Deep Learning for Computer Vision

Ahmed Hosny Abdel-Gawad

Senior AI/CV Engineer

Table Content

1 From Traditional CV to DL

1.1

From Traditional Convolution Neural Networks to Deep Learning

Soft introduction about OpenCV and what exactly is traditional computer vision.

1.2

Convolution Neural Networks (CNNs)

Motivations behind CNNs, Image Classifications, calculation of sizes of filters, input and output layers.

1.3

Convolution Neural Network Meta Architectures

How to Design CNNs and the well-known architectures.

1.1 From Traditional ConvNets to DL

Semantic gap and CV tasks

The gap between what a computer sees and what we want it to see.

Traditional CV Pipeline

Traditional images processing techniques and its limitations.

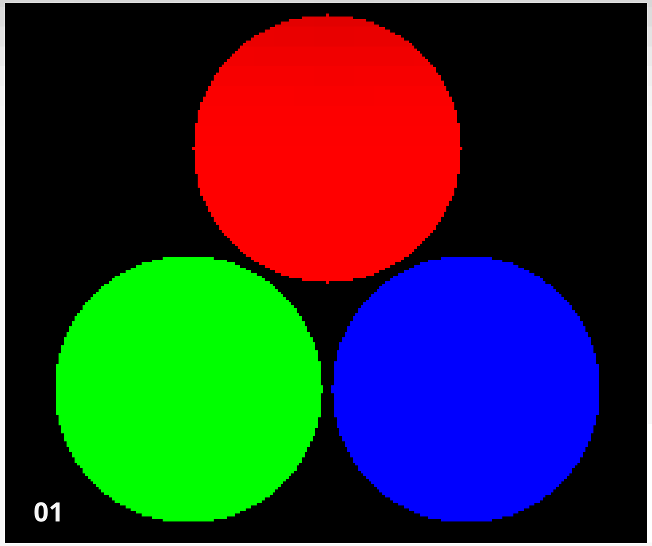
From traditional to learnable convolution filters

Introducing the learning processes and convolution neural networks.

ConvNets and GPU

Why GPUs works perfect with convolution neural networks.

Semantic Gap



01

What Does A Computer See?

Using RGB model to help us understand how a computer looks at an image.

02

What We Want the Computer to see?

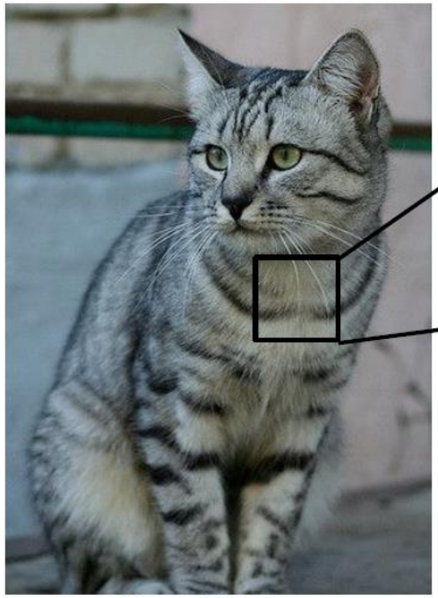
The core computer vision task: Image classification.

03

OpenCV Primer

Introduction to OpenCV.

What Does A Computer See?



This image by Nikita is
licensed under CC-BY 2.0

```
[[105 112 108 111 104  99 106  99  96 103 112 119 104  97  93  87]
 [ 91  98 102 106 104  79  98 103  99 105 123 136 110 105  94  85]
 [ 76  85  90 105 128 105  87  96  95  99 115 112 106 103  99  85]
 [ 99  81  81  93 120 131 127 100  95  98 102  99  96  93 101  94]
 [106  91  61  64  69  91  88  85 101 107 109  98  75  84  96  95]
 [114 108  85  55  55  69  64  54  64  87 112 129  98  74  84  91]
 [133 137 147 103  65  81  80  65  52  54  74  84 102  93  85  82]
 [128 137 144 140 109  95  86  70  62  65  63  63  60  73  86 101]
 [125 133 148 137 119 121 117  94  65  79  80  65  54  64  72  98]
 [127 125 131 147 133 127 126 131 111  96  89  75  61  64  72  84]
 [115 114 109 123 150 148 131 118 113 109 100  92  74  65  72  78]
 [ 89  93  90  97 108 147 131 118 113 114 113 109 106  95  77  80]
 [ 63  77  86  81  77  79 102 123 117 115 117 125 125 130 115  87]
 [ 62  65  82  89  78  71  80 101 124 126 119 101 107 114 131 119]
 [ 63  65  75  88  89  71  62  81 120 138 135 105  81  98 110 118]
 [ 87  65  71  87 106  95  69  45  76 130 126 107  92  94 105 112]
 [118  97  82  86 117 123 116  66  41  51  95  93  89  95 102 107]
 [164 146 112  80  82 120 124 104  76  48  45  66  88 101 102 109]
 [157 170 157 120  93  86 114 132 112  97  69  55  70  82  99  94]
 [130 128 134 161 139 100 109 118 121 134 114  87  65  53  69  86]
 [128 112  96 117 150 144 120 115 104 107 102  93  87  81  72  79]
 [123 107  96  86  83 112 153 149 122 109 104  75  80 107 112  99]
 [122 121 102  80  82  86  94 117 145 148 153 102  58  78  92 107]
 [122 164 148 103  71  56  78  83  93 103 119 139 102  61  69  84]]
```

What the computer sees

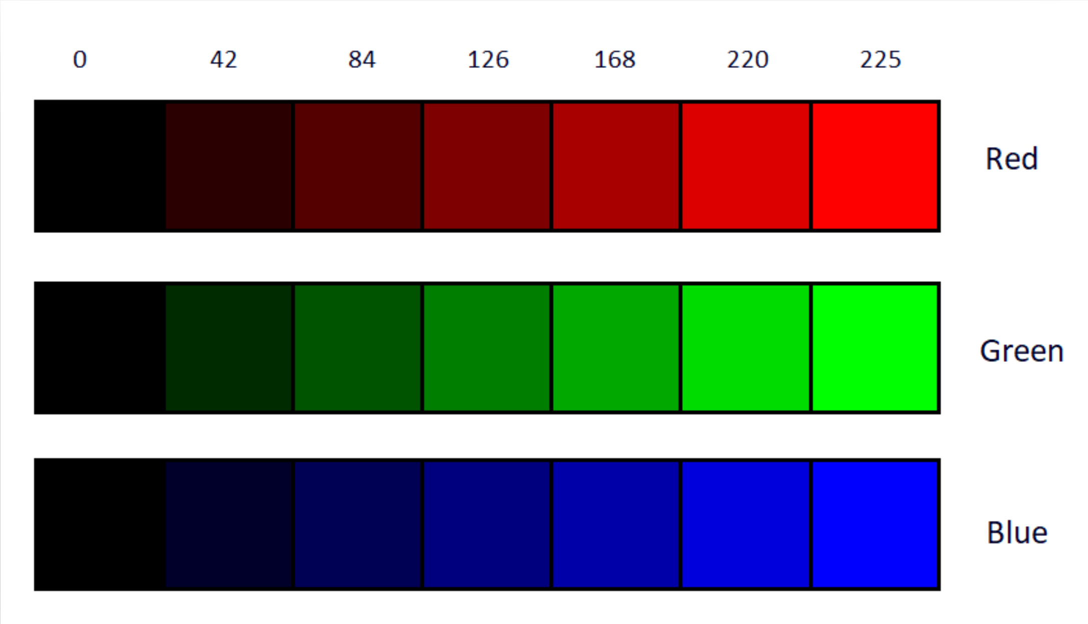
An image is just a big grid of numbers between [0, 255]:

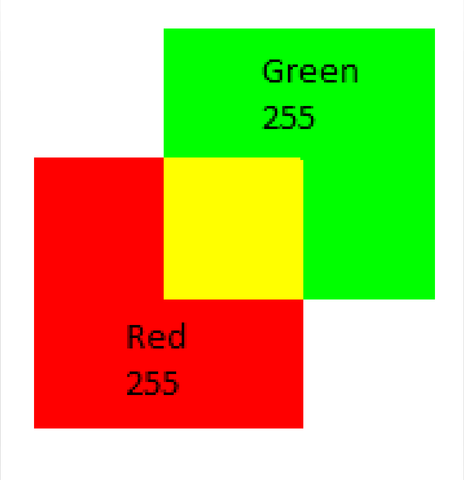
e.g. 800 x 600 x 3
(3 channels RGB)

What Does A Computer See?

Every major color has a range from 0 to 255, we can infer that higher the value, the brighter is the color.

When we combine two colors, say red and green, the resulting color is Yellow. It is represented in the three dimensional space as 255,255,0 (R,G,B).

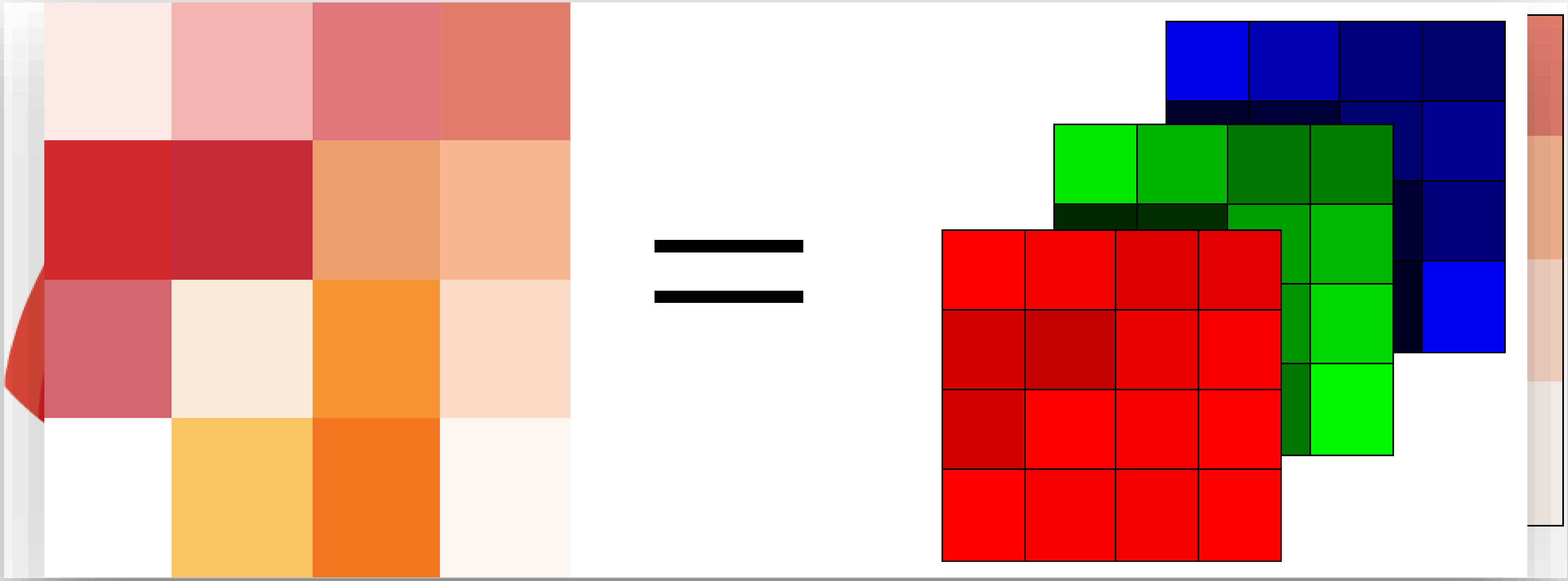




What Does A Computer See?

An image is made up of pixels placed adjacent to one another. These colored pixels are made up of three channels which are placed one behind the another.

The below pixelated image is a combination of the three channels which are placed one behind the other.



Basics of image processing with filtering

When it comes to detecting edges and contours, noise gives a great impact on the accuracy of detection. Therefore removing noises and controlling the intensity of the pixel values can help the model to focus on the general details and get higher accuracy.

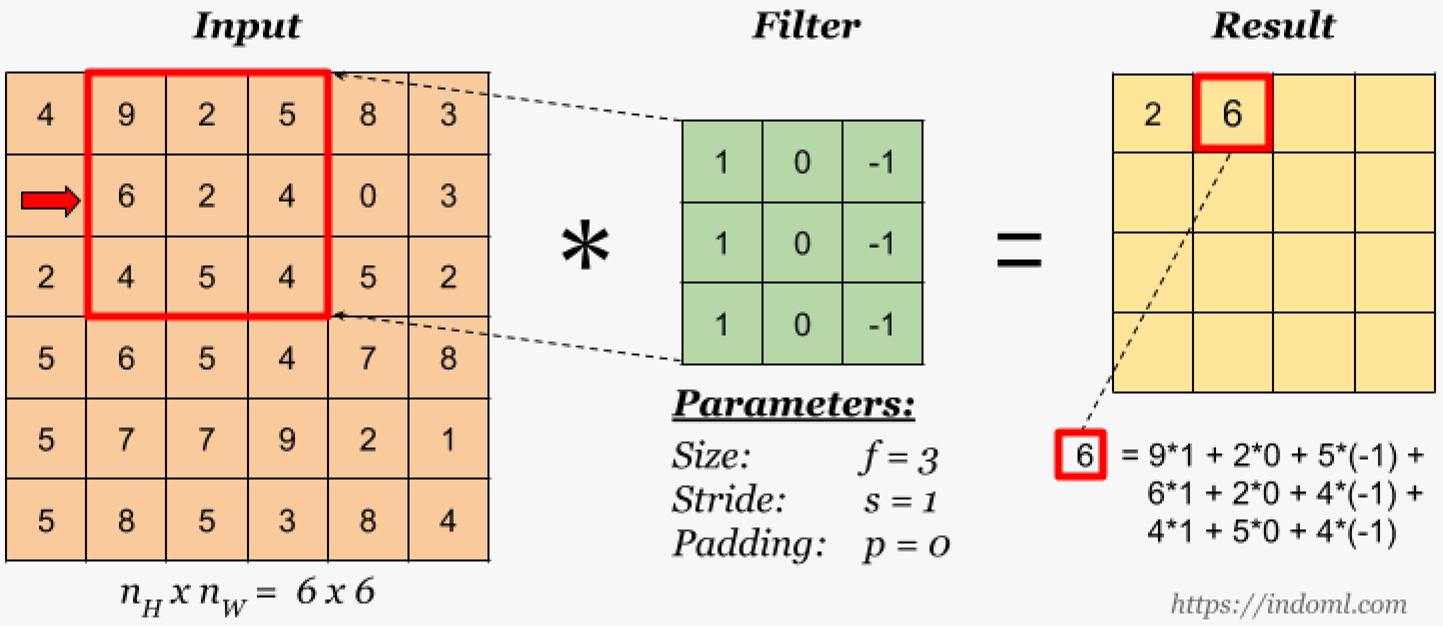
Blurring, thresholding, and morphological transformation are the techniques we use for this purpose.

Blurring

The main goal of blurring is to perform noise reduction.

Four major blurring techniques used in OpenCV: **Averaging blurring**, **Gaussian blurring**, **median blurring** and bilateral filtering.

All four techniques have a common basic principle, which is applying **convolutional operations** to the image with a filter (**kernel**). The values of the applying filters are different between the four blurring methods.



Average **Blurring**

Average blurring is taking the average of all the pixel values under the given kernel area and replace the value at the center. **Then what will it be like if we increase the size of the kernel?**

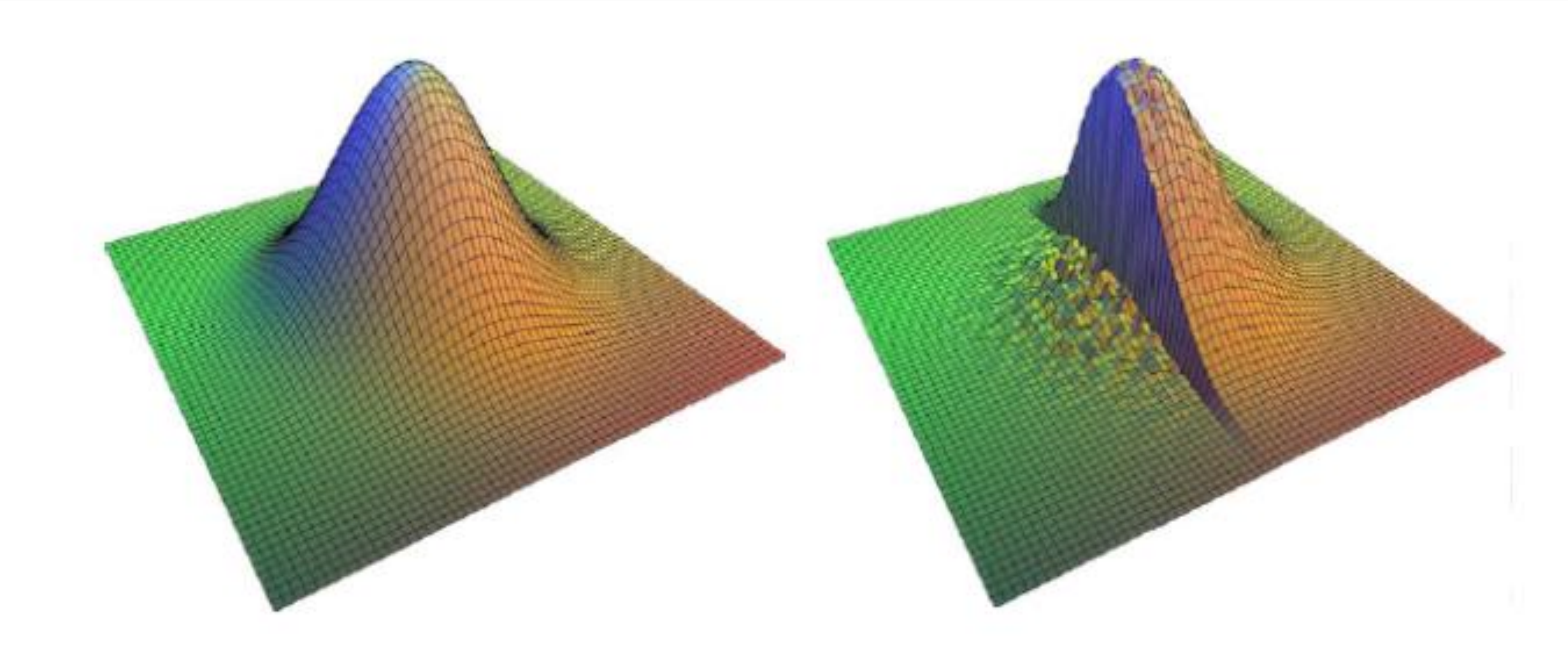
$$\frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \rightarrow$$

156	124	112	150	130	142
128	180	130	144	145	140
111	150	110	200	142	128
101	130	170	250	133	120
121	150	200	170	119	150
130	150	150	135	200	150

Gaussian/Bilateral Blurring

Gaussian blurring is nothing but using the kernel whose values have a Gaussian distribution. The values are generated by a Gaussian function so it requires a sigma value for its parameter.

Bilateral ... olving
noises bu ... s. It
uses Gau ... count.
Therefor

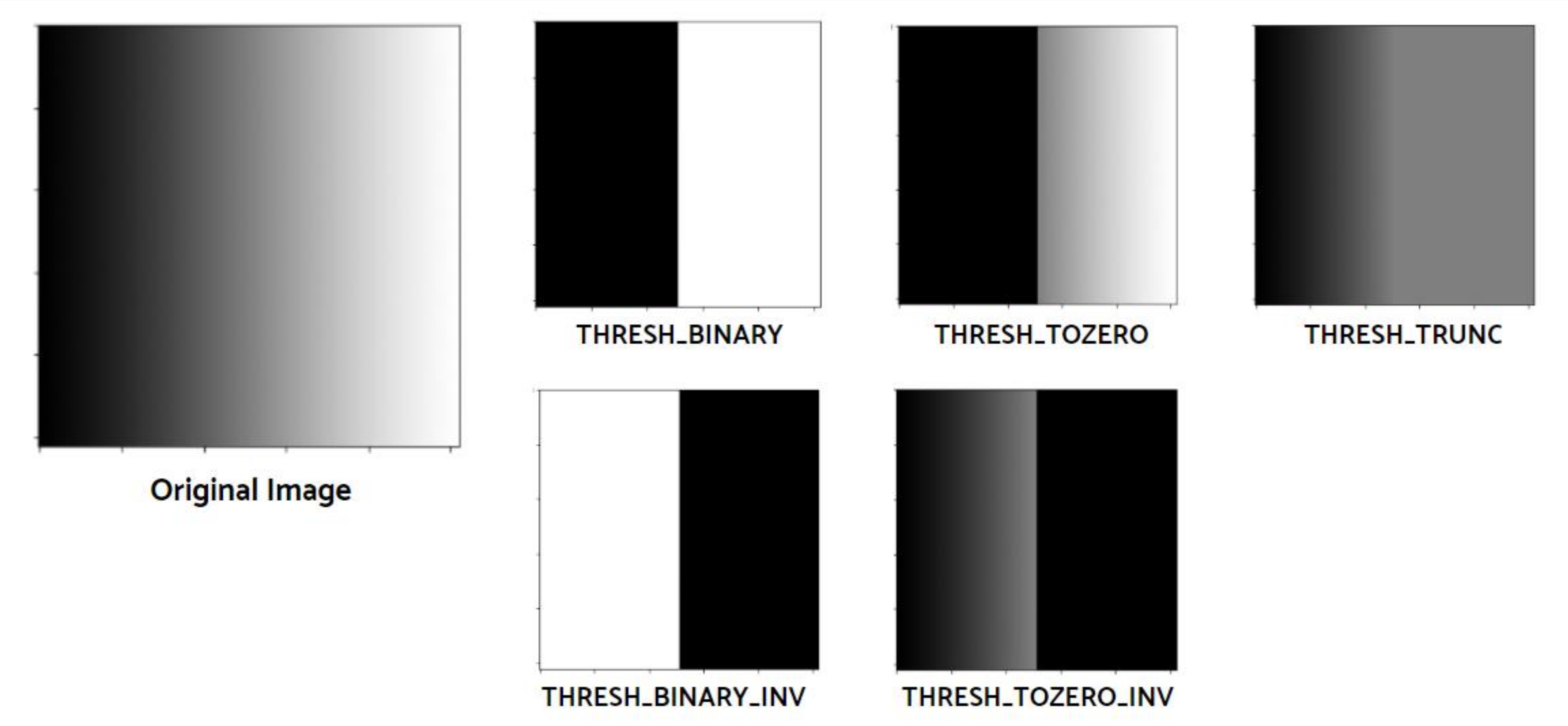


The shape of a Gaussian filter (on the left) and a Bilateral filter (on the right)

Thresholding

Thresholding transforms images into binary images.

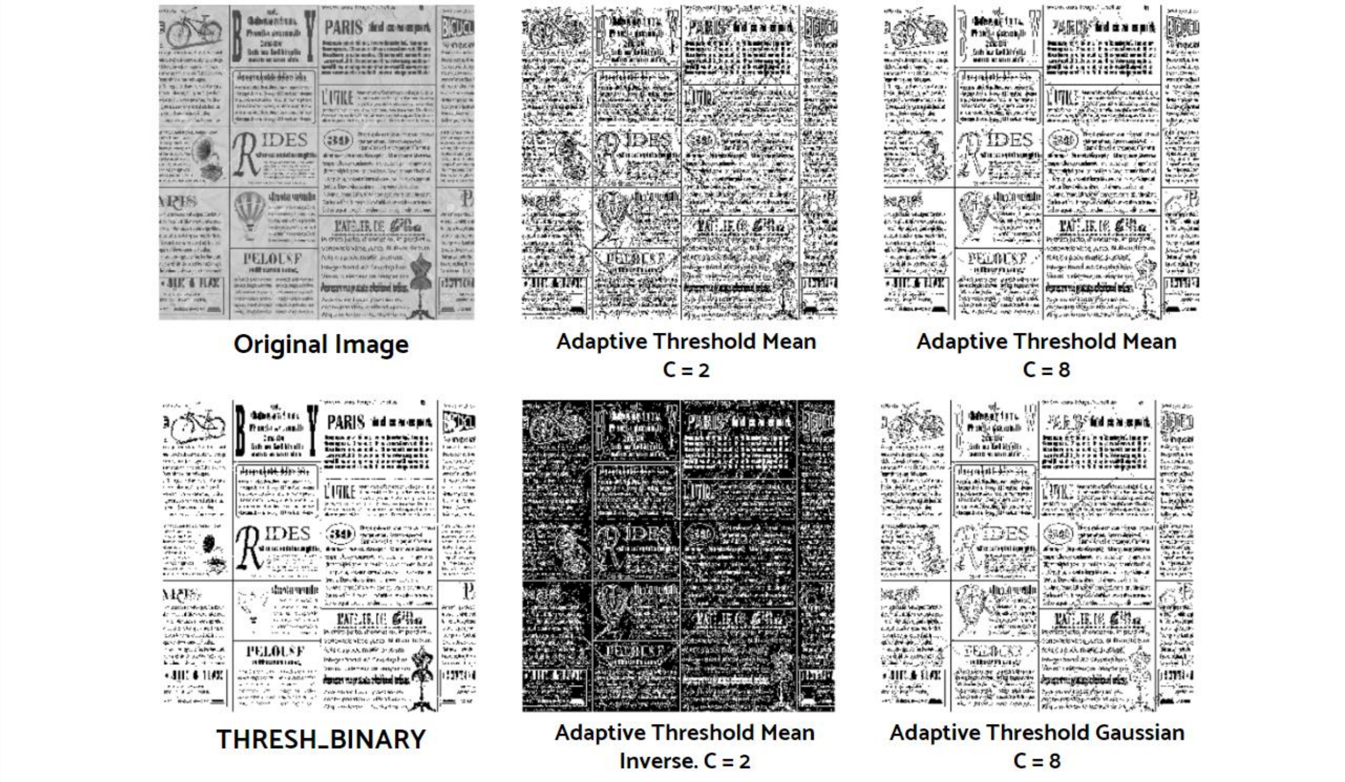
We need to set the threshold value and max values and then we convert the pixel values accordingly. There are five different types of thresholding: Binary, the inverse of Binary, Threshold to zero, the inverse of Threshold to Zero, and Threshold truncation. link



Adaptive Thresholding

What if we have a picture with various amount of lighting in different areas?

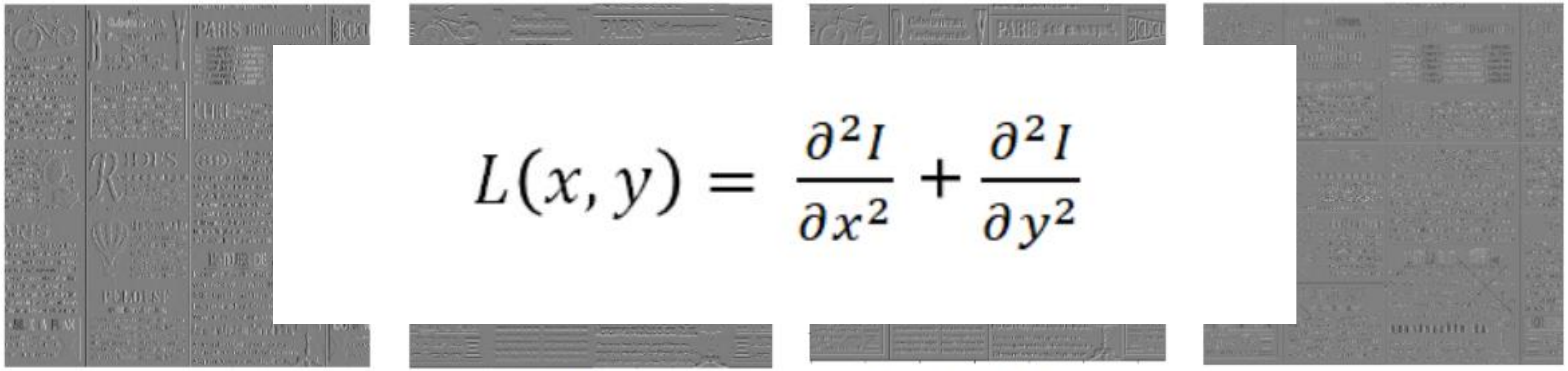
In this case, applying one value to the whole image would be a bad choice. A better approach would be using different thresholds for each part of the image. **Adaptive thresholding**, which serves this issue. By calculating the threshold within the neighborhood area of the image, we can achieve a better result from images with varying illumination.



Gradient

In mathematics, the gradient geometrically represents the slope of the graph of a function with multi-variables.

The image gradient represents directional changes in the intensity or color mode and we can use this concept for locating edges.



$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Sobel X | Sobel Y | Sobel X + Sobel Y | Laplacian

Morphological transformations

It's also possible to manipulate the figures of images by filtering, which is called as **morphological transformation**.

Erosion is the technique for shrinking figures and it's usually processed in a grayscale. The shape of filters can be a **rectangle**, an **ellipse**, and a **cross** shape. By applying a filter we remove any 0 values under the given area.

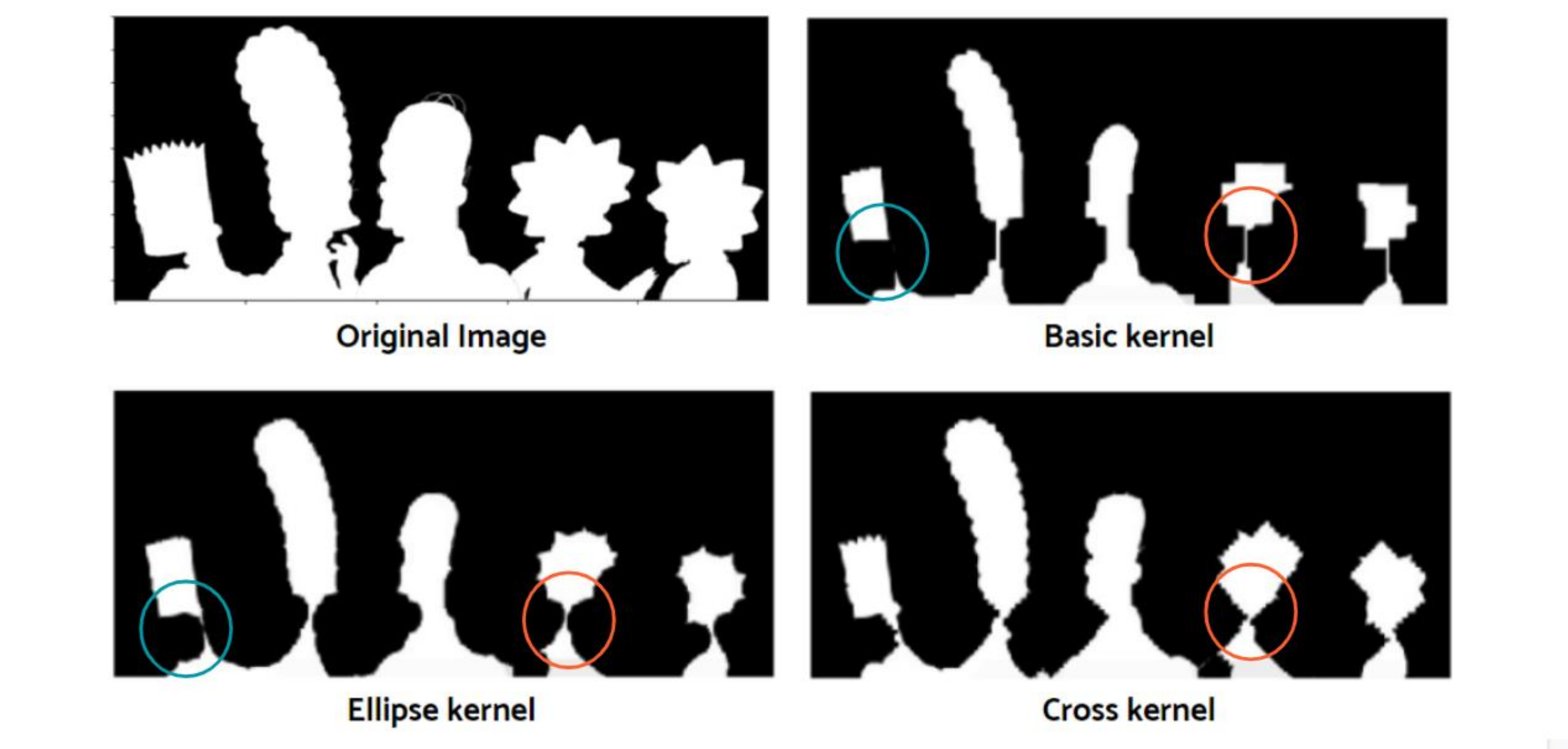
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	0	0	0	0	0

Erosion →

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Morphological transformations

Erosion is the technique for shrinking figures and it's usually processed in a grayscale. The shape of filters can be a **rectangle**, an **ellipse**, and a **cross** shape. By applying a filter we remove any 0 values under the given area.



Morphological transformations

It’s also possible to manipulate the figures of images by filtering, which is called as **morphological transformation**.

Dilation is the opposite to erosion. It is making objects expand and the operation will be also opposite to that of erosion.

0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	0	0	0	0	0

$\xrightarrow{\textbf{Dilation}}$

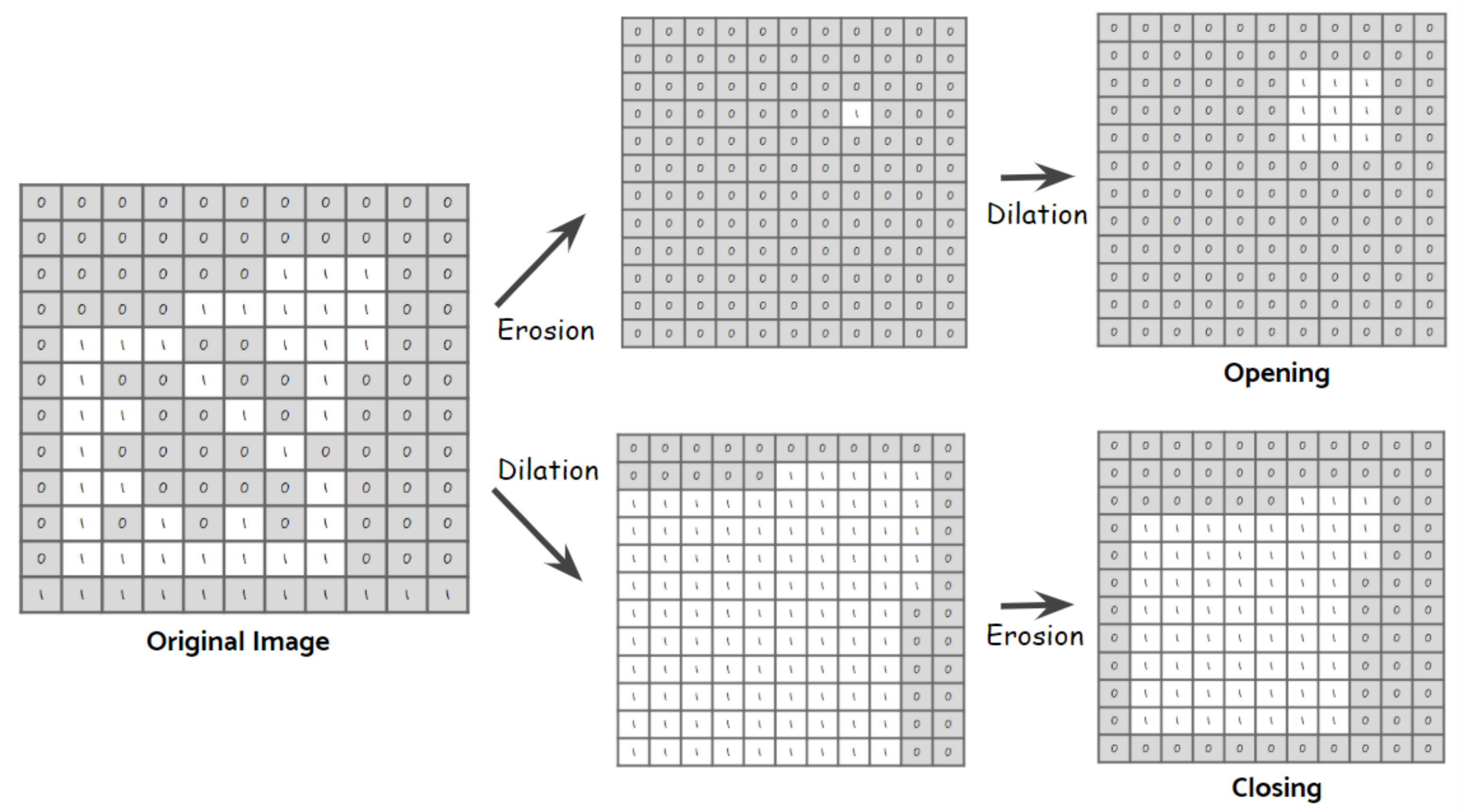
0	0	0	0	0	0	0
0	1	1	1	1	1	0
1	1	1	1	1	1	0
1	1	1	1	1	1	0
1	1	1	1	1	1	0
1	1	1	1	1	1	0
1	1	1	1	1	1	0

Morphological transformations

Dilation is the opposite to erosion. It is making objects expand and the operation will be also opposite to that of erosion.

Morphological transformations

Opening and **closing** operation is the mixed version of erosion and dilation. Opening performs erosion first and then dilation is performed on the result from the erosion while closing performs dilation first and the erosion.

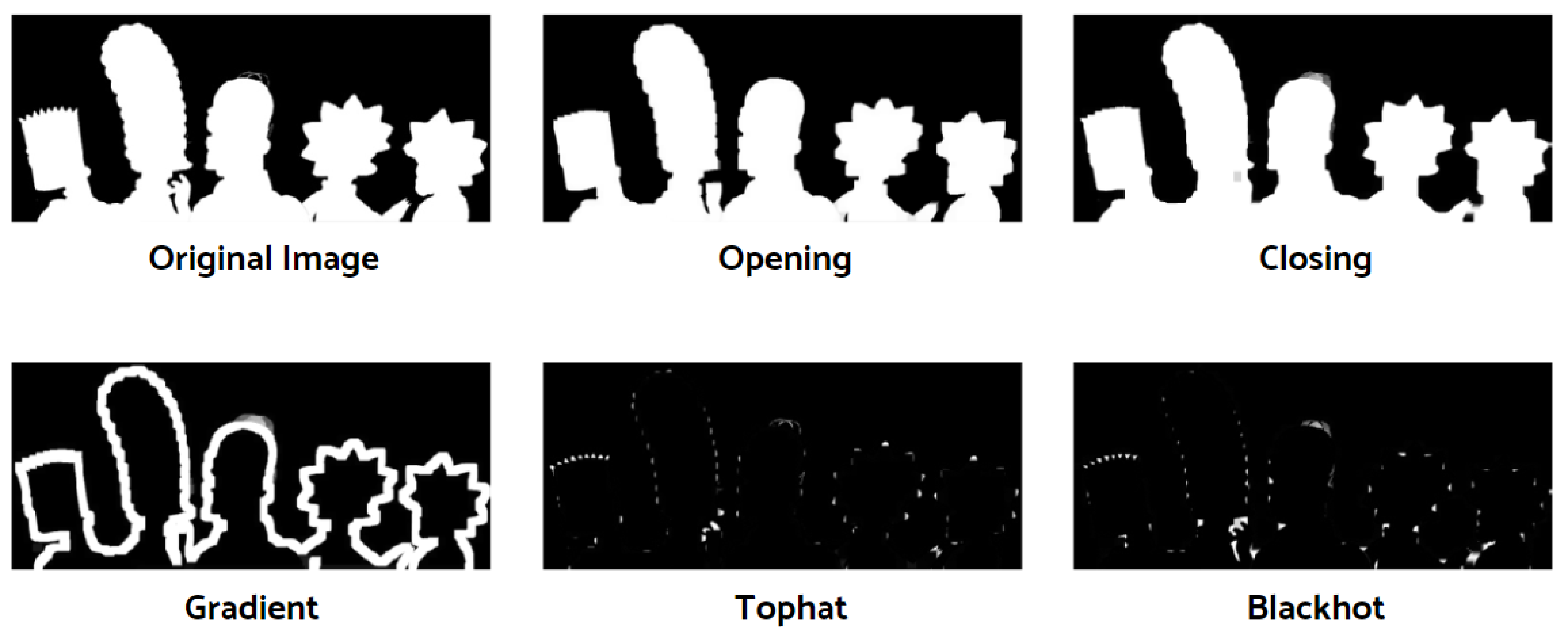


Subpatterns

Contour

Morphological transformations

Top hat filter is the subtracted area from opening to the original image while **black hat filter** is that from closing.



More...

A core task in CV:
Image Classification

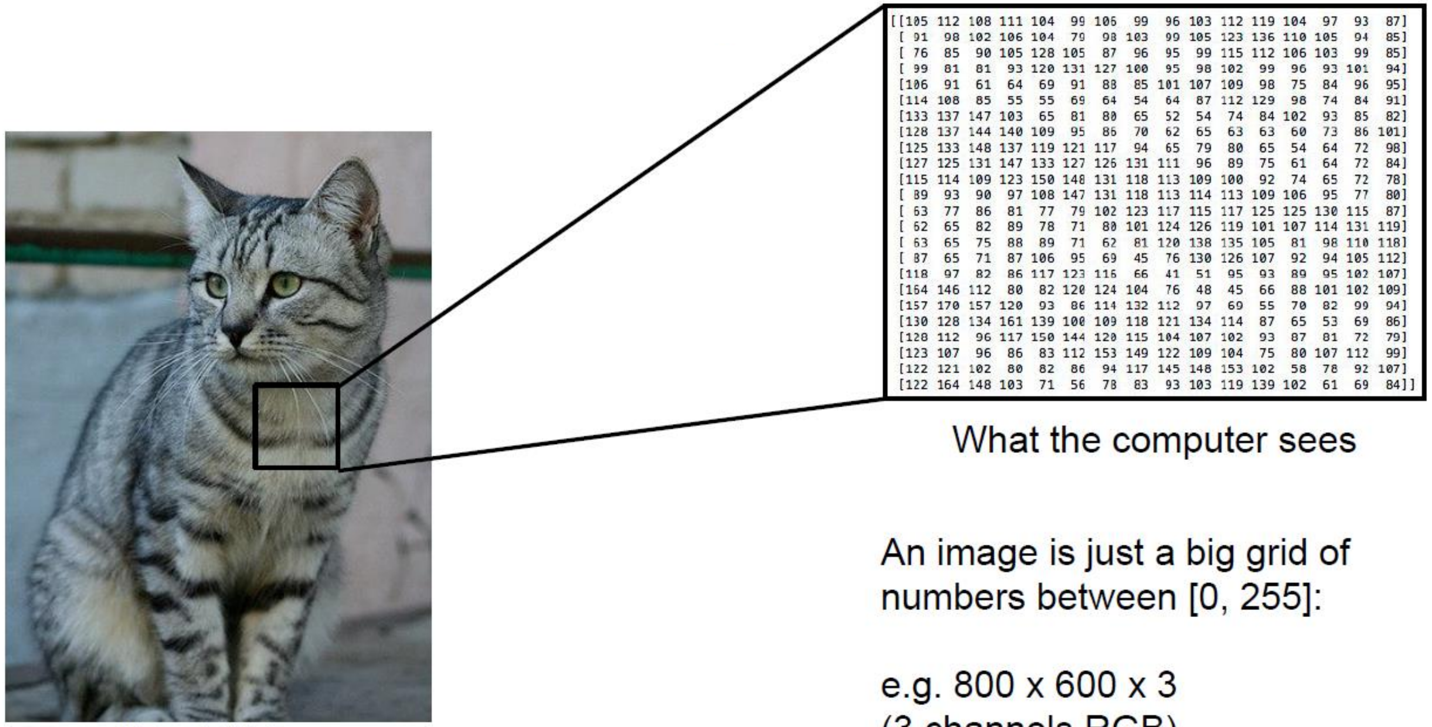
Image Classification

This image by Nikita is
licensed under CC-BY 2.0

(assume given set of discrete labels)
{dog, cat, truck, plane, ...}

→ cat

The Problem: Semantic Gap



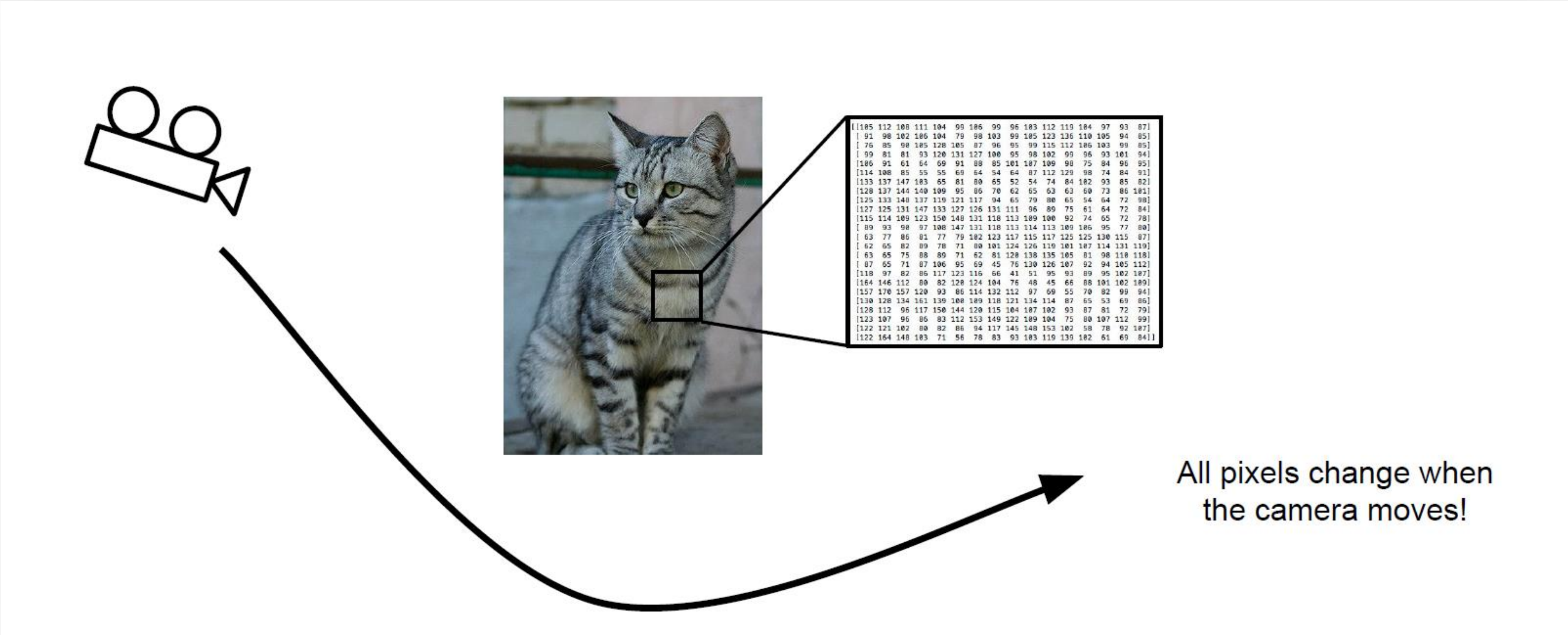
This image by Nikita is licensed under CC-BY 2.0

What the computer sees

An image is just a big grid of numbers between [0, 255]:

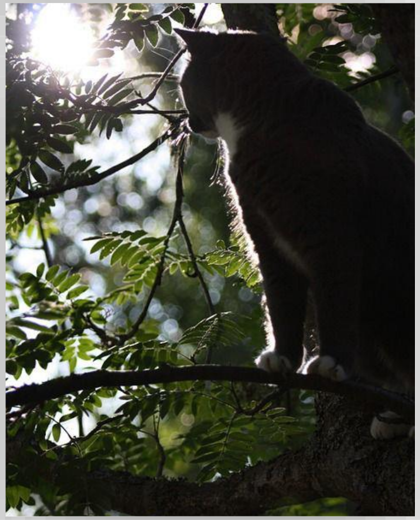
e.g. 800 x 600 x 3
(3 channels RGB)

Challenges: Viewpoint variation



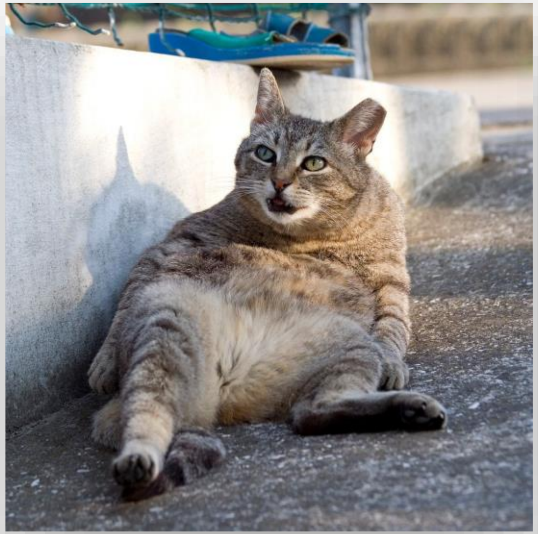
Challenges: Illumination

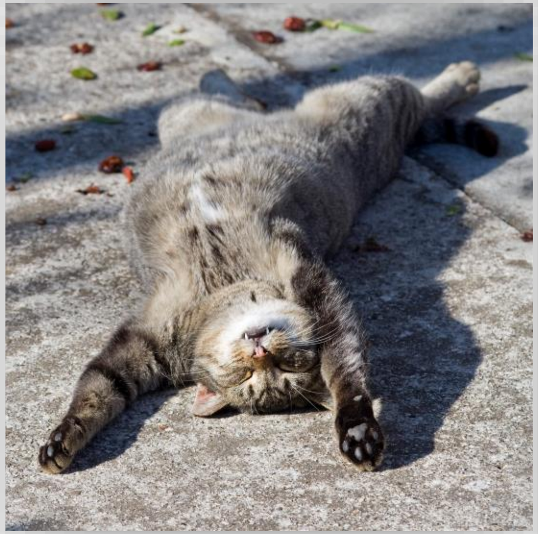


We need the computer to classify all of them as cats!

Challenges: Deformation






Again, we need the computer to classify all of them as cats!

Challenges: Occlusion

Challenges: Background Clutter

Challenges: Intraclass variation

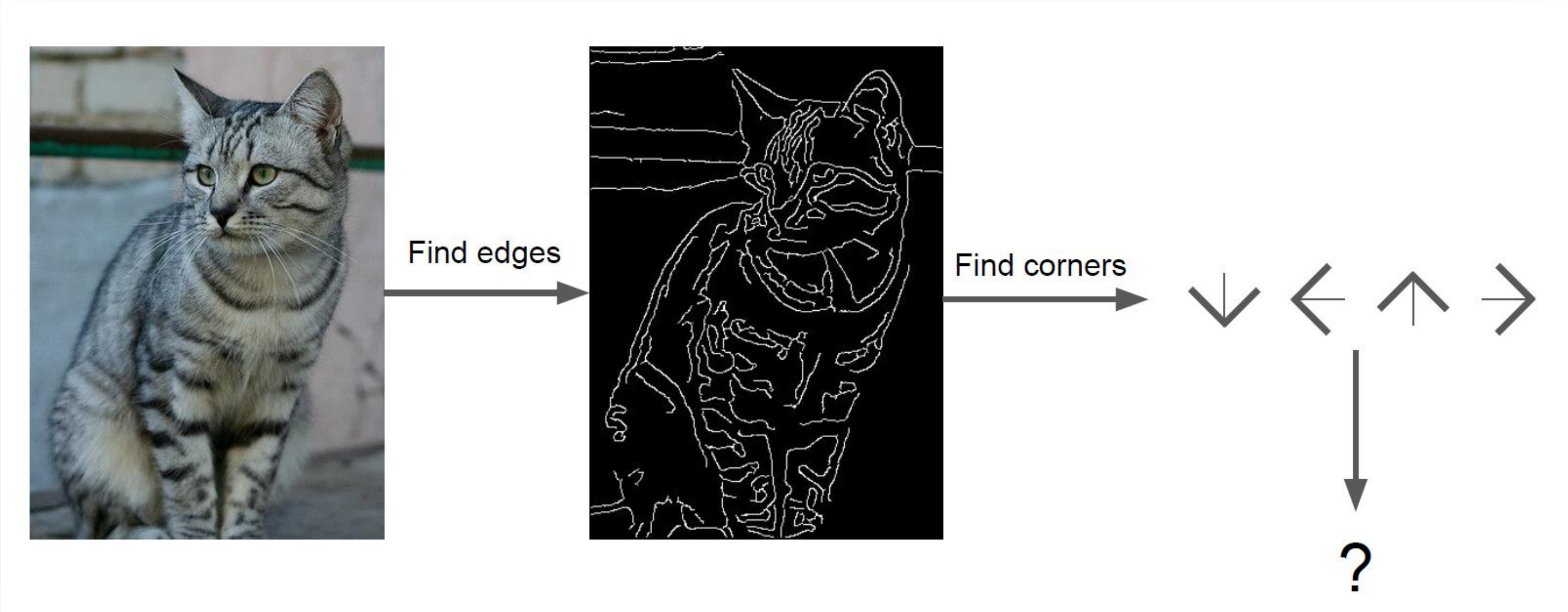
An Image Classifier

```
def classify_image(image):
  # Some magic?
  return class_label
```

Unlike e.g. sorting a list of numbers,

No obvious way to hard-code the algorithm for recognizing a cat, or other classes.

Attempts: Traditional CV

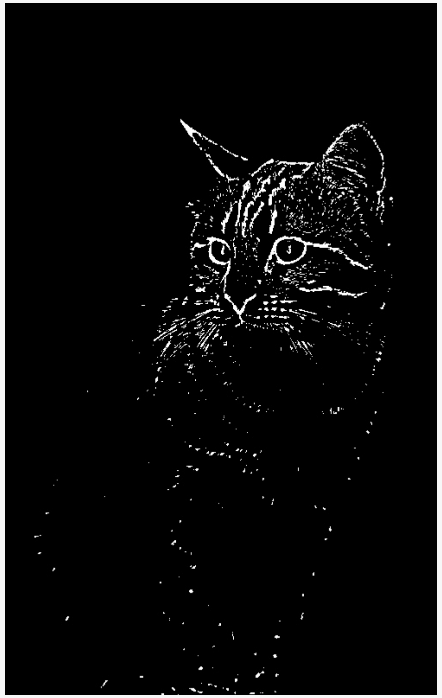


Layers of features

Pixels

Gray Pixels

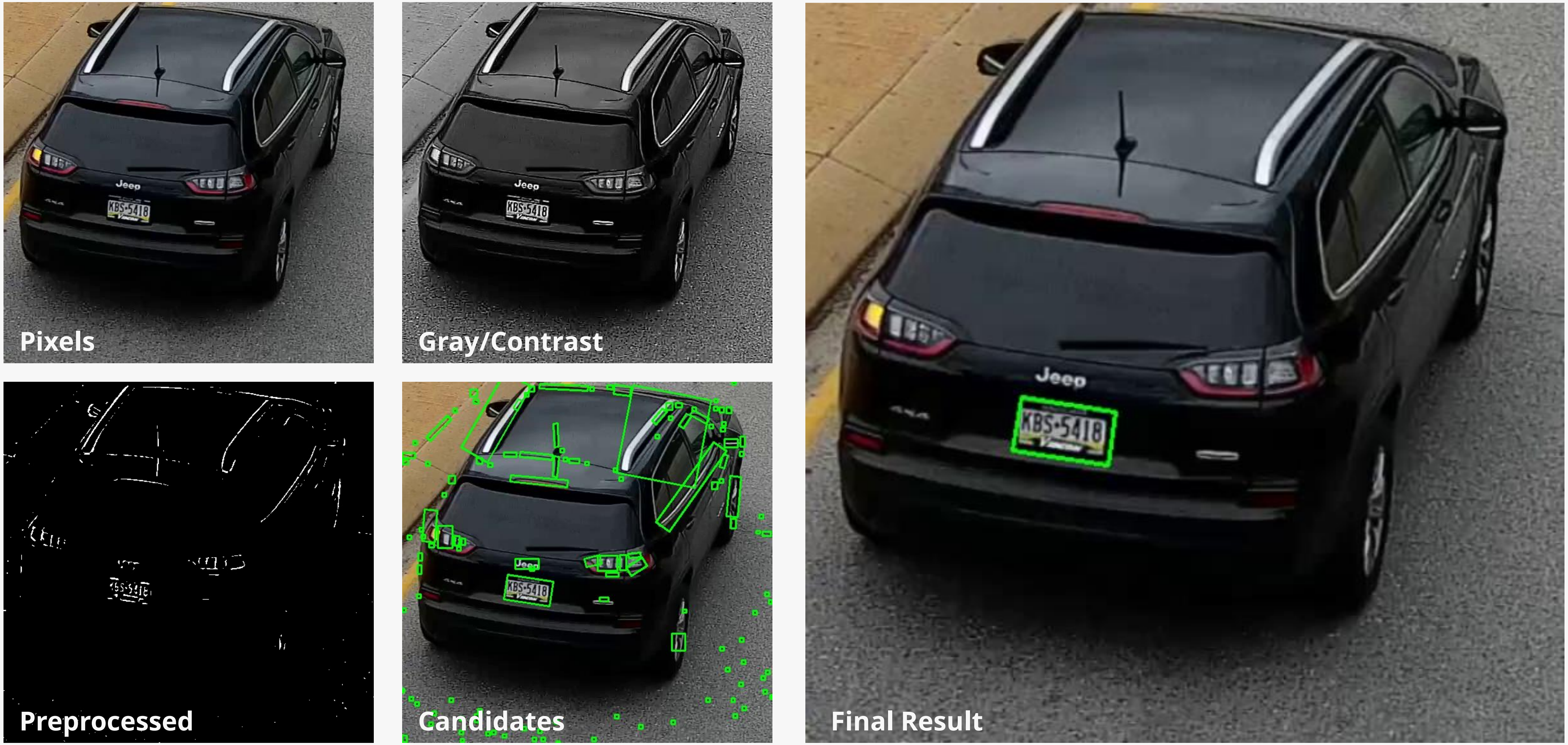
Threshold Pixels



Edges/Dilation

Contours

Traditional LPD



Thank You!