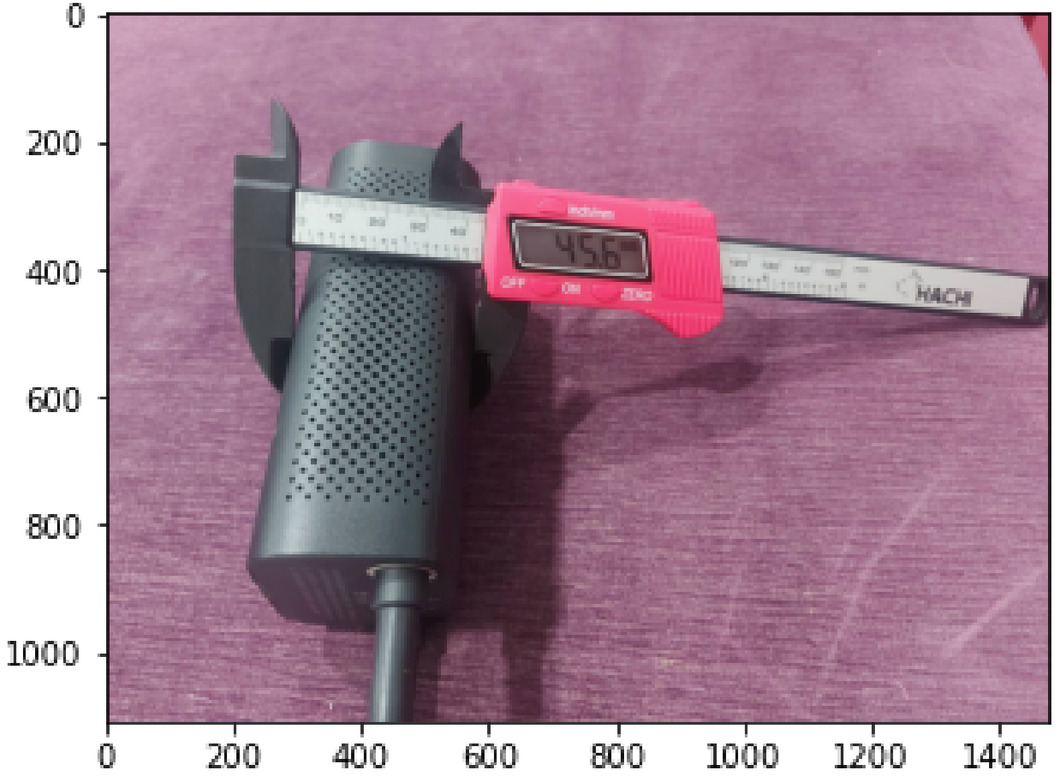
Erosion



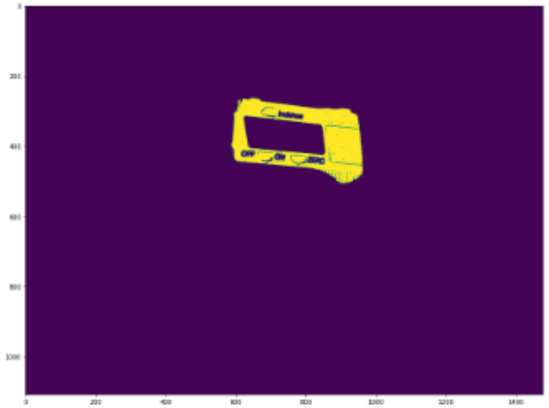
Dilation

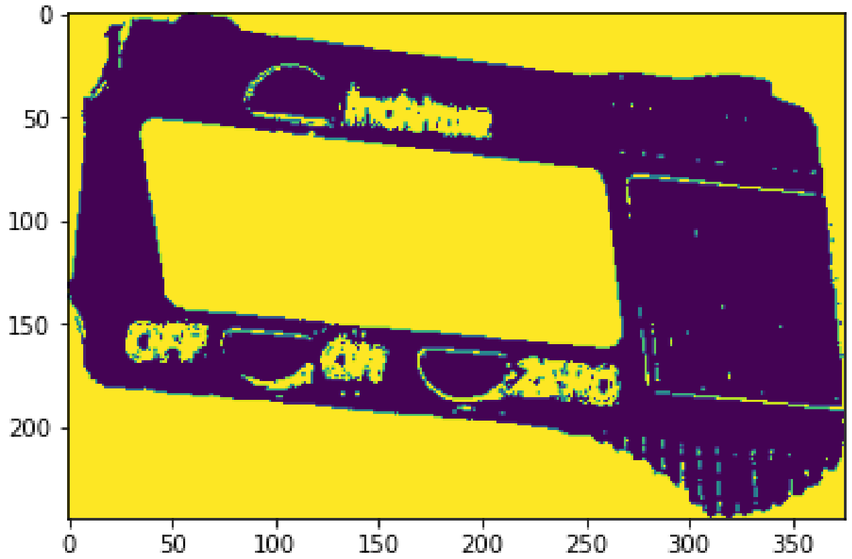


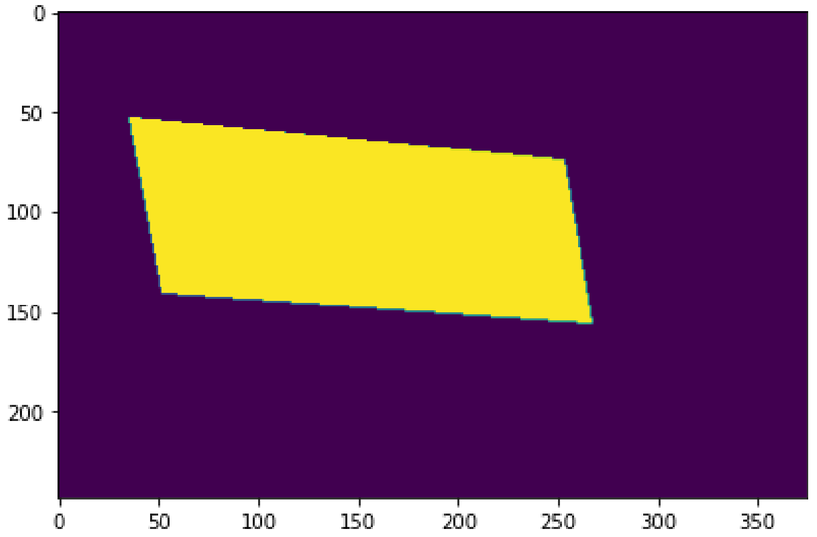
Home Works


0
200
400
600
800
1000
0
200
400
600
800
1000
1200
1400
456
HACH


456
HACH




0
50
100
150
200
0
50
100
150
200
250
300
350




```
import easyocr
reader = easyocr.Reader(['en'])
```

```
bounds = reader.readtext(img5)
bounds
```

```
[([[48, 0], [166, 0], [166, 62], [48, 62]], '4561', 0.29313910007476807)]
```

```
bounds[0][1]
```

```
'4561'
```