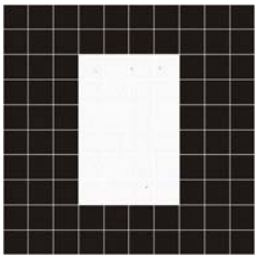
Esame del corso di
Tecniche Avanzate per il Trattamento delle Immagini

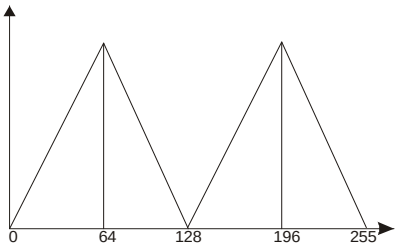
Data: 18 Settembre 2007

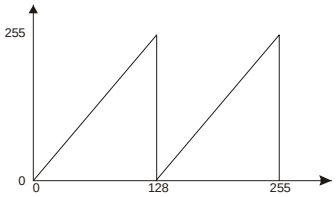
Es.1 [pt. **5**]: Nella figura (10x10 pixel) riportata a fianco il rettangolo bianco è di dimensioni 6x4 pixel. Indicando con 1 il valore del bianco e con zero il valore del nero indicare il valore di grigio che si otterrebbe in ciascun pixel dell'immagine applicando:

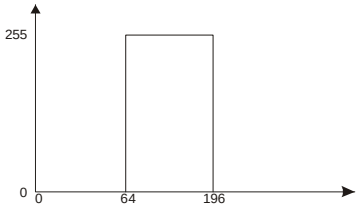


1. un filtro Medio (media aritmetica) 3x3.
2. un filtro Mediano 3x3.
3. un filtro Laplaciano 3x3.

Es.2 [pt. **9**]: Un'immagine presenta l'istogramma riportato a fianco. Indicare come si trasforma tale istrogramma applicando all'immagine le seguenti trasformazioni di intensità (funzioni di trasferimento):

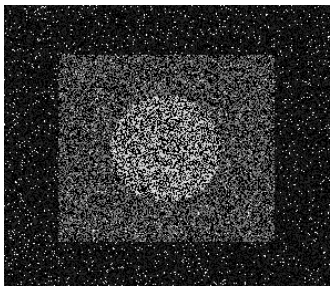


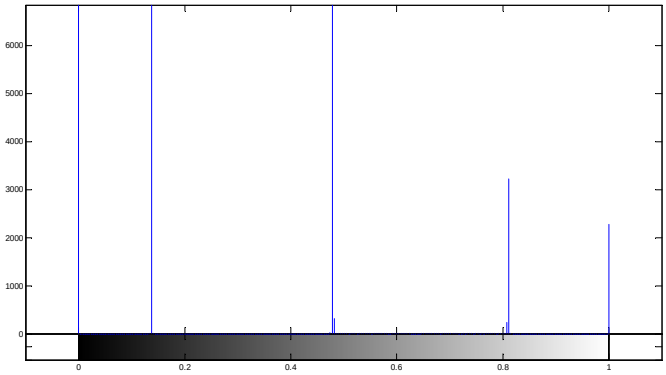




Indicare, inoltre, la trasformazione che consente di equalizzare l'istogramma dell'immagine di partenza.

Es.3[pt. **6**]: Si consideri la seguente immagine che presenta l'istogramma riportato a destra. Descrivere la distribuzione





statistica del rumore e, previa l'assunzione di opportune ipotesi, discutere un possibile criterio di filtraggio.

Es.4[pt. **8**]**:** Si descriva una procedura con la quale sia possibile scomporre un'immagine truecolor A raffigurante tre oggetti di differente colorazione media ottenendo 3 immagini distinte (di dimensioni size(A,1) x size(A,2)), ognuna contenente, al meglio, uno solo dei 3

oggetti nella propria posizione, nella scala e nella colorazione originarie. Si implementino i passi necessari mediante codice Matlab.

Es.5[pt. **6**]: Data un'immagine truecolor memorizzata in un file immagine.bmp ed avente i dati rappresentati ad 8bit rispettivamente nel piano R, piano G e piano B, si implementino i seguenti punti mediante codice Matlab:
Leggere, caricare nel workspace e visualizzare l'immagine.
Convertire l'immagine di partenza ad una contenente per ogni pixel il solo valore di luminanza (calcolato mediante la formula 0.2989 * R + 0.5870 * G + 0.1140 * B).
Calcolare la trasformata di Fourier mediante FFT dell'immagine ottenuta al passo precedente e visualizzare modulo, fase e spettro di potenza ottenuti.
Fare in modo che il valore del modulo della componente continua, risultante dalla FFT, appaia al centro dei rispettivi grafici (e, di conseguenza, i valori della altre frequenze spaziali).
Si visualizzino modulo e spettro di potenza ottenuti al passo precedente mediante una scala capace di mitigare i valori più alti.
Ricostruire l'immagine nel dominio spaziale a partire dalla sola fase calcolata al secondo passo riportando il range del suo istogramma a quello corretto per la rappresentazione originale dei suoi valori (non usare le funzioni di conversione dei tipi di dato o dei tipi di immagine predefinite es. `im2double(), im2uint8(), double(), uint8(),` ecc.).

Nomi funzioni Matlab:

```
zeros
size
kmeans
floor
mod
figure
imshow
imread
getimage
rgb2gray
fft2
im2double
abs
angle
real
imag
imagesc
double
fftshift
log
complex
cos
sin
ifft2
min
max
```

Soluzioni

Soluzione Es1

Filtrando con un filtro medio del tipo: $\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$ si ottiene:

0	0	0	0	0	0	0	0	0	0
0	0	0.1111	0.2222	0.3333	0.3333	0.2222	0.1111	0	0
0	0	0.2222	0.4444	0.6667	0.6667	0.4444	0.2222	0	0
0	0	0.3333	0.6667	1	1	0.6667	0.3333	0	0
0	0	0.3333	0.6667	1	1	0.6667	0.3333	0	0
0	0	0.3333	0.6667	1	1	0.6667	0.3333	0	0
0	0	0.3333	0.6667	1	1	0.6667	0.3333	0	0
0	0	0.2222	0.4444	0.6667	0.6667	0.4444	0.2222	0	0
0	0	0.1111	0.2222	0.3333	0.3333	0.2222	0.1111	0	0
0	0	0	0	0	0	0	0	0	0

Mentre filtrare con un filtro mediano 3x3 porta ad una soluzione del tipo:

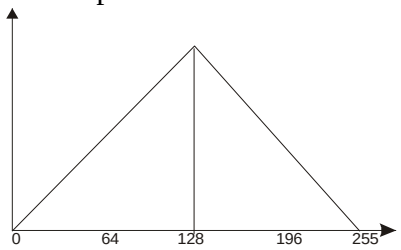
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Con un filtro laplaciano del tipo $\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix}$ si ottiene:

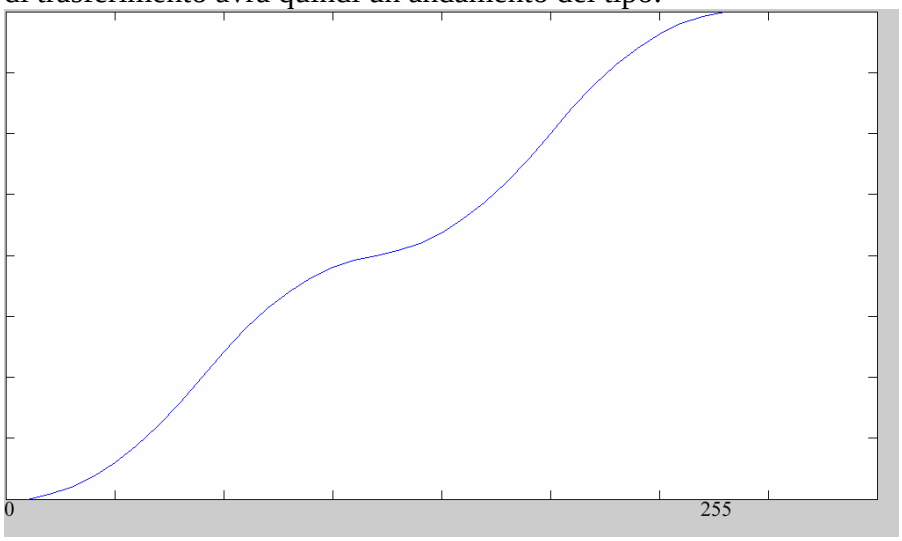
0	0	0	0	0	0	0	0	0	0
0	0	1	2	3	3	2	1	0	0
0	0	2	-5	-3	-3	-5	2	0	0
0	0	3	-3	0	0	-3	3	0	0
0	0	3	-3	0	0	-3	3	0	0
0	0	3	-3	0	0	-3	3	0	0
0	0	3	-3	0	0	-3	3	0	0
0	0	2	-5	-3	-3	-5	2	0	0
0	0	1	2	3	3	2	1	0	0
0	0	0	0	0	0	0	0	0	0

Soluzione Es2

Con la prima funzione di trasferimento l'istogramma risultante risulta pari a:



Mentre nel secondo caso l'istogramma risultante è concentrato in due valori di 0 e 255 di uguale ampiezza.
L'equalizzatore dell'istogramma sarà proporzionale all'integrale dell'istogramma iniziale e la funzione di trasferimento avrà quindi un andamento del tipo:



Soluzione Es3

La figura presenta rumore "sale e pepe" con valori 0 e 1, la statistica è di tipo impulsivo (bernoulliano) con probabilità di circa 0.75 per quanto riguarda il valore di 0 e 0.25 per quanto riguarda il valore 1. Diverse tecniche possono essere utilizzate per rimuovere tale tipo di rumore, da un filtro mediano ad un algoritmo che, analizzando l'intera immagine, nel caso in cui trovi un pixel di valore pari a 0 o 1 lo sostituisca con uno di valore più appropriato(ad es. media tra i primi vicini che presentino valori diversi da 0 o 1).

Soluzione Es4

E' possibile applicare una procedura di tipo K-Means (algoritmo di clustering non-supervisionato) ai pixel dell'immagine truecolor per partizionarli in 3 gruppi in base ai loro valori RGB.

- Passi algoritmo:
  - Stima iniziale centroidi $\mu^i \quad i = 1,...,C$
  - Esegui:
    - Assegna ogni x_j ad un cluster i in base al centroide più vicino (secondo la $d(x_j, \mu^i)$) → x_j^i
    - Ricalcola centroidi μ^i come media delle distanze dai x_j^i assegnati ai rispettivi clusters $i = 1,...,C$

  Fino a quando non si ha variazione significativa dei centroidi μ^i tra due iterazioni successive
- N.B.: la stima iniziale dei centroidi può essere random od in base ad un altro criterio

Realizzazione in Matlab:

- **Lettura lessicografica dell'immagine → linearizzazione in un vettore dei valori RGB di ogni pixel**

```
A_vect = zeros(size(A, 1)*size(A, 2), 3);
index = 1;
for i = 1:size(A, 1)
    for j = 1:size(A, 2)
        A_vect(index,:) = [A(i,j,1) A(i,j,2) A(i,j,3)];
        index = index + 1;
    end
end
```

- **Applicazione kmeans (3 clusters)**

```
clust_idx = kmeans(A_vect, 3);
```

- **Lettura degli elementi di ogni cluster e creazione delle immagini corrispondenti**

```
A_clust_1 = zeros(size(A,1), size(A,2), 3);
A_clust_2 = zeros(size(A,1), size(A,2), 3);
A_clust_3 = zeros(size(A,1), size(A,2), 3);
for i = 1:size(clust_idx, 1)
    if (clust_idx(i) == 1)
        A_clust_1(1+floor(i / size(A, 2)), 1+mod(i,size(A, 2)), :) =
A(1+floor(i / size(A, 2)), 1+mod(i,size(A, 2)), :);
    else if (clust_idx(i) == 2)
        A_clust_2(1+floor(i / size(A, 2)), 1+mod(i,size(A, 2)) , :) =
A(1+floor(i / size(A, 2)), 1+mod(i,size(A, 2)));
        else
        A_clust_3(1+floor(i / size(A, 2)), 1+mod(i,size(A, 2)) , :) =
A(1+floor(i / size(A, 2)), 1+mod(i,size(A, 2)), :);
        end
    end
end
```

- **Eventuale visualizzazione delle 3 immagini ottenute**

```
figure
imshow(A_clust_1); title('CLUSTER 1 - Clustering k-Means con 3 clusters');
figure
imshow(A_clust_2); title('CLUSTER 2 - Clustering k-Means con 3 clusters');
figure
imshow(A_clust_3); title('CLUSTER 3 - Clustering k-Means con 3 clusters');
```

Soluzione Es5

img_rgb = imread('immagine.bmp');
figure; imshow(img_rgb);
(oppure: figure
imshow('immagine.bmp');
img_rgb = getimage;)

```
img_gray = rgb2gray(img_rgb);

img_gray_FFT = fft2(im2double(img_gray));
img_gray_FFT_magnitude = abs(img_gray_FFT);
img_gray_FFT_phase = angle(img_gray_FFT);
img_gray_FFT_power_spectrum = real(img_gray_FFT).^2 + imag(img_gray_FFT).^2;
figure
imagesc(img_gray_FFT_magnitude)); colorbar; axis equal; axis tight; title('LENA -
Modulo spettro immagine');
figure
```

```
imagesc(img_gray_FFT_phase); colorbar; axis equal; axis tight; title('LENA - Fase
spettro immagine');
figure
imagesc(img_gray_FFT_power_spectrum)); colorbar; axis equal; axis tight;
title('LENA - Spettro di potenza immagine');
```

(nel dominio dei tempi:

```
      for i = 1:size(img_gray, 1)
          for j = 1:size(img_gray, 2)
              img_gray_shift(i,j) = double(img_gray(i,j))*((-1)^(i+j));
          end
      end
      img_gray_FFT = fft2(im2double(img_gray_shift));)
```

nelle frequenze:

```
img_gray_FFT = fftshift(img_gray_FFT);

img_gray_FFT_magnitude = abs(img_gray_FFT);
img_gray_FFT_power_spectrum = real(img_gray_FFT).^2 + imag(img_gray_FFT).^2;
figure
imagesc(0.5*log(1+ img_gray_FFT_magnitude)); colorbar; axis equal; axis tight;
title('TREES - Modulo spettro immagine (trasformazione log)');
figure
imagesc(0.5*log(1+ img_gray_FFT_power_spectrum)); colorbar; axis equal; axis
tight; title('TREES - Spettro di potenza immagine (trasformazione log)');

img_gray_FFT_rec = complex(1.*cos(img_gray_FFT_phase),
1.*sin(img_gray_FFT_phase));
img_gray_rec = ifft2(img_gray_FFT_rec, 'symmetric');
img_gray_rec = (img_gray_rec -min(min(img_gray_rec))) / (max(max(img_gray_rec)) -
min(min(img_gray_rec)));
```