UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Dipartimento
di Ingegneria Gestionale,
dell'Informazione e della Produzione

Lesson 10.

Computer vision - part I

Introduction and classical approaches

DATA SCIENCE AND AUTOMATION COURSE

MASTER DEGREE SMART TECHNOLOGY ENGINEERING

TEACHER
Mirko Mazzoleni

Michele Ermidoro

PLACE
University of Bergamo

Outline

1. Computer vision: motivation
2. Image representation
3. Image processing
4. Computer vision: classic approaches
5. Application to strand ropes fault detection

Outline

1. **Computer vision: motivation**
2. Image representation
3. Image processing
4. Computer vision: classic approaches
5. Application to strand ropes fault detection

Computer vision and machine vision

Computer vision (CV)

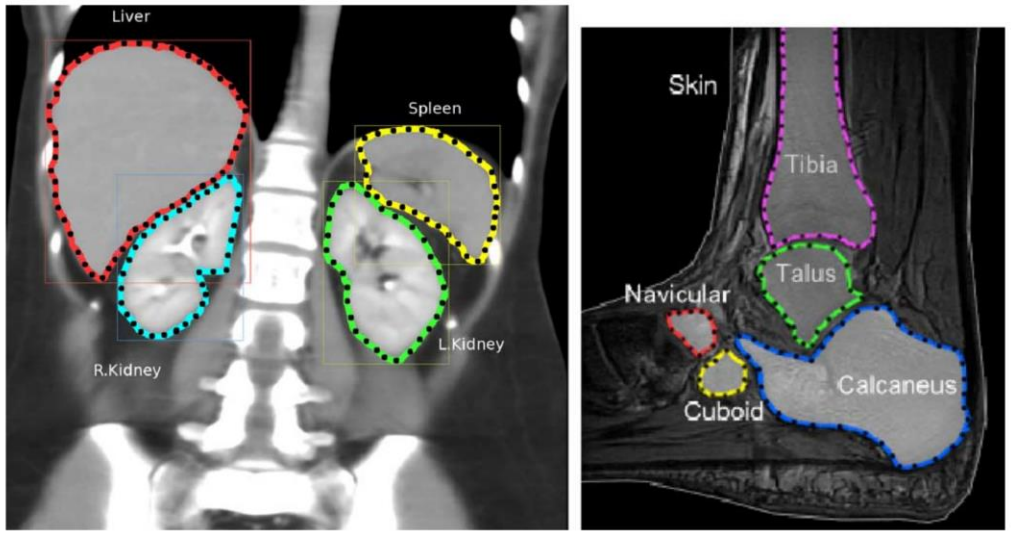
From the perspective of engineering, it seeks to automate tasks that the human visual system can do. Computer vision tasks include methods for acquiring, processing and understanding digital images. *(https://en.wikipedia.org/wiki/Computer_vision)*

Machine vision (MV)

Is the set of technologies and methods used to provide imaging-based automatic inspection and analysis for applications as automatic inspection, process control, and robot guidance, usually in industry. It attempts to integrate existing technologies in new ways and apply them to solve real world problems. *(https://en.wikipedia.org/wiki/Machine_vision)*

Computer vision: why?

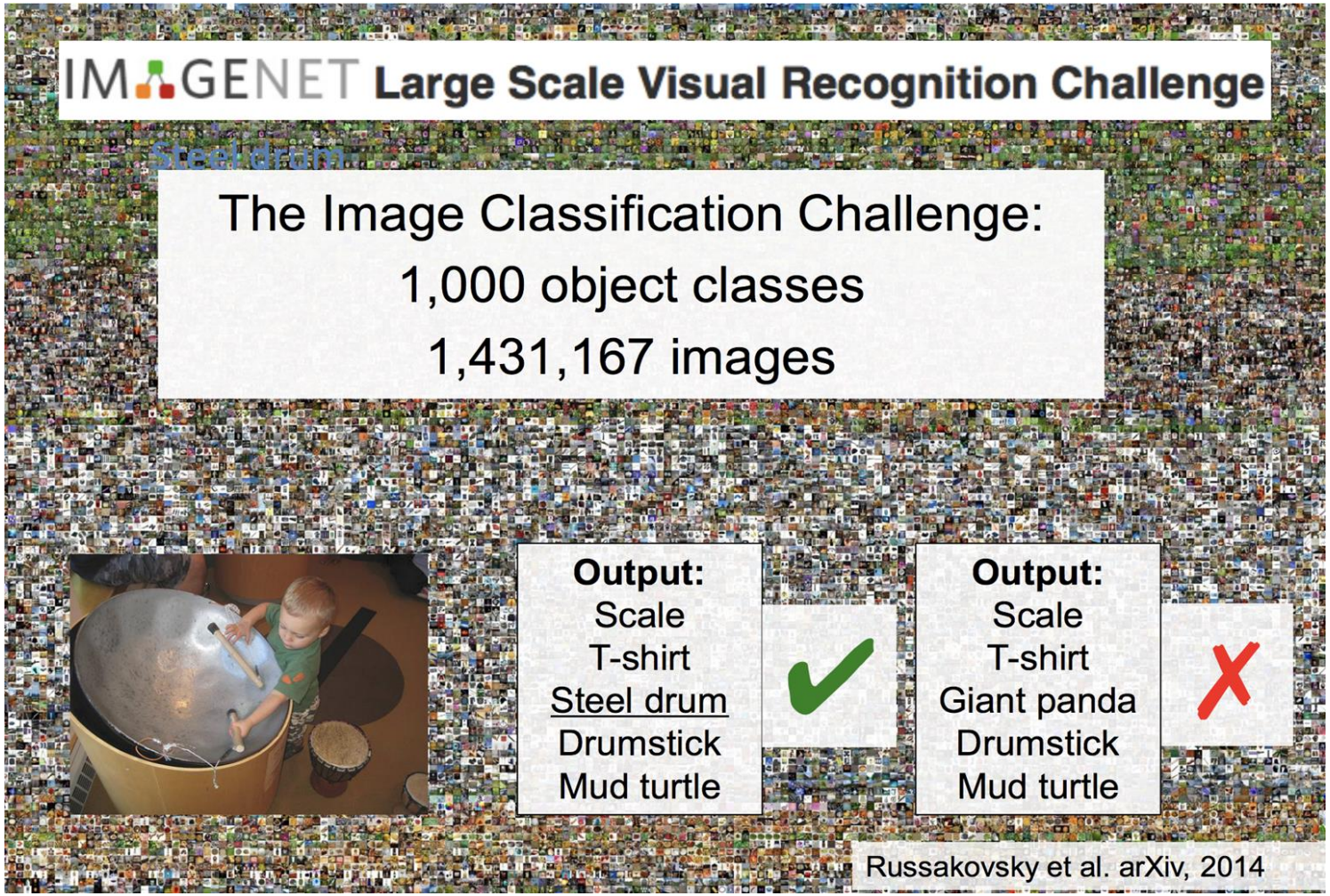
- Almost **80% of the data** traveling on the net is visual data
- Everybody has a smartphone, and every smartphone has **at least 2 cameras**
- "**A picture is worth a thousand words**" is an English language-idiom
- It has **wide** fields of application:
  - ✓ Robotics
  - ✓ Surveillance
  - ✓ Industry
  - ✓ Self-driving cars/drones/buses..
  - ✓ Medical



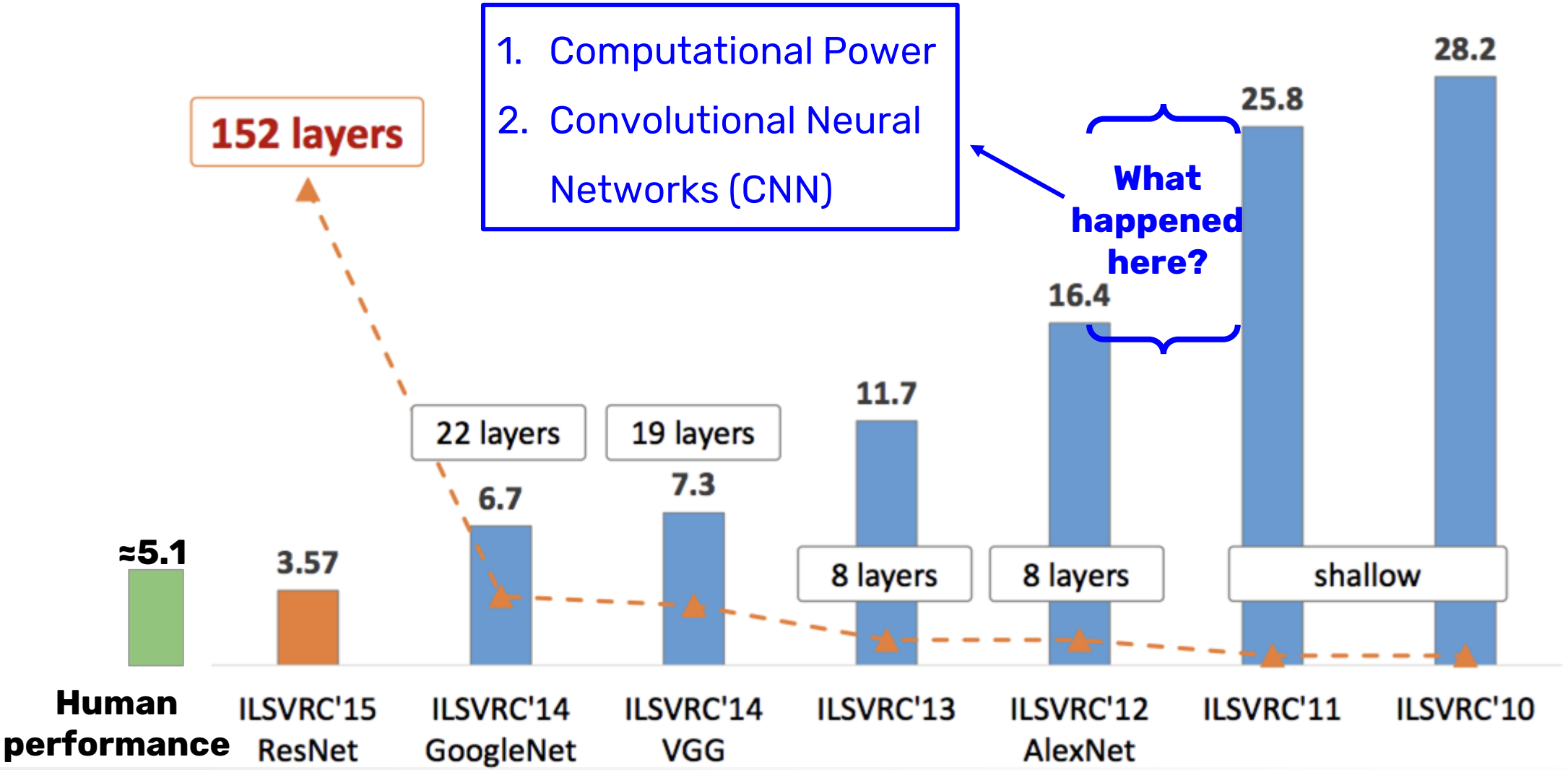
http://crcv.ucf.edu/people/faculty/Bagci/research.php

Computer vision: hype?

- Consider the **image-net** dataset *(http://www.image-net.org/challenges/LSVRC/)*
- 1000 classes
- More than 1 million images



Computer vision: hype?



Computer vision tasks

1. Classification

What's in the image?

- People
- Car
- Traffic light
- Clock
- …

http://vision.stanford.edu

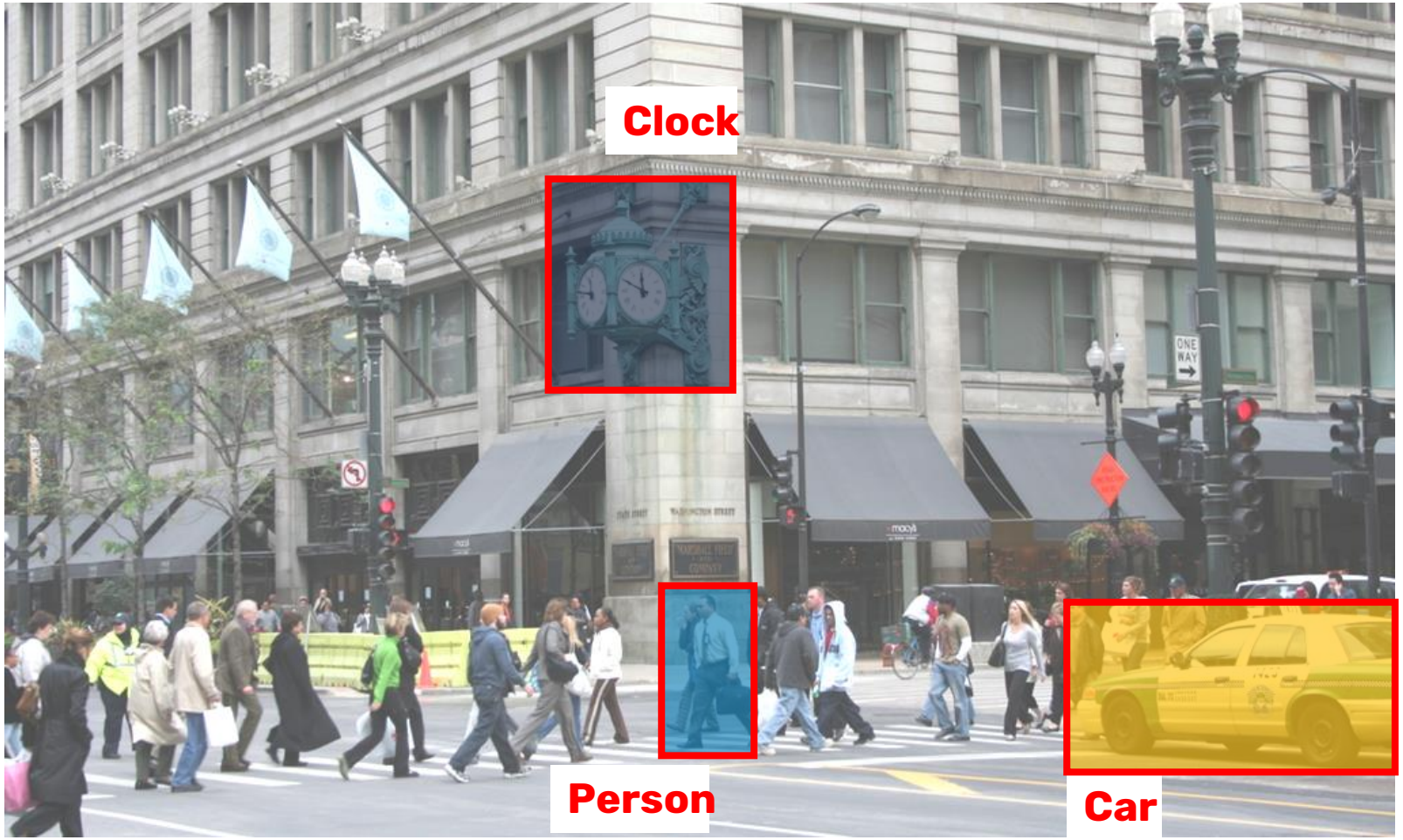
Computer vision tasks

2. Detection

What's in the image?

Where it is?

- A car in the orange box
- A person in the blue box
- A clock in the green box



http://vision.stanford.edu

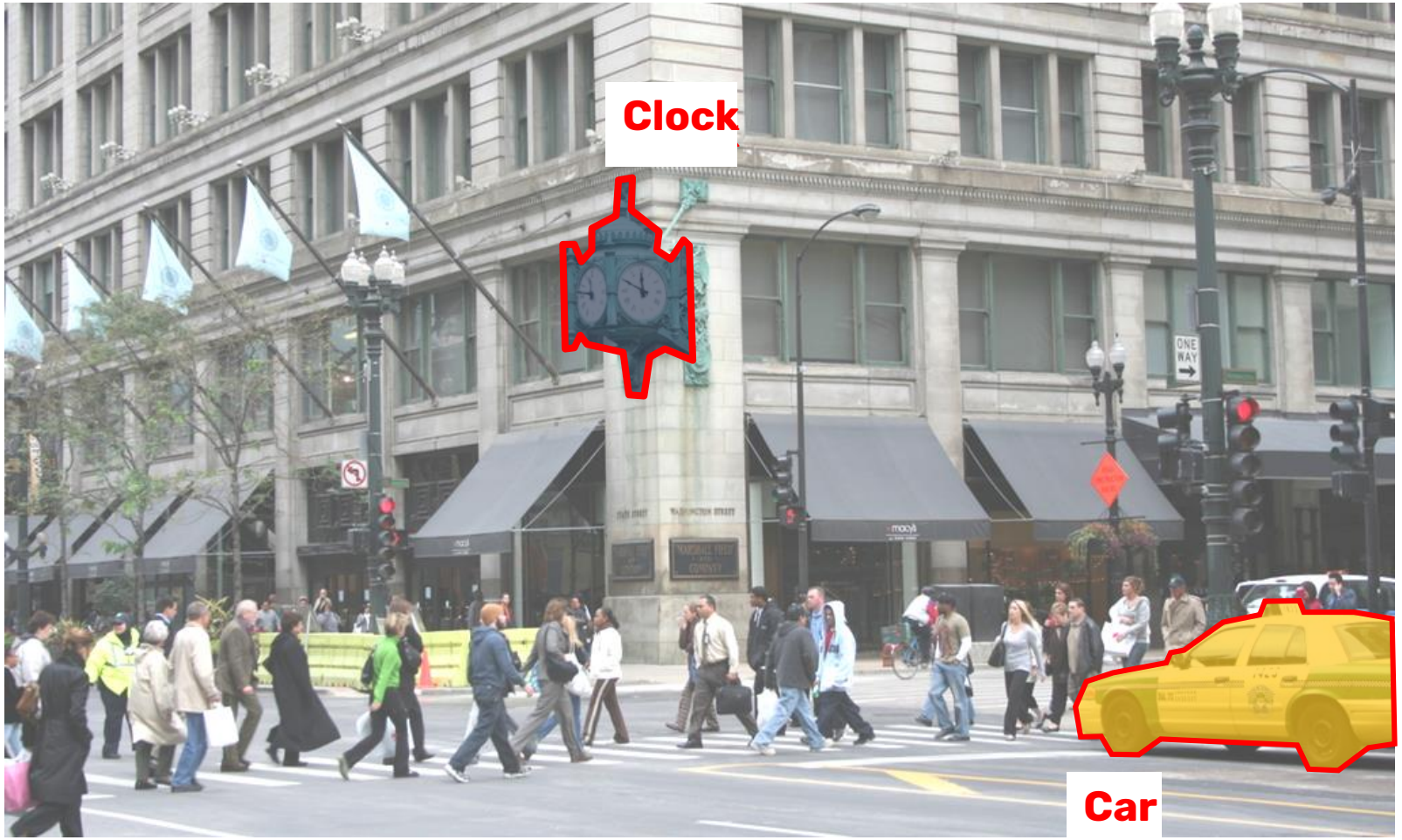
Computer vision tasks

3. Segmentation

What's in the image?

Where it is?

In **which** pixels?



http://vision.stanford.edu

Computer vision tasks

COCO Common Objects in Context

4. Annotation

How would you describe the picture?

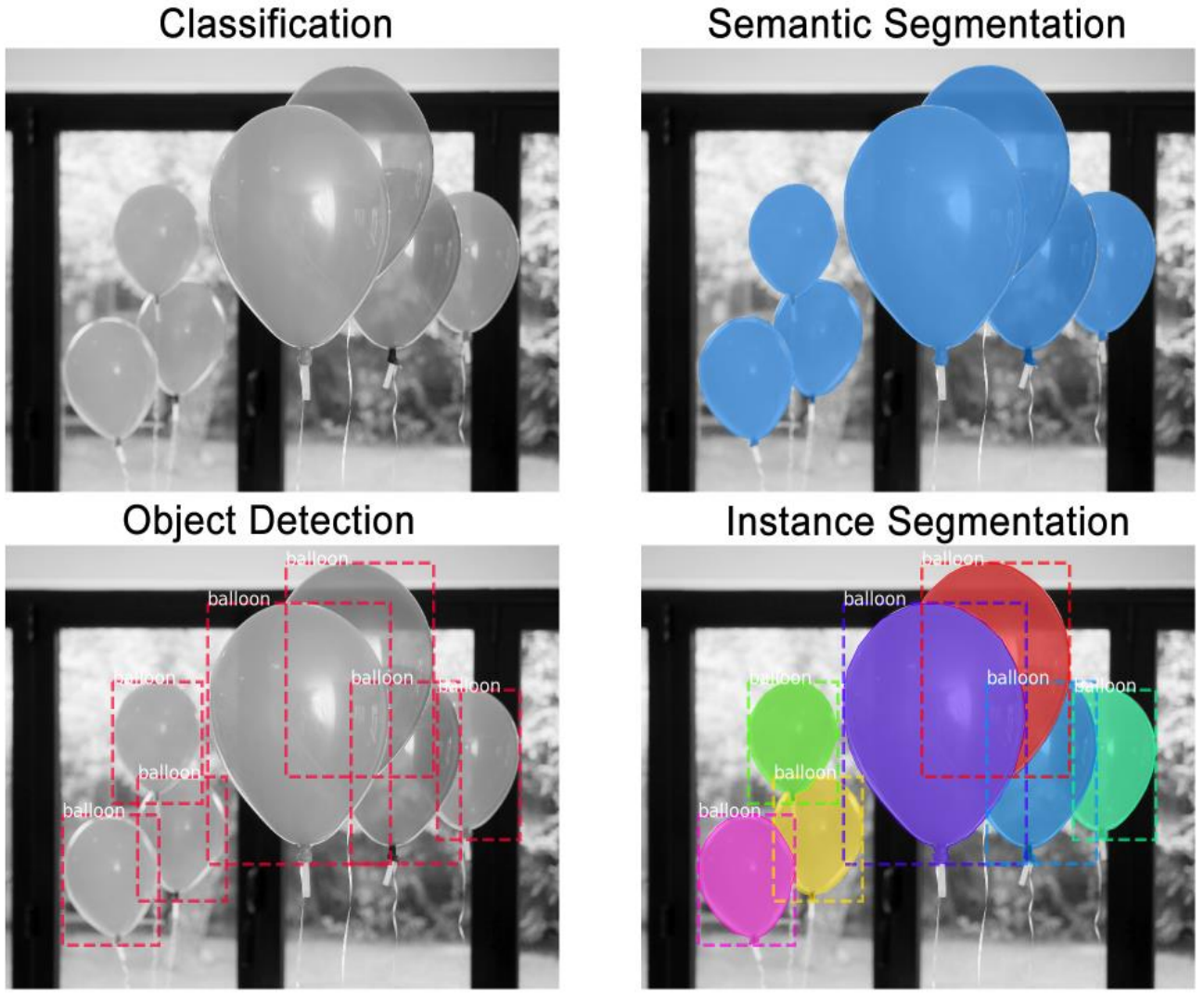
"People crossing a street while a car is waiting"

http://vision.stanford.edu

Computer vision tasks

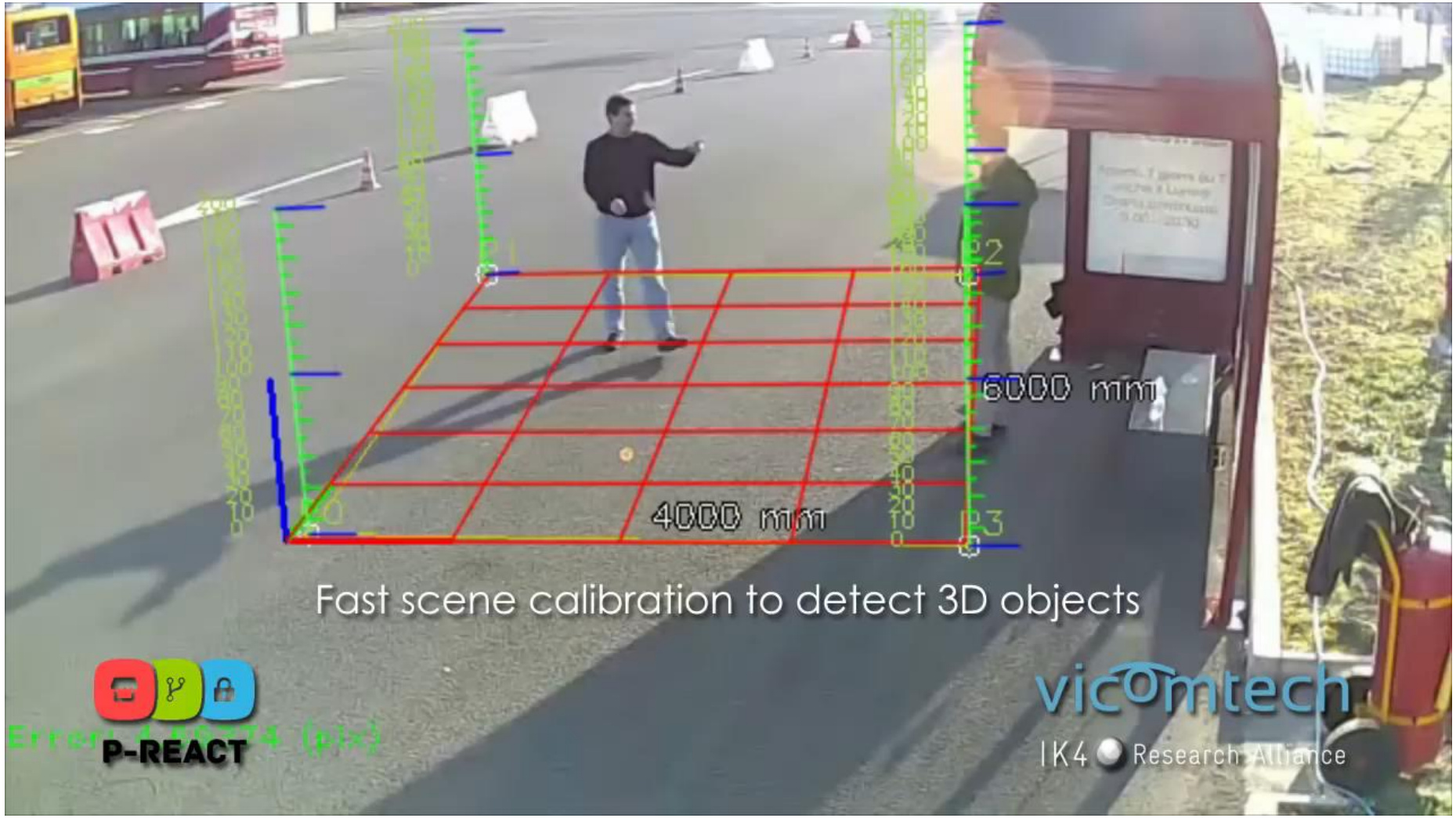
5. <u>...and many other</u>

- Face detection
- Smile detection
- Eye-open detection

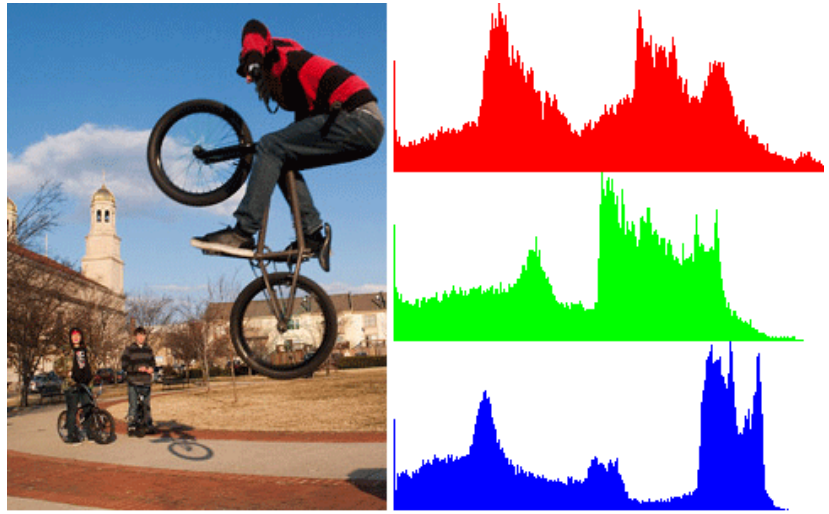


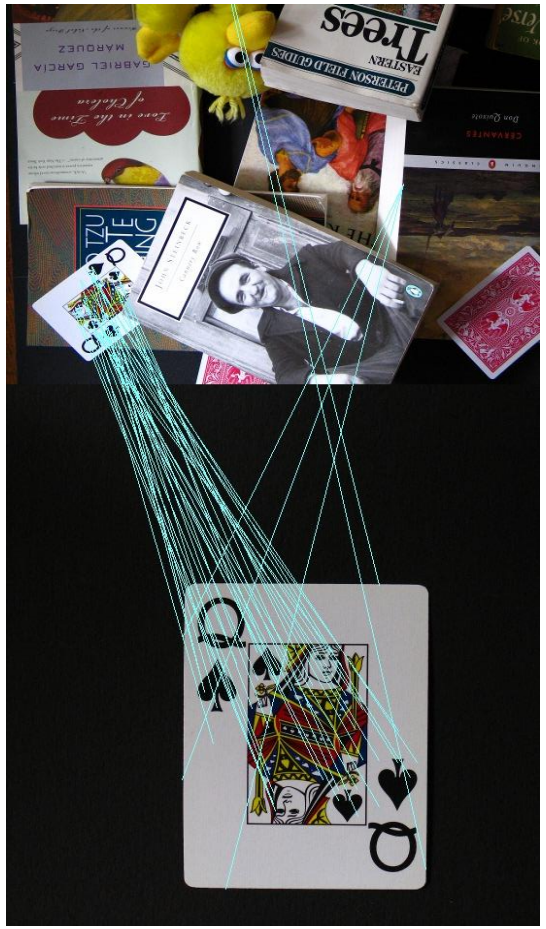
https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46

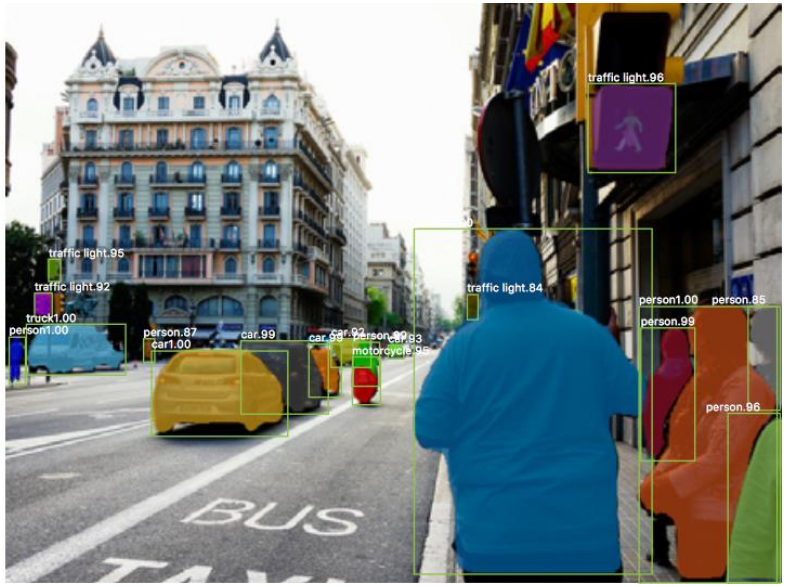
Computer vision tasks



What's next







Digital images
Basics

Classic approach
Feature engineering

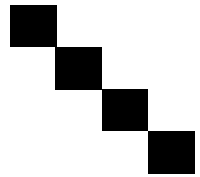
Deep learning
Convolutional neural networks

Outline

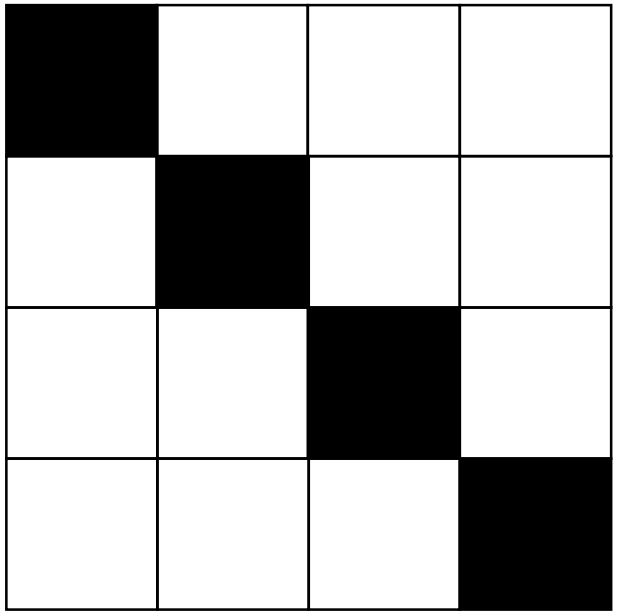
1. Computer vision: motivation
2. **Image representation**
3. Image processing
4. Computer vision: classic approaches
5. Application to strand ropes fault detection

Image representation

An image, inside a PC, is just a **matrix of numbers**



- 1: white
- 0: black
- Dimensions:4x4x1 matrix
- Color depth: 1 bit
- Color channels: 1
- # of pixels: 16

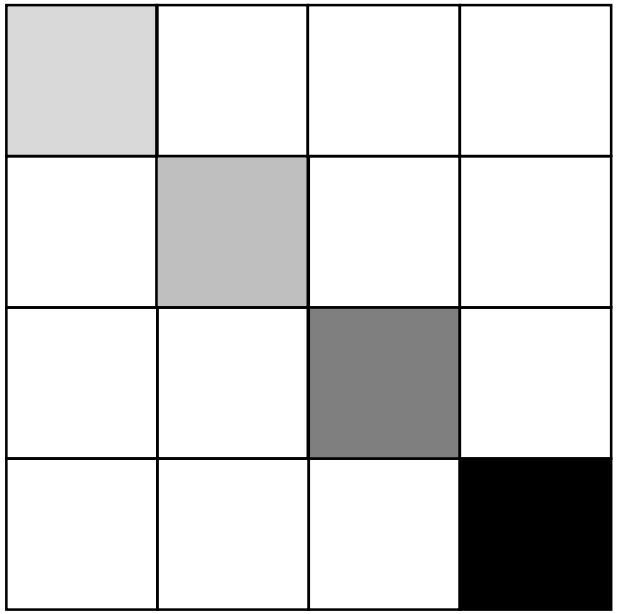


0	1	1	1
1	0	1	1
1	1	0	1
1	1	1	0

Image representation

An image, inside a PC, is just a **matrix of numbers**

- **255**: white
- 0: black
- Dimensions:4x4x1 matrix
- Color depth: **8 bit**
- Color channels: 1
- # of pixels: 16

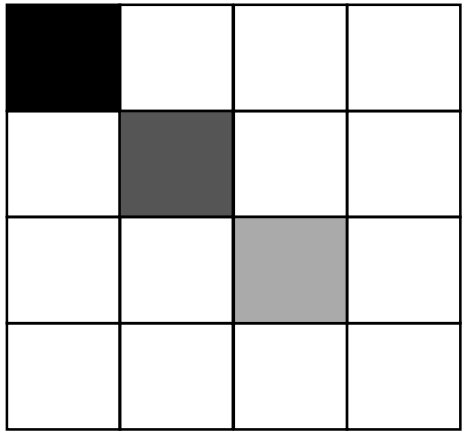


217	255	255	255
255	191	255	255
255	255	127	255
255	255	255	0

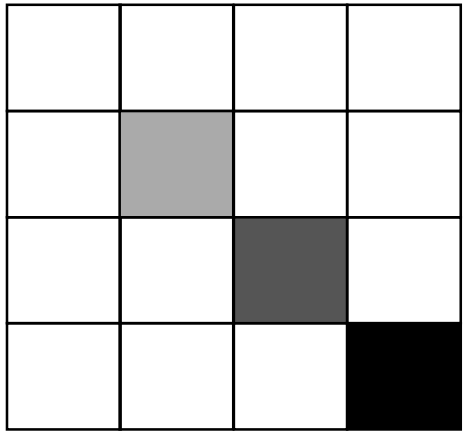
Image representation

An image, inside a PC, is just a **matrix of numbers**

- 255: white
- 0: black
- Dimensions:**4x4x2** matrix
- Color depth: 8 bit
- Color channels: 2
- # of pixels: **32**



255	255	255	255
255	170	255	255
255	255	85	255
255	255	255	0

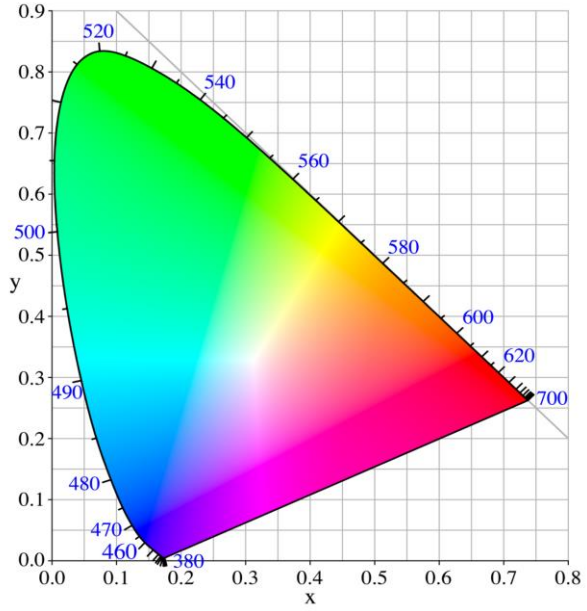


0	255	255	255
255	85	255	255
255	255	170	255
255	255	255	255

Color spaces

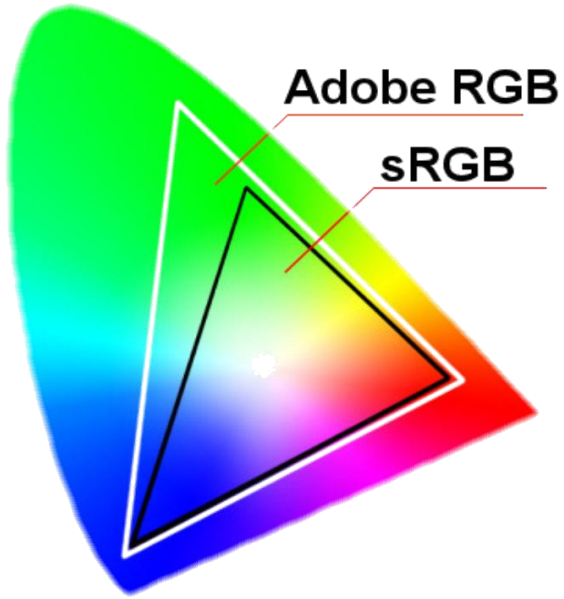
CIE XYZ

It is the most accurate from a scientific point of view. It tries to represent all the colors that an human eye can see.



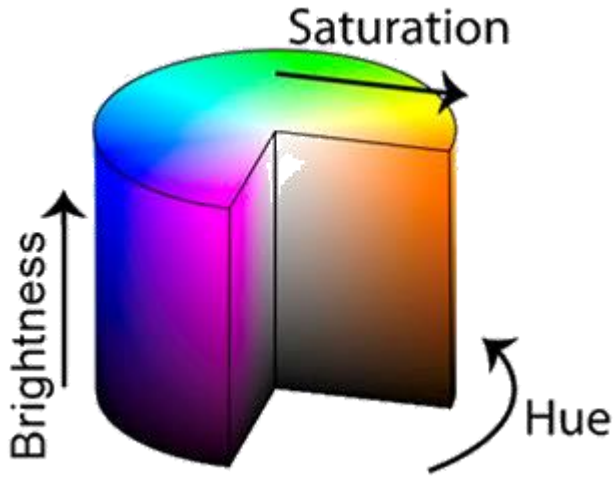
RGB

The RGB color model is an additive color model in which red, green and blue light are added.



HSB (Hue/Sat/Bright)

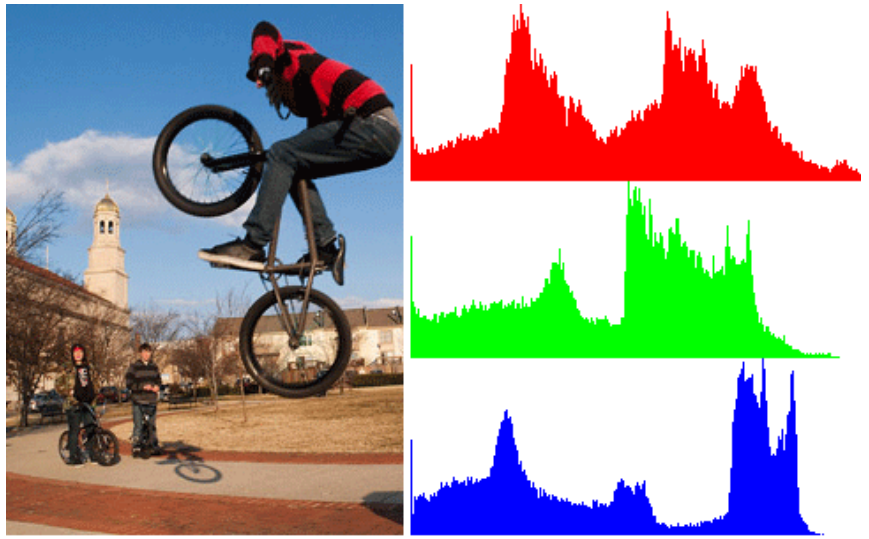
Designed in 1970 by computer graphics researchers to more closely align with the way human vision perceives color-making attributes



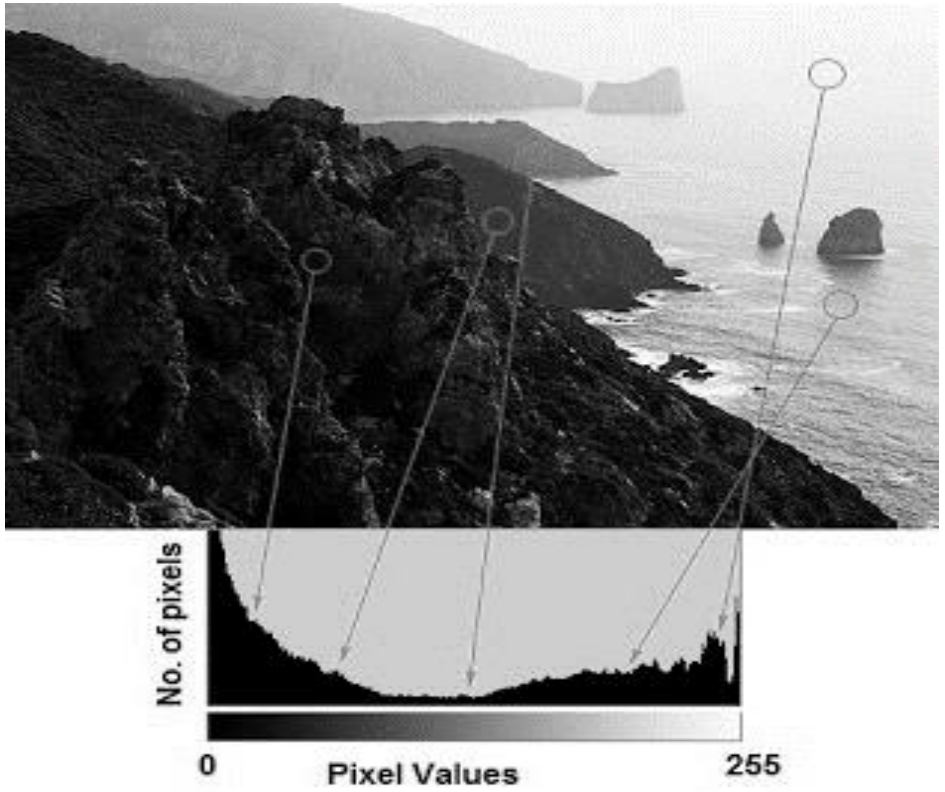
Histogram

A **color histogram** is a representation of the distribution of the colors in an image

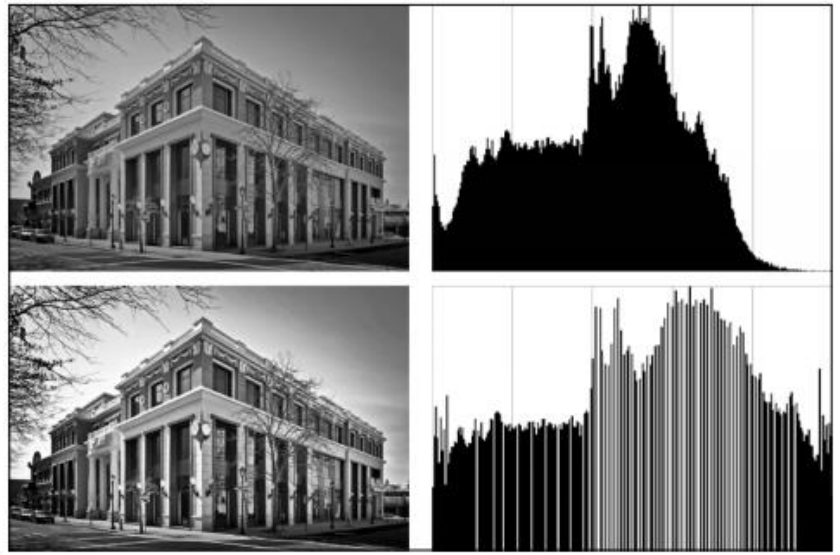
Color histogram



Gray-scale histogram



Histogram equalization



Outline

1. Computer vision: motivation
2. Image representation
3. **Image processing**
4. Computer vision: classic approaches
5. Application to strand ropes fault detection

Filters

Most of the operations on images are made using **filters**

- A **filtering** operation is a **mathematical function** which is applied to the image pixels

Input image

Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

- The filter is represented using a matrix, called **kernel**
- Depending on the size of the kernel ($3 \times 3, 5 \times 5, 7 \times 7$, ...), the filtering functions will involve a single pixel and its **neighbors**
- The filters are applied to an image by **convolving** the image and the kernel

Filters

Convolution is the process of adding each element of the image to its local neighbors, weighted by the kernel.

$$y[0,0] = \sum_j \sum_i x[i,j] \cdot k[0-i, 0-j]$$

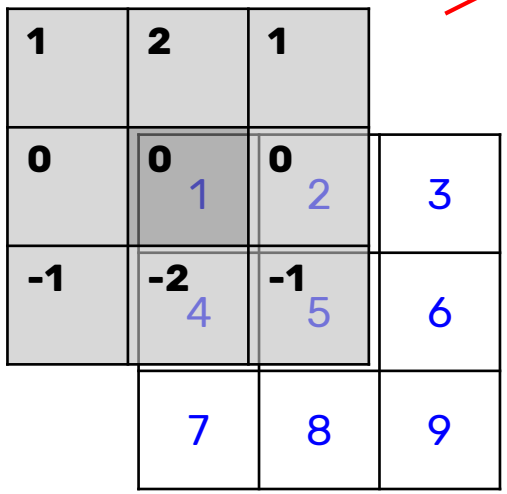
Input image $x[\cdot,\cdot]$

1	2	3
4	5	6
7	8	9

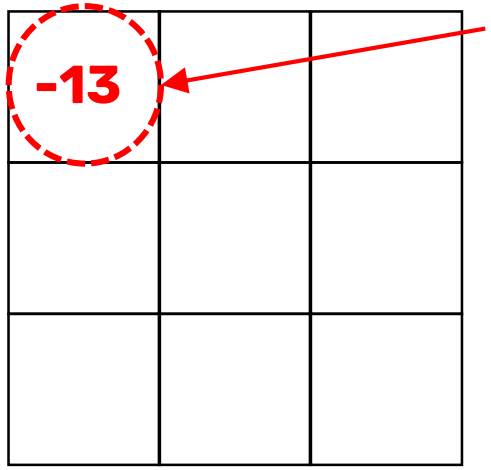
Kernel $k[\cdot,\cdot]$

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

STEP 1



Output $y[\cdot,\cdot]$



$$0 \cdot \mathbf{1} + 0 \cdot \mathbf{2} + 0 \cdot \mathbf{1} + 0 \cdot \mathbf{0} + 1 \cdot \mathbf{0} + 2 \cdot \mathbf{0} + 0 \cdot (-\mathbf{1}) + 4 \cdot (-\mathbf{2}) + 5 \cdot (-\mathbf{1})$$

Filters

Convolution is the process of adding each element of the image to its local neighbors, weighted by the kernel.

$$y[1,1] = \sum_j \sum_i x[i,j] \cdot k[1-i, 1-j]$$

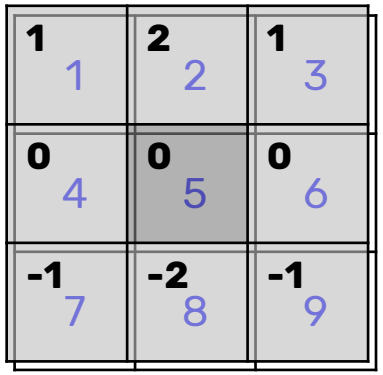
Input image $x[\cdot,\cdot]$

1	2	3
4	5	6
7	8	9

Kernel $k[\cdot,\cdot]$

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

STEP 5



Output $y[\cdot,\cdot]$

-13	-20	-17
-18	**-24**	

$$1 \cdot \mathbf{1} + 2 \cdot \mathbf{2} + 3 \cdot \mathbf{1} + 4 \cdot \mathbf{0} + 5 \cdot \mathbf{0} + 6 \cdot \mathbf{0} + 7 \cdot (-\mathbf{1}) + 8 \cdot (-\mathbf{2}) + 9 \cdot (-\mathbf{1})$$

Filters

Convolution is the process of adding each element of the image to its local neighbors, weighted by the kernel.

$$y[2,2] = \sum_j \sum_i x[i,j] \cdot k[2-i, 2-j]$$

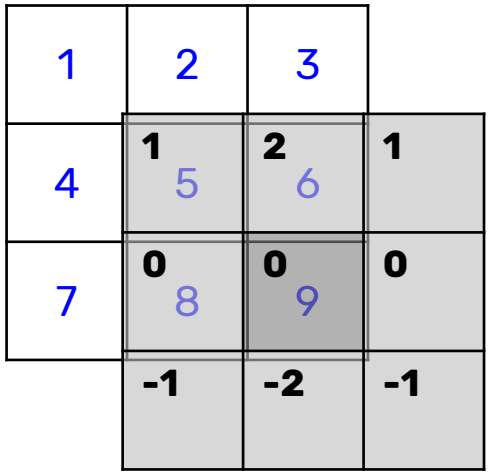
Input image $x[\cdot,\cdot]$

1	2	3
4	5	6
7	8	9

Kernel $k[\cdot,\cdot]$

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

STEP 9

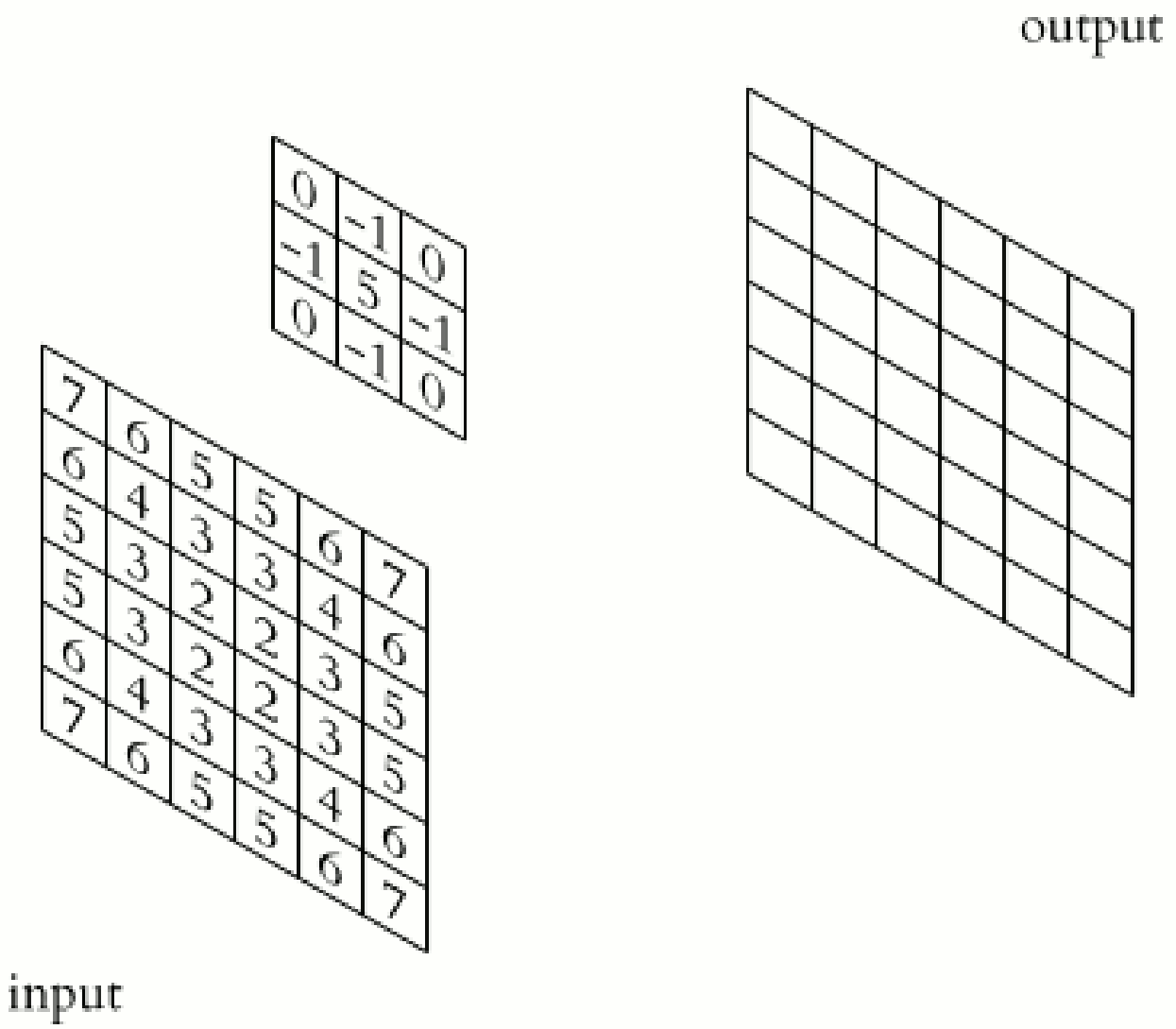


Output $y[\cdot,\cdot]$

-13	-20	-17
-18	-24	-18
13	20	**17**

$$5 \cdot \mathbf{1} + 6 \cdot \mathbf{2} + 0 \cdot \mathbf{1} + 8 \cdot \mathbf{0} + 9 \cdot \mathbf{0} + 0 \cdot \mathbf{0} + 0 \cdot (\mathbf{-1}) + 0 \cdot (\mathbf{-2}) + 0 \cdot (\mathbf{-1})$$

Convolutior



https://it.m.wikipedia.org/wiki/File:3D_Convolution_Animation.gif

Kernel examples

Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Sharpen

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Edge-Detection

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Blur/moving average

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

Kernel examples

Edges

-1	-1	-1
-1	8	-1
-1	-1	-1

45° Lines

-1	-1	2
-1	2	-1
2	-1	-1

Horizontal lines

-1	-1	-1
2	2	2
-1	-1	-1

Vertical lines

-1	2	-1
-1	2	-1
-1	2	-1

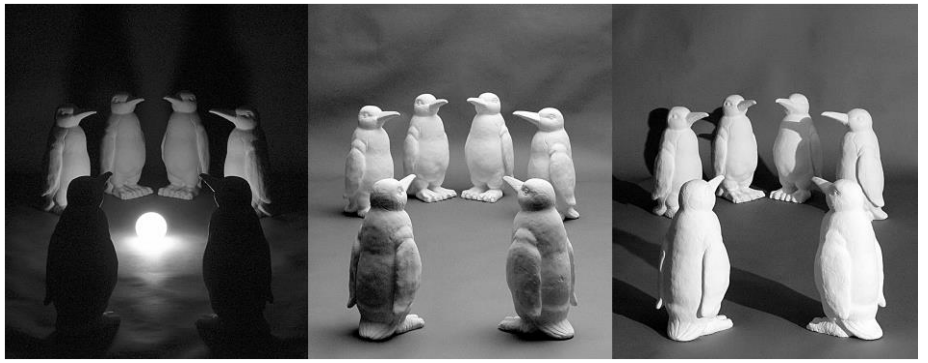
Outline

1. Computer vision: motivation
2. Image representation
3. Image processing
4. **Computer vision: classic approaches**
5. Application to strand ropes fault detection

Classic approaches

Computer vision tasks have to be **robust** against several perturbations:

Different viewpoint



Different illumination

Deformations

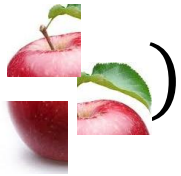
Different shapes of the same object

How to identify an object?

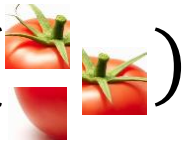
The idea is to extract **features**, some important characteristics of the image

OBJECT

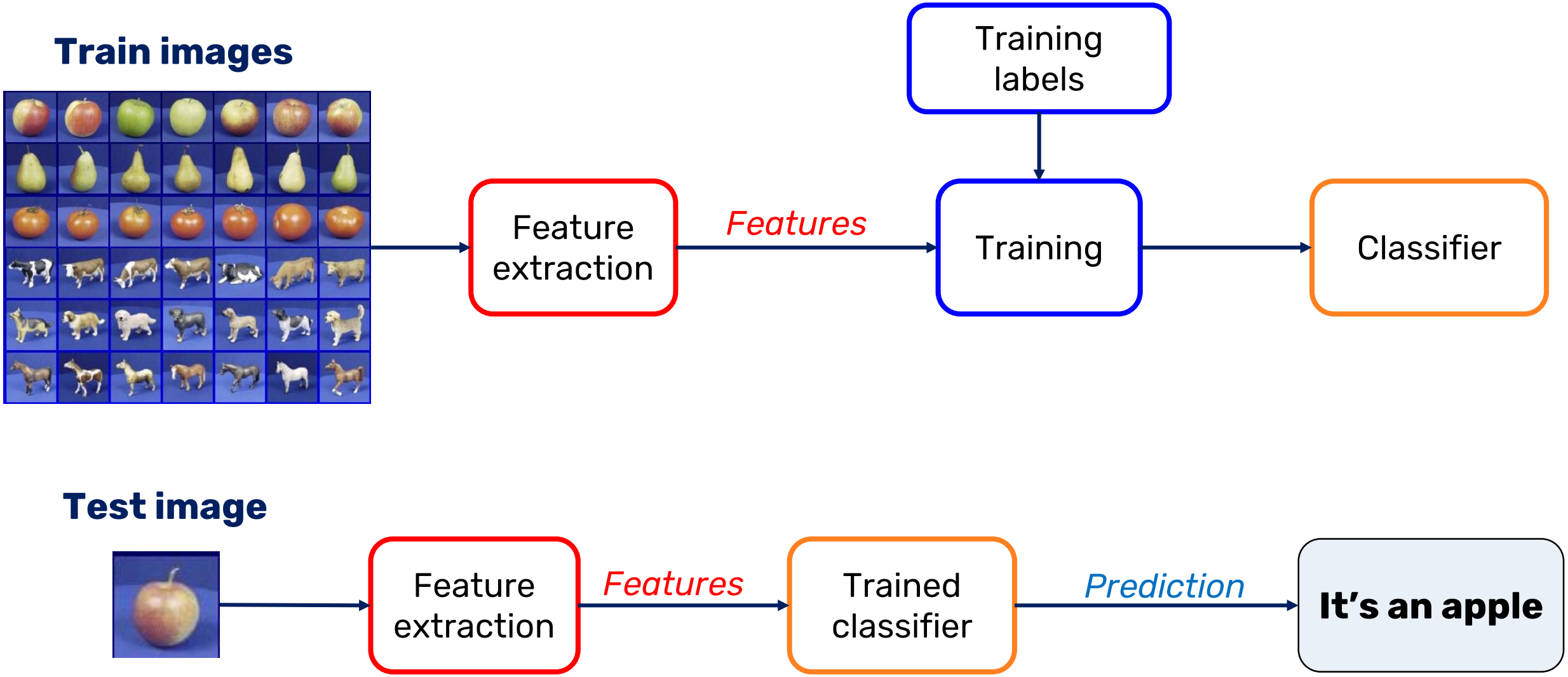
“BAG OF FEATURES”

$f($  $) =$ **apple**

$f($  $) =$ **cow**

$f($  $) =$ **tomato**

Classification pipeline



Features for images

In Computer Vision, a feature is a piece of information that is **relevant** in the solution of a **certain problem** (e.g. in face detection, the presence of two eyes is a good feature)

Three hierarchical categories of features:

Low-level features

- Colors
- Edges
- Blob
- Corners

Mid-level features

- Scale-invariant features
- SIFT
- SURF
- SVD

High-level features

- Histograms of gradients (HOG)
- Region descriptors
- Haar features

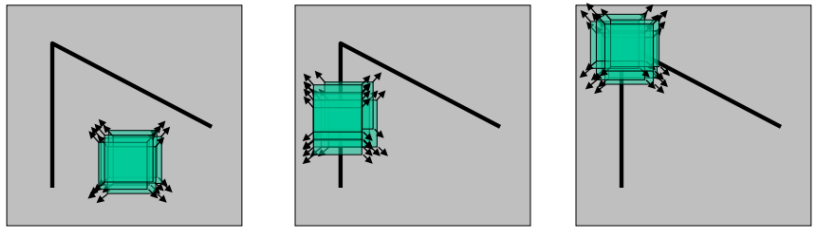
Low-level features

Edges and corners: since the process of image classification involve the exploitation of edges and corners, there are a lot of methods for finding them

Canny edge detector

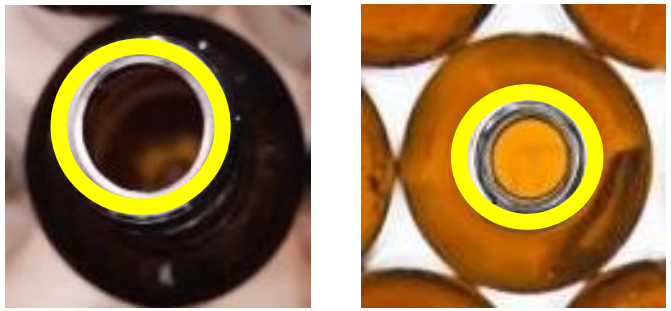
It uses a 5 steps algorithm, involving some filters (gradient) to compute all the edges in an image.

Harris corner detection



A mobile window slides over the image and computes the Hessian. The corners are evaluated by taking the eigenvalue of each matrix

Hough transform



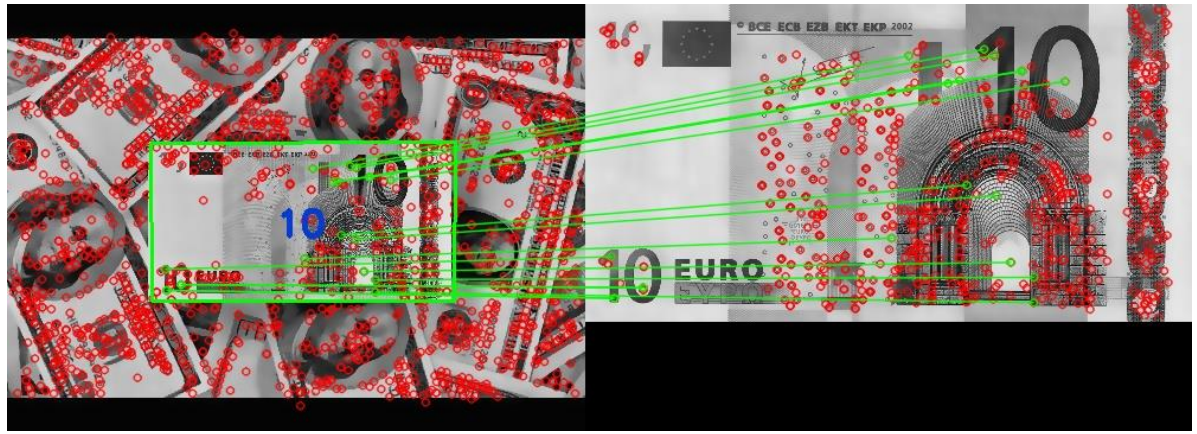
Initially devised for discovering lines in images, it has been extended to detect circles and ellipses

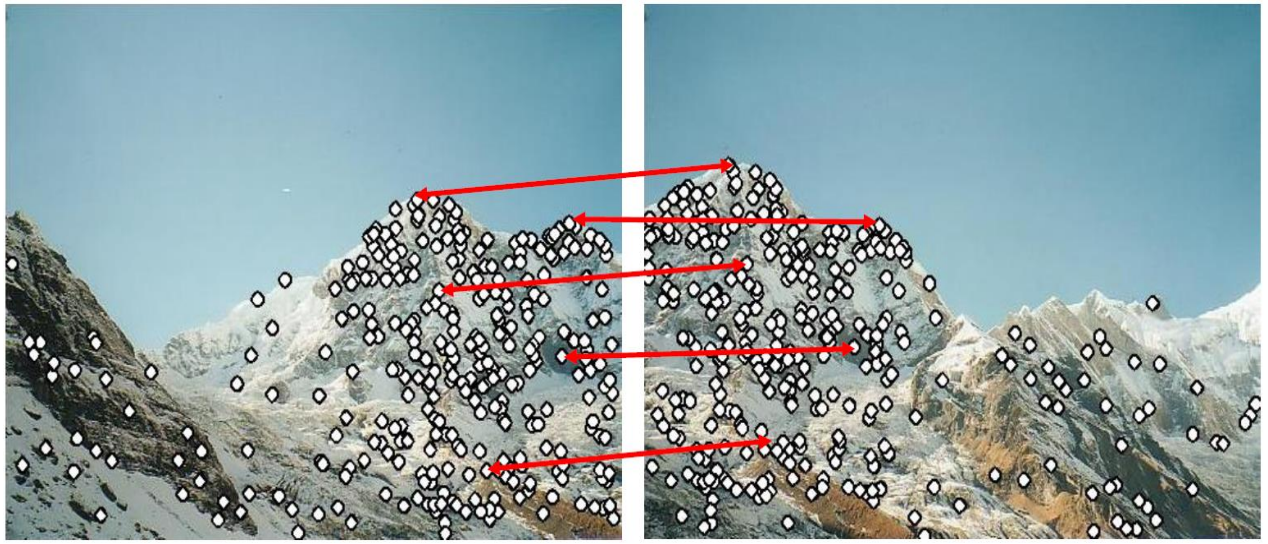
Mid-level features

SIFT – Scale Invariant Feature Transform [1999]

It is an algorithm which is able to detect and describe features in an image at different

scales and rotations

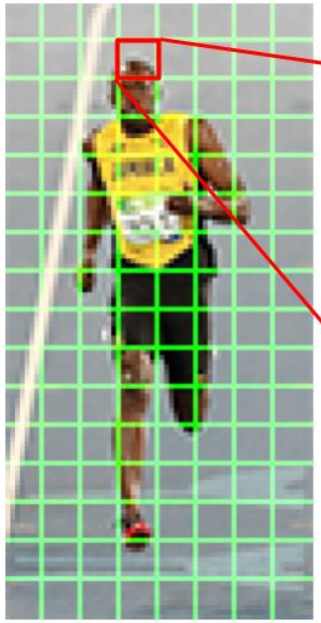


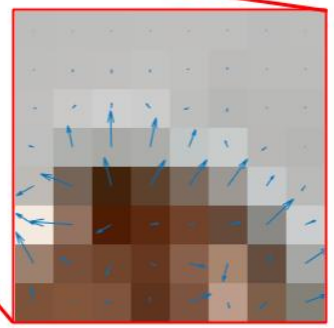


High-level features

HOG – Histogram of Oriented Gradients [2005]

- Counts occurrences of gradient orientations in localized portions of an image
- The distribution (histograms) of directions of gradients (oriented gradients) is used as feature *- magnitude of gradients is large around edges and corners (regions of abrupt intensity changes)*





-1	0	1

x derivative

-1
0
1

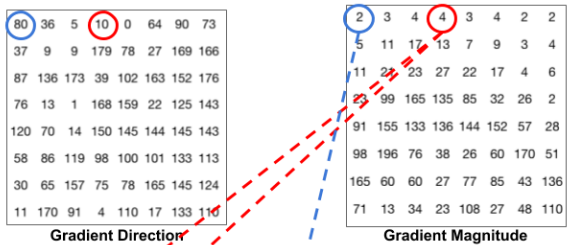
y derivative

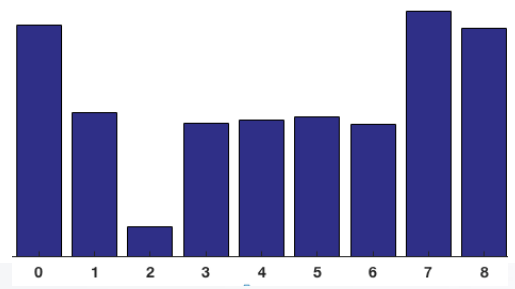
2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction





9 cells corresponding to angles

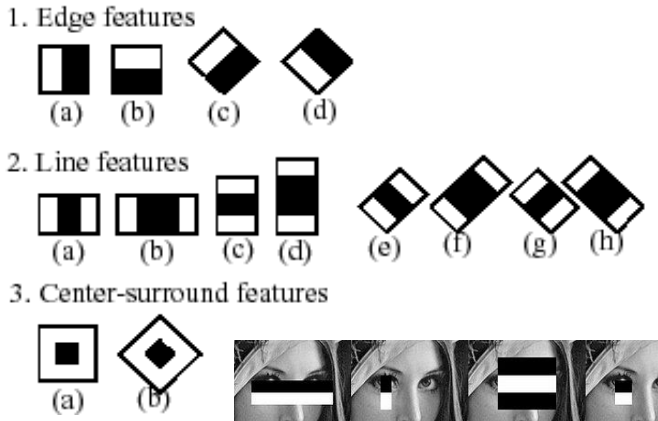
$$g = \sqrt{g_x^2 + g_y^2}$$

$$\theta = \arctan \frac{g_y}{g_x}$$

https://www.learnopencv.com/histogram-of-oriented-gradients/

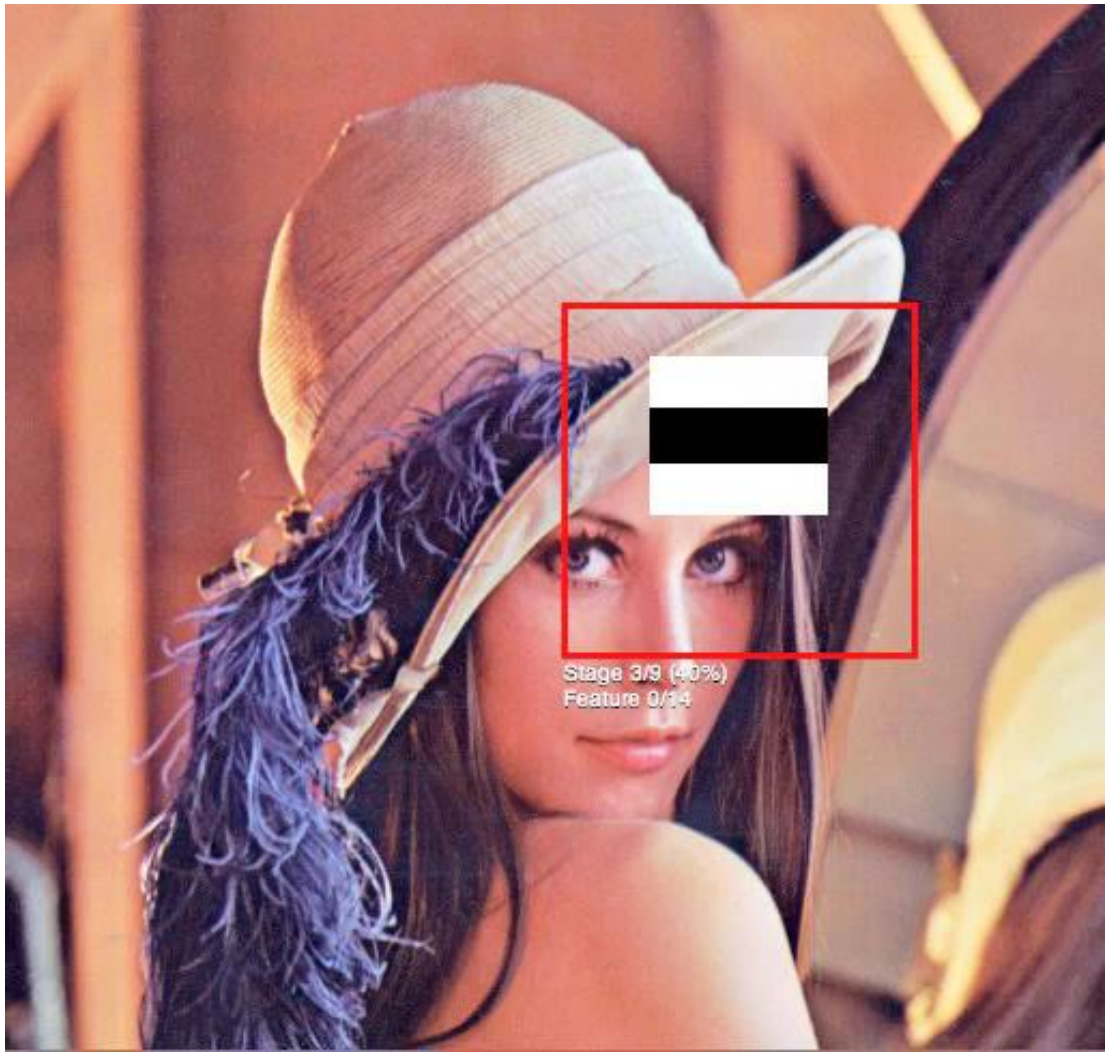
Viola-Jones object detector algorithm

- First **object detection** framework to provide high object detection rates in **real-time**
- Employs **Haar features** to characterize the input image
- Each feature is computed by **subtracting** the sum of pixels under **white rectangle** from the sum of pixels under **black rectangle**
- **Moving window**. An object is recognized if the window **contains all the features** that characterize the object
- An **adaboost classifier** is trained by a set of labeled images of the object of interested and other objects



(Viola, Jones: Robust Real-time Object Detection, IJCV 2001)

Viola-Jones object detector algorithm



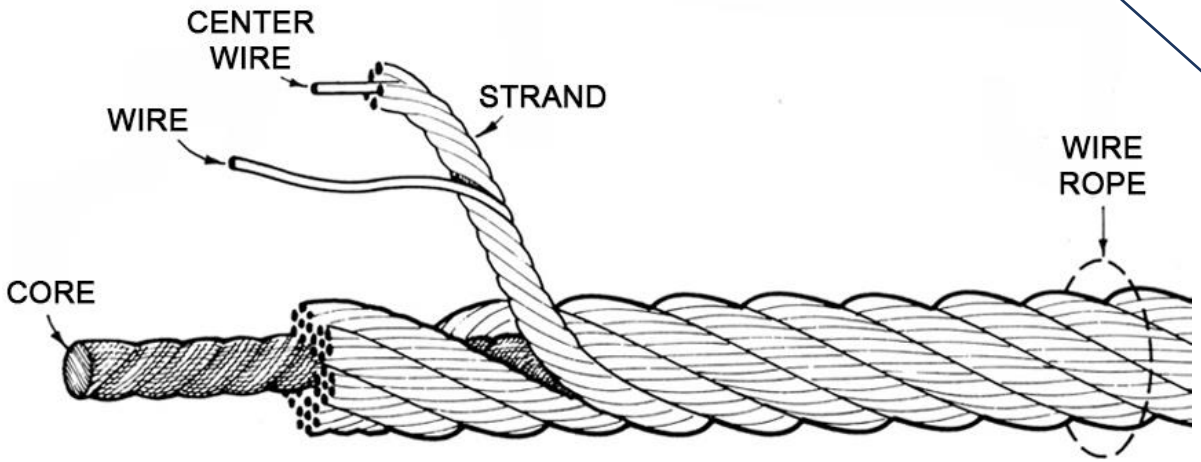
https://vimeo.com/12774628

Outline

1. Computer vision: motivation
2. Image representation
3. Image processing
4. Computer vision: classic approaches
5. **Application to strand ropes fault detection**

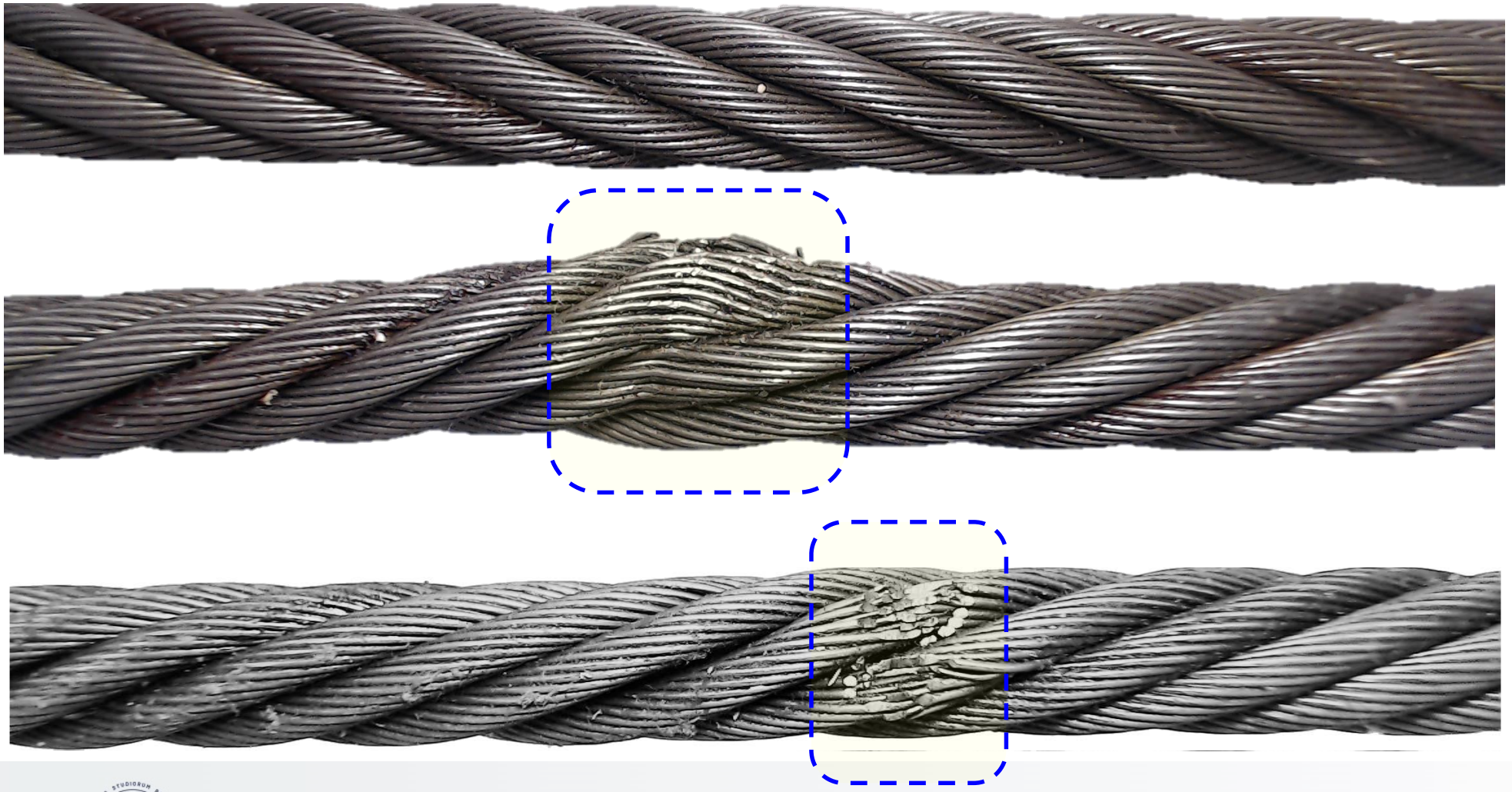
Strand ropes fault detection

In this project, we applied classic computer vision techniques to detect faults in **strand ropes**, used in bridge cranes



VINATI

Strand ropes fault detection



Healthy rope

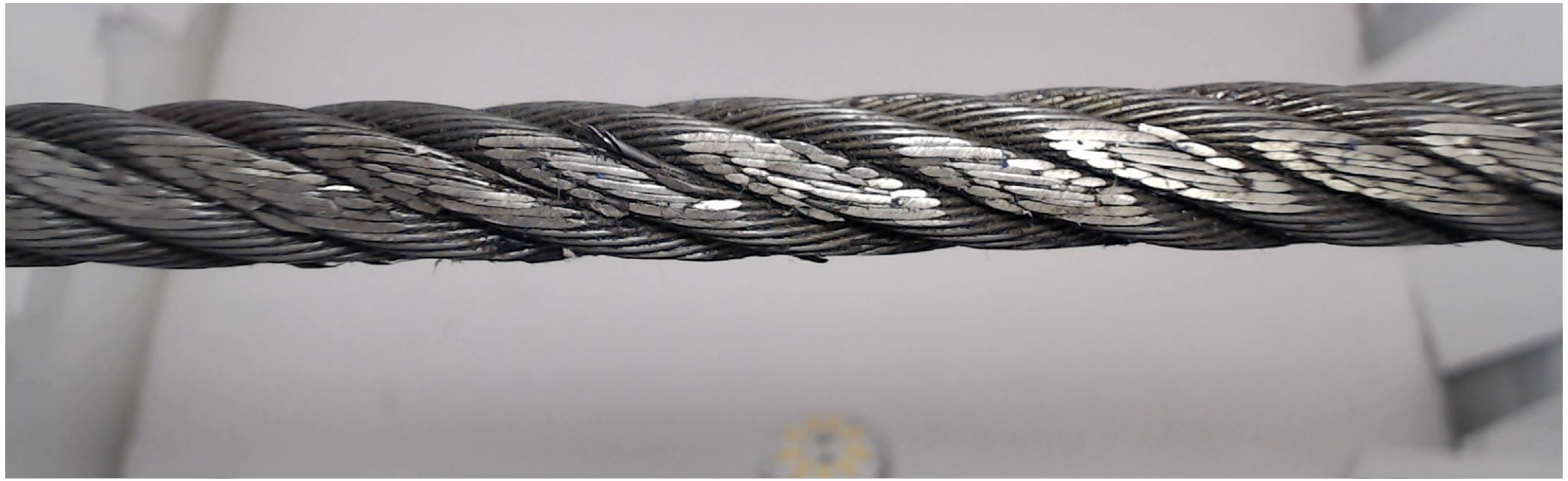
Faulty rope 1

Faulty rope 2

Strand ropes fault detection

Steps of the algorithm

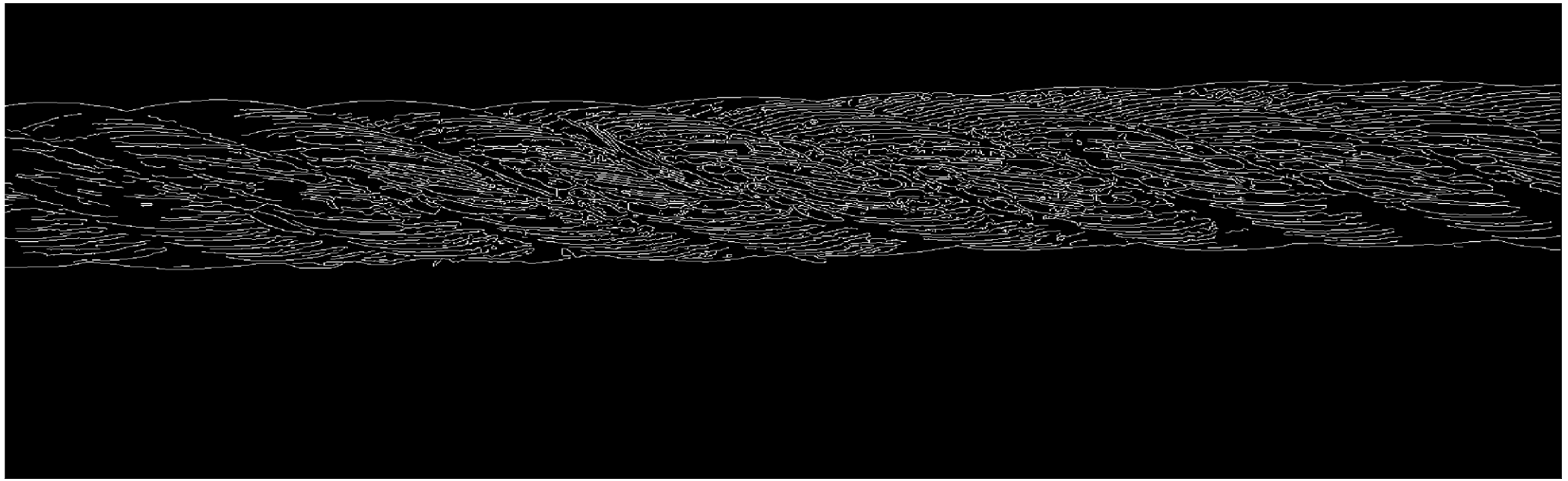
1. Convert to gray scale



Strand ropes fault detection

Steps of the algorithm

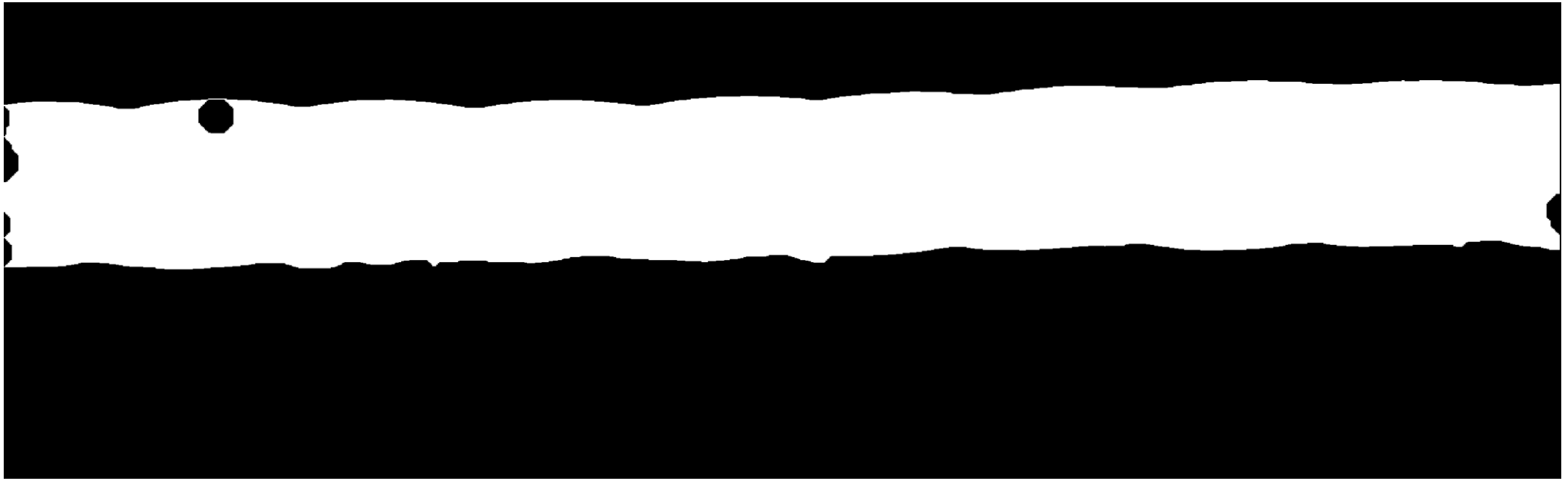
2. Apply Canny edge detection algorithm



Strand ropes fault detection

Steps of the algorithm

3. Fill the inside region of the rope to obtain a continuous border



Strand ropes fault detection

Steps of the algorithm

4. Compute the rope border (in blue)

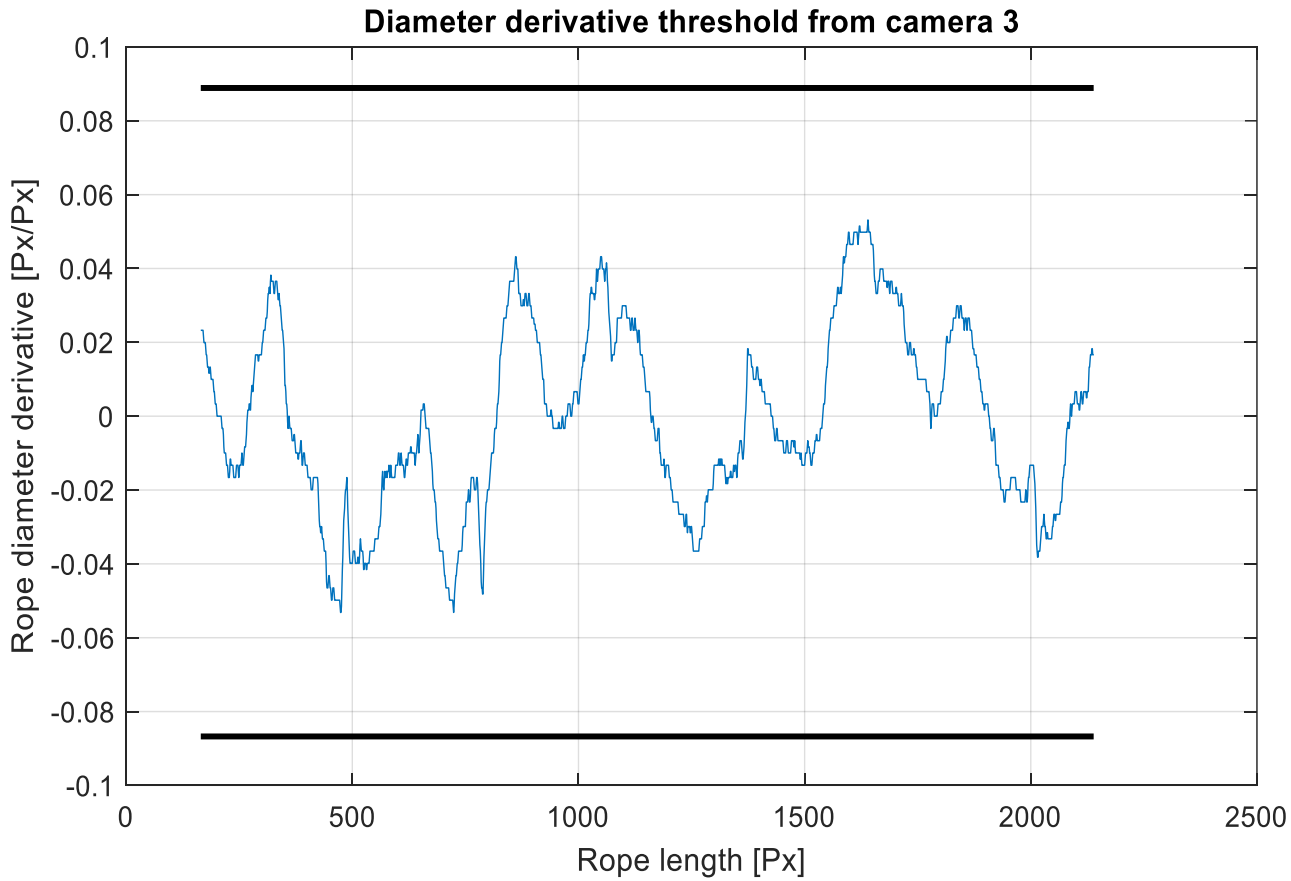
Strand ropes fault detection

Steps of the algorithm

5. Compute the **rope diameter** for the entire length of the rope

6. Compute the **derivative** of the rope diameter (robust to different rope inclinations and camera zoom)

7. Compare with pre-defined **thresholds**



Strand ropes fault detection

Steps of the algorithm

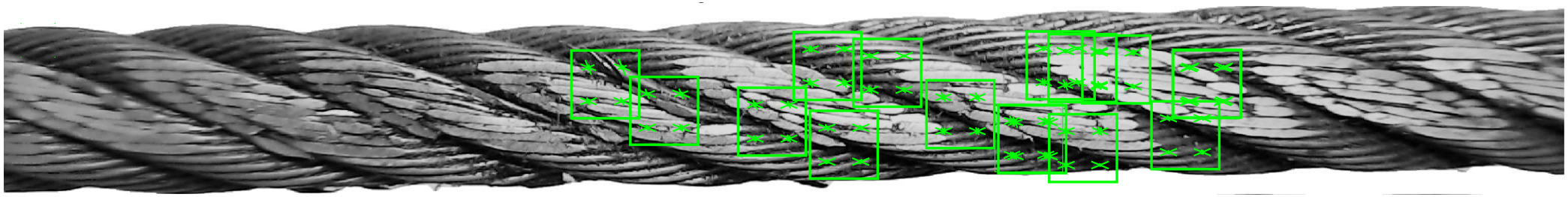
8. Image equalization

Strand ropes fault detection

Steps of the algorithm

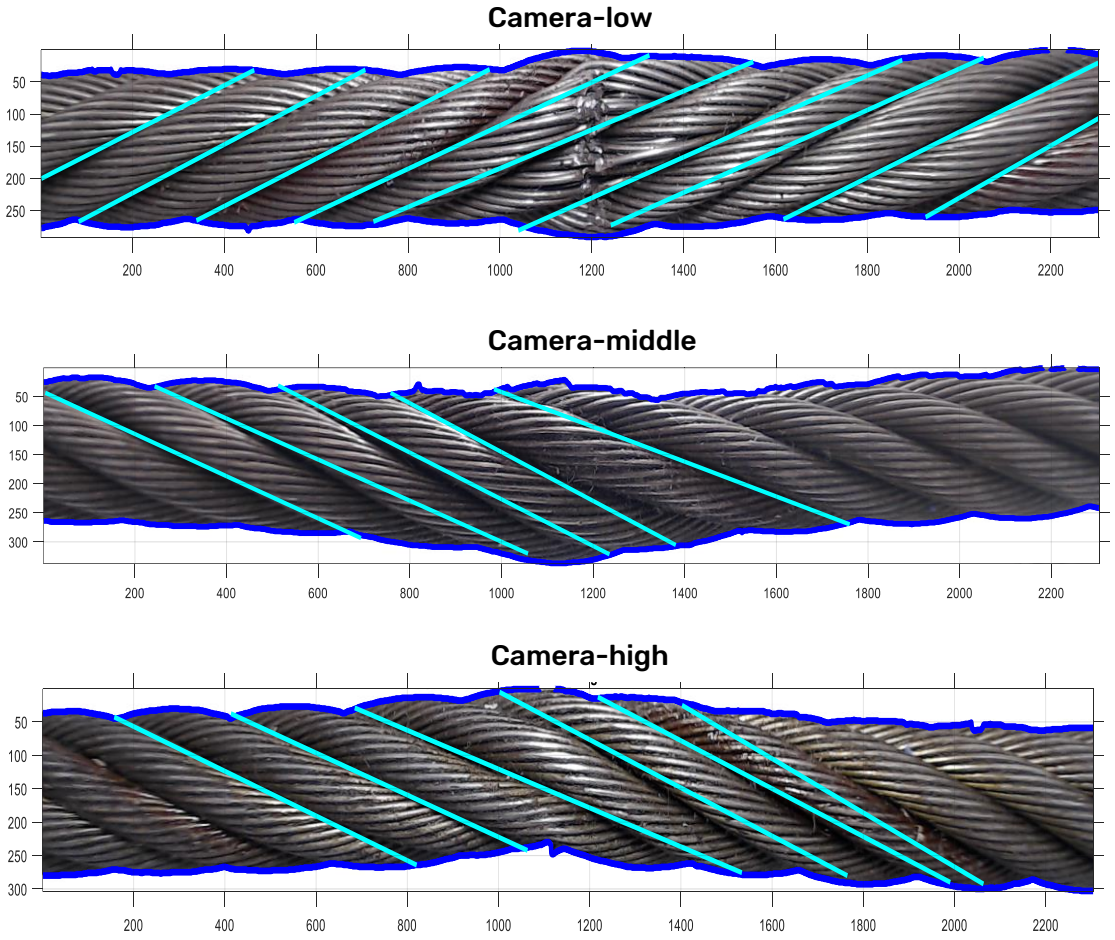
8. Image equalization

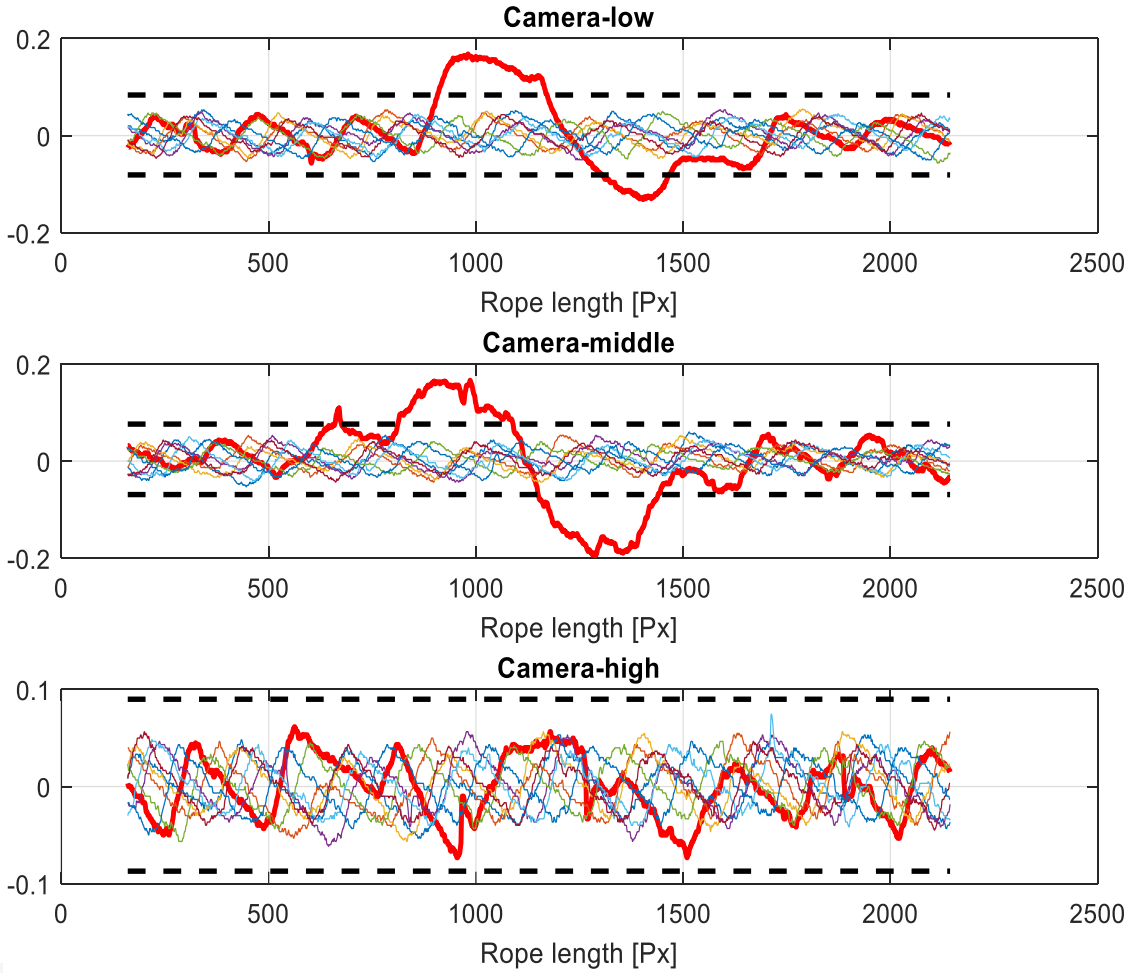
9. Corner detection + HOG features computation + Principal Component Analysis



Strand ropes fault detection

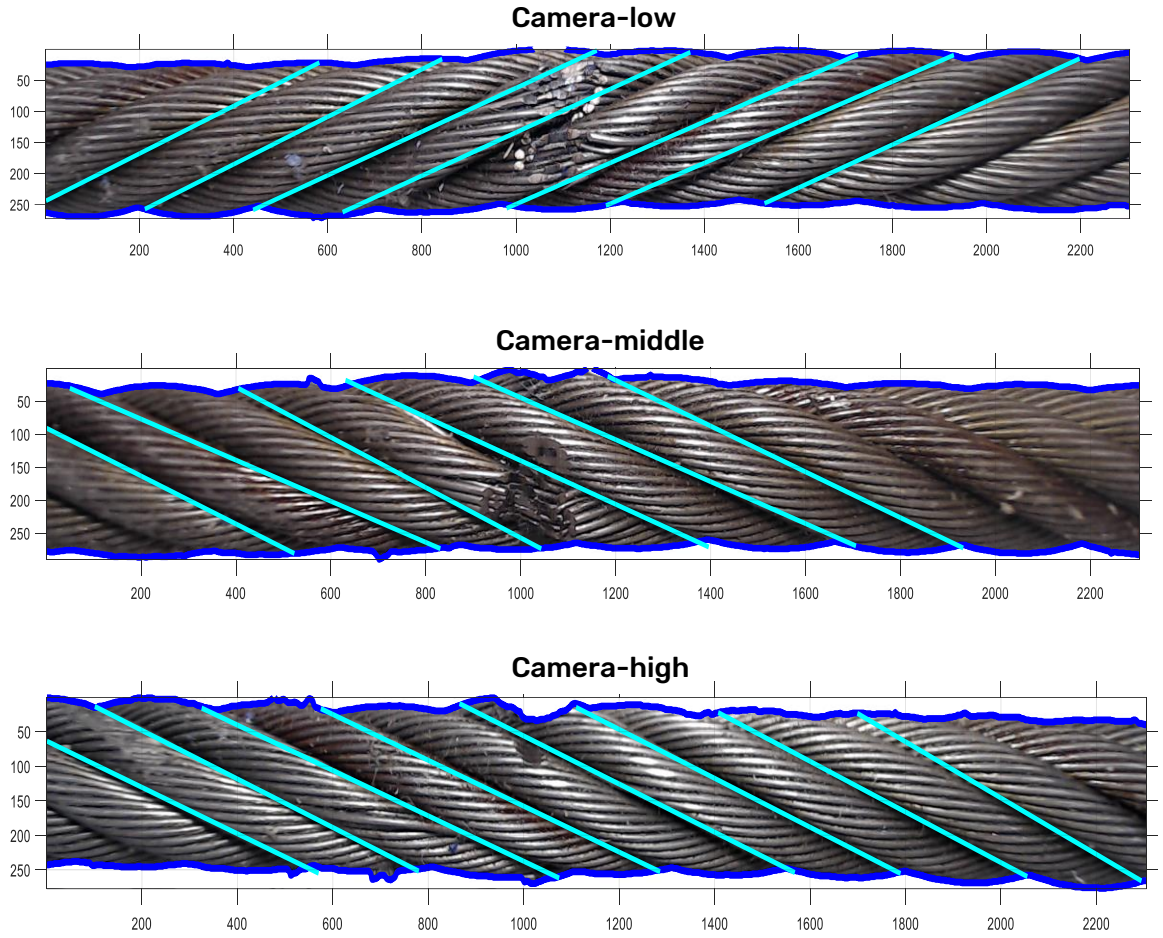
Faulty rope

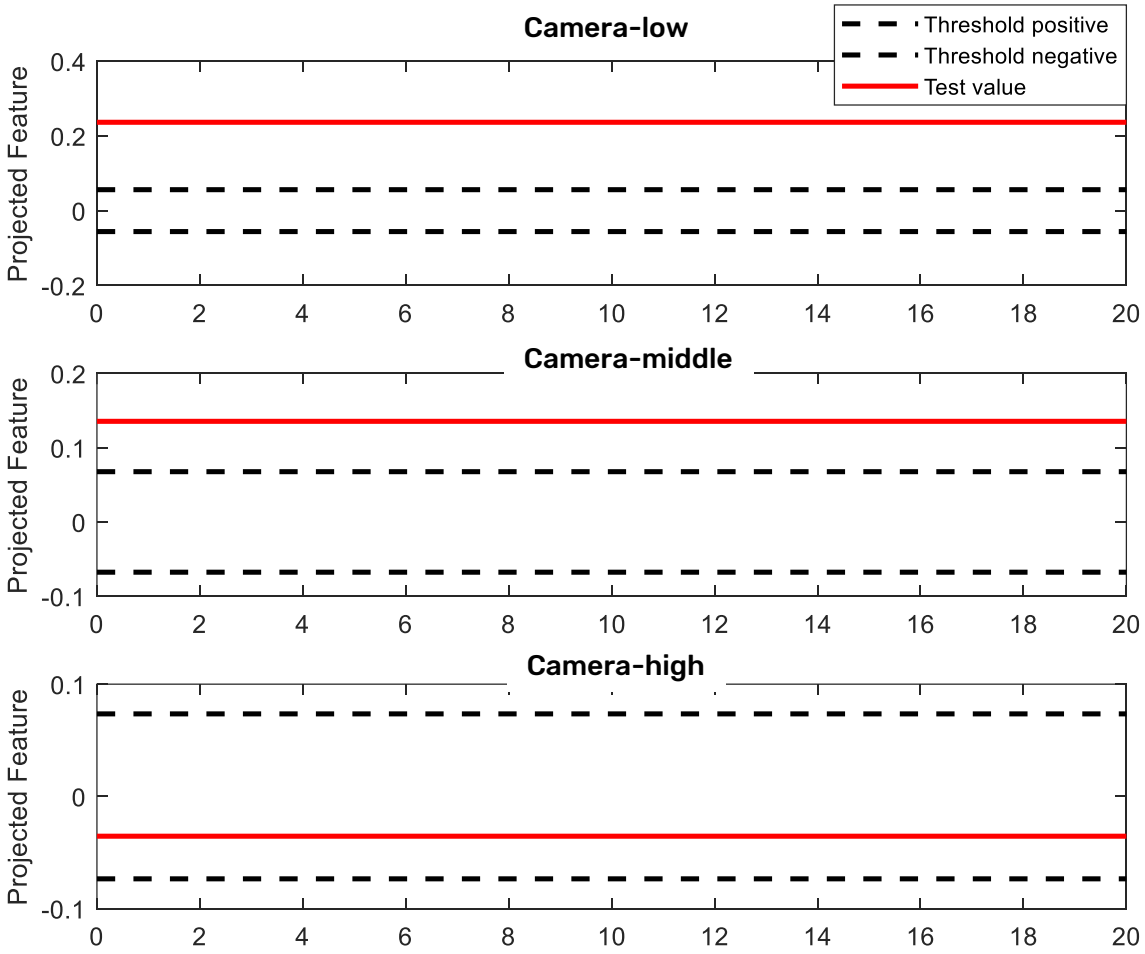




Strand ropes fault detection

Faulty rope





Strand ropes fault detection

Healthy rope