Azzolini Riccardo 2019-10-16

Operatori puntuali

1 Trasformazioni puntuali di base

Le trasformazioni puntuali di base sono

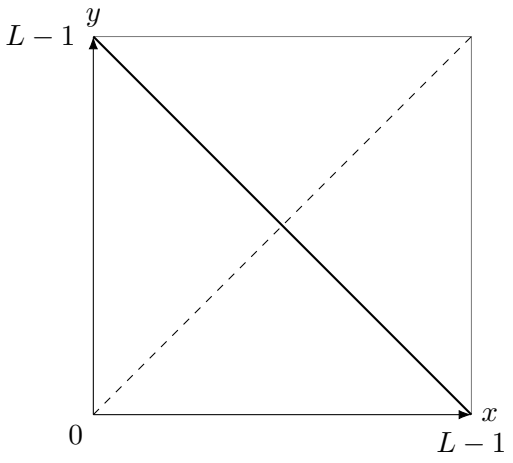
- **lineari** (tra cui l'**identità**, $y = x$, e il **negativo**);
- **logaritmiche**;
- **esponenziali**.

2 Negativo di un'immagine

Un'immagine negativa è un'immagine con i valori dei pixel invertiti:

$$y = (L - 1) - x$$

dove $x \in \{0, \dots, L - 1\}$ (con $L = 256$ per immagini a 8 bit).

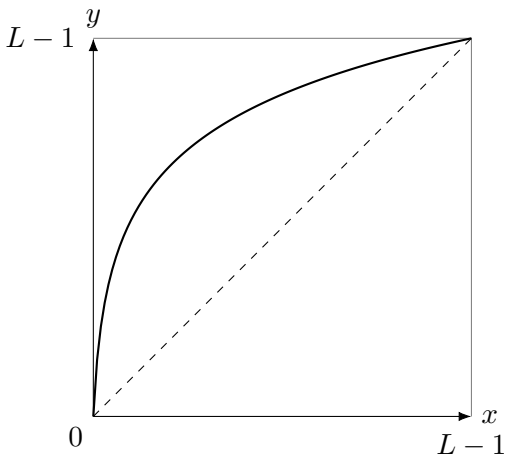


Questa trasformazione equivale all'operatore logico NOT, applicato bit a bit per ogni pixel.

Il negativo si usa spesso in ambito medico, per migliorare l'ispezione visuale di immagini nelle quali si avrebbero normalmente oggetti di interesse chiari su sfondo scuro: la nostra capacità visiva aumenta visionando oggetti scuri su sfondo chiaro.

3 Trasformazione logaritmica

Quando un'immagine ha una dinamica dei livelli di grigio molto più amplia rispetto al range di valori visualizzabili, se si scalassero linearmente i valori rimarrebbero visibili solo le parti più chiare dell'immagine. In questi casi è utile applicare una trasformazione logaritmica, che dilata il range occupato dai livelli scuri e comprime la dinamica dei livelli chiari.



La formula generale di una trasformazione logaritmica è

$$s = c \log(1 + |r|)$$

La costante c assicura che il valore massimo in output sia $L - 1$ (per un'immagine a L livelli di grigio, da 0 a $L - 1$), ed è definita come

$$c = \frac{L - 1}{\log(1 + |R|)}$$

dove R è il massimo valore di grigio nell'immagine di input.

L'uso del valore assoluto nelle formule permette di applicare la trasformazione anche a immagini che contengono valori negativi, o perché sono state acquisite da sensori particolari, oppure perché sono il risultato di alcune trasformazioni matematiche (come ad esempio l'analisi in frequenza).

Il grado di compressione (che corrisponde alla curvatura della funzione di trasformazione) dipende dal range dei valori r in input: un'immagine che contiene valori più alti subisce una maggiore compressione della dinamica delle zone chiare.

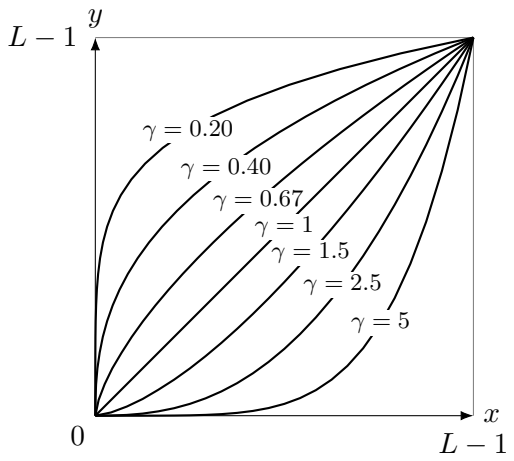
In generale, la trasformazione logaritmica è indicata quando i dettagli di interesse risiedono nelle zone scure e si vogliono enfatizzare. Non è invece adatta se i dettagli si trovano nelle zone chiare dell'immagine, perché questi perderebbero contrasto a causa della compressione della dinamica dei livelli chiari.

4 Trasformazioni esponenziali

Le trasformazioni esponenziali hanno la forma

$$s = c\, r^{\gamma}$$

- se $0 < \gamma < 1$, si ha un effetto di espansione della dinamica dei livelli scuri e comprensione di quella dei livelli chiari;
- se $\gamma > 1$, si ottiene una compressione della dinamica dei livelli scuri e un'espansione di quella dei livelli chiari.



L'esponente che dà risultati migliori deve solitamente essere determinato provando valori diversi, finché non si individua quello che produce l'immagine più soddisfacente.

Oltre all'enhancement finalizzato all'ispezione visuale di particolari immagini, un esempio di uso delle trasformazioni esponenziali è la *gamma correction* per la visualizzazione su monitor a raggi catodici: questi introducono una distorsione di tipo esponenziale (ad esempio, con esponente 2.5) nei livelli di grigio, che può essere compensata applicando prima all'immagine una trasformazione con esponente reciproco (es. $\frac{1}{2.5} = 0.4$).

5 Contrast stretching

Le trasformazioni logaritmiche ed esponenziali possono in alcuni casi migliorare il contrasto, e sono comode da usare in quanto hanno solo 0/1 parametri, ma può essere invece utile migliorare il contrasto mediante trasformazioni lineari.

Le immagini poco contrastate hanno istogrammi concentrati, solitamente nei livelli intermedi della dinamica. Il **contrast stretching** applica una trasformazione lineare per "stirare" l'istogramma in modo che sfrutti l'intera dinamica dei livelli di grigio, migliorando quindi il contrasto.

Siano:

- P_{in} i valori di un pixel nell'immagine di input;
- P_{out} il valore del pixel corrispondente nell'immagine di output;
- c e d il valore minimo e il massimo presenti nell'immagine di input;
- $[a, b]$ l'intervallo di livelli di grigio desiderato nell'immagine di output.

La trasformazione di contrast stretching è data dalla formula:

$$P_{out} = (P_{in} - c)\frac{b-a}{d-c} + a$$

Ad esempio, nel caso più comune, l'intervallo desiderato è $[0, 255]$, e la formula si riduce a:

$$P_{out} = (P_{in} - c)\frac{255}{d-c}$$

Il problema di questo metodo, però, è che la presenza di eventuali *outliers*, cioè pochi pixel con valori lontani dalla media (vicini agli estremi della dinamica, se la maggior parte dei pixel sono concentrati nei livelli intermedi), influenza i valori di c e d, quindi compromette l'efficacia del contrast stretching (perché "sembra" che l'istogramma occupi già una parte ampia della dinamica). Serve quindi un modo di trascurare gli outliers nel calcolo di c e d.

- Una soluzione è considerare non il valore minimo e il massimo, ma (ad esempio) il 5° e il 95° percentile dell'istogramma:
  - il 5 % dei pixel nell'immagine hanno valori inferiori al 5° percentile;
  - il 95 % dei pixel nell'immagine hanno valori inferiori al 95° percentile, quindi il 5 % dei pixel hanno valori superiori.

  In pratica, si escludono così dal calcolo di c e d il 5 % più chiaro e il 5 % più scuro dei pixel.
- Un altro metodo è fissare una frazione della barra più alta dell'istogramma, chiamata *cutoff fraction*, e considerare solo i valori dell'istogramma che hanno barre di altezza superiore alla cutoff fraction. Allora, per calcolare c si scorrono i valori sull'ascissa dell'istogramma in ordine crescente, a partire dal minimo (ad esempio 0), e si seleziona il primo che ha una barra più alta della cutoff fraction. Analogamente, d si calcola scorrendo i valori in ordine decrescente dal massimo (ad esempio 255) e selezionando il primo che ha una barra più alta della cutoff fraction.

Quando c e d non corrispondono effettivamente a minimo e massimo, la formula di contrast stretching può produrre valori fuori dal range $[a, b]$, ma è sufficiente riportare ad a i valori minori di a e a b i valori maggiori di b.

5.1 Confronto con l'equalizzazione

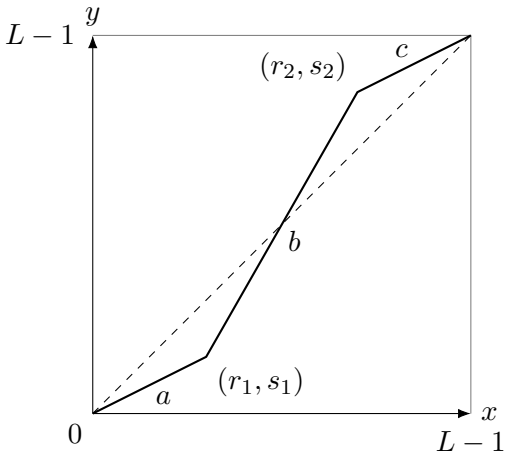
Per determinare se utilizzare l'equalizzazione o il contrast stretching, si osserva l'istogramma:

- quando le barre (eccetto eventuali outliers) occupano un range ristretto di valori di grigio, si applica il contrast stretching;
- quando, invece, le barre occupano tutta la dinamica, ma con diverse "concentrazioni", è necessario usare l'equalizzazione.

5.2 Funzioni lineari a pezzi

Il contrast stretching può essere effettuato mediante **funzioni lineari a pezzi**, che permettono di "stirare" diversamente le barre dell'istogramma a seconda dei livelli di grigio.

Una generica trasformazione lineare composta da tre pezzi ha la forma:

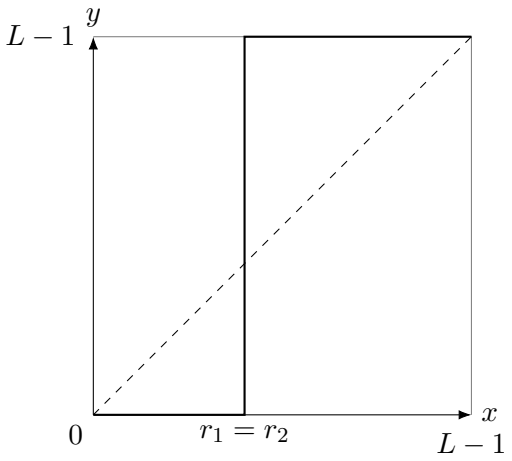


$$y = \begin{cases} ax & 0 \le x \le r_1 \\ b(x - r_1) + s_1 & r_1 \le x \le r_2 \\ c(x - r_2) + s_2 & r_2 \le x \le L - 1 \end{cases}$$

Esse offrono quindi molta flessibilità, ma sono complesse da definire, perché è necessario impostare numerosi parametri per specificare i vari pezzi della funzione.

Di conseguenza, è sconsigliato usarle se non sono strettamente necessarie.

Un caso particolare di questo tipo di funzioni è la sogliatura, che si ottiene dalla trasformazione a tre pezzi raffigurata sopra, ponendo $r_1 = r_2$, $s_1 = 0$ e $s_2 = L - 1$.

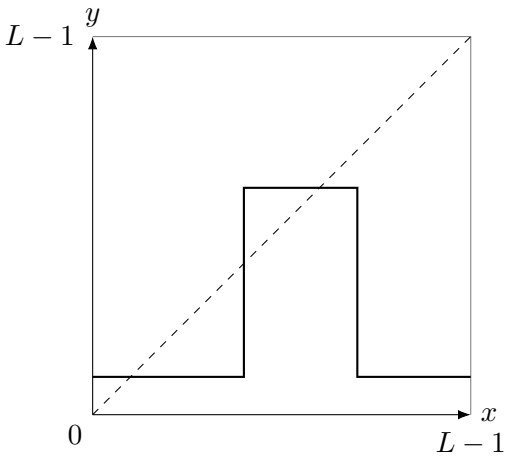


6 Gray level slicing

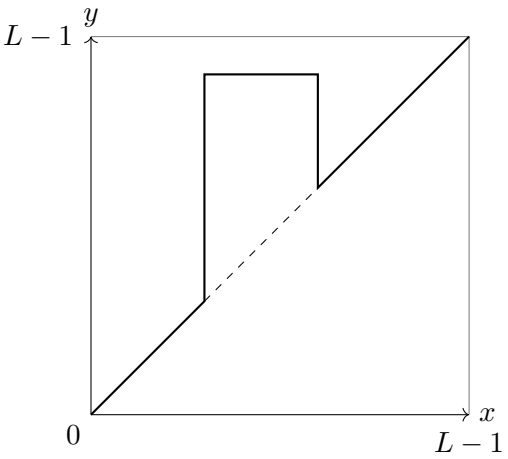
Con la trasformazione di **gray level slicing** (o **intensity slicing**) si enfatizza l'informazione contenuta in un determinato intervallo di livelli di grigio: a tutti i pixel con valori appartenenti a tale intervallo viene assegnato un valore costante, più chiaro (o scuro) dello sfondo, mentre gli altri pixel possono essere:

- impostati tutti a un valore costante di sfondo;
- lasciati inalterati.

Altri pixel ridotti a un livello costante



Altri pixel inalterati



7 Bit plane slicing

L'operazione di **bit plane slicing**

1. scompone un'immagine a n bit (es. 8) in n immagini binarie, chiamate **piani immagine** (**bit planes**), ciascuna contenente i valori dell'i-esimo bit di ciascun pixel (con $i = 0, \ldots, n-1$);
2. ricompone l'immagine, usando solo i k piani più significativi (con $k < n$).

I bit più significativi corrispondono alla maggior parte dell'informazione, mentre i meno significativi tendono a rappresentare il rumore. Di conseguenza, si può ottenere una buona ricostruzione dell'immagine memorizzando solo alcuni dei bit bit più significativi. Il bit plane slicing è quindi una tecnica di compressione, ma di tipo lossy, perché dall'immagine compressa non si può tornare esattamente a quella originale.