Elaborazione di Segnali Multimediali
a.a. 2017/2018

Elaborazione di immagini a colori
Soluzioni

1 Gli spazi di colore

1. *Lo spazio CMY e CMYK.*

```
function z = rgb2cmy(x);

x = double(x)/255;
c = 1 - x(:,:,1);
m = 1 - x(:,:,2);
y = 1 - x(:,:,3);
z = cat(3,c,m,y);
figure; imshow(c,[]); title('Ciano');
figure; imshow(m,[]); title('Magenta');
figure; imshow(y,[]); title('Giallo');
```

Notate che in questo modo le componenti C, M e Y sono normalizzate. E' anche possibile visualizzare le componenti in scala di colore, modificando opportunamente la palette:

```
L = [0:1/255:1]';             % definizione dei livelli di luminanza
figure; imagesc(c); axis image;
colormap([zeros(256,1) L L]); % creazione della palette per il Ciano
figure; imagesc(m); axis image;
colormap([L zeros(256,1) L]); % creazione della palette per il Magenta
figure; imagesc(y); axis image;
colormap([L L zeros(256,1)]); % creazione della palette per il Giallo
```

Per quanto riguarda invece lo spazio CMYK:

```
function z = rgb2cmyk(x);

x = double(x)/255;
k = 1 - max(x,[],3);
c = 1 - k - x(:,:,1);
m = 1 - k - x(:,:,2);
y = 1 - k - x(:,:,3);
z = cat(4,c,m,y,k);
figure; imshow(c,[]); title('Ciano');
figure; imshow(m,[]); title('Magenta');
figure; imshow(y,[]); title('Giallo');
figure; imshow(k,[]); title('Nero');
```

2 Tecniche per l'elaborazione

1. *Negativo.*

```
x = imread('colori.jpg');
figure(1); image(x); axis image;
y = 255-x;            % negativo
figure(2); image(y); axis image;
z = imcomplement(x);  % verifica
figure(3); image(z); axis image;
```

2. *Correzioni di toni e colori.*

```
clear all; close all; clc;
x = imread('colori.jpg');
figure(1); imshow(x); axis image;
x = double(x)/255;
% Elaborazione in RGB
gamma = 0.6;
y = x.^gamma;
y = y/max(y(:));
figure(2); imshow(y);
title('Elaborazione in RGB');
% Elaborazione in HSI
w = rgb2hsi(x);
w(:,:,3) = w(:,:,3).^gamma;
z = hsi2rgb(w);
z = z/max(z(:));
figure(3); imshow(z);
title('Elaborazione in HSI');
```

3. *Equalizzazione*.

```
x = imread('volto.tiff');
figure; imshow(x);
y = colorhisteq(x);
```

dove:

```
function y = colorhisteq(x);
x = double(x);

% Elaborazione in RGB
y(:,:,1) = histeq(x(:,:,1));
y(:,:,2) = histeq(x(:,:,2));
y(:,:,3) = histeq(x(:,:,3));
figure; imshow(y); title('Equalizzazione spazio RGB');

% Elaborazione in HSI
w = rgb2hsi(x);
w(:,:,3) = w(:,:,3)*255;
w(:,:,3) = histeq(w(:,:,3));
z = hsi2rgb(w);
z = z/max(z(:));
figure; imshow(z); title('Equalizzazione spazio HSI');

function y = histeq(x);

p = hist(double(x(:)), [0:255])/length(x(:));
cdf = cumsum(p);
y = cdf(uint8(x)+1);
```

Notate come sia possibile definire una funzione (`histeq`) all'interno di un'altra funzione `colorhisteq`.

4. *Color balancing*. Nello spazio CMY:

```
x = imread('foto.jpg');
subplot(1,3,1); imshow(x); title('troppo ciano');
x = double(x)/255;
c = 1 - x(:,:,1);
y(:,:,1) = c.^(1.8);
x(:,:,1) = 1 - y(:,:,1);
subplot(1,3,2); imshow(x); title('immagine elaborata');
w = imread('foto_originale.tif');
subplot(1,3,3); imshow(w); title('immagine originale');
```

E' anche possibile lavorare nello spazio RGB: per diminuire la quantita' di ciano bisogna aumentare il rosso:

```
x(:,:,1) = (x(:,:,1)).^(0.6);
```

5. *Filtraggio spaziale*.

```
x = imread('lenac.jpg');
x = imcrop(x,[50 50 300 300]);    % porzione dell'immagine
figure(1); imshow(x); title('immagine originale');
h = fspecial('average',[5 5]);
% filtraggio nello spazio RGB
y1 = imfilter(x,h,'replicate');
figure(2); subplot(1,2,1); imshow(y1);
title(sprintf('spazio RGB'));
% filtraggio nello spazio HSI
w = rgb2hsi(x);
w(:,:,3) = imfilter(w(:,:,3)*255,h,'replicate');
w(:,:,3) = w(:,:,3)/255;
y2 = hsi2rgb(w);
subplot(1,2,2); imshow(y2);
title(sprintf('spazio HSI'));
```

La funzione `imcrop(offsetc, offsetr, Nc, Nr)` permette di selezionare una porzione dell'immagine definendo le coordinate del pixel in alto a sinistra `(offsetr, offsetc)` e la dimensione del rettangolo `(Nr,Nc)`, non necessariamente coincidente con la dimensione della porzione di immagine (si veda l'help in linea per maggiori informazioni).

Per quanto riguarda l'enhancement mediante laplaciano, anziché determinare il laplaciano per poi sottrarlo all'immagine applichiamo direttamente la maschera che realizza entrambe queste operazioni.

```
x = imread('fiori.jpg');
figure(1); imshow(x); title('immagine originale');
h = [0 -1 0; -1 5 -1; 0 -1 0];
% filtraggio nello spazio RGB
y1 = imfilter(x,h,'replicate');
figure(2); imshow(y1);
title('Enhancement spazio RGB');
% filtraggio nello spazio HSI
w = rgb2hsi(x);
w(:,:,3) = imfilter(w(:,:,3)*255,h,'replicate');
w(:,:,3) = w(:,:,3)/255;
y2 = hsi2rgb(w);
figure(3); imshow(y2);
title('Enhancement spazio HSI');
```

2.1 Esercizi proposti

1. *Variazione del colore.*

```
clear all; close all; clc;
x = imread('azzurro.jpg');
hsi=rgb2hsi(x); figure(1);
H=hsi(:,:,1); subplot(1,3,1); imshow(H,[]);
S=hsi(:,:,2); subplot(1,3,2); imshow(S,[]);
I=hsi(:,:,3); subplot(1,3,3); imshow(I,[]);
mask=((H>0.35)&(H<0.66)&(S>0.20)&(S<0.80)&(I>0.35));
figure(2); imshow(mask,[]);
H=H+(mask*0.38);
hsi(:,:,1)=H;
y=hsi2rgb(hsi);
figure(3); subplot(1,2,1); imshow(x);
subplot(1,2,2); imshow(y);
```

2. *Equalizzazione.*

```
clear all; close all; clc;
x = imread('Tiffany.tiff');
figure(1); imshow(x); title('immagine originale');
[M,N,L] = size(x);
% correzione bordo
for b=1:3,
  x(M,:,b) = x(M-1,:,b);
  x(:,1,b) = x(:,3,b);
  x(:,2,b) = x(:,3,b);
end
figure(2); imshow(x); title('correzione bordo');

% equalizzazione
y = colorhisteq(x);   % funzione definita a pag.4
```

3. *Filtraggio in frequenza.*

```
clear all; close all; clc;
x = imread('foto_originale.jpg');
figure(1); imshow(x); title('immagine originale');
```

```
% filtraggio nello spazio HSI
w = rgb2hsi(x);
I = w(:,:,3);
[M,N] = size(I);
du = 1/M; dv = 1/N;
m = -1/2:du:1/2-du;
n = -1/2:dv:1/2-dv;
[k,l] = meshgrid(n,m);
X=fft2(I,M,N);
X = fftshift(X);
H = (abs(k)<=0.10 & abs(l)<=0.25);
Y = H.*X;
y = real(ifft2(fftshift(Y)));
w(:,:,3) = y;
z = hsi2rgb(w);
figure(2); imshow(z); title('immagine filtrata');
```

4. *Demosaicking*.

```
clc; close all; clear all;
mosaic = imread('Mosaic.bmp');
figmos = imread('Fiori_mosaic.bmp');
figor  = imread('Fiori256.bmp');

figure;
subplot(1,2,1); imshow(mosaic); title('maschera');
subplot(1,2,2); imshow(figmos); title('mosaico');
y = demosa(figmos);

figure;
subplot(1,2,1); imshow(figor); title('immagine originale');
xo = double(figor); varx = var(xo(:));

subplot(1,2,2); imshow(y/255);
y = double(y);
mse = mean((xo(:)-y(:)).^2);
snr = 10.*log10(varx./mse);
title(['immagine ricostruita, SNR=' num2str(snr)]);
```

```
function y = demosa(x)
   x = double(x);
   R = x(:,:,1);
   G = x(:,:,2);
   B = x(:,:,3);
   hg = [0 1/4 0; 1/4 1 1/4; 0 1/4 0];
   hr = [0   0   0  1/8  0   0   0;
         0   0   0  1/4  0   0   0;
         0   0   0  3/8  0   0   0;
        1/8 1/4 3/8  1  3/8 1/4 1/8;
         0   0   0  3/8  0   0   0;
         0   0   0  1/4  0   0   0;
         0   0   0  1/8  0   0   0];
   Ry = imfilter(R,hr,'circular','same');
   Gy = imfilter(G,hg,'circular','same');
   By = imfilter(B,hr,'circular','same');
   y = cat(3,Ry,Gy,By);
```