# MLRF Lecture 02

J. Chazalon, LRDE/EPITA, 2021

# Agenda for lecture 2

1. Introduction
2. Global image descriptors
3. Clustering
4. Local feature detectors

# Introduction

Lecture 02 part 01

# Previously, in MLRF…

# Summary of last lecture

Machine learning

- Machine learning = searching for the best model in a hypothesis space
- Inductive machine learning, optimization-based
- Inductive bias, biais/variance compromise
- Supervised, reinforcement, unsupervised learning
- Regression, classification, density estimation
- Model validation: test generalisation, separate/decorrelate test & training sets

Template matching

- Sum of squared differences $(T-I)^2$, or correlation-based methods $(T \times I)$
- Normalization needed for correlation-based methods
- Tolerates translation and small noise, but not rotation, intensity shift, …

# Debriefing of practice session 1

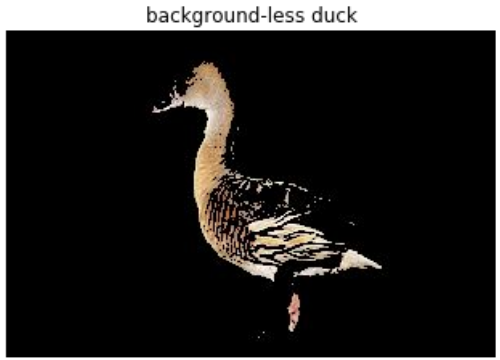
**PS1 content**

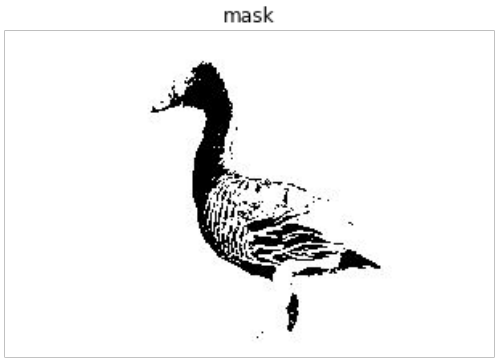
1. Jupyter tricks
2. NumPy reminders
3. Intro to image manipulations
4. Twin it! part 1: Template matching
5. (Bonus level: segmentation)

**Discussion**

- Who completed part 1? 2? 3? 4? 5?
- Any remarks, comments, questions?
- Things to keep, change, remove?

# Practice session 1: Take home messages (1/2)




*How annoying was it*
*to manually adjust color thresholds*
*to select the duck?*

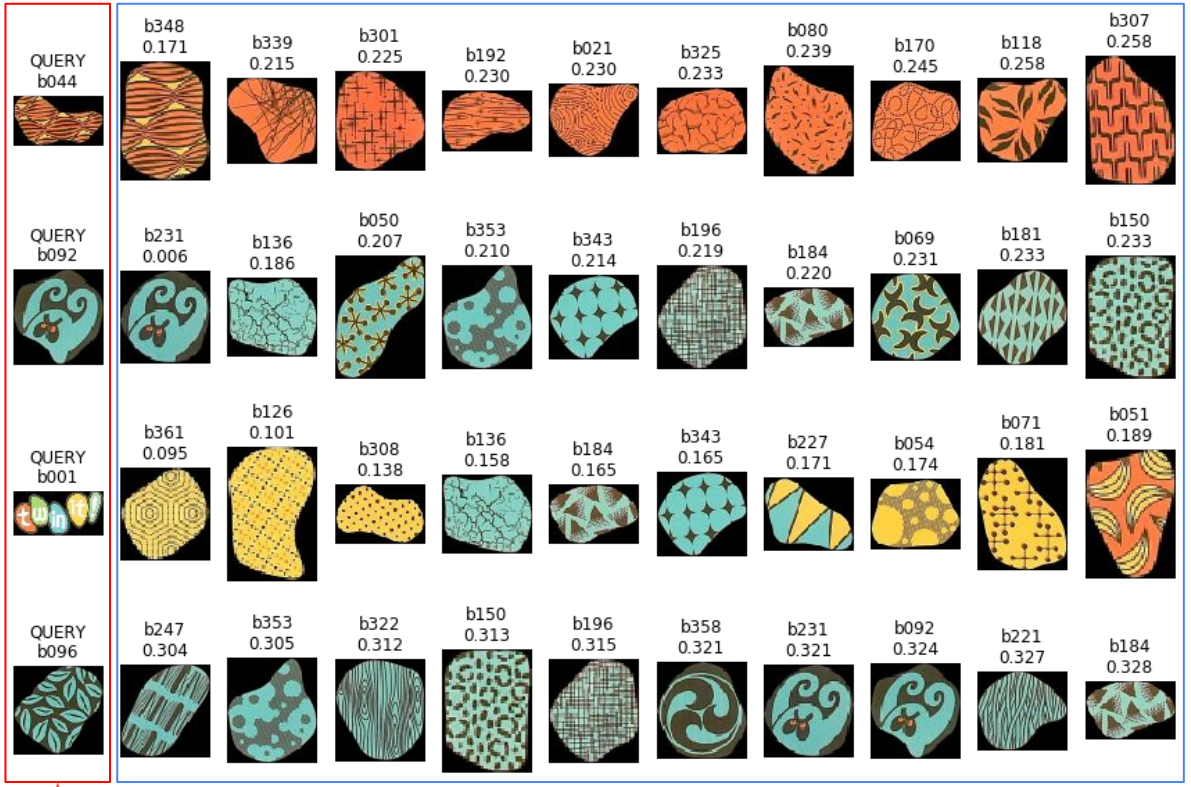
*How could have we automated it?*

# Practice session 1: Take home messages (2/2)

Results with method SQDIFF_NORMED → (lower is better)

*Strengths and weaknesses of template matching for the* Twin it! *case?*

*Effects of normalization?*




query images

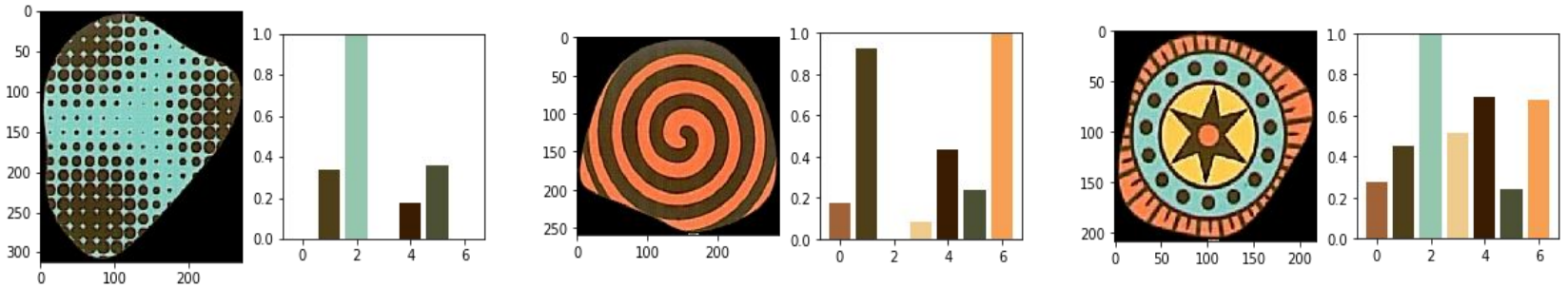
result images (closest to query according to method)

# Next practice session(s)

# Next practice session

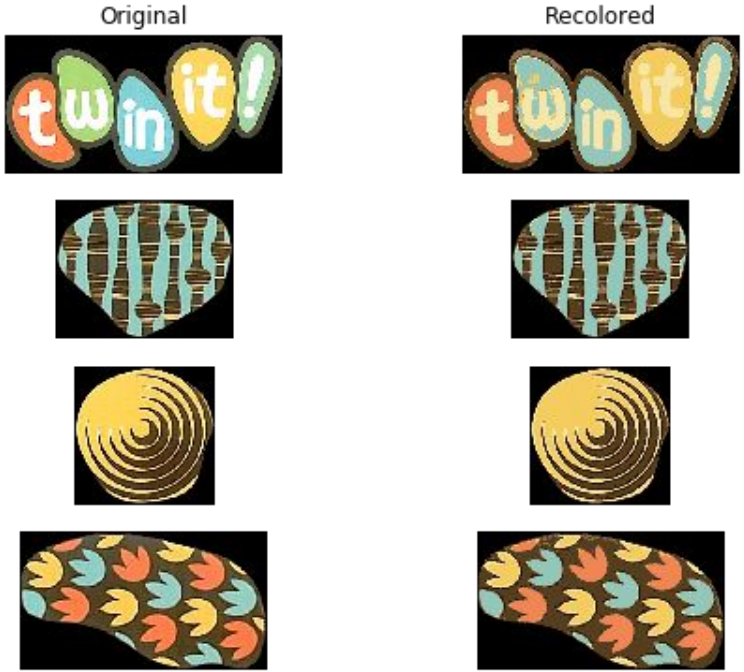
*Twin it!*, again, with a slightly more elaborated approach:

1. **Pre-select bubbles based on their colors ⇒ Color histograms**



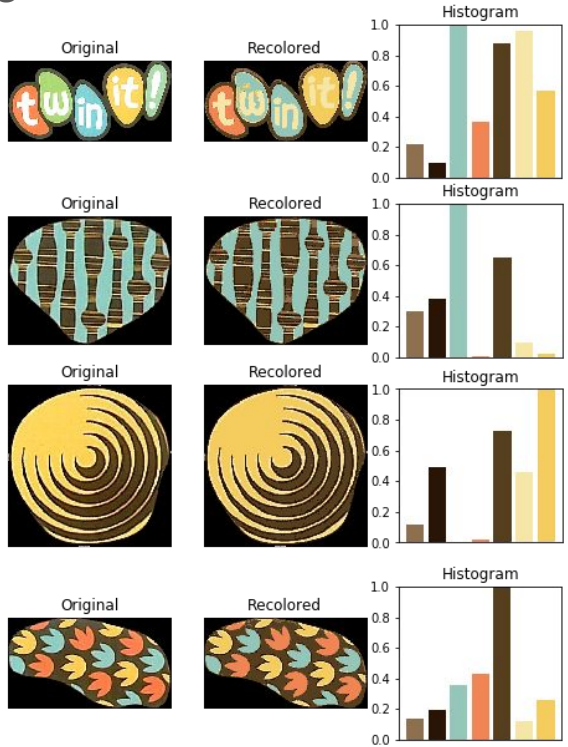
# Color histogram: In details

1.1. Color quantization: reduce the colors of the bubbles.



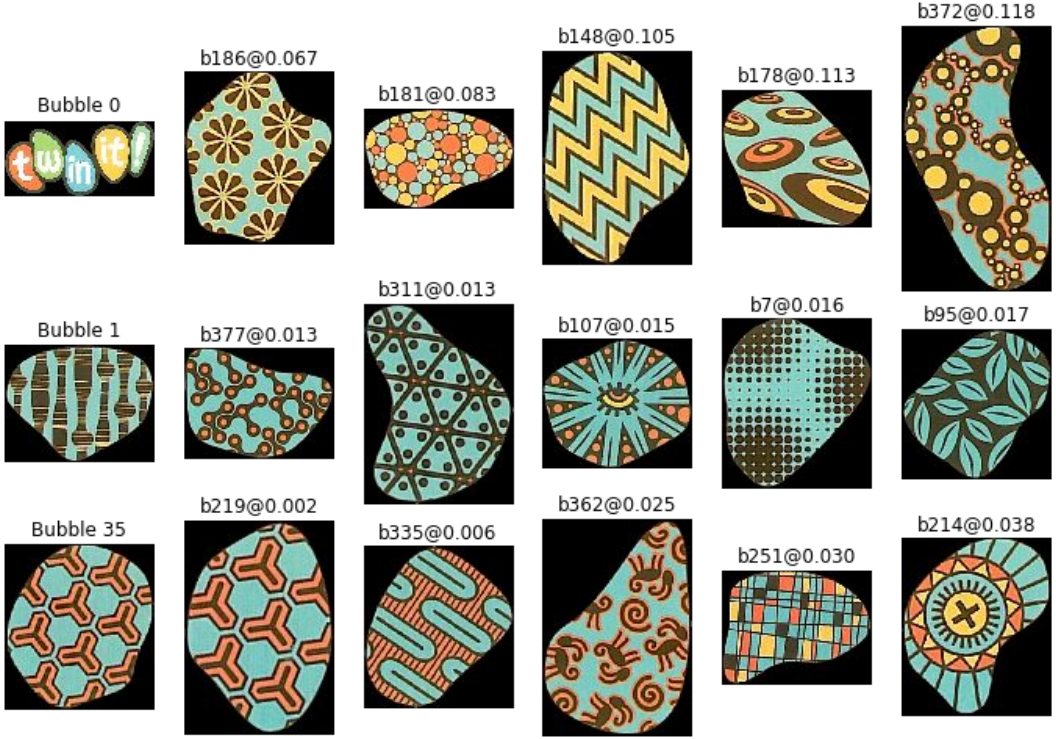
# Color histogram: In details

1.2. Compute the color histogram of each bubble.



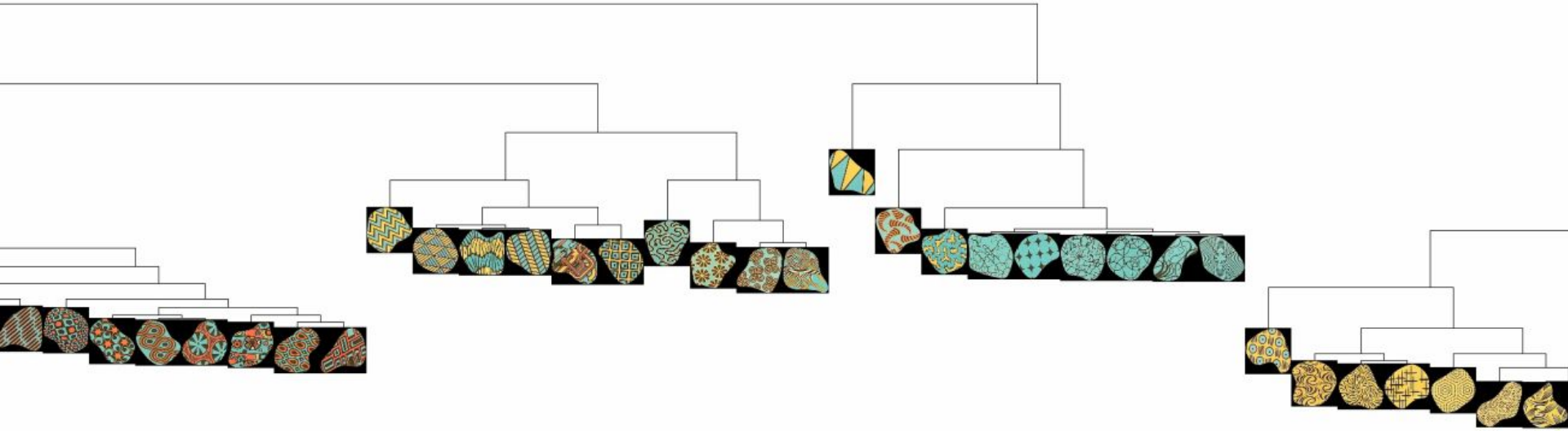
# Color histogram: In details

1.3. Compute the distance matrix between each bubble, using its color histogram.



# Color histogram: In details

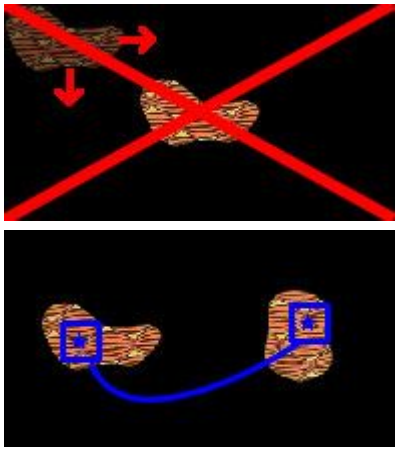
1.4. Visualize the bubbles in an interesting way using hierarchical clustering.

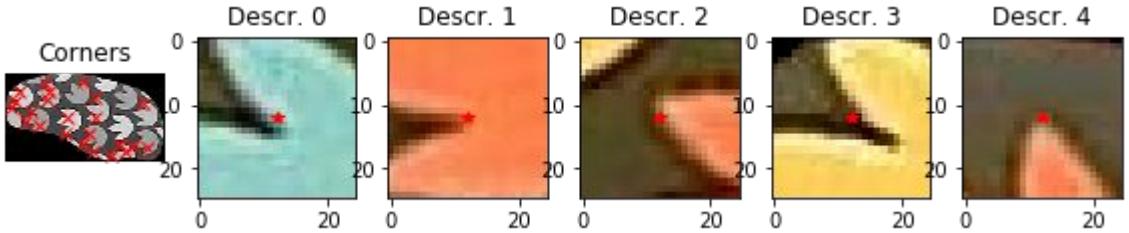


# Next practice session

*Twin it!*, again, with a slightly more elaborated approach:

1. Pre-select bubbles based on their colors ⇒ Color histograms
2. **For the pre-selected bubbles, check their content is similar**
   **⇒ Detect stable points and extract the patches around them**

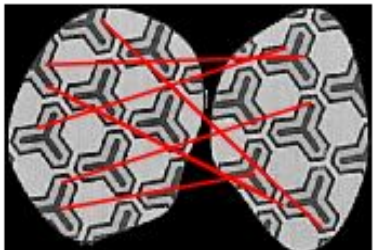


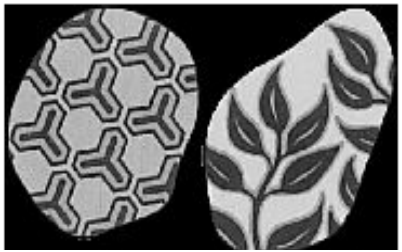


# Next practice session

*Twin it!*, again, with a slightly more elaborated approach:

1. Pre-select bubbles based on their colors ⇒ Color histograms
2. **For the pre-selected bubbles, check their content is similar**
   ⇒ Detect stable points and extract the patches around them
   **⇒ Compare (match) those patches**





# Image descriptors

# Issues with methods based on pixel comparison

What is important? What do they consider? **Raw pixels!**

⇒ We want to be able to make use of **domain knowledge**!

*Like sensitivity to shape, or dominant color information.*

They are terribly **slow** and works **only for small images.**

⇒ We want to **summarize an image** to a much smaller vector.

They are **sensible to rotation, scaling**, and many other perturbations.

⇒ We want to adjust sensitivity/invariance to perturbations.

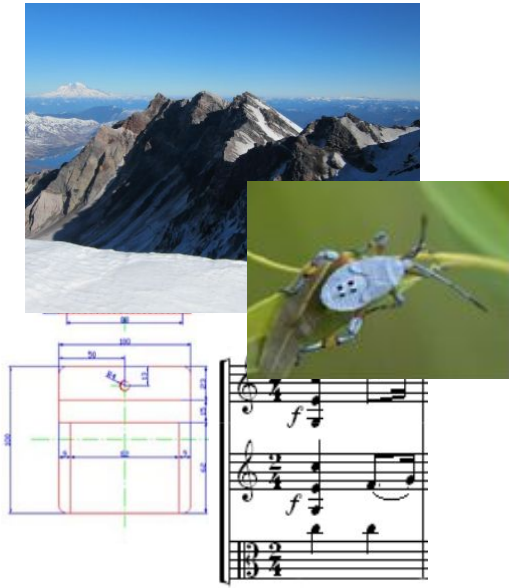
*Do we tolerate translation? Rotation? Intensity shift?*

How can we compare different pairs of images? **Metric issues.**

⇒ We want to be able to achieve **more than 1 vs all comparisons.**

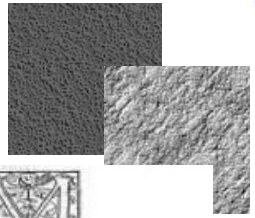
# Image descriptors: Overview

**Different sizes and contents ⇒ Different kind of descriptors**



**Large images, many parts**

**Complex small images**

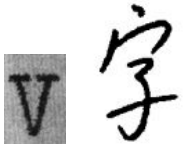


**Textured areas**

**Logos**

**Local image patches, lines, etc.**


**Words**



**Isolated symbols and letters**

≈ 500×500 px and more

≈ 20×20 px and less

# Image descriptors: Overview

**Different sizes and contents ⇒ Different kind of descriptors**

**Different problems ⇒ Different choices**

- Computation / memory constraints
- Which perturbations to we have to tolerate?
  *rotation, translation…*
- What is the expected output?
  *classification, detection, ranking, segmentation…*

**Many, many approaches ⇒ Impossible to list them all**

- Examples of several categories
- Focus on very useful or instructive ones