Analisi di Immagini e Video (Computer Vision)

Giuseppe Manco

Outline

- Preliminari
- Classificazione
  - Esempio: regressione logistica
- Quali features?

Crediti

- Slides adattate da vari corsi e libri
  - Analisi di Immagini (F. Angiulli) – Unical
  - Intro to Computer Vision (J. Tompkin) – CS Brown Edu
  - Computer Vision (I. Gkioulekas) - CS CMU Edu
  - Cmputational Visual Recognition (V. Ordonez), CS Virgina Edu
  - Pattern Recognition and Machine Learning (C. Bishop, 2005)
  - Deep Learning (Bengio, Courville, Goodfellow, 2017)

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

- f di forma non nota
- x campionato da un dominio X e y campionato da un dominio Y
- Tipicamente, $x \in \mathbb{R}^n$

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

- f di forma non nota
- x campionato da un dominio X e y campionato da un dominio Y
- Tipicamente, $x \in \mathbb{R}^n$
- Supervised Learning
  - Apprendi f da $D \subset X \times Y$

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

- f di forma non nota
- x campionato da un dominio X e y campionato da un dominio Y
- Tipicamente, $x \in \mathbb{R}^n$
- Supervised Learning
  - Apprendi f da $D \subset X \times Y$
- Unsupervised Learning
  - Apprendi f da $D \subset X$

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

- f di forma non nota
- x campionato da un dominio X e y campionato da un dominio Y
- Tipicamente, $x \in \mathbb{R}^n$
- Supervised Learning
  - Apprendi f da $D \subset X \times Y$
- Semi-supervised Learning
  - Apprendi f da $D = (D_1, D_2)$ dove $D_1 \subset X \times Y$ e $D_2 \subset X$

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

- f di forma non nota
- x campionato da un dominio X e y campionato da un dominio Y
- Tipicamente, $x \in \mathbb{R}^n$

$$f(x) \equiv f(x;\theta) \equiv f_\theta(x)$$

- $\theta \in \Theta$ parametro scelto dal parameter space Θ (il linguaggio)

Inferenza vs. Learning

- Stima/Apprendimento: Selezionare
  - I parametri più appropriati
  - Una distribuzione sui parametri
  - Un insieme di distribuzioni
- Inferenza:
  - Predizioni
  - Statistiche
  - Valori attesi
  - Margini di un modello statistico

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

- f di forma non nota
- x campionato da un dominio X e y campionato da un dominio Y
- Tipicamente, $x \in \mathbb{R}^n$

$$f(x) \equiv f(x;\theta) \equiv f_\theta(x)$$

- $\theta \in \Theta$ parametro scelto dal parameter space Θ (il linguaggio)
- Cos'è y?

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

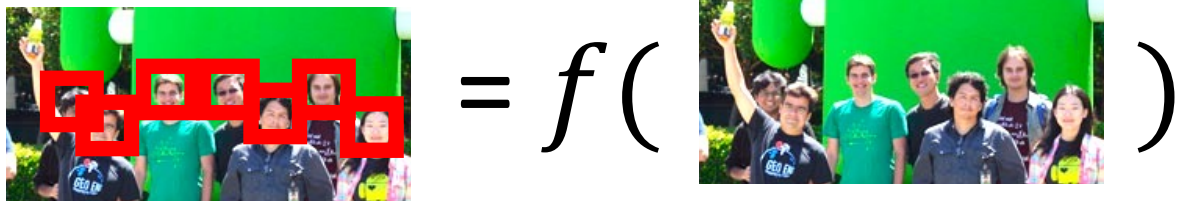
- Cos'è y?
- Una classe

cat $= f($  $)$

Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

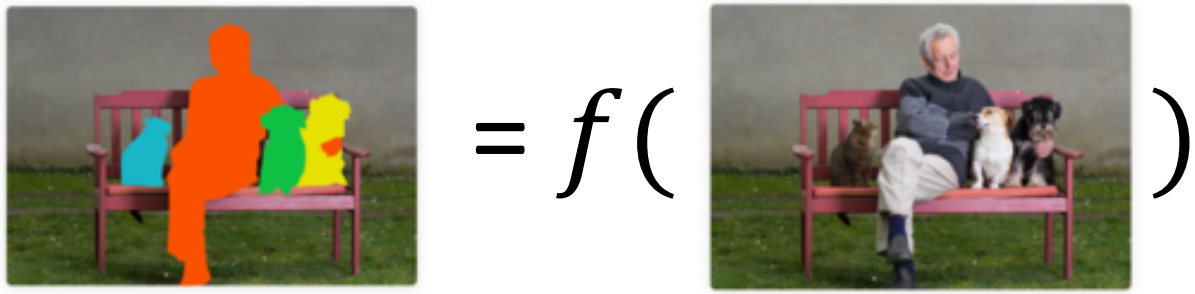
- Cos'è y?
- Un insieme finito di regioni



Supervised, unsupervised e semi-supervised learning

$$y = f(x)$$

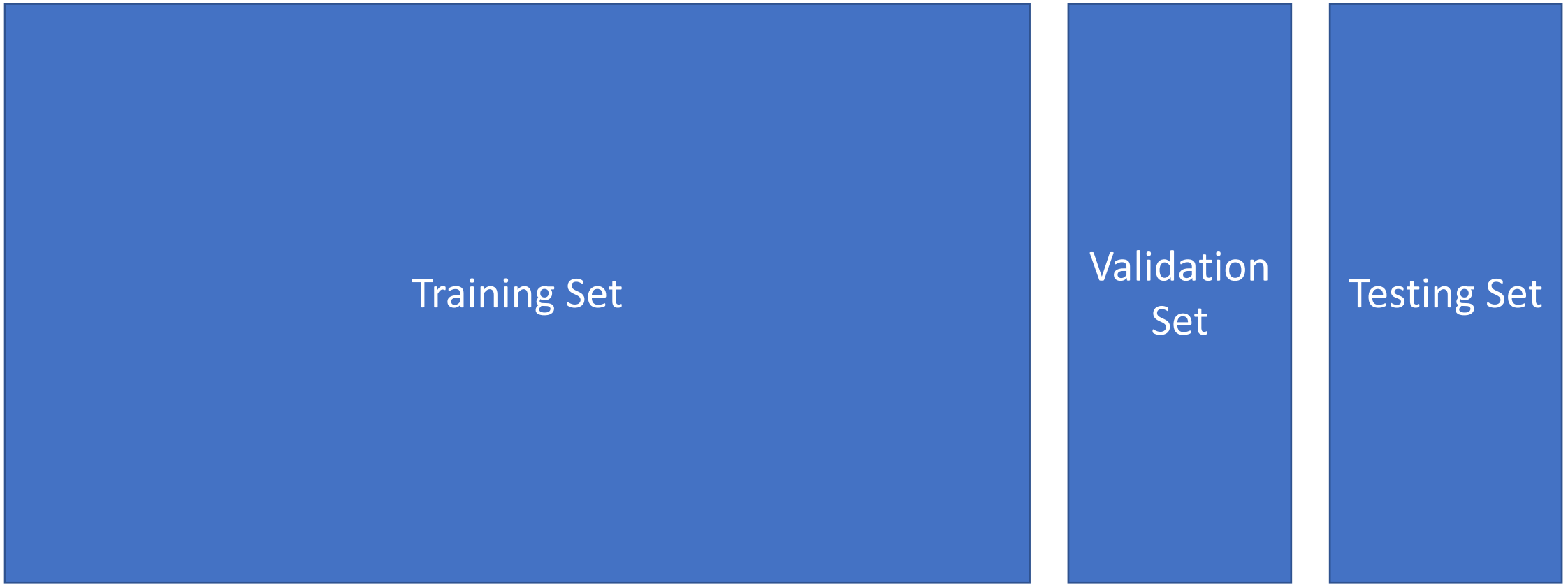
- Cos'è y?
- Segmenti



Inferenza vs. Learning

- Stima/Apprendimento: Selezionare
  - I parametri più approriati
  - Una distribuzione sui parametri
  - Un insieme di distribuzioni
- Inferenza:
  - Predizioni
  - Statistiche
  - Valori attesi
  - Margini di un modello statistico

Training, Validation, Test Sets



D

Training, Validation (Dev), Test Sets

D_{train}

Usato nella fase di learning

Training, Validation (Dev), Test Sets

Usato nella fase di valutazione

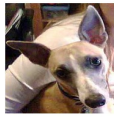
D_{test}

Classificazione

Classificazione

Training Set

cat



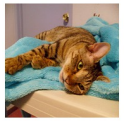
dog

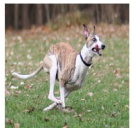
cat

.
.
.

bear

Test Set



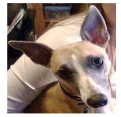


.
.
.

Classificazione

Training Set

$x_1 = [$  $]$ $y_1 = [$cat $]$

$x_2 = [$  $]$ $y_2 = [$dog $]$

$x_3 = [$  $]$ $y_3 = [$cat $]$

.
.
.

$x_n = [$  $]$ $y_n = [$bear $]$

Classificazione

Training Set

inputs

targets / ground truth

Predizioni

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $y_1 = 1$ $\hat{y}_1 = 1$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $y_2 = 2$ $\hat{y}_2 = 2$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $y_3 = 1$ $\hat{y}_3 = 2$

.
.
.

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $y_n = 3$ $\hat{y}_n = 1$

Dato il parametro θ possiamo calcolare $\hat{y}$

$$\hat{y} = f_\theta(x)$$

Problema:
Come trovare il parametro θ?
Soluzione:
Funzione di costo

$$loss = \sum_i Cost(y_i, \hat{y}_i)$$

Modello lineare di classificazione

- Caso semplice: classificazione binaria
  - $y \in \{0,1\}$
    - (o $y \in \{1,2\}$, $y \in \{-1,1\}$, …)

  - Il decision boundary tra due classi è un iperpiano nello spazio delle features
  - Le due regioni sono separate dall'iperpiano

$$w_1 x_1 + w_2 x_2 + \cdots w_m x_m = b$$

Regressione logistica

Training Data

inputs

targets /
labels /
ground truth

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $\quad y_1 = 1$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $\quad y_2 = 2$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $\quad y_3 = 1$

.
.
.

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $\quad y_n = 2$

Regressione logistica

Training Data

inputs

targets / labels / ground truth

predizioni

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $y_1 = 1$ $\hat{y}_1 = [0.80 \quad 0.10]$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $y_2 = 0$ $\hat{y}_2 = [0.30 \quad 0.70]$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $y_3 = 1$ $\hat{y}_3 = [0.40 \quad 0.60]$

.
.
.

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $y_n = 0$ $\hat{y}_n = [0.650 \quad 0.35]$

Regressione logistica

$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$ $\quad y_i = \quad 1$ $\quad \hat{y}_i = \quad [f_1 \quad f_2]$

$$g_1 = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + w_4 x_{i4} + b$$

Regressione logistica

$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$ $\quad y_i = 1 \quad$ $\hat{y}_i = [f_1 \quad f_2]$

$$logit = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + w_4 x_{i4} + b$$

- Cosa rappresenta $\hat{y}_i$?
- Cosa è $logit$?

Regressione logistica

$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$ $\quad y_i = \quad 1$ $\quad \hat{y}_i = \quad [f_1 \quad f_2]$

$$logit = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + w_4 x_{i4} + b$$

- Cosa rappresenta $\hat{y}_i$?
- Definiamo $f_\theta(x)$ come la funziona che fornisce le misure di probabilità per ogni classe

$$f_\theta(x) = [\Pr(y = 0|x, \theta), \Pr(y = 1|x, \theta)] \equiv [\Pr(neg|x, \theta), \Pr(pos|x, \theta)]$$

Regressione Logistica

- Preserva i classification boundaries lineari

- il decision boundary tra le classi pos e neg è determinato dalla seguente equazione:

$$\Pr(pos|x,\theta) = \Pr(neg|x,\theta)$$

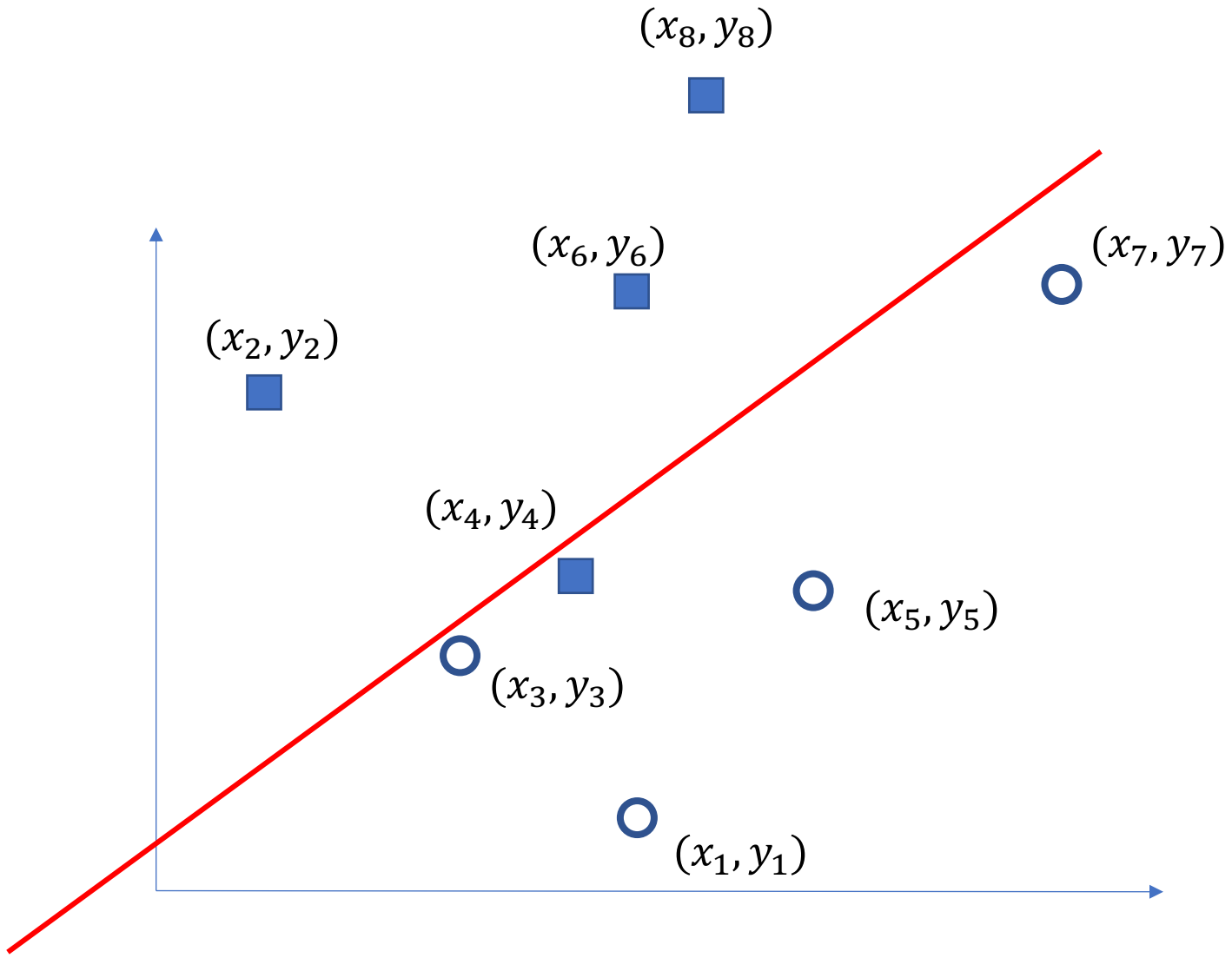
- rapportando e prendendo il logaritmo...

$$\log \frac{\Pr(pos|x,\theta)}{\Pr(neg|x,\theta)} = 0$$

- poiché il bordo deve essere lineare,

$$\log \frac{\Pr(pos|x,\theta)}{\Pr(neg|x,\theta)} = b + \sum_j w_j x_{i,j}$$

- Il rapporto è chiamato log odds, o anche logit
  - $\theta = \{b, w_1, \ldots, w_m\}$ è l'insieme dei parametri da apprendere


(x_8, y_8)
(x_6, y_6)
(x_7, y_7)
(x_2, y_2)
(x_4, y_4)
(x_5, y_5)
(x_3, y_3)
(x_1, y_1)

- dalla precedente, le probabilità a posteriori diventano:

$$\Pr(pos|x,\theta) = \frac{exp\left(b + \sum_j w_j x_{i,j}\right)}{1 + exp\left(b + \sum_j w_j x_{i,j}\right)}$$

$$\Pr(neg|x,\theta) = \frac{1}{1 + exp\left(b + \sum_j w_j x_{i,j}\right)}$$

- Le probabilità sommano a 1

32

- Introduciamo le funzioni
  - Logistica

$$\sigma(x) = \frac{1}{1 + exp(-x)}$$

  - logit

$$\text{logit}(\text{x}; \theta) = b + \sum_j w_j x_{i,j}$$

- Riscriviamo le formule

$$\Pr(pos|x, \theta) = \sigma(logit(x; \theta))$$

$$\Pr(neg|x, \theta) = 1 - \sigma(logit(x; \theta))$$

Model learning

- Obiettivo
  - trovare i parametri $\{b, w_1, \dots, w_m\}$ che massimizzano la verosimiglianza sul training set

$$\mathcal{L}(\mathrm{D}, \theta) = \prod_{i=1}^{n} \Pr(y_i | x_i, \theta)$$

  - Equivalentemente, minimizziamo

$$nll(D, \theta) = -\sum_{i=1}^{n} \log \Pr(y_i | x_i, \theta)$$

- Se $y_i = 1$ abbiamo

$$\begin{aligned}\log \Pr(y_i|x_i,\theta) &= \log \Pr(pos|x_i,\theta) \\ &= 1 \cdot \log \Pr(pos|x_i,\theta) \\ &= y_i \cdot \log \Pr(pos|x_i,\theta)\end{aligned}$$

- analogamente, se $y_i = 0$

$$\begin{aligned}\log \Pr(y_i|x_i,\theta) &= \log(1 - \Pr(pos|x_i,\theta)) \\ &= 1 \cdot \log (1 - \Pr(pos|x_i,\theta)) \\ &= (1 - y_i) \cdot \log \Pr(pos|x_i,\theta)\end{aligned}$$

- Poiché $y_i = 1$ oppure $y_i = 0$,

$$\log \Pr(y_i|x_i,\theta) = \ y_i \cdot \log \Pr(pos|x_i,\theta) + \ (1 - y_i) \cdot \log(1 - \Pr(pos|x_i,\theta))$$

- La verosimiglianza...

$$nll(D,\theta) = -\sum_{i=1}^{n} \log \Pr(y_i|x,\theta)$$

$$= -\sum_{i=1}^{n} \{\, y_i \cdot \log \Pr(pos|x,\theta) + \ (1-y_i) \cdot \log(1 - \Pr(pos|x,\theta))\}$$

$$= -\sum_{i=1}^{n} \{\, y_i \cdot \log \hat{y}_i + \ (1-y_i) \cdot \log(1-\hat{y}_i)\}$$

$$= -\sum_{i=1}^{n} Cost(y_i, \hat{y}_i)$$

- Dove
  - $\hat{y}_i = \Pr(\text{pos}|\text{x},\theta)$
  - $Cost(y_i, \hat{y}_i) = \ y_i \cdot \log \hat{y}_i + \ (1-y_i) \cdot \log(1-\hat{y}_i)$

Che succede se abbiamo più classi?

Training Data

inputs

targets / labels / ground truth

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $y_1 = 1$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $y_2 = 2$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $y_3 = 1$

.
.
.

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $y_n = 3$

Che succede se abbiamo più classi?

Training Data

inputs

targets / labels / ground truth

Predizioni

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $y_1 = [1 \quad 0 \quad 0]$ $\hat{y}_1 = [0.85 \quad 0.10 \quad 0.05]$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $y_2 = [0 \quad 1 \quad 0]$ $\hat{y}_2 = [0.20 \quad 0.70 \quad 0.10]$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $y_3 = [1 \quad 0 \quad 0]$ $\hat{y}_3 = [0.40 \quad 0.45 \quad 0.15]$

.
.
.

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $y_n = [0 \quad 0 \quad 1]$ $\hat{y}_n = [0.40 \quad 0.25 \quad 0.35]$

Che succede se abbiamo più classi?

- Un logit per ogni classe

$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$ $\quad y_i = [1 \quad 0 \quad 0]$ $\quad \hat{y}_i = [f_c \quad f_d \quad f_b]$

$$g_c = w_{c1}x_{i1} + w_{c2}x_{i2} + w_{c3}x_{i3} + w_{c4}x_{i4} + b_c$$

$$g_d = w_{d1}x_{i1} + w_{d2}x_{i2} + w_{d3}x_{i3} + w_{d4}x_{i4} + b_d$$

$$g_b = w_{b1}x_{i1} + w_{b2}x_{i2} + w_{b3}x_{i3} + w_{b4}x_{i4} + b_b$$

$$f_c = e^{g_c}/(e^{g_c}+e^{g_d} + e^{g_b})$$

$$f_d = e^{g_d}/(e^{g_c}+e^{g_d} + e^{g_b})$$

$$f_b = e^{g_b}/(e^{g_c}+e^{g_d} + e^{g_b})$$

Che succede se abbiamo più classi?

- La loss può essere adattata

$$nll(D, \theta) = -\sum_{i=1}^{n} Cost(y_i, \hat{y}_i)$$

$$Cost(y_i, \hat{y}_i) = \sum_{j=1}^{k} y_{i,j} \cdot \log \hat{y}_{i,j}$$

$$\hat{y}_{i,j} = \Pr\left(y_{i,j} = 1 \middle| x_i, \theta\right) = softmax_j(x_i; \theta)$$

$$softmax_j(x_i; \theta) = \frac{\exp(logit_j(x_i, \theta))}{\sum_{h=1}^{k} \exp(logit_h(x_i, \theta))}$$

Ottimizziamo la funzione di costo

$$nll(D, \theta) = -\sum_{i=1}^{n} Cost(y_i, \hat{y}_i)$$

- Due strade:
  - Gradient-Descent (primo ordine)
  - Newton-Raphson (secondo ordine)

Gradiente discendente

$$l(\theta) \equiv nll(D, \theta) = -\sum_{i=1}^{n} Cost(y_i, \hat{y}_i)$$

- Dato λ (learning rate) e N (numero di epoche)

```
Inizializza θ_0 in maniera random
for e in range(N)
        θ_{e+1} ← θ_e − λ ∂l(θ_e)/∂θ
```

Idea di fondo

- Per valori opportuni di λ produce una sequenza $l(\theta_0) \geq l(\theta_1) \ldots \geq l(\theta_e)$
- Perché funziona?
  - Approssimazione in serie di Taylor al primo ordine

$$l(\theta) \approx l(\theta_e) + (\theta - \theta_e)^T \frac{\partial l(\theta_e)}{\partial \theta}$$

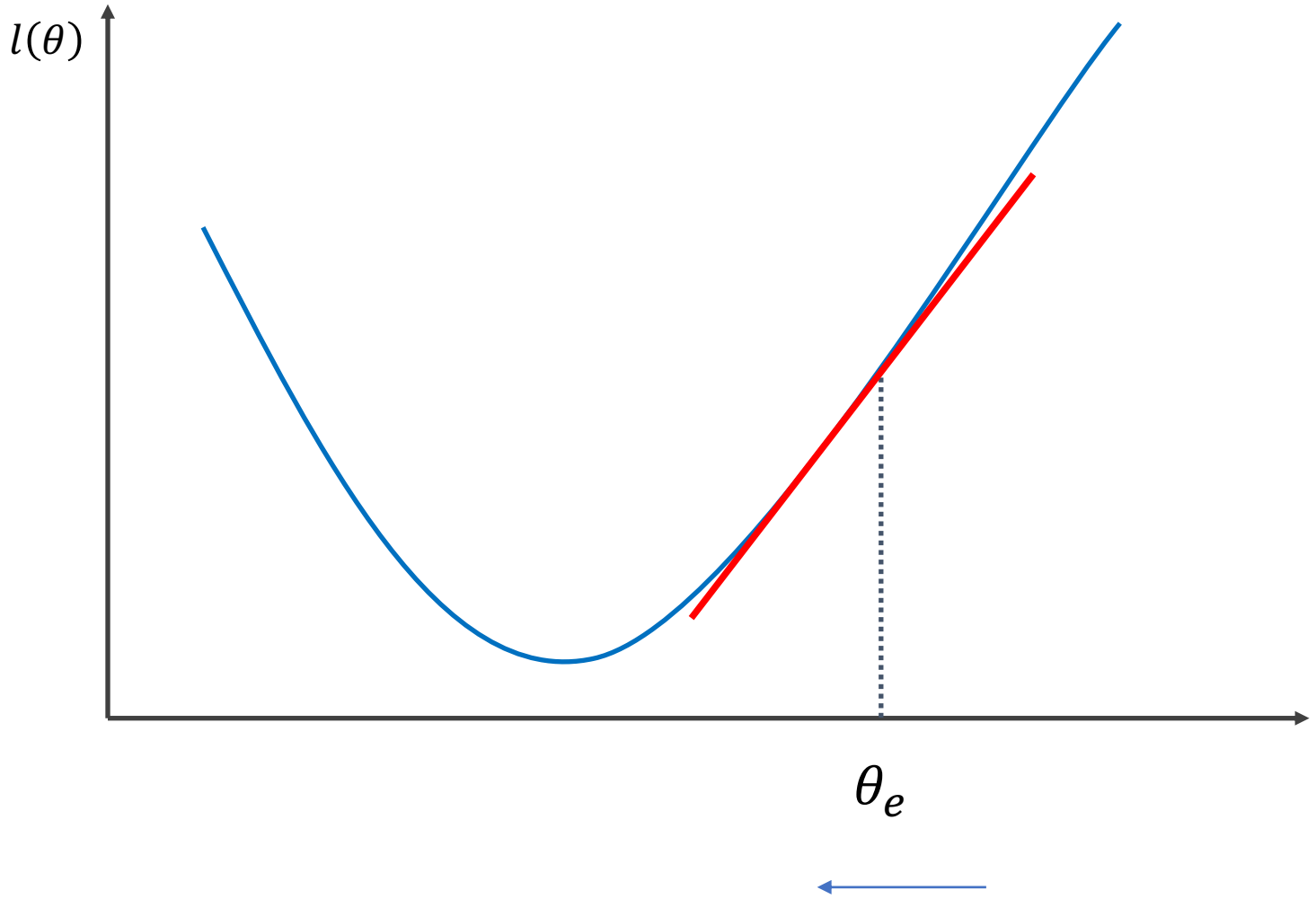
  - Calcolato sul punto θ_{e+1} e assumendo l'update

$$l(\theta_{e+1}) \approx l(\theta_e) + (\theta_{e+1} - \theta_e)^T \frac{\partial l(\theta_e)}{\partial \theta}$$

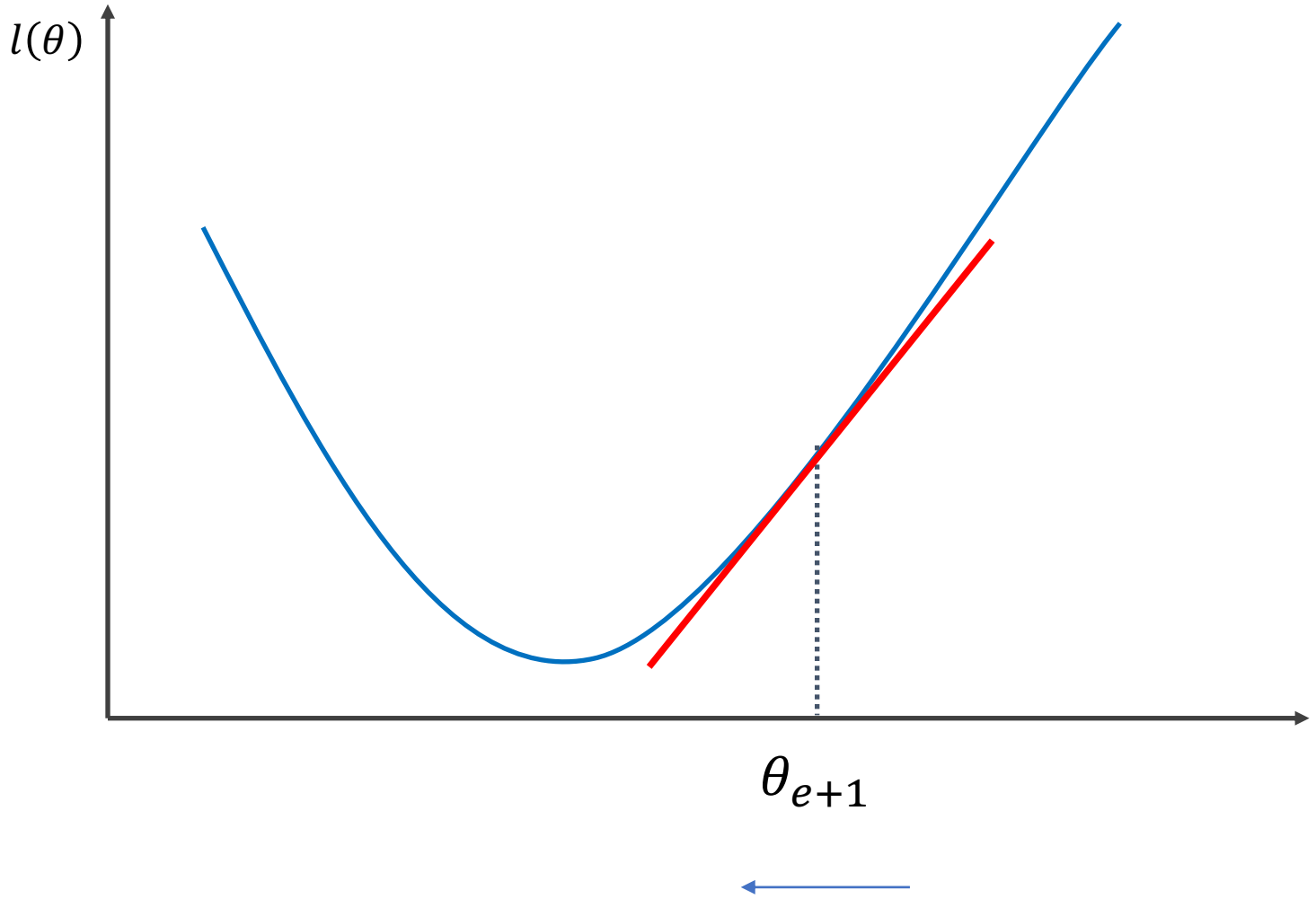
$$= l(\theta_e) - \lambda \cdot \left\| \frac{\partial l(\theta_e)}{\partial \theta} \right\|^2$$

  - Il termine $\left\| \frac{\partial l(\theta_e)}{\partial \theta} \right\|^2$ è sempre positivo

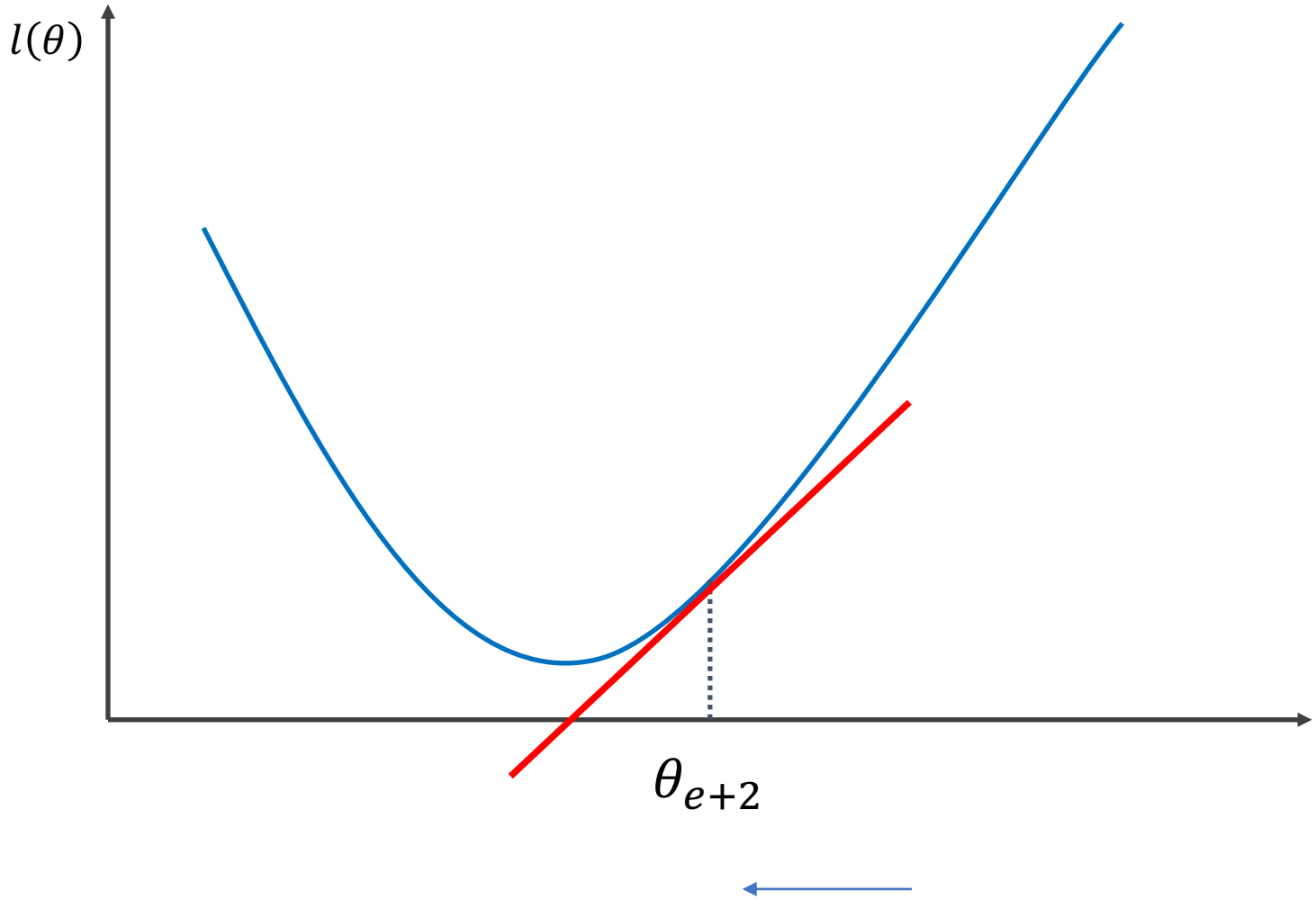
Idea di base



Idea



Idea



Gradiente discendente

costoso

$$l(\theta) \equiv nll(D, \theta) = -\sum_{i=1}^{n} Cost(y_i, \hat{y}_i)$$

- Dato λ (learning rate) e N (numero di epoche)

```
Inizializza θ_0 in maniera random
for e in range(N)
        θ_{e+1} ← θ_e − λ ∂l(θ_e)/∂θ
```

Minibatch (stochastic) GD

$$l_B(\theta) = -\sum_{i \in B} Cost(y_i, \hat{y}_i)$$

- Dato λ (learning rate) e N (numero di epoche), M (numero di batches)

```
Inizializza θ_0 in maniera random
for e in range(N)
    for b in range(M)
        θ_{e+1} ← θ_e − (λ/B) ∂l_{B_b}(θ_e)/∂θ
```

Learning

- Gradient Descent

$$\theta_{e+1} \leftarrow \theta_e - \lambda \frac{\partial l(\theta_e)}{\partial \theta}$$

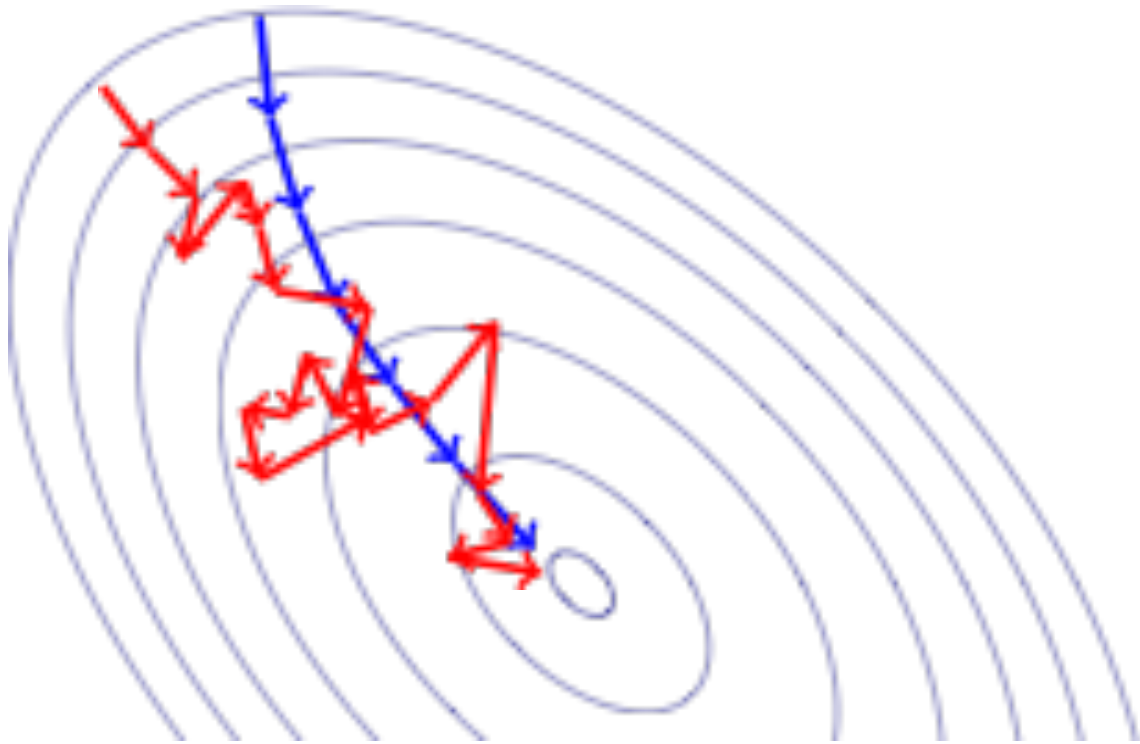
- Stochastic GD

$$\theta_{e+1} \leftarrow \theta_e - \frac{\lambda}{N} \frac{\partial l_i(\theta_e)}{\partial \theta}$$

- Mini-batch SGD

$$\theta_{e+1} \leftarrow \theta_e - \frac{\lambda}{B} \sum_{i \in B} \frac{\partial l_i(\theta_e)}{\partial \theta}$$

$$l_i(\theta) = Cost(y_i, \hat{y}_i)$$



More to come later …

- Regularization
- Momentum updates
- Hinge Loss, Least Squares Loss, Logistic Regression Loss…

Riassumendo

$$\hat{y}_i = [f_c \quad f_d \quad f_b]$$

Estraiamo le features

$$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$$

Calcoliamo I logits

$$g_c = w_{c1}x_{i1} + w_{c2}x_{i2} + w_{c3}x_{i3} + w_{c4}x_{i4} + b_c$$

$$g_d = w_{d1}x_{i1} + w_{d2}x_{i2} + w_{d3}x_{i3} + w_{d4}x_{i4} + b_d$$

$$g_b = w_{b1}x_{i1} + w_{b2}x_{i2} + w_{b3}x_{i3} + w_{b4}x_{i4} + b_b$$

$$f_c = e^{g_c}/(e^{g_c}+e^{g_d} + e^{g_b})$$

$$f_d = e^{g_d}/(e^{g_c}+e^{g_d} + e^{g_b})$$

$$f_b = e^{g_b}/(e^{g_c}+e^{g_d} + e^{g_b})$$

Otteniamo le predizioni

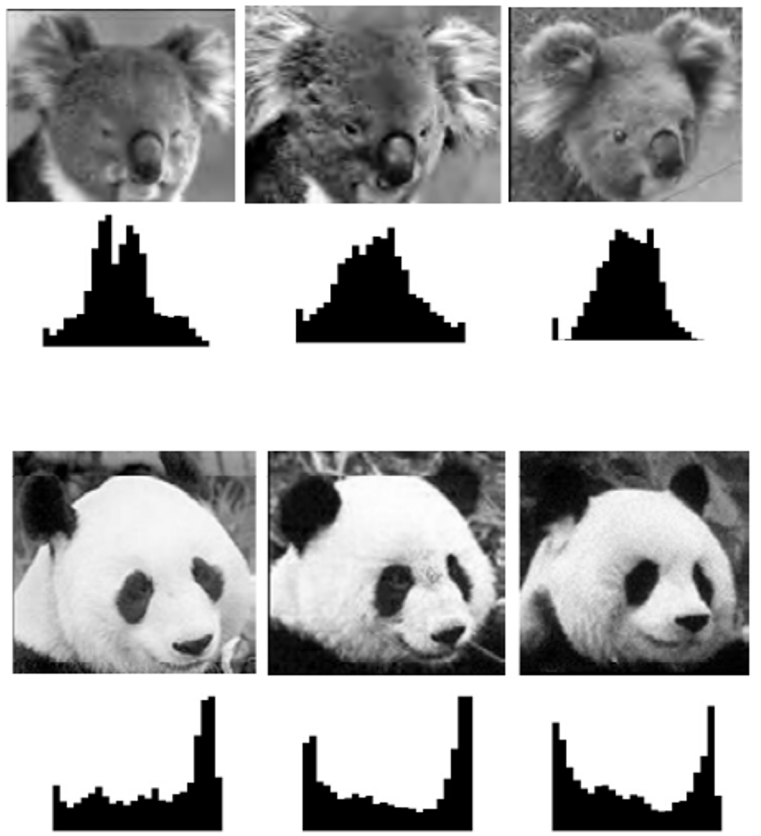
Quali features?

Quali features?

- Global features
  - Colori
  - Istogrammi
- Local features
  - Edges
  - Corners
  - Histogram of Oriented Gradients (HOG)
  - Haar Cascades
  - Scale-Invariant Feature Transform (SIFT)
  - Speeded Up Robust Feature (SURF)
  - …

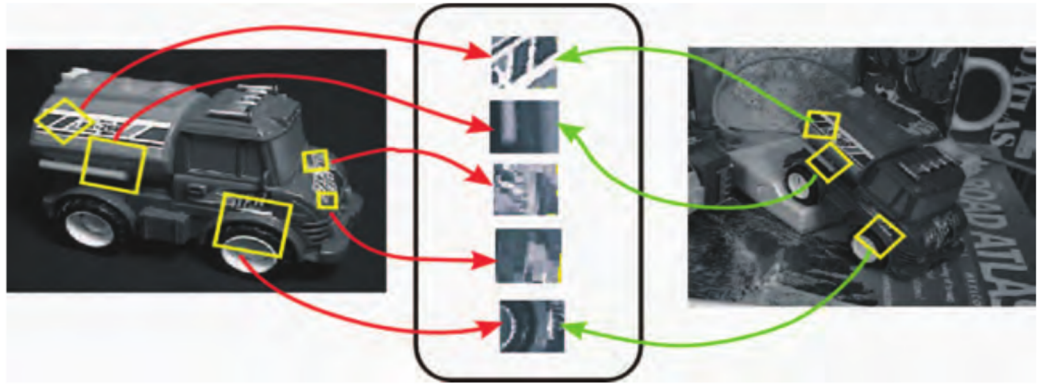
Global Features

- Rappresentazione olistica dell'immagine
- Esempio: istogramma delle intensità dell'immagine
- Immagini simili hanno istogramma simile, ma in generale non è vero il viceversa
- Permettono di rappresentare la struttura globale dell'oggetto, ma non di gestire l'occlusione, il cambiamento di punto di vista e le altre variabilità



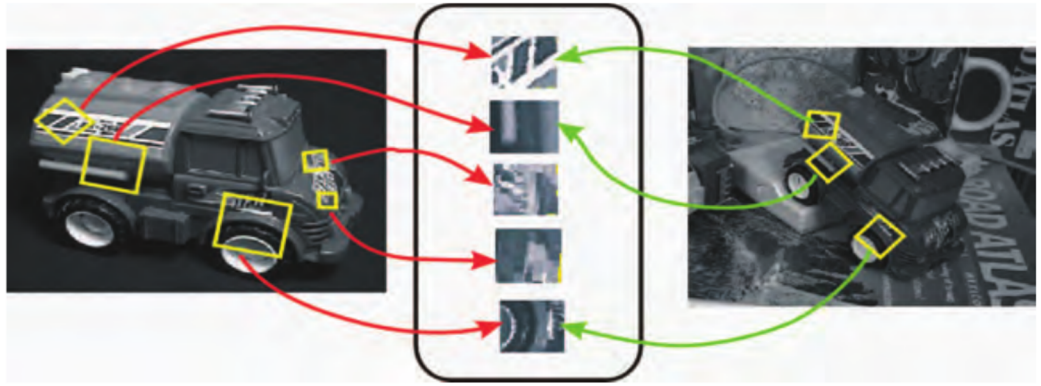
Local Features

- Le local features rappresentano un *insieme sparso* di misurazioni locali che catturano l'essenza delle strutture all'interno dell'immagine
- Diverse proprietà richieste: precise, distintive, invarianti, numerose



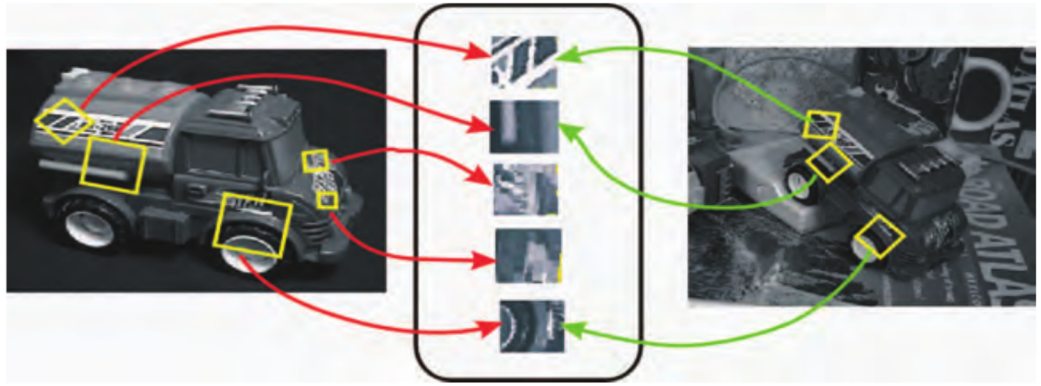
Local Features

- Il processo di estrazione dev'essere *ripetibile* e *preciso*, in modo che le stesse features siano restituite da due immagini che riguardano lo stesso oggetto



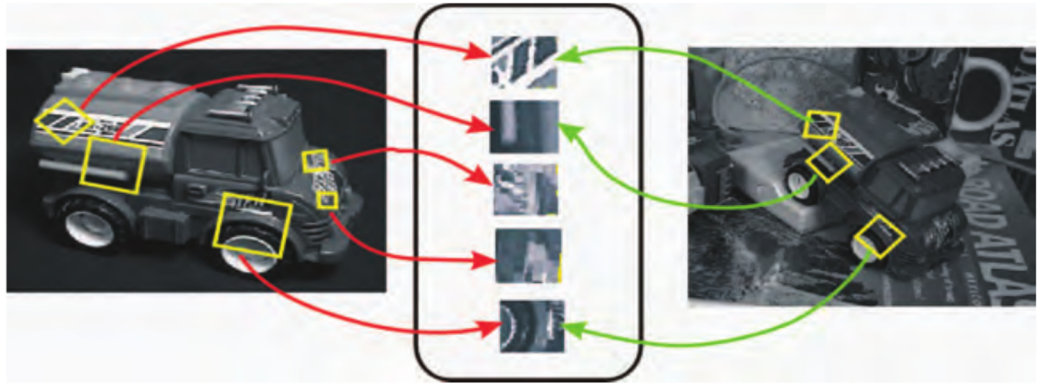
Local Features

- Le features devono essere *distintive*, in modo che le diverse strutture presenti all'interno dell'immagine possano essere discriminate



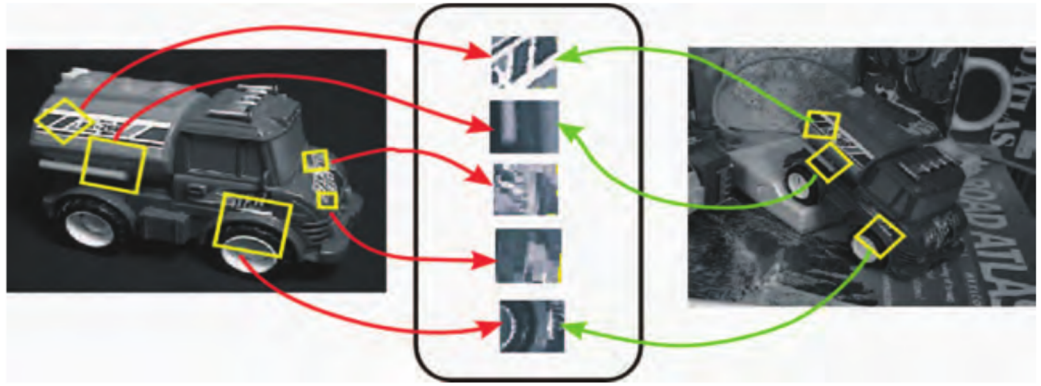
Local Features

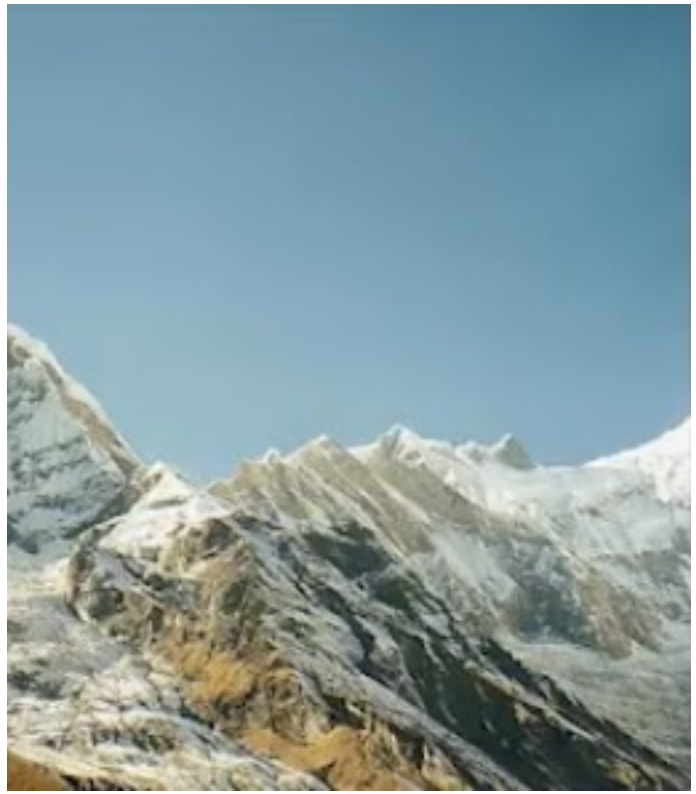
- Le local features dovrebbero essere *invarianti* rispetto a diverse trasformazioni applicate all'immagine
  - traslazioni, rotazioni, scalatura, …

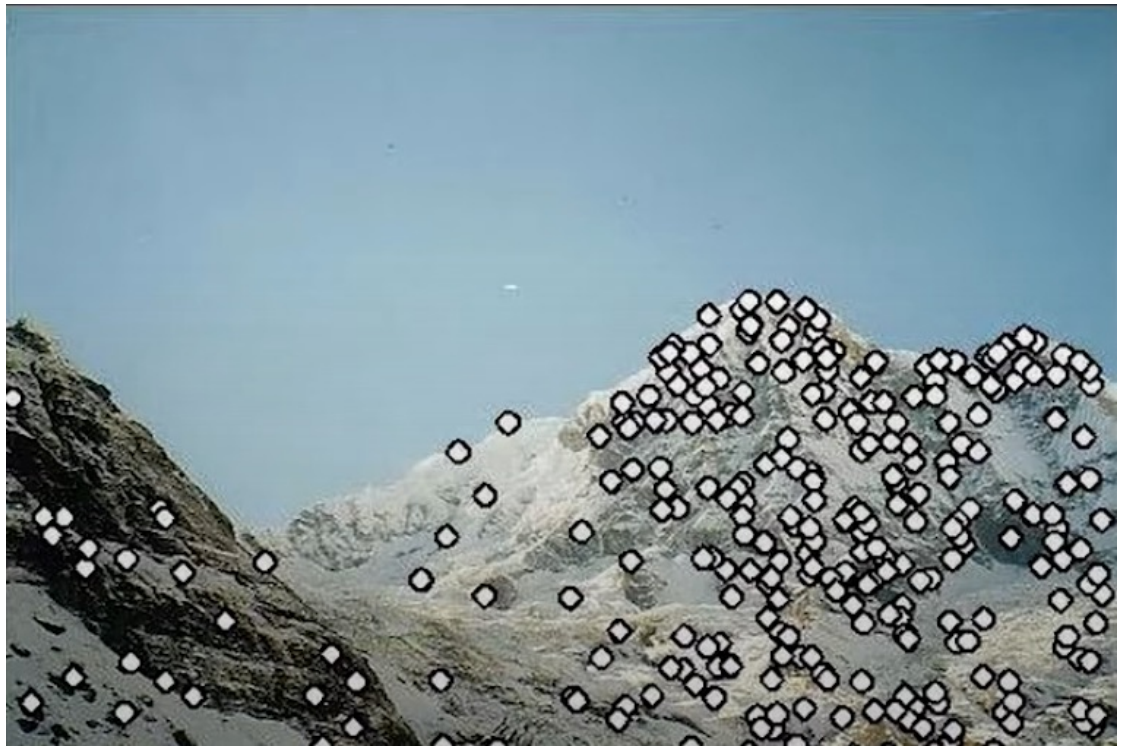


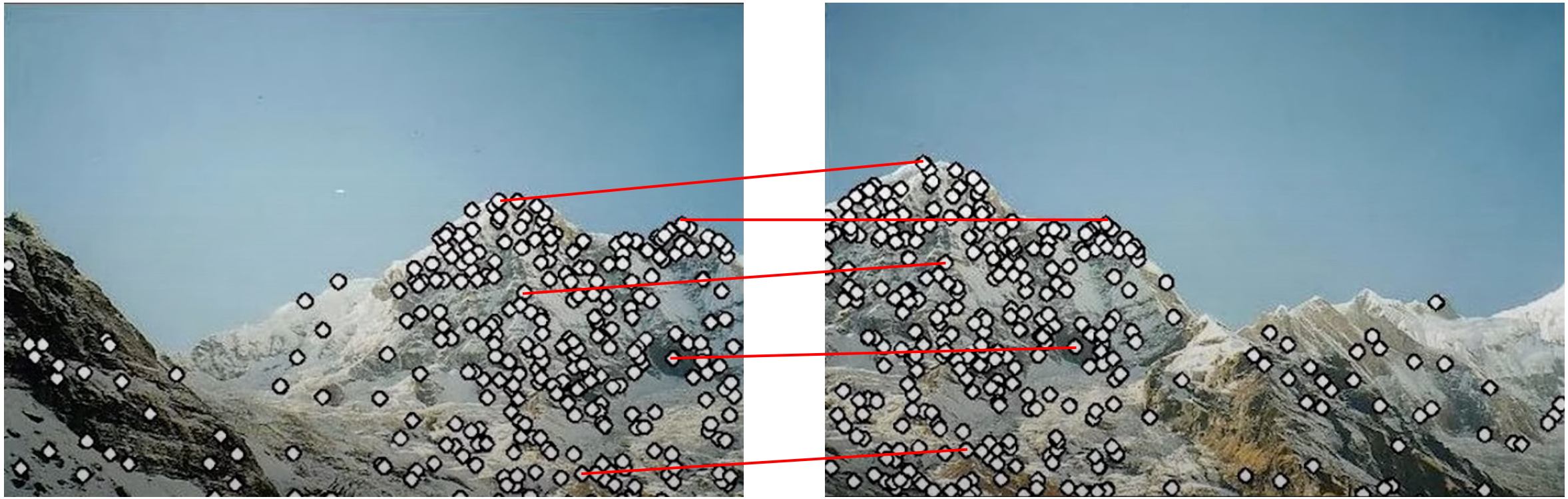
Local Features

- Le features estratte devono essere abbastanza *numerose* da assicurare una buona copertura dell'immagine
  - ad esempio, per permettere il riconoscimento di oggetti anche parzialmente occlusi



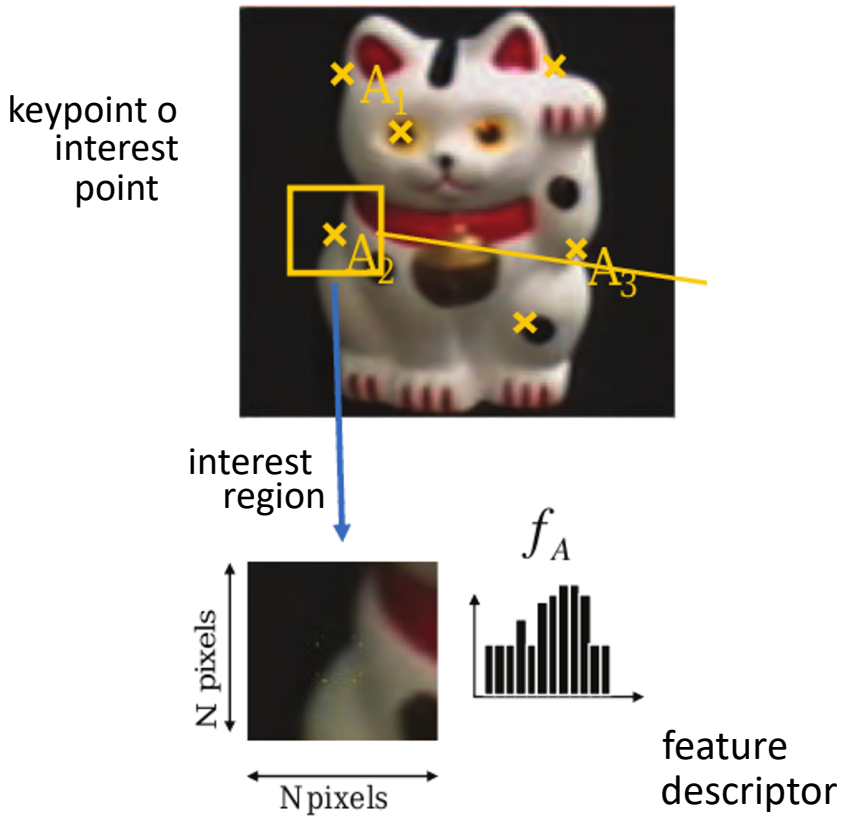






Feature Extraction

- Passi del processo di *Feature Extraction*:
  1. Individua un insieme di *keypoints* o *interest points* (*Keypoint Localization* o *Feature Detection*)
  2. Determina una *interest region* intorno ad ogni keypoint
  3. Estrai e *normalizza* il contenuto della regione e rappresentalo mediante un *feature descriptor*



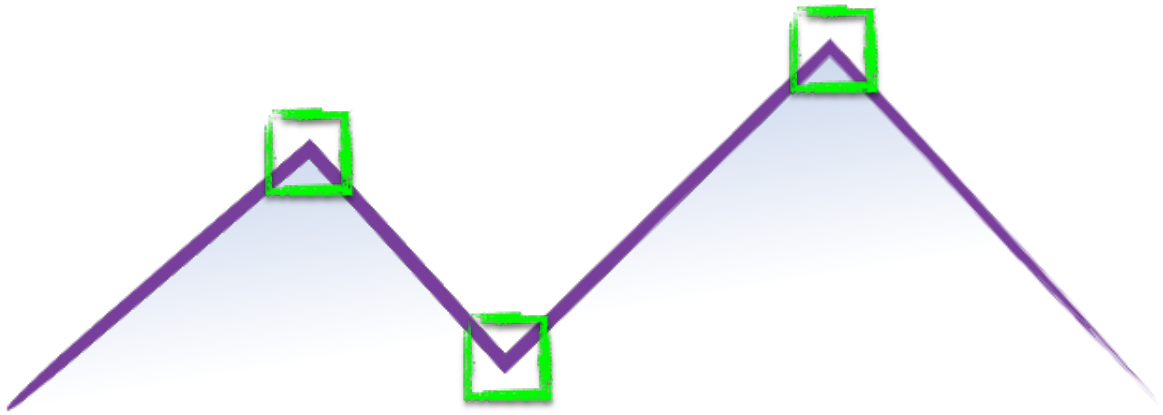
Keypoints, features and feature descriptors

- Si estrae un insieme di features da ogni immagine del training set
  - Ogni feature ha una rappresentazione multidimensionale
- Si trovano i K raggruppamenti di tutte le features
  - Ad esempio, usando il K-Means
  - Ogni feature è associata ad un cluster
- Per ogni immagine, si calcola l'istogramma delle features sui K gruppi
  - Ogni feature contribuisce al cluster a cui è associata
- L'istogramma (normalizzato) rappresenta la rappresentazione K-dimensionale dell'immagine

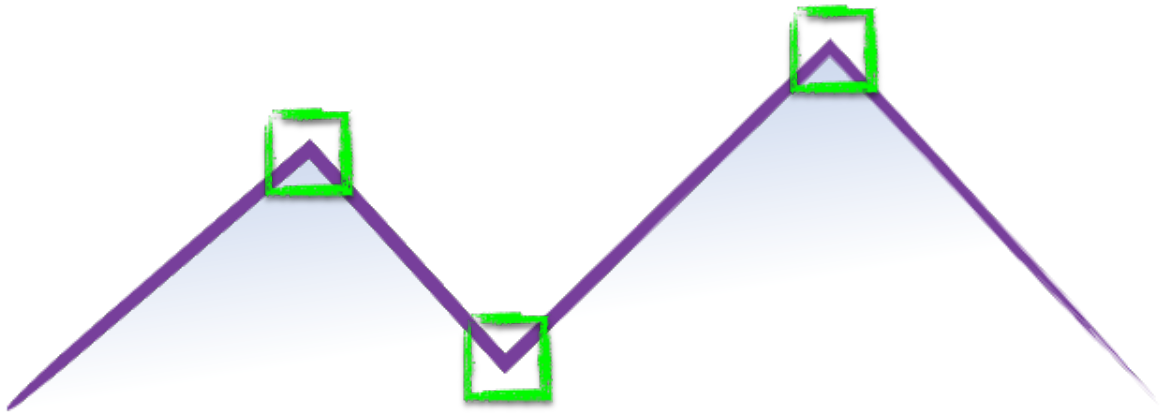
Keypoint Localization

- I **keypoint** (o **interest point**) sono punti *distintivi* che possono essere localizzati anche in presenza di variazioni nell'immagine, quali traslazioni o rotazioni, cambiamenti di punti di vista e presenza di rumore
- Escludiamo dai punti candidati ad essere distintivi quelli per cui non si manifestano cambiamenti in qualche direzione:
  - Non possiamo determinare il moto dei punti appartenenti ad una regione uniforme
  - Per i punti appartenenti ad una linea, possiamo misurare solo il moto perpedicolarmente alla linea

Come identificare un angolo?



Come identificare un angolo?

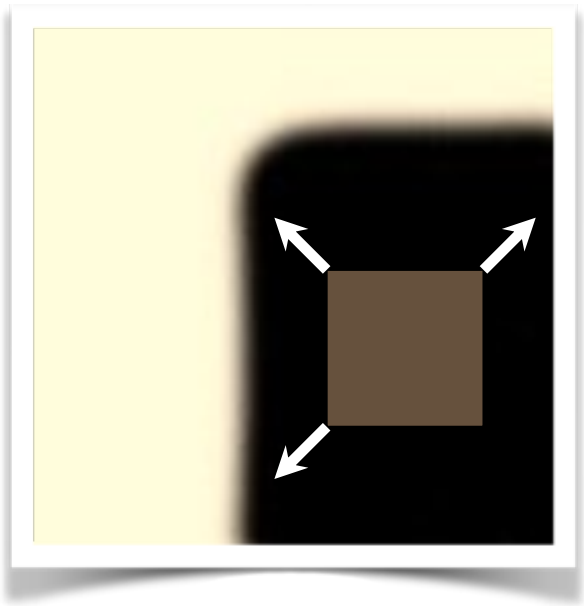


Si analizza una finestra

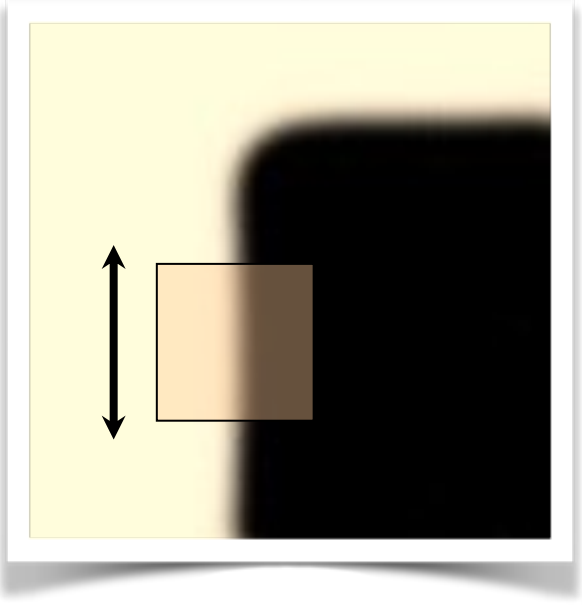
Muovendo la finestra si osserva un significativo cambiamento di intensità

Si analizza una finestra

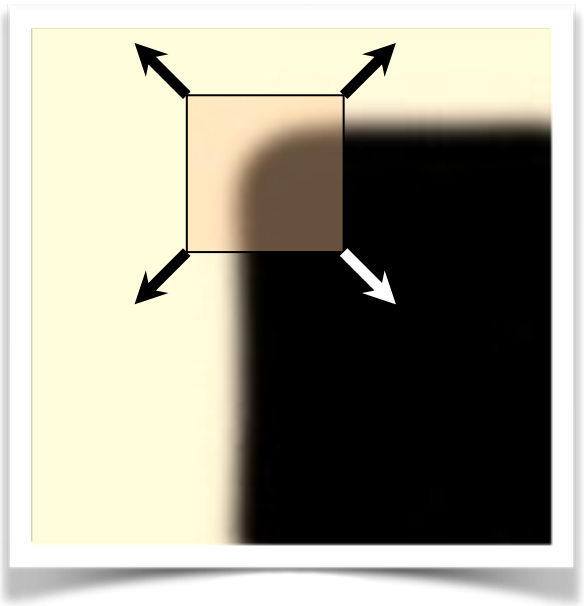
Muovendo la finestra si osserva un significativo cambio di intensità



"flat" nessun cambiamento

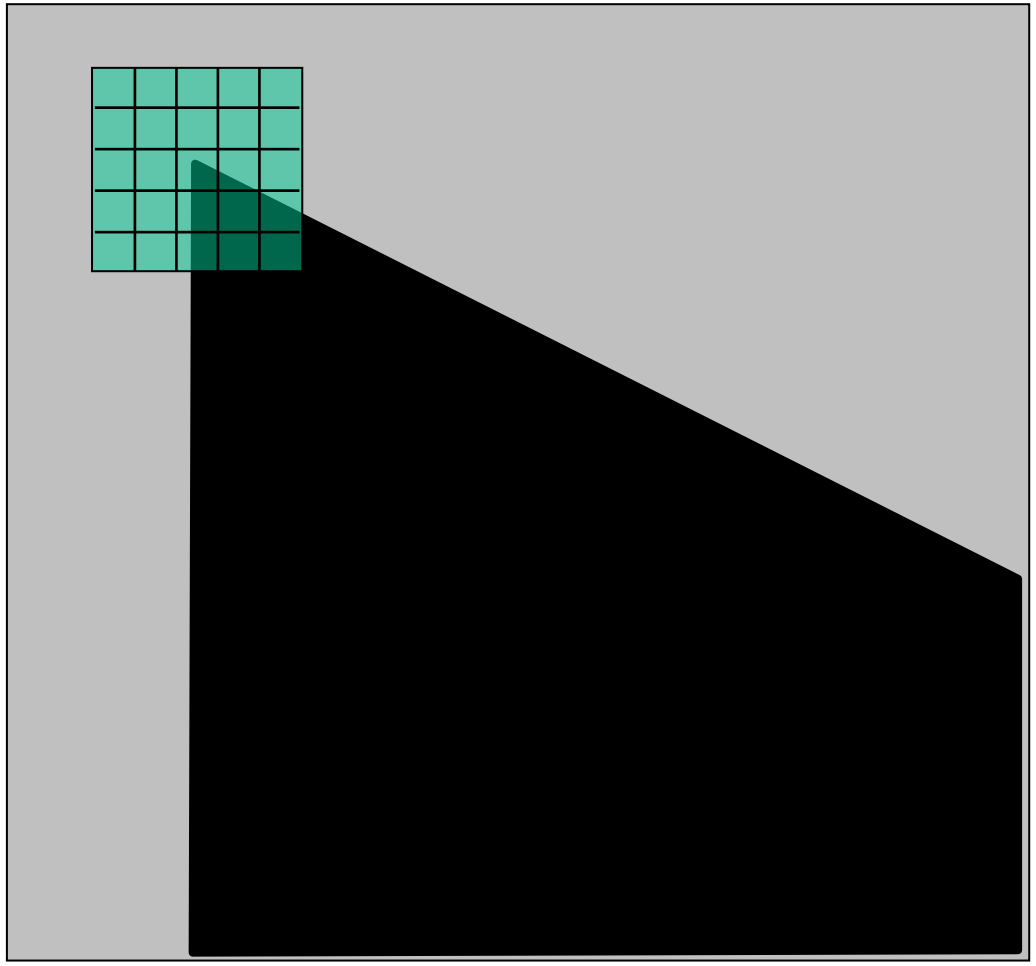


"edge": cambiamento lungo una direzione

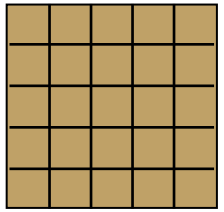


"corner": Cambiamento lungo entrambe le direzioni

Il gradiente su un punto

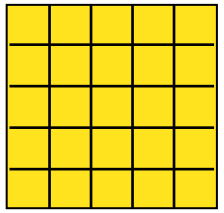


$$I_x = \frac{\partial I}{\partial x}$$



array of y gradients

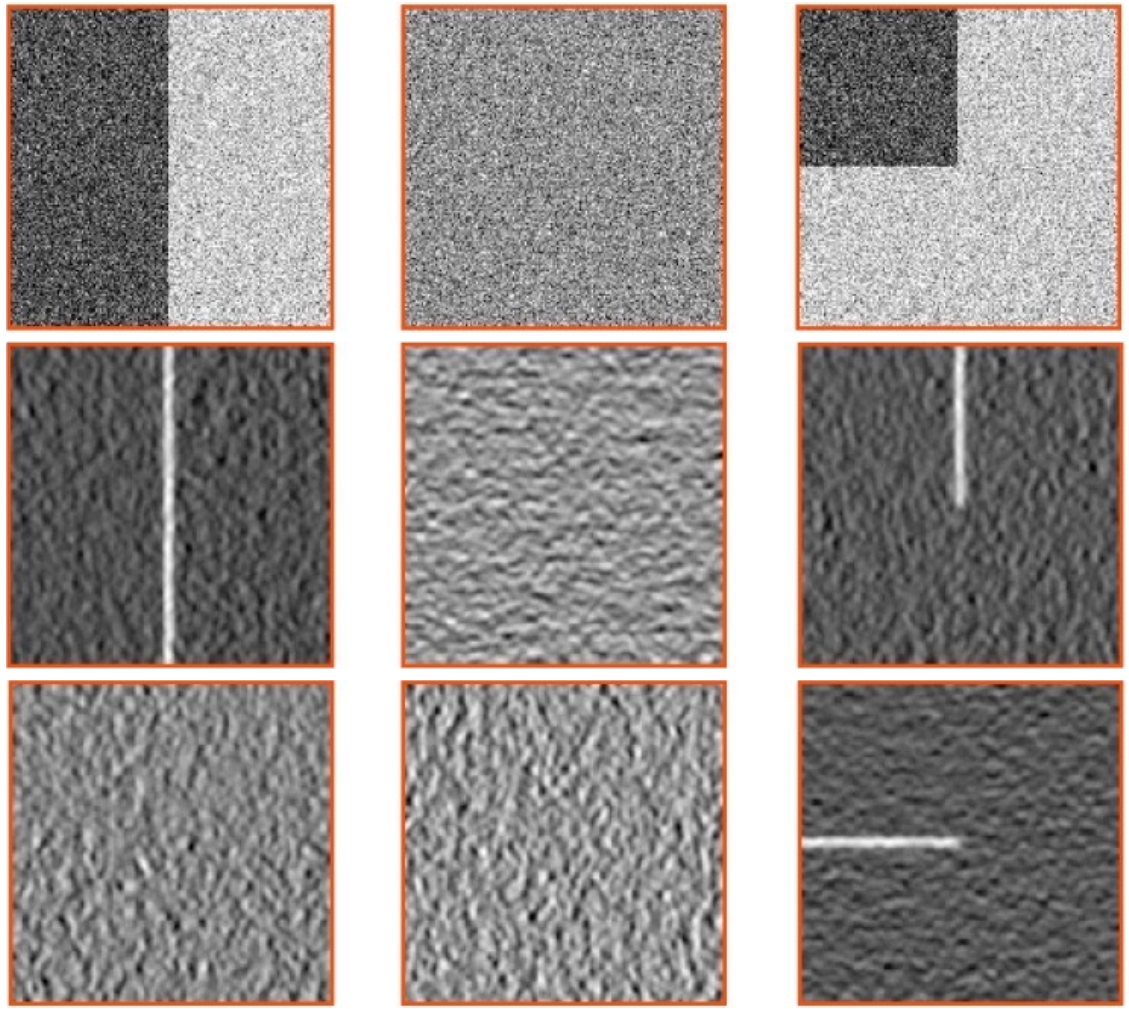
$$I_y = \frac{\partial I}{\partial y}$$



Immagine

Gradiente su X

Gradiente su Y



La matematica del corner detection

Cambio di intensità intorno allo shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

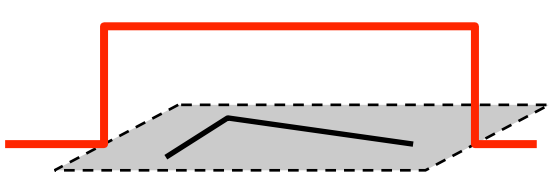
Error function

Window function

Shifted intensity

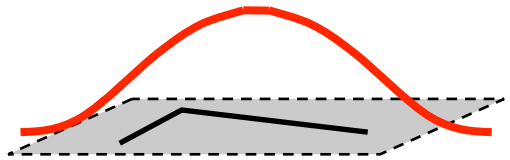
Intensity

Window function $w(x,y)$ =



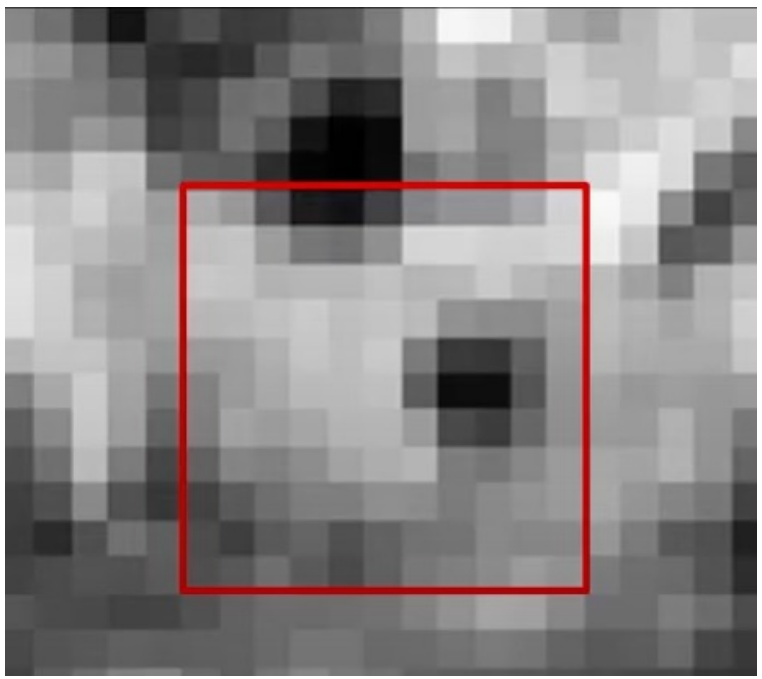
1 in window, 0 outside

o

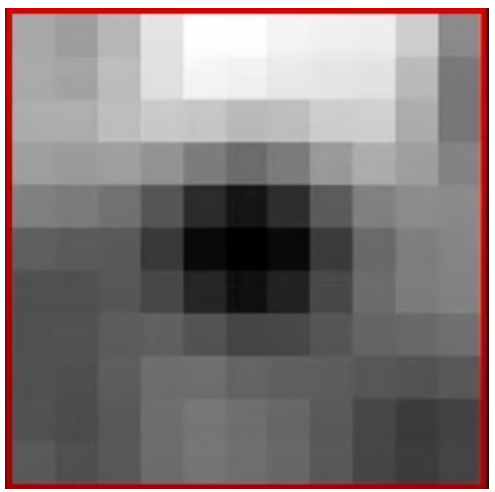


Gaussian

$I(x, y)$



$\mathrm{E}(u, v)$



La matematica del corner detection

Cambio di intensità intorno allo shift **[*u*,*v*]:**

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

- Come cambia $E(u,v)$ per piccoli cambiamenti di u, v?
  - Approssimazione di Taylor al secondo ordine!

$$F(\delta x) \approx F(0) + \delta x \frac{dF(0)}{dx} + \frac{1}{2}\,\delta x^2 \frac{d^2F(0)}{dx^2}$$

La matematica del corner detection

Cambio di intensità intorno allo shift **[*u*,*v*]:**

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

$$F(\delta x) \approx F(0) + \delta x \frac{dF(0)}{dx} + \frac{1}{2}\,\delta x^2 \frac{d^2F(0)}{dx^2}$$

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{vu}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

La matematica del corner detection

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

$$E(u,v) \approx E(0,0) + \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{vu}(0,0) & E_{vv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$E_u(0,0) = E(0,0) = E_v(0,0) = 0$$

$$E_{uu}(0,0) = \sum_{xy} 2w(x,y) I_x(x,y) I_x(x,y)$$

$$E_{uv}(0,0) = \sum_{xy} 2w(x,y) I_x(x,y) I_y(x,y)$$

La matematica del corner detection

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

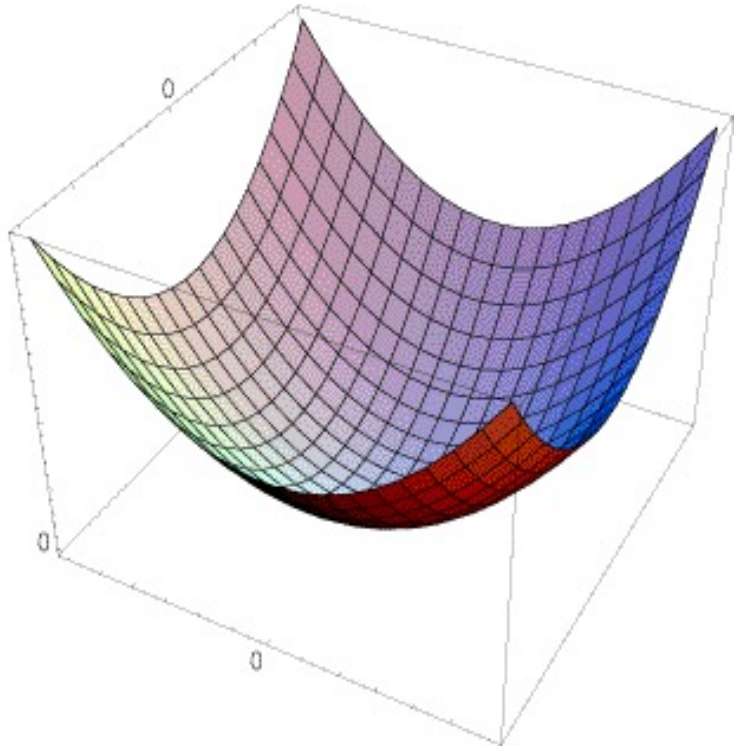
$$E(u,v) \approx \frac{1}{2}\begin{bmatrix} u & v \end{bmatrix}\begin{bmatrix} \sum_{xy} 2w(x,y)I_x^2(x,y) & \sum_{xy} 2w(x,y)I_x(x,y)I_y(x,y) \\ \sum_{xy} 2w(x,y)I_x(x,y)I_y(x,y) & \sum_{xy} 2w(x,y)I_y^2(x,y) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

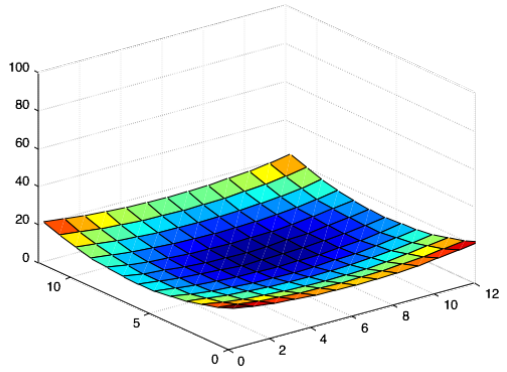
La matematica del corner detection

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

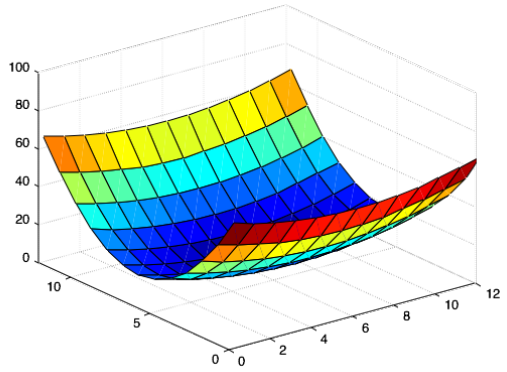
$$E(u,v) \approx [u \quad v] M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_{xy} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

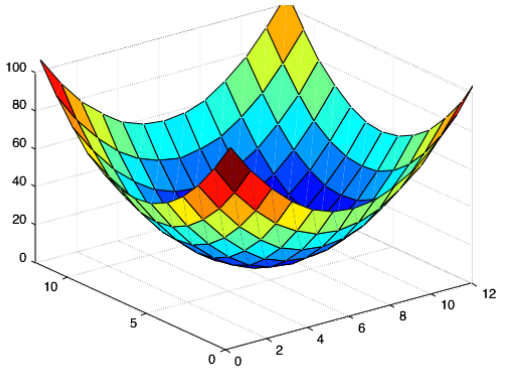




flat



edge
‘line’



corner
‘dot’

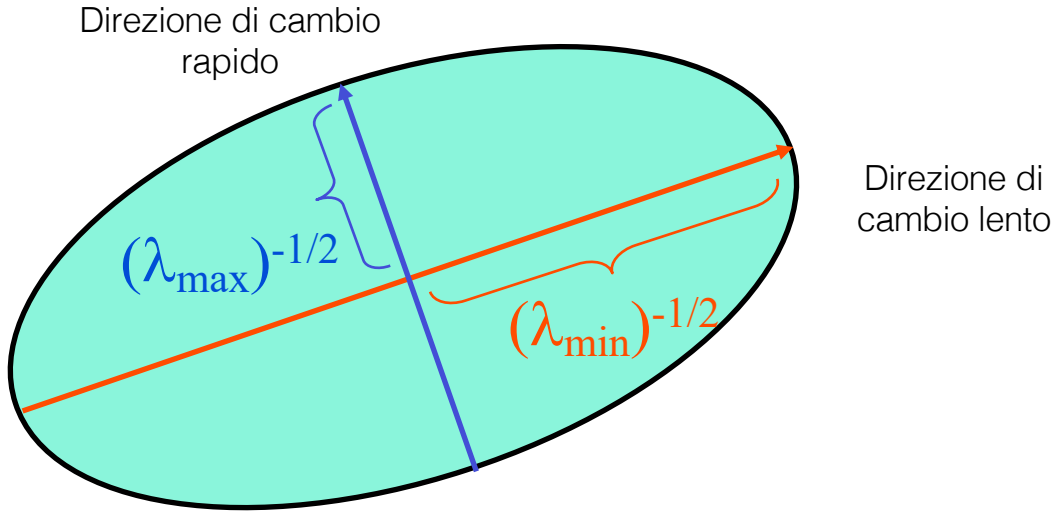
Visualizzazione

Poiché M è simmetrica,

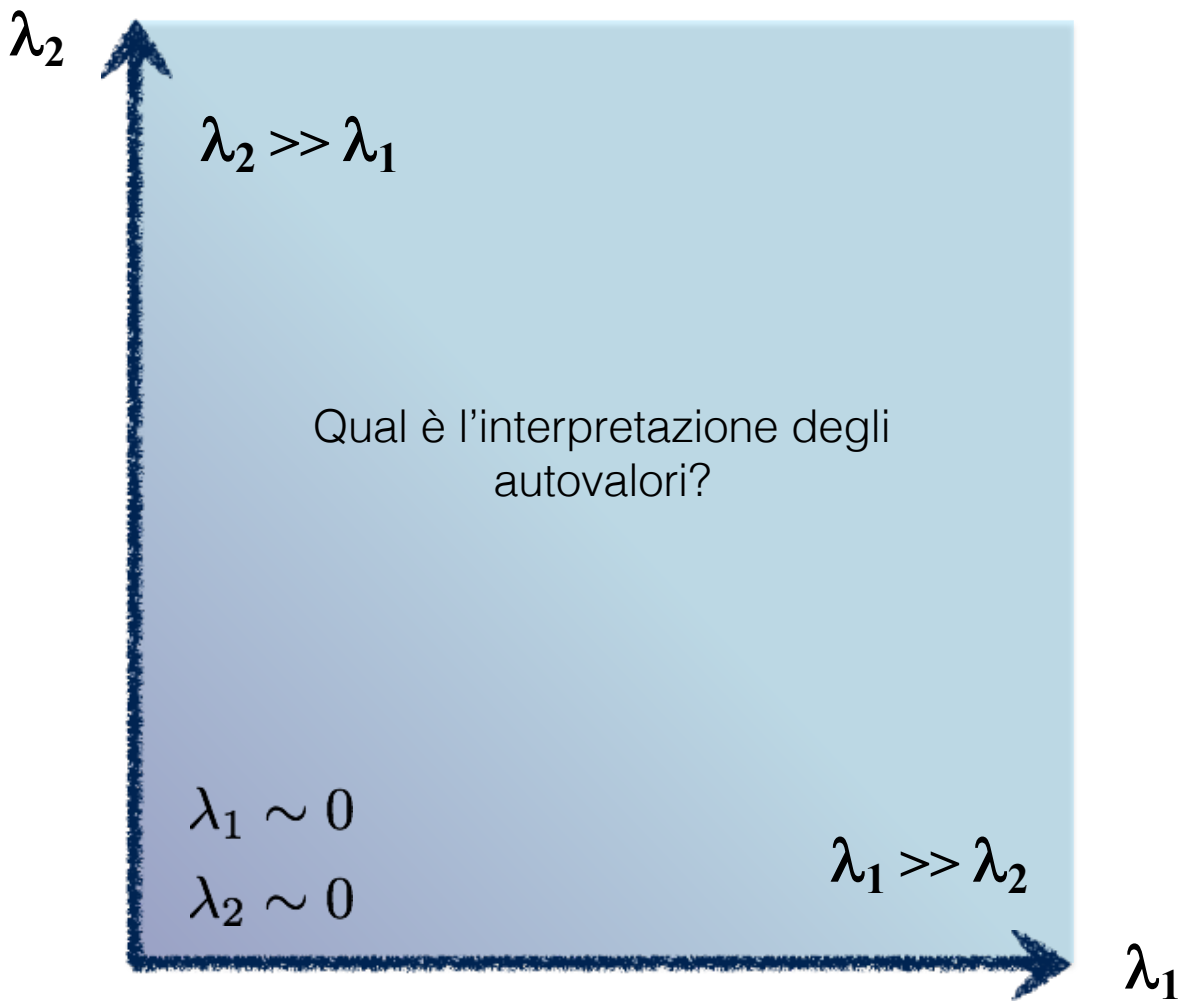
$$M = R^{-1}\begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix}R$$

Equazione dell'ellisse

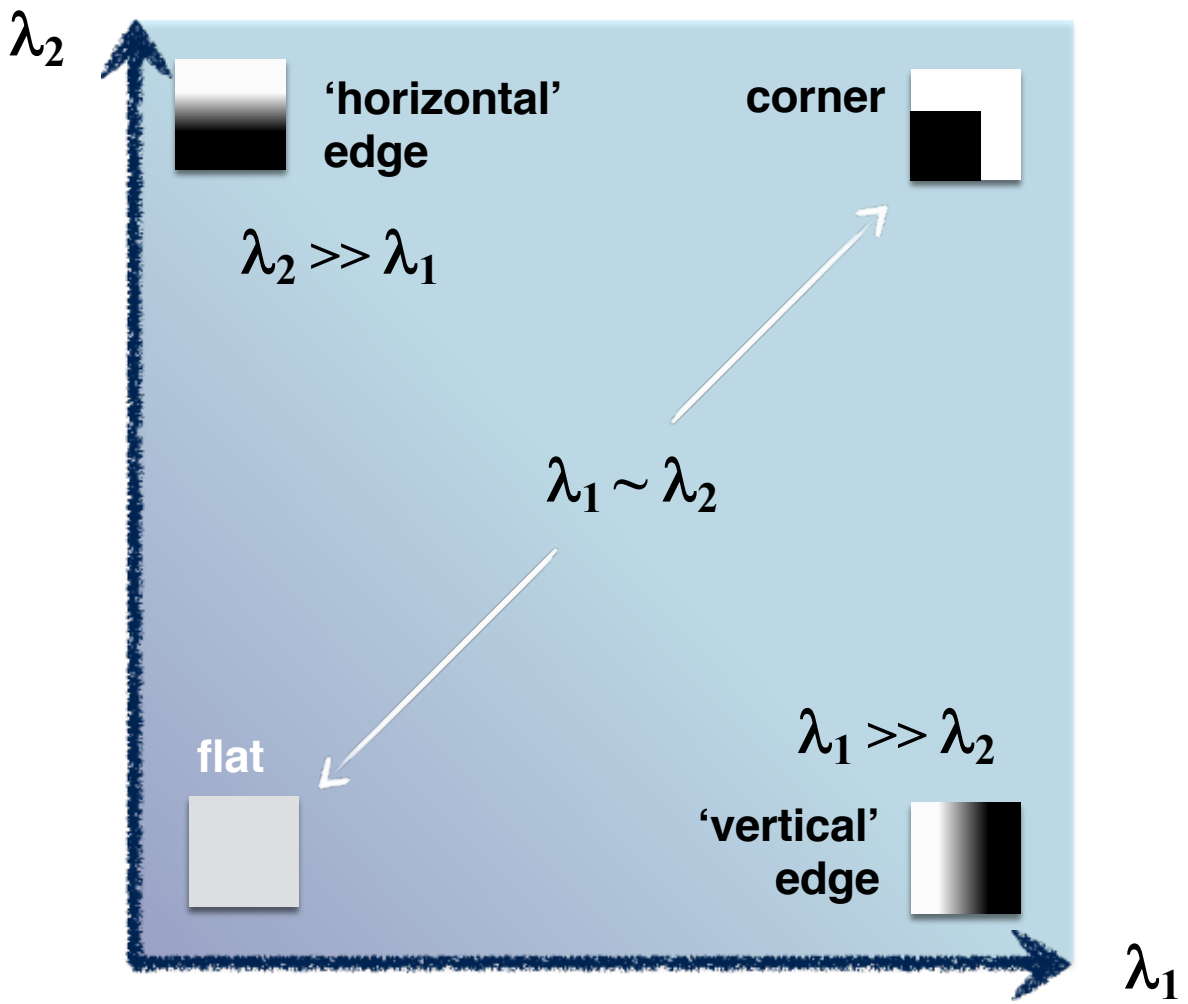
$$[u \;\; v]\;\; M \begin{bmatrix}u\\ v\end{bmatrix} = \text{const}$$



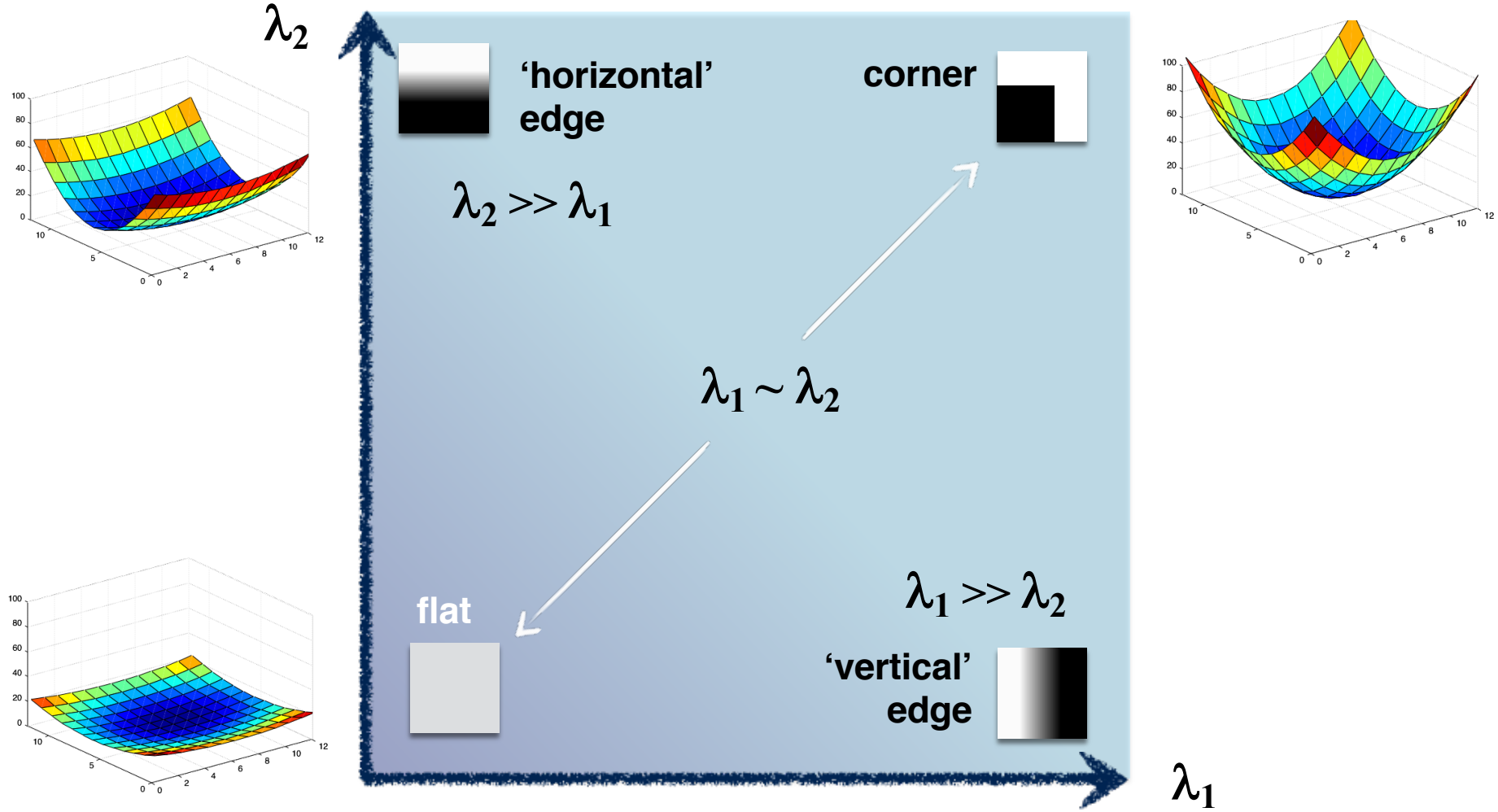
Gli autovalori



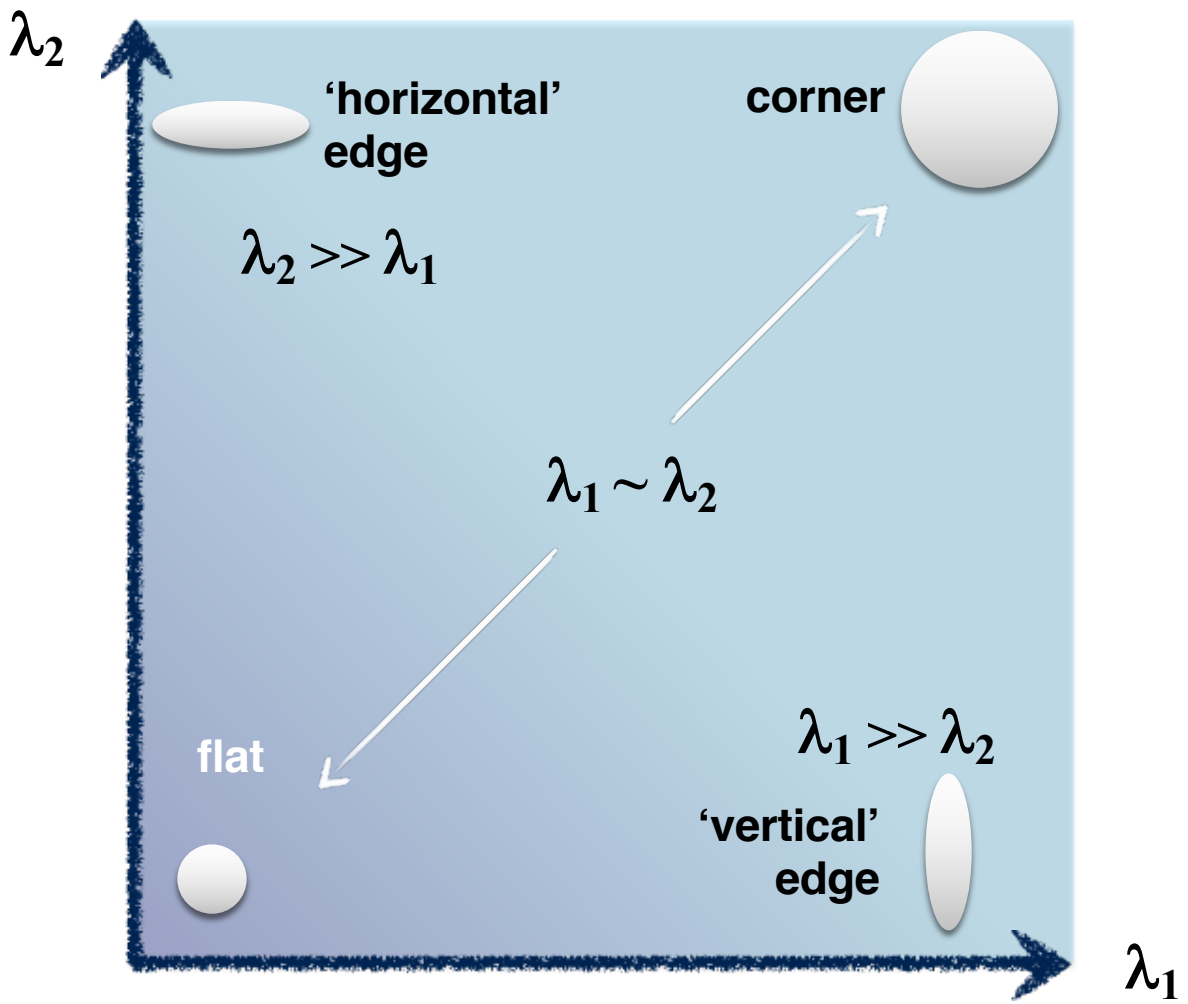
Gli autovalori



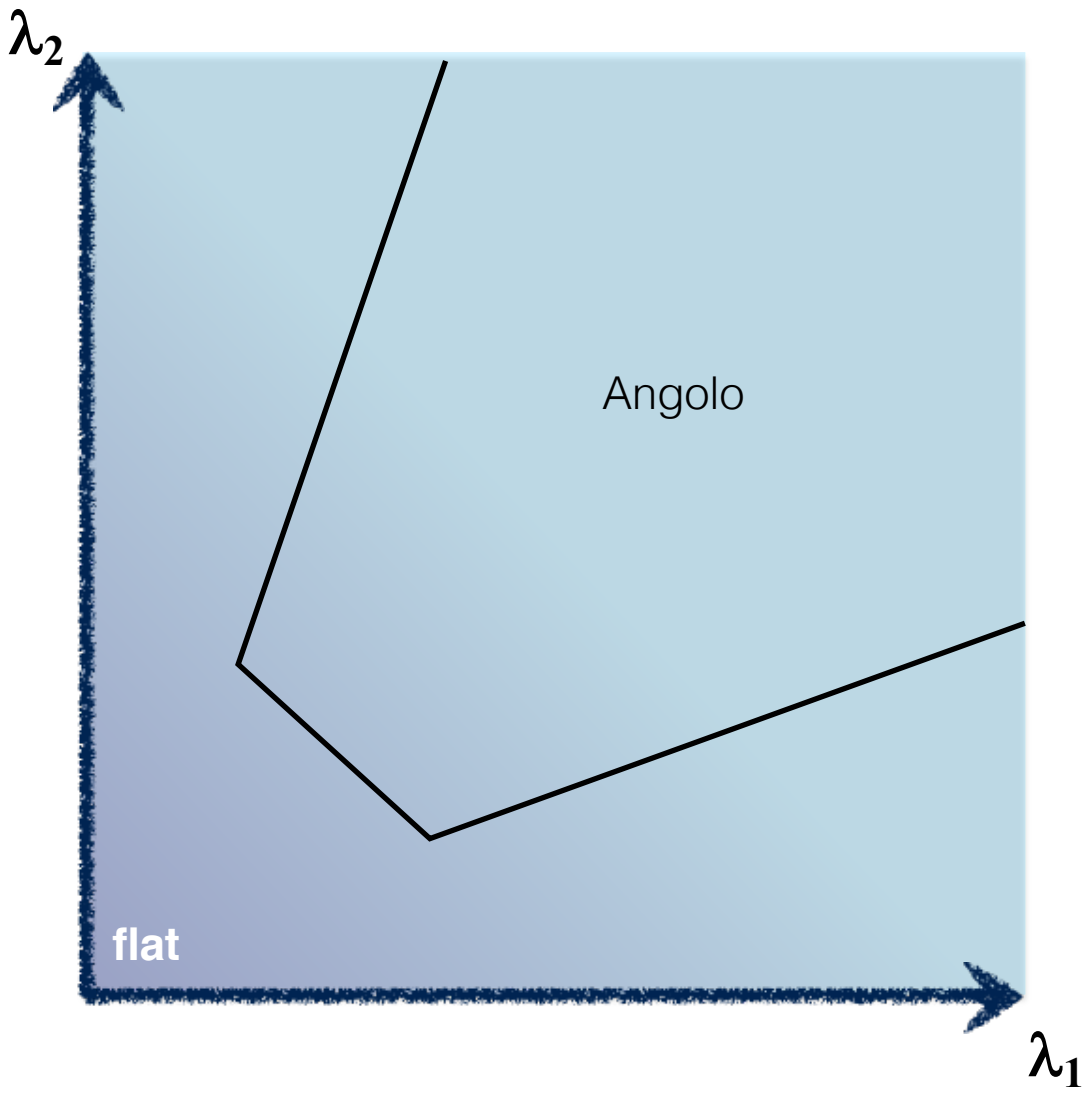
Gli autovalori



Gli autovalori

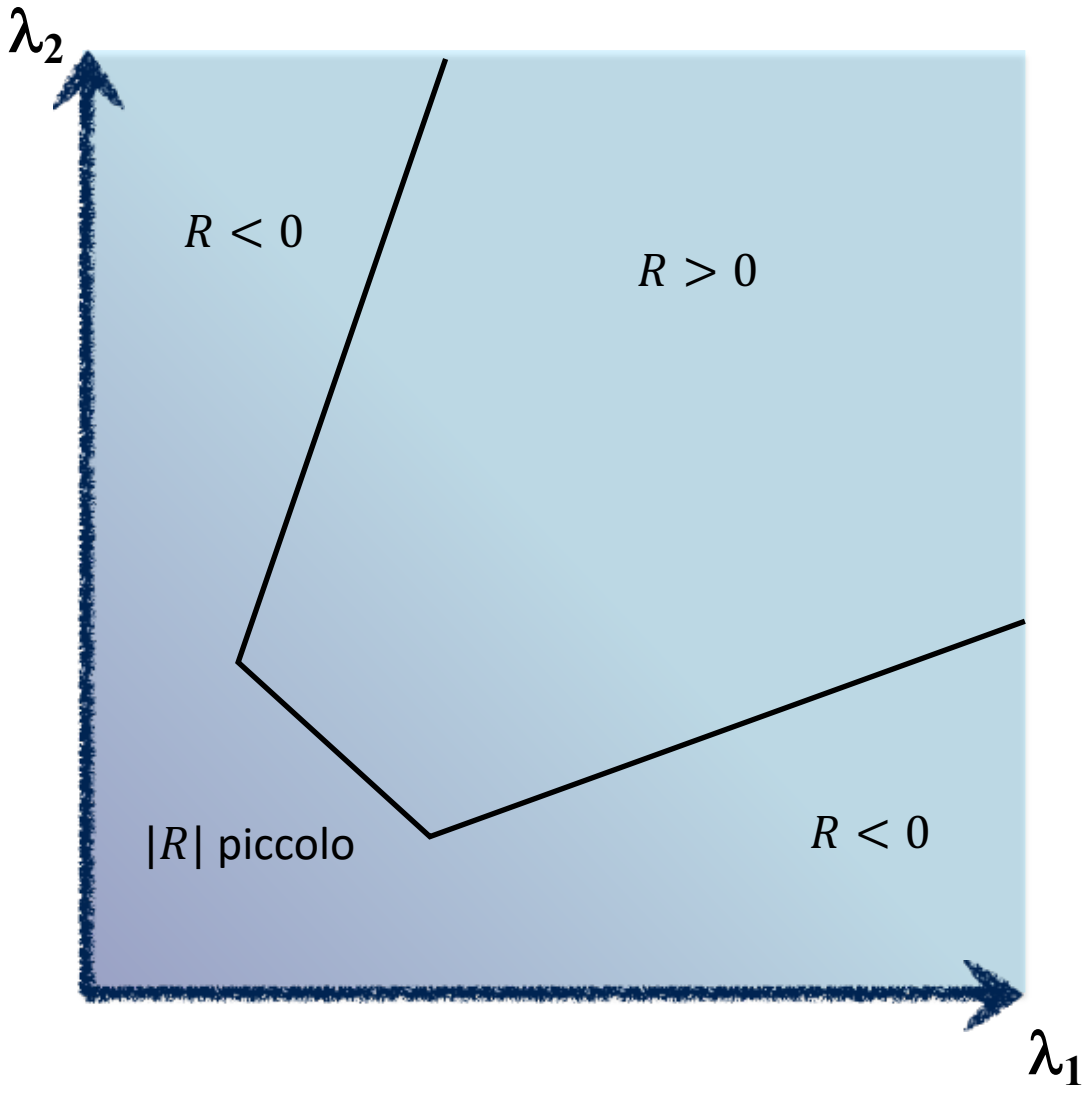


Gli autovalori



Gli autovalori

$$R = \det(M) - \alpha Tr(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$



Harris Detector

1. Calcola le derivate di un'immagine

$$I_x = G_\sigma^x * I \qquad I_y = G_\sigma^y * I$$

2. Calcolo i prodotti per ogni pixel

$$I_x^2 = I_x I_x \qquad I_y^2 = I_y I_y \qquad I_{xy} = I_x I_y$$

3. Calcola la somma dei prodotti ad ogni pixel

$$S_x^2 = G_{\sigma'} * I_x^2 \qquad S_y^2 = G_{\sigma'} * I_y^2 \qquad S_{xy} = G_{\sigma'} * I_{xy}$$

Harris Detector

4. Calcola la matrice per ogni pixel

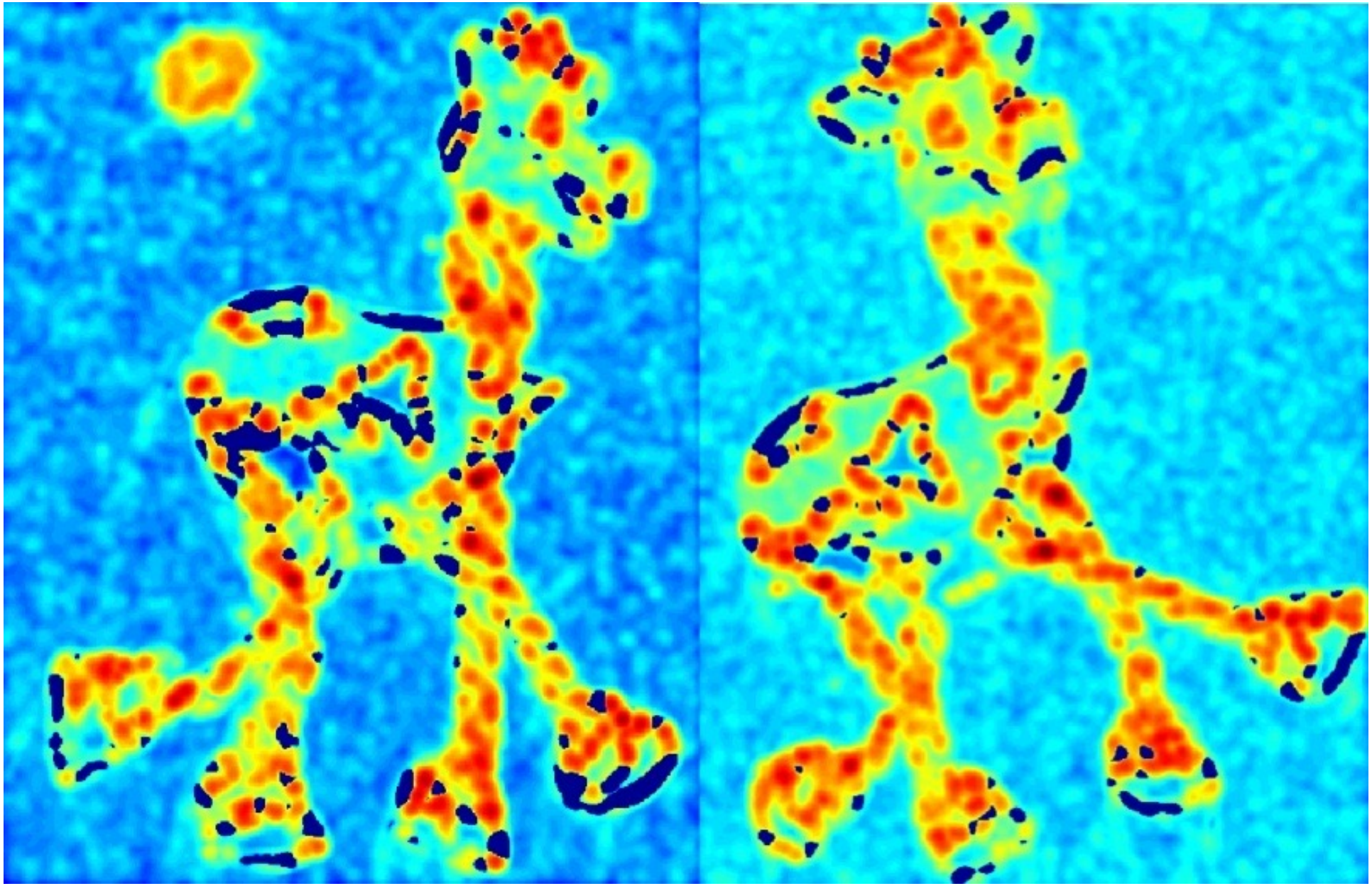
$$M = \begin{bmatrix} S_x^2 & S_{xy} \\ S_{xy} & S_y^2 \end{bmatrix}$$

5. Calcola il responso

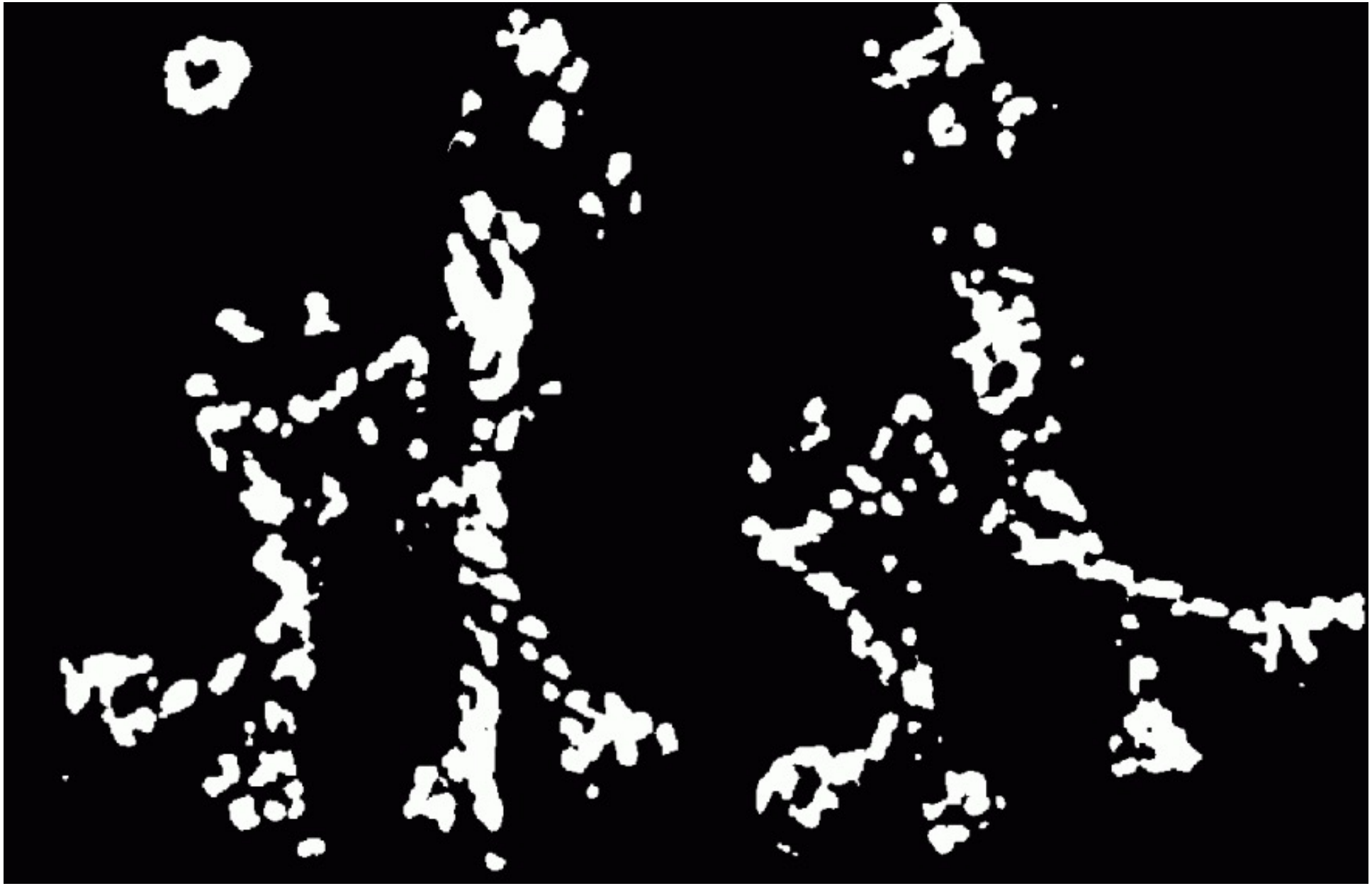
$$R = \det(M) - \alpha Tr(M)^2$$

6. Applica una soglia a R
7. Calcola la non-maximal suppression

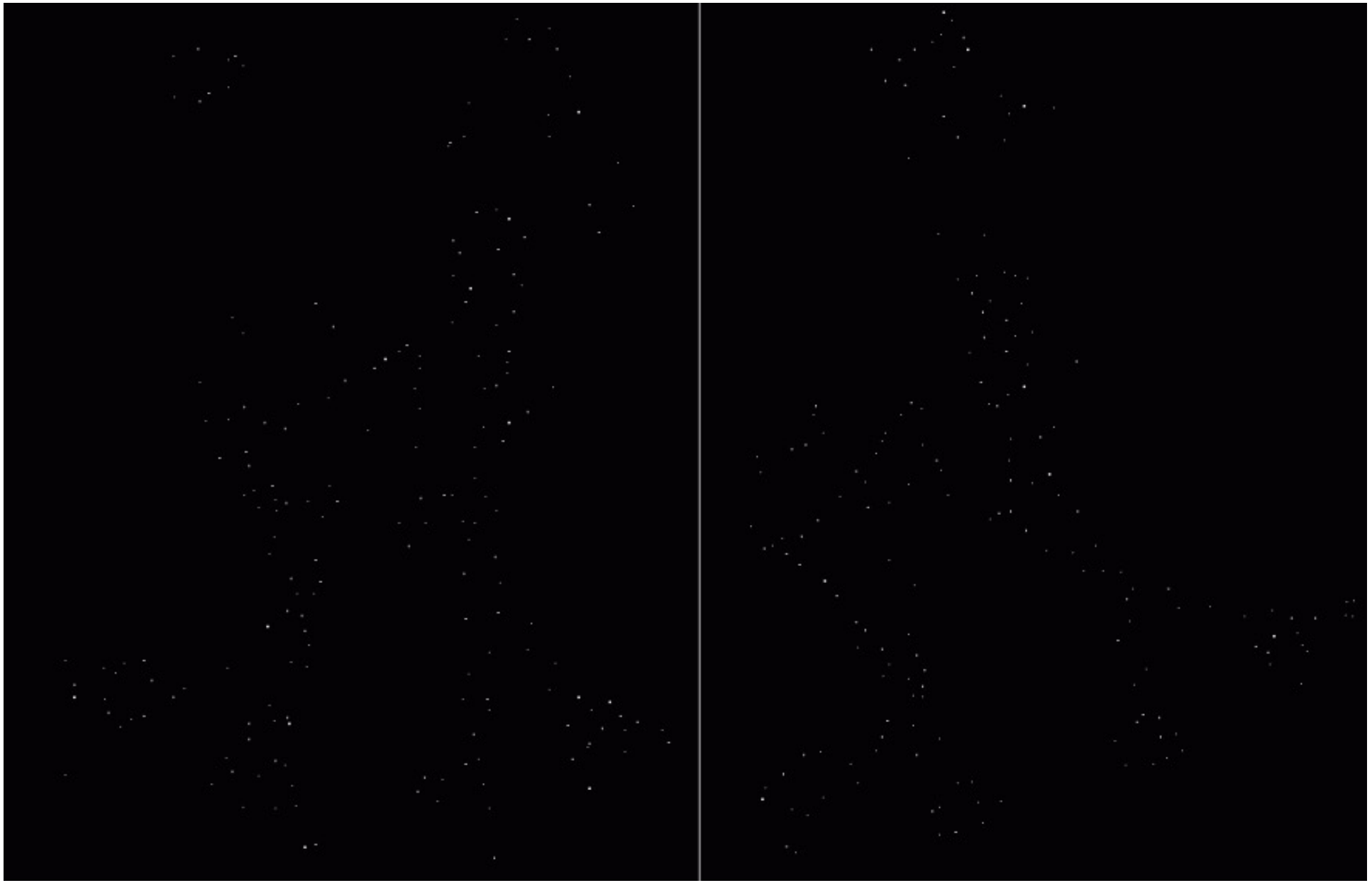
Calcolo di R

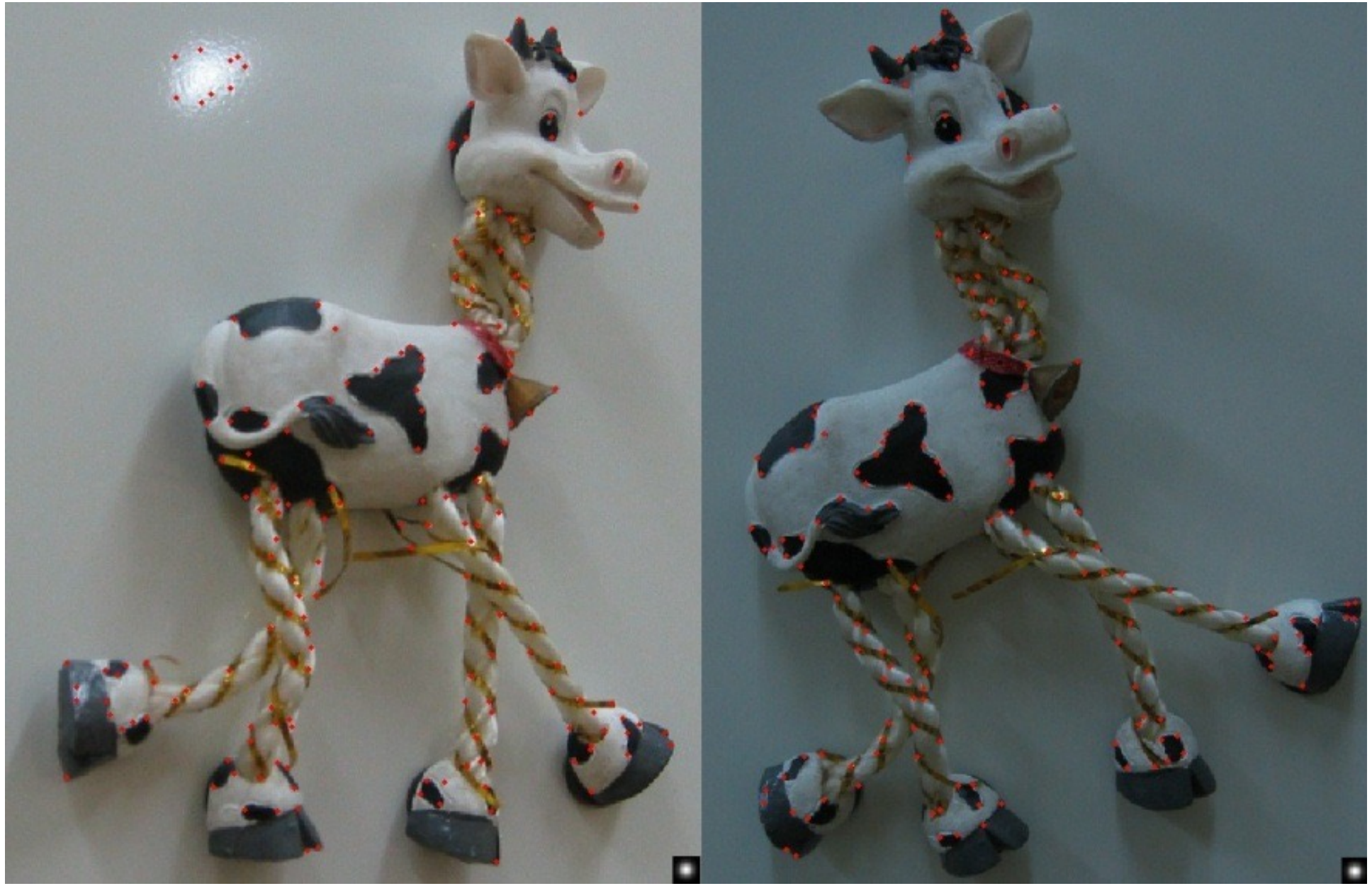


Thresholding



Non-maximal suppression





Qual è la qualità?

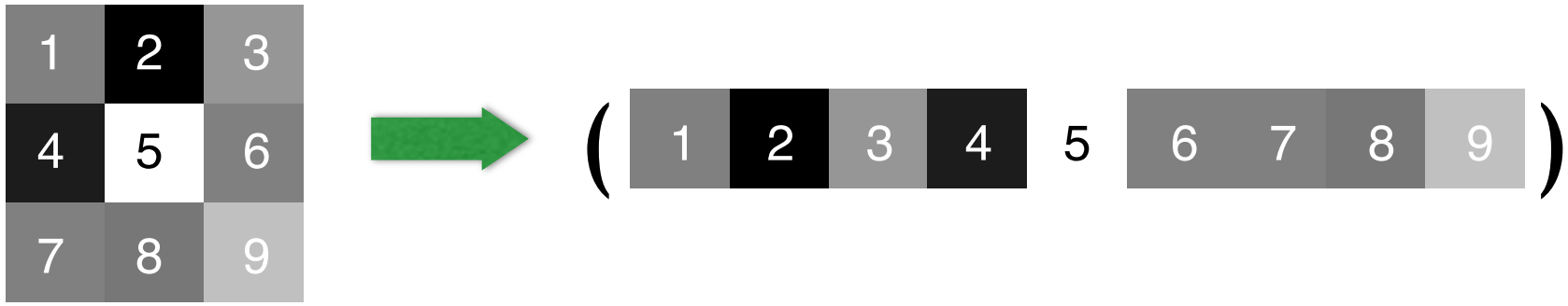
- Invarianza
  - Rotazione?
  - Variazioni di intensità?
  - Scala?

Quali sono i descrittori di un keypoint?

- Vettori
- Istogrammi
- Risposte su patches

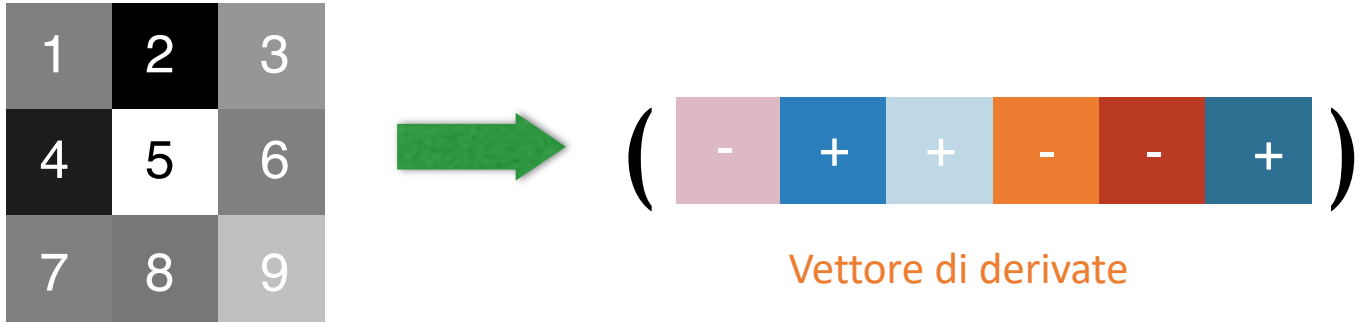
Image patch

Vettore di intensità dei colori, combinato con downsampling

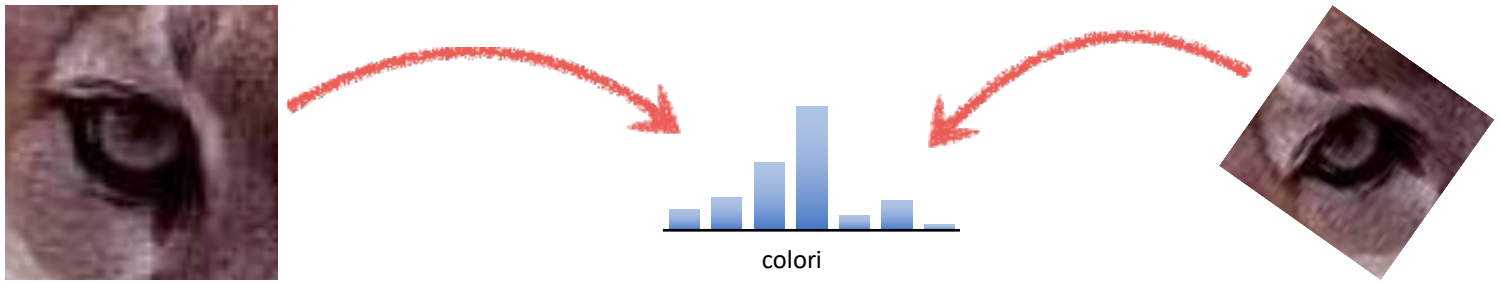


Gradienti

Differenze sui pixel



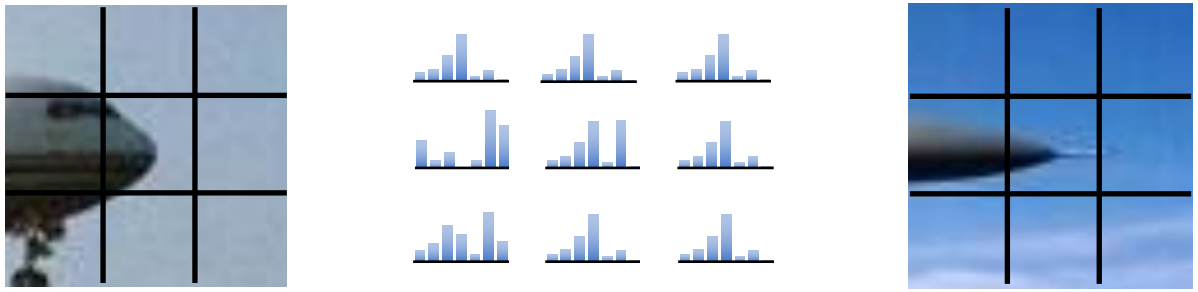
Istogrammi di colori



Invarianti ai cambiamenti di scala e rotazione

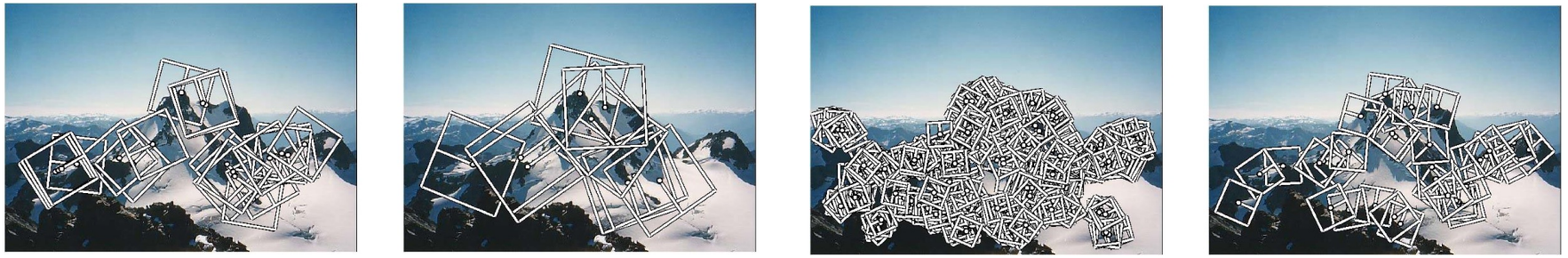
Istogrammi spaziali

Calcoliamo gli istogrammi su celle



Invariante alle deformazioni

MOPS: Multi-Scale oriented patches



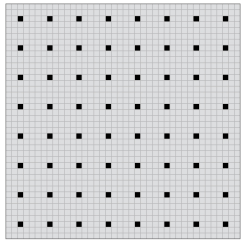
MOPS

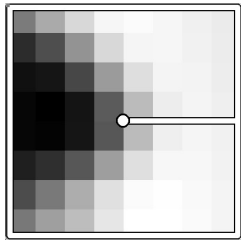
Multi-Image Matching using Multi-Scale Oriented Patches. M. Brown, R. Szeliski and S. Winder. International Conference on Computer Vision and Pattern Recognition (CVPR2005). pages 510-517

$$(x, y, s, \theta)$$

Data una feature

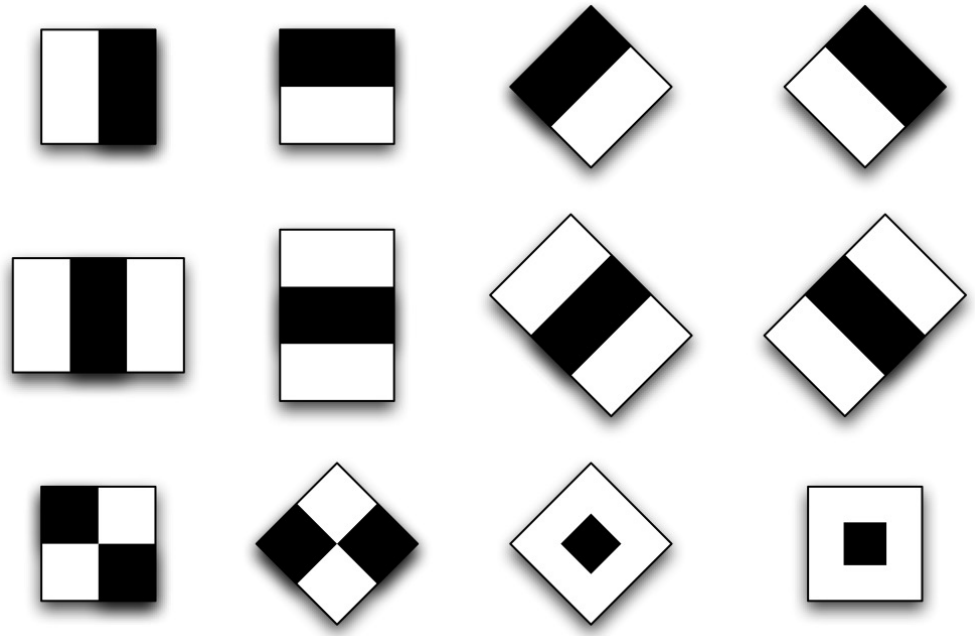
- Considera una patch 40 x 40, campionata
- Standardizza i valori
- Applica i filtri di Haar





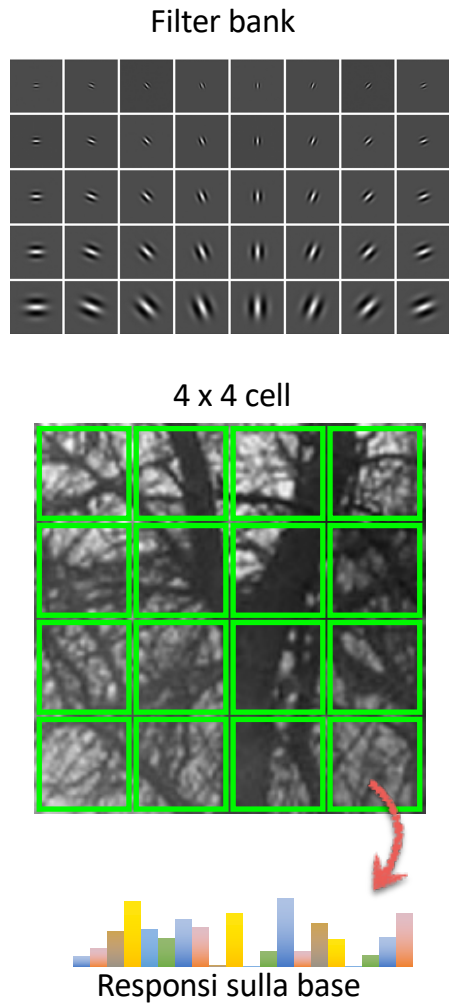
Haar-like features

Si calcola il responso della patch su ogni filtro come descrittore



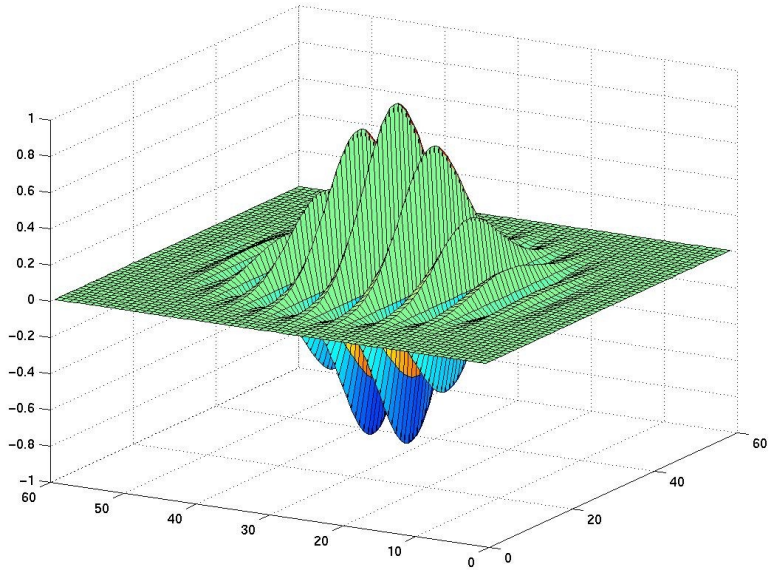
GIST

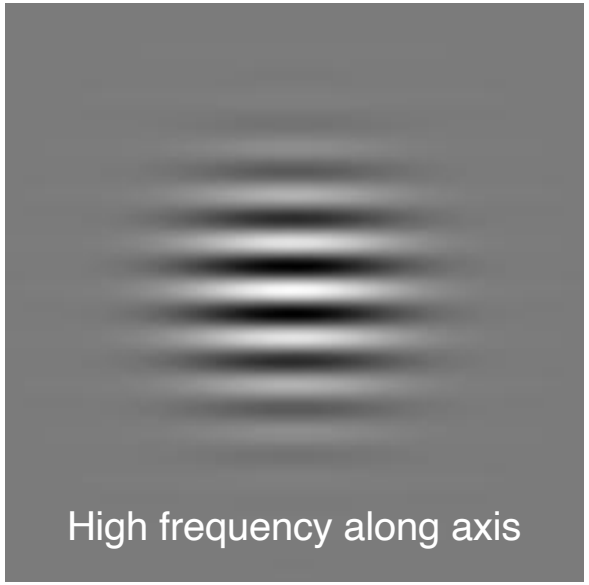
1. Considera una base di Gabor filters
2. Dividi la patch in celle 4 x 4
3. Calcola la media del responso del filtro per ogni cella
4. Il descrittore finale è 4 x 4 x N, dove N è la dimensione della base

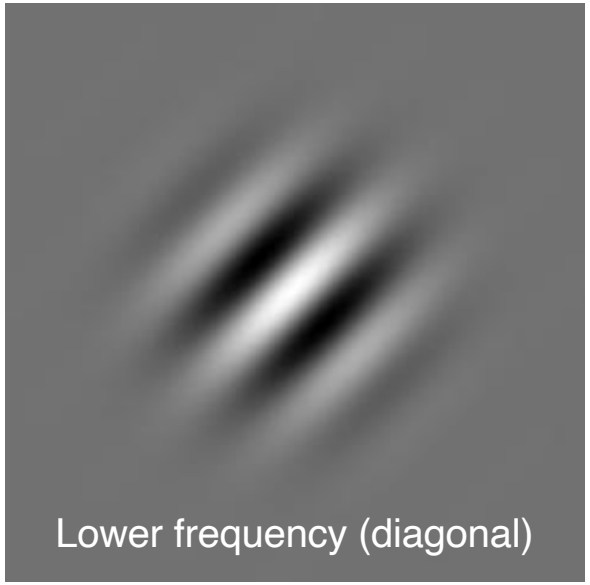


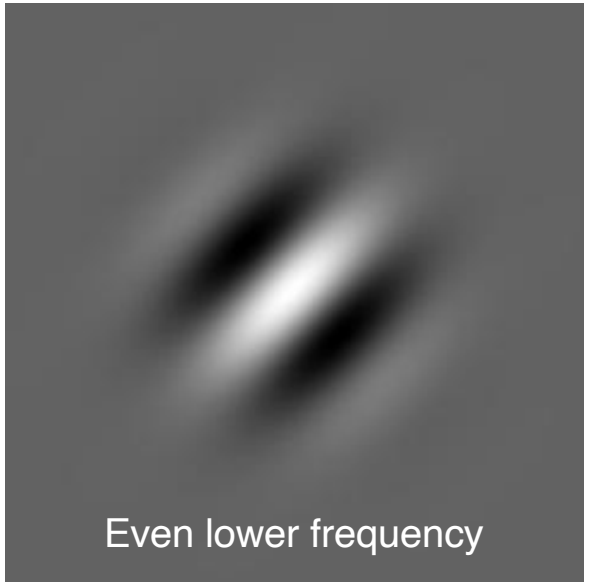
2D Gabor Filters

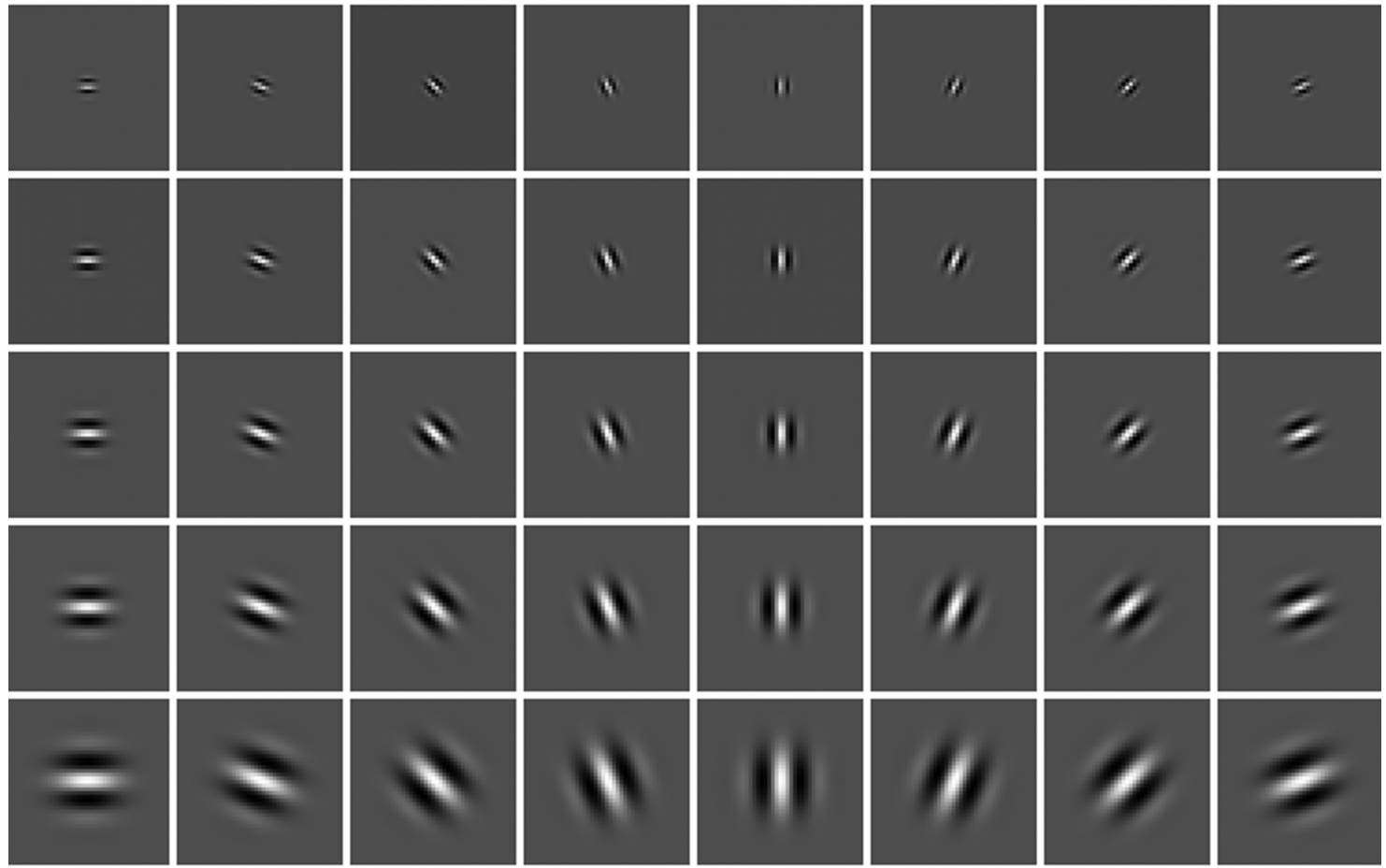
$$e^{-\frac{x^2+y^2}{2\sigma^2}} \cos\left(2\pi\left(k_x x + k_y y\right)\right)$$





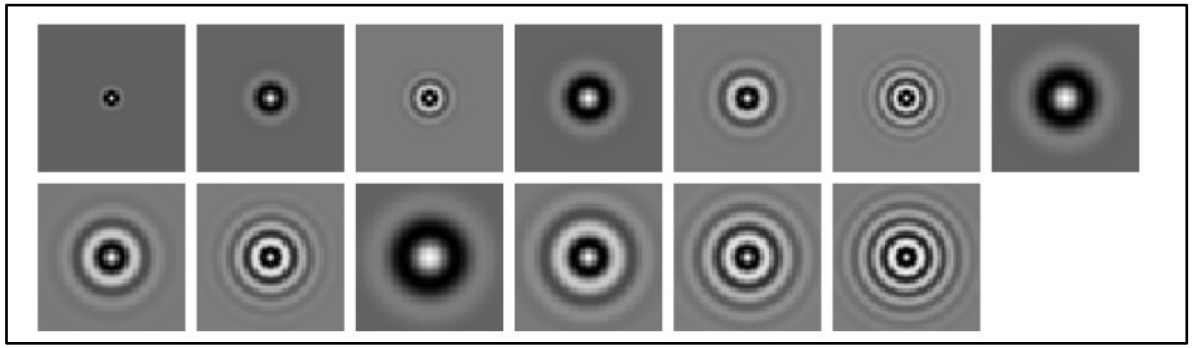






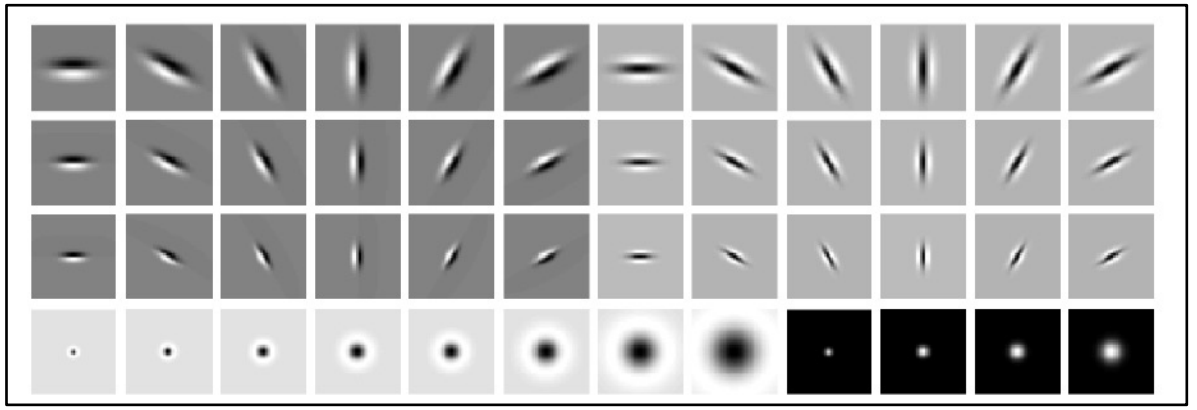
Altri esempi di basi di filtri

Gabor isotropico

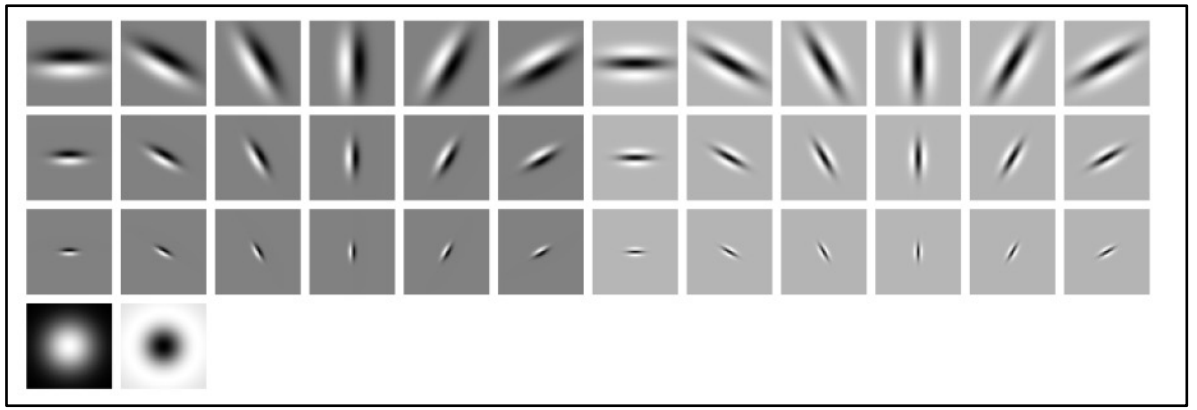


'S'

Derivate gaussiane a varie scale direzioni



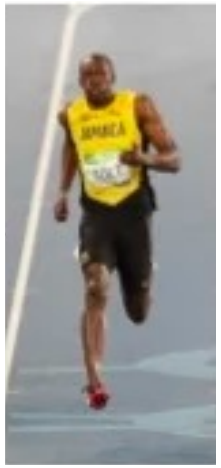
'LM'



'MR8'

HoG: Istogrammi di gradienti orientati

Calcoliamo il gradiente su una patch

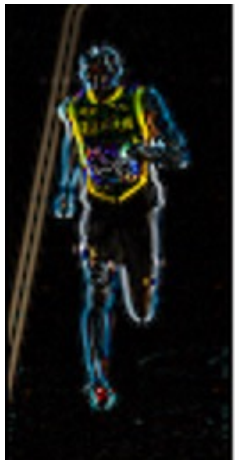


$*$

-1	0	1

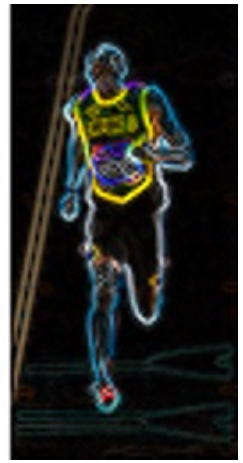
-1
0
1

I_x



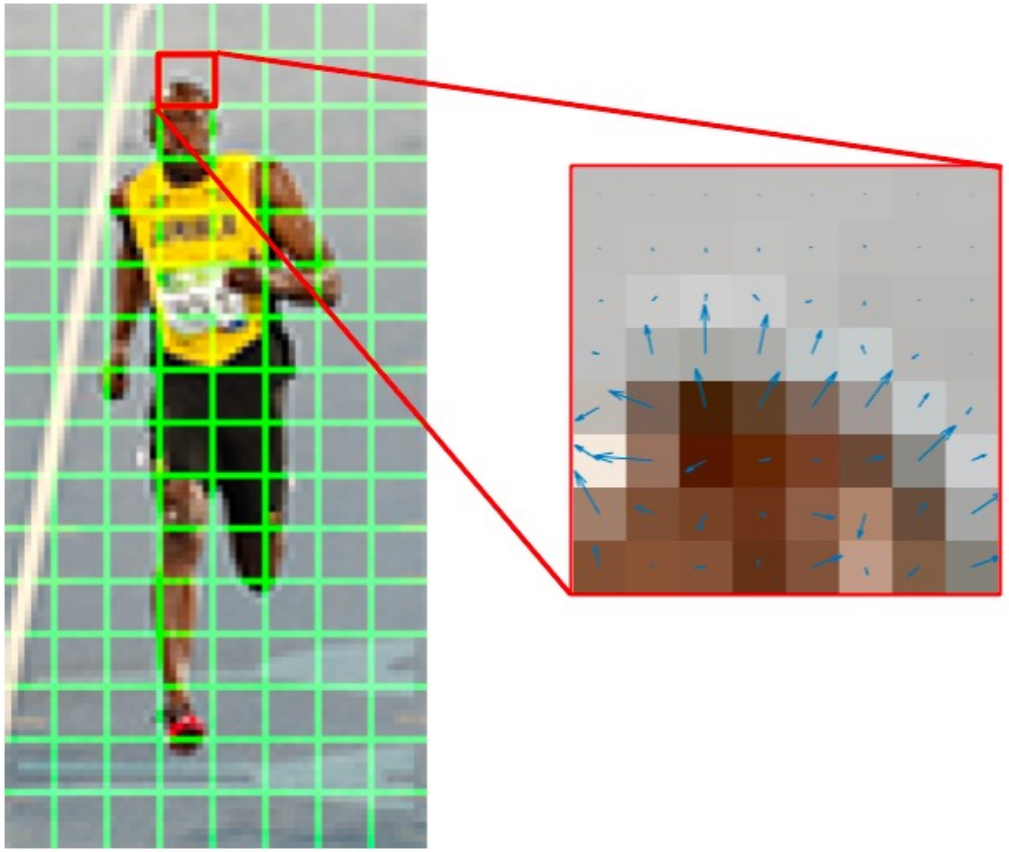
I_y

$\sqrt{I_x^2 + I_y^2}$



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

HoG: Istogrammi di gradienti orientati



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

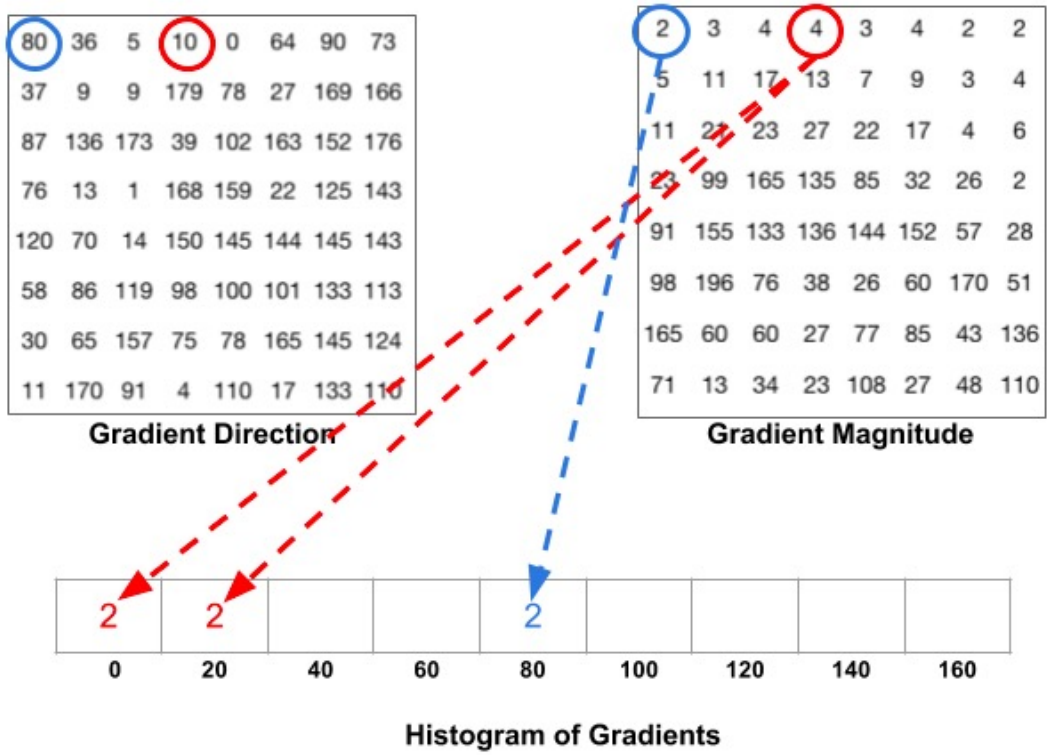
Gradient Magnitude

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

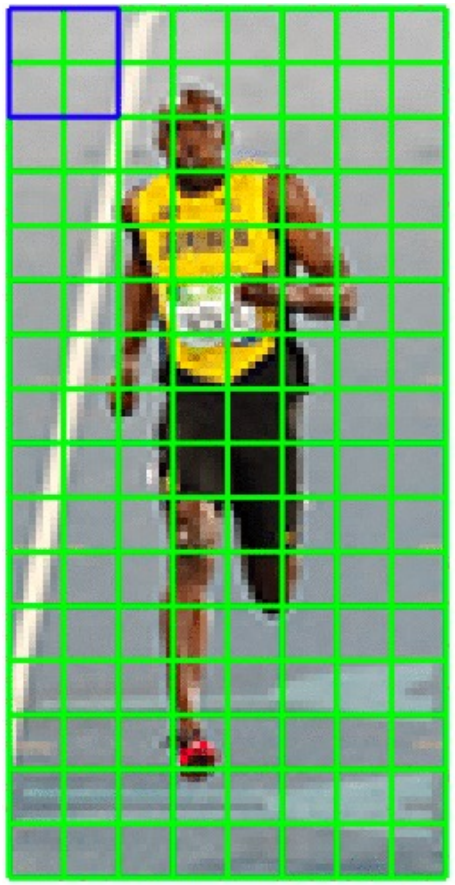
Aggreghiamo le ampiezze in bin “direzionali” in regioni 8x8

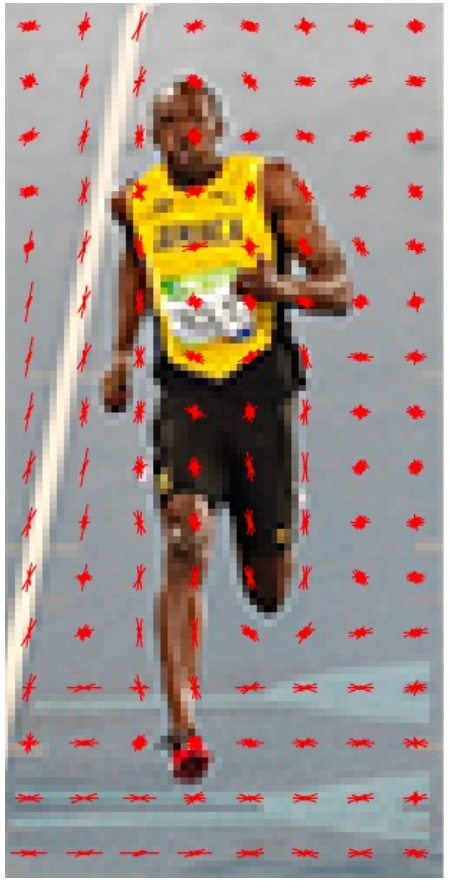
HoG: Istogrammi di gradienti orientati



Ogni valore di ampiezza contribuisce al bin associato

HoG: Istogrammi di gradienti orientati

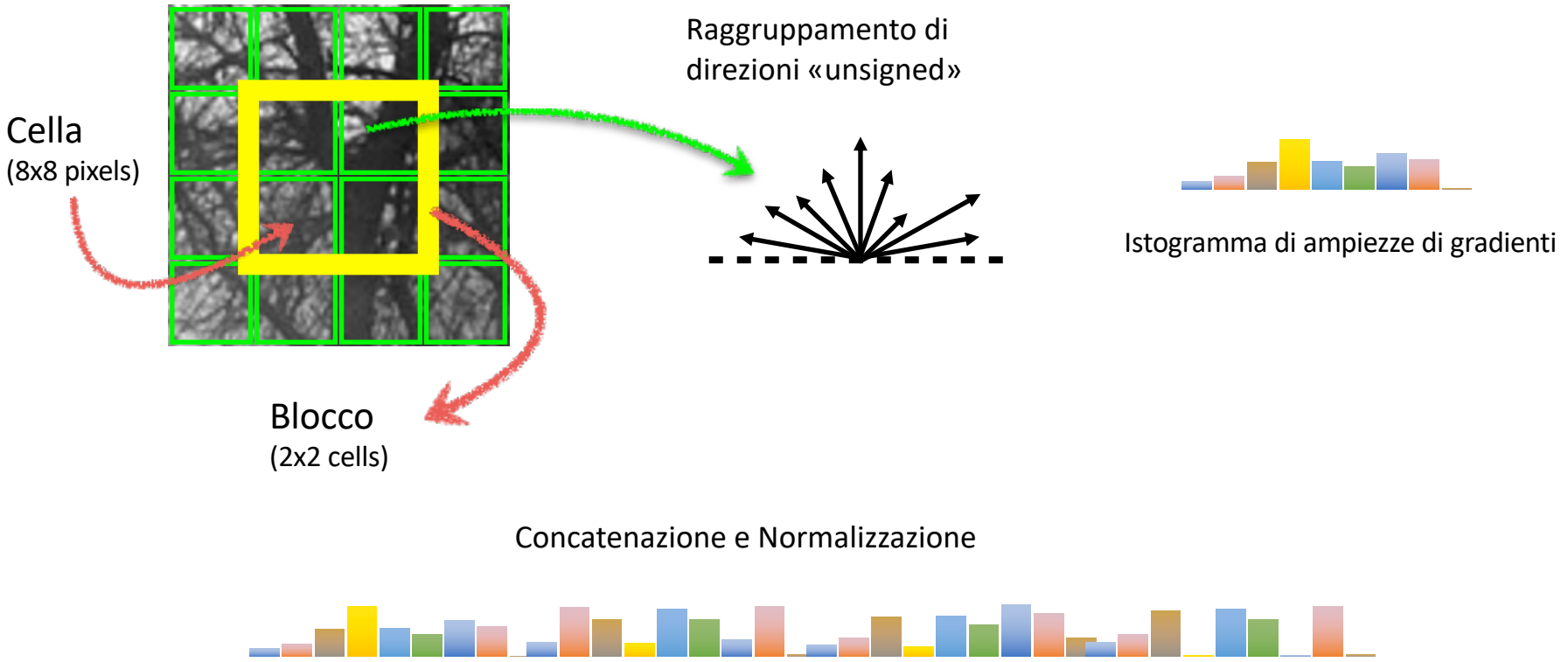




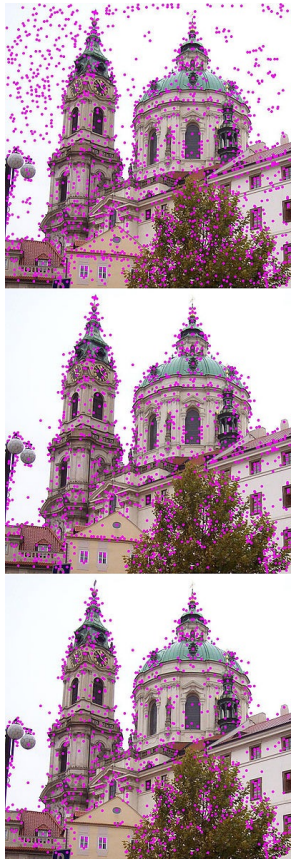
Normalizziamo l’istogramma rispetto agli istogrammi adiacenti

Il descrittore finale è la concatenazione di tutti gli istogrammi

HoG: Istogrammi di gradienti orientati



SIFT
(Scale Invariant Feature Transform)

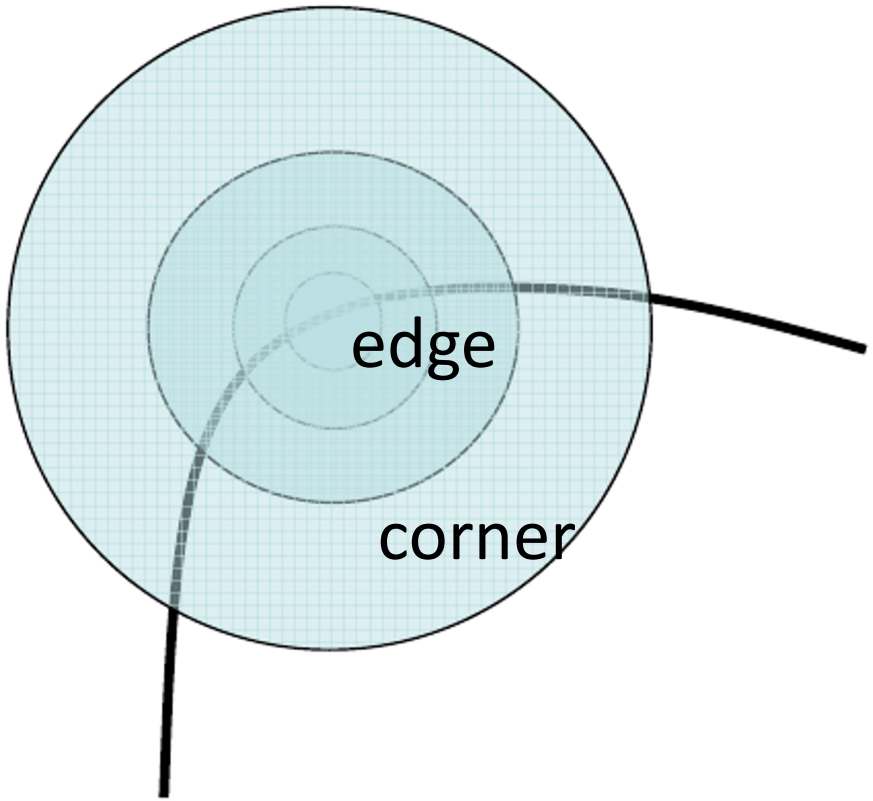


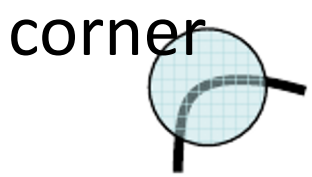
Descrive sia un detector che un descrittore

1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

https://docs.opencv.org/3.1.0/da/df5/tutorial_py_sift_intro.html

Scale Invariant Local Features





Scale Space

- Lo **scale space** è una rappresentazione multiscala dell'immagine basata su un parametro di scala continuo σ
- La rappresentazione a scala σ si ottiene effettuando la convoluzione di *f* con il kernel Gaussiano avente deviazione standard σ
- Le informazioni a scala più fine vengono progressivamente soppresse

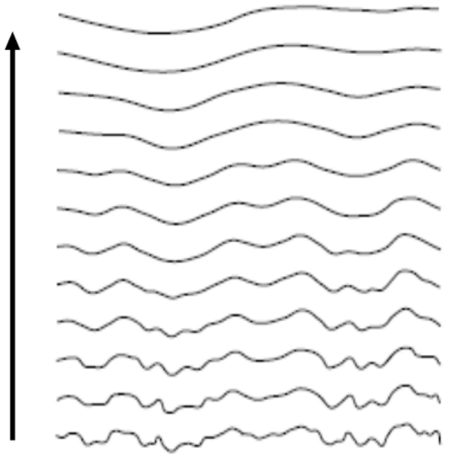


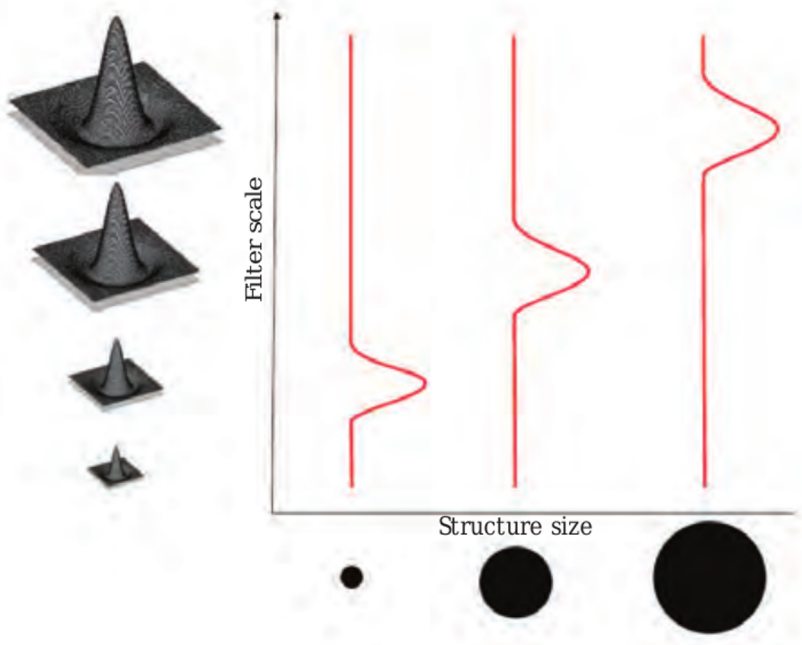
Immagine originale | σ=1 | σ=2 | σ=4

Laplacian of Gaussian Detector

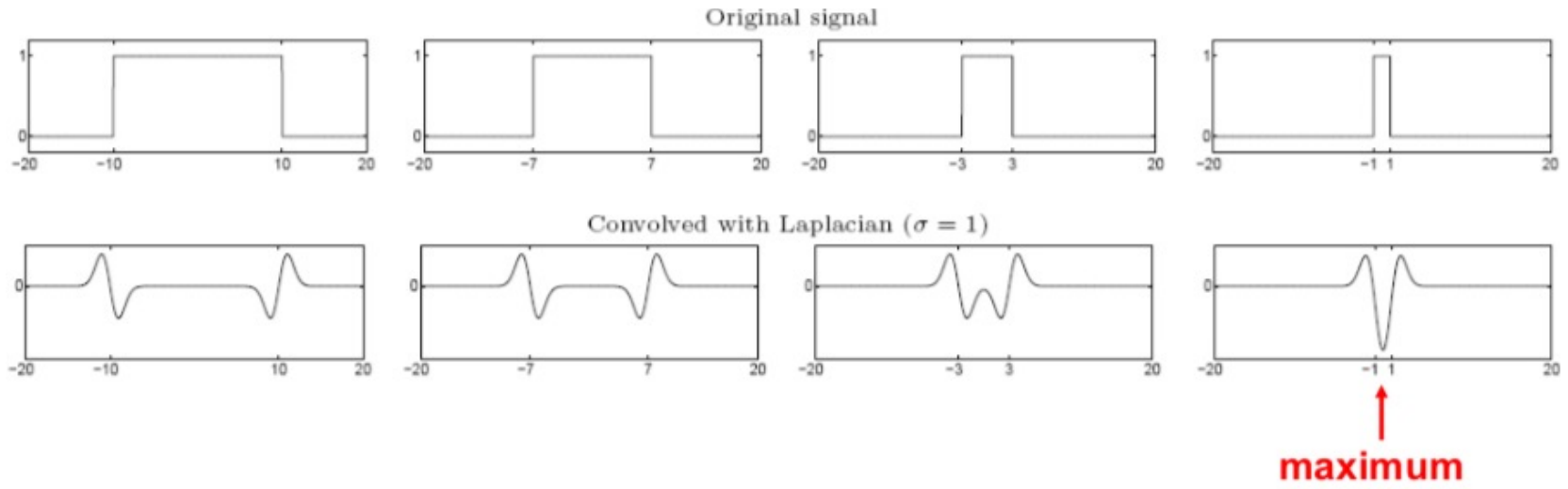
- LoG scale invariant keypoint detector:

$$L(x, y; \sigma) = \sigma^2 \left[\nabla^2 G(x, y; \sigma) \star f(x, y)\right]$$

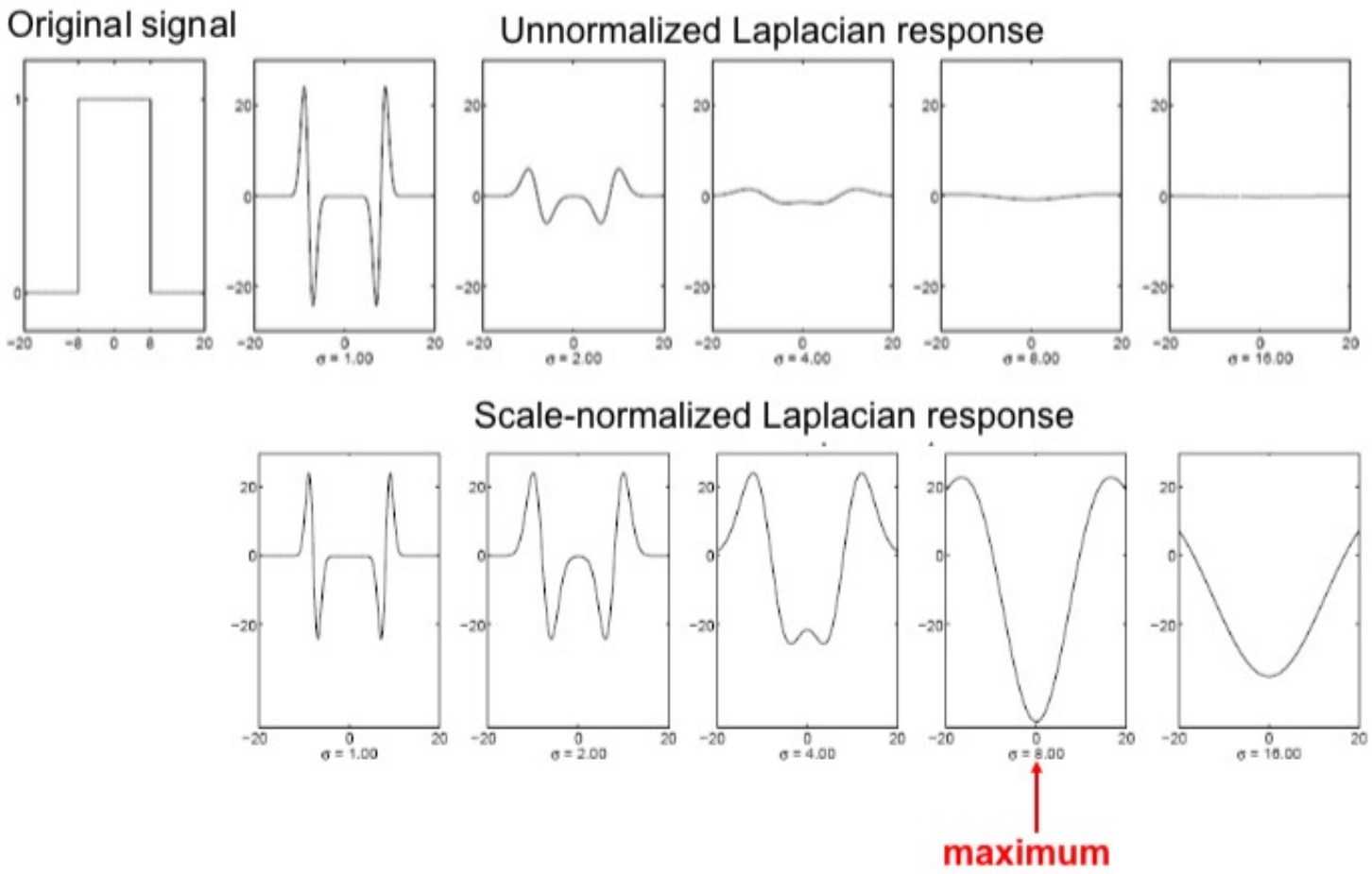
- dove il fattore σ^2 è richiesto per l'invarianza di scala
- Il LoG filter massimizza la risposta quando viene applicato ad una regione che contiene una struttura circolare approssimativamente della dimensione del filtro stesso (*blob detector*)



Laplacian response (in 1D)

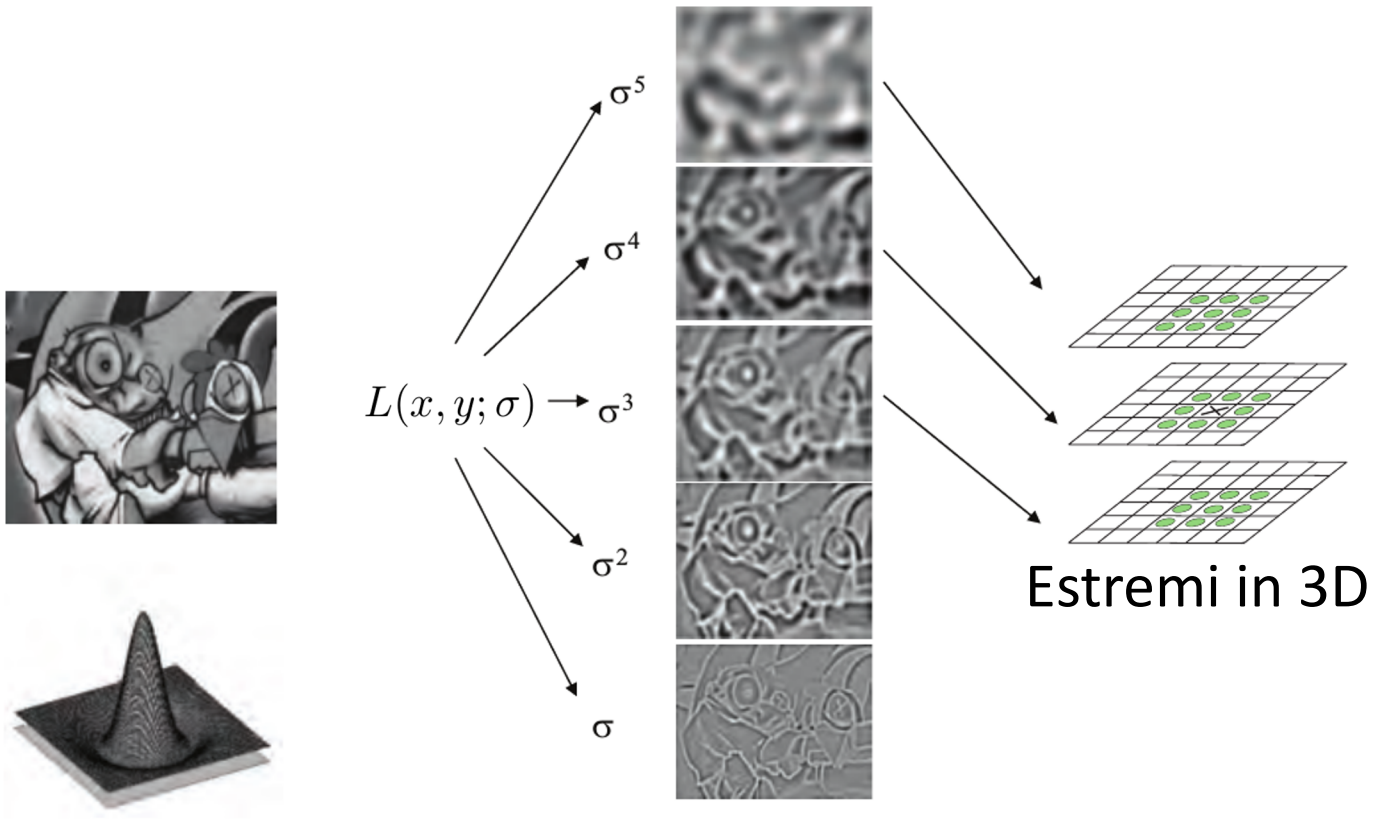


Laplacian response (in 1D)



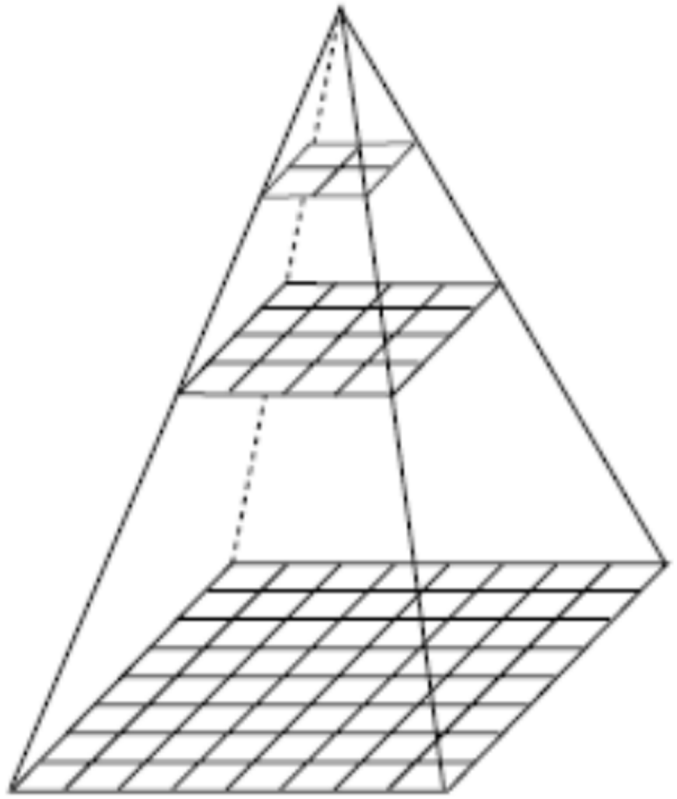
Laplacian of Gaussian Detector

- I LoG keypoint sono gli estremi in questa rappresentazione scale space



Piramide Gaussiana

- La piramide è una collezione di rappresentazioni di un'immagine strutturate su più livelli, in cui ogni livello ha in genere dimensione pari ad un quarto del livello sottostante
- In una piramide gaussiana, ogni livello è ottenuto da quello sottostante effettuando (1) un filtraggio con un kernel Gaussiano di deviazione standard σ e quindi (2) una scalatura di un fattore 0.5



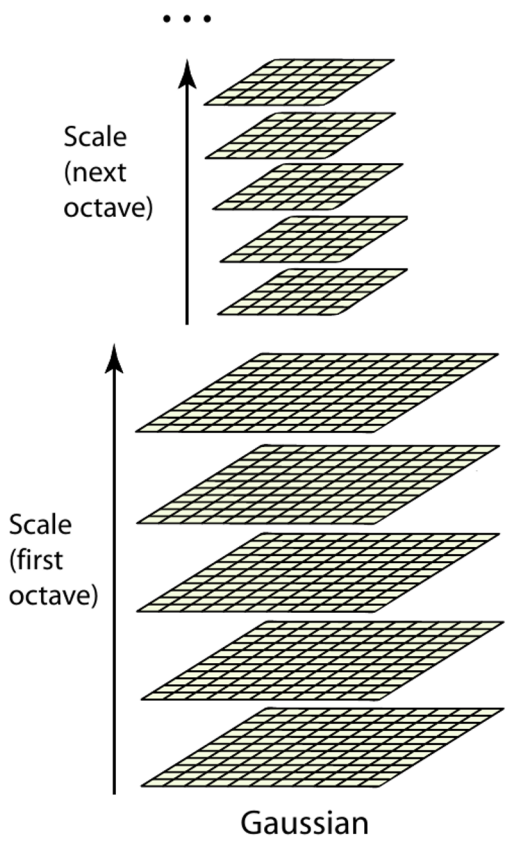
Piramide Gaussiana

- Dato un fattore di scala σ ed un operatore di scalatura *S*:
  - $f_0(x,y) = f(x,y)$
  - $f_{n+1}(x,y) = S(G(x,y,\sigma)*f_n(x,y))$
- dove $f_{n+1}(x,y)$ ha sempre dimensione pari ad un quarto di $f_n(x,y)$ per effetto del dimezzamento del numero di pixel in verticale ed in orizzontale (l'immagine $f_n(x,y)$ al livello *n* della piramide è l'equivalente dell'immagine $G(x,y,2^{n-1}\sigma)*f_0(x,y)$)

Piramide Gaussiana

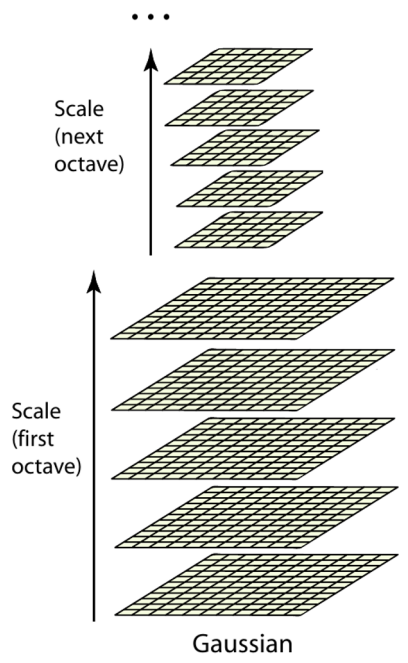
Piramide Gaussiana: Ottave

- Per rendere più efficiente l'analisi, lo scale space è diviso in **ottave**, ognuna delle quali copre un livello della piramide Gaussiana (scale da σ_L a $2\sigma_L$)
- Sia σ_L la scala di un certo livello della piramide, l'ottava (da σ_L a $2\sigma_L$) viene a sua volta suddivisa in s intervalli (n=0,1,...,s) aventi scala
  - $\sigma_n = k^n \sigma_L$ con $k = 2^{1/s}$
- Si noti che $\sigma_0=\sigma_L$ e $\sigma_n=2\sigma_L$ e quindi l'ultima scala di ogni ottava corrisponde alla prima scala dell'ottava successiva (in quanto il kernel Gaussiano utilizzato raddoppia ad ogni livello della piramide)
- Nel passaggio all'ottava successiva, si dimezza la dimensione l'immagine ed è possibile riapplicare gli stessi kernel gaussiani utilizzati nell'ottava precedente (anziché raddoppiare la dimensione dei kernel)



Piramide Gaussiana: Ottave

- Ad esempio, per *s*=3 avremo:
  - $k = 2^{1/s} = 1.2599$
  - $\sigma_0 = k^0 = 1 \rightarrow$ kernel 7x7
  - $\sigma_1 = k^1 = 1.2599 \rightarrow$ kernel 9x9
  - $\sigma_2 = k^2 = 1.5874 \rightarrow$ kernel 11x11
  - $\sigma_3 = k^3 = 2 \rightarrow$ kernel 13x13
- Per ottenere le diverse ottave è sufficiente applicare sempre gli stessi kernel ad ogni livello della piramide



Difference of Gaussian Detector

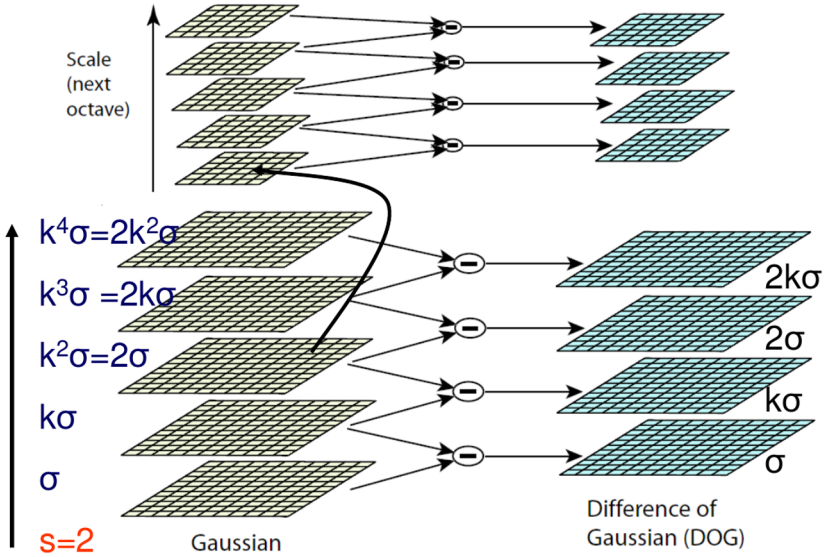
- Il filtro LoG può essere approssimato dalla differenza di due Gaussiane (DoG) aventi differente deviazione standard:

$$\begin{aligned} D(x,y;\sigma) &= [G(x,y;k\sigma) - G(x,y;\sigma)] \star f(x,y) \approx \\ &\approx \left[(k-1)\sigma^2 \nabla^2 G(x,y;\sigma)\right] \star f(x,y) = \\ &= (k-1)L(x,y;\sigma). \end{aligned}$$

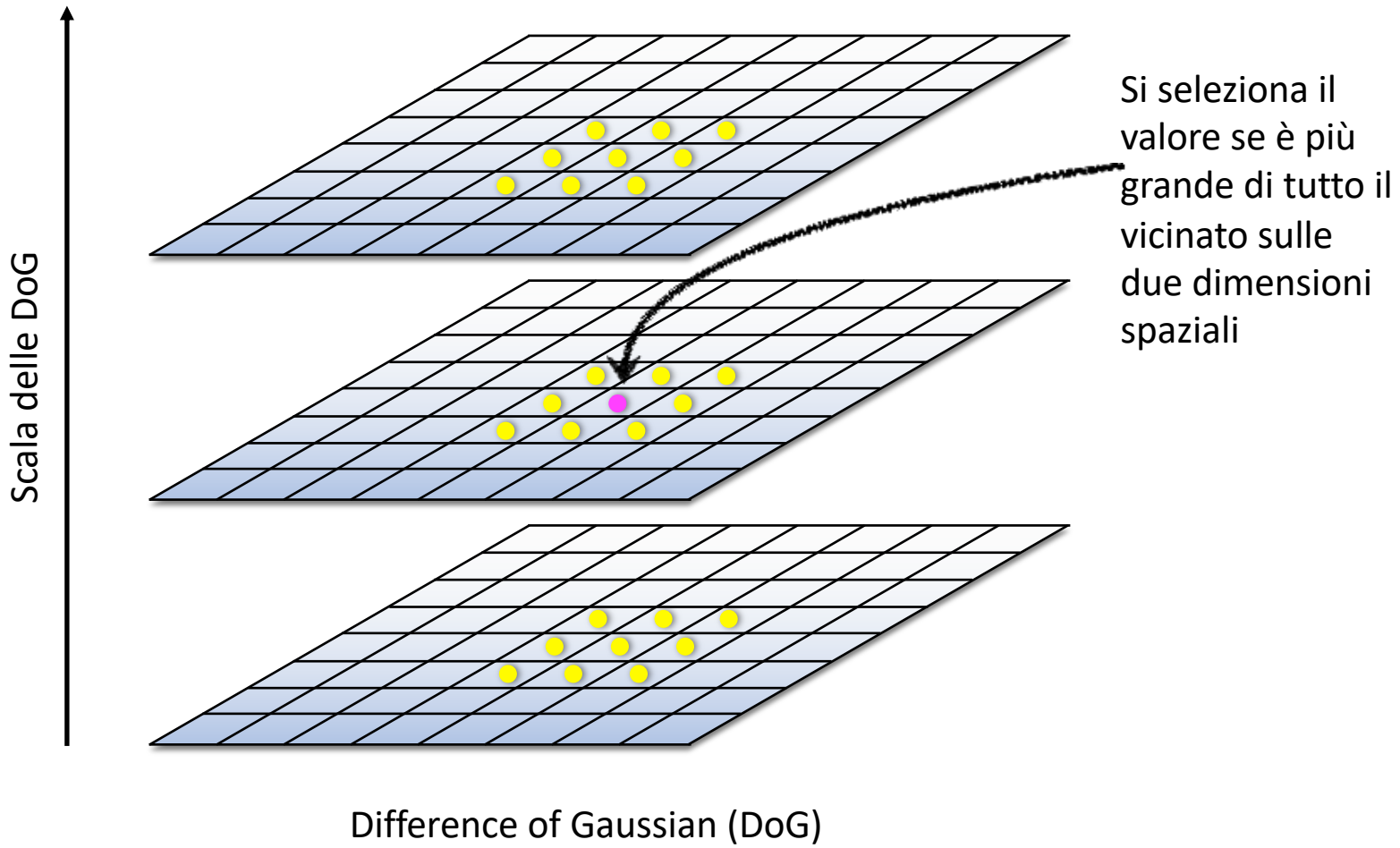
- Il vantaggio di questa espressione è che può essere efficientemente calcolata come differenza di due scale adiacenti

Difference of Gaussian Detector

- Al fine di determinare anche i veri massimi nella prima immagine di ogni ottava, per ogni livello della piramide si determinano *s*+3 intervalli
- Per *s*=2 si calcolano *s*+3=5 immagini con σ crescente di un fattore $k=\sqrt{2}$
- Si calcolano le differenze tra livelli adiacenti (DoG) e si determinano gli estremi nella corrispondente rappresentazione 3D



Scale-space extrema & Keypoint location



Interest Region Extraction

- Per catturare la struttura nell'intorno del keypoint, viene considerata la regione di raggio *r* centrata nel punto individuato (raggio $r=3\sigma$ viene utilizzato da diversi detector, dove σ denota la scala caratteristica del keypoint) detta anche *interest region*
- Dopo che l'interest region è stata selezionata, dev'essere *normalizzata* per invarianza alle rotazioni:
  - Si determina l'*orientazione dominante* e quindi si ruota il contenuto della regione in accordo all'associato angolo in modo da portarla in un'*orientazione canonica*

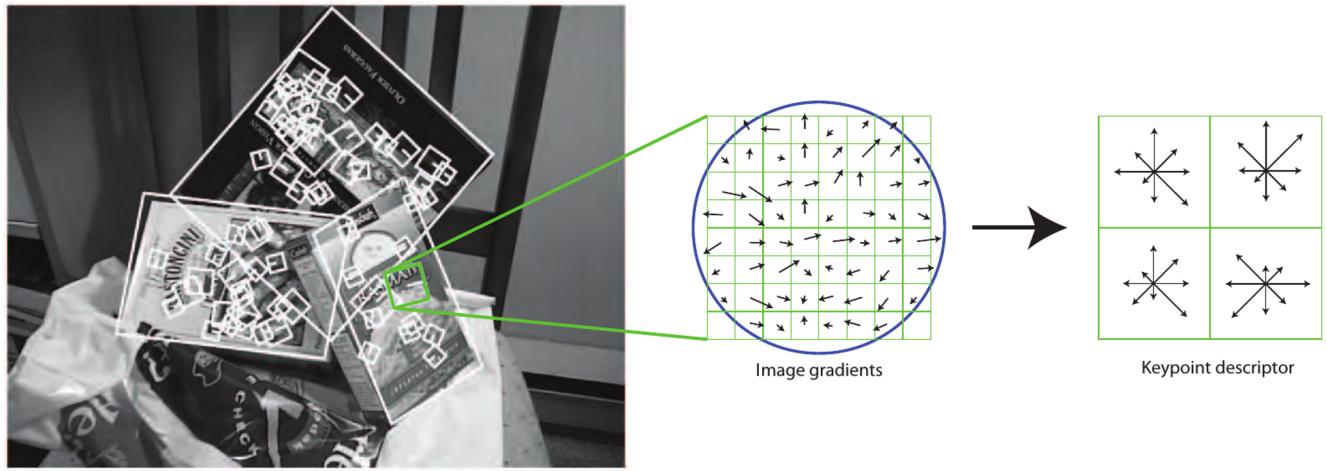
Interest Region Extraction

- Determinazione dell'**orientazione dominante** (Lowe 2004):
  - si costruisce un istogramma delle orientazioni con 36 intervalli (bins) che coprono 360 gradi (con un passo di 10 gradi)
  - Per ogni pixel nella regione considerata, si calcola il gradiente e la corripondente orientazione viene inserita nell'istogramma dopo averla pesata in base al rispettivo modulo e ad una funzione Gaussiana di deviazione standard 1.5σ centrata nel keypoint
  - Il picco dell'istogramma viene scelto come orientazione dominante (per determinare l'orientazione precisa si effettua un'interpolazione tra usando i 3 bin centrati nel picco)

SIFT Feature Descriptor

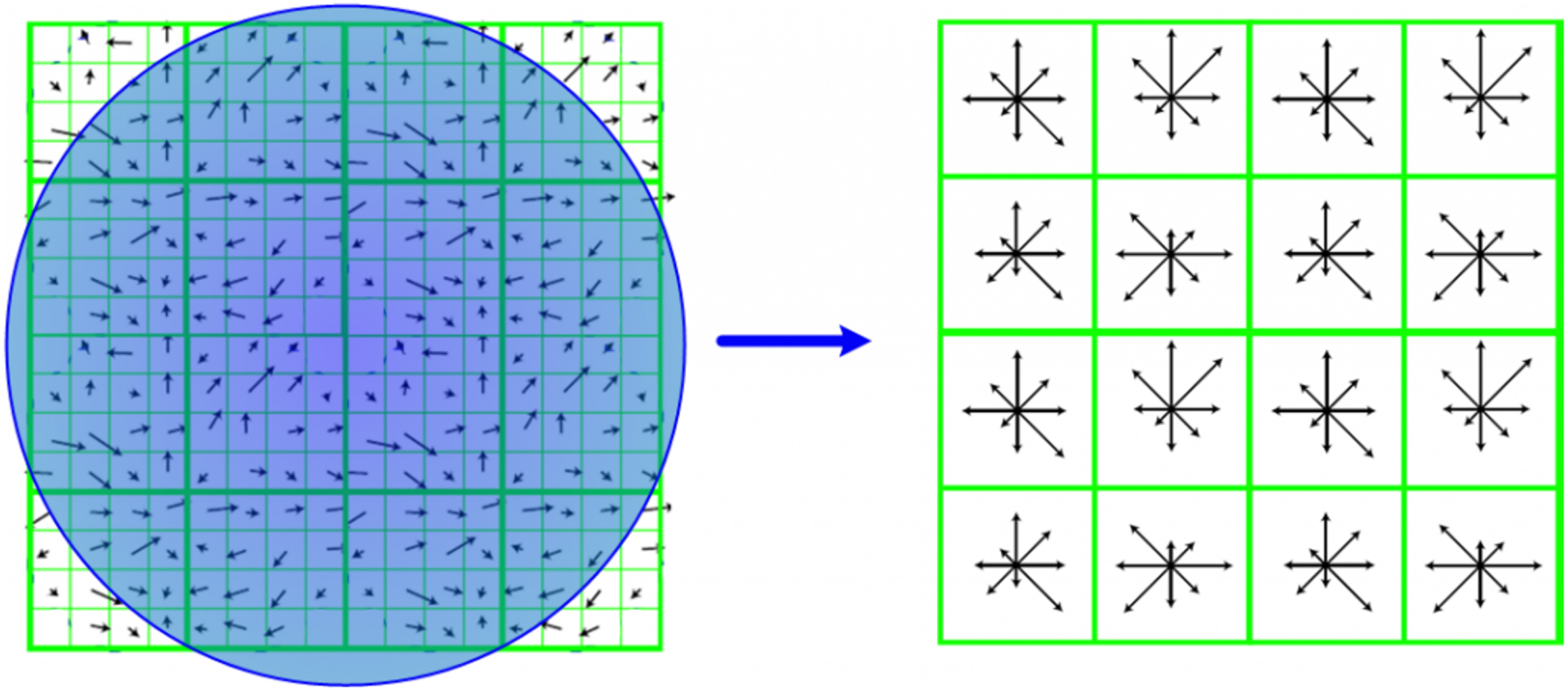
- La **Scale Invariant Feature Transform** (SIFT) è stata introdotta da Lowe (2004) come una combinazione
  - del DoG interest point/region detector e
  - del SIFT feature descriptor
- Queste due componenti sono state utilizzate in seguito anche come tecniche a sé stanti

SIFT Feature Descriptor

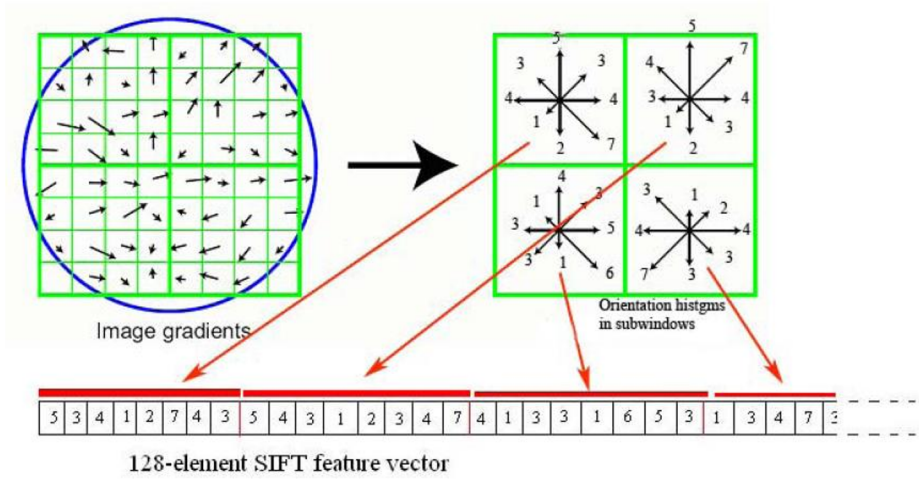


- Dopo che l'interest region è stata portata nell'orientazione canonica, viene costruita una griglia di 16x16 celle che copre l'intera interest region (per ottenere invarianza alla scala)
- Si determinano i gradienti in corrispondenza delle suddette celle e le rispettive orientazioni, dopo essere state pesate in base al rispettivo modulo e ad un kernel Gaussiano con $\sigma=r/2$, vengono inserite in una griglia 4x4 (avente grana più grossa della precedente) di istogrammi di orientazioni, ognuno dei quali è composto da 8 bin (associati alle 8 direzioni principali)

SIFT Feature Descriptor



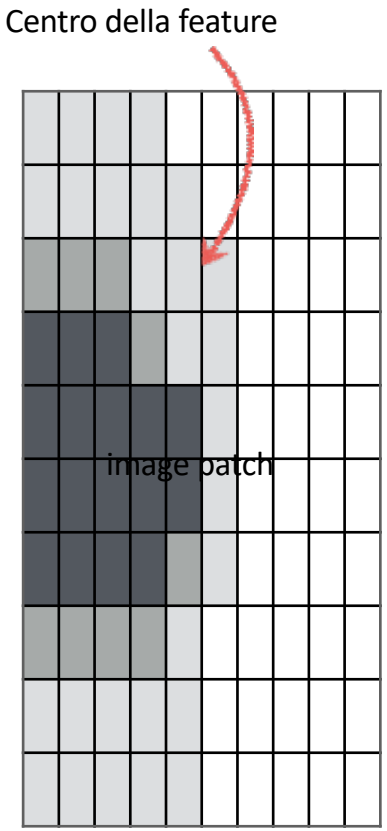
SIFT Feature Descriptor



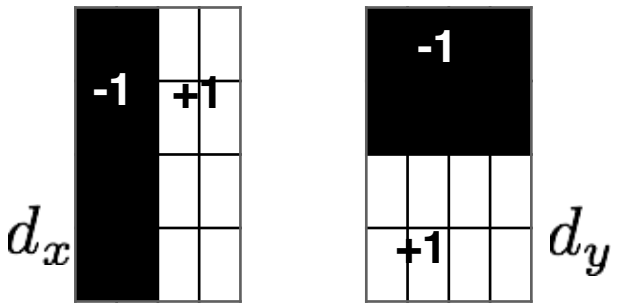
- Il SIFT descriptor (*localized set of gradient orientation histograms*) si ottiene concatenando i 16 istogrammi (4 x 4) da 8 bins, ottenendo un vettore a 4 x 4 x 8 = 128 dimensioni:
  - L'alta dimensionalità rende il descrittore distintivo
  - La suddivisione spaziale permette ai gradienti di subire piccoli spostamenti
- Per rendere il descrittore meno sensibile ai cambiamenti d'illuminazione il vettore viene normalizzato all'unità di lunghezza

SURF
(‘Speeded’ Up Robust Features)

Calcola il responso dei filtri Haar su ogni pixel nella patch



Haar wavelets filters

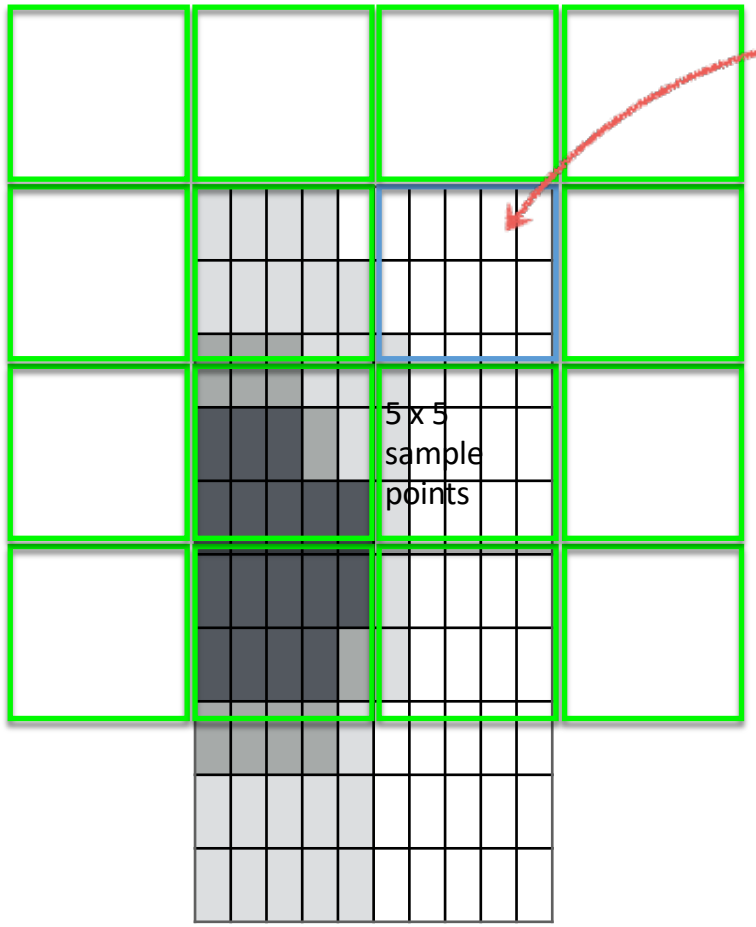


Con un peso gaussiano

https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html

SURF
(‘Speeded’ Up Robust Features)

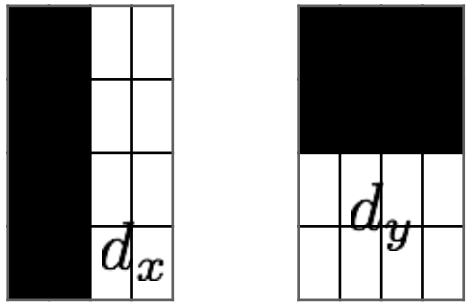
4x4 regioni su un intorno 20x20 del keypoint



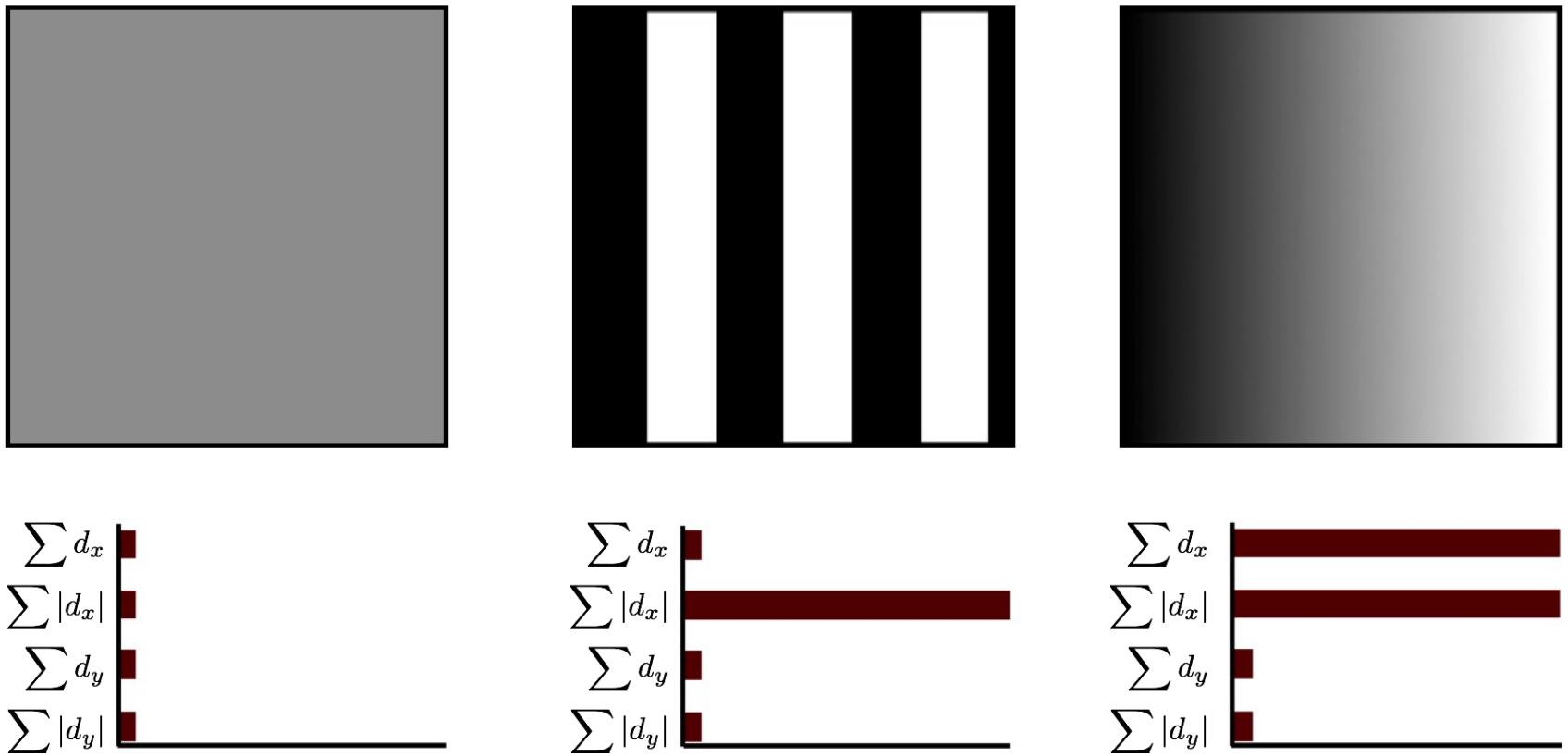
Ogni cella è rappresentata da 4 valori:

$$\left[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right]$$

Haar wavelets filters
(Gaussian weighted from center)



64 dimensioni in totale


$\sum d_x$
$\sum |d_x|$
$\sum d_y$
$\sum |d_y|$
$\sum d_x$
$\sum |d_x|$
$\sum d_y$
$\sum |d_y|$
$\sum d_x$
$\sum |d_x|$
$\sum d_y$
$\sum |d_y|$

Sommario: Image Features

- Largamente rimpiazzate dalle reti neurali
- Ancora utili da studiare
  - Confronti con le features calcolate dalle reti