Esempi di elaborazione di immagini

Marcello Calisti[1]

[1]The BioRobotics Institute
Scuola Superiore Sant'Anna

April 11th, 2013

Scopo delle lezioni

- Presentarvi alcune applicazioni di visione robotica
- Richiamare alcuni concetti di elaborazione di immagini (con alcune osservazioni e implementazioni)
- Introdurre alcuni strumenti nuovi rispetto alle precedenti lezioni

Sommario

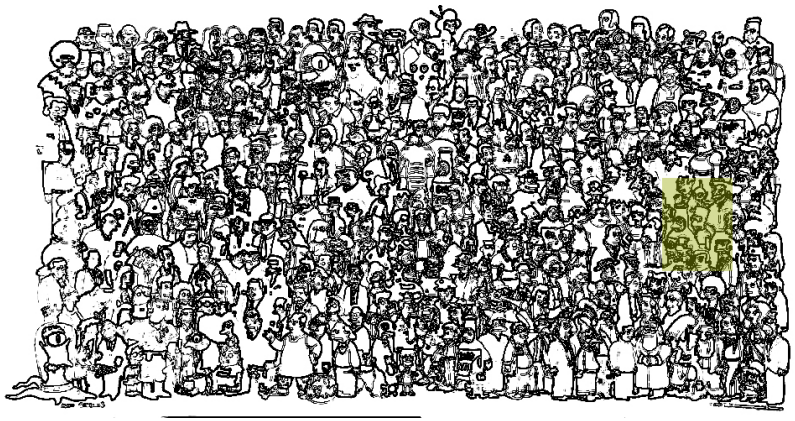
Io, modestamente, può!

Ho visto cose che voi umani...

anche i ricercatori sognano pecore elettriche?

Where is Bart?

Può essere facile per un computer…



- so esattamente cosa cercare
- (so anche come cercarlo)

Digita un nome…

Domanda

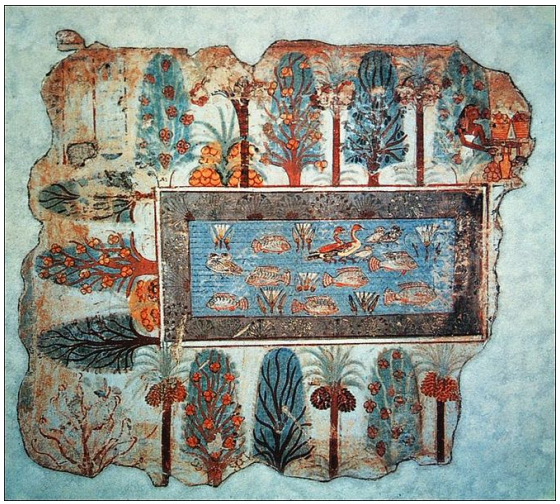
Un algoritmo per il riconoscimento di forme viene impiegato in una catena di montaggio di cerchioni per auto. L'algoritmo guida un braccio robotico riconoscendo la forma del cerchione e individuandone la posizione sul nastro trasportatore. Tutto funziona alla perfezione da Ottobre a Marzo, quando improvvisamente l'algoritmo non riconose più la forma. Cosa è successo?

SW & HW

Fatevi aiutare dall'hardware!

Spesso gli algoritmi di visione sfruttano conoscenze pregresse: forme, ambiente di lavoro, etc... Questo è ovviamente un limite per la generalità, ed è il problema dell'algoritmo dei cerchioni

La ricostruzione 3D è l'estrapolazione delle coordinate di un punto p nello spazio (x, y, z) date le coordinate bidimensionali del punto stesso $p = (u, v)$.



- misurazioni biometriche
- ricostruzioni di ambienti
- pose e posizioni di robot
- scopi forensi
- prestazioni di sportivi

La ricostruzione si compone generalmente di tre passi:

1. individuare il punto da ricostruire
   - automatica
   - semi-automatica/manuale
2. tracciare il punto (ottenere le coordinate 2d nel tempo)
   - siamo interessati all'andamento nel tempo?
   - dobbiamo usare la ricostruzione come feedback di controllo?
3. ricostruire il punto nello spazio
   - l'algoritmo di ricostruzione vero e proprio

Nel caso più semplice, le immagini sono statiche e l'estrazione del punto avviene manualmente: in questo caso $p = (x, y)$ è identificato immediatamente dalle coordinate dei pixel dell'immagine (u, v)

Estrazione automatica o semi-automatica

Marker

Sono degli elementi facilmente riconoscibili nella forma e che si distinguono in maniera evidente dallo sfondo per il colore. Esistono sia *attivi* che *passivi*. Possono essere verniciati o costituiti da materiale sensibile a una particolare emissione (infrarosso, calore, etc. . .).

Punti caratteristici

Si tratta di punti delle immagini che sono riconoscibili e definiti in maniera rigorosa ed efficace. Possono essere particolarità di immagini specifiche o caratteristiche come angoli, bordi, etc. . . .

Punti caratteristici: angoli

Vengono utilizzati per avere delle caratteristiche buone da seguire.
Punti di sella

Esempio di ricostruzione

Video: 

Esempi di marker:

- sferici o semisferici
- in strutture complesse (vertici di triangoli, poligoni)
- superfici deformabili

E' una buona immagine?

Pessima immagine così come è

Aumento del contrasto

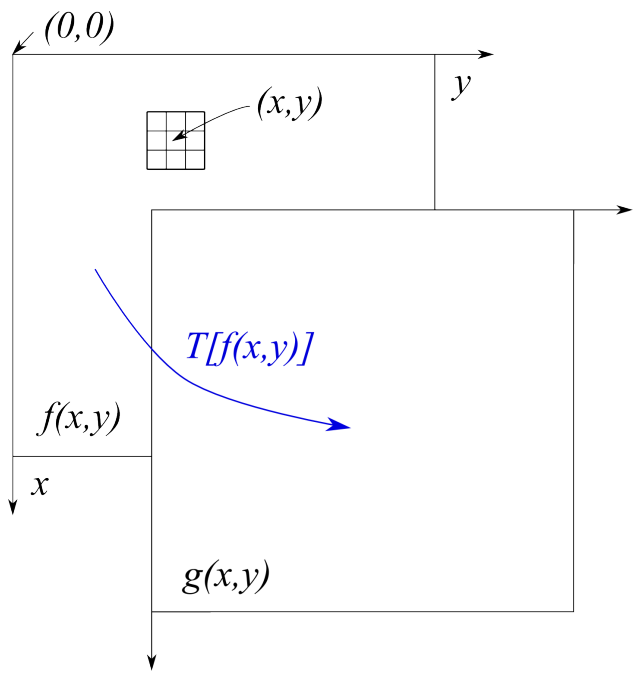
Non esiste una teoria generale sul miglioramento delle immagini. Quando una immagine è processata per aumentare certe caratteristiche, il risultato finale è l'unico giudice del processo stesso.

Quale trasformazione?

- trasformazioni logaritmiche
- trasformazioni sigmoidee
- trasformazioni lineari a tratti

Trasformazioni

$$g(x,y) = T[f(x,y)]$$



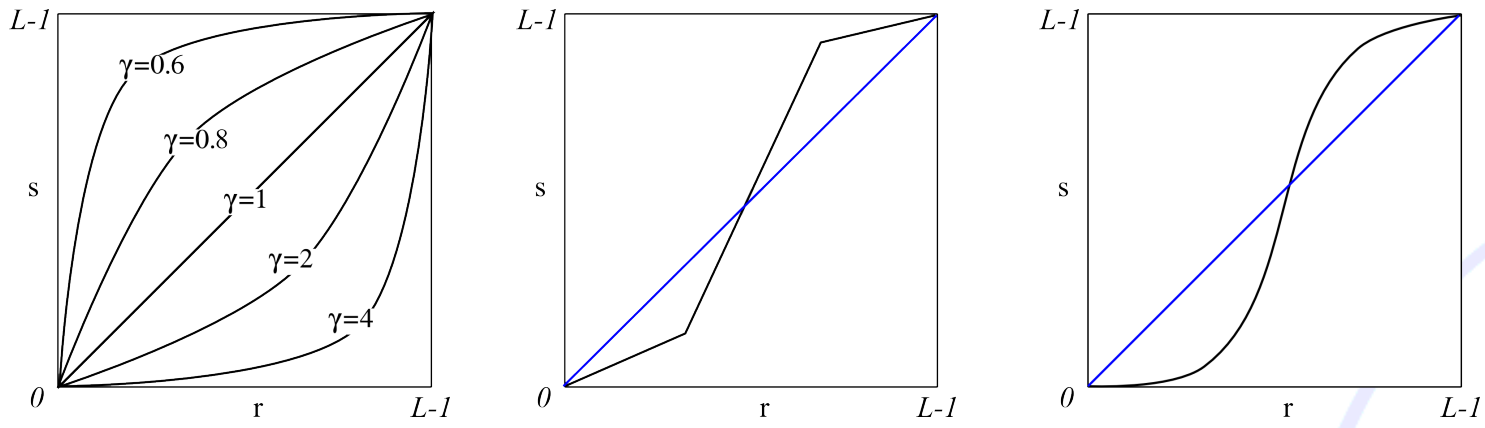
Trasformazioni

- puntuali
- locali
- globali

Definiamo per comodità $r = f(x, y)$ valore del pixel (x, y) per l'immagine f(x,y), mentre $s = g(x, y)$ valore del pixel (x, y) per l'immagine g(x,y). Per cui:

$$s = T[r]$$

con L livelli di grigio (o valori assumibili dai pixel).



(a) Potenza (b) Spezzata (c) Sigmoide

Figura: Transformazioni

Potenza schiarisce o scurisce

Spezzata flessibile

Sigmoide contrasta

$$s(r) = c \cdot r^{\gamma}$$

$$s(r) = \begin{cases} c_1 \cdot r & 0 \leq r < r_{min} \\ c_2 \cdot r & r_{min} \leq r < r_{max} \\ c_3 \cdot r & r_{max} \leq r < L-1 \end{cases}$$

$$s(r) = \frac{c_1}{1 + e^{-r}}$$

- potenza
- spezzata
- sigmoide

Ogni funzione ha alcuni parametri da definire

Alcune osservazioni

```
\%Power-Law Transformation: apply the power-law transform for
\%gamma = 0.05,0.2,0.67,1.5,2,5 and comment your results

f = imread('pout.tif');
[m n]=size(f);
c = 1;
y=input('Gamma value:');
for i=1:m
for j=1:n
s(i,j)=c*(f(i,j)^y);
end
end
figure,imshow(f),title('Original Image');
figure,imshow(s),title('After Power-Law Transformation');
```

$$\begin{aligned} L &= 21 \\ c &= 1 \\ \gamma &= 2 \\ s(r) &= c \cdot r^{\gamma} \end{aligned}$$

$$\begin{aligned} s(1) &= 1 \\ s(2) &= 2^2 = 4 \\ s(3) &= 3^2 = 9 \\ s(4) &= 4^2 = 16 \\ s(5) &= 5^2 = 25 \\ \ldots &= \ldots \\ s(20) &= 20^2 = 400 \end{aligned}$$

???

Abbiamo 21 livelli ma:

$$s(5) = 5^2 = 25$$

Riportare l'uscita tra $[0, L-1]$ è semplice:

$$\frac{s'}{s} = \frac{20}{400}$$

$$\frac{20}{400} = \frac{L-1}{(L-1)^{\gamma}} = (L-1)^{1-\gamma}$$

$$s' = (L-1)^{1-\gamma} s = c \cdot s = c \cdot r^{\gamma}$$

La scelta del parametro c dipende da L e γ.

Contrasto aumentato

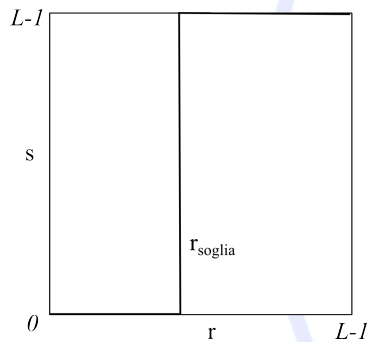
Normale

Contrastata

L'aumento del contrasto, è una pre-elaborazione propedeutica al prossimo passo:

- la binarizzazione (o sogliatura)

$$s(r) = \begin{cases} 0 & 0 \leq r < r_{soglia} \\ 255 & r_{soglia} \leq r < L-1 \end{cases}$$



Ma come si sceglie la soglia?

Soglia con o senza contrasto

Individuare le caratteristiche di interesse

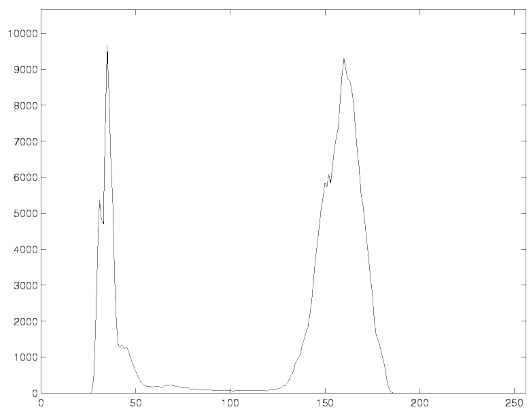
1. il contrasto distanzia le caratteristiche dallo sfondo
2. la soglia le separa

(a) Soglia

(b) Contrasto–soglia

Lo scopo generale della sogliatura è separare le caratteristiche di interesse dallo sfondo.

La procedure di sogliatura Otsu massimizza la varianza tra le classi. Inoltre, assunto un istogramma bimodale, Otsu ha dimostrato che la massimizzazione tra classe equivale alla minimizzazione inter-classe.



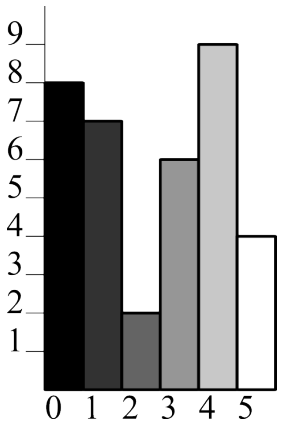
Local threshold procedures, such Niblack, cope with global gradient but identify several false positive.

Senza entrare nei dettagli matematici, un esempio di implementazione è dato da:

1. Ricavare l'istogramma dell'immagine
2. Calcolare, per ogni valore di soglia $t = 0, ..., L-1$ le seguenti variabili:
3.

$$
\begin{aligned}
{\mu_s}^t &= \frac{\sum_{i=0}^{t} \#(i) \cdot i}{\#s} \\
{\mu_o}^t &= \frac{\sum_{i=t+1}^{L-1} \#(i) \cdot i}{\#o} \\
W_s &= \frac{\#s}{\#s + \#o} \\
W_o &= \frac{\#o}{\#s + \#o} \\
\sigma_b^2 &= W_s W_o ({\mu_s}^t - {\mu_o}^t)^2
\end{aligned}
$$

4. Il massimo $\sigma_b^2(t)$ individua la soglia t esatta.



Per $t = 0$:

$$
\begin{aligned}
W_s &= \frac{8}{36} = 0.22 \\
\mu_s &= \frac{8 \cdot 0}{8} = 0 \\
W_o &= \frac{7+2+6+9+4}{36} = \frac{28}{36} = 0.78 \\
\mu_o &= \frac{7 \cdot 1 + 2 \cdot 2 + 6 \cdot 3 + 9 \cdot 4 + 4 \cdot 5}{28} = 3.04 \\
\sigma_b^2 &= 0.22 \cdot 0.78(3.04 - 0)^2 = 1.59
\end{aligned}
$$

Soglia	t=0	t=1	t=2	t=3	t=4
Varianza	1.59	2.56	2.63	2.14	0.87

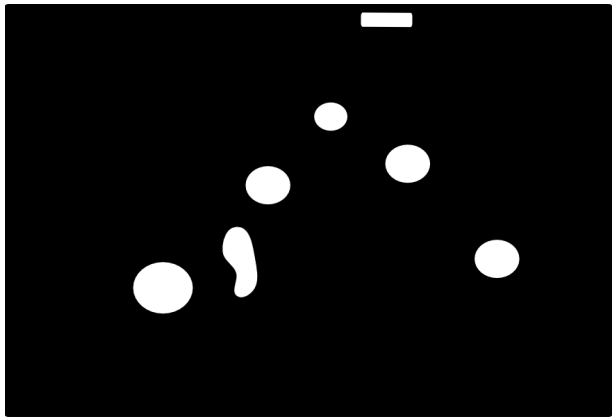
L'immagine binarizzata evidenzia i marker che ci interessano.

Contrasto allarga l'istogramma, escludendo una parte degli artefatti

Sogliatura identifica alcuni dei possibili elementi di interessi

? separazione degli elementi corretti rispetto agli ultimi artefatti

Zoom sugli elementi



1. forma dei marker (circolari)
2. dimensione

Il riconoscimento di forma è molto comune in computer vision:

- trasformata di Hough
- Gradiente
- Erosione

Hough in breve

Trasforma l'immagine originale in una matrice di accumulazione nel piano dei parametri.
Ogni punto che appartiene alla stessa curva di ricerca (retta, circonferenza, ...) aumenta l'accumulazione.

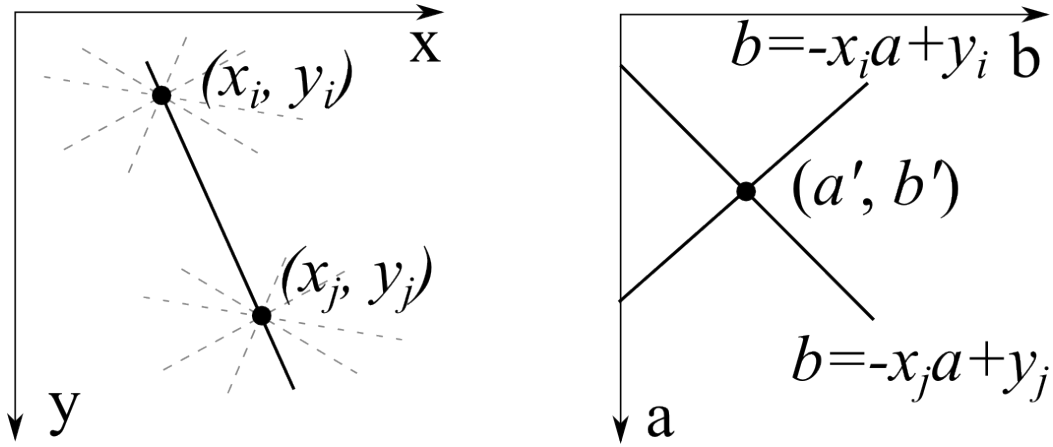
Alcuni svantaggi

- elevato costo computazionale
- richiede forme perfette o genera molti falsi positivi

Alcune osservazioni

$$y = a \cdot x + b \rightarrow b = -x_i \cdot a + y_i$$

La retta $b = -x_i \cdot a + y_i$ nel piano dei parametri rappresenta tutte le rette passanti per un generico punto (x_i, y_i).



Ogni punto appartenente alla stessa retta aumenta l'accumunlazione (a', b').

Posso fare Hough sull'immagine originale?

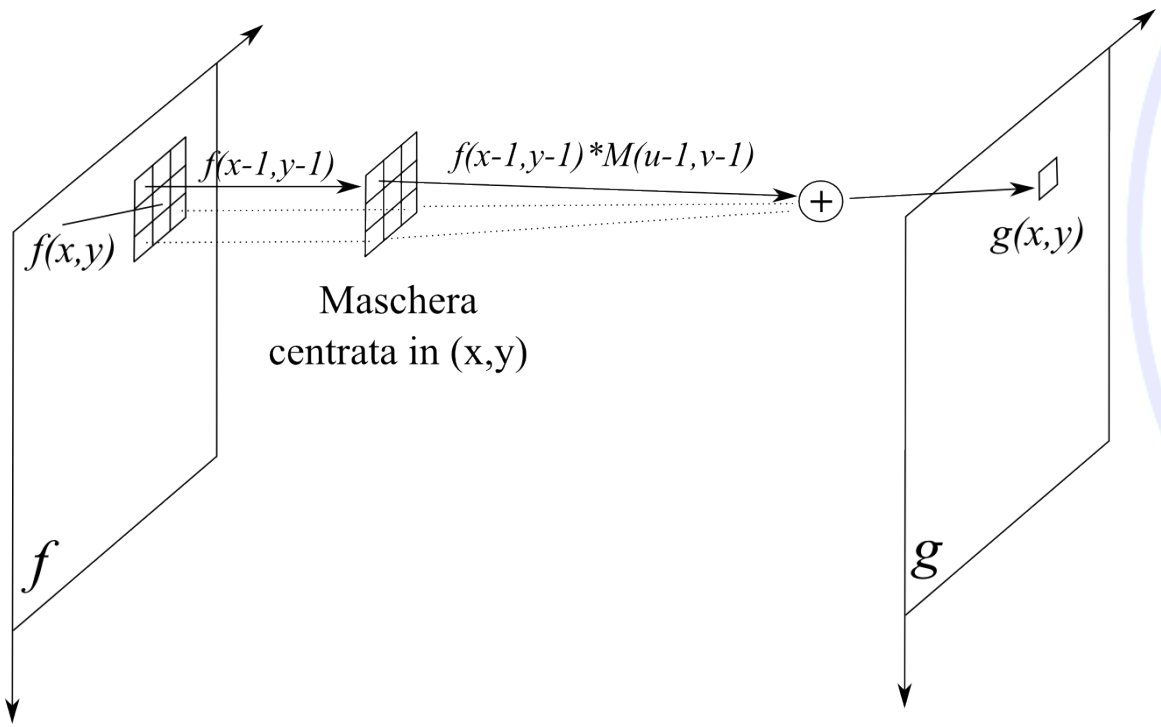
- L'immagine deve essere binarizzata
- Accumulerei ovunque!
- Devo avere i bordi degli oggetti!

Individuazione dei bordi:

1. gradiente e soglia
2. laplaciano
3. snakes
4. ...

Convoluzione

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} f(x+s, y+t) \cdot M(u+s, v+t)$$



- La convoluzione è un'operatore *locale*, a differenza delle curve di trasformazione viste finora che operavano a livello *puntuale*.
- La grandezza della maschera di convoluzione varia in base allo scopo che si vuole ottenere, in particolare dipende dalle dimensioni delle caratteristiche dell'immagine che si vogliono evidenziare (o rimuovere).
- Ovviamente, la degenerazione della maschera alla dimensione di un pixel, riduce la convoluzione a una operazione puntuale.
- In base ai valori assunti dalle maschere, diverse caratteristiche sono evidenziate.

Gradiente

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

Il modulo è dato da:

$$G = \sqrt{G_x^2 + G_y^2}$$

Implementazione di Sobel (ce ne sono altre: Prewit, Robert, ...)

G_x=

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

G_y=

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

Laplaciano

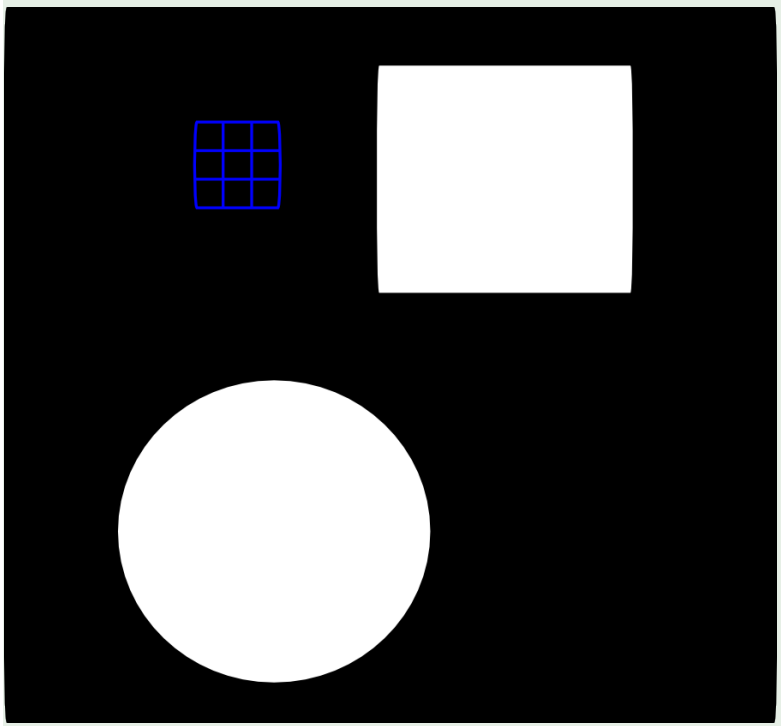
Analitica

$$\nabla^2 = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Discreta (una delle varie possibili)

0	-1	0
-1	4	-1
0	-1	0

Example



Finestra dell'immagine:

$$f(x,y) = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$$

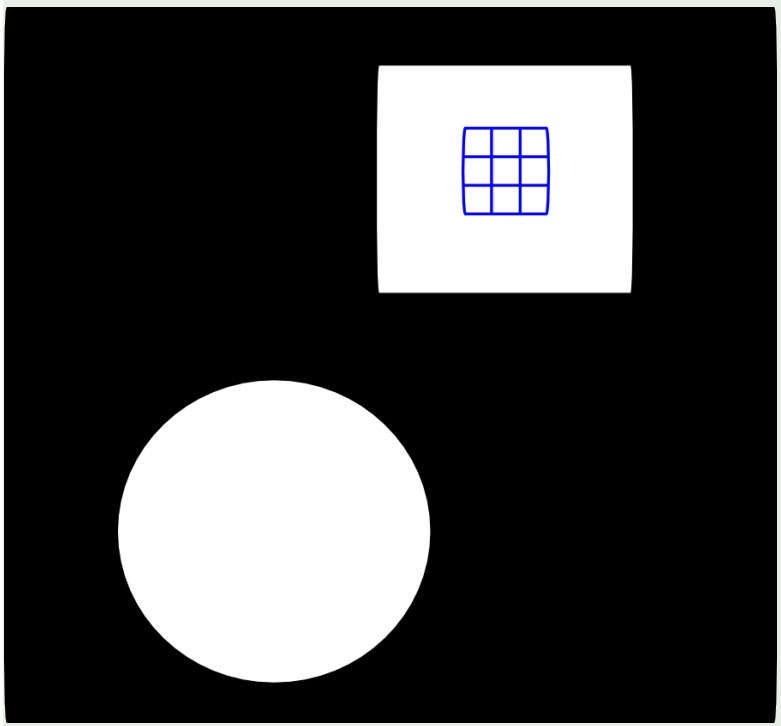
$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

$$G_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

I prodotti sono:

$$\nabla^2 \otimes f(x,y) = 0$$
$$G_x \otimes f(x,y) = 0$$

Example



Finestra dell'immagine:

$$f(x,y) = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

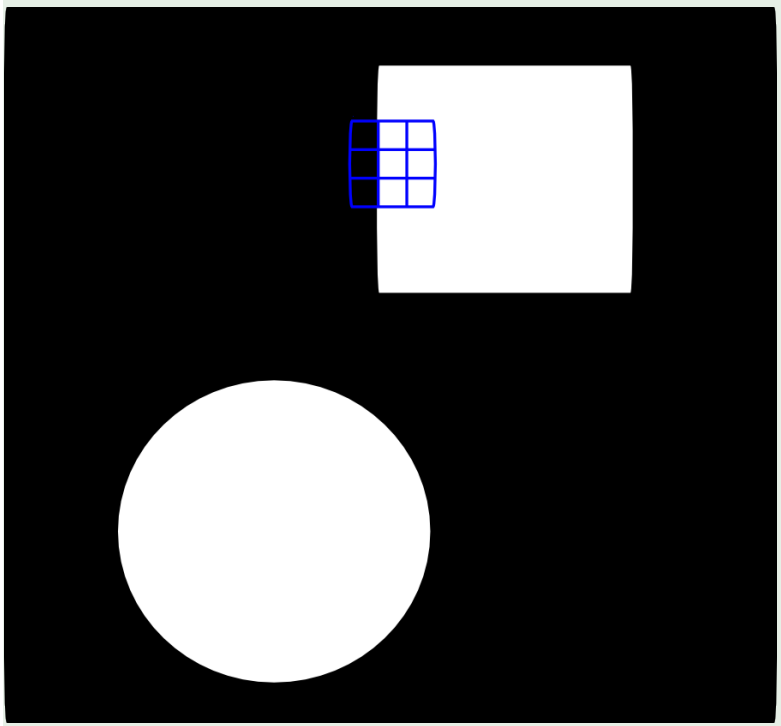
$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

$$G_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

I prodotti sono:

$$\nabla^2 \otimes f(x,y) = 0$$
$$G_x \otimes f(x,y) = 0$$

Example



Finestra dell'immagine:

$$f(x,y) = \begin{array}{|c|c|c|}\hline 0 & 1 & 1 \\ \hline 0 & 1 & 1 \\ \hline 0 & 1 & 1 \\ \hline \end{array}$$

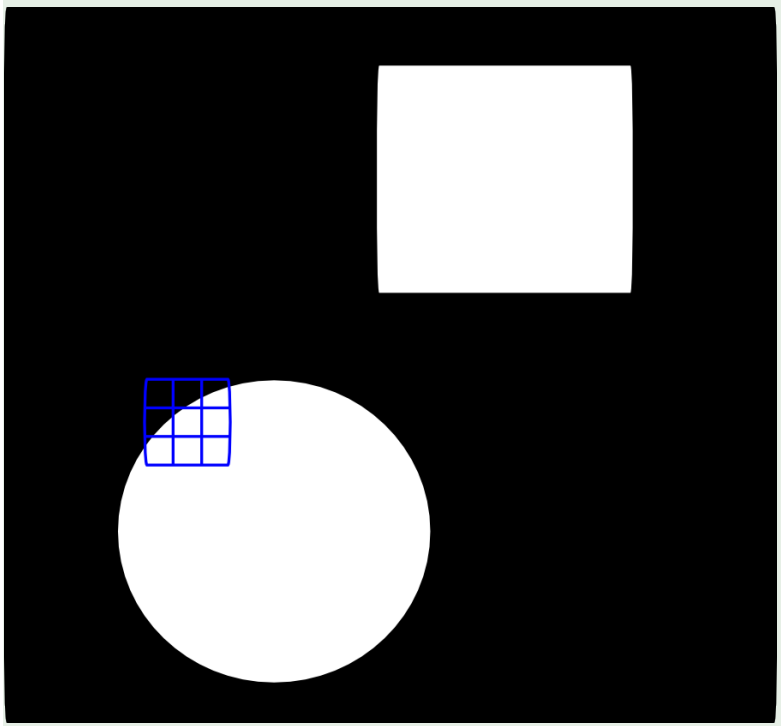
$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

$$G_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

I prodotti sono:

$$\nabla^2 \otimes f(x,y) = 1$$
$$G_x \otimes f(x,y) = 4$$

Example



Finestra dell'immagine:

$$f(x,y) = \begin{array}{|c|c|c|}\hline 0 & 0 & 1 \\ \hline 0 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

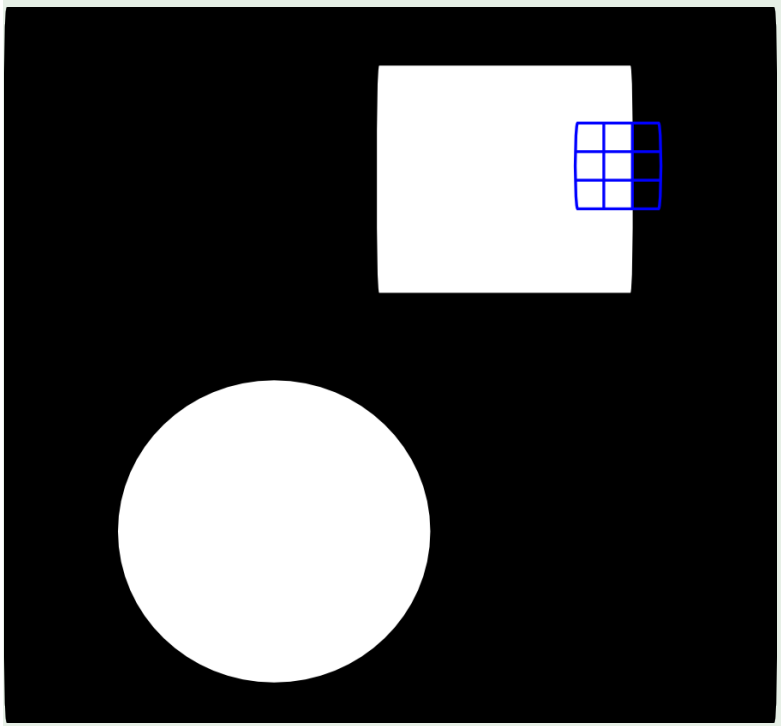
$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

$$G_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

I prodotti sono:

$$\nabla^2 \otimes f(x,y) = 2$$
$$G_x \otimes f(x,y) = 3$$

Example



Finestra dell'immagine:

$$f(x,y) = \begin{array}{|c|c|c|}\hline 1 & 1 & 0 \\ \hline 1 & 1 & 0 \\ \hline 1 & 1 & 0 \\ \hline \end{array}$$

$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

$$G_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

I prodotti sono:

$$\nabla^2 \otimes f(x,y) = 1$$
$$G_x \otimes f(x,y) = -4$$

...e per casa

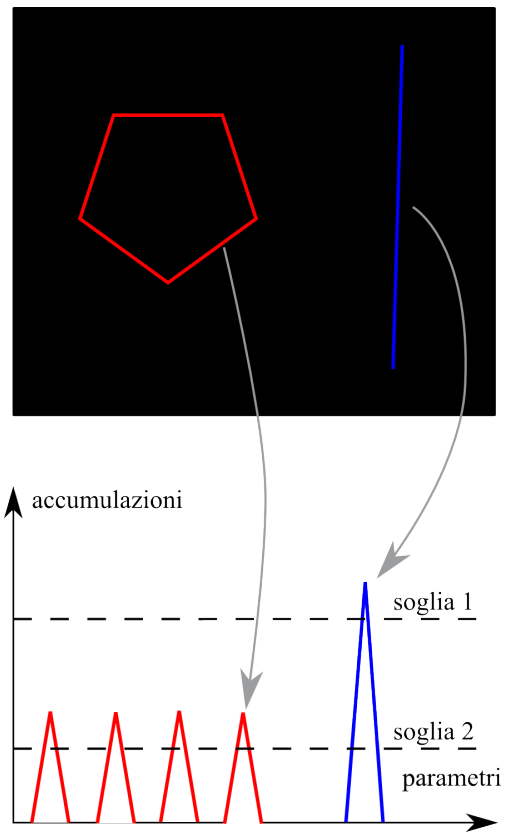
Cosa succede se l'immagine non è binarizzata?

Il nostro problema era applicare Hough.

Immagine binarizzata $\rightarrow$ Contorni individuati

Ora possiamo applicare la trasformata ai punti del contorno.

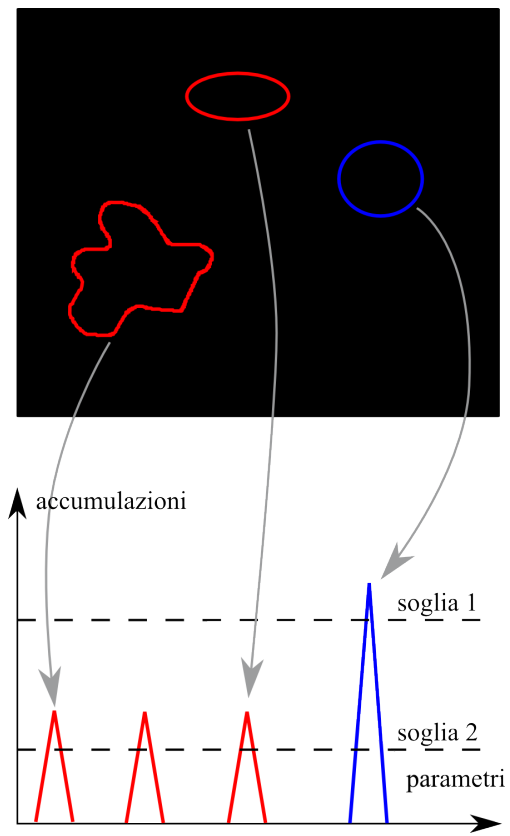
Rombo e retta



- dipendente dalla lunghezza della retta, l'accumulazione ha valori più o meno bassi
- va scelta una soglia che mi separi gli elementi che non desidero dalle caratteristiche da identificare

In questo caso, basta abbassare la soglia e comunque individuerò tutte rette.

Elissi e cerchi



- stesso discorso per il cerchio
- la differenza sta in artefatti della dimensione del raggio di ricerca

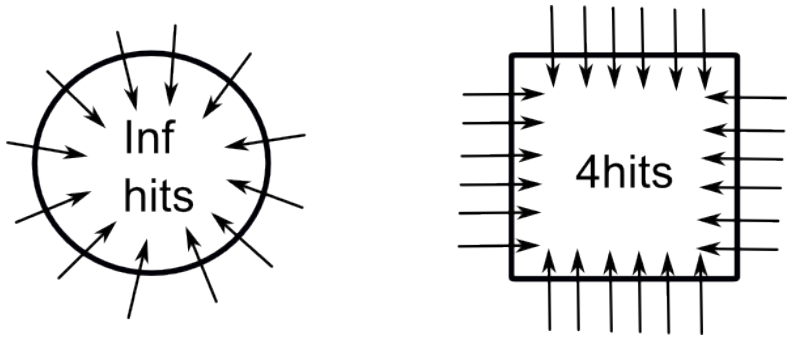
Usiamo il gradiente

Abbiamo detto che il gradiente può essere usato per individuare i bordi delle immagini.
In realtà, il gradiente ci fornisce anche informazioni aggiuntive:

- intensità (modulo)
- direzione

Perciò il gradiente caratterizza una forma e anche la velocità della variazione dei livelli di grigio

Applichiamo ancora il concetto dell'accumulazione:



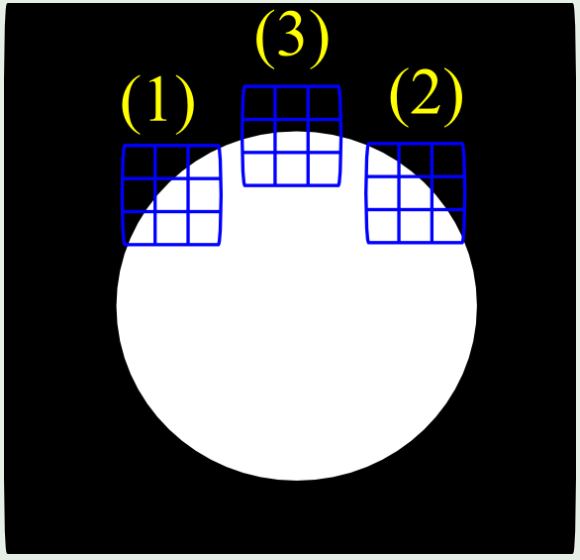
Accumuliamo punti in direzione del gradiente, tracciando delle rette dal bordo.

Implementazione

Valutando tutto nel punto di bordo (x,y)

$$y = a \cdot x + b \qquad a = \frac{G_y}{G_x}$$

Example

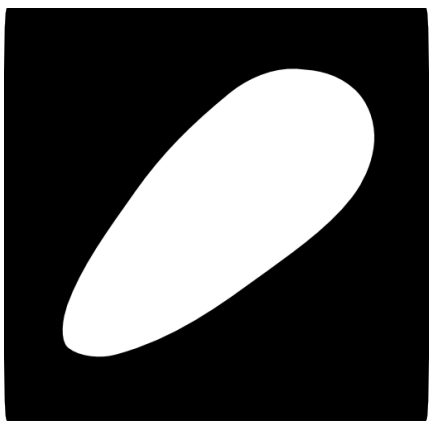


$$
\begin{aligned}
y &= a \cdot (x - x_0) + b \\
a &= \frac{G_y}{G_x} \\
b &= (x_0, y_0)
\end{aligned}
$$

1. $G_x = 3; G_y = 3; a = 1;$
2. $G_x = -3; G_y = 3; a = -1;$
3. $G_x = 0; G_y = 4; a = \infty;$

Forme riconosciute

- Cerchi
- Ellissi
- Elementi a goccia



Sono molto comuni quando il soggetto è veloce e l'acquisizione è lenta

Usiamo l'erosione

L'erosione fa parte di una procedura più generale chiamata processazione morfologica delle immagini.

Questa sfrutta i concetti di morfologia matematica, ed è strettamente legata all'insiemistica

Qui accenniamo solo i concetti essenziali per comprendere l'erosione, usata nel contesto dell'assottigliamento

Alcune notazioni

Sia A un insieme in Z^2 (interi 2 dimensioni).

Definition

Se $a = (a_1, a_2)$ è un elemento di A, allora possiamo scrivere:

$$a \in A$$

Definition

e analogamente se a non è un elemento di A:

$$a \notin A$$

Un insieme di elementi è definito dalle parentesi graffe $\{\cdot\}$. Per noi gli elementi degli insiemi sono le coordinate di quei pixel che fanno parte di un oggetto dell'immagine o di una zona di interesse. Per esempio, quando scriviamo:

Example

$$C = \{w | w = -d, \ per \ d \in D\}$$

intendiamo dire che C è composto da elementi w tali che w sono ottenuti moltiplicando per -1 le due coordinate degli elementi d di D.

Definition

Se ogni elemento di A è anche elemento di B, allora si dice che A è un sottoinsieme di B e si scrive:

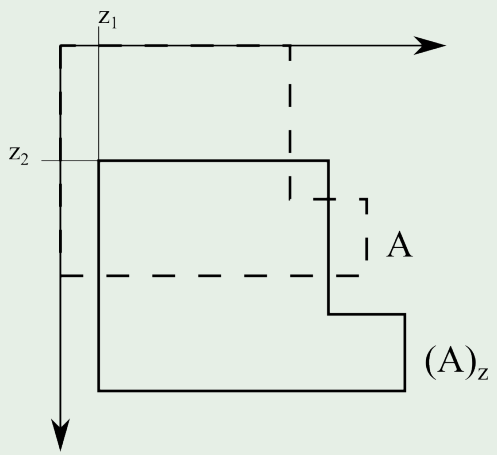
$$A \subseteq B$$

Definition

La *traslazione* di un insieme A di elemento $z = (z_1, z_2)$, indicato come $(A)_z$, è definita come:

$$(A)_z = \{c | c = a + z, \ per \ a \in A\}$$

Example



Ora che abbiamo tutti i mattoncini che ci servono, è immediato dire che...

Definition

$$A \ominus B = \{z | (B)_z \subseteq A\}$$

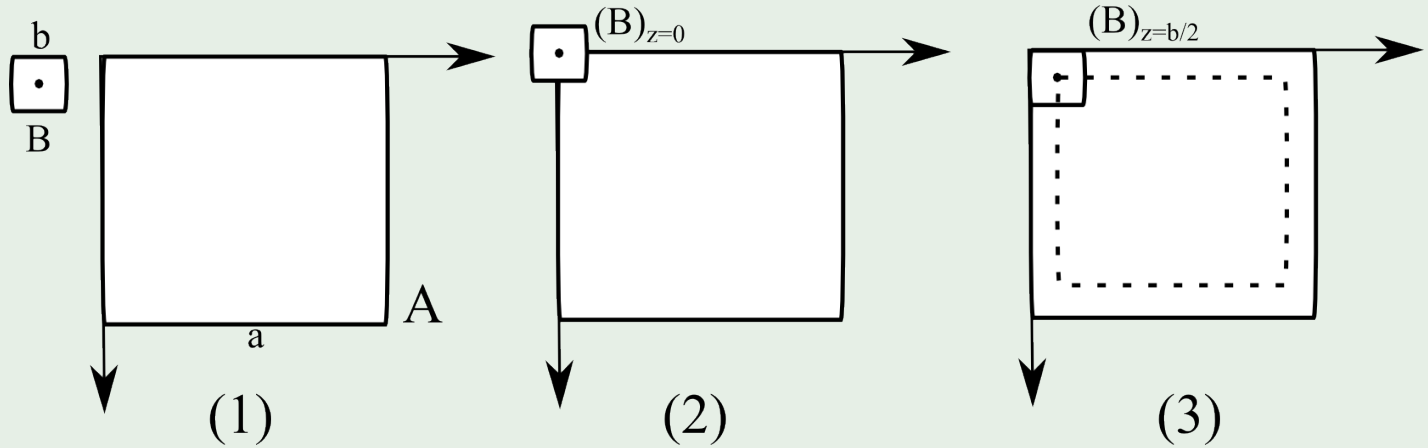
...questa definizione rappresenta una:

erosione

In parole, l'equazione indica che l'erosione di A attraverso B è l'insieme di tutti i punti z tali che la traslazione di B è un sottoinsieme di A.

Verifichiamo

Example



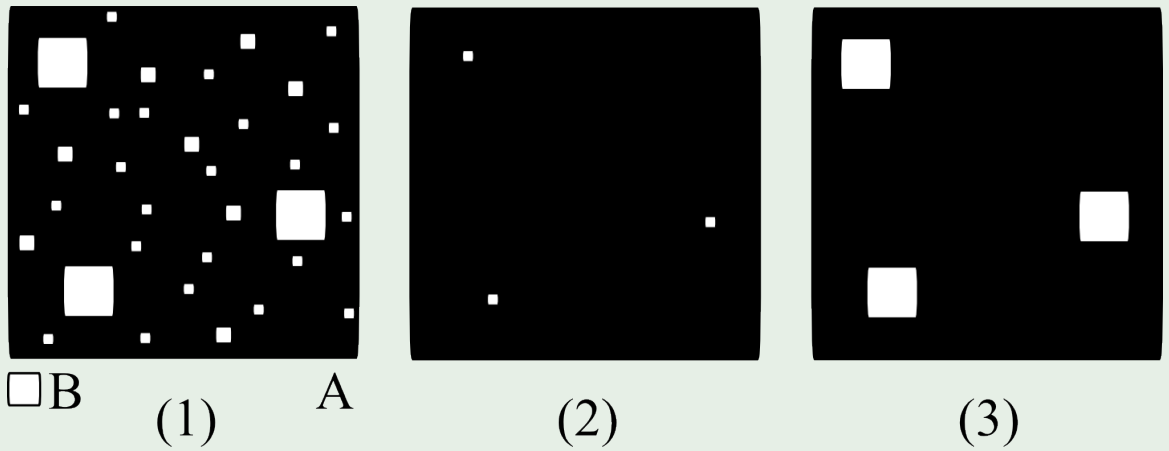
1. I due insiemi A e B.
2. $(B)_0 \subseteq A$? No
3. $(B)_0 \subseteq A$? Sì

L'insieme eroso sarà costituito da:

$$A \ominus B = [\frac{b}{2} : a - \frac{b}{2}; \frac{b}{2} : a - \frac{b}{2}]$$

Possiamo vedere l'erosione come una rimozione della parte più esterna dell'oggetto, di forma B.

Example



1. Immagine originale.
2. Erosione con insieme quadrato B.
3. Dilatazione

Alcuni vantaggi

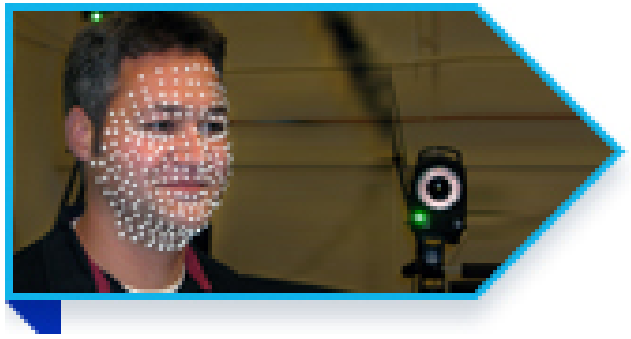
- Non necessita della ricerca dei bordi
- Elimina tutti i piccoli artefatti

Alcuni limiti

- Gli oggetti devono essere più grandi degli artefatti
- Devo stimare correttamente la maschera di erosione

A che punto siamo:

- Abbiamo contrastato l'immagine
- Abbiamo binarizzato l'immagine
- Abbiamo individuato le coordinate di interesse
- Dobbiamo trasformare i punti 2d p=(x,y) nei punti 3d p=(x,y,z)

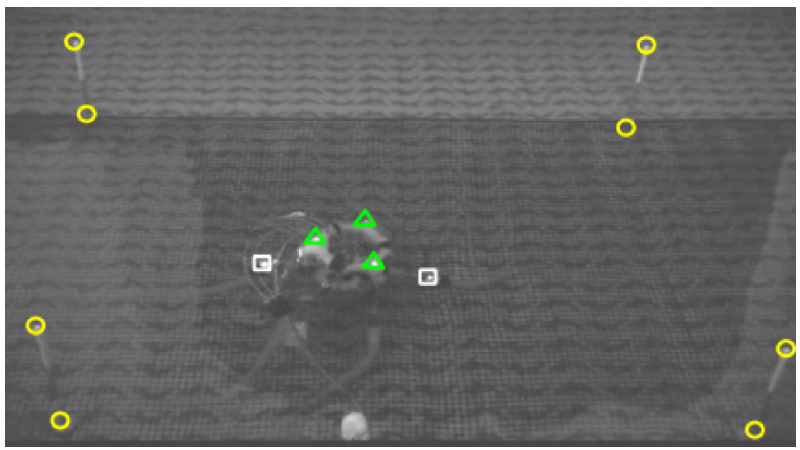


Ma prima...

Un altro po' di esempi...

Quale trasformazione è stata usata?

Per la pubblicazione, la lettura dell'immagine è lo scopo principale del processo di miglioramento.

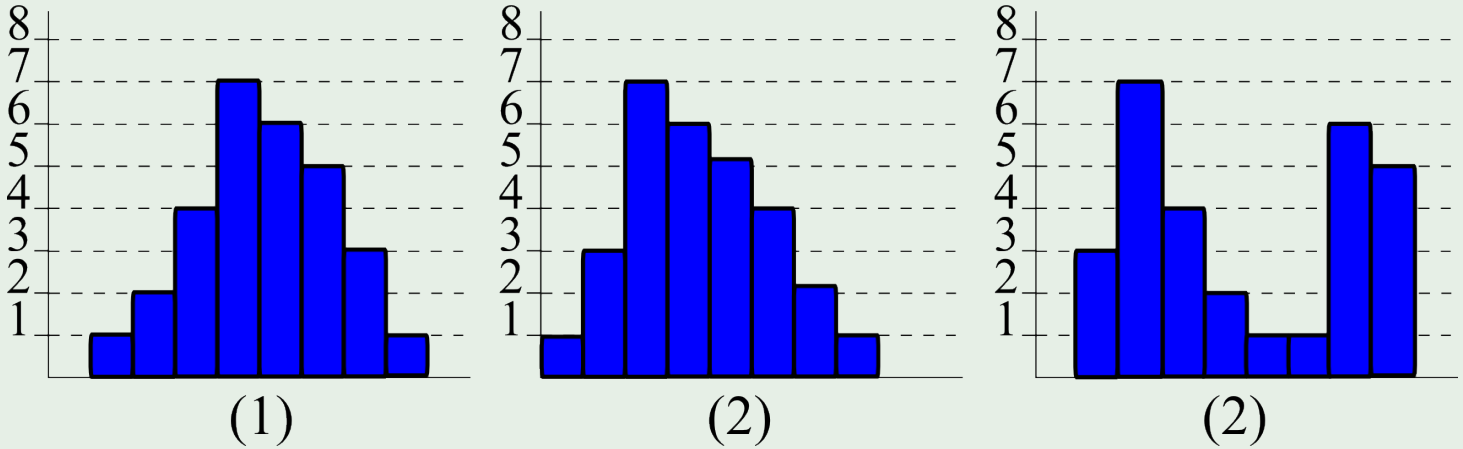


Quale trasformazione?

- trasformazioni di potenza
  - $\gamma > 1$
  - $\gamma = 1$
  - $\gamma < 1$
- trasformazioni sigmoidee
- trasformazioni lineari a tratti

A quale immagine si associano gli istogrammi

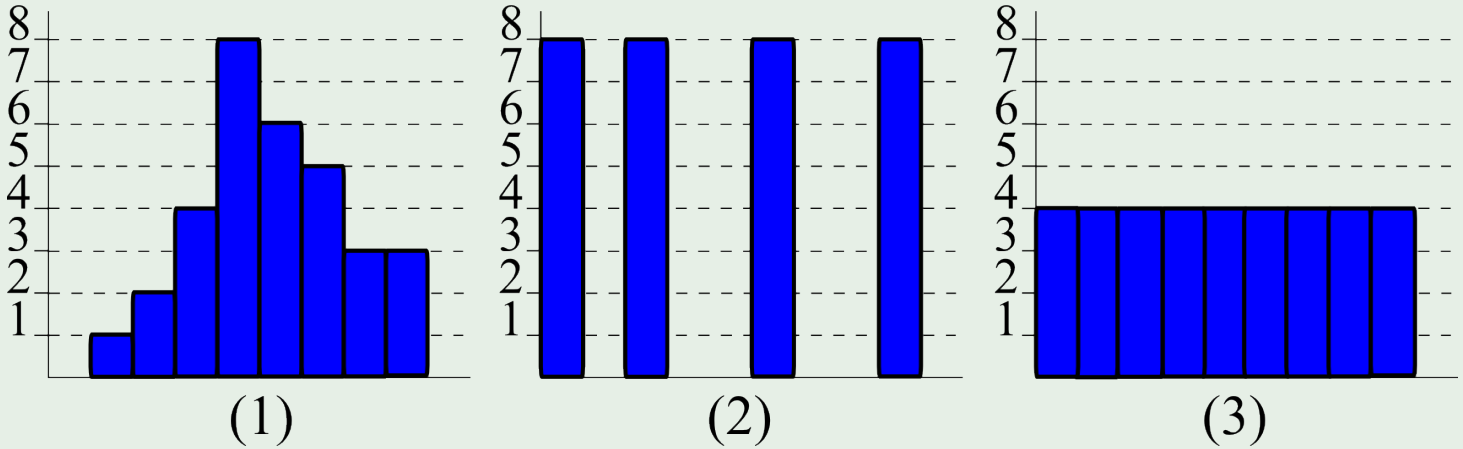
Example



1. Immagine originale
2. Immagine scurita
3. Immagine con contrasto aumentato

Quale istogramma è impossibile?

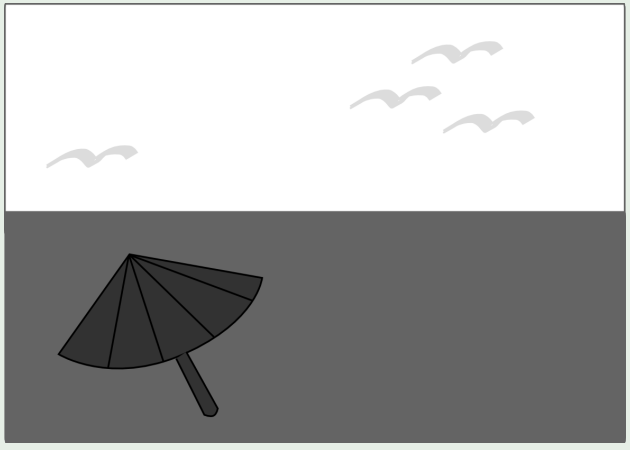
Example

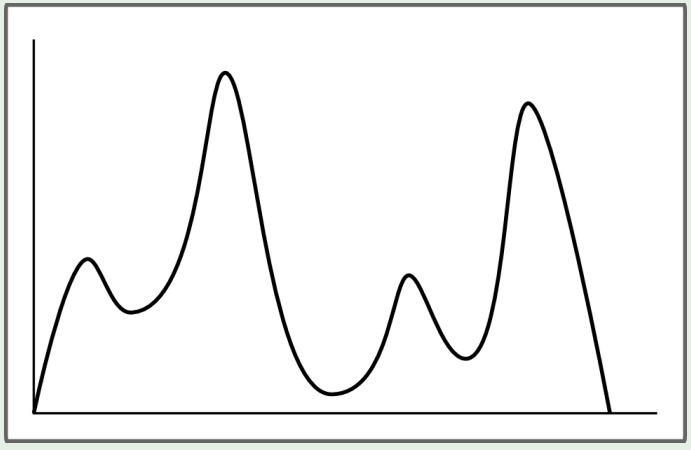


1. Istogramma originale
2. Istogramma numero 2
3. Istogramma numero 3

Più oggetti di interesse

Example



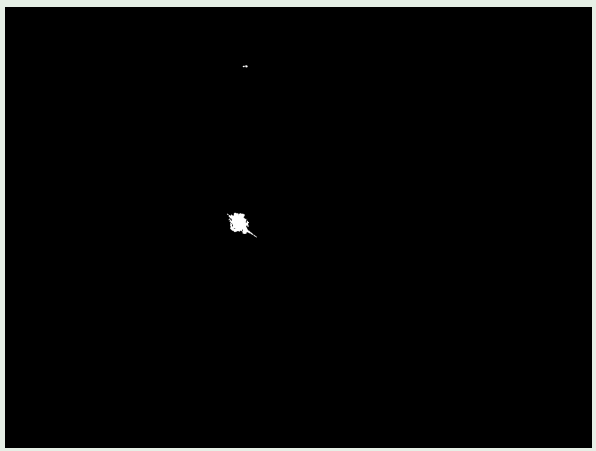


Come separo (automaticamente) tutti gli elementi?

- Reitero Otsu da 0 al livello soglia, e dal livello soglia al livello massimo!
- Opero la sogliatura localmente!

Marker con aloni

Example



Quale procedura?

- Hough
- Gradiente
- Erosione

Regioni di interesse (ROI)

Abbiamo detto che immagini con grossi artefatti non sono adatte per essere l'erosione...

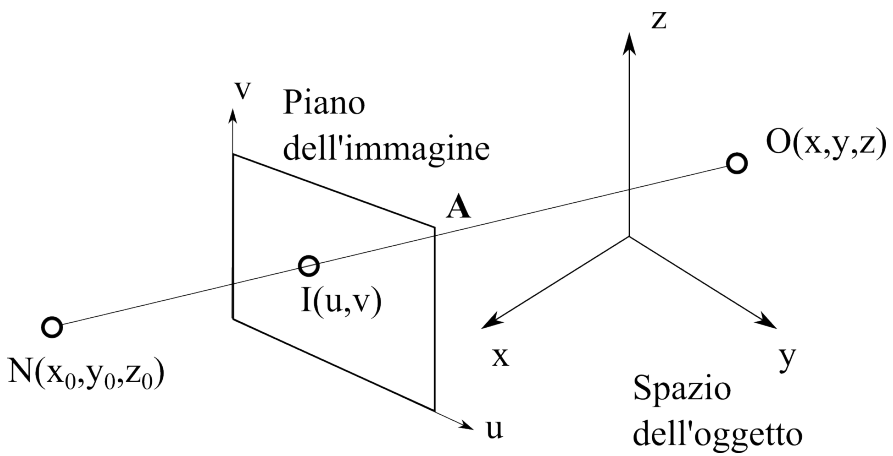
Come si può usare l'erosione su questa immagine?
Se sono sicuro che l'artefatto è fuori della mia regione di interesse (spazio dove si muove il robot, struttura anatomica in cui non ho una certa malattia,...) allora posso restringere il campo di applicazione

Direct Linear Transformation

E' una procedura molto usata per la ricostruzione tridimensionale, che ipotizza la linearità tra l'oggetto visto e la sua immagine proiettata sul piano di registrazione.

Necessita l'utilizzo di più telecamere e di alcuni punti di controllo nello spazio di lavoro

Consideriamo l'oggetto *O* mappato direttamente sull'immagine proiettata *I*.



- N centro di proiezione
- O oggetto
- I immagine proiettata

Spazio dell'oggetto

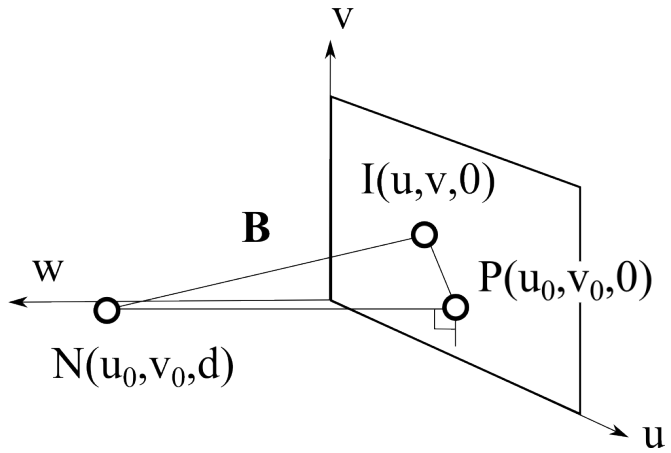
Nello spazio dell'oggetto, il centro di proiezione è:

$$N = (x_0, y_0, z_0)$$

e il vettore da N a O è perciò:

$$\mathbf{A} = (x - x_0, y - y_0, z - z_0)$$

Ma possiamo descrivere *N* anche rispetto al piano dell'immagine



P punto principale

d distanza principale

Dove la linea congiungente N e P è parallela all'asse *w* e perpendicolare al piano *uv*.

Collinearità

Il vettore **B** è quindi descritto da:

$$\mathbf{B} = (u - u_0, v - v_0, -d)$$

Per l'ipotesi di collinearità, cioè l'ipotesi che sia una linea retta a congiungere l'oggetto al centro di proiezione:

$$\mathbf{B} = c \cdot \mathbf{A}$$

dove c è semplicemente uno scalare.

Notiamo che **B** è descritto rispetto il piano dell'immagine, mentre **A** rispetto al spazio dell'oggetto. Devo descriverli in un unico riferimento.

Trasformiamo **A** al riferimento piano-immagine:

$$\mathbf{A}^i = \mathbf{T}_{i/o}\mathbf{A}^o = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \mathbf{A}^o \quad (1)$$

dove $\mathbf{A}^i$ è **A** rispetto al piano-immagine, mentre $\mathbf{A}^o$ rispetto allo spazio-oggetto.
$\mathbf{T}_{i/o}$ è la trasformata che porta **A** da un riferimento all'altro. In pratica noi non sappiamo a priori $\mathbf{T}_{i/o}$ e inoltre vedremo come questa ha un ruolo centrale per lo scopo finale di ricostruire, dalle nostre coordinate 2D (u,v) le posizioni 3d (x,y,z) dell'oggetto.

Sostituiamo in (1) le espressioni di **B** e $\mathbf{A}^o$:

$$\begin{bmatrix} u-u_0 \\ v-v_0 \\ -d \end{bmatrix} = c \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{bmatrix} \quad (2)$$

Svolgendo il prodotto:

$$\begin{aligned} u-u_0 &= c[r_{11}(x-x_0)+r_{12}(y-y_0)+r_{13}(z-z_0)] \\ v-v_0 &= c[r_{21}(x-x_0)+r_{22}(y-y_0)+r_{23}(z-z_0)] \\ -d &= c[r_{31}(x-x_0)+r_{32}(y-y_0)+r_{33}(z-z_0)] \end{aligned} \quad (3)$$

Possiamo quindi derivare c da (3):

$$c = \frac{-d}{r_{31}(x - x_0) + r_{32}(y - y_0) + r_{33}(z - z_0)} \tag{4}$$

Combinando (4) e (3):

$$\begin{aligned} u - u_0 &= -d \frac{r_{11}(x - x_0) + r_{12}(y - y_0) + r_{13}(z - z_0)}{r_{31}(x - x_0) + r_{32}(y - y_0) + r_{33}(z - z_0)} \\ v - v_0 &= -d \frac{r_{21}(x - x_0) + r_{22}(y - y_0) + r_{23}(z - z_0)}{r_{31}(x - x_0) + r_{32}(y - y_0) + r_{33}(z - z_0)} \end{aligned} \tag{5}$$

Equazione DLT standard

L'equazione (5) può essere convenientemente riarrangiata in:

$$u = \frac{L_1 x + L_2 y + L_3 z + L_4}{L_9 x + L_{10} y + L_{11} z + 1}$$
$$v = \frac{L_5 x + L_6 y + L_7 z + L_8}{L_9 x + L_{10} y + L_{11} z + 1} \quad (6)$$

Dove i coefficienti $L_1 \cdots L_{11}$ sono chiamati parametri della DLT.

Equazione DLT standard

Variabili

- u e v sono le coordinate nel piano-immagine del punto da ricostruire
- x,y e z sono le coordinate nello spazio oggetto del punto da ricostruire

Parametri

- $L_1 \cdots L_{11}$ sono parametri che dipendono dal sistema di acquisizione, e rimangono fissi una volta derivati
- la derivazione di questi parametri si chiama calibrazione

Calibrazione

La calibrazione viene realizzata riarrangiando (6) in:

$$\begin{aligned}\frac{1}{R}u &= \frac{1}{R}(L_1x + L_2y + L_3z + L_4 - L_9ux - L_{10}uy - L_{11}uz)\\ \frac{1}{R}v &= \frac{1}{R}(L_1x + L_2y + L_3z + L_4 - L_9vx - L_{10}vy - L_{11}vz)\end{aligned}$$

dove $R = L_9x + L_{10}y + L_{11}z + 1$. In forma matriciale:

$$\frac{1}{R}\begin{bmatrix}u\\ v\end{bmatrix} = \frac{1}{R}\begin{bmatrix}x & y & z & 1 & 0 & 0 & 0 & 0 & -ux & -uy & -uz\\ 0 & 0 & 0 & 0 & x & y & z & 1 & -vx & -vy & -vz\end{bmatrix}\begin{bmatrix}L_1\\ \vdots\\ L_{11}\end{bmatrix}$$

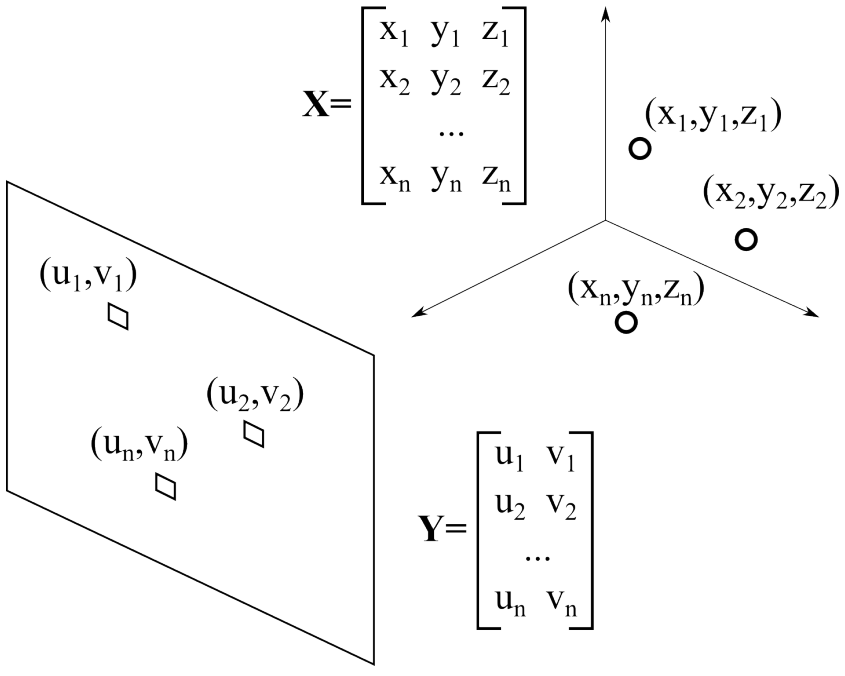
Possiamo vedere l'equazione nella forma:

$$\mathbf{Y} = \mathbf{X} \cdot \mathbf{L}$$

Lo scopo della calibrazione è ottenere i parametri che caratterizzano il sistema $\begin{bmatrix} L_1 \\ \vdots \\ L_{11} \end{bmatrix}$

Per cui è necessario conoscere **X** e **Y**.

Calibrazione



Posso prendere un numero *n* di punti nello spazio dell'oggetto e usare le coordinate misurate per definire **X**. Analogamente per gli stessi punti misuro le coordinate nello spazio dell'immagine, ottenendo **Y**.

Calibrazione

Nota bene che per risolvere il sistema $\mathbf{Y} = \mathbf{X} \cdot \mathbf{L}$ rispetto $\mathbf{L}$ devo avere alcune accortezze:

Attenzione!

- R è funzione di $L_9, \ldots, L_{11}$, per cui è necessario usare un approccio iterativo
- il sistema può essere risolto utilizzando il metodo dei minimi quadrati, ma per far questo è necessario conoscere un numero di punti maggiore del numero di parametri da derivare

Per completezza possiamo scrivere l'estensione a n punti di controllo dell'equazione di calibrazione:

$$\begin{bmatrix} \frac{x_1}{R_1} & \frac{y_1}{R_1} & \frac{z_1}{R_1} & \frac{1}{R_1} & 0 & 0 & 0 & 0 & \frac{-u_1 x_1}{R_1} & \frac{-u_1 y_1}{R_1} & \frac{-u_1 z_1}{R_1} \\ 0 & 0 & 0 & 0 & \frac{x_1}{R_1} & \frac{y_1}{R_1} & \frac{z_1}{R_1} & \frac{1}{R_1} & \frac{-v_1 x_1}{R_1} & \frac{-v_1 y_1}{R_1} & \frac{-v_1 z_1}{R_1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{x_n}{R_n} & \frac{y_n}{R_n} & \frac{z_n}{R_n} & \frac{n}{R_n} & 0 & 0 & 0 & 0 & \frac{-u_n x_n}{R_n} & \frac{-u_n y_n}{R_n} & \frac{-u_n z_n}{R_n} \\ 0 & 0 & 0 & 0 & \frac{x_n}{R_n} & \frac{y_n}{R_n} & \frac{z_n}{R_n} & \frac{n}{R_n} & \frac{-v_n x_n}{R_n} & \frac{-v_n y_n}{R_n} & \frac{-v_n z_n}{R_n} \end{bmatrix} \begin{bmatrix} L_1 \\ \vdots \\ L_{11} \end{bmatrix} = \begin{bmatrix} \frac{u_1}{R_1} \\ \frac{v_1}{R_1} \\ \vdots \\ \frac{u_n}{R_n} \\ \frac{v_n}{R_n} \end{bmatrix}$$

Per 11 parametri, siccome ogni punto di controllo fornisce 2 equazioni, è necessario utilizzare almeno 6 punti di controllo.
L viene quindi derivato come segue:

$$\begin{aligned} \mathbf{X} \cdot \mathbf{L} &= \mathbf{Y} \\ (\mathbf{X}^t \cdot \mathbf{X}) \cdot \mathbf{L} &= \mathbf{X}^t \cdot \mathbf{Y} \\ (\mathbf{X}^t \cdot \mathbf{X})^{-1} (\mathbf{X}^t \cdot \mathbf{X}) \cdot \mathbf{L} &= (\mathbf{X}^t \cdot \mathbf{X}) \cdot (\mathbf{X}^t \cdot \mathbf{Y}) \\ \mathbf{L} &= (\mathbf{X}^t \cdot \mathbf{X})^{-1} \cdot (\mathbf{X}^t \cdot \mathbf{Y}) \end{aligned}$$

Ricostruzione

Una volta ottenuti i parametri DLT, per la ricostruzione devo riarrangiare (6) in modo da ricavare le coordinate nello spazio dell'oggetto dei punti sul piano-immagine:

$$\begin{bmatrix} \frac{u^1 L_9^1 - L_1^1}{R^1} & \frac{u^1 L_{10}^1 - L_2^1}{R^1} & \frac{u^1 L_{11}^1 - L_3^1}{R_1} \\ \frac{v^1 L_9^1 - L_1^1}{R^1} & \frac{v^1 L_{10}^1 - L_2^1}{R^1} & \frac{v^1 L_{11}^1 - L_3^1}{R_1} \\ \frac{u^m L_9^m - L_1^m}{R^m} & \frac{u^m L_{10}^m - L_2^m}{R^m} & \frac{u^m L_{11}^m - L_3^m}{R_m} \\ \frac{v^m L_9^m - L_1^m}{R^m} & \frac{v^m L_{10}^m - L_2^m}{R^m} & \frac{v^m L_{11}^m - L_3^m}{R_m} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{L_4^1 - u^1}{R^1} \\ \frac{L_8^1 - v^1}{R^1} \\ \frac{L_4^m - v^m}{R^m} \\ \frac{L_8^m - u^m}{R^m} \end{bmatrix}$$

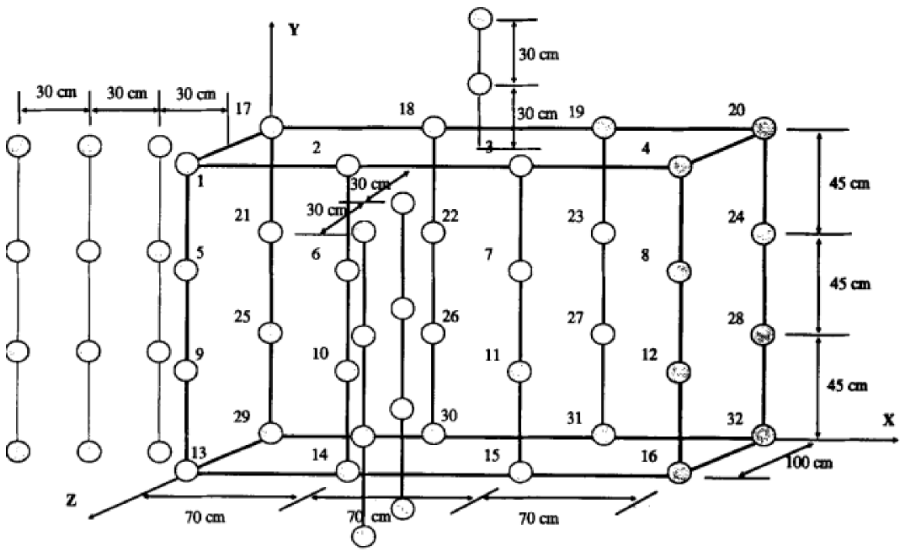
Siccome i punti hanno 3 coordinate, il numero minimo di camere necessarie alla ricostruzione sono perciò $m \geq 2$.

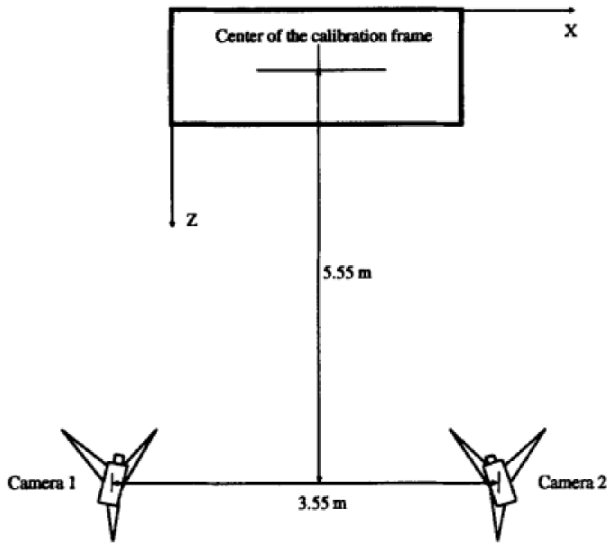
```
function param=calibration(Lin,ucam,p)

-----definizioni------------------
ics=zeros(1,11);
ipsilon=zeros(1,1);
l9=Lin(1,1); l10=Lin(1,2); l11=Lin(1,3);
M=zeros(11,1);
niterazioni=50;
\%[...]
for iterazioni=1:niterazioni
    for i=1:8
        R=(l9*p(i,1)+l10*p(i,2)+l11*p(i,3)+1);
        tmp=[p(i,1)/R p(i,2)/R p(i,3)/R 1/R 0 0 0 0 -...
ucam(i,1)*p(i,1)/R -ucam(i,1)*p(i,2)/R -ucam(i,1)*p(i,3)/R;
             0 0 0 0 p(i,1)/R p(i,2)/R p(i,3)/R 1/R -...
ucam(i,2)*p(i,1)/R -ucam(i,2)*p(i,2)/R -ucam(i,2)*p(i,3)/R];
        ics=cat(1,ics,tmp);
        tmp2=[ucam(i,1)/R; ucam(i,2)/R];
        ipsilon=cat(1,ipsilon,tmp2);
    end
    X=ics(2:end,:);
    Y=ipsilon(2:end,:);

    L=inv(X'*X)*(X'*Y);
    %% --- M per track
    M=cat(2,M,L);
    %% ----------
    l9=L(9,1); l10=L(10,1); l11=L(11,1);
    % ---- clear prima delle prossime iterazioni
    clear X L Y ics ipsilon; ics=zeros(1,11); ipsilon=zeros(1,1);
    % ----------------------------------------
end
```

Avvertenze finali

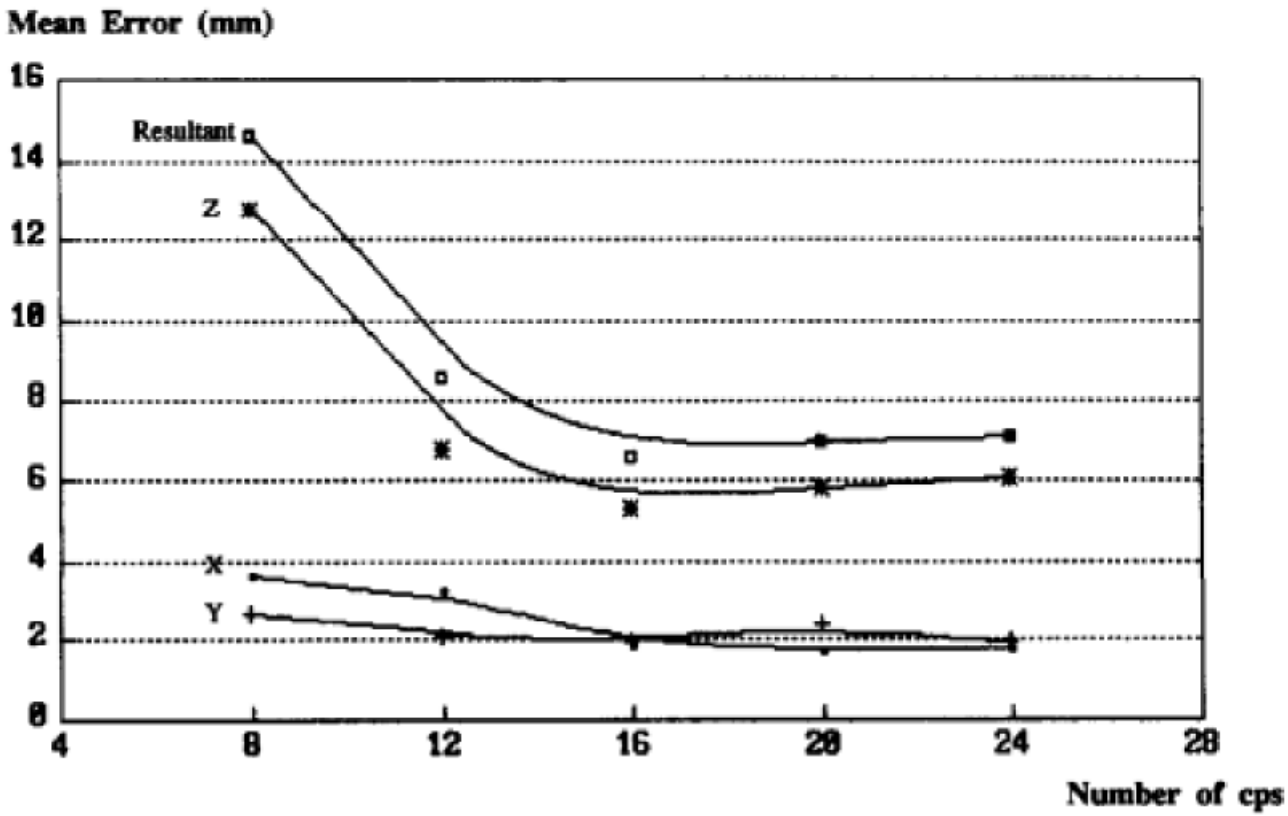
- Ogni camera deve essere calibrata! Per cui la procedura di calibrazione deve essere effettuata almeno 2 volte.
- I punti di controllo dello spazio di calibrazione non devono essere complanari, cioè devono definire un volume
- Lo spazio di calibrazione deve essere sufficientemente grande da includere lo spazio di movimento.
- Includere più punti di controllo possibile aumenta la ridondanza del sistema e l'accuratezza della calibrazione.
- Stesso discorso per la ridondanza delle camere: anche se va evitato di posizionare le camere una affacciata all'altra, o almeno non usare quella combinazione di camere per la ricostruzione.

Qualche numero sull'accuratezza

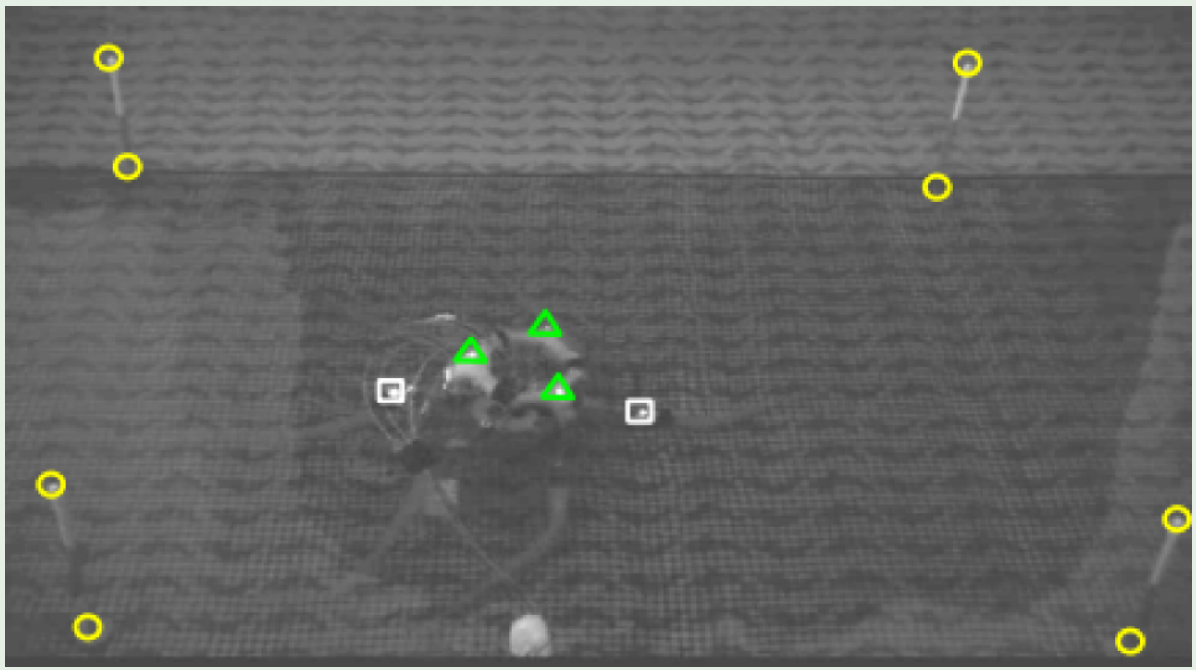




Errore



Example

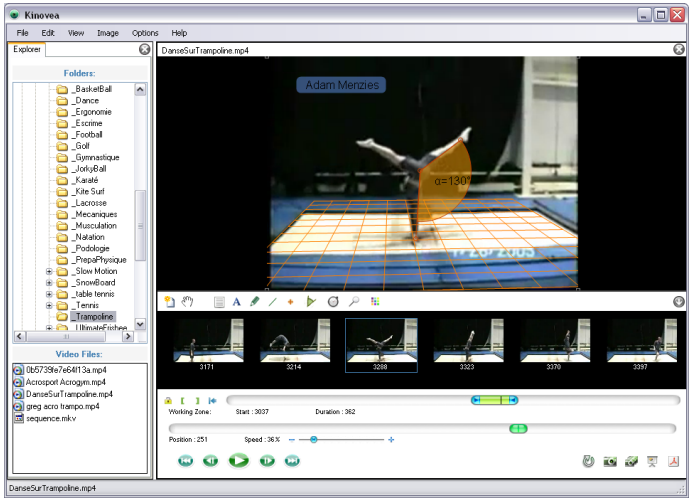


Esempio di ricostruzione

Video: 

Ancora curiosi?

. . . www.kinovea.com



Cosa succede se non posso utilizzare dei marker?
Devo cercare dei punti di interesse che abbiano una posizione ben definita e che possano essere determinati in maniera robusta.
Per il tracking, li voglio anche seguire immagine dopo immagine, come per i marker.
Una stima di dove i marker, o altri punti di interesse, si stanno muovendo aiuta la ricostruzione.

Flusso ottico

Il movimento è una parte integrante del nostro processo visivo, e spesso viene utilizzato per tutta una varietà di compiti:

- riconoscere forme tridimensionali
- controllo oculomotorio
- organizzazione percettiva
- riconoscimento di oggetti
- predizioni
- comprensione della scena

Flusso ottico

Una superficie che si muove nello spazio, proiettata nel piano delle immagini, produce un percorso bidimensionale di velocità, $dx(t)/dt$ e $dy(t)/dt$ al quale spesso ci si riferisce come *campo motorio bidimensionale*.
Stimare il *flusso ottico* ha lo scopo di approssimare il campo motorio dalla variazione nel tempo delle intensità dell'immagine.

Un punto di partenza per stimare il flusso ottico è assumere che l'intensità di un pixel in un istante t si è spostato in un punto vicino nell'istante $t + dt$.

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{7}$$

Si suppone che:

1. Superfici a luminanza è isotropica
2. La fonte luminosa è molto distante
3. Non ci sono rotazioni degli oggetti
4. Non ci sono altre fonte di illuminazione

Se il movimento è piccolo, l'espansione in serie di Taylor ci dice che:

$$I(x+dx, y+dy, t+dt) = I(x,y,t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt$$

che sostituita in (7) ci da:

$$\frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt = 0$$

Questa equazione viene chiamata *gradient constraint equation*.

$$\frac{\partial I}{\partial x}\dot{x} + \frac{\partial I}{\partial y}\dot{y} + \frac{\partial I}{\partial t} = 0$$

e può anche essere scritta:

$$\nabla I \cdot V^T + \frac{\partial I}{\partial t} = 0$$

con V=$(\dot{x}, \dot{y})$

Problema dell'apertura

Vediamo che l'equazione ha due incognite $(\dot{x}, \dot{y})$, per cui non può essere risolta direttamente.

Lucas-Kanade

Per risolvere il problema dell'apertura, Lucas e Kanade ipotizzarono le seguenti condizioni:

- Lo spostamento dei pixel tra due frame adiacenti sia piccolo
- Lo spostamento dei pixel sia costante in un intorno del punto p

Questo equivale a dire che il flusso ottico è costante per tutti i pixel centrati in p, ovvero:

$$\begin{aligned} I_x(q_1)\dot{x} + I_y(q_1)\dot{y} &= -I_t(q_1) \\ I_x(q_1)\dot{x} + I_y(q_1)\dot{y} &= -I_t(q_1) \\ &\vdots \\ I_x(q_1)\dot{x} + I_y(q_1)\dot{y} &= -I_t(q_1) \end{aligned}$$

dove $q_1, \ldots, q_n$ sono i pixel dentro la finestra centrata in p, e $I_{x,y,t}$ le derivate dell'immagine rispetto x,y e t.

Scrivendo le equazioni in forma matriciale: $A \cdot v = b$ con:

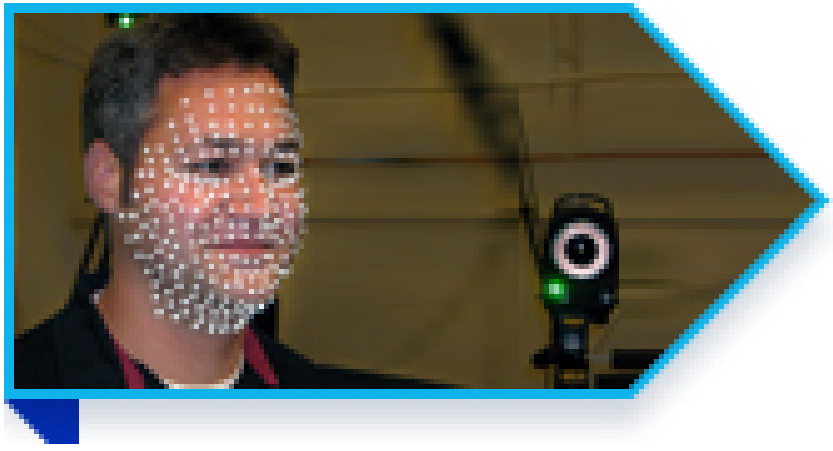
$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix}, v = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}, b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix}$$

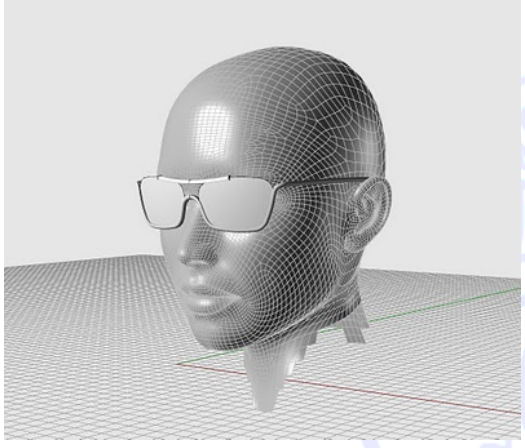
Il sistema ha più equazioni che incognite, e quindi può essere nuovamente risolto con il metodo dei minimi quadrati:

$$v = (A^T A)^{-1} A^T b$$

Specchio 3D

Specchio, servo delle mie brame...





3leD Mouse

Un led su pollice, indice e medio.

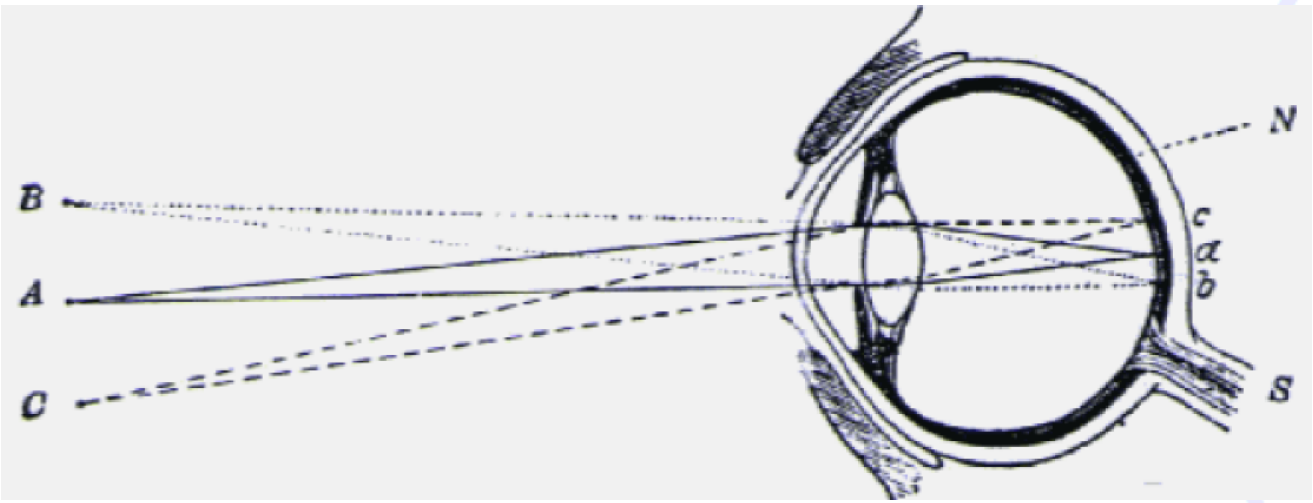
Procedura che abbiamo visto

- Contrasto-soglia
- Erosione-flusso ottico
- Ricostruzione 3D

Possiamo aggiungere anche la profondità allo schermo!

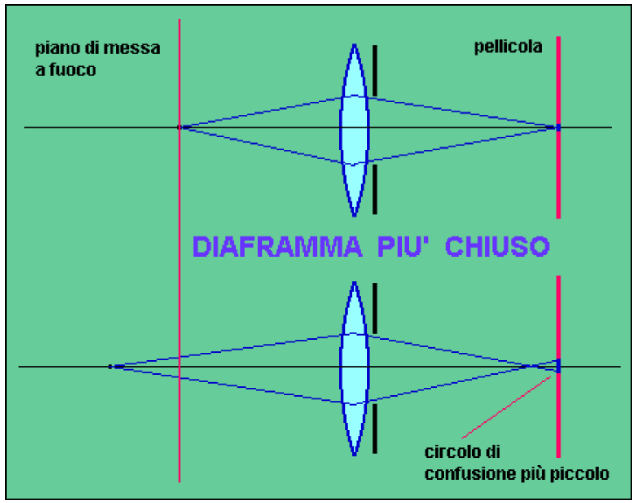
Come si crea l'immagine

L'immagine viene proiettata dalla lente verso il piano focale.



Cosa è la sfocatura

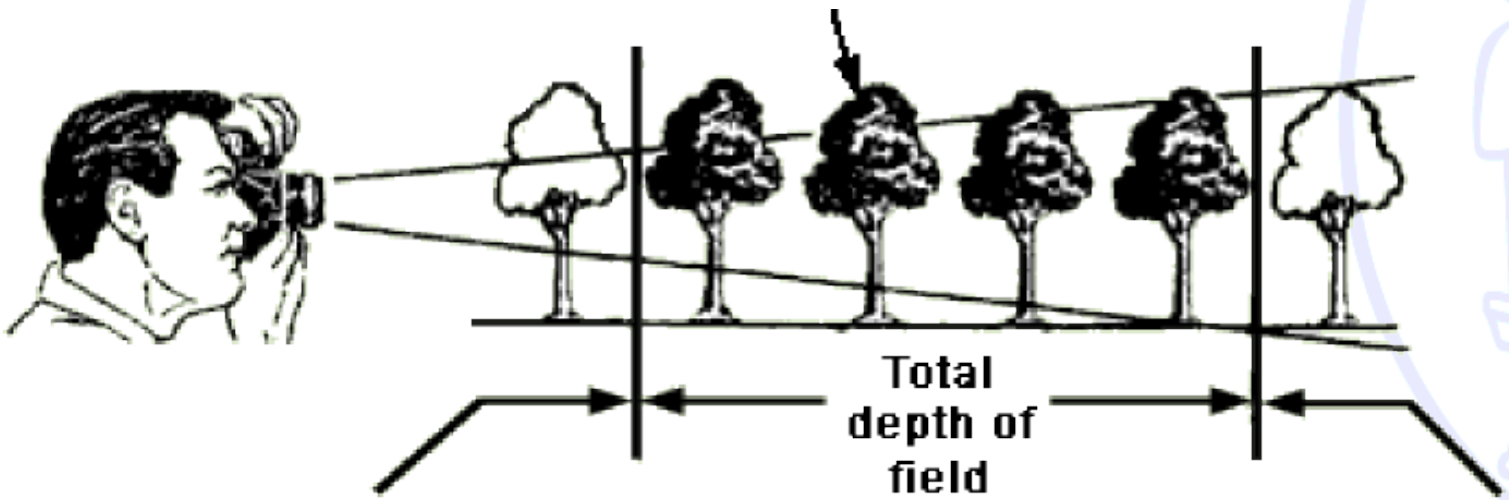
Quando la convergenza della proiezione non è esattamente sul piano del sensore, si crea una sfumatura dei bordi dell'immagine che è chiamata sfocatura.



In pratica, un punto non è recepito più come un punto, ma come un disco (circolo di confusione) la cui dimensione determina l'entità della sfocatura.

Profondità di campo

Anche gli elementi al di fuori del piano di fissazione risultano sfocati: è chiamata *profondità di campo* la porzione si spazio, ortogonale al piano di proiezione dell'immagine, in cui una figura risulta correttamente messa a fuoco.



Profondità di campo

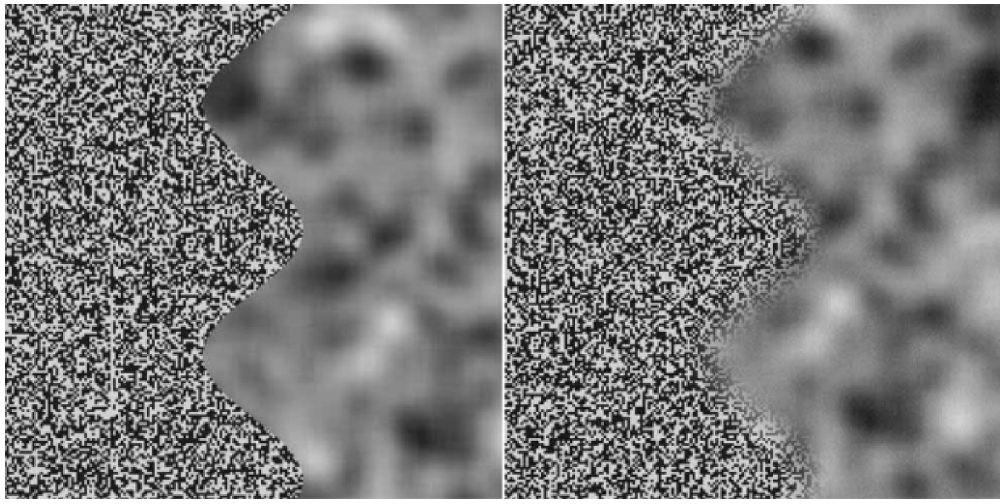
$$
\begin{aligned}
H &= \frac{f^2}{N \cdot c} + f \\
D_n &= \frac{s(H-f)}{H+s-2f} \\
D_f &= \frac{s(H-f)}{H-s}
\end{aligned}
$$

La PdC aumenta:

1. all'aumentare del f-number, N
2. della distanza dal soggetto, s
3. al decrescere della distanza focale dell'obbiettivo, f

Percezione di profondità

Immagini sfocate artificialmente hanno dimostrato che la sfocatura viene utilizzata come mezzo per percepire la profondità.



Le immagini a fuoco sono percepite come più vicine rispetto alle altre, ma non è possibile dare una stima corretta.

Accomodazione e blur retinale

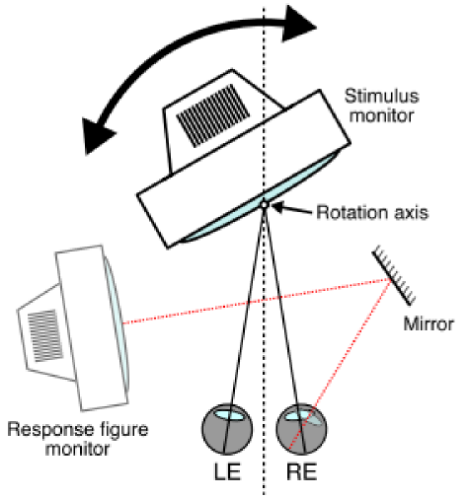
La percezione della profondità è principalmente *monoculare* sopra i 30 metri (profondità di campo) mentre è principalmente *binoculare* sotto questa soglia (disparità ottica).
Vari indizi vengono utilizzati per percepire la profondità:

- dimensioni
- parallasse
- ...
- accomodazione
- blur retinale

L'incoerenza tra i vari indizi altera la percezione delle pendenze!

Accomodazione e blur retinale

Lo schermo a una certa distanza fornisce degli indizi dovuti alla distanza fisica dell'osservatore.



Questo è quello che succede spesso nei videogiochi: gli indizi sono contrastanti, perchè le immagini dicono una cosa, l'accomodazione e il blur un'altra.

Point Spread Function (PSF)

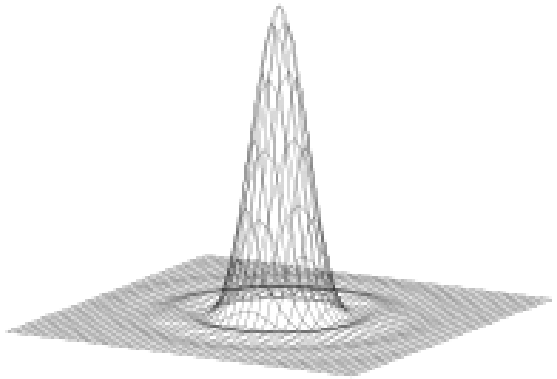
Per le immagini nel dominio delle frequenze spaziali, possiamo assegnare una funzione che rappresenti, dall'oggetto originale di dimensioni trascurabili (un punto), la sua rappresentazione nell'immagine:

Funzione di risposta impulsiva (PSF)

L'effetto del sistema di acquisizione farà apparire il punto come una versione sfocata dell'originale, producendo perciò una macchia o un cerchio

Questa può essere vista anche nel dominio delle frequenze (e viene chiamata *Optical Transfer Funcion*) e la sua ampiezza è chiamata *Modulation Transfer Funcion*.

La PSF ha questa forma:

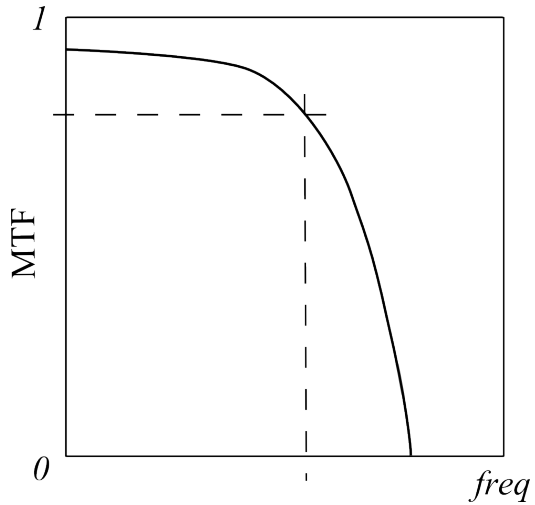


Per cui capiamo come un punto venga spalamato in una macchia.
Evidenziamo due effetti collegati all'aumentare dell'ampiezza della PSF:

Effetti

- vengono attenuate le alte frequenze delle immagini
- scompaiono spikes dalla immagini

La MTF è una funzione di questo tipo:



Spesso la sfocatura viene descritta tramite queste due funzioni, ma non solo: la sfida di molti ricercatori è stata trovare la PSF che ha generato una certa immagine sfocata per poter tornare all'immagine originale.

Come possiamo valutare se un'immagine è messa a fuoco o meno?

La messa a fuoco può essere valutata in maniera:

diretta: alcuni sensori misurano la distanza dal soggetto, e il sistema di messa a fuoco si muove di conseguenza

indiretta: varie immagini della stessa scena vengono acquisite e una metrica determina quale è l'immagine a fuoco

Valutazione veloce

Marziliano *et al*, (2008), introducono una valutazione veloce della sfocatura sfruttando un concetto molto semplice: che il bordo sfocato sia distribuito su una maggiore superficie di spazio rispetto a un bordo a fuoco.

La loro metrica è indipendente dal tipo di sfocatura:

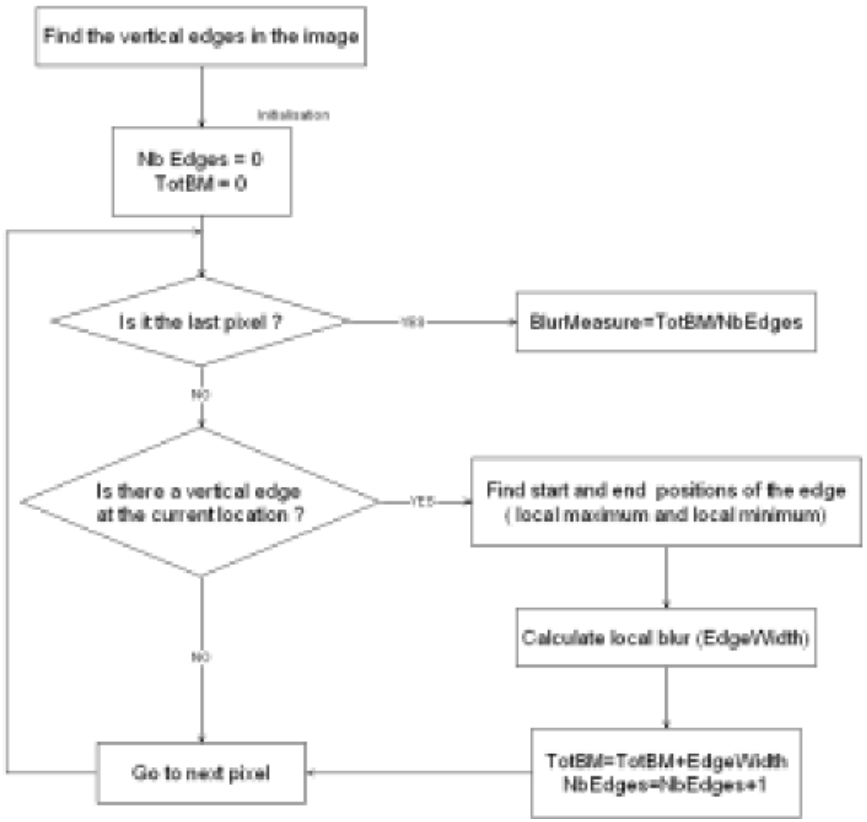
- di movimento
- di messa a fuoco
- di aberrazione
- ...

Possono essere utilizzati sia gli edge verticali che orizzontali. Nella pratica, sono stati usati solo gli edge verticali, ricavati con la maschera di Sobel appropriata.

Algoritmo

1. Trova i bordi
2. Valuta lo spessore dell'edge locale
3. Media tutti gli spessori sull'immagine

Diagramma di flusso

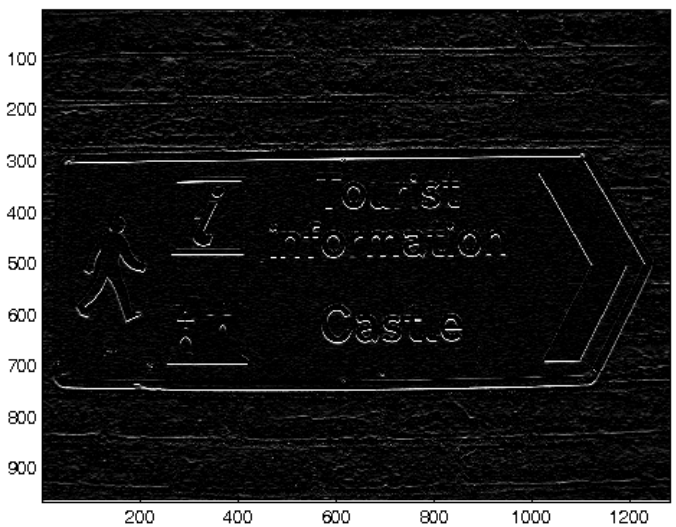


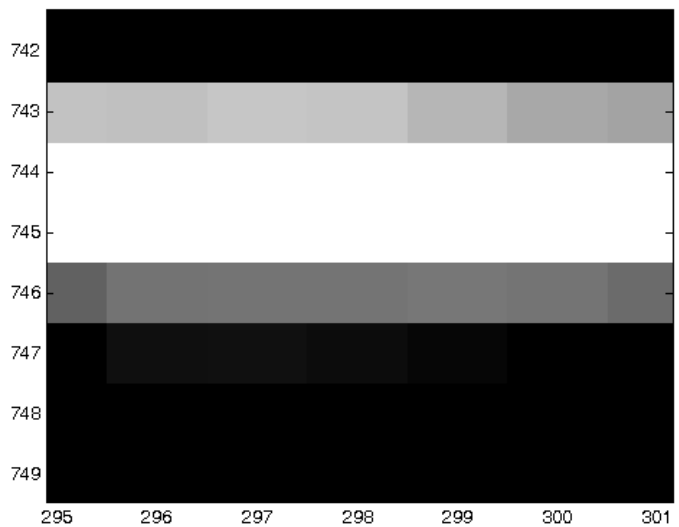
Example

Immagine perfettamente a fuoco

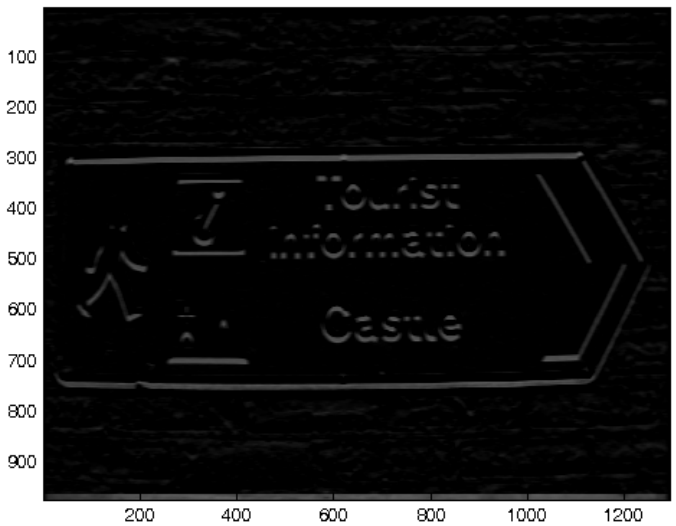
Immagine con una leggera sfocatura

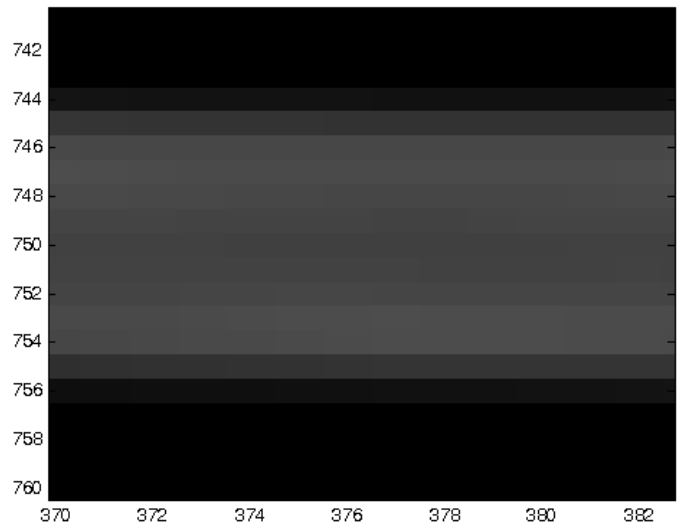
Example


100
200
300
400
500
600
700
800
900
200
400
600
800
1000
1200
Tourist
Information
Castle


742
743
744
745
746
747
748
749
295
296
297
298
299
300
301

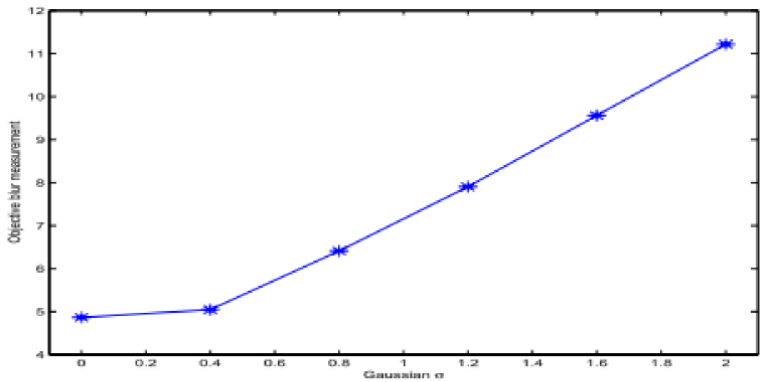
Example



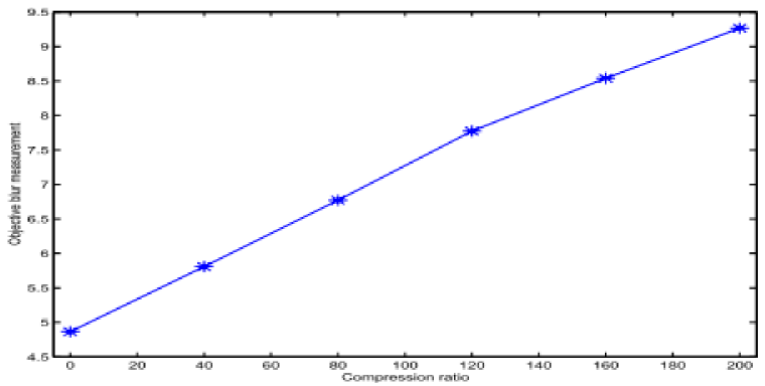


Notate anche come gli spikes dell'immagine sono scomparsi

Andamenti


Objective blur measurement
Gaussian σ
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
4
5
6
7
8
9
10
11
12

(a)


Objective blur measurement
Compression ratio
0
20
40
60
80
100
120
140
160
180
200
4.5
5
5.5
6
6.5
7
7.5
8
8.5
9
9.5

Metodo degli autovalori

Molte metriche si basano sul fatto che la PSF sopprime (o attenua) le componenti ad alta frequenza.
Quindi la differenza tra le varie immagini nelle componenti ad alta frequenza viene spesso utilizzata come stima per la messa a fuoco.
Wee e Paramesran (2008) propongono una metrica che si basi sulla determinazione degli autovalori dell'immagine:

- i primi autovalori contengono le informazioni sulle immagini
- gli ultimi contengono le informazioni sul rumore

In questo modo, scartando solo gli ultimi valori si ottengono le informazioni sulla nitidezza senza essere influenzati dal rumore

Procedura

L'immagine $g(x, y)$ viene normalizzata secondo la sua energia:

$$\tilde{g}(x, y) = \frac{g(x, y)}{\sqrt{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} [g(x, y)]^2}}$$

Viene eseguita la media di $\tilde{g}(x, y)$:

$$\mu = \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \tilde{g}(x, y)$$

Procedura

Deviazione della matrice di covarianza:

$$\mathbf{S}_{\tilde{g}} = \frac{1}{N-1}\mathbf{G}\mathbf{G}^T$$

con $\mathbf{G} = (g - \tilde{g})$
Viene diagonalizzata la matrice tramite SVD per ottenere la matrice a valori singolari, **A**.

Procedura

Gli ultimi passaggi prevedono il calcolo della matrice diagonale di autovalori:

$$\Lambda = A^T \cdot A$$

e infine, la metrica è valutata selezionando i primi k autovalori:

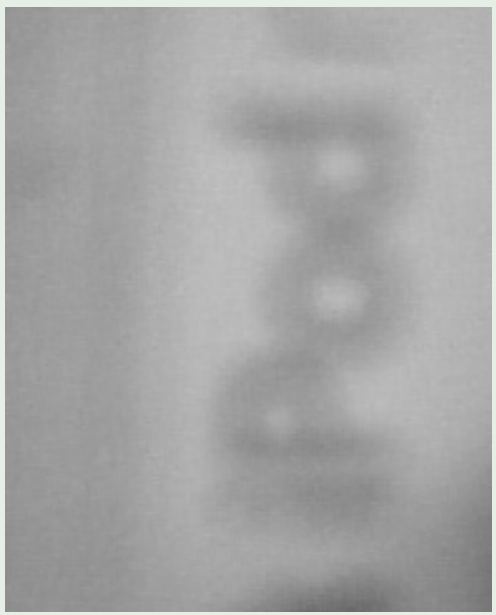
$$M_e = trace(\Lambda)$$

Conclusioni

Secondo gli autori, questa metrica fornisce una valutazione della sfocatura più accurata per un range maggiore di distorsioni.

Example

Example



Visto che sia la sfocatura retinale che l'accomodazione sono indizi importanti nel biologico...
perchè non provare ad implementarli anche nel robotico?

Grasping in vicinanza

- Valutare la capacità (monoculare) di distinguere oggetti in stretta vicinanza e di fornire informazioni sulla distanza degli oggetti (confrontando o sviluppando vari algoritmi di stima della nitidezza/sfocatura)
- Unire le informazioni stereoscopiche a quelle di fuoco per migliorare la qualità del grasping

La rappresentazione dell'ambiente e degli oggetti può anche essere ricavata con altri sistemi sensoriali rispetto a camere che creino immagini bidimensionali.
Una metodologia comune è la scansione laser: generalmente, il laser restituisce la distanza, rispetto alla fonte di emissione, di un eventuale ostacolo che riflette il fascio.
In questo modo possono essere create rappresentazioni di oggetti, o di ambienti, che sono descritte da nubi di punti.

Qui vedremo 2 metodologie per estrarre bordi da immagini in nubi di pixel, allo scopo di effettuare la registrazione tra più viste.

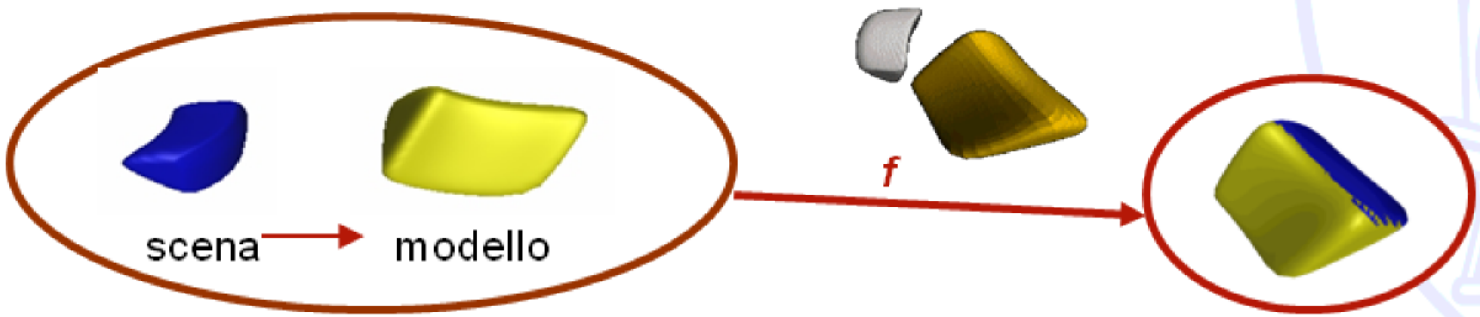
Registrazione di immagini

La registrazione di immagini ha lo scopo di:

- allineare due o più viste di uno stesso oggetto

Ottimizzazione

Viene ottimizzato riducendo gli elementi descrittivi necessari ad essere allineati (possono essere composti da milioni di punti!)



Data un'immagine $I_s \in R^3$, composta da $I_s = \{p_1, p_2, \ldots, p_s\}$ chiamata scena, e un'immagine $I_m \in R^3$, composta da $I_m = \{p_1, p_2, \ldots, p_m\}$ chiamata modello, e tali che la loro intersezione non sia nulla, lo scopo della registrazione è costruire una immagine che sia composta dai due insiemi. E' come girare una tessera di un puzzle affinchè due parti possano coincidere correttamente.

Che nella scena e nel modello siano presenti esattamente gli stessi punti è molto improbabile se non impossibile.
Varie metodologie sono perciò state sviluppate affinché le due parti possano essere allineate correttamente.
Punto fondamentale di ogni metodologia è quello ridurre il numero di punti da dover analizzare:

$$\begin{aligned} I'_s &= \{p_1, p_2, \ldots, p'_s\} \\ I'_m &= \{p_1, p_2, \ldots, p'_m\} \end{aligned}$$

con $s' < s$ e $m' < m$ tali che $I'_s \cap I'_m \neq \emptyset$.

Gli elementi selezionati dovranno essere:

- Caratteristici degli oggetti da ricostruire
- Presenti in entrambe le viste da accoppiare
- Rappresentati da un numero di punti più ridotto possibile

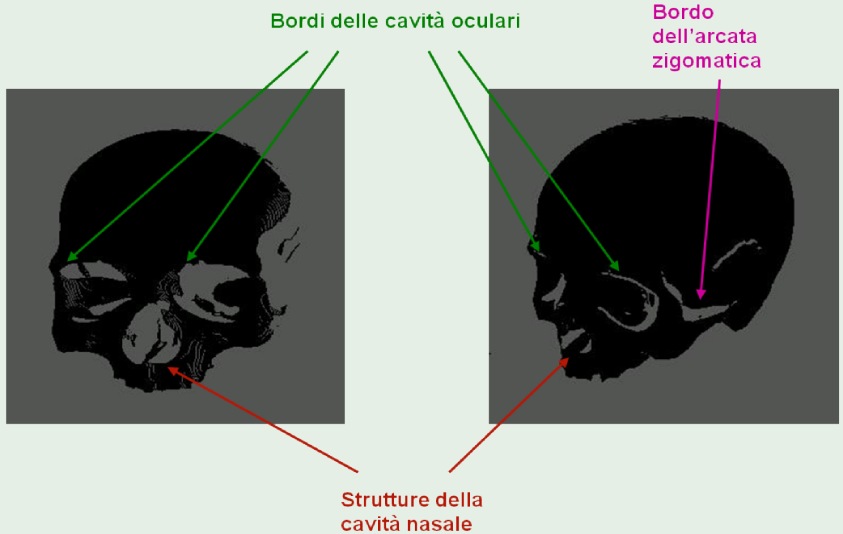
Il processo di estrazione deve essere:

- Completamente automatico
- Indipendenti da piccole differenze negli oggetti
- Indipendente dalla vista e dall'acquisizione

Nel forense

Example

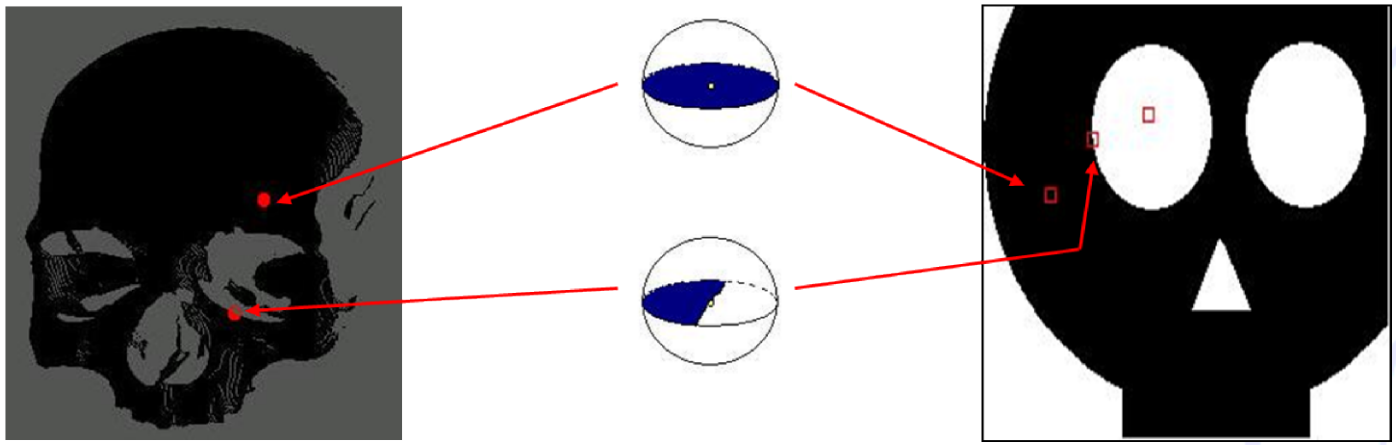
Strutture di interesse anatomico



- Bolle di densità
- Estrazione per smussamento

Bolla di densità

Analogia con il 2D...



e si dice che un punto $q = (q_x, q_y, q_z)$ appartiene alla bolla **B** di raggio r centrata in p, $q \in \mathbf{B}_p$, se:

$$\sqrt{(p_x - q_x)^2 + (p_x - q_x)^2 + (p_x - q_x)^2} \leq r$$

Il numero di punti che appartengono ad ogni bolla definiscono la *densità della bolla*.

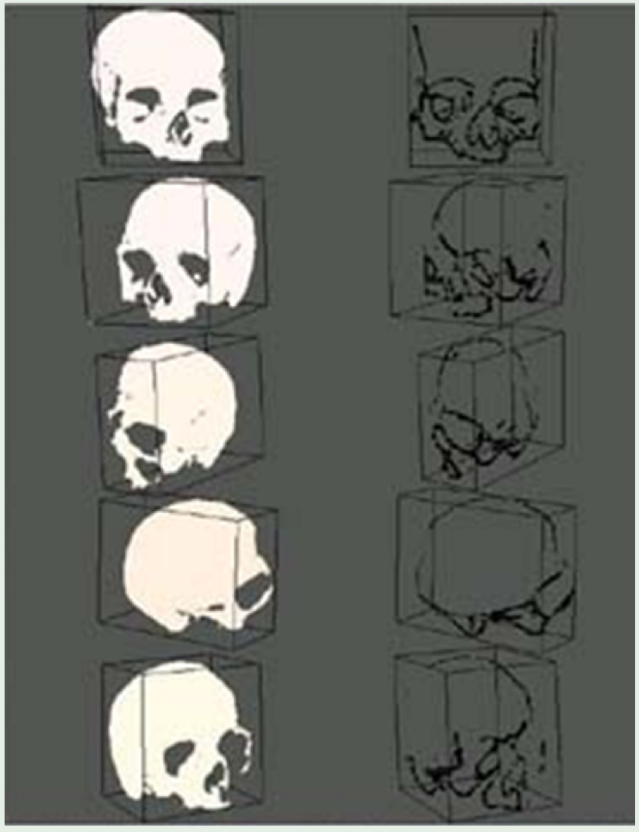
$$d_i = \sum_j q_j, \forall q_j \in \mathbf{B}_{p_i}$$

Il criterio di selezione o meno di un punto come appartente alle caratteristiche di interesse viene ottenuto scegliendo una densità di taglio:

$$\alpha \cdot d^{\gamma} \leq d_t$$

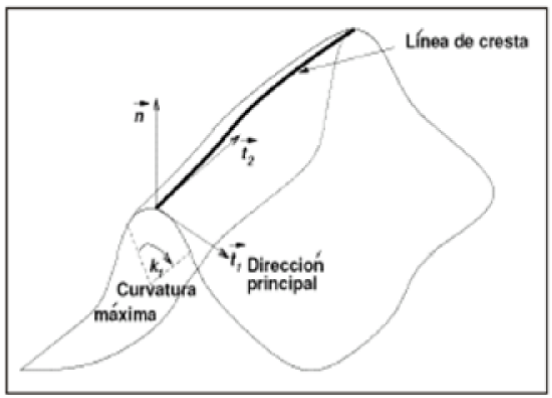
con α e γ parametri euristici.

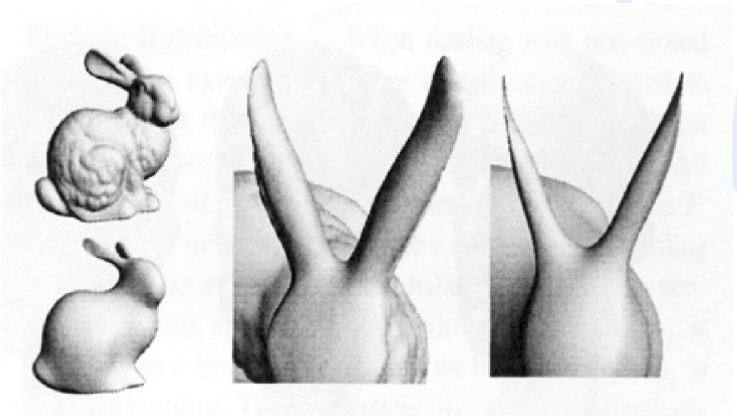
Example



Altre caratteristiche di interesse, oltre i bordi, possono essere:

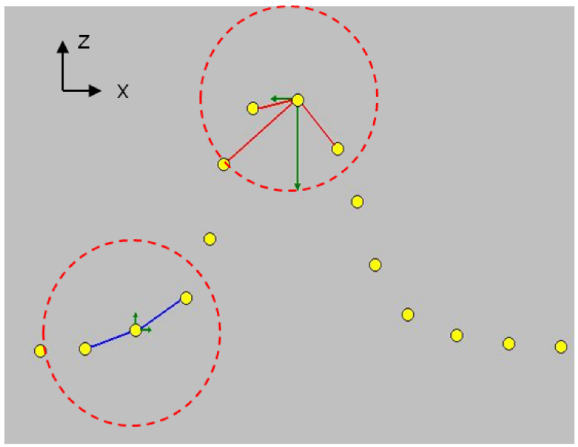
- Bordi (già estratti con le bolle)
- Creste e valli
- Spigoli

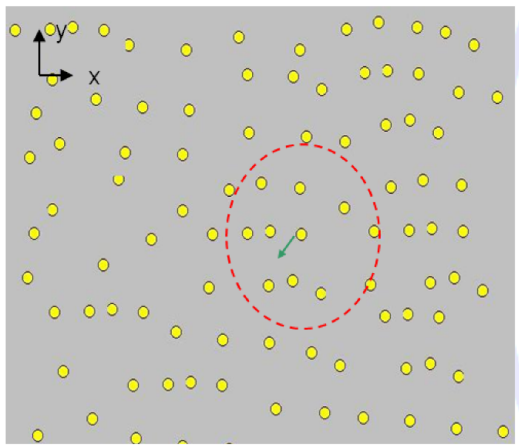




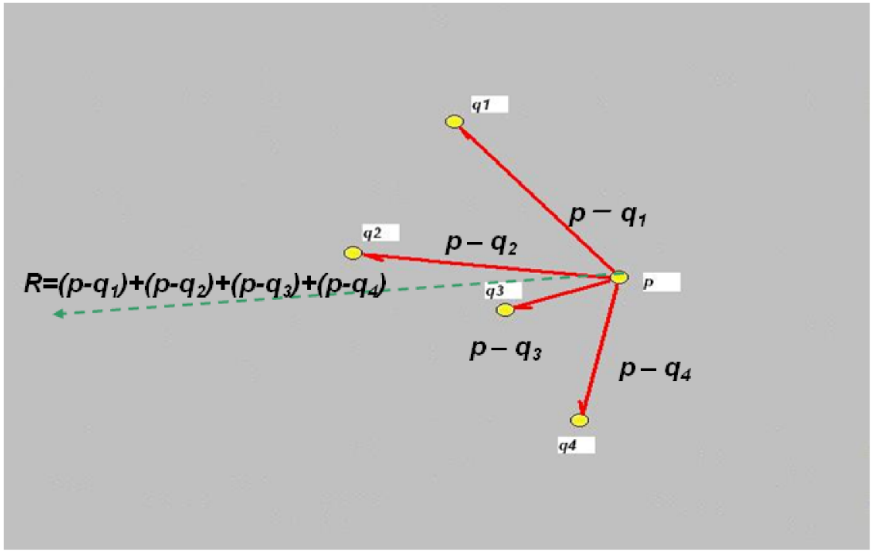
Eppur si muove...?

Nuovo criterio basato sull'attrazione e spostamento tra punti.
Punti $q_i \in \mathbf{B}$ attraggono p.





La risultante dello spostamento sarà data da:

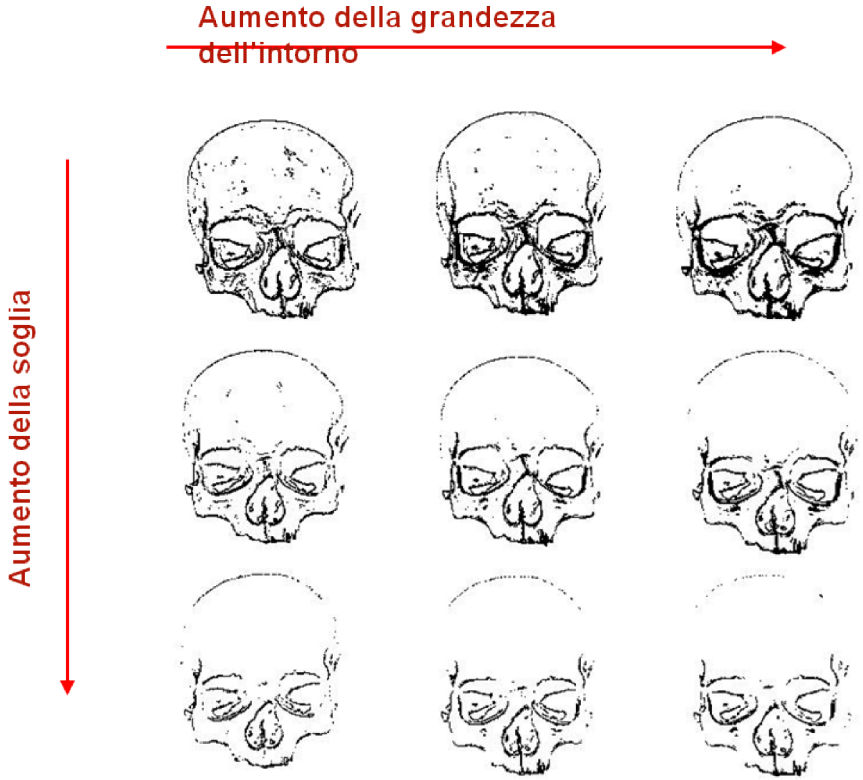


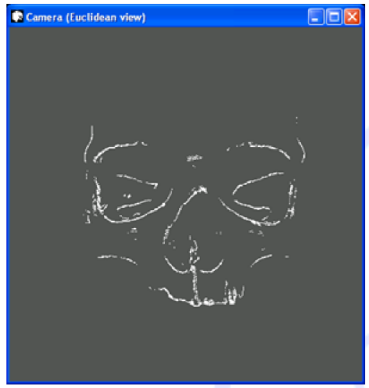
e la posizione finale del punto:

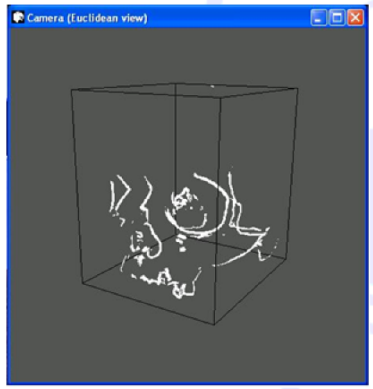
$$(p'_x, p'_y, p'_z) = \sum_{i \in \mathbf{B}} (\beta(p_x - q_{x_i}), \beta(p_y - q_{y_i}), \beta(p_z - q_{z_i}))$$

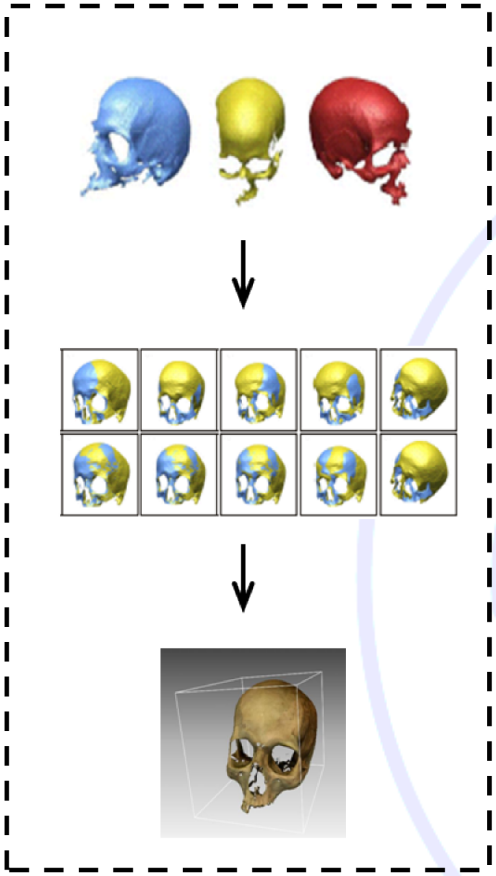
Il criterio di separazione sarà su quanto i punti si sono spostati:

$$s_i = \sqrt{(p'_x - p_x)^2 + (p'_y - p_y)^2 + (p'_z - p_z)^2}$$


Aumento della grandezza dell'intorno
Aumento della soglia


Camera (Euclidean view)


Camera (Euclidean view)



Dobbiamo classificare

Fino a questo punto, le armi per il riconoscimento (classificazione) sono state:

- Abbiamo riconosciuto cerchi (e simili)
- Possiamo estenderci ad altre forme geometriche (semplici)
- Abbiamo anche visto l'estrazione di alcune caratteristiche

La classificazione ruota tutta intorno alla estrazione di queste caratteristiche e alla loro separazione nello spazio di esistenza

Insieme di descrittori

Descrittori, caratteristiche, features sono tutti sinonimi. Tre comuni arrangiamenti di descrittori (classi) sono:

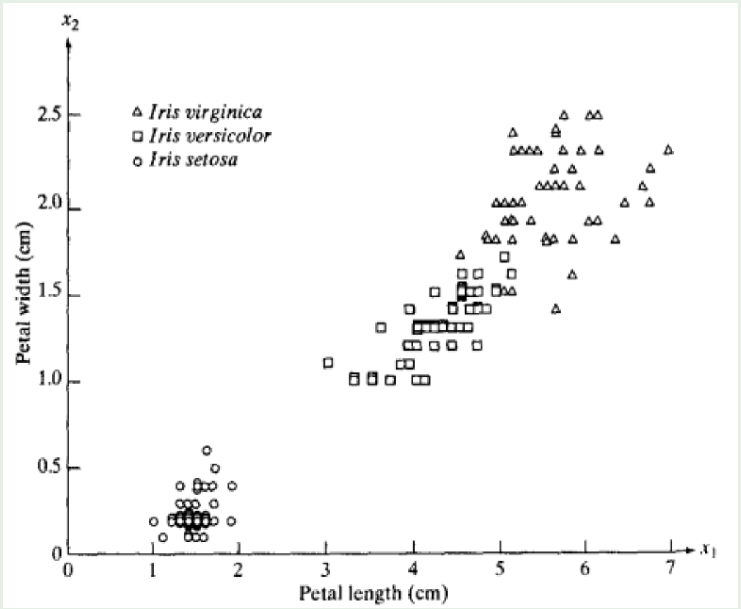
- vettori
- stringhe
- alberi

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

dove l'elemento x_i rappresenta l'i-esimo descrittore su gli n descrittori che rappresentano la classe.

Generalmente, non esistono a priori delle caratteristiche preferenziali per riconoscere un oggetto qualsiasi: la feature DNA non è estraibile dalle immagini!

Example



Classificatore sulla distanza minima

E' uno dei più semplici classificatori, e presuppone la conoscenza della classe.

Media della classe

Supponiamo di poter definire come l'archetipo della classe il valor medio dei suoi elementi:

$$m_j = \frac{1}{N_j} \sum_{N \in \omega_j} x_j, j = 1, 2, \ldots, W$$

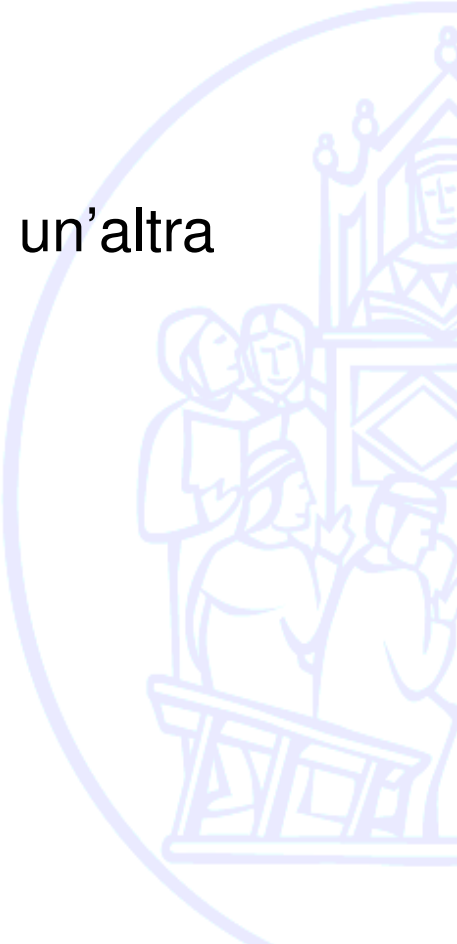
Con N_j numero dei vettori di caratteristiche per la determinata classe ω_j e W il numero totale delle classi

Minimum distance classifier

Un nuovo elemento x apparterrà ad una classe o ad un'altra confrontando la sua distanza con l'archetipo.

$$D_j = \|x - m_j\|$$

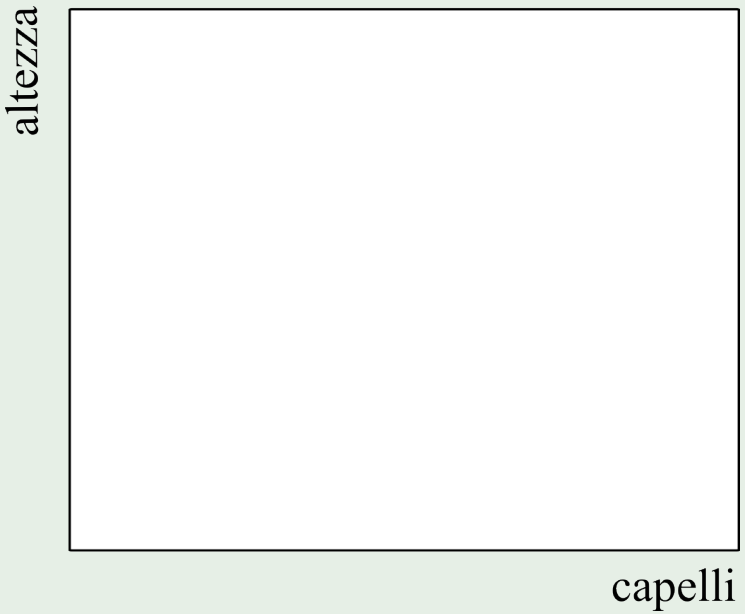
Il problema si riduce al calcolo della distanza.



Minimum distance classifier

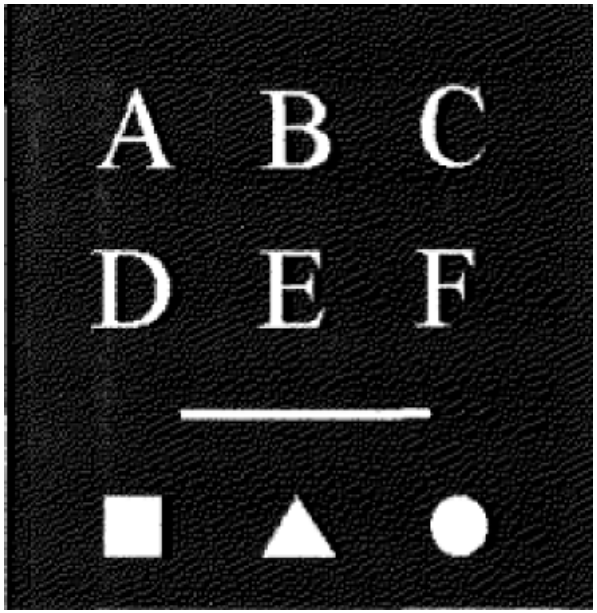
Lunghezza dei capelli/altezza della classe:

Example



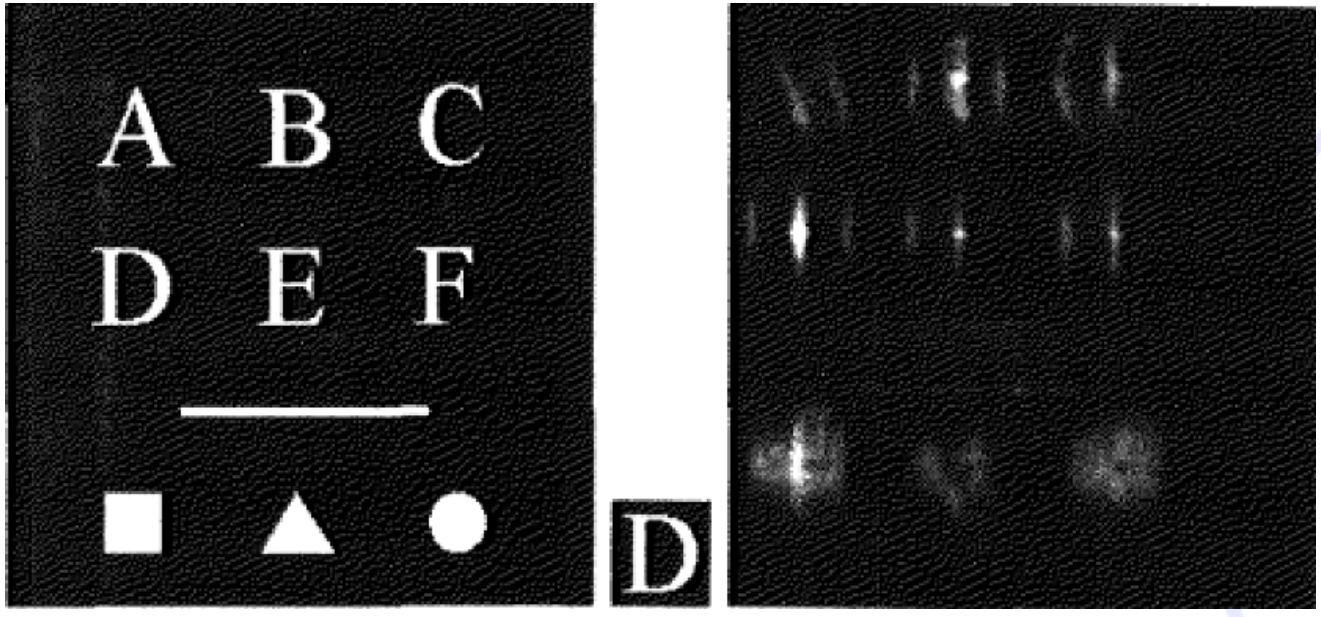
Correlation matching

Ci possiamo già arrivare...
poniamo di voler riconoscere la lettera *D* in questa immagine:



Correlation matching

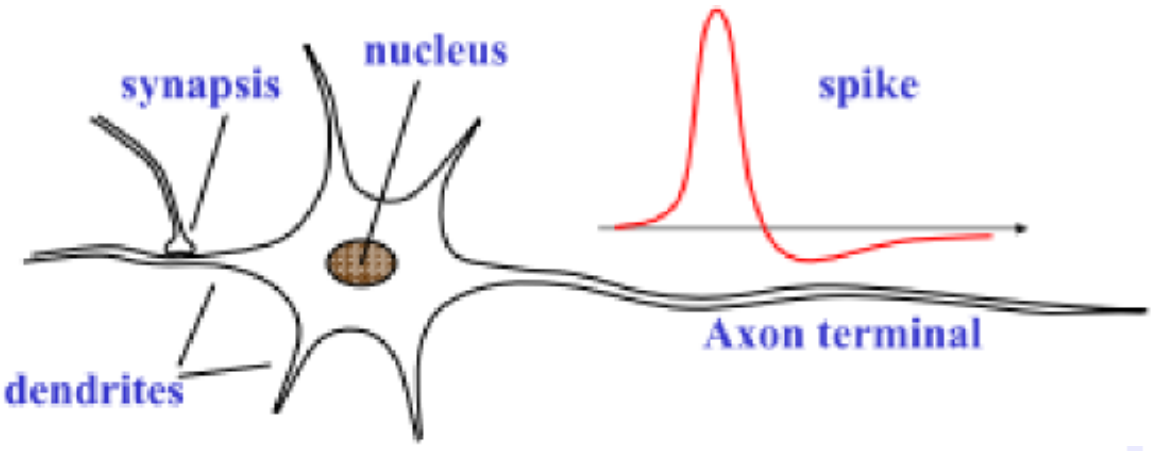
Usiamo una finestra con la nostra feature di ricerca!



Features resistenti

Esistono caratteristiche invarianti rispetto scala, rotazioni, sfocatura, ...

Neurone biologico



Interessanti proprietà

- Tollerante al rumore
- Consumo ridotto

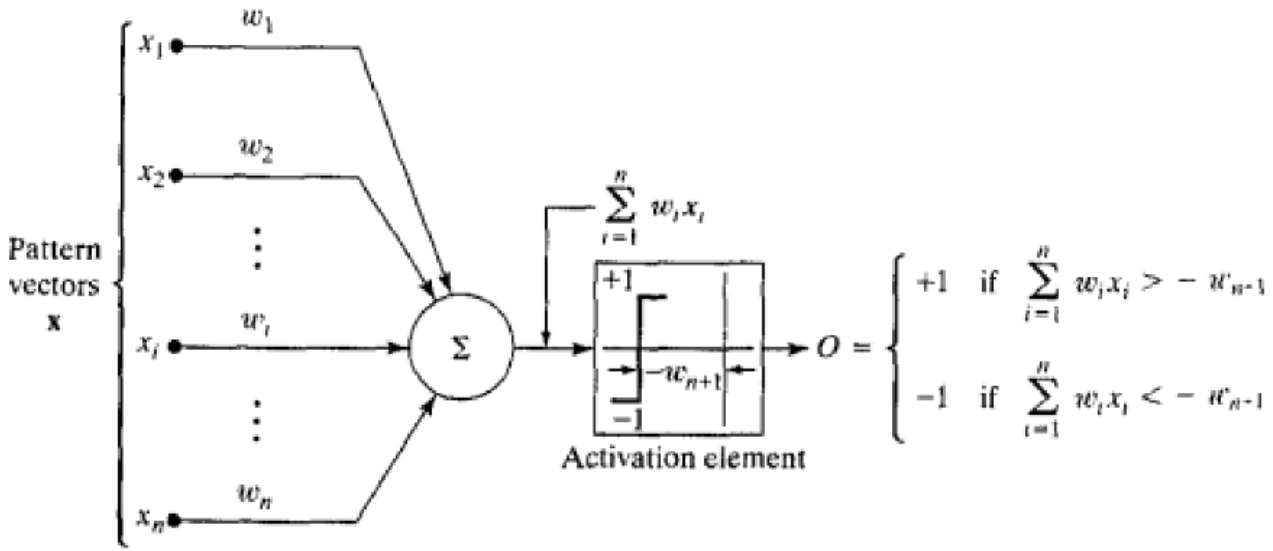
Un po' di storia...

- 1943, McCulloch e Pitts: propongono il primo modello di neurone a soglia di attivazione binaria
- 1946, Hebb scopre che non è il neurone a cambiare, ma l'apprendimento è legato alle sinapsi
- 1962, Rosenblatt propone un modello di neurone capace di imparare da esempi: è nato il *percettrone*
- 1982, Hopfield propone una *rete* di neuroni per creare memoria associativa
- 1982, Kohoen propone una reta chiamata mappa auto-organizzante
- 1985, Rumelhart, Hinton e Williams: formalizzano il processo di apprendimento back-propagation

Per definire un modello di neurone, abbiamo bisogno di alcuni elementi:

- Numero di ingressi N
- tipo di segnali di ingresso x_i
- i pesi delle connessioni w_i
- la funzione di attivazione f
- la funzione di integrazione F

Da cui il modello può essere rappresentato così:



$$d(x) = \sum_{i=1}^{N} w_i x_i + w_{n+1}$$

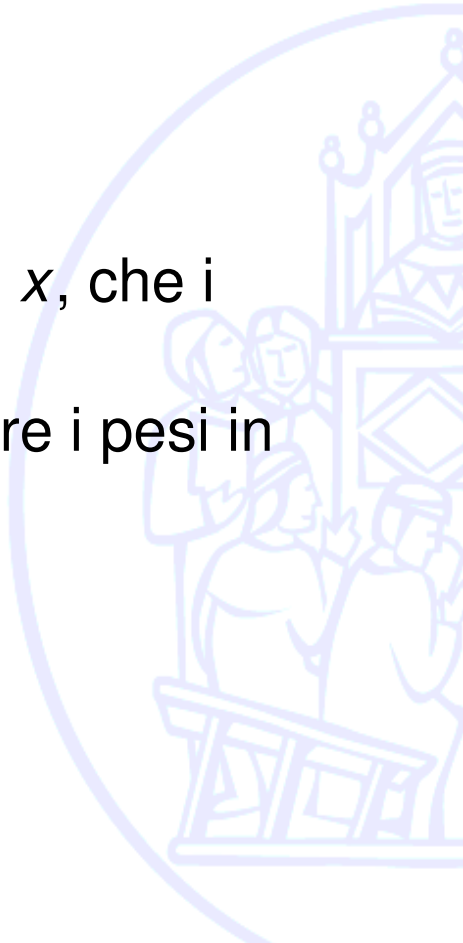
Tutto il gioco consiste nell'individuare i pesi della rete.

Addestramento

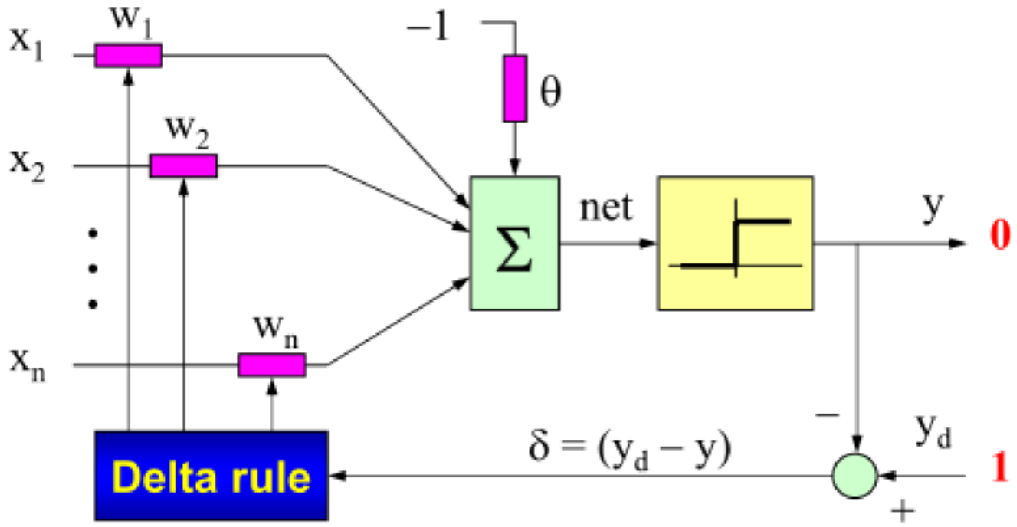
- supervisionato
- competitivo
- apprendimento con rinforze

Addestramento supervisionato

Vengono presentati alla rete sia i pattern di ingresso x, che i pattern di uscita desiderati y_d.
Una legge associativa di apprendimento dovrà variare i pesi in modo da far coincidere ingressi ed uscite.



Una semplice regola di apprendimento è quella lineare, chiamata *delta rule* perchè utilizza la differenza tra uscita desiderata e reale $\delta = y_d - y$.



con l'aggiornamento dei pesi:

$$w_i(t+1) = w_i(t) + \mu\delta x_i \tag{8}$$

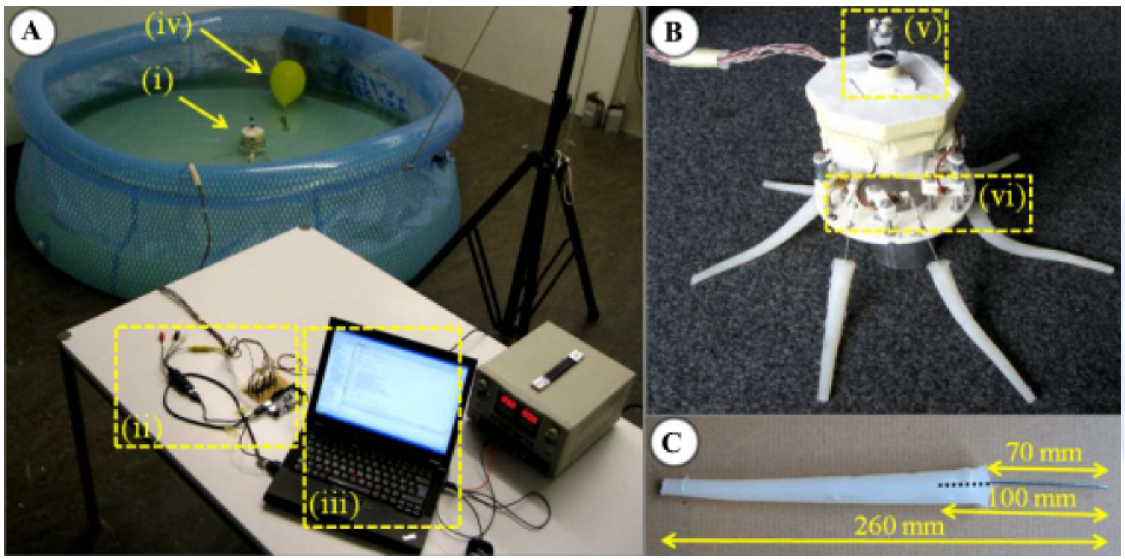
L'algoritmo può essere composto dai seguenti passi:

1. Fissare un set di addestramento di M esempi: $A = \{(X_k, y_{dk}), k = 1, \ldots, M\}$
2. Inizializzare i pesi con valori casuali w_i
3. Prendere una coppia (X_k, y_{dk}) come ingresso
4. Calcolare l'uscita della rete neurale y_k;
5. Aggiornare i pesi con la regola del delta: $(\Delta w = \delta dx)$
6. Ripetere il ciclo dallo step 3 finché tutte le uscite sono corrette: $y_k = y_{dk} \forall k \in [1, M]$

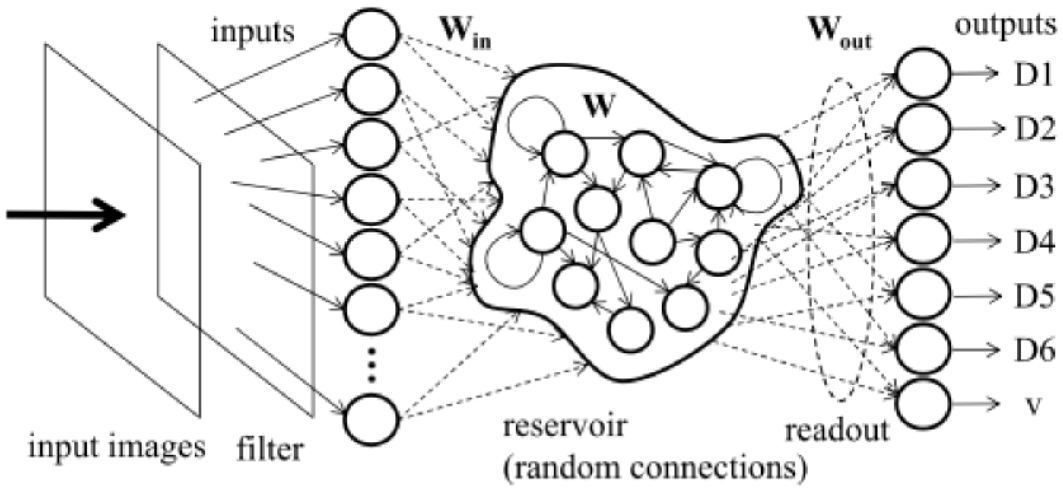
1. E' norma prendere una piccola parte di A per costruire un set di validazione V, sul quale viene testata la rete per valutarne le performance
2. Una volta che la rete è addestrata correttamente (l'errore con V è accettabile) la rete può essere utilizzata come classificatore

Un esempio classico è impiegare il percettrone come classificatore di numeri (o caratteri).
Data un'immagine composta da $n \cdot m$ pixel, è possibile costruire una rete con $n \cdot m$ ingressi e 10 uscite che rappresentino i numeri da 0 a 9.
Nonostante la semplicità, il percettrone distingue correttamente i numeri, anche in presenza di distorsione o rumore.

Reti più complesse possono essere utilizzate anche per apprendere una coordinazione diretta sensoro-motoria.

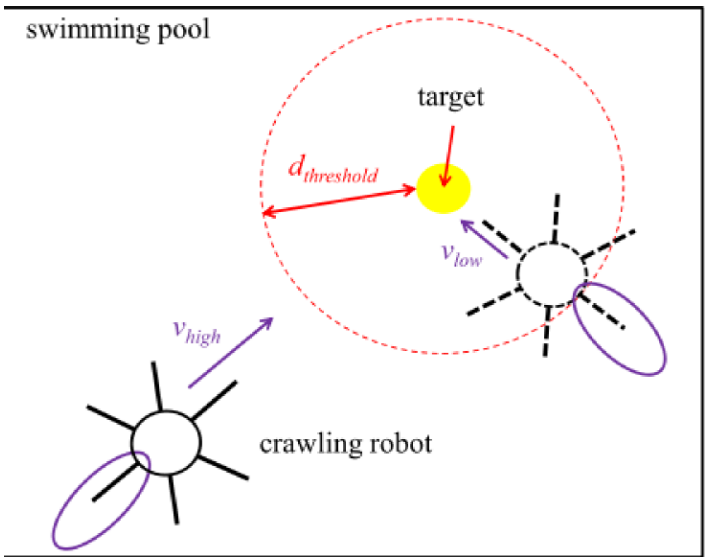


1. l'immagine viene acquisita
2. si estrae il target
3. si associa la direzione



Il training è effettuato sul robot mentre avviene l'azione.

Il comportamento risultante è il seguente:



Esponenziale e sigmoide

```
img=imread('castle_sign.jpg');
img=img(:,:,1);
figure();
imshow(img);
[r c]=size(img);
img_mod=zeros(r,c);
gamma=0.6;
e=2.7182818284590;
%e=1.05;
t=128;
for i=1:r
    for j=1:c
       % esponenziale
       %tmp=double(img(i,j))^gamma;
       %img_mod(i,j)=(256-1)^(1-gamma)*double(img(i,j))^gamma;
       % sigmoide
       tmpimg=double(img(i,j))-t;
       tmp=256./(1+e.^double(-tmpimg));
       img_mod(i,j)=tmp;
    end
end
figure();
imshow(img_mod);
```

Otsu

```
img=imread('castle_sign.jpg');
img=img(:,:,1); figure(); imshow(img);
[r c]=size(img); occ=isto_esercizio(img);
for t=1:256 sum_tmp=0; sum_sf=0;
media_sf=0; sum_ogg=0; media_ogg=0;
peso_sf=0; peso_ogg=0; max=0;
    for i=1:t
        sum_tmp=occ(i)*i+sum_tmp;
        sum_sf=occ(i)+sum_sf;
    end
    media_sf=sum_tmp/sum_sf; sum_tmp=0;
    for j=t+1:256
       sum_tmp=occ(j)*j+sum_tmp;
       sum_ogg=occ(j)+sum_ogg;
    end
    media_ogg=sum_tmp/sum_ogg;
    peso_sf=sum_sf/(sum_ogg+sum_sf);
    peso_ogg=sum_ogg/(sum_ogg+sum_sf);
    var=peso_sf*peso_ogg*(media_sf-media_ogg)^2;
    if var>max
        max=var;
        soglia=t;
    end
end
```

Erosione

```
function erosione_esercizio(a)
l=7;
mask_ero=ones(l*2+1);
figure()
imshow(a);
[r c]=size(a);
img_erosa=zeros(r,c);

for i=1+(l+1):r-(l+1)
    for j=1+(l+1):c-(l+1)
        %conv=conv2(mask_ero,a);
        tmp_conv=0;
        for u=-l:l
            for v=-l:l
                tmp_conv=tmp_conv+mask_ero(l+1+u,l+1+v)*a(i+u,j+v);
            end
        end
        if tmp_conv==255*((l*2+1)^2)
            img_erosa(i,j)=255;
        end
    end
end

figure()
imshow(img_erosa);
end
```

```
img=imread('retta3.png');
imshow(img);  img2=img(:,:,1);

[r c]=size(img2);

acc=zeros(1,1000);
tmp=zeros(1,1000);
for i=1:r
   for j=1:c
       if img2(i,j)>=225
          for a=1:1000
              tmp(a)=-a*j/100+i;
          end
          acc=cat(1,acc,tmp);
       end
   end
end

figure();
[rig col]=size(acc);
for ii=2:20:rig
   plot(acc(ii,:));
   hold on;
end
```

Correlation Match 1

```
sfondo=imread('sfondo_piccolo.png');
colormap(gray(256));
sfondo=sfondo(:,:,1);
image(sfondo)
match=imread('match_piccolo.png');
match=match(:,:,1);
figure()
colormap(gray(256));
image(match);
[r c]=size(sfondo);
[rm cm]=size(match);
img_conv=zeros(r,c);
nor=0;

for k=1:rm
    for u=1:cm
    nor=nor+double(match(k,u));
    end
end
```

Correlation Match 2

```
for i=1+rm:r-rm
   for j=1+cm:c-cm
       tmp=0;
       for ii=1:rm
           for jj=1:cm
               tmp_match=match(ii,jj);
               tmp_sfondo=sfondo(i-rm+ii,j-cm+jj);
              tmp=tmp+1/nor*double(match(ii,jj))*double(sfondo(i-rm+ii,j-cm+jj));
           end
       end
       img_conv(i,j)=tmp;
   end
end

figure()
colormap(gray(256));
image(img_conv);
```

Correlation Match 3

```
max=0;
for i=1:r
   for j=1:c
       if img_conv(i,j)>=max
            max=img_conv(i,j);
       end
   end
end

for i=1:r
   for j=1:c
       if img_conv(i,j)>=max-15
            img_conv(i,j)=255;
       else
            img_conv(i,j)=0;
       end
   end
end

figure()
colormap(gray(256));
image(img_conv);
```