



ELABORAZIONE DELLE IMMAGINI

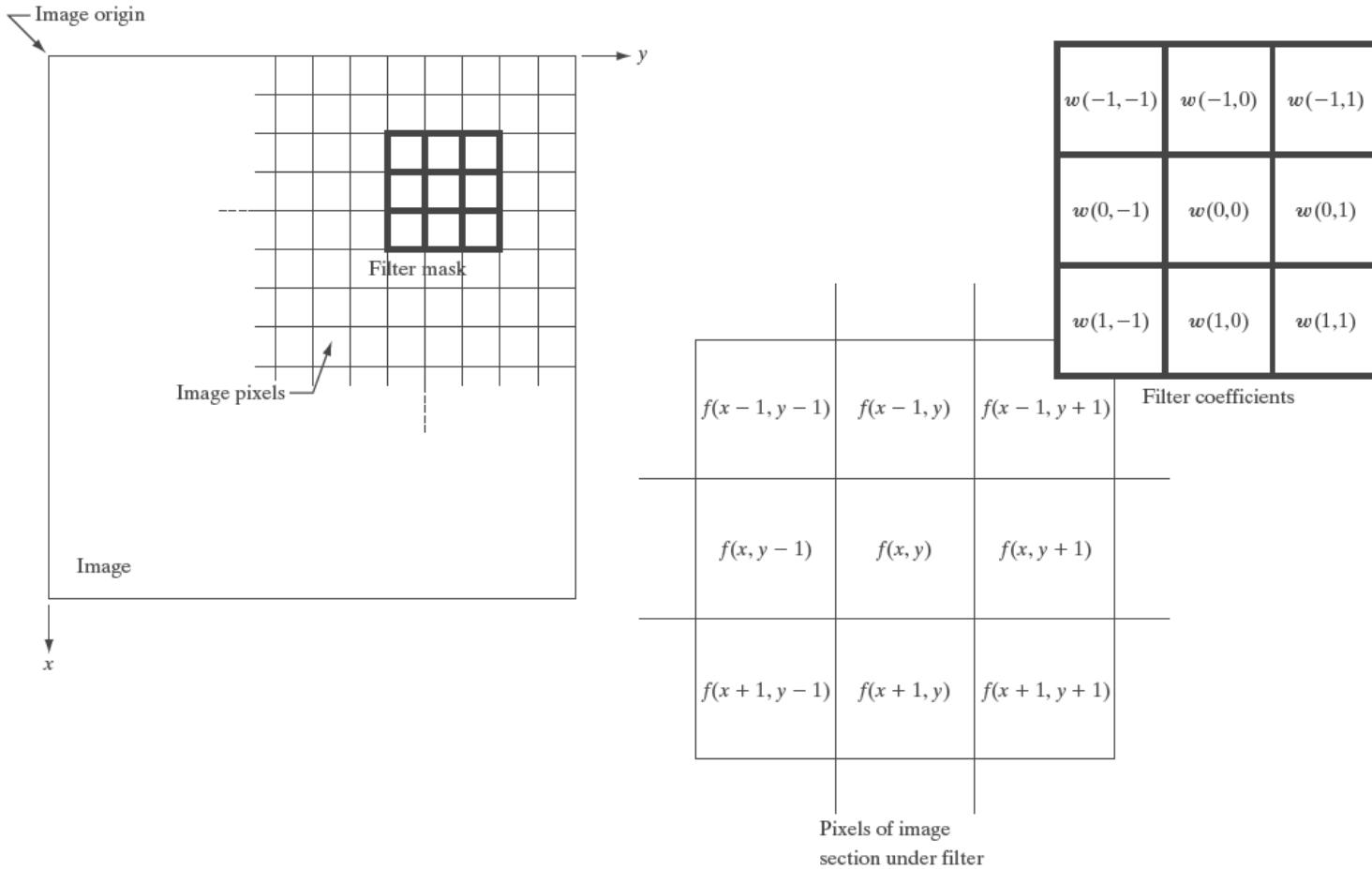
FILTRAGGIO SPAZIALE

FILTRAGGIO SPAZIALE

- Le tecniche di **filtraggio spaziale** operano sui pixel di un'immagine prendendo in considerazione i valori di intensità in un intorno (**neighborhood**)
 - Per **ogni pixel** dell'**immagine originale**, è calcolata l'**intensità** del pixel corrispondente nell'**immagine filtrata**
 - La **regola di trasformazione** spesso è descritta da una **matrice**, chiamata **filtro** (maschera o kernel), della stessa dimensione dell'**intorno**
 - Se la regola di **trasformazione** è una funzione **lineare** delle intensità nell'intorno, la tecnica è chiamata **filtraggio lineare spaziale** (altrimenti, non lineare)



FILTRAGGIO LINEARE



Elim Parte II – Prof. A. Ferone

FILTRAGGIO LINEARE

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

FILTRAGGIO LINEARE

- Il **pixel** nell'**immagine filtrata**, $g(x,y)$, è ottenuto come **combinazione lineare** dei pixel nell'immagine originale, $f()$, in un intorno di (x, y) :

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

- La matrice peso, $w()$, è il filtro (maschera o kernel)
- Per comodità, spesso è usata una **matrice** con un numero **dispari** di righe $2a+1$, e colonne, $2b+1$

CORRELAZIONE E CONVOLUZIONE 2-D

- Correlazione

$$g(x, y) = w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x+s, y+t)$$

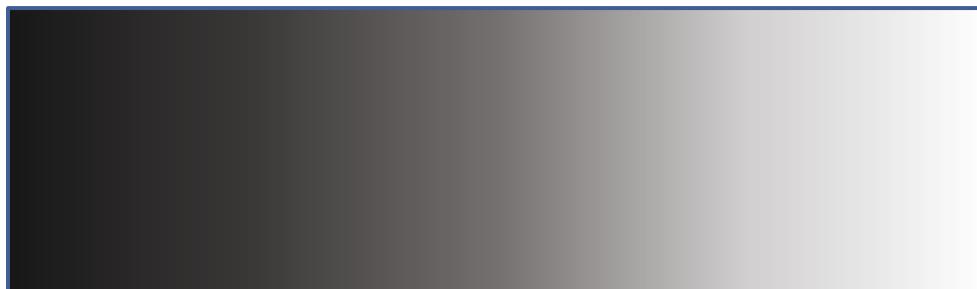
- Convoluzione

$$g(x, y) = w(x, y) * f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x-s, y-t)$$

SHARPENING

- Il termine **sharpening** si riferisce alle tecniche adatte ad **evidenziare** le **transizioni** di intensità
- Nelle immagini, i bordi (**edge**) tra gli **oggetti** sono percepiti per il **cambio di intensità**: più nette sono le transizioni di intensità e più l'immagine è percepita in modo definito
- La transizione di intensità tra i pixel adiacenti è connessa alla **derivata dell'immagine** in quella posizione
- Definiremo gli **operatori di sharpening** attraverso l'uso delle **derivate**, in cui la qualità della **risposta** è **proporzionale** al grado di **variazione di intensità** nel punto in cui è applicato l'operatore.

FILTRAGGIO



DERIVATA PRIMA DI UN'IMMAGINE

- Poiché l'immagine è una funzione discreta, la definizione classica di derivata non può essere applicata
- È necessario definire un operatore che soddisfi le principali **proprietà della derivata prima**:
 1. Uguale a zero dove l'intensità è costante
 2. Diversa da zero per una transizione di intensità
 3. Costante sulle rampe in cui la transizione di intensità è costante
- L'operatore di derivazione naturale è la differenza tra l'intensità dei pixel vicini (**differenziazione spaziale**)

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

Elim Parte II – Prof. A. Ferone

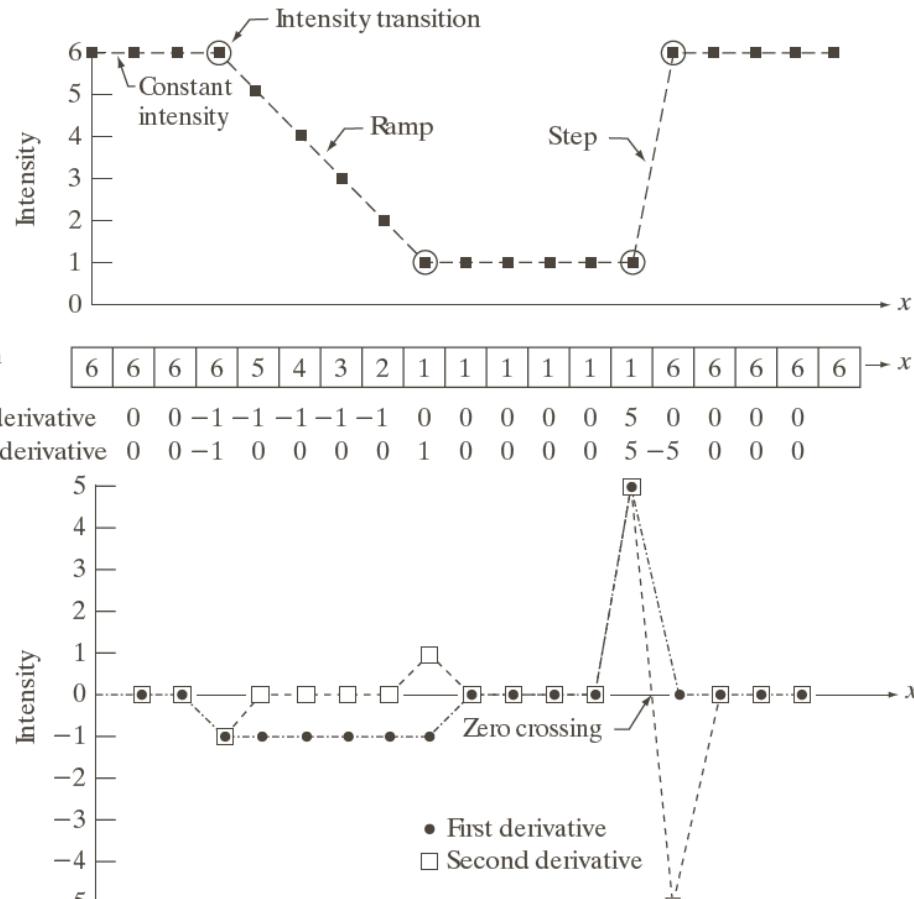
DERIVATA SECONDA DI UN'IMMAGINE

- Analogamente, l'operatore di **derivata seconda** può essere definito come:

$$\begin{aligned}\frac{\partial^2 f}{\partial x^2} &= f(x+1) - f(x) - (f(x) - f(x-1)) \\ &= f(x+1) - 2f(x) + f(x-1)\end{aligned}$$

- Soddisfa le seguenti **proprietà**:
 1. È uguale a zero dove l'intensità è costante
 2. È diverso da zero all'inizio di un passo (o rampa) di intensità
 3. È uguale a zero sulle pendenze costanti delle rampe
- $\frac{\partial^2 f}{\partial^2 x}$ è definita usando i pixel precedente e successivo

ESEMPIO: DERIVATA DISCRETA



Elim Parte II – Prof. A. Ferone

LAPLACIANO

- Il **Laplaciano** implementa una **derivata del secondo ordine 2D**
 - Definire una formulazione discreta della derivata seconda
 - Costruire una maschera filtro che la implementi
- In particolare vogliamo realizzare **filtri isotropici** la cui risposta è **indipendente** dalla **direzione** delle **disconitnuità** dell'immagine (invarianti per rotazione)
- Il Laplaciano è l'operatore derivativo isotropico più semplice, definito per un'immagine $f(x,y)$

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

FILTRO LAPLACIANO

- In un'immagine digitale, le derivate seconde rispetto a x e y sono calcolate come:

$$\begin{aligned}\frac{\partial^2 f}{\partial x^2} &= f(x+1, y) - 2f(x, y) + f(x-1, y) \\ \frac{\partial^2 f}{\partial y^2} &= f(x, y+1) - 2f(x, y) + f(x, y-1)\end{aligned}$$

- Quindi, il Laplaciano risulta:

$$\begin{aligned}\nabla^2 f(x, y) &= f(x+1, y) + f(x-1, y) + f(x, y+1) \\ &\quad + f(x, y-1) - 4f(x, y)\end{aligned}$$

- Può essere considerata anche la derivata lungo la diagonale:

$$\begin{aligned}\nabla^2 f(x, y) &+ f(x-1, y-1) + f(x+1, y+1) \\ &+ f(x-1, y+1) + f(x+1, y-1) - 4f(x, y)\end{aligned}$$

Elim Parte II – Prof. A. Ferone

FILTRO LAPLACIANO

0	1	0
1	-4	1
0	1	0

$$\begin{aligned}\nabla^2 f(x, y) = & f(x+1, y) + f(x-1, y) + f(x, y+1) \\ & + f(x, y-1) - 4f(x, y)\end{aligned}$$

Filtro Laplaciano invariante alle rotazioni di 90°

1	1	1
1	-8	1
1	1	1

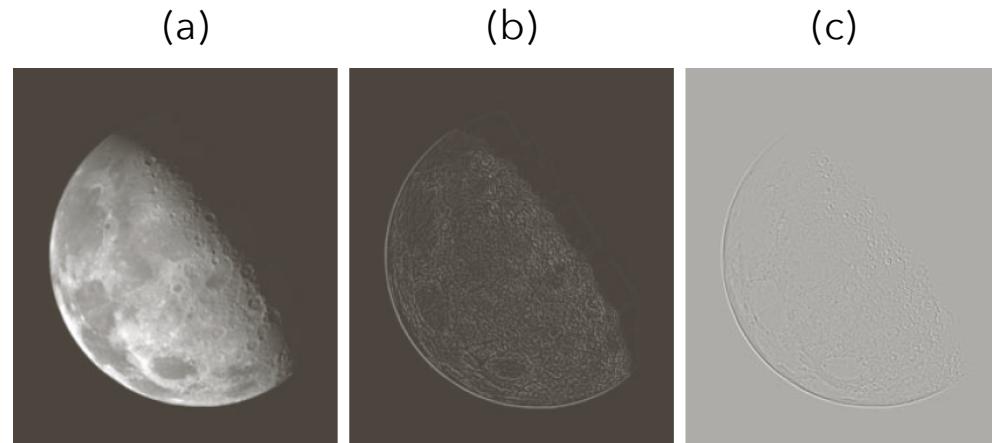
$$\begin{aligned}\nabla^2 f(x, y) = & f(x-1, y-1) + f(x+1, y+1) \\ & + f(x-1, y+1) + f(x+1, y-1) - 4f(x, y)\end{aligned}$$

Filtro Laplaciano invariante alle rotazioni di 45°

Elim Parte II – Prof. A. Ferone

ESEMPIO: FILTRO LAPLACIANO

- Il Laplaciano spesso ha valori negativi
- Per visualizzarlo, deve essere opportunamente scalato nell'intervallo di rappresentazione $[0, L-1]$



- (a) Immagine originale, (b) il suo Laplaciano, (c) il suo Laplaciano scalato in modo che 0 sia visualizzato come livello di grigio intermedio

ESEMPIO: FILTRO LAPLACIANO

- Sottraendo il Laplaciano (o una sua frazione) dall'immagine, l'altezza del gradino è aumentata

$$g = f + c \nabla^2 f, -1 \leq c \leq 0$$



(a)



(b)



(c)

- (a) Immagine originale (b) filtrato con Laplaciano 45° (c) filtrato con Laplaciano 90°

Elim Parte II – Prof. A. Ferone

LAPLACIANO IN OPENCV

- In OpenCV la funzione che implementa il Laplaciano è

```
void cv::Laplacian(  
    cv::InputArray src,           // Input image  
    cv::OutputArray dst,          // Result image  
    int ddepth,                  // Depth of output image (e.g., CV_8U)  
    cv::Size ksize = 3,           // Kernel size  
    double scale = 1,             // Scale applied before assignment to dst  
    double delta = 0,             // Offset applied before assignment to dst  
    int borderType = cv::BORDER_DEFAULT // Border extrapolation to use  
);
```

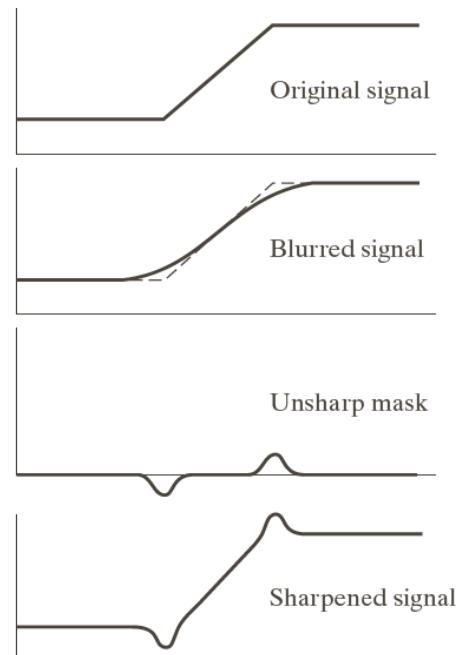
UNSHARP MASKING

- E' un metodo di uso comune in grafica per rendere le immagini più nitide
- Consiste:
 1. Sfocare l'immagine originale (smoothing)
 2. Ottenere una maschera come differenza tra l'immagine originale e quella sfocata
 3. Aggiungere la maschera all'immagine originale
- Il processo può essere formalizzato come:

$$g = f + k \cdot (f - f * h)$$

UNSHARP MASKING

$$g = f + k \cdot (f - f * h)$$



DIP-XE

Immagine originale

DIP-XE

Immagine sfocata

DIP-XE

Maschera di unsharpening

DIP-XE

Immagine unsharped ($k \leq 1$)

DIP-XE

Immagine highboosted ($k > 1$)

Elim Parte II – Prof. A. Ferone

GRADIENTE

- La **derivata prima** viene implementata con un'approssimazione del **gradiente**
- Il gradiente è un **vettore** formato dalle sue **derivate parziali**

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- Il vettore **gradiente punta** verso la **direzione di massima variazione**
- La **magnitudo del gradiente**, $M(x, y)$ è un'immagine delle stesse dimensioni di $f()$

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

$$M(x, y) \approx |g_x| + |g_y|$$

Elim Parte II – Prof. A. Ferone

OPERATORI DI DERIVAZIONE

- Definizioni base:

$$g_x(x, y) = f(x + 1, y) - f(x, y)$$

$$g_y(x, y) = f(x, y + 1) - f(x, y)$$

$$g_x: \begin{array}{|c|c|} \hline -1 & 1 \\ \hline \end{array}$$

$$g_y: \begin{array}{|c|} \hline -1 \\ \hline 1 \\ \hline \end{array}$$

- Operatori di Roberts:

$$g_x(x, y) = f(x + 1, y + 1) - f(x, y)$$

$$g_y(x, y) = f(x, y + 1) - f(x - 1, y)$$

$$g_x: \begin{array}{|c|c|} \hline -1 & 0 \\ \hline 0 & 1 \\ \hline \end{array}$$

$$g_y: \begin{array}{|c|c|} \hline 0 & -1 \\ \hline 1 & 0 \\ \hline \end{array}$$

Elim Parte II – Prof. A. Ferone

OPERATORI DERIVATIVI

- Operatori di Sobel:

$$\begin{aligned}g_x(x, y) = & -f(x-1, y-1) - 2f(x-1, y) \\& - f(x-1, y+1) + f(x+1, y-1) \\& + 2f(x+1, y) + f(x+1, y+1)\end{aligned}$$

$$\begin{aligned}g_y(x, y) = & -f(x-1, y-1) - 2f(x, y-1) \\& - f(x+1, y-1) + f(x-1, y+1) \\& + 2f(x, y+1) + f(x+1, y+1)\end{aligned}$$

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

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

Elim Parte II – Prof. A. Ferone

OPERATORI DERIVATIVI

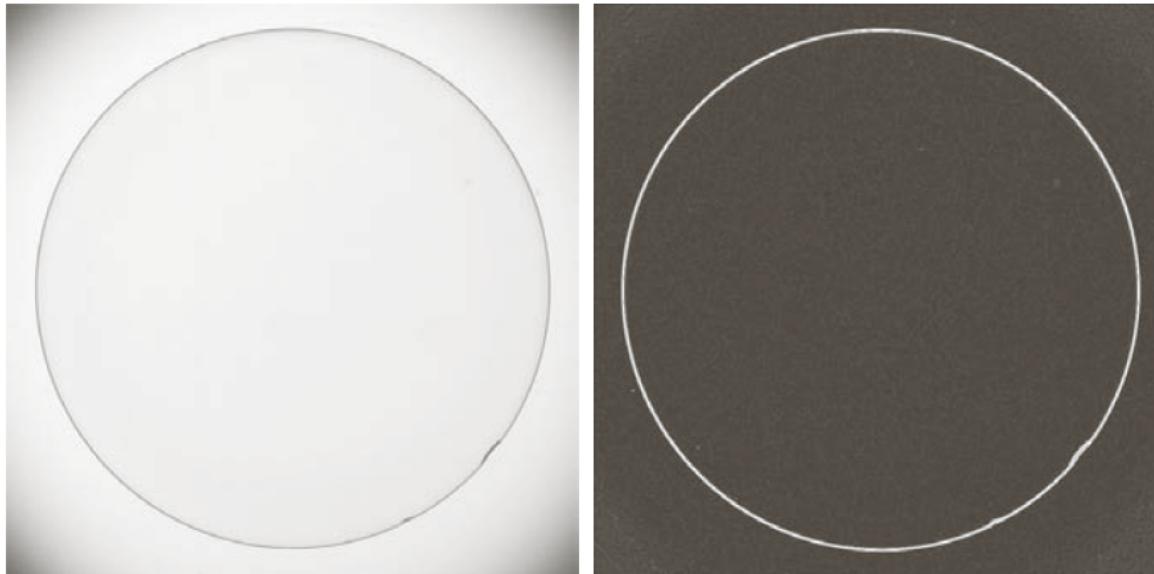
- Operatori di Prewitt:

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

ESEMPIO: APPLICAZIONE BASATA SU GRADIENTE

- Il filtraggio di **Sobel** riduce la visibilità di quelle regioni in cui l'intensità cambia lentamente, permettendo di **evidenziare i dettagli** (rendendo l'individuazione dei difetti più semplice per un'elaborazione automatica)



Elim Parte II – Prof. A. Ferone

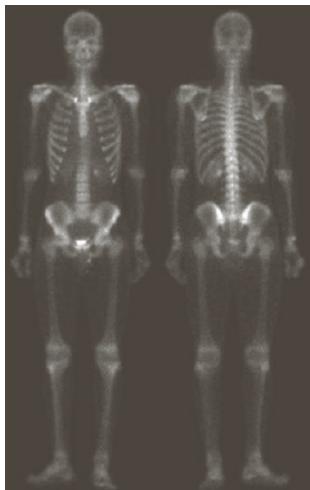
SOBEL IN OPENCV

- In OpenCV la funzione che implementa il filtro di Sobel è

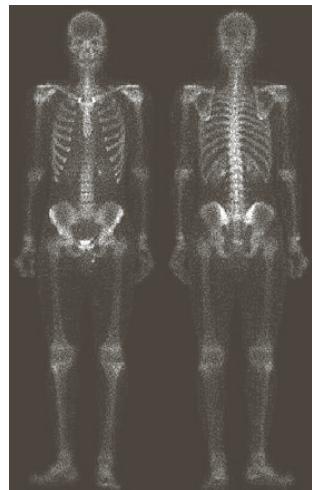
```
void cv::Sobel(  
    cv::InputArray src,           // Input image  
    cv::OutputArray dst,         // Result image  
    int ddepth,                 // Pixel depth of output (e.g., CV_8U)  
    int xorder,                 // order of corresponding derivative in x  
    int yorder,                 // order of corresponding derivative in y  
    cv::Size ksize = 3,          // Kernel size  
    double scale = 1,            // Scale (applied before assignment)  
    double delta = 0,            // Offset (applied before assignment)  
    int borderType = cv::BORDER_DEFAULT // Border extrapolation  
);
```

- xorder e yorder servono a specificare l'ordine della derivata lungo la direzione x e y

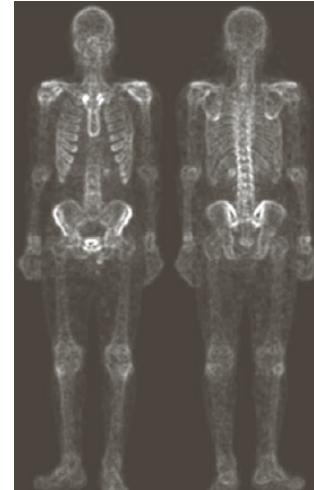
METODI COMBINATI



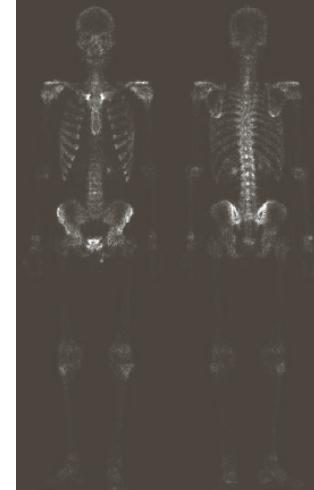
(a)



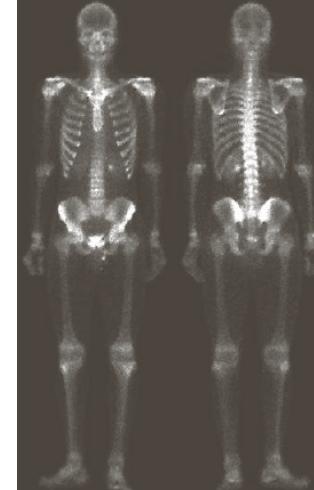
(b)



(c)



(d)



(e)



(f)

ESERCIZI

- Implementare il Laplaciano con kernel isotropico a 45° e 90° utilizzando la funzione di correlazione/convoluzione (o filter2D())
 - Per normalizzare i livelli di grigio è possibile usare la funzione normalize()
 - **normalize(src, dst, 0, 255, NORM_MINMAX, CV_8U);**
- Implementare il filtro di Sobel (g_x e g_y) utilizzando la funzione di correlazione/convoluzione (o filter2D())
 - Calcolare la risposta di entrambi i filtri
 - Calcolare la magnitudo del gradiente (entrambe le formulazioni)
- Utilizzare la risposta ottenuta per effettuare lo sharpening di un'immagine

OPERAZIONI UTILI

- $\text{dst} = \text{src1} +/\!- \text{src2}$
- $\text{dst} = \text{src} * k$
- $\text{dst} = \text{abs(src)}$
- pow(src,2,dst)
- sqrt(src,dst)