



UNIVERSITÀ DEGLI STUDI DI PARMA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
CORSO DI LAUREA IN INGEGNERIA
INFORMATICA, ELETTRONICA E DELLE TELECOMUNICAZIONI

**ESTRAZIONE DI SPIGOLI IN IMMAGINI
MEDIANTE L'ALGORITMO DI HARRIS**

IMAGE CORNERS EXTRACTION BY HARRIS ALGORITHM

Relatore:
Prof. Ing. RICCARDO RAHELI

Correlatore:
Dott. Ing. CARLO TRIPODI

Tesi di Laurea di:
LEONARDO TANZI

ANNO ACCADEMICO 2015/2016

Indice

| | |
|---|-----------|
| Introduzione | 1 |
| 1 Estrazione di spigoli in immagini | 3 |
| 1.1 Algoritmo di Moravec | 5 |
| 1.2 Algoritmo di Harris | 13 |
| 2 Algoritmo per l'affinamento dell'estrazione di spigoli | 29 |
| Conclusioni | 39 |

Introduzione

Lo scopo di questo lavoro è lo studio dell'estrazione delle caratteristiche di un'immagine. In particolare, si è interessati all'estrazione degli spigoli. L'algoritmo, introdotto da Chris Harris e Mike Stephens nel 1988 [1], noto come algoritmo di Harris, è tuttora il mezzo più utilizzato per il rilevamento di punti notevoli in immagini in scala di grigi. Vista l'enorme quantità di dati contenuti in un'immagine, è fondamentale avere a disposizione un algoritmo che, a partire da questi dati, riesca a riconoscere e selezionare solo quelli di interesse per lo scopo che ci si è preposti. Le applicazioni di questo algoritmo sono numerose: visione artificiale, tracciamento del movimento, riconoscimento di oggetti, ricostruzione tridimensionale, allineamento di immagini e tante altre.

In questa tesi affronteremo nel dettaglio due algoritmi: l'algoritmo di Moravec [2], cioè l'algoritmo da cui Harris e Stephens partirono per formulare il loro, e l'algoritmo di Harris stesso. L'algoritmo di Harris funziona a livello di pixel, associa cioè gli spigoli a coordinate intere dell'immagine. Nella seconda parte di questa tesi si è studiato un ulteriore algoritmo per affinare la precisione dell'algoritmo a livello sub-pixel, lavorando quindi con coordinate decimali.

Capitolo 1

Estrazione di spigoli in immagini

Lo scopo di questa tesi è studiare l'estrazione di punti notevoli in immagini mediante l'algoritmo di Harris. Un buon punto notevole è contraddistinto dalle seguenti

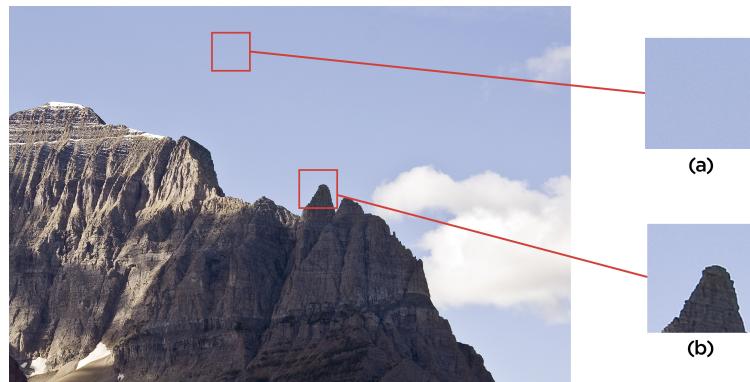


Figura 1.1: Individuazione di punti notevoli in una immagine

caratteristiche [3]:

1. ripetibilità: lo stesso punto notevole deve poter essere riconosciuto nella stessa immagine in diverse condizioni;
2. salienza: ogni punto notevole deve avere una descrizione distintiva, cioè che lo distingua da tutti gli altri punti dell'immagine;

3. locazione: un punto notevole deve occupare una zona ristretta dell'immagine.

Si osservi la figura 1.1. Si sono messe in evidenza due zone dell'immagine. Se si dovesse scegliere un candidato punto notevole fra questi due, cioè un punto che rispetti le tre proprietà elencate sopra, si sceglierrebbe senza dubbio il (b). Per mettere questo concetto in pratica si osservi ora la figura 1.2. I punti più facilmente riconoscibili in entrambe le immagini, cioè i sopra citati punti notevoli, sono evidenziati da ovali ed i punti corrispondenti sono collegati da linee. Si nota, inoltre, che i punti notevoli sono spesso gli spigoli, o angoli, in un'immagine (per questo d'ora in poi i due termini verranno trattati come sinonimi).

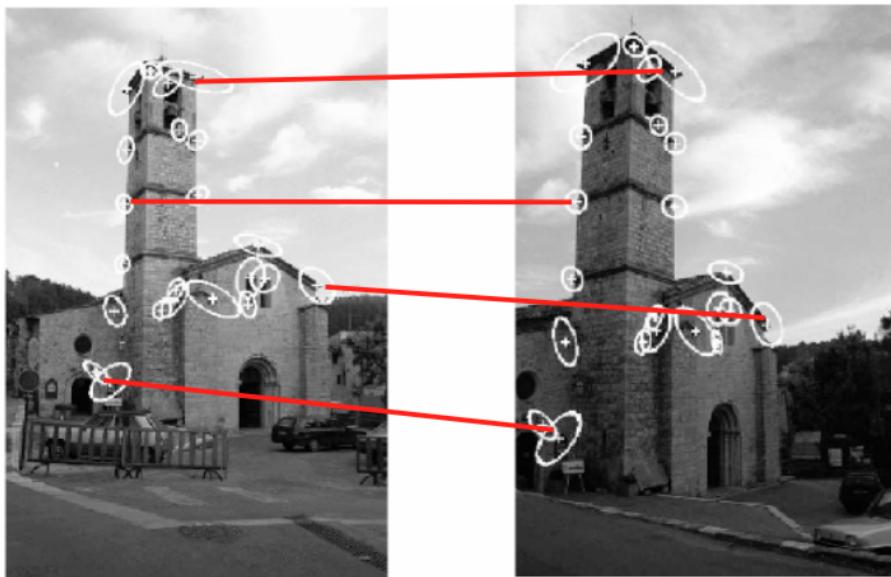


Figura 1.2: Punti notevoli individuabili da angolazioni differenti, tratta da [4]

Prima di affrontare nel dettaglio la descrizione dell'algoritmo, è necessario spiegare l'idea di base: si immagini di far scorrere una finestra all'interno dell'immagine in figura 1.3 e di compiere per ogni pixel tanti spostamenti di questa finestra in tutte le direzioni. Misurando la variazione di intensità media della finestra, cioè il valore

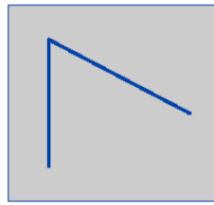


Figura 1.3: Spigolo generico, tratta da [4]

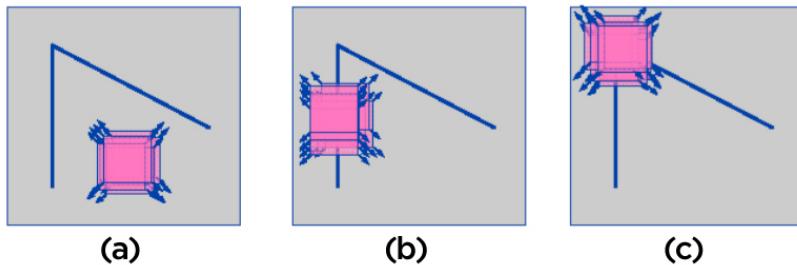


Figura 1.4: Studio di tre zone differenti, tratta da [4]

del livello di grigio che va da 0 (nero) a 255 (bianco), per i diversi spostamenti, si può capire la natura del pixel considerato, come si può vedere in figura 1.4 e come si vedrà meglio in seguito. Nella figura 1.4a, non si riscontra alcuna variazione di intensità media spostando la finestra in ogni direzione: la zona considerata sarà quindi una zona piatta. Nella figura 1.4b, si verifica una significativa variazione di intensità media spostando la finestra in una sola direzione: la zona considerata sarà quindi un bordo. Infine, nella figura 1.4c, si verifica una significativa variazione di intensità media spostando la finestra in tutte le direzioni: la zona considerata sarà quindi uno spigolo.

1.1 Algoritmo di Moravec

Prima di introdurre l'algoritmo di Harris è necessario soffermarsi su ciò da cui Harris e Stephens partirono per formulare la loro deduzione: l'algoritmo di Moravec [2].

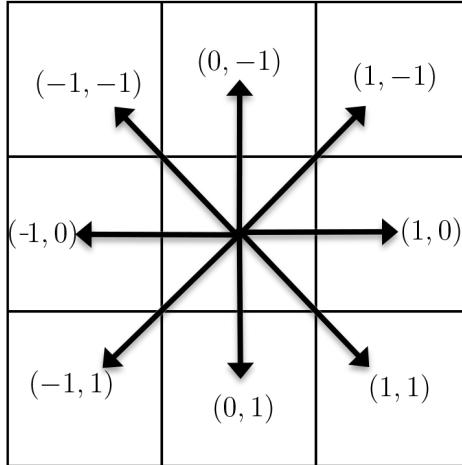


Figura 1.5: I possibili spostamenti in otto direzioni.

Approssimativamente, l'algoritmo considera, all'interno di un'immagine, una finestra locale intorno ad un singolo pixel p e studia la variazione dell'intensità nell'intorno di p , valutando la variazione di ogni singolo pixel per gli spostamenti nelle otto possibili direzioni rappresentate in figura 1.5, in cui lo spostamento è identificato dalla notazione (u, v) e $u, v \in [-1, 0, 1]$. In particolare, non si considera lo spostamento $(0, 0)$, cioè lo spostamento nullo. Essendo uno spigolo, per definizione, un punto caratterizzato da una grossa variazione di intensità in ogni direzione in un suo intorno, più alta sarà la variazione più il pixel sarà un buon candidato ad essere classificato spigolo. Questo procedimento viene svolto per tutti i pixel dell'immagine.

Analiticamente, definita l'intensità nel pixel in posizione (x, y) come $I(x, y)$, la differenza di intensità E per uno spostamento (u, v) è data da:

$$E(u, v) = \sum_{x,y} w(x - x_0, y - y_0) [I(x + u, y + v) - I(x, y)]^2 \quad (1.1)$$

dove $w(x, y)$ è la finestra che si prende in considerazione e che viene via via spostata in ogni pixel dell'immagine. $w(x, y)$ è una finestra quadrata $d \times d$ con d dispari centrata

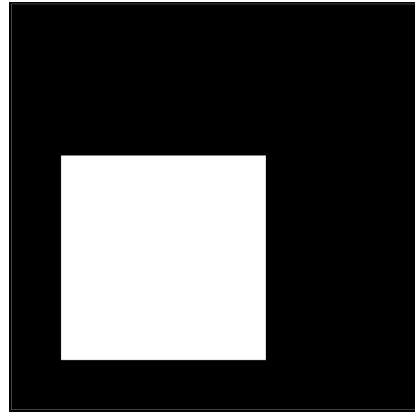


Figura 1.6: Immagine originale.

in (x_0, y_0) definita come:

$$w(x, y) = \begin{cases} 1 & x_0 - \lfloor \frac{d}{2} \rfloor \leq x \leq x_0 + \lfloor \frac{d}{2} \rfloor, y_0 - \lfloor \frac{d}{2} \rfloor \leq y \leq y_0 + \lfloor \frac{d}{2} \rfloor \\ 0 & \text{altrimenti} \end{cases}$$

e $\sum_{x,y} w(x, y)[I(x+u, y+v) - I(x, y)]$ è la sommatoria della differenza fra le intensità dei pixel in posizione $(x+u, y+v)$, cioè $I(x+u, y+v)$, e le intensità dei pixel in posizione (x, y) , cioè $I(x, y)$. L'elevamento al quadrato permette di evidenziare al meglio le differenze di intensità e di lavorare solo con valori positivi.

Applichiamo questo ad un esempio concreto costituito dall'immagine di 8 pixel di larghezza per 8 pixel di altezza, per semplicità composta solo da bianco e nero, mostrata in figura 1.6. In figura 1.7 si rappresenta questa immagine con il valore 1 in corrispondenza del colore bianco e 0 in corrispondenza del colore nero, evidenziando i due assi x e y . Il pixel evidenziato in rosso, chiamato per comodità p , è per l'appunto il pixel intorno al quale ci si concentra. La finestra $w(x, y)$ costruita nell'intorno di p ha dimensione 3×3 con $3 \leq x \leq 5$ e $3 \leq y \leq 5$ ed è rappresentata dai pixel evidenziati in giallo. In figura 1.8 si può notare inoltre come vengono definiti gli assi u e v . Una

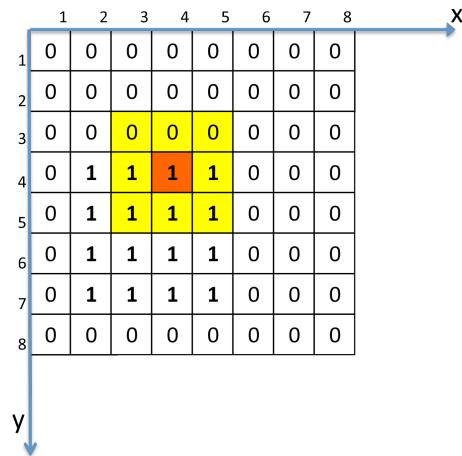


Figura 1.7: Immagine rappresentata con 0 e 1.

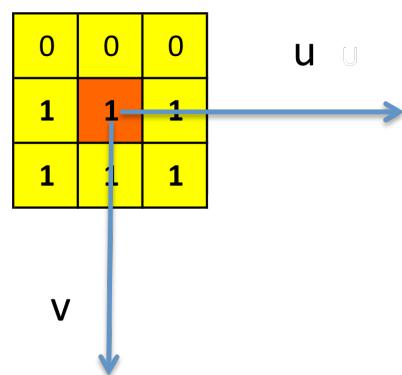


Figura 1.8: Ingrandimento della finestra considerata.

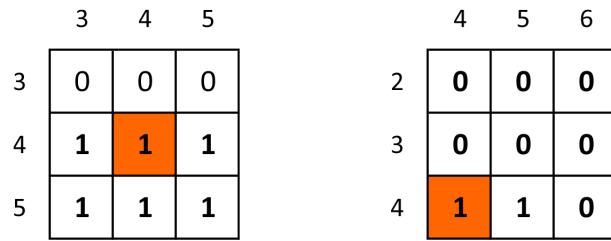


Figura 1.9: Finestra considerata in partenza e finestra considerata dopo lo spostamento.

volta compresa l'impostazione della equazione (1.1), è necessario calcolare le differenze di intensità di ogni pixel per gli otto possibili spostamenti.

Nelle righe seguenti, si esemplifica il procedimento per la coppia $(u = 1, v = -1)$. Il metodo è lo stesso per qualunque coppia di (u, v) . Si immagini di spostarsi di 1 in direzione u e di -1 in direzione v . In figura 1.9 sono rappresentati i 9 pixel all'interno di $w(x, y)$, e i 9 pixel nell'intorno di $(x + 1, y - 1)$. Seguendo il procedimento dettato dalla formula matematica, si calcola la differenza di intensità pixel per pixel per le due matrici 3×3 , e si eleva ciascuna differenza al quadrato (nell'esempio in questione inutile, essendo un'immagine composta solo da 1 e 0). La differenza di intensità relativa pixel per pixel per le due matrici è quindi:

$$\begin{aligned}
 E(1, -1) &= \sum_{x,y} w(x, y) [I(x + 1, y - 1) - I(x, y)]^2 \\
 &= \sum_{x=3}^5 \sum_{y=3}^5 [I(x + 1, y - 1) - I(x, y)]^2 \\
 &= \sum_{x=3}^5 \{[I(x + 1, 2) - I(x, 2)]^2 + [I(x + 1, 3) - I(x, 3)]^2 + [I(x + 1, 4) - I(x, 5)]^2\} \\
 &= [I(4, 2) - I(3, 3)]^2 + [I(4, 3) - I(3, 4)]^2 + [I(4, 4) - I(3, 5)]^2 + \\
 &= +[I(5, 2) - I(4, 3)]^2 + [I(5, 3) - I(4, 4)]^2 + [I(5, 4) - I(4, 5)]^2 +
 \end{aligned}$$

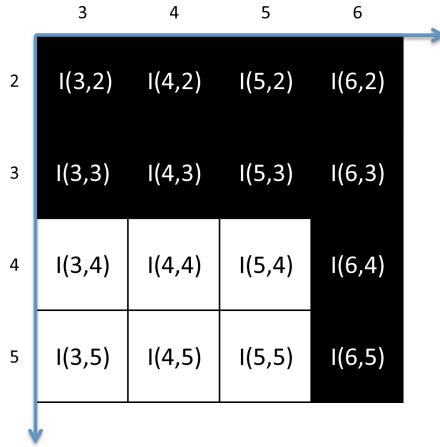


Figura 1.10: Figura di riferimento per la formula 1.2.

$$= +[I(6,2) - I(5,3)]^2 + [I(6,3) - I(5,4)]^2 + [I(6,4) - I(5,5)]^2 = 4 \quad (1.2)$$

Il perché il risultato sia 4 è comprensibile osservando la figura 1.10, che mette in evidenza i valori delle intensità presenti nell'equazione (1.2) per ogni pixel. Successivamente si effettua lo stesso procedimento per i sette spostamenti rimanenti, come si può vedere in figura 1.11, dove la finestra bordata rappresenta i 9 pixel da confrontare con i pixel di partenza per gli otto possibili spostamenti considerati. Svolgendo lo stesso calcolo visto per $E(1, -1)$, i risultati sono i seguenti:

$$E(1, -1) = 4$$

$$E(1, 0) = 2$$

$$E(1, 1) = 4$$

$$E(0, -1) = 3$$

$$E(0, 1) = 3$$

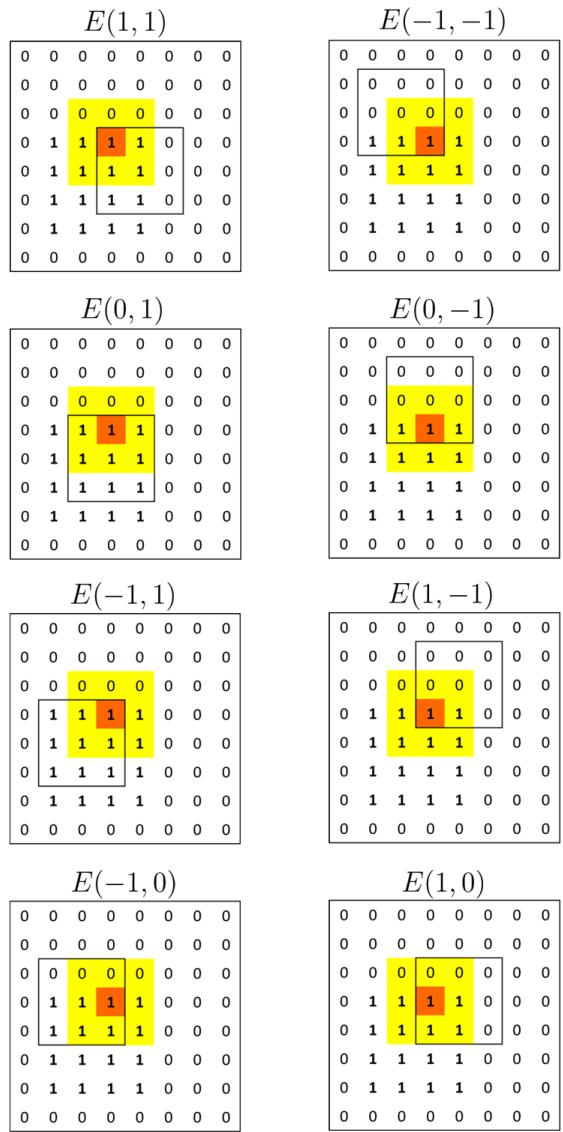


Figura 1.11: Otto possibili spostamenti della finestra di partenza.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 4 | 1 | 2 | 0 | 0 | 2 | 1 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 1 | 2 | 0 | 0 | 2 | 1 | 0 | 0 |
| 8 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

Figura 1.12: Immagine con i valori associati ad ogni pixel.

$$E(-1, -1) = 3$$

$$E(-1, 0) = 0$$

$$E(-1, 1) = 3.$$

Tra questi si sceglie il valore minimo, in questo caso $E(-1, 0) = 0$, e lo si associa al pixel p . Già da questo si intuisce che il pixel p preso in considerazione non sarà un buon candidato ad essere classificato spigolo. Ci si sposta poi su un altro pixel e si ripete il procedimento. Il risultato finale dell'associazione è rappresentato in figura 1.12, dove i pixel evidenziati in giallo sono i massimi locali, e corrispondono agli spigoli dell'immagine di partenza.

Riassumendo il tutto, l'algoritmo considera, all'interno di un'immagine, una finestra locale intorno ad un singolo pixel e studia la variazione dell'intensità dei pixel all'interno della finestra confrontando questa con i pixel associati a tutti i possibili spostamenti nelle otto direzioni. Una volta fatto questo per tutte le direzioni, associa il valore minimo della differenza di intensità pixel per pixel, che chiamiamo $\min[E(u, v)]$,

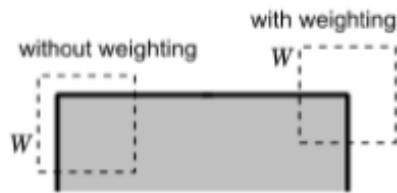


Figura 1.13: Finestra binaria e finestra gaussiana, tratta da [5].

in modo tale che il numero associato a quel pixel indichi che, se si parte da una finestra centrata in quel pixel e ci si sposta in qualunque direzione, l'intensità dei pixel considerati varierà al minimo di $\min[E(u, v)]$. Più $\min[E(u, v)]$ è alto più il pixel preso in considerazione sarà un buon candidato ad essere classificato spigolo. Fatto questo pixel per pixel, si selezionano i massimi (in intorni locali) fra i valori ottenuti. La posizione di questi massimi indica gli spigoli dell'immagine studiata. Harris e Stephens partirono da questo per formulare il loro metodo.

1.2 Algoritmo di Harris

L'algoritmo di Harris e Stephens, introdotto nel 1988, è tutt'ora il mezzo più utilizzato per il rilevamento di punti notevoli in immagini [1]. L'idea di partenza per spiegare questo algoritmo è quella di ottimizzare l'equazione (1.1) che esprime l'operatore di Moravec, introducendo in particolare due miglioramenti proposti da Harris.

In primo luogo, l'operatore di Moravec utilizza una finestra binaria e quadrata, che porta ad un risultato impreciso, per cui la posizione del punto corrispondente allo spigolo rilevato potrebbe non trovarsi al centro della finestra considerata, poiché questa finestra tende a massimizzare tutte i pixel presenti all'interno della porzione di immagine allo stesso modo, senza privilegiarne alcuno. Questo inconveniente può essere facilmente superato grazie all'utilizzo di una funzione di peso, che assegna un'impor-

tanza maggiore ai pixel centrali. Per questo Harris propone l'utilizzo di una finestra gaussiana $w(x, y)$, limitata ad una data regione, in modo da evidenziare al meglio le variazioni più caratteristiche. La finestra $w(x, y)$ centrata in (x_0, y_0) e di dimensioni $d \times d$ con d dispari è definita come:

$$w(x, y) = \begin{cases} e^{-\frac{(x^2+y^2)}{2\sigma^2}} & x_0 - \lfloor \frac{d}{2} \rfloor \leq x \leq x_0 + \lfloor \frac{d}{2} \rfloor, y_0 - \lfloor \frac{d}{2} \rfloor \leq y \leq y_0 + \lfloor \frac{d}{2} \rfloor \\ 0 & \text{altrimenti} \end{cases}$$

dove σ è la deviazione standard ed è solitamente posta pari a 1. Un esempio degli effetti della pesatura è mostrato in figura 1.13.

In secondo luogo, ricordando che in corrispondenza di spigoli si hanno significative variazioni nelle intensità delle immagini, volendo rilevare queste nell'immagine I è possibile esprimere la I stessa attraverso il suo sviluppo di Taylor nell'origine dello spostamento, per semplicità troncato al primo ordine. Questo permette una buona approssimazione della misura. Per capire meglio cosa significhi questa approssimazione, è necessario osservare la figura 1.14. I grafici 1.14a e 1.14b sono la rappresentazione tridimensionale di $E(u, v)$ con la formulazione di Moravec, 1.14c e 1.14d con la formulazione di Harris. I due grafici 1.14a e 1.14c rappresentano rispettivamente uno spigolo, mentre le figure 1.14b e 1.14d rappresentano un bordo. Dalla figura si può vedere come, sviluppando in serie di Taylor e troncando al primo ordine, le due funzioni $E(u, v)$ vengano approssimate mantenendo le caratteristiche generali immutate.

Lo sviluppo di Taylor troncato al primo ordine per uno spostamento $f(x+u, y+v)$ è definito come:

$$f(x+u, y+v) \simeq f(x, y) + u f_x(x, y) + v f_y(x, y)$$

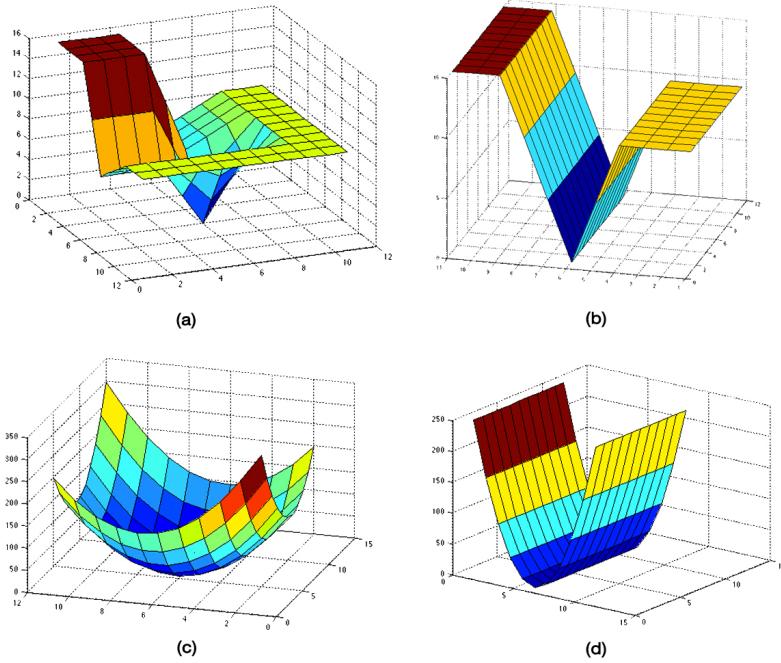


Figura 1.14: Effetti dello sviluppo di Taylor.

dove $f_x(x, y)$ e $f_y(x, y)$ sono rispettivamente la derivata rispetto a x e rispetto a y .

Prima di procedere occorre definire il significato di derivata su un insieme discreto. Nel documento originale, Harris approssima le derivate nelle due direzioni con la

convoluzione fra l'immagine e il vettore $\begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$ per la direzione x , e il vettore

$$\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

per la direzione y . Poiché la derivata direzionale di un'immagine deve esprimere

quanto l'intensità dell'immagine in questione vari in quella direzione, è perfettamente

comprendibile la soluzione proposta da Harris[1].

Oggi giorno, il metodo più utilizzato per approssimare la derivata su un insieme discreto è calcolare la correlazione fra l'immagine di partenza e la derivata nelle due direzioni di una gaussiana [6]. Il concetto è identico a quello proposto da Harris,

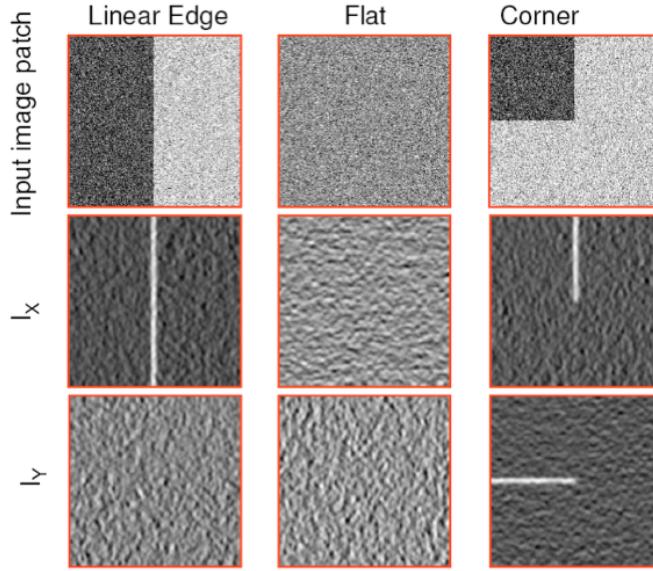


Figura 1.15: Esempi di I_x e I_y , tratta da [4].

ma utilizzando le derivate di una gaussiana si ottiene una demarcazione più precisa e accurata. Noi utilizzeremo il metodo proposto da Harris nel documento originale. Il risultato di queste convoluzioni verrà espresso nei prossimi passaggi come $I_x(x, y)$ e $I_y(x, y)$. Il concetto appena espresso è esemplificato in figura 1.15, dove sono visibili $I_x(x, y)$ e $I_y(x, y)$ per un generico spigolo, un generico bordo e una zona piatta.

Applicando questo procedimento all’equazione (1.1), quello che otteniamo è:

$$\begin{aligned}
 E(u, v) &= \sum_{x,y} w(x - x_0, y - y_0) [I(x + u, y + v) - I(x, y)]^2 \\
 &\simeq \sum_{x,y} w(x - x_0, y - y_0) [I(x, y) + uI_x(x, y) + vI_y(x, y) - I(x, y)]^2 \\
 &= \sum_{x,y} w(x - x_0, y - y_0) [uI_x(x, y) + vI_y(x, y)]^2 \\
 &= \sum_{x,y} w(x - x_0, y - y_0) [u^2I_x^2(x, y) + 2uvI_x(x, y)I_y(x, y) + v^2I_y^2(x, y)]
 \end{aligned}$$

Svolgendo gli opportuni calcoli, questo risultato è esprimibile in forma matriciale come:

$$E(u, v) \simeq \sum_{x,y} w(x - x_0, y - y_0) \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2(x, y) & I_x(x, y)I_y(x, y) \\ I_x(x, y)I_y(x, y) & I_y^2(x, y) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \quad (1.3)$$

L'equazione (1.3) può quindi essere scritta come:

$$E(u, v) \cong [u, v] M(x_0, y_0) \begin{bmatrix} u \\ v \end{bmatrix} \quad (1.4)$$

dove $M(x_0, y_0)$ è una matrice 2×2 relativa al pixel in posizione (x_0, y_0) definita come:

$$\begin{aligned} M(x_0, y_0) &= \sum_{x,y} w(x - x_0, y - y_0) \begin{bmatrix} I_x^2(x, y) & I_x(x, y)I_y(x, y) \\ I_x(x, y)I_y(x, y) & I_y^2(x, y) \end{bmatrix} \\ &= \begin{bmatrix} \sum_{x,y} w(x - x_0, y - y_0) I_x^2(x, y) & \sum_{x,y} w(x - x_0, y - y_0) I_x(x, y)I_y(x, y) \\ \sum_{x,y} w(x - x_0, y - y_0) I_x(x, y)I_y(x, y) & \sum_{x,y} w(x - x_0, y - y_0) I_y^2(x, y) \end{bmatrix} \end{aligned}$$

I quattro elementi della matrice $M(x_0, y_0)$ sono scalari, essendo ciascuno di essi la somma pesata delle singole intensità di ogni pixel all'interno della finestra $w(x - x_0, y - y_0)$. $I_x(x, y)$ e $I_y(x, y)$ sono elevate al quadrato, in modo da evidenziare al meglio le variazioni di intensità, e $I_x(x, y)I_y(x, y)$ è il prodotto puntuale fra $I_x(x, y)$ e $I_y(x, y)$. In figura 1.16 sono rappresentati i gradienti in ogni pixel per tre ipotetiche finestre considerate, una contenente un bordo, una contenente uno spigolo e una contenente una zona piatta. Il gradiente $\nabla I(x_0, y_0)$ nel generico pixel in (x_0, y_0) è definito come il vettore che ha come elementi $I_x(x_0, y_0)$ e $I_y(x_0, y_0)$. Analizzando i pixel all'interno della finestra $w(x + x_0, y + y_0)$, se entrambi gli elementi del gradiente sono piccoli per la maggior parte

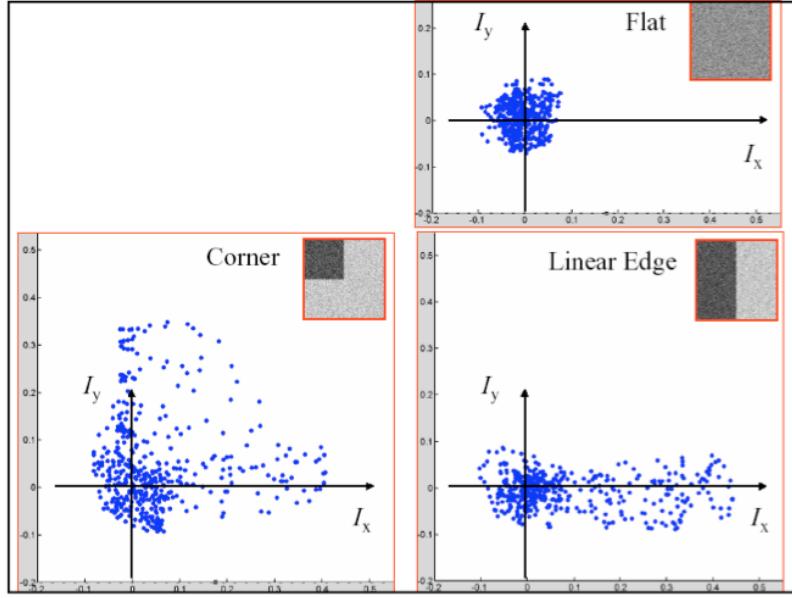


Figura 1.16: Distribuzione di I_x e I_y , tratta da [4].

dei pixel, ci si trova in una zona piatta senza alcuna variazione in tutte le direzioni, se entrambi gli elementi del gradiente sono grandi per buona parte dei punti, la finestra conterrà uno spigolo, se uno dei due elementi del gradiente è molto maggiore dell'altro per buona parte dei punti la finestra conterrà un bordo [7].

Si vuole ora capire come sia possibile ricavare una misura della presenza o meno di uno spigolo. Prima di trasformare nella forma matriciale l'equazione (1.1), si era ricavata l'espressione di $E(u, v)$ come:

$$E(u, v) \approx \sum_{x,y} w(x + x_0, y + y_0) [u^2 I_x^2(x, y) + 2uv I_x(x, y) I_y(x, y) + v^2 I_y^2(x, y)].$$

$E(u, v)$ è l'equazione di un paraboloide ellittico. A seconda dei valori di $I_x(x, y)$ e $I_y(x, y)$ all'interno della finestra $w(x + x_0, y + y_0)$ il paraboloide avrà una forma diversa, come è chiaro in figura 1.17. Se si pone $E(u, v)$ uguale a una costante è come se si

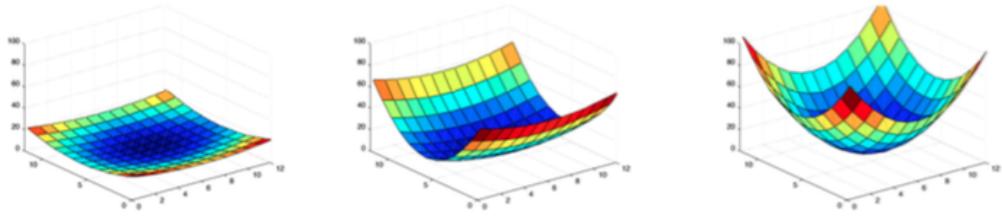


Figura 1.17: Paraboloidi associati a spigoli, bordi e zone piatte, tratta da [8].

tagliasse il paraboloide con un piano trasversale che, indifferentemente dal valore della costante, sarà un’ellisse con lo stesso rapporto tra semiasse maggiore e minore. Se si immagina di tagliare con piani orizzontali i tre paraboloidi di figura 1.17, si intuisce che si otterranno equazioni di ellissi con valori dei semiassi differenti a seconda che sia una zona piatta, di bordo o un spigolo. Se si riuscisse a trovare una relazione tra la matrice $M(x_0, y_0)$ e queste ellissi si otterrebbe una descrizione accurata della natura del pixel in (x_0, y_0) . Essendo $M(x_0, y_0)$ una matrice simmetrica, per il teorema spettrale [9], essa è diagonalizzabile. Se la si diagonalizza, partendo dall’equazione (1.4), non prendendo in considerazione la convoluzione con la finestra gaussiana $w(x + x_0, y + y_0)$, in questo momento non necessaria per il concetto che si vuole esprimere, si ottiene:

$$M(x_0, y_0) = \begin{bmatrix} e_1 & e_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} e_1^T \\ e_2^T \end{bmatrix}$$

dove e_1 e e_2 sono gli autovettori associati agli autovalori λ_1 e λ_2 . Sfruttando questa diagonalizzazione, possiamo esprimere la variazione di intensità per uno spostamento (u, v) :

$$\begin{aligned}
E(u, v) &= \begin{bmatrix} u & v \end{bmatrix} M(x_0, y_0) \begin{bmatrix} u \\ v \end{bmatrix} \\
&= \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} e_1 & e_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} e_1^T \\ e_2^T \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \\
&\stackrel{(a)}{=} \begin{bmatrix} u' & v' \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} u' \\ v' \end{bmatrix} \\
&\stackrel{(b)}{=} \lambda_1 u'^2 + \lambda_2 v'^2
\end{aligned} \tag{1.5}$$

dove nel passaggio (a) si sono definite le nuove coordinate (u', v') che rappresentano le coordinate (u, v) nel sistema di riferimento (e_1, e_2) . La trasformazione è una rotazione, che non influisce in alcun modo sul rapporto fra i semiassi delle ellissi ottenute tagliando il piano con un piano orizzontale. In (b) si sono svolte delle semplici operazioni di moltiplicazione fra matrici. Supponendo quindi $E(u, v) = \text{cost}$, in modo da tagliare il nostro paraboloide con un piano orizzontale, e che (e_1, λ_1) e (e_2, λ_2) siano le coppie di autovalori e autovettori della matrice $M(x_0, y_0)$, è possibile esprimere l'equazione (1.5) come:

$$\begin{aligned}
\frac{\lambda_1 u'^2}{E(u, v)} + \frac{\lambda_2 v'^2}{E(u, v)} &= 1 \\
\frac{u'^2}{\frac{E(u, v)}{\lambda_1}} + \frac{v'^2}{\frac{E(u, v)}{\lambda_2}} &= 1 \\
\frac{u'^2}{\left(\sqrt{\frac{E(u, v)}{\lambda_1}}\right)^2} + \frac{v'^2}{\left(\sqrt{\frac{E(u, v)}{\lambda_2}}\right)^2} &= 1 \\
\frac{u'^2}{a^2} + \frac{v'^2}{b^2} &= 1
\end{aligned} \tag{1.6}$$

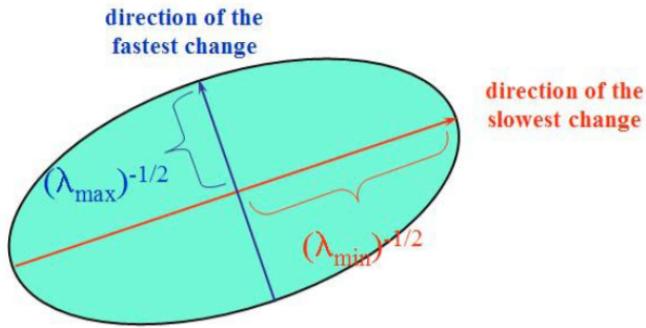


Figura 1.18: Ellisse per un generico valore costante di $E(u, v)$, tratta da [10].

dove, nell'equazione (1.6), $a = \sqrt{\frac{E(u,v)}{\lambda_1}}$ e $b = \sqrt{\frac{E(u,v)}{\lambda_2}}$ sono i semiassi dell'ellisse.

Da questo si intuisce che il semiasse associato all'autovalore maggiore sarà minore di quello associato all'autovalore minore, vigendo una relazione inversa. Questo è visibile in figura 1.18, dove è rappresentato l'ellisse per un generico valore costante di $E(u, v)$, con le due coppie di autovalori e autovettori associati.

Ad un generico pixel in (x_0, y_0) è quindi associabile una matrice $M(x_0, y_0)$ diagonalizzabile, i cui autovalori sono inversamente proporzionali ai valori dei semiassi.

Quindi:

- una matrice $M(x_0, y_0)$ con un autovalore piccolo e uno grande, rappresenterà un'ellisse schiacciata che indicherà una variazione solo nella direzione del semiasse minore, quindi un bordo in direzione del semiasse maggiore;
- una matrice $M(x_0, y_0)$ con entrambi i valori degli autovalori piccoli rappresenterà un'ellisse con entrambi i semiassi grandi, quindi poca variazione di intensità in entrambe le direzioni, cioè una zona piatta;
- una matrice $M(x_0, y_0)$ con entrambi i valori degli autovalori grandi rappresenterà un'ellisse con entrambi i semiassi piccoli, quindi molta variazione di intensità in

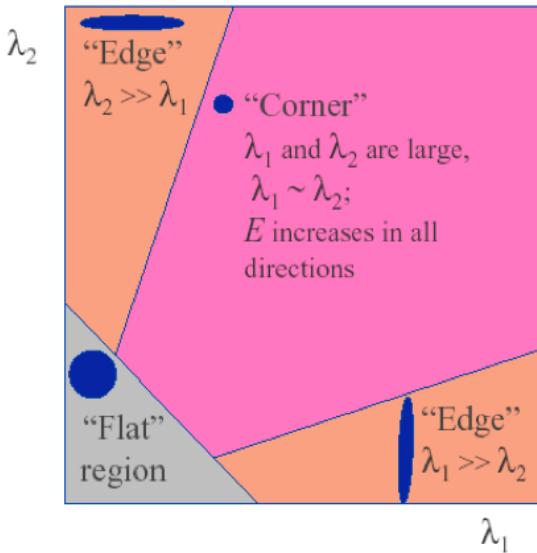


Figura 1.19: Tabella con autovalori, tratta da [4].

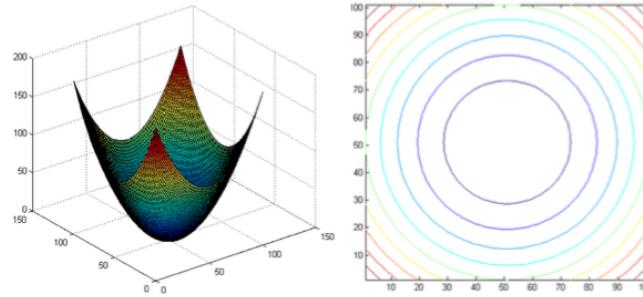
entrambe le direzioni, quindi uno spigolo.

Queste considerazioni sono rappresentate schematicamente in figura 1.19, in cui si evidenzia in blu come sarà l'ellisse per i diversi valori dei due autovalori.

Per comprendere meglio tutto questo, si osservi la figura 1.20: nella figura 1.20a è rappresentato il paraboloida che si potrebbe associare ad un candidato spigolo (si trovano circonferenze concentriche e non ellissi perché in questo caso gli autovalori sono uguali, quindi un'ellisse con assi uguali è una circonferenza). Le circonferenze concentriche sono i valori della circonferenza al variare di $E(u, v)$. Più $E(u, v)$ cresce più il raggio della circonferenza cresce, ma il rapporto tra gli autovalori rimane costante. Nelle tre immagini in figura 1.20 si vede in che modo il grafico bidimensionale e tridimensionale di $E(u, v)$ varino in relazione alla matrice $M(x_0, y_0)$, espressa in figura come A poiché tratta da [8]. In figura 1.20a si vede cosa succede di fronte ad un candidato spigolo, quindi nella situazione in cui entrambi gli autovalori siano grandi. In

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$

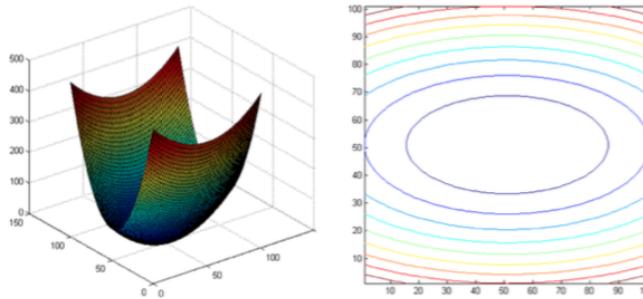
Eigenvalues
Eigenvectors Eigenvalues
Eigenvectors



(a)

$$\mathbf{A} = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$

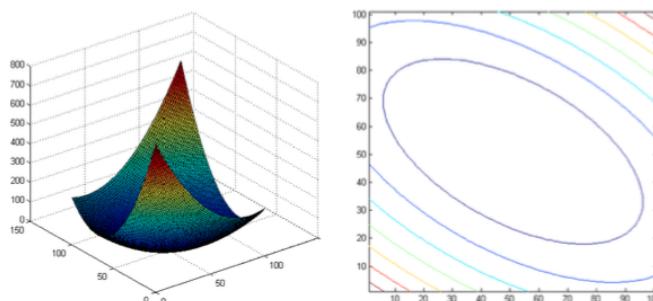
Eigenvalues
Eigenvectors Eigenvalues
Eigenvectors



(b)

$$\mathbf{A} = \begin{bmatrix} 3.25 & 1.30 \\ 1.30 & 1.75 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T$$

Eigenvalues
Eigenvectors Eigenvalues
Eigenvectors



(c)

Figura 1.20: Esempi di $E(u, v)$, tratta da [8].

figura 1.20b invece si vede cosa succede di fronte ad un candidato bordo, quindi nella situazione in cui un autovalore sia maggiore dell'altro. La figura 1.20c infine rappresenta lo stesso paraboloide di figura 1.20b ma rotato a causa degli autovettori associati agli autovalori, non più unitari come in figura 1.20a e 1.20b. In questo caso quello che si vuole sottolineare è che nonostante la rotazione il rapporto fra i due autovalori rimane lo stesso, a prova che l'operatore di Harris è invariante alla rotazione.

Una volta capito come individuare gli spigoli in un'immagine I , è necessario trovare una misura di qualità R per classificare e selezionare i migliori, cioè quelli contraddistinti dalle tre proprietà associate ad un buon candidato spigolo, elencate all'inizio di questo capitolo. La formulazione proposta da Harris è:

$$R = \text{Det}[M(x_0, y_0)] - k\text{Tr}[M(x_0, y_0)]^2$$

dove k è una costante dal valore convenzionale pari a 0.04. La scelta di utilizzare determinante e traccia della matrice $M(x_0, y_0)$ è fondamentale, perché esula dalla diagonalizzazione di $M(x_0, y_0)$ e dall'utilizzo dei suoi autovalori, processo molto macchinoso e di difficile implementazione. Seguendo questa definizione, il valore di R sarà negativo nei pixel sui bordi, positivo e grande nei pixel sugli spigoli e positivo e piccolo nei pixel nelle zone piatte. Una volta trovati i valori di R di ogni pixel dell'immagine, si sceglie una soglia e si selezionano tutti i pixel con valore di R associato superiore a quella soglia e che sia inoltre massimo locale.

Fino ad ora si è trattato l'algoritmo di Harris applicando il procedimento ad ogni singolo pixel. Per svolgere l'esempio seguente è necessario utilizzare una differente notazione, quella implementata in Matlab, il software da noi utilizzato per compiere tutti i passaggi e mostrare le figure dei vari componenti. La definizione di $I_x(x, y)$ e

$I_y(x, y)$ non varia rispetto a quella utilizzata finora, le due immagini vengono ottenute effettuando la convoluzione dell'immagine I con il vettore $\begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$ per la direzione

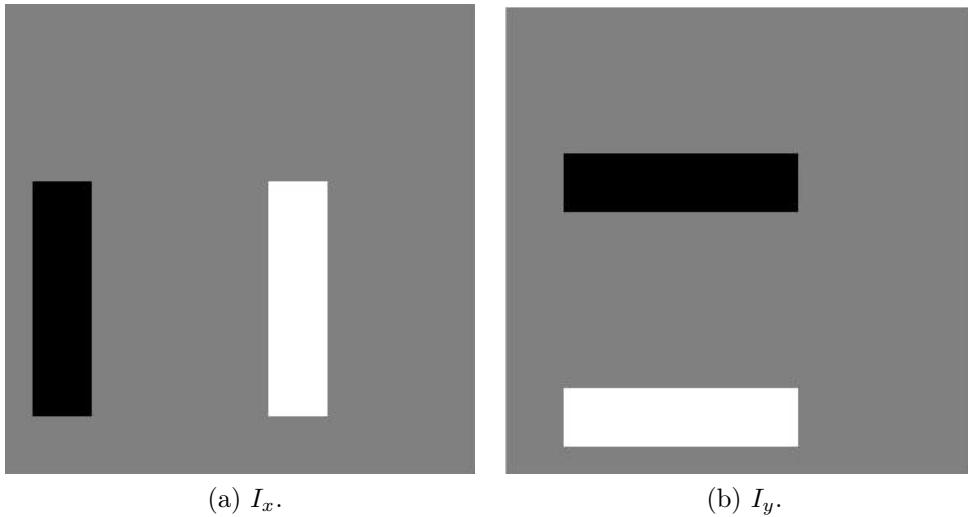
x , e il vettore $\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$ per la direzione y . Per comodità e chiarezza da qui in poi $I_x(x, y)$ e

$I_y(x, y)$ verranno chiamate semplicemente I_x e I_y . Se prima si era definita una matrice $M(x, y)$ di dimensione 2×2 per ogni pixel dell'immagine, si definisce ora una matrice \mathbf{M} di dimensione $2n \times 2n$ unica per l'intera immagine, di dimensione $n \times n$:

$$\begin{aligned} \mathbf{M} &= w_{d \times d}(x, y) \otimes \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \\ \mathbf{M} &= \begin{bmatrix} w_{d \times d}(x, y) \otimes I_x^2 & w_{d \times d}(x, y) \otimes I_x I_y \\ w_{d \times d}(x, y) \otimes I_x I_y & w_{d \times d}(x, y) \otimes I_y^2 \end{bmatrix} \end{aligned} \quad (1.7)$$

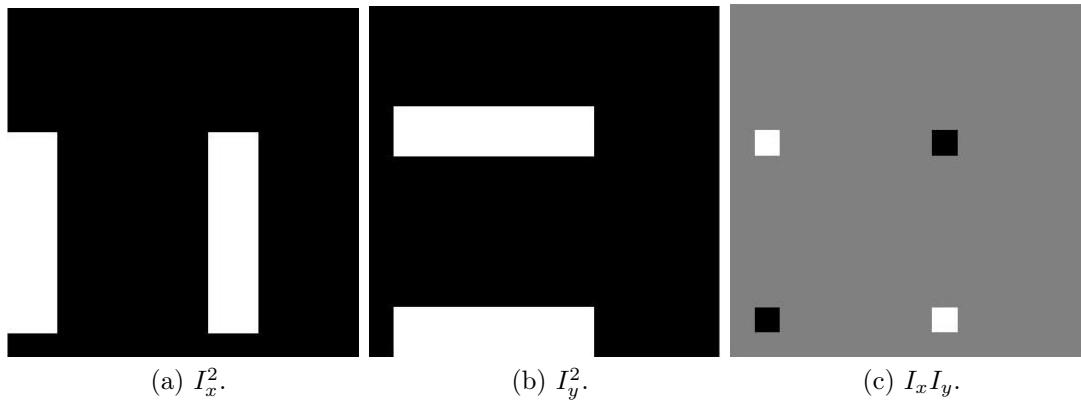
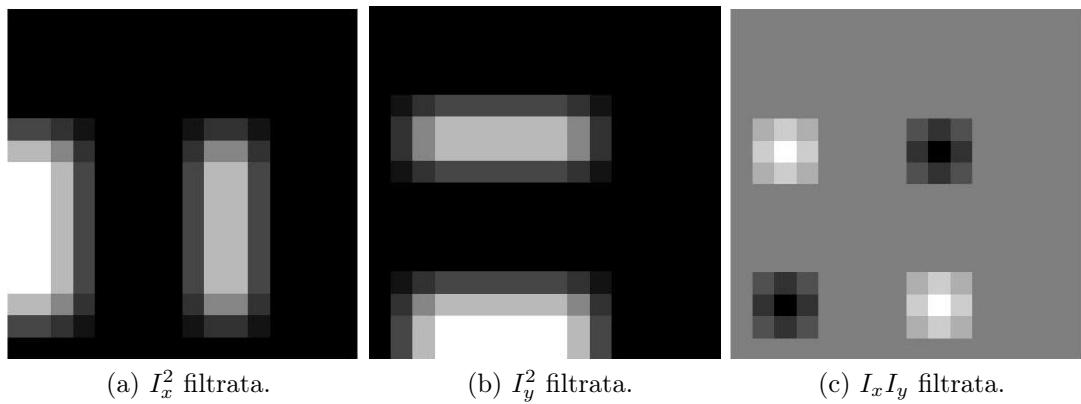
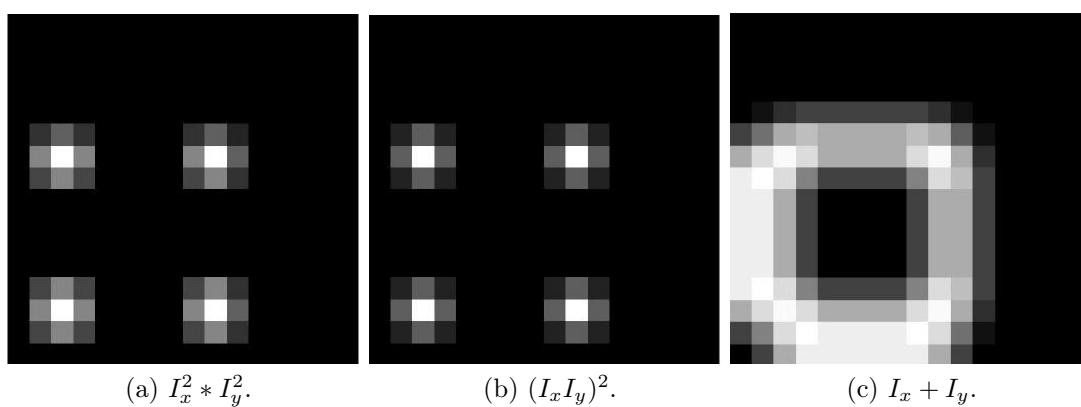
dove $w_{d \times d}(x, y)$ è una finestra gaussiana troncata alla dimensione $d \times d$ e l'operatore \otimes è il simbolo di convoluzione. In Matlab quindi la pesatura viene effettuata con una convoluzione di ciascuna delle tre immagini con $w_{d \times d}(x, y)$. Se prima i quattro elementi della matrice $M(x, y)$ erano scalari, ora ciascun elemento è un'immagine di dimensione $n \times n$. Allo stesso modo viene definita una matrice \mathbf{R} di dimensione $n \times n$ in cui ad ogni coppia (x, y) è associato il valore scalare di $R(x, y)$. Riassumendo e applicando il procedimento all'immagine in figura 1.10 come fatto per Moravec, utilizzando una finestra con $d = 5$, l'algoritmo di Harris si può descrivere attraverso i seguenti passaggi:

1. data un'immagine I , calcolare le derivate I_x e I_y nelle due direzioni con $I_x =$

Figura 1.21: I_x e I_y .

$$I \otimes \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \text{ e } I_y = I \otimes \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \quad (\text{figura 1.21});$$

2. calcolare I_x^2 , I_y^2 e $I_x I_y$ (figura 1.22);
3. per ciascuna delle tre immagini, calcolare la convoluzione con la finestra $w_{d \times d}(x, y)$ gaussiana (figura 1.23), come mostrato nell'equazione 1.7;
4. calcolare $\mathbf{R} = \text{Det}(M) - k \text{Tr}(M)^2 = [(I_x^2 * I_y^2) - (I_x I_y)^2] - k * (I_x + I_y)$, i singoli componenti sono rappresentati in figura 1.24, il risultato \mathbf{R} invece in figura 1.25;
5. sperimentalmente, si trova che valori alti e massimi locali di \mathbf{R} indicano la presenza di spigoli, al contrario valori molto negativi di \mathbf{R} indicano la presenza di bordi.

Figura 1.22: I_x^2 , I_y^2 , I_xI_y .Figura 1.23: Filtraggio I_x^2 , I_y^2 , I_xI_y .Figura 1.24: Componenti di \mathbf{R} .

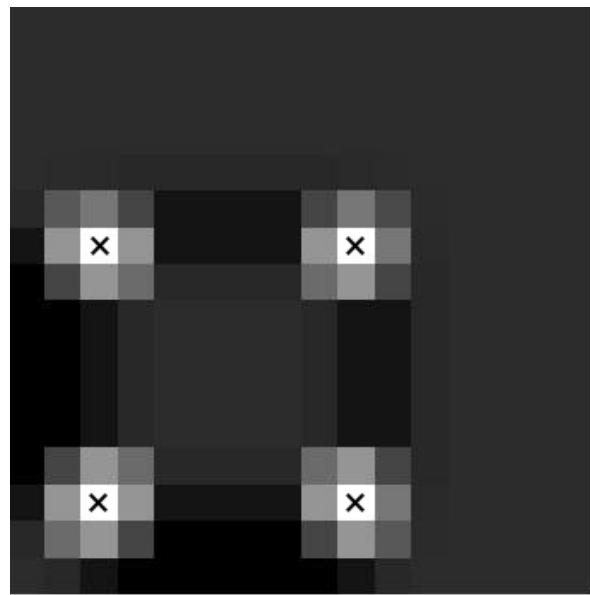


Figura 1.25: **R**.

Osservando quindi la figura 1.25, vediamo che i punti di massimo, messi in evidenza, sono effettivamente gli spigoli dell'immagine di partenza e, come sottolineato prima, corrispondono a pixel con valori di R molto alti.

Capitolo 2

Algoritmo per l'affinamento dell'estrazione di spigoli

Finora si è descritta la teoria dell'algoritmo di Harris. In questo capitolo si cercherà di migliorare e affinare la precisione di questo algoritmo. Come già detto, l'algoritmo di Harris lavora a livello di pixel, associa cioè coordinate intere agli spigoli di un'immagine. E' possibile usare tecniche di elaborazione delle immagini per affinare la precisione ottenuta con l'algoritmo di Harris e individuare le coordinate degli spigoli a "livello sub-pixel", ovvero, identificare delle coordinate frazionarie del punto notevole. Tra i numerosi algoritmi per l'identificazione a livello sub-pixel, quello implementato da Matlab, il software qui utilizzato per l'analisi, si basa sul concetto di "curve fitting", cioè sull'utilizzo di una quadrica che viene adattata (in inglese 'fitted') nei punti dell'intorno del candidato spigolo, il punto di minimo o massimo di questa quadrica è lo spigolo a livello sub-pixel [11].

Un altro possibile algoritmo a livello sub-pixel è quello proposto da Z. M. Liang nel 2006 [12] (in lingua cinese) e da noi analizzato sulla base di [13]. Cerchiamo di

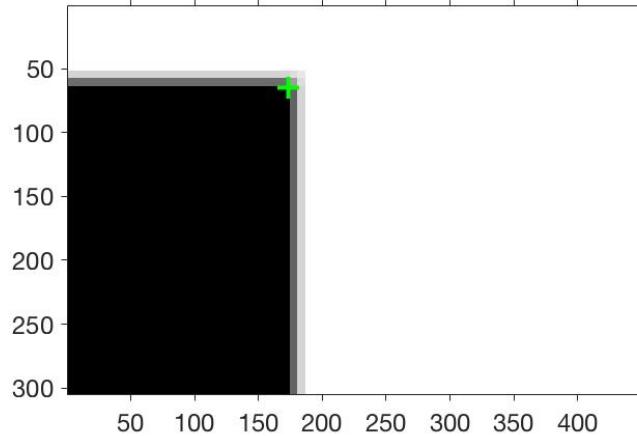


Figura 2.1: Spigolo individuato dall'algoritmo di Harris.

spiegare il funzionamento di questo algoritmo partendo dallo spigolo con coordinate intere trovato dall'algoritmo di Harris, secondo il procedimento illustrato nel capitolo 1, sull'immagine rappresentata in figura 2.1. Lavoriamo in un intorno di questo punto, \mathbf{c} , che indichiamo come $\mathbf{c} = [c_x, c_y]^T$. Ciascun punto in questo intorno, che chiameremo \mathbf{p} , si troverà o su un bordo o su una zona piatta. Indichiamo ora con \mathbf{q} lo spigolo cercato a livello sub-pixel, quindi con coordinate non intere e per ora incognite. Ricordando che il prodotto scalare tra due generici vettori colonna \mathbf{a} e \mathbf{b} è $\mathbf{a}^T \cdot \mathbf{b} = 0$ se o \mathbf{a} o \mathbf{b} sono nulli o se i due vettori sono perpendicolari. Dato un generico punto \mathbf{p} nell'intorno di \mathbf{c} , possiamo considerare il prodotto scalare

$$\nabla \mathbf{p}^T (\mathbf{q} - \mathbf{p}) \quad (2.1)$$

dove $\nabla \mathbf{p}^T$ è la trasposta del gradiente nel punto \mathbf{p} e $\mathbf{q} - \mathbf{p}$ è il vettore differenza tra \mathbf{q} e \mathbf{p} . Il gradiente in \mathbf{p} è definito come il vettore che ha come elementi le derivate direzionali nel punto \mathbf{p} , cioè quelle che, rifacendoci alla notazione utilizzata nel capitolo 1 per un'immagine I , chiameremmo $I_x(\mathbf{p})$ e $I_y(\mathbf{p})$. Dunque $\nabla \mathbf{p} = [I_x(\mathbf{p}), I_y(\mathbf{p})]^T$. Se \mathbf{p}

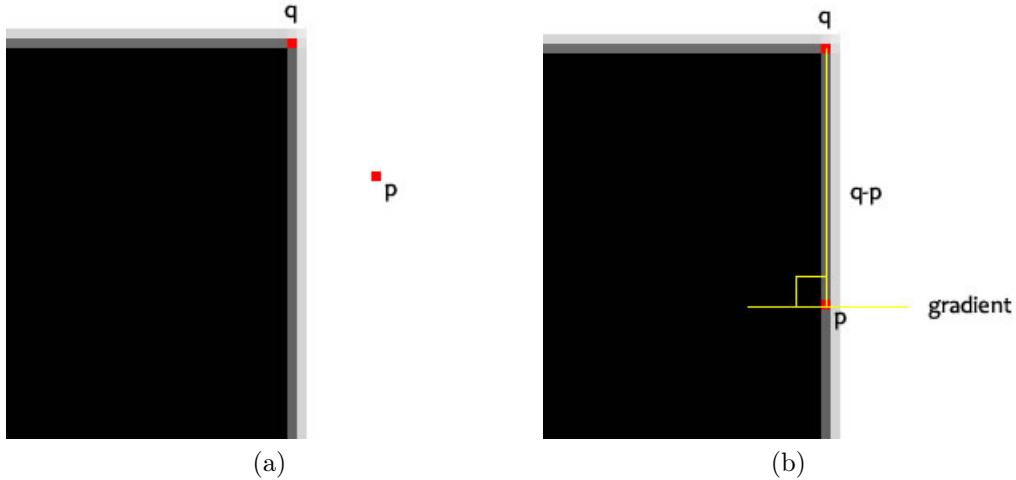


Figura 2.2: Punto \mathbf{p} in una zona piatta (a) e in una zona di bordo (b), tratta da [14].

si trova in una zona piatta come in figura 2.2a, allora $\nabla \mathbf{p}$ sarà uguale a 0, e quindi anche il prodotto scalare definito nell’equazione (2.1) sarà 0. Se invece \mathbf{p} si trova su un bordo come in figura 2.2b, l’equazione (2.1) risulterà comunque 0, data la perpendicolarità tra $\nabla \mathbf{p}$ e il vettore differenza $\mathbf{q} - \mathbf{p}$. Questa osservazione suggerisce di considerare equazioni nella forma $\nabla \mathbf{p}_n^T (\mathbf{q} - \mathbf{p}_n) = 0$, in cui \mathbf{p}_n sono punti in un intorno di \mathbf{c} . Tali equazioni possono essere poste come

$$\nabla \mathbf{p}_n^T \mathbf{q} = \nabla \mathbf{p}_n^T \mathbf{p}_n$$

Premoltiplicando ora entrambi i membri per il vettore $\nabla \mathbf{p}_n$ si ottiene

$$(\nabla \mathbf{p}_n \nabla \mathbf{p}_n^T) \mathbf{q} = (\nabla \mathbf{p}_n \nabla \mathbf{p}_n^T) \mathbf{p}_n$$

in cui le parentesi evidenziano che i membri dell’equazione possono essere rispettiva-

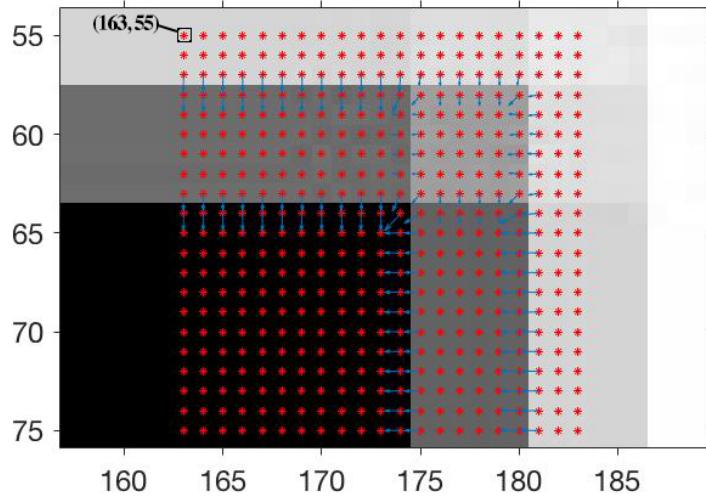


Figura 2.3: Rappresentazione dei pixel considerati con rispettivo gradiente.

mente interpretati come prodotti riga colonna fra la matrice $\nabla \mathbf{p}_n \nabla \mathbf{p}_n^T$ e i vettori \mathbf{q} e \mathbf{p}_n . Considerando N punti \mathbf{p}_n , tali equazioni devono essere verificate per tutti questi punti. Possiamo dunque sommare tali equazioni sull'indice n ottenendo:

$$\sum_n \nabla \mathbf{p}_n \nabla \mathbf{p}_n^T \mathbf{q} = \sum_n \nabla \mathbf{p}_n \nabla \mathbf{p}_n^T \mathbf{p}_n.$$

Premoltiplicando entrambi i membri dell'equazione per la matrice inversa $(\sum_n \nabla \mathbf{p}_n \nabla \mathbf{p}_n^T)^{-1}$ si ottiene:

$$\mathbf{q} = (\sum_n \nabla \mathbf{p}_n \nabla \mathbf{p}_n^T)^{-1} \sum_n \nabla \mathbf{p}_n \nabla \mathbf{p}_n^T \mathbf{p}_n. \quad (2.2)$$

Considerando un sistema di assi cartesiani in cui l'origine è in alto a sinistra, l'algoritmo di Harris individua lo spigolo in $[173, 65]$, ovvero, $\mathbf{c}_x = 173$ e $\mathbf{c}_y = 65$, come mostrato in figura 2.1. Scegliamo una finestra 21×21 nell'intorno di questo spigolo, perciò $N = 441$ è il numero di punti che consideriamo. Si noti che degli $N = 441$ punti dell'intorno uno corrisponde a \mathbf{c} . In figura 2.3 sono rappresentati i 441 pixel e i

relativi gradienti, evidenziati da frecce blu, nell'intorno della zona in cui si trova lo spigolo. L'algoritmo prende in considerazione dei punti appartenenti all'intorno definito e calcola il valore di \mathbf{q} attraverso l'equazione (2.2).

Per illustrare il funzionamento dell'algoritmo, consideriamo i punti \mathbf{p}_n in figura 2.3 a partire dal punto evidenziato, cioè quello in alto a sinistra con coordinate [163, 55], incrementando via via il valore di N . Per $N = 1$

$$\mathbf{q} = (\nabla \mathbf{p}_{[163,55]} \nabla \mathbf{p}_{[163,55]}^T)^{-1} \nabla \mathbf{p}_{[163,55]} \nabla \mathbf{p}_{[163,55]}^T \mathbf{p}_{[163,55]}. \quad (2.3)$$

Dato che in questo caso il gradiente nel punto è prossimo a zero la (2.3) perde di significato, in particolare l'inversa non è definita. Lo stesso accade quando consideriamo il secondo punto, [164, 55] ed $N = 2$

$$\begin{aligned} \mathbf{q} = & (\nabla \mathbf{p}_{[163,55]} \nabla \mathbf{p}_{[163,55]}^T + \nabla \mathbf{p}_{[164,55]} \nabla \mathbf{p}_{[164,55]}^T)^{-1} \\ & \cdot (\nabla \mathbf{p}_{[163,55]} \nabla \mathbf{p}_{[163,55]}^T \mathbf{p}_{[163,55]} + \nabla \mathbf{p}_{[164,55]} \nabla \mathbf{p}_{[164,55]}^T \mathbf{p}_{[164,55]}) \end{aligned}$$

poiché anche in questo caso il gradiente nel punto è prossimo a zero. Considerando alcuni punti \mathbf{p}_n sulla prima riga, il primo valore di \mathbf{q} che si riesce a determinare è quello evidenziato in figura 2.4, in [168, 55]. Considerando poi ulteriori punti \mathbf{p}_n , il risultato del calcolo al crescere di N è mostrato dalla successione di punti in figura 2.4.

In figura 2.5 vengono messe in evidenza tre misure: in verde lo spigolo individuato dall'algoritmo di Harris (coordinate [173, 65]), in rosso quello migliorato con la tecnica di 'curve fitting' implementata in Matlab (coordinate [173.49514, 64.63066]) e in blu quello individuato attraverso l'algoritmo analizzato (coordinate [173.83724, 64.41271]). Si nota quindi un buon miglioramento della misura rispetto al pixel individuato dal-

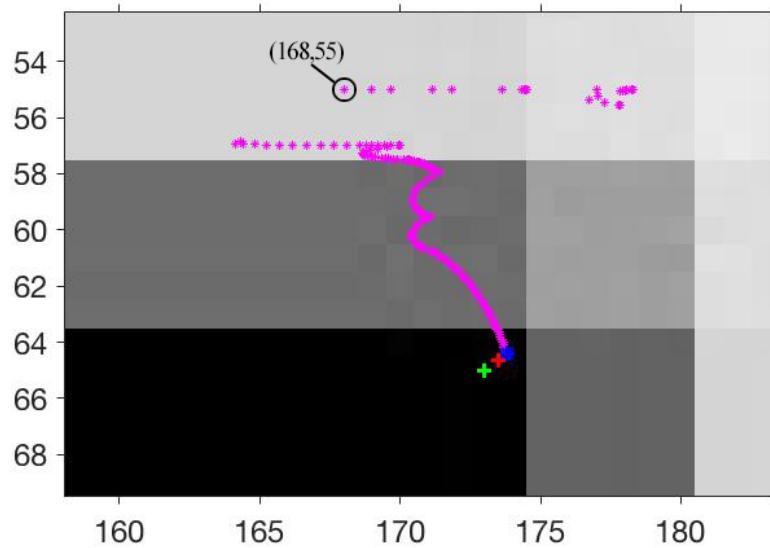


Figura 2.4: Valori di \mathbf{q} al variare di N .

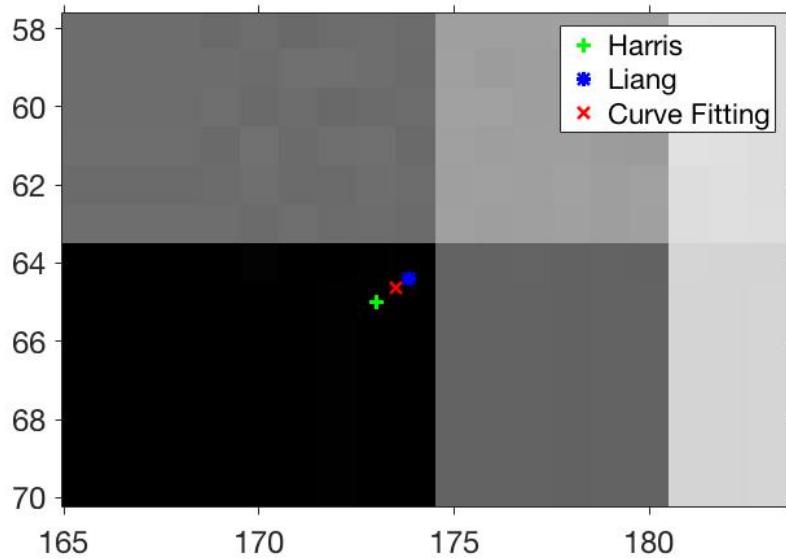


Figura 2.5: Tre misure di spigolo individuate.

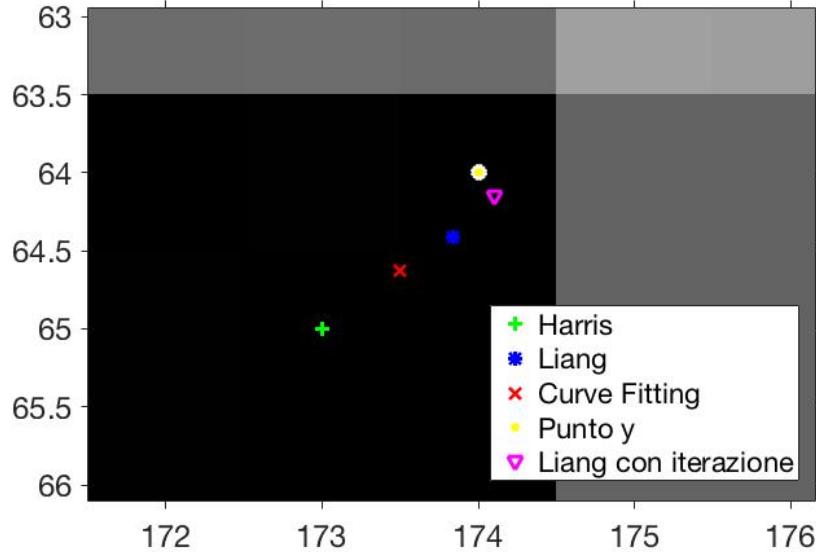


Figura 2.6: Quattro misure di spigolo individuate.

l’algoritmo di Harris. In base ai nostri esperimenti, è molto importante il numero di punti considerati nell’intorno: un numero eccessivo di punti considerati porterà a una grossa perdita di precisione, poiché verrano considerati pixel lontani dallo spigolo e quindi fuorvianti nell’identificazione di questo.

Introduciamo ora un nuovo concetto. Approssimiamo il valore a livello sub-pixel trovato $[173.83724, 64.41271]$ con le coordinate intere più vicine, quindi $[174, 64]$, e chiamiamo questo punto \mathbf{y} . Una volta fatto ciò ripetiamo tutto il procedimento considerando queste coordinate come se fossero le coordinate di partenza individuate dall’algoritmo di Harris. Consideriamo quindi i pixel nell’intorno di \mathbf{y} e andiamo a calcolare il nuovo valore di \mathbf{q} . Si osservi la figura 2.6, dove in giallo è rappresentato il punto \mathbf{y} , e in magenta il nuovo spigolo con precisione a livello sub-pixel, con coordinate $[174.10658, 64.14371]$. Siamo quindi riusciti ad ottenere un miglioramento ulteriore nella precisione in termini di vicinanza con il punto ideale, identificabile visivamente.

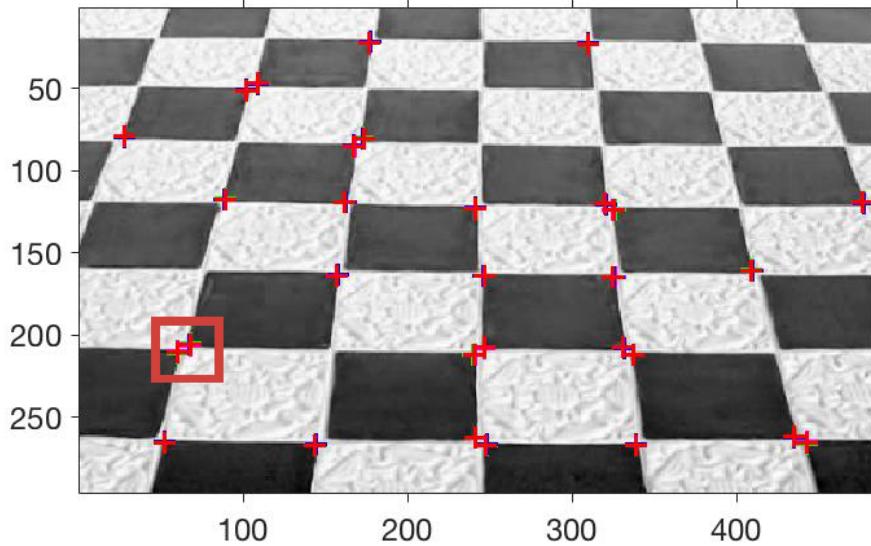


Figura 2.7: Scacchiera.

Questo procedimento si può ripetere in modo iterativo, ma si è osservato che il valore di \mathbf{q} si assesta quasi sempre dopo una sola iterazione.

In figura 2.7 e nell'ingrandimento in figura 2.8 sono mostrati i risultati ottenuti applicando l'algoritmo ad un'immagine diversa, rappresentante una scacchiera. Anche in questo caso si può notare un miglioramento grazie all'algoritmo di affinamento analizzato. A volte però i risultati peggiorano rispetto all'algoritmo originale di Harris, come si può vedere nella stella rappresentata in figura 2.9, in cui abbiamo utilizzato un'immagine molto "sporca". Osservando gli ingrandimenti in figura 2.10, relativi alle zone evidenziate in figura 2.9, si osservano risultati differenti. Vediamo che in 2.10a l'algoritmo analizzato funziona bene, in 2.10b invece l'algoritmo sposta le coordinate più lontane dallo spigolo.

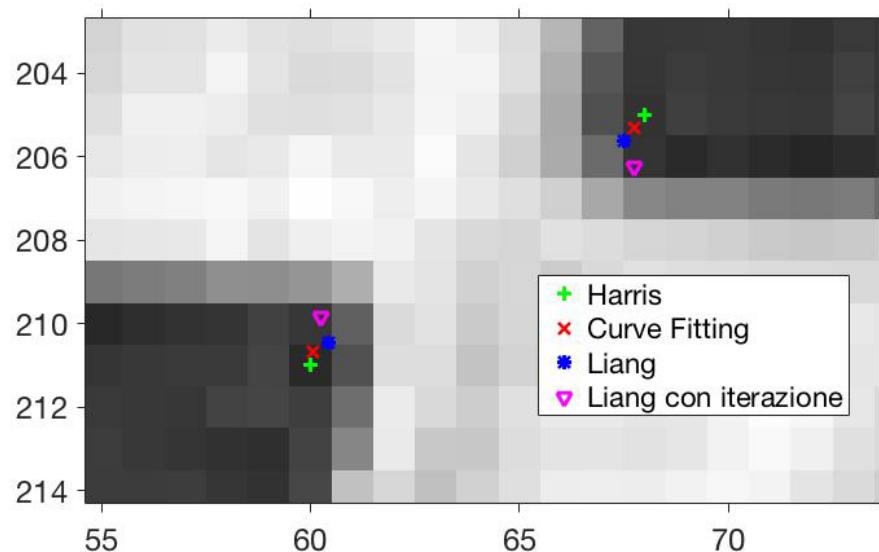


Figura 2.8: Ingrandimento scacchiera.

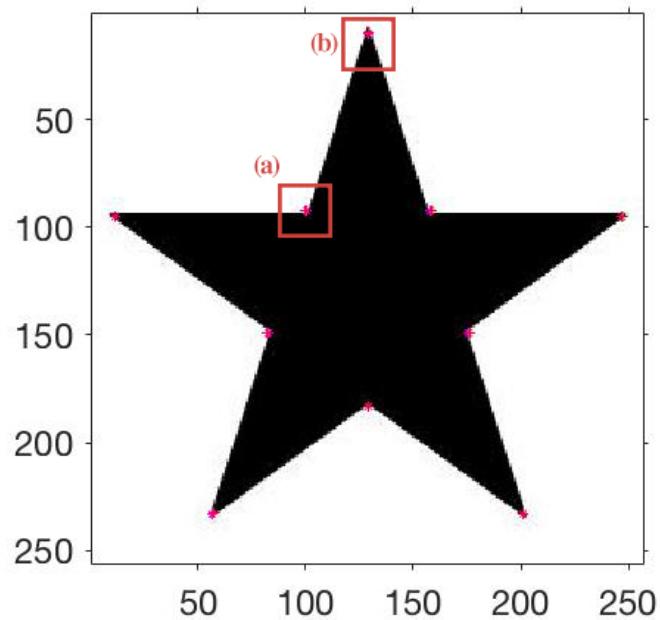
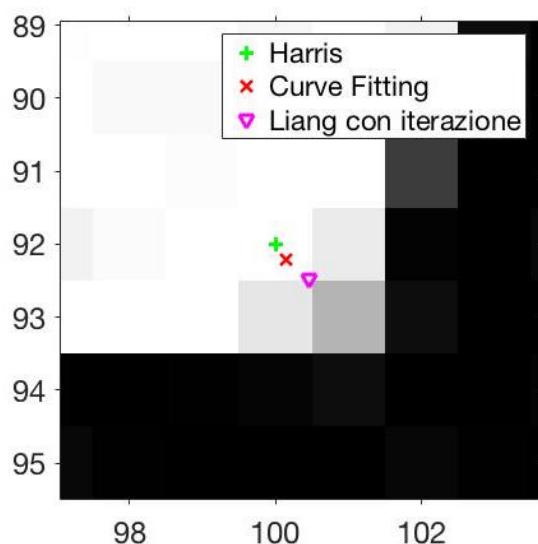
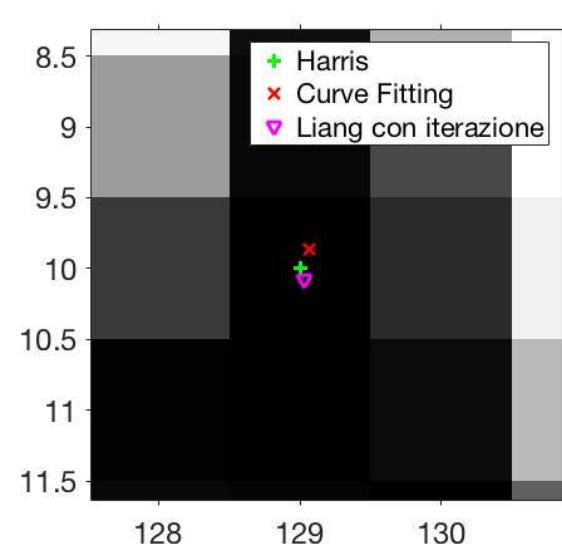


Figura 2.9: Stella.



(a) Ingrandimento finestra (a).



(b) Ingrandimento finestra (b).

Figura 2.10: Ingrandimento delle due finestre evidenziate in figura 2.9.

Conclusioni

In questo lavoro abbiamo studiato l'estrazione di spigoli in immagini, processo fondamentale per numerose applicazioni nel campo dell'elaborazione delle immagini. Dopo aver parlato dell'algoritmo di Moravec, un algoritmo di estrazione su cui Harris e Stephens lavorarono direttamente per realizzare l'algoritmo oggetto di questa tesi, ci siamo concentrati sulla teoria intorno all'algoritmo di Harris stesso. Studiando questo algoritmo, abbiamo dimostrato la sua efficacia nell'individuazione di spigoli in un'immagine in scala di grigi.

Per cercare di migliorare questo algoritmo, ci siamo focalizzati sul fatto che le coordinate dello spigolo trovate dall'algoritmo di Harris siano intere, quindi possono essere rese più precise con un qualche algoritmo che lavora a “livello sub-pixel”, come per esempio l'algoritmo utilizzato da Matlab, che si basa sul concetto di “curve fitting”. Per questo, abbiamo cercato di affinare il valore delle coordinate dello spigolo individuato dall'algoritmo di Harris. Per fare ciò abbiamo studiato e implementato un algoritmo preesistente che lavora su un determinato numero di punti N in un intorno di questo spigolo. Abbiamo provato questo algoritmo su diverse immagini e al variare di N e nella maggior parte dei casi abbiamo notato un miglioramento nella precisione.

Dopo aver studiato l'efficacia di questo metodo, abbiamo cercato di affinare ulteriormente i risultati. Una volta individuato lo spigolo a livello sub-pixel, abbiamo

ripetuto il processo iniziale in modo iterativo, prendendo come coordinate di partenza le coordinate intere più vicine a quelle decimali dello spigolo a livello sub-pixel. Anche in questo caso si sono ottenuti buoni risultati, spesso anche più precisi dei precedenti.

Bibliografia

- [1] C. Harris and M. Stephens, “A combined corner and edge detector.” in *Proc. 4th Alvey Vision Conference*, vol. 15, 1988, p. 50.
- [2] H. P. Moravec, “Obstacle avoidance and navigation in the real world by a seeing robot rover.” Stanford University DTIC Document, Tech. Rep., 1980.
- [3] R. Fergus, “Corners, Blobs and Descriptors,” NYU Computer Science, 2008, Disponibile (scaricato nel 09/2016). [Online]. Available: http://cs.nyu.edu/~fergus/teaching/vision_2012/3_Corners_Blobs_Descriptors.pdf
- [4] R. Collins, “CSE486 Computer Vision I,” The Pennsylvania State University - Department of Computer Science & Engineering, May 2007, Disponibile (scaricato nel 09/2016). [Online]. Available: <http://www.cse.psu.edu/~rtc12/CSE486/>
- [5] F. Frassinelli and D. Ducco, “Studio e sviluppo di sistemi di odometria visuale per esplorazione planetaria,” University of Genova - Department of Information Engineering, Tech. Rep., May 2010, Disponibile (scaricato nel 09/2016). [Online]. Available: <http://www.graal.dibris.unige.it/files/34>
- [6] C. Schmid, R. Mohr, and C. Bauckhage, “Evaluation of interest point detectors,” *International Journal of computer vision*, vol. 37, no. 2, pp. 151–172, 2000.

- [7] E. Agu, “Digital Image Processing (CS/ECE 545), Lecture 5: Edge Detection and Corner Detection,” Carnegie Mellon University, 2014, Disponibile (scaricato nel 09/2016). [Online]. Available: <http://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/slides/lecture04.pdf>
- [8] K. Kitani, “Detecting Corner - 16-385 Computer Vision I,” Carnegie Mellon University, 2015, Disponibile (scaricato nel 09/2016). [Online]. Available: <http://www.cs.cmu.edu/~16385/lectures/Lecture7.pdf>
- [9] T. Hawkins, “Cauchy and the spectral theory of matrices,” *Historia Mathematica*, vol. 2, no. 1, pp. 1–29, 1975.
- [10] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.
- [11] Z. Zhang, H. Lu, X. Li, W. Li, and W. Yuan, “Application of improved Harris algorithm in sub-pixel feature point extraction,” *International Journal of Computer and Electrical Engineering*, vol. 6, no. 2, p. 101, 2014, Disponibile (scaricato nel 09/2016). [Online]. Available: <http://search.proquest.com/openview/e58c3476725018c2ad370010e218019b/1?pq-origsite=gscholar>
- [12] Z. Liang, H. Gao, Z. Wang, and L. WU, “Sub-pixels corner detection for camera calibration,” *Transactions-China welding institution*, vol. 27, no. 2, p. 102, 2006.
- [13] A. Datta, J.-S. Kim, and T. Kanade, “Accurate camera calibration using iterative refinement of control points,” in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 1201–1208.

- [14] J. Howse, S. Puttemans, Q. Hua, and U. Sinha, *OpenCV 3 Blueprints*. Packt Publishing Ltd, 2015, Disponibile (scaricato nel 09/2016). [Online]. Available: <http://aishack.in/tutorials/subpixel-corners-increasing-accuracy/>

Ringraziamenti

Innanzitutto desidero ringraziare il professore Riccardo Raheli per avermi dato la possibilità di intraprendere questa esperienza formativa, aiutandomi e mostrando un reale interessamento al mio lavoro. Ringrazio Carlo Tripodi per la pazienza, i consigli e la disponibilità dimostrati in questi mesi. Un grazie va alla mia famiglia, a mia madre per avermi sempre appoggiato, a mio padre per avermi inconsapevolmente convinto ad intraprendere questa strada, a mia sorella per aver lavato i piatti al posto mio quando non avevo tempo, a mio zio per aver condiviso con me la sua infinita cultura e ai miei nonni per tutti i pranzi che mi hanno preparato. Grazie inoltre a Valentina, per avermi sopportato e supportato in qualsiasi momento. Grazie ai ragazzi della workstation che mi hanno accompagnato in tutto questo percorso, la mia compagna di studi Angelica, la bibbia delle telecomunicazioni Mattia, la calcolatrice scientifica Giulia, l'altissima Marta, la regina di Lyx Chiara e lo stilosissimo Silvano. Grazie infine ai miei amici, con cui non parlo da mesi poiché troppo impegnato a studiare.