

Università degli studi di Catania
Corso di Laurea in Informatica

COMPUTER FORENSICS

2010/2011

Prof. S.Battiato

Relazione del progetto:

Estensione del software ImageJ con un plugin che implementa l'equalizzazione dell'istogramma



a cura di:

Roberto Pera
ropert861@gmail.com

INDICE

Istogramma.....3

Istogramma Cumulativo.....3

Equalizzazione Istogramma.....4

Istogramma Binning.....4

Interfaccia del PlugIn.....5

Descrizione del PlugIn.....6

Immagini in scala di grigio a 8 e 16 bit.....6

Immagini RGB.....6

Immagini in scala di colore a 8 bit.....7

Immagini in scala di grigio a 32 bit.....8

Report…….....9

ISTOGRAMMA

Gli istogrammi sono distribuzioni di frequenza e gli istogrammi delle immagini descrivono la frequenza dei valori di intensità che sono presenti in un immagine.

Realizzare l'istogramma di un immagine è molto semplice. Tutto quello di cui c'è bisogno è un set di contatori, uno per ogni possibile valore di intensità. Inizialmente tutti i contatori sono inizializzati a zero. Successivamente si itera attraverso l'immagine I(u,v), determinando il valore del pixel ad ogni locazione e si incrementa il suo corrispondente contatore di uno. Alla fine, ogni contatore conterrà il numero di pixel nell'immagine che hanno il corrispondente valore di intensità. Dato che un' immagine con K possibili valori di intensità richiederà esattamente K variabili contatori è stato usato un array (H) di K posizioni.

```
for (int v = 0; v < h; v++) {
    for (int u = 0; u < w; u++) {
        int i = ip.getPixel(u,v);
        H[i] = H[i] + 1;
    }
}
```

ISTOGRAMMA CUMULATIVO

L'istogramma cumulativo è derivato dall'istogramma ordinario ed è definito come:

$$\mathsf{H}(i) = \sum_{j=0}^{i} \mathsf{h}(j) \quad \text{for } 0 \le i < K.$$

Un particolare valore H(i) sarà quindi la somma di tutti i valori h(j) con j<= i, nell'istogramma originale. Ragionando in modo ricorsivo otteniamo:

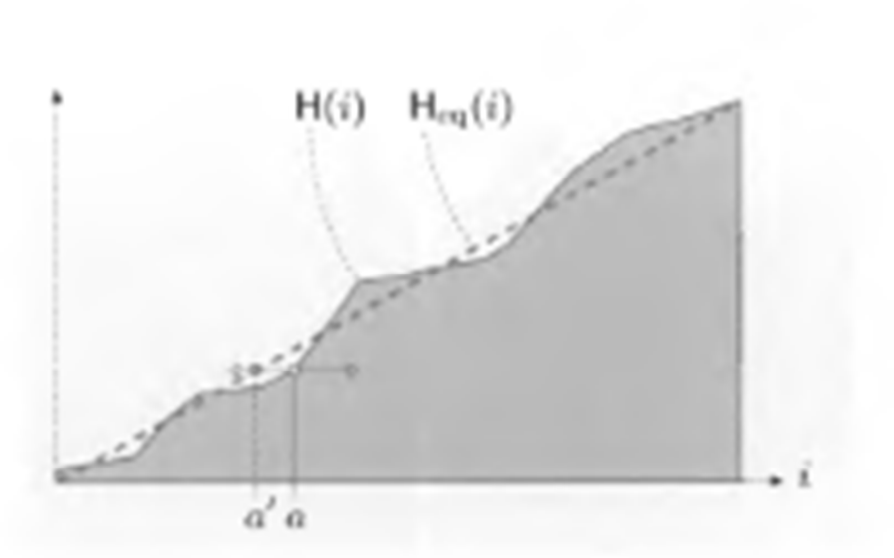
$$\mathsf{H}(i) = \begin{cases} \mathsf{h}(0) & \text{for } i = 0 \\ \mathsf{H}(i-1) + \mathsf{h}(i) & \text{for } 0 < i < K. \end{cases}$$

Il valore massimo sarà uguale al numero totale dei pixel di un'immagine di larghezza M e altezza N.

$$\mathsf{H}(K-1) = \sum_{j=0}^{K-1} \mathsf{h}(j) = M \cdot N;$$

EQUALIZZAZIONE ISTOGRAMMA

L'obiettivo dell'equalizzazione dell'istogramma è quello di ridistribuire i valori dei livelli in modo che l'istogramma dell'immagine modificata sia quanto più uniforme possibile. In particolare osservando che l'istogramma cumulativo di un'immagine con distribuzione uniforme è una retta lineare l'obiettivo si traduce nel trovare un operazione puntuale che shifti le linee dell'istogramma cosìcchè il risultante istogramma cumulativo sia approssimativamente lineare.



L'operazione puntuale desiderata è semplicemente ottenuta dall'istogramma cumulativo H dell'immagine originale come:

$$f_{eq}(a) = \left\lfloor H(a) \cdot \frac{K-1}{MN} \right\rfloor$$

Per un'immagine di dimensioni M x N con pixel nell'intervallo [0,K-1].
Il nuovo valore così ottenuto andrà quindi a sostituire il vecchio ottenendo la nuova immagine equalizzata.

ISTOGRAMMA "BINNING"

Poiché per le immagini a 32 bit non è possibile rappresentare ogni valore di intensità con una singola posizione nell'array, si è deciso quindi di rappresentare con ogni singola voce nell'istogramma un intervallo di valori di intensità. Questa tecnica è spesso denominata "binning". In un istogramma binning di dimensioni B, ogni bin h(j) contiene il numero di elementi dell'immagine che hanno il valore compreso tra $a_j <= a < a_{j+1}$ e così via. Tipicamente l'intervallo di possibili valori in B è diviso in bin di uguale dimensione $k_B=K/B$.

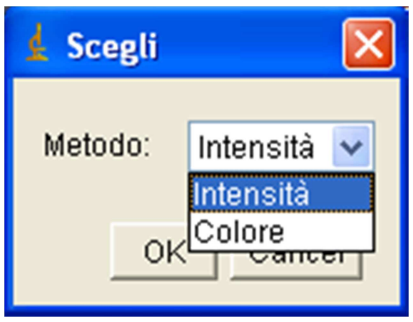
Dato il valore di un singolo pixel il corretto elemento j dell'istogramma sarà facilmente individuabile semplicemente dividendo il valore del pixel per la lunghezza del l'intervallo k_B e arrotondando il valore all'intero minore più vicino attraverso la funzione math.floor.

INTERFACCIA DEL PLUGIN

Per le immagini RGB e per quelle in scala di colore a 8 bit è possibile equalizzare l'istogramma scegliendo da un popup menu creato con una GenericDialog una delle seguenti modalità:

"Modalità Intensità": l'equalizzazione viene applicata alla luminanza totale, utilizzando lo spazio di colore YCbCr.

"Modalità Colore": l'equalizzazione viene applicata separatamente sui tre canali di colore RGB.



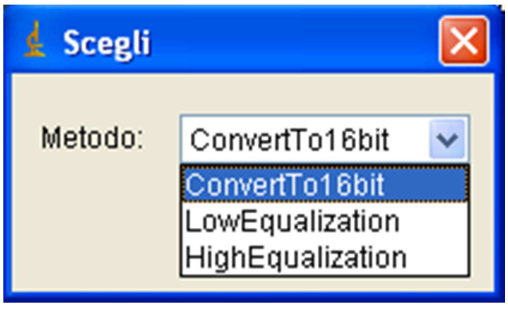
Per le immagini in scala di grigio a 8 e 16 bit non è previsto alcun menu poiché le due modalità viste sopra forniscono gli stessi risultati.

Per le immagini in scala di grigio a 32 bit è invece possibile scegliere una delle seguenti modalità:

"ConvertTo16bit": Converte l'immagine in una a scala di grigio a 16 bit e ne equalizza l'istogramma;

"LowEqualization":equalizza l'istogramma binning dei soli pixel con valore inferiore ad un determinata soglia, settando gli altri pixel a 65535;

"HighEqualization":equalizza l'istogramma binning dei soli pixel con valore superiore ad una determinata soglia, settando gli altri pixel a 0;



DESCRIZIONE DEL PLUGIN

Il plugin tratta tutti i tipi di immagine (eccetto gli stack) e necessita di un'immagine in input (motivo per cui implementa l'interfaccia PlugInFilter).
Per ogni immagine vengono recuperati le dimensioni attraverso i metodi getWidth() e getHeight() e si procede all'equalizzazione dell'istogramma in base al tipo, recuperato attraverso il metodo getType():

Immagini in scala di grigio a 8 e 16 bit:
La realizzazione dell'istogramma è già implementata in ImageJ ed è accessibile attraverso il metodo getHistogram(); si costruisce quindi l'istogramma cumulativo sulla base dell'istogramma appena ottenuto e si procede con l'equalizzazione e la scrittura della nuova immagine. Queste ultime due fasi sono state unificate nell' unico metodo ScriviGray.

Immagini RGB:

Modalità Intensità:
Le componenti R,G,B nelle immagini RGB sono ottenute con il metodo getRGB (byte [] R, byte [] G, byte[]B) della classe ColorProcessor (è necessario effettuare il cast del processore).
Le opportune conversioni tra lo spazio di colore RGB e quello YCbCr vengono effettuate attraverso il metodo RgbToYcbcr che setta i valori calcolati di Y, Cb e Cr nei tre array vuoti che gli vengono passati come parametro insieme ai tre array contenenti i valori di R,G e B.
Le dimensioni di Y, Cb e Cr sono pari a M, in modo tale da far corrispondere ad ogni pixel dell'immagine i suoi valori di luminanza e crominanza semplicemente in base alla posizione occupata nell'array.
Successivamente vengono calcolati l'istogramma (vedi istogramma) della luminanza e l'istogramma cumulativo (vedi istogramma cumulativo) della luminanza rispettivamente con i metodo Istogramma e Cumulativo.
La fase di equalizzazione viene realizzata con il metodo Equalizza che salva i nuovi valori equalizzati della luminanza in un array di dimensioni M che gli viene passato come parametro insieme all'array con i vecchi valori di luminanza e all'istogramma cumulativo.
Dopo aver riconvertito i valori da ycbcr a Rgb con il metodo ycbcrToRgb ed aver ricompattato i valori delle tre componenti del colore in un'unica variabile int settando eventualmente il nuovo valore a 0 o 255 nel caso in cui fosse rispettivamente minore di 0 o maggiore di 255 con il metodo Controlla, si salvano i valori ottenuti nell'array vuoto passato come parametro e si procede alla scrittura della nuova immagine con il metodo ScriviRGB che associa ad ogni pixel il suo nuovo valore in base alla posizione occupata nell'array.

Modalità Colore:

Per recuperare il valore delle tre componenti RGB è necessario recuperare prima il valore dei pixel e per far ciò viene utilizzato il metodo getPixel() che restituisce un riferimento all'array dei pixel dell'immagine. Poiché il tipo di array dipende dal tipo di immagine i valori delle tre componenti del colore sono compattati in un'unica variaibile int. Ogni singola componente viene calcolata quindi nel seguente modo: R[i] = (pix[i] & 0x00ff0000)>>16; G[i] = (pix[i] & 0x0000ff00)>>8;B[i] = pix[i] & 0x000000ff; con 0=<i<pix.length;

Successivamente si procede come nel caso di scelta della modalità intensità operando però separatamente sui tre canali, ovvero con la creazione degli istogrammi del Rosso, del Verde e del Blu con il metodo Istogramma, con la creazione degli istogrammi cumulativi del Rosso, del Verde e del Blu con il metodo Cumulativo e con l'equalizzazione dell'istogramma del Rosso, del Verde e del Blu con il metodo Equalizza. Poiché in questo caso non è necessario un metodo per la conversione da ycbcr a rgb si procede direttamente a ricompattare i valori delle tre componenti del colore in un'unica variabile int ed a salvare il tutto in un array (passato come parametro) con il metodo Unisci.

Come per la modalità Intensità si usa il metodo ScriviRGB per settare i nuovi valori nell'immagine ed ottenere l'immagine equalizzata.

Immagini in scala di colore a 8 bit:

Ogni pixel nelle immagini a scala di colore contiene un indice k. Il valore del colore per ogni k è definito da un accesso nella tabella dei colori "palette" P[k].

Le componenti R,G,B nelle immagini a scala di colore sono recuperate attraverso i metodi getReds(byte r[]), getGreens(byte [] g) e getBlues(byte b[]) della classe IndexColorModel ottenuta come cast del ColorModel del processore restituito con il metodo getColorModel(). I metodi ritornano la corrente lookup table per R,G,B in tre separati array di byte. Le dimensioni di queste tabelle possono essere determinate con il metodo getMapSize().

N.B. Poiché gli elementi della palette sono byte con segno, se essi devono essere interpretati come byte senza segno con valori nell'intervallo tra 0 e 255 è necessario effettuare gli opportuni cast (0xff).

Modalià Intensità:

Si opera esattamente come nel caso della modalità intensità per le immagini RGB; le uniche differenze riguardano: le dimensioni degli array che in questo caso devono essere pari a mapSize in modo da far corrispondere ad ogni colore della palette i suoi valori di luminanza e crominanza, il metodo IstogrammaPal poiché ogni pixel adesso rappresenta un indice nella tabella dei colori e quindi nell'array della luminanza, il metodo YCbCrToRGB dato che deve restituire i valori R,G e B ricompattati e controllati in 3 separati array di byte che servono

per la scrittura della nuova immagine che in questo caso avviene creando un nuovo IndexColorModel e assegnandolo al processore dell'immagine.

Modalità Colore:
In questo caso si opera come nel caso della modalità intensità appena esaminata agendo però separatamente sui tre canali di colore. Ciò implica delle modifiche ai metodi: IstogrammaPalette poiché oltre al valore del pixel il cast deve essere effettuato anche al valore del colore nella palette ed EqualizzaPalette dato che non è più necessario iterare per tutti i pixel dell'immagine ma solamente per i valori presenti nella palette. Ovviamente non saranno presenti i metodi per le conversioni tra gli spazi di colore.

Immagini in scala di grigio a 32 bit:
Nelle immagini in scala di grigio a 32 bit ogni pixel è rappresentato in formato float con valori compresi nell'intervallo tra 0.0 e 1.0.

Modalità ConvertTo16bit:
Converte l'immagine in scala di grigio a 32 bit in una a scala di grigio a 16 bit, in particolare viene creata una nuova istanza della classe ImageConverter e successivamente viene invocato il metodo di conversione convertToGray16().
Una volta ottenuta la nuova immagine si procede con l'equalizzazione (vedi immagini in scala di grigio a 8 e 16 bit);

Modalità LowEqualization:
Equalizza solamente i pixel "scuri" dell'immagine, ossia pixel il cui valore sia inferiore a (1.0/2 = 0.5). Poiché sono possibili 2^32=4.292.967.296 valori di intensità compresi tra 0.0 e 1.0 si è diviso l'intervallo di possibili valori (0.5) in 65536 intervalli uguali, ognuno di lunghezza k_B = 0.5/65536=0.000007629 e viene creato l'istogramma binning con il metodo Binning (vedi istogramma binning).
Dopo aver realizzato l'istogramma cumulativo si crea una nuova immagine in scala di grigio a 16 bit e si procede alla scrittura del corrispondente valore equalizzato se il valore del pixel è <0.5, altrimenti lo si setta a 65535;

Modalità HighEqualization:
Si opera esattamente come per la modalità LowEqualization con la differenza che vengono equalizzati solamente i pixel "chiari" dell'immagine, ossia i pixel il cui valore sia superiore a 0.5 e quindi bisogna sottrarre 65535 per riportare correttamente i valori nell'intervallo tra 0 e 65535; ovviamente si procederà alla scrittura del corrispondente valore equalizzato solo se il valore del pixel sia >= 0.5, altrimenti lo si setta a 0.

REPORT

Se tutte le operazioni sono state eseguite correttamente il plugin salverà un file di report dinamico in formato html nella stessa directory dell'immagine contenente alcune informazioni tra cui il nome dell'utente, la data e l'ora di generazione del report, la workstation di lavoro, il path completo del file e la modalità di equalizzazione scelta (per le immagini in scala di grigio a 8 e 16 bit la modalità di equalizzazione è stata definita come "Intensità").