Elaborazione e Analisi delle Immagini

Giuseppe Bellisano

Indice

Capitolo 1

Immagini

1.1 Image processing, analysis e understanding. Spiegarne la differenza

Abbiamo tre categorie principali di manipolazione delle immagini:

- **image processing** (input: immagine -> output: immagine);
- **image analysis** (input: immagine -> output: metriche);
- **image understanding** (input: immagine -> output: descrizione ad alto livello)

Nell'**image processing** l'input della manipolazione è un'immagine e l'output è ancora un'immagine e più precisamente si tratta dell'immagine originale opportunamente processata (grazie a tecniche di miglioramento e rimozione del rumore)

Nell'**image analysis** è la trasformazione che fa passare dall'immagine a una sua rappresentazione astratta, cioè ad una sua descrizione ad alto livello

Nell'**image understanding** data un'immagine otteniamo invece descrizioni di alto livello sul contenuto dell'immagine (ad esempio il riconoscimento degli oggetti all'interno delle immagini e le relazioni tra di essi).

1.2 Definire una immagine digitale

Un'immagine digitale è una rappresentazione numerica/digitale di un'immagine attraverso una matrice rettangolare, in cui ad ogni posizione (che prende il nome di pixel) sono assegnati uno o più valori numerici e ad ogni posizione corrisponde una coordinata reale sul piano immagine.

1.2.1 Dettagli

Perché le immagini siano elaborate al computer occorre trasformarle in una rappresentazione numerica/digitale attraverso un processo chiamato digitalizzazione.

Le immagini digitali sono matrici rettangolari in cui ad ogni posizione corrispondono uno o più valori numerici: ad ogni posizione corrisponde una coordinata reale sul piano immagine; ogni punto prende il nome di pixel (picture per element).

Un immagine digitale f[m,n] descritta in uno spazio discreto 2D è derivata da un immagine analogica f(x,y) in uno spazio continuo 2D attraverso il processo di campionamento (sampling) detto digitalizzazione. L'immagine continua in 2D f(x,y) viene divisa in N righe ed M colonne.

L'intersezione di una riga con una colonna determina un pixel. Il valore assegnato alle coordinate intere [m,n] con m = O, 1, 2,..., M-1 e n = O, 1, 2,..., N-1 è f[m,n]. e rappresenta l'intensità di grigio associata al punto identificato. Il numero dei valori di grigi generalmente è una potenza di 2: L=2 B (se B>1 si parla di immagine in scala di grigi, se B=1 si parla di immagini binarie).

Quando un'immagine viene digitalizzata il sampling avviene sia a livello spaziale (con le righe e le colonne) sia a livello di ampiezza (con la quantizzazione dei livelli di grigio).

1.3 Cosa si intende per risoluzione di una immagine?

La risoluzione è un indice della qualità di un'immagine. E' la quantità dei suoi punti elementari (pixel) rapportata ad un'unità di lunghezza, dove quest'ultima di solito è il pollice (dpi, dot per inch).

1.3.1 Dettagli

Possiamo distinguere due tipi di risoluzione: spaziale e dei livelli d'intensità.

- **La risoluzione spaziale** indica il più piccolo dettaglio distinguibile nell'immagine. È quindi una misura della densità dei pixel di un'immagine (che si misura in punti per unità di lunghezza, che solitamente è il pollice).

  Poiché un'immagine digitale è discreta ed è rappresentata come un insieme di punti, se il numero di punti per unità di lunghezza è molto elevato, allora ogni pixel rappresenta dettagli più piccoli e quindi aumentando il numero di pixel aumento il dettaglio percepito dall'occhio umano.

  La risoluzione spaziale è quindi strettamente dipendente dalla frequenza di campionamento dell'immagine digitale.

- **La risoluzione dei livelli d'intensità**, in modo simile, consiste nel più piccolo cambiamento del livello d'intensità distinguibile nell'immagine. Anche in questo caso se l'immagine presenta un elevato numero di livelli d'intensità differenti, allora la risoluzione sarà maggiore, in quanto non verrà percepito il passaggio da un livello all'altro. La risoluzione dei livelli d'intensità è quindi legata alla frequenza di campionamento in ampiezza (quantizzazione dei livelli d'intensità) dell'immagine.

  La quantizzazione (campionamento in ampiezza) può essere uniforme, ovvero le intensità degli oggetti sono mappate in modo diretto sui livelli di grigio dell'immagine, oppure può essere logaritmica (maggior risoluzione

nelle zone scure). L'occhio umano ha una percezione logaritmica dei livelli di grigio.

La scelta del campionamento viene fatta in base all'uso dell'immagine, alle limitazioni della memoria e della velocità dell'elaborazione, alla necessità o meno di analizzare l'immagine e alle informazioni che bisogna estrarne.

1.4 Illustrare il processo di campionamento e quantizzazione

La digitalizzazione di un'immagine comprende il campionamento e la quantizzazione. Sono entrambi processi di discretizzazione. Il campionamento consiste nella discretizzazione del dominio spaziale e in genere è uniforme, mentre la quantizzazione consiste nella discretizzazione dei livelli di grigio e può essere sia uniforme che logaritmica.

1.4.1 Dettagli

Quando un immagine viene digitalizzata, il campionamento (sampling) avviene sia a livello spaziale (con le righe e le colonne) sia a livello di ampiezza (con la quantizzazione dei livelli di grigio).

Per campionare un'immagine è necessario scegliere innanzitutto quanti pixel vogliamo rappresentare (campionamento spaziale), scegliendo quindi il numero di righe e di colonne della matrice che rappresenterà l'immagine. Da questa scelta dipenderà ovviamente la risoluzione dell'immagine digitalizzata.

- Il sampling spaziale può essere uniforme, ovvero il campionamento ha la stessa frequenza su tutta l'immagine, oppure adattivo ovvero si utilizza maggiore frequenza nelle aree con più dettaglio. Il valore discreto frutto del sampling si chiama pixel (picture element) nello spazio 2D e voxel (volume element) nello spazio 3D.

- La risoluzione indica il più piccolo dettaglio ottenibile dall'immagine. La frequenza di campionamento limita la risoluzione. La quantizzazione dei livelli di luminosità dei pixel (ossia il campionamento in ampiezza) consiste invece nel decidere il numero di valori di intensità rappresentabili per ogni pixel (che solitamente è una potenza di 2).

- La quantizzazione (campionamento in ampiezza) può essere uniforme, ovvero le intensità degli oggetti sono mappate in modo diretto sui livelli di grigio dell'immagine, oppure può essere logaritmica (maggior risoluzione nelle zone scure). L'occhio umano ha una percezione logaritmica dei livelli di grigio. La scelta del campionamento viene fatta in base all'uso dell'immagine, alle limitazioni della memoria e della velocità dell'elaborazione, alla necessità o meno di analizzare l'immagine e alle informazioni che bisogna estrarne.

- Le immagini a colori contengono maggiori informazioni di quelle monocromatiche. L'uomo rileva i colori come una combinazione dei tre colori primari: rosso, verde e blu. Il computer usa quindi il modello RGB in cui un pixel può essere associato ad un vettore tridimensionale che fornisce

le rispettive intensità dei tre colori; abbiamo quindi ad esempio: nero=(O,O,O); bianco=(k-1,k-1,k-1); rosso puro=(k-1,O,O) dove k è il range di intensità possibile (quindi i colori rappresentabili col modello RGB sono k^3).

Un altro modo per rappresentare le immagini a colori è il modello HSL (hue = tonalità, saturation = saturazione, lightness = luminosità) simile al modello RGB nel funzionamento.

1.5 Descrivere i passi fondamentali di un processo di elaborazione di immagini digitali

I passi fondamentali di un processo di elaborazione di immagini digitali sono quattro:

- **Acquisizione dell'immagine**: consiste nella digitalizzazione dell'immagine da elaborare, ossia il processo di campionamento con cui si cerca di derivare da un'immagine reale, definita nello spazio continuo, un'immagine discreta, rappresentabile come matrice. In questa fase avviene quindi la scelta del campionamento spaziale e in ampiezza da cui dipende la risoluzione dell'immagine.
- **Preprocessing**: comprende tutte quelle operazioni di basso livello che si eseguono sulle immagini. Lo scopo è quello di enfatizzare una certa caratteristica allo scopo di rendere più adatta l'immagine a computazioni future di livello più alto. Esempi tipici sono il miglioramento del contrasto, lo smoothing, correzione di distorsioni, eliminazione del rumore.
- **Segmentazione**: consiste nella suddivisione di un'immagine in modo da identificare ed isolare gli oggetti che compaiono nell'immagine stessa. È uno degli aspetti più complicati del processo di elaborazione delle immagini digitali.
- **Descrizione e classificazione degli oggetti ottenuti dalla segmentazione**: con metodi ad alto livello si possono ricavare informazioni e modelli (image understanding, visione) e a trovare le relazioni tra di essi (ad esempio la posizione di un oggetto rispetto ad un altro).

1.6 Dettagli

L'elaborazione di immagini digitali è una disciplina che definisce tecniche e algoritmi informatici per modificare immagini digitali allo scopo di: migliorarne la qualità (riduzione del rumore o aumento del contrasto); generare codifica utile, compressione; generare visualizzazioni (grafica computazionale); estrarre semplici features; ricavare informazioni/modelli (image understanding, visione).

La Computer Vision mira a riprodurre l'effetto della vista umana utilizzando la percezione elettronica di un'immagine e la sua successiva comprensione. Possiamo distinguere due livelli: High Level Image Understanding e Low Level Image Processing.

- il processing di basso livello utilizza poche informazioni riguardo il contenuto dell'immagine
- il processing di alto livello fa l'esatto opposto: utilizza tali informazioni per cercare di simulare le capacità cognitive umane e la nostra abilità nel prendere decisioni in base a ciò che l'immagine rappresenta. Per fare questo spesso nel processing di alto livello si ricorre a tecniche tipiche dell'intelligenza artificiale.

La vista umana non dà un'immagine fedele del mondo esterno ma ne dà un'interpretazione spesso fallace. Questo perché il cervello umano tende ad associare le informazioni in insiemi correlati sulla base di: raggruppamento, somiglianza, continuità, chiusura, figura/sfondo, profondità, costanza forme in prospettiva e moto. La visione umana è un processo di interpretazione molto complesso, ma in qualche modo computazionale. Si basa su informazioni a priori, innate ed acquisite. Difficilmente imitabile da un sistema artificiale. Sono imitabili però alcune idee e procedure. Il computer invece misura valori assoluti, esegue calcoli ed è: instancabile, economico, veloce e oggettivo.

Capitolo 2

Concetti base sulle immagini digitali

2.1 Come si classificano le operazioni spaziali da applicare su un'immagine digitale?

I tipi di operazioni che possono essere applicati ad immagini digitali per trasformare un'immagine input $a[M, N]$ in un'immagine output $b[M, N]$ (o altre rappresentazioni) possono essere classificate in tre categorie:

- **Puntuali**: il valore di output ad una specifica coordinata dipende solo dal valore in input alla medesima coordinata. La complessità associata a questo tipo di operazioni è costante
- **Locali**: il valore di output ad una specifica coordinata dipende non solo dall'input del valore alla medesima coordinata ma anche da un suo intorno. La complessità è quadratica sulle dimensioni dell'intorno scelto
- **Globali**: il valore di output ad una specifica coordinata dipende da tutti i valori dell'immagine input. La complessità è quadratica sulle dimensioni dell'immagine stessa

2.2 Definire il concetto di adiacenza nelle immagini digitali

L'adiacenza è quella relazione fra due pixel di un'immagine digitale per cui la loro distanza è uguale a 1.

- **City Block** La distanza City Block da un dato pixel è dato da tutti quei pixel che hanno distanza 1 (dal suddetto pixel) compiendo un solo passo in verticale o in orizzontale (4-adiacenza)
- **Scacchiera (Chessboard)** La distanza Scacchiera (Chessboard da un dato pixel è dato da tutti quei pixel che hanno distanza 1 (dal suddetto pixel) compiendo un solo passo in verticale, in orizzontale o anche in diagonale (8-adiacenza)

2.3 Definire una regione di un'immagine binaria

Una regione è un insieme connesso di pixel in cui esiste un percorso fra ogni coppia di questi pixel. Questo percorso è tracciato saltando sempre da un pixel al pixel adiacente, e ciascuno di questi pixel appartiene alla regione. Una regione può avere buchi al suo interno.

2.4 Definire un contorno in un'immagine binaria

- il bordo interno è l'insieme dei pixel di una regione che hanno almeno un pixel adiacente non appartenente alla regione
- il bordo esterno è invece costituito dai pixel esterni alla regione che hanno almeno un pixel adiacente appartenente alla regione

2.5 Definire un edge in un'immagine digitale

Formalmente l'edge è una proprietà di un pixel e del suo intorno. In un'immagine digitale, è detto "di edge" un pixel nel cui intorno sono presenti dei pixel con livelli di grigio che variano bruscamente.

2.6 Illustrare il concetto di distanza tra punti in una immagine digitale

La distanza tra punti di coordinate (i, j) e (h, k) può essere definita come distanza:

- **Euclidea** $D_e[(i,j),(h,k)] = \sqrt{(i-h)^2 + (j-k)^2}$
- **City Block** $D_4[(i,j),(h,k)] = |i-h| + |j-k|$ La distanza City Block da un dato pixel è dato da tutti quei pixel che hanno distanza 1 (dal suddetto pixel) compiendo un solo passo in verticale o in orizzontale
- **Scacchiera (Chessboard)** $D_8[(i,j),(h,k)] = max\{|i-h|, |j-k|\}$ La distanza Scacchiera (Chessboard da un dato pixel è dato da tutti quei pixel che hanno distanza 1 (dal suddetto pixel) compiendo un solo passo in verticale, in orizzontale o anche in diagonale

2.7 Fornire la definizione di istogramma di un'immagine

L'istogramma di un'immagine digitale $f[m,n]$ è una funzione che a ciascun livello di grigio definito nella quantizzazione dell'immagine, fa corrispondere un valore pari al numero di pixel dell'immagine a cui è assegnato quel livello di grigio.

$$H(k) = card\{f(x,y) | f(x,y) = k\}$$

con $k = 0, 1, \ldots, 2^B - 1$

per cui vale:

$$\sum_{k=0}^{2^B-1} H(k) = M \cdot N$$

dove $card = \#$

2.7.1 Dettagli

Le immagini a colori hanno tre istogrammi: uno per ogni componente.

Ecco un esempio di istogramma per un'immagine monocromatica 4x4 a 3 bit:

6	7	3	3
2	6	1	0
4	6	1	0
6	6	6	0

L'istogramma è quindi: $H(k), k = 0, \ldots, 7$ [4 2 1 1 1 0 6 1]

Possiamo definire l'algoritmo di creazione dell'istogramma:

- Assegna zero a tutti gli elementi dell'array H;
- Per ogni pixel (x,y) dell'immagine f, incrementa H(f(x,y)) di 1.

L'istogramma può avere molti punti di minimo e massimo locale, il problema può essere risolto attraverso un operazione di smoothing. L'istogramma H' a cui è stato applicato uno smoothing in un k intorno quadrato di lato k è definito come:

$$H'(z) = \frac{1}{2k+1} \sum_{i=-k}^{k} H(z+i)$$

2.8 Illustrare le strutture dati tradizionali per rappresentare un'immagine

Ci sono due tipi di strutture dati tradizionali: le matrici e le catene.

- **Le matrici (array 2D)** sono il tipo di struttura dati più usato per la rappresentazione a basso livello di un'immagine
  - le immagini binarie sono rappresentate come matrici binarie
  - le immagini monocramatiche sono rappresentate come immagini d'intensità
  - le immagini a colori sono rappresentate come tre immagini (RGB o HSI)

  Gli elementi delle matrici sono numeri interi. Le caratteristiche spaziali sono ottenibili in modo implicito. I dati delle immagini di questo tipo sono ottenuti generalmente come output diretto del dispositivo di cattura dell'immagine come ad esempio lo scanner.

- Le **chain** vengono utilizzate per le immagini binarie. Per fare questo si sceglie un punto del bordo a partire dal quale si indicano i passi discreti da eseguire in serie. Ad ogni direzione è associata un etichetta. Vediamo due tipi di catene:

  – **Codifica di Freeman**: si utlizza per rappresentare i bordi all'interno delle immagini; si parte generalmente dal punto più in alto a sinistra del contorno e lo si percorre utilizzando spostamenti secondo 8 direzioni prestabilite (a cui sono assegnati i valori da O a 7). In ogni punto vengono provate le 8 direzioni per trovare i pixel adiacenti tra loro che definiscono il contorno. Alla fine il codice di contorno sarà costituito dalla lista delle direzioni usate in successione per individuare i punti di contorno.

  – **Run Length Coding**: sono spesso utilizzate per rappresentare le aree all'interno degli oggetti, e le stringhe di simboli su matrici (i fax funzionano così ad esempio). La rappresentazione consiste in una serie di liste in cui ogni riga rappresenta una sottolista. Il primo elemento di ogni sottoriga indica l'indice della riga stessa, i restanti elementi vanno esaminati in coppie e rappresentano la colonna iniziale e la colonna finale di una sequenza. In questo modo vengono identificati pixel adiacenti appartenenti allo stesso oggetto riga per riga.

2.9 Illustrare le strutture dati topologiche per rappresentare un'immagine

Le strutture dati di tipo topologico descrivono un'immagine come un insieme di elementi e le relazioni tra di essi. Relazioni come ad esempio "la regione 5 è adiacente alla regione 12" oppure "l'immagine dell'uomo è vicino all'immagine dell'albero" sono spesso rappresentate come grafi. Ecco un esempio:

Inoltre abbiamo anche dei database relazionali in cui tutte le informazioni sono concentrate in relazioni tra parti semanticamente importanti dell'immagine, ossia oggetti risultanti dalla segmentazione dell'immagine. Queste sono utili per la comprensione ad alto livello delle immagini. Ecco un esempio:

2.10 Illustrare le strutture dati topologiche per rappresentare un'immagine

Le strutture dati topologiche permettono di descrivere le relazioni tra gli oggetti all'interno di un'immagine, solitamente una volta che l'immagine è stata segmentata. I grafi di adiacenza hanno dei nodi corrispondenti agli oggetti dell'immagine, collegati da un lato se esiste una relazione di adiacenza o inclusione tra di essi. Con i database relazionali si possono assegnare delle chiavi ai vari oggetti definendo per ciascuno di essi una serie di attributi e indicando con opportune relazioni le adiacenze e le inclusioni. In questo modo l'immagine può essere trattata come un database e si possono eseguire delle query.

2.11 Illustrare le strutture dati gerarchiche per rappresentare un'immagine

Le strutture dati gerarchiche per la memorizzazione di immagini sono concepite allo scopo di rendere la computazione più leggera. L'immagine viene resa a diversi livelli di risoluzione, lavorando alla risoluzione massima solo nelle parti dell'immagine in cui ciò è necessario. Strutture dati di questo tipo sono le M-Piramidi e le T-piramidi.

- le M-piramidi (piramidi di matrici) consistono in una sequenza di matrici ordinate per risoluzione decrescente La prima matrice è l'immagine originale alla risoluzione massima. Le matrici successive rappresentano l'immagine via via dimezzata (occupando quindi 1/4 dello spazio)e consentono quindi di lavorare con un'immagine scegliendo la risoluzione più opportuna. Vengono quindi generate le matrici a tutti i livelli di risoluzione fino ad arrivare a una matrice di dimensione 1x1. Per determinare i valori dei pixel della matrice $M(i)$ si utilizza solitamente la media o il valore massimo dei 4 pixel corrispondenti nella matrice $M(i-1)$
- le T-piramidi è un modo alternativo rispetto alle M-Pyramid e si basa sull'utilizzo degli alberi. Ogni nodo ha sempre 4 figli. Le foglie corrispondono al singolo pixel, proprio come i QuadTree che sono una loro specializzazione
- il Quad tree è sempre un albero ma rispetto alla T-pyramid ogni nodo può avere o 4 figli, quando una regione è non omogenea, oppure nessun figlio se una regione è omogenea. E' una struttura dati vantaggiosa nel caso in cui l'immagine abbia grosse regioni omogenee. Il vantaggio è quindi il risparmio di memoria ma piccole variazioni da un'immagine all'altra possono comportare grandi differenze tra i rispettivi quad tree rendendo quindi difficile valutare la similarità fra immagini

Problemi legati alle strutture dati gerarchiche sono:

- Dipendenza dalla posizione, orientamento e dimensione degli oggetti
- Due immagini simili possono avere rappresentazione molto diverse

2.11.1 Dettagli

La computer vision è molto costosa dal punto di vista computazionale, anche solo a causa della gran quantità di dati da processare. Una soluzione può essere quella di utilizzare il calcolo parallelo (metodo della forza bruta), tuttavia molti problemi legati alla computer vision sono difficili da suddividere tra i processori. Le strutture dati gerarchiche permettono invece di realizzare algoritmi che devono avere a che fare con una quantità di dati inferiore.

2.12 Altro: Quali sono le proprietà topologiche delle immagini?

Sono invarianti rispetto alle rubber sheet (foglio) transformations; ad esempio allungando l'immagine non cambia la continuità delle sue parti e non cambia

il numero dei buchi al suo interno. La caratteristica di Eulero-Poincare è definita come la differenza tra il numero di regioni ed il numero di buchi in esse. L'**insieme convesso** è una regione di punti dei quali prendendone due qualsiasi il segmento che li unisce è del tutto compreso nell'insieme. Il **guscio convesso** è invece la più piccola regione convessa che racchiude un insieme di punti.

2.13 Altro: Strutture dati di supporto (matrici - grafi di adiacenza - chains - grafi - DB relazionali)

L'organizzazione dei dati può condizionare sensibilmente la scelta e la semplicità dell'algoritmo. Le immagini possono essere a vari livelli di rappresentazione:

- Iconic images: consistono in immagini che contengono i dati originali (matrice di interi con informazioni sulla luminosità del pixel)
- Immagini segmentate: parti delle immagini riunite i gruppi che probabilmente appartengono allo stesso oggetto;
- Rappresentazioni geometriche: forme geometriche 2D e 3D
- Modelli relazionali: per trattare i dati in modo più efficiente e ad un livello di astrazione più alto (es. posizione degli oggetti, relazioni tra di essi, ricerca di oggetti)

Capitolo 3

Pre-processing, filtri spaziali, trasformazioni

3.1 Cosa si intende per preprocessing di una immagine?

Il preprocessing di un'immagine comprende le operazioni svolte al livello più basso nel processo di elaborazione di un'immagine. L'immagine in input viene manipolata al fine di rimuovere o ridurre l'eventuale presenza di distorsione o rumore oppure amplificare e rendere più evidenti determinate caratteristiche utili per le successive fasi di elaborazione di livello più alto. In output viene restituita l'immagine modificata. Esistono quattro categorie di metodi:

- **trasformazioni della luminosità**: In questo ambito si parla di:
  - **correzione di luminosità** quando per modificare il valore di un pixel si tiene conto del valore precedentemente associato al pixel stesso e della sua posizione nell'immagine
  - **trasformazione della scala di grigio** quando si modificano i toni senza tener conto della posizione dei pixel nell'immagine
- **trasformazioni geometriche** che permettono l'eliminazione di distorsioni geometriche che si presentano alla cattura dell'immagine. Esempi di trasformazione geometrica sono la rototraslazione e il cambiamento di scala
- **metodi di preprocessing** che usano un intorno del pixel processato
- **ristrutturazione di immagini** (richiede info sull'intera immagine)

3.2 Descrivere i metodi di preprocessing di una immagine

- **trasformazioni della luminosità**

  - **Correzione di luminosità (dipendente dalla posizione)**: Spesso l'immagine può essere degradata a causa dell'illuminazione dell'oggetto ripreso o della sensibilità del dispositivo di cattura dell'immagine. Sia e(i, j) la funzione che perturba la funzione ideale $g(i, j)$ e sia $f(i, j) = e(i, j) \cdot g(i, j)$ l'immagine degradata. Se l'immagine ideale g(i, j) è nota a priori (ad esempio unafunzione costante che assume il valore e in tutto il suo dominio) indichiamo con $f_c(i, j)$ l'immagine affetta da degrado e possiamo concludere che:

$$g(i, j) = f(i, j) = c \cdot f)i, j) = c \cdot (i, j)$$

  - **Trasformazione della scala di grigio**: È una tecnica che non tiene conto della posizione dei pixel e viene spesso usata quando il risultato deve essere direttamente mostrato all'occhio umano. Si può ottenere mediante contrast stretching o equalizzazione dell'istogramma

- **Trasformazioni geometriche**: Permettono l'eliminazione di distorsioni geometriche che si presentano alla cattura dell'immagine. Esempi di trasformazione geometrica sono la rotazione (che permette di ruotare un'immagine in modo da poterla confrontare con un'altra immagine), la traslazione, la distorsione e il cambiamento di scala.

  Una trasformazione è una funzione vettore T che mappa un pixel in posizione (x,y) in una nuova posizione (x',y'):

$$x' = Tx(x, y) \qquad y' = Ty(x, y)$$

  Una trasformazione geometrica consiste quindi di due operazioni principali:

  - trasformazione spaziale delle coordinate dei pixel, in modo da determinare la loro nuova posizione (cioè date (x,y) determinare (x',y')
  - determinare l'intensità del pixel nelle nuove coordinate (generalmente utilizzando l'interpolazione dei valori di intensità dei pixel dell'intorno del pixel trasformato e si possono distinguere diversi metodi quali il 'nearest neighbor interpolation', in cui si prende il valore del pixel più vicino, la 'bilinear interpolation' in cui si analizzano i 4 pixel più vicini, e la 'bicubic interpolation' in cui si analizzano i 16 pixel più vicini)

- **Metodi di preprocessing che usano un intorno del pixel processato**: I metodi di pre-processing che usano un piccolo intorno di pixel nell'immagine in input per determinare il nuovo valore di luminosità nell'output sono definiti metodi di filtrazione. Essi si possono dividere in due macro categorie:

  - **operatori di smoothing**: si occupano di ridurre il rumore o piccole fluttuazioni in un immagine
  - **operatori gradiente**: si occupano di rilevare i punti di edge dell'immagine

  L'operazione tipica di questa categoria di metodi è la convoluzione, un operazione lineare che produce un valore di luminosità per l'immagine in

output in base alla combinazione lineare dei pixel in input che appartengono all'intorno del punto in esame. Ad ogni pixel dell'intorno è associato un peso che ne determina il contributo.

- **ristrutturazione di immagini** (richiede info sull'intera immagine): Il restauro infine serve per ricostruire il contenuto informativo e visivo delle immagini corrotte da un punto di vista oggettivo, a differenza delle operazioni che ricadono nell'image enhancement (miglioramento dell'immagine) che invece sono soggettive e dipendono da ciò che un singolo individuo ritiene essere un "buon" risultato.

  Prendendo in ingresso un'immagine degradata, il restauro ha come obbiettivo quello di ottenere l'immagine originale senza degrado, avendo a disposizione delle informazioni riguardo al degrado stesso e applicando quindi un "degrado inverso".

3.3 Cosa si intende per filtro spaziale?

Con filtraggio spaziale si intende un'operazione che coinvolge un intorno di pixel dell'immagine ed una sottoimmagine (chiamata generalmente maschera o filtro) della stessa dimensione dell'intorno. Si effettuano operazioni di filtraggio per evidenziare certe caratteristiche o rimuoverne altre.

I metodi di filtrazione possono essere divisi in due macro categorie:

- **operatori di smoothing**: si occupano di ridurre il rumore o piccole fluttuazioni in un immagine. Sfortunatamente questo comporta una perdita di dettaglio nei punti di edge, che contengono molte delle informazioni associate all'immagine stessa. (Corrispondono alla soppressione delle alte frequenze nel dominio delle frequenze).

- **operatori gradiente**: si basano sulle derivate parziali locali della funzione immagine. Le derivate sono maggiori nei punti in cui la funzione subisce un rapido cambiamento (sono i punti di edge). L'operatore gradiente ha il compito di indicare una tali punti aumentandone il dettaglio. Sfortunatamente applicando un operatore gradiente su un immagine il livello di rumore cresce notevolmente.

In pratica il processo di filtraggio nel dominio spaziale consiste nello spostare la maschera di filtraggio (che ha dimensione dispari in modo da poter far coincidere il centro con ogni punto dell'immagine) da un punto all'altro dell'immagine. Per ogni punto di coordinate (x,y) nell'immagine originale, il risultato del filtraggio in quel punto è calcolato valutando i valori dell'intorno di quel punto con i valori contenuti nella maschera, ponendo infine il risultato nel punto di coordinate (x,y) dell'immagine finale.

Con filtro spaziale lineare in particolare si intende che il risultato è dato dalla somma dei prodotti dei valori della maschera per i valori dell'intorno del punto con cui coincide il centro della maschera nell'immagine originale.

Per una maschera w di dimensioni 3X3 applicata al punto (x,y) di un'immagine, il risultato R sarebbe:

$$R = w(-1,-1)*f(x-1,y-1)+w(-1,O)*f(x-1,y)+w(-1,1)*f(x-1,y+1)+w(O,-1)*f(x,y-1)+w(O,O)*f$$

Con filtro spaziale non lineare si intende che il risultato è basato sull'ordinamento dei pixel contenuti nell'intorno di un'immagine e sulla sostituzione del pixel centrale dell'intorno con il valore determinato dal risultato dell'ordinamento (ranking filters)

3.4 Cosa si intende per trasformata di una immagine?

Per trasformata di un'immagine si intende un generico operatore che fa corrispondere ad un'immagine descritta da una funzione $f(x, y)$ un'altra immagine descritta da una funzione $g(u, v)$. Alcuni task dell'image processing sono meglio formulati trasformando l'immagine di input, ovvero portandola su di un nuovo dominio di trasformazione, e quindi risolvendo il generico problema da affrontare. Si può riportare l'immagine in output al dominio spaziale originale utilizzando la trasformazione inversa.

Una particolare classe di trasformate lineare 2D T(u,v) può essere espresso nella forma generale:

$$T(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)\, r(x, y, u, v)$$

dove f(x,y) è l'immagine originale; r(x,y,u,v) è detta forward transformation kernel e u,v sono le variabili di trasformazione.

La trasformata inversa invece è:

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u, v)\, s(x, y, u, v)$$

dove s(x,y,u,v) è detta inverse transformation kernel

3.5 Cosa si intende per equalizzazione di un istogramma? A cosa serve?

È una tecnica utilizzata per spalmare i livelli di intensità di un immagine sull'intero range disponibile, ottenendo così un nuovo istogramma il più possibile costante. In pratica si fa in modo che un qualsiasi tono di grigio possa presentarsi in un qualsiasi pixel con la stessa probabilità.

Se H(x) è l'istogramma dell'immagine originale è possibile cambiare l'istogramma attraverso l'uso di una funzione $y = y(x)$ cosi che l'istogramma $G(y)$ dell'immagine trasformata è costante per tutti i valori di luminosità, $G(y) = C$. La relazione tra H e G e la funzione y è data da:

$$H(x)dx = G(y)dy = Cdy$$

L'algoritmo per l'equalizzazione è quindi:

1. valutare l'istogramma H(x)

2. computare:

$$y(x) = \frac{256}{\sum_{k=0}^{255} H(k)^{k=0}} \sum_{k=0}^{x} H(k)$$

3. applicare la trasformazione y(x)

3.6 Come si può realizzare uno stretching del contrasto presente in una immagine?

Se l'immagine non utilizza il 1OO% del range di luminosità a disposizione è possibile migliorare l'immagine spalmando i valori di luminosità dei singoli pixel lungo tutta la gamma dinamica. Se i valori di luminosità vanno da 0 a $2^B - 1$ allora il valore minimo (min) dovrà essere mappato sullo 0 a ed il valore massimo (max) sul $2^B - 1$ con i valori intermedi che saranno mappati di conseguenza secondo la formula:

$$b[m, n] = 2^B - 1 \cdot \frac{a[m, n - min]}{max - min}$$

Il risultato ottenuto sarà quindi un miglioramento del contrasto.

3.7 Definire un filtro spaziale lineare

Con filtro spaziale si intende un'operazione che coinvolge un intorno di pixel dell'immagine ed una sottoimmagine (chiamata generalmente maschera o filtro) della stessa dimensione dell'intorno. In pratica il processo di filtraggio nel dominio spaziale consiste nello spostare la maschera di filtraggio (che ha dimensione dispari in modo da poter far coincidere il centro con ogni punto dell'immagine) da un punto all'altro dell'immagine. Per ogni punto di coordinate (x,y) nell'immagine originale, il risultato del filtraggio in quel punto è calcolato valutando i valori dell'intorno di quel punto con i valori contenuti nella maschera, ponendo infine il risultato nel punto di coordinate (x,y) dell'immagine finale.

Un filtro spaziale lineare è un filtro in cui il valore di un pixel di output è combinazione lineare dei valori dei pixel dell'intorno del pixel in input. A ogni filtro spaziale lineare corrisponde una matrice di convoluzione.

Il risultato è dato dalla somma dei prodotti dei valori della maschera per i valori dell'intorno del punto con cui coincide il centro della maschera nell'immagine originale. Per una maschera w di dimensioni 3X3 applicata al punto (x,y) di un'immagine, il risultato R sarebbe:

$$R = w(-1,-1)*f(x-1,y-1)+w(-1,O)*f(x-1,y)+w(-1,1)*f(x-1,y+1)+w(O,-1)*f(x,y-1)+w(O,O)*f$$

Con filtro spaziale non lineare si intende che il risultato è basato sull'ordinamento dei pixel contenuti nell'intorno di un'immagine e sulla sostituzione del pixel centrale dell'intorno con il valore determinato dal risultato dell'ordinamento (ranking filters)

3.8 Cosa si intende per operatore lineare?

Per operatore lineare si intende un operatore che effettua una combinazione lineare (una media pesata) dei livelli di intensità dei pixel di un intorno per ottenere in output un nuovo livello di intensità. Sia H un operatore che produce in output un'immagine g(x,y) dall'immagine input f(x,y). H è definito operatore lineare se:

$$H[a_i f_i(x,y) + a_j f_j(x,y)] = a_i H[f_i(x,y)] + a_j H[f_j(x,y)] = a_i g_i(x,y) + a_j g_j(x,y)$$

Dove $f_i(x,y)$ ed $f_j(x,y)$ sono immagini ed a_i e a_j sono costanti della stessa dimensione delle immagini.

3.9 Correlazione e convoluzione spaziale. Spiegarne le differenze

La convoluzione è un operazione lineare che produce un valore di luminosità per l'immagine in output in base alla combinazione lineare dei pixel in input che appartengono all'intorno del punto in esame. Ad ogni pixel dell'intorno è associato un peso che ne determina il contributo. Le maschere di convoluzione di solito hanno righe e colonne dispari così da identificare nel loro centro il pixel in esame.

$$f(i,j) = \sum_{m=-r_i}^{r_i} \sum_{n=-r_j}^{r_j} h(r_i + 1 - m, r_j + 1 - n) g(i + m, j + n)$$

Questa sopra è l'equazione discreta della convoluzione con kernel h, che indichiamo con la notazione g * h . O rappresenta invece il rettangolo di dimensioni dispari dei vicini del punto in esame. Dal punto di vista matematico l'operatore di convoluzione gode delle seguenti proprietà:

- commutativa: X * Y = Y * X
- associativa: (X * Y) * Z = X * (Y * Z)
- distributiva: X * (Y + Z) = X * Y + X * Z

La correlazione segue lo stesso procedimento della convoluzione, ma nella convoluzione la maschera viene ruotata di 180° prima di essere utilizzata.

3.10 Descrivere i filtri per lo smoothing nel dominio spaziale

Gli operatori di smoothing si occupano di ridurre il rumore o piccole fluttuazioni in un immagine e si basano sul calcolo della media dei livelli di intensità degli intorni di alcuni pixel. Sfortunatamente questo comporta una perdita di dettaglio nei punti di edge, che contengono molte delle informazioni associate all'immagine stessa, per cui si cerca di avere degli operatori di smoothing edge preserving. Lo smoothing corrisponde alla soppressione delle alte frequenze nel dominio delle frequenze. Vengono solitamente applicati su intorni in cui i livelli di intensità

sono omogenei, non su intorni di pixel di edge. I filtri di smoothing sono efficaci in presenza di rumori come sale e pepe o strisce sottili, ma inefficaci in presenza di grandi macchie o strisce di grande spessore. Tra i diversi filtri abbiamo esaminato i seguenti:

- **Averaging**: Si assuma che il livello di rumore v in ogni pixel sia determinato da una variabile casuale di media zero e varianza σ. Avendo a disposizione n acquisizioni di una stessa scena statica il risultato dello smoothing sarebbe, per ogni pixel dell'immagine, la media delle n acquisizioni sommata alla media degli n rumori presenti sul singolo punto:

  Se non sono disponibili n acquisizioni di una stessa immagine il rumore si elimina mediando con l'intorno di un ogni pixel. Per ottenere risultati soddisfacenti occorre che il rumore sia di dimensione inferiore rispetto al più piccolo oggetto d'interesse presente nell'immagine. Gli edge verranno comunque sporcati. In questo senso l'averaging rappresenta un caso particolare di convoluzione in cui le maschere h sono del tipo

  Nel secondo e terzo esempio si da un maggior peso al pixel centrale e ai pixel del 4-intorno per meglio riflettere le proprietà del rumore Gaussiano.

- **Averaging con valori limitati**: Rimane il problema dell'edge preserving. Esiste una variante dell'averaging classico che consente di calcolare la media solo sui pixel che soddisfano un certo criterio, queste tecniche sono non lineari. Il primo criterio consiste nel prendere solo i pixel la cui luminosità rientra in un certo range; considerando un pixel (m,n):

  - se il valore associato ad (m,n) è valido non si effettua nessuna operazione
  - se, invece, il valore non è valido si applica l'averaging sul suo intorno O attraverso la maschera di convoluzione definita come:

$$h(i,j) = \begin{cases} 1 & \text{se } g(m+1, n+j) \notin [min, max] \\ 0 & \text{altrimenti} \end{cases}$$

  Un secondo metodo potrebbe essere quello di calcolare l'averaging su ogni pixel ma scartare i valori trovati che non rientrano in un certo range. Un terzo metodo utilizza la magnitudine degli edge come criterio (di fatto la norma del gradiente). La norma viene calcolata a priori per ogni punto dell'immagine e solo i punti il cui gradiente sta al di sotto di una certa soglia vengono utilizzati per l'averaging.

- **Median Smoothing**: La mediana di un insieme di valori è il valore centrale. Il median smoothing sostituisce il valore di luminosità di ogni pixel con la mediana delle luminosità dei pixel del suo intorno. Questa tecnica non produce picchi di rumore, elimina il rumore impulsivo molto bene (rumore sale e pepe), non sporca eccessivamente gli edge delle immagini e può essere implementata in modo iterativo.

  Il grosso svantaggio del median smoothing è che un intorno rettangolare rovina linee sottili e angoli acuti nell'immagine, per evitare ciò si ricorre ad un intorno di forma differente

Capitolo 4

Edge detection

4.1 Illustrare i differenti tipi di edge presenti in un'immagine

Nei punti di edge abbiamo un cambiamento brusco dell'intensità dei toni di grigio. L'edge è una proprietà associata ad un pixel calcolata dall'immagine in base al comportamento dei pixel nell'intorno del punto stesso. Un cambiamento della funzione immagine può essere descritto come un gradiente che punti in direzione della maggiore crescita della funzione immagine. Il gradiente avrà una sua magnitudo ed una sua direzione. La direzione dell'edge è perpendicolare rispetto la direzione del gradiente.

Possiamo identificare 4 tipi di edge, indicando nelle ordinate la magnitudo del gradiente e nelle ascisse una direzione x:

- step: caratterizzato da un'improvvisa variazione del livello di grigio. E' rappresentato con una funzione a gradino
- roof: detto anche "doppia rampa", caratterizzato da una costante e graduale variazione fino a un picco massimo dopo il quale si ritorna, in modo simmetrico, al valore di partenza
- line: il line, caratterizzato da un improvviso salto di livello di grigio al quale ne segue subito un altro verso il valore originario
- noisy: che è un edge a cui corrisponde una funzione della variazione dei livelli di grigio che parte da un valore e arriva a un altro, ma in modo irregolare

4.2 Classificazione degli edge detector

L'edge detection è molto importante perché il successo del processing di più alto livello dipende molto da una buona rilevazione degli edge. Le immagini in scala di grigio contengono tantissime informazioni, molte delle quali sono irrilevanti. L'idea è quella di ridurre la mole di dati eliminando le informazioni di scarso interesse. Si cerca quindi di evidenziare quelli che poi saranno i contorni

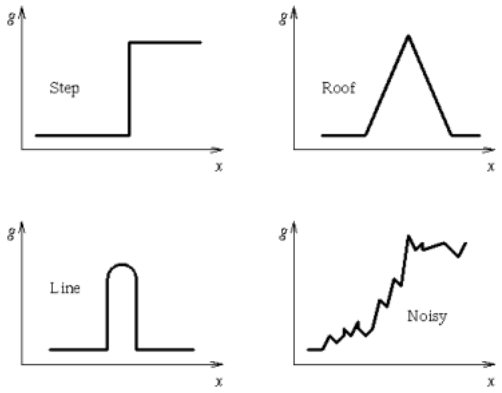

Figura 4.1: Differenti tipi di edge

dell'immagine. Gli edge detector localizzano cambi improvvisi di intensità nella funzione immagine.

Per localizzarli vi sono diversi modi e si possono suddividere in diverse classi:

- Una prima classe è costituita da operatori di tipo gradiente che si basano su approssimazioni della derivata prima
- La seconda classe è costituita da operatori che trattano l'immagine come una superficie vista come approssimazione discreta di una funzione bidimensionale continua, di cui si fa la derivata prima (in tal caso gli edge sono i punti in cui la derivata assume valori elevati) o la derivata seconda (gli edge sono i punti in cui la derivata attraversa lo zero)
- La terza classe di operatori cerca di risolvere il problema del rumore che può influenzare la bontà dell'edge detector. Viene inizialmente applicato un filtro (per esempio il filtro gaussiano) all'immagine su cui successivamente viene applicato un operatore differenziale

4.2.1 Dettagli?

Essi si basano sulle derivate parziali locali della funzione immagine. Le derivate sono maggiori nei punti in cui la funzione subisce un rapido cambiamento (sono i punti di edge). L'operatore gradiente ha il compito di indicare una tali punti aumentandone il dettaglio. Sfortunatamente applicando un operatore gradiente su un immagine il livello di rumore cresce notevolmente. Gli operatori basati su gradiente possono essere divisi in tre categorie:

1. operatori che approssimano le derivate tramite differenze (Roberts, Laplace, Prewitt, Sobel, Robinson, Kirsch)
   (a) invarianti alla rotazione (come il Laplaciano) richiedono una sola maschera di convoluzione
   (b) per approssimare la derivata prima servono diverse maschere di convoluzione e l'orientazione viene stimata sulla base del miglior matching tra diversi semplici pattern

2. operatori basati sullo zero crossing della derivata seconda
   (a) Marr-Hildreth
   (b) Canny Edge Detector
3. operatori che confrontano l'immagine con modelli parametrici di edge
   - i modelli parametrici descrivono molto dettagliatamente gli edge (più di norma e ŏdirezione) ma sono computazionalmente più costosi

Una categoria degli operatori basati su gradiente approssimano le derivate prime tramite differenze. Gli operatori gradiente che esaminano un piccolo intorno del punto sono di fatto delle convoluzioni e possono pertanto essere espressi per mezzo di maschere. Gli operatori che sono in grado di rilevare gli edge sono rappresentati da una serie di maschere, ognuna corrispondente ad una certa direzione.

4.3 Illustrare i metodi per l'edge detection basati sull'approssimazione delle derivate prime

Gli operatori che si basano sull'utilizzo della derivata prima (nella sua approssimazione nel caso discreto) consistono nell'applicazione di filtri, attraverso delle maschere di convoluzione. Ogni specifico operatore ha le sue maschere di convoluzione, solitamente per due o più diverse direzioni del gradiente.

- L'operatore di Roberts è quello computazionalmente più efficiente (utilizza una maschera 2x2) però va a evidenziare anche punti che non sono di edge, essendo molto influenzato dal rumore presente nell'immagine.
- Gli operatori di tipo compass consentono di individuare non solo il valore del gradiente ma anche l'orientazione degli edge, perché utilizzano delle maschere diverse per ogni tipo di orientazione. A ogni punto vengono applicate tutte le maschere, dopodiché per capire qual è la direzione dell'edge basta vedere quale maschera ha fornito la risposta massima. La risposta totale del gradiente è o la somma o il massimo delle risposte delle singole matrici.

4.4 Descrivere la tecnica della non-maxima suppression

La non-maxima suppression è una tecnica di edge detection utilizzata per identificare i pixel di edge partendo dalla mappa del gradiente. I pixel considerati di edge sono quelli che hanno un valore di gradiente superiore ai due pixel che lo precedono e che lo seguono, nella direzione del gradiente.

4.5 Illustrare il Marr-Hildreth edge detector

L'operatore Marr-Hildreth (detto anche LoG) è un operatore di edge detection che unisce gli operatori gaussiano e laplaciano. La sua applicazione si compone di tre passi fondamentali:

- applicazione del filtro di smoothing gaussiano così da attenuare il rumore presente nell'immagine
- applicazione dell'operatore laplaciano che calcola la derivata seconda dell'immagine evidenziando i punti in cui subisce brusche variazioni
- localizzazione dei punti in cui la derivata seconda attraversa lo zero (zero crossing)

In realtà, grazie alla linearità delle operazioni, viene applicato il laplaciano sulla matrice di convoluzione dell'operatore gaussiano, che viene poi applicata all'immagine.

4.6 Illustrare il Canny edge detector

Il Canny Edge Detector è ottimo edge detector per il white noise. È riconosciuto come l'operatore migliore per l'edge detection per tre motivi:

1. Ricerca: gli edge importanti vengono sempre rilevati
2. Localizzazione: la distanza tra il punto in cui l'edge viene rilevato e la sua locazione reale è minimale
3. Le risposte multiple (più localizzazioni per un singolo edge) sono ridotte al minimo (questo punto ricopre in parte ciò che viene definito nel primo punto)

4.6.1 Implementazione

- Si applica un filtro Gaussiano all'immagine
- Si calcola il gradiente (norma e direzione) mediante la tecnica delle differenze: a questo scopo vengono utilizzate due maschere di convoluzione 2x2 che restituiscono rispettivamente $p(x, y)$ e $q(x, y)$ dell'edge secondo la formula:
- Si applica la soppressione dei non-maxima alla norma del gradiente in modo da assottigliare gli edge che altrimenti sarebbero troppo spessi
- Si applica la soppressione dei non-maxima alla norma del gradiente in modo da assottigliare gli edge che altrimenti sarebbero troppo spessi
- Si usa una doppia sogliatura per trovare e collegare gli edge: si definiscono due soglie τ_1 e τ_2 tali che $\tau_2 \approx 2\tau_1$ e si producono due mappe degli edge $T_1(x, y)$ e $T_2(x, y)$. La mappa costruita con la soglia più alta conterrà sicuramente pochissimi edge falsi ma quelli veri risulteranno probabilmente spezzati per cui la mappa finale viene costruita partendo da $T_2(x, y)$ e collegando i contorni solo se, posizionatisi all'estremo di un edge, nell'8-intorno associato a tale punto in $T_1(x, y)$ ci sono degli edge che possono essere collegati. L'algoritmo continua a collegare edge sino a quando è stato colmato il percorso vuoto che conduce ad un altro edge in $T_2(x, y)$

4.7 Quali sono i vantaggi e gli svantaggi degli edge detector?

Gli edge detector in generale sono fondamentali per poter proseguire ai livelli superiori dell'elaborazione di un'immagine (come la segmentazione). Per descrivere vantaggi e svantaggi degli edge detector è opportuno dividere gli operatori in classi.

1. La **prima classe** contiene operatori gradiente di dimensione 3x3 o 5x5 (Prewitt, Robert, Sobel, Laplacian). Questi operatori lavorano abbastanza bene su scene sintetiche ma disegnano degli edge molto spessi, per cui spesso necessitano di un operazione di thinning successiva. L'operatore di Roberts ha un'alta sensibilità al rumore a causa del basso numero di pixel usati per calcolare il gradiente. Uno svantaggio dell'operatore laplaciano è che può rispondere in modo doppio a certi edge a causa del rumore.Gli operatori che si basano sul calcolo della derivata prima (Prewitt, Robert, Sobel, Kirsch, Robinson) forniscono, oltre al suo modulo, anche la direzione dell'edge.

2. Nella **seconda classe** sono contenuti gli operatori che si basano sul concetto di zero crossing. Ovvero individuando l'intorno in cui si presenta l'attraversamento dello zero nelle differenze fra le derivate seconde. Fanno parte di questo gruppo il LoG e il Canny edge detector. I LoG ha il vantaggio, rispetto agli operatori della prima classe, di ridurre il rumore in fase di preprocessing e quindi non fornisce risposte doppie agli per gli edge.

   Il Canny edge detector è l'ideale per edge di tipo step corrotte da white noise. La sua ottimalità è dovuta al rispetto di 3 criteri fondamentali:

   - Trova tutti gli edge importanti
   - La distanza tra la posizione reale dell'edge e l'edge individuato è minima
   - Minimizza le risposte multiple perché effettua la soppressione degli edge che non hanno il gradiente massimo

4.8 Quali sono i criteri per valutare la bontà di un edge detector?

I criteri per valutare la bontà di un edge detector sono:

- la probabilità che vengano classificati come edge degli edge falsi
- la probabilità che non vengano riportati degli edge significativi
- l'errore commesso nella valutazione degli angoli fra gli edge
- la distanza tra gli edge estratti e gli edge reali, cioè quanto effettivamente gli edge estratti corrispondono agli edge reali
- la tolleranza agli angoli, alle giunzioni, ai punti che rappresentano degli spigoli, cioè quanto l'operatore riesce a localizzarli bene

La bontà di un edge detector, e quindi l'insieme di questi requisiti, viene valutata mediante un parametro F [1]. Il risultato è sempre compreso fra 0 e 1. Un valore corrispondente a 1 indica che l'edge detector estrae esattamente tutti gli edge significativi. La bontà di un edge detector si valuta quindi facendolo lavorare su un'immagine teorica e aggiungendo del rumore per vedere se il parametro F rimane vicino a 1.

[1] vedi slide 13 lect09 new

Capitolo 5

Image segmentation

5.1 Cosa è l'image segmentation?

La segmentazione è uno dei passi più importanti nell'analisi delle immagini processate

Il suo scopo è quello di **dividere** l'immagine in parti che hanno una forte correlazione con oggetti o aree che si trovano nel mondo reale.

5.2 Quanti tipi di segmentazione abbiamo?

Abbiamo due tipi di segmentazione:

1. **Segmentazione Completa** in cui le regioni corrispondono direttamente con gli oggetti dell'immagine in ingresso. In pratica ogni elemento dell'immagine viene riconosciuto e isolato. Per ottenere questo risultato è necessario avere informazioni di processing di alto livello (high level processing) da utilizzare per comprendere il dominio del problema da risolvere (che ci permette di riconoscere gli oggetti/regioni)

2. **Segmentazione Parziale** in cui le regioni/segmenti di immagini non corrispondono direttamente con gli oggetti reali rappresentati nell'immagine. In questo caso l'immagine è divisa in regioni omogenee secondo una determinata proprietà scelta (colore, texture, riflettività).

   La segmentazione parziale è seguita da un ulteriori processi sino ad arrivare all'immagine finale di segmentazione che può essere trovata grazie all'aiuto di informazioni di alto livello.

5.3 In quali campi viene utilizzata la segmentazione?

Ecco alcune delle problematiche più comuni che possono essere risolte con la segmentazione:

- ottenere dati da immagini ambigue

- rumore nelle informazioni
- ottenere oggetti con discreto contrasto su di uno sfondo uniforme
- semplici lavori di raggruppamento come cellule del sangue, caratteri stampati

In generale è difficile ottenere una corretta e completa segmentazione; per questo spesso di utilizza una segmentazione parziale come input in un processo di alto livello.

5.4 Quali sono i metodi utilizzati per la segmentazione?

- Thresholding (Approccio Globale) che si basa sull'istogramma di alcune caratteristiche dell'immagine come il **gray level thresholding** o **soglia del livello di grigio**. Si ricercano i toni soglia che separano oggetti differenti nell'immagine. E' il metodo più semplice e veloce
- Segmentazione basata sugli Edge in cui ci cercano i contorni degli oggetti basandosi sulla discontinuità dei toni, cioè sugli edge
- Segmentazione basata sulle Regioni in cui si ricercano le regioni, cioè degli oggetti uniformi secondo un certo criterio (stesso tono di grigio, o tono di grigio che si differenzia al massimo di una certa soglia)
  - region growing
  - region splitting and merging

5.5 Cosa si intende con Approccio Globale: Gray level thresholding?

E' il sistema più semplice a livello concettuale di segmentazione e per questo è anche il sistema più efficiente computazionalmente e veloce.

Esso viene applicato alle immagini che contengono regioni o oggetti con un costante livello di riflettività o assorbimento di luce. Si sfrutta questa caratteristica per partizionare l'immagine grazie ai valori di intensità e/o alle proprietà di questi valori.

5.6 Quale è l'idea alla base del gray level thresholding con due elementi (mode) da separare?

1. Si supponga di avere un istogramma di intensità di una immagine *f(x,y)* che è composta da un oggetto bianco (level 255) e di uno sfondo scuro (level 0) e che si voglia estrarre l'oggetto dallo sfondo.

2. Inizialmente si seleziona una soglia T con cui si ottiene l'immagine segmentata *g(x, y)*:

$$g(x,y) = \begin{cases} 1 & se \quad f(x,y) > T \\ 0 & se \quad f(x,y) \leq T \end{cases}$$

3. Se il valore di intensità dell'immagine è superiore a quello della soglia, si assegna a *g(x, y) il valore 1, altrimenti 0*

5.7 Quale è l'idea alla base del gray level thresholding con tre elementi (mode) da separare?

- Maggiore è il numero di elementi (mode), più complessa sarà la segmentazione. Esaminiamo il caso di una immagine con due oggetti chiari posti su uno sfondo scuro.
- L'immagine segmentata risultante *g(x, y)* sarà:

$$g(x,y) = \begin{cases} a & se \quad f(x,y) > T_2 \\ b & se \quad T_1 < f(x,y) \leq T_2 \\ c & se \quad f(x,y) \leq T_1 \end{cases}$$

dove *a, b, c* sono tre diversi valori di intensità e T_1 e T_2 sono due differenti valori di soglia.

5.8 Nel gray level thresholding quanti e quali tipi di soglia possiamo avere?

Possiamo avere due tipi di soglia:

1. Soglia Globale
2. Soglia Variabile, che si divide in soglia locale o regionale

5.9 Il thresholding applicato ad immagini con rumore e variazioni di luminosità: cosa cambia?

Esaminando l'istogramma dell'immagine si nota che questo subisce delle variazioni a seconda che l'immagine si affetta da rumore o una intensità variabile.

5.10 Cosa è il single global threshold e quando si applica?

Il single global threshold si applica a quelle immagini in cui l'intensità degli oggetti e dello sfondo rende questi facilmente distinti. L'algoritmo che si applica all'intera immagine si basa sui seguenti passi:

1. Si seleziona una soglia T
2. Si effettua la segmentazione dell'immagine usando la soglia T producendo due gruppi di pixel:
   - G_1 dato da tutti i pixel che hanno una intensità $\geq T$
   - G_2 dato da tutti i pixel che hanno una intensità $< T$
3. Si calcolano i valori di **intensità medi** m_1 e m_2 rispettivamente per i gruppi di pixel G_1 e G_2
4. Si determina un nuovo livello di soglia T secondo la formula:

$$T = \frac{1}{2}(m_1 + m_2)$$

5. Si ripetono i passi dal 2 al 4 sino a quando la differenza tra i valori di T (delle successive iterazioni) è più piccolo rispetto al valore predefinito T_0

5.11 Quali sono le caratteristiche del single global threshold?

- E' un algoritmo semplice.
- Funziona bene nei casi in cui l'istogramma dell'immagine presenta delle campane (le mode) corrispondenti agli oggetti e allo sfondo separate da una valle.
- Il valore di soglia iniziale deve essere maggiore rispetto al minimo valore di intensità e minore rispetto al massimo valore di intensità nell'immagine.

5.12 Qual è l'algoritmo Matlab per il global thresholding?

Indicando con:

- f per l'immagine di input
- g per l'immagine di output
- T per la soglia

```
function [g]=iter_thresh(f)
T = 0.5 * (double(min(f(:)))+ double(max(f(:))));
flag = false;
while ~flag
  g = f >= T;
  Tnext = 0.5 * (mean(f(g)) + mean(f(~g)));
  flag = abs (T - Tnext) < 0.5;
  T = Tnext;
end
```

5.13 Cosa è l'Optimum global thresholding o metodo Otsu?

Meglio conosciuto come Metodo Otsu è un è un metodo di sogliatura automatica dell'istogramma nelle immagini digitali.

Viene definito come "ottimo" perché **massimizza la varianza tra le classi**. Un algoritmo di sogliatura (threshold) che fornisce la migliore separazione tra le classi in termini dei loro valori di intensità è il miglior threshold.

5.14 Come funziona il metodo Otsu?

- Si considerino:
  - {0, 1, 2, ... , L-1} siano gli L distinti livelli di intensità in una immagine
  - MxN sia la dimensione dell'immagine di M righe X N colonne
  - n_i indica il numero di pixel con intensità i
  - $MN = n_0 + n_1 + n_2 + \dot{,} + n_{L-1}$
- l'istogramma normalizzato ha componenti $p_i = \frac{n_i}{MN}$ da cui segue:

$$\sum_{i=0}^{L-1} p_i = 1 \quad p_i \geq 0$$

- Selezionando e utilizzando una soglia $T(k) = k$ con $0 < k < L-1$ con l-immagine di input, si ottengono le due classi c_1 e C_2 dove:
  - C_1 è la classe di tutti i pixel con valori di intensità compresi nel range $[0, k]$
  - C_2 è la classe di tutti i pixel con valori di intensità compresi nel range $[k+1, L-1]$
- Usando questa soglia, si può calcolare la *probabilità* $P_1(k)$ che un pixel appartenga alla classe C_1:

$$P_1(k) = \sum_{i=0}^{k} p_i$$

- La *probabilità* che il pixel invece appartenga alla classe C_2 è data da:

$$P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$$

- Il valore di *intensità media* di tutti i pixel della classe C_1 è:

$$m_1(k) = \sum_{i=0}^{k} iP(\frac{i}{C_1}) = \sum_{i=0}^{k} iP(\frac{C_1}{i})\frac{P(i)}{P(C_1)} = \frac{1}{P_1(k)}\sum_{i=0}^{k} ip_i$$

- Il valore di *intensità media* di tutti i pixel della classe C_2 è invece:

$$m_2(k) = \sum_{i=k+1}^{L-1} iP(\frac{i}{C_2}) = \frac{1}{P_2(k)}\sum_{i=k+1}^{L-1} ip_i$$

- La *media complessiva* sino al livello k è data da:

$$m(k) = \sum_{i=0}^{k} ip_i$$

- Invece l'*intensità media* dell'intera immagine (**la media globale**) è data da:

$$m_G = \sum_{i=0}^{L-1} ip_i$$

- Inoltre valgono le seguenti:

$$P_1 m_1 + P_2 m_2 = m_G$$

$$P_1 + P_2 = 1$$

- Per valutare la bontà della soglia al livello k si usa la *metrica normalizzata*:

$$\eta = \frac{\sigma_B^2}{\sigma_G^2}$$

  - dove σ_G^2 è la **varianza globale** ed è *costante*:

$$\sigma_G^2 = \sum_{i=0}^{L-1}(i - m_G)^2 p_i$$

  - dove σ_B^2 è la **varianza tra le classi** che indica la misura della *separabilità* tra le classi:

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 = P_1 P_2(m_1 - m_2)^2 = \frac{(m_G P_1 - m)^2}{P_1(1 - P_1)}$$

- (The farther the two means m 1 and m 2 are from each other the larger will be.)
- Poiché σ_G^2 è una costante ne consegue che η è una *misura della separabilità*. Inoltre massimizzare questa metrica, è identico a massimizza σ_B^2
- l'obiettivo principale è determinare il valore k di soglia che massimizza la varianza di classe.
- Introducendo nuovamente k si ha:

$$\eta(k) = \frac{\sigma_B^2}{\sigma_G^2}$$

e

$$\sigma_B^2(k) = \frac{[m_g P_1(k - m(k)]^2}{P_1(k[1 - P_1(k)]}$$

- La **soglia ottimale** è il valore k^* che massimizza $\sigma_B^2(k)$:

$$\sigma_B^2(k^*) = \max_{0 \leq k \leq L-1} \sigma_B^2(k)$$

- Per trovare K^* si deve valutare questa equazione per tutti i valori interi di k e scegliere il valore che produce il massimo $\sigma_B^2(k)$
- Se il massimo esiste *per più di un valore di k* è consuetudine mediare i diversi valori di k per cui $\sigma_B^2(k)$ è massima.
- La metrica normalizzataη valutata per il valore di soglia ottimale $\eta(k^*)$ può essere utilizzata per ottenere una stima quantitativa della separabilità delle classi, che a sua volta fornisce un'idea della facilità di sogliatura di una data immagine. Il suo valore oscilla tra 0 e 1:
  - il valore più basso è ottenibile solo da immagini con un singolo livello di intensità costante
  - Il limite superiore invece è ottenibile solo dalle immagini 2-valori con intensità uguali a 0 e $L - 1$

5.15 Spiegare le sogliature multiple

Il metodo di Otsu può essere esteso ad un numero arbitrario di soglie in modo da valutare un diverso numero di classi.

Generalmente i metodi che utilizzano più di due soglie vengono risolte con più valori di intensità (colore).

Un altro metodo è quello di suddividere l'immagine in *rettangoli sovrapposti*

- questo sistema viene utilizzato per compensare i casi di illuminazione non uniforme
- i rettangoli vengono scelti di piccole dimensioni in modo che l'illuminazione da essi coperta, sia uniforme

5.16 Illustra il metodo Otsu in Matlab

La funzione *graythresh* di Matlab permette di calcolare una soglia utilizzando il metodo di Otsu.

5.17 Tecniche per migliorare il global thresholding

- Applicare lo smoothing all'immagine prima del thresholding: è una tecnica che si applica quando il rumore non può essere ridotto e come metodo di sogliatura si seleziona il global thresholding. Di seguito un esempio di codice Matlab.

```
f=imread('large_septagon_gaussian_noise_mean_0_std_50_added.
    tif');
figure, imshow(f);
figure, imhist(f);
T=graythresh(f);
T=T*255
g=f>=T;
figure, imshow(g);
filt=fspecial('average', [5 5]);
ff=imfilter(f, filt);
figure, imshow(ff);
figure, imhist(ff);
ylim([0 16000])
T1=graythresh(ff);
T1=T1*255
g1=ff>=T1;
figure, imshow(g1);
```

- Thresholding variabile attraverso il partizionamento dell'immagine. Di seguito un esempio di codice Matlab.

Capitolo 6

Segmentazione

6.1 Formulazione matematiche di base

- R rappresenta l'intera regione dell'immagine
- la segmentazione è il processo che divide la regione R nelle sottoregioni $R_1, R_2, \ldots, R_n$ per cui valgono
  1. $\bigcup_{i=1}^{n} R_i = R$ ovvero la somma di tutte le sottoregioni è uguale all'intera regione R dell'immagine
  2. R_i è un insieme connesso dove $i = 1, 2, \ldots, n$
  3. $R_i \cap R_j = \emptyset \quad \forall i, j \quad i \neq j$
  4. $Q(R_i) = TRUE \quad i = 1, 2, \ldots, n$
  5. $Q(R_i \cup R_j) = FALSE$ per ogni regione adiacente R_i e R_j

6.2 Region growing

E' una procedura che raggruppa i pixel o le sottoregioni in regioni più grandi basandosi su criteri predefiniti.

L'approccio base è il seguente:

- si parte con un insieme di punti "seme"
- da queste grow regioni si aggiungono ad ogni seme i suoi pixel vicini che hanno dele predefinite proprietà simili a quelle del seme (come uno specifico range di intensità o un colore)
- è importante selezionare con attenzione i punti *da cui partire* in base al tipo di problema o all'immagine da processare.
  - le immagini satellitari dipendono dal colore
  - le immagini monocromatiche dipendono dai descrittori basati sui livelli di intensità e sulle proprietà spaziali (texture, momenti)
- devono essere usate le proprietà della connessione
- si deve formulare una regola di stop

6.3 Qual è l'algoritmo di un region growing?

- si assume che:
  - $f(x, y)$ è un'immagine in ingresso
  - $S(x, y)$ indica un array di semi contenente un 1s nei posti dei punti del seme e 0 negli altri punti
  - Q indica un predicato che verrà applicato ad ogni locazione (x, y)
- si assume che gli array f ed S abbiano la stessa dimensione
- si utilizza una 8-connessione

Con queste premesse si applica il seguente algoritmo:

1. si trovano tutte le componenti in $S(x, y)$
   - si erode ogni componente connessa sino ad ottenere un solo pixel
   - si etichetta (label) ogni pixel trovato con il valore 1
   - si etichettano tutti gli altri pixel in S con il valore 0
2. si realizza un'immagine f_q, tale che per ogni coppia di coordinate (x, y) si abbia che:
   - $F_q(x, y) = 1$ se l'immagine in ingresso soddisfa il predicato Q a queste coordinate
   - $f_q(x, y) = 0$ negli altri casi
3. si forma così un'immagine g realizzata aggiungendo ad ogni punto seme in S tutti i punti 1-valutati in f_q che sono 8-connessioni al punto seme
4. si etichetta ogni componente connessa in g con una etichetta di regione differente (1, 2, 3, ...). Questa è così l'immagine segmentata ottenuta attraverso l'algoritmo region growing

6.4 Region Splitting and Merging

6.4.1 Definizione

Si tratta di una procedura che suddivide un'immagine prima di tutto in un insieme arbitrario di regioni disgiunte e quindi le fonde (merge) e/o le suddivide in regioni che soddisfino i requisiti della segmentazione.

6.4.2 Region Splitting and Merging: algoritmo

- si assuma che:
  - R è la regione che rappresenta l'intera immagine
  - sia Q il predicato utilizzato

- si parte con l'intera regione R suddividendo questa in quadranti sempre più piccoli R_i, dove per ognuno di essi vale che $Q(R_i) = TRUE$.
  - se $Q(R) = FALSE$ si divide l'immagine in nuovi quadranti
  - se $Q = FALSE$ per ogni quadrante, si suddivide il quadrante in ulteriori sotto-quadranti e così via

6.4.3 Region Splitting and Merging: tecniche di rappresentazione

Una delle tecniche più utilizzata è denominata **Quadtree**:

- si tratta di un albero in cui cui ogni nodo ha 4 discendenti
- la radice corrisponde all'intera immagine
- ogni nodo corrisponde a 4 foglie

6.4.4 Caratteristiche

- se viene usato solo lo *splitting* la partizione finale contiene regioni adiacenti con proprietà identiche
- soddisfare i vincoli di segmentazione richiede la fusione solo delle regioni adiacenti i cui pixel soddisfano il predicato Q:
  - due regioni adiacenti R_i e R_j sono fuse solo se vale: $Q(R_i \cup R_j) = TRUE$

6.4.5 Algoritmo per lo splitting and merging

1. si suddivide in 4 quadranti disgiunti ogni regione R_j per cui $Q(R_i) = FALSE$
2. quando non è possibile un ulteriore suddivisione, si fonde ogni regione R_i e R_j adiacente per cui $Q(R_i \cup R_j) = TRUE$
3. ci si ferma quando non sono possibili ulteriori fusioni

Si è soliti indicare una dimensione minima quadregion oltre il quale nessuna ulteriore frazionamento sia effettuato.

una variante: la fusione di due qualsiasi regioni adiacenti R_i e R_j se ognuno soddisfa il predicato singolarmente.

6.4.6 Region growing Matlab

```
function [g, NR, SI, TI] = regiongrow(f, S, T)
f = double(f);
% if S is a scalar, obtain the seed image
if numel(S) == 1
SI = f == S;
S1 = S;
else
SI = bwmorph(S, 'shrink', Inf);
```

```
J = find(SI);
S1 = f(J); % Array of seed values
end
TI = false(size(f));
for K = 1:length(S1)
seedvalue = S1(K);
S = abs(f - seedvalue) <= T;
TI = TI | S;
end
[g, NR] = bwlabel(imreconstruct(SI, TI));
```

S can be an array (the same size as f) with a 1 at the coordinates of every seed point and 0s elsewhere. S can also be a single seed value. (Our example S = 255)

Similarly, T can be an array (the same size as f) containing a threshold value for each pixel in f. T can also be a scalar, in which case it becomes a global threshold. (Our example T = 65)

g is the result of region growing, with each region labelled by a different integer.

NR is the number of regions.

SI is the final seed image used by the algorithm.

TI is the image consisting of the pixels in f satisfied the threshold test.

Use function imreconstruct with SI as the marker image to obtain the regions corresponding to each seed in S.

bwlabel assigns a different integer to each connected region.

6.4.7 Region growing Matlab: le funzioni

- la funzione predefinita per implementare il quadtree è *qtdecomp* con la sintassi:

$$S = qtdecomp(f, @split_test, parameters)$$

  - f è l'immagine di ingresso
  - S è una matrice sparsa contenente la struttura quadtree
    - se $S(k,m) \neq 0$, allora (K,m) è l'estremo superiore sinistro del blocco in decomposizione e la dimensione del blocco è data da $S(k,m)$
  - la funzione $split_test$ è usata per determinare se una regione deve essere divisa o meno
  - $parameters$ indica eventuali parametri addizionali

- per ottenere i valori dei pixel della quadregion in una decomposizione quadtree, si utilizza la funzione **qtgetblk** con la sintassi:

$$[vals, r, c] = qtgetblk(f, S, m)$$

  - $vals$ è un array che contiene i valori dei blocchi di dimensione mXm nella decomposizione quadtree f

  - S è la matrice sparsa restituita dalla funzione *qtdecomp*
  - i parametri r, c sono dei vettori che contengono le coordinate della riga e della colonna degli angoli in alto a sinistra dei blocchi

- la funzione che implementa l'algoritmo di segmentazione è **splitmerge** che la seguente sintassi:

$$g = splitmerge(f, mindim, @predicate)$$

  beginitemize

- f è l'immagine di ingresso
- g è l'immagine in uscita
- *mindim* definisce la dimensione del più piccolo blocco nella decomposizione (potenza di 2)
- *predicate* è una funzione definita dall'utente che deve essere inclusa nell'ambiente (path) di Matlab. La sua sintassi è:
  - $flag = predicate(region)$
  - restituisce *true* (1) se i pixel nella regione soddisfa il predicato definito dal codice nella funzione
  - restituisce $false$ (0) negli altri casi
  -

```
function flag = predicate(region)
sd = std2(region);
m = mean2(region);
flag = (sd > 10) & (m > 0) & (m < 125);
```

6.4.8 Region Splittin and Merging in Matlab: splitmerge

```
function g = splitmerge(f, mindim, fun)
Q = 2^nextpow2(max(size(f)));
[M, N] = size(f);
f = padarray(f, [Q - M, Q - N], 'post');

S = qtdecomp(f, @split_test, mindim, fun);

Lmax = full(max(S(:)));

g = zeros(size(f));
MARKER = zeros(size(f));

for K = 1: Lmax
  [vals, r, c] = qtgetblk(f, S, K);
  if ~isempty(vals)
    for I = 1:length(r)
      xlow = r(I);
      ylow = c(I);
      xhigh = xlow + K -1;
```

```
      yhigh = ylow + K -1;
      region = f(xlow:xhigh, ylow:yhigh);
      flag = feval(fun, region);
      if flag
        g(xlow:xhigh, ylow:yhigh) = 1;
        MARKER(xlow, ylow) = 1;
      end
    end
  end
end

g = bwlabel(imreconstruct(MARKER, g));
g = g(1:M, 1:N);
```

6.4.9 Region Splitting and Merging in Matlab: split test

```
function v = split_test(B, mindim, fun)

k = size(B,3);
v(1:k) = false;

for I = 1:k
  quadregion = B(:, :, I);
  if size(quadregion, 1) <= mindim
    v(I) = false;
    continue;
  end

  flag = feval(fun, quadregion);
  if flag
    v(I) = true;
  end
end
```

Capitolo 7

Morfologia

7.1 Introduzione

La morfologia è una branca della biologia che tratta la forma e la struttura di animali e piante. Nel contesto dell'elaborazione delle immagini, la **morfologia matematica** è uno strumento che è utile anch'esso per:

- estrarre componenti dall'immagine che siano utili per rappresentare e descrivere regioni come confini, scheletri, e superfici convessi
- attuare tecniche di pre- post-processing come il filtraggio morfologico il thinning (assottigliamento) e il pruning (riduzione)

7.1.1 Campi di utilizzo della morfologia matematica

- image enhancement
- image restoration
- noise reduction
- space-time filtering
- image segmentation
- edge detection
- texture analysis
- particle analysis
- component analysis
- shape analysis
- feature generation
- feature detection
- skeletonization

- general thinning
- curve filling
- image compression

7.1.2 Il linguaggio della morfologia matematica

Il linguaggio alla base della morfologia matematica (denominato semplicemente MM) è basato sulla **teoria degli insiemi**, dove gli insiemi rappresentano degli oggetti all'interno delle immagini.

Si campionano le coordinate xy del piano e si rappresentano in una matrice. Nelle immagini binarie gli elementi delle immagini sono membri dell'insime $ZxZ(Z^2)$, dove ogni elemento di un insieme è una tupla (vettore 2D) le cui coordinate sono le coordinate (x, y) di un dato pixel.

Le immagini digitali in scala di grigio sono rappresentate come insiemi le cui componenti si trovano in Z^3, dove le prime due coordinate fanno riferimento alle coordinate di un pixel e la terza corrisponde al suo valore discreto di intensità.

Gli spazi dimensionali di dimensioni maggiori possono contenere ulteriori attributi dell'immagine (colore, tempo) Gli **insiemi** nella MM rappresentano le forme degli oggetti in una immagine.

- l'insieme di tutti i pixel neri in una immagine binaria è una descrizione completa dell'immagine
- in una immagine binari gli insiemi in questione sono membri Z^2 dove ogni elemento di un insieme è una 2-tupla (o un 2-D vettore) le cui le coordinate x, y sono le coordinate di un pixel bianco (o nero a seconda della convenzione adottata) nell'immagine

7.1.3 Altri concetti della teoria degli insiemi

- sia A un insieme in z^2 con elementi (x, y):
  - se $w = (x, y)$ è un elemento di A, $\quad w \in A$
  - se $w = (x, y)$ non è un elemento di A, $\quad w \notin A$
- un insieme di pixel B che soddisfa una particolare condizione è scritto come:
$$B = \{w|\text{condizione}\}$$
- il **complemento di A** è l'insieme di tutte le coordinate di pixel che non appartengono ad A è indicato con A^c:
$$A^C = \{w|w \notin A\}$$
- l'**unione** di due insiemi A e B è l'insieme formato da tutti gli elementi di A e B ed è indicato come:
$$C = A \cup B$$
- l'**intersezione** di due insiemi A e B è l'insieme formato da tutti gli elementi che appartengono sia ad A che a B ed è indicato come:
$$C = A \cap B$$

- la **differenza** di due insiemi A e B è l'insieme degli elementi che appartengono ad A ma non a B ed è indicato come:

$$A \breve{} B = \{w | w \in A, w \notin B\}$$

- la **riflessione** di un insieme B indicato con $\hat{B}$:

$$\hat{B} = \{w | w = -b, \quad b \in B\}$$

- la **traslazione** di un insieme B di un punto $z = (z_1, z_2)$ è indicato con $(B)_z$ e definito come:

$$(B)_z = \{c | c = b + z, \quad b \in B\}$$

7.1.4 Altri concetti dalla teoria degli insiemi

- una immagine binaria può essere vista come una funzione bivalued di x e y
- per la teoria MM una immagine binaria è un insieme di pixel in primo piano (1-valued), che si trovano in Z^2
- le operazioni come unione e intersezione possono essere applicate direttamente gli insiemi di immagini binarie
- le operazioni di riflessione e traslazione sono impiegate spesso nella morfologia per formulare operazioni basate sugli **elementi strutturati** (SEs)

7.1.5 Elemento strutturante

Gli elementi strutturanti, indicati come SE (Structuring Elements) sono piccoli insiemi o sottoimmagini usate per sondare un'immagine riguardo alle proprietà desiderate.

- per ogni SE deve essere specificata con precisione anche la sua origine, solitamente indicata con un puntino nero. Quando si ha a che fare con SE simmetrici si assume che tale origine coincida con il centro della simmetria.
- gli SE quando lavorano con le immagini, vengono rappresentate come delle matrici rettangolari e ciò si ottiene aggiungendo il minor numero possibile di elementi di sfondo (mostrati come non ombreggiati) necessari per formare una matrice rettangolare
- nel caso degli insiemi (su cui opera l'SE) anche questo viene convertito in una matrice rettangolo aggiungendo elementi di sfondo. Il bordo dello sfondo deve essere abbastanza grande da contenere l'intero elemento strutturante quando la sua origine si trova sul bordo dell'insieme originale (operazione analoga a quella vista nel padding)
- si crea un nuovo insieme facendo scorrere l'elemento strutturante B sull'insieme A in modo che l'origine di B visiti ogni elemento di A
- se una posizione dell'insieme A contiene completamente B, allora questa posizione viene marcata come appartenente ad un nuovo insieme (indicato come ombreggiato), altrimenti non viene marcato (indicato come non ombreggiato)

7.2 Operazioni MM

Qui di seguito verranno illustrate alcune delle operazioni di uso più frequente nella morfologia che sono invarianti alla traslazion:

- erosione (shrinking)
- dilatazione (growing)
- opening (smoothing)
- closing (smoothing)
- trasformazioni Hit-or-Miss

7.2.1 Erosione

Se A e B sono due insiemi in Z^2, l'erosione di A attraverso B in Z^2 è indicata come:

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

e viene definita come l'insieme di tutti i punti z tali che B traslato di z, sia contenuto in A.

L'affermazione che B è contenuto in A è equivalente a dire che B non ha elementi in comune con lo sfondo. Per questo motivo l'erosione può anche essere definita come:

$$A \ominus B = \{z|(B)_z \cap A^c = \emptyset\}$$

7.2.2 Dilatazione

Se A e B sono due insiemi in Z^2, la dilatazione di A attraverso B in Z^2 è indicata come:

$$A \oplus B = \{z|(\hat{B})_z \cap A \neq \varnothing\}$$

Questa equazione è basata sulla riflessione di B rispetto alla sua origine, e sulla traslazione di questa riflessione attraverso z. La dilatazione di A attraverso B è l'insieme di tutti gli spostamenti z, tali che $\hat{B}$ e A si sovrappongono almeno di un elemento.

La definizione può essere scritta in maniera equivalente come:

$$A \oplus B = \{z|[(\hat{B})_z \cap A] \subseteq A\}$$

Il processo di base di invertire (ruotare) B rispetto alla sua origine e successivamente spostarlo in modo tale da scorrere sull'insieme (immagine) A è analogo alla convoluzione spaziale vista precedentemente.

La dilatazione si basa su operazioni con insiemi ed inoltre è un'operazione non lineare, mentre la convoluzione è un'operazione lineare

A differenza dell'erosione, che è un'operazione di eliminazione o assottigliamento, la dilatazione "accresce" o "ispessisce" gli oggetti di un'immagine binaria.

Le modalità e le dimensioni di questo ispessimento vengono controllate dalla forma dell'elemento strutturante utilizzato.

7.2.3 Erosione e dilazione: dualità

L'erosione e la dilatazione sono duali l'una dell'altra rispetto al complementare e alla riflessione di un insieme. Cioè:

$$(A \ominus B)^c = A^c \oplus \hat{B}) \quad \text{e} \quad (A \oplus B)^c = A^c \ominus \hat{B}$$

La prima equazione indica che l'erosione di A attraverso B è il complemento della dilatazione di A^c attraverso $\hat{B}$ e viceversa.

La proprietà duale è molto utile quando l'elemento strutturante è simmetrico rispetto alla sua origine (come avviene spesso) così che $\hat{B} = B$. Quindi è possibile ottenere l'erosione di una immagine attraverso B semplicemente dilatando il suo sfondo (cioè dilatando A^c) con lo stesso elemento strutturante e complementando il risultato. Considerazioni simili possono essere fatta con la seconda equazione.

Verifichiamo in maniera formale la validità della prima equazione. Dalla definizione di erosione ne segue che:

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

Se l'insieme $(B)_z$ è contenuto in A, allora $(B)_z \cap A^c = \varnothing$, caso in cui l'equazione precedente diventa:

$$(A \ominus B)^c = \{z|(B)_z \cap A^c = \varnothing\}^c$$

Ma il complemento dell'insieme di valori z che soddisfa $(B)_z \cap A^c = \varnothing$ è l'insieme di valori z tali che $(B)_z \cap A^c \neq \varnothing$. Quindi:

$$(A \ominus B)^c = \{z|(B)_z \cap A^c \neq \varnothing\}$$

dove l'ultimo passaggio deriva dall'equazione $A \oplus B = \{z|(\hat{B})_z \cap A \neq \varnothing\}$ vista precedentemente nel paragrafo sulla dilatazione.

In maniera analoga si può dimostrare la seconda equazione $(A \oplus B)^c = A^c \ominus \hat{B}$

7.3 Utilità

- erosione: rimuove la struttura di certe forme e dimensione in base all'SE scelto
- dilatazione: riempe certe forme e dimensione in base all'SE scelto
- Cosa di cerca
  - rimuove le strutture / riempe i buchi
  - senza influenzare le parti rimanenti
- Solution
  - combina l'erosione e la dilatazione
  - utilizza lo stesso SE

7.3.1 Opening e Closing: proprietà

Le due operazioni morfologiche definite rispettivamente come apertura e chiusura vengono utilizzate al seguente scopo:

- opening: serve per rendere più omogenei i contorni di un oggetto, elimina per piccole interruzioni e le protuberanze sottili
- closing: anch'esso rende più omogenee le sezioni del contorno, ma al contrario dell'opening, generalmente fonde insieme le interruzioni sottili e i segmenti stretti e lunghi, elimina i piccoli buchi e riempe vuoti nel contorno
- l'opening e closing sono duali l'uno rispetto all'altro rispetto all'insieme complementare e alla riflessione:

$$(A \bullet B)^c = A^C \circ \hat{B} \quad \text{e}$$

$$(A \circ B)^c = A^C \bullet \hat{B}$$

7.3.2 Opening

L'opening (apertura) di un insieme A attraverso l'SE B, denotato come $A \circ B$ è definito come:

$$A \circ B = (A \ominus B) \oplus B$$

Quindi l'opening di A attraverso B si ottiene eseguendo in sequenza l'erosione di A attraverso B e la dilatazione del risultato attraverso B

L'interpretazione geometrica si basa sul considerare l'immagine SE B come una palla rotolante (appiattita).

Il **contorno** di $A \circ B$ viene quindi stabilito dai punti in B che raggiungono il punto più lontano nel bordo di A mentre B gira all'interno di questo confine.

l'opening di A attraverso B, da un punto di vista della teoria degli insieme, perciò può essere anche essere ottenuto dall'unione di tutte le traslazioni di B che si adattano in A. Cioè, l'apertura può essere espressa come un processo di fitting (adattamento) tale che:

$$A \circ B = \bigcup \{(B)_z | (B)_z \subseteq A\}$$

L'opening soddisfa le seguenti proprietà:

- $A \circ B$ è un sottoinsieme(sottoimmagine) di A
- if C è un sottoinsieme di D allora $C \circ B$ è un sottoinsieme di $D \circ B$
- $(A \circ B) \circ B = A \circ B$

7.3.3 Closing

Il closing (chiusura) di un insieme A attraverso l'SE B, denotato come $A \bullet B$, è definito come:

$$A \bullet B = (A \oplus B) \ominus B$$

Quindi il closing di A attraverso B è la dilatazione di A attraverso B seguita dall'erosione del risultato attraverso B.

L'interpretazione geometrica del closing è simile a quella dell'opening, eccetto per il fatto che si fa scorrere B sul lato esterno del contorno.

Geometricamente un punto w è un elemento di $a \bullet B$ se e solo se $(B)_z \cap A \neq \varnothing$ per ogni traslazione $((B)_z$ che contiene w.

Il closing soddisfa le seguenti proprietà:

- A è un sottoinsieme (sottoimmagine) di $A \bullet B$
- se C è un sottoinsieme di D, allora $C \bullet B$ è un sottoinsieme di $D \bullet B$
- $()A \bullet B \bullet B = A \bullet B$

7.3.4 Le proprietà Opening e Closing

L'opening e closing sono duali l'uno rispetto all'altro rispetto all'insieme complementare e alla riflessione:

$$(A \bullet B)^c = A^C \circ \hat{B} \quad \text{e} \quad (A \circ B)^c = A^C \bullet \hat{B}$$

L'opening soddisfa le seguenti proprietà:

- $A \circ B$ è un sottoinsieme(sottoimmagine) di A
- if C è un sottoinsieme di D allora $C \circ B$ è un sottoinsieme di $D \circ B$
- $(A \circ B) \circ B = A \circ B$

Il closing soddisfa le seguenti proprietà:

- A è un sottoinsieme (sottoimmagine) di $A \bullet B$
- se C è un sottoinsieme di D, allora $C \bullet B$ è un sottoinsieme di $D \bullet B$
- $()A \bullet B \bullet B = A \bullet B$

7.4 Trasformazione Hit-or-Miss

La trasformazione hit-or-miss (colpisci o manca) è una trasformazione morfologica fondamentale per l'individuazione della forma. Spesso è usato per individuare specifiche configurazioni di pixel (i pixel isolati dello sfondo in primo piano o pixel sono punti finali della linea di segmenti).

La trasformazione Hit-or-Miss di A attraverso B è denotata come $A * B$, dove B è un accoppiamento SE $B = (B_1, B_2)$ rispetto che un singolo elemento.

La trasformazione Hit-or-Miss è definita in funzione di questi due SE:

$$A * B = (A \ominus B_1) \cap (A^c \ominus B_2)$$

7.5 Algoritmi di Morfologia basica

- estrazione di contorno
- riempimento di buchi
- estrazione di componenti connessi
- superficie convesso
- thinning (assottigliamento)
- thickening (ispessimento)
- scheletri
- pruning (riduzione)
- ricostruzione morfologica

7.5.1 Estrazione di contorni

Il bordo di un insieme A indicato come $\beta(A)$, può essere ottenuto inizialmente erodendo A attraverso B e successivamente applicando la differenza insiemistica tra A e la sua erosione:

$$\beta(A) = A - (A \ominus B)$$

dove B è un SE adatto (solitamente 3x3, anche se la dimensione può variare portando a bordi di diversi spessori)

7.5.2 Riempimento di buchi

Un buco è definito come uno sfondo di regione circondato da un bordo connesso di pixel in primo piano.

A denota un insieme di elementi che sono un confine 8-connessi dove ogni confine racchiude un buco.

Fornendo un punto in ogni buco, l'obiettivo è riempire tutti questi buchi con un 1s.

Viene formato un array X_0 di 0s (la stessa dimensione dell'array che contiene A), eccetto nella locazione in X_0 corrispondente al punto fornito in ogni buco, che abbiamo impostato a 1.

La seguente procedura riempe tutti i buchi con 1s:

- $$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \ldots$$

  dove B è il SE simmetrico visualizzato nella figura
- l'algoritmo termina al passo k se $X_k = X_{k-1}$
- l'insieme X_k contiene tutti i buchi riempiti
- l'insieme unione di X_k e A contiene tutti i buchi riempiti e i loro confini

7.6 Estrazione di componenti connessi

L'estrazione di componenti connessi da un'immagine binaria è fondamentale in molte applicazioni basati sull'analisi delle immagini.

Si prenda l'insieme A che contiene uno o più componenti connessi e si formi un array X_0 (della stessa dimensione di A) i cui elementi sono $0s$ (i valori di sfondo), eccetto alla locazione che corrisponde al punto in ogni componente connessa in A, a cui daremo il valore 1 (valore in primo piano).

L'obiettivo è incominciare con X_0 e trovare tutti i componenti connessi.

La procedura interattiva permette di raggiungere l'obiettivo:

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3, \ldots$$

dove B è un opportuno SE

La procedura termina quando $X_k = X_{k-1}$ con X_k contenente tutti i componenti connessi dell'immagine di ingresso.

7.7 superficie convesso

Un insieme A è detto convesso se la linea retta che unisce due punti qualsiasi di A si trova interamente all'interno di A. Il superficie convesso H di un insieme arbitrario S è il più piccolo insieme convesso contenente S.

L'insieme differenza $H - S$ è chiamato convex deficiency di S.

Qui di seguito viene presentato un semplice algoritmo morfologico per ottenere un superficie connesso $C(A)$ di un insieme A.

Dato $B^i = i = 1, 2, 3, 4$ rappresentando i quattro SE in figura. La procedura consiste nell'implementazione dell'equazione:

$$X_k^i = (X_{k-1}^i * B^i) \cup A \quad i = 1, 2, 3, 4 \quad k = 1, 2, 3 \ldots$$

con $X_O^i = A$

Quando la procedura converge (per esempio quando $X_k^i = X_{k-1}^i$) si ha $D^i = X_k^i$

Il superficie convesso di A è:

$$C(A) = \bigcup_{i=1}^{4} D^i$$

Il metodo consiste nell'applicare iterativamente la trasformata Hit-or-Miss su A con B^1. Quando non ci sono più cambiamenti, si effettua l'unione con A e si chiama il risultato D^1

La procedura è ripetuta con B^2 (applicata su A) sino a quando non ci hanno cambiamenti.

L'unione dei quattro D risultanti costituisce il superficie convesso di A

7.8 Thinning (assottigliamento)

Il thinning di un insieme A attraverso un SE B è denotato come $A \otimes B$ e può essere definito nei termini di una trasformata della trasformata Hit-or-Miss:

$$A \otimes B = A - (A * B) = A \cap (A * B)$$

Una espressione più utile per il thinning simmetrico di A è basato sula sequenza di SE:

$$\{B = \{B^1, B^2, B^3, \ldots, B^n\}\}$$

dove B^i è una versione ruotata di B^{i-1}

Si può definire il thinning come una sequenza di SE come:

$$A \otimes \{B\} = ((\ldots((A \otimes B^1)B^2)\ldots) \otimes B^n)$$

Il processo assottiglia A con una passata di B^1 quindi assottiglia il risultato nuovamente con una passata di B^2 e così via, sino a quando A è assottigliato con una passata di B^n. Questo processo è ripetuto sino a quando non si registrano ulteriori cambiamenti.

7.9 Thickening (ingrossamento)

Si può definire come il morfologico duale rispetto al thinning ed è definito dall'espressione:

$$A \odot B = A \cup (A * B)$$

dove B è un elemento strutturale utilizzato per il thickening.

Il thickening si può definire come un'operazione sequenzale:

$$A \odot \{B\} = ((\ldots((A \odot B^1) \odot B^2)\ldots) \odot B^n)$$

Gli SE utilizzati per il thickening hanno la stessa forma di quelle usate per il thinning, ma con tutti gli 1s e gli 0s scambiati.

Per effettuare l'ingrossamento di un insieme A, si può formare $C = A^c$, assottigliare C e quindi formare C^c.

L'operazione dipende dalla natura di A e può portare ad avere punti sconnessi. Essa inoltree è seguita da un postprocessing per rimuovere i punti disconnessi.

7.10 Scheletro

Se z è un punto di $S(A)$ (lo scheletro) e $(D)_z$ è il più grande disco centrato in z e contenuto in A, allora non esiste un disco più grande (non necessariamente centrato in z) contenente $(D)_z$ (il disco massimo) e incluso in A.

Il disco massimo $(D)_z$ tocca il confine di A in due o più posti.

Lo scheletro di A può essere espresso in termini di erosione e opening. Esso può essere visto (Serra 1982) come:

$$S(A) = \bigcup_{k=0}^{k} S_k(A)$$

con

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B$$

dove B è un elemento strutturante e $(A \ominus kB)$ indica k successive erosioni di A:

$$(A \ominus kB) = ((\ldots((A \ominus B) \ominus B) \ominus \ldots) \ominus B)$$

k volte e K è l'ultimo passo iterattivo prima che A si eroda in un insieme vuoto. In altre parole;

$$K = max\{k|(A \ominus kB) \neq \varnothing\}$$

Questa formulazione afferma che $S(A)$ può essere ottenuto come unione di sottoinsiemi di scheletri $S_k(A)$

Questo può essere visto come A che può essere ricostruito da questi sottoinsiemi usando l'equazione:

$$A = \bigcup_{k=0}^{K} (S_k(A) \oplus kB)$$

dove $(S_k(A) \oplus kB)$ denota le k successive dilazioni di $S_k(A)$:

$$(S_k(A) \oplus kB) = ((\dots((S_k(A) \oplus B) \oplus B) \oplus \dots) \oplus B))$$

Lo scheletro trovato può essere più spesso del necessario e può non essere connesso.

7.11 Prunning (riduzione)

Metodi di riduzione sono un complemento essenziale degli algoritmi di thinning e di scheletrizzazione perché queste procedura tendono a lasciare un componenti parassitarie che necessitano di essere ripulite.

- lo scopo nel riconoscimento automatico dei caratteri è quello di analizzare forma dello scheletro di ogni carattere
- gli scheletri sono spesso caratterizzati da spurs, causati dall'erosione
- i componenti parassitari devono essere eliminati senza eccedere uno determinato numero di pixel

7.11.1 Metodo operativo

La soluzione è basata sulla soppressione del ramo parassitario eliminando via via i suoi punti terminali (nel nostro caso si tratta di rami con 3 pixel o meno)

- si effettua il thinning di un insieme A con una sequenza di SE disegnati per individuare i soli punti finali, sino a trovare il risultato desiderato
- si prende:
  $$X_1 = A \otimes \{B\}$$
  dove $\{B\}$ denota la sequenza in figura, che consiste in due differenti strutture, ognuna delle quali è ruotata di 90°
- applicando tre volte la precedente equazione ad A si ottiene l'insieme X_1

- il passo successivo è ripristinare i caratteri alla loro forma originale con i rami parassitari rimossi. Per fare questo, inizialmente si forma un insieme X_2 contenente tutti i punti finali in X_1:

$$X_2 = \bigcup_{k=1}^{8} (X_1 * B^k)$$

  dove B^k sono gli stessi rilevatori di punti finali

- successivamente si effettua per tre volte la dilatazione dei punti final, usando un insieme A come delimitatore:

$$X_3 = (X_2 \oplus H) \cap A$$

  dove H è $3x3$ SE di 1s e l'intersezione con A è applicata dopo ogni passo

- l'unione di X_1 e X_3 permette di ottenere il risultato desiderato $X_4 = X_1 \cup X_3$

7.12 Ricostruzione morfologica

La ricostruzione morfologica coinvolge due immagini e un elemento strutturante.

- un'immagine viene definita **marker** (marcatore) e contiene i punti iniziali della trasformazione
- l'altra immagine è la **mask** (maschera) che vincola la trasformazione
- l'elemento strutturante è usato per definire la connettività

7.12.1 Dilatazione Geodesic

La dilatazione geodesic di dimensione 1 di una immagine marcatore rispetto alla maschera, indicata da $D_G^{(1)}(F)$ è definita come:

$$D_G^{(1)}(F) = (F \oplus B) \cap G$$

dove:

- F indica l'immagine marker
- G è l'immagine mask
- immagine binaria e $F \subseteq G$

La dilatazione geodesic di dimensione n di F rispetto a G è definita come:

$$D_G^{(n)}(F) = D_G^{(1)}[D_G^{(n-1)}(F)]$$

dove $D_G^{(0)}(F) = F$

L'intersezione, effettuata ad ogni passaggio, garantisce che la maschera G limiterà la crescita (dilatazione) del marker F.

7.12.2 Erosione Geodesic

L'erosione geodesic di dimensione 1 di una immagine marcatore F rispetto alla maschera G, denotata da $E_G^(1)(F)$, è definita come:

$$E_G^(1)(F) = (F \ominus B) \cup G$$

L'erosione geodesic di dimensione n di F rispetto a G è definita come:

$$E_G^{(n)}(F) = E_G^{(1)}[E_G^{n-1}(F)]$$

dove $E_G^{(0)}(F) = (F)$

L'unione, effettuata ad ogni passo, garantisce che l'erosione geodesic di una immagine rimanga uguale o superiore a quella della maschera dell'immagine.

L'erosione e la dilatazione geodesic sono duali rispetto al complemento dell'insieme

7.12.3 Ricostruzione morfologica attraverso la dilatazione e l'erosione

- **ricostruzione morfologica attraverso la dilatazione** di una maschera di immagine G da una immagine marker F, indicata con $R_G^{(D)}(F)$, è definita come una dilatazione geodesic di F rispetto a G, iterato sino a quando si ottiene la stabilità:
$$R_G^{(D)}(F) = D_G^{(k)}(F)$$
con k tale che $D_G^k(F) = D_G^{k+1}(F)$

- **ricostruzione morfologica attraverso l'erosione** di una maschera di immagine G da una immagine marker F, indicata con$R_G^{(E)}(F)$, è definita come l'erosione geodesic di F rispetto a G, iterata sino a quando si ottiene la stabilità:
$$R_G^(E)(F) = E_G^{(k)}(F)$$
con k tale che $E_G^{(k)(F)}E_G^{k+1}(F)$

7.13 Applicazioni

7.13.1 Opening e closing attraverso la ricostruzione

L'opening attraverso la ricostruzione ripristina esattamente le forme degli oggetti che rimangono dopo l'erosione.

L'opening attraverso la ricostruzione di dimensione n di una immagine F è definita come la ricostruzione attraverso la dilatazione di F a partire dall'erosione di dimensione n di F:

$$O_R^{(n)}(F) = R_F^D[(F \ominus nB)]$$

dove $(F \ominus nB)$ indica le n erosioni di F attraverso B F è usata come maschera
Una espressione simile è scritta per il closing attraverso la ricostruzione

7.13.2 Border cleaning (pulizia dei bordi)

Si tratta di una procedura basata su una ricostruzione morfologica. E' un algoritmo che rimuove gli oggetti che toccano il bordo, che risulta in diversi casi:

- per effettuare lo screen di immagini in cui completare gli oggetti rimanenti per successive elaborazioni
- come segnale di oggetti parziali presente nel campo di vista

L'immagine originale I è usato come maschera, mentre l'immagine marcatore è la seguente:

$$F(x,y) = \begin{cases} I(x,y) \text{ se } (x,y) \text{ è sul bordo di } I \\ 0 \text{ negli altri casi} \end{cases}$$

L'algoritmo prima calcola la ricostruzione morfologica $R_i^{(D)(F)}$ (semplicemente estraendo gli oggetti che toccano il bordo) e successivamente calcola la differenza:

$$X = I - R_I^{(D)}(F)$$

Questo permette di ottenere un'immagine X senza oggetti che toccano il bordo.

Capitolo 8

Morfologia matematica scala di grigi

Nella morfologia scala di grigi si utilizzano funzioni discrete definite in Z^2 nella forma:

- $f(x, y)$ che indica una immagine in scala di grigi
- $b(x, y)$ che indica un SE, utilizzata come sonda per esaminare un'immagine e le sue specifiche proprietà
  - gli SE nella morfologia scala di grigi può essere $flat$ oppure $nonflat$
  - l'origine deve essere definita e solitamente risulta essere simmetrica con l'origine situata al centro
  - la riflessione di SE è indicata come:

$$\hat{b}(x, y) = b(-x, -y)$$

8.1 Erosione e dilatazione tramite un SE flat

- L'**erosione** di f tramite un SE flat b in ogni punto (x, y) è definito come il **minimo** valore di uan immagine in una regione coincidente con b quando l'origine di b si trova in (x, y):

$$[f \ominus b](x, y) = \min_{(s,t)\in b} \{f(x + s, y + t)\}$$

- la **dilazione** di f tramite un SE flat b in ogni punto (x, y) è definito come il **massimo** valore di una immagine nella finestra delineata da $\hat{b}$ quando l'origine di $\hat{b}$ si trova in (x, y):

$$[f \oplus b](x, y) = \max_{(s,t)\in b} \{f(x - s, y - t)\}$$

8.2 Erosione e dilatazione tramite un SE nonflat

- l'**erosione** di f tramite un SE nonflat b_N è definito come:

$$[f \ominus b_N](x,y) = \min_{(s,t)\in b_N} \{f(x+s, y+t) - b_N(s,t)\}$$

- la **dilatazione** di f tramite un SE nonflat b_N è definito come:

$$[f \oplus b_N](x,y) = \max_{(s,t)\in b_N} \{f(x-s, y-t) + b_N(s,t)\}$$

- quando tutti gli elementi di b_N sono costanti (per esempio l'SE è flat) la definizione si riduce alla precedente all'interno di una costante scalare uguale all'ampiezza di SE
- gli SE scala di grigi sono raramente utilizzati
- l'erosione e la dilatazione sono duali rispetto alla funzione di complemento e riflessione

8.3 Opening e Closing

- l'opening è usato per rimuovere piccoli dettagli luminosi, mentre i dettagli più grandi vengono trascurati.
- l'opening di una immagine f attraverso un SE b è denotato come $f \circ b$:

$$f \circ b = (f \ominus b) \oplus b$$

- il closing è utilizzato per attenuare le caratteristiche scure con una attenuazione graduale che dipende dalle dimensioni delle caratteristiche rispetto a SE:
- il closing di una immagine f tramite un SE b è indicato come $f \bullet b = (f \oplus b) \ominus b$

L'opening e il closing sono duali rispetto al complemento e alla SE riflessione:

$$(f \circ b)^c = f^c \bullet \hat{b} \quad \text{e} \quad (f \bullet b)^c = f^c \circ \hat{b}$$

8.3.1 Opening e Closing: proprietà

- l'opening scala di grigi soddisfa le seguenti proprietà:
  - $f \circ b \downarrow f$
  - se $f_1 \downarrow f_2$ allora $(f_1 \circ b) \downarrow (f_2 \circ b)$
  - $(f \circ b) \circ b = f \circ b$
- il closing scala di grigi soddisfa le seguenti proprietà:
  - $f \downarrow f \bullet b$
  - se $f_1 \downarrow f_2$ allora $(f_1 \bullet b) \downarrow (f_2 \bullet b)$
  - $(f \bullet b) \bullet b = f \bullet b$

NOTA: la notazione $e \downarrow r$ è usata per indicare che il dominio di e è un sottoinsieme del dominio di r e che $e(x,y) \leq r(x,y)$ per ogni (x,y) ne dominio di e

8.4 Algoritmi basati sulla scala di grigi morfologica

8.4.1 Smoothing morfologico

L'opening elimina piccoli dettagli luminosi, piccoli rispetto ad uno specifico SE. Il closing invece elimina i dettagli scuri. Essi sono usati in combinazione come filtri morfologici per lo smoothing dell'immagine e per rimuovere il rispettivo rumore.

In maniera alternativa, è possibile ottenere una sequenza di filtering nel seguente modo: La sequenza di opening-closing incomincia con l'immagine originale, ma i singoli passi dell'opening e del closing vengono effettuati sul risultato del passo precedente.

8.4.2 Gradiente morfologico

La dilatazione e l'erosione possono essere usate in combinazione con la "sottrazione dell'immagine" per ottenere il gradiente morfologico g di una immagine:

$$g = (f \oplus b) - (f \ominus b)$$

La dilatazione addensa le regioni in una immagine e l'erosione le restringe. La loro differenza enfatizza i confini tra le regioni.

8.4.3 Trasformazioni Top-hat e Bottom-hat

- la trasformazione top-hat di una immagine in toni di grigio f è definita come "f minus its opening":

$$T_hat(f) = f - (f \circ b)$$

- la trasformazione bottom-hat di una immagine in toni di grigio f è definita come il closing di f minus f:

$$B_hat(f) = (f \bullet b) - f$$

Queste trasformazioni sono usate per rimuovere gli oggetti dall'immagine usando un SE nelle operazioni di opening o closing che non si adatta alle immagini da rimuovere.

L'operazione differenza produce un'immagine in cui rimangono solo i componenti rimossi:

- la trasformazione top-hat è usata per gli oggetti luminosi su uno sfondo scuro
- la trasformazione bottom-hat è usata per il contrario.

8.4.4 Granulometria

E' un campo che si occupa di determinare la dimensione della distribuzione delle particelle in una immagine.

La morfologia può essere usata per stimare la dimensione della distribuzione di particelle nella distribuzione senza identificare e misurare ogni particella nell'immagine.

Con particelle aventi forma regolare e che sono più luminose rispetto allo sfondo, il metodo consiste nell'applicare l'opening con SE di dimensioni maggiori.

Le operazioni di opening con una certa dimensione hanno un effetto maggiore nelle regioni dell'immagine di input che contiene particelle di dimensioni simili.

Per ogni openint, la somma dei valori dei pixel nell'opening viene calcolato. Questa somma è chiamata **surface area (superficie)** e diminuisce in funzione all'incremento della dimensione di SE.

Questa procedura produce una matrice 1-D di tali numeri, con ogni elemento nella matrice che diventa uguale alla somma dei pixel nell'opening per la dimensione SE corrispondente a quella posizione nella matrice.

Per enfatizzare i cambiamenti tra successivi opening, si calcola la defferenza tra gli elementi adiacenti della matrice 1-D.

Le visualizzare i risultati, la differenza viene tracciata (plotted). I picchi nel grafico indicano la dimensione predominante della distribuzione di particelle nell'immagine.

8.4.5 Segmentazione Texturale

L'immagine rumorosa qui rappresentata ha due regioni strutturali:

- una regione composta da chiazza larghe sulla destra
- una regione composta da chiazze piccole sulla sinistra

8.4.6 Ricostruzione morfologica su scala di grigi

Sia f il marker e g la maschera dell'immagine. Entrambi sono immagini in tono di grigio della stessa dimensione e $f \leq g$

La **dilatazione geodetica di dimensione 1** di f rispetto a g è definita come:

$$D_g^{(n)}(f) = D_g^{(1)}[D_g^{(n-1)}(f)]$$

dove:

$$D_g^{(0)}(f) = f$$

L'**erosione geodetica di dimensione 1** di f rispetto a g è definita come:

$$E_g^{(1)}(f) = (f \ominus b) \vee g$$

dove $\vee$ indica " the point-wise maximum operator"

L'**erosione geodetica di dimensione n** di f rispetto a g è definita come:

$$E_g^{(n)}(f) = E_g^{(1)}[E_g^{(n-1)}(f)]$$

dove:

$$E_g^{(0)}(f) = f$$

La ricostruzione morfologica su scala di grigi tramite la dilatazione di maschera g di una immagine f è definita come la dilatazione geodetica di f rispetto a g, iterata sino a che viene raggiunta la stabilità:

$$R_g^{(D)}(f) = R_g^{(k)}(f)$$

con k tale che $D_g^{(k)}(f) = D_g^{(k+1)}(f)$

La ricostruzione morfologica su scala di grigi tramite l'erosione di maschera g di una immagine f è definita come l'erosione geodetica di f rispetto a g, iterata sino a che viene raggiunta la stabilità:

$$R_g^{(E)}(f) = E_g^{(k)}(f)$$

con k tale che $E_g^{(k)}(f) = E_g^{(k+1)}(f)$

L'opening tramite ricostruzione di dimensione n di una immagine f è definita come la ricostruzione tramite dilatazione di f partendo dall'erosione di dimensione n di f:

$$O_R^n(f) = R_f^{(D)}[()f \ominus nb)]$$

dove $(f \ominus nb)$ denota n erosioni di f tramite b.

Il closing tramite ricostruzione di dimensione n di una immagine f è definita come la ricostruzione tramite erosione di f partente dalla dilatazione di dimensione n di f:

$$C_R^{(n)}(f) = R_f^{(E)}[(f \oplus nb)]$$

dove $f \oplus nb)$ denota n dilatazioni di f tramite b.

Capitolo 9

Segmentazione Watershed

9.1 Segmentazione attraverso la morfologia

La segmentazione sinora vista può essere ottenuta tramite:

- rilevamento degli edge
- thresholding (sogliatura)
- region growing (accrescimento delle regioni)/split-and-merge (dividi e combina)

L'approccio del Watershed (spartiacque) morfologico produce dei risultati di segmentazione stabili, compresi anche i confini di segmentazione connessi.

Essa fornisce un quadro semplice per incorporare vincoli di conoscenza basati nel processo di segmentazione.

9.2 Watershed

Il concetto dietro alla sogliatura watershed è basato sulla visualizzazione di una immagine in tre dimensioni (due coordinate spaziali e l'intensità).

La segmentazione watershed è utilizzata principalmente per l'estrazione dello sfondo degli oggetti pressoché uniformi (bloblike). Le regioni caratterizzate da piccole variazioni di intensità hanno piccoli valori di gradiente. Per questo motivo si preferisce applicare tale segmentazione al gradiente di una immagine, anziché all'immagine stessa. Nell'esempio che vedremo in seguito, i minimi locali si correlano bene con valori piccoli del gradiente che corrispondono agli oggetti di interesse.

In una topografia consideriamo tre tipi di punti:

1. punti appartenenti al minimo locale
2. punti in cui una goccia d'acqua scorrerebbe verso uno di tali minimi
3. punti in cui l'acqua potrebbe egualmente cadere in più di un punto di minimo

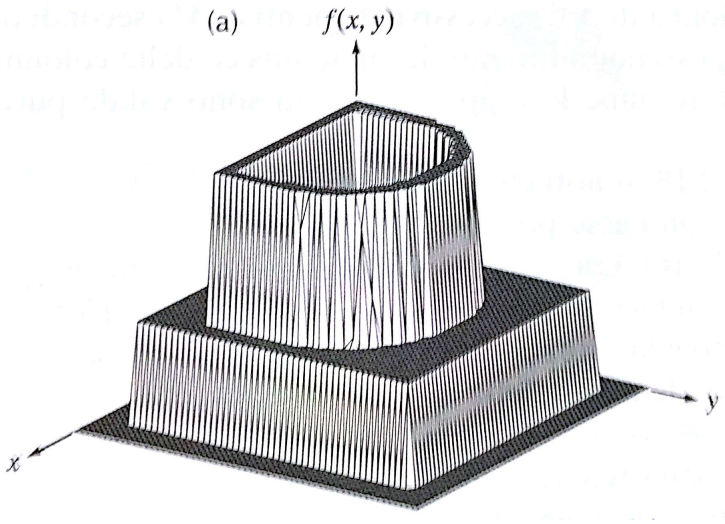

Figura 9.1: File `mission.eps`

Per un particolare minimo locale, l'insieme dei punti che soddisfano la condizione 2) è chiamato bacino di raccolta o watershed di questo minimo.

I punti che soddisfano la condizione 3) formano le creste sulla superficie topografica e sono definite linee di divisione o linee di watershed.

Lo scopo dell'algoritmo di segmentazione è quello di trovare le linee di watershed.

Supponiamo che ogni minimo locale venga perforato e che l'intera topografia si riempita dal basso lasciando che l'acqua risalga uniformemente attraverso questi fori.

Quando l'acqua dei differenti bacini rischia di debordare, viene costruita una diga al fine di prevenire tale merging.

L'acqua può raggiungere anche livelli in cui solo le cime di tali dighe risultano visibili. Questi contorni della diga corrispondono alle linee di divisione watershed. Si tratta di contorni connessi estratti dall'algoritmo di segmentazione watershed.

La Fig.10.54 mostra una visualizzazione topografica dell'applicazione dell'algoritmo watershed dove bisogna considerare che:

- l'altezza delle "montagne" è proporzionale ai valori di intensità dell'immagine di input
- per prevenire la crescita del livello dell'acqua oltre i contorni dell'immagine, si chiude il perimetro dell'intera topografia (immagine) con delle dighe che hanno un'altezza maggiore delle montagne, il cui valore viene determinato dal più alto valore possibile di intensità nell'immagine di input
- si supponga di perforare ogni minimo (le aree in nero in Fig.10.54b) e che l'intera topografia sia sommersa lasciando che il livello dell'acqua risalga da questi buschi a un tasso uniforme

- la Fig.10.54c mostra un primo livello di inondazione in cui l'acqua (rappresentata in grigio chiaro) ricopre solo le aree che corrispondono allo sfondo nero
- successivamente, l'acqua risalendo arriva sino ad arrivare rispettivamente al primo e secondo bacino di raccolta (vedi Fig.10.54d e Fig.10.54e)
- se l'acqua continuasse a salire, questa potrebbe straripare da un bacino all'altro, come accade in Fig.10.54f, dove se non ci fosse una diga (una serie di singoli pixel) l'acqua straripererebbe dal bacino di sinistra a quello di destra
- in Fig.10.54g l'acqua continua a salire e sono messe in risalto le dighe tra i due bacini e quella in alto del bacino di destra; quest'ultima evita che l'acqua del bacino possa confluire con le aree corrispondenti allo sfondo
- questo processo continua sino a raggiungere il massimo livello di allagamento corrispondente al valore più alto di intensità dell'immagine
- la diga finale corrisponde alle linee di watershed, che corrisponde alla segmentazione desiderata come in Fig.10.54h dove si può notare un path di 1 pixel (in nero) sovrapposto all'immagine originale
- le linee di watershed formano dei path connessi e di conseguenza dei contorni continui tra le regioni

9.3 L'uso dei marcatori

L'applicazione diretta dell'algoritmo di segmentazione porta a sovrasegmentazione (oversegmentation) dovuta al rumore o ad altre irregolarità locali del gradiente e questo problema può rendere l'algoritmo di scarsa utilità pratica. Si può migliorare la situazione incorporando una fase di pre-processing (pre-elaborazione) facendo uso per esempio dei marker.

Un marker è una componente connessa appartenente ad una immagine. Si possono avere:

- marker interni associati agli oggetti della figura. Una definizione più precisa è:
  1. una marker interno è una regione circondata da punti di altitudine elevata
  2. tale che i punti nella regione formino una componente connessa e
  3. in cui tutti i punti della componente connessa abbiamo la stessa intensità
- marker esterni associati allo sfondo

9.3.1 Selezione dei marker

La tipica procedura per la selezione dei marker consiste di due fasi:

- la pre-elaborazione

- la determinazione di un insieme di criteri che devono essere soddisfatti dai marker.

La selezione del marker può prevedere:

- procedure molto semplici basate sui valori di intensità e sulla connettività
- descrizioni più complesse) (dimensione, forma, posizione, distanze relative, tessitura, ecc)

L'uso dei marker aggiunge una conoscenza a priori nel processo di segmentazione analoga per certi versi al alcune peculiarità tipiche del sistema visivo umano.

Se consideriamo l'immagine 10.57 noteremo che parte dei problemi sarà dovuta all'eccessiva segmentazione, ovvero di minimi potenziali. A causa della dimensione, molti di questi minimi si riferiscono a dettagli di poco conto. Un metodo efficace per ridurre questo problema, è quello di applicare un filtro di smoothing.

Dopo aver applicato il filtro di smoothing (vedi Fig.10.58a) i marker interni risultanti sono mostrati in grigio chiaro.

In seguito viene applicato l'algoritmo watershed all'immagine con la condizione che *solo* i marker interni siano possibili minimi regionali.

La Fig.10.58a mostra le linee di watershed risultanti e sono definite come marker esterni. Si noti che i punti lungo le linee di watershed passano lungo i punti alti tra i marker vicini.

I marker esterni dividono efficacemente l'immagine in regione, e ognuna di queste contiene un solo marker interno e parte dello sfondo.

A questo punto si possono seguire varie strade:

- applicare un qualsiasi algoritmo di segmentazione tra quelli visti in precedenza
- applicare l'algoritmo di segmentazione ad ogni singola regione. In questo modo si prende in considerazione solo il gradiente dell'immagine e successivamente si applica l'algoritmo solo alle singole watershed che contengono i marker in quella regione.

Il risultato finale di buon livello è quello mostrato in Fig.10.58b.

Capitolo 10

Segmentazione Watershed in Matlab

La trasformata watersher è calcolata dalla funzione *watershed*, la cui sintassi è la seguente:

$$L = watershed(f)$$

dove L è una etichetta di matrice con interi positivi che corrispondo ai bacini e con i valori zero usati per indicare le creste watershed.

Uno strumento comunemente utilizzato con la trasformata watershed per la segmentazione è la **trasformata della distanza**.

La trasformate della distanza di una immagine binaria è la distanza da ogni pixel dal pixel più vicino di valore diverso da zero.

La trasformata della distanza può essere calcolata usando la funzione *bwdist*, la cui sintassi è:

$$D = bwdist(f)$$

10.1 Segmentazione di una immagine binaria

```
f = imread('binary-dowel-image.tif');
figure, imshow(f), title('Binary image');

fc = ~f ;
figure, imshow(fc), title('Complement image');

D = bwdist(fc);
figure, imshow(imadjust(uint16(D))), title('Distance transform');

L = watershed(-D);
w = L==0;
figure, imshow(w)
title('Watershed lines of the negative of the distance transform');

g2 = f & ~w ;
figure, imshow(g2)
title('Watershed lines superimposed in black over original binary
    image');
```

10.2 Segmentazione di una immagine in toni di grigio

```
f = imread('small-blobs.tif');
figure, imshow(f), title('Original image');

h = fspecial('sobel');
fd = double(f);
g = sqrt(imfilter(fd, h, 'replicate').^2 + imfilter(fd, h, '
    replicate').^2);
figure, imshow(imadjust(uint16(g))), title('Gradient image');

L = watershed(g);
wr = L==0;
figure, imshow(wr), title('Watershed segmentation');

g2 = imclose(imopen(g, ones(3,3)), ones(3,3));
L2 = watershed(g2);
wr2 = L2==0;
f2=f;
f2(wr2)=255;
figure, imshow(f2)
title('Watershed segmentation on smoothed image');
```

10.3 Segmentazione usando i marcatori

Per controllare la sovra-segmentazione si utilizzano i marker (marcatori) interni ed esterni.

La funzione IPT *imregionalmin* calcola la posizione di tutti i minimi locali in una immagine. La sintassi è la seguente:

$$rm = imregionalmin(f)$$

dove f è l'immagine in toni di grigio e rm è una immagine binaria in cui i pixel in primo piano marcano le posizioni del minimo locale; questi ultimi possono rappresentare dettagli irrilevanti.

Per eliminare i minimi estranei si utilizza la funzione IPT *imextendedmin* che calcola l'insieme di punti "bassi" nell'immagine that are deeper than their immediatesurroundings (by a certain height threshold).

Fornendo sia i marcatori interni che esterni, è possibile modificare l'immagine del gradiente usando una procedura chiamata **minima imposition**. Questa tecnica modifica una immagine in toni di grigio in modo che i minimi locali si verificano solo nelle posizioni marcate. Altri calori dei pixel sono spinti verso l'alto per rimuovere tutti gli altri minimi locali

Capitolo 11

Image processing con Matlab

11.1 Ridimensionamento

comando: imresize
possibili domande:

- Leggere un'immagine e rimpicciolirla o farne lo zoom di un valore intero
- Leggere un'immagine e rimpicciolirla o farne lo zoom di un valore decimale con interpolazione lineare

11.2 Trasformazioni geometriche

In Matlab è possibile effettuare una trasformazione geometrica affine specificando la matrice di trasformazione T attraverso il comando *maketform*. Per effettuare la trasformazione si usa il comando *imtransform*.
comando: maketform, imtransform, imrotate, impixelinfo (pixval),
possibili domande:

- Effettuare la rotazione di un'immagine qualsiasi
- Confrontare il risultato con quello ottenuto mediante la funzione *imrotate*
- Leggere l'immagine di 'lena' e realizzare l'ingrandimento di una zona dell'immagine usando la matrice di trasformazione T la sezione da ingrandire è intorno all'occhio di lena. Per individuare la sezione e, quindi, avere informazioni sulla posizione dei pixel potete usare il comando *impixelinfo* o *pixval* (in base alla versione di Matlab più o meno recente).
- Fare degli esperimenti modificando il tipo di interpolazione e notate l'effetto di blocchettatura causa
- La combinazione di diverse trasformazioni affini è ancora una trasformazione affine, che può essere ottenuta tramite il prodotto (matriciale) delle matrici che le definiscono.

- Scrivere una funzione dal prototipo `function g=rot_dist(f, alfa, c)` per realizzare prima una rotazione e poi una distorsione verticale.
- Creare l'immagine di ingresso usando il seguente comando `f = checkerboard(40);` in modo da generare una scacchiera su cui le modifiche risultano essere più facilmente visibili.

11.3 Rumore

- Aggiungere del rumore gaussiano bianco ad un immagine f con il comando `noisy = f + n con n = d*randn(size(f))` dove **d è la deviazione standard del rumore**.
- Rimuovere il rumore dall'immagine con i filtri a media mobile (al variare della dimensione della finestra).
- Valutare l'efficacia del filtraggio sia visivamente sia calcolando l'errore quadratico medio tra f e l'immagine "ripulita" .
- L'errore quadratico medio rappresenta una misura quantitativa per stabilire quanto l'immagine elaborata sia simile all'originale.
- L'MSE (Mean Squared Error) tra due immagini si definisce come:

$$MSE = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} |f(m,n) - g(m,n)|^2$$

11.4 Smoothing con thresholding

- Consideriamo l'immagine 'telescopio.jpg', proveniente dal telescopio Hubble, in orbita intorno alla terra. Rilevare gli oggetti grandi realizzando le seguenti operazioni:
  - Visualizzare l'immagine
  - Applicare il filtro che effettua la media aritmetica su una finestra di dimensioni 15x15 e visualizzare il risultato
  - Applicare un'operazione a soglia per eliminare gli oggetti piccoli (considerare una soglia pari al 25 per cento del valore massimo presente nell'immagine filtrata)
  - Visualizzare il risultato dell'elaborazione

Capitolo 12

Morfologia Matematica con Matlab

12.1 Introduzione

Nel contesto dell'image processing l'MM è uno strumento che permette di:

- estrarre dall'immagine componenti utili alla rappresentazione e descrizione delle regioni come contorni, scheletri e superfici convessi
- attuare tecniche particolari come filtri morfologici, thinning e prunning

Gli operatori morfologici sono basati sugli **elementi strutturali** (SE)

- gli SE sono piccoli insiemi o sottoimmagini usati per sondare un'immagine per studiarne le proprietà di interesse
- gli SE, quando lavorano con le immagini sono definiti come array rettangolari

12.2 Le funzioni MM

12.2.1 strel

Questa funzione permette di definire gli SE basandosi su diverse forme e dimensioni. La sintassi è:

```
1 se = strel(shape, parameters)
```

- **shape** è una stringa che specifica la forma
- **parameters** è una lista di parametri basati sulla forma scelta, come per esempio la dimensione

12.2.2 imdilate

Questa funzione permette la dilatazione secondo la sintassi:

```
IM2 = imdilate(IM, SE)
```

- **IM2** è l'immagine in output
- **IM** è l'immagine in input (scala di grigio, binaria)
- **SE** è l'elemento strutturante, o un array di elementi strutturanti restituiti dalla funzione *strel*

12.2.3 imerode

Permette l'erosione:

```
IM2 = imerode(IM,SE)
```

- **IM2** è l'immagine in output
- **IM** è l'immagine in input (scala di grigio, binaria)
- **SE** è l'elemento strutturante, o un array di elementi strutturanti restituiti dalla funzione *strel*

12.2.4 imopen

Questa funzione permette l'opening:

```
IM2 = imopen(IM,SE)
```

12.2.5 imclose

Questa funzione permette il closing:

```
IM2 = imclose(IM,SE)
```

12.2.6 bwhitmiss

Si tratta della funzione con cui si implementa la trasformazione Hit-or-Miss in base alla sintassi:

```
BW2 = bwhitmiss(BW,SE1,SE2)
```

- SE1, SE2 sono gli elementi strutturanti
- l'operazione Hit-or-Miss preserva i pixel i cui vicini corrispondono alla forma di SE1 e non corrispondono alla forma di SE2
- SE1 e SE2 possono essere elementi strutturanti piatti restituiti dalla funzione *strel* o possono essere array vicini
- Il dominio di SE1 e SE2 non dovrebbe avere elementi in comune, ovvero:

```
BW2 = bwhitmiss(BW,SE1,SE2)
```

è equivalente a:

```
imerode(BW,SE1)
```

e

```
imerode(~BW,SE2)
```

12.2.7 imfill

Questa funzione è utilizzata per permettere la copertura di buchi in una immagine binaria, tenendo conto che per buco si intende un insieme di punti dello sfondo che non possono essere raggiunti dal confine dell'immagine. La sintassi è la seguente:

```
BW2 = imfill(BW1,'holes')
```

12.2.8 bwlabel

Questa funzione permette di etichettare le componenti connesse in una immagine binaria 2D secondo la sintassi:

```
L = bwlabel(BW,N)
```

- L è la matrice restituita, della stessa dimensione di BW che contiene l'etichetta per le componenti connesse in BW
- N può assumere il valore di 4 o 8 corrispondente a oggetti 4-connessi o 8-connessi. Se l'argomento è omesso, il valore di default sarà 8
- L è un intero che è uguale o maggiore di 0:

  - i pixel etichettati con 0 riguardano lo sfondo
  - i pixel etichettati con 1 indicano un oggetto
  - i pixel etichettati con 2 indicano il secondo oggetto e così via

Una seconda sintassi è la seguente:

```
[L, NUM] = bwlabel(BW,N)
```

- restituisce in NUM il numero di oggetti connessi trovati in BW

Capitolo 13

Morfologia matematica binaria

13.1 Funzioni

13.1.1 bwmorph

Questa funzione implementa diverse operazioni basate sulla combinazione di erosione e dilatazione. La sintassi è:

```
g = bwmorph(f, operation, n)
```

- f è un'immagina binaria di input
- *operation* è una stringa che specifica le operazioni desiderate
- n (opzionale) è un numero intero positivo che indica il numero di volte che l'operazione deve essere ripetuta. Se n viene omesso, l'operazione viene eseguita una sola volta

13.1.2 imreconstruct

Implementa la ricostruzione morfologica ed è esplicitata dalla sintassi:

```
out = imreconstruct(marker, mask)
```

- *marker* è l'immagine marcatore
- *mask* è l'immagine maschera

13.1.3 imclearborder

Permette la pulizia dei bordi degli oggetti, con la sintassi:

```
g = imclearborder(f, conn)
```

- f è l'immagine di input
- g è il risultato
- *conn* può assumere i valori di 4 oppure 8 (valore di default)

13.2 Esercizi

13.2.1 Esercizio 1

Calcola e visualizza il centro di massa di ogni componente connesso.

```
f= imread('ten-objects.tif');
imshow(f);
[L, n]=bwlabel(f);
n
hold on; % to plot on the top of the image

for k=1:n
  [r,c] = find(L==k);
  rbar=mean(r);
  cbar=mean(c);
  plot(cbar, rbar, 'Marker', 'o', 'MarkerEdgeColor', 'k', '
    MarkerFaceColor', 'k', 'Markersize', 10);
  plot(cbar, rbar, 'Marker', '*', 'MarkerEdgeColor', 'w');
end
```

13.2.2 Esercizio 2

Effettua il thinning dell'immagine $finger-print.tif$ sino ad ottenere la stabilità.

```
f=imread('noisy-fingerprint.tif');
imshow(f);
se = strel('square',3);
fo = imopen(f,se);
figure, imshow(fo);
foc = imclose(fo,se);
figure, imshow(foc);

g1 =bwmorph(foc,'thin',1);
figure, imshow(g1);
g2 =bwmorph(foc,'thin',2);
figure, imshow(g2);
ginf =bwmorph(foc,'thin',Inf);
figure, imshow(ginf);
```

13.2.3 Esercizio 3

Effettuare la sclettrizzazione dell'immagine *bone* e si rimuovano i punti finali (5 pixel).

```
f=imread('bone.tif');
imshow(f)
fs = bwmorph(f,'skel',Inf);
figure, imshow(fs);
for k=1:5
   fs=bwmorph(fs,'spur');
end
figure, imshow(fs);
```

13.2.4 Esercizio 4

Ricostruzione morfologica

```
marker =imread('recon-marker.tif');
mask = imread('recon-mask.tif');
figure, imshow(mask);
figure, imshow(marker);
recon = imreconstruct(marker, mask);
??? MARKER pixels must be <= MASK
pixels.

Error in ==> imreconstruct at 71
im = imreconstructmex(marker,mask);

mm=marker==255;
mm= uint8(mm*255);
recon = imreconstruct(mm, mask);
figure, imshow(recon);
```

13.2.5 Esercizio 5

Effettuare la ricostruzione morfologica per trovare quali caratteri contengono un'asticella verticale.

```
f=imread('book-text.tif');
figure, imshow(f);

fe=imerode(f,ones(51,1));
figure, imshow(fe);

fo=imopen(f, ones(51,1));
figure, imshow(fo);

fobr=imreconstruct(fe,f);
figure, imshow(fobr);
```

13.2.6 Esercizio 6

Effettuare una pulizia dei caratteri che toccano lo schermo.

```
g = imclearborder(f);
figure, imshow(g);
figure, imshow(f - g);
```

Capitolo 14

Segmentazione Watershed in Matlab

La trasformata watersher è calcolata dalla funzione *watershed*, la cui sintassi è la seguente:

$$L = watershed(f)$$

dove L è una etichetta di matrice con interi positivi che corrispondo ai bacini e con i valori zero usati per indicare le creste watershed.

Uno strumento comunemente utilizzato con la trasformata watershed per la segmentazione è la **trasformata della distanza**.

La trasformate della distanza di una immagine binaria è la distanza da ogni pixel dal pixel più vicino di valore diverso da zero.

La trasformata della distanza può essere calcolata usando la funzione *bwdist*, la cui sintassi è:

$$D = bwdist(f)$$

14.1 Segmentazione di una immagine binaria

```
f = imread('binary-dowel-image.tif');
figure, imshow(f), title('Binary image');

fc = ~f ;
figure, imshow(fc), title('Complement image');

D = bwdist(fc);
figure, imshow(imadjust(uint16(D))), title('Distance transform');

L = watershed(-D);
w = L==0;
figure, imshow(w)
title('Watershed lines of the negative of the distance transform');

g2 = f & ~w ;
figure, imshow(g2)
title('Watershed lines superimposed in black over original binary
    image');
```

14.2 Segmentazione di una immagine in toni di grigio

```
f = imread('small-blobs.tif');
figure, imshow(f), title('Original image');

h = fspecial('sobel');
fd = double(f);
g = sqrt(imfilter(fd, h, 'replicate').^2 + imfilter(fd, h, '
    replicate').^2);
figure, imshow(imadjust(uint16(g))), title('Gradient image');

L = watershed(g);
wr = L==0;
figure, imshow(wr), title('Watershed segmentation');

g2 = imclose(imopen(g, ones(3,3)), ones(3,3));
L2 = watershed(g2);
wr2 = L2==0;
f2=f;
f2(wr2)=255;
figure, imshow(f2)
title('Watershed segmentation on smoothed image');
```

14.3 Segmentazione usando i marcatori

Per controllare la sovra-segmentazione si utilizzano i marker (marcatori) interni ed esterni.

La funzione IPT *imregionalmin* calcola la posizione di tutti i minimi locali in una immagine. La sintassi è la seguente:

$$rm = imregionalmin(f)$$

dove f è l'immagine in toni di grigio e rm è una immagine binaria in cui i pixel in primo piano marcano le posizioni del minimo locale; questi ultimi possono rappresentare dettagli irrilevanti.

Per eliminare i minimi estranei si utilizza la funzione IPT *imextendedmin* che calcola l'insieme di punti "bassi" nell'immagine that are deeper than their immediatesurroundings (by a certain height threshold).

Fornendo sia i marcatori interni che esterni, è possibile modificare l'immagine del gradiente usando una procedura chiamata **minima imposition**. Questa tecnica modifica una immagine in toni di grigio in modo che i minimi locali si verificano solo nelle posizioni marcate. Altri calori dei pixel sono spinti verso l'alto per rimuovere tutti gli altri minimi locali

Capitolo 15

Le domande da esame

15.1 Immagini digitali

- Image processing, analysis e understanding. Spiegarne la differenza
- Definire una immagine digitale
- Cosa si intende per risoluzione di una immagine?
- Illustrare il processo di campionamento e quantizzazione
- Descrivere i passi fondamentali di un processo di elaborazione di immagini digitali

15.2 Concetti base sulle immagini digitali

- Come si classificano le operazioni spaziali da applicare su un'immagine digitale?
- Definire il concetto di adiacenza nelle immagini digitali
- Definire una regione di un'immagine binaria
- Definire un contorno in un'immagine binaria
- Definire un edge in un'immagine digitale
- Illustrare il concetto di distanza tra punti in una immagine binaria
- Fornire la definizione di istogramma di un'immagine
- Illustrare le strutture dati tradizionali per rappresentare un'immagine
- Illustrare le strutture dati topologiche per rappresentare un'immagine
- Illustrare le strutture dati gerarchiche per rappresentare un'immagine

15.3 Pre-processing, filtri spaziali, trasformazioni

- Cosa si intende per preprocessing di una immagine
- Descrivere i metodi di preprocessing di una immagine
- Descrivere le trasformazioni geometriche
- Cosa si intende per filtro spaziale?
- Cosa si intende per trasformata di una immagine?
- Cosa si intende per equalizzazione di un istogramma? A cosa serve?
- Come si può realizzare uno stretching del contrasto presente in una immagine?
- Definire un filtro spaziale lineare
- Cosa si intende per operatore lineare?
- Correlazione e convoluzione spaziale. Spiegarne le differenze
- Descrivere i filtri per lo smoothing nel dominio spaziale

15.4 Edge detection

- Illustrare i differenti tipi di edge presenti in un'immagine
- Classificazione degli edge detector
- Illustrare i metodi per l'edge detection basati sull'approssimazione delle derivate prime.
- Descrivere la tecnica della non-maxima suppression
- Illustrare il Marr-Hildreth edge detector
- Illustrare il Canny edge detector
- Quali sono i vantaggi e gli svantaggi degli edge detector?
- Quali sono i criteri per valutare la bontà di un edge detector?
- Descrivere le tecniche locali e regionali per l'edge linking
- Illustrare la trasformata di Hough

15.5 Segmentazione

- Definizione di segmentazione. Metodi e problemi
- Illustrare la segmentazione basata su thresholding
- Descrivere il metodo iterativo per la segmentazione mediante soglia globale
- Descrivere il metodo per la segmentazione mediante soglia ottimale
- Illustrare il metodo region growing per la segmentazione
- Illustrare il metodo region splitting and merging per la segmentazione
- Descrivere la segmentazione mediante watershed morfologico

15.6 Descrizione e rappresentazione

- Cosa si intende per rappresentazione e descrizione di un oggetto?
- Descrivere i metodi per la rappresentazione del contorno
- Illustrare i descrittori del contorno
- Illustrare i descrittori semplici di una regione
- Illustrare i descrittori topologici di una regione
- Illustrare i descrittori basati sulla texture

15.7 Trasformata di Fourier

- Illustrare la rappresentazione di un'immagine nel dominio spaziale e nel dominio delle frequenze
- Fornire la definizione di funzione armonica
- Descrivere brevemente la trasformata di Fourier
- Quali sono le proprietà della DFT?
- Spettro di Fourier, angolo di fase e spettro di potenza. Fornire la loro definizione
- Illustrare il teorema di convoluzione
- Quali sono i passi da eseguire per il filtraggio nel dominio delle frequenze?
- Illustrare la corrispondenza tra filtraggio nel dominio spaziale e filtraggio nel dominio delle frequenze
- Descrivere i filtri per lo smoothing nel dominio delle frequenze
- Descrivere i filtri per lo sharpening nel dominio delle frequenze
- Descrivere i filtri selettivi nel dominio delle frequenze

15.8 Rumore

- Descrivere il rumore e le sue proprietà
- Illustrare i filtri per il rumore (basati sulla media)
- Illustrare i ranking filter
- Riduzione del rumore periodico mediante filtraggio nel dominio delle frequenze

15.9 Morfologia matematica

- Cosa è la Morfologia matematica?
- In cosa consiste una operazione morfologica?
- Fornire la definizione di elemento strutturante nella morfologia matematica
- Definire l'erosione e la dilatazione nella morfologia binaria. Illustrarne le proprietà
- Definire l'opening e il closing nella morfologia binaria. Illustrarne le proprietà
- Definire la trasformazione Hit-or-Miss della morfologia binaria
- Come si può realizzare un filtro morfologico?
- Illustrare l'algoritmo per l'estrazione del contorno mediante operatori morfologici
- Illustrare l'algoritmo per il riempimento di buchi (hole filling) mediante operatori morfologici
- Illustrare l'algoritmo per l'estrazione delle componenti connesse mediante operatori morfologici
- Illustrare l'algoritmo per l'individuazione della convex hull mediante operatori morfologici
- Illustrare l'algoritmo per il thinning mediante operatori morfologici
- Illustrare l'algoritmo per il thickening mediante operatori morfologici
- Illustrare l'algoritmo per l'estrazione dello skeleton mediante operatori morfologici
- Illustrare l'algoritmo per il pruning mediante operatori morfologici
- Definire l'erosione e la dilatazione geodesica nella morfologia binaria
- Illustrare la ricostruzione morfologica
- Illustrare alcune applicazioni della ricostruzione morfologica binaria
- Definire l'erosione e la dilatazione nella morfologica in toni di grigio

- Definire l'opening e il closing nella morfologia in toni di grigio
- Illustrare lo smoothing morfologico
- Definire il gradiente morfologico
- Descrivere le trasformazioni morfologiche top-hat e bottom-hat
- Descrivere la granulometria morfologica
- Illustrare la ricostruzione morfologica in tono di grigio

15.10 Image Processing Toolbox and Matlab

- Quali sono le Data classes per rappresentare un'immagine in Matlab
- Quali sono i tipi di immagini supportate da Matlab
- Lettura, visualizzazione e salvataggio di immagini in Matlab
- Metodi per il miglioramento del contrasto in Matlab
- Filtri lineari in Matlab. Come si implementano?
- Filtri non lineari in Matlab. Come si implementano?
- Edge detection in Matlab. Come si implementa?
- Rumore in Matlab. Come si gestisce?
- Come si implementa in Matlab la segmentazione mediante soglia?
- Descrivere il metodo per il calcolo della DFT 2D in Matlab
- Descrivere il metodo per la visualizzazione della DFT 2D in Matlab
- Come si crea un elemento strutturante in Matlab?
- Descrivere i metodi per realizzare gli operatori morfologici in Matlab
- Come si etichettano le componenti connesse in Matlab?
- A cosa serve la funzione Matlab bwmorph?
- A cosa serve la funzione Matlab imreconstruct?

Indice analitico

Glossario

connesso pulsazione naturale.

trasformazione geometrica affine Esempi di affinità sono rotazioni, omotetie, traslazioni, rototraslazioni, riflessioni. Le affinità non sono necessariamente isometrie, non preservano cioè angoli e distanze, mentre mantengono sempre il parallelismo tra le rette..