



Università degli Studi di Cagliari
Facoltà di Ingegneria e Architettura
Corso di Laurea Magistrale in Ingegneria
Meccanica

**Sviluppo di un modello alternativo
alla visione stereoscopica per la
correlazione digitale d'immagine
tridimensionale**

Relatore:
Prof.ing. Antonio Baldi

Tesi di Laurea di:
Dial Sarbjot Singh
Mattar Sunny

Controrelatore:
Prof.ing. Filippo
Bertolino

A.A. 2016-2017

Indice

| | |
|--------------------------------------------------------------------------------------------|----------|
| INTRODUZIONE | 5 |
| | |
| CAPITOLO 1 –CORRELAZIONE DIGITALE D’IMMAGINE TRIDIMENSIONALE | |
| SECONDO IL MODELLO DI VISIONE STEREOSCOPICA 7 | |
| 1.1 Modello di acquisizione d’immagine per singola fotocamera..... | 7 |
| 1.1.1 Trasformazione dal sistema di riferimento globale a quello della fotocamera | 10 |
| 1.1.2 Trasformazione di proiezione centrale | 13 |
| 1.1.3 Trasformazione fra sistema del piano immagine e del sensore | 17 |
| 1.1.4 Matrice di calibrazione e di proiezione | 20 |
| 1.1.5 Modello geometrico del DIC3D | 22 |
| 1.2 Calibrazione della singola fotocamera..... | 32 |
| 1.2.1 Algoritmo Direct Linear Transform | 32 |
| 1.2.2 Algoritmo di Zang | 38 |
| 1.2.3 Modello della distorsione | 44 |
| 1.2.4 <i>Interest points detection</i> | 46 |
| 1.2.5 Esempio di calibrazione | 51 |
| 1.3 Analisi del DIC3D secondo il modello di visione stereoscopico | 54 |
| 1.3.1 Matrice fondamentale e calibrazione stereo | 55 |
| 1.3.2 Geometria epipolare | 61 |
| 1.3.3 Rettificazione delle immagini | 66 |
| 1.3.4 Template matching per il calcolo della mappa di disparità | 68 |
| | |
| CAPITOLO 2 – CORRELAZIONE DIGITALE D’IMMAGINE 2D 74 | |
| 2.1 L’ image matching e il problema dell’apertura e della corrispondenza | 74 |
| 2.1.1 Analisi differenziale del problema | 77 |
| 2.1.2 Calcolo delle derivate | 80 |
| 2.1.3 Algoritmo Horn&Schunck | 84 |

| | |
|-----------------------------------------------------------------------------------------------------------------|------------|
| 2.1.4 Algoritmo Lucas-Kanade | 88 |
| 2.1.5 Implementazione iterativa dell'algoritmo Lucas-Kanade | 94 |
| 2.1.6 Modello della compensazione della trasformazione radiometrica | 97 |
| 2.1.7 Modello della trasformazione affine | 101 |
| 2.1.8 Calcolo degli spostamenti superiori al pixel | 106 |
| 2.1.9 Esempio d'applicazione ad un caso reale | 109 |
| CAPITOLO 3 – MODELLO ALTERNATIVO PER LA CORRELAZIONE DIGITALE D'IMMAGINE TRIDIMENSIONALE | 114 |
| 3.1 Tecnica del phase shifting a proiezione di frange..... | 116 |
| 3.1.1 Phase shifting per il recupero della fase | 124 |
| 3.1.2 Modello geometrico per la stima degli angoli di proiezione..... | 128 |
| 3.1.3 Determinazione degli angoli di proiezione sulla base del datasheet | 141 |
| 3.1.4 Modello geometrico per la costruzione di griglie compensate | 147 |
| 3.1.5 Metodo per l'applicazione del phase shifting con DIC2D in assenza di strumentazione telecentrica | 154 |
| 3.2 Applicazione ad un caso reale e risultati | 158 |
| 3.2.1 Variazione dello spessore di frangia con la distanza di proiezione..... | 159 |
| 3.2.2 Applicazione del phase shifting a proiezione di frangia e DIC2D | 165 |
| CONCLUSIONI E FUTURI SVILUPPI..... | 181 |
| BIBLIOGRAFIA | 182 |
| APPENDICE CODICI D'IMPLEMENTAZIONE | 187 |

Introduzione

L'analisi meccanica svolta per via sperimentale al fine di determinare sforzi e deformazioni di un dato oggetto di interesse ingegneristico, pone in taluni casi problematiche relative all'effettiva applicabilità delle soluzioni già note: generalmente il problema è legato ai limiti sulle condizioni nelle quali deve essere effettuata l'analisi.

Ciò ha portato il settore della meccanica sperimentale a concentrare sempre più l'attività di ricerca sui metodi senza contatto, non solo ai fini della determinazione delle caratteristiche meccaniche dei materiali, ma anche del loro comportamento globale sotto le più differenti condizioni operative ai quali possono essere sottoposti.

Gli sviluppi tecnologici degli ultimi decenni soprattutto per quanto riguarda il settore tecnologico dell'informatica, con la produzione di processori in grado di effettuare sempre un maggior numero di calcoli nel minor tempo, e le schede video che permettono di visualizzare in maniera semplice e veloce i risultati dei calcoli, ha suggerito, col passare degli anni, di concentrarsi sempre più sullo sviluppo di tecniche che utilizzino il mezzo informatico al fine di poterne sfruttare appieno le potenzialità a proprio vantaggio.

Dagli anni 80 l'enorme sviluppo in campo di *computer vision* ed *image processing* ha suggerito la possibilità di sfruttarne i risultati ottenuti in applicazione alle materie di forte interesse ingegneristico meccanico.

Ciò ha permesso la nascita di veri e propri metodi sistematici di indagine sperimentale prima sconosciuti: si consideri per esempio la correlazione digitale d'immagine bidimensionale (DIC2D), la quale partendo dalla nascita del concetto di flusso ottico ha subito una forte evoluzione relativa al campo dell'ingegneria meccanica ed è oggi riconosciuta globalmente come metodo valido per la caratterizzazione dei materiali.

Punto di forza dell'enorme sviluppo di queste nuove metodiche è legato principalmente al loro contenuto onere in termini economici rispetto alle soluzioni tradizionali, e come precedentemente affermato, al fatto che possono interfacciarsi in maniera diretta con i calcolatori per l'elaborazione dei dati.

Risulta perciò che vi sia sempre più la tendenza a realizzare applicazioni che prevedano l'utilizzo dell'informazione contenuta nelle immagini digitali: tale fatto è riscontrabile in svariati campi non solo della correlazione digitale ma anche dell'interferometria, della

tomografia, della navigazione visuale autonoma di robot, dell'astronomia e quello aerospaziale, come testimonianza delle effettive potenzialità in campo multidisciplinare offerte dalla potenza di calcolo degli odierni computers.

Per quanto concerne il nostro campo d'interesse si precisa che l'analisi meccanica eseguita secondo la correlazione digitale bidimensionale pone dei forti limiti alle effettive informazioni estrapolabili dal materiale studiato.

Come noto non risulta possibile estrapolare con tale metodo l'informazione relativa a stati di deformazione tridimensionale: ogni spostamento fuori dal piano risulta essere apparente.

In quest'ottica si inserisce la correlazione digitale d'immagine tridimensionale la quale si è imposta, anche a livello commerciale nell'ultimo decennio, come valido metodo di analisi sperimentale di tipo meccanico che sfruttando l'informazione relativa a più viste dello stesso oggetto risulta in grado di determinarne gli spostamenti e la variazione di forma a seguito di un carico applicato.

Il successo anche in questo caso è legato al fatto che rappresenta una valida alternativa a metodologie più precise, ma più costose come la tomografia e la correlazione digitale volumetrica d'immagine.

Come si può evincere, dalla necessità di sfruttare più viste dell'oggetto studiato, le soluzioni a livello commerciale sono studiate generalmente sui principi della geometria epipolare e della visione binoculare: in pratica necessitano dell'utilizzo di due fotocamere digitali separate che inquadrino in istanti successivi l'oggetto di studio al fine di caratterizzarne la deformazione attraverso la misura dello spostamento dei suoi punti superficiali.

Il presente lavoro di tesi si pone perciò l'obiettivo della ricerca di modelli alternativi alla visione stereoscopica in grado di risolvere il problema della correlazione digitale d'immagine tridimensionale, e risulta essere sviluppato su tre capitoli:

- Capitolo 1: si analizza il metodo DIC3D secondo il modello della visione binoculare o stereoscopica al fine di chiarirne la *pipeline* di processi.
- Capitolo 2: si analizza il metodo DIC2D essendo parte intergrante del metodo che si andrà a proporre di phase shifting a proiezione di frangia.
- Capitolo 3: si sviluppa il metodo misto DIC2D e phase shifting a proiezione di frangia per l'applicazione in condizioni generali.

Capitolo 1

DIC 3D secondo il modello stereoscopico

Si affronta in primo luogo l'analisi del metodo della correlazione digitale d'immagine tridimensionale secondo il modello di visione binoculare la cui trattazione dettagliata è riportata in [1],[2],[7].

Tale modello risulta infatti quello principalmente utilizzato dalle aziende produttrici di sistemi di analisi meccanica di deformazioni tridimensionali, e poiché presuppone l'utilizzo di un sistema di acquisizione d'immagini costituito da almeno due fotocamere digitali, appare chiara la necessità di delineare in maniera esaustiva sia il processo effettivo che porta alla formazione dell'immagine digitale, sia le grandezze proprie della problematica affrontata senza le quali la trattazione risulterebbe difficilmente sviluppabile.

Nella prima parte del capitolo si concentra perciò l'attenzione sul modello d'acquisizione dell'immagine per singola fotocamera, noto il quale risulta possibile esporre il modello geometrico della correlazione digitale tridimensionale d'immagine.

Nella seconda e terza parte si affrontano in primo luogo le metodologie di calibrazione per la singola fotocamera, ed infine si effettua l'analisi del DIC3D secondo il modello di visione stereoscopico.

1.1 Modello di acquisizione d'immagine per singola fotocamera

L'acquisizione di un'immagine consta di una serie di processi che risultano in una mappatura di punti dal mondo tridimensionale al piano dell'immagine bidimensionale.

Considerando infatti un generico punto nello spazio la cui posizione può essere espressa in unità metriche (per esempio metri o millimetri) si ottiene alla fine del processo una sua proiezione sul piano dell'immagine nella quale la posizione viene espressa, per convenzione, in unità di pixel.

Considerando la seguente figura 1.0, risulta che lo studio del processo d'acquisizione d'immagine fa riferimento al modello *pinhole* della fotocamera: in pratica si considera che tutti i fasci luminosi passino per un foro chiamato appunto *pinhole* o centro di proiezione, e vadano a proiettarsi su punti del piano immagine posto alla distanza focale c , calcolata lungo l'asse ottico perpendicolare al piano stesso.

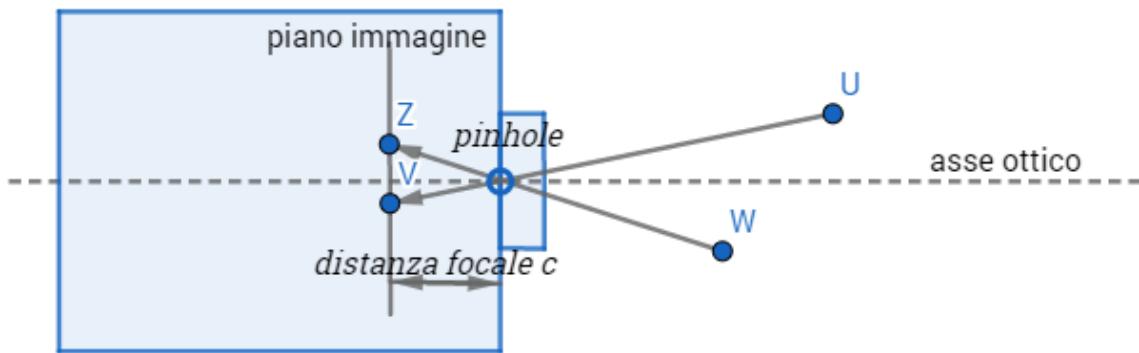


Fig.1.0. Rappresentazione schematica del processo di acquisizione immagine.

Relativamente alla figura 1.1, rappresentante la schematizzazione di un generico scenario con una fotocamera avente pinhole in cO e posizionata rispetto ad un sistema di riferimento globale con origine in wO , si distinguono 4 sistemi di coordinate:

- $^w\{OXYZ\}$: sistema di riferimento globale, coincidente per esempio con un osservatore posto ad una certa distanza rispetto alla macchina fotografica.
- $^c\{OXYZ\}$: sistema di riferimento della fotocamera con origine nel pin-hole.
- $^i\{OXYZ\}$: sistema di coordinate del piano immagine.
- $^s\{OXYZ\}$: sistema di coordinate del piano del sensore.

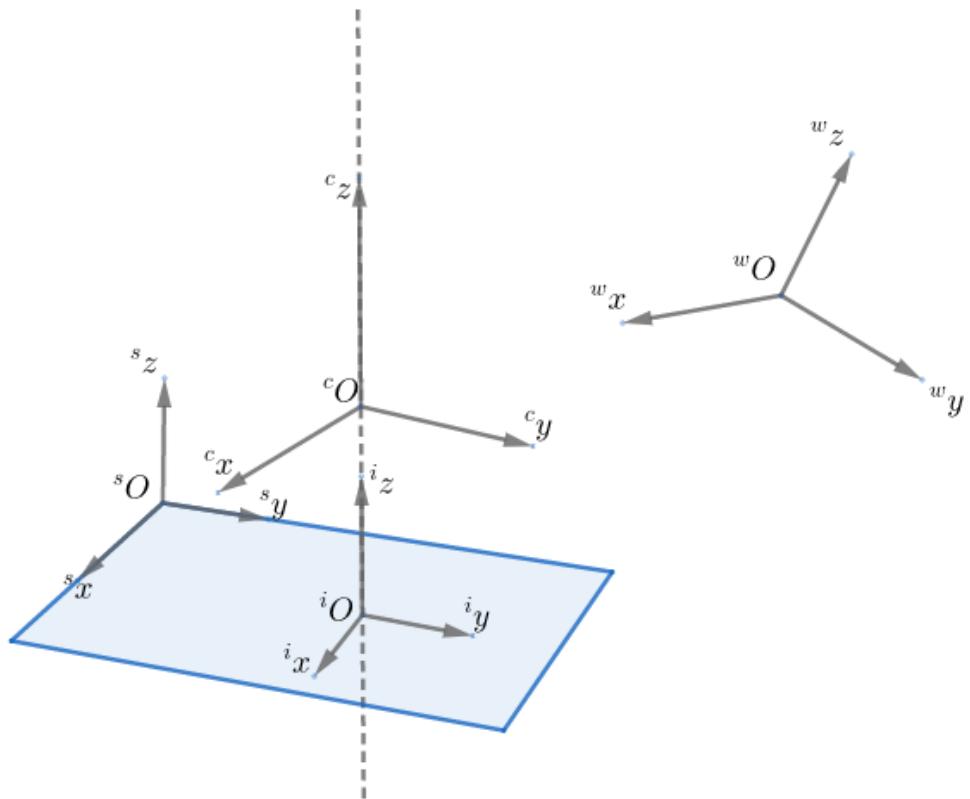


Fig.1.1. Rappresentazione schematica di un sistema d'acquisizione d'immagine

Come si può notare gli ultimi tre sistemi di riferimento, e cioè quello della camera ${}^C\{\text{OXYZ}\}$, del piano immagine ${}^I\{\text{OXYZ}\}$ e del sensore ${}^S\{\text{OXYZ}\}$ sono interni alla macchina fotografica, e servono principalmente alla modellazione dei processi che portano alla formazione dell'immagine finale, a partire dall'inquadratura.

Risulta altresì che idealmente questi sistemi di riferimento sono fra loro legati da alcune condizioni che si riportano di seguito:

- Fra il sistema ${}^C\{\text{OXYZ}\}$ ed ${}^I\{\text{OXYZ}\}$ vi è orientazione relativa nulla, che equivale alla condizione che la matrice di rotazione che lega i due sistemi sia pari alla matrice identità (${}^I[\mathbf{R}]_C = [I]$);
- Il sistema ${}^I\{\text{OXYZ}\}$ è semplicemente traslato rispetto al sistema ${}^C\{\text{OXYZ}\}$ della distanza focale indicata attraverso la costante c della fotocamera ${}^C\mathbf{O}_i = [0,0,-c]^t$;
- Il piano sensore è perfettamente complanare con quello dell'immagine, e il relativo sistema di riferimento ${}^S\{\text{OXYZ}\}$ risulta semplicemente traslato rispetto a quello del piano immagine ${}^I\{\text{OXYZ}\}$ ($\overrightarrow{{}^S\mathbf{O}} \overrightarrow{{}^I\mathbf{O}} \neq 0$).

- L'asse z del sistema di riferimento dell'immagine $^i\{\text{OXYZ}\}$ coincide con l'asse ottico di proiezione, e l'origine di questo sistema di riferimento è chiamato punto principale dell'immagine.

Bisogna precisare fin da subito che alcune delle condizioni sopracitate di raro trovano effettivo riscontro nei sistemi di acquisizione reali, fenomeno legato ai limiti tecnologici con i quali vengono costruiti tali strumenti. Nonostante non sia quindi possibile conoscere a priori in modo preciso la deriva rispetto alle condizioni ideali, si può comunque tenere conto di questi fenomeni introducendo il loro effetto secondo modelli di compensazione non lineari.

Complessivamente considerato perciò il processo d'acquisizione si configura come una mappatura sequenziale di punti a partire dal sistema $^w\{\text{OXYZ}\}$ a quello $^s\{\text{OXYZ}\}$ passando per i due sistemi di riferimento $^c\{\text{OXYZ}\}$ ed $^i\{\text{OXYZ}\}$.

1.1.1 Trasformazione dal sistema di riferimento globale a quello della fotocamera

Si consideri ora un punto $\mathbf{P}(^w x_p, ^w y_p, ^w z_p)$ espresso nel sistema $^w\{\text{OXYZ}\}$ nelle rispettive unità di misura metriche: il processo che porta alla sua proiezione sul corrispettivo punto sul piano dell'immagine a $\mathbf{P}(^i x_p, ^i y_p)$ può essere visualizzato come in figura 1.2.

La posizione del punto P sul piano dell'immagine finale dipenderà sia dalla posizione del sistema $^c\{\text{OXYZ}\}$ della fotocamera, ma anche dalla distanza del piano dell'immagine rispetto al pinhole ($^c O$) della fotocamera, pari appunto alla distanza focale c .

Riguardo alla simbologia utilizzata nel seguito della trattazione si indica con $^j \mathbf{x}_i$ il generico vettore posizione del punto i espresso nel sistema di riferimento j ; si indica invece con $^l R_k$ la matrice che definisce l'orientazione del sistema di riferimento k espressa nel sistema di riferimento l .

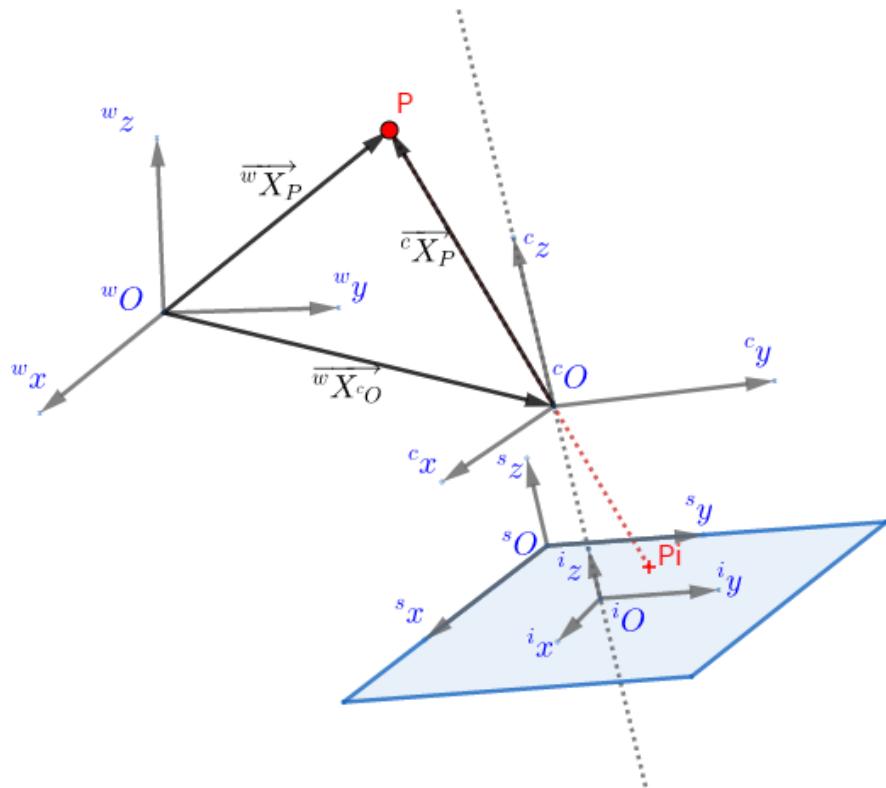


Fig.1.2. Rappresentazione della posizione del punto P rispetto ai diversi sistemi di riferimento.

Relativamente alla figura 1.2 si ha quindi che il vettore posizione di P nel sistema di riferimento globale ${}^W\{\text{OXYZ}\}$ equivale ad:

$${}^W\mathbf{X}_P = {}^W\mathbf{X}_{c_0} + {}^W\mathbf{R}_C {}^C\mathbf{X}_P \quad (1.1)$$

Dalla quale risulta:

$${}^C\mathbf{X}_P = {}^W\mathbf{R}_C^{-1} ({}^W\mathbf{X}_P - {}^W\mathbf{X}_{c_0}) = {}^C\mathbf{R}_W ({}^W\mathbf{X}_P - {}^W\mathbf{X}_{c_0}) \quad (1.2)$$

Passando ora alla notazione in coordinate omogenee, si può riscrivere l'espressione in forma matriciale:

$$\begin{bmatrix} {}^c\mathbf{X}_P \\ {}^c\mathbf{y}_P \\ {}^c\mathbf{z}_P \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{11} & \mathbf{r}_{12} & \mathbf{r}_{13} & 0 \\ \mathbf{r}_{21} & \mathbf{r}_{22} & \mathbf{r}_{23} & 0 \\ \mathbf{r}_{31} & \mathbf{r}_{32} & \mathbf{r}_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -{}^w\mathbf{x}_c_0 \\ 0 & 1 & 0 & -{}^w\mathbf{y}_c_0 \\ 0 & 0 & 1 & -{}^w\mathbf{z}_c_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^w\mathbf{X}_P \\ {}^w\mathbf{y}_P \\ {}^w\mathbf{z}_P \\ 1 \end{bmatrix} \quad (1.3)$$

In forma compatta si ottiene la seguente espressione:

$$\begin{bmatrix} {}^c\mathbf{X}_P \\ 1 \end{bmatrix} = \begin{bmatrix} {}^c[\mathbf{R}]_W & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} [\mathbf{I}] & -{}^w\mathbf{X}_c_0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^w\mathbf{X}_P \\ 1 \end{bmatrix} \quad (1.4)$$

Indicando ora con la matrice H il termine seguente:

$$[\mathbf{H}] = \begin{bmatrix} {}^c[\mathbf{R}]_W & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} [\mathbf{I}] & -{}^w\mathbf{X}_c_0 \\ 0 & 1 \end{bmatrix} \quad (1.5)$$

L'espressione risulta in forma ancora più compatta:

$$\begin{bmatrix} {}^c\mathbf{X}_P \\ 1 \end{bmatrix} = [\mathbf{H}] \begin{bmatrix} {}^w\mathbf{X}_P \\ 1 \end{bmatrix} \quad (1.6)$$

Come si può notare la matrice H contiene tutti i parametri caratterizzanti la posizione e orientazione del sistema ${}^c\{\text{OXYZ}\}$ rispetto a ${}^w\{\text{OXYZ}\}$. Questi sono noti come i parametri estrinseci del sistema di acquisizione dell'immagine e constano appunto dei 3 angoli di rotazione e delle 3 componenti del vettore traslazione della fotocamera rispetto al sistema di riferimento globale. La relazione trovata permette perciò di esprimere il punto P nel sistema di riferimento della fotocamera.

1.1.2 Trasformazione di proiezione centrale

Noto il vettore cX_P ha quindi luogo la trasformazione di proiezione centrale secondo il modello pin-hole, la quale mappa il punto sul piano dell'immagine, come riportato in figura seguente:

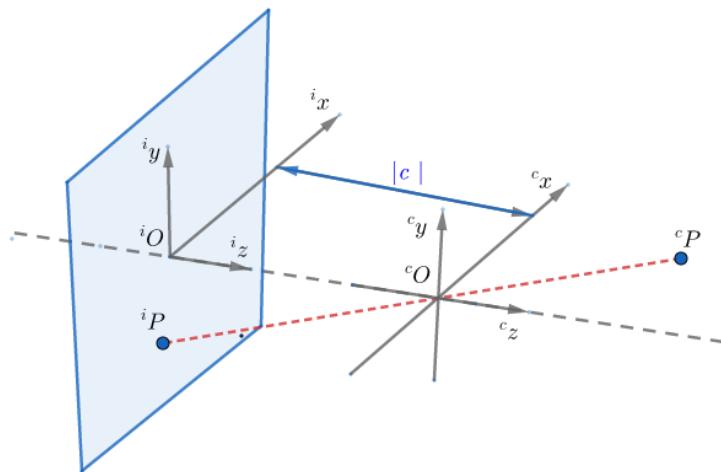


Fig.1.3. schematizzazione della trasformazione di proiezione centrale.

Considerando i diversi piani sui quali ha luogo la proiezione come riportato nelle seguenti figure 1.4 ed 1.5, si possono trovare facilmente le relazioni che legano le due posizioni del punto sul piano immagine e rispetto alla fotocamera:

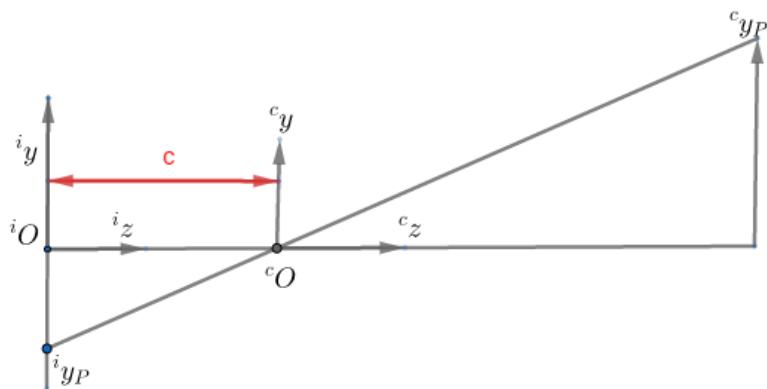


Fig.1.4. Rappresentazione della trasformazione di proiezione centrale sul piano yz.

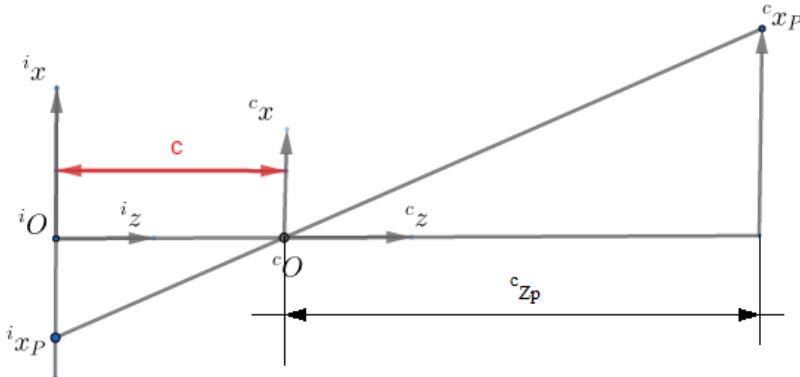


Fig.1.5. Rappresentazione della trasformazione di proiezione centrale sul piano xz.

Risultano verificate le seguenti relazioni:

$$\frac{i_x_p}{c} = \frac{c_{x_p}}{c_{z_p}} \quad (1.7)$$

$$\frac{i_y_p}{c} = \frac{c_{y_p}}{c_{z_p}}$$

Dalle quali risulta che:

$$i_x_p = c \frac{c_{x_p}}{c_{z_p}} \quad (1.8)$$

$$i_y_p = c \frac{c_{y_p}}{c_{z_p}}$$

Come si può notare la trasformazione proiettiva è caratterizzata dalla perdita di informazione sull'effettiva profondità lungo l'asse Z del punto considerato: ogni punto appartenente alla retta di proiezione viene infatti mappato in un unico punto sul piano dell'immagine finale, così che per ogni punto sul piano immagine non risulta univocamente definita la sua posizione nella rispettiva scena reale.

Introducendo ora la matrice di proiezione centrale [P] si è in grado di trovare la relazione che lega il punto P espresso nel sistema della fotocamera e in quello del piano immagine:

$$[P] = \begin{bmatrix} c & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (1.9)$$

Tale che risulta verificata la relazione:

$$\begin{bmatrix} {}^iX_P \\ {}^iY_P \\ 1 \end{bmatrix} = \begin{bmatrix} c & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} {}^cX_P \\ {}^cY_P \\ {}^cZ_P \\ 1 \end{bmatrix} \quad (1.10)$$

Si può ora riscrivere in forma più compatta:

$$\begin{bmatrix} {}^iX_P \\ 1 \end{bmatrix} = [P] \begin{bmatrix} {}^cX_P \\ 1 \end{bmatrix} \quad (1.11)$$

Riutilizzando ora i risultati precedenti per il valore cX_P (relazione 1.6) si può riscrivere l'espressione (1.11) nel seguente modo:

$$\begin{bmatrix} {}^iX_P \\ 1 \end{bmatrix} = [P] [H] \begin{bmatrix} {}^wX_P \\ 1 \end{bmatrix} \quad (1.12)$$

Sviluppando i diversi termini si ottiene perciò:

$$\begin{bmatrix} {}^iX_P \\ 1 \end{bmatrix} = \begin{bmatrix} c & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} {}^c[R]_W & - {}^c[R]_W {}^wX_{c_O} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^wX_P \\ 1 \end{bmatrix} \quad (1.13)$$

Sviluppando la (1.13) risulta:

$$\begin{bmatrix} {}^iX_p \\ {}^iY_p \\ 1 \end{bmatrix} = \begin{bmatrix} c & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & -(r_{11}{}^wX_{cO} + r_{12}{}^wY_{cO} + r_{13}{}^wZ_{cO}) \\ r_{21} & r_{22} & r_{23} & -(r_{21}{}^wX_{cO} + r_{22}{}^wY_{cO} + r_{23}{}^wZ_{cO}) \\ r_{31} & r_{32} & r_{33} & -(r_{31}{}^wX_{cO} + r_{32}{}^wY_{cO} + r_{33}{}^wZ_{cO}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^wX_p \\ {}^wY_p \\ {}^wZ_p \\ 1 \end{bmatrix} \quad (1.14)$$

Eseguendo ora il prodotto matriciale della 1.14 risulta:

$$\begin{bmatrix} {}^iX_p \\ {}^iY_p \\ 1 \end{bmatrix} = \begin{bmatrix} c \cdot r_{11} & c \cdot r_{12} & c \cdot r_{13} & -c \cdot (r_{11}{}^wX_{cO} + r_{12}{}^wY_{cO} + r_{13}{}^wZ_{cO}) \\ c \cdot r_{21} & c \cdot r_{22} & c \cdot r_{23} & -c \cdot (r_{21}{}^wX_{cO} + r_{22}{}^wY_{cO} + r_{23}{}^wZ_{cO}) \\ r_{31} & r_{32} & r_{33} & -(r_{31}{}^wX_{cO} + r_{32}{}^wY_{cO} + r_{33}{}^wZ_{cO}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^wX_p \\ {}^wY_p \\ {}^wZ_p \\ 1 \end{bmatrix} \quad (1.15)$$

ed in definitiva:

$$\begin{bmatrix} {}^iX_p \\ {}^iY_p \\ 1 \end{bmatrix} = \begin{bmatrix} c \cdot r_{11}({}^wX_p - {}^wX_{cO}) + c \cdot r_{12}({}^wY_p - {}^wY_{cO}) + c \cdot r_{13}({}^wZ_p - {}^wZ_{cO}) \\ c \cdot r_{21}({}^wX_p - {}^wX_{cO}) + c \cdot r_{22}({}^wY_p - {}^wY_{cO}) + c \cdot r_{23}({}^wZ_p - {}^wZ_{cO}) \\ r_{31}({}^wX_p - {}^wX_{cO}) + r_{32}({}^wY_p - {}^wY_{cO}) + r_{33}({}^wZ_p - {}^wZ_{cO}) \end{bmatrix} \quad (1.16)$$

La quale permette di invertire la notazione omogenea ed esplicitare l'espressione per le due coordinate del punto P sul piano immagine:

$${}^iX_p = \frac{c \cdot [r_{11}({}^wX_p - {}^wX_{cO}) + r_{12}({}^wY_p - {}^wY_{cO}) + r_{13}({}^wZ_p - {}^wZ_{cO})]}{r_{31}({}^wX_p - {}^wX_{cO}) + r_{32}({}^wY_p - {}^wY_{cO}) + r_{33}({}^wZ_p - {}^wZ_{cO})} \quad (1.17)$$

$${}^iY_p = \frac{c \cdot [r_{21}({}^wX_p - {}^wX_{cO}) + r_{22}({}^wY_p - {}^wY_{cO}) + r_{23}({}^wZ_p - {}^wZ_{cO})]}{r_{31}({}^wX_p - {}^wX_{cO}) + r_{32}({}^wY_p - {}^wY_{cO}) + r_{33}({}^wZ_p - {}^wZ_{cO})} \quad (1.18)$$

Tuttavia analizzando la (1.13) si evince un modo alternativo, che può tornare utile in seguito, per riscrivere l'espressione (1.14) come prodotto di matrici 3x3.

Si ha infatti che l'espressione in questione equivale anche ad:

$$\begin{bmatrix} {}^i\mathbf{x}_p \\ {}^i\mathbf{y}_p \\ 1 \end{bmatrix} = \begin{bmatrix} c & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} {}^w\mathbf{x}_p - {}^w\mathbf{x}_o \\ {}^w\mathbf{y}_p - {}^w\mathbf{y}_o \\ {}^w\mathbf{z}_p - {}^w\mathbf{z}_o \end{bmatrix} \quad (1.19)$$

1.1.3 Trasformazione fra sistema del piano immagine e del sensore.

Rimane ora da analizzare la mappatura fra sensore e piano immagine: processo che permette l'effettiva conversione da valori espressi in unità metriche a quelli in pixel.

Al fine di non compromettere la generalità della trattazione, si suppone di seguito che i sistemi di riferimento del piano immagine e sensore, nonostante siano perfettamente complanari possano differire l'uno dall'altro per la non perfetta ortogonalità fra i rispettivi assi principali [5].

Quanto detto può risultare sicuramente valido per il sistema sensore, essendo quest'ultimo un componente fisico della macchina fotografica a differenza del sistema del piano immagine che viene usato come mezzo matematico per modellare il processo di acquisizione: si suppone di seguito quindi che il sistema di riferimento del piano immagine sia perfettamente ortogonale, mentre quello del sensore sia distorto.

Indicando ora con ϑ l'angolo di distorsione del sensore, e spostando ${}^i\{\text{OXYZ}\}$ tale che abbia origine comune con ${}^s\{\text{OXYZ}\}$ risulta ottenibile la relazione che lega i due sistemi di riferimento; Successivamente si tiene conto della traslazione di ${}^i\{\text{OXYZ}\}$ rispetto ad ${}^s\{\text{OXYZ}\}$ dato dalle quantità iT_x , iT_y espresse nel sistema ${}^i\{\text{OXYZ}\}$.

Relativamente alla seguente figura 1.6 si ha infatti che un generico punto Q sul piano dell'immagine può essere espresso nel sistema del sensore attraverso le seguenti relazioni:

$$\begin{aligned} {}^s\mathbf{x}_Q &= {}^i\mathbf{x}_Q - \overline{{}^s\mathbf{x}_Q \cdot {}^i\mathbf{x}_Q} = {}^i\mathbf{x}_Q - \frac{{}^i\mathbf{y}_Q}{\operatorname{tg}(\vartheta)} \\ {}^s\mathbf{y}_Q &= \frac{{}^i\mathbf{y}_Q}{\sin(\vartheta)} \end{aligned} \quad (1.20)$$

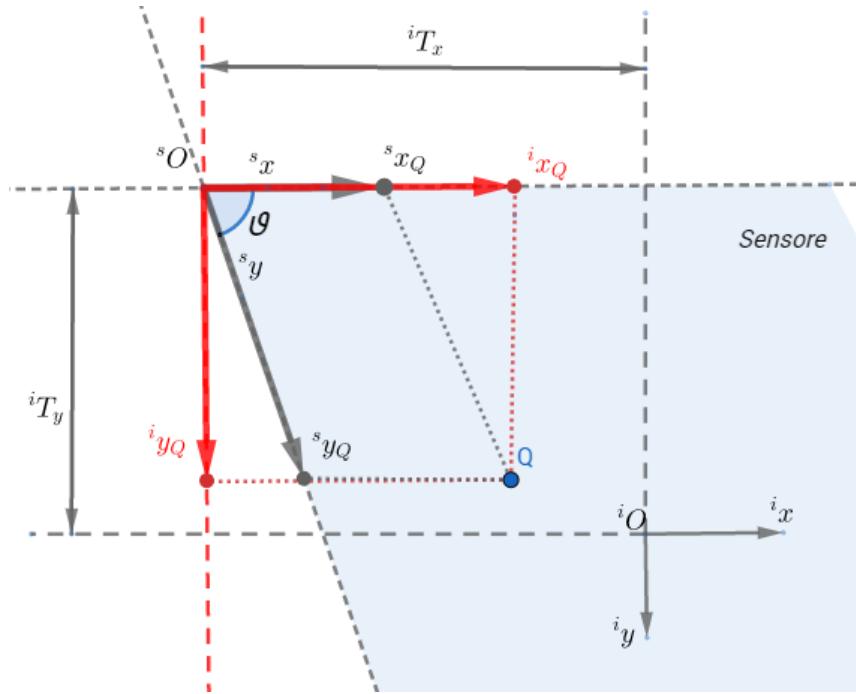


Fig.1.6. Analisi della posizione di un punto Q sul piano immagine rispetto al sistema di riferimento del sensore.

Quanto trovato vale per ogni punto, quindi si può omettere il pedice relativo al singolo punto Q e riscrivere la relazione in forma matriciale come di seguito:

$$\begin{bmatrix} {}^s \mathbf{x} \\ {}^s \mathbf{y} \end{bmatrix} = \begin{bmatrix} 1 & -\frac{1}{\tan(\theta)} \\ 0 & \frac{1}{\sin(\theta)} \end{bmatrix} \begin{bmatrix} {}^i \mathbf{x} \\ {}^i \mathbf{y} \end{bmatrix} \quad (1.21)$$

Supponendo ora due diversi fattori di scala S_x, S_y per la conversione da unità metriche a pixel espressi solitamente come [pixel/mm] per le due direzioni principali, si ha che i valori effettivi ${}^s \mathbf{X}, {}^s \mathbf{Y}$ saranno proporzionali a queste costanti.

In definitiva tenendo conto anche della traslazione del sistema di riferimento del piano immagine rispetto al sistema di riferimento del sensore (fig.1.7), si ottiene la seguente espressione che lega i due sistemi di riferimento:

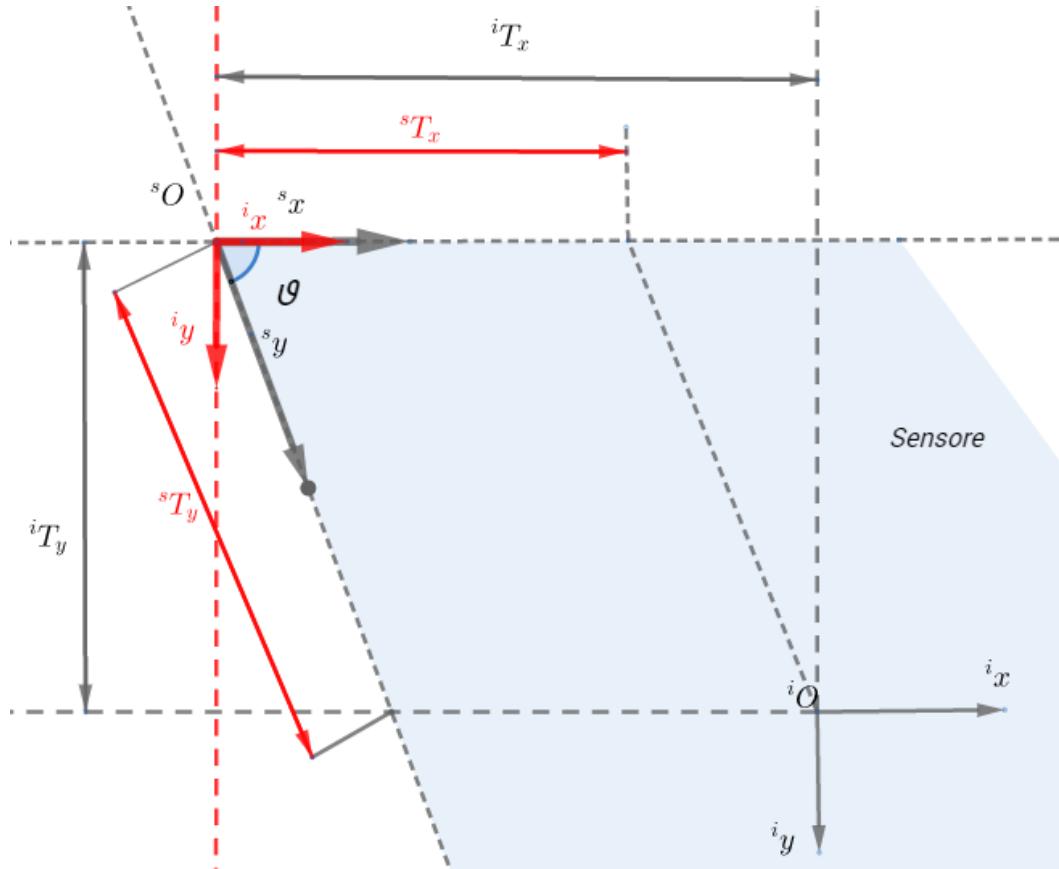


Fig.1.7. Rappresentazione della traslazione del sistema di riferimento del piano immagine rispetto a quella del sensore.

$$\begin{bmatrix} {}^s x \\ {}^s y \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} 1 & -\frac{1}{\tan(\theta)} \\ 0 & \frac{1}{\sin(\theta)} \end{bmatrix} \begin{bmatrix} {}^i x \\ {}^i y \end{bmatrix} - \begin{bmatrix} {}^s T_x \\ {}^s T_y \end{bmatrix} \quad (1.22)$$

La (1.22) può essere riscritta come:

$$\begin{bmatrix} {}^s x \\ {}^s y \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} 1 & -\frac{1}{\tan(\theta)} \\ 0 & \frac{1}{\sin(\theta)} \end{bmatrix} \begin{bmatrix} {}^i x \\ {}^i y \end{bmatrix} - \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} {}^i T_x & -\frac{{}^i T_x}{\tan(\theta)} \\ 0 & \frac{{}^i T_y}{\sin(\theta)} \end{bmatrix} \quad (1.23)$$

Passando ora alla notazione omogenea risulta la seguente forma più compatta per l'espressione (1.23):

$$\begin{bmatrix} {}^s\mathbf{x} \\ {}^s\mathbf{y} \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & -\frac{1}{\tan(\theta)} & -S_x({}^iT_x - \frac{{}^iT_y}{\tan(\theta)}) \\ 0 & \frac{S_y}{\sin(\theta)} & -\frac{S_y}{\sin(\theta)} {}^iT_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^i\mathbf{x} \\ {}^i\mathbf{y} \\ 1 \end{bmatrix} \quad (1.24)$$

1.1.4 Matrice di calibrazione e di proiezione

Nota dalla trattazione precedente l'insieme delle trasformazioni che intercorrono fra i diversi sistemi di riferimento risulta ora possibile dare una rappresentazione globale del processo analizzato.

Sostituendo infatti l'espressione (1.19) nella (1.24) si ottiene la seguente espressione finale che governa l'intero processo dell'acquisizione d'immagine:

$$\begin{bmatrix} {}^s\mathbf{x}_p \\ {}^s\mathbf{y}_p \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & -\frac{1}{\tan(\theta)} & -S_x({}^iT_x - \frac{{}^iT_y}{\tan(\theta)}) \\ 0 & \frac{S_y}{\sin(\theta)} & -\frac{S_y}{\sin(\theta)} {}^iT_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} {}^w\mathbf{x}_p - {}^w\mathbf{x}_{c_0} \\ {}^w\mathbf{y}_p - {}^w\mathbf{y}_{c_0} \\ {}^w\mathbf{z}_p - {}^w\mathbf{z}_{c_0} \end{bmatrix} \quad (1.25)$$

Analizzando l'espressione ottenuta si consideri il prodotto delle prime due matrici a secondo membro dell'espressione (1.25) e lo si indichi con [K]:

$$[K] = \begin{bmatrix} S_x \cdot c & -\frac{1}{\tan(\theta)} \cdot c & -S_x ({}^i T_x - \frac{{}^i T_y}{\tan(\theta)}) \\ 0 & \frac{S_y}{\sin(\theta)} \cdot c & -\frac{S_y}{\sin(\theta)} {}^i T_y \\ 0 & 0 & 1 \end{bmatrix} \quad (1.26)$$

La matrice $[K]$ appena ottenuta è nota come matrice di calibrazione della fotocamera, e contiene i sei parametri intrinseci del sistema di acquisizione: $S_x, S_y, T_x, T_y, c, \theta$.

Si noti inoltre che nel caso $\theta=90^\circ$ (caso che per altro rispecchia la maggior parte dei sensori di fabbricazione industriale), la matrice $[K]$ risulta notevolmente semplificata come di seguito:

$$[K] = \begin{bmatrix} S_x \cdot c & 0 & -S_x \cdot {}^i T_x \\ 0 & S_y \cdot c & -S_y \cdot {}^i T_y \\ 0 & 0 & 1 \end{bmatrix} \quad (1.27)$$

In definitiva passando alla notazione omogenea è possibile trovare una forma più compatta che lega la posizione del generico punto P nello spazio, espresso in unità metriche, alla sua proiezione sul sensore ad una posizione espressa in pixels, mediante un'unica matrice $[P]$:

$$\begin{bmatrix} {}^s X_p \\ {}^s Y_p \\ 1 \end{bmatrix} = [P] \begin{bmatrix} {}^w X_p \\ {}^w Y_p \\ {}^w Z_p \\ 1 \end{bmatrix} \quad (1.28)$$

La matrice $[P]$ è nota come matrice di proiezione e risulta pari ad:

$$[P] = \begin{bmatrix} S_x \cdot c & -\frac{1}{\tan(\theta)} \cdot c & -S_x ({}^i T_x - \frac{{}^i T_y}{\tan(\theta)}) & 0 \\ 0 & \frac{S_y}{\sin(\theta)} \cdot c & -\frac{S_y {}^i T_y}{\sin(\theta)} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -{}^w x_c o \\ 0 & 1 & 0 & -{}^w y_c o \\ 0 & 0 & 1 & -{}^w z_c o \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.29)$$

Come si può notare la matrice di proiezione contiene ora tutti i parametri sia intrinseci che estrinseci del sistema di acquisizione.

Nelle applicazioni di interesse ingegneristico come la correlazione digitale d'immagine tridimensionale risulta necessaria la conoscenza di tutti questi parametri, in quest'ottica si inserisce il processo di calibrazione consistente nella determinazione di questi parametri a partire da punti dei quali sono note sia le posizioni nello spazio (in unità metriche), sia le rispettive posizioni sulle immagini (in unità di pixel).

1.1.5 Modello geometrico del DIC3D

La trattazione sviluppata fino a questo punto fornisce gli strumenti necessari all'esposizione del problema della correlazione digitale d'immagine tridimensionale.

Nel campo della *computer vision* uno dei casi più importanti è rappresentato dalla configurazione stereo normale di due fotocamere, noto come sistema *stereo rig* o *3D rig*: in pratica costruisce un sistema formato da due fotocamere delle quali si conosce la distanza focale (c), la distanza fra i rispettivi pinhole (d), i punti principali sui piani immagine, e l'orientazione relativa (fig 1.8):

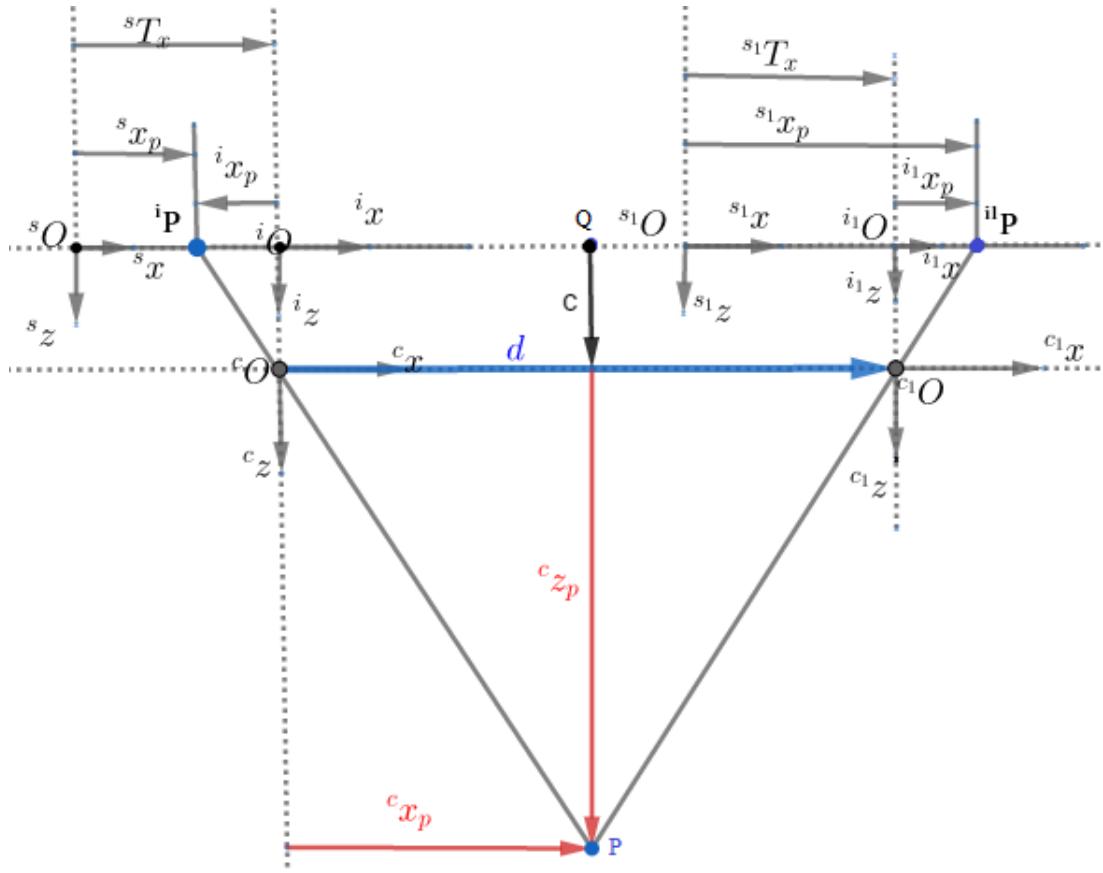


Fig. 1.8. Rappresentazione della configurazione di un sistema composto da sue fotocamere in configurazione stereo normale.

In riferimento alla figura 1.8 risulta che nella configurazione stereo normale le due fotocamere sono caratterizzate da orientazione relativa nulla ${}^{ii}[R]_i = [I]$, e la posizione dei rispettivi pinhole è caratterizzata dall' avere la stessa quota lungo l'asse y, e distanza pari a d lungo l'asse x.

Risulta che in questa configurazione, sfruttando la similarità fra i triangoli:

$${}^c\mathbf{O} \overset{\Delta}{\sim} {}^c\mathbf{OP}, \quad {}^{ii}\mathbf{P} \overset{\Delta}{\sim} {}^i\mathbf{P} \quad (1.30)$$

Si può verificare la seguente relazione:

$$\frac{d + ({}^i\mathbf{x}_p - {}^i\mathbf{x}_p)}{d} = \frac{{}^c\mathbf{z}_p + c}{{}^c\mathbf{z}_p} \quad (1.31)$$

valida considerando che la norma dei vettori d e c non cambia a seconda del sistema di riferimento considerato, e cioè:

$$\begin{aligned} {}^c d &= {}^i d = {}^s d = d \\ {}^c c &= {}^i c = {}^s c = c \end{aligned} \quad (1.32)$$

Si hanno come visto precedentemente le seguenti relazioni che intercorrono fra il sensore e il piano immagine:

$${}^i x_p = {}^s x_p - {}^s T_x \quad (1.33)$$

$${}^i x_p = {}^s x_p - {}^s T_x$$

che permettono di riscrivere la (1.31) nella seguente maniera:

$$\frac{d + \left[\left({}^s_i x_p - {}^s_i T_x \right) - \left({}^s x_p - {}^s T_x \right) \right]}{d} = \frac{{}^c z_p + c}{{}^c z_p} \quad (1.34)$$

Dalla (1.34) può essere estrapolata la quota z del punto osservato:

$${}^c z_p = \frac{c \cdot d}{\left[\left({}^s_i x_p - {}^s_i T_x \right) - \left({}^s x_p - {}^s T_x \right) \right]} \quad (1.35)$$

Come si può notare dalla (1.35) il problema della determinazione della coordinata z del punto P dipende da una quantità che prende il nome di *disparità*:

$$D = \left[\left({}^s_i x_p - {}^s_i T_x \right) - \left({}^s x_p - {}^s T_x \right) \right] \quad (1.36)$$

Noto il valore della disparità per il punto P si possono facilmente trovare tutte le sue componenti del vettore posizione.

Si ha infatti la seguente espressione di facile verifica:

$$-\frac{{}^c x_p}{{}^i x_p} = \frac{{}^c z_p}{{}^c c} \quad (1.37)$$

che permette il calcolo dell'ascissa del punto P nel sistema di riferimento della prima fotocamera come di seguito riportato:

$${}^c x_p = -{}^i x_p \cdot \frac{{}^c Z_p}{c} \quad (1.38)$$

Si ha inoltre che:

$$-\frac{{}^c y_p}{{}^i y_p} = \frac{{}^c x_p}{{}^i x_p} \quad (1.39)$$

che permette di calcolare l'ordinata del punto P rispetto al sistema della fotocamera:

$${}^c y_p = -{}^i y_p \cdot \frac{{}^c x_p}{{}^i x_p} \quad (1.40)$$

Si noti come nel caso si faccia utilizzo di due fotocamere identiche si avrebbe che i punti principali sui piani immagine avrebbero la stessa posizione, e quindi si avrebbe la seguente relazione sui vettori traslazione del sistema dell'immagine rispetto al sensore:

$${}^{s_i} T_x = {}^s T_x \quad (1.41)$$

La disparità in questo caso equivarrebbe semplicemente alla seguente espressione:

$$D = {}^{s_i} x_p - {}^s x_p = {}^{iI} x_p - {}^i x_p \quad (1.42)$$

Un'importante conseguenza della determinabilità della posizione di un singolo punto nello spazio è rappresentato dalla possibilità di poter studiare anche lo spostamento del punto lungo le tre direzioni spaziali.

Per semplicità si fa ora riferimento direttamente alle immagini prodotte dalle due fotocamere, essendo la trasformazione intercorrente fra sensore e immagine semplicemente di traslazione e le rispettive dimensioni risultano essere legate fra loro da un semplice fattore di scala.

Si consideri relativamente alla seguente figura 1.9 un punto P che passa in istanti successivi dalla posizione A a quella B:

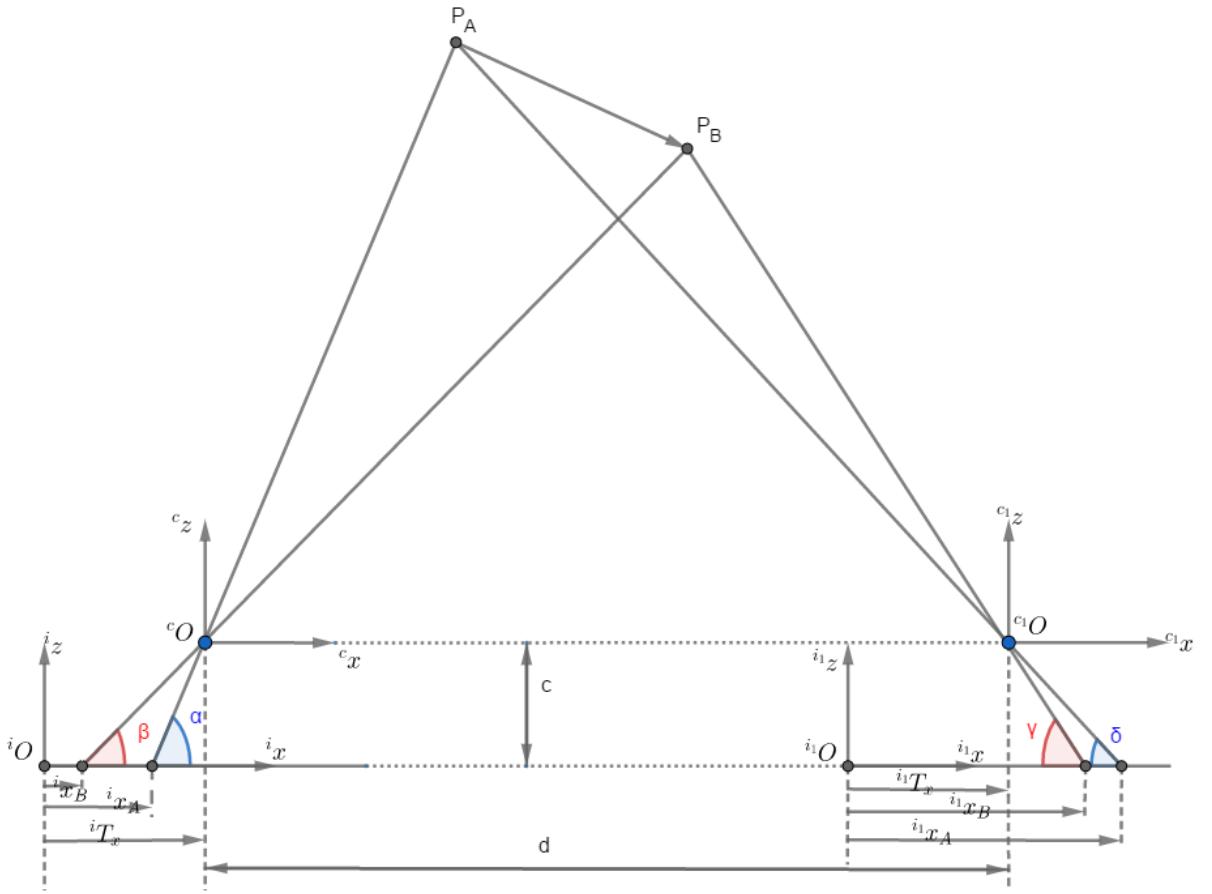


Fig. 1.9. Punto P soggetto all'acquisizione dell'immagine da parte dello stereo rig in istanti successivi di movimento.

Si analizza ora quello che succede a livello della singola fotocamera, per definire un modello che permetta di determinare lo spostamento.

A tal fine si consideri per esempio l'immagine nella prima fotocamera, figura 1.10:

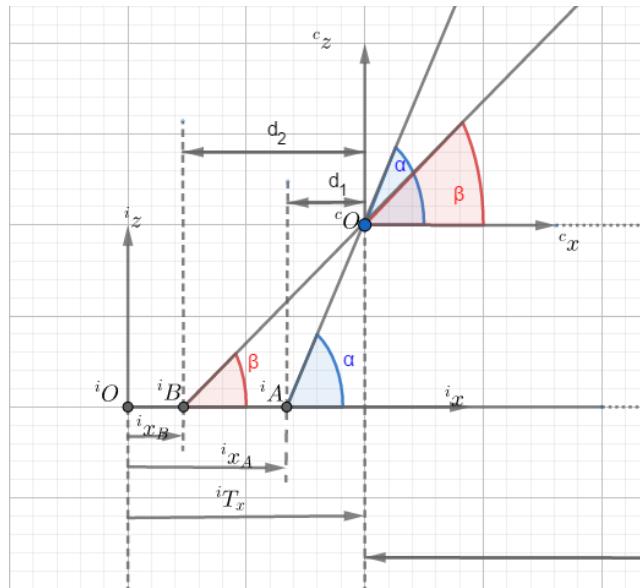


Fig. 1.10. Punto P soggetto all'acquisizione dell'immagine da parte della fotocamera sinistra.

Il punto P posto inizialmente nella posizione A viene mappato sull'immagine secondo la direzione di proiezione centrale al punto ${}^iA'$, successivamente a seguito dello spostamento di P alla posizione B il suo punto di proiezione sul piano immagine si sposta al punto ${}^iB'$.

Si indicano ora con α e β gli angoli compresi fra le direzioni di proiezione e il piano immagine, mentre i valori d_1 e d_2 risultano pari ad:

$$\begin{aligned} d_1 &= {}^iT_x - {}^iA \\ d_2 &= {}^iT_x - {}^iB \end{aligned} \quad (1.43)$$

Se si considera ora il segmento congiungente il centro del pinhole con il punto ${}^iA'$ risulta:

$$\begin{aligned} {}^cO{}^iA' &= \sqrt{(d_1)^2 + c^2} \\ {}^cO{}^iA' \cdot \cos(\alpha) &= d_1 \end{aligned} \quad (1.44)$$

Dalle quali risulta calcolabile l'angolo α :

$$\alpha = \arccos\left(\frac{d_1}{{}^cO{}^iA'}\right) \quad (1.45)$$

Lo stesso ragionamento effettuato per il punto ${}^iB'$ permette di trovar le seguenti relazioni:

$$\begin{aligned}
 \overline{^cO^iB} &= \sqrt{(d_2)^2 + c^2} \\
 \overline{^cO^iB} \cdot \cos(\beta) &= d_2 \\
 \beta &= \arccos\left(\frac{d_2}{\overline{^cO^iB}}\right)
 \end{aligned} \tag{1.46}$$

Si consideri ora di costruire una retta nel sistema di riferimento fissato nel pinhole della fotocamera e passante per il punto iA , si ha in tal caso che l'equazione generale di tale retta sarebbe:

$${}^cZ = m_{i_A} \cdot {}^cX + {}^cC \tag{1.47}$$

Nella quale cC è il termine noto dell'equazione della retta e nulla ha a che vedere con la costante c pari alla distanza focale.

Nel nostro caso poiché la retta passa per l'origine del pinhole risulta:

$$\begin{aligned}
 m_{i_A} &= \operatorname{tg}(\alpha) \\
 {}^cC &= 0
 \end{aligned} \tag{1.48}$$

La retta passante per il punto iB invece è tale che valgono le seguenti relazioni:

$$\begin{aligned}
 {}^cZ &= m_{i_B} \cdot {}^cX + {}^cC \\
 m_{i_B} &= \operatorname{tg}(\beta) \\
 {}^cC &= 0
 \end{aligned} \tag{1.49}$$

Si ottengono in definitiva le due rette cercate:

$$\begin{aligned}
 {}^cZ &= m_{i_A} \cdot {}^cX \\
 {}^cZ &= m_{i_B} \cdot {}^cX
 \end{aligned} \tag{1.50}$$

In maniera analoga si consideri ora la seconda fotocamera fig 34:

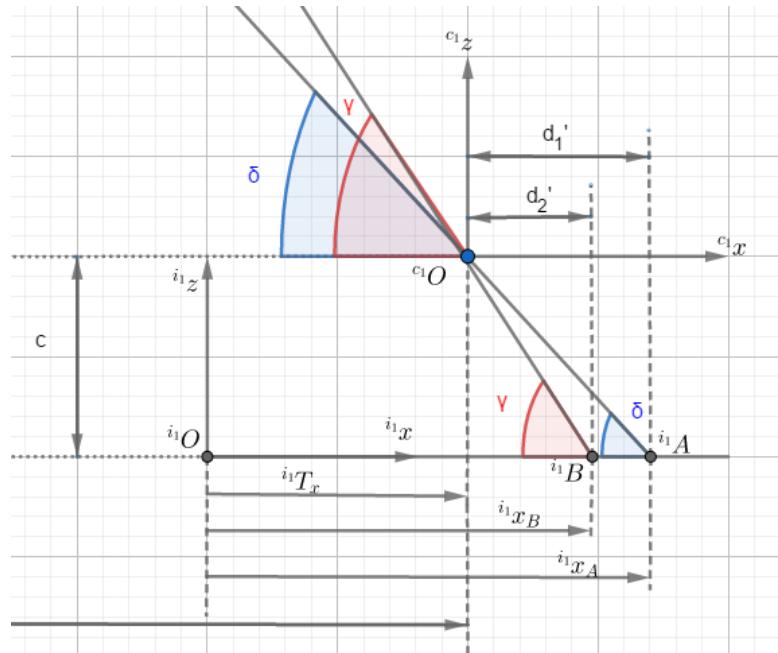


Fig.1.11. Punto P soggetto all'acquisizione dell'immagine da parte della fotocamera destra.

Valgono in questo caso le seguenti relazioni:

$$\begin{aligned}
 d'_1 &= {}^{i_1}x_A - {}^{i_1}T_x \\
 d'_2 &= {}^{i_1}x_B - {}^{i_1}T_x \\
 \overline{c_1 O {}^{i_1}A} &= \sqrt{(d'_1)^2 + c^2} \\
 \overline{c_1 O {}^{i_1}A} \cdot \cos(\delta) &= d'_1 \\
 \delta &= \arccos\left(\frac{d'_1}{\overline{c_1 O {}^{i_1}A}}\right) \\
 \overline{c_1 O {}^{i_1}B} &= \sqrt{(d'_2)^2 + c^2} \\
 \overline{c_1 O {}^{i_1}B} \cdot \cos(\gamma) &= d'_2 \\
 \gamma &= \arccos\left(\frac{d'_2}{\overline{c_1 O {}^{i_1}B}}\right)
 \end{aligned} \tag{1.51}$$

Che permettono di calcolare le due rette passanti per i punti ${}^{i_1}A$ e ${}^{i_1}B$ e l'origine del sistema di riferimento centrato nel pinhole della seconda fotocamera:

$$\begin{aligned}
{}^c z - {}^c z_{cl O} &= m_{il A} \left({}^c x - {}^c x_{cl O} \right) \\
{}^c z - {}^c z_{cl O} &= m_{il B} \left({}^c x - {}^c x_{cl O} \right) \\
{}^c z_{cl O} &= 0 \\
m_{il A} &= \operatorname{tg}(180 - \delta) \\
m_{il B} &= \operatorname{tg}(180 - \gamma) \\
{}^c x_{cl O} &= d
\end{aligned} \tag{1.51}$$

Si ottengono in questo caso le seguenti espressioni per le due rette:

$$\begin{aligned}
{}^c z &= m_{il A} \left({}^c x - d \right) \\
{}^c z &= m_{il B} \left({}^c x - d \right)
\end{aligned} \tag{1.52}$$

Studiando ora l'intersezione fra le rette passanti per il punto P quando esso si trova alla posizione A, si ottiene:

$$\begin{cases} {}^c z = m_{i A} \cdot {}^c x \\ {}^c z = m_{il A} \left({}^c x - d \right) \end{cases} = \begin{cases} {}^c z = m_{i A} \cdot {}^c x \\ m_{i A} \cdot {}^c x = m_{il A} \cdot {}^c x - m_{il A} \cdot d \end{cases} = \begin{cases} {}^c x_A = \frac{m_{il A} \cdot d}{m_{il A} - m_{i A}} \\ {}^c z_A = \frac{m_{il A} \cdot d}{m_{il A} - m_{i A}} \cdot m_{i A} \end{cases} \tag{1.53}$$

Tale operazione permette quindi di ottenere la posizione del punto P quando quest'ultimo si trova nella posizione A, espressa in un sistema di riferimento globale centrato sul pinhole della prima fotocamera.

L'intersezione delle rette passanti per il punto P quando si trova alla posizione B permette invece di ottenere la posizione di P in B sempre riferita allo stesso sistema di riferimento:

$$\begin{cases} {}^c z = m_{i B} \cdot {}^c x \\ {}^c z = m_{il B} \left({}^c x - d \right) \end{cases} = \begin{cases} {}^c z = m_{i B} \cdot {}^c x \\ m_{i B} \cdot {}^c x = m_{il B} \cdot {}^c x - m_{il B} \cdot d \end{cases} = \begin{cases} {}^c x_B = \frac{m_{il B} \cdot d}{m_{il B} - m_{i B}} \\ {}^c z_B = \frac{m_{il B} \cdot d}{m_{il B} - m_{i B}} \cdot m_{i B} \end{cases} \tag{1.54}$$

Lo spostamento globale di P risulta quindi presto determinato come riportato di seguito:

$$\begin{aligned}
W &= {}^c z_B - {}^c z_A \\
U &= {}^c x_B - {}^c x_A \\
V &= {}^c y_B - {}^c y_A
\end{aligned} \tag{1.55}$$

dove lo spostamento V nella (1.55) è calcolato in maniera analoga facendo le stesse considerazioni nel piano zy.

Come si può notare perciò l'effettivo calcolo degli spostamenti lungo le tre direzioni presuppone in primo luogo la determinazione di punti di corrispondenza fra le due immagini: si necessita in pratica della determinazione di un criterio che permetta di riconoscere l'identità di un stesso punto in immagini diverse; in secondo luogo si ha la necessità di conoscere lo spostamento che esso subisce nelle due immagini considerate singolarmente.

Il primo problema viene affrontato nel contesto della geometria epipolare la quale offre uno strumento altamente ottimizzato in termini di performance tempistiche e quantità di calcolo necessario sia alla ricerca che all'effettivo riconoscimento di punti di corrispondenza.

Il secondo problema invece viene affrontato nel contesto della correlazione digitale d'immagine bidimensionale che permette di trovare gli spostamenti sui piani immagine delle due fotocamere.

Tuttavia, poiché l'intero processo risulta dipendente dalla previa conoscenza sia dei parametri intrinseci delle due fotocamere separatamente considerate, che dalla conoscenza della loro relativa posizione ed orientazione, si affronta in primo luogo il problema della calibrazione della singola fotocamera, risolto il quale sarà poi possibile ottenere tutti i parametri del sistema a due fotocamere mediante la calibrazione stereo.

Per ulteriori approfondimenti sulla tecnica DIC3D e sue applicazioni si rimanda alla bibliografia [27],[28].

1.2 Calibrazione della singola fotocamera

In relazione alla problematica affrontata si presentano di seguito due metodi per la risoluzione del problema di calibrazione:

- Algoritmo Direct Linear Transform (DLT)[1,2]: presuppone la conoscenza esatta dei vettori posizione dei punti utilizzati nei calcoli sia nel sistema di riferimento reale ${}^w\{\text{XYZ}\}$ che in quello del sensore ${}^s\{\text{XYZ}\}$, e permette di ottenere direttamente la matrice di proiezione [P].
- Metodo di Zang [3,4]: semplifica la trasformazione proiettiva in omografica mediante un'opportuno posizionamento della terna ${}^w\{\text{XYZ}\}$ rispetto alle posizioni dei punti utilizzati nei calcoli, e permette di ottenere quindi la matrice della trasformazione omografica.

Al termine di entrambi i procedimenti si ottengono quindi le matrici associate alla trasformazione complessiva, sia essa proiettiva oppure omografica, che descrivono la mappatura dal mondo reale a quello del sensore.

La determinazione dei singoli parametri estrinseci ed intrinseci risulterà perciò postuma ad un'opportuna scomposizione delle matrici risultanti dai metodi sopraelencati.

1.2.1 Algoritmo Direct Linear Transform

Sia dato un punto P con vettore posizione ${}^w\mathbf{X}_P$ nel sistema di riferimento ${}^w\{\text{XYZ}\}$, ed ${}^s\mathbf{X}_P$ nel sistema di riferimento ${}^s\{\text{XYZ}\}$. Come precedentemente trovato vale la relazione (1.28) che può essere riscritta come:

$$\begin{bmatrix} {}^s\mathbf{X}_P \\ {}^s\mathbf{y}_P \\ 1 \end{bmatrix} = [\mathbf{P}] \begin{bmatrix} {}^w\mathbf{X}_P \\ {}^w\mathbf{y}_P \\ {}^w\mathbf{Z}_P \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} {}^w\mathbf{X}_P \\ {}^w\mathbf{y}_P \\ {}^w\mathbf{Z}_P \\ 1 \end{bmatrix} \quad (1.2.1)$$

Per ogni punto P risultano quindi impostabili due equazioni che legano le coordinate sul piano sensore e quelle nel sistema reale, si ha infatti sviluppando il precedente prodotto matriciale il seguente risultato:

$$\begin{cases} {}^s\mathbf{x}_p = {}^w\mathbf{x}_p \cdot p_{11} + {}^w\mathbf{y}_p \cdot p_{12} + {}^w\mathbf{z}_p \cdot p_{13} + p_{14} \\ {}^s\mathbf{y}_p = {}^w\mathbf{x}_p \cdot p_{21} + {}^w\mathbf{y}_p \cdot p_{22} + {}^w\mathbf{z}_p \cdot p_{23} + p_{24} \\ 1 = {}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34} \end{cases} \quad (1.2.2)$$

Invertendo ora la notazione omogenea si ottiene:

$$\begin{cases} {}^s\mathbf{x}_p = \frac{{}^w\mathbf{x}_p \cdot p_{11} + {}^w\mathbf{y}_p \cdot p_{12} + {}^w\mathbf{z}_p \cdot p_{13} + p_{14}}{{}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34}} \\ {}^s\mathbf{y}_p = \frac{{}^w\mathbf{x}_p \cdot p_{21} + {}^w\mathbf{y}_p \cdot p_{22} + {}^w\mathbf{z}_p \cdot p_{23} + p_{24}}{{}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34}} \end{cases} \quad (1.2.3)$$

Riscrivendo il sistema (1.2.3) in forma nella seguente maniera:

$$\begin{cases} {}^s\mathbf{x}_p \cdot ({}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34}) = {}^w\mathbf{x}_p \cdot p_{11} + {}^w\mathbf{y}_p \cdot p_{12} + {}^w\mathbf{z}_p \cdot p_{13} + p_{14} \\ {}^s\mathbf{y}_p \cdot ({}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34}) = {}^w\mathbf{x}_p \cdot p_{21} + {}^w\mathbf{y}_p \cdot p_{22} + {}^w\mathbf{z}_p \cdot p_{23} + p_{24} \end{cases} \quad (1.2.4)$$

$$\begin{cases} {}^s\mathbf{x}_p \cdot ({}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34}) - {}^w\mathbf{x}_p \cdot p_{11} - {}^w\mathbf{y}_p \cdot p_{12} - {}^w\mathbf{z}_p \cdot p_{13} - p_{14} = 0 \\ {}^s\mathbf{y}_p \cdot ({}^w\mathbf{x}_p \cdot p_{31} + {}^w\mathbf{y}_p \cdot p_{32} + {}^w\mathbf{z}_p \cdot p_{33} + p_{34}) - {}^w\mathbf{x}_p \cdot p_{21} - {}^w\mathbf{y}_p \cdot p_{22} - {}^w\mathbf{z}_p \cdot p_{23} - p_{24} = 0 \end{cases} \quad (1.2.5)$$

ed introducendo ora i seguenti vettori:

$$\begin{aligned}\bar{x}_p^t &= [-{}^w x_p, -{}^w y_p, -{}^w z_p, -1, 0, 0, 0, 0, {}^s x_p \cdot {}^w x_p, {}^s x_p \cdot {}^w y_p, {}^s x_p \cdot {}^w z_p, {}^s x_p] \\ \bar{y}_p^t &= [0, 0, 0, 0, -{}^w x_p, -{}^w y_p, -{}^w z_p, -1, {}^s y_p \cdot {}^w x_p, {}^s y_p \cdot {}^w y_p, {}^s y_p \cdot {}^w z_p, {}^s y_p] \\ p^t &= [p_{11}, p_{12}, p_{13}, p_{14}, p_{21}, p_{22}, p_{23}, p_{24}, p_{31}, p_{32}, p_{33}, p_{34}]\end{aligned}\quad (1.2.6)$$

si può riscrivere la (1.2.5) nella seguente forma:

$$\begin{cases} \bar{x}_p^t \cdot p = 0 \\ \bar{y}_p^t \cdot p = 0 \end{cases}\quad (1.2.7)$$

La relazione (1.2.7) dimostra che per il singolo punto si possono ottenere al massimo 2 equazioni in 12 incognite, e che di conseguenza il problema della determinazione della matrice di proiezione richiede la conoscenza della posizione reale di almeno 6 punti dell'immagine.

Risulta tuttavia che al fine di ottenere una soluzione più robusta si utilizza un numero di punti molto maggiore, in tale caso si può scrivere la seguente espressione matriciale:

$$[A]_{2n \times 12} \{p\}_{12 \times 1} = 0 \quad (1.2.8)$$

con la matrice A pari ad:

$$[A] = \begin{bmatrix} -{}^w x_{p1}, -{}^w y_{p1}, -{}^w z_{p1}, -1, 0, 0, 0, 0, {}^s x_{p1} \cdot {}^w x_{p1}, {}^s x_{p1} \cdot {}^w y_{p1}, {}^s x_{p1} \cdot {}^w z_{p1}, {}^s x_{p1} \\ 0, 0, 0, 0, -{}^w x_{p1}, -{}^w y_{p1}, -{}^w z_{p1}, -1, {}^s y_{p1} \cdot {}^w x_{p1}, {}^s y_{p1} \cdot {}^w y_{p1}, {}^s y_{p1} \cdot {}^w z_{p1}, {}^s y_{p1} \\ -{}^w x_{p2}, -{}^w y_{p2}, -{}^w z_{p2}, -1, 0, 0, 0, 0, {}^s x_{p2} \cdot {}^w x_{p2}, {}^s x_{p2} \cdot {}^w y_{p2}, {}^s x_{p2} \cdot {}^w z_{p2}, {}^s x_{p2} \\ 0, 0, 0, 0, -{}^w x_{p2}, -{}^w y_{p2}, -{}^w z_{p2}, -1, {}^s y_{p2} \cdot {}^w x_{p2}, {}^s y_{p2} \cdot {}^w y_{p2}, {}^s y_{p2} \cdot {}^w z_{p2}, {}^s y_{p2} \\ \vdots \\ -{}^w x_{pn}, -{}^w y_{pn}, -{}^w z_{pn}, -1, 0, 0, 0, 0, {}^s x_{pn} \cdot {}^w x_{pn}, {}^s x_{pn} \cdot {}^w y_{pn}, {}^s x_{pn} \cdot {}^w z_{pn}, {}^s x_{pn} \\ 0, 0, 0, 0, -{}^w x_{pn}, -{}^w y_{pn}, -{}^w z_{pn}, -1, {}^s y_{pn} \cdot {}^w x_{pn}, {}^s y_{pn} \cdot {}^w y_{pn}, {}^s y_{pn} \cdot {}^w z_{pn}, {}^s y_{pn} \end{bmatrix}\quad (1.2.9)$$

Il vettore $\{p\}$ invece è costruito con gli elementi della matrice di proiezione nel seguente modo:

$$p^t = [p_{11}, p_{12}, p_{13}, p_{14}, p_{21}, p_{22}, p_{23}, p_{24}, p_{31}, p_{32}, p_{33}, p_{34}] \quad (1.2.10)$$

In pratica quello che si vuole fare è una minimizzazione quadratica dell'errore della (1.2.8).

Indicando l'errore sul calcolo col termine ξ si ottiene la seguente relazione:

$$[A] \cdot \{p\} = \{\xi\} \quad (1.2.11)$$

Il problema viene perciò posto nei seguenti termini: si deve ricercare il vettore $\{\tilde{p}\}$ soluzione della (1.2.8) tale che esso minimizzi l'errore quadratico sul calcolo:

$$\tilde{p} = \operatorname{argmin}_{\xi} \xi^t \cdot \xi \quad (1.2.12)$$

con la seguente ulteriore condizione legata alla notazione omogenea nella quale è espresso il vettore p (si ricorda infatti che in coordinate omogenee ogni vettore è pari a se stesso a meno di un fattore di scala) e riportata di seguito:

$$\|\tilde{p}\|_2 = 1 = \sum \tilde{p}_{ij}^2 = 1 \quad (1.2.13)$$

in modo da trovare la soluzione normalizzata.

Matematicamente risulta che l'autovettore associato al più piccolo valore singolare della matrice $[A]$ è tale da minimizzare l'errore ξ , fatto che permette di concludere che il vettore $\{\tilde{p}\}$ cercato è pari a tale autovettore trovato (si veda a tal proposito la letteratura inerente il metodo SVD per la risoluzione di problemi inerenti la minimizzazione quadratica).

Gli elementi del vettore $\{\tilde{p}\}$ come visto nella (1.2.10) sono esattamente quelli della matrice di proiezione $[P]$, la quale contiene tutti i parametri instrinseci ed estrinseci del sistema di acquisizione, vale a dire:

$$[P] = \begin{bmatrix} \tilde{p}_{11} & \tilde{p}_{12} & \tilde{p}_{13} & \tilde{p}_{14} \\ \tilde{p}_{21} & \tilde{p}_{22} & \tilde{p}_{23} & \tilde{p}_{24} \\ \tilde{p}_{31} & \tilde{p}_{32} & \tilde{p}_{33} & \tilde{p}_{34} \end{bmatrix} \quad (1.2.14)$$

Effettuata perciò la scomposizione al valore singolare della matrice [A], trovato il vettore soluzione ai minimi quadrati $\{\tilde{p}\}$, e ricostruita la matrice di proiezione [P], si può procedere alla sua scomposizione al fine di estrarre la matrice di calibrazione [K], quella di rotazione [R], ed il vettore posizione ${}^wX_{c_0}$.

Considerando la matrice di proiezione nella forma generale, ottenuta precedentemente:

$$[P] = \begin{bmatrix} S_x \cdot c & -\frac{1}{\tan(\theta)} \cdot c & -S_x \left({}^iT_x - \frac{{}^iT_y}{\tan(\theta)} \right) & 0 \\ 0 & \frac{S_y}{\sin(\theta)} \cdot c & -\frac{S_y}{\sin(\theta)} {}^iT_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -{}^wX_{c_0} \\ 0 & 1 & 0 & -{}^wY_{c_0} \\ 0 & 0 & 1 & -{}^wZ_{c_0} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.2.15)$$

Si può riscrivere quest'ultima in maniera più compatta come di seguito:

$$[P] = [[K] \ 0] \begin{bmatrix} {}^c[R]_w & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} [I] & -{}^wX_{c_0} \\ 0 & 1 \end{bmatrix} = [[K] \cdot {}^c[R]_w \ 0] \begin{bmatrix} [I] & -{}^wX_{c_0} \\ 0 & 1 \end{bmatrix} \quad (1.2.16)$$

$$[P] = [[K] \cdot {}^c[R]_w \ - [K] \cdot {}^c[R]_w \cdot {}^wX_{c_0}] \quad (1.2.17)$$

Confrontando quindi la matrice di proiezione [P] calcolata con la DLT e la forma generale data dalla (1.2.17) si ottengono le seguenti condizioni:

$$\begin{bmatrix} \tilde{p}_{11} & \tilde{p}_{12} & \tilde{p}_{13} \\ \tilde{p}_{21} & \tilde{p}_{22} & \tilde{p}_{23} \\ \tilde{p}_{31} & \tilde{p}_{32} & \tilde{p}_{33} \end{bmatrix} = [\tilde{P}_{KR}] = [K] \cdot {}^C[R]_W$$

(1.2.18)

$$\begin{bmatrix} \tilde{p}_{14} \\ \tilde{p}_{24} \\ \tilde{p}_{34} \end{bmatrix} = \{\tilde{p}_c\} = -[K] \cdot {}^C[R]_W \cdot {}^W X_c$$

(1.2.19)

Come si può notare dalla condizione (1.2.18), la sottomatrice di proiezione $[\tilde{P}_{KR}]$ equivale al prodotto della matrice di calibrazione $[K]$, che si ricorda essere triangolare superiore, e della matrice di rotazione $[R]$, che risulta essere ortogonale.

Alla luce di queste considerazioni risulta che la decomposizione QR di $[\tilde{P}_{KR}]$ permette di ottenere sia la matrice di calibrazione che quella di rotazione, note le quali si può calcolare il valore ${}^W X_c$:

$${}^W X_c = (-[K] \cdot {}^C[R]_W)^{-1} \cdot \{\tilde{p}_c\}$$

(1.2.20)

In definitiva risultano quindi trovati tutti i parametri sia estrinseci che intrinseci della fotocamera.

Bisogna precisare tuttavia, riguardo all'algoritmo DLT, che in caso i punti considerati nei calcoli giacciono su un piano nel sistema di riferimento globale ${}^W\{OXYZ\}$, o comunque siano caratterizzati da un valore ${}^W z_{pi}$ uguale per tutti, non risulta possibile trovare la soluzione al problema del calcolo della matrice di proiezione a causa del rank deficit della matrice $[A]$. Tutti i punti quindi devono trovarsi a posizioni diverse ed inoltre risulta chiaro come i risultati ottenuti dipenderanno in primo luogo dalla precisione con i quali questi possono essere localizzati.

1.2.2 Algoritmo di Zhang.

Il metodo proposto da Z. Zhang semplifica la trasformazione proiettiva in omografica imponendo la condizione che i punti da utilizzare nei calcoli siano confinati a stare su un piano Π nel sistema di riferimento globale ${}^W\{OXYZ\}$, il quale sistema viene quindi posizionato su Π in modo che la coordinata Z risulti identicamente nulla per ogni punto considerato nei calcoli, come schematizzato di seguito in figura 1.12:

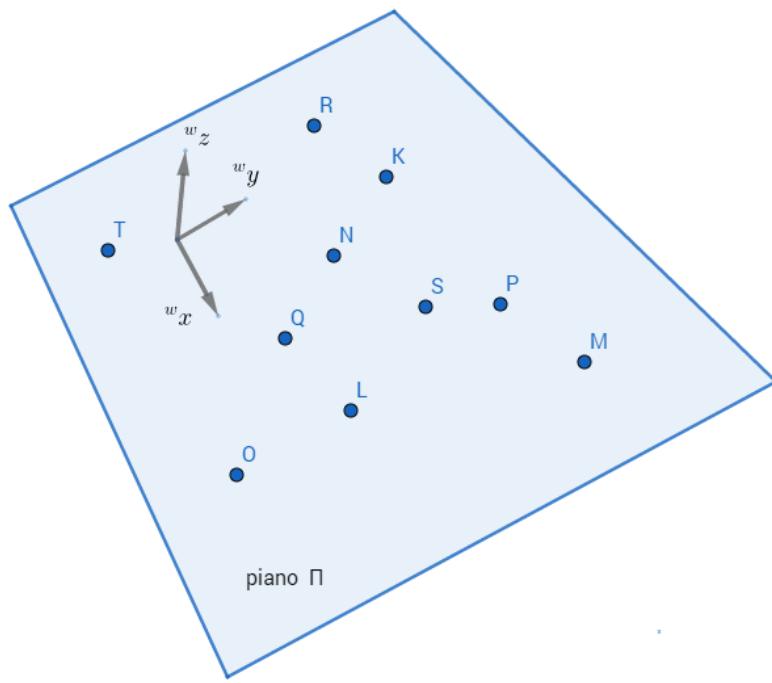


Fig.1.12. Insieme di punti sul piano Π utilizzati nel processo di calibrazione, ed aventi quota nulla rispetto al sistema di riferimento globale.

In tal caso la relazione sulla trasformazione di proiezione risulta semplificata come si riporta di seguito:

$$\begin{bmatrix} {}^Sx_p \\ {}^Sx_p \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} {}^Wx_p \\ {}^Wy_p \\ 1 \end{bmatrix} \quad (1.2.21)$$

Essendo nulla la quota z di tutti i punti si elimina la terza colonna della matrice di proiezione e il componente relativo alla quota nel vettore posizione dei punti nel sistema di riferimento globale.

Ciò permette di riscrivere la relazione (1.2.21) nella seguente maniera:

$$\begin{bmatrix} {}^s \mathbf{x}_p \\ {}^s \mathbf{y}_p \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{bmatrix} {}^w \mathbf{x}_p \\ {}^w \mathbf{y}_p \\ 1 \end{bmatrix} = [\mathbf{O}] \begin{bmatrix} {}^w \mathbf{x}_p \\ {}^w \mathbf{y}_p \\ 1 \end{bmatrix} \quad (1.2.22)$$

Impostando ora in maniera analoga al metodo DLT i seguenti vettori:

$$\begin{aligned} \bar{\mathbf{x}}_p^t &= [-{}^w \mathbf{x}_p, -{}^w \mathbf{y}_p, -1, 0, 0, 0, {}^s \mathbf{x}_p \cdot {}^w \mathbf{x}_p, {}^s \mathbf{x}_p \cdot {}^w \mathbf{y}_p, {}^s \mathbf{x}_p] \\ \bar{\mathbf{y}}_p^t &= [0, 0, 0, -{}^w \mathbf{x}_p, -{}^w \mathbf{y}_p, -1, {}^s \mathbf{y}_p \cdot {}^w \mathbf{x}_p, {}^s \mathbf{y}_p \cdot {}^w \mathbf{y}_p, {}^s \mathbf{y}_p] \end{aligned} \quad (1.2.23)$$

risulta che per ogni punto si può impostare il seguente sistema di equazioni:

$$\begin{cases} \bar{\mathbf{x}}_p^t \cdot \mathbf{p} = 0 \\ \bar{\mathbf{y}}_p^t \cdot \mathbf{p} = 0 \end{cases} \quad (1.2.24)$$

che per un numero di punti N può essere riscritto nel seguente modo:

$$[\mathbf{A}]_{2nx9} \{ \mathbf{p} \}_{9x1} = 0 \quad (1.2.25)$$

con la matrice [A] pari in questo caso ad:

$$[\mathbf{A}] = \begin{bmatrix} -{}^w \mathbf{x}_{p1}, -{}^w \mathbf{y}_{p1}, -1, 0, 0, 0, {}^s \mathbf{x}_{p1} \cdot {}^w \mathbf{x}_{p1}, {}^s \mathbf{x}_{p1} \cdot {}^w \mathbf{y}_{p1}, {}^s \mathbf{x}_{p1} \\ 0, 0, 0, -{}^w \mathbf{x}_{p1}, -{}^w \mathbf{y}_{p1}, -1, {}^s \mathbf{y}_{p1} \cdot {}^w \mathbf{x}_{p1}, {}^s \mathbf{y}_{p1} \cdot {}^w \mathbf{y}_{p1}, {}^s \mathbf{y}_{p1} \\ -{}^w \mathbf{x}_{p2}, -{}^w \mathbf{y}_{p2}, -1, 0, 0, 0, {}^s \mathbf{x}_{p2} \cdot {}^w \mathbf{x}_{p2}, {}^s \mathbf{x}_{p2} \cdot {}^w \mathbf{y}_{p2}, {}^s \mathbf{x}_{p2} \\ 0, 0, 0, -{}^w \mathbf{x}_{p2}, -{}^w \mathbf{y}_{p2}, -1, {}^s \mathbf{y}_{p2} \cdot {}^w \mathbf{x}_{p2}, {}^s \mathbf{y}_{p2} \cdot {}^w \mathbf{y}_{p2}, {}^s \mathbf{y}_{p2} \\ \vdots \\ -{}^w \mathbf{x}_{pn}, -{}^w \mathbf{y}_{pn}, -1, 0, 0, 0, {}^s \mathbf{x}_{pn} \cdot {}^w \mathbf{x}_{pn}, {}^s \mathbf{x}_{pn} \cdot {}^w \mathbf{y}_{pn}, {}^s \mathbf{x}_{pn} \\ 0, 0, 0, -{}^w \mathbf{x}_{pn}, -{}^w \mathbf{y}_{pn}, -1, {}^s \mathbf{y}_{pn} \cdot {}^w \mathbf{x}_{pn}, {}^s \mathbf{y}_{pn} \cdot {}^w \mathbf{y}_{pn}, {}^s \mathbf{y}_{pn} \end{bmatrix} \quad (1.2.26)$$

La risoluzione del sistema mediante SVD permette di trovare la soluzione ottimale che minimizza l'errore per il vettore $\{p\}$.

Noto $\{p\}$ risulta altresì nota la matrice dell'omografia della relazione iniziale (1.2.22), essendo questa costruita con gli elementi di $[O]$.

A questo punto risulta che ai fini della determinazione della matrice di calibrazione e di rotazione non si può più applicare la scomposizione QR della matrice $[O]$ (equazione 1.2.22) non essendo più quest'ultima proporzionale al prodotto $[K][R]$, quindi non esprimibile come prodotto di una matrice triangolare per una matrice ortogonale.

Osservando tuttavia i diversi elementi di $[O]$ si possono trovare delle condizioni che permettono ancora di ricavare le matrici di calibrazione $[K]$ e di rotazione $[R]$ cercate nonché il vettore ${}^W X_{c_O}$.

Considerando infatti la relazione generale di proiezione ottenuta precedentemente e riscritta come:

$$[P] = [K] \begin{bmatrix} r_{11} & r_{12} & r_{13} & - (r_{11} {}^W X_{c_O} + r_{12} {}^W Y_{c_O} + r_{13} {}^W Z_{c_O}) \\ r_{21} & r_{22} & r_{23} & - (r_{21} {}^W X_{c_O} + r_{22} {}^W Y_{c_O} + r_{23} {}^W Z_{c_O}) \\ r_{31} & r_{32} & r_{33} & - (r_{31} {}^W X_{c_O} + r_{32} {}^W Y_{c_O} + r_{33} {}^W Z_{c_O}) \end{bmatrix} \quad (1.2.27)$$

la condizione dell'appartenenza dei punti al piano, e quindi aventi tutti quota nulla modifica $[P]$ come di seguito:

$$[P] = [K] \cdot \begin{bmatrix} r_{11} & r_{12} & - (r_{11} {}^W X_{c_O} + r_{12} {}^W Y_{c_O} + r_{13} {}^W Z_{c_O}) \\ r_{21} & r_{22} & - (r_{21} {}^W X_{c_O} + r_{22} {}^W Y_{c_O} + r_{23} {}^W Z_{c_O}) \\ r_{31} & r_{32} & - (r_{31} {}^W X_{c_O} + r_{32} {}^W Y_{c_O} + r_{33} {}^W Z_{c_O}) \end{bmatrix} \quad (1.2.28)$$

Risulta inoltre che la matrice dell'omografia appena calcolata è pari alla matrice $[P]$, fatto che permette di scrivere:

$$\begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} = [K] \cdot \begin{bmatrix} r_{11} & r_{12} & -(r_{11}^w x_{c_0} + r_{12}^w y_{c_0} + r_{13}^w z_{c_0}) \\ r_{21} & r_{22} & -(r_{21}^w x_{c_0} + r_{22}^w y_{c_0} + r_{23}^w z_{c_0}) \\ r_{31} & r_{32} & -(r_{31}^w x_{c_0} + r_{32}^w y_{c_0} + r_{33}^w z_{c_0}) \end{bmatrix} \quad (1.2.29)$$

L'analisi della (1.2.29) permette di osservare come le prime due colonne della matrice dell'omografia calcolata siano pari ad:

$$\begin{bmatrix} p_{11} \\ p_{21} \\ p_{31} \end{bmatrix} = [K] \cdot \begin{bmatrix} r_{11} \\ r_{21} \\ r_{31} \end{bmatrix} ; \quad \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix} = [K] \cdot \begin{bmatrix} r_{12} \\ r_{22} \\ r_{32} \end{bmatrix} \quad (1.2.30)$$

Che permette di esprimere le colonne della matrice di rotazione in funzione della matrice di calibrazione e dei componenti della matrice d'omografia [O] calcolata:

$$\begin{bmatrix} r_{11} \\ r_{21} \\ r_{31} \end{bmatrix} = \{r_1\} = [K]^{-1} \cdot \begin{bmatrix} p_{11} \\ p_{21} \\ p_{31} \end{bmatrix} ; \quad \begin{bmatrix} r_{12} \\ r_{22} \\ r_{32} \end{bmatrix} = \{r_2\} = [K]^{-1} \cdot \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix} \quad (1.2.31)$$

Siccome i vettori $\{r_1\}$ ed $\{r_2\}$ sono componenti della matrice di rotazione $[R]^w$ valgono le seguenti relazioni vista l'ortogonalità degli autovettori della matrice di rotazione:

$$\begin{aligned} \{r_1\} \cdot \{r_2\} &= \{r_1\}^t \{r_2\} = 0 \\ \|r_1\| &= \|r_2\| = 1 \end{aligned} \quad (1.2.32)$$

Sostituendo ora le espressioni per $\{r_1\}$ ed $\{r_2\}$ dalla (1.2.31) nella (1.2.32) si può impostare il seguente sistema di equazioni:

$$\left\{ \begin{array}{l} \begin{bmatrix} p_{11} & p_{21} & p_{31} \end{bmatrix} \cdot [K^{-1}]^t \cdot [K^{-1}] \cdot \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix} = 0 \\ \\ \begin{bmatrix} p_{11} & p_{21} & p_{31} \end{bmatrix} \cdot [K^{-1}]^t \cdot [K^{-1}] \cdot \begin{bmatrix} p_{11} \\ p_{21} \\ p_{31} \end{bmatrix} - \begin{bmatrix} p_{12} & p_{22} & p_{32} \end{bmatrix} \cdot [K^{-1}]^t \cdot [K^{-1}] \cdot \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix} = 0 \end{array} \right. \quad (1.2.33)$$

Indicando ora il prodotto:

$$[\mathbf{K}^{-1}]^t \cdot [\mathbf{K}^{-1}] = [\mathbf{B}] \quad (1.2.34)$$

Si può riscrivere il sistema come:

$$\begin{cases} \begin{bmatrix} p_{11} & p_{21} & p_{31} \end{bmatrix} \cdot [\mathbf{B}] \cdot \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix} = \mathbf{0} \\ \begin{bmatrix} p_{11} & p_{21} & p_{31} \end{bmatrix} \cdot [\mathbf{B}] \cdot \begin{bmatrix} p_{11} \\ p_{21} \\ p_{31} \end{bmatrix} - \begin{bmatrix} p_{12} & p_{22} & p_{32} \end{bmatrix} \cdot [\mathbf{B}] \cdot \begin{bmatrix} p_{12} \\ p_{22} \\ p_{32} \end{bmatrix} = \mathbf{0} \end{cases} \quad (1.2.35)$$

Dall'espressione (1.2.34) si nota che la matrice $[\mathbf{B}]$ risulta una matrice ortogonale tale che dei 9 elementi solo sei sono indipendenti:

$$[\mathbf{B}] = \begin{bmatrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \mathbf{b}_{13} \\ \mathbf{b}_{12} & \mathbf{b}_{22} & \mathbf{b}_{23} \\ \mathbf{b}_{13} & \mathbf{b}_{23} & \mathbf{b}_{33} \end{bmatrix} \quad (1.2.36)$$

Impostando perciò i seguenti vettori:

$$\mathbf{b}^t = [b_{11}, b_{12}, b_{13}, b_{22}, b_{23}, b_{33}]$$

$$\mathbf{h}_{11}^t = [p_{11}^2, 2 \cdot p_{11} \cdot p_{21}, 2 \cdot p_{31} \cdot p_{11}, p_{21}^2, 2 \cdot p_{21} \cdot p_{31}, p_{31}^2]$$

$$\mathbf{h}_{12}^t = [p_{11} \cdot p_{12}, p_{11} \cdot p_{22} + p_{21} \cdot p_{12}, p_{31} \cdot p_{12} + p_{11} \cdot p_{32}, p_{21} \cdot p_{22}, p_{31} \cdot p_{22} + p_{21} \cdot p_{32}, p_{31} \cdot p_{32}]$$

$$\mathbf{h}_{22}^t = [p_{12}^2, 2 \cdot p_{12} \cdot p_{22}, 2 \cdot p_{32} \cdot p_{12}, p_{22}^2, 2 \cdot p_{22} \cdot p_{32}, p_{32}^2] \quad (1.2.37)$$

Si può riscrivere il sistema in maniera più compatta come di seguito:

$$\begin{cases} \mathbf{h}_{11}^t \cdot \mathbf{b} = \mathbf{0} \\ (\mathbf{h}_{11}^t - \mathbf{h}_{22}^t) \cdot \mathbf{b} = \mathbf{0} \end{cases} \quad (1.2.38)$$

La cui soluzione mediante SVD permette di ottenere il vettore $\{\mathbf{b}\}$ cercato e quindi la matrice $[\mathbf{B}] = [\mathbf{K}^{-1}]^t [\mathbf{K}^{-1}]$.

Per come è definita $[B]$ risulta in definitiva che la decomposizione di Cholesky permette di estrapolare la matrice di calibrazione $[K]$ da quest'ultima, risulta infatti:

$$\begin{aligned} \text{chol}([B]) &= [F] \\ [F] \cdot [F]^t &= [B] \\ [B] &= [K^{-1}]^t \cdot [K^{-1}] \end{aligned} \quad (1.2.39)$$

La (1.2.39) permette di concludere che:

$$[K^{-1}]^t = [F] \quad (1.2.40)$$

con la matrice $[F]$ pari al risultato della scomposizione di Cholesky della matrice $[B]$.

Trovata quindi la matrice di calibrazione risultano determinabili anche tutti i parametri estrinseci del sistema d'acquisizione immagine: dalle relazioni (1.2.31) si ottengono le due colonne della matrice di rotazione ${}^c[R]_w$, successivamente la terza colonna è determinata come prodotto vettoriale delle precedenti, e risulta anche calcolabile il vettore posizione del pinhole della fotocamera dalle seguenti relazioni:

$$\begin{aligned} \{r_3\} &= \{r_1\} x \{r_3\} \\ {}^wX_{c_O} &= (-[K] \cdot [{}^cR_w])^{-1} \cdot \begin{bmatrix} p_{13} \\ p_{23} \\ p_{33} \end{bmatrix} \end{aligned} \quad (1.2.41)$$

Per ogni immagine quindi si possono impostare due equazioni come espresso dalle (1.2.35) ed (1.2.38).

Siccome la matrice $[B]$ consta di 6 elementi indipendenti risulta chiaro che il problema è risolvibile utilizzando al minimo 3 immagini con viste differenti dello stesso oggetto.

1.2.3 Modello della distorsione.

Come affermato al paragrafo 1.1 relativamente alle assunzioni che stanno alla base del modello di acquisizione d'immagine, risulta ora chiaro come i metodi di calibrazione in generale non tengano conto di tutta una serie di fenomeni di disturbo, legati sia ai limiti tecnologici nella realizzazione delle lenti, dei sensori e dei diversi componenti meccanici, sia alle condizioni operative nelle quali le fotocamere si trovano ad operare. L'insieme di questi disturbi se non tenuti in conto compromette la validità dei risultati che si ottengono alla fine del processo di calibrazione.

Si è assunto per esempio fino ad ora che i piani immagine e sensore fossero perfettamente complanari, cosa che risulta impossibile da realizzare in modo esatto e che quindi genera una mancanza di parallelismo fra gli assi ottici; anche le caratteristiche delle lenti come il loro angolo di copertura, oppure la distanza focale c usata per una generica immagine possono generare degli effetti di distorsione rispetto alla semplice approssimazione dell'ottica di gauss che sta alla base della trattazione di calibrazione.

Si tiene pertanto in conto la presenza di questi fenomeni attraverso lo studio della distorsione che introducono nelle immagini finali mediante modelli non lineari.

Poiché esistono in letteratura una moltitudine di modelli le cui complessità variano a seconda della relativa applicazione, si riporta di seguito il modello di distorsione utilizzato dalla libreria OpenCV [6] (*Open Source Computer Vision*), sviluppata originariamente dalla *Intel* e che gode di ottima reputazione a livello globale.

Si considerano principalmente due tipologie di distorsione:

- “*Distorsione barrel*” modellata come:

$$\begin{aligned}{}^i x' &= {}^i x \cdot (1 + k_1 \cdot r^2 + k_2 \cdot r^4 + k_3 \cdot r^6) \\{}^i y' &= {}^i y \cdot (1 + k_1 \cdot r^2 + k_2 \cdot r^4 + k_3 \cdot r^6)\end{aligned}\quad (1.2.42)$$

- Distorsione tangenziale dovuta a mancanza di parallelismo fra gli assi ottici del piano immagine e lenti modellata come:

$$\begin{aligned}{}^i x' &= {}^i x + [2 \cdot \rho_1 \cdot {}^i x \cdot {}^i y + \rho_2 \cdot (r^2 + 2 \cdot {}^i x^2)] \\{}^i y' &= {}^i y + [2 \cdot \rho_2 \cdot {}^i x \cdot {}^i y + \rho_1 \cdot (r^2 + 2 \cdot {}^i y^2)]\end{aligned}\quad (1.2.43)$$

Nelle espressioni appena riportate si indica con “r” la distanza del generico punto rispetto al punto principale dell’immagine:

$$r = \sqrt{^i x^2 + ^i y^2}$$

Il problema si configura ora come la ricerca dei 5 parametri $k_1, k_2, k_3, \rho_1, \rho_2$ che minimizzano l’errore di riproiezione, espresso come la distanza fra i punti del piano immagine utilizzati per la calibrazione e quelli ottenuti dalla riproiezione sullo stesso piano effettuata sulla base della conoscenza dei parametri estrinseci ed intrinseci del sistema di acquisizione.

In altri termini, data una serie di punti P_i con vettori posizione ${}^w X_{pi}$ ed ${}^i X_{pi}$ noti e calcolate sulla base di questi la matrice di calibrazione $[K]$ della fotocamera, la matrice di rotazione ${}^C [R]_w$ ed il vettore posizione del sistema di riferimento della camera ${}^w X_{c_o}$, risulta possibile riproiettare i punti P_i su un generico piano immagine alle posizioni ${}^i X_{pin}$ in funzione dei parametri appena calcolati; considerando ora come errore la distanza fra la posizione dei punti P_i originali e quella dei punti P_{pin} a seguito della riproiezione si calcolano i parametri di distorsione e si correggono i valori dei parametri estrinseci ed intrinseci in modo da minimizzare l’errore di riproiezione:

$$\text{Errore Riproiezione} = \underset{X_{pin}}{\operatorname{argmin}} \sum_{j=1}^{N(\text{viste})} \sum_{i=1}^{N(\text{punti})} \|X_{pi} - X_{pin}\|^2 \quad (1.2.44)$$

Con:

$$X_{pin} = f(k_1, k_2, k_3, \rho_1, \rho_2, [K], [{}^C R_w], {}^w X_{c_o}) \quad (1.2.45)$$

In conclusione per ogni ulteriore approfondimento inerente la calibrazione della singola fotocamera e dei modelli di distorsione si rimanda alla bibliografia [1],[2],[3],[4],[5].

1.2.4 *Interest point detection.*

Come visto il problema della calibrazione si focalizza in prima istanza sulla definizione di una serie di punti dei quali si conosce sia la posizione nel sistema di riferimento globale che quella nella singola immagine.

Quanto affermato pone problematiche relative sia all' effettiva accuratezza con la quale gli strumenti di misura sono in grado di rilevare la posizione, ma anche all'effettiva possibilità di poterne associare successivamente la posizione ai corrispondenti punti sulle immagini.

In quest'ottica si può ben comprendere l'enorme successo della metodologia proposta da Zhang, in quanto semplifica di una dimensione il problema della definizione della posizione spaziale dei punti utilizzati nel calcolo della calibrazione: tutti i punti essendo costretti a trovarsi su un piano sul quale si fissa in maniera opportuna il sistema di riferimento globale sono accumunati dall' avere lo stesso valore finale della quota z, ed in particolare quota z pari a zero.

Trovata perciò una maniera semplificata per la determinazione della posizione spaziale dei punti, rimane ora da affrontare il problema relativo alla possibilità di riconoscerli nelle immagini in maniera automatizzata.

Si nota in generale che gli algoritmi sviluppati per la risoluzione del problema esposto basano il loro funzionamento su delle caratteristiche (*features*) che presentano determinate regioni nella generica immagine considerata.

In particolare si considerano le regioni per le quali non esiste il problema dell'apertura, e sono quindi caratterizzate dall'averne forti gradienti d'intensità nelle due direzioni del piano immagine.

Risulta perciò chiaro che al fine di sfruttare al meglio questa caratteristica per identificare i punti nelle immagini si utilizzano nel processo della calibrazione oggetti riportanti sulla superficie dei pattern o delle texture appositamente studiate, per le quali risulta possibile misurare in maniera semplificata sia la posizione dei punti in unità metriche sull'oggetto fisico, che determinarne la posizione nelle immagini acquisite.

Per la trattazione matematica dettagliata del problema si rimanda alla bibliografia [13],[14],[16].

Si presenta invece ora come esempio di applicazione di un algoritmo di point detection l'implementazione effettuata in matlab e riportata in appendice dell'*Harris corner detector* [13], applicato alla seguente immagine:

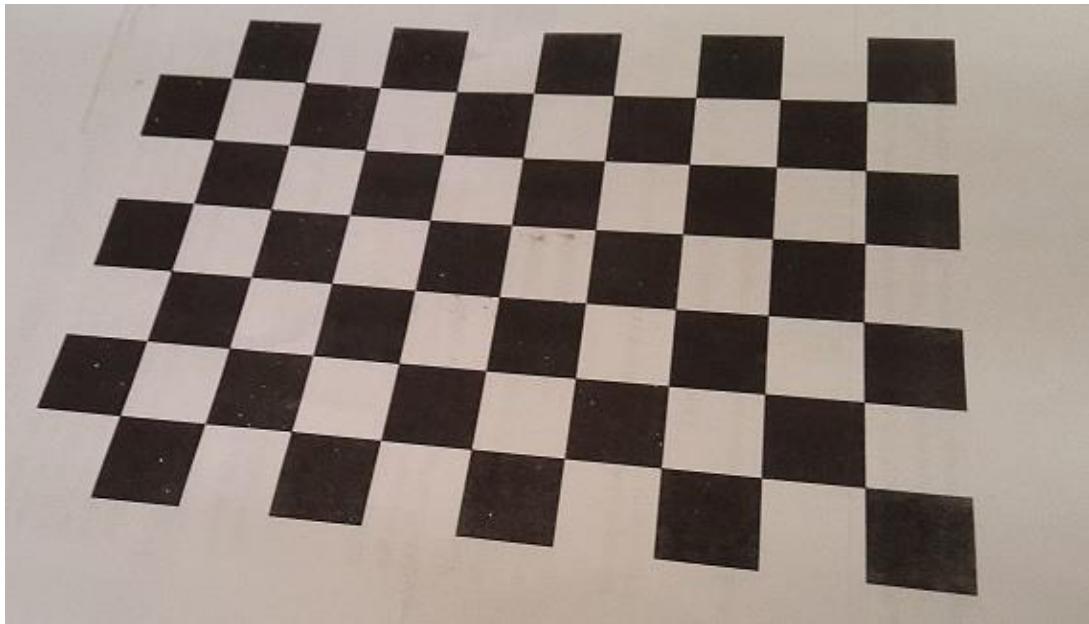


Fig.1.13. Immagine scacchiera utilizzata per la ricerca dei corners.

In sintesi l'algoritmo basa il suo funzionamento sull' analisi degli autovalori della matrice dei covarianti dell'immagine: ottenuti gli autovalori si imposta una funzione errore R che a seconda di come viene definita può assumere valori maggiori o minori di zero.

Nel caso specifico utilizzando la forma dell'errore $R = \lambda_1 \cdot \lambda_2 - 0.15 \cdot (\lambda_1 + \lambda_2)^2$ si ottiene la seguente distribuzione dei valori di R lungo la superficie dell'immagine:

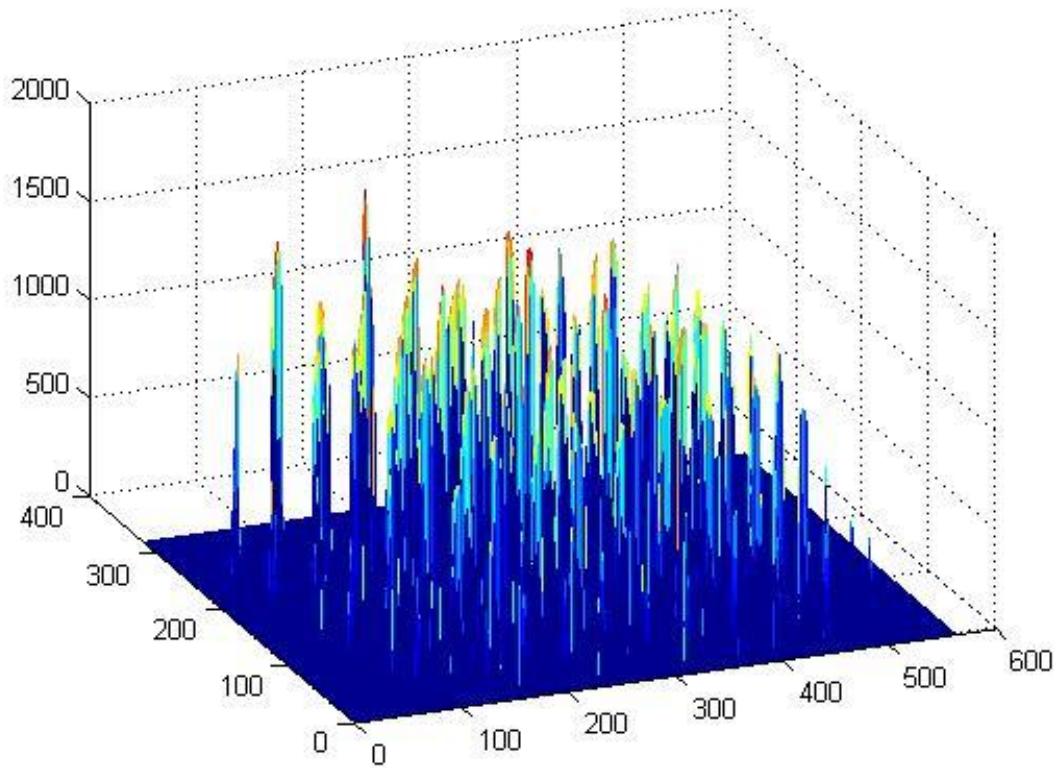


Fig.1.14.Rappresentazione 3D della distribuzione dell'errore R.

Il criterio che permette quindi di riconoscere il generico punto come effettivo corner è rappresentato dal massimo locale positivo del valore R.

Si riporta in seguente figura 1.15 la distribuzione dei valori positivi di R sull'immagine:

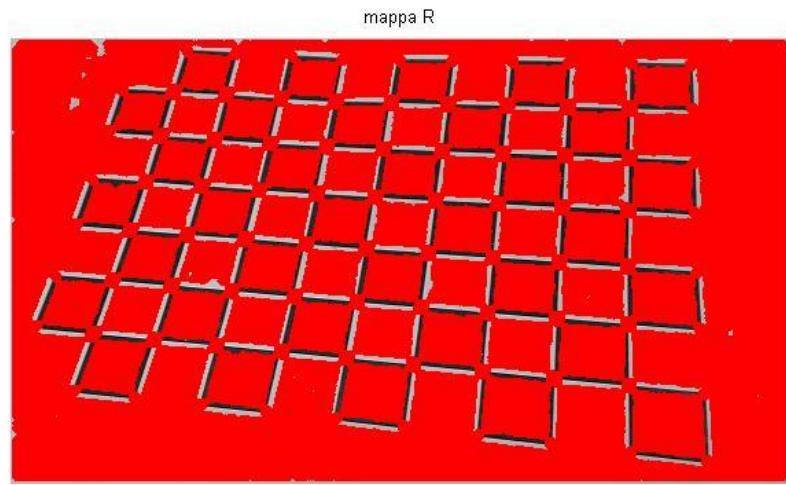


Fig.1.15. Distribuzione del valore $R > 0$ sulla superficie dell'immagine.

La ricerca dei massimi locali permette di ottenere il seguente risultato in figura 1.16:

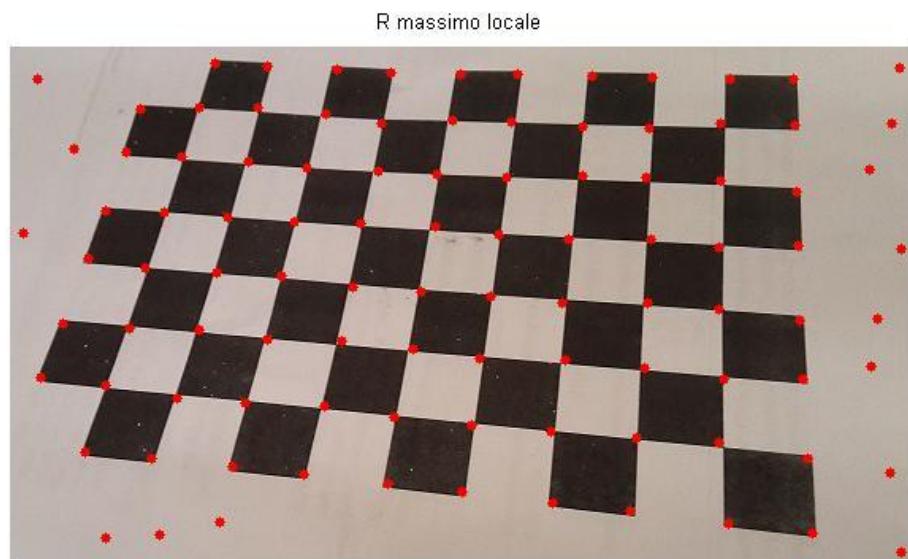


Fig.1.16. Massimi locali di R .

Avendo a disposizione diverse viste della scacchiera e trovati i punti di corner per ognuna di queste risultano in definitiva impostabili i sistemi di equazioni necessari alla calibrazione della fotocamera.

Bisogna tuttavia precisare che nonostante si disponga di un modello matematico che permette di localizzare i punti come riportato dall'analisi effettuata, l'effettiva accuratezza del calcolo dipende da una molteplicità di fattori.

In generale i software non implementano quasi mai l'applicazione diretta degli Harris corners nella forma originale, o comunque sono caratterizzati da un massiccio post e pre processing delle immagini e dei risultati al fine di renderli accettabili.

Nell'esempio dell'applicazione riportata infatti la determinazione dei punti risultanti in figura 1.16 richiede in primo luogo un thresholding della distribuzione di R riportata in figura 1.14, e successivamente la ricerca dei massimi locali.

L'effettiva determinazione di tutti i punti di corner dipende in questo caso quindi anche dalla grandezza della finestra di volta in volta utilizzata per il confronto dei valori di un dato punto con quelli circostanti in modo da determinare il massimo locale: l'utilizzo di una finestra di dimensioni molto ridotte pone il problema che vengono riconosciuti come corner anche punti che non lo sono, viceversa una finestra troppo grande causa l'eliminazione di punti effettivamente di corner nel caso due punti di corner rientrino nello stesso range di controllo.

Nell'esempio riportato si è utilizzata una finestra di dimensione 50x50 pixels e risulta chiaro che in generale la finestra da usare dipenderà principalmente dalle dimensioni dei singoli quadrati risultanti nelle immagini.

Tutto ciò poi senza considerare il fatto che anche il risultato in figura 1.16 richiede una pulizia ulteriore dei punti di falso corner, e successivamente si pone il problema di associare i punti omologhi nelle diverse viste.

In definitiva alla luce delle considerazioni effettuate si comprende chiaramente come gli algoritmi di *interest point detection* siano ancora in continua evoluzione così come in generale anche gli algoritmi di calibrazione delle fotocamere secondo modelli sempre più complessi.

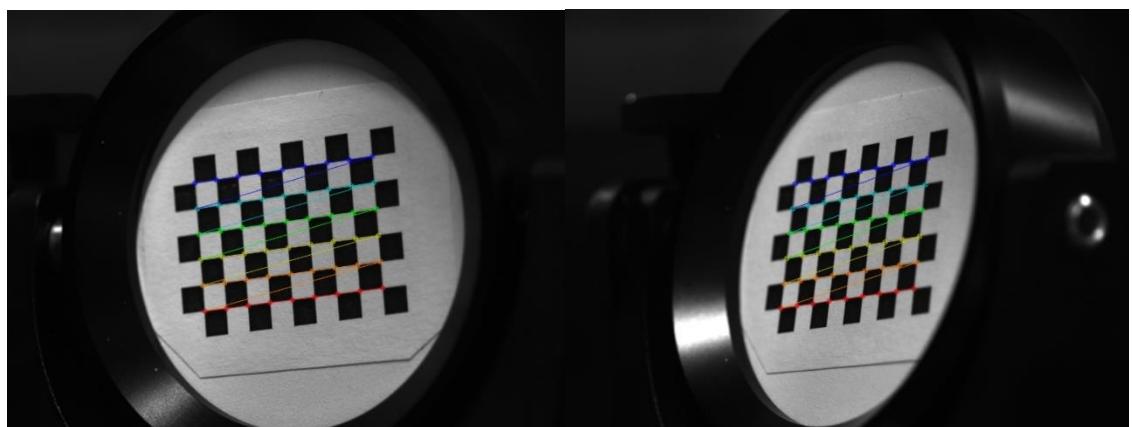
Per l'effettiva calibrazione della fotocamera risulta quindi opportuno fare riferimento ai software considerati standard al momento: è per questo motivo in particolare che si è scelto di fare riferimento alla libreria OpenCV versione 3.0, la quale rappresenta talvolta un vero e proprio termine di paragone per le nuove soluzioni studiate nell'ambito [29], [30].

1.2.5 Esempio di calibrazione

Si riporta in conclusione un esempio della calibrazione di fotocamera, che implementa il metodo proposto da Zhang.

Relativamente al software allegato in appendice si precisa che quest'ultimo è stato sviluppato in C++ utilizzando la libreria OpenCV: il software richiede in input una serie d'immagini riportanti il pattern a scacchiera, e la dimensione in millimetri del lato di uno dei quadrati in esso presenti.

Si riportano di seguito quattro delle venticinque immagini complessivamente utilizzate nel processo della calibrazione, i risultati ovviamente sono tanto migliori quanto è maggiore il numero di viste utilizzate:



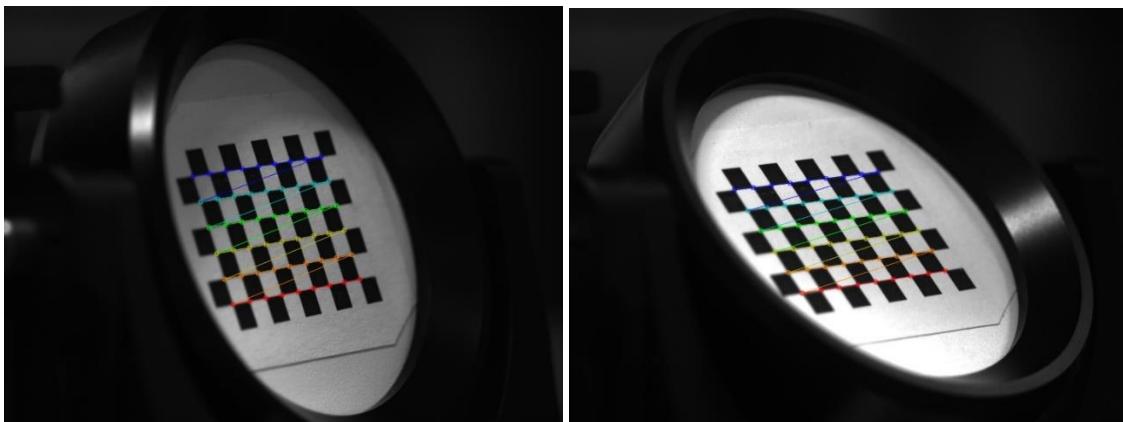


Fig.1.17. Immagini utilizzate nel processo della calibrazione.

Come si può notare in figura 1.17 il software permette di visualizzare i punti utilizzati nella calibrazione per ogni vista, inoltre gli ordina in maniera tale da poter associare i punti omologhi sull'oggetto nelle diverse viste.

Il software fornisce inoltre in output a seconda delle esigenze dell'utilizzatore la possibilità di poter visualizzare la matrice di rotazione per ogni vista, la matrice di calibrazione, i parametri di distorsione, l'errore di riproiezione e le immagini risultanti dalla correzione geometrica in caso fosse necessaria tale operazione.

Si riporta di seguito il valore calcolato della matrice di calibrazione [K], il vettore [D] contenente i valori dei cinque parametri di distorsione e l'errore di riproiezione dell'esempio considerato:

$$[K] = \begin{bmatrix} 7792,917 & 0 & 579.479 \\ 0 & 7852.667 & 479.255 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[D] = [-3.2386 \ 1476.94 \ -0.003470 \ -0.0001258 \ 2.8946]$$

$$\text{Errore riproiezione} = 0.3145$$

Infine si riporta l'esempio della correzione geometrica effettuata su un dato frame:

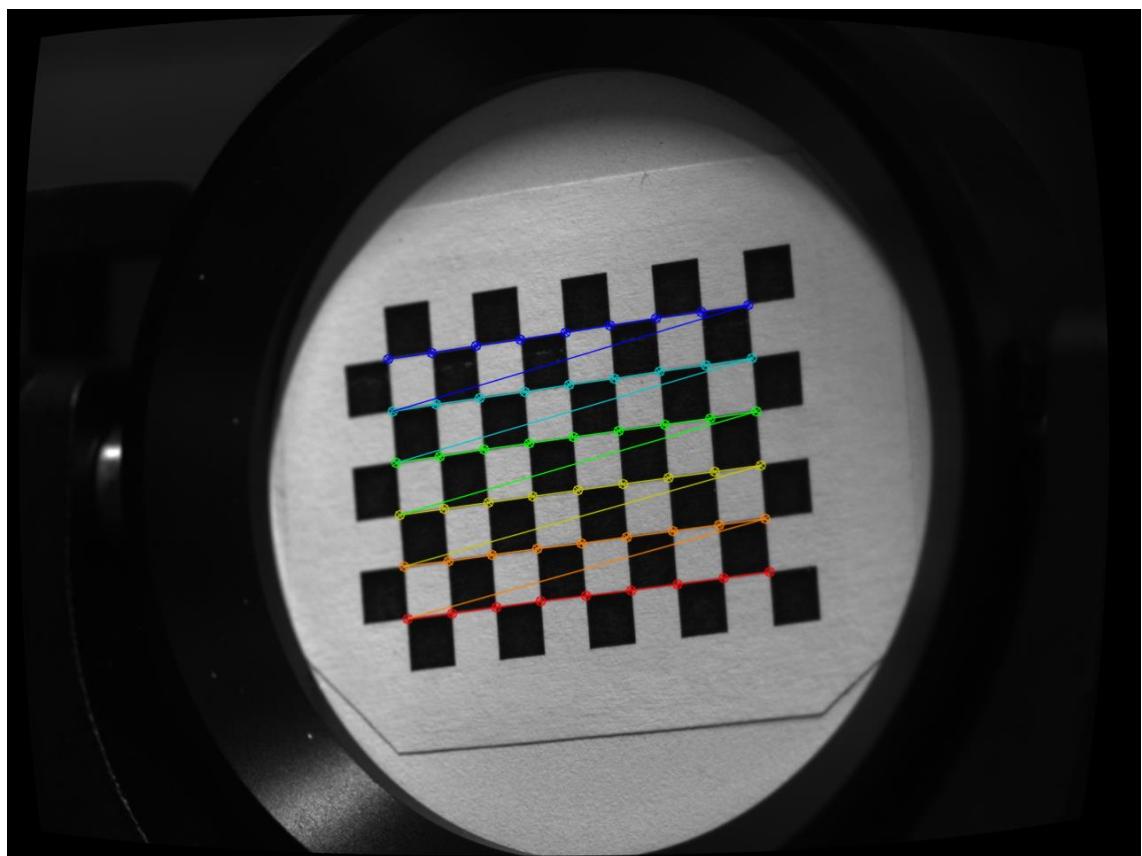


Fig.1.18. Immagini ottenuta a seguito della correzione geometrica: si presti attenzione ai lati dell'immagine.

1.3 Analisi del DIC3D secondo il modello di visione stereoscopico

Nota la trattazione del paragrafo 1.2 relativa alle metodologie per calcolare i parametri estrinseci ed intrinseci di un sistema composto da singola fotocamera, si affronta ora il problema della definizione di un modello che permetta di correlare le viste di un sistema costituito da più macchine fotografiche.

Relativamente alla trasformazione di proiezione centrale si ricorda infatti, che quest'ultima è caratterizzata dalla perdita di informazione sulla quota dei punti oggetto della trasformazione: tutti i punti sulla retta di proiezione vengono mappati su un unico punto nel piano dell'immagine, e da qui la necessità di disporre di più viste dell'oggetto esaminato al fine di poterne ricostruire la forma e studiarne il movimento.

Si ricorda a tal proposito (si veda paragrafo 1.1.5) che la possibilità di studiare il movimento di punti seguiti da un sistema di più fotocamere necessita oltre che della conoscenza delle distanze focali e di tutti i parametri intrinseci delle singole fotocamere, anche della distanza fra i rispettivi pinhole e della loro orientazione relativa.

Ulteriore difficolta rispetto al metodo presentato al paragrafo 1.1.5 è rappresentata dal fatto che in generale le due fotocamere possono avere qualsiasi orientazione e posizione relativa.

Risulta quindi chiaro che preliminarmente al calcolo degli spostamenti e della mappa di disparità gli algoritmi di correlazione digitale effettuano una serie di trasformazioni sulle immagini tale da riportarle al caso in cui fossero acquisite da un sistema in configurazione stereo normale.

In quest'ottica si può individuare una vera e propria pipeline di processi dei quali si compone la correlazione digitale d'immagine tridimensionale:

- 1) mediante gli algoritmi di calibrazione stereo si ottengono tutti i parametri relativi alle fotocamere come i loro parametri intrinseci, la loro distanza ed orientazione reciproca. La calibrazione stereo permette inoltre anche di determinare delle grandezze proprie della geometria epipolare le quali vengono sfruttate nelle fasi successive.

- 2) Effettuata la calibrazione stereo ed acquisite le immagini dell'oggetto studiato si procede con la fase di rettifica sulla base della geometria epipolare delle immagini, col fine di riportarsi alla configurazione d'acquisizione stereo normale: cioè orientazione relativa nulla fra le fotocamere (${}^{11}[R]_i=[I]$) e i loro pinhole alla stessa quota.
- 3) Successiva alla fase di rettifica risulta quella del calcolo dei valori di disparità per tutti i punti delle immagini che permette di identificare i punti di corrispondenza fra le diverse viste. Noti i valori di disparità risultano infatti anche costruibili le rette le cui intersezioni determinano la posizione spaziale dei punti considerati.
- 4) Si applica la correlazione digitale d'immagine bidimensionale per ottenere lo spostamento dei punti nel piano immagine di entrambe le fotocamere, noto il quale risultano ricavabili le nuove posizioni spaziali dei punti studiati e quindi anche il loro spostamento reale nelle tre direzioni ortogonali.

Si riporta perciò di seguito la trattazione relativa alle prime tre fasi appena citate, essendo la correlazione digitale d'immagine bidimensionale un argomento che per l'importanza ricoperta come metodo autonomo di caratterizzazione dei materiali, merita un capitolo a parte.

1.3.1 Matrice Fondamentale e calibrazione stereo

Considerando la figura 1.13 e richiamando i risultati del paragrafo 1.1.5 si ricorda come l'utilizzo di un sistema composto da anche solo due fotocamere permette di determinare la posizione di un punto P come intersezione fra le rispettive rette di proiezione sui due piani immagine.

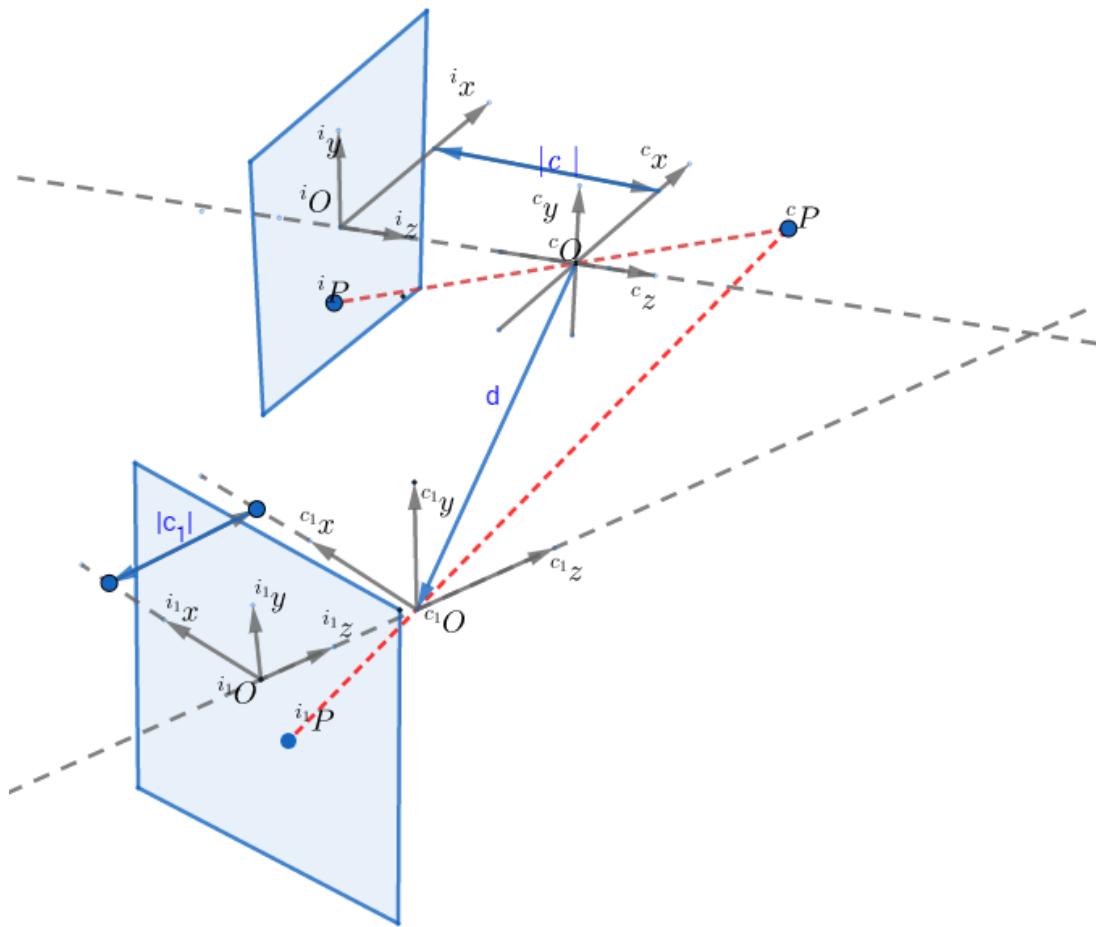


Fig. 1.13. Punto P visto da un sistema composto da due fotocamere.

Risulta in via preliminare opportuno precisare, che nonostante con un sistema a più viste come quello in figura 1.13, risulti determinata la posizione del generico punto P appartenente all'oggetto del quale si vuole ricostruire la forma, quest'ultima al termine dei calcoli potrà essere riprodotta soltanto a meno di un fattore di scala nel caso non si abbia alcuna informazione ulteriore sulla geometria reale dell'oggetto esaminato.

Quanto detto è facilmente intuibile in riferimento al fatto che indipendentemente dalle dimensioni fisiche reali dell'oggetto esaminato la sua immagine dipende direttamente dalla distanza di osservazione e dai parametri intrinseci del sistema di acquisizione. Si considerino per esempio due oggetti geometricamente simili con dimensioni molto diverse che possono presentare una stessa immagine in funzione semplicemente alla distanza con la quale si fotografano.

Relativamente alla figura seguente 1.14 si può notare ulteriormente un'importante caratteristica del sistema.

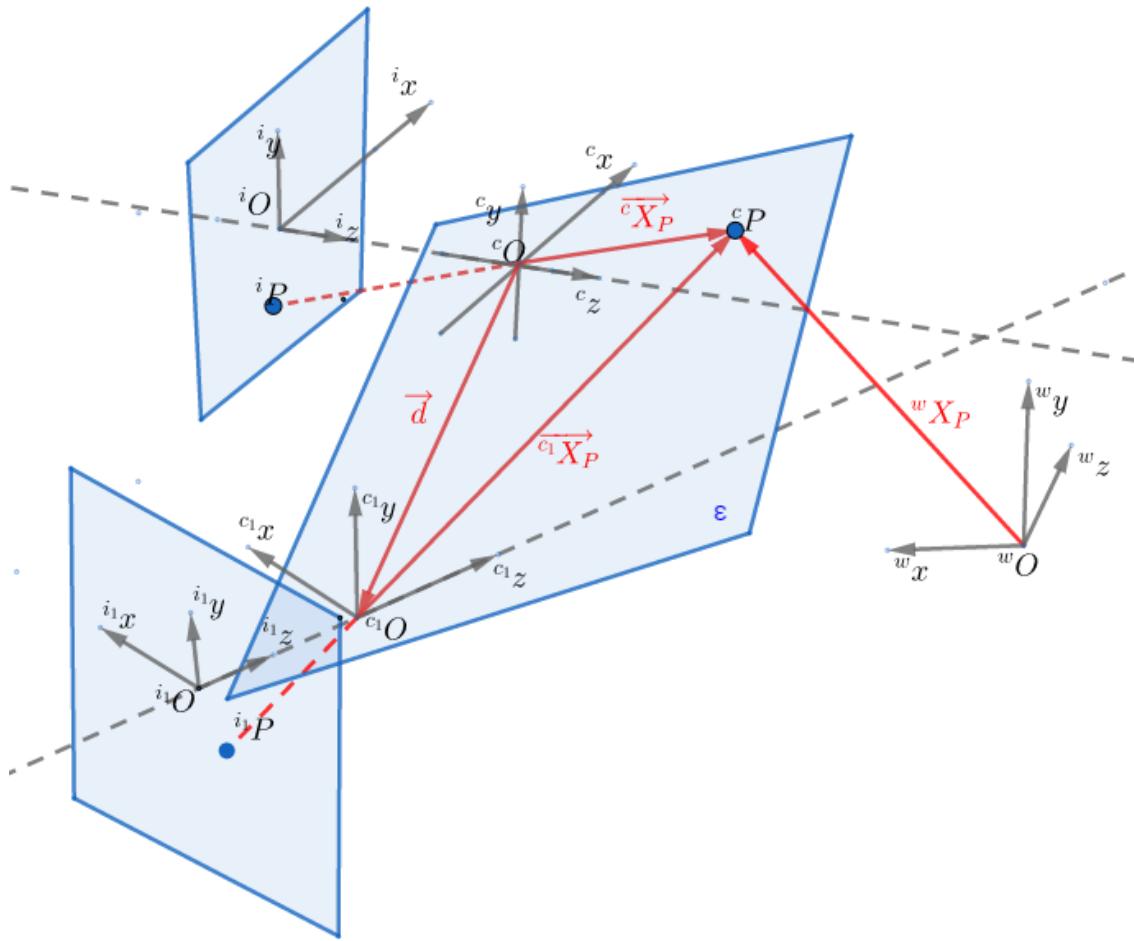


Fig. 1.14. Rappresentazione del piano epipolare ε e dei diversi sistemi di coordinate.

Risulta infatti che se il punto P è un punto di corrispondenza per le due immagini allora i vettori posizione \vec{d} del sistema di riferimento della seconda fotocamera rispetto alla prima, il vettore posizione di P nel sistema di riferimento della prima fotocamera $\overrightarrow{cX_P}$ e il vettore posizione di P nel sistema di riferimento della seconda fotocamera $\overrightarrow{c_1X_P}$ sono complanari in un piano ε .

Sempre in riferimento alla figura 1.14 si può quindi tradurre la condizione di complanarità facendo riferimento all'annullarsi del prodotto misto dei tre vettori \vec{d} , $\overrightarrow{cX_P}$, $\overrightarrow{c_1X_P}$ ed in particolare al loro prodotto triplo (che si ricorda rappresentare il volume del parallelepipedo costruito sui vettori considerati):

$$\left(\overrightarrow{cX_P} \times \vec{d} \right) \cdot \overrightarrow{c_1X_P} = \overrightarrow{cX_P} \cdot \left(\vec{d} \times \overrightarrow{c_1X_P} \right) = 0 \quad (1.3.1)$$

Affinchè si possa eseguire il calcolo bisogna però che i tre vettori siano definiti rispetto ad un sistema di riferimento comune a tutti.

Risulta inoltre più comodo fare riferimento alle posizioni del punto P sui piani immagine delle diverse fotocamere essendo proprio queste le uniche ottenibili direttamente dalle immagini.

Considerando perciò la relazione generale di proiezione vista nei capitoli precedenti si hanno rispettivamente per i due sistemi i , i_1 delle fotocamere le seguenti relazioni:

$${}^iX_P = [{}^iK_C] [{}^C R_w] \cdot ({}^w X_P - {}^w X_{c_o}) \quad (1.3.2)$$

$${}^{i_1}X_P = [{}^{i_1}K_{C_1}] [{}^{C_1}R_w] \cdot ({}^w X_P - {}^w X_{c_{i_1}o}) \quad (1.3.3)$$

attraverso le quali risulta possibile ottenere i vettori normalizzati della direzione di P rispetto ai due sistemi di riferimento come riportato di seguito:

$$[{}^C R_w]^{-1} [{}^i K_C]^{-1} \cdot {}^i X_P = ({}^w X_P - {}^w X_{c_o}) \quad (1.3.3)$$

$$[{}^{C_1} R_w]^{-1} [{}^{i_1} K_{C_1}]^{-1} \cdot {}^{i_1} X_P = ({}^w X_P - {}^w X_{c_{i_1}o}) \quad (1.3.4)$$

Per quanto riguarda invece il vettore posizione del sistema di riferimento i_1 rispetto ad i , risulta che esso vale semplicemente:

$$\overrightarrow{w d} = ({}^w X_{c_{i_1}o} - {}^w X_{c_o}) \quad (1.3.5)$$

Dalle relazioni appena ottenute risulta perciò possibile riscrivere la condizione di complanarità nella seguente forma:

$$({}^w X_p - {}^w X_{c_o}) \cdot \left[\left({}^w X_{c_{i_o}} - {}^w X_{c_o} \right) \times \left({}^w X_p - {}^w X_{c_{i_o}} \right) \right] = 0 \quad (1.3.6)$$

Sostituendo ora le espressioni trovate precedentemente per le diverse quantità nell'ultima espressione si ottiene:

$$\{{}^i X_p\}^t \left([{}^i K_c]^{-1} \right)^t \left([{}^c R_w]^{-1} \right)^t \cdot \left[\overrightarrow{d} \times \left([{}^c R_w]^{-1} [{}^i K_{c_i}]^{-1} \cdot \{{}^i X_p\} \right) \right] = 0 \quad (1.3.7)$$

La (1.3.7) utilizzando la matrice antisimmetrica [D] per esprimere il prodotto vettoriale come:

$$\overrightarrow{d} \times \left([{}^c R_w]^{-1} [{}^i K_{c_i}]^{-1} \cdot \{{}^i X_p\} \right) = [D] \cdot [{}^c R_w]^{-1} [{}^i K_{c_i}]^{-1} \cdot \{{}^i X_p\} \quad (1.3.8)$$

con [D] pari ad:

$$[D] = \begin{bmatrix} 0 & -d_3 & d_2 \\ d_3 & 0 & -d_1 \\ -d_2 & d_1 & 0 \end{bmatrix} \quad (1.3.9)$$

può essere riscritta come:

$$\{{}^i X_p\}^t \left([{}^i K_c]^{-1} \right)^t \left([{}^c R_w]^{-1} \right)^t \cdot [D] \cdot [{}^c R_w]^{-1} [{}^i K_{c_i}]^{-1} \{{}^i X_p\} = 0 \quad (1.3.10)$$

In virtù dell'ortogonalità delle matrici di rotazione si può anche riscrivere la (1.3.10) in maniera più semplificata come:

$$\{{}^i X_p\}^t \left([{}^i K_c]^{-1} \right)^t [{}^c R_w] \cdot [D] \cdot [{}^c R_w]^t [{}^i K_{c_i}]^{-1} \{{}^i X_p\} = 0 \quad (1.3.11)$$

Ora in considerazione al fatto che il risultato dei prodotti matriciali nella (1.3.11) equivale ad una matrice 3x3, si indica tale prodotto con la matrice [F] come riportato di seguito:

$$[F] = \left([{}^i K_c]^{-1} \right)^t \left([{}^c R_w]^{-1} \right)^t \cdot [D] \cdot [{}^c R_w]^{-1} [{}^i K_{c_i}]^{-1} \quad (1.3.12)$$

Questa ulteriore semplificazione formale permette di riscrivere in maniera molto più compatta la condizione di complanarità come segue:

$$\{{}^i X_p\}^t \cdot [F] \cdot \{{}^i X_p\} = 0 \quad (1.3.13)$$

Riguardo all'equazione trovata si può capire come questa rappresenti un risultato molto importante, in quanto permette dati due punti di corrispondenza per diverse viste dello stesso oggetto di esprimere il loro legame mediante una singola matrice 3×3 funzione di tutti i parametri intriseci ed estrinseci dei due sistemi di acquisizione nonché dell'informazione sulla relativa posizione di questi.

La matrice $[F]$ è nota infatti come matrice Fondamentale ed esprime l'orientazione relativa di due immagini per fotocamere non calibrate, rappresenta perciò uno dei risultati più importanti della *computer vision* negli ultimi decenni.

Si noti inoltre che la relazione (1.3.13) vale per tutti i punti delle due immagini essendo la matrice fondamentale dipendente soltanto dai parametri di calibrazione e della relativa orientazione delle viste considerate e non dal particolare pixel considerato.

In conclusione risulta chiaro come la determinazione della matrice fondamentale e la seguente scomposizione di quest'ultima, secondo metodologie simili a quelle viste nella calibrazione della singola fotocamera, permette di ottenere tutti i parametri caratteristici del sistema delle due fotocamere.

La calibrazione stereo infatti si sostanzializza nella determinazione della matrice fondamentale e nella successiva estrapolazione da quest'ultima di tutte le informazioni relative ai parametri intriseci delle fotocamere, alla loro reciproca posizione ed orientazione.

A tal proposito si rimanda alla bibliografia [8],[9] relativa all'algoritmo ad 8 punti per la risoluzione della $[F]$ il quale è ritenuto il metodo standard utilizzato nella calibrazione stereo.

1.3.2 Geometria Epipolare

Al fine di esporre in maniera chiara l'insieme dei processi seguenti alla fase della calibrazione stereo risulta ora fondamentale introdurre la nomenclatura propria delle quantità che verranno utilizzate nelle seguenti trattazioni.

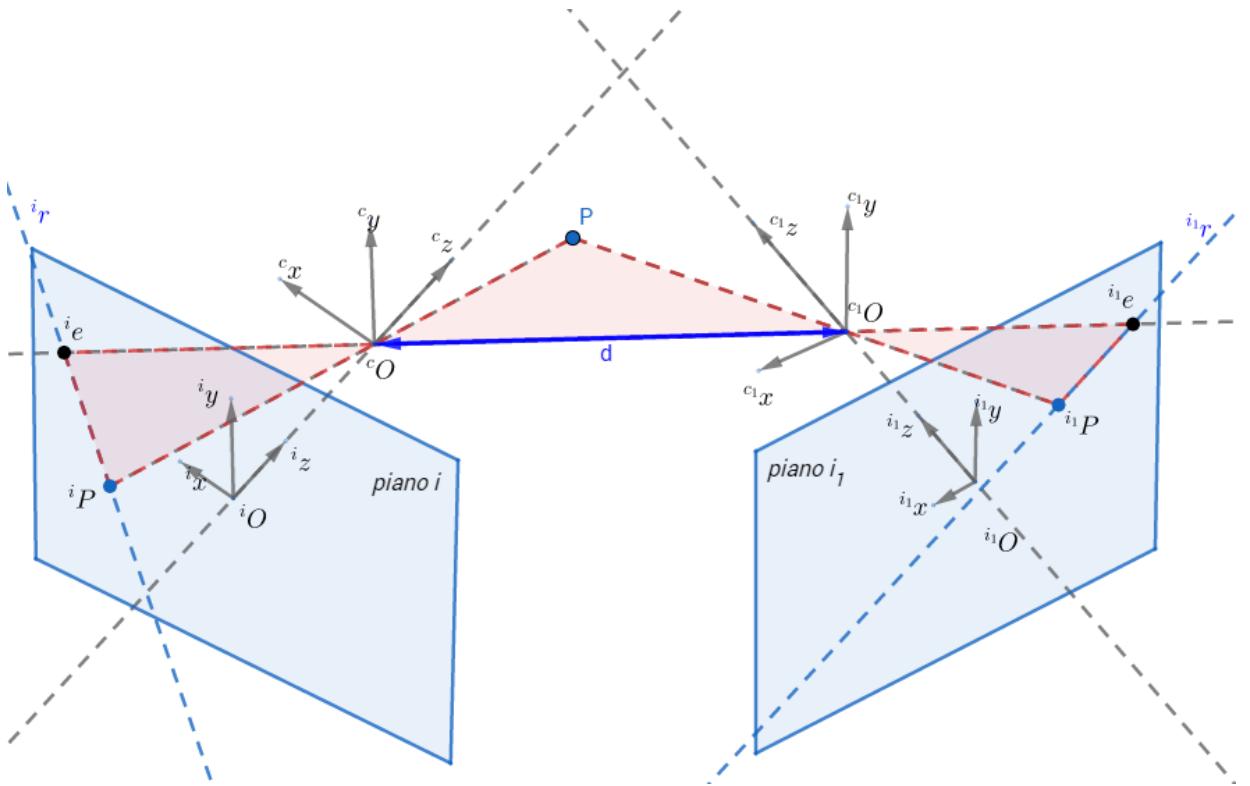


Fig. 1.15. Rappresentazione della proiezione del punto *P* sui piani immagine di due fotocamere.

In riferimento alla figura 1.15, riportante un sistema di due fotocamere genericamente orientate l'una rispetto all'altra e posizionate ad una distanza relativa pari a *d*, si indicano di seguito i diversi sistemi di riferimento che si possono osservare:

- ${}^c\{\text{OXYZ}\}$: sistema di riferimento centrato sul centro di proiezione (*pinhole*) della fotocamera sinistra.
- ${}^{c_1}\{\text{OXYZ}\}$: sistema di riferimento centrato sul centro di proiezione della fotocamera destra.
- ${}^i\{\text{OXYZ}\}$: sistema di riferimento del piano immagine (*i*) della fotocamera sinistra avente asse ottico (asse i_z) coincidente con quello di ${}^c\{\text{OXYZ}\}$.

- ${}^{i1}\{OXYZ\}$: sistema di riferimento del piano immagine (i_1) della fotocamera destra avente asse ottico (asse ${}^{i1}z$) coincidente con quello di ${}^{c1}\{OXYZ\}$.

Come si può osservare considerando la retta passante per i centri di proiezione delle due fotocamere, si ha che questa interseca i due piani immagine rispettivamente ai punti ${}^i e$ ed ${}^{iI} e$, questi punti d'intersezione prendono il nome di epipoli.

Si consideri ora un punto P tale che rientri nella *field of view* di entrambe: come visto precedentemente la trasformazione di proiezione centrale mappa il punto P sui rispettivi piani immagine delle due fotocamere.

Si indica con iP la proiezione del punto P sul piano immagine della fotocamera sinistra, mentre con ${}^{iI}P$ la proiezione di P sul piano immagine della fotocamera destra.

Come visto precedentemente se iP ed ${}^{iI}P$ sono punti di corrispondenza nelle due immagini, e cioè, come supposto nel nostro caso, sono la proiezione di un medesimo punto P sui due piani immagine, allora i vettori costruiti sulle direzioni di proiezione di P su iP e P su ${}^{iI}P$ appartengono ad uno stesso piano, il quale prende nome di piano epipolare.

Si noti come qualsiasi sia il punto P interno alla *field of view* delle fotocamere il piano epipolare contiene anche il vettore posizione relativa di ${}^{c1}O$ rispetto ad cO o viceversa, essendo appunto questi i centri di proiezione delle due fotocamere.

Richiamando alla memoria i risultati visti per la trasformazione di proiezione centrale, si era concluso come questa fosse caratterizzata da perdita di informazione nel processo di acquisizione dell'immagine.

Se per l'acquisizione con singola fotocamera tutti i punti su una generica retta di proiezione venivano mappati su un unico punto sul piano immagine, si può invece considerare ora sia la proiezione dei punti che si trovano sulla retta di proiezione di P su ${}^{iI}P$ nel piano immagine della fotocamera sinistra (piano i), che la proiezione dei punti sulla retta di proiezione di P su iP sul piano immagine della fotocamera destra (piano i_1).

Come è facile osservare, essendo la proiezione centrale una trasformazione *straight line preserving*, i punti proiezione delle rette considerate risultano collineari su entrambi i piani immagine.

In particolare si distinguono rispettivamente per i due piani immagine, le rette ${}^i\mathbf{r}$ ed ${}^{iI}\mathbf{r}$ costruite sui punti considerati, queste rette prendono il nome di rette epipolari, e come ovvio concludere appartengono anche queste al piano epipolare.

Altra caratteristica risulta dal fatto che tutte le rette epipolari passano per il punto epipolare.

Introdotta perciò la nomenclatura opportuna alla trattazione, si procede al calcolo delle diverse quantità discusse.

Risulta che il vettore distanza relativa fra i due sistemi di riferimento delle fotocamere è pari a:

$$\overrightarrow{\mathbf{d}} = \left({}^w\mathbf{X}_{c_1} - {}^w\mathbf{X}_{c_0} \right) \quad (1.3.14)$$

Per quanto riguarda le rette epipolari, si possono scrivere per queste le loro espressioni sui due piani:

$$\begin{aligned} {}^i\mathbf{r} &= {}^i\mathbf{a} \cdot {}^i\mathbf{x} + {}^i\mathbf{b} \cdot {}^i\mathbf{y} + {}^i\mathbf{c} \\ {}^{iI}\mathbf{r} &= {}^{iI}\mathbf{a} \cdot {}^{iI}\mathbf{x} + {}^{iI}\mathbf{b} \cdot {}^{iI}\mathbf{y} + {}^{iI}\mathbf{c} \end{aligned} \quad (1.3.15)$$

Indicando ora i seguenti vettori contenenti gli argomenti delle variabili delle rette:

$$\begin{aligned} \{{}^i\mathbf{s}\}^t &= [{}^i\mathbf{a}, {}^i\mathbf{b}, {}^i\mathbf{c}] \\ \{{}^{iI}\mathbf{s}\}^t &= [{}^{iI}\mathbf{a}, {}^{iI}\mathbf{b}, {}^{iI}\mathbf{c}] \end{aligned} \quad (1.3.16)$$

Si può esprimere la relazione di appartenenza del generico punto P alla linea rispettiva retta epipolare nella seguente maniera:

$$\begin{aligned} \{{}^i\mathbf{X}_P\}^t \cdot {}^i\mathbf{s} &= 0 \\ \{{}^{iI}\mathbf{X}_P\}^t \cdot {}^{iI}\mathbf{s} &= 0 \end{aligned} \quad (1.3.17)$$

con:

$$\begin{aligned}\left\{{}^i X_p\right\}^t &= \left[{}^i x_p, {}^i y_p, 1 \right] \\ \left\{{}^i X_p\right\}^t &= \left[{}^{i_1} x_p, {}^{i_1} y_p, 1 \right]\end{aligned}\quad (1.3.18)$$

Si confrontino ora le espressioni ottenute con la condizione di complanarità espressa attraverso la matrice fondamentale $[F]$ (equazione (1.3.13)), e la prima espressione in (1.3.17):

$$\begin{aligned}\left\{{}^i X_p\right\}^t \cdot {}^i s &= 0 \\ \left\{{}^i X_p\right\}^t \cdot [F] \cdot \left\{{}^{i_1} X_p\right\} &= 0\end{aligned}$$

Si può concludere che:

$${}^i s = [F] \cdot \left\{{}^{i_1} X_p\right\} \quad (1.3.19)$$

La relazione appena trovata permette di ottenere la retta epipolare nel piano immagine della fotocamera sinistra.

Confrontando invece la seconda relazione in (1.3.17) con l'equazione della complanarità si ottiene:

$$\left\{{}^{i_1} s\right\}^t = \left\{{}^i X_p\right\}^t \cdot [F] \quad (1.3.20)$$

che permette di ricavare la retta epipolare nel sistema di riferimento del piano immagine della fotocamera destra.

Per quanto concerne gli epipoli si noti che essi sono di immediato calcolo in caso siano note le matrici di proiezione ${}^i[P]_w$ e ${}^{i_1}[P]_w$ delle due fotocamere e le posizioni dei loro pinhole, si hanno infatti le seguenti relazioni:

$$\begin{aligned}{}^i e &= {}^i[P]_w \cdot {}^w X_{c_o} \\ {}^{i_1} e &= {}^{i_1}[P]_w \cdot {}^w X_{c_{i_o}}\end{aligned}\quad (1.3.21)$$

In alternativa osservando che le rette epipolari per ogni immagine formano un fascio centrato nell' epipolo risulta verificata la condizione di appartenenza di quest'ultimo alla generica retta epipolare espressa come:

$$\begin{aligned} \left\{ {}^i e \right\}^t \cdot {}^i s &= 0 \\ \left\{ {}^{i_1} s \right\}^t \cdot {}^{i_1} e &= 0 \end{aligned} \quad (1.3.22)$$

Sostituendo ora le espressioni precedentemente trovate per ${}^i s$ ed ${}^{i_1} s$, equazioni (1.3.19) e (1.3.20) si ottiene:

$$\begin{aligned} \left\{ {}^i e \right\}^t \cdot [F] \cdot \left\{ {}^i X_p \right\} &= 0 \\ \left\{ {}^i X_p \right\}^t \cdot [F] \cdot {}^i e &= 0 \end{aligned} \quad (1.3.23)$$

Le quali risultano verificate solo nel caso:

$$\begin{aligned} [F]^t \cdot {}^i e &= 0 \\ [F] \cdot {}^i e &= 0 \end{aligned} \quad (1.3.24)$$

essendo i vettori $\{ {}^i X_p \}$ ed $\{ {}^{i_1} X_p \}$ vettori posizione del punto P sul piano immagine e quindi mai nulli.

Si conclude in definitiva che i due epipoli equivalgono allo spazio nullo della matrice fondamentale e della sua trasposta, in particolare:

$$\begin{aligned} {}^i e &= \text{null}([F]^t) \\ {}^i e &= \text{null}([F]) \end{aligned} \quad (1.3.25)$$

1.3.3 Rettificazione delle immagini

Una volta calcolata la matrice fondamentale e noti per il sistema di acquisizione delle immagini gli epipoli e le rette epipolari risulta possibile il processo di rettificazione delle immagini il quale permetterà successivamente di ottenere la mappa di disparità nella fase di stereo matching.

Relativamente al metodo presentato si fa riferimento alla [11] (Fusiello, A., Trucco, E. Verri).

In riferimento alla figura 1.15 si può verificare che considerando un perfetto allineamento fra asse ottico del piano immagine e l'asse z del sistema di riferimento della fotocamera, si ha che il vettore posizione dell'epipolo espresso in quest'ultimo sistema di riferimento (${}^C\{OXY\}$) risulta caratterizzato dalla medesima direzione del vettore distanza d relativa fra i centri di proiezione delle due fotocamere.

Tale fatto può essere sfruttato per costruire una matrice i cui elementi siano una base ortonormale che esprima la rotazione da applicare al piano immagine tale da rettificarla.

La matrice $[R]$ ricercata sarà della seguente forma:

$$[R] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} r_1^t \\ r_2^t \\ r_3^t \end{bmatrix} \quad (1.3.26)$$

Con i vettori r_1 , r_2 ed r_3 che verranno calcolati nella seguente trattazione, in riferimento indifferentemente ad uno dei due piani immagine essendo il procedimento di rettificazione lo stesso per entrambi.

Il primo passo consiste nel scegliere il vettore r_1 di norma unitaria costruito sulla retta congiungente i centri di proiezione delle due fotocamere, come facile intuire questo sarà pari quindi al vettore normalizzato della distanza relativa fra i centri di proiezione.

In definitiva si ha che il vettore distanza espresso nel sistema di riferimento della fotocamera vale:

$$\{^c d\}^t = [^c d_x, ^c d_y, ^c d_z] \quad (1.3.27)$$

Si sceglie perciò il vettore r_1 come:

$$\{^c r_1\}^t = \left[\frac{^c d_x}{\sqrt{^c d_x^2 + ^c d_y^2 + ^c d_z^2}}, \frac{^c d_y}{\sqrt{^c d_x^2 + ^c d_y^2 + ^c d_z^2}}, \frac{^c d_z}{\sqrt{^c d_x^2 + ^c d_y^2 + ^c d_z^2}} \right] \quad (1.3.28)$$

Si costruisce ora il secondo vettore r_2 normalizzato come prodotto vettoriale di r_1 e dell'asse ottico della fotocamera:

$${}^c r_2 = \frac{{}^c r_1 \times {}^c \{\text{opt.axes}\}}{\|{}^c r_1 \times {}^c \{\text{opt.axes}\}\|} = \frac{{}^c r_1 \times [0, 0, 1]}{\|{}^c r_1 \times [0, 0, 1]\|} \quad (1.3.29)$$

Infine il terzo vettore r_3 è pari al prodotto vettoriale dei precedenti due:

$${}^c r_3 = {}^c r_1 \times {}^c r_2 \quad (1.3.30)$$

Costruita perciò la matrice $[R]$ si può procedere col calcolo della nuova posizione dei punti per l'immagine rettificata.

Risulta che ogni punto sull'immagine originale è definito rispetto al centro di proiezione come:

$${}^c X_i^t = [{}^c x_i, {}^c y_i, -c] \quad (1.3.31)$$

Per cui risulta che la nuova posizione nell'immagine rettificata è pari ad:

$${}^c X_{i-\text{rect}} = [R] \cdot {}^c X_i = \begin{bmatrix} r_{11} \cdot {}^c x_i + r_{12} \cdot {}^c y_i - r_{13} \cdot c \\ r_{21} \cdot {}^c x_i + r_{22} \cdot {}^c y_i - r_{23} \cdot c \\ r_{31} \cdot {}^c x_i + r_{32} \cdot {}^c y_i - r_{33} \cdot c \end{bmatrix} \quad (1.3.31)$$

Dalla quale risulta calcolabile la posizione espressa sul piano immagine come:

$${}^iX_{i\text{-rect}} = \begin{bmatrix} r_{11} \cdot {}^c x_i + r_{12} \cdot {}^c y_i - r_{13} \cdot c \\ r_{31} \cdot {}^c x_i + r_{32} \cdot {}^c y_i - r_{33} \cdot c \\ r_{21} \cdot {}^c x_i + r_{22} \cdot {}^c y_i - r_{23} \cdot c \\ r_{31} \cdot {}^c x_i + r_{32} \cdot {}^c y_i - r_{33} \cdot c \\ 1 \end{bmatrix} \quad (1.3.32)$$

Il calcolo appena presentato, effettuato separatamente per le due immagini permette di rettificarle.

Per ogni ulteriore approfondimento si rimanda alla bibliografia in [11].

1.3.4 Template matching per il calcolo della mappa di disparità

La conoscenza delle coordinate spaziali dell'oggetto del quale si vuole ricostruire la forma, o studiare la deformazione presuppone la conoscenza del valore della disparità per ogni punto delle due immagini.

In generale si nota come la conoscenza delle rette epipolari semplifichi di una dimensione il problema della ricerca di regioni uguali fra le diverse immagini da problema bidimensionale a monodimensionale: per ogni regione della prima immagine basterà ricercare quella corrispondente nella seconda lungo la rispettiva retta epipolare.

Come si può notare ulteriormente poi l'operazione di stereo-rettifica effettuata dopo la calibrazione del sistema delle due fotocamere è tale da rendere le rette epipolari parallele e collineari per le due immagini, fatto che semplifica ulteriormente il problema della ricerca dei valori di disparità, essendo le immagini stereo-rettificate caratterizzate solamente da una traslazione orizzontale relativa.

Quanto detto permette di concludere come la ricerca della mappa di disparità si configuri come un problema di *template matching*, per il quale esistono diversi algoritmi di risoluzione.

Relativamente alla figura 1.16 riportata di seguito si può esporre il problema del template matching nei seguenti termini: data una immagine 1I ed un template 2I si determini la posizione in termini di u, v della regione su 1I che risulti di massima similarità con 2I .

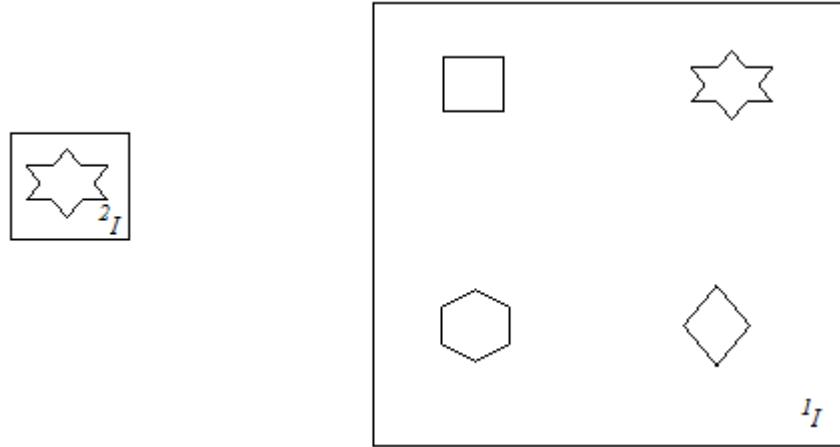


Fig.1.16. Problema del template matching con 2I rappresentante il template ed 1I l'immagine.

Esistono come detto diversi criteri alla soluzione del problema, come riportato di seguito:

- *Sum of squares differences:*

$$\text{SSD} = \sum_{j=1}^n \sum_{k=1}^m ({}^1I(j-u, k-v) - {}^2I(j, k))^2 \quad (1.3.33)$$

- *Sum of absolute differences:*

$$\text{SAD} = \sum_{j=1}^n \sum_{k=1}^m \| {}^1I(j-u, k-v) - {}^2I(j, k) \| \quad (1.3.34)$$

- *Cross correlation:*

$$\text{CC} = \sum_{j=1}^n \sum_{k=1}^m ({}^1I(j-u, k-v) * {}^2I(j, k)) \quad (1.3.35)$$

e tutti caratterizzati dal risolvere il problema della determinazione del vettore spostamento minimizzando la somma delle differenze (SSD, SAD) oppure massimizzando il prodotto delle intensità (*Cross correlation*) delle due subimmagini.

Bisogna specificare tuttavia che nonostante i criteri citati offrano una maniera relativamente semplice e intuitiva per la risoluzione del problema non sono immuni a problemi legati alla variazione di contrasto e luminosità (dovuti a variazione del tempo di esposizione o del livello di luce ambientale) fra le diverse immagini, motivo per il quale si usano criteri normalizzati che riducono drasticamente il problema.

Si riporta di seguito per esempio il criterio ZNCC (*Zero Mean Normalized Cross Correlation*), definito come il rapporto fra la covarianza dell'immagine e del template e il prodotto delle rispettive deviazioni standard calcolate per diversi valori del vettore di traslazione u, v :

$$ZNCC = \frac{\sigma_{I^2_I}(u, v)}{\sigma_{I_I}(u, v)\sigma_{I^2_I}} \quad (1.3.36)$$

Con riferimento ad un template di dimensioni $n \times m$ pixels e numero totale di pixel indicato con N si hanno le seguenti espressioni per le diverse quantità indicate nella relazione precedente:

$$\sigma_{I_I}(u, v) = \sqrt{\frac{1}{N-1} \cdot \left[\sum_{j=1}^n \sum_{k=1}^m I^2(j-u, k-v) - \frac{1}{N} \left(\sum_{j=1}^n \sum_{k=1}^m I(j-u, k-v) \right)^2 \right]} \quad (1.3.37)$$

$$\sigma_{I^2_I} = \sqrt{\frac{1}{N-1} \cdot \left[\sum_{j=1}^n \sum_{k=1}^m I^2(j, k) - \frac{1}{N} \left(\sum_{j=1}^n \sum_{k=1}^m I(j, k) \right)^2 \right]} \quad (1.3.38)$$

$$\sigma_{I^2_I}(u, v) = \sqrt{\frac{1}{N-1} \cdot \left[\sum_{j=1}^n \sum_{k=1}^m I(j-u, k-v) \cdot I(j, k) - \frac{1}{N} \left(\sum_{j=1}^n \sum_{k=1}^m I(j-u, k-v) \cdot I(j, k) \right) \right]} \quad (1.3.39)$$

Come si può notare il processo è di tipo iterativo nelle variabili u e v e i valori della ZNCC sono compresi nell' intervallo $[0, 1]$. La soluzione del problema fra tutti i valori trovati attraverso il criterio sarà tale da massimizzare la quantità riportata in (1.3.36) e cioè:

$$X = \underset{u, v}{\operatorname{argmax}} \frac{\sigma_{I^2_I}(u, v)}{\sigma_{I_I}(u, v)\sigma_{I^2_I}} \quad (1.3.40)$$

la soluzione trovata tuttavia risulta avere l'accuratezza al pixel intero essendo le immagini definite in queste unità.

Si può quindi estendere la soluzione ad una risoluzione subpixel mediante opportune operazioni matematiche come riportato di seguito:

Si può pensare anzitutto di operare il fitting di una forma quadratica del tipo $f(x,y)=ax^2+2bxy+cy^2+q$ nell'intorno della soluzione trovata, tale da passare da una serie discreta di punti ad una superficie continua, per poi trovare il massimo locale della superficie che sarà appunto la posizione esatta che risolve il problema del template matching.

Nel nostro caso in forma matriciale si ha che indicato il vettore $x_{u,v}$ soluzione trovata col ZCNN e il vettore x_{\max} posizione del massimo locale :

$$[x_{u,v}] = \begin{bmatrix} u_{zncc} \\ v_{zncc} \end{bmatrix} \quad (1.3.41)$$

$$[x_{\max}] = \begin{bmatrix} u_{\max} \\ v_{\max} \end{bmatrix} \quad (1.3.42)$$

Si può riscrivere la forma quadratica di fitting nel seguente modo:

$$f(u,v) = \begin{bmatrix} u_{zncc} \\ v_{zncc} \end{bmatrix} - \begin{bmatrix} u_{\max} \\ v_{\max} \end{bmatrix}^t \cdot \begin{bmatrix} a & b \\ b & c \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} u_{\max} \\ v_{\max} \end{bmatrix} + q \quad (1.3.43)$$

Oppure equivalentemente:

$$f(x) = (x_{u,v} - x_{\max})^t \cdot [A] \cdot (x_{u,v} - x_{\max}) + q \quad (1.3.44)$$

Risulta ora che il Jacobiano della funzione $f(x)$ (e cioè il vettore delle sue derivate) vale:

$$\nabla f(x_{u,v}) = 2 \cdot [A] \cdot (x_{u,v} - x_{\max}) \quad (1.3.45)$$

Mentre l'Hessiana (derivata seconda di $f(x)$) vale:

$$H_{\text{hess}}(x_{u,v}) = 2 \cdot [A] \quad (1.3.46)$$

Dalla quale si ottiene la seguente espressione che lega il massimo e la soluzione trovata inizialmente:

$$\nabla f(x_{u,v}) = H_{\text{hess}}(x_{u,v}) \cdot (x_{u,v} - x_{\max}) \quad (1.3.47)$$

In definitiva risulta calcolabile il punto di massimo:

$$x_{\max} = x_{u,v} - H_{\text{hess}}^{-1}(x_{u,v}) \cdot \nabla f(x_{u,v}) \quad (1.3.48)$$

Riscrivendo la (1.3.48) in forma matriciale espansa al fine di chiarire ulteriormente il calcolo si ottiene:

$$\begin{bmatrix} u_{\max} \\ v_{\max} \end{bmatrix} = \begin{bmatrix} u_{\text{zncc}} \\ v_{\text{zncc}} \end{bmatrix} - \begin{bmatrix} \frac{\partial^2 f(u,v)}{\partial u^2} & \frac{\partial^2 f(u,v)}{\partial u \partial v} \\ \frac{\partial^2 f(u,v)}{\partial u \partial v} & \frac{\partial^2 f(u,v)}{\partial v^2} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \frac{\partial f(u,v)}{\partial u} \\ \frac{\partial f(u,v)}{\partial v} \end{bmatrix}_{\begin{bmatrix} u_{\text{zncc}} \\ v_{\text{zncc}} \end{bmatrix}} \quad (1.3.49)$$

Per completezza si riportano di seguito a titolo di esempio i kernel derivativi di Sobel, i quali permettono mediante semplice convoluzione (come si vedrà anche in seguito) il calcolo delle diverse quantità che compaiono nell'ultima espressione:

$$\frac{\partial^2 f(u,v)}{\partial u^2} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ -2 & -4 & -2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\frac{\partial^2 f(u,v)}{\partial v^2} = \frac{1}{4} \begin{bmatrix} 1 & -2 & 1 \\ 2 & -4 & 2 \\ 1 & -2 & 1 \end{bmatrix}$$

$$\frac{\partial f(u,v)}{\partial u \partial v} = \frac{1}{4} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\frac{\partial f(u,v)}{\partial u} = \frac{1}{8} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\frac{\partial f(u,v)}{\partial v} = \frac{1}{8} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (1.3.50)$$

Ovviamente esistono una molteplicità di questi operatori la cui scelta dipende caso per caso dalle caratteristiche della soluzione che si ricerca.

Capitolo 2

Correlazione digitale d’immagine 2D

Si presenta ora la tecnica della correlazione digitale d’immagine bidimensionale, la quale come precedentemente affermato partendo dalla nascita del concetto di flusso ottico ([15],[18]) ha subito nel tempo una forte evoluzione relativa al campo dell’ingegneria meccanica ed è oggi riconosciuta universalmente come metodo valido per la caratterizzazione dei materiali.

Partendo perciò dal problema della definizione di un criterio che permetta l’effettiva correlazione di più immagini, si sviluppa la trattazione considerando via via i modelli di spostamento e di compensazione che permettono l’applicazione del DIC2D in maniera sempre più accurata.

Risulta opportuno precisare, vista la grande mole di pubblicazioni relative alla tematica, che la trattazione si focalizza sulla teoria matematica che sta alla base del metodo.

Per ogni ulteriore approfondimento riguardo all’applicazione di modelli più complicati, e lo studio degli errori associati si rimanda alla bibliografia in [19](E.g. M. A. Sutton, J.-J. Orteu, H. W. Schreier) e [20](Baldi A., Bertolino F.).

2.1 L’ image matching e il problema dell’apertura e della corrispondenza

L’indagine sperimentale mediante l’utilizzo del DIC si concentra, in via preliminare, sul problema della definizione di un criterio di accoppiamento delle immagini (*image matching*) per lo studio delle deformazioni e degli spostamenti, in quanto quest’ultimi sono il risultato del confronto di immagini successive del componente in esame, nel suo passaggio dalla configurazione indeformata a quella deformata, in seguito all’

applicazione di carichi.

Va sottolineato che tale confronto è possibile solo sotto opportune condizioni: generalmente infatti risulta alquanto difficile trovare la corrispondenza di un singolo pixel in immagini successive dello stesso pezzo. Il livello di grigio di un singolo *pixel* può presentarsi anche in corrispondenza delle posizioni occupate da migliaia di altri *pixel*, viene così a mancare l'univocità fra i livelli di grigio competenti a un dato gruppo di pixel in una data posizione nelle diverse immagini.

Per ovviare a queste problematiche si considera generalmente una sub-immagine (*template*) della configurazione indeformata, centrata sul livello di grigio del pixel corrispondente al punto che si vuole rintracciare.

Si nota, tuttavia, che anche in queste condizioni, nonostante si sia aggiunta ulteriore informazione alla successiva applicazione del metodo DIC, il problema dell'assenza di univocità fra le posizioni occupate da ogni singolo *pixel* in scatti successivi permane ancora, come si può dimostrare di seguito, dove per semplicità si fa riferimento al caso semplificato di una linea.

Si nota in riferimento alla (fig. 2.1). che, mentre si può determinare la componente perpendicolare alla linea del vettore spostamento, quella parallela è indefinita: ogni punto appartenente alla linea nella configurazione indeformata, può muoversi arbitrariamente rispetto alla posizione iniziale, lungo tale direzione.

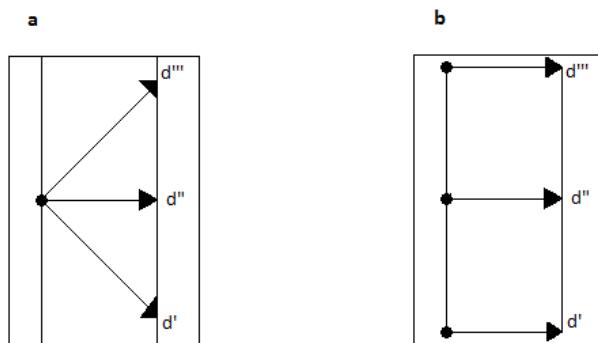


Fig. 2.1 - Il problema dell'apertura nel confronto fra immagini. a Lo spostamento del generico punto appartenente alla linea non risulta determinabile in tutte le sue componenti. b Aumentando l'apertura della finestra, vengono inglobati anche gli estremi

della linea. Ora il vettore spostamento è univocamente determinato.

Questa ambiguità viene generalmente inquadrata come *il problema dell'apertura*, il quale a sua volta rappresenta un caso particolare del più generico *problema della corrispondenza*. Nel caso della linea questa caratteristica è rappresentata dagli estremi: noti questi, infatti, si può associare univocamente un vettore spostamento per ogni punto appartenente ad essa.

Nella pratica comune il problema viene arginato, mediante l'applicazione sulla superficie dell'oggetto studiato, di *texture* come lo *speckle pattern* (fig. 2.2), caratterizzato dall'essere casuale, isotropico e non periodico, il quale in virtù della completa adesione alla superficie del provino subisce istante per istante la stessa deformazione del pezzo, assicurando sempre l'univocità della corrispondenza anche in presenza di spostamenti.

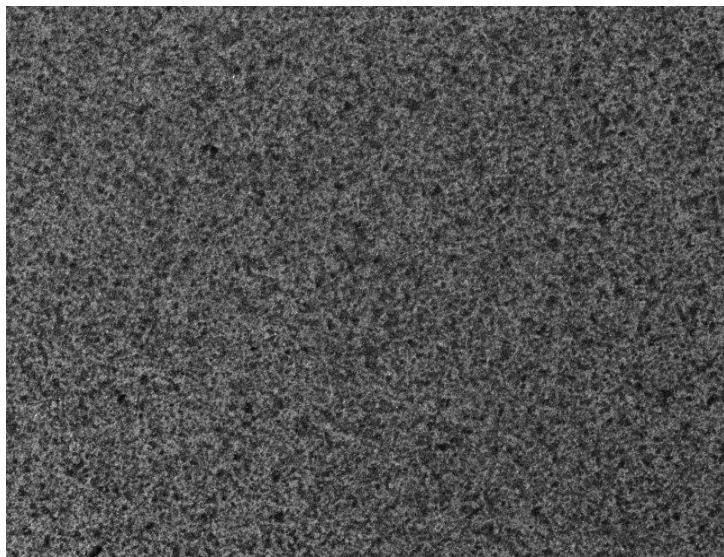


Fig. 2.2 - Esempio di speckle pattern.

Una volta definita la sub-immagine e nota la distribuzione d' intensità di quest'ultima, il problema dell'*image matching* si traduce in quello della ricerca di quella sub-immagine nella configurazione deformata, che presenti una distribuzione di intensità più simile alla configurazione di partenza, indeformata (*template matching*). Esistono vari algoritmi che permettono questo confronto, si considerano perciò di seguito i principali.

2.1.1 Analisi differenziale del problema

Per definire il problema della stima degli spostamenti nel piano o nello spazio, risulta utile affrontare il problema in termini monodimensionali (fig 2.3).

Si consideri di acquisire due immagini in momenti successivi, al tempo t e $t+\Delta t$.

Sia ora I_t il valore del livello d'intensità del livello di grigio in una data posizione nella matrice dei pixels ad un dato istante t , ed $I_{t+\Delta t}$ il livello di grigio nella stessa posizione al tempo $t+\Delta t$. Nel caso le due immagini rappresentino lo stesso oggetto dotato di moto rigido, si può supporre a condizione che l'irradianza superficiale si mantenga costante fra le diverse immagini, che $I_{t+\Delta t}$ per ogni posizione al tempo $t+\Delta t$ sia esattamente pari a I_t al tempo t , dove Δt quantifica l'intervallo di tempo intercorso, fra i due scatti. Quanto detto equivale ad affermare che per tutte le posizioni dei pixel le intensità risultano traslate di una certa quantità u .

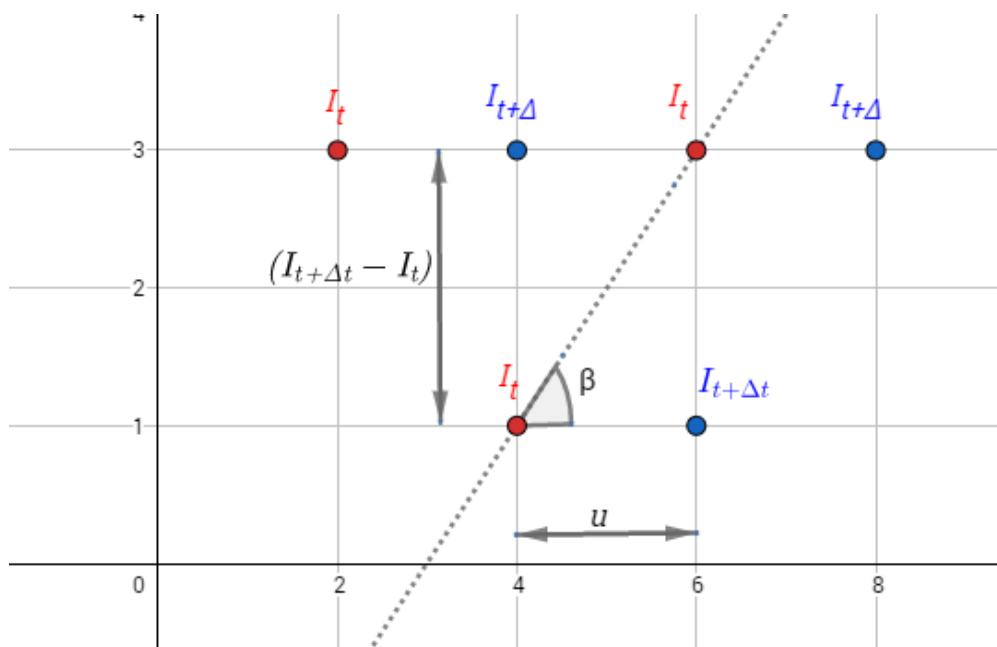


Fig. 2.3- Rappresentazione del movimento rigido monodimensionale, della distribuzione d'intensità in istanti successivi.

Risulta pertanto valida la seguente relazione che esprime la condizione della costanza dell'irradianza superficiale nel caso monodimensionale:

$$I(x + u, t + \Delta t) = I(x, t) \quad (2.1.1)$$

Accettando l'ipotesi di trovarsi in presenza piccoli spostamenti, possiamo sviluppare in serie di Taylor il membro a sinistra dell'equazione (2.1.1), troncando lo sviluppo al primo ordine si ottiene:

$$I(x + u, t + \Delta t) = I(x, t) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial t} \Delta t + R \quad (2.1.2)$$

Sostituendo l'ultima relazione nella (2.1.1) si ottiene:

$$I(x, t) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial t} \Delta t = I(x, t) \quad (2.1.3)$$

La semplificazione della (2.1.3) porta alla seguente relazione:

$$\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial t} \Delta t = 0 \quad (2.1.4)$$

esplicitando lo spostamento u risulta ora:

$$u = -\frac{\frac{\partial I}{\partial t} \Delta t}{\frac{\partial I}{\partial x}} \quad (2.1.5)$$

Come verrà chiarito nel seguito della trattazione i termini di derivate sono calcolati semplicemente come differenze finite, questo permette ora di poter riscrivere il termine di derivazione rispetto al tempo come:

$$\frac{\partial I}{\partial t} = \frac{I(x, t) - I(x + u, t + \Delta t)}{\Delta t} \quad (2.1.6)$$

che permette di ottenere la seguente espressione per lo spostamento:

$$u = \frac{[I(x+u, t + \Delta t) - I(x, t)]}{\frac{\partial I}{\partial x}} \quad (2.1.7)$$

Risulta perciò, che in presenza di piccoli spostamenti, questi sono pari al rapporto fra la variazione di intensità, in corrispondenza di uno stesso pixel e il coefficiente angolare della distribuzione dell'intensità calcolato nella medesima posizione. In termini geometrici, sempre in riferimento alla fig2.3, si può infatti notare come il calcolo fin qui sviluppato equivalga al rapporto fra il segmento ($I_{t+\Delta t} - I_t$) che definisce la variazione di intensità, e la tangente dell'angolo β , ottenuta derivando localmente la distribuzione di intensità.

Procedendo oltre nell'esposizione del problema in termini generali, e considerando ora il caso 2D+t dimensionale, che di più si avvicina alla realtà dei fatti, risulta che la distribuzione di intensità I sarà funzione delle due coordinate spaziali x, y e del tempo t . Sfruttando nuovamente la condizione della costanza dell'irradianza superficiale, in inglese *brightness constancy constraint*, si può scrivere:

$$I(x+u, y+v, t+\Delta t) = I(x, y, t) \quad (2.1.8)$$

L'espressione (2.1.8) rappresenta l'invarianza del livello di grigio del generico punto di coordinate (x,y) sul piano immagine al tempo t e caratterizzato da un'intensità $I(x,y,t)$, a seguito di uno spostamento $\Delta x, \Delta y$, al tempo $t+\Delta t$ calcolato sulla base di immagini successive.

Sviluppando in serie di Taylor il membro a destra dell'equazione, e trascurando i termini di ordine superiore risulta:

$$I(x+u, y+v, t+\Delta t) = I(x, t) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} \Delta t + R \quad (2.1.9)$$

Sostituendo ora la (2.1.9) nella (2.1.8) si perviene alla seguente espressione:

$$\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} \Delta t = 0 \quad (2.1.10)$$

Dividendo ora ogni membro della (1.2.10) per Δt l'espressione può essere riscritta in termini delle componenti del vettore velocità (\tilde{u}, \tilde{v}) come:

$$\frac{\partial I}{\partial x} \tilde{u} + \frac{\partial I}{\partial y} \tilde{v} + \frac{\partial I}{\partial t} = 0 \quad (2.1.11)$$

Come si può notare la (1.2.11) essendo funzione delle due incognite (\tilde{u}, \tilde{v}) non può essere risolta mediante esplicitazione diretta.

Esistono, tuttavia numerosi algoritmi, che riescono a determinare il flusso ottico, a diversi gradi di accuratezza. Questi algoritmi sono tutti accumunati dall'introduzione di ulteriori condizioni di restrizione senza i quali la soluzione del problema risulterebbe non unica.

2.1.2 Calcolo delle derivate

Risulta fondamentale a questo punto della trattazione specificare come sono calcolate le derivate espresse nelle precedenti equazioni.

Numericamente sono calcolate come rapporti incrementali, non essendo in generale la mappa di intensità una funzione continua nelle due direzioni spaziali.

Si precisa a tal proposito che a livello pratico nel calcolo della correlazione digitale si effettua sempre a priori un'interpolazione dei livelli d'intensità delle immagini originali al fine di ottenere una distribuzione continua, questo permette il calcolo delle derivate anche nelle posizioni non intere dei pixel.

Successivamente in generale si applica un'interpolazione mediante fitting di una funzione di forma al campo degli spostamenti calcolati, al fine di poter utilizzare a seconda dei casi i modelli di deformazione più consoni al particolare problema studiato.

In pratica ponendo x come la posizione di un punto espressa in pixel risulta che la derivata in direzione x sarà pari al rapporto incrementale delle intensità nel punto come riportato di seguito:

$$\frac{\partial[I(x)]}{\partial x} = \frac{I(x + \Delta x) - I(x)}{\Delta x} \quad (2.1.12)$$

dove per mantenere un certo grado di correlazione Δx generalmente assume valore pari ad 1: si considerano cioè di solito pixel adiacenti nel calcolo differenziale.

Per completezza si specifica anche che le derivate possono essere calcolate in diverse maniere a seconda che si considerino le differenze all'indietro, in avanti oppure le differenze centrali, come di seguito riportato:

$$\begin{aligned} \frac{\partial[I(x)]}{\partial x} &= \frac{I(x) - I(x + \Delta x)}{\Delta x} \\ \frac{\partial[I(x)]}{\partial x} &= \frac{I(x + \Delta x) - I(x - \Delta x)}{2\Delta x} \end{aligned} \quad (2.1.13)$$

Per ognuno di questi modi si può definire quella che in gergo tecnico è chiamata *Derivative Mask*, che matematicamente equivale a seconda dei casi ad un vettore o ad una matrice.

Per chiarire meglio il significato si consideri una funzione d'intensità che assume in sequenza i valori interi sotto riportati:

$$I(x) = 12 \quad 16 \quad 4 \quad 4 \quad 56 \quad 34 \quad 32 \quad 12$$

Calcolando la differenza finita all'indietro si ottiene la seguente funzione derivata:

$$\frac{\partial[I(x)]}{\partial x} = \frac{I(x) - I(x + \Delta x)}{\Delta x} = 4 \quad -12 \quad 0 \quad 52 \quad -22 \quad -2 \quad 20$$

Quanto effettuato equivale all'applicazione alla funzione di partenza, della seguente mask derivativa:

$$h=[1,-1]$$

Nel caso più generale, in relazione al nostro campo d'applicazione, dove si ha a che fare con immagini e quindi matrici bidimensionali, l'applicazione di una mask derivativa ad una immagine equivale al calcolo della convoluzione fra l'immagine e il suo kernel, che nel nostro caso è rappresentato dal vettore o dalla matrice della mask.

Considerando un kernel di grandezza $n \times m$ si definisce la funzione correlazione come:

$$F(i,j) = f \otimes h = \sum_{k=-\frac{n}{2}}^{\frac{n}{2}} \sum_{l=-\frac{m}{2}}^{\frac{m}{2}} h(k, l) \cdot f(i+k, j+l) \quad (2.1.14)$$

E la convoluzione come:

$$F(i,j) = f * h = \sum_{k=-\frac{n}{2}}^{\frac{n}{2}} \sum_{l=-\frac{m}{2}}^{\frac{m}{2}} h(k, l) \cdot f(i-k, j-l) \quad (2.1.15)$$

Nelle quali i diversi termini hanno il seguente significato

- *F: immagine*
- *H: kernel*
- *F: immagine finale*

Si riporta di seguito un esempio numerico al fine di chiarire meglio la differenza fra le due operazioni.

Scelta un'immagine *f* ed un kernel *h* 3x3 come riportato di seguito:

$$f = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 8 & 7 & 6 \\ 4 & 5 & 6 & 3 \end{bmatrix} \quad h = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

si ottiene dalla correlazione la seguente immagine finale F:

$$F = f \otimes h = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \textcolor{red}{2} & 2 & 0 \\ 0 & 2 & -2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Prendendo come esempio il calcolo del termine nella seconda riga e nella seconda colonna della matrice F, possiamo dire che tutti i termini della matrice risultante sono calcolati maniera simile a quella riportata di seguito:

$$F(2,2) = 1*3 + 0*2 - 1*1 + 1*7 + 0*6 - 1*5 + 1*7 + 0*8 - 1*9 = \textcolor{red}{2}.$$

Calcolando invece la convoluzione si avrebbe:

$$F = f * h = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2 & -2 & 0 \\ 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Risulta perciò chiaro come la convoluzione fra l'immagine e il suo kernel, equivalga alla correlazione fra l'immagine ed il kernel ruotato di 180° come si può notare dal seguente esempio dove si ruota il kernel h dell'esempio considerato:

$$f = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 8 & 7 & 6 \\ 4 & 5 & 6 & 3 \end{bmatrix} \quad h = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

$$F(2,2) = -1*3 + 0*2 + 1*1 - 1*7 + 0*6 + 1*5 - 1*7 + 0*8 + 1*9 = -2.$$

Si noti che in caso di kernel simmetrico non vi è differenza fra il calcolo di correlazione e convoluzione.

Per quanto riguarda invece l'immagine finale F si nota che risultano persi i valori ai bordi dell'immagine, questi non risultano calcolabili non essendoci per queste posizioni un *neighbourhood* di pixel idoneo all'applicazione del kernel.

Le dimensioni del contorno perso dipendono dalla dimensione del kernel, e sono facilmente calcolabili note la dimensione di quest'ultimo.

Si ha per esempio che se il kernel ha dimensione $n \times m$ allora il numero di pixel persi nelle due direzioni è dato dall'arrotondamento per difetto delle dimensioni del kernel come riportato di seguito:

$$s_{verticale} = \text{round}\left(\frac{n}{2}\right)$$

$$s_{orizzontale} = \text{round}\left(\frac{m}{2}\right) \quad (2.1.16)$$

Esistono tuttavia numerosi metodi per risolvere il problema sopra citato, la cui scelta dipende dall' accuratezza che si vuole ottenere: di solito si effettua uno *zero padding*, estendendo i contorni dell'immagine iniziale con tanti zeri quanti sarebbero quelli persi, in modo che il kernel possa essere convolto in ogni punto dell'immagine originale, oppure si replica l'immagine originale lungo i bordi tale da risolvere il problema.

Risulta chiaro quindi che in funzione al kernel scelto si possono calcolare tutti i vari tipi di differenze finite, che equivalgono alle derivate spaziali presenti nei diversi metodi che verranno presentati.

2.1.3 Algoritmo Horn&Schunck

Uno delle prime soluzioni proposte al problema della determinazione del flusso ottico è attribuita ai ricercatori *B.K.P. Horn e B.G. Schunck*.[18].

La soluzione in questo caso consiste nel considerare il flusso ottico come un funzionale dell'energia globale sul piano immagine:

$$E = \iint \left\{ \left(\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} \Delta t \right)^2 + \lambda \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 \right] \right\} dx dy \quad (2.1.17)$$

Ciò che hanno fatto Horn e Schunck è vedere la risoluzione del flusso ottico come un problema di ottimizzazione, consistente nel minimizzare l'energia globale associata al flusso ottico.

In riferimento alla formulazione presentata nella (2.1.17) possiamo chiaramente distinguere i diversi contributi dati dai due membri dentro le parentesi graffe, in particolare risulta che il termine:

$$\left(\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} \Delta t \right)^2 \quad (2.1.18)$$

rappresenta proprio la condizione per la quale le componenti u e v del flusso ottico devono soddisfare la condizione della costanza dell'irradianza superficiale: in un caso ideale quindi se u e v rappresentassero i valori esatti delle componenti ricercate, la quantità espressa dentro parentesi della (2.1.18) dovrebbe essere rigorosamente pari a zero.

Poiché quanto affermato non risulta mai verificato essendo l'intero processo dell'acquisizione delle immagini affetto da disturbo, si quadra quest'ultima quantità nel processo di ottimizzazione.

Altra condizione introdotta nel metodo è che il flusso ottico debba essere liscio (*smooth*) nell'intorno del pixel considerato, dove per soddisfare tale condizione ci si limita al calcolo delle derivate prime.

Quest'ultima condizione è espressa dal termine:

$$\lambda \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 \right] \quad (2.1.19)$$

Siccome ci si aspetta che il flusso ottico sia liscio, le derivate nell'equazione sopra riportata sono numericamente poco significative, per minimizzare l'errore anche in questo caso quindi si eleva al quadrato.

Il problema definito dall'equazione iniziale si traduce quindi in una minimizzazione delle due componenti appena discusse.

Poiché la derivazione matematica effettuata in maniera rigorosa presuppone la conoscenza di nozioni di calcolo variazionale, risulta più conciso riportare i risultati ottenuti.

Si ha la seguente soluzione per i valori del vettore spostamento ricercato:

$$u = u_{\text{medio}} - \frac{\partial I}{\partial x} \cdot \frac{\frac{\partial I}{\partial x} \cdot u_{\text{medio}} + \frac{\partial I}{\partial y} \cdot v_{\text{medio}} + \frac{\partial I}{\partial t}}{\lambda + \left(\frac{\partial I}{\partial x} \right)^2 + \left(\frac{\partial I}{\partial y} \right)^2} \quad (2.1.20)$$

$$v = v_{\text{medio}} - \frac{\partial I}{\partial y} \cdot \frac{\frac{\partial I}{\partial x} \cdot u_{\text{medio}} + \frac{\partial I}{\partial y} \cdot v_{\text{medio}} + \frac{\partial I}{\partial t}}{\lambda + \left(\frac{\partial I}{\partial x} \right)^2 + \left(\frac{\partial I}{\partial y} \right)^2} \quad (2.1.21)$$

con u_{medio} e v_{medio} pari al valore medio delle componenti del vettore spostamento calcolate alle posizioni che si trovano immediatamente sopra, sotto, destra e sinistra al pixel del quale si vuole determinare lo spostamento.

Risulta in definitiva che la procedura per ottenere il campo di spostamenti è di tipo iterativo, infatti in partenza sia u_{medio} e v_{medio} sono ignoti; per questo motivo si inizializza la matrice degli spostamenti al valore zero e si procede iterando finché non si raggiunge una condizione di uscita dal calcolo che dipende dalla risoluzione cercata.

Come esempio di applicazione si usano le seguenti due immagini in figura 2.4 ottenute mediante il software Paint su Windows tramite il quale si è applicato lo speckle pattern:

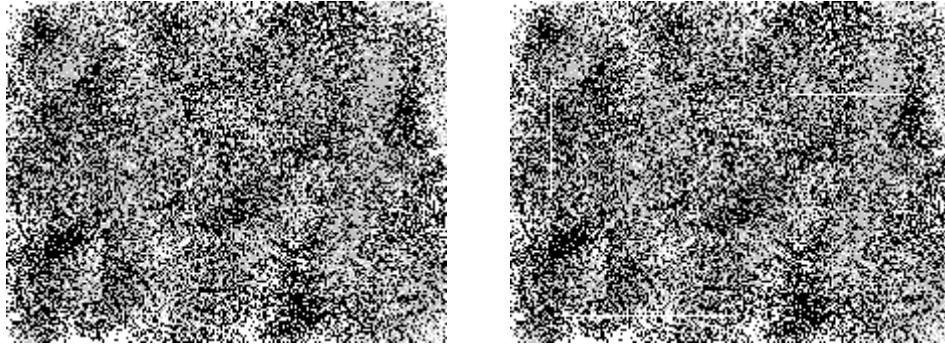


Fig. 2.4- Immagini digitali utilizzate nell'applicazione della correlazione digitale: a sinistra l'immagine originale, a destra l'immagine nella quale si è imposta una traslazione orizzontale e verticale di alcune regioni.

Anche se visivamente poco chiaro la seconda immagine fig. 2.4 è stata ottenuta semplicemente applicando uno spostamento di 1 pixel a diverse parti rettangolari della prima immagine.

Relativamente al software allegato in appendice si precisa che la funzione riportata per il calcolo prende in ingresso oltre alle due immagini anche il valore del coefficiente λ , posto pari a 1000 nel caso studiato, e il numero di iterazioni che si vogliono effettuare, nel nostro caso risulta pari a 200.

Il risultato fornito dal software è rappresentato nella seguente figura 2.5:

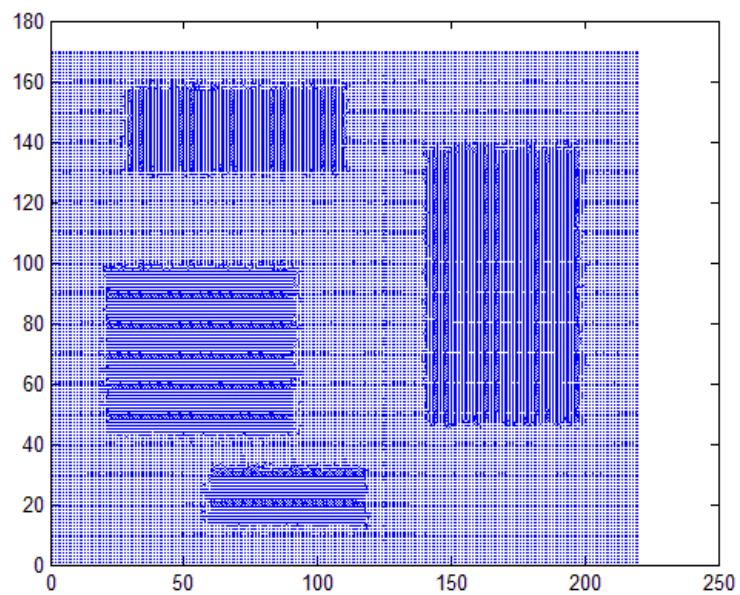


Fig. 2.5- Risultato della correlazione.

Andando ad ingrandire le varie parti dell'output fornito si visualizzano i seguenti risultati:

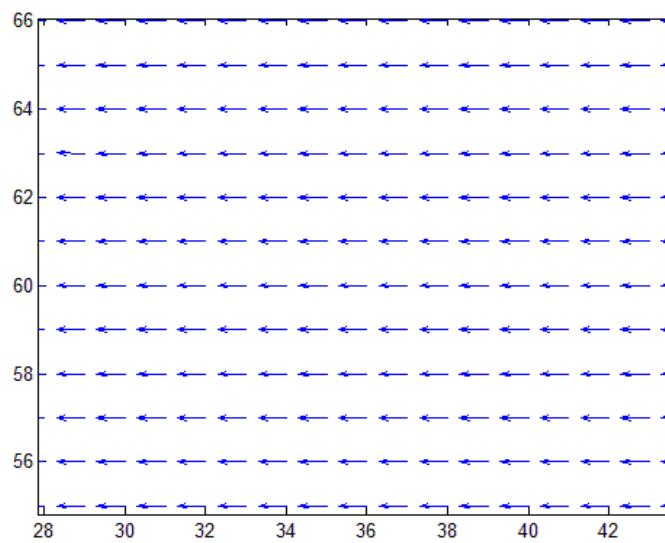


Fig. 2.6- Mappa spostamenti orizzontali.

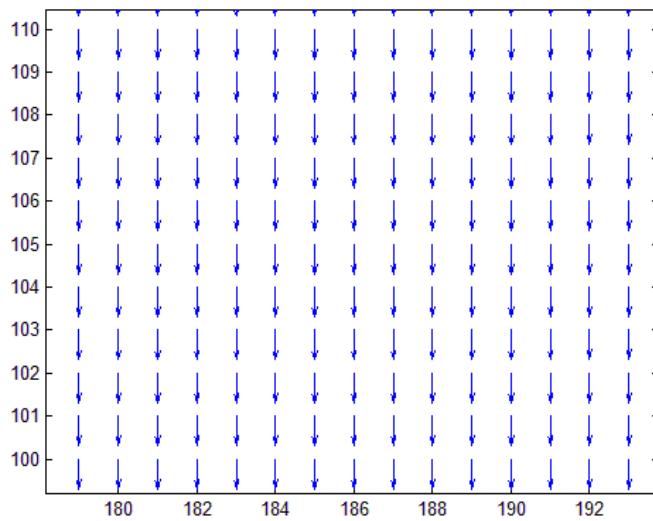


Fig. 2.7- Mappa spostamenti verticali.

Come si può osservare dai risultati ottenuti, nonostante il metodo riesca effettivamente a determinare il flusso ottico, risulta tuttavia di difficile gestione ed applicazione ai casi paratici, fatto legato principalmente al fattore di *smoothing* λ la cui scelta si ripercuote in maniera marcata sui risultati finali.

2.1.4 Algoritmo Lucas-Kanade

Un' altro dei metodi più intuitivi mai utilizzati per risolvere il flusso ottico è dovuto ai ricercatori Lucas-Kanade [15].

Come punto iniziale della trattazione, si consideri l'equazione (2.1.10), riportata per semplicità di seguito:

$$\frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \frac{\partial I}{\partial t} \Delta t = 0 \quad (2.1.10)$$

Per semplicità si consideri ora come esempio una finestra di 3x3 pixel, una sub immagine in pratica, centrata sul pixel del quale si vuole trovare il flusso come di seguito mostrato:

$$W = \begin{bmatrix} 34 & 56 & 67 \\ 78 & 23 & 56 \\ 112 & 34 & 78 \end{bmatrix}$$

Per ognuno dei pixel della finestra risulta possibile scrivere la (2.1.10), ottenendo quindi 9 equazioni:

$$\begin{aligned} \frac{\partial [^1I(x_1, y_1, t)]}{\partial x} u + \frac{\partial [^1I(x_1, y_1, t)]}{\partial y} v + \frac{\partial [^1I(x_1, y_1, t)]}{\partial t} \Delta t &= 0 \\ \frac{\partial [^1I(x_2, y_2, t)]}{\partial x} u + \frac{\partial [^1I(x_2, y_2, t)]}{\partial y} v + \frac{\partial [^1I(x_2, y_2, t)]}{\partial t} \Delta t &= 0 \\ &\vdots \\ \frac{\partial [^1I(x_9, y_9, t)]}{\partial x} u + \frac{\partial [^1I(x_9, y_9, t)]}{\partial y} v + \frac{\partial [^1I(x_9, y_9, t)]}{\partial t} \Delta t &= 0 \end{aligned} \quad (2.1.22)$$

Relativamente alle equazioni riportate si può notare che essendo le derivate calcolate come differenze finite si ha per la derivata rispetto al tempo un'espressione della forma:

$$\frac{\partial [^1I(x, y, t)]}{\partial t} = \frac{I(x, y, t + \Delta t) - I(x, y, t)}{\Delta t} \quad (2.1.23)$$

Dalla quale risulta che quantitativamente l'intervallo di tempo fra le due immagini è irrilevante ai fini della determinazione dello spostamento essendo l'intervallo di tempo semplificabile una volta sostituita la (2.1.23) nella (2.1.10) come di seguito riportato:

$$\frac{\partial [^1I(x, y, t)]}{\partial t} \Delta t = I(x, y, t + \Delta t) - I(x, y, t) \quad (2.1.24)$$

Se ora si suppone che lo spostamento u, v per tutti i pixel della finestra considerata sia lo stesso si può scrivere il seguente sistema di equazioni:

$$\begin{cases} \frac{\partial [^1I(x_1, y_1, t)]}{\partial x} u + \frac{\partial [^1I(x_1, y_1, t)]}{\partial y} v + [I(x_1, y_1, t + \Delta t) - I(x_1, y_1, t)] = 0 \\ \vdots \\ \frac{\partial [^1I(x_9, y_9, t)]}{\partial x} u + \frac{\partial [^1I(x_9, y_9, t)]}{\partial y} v + [I(x_9, y_9, t + \Delta t) - I(x_9, y_9, t)] = 0 \end{cases} \quad (2.1.25)$$

In forma matriciale la (2.1.25) equivale anche ad:

$$\begin{bmatrix} \frac{\partial [^1I(x_1, y_1, t)]}{\partial x} & \frac{\partial [^1I(x_1, y_1, t)]}{\partial y} \\ \vdots & \\ \frac{\partial [^1I(x_9, y_9, t)]}{\partial x} & \frac{\partial [^1I(x_9, y_9, t)]}{\partial y} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I(x_1, y_1, t) - I(x_1, y_1, t + \Delta t) \\ \vdots \\ I(x_9, y_9, t) - I(x_9, y_9, t + \Delta t) \end{bmatrix} \quad (2.1.26)$$

Indicando ora la matrice dei coefficienti della (2.1.26) con $[A]$, il vettore delle incognite con $\{u\}$ e quello dei termini noti con I_t , si può riscrivere il sistema in forma più compatta come segue:

$$[A] \cdot \{u\} = I_t \quad (2.1.27)$$

Come si può notare dall'ultima espressione il sistema associato alla risoluzione del problema risulta sovradeterminato, la matrice $[A]$ essendo rettangolare non è invertibile. La soluzione consta nel forzare la matrice $[A]$ a diventare quadrata tale che si possa invertire.

Questo processo viene effettuato nel seguente modo:

$$[A]^t [A] \cdot \{u\} = [A]^t I_t \quad (2.1.28)$$

In pratica si moltiplica per la trasposta della matrice $[A]$ da entrambe le parti, questo fa sì che la nuova matrice dei coefficienti $[A]^t[A]$ sia quadrata, e in particolare risulta una 2×2 ; anche $[A]^t I_t$ risulta un vettore 2×1 .

A questo punto il sistema risulta risolvibile e quindi si procede con il calcolo del vettore delle incognite:

$$\{u\} = ([A]^t [A])^{-1} [A]^t I_t \quad (2.1.29)$$

Il prodotto $([A]^t [A])^{-1} [A]^t$ nella (2.1.29) rappresenta matematicamente la *pseudo inversa* della matrice dei coefficienti $[A]$.

Procedendo in maniera diversa ora al fine di esplicare meglio le assunzioni sul modello di spostamento scelto nonché il valore finale del vettore spostamento, si pone il problema in termini di un'ottimizzazione, mediante minimizzazione quadratica della (2.1.10) per ogni pixel della finestra considerata.

In particolare si imposta il criterio *SSD (sum of squared differences)*, applicato di volta in volta a regioni che definiscono delle subimmagini di quelle iniziali.

Scelta quindi una finestra di dimensioni $n \times m$ pixels nell'immagine oggetto di studio, il problema viene posto come la ricerca del vettore spostamento che minimizzi l'errore E come riportato di seguito:

$$E = \underset{u,v}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j, y_i, t)}{\partial x} u + \frac{\partial I(x_j, y_i, t)}{\partial y} v + [I(x_j, y_i, t + \Delta t) - I(x_j, y_i, t)] \right)^2 \quad (2.1.30)$$

Supponendo che lo spostamento cercato sia semplicemente di traslazione si può scrivere:

$$\begin{aligned} u &= c_1 \\ v &= c_2 \end{aligned} \quad (2.1.31)$$

La minimizzazione dell'errore E riportato nella (2.1.30) conduce al seguente sistema di equazioni:

$$\begin{cases} \frac{\partial E}{\partial u} = \frac{\partial E}{\partial u} \frac{\partial u}{\partial c_1} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I(x_j, y_i, t)}{\partial x} u + \frac{\partial I(x_j, y_i, t)}{\partial y} v - I_t \right) \left(\frac{\partial I(x_j, y_i, t)}{\partial x} \cdot \frac{\partial u}{\partial c_1} \right) = 0 \\ \frac{\partial E}{\partial v} = \frac{\partial E}{\partial v} \frac{\partial v}{\partial c_2} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I(x_j, y_i, t)}{\partial x} u + \frac{\partial I(x_j, y_i, t)}{\partial y} v - I_t \right) \left(\frac{\partial I(x_j, y_i, t)}{\partial y} \cdot \frac{\partial v}{\partial c_2} \right) = 0 \end{cases} \quad (2.1.32)$$

Si precisa che nell'espressione (2.1.32) si è usata la seguente notazione compatta per il termine della derivata temporale:

$$I_t = I(x_j, y_i, t) - I(x_j, y_i, t + \Delta t) \quad (2.1.33)$$

e si userà inoltre la seguente notazione per i termini delle derivate:

$$\frac{\partial I}{\partial x} = \frac{\partial I(x_j, y_i, t)}{\partial x}, \quad \frac{\partial I}{\partial y} = \frac{\partial I(x_j, y_i, t)}{\partial y} \quad (2.1.34)$$

Risulta a questo punto che poiché lo spostamento ricercato è modellato come di pura traslazione si ottiene il seguente valore per le derivate dei termini dello spostamento:

$$\frac{\partial u}{\partial c_1} = 1; \quad \frac{\partial v}{\partial c_2} = 1 \quad (2.1.35)$$

In accordo con le considerazioni fatte si può riscrivere il sistema (2.1.32) nel seguente modo:

$$\begin{cases} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right)^2 \cdot u + \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \cdot v = \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) I_t \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \cdot u + \left(\frac{\partial I}{\partial y} \right)^2 \cdot v = \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right) I_t \end{cases} \quad (2.1.36)$$

In forma matriciale la (2.1.36) equivale anche ad:

$$\begin{bmatrix} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right)^2 & \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) & \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right)^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) I_t \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right) I_t \end{bmatrix} \quad (2.1.37)$$

Si è in grado perciò di calcolare il vettore spostamento come riportato di seguito:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right)^2 & \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) & \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right)^2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) I_t \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right) I_t \end{bmatrix} \quad (2.1.38)$$

In definitiva esplicitando le espressioni delle singole componenti dello spostamento, si ottiene la seguente espressione finale per esse:

$$u = \frac{\sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \cdot \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right) I_t - \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right)^2 \cdot \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) I_t}{\sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right)^2 \cdot \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right)^2 - \left(\sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \right)^2} \quad (2.1.39)$$

$$v = \frac{\sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) I_t \cdot \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) - \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right)^2 \cdot \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right) I_t}{\sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right)^2 \cdot \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial y} \right)^2 - \left(\sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \right) \left(\frac{\partial I}{\partial y} \right) \right)^2} \quad (2.1.40)$$

2.1.5 Implementazione iterativa dell'algoritmo Lucas-Kanade

I metodi visti per la risoluzione del flusso ottico sono tutti accumunati dall'assunzione base che lo spostamento da ricercare sia abbastanza piccolo, compreso nel range del pixel.

La formulazione vista al paragrafo precedente suggerisce tuttavia la possibilità di ottimizzare il processo della ricerca del vettore spostamento, considerando successivamente anche spostamenti superiori al pixel.

Quanto detto infatti risulta corretto in relazione alla possibilità di poter impostare il calcolo del vettore spostamento secondo un'implementazione iterativa dell'algoritmo Lucas-Kanade.

Si supponga infatti di aver ottenuto attraverso l'applicazione dell'algoritmo di ricerca uno spostamento di valore \bar{u}, \bar{v} ; in questo caso si può riscrivere ulteriormente l'equazione (2.1.10) come:

$${}^1I(x + \bar{u} + \Delta u, y + \bar{v} + \Delta v, t + \Delta t) = {}^1I(x + \bar{u}, y + \bar{v}, t) \quad (2.1.41)$$

Nella quale si indica con Δu e Δv lo spostamento che raffina il risultato precedentemente ottenuto \bar{u}, \bar{v} .

Sviluppando in serie di Taylor il termine a sinistra della (2.1.41) si ottiene:

$$\begin{aligned} {}^1I(x + \bar{u} + \Delta u, y + \bar{v} + \Delta v, t + \Delta t) &= {}^1I(x + \bar{u}, y + \bar{v}, t) + \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial x} \Delta u + \\ &+ \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial y} \Delta v + \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial t} \Delta t \end{aligned} \quad (2.1.42)$$

Sostituendo l'ultima espressione nella (2.1.41) la si può riscrivere come:

$${}^1I(x + \bar{u}, y + \bar{v}, t) + \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial x} \Delta u + \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial y} \Delta v +$$

$$+ \frac{^1I(x + \bar{u} + \Delta u, y + \bar{v} + \Delta v, t + \Delta t) - ^1I(x + \bar{u}, y + \bar{v}, t)}{\Delta t} \Delta t = ^1I(x + \bar{u}, y + \bar{v}, t)$$

(2.1.43)

Semplificando la (2.1.43), ed indicando il termine della derivata temporale nel seguente modo:

$$I_t = I(x, y, t) - I(x + \bar{u} + \Delta u, y + \bar{v} + \Delta v, t + \Delta t) \quad (2.1.44)$$

risulta definito il seguente nuovo problema di ottimizzazione:

$$E = \underset{\Delta u, \Delta v}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x + \bar{u}, y + \bar{v}, t)}{\partial x} \Delta u + \frac{\partial I(x + \bar{u}, y + \bar{v}, t)}{\partial y} \Delta v - I_t \right)^2 \quad (2.1.45)$$

Modellando come in precedenza lo spostamento come di semplice traslazione si impone:

$$\begin{aligned} \Delta u &= c_1 \\ \Delta v &= c_2 \end{aligned} \quad (2.1.46)$$

La minimizzazione della (2.1.45) porta in questo caso al seguente sistema:

$$\begin{cases} \frac{\partial E}{\partial \Delta u} = \frac{\partial E}{\partial \Delta u} \frac{\partial \Delta u}{\partial c_1} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \Delta u + \frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \cdot \frac{\partial (\Delta u)}{\partial c_1} \right) = 0 \\ \frac{\partial E}{\partial \Delta v} = \frac{\partial E}{\partial \Delta v} \frac{\partial \Delta v}{\partial c_2} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \Delta u + \frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \cdot \frac{\partial (\Delta v)}{\partial c_2} \right) = 0 \end{cases} \quad (2.1.47)$$

Osservando che:

$$\begin{aligned} \frac{\partial (\Delta u)}{\partial c_1} &= 1 \\ \frac{\partial (\Delta v)}{\partial c_2} &= 1 \end{aligned} \quad (2.1.48)$$

si ottiene:

$$\begin{aligned}
& \left\{ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \right)^2 \cdot \Delta u + \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \right) \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \right) \cdot \Delta v = \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \right) \cdot I_t \right. \\
& \left. \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial x} \right) \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \right) \cdot \Delta u + \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \right)^2 \cdot \Delta v = \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}, y_i + \bar{v}, t)}{\partial y} \right) \cdot I_t \right\} \\
& \quad (2.1.49)
\end{aligned}$$

Indicando ora con la matrice $[J]^{-1}$ al generico passo k l'inversa della matrice dei coefficienti associata al sistema (2.1.49):

$$[J]_k^{-1} = \begin{bmatrix} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial x} \right)^2 & \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial x} \right) \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial y} \right) \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial x} \right) \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial y} \right) & \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial y} \right)^2 \end{bmatrix}^{-1}$$

(2.1.50)

e con $\{B\}$ il vettore dei termini noti al passo k :

$$\{B\}_k = \begin{bmatrix} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial x} \right) \cdot I_t \\ \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x_j + \bar{u}_{k-1}, y_i + \bar{v}_{k-1}, t)}{\partial y} \right) \cdot I_t \end{bmatrix} \quad (2.1.51)$$

si trova la soluzione del sistema in maniera iterativa, fino a raggiungimento della convergenza:

$$\begin{bmatrix} \Delta u_k \\ \Delta v_k \end{bmatrix} = [J]_k^{-1} \cdot \{B\}_k \quad (2.1.52)$$

L'espressione trovata permette infatti di volta in volta di aggiornare il vettore spostamento per la successiva iterazione in caso non si sia già raggiunta la convergenza, come riportato di seguito:

$$\begin{aligned}\bar{\mathbf{u}}_k &= \bar{\mathbf{u}}_{k-1} + \Delta \mathbf{u}_k \\ \bar{\mathbf{v}}_k &= \bar{\mathbf{v}}_{k-1} + \Delta \mathbf{v}_k\end{aligned}\quad (2.1.53)$$

2.1.6 Modello di compensazione della trasformazione radiometrica

La trattazione finora sviluppata permette come visto di ottenere il vettore spostamento per un generico punto in instanti successivi di movimento. Risulta tuttavia che gli algoritmi presentati non tengono conto né dei disturbi legati al processo stesso di acquisizione d'immagine, né alle variazioni dell'intensità di fondo nelle immagini successive dello stesso oggetto.

La prima serie di disturbi essendo legata alla tecnologia utilizzata nella produzione delle macchine fotografiche viene per ovvi motivi studiata in contesti diversi dalla correlazione digitale d'immagine, la seconda serie di disturbi legati principalmente alla variazione dell'intensità luminosa di fondo invece non può non essere considerata nel processo di correlazione.

A tal proposito si ricorda infatti che gli algoritmi di correlazione digitale d'immagine trovano lo spostamento cercato in termini di minimizzazione quadratica della differenza fra i livelli d'intensità per regioni diverse in immagini successive.

In quest'ottica si comprende come una variazione dell'intensità luminosa di fondo possa condurre a risultati completamente errati.

Si considera perciò ora che fra le successive immagini vi possa essere anche una variazione lineare di intensità, in questo caso la condizione della costanza dell'irradianza superficiale può essere riscritta seguente forma:

$${}^1I(x + \bar{u} + \Delta u, y + \bar{v} + \Delta v, t + \Delta t) = {}^1I(x + \bar{u}, y + \bar{v}, t) \cdot \alpha + \beta \quad (2.1.54)$$

nella quale α rappresenta un termine di scala mentre β rappresenta un termine di *offset*.

Volendo trattare i due nuovi termini come incognite da ottimizzare durante la fase di calcolo, si ha sviluppando in serie di Taylor il termine a destra della (1.1.54):

$${}^1I(x + \bar{u} + \Delta u, y + \bar{v} + \Delta v, t + \Delta t) = {}^1I(x + \bar{u}, y + \bar{v}, t) + \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial x} \Delta u +$$

$$+ \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial y} \Delta v + \frac{\partial [{}^1I(x + \bar{u}, y + \bar{v}, t)]}{\partial t} \Delta t \quad (2.1.55)$$

Indicando con:

$$I = {}^1I(x + \bar{u}, y + \bar{v}, t) \quad (2.1.56)$$

ed effettuando le opportune sostituzioni si può scrivere la (2.1.54) in maniera compatta come segue:

$$I \cdot (\alpha - 1) + \frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v + \frac{\partial I}{\partial t} \Delta t - \beta = 0 \quad (2.1.57)$$

Impostando ora il problema di ottimizzazione risulta la seguente funzione obiettivo:

$$E = \underset{\Delta u, \Delta v, \alpha, \beta}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^m \left[I \cdot (\alpha - 1) + \frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v + \frac{\partial I}{\partial t} \Delta t - \beta \right]^2 \quad (2.1.58)$$

la cui minimizzazione permette di ottenere tutti i valori cercati.

Tuttavia poichè la minimizzazione porta ad un sistema di equazioni non lineari risulta, al fine di semplificare la mole di calcoli, più utile procedere in maniera diversa cercando a priori il valore ottimale dei parametri della trasformazione radiometrica.

In questo senso il problema si configura in maniera semplice come riportato di seguito:

$$\Psi = \underset{\beta, \alpha}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^m \left({}^1I(x, y, t + \Delta t) \cdot \alpha + \beta - {}^1I(x, y, t) \right)^2 \quad (2.1.59)$$

Procedendo perciò con la minimizzazione si ottiene:

$$\begin{cases} \frac{\partial \Psi}{\partial \alpha} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot [{}^1I(x, y, t + \Delta t) \cdot \alpha + \beta - {}^1I(x, y, t)] \cdot [{}^1I(x, y, t + \Delta t)] = 0 \\ \frac{\partial \Psi}{\partial \beta} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot [{}^1I(x, y, t + \Delta t) \cdot \alpha + \beta - {}^1I(x, y, t)] \cdot 1 = 0 \end{cases} \quad (2.1.60)$$

$$\begin{cases} \sum_{i=1}^n \sum_{j=1}^m [{}^l I(x, y, t + \Delta t)^2 \cdot \alpha + \beta \cdot {}^l I(x, y, t + \Delta t) - {}^l I(x, y, t) \cdot {}^l I(x, y, t + \Delta t)] = 0 \\ \sum_{i=1}^n \sum_{j=1}^m [{}^l I(x, y, t + \Delta t) \cdot \alpha + \beta - {}^l I(x, y, t)] \cdot 1 = 0 \end{cases} \quad (2.1.61)$$

$$\begin{cases} \alpha = \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^l I(x, y, t) - \beta] \cdot [{}^l I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^l I(x, y, t + \Delta t)]^2} \\ \beta = \frac{\sum_{i=1}^n \sum_{j=1}^m {}^l I(x, y, t) - {}^l I(x, y, t + \Delta t) \cdot \alpha}{\sum_{i=1}^n \sum_{j=1}^m 1} \end{cases} \quad (2.1.62)$$

Indicando ora con $N=nxm$ il numero di pixel complessivi delle immagini, si può riscrivere il termine β come:

$$\beta = \frac{\sum_{i=1}^n \sum_{j=1}^m {}^l I(x, y, t)}{N} - \frac{\sum_{i=1}^n \sum_{j=1}^m {}^l I(x, y, t + \Delta t) \cdot \alpha}{N} \quad (2.1.63)$$

Indicando ora col termine:

$${}^l \bar{I}_t = \frac{\sum_{i=1}^n \sum_{j=1}^m {}^l I(x, y, t)}{N} \quad (2.1.64)$$

che equivale alla media aritmetica delle intensità sull'immagine al tempo t , e con il termine:

$${}^l \bar{I}_{t+\Delta t} = \frac{\sum_{i=1}^n \sum_{j=1}^m {}^l I(x, y, t + \Delta t)}{N} \quad (2.1.65)$$

che rappresenta la media aritmetica delle intensità sull'immagine al tempo $t + \Delta t$, si ha che il sistema (2.1.62) assume la seguente forma:

$$\left\{ \begin{array}{l} \alpha = \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t) - {}^I \bar{I}_t + ({}^I \bar{I}_{t+\Delta t}) \cdot \alpha] \cdot [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2} \\ \\ \beta = {}^I \bar{I}_t - ({}^I \bar{I}_{t+\Delta t}) \cdot \alpha \end{array} \right. \quad (2.1.66)$$

Sviluppando la prima equazione della (2.1.66) si ottiene:

$$\alpha = \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t) \cdot {}^I I(x, y, t + \Delta t) - {}^I \bar{I}_t \cdot {}^I I(x, y, t + \Delta t) + ({}^I \bar{I}_{t+\Delta t}) \cdot \alpha \cdot {}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2} \quad (2.1.67)$$

$$\alpha = \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t) - {}^I \bar{I}_t] \cdot [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2} + \alpha \cdot ({}^I \bar{I}_{t+\Delta t}) \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2} \quad (2.1.68)$$

$$\alpha \cdot \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2 - ({}^I \bar{I}_{t+\Delta t}) \cdot [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2} = \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t) - {}^I \bar{I}_t] \cdot [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2} \quad (2.1.69)$$

Risultano perciò trovati i valori ottimali dei parametri come di seguito riportato:

$$\alpha = \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t) - {}^I \bar{I}_t] \cdot [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2 - ({}^I \bar{I}_{t+\Delta t}) \cdot [{}^I I(x, y, t + \Delta t)]} \quad (2.1.70)$$

$$\beta = {}^I \bar{I}_t - ({}^I \bar{I}_{t+\Delta t}) \cdot \frac{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t) - {}^I \bar{I}_t] \cdot [{}^I I(x, y, t + \Delta t)]}{\sum_{i=1}^n \sum_{j=1}^m [{}^I I(x, y, t + \Delta t)]^2 - ({}^I \bar{I}_{t+\Delta t}) \cdot [{}^I I(x, y, t + \Delta t)]} \quad (2.1.71)$$

Come si può notare dalle ultime relazioni i parametri dipendono unicamente dalle intensità delle immagini e possono essere perciò calcolati a priori e inseriti successivamente in maniera opportuna nel problema della ricerca dello spostamento.

2.1.7 Modello della trasformazione affine

Risulta che fino a questo punto dei calcoli si è considerato per lo spostamento un modello di pura traslazione, ciò equivale ad accettare il fatto che una data finestra di pixel mantenga la stessa forma nel passare da una configurazione indeformata ad una deformata.

Nel caso generale accade però che un dato subset fra le diverse immagini può variare forma, con deformazioni angolari e di scala, per questo motivo risulta chiara la necessità di impostare un modello per lo spostamento che tenga in conto questi ulteriori fatti.

Si specifica che nella seguente trattazione si riporta la risoluzione del problema dello spostamento secondo il modello della trasformazione affine, nonostante nulla vietи di utilizzare modelli di spostamento più complessi.

Si consideri al tal proposito il problema di minimizzazione come riportato nella (2.1.45):

$$E = \underset{\Delta u, \Delta v}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I(x + \bar{u}, y + \bar{v}, t)}{\partial x} \Delta u + \frac{\partial I(x + \bar{u}, y + \bar{v}, t)}{\partial y} \Delta v - I_t \right)^2 \quad (2.1.45)$$

Impostando un sistema di riferimento locale (η, ξ) centrato sul pixel del quale si vuole trovare lo spostamento si esprime il modello dello spostamento dei pixel nel subset come:

$$\begin{aligned} \Delta u(\eta, \xi) &= \alpha_1 \eta + \beta_1 \xi + c_1 \\ \Delta v(\eta, \xi) &= \alpha_2 \eta + \beta_2 \xi + c_2 \end{aligned} \quad (2.1.72)$$

Indicando ora col termine I :

$$I = I(x + \bar{u}, y + \bar{v}, t) \quad (2.1.73)$$

Si può riscrivere la (2.1.45) nella seguente maniera:

$$E = \underset{\Delta u, \Delta v}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} (\alpha_1 \eta + \beta_1 \xi + c_1) + \frac{\partial I}{\partial y} (\alpha_2 \eta + \beta_2 \xi + c_2) - I_t \right)^2 \quad (2.1.74)$$

La minimizzazione della funzione errore si configura perciò come la ricerca ottimale di tutti i parametri della trasformazione affine, e si ricorda inoltre che risulta ora conglobabile anche il modello di compensazione della trasformazione radiometrica.

Tuttavia per evitare di complicare ulteriormente la trattazione non si fa riferimento esplicito ai parametri della trasformazione radiometrica, ma si suppone siano già stati calcolati e inseriti nel problema.

La minimizzazione della (2.1.74) porta alla definizione del seguente sistema di equazioni:

$$\left\{
\begin{aligned}
\frac{\partial E}{\partial \Delta u} &= \frac{\partial E}{\partial \Delta u} \frac{\partial \Delta u}{\partial \alpha_1} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial x} \cdot \frac{\partial (\Delta u)}{\partial \alpha_1} \right) = 0 \\
\frac{\partial E}{\partial \Delta u} &= \frac{\partial E}{\partial \Delta u} \frac{\partial \Delta u}{\partial \beta_1} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial x} \cdot \frac{\partial (\Delta u)}{\partial \beta_1} \right) = 0 \\
\frac{\partial E}{\partial \Delta u} &= \frac{\partial E}{\partial \Delta u} \frac{\partial \Delta u}{\partial c_1} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial x} \cdot \frac{\partial (\Delta u)}{\partial c_1} \right) = 0 \\
\frac{\partial E}{\partial \Delta v} &= \frac{\partial E}{\partial \Delta v} \frac{\partial \Delta v}{\partial \alpha_2} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial y} \cdot \frac{\partial (\Delta v)}{\partial \alpha_2} \right) = 0 \\
\frac{\partial E}{\partial \Delta v} &= \frac{\partial E}{\partial \Delta v} \frac{\partial \Delta v}{\partial \beta_2} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial y} \cdot \frac{\partial (\Delta v)}{\partial \beta_2} \right) = 0 \\
\frac{\partial E}{\partial \Delta v} &= \frac{\partial E}{\partial \Delta v} \frac{\partial \Delta v}{\partial c_2} = \sum_{i=1}^n \sum_{j=1}^m 2 \cdot \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial y} \cdot \frac{\partial (\Delta v)}{\partial c_2} \right) = 0
\end{aligned}
\right. \quad (2.1.75)$$

Risulta ora che le derivate dello spostamento rispetto ai parametri della trasformazione affine sono pari ad:

$$\begin{aligned}
\frac{\partial (\Delta u)}{\partial \alpha_1} &= \eta & \frac{\partial (\Delta u)}{\partial \alpha_2} &= \eta \\
\frac{\partial (\Delta u)}{\partial \beta_1} &= \zeta & \frac{\partial (\Delta u)}{\partial \beta_2} &= \zeta \\
\frac{\partial (\Delta u)}{\partial c_1} &= 1 & \frac{\partial (\Delta u)}{\partial c_2} &= 1
\end{aligned} \quad (2.1.76)$$

Sostituendo perciò le espressioni (2.1.76) in (2.1.75) si ottiene il seguente sistema:

$$\left\{
\begin{aligned}
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial x} \cdot \eta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial x} \cdot \zeta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial x} \cdot 1 \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial y} \cdot \eta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial y} \cdot \zeta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \Delta u + \frac{\partial I}{\partial y} \Delta v - I_t \right) \left(\frac{\partial I}{\partial y} \cdot 1 \right) = 0
\end{aligned}
\right. \quad (2.1.77)$$

Sviluppando ora le equazioni presenti nel sistema (2.1.77) si ottiene:

$$\left\{
\begin{aligned}
& \sum_{i=1}^n \sum_{j=1}^m \left[\left(\frac{\partial I}{\partial x} \right)^2 (\alpha_1 \eta + \beta_1 \zeta + c_1) \eta + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta (\alpha_2 \eta + \beta_2 \zeta + c_2) - \frac{\partial I}{\partial x} \eta I_t \right] = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left[\left(\frac{\partial I}{\partial x} \right)^2 (\alpha_1 \eta + \beta_1 \zeta + c_1) \zeta + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} (\alpha_2 \eta + \beta_2 \zeta + c_2) \zeta - I_t \frac{\partial I}{\partial x} \zeta \right] = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left[\left(\frac{\partial I}{\partial x} \right)^2 (\alpha_1 \eta + \beta_1 \zeta + c_1) + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} (\alpha_2 \eta + \beta_2 \zeta + c_2) - I_t \frac{\partial I}{\partial x} \right] = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left[\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} (\alpha_1 \eta + \beta_1 \zeta + c_1) \eta + \left(\frac{\partial I}{\partial y} \right)^2 (\alpha_2 \eta + \beta_2 \zeta + c_2) \eta - I_t \frac{\partial I}{\partial y} \eta \right] = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left[\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} (\alpha_1 \eta + \beta_1 \zeta + c_1) \zeta + \left(\frac{\partial I}{\partial y} \right)^2 (\alpha_2 \eta + \beta_2 \zeta + c_2) \zeta - I_t \frac{\partial I}{\partial y} \zeta \right] = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left[\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} (\alpha_1 \eta + \beta_1 \zeta + c_1) + \left(\frac{\partial I}{\partial y} \right)^2 (\alpha_2 \eta + \beta_2 \zeta + c_2) - I_t \frac{\partial I}{\partial y} \right] = 0
\end{aligned}
\right. \quad (2.1.78)$$

$$\begin{aligned}
& \sum_{i=1}^n \sum_{j=1}^m \left(\left(\frac{\partial I}{\partial x} \right)^2 \eta^2 \alpha_1 + \left(\frac{\partial I}{\partial x} \right)^2 \eta \zeta \beta_1 + \left(\frac{\partial I}{\partial x} \right)^2 \eta c_1 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta^2 \alpha_2 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \zeta \beta_2 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta c_2 - \frac{\partial I}{\partial x} \eta I_t \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\left(\frac{\partial I}{\partial x} \right)^2 \eta \zeta \alpha_1 + \left(\frac{\partial I}{\partial x} \right)^2 \zeta^2 \beta_1 + \left(\frac{\partial I}{\partial x} \right)^2 \zeta c_1 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta \eta \alpha_2 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta^2 \beta_2 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta c_2 - I_t \frac{\partial I}{\partial x} \zeta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\left(\frac{\partial I}{\partial x} \right)^2 \eta \alpha_1 + \left(\frac{\partial I}{\partial x} \right)^2 \zeta \beta_1 + \left(\frac{\partial I}{\partial x} \right)^2 c_1 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \alpha_2 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta \beta_2 + \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} c_2 - I_t \frac{\partial I}{\partial x} \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta^2 \alpha_1 + \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta \zeta \beta_1 + \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta c_1 + \left(\frac{\partial I}{\partial y} \right)^2 \eta^2 \alpha_2 + \left(\frac{\partial I}{\partial y} \right)^2 \eta \zeta \beta_2 + \left(\frac{\partial I}{\partial y} \right)^2 \eta c_2 - I_t \frac{\partial I}{\partial y} \eta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta \eta \alpha_1 + \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta^2 \beta_1 + \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta c_1 + \left(\frac{\partial I}{\partial y} \right)^2 \zeta \eta \alpha_2 + \left(\frac{\partial I}{\partial y} \right)^2 \zeta^2 \beta_2 + \left(\frac{\partial I}{\partial y} \right)^2 \zeta c_2 - I_t \frac{\partial I}{\partial y} \zeta \right) = 0 \\
& \sum_{i=1}^n \sum_{j=1}^m \left(\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta \alpha_1 + \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta \beta_1 + \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} c_1 + \left(\frac{\partial I}{\partial y} \right)^2 \eta \alpha_2 + \left(\frac{\partial I}{\partial y} \right)^2 \zeta \beta_2 + \left(\frac{\partial I}{\partial y} \right)^2 c_2 - I_t \frac{\partial I}{\partial y} \right) = 0
\end{aligned}$$

(2.1.79)

Risulta in definitiva quindi il seguente problema in termini matriciali:

$$\begin{bmatrix}
\sum \left(\frac{\partial I}{\partial x} \right)^2 \eta^2 & \sum \left(\frac{\partial I}{\partial x} \right)^2 \eta \zeta & \sum \left(\frac{\partial I}{\partial x} \right)^2 \eta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta^2 & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \\
\sum \left(\frac{\partial I}{\partial x} \right)^2 \eta \zeta & \sum \left(\frac{\partial I}{\partial x} \right)^2 \zeta^2 & \sum \left(\frac{\partial I}{\partial x} \right)^2 \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta^2 & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta \\
\sum \left(\frac{\partial I}{\partial x} \right)^2 \eta & \sum \left(\frac{\partial I}{\partial x} \right)^2 \zeta & \sum \left(\frac{\partial I}{\partial x} \right)^2 & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \\
\sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta^2 & \sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta \zeta & \sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta^2 & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta \zeta & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta \\
\sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta \zeta & \sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta^2 & \sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta \zeta & \sum \left(\frac{\partial I}{\partial y} \right)^2 \zeta^2 & \sum \left(\frac{\partial I}{\partial y} \right)^2 \zeta \\
\sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \eta & \sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \zeta & \sum \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta & \sum \left(\frac{\partial I}{\partial y} \right)^2 \zeta & \sum \left(\frac{\partial I}{\partial y} \right)^2
\end{bmatrix} \begin{bmatrix} \alpha_1 \\ \beta_1 \\ c_1 \\ \alpha_2 \\ \beta_2 \\ c_2 \end{bmatrix} = \begin{bmatrix} \sum \frac{\partial I}{\partial x} \eta I_t \\ \sum I_t \frac{\partial I}{\partial x} \zeta \\ \sum I_t \frac{\partial I}{\partial x} \\ \sum I_t \frac{\partial I}{\partial y} \eta \\ \sum I_t \frac{\partial I}{\partial y} \zeta \\ \sum I_t \frac{\partial I}{\partial y} \end{bmatrix}$$

(2.1.80)

la cui soluzione è presto calcolata invertendo la matrice dei coefficienti:

$$\begin{bmatrix} \alpha_1 \\ \beta_1 \\ c_1 \\ \alpha_2 \\ \beta_2 \\ c_2 \end{bmatrix} = \begin{bmatrix} \sum \left(\frac{\partial I}{\partial x} \right)^2 \eta^2 & \sum \left(\frac{\partial I}{\partial x} \right)^2 \eta \zeta & \sum \left(\frac{\partial I}{\partial x} \right)^2 \eta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta^2 & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \\ & \sum \left(\frac{\partial I}{\partial x} \right)^2 \zeta^2 & \sum \left(\frac{\partial I}{\partial x} \right)^2 \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta^2 & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta \\ & & \sum \left(\frac{\partial I}{\partial x} \right)^2 & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \eta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \zeta & \sum \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} \\ & & & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta^2 & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta \zeta & \sum \left(\frac{\partial I}{\partial y} \right)^2 \eta \\ & & & & \sum \left(\frac{\partial I}{\partial y} \right)^2 \zeta^2 & \sum \left(\frac{\partial I}{\partial y} \right)^2 \zeta \\ & & & & & \sum \left(\frac{\partial I}{\partial y} \right)^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum \frac{\partial I}{\partial x} \eta I_t \\ \sum I_t \frac{\partial I}{\partial x} \zeta \\ \sum I_t \frac{\partial I}{\partial x} \\ \sum I_t \frac{\partial I}{\partial y} \eta \\ \sum I_t \frac{\partial I}{\partial y} \zeta \\ \sum I_t \frac{\partial I}{\partial y} \end{bmatrix}$$

(2.1.81)

In generale risulta quindi che noti per l'iterazione k-1 i valori dei parametri e dello spostamento, risultano aggiornabili i valori da utilizzare nella successiva iterazione k. Il modello di trasformazione affine perciò permette come visto di quantificare non soltanto lo spostamento rigido delle regioni di pixel considerate ma anche di tenerne in conto le distorsioni angolari.

2.1.8 Calcolo degli spostamenti superiori al pixel

Si affronta ora il problema della determinazione di un metodo che permetta di calcolare spostamenti superiori al pixel.

La trattazione finora sviluppata infatti perde validità nel caso in cui si applichino gli algoritmi nella maniera presentata con lo scopo di quantificare grandi spostamenti: il problema è legato principalmente alle derivate che per ogni punto dell'immagine permettono di ottenere il gradiente d'intensità locale sulla base dell'informazione contenuta in pixels limitrofi al punto di volta in volta considerato.

Relativamente però alla trattazione legata all'implementazione iterativa si può notare come quest'ultima suggerisca un'astuzia che offre la possibilità di poter quantificare anche spostamenti maggiori al pixel.

Si consideri infatti la seguente figura 2.8 che riporta lo spostamento di un punto P sul piano immagine in istanti successivi di acquisizione:

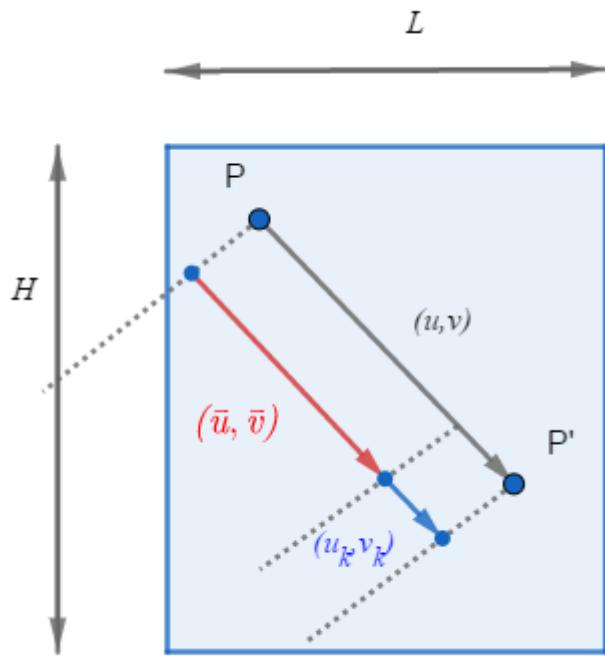


Fig. 2.8- Scomposizione dello spostamento di un punto P che passa alla posizione P' sul piano immagine di dimensioni $L \times H$, a seguito di uno spostamento (u, v) .

Come si può notare risulta ovvio che in generale lo spostamento (u, v) del punto P può essere scomposto nella somma di due contributi: nel nostro caso si considera il primo caratterizzato dal non essere limitato al range del singolo pixel (\bar{u}, \bar{v}) e il secondo caratterizzato dall'essere inferiore al pixel e risultante dalla k -esima iterazione (u_k, v_k) .

In caso sia noto quindi lo spostamento iniziale (\bar{u}, \bar{v}) corretto, emerge chiaramente la possibilità di inserire tale valore all'inizio del calcolo iterativo, al termine del quale risulterà determinato lo spostamento complessivo (u, v) .

Si può affermare quindi che lo spostamento complessivo ricercato è determinabile purchè sia noto lo spostamento iniziale \bar{u}, \bar{v} corretto.

Quanto detto poi risulta ovvio anche in relazione alla distribuzione dell'intensità dell'immagine: gli algoritmi sfruttando l'informazione in termini delle derivate trovano la soluzione di miglior accoppiamento fra i diversi subset considerati. In considerazione però al fatto che anche lungo una data riga della matrice dei pixel è possibile trovare molteplici valori di una data distribuzione d' intensità, l'applicazione degli algoritmi in

assenza della conoscenza dello spostamento iniziale porterebbe a risultati completamente errati ai fini della determinazione reale dello spostamento ricercato.

Esiste tuttavia un approccio semplificato alla risoluzione del problema: in pratica quello si fa è un'operazione di convoluzione volta a scalare le dimensioni delle immagini. Quanto affermato risulta chiaro in riferimento alla seguente figura 2.9:

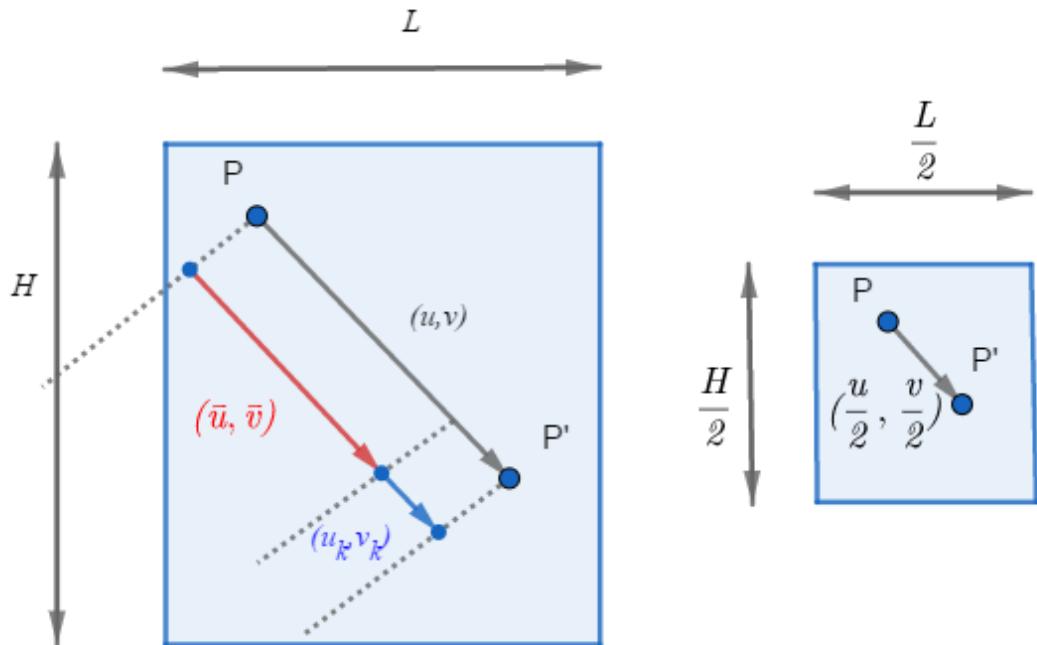


Fig. 2.9- Spostamento risultante a seguito del dimezzamento delle dimensioni dell'immagine originale.

In particolare si nota che se lo spostamento nell'immagine originale di dimensioni $L \times H$ è pari ad (u, v) , l'operazione di convoluzione che dimezza le dimensioni dell'immagine originale renderà lo spostamento nella nuova immagine pari alla metà $(u/2, v/2)$.

L'operazione di dimezzamento delle dimensioni effettuata per un numero n di volte scala il vettore spostamento di un fattore pari a 2^n , risulta in pratica che all' n -esimo livello di riduzione il valore delle componenti del vettore spostamento sarà pari ad:

$$\begin{aligned} u_n &= \frac{u}{2^n} \\ v_n &= \frac{v}{2^n} \end{aligned} \tag{2.1.82}$$

Nella pratica perciò si riducono le dimensioni dell'immagine iniziale fintanto che lo spostamento non rientra nel range del pixel, successivamente poiché risulta noto il valore del numero di volte che si è scalata la dimensione, risulta effettuabile una stima dello spostamento iniziale originale (u, v) moltiplicando il valore dello spostamento ottenuto sulla base delle immagini ridotte per n .

Si usa quindi tale valore come vettore spostamento (\bar{u}, \bar{v}) , e si imposta il calcolo iterativo dei valori (u_k, v_k) che permettono di raffinare il valore di primo tentativo.

Risulta opportuno precisare in fine che la trattazione vista presenta in maniera semplificata il concetto base sul quale si imposta l'implementazione piramidale del DIC2D, si precisa inoltre che esistono tuttavia altri approcci al problema che prevedono l'utilizzo di tecniche basate sul *template matching* o sull'*image warping*.

Per ogni ulteriore approfondimento si rimanda alla bibliografia [25] e [26].

2.1.9 Esempio d'applicazione ad un caso reale

Si riporta in conclusione come esempio d'applicazione della correlazione digitale d'immagine lo studio effettuato su un provino in materiale composito sottoposto a trazione.

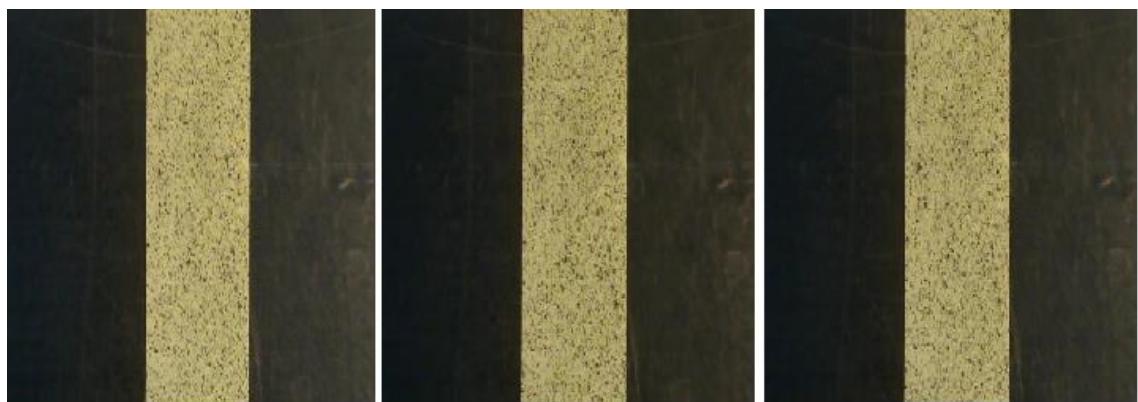


Fig. 2.11- Alcuni dei frames utilizzati per la correlazione.

Il software sviluppato in ambiente Matlab e allegato in appendice implementa il modello della trasformazione affine con compensazione radiometrica non iterativo: in pratica fornisce il risultato della prima iterazione del calcolo degli spostamenti.

Per mantenere in questo caso un certo grado di correlazione si è utilizzato un video che riporta l'intera prova di trazione fino a rottura del provino, quindi le immagini utilizzate nella correlazione sono state estratte dal video ad intervalli di tempo costanti pari a due secondi, per un tempo ragionevolmente anteriore alla fase di rottura.

La mappa complessiva degli spostamenti è stata quindi calcolata analizzando in cascata tutte le immagini ed integrando di volta in volta lo spostamento risultante dalla coppia di immagini considerate.

Successivamente, essendo note le dimensioni del provino (area trasversale pari a 30.28 mm² e distanza fra gli incastri pari a 150 mm), si è effettuato un confronto fra la curva sforzo deformazione costruita con i dati prodotti dalla macchina di prova e quella costruita utilizzando la correlazione digitale 2D con sub immagini di grandezza 25x25 pixels e distanza per il calcolo delle deformazioni pari a 440 pixels.

Si riporta in seguente figura 2.10 la mappa a tutto campo degli spostamenti verticali v :

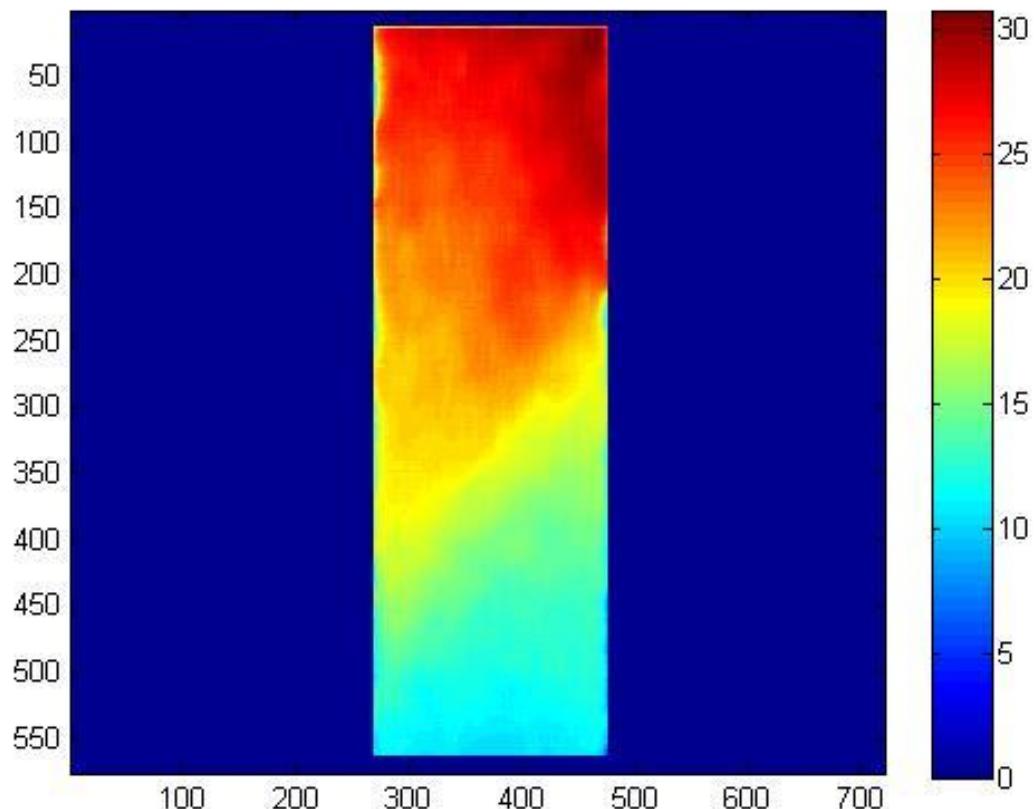


Fig. 2.11- Mappa degli spostamenti verticali v .

In seguente figura 2.11 si riporta invece la mappa degli spostamenti orizzontali u :

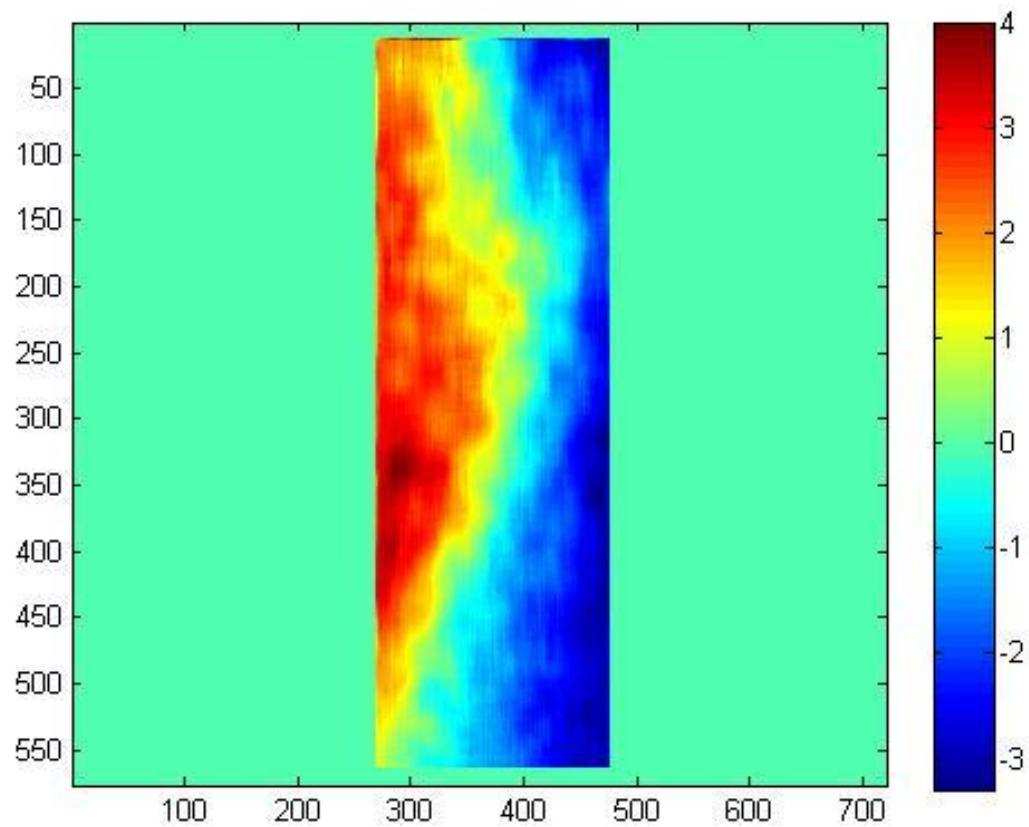


Fig. 2.12- Mappa degli spostamenti orizzontali u .

Infine nella seguente figura 2.12 si riportano in unico grafico le curve sforzo-deformazione costruite coi dati della macchina di prova e con il DIC2D:

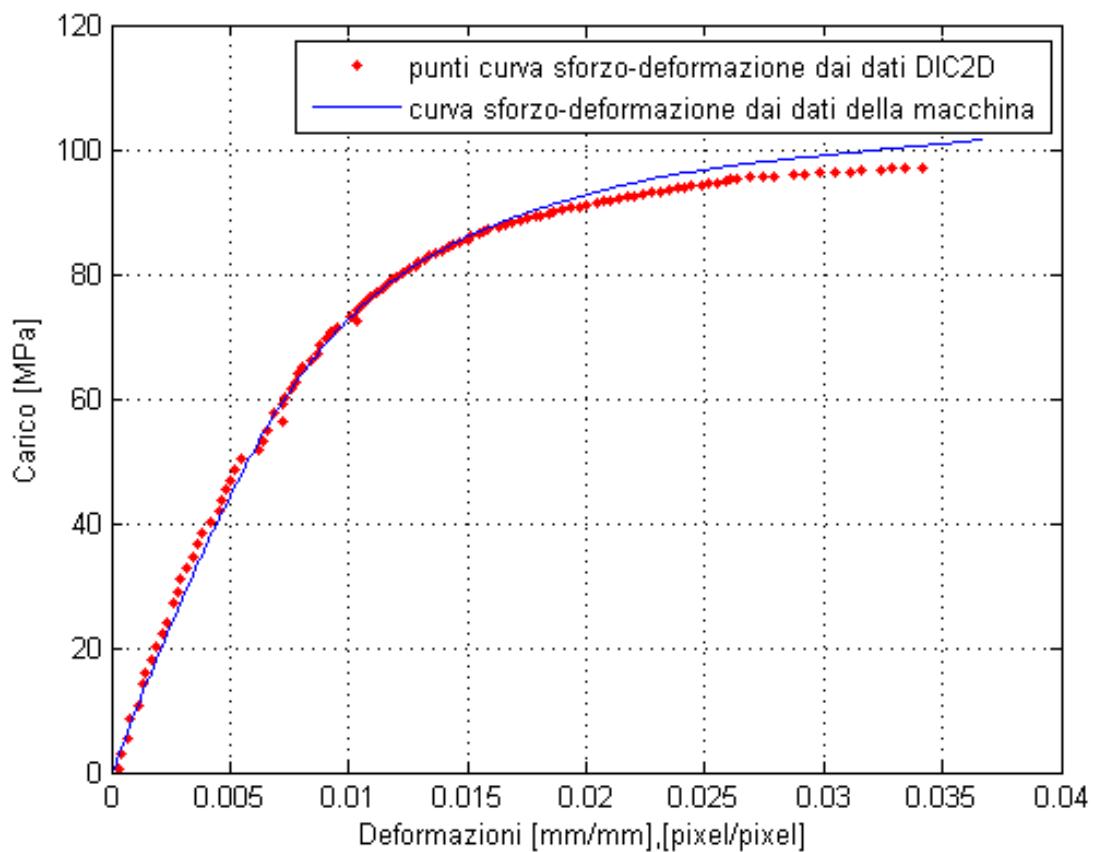


Fig. 2.13- curve sforzo-deformazione.

Successivamente il calcolo del modulo di Young del materiale dai dati del DIC2D e della macchina presenta i seguenti risultati:

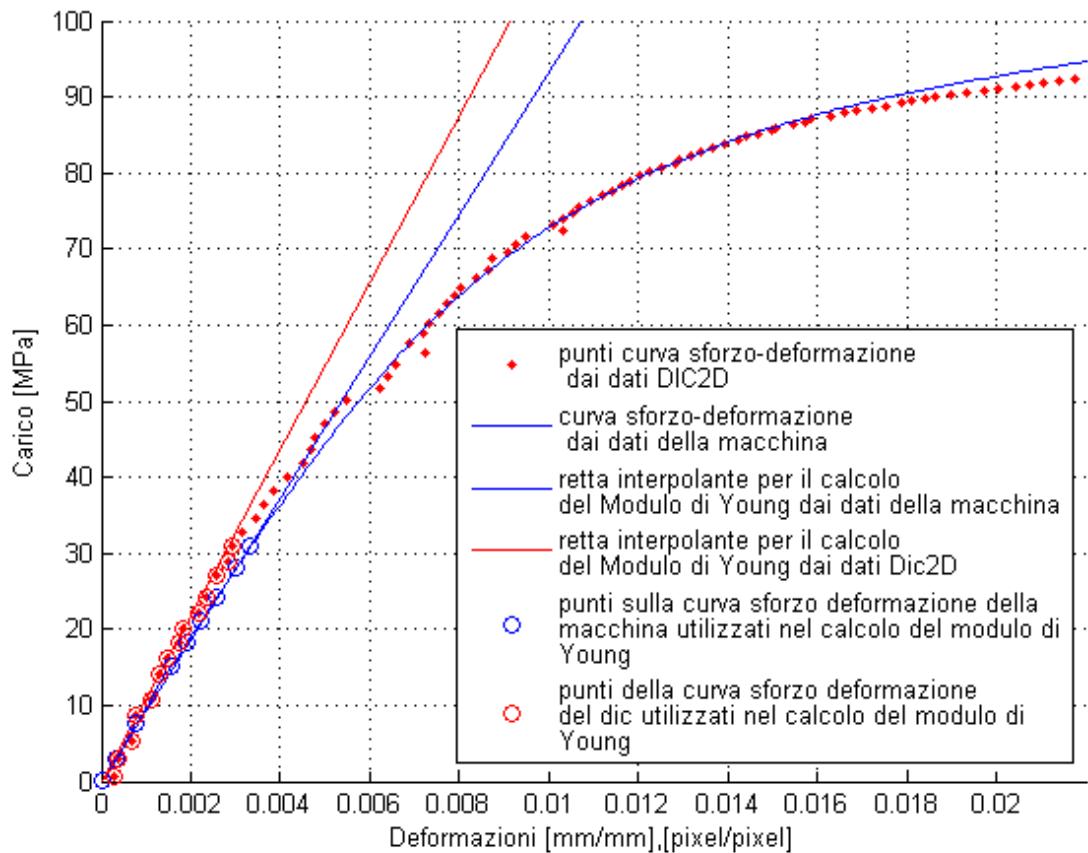


Fig. 2.14- Analisi del Modulo di Young.

Risultano i seguenti valori del modulo di Young calcolati dai dati della macchina di prova e del DIC2D pari a:

$$E_{\text{dati macchina}} = 10.927 \text{ [GPa]}$$

$$E_{\text{dati DIC2D}} = 9.339 \text{ [GPa]}$$

Con un errore relativo ai dati della macchina pari al 14.53%.

Come emerge dai risultati si può quindi ritenere la correlazione digitale d'immagine 2D come metodo valido per l'applicazione in campo ingegneristico.

In relazione al discostamento risultante rispetto ai valori della macchina, si precisa che quest'ultimo può essere sicuramente diminuito migliorando ulteriormente il software sviluppato per l'esempio, in aggiunta al fatto di non utilizzare immagini formato .jpeg che nonostante offrano ottime soluzioni a livello di compressione causano per quest'ultimo processo anche la perdita di informazione effettivamente elaborabile.

Capitolo 3

Modello alternativo per la correlazione digitale d'immagine tridimensionale.

Affrontata nei capitoli precedenti la trattazione relativa alla serie di processi che stanno alla base della correlazione digitale d'immagine tridimensionale secondo il modello di visione stereoscopica, si concentra ora l'attenzione sulla ricerca di metodologie alternative che permettano di ottenere la stessa tipologia di risultati.

Come si può comprendere l'applicazione del DIC3D sulla base del modello di visione stereoscopico presenta una molteplicità di problematiche legate sia all'onere economico per realizzazione del sistema d'acquisizione, sia alla mole di calcolo necessario ai fini dell'ottenimento dei risultati: anzitutto si necessita di un numero minimo di due fotocamere, bisogna quindi calibrare ognuna di esse e il sistema stereo rig complessivo, successivamente le operazioni di rettifica e calcolo delle disparità e l'applicazione della DIC2D permettono di ottenere i risultati cercati.

Nonostante risulti comunque un livello di propagazione degli errori tale da giustificare l'applicabilità in campo ingegneristico [31], risulta in generale un limite sugli angoli con i quali le due fotocamere possono osservare l'oggetto studiato: il metodo fallisce in caso di angoli di visione troppo grandi.

Analizzando perciò le diverse tecniche utilizzate nell'ambito della ricostruzione di forma e analisi delle sollecitazioni, risulta in particolare che un metodo misto a proiezione di frange con phase shifting e DIC2D può rappresentare una valida alternativa al modello stereoscopico del DIC3D.

L'idea è quella di utilizzare il phase shifting con proiezione di frangia per trovare la componente w dello spostamento (fuori dal piano), abbinato alla correlazione digitale d'immagine 2D per ottenere le componenti u, v (nel piano) in maniera da trovare lo spostamento complessivo.

In generale quanto detto tuttavia non risulta attuabile in maniera corretta se non sotto certe condizioni legate alle due metodologie che si vogliono utilizzare, in particolare si hanno i seguenti limiti:

- La correlazione digitale d'immagine 2D utilizzata per calcolare lo spostamento esatto dei punti richiede che gli spostamenti siano confinati a stare nel piano: ogni spostamento fuori dal piano genera spostamenti apparenti nei risultati ottenuti dal DIC2D, tale problema è legato infatti alla prospettiva.
- L'analisi della forma col metodo classico a proiezione di frangia richiede l'utilizzo di un fascio di proiezione collimato.

Risulterebbe quindi a prima vista che il metodo pensato non potrebbe trovare effettiva applicazione: il problema principale è rappresentato dalla correlazione digitale d'immagine 2D che sarebbe utilizzata in questo caso per quantificare spostamenti di un oggetto che si muove nello spazio.

Tuttavia si può notare che in caso si conosca esattamente la componente w dello spostamento il problema degli spostamenti apparenti u_{app}, v_{app} prodotti dal DIC2D può essere compensato a posteriori.

L'idea in questo caso è quella di esprimere lo spostamento apparente u_{app}, v_{app} in funzione dello spostamento esatto w .

Si consideri in tal senso di fotografare a diverse distanze d_i un oggetto piano esente da carichi esterni, con asse ottico della fotocamera perfettamente perpendicolare al piano oggetto: si ha in questo caso che la correlazione fra l' i -esima immagine scattata alla distanza d_i e l'immagine $i+1$ scattata alla distanza d_{i+1} produce soltanto spostamenti apparenti.

Note quindi le distanze d_i e d_{i+1} risulta calcolabile anche lo spostamento esatto imposto $w_i = d_i - d_{i+1}$, e risulta quindi trovato un punto della funzione che esprime il legame fra spostamento apparente misurato dal DIC2D e spostamento w esatto.

Ripetendo il procedimento per diverse distanze si ottiene in definitiva la relazione cercata.

A livello teorico risulta quindi applicabile in maniera esatta il metodo misto a proiezione di frangia con DIC2D: noto infatti lo spostamento w esatto dei punti sulla superficie dell'oggetto studiato basta eliminare per ognuno di essi il contributo u_{app}, v_{app} nei risultati

della correlazione digitale, ottenendo in definitiva il valore esatto dello spostamento tridimensionale cercato.

A livello pratico tuttavia si nota che l'applicazione classica del metodo a proiezione di frangia per il calcolo delle w esatte richiede l'utilizzo di un fascio di proiezione collimato e fotocamera con obiettivo telecentrico: questo fatto, nonostante non rappresenti un vero e proprio problema in quanto basterebbe utilizzare entrambi i dispositivi con obiettivi telecentrici, ha spinto alla ricerca di soluzioni pratiche che permettano di superare tale limite e giustificare l'applicazione del metodo anche in assenza dei requisiti classici.

Il presente capitolo perciò si concentra sullo sviluppo del metodo discusso, in particolare nella prima parte si discute sul metodo del phase shifting a proiezione di frangia, nella seconda parte si analizza il processo di proiezione dell'immagine e infine si riportano i risultati ottenuti da un esempio di applicazione.

3.1 Tecnica del phase shifting a proiezione di frange

In termini generali si può introdurre la tecnica del phase shifting a proiezione di frange come metodo che permette di determinare la forma di un dato oggetto in funzione della misura della variazione di fase fra segnali periodici simili applicati in sequenza sulla superficie di riferimento dell'oggetto (per esempio il suo piano portante) e quella dell'oggetto stesso.

Relativamente alla seguente figura 3.1 si supponga di proiettare su un piano ad una certa angolazione tramite un fascio collimato un segnale periodico di una generica forma, ma di periodo 2π e passo p costante espresso in qualsivoglia unità metrica.

Come ovvio aspettarsi tale proiezione genera sul piano un segnale similmente periodico (di periodo 2π) con la differenza sul passo che assume il nuovo valore p_x .

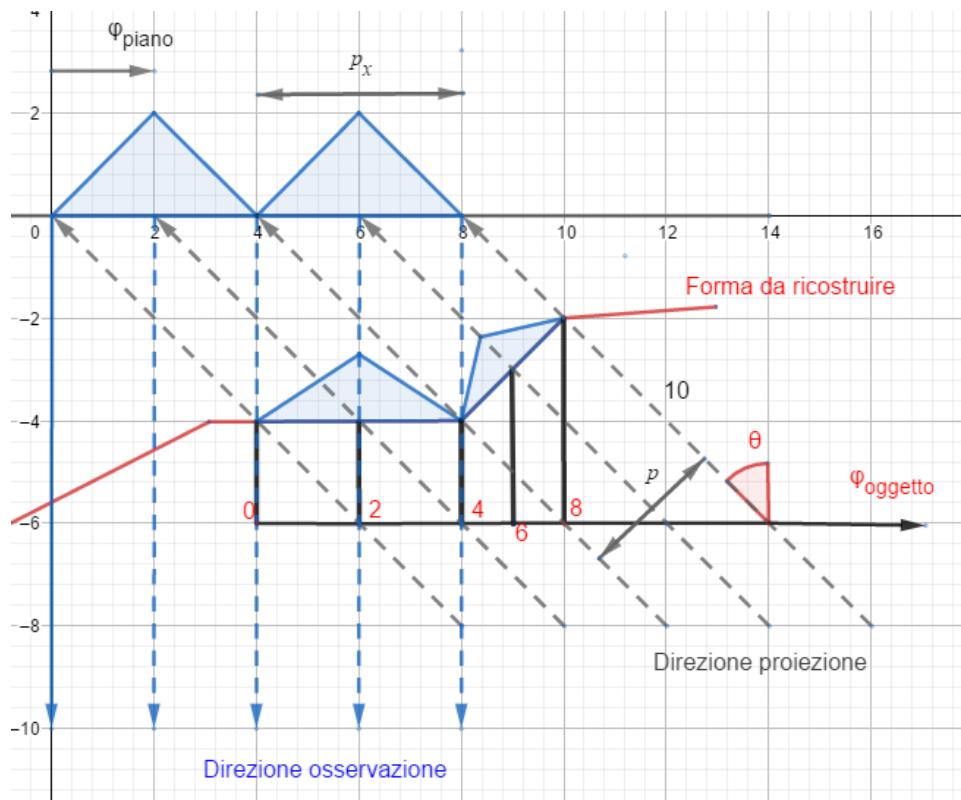


Fig. 3.1- Rappresentazione della proiezione di un segnale periodico di forma triangolare mediante fasci collimato e direzione d'osservazione perpendicolare al piano.

Come si può osservare dalla seguente figura 3.2 la relazione che lega la variazione del passo p a p_x è direttamente funzione dell'angolo di proiezione θ :

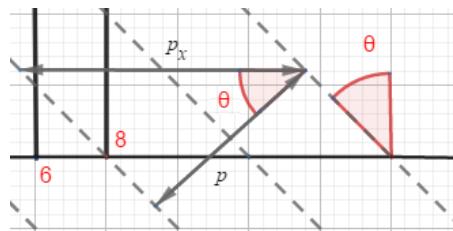


Fig. 3.2- Analisi del legame geometrico fra i passi p e p_x .

Risulta in particolare che:

$$P_x = \frac{P}{\cos(\theta)} \quad (3.1)$$

Sempre relativamente alla figura 3.1 si consideri ora di interporre fra il piano e il fascio di proiezione l'oggetto del quale si vuole ricostruire la forma: la mappatura del segnale proiettato sulla superficie dell'oggetto genera in questo caso un segnale sempre caratterizzato da stessa ampiezza, stesso periodo 2π ma passo variabile in funzione della forma stessa.

La conservazione della periodicità del segnale permette quindi poterne sfruttare il contenuto in fase ϕ : in pratica risulta possibile per ogni punto, sia della superficie del piano che quella dell'oggetto riportanti la mappatura del segnale proiettato, associare un valore di fase contenuto nel periodo del segnale proiettato.

L'integrazione della fase lungo le superfici del piano e dell'oggetto permette di ottenere importanti risultati ai fini della ricostruzione della superficie dell'oggetto studiato.

Relativamente alla seguente figura 3.3 si consideri infatti un punto A_1 sul piano:

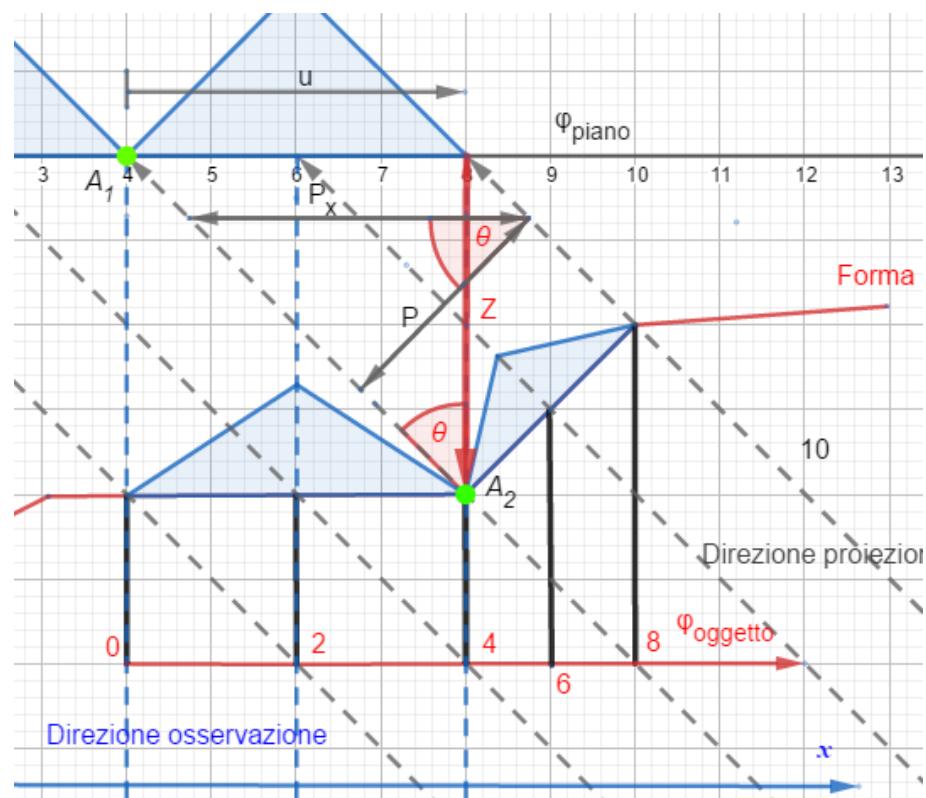


Fig. 3.3- Rappresentazione dello spostamento misurato secondo la direzione di osservazione.

l'interposizione della forma fra il piano e la sorgente dei fasci proiettati ha l'effetto per un osservatore posto lungo la direzione di osservazione, di veder spostato il punto A_1 in una posizione A_2 di una quantità pari ad u .

Esprimendo ora i gli andamenti della fase sul piano e sulla forma rispetto ad un riferimento comune perpendicolare alla direzione di osservazione, si può notare che la differenza del valore integrale della fase al punto di proiezione di A_2 sul piano iniziale e quello del valore integrale sempre al punto A_2 sulla superficie da ricostruire, risulta pari al valore dello spostamento u :

$$u = \varphi_{\text{piano}}(A_2) - \varphi_{\text{oggetto}}(A_2) \quad (3.2)$$

Risulta altresì che tale spostamento è legato in maniera diretta al valore della quota z del punto A_2 rispetto al piano come espresso dalla seguente relazione:

$$u = z \cdot \operatorname{tg}(\theta) \quad (3.3)$$

La quale permette di concludere che:

$$z_{A_2} = \frac{\varphi_{\text{piano}}(A_2) - \varphi_{\text{oggetto}}(A_2)}{\operatorname{tg}(\theta)} \quad (3.4)$$

Alla luce della trattazione riportata, considerando anche il fatto che l'osservatore è rappresentato da una fotocamera con asse ottico parallelo alla direzione di osservazione, si può concludere che il problema della ricostruzione della forma si concentra in prima analisi sulla ricerca della fase per i due frame riportanti rispettivamente l'immagine del piano e quella dell'oggetto. Successivamente ottenuta la fase integrale la forma può essere ricostruita punto per punto secondo la (3.4).

Va precisato che nonostante l'esempio di figura 3.1 riporta una configurazione con asse ottico della fotocamera posto in direzione perpendicolare al piano, la trattazione risulta valida anche ad angoli di osservazione diversi.

Si consideri per esempio la seguente figura 3.4:

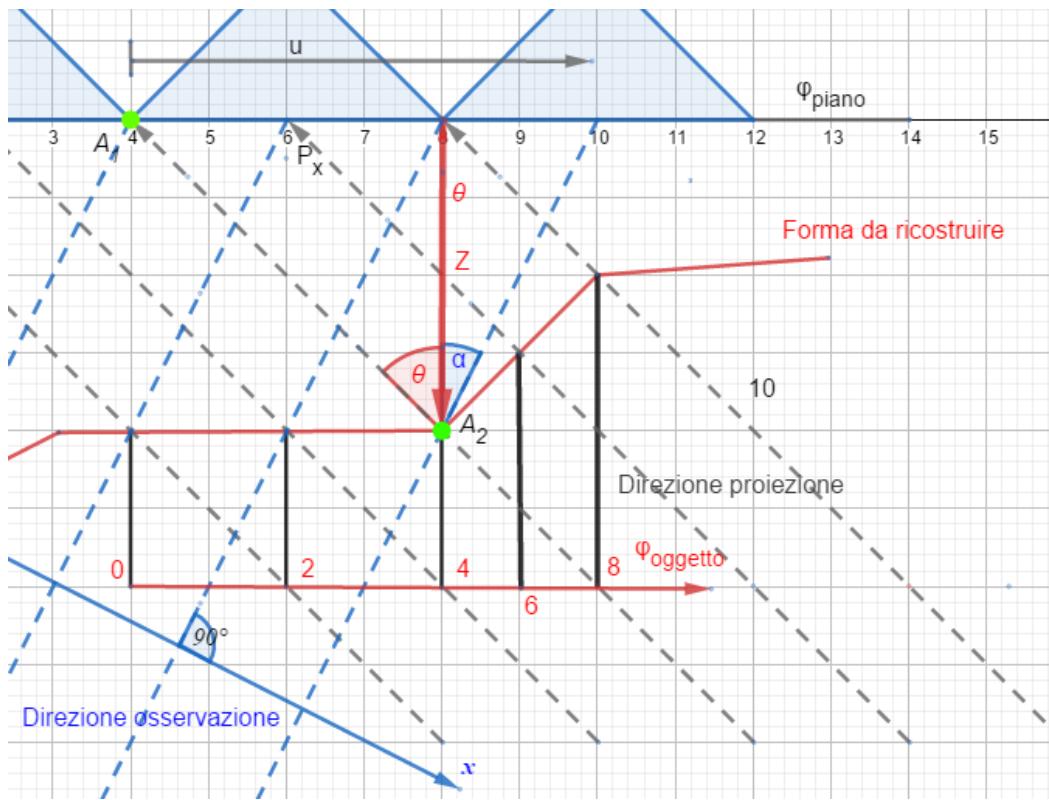


Fig. 3.4- Rappresentazione della proiezione di un fascio collimato e direzione d'osservazione qualsiasi.

Anche in questo caso si ha che lo spostamento u sarà pari ad:

$$u = \varphi_{\text{piano}}(A_2) - \varphi_{\text{oggetto}}(A_2) \quad (3.2)$$

Mentre il valore della quota z risulta ricavabile come:

$$u = z \cdot [\tan(\theta) + \tan(\alpha)] \quad (3.5)$$

Con θ angolo di proiezione ed α quello di osservazione.

Si ottiene in definitiva che:

$$z_{A_2} = \frac{\varphi_{\text{piano}}(A_2) - \varphi_{\text{oggetto}}(A_2)}{[\tan(\theta) + \tan(\alpha)]} \quad (3.6)$$

Volendo ora esprimere lo spostamento in termini del passo P_x si ottiene la funzione di modulazione:

$$\begin{aligned}
\Psi(x) = \frac{u(x)}{P_x} &= \frac{z(x) \cdot [\tan(\theta) + \tan(\alpha)]}{\frac{P}{\cos(\theta)}} = \frac{z(x) \cdot \left[\frac{\sin(\theta)}{\cos(\theta)} + \frac{\sin(\alpha)}{\cos(\alpha)} \right]}{\frac{P}{\cos(\theta)}} = \\
&= \frac{\cos(\theta)}{P} \cdot z(x) \cdot \left[\frac{\sin(\theta)}{\cos(\theta)} + \frac{\sin(\alpha)}{\cos(\alpha)} \right] = \frac{z(x)}{P} \cdot \left[\frac{\sin(\theta + \alpha)}{\cos(\alpha)} \right]
\end{aligned} \tag{3.7}$$

Dalla (3.7) risulta che la quota z è esprimibile come:

$$z(x) = \frac{\Psi(x) \cdot P \cdot \cos(\alpha)}{\sin(\theta + \alpha)} \tag{3.8}$$

Quanto effettuato risulta fondamentale in virtù delle caratteristiche delle griglie di proiezione effettivamente utilizzate di solito, in particolare risulta che esse possono essere modulate in fase, ed inoltre la loro distribuzione dell'intensità può essere espresso tramite una funzione sinusoidale.

Si avrà quindi per l'immagine del piano:

$$I_{\text{piano}}(x,y) = I_{\text{o}}(x,y) + A_{\text{piano}}(x,y) \cdot \cos(\varphi_{\text{piano}}(x,y)) \tag{3.9}$$

Mentre per l'immagine della forma:

$$I_{\text{forma}}(x,y) = I_{\text{o}}(x,y) + A_{\text{forma}}(x,y) \cdot \cos(\varphi_{\text{forma}}(x,y)) \tag{3.10}$$

Con i due valori di fase legati dallo sfasamento ψ :

$$\varphi_{\text{forma}}(x,y) = \varphi_{\text{piano}}(x,y) + \Psi(x,y) \tag{3.11}$$

ed $A(x,y)$ pari all'ampiezza della funzione sinusoidale.

Si comprende quindi chiaramente come una volta trovati i valori di fase per le due immagini la forma sia facilmente ricostruibile per semplice sottrazione puntuale dei valori integrali delle fasi.

L'applicazione del metodo presentato però è sottoposto a dei limiti sulle caratteristiche dei fasci di proiezione e di osservazione in assenza dei quali l'intero processo porterebbe a risultati completamente errati in caso di applicazione diretta della formula 3.8.

Tali limiti sono principalmente due:

- Il fascio di proiezione delle frange deve essere perfettamente collimato, tale che l'angolo di proiezione si mantenga costante.
- L'obiettivo della fotocamera utilizzata nell'acquisizione delle immagini deve essere telecentrico tale che non cambi l'angolo di osservazione punto per punto (fenomeno legato alla trasformazione di proiezione centrale).

A tal proposito se si osserva la seguente figura si può notare la sostanziale differenza fra i due casi:

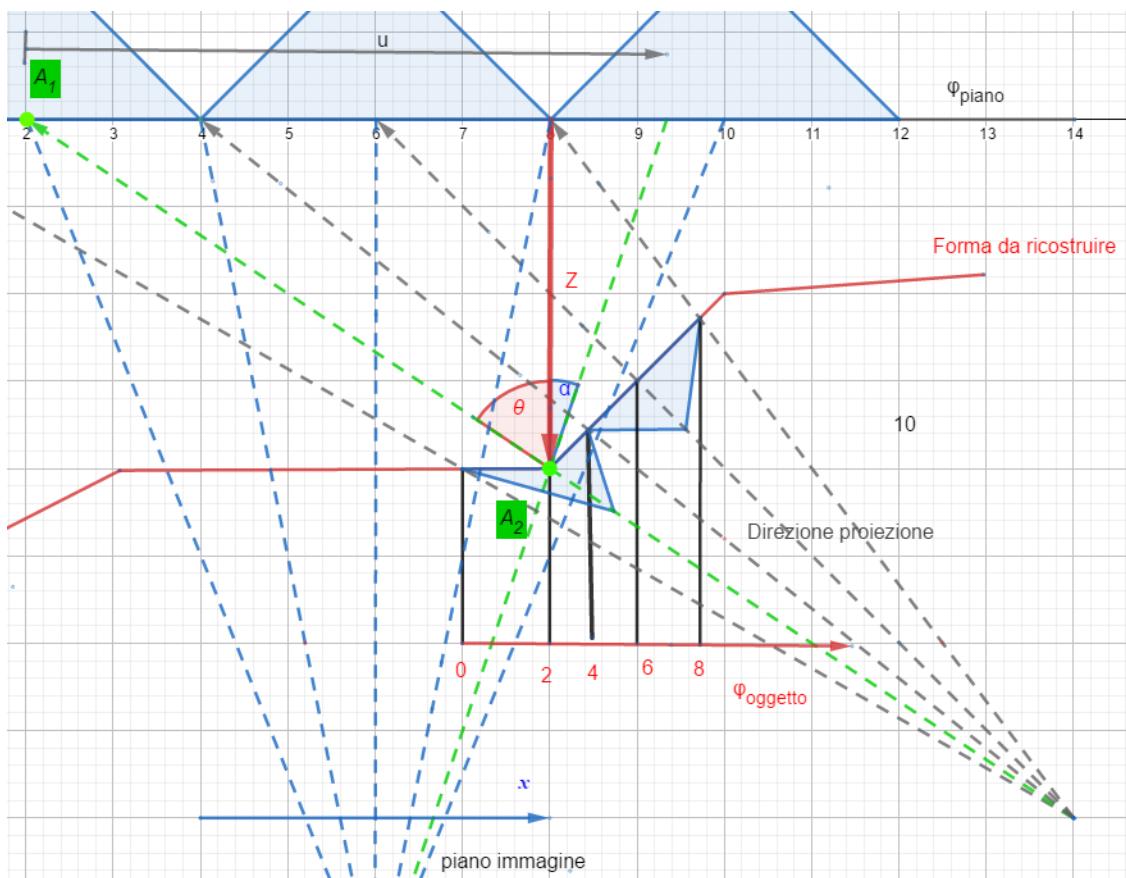


Fig. 3.5- Rappresentazione del fenomeno della proiezione e dell'osservazione secondo la trasformazione di proiezione centrale.

In particolare risulta sempre verificata la relazione (3.2), mentre per quanto riguarda l'effettiva quota dei punti (relazione 3.8) si ha che il valore esatto della quota z punto per punto è comunque calcolabile a condizione che siano noti gli angoli α di osservazione ed θ di proiezione punto per punto.

I valori di α sono calcolabili in maniera diretta una volta effettuata la calibrazione della fotocamera.

Quest'ultima come visto permette infatti di quantificare sia la distanza focale che la posizione del punto principale sul sensore della macchina fotografica.

Relativamente alla seguente figura 3.6 si consideri infatti che nota la distanza focale, le dimensioni in pixel dell'immagine o del sensore (esistendo fra i due semplicemente un fattore di scala), e la posizione del punto principale (che si ricorda essere l'intersezione dell'asse ottico sul piano dell'immagine), risulta calcolabile per ogni posizione sulla matrice dei pixels dell'immagine l'angolo α di osservazione:

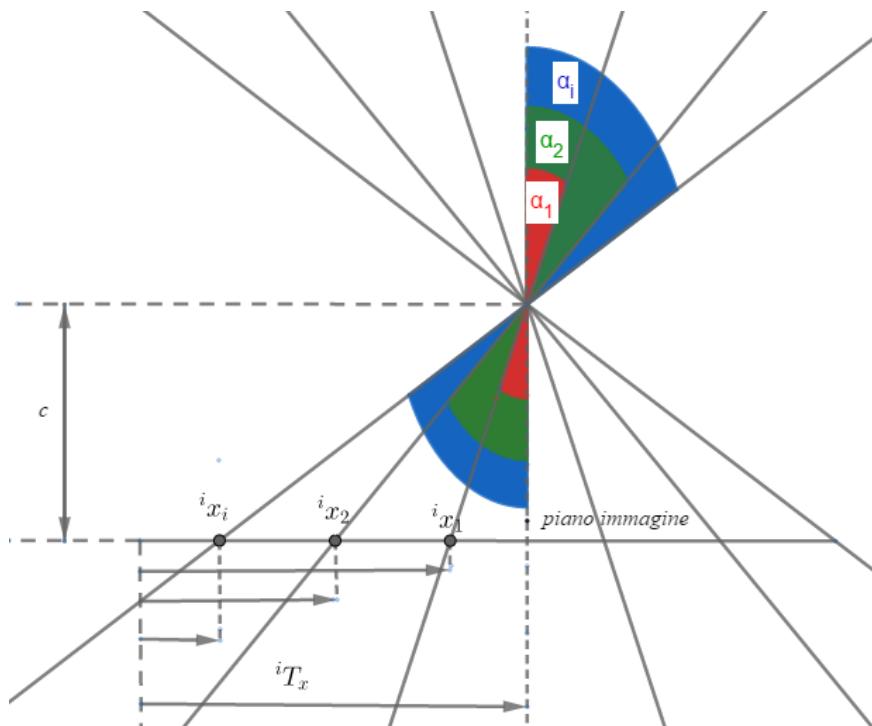


Fig. 3.6- Rappresentazione degli angoli di osservazione nota la distanza focale c e il punto principale.

Dato infatti il vettore posizione del punto principale sull'immagine iT_x e la distanza focale c , risulta per ogni posizione i -esima del pixel sulla matrice del piano immagine che:

$${}^i\mathbf{x}_i - {}^i\mathbf{T}_x = c \cdot \mathbf{tg}(\alpha_i) \quad (3.12)$$

E quindi:

$$\alpha_i = \text{arctg} \left(\frac{{}^i\mathbf{x}_i - {}^i\mathbf{T}_x}{c} \right) \quad (3.13)$$

Trovato il modo di determinare gli angoli di osservazione per ogni pixel sull'immagine si concentra ora l'attenzione sul problema della ricerca della fase delle immagini e sugli angoli di proiezione per i diversi punti.

3.1.1 Phase shifting per il recupero della fase

Come visto nel paragrafo precedente l'intero processo della ricostruzione della forma si basa sul recupero della fase per le due immagini riportanti il piano e l'oggetto studiato.

A tal proposito si scriva la distribuzione dell'intensità sull'immagine riportante l'oggetto o il piano soggetti alla proiezione della generica griglia di periodo fissato, come:

$$I(x,y) = I_o(x,y) + A(x,y)\cos(\varphi(x,y)) \quad (3.1.1)$$

Il significato dei vari termini è il seguente:

- $I(x,y)$: livello intensità al generico pixel in posizione x,y
- $I_o(x,y)$: livello intensità di fondo dovuto alla luce ambiente
- $A(x,y)$: ampiezza massima dell'intensità che per immagini in scala di grigio ad 8 bit corrisponde ad 255, ed in generale dato il numero “n” di bit per canale dell'immagine ha valore 2^n-1 .
- $\cos(\varphi(x,y))$: esprime l'evoluzione dell'intensità luminosa, col termine $\varphi(x,y)$ pari alla fase.

Quanto detto permette altresì di concludere che in caso si proietti la stessa griglia sfasata di uno sfasamento imposto δ , tale sfasamento si potrà osservare sull'intera immagine finale, e cioè risulterà in questo caso che la distribuzione d'intensità sarà della forma:

$$I(x,y) = I_o(x,y) + A(x,y)\cos(\varphi(x,y) + \delta) \quad (3.1.2)$$

Elaborando ulteriormente l'espressione ottenuta si può scrivere:

$$I(x,y) = I_o(x,y) + A(x,y)\cos(\varphi(x,y))\cos(\delta) - A(x,y)\sin(\varphi(x,y))\sin(\delta) \quad (3.1.3)$$

Indicando ora coi termini:

$$\begin{aligned} X_o &= I_o(x,y) \\ X_1 &= A(x,y)\cos(\varphi(x,y)) \\ X_2 &= -A(x,y)\sin(\varphi(x,y)) \end{aligned} \quad (3.1.4)$$

Si può riscrivere l'equazione (3.1.3) come:

$$I(x,y) = X_o + X_1 \cos(\delta) + X_2 \sin(\delta) \quad (3.1.5)$$

Il recupero della fase viene perciò posto in termini della ottimizzazione della seguente funzione obiettivo:

$$\psi = \operatorname{argmin} \sum_{i=1}^n [^i I(x,y) - I(x,y)]^2 \quad (3.1.6)$$

In pratica data una immagine ${}^i I(x,y)$ di un qualsivoglia numero di pixels si cerca per ognuno di essi i valori ottimali di X_o, X_1, X_2 che minimizzino la funzione obiettivo ψ .

Emerge quindi chiaramente che il problema essendo posto come la ricerca dei valori ottimali di tre variabili per ogni pixel non è risolvibile a meno che non si disponga dell'informazione su come varia l'intensità a seguito dell'applicazione di tre distinti segnali.

Ciò equivale alla necessità di acquisire almeno $n=3$ immagini nelle quali si sono applicati tre distinti valori di sfasamento δ nella proiezione della griglia originale.

Rielaborando l'espressione della funzione obiettivo si ha:

$$\psi = \operatorname{argmin}_{X_o, X_1, X_2} \sum_{i=1}^n [{}^i I(x,y) - (X_o + X_1 \cos(\delta) + X_2 \sin(\delta))]^2 \quad (3.1.7)$$

la cui minimizzazione porta al seguente sistema di equazioni:

$$\begin{cases} \frac{\partial \Psi}{\partial X_0} = \sum_{i=1}^n 2 \left[i I(x,y) - (X_0 + X_1 \cos(\delta) + X_2 \sin(\delta)) \right] \cdot (-1) = 0 \\ \frac{\partial \Psi}{\partial X_1} = \sum_{i=1}^n 2 \left[i I(x,y) - (X_0 + X_1 \cos(\delta) + X_2 \sin(\delta)) \right] \cdot (-\cos(\delta)) = 0 \\ \frac{\partial \Psi}{\partial X_2} = \sum_{i=1}^n 2 \left[i I(x,y) - (X_0 + X_1 \cos(\delta) + X_2 \sin(\delta)) \right] \cdot (-\sin(\delta)) = 0 \end{cases} \quad (3.1.8)$$

che in forma matriciale equivale a:

$$\begin{bmatrix} \sum_{i=1}^n 1 & \sum_{i=1}^n \cos(\delta) & \sum_{i=1}^n \sin(\delta) \\ \sum_{i=1}^n \cos(\delta) & \sum_{i=1}^n \cos^2(\delta) & \sum_{i=1}^n \sin(\delta) \cos(\delta) \\ \sum_{i=1}^n \sin(\delta) & \sum_{i=1}^n \sin(\delta) \cos(\delta) & \sum_{i=1}^n \sin^2(\delta) \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n i I(x,y) \\ \sum_{i=1}^n i I(x,y) \cos(\delta) \\ \sum_{i=1}^n i I(x,y) \sin(\delta) \end{bmatrix} \quad (3.1.9)$$

ed ha soluzione pari a:

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n 1 & \sum_{i=1}^n \cos(\delta) & \sum_{i=1}^n \sin(\delta) \\ \sum_{i=1}^n \cos(\delta) & \sum_{i=1}^n \cos^2(\delta) & \sum_{i=1}^n \sin(\delta) \cos(\delta) \\ \sum_{i=1}^n \sin(\delta) & \sum_{i=1}^n \sin(\delta) \cos(\delta) & \sum_{i=1}^n \sin^2(\delta) \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^n i I(x,y) \\ \sum_{i=1}^n i I(x,y) \cos(\delta) \\ \sum_{i=1}^n i I(x,y) \sin(\delta) \end{bmatrix} \quad (3.1.10)$$

Ottenuti perciò i valori del vettore delle incognite la fase risulta facilmente calcolabile come di seguito:

$$\frac{-X_2}{X_1} = \frac{A(x,y) \sin(\varphi(x,y))}{A(x,y) \cos(\varphi(x,y))} = \operatorname{tg}(\varphi(x,y)) \quad (3.1.11)$$

$$\varphi(x,y) = \operatorname{arctg}\left(\frac{-X_2}{X_1}\right) \quad (3.1.12)$$

La relazione (3.1.12) permette in definitiva di ottenere il valore della fase ricercato.

Bisogna precisare tuttavia che il metodo illustrato non è l'unico.

Si considerino per esempio 4 immagini ottenute proiettando sul piano o sull'oggetto una griglia sfasata rispettivamente degli angoli $0, \pi/2, \pi, (3/2)\pi$, le risultanti distribuzioni di intensità possono essere scritte come:

$$\begin{aligned} {}^1I(x,y) &= I_o(x,y) + A(x,y)\cos(\varphi(x,y)) \\ {}^2I(x,y) &= I_o(x,y) + A(x,y)\cos\left(\varphi(x,y) + \frac{\pi}{2}\right) = I_o(x,y) - A(x,y)\sin(\varphi(x,y)) \\ {}^3I(x,y) &= I_o(x,y) + A(x,y)\cos(\varphi(x,y) + \pi) = I_o(x,y) - A(x,y)\cos(\varphi(x,y)) \\ {}^4I(x,y) &= I_o(x,y) + A(x,y)\cos\left(\varphi(x,y) + \frac{3\pi}{2}\right) = I_o(x,y) + A(x,y)\sin(\varphi(x,y)) \end{aligned} \quad (3.1.13)$$

Dalle quali si ha considerando la seguente espressione:

$$\frac{{}^4I(x,y) - {}^2I(x,y)}{{}^1I(x,y) - {}^3I(x,y)} = \frac{2 \cdot A(x,y)\sin(\varphi(x,y))}{2 \cdot A(x,y)\cos(\varphi(x,y))} = \operatorname{tg}(\varphi(x,y)) \quad (3.1.14)$$

che la fase risulta calcolabile come:

$$\varphi(x,y) = \operatorname{arctg} \left(\frac{{}^4I(x,y) - {}^2I(x,y)}{{}^1I(x,y) - {}^3I(x,y)} \right) \quad (3.1.15)$$

Si nota in conclusione che esistono in letteratura diversi algoritmi di phase shifting (a tre, quattro, cinque immagini [21]) per il recupero della fase, tutti accumunati dal ricavare la fase $\varphi(x,y)$ introducendo di volta in volta una serie di sfasamenti noti in maniera che l'effetto complessivo dello sfasamento imposto possa essere eliminato a posteriori mediante semplificazioni di tipo trigonometrico.

Si noti bene che nonostante la (3.1.15) permetta di calcolare la fase punto per punto quest'ultima non può essere utilizzata in maniera diretta per calcolare la forma dell'oggetto.

La funzione arcotangente è infatti periodica e risulta continua solo nell'intervallo $[-\pi, \pi]$ motivo per il quale ogni qualvolta si raggiunge uno dei valori limite dell'intervallo si verifica un salto di 2π .

L'applicazione della (3.1.15) quindi restituisce quella che viene indicata come fase frazionaria, data la quale la fase intera può essere ricostruita mediante il procedimento noto come phase unwrapping.

Poiché la trattazione delle diverse metodologie di phase unwrapping esistenti va oltre l'obiettivo dell'elaborato si indirizza il lettore interessato alla bibliografia attinente [21],[23],[24].

3.1.2 Modello geometrico per la stima degli angoli di proiezione.

Relativamente al problema della definizione degli angoli di proiezione, senza conoscere i quali non risulta applicabile la metodologia proposta, si concentra l'attenzione in primo luogo alla definizione di un modello per la proiezione.

Inizialmente in via semplificativa trascurando le effettive dimensioni geometriche del pixel e la presenza di disturbo esterno dovuto all'ambiente, ed ipotizzando una sorgente luminosa di tipo puntuale si ha che risultano calcolabili per una data configurazione tutti gli angoli di proiezione.

Infatti accettando per la propagazione della luce un modello di tipo lineare in direzione radiale rispetto alla sorgente luminosa, risulta calcolabile sia l'angolo che lo spessore competente alla singola frangia mediante l'analisi dei triangoli costruiti sulle direzioni della propagazione luminosa.

Come ovvio aspettarsi risulta che queste quantità sono funzione non solo all'angolazione relativa fra piano di proiezione e sorgente ma anche alla distanza fra i due elementi calcolata sulla congiungente il punto focale della sorgente e il baricentro geometrico del piano di proiezione.

Punto di partenza dell'intera trattazione consiste nel considerare il caso più semplice con angolazione di proiezione nulla: il piano sul quale viene proiettato il pattern risulta parallelo al piano del sensore del proiettore contenente i pixels da proiettare.

Si riporta in seguente figura 3.7 una schematizzazione della configurazione:

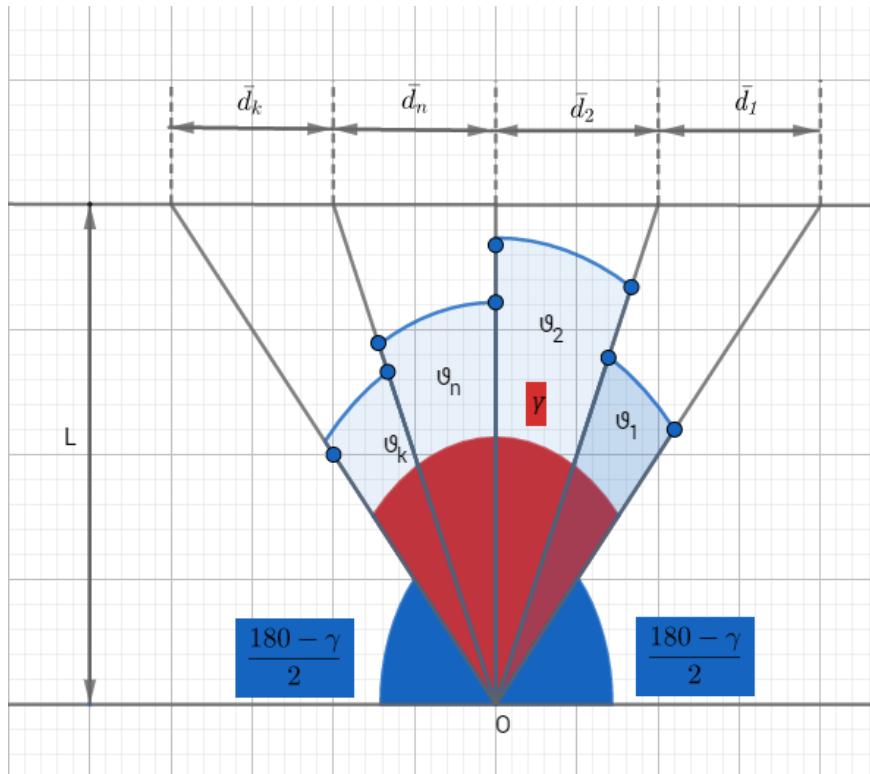


Fig. 3.7-Rappresentazione della proiezione con piano illuminato perpendicolare all'asse ottico del proiettore, e γ angolo di apertura complessivo.

Relativamente alla figura 3.7 bisogna precisare che una volta scelto il numero di frange k da proiettare gli angoli ϑ_i assumono valore costante pari a:

$$\theta^* = \frac{\gamma}{k} = \text{cost} \quad (3.1.16)$$

L'espressione trovata equivale appunto alla condizione che le frange vengano proiettate in direzione radiale ad angoli equispaziati, considerando il sensore contenente la matrice dei pixel come una sorgente puntiforme con origine nel punto 0.

Risulta a questo punto che lo spessore occupato dalla proiezione della generica frangia sul piano di proiezione è quantificabile mediante l'analisi del triangolo costruito sulle rette di proiezione come riportato in seguente figura 3.8:

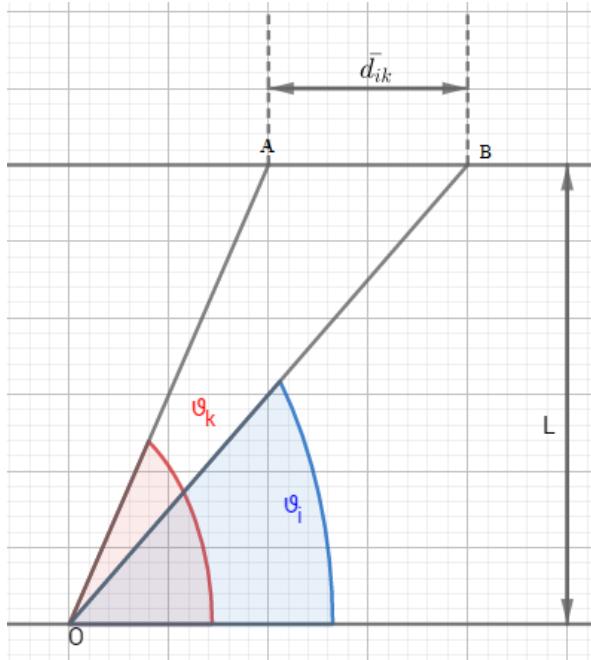


Fig. 3.8-Rappresentazione della proiezione di due fasci con riferimento agli angoli di proiezione.

Risultano quindi le seguenti relazioni:

$$\overline{OB} = \frac{L}{\sin(\theta_i)}$$

$$\overline{OA} = \frac{L}{\sin(\theta_k)}$$

$$\overline{d_{ik}} = \overline{OB} \cdot \cos(\theta_i) - \overline{OA} \cdot \cos(\theta_k)$$

$$\overline{d_{ik}} = \frac{L}{\tan(\theta_i)} - \frac{L}{\tan(\theta_k)} \quad (3.1.17)$$

Le relazioni (3.1.17) permettono di correlare lo spessore della frangia sia con la distanza fra il piano di proiezione e il proiettore, che all'angolo di apertura del proiettore, noto questo infatti tutti i valori θ_i e θ_k sono noti a priori: risulta infatti che i valori θ_i sono associati all'angolo di apertura del proiettore γ tramite la seguente relazione:

$$\theta_i = \frac{180 - \gamma}{2} + i \cdot \theta^* = \frac{180 - \gamma}{2} + i \cdot \frac{\gamma}{k} \quad (3.1.18)$$

con parametro k pari al numero di frange che si sceglie di proiettare.

Risulta perciò indispensabile in prima istanza la stima del valore dell'angolo di apertura del proiettore: questo risulta facilmente calcolabile nota la legge che lega le dimensioni della finestra di proiezione alla distanza di posizionamento della sorgente.

Analizzando infatti la seguente figura 3.9 che rappresenta la larghezza della finestra di proiezione per una generica distanza L alla quale viene posizionato il proiettore si ha:

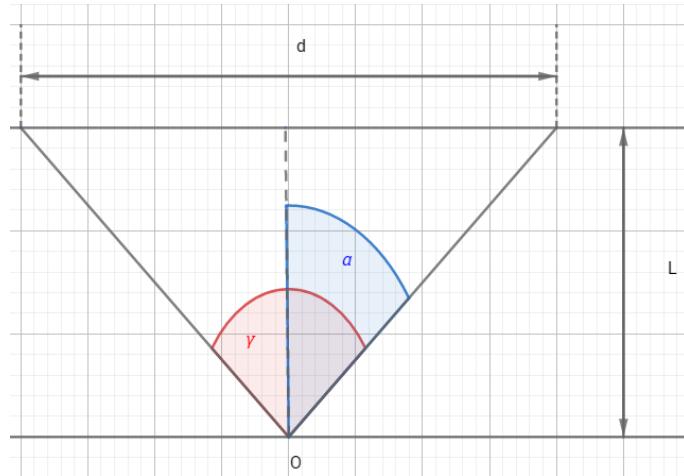


Fig. 3.9-Rappresentazione dell'angolo di apertura del proiettore γ , della larghezza d propria della finestra proiettata, e della distanza L di proiezione.

$$\frac{d}{2} = L \cdot \operatorname{tg}(\alpha)$$

$$\gamma = 2\alpha$$
(3.1.19)

Effettuando delle misure per diversi valori della distanza L risulta quindi per interpolazione lineare definita la legge che lega la larghezza della finestra di proiezione alla distanza dal proiettore, nota la quale è quantificabile appunto anche l'angolo di apertura γ del proiettore.

Si riportano di seguito i risultati ottenuti per via sperimentale sulla base delle misure effettuate:

Tabella 3.1.1-Valori della larghezza d della finestra proiettata per le diverse distanze L .

| L [mm] | d [mm] |
|----------|----------|
| 277 | 218 |
| 274 | 217 |
| 287 | 226 |

| | |
|-----|-----|
| 335 | 258 |
| 368 | 282 |

Riportando i dati su un piano cartesiano e procedendo mediante interpolazione lineare si ottiene il risultato riportato nella seguente figura:

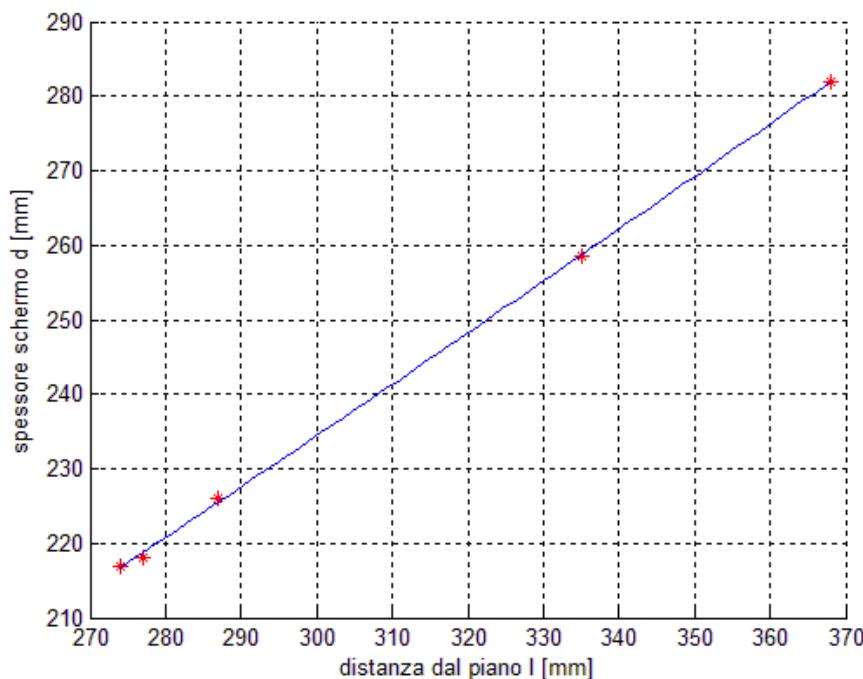


Fig. 3.10-Curva risultante dall'analisi effettuata.

In definitiva si ottiene un valore dell'angolo di apertura del proiettore pari a:

$$\gamma = 42.65^\circ$$

Quanto calcolato permette già in prima approssimazione di costruire una semplice griglia da proiettare, data la generica distanza fra piano di proiezione e proiettore.

Supponendo per esempio un valore della distanza L pari a 300[mm] e un numero di frange pari a 10.5 risulta la seguente distribuzione sul piano in termini di spessore per ogni rettangolo costituente metà della frangia:

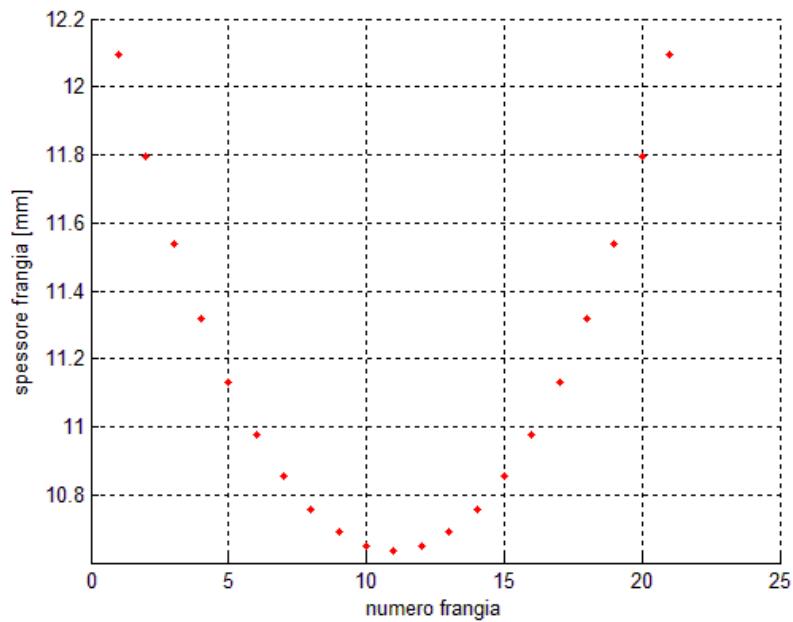


Fig. 3.11-Valori dello spessore per ogni rettangolo del reticolo proiettato.

con il seguente reticolo osservabile in direzione perpendicolare al piano proiettato:

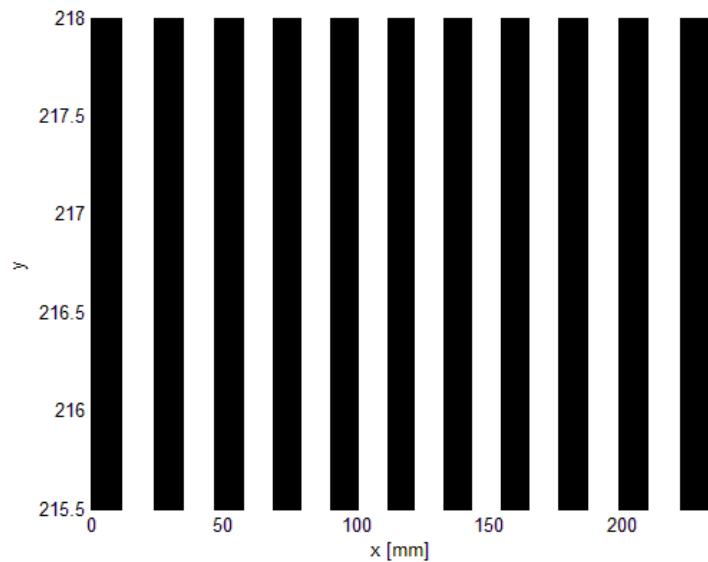


Fig. 3.12-Immagine che si forma sul piano a seguito della proiezione.

Come si nota essendo nulla l'angolazione relativa fra piano di proiezione e piano proiettato, la distribuzione degli spessori è simmetrica con asse di simmetria coincidente con quello del piano.

Questa caratteristica può essere meglio compresa analizzando il caso con angolo di apertura del proiettore maggiore, pari ad esempio a 120° a parità del numero complessivo di frange proiettate, come riportato di seguito:

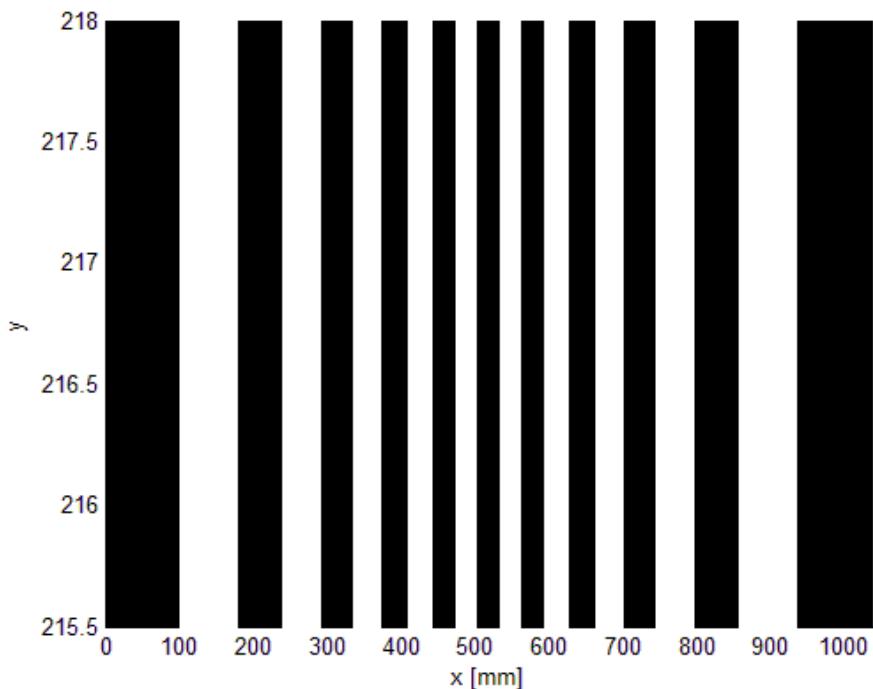


Fig. 3.13-Immagine che si forma sul piano a seguito della proiezione con angolo di apertura pari a 120° .

Si nota come aumentando l'angolo di apertura del proiettore nonostante la griglia rimanga simmetrica si possono presentare variazioni anche marcate di spessore fra le diverse frange.

Se ora si suppone di ruotare il piano soggetto alla proiezione la configurazione si modifica come riportato in figura seguente:

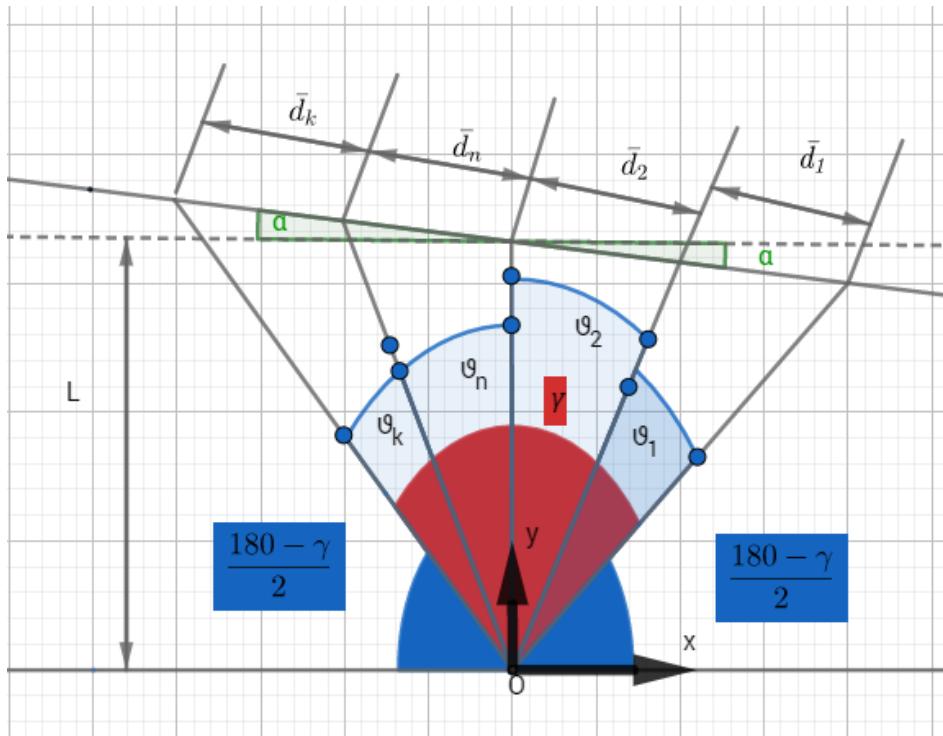


Fig. 3.14-Rappresentazione della proiezione con angolo di proiezione non nullo.

Come si può notare dalla figura 3.14 variano gli spessori delle singole frange, fenomeno dovuto principalmente alla variazione per ognuna di esse della distanza del punto di intersezione fra la retta del fascio luminoso ed il piano illuminato.

Anche la distribuzione dello spessore quindi risulterà per ovvi motivi non più simmetrica.

Il problema della ricerca degli spessori di ogni singola frangia risulta anche in questo caso di semplice risoluzione se viene espresso in termini di ricerca dei punti di intersezione fra il fascio di rette centrato nell'origine e la retta rappresentante il piano illuminato, posta a distanza L con coefficiente angolare $\tan(\alpha)$, con α pari appunto all'angolo di proiezione.

Noti infatti i punti di intersezione per ogni retta del fascio con il piano illuminato è calcolabile lo spessore di ogni frangia semplicemente esprimendola come distanza fra punti consecutivi di intersezione.

Utilizzando una distanza fra piano e proiettore pari ad 300mm, $\alpha=30^\circ$ e $k=13$ (con k pari al numero di frange da proiettare) si ottengono per esempio i seguenti risultati:

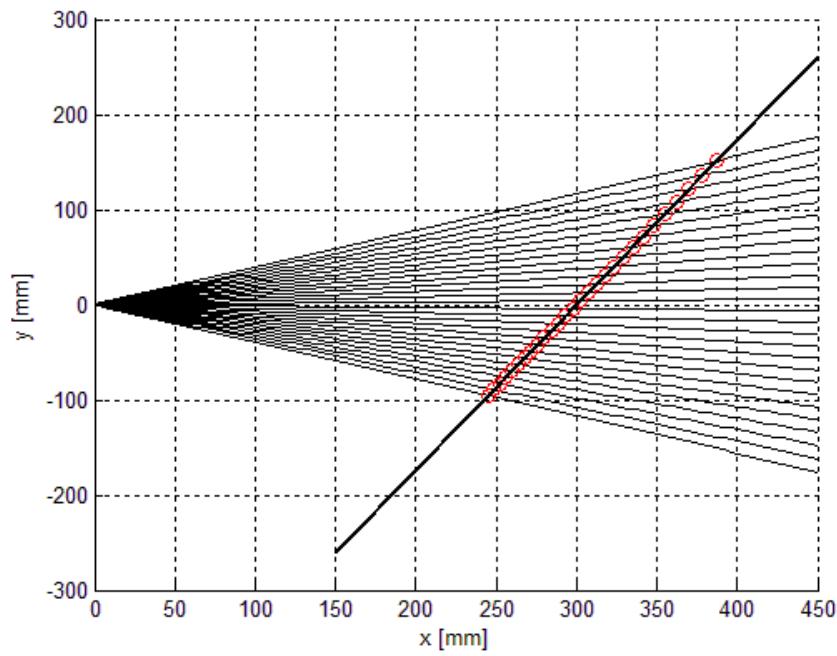


Fig. 3.15-Schema delle intersezioni del fascio di rette rappresentante la proiezione e la retta rappresentante il piano illuminato.

con il seguente reticolo risultante sul piano visto da una direzione perpendicolare allo stesso:

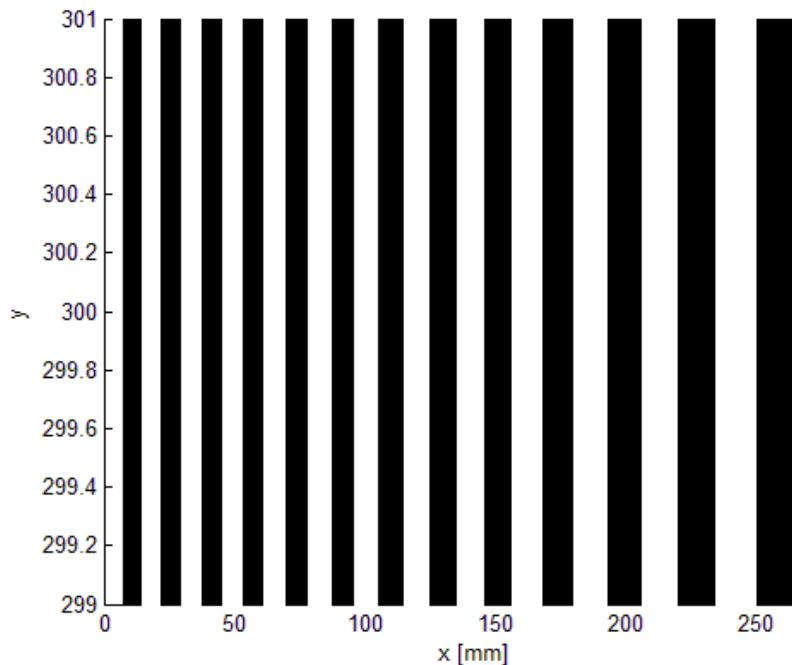


Fig. 3.16-Immagine risultante sul piano dalle intersezioni di figura 3.15.

In relazione alle intersezioni studiate si ha la seguente distribuzione di spessore per frangia:

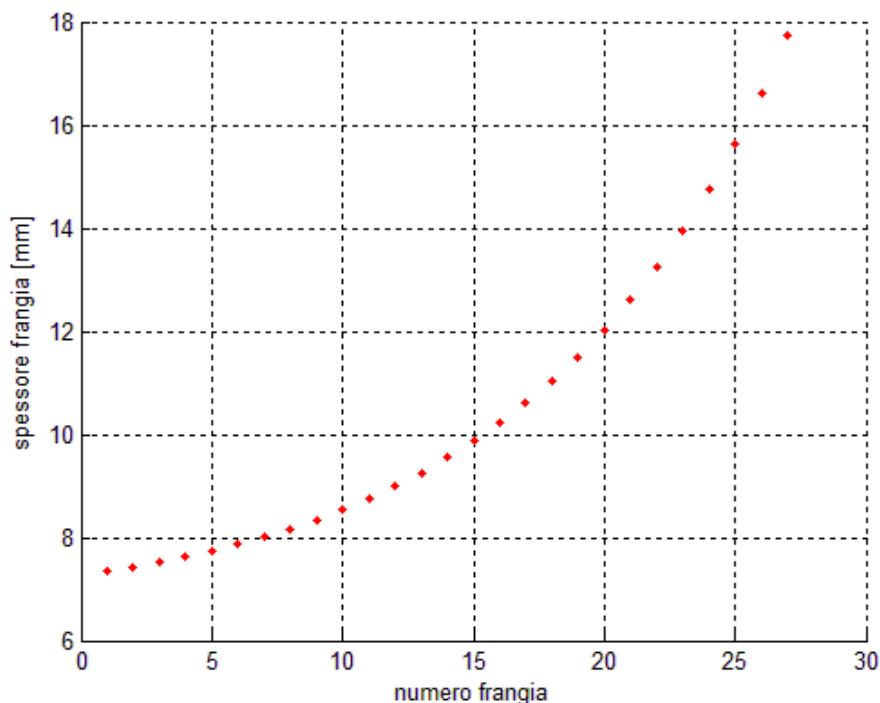


Fig. 3.17-distribuzione degli spessori di frangia sul piano relativi al reticolo di figura 3.16.

Il risultato ottenuto suggerisce inoltre un'importante astuzia tecnica: se la griglia calcolata in figura 3.26 venisse utilizzata come immagine da proiettare con angolo di proiezione pari a $-\alpha$ si otterebbe sul piano proiettato una griglia perfettamente simmetrica e di spessore costante per singola frangia: come conseguenza in pratica si avrebbe la compensazione della variazione del passo delle frange che diverrebbe costante, e sul piano di proiezione si passerebbe perciò al caso di proiezione di un fascio collimato.

Dati i parametri geometrici caratterizzanti le dimensioni del provino risulta inoltre a rigore di logica, vista la fisica della propagazione della luce, un numero finito di frange proiettabili, dipendenti oltre che dalle caratteristiche in termini quantitativi delle unità emissive e dimensioni geometriche dei pixels del sensore del proiettore, anche dalle dimensioni geometriche del provino stesso, soggetto alla proiezione: anche utilizzando il

massimo numero di frange proiettabili dal sensore, il numero di frange effettivamente visibili sul provino dipende comunque principalmente dalla distanza al quale si pone la sorgente luminosa ma anche dall'angolo relativo fra i due piani.

Si riporta si seguito una sintesi dei risultati nella seguente figura che lega il numero di frange intercettabili dal provino in funzione della distanza del proiettore (ipotizzata fra i 40 e 20 cm a causa delle dimensioni fisiche del pezzo) e dell'angolo di proiezione fra piano del sensore e quello soggetto alla proiezione, considerando un numero di frange proiettate pari a mille e una superficie illuminabile pari a 10 cm:

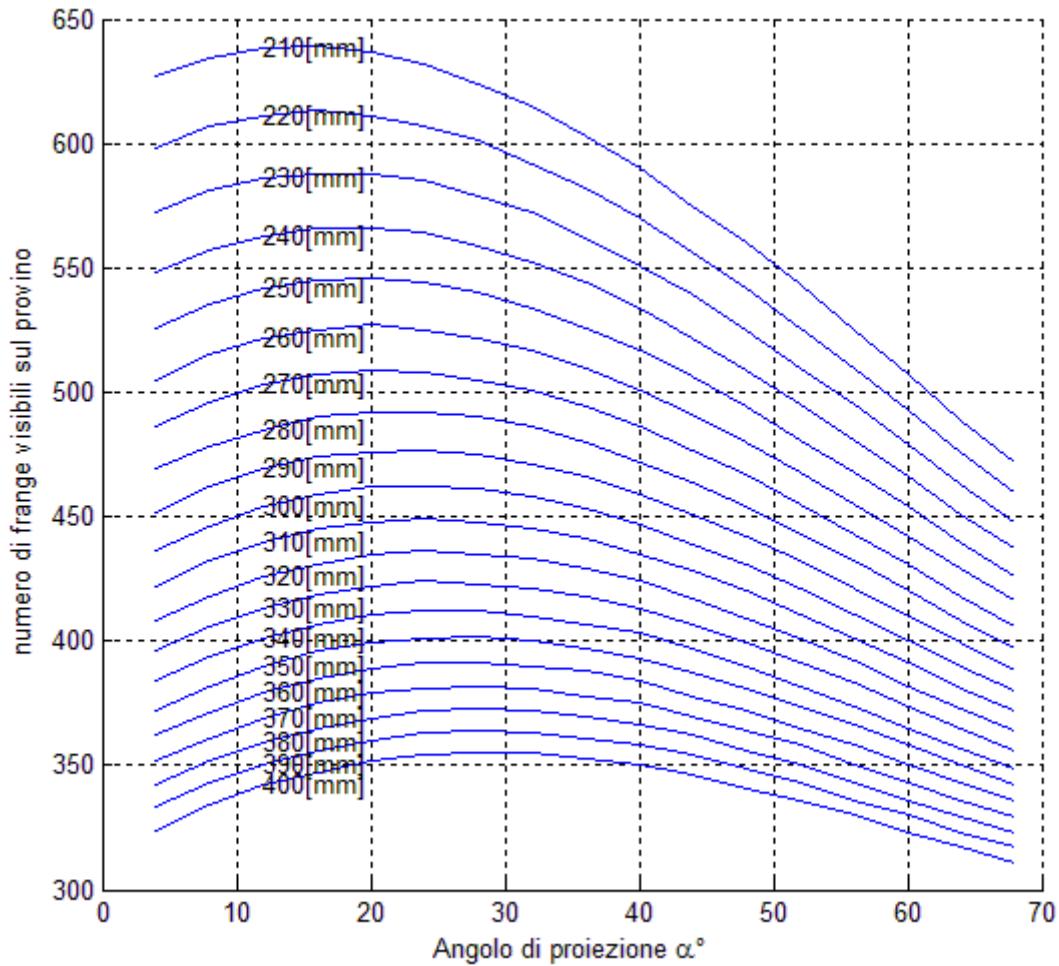


Fig. 3.18-Numeri di frange intercettabili dal provino in funzione della distanza di proiezione e all'angolo di proiezione.

Quanto affermato finora non risolverebbe tuttavia il problema dell'applicazione diretta della 3.8 nel calcolo della quota z in quanto ogni spostamento fuori dal piano porterebbe comunque una variazione seppur lineare e contenuta con l'entità dello spostamento stesso, nel passo delle frange.

Bisogna precisare inoltre che i risultati ottenuti fino a questo punto risultano essere validi solo in caso che il fascio di proiezione sia confinato a propagarsi nel piano: considerando infatti un sistema di riferimento tridimensionale, l'analisi effettuata analizza ciò che accade lungo un piano ε contenente il fascio luminoso e che interseca il piano di proiezione Π come mostrato in figura 3.19 seguente:

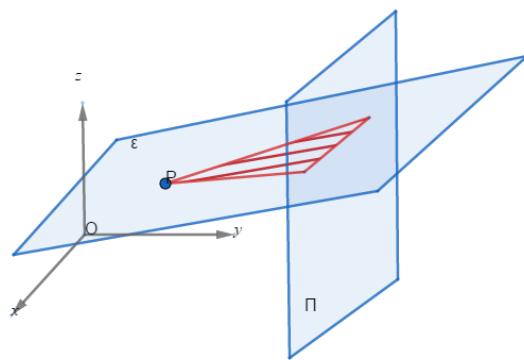


Fig. 3.19-Rappresentazione geometrica nello spazio del problema finora analizzato.

Risulta perciò chiara la necessità di raffinare la soluzione del problema considerando un modello che tenga conto di entrambi gli angoli di apertura γ e δ come riportato in figura seguente:

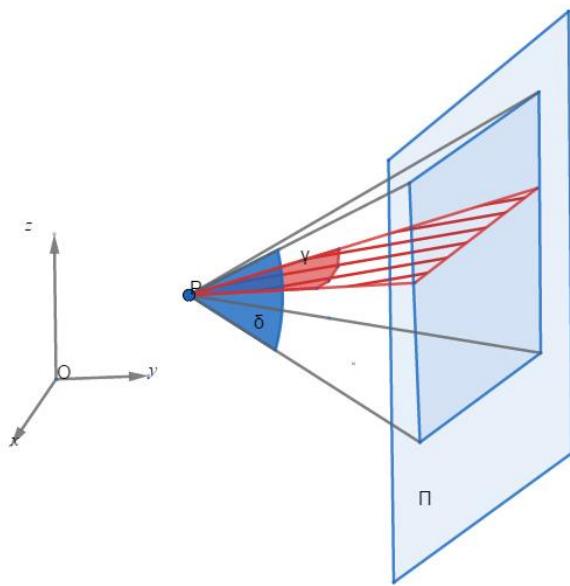


Fig. 3.20-Rappresentazione geometrica del problema effettivo: come si può notare il processo di proiezione dipende dagli angoli γ e δ .

Al fine della ricerca di una soluzione veloce ma allo stesso tempo affidabile al problema si considera ora di effettuare un'analisi delle caratteristiche del proiettore in dotazione (BENQW1070) sulla base del relativo datasheet fornito dall'azienda costruttrice, al fine di trovare in primo luogo i valori dei diversi angoli di apertura, e successivamente sul modo opportuno per utilizzarli.

3.1.3 Determinazione degli angoli di proiezione sulla base del datasheet.

Si riporta a tale fine nella seguente figura 3.21 il datasheet fornito dall'azienda:

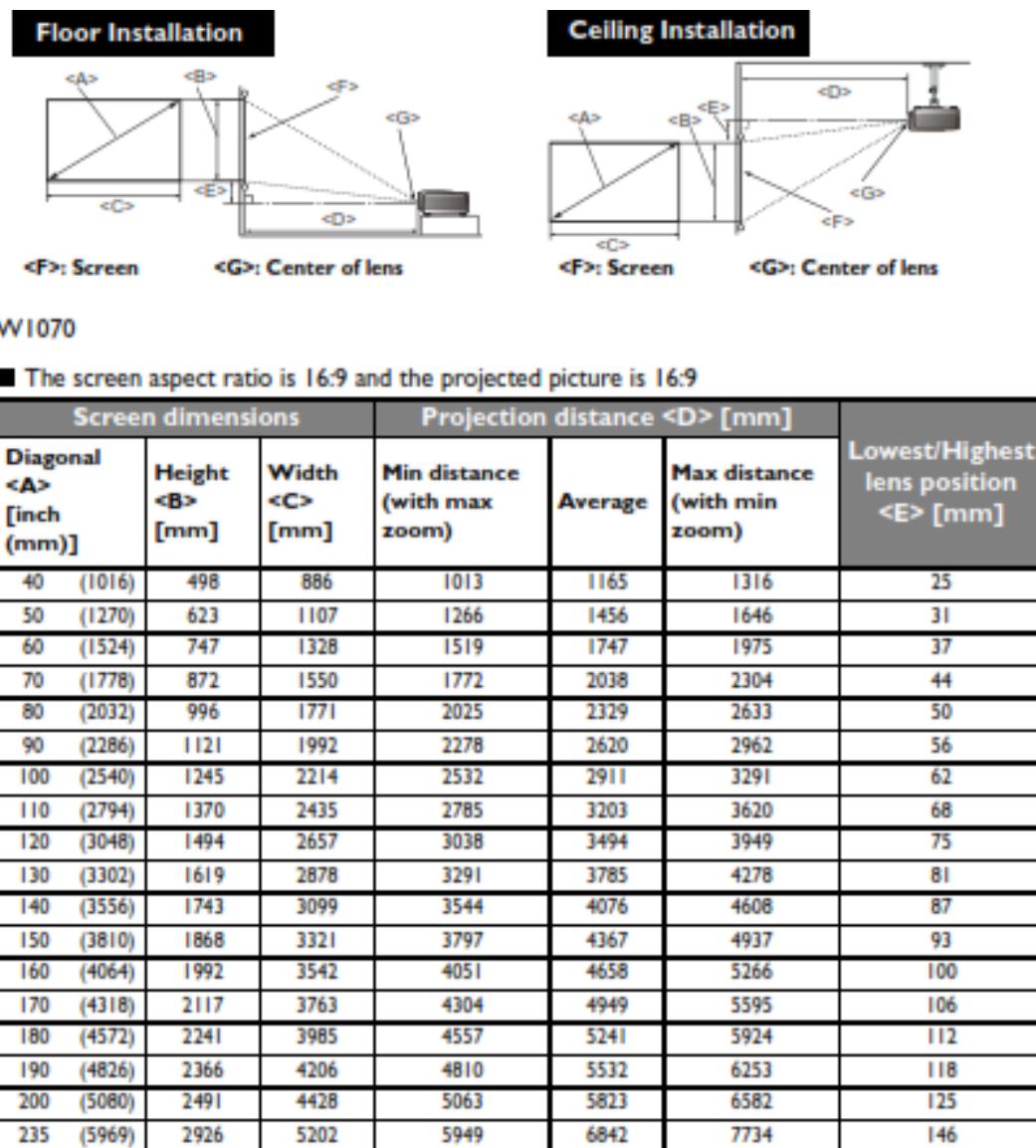


Fig. 3.21-Dati sulla proiezione forniti nel datasheet del proiettore Benq W1070.

Si procede quindi in prima istanza al calcolo dell'angolo di apertura γ (fig 3.22), al fine di confrontarne il valore con quello ottenuto sperimentalmente. Si ottiene in questo modo

il grafico di figura 3.23 che riporta il valore dell'angolo di apertura in funzione della distanza, calcolato secondo la seguente relazione:

$$d \cdot \operatorname{tg}\left(\frac{\gamma}{2}\right) = \frac{\text{width}}{2} \quad (3.1.20)$$

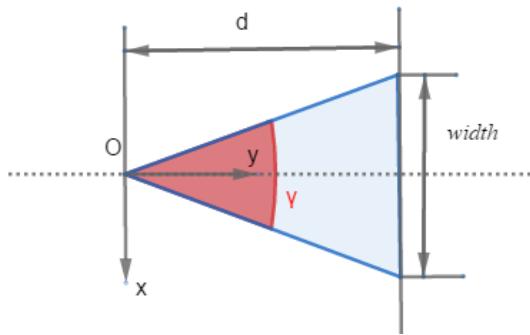


Fig. 3.22-Schema riportante l'angolo γ .

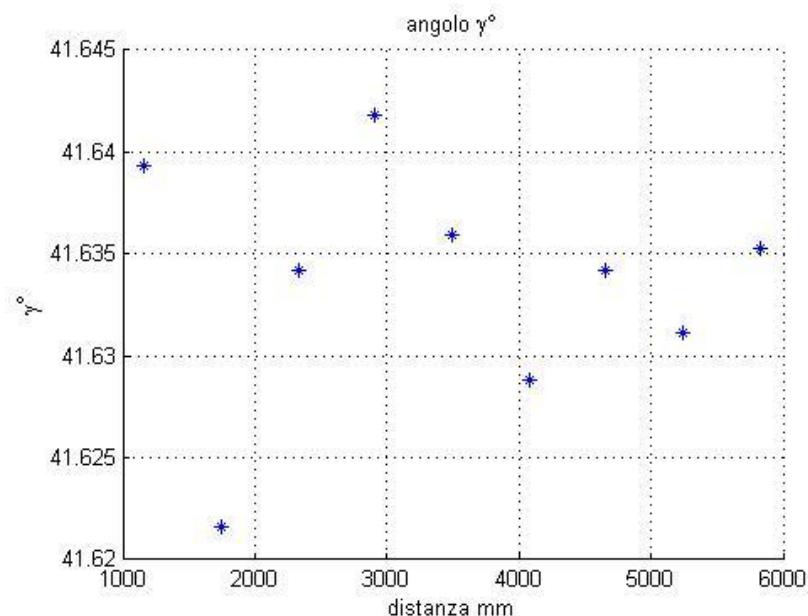


Fig. 3.23-Variazione dell'angolo γ con la distanza di proiezione.

Come si può notare dal grafico di figura 3.23, nonostante l'andamento non risulti lineare resta il fatto che i valori dell'angolo di apertura γ rimangono all'interno di un range di valori abbastanza ristretto, risulta quindi giustificato considerare γ pari al valor medio dei punti risultanti in figura 3.23, ottenendo quindi un valore pari a:

$$\gamma = 41.63^\circ$$

Il valore trovato risulta poco diverso dal valore precedentemente calcolato per via sperimentale di $42,65^\circ$.

Come mostrato nella seguente figura 3.24, e considerando i dati presenti nell'ultima colonna nel datasheet di figura 3.21 risulta l'esistenza di un angolo di offset di proiezione:

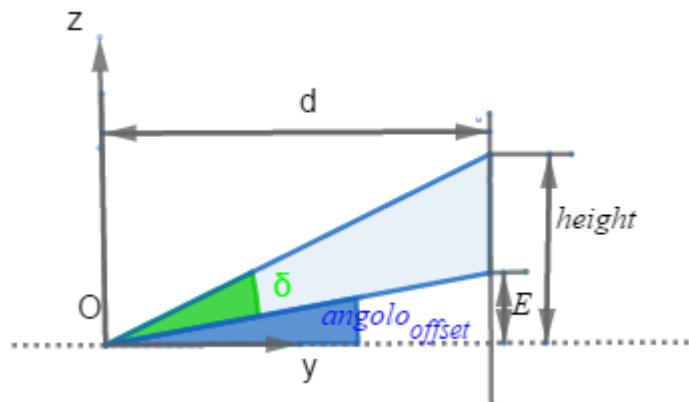


Fig. 3.24-schema riportante gli angoli di offset e δ .

anche quest'ultimo può essere calcolato in maniera simile a quella usata per il calcolo di γ . Relativamente alla seguente figura 3.25:

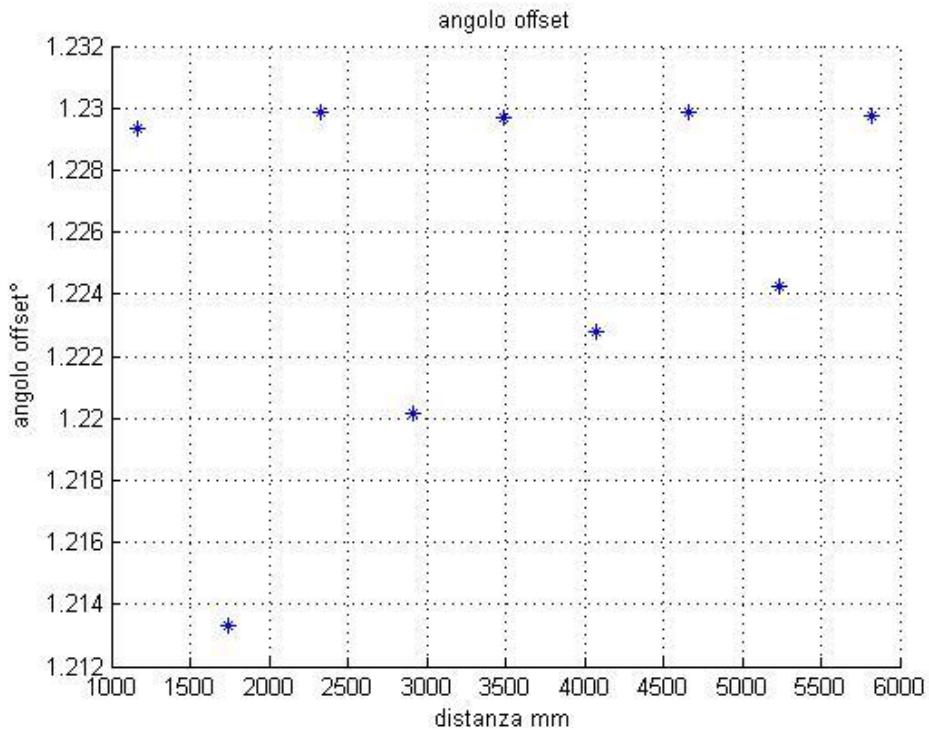


Fig. 3.25-Variazione dell'angolo di offset con la distanza di proiezione.

risulta un valore dell'angolo di offset pari a:

$$\text{angolo offset} = 1.22^\circ$$

calcolato come valore medio dei punti riportati in figura 3.25.

Al fine di determinare l'angolo di apertura δ si procede nella seguente maniera: si considera l'altezza di proiezione (indicata come *height* fig3.24) dalla quale risulta calcolabile l'angolo di proiezione complessivo rispetto dall'asse y, successivamente tolto il contributo legato all'angolo di offset risulta determinato l'angolo di apertura δ .

Si ottiene in questo modo il seguente grafico di figura 3.26:

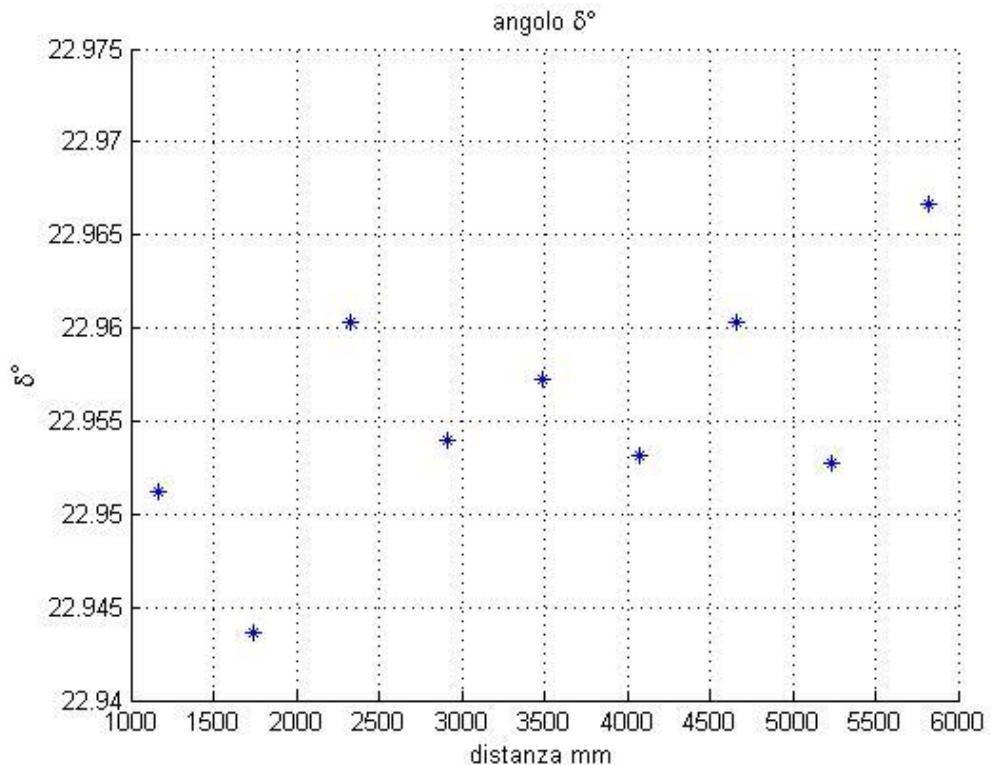


Fig. 3.26-Variazione dell'angolo δ con la distanza di proiezione.

Anche in questo caso visto il range ristretto di variazione si considera il valore medio ottenendo:

$$\delta = 22.95^\circ$$

A questo punto la conoscenza degli angoli e delle dimensioni espresse in unità metriche ([mm] nel nostro caso) della finestra di proiezione al variare della distanza, unitamente all'informazione sulle dimensioni in pixel del sensore del proiettore, permettono in maniera intuitiva di calcolare anche il fattore di scala che lega lo spessore sulla finestra di proiezione effettivamente competente al singolo pixel del sensore.

Noto infatti che il proiettore è caratterizzato da un sensore di 1920x1080 pixels, il rapporto fra la larghezza della finestra di proiezione competente alla generica distanza ' d ' di proiezione espressa in millimetri, e la rispettiva dimensione del sensore espressa in pixels permette di ottenere i fattori di scala ricercati.

Si riporta in seguente figura 3.27 il risultato del calcolo:

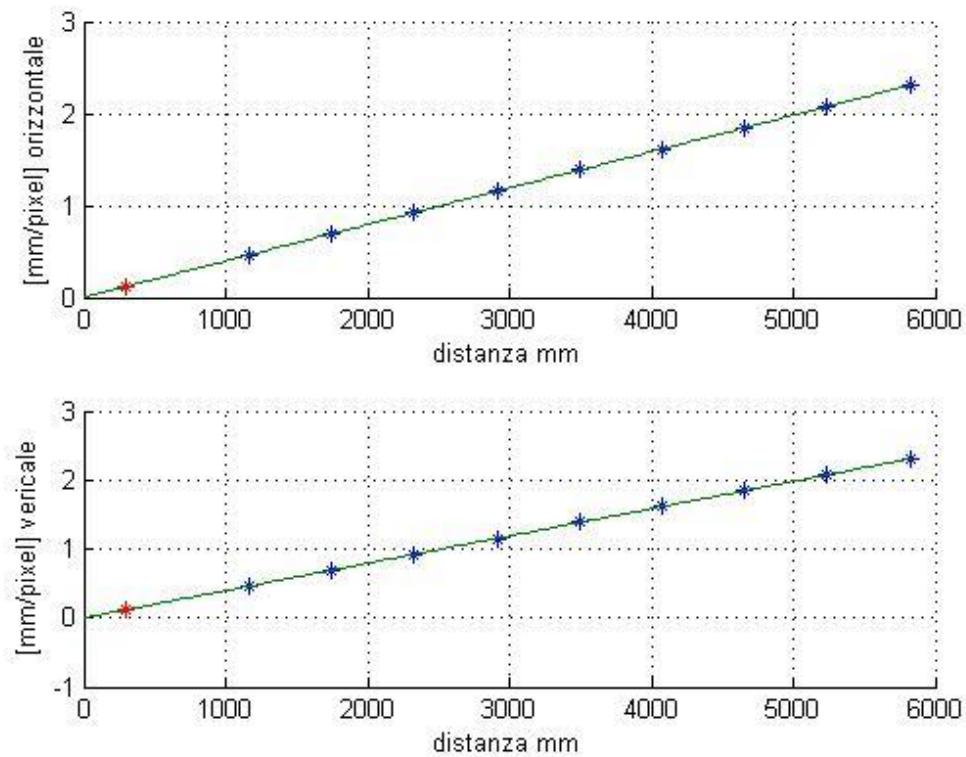


Fig. 3.27-Variazione dei fattori di scala con la distanza di proiezione: in rosso il valore competente alla distanza utilizzata nelle prove pari a 290 mm.

Successivamente risulta anche determinabile l'angolo competente al singolo pixel del sensore come:

$$[\text{angolo/pixel}]_x = S_x = \frac{\gamma}{N_{\text{pixel}}} = \frac{41.63^\circ}{1920} = 0.0217^\circ \text{ [gradi/pixel]}$$

$$[\text{angolo/pixel}]_y = S_y = \frac{\delta}{N_{\text{pixel}}} = \frac{22.95^\circ}{1080} = 0.0213^\circ \text{ [gradi/pixel]}$$

Per la generica distanza di proiezione, scelta una finestra di $n \times m$ pixel da proiettare si è quindi in grado anche di prevedere l'effettiva dimensione in unità metriche della finestra di proiezione risultante sul piano illuminato.

Rimane perciò da definire ora un modello che permetta in funzione agli angoli ricavati, di costruire una griglia secondo una specifica geometria la cui proiezione permetta di

ottenere sul piano soggetto alla proiezione una distribuzione d'intensità equivalente a quella che si avrebbe in caso di proiezione di un fascio collimato.

3.1.4 Modello geometrico per la costruzione di griglie compensate.

L'approccio al problema è di tipo inverso infatti partendo dall'immagine proiettata che si vuole realizzare sul piano si studia la soluzione geometrica alla trasformazione di proiezione centrale che l'ha prodotta.

Primo passo della trattazione è la scelta della dimensione in pixel dell'immagine sul pc che si vuole proiettare, e relativamente alla seguente figura 3.28 si indicano le due dimensioni in pixel n ed m di quest'ultima:

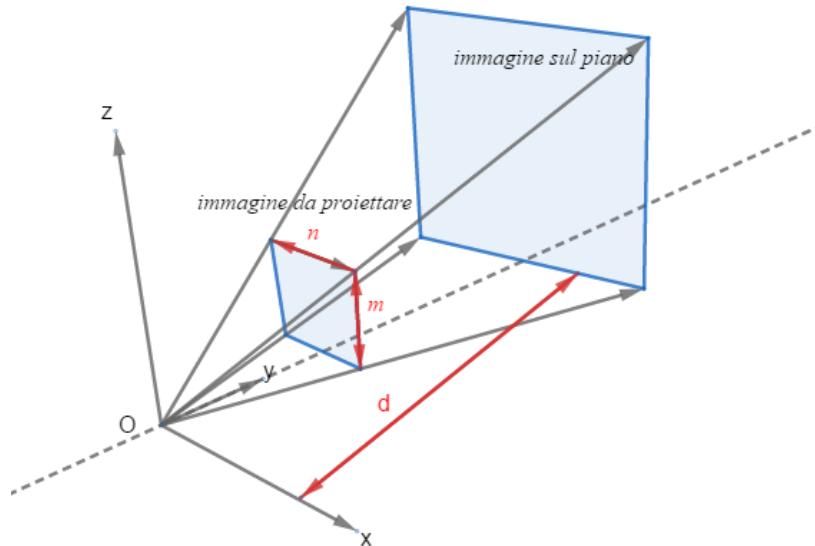


Fig. 3.28-Rappresentazione della finestra di proiezione e dell'immagine risultante sul piano.

Si definiscono ora i due angoli effettivi di proiezione dipendenti dalle dimensioni della finestra da proiettare come riportato in figura 3.29 seguente:

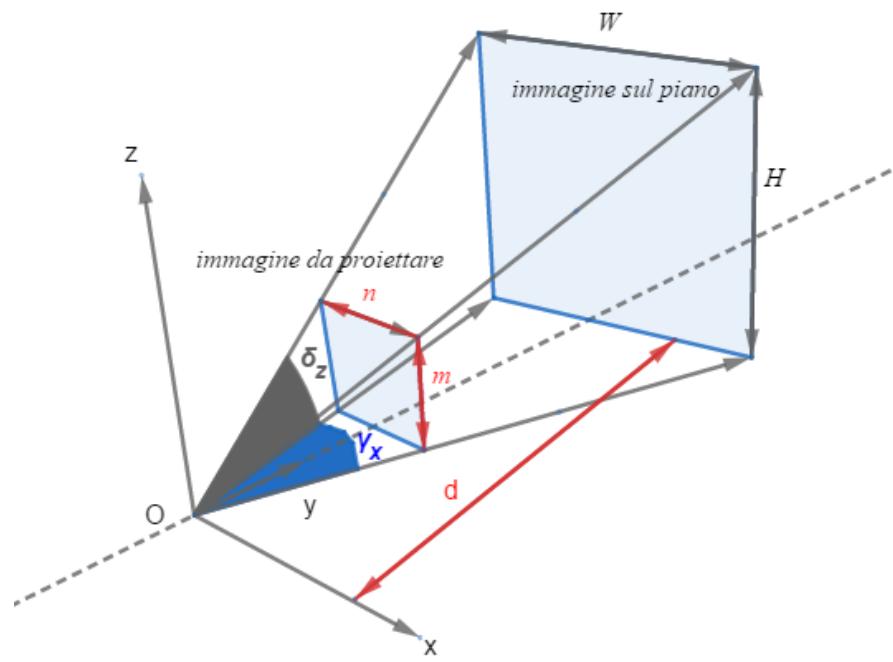


Fig. 3.29-Rappresentazione della finestra di proiezione e dell'immagine risultante sul piano in relazione agli angoli di proiezione.

Risulta perciò che:

$$\begin{aligned}\gamma_x &= S_x \cdot n \\ \delta_z &= S_y \cdot m\end{aligned}\tag{3.1.21}$$

Nota quindi la distanza d di proiezione risulta possibile calcolare le dimensioni della risultante finestra proiettata sul piano, in particolare si hanno le seguenti relazioni:

$$\begin{aligned}W &= 2 \cdot d \cdot \sin\left(\frac{\gamma_x}{2}\right) \\ H &= d \cdot [\sin(\delta_z + \text{angolo offset}) - \sin(\text{angolo offset})]\end{aligned}\tag{3.1.22}$$

Il calcolo effettuato permette quindi di costruire l'immagine che si vorrebbe ottenere a seguito della proiezione.

Imponendo per esempio un numero di frange pari a 12 aventi passo costante, e considerando un valore $d=300$ mm e una dimensione della finestra da proiettare pari ad $n=420$ [pixel] ed $m=420$ [pixel] si ottiene il seguente risultato per gli effettivi angoli di apertura:

$$\gamma_x = 8.83^\circ$$

$$\delta_z = 8.92^\circ$$

E un'immagine risultante sul piano illuminato come di seguito riportato in figura 3.30:

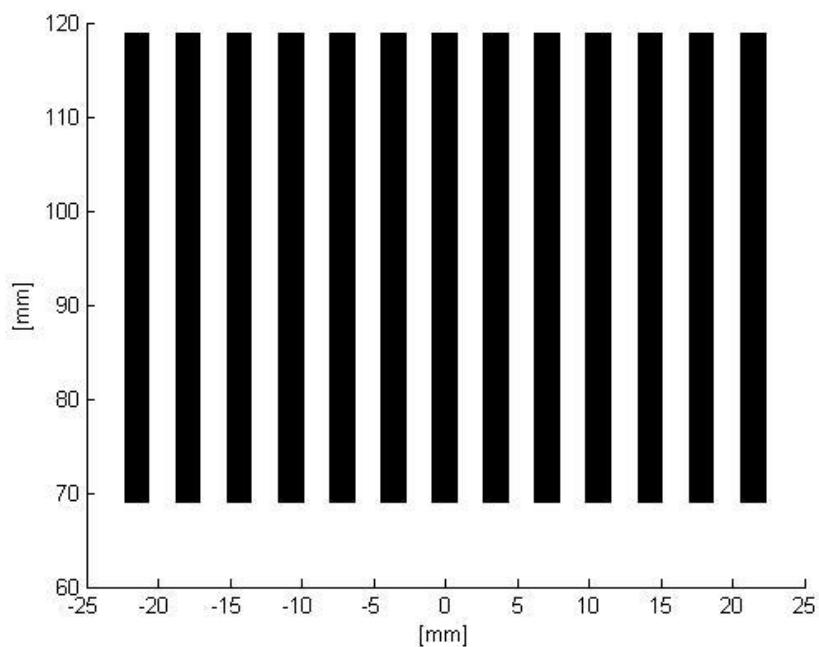


Fig. 3.30-Rappresentazione dell'immagine risultante sul piano a seguito di proiezione con angoli $\delta_z=8.92^\circ$ e $\gamma_x=8.83^\circ$.

Rimane ora da definire come debba essere strutturata la distribuzione d'intensità nella finestra da proiettare in modo che la sua proiezione equivalga appunto al caso di fascio collimato e permetta di ottenere un'immagine sul piano come quella riportata in figura 3.30.

A tal fine si può pensare di risolvere il problema studiando la proiezione dell'immagine in figura 3.30 su un piano posto ad una distanza d_I rispetto al centro di proiezione centrale.

Si consideri infatti che essendo nota la posizione di tutti i punti sull'immagine che si vuole ottenere, risulta facilmente calcolabile la loro proiezione su un piano interposto fra la sorgente luminosa del sensore e il piano.

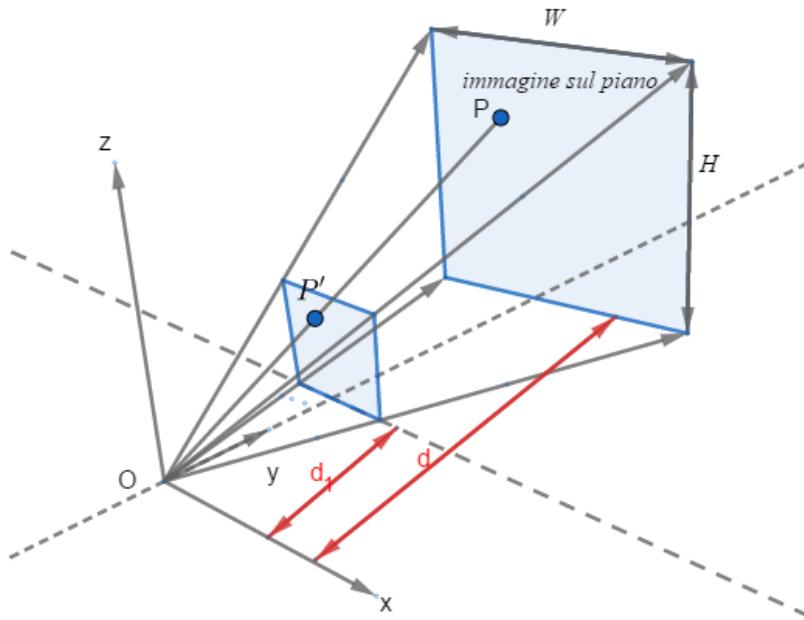


Fig. 3.31-Rappresentazione della proiezione del generico punto P dell'immagine sul punto P' appartenente ad un piano posto a distanza d_1 .

relativamente alla figura 3.30 si indichi il vettore posizione del punto P come:

$$\{P\} = (x_p, y_p, z_p) \quad (3.1.23)$$

E la posizione dell'origine coincidente al centro di proiezione come:

$$\{O\} = (0,0,0) \quad (3.1.24)$$

Si considera ora l'equazione parametrica della generica retta passante per i due punti, data da:

$$\begin{cases} x(t) = x_0 + t \cdot (x_p - x_0) \\ y(t) = y_0 + t \cdot (y_p - y_0) \\ z(t) = z_0 + t \cdot (z_p - z_0) \end{cases} \quad \begin{cases} x(t) = t \cdot x_p \\ y(t) = t \cdot y_p \\ z(t) = t \cdot z_p \end{cases} \quad (3.1.25)$$

Per quanto riguarda ora il piano posto a distanza d_1 si ha che la sua espressione generica risulta:

$$f(x,y,z) = a \cdot x + b \cdot y + c \cdot z + d = 0 \quad (3.1.26)$$

con a,b,c termini del vettore direzionale del piano.

Conoscendo perciò che la posizione del piano risulta parallela al piano xz e distante della quantità d_1 lungo y , si può trovare sia il valore del termine noto d sfruttando la condizione di appartenenza del punto di coordinate $x=0, y=d_1$ e $z=0$ al piano stesso, che il valore dei termini del vettore direzione che in questo caso valgono $a=0, b=1, c=0$, sostituendo questi valori in (3.1.26) si ottiene:

$$d = -d_1 \cdot b \quad (3.1.27)$$

Sostituendo ora l'equazione parametrica della retta congiungente il punto P con l'origine O , nell'espressione del piano si ottiene:

$$a \cdot t \cdot x_p + b \cdot t \cdot y_p + c \cdot t \cdot z_p - d_1 \cdot b = 0 \quad (3.1.28)$$

Che ha soluzione in t pari a:

$$t = \frac{d_1 \cdot b}{a \cdot x_p + b \cdot y_p + c \cdot z_p} \quad (3.1.29)$$

Sostituendo il valore di t nell'espressione della retta (3.1.25) si ottiene:

$$\left\{ \begin{array}{l} x(t) = \frac{d_1 \cdot b}{a \cdot x_p + b \cdot y_p + c \cdot z_p} \cdot x_p \\ y(t) = \frac{d_1 \cdot b}{a \cdot x_p + b \cdot y_p + c \cdot z_p} \cdot y_p \\ z(t) = \frac{d_1 \cdot b}{a \cdot x_p + b \cdot y_p + c \cdot z_p} \cdot z_p \end{array} \right. \quad (3.1.30)$$

Che rappresenta esattamente il punto di proiezione di P su P' sul piano posto a distanza d_1 , in quanto è semplicemente l'intersezione della retta col piano.

Per quanto riguarda i valori di a, b, c del vettore direzione del piano risulta chiaro che se venissero calcolati secondo la direzione di proiezione si otterrebbe successivamente dal calcolo delle intersezioni l'immagine da proiettare che permette d'ottenere un reticolo sul piano illuminato come in figura 3.30.

In tal senso noto l'angolo di proiezione α e quello di offset che si indica ora con β si ha che il vettore direzione del piano è pari ad:

$$\{r\} = [a, b, c] = [0, 1, 0] \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \\ 0 & \cos(\beta) & -\sin(\beta) \\ 0 & \sin(\beta) & \cos(\beta) \end{bmatrix} \quad (3.1.31)$$

Essendo α calcolato attorno all'asse z e β attorno all'asse x.

Come esempio si consideri l'intersezione dell'immagine in figura 3.30 con un piano caratterizzato da $\alpha=30^\circ$ e $\beta=10^\circ$, il risultato è rappresentato in figura 3.32 seguente:

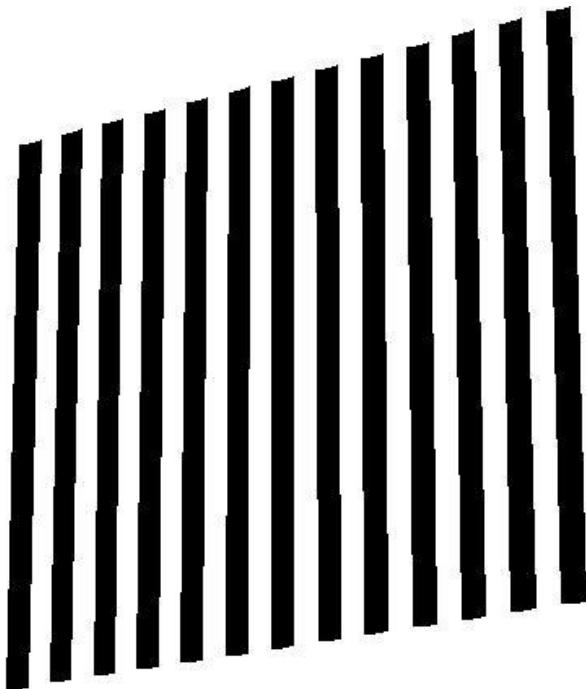


Fig. 3.32-Reticolo risultante dalla proiezione dell'immagine in figura 3.29 sul piano posto a distanza d_1 rispetto al centro di proiezione e ruotato degli angoli $\alpha=30^\circ$ e $\beta=10^\circ$.

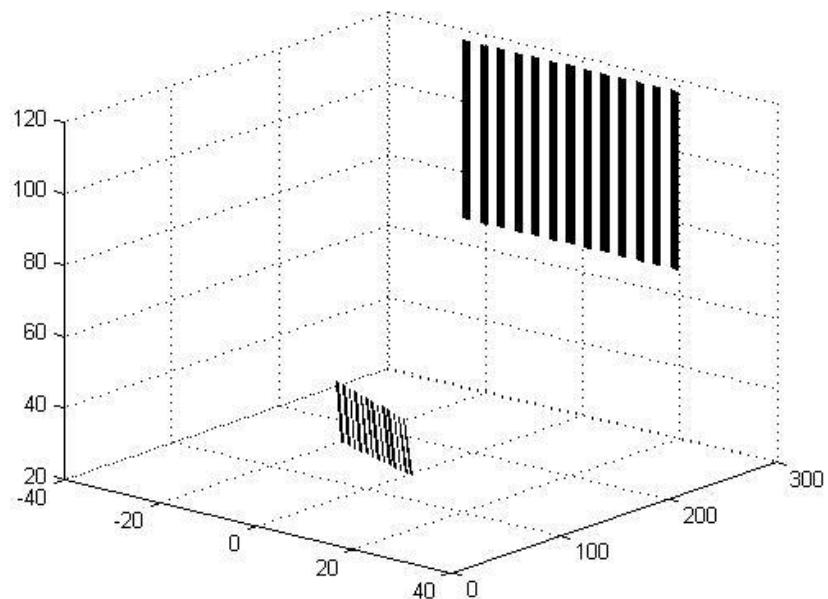


Fig. 3.33-Rappresentazione tridimensionale della proiezione dell'immagine sul piano ruotato.

Quindi proiettando l'immagine in figura 3.32 dopo aver posizionato il proiettore tale che presenti un angolo di proiezione $\alpha=30^\circ$ e $\beta=10^\circ$ si otterrebbe sul piano proiettato l'immagine in figura 3.29, oppure fissata la posizione del proiettore e calcolati gli angoli basterebbe proiettare un'immagine costruita utilizzando il valore degli angoli di proiezione.

Risulta altresì chiaro che la posizione d_1 al quale si pone il piano necessario al calcolo delle intersezioni non risulta rilevante ai fini del risultato ottenuto essendo quest'ultimo necessario solo ai fini della ricostruzione dell'immagine da proiettare, le cui dimensioni si ricorda essere imposte a priori in termini di pixel e quindi slegate dalle effettive dimensioni metriche dell'immagine risultante dal calcolo delle intersezioni.

3.1.5 Metodo per l'applicazione del phase shifting e DIC2D in assenza di strumentazione telecentrica.

Ottenuta quindi l'immagine da proiettare che equivale ad una griglia compensata risulta altresì determinato per ogni pixel di quest'ultima anche il valore degli angoli di proiezione.

Relativamente alla seguente figura 3.34 si consideri infatti un punto P' calcolato dall'intersezione:

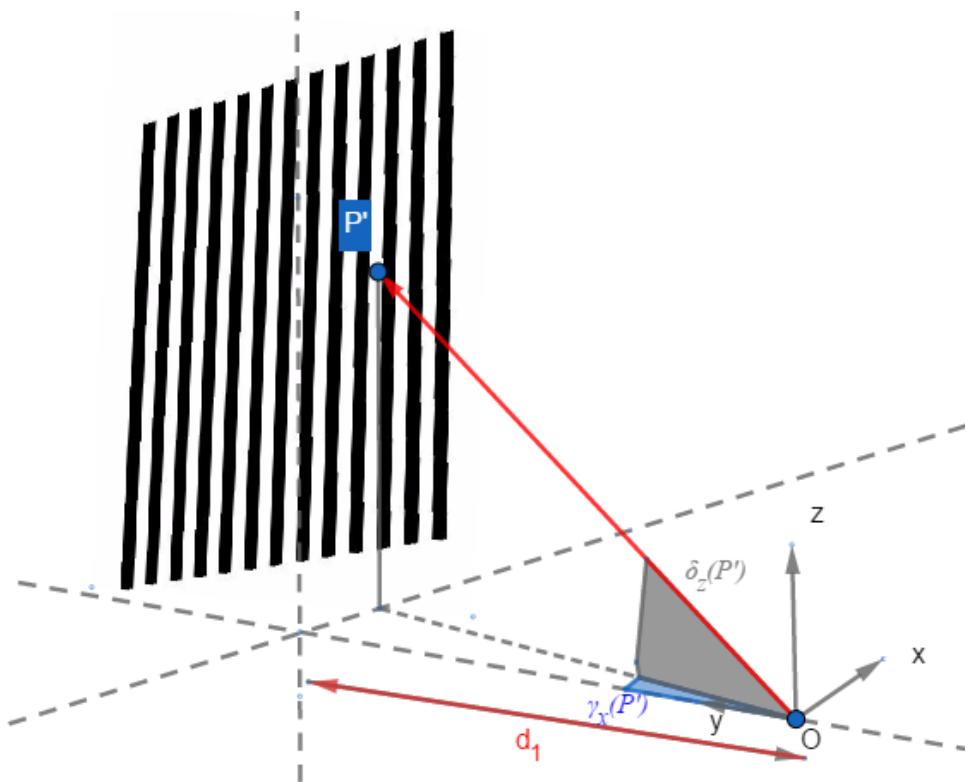


Fig. 3.34-Determinazione degli angoli di proiezione del generico punto P' sulla griglia compensata da proiettare.

Risulta per ques'ultimo calcolabile l'angolo $\gamma_x(P')$ rispetto al centro di proiezione in direzione x ed $\delta_z(P')$ rispetto a z, valgono infatti le seguenti relazioni:

$$\begin{aligned}\gamma_x(P') &= \arctg\left(\frac{x_{P'}}{d_1}\right) \\ \delta_z(P') &= \arctg\left(\frac{z_{P'}}{d_1}\right)\end{aligned}\tag{3.1.32}$$

Se l'operazione viene ripetuta per tutti i punti si ottiene una matrice $n \times m \times 2$ delle stesse dimensioni della finestra da proiettare che contiene la griglia compensata, i cui elementi contengono gli angoli di proiezione associati ad ogni pixel.

Si concentra ora l'attenzione sulla immagine che verrà scattata dalla fotocamera.

Vista la relazione che lega la dimensione metrica che ogni pixel proiettato andrà ad assumere sul piano illuminato (rappresentato in nostro caso dal provino), e che si ricorda dipendere dalla distanza, risulta chiara la possibilità che l'immagine finale sul sensore

della fotocamera assuma una dimensione in pixels diversa da quella proiettata e una forma completamente diversa da quella visibile sul piano.

Tale fenomeno è legato alla trasformazione di proiezione centrale della fotocamera, e dipende in questo caso dagli angoli con i quali si osserva il piano e la distanza.

Risulta quindi chiaro che quanto detto pone un problema concernente l'effettiva corrispondenza dell'angolo di proiezione precedentemente calcolato competente al generico pixel dell'immagine finale ottenuta.

A seguito della proiezione si forma un'immagine sul piano di dimensioni W ed H , successivamente scattata l'immagine con una fotocamera risulta un'immagine finale di dimensioni L ed A come si può notare dalla seguente figura 3.35:

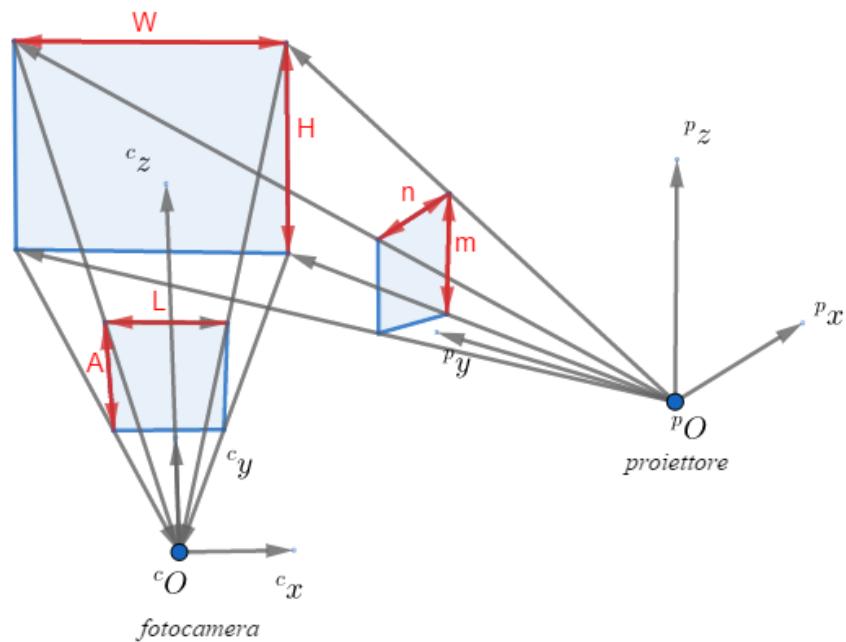


Fig. 3.35-Variazione delle dimensioni della finestra proiettata sul piano illuminato e sul sensore della fotocamera.

Il problema sarebbe comunque risolvibile in caso si trovi un modo per conglobare nell'immagine proiettata anche l'informazione relativa all'angolo di proiezione dei suoi punti: qualsiasi sia la dimensione risultante della finestra sul piano illuminato si avrebbe

la conoscenza per ogni punto nell'immagine sul sensore, dell'angolo di proiezione che l'ha prodotto.

Una soluzione al problema è rappresentata dalla proiezione ed acquisizione in sequenza di due immagini: la prima riportante l'immagine della griglia compensata, e la seconda riportante una matrice di punti di controllo estratti dall'immagine della griglia compensata associati ai relativi valori degli angoli di proiezione come in figura 3.36 seguente:

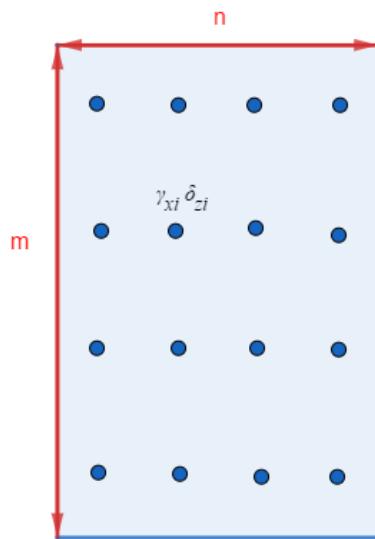


Fig. 3.36-Griglia di controllo costruita utilizzando un campione di punti estratti dalla finestra di proiezione dei quali sono note le angolazioni di proiezione..

Successivamente attraverso gli algoritmi di *interest point detection* determinata in termini di pixel la loro posizione nel secondo scatto, risulterebbero noti gli angoli di proiezione dei punti nella prima immagine alle medesime posizioni.

Ovviamente in questo caso a meno che non si costruisca un modello che permetta di interpolare in maniera corretta i valori degli angoli lungo l'intera superficie dell'immagine scattata, la risoluzione dipenderebbe direttamente dal numero di punti di controllo proiettati e da quelli che effettivamente gli algoritmi di detection riuscirebbero ad estrarre.

Tuttavia in questa maniera sarebbe risolto il problema dell'applicabilità della relazione (3.8) essendo a questo punto sia gli angoli di proiezione e quelli di osservazione dopo opportuna calibrazione della fotocamera.

3.2 Applicazione ad un caso reale e risultati

Relativamente alla trattazione sviluppata al capitolo precedente si nota come la conoscenza degli angoli di apertura del proiettore, e delle dimensioni della matrice dei pixel del proiettore, permetta di concludere che l'angolo di proiezione competente al singolo pixel risulta molto piccolo in termini quantitativi.

Si riportano a tal proposito anzitutto i risultati ottenuti dall'analisi precedentemente effettuata (tabella 3.2):

Tabella 3.2-Valori degli angoli di proiezione precedentemente calcolati.

| | |
|--------------------------|------------------------------------|
| Angolo apertura γ | 41.63° |
| Angolo apertura δ | 22.95° |
| Angolo offset β | 1.22° |
| Dimensioni sensore | 1920 x 1080 |
| S_x | 0.0217 [gradi/pixel _x] |
| S_y | 0.0213 [gradi/pixel _y] |

Quanto appena affermato suggerisce che la variazione del passo delle frange a seguito di spostamenti fuori dal piano dovrebbe essere comunque di lieve entità in caso di piccoli spostamenti. Al fine di verificare tale ipotesi si effettua quindi in primo luogo un'analisi quantitativa per via sperimentale sulla variazione dello spessore delle frange a seguito di uno spostamento rigido del piano illuminato verso il centro di proiezione della fotocamera.

Per quanto riguarda la correlazione digitale si è invece usata un'astuzia tecnica in modo da eliminare il problema dovuto all'eventuale sovrapposizione dello speckle e delle frange nelle immagini risultanti: in particolare si è fatto uso di una vernice fluorescente per l'applicazione dello speckle in maniera tale che illuminando il provino con una Lampada di Wood sia visibile solo lo speckle necessario alla DIC2D, mentre in luce bianca proiettando le griglie siano visibili solo ques'ultime.

Inoltre, poiché risulta che un altro fattore importante nello sviluppo di metodologie simili è rappresentato dalla quantità di pre e post processing da effettuare sui dati trattati che sui risultati ottenuti, si procede in primo luogo all'applicazione del metodo misto a phase shifting e correlazione digitale d'immagine non calibrato al fine di ottenere l'informazione sulla qualità dei risultati da aspettarsi: non si è fatto uso in pratica né di una fotocamera calibrata per quantificare in maniera esatta gli angoli di osservazione, e nemmeno dell'utilizzo della proiezione delle matrici dei punti di controllo (figura 3.36), per la conoscenza puntuale degli angoli di proiezione.

In tal senso l'esempio trattato rappresenta un semplice calcolo di primo tentativo al fine di delineare in maniera chiara sia la pipeline dei processi necessari per ottenere gli spostamenti tridimensionali, sia la quantità di post processing richiesta sui risultati in relazione all'effettiva informazione che si va a perdere sul risultato finale. La seguente trattazione perciò è sviluppata secondo un'impostazione atta ad esplorare a livello qualitativo e grafico i risultati finali che si possono ottenere alla fine del processo di calcolo: in funzione a quest'ultimi risulterà infatti giustificato o meno, sia uno studio comparativo con i risultati delle soluzioni teoriche note, sia un paragone con l'applicazione esatta secondo il modello di fotocamera calibrata con angoli di proiezione noti e compensazione degli spostamenti apparenti, sia infine un'analisi comparativa dei risultati con un software di tipo commerciale.

3.2.1 Variazione dello spessore di frangia con la distanza di proiezione

In via preliminare si riporta anzitutto la differenza fra i reticolli risultanti sul piano dalla proiezione di una griglia compensata secondo il metodo proposto al paragrafo 3.1.4 e quella non compesata.

Come si può osservare in seguente figura 3.37, la differenza è sostanziale:

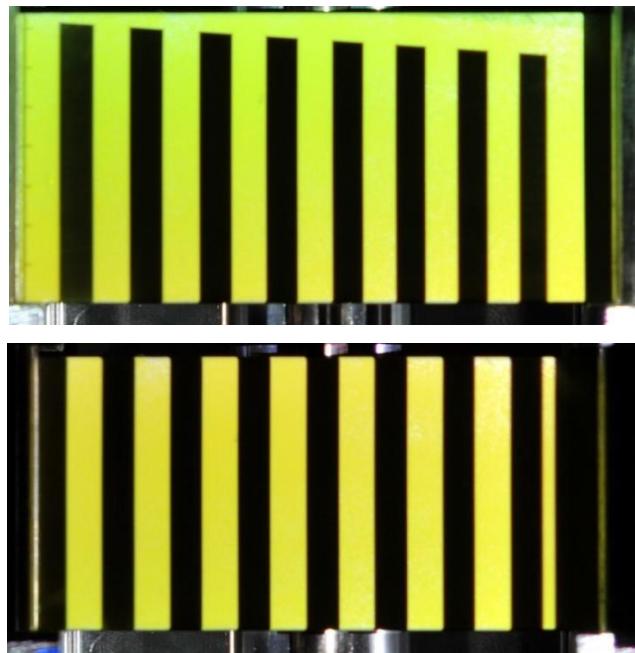


Fig. 3.37-Immagini ottenute a seguito della proiezione di griglia non compensata (immagine in alto), e quella compensata (immagine in basso).

Al fine di determinare quindi l'entità della variazione nello spessore delle frange a seguito di uno spostamento fuori dal piano si procede nella seguente maniera: a provino completamente scarico si acquisiscono due immagini proiettando le griglie compensate, riportanti la figura del provino in due posizioni caratterizzate da uno spostamento relativo pari a 5 [mm], misurato lungo la direzione dell'asse ottico della fotocamera.

Le immagini utilizzate sono riportate in seguente figura 3.38:

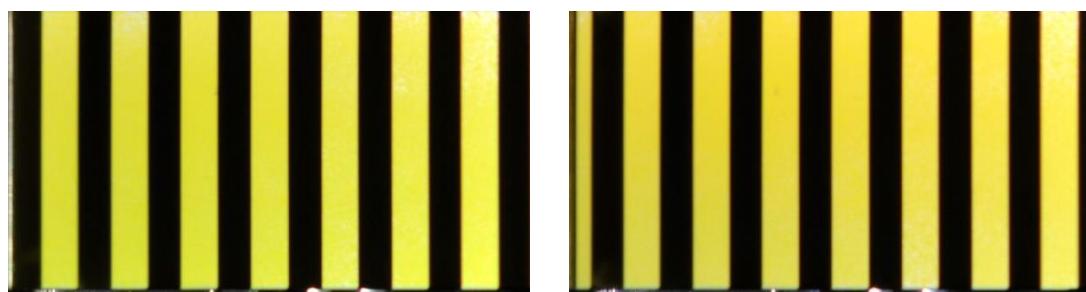


Fig. 3.38-Immagini del provino acquisite a seguito di uno spostamento relativo di 5[mm].

Relativamente alla figura 3.38 si osserva come uno spostamento rigido del piano verso il centro di proiezione della fotocamera generi sia una variazione della distribuzione delle frange, che risulta tanto più marcata quanto è maggiore l'angolo di proiezione (nel

nostro caso pari a 30°), sia una variazione dello spessore che si andra ora a quantificare in maniera precisa.

Procedendo perciò al calcolo degli spessori della prima immagine risulta opportuno in via preliminare effettuare una binarizzazione dell'immagine, ottenendo il seguente risultato in figura 3.39:

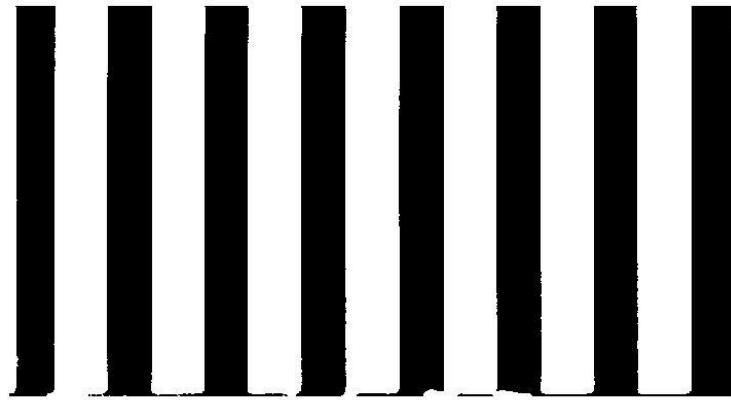


Fig. 3.39-Immagine sinistra di figura 3.38 binarizzata con threshold normalizzato pari a 70/255..

Successivamente effettuata una convoluzione mediante kernel sobeliano per ottenere i contorni si ottiene il seguente risultato di figura 3.40:

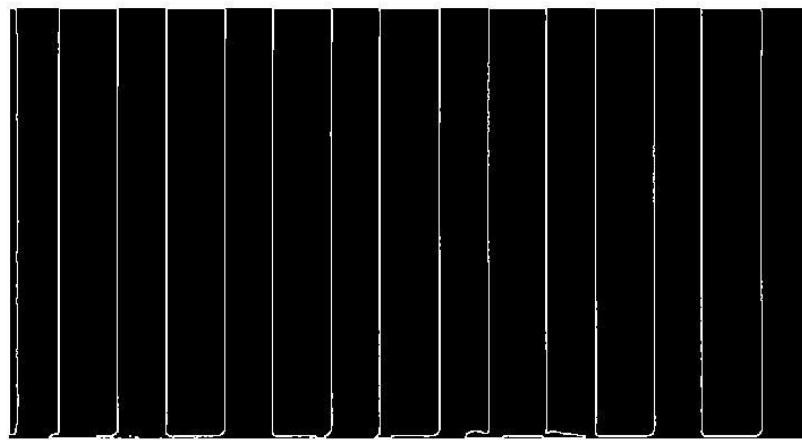


Fig. 3.40-Risultato del calcolo di edge detection con kernel sobeliano applicato all'immagine binarizzata di figura 3.39..

Ottenuti perciò i contorni delle frange si è proceduto al calcolo degli spessori lungo una linea dell’immagine 3.40.

I risultati sono riportati in seguente figura 3.41:

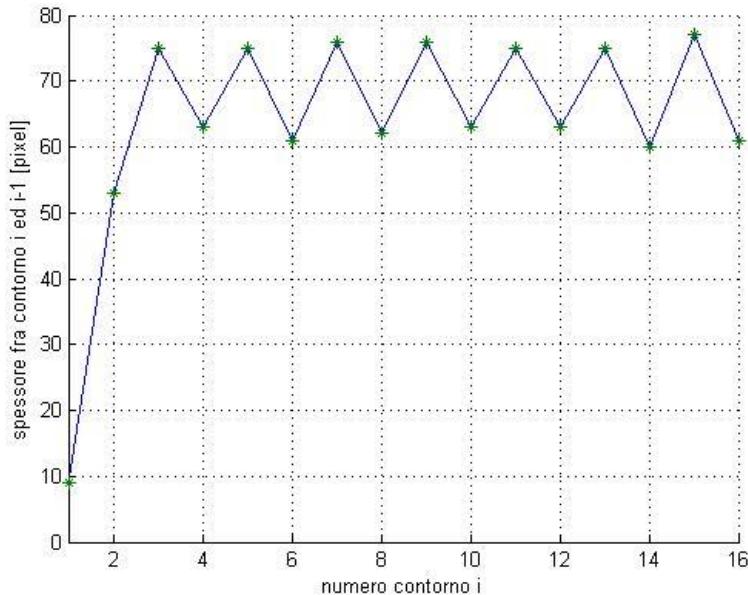


Fig. 3.41-Andamento degli spessori della frangia lungo la direzione orizzontale dell’immagine 3.40.

Come si può notare in figura 3.41, non considerando i valori iniziali relativi al contorno di una frangia tagliata fuori dall’immagine, risulta una certa oscillazione nei valori degli spessori: questo fatto è principalmente legato alla binarizzazione effettuata per semplice thresholding non iterativo e non rappresenta un problema in caso si applichi lo stesso calcolo per l’analisi della seconda immagine.

Effettuando lo stesso calcolo ora per la seconda immagine si ottiene una forma binarizzata come riportato in figura 3.42 seguente:



Fig. 3.42-Immagine destra della figura 3.38 binarizzata con threshold normalizzato pari a 70/255..

La ricerca dei contorni fornisce in questo caso il seguente risultato:

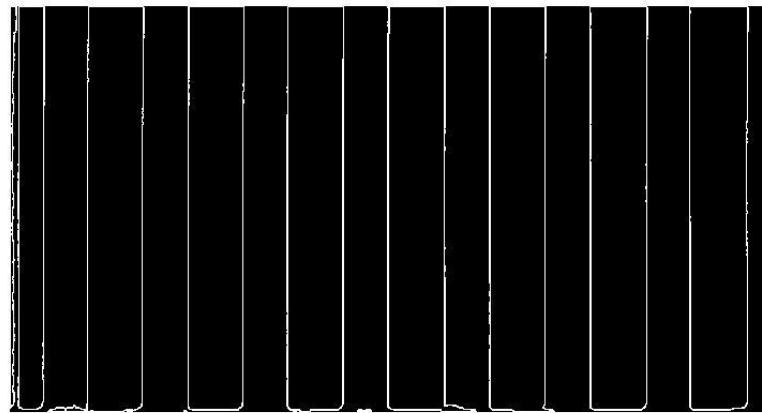


Fig. 3.43-Risultato del calcolo di edge detection con kernel sobeliano applicato all'immagine binarizzata di figura 3.42..

Si ottiene alla fine del calcolo degli spessori il seguente risultante in figura 3.44:

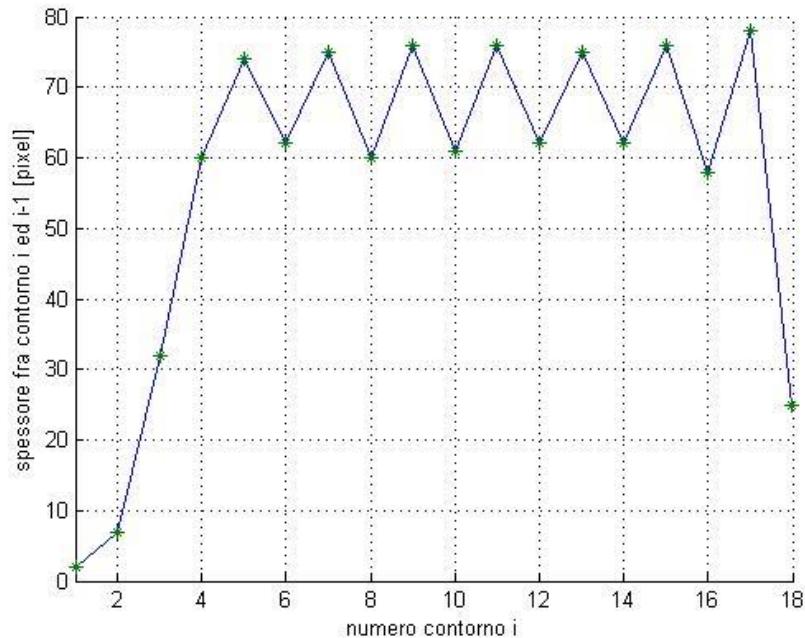


Fig. 3.44-Andamento degli spessori della frangia lungo la direzione orizzontale dell'immagine 3.43.

Procedendo ora con il calcolo dello spessore medio per le due distribuzioni di figure 3.40 e 3.43, senza considerare per ovvi motivi i valori falsi legati a margini tagliati fuori e rumore dovuto alla binarizzazione, si ottiene per la prima immagine un valor medio dello spessore pari a:

$$\text{Mean}_1 = 68.714 \text{ [pixel]}$$

Con deviazione standard pari ad:

$$\text{St. dev}_1 = 7.184 \text{ [pixel]}$$

Per la seconda immagine invece si hanno i seguenti valori:

$$\text{Mean}_2 = 68.215 \text{ [pixel]}$$

$$\text{St. dev}_2 = 7.895 \text{ [pixel]}$$

Per quanto riguarda i valori trovati risulta chiaro che il valore della deviazione standard dipende direttamente dal metodo di binarizzazione utilizzato, mentre per i valori medi come aspettato si ha una variazione dello spessore medio che risulta pari a circa 0,5 pixel. Si procede perciò a questo punto con l'applicazione del metodo a phase shifting e DIC2D.

3.2.2 Applicazione del phase shifting a proiezione di frange con DIC2D

Si riporta anzitutto in seguente figura 3.44 lo schema del setup sperimentale utilizzato nella prova:

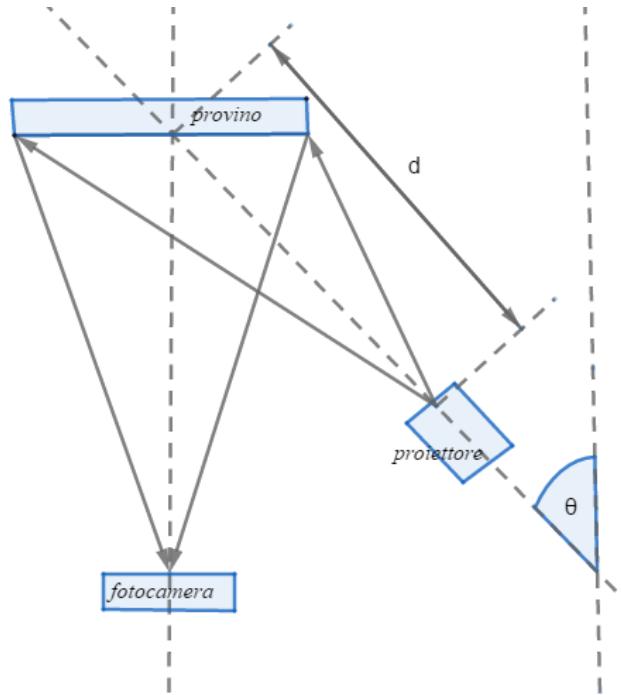


Fig. 3.45-Schema del setup utilizzato nei calcoli: 'd' pari alla distanza di proiezione, θ pari all'angolo di proiezione.

L'angolo di proiezione utilizzato risulta pari a 30° e la distanza di proiezione pari a 290 [mm].

Il provino utilizzato invece è in acciaio di forma rettangolare e superficie di dimensioni 100[mm] x 30 [mm] e spessore di circa 3 mm, come risportato in figura 3.46:

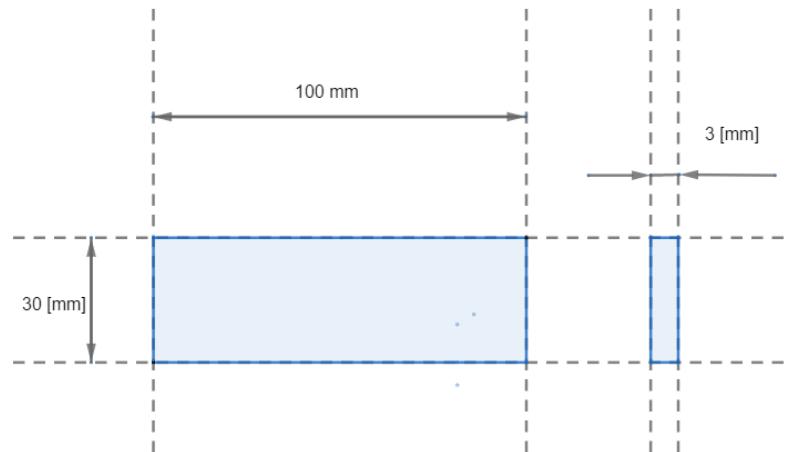


Fig. 3.46-Caratteristiche geometriche del provino analizzato.

La prova effettuata consiste nella flessione a tre punti con carico applicato in mezzeria secondo il seguente schema di figura 3.47:

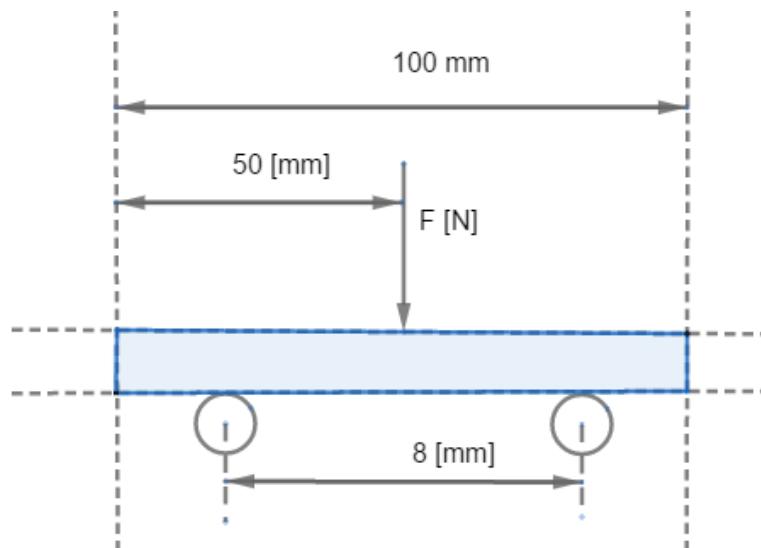


Fig. 3.47-Sistema di carico del provino.

In relazione alla figura 3.47 per quanto riguarda l'effettiva prova effettuata sul provino, risulta opportuno precisare che si è applicato uno spostamento di 1,5 millimetri in mezzeria misurato mediante una vite con scala millimetrica.

Si riporta di seguito un'immagine reale dell'intero setup:

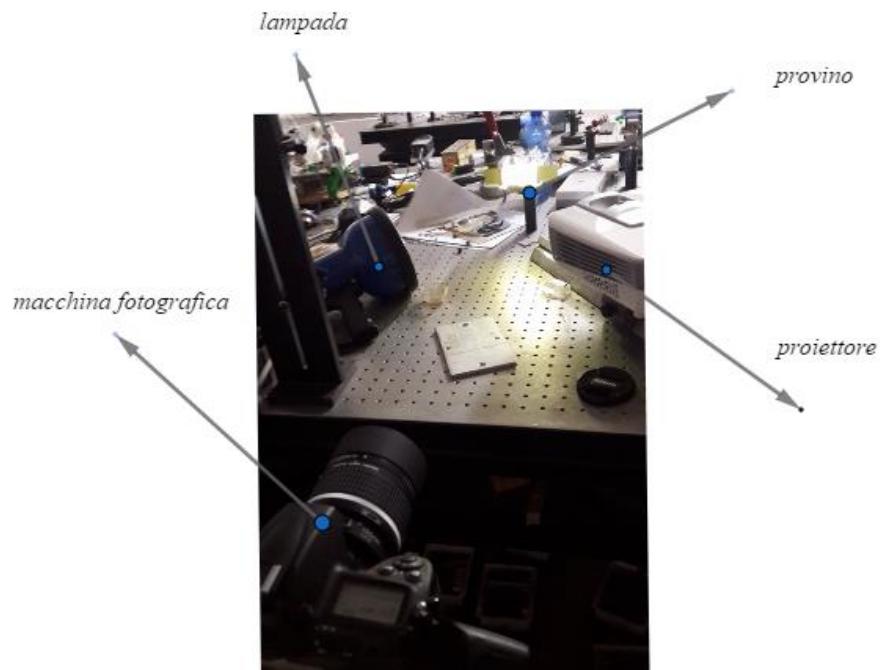


Fig. 3.48-Schema del setup complessivo utilizzato nella prova..

Si procede in primo luogo quindi alla costruzione di una griglia compensata secondo gli angoli di proiezione e la distanza fra il piano del provino e il proiettore.

Al fine di ottenere una distribuzione più uniforme dei livelli d'intensità si è inoltre applicata una legge di tipo sinusoidale alle intensità della singola frangia e utilizzato un numero complessivo di 50 frange.

Il reticolo proiettato utilizzando una dimensione di 420 x 420 pixel per la finestra di proiezione è riportato in seguente figura 3.49:

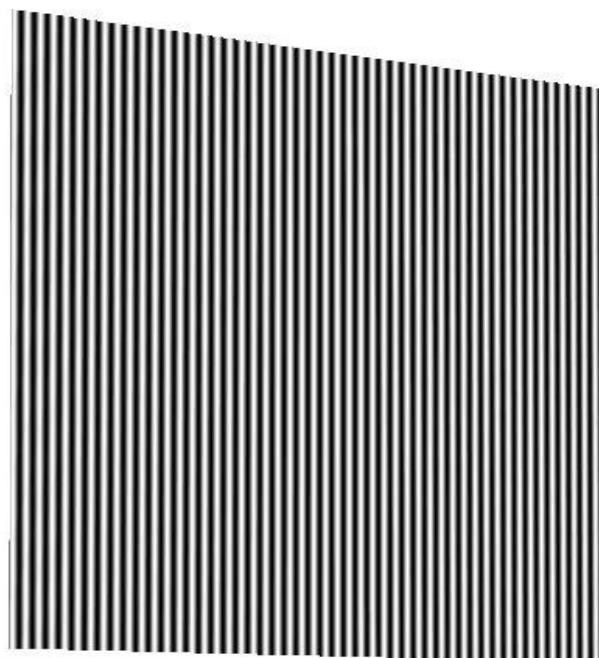


Fig. 3.49-Griglia sinusoidale compensata secondo gli angoli di proiezione utilizzati.

Relativamente alla figura 3.49 si riporta di seguito anche la variazione d'intensità lungo una linea orizzontale della griglia stessa:

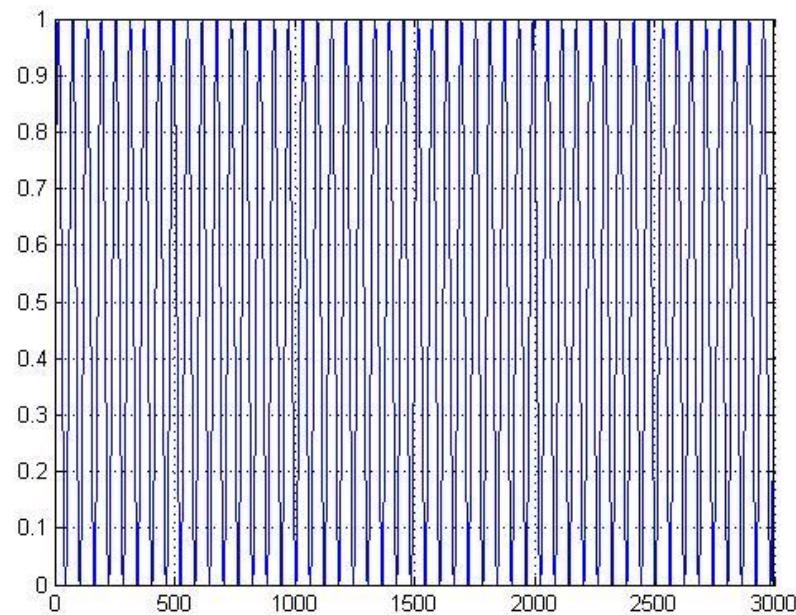


Fig. 3.50- Distribuzione dell'intensità luminosa lungo una riga della griglia di proiezione.

Procedendo perciò che dalla proiezione della griglia appena costruita sul provino si ottiene la seguente immagine di figura 3.51:

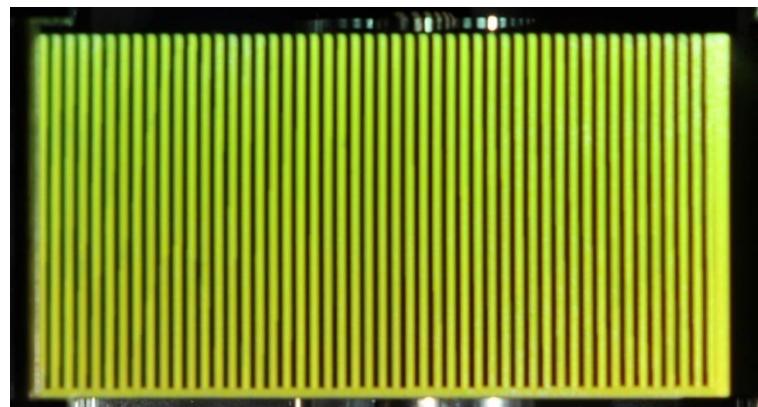


Fig. 3.51-Immagine risultante dalla proiezione della griglia di figura 3.49 sul provino in esame.

Si proiettano a questo punto quattro immagini con sfasamenti imposti di $\pi/2$ sia sul provino scarico che in configurazione deformata, e si procede quindi al calcolo della fase per le due serie d'immagini.

Il calcolo della fase effettuato sul provino scarico genera la seguente immagine di figura 3.52 riportante la distribuzione della fase frazionaria:

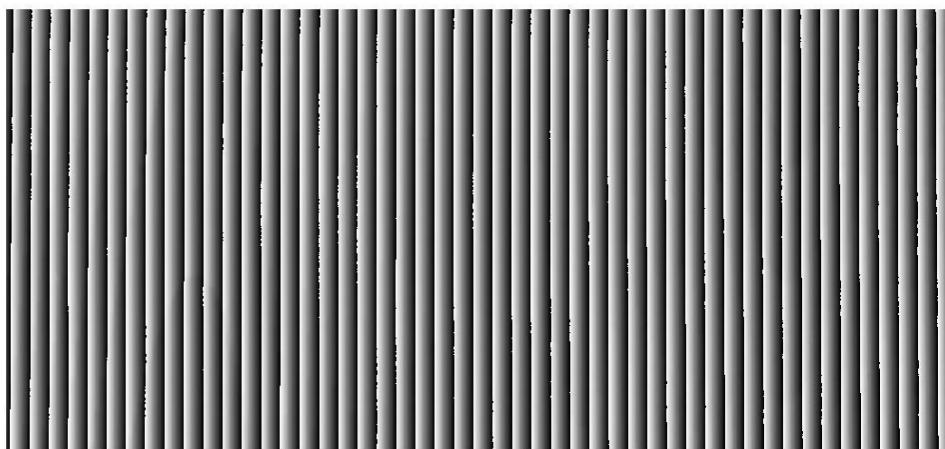


Fig. 3.52-Immagine risultante dal calcolo della fase frazionaria sul provino scarico.

In maniera simile, il calcolo effettuato sulla configurazione deformata produce i seguenti risultati di figura 3.53:

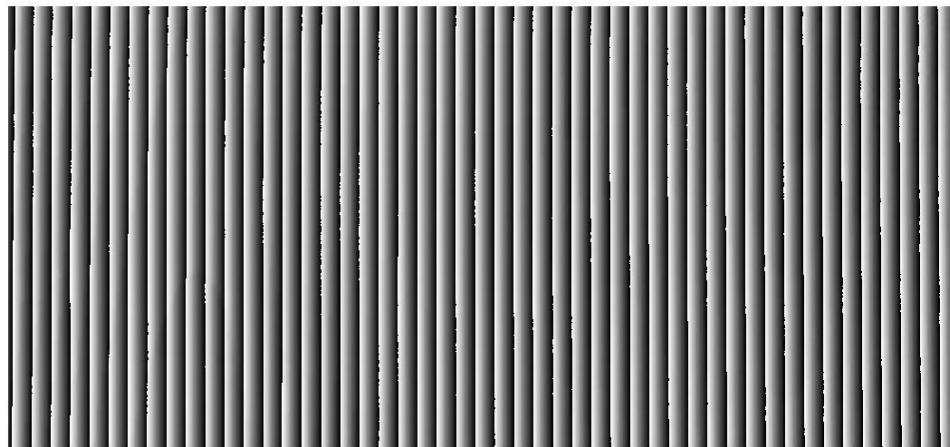


Fig. 3.53-Immagine risultante dal calcolo della fase frazionaria sul provino deformato.

Risulta chiara la necessità a questo punto dei calcoli di effettuare il calcolo dell'unwrapping per l'ottenimento della fase intera delle due configurazioni: per tale calcolo si è utilizzato l'algoritmo basato sulla crescita competitiva di regioni riportato in [24].

Risulta a seguito dell'unwrapping il seguente risultato riportato nelle figure 3.54 e 3.55:



Fig. 3.54-Immagine risultante dal calcolo del phase unwrapping sulla fase frazionaria del provino scarico.



Fig. 3.55-Immagine risultante dal calcolo del phase unwrapping sulla fase frazionaria del provino deformato.

Con i seguenti andamenti della fase intera misurati lungo una linea orizzontale posta a 200 pixel lungo la direzione y per le due immagini di figura 3.54 e 3.55 e riportati nelle figure 3.56, 3.57:

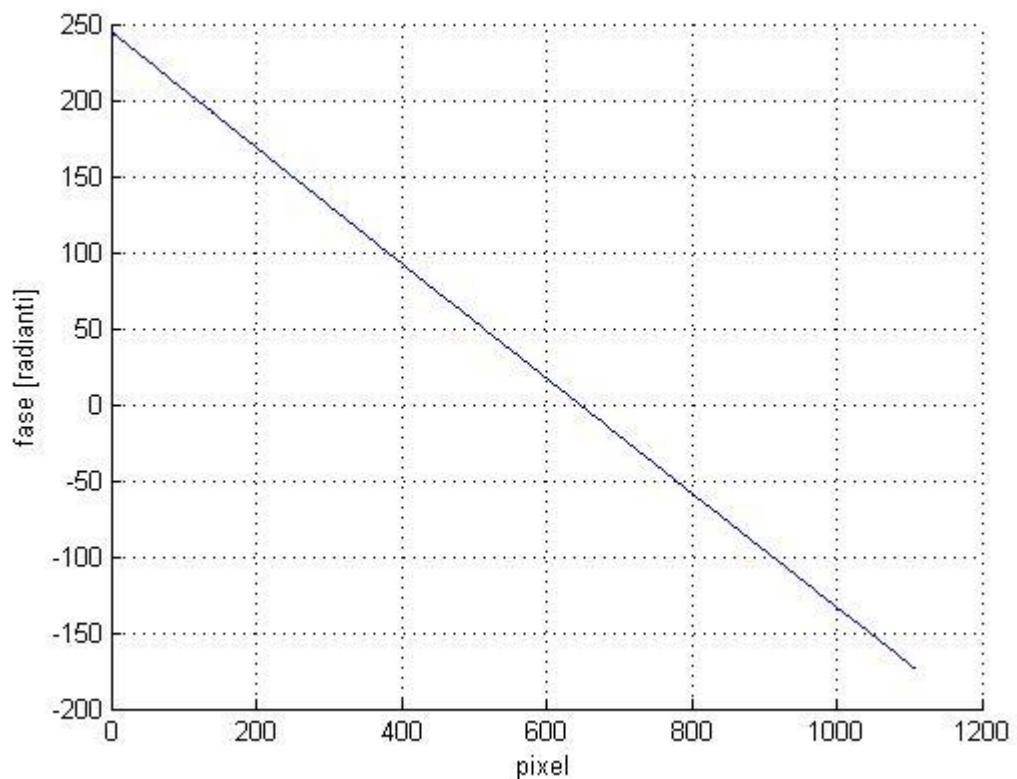


Fig. 3.56-Andamento della fase intera lungo una linea orizzontale per la configurazione a provino scarico.

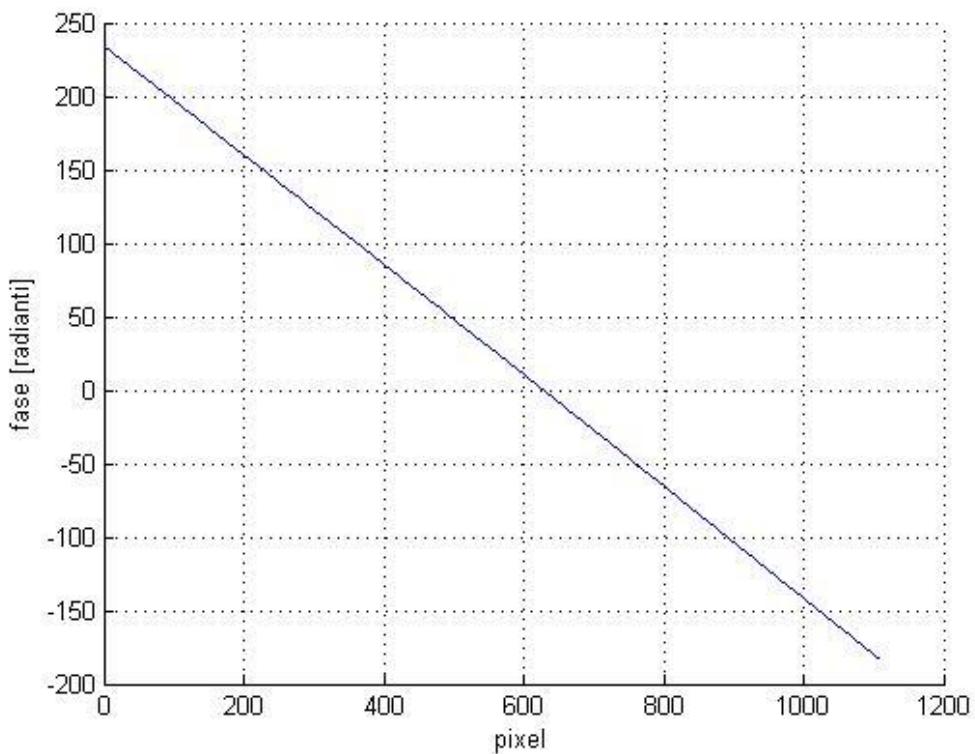


Fig. 3.57- Andamento della fase intera lungo una linea orizzontale per la configurazione a provino caricato.

Si procede perciò alla sottrazione puntuale dei valori delle fasi intere di figura 3.56 e 3.57 al fine di analizzare i risultati.

Si ottiene dal calcolo sopracitato il seguente risultato in figura 3.58:

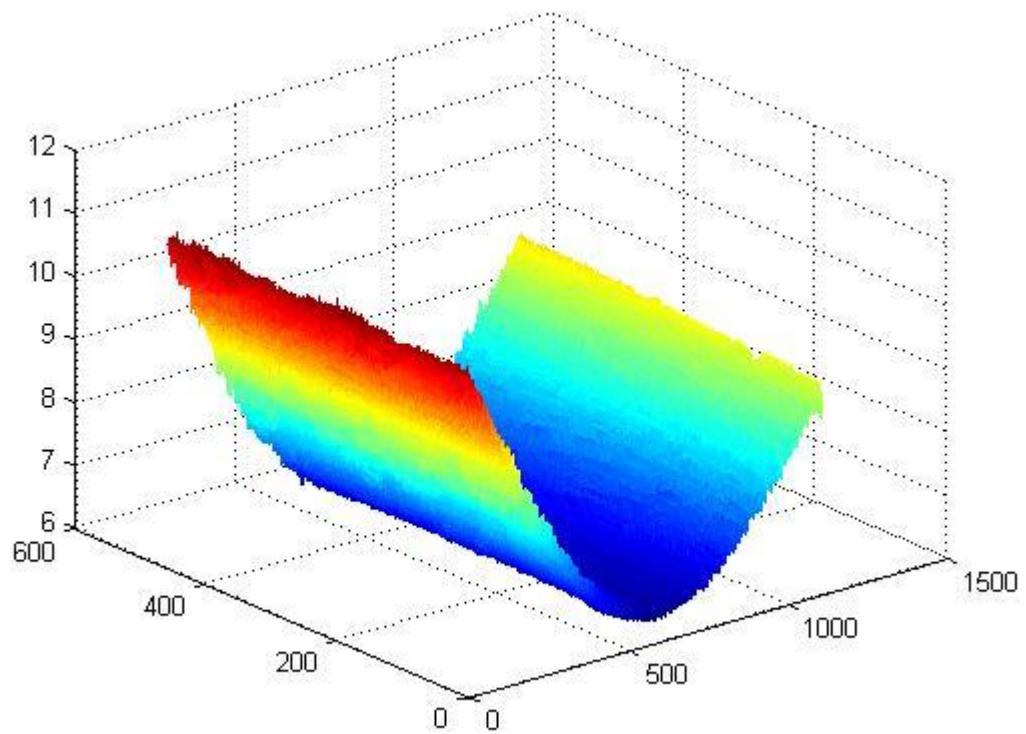
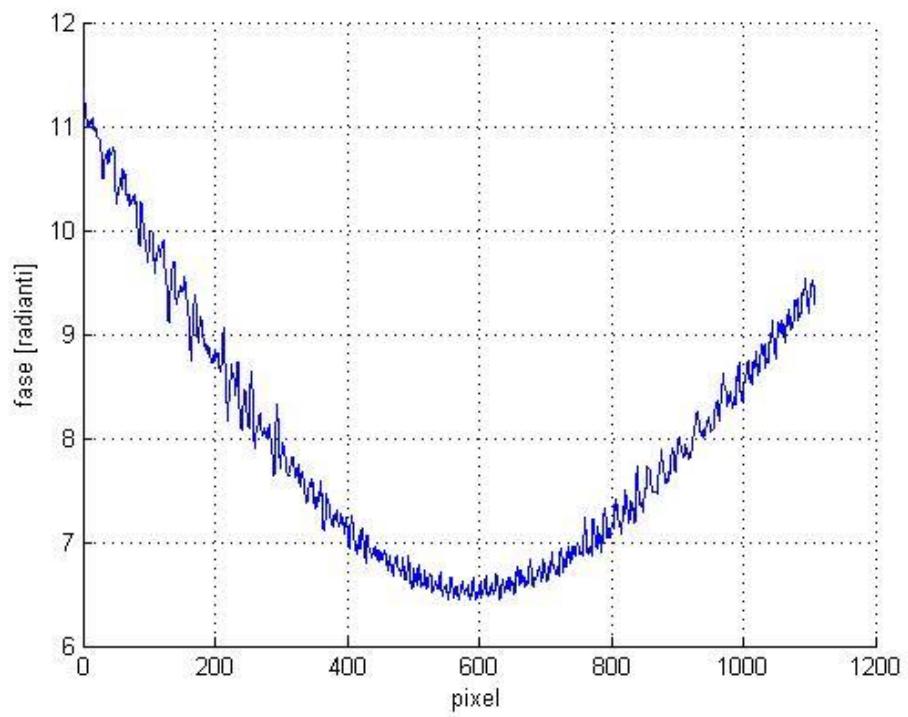


Fig. 3.58- Risultato della differenza delle fasi intere per provino scarico e deformato: in alto l'andamento della differenza lungo una linea, in basso il risultato della sottrazione delle matrici delle fasi intere.

Come si può notare in figura 3.58, il risultato del calcolo effettuato è fortemente soggetto a rumore, tuttavia, il fatto che i valori sembrano oscillare attorno ad un valore medio suggerisce l'applicazione di una denoisificazione di tipo gaussiano.

Si imposta quindi un ciclo di cinquanta convoluzioni gaussiane a deviazione standard unitaria utilizzando un kernel di dimensioni 23×23 .

Come si può notare in seguente figura 3.59: sebbene il risultato della denoisificazione permette effettivamente di eliminare in maniera robusta il rumore presente nei risultati di figura 3.58, si ha la perdita dei valori competenti ad alcuni punti posti agli estremi, fatto legato a quanto affermato al paragrafo 2.1.2 per il quale non risulta esserci per questi punti un *neighborhood* idoneo ad effettuare la convoluzione.

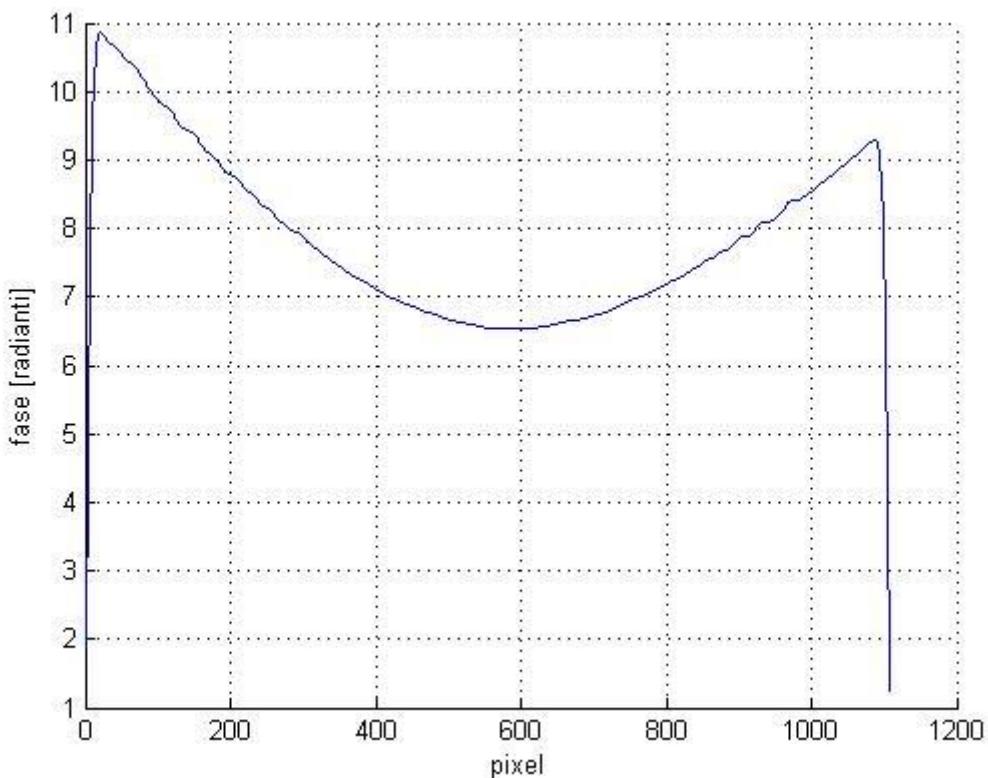


Fig. 3.59- Risultato della differenza delle fasi intere su una riga per provino scarico e deformato a seguito della denoisificazione gaussiana.

Tralasciando per semplicità lo zero padding o la ripetizione ciclica del vettore si decide di eliminare i punti in questione.

Rieffettuando quindi un ciclo di 180 convoluzioni con kernel gaussiano delle stesse dimensioni, si ha la perdita di 30 pixel per ogni estremo, tuttavia si riesce ad ottenere il seguente risultato che mostra le stesse zone nella distribuzione denoisificata e quella rumorosa:

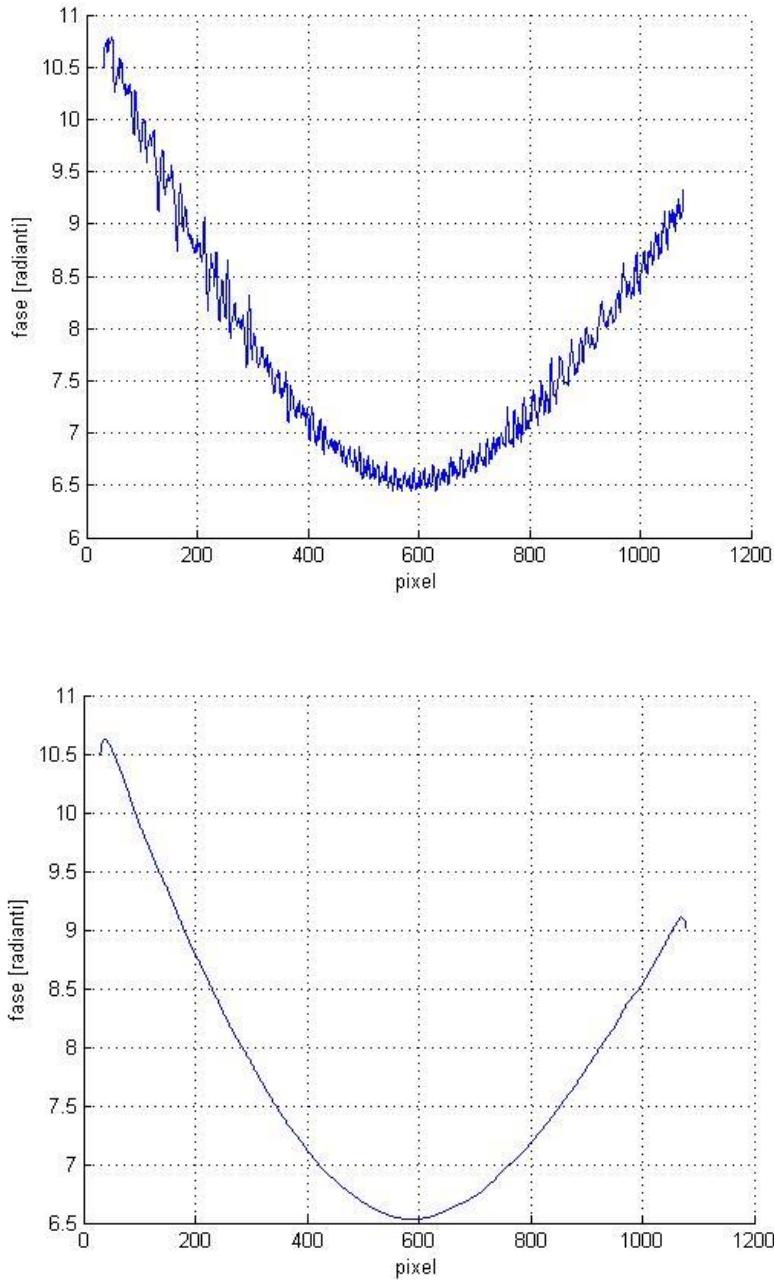


Fig. 3.60- Risultato della differenza delle fasi intere su una riga per provino scarico e deformato a seguito della denoisificazione gaussiana: in alto il segnale rumoroso, in basso il medesimo segnale a seguito del processo di denoisificazione.

Si decide perciò di applicare quanto visto all'intera matrice contenente la differenza delle fasi e si riportano di seguito i risultati in figura 3.61:

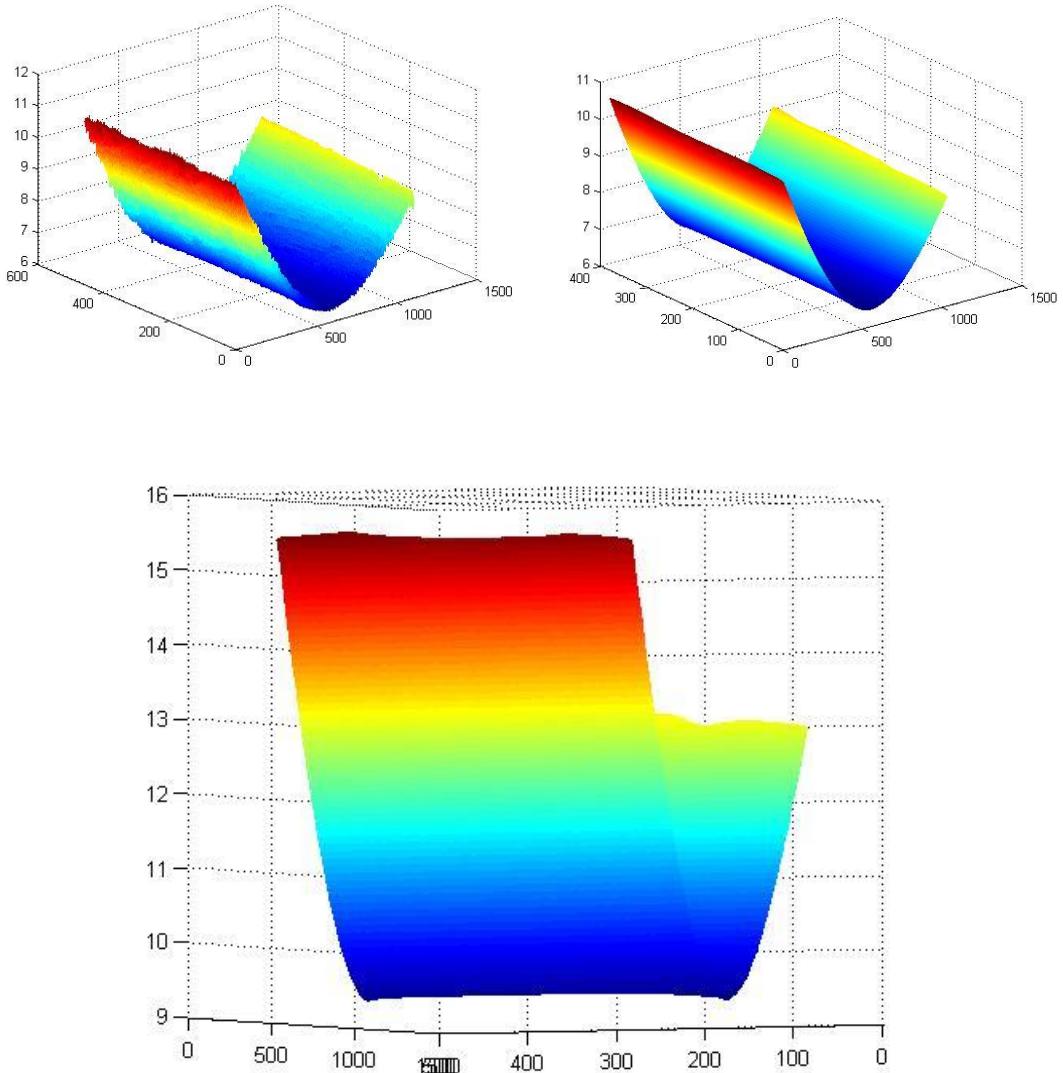


Fig. 3.61- Risultato della ricostruzione della forma del provino deformato a seguito della denoisificazione.

Come si può notare dalla figura 3.61 risulta in definitiva sia un livello contenuto di postprocessing necessario all'ottenimento di una superficie liscia, sia una quantità esigua di informazione persa sul risultato finale, avendo dovuto eliminare un contorno di soli 30 pixel su un'immagine di dimensioni 460 x 1108 pixel.

Analizzato il problema della ricostruzione della forma si affronta ora l'applicazione del metodo della correlazione digitale d'immagine.

Risulta in questo caso che lo spostamento imposto di 1,5 [mm] in mezzeria è stato suddiviso in quattro passi, ottenendo quattro immagini per l'applicazione del DIC2D. Come precedentemente affermato poi si è utilizzato uno speckle in vernice fluorescente: solo in caso in cui il provino venga illuminato con una lampada UV risulta visibile lo speckle pattern per la correlazione.

Si riporta perciò di seguito un'immagine della superficie analizzata del provino riportante lo speckle in vernice fluorescente e illuminato in luce ultravioletta:

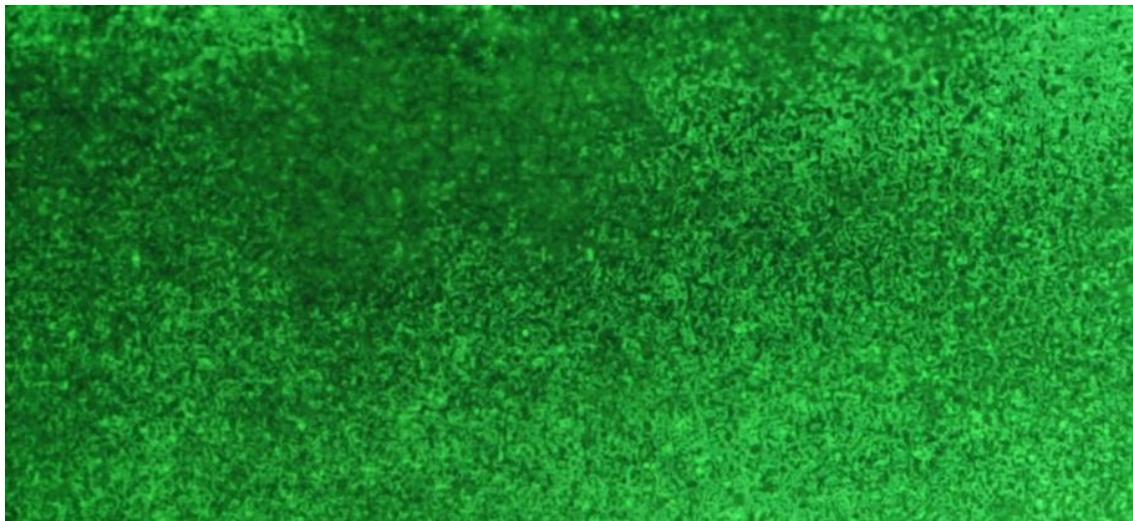


Fig. 3.62- Immagine della superficie del provino illuminato in luce ultravioletta.

Procedendo con il calcolo della correlazione digitale si ottengono i seguenti risultati per gli spostamenti orizzontali u , e quelli verticali v :

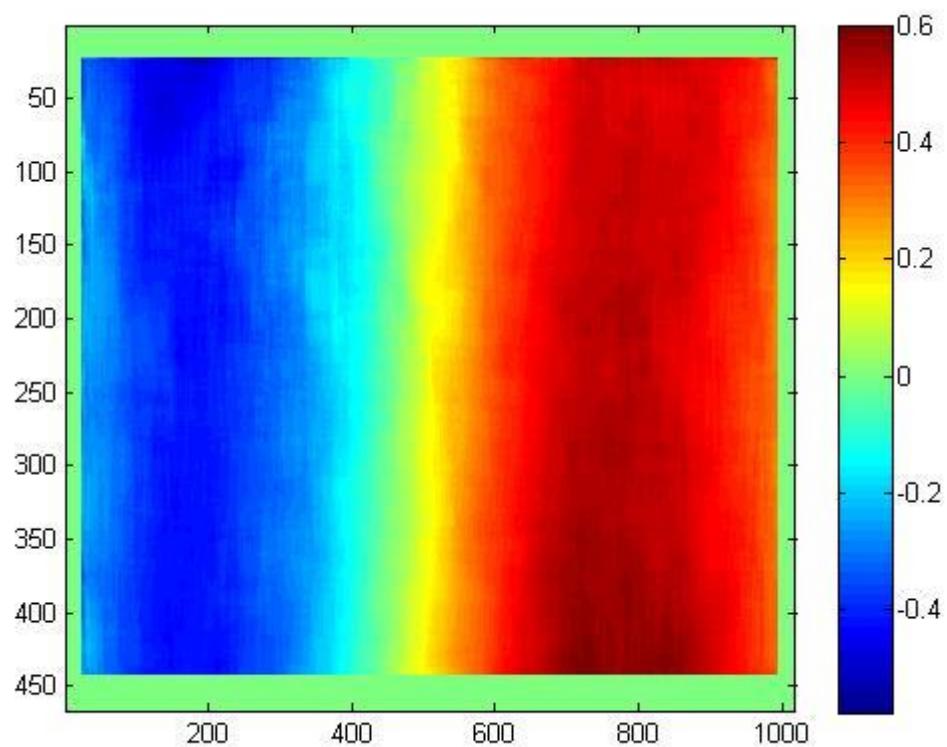


Fig. 3.63- Mappa degli spostamenti ‘ u ’ orizzontali calcolati dal DIC.

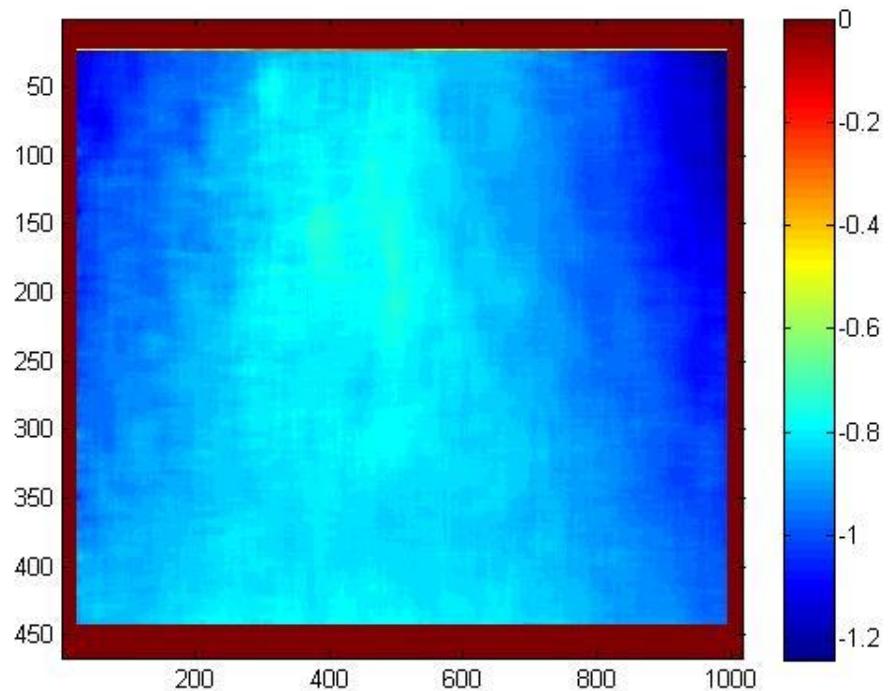


Fig. 3.64- Mappa degli spostamenti ‘ v ’ verticali calcolati dal DIC.

In definitiva risulta quindi determinato lo spostamento tridimensionale dei punti superficiali del provino analizzato.

Volendo infatti effettuare la conversione dei valori delle z in figura 3.61 da radianti a millimetri, si ha che note le dimensioni della superficie del provino oggetto della proiezione (circa 100 [mm]) e il numero di frange proiettate (pari a 50), il passo P_x della griglia sul piano è facilmente calcolabile e risulta pari a 2 [mm], avendo proiettato una griglia compensata.

A questo punto tramite la seguente proporzione si ottiene il fattore di conversione da radianti a millimetri:

$$2\pi:P_x = (\varphi_{\text{piano}} - \varphi_{\text{oggetto}}):x$$

$$x = \frac{P_x \cdot (\varphi_{\text{piano}} - \varphi_{\text{oggetto}})}{2\pi} \quad (3.2.1)$$

Applicando la (3.2.1) ai valori della differenza di fase di figura 3.61 si ottiene il risultato riportato in seguente figura 3.65:

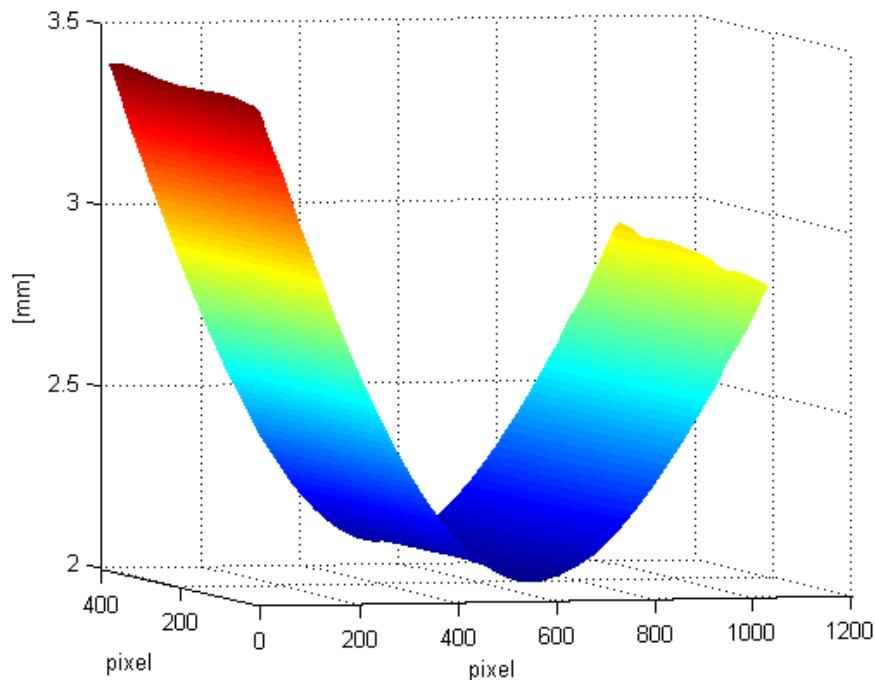


Fig. 3.65- Risultato della ricostruzione della forma del provino deformato.

il quale risulta in buon accordo con lo spostamento imposto, trascurando il bias legato all'incertezza sulla fase iniziale, e l'arrotondamento del valore del valore del passo P_x utilizzato nel calcolo, si noti infatti in figura 3.51 che non tutta la superficie del provino risulta effettivamente coperta dalla proiezione delle frange.

Determinato lo spostamento w dei punti, i valori rappresentati nelle figure 3.64 e 3.63, permettono di quantificare lo spostamento u, v fornendo in definitiva tutti gli spostamenti cercati.

Come precedentemente affermato risulta tuttavia che il calcolo sviluppato fino a questo punto rappresenta solo in maniera qualitativa, dal punto di vista della coerenza fisica e della rumorosità, i risultati finali che il metodo è in grado di fornire.

CONCLUSIONI E FUTURI SVILUPPI

A conclusione dell'intera trattazione sviluppata nei precedenti capitoli risulta chiara la possibilità di percorrere vie alternative alla correlazione digitale d'immagine tridimensionale sviluppata secondo il modello di visione stereoscopica: si ricorda infatti che il metodo a phase shifting con proiezione di frangia e DIC2D con compensazione degli spostamenti apparenti, abbinato all'utilizzo di apparecchiature telecentriche risolve in maniera esatta il problema della determinazione degli spostamenti tridimensionali, e non risente inoltre dei forti limiti imposti sugli angoli di osservazione del sistema stereo rig.

In quest'ottica la trattazione sviluppata si è concentrata sull'obbiettivo di generalizzare quanto più possibile il metodo sopracitato in maniera da permetterne l'applicazione anche con le normali attrezzature commerciali.

L'analisi qualitativa effettuata al paragrafo 3.2 suggerisce quindi l'opportunità di poter sviluppare ulteriormente la trattazione, al fine di effettuare un'analisi comparativa in primo luogo con i risultati teorici attesi e quindi con le soluzioni offerte dai software commerciali in circolazione.

Bibliografia

Modello e calibrazione fotocamera:

- [1] Faugeras, Olivier. (1993). Three-dimensional computer vision: a geometric viewpoint. MIT press. capitolo 3.
- [2] Multiple View Geometry in Computer Vision, Richard Hartley, Andrew Zisserman Cambridge University Press, 25 mar 2004. capitolo 4.
- [3] Z. Zhang. A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research, 1998.
- [4] Zhang's Camera Calibration Algorithm: In-Depth Tutorial and Implementation, Technical Report HGB16-05, 16th May, 2016, Wilhelm Burger, University of Applied Sciences Upper Austria.
- [5] E.g. M. A. Sutton, J.-J. Orteu, H. W. Schreier, Book - Image Correlation for Shape, Motion and Deformation Measurements, Hardcover ISBN 978-0-387-78746-6. Capitolo 3.
- [6] Open source computer vision library (OpenCV) 3.0.0 C++ official documentation on distortion model.

Geometria epipolare e Calibrazione Stereo:

- [7] Multiple View Geometry in Computer Vision, Richard Hartley, Andrew Zisserman Cambridge University Press, 25 mar 2004. Capitoli: 9,11.

- [8] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981
- [9] R. I. Hartley, "In defense of the eight-point algorithm," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 580-593, Jun 1997.doi: 10.1109/34.601246
- [10] Rostam Affendi Hamzah and Haidi Ibrahim, "Literature Survey on Stereo Vision Disparity Map Algorithms," *Journal of Sensors*, vol. 2016, Article ID 8742920, 23 pages, 2016. doi:10.1155/2016/8742920
- [11] Fusiello, A., Trucco, E. & Verri, A. *Machine Vision and Applications* (2000) 12: 16. <https://doi.org/10.1007/s001380050120>
- [12]R. Brunelli, *Template Matching Techniques in Computer Vision: Theory and Practice*, Wiley, ISBN 978-0-470-51706-2, 2009
- [13] Harris, Chris & Stephens, Mike. (1988). A combined corner and edge detector. 50.
- [14] Jianbo Shi and C. Tomasi, "Good features to track," 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, 1994, pp. 593-600.doi: 10.1109/CVPR.1994.323794

Correlazione digitale d'immagine 2D e 3D, phase shifting, unwrapping, chessboard detection:

[15] Bruce D. Lucas and Takeo Kanade. 1981. An iterative image registration technique with an application to stereo vision.In Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2 (IJCAI'81), Vol. 2. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 674-679.

[16] Tomasi, Carlo & Kanade, Takeo. (1991). Detection and Tracking of Point Features. International Journal of Computer Vision. 9. 137-154.

[17] Baker, S. & Matthews, I. International Journal of Computer Vision (2004) 56: 221.
<https://doi.org/10.1023/B:VISI.0000011205.11775.fd>

[18] Berthold K.P. Horn, Brian G. Schunck, Determining optical flow, Artificial Intelligence, Volume 17, Issues 1–3, 1981, Pages 185–203, ISSN 0004-3702,
[https://doi.org/10.1016/0004-3702\(81\)90024-2](https://doi.org/10.1016/0004-3702(81)90024-2).(<http://www.sciencedirect.com/science/article/pii/0004370281900242>)

[19] E.g. M. A. Sutton, J.-J. Orteu, H. W. Schreier, Book - Image Correlation for Shape, Motion and Deformation Measurements, Hardcover ISBN 978-0-387-78746-6. Capitolo 5.

[20] Baldi, A, and Bertolino, F (2015), Experimental Analysis of the Errors due to Polynomial Interpolation in Digital Image Correlation. Strain, 51, 248–263. doi: 10.1111/str.12137

[21] D.W. Robinson and D.T. Reid: Interferogram analysis: Digital Fringe Pattern Measurement Techniques Robinson and G.T. Reid eds, Institute of Physic Publishing, Bristol and Philadelphia, 194–229, 1993.Capitoli 2,4,6

[22] A. Baldi, F. Bertolino, F. Ginesu,"On the performance of some unwrapping algorithms",Optics and Lasers in Engineering,Volume 37, Issue 4,2002,Pages 313-330,ISSN 0143-8166,

[23] A. Baldi, "Two-dimensional phase unwrapping by quad-tree decomposition," *Appl. Opt.* 40, 1187-1194 (2001).

[24] A. Baldi, F. Bertolino, F. Ginesu, "Un nuovo algoritmo di Phase Unwrapping basato sulla crescita competitiva di regioni",4 Contributo in Atti di Convegno (Proceeding)::4.1 Contributo in Atti di convegno 2001

[25] Bouguet, Jean-Yves. (2000). Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm. OpenCV Document, Intel, Microprocessor Research Labs. 1. .

[26] S. N. Tamgade and V. R. Bora, "Motion Vector Estimation of Video Image by Pyramidal Implementation of Lucas Kanade Optical Flow," *2009 Second International Conference on Emerging Trends in Engineering & Technology*, Nagpur, 2009, pp. 914-917. doi: 10.1109/ICETET.2009.154

[27] M. Malesa, K. Malowany, U. Tomczak, B. Siwek, M. Kujawinska, A. Sieminska-LewandowskaApplication of 3d digital image correlation in maintenance and process control in industry Computers in Industry, 64 (2013), pp. 1301-1315

[28] Z. Tang, J. Liang, Z. Xiao, Large deformation measurement scheme for 3d digital image correlation method, *Optics and lasers in engineering*, 50 (2012), pp. 122-130

[29] Y. Zhang, G. Li, X. Xie and Z. Wang, "A new algorithm for accurate and automatic chessboard corner detection," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, 2017, pp. 1-4. doi: 10.1109/ISCAS.2017.8050637

[30] Y. Liu, S. Liu, Y. Cao and Z. Wang, "A practical algorithm for automatic chessboard corner detection," *2014 IEEE International Conference on Image Processing (ICIP)*, Paris, 2014, pp. 3449-3453.doi: 10.1109/ICIP.2014.7025701

[31] Becker, Thomas & Splitthof, Karsten & Siebert, Thorsten & Kletting, Peter. (2006). Error estimations of 3D digital image correlation measurements. 63410F-63410F. 10.11117/12.695277.

*Un Ringraziamento a tutti coloro
che mi hanno sostenuto in questo
lavoro di tesi.*

Appendice codici:

Codice opencv di calibrazione per singola fotocamera:

```
#include <iostream>
#include <sstream>
#include <time.h>
#include <stdio.h>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/highgui/highgui.hpp>
using namespace std;
using namespace cv;
vector< vector< Point2f > > vettore_punti_immagine;
vector< vector< Point3f > > vettore_punti_oggetto;
vector< Point2f > corners;
vector <Mat> serie_img;
Mat img, gray,undistorted;
Size im_size;
int avanti=0;
void setup_calibration(int larghezza_schacchiera, int altezza_schacchiera, int num_imgs,
                      float dimensione_quadrato, char* imgs_filename,
                      char* extension) {
    Size board_size = Size(larghezza_schacchiera, altezza_schacchiera);
    int board_n = larghezza_schacchiera * altezza_schacchiera;
    for (int k = 1; k <= num_imgs; k++) {
        char img_file[300];
        sprintf(img_file, "%s (%d)%s", imgs_filename, k, extension);
        img = imread(img_file, CV_LOAD_IMAGE_COLOR);
        serie_img.push_back(img);
        if(! img.data ) // Check for invalid input
```

```

{
    cout << "immagini? bohhh" << std::endl ;

    waitKey();
    exit(1);
}

cv::cvtColor(img, gray, CV_BGR2GRAY);

bool found = false;

found = cv::findChessboardCorners(img, board_size, corners,CALIB_CB_ADAPTIVE_THRESH|CALIB_CB_FILTER_QUADS);

if(found==false){

    cout<<"non riesco a trovare una griglia "<<larghezza_schacchiera<<" x
" <<altezza_schacchiera<<endl;

}
else if (found)

{

    cornerSubPix(gray, corners, cv::Size(5, 5), cv::Size(-1, -1), TermCriteria(CV_TERMCRIT_EPS | CV_TERMCRIT_ITER, 30, 0.1));

    drawChessboardCorners(img, board_size, corners, found);

}

vector< Point3f > obj;

for (int i = 0; i < altezza_schacchiera; i++)

    for (int j = 0; j < larghezza_schacchiera; j++)

        obj.push_back(Point3f((float)j * dimensione_quadrato, (float)i * dimensione_quadrato, 0));

if (found) {

    cout<<"immagine: "<<k << endl;

    vettore_punti_immagine.push_back(corners);

    vettore_punti Oggetto.push_back(obj);

    namedWindow("Display window2",WINDOW_AUTOSIZE );

    imshow( "Display window2", img );

    sprintf(img_file, "griglia (%d)%s",k, extension);

    imwrite( img_file, img );
}

```

```

        waitKey();
    }

}

}

double computeReprojectionErrors(const vector< vector< Point3f > >& punti_oggetto,
                                const vector< vector< Point2f > >& punti_immagine,
                                const vector< Mat >& rvecs, const vector< Mat >& tvecs,
                                const Mat& matrice_calibrazione , const Mat& coefficienti_distorsione) {
    vector< Point2f > punti_immagine2;
    int i, totalPoints = 0;
    double totalErr = 0, err;
    vector< float > perViewErrors;
    perViewErrors.resize(punti_oggetto.size());
    for (i = 0; i < (int)punti_oggetto.size(); ++i) {
        projectPoints(Mat(punti_oggetto[i]), rvecs[i], tvecs[i], matrice_calibrazione,
                      coefficienti_distorsione, punti_immagine2);
        err = norm(Mat(punti_immagine[i]), Mat(punti_immagine2), CV_L2);
        int n = (int)punti_oggetto[i].size();
        perViewErrors[i] = (float) std::sqrt(err*err/n);
        totalErr += err*err;
        totalPoints += n;
    }
    return std::sqrt(totalErr/totalPoints);
}

int main(int argc, char const **argv)
{
    int punti_orizz=9;
    int punti_vert=6;
    int num_imgs;
    float lato_quadrato=20.00;//mm
}

```

```

char nome_imgs[10];
char file_salvataggio[30];
char estensione[10];
float errore_riproj;
cout<<"inserisci numero corner orizzontali";
cin>>punti_orizz;
cout<<"inserisci numero corner verticali";
cin>>punti_vert;
cout<<"inserisci nome serie immagini:"<<endl;
cin>>&nome_imgs[0];
cout<<"inserisci estensione immagini:"<<endl;
cin>>&estensione[0];
cout<<"inserisci numero immagini"<<endl;
cin>>num_imgs;
cout<<"inserisci dimensione lato [mm]"<<endl;
cin>>lato_quadrato;
cout<<"inserisci file salvataggio"<<endl;
cin>>&file_salvataggio[0];
int c;
setup_calibration(punti_orizz, punti_vert, num_imgs, lato_quadrato, nome_imgs, estensione);
cout<<"setup_finito, inizio calibrazione"<<endl;
Mat K;
Mat D;
vector< Mat > vR, vT;
int flag = 0;
flag |= CALIB_FIX_K4;
flag |= CALIB_FIX_K5;
calibrateCamera(vettore_punti Oggetto, vettore_punti Immagine, img.size(), K, D, vR, vT,
flag);
errore_riproj=computeReprojectionErrors(vettore_punti Oggetto, vettore_punti Immagine,
vR, vT, K, D);

```

```

cout << "Errore Riproiezione:" << errore_riproi << endl;
cout << "matrice di calibrazione\n:" << K << endl;
FileStorage fs(file_salvataggio, FileStorage::WRITE);
fs << "Matrice_K_di_calibrazione" << K;
fs << "Matrice_D_coefficienti_di_distorsione" << D;
fs << "errore_riproiezione" << errore_riproi;
fs << "numero_punti_orizzontali_griglia" << punti_orizz;
fs << "numero_punti_verticali_griglia" << punti_vert;
fs << "lato_quadrato_mm" << lato_quadrato;
fs << "vettore_traslazione" << vT;
fs << "vettore_rotazione" << vR;
cout << "desideri vedere la undistorsione bella figa??[1/si][2/no]" << endl;
int ans;
cin >> ans;
if (ans == 1)
for (int i = 0; i < num_imgs; i++)
{
    char img_file[200];
    undistort(serie_img[i], undistorted, K, D);
    sprintf(img_file, "undistorted (%d).png", i);
    imwrite(img_file, undistorted);
/*     namedWindow("Display original", WINDOW_AUTOSIZE );
    imshow("Display original", serie_img[i] );
    namedWindow("Display undistort", WINDOW_AUTOSIZE );
    imshow("Display undistort", undistorted );
*/
    waitKey();
}
waitKey();
return 0;

```

```
}
```

Codice harris corners detector:

```
clear all
clear all
dim_fin=9;

foto.imread('2.jpg');
foto1=rgb2gray(foto);
foto1 = im2double(foto1);
Ix=derivata_x(foto1,size(foto1,2),size(foto1,1));
Iy=derivata_y(foto1,size(foto1,2),size(foto1,1));

IxIx=Ix.^2;
IyIy=Iy.^2;
IxIy=Ix.*Iy;

dim_fin=round(dim_fin/2)-1;

    somma_xx= zeros(size(IxIx));
    somma_yy= zeros(size(IyIy));
    somma_xy= zeros(size(IxIy));
for i=dim_fin:size(IxIx,1)-dim_fin

    for j=dim_fin:size(IxIx,2)-dim_fin
        k=1;

        for y=i-(dim_fin-1):i+(dim_fin-1)
            for x=j-(dim_fin-1):j+(dim_fin-1)

                somma_xx(i,j)= somma_xx(i,j) + IxIx(y,x);
                somma_yy(i,j)= somma_yy(i,j) + IyIy(y,x);
                somma_xy(i,j)= somma_xy(i,j) + IxIy(y,x);
            k=k+1;
        end

        end
        M=[somma_xx(i,j) somma_xy(i,j); somma_xy(i,j) somma_yy(i,j)];
        e=eig(M);
        R(i,j)=e(1)*e(2)-0.15*(e(1)+e(2))^2;
        %
        t=e(1)+e(2);
        %
        if(t~=0)
        R(i,j)=e(1)*e(2)/(e(1)+e(2)); % seconda forma errore
        else R(i,j)=0;
        end
        %
        % R(i,j)=e(1); % terza forma errore
    end
end
xs=1;
ys=1;
for i=1:size(R,1)
    for j=1:size(R,2)
        if(R(i,j)>0)
            HC(1, xs)=j;
            HC(2, ys)=i;
            xs=xs+1;
        end
    end
end
```

```

        ys=ys+1;
    else if(R(i,j)<0)
        R(i,j)=0.0;
    end
end
end

figure(1)
imshow(foto1);
hold on;
title('mappa R');
plot(HC(1,:),HC(2,:),'r')
hold off
figure(100)
surf(R,'linestyle','none')

[vett,R_]=local_max1(HC,R,5);

for i=1:19
X=vett(1,:);
Y=vett(2,:);
A=[X;Y]';
[UniXY,Index]=unique(A,'rows');
DupIndex=setdiff(1:size(A,1),Index);
A(DupIndex,:)=[];
X=A(:,1)';
Y=A(:,2)';
A=[X;Y];
[vett,R_]=local_max1(A,R,50);
R=R_.*R;
end
figure(12)
imshow(foto);
hold on;
title('R massimo locale');

plot(A(1,:),A(2,:),'r','MarkerSize',15);
hold off

funzione ricerca massimi locali:
function [ M,P ] = local_max1( HC,R,dim_fin)
dx=round(dim_fin/2)-1;
xs=1;
ys=1;
tx=1;
ty=1;
P=zeros(size(R));
for i=1:size(HC,2)
    xi= HC(1,i);
    yi= HC(2,i);
    temp=R(yi,xi);
    for k=yi-dx:yi+dx
        if(k<size(R,1) && k>0)
            for l=xi-dx:xi+dx
                if(l<size(R,2) && l>0)
                    if(R(k,l)>temp || temp==R(k,l) )
                        temp=R(k,l);
                        P(ty,tx)=1;

```

```

        tx=1;
        ty=k;

        end
    end
end
end
end
end
M(1, xs)=tx;
M(2, ys)=ty;

xs=xs+1;
ys=ys+1;

end
end

```

Codice stereo calibrazione:

```

#include <iostream>
#include <time.h>
#include <stdio.h>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/highgui/highgui.hpp>
using namespace std;
using namespace cv;
vector< vector< Point3f > > object_points;
vector< vector< Point2f > > imagePoints1, imagePoints2;
vector< Point2f > corners1, corners2;
vector< vector< Point2f > > left_img_points, right_img_points;
Mat img1, img2, gray1, gray2, imgcorn1, imgcorn2;
void load_image_points(int board_width, int board_height, int num_imgs, float square_size,
                      char* leftimg_filename, char* rightimg_filename, char* extens) {
    Size board_size = Size(board_width, board_height);
    int board_n = board_width * board_height;
    for (int i = 1; i <= num_imgs; i++) {
        char left_img[400], right_img[400];
        sprintf(left_img, "%s (%d)%s", leftimg_filename, i, extens);

```

```

sprintf(right_img, "%s (%d)%s", rightimg_filename, i,extens);
img1 = imread(left_img, CV_LOAD_IMAGE_COLOR);
img2 = imread(right_img, CV_LOAD_IMAGE_COLOR);

    if(! img1.data )           // Check for invalid input

{
    cout << "immagini? bohhh" << std::endl ;
    system("pause");
    exit(1);

}

    waitKey();

cvtColor(img1, gray1, CV_BGR2GRAY);
cvtColor(img2, gray2, CV_BGR2GRAY);

    imgcorn1=img1;
    imgcorn2=img2;

bool found1 = false, found2 = false;

found1 = cv::findChessboardCorners(gray1, board_size, corners1,
CV_CALIB_CB_ADAPTIVE_THRESH | CV_CALIB_CB_FILTER_QUADS);
found2 = cv::findChessboardCorners(gray2, board_size, corners2,
CV_CALIB_CB_ADAPTIVE_THRESH | CV_CALIB_CB_FILTER_QUADS);

if (found1)

{cout<<"foto"<<i<<endl;

    cv::cornerSubPix(gray1, corners1, cv::Size(5, 5), cv::Size(-1, -1),
cv::TermCriteria(CV_TERMCRIT_EPS | CV_TERMCRIT_ITER, 30, 0.1));

    cv::drawChessboardCorners(imgcorn1, board_size, corners1, found1);

    namedWindow("Display window1",WINDOW_AUTOSIZE );
    imshow( "Display window1", imgcorn1 );

}

if (found2)

{cout<<"foto"<<i+1<<endl;

    cv::cornerSubPix(gray2, corners2, cv::Size(5, 5), cv::Size(-1, -1),
cv::TermCriteria(CV_TERMCRIT_EPS | CV_TERMCRIT_ITER, 30, 0.1));

    cv::drawChessboardCorners(imgcorn2, board_size, corners2, found2);
}

```

```

namedWindow("Display window2",WINDOW_AUTOSIZE );

imshow( "Display window2", imgcorn2 );

}

waitKey();

vector< Point3f > obj;

for (int i = 0; i < board_height; i++)
    for (int j = 0; j < board_width; j++)
        obj.push_back(Point3f((float)j * square_size, (float)i * square_size, 0));

if (found1 && found2) {
    cout << i << ". Found corners!" << endl;
    imagePoints1.push_back(corners1);
    imagePoints2.push_back(corners2);
    object_points.push_back(obj);
}
}

for (int i = 0; i < imagePoints1.size(); i++) {
    vector< Point2f > v1, v2;
    for (int j = 0; j < imagePoints1[i].size(); j++) {
        v1.push_back(Point2f((double)imagePoints1[i][j].x, (double)imagePoints1[i][j].y));
        v2.push_back(Point2f((double)imagePoints2[i][j].x, (double)imagePoints2[i][j].y));
    }
    left_img_points.push_back(v1);
    right_img_points.push_back(v2);
}
}

int main(int argc, char const *argv[])
{
    char leftcalib_file[20];
    char rightcalib_file[20];
    char leftimg_filename[10];
    char rightimg_filename[10];
    char estensione[10];
}

```

```

char* out_file="Risultati_calibrazione_stereo.txt";
int num_imgs;
cout<<"inserisci nome file calibrazione di sinistra:"<<endl;
cin>>&leftcalib_file[0];
cout<<"inserisci nome file calibrazione di destra:"<<endl;
cin>>&rightcalib_file[0];
cout<<"inserisci nome serie immagini di calibrazione di sinistra:"<<endl;
cin>>&leftimg_filename[0];
cout<<"inserisci nome serie immagini di calibrazione di destra:"<<endl;
cin>>&rightimg_filename[0];
cout<<"inserisci estensione immagini:"<<endl;
cin>>&estensione[0];
cout<<"inserisci numero immagini per vista:"<<endl;
cin>>num_imgs;
FileStorage fsl(leftcalib_file, FileStorage::READ);
FileStorage fsr(rightcalib_file, FileStorage::READ);
cout<<"caricato il file dati"<<endl;
Mat K1, K2, R, F, E;
Mat T;
Mat D1, D2;
float dim;
fsl["Matrice_K_di_calibrazione"] >> K1;
fsr["Matrice_K_di_calibrazione"] >> K2;
fsl["Matrice_D_coefficienti_di_distorsione"] >> D1;
fsr["Matrice_D_coefficienti_di_distorsione"] >> D2;
fsr["lato_quadrato_mm"]>>dim;
cout<<dim<<endl;
load_image_points(9, 6, num_imgs, dim, leftimg_filename, rightimg_filename, estensione);
printf("inizio calibrazione\n");
int flag = 0;
flag |= CV_CALIB_FIX_INTRINSIC;
cout << "Read intrinsics" << endl;

```

```

stereoCalibrate(object_points, left_img_points, right_img_points, K1, D1, K2, D2, img1.size(),
R, T, E, F,flag);

cv::FileStorage fs1(out_file, cv::FileStorage::WRITE);

fs1 << "K1" << K1;

fs1 << "K2" << K2;

fs1 << "D1" << D1;

fs1 << "D2" << D2;

fs1 << "R" << R;

fs1 << "T" << T;

fs1 << "E" << E;

fs1 << "F" << F;

fs1 << "lato_quadrato" << dim;

printf("Done Calibration\n");

waitKey();

return 0;

}

```

Codice mappa disparità:

```

#include "opencv2/calib3d/calib3d.hpp"

#include "opencv2/imgproc.hpp"

#include "opencv2/imgcodecs.hpp"

#include "opencv2/highgui.hpp"

#include "opencv2/core/utility.hpp"

#include <iostream>

using namespace std;

using namespace cv;

static void salva_punti_3D(const char* filename, const Mat& mat)

{

    const double max_z = 1.0e4;

    FILE* fp = fopen(filename, "wt");

    for(int y = 0; y < mat.rows; y++)

    {

        for(int x = 0; x < mat.cols; x++)

```

```

{
    Vec3f punti = mat.at<Vec3f>(y, x);

    if(fabs(punti[2] - max_z) < FLT_EPSILON || fabs(punti[2]) > max_z) continue;
    fprintf(fp, "%f %f %f\n", punti[0], punti[1], punti[2]);
}

}

fclose(fp);
}

int main()
{
    waitKey();

    char img1_filename[30];
    char img2_filename[30];
    char* file_name_param_intrisec = "Risultati_calibrazione_stereo.txt";
    char* file_name_param_estrarinsec = "Risultati_calibrazione_stereo.txt";
    char* file_name_salv_dispar = "mappa_disparita.jpg";
    char* punti_cloud_filename = "punti_xyz.txt";
    cout<<"inserisci nome immagine sinistra per esteso:"<<endl;
    cin>>&img1_filename[0];
    cout<<"inserisci nome immagine destra per esteso:"<<endl;
    cin>>&img2_filename[0];
    enum { STEREO_BM=0, STEREO_SGBM=1, STEREO_HH=2, STEREO_VAR=3, STEREO_3WAY=4
};

    int alg = STEREO_SGBM;

    cout<<"scegli algoritmo:"<<endl<<"0)STEREO_BM"<<endl<<"1)STEREO_SGBM"<<endl<<"2)
STEREO_HH"<<endl<<"3)STEREO_VAR"<<endl<<"4)STEREO_3WAY"<<endl;

    cin>>alg;
//    alg = STEREO_BM;
//    alg = STEREO_SGBM;
//    alg = STEREO_HH;
//    alg = STEREO_VAR;
//    alg = STEREO_3WAY;

    int SADWindowSize=21;
    int numberofDisparities=16;
}

```

```

    waitKey();

    Ptr<StereoBM> bm = StereoBM::create(16,21);

    Ptr<StereoSGBM> sgbm = StereoSGBM::create(0,16,9);

    if( alg < 0 )

    {

        cout<<"Command-line parameter error: Unknown stereo algorithm\n\n";

        waitKey();

        return -1;

    }

    int color_mode = alg == STEREO_BM ? 0 : -1;

    Mat img1 = imread(img1_filename, color_mode);

    Mat img2 = imread(img2_filename, color_mode);

    Size img_size = img1.size();

    Rect roi1, roi2;

    Mat Q;

    // reading intrinsic parameters

    FileStorage fs(file_name_param_intrinsic, FileStorage::READ);

    if(!fs.isOpened())

    {

        cout<<"failed open intrinsic"<<endl;

        waitKey();

        return -1;

    }

    Mat R, T, R1, P1, R2, P2;

    Mat K1, D1, K2, D2;

    fs["R"] >> R;

    fs["K1"] >> K1;

    fs["K2"] >> K2;

    fs["D1"] >> D1;

    fs["D2"] >> D2;

    fs["T"] >> T;

    cout<<T<<endl;

```

```

    waitKey();

    K1 *= scale;
    K2 *= scale;

    stereoRectify( K1, D1, K2, D2, img_size, R, T, R1, R2, P1, P2, Q, 0, 1, img_size, &roi1, &roi2
);

    Mat map11, map12, map21, map22;

    initUndistortRectifyMap(K1, D1, R1, P1, img_size, CV_16SC2, map11, map12);
    initUndistortRectifyMap(K2, D2, R2, P2, img_size, CV_16SC2, map21, map22);

    Mat img1r, img2r;

    remap(img1, img1r, map11, map12, INTER_LINEAR);
    remap(img2, img2r, map21, map22, INTER_LINEAR);

    img1 = img1r;
    img2 = img2r;

    numberOfDisparities = numberOfDisparities > 0 ? numberOfDisparities : ((img_size.width/8)
+ 15) & -16;

    bm->setROI1(roi1);
    bm->setROI2(roi2);
    bm->setPreFilterCap(31);
    bm->setBlockSize(SADWindowSize > 0 ? SADWindowSize : 9);
    bm->setMinDisparity(0);
    bm->setNumDisparities(numberOfDisparities);
    bm->setTextureThreshold(10);
    bm->setUniquenessRatio(15);
    bm->setSpeckleWindowSize(100);
    bm->setSpeckleRange(32);
    bm->setDisp12MaxDiff(1);
    sgbm->setPreFilterCap(63);

    int sgbmWinSize = SADWindowSize > 0 ? SADWindowSize : 3;
    sgbm->setBlockSize(sgbmWinSize);
    int cn = img1.channels();
    sgbm->setP1(8*cn*sgbmWinSize*sgbmWinSize);
    sgbm->setP2(32*cn*sgbmWinSize*sgbmWinSize);
    sgbm->setMinDisparity(0);

```

```

sgbm->setNumDisparities(numberOfDisparities);
sgbm->setUniquenessRatio(10);
sgbm->setSpeckleWindowSize(100);
sgbm->setSpeckleRange(32);
sgbm->setDisp12MaxDiff(1);
if(alg==STEREO_HH)
    sgbm->setMode(StereoSGBM::MODE_HH);
else if(alg==STEREO_SGBM)
    sgbm->setMode(StereoSGBM::MODE_SGBM);
Mat disp, disp8;
//Mat img1p, img2p, dispp;
//copyMakeBorder(img1, img1p, 0, 0, numberOfDisparities, 0, IPL_BORDER_REPLICATE);
//copyMakeBorder(img2, img2p, 0, 0, numberOfDisparities, 0, IPL_BORDER_REPLICATE);
int64 t = getTickCount();
if( alg == STEREO_BM )
    bm->compute(img1, img2, disp);
else if( alg == STEREO_SGBM || alg == STEREO_HH || alg == STEREO_3WAY )
    sgbm->compute(img1, img2, disp);
t = getTickCount() - t;
printf("Time elapsed: %fms\n", t*1000/getTickFrequency());
//disp = dispp.colRange(numberOfDisparities, img1p.cols);
if( alg != STEREO_VAR )
    disp.convertTo(disp8, CV_8U, 255/(numberOfDisparities*16.));
else
    disp.convertTo(disp8, CV_8U);
//if( !no_display )
{
    namedWindow("left", 1);
    imshow("left", img1);
    namedWindow("right", 1);
    imshow("right", img2);
    namedWindow("disparity", 0);
}

```

```

imshow("disparity", disp8);

printf("press any key to continue...");

fflush(stdout);

waitKey();

printf("\n");

}

imwrite(file_name_salv_dispar, disp8);

{

printf("storing the punti cloud...");

fflush(stdout);

Mat xyz;

reprojectImageTo3D(disp, xyz, Q, true);

salva_punti_3D(punti_cloud_filename, xyz);

printf("\n");

}

return 0;
}

```

Codice correlazione d'immagine bidimensionale secondo l'algoritmo lucas kanade con trasformazione affine e compensazione radiometrica. codice elaborazione dati macchina:

```

function [ load, strain ] =
Curv_SD_machine(nome_file,colonna_spostamenti,lunghezza_l_zero,numero_
figura )
valori_macchina=importdata(nome_file);
temp_val_load=0;
counter=0;
cv=1;
counter_second=1;
temp_val_deform=0;
tmp_sec=1.0;
spostamenti=valori_macchina(:,colonna_spostamenti);
deformazioni=spostamenti./lunghezza_l_zero;
for i=1:size(valori_macchina,1)
    if(valori_macchina(i,6)==1)%piglio solo parte carico
        if(mod(valori_macchina(i,5),1)==0)%piglio i carichi ad ogni
secondo
            counter_second=valori_macchina(i,5);

        if(tmp_sec==counter_second)

```

```

temp_val_load=temp_val_load+valori_macchina(i,1);
temp_val_deform=temp_val_deform+deformazioni(i);
counter=counter+1;
load(cv,1)=temp_val_load/counter;
strain(cv,1)=temp_val_deform/counter;
else
cv=cv+1;
    temp_val_load=temp_val_load+valori_macchina(i,1);
    temp_val_deform=temp_val_deform+deformazioni(i);
    counter=counter+1;
    load(cv,1)=temp_val_load/counter;
    strain(cv,1)=temp_val_deform/counter;
end
tmp_sec=counter_second;
end;
end
if(mod(valori_macchina(i,5),1)~=0)
    counter=0;
    temp_val_load=0;
    temp_val_deform=0;
end;
for j=1:size(load,1)
    if(load(j)==0)
        load(j)=(load(j+1)+load(j-1))/2;
    end
    if(strain(j)==0)
        strain(j)=(strain(j+1)+strain(j-1))/2;
    end
end
%load contiene ora il carico ad ogni secondo in Newton dai dati
figure(numero_figura)
hold on
title('Sigma-Epsilon dati macchina')
xlabel('epsilon [mm/mm]')
ylabel('sigma [MPa]')
plot(strain,load,'*')
hold off
end

```

codice estrazione frames dal video:

```

source='prov.mp4'; %indicare fra parentesi nome video con estensione

vidobj=VideoReader(source);
frames=vidobj.Numberofframes;
counter=0;
for f=1:frames

b = mod(f,); %indicare il numero di secondi da estrarre intrervallo:1
secondo
if(b==0)
thisframe=read(vidobj,f);
thisfile=sprintf('vale4sec/%s.jpg', secs2hms(counter));
counter=counter+1;
imwrite(thisframe,thisfile);
end

```

```
end
```

derivata temporale con modello compensazione trasformazione radiometrica:

```
function [A,aa] = derivata_tt(matrice1,matrice2,ordx,ordy)
%#coder
temp_med_1=0;
temp_med_2=0;
n=ordx*ordy;

num=0;
den=0;
for i=1:ordy
    for j=1:ordx

        temp_med_1=temp_med_1+matrice1(i,j);
        temp_med_2=temp_med_2+matrice2(i,j);

    end
end
Fm=temp_med_1/n;
Gm=temp_med_2/n;
for i=1:ordy
    for j=1:ordx
        den=den+ (matrice2(i,j)-Gm)^2;
        num=num+ (matrice1(i,j)-Fm)*(matrice2(i,j)-Gm) ;
    end
end

aa=num/den;
bb=Fm-(aa*Gm);
A=matrice2.*aa+bb-matrice1;

end
```

derivate x ed y

```
function A = derivata_x(matrice1)
f = [-1 0 1;
      -2 0 2;
      -1 0 1];
f=f.*1/8;
A=conv2(matrice1,f,'same');
end
function A = derivata_y(matrice1)

f = [1 2 1;
      0 0 0;
      -1 -2 -1];
f=f.*1/8;
A=conv2(matrice1,f,'same');
end
```

codice caricamento immagini:

```

function [ imgdata ] = load_imgf(f,path)
    counter_min=floor(f/60);
%
%      if (flag==0)
%          counter_min=counter_min+1;
%
%      end
%
%      if (counter_min==0)
%          thisfile=sprintf('%s%.1f secs.jpg',path,f);
%
%      else if (counter_min==1)
%          thisfile=sprintf('%s%d min, %.1f secs.jpg',path,counter_min,f-
%          (counter_min*60));
%
%          else if (counter_min>1)
%              thisfile=sprintf('%s%d mins, %.1f secs.jpg',path,counter_min,f-
%              (counter_min*60));
%
%          end
%
%      end
%
%      imgdata=imread(thisfile);
%      disp( thisfile );
%      disp('   ');
end

```

codice correlazione :

```

function [ U, V ] = Lucas_kanade( foto1, foto2,dim_fin,trasf)
%#codegen
ordy=size(foto1,1);
ordx=size(foto2,2);

x_fin=dim_fin;
dx=round(x_fin/2);
fx=derivata_x(foto1);
fy=derivata_y(foto1);
[ft aa]=derivata_tt(foto1,foto2,ordx,ordy);
%ft=derivata_t(foto1,foto2,ordx,ordy);
U=zeros(ordy,ordx);
V=zeros(ordy,ordx);
n=dim_fin*dim_fin;
A=zeros(n,6);
b=zeros(1,n);
%270:475 per mio
m=1
%    temp_med_1=0;
%    temp_med_2=0;
for i=dx:ordy-dx-1
for j=270:475:j=dx:ordx-dx-1
    %j=dx:ordx-x_fin-1:j=260:490-x_fin-1%
    %% carico matrice A
    for k=-(dx-1):(dx-1)
        for l=-(dx-1):(dx-1)
            if(trasf==1)
                A(m,1)=aa*fx(i+k,j+l)*k;
                A(m,2)=aa*fx(i+k,j+l)*l;
                A(m,3)=aa*fx(i+k,j+l);
                A(m,4)=aa*fy(i+k,j+l)*k;
                A(m,5)=aa*fy(i+k,j+l)*l;
                A(m,6)=aa*fy(i+k,j+l);
                b(m)=ft(i+k,j+l);
            m=m+1;
        end
    end
end

```

```

    end
    if(trasf==2)
        A(m,1)=aa*fx(i+k,j+l);
        A(m,2)=aa*fy(i+k,j+l);
        b(m)=ft(i+k,j+l);
    %     b(m)=aa*foto2(i+k,j+l)+bb-foto1(i+k,j+l);
        m=m+1;
    end
end
end
m=1;
E=(transpose(A))*transpose(b);
D= pinv(transpose(A)*A);
C= D * E;
if(trasf==1)
    U(i,j)=(C(1)*j+C(2)*i+C(3));
    %U(i+dx,j+dx)=C(3);
    U(i,j)=C(3);
    V(i,j)=(C(4)*j+C(5)*i+C(6));
    %V(i+dx,j+dx)==C(6);
    V(i,j)=C(6);
else if(trasf==2)
    U(i,j)=C(1);
    V(i,j)=C(2);
end
end
end
%quiver(U,V)

end

```

routine di richiamo delle funzioni nell'ordine:

```

clear all
clc
foto1=load_imgf(1,'frames_video2sec/');
%foto1 imread('foto/1.jpg');
ordy=size(foto1,1);
ordx=size(foto1,2);
Ut=zeros(ordy,ordx);
Vt=zeros(ordy,ordx);
offset_sec=35;
num_frames=114;
temp_val=0;
strains=zeros(1,offset_sec+num_frames-offset_sec);
%h1 = fspecial('gaussian',150,1);

for f=offset_sec:offset_sec+num_frames-1

foto1=load_imgf(f,'frames_video2sec/');
%foto1=imfilter(foto1,h1);
%foto1 = imsharpen(foto1,'Radius',2,'Amount',1);
foto2=load_imgf(f+1,'frames_video2sec/');
%foto2=imfilter(foto2,h1);
%foto2 = imsharpen(foto2,'Radius',2,'Amount',1);
[a b]=set_immag(foto1,foto2);


```

```

%a=imfilter(a,h1);
%b=imfilter(b,h1);
[U,V] = Lucas_kanade_mex( a,b,25,1);
Ut=Ut+U;
Vt=Vt+V;
for i=300:440
    epsilon=(Vt(60,i)-Vt(500,i))/(500-60);%riga 26 e riga 555
    temp_val=temp_val+epsilon;
end
deform=temp_val/(440-300);%media sulla linea
strains(f-(offset_sec-1))=deform;
temp_val=0;

end

%[load strain] = Curv_SD_machine('Dic_val.txt',4,150,5 );
[load strain] = Curv_SD_machine('valori macchina.txt',4,150,5 );
figure(1)
imagesc(Vt);
colorbar
figure(2);
imagesc(Ut);
colorbar

counter=1;
for i=1:size(load,1)
    if(mod(i,2)==0)
        l(counter,1)=load(i,1);
        counter=counter+1;
    end
end
load=load/30.28;
l=l/30.28;
numero_punti_dic=14;
numero_punti_m=30;
lunghezz=5;
clear xx1 yy2 xx2 yy1 xx2_d yy2_d xx1_m yy1_m
figure(4)
hold
xx1=strain(1:3:numero_punti_m);
yy1=load(1:3:numero_punti_m);
p=polyfit(xx1,yy1,1);
xx1_m=linspace(0,lunghezz,100);
yy1_m=xx1_m.*p(1)+xx1_m.*p(2);

xx2=strains(1:numero_punti_dic);
yy2=transpose(l(1:numero_punti_dic));
p1=polyfit(xx2,yy2,1);
xx2_d=linspace(0,lunghezz,100);
yy2_d=xx2_d.*p1(1)+xx2_d.*p1(2);
xlim([0 0.022])
ylim([0 100])
plot( strains(1,:) , l( 1 : size(strains(1,:),2)),'.r',strain,load,xx1_m,yy1_m,'b',xx2_d,yy2_d,'r'
,xx1,yy1,'ob',xx2,yy2,'or'),grid on

xlabel('Deformazioni [mm/mm], [pixel/pixel]')
ylabel('Carico [MPa]')

```

```

legend( 'punti curva sforzo-deformazione\n dai dati
DIC2D', 'curva sforzo-deformazione\n dai dati della
macchina', 'retta interpolante per il calcolo\n newlinedel Modulo di
Young dai dati della macchina', 'retta interpolante per il
calcolo\n newlinedel Modulo di Young dai dati Dic2D', 'punti sulla curva
sforzo deformazione della\n newlinemacchina utilizzati nel calcolo del
modulo di\n newlineYoung', 'punti della curva sforzo
deformazione\n newlinedel dic utilizzati nel calcolo del modulo
di\n newlineYoung')

%legend( 'punti curva sforzo-deformazione dai dati DIC2D', 'curva
sforzo-deformazione dai dati della macchina')
modulo_young_dic=p1(1)
modulo_young_m=p(1)

```

implementazione iterativa in forma funzionale dell'algoritmo Horn & Schunck:

```

function [U,V] = Horn_Schunck( foto1,foto2,num_iteration,lambda)

ordy=size(foto1,1);
ordx=size(foto2,2);

fx=derivata_x(foto1,foto2,ordx,ordy);
fy=derivata_y(foto1,foto2,ordx,ordy);
ft=derivata_t(foto1,foto2,ordx,ordy);
U=zeros(ordy,ordx);
Uav=zeros(ordy,ordx);
Vav=zeros(ordy,ordx);
V=zeros(ordy,ordx);
l=lamda;
for n=1:num_iteration
    for i=1:ordy
        for j=1:ordx
            U(i,j)=Uav(i,j)-
fx(i,j)*((fx(i,j)*Uav(i,j)+fy(i,j)*Vav(i,j)+ft(i,j))/(1+(fx(i,j)^2)+(f
y(i,j)^2)));
            V(i,j)=Vav(i,j)-
fy(i,j)*((fx(i,j)*Uav(i,j)+fy(i,j)*Vav(i,j)+ft(i,j))/(1+(fx(i,j)^2)+(f
y(i,j)^2)));
        end
    end
    Uav=average(U,ordx,ordy);
    Vav=average(V,ordx,ordy);
    quiver(U,V)
end
end

```

codice analisi sperimentale del proiettore e costruzione griglia compensata sul piano:

```

clear all
l=[27.7 27.4 28.7 33.5 36.8]
d=[21.8 21.7 22.6 25.85 28.2]
l=l.*10;
d=d.*10;
p = polyfit(l,d,2);

```

```

x1=linspace(min(l),max(l),100)
y1 = polyval(p,x1);
figure (1)
hold on
plot(l,d,'*r'), grid on
plot(x1,y1), grid on
xlabel('distanza dal piano l [mm]')
ylabel('spessore schermo d [mm]')
gamma=2*atan((d./2)./l)
gamma=mean(gamma)
l=300;
d=y1(1);
k=27;
theta_=gamma/k;
theta_st=(180-gamma)/2;
angle=theta_st;
lunghezza=0;
for i=1:k
    angle=angle+theta_;
    s(i)=(1/tand(angle-theta_))-(1/tand(angle));
    ll(i)=i;
    lunghezza=lunghezza+s(i);
end
%con triangolazione
alfa=30;
angle=0;
lunghezza(1)=0;
h(1)=d;
for i=1:k
    ss(i)=s(i)*sin(theta_st+angle)/(sin(90+(gamma/2)-angle-alfa));
    angle=angle+theta_;
    ll(i)=i;
    lunghezza(i+1)=lunghezza(i)+ss(i);
    h(i+1)=d;
end
figure(2)
hold
plot(ll,ss,'.r'), grid on
xlabel('numero frangia')
ylabel('spessore frangia [mm]')
figure(4)
hold on
xlim([0 lunghezza(k+1)])
for i=1:k
    j=mod(i,2);
    if j==1
        l='k';
    else l='w';
    end
    plot([lunghezza(i) lunghezza(i+1)], [h(i)
h(i+1)],l,'LineWidth',10000),
end
xlabel('x [mm]')
ylabel('y')
lunghezza(i+1)
kk=0;
i=0;
while (kk<100)
    i=i+1;

```

```

    kk=kk+ss(i);
end

```

codice analisi proiezione effettuata sulla base del datasheet (funzione):

```

function [
scala_mm_per_pixel_orizzontale, scala_mm_per_pixel_verticale,
angolo_apert_x, angolo_apert_z ,angolo_i] = trova_scala_pixel(
distanza,flag )
d=[1165 1747 2329 2911 3494 4076 4658 5241 5823];%mm
vertical_offset=[25 37 50 62 75 87 100 112 125];%mm
altezza=[498 747 996 1245 1494 1743 1992 2241 2491];%mm
larghezza=[886 1328 1771 2214 2657 3099 3542 3985 4428];%mm
pixel_w=1920;
pixel_h=1080;
for i=1:size(d,2)
    angolo_offset(i)=atan2(vertical_offset(i)/d(i));
end
%angolo iniziale
angolo_i=mean(angolo_offset)
%% %angolo apertura_z
for i=1:size(d,2)
    angolo_apertura_z(i)=atan2((vertical_offset(i)+altezza(i))/d(i))-angolo_i;
end
angolo_apert_z=mean(angolo_apertura_z);
%% legge mm/pixel
for i=1:size(d,2)
    pixel_mm_w(i)=larghezza(i)/pixel_w;
    pixel_mm_h(i)=altezza(i)/pixel_h;
end
p_w=polyfit(d,pixel_mm_w,1);
p_h=polyfit(d,pixel_mm_h,1);
x_w=linspace(0,d(9),200);
y_w=x_w.*p_w(1)+p_w(2);
y_h=x_w.*p_h(1)+p_h(2);
%inserire distanza così da sapere quanti millimetri equivalgono al pixel
scala_mm_per_pixel_verticale=distanza*p_h(1)+p_h(2)
scala_mm_per_pixel_orizzontale=distanza*p_w(1)+p_w(2)
%% angolo apertura x
for i=1:size(d,2)
    angolo_apertura_x(i)=2*atan2((larghezza(i)/2)/d(i));
end
angolo_apert_x=mean(angolo_apertura_x)
%% grafici
if(flag==1)
figure(10)
hold
title('angolo offset')
xlabel('distanza mm');
ylabel('angolo offset°');
plot(d,angolo_offset,'*'), grid on
figure(20)
hold
title('angolo \delta°')
xlabel('distanza mm');
ylabel('\delta°');

```

```

plot(d,angolo_apertura_z,'*'), grid on
figure(30)
P1=subplot(2,1,1)
hold
xlabel(P1, 'distanza mm');
ylabel(P1, '[mm/pixel] orizzontale');
plot(d,pixel_mm_w,'*',x_w,y_w,distanza,scala_mm_per_pixel_orizzontale,
'*r'), grid on
P2=subplot(2,1,2)
hold
xlabel(P2, 'distanza mm');
ylabel(P2, '[mm/pixel] verticale');
plot(d,pixel_mm_h,'*',x_w,y_h,distanza,scala_mm_per_pixel_verticale,'*r'),
grid on
figure(40)
hold
title('angolo \gamma')
xlabel('distanza mm');
ylabel('\gamma');
plot(d,angolo_apertura_x,'*'), grid on
end
end

```

codice costruzione griglia compensata in funzione degli angoli γ, δ :

```

clear all
clc
tipo_frange=1;%1 sinusoidali, 2 rettangolari
distanza_proiezione=290;
[sx,sy,angolo_ap_x,angolo_apertura_z,angolo_i]=trova_scala_pixel(dista
nza_proiezione,0)
A=0.5;
dimesion_x_sensore_pixels=1980;
dimesion_y_sensore_pixels=1080;
angolo_per_pixel_x=angolo_ap_x/dimesion_x_sensore_pixels;
angolo_per_pixel_y=angolo_apertura_z/dimesion_y_sensore_pixels;
dimensioni_finestra_proiezione_x=420;
dimensioni_finestra_proiezione_y=420;
offset_finestra_y=100;
angolo_i=angolo_i+offset_finestra_y*angolo_per_pixel_y
angolo_apertura_x_eff=angolo_per_pixel_x*dimensioni_finestra_proiezion
e_x
angolo_apertura_z_eff=angolo_per_pixel_y*(dimensioni_finestra_proiezio
ne_y)
T=60;%frange/periodo
pii(1)=0;
pii(2)=pi/2;
pii(3)=pi;
pii(4)=(3/2)*pi;
phase=pii(1);
angolo_proi=35;
angolo_i=20;
ang_finale=angolo_i+angolo_apertura_z_eff;
numero_sin=12;
numero_frange=numero_sin*T;
xb=linspace(-
distanza_proiezione*sind(angolo_apertura_x_eff/2),distanza_proiezione*
sind(angolo_apertura_x_eff/2),(numero_frange)+1);

```

```

zb_alto=linspace(distanza_proiezione*tand(ang_finale),distanza_proiezione*tand(ang_finale),numero_frange+1);
zb_basso=linspace(distanza_proiezione*tand(angolo_i),distanza_proiezione*tand(angolo_i),numero_frange+1);
yb=linspace(distanza_proiezione,distanza_proiezione,numero_frange+1);
figure(1)
xlabel(' [mm]');
ylabel(' [mm]');
for i=1:numero_frange
    if(tipo_frange==2)
        if(mod(i,2)==1)
            color(i)=0;
        else color(i)=1;
        end
    else if(tipo_frange==1)
        if(i==100)
            phase=pi/4;
        else
            phase=0;
        end
    end
    color(i)=0.5+A*sin((2*pi/T)*(i-1)+phase);
end
patch([xb(i) xb(i) xb(i+1) xb(i+1)],[zb_alto(i) zb_basso(i)
zb_basso(i+1) zb_alto(i+1)], [color(i) color(i)
color(i)],'LineStyle','none');
end
%piano
R1=[cosd(angolo_proi) -sind(angolo_proi) 0;
    sind(angolo_proi) cosd(angolo_proi) 0;
    0 0 1];
R2=[0 0 1;
    0 cosd(angolo_i) -sind(angolo_i);
    0 sind(angolo_i) cosd(angolo_i)];
dist_plane=100;
Q=R2*R1;
k=[0,1,0]*Q;
a=k(1);
b=k(2);
c=k(3);
d=-b*dist_plane;
for i=1:numero_frange+1
t1=(-d)/(a*xb(i)+b*yb(i)+c*zb_basso(i));
t2=(-d)/(a*xb(i)+b*yb(i)+c*zb_alto(i));
x_intersez_alto(i)=t1*xb(i);
y_intersez_alto(i)=t1*yb(i);
z_intersez_alto(i)=t1*zb_alto(i);
x_intersez_basso(i)=t2*xb(i);
y_intersez_basso(i)=t2*yb(i);
z_intersez_basso(i)=t2*zb_basso(i);
end
figure(2)
hold
grid
%ylim([0 301])
for i=1:numero_frange
    xx=[x_intersez_alto(i) x_intersez_basso(i) x_intersez_basso(i+1)
x_intersez_alto(i+1)];

```

```

yy=[y_intersez_alto(i) y_intersez_basso(i) y_intersez_basso(i+1)
y_intersez_alto(i+1)];
zz=[z_intersez_alto(i) z_intersez_basso(i) z_intersez_basso(i+1)
z_intersez_alto(i+1)];
fill3(xx,yy,zz,[color(i) color(i) color(i)],'LineStyle','none');
xx1=[xb(i) xb(i) xb(i+1) xb(i+1)];
yy1=[yb(i) yb(i) yb(i+1) yb(i+1)];
zz1=[zb_alto(i) zb_basso(i) zb_basso(i+1) zb_alto(i+1)];
fill3(xx1,yy1,zz1,[color(i) color(i)
color(i)],'LineStyle','none');
end
%2100 -325
figure('Units','pixels','pos',[100 100 420 420])
hold
set(gca,'position',[0 0 1 1],'units','normalized')
axis 'equal'
%xlim([x_intersez_basso(1),x_intersez_basso(numero_frange)]);
% ylim([z_intersez_basso(numero_frange),5.0]);
%
axis off
% axis 'off'
%set(gca,'Xdir','reverse')
for i=1:numero_frange
%    if(mod(i,2)==1)
%        color=[0 0 0];
%    else color=[0.8 0.8 0.8];
%    end
x(i)=i-1;
x_p=[x_intersez_alto(i) x_intersez_basso(i) x_intersez_basso(i+1)
x_intersez_alto(i+1)];
z_p=[z_intersez_alto(i) z_intersez_basso(i) z_intersez_basso(i+1)
z_intersez_alto(i+1)];
j= patch(x_p,z_p,[color(i) color(i) color(i)],'LineStyle','none');
grid
end
%
hold off
figure(4)
plot(x,color),grid

```

codice analisi variazione passo frangia:

```

clear all
I=imread('n.jpg');
figure(1)
imshow(I);
I=rgb2gray(I);
I1=im2bw(I,90/255);
I1=im2double(I1);
figure(2)
imshow(I1);
kernelx=[-1 0 1;
          -2 0 2;
          -1 0 1;]
kernely=[1 2 1;
          0 0 0;
          -1 -2 -1;]
k1=conv2(I1,kernelx,'same');

```

```

k2=conv2(I1,kernely,'same');
k=sqrt(k1.^2+k2.^2);
figure(3)
imshow(k);
spessore=zeros(1,31);
n=1;
flag=1;
for i=1:size(k,2)
    valore=k(230,i);
    if(valore==0)
        spessore(n)=spessore(n)+1;
        flag=0;
    else if(valore~=0 && flag==0)
        n=n+1;
        flag=1;
    end
end
x=linspace(1,n,n-1);
figure(4)
plot(x,spessore,x,spessore,'*'), grid
Std = std(spessore(2:size(spessore,2)-1))
mean=mean(spessore(2:size(spessore,2)-1))

```

codice per calcolo fase frazionaria date le quattro immagini con gli sfasamenti imposti:

```

% clear all

I_1=imread('1.tif');%0
I_1=rgb2gray(I_1);
I_1=im2double(I_1);
I_2=imread('2.tif');%pi/2
I_2=rgb2gray(I_2);
I_2=im2double(I_2);
I_3=imread('3.tif');%pi
I_3=rgb2gray(I_3);
I_3=im2double(I_3);
I_4=imread('4.tif');%3/2pi
I_4=rgb2gray(I_4);
I_4=im2double(I_4);
for i=1:size(I_1,1)
    for j=1:size(I_1,2)
        phase_plan(i,j)=atan2((I_4(i,j)-I_2(i,j)),(I_1(i,j)-I_3(i,j)));
    end
end
figure(1)
I = mat2gray(phase_plan);
imshow(I)

```

codice per il calcolo del denoise dati i file .dat contenenti la fase intera per piano e oggetto:

```

clear all
l=40;
deform=load('ph_deform.rg-c.dat');
plan=load('ph_piano.rg-c.dat');

```

```

num_denoise=180;
fase_fraz=plan-deform;
kernel=fspecial('gaussian',[23 23],1);
I=conv2(fase_fraz,kernel,'same');
for i=1:num_denoise
    I=conv2(I,kernel,'same');
end
Px=2;%mm
I=(I.*Px)./(2*pi);
figure(7)
surf(fase_fraz,'linestyle','none');
figure(8)
hold
xlabel('pixel')
ylabel('pixel')
zlabel('[mm]')
surf(I(1:size(I,1)-1,1:size(I,2)-1),'linestyle','none'),grid
plan1=mat2gray(plan);
xx_plan=linspace(1,size(plan1,2),size(plan1,2));
phase_int_plan=plan(200,1:size(plan,2));
figure(1)
hold
plot(xx_plan,phase_int_plan), grid on
xlabel('pixel')
ylabel('fase [radiani]')
% figure(2)
% imshow(plan);
deform1=mat2gray(deform);
xx_deform=linspace(1,size(deform,2),size(deform,2));
phase_int_def=deform(200,1:size(deform1,2));
figure(3)
hold on
xlabel('pixel')
ylabel('fase [radiani]')
plot(xx_deform,phase_int_def), grid on
% figure(4)
% imshow(deform), grid on
k=phase_int_plan-phase_int_def;
ppp=fspecial('gaussian',[1 23],1);
ll=conv(k,ppp,'same');
for i=1:180
    ll=conv(ll,ppp,'same');
end
figure(5)
hold on
xlabel('pixel')
ylabel('fase [radiani]')
plot(xx_plan(1:size(k,2)-1),k(1:size(k,2)-1)), grid on
k1=1;
k2=1;
figure(6)
hold on
xlabel('pixel')
ylabel('fase [radiani]')
%plot(xx_plan,ll), grid on
plot( xx_plan(k1: size(xx_plan,2)-k2), ll( k1:size(ll,2)-k2)), grid on

```

