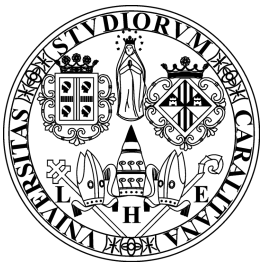

Università degli studi di Cagliari

facoltà di ingegneria

Corso degli studi di Ingegneria Elettronica

Un algoritmo per la Camera Calibration basato sulla visione stereoscopica

Tesi di Laurea Triennale

Relatore:
Prof. Giuseppe Rodriguez

Candidato:
David Melis

Anno Accademico 2016/2017

Alla mia famiglia, ai miei amici
e tutti quelli che mi hanno sostenuto
durante questo percorso, siete la mia forza.
Ringrazio il Professor G. Rodriguez,
che mi ha sopportato pazientemente.
Infine,
un ringraziamento al caffè è d'obbligo.

Indice

Introduzione

La *Camera Calibration* è un tema che negli ultimi decenni, grazie anche allo sviluppo tecnologico riguardante hardware sempre più prestanti, si è sempre più studiato, migliorato e implementato in vari settori, per esempio nella robotica industriale. La *Camera Calibration* è una parte molto importante di quella che viene definita *Computer Vision*[1], un ambito multidisciplinare volto alla ricostruzione della vista umana, quindi una serie di processi propensi a estrapolare informazioni e caratteristiche geometriche da una o più immagini (raffiguranti la stessa scena o ambiente). Lo scopo quindi è riuscire a far interpretare al meglio una scena al sistema di visione. A tal fine, il sistema di visione ha quindi bisogno di un sistema di acquisizione, per esempio una macchina fotografica, e di un sistema che elabori la scena attraverso un software dedicato allo scopo. Come detto precedentemente, grazie a un costante miglioramento dell'hardware in termini prestazionali e a tanti metodi proposti in questi ultimi decenni, la calibrazione risulta essere sempre più veloce e precisa.
La presente tesi è strutturata in modo tale da fornire concetti generali, esporre due metodi di calibrazione ed elaborare un algoritmo che riprende alcune fasi del secondo metodo di calibrazione. Il primo metodo descritto è il *metodo di Zhang*, seguito dal *metodo di Caprile e Torre* più una sua recente ottimizzazione. L'algoritmo verrà scritto su **Matlab** e verranno riportati i risultati in alcune tabelle.

[1] Tradotto in Italiano: Visione Artificiale

Capitolo 1

Nozioni di algebra lineare e di analisi numerica

In questo capitolo verranno esposti vari concetti di base.

1.1 Matrici

Una matrice $m \times n$ è un quadro di mn numeri reali o complessi disposti in m righe e n colonne

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Una matrice è quadrata se $m = n$, in caso contrario viene detta rettangolare.
Utilizzeremo $\mathbb{R}^{m \times n}$ o $\mathbb{C}^{m \times n}$ come riferimento a una matrice reale o complessa.

1.1.1 Tipologie

Esistono tanti tipi di matrici, utili alla risoluzione di determinati problemi.

Matrice aggiunta

Sia $A \in \mathbb{C}^{m \times n}$. La matrice aggiunta A^* si ottiene scambiando le righe di A con le sue colonne e coniugandone gli elementi, per cui:

$$(A^*)_{ij} = \overline{a_{ji}}$$

Matrice trasposta

La matrice trasposta A^T ha le righe scambiate con le colonne, dunque

$$(A^T)_{ij} = a_{ji}$$

Se la matrice A è reale ($a_{ij} \in \mathbb{R}$), la trasposta coincide con l'aggiunta.

Matrice inversa

Una matrice quadrata A si dice invertibile o non singolare se esiste una matrice A^{-1}, detta matrice inversa, tale che $AA^{-1} = A^{-1}A = I$.
Alcune proprietà, più o meno facilmente deducibili dalle definizioni, sono le seguenti:

- $(AB)^T = B^T A^T$;
- $(AB)^{-1} = B^{-1}A^{-1}$;
- $(A^T)^{-1} = (A^{-1})^T$ (si scrive spesso, per brevità, A^{-T});
- A è invertibile se e solo tutte le righe (e colonne) sono linearmente indipendenti.

Matrice Hermitiana

Una matrice complessa A si dice Hermitiana se coincide con la sua aggiunta ($A = A^*$)

Matrice unitaria

Una matrice complessa A si dice unitaria se $A^*A = AA^* = I$.
Se A è unitaria allora $|\det(A)|$

Matrici triangolari

Una matrice A si dice triangolare superiore se $a_{ij} = 0$ per $i > j$

$$A = \begin{bmatrix} a_{11} & \dots & \dots & a_{1j} \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 0 & a_{ij} \end{bmatrix}$$

Al contrario, una matrice A si dice triangolare inferiore se $a_{ij} = 0$ per $i < j$

$$A = \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ a_{i1} & \dots & \dots & a_{ij} \end{bmatrix}$$

Matrice diagonale

Una matrice A si dice diagonale se $a_{ij} \neq 0$ per $i = j$.

$$A = \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & a_{ij} \end{bmatrix}$$

Il determinante di una matrice triangolare (sia essa superiore o inferiore) o di una matrice diagonale è il prodotto degli elementi diagonali.

Matrice omografica

In matematica e geometria un'omografia è una relazione tra punti di due spazi tali per cui ogni punto di uno spazio corrisponde ad uno ed un solo punto del secondo spazio. Dato un insieme di punti x_i ed un insieme di corrispondenti punti x'_i espressi in coordinate omogenee, si vuole stabilire una trasformazione in grado di trasformare i punti x_i nei punti x'_i. Generalmente tale trasformazione riveste grande importanza nella trasformazione di punti da un piano ad un altro nell'ambito della **visione artificiale**.

Omografia bidimensionale

Il problema dell'omografia bidimensionale consiste nella determinazione di una trasformazione in grado di mappare punti di un piano in punti di un altro piano. Si definisce quindi la relazione $x_i \leftrightarrow x'_i$ tra due insiemi di punti. Tale trasformazione si esprime matematicamente tramite il prodotto dei punti per una matrice H 3×3 tale che

$$x'_i = H \cdot x_i, \quad \forall i$$

dove della matrice H non sono importanti i valori di tutti gli elementi, bensì i rapporti tra di essi, con il risultato di avere quindi otto gradi di libertà. Tale equazione può essere riscritta in forma estesa:

$$\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{1.1}$$

1.1.2 Operazioni

Determinante

Il determinante è una funzione che associa a ciascuna matrice quadrata un numero; tale numero è reale se la matrice è reale. Fissata una qualsivoglia riga i, esso può essere calcolato mediante la formula di Laplace.

$$\det(A) = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det(A_{ij})$$

essendo A_{ij} la sottomatrice che si ottiene da A eliminando la i-esima riga e la j-esima colonna. Richiamiamo varie proprietà di che maggiormente interessano numericamente:

- $\det(A^T) = \det(A), \quad \det(A^*) = \overline{\det(A)}, \quad \det(A^{-1}) = (\det(A))^{-1}$;
- $\det(A)\det(B), \quad det(\alpha A) = \alpha^n \det(A)$;
- uno scambio di due righe (o colonne) in una matrice induce un cambio di segno del suo determinante;
- A è non singolare se e solo se $\det(A) \neq 0$.

Rango

Il rango di una matrice A può essere definito indifferentemente come il massimo numero di righe (o colonne) linearmente indipendenti o come l'ordine massimo delle sottomatrici con determinante non nullo.

Somma di matrici

Due matrici A e B, entrambe $m \times n$ possono essere sommate. La loro somma $A + B$ è definita come la matrice $m \times n$ i cui elementi sono ottenuti sommando i corrispettivi elementi di A e B.

$$[A + B]_{ij} = [A]_{ij} + [B]_{ij}$$

Moltiplicazione per un scalare

Data una matrice A ed uno scalare α, ciascun elemento di A va moltiplicato per α.

Prodotto

Il prodotto tra una matrice A e una matrice B è possibile solo se A è una matrice $m \times p$ e B è una matrice $p \times n$; si nota quindi che il numero di colonne di A e il numero di righe di B deve corrispondere.
La definizione di moltiplicazione che segue è motivata dal fatto che una matrice modellizza una applicazione lineare, e il prodotto di matrici corrisponde alla composizione di applicazioni lineari.
Il risultato del prodotto, è una matrice C di dimensioni $m \times n$, in cui ogni elemento è dato dalla seguente somma:

C_{ij} =Riga$_i(A)$ $\times$ Colonna$_j(B) = A_{i1}(B_{1j}) + A_{i2}(B_{2j}) + \cdots + A_{ip}(B_{pj})$
Questo prodotto si definisce *riga per colonna.*
Esempio:
Siano A e B rispettivamente 2×3 e 3×3:

$$A \times B = \begin{bmatrix} 3 & 4 & 5 \\ 1 & 2 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 5 & 2 \\ 2 & 2 & 3 \\ 1 & 4 & 1 \end{bmatrix}$$

Il risultato è una matrice C 2×3:

$$C = \begin{bmatrix} 16 & 43 & 23 \\ 6 & 13 & 9 \end{bmatrix}$$

A differenza di ciò che avviene moltiplicando due numeri, il prodotto tra matrici non è commutativo, cioè $AB \neq BA$.
Nell'esempio sopra riportato, non sarebbe nemmeno possibile il prodotto BA in quanto le colonne di B (essendo tre) non corrispondono alle righe di A (che sono due).

1.1.3 Proprietà

Di seguito vengono elencate le proprietà delle matrici.

- $A + 0 = 0 + A = A$

  dove 0 è chiamata matrice nulla;

- $A + (-A) = 0$

  $-A$ è il risultato del prodotto $(-1)A$;

- $(A + B) + C = A + (B + C)$

  proprietà associativa della somma;

- $A + B = B + A$

  proprietà commutativa della somma;

- $1A = A$

  dove 1 è l'elemento neutro;

- $(ab)A = a(bA)$

  proprietà associativa del prodotto per uno scalare;

- $a(A + B) = aA + aB$

  proprietà distributiva della somma rispetto al prodotto per uno scalare.

1.1.4 Autovalori e autovettori

Si definiscono autovalore ed autovettore di una matrice A quadrata, uno scalare $\lambda \in \mathbb{C}$ ed un vettore $x \neq 0$ che verificano la relazione:

$$Ax = \lambda x$$

che riscriviamo

$$(A - \lambda I)x = 0$$

Affinché questo sistema lineare omogeneo ammetta un $\lambda \in \mathbb{C}$ al quale corrisponde una soluzione non nulla,è necessario e sufficiente che il suo determinante

$$p_A(\lambda) = \det(A - \lambda I)$$

si annulli. Dal momento che $p_A(\lambda)$, detto *polinomio caratteristico* di A, è un polinomio di grado n in λ, il *Teorema fondamentale dell'Algebra* assicura l'esistenza di n zeri di $p_A(\lambda)$ (non necessariamente reali e distinti) che, per definizione, sono gli autovalori di A.
Per ciascun autovalore λ_k con $k = 1, \ldots, n$, ogni soluzione non nulla del sistema singolare omogeneo

$$(A - \lambda_k I)x = 0$$

fornisce un autovettore ad esso corrispondente. Nel caso la matrice $A - \lambda_k I$ abbia rango $n - 1$, esso resta determinato a meno di una costante moltiplicativa. Qualora, più in generale, il rango di $A - \lambda_k I$ sia $n - p$, resta determinato un sottospazio di autovettori (*autospazio*) di dimensione p, caratterizzato da p vettori linearmente indipendenti.

Definiamo lo **spettro** di A l'insieme dei suoi autovalori

$$\sigma(A) = \{\lambda_1, \ldots, \lambda_n\}$$

e **raggio spettrale** il massimo dei moduli degli autovalori

$$\rho(A) = \max_{k=1,\ldots,n} |\lambda_k|$$

Proprietà

Qui di seguito, verranno elencate alcune proprietà

- $\det(A) = \prod_{k=1}^{n} \lambda_k$;
- $\sigma(A^T) = \sigma(A), \quad \sigma(A^{-1}) = \{\lambda_1^{-1}, \ldots, \lambda_n^{-1}\}, \quad \sigma(A^p) = \{\lambda_1^p, \ldots, \lambda_n^p\}$;
- ad autovalori distinti corrispondono autovettori indipendenti;
- se un autovettore x è noto, il **quoziente di Raylegh** $\dfrac{x^T Ax}{x^T x}$ fornisce il corrispondente autovalore.

La **molteplicità algebrica** di un autovalore è la sua molteplicità come zero del polinomio caratteristico. Si definisce, invece, **molteplicità geometrica** di un autovalore il massimo numero di autovettori linearmente indipendenti ad esso corrispondenti.
per ogni autovalore

$$\text{molteplicità geometrica} \leq \text{molteplicità algebrica}$$

Qualora gli autovalori siano tutti distinti, per ognuno di essi la molteplicità algebrica e geometrica sono coincidenti e uguali a 1.

1.2 Problemi ai minimi quadrati

Nelle applicazioni si incontrano frequentemente sistemi lineari con un numero di equazioni differente dal numero delle incognite, cioè del tipo

$$Ax = \mathbf{b}, \tag{1.2}$$

con A matrice $m \times n$, $\mathbf{b} \in \mathbb{R}^m$ e $x \in \mathbb{R}^n$. Ipotizzando che A sia a rango pieno, si possono presentare due casi

1. $\boxed{m > n}$ sono disponibili più equazioni che incognite, per cui il problema potrebbe non avere soluzioni. Il sistema si dice **sovradeterminato**.
2. $\boxed{m < n}$ vi sono più incognite che equazioni, il sistema potrebbe avere infinite soluzioni. Il sistema si dice **sottodeterminato**.

Anche in caso la matrice A non sia a rango, la si può riportare ai casi sopra citati.
Quando i dati sono affetti da errore, è possibile che nessuna equazione che compone il sistema lineare si possa verificare esattamente. Se tutte le equazioni non possono

essere verificate contemporaneamente, è ragionevole che lo scarto quadratico medio (varianza) tra il primo e il secondo membro del sistema sia minimo. Si passa così ad un **problema ai minimi quadrati** (dall'inglese: *least squares problem*). La condizione di varianza minima si può esprimere nella forma

$$\min_{x\in\mathbb{R}^n} \|Ax - \mathbf{b}\|_2. \tag{1.3}$$

$(Ax - \mathbf{b})$ viene chiamato **residuo**.
Se il minimo del residuo risulta essere zero, allora il sistema originale ammette la soluzione in senso classico. In caso contrario si ottiene la soluzione nel senso dei minimi quadrati del sistema lineare sovradeterminato $Ax = \mathbf{b}$.

Esistono due metodi per risolvere il problema ai minimi quadrati

1. Metodo delle equazioni normali.
2. Risoluzione mediante **fattorizzazione QR**.

Metodo delle equazioni normali

Il metodo delle equazioni normali è la più classica delle tecniche per risolvere un problema ai minimi quadrati. Il metodo opera risolvendo il *sistema normale* associato al problema.
Richiamiamo le formule per il calcolo del gradiente

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right)^T \tag{1.4}$$

di alcune funzioni da $\mathbb{R}^n$ in $\mathbb{R}$ espresse in forma matriciale.
Se $f(x) = \mathbf{b}x$ con $x, \mathbf{b} \in \mathbb{R}^n$ (equivalentemente, $f(x) = x^T\mathbf{b}$) è facile verificare che

$$\nabla(\mathbf{b}^T x) = \mathbf{b}. \tag{1.5}$$

In modo analogo si può dimostrare che se A è una matrice $n \times n$

$$\nabla(x^T Ax) = Ax + A^T x.$$

Da quest'ultima formula segue che se A è simmetrica si ha

$$\nabla(x^T Ax) = 2Ax \tag{1.6}$$

Dal momento che vogliamo minimizzare una quantità non negativa, non è restrittivo considerare il quadrato della norma del residuo, che esprimiamo in forma matriciale

$$\|Ax - \mathbf{b}\|^2 = x^T A^T Ax - 2x^T A^T \mathbf{b} + \mathbf{b}^T \mathbf{b}. \tag{1.7}$$

Per minimizzare la norma euclidea del residuo imponiamo l'annullarsi del gradiente

$$\frac{1}{2}\nabla(\|Ax - \mathbf{b}\|^2) = A^T Ax - A^T \mathbf{b} = 0, \tag{1.8}$$

giungendo così al **sistema normale**

$$A^T Ax = A^T \mathbf{b} \tag{1.9}$$

Se A ha rango pieno, la matrice $A^T A \in \mathbb{R}^{n\times n}$ è invertibile, e la soluzione del sistema lineare è unica

$$x_{LS} = (A^T A)^{-1} A^T \mathbf{b}.$$

Se A ha rango inferiore a n, la matrice $A^T A$ risulta singolare, ma il sistema resta consistente in quanto il vettore $A^T \mathbf{b}$ appartiene all'immagine di A^T, che coincide con l'immagine di $A^T A$. In questo caso, tra gli infiniti vettori che soddisfano il sistema, usualmente si assume come soluzione x_{LS} quella dotata di minima norma euclidea (soluzione normale).

Risoluzione tramite fattorizzazione QR

La fattorizzazione QR fornisce un metodo alternativo, stabile ed efficiente, per calcolare la soluzione del problema (1.3). Anche in questo caso è conveniente considerare il quadrato della norma del residuo. Sostituendo al suo interno la fattorizzazione $A = QR$ otteniamo

$$\|Ax - \mathbf{b}\|^2 = \|QRx - \mathbf{b}\|^2 = \|Q(Rx - Q^T \mathbf{b})\|^2 = \|Rx - \mathbf{c}\|^2,$$

ponendo $\mathbf{c} = Q^T \mathbf{b}$ e ricordando che una matrice ortogonale non modifica la norma-2 di un vettore.
La matrice R ha la forma

$$R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix},$$

con R_1 triangolare superiore, quadrata e non singolare in caso A sia a rango pieno. Partizionando in modo coerente il vettore $\mathbf{c}$

$$\mathbf{c} = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix}, \quad \mathbf{c}_1 \in \mathbb{R}^n, \mathbf{c}_2 \in \mathbb{R}^{m-n},$$

otteniamo

$$\|Ax - \mathbf{b}\|^2 = \|Rx - \mathbf{c}\|^2 = \left\| \begin{bmatrix} R_1 x \\ 0 \end{bmatrix} - \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} \right\|^2 = \|R_1 x - \mathbf{c}_1\|^2 + \|\mathbf{c}_2\|.$$

Se $\det(R_1) \neq 0$, il sistema

$$R_1 x = \mathbf{c}_1$$

ammette una e una sola soluzione, per la quale si $\|R_1 x = \mathbf{c}_1\| = 0$.
In corrispondenza a tale soluzione

$$\min_{x \in \mathbb{R}^n} \|Ax - \mathbf{b}\| = \|\mathbf{c}_2\|$$

perciò l'eventuale annullarsi del vettore $\mathbf{c}_2$ significa che x è la soluzione classica del sistema iniziale $Ax- = \mathbf{b}$. In caso contrario, essa è la soluzione nel senso dei minimi quadrati, e la norma di $\mathbf{c}_2$ fornisce la misura del residuo.
Rispetto all'approccio basato sulle equazioni normali questo metodo ha il vantaggio di operare direttamente sulla matrice A, il cui condizionamento è pari alla radice quadrata di quello di $A^T A$.

1.2.1 Approssimazione di funzioni ai minimi quadrati

Calcolare il polinomio di **migliore approssimazione** per una funzione f consiste nel determinare il polinomio di grado n che minimizzi un norma dell'errore

$$\min_{p_n \in \Pi_n} \|p_n - f\|.$$

Se viene utilizzata la norma infinito

$$\|p_n - f\|_\infty = \max_{x \in [a,b]} |p_n(x) - f(x)|$$

Si parla di **migliore approssimazione uniforme**, mentre il polinomio che minimizza la norma di $L^2[a, b]$

$$\|p_n - f\|_2 = \sqrt{\int_a^b [p_n(x) - f(x)]^2 dx}$$

è detto **migliore approssimazione nel senso dei minimi quadrati**. Il primo problema è di difficile soluzione, mentre il secondo è più facilmente trattabile dal punto di vista computazionale. Questo è uno dei motivi per cui la norma-2 viene frequentemente utilizzata nell'approssimazione di funzioni.
In molti problemi applicativi la funzione da approssimare è nota solo attraverso un certo numero di valori affetti da errore. In questi casi non è conveniente effettuare un'interpolazione, dal momento che i dati sono affetti da errori. Si preferisce quindi effettuare un'approssimazione ai minimi quadrati ricorrendo ad una discretizzazione della norma-2.
Siano $x_0, x_1, \ldots, x_m$ le ascisse degli $m+1$ punti per i quali sono noti i valori$y_1, y_2, \ldots, y_m$ della funzione $f(x)$ ($y_i = f(x_i)$). Fissato un numero naturale $n \leq m$, per ogni $p_n \in \Pi_n$ si consideri la norma

$$\|p_n - f\|_2 = \sqrt{\sum_{i=0}^{m} [p_n(x_i) - y_i]^2} \tag{1.10}$$

Intendiamo determinare il polinomio $p_n^*(x)$ di grado n che risolva il problema di minimo

$$\min_{p_n \in \Pi_n} \|p_n - f\|_2^2.$$

Dal momento che i problemi sono equivalenti, minimizzeremo il quadrato della norma. Se $m = n$ la soluzione coincide col polinomio interpolante, se invece $m \geq n$ essa fornisce la migliore approssimazione nel senso dei minimi quadrati rispetto alla norma discreta (1.10).
Utilizzando la base canonica si ottiene

$$p_n(x_i) = \sum_{j=0}^{n} a_j x_i^j = (X\mathbf{a})_i, \quad i = 0, \ldots, m$$

dove $\mathbf{a}= (a_0, \ldots, a_n)^T \in \mathbb{R}^{n+1}$ è il vettore dei coefficienti del polinomio e X è la *matrice di Vandermonde* di dimensione $(m+1) \times (n+1)$

$$X = \begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^n \end{bmatrix}.$$

Di conseguenze si ha

$$\|p_n - f\|_2^2 = \sum_{i=0}^{m} [(X\mathbf{a})_i - y_i]^2 = \|X\mathbf{a} - y\|_2^2,$$

e il problema (1.2.1) risulta equivalente alla risoluzione nel senso dei minimi quadrati del sistema lineare sovradeterminato

$$X\mathbf{a} = y.$$

La risoluzione di tale sistema lineare può essere calcolata con i metodi di *Cholesky* oppure utilizzando la *fattorizzazione QR* della matrice X (metodo più conveniente dal punto di vista della stabilità).

1.2.2 Problemi ai minimi quadrati non lineari

Data una funzione vettoriale $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ con $m \geq n$, vogliamo minimizzare $\|f(x)\|$ o equivalentemente

$$x^* = \underset{x}{\operatorname{argmin}}\, F(x), \tag{1.11}$$

dove

$$F(x) = \frac{1}{2}\sum_{i=1}^{m}(f_i(x))^2 = \frac{1}{2}\|f(x)\|^2 = \frac{1}{2}f(x)^T f(x). \tag{1.12}$$

Il problema può essere risolto tramite metodi di ottimizzazione generali, qui verrà discusso e preso in considerazione un metodo speciale molto più efficiente, che si basa sul *metodo di Gauss-Newton.*

Metodo di Levenberg-Marquardt

Per comprendere il metodo, dobbiamo esporre i primi passi del *metodo di Gauss-Newton.* Il *metodo di Gauss-Newton* è basato su un'approssimazione lineare delle componenti di f (un modello lineare di f) nei dintorni di x. Per un valore di $\|h\|$ piccolo vediamo che dall'espansione di *Taylor*

$$f(x+h) \simeq l(h) \equiv f(x) + J(x)h. \tag{1.13}$$

Inserendola nella (1.12) otteniamo

$$
\begin{aligned}
F(x+h) \simeq L(h) &= \frac{1}{2} l(h)^T l(h) \\
&= \frac{1}{2} f^T f + h^T J^T f + \frac{1}{2} h^T J^T J h \\
&= F(x) + h^T J^T f + \frac{1}{2} h^T J^T J h
\end{aligned} \tag{1.14}
$$

con $f = f(x)$ e $J = J(x)$.
Il *passo di Gauss-Newton* h_{gn} minimizza $L(h)$,

$$
h_{gn} = \underset{h}{\operatorname{argmin}}\, L(h), \tag{1.15}
$$

È facile vedere che la matrice Jacobiana e la matrice Hessiana di L sono

$$
L'(h) = J^T f + J^T J h, \qquad L''(h) = J^T J \tag{1.16}
$$

ed è facile notare che $L'(0) = F'(x)$ e $L''(h)$ è indipendente da h. Se J ha rango pieno, $L''(h)$ è sempre positiva. $L(h)$ può essere trovata risolvendo

$$
(J^T J) h_{gn} = -J^T f. \tag{1.17}
$$

Qui entra in gioco Levenberg-Marquardt: Modifichiamo la (1.17),

$$
(J^T J + \mu I) h_{lm} = -g \quad \text{con} \quad g = J^T f \quad \text{e} \quad \mu \geq 0. \tag{1.18}
$$

Il parametro di smorzamento μ ha effetti importanti:

- Per tutti i $\mu > 0$ la matrice dei coefficienti è definita positiva e questo assicura che h_{lm} abbia una direzione discendente.

- Per valori elevati di μ otteniamo

$$
h_{lm} \simeq -\frac{1}{\mu} g = -\frac{1}{\mu} F'(x),
$$

che è un piccolo passo nella direzione di discesa più ripida. Questo è ottimo se l'iterazione corrente è molto lontana dalla soluzione.

- Se μ è molto piccolo, allora $h_{lm} \simeq h_{gn}$, che è un buon passo quando ci si ritrova verso la fine delle iterazioni, cioè quando x vicino a x^*. Se $F(x^*) = 0$ o comunque molto piccolo, possiamo ottenere (quasi) la convergenza quadratica.

La scelta del valore iniziale di μ dovrebbe essere relazionata agli elementi in $A_0 = J(x_0)^T J(x_0)$

$$\mu_0 = \tau \cdot \max_i \{a_{ii}^{(0)}\} \tag{1.19}$$

dove τ lo scegliamo noi. Durante le iterazioni, il valore di μ può essere modificato e viene usato un *rapporto di guadagno*

$$\varrho = \frac{F(x) - F(x + h_{lm})}{L(0) - L(h_{lm})}$$

dove il denominatore è il guadagno predetto dal modello lineare (1.14),

$$\begin{aligned} L(0) - L(h_{lm}) &= -h_{lm}^T J^T f - \frac{1}{2} h_{lm}^T J^T J h_{lm} \\ &= -\frac{1}{2} h_{lm}^T (2g + (J^T J + \mu I - \mu I) h_{lm}) \\ &= \frac{1}{2} h_{lm}^T (\mu h_{lm} - g). \end{aligned} \tag{1.20}$$

Notare che $h_{lm}^T h_{lm}$ e $-h_{lm}^T g$ sono positivi, quindi $L(0) - L(h_{lm})$ è positivo.
Un grande valore di ϱ indica che $L(h_{lm})$ è una buona approssimazione di $F(x + h_{lm})$, e possiamo diminuire il valore di ϱ in modo che il passo successivo del metodo sia più vicino a quello di Gauss-Newton. Se ϱ è piccolo (anche negativo), allora $L(h_{lm})$ risulta un'approssimazione scarsa, quindi bisogna aumentare il valore di μ con il duplice obiettivo di avvicinarci alla direzione di discesa ripida e riducendo la lunghezza del passo. Questi obiettivi possono essere raggiunti in diversi modi. Il criterio di stop per l'algoritmo dovrebbe riflettere che a un valore minimo globale abbiamo
$F'(x^*) = g(x^*) = 0$ quindi possiamo utilizzare

$$\|g\|_\infty \leq \varepsilon_1, \tag{1.21}$$

dove ε_1 è piccolo, positivo e scelto dall'utente.
Un altro criterio rilevante è dato dal fermare il calcolo se x non varia troppo,

$$\|x_{\text{new}} - x\| \leq \varepsilon_2(\|x\| + \varepsilon_2). \tag{1.22}$$

Infine, come in tutti i processi iterativi, si ha la necessità di salvaguardarci da un loop infinito,

$$k \geq k_{\max} \tag{1.23}$$

Anche ε_2 e $k_{\max}$ sono scelti dall'utente.
Gli ultimi due criteri entrano in vigore se, per esempio, ε_1 è scelto così piccolo che gli effetti dell'errore di arrotondamento influiscono in gran misura. Questo in genere si manifesterà in una scarsa concordanza tra il guadagno reale in F ed il guadagno previsto dal modello lineare (1.14) e il risultato sarà un aumento in ogni passo di μ.

1.3 Propagazione degli errori

In luogo delle normali operazioni aritmetiche $+, -, *$ e $/$, un calcolatore utilizza le così dette *operazioni di macchina* $\oplus, \ominus, \otimes$ e $\oslash$, qui indicate con simboli differenti per enfatizzare il fatto che non coincidono con le operazioni esatte e non ne condividono tutte le proprietà.
Ci porremo nell'ipotesi fondamentale che, qualsiasi siano $x, y \in \mathbb{F}$, sia

$$x \mathbin{\textcircled{\triangle}} y = \text{fl}(x \triangle y) = (x \triangle y)(1 + \varepsilon), \quad |\varepsilon| < u$$

dove $\triangle$ in dica una qualsiasi operazione aritmetica e $\textcircled{\triangle}$ la corrispondente operazione di macchina. L'errore relativo commesso da un'operazione di macchina deve quindi essere dello stesso ordine di grandezza di quello di arrotondamento commesso nel memorizzare il risultato esatto. Quando un sistema in virgola mobile non verifica questa ipotesi si parla di *aritmetica aberrante*.
L'ipotesi è verificata dallo standard IEEE 754. Ciò è reso possibile dal fatto che il processore dispone di registri con una precisione superiore alla doppia (*extended precision*) per la memorizzazione temporanea di operandi e risultato. Il risultato viene poi arrotondato al momento del trasferimento in memoria.
Il fatto veramente importante è che una singola operazione di macchina, se implementata correttamente, non può essere ritenuta responsabile di un errore elevato nel

risultato. Infatti, l'amplificazione degli errori presenti sui dati è spesso causata dal condizionamento dell'operazione aritmetica esatta. Le cose si complicano in presenza di molte operazioni aritmetiche (in un algoritmo, quindi) in quanto i singoli errori compiuti ad ogni passo si possono accumulare in modo diverso a seconda dell'ordine in cui vengono eseguite le operazioni (per le operazioni di macchina non vale la proprietà associativa).

Nell'**analisi in avanti** (*forward analysis o pertubation analysis*) si introducono degli errori nei dati iniziali e si studia come essi si propagano nei calcoli. Studiamo la propagazione degli errori nel prodotto tra due numeri x e y. Introducendo una perturbazione su ciascuno di essi. Siano ε_x e ε_y tali che $|\varepsilon_x|$,$|\varepsilon_y| \leq \tau$. Tralasciando gli infinitesimi di ordine superiore, si ha:

$$x(1+\varepsilon_x) \cdot y(1+\varepsilon_y) \simeq xy(1+\varepsilon_x+\varepsilon_y).$$

Poiché la perturbazione relativa sul prodotto $\varepsilon_{xy} = \varepsilon_x + \varepsilon_y$ verifica la diseguaglianza $|\varepsilon_{xy}| \leq 2\tau$, il numero di condizionamento è 2. Questo valore è decisamente soddisfacente perché indica che, nel peggiore dei casi, gli errori relativi presenti sugli operandi si sommeranno tra loro. Potrebbe quindi verificarsi un accumulo eccessivo solo nel prodotto di molti fattori.

Effettuando l'analisi in avanti per la divisione, si ottiene un risultato analogo a quello ottenuto per la moltiplicazione (condizionamento pari a 2).

Vediamo ora la propagazione degli errori nella somma algebrica (somma o sottrazione di numeri con segno). Siano ε_x e ε_y tali che $|\varepsilon_x|$,$|\varepsilon_y| < \tau$ (se l'errore è dovuto solo all'arrotondamento sarà $\tau = u$). Poiché

$$x(1+\varepsilon_x) + y(1+\varepsilon_y) = x + y + x\varepsilon_x + y\varepsilon_y$$

l'errore relativo nella somma algebrica è

$$\varepsilon_{x+y} = \frac{x}{x+y}\varepsilon_x + \frac{y}{x+y}\varepsilon_y,$$

da cui

$$|\varepsilon_{x+y}| \leq \frac{|x|+|y|}{|x+y|}\tau.$$

Il numero di condizionamento è quindi

$$\kappa = \tfrac{|x|+|y|}{|x+y|}$$

Se x e y hanno lo stesso segno il numero di condizionamento è pari a 1, mentre il suo valore aumenta quando i due numeri hanno segno discorde. In particolare, se il segno è discorde e i loro valori assoluti sono molto vicini ($x \simeq -y$) il valore di κ può crescere in maniera incontrollata. Si parla in questo caso di **cancellazione**. La cancellazione è però dannosa solo quando gli operandi sono affetti da errore, mentre porta a un risultato esatto quando si opera sui numeri di macchina.

Capitolo 2

Camera Calibration

La Camera Calibration è quel processo volto a stimare parametri necessari per correlare i punti del mondo reale (sistema di coordinate mondo) con i punti dell'immagine catturata tramite una telecamera. Affinché si ottengano parametri migliori, si possono scattare più immagini della stessa scena. I parametri si dividono in due categorie:

- parametri intrinseci: sono parametri che dipendono dalla telecamera (per esempio la lunghezza focale f),
- parametri estrinseci: sono parametri che consentono il cambio di coordinate tra due sistemi di riferimento.

Ottenere questi parametri viene detto Camera Calibration.

2.1 Parametri geometrici

I parametri geometrici determinano la posizione dei punti 3D nell'immagine catturata dalla telecamera. Questi parametri sono:

- Tipo di proiezione
- Posizione e orientamento della telecamera nello spazio
- Distorsione dovuta al processo d'immagine
- Proprietà fisiche della matrice del fotosensore della telecamera
- Natura discreta del fotosensore
- Quantizzazione della scala d'intensità

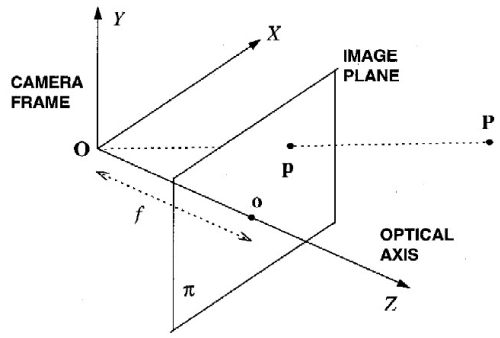

Figura 2.1: La prospettiva della telecamera

Il nostro scopo è collegare la posizione dei punti nella scena con i corrispettivi punti nel piano immagine. Il modello geometrico più comune è rappresentato in figura 2.1

Il modello consiste in un piano π chiamato *piano immagine* e un punto **O** chiamato *centro di proiezione*. La distanza tra **O** e π è chiamata *lunghezza focale*. La linea che attraversa **O** e perpendicolare al piano π è l'*asse ottico* e **o** è l'intersezione tra asse ottico e piano immagine, chiamato *centro immagine*. Come mostrato nella figura 2.1, **p** è il punto immagine del punto **P**, e cioè il punto in cui la retta tra **P** e **O** attraversa π. Preso **O** come origine del sistema di riferimento e π ortogonale all'asse Z, e preso $\mathbf{P}=[X,Y,Z]^T$ e $\mathbf{p}=[x,y,z]^T$. Scriviamo ora le equazioni della proiezione prospettica:

$$x = f\frac{X}{Z} \qquad\qquad y = f\frac{Y}{Z} \tag{2.1}$$

La terza componente è la lunghezza focale f, che nel piano immagine π è costante ($z = f$). Per questo motivo la ometteremo quando dovremo scrivere **p**, utilizzando $\mathbf{p}=[x,y]^T$ invece di $\mathbf{p}=[x,y,z]^T$. Da notare che l'equazione (2.1) è ***non lineare*** a causa di 1/Z e non preserva le distanze tra i punti o gli angoli tra linee.

2.2 Parametri della telecamera

Continuiamo la discussione della geometria di un sistema visivo e in particolar modo di quella che caratterizza i parametri della telecamera. La visione artificiale (dall'inglese: *Computer Vision*) è l'insieme dei processi che mirano a creare un modello approssimato del mondo reale 3D partendo da immagini bidimensionali 2D. Per fare ciò, si necessita di equazioni che leghino il mondo reale a quello descritto in una o più immagini

delle medesima scena. Queste equazioni sono scritte nel sistema di riferimento della telecamera e molto spesso si assume che:

- Il sistema di riferimento della telecamera è posizionato rispetto a un altro sistema di riferimento, conosciuto come sistema di riferimento mondo
- Le coordinate dei punti dell'immagine (o delle immagini) nel sistema di riferimento della telecamera possono essere ottenute dalle coordinate pixel, le uniche direttamente disponibili dall'immagine.

A questo scopo abbiamo bisogno di conoscere le caratteristiche della telecamera (o telecamere, se si parla di **visione stereoscopica**), conosciute come **parametri intrinseci** e **parametri estrinseci**. Il prossimo passo è quindi quello di descrivere l'esatta natura di questi parametri.

Definizione: Parametri della telecamera.
I **parametri estrinseci** sono parametri che definiscono la posizione e l'orientamento del sistema di riferimento della telecamera rispetto al sistema di riferimento mondiale.
I **parametri intrinseci** sono parametri necessari a collegare le coordinate pixel di un'immagine con le corrispondenti coordinate nel sistema di riferimento della telecamera.

Nelle prossime due sezioni descriveremo le equazioni che ci consentiranno di definire i parametri (estrinseci e intrinseci) in termini pratici.

Determinare questi set di parametri viene definito **Camera Calibration**.

2.2.1 Parametri estrinseci

Il sistema di riferimento della telecamera è stato introdotto allo scopo di scrivere le equazioni fondamentali della proiezione prospettica (2.1) in una forma più comoda e semplice. Il sistema di riferimento della telecamera è molto spesso sconosciuto. Un problema comune è determinare la posizione e l'orientamento rispetto a un sistema di riferimento conosciuto, utilizzando esclusivamente informazioni ottenute attraverso le immagini catturate. I parametri estrinseci sono definiti come un *set di parametri geometrici che identifica univocamente la trasformazione tra il sistema di riferimento della telecamera a noi sconosciuto e il sistema di riferimento mondo a noi noto.*
Una tipica scelta per descrivere la trasformazione tra questi due sistemi di riferimento consiste nell'usare

- Un vettore di traslazione T a tre dimensioni che descrive la relativa posizione delle origini nei due sistemi di riferimento.

- Una matrice di rotazione R 3×3 ortogonale, che porta gli assi corrispondenti dei due sistemi di riferimento l'uno sopra l'altro.

Proprietà generali della matrice di rotazione

La rotazione di un vettore v in un vettore v' può essere rappresentata tramite una trasformazione lineare definita da una matrice di rotazione R 3×3

$$v' = Rv$$

soggetta ai vincoli

$$RR^T = R^T R = I \quad \det(R) = 1$$

con I matrice identità. Il primo vincolo è dato dal fatto che la matrice *inversa* è identica alla matrice *trasposta*. Il secondo vincolo afferma che *la trasformazione preserva il relativo orientamento del sistema di riferimento, senza modificarlo.*
Una matrice R per la quale vale $RR^T = I$ è detta *ortogonale* può essere meglio apprezzata assumendo che v e v' sono espressi in due differenti sistemi di riferimento, definiti attraverso i versori $[e_1, e_2, e_3]$ e $[e'_1, e'_2, e'_3]$. Si nota facilmente che il generico elemento r_{ij} di R è il *coseno dell'angolo formato tra il versore e_i e il versore ruotato e'_j*

$$r_{ij} = e_i^T e'_j$$

quindi abbiamo che

$$\sum_{j=1}^{3} r_{ij} r_{kj} = \sum_{j=1}^{3} r_{ji} r_{jk} = \begin{cases} 0 & i \neq k \\ 1 & i = k \end{cases} \tag{2.2}$$

ovvero, le righe (e colonne) di R *sono versori mutuamente ortogonali.* Una matrice 3×3 ortogonale ha nove elementi che devono soddisfare sei vincoli di ortogonalità (vedi eq. (2.2)). Questo riduce i *gradi di libertà* di una rotazione 3D a tre (9-6=3). La relazione tra le coordinate di un punto P_w nel sistema di riferimento mondo e del punto P_c nel sistema di riferimento della telecamera è

$$P_c = R(P_w - T) \tag{2.3}$$

con

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.4}$$

2.2.2 Parametri intrinseci

I parametri intrinseci possono essere definiti come il set di parametri necessari a descrivere le caratteristiche ottiche, geometriche e digitali di una telecamera.
Per il modello del *foro stenopeico* abbiamo tre set di parametri:

- Lunghezza focale f.
- Trasformazione tra coordinate della telecamera e coordinate pixel.
- Distorsione geometrica introdotta dall'ottica della telecamera.

Dalle coordinate della telecamera alle coordinate pixel

Per trovare il secondo set di parametri, dobbiamo collegare le coordinate (x_{im}, y_{im}) di un punto dell'immagine in unità pixel con le coordinate (x, y) dello stesso punto in coordinate della telecamera. Le coordinate (x_{im}, y_{im}) possono essere pensate come coordinate di un nuovo sistema di riferimento, molto spesso chiamato sistema di riferimento dell'immagine.
Trascurando qualsiasi possibile distorsione geometrica introdotta dall'ottica, abbiamo:

$$x = -(x_{im} - o_x)s_x \qquad y = -(y_{im} - o_y)s_y \tag{2.5}$$

con

- (o_x, o_y) coordinate pixel del centro dell'immagine (il punto principale),
- (s_x, s_y) dimensioni del pixel nell'unità di misura della telecamera rispettivamente per x e y,

Il segno negativo in (2.5) è dato dal fatto che gli assi x (orizzontali) e y (verticali) nel sistema di riferimento dell'immagine e nel sistema di riferimento della telecamera hanno orientazione opposta.
Possiamo dire quindi che i parametri intrinseci sono: f, o_x, o_y, s_x, s_y.

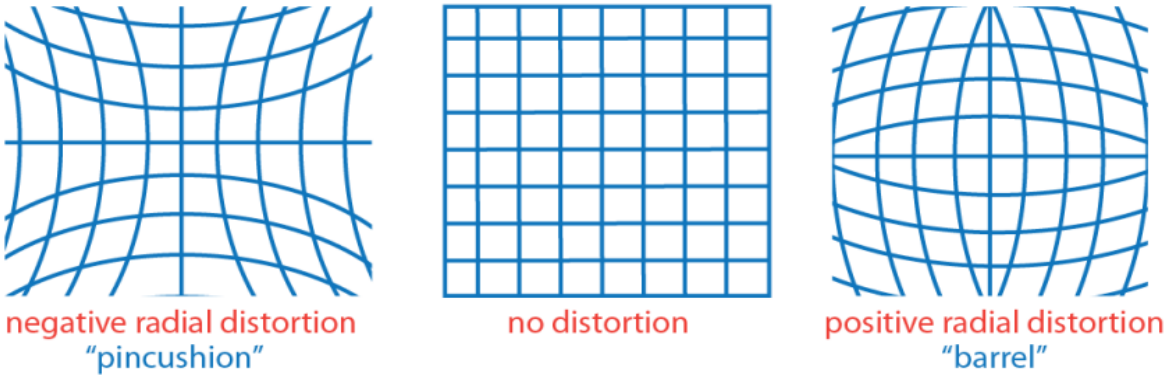

Figura 2.2: Tipi di distorsione: radiale positiva e negativa, nessuna distorsione

Distorsione introdotta dall'ottica

In molti casi, l'ottica introduce distorsioni nell'immagine che risultano evidenti specialmente nella regione periferica dell'immagine. Fortunatamente, queste distorsioni possono essere modellate abbastanza accuratamente come semplice *distorsione radiale*, in accordo alle relazioni

$$x = x_d(1 + k_1 r^2 + k_2 r^4) \qquad y = y_d(1 + k_1 r^2 + k_2 r^4) \tag{2.6}$$

con

- (x_d, y_d) coordinate del punto distorto,
- $r^2 = x_d^2 + y_d^2$

Come mostrato dall'equazione sopra riportata, questa distorsione è un dislocamento radiale dei punti d'immagine. Non si ha distorsione al centro dell'immagine, ma portandoci verso la periferia si nota l'aumentare della distorsione che aumenta tanto più ci si allontana dal centro dell'immagine. k_1 e k_2 sono ulteriori parametri intrinseci. Fintanto che la distorsione risulta piccola o non abbiamo necessità di una più alta accuratezza, allora essa può essere ignorata, altrimenti dovremo tenerne conto. In aggiunta, per una maggiore accuratezza, alle relazioni (2.6) può essere aggiunto un fattore $k_3 r^6$ con k_3 ulteriore fattore di distorsione.

2.2.3 Modello completo

Aver descritto i parametri estrinseci e intrinseci ci consente di poter collegare in maniera diretta le coordinate pixel di un punto dell'immagine con le coordinate mondiali del punto corrispondente, senza riferimenti espliciti al sistema di riferimento della telecamera (vedi (2.1)).

Versione lineare delle equazioni di proiezione prospettica

Sostituendo nella (2.1) le (2.3) e (2.5) otteniamo

$$-(x_{im} - o_x)s_x = f\frac{R_1^T(P_w - T)}{R_3^T(P_w - T)} \qquad -(y_{im} - o_y)s_y = f\frac{R_2^T(P_w - T)}{R_3^T(P_w - T)} \tag{2.7}$$

dove R_i con $i = 1, 2, 3$ è un vettore 3D formato dall'i-esima riga della matrice R. Infatti, la relazione (2.7) lega le coordinate 3D di un punto nel sistema di riferimento mondiale con le coordinate di un punto nell'immagine, attraverso i parametri intrinseci ed estrinseci. Possiamo riscrivere la (2.7) come semplice prodotto tra matrici.
A questo scopo, definiamo due matrici M_{int},M_{est}

$$M_{int} = \begin{bmatrix} -\frac{f}{s_x} & 0 & o_x \\ 0 & -\frac{f}{s_y} & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

e

$$M_{est} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & -R_1^T T \\ r_{21} & r_{22} & r_{23} & -R_2^T T \\ r_{31} & r_{32} & r_{33} & -R_3^T T \end{bmatrix}$$

In questo modo M_{int} dipende solo dai parametri intrinseci e M_{est} sol da quelli estrinseci.
Se a P_w aggiungiamo "1" come quarta coordinata, otteniamo P_w in coordinate omogenee[1]. Utilizzando le coordinate omogenee per P_w anche il punto ottenuto sarà in coordinate omogenee (come dimostrato di seguito). Utilizzando questa forma ed eseguendo il prodotto matriciale $M_{int}M_{est}P_w$ otteniamo un'equazione che descrive la proiezione prospettica

Equazione lineare matriciale

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = M_{int}M_{est} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

La cosa interessante che riguarda il vettore $[x_1, x_2, x_3]^T$ è il fatto che i rapporti x_1/x_3 e x_2/x_3 altro non sono che le coordinate immagine:

[1]introdotte da August Ferdinand Möbius intorno al 1837, sono uno strumento usato per descrivere i punti nella geometria proiettiva. Sono ampiamente usate nell'arte digitale per la rappresentazione di oggetti nello spazio e dei loro movimenti.

$$x_1/x_3 = x_{im}$$
$$x_2/x_3 = y_{im}$$

Il punto immagine è quindi anch'esso in coordinate omogenee

$$\begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix}.$$

È interessante notare che nella relazione matriciale è sparita la non linearità.
Inoltre, abbiamo separato i due passi della proiezione ***immagine-mondo***:

- M_{est} produce la trasformazione tra sistema di riferimento mondiale e sistema di riferimento della telecamera;
- M_{int} produce la trasformazione tra sistema di riferimento della telecamera e sistema di riferimento dell'immagine.

2.3 Spazio proiettivo

In geometria, lo spazio proiettivo è lo spazio ottenuto da uno spazio euclideo (ad esempio, la retta o il piano) aggiungendo i "punti all'infinito". A seconda della dimensione, si parla quindi di retta proiettiva, piano proiettivo, ecc. Lo spazio proiettivo è stato introdotto nel XVI secolo per modellizzare lo spazio visto dall'occhio umano, negli studi sulla prospettiva. Dal punto di vista geometrico, è uno spazio che presenta numerosi vantaggi rispetto a quello euclideo o affine: nello spazio proiettivo ci sono meno "casi particolari" da considerare.

2.4 Punti di fuga

Nella prospettica il punto di fuga (dall'inglese: Vanishing Point) è un punto verso il quale le linee parallele sembrano convergere. In particolare, il punto di fuga di una retta r è un punto F_r sul piano di proiezione, comune alle immagini prospettiche di ogni retta parallela a quella data. In altri termini, il punto di fuga di una retta è la proiezione del suo punto improprio (o punto all'infinito, o direzione).

Punto improprio
Un punto improprio è la direzione comune ad un fascio di rette parallele. Corrisponde ad un punto all'infinito dello spazio proiettivo, unico punto di intersezione di due delle suddette rette. In prospettiva un punto improprio r viene rappresentato sul quadro da un punto F_r.

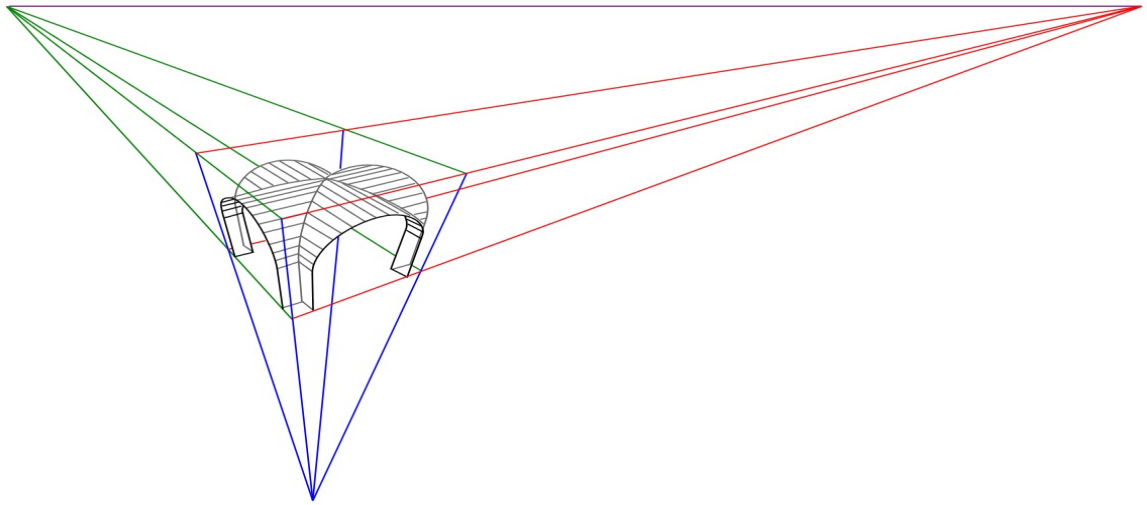

Figura 2.3: Punti di fuga

2.4.1 Fuga di un piano

Si può considerare anche la retta di fuga di un piano: questa contiene i punti di fuga di tutte le rette giacenti sul piano dato (o su piani paralleli al piano dato). Ad esempio, il punto di fuga di ogni retta orizzontale giace sulla linea dell'orizzonte. La retta di fuga di un piano parallelo al piano di proiezione è la retta all'infinito, passante per tutti i punti all'infinito. La retta di fuga di un piano non parallelo al piano di proiezione è l'intersezione tra il quest'ultimo ed il piano passante per il centro di proiezione e parallelo a quello dato.

2.5 Visione Stereoscopica

2.5.1 Sistema a doppia telecamera

Un sistema di *visione stereoscopica* necessita di almeno due telecamere, possibilmente identiche. La distanza base b è la traslazione di distanza tra i due centri di proiezione delle telecamere. Dopo la calibrazione delle due telecamere "parallele", la distanza di base è l'unico parametro rimanente che definisce la posa di una telecamera rispetto all'altra. Descriviamo ogni telecamera utilizzando il modello della PINHOLE CAMERA. Il sistema a doppia telecamera è caratterizzato dall'avere una copia identica della telecamera sinistra ma traslata di una quantità b lungo l'asse X_S nel sistema di coordinate (X_S, Y_S, Z_S) centrato nella telecamera sinistra. Si ha quindi che l'origine $(0, 0, 0)$ identifica il centro di proiezione della telecamera sinistra, mentre il centro di proiezione della telecamera destra avrà come coordinate $(b, 0, 0)$.
In altre parole abbiamo:

1. Due immagini complanari di uguale misura/risoluzione $N_{colonne} \times N_{righe}$.
2. Assi ottici paralleli.
3. Un'identica lunghezza focale f.
4. Righe delle due immagini collineari (la riga y di un'immagine è collineare alla riga y della seconda immagine).

Utilizzando le equazioni del centro di proiezione:

$$x_u = \frac{fX_S}{Z_S} \quad e \quad y_u = \frac{fY_S}{Z_S} \tag{2.8}$$

in cui (x_u, y_u) sono le coordinate di un punto p nel piano immagine e (X_S, Y_S, Z_S) sono le coordinate del punto P nel sistema di coordinate mondo, possiamo ricavare un punto p

$$p_L = (x_{uL}, y_{uL}) = \left(\frac{fX_S}{Z_S}, \frac{fY_S}{Z_S} \right) \tag{2.9}$$

$$p_R = (x_{uR}, y_{uR}) = \left(\frac{f(X_S - b)}{Z_S}, \frac{fY_S}{Z_S} \right) \tag{2.10}$$

rispettivamente, per la telecamera sinistra e destra. La calibrazione deve provvedere a valutare accuratamente b e f quando si usa un sistema di Stereo Vision.

Capitolo 3

Metodi di calibrazione

Introduciamo ora i due metodi di calibrazione.
Il primo è il metodo di calibrazione di Zhang, un metodo semplice in cui basta una telecamera.
Il secondo metodo è quello di Caprile e Torre che prevede l'utilizzo di due telecamere, dunque si parla di **Visione Stereoscopica**.

3.1 Calibrazione di Zhang

Il metodo di Zhang è un metodo di calibrazione semplice e prevede l'acquisizione, nella stessa immagine o in immagini successive, di pattern piani caratterizzati da giaciture diverse (traslati o ruotati). Il grosso vantaggio risiede nel fatto che non è necessario conoscere la posizione relativa di tali piani, quindi risulta essere un metodo estremamente pratico. Il metodo di Zhang è lineare, quindi non tiene in considerazione le distorsioni ottiche. L'autore ha proposto un approccio per risolvere anche la parte non lineare del problema, ma i risultati non sono ottimali. Seguendo la letteratura, risultano due le metodiche di ottimizzazione non lineare più utilizzate: la massima discesa di gradiente ed il Levenberg – Marquardt. Il metodo di Zhang è utilizzato per fornire una buona stima di primo tentativo all'algoritmo non lineare. Entrambe le metodiche sono state implementate e confrontate: il Levenberg – Marquardt garantisce generalmente risultati più robusti ed affidabili. Come prima cosa verranno proposte delle equazioni base che useremo e successivamente verrà descritto il metodo.

3.1.1 Equazioni base

Notazione

Un punto 2D è denotato da $m = [u, v]^T$. Un punto 3D è denotato da $M = [X, Y, Z]^T$. Si utilizzerà $\widetilde{x}$ per denotare il vettore omogeneo: $\widetilde{m} = [u, v, 1]^T$ e $\widetilde{M} = [X, Y, Z, 1]^T$.

Verrà utilizzato il modello del foro stenopeico: la relazione tra un punto 3D M e la sua proiezione nell'immagine m è data da

$$s\widetilde{m} = A \begin{bmatrix} R & t \end{bmatrix} \widetilde{M}, \qquad \text{con} \qquad A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

dove s è un **fattore di scala arbitrario**, (R, t) sono chiamati **parametri estrinseci** (R matrice di rotazione e t vettore di traslazione), A è chiamata **matrice intrinseca**, con (u_0, v_0) coordinate del punto principale, α e β **fattori di scala** per gli assi x e y dell'immagine e γ che è un parametro che descrive l'inclinazione tra i due assi dell'immagine. Utilizzeremo l'abbreviazione A^{-T} per $(A^{-1})^T$ o per $(A^T)^{-1}$

Omografia tra modello planare e sua immagine

Rinominiamo la i-esima colonna della matrice di rotazione R con r_i.
Assumere che il modello planare sia su $Z = 0$ nel sistema di coordinate mondo semplifica i passaggi che andremo a eseguire per determinare H, matrice omografica. Un'ulteriore giustificazione di questa scelta risiede nel fatto che la terza colonna della matrice di rotazione R (cioè r_3) si può calcolare grazie a $r_3 = r_1 \times r_2$, una volta determinati r_1 e r_2. Dalla (3.1) abbiamo

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = A \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{3.2}$$

Abusando della notazione, utilizziamo ancora M per denotare un punto nel modello planare, ma $M = [X, Y]^T$ in quanto Z è sempre equivalente a zero. Allo stesso modo $\widetilde{M} = [X, Y, 1]^T$. Perciò, un punto M e la sua immagine m sono legate dalla **matrice omografica** H.

$$s\widetilde{m} = H\widetilde{M} \qquad \text{con} \qquad H = A \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \tag{3.3}$$

Stima dell'omografia

Ci sono molti modi di stimare la matrice omografica, quello scelto è basato sul **metodo della massima verosimiglianza**[1]. Presi M_i e m_i i punti nel modello planare e i punti nel piano immagine, essi dovrebbero soddisfare la (3.3) (idealmente). Nella pratica non succede perché le immagini sono affette da rumore. Assumiamo quindi che m_i sia corrotto attraverso *rumore gaussiano* con media 0 e matrice di covarianza Λ_{m_i}. I punti M_i sono conosciuti mentre i punti m_i sono i punti ideali, conosciuti grazie al fatto che si conosce la geometria della scacchiera. Noi troveremo i punti $\hat{m}_i$ che si avvicinano maggiormente ai punti m_i. La stima della massima verosimiglianza di H viene ottenuta minimizzando la seguente

$$\sum_i (m_i - \hat{m}_i)^T \Lambda_{m_i}^{-1} (m_i - \hat{m}_i) \tag{3.4}$$

dove

$$\hat{m}_i = \frac{1}{\bar{h}_3^T M_i} \begin{bmatrix} \bar{h}_1^T M_i \\ \bar{h}_2^T M_i \end{bmatrix} \tag{3.5}$$

con $\bar{h}_i$ la i-esima riga di H.
In pratica, assumiamo che $\Lambda_{m_i} = \sigma^2 I$ per qualsiasi i. Questo è ragionevole se i punti sono estratti indipendentemente con la stessa procedura. In questo caso, il problema diventa un problema ai minimi quadrati non lineare, che è $\min_H \sum_i \|m_i - \hat{m}_i\|^2$. La minimizzazione non lineare è condotta con l'algoritmo di Levenberg-Marquardt (LMA), che richiede un'ipotesi iniziale, come segue:
Preso $x = [\bar{h}_1^T, \bar{h}_2^T, \bar{h}_3^T]^T$, allora la (3.3) può essere riscritta come

$$\begin{bmatrix} \widetilde{M}^T & 0^T & -u\widetilde{M}^T \\ 0^T & \widetilde{M}^T & -v\widetilde{M}^T \end{bmatrix} x = 0 \tag{3.6}$$

Quando ci vengono dati n punti, avremo n di queste equazioni, che possono essere riscritte con un'equazione matriciale del tipo $Lx = 0$, dove L è una matrice $2n \times 9$. Per come x è stato definito, la soluzione è ben nota per essere il giusto vettore singolare di L associato con il più piccolo valore singolare (o equivalentemente, l'autovettore di $L^T L$ associato al più piccolo autovalore). In L, alcuni elementi sono delle costanti (per la precisione valgono 1),alcuni sono in pixel, alcuni sono in coordinate mondo e alcuni

[1] Dall'inglese *maximum-likelihood criterion*

sono moltiplicazioni di entrambi. Questo rende L numericamente mal condizionata[2] (si veda [1]). Migliori risultati possono essere ottenuti eseguendo una normalizzazione dei dati prima di eseguire questa procedura.

Vincoli nei parametri intrinseci

Definiamo $H = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix}$.
Dall'equazione (3.3) abbiamo

$$\begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = \lambda A \begin{bmatrix} r_1 & r_2 & t \end{bmatrix}$$

dove λ è uno scalare arbitrario. Sapendo che r_1 e r_2 sono ortonormali, abbiamo

$$h_1^T A^{-T} A^{-1} h_2 = 0 \tag{3.7}$$
$$h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \tag{3.8}$$

Questi sono i due vincoli base nei parametri intrinseci, data un'omografia. Siccome un'omografia ha otto gradi di libertà e ci sono sei parametri estrinseci (tre per rotazione e tre per traslazione), possiamo ottenere solamente due vincoli nei parametri intrinseci. Da notare che $A^{-T}A^{-1}$ descrive l'immagine della conica assoluta. La conica assoluta è una conica di punti complessi data dall'intersezione tra una qualunque sfera e il piano all'infinito.
Mediante una similitudine una sfera è trasformata in una (altra) sfera: ma la sua intersezione con il piano all'infinito resta invariata e pari alla conica assoluta. Anche l'immagine della conica assoluta pertanto resta costante per effetto di similitudini applicate al mondo rispetto alla telecamera: dunque l'immagine della conica assoluta non varia per effetto di spostamento/rotazione della telecamera. Essa dipende solo dai parametri intrinseci della telecamera, quindi da $A^{-T}A^{-1}$.

Interpretazione geometrica dei vincoli

Il modello planare è descritto nel sistema di coordinate della telecamera attraverso la seguente equazione:

[2]Il condizionamento in matematica, in particolare nel calcolo numerico, riguarda il rapporto tra errore commesso sul risultato di un calcolo e incertezza sui dati in ingresso. Un problema è ben condizionato quando la soluzione del problema con delle piccole variazioni, non differisce molto dalla soluzione del problema originale; al contrario, un problema mal condizionato è un problema dove le soluzioni sono molto sensibili a piccole perturbazioni dei dati iniziali.

$$\begin{bmatrix} r_3 \\ r_3^T t \end{bmatrix}^T \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = 0$$

dove $w = 0$ per punti all'infinito e $w = 1$ per tutti gli altri. Questo piano interseca il piano all'infinito in una linea e possiamo notare facilmente che

$$\begin{bmatrix} r_1 \\ 0 \end{bmatrix}$$

e

$$\begin{bmatrix} r_2 \\ 0 \end{bmatrix}$$

sono due punti particolari di questa linea.
Qualsiasi punto che giace su di essa, è una combinazione lineare di questi due punti.

$$x_\infty = a \begin{bmatrix} r_1 \\ 0 \end{bmatrix} + b \begin{bmatrix} r_2 \\ 0 \end{bmatrix} = \begin{bmatrix} ar_1 + br_2 \\ 0 \end{bmatrix}$$

Dalla definizione, il punto x_∞, noto come **punto ciclico**[3], soddisfa: $x_\infty^T x_\infty = 0$, cioè $(ar_1 + br_2)^T(ar_1 + br_2) = 0$ oppure $a^2 + b^2 = 0$. La soluzione è $b = \pm ai$. I due punti di intersezione sono

$$x_\infty = a = \begin{bmatrix} r_1 \pm ir_2 \\ 0 \end{bmatrix}$$

L'importanza di questa coppia di punti complessi coniugati sta nel fatto che sono invarianti con le trasformazioni euclidee. La loro proiezione nel piano immagine è data da

$$\widetilde{m}_\infty = A(r_1 \pm ir_2) = h_1 \pm ih_2$$

Il punto $\widetilde{m}_\infty$ è nell'immagine della conica assoluta, descritta da $A^{-T}A^{-1}$.
Questo da

$$(h_1 \pm ih_2)^T A^{-T} A^{-1} (h_1 \pm ih_2) = 0$$

Questo richiede che entrambe le parti (reale e immaginaria) nelle equazioni (3.7) e (3.8) siano annullate.

[3]Dall'inglese: *cyclic point*. Si dicono punti ciclici i due punti immaginari impropri di coordinate $x = 1; y = \pm i; z = 0 \quad \text{dove} \quad i = \sqrt{-1}$

3.1.2 Calibrazione

In questa sezione saranno dati tutti i dettagli di come effettivamente si risolve la calibrazione della telecamera. Partiamo con una soluzione analitica, seguita da una tecnica di ottimizzazione non lineare basata sul metodo della massima verosimiglianza. Per finire, terremo in considerazione la distorsione dell'ottica, dando sia la soluzione analitica che quella non lineare.

Soluzione in forma chiusa

Preso

$$B = A^{-T}A^{-1} \equiv \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

cioè

$$= \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix} \tag{3.9}$$

Si può notare facilmente che B è simmetrica e definita da un vettore

$$b = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T \tag{3.10}$$

Presa la i-esima colonna della matrice omografica H, $h_i = [h_{i1}, h_{i2}, h_{i3}]^T$, abbiamo

$$h_i^T B h_j = v_{ij}^T b \tag{3.11}$$

con $v_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T$ I due vincoli fondamentali (3.7) e (3.8), dati da un'omografia, possono essere riscritti come due equazioni omogenee in b:

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \tag{3.12}$$

Se vengono osservate n immagini di un modello planare, impilando n equazioni come la (3.12), otteniamo

$$Vb = 0 \tag{3.13}$$

dove V è un matrice $2n \times 6$. Se $n \geq 3$ avremo, in generale, un'unica soluzione di b. Se $n = 2$ possiamo imporre che lo *skew* sia nullo ($\gamma = 0$), cioé $[0, 1, 0, 0, 0, 0]b = 0$ a cui è aggiunta un'ulteriore equazione alla (3.13). Se $n = 1$ possiamo ottenere solo due parametri intrinseci, come α e β, assumendo che u_0 e v_0 siano gia noti e $\gamma = 0$. Una volta stimato il vettore b, possiamo calcolare tutti i parametri intrinseci. La matrice B viene stimata, cioè $B = \lambda A^{-T} A^{-1}$ con λ fattore di scala arbitrario. Senza grosse difficoltà possiamo unicamente estrarre i parametri intrinseci dalla matrice B.

- $v_0 = (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2)$
- $\lambda = B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11}$
- $\alpha = \sqrt{\lambda/B_{11}}$
- $\beta = \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)}$
- $\gamma = -B_{12}\alpha^2\beta/\lambda$
- $u_0 = \gamma v_0/\alpha - B_{13}\alpha^2/\lambda$

Una volta ottenuto A possiamo calcolare i parametri estrinseci per ogni immagine. Dalla (3.3) abbiamo
$r_1 = \lambda A^{-1}h_1 \quad r_2 = \lambda A^{-1}h_2 \quad r_3 = r_1 \times r_2 \quad t = \lambda A^{-1}h_3$
con $\lambda = 1/\|A^{-1}h_1\| = 1/\|A^{-1}h_2\|$. A causa del rumore nei dati, la matrice R non soddisfa, in generale, le proprietà della matrice di rotazione. Una matrice di rotazione migliore può essere ottenuta attraverso **SVD**[4].

Metodo della massima verosimiglianza

Prese n immagini di un modello planare nel quale si hanno m punti. Assunto che i punti nell'immagine siano affetti da rumore, possiamo usare il metodo della massima verosimiglianza per minimizzare il rumore tramite la seguente funzione:

[4]Dall'inglese: *Singular Value Decomposition*. La decomposizione ai valori singolari è una particolare fattorizzazione di una matrice che generalizza il concetto di autovalori e autovettori.

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\|m_{ij} - \hat{m}(A, R_i, t_i, M_j)\|^2 \tag{3.14}$$

dove $\hat{m}(A, R_i, t_i, M_j)$ è la proiezione del punto M_j nell'immagine i, in accordo con l'equazione (3.3) Una rotazione R è parametrizzata tramite un vettore composto da tre parametri r. R e r sono correlati grazie alla formula di Rodrigues[5]. Minimizzare la (3.14) è un problema di minimizzazione non lineare, il quale è risolto tramite l'LMA (Levenberg-Marquardt Algorithm). L'algoritmo richiede delle ipotesi iniziali di A e R_i, t_i (con $i = 1 \dots n$) che possono essere ottenute tramite la tecnica descritta precedentemente.

Sommario

La procedura di calibrazione raccomandata è la seguente:

1. Stampare un pattern (una scacchiera) è incollarla a una superficie rigida e planare.
2. Scattare varie fotografie del modello planare.
3. Individuare i punti caratteristici nell'immagine.
4. Stimare i cinque parametri intrinseci e tutti gli estrinseci usando la soluzione in forma chiusa, come descritta nell'apposita sezione.
5. Affinare tutti i parametri, inclusi i parametri di distorsione.

3.2 Calibrazione basata sui punti di fuga

3.2.1 Metodo di calibrazione di Caprile e Torre

Il metodo di Caprile e Torre (1990) è un metodo di calibrazione a due passi che utilizza le proprietà dei punti di fuga per ricavare i parametri intrinseci e estrinseci. Esso utilizza inoltre due telecamere, possibilmente identiche. Si ha quindi una **calibrazione stereo**. Vengono scattate varie coppie di fotografie per ottenere una calibrazione accurata. Le due telecamere, una a destra e l'altra a sinistra, scatteranno allo stesso istante, in modo da ottenere le coppie che verranno poi utilizzate per calcolare i parametri. L'utilizzo dei punti di fuga è motivato dal fatto che la determinazione della matrice di rotazione R è immediata; Essa si calcola prima del vettore di traslazione T, che si ottiene successivamente per semplice triangolazione. La semplicità nel ricavare i due

[5]La formula di Rodrigues è un efficiente algoritmo per ruotare un vettore nello spazio, dato un asse e un angolo di rotazione.

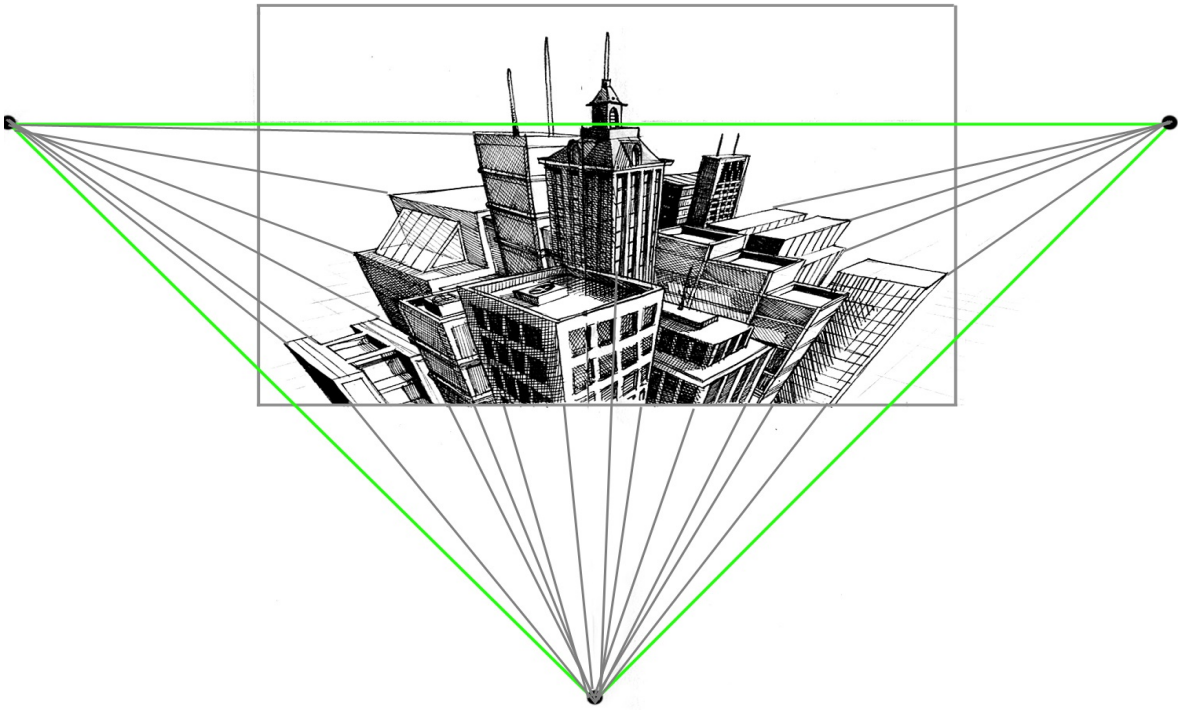

Figura 3.1: Immagine costruita mediante l'utilizzo di tre punti di fuga

parametri estrinseci è quindi il punto di forza del metodo. Di seguito sono esposti i due passi per la calibrazione:

1. Sono ricavati i parametri intrinseci di tutte e due le telecamere da una singola immagine di un cubo. Ricordiamo che i parametri intrinseci sono:
   - Lunghezza focale, espressa in X-pixel, che chiameremo k_2.
   - Le coordinate pixel dell'intersezione tra asse ottico e piano immagine, a e b.
2. I parametri estrinseci della coppia di telecamere sono stimati da una coppia stereo di immagini di un modello planare adatto. Ricordiamo che i parametri intrinseci sono:
   - Matrice di rotazione R.
   - Vettore di traslazione T.

Proprietà dei punti di fuga.

In questa sezione introdurremo alcune proprietà dei punti di fuga, utili allo scopo della calibrazione della telecamera.
Presi x, y, z siano essi gli assi di un sistema di riferimento ortogonale associato a una telecamera, in modo tale che l'origine del sistema di riferimento coincida con il centro

di proiezione (o centro della lente) e l'asse z coincida con l'asse ottico.
Il piano immagine viene definito dall'equazione $z{=}f$ dove f è la lunghezza focale. In questo modo, le coordinate X, Y (espresse in pixel) di un punto q_p nel piano immagine e proiezione prospettica del punto Q_p nello spazio sono:

$$\begin{cases} x = \frac{f}{k_1}(Y_1 - b) \\ y = \frac{f}{k_2}(X_1 - a) \end{cases} \qquad k_1, k_2 > 0 \tag{3.15}$$

dove:

- k_1 e k_2 sono lunghezze focali del dispositivo in Y-pixel e X-pixel.
- La lunghezza focale f è in mm.
- a, b sono le coordinate pixel dell'intersezione tra asse ottico e piano immagine.

È evidente che la relazione (3.15) può essere usata come relazione tra coordinate pixel X, Y e coordinate x, y solamente se son noti $\frac{f}{k_1}$, $\frac{k_1}{k_2}$, a,b.

Riprendiamo il concetto di punto di fuga e altre proprietà utili della proiezione prospettica.
Consideriamo una retta S in $\mathbb{R}^3$, passante in un punto (a_x, a_y, a_z) parametrizzato come segue:

$$\begin{cases} x = a_x + tn_x \\ y = a_y + tn_y \\ z = a_z + tn_z \end{cases} \tag{3.16}$$

dove $n = (n_x, n_y, n_z)$ è il vettore direttore di S e t è un parametro.
Il punto di fuga della retta S è il punto $v_S = (x_\infty, y_\infty, z_\infty)$ nel piano immagine, dove f è la distanza focale:

$$\begin{cases} x_\infty = \lim_{t\to\infty} f\frac{a_x+tn_x}{a_z+tn_z} = f\frac{n_x}{n_z} \\ y_\infty = \lim_{t\to\infty} f\frac{a_y+tn_y}{a_z+tn_z} = f\frac{n_y}{n_z} \\ z_\infty = \lim_{t\to\infty} f\frac{a_z+tn_z}{a_z+tn_z} = f \end{cases} \tag{3.17}$$

DEFINIZIONE
Dato $q = (x, y, f)$, sia esso un punto nel piano immagine e sia $\overline{v_q} = (\frac{x}{h}, \frac{y}{h}, \frac{z}{h})$ il vettore normalizzato associato a q, dove $h = \sqrt{x^2 + y^2 + f^2}$. Presa una retta S nello spazio e il suo punto di fuga v_s, segue che il vettore associato a v_s è identico a n_s, che è il vettore direttore di S (se entrambe puntano al piano immagine).

PROPRIETÀ

Le proprietà utili al nostro scopo sono tre. Le prime due proprietà ci servono per poter ricavare i parametri estrinseci mentre l'ultima ci fornisce uno dei parametri intrinseci.

PROPRIETÀ 1.
Sia dato $A = \{S_i\}$ un insieme di rette (non parallele al piano immagine).
L'insieme $\alpha = \{v_{s_i}\}$ dei punti di fuga associati agli elementi di A giace su una retta se e solo se le rette in A sono parallele allo stesso piano Π_A.

Dimostrazione:
Se i punti di fuga giacciono su una retta l, è evidente che le rette in A sono parallele al piano Π_A che passa attraverso l'origine, centro della proiezione, e la linea l.
Al contrario, se le rette in A sono parallele allo stesso piano Π_A, e Π_α è il piano passante per l'origine, è evidente che Π_α contiene i punti di fuga associati a ciascuna retta in A. L'intersezione tra Π_α e il piano immagine, è una retta che contiene tutti i punti di fuga.

PROPRIETÀ 2.
Siano Q, R, S tre rette nello spazio, ortogonali tra di loro e siano v_Q=(x_Q, y_Q, f), v_R=(x_R, y_R, f), v_S=(x_S, y_S, f), i tre punti di fuga relativi a esse.
Se conosciamo le coordinate di un punto di fuga, sia esso v_Q, e la direzione di una retta nel piano immagine che passa attraverso un secondo punto, sia esso v_R, allora è possibile ricavare le coordinate degli altri due punti v_R, v_S.

Dimostrazione:
Siano l, m, n i vettori associati ai punti di fuga v_Q, v_R, v_S.
Essendo le rette ortogonali tra di loro si ha:

$$\begin{cases} l \cdot m = 0 \\ l \cdot n = 0 \\ m \cdot n = 0 \end{cases} \tag{3.18}$$

Si ottengono tre equazioni:

$$\begin{cases} x_Q x_R + y_Q y_R + f^2 = 0 \\ x_Q x_S + y_Q y_S + f^2 = 0 \\ x_R x_S + y_R y_S + f^2 = 0 \end{cases} \tag{3.19}$$

Avendo tre equazioni e quattro incognite, risulta immediato andare alla ricerca di una quarta equazione che leghi in qualche modo due punti di fuga. La strada più semplice

è quella di pensare all'equazione di una retta passante per due punti.

$$y - y_Q = m(x - x_Q) \tag{3.20}$$

A questo punto, si impone il passaggio della retta attraverso il secondo punto di fuga, in questo caso prendiamo il punto v_R.
Si ha dunque

$$y_R - y_Q = m(x_R - x_Q) \tag{3.21}$$

Riscrivendo l'equazione (3.21) in maniera più comoda, essa è utile al calcolo in quanto elimina dalle equazione (3.19) un'incognita, sia essa x_R o y_R.

$$x_R = \frac{y_R - y_Q}{m} + x_Q \quad oppure \quad y_R = m(x_R - x_Q) + y_Q \tag{3.22}$$

Noti il punto $v_Q = (x_Q, y_Q)$ e il coefficiente angolare m, l'equazione (3.21) è la quarta equazione utile alla risoluzione delle coordinate dei punti di fuga.

PROPRIETÀ 3.
Siano Q, R, S tre rette nello spazio, ortogonali tra di loro e v_Q=(x_Q, y_Q, f), v_R=(x_R, y_R, f), v_S=(x_S, y_S, f) i tre punti di fuga associati.
L'ortocentro del triangolo con vertici nei tre punti di fuga è l'intersezione dell'asse ottico con il piano immagine.

Dimostrazione:
L'equazione della retta T_Q passante attraverso v_Q e ortogonale al segmento che congiunge v_R e v_S è:

$$y - y_Q = \frac{x_S - x_R}{y_R - y_S}(x - x_Q) \tag{3.23}$$

Dalle equazioni (3.19) e (3.15) segue immediatamente che T_Q passa attraverso $X = a$, $Y = b$.
Il medesimo procedimento si applica agli altri punti di fuga.

Parametri estrinseci: MATRICE DI ROTAZIONE.

Uno spostamento rigido o equivalentemente, la relativa posizione e orientamento di una coppia di telecamere, è caratterizzato da una matrice di rotazione R 3×3 ortogonale e da un vettore di traslazione T. Rappresentano la rotazione tra vecchie e nuove coordinate e la traslazione dell'origine. Per definizione, due rette parallele hanno lo stesso punto di fuga, quindi, il punto di fuga (e il suo vettore associato) non modifica le sue coordinate se si effettua una traslazione rigida del sistema di visualizzazione. Da que-

sto abbiamo che i cambiamenti di posizione dei punti di fuga per movimenti arbitrari della telecamera, o della coppia di telecamere, sono dovute ai relativi componenti di rotazione.
Dall'algebra lineare, abbiamo che la trasformazione di coordinate di un vettore a cui è associato un punto di fuga, per un movimento arbitrario tra sistemi di coordinate è espresso dalla relazione:

$$v' = Rv \tag{3.24}$$

Supponiamo che le coordinate dei tre punti di fuga che non stanno sulla stessa retta siano conosciuti. Se consideriamo una matrice V 3×3 le cui colonne sono tre vettori v_i (con $i = (1, 2, 3)$) associati ai tre punti di fuga, allora possiamo scrivere:

$$V' = RV \tag{3.25}$$

L'equazione (3.25) è un sistema di equazioni per gli elementi di R che può essere risolta se V è invertibile.
È ovvio che

$$|\det V| = v_1 \cdot (v_2 \wedge v_3) \neq 0 \tag{3.26}$$

se i tre vettori sono non complanari, peraltro sappiamo dalla Proprietà 1 che i punti di fuga con direzioni non complanari non sono allineati. Il problema è ridotto alla risoluzione delle coordinate dei tre punti di fuga non allineati in due immagini. Nel caso di tre punti di fuga associati a tre rette ortogonali, la matrice V è ortonormale e si ottiene:

$$R = V'V^T \tag{3.27}$$

dove V^T è la matrice trasposta di V.
L'equazione (3.27) è estremamente semplice e si ricava facilmente R.

Parametri estrinseci: VETTORE DI TRASLAZIONE.

Una volta ottenuta la matrice di rotazione R, è possibile recuperare il vettore di traslazione T dalla corrispondenza delle proiezioni di segmenti di linea di lunghezza nota e dall'orientazione in due istanti di tempo diversi. Questa procedura è una semplice triangolazione.
Dato P, sia esso il vettore avente le coordinate di un punto p nello spazio, tenendo presente che il sistema di riferimento è quello dato della nostra telecamera. Se la telecamera si muove, esso cambierà.
La relazione che descrive questo cambiamento è la seguente:

$$P' = R(P - T) \tag{3.28}$$

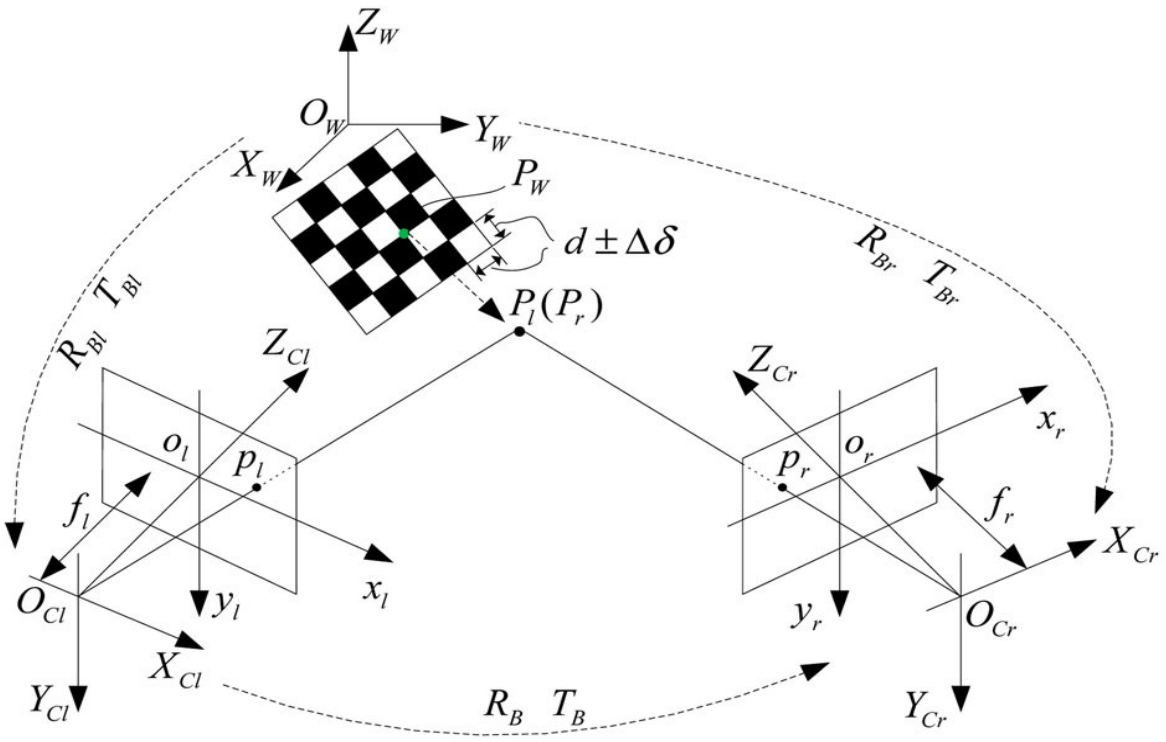

Figura 3.2: Modello di calibrazione stereo

Ricordiamo che R è la matrice di rotazione e T è il vettore traslazione.
Una volta ottenuto R, se conosciamo i punti P, P' possiamo ricavare T:

$$T = P - R^T P' \tag{3.29}$$

Parametri intrinseci.

I parametri intrinseci sono:

- Lunghezza focale espressa in X-pixel, che chiamiamo k_2
- Il rapporto $\frac{k_1}{k_2}$ tra il valore della lunghezza focale in Y-pixel e quello in X-pixel
- Le coordinate pixel dell'intersezione tra asse ottico e piano immagine, che chiamiamo a, b

Il valore di $\frac{k_1}{k_2}$ si ottiene semplicemente dalla matrice del fotosensore del dispositivo. Prendiamo un cubo e lo posizioniamo in modo tale da visualizzare solamente tre facce. In ogni faccia avremo varie rette parallele tra loro. In questo modo avremo tre famiglie di rette ortogonali tra loro, e avremo tre punti di fuga, uno per ogni famiglia di rette parallele e quindi uno per ogni faccia visibile.
Data un'immagine del cubo, è possibile localizzare i tre punti di fuga nel piano immagine e tramite la Proprietà 3 ricavare i parametri intrinseci mancanti.

Grazie all'ortogonalità, viene soddisfatta l'equazione (3.18) dove l, m, n, sono associati ai tre punti di fuga.
In tal modo, se (X_i, Y_i) (con $i = (1, 2, 3)$) sono le coordinate pixel dei punti di fuga, dall'equazione (3.18) e (3.15) dopo alcuni passaggi otteniamo:

$$\begin{cases} (X_3 - X_1)a + k^2(Y_3 - Y_1)b + (X_1 - X_3)X_2 + k^2(Y_1 - Y_3)Y_2 = 0 \\ (X_3 - X_2)a + k^2(Y_3 - Y_2)b + (X_2 - X_3)X_1 + k^2(Y_2 - Y_3)Y_1 = 0 \end{cases} \tag{3.30}$$

dove $k = \frac{k_1}{k_2}$ Si possono quindi ricavare a, b.
Per ottenere una buona accuratezza nella localizzazione dei punti di fuga, è consigliato disporre il cubo in una posizione tale che le facce del cubo visibili formino un angolo di 45 gradi con il piano immagine.

3.2.2 Ottimizzazione del metodo

Il metodo che sfrutta i punti di fuga è stato via via migliorato, specialmente per quel che concerne l'accuratezza della localizzazione dei punti di fuga.
Di recente, H. Sun, J. Lu e Z. Chang hanno ottimizzato l'utilizzo dei punti di fuga, proponendo un articolo dettagliato.
Di seguito sono elencati i vari punti cardine del loro lavoro.

1. Scattare molte immagini di una scacchiera tramite due telecamere nello stesso istante.
2. Estrarre gli angoli tramite l'algoritmo di Harris.
3. Adattare le rette parallele tramite il metodo dei minimi quadrati.
4. Ottimizzare i punti di fuga tramite l'LMA (Levenberg-Marquardt Algorithm).
5. Calcolo dei parametri estrinseci/intrinseci a seconda della natura dei punti di fuga e calcolare i parametri estrinseci tramite il metodo di Tsai.
6. Ottimizzazione del risultato di calibrazione minimizzando il residuo d'immagine.
7. Valutazione dell'errore di calibrazione di ciascuna telecamera, ripetendo l'intero processo se l'errore non è trascurabile.
8. Calcolo dei parametri stereo, calcolo dei punti 3D in coordinate mondiali.
9. Ottimizzazione del risultato di calibrazione minimizzando la differenza di distanza degli angoli.
10. Valutazione dell'errore di calibrazione stereo. Se i risultati sono accettabili, la calibrazione è finita.

Nel modello prospettico, due rette parallele (o più di due, ma sempre parallele tra di loro) si intersecheranno in un punto. Quest'ultimo è chiamato punto di fuga.
Considerando una scacchiera, possono essere creati quattro gruppi di rette parallele. Ogni gruppo avrà quindi un punto di fuga in cui si intersecano tutte le rette di un gruppo e in totale possiamo avere quattro punti di fuga.
Questi gruppi di rette sono:

- Rette orizzontali,
- Rette verticali,
- Rette della diagonale principale,
- Rette della vice diagonale;

Teoricamente, le rette di ogni gruppo si incontrano in un solo punto di fuga.
I quattro punti di fuga sono:

- V_{or}
- V_{ver}
- V_{Pdiag}
- V_{Vdiag}

I quattro punti di fuga giacciono tutti su una retta L_V, se le trasformazioni proiettive sono prese in considerazione.
Normalmente l'errore di calibrazione è determinato dalla precisione delle rette parallele e dal punto di fuga ottimo. Discuteremo qui di seguito questi aspetti:

Adattamento delle rette parallele

Supposto un punto d'angolo della scacchiera $p_{i,j} = [u_{ij}, v_{ij}]^T$ giacente nella riga i e nella colonna j nel piano immagine, le sue coordinate omogenee sono $\hat{p}_{i,j} = [u_{ij}, v_{ij}, 1]^T$ con $i = 1, 2, \ldots, m$ e $j = 1, 2, \ldots, n$.
L'espressione dei parametri del modello della retta orizzontale $v_{ij} = a^i_{or} u_{ij} + b^i_{or}$ da risolvere è $[a^i_{or}, -1, b^i_{or}]$.
L'equazione (3.31) dà una stima ai minimi quadrati:

$$\begin{aligned} E^i_{LS} &= \varepsilon(a^i_{or}, b^i_{or}) \\ &= \sum_{j=1}^{n} ([a^i_{or}, -1, b^i_{or}][u_{ij}, v_{ij}, 1]^T)^2 \\ &= \sum_{j=1}^{n} (a^i_{or} u_{ij} + b^i_{or} - v_{ij})^2 \end{aligned} \tag{3.31}$$

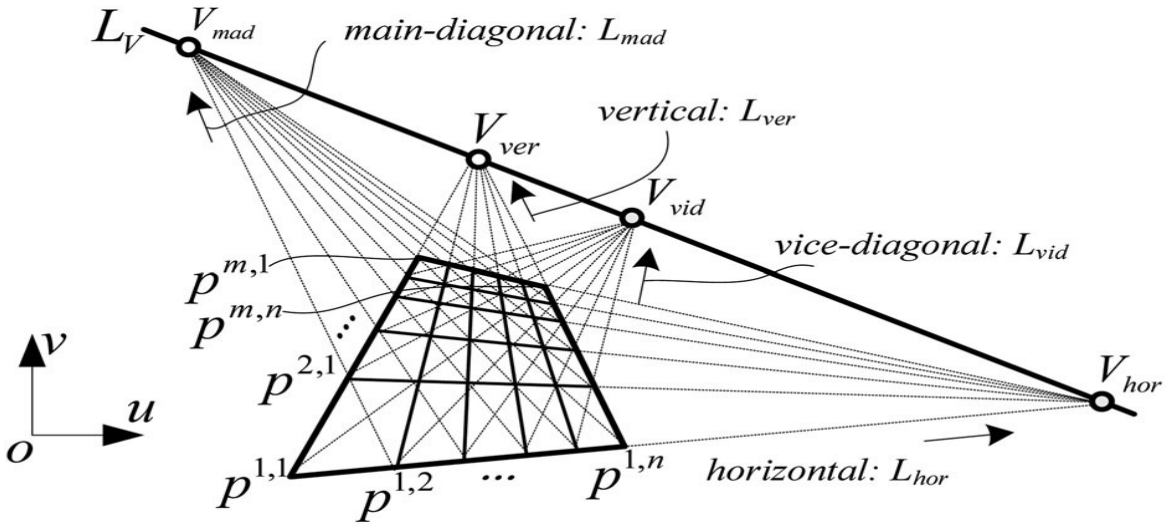

Figura 3.3: Modello dei punti di fuga

Risolvendo l'equazione matriciale, si ottiene il parametro di retta ottima nota come $l^i_{or} = (a^i_{or}, -1, b^i_{or})$. In maniera identica si ottengono gli altri tre parametri di retta ottima:

- $l^j_{ver} = (a^j_{ver}, -1, b^j_{ver})$ per la rette verticali,
- $l^p_{Pdiag} = (a^p_{Pdiag}, -1, b^p_{Pdiag})$ per le rette diagonali,
- $l^q_{Vdiag} = (a^q_{Vdiag}, -1, b^q_{Vdiag})$ per le rette vice diagonali.

Punto di fuga ottimo

A causa di vari fattori, le rette di ogni gruppo non si intersecano nel punto di fuga teorico, e ci si ritrova con tanti (seppur vicini) punti di fuga per ogni gruppo di rette. Il punto di fuga ottimo significa il punto di intersezione più vicino a ciascuna retta nella stessa direzione. Prendendo a titolo di esempio la direzione orizzontale, il punto di fuga ottimo è determinato dalla seguente funzione:

$$\begin{aligned} F_{or} &= \underset{l^i_{or}}{\operatorname{argmin}} \sum_{i=1}^{m} \operatorname{dis}(V_{or}(u_{or}, v_{or}), l^i_{or}(a^i_{or}, -1, b^i_{or})) \\ &= \underset{l^i_{or}}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{|a^i_{or} u_{or} - v_{or} + b^i_{or}|}{\sqrt{{a^i_{or}}^2 + (-1)^2}} \end{aligned} \tag{3.32}$$

Qui, l'algoritmo di Levenberg-Marquard (LMA) è stato utilizzato per trovare il punto di fuga ottimo.
Similmente al metodo di Zhang, è possibile calibrare i seguenti parametri intrinseci:

- (u_0, v_0) - coordinate del punto in cui l'asse focale interseca il piano immagine,

- f_x e f_y - lunghezza focale in pixel lungo x e y, fattori di scala u e v nel piano immagine.

Viene invece utilizzato il metodo di Tsai per ricavare i parametri estrinseci.
Di seguito vengono esposti i passi da seguire.
Posizionare le coordinate mondo (O_W, X_W, Y_W, Z_W) nella scacchiera facendo coincidere il piano X_W, Y_W, O_W con la scacchiera. Da notare che Z_W è è perpendicolare alla scacchiera e vale $Z_W = 0$.
In accordo con il modello del foro stenopeico, vale il seguente sistema di equazioni

$$\begin{cases} u = f_x \dfrac{r_1 X_W + r_2 Y_W + t_x}{r_7 X_W + r_8 Y_W + t_z} + u_0 \\ \\ v = f_y \dfrac{r_4 X_W + r_5 Y_W + t_y}{r_7 X_W + r_8 Y_W + t_z} + v_0 \end{cases} \tag{3.33}$$

in cui

- (t_x, t_y, t_z) rappresentano il vettore di traslazione T,
- (X_W, Y_W) sono le coordinate del punto nel sistema di riferimento mondo,
- $(r_1, r_2, r_4, r_5, r_7, r_8)$ sono alcuni parametri della matrice di rotazione R, in particolare r_1, r_4, r_7 formano la prima colonna di R e r_2, r_5, r_8 formano la seconda colonna.

Il sistema (3.33) può essere trasformato in

$$\begin{bmatrix} f_x X_W & f_x Y_W & f_x & 0 & 0 & 0 & -U_x X_W & -U_x Y_W \\ 0 & 0 & 0 & f_y X_W & f_y Y_W & f_y & -V_x X_W & -V_x Y_W \end{bmatrix} b = \begin{bmatrix} U_x \\ V_x \end{bmatrix} \tag{3.34}$$

dove

- $b = \begin{bmatrix} b_1, & b_2, & b_3, & b_4, & b_5, & b_6, & b_7, & b_8 \end{bmatrix}^T$

$$b_1 = \frac{r_1}{t_z} \qquad b_2 = \frac{r_2}{t_z} \qquad b_3 = \frac{t_x}{t_z} \qquad b_4 = \frac{r_4}{t_z},$$
$$b_5 = \frac{r_5}{t_z} \qquad b_6 = \frac{t_y}{t_z} \qquad b_7 = \frac{r_7}{t_z} \qquad b_8 = \frac{r_8}{t_z}$$

- $U_x = u - u_0$
- $V_x = v - v_0$

sono dati.
Similarmente, come per gli $m \times n$ punti d'angolo, in un'immagine di calibrazione, l'equazione (3.34) può essere utilizzata per formare il seguente sistema lineare sovradeterminato,

$$\begin{bmatrix} f_x{}^1X_W & f_x{}^1Y_W & f_x & 0 & 0 & 0 & -U_x{}^1X_W & -U_x{}^1Y_W \\ 0 & 0 & 0 & f_y{}^1X_W & f_y{}^1Y_W & f_y & -V_x{}^1X_W & -V_x{}^1Y_W \\ f_x{}^2X_W & f_x{}^2Y_W & f_x & 0 & 0 & 0 & -U_x{}^2X_W & -U_x{}^2Y_W \\ 0 & 0 & 0 & f_y{}^2X_W & f_y{}^2Y_W & f_y & -V_x{}^2X_W & -V_x{}^2Y_W \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_x{}^{mn}X_W & f_x{}^{mn}Y_W & f_x & 0 & 0 & 0 & -U_x{}^{mn}X_W & -U_x{}^{mn}Y_W \\ 0 & 0 & 0 & f_y{}^{mn}X_W & f_y{}^{mn}Y_W & f_y & -V_x{}^{mn}X_W & -V_x{}^{mn}Y_W \end{bmatrix} b = \begin{bmatrix} {}^1U_x \\ {}^1V_x \\ {}^2U_x \\ {}^2V_x \\ \vdots \\ {}^{mn}U_x \\ {}^{mn}V_x \end{bmatrix} \tag{3.35}$$

che può essere rappresentato in forma matriciale

$$Fb = U \tag{3.36}$$

I parametri intrinseci ottenuti sono considerati come input di questa equazione. I minimi quadrati sono ancora utilizzati per trovare la migliore soluzione, che può essere calcolata tramite $b = (F^TF)^{-1}F^TU$. In accordo con le proprietà della matrice di trasformazione del corpo rigido, vale la seguente equazione $R^TR = RR^T = I$.

$$\begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}^T$$

$$= \begin{bmatrix} r_1^2 + r_2^2 + r_3^2 & r_1r_4 + r_2r_5 + r_3r_6 & r_1r_7 + r_2r_8 + r_3r_9 \\ r_1r_4 + r_2r_5 + r_3r_6 & r_4^2 + r_5^2 + r_6^2 & r_4r_7 + r_5r_8 + r_6r_9 \\ r_1r_7 + r_2r_8 + r_3r_9 & r_4r_7 + r_5r_8 + r_6r_9 & r_7^2 + r_8^2 + r_9^2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.37}$$

L'equazione (3.37) possiamo riscriverla e si ottiene

$$\begin{cases} r_1 r_4 + r_2 r_5 + (\pm\sqrt{1 - r_1^2 - r_2^2})(\pm\sqrt{1 - r_4^2 - r_5^2}) = 0 \\ r_4 r_7 + r_5 r_8 + (\pm\sqrt{1 - r_4^2 - r_5^2})(\pm\sqrt{1 - r_7^2 - r_8^2}) = 0 \\ r_1 r_7 + r_2 r_8 + (\pm\sqrt{1 - r_1^2 - r_2^2})(\pm\sqrt{1 - r_7^2 - r_8^2}) = 0 \end{cases} \tag{3.38}$$

Introducendo elementi nel vettore b, l'equazione può essere espressa tramite

$$\begin{cases} Se_1 t_{z1}^4 - Sr_1 t_{z1}^2 + 1 = 0 \\ Se_2 t_{z2}^4 - Sr_2 t_{z2}^2 + 1 = 0 \\ Se_3 t_{z3}^4 - Sr_3 t_{z3}^2 + 1 = 0 \end{cases} \tag{3.39}$$

dove

- $Sr_1 = b_1^2 + b_2^2 + b_4^2 + b_5^2$
- $Se_1 = (b_1 b_5 - b_2 b_4)^2$
- $Sr_2 = b_4^2 + b_5^2 + b_7^2 + b_8^2$
- $Se_2 = (b_5 b_7 - b_4 b_8)^2$
- $Sr_3 = b_1^2 + b_2^2 + b_7^2 + b_8^2$
- $Se_3 = (b_1 b_8 - b_2 b_7)^2$

L'elemento t_{z1} può essere calcolato tramite l'equazione (3.40).

$$t_{z1}^2 = \frac{(Sr \pm \sqrt{Sr^2 - 4Se})}{2Se} \tag{3.40}$$

In base al problema riguardante t_y^2 nel metodo di Tsai, l'elemento t_{z1} può essere calcolato dalla (3.40)

$$t_{z1}^2 = \frac{(Sr - \sqrt{Sr^2 - 4Se})}{2Se} \tag{3.41}$$

Similmente anche t_{z2}^2 e t_{z3}^2 possono essere calcolate. Di norma il segno $\pm$ per t_{z1}^2,t_{z2}^2 e t_{z3}^2 dovrebbe essere determinato. In questo modello, tutti loro giacciono in direzione positiva della dimensione Z_C e sono positivi. La media di t_{z1}^2,t_{z2}^2 e t_{z3}^2 può essere calcolata.

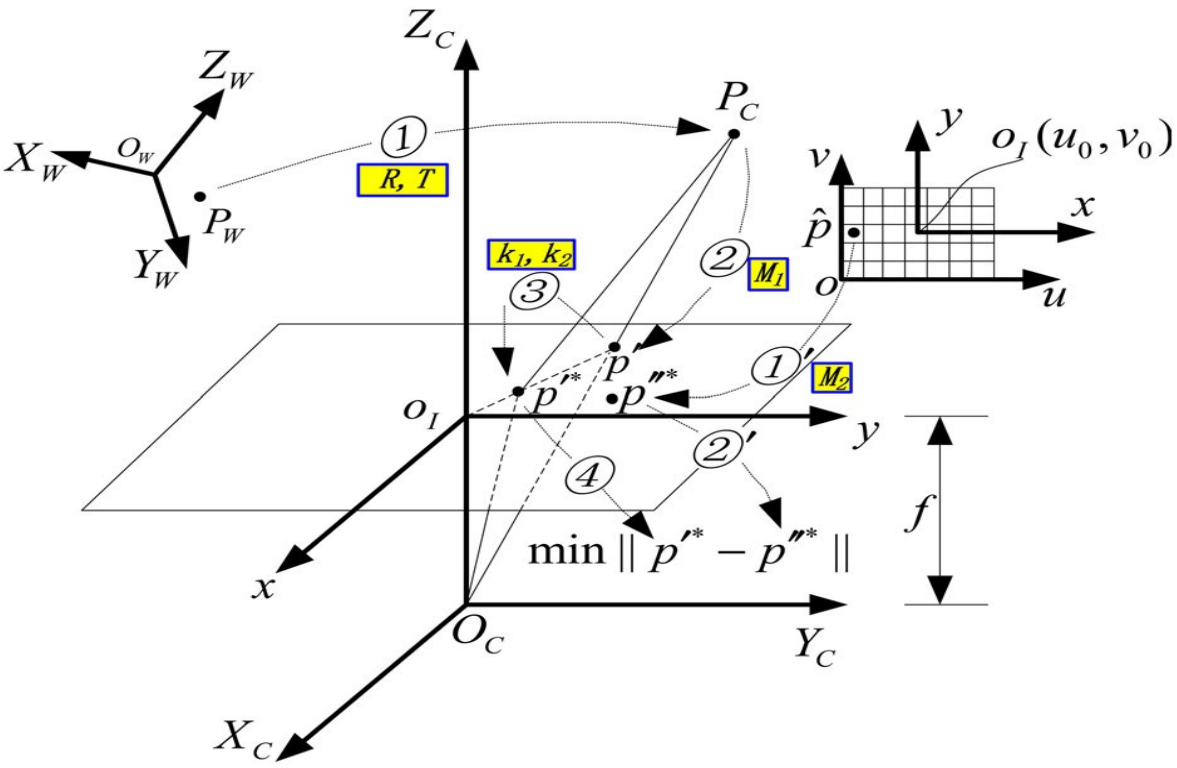

Figura 3.4: Passi per l'ottimizzazione non lineare

$$t_z = \frac{(t_{z1} + t_{z2} + t_{z3})}{3} \tag{3.42}$$

Pertanto, $r_1, r_2, r_4, r_5, r_7, r_8, t_x, t_y$ possono essere calcolati in accordo alle equazioni (3.38) e (3.39). Gli elementi nella terza colonna r_3, r_6, r_9 possono essere ottenuti grazie alle prime due colonne senza tener conto del segno.

$$\begin{bmatrix} r_3 & r_6 & r_9 \end{bmatrix}^T = \begin{bmatrix} r_1 & r_4 & r_7 \end{bmatrix}^T \times \begin{bmatrix} r_2 & r_5 & r_8 \end{bmatrix}^T \tag{3.43}$$

Algoritmo di ottimizzazione non lineare

Verrano ora introdotti i coefficienti di distorsione radiale (k_1 e k_2). Come mostrato nell'immagine 3.4 il punto P_W può essere trasformato nel punto distorto p'^* dalle coordinate mondo $O_W X_W Y_W Z_W$ alle coordinate del piano immagine $o_I xy$ attraverso gli step (1), (2) e (3). Il punto $p''^* = [x'^*, y'^*]^T$ è la proiezione distorta del punto P_W trasformato dalle coordinate pixel ouv alle coordinate dell'immagine $o_I xy$ attraverso il passo (1)'. Esiste un residuo tra p'^* e p''^*.
Il risultato è che la seguente funzione di ottimizzazione del residuo è progettata per

minimizzare il residuo utilizzando il metodo di ottimizzazione non lineare.

$$
\begin{aligned}
F_{2D} &= \min \| p^{'*}([R,T], M_1, k_1, k_2) - p^{''*}(M_2) \| \\
&= \min \sqrt{\frac{\sum_{i_1}^{m}\sum_{j=1}^{n}(x_{ij}^{*} - x_{ij}^{'*})^2 + \sum_{i=1}^{m}\sum_{j=1}^{n}(y_{ij}^{*} - y_{ij}^{'*})^2}{mn}}
\end{aligned}
\tag{3.44}
$$

dove M_1 e M_2 sono le trasformazioni dalle coordinate della camera e delle coordinate pixel in coordinate immagine. Come mostrato in figura 3.4 i passi (2) e (1)' sono proprio queste trasformazioni.

$$
M_1 = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad M_2 = \begin{bmatrix} \frac{1}{d_x} & 0 & u_0 \\ 0 & \frac{1}{d_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.45}
$$

Algoritmo di calibrazione stereo

La calibrazione stereo può essere eseguita sulle basi di una singola telecamera. In accordo con le relazioni trigonometriche tra due telecamere e con la scacchiera, possiamo calcolare la matrice di rotazione R_B e il vettore di traslazione T_B.
Usando una scacchiera 2D, teoricamente la distanza tra due angoli è d con un errore massimo $\pm\Delta\delta$. Come mostrato in figura 3.2, $O_W X_W Y_W Z_W$ sono le coordinate mondo, mentre $O_{Cl} X_{Cl} Y_{Cl} Z_{Cl}$ e $O_{Cr} X_{Cr} Y_{Cr} Z_{Cr}$ sono le coordinate delle due telecamere.
$o_l x_l y_l$ e $o_r x_r y_r$ sono invece le coordinate delle immagini. Le lunghezze focali sono f_l e f_r. $P_W = [X_W, Y_W, Z_W]^T$ è un punto d'angolo della scacchiera in coordinate mondo, ed è trasformato in $P_l = [X_l, Y_l, Z_l]^T$ e $P_r = [X_r, Y_r, Z_r]^T$ attraverso $[R_{Bl}, T_{Bl}]$ e $[R_{Br}, T_{Br}]$ rispettivamente, per la telecamera sinistra e destra. Le loro proiezioni in coordinate immagine sono i punti $p_l = [x_l, y_l]^T$ e $p_r = [x_r, y_r]^T$.
Quindi la trasformazione del punto P_W da coordinate mondo alle coordinate della telecamera sinistra e destra possono essere descritte da

$$
\begin{bmatrix} P_l \\ P_r \end{bmatrix} = \begin{bmatrix} R_{Bl} \\ R_{Br} \end{bmatrix} P_W + \begin{bmatrix} T_{Bl} \\ T_{Br} \end{bmatrix} \tag{3.46}
$$

La trasformazione di coordinate tra telecamera sinistra e destra può essere eseguita attraverso

$$P_r = R_B P_l + T_B \tag{3.47}$$

Qui:

$$R_B = R_{Br} R_{Bl}^{-1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad T_B = T_{Br} - R_B T_{Bl} = \begin{bmatrix} t_{Bx} \\ t_{By} \\ t_{Bz} \end{bmatrix} \tag{3.48}$$

vengono risolti i parametri R_B e T_B. Per migliorare la loro precisione, la coppia dei punti d'angolo (punto d'angolo sinistro e punto d'angolo destro) vengono ricostruiti in coordinate mondo tramite i principi trigonometrici. La differenza tra la distanza teorica e quella reale di due angoli adiacenti, è calcolata e considerata come errore. Riscriviamo l'equazione (3.47) come

$$\begin{bmatrix} X_r \\ Y_r \\ Z_r \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_l \\ Y_l \\ Z_l \\ 1 \end{bmatrix} \tag{3.49}$$

La relazione tra un punto 3D in coordinate mondo e coordinate della telecamera può essere descritta dall'equazione (3.50).

$$s_l \begin{bmatrix} x_l \\ y_l \\ 1 \end{bmatrix} = \begin{bmatrix} f_l & 0 & 0 & 0 \\ 0 & f_l & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_l \\ Y_l \\ Z_l \\ 1 \end{bmatrix} \quad s_r \begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} = \begin{bmatrix} f_r & 0 & 0 & 0 \\ 0 & f_r & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_r \\ Y_r \\ Z_r \\ 1 \end{bmatrix} \tag{3.50}$$

Cercando di semplificare i calcoli, assumiamo che le coordinate della telecamera sinistra siano le coordinate mondo. In questo modo il punto P_l coincide con il punto P_W.

$$\begin{cases} X_W = Z_W \dfrac{x_l}{f_1} \\ \\ Y_W = Z_W \dfrac{y_l}{f_1} \\ \\ Z_W = \dfrac{f_l(f_r t_x - x_r t_z)}{x_r(r_{31}x_l + r_{32}y_l + r_{33}f_l) - f_r(r_{11}x_l + r_{12}y_l + r_{13}f_l)} \\ \\ \qquad = \dfrac{f_l(f_r t_y - y_r t_z)}{y_r(r_{31}x_l + r_{32}y_l + r_{33}f_l) - f_r(r_{21}x_l + r_{22}y_l + r_{23}f_l)} \end{cases} \tag{3.51}$$

Supposto che il set $\{P_W^{i,j} | (X_W^{i,j}, Y_W^{i,j}, Z_W^{i,j}) i = 1, 2, \ldots, m, j = 1, 2, \ldots, n\}$ includa tutti i punti d'angolo della scacchiera, la reale distanza tra due punti d'angolo adiacenti nella colonna j è

$$\begin{aligned} \hat{d}^{i,j} &= |P_W^{i+1,j} - P_W^{i,j}| \\ &= \sqrt{(X_W^{i+1,j} - X_W i, j)^2 - (Y_W i + 1, j - Y_W^{i,j})^2 - (Z_W^{i+1,j} - Z_W^{i,j})^2} \end{aligned} \tag{3.52}$$

con $i = 1, 2, \ldots, M - 1$.
Esiste una differenza tra distanza reale e distanza teorica. Per minimizzare la differenza, usiamo la funzione

$$\begin{cases} \zeta = \min \frac{1}{n(m-1)} \displaystyle\sum_{j=1}^{n} \sum_{i=1}^{m-1} \|\hat{d}^{i,j} - d\|_2 \\ \\ \zeta \leq \Delta\delta \end{cases} \tag{3.53}$$

Il metodo della funzione di penalità è utilizzato per ottimizzare globalmente ζ. Da qui si possono ottenere migliori parametri, sia intrinseci che estrinseci.

Capitolo 4

Prove sperimentali

In questo capitolo esaminiamo una parte fondamentale del metodo di Caprile e Torre, riguardante il problema ai minimi quadrati per il calcolo dei punti di fuga. Per la scrittura e il calcolo degli algoritmi abbiamo utilizzato **MATLAB**.
MATLAB (abbreviazione di Matrix Laboratory) permette la risoluzione di numerosi problemi di calcolo tecnici, in particolare quelli caratterizzati da formulazioni di tipo vettoriale e matriciale, attraverso algoritmi molto più semplici e snelli rispetto a quelli che sarebbero necessari in un programma in linguaggio scalare non interattivo (linguaggio C o Fortran). È usato da milioni di persone nell'industria e nelle università per via dei suoi numerosi strumenti a supporto dei più disparati campi di studio applicati e funziona su diversi sistemi operativi, tra cui Windows, Mac OS, GNU/Linux e Unix.
Il problema ai minimi quadrati, per determinare i punti di fuga, è un'operazione che deve essere eseguita per ogni singola immagine perché ogni scacchiera (essendo posizionata diversamente in ogni immagine) ha i suoi punti di fuga. Per testare gli algoritmi prenderemo una singola immagine, sufficiente allo scopo da raggiungere.
Sarà poi possibile modificare gli algoritmi per elaborare tutte le immagini insieme ed effettuare la calibrazione (questa parte non verrà trattata, rimandando a futuri sviluppi).

4.1 Acquisizione immagini e corner detection

Il primo passo consiste nell'acquisire delle immagini su cui lavorare.
Nel nostro caso, non abbiamo bisogno di acquisire immagini attraverso una telecamera perché MATLAB, grazie al programma *Stereo Camera Calibrator* inserito nel toolbox *Computer Vision*, fornisce un set di dieci coppie di immagini (dieci per la telecamera sinistra e dieci per la telecamera destra) di una scacchiera.
Le coppie di immagini possono essere reperite attraverso una cartella interna, facilmente raggiungibile:
`C:\Program Files\MATLAB\R2016b\toolbox\vision\visiondata\calibration\stereo`.

In questa cartella avremo altre due cartelle, **left** e **right**, contenenti ciascuna le dieci immagini.
Come detto precedentemente, si lavorerà solo con un'immagine.
Il primo passo da fare, una volta caricata l'immagine su MATLAB, è determinare gli angoli dei quadrati della scacchiera. Per fare questo ci viene in soccorso la funzione **detectCheckerboardPoints(I)** che, data in ingresso un'immagine **I**, restituisce i punti della scacchiera (**imagePoints**) e il numero di righe e colonne della scaccchiera (**boardSize**).

```
I = imread('i1.png');
[imagePoints, boardSize] = detectCheckerboardPoints(I);
imshow(I);
hold on;
plot(imagePoints(:,1), imagePoints(:,2), 'm*');
```

Presa l'immagine e caricata

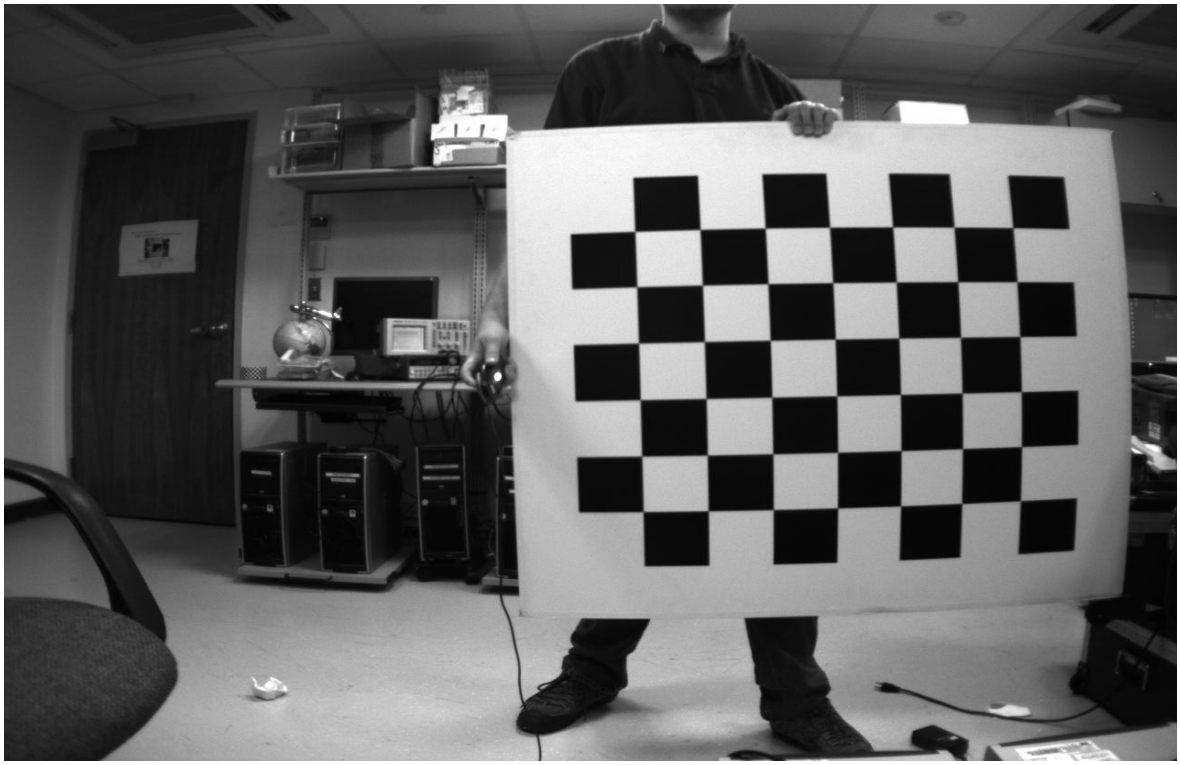

Figura 4.1: Immagine originale

il codice restituirà un'immagine con i punti della scacchiera evidenziati.

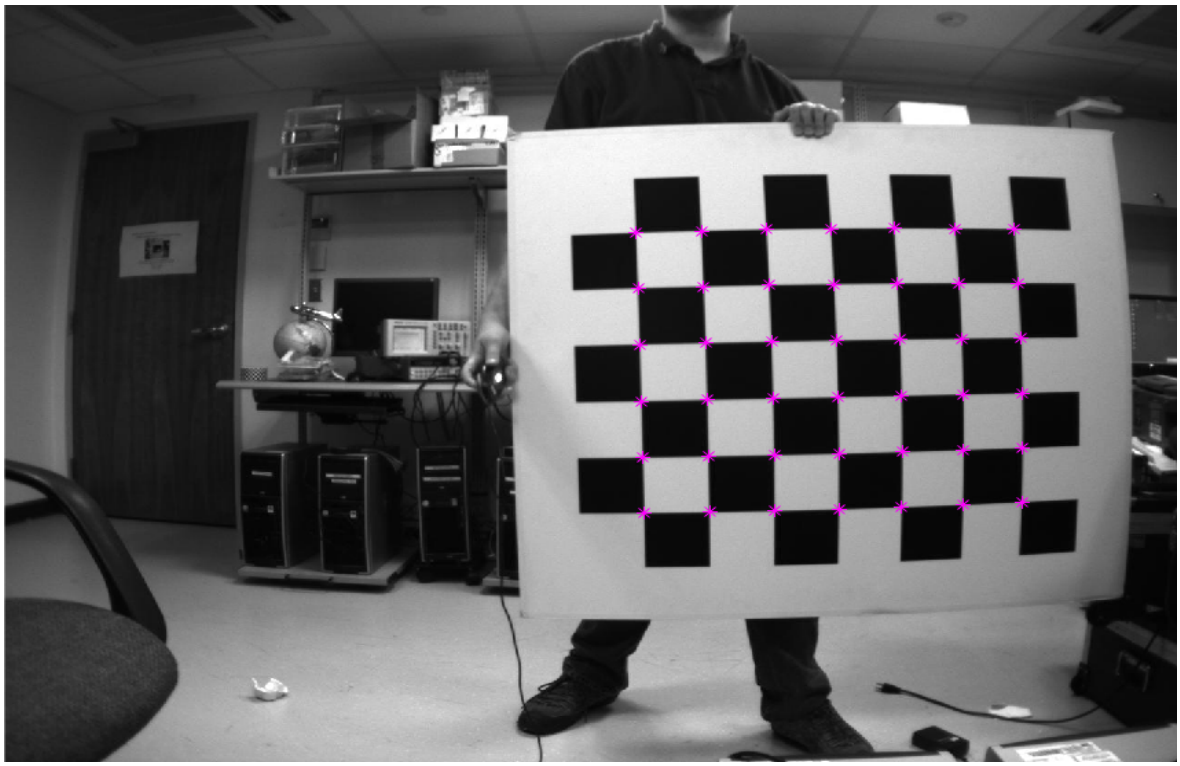

Figura 4.2: Immagine con evidenziati i punti della scacchiera

Determinati i punti della scacchiera, abbiamo risolto l'equazione delle rette, orizzontali e verticali, determinando i parametri a e b.
Ricordando il passo (3.31) del metodo abbiamo scritto il codice

$$\begin{aligned} E^i_{LS} &= \varepsilon(a^i_{or}, b^i_{or}) \\ &= \sum_{j=1}^{n} ([a^i_{or}, -1, b^i_{or}][u_{ij}, v_{ij}, 1]^T)^2 \\ &= \sum_{j=1}^{n} (a^i_{or} u_{ij} + b^i_{or} - v_{ij})^2 \end{aligned}$$

```
m = 6; n = 7;
U = reshape(imagePoints(:,1),m,n);
V = reshape(imagePoints(:,2),m,n);
for i = 1:m
    Uh = [U(i,:)' ones(n,1)];
    Ch(:,i) = Uh\V(i,:)';
end
for j = 1:n
    Uv = [U(:,j) ones(m,1)];
    Cv(:,j) = Uv\V(:,j);
end
```

dove:

- m è il numero di righe,
- n è il numero di colonne,
- U è una matrice 6×7 contenente le coordinate X in pixel,
- V è una matrice 6×7 contenente le coordinate Y in pixel,
- Uh è una matrice 7×2 che prende le righe una per volta,
- Uv è una matrice 6×2 che prende le colonne una per volta,
- Ch (da determinare) è una matrice formata dai parametri a, b delle sei rette orizzontali,
- Cv (da determinare) è una matrice formata dai parametri a, b delle sette rette verticali.

Una volta determinati i parametri a, b si può passare a determinare i punti di fuga.
Il passo (3.32), ci permette di ottenere i punti di fuga

$$\begin{aligned} F_{or} &= \underset{l^i_{or}}{\operatorname{argmin}} \sum_{i=1}^{m} \operatorname{dis}(V_{or}(u_{or}, v_{or}), l^i_{or}(a^i_{or}, -1, b^i_{or})) \\ &= \underset{l^i_{or}}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{|a^i_{or} u_{or} - v_{or} + b^i_{or}|}{\sqrt{{a^i_{or}}^2 + (-1)^2}} \end{aligned}$$

Scriviamo la funzione **fugadis**.

```
function F = fugadis(pf,C)
n = size(C,2);
num = abs([C(1,:)' -ones(n,1)]*pf+C(2,:)');
den = C(1,:)'.^2+1;
F = sum(num./den);
```

Fugadis prende in ingresso un punto di fuga **pf** e una matrice tra **Ch** o **Cv**.
A questo punto, abbiamo intrapreso due strade:

1. utilizzare la funzione **fminsearch**,
2. scrivere la funzione **Levmarq**, che riprende l'algoritmo di Levenberg-Marquardt.

Partiamo da **fminsearch**:
È una funzione contenuta in matlab e prevede l'utilizzo al suo interno del metodo di Nelder-Mead. Il metodo di Nelder-Mead, noto anche come metodo del simplesso (da non confondere con l'omonimo metodo di programmazione lineare) o metodo ameba,

è un metodo di ottimizzazione non lineare di funzioni definite su un dominio a n dimensioni. Il metodo prende il nome dai matematici John Nelder e Roger Mead, che l'hanno pubblicato nel 1965. Il metodo non fa uso delle derivate e si basa sul concetto di simplesso, un particolare tipo di politopo con n+1 vertici in uno spazio ad n dimensioni; esempi di simplesso sono un segmento in una retta, un triangolo nel piano, un tetraedro nello spazio etc. Il metodo approssima il punto di ottimo locale di un problema in n variabili quando la funzione obiettivo è liscia ed unimodale. Il metodo genera una nuova posizione di test estrapolando il comportamento della funzione obiettivo valutata in ogni punto del dominio ai vertici di un simplesso. L'algoritmo quindi sceglie uno di questi punti, da rimpiazzare con un nuovo punto. Il passo più semplice è rimpiazzare il punto più lontano dall'ottimo con il centroide dei restanti n punti: se la valutazione nel nuovo punto è migliore rispetto al punto corrente, si continua la ricerca con andamento esponenziale nella direzione individuata dal punto, altrimenti si cerca in direzione di un punto che fornisca una valutazione migliore.
Scrivendo

```
funF = @(pf) fugadis(pf,Ch);
[Vh1,fvalh1,flagh,outputh] = fminsearch(funF,X0);
funF = @(pf) fugadis(pf,Cv);
[Vv1,fvalv1,flagv,outputv] = fminsearch(funF,X0);
```

Vengono determinati i due punti di fuga (uno per le rette orizzontali e uno per le rette verticali).
Passiamo ora a scrivere la funzione **levmarq**
Riprendiamo l'equazione (1.18):

$$(J^T J + \mu I)h_{lm} = -g \quad \text{con} \quad g = J^T f \quad \text{e} \quad \mu \geq 0.$$

Ciò che ci serve determinare è la matrice jacobiana J.
Scriviamo la funzione **jack** per l'approssimazione della matrice Jacobiana mediante diffferenze finite:

```
%Calcolo Jacobiano con differenze finite
function J = jack(funz,x,dx)
x=x(:);
n = length(x);
if nargin < 3 || isempty(dx) , dx = 1e-2; end
y = funz(x);
m = length(y);
J = zeros(m,n);
for j = 1:n
```

```
        x_dx = x;
        x_dx(j) = x(j) + dx;
        y_dx = funz(x_dx);
        J(:,j) = (y_dx-y)./dx;
end
```

Presa una funzione 'funz', un punto 'x' e scelto un incremento 'dx', viene calcolata la matrice Jacobiana J.

Scriviamo ora la funzione **levmarq**

```
function [x,k,fval] = levmarq(funz,x0,mu,tau,nmax)
if nargin<5, nmax = 100000; end
if nargin<4, tau = 1e-3; end
if nargin<3, mu = 1e-16; end
k = 0;
x = x0;
flag = 1;
while flag
        k = k+1;
        xv = x;
        J = jack(funz,x);
        f = funz(x);
        % Gauss
        %h = (J'*J+mu*eye(2)) \ (-J'*f);
        % Cholesky
        R = chol(J'*J+mu*eye(2));
        y = R' \ (-J'*f);
        h = R \ y;
        x = x+h;
        flag = (norm(x-xv)>tau*norm(x)) && (k<=nmax);
end
fval=funz(x);
```

Levmarq prende in ingresso una funzione 'funz', un punto iniziale 'x0', un parametro 'mu', un parametro 'tau' per fermare l'algoritmo e un parametro che determina il numero massimo di iterazioni 'nmax'.
Come si può facilmente notare, abbiamo due modi per determinare il punto di fuga:

- utilizzare il metodo di Gauss, ricavando direttamente h,
- utilizzare la decomposizione di Cholesky per determinare la matrice R e ricavare successivamente h.

Utilizziamo la decomposizione di Cholesky, in quanto la fattorizzazione è già implementata e ottimizzata su MATLAB tramite la funzione **chol()**.
Scritto **levmarq**, non ci resta che confrontarlo con **fminsearch**.

```
X0 = [500;500];
funF = @(pf) fugadis(pf,Ch);
[Vh1,fvalh1,flagh,outputh] = fminsearch(funF,X0);
[Vh2,k1,fvalh2] = levmarq(funF,X0);
funF = @(pf) fugadis(pf,Cv);
[Vv1,fvalv1,flagv,outputv] = fminsearch(funF,X0);
[Vv2,k2,fvalv2] = levmarq(funF,X0);
```

Il test è stato realizzato facendo variare i parametri (mu, tau) della funzione **levmarq** e (dx) della funzione **jack**.
Per quanto riguarda **levmarq** abbiamo scelto:

- punto iniziale X0=$[500, 500]$,
- mu:$(1e-13,\ 1e-14,\ 1e-15,\ 1e-16)$,
- tau:$(1e-2,\ 1e-3,\ 1e-4,\ 1e-5)$.

Per la funzione **jack** abbiamo scelto quattro valori per l'incremento dx:
dx =$(1e-1,\ 1e-2,\ 1e-3,\ 1e-4)$
Abbiamo impostato il numero massimo di iterazioni $nmax = 100000$,
Per **fminsearch** i valori dei due punti di fuga cambiano solo in relazione al punto iniziale X0, che noi abbiamo scelto essere [500,500].
I risultati di **fminsearch** sono:
Vh1=$(1.67e+4, 109.12)$ [106 iterazioni, fvalh=7.45]
Vv1=$(-338.38, -4.39e+4)$ [134 iterazioni, fvalv=4.82]
dove fvalh e fvalv sono le **funzioni obiettivo** e stimano la precisione del punto determinato. Più è piccolo il valore di fval, più preciso è il valore del punto determinato.
Sia per **fminsearch** che per **levmarq**:
fval= $fun(x)$
Per **levmarq** $fun(x)$ è la nostra funF e x è il punto determinato.
Di seguito sono proposte le tabelle che riepilogano i risultati di **levmarq**:

Vh2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(1.66e+4, 109.12)$ 3 it. fvalh=30.88	$(1.66e+4, 115.78)$ 3 it. fvalh=30.81	$(1.66e+4, 115.86)$ 3 it. fvalh=30.89	$(1.70e+4, 100.77)$ 4 it. fvalh=18.20
	1e-3	$(1.66e+4, 115.85)$ 3 it fvalh=30.88	$(1.66e+4, 115.78)$ 3 it. fvalh=30.81	$(1.66e+4, 115.86)$ 3 it. fvalh=30.89	$(1.70e+4, 100.77)$ 4 it. fvalh=18.20
	1e-4	$(1.68, 109.43)$ 100000 it. fvalh=15.88	$(1.68e+4, 110.85)$ 100000 it. fvalh=27.60	$(1.69e+4, 110.54)$ 100000 it. fvalh=37.27	$(1.67e+4, 113.88)$ 100000 it. fvalh=29.04
	1e-5	$(1.68, 109.43)$ 100000 it. fvalh=15.88	$(1.68e+4, 110.85)$ 100000 it. fvalh=27.60	$(1.69e+4, 110.54)$ 100000 it. fvalh=37.27	$(1.67e+4, 113.88)$ 100000 it. fvalh=29.04

Vv2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(410.32, -2.63e+4)$ 9248 it. fvalv=52.54	$(410.36, -2.63e+4)$ 9234 it. fvalv=52.55	$(410.40, -2.63e+4)$ 8786 it. fvalv=52.55	$(410.39, -2.63e+4)$ 6058 it. fvalv=52.55
	1e-3	$(-353.33, -4.37e+4)$ 100000 it fvalv=5.40	$(-354.09, -4.37e+4)$ 100000 it. fvalv=5.40	$(-357.95, -4.39e+4)$ 100000 it. 5.40	$(-394.88, -4.52e+4)$ 100000 it. fvalv=5.37
	1e-4	$(-353.33, -4.37e+4)$ 100000 it. fvalv=5.40	$(-354.09, -4.37e+4)$ 100000 it. fvalv=5.40	$(-357.95, -4.39e+4)$ 100000 it. fvalv=5.40	$(-394.88, -4.52e+4)$ 100000 it. fvalv=5.37
	1e-5	$(-353.33, -4.37e+4)$ 100000 it. fvalv=5.40	$(-354.09, -4.37e+4)$ 100000 it. fvalv=5.40	$(-357.95, -4.39e+4)$ 100000 it. fvalv=5.40	$(-394.88, -4.52e+4)$ 100000 it. fvalv=5.37

Tabella 4.1: Valori di Vh2,Vv2 per dx=1e-1, nmax=100000.

Vh2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(1.66e+4, 115.85)$ 3 it. fvalh=30.88	$(1.66e+4, 115.88)$ 3 it. fvalh=30.91	$(1.70e+4, 100.95)$ 4 it. fvalh=17.90	$(1.65e+4, 116.82)$ 3 it. fvalh=31.78
	1e-3	$(1.66e+4, 115.85)$ 3 it. fvalh=30.88	$(1.66e+4, 115.88)$ 3 it. fvalh=30.91	$(1.70e+4, 100.95)$ 4 it. fvalh=17.90	$(1.65e+4, 116.82)$ 3 it. fvalh=31.78
	1e-4	$(1.68e+4, 109.51)$ 100000 it. fvalh=15.60	$(1.68e+4, 110.91)$ 100000 it. fvalh=27.96	$(1.68e+4, 110.81)$ 100000 it. fvalh=30.58	$(1.68e+4, 108.88)$ 100000 it. fvalh=13.65
	1e-5	$(1.68e+4, 109.51)$ 100000 it. fvalh=15.60	$(1.68e+4, 110.91)$ 100000 it. fvalh=27.96	$(1.68e+4, 110.81)$ 100000 it. fvalh=30.58	$(1.68e+4, 108.88)$ 100000 it. fvalh=13.65

Vv2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(410.34, -2.63e+4)$ 9290 it. fvalv=52.55	$(410.34, -2.63e+4)$ 9288 it. fvalv=52.55	$(410.41, -2.63e+4)$ 9292 it. fvalv=52.55	$(410.35, -2.63e+4)$ 9036 it. fvalv=52.55
	1e-3	$(-352.35, -4.37e+4)$ 100000 it. fvalv=5.41	$(-353.26, -4.37e+4)$ 100000 it. fvalv=5.40	$(-357.28, -4.39e+4)$ 100000 it. fvalv=5.40	$(-403.28, -4.55e+4)$ 100000 it. fvalv=5.36
	1e-4	$(-352.35, -4.37e+4)$ 100000 it. fvalv=5.41	$(-353.26, -4.37e+4)$ 100000 it. fvalv=5.40	$(-357.28, -4.39e+4)$ 100000 it. fvalv=5.40	$(-403.28, -4.55e+4)$ 100000 it fvalv=5.36
	1e-5	$(-352.35, -4.37e+4)$ 100000 it. fvalv=5.41	$(-353.26, -4.37e+4)$ 100000 it. fvalv=5.40	$(-357.28, -4.39e+4)$ 100000 it. fvalv=5.40	$(-403.28, -.55e+4)$ 100000 it. fvalv=5.36

Tabella 4.2: Valori di Vh2,Vv2 per dx=1e-2, nmax=100000.

Vh2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(1.66e+4, 115.86)$ 3 it. fvalh=30.88	$(1.66e+4, 115.87)$ 3 it. fvalh=30.90	$(1.66e+4, 114.74)$ 3 it. fvalh=29.84	$(1.70e+4, 100.67)$ 4 it. fvalh=18.36
	1e-3	$(1.66e+4, 115.86)$ 3 it. fvalh=30.88	$(1.66e+4, 115.87)$ 3 it. fvalh=30.90	$(1.66e+4, 114.74)$ 3 it. fvalh=29.84	$(1.70e+4, 100.67)$ 4 it. fvalh=18.36
	1e-4	$(1.67e+4, 109.55)$ 100000 it. fvalh=15.73	$(1.68e+4, 109.02)$ 100000 it. fvalh=16.64	$(1.68e+4, 110.90)$ 100000 it. fvalh=28.24	$(1.70e+4, 110.00)$ 100000 it. fvalh=51.01
	1e-5	$(1.67e+4, 109.55)$ 100000 it. fvalh=15.73	$(1.68e+4, 109.02)$ 100000 it. fvalh=16.64	$(1.68e+4, 110.90)$ 100000 it. fvalh=28.24	$(1.70e+4, 110.00)$ 100000 it. fvalh=51.01

Vv2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(410.32, -2.63e+4)$ 9290 it. fvalv=52.54	$(410.41, -2.63e+4)$ 9282 it. fvalv=52.55	$(410.41, -2.63e+4)$ 9224 it. fvalv=52.55	$(410.41, -2.63e+4)$ 9030 it. fvalv=52.55
	1e-3	$(-352.21, -4.37e+4)$ 100000 it. fvalv=5.41	$(-352.70, -4.37e+4)$ 100000 it. fvalv=5.41	$(-355.06, -4.38e+4)$ 100000 it. fvalv=5.40	$(-405.32, -4.55e+4)$ 100000 it. fvalv=5.36
	1e-4	$(-352.21, -4.37e+4)$ 100000 it. fvalv=5.41	$(-352.70, -4.37e+4)$ 100000 it. fvalv=5.41	$(-355.06, -4.38e+4)$ 100000 it. fvalv=5.40	$(-405.32, -4.55e+4)$ 100000 it. fvalv=5.36
	1e-5	$(-352.21, -4.37e+4)$ 100000 it. fvalv=5.41	$(-352.70, -4.37e+4)$ 100000 it. fvalv=5.41	$(355.06, -4.38e+4)$ 100000 it. fvalv=5.40	$(-405.32, -4.55e+4)$ 100000 it. fvalv=5.36

Tabella 4.3: Valori di Vh2,Vv2 per dx=1e-3, nmax=100000.

Vh2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(1.66e+4, 115.86)$ 3 it. fvalh=30.89	$(1.66e+4, 115.93)$ 3 it. fvalh=30.96	$(1.70e+4, 100.93)$ 4 it. fvalh=17.93	$(1.68e+4, 110.61)$ 3 it. fvalh=35.50
	1e-3	$(1.66e+4, 115.86)$ 3 it. fvalh=30.89	$(1.66e+4, 115.93)$ 3 it. fvalh=30.96	$(1.70e+4, 100.93)$ 4 it. fvalh=17.93	$(1.68e+4, 110.61)$ 3 it. fvalh=35.50
	1e-4	$(1.67e+4, 109.15)$ 100000 it. fvalh=13.35	$(1.68e+4, 108.58)$ 100000 it. fvalh=13.99	$(1.70e+4, 110.08)$ 100000 it. fvalh=49.10	$(1.67e+4, 111.86)$ 100000 it. fvalh=27.15
	1e-5	$(1.67e+4, 109.15)$ 100000 it. fvalh=13.35	$(1.68e+4, 108.58)$ 100000 it. fvalh=13.99	$(1.70e+4, 110.08)$ 100000 it. fvalh=49.10	$(1.67e+4, 111.86)$ 100000 it. fvalh=27.15

Vv2 (X,Y)		**mu**			
		$1e-13$	$1e-14$	$1e-15$	$1e-16$
tau	1e-2	$(410.42, -2.63e+4)$ 9288 it. fvalv=52.55	$(410.35, -2.63e+4)$ 9278 it. fvalv=52.55	$(410.32, -2.63e+4)$ 9190 it. fvalv=52.54	$(410.38, -2.63e+4)$ 9074 it. fvalv=52.55
	1e-3	$(-352.34, -4.37e+4)$ 100000 it. fvalv=5.41	$(-353.19, -4.37e+4)$ 100000 it. fvalv=5.40	$(-361.87, -4.40e+4)$ 100000 it. fvalv=5.40	$(-384.57, -4.48e+4)$ 100000 it. fvalv=5.38
	1e-4	$(-352.34, -4.37e+4)$ 100000 it. fvalv=5.41	$(-353.19, -4.37e+4)$ 100000 it. fvalv=5.40	$(-361.87, -4.40e+4)$ 100000 it. fvalv=5.40	$(-384.57, -4.48e+4)$ 100000 it. fvalh=5.38
	1e-5	$(-352.34, -4.37e+4)$ 100000 it. fvalv=5.41	$(-353.19, -4.37e+4)$ 100000 it. fvalv=5.40	$(-361.87, -4.40e+4)$ 100000 it. fvalv=5.40	$(-384.57, -4.48e+4)$ 100000 it. fvalv=5.38

Tabella 4.4: Valori di Vh2,Vv2 per dx=1e-4, nmax=100000.

Analizzando i dati e confrontandoli si ha un quadro completo dell'algoritmo scritto. L'algoritmo **levmarq** risulta abbastanza preciso ma necessita di molte iterazioni e quindi risulta lento. Questo è dovuto al fatto che l'algoritmo ha un 'mu' "statico" e non "dinamico" (come previsto da Levenberg - Marquardt). Apportando questa modifica l'algoritmo risulterebbe più veloce e preciso, ma sarebbe più complesso da scrivere. Qui, abbiamo scritto una versione iniziale di **levmarq** e successivamente esso potrà essere ottimizzato e migliorato. Infatti, pur essendo un buon algoritmo, esso non raggiunge (e non scende mai oltre) il valore della funzione obbiettivo di **fminsearch** che, ricordiamo, vale 7.45 per il punto di fuga orizzontale (contro quello di **levamrq** il cui miglior valore trovato vale 13.35) e 4.82 per il punto di fuga verticale (contro quello di **levmarq** il cui miglior valore trovato vale 5.36).
Abbiamo elaborato un test per verificare l'accuratezza dell'approssimazione dello Jacobiano mediante differenze finite. In esso la matrice Jacobiana viene valutata in un punto fissato, al decrescere dell'incremento 'dx':

```
%-----provajack-----
pf = [500;500];
dx = 1e-11;

funF = @(pf) fugadis(pf,Ch);
J = jack(funF,pf,dx)
```

La matrice J (nella rappresentazione 'long' di MATLAB) risultante è:

- dx=1e-1: -0.040560304388464 0.001816178366880
- dx=1e-2: -0.040560304387327 0.001816178371428
- dx=1e-3: -0.040560304341852 0.001816178382796
- dx=1e-4: -0.040560304341852 0.001816179064917
- dx=1e-5: -0.040560303204984 0.001816181338654
- dx=1e-6: -0.040560166780779 0.001816260919441
- dx=1e-7: -0.040558916225564 0.001817852535169
- dx=1e-8: -0.040552095015300 0.001818989403546
- dx=1e-9: -0.040472514228895 0.001818989403546
- dx=1e-10: -0.040927261579782 0.002273736754432
- dx=1e-11: -0.022737367544323 0.011368683772162
- dx=1e-12: 0.113686837721616 0

- dx=1e-13: 0 0
- dx=1e-14: 0 0
- dx=1e-15: 0 0
- dx=1e-16: 0 0
- dx=1e-17: 0 0

Per esempio:
dx= $1e-12$ produce una matrice J inutilizzabile per Cholesky.
(mu=$1e-16$, tau= $1e-5$)

```
J =

    0.113686837721616                    0
```

```
Error using chol
Matrix must be positive definite.

Error in levmarq (line 18)
        R = chol(J'*J+mu*eye(2));
```

Per valori di 'dx' compresi tra $1e-1 \div 1e-8$ la matrice J risulta simile. Con dx= $1e-9$ si inizia a notare una piccola deviazione, che aumenta sempre più con i successivi valori, fino a risultare completamente inutilizzabile. Da dx=$1e-12$ in poi, il risultato non è accettabile e **levmarq** si ferma subito.
Il motivo per cui i valori in J sono nulli è la **cancellazione**, che ha luogo gradatamente man mano che diventano più piccole le differenze a numeratore e denominatore dell'approssimazione delle derivate parziali. La **cancellazione** è stata esposta nel primo capitolo (Nozioni di algebra lineare e analisi numerica, pag. 15).
Le prove sono state eseguite su:
Portatile: Lenovo Ideapad Y700 - 15ISK 80NW
Processore: Intel Core i7 6700HQ 2.6 GHz
Scheda grafica dedicata: NVIDIA GeForce GTX 960M 4GB
Memoria ram: 16GB

Capitolo 5

Conclusioni

In questa tesi sono stati esposti due metodi per la *camera calibration*, l'algoritmo di **Zhang** e quello di **Caprile e Torre**. È stata implementata in MATLAB una parte del metodo di **Caprile e Torre**. Nello specifico, presa un'immagine contenente una scacchiera, è stata utilizzata una funzione (detectCheckerboardPoints) per determinare i punti d'angolo della scacchiera. In seguito son stati determinati i parametri a, b di ogni retta passante per i punti trovati (orizzontale e verticale) e infine son stati determinati i due punti di fuga, uno per le rette orizzontali e uno per le rette verticali.
È stata scritta la funzione **levmarq** che implementa il metodo di Levenberg-Marquardt e i risultati sono stati confrontati con la funzione **fminsearch**, contenuta nella libreria base di MATLAB.
In conclusione possiamo affermare che l'algoritmo **levmarq** è un buona base di partenza per scrivere un codice completo che effettui la *camera calibration*, ma necessita di essere migliorato e ottimizzato, specialmente per quanto riguarda la variazione del parametro *mu* che nella nostra implementazione è tenuto fissato, cioè durante l'esecuzione dell'algoritmo esso non varia. Questo porterebbe a risultati migliori con meno iterazioni e una precisione del punto di fuga indubbiamente più alta, riducendo la propagazione degli errori dovuta ad una elevata complessità computazionale.

Bibliografia

[1] Z. Zhang (2000), *A Flexible New Technique for Camera Calibration*, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[2] G. Rodriguez (2010), *Introduzione alla Matematica Applicata e Computazionale*, Pitagora Editrice Bologna.

[3] G. Rodriguez (2008), *Algoritmi Numerici*, Pitagora Editrice Bologna.

[4] B. Caprile, V. Torre (1990), *Using Vanishing Points for Camera Calibration*, International Journal of Computer Vision.

[5] H. Sun, J. Lu, Z. Chang (2016) *An efficient camera calibration and optimisation method based on othogonal vanishing points*, The Imaging Science Journal.

[6] E. Trucco, A. Verri (1998), *Introductory Techniques for 3-D Computer Vision*, Prentice Hall.

[7] Reinhard Klette (2014), *Concise Computer Vision, An Introduction into Theory and Algorithms*, Springer.

[8] K. Madsen, H.B. Nielsen, O. Tingleff (2004), *Methods for Non-Linear Least Square Problems*, Informatics and Mathematical Modelling, Technical University of Denmark.

[9] Lagarias, J. C., J. A. Reeds, M. H. Wright, and P. E. Wright. (1998), *Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions.*, SIAM Journal of Optimization. Vol. 9, Number 1.