Elaborazione delle Immagini

Laboratorio 4

Obiettivi:

- Binarizzazione
- Segmentazione
- Labelling delle componenti connesse

Ricordate: per processare le immagini è sempre conveniente trasformale in valori double tra 0 e 1 con **im2double**.

Ricordate: imshow visualizza le immagini in modo corretto se hanno valori tra 0 e 255 (uchar8), se hanno valori tra 0 e 1 (double) o sono valori logici.

Ricordate: se volete saperne di più sulle funzioni Matlab usate, consultate l'help o la documentazione con i seguenti comandi da console:
help <funzione>
doc <funzione>

Scrivete il codice di ogni esercizio in uno script separato (labX_1.m, labX_2.m, ...)

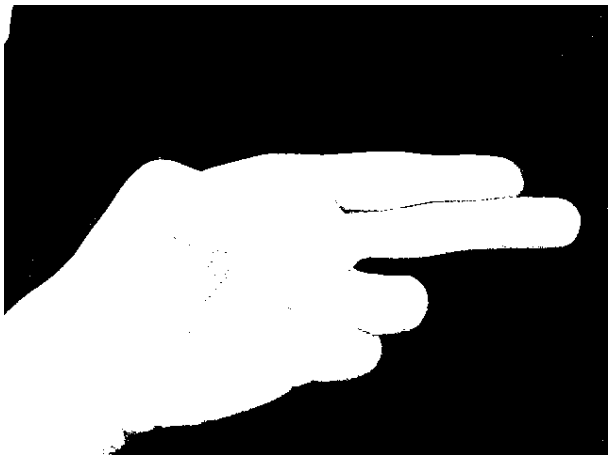
A1. Caricate l'immagine '**hand1.jpg**' in una variabile **hand**.

B1. L'immagine è a livelli di grigio. Visualizzate il suo istogramma con **imhist** e **plot.**

C1. Create una maschera **mask** (binarizzazione) della sola mano usando la funzione Matlab **im2bw.** La funzione vuole una soglia (in percentuale tra 0 e 1 rispetto al massimo valore rappresentabile nell'immagine; se l'immagine è a valori tra 0 e 1 la percentuale è già il valore di soglia). Determinate la soglia (**T1**) dall'istogramma. Come è fatto l'istogramma? Che soglia avete usato?

D1. Calcolate automaticamente la soglia (**T2**) usando la funzione Matlab **graythresh.** Come si discosta la soglia T2 da quella T1?

E1. Rifate la maschera (**mask2**) usando la soglia **T2** con **im2bw.** Confrontate i risultati.



La funzione graythresh usa l'algoritmo di Otsu per determinare la soglia di binarizzazione. L'algoritmo di Otsu trova la soglia t che massimizza la seguente funzione:

$$\sigma_b^2(t) = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2$$

Con

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i) \qquad \mu_0(t) = \sum_{i=0}^{t-1} i\frac{p(i)}{\omega_0}$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i) \qquad \mu_1(t) = \sum_{i=t}^{L-1} i\frac{p(i)}{\omega_1}$$

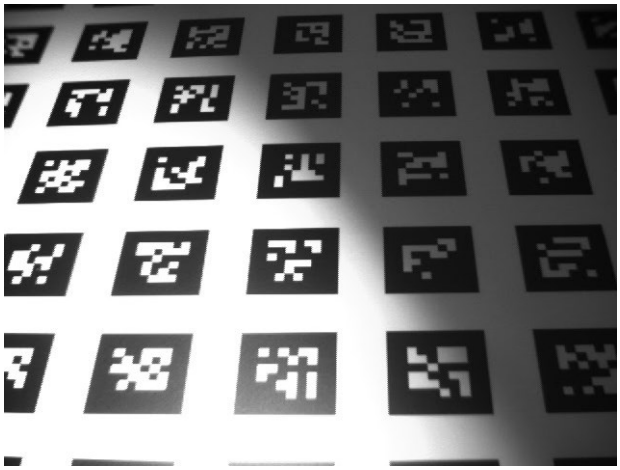
p(i) è la probabilità di avere un valore di intensità i, L è il valore massimo delle intensità.

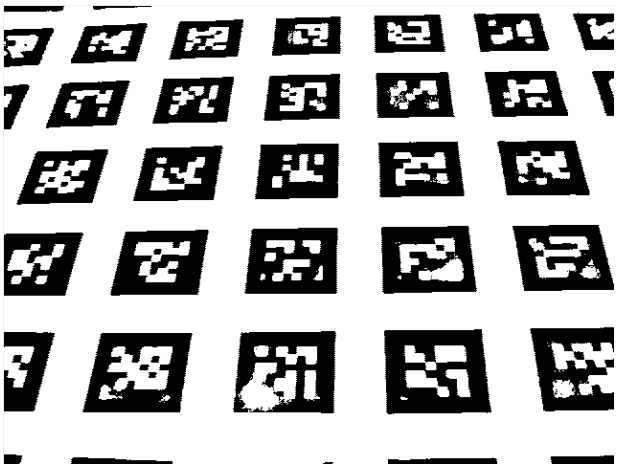
A2. Caricate l'immagine '**markers.png**' in una variabile **markers**.

B2. Provate a binarizzare l'immagine come nell'esercizio precedente creando una maschera **mask**. Cosa notate?

C2. Usate la funzione **sauvola** (presente tra i file di laboratorio) per binarizzare l'immagine creando una maschera **mask2**. La funzione implementa un algoritmo di binarizzazione automatico. Che cosa notate?

D2. La funzione sauvola, prende in input una immagine ed un intorno [M,N] di analisi. Di default usa un intorno [3,3]. Provate ad ingrandire l'intorno fino a quando ottenete un risultato accettabile. Che cosa avete notato?





L'algoritmo di binarizzazione di Sauvola, è un algoritmo locale. A differenza della semplice sogliatura, che è un algoritmo globale, Sauvola analizza delle regioni locali dell'immagine sulle quali determina la soglia che meglio separa le regioni chiare da quelle scure. E' un algoritmo che meglio si adatta in presenza di ombre e variazioni di luminosità nell'immagine. E' in grado di compensare differenze nell'immagine. La dimensione delle regioni di analisi deve essere settata in modo tale da avere estensione pari alla più grande regione che si vuole binarizzare.

A3. Caricate l'immagine '**hand2.jpg'** in una variabile **im.** L'obiettivo è quello di segmentare l'immagine e creare una maschera corrispondente alla sola mano.

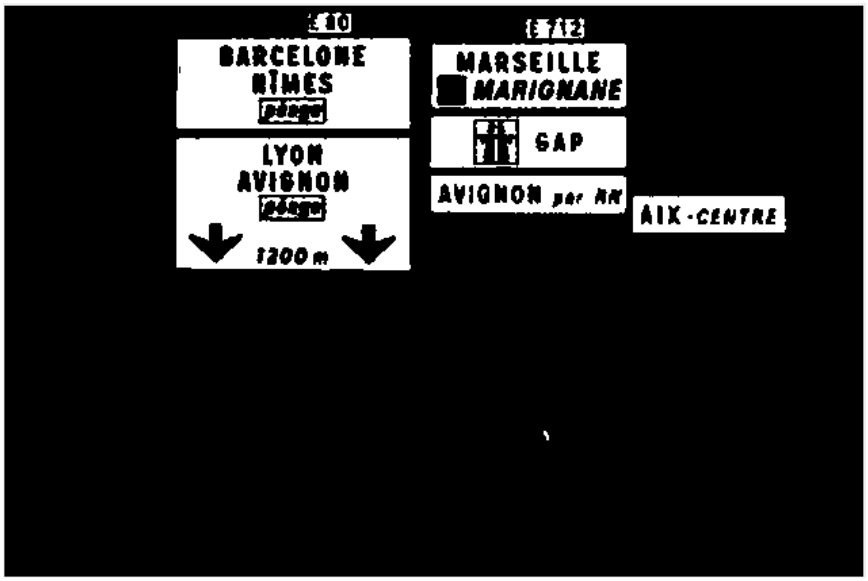
B3. Convertite l'immagine a livelli di grigio (**gray**) e provate a binarizzarla per trovare la regione della mano. Avete usato im2bw? Che soglia avete usato? Che cosa avete notato?

C3. Provate ad analizzare i canali RGB dell'immagine. E' possibile usare uno dei tre canali per la segmentazione? Se sì, quale?

D3. Provate a convertire l'immagine in un altro spazio colore. C'è qualche canale di qualche spazio colore che può essere usato ottenere una migliore segmentazione? Quale?

A4. Caricate l'immagine '**signs.jpg**' in una variabile **signs.**

B4. Usando ciò che avete osservato negli esercizi precedenti segmentate l'immagine in modo tale da avere una maschera (possibilmente) dei soli cartelli blu e verdi. Potrebbe essere necessario combinare insieme più maschere di segmentazione. Prima di provare a segmentare, osservate attentamente l'immagine e le caratteristiche delle regioni da segmentare... Che spazio/spazi colore, canale/canali colore avete usato? Perché?



A5. Caricate l'immagine '**euro_coins.png**' in **coins**.

B5. Visualizzate l'immagine caricata.

C5. Trasformate l'immagine a livelli di grigio (**gray**) e binarizzatela in modo da avere le regioni delle monete con pixel di valore 1. Mettete il risultato in una immagine **bw** e visualizzatela.

D5. Usate la funzione Matlab **bwlabel** sull'immagine bw per calcolare l'array bidimensionale con il labelling delle componenti connesse (**labels**). La funzione etichetta ciascuna regione connessa nell'immagine con un numero intero diverso.

E5. Visualizzate con **imagesc** l'immagine **labels.** Usate axis image e colorbar. Quante regioni sono state etichettate? Che etichetta hanno le monete?

F5. Trovate, dall'immagine, l'etichetta della moneta da 50 centesimi e create una maschera che identifichi i pixel di questa moneta (**mask50**).

G5. Combinate **mask50** e **coins** per ottenere una immagine a colori della sola moneta e visualizzate il risultato.

A6. Con l'immagine delle componenti connesse dell'esercizio precedente (**labels**), scrivete un algoritmo automatico per trovare la regione della moneta da 1 centesimo. In pratica dovete trovare in automatico l'etichetta della regione connessa più piccola presente nell'immagine. Usate i suggerimenti solo se necessario.

B6. Testate il vostro algoritmo sull'immagine '**coins.png**'.

SUGGERIMENT0 1 (selezionate il testo nel box e copiatelo in matlab):

SUGGERIMENT0 2: (selezionate il testo nel box e copiatelo in matlab):

SUGGERIMENT0 3: (selezionate il testo nel box e copiatelo in matlab):

SUGGERIMENT0 4: (selezionate il testo nel box e copiatelo in matlab):

Compiti a casa – Lab4

Scrivete gli script o le funzioni Matlab richieste e consegnatele in un file zip tramite l'apposito link sul sito del corso. NON potete usare le rispettive funzioni Matlab!

Scrivete una funzione **mygreythresh** che determina la soglia di binarizzazione di una immagine a livelli di grigio con l'algoritmo di Otsu (basatevi sulle istruzioni dell'esercizio 1):

```
function out_t = mygreythresh(image)
…
end
```

Scrivete una funzione **mylabeling** che, data una immagine binaria, etichetta tutte le regioni usando la 4 connessione. Le etichette sono valori interi:

```
function out_labels = mylabeling4c(bw_image)
…
end
```