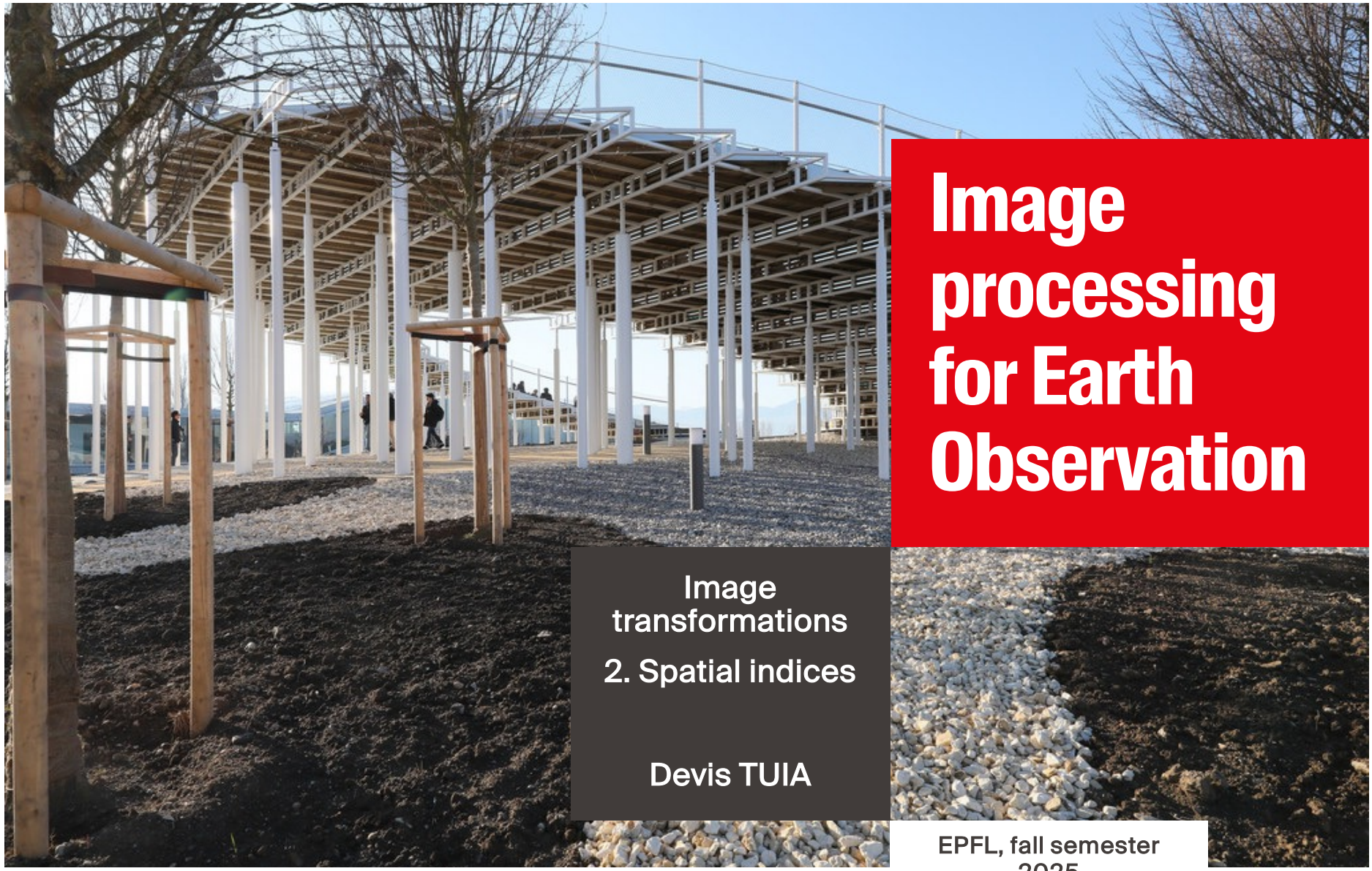


Image processing for Earth Observation

Image transformations

2. Spatial indices

Devis TUIA

École polytechnique fédérale de Lausanne

EPFL, fall semester 2025

Reminder: In the next two weeks

- We will study a number of information extraction techniques
  - spectral indices: enhance spectral relations between the bands of a pixel
  - spatial indices: extract information about spatial relationships
- We will also discuss how to deal with the increase in number of variables and see some data reduction techniques

More precisely

We will talk about how to describe local image appearance

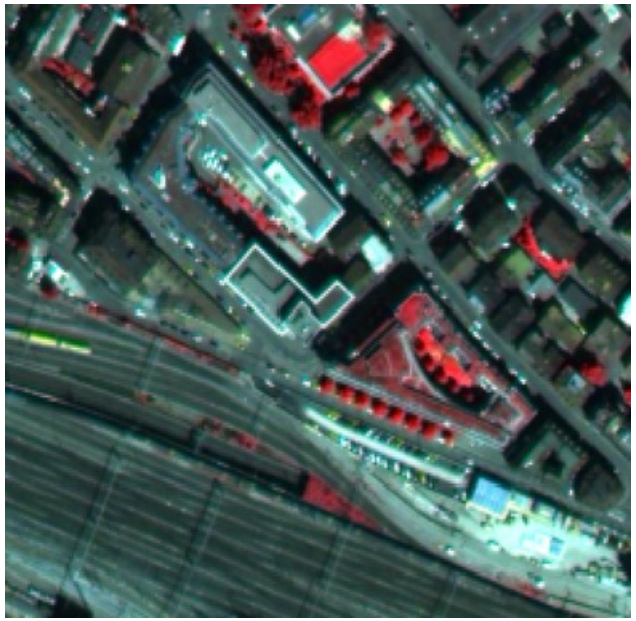
- Spatial supports
- Families of descriptors

What is appearance description?

- Images show ambiguous appearance, especially at pixel level.
- Color / spectra is often not discriminative.
- E.g. in RGB images.
- Can you tell me which land coverage are these?

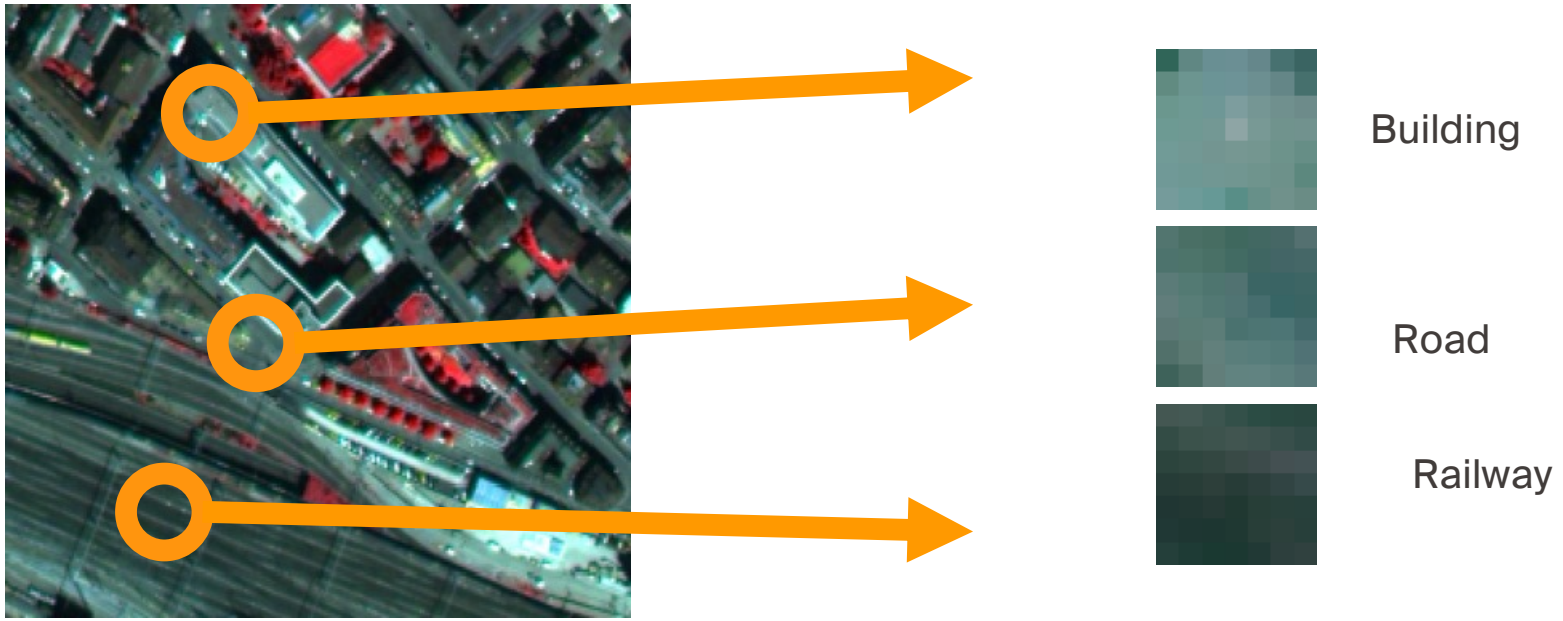
What is appearance description?

- This is particularly true at very high spatial resolution.
- Seeing the full image, you start to have an idea.



What is appearance description?

- But if you add spatial context and localisation, ambiguity is resolved!



Why?

- Let's consider the input pixel space
- Aerial / sat VHR imagery holds often little spectral information
- Same color /spectral signature is observed for different classes

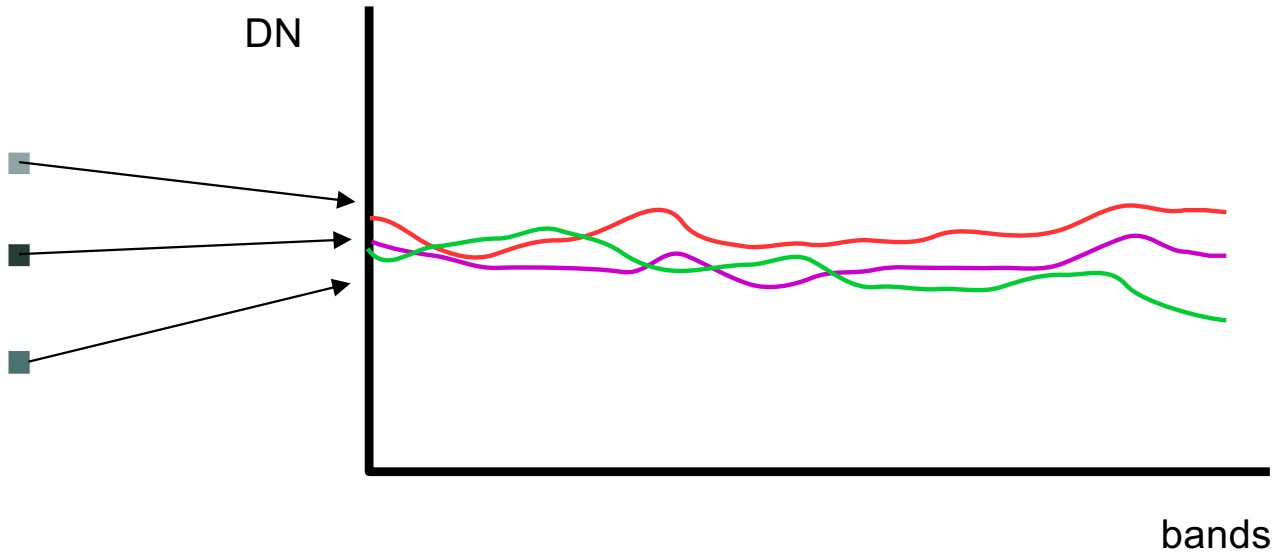
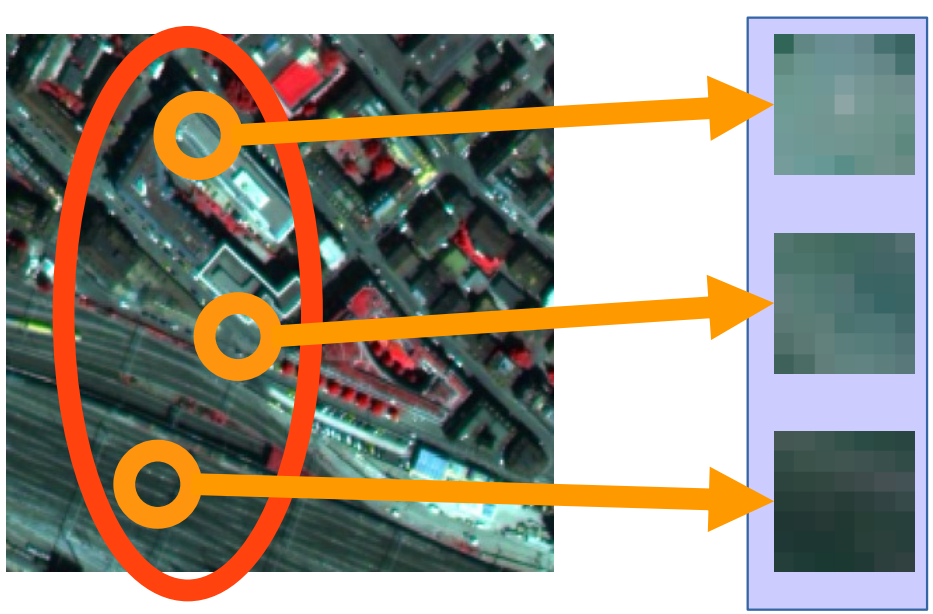
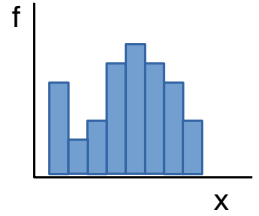
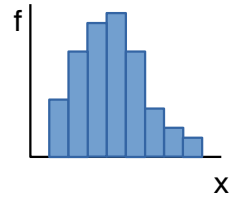
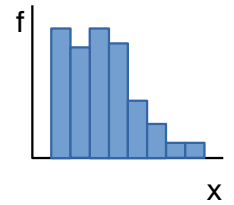


Image processing pipeline









Building

Road

Railways

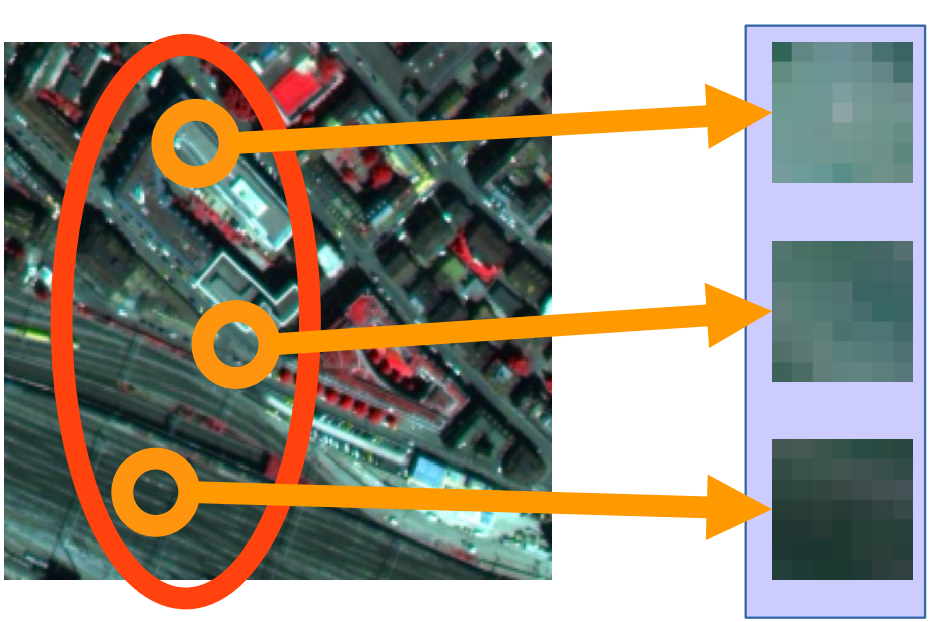
Image representation
→ SPATIAL SUPPORT

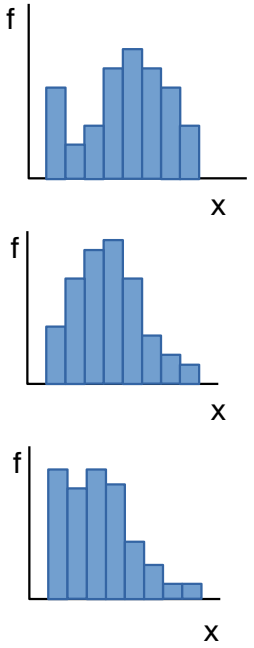
Image representation
→APPEARANCE DESCRIPTOR

Semantic class
→ CLASSIFIER
→ INFERENCE

D. Tuia. ECEO

Image processing pipeline





Building

Road

Railways

Image representation
→ SPATIAL SUPPORT

Image representation
→ APPEARANCE DESCRIPTOR

Semantic class
→ CLASSIFIER
→ INFERENCE

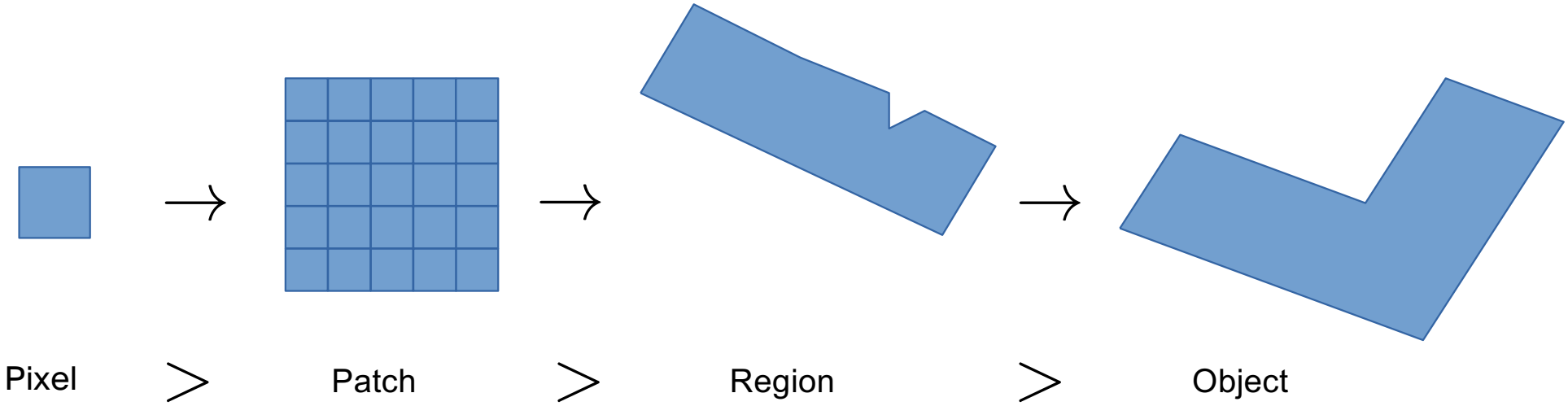
THIS COURSE

NEXT COURSES

Spatial supports

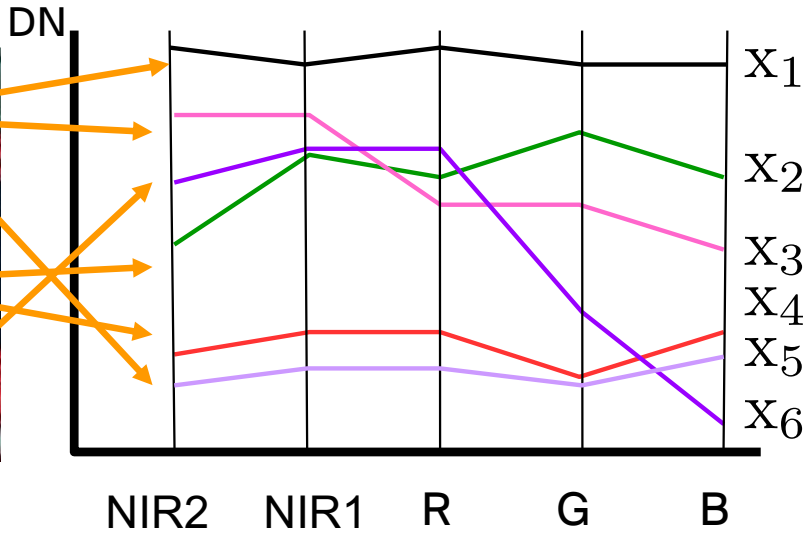
Spatial support types

- Defines *spatial boundaries* on which to compute descriptors
- Ideally, provide spatial boundaries *as close as possible to those of the objects*
- *Large enough* to contain relevant info, *tight enough* to preserve image resolution



1. Pixel

- Smallest possible mapping unit
- No spatial information
- Low(est)-level descriptor
- “Vector” of spectral information



$$\mathbf{X} = [\mathbf{x}_1^\top \dots \mathbf{x}_N^\top]^\top$$

$$\mathbf{X} \in \mathbb{R}^{N \times d}$$

$$\mathbf{x}_i = [x^{\mathrm{NIR2}} \quad x^{\mathrm{NIR1}} \quad x^{\mathrm{R}} \quad x^{\mathrm{G}} \quad x^{\mathrm{B}}]$$

1. Pixel  pros and cons

- ✓ Very simple to implement and run
- ✓ When the spectral information is rich, results can be good

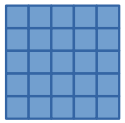
  → e.g. image spectroscopy channels + normalized band ratios

- ✖ No higher level semantic concepts

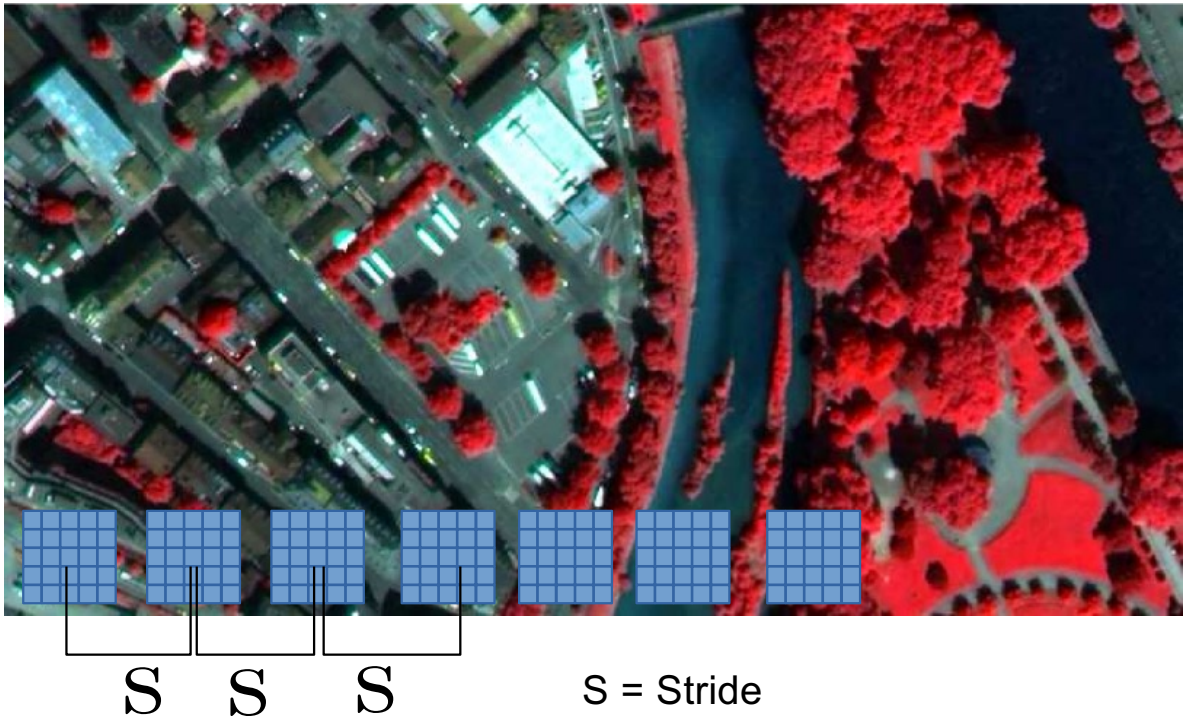
  → parking, highway, tar rooftop, railway banks = “asphalt”

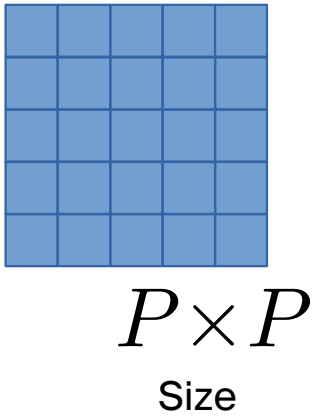
- ✖ Lots of data samples to be processed

  → e.g. 2000 x 2000 x 5 image = $4*10^6$ samples and $2*10^7$ floats

2. Patch 

- Square “windows”, simplest possible spatial information
- For each patch, a set of descriptors (e.g. spatial statistics) is extracted
- 2 hyperparameters: stride and size





2. Patch

The filter is defined by a *function* taking as input all the pixels contained in the image

$$\mathbf{x} \mapsto f(\mathbf{x}) = \mathbf{x}'$$
$$f : \mathbb{R}^{P \times P} \rightarrow \mathbb{R}$$

E.g. the spatial average

$$f(\mathbf{x}) = \frac{1}{P^2} \sum_{p=1}^{P^2} \mathbf{x}_p$$

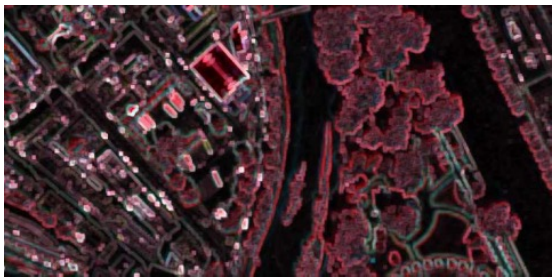
(or expressed as a convolution:

$$f(\mathbf{x}) = \mathbf{W}_p \star \mathbf{x}$$
$$\mathbf{W}_p = \frac{1}{P^2}$$

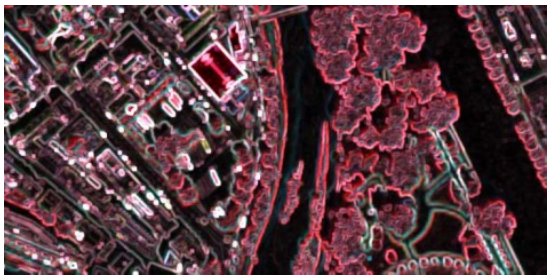
Spatial statistics on patches

- By using $S = 1$, we create an image with the same spatial size of the original but with “new” signals
- Each pixel is now explicitly dependent on neighbors
- Compute descriptors related to local moments
- Family of descriptors to use usually problem / domain specific

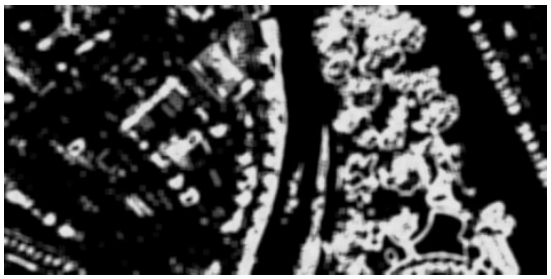
$$\mathbf{X} = \left[\mathbf{x}_1^\top \dots \mathbf{x}_N^\top\right]^\top$$



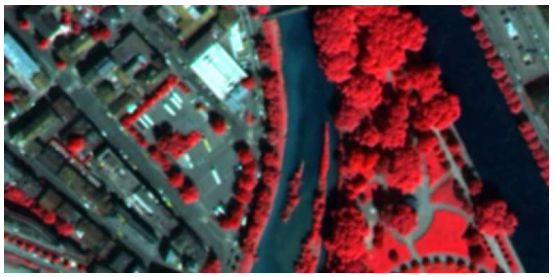
St. dev



Range

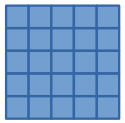


Energy



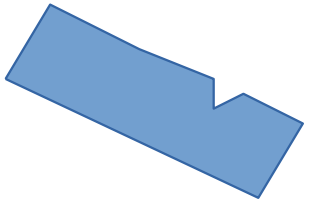
Average

2. Patch pros and cons



- ✔ Very simple to implement functions taking as inputs square patches
- ✔ Lots of descriptors can be stacked (in *d*-dimensional arrays) and processed as *standard images*
- ✔ Off the shelf implementations in many software
- ✖ Object boundaries not preserved by spatial statistics
- ✖ The spatial extent of the filtered image is the same (stride = 1), or the resolution degraded (stride > 1)

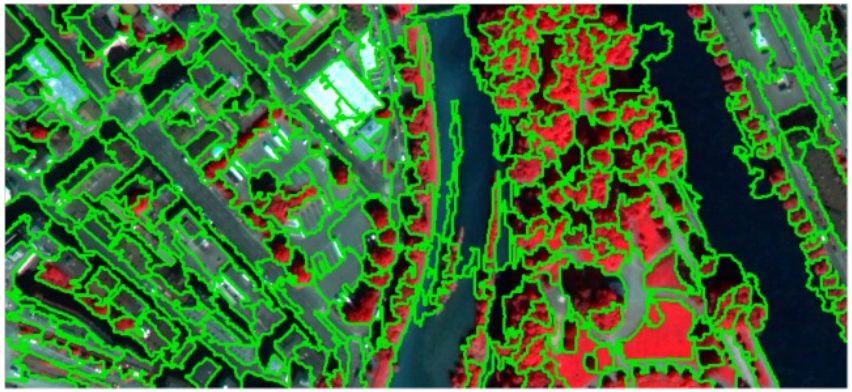
3. Regions / superpixels



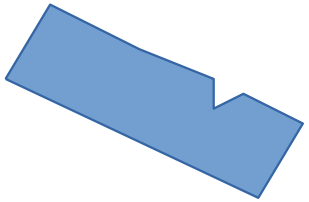
- Edges of regions correspond to gradients in image
- For each region, a set of descriptor is extracted
- Many hyperparameters, depending on method to be used (thresholds, min size, smoothing, etc)

Each region contains *similar* pixels

Decompose the image into *nonoverlapping* regions

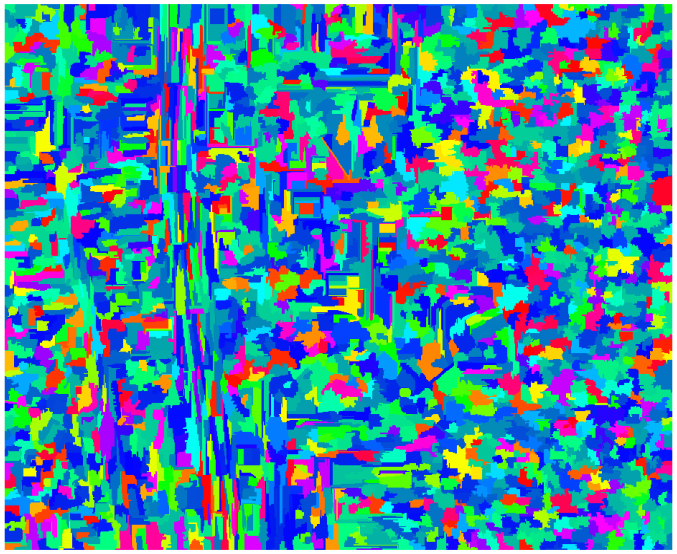


3. Regions / superpixels



- Cluster pixels in groups that are
  - Nearby
  - Of similar spectral properties

Superpixels (each color is a SP)

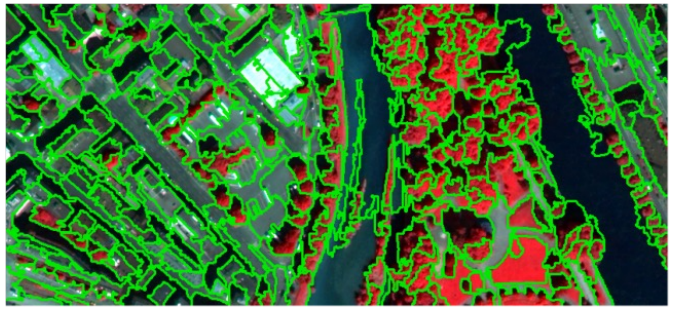


3. Regions / superpixels

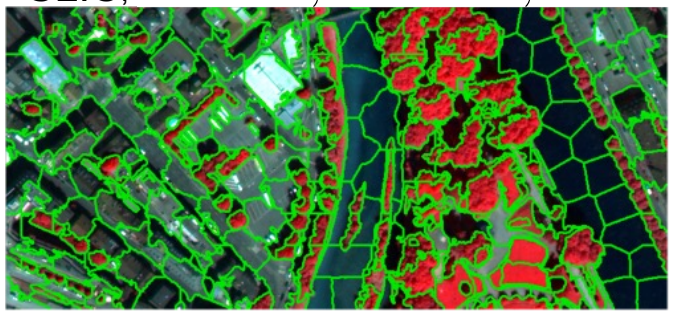
- Different systems producing such decompositions exist
- Each region is uniform under some criteria (color variance, texture, homogeneity, etc.)
- Compute discriminative / informative descriptors from each region
- Problem decomposed in N^r regions, and $N^r << N$

(here ~200 to 500 instead of 10E6)

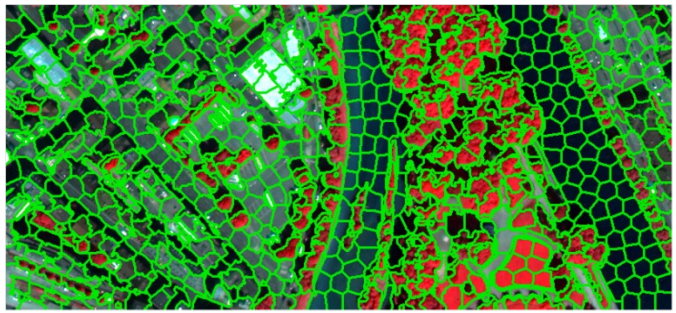
Felzenszwalb and Huttenlocher



SLIC, $\sigma = 0.5$, $\tau = 500$, $m = 20$

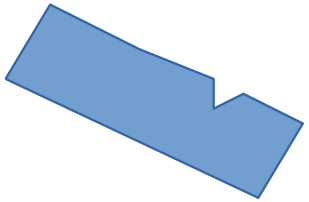


SLIC, $\sigma = 0.5$, $\tau = 500$, $m = 40$

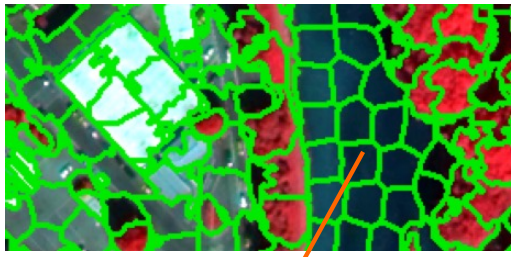


D. Tuia. ECEO

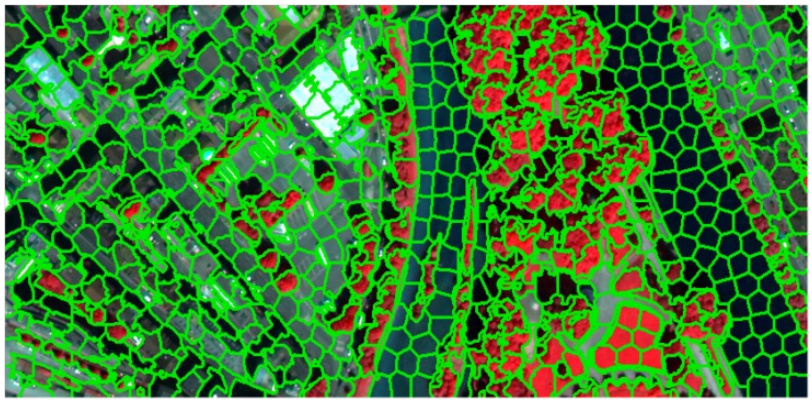
3. Regions / superpixels

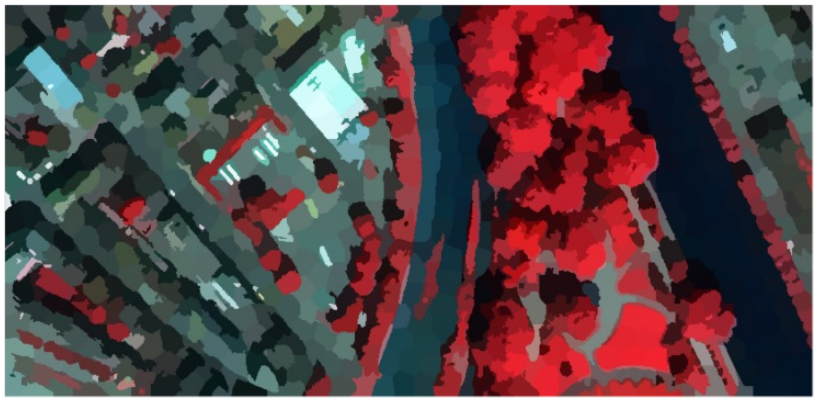


- Same reasoning as for “patch” descriptors, but the size of each region $|R|$ is different!

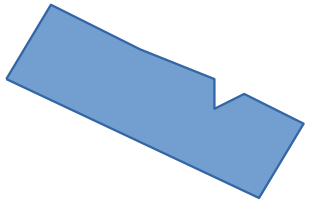


$$f(\mathbf{x}) = \frac{1}{|R|} \sum_{p=1}^{|R|} \mathbf{x}_p$$



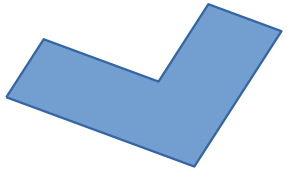


3. Regions /spix pros and cons



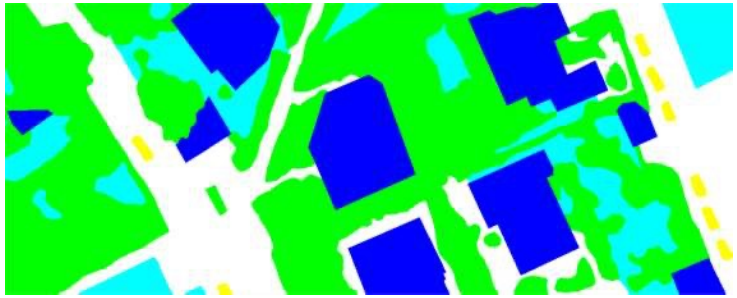
- ✔ Natural border of the objects composing the image are preserved
- ✔ Size of the problem greatly reduced without loss of resolution (label assumed to be homogeneous in each region)
- ✔ Off the shelf libraries to compute regions
- ✖ Regions are not representative for real *objects* (1 object = 1 or + regions)
- ✖ Errors in region computation cannot be recovered in later steps
  → regions become the image “atomic regions”

4. Objects

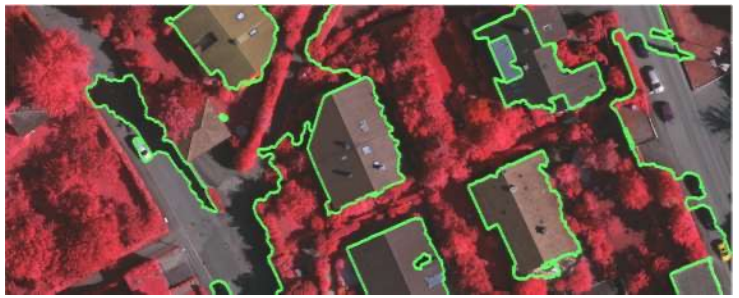


- Edges correspond to *real* object boundaries
- Each object has fully described appearance
- Object segmentation is usually harder than classifying all pixels!

+



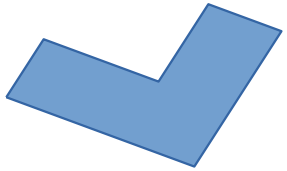
Objects reference



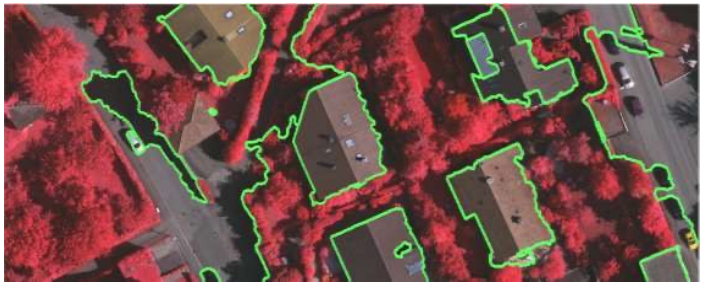
"Building" proposals

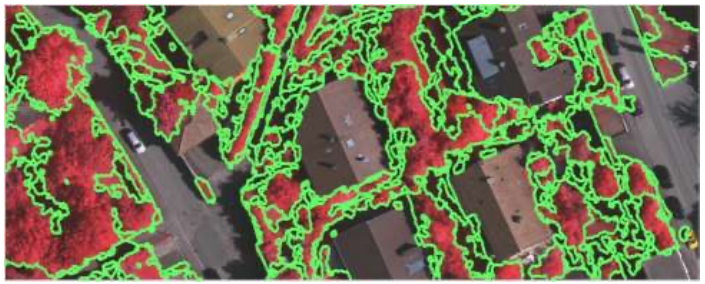
"Car" proposals

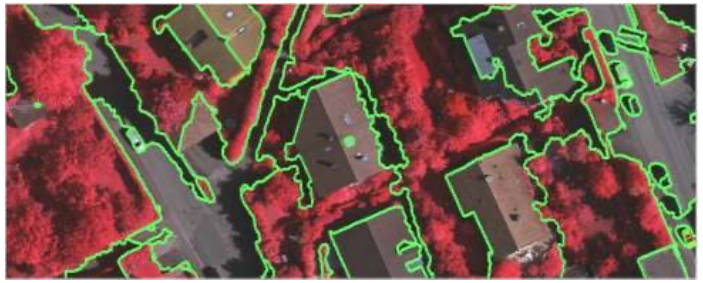
4. Objects



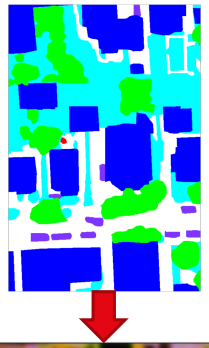
- Usually obtained by merging regions
- Ideally, an image image decomposed perfectly has O < R << N
- It is common to work on *object proposals* O^p, and to select among them the ones best scored by our models
- Problem decomposed in O^p objects, and $R < O^p << N$
- *Information contained in O is semantically meaningful, while on pixel or regions it is not!*







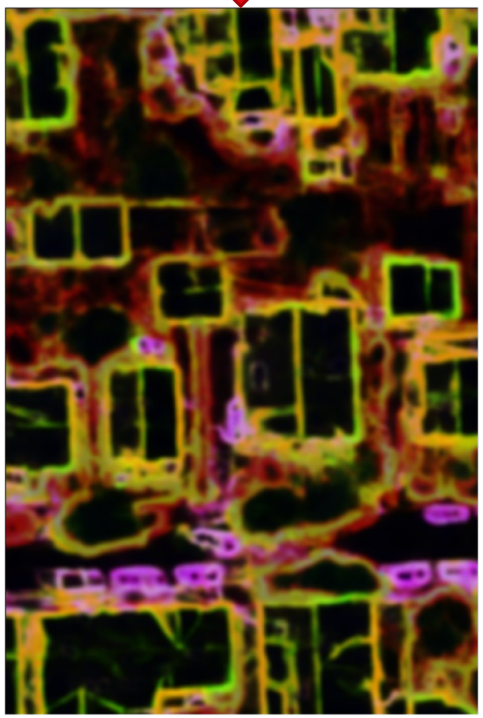
Creating object proposals



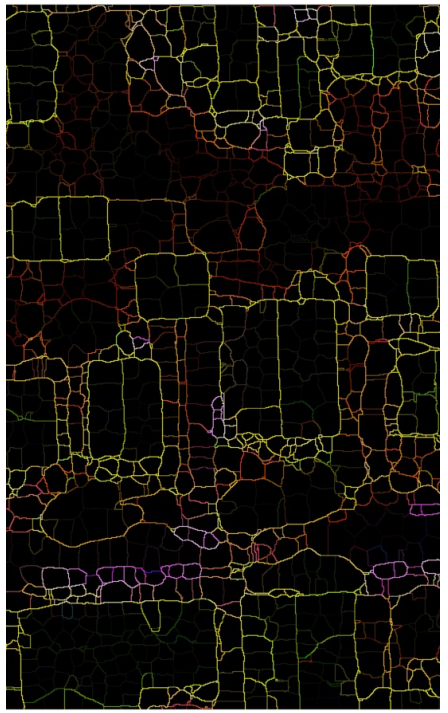
Object proposals

Image

Superpixels

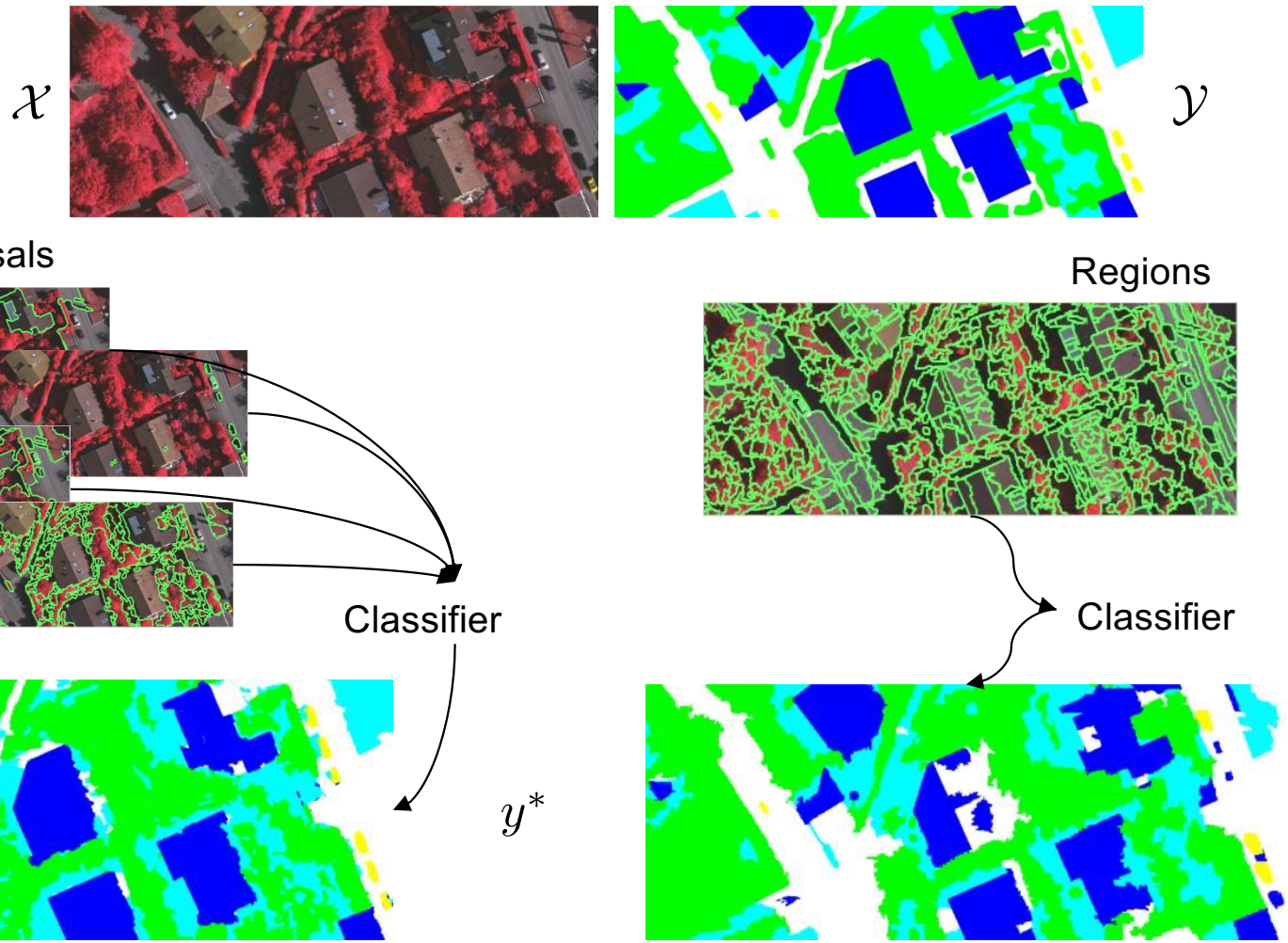


Class-specific borders
(learned with machine learning model)

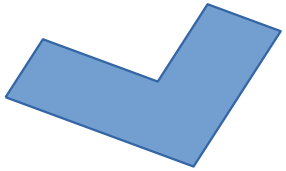


M. Volpi and D. Tuia. Deep multi-task learning for a geographically-regularized semantic segmentation of aerial images. *ISPRS J. Int. Soc. Photo. Remote Sens.*, 144:48–60, 2018.

Regions vs objects



4. Objects pros and cons



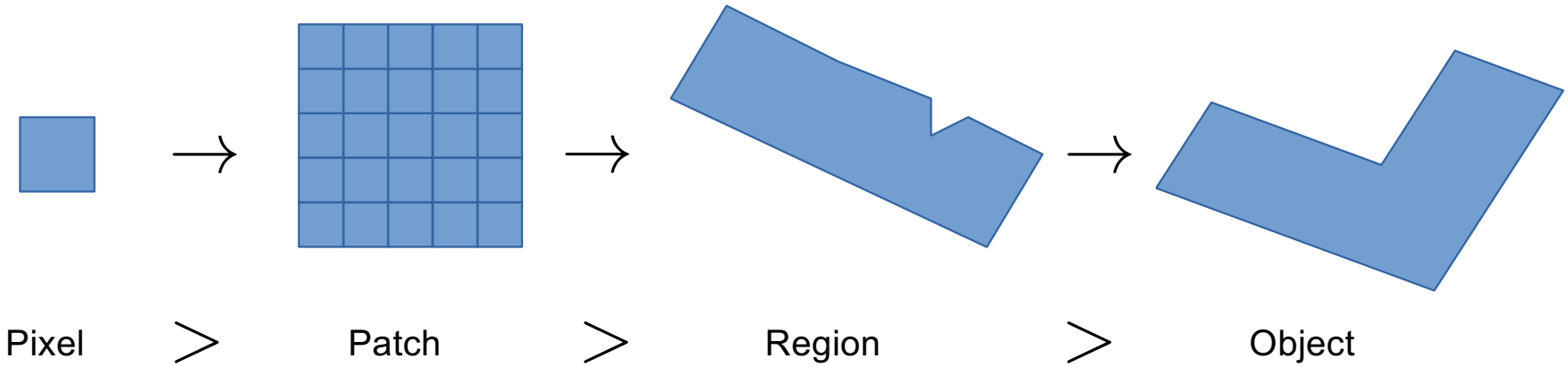
- ✔ Natural edges AND “semantic edges” are preserved, if they correspond
- ✔ Objects are not homogeneous in color, but their label is! (semantic + sensory gap)
- ✔ Allows to learn exhaustive and informative aspects of our data

  → e.g. rooftops are composed by tar, concrete, chimneys, windows, veg.
- ✖ Objects can be trivially decomposed in regions, the opposite is hard
- ✖ Most of the time, finding objects is a very difficult task since size and shape is varying significantly across classes

Which one to chose?

Always study the problem:

- If spectral information is sufficient → go pixel
- If spatial resolution is not a concern → patches
- If unsure → regions
- If good geometry and you're good with statistical models → object



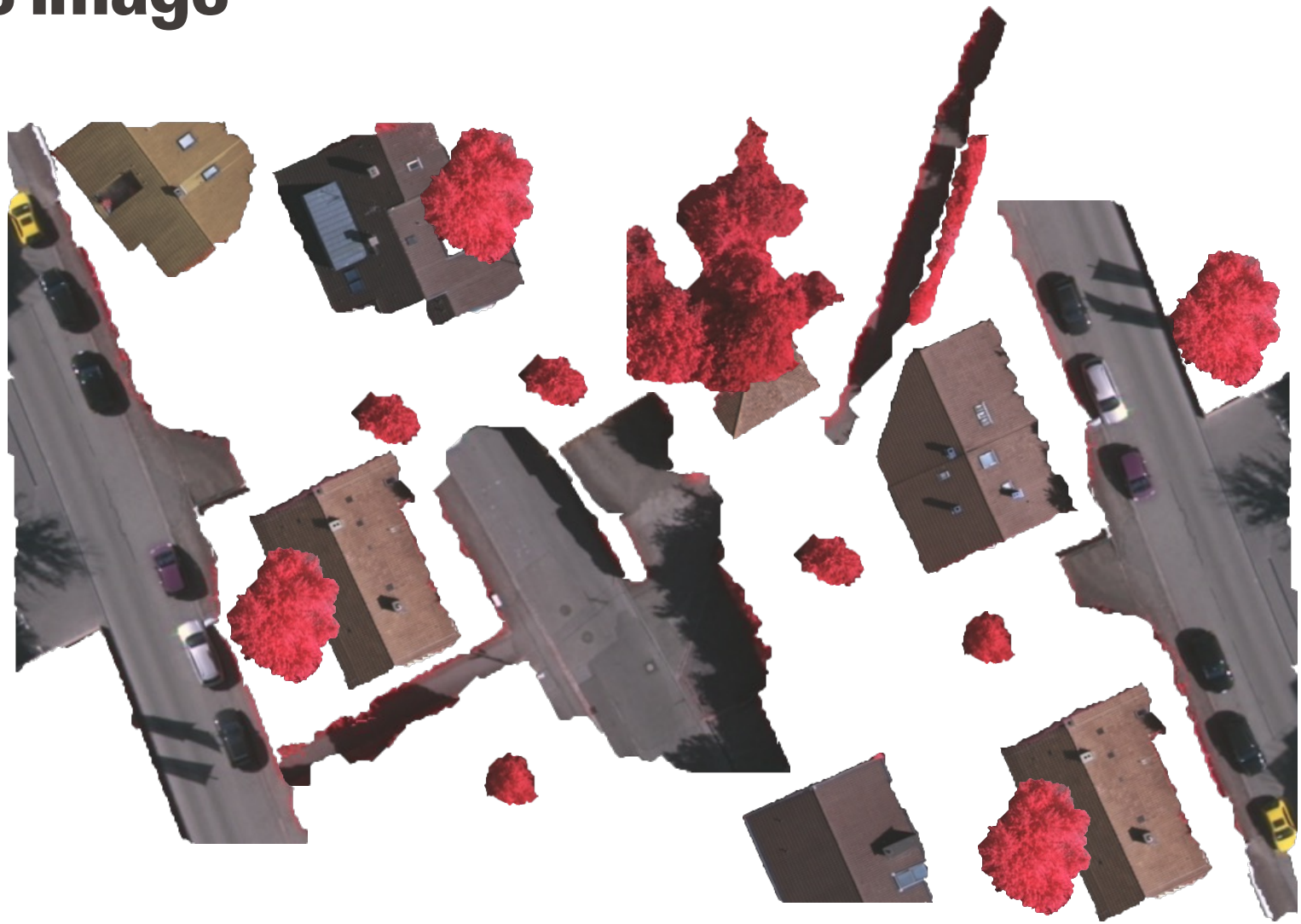
Appearance descriptors

What to extract? And why?

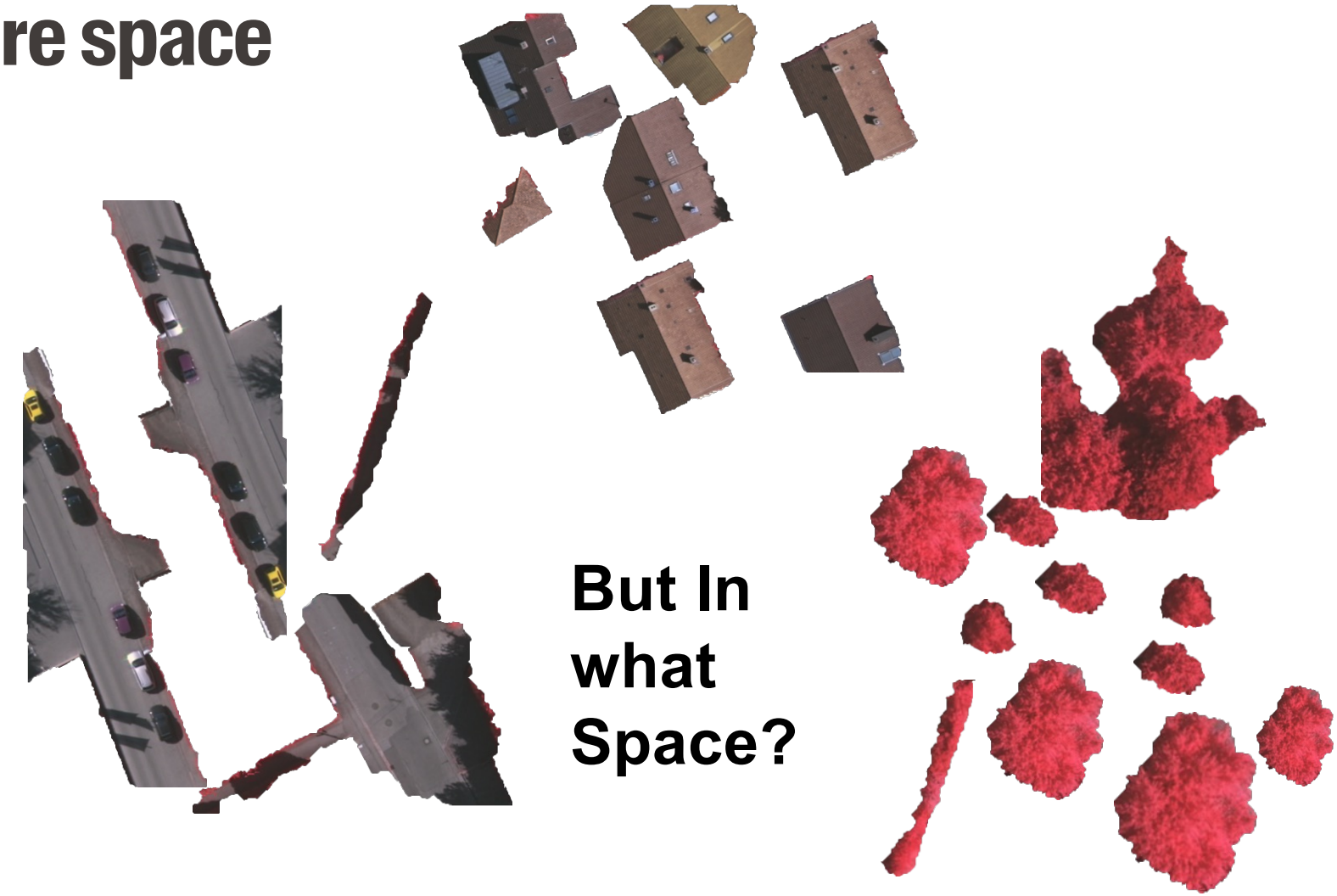
Images are spatial entities, and so is their content

- Color and spatial arrangement of colors (e.g. texture, gradients, orientations) are equally important to label and detect things
- Important aspects can be approximated by simple statistics (generally first [e.g. mean, histogram] and second moments [e.g. standard deviation, variance])

From the image space...



To a meaningful feature space

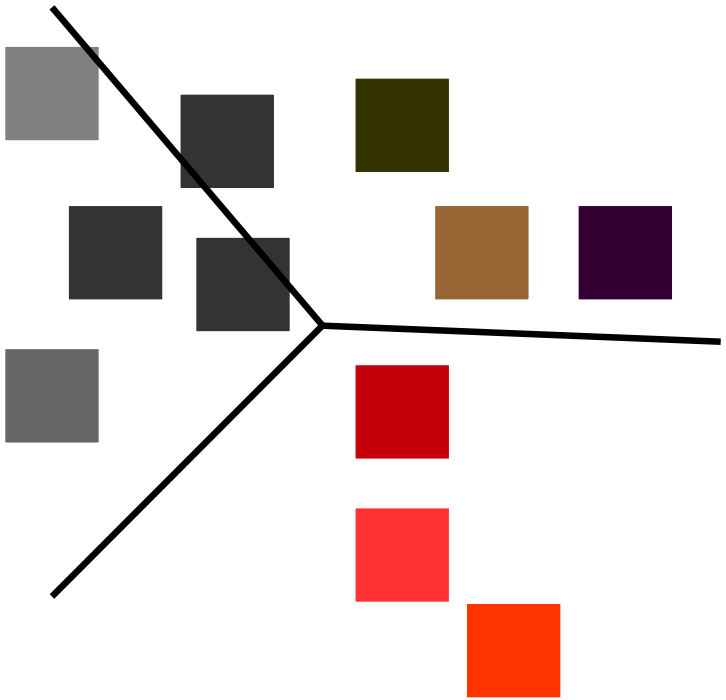


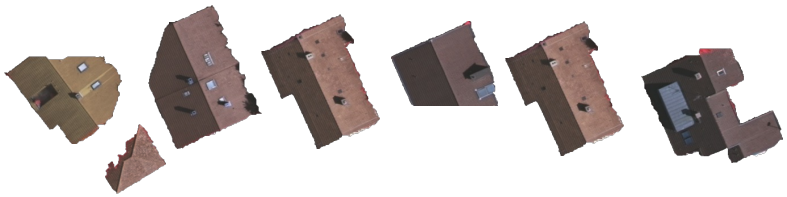
Example 1: Average color per region

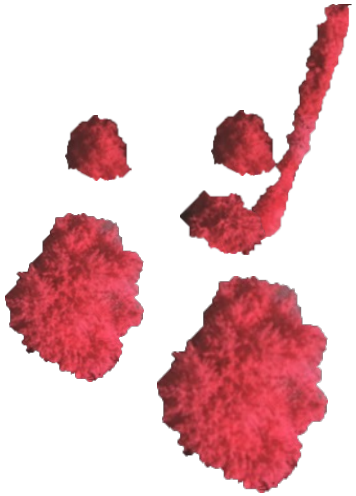
R is the set containing all the pixels *p* in R

$$\bar{\mathbf{x}} = \frac{1}{|R|} \sum_{p=1}^{|R|} \mathbf{x}_p$$

$$\bar{\mathbf{x}} = [\bar{R} \quad \bar{G} \quad \bar{B}]$$





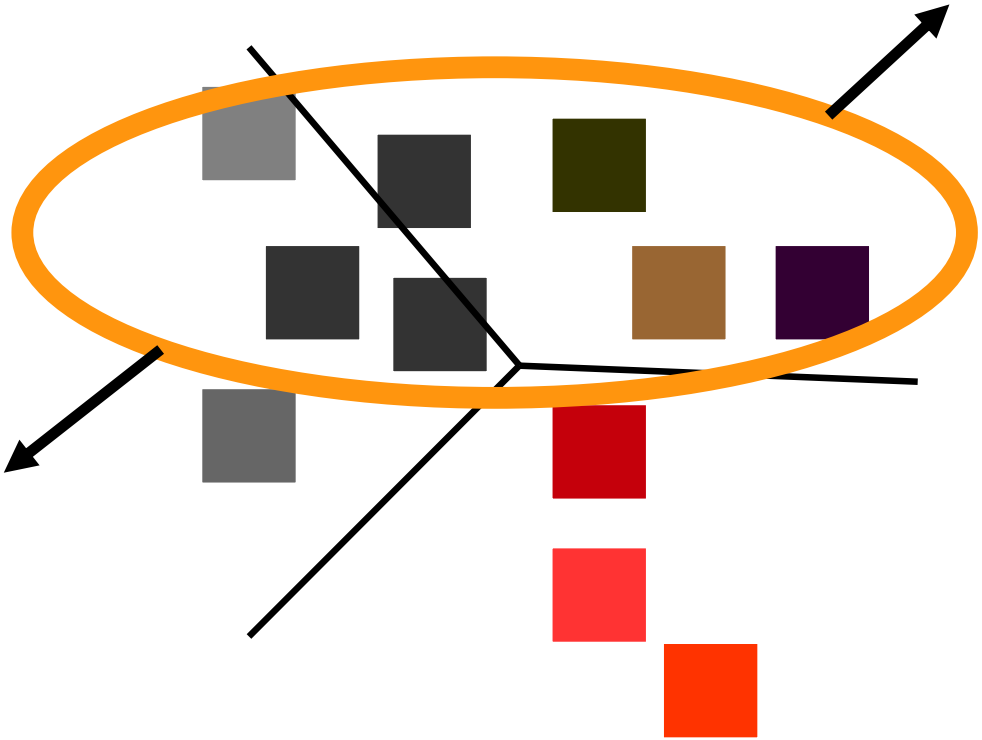


Example 1: Average color per region

$$\bar{\mathbf{x}} = \frac{1}{|R|} \sum_{p=1}^{|R|} \mathbf{x}_p$$

$$\bar{\mathbf{x}} = [\bar{R} \quad \bar{G} \quad \bar{B}]$$

The average color is not discriminative enough between objects with similar colors!



It tells little about texture or spatial arrangement!

Appearance descriptors

- Encode *discriminative* statistical properties of the patch / regions / objects
- Try to encode what you see, what makes a road a “road” and a rooftop a “rooftop” ?
- Most of the time, stack together all the information you can!

$$\mathbf{x}_i = [\mathbf{x}_i^{\text{av}} \quad \mathbf{x}_i^{\text{std}} \quad \mathbf{x}_i^{\text{entr}} \quad \mathbf{x}_i^{\text{hist}} \quad \dots]$$

i is the index of the patch / region / object.

Example descriptors: average color in the region

$$\mathbf{x}_i = [\mathbf{x}_i^{\text{av}} \quad \mathbf{x}_i^{\text{std}} \quad \mathbf{x}_i^{\text{entr}} \quad \mathbf{x}_i^{\text{hist}} \quad \dots]$$

(information about average spectral signature / color)

$$\bar{\mathbf{x}} = \frac{1}{|R|} \sum_{j=1}^{j \in R} \mathbf{x}_j$$

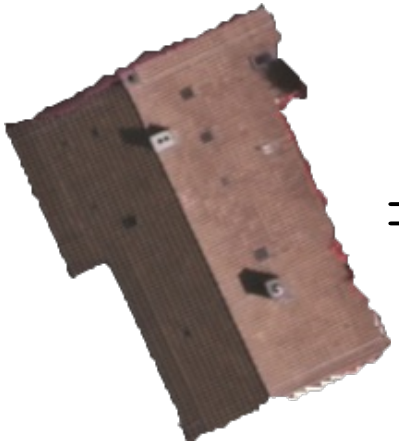
$$= [x_i^{\text{av,R}} \quad x_i^{\text{av,G}} \quad x_i^{\text{av,B}}]$$

Example descriptors: standard deviation in the region

$$\mathbf{x}_i = [\mathbf{x}_i^{\text{av}} \quad \mathbf{x}_i^{\text{std}} \quad \mathbf{x}_i^{\text{entr}} \quad \mathbf{x}_i^{\text{hist}} \quad \dots]$$

(information about variance of the color, in each object)

$$\mathbf{x}_i^{\text{std}} = \sqrt{\frac{1}{|R|}\sum_{j \in R}(\mathbf{x}_j - \bar{\mathbf{x}})^2}$$



$$= [x_i^{\text{std,R}} \quad x_i^{\text{std,G}} \quad x_i^{\text{std,B}}]$$

Example descriptors: entropy in the region

$$\mathbf{x}_i = [\mathbf{x}_i^{\text{av}} \quad \mathbf{x}_i^{\text{std}} \quad \mathbf{x}_i^{\text{entr}} \quad \mathbf{x}_i^{\text{hist}} \quad \dots]$$

(information about variance and "disorder" of color values, spatially)

$$\mathbf{x}_i^{\text{entr}} = -\sum_{j=1}^{\text{nbins}} p_j \log(p_j)$$

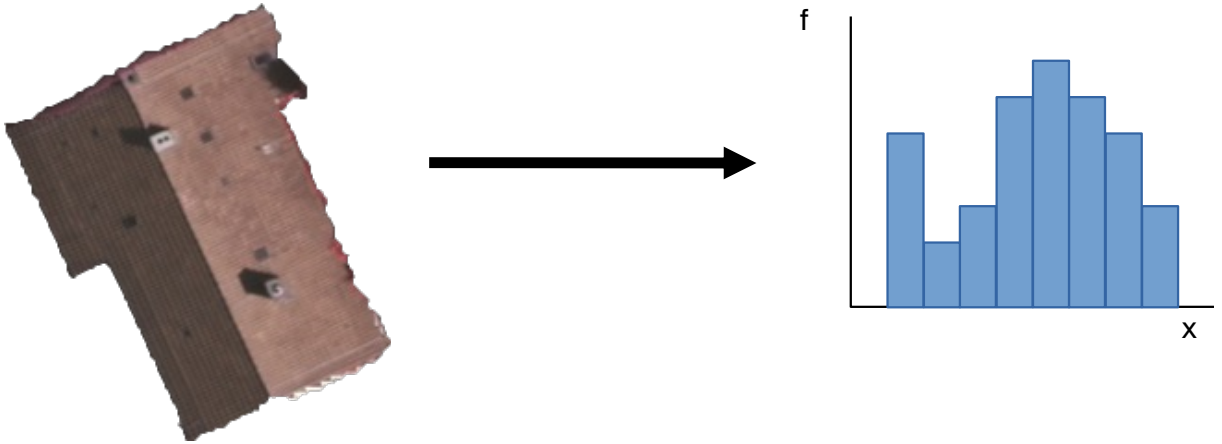
p_j is the histogram of the region, with colors divided in bins

$$= [x_i^{\text{entr,R}} \quad x_i^{\text{entr,G}} \quad x_i^{\text{entr,B}}]$$

Example descriptors: histogram of colors in the region

$$\mathbf{x}_i = [\mathbf{x}_i^{\text{av}} \quad \mathbf{x}_i^{\text{std}} \quad \mathbf{x}_i^{\text{entr}} \quad \mathbf{x}_i^{\text{hist}} \quad \dots]$$

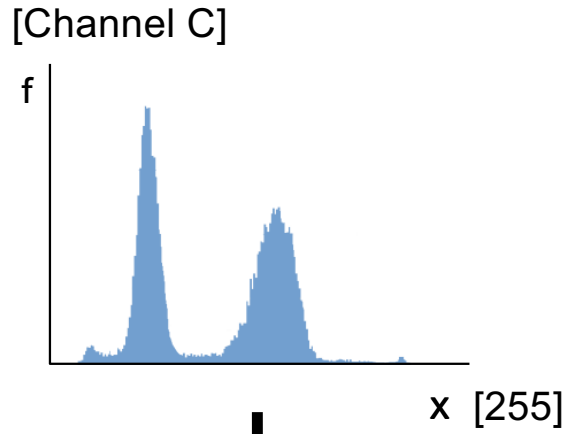
Distribution of the color (information frequency of given color ranges)



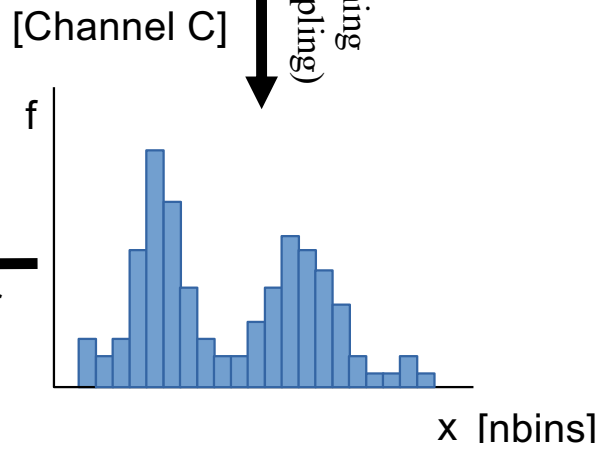
Building the histogram

Count DN occurrences

Per each channel



Binning (resampling)



Vectorization

(1 such vector per color channel !)

$$\mathbf{x}_i^{\text{hist}} = [\mathbf{x}_i^{\text{hist,R}} \quad \mathbf{x}_i^{\text{hist,G}} \quad \mathbf{x}_i^{\text{hist,B}}]$$

$$\mathbf{x}_i^{\text{hist,C}} = [f_i^1 \quad f_i^2 \quad \dots \quad f_i^{\text{nbins}}]$$

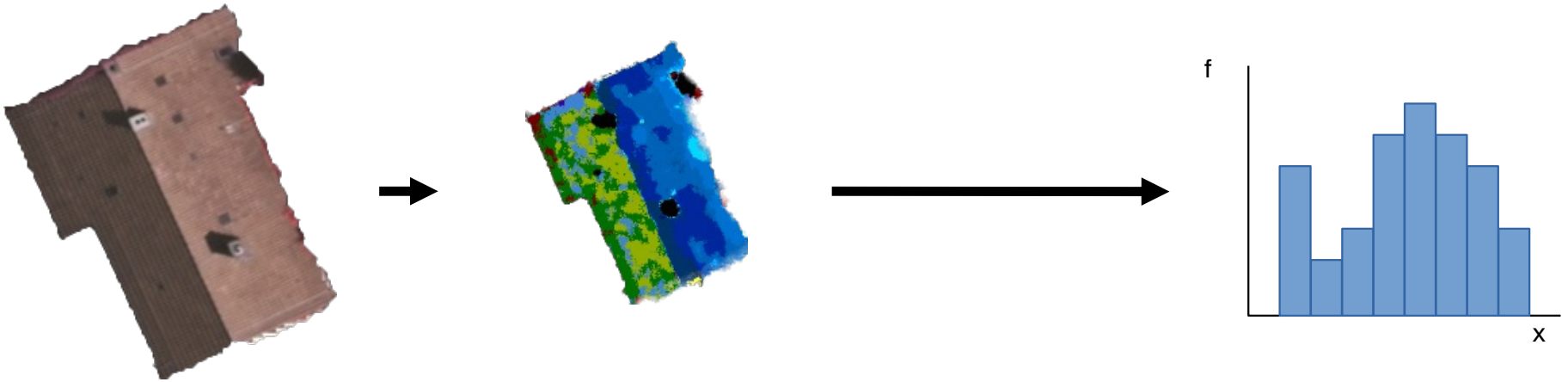
$$\mathbf{x}_i^{\text{hist,C}} \leftarrow \sum f = 1$$

e.g. $[0.01 \quad 0.04 \quad \dots \quad 0.1]$

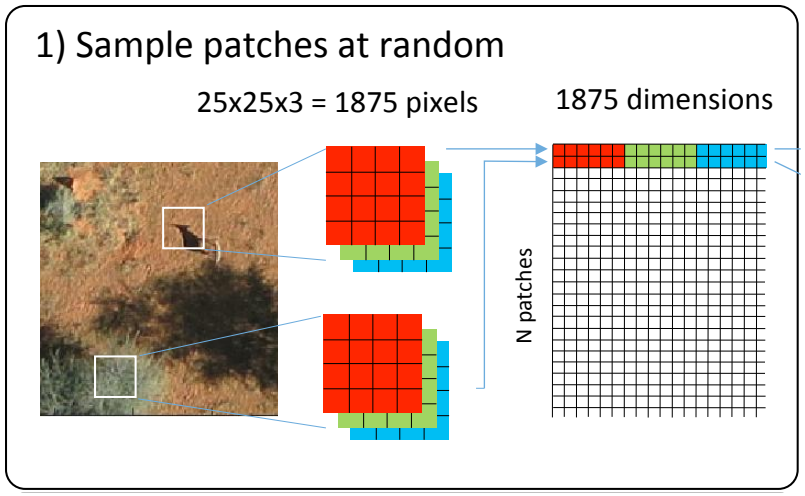
Example descriptors: bag of visual words (BOW)

$$\mathbf{x}_i = [\mathbf{x}_i^{\text{av}} \quad \mathbf{x}_i^{\text{std}} \quad \mathbf{x}_i^{\text{entr}} \quad \mathbf{x}_i^{\text{hist}} \quad \mathbf{x}_i^{\text{bow}} \quad \dots]$$

Distribution of *features*
Or spatial prototypes



Building the BOW (1)



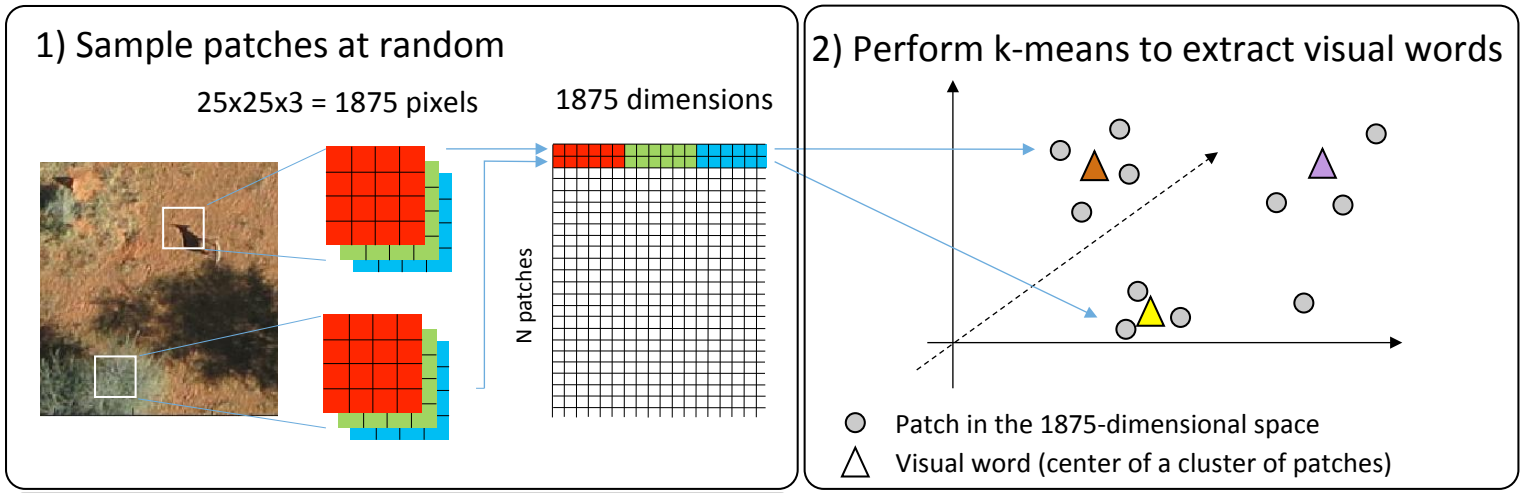
- extract random patches from the image,
- enough to have all representative structures
- Every patch is a 1875 dim vector, 1 dimension per pixel composing it

We want to use this 1875-dimensional space as a space to extract the descriptors from

But how?

N. Rey, M. Volpi, S. Joost, and D. Tuia. Detecting animals in african savanna with UAVs and the crowds. *Remote Sens. Environ.*, 200C:341–351, 2017.

Building the BOW (2)



- Run a k-means, i.e. cluster the patches into a number of representative types
- Each cluster center is a typical pattern seen in the images
- We call them **visual words**

N. Rey, M. Volpi, S. Joost, and D. Tuia. Detecting animals in african savanna with UAVs and the crowds. *Remote Sens. Environ.*, 200C:341–351, 2017.

A brief explanation of k-means

D. Tuia. ECEO

- It's a method to partition the data into groups, in an unsupervised way
- Aims at finding compact clusters

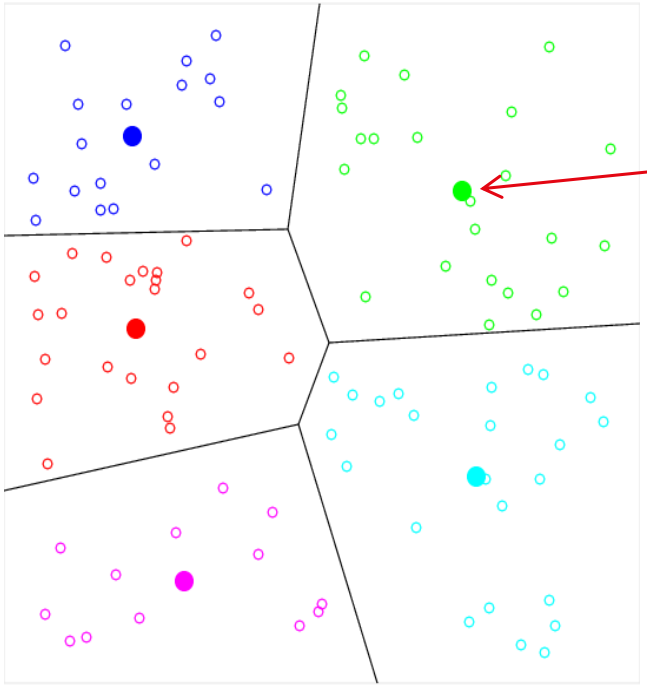
  → minimize variance within the clusters!

- Variance? (statistics course)

$$\mathrm{var}(x) = \sum_i (x_i - \mu)^2$$

Mean of x

Example of k-means (k = 5)



Mean of data assigned
to the green cluster
= Centroid of the cluster μ

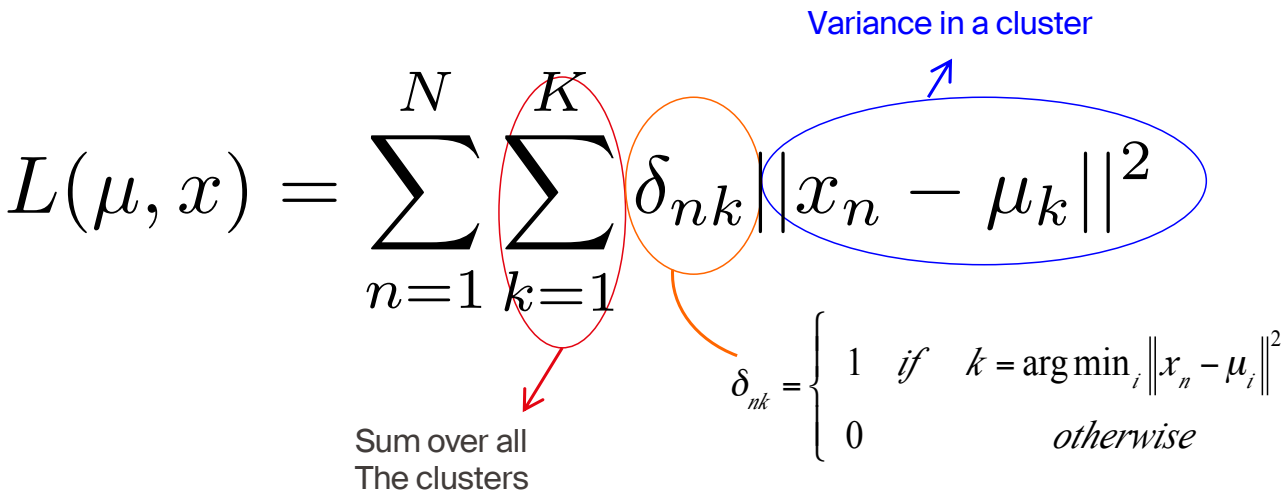
The centroids is
What we are interestd in!

Because they “represent well”
Common patterns in the data!

D. Tuia. ECEO

k-means objective function

- We define a cost function L(m,x) in this sense:



- The smaller L(m,x) is, the more desirable the solution.

D. Tuia. ECEO

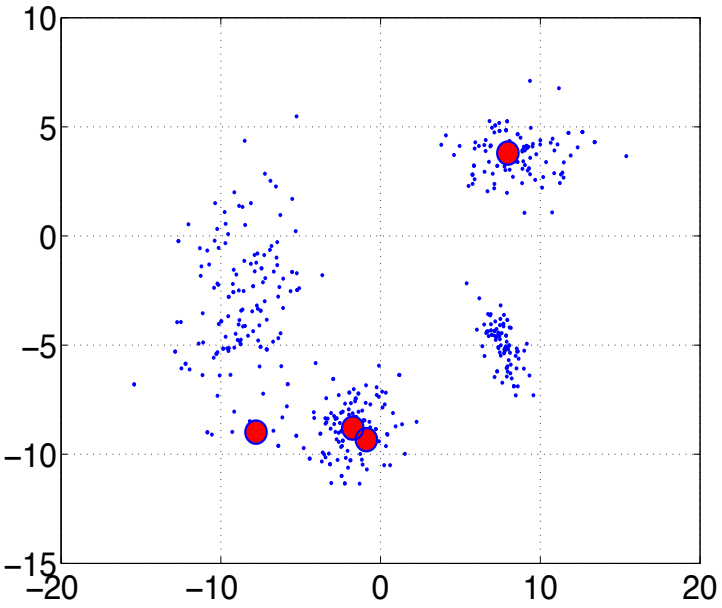
k-means in a nutshell

- K-means is an iterative method
- It starts with a guess of the cluster centers (often random)
- Computes the assignment by minimizing L
- Updates the centers as the means of the samples assigned to each center
- Repeats 2.-3. until stability is reached.

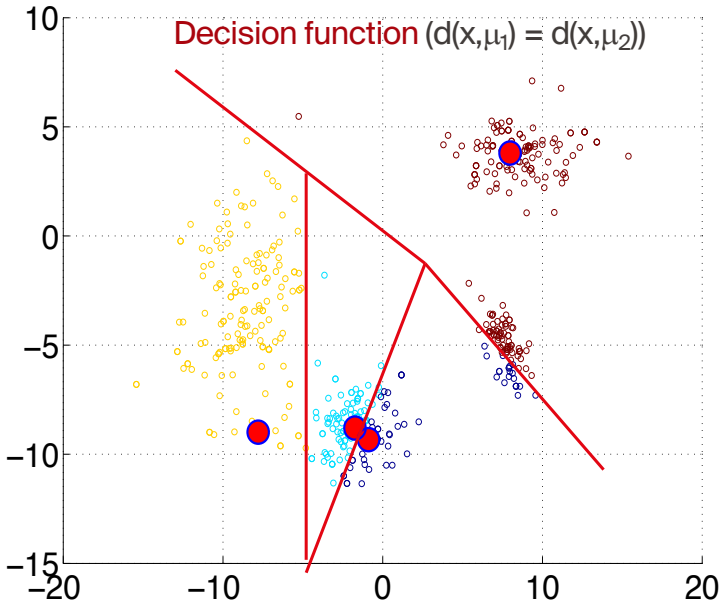
- Stability can be: - a fixed number of iterations
- - when the centroids do not move anymore

D. Tuia. ECEO

Example



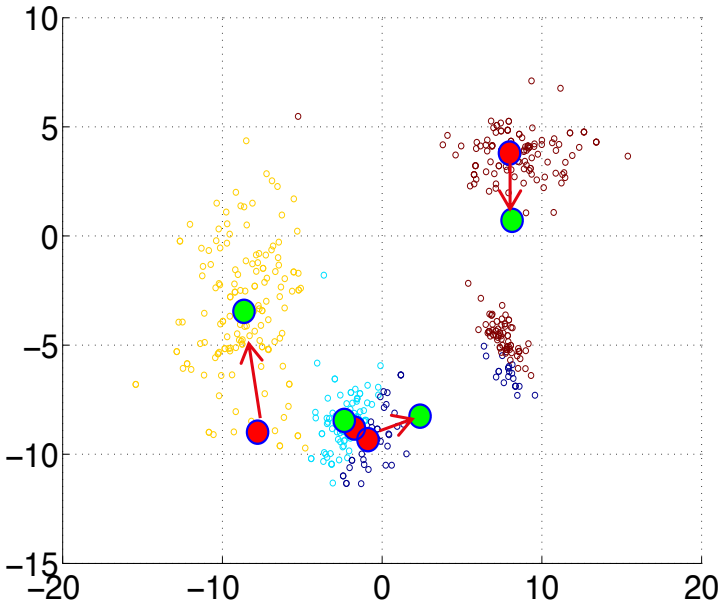
Data and initial (random) centroids

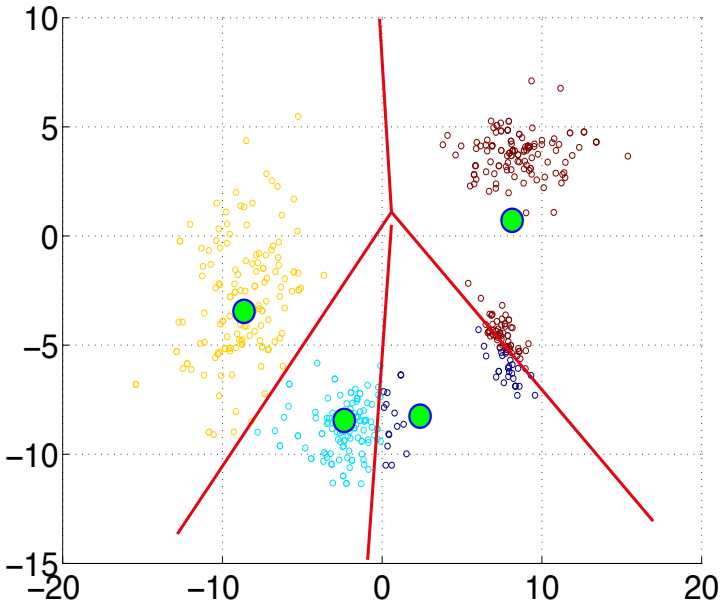


Initial cluster assignment

D. Tuia. ECEO

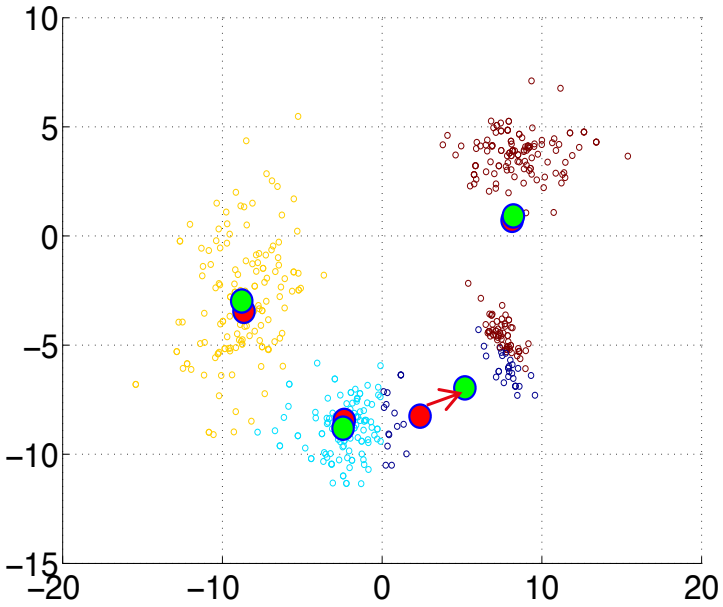
Iteration 1: solving 3 clusters

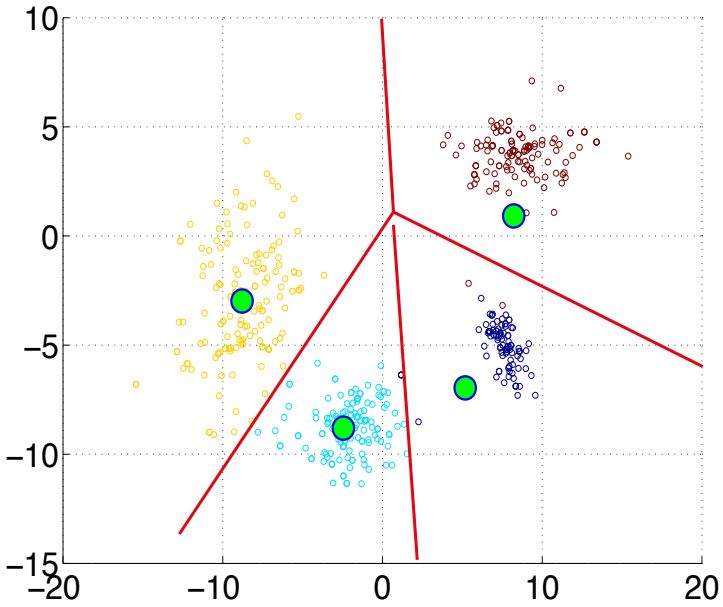




D. Tuia. ECEO

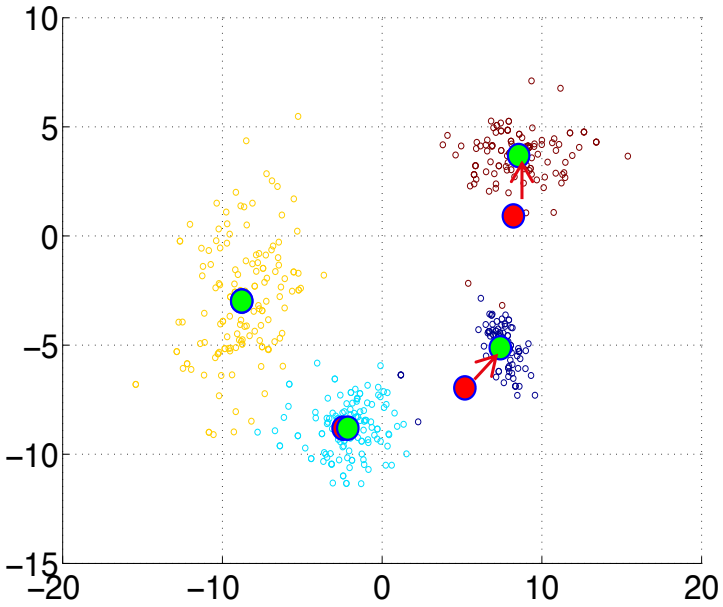
Iteration 2: solving the blue cluster

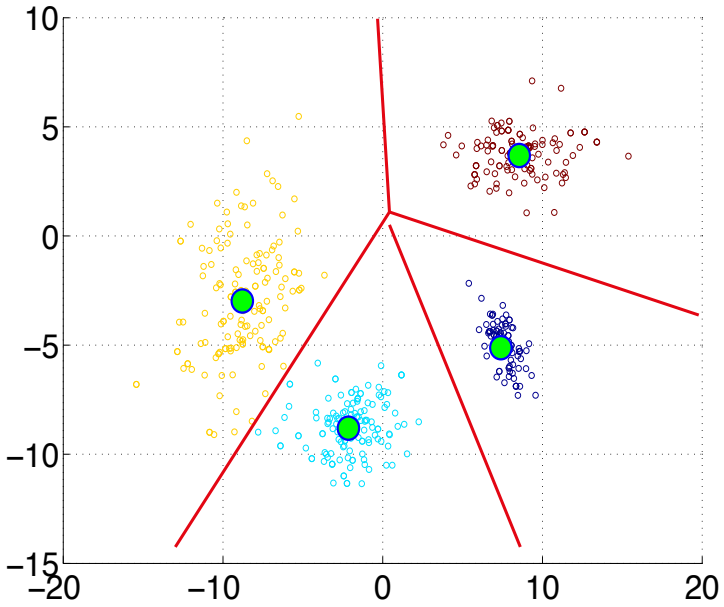




D. Tuia. ECEO

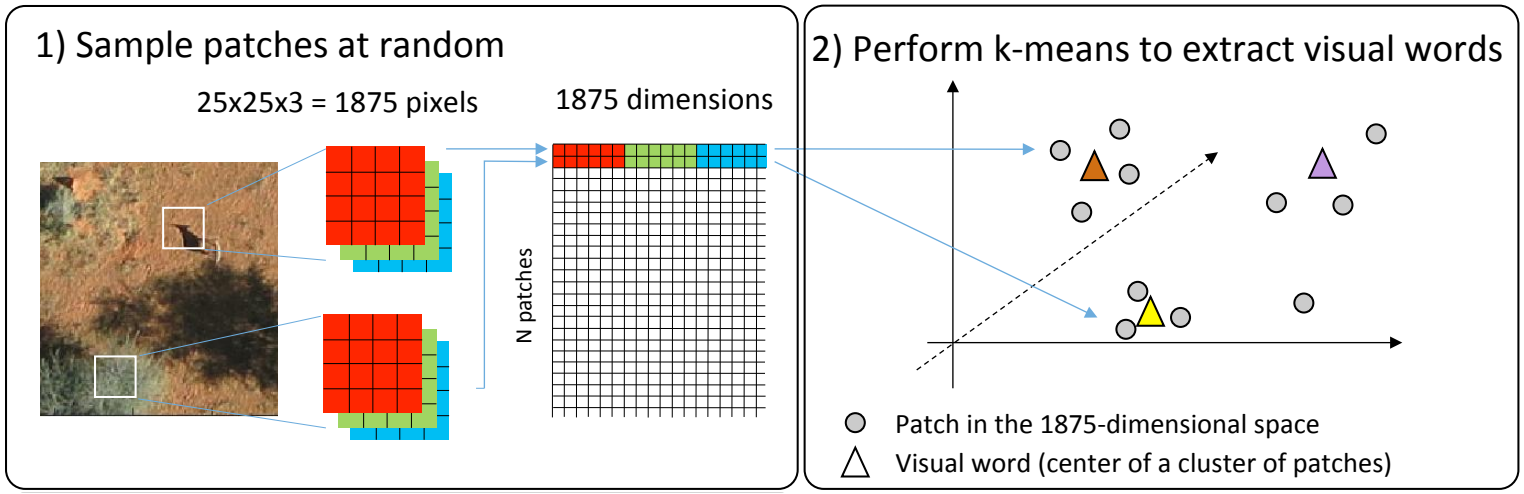
Iteration 3: convergence!





D. Tuia. ECEO

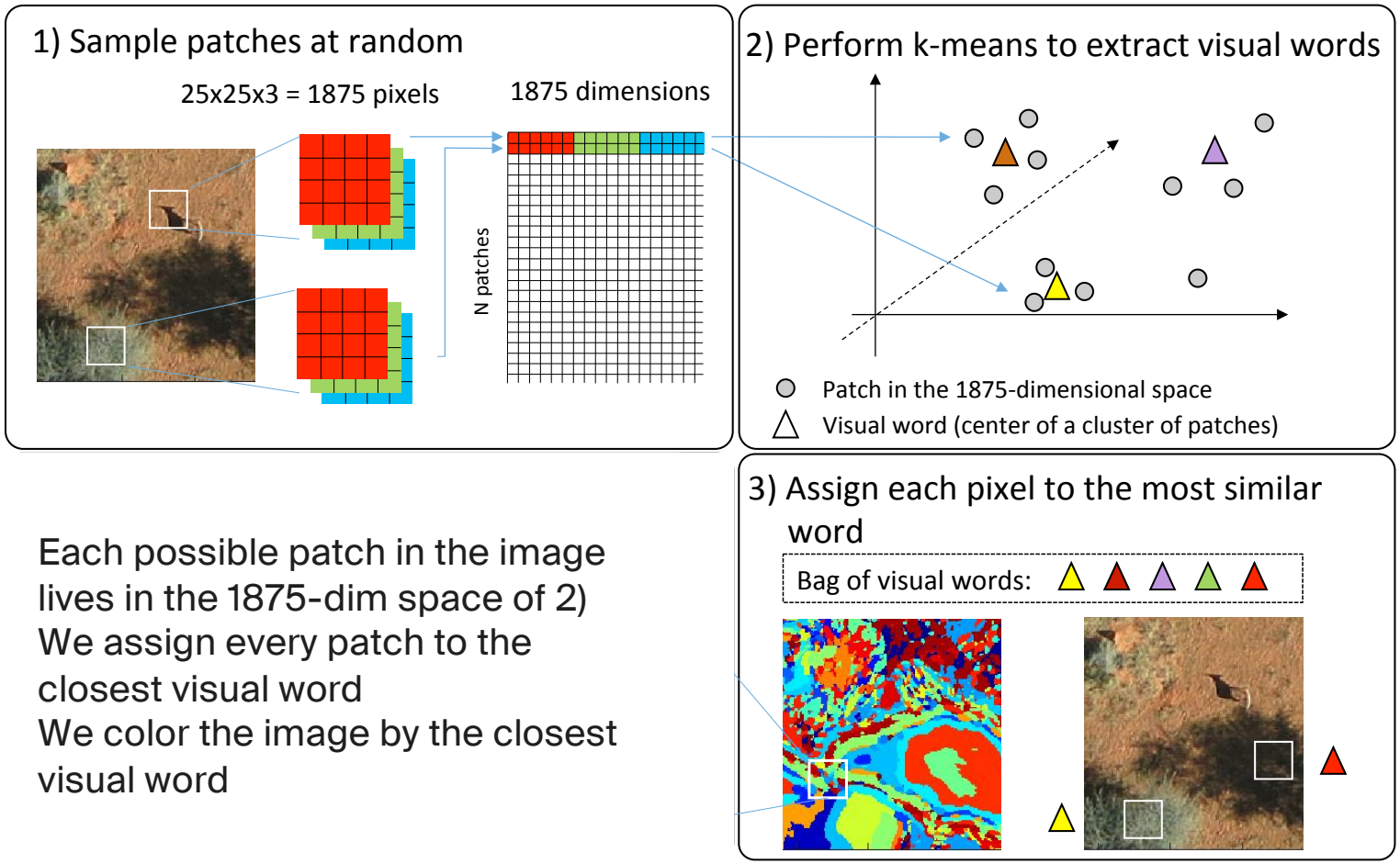
Building the BOW



- Run a k-means, i.e. cluster the patches into a number of representative types
- Each cluster center is a typical pattern seen in the images
- We call them **visual words**

N. Rey, M. Volpi, S. Joost, and D. Tuia. Detecting animals in african savanna with UAVs and the crowds. *Remote Sens. Environ.*, 200C:341–351, 2017.

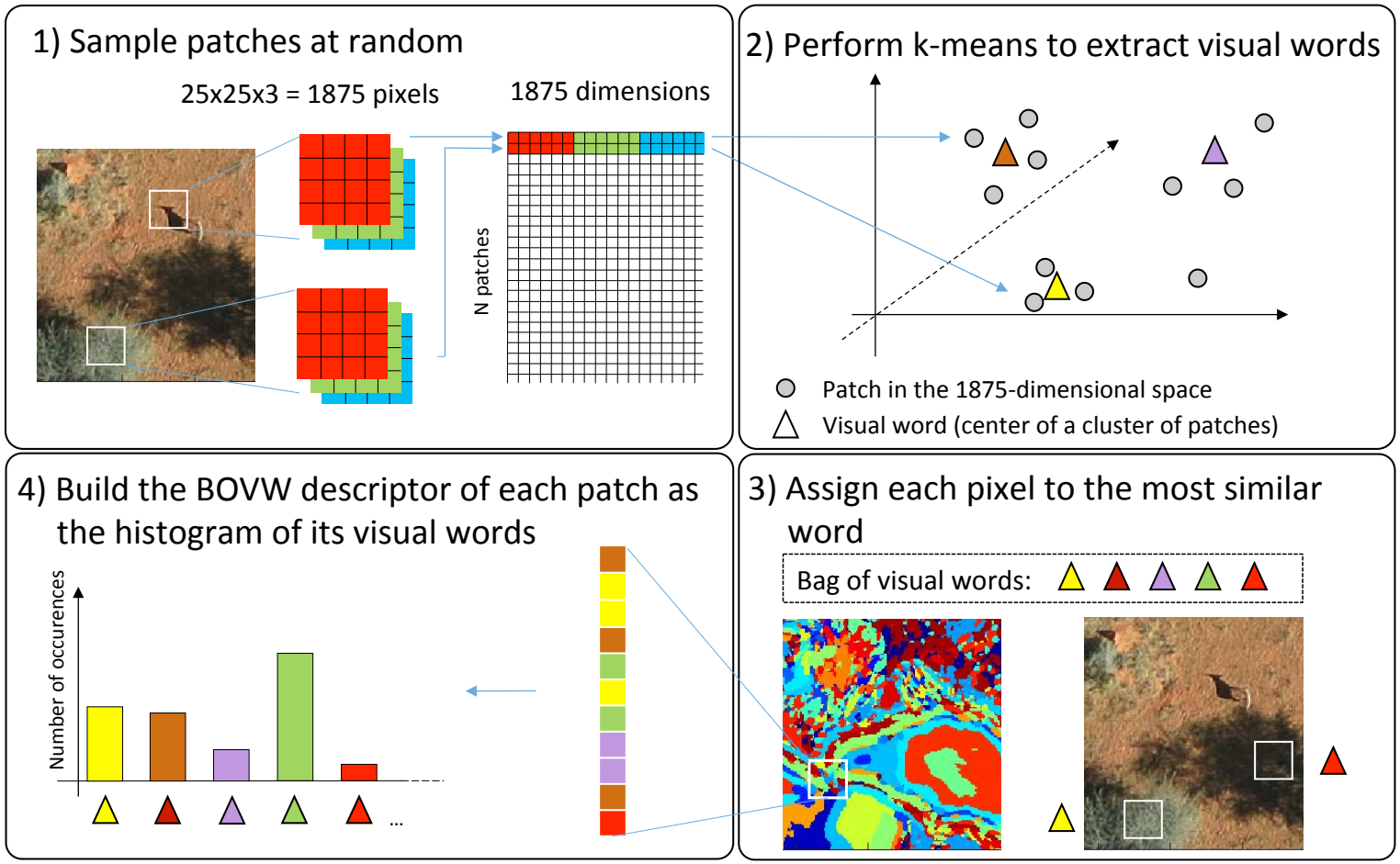
Building the BOW



- Each possible patch in the image lives in the 1875-dim space of 2)
- We assign every patch to the closest visual word
- We color the image by the closest visual word

N. Rey, M. Volpi, S. Joost, and D. Tuia. Detecting animals in african savanna with UAVs and the crowds. *Remote Sens. Environ.*, 200C:341–351, 2017.

Building the BOW

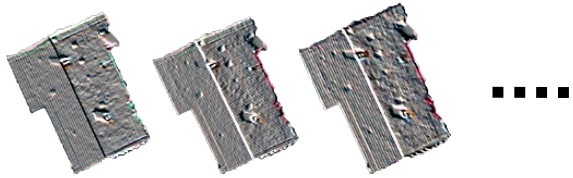


N. Rey, M. Volpi, S. Joost, and D. Tuia. Detecting animals in african savanna with UAVs and the crowds. *Remote Sens. Environ.*, 200C:341–351, 2017.

Summing up the BOW

Compute filters on all channels

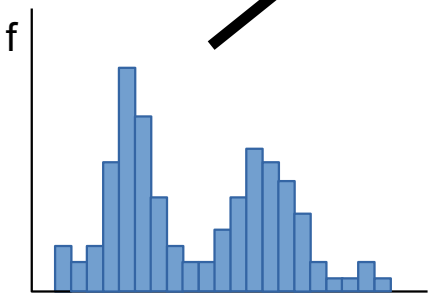
e.g. oriented filters, gradients, etc.

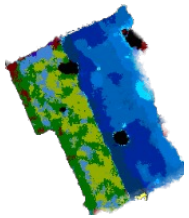


....

Visual word extraction

$$\mathbf{x}_i^{\text{bow}} = [f_i^1 \quad f_i^2 \quad \dots \quad f_i^k]$$



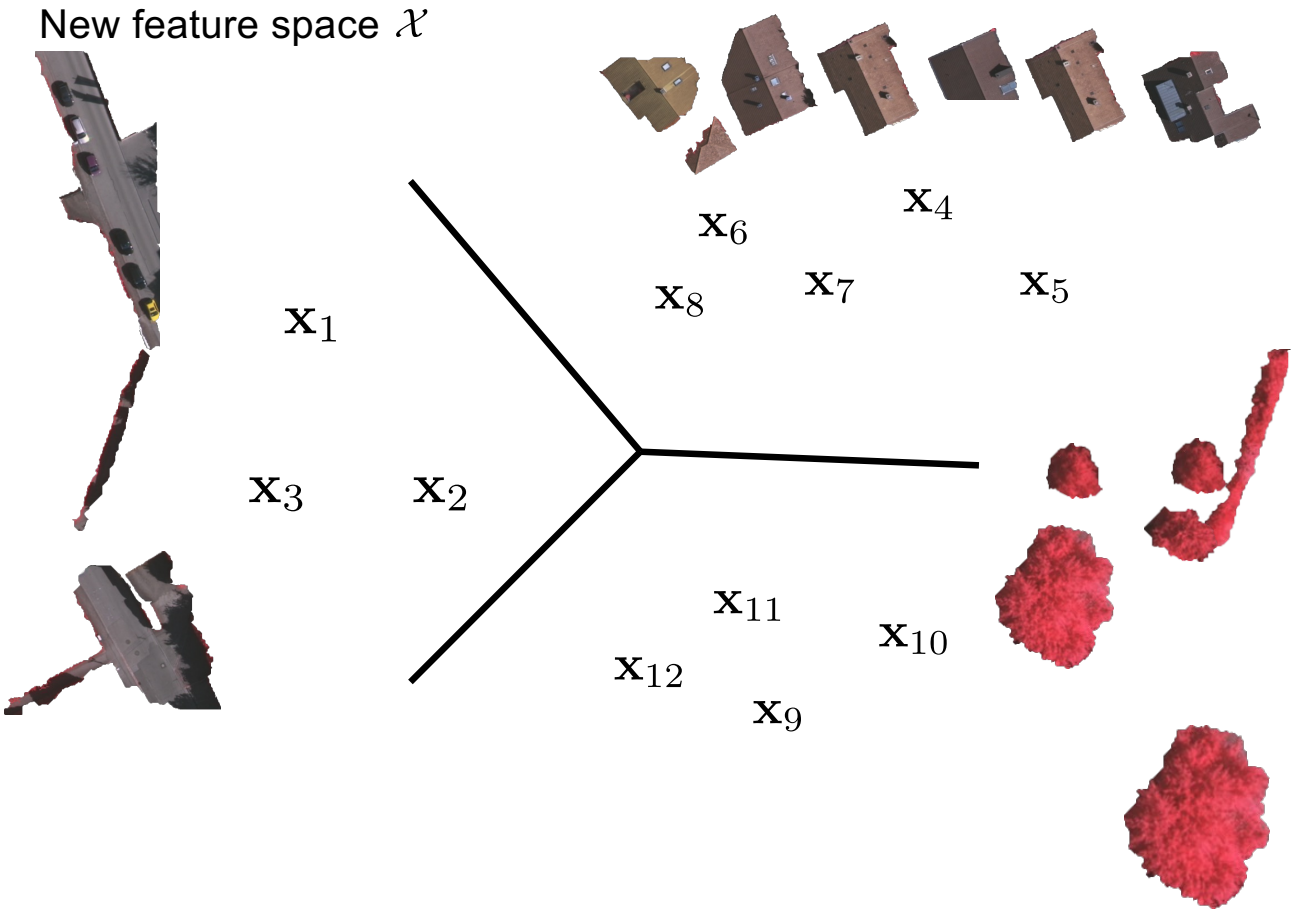


Word counts per object

(binning)

- Stack filter responses (*D*-dimensional image)
- Run *k*means clustering over *all* pixels with many centers (*k*)
- Assign to each pixel the label of the closes centroid

From colors to feature space



In summary

- The appearance of patches / regions / objects requires a careful extraction of information:
  - The **spatial support** must be as precise as possible
  - The **descriptors** must cover all the possible data variability
  - The descriptors must be related to the problem to be solved

    e.g. classifying

    - vegetated areas vs non-vegetated areas requires spectral information / spectral indexes;

    - buildings vs roads requires texture and “shape” information; etc.