UNIVERSITÀ DEGLI STUDI DI ROMA
"TOR VERGATA"

FACOLTÀ DI INGEGNERIA

Corso di Laurea in Ingegneria dell'Automazione

Tesi di Laurea di Primo Livello

Progetto, realizzazione e calibrazione di un sistema di visione stereoscopica

Relatore
Ing. F. Martinelli

Laureando
Paolo Pazziani

Anno Accademico 2010/2011

Ringraziamenti

Una tesi è sempre frutto di un intenso lavoro, durante il quale si possono presentare imprevisti e momenti di scoraggiamento che ho potuto superare soltanto grazie al sostegno delle persone che mi sono state accanto.

Per tale motivo desidero ringraziare i miei genitori che hanno creduto in me anche nei momenti più difficili, senza i cui sacrifici non avrei potuto portare a termine il mio progetto.

Un particolare ringraziamento va a Gabriella con la quale ho avuto la fortuna di dividere momenti importanti della mia vita. Le sarò sempre grato perché mi è sempre stata vicino ogni qualvolta si sia presentata una difficoltà, dandomi la forza di completare questo lavoro.

Vorrei esprimere tutta la mia gratitudine a Fabio il cui supporto è stato indispensabile.

Desidero ringraziare il prof. Martinelli, che mi ha offerto l'opportunità di conoscere il mondo della visione artificiale.

Ringrazio inoltre tutti coloro che mi hanno sostenuto durante il percorso dei miei studi.

Indice

Introduzione

Negli ultimi decenni si è assistito ad un rapido sviluppo della visione computazionale che ha reso verosimile il tentativo di imitare la capacità delle specie animali più evolute di ricostruire l'ambiente tridimensionale.
Nuovi metodi ed algoritmi nel riconoscimento di oggetti e nella ricostruzione di scene e modelli 3D stanno aprendo la strada a nuove applicazioni.
In campo industriale si sta affermando l'utilizzo di sistemi di visione computazionale [Scaramuzza 2003] [Trucco-98] per:

- controllo di qualità, allo scopo di identificare difetti e garantire il rispetto delle tolleranze;
- navigazione e coordinazione di robot mediante asservimento visuale;
- riconoscimento di oggetti per classificazione.

Impieghi altrettanto importanti si trovano nella video-sorveglianza, grazie alla possibilità di riconoscere volti, in ambiente militare e spaziale, nella robotica chirurgica nonché nello sviluppo di intelligenze artificiali e realtà virtuali.
Questa ampia varietà di applicazioni che sta emergendo conta sul fatto che la cattura dell'informazione dall'ambiente è sempre più veloce, efficiente ed a basso costo.
La crescente capacità di elaborazione dei processori e la diminuzione del costo delle webcam rende accessibile sia ai ricercatori che agli studenti la

strumentazione che in passato era prerogativa soltanto di coloro che erano specializzati nel settore. Lo studente che non è ancora esperto in queste tecniche non ha difficoltà nell'allestire la strumentazione per lo sviluppo di un sistema di visione stereoscopico perciò i suoi sforzi saranno rivolti alla comprensione degli aspetti matematici e all'implementazione degli algoritmi alla base della computer vision.

Per rendere ancor più alla portata di tutti la visione artificiale, si è spostata l'attenzione verso la facilità di utilizzo del software e la sua immediata comprensione. In tale direzione vanno lo sviluppo delle librerie OpenCV e di altri software come l'Image Processing Toolbox di Matlab che facilitano l'approccio ad una materia che può risultare ostica per la complessità degli algoritmi e per le profonde basi geometriche.

Questi software, trattando alcuni aspetti come black box, celano all'utente gli aspetti più difficili e spostano l'impegno verso lo sviluppo di nuove applicazioni.

In questa tesi viene illustrata la progettazione di un sistema di visione stereoscopico e la realizzazione di un software user friendly per il processamento delle scene, la calibrazione di una coppia di telecamere e l'analisi stereo. Come esempio applicativo viene proposto il riconoscimento di un oggetto sferico nello spazio e il calcolo delle sue coordinate spaziali, nonché della dimensione del suo raggio attraverso la rettificazione delle immagini e la triangolazione.

Capitolo 1
Nozioni Preliminari

1.1 Visione computazionale

La visione computazionale è la scienza che si occupa dell'analisi delle immagini attraverso un calcolatore allo scopo di determinare le proprietà geometriche, fisiche e dinamiche del mondo che ci circonda.

Lo scopo ultimo della Computer vision è quello di creare un modello del mondo reale, riconoscere e classificare gli oggetti che ne fanno parte ed estrarre le relazioni spaziali e logiche esistenti tra di essi. La comprensione dello spazio tridimensionale può essere considerata un problema di alto livello a cui si può giungere soltanto una volta affrontati problemi di più basso livello ovvero l'estrazione delle proprietà geometriche e fisiche dell'ambiente visibile, come i contorni degli oggetti, la loro forma e posizione.

L'inizio delle ricerche sull'intelligenza artificiale si fa spesso risalire al 1950 con la formulazione del test di Turing. In questo senso la visione artificiale può essere considerata come una branca dell'intelligenza artificiale in quanto si potrebbe formulare un test che giudichi "intelligente" una macchina in grado di rendersi consapevole del mondo che la circonda in modo indistinguibile da un essere umano.

I cosiddetti *chatterbot, s*eppur non a lungo, riescono talvolta ad ingannare i loro interlocutori, dimostrando una capacità di utilizzo del linguaggio

naturale molto vicina a quella umana. La vista è il senso che ci permette di conoscere meglio il mondo [Aristotele, incipit della Metafisica]; le capacità dei calcolatori sono tuttavia molto distanti da quelle umane per quanto riguarda la comprensione del mondo esterno. Un computer che analizzi un'immagine in modo da essere consapevole dello spazio da essa rappresentato e delle interazioni oggetto-oggetto e oggetto-scena ha bisogno di una quantità di elaborazione assai maggiore di quella necessaria per l'interrogazione linguistica di un database [Koch-Tononi-2008].

La distanza tra le capacità umane e quelle delle macchine non sta nel numero di informazioni elaborate, ma nella capacità di integrarle. Un sistema di visione artificiale moderno è oggi in grado di distinguere una molteplicità di oggetti in una scena con un discreto grado di complessità, ma non è in grado di metterli in relazione tra di loro e di confrontarli con altre immagini viste in precedenza. Il cervello umano mette in relazione le informazioni a molteplici livelli; non solo distingue colore, posizione e forma di un singolo oggetto, ma unisce tutte le informazioni in un'unica scena dalla quale estrae le informazioni che ritiene più importanti in quel momento.

1.2 La percezione visiva

La visione è l'insieme dei processi di registrazione, elaborazione, trasmissione ed interpretazione degli stimoli luminosi, che portano alla percezione delle immagini ed all'analisi delle loro caratteristiche. Tali stimoli luminosi derivano dalla percezione della radiazione

elettromagnetica appartenente al visibile di lunghezza d'onda compresa tra 350 nm e 780 nm emessi da una sorgente luminosa oppure derivante da luce riflessa. [UTET-91].

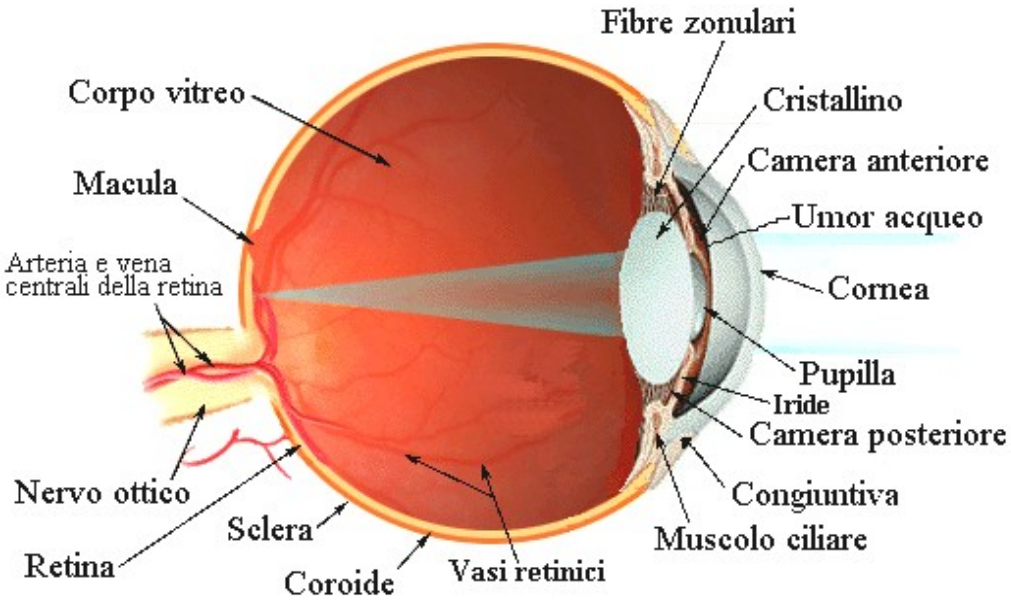


Illustrazione 1: Sezione trasversale di un occhio umano

Negli animali il principale organo deputato alla visione è l'occhio. In particolare la luce entra nell'occhio umano attraverso la cornea, una membrana trasparente che ha la funzione di imprimere alle radiazioni luminose che raggiungono l'occhio la convergenza necessaria a essere deviate sul cristallino, il quale costituisce la lente biologica che fa convergere le immagini su un'altra sottile membrana trasparente che riveste la superficie interna del globo oculare, la retina.

La retina è composta da coni e bastoncelli, due tipi di fotorecettori che trasformano gli stimoli luminosi in segnali nervosi e li inviano attraverso il nervo ottico. I coni sono presenti maggiormente nella zona centrale della

retina, detta macula, risultano sensibili solo a luci piuttosto intense e sono responsabili della visione dei colori. I bastoncelli sono più numerosi nelle zone periferiche, aiutano la visione notturna essendo particolarmente sensibili a basse intensità di luce e allargano il campo visivo. Pertanto, i coni operano soprattutto in condizione di luce piena, mentre i bastoncelli permettono la visione anche quando la luce è scarsa. La luce percepita dal sistema visivo umano provoca tre sensazioni principali: luminosità, tonalità e saturazione.

La luminosità (brightness) è di solito fisicamente correlata alla intensità di radiazione luminosa, mentre la tonalità (hue) con la lunghezza d'onda. Questa corrispondenza tuttavia non è esatta in quanto oggetti con la medesima intensità possono apparire all'occhio umano di differente luminosità. Inoltre con tonalità noi intendiamo la distinzione tra i vari colori derivante dalla interazione delle tre differenti tipologie di coni, responsabili rispettivamente della visione dei tre colori primari (rosso, verde e blu).

In realtà non esiste una corrispondenza biunivoca tra colore e lunghezza d'onda, infatti raggi luminosi di differente lunghezza d'onda possono risultare alla vista con la stessa tonalità e, allo stesso tempo, è necessario combinare fasci di luce di differente lunghezza d'onda per produrre alcune tonalità.

Alla saturazione solitamente non è associata alcuna grandezza fisica quantitativa in quanto essa indica il candore della sorgente luminosa.

1.3 Formazione dell'immagine

Si può facilmente constatare come la progettazione delle telecamere si sia ispirata alla fisiologia dell'occhio umano, infatti esso è composto da un sistema di lenti, da un diaframma (iride) e un'area fotosensibile.

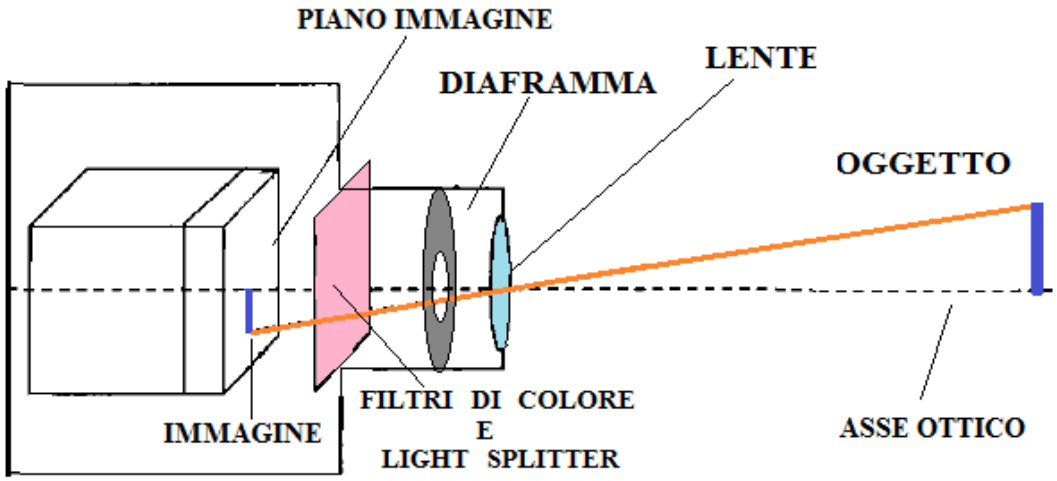


Illustrazione 2: Struttura di una telecamera

In uscita dall'occhio partono attraverso le fibre del nervo ottico una serie di segnali bioelettrici che vengono trasmessi al calcolatore neurale rappresentato dal nostro cervello. Inoltre, l'occhio può percepire i tre colori primari (rosso, verde e blu).

Una telecamera è un dispositivo che memorizza l'intensità di radiazione luminosa in funzione della posizione grazie all'utilizzo di un sistema di

acquisizione dell'immagine (imaging). Questo complesso sistema può schematicamente considerarsi costituito da una lente, un diaframma, un divisore di luce, un sensore fotosensibile e dei dispositivi elettronici. La lente fa convergere la radiazione nel *centro ottico* della lente detto *fuoco* o centro di proiezione, che verrà in seguito indicato con Oc.

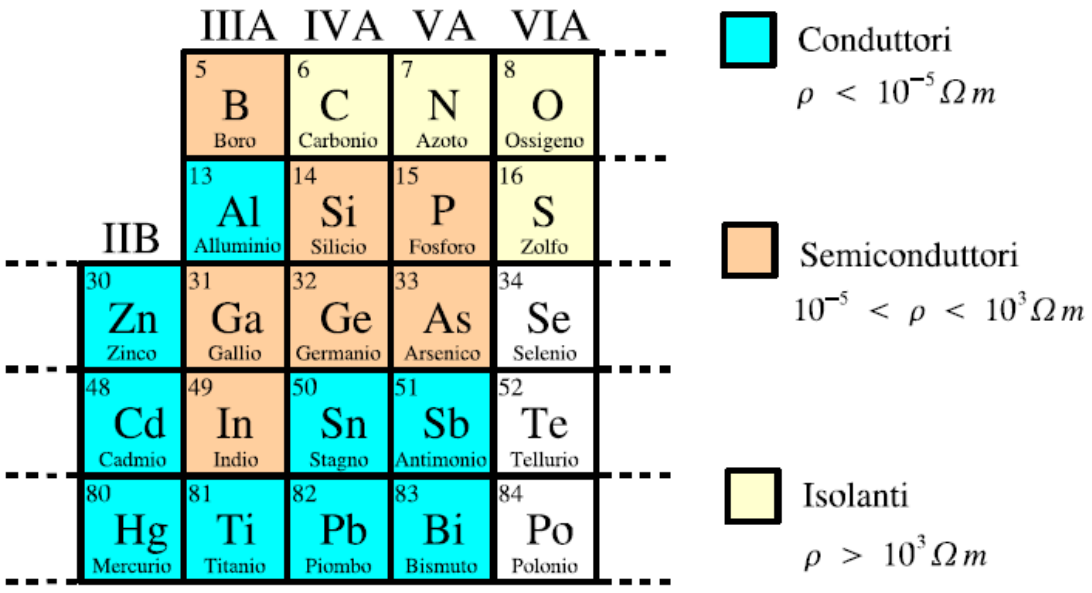
1.4 Foto sensori

Il cuore di ogni telecamera digitale è il sensore, cioè il dispositivo che converte le luce in cariche elettriche, per trasformarle poi in informazioni digitali. La superficie, che possiamo approssimativamente considerare piana, del dispositivo foto-rilevatore viene detta piano immagine. La maggior parte delle telecamere oggi in commercio si basano su sensori allo stato solido, realizzati con chip di silicio, mentre i dispositivi basati sulla tecnologia a tubo catodico sono divenuti obsoleti dagli anni '90 e hanno oramai un interesse soltanto storico [Di Stefano]. Tra i sensori allo stato solido le tecnologie più diffuse sono la tecnologia CCD (Charge-Coupled Devices) e CMOS (Complementary Metal-Oxide-Semiconductor). Entrambi i sensori sfruttano l'effetto fotoelettrico, vale a dire il fenomeno fisico per cui quando dei raggi di luce colpiscono gli atomi sulla superficie di un materiale, questi salgono ad un livello energetico superiore. Per alcuni materiali gli elettroni orbitanti possono anche oltrepassare l'orbita stabile più grande divenendo liberi di muoversi all'interno del materiale. Questo fenomeno permette ad ogni elemento fotosensibile di convertire l'energia luminosa in energia elettrica che a sua volta viene codificata in informazione digitale anche attraverso un

processore numerico.

1.4.1 Semiconduttori

Alla base dei circuiti elettronici, e quindi anche dei sensori CMOS e CCD, stanno le proprietà dei semiconduttori, in particolare quella di comportarsi a basse temperature come isolanti, ma se forniti di energia come quella termica o luminosa si comportano in maniera prossima ad un metallo, con valori di conducibilità elettrica non nulli.



Nei semiconduttori intrinseci, come germanio e silicio puri, che hanno una struttura regolare direzionale a reticolo e legami covalenti i cui livelli sono completi, basta il contributo di energia termica dovuto alla temperatura ambiente per spezzare il legame covalente e far saltare un elettrone verso un livello energetico libero. Ciò corrisponde al fatto che quell'elettrone è

passato dalla banda con energia più bassa (banda di valenza) alla banda libera, superando la banda di conduzione, cioè quella più alta in energia fra quelle non occupate da elettroni. Nella banda di valenza viene così a mancare un elettrone, la lacuna che si genera corrisponde in quella zona ad una area sostanzialmente a polarità positiva, che facilmente attrarrà un altro elettrone. Lacune ed elettroni si generano continuamente a coppie in seguito alla rottura di legami covalenti per effetto della agitazione termica e scompaiono a coppie, quando una lacuna ed un elettrone si ricombinano a riformare un legame covalente. Se un reticolo di materiale semiconduttore tetravalente viene drogato con un altro materiale semiconduttore pentavalente (Fosforo, Arsenico) o trivalente (Boro, Alluminio, Gallio, Indio) si ottiene un semiconduttore estrinseco e può avvenire nel primo caso che venga rilasciato un elettrone dall'atomo del materiale dopante pentavalente avendo il quinto elettrone spaiato che deve compiere un basso salto energetico per passare al di sopra del livello di conduzione del materiale principale, mentre nel secondo caso in corrispondenza della lacuna introdotta dal trivalente è basso il salto energetico che un elettrone del materiale principale deve fare per raggiungere il livello energetico vuoto corrispondente al suo livello di conduzione. Nel primo caso si hanno semiconduttori di tipo n il cui livello di partenza dell'elettrone si dice livello donatore. Nel secondo caso si hanno semiconduttori di tipo p in cui il livello della lacuna si dice livello accettore.

In presenza di un campo elettrico si stabilisce una corrente di deriva (drift

current), analogamente a quanto avviene nei metalli, a cui si aggiunge un secondo meccanismo legato alla diffusione, il quale invece è irrilevante nei metalli, per cui anche in assenza di un campo elettrico il gradiente di concentrazione dei portatori di carica dà origine a densità di corrente.
Modificando il drogaggio si può modificare la conduttività dei semiconduttori e questo è il principale motivo per cui sono così utili.

1.4.2 Il Dispositivo MOS

Il nucleo dei sensori CCD e CMOS è costituito dal condensatore MOS (Metal-Oxid-Semiconductor) formato da un substrato di semiconduttore drogato e da un'armatura, consistente in uno strato metallico, detta gate, separate tra loro da un ossido isolante.

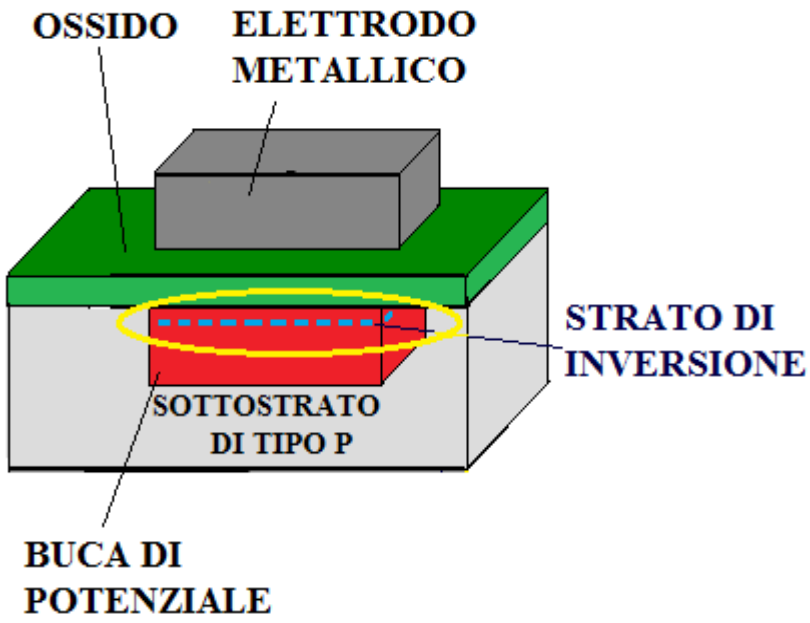


Illustrazione 3: Struttura di un condensatore MOS

Il substrato di silicio presenta una distribuzione uniforme di lacune che, se viene applicato un potenziale esterno ai capi dell'elettrodo metallico e del substrato, tendono ad allontanarsi dall'elettrodo. Si crea così una zona di svuotamento associata ad una buca di potenziale che va sempre più in profondità con l'aumentare del valore della differenza di potenziale.
Il processo di svuotamento termina quando la buca di potenziale è tanto profonda da attirare nella zona in cui il potenziale è minimo, vale a dire in corrispondenza della giunzione tra semiconduttore e ossido, uno strato di elettroni detto strato di inversione.

1.4.3 La Giunzione P-N

Un altro aspetto in comune è l'utilizzo della giunzione P-N, formata dal contatto di una regione di tipo P con una di tipo N in modo che si passi rapidamente dalla regione con atomi accettori, e quindi ricca di portatori di carica positivi (lacune), alla regione con donatori e ricca di portatori di carica negativi (elettroni). Nella giunzione si vengono quindi ad avere fortissimi gradienti di concentrazione che attivano un flusso elettronico dalla zona N alla zona P che, raggiunto il punto di equilibrio elettrostatico, determina un eccesso di carica positiva nella zona n creando inoltre una regione intermedia detta zona di svuotamento o di carica spaziale. Il risultato è un campo elettrico interno al dispositivo.

1.4.4 Charged Coupled Devices

Un rilevatore di fotoni CCD è costituito da una matrice di condensatori

MOS foto-sensibili che catturano e immagazzinano l'informazione nella forma di carica elettrica localizzata la quale varia con l'intensità luminosa incidente. La matrice di sottili strati di silicio (wafer) è ripartita da una griglia ortogonale di strette strisce di elettrodi sotto carico, o Gate, depositata sul chip la quale definisce i singoli elementi dell'immagine detti pixel [K. R. Spring].

Per non adibire dell'elettronica di misura per ogni condensatore, la quantità di carica non viene misurata sul singolo MOS, ma viene trasferita attraverso più condensatori contigui fino ad unico nodo di lettura.

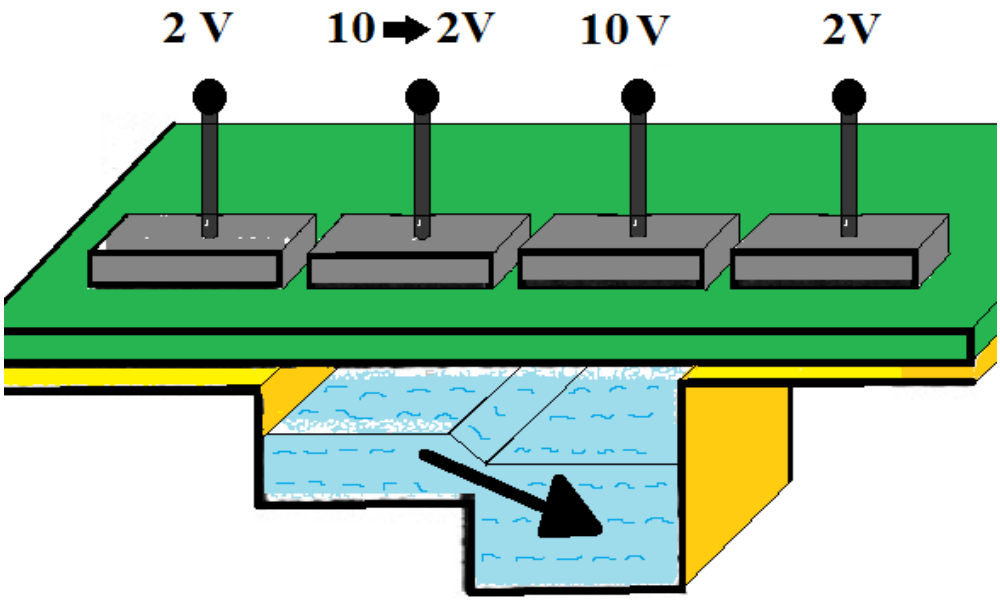


Illustrazione 4: Flusso di carica mediante modulazione del potenziale

Per trasferire il pacchetto di carica immagazzinato nel condensatore a potenziale più basso in un condensatore adiacente occorre applicare al suo

Gate una tensione più positiva in modo che le cariche migrino verso di esso. Dopo che il trasferimento di carica da un elettrodo all'altro è avvenuto si può riportare entrambe le tensioni a valori inferiori, in modo da consentire con le stesse tensioni un successivo trasferimento. Quindi, terminato l'intervallo di esposizione, ogni condensatore trasferisce la carica al suo vicino operando come un registro a scorrimento. L'ultimo condensatore svuota il carico in un amplificatore di carica che lo converte in voltaggio. Per evitare che la carica si disperda in direzione ortogonale alla linea di elettrodi vengono disposti degli impianti di lacune ad alta concentrazione che formano due "barriere" dette channel stops. Mentre gli elementi fotosensibili accumulano la carica relativa al nuovo ciclo, il contenuto del *CCD shift-register* viene trasferito all'esterno generando un segnale che rispetta uno degli standard video.

Nel CCD così descritto, chiamato surface channel, la differenza tra i reticoli del semiconduttore e dell'isolante creano delle imperfezioni che producono una notevole perdita di carica e rumore. Nel CCD di tipo buried channel si è suddiviso il substrato del semiconduttore creando una giunzione P-N che trasporta la carica con minore rumore e maggior efficienza.

La matrice di acquisizione immagini funziona nel seguente modo:

- Fase 1: scarica tutti i pixel del CCD.
- Fase 2: esposizione alla luce dei pixel
- Fase 3: Trasferimento di carica orizzontale
- Fase 4: Trasferimento di carica verticale

- Fase 5: Amplificazione delle cariche elettriche
- Fase 6: Conversione dei segnali elettrici in una immagine.

1.4.5 Complementary metal-oxide semiconductor

I sensori per l'acquisizione di immagini CMOS sono così chiamati poiché sfruttano la tecnologia Complementary Metal Oxide Semiconductor utilizzata per la progettazione di circuiti integrati. Il suo più importante vantaggio è quello del basso costo grazie alla possibilità di utilizzare un'unica linea di produzione con altri dispositivi digitali essendo analogo il processo di fabbricazione di microprocessori, memorie e altri chip.

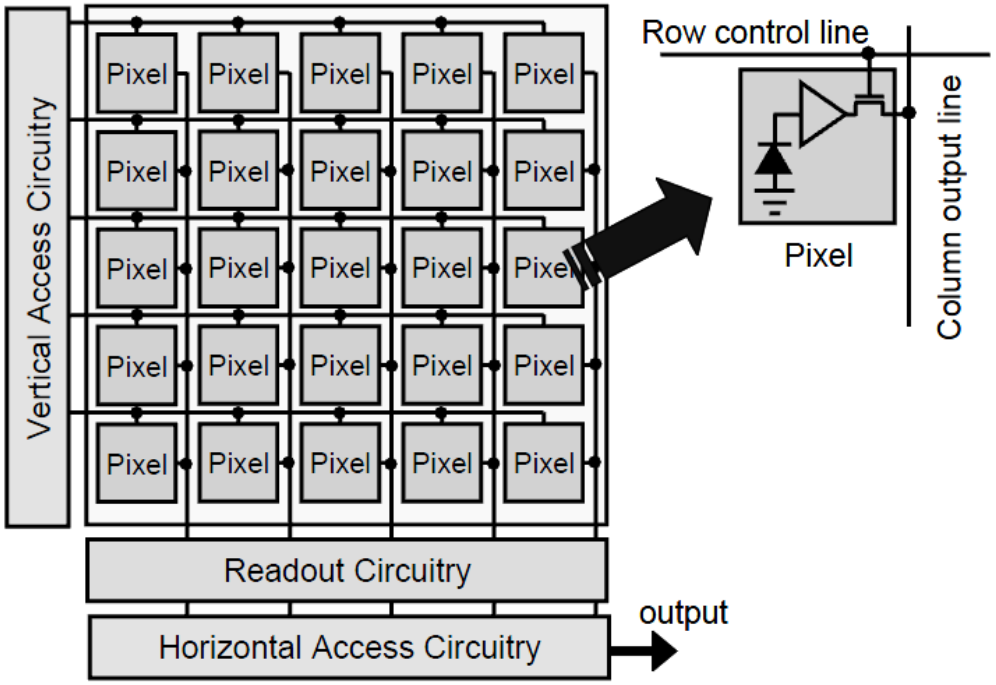


Illustrazione 5: Architettura di un sensore CMOS

Mentre in un sensore CCD il fotodiodo si comporta come un condensatore, nel CMOS si comporta come un generatore di corrente [Xie-2003].

Un sensore CMOS generalmente è costituito da un'area di acquisizione immagine che consiste in una matrice di active pixel sensors (APS) nella quale sono incorporati in ogni pixel sia il fotodiodo che dei transistor in grado di resettare il fotodiodo, convertire la carica elettrica in una tensione misurabile, amplificare il segnale (source follower) e ridurre il rumore. A questi si aggiungono anche dei circuiti ad accesso orizzontale e verticale e dei circuiti di lettura che servono per accedere a un pixel e a leggerne il valore del segnale oltre a dei circuiti di digitalizzazione [Otha 2007]. Questo tipo di architettura consente un indirizzamento più semplice ad ogni pixel rispetto ai CCD, tuttavia questi transistor di supporto non possono essere utilizzati per la rilevazione di fotoni, perciò la percentuale dell'area di un fotosito che risulta utile per raccogliere la luce, detta "fill factor" (fattore di riempimento), è soltanto il 30% dell'area totale.

La conseguenza è una perdita significativa di sensibilità, una corrispondente riduzione del rapporto segnale-rumore ed una limitata gamma dinamica. Un sensore CCD ha un fill factor del 100%, un sensore CMOS molto meno. In ogni caso, se ben progettati, sia i sensori CCD che CMOS possono offrire un'eccellente qualità dell'immagine.

Nel funzionamento di un CMOS il primo passo è quello di inizializzare il transistor di reset al fine di smaltire la carica della regione fotosensibile. Successivamente per effetto fotoelettrico vengono prodotti elettroni che vengono immagazzinati nella buca di potenziale sotto la superficie.

Terminato il tempo di esposizione, che dipende dall'intensità di luce, le cariche su ogni colonna vengono convertite in una voltaggio dal source follower dedicato.

1.4.6 Confronto fra CCD e CMOS

Ognuna delle due tecnologie mostra vantaggi e svantaggi che la rendono più adatta per certe applicazioni e non altre.

Le differenze principali sonno schematizzate nella seguente tabella:

CARATTERISTICA	CCD	CMOS
OUTPUT FOTODIODO	carica elettrica	voltaggio
OUTPUT DEL CHIP	voltaggio (analogico)	bit (digitale)
RUMORE	basso	moderato
COMPLESSITà COSTRUTTIVA	bassa	alta
PRECISIONE CROMATICA	alta	media
FATTORE DI RIEMPIMENTO	alto	moderato
CONSUMO DI ENERGIA	alto	basso
COSTO RELATIVO	medio - alto	medio - basso
DIMENSIONE OCCUPATA	medio - bassa	bassa

1.5 Image Processing

La percezione visuale artificiale consiste in un processo di informazione che prende immagini digitali come input e produce descrizioni degli oggetti in una scena come output [Xie-2003].

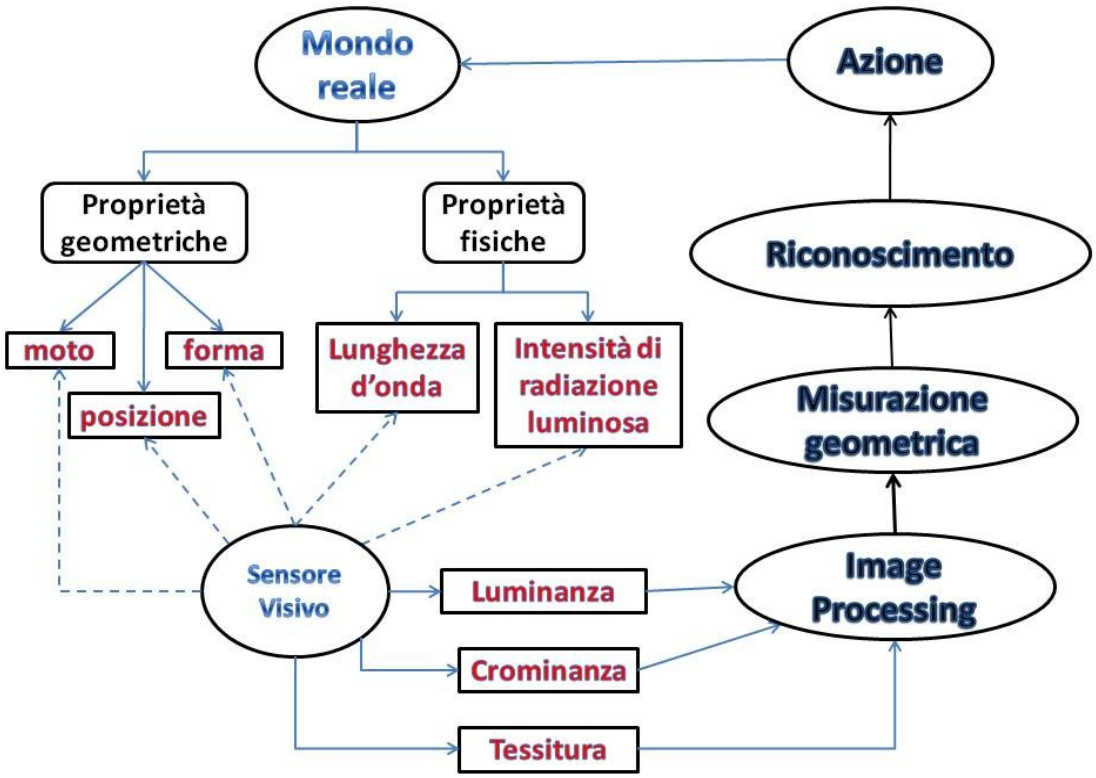


Illustrazione 6: Flusso di informazioni in un sistema di percezione visuale

Essa sfrutta in prima istanza le proprietà d'aspetto come crominanza, luminanza e tessitura (texture), contenute esplicitamente nelle immagini digitali a colori, per poter effettuare una elaborazione dell'immagine in modo da rilevare la continuità, la discontinuità e l'uniformità nelle

proprietà d'aspetto stesse.
Una volta terminato l'image processing è possibile estrarre i punti salienti, chiamati in letteratura in vari modi: features, keypoints o con i corrispettivi in lingua madre. Attraverso le features si ricavano le proprietà geometriche della scena, in termini di posizione, forma e moto.

1.5.1 Rumore nelle immagini

Nella computer vision, come in molte altre attività scientifiche, è importante attenuare quelle entità nei dati e nei risultati che ne possono degradare la qualità e che non sono di interesse.
In genere, in assenza di informazioni, si ipotizza che il rumore dell'immagine n(i,j) sia additivo e casuale così che l'immagine risultante sia data dalla somma dell'immagine "vera" più il contributo del rumore per ogni pixel in posizione (i,j). Inoltre si assume che n(i,j) sia caratterizzato dall'assenza di periodicità nel tempo e da ampiezza costante su tutto lo spettro di frequenze, cioè si tratti di rumore bianco e che segua una funzione di distribuzione Gaussiana a media zero di deviazione standard fissa. Questo garantisce bassi livelli di rumore ed è ciò che ci si aspetta da un buon sistema di acquisizione [Trucco-98].

1.5.2 Riduzione del rumore

Nel caso specifico si vuole cercare un filtro che attenui il più possibile il rumore in un'immagine senza alterarla. Il metodo più comune è lo smoothing che consiste nell'utilizzare un filtro lineare che si ottiene dalla

convoluzione tra l'immagine ed una matrice costante detta maschera o finestra, allo scopo di produrre una sfocatura in grado di celare le repentine variazioni di rapidità. Se sono esatte le ipotesi fatte sul tipo di rumore, questo può essere ridotto attraverso una media sul vicinato (neighborhood averaging) che si può ottenere semplicemente utilizzando una maschera di valori non negativi. Con questo metodo in genere si ha perdita di informazione, in quanto vengono perse le frequenze condivise con il rumore rendendo l'immagine sfocata e diffondendo allo stesso tempo il rumore impulsivo.

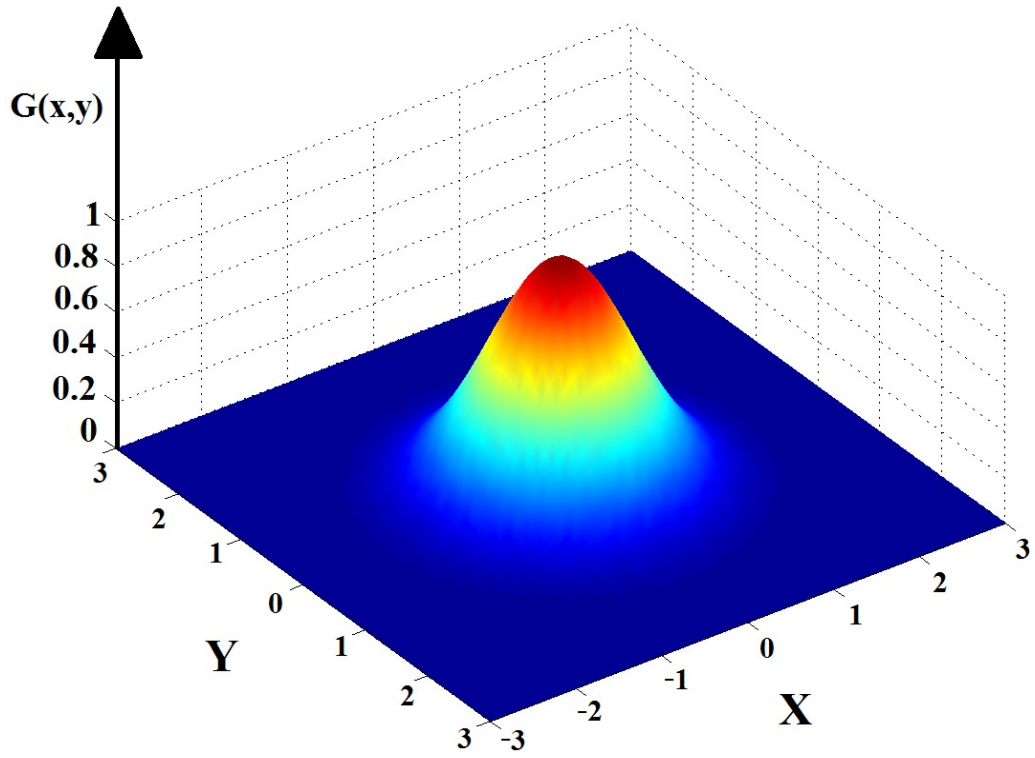


Illustrazione 7: Distribuzione gaussiana bidimensionale

Se i valori della maschera sono calcolati a partire dalla distribuzione Gaussiana 2D si parla di Gaussian smoothing.

La distribuzione Gaussiana bidimensionale si può ottenere dal prodotto di singole distribuzioni monodimensionali:

$$G(x,y)=\frac{1}{\sqrt{2\pi}\sigma_x}e^{-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2}\cdot\frac{1}{\sqrt{2\pi}\sigma_y}e^{-\frac{1}{2}\left(\frac{y-\mu_y}{\sigma_y}\right)^2}=\frac{1}{2\pi\sigma_x\sigma_y}e^{-\frac{1}{2}\left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2+\left(\frac{y-\mu_y}{\sigma_y}\right)^2\right]} \quad (1.1)$$

Se poniamo entrambi i valori medi a zero e ipotizziamo che le deviazioni standard lungo le due dimensioni siano identiche otteniamo la funzione semplificata.

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{1}{2}\frac{x^2+y^2}{\sigma^2}} \quad (1.2)$$

Una Gaussiana non è mai nulla, ma è prossima a zero a partire da tre deviazioni standard dalla media, perciò troncando adeguatamente i valori si può ottenere una finestra di dimensioni opportune.

1.6 Estrazione di feature da immagini

Le immagini digitali acquisite dai sistemi di visione dipendono dalla prospettiva e dalla illuminazione perciò le cause comuni per la discontinuità in una immagine sono dovute ai contorni delle superfici, alle ombre e alle occlusioni.

Possiamo definire formalmente una feature come qualsiasi discontinuità o uniformità che sia significativa e utile [Xie-2003].

Uno dei più grandi vantaggi dell'estrazione di feature sta nella significativa

riduzione dell'informazione (comparata all'immagine originale) nella rappresentazione di una immagine allo scopo di capirne il contenuto. La molteplicità di feature contenute in una immagine, sommate alla presenza di rumore, rendono arduo il compito di progettare un filtro che sia in grado di isolare una specifica caratteristica dalla scena. In genere se la qualità dell'immagine non è buona è necessario ridurre il rumore e successivamente esaltare la caratteristica di interesse attraverso un filtro specifico chiamato **feature** (o keypoint) **detector** che risponde in modo diverso a caratteristiche diverse. A questo punto è possibile avviare un processo decisionale per la selezione delle caratteristiche desiderate.

Una possibile distinzione tra tipi di features estratte è quella tra General Features e Domain Features [Lei-99]. Le prime sono dette descrittori di basso livello e sono indipendenti dal contesto dell'immagine.

Le Domain Features dipendono dal contesto dell'immagine e sono utilizzate prevalentemente per il riconoscimento di volti (face recognition) e per l'analisi delle impronte digitali (fingerprint identification). Qualora si voglia identificare questo tipo di descrittori è necessaria l'analisi delle features di basso livello, per tale motivo le Domain Features sono dette di alto livello.

I descrittori di basso livello possono essere in generale classificati in 3 tipi:

- *Blob*. Descrittori di livello zero, costituito da regioni uniformi, caratterizzate da ridotte variazioni dell'intensità.
- *Edge* (spigolo). Descrittore di primo livello che si ha in corrispondenza di discontinuità del gradiente monodirezionali.

- *Point features*, o *corner*, sono punti distinti dell'immagine localizzati in corrispondenza di discontinuità bidirezionali del gradiente dell'intensità luminosa. A seconda della forma assunta dal corner si distinguono giunzioni a T, Y, X o L.

1.6.1 Edge detection

Per la rilevazione di feature di alto livello è fondamentale la segmentazione dell'immagine per la quale l'edge detection gioca un ruolo molto importante. Un edge può essere considerato come un confine tra due regioni dissimili ed essenzialmente il contorno di un oggetto corrisponde ad un brusco cambiamento nei livelli di intensità. L'output dell'edge detection dovrebbe essere una mappa dei bordi nella quale il valore di ogni pixel riflette quanto sono verificati i requisiti per essere parte di un bordo da parte del pixel corrispondente nell'immagine originale.

1.6.1.1 Rilevazione dei bordi mediante derivata prima

Per rilevare la posizione di un bordo spesso è utilizzata la derivata del primo ordine, che è nulla in presenza di un segnale costante, mentre dovrebbe presentare il massimo in presenza di un edge.

La funzione di magnitudine del gradiente in un'immagine digitalizzata può essere così formulata:

$$d(x, y) = \sqrt{\Delta x^2 + \Delta y^2} \qquad (1.3)$$

con

$$\Delta x = I(x+1, y) - I(x, y)$$
$$\Delta y = I(x, y+1) - I(x, y) \qquad (1.4)$$

Spesso nell'image processing la derivata prima di un'immagine digitale viene effettuata attraverso la convoluzione con una maschera chiamata edge operator.

Per rilevare un edge in una immagine reale è opportuno effettuare una preliminare riduzione del rumore, dopodiché si effettua la convoluzione con l'edge operator e infine dall'output si eliminano i valori non corrispondenti ai massimi.

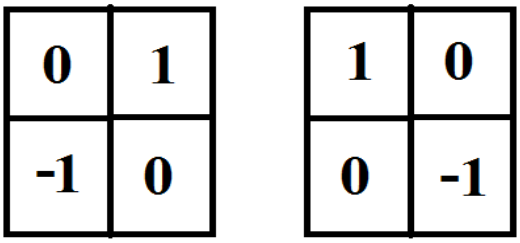


Illustrazione 8: Robert's Cross Operator

Le maschere più semplici per effettuare una derivata del primo ordine sono costituite da un vettore [1,0,-1] per rilevare bordi orizzontali e dal suo trasposto per bordi verticali.

Uno dei primi operatori utilizzati fu il Robert's Cross Operator che utilizza una maschera 2×2 , ma poiché la derivata prima esalta il rumore, analogamente ad un filtro passa-alto, i successivi edge operator sono costituiti da matrici 3×3 in modo da includere all'interno anche il neighborhood averaging.

Seguendo tale principio è stato ideato il filtro di Prewitt e il filtro di Sobel;

quest'ultimo però pesa in modo differente i pixel in base alla distanza dal pixel centrale.

-1	0	1
-1	0	1
-1	0	1

-1	-1	-1
0	0	0
1	1	1

Illustrazione 9: Operatore di Prewitt

I filtri di Prewitt e Sobel mostrano degli output frammentati seppur ben visibili.

-1	0	1
-2	0	2
-1	0	1

-1	-2	-1
0	0	0
1	2	1

Illustrazione 10: Operatore di Sobel

1.6.1.2 Rilevazione dei bordi mediante derivata del secondo ordine

Un'alternativa alla ricerca dei massimi della derivata del primo ordine è quella di trovare l'attraversamento dello zero nella derivata del second'ordine. La derivata seconda può essere approssimata con la differenza tra le derivate del primo ordine di due pixel adiacenti:

$$f''(x) \simeq \Delta_x - \Delta_{x+1} \qquad (1.5)$$

considerando la (1.4) si ha

$$f''(x) \simeq -\Delta_x + 2\Delta_{x+1} - \Delta_{x+2} \qquad (1.6)$$

0	-1	0
-1	4	-1
0	-1	0

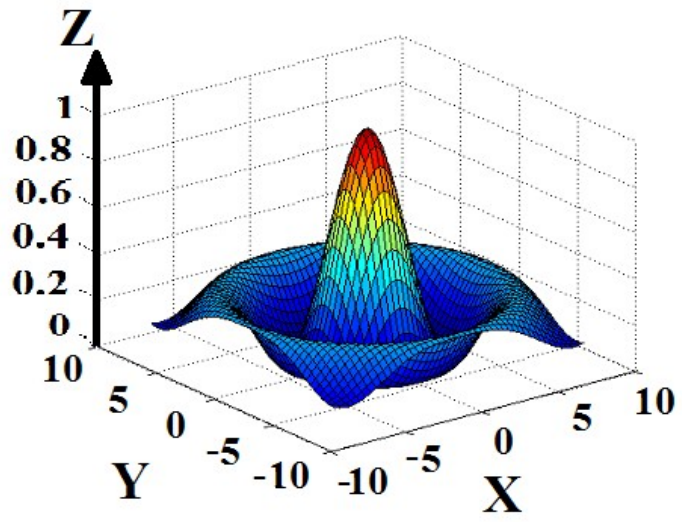
-1	-1	-1
-1	8	-1
-1	-1	-1

Illustrazione 11: Operatore Laplaciano

Un operatore che utilizza la derivata del secondo ordine è quello Laplaciano che sfrutta le maschere, rappresentate nell'illustrazione 11, ottenute dalla combinazione del vettore [-1, 2, -1] in orizzontale, verticale e diagonale.

Il laplaciano, non essendo utile per stabilire la direzione di un edge, viene utilizzato in genere per stabilire la posizione di uno spigolo che si fa coincidere con lo zero crossing. Inoltre questo filtro non viene utilizzato per il rilevamento dei bordi nella sua forma originale, sia perché eccessivamente sensibile al rumore, sia per il fatto che produce dei doppi bordi che complicano la segmentazione.

In genere si effettua il Laplaciano della Gaussiana (LoG) (illustrazione 12) ottenuto facendo, preliminarmente al Laplaciano, uno smoothing Gaussiano per ridurre il rumore e neutralizzare l'effetto di amplificazione del rumore causato dalla derivata seconda.



0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

Illustrazione 12: Laplaciano della Gaussiana e maschera che ne approssima la forma

1.6.1.3 Canny edge detector

Canny ha cercato di ottenere un rilevatore di bordi che fosse in grado di verificare i seguenti requisiti [Canny-83]:

1. Buon riconoscimento: rilevare il maggior numero possibile di spigoli nell'immagine
2. Buona localizzazione: i contorni rilevati devono essere il più vicino possibile ai contorni reali dell'immagine.
3. Singola risposta: Per ogni edge reale c'è solo una risposta, vale a dire che non è permessa la rilevazione di “falsi” edge.
4. Massimizzare il rapporto segnale-rumore.

5. Minimizzare la distanza della radice della media dei quadrati dell'edge rilevato dal centro del vero edge.
6. Minimizzare la distanza media tra i massimi locali causati dal rumore

L'algoritmo inizia con uno smoothing Gaussiano, di cui la dimensione della maschera è uno dei parametri arbitrari, per ridurre il rumore seguito dall'applicazione di quattro filtri per individuare gli spigoli nelle direzioni principali. Il filtro che ottiene il valore più grande corrisponde al massimo gradiente di luminosità e quindi stabilisce la direzione dell'edge per uno specifico pixel.

A questo punto vanno selezionati i massimi locali che in genere corrispondono ai bordi, ma a differenza degli algoritmi precedenti in questo caso si utilizza un thresholding con due soglie. La soglia inferiore serve ad eliminare tutti i valori troppo piccoli, quella superiore invece divide i pixel tra quelli maggiori della soglia che fanno sicuramente parte di un contorno e quelli compresi tra le due soglie che vengono selezionati soltanto se contigui ad un pixel precedentemente scelto. È importante impostare opportunamente i valori delle soglie per non incorrere in perdita di informazione o al contrario in eccesso di elementi non significativi.

1.6.2 Corner detection

Gli angoli sono features più abbondanti rispetto ai contorni rettilinei nel mondo naturale, cosa che li rende ideali da tracciare; sono inoltre invarianti rispetto ai cambiamenti di illuminazione. Questa caratteristica li rende

privilegiati rispetto ad altri descrittori nel motion tracking in quanto sono esplicitamente inseguibili perchè meno soggetti ad ambiguità e perciò più facilmente identificabili in immagini successive.
Queste caratteristiche sono estraibili anche da immagini in toni di grigio, riducendo a un terzo il carico computazionale e facendo risultare l'utilizzo di tali features adatto al real time processing.
Fino ad oggi sono stati sviluppati numerosi algoritmi per la rilevazione di feature, che sostanzialmente si differenziano per la replicabilità dei corner estratti dall'immagine e per il carico computazionale utilizzato. Alcuni possiedono anche le caratteristiche di essere invarianti alle rotazioni e/o ai cambiamenti di scala. La capacità dell'estrattore di rilevare le stesse feature in immagini che mostrano lo stesso contenuto da un punto di vista poco differente è fondamentale per la ricerca delle corrispondenze. In assenza di tale proprietà, feature estratte possono diventare inutili per i passi successivi dell'algoritmo o rappresentare una sorgente d'errore, conducendo a corrispondenze errate e rendendo meno robusta la stima del moto.

1.6.2.1 Classificazione degli algoritmi di edge detection

Negli ultimi anni sono stati pubblicati vari algoritmi per il rilevamento dei *corner*, che possiamo dividere secondo due approcci:

- *Edge Based*: Algoritmi che si basano sull'estrazione dei contorni e la successiva identificazione dei punti a massima curvatura, o ai punti di intersezione tra gli edge. Questo approccio risente tuttavia della qualità

della segmentazione dell'immagine acquisita.

- *Raw Based*: Approccio che si basa sull'analisi diretta delle variazioni di intensità in immagini in scala di grigio. I corner sono indipendenti dalle variazioni di luminosità, quindi si può giungere all'identificazione attraverso una semplice analisi dell'intensità del gradiente dell'immagine acquisita. I metodi raw based sono i più numerosi, ma di solito necessitano di un ulteriore analisi degli intorni degli angoli in quanto tendono a rilevare falsi corner.

Gli algoritmi di corner detection sono più frequentemente divisi in:

- Rilevatori a scala fissa (*fixed scale detector*)
- Rilevatori multi scala (*multi scale detector*).

Tra i primi i principali algoritmi sono:

- Moravec
- Harris
- SUSAN
- FAST

I fixed scale detector calcolano la risposta a partire da maschere di dimensioni prefissate. Poiché gli oggetti in genere hanno significato soltanto in un certo intervallo di scale, l'operatore applicato ha successo soltanto se la grandezza della finestra è proporzionata a quella della struttura di cui si vogliono identificare i punti caratteristici. Questi algoritmi in genere mostrano problemi nella ripetibilità dei risultati su diverse scale della stessa immagine. Per risolvere questo genere di inconveniente sono stati sviluppati algoritmi invarianti alla scala che

riescono ad individuare sulla stessa immagine, ingrandita o rimpicciolita, gli stessi punti salienti con un discreto margine di errore.
Tra i multi scale detector i più noti sono:

- SIFT
- Harris-Laplace
- SURF

1.6.2.2 Algoritmo di Moravec

Tra gli algoritmi di rilevamento raw based troviamo il metodo di Moravec [Moravec-79].
L'operatore di Moravec considera punti di interesse quei punti che sono caratterizzati da una considerevole variazione di intensità in verticale, orizzontale o diagonale. È considerato un corner detector pur rilevando anche punti isolati.
La variazione di intensità viene misurata a partire dal pixel $I(x,y)$ su cui viene centrata una piccola finestra quadrata W dalle misure in pixel 3×3, 5×5, o 7×7. Spostando la finestra nelle direzioni suddette di una quantità discreta $(\Delta x, \Delta y)$ viene calcolata la variazione di intensità attraverso il criterio SSD (Sum of Squared Differences) che ha il duplice effetto di eliminare il segno e di amplificare la differenza tra piccole e grosse variazioni:

$$SSD(x,y)=\sum_{\forall x_k, y_k \in W}\left[I(x_k,y_k)-I(x_k+\Delta x, y_k+\Delta y)\right]^2 \qquad (1.7)$$

Il passo successivo è quello di costruire una mappa che indichi per ogni pixel dell'immagine il minimo tra le variazioni di intensità calcolate sulle

otto direzioni principali.

Questo coefficiente, detto cornerness, sara dato da:

$$C(x,y)=min(SSD(x,y)) \tag{1.8}$$

Il passo finale è quello di fissare una soglia al di sotto della quale i valori di C(x,y) siano posti a zero in modo da sopprimere i non massimi allo scopo di trovare i massimi locali. I punti rimanenti diversi da zero nella mappa sono corner.

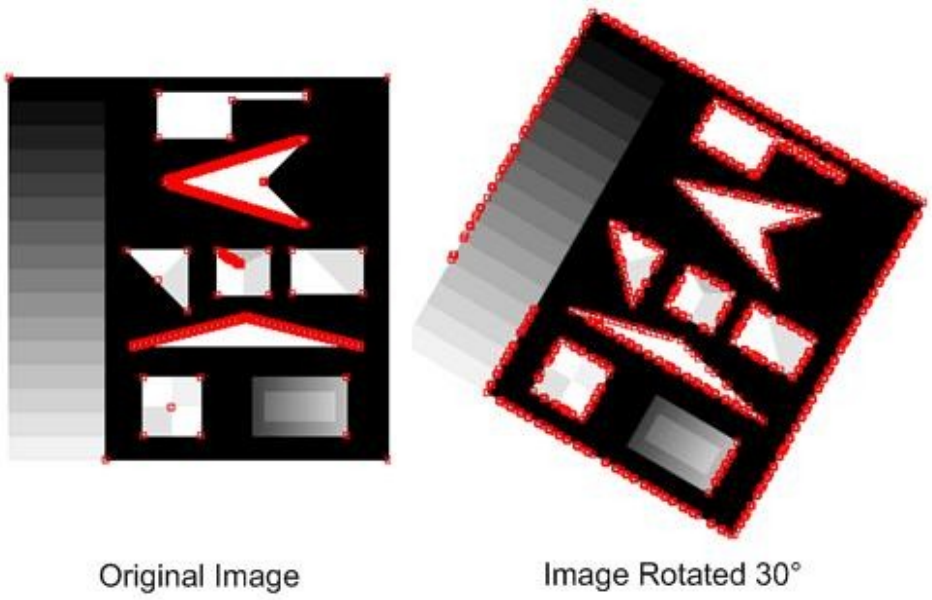


Illustrazione 13: Risposta anisotropa dell'operatore di Moravec
http://kiwi.cs.dal.ca/~dparks/CornerDetection/moravec.htm

Uno dei difetti più evidenti di questo approccio è che, poiché la finestra si muove soltanto lungo le direzioni principali, gli edge aventi direzione diversa hanno valori di cornerness molto elevati pur non essendo né corner né punti isolati. Questo significa che la risposta è anisotropa, essendo

calcolata su un insieme finito di spostamenti, e, di conseguenza, l'operatore ha un basso tasso di ripetitività non essendo invariante alla rotazione.La finestra usata da Moravec è quadrata, perciò per i pixel lungo i bordi della finestra la distanza Euclidea dal centro è maggiore ed è ulteriormente accentuata per quelli ai quattro angoli. Inoltre, tutti i pixel della finestra hanno lo stesso peso, ma intuitivamente i pixel vicini al centro danno una indicazione migliore della variazione di intensità locale.

L'uso di una finestra gaussiana risolve entrambi i problemi, infatti i valori lontani dal centro sono molto piccoli e crescono avvicinandosi al centro, approssimando una finestra circolare. La deviazione standard della gaussiana fissa la scala spaziale alla quale si vogliono determinare gli angoli.

W1 .004	W2 .015	W3 .026	W4 .015	W5 .004
W6 .015	W7 .059	W8 .095	W9 .059	W10 .015
W11 .026	W12 .095	W13 .15	W14 .095	W15 .026
W16 .015	W17 .059	W18 .095	W19 .059	W20 .015
W21 .004	W22 .015	W23 .026	W24 .015	W25 .004

Illustrazione 14:
Gaussian window

La variazione di intensità allora viene calcolata come:

$$V(x,y)=\sum_{\forall x_k, y_k \in W} w_k \left[I(x_k, y_k) - I(x_k+\Delta x, y_k+\Delta y) \right]^2 \qquad (1.9)$$

Per evitare di calcolare la funzione nelle varie direzioni si considera lo

sviluppo in serie di Taylor troncato al primo ordine.

$$I(\bar{x}+h) \simeq I(\bar{x}) + \nabla I(\bar{x})^T h = I(\bar{x}) + \left[\frac{\partial I(\bar{x})}{\partial x} \quad \frac{\partial I(\bar{x})}{\partial y} \right]^T h \tag{1.10}$$

In cui $\bar{x} = \begin{pmatrix} x_k \\ y_k \end{pmatrix}$ e $h = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$ è un piccolo vettore spostamento.

La (1.10) può essere riscritta in modo più intuitivo considerando che l'immagine è descritta su un dominio discreto

$$I\begin{pmatrix} x_k + \Delta x \\ y_k + \Delta y \end{pmatrix} \simeq I\begin{pmatrix} x_k \\ y_k \end{pmatrix} + \left[I\begin{pmatrix} x_k + \Delta x \\ y_k \end{pmatrix} - I\begin{pmatrix} x_k \\ y_k \end{pmatrix} \quad I\begin{pmatrix} x_k \\ y_k + \Delta y \end{pmatrix} - I\begin{pmatrix} x_k \\ y_k \end{pmatrix} \right] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{1.11}$$

Dalla (1.10) con dei semplici passaggi algebrici si ottiene:

$$\left(I(\bar{x}+h) - I(\bar{x}) \right)^2 = \left(\nabla I(\bar{x})^T h \right)^2 = h^T \nabla I(\bar{x}) \left(\nabla I(\bar{x}) \right)^T h \tag{1.12}$$

Che può essere sostituita nella (1.9)

$$\begin{aligned} V(x,y) &= \sum_{\forall \bar{x} \in W} w(\bar{x}) h^T \nabla I(\bar{x}) \left(\nabla I(\bar{x}) \right)^T h = \\ &= h^T \left(\sum_{\forall \bar{x} \in W} w(\bar{x}) \nabla I(\bar{x}) \left(\nabla I(\bar{x}) \right)^T \right) h \end{aligned} \tag{1.13}$$

Sviluppando il prodotto all'interno della sommatoria

$$V(x,y) = h^T M h \tag{1.14}$$

con

$$M = \begin{bmatrix} \sum_{\forall \bar{x} \in W} w(\bar{x}) \left(\frac{\partial I(\bar{x})}{\partial x} \right)^2 & \sum_{\forall \bar{x} \in W} w(\bar{x}) \frac{\partial I(\bar{x})}{\partial x} \frac{\partial I(\bar{x})}{\partial y} \\ \sum_{\forall \bar{x} \in W} w(\bar{x}) \frac{\partial I(\bar{x})}{\partial x} \frac{\partial I(\bar{x})}{\partial y} & \sum_{\forall \bar{x} \in W} w(\bar{x}) \left(\frac{\partial I(\bar{x})}{\partial y} \right)^2 \end{bmatrix} \tag{1.15}$$

Questa relazione è strettamente connessa con la funzione di autocorrelazione locale.

M è semi-definita positiva, perciò i suoi autovalori saranno non negativi, inoltre M è simmetrica, pertanto può essere scomposta tramite

Diagonalizzazione di Schur attraverso una matrice ortogonale S le cui colonne sono gli autovettori di M.

$$S^T M S = \Lambda = diag(\lambda_1, \lambda_2) \quad (1.16)$$

Gli autovalori sono un indice della variazione di intensità dell'immagine essendo proporzionali alle curvature principali dell'autocorrelazione locale e formano un descrittore invariante per rotazione della matrice M, infatti:

- Se gli autovalori sono entrambi nulli o molto vicini a zero nel punto considerato, l'immagine sarà uniforme e quindi è presente un blob.
- Se soltanto uno è prossimo a zero, mentre l'altro è molto più grande si ha un edge. L'autovettore associato all'autovalore più grande è perpendicolare all'edge
- Se entrambi gli autovalori sono grandi si ha un corner.

Essendo $M = \begin{bmatrix} A & B \\ B & C \end{bmatrix}$ una matrice 2×2 , gli autovalori sono facilmente calcolabili in forma chiusa dal calcolo del polinomio caratteristico

$$\lambda_{1,2} = \frac{1}{2}\left(A + C \pm \sqrt{(A-C)^2 + 4B^2}\right) \quad (1.17)$$

1.7 Trasformata di Hough

Un metodo di feature extraction di cui si è fatto un importante uso nel progetto in discussione è la trasformata di Hough. Questa procedura consente di trovare linee, cerchi, o altre semplici forme in un immagine. La trasformata di Hough classica [Hough59] concerneva l'identificazione di linee per uso in esperimenti fisici, mentre l'attuale uso della trasformata,

detta standard (SHT) o generalizzata, è dovuto a R. Duda e P. Hart [Duda72] e può essere esteso ad altri casi rispetto alle semplici linee.

1.7.1 La trasformata di Hough per le linee

Il caso più semplice è quello in cui si vogliono rilevare gruppi di punti collineari in una immagine in bianco e nero. Dati n punti verificare ogni coppia di punti è computazionalmente proibitivo risultando $O(n^2)$.

Il metodo di Hough prevede la trasformazione di tutti i punti della figura su una linea retta in uno spazio dei parametri. La descrizione in forma esplicita della retta consente di parametrizzare ogni linea con inclinazione m e intercetta q , trasformando ogni punto nell'immagine nel luogo dei punti nel piano (m,q) corrispondente a tutte le linee passanti attraverso quel punto. Se preso ogni pixel diverso da 0 nell'immagine in input lo convertiamo in tale insieme di punti e sommiamo tutti tali contributi, allora alle linee che appaiono nell'immagine corrisponderanno dei massimi locali nel piano (m,q) in output, detto piano accumulatore. In altre parole ciascun punto nello spazio dei parametri fornisce il proprio contributo, o voto, alla ricerca della soluzione.

Purtroppo il dominio in cui sono definite pendenza e intercetta non sono limitati e quindi difficilmente rappresentabili computazionalmente. L'insieme delle linee rette nel piano costituisce una famiglia a due parametri, se uno di questi due parametri viene fissato possiamo rappresentare una linea retta attraverso un singolo punto. La parametrizzazione proposta da Duda e Hart consiste nel rappresentare ogni

linea come un punto in coordinate polari (ρ,θ), in cui ρ indica la distanza dall'origine, mentre θ è l'angolo che la normale alla retta forma con il semiasse positivo delle ascisse. In altre parole la linea passante attraverso il punto indicato è perpendicolare al raggio dall'origine a quel punto. L'equazione per tale linea è

$$\rho = x\cos\theta + y\sin\theta$$

Se restringiamo il dominio di theta all'intervallo $[0,\pi]$ allora il parametro relativo all'angolo è unico. Con questa restrizione ogni linea nel piano (x,y) corrisponde ad un unico punto nel piano (ρ,θ).

Dato un insieme di n punti dell'immagine possiamo trovare un insieme di linee rette passanti tra loro trasformando i punti in curve sinusoidali nel piano (ρ,θ); infatti le curve corrispondenti a punti collineari hanno un punto in comune di intersezione in questo piano. Il problema della rilevazione di punti collineari viene così convertito nel problema di trovare le curve concorrenti. Essendo valida anche la proprietà duale possiamo riassumere queste interessanti proprietà della trasformazione punto-curva come segue

1. Un punto nel piano immagine corrisponde ad una curva sinusoidale nel piano dei parametri.
2. Un punto nel piano dei parametri corrisponde ad una linea retta nel piano immagine.
3. Punti giacenti sulla stessa linea retta nel piano immagine corrispondono a curve attraverso un punto comune nel piano dei parametri.

4. Punti giacenti sulla stessa curva nel piano dei parametri corrispondono a linee attraverso lo stesso punto nel piano immagine.

I punti di accumulazione rappresentano le candidate rette presenti nell'immagine. Il costo computazionale di questo algoritmo è proporzionale al numero di punti che costituiscono la regione di interesse. Per ridurre ulteriormente la complessità si può far uso della trasformata di Hough progressiva probabilistica (PPHT), la quale piuttosto che accumulare ogni possibile punto nel piano accumulatore, lo fa solo per una loro frazione. Si congettura che in questo modo il picco possa essere comunque abbastanza alto, quindi è possibile trovare una soluzione riducendo considerevolmente il tempo computazionale. La PPHT permette oltre al calcolo dell'orientazione delle linee anche quello della loro estensione.

1.7.2 La trasformata di Hough per i cerchi (CHT)

L'approccio generale della trasformata di Hough può essere esteso ad altri generi di curve. Ad esempio per rilevare configurazioni circolari di punti si può scegliere una rappresentazione parametrica per la famiglia di tutti i cerchi. Una rappresentazione può essere data da

$$(x-a)^2+(y-b)^2=R^2$$

In questo modo possiamo trasformare un arbitrario punto della figura nello spazio tridimensionale dei parametri (a,b,R) . Se il punto giace su un cerchio il luogo nello spazio dei parametri è dato da un cono circolare

retto. Se un insieme di punti sono disposti sul perimetro di un cerchio con parametri (a_0, b_0, R_0) allora i luoghi dei parametri risultanti da ognuno di tali punti passeranno attraverso lo stesso punto (a_0, b_0, R_0) nello spazio dei parametri. Quindi tutti i rispettivi coni circolari retti si intersecheranno in un punto comune.

Per rappresentare le triplette che descrivono ogni cerchio si dovrebbe fare uso di un array a tre dimensioni rimpiazzando il piano accumulatore con un volume accumulatore e di conseguenza con maggiore lentezza rispetto al caso delle linee e grande richiesta di memoria. L'uso del metodo del gradiente di Hough [Ballard-79] permette di evitare tali problemi. Se conosciamo l'informazione associata al gradiente di un punto su un edge possiamo ridurre il luogo da un cono ad una linea, poiché il centro del cerchio per quel punto dovrà giacere ad una distanza R lungo la direzione del gradiente. Prima l'immagine passa attraverso una fase di edge detection, ad esempio attraverso l'algoritmo di Canny. In tal modo si può passare da una immagine a colori ad una immagine in bianco e nero nella quale per i pixel diversi da zero deve essere considerato il gradiente locale. Fortunatamente questo può essere fatto semplicemente applicando l'operatore di Sobel in grado di stimare l'orientazione di un edge. Una volta applicata la convoluzione con la maschera di Sobel sono disponibili le componenti locali del gradiente (g_x, g_y) da cui possiamo calcolare la grandezza del vettore del gradiente dell'intensità locale [Davies-2005]:

$$g = \sqrt{\left(g_x^2 + g_y^2\right)}$$

e la sua orientazione

$$\theta = \arctan(g_y / g_x)$$

Scrivendo l'equazione parametrica del cerchio finora considerato come

$$x = a + R\cos\theta$$
$$y = b + R\sin\theta$$

Essendo poi

$$\cos\theta = g_x / g \quad \text{e} \quad \sin\theta = g_y / g$$

possiamo stimare le coordinate del luogo dei centri con

$$a = x - R(gx/g)$$
$$b = y - R(gy/g) \qquad (1.18)$$

Se la direzione dell'egde è precisa e si conosce il valore di R allora nell'accumulatore vengono aumentati i punti per i valori (a,b) che soddisfano la (1.18). Tenendo però conto che è improbabile che sia precisa la stima della direzione di un edge possiamo aggiunge una tolleranza dovuta all'errore. In questo accumulatore bidimensionale vengono poi selezionati i centri candidati, i quali devono superare una certa soglia ed essere maggiori dei punti a loro più vicini. I candidati sono poi ordinati in base ai valori dell'accumulatore in ordine decrescente, in modo da individuare prima i centri con più voti. In molti casi non si conosce con certezza il valore del raggio e perciò è opportuno imporre per questo una distanza massima e minima. I pixel non nulli vengono quindi ordinati in base alla loro distanza dal centro e selezionati in modo da lavorare nell'intervallo scelto. Il singolo raggio maggiormente supportato da questi pixel viene infine selezionato se tale supporto è sufficientemente elevato e se ha una distanza sufficiente da qualsiasi altro centro selezionato in precedenza.

Capitolo 2

Modello della telecamera e calibrazione

2.1 Modello della telecamera

Il modello geometrico più semplice che descrive il processo di acquisizione dell'immagine da parte di un sistema di visione artificiale è quello della proiezione centrale di un punto dello spazio su un piano.

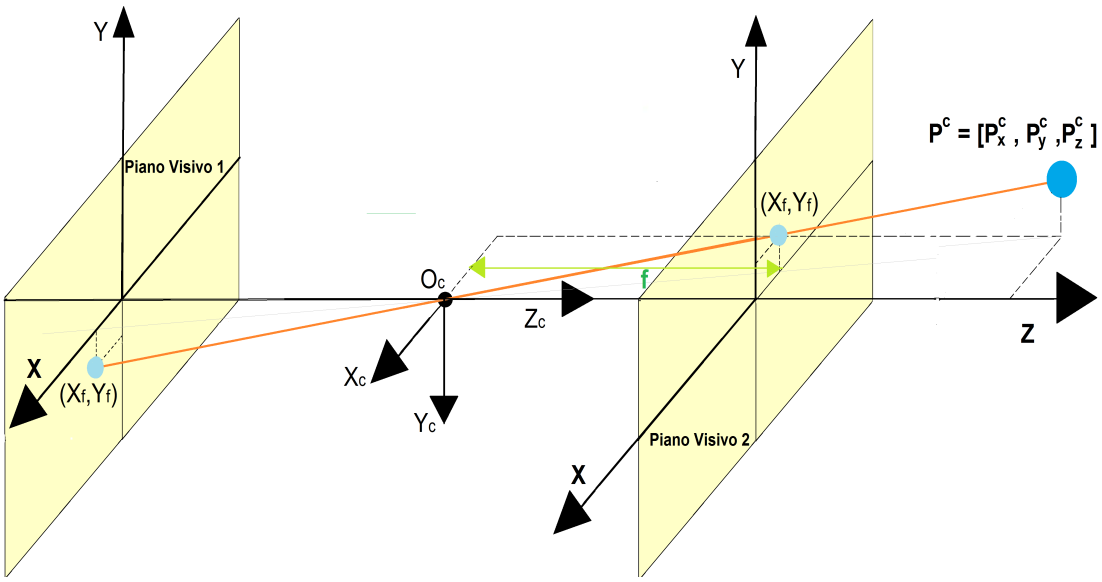


Illustrazione 15: Trasformazione prospettica

2.1.1 Proiezione prospettica

La proiezione della scena tridimensionale sul piano immagine effettuata da ogni telecamera può essere rappresentata da una matrice che mappa i punti da $\mathbb{R}^3$ in $\mathbb{R}^2$. Si consideri a tal proposito una terna $O_c - X_c Y_c Z_c$ solidale alla telecamera con l'origine nel centro di proiezione e Z_c perpendicolare

al piano immagine e detto *asse principale* o asse ottico. Il piano parallelo al piano visivo e passante per il centro ottico viene chiamato *piano focale*. Il punto di intersezione tra l'asse principale e il piano immagine è detto *punto principale*. Tale punto nella fotocamera stenopeica che rappresenta il modello ideale più semplice, anche detta stenoscopio (o pinhole camera), coincide con l'origine della griglia di pixel del sensore visivo.

Sfruttando la similitudine tra triangoli si può ricavare la trasformazione prospettica frontale che lega le coordinate $P^c = \left[P_x^c P_y^c P_z^c \right]$ di un punto nello spazio 3D in un punto nel piano immagine (il piano visivo 2 dell'illustrazione 15) distante f (*lunghezza focale*) da O_c , cioè:

$$\mathbb{R}^3 \to \mathbb{R}^2 : \begin{pmatrix} P_x^c \\ P_y^c \\ P_z^c \end{pmatrix} \to \begin{pmatrix} f \dfrac{P_x^c}{P_z^c} \\ f \dfrac{P_y^c}{P_z^c} \end{pmatrix} = \begin{pmatrix} X_f \\ Y_f \end{pmatrix} \qquad (2.1)$$

La divisione per la componente lungo l'asse principale causa l'effetto di scorcio per cui la dimensione di un oggetto risulta inversamente proporzionale alla distanza dello stesso dall'osservatore. Tale divisione rende inoltre non lineare questa trasformazione detta proiezione prospettica.

Le coordinate nel piano immagine sono solitamente campionate in pixel. L'espressione che lega le coordinate relative alla terna solidale alla telecamera (*camera reference frame*) a quelle della griglia di pixel del sensore visivo, si può ottenere attraverso una trasformazione affine che introduce due fattori di scala diversi lungo x e y che legano le unità

metriche al pixel e aggiungendo l'offset tra l'origine della griglia di pixel e il punto principale:

$$\begin{pmatrix} P_x^c \\ P_y^c \\ P_z^c \end{pmatrix} \rightarrow \begin{pmatrix} \alpha_x f \dfrac{P_x^c}{P_z^c} + X_0 \\ \alpha_y f \dfrac{P_y^c}{P_z^c} + Y_0 \end{pmatrix} = \begin{pmatrix} X_I \\ Y_I \end{pmatrix} \tag{2.2}$$

2.1.2 Parametri intrinseci ed estrinseci

Questa relazione non lineare si può esprimere in coordinate omogenee ottenendo:

$$P_z^c \begin{pmatrix} X_I \\ Y_I \\ 1 \end{pmatrix} = P_z^c \, p = \begin{bmatrix} \alpha_x f & 0 & X_0 & 0 \\ 0 & \alpha_y f & Y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} P_x^c \\ P_y^c \\ P_z^c \\ 1 \end{pmatrix} = \Psi \, P^c \tag{2.3}$$

La matrice $\Psi = \begin{bmatrix} \alpha_x f & 0 & X_0 & 0 \\ 0 & \alpha_y f & Y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [M \;\; 0]$ è detta **matrice di proiezione prospettica**, mentre la matrice M costituita dalle prime tre colonne della matrice Ψ codifica i parametri intrinseci della telecamera ed è detta **matrice di prospezione** [Scaramuzza 2003].

Per passare dalle coordinate telecamera a quelle spaziali (*world reference frame*), relative alla terna base, si deve effettuare il seguente cambio di coordinate:

$$\begin{pmatrix} P_x^c \\ P_y^c \\ P_z^c \end{pmatrix} = R \begin{pmatrix} P_x^s \\ P_y^s \\ P_z^s \end{pmatrix} + T$$

che in coordinate omogenee diventa:

$$P^c = \begin{pmatrix} P_x^c \\ P_y^c \\ P_z^c \\ 1 \end{pmatrix} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{pmatrix} P_x^s \\ P_y^s \\ P_z^s \\ 1 \end{pmatrix} = G \begin{pmatrix} P_x^s \\ P_y^s \\ P_z^s \\ 1 \end{pmatrix} \tag{2.4}$$

La matrice G codifica i *parametri estrinseci* della telecamera.

Sostituendo la (2.4) nella (2.3) si ottiene:

$$P_z^c \begin{pmatrix} X_I \\ Y_I \\ 1 \end{pmatrix} = P_z^c \, p = \Psi \, G \begin{pmatrix} P_x^s \\ P_y^s \\ P_z^s \\ 1 \end{pmatrix} = \overline{\Psi} \begin{pmatrix} P_x^s \\ P_y^s \\ P_z^s \\ 1 \end{pmatrix} \tag{2.5}$$

con $\overline{\Psi} = \Psi \, G = [M \quad 0] \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} = M \, [R \quad T]$

Ponendo $R = \begin{bmatrix} r_1^T & r_2^T & r_3^T \end{bmatrix}^T$ e $T = \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^T$ si ottiene

$$\overline{\Psi} = \begin{bmatrix} f \alpha_x r_1^T + X_0 r_3^T & f \alpha_x t_1 + X_0 t_3 \\ f \alpha_y r_2^T + Y_0 r_3^T & f \alpha_y t_2 + Y_0 t_3 \\ r_3^T & t_3 \end{bmatrix} \tag{2.6}$$

La matrice $\overline{\Psi}$ è una matrice 3×4 , perciò ha 12 elementi, ma ha solo 11 gradi di libertà perdendone uno a causa del fattore di scala P_z^c .

Esprimendo con $p_{i,j}$ gli elementi della matrice si ottiene:

$$P_z^c \begin{pmatrix} X_I \\ Y_I \\ 1 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{pmatrix} P_x^s \\ P_y^s \\ P_z^s \\ 1 \end{pmatrix} \tag{2.7}$$

Per eliminare il fattore di scala imposto le seguenti due equazioni non lineari

$$\begin{cases} X_I = \dfrac{p_{11} P_x^s + p_{12} P_y^s + p_{13} P_z^s + p_{14}}{p_{31} P_x^s + p_{32} P_y^s + p_{33} P_z^s + p_{34}} \\ Y_I = \dfrac{p_{21} P_x^s + p_{22} P_y^s + p_{23} P_z^s + p_{24}}{p_{31} P_x^s + p_{32} P_y^s + p_{33} P_z^s + p_{34}} \end{cases} \tag{2.8}$$

Ponendo $\overline{\Psi} = \begin{bmatrix} p_1^T & p_2^T & p_3^T \end{bmatrix}^T$, avendo messo in evidenza le righe che compongono $\overline{\Psi}$, possiamo riscrivere la relazione (2.8) in modo sintetico:

$$\begin{cases} X_I = \dfrac{p_1^T P^s}{p_3^T P^s} \\ Y_I = \dfrac{p_2^T P^s}{p_3^T P^s} \end{cases} \tag{2.9}$$

2.1.3 Il nucleo di $\overline{\Psi}$

Il centro ottico può essere pensato come l'intersezione tra i tre piani definiti dagli assi del sistema di riferimento telecamera. I punti dello spazio appartenenti al piano passante per X_c e l'asse principale, sono proiettati lungo l'asse X_c pertanto hanno equazione $p_2^T P^s = 0$, analogamente i punti che si proiettano lungo Y_c hanno equazione $p_1^T P^s = 0$. I punti appartenenti al piano focale invece vengono proiettati all'infinito perciò sarà il denominatore delle equazioni (2.9) ad annullarsi $p_3^T P^s = 0$. In definitiva il centro ottico si può ottenere dal seguente sistema di equazioni:

$$\begin{cases} p_1^T \begin{bmatrix} O_c \\ 1 \end{bmatrix} = 0 \\ p_2^T \begin{bmatrix} O_c \\ 1 \end{bmatrix} = 0 \\ p_3^T \begin{bmatrix} O_c \\ 1 \end{bmatrix} = 0 \end{cases} \Rightarrow \overline{\Psi} C = M \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} O_c \\ 1 \end{bmatrix} = M R O_c + M T = 0 \tag{2.10}$$

Da questa si deduce che si può esprimere T in funzione del centro ottico:

$$T = -R\,O_c \tag{2.11}$$

Quindi la matrice di proiezione prospettica può essere espressa così:

$$\overline{\Psi} = M\begin{bmatrix} R & -R\,O_c \end{bmatrix}$$

La retta passante per il centro ottico e per un punto (X_I, Y_I) è detta *raggio ottico* e tutti i punti appartenenti a tale retta sono proiettati su tale punto.

Indicando con $\overline{\Psi}^+$ la pseudoinversa della matrice di proiezione prospettica, tutti i punti

$$P^s = \overline{\Psi}^+ p = \begin{bmatrix} (M\ R)^{-1}\begin{bmatrix} X_I \\ Y_I \\ 1 \end{bmatrix} \\ 0 \end{bmatrix}$$

appartengono al raggio ottico, infatti sostituendo tali punti nella (2.5) si ottengono le coordinate nel piano immagine.

Considerando che l'equazione (2.5) dipende da uno scalare e che per la (2.10) il centro ottico è il nucleo di $\overline{\Psi}$ possiamo ricavare l'equazione parametrica del raggio ottico:

$$P^S = \lambda\, C + \overline{\Psi}^+ p = \begin{bmatrix} O_c \\ 1 \end{bmatrix} + \lambda \begin{bmatrix} (M\ R)^{-1}\begin{bmatrix} X_I \\ Y_I \\ 1 \end{bmatrix} \\ 0 \end{bmatrix}, \qquad \lambda \in \mathbb{R} \cup \{\infty\} \tag{2.12}$$

2.1.4 Distorsioni delle lenti

Il modello finora trattato non è ancora sufficiente per descrivere con un margine di errore trascurabile il sistema ottico di una telecamera reale

poiché approssima l'obiettivo usato per la proiezione dell'immagine luminosa ad una lente di spessore infinitesimo.

Le imperfezioni delle lenti oppure l'esigenza di ampliare il campo visivo utilizzando una telecamera con distanza focale corta causano un'aberrazione, detta distorsione, che non permette una perfetta proiezione rettilinea.

La mappatura tra punti 3D e punti 2D della immagine può essere scomposta in una proiezione prospettica e una funzione che modella le differenze rispetto alla macchina stenopeica ideale.

La funzione di distorsione delle immagini può essere scomposta in due componenti. Nella prima componente, chiamata distorsione radiale, i punti dell'immagine sono distorti simmetricamente lungo direzioni radiali dal centro di distorsione. Questo tipo di distorsione è causato dalla forma imperfetta delle lenti e dipende soltanto dalla distanza dal centro del punto considerato $\rho = \sqrt{X_I^2 + Y_I^2}$.

La seconda è detta distorsione tangenziale o decentrante ed è causata da un assemblaggio improprio della lente. Questa dipende sia da ρ che dall'angolo da esso formato con l'asse delle ascisse $\theta = \arctan(\frac{Y_I}{X_I})$.

A questo punto si deve introdurre una funzione $f_D(X_I, Y_I, \rho, \theta)$ che ci permetta di passare dalle coordinate non distorte a quelle distorte e viceversa.

La distorsione può essere modellata come la somma del contributo radiale e quello tangenziale [Brown-66]:

$$\begin{pmatrix} X_D \\ Y_D \end{pmatrix} = \begin{pmatrix} X_{Dr} \\ Y_{Dr} \end{pmatrix} + \begin{pmatrix} X_{Dt} \\ Y_{Dt} \end{pmatrix} \tag{2.13}$$

I due addendi della (2.13) possono essere espressi come una serie di potenze di ρ :

$$\begin{pmatrix} X_{Dr} \\ Y_{Dr} \end{pmatrix} = \left(1 + K_{Dr1}\rho^2 + K_{Dr2}\rho^4 + K_{Dr3}\rho^6\right)\begin{pmatrix} X_I \\ Y_I \end{pmatrix} \tag{2.14}$$

$$\begin{pmatrix} X_{Dt} \\ Y_{Dt} \end{pmatrix} = \begin{bmatrix} 2K_{Dt1}X_I Y_I + K_{Dt2}(\rho^2 + 2X_I^2) \\ 2K_{Dt2}X_I Y_I + K_{Dt1}(\rho^2 + 2Y_I^2) \end{bmatrix} \tag{2.15}$$

Si noti che nelle precedenti equazioni è stato utilizzato il pedice "*Dr*" per indicare gli elementi associati al contributo della distorsione radiale e il pedice "*Dt*" per quelli relativi alla distorsione tangenziale.

La funzione f_D è dominata dalla componente radiale [Brown-71], della quale il coefficiente più influente è K_{Dr1} . Gli altri addendi sono via via sempre meno influenti, perciò un modello più complicato introduce elementi trascurabili oltre a portare instabilità numerica [Tsai-87].

2.2 Calibrazione

Per conoscere la matrice di proiezione prospettica $\overline{\overline{P}}$ è necessario valutare i parametri intrinseci che costituiscono la matrice M e quelli estrinseci relativi alla matrice G. I parametri intrinseci rappresentano la geometria interna della telecamera e le caratteristiche ottiche, mentre i parametri estrinseci si riferiscono a posizione e orientazione del sistema di riferimento telecamera rispetto al sistema di riferimento mondo.

Il processo di misurazione di tali parametri è detto calibrazione e si basa sul presupposto che si conoscano le proiezioni di alcuni punti 3D, detti **punti di calibrazione** (o punti di controllo, mire, punti fiduciali) le cui coordinate sono note.

Sono stati sviluppati diversi metodi di calibrazione la maggior parte dei quali utilizzano un oggetto (*target di calibrazione*) sul quale sono tracciati i punti di calibrazione costituiti da N elementi che nella immagine devono essere riconoscibili senza ambiguità ed avere coordinate note con accuratezza. Gli N elementi in genere sono quadrati disposti a scacchiera o dischi circolari, solitamente di colore nero su sfondo bianco.

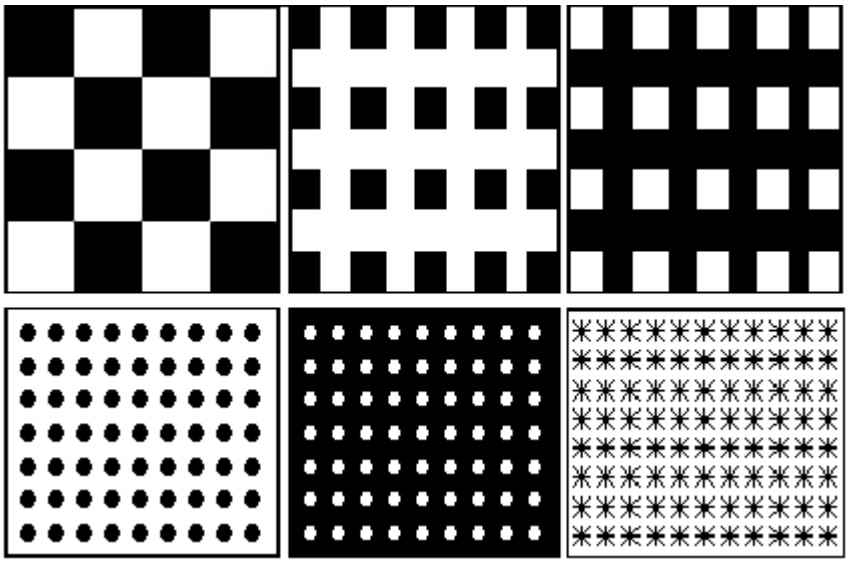


Illustrazione 16: Schemi di punti tracciati sui target di calibrazione

2.2.1 Metodi di calibrazione

Tra questi metodi uno dei più diffusi è quello di [Tsai-87] il quale assume che i parametri X_0, Y_0, α_x e α_y siano forniti dal costruttore della

telecamera o almeno possano essere desunti dai dati a disposizione. Esso richiede la conoscenza delle coordinate spaziali di n punti per immagine e risolve il problema di calibrazione con un insieme di n equazioni lineari basate sul vincolo di allineamento radiale.

Il metodo descritto in [Caprile-90] non necessita delle coordinate assolute di punti della scena, ma sfrutta le proprietà dei punti di fuga (*vanishing point*) di un target di calibrazione, avente almeno tre piani ortogonali tra loro. La calibrazione mediante vanishing point è meno utilizzata poiché sperimentalmente si è dimostrato non essere particolarmente accurata.

In [Faugeras-93] è descritta una procedura che consente di ottenere direttamente la matrice di proiezione prospettica, potendo ricavare in un secondo momento i parametri intrinseci ed estrinseci sulla base di $\overline{\Psi}$. Questo metodo che sfrutta la Direct Linear Trasform (DLT) necessita di almeno sei punti non complanari, ma per minimizzare l'errore di misura è necessario un numero maggiore di punti in modo da poter risolvere il sistema ai minimi quadrati.

Questi punti vengono prelevati su un oggetto costituito da almeno due piani ortogonali tra loro in modo che il sistema di riferimento mondo possa coincidere con i suoi spigoli. Questo accorgimento rende particolarmente agevole la determinazione delle coordinate dei punti di calibrazione che in genere vengono prese attraverso griglie spaziate in maniera uniforme, appositamente disegnate sulle facciate dell'oggetto.

2.2.2 Metodo di Zhang

Nonostante il metodo di Faugeras sia particolarmente accurato, in alcuni casi si preferisce il metodo illustrato in [Zhang-98] che risparmia la realizzazione dell'oggetto suddetto sostituendolo con un semplice oggetto planare. In particolare nel presente progetto è stato scelto questo metodo in cui l'oggetto planare consiste in una scacchiera i cui vertici delle caselle costituiscono i punti di calibrazione. Nel caso in questione i punti di calibrazione giacciono su un piano Π, perciò esiste una corrispondenza biunivoca tra questi e i punti sul piano immagine.

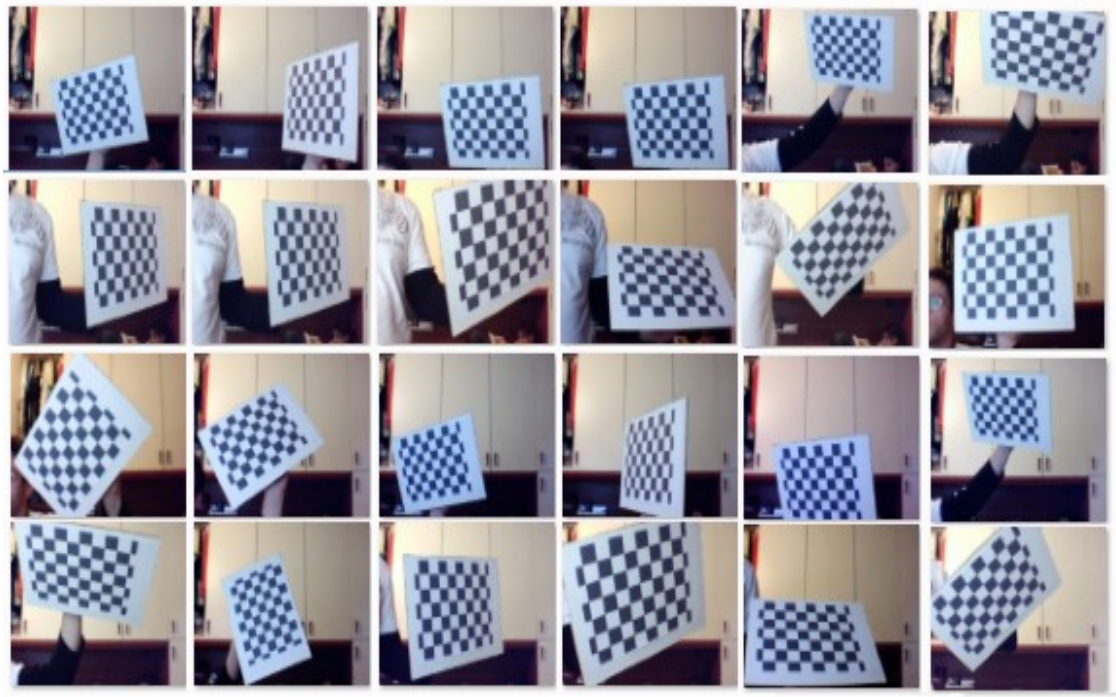


Illustrazione 17: Mosaico con varie orientazioni della scacchiera

Le corrispondenze, pertanto, sono codificate da una omografia che riduce i parametri incogniti ad 8, vale a dire una mappatura invertibile da P^2 a se

stesso, tale che tre punti giacciono sulla stessa linea se e solo se anche i loro punti mappati sono collineari.

2.2.3 Omografia tra l'oggetto planare e la sua immagine

Una mappatura $P^2 \rightarrow P^2$ è una proiettività se e solo se esiste una matrice 3×3 non singolare H tale che per ogni punto in P^2 è vero che il suo punto mappato vale $H\,x$ [Hartley-2003].

Infatti se si sceglie il sistema di riferimento mondo $O_s - X_s Y_s Z_s$ tale che gli assi X_S e Y_S siano giacenti sul piano, vale a dire in modo che l'equazione del piano sia $Z_S=0$, l'espressione (2.7) per un punto P^{Π} appartenente al piano diventa:

$$P_z^c \begin{pmatrix} X_I \\ Y_I \\ 1 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{pmatrix} P_x^s \\ P_y^s \\ 0 \\ 1 \end{pmatrix} \tag{2.16}$$

Pertanto il piano Π induce una omografia rappresentata da una matrice 3×3 la quale permette di scrivere la precedente relazione come:

$$\begin{pmatrix} X_I \\ Y_I \\ 1 \end{pmatrix} = \lambda \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{pmatrix} P_x^s \\ P_y^s \\ 1 \end{pmatrix} \quad \Rightarrow \quad p = \lambda H\, P^{\Pi} \tag{2.17}$$

La matrice H possiede 9 elementi, ma considerato lo scalare incognito λ, che fa variare la matrice senza alterare la trasformazione, i gradi di libertà si riducono a 8.

Il primo passo consiste nel trovare gli elementi della omografia H

2.2.3.1 Stima dell'omografia H

La prima domanda da porsi è quanti punti corrispondenti $\begin{pmatrix} X_I \\ Y_I \end{pmatrix} \leftrightarrow \begin{pmatrix} P_x^s \\ P_y^s \end{pmatrix}$ sono richiesti per calcolare la trasformazione proiettiva.

Nell'equazione (2.16) per le ipotesi fatte abbiamo posto $P_z^s = 0$ per cui possiamo semplificare anche l'espressione (2.8) ottenendo:

$$\begin{cases} X_I = \dfrac{p_{11} P_x^s + p_{12} P_y^s + p_{14}}{p_{31} P_x^s + p_{32} P_y^s + p_{34}} \\ Y_I = \dfrac{p_{21} P_x^s + p_{22} P_y^s + p_{24}}{p_{31} P_x^s + p_{32} P_y^s + p_{34}} \end{cases} \tag{2.18}$$

che po' essere riscritta come

$$\begin{aligned} \left(p_{31} P_x^s + p_{32} P_y^s + p_{34}\right) X_I - p_{11} P_x^s - p_{12} P_y^s - p_{14} = 0 \\ \left(p_{31} P_x^s + p_{32} P_y^s + p_{34}\right) Y_I - p_{21} P_x^s - p_{22} P_y^s - p_{24} = 0 \end{aligned} \tag{2.19}$$

Portando in forma matriciale si ha:

$$A_i \boldsymbol{h} = \begin{pmatrix} -P_x^s & -P_y^s & -1 & 0 & 0 & 0 & P_x^s X_I & P_y^s X_I & X_I \\ 0 & 0 & 0 & -P_x^s & -P_y^s & -1 & P_x^s Y_I & P_y^s Y_I & Y_I \end{pmatrix} \begin{pmatrix} p11 \\ p12 \\ p14 \\ p21 \\ p22 \\ p24 \\ p31 \\ p32 \\ p34 \end{pmatrix} = 0 \tag{2.20}$$

Ogni punto fornisce due equazioni corrispondenti alle sue componenti perciò impilando quattro matrici 2×9 si ottiene una matrice A di dimensione 8×9 il cui spazio nullo è la soluzione per **h**. Se non sono presenti triplette allineate, sono sufficienti quattro punti per vincolare H pienamente e calcolare una soluzione esatta.

2.2.3.2 Soluzione sovra-determinata

In genere i dati sono sporcati da rumore, perciò quattro punti costituiscono soltanto la dimensione del sottoinsieme più piccolo per ottenere una soluzione minima. La stima ai minimi quadrati è la tecnica migliore in presenza di rumore con distribuzione Gaussiana, ma tra i punti si possono trovare degli *outliers*, vale a dire un insieme di valori anomali e aberranti che possono essere dovuti a errori di localizzazione o a falsi accoppiamenti e che non possono essere trattati come rumore Gaussiano dal momento che anche solo uno di essi può essere in grado di alterare drasticamente la stima del modello. Gli errori di localizzazione sono previsti nel comportamento gaussiano e per la maggior parte dei punti di interesse sono di piccole dimensioni (uno o due pixel). Alcuni punti però a causa di rumore impulsivo, si collocano nella coda della gaussiana, ed essendo lontani dalla tendenza generale dei dati, riducono notevolmente la precisione della stima. Per fronteggiare la presenza di campioni periferici che danneggiano la precisione di H, sono pertanto necessari metodi di stima robusti che adattano un modello alla maggioranza dei dati e in seguito individuano come *outliers* quei dati che presentano un residuo elevato.

2.2.3.3 Scelta della funzione di costo

Con un numero maggiore di corrispondenze non si può ottenere una soluzione esatta che non sia il vettore nullo. In questo caso è necessario imporre un'appropriata funzione di costo da minimizzare in modo che la

soluzione per il vettore **h** sia unica e non nulla.

La funzione di costo più semplice è quella che minimizza la norma $\|A\boldsymbol{h}\|$ ovvero quella che rende minimo il vettore di errore algebrico. Per essere certi che la soluzione non sia il vettore nullo, si aggiunge il vincolo che la norma di **h** sia unitaria. La soluzione a questo problema può essere trovata con la DLT e la decomposizione ai valori singolari (SVD) preceduti però da un processo di normalizzazione [Hartley-2003] descritto in maggiore dettaglio nel capitolo 3. Sebbene questa funzione di costo sia lineare e computazionalmente poco onerosa ha lo svantaggio che la quantità minimizzata non ha significativo geometrico né statistico.

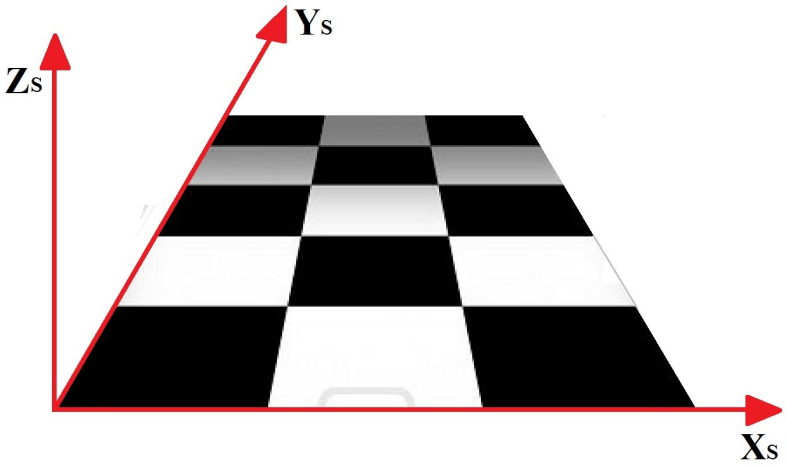


Illustrazione 18: Sistema di riferimento mondo sul modello di calibrazione

Come detto in precedenza abbiamo posto il sistema di riferimento mondo con l'asse Z_S giacente sul piano rappresentato dal modello di calibrazione. Se tale modello è una scacchiera risulta agevole porre gli altri due assi coincidenti con i due bordi più esterni della scacchiera come

nell'illustrazione 18[1]. Se si conosce la dimensione di una casella in questo sistema di riferimento si possono determinare immediatamente le coordinate di tutti i vertici della scacchiera. Una funzione di costo quindi si può basare sulla minimizzazione della distanza euclidea tra i punti dei corner nell'immagine p_j e quelli generati dalla stima mediante il prodotto di H per le coordinate della scacchiera nel world reference frame P_j^{Π} . Possiamo quindi scrivere:

$$\min_{H_i} \boldsymbol{r}_j = \min_{H_i} \sum_j \| p_j - H_i P_j^{\Pi} \|, \quad j = 1,\ldots,\mu \tag{2.21}$$

in cui μ è il numero di corner rilevati.

Questo errore geometrico può essere minimizzato ai minimi quadrati, se si prevede che l'errore non sia perfettamente gaussiano e quindi siano presenti degli outlier il risultato ottenuto può essere raffinato utilizzando metodi robusti tra i quali i più utilizzati sono il metodo LmedS e il RANSAC.

2.2.3.4 Least median of Squares

Il metodo della minima mediana dei quadrati (LmedS) [Rousseeuw-87] è una delle tecniche di stima robusta più utilizzate. Questa tecnica stima i parametri attraverso il problema di ottimizzazione non lineare:

$$\min_{\boldsymbol{H}_i} \left\{ \underset{j}{med}\, r_j^2 \right\} \tag{2.22}$$

cioè si cercano i valori che minimizzano la mediana del quadrato dei residui, e non la loro somma come si fa con il metodo dei minimi quadrati.

1 In realtà, come si vedrà in seguito, gli angoli non sono rilevati a partire dal margine più esterno, ma dalla prima fila in cui si distinguono le intersezioni lungo le due direzioni e quindi anche il sistema di riferimento risulta più spostato verso l'interno.

Rousseeuw congettura che non sia possibile formulare una soluzione in forma chiusa per la (2.22), né impiegare algoritmi iterativi, ma deve essere risolto attraverso una ricerca su un campione casuale (*random resampling*) nello spazio delle stime possibili generate dai dati, esplorando m volte tale insieme in maniera casuale. Se q è il minimo numero di dati necessario a determinare un vettore di parametri e ϵ è la percentuale di outlier, la probabilità di scegliere almeno un insieme privo di outlier è

$$P = 1 - \left(1 - (1-\epsilon)^q\right)^m \tag{2.23}$$

Ponendo $P = 0.99$ e risolvendo rispetto a m , si determina quanto deve valere m per ottenere una stima corretta con buona approssimazione:

$$m = \frac{\log(1-P)}{\log\left(1-(1-\epsilon)^q\right)} \tag{2.24}$$

In genere l'algoritmo non è molto efficiente in presenza di rumore Gaussiano, perciò la soluzione viene raffinata attraverso una minimizzazione ai minimi quadrati sui residui pesati

$$min \sum_i w_i r_i^2$$

con

$$w_i = \begin{cases} 1 & se\ |r_i| \le 3.7065\left(1 + \frac{5}{m-s}\right)\sqrt{\underset{i}{med}\, r_i^2} \\ 0 & altrimenti \end{cases}$$

I dati a cui viene associato 0 sono considerati outliers.

2.2.3.5 RANSAC

L'algoritmo RANSAC (*RANdom SAmple Consensus*) adotta la stessa

tecnica di campionamento random del metodo LmedS. In questo caso però sul sottoinsieme di q elementi si determina quanti dati hanno uno scostamento dalla stima inferiore ad una certa soglia τ . In una distribuzione Gaussiana con media zero e deviazione standard σ , per avere una probabilità del 95% che un punto dell'insieme sia un inlier si deve avere $\tau = 1.96\sigma$.

Successivamente si determina l'insieme di consenso C maggiore, vale a dire quello la cui cardinalità delle osservazioni che concordano con la stima è più numerosa.

La percentuale ϵ di outlier che si prevede nell'insieme di osservazioni moltiplicata per il numero complessivo di dati ci fornisce una stima per il valore atteso per il consenso C .

Se il consenso è inferiore a tale valore vengono ripetuti i primi passi per un altro sottoinsieme casuale, finché non viene trovato un insieme di consenso superiore sul quale viene effettuata la regressione.

In caso di conoscenza a priori τ e C sono determinate, ma nei casi sperimentali difficilmente sono noti i valori di ϵ e σ , perciò si deve procedere per tentativi.

Nel caso in cui l'algoritmo non termina dopo un certo numero di iterazioni significa che potrei aver scelto un valore di soglia per C troppo elevato e in tal caso si può decidere di far fallire l'algoritmo oppure aggiornare il numero di tentativi m durante l'esecuzione in base alla (2.24) ed effettuare la regressione sull'insieme di consenso maggiore.

2.2.4 Stima della matrice dei parametri intrinseci

Considerando che $\overline{\Psi} = M[R \quad T]$ ed $R = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix}$, si ha:

$$H = \lambda M \begin{bmatrix} r_1 & r_2 & T \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} \tag{2.25}$$

Da cui si ottiene:

$$\begin{aligned} r_1 &= \lambda M^{-1} h_1 \\ r_2 &= \lambda M^{-1} h_2 \end{aligned} \tag{2.26}$$

Grazie al fatto che le colonne di R sono ortonormali, ovvero che $r_1^T r_2 = 0$ e $r_1^T r_1 = r_2^T r_2$ si ha

$$\begin{aligned} h_1^T M^{-T} M^{-1} h_2 &= 0 \\ h_1^T M^{-T} M^{-1} h_1 &= h_2^T M^{-T} M^{-1} h_2 \end{aligned} \tag{2.27}$$

Zhang dimostra che, indicando come $h_i = \begin{bmatrix} h_{i1} \\ h_{i2} \\ h_{i3} \end{bmatrix}$ la i-esima colonna di H

e con B la matrice simmetrica

$$B = M^{-T} M^{-1} = \begin{bmatrix} \frac{1}{(\alpha_x f)^2} & 0 & \frac{-X_0}{(\alpha_x f)^2} \\ * & \frac{1}{(\alpha_y f)^2} & \frac{-Y_0}{(\alpha_y f)^2} \\ * & * & 1 + \frac{X_0^2}{(\alpha_x f)^2} + \frac{Y_0^2}{(\alpha_y f)^2} \end{bmatrix} \tag{2.28}$$

è possibile effettuare la decomposizione

$$h_i^T M^{-T} M^{-1} h_j = v_{ij}^T b \tag{2.29}$$

posto

$$b=\begin{bmatrix} B_{11} \\ B_{12} \\ B_{22} \\ B_{13} \\ B_{23} \\ B_{33} \end{bmatrix} \quad \text{e} \quad v_{ij}=\begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2}+h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1}+h_{i1}h_{j3} \\ h_{i3}h_{j2}+h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix} \tag{2.30}$$

avendo indicato con B_{ij} il generico elemento di B.

di cui soltanto sei dei nove elementi non sono duplicati.

Le (2.27) possono pertanto essere riscritte come

$$\begin{bmatrix} v_{12}^T \\ (v_{11}-v_{22})^T \end{bmatrix} b = V\,b = 0 \tag{2.31}$$

dove V è una matrice 2×6 .

Se sono osservate n immagini del piano, impilando le equazioni ottenute, V diventa $2n\times 6$ e per $n\geq 3$ si ottiene un'unica soluzione definita a meno di un fattore di scala. Tenendo in considerazione che la (2.28) è nota a meno di un fattore di scala λ , cioè $B=\lambda M^{-T}M^{-1}$; una volta noti gli elementi di B si possono ricavare i parametri intrinseci:

$$\begin{aligned} X_0 &= \frac{-B_{13}}{B_{11}} \\ Y_0 &= \frac{-B_{23}}{B_{22}} \\ \lambda &= B_{33}+X_0 B_{13}+B_{23}Y_0 \\ \alpha_x f &= \sqrt{\frac{\lambda}{B_{11}}} \\ \alpha_y f &= \sqrt{\frac{\lambda}{B_{22}}} \end{aligned} \tag{2.32}$$

2.2.5 Stima dei parametri estrinseci

Per quanto riguarda i parametri estrinseci, nota M possiamo ricavare gli elementi della matrice $\overline{\Psi} = M[R \quad T]$ sfruttando le relazioni:

$$\begin{aligned} r_1 &= \lambda M^{-1} h_1 \\ r_2 &= \lambda M^{-1} h_2 \\ r_3 &= r_1 \times r_2 \\ t &= \lambda M^{-1} h_3 \end{aligned} \tag{2.33}$$

In cui il fattore di scala è determinato dalla condizione di ortonormalità

$$\lambda = \frac{1}{\left\| M^{-1} h_1 \right\|} = \frac{1}{\left\| M^{-1} h_2 \right\|} \quad .$$

Successivamente in genere è opportuno applicare una decomposizione ai valori singolari della matrice $R = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix}$ ottenuta, per aggirare il problema che, a causa del rumore, questa non soddisfa le proprietà di una matrice di rotazione.

Capitolo 3
Visione stereoscopica

L'uomo e molte altre specie animali sono dotati di visione binoculare, ovvero l'unica percezione visiva della realtà è il risultato del contributo dei due occhi che da posizioni differenti, attraverso il cristallino e la cornea, mettono a fuoco la luce catturata dall'esterno e mediante il nervo ottico inviano impulsi elettrici al cervello per l'elaborazione e l'interpretazione.
Poiché i due occhi hanno una certa distanza tra loro (per l'uomo circa 6 cm) le due viste presentano una certa disparità. La *stereopsi* è la capacità del cervello di unire le due immagini sfruttando tale disparità per assegnare un senso di maggiore o minore profondità agli oggetti dello spazio visivo che consente la visione tridimensionale.

3.1 Triangolazione

La possibilità di individuare la distanza a cui si trova un oggetto è di fondamentale importanza non solo nel mondo animale, ma anche in ambito tecnologico. Problemi tipici che possono avere tale necessità sono l'Object recognition, la ricostruzione di modelli 3D di una scena e la manipolazione di robot mediante asservimento visuale (visual servoing).
Il caso più semplice si ha in presenza di un impianto stereo canonico, vale a dire quando si hanno immagini corrispondenti senza distorsioni, perfettamente complanari e allineate per riga di pixel, con assi principali

perfettamente paralleli, lunghezze focali coincidenti e i punti principali c_L e c_R sono stati calibrati in modo da trovarsi alle stesse coordinate in pixel nelle due immagini. Tale disposizione è detta frontale-parallela ed è sintetizzata con una vista aerea nell'illustrazione 19.

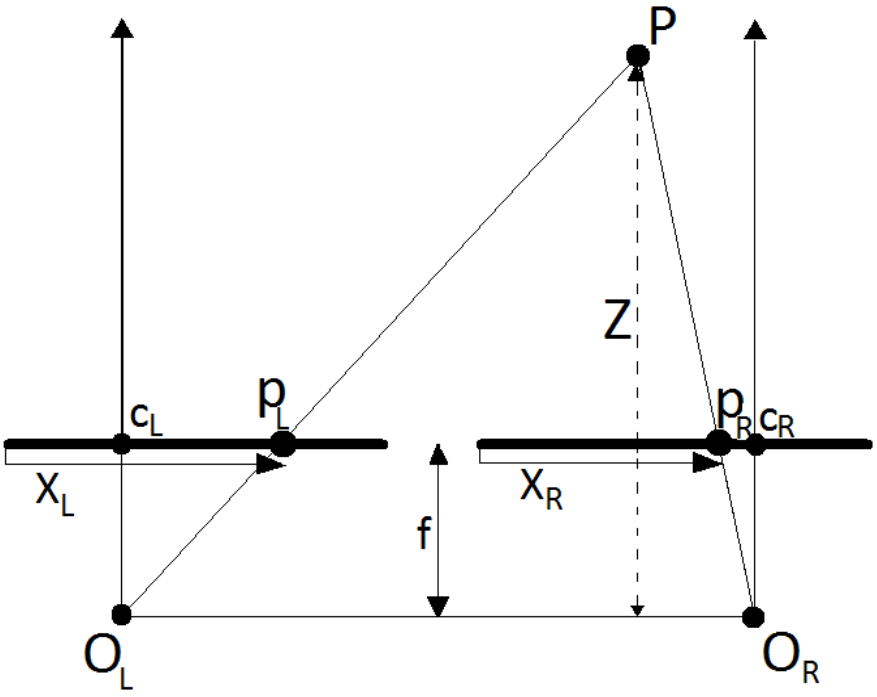


Illustrazione 19: Vista aerea di un impianto stereo canonico

La retta che congiunge i centri ottici O_L e O_R delle due telecamere è detta linea di base o baseline. Con il termine baseline si può indicare anche il segmento $\overline{O_L O_R}$ nel qual caso si può anche parlare di lunghezza della baseline.

Per ogni punto P nello spazio, individuato da entrambe le telecamere, la disparità sarà data dalla differenza tra le ascisse X_L e X_R dei sistemi di

coordinate sul piano immagine che hanno origine in alto a sinistra nelle immagini in pixel come mostrato nell'illustrazione 20.

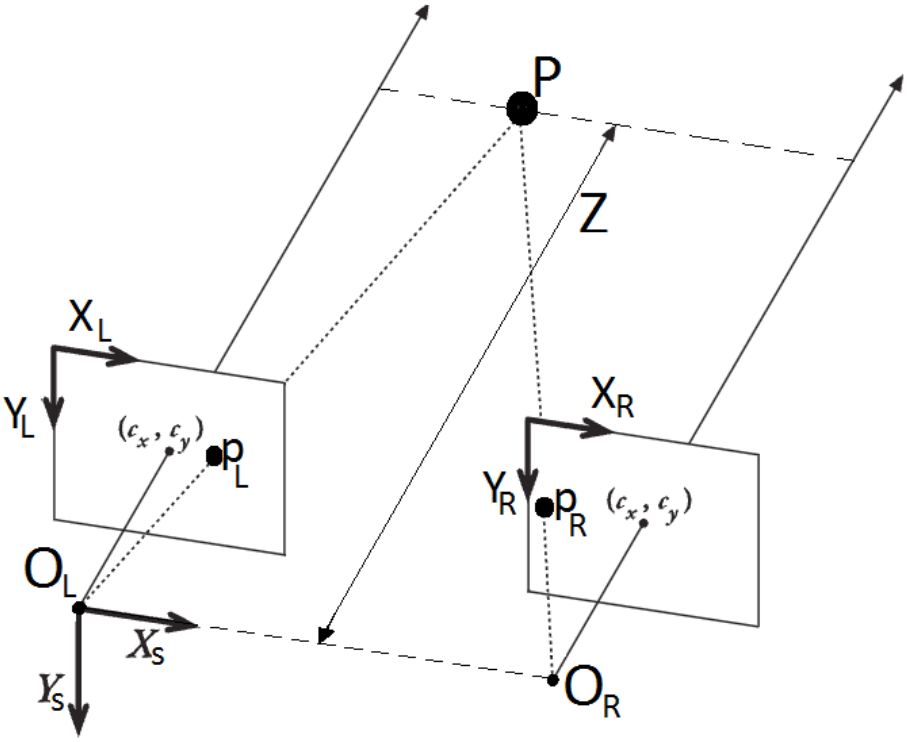


Illustrazione 20: Assonometria di un impianto stereo in disposizione frontale-parallela

Consideriamo ora i due triangoli simili aventi come vertice in comune il punto dello spazio P. Il lato opposto al vertice di uno dei due triangoli è rappresentato dal segmento $\overline{p_L p_R}$, mentre per l'altro triangolo è la baseline $\overline{O_L O_R}$. Dall'analisi dei due triangoli si desume che la profondità Z è inversamente proporzionale alla disparità $X_L - X_R$ tra le due viste:

$$\frac{\overline{O_L O_R} - (X_L - X_R)}{Z - f} = \frac{\overline{O_L O_R}}{Z} \quad \Rightarrow \quad Z = \frac{f \cdot \overline{O_L O_R}}{X_L - X_R} \tag{3.1}$$

Quando la distanza Z è piccola si ha una disparità molto grande e quindi

una risoluzione rispetto alla profondità elevata, vale il contrario per distanze grandi.

Nel mondo reale le telecamere non saranno mai esattamente allineate nella configurazione frontale-parallela, perciò è necessario capire come si può mappare coppie di immagini prelevate dal mondo reale in coppie che rispettino la geometria ideale.

3.2 Geometria epipolare

Per ottenere informazioni sulla profondità da una coppia di immagini provenienti da telecamere è necessaria una stereopsi computazionale che individui i punti corrispondenti nelle due immagini ovvero che sono proiezione di uno stesso punto 3D (problema della corrispondenza o di matching). Questi punti, detti *punti coniugati* [Fusiello-2010], possono essere computazionalmente onerosi da calcolare se ci si basa solo su vincoli di similarità. Per ogni punto di una immagine sarebbe necessario verificare la similarità su tutte le righe e tutte le colonne dell'altra immagine generando inoltre numerosi falsi accoppiamenti. Fortunatamente il punto coniugato sull'altra immagine può trovarsi soltanto lungo una retta, detta retta epipolare. Tale vincolo deriva dallo studio della geometria dell'intersezione dei piani immagine con il fascio di piani passanti per la *baseline.*

Questa geometria, chiamata geometria epipolare, dipende solo dai parametri intrinseci ed estrinseci delle due telecamere e non dalla geometria tridimensionale della scena.

Una qualsiasi retta nel piano ha come equazione in forma implicita $ax+by+c=0$, quindi una generica retta epipolare sull'immagine può essere rappresentata dal vettore: $l_i=\begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix}$.

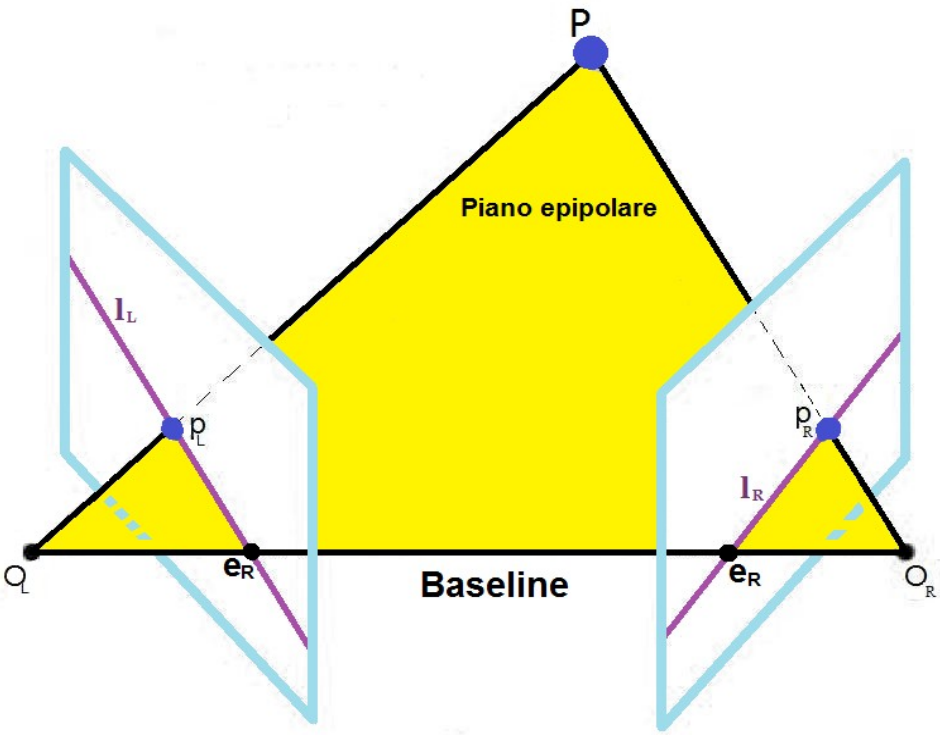


Illustrazione 21: Intersezione dei piani immagine con il piano epipolare

Consideriamo tra tutti i piani passanti per la baseline quello passante per il punto P, il piano epipolare. L'intersezione del piano epipolare con i due piani immagine genera due rette l_L e l_R dette linee epipolari passanti per le proiezioni p_L e p_R di P sui due piani immagine. Da ciò si deduce che il punto corrispondente di uno dei due punti dovrà giacere sulla linea

epipolare corrispondente sull'altro piano immagine.

Da questa geometria si individuano anche i punti e_L ed e_R derivanti dalla intersezione della baseline con i piani immagine detti *epipoli*.

3.2.1 La matrice fondamentale

Un epipolo corrisponde anche alla proiezione sul piano immagine del centro ottico dell'altra telecamera, perciò, espressa con $\overline{\Psi}_L$ la matrice di proiezione prospettica riferita al piano immagine sinistro, si ha:

$$e_R=\overline{\Psi}_R\begin{bmatrix}O_L\\1\end{bmatrix}=M_R\begin{bmatrix}R_R & T_R\end{bmatrix}\begin{bmatrix}O_L\\1\end{bmatrix}=M_R R_R O_L+M_R T_R \qquad (3.2)$$

Dalla (11) si ricava che $O_L=-R_L^{-1}T_L$ quindi

$$e_R=\overline{\Psi}_R\begin{bmatrix}O_L\\1\end{bmatrix}=M_R T_R-M_R R_R R_L^{-1}T_L \qquad (3.3)$$

È semplice notare che la retta epipolare è la proiezione di un raggio ottico sull'altro piano immagine per cui dalla (3.4)

$$\overline{\Psi}_R\begin{bmatrix}(M_L R_L)^{-1}p_L\\0\end{bmatrix}=M_R R_R(M_L R_L)^{-1}p_L=\overline{\Psi}_R\overline{\Psi}_L^{+}p_L \qquad (3.5)$$

da cui si ottiene l'equazione parametrica:

$$l_R:\ \ p_R=e_R+\lambda\,\overline{\Psi}_R\overline{\Psi}_L^{+}p_L \qquad (3.6)$$

oppure in forma di prodotto vettoriale

$$l_R=\left(\overline{\Psi}_R C\right)\times\left(\overline{\Psi}_R\overline{\Psi}_L^{+}p_L\right)=e_R\times\left(M_R R_R(M_L R_L)^{-1}p_L\right) \qquad (3.7)$$

La (3.7) può essere riscritta, con l'ausilio della matrice antisimmetrica, come

$$l_R=S(e_R)\overline{\Psi}_R\overline{\Psi}_L^{+}p_L=F\,p_L \qquad (3.8)$$

avendo definito la **matrice fondamentale** 3×3 di rango 2

$$F = S(e_R)\overline{\Psi}_R\overline{\Psi}_L^{+} = S(e_R)M_R R_R (M_L R_L)^{-1} \tag{3.9}$$

La matrice $A = \overline{\Psi}_R\overline{\Psi}_L^{+}$ rappresenta una omografia che mappa i punti da un piano immagine all'altro [Hartley-2003].

3.2.2 Le matrici $\overline{\Psi}_L$ e $\overline{\Psi}_R$ in un sistema stereoscopico

Se nell'impianto stereo canonico dell'illustrazione 20 poniamo il sistema di riferimento spaziale in modo da far coincidere l'origine O_S con il centro di proiezione della telecamera sinistra O_L otteniamo una semplificazione dell'equazione (2.5). La matrice dei parametri estrinseci relativi alla telecamera sinistra G_L è data quindi dalla matrice identità per cui si ottiene $\overline{\Psi}_L = M_L[I \quad \mathbf{0}]$. Nella configurazione mostrata nell'illustrazione 21 per passare da un sistema di riferimento sinistro al destro si deve fare ancora ricorso alla (2.5). L'origine della telecamera destra si trova ad una distanza $T = \overline{O_L O_R}$ per cui si ha $\overline{\Psi}_R = M_R[R \quad T]$. Da quanto affermato si ricava che

$$\overline{\Psi}_L^{+} = \begin{bmatrix} M_L^{-1} \\ \mathbf{0} \end{bmatrix} \quad \text{e} \quad C_L = \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} \tag{3.10}$$

La proprietà (3.2) diventa con l'attuale impostazione delle coordinate e sfruttando la proprietà $R^T = R^{-1}$:

$$\begin{aligned} e_L &= \overline{\Psi}_L C_R = M_L[I \quad \mathbf{0}]\begin{bmatrix} -R^T T \\ 1 \end{bmatrix} = M_L R T \\ e_R &= \overline{\Psi}_R C_L = M_R[R \quad T]\begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} = M_R T \end{aligned} \tag{3.11}$$

3.2.3 Proprietà della matrice fondamentale

Premoltiplicando la (3.6) a sinistra e a destra per $p_R^T S(e_R)$ si ottiene:

$$p_R^T S(e_R)\, p_R = \lambda\, p_R^T S(e_R)\, A\, p_L + p_R^T S(e_R)\, e_R$$

Poiché il prodotto vettoriale di vettori linearmente dipendenti è nullo $S(e_R)e_R=0$ ed è nullo anche il prodotto scalare di vettori ortogonali $p_R^T S(e_R)\, p_R=0$ si ricava l'equazione di Longuet-Higgins:

$$p_R^T F\, p_L = 0 \tag{3.12}$$

Con questa equazione è possibile caratterizzare la matrice fondamentale solo in termini di punti corrispondenti senza conoscere la matrice $\overline{\Psi}$.

Le più importanti proprietà della matrice fondamentale [Hartley-2003] sono:

1. Se F è la matrice fondamentale di una coppia di telecamere, la matrice F^T lo è della coppia nell'ordine opposto; quindi dalla (3.9) si ha

$$F = S(e_R)A = A^{-T} S(e_L) \tag{3.13}$$

2. La (3.8) fornisce la linea epipolare sull'immagine sinistra. La linea epipolare sull'altra immagine sarà $l_L = F^T p_R$

3. Ad ogni punto p_R e p_L nel piano immagine destro corrispondono rispettivamente la linea epipolare sinistra contenente l'epipolo e_L e la linea epipolare destra contenente l'epipolo e_R . Per questo si ha $e_R^T F\, p_L = 0$ e dalla prima proprietà $e_L^T F^T p_R = 0$. Questo ci conduce alla seguente proprietà:

$$e_R^T F = 0 \Rightarrow F^T e_R = 0$$
$$e_L^T F^T = 0 \Rightarrow F e_L = 0 \quad (3.14)$$

4. F ha sette gradi di libertà. I due gradi di libertà rimossi dipendono dal solito fattore di scala e dal vincolo $det\, F = 0$. La matrice F infatti non avendo rango pieno non è invertibile. Questo può essere spiegato sottolineando il fatto che la matrice fondamentale mappa, attraverso la relazione (3.7), un punto sul primo piano immagine in una linea nel secondo, perciò non è possibile ricostruire il punto corrispondente sull'altro piano immagine soltanto dalla conoscenza di F e della linea epipolare.

Possiamo dimostrare la prima uguaglianza della proprietà (3.13) sostituendo le (3.10) e (3.11) nella (3.9) ottenendo

$$F = S(\overline{Y}_R C_L) M_R [R \quad T] \begin{bmatrix} M_L^{-1} \\ \mathbf{0} \end{bmatrix} = S(M_R T) M_R R M_L^{-1} = S(e_R) M_R R M_L^{-1} \quad (3.15)$$

Grazie alla proprietà delle matrici antisimmetriche per cui

$$S(a)M = M^{-T} S(M^{-1} a) \quad (3.16)$$

si ricava la seconda uguaglianza nella proprietà (3.13)

$$F = \left(M_R R M_L^{-1}\right)^{-T} S\left((M_R R M_L^{-1})^{-1} M_R T\right) = M_R^{-T} R M_L^T \; S\left(M_L R T\right) = M_R^{-T} R M_L^T S(e_L)$$

3.2.4 La matrice Essenziale

Sfruttando ancora la (3.16) nella (3.15) possiamo ricavare un'altra forma per F:

$$F = M_R^{-T} \; S\left((M_R^{-1} e_R)\right) R M_L^{-1} = M_R^{-T} \; S\left((M_R^{-1} M_R T)\right) R M_L^{-1} \quad (3.17)$$

Andando a definire la matrice Essenziale

$$E = S(T)R \tag{3.18}$$

l'equazione (3.17) può essere ridotta in

$$F = M_R^{-T} E M_L^{-1} \tag{3.19}$$

Dalla (2.9) è semplice dedurre che

$$p_R^T M_R^{-T} E M_L^{-1} p_L = \left(M_R^{-1} p_R\right)^T E M_L^{-1} p_L = 0 \tag{3.20}$$

la quale introdotte le coordinate normalizzate $\tilde{p} = M^{-1} p$ assume la forma

$$\tilde{p}_R{}^T E \tilde{p}_L = 0 \tag{3.21}$$

3.2.5 La matrice F in un sistema stereoscopico canonico

In un impianto stereo canonico, come quello nelle illustrazioni 19 e 20, i piani focali contengono la baseline $\overline{O_L O_R}$ e sono complanari, gli epipoli giacciono all'infinito e le linee epipolari sono parallele, orizzontali e presentano la proprietà di collinearità, cioè la retta epipolare destra risulta essere la prosecuzione di quella sinistra. La collinearità comporta che un punto abbia il suo coniugato sulla stessa *scanline*, ovvero sulla stessa riga di pixel, favorendo il calcolo delle corrispondenze.

In questo caso la matrice fondamentale assume la forma [Cyganek-09]

$$F_C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & c \\ 0 & -c & 0 \end{bmatrix} \tag{3.22}$$

dove c è un valore costante diverso da 0. Sostituendo F_C nella (3.12) otteniamo

$$p_R^T F_C p_L = \begin{bmatrix} X_R & Y_R & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & c \\ 0 & -c & 0 \end{bmatrix} \begin{bmatrix} X_L \\ Y_L \\ 1 \end{bmatrix} = 0$$

che equivale a $Y_R = Y_L$, ovvero conferma che in un impianto stereo canonico due punti corrispondenti si trovano sulla medesima ordinata, mentre hanno differente ascissa che genera la disparità necessaria alla triangolazione.

Dalla (3.8) si ricava anche la formula che ribadisce l'orizzontalità della linea epipolare in una disposizione frontale-parallela:

$$l_R = F_C\, p_L = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & c \\ 0 & -c & 0 \end{bmatrix} \begin{bmatrix} X_L \\ Y_L \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ c \\ -c\,Y_L \end{bmatrix}$$

3.3 Calcolo della matrice fondamentale

La matrice F dipende dai parametri intrinseci ed estrinseci della telecamera, questo si può facilmente constatare osservando la relazione (3.13). Il calcolo di F è possibile anche senza tali informazioni, è sufficiente infatti un sufficiente numero di punti coniugati. La matrice fondamentale che si ricava contiene tutta l'informazione sulla geometria del sistema, ma se F è calcolata da un sistema stereoscopico non è possibile estrarre Ψ e G, bensì è necessario avere almeno una terna di immagini prese da posizioni diverse mediante la stessa telecamera (per esempio nel caso di telecamera in movimento). Questo problema è detto auto-calibrazione, ma non verrà trattato.

La determinazione della matrice fondamentale in genere serve per ottenere una nuova configurazione del sistema di visione stereoscopica in modo da poter considerare le immagini come acquisite da un impianto stereo

canonico. Per poter sfruttare anche nel caso generale le proprietà di quest'ultimo si ricorre alla rettificazione epipolare, cioè una trasformazione che lascia inalterata la baseline e attraverso una rotazione attorno ai centri ottici rende i piani focali paralleli tra loro e alla baseline.

La matrice fondamentale ha rango pari a 2 ed è definita a meno di un fattore di scala, perciò avendo nove elementi ha 7 gradi di libertà.

Il vincolo epipolare $p_R^T F\, p_L = 0$ valido per ogni coppia di punti corrispondenti può essere usato per stimare la matrice fondamentale qualora si conoscano un numero sufficiente di punti. Questa idea fu usata inizialmente da Longuet-Higgins per trovare i coefficienti della matrice Essenziale [LonguetHiggins-81].

Ogni corrispondenza fornisce la seguente equazione

$$\begin{pmatrix} X_R & Y_R & 1 \end{pmatrix} \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{pmatrix} X_L \\ Y_L \\ 1 \end{pmatrix} = 0 \qquad (3.23)$$

Denotando con $\boldsymbol{f}$ il vettore costituito dagli elementi di F ordinati per riga possiamo esprimere questa equazione come

$$\begin{pmatrix} X_R X_L, X_R Y_L, X_R, Y_R X_L, Y_R Y_L, Y_R, X_L, Y_L, 1 \end{pmatrix} \boldsymbol{f} = 0$$

Da un insieme di n coppie di punti si ottengono un insieme di equazioni lineari nei coefficienti della matrice fondamentale F che definiscono un sistema lineare omogeneo $Q\boldsymbol{f} = 0$, con

$$Q = \begin{bmatrix} X_{R1}X_{L1} & X_{R1}Y_{L1} & X_{R1} & Y_{R1}X_{L1} & Y_{R1}Y_{L1} & Y_{R1} & X_{L1} & Y_{L1} & 1 \\ X_{R2}X_{L2} & X_{R2}Y_{L2} & X_{R2} & Y_{R2}X_{L2} & Y_{R2}Y_{L2} & Y_{R2} & X_{L2} & Y_{L2} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_{Rn}X_{\ln} & X_{Rn}Y_{\ln} & X_{Rn} & Y_{Rn}X_{\ln} & Y_{Rn}Y_{\ln} & Y_{Rn} & X_{\ln} & Y_{\ln} & 1 \end{bmatrix}$$

la cui soluzione è il nucleo di Q.

Analogamente a quanto visto nel capitolo 2, numerosi algoritmi permettono di calcolare la soluzione cercata.

Conoscere il funzionamento di tali metodi consente la scelta della tecnica più opportuna in base al numero di corrispondenze richieste, alla complessità computazionale e alla possibilità che siano presenti valori anomali.

3.3.1 Decomposizione ai valori singolari

Gli algoritmi che prevedono la conoscenza del minor numero di corrispondenze sono l'algoritmo dei sette punti e quello degli otto punti. Questi algoritmi si basano su una applicazione comune per la risoluzione di sistemi di equazioni sovradeterminati chiamata Singular Value Decomposition (SVD).

La matrice Q viene decomposta come $Q = U D V^T$, dove U e V sono matrici ortogonali e D è una matrice diagonale i cui elementi diagonali sono detti valori singolari e corrispondono alle radici quadrate degli autovalori della matrice $Q^T Q$. Essendo $Q^T Q$ simmetrica e semidefinita positiva allora i valori singolari sono reali e non negativi.

Le colonne di V corrispondenti ai valori singolari nulli sono una base per $Ker(Q)$ perciò possono fornire una soluzione per $Q\boldsymbol{f} = 0$.

3.3.2 Soluzione esatta con sette punti

I sette gradi di libertà di F comportano un numero minimo di coppie per

avere una soluzione riguardo alla geometria epipolare pari a 7.

La matrice Q con $n=7$ ha un nucleo di dimensione 2 dato dalla combinazione lineare $\alpha F_1+(1-\alpha)F_2$, in cui le matrici F_1 e F_2 sono ottenute dai vettori corrispondenti $\boldsymbol{f}_1$ e $\boldsymbol{f}_2$, che si possono ottenere attraverso la SVD. La matrice fondamentale deve avere determinante nullo perciò si ha $det\left(\alpha F_1+(1-\alpha)F_2\right)=0$ da cui si ottiene un polinomio cubico in α che ha al massimo 3 soluzioni reali [Zhang-96].

Questo algoritmo non genera sempre un unica soluzione, la quale comunque non fornisce valori attendibili per il basso numero di punti usati. Comunque questo algoritmo viene ripreso nei metodi robusti come passo di algoritmi iterativi.

3.3.3 Metodo degli otto punti

In assenza di rumore, con n pari a otto, il nucleo di Q è uno spazio vettoriale di dimensione 1, poiché i parametri di $\boldsymbol{f}$ sono definiti a meno di una costante moltiplicativa. La soluzione può essere calcolata con il cosiddetto algoritmo degli 8 punti, variante del DLT, il quale però può risultare instabile a causa del rumore, degli errori dovuti ad arrotondamenti numerici, ma soprattutto a causa del malcondizionamento dovuto all'utilizzo di coordinate pixel omogenee che rendono la terza dimensione molto diversa dalle altre due [Hartley-97].

3.3.4 Normalizzazione

Gli elementi della matrice Q in cui compaiono le singole coordinate pixel

sono dell'ordine di 10^2 , mentre si ha un ordine di 10^4 per quelli in cui sono presenti i prodotti delle coordinate e valori unitari per l'ultima colonna. Per ridurre tale sproporzione viene applicata una trasformazione che rende molto più stabile l'algoritmo in modo che i risultati possano essere confrontati con gli algoritmi iterativi che saranno analizzati in seguito. Tale trasformazione, chiamata normalizzazione da Hartley, è una trasformazione affine che deve essere applicata indipendentemente alle due immagini e consiste in due passi principali. Il primo consiste in una traslazione tale da far coincidere l'origine $\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$ con il centroide dell'insieme di punti. Il secondo passo è quello di scalare gli stessi punti affinché la distanza media dall'origine sia pari a $\sqrt{2}$ in modo che il punto medio sia pari a $\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T$.

Data una matrice di normalizzazione N si avrà [Cyganek-09]:

$$\breve{p} = N\,P = \begin{bmatrix} s_x & 0 & -X_m s_x \\ 0 & s_y & -Y_m s_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix} = \begin{bmatrix} s_x\left(X_i - X_m\right) \\ s_y\left(Y_i - Y_m\right) \\ 1 \end{bmatrix} \tag{3.24}$$

in cui compaiono le coordinate omogenee del punto medio calcolato rispetto ai punti non ancora normalizzati

$$p_m = \begin{bmatrix} X_m \\ Y_m \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{n}\sum_{i=1}^{n} X_i \\ \frac{1}{n}\sum_{i=1}^{n} Y_i \\ 1 \end{bmatrix}$$

e i fattori di scala $s_x = \left(\frac{1}{n}\sum_{i=1}^{n}\left(X_i - X_m\right)\right)^{-\frac{1}{2}}$ e $s_y = \left(\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - Y_m\right)\right)^{-\frac{1}{2}}$

Nel dominio delle coordinate normalizzate si avrà

$$\breve{p}_R^{\,T}\breve{F}\,\breve{p}_L=\left(N_R p_R\right)^T\breve{F}\left(N_L p_L\right)=p_R^T N_R^T\breve{F}\,N_L p_L$$

da cui si deduce che

$$F=N_R^T\breve{F}\,N_L \tag{3.25}$$

Questa relazione è determinante per la denormalizzazione finale.

3.3.5 Soluzione ai minimi quadrati

Nel caso in cui a causa del rumore o di errori di misura Q possegga rango pieno, risultando non singolare si può cercare una soluzione approssimata ai minimi quadrati.

Questa soluzione può essere trovata a meno di un fattore di scala, per questo è necessario porre un ulteriore vincolo; in questo modo si esclude anche la soluzione banale $\boldsymbol{f}=0$. Per tale vincolo esistono varie opzioni tra le quali la più utilizzata è quella di porre $\|f\|=1$.

La robustezza dell'algoritmo può essere aumentata con $n>8$. In questo modo si hanno più equazioni lineari che incognite e si può cercare di minimizzare il criterio di errore [Faugeras-01]:

$$C(F)=\sum_{i=1}^{n}\left(p_{Ri}^T F\,p_{Li}\right)^2 \tag{3.26}$$

che equivale a trovare il minimo di

$$C(F)=\|Q\,f\|^2 \tag{3.27}$$

Il vantaggio del metodo degli otto punti è che porta ad un algoritmo non iterativo e quindi è più semplice da implementare, ma oltre alla sensibilità al rumore presenta due problemi rilevanti:

- Non è rispettato il vincolo $det(F)=0$.

- La (4.4) non ha alcuna interpretazione geometrica

3.3.6 Vincolo di singolarità

La proprietà che gli epipoli siano autovettori della matrice F sottolinea il fatto che la matrice F sia singolare e in particolare abbia rango pari a 2.

Per forzare la singolarità della matrice fondamentale normalizzata in modo che le linee epipolari passino tutte per l'epipolo comune, $\breve{F}$ viene rimpiazzata dalla matrice $\overline{F}$ più vicina in norma di Frobenius che soddisfi i requisiti, vale a dire dalla matrice che, sotto la condizione $det(\overline{F})=0$, minimizzi la norma $\|\breve{F}-\overline{F}\|$.

La SVD permette di porre in ordine decrescente i valori singolari di D, perciò si può far corrispondere all'ultima colonna della matrice V il valore singolare più piccolo. Applicando la SVD a $\breve{F}$, per avere una $\overline{F}$ che sia la matrice singolare più vicina in norma di Frobenius si pone a zero il valore singolare minore e si calcola la matrice cercata come

$$\overline{F}=U\,\overline{D}\,V^T=U\,diag(r,s,0)\,V^T$$

in cui U e V sono le matrici ortogonali ottenute dalla SVD di $\breve{F}$.

In questo modo si può ottenere, dopo la denormalizzazione (3.25), una matrice fondamentale singolare, la quale però è stata ottenuta minimizzando una funzione che non possiede una interpretazione geometrica.

3.3.7 Metodi robusti

Abbiamo detto in precedenza di voler minimizzare una quantità

geometricamente significativa, perciò potremmo utilizzare una funzione di errore che tenga conto delle distanze dalle linee epipolari.

In questo modo si può ottenere un residuo r_i per cui vale [Faugeras-01]

$$r_i^2 = d^2(p_{Ri}, F\,p_{Li}) + d^2(p_{Li}, F^T p_{Ri}) \tag{3.28}$$

La struttura per gli algoritmi RANSAC e LmedS, descritti nel capitolo 2, per la stima della matrice fondamentale è la seguente:

Date N corrispondenze $\{(p_{Li}, p_{Ri}) | i = 1,\ldots, N\}$ si seguono i seguenti passi:

1. Vengono generati m sotto-campioni casuali attraverso una tecnica Monte Carlo estraendo per ogni sotto-campione q=7 corrispondenze.

2. Attraverso il metodo dei 7 punti vengono calcolate al più 3 soluzioni F_J per ogni sotto-campione J .

3. Se utilizzo LmedS calcolo m volte

$$M_J = \left\{ \underset{i=1,\ldots,N}{med} \left[d^2(p_{Ri}, F_J\,p_{Li}) + d^2(p_{Li}, F_J^T p_{Ri}) \right] \right\}$$

La F_J che presenta il minor M_J sarà la nostra stima finale per la matrice fondamentale. Se utilizzo RANSAC calcolo il numero di inlier che presentano una distanza minore di τ per ogni soluzione generata al passo 2 e prendo la F_J con maggior consenso.

3.4 Rettificazione

Riassumendo quanto detto in precedenza, lo scopo finale della visione stereo è quello di fornire una stima della profondità degli oggetti attraverso la triangolazione.

Il calcolo della (3.1) non può essere determinato se non si è risolto in

precedenza il problema della corrispondenza, infatti per sapere il valore di X_L e X_R si deve conoscere quali parti delle due immagini sono proiezione dello stesso elemento della scena.

Una mappa delle corrispondenze in grado di fornire per ogni punto dell'immagine di partenza la posizione del pixel corrispondente potrebbe risolvere questo problema.

Se si conosce la geometria epipolare, codificata nella matrice fondamentale, è possibile ridurre il problema ad una ricerca 1D grazie al fatto che punti coniugati si trovano sulla stessa linea epipolare. Naturalmente la ricerca dei punti corrispondenti risulta semplificata se le due immagini possono essere distorte in modo che qualsiasi coppia di punti coniugati si trovi sulla stessa scanline nelle due immagini. In altre parole si vuole trovare una trasformazione geometrica che alteri l'arrangiamento spaziale dei pixel (*image warping*) [Fusiello-2010] in modo che le linee epipolari coincidano nelle due immagini e siano parallele all'asse orizzontale.

La rettificazione epipolare è una omografia che porta l'impianto stereo in una disposizione frontale-parallela riducendo la geometria epipolare in forma canonica.

Le matrici generali delle telecamere in configurazione canonica sono data da [Hartley-2003]

$$\hat{\Psi}_L=[I \quad \mathbf{0}] \quad \text{e} \quad \hat{\Psi}_R=\left[\hat{M}_R \quad \hat{e}_R\right] \tag{3.29}$$

e più nel dettaglio

$$\hat{M}_R=S(\hat{e}_R)F+\hat{e}_R\boldsymbol{v}^T \tag{3.30}$$

in cui ν è un vettore di tre elementi.

La seguente tabella riassume le proprietà fondamentali di un sistema stereoscopico qualsiasi e di quello canonico.

Configurazione non rettificata	**Sistema con epipoli ad ∞**
$\overline{\Psi}_R = M_R [R \quad T]$	$\hat{\Psi}_R = [\hat{M}_R \quad \hat{e}_R]$
$\overline{\Psi}_L = M_L [I \quad \mathbf{0}]$	$\hat{\Psi}_L = [I \quad \mathbf{0}]$
$A = \overline{\Psi}_R \overline{\Psi}_L^{+} = M_R R M_L^{-1}$	$\hat{A} = \hat{M}_R$
$F = S(e_R) A = A^{-T} S(e_L)$	$F = S(\hat{e}_R) \hat{M}_R = \hat{M}_R^{-T} S(\hat{e}_L)$
$e_L = \overline{\Psi}_L C_R = M_L R T$ $e_R = \overline{\Psi}_R C_L = M_R T$	$\hat{e}_L = T$ $\hat{e}_R = \hat{\Psi}_R C_L = \hat{M}_R \hat{e}_L$

Tabella 1: Sommario delle proprietà delle matrici delle telecamere

Si noti che la maggior parte degli algoritmi di visione stereoscopica presentati nella letteratura della computer vision assumono immagini rettificate[Isgrò-99].

Per ottenere delle linee epipolari orizzontali l'omografia rettificante deve mappare gli epipoli ad infinito, perciò

$$\hat{e_L} = \hat{e_R} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.31}$$

In questo modo l'immagine del centro di proiezione in una immagine è parallela all'altro piano immagine. L'esecuzione degli algoritmi di rettificazione di un immagine in genere ha scarso successo quando gli epipoli si trovano all'interno dell'immagine, vale a dire quando l'impianto stereo è caratterizzato da una baseline troppo ampia o con assi ottici troppo convergenti. La trasformazione proiettiva H che mappa gli epipoli all'infinito sfrutta 3 gradi di libertà, quindi rimangono 4 gradi di libertà dai

7 totali di un'omografia qualsiasi. Se i restanti vincoli non vengono scelti opportunamente si rischia di scegliere un'omografia che causa severe distorsioni all'immagini. Poiché ci sono un infinito numero di possibili piani frontali-paralleli da scegliere, si devono introdurre ulteriori vincoli come minimizzare la distorsione o massimizzare la sovrapposizione delle viste.

Esistono algoritmi che necessitano di una completa calibrazione dell'impianto ed altri che invece sono indipendenti dalla calibrazione. In questo senso possiamo distinguere due categorie di algoritmi di rettificazione:

- Uncalibrated stereo rectification
- Calibrated stereo rectification

3.4.1 Algoritmo di Hartley

Questo metodo non utilizza i parametri intrinseci e estrinseci delle telecamere, ma sfrutta il fatto che tale informazione è contenuta nella matrice fondamentale. La rettificazione quindi può essere effettuata semplicemente individuando almeno 7 punti necessari per il calcolo di F. Questo algoritmo tenta di minimizzare le disparità esistenti tra le due viste, ma in genere produce immagini più distorte di quelli che prevedono l'utilizzo dei parametri della calibrazione. Un ulteriore svantaggio è che non essendo a conoscenza dei parametri intrinseci e in particolare della lunghezza focale, non si ha alcuna sensibilità alla scala. Ciò comporta che oggetti nello spazio aventi differente scala e, di conseguenza, punti

caratteristici con differenti coordinate 3D, possano avere medesima proiezione [Bradski-2008].

Il procedimento presentato in [Hartley-98] inizia applicando all'immagine destra l'omografia H_R che mappa all'infinito l'epipolo destro e in un secondo tempo calcola l'omografia corrispondente H_L da applicare sull'immagine di sinistra in modo che sia minimizzata la disparità e quindi la quantità

$$\sum_i d\left(H_L p_{Li}, H_R p_{Ri}\right)^2 \tag{3.32}$$

L'algoritmo per il calcolo della prima omografia può essere schematizzato nel seguente modo:

1. Si sceglie un punto p_O nel cui vicinato può essere desiderabile avere una distorsione minima. Nei successivi passi verrà applicata una trasformazione che nell'origine può essere approssimata ad una trasformazione rigida. L'applicazione di una traslazione (prima di applicare la trasformazione rigida) in grado di portare il punto p_O nell'origine permette di rendere quello scelto il punto a distorsione minima. È ragionevole scegliere come punto da mappare nell'origine il centro dell'immagine $p_O=\begin{pmatrix} X_O & Y_O & 1 \end{pmatrix}^T$, in tal caso la traslazione ha la forma:

$$T=\begin{bmatrix} 1 & 0 & -X_O \\ 0 & 1 & -Y_O \\ 0 & 0 & 1 \end{bmatrix} \tag{3.33}$$

2. Il secondo passo è quello di trovare una rotazione che porti l'epipolo destro sull'asse X. Per la (3.14) possiamo calcolare l'epipolo destro

trovando il nucleo di F^T . Da notare che l'epipolo ha già subito la traslazione al passo 1. Quindi avremo:

$$e_{R1}=\begin{pmatrix} e_{R1x} \\ e_{R1y} \\ 1 \end{pmatrix}=T\,e_R=\begin{pmatrix} e_{Rx}-X_O \\ e_{Ry}-Y_O \\ 1 \end{pmatrix} \tag{3.34}$$

da cui possiamo ricavare l'angolo che allinea e_{R1} con l'ascissa

$$\phi_{eR1}=-atan2(e_{R1y}, e_{R1x}) \tag{3.35}$$

e infine la matrice di rotazione cercata attorno all'asse Z

$$R=\begin{bmatrix} \cos(\phi_{eR1}) & -\sin(\phi_{eR1}) & 0 \\ \sin(\phi_{eR1}) & \cos(\phi_{eR1}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.36}$$

3. L'ultimo passo è quello di trovare una trasformazione che mappa all'infinito l'epipolo $e_{R2}=R\,e_{R1}$ che ora giace sull'asse X e quindi ha la forma $e_{R2}=\begin{pmatrix} f & 0 & 1 \end{pmatrix}^T$ a meno di un fattore di scala che può essere eliminato dividendo per la terza componente. Hartley propone la seguente omografia:

$$K=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{f} & 0 & 1 \end{bmatrix} \tag{3.37}$$

4. Infine si calcola l'omografia finale dalla composizione delle tre trasformazioni:

$$H_R=K\,R\,T \tag{3.38}$$

Dalla (3.30) possiamo ora calcolare $\hat{M}_R$ e le matrici delle telecamere in (2.29).

Hartley dimostra che se la matrice fondamentale è data da $F=S(\hat{e}_R)\hat{M}_R$

allora la trasformazione proiettiva H_L dell'immagine I_L corrisponde ad H_R se e solo se

$$H_L = \left(I + H_R e_R \boldsymbol{a}^T\right) H_R M \tag{3.39}$$

per qualche vettore $\boldsymbol{a}$. Nel nostro caso la (3.39) si può semplificare poiché H_R mappa l'epipolo destro all'infinito:

$$H_L = \left(I + H_R e_R \boldsymbol{a}^T\right) H_R \hat{M}_R = \left(I + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{a}^T\right) H_R \hat{M}_R = H_A H_R \hat{M}_R$$

in cui H_A è una trasformazione affine della forma

$$H_A = \begin{bmatrix} a & b & c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.40}$$

A questo punto trovare una matrice H_L che minimizzi la disparità secondo la (3.32) può essere così riformulata

$$\sum_i d\left(H_A H_R \hat{M}_R p_{Li}, H_R p_{Ri}\right)^2 = \sum_i d\left(H_A \tilde{p}_{Li}, \tilde{p}_{Ri}\right)^2 \tag{3.41}$$

In particolare siano

$$\tilde{p}_{Li} = \begin{pmatrix} \tilde{X}_{Li} \\ \tilde{Y}_{Li} \\ 1 \end{pmatrix} \quad \text{e} \quad \tilde{p}_{Ri} = \begin{pmatrix} \tilde{X}_{Ri} \\ \tilde{Y}_{Ri} \\ 1 \end{pmatrix}$$

allora la (3.41) diventa

$$\sum_i d\left(\left[a\tilde{X}_{Li} + b\tilde{Y}_{Li} + c, \quad \tilde{Y}_{Li}, \quad 1\right]^T, \left[\tilde{X}_{Ri}, \quad \tilde{Y}_{Ri}, \quad 1\right]^T\right)^2$$

e ancora, considerando che con la rettificazione i punti coniugati devono giacere sulla stessa scanline, la disparità dipende soltanto dalla distanza tra le ascisse

$$\sum_i \left(a\,\tilde{X}_{Li} + b\,\tilde{Y}_{Li} + c - \tilde{X}_{Ri}\right)^2 \tag{3.42}$$

Che è un semplice problema di minimizzazione ai minimi quadrati lineare.

3.4.2 Algoritmo di Bouguet

Basandosi sulla conoscenza dei parametri della calibrazione Bouguet spartisce la rotazione determinata dalla matrice R, calcolata a partire dalla (2.33), tra le due telecamere in modo da renderle complanari. In questo modo minimizza la distorsione prodotta dalla riproiezione e rende massima l'area sovrapponibile tra le due immagini[Bouguet]. In particolare dato il vettore di rotazione $\boldsymbol{v}_{LR}$ le cui componenti descrivono l'asse di rotazione tra i due piani immagine e la cui norma $\|\boldsymbol{v}_{LR}\|$ fornisce il valore dell'angolo di tale rotazione, possiamo trovare le due matrici di rotazione

$$r_R = f_{\boldsymbol{ROD}}\left(\frac{-\boldsymbol{v}_{LR}}{2}\right) \quad \text{e} \quad r_L = r_R^T \tag{3.43}$$

rispettivamente per la telecamera destra e sinistra.

Si noti che per la rotazione di destra si è indicato con $f_{\boldsymbol{ROD}}()$ la formula di Rodrigues che converte un vettore di rotazione nella corrispettiva matrice di rotazione, mentre per la rotazione di sinistra è stata utilizzata la proprietà di ortogonalità. Le due "mezze rotazioni" rendono gli assi principali paralleli tra loro e alla somma degli assi ottici originali. Dopo tali trasformazioni i due piani sono complanari, ma non ancora allineati per riga, ciò significa che l'asse di riferimento X_S non è parallelo alla baseline la cui direzione è data dal nuovo vettore traslazione $t = r_R T$. Si deve quindi fare in modo che il vettore t abbia la stessa direzione del

vettore $(1 \quad 0 \quad 0)^T$. Per determinare il vettore di rotazione si deve calcolare il vettore che descrive l'asse della rotazione cercata e l'angolo. Il primo si può calcolare attraverso il prodotto vettoriale

$$\boldsymbol{\omega} = t \times \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{3.44}$$

mentre si può risalire all'angolo di rotazione sfruttando la proprietà del prodotto scalare nel piano definito dagli stessi vettori

$$\phi = \arccos\left(\frac{t \cdot (1 \quad 0 \quad 0)^T}{\|t\| \ \left\|(1 \quad 0 \quad 0)^T\right\|} \right) \tag{3.45}$$

È possibile stabilire il vettore di rotazione moltiplicando il risultato della (3.44) con quello della (3.45) perciò le rotazioni globali da applicare ad entrambe le viste saranno:

$$\begin{aligned} \tilde{R}_R &= f_{\boldsymbol{ROD}}(\omega\phi) r_R \\ \tilde{R}_L &= f_{\boldsymbol{ROD}}(\omega\phi) r_L \end{aligned} \tag{3.46}$$

Per soddisfare la richiesta di collinearità le due nuove matrici dei parametri intrinseci devono avere lo stesso valore $\alpha_y f$. Per semplicità i fattori di scala α_x e α_y presenti nella (2.3) vengono resi uguali ponendo $f_{new} = \alpha_x f = \alpha_y f$, in modo che i pixel risultino quadrati. Il valore di f_{new} viene scelto cercando di trovare una lunghezza focale in grado di mantenere il maggior numero di informazioni contenute nelle due immagini originali distorte. I nuovi punti principali $\hat{Y}_{0L}$ e $\hat{Y}_{0R}$ sono inizialmente calcolati in modo da massimizzare l'area nelle immagini rettificate; poi per la componente verticale viene posto un valore comune facendo la media tra le coordinate verticali dei due punti principali, vale a

dire in formule $\hat{Y}_0 = \frac{\hat{Y}_{0L} + \hat{Y}_{0R}}{2}$.

Le nuove matrici di prospezione avranno quindi la forma:

$$\hat{M}_L = \begin{bmatrix} f_{new} & 0 & \hat{X}_{0L} \\ 0 & f_{new} & \hat{Y}_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \hat{M}_R = \begin{bmatrix} f_{new} & 0 & \hat{X}_{0R} \\ 0 & f_{new} & \hat{Y}_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.47}$$

Le nuove matrici di proiezione prospettica non avranno quindi la forma (3.29), ma avranno matrici dei parametri intrinseci quasi identiche e differiranno tra loro anche per il vettore di traslazione $\hat{T} = \tilde{R}_R T$:

$$\begin{aligned} \hat{\Psi}_L &= \hat{M}_L [I \quad \mathbf{0}] \\ \hat{\Psi}_R &= \hat{M}_L [I \quad \hat{T}] \end{aligned} \tag{3.48}$$

3.4.3 Mappa di rettificazione

Una volta calcolate le trasformazioni di rettificazione con uno dei due metodi si costruisce una mappa in grado di indicare dove deve essere collocato ogni pixel originario nell'immagine destinazione. Tuttavia alle coordinate intere dell'immagine originale corrispondono coordinate reali nell'immagine finale perciò si procede al contrario:

1. Per ogni posizione intera dell'immagine destinazione $\hat{p}$ calcoliamo da quale coordinata in virgola mobile dell'immagine sorgente dovrebbe provenire.
2. Il valore di quel pixel destinazione viene poi calcolato attraverso una interpolazione bilineare dall'immagine sorgente dei valori interi dei pixel più prossimi a quello calcolato al passo 1.

Vediamo ora come determinare la mappa nel caso in cui si è utilizzato il metodo di Bouguet.

La trasformazione (3.48) produce due nuove equazioni corrispondenti alla (2.5) in cui si ha un nuovo punto nel piano immagine

$$\begin{aligned} \hat{p}_L &= \hat{\Psi}_L P^s \\ \hat{p}_R &= \hat{\Psi}_R P^s \end{aligned} \tag{3.50}$$

Dalla equazione (2.12) possiamo trovare i punti del raggio ottico corrispondenti ad ogni pixel intero delle immagini riproiettate per la configurazione (3.48)

$$\begin{aligned} P^s &= \lambda \begin{bmatrix} \hat{M}_L^{-1} \hat{p}_L \\ 0 \end{bmatrix} \\ P^s &= \begin{bmatrix} O_R \\ 1 \end{bmatrix} + \lambda \begin{bmatrix} \hat{M}_R^{-1} \hat{p}_R \\ 0 \end{bmatrix} \end{aligned} \tag{3.51}$$

Applicando una rotazione inversa a quella applicata in (3.46) ricaviamo i punti rispetto al sistema di riferimento telecamera:

$$\begin{aligned} P^c &= \lambda \begin{bmatrix} \tilde{R}_L{}^{-1} \hat{M}_L^{-1} \hat{p}_L \\ 0 \end{bmatrix} \\ P^c &= \lambda \begin{bmatrix} \tilde{R}_R{}^{-1} \hat{M}_R^{-1} \hat{p}_R \\ 0 \end{bmatrix} \end{aligned} \tag{3.51}$$

Dividendo le prime due componenti per la terza si elimina il fattore di scala e si determina la corrispondenza tra le nuove coordinate e i rapporti $\hat{X}_u = P_x^c / P_z^c$ e $\hat{Y}_u = P_y^c / P_z^c$ nelle vecchie coordinate delle immagini originali non distorte.

È necessario aggiungere il contributo della distorsione radiale e tangenziale che abbiamo calcolato durante la calibrazione. Tradizionalmente i coefficienti di distorsione in (2.14) e (2.15) vengono

posti in un vettore 5×1 e ordinati nel seguente modo:

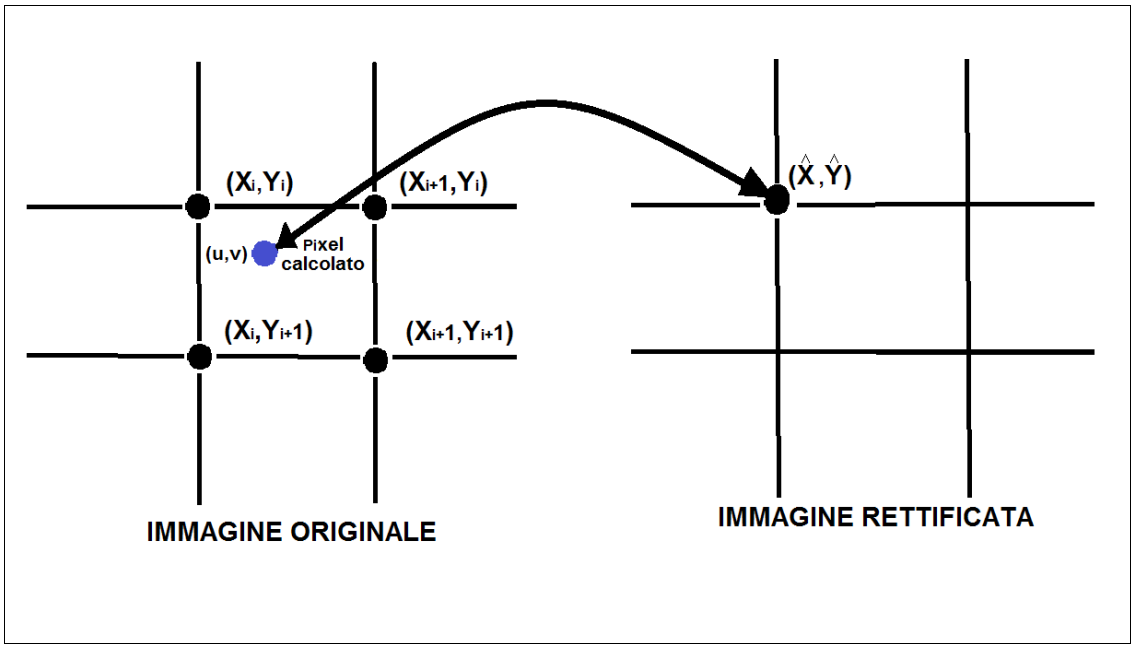
$$DC=\begin{bmatrix} K_{Dr1} & K_{Dr2} & K_{Dt1} & K_{Dt2} & K_{Dr3} \end{bmatrix}$$

Come si può notare il terzo elemento della componente radiale è posto per ultimo, questo per il fatto che è stato inserito solo in un secondo tempo al fine di rendere più accurata la stima della distorsione per le telecamere grandangolari. Le espressioni (2.14) (2.15) possono essere quindi espresse nella forma:

$$\begin{pmatrix} \hat{X}_D \\ \hat{Y}_D \end{pmatrix}=\begin{bmatrix} 1 & DC_1 & DC_2 & DC_5 \end{bmatrix}\begin{bmatrix} 1 \\ \rho^2 \\ \rho^4 \\ \rho^6 \end{bmatrix}\begin{pmatrix} \hat{X}_u \\ \hat{Y}_u \end{pmatrix}+\begin{bmatrix} DC_3 & DC_4 & 0 \\ DC_4 & 0 & DC_3 \end{bmatrix}\begin{bmatrix} 2X_uY_u \\ \rho^2+2X_u^2 \\ \rho^2+2Y_u^2 \end{bmatrix}$$

Utilizzando la (2.3) si può passare alle coordinate immagine

$$\begin{aligned} \left(u_L, v_L, 1\right)^T &= M_L\left(\hat{X}_{DL}, \hat{Y}_{DL}, 1\right)^T \\ \left(u_R, v_R, 1\right)^T &= M_R\left(\hat{X}_{DR}, \hat{Y}_{DR}, 1\right)^T \end{aligned} \qquad (3.52)$$



Questa relazione sta ad indicare che il valore assunto dall'immagine destinazione nel punto $(\hat{X},\hat{Y})$ a coordinate intere dovrebbe essere lo stesso del punto (u,v) a coordinate reali nell'immagine originale. In realtà tale punto non corrisponde ad alcun punto (X,Y) a coordinate intere dell'immagine originale perciò si utilizza un processo di interpolazione. Più precisamente dalle coordinate (u,v) calcolate nella (3.52) si determina il punto (X_i,Y_i) le cui coordinate sono il risultato dell'approssimazione di u e v al valore intero minore uguale più vicino. A partire da tali coordinate si valutano i seguenti parametri:

$$\begin{aligned} dx &= u - X_i \\ dy &= v - Y_i \end{aligned} \tag{3.53}$$

che servono poi per determinare i coefficienti a_i dell'interpolazione bilineare:

$$\begin{aligned} a_1 &= (1-dx)\cdot(1-dy) \\ a_2 &= dx\cdot(1-dy) \\ a_3 &= (1-dx)\cdot dy \\ a_4 &= dx\cdot dy \end{aligned} \tag{3.54}$$

Il valore finale del pixel dell'immagine rettificata allora sarà

$$I(\hat{X},\hat{Y}) = a_1 I(X_i,Y_i) + a_2 I(X_{i+1},Y_i) + a_3 I(X_i,Y_{i+1}) + a_4 I(X_{i+1},Y_{i+1}) \tag{3.54}$$

I parametri (3.54) devono essere calcolati per ogni pixel dell'immagine destinazione, ma per ridurre i tempi di calcolo possono essere determinati soltanto una volta.

Se si è utilizzato il metodo di Hartley non sarebbe necessario utilizzare i parametri intrinseci, ma nel caso in cui siano comunque disponibili i parametri da una calibrazione, per praticità si può riutilizzare la stessa procedura che sfrutta il calcolo della mappa di rettificazione utilizzata con

il primo metodo. In tal caso le matrici utilizzate in (3.51) possono essere sostituite nel seguente modo [Bradski-2008]:

$$\begin{aligned} \tilde{R}_L^{-1}\hat{M}_L^{-1} &\rightarrow M_L^{-1}H_L M_L \\ \tilde{R}_R^{-1}\hat{M}_R^{-1} &\rightarrow M_R^{-1}H_R M_R \end{aligned} \tag{3.55}$$

Capitolo 4
Hardware e software impiegati

4.1 L'Hardware

Il software sviluppato necessita di una parte fisica costituita, oltre che dal calcolatore senza il quale non potrebbe essere eseguito, anche da un sistema con delle caratteristiche specifiche senza le quali non è in grado di svolgere i compiti per i quali è stato progettato. Questo sistema è costituito da una coppia di telecamere poste su un supporto in grado di predisporle nella maniera più simile possibile a quella di un sistema visivo binoculare.

4.1.1 Sistema di elaborazione

Un ambiente di elaborazione live deve fornire prestazioni elevate in modo tale che le scene acquisite presentino meno scarti possibile.
Un primo punto fondamentale è costituito dalla potenza del processore e dal numero di core in esso contenuti. Un secondo punto è costituito dalla memoria di sistema: entrano in gioco sia la memoria primaria (RAM) che quella della scheda grafica del sistema su cui verranno eseguiti i programmi.
È stato possibile testare il software con risultati soddisfacenti sui seguenti calcolatori:

- AMD Phenom(tm) II Quad-Core 3.30GHz con una

RAM istallata di 4.00 GB, scheda video ATI Radeon X1950 GT con 256MB di memoria dedicata.

- AMD Athlon(tm) Dual-Core 2.00GHz con una RAM istallata di 4.00 GB
- Intel centrino duo (T7200) 2.00 Ghz, RAM 2.00 GB, scheda video Invidia GeForce 7600 con 256 di Ram interna dedicata

Questo dimostra come, nonostante i tempi di risposta giochino un ruolo centrale e quindi siano auspicabili elevate prestazioni, sistemi di elaborazione di fascia media forniscano prestazioni sufficienti.

4.1.2 Telecamere e loro collocazione

L'applicazione è stata sviluppata sfruttando principalmente due webcam Hercules Dualpix Exchange con collegamento al calcolatore mediante USB. Queste telecamere possiedono un sensore VGA CMOS, forniscono una risoluzione in modalità video con interpolazione di 1280×960 e un frame rate fino a 30 fps.

Poter utilizzare il programma con vari tipi di telecamere è un aspetto considerato rilevante; pertanto il software è stato testato anche con una telecamera integrata Sony Visual Communication VGP-VCC2 con pixel effettivi 640×480 dotata di un sensore VGA CMOS.

La coppia di telecamere deve essere fissata in modo da risultare disposta in modo analogo al sistema visivo degli animali dotati di visione binoculare. Per far questo si è usato un supporto in legno, intagliato in modo da far

incastrare il sistema di fissaggio proprio delle webcam Hercules Dualpix Exchange.

Illustrazione 22: Posizionamento delle webcam con supporto in legno per il fissaggio (a destra) e senza (a sinistra)

Questa struttura elementare si è dimostrata spesso non necessaria, ciò sta a dimostrare che l'applicazione è facilmente utilizzabile senza un grande sforzo nel progettare un telaio complesso. Può essere sufficiente posizionare con cura le webcam su qualsiasi supporto facendo attenzione che siano sufficientemente rigide, parallele e non troppo distanti. Un accidentale (o volontario) spostamento delle telecamere costringe l'utente a ricalibrarle, ma il software permette di farlo in pochi minuti.

La maggior parte delle webcam non è stata creata per questo genere di applicazioni in cui la rigidità è una prerogativa centrale. Le Hercules, ad esempio, hanno rotazione orizzontale fino a 360 gradi e rotazione verticale fino a 30 gradi che nel contesto per le quali sono state progettate è un importante pregio.

Sistemi di visione stereoscopica progettati per l'elaborazione delle scene

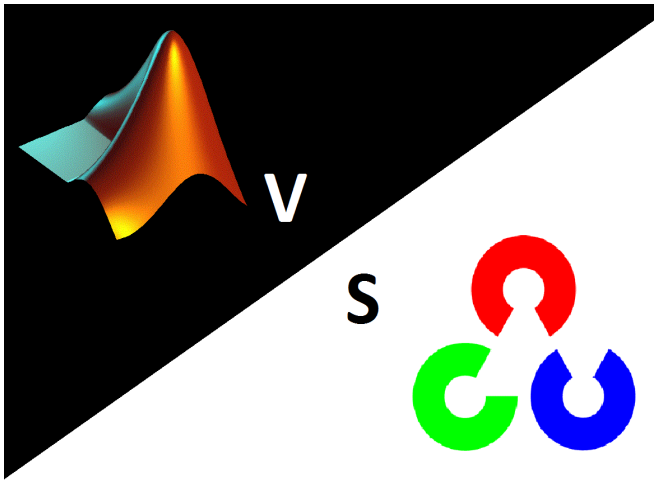
reali attualmente hanno costi relativamente elevati che difficilmente lo studente o il ricercatore, che voglia acquistare personalmente la strumentazione, vorrà affrontare. Allo stesso tempo per poter riutilizzare le webcam anche in altre attività è necessario un sistema che permetta di passare dalla configurazione stereoscopica ad una all'occorrenza più flessibile che permetta di sfruttarle anche per attività lavorative, come videoconferenze, o ludiche. Un sistema semplice come quelli mostrati nell'illustrazione 22 permette tutto questo semplicemente sganciando le telecamere dal supporto e rimuovendo gli elastici utilizzati per impedirne la rotazione. L'utente è libero di creare il proprio supporto senza eccessivo sforzo.

4.2 Software utilizzato

4.2.1 Il linguaggio di programmazione

La prima scelta fondamentale per l'applicazione sviluppata è stata quella del linguaggio di programmazione. Nell'ambito della computer vision sono stati utilizzati molti linguaggi di programmazione e naturalmente è possibile dire quale sia il migliore solo in relazione agli scopi del progetto che si vuole sviluppare. Storicamente si è ricorso con maggior frequenza, tra le varie possibilità esistenti, alle librerie di computer vision OpenCV o all'ambiente di sviluppo Matlab. In questo progetto la scelta è ricaduta sul primo sebbene Matlab abbia degli innegabili vantaggi. Infatti anche se per un programmatore poco esperto Matlab può risultare più intuitivo,

l'utilizzo di librerie sviluppate in linguaggio C/C++ permette di guadagnare velocità di esecuzione, fattore che nell'elaborazione delle immagini in tempo reale è di fondamentale importanza.



Questo aspetto è a volte messo in secondo piano per il fatto che l'Image Processing Toolbox di Matlab consente di trattare le immagini come matrici, che sono un tipo di dato universale in questo ambiente, riducendo le perdite in velocità di esecuzione. La "lentezza" in realtà è dovuta soprattutto al fatto che uno script Matlab viene prima convertito in un linguaggio intermedio detto p–code e poi ogni istruzione del p–code viene eseguita in sequenza dall'interprete, generando un certo overhead.
L'incremento di efficienza delle prestazioni si traduce in un aumento anche di 5 volte in termini di frame elaborati al secondo.

4.2.2 OpenCV

OpenCV è una libreria open source sotto licenza BSD, originariamente sviluppata da Intel.

Questa libreria è stata progettata per l'efficienza computazionale, soprattutto in considerazione di applicazioni in tempo reale, con la possibilità di trarre vantaggio dall'utilizzo di processori multicore; inoltre è multipiattaforma e può essere utilizzata con vari linguaggi di programmazione come il Python oltre che con il C/C++.
Le librerie sono inoltre corredate da numerosi esempi nonché di manuali che illustrano dettagliatamente tutte le funzioni a disposizione.
L'utilizzo della libreria OpenCV consente, con poco sforzo, di ottenere risultati apprezzabili sia dal punto di vista prestazionale che da quello fornito dagli stessi esempi.

4.2.3 L'ambiente di sviluppo e il Sistema Operativo

Un buon ambiente di sviluppo è fondamentale nel creare un'applicazione con un grado accettabile di usabilità e determina inoltre specifiche scelte per quanto riguarda il S.O. su cui si può eseguire il programma.
Il progetto in analisi è stato interamente sviluppato mediante l'ambiente di sviluppo Microsoft Visual Studio e pertanto il software è eseguibile soltanto su S.O. Microsoft, ma si è cercato di tenere conto anche della portabilità delle funzionalità di base. Queste non dipendono dal sistema operativo e perciò senza dover ritoccare il codice potranno essere riscritte in qualsiasi altra piattaforma, adeguando però quelle parti inerenti all'interfaccia grafica proprie di VS.
Per quanto riguarda le versioni dei sistemi operativi Microsoft sono stati effettuati test sulle versioni Windows: XP Media Center 2005, Windows

Vista e Windows 7.

L'applicazione ha dimostrato di funzionare su tutte queste con buoni risultati; in particolare la prima delle versioni citate, nonostante sia ormai considerata obsoleta, regge la complessità delle operazioni real time nel senso che:

- Non mostra interruzioni nella fase di visualizzazione ed identificazione durante l'acquisizione delle immagini (sia con singola che con doppia telecamera).
- Non perde fotogrammi nella fase di scrittura del dato acquisito sul disco di sistema
- Procede senza scarti di fotogramma nella fase di sincronizzazione dei processi che recuperano il dato.

Durante la fase di elaborazione lo sforzo di CPU e la memoria dinamica utilizzate restano sempre, indipendentemente dalla complessità delle immagini, al di sotto delle soglie limite previste. Il consumo in MB durante la fase di esecuzione dei processi di acquisizione immagine con 2 telecamere in funzione, è di circa 40 MByte mentre le due CPU viste dal sistema operativo, presentano rari picchi comunque al di sotto del 40%.

Capitolo 5
Distribuzione dei Processi

5.1 Moltiplicare i processi per gestire più telecamere

Le librerie OpenCV utilizzate non consentono di gestire facilmente, attraverso un singolo programma o processo, un sistema costituito da più telecamere. Al contrario, la scissione del generico programma di acquisizione in più processi distinti, associati, ognuno, ad una singola telecamera, risolve semplicemente il problema.

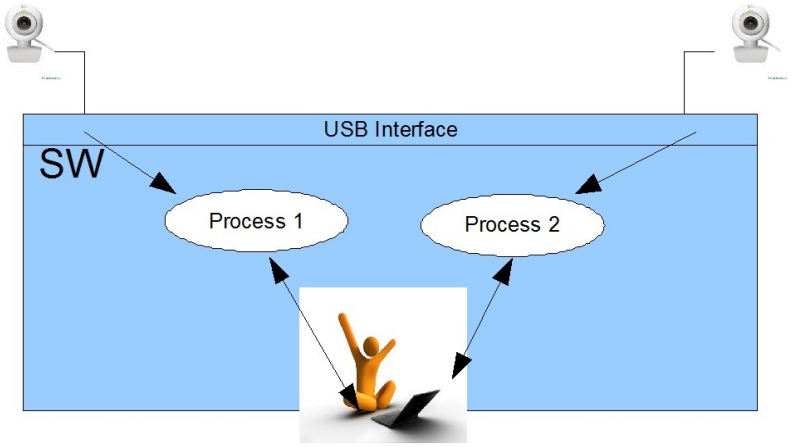


Illustrazione 23: Schema della gestione di più telecamere mediante scissione in più processi

5.1.1 Puntare sul parallelismo

Al di fuori della semplificazione del problema, pensando ai moderni processori multi-core o ad ambienti multi-processore, la scelta di scindere

il problema in più sotto-problemi risulta un ottimo approccio per la distribuzione dei compiti a livello di singolo core o singolo processore.
In altre parole questa scelta permette di attribuire l'elaborazione di un compito specifico ad un singolo processore e di lasciare gli altri a disposizione del sistema o di altri compiti.
Considerando, ad esempio, un sistema di acquisizione stereo ed un processore quad-core si potrebbe attribuire due core alla gestione delle telecamere (acquisizione, elaborazione di base delle immagini ecc.) e gli altri due core per la gestione del sistema e di un terzo processo che permetterà all'utente di interagire con i processi in questione passando loro i parametri ed acquisendone dei risultati.

5.1.2 Interazione utente/processi

In una situazione di questo tipo l'interazione diretta dell'utente con i processi dedicati deve essere ridotta al minimo e, se possibile, addirittura non esservi. Il tipo di interazione desiderata dovrà essere puramente indiretta.
In altre parole l'utente dovrà disporre di una interfaccia grafica che gli permetta di interagire indirettamente con i processi (eseguendone - ad esempio - startup, shutdown, ripristino e l'insieme di tutte le operazioni a cui gli stessi sono dedicati).
L'introduzione di questo strato di software che si frappone fra l'utente ed il core dell'applicazione deve risolvere i problemi di comunicazione fra i vari

processi che si verranno a creare.
Ci sono differenti modi per risolvere questo problema e, in funzione dell'ambiente (Microsoft, Unix ecc) in cui si presenta, le soluzioni sono più o meno performanti.
L'idea di base è quella di fare in modo che il sistema possa mantenere la propria logica di base nel miglior modo possibile e che possa essere compilato ed eseguito sulla maggior parte degli ambienti in commercio.

5.2 Scelta del metodo di comunicazione tra processi

Fra i metodi che permettono a due o più processi di comunicare, la *shared memory* presenta una filosofia comune in ambienti Microsoft e Unix.
In generale possiamo dire che la *shared memory* è un'area di memoria primaria le cui dimensioni vengono stabilite a priori dal programmatore, il quale deve fare in modo che non si verifichino conflitti durante la fase di lettura e di scrittura che possano compromettere il funzionamento dell'intero sistema che i processi stanno gestendo. È possibile fare questo mediante timer o altre azioni particolari in modo da avere una opportuna sincronizzazione.

In ambienti Microsoft la shared memory è implementata come “file mapping” per condividere un'area di memoria tra processi o threads. Il primo processo crea l'oggetto di file mapping chiamando la funzione `CreateFileMapping` con permessi di lettura e scrittura in memoria. Questa funzione associa un nome all'oggetto di file mapping e ritorna un

handle allo stesso oggetto che viene poi passato alla funzione `MapViewOfFile` per mappare una regione selezionata del file in memoria. Questa funzione restituisce un puntatore `pBuf` al primo byte della regione del file mappato. Il processo utilizza quindi la funzione `CopyMemory` per scrivere nella regione puntata da `pBuf`. Quando il processo non deve più accedere alla memoria condivisa è opportuno chiamare la funzione `CloseHandle` a cui verrà passato l'handle da chiudere associato all'oggetto di file mapping.

Un secondo processo può accedere alla regione scritta nella memoria condivisa dal primo processo, chiamando la funzione `OpenFileMapping` specificando lo stesso nome per l'oggetto di file mapping. Ora si può utilizzare la funzione `MapViewOfFile` passandole l'handle all'oggetto di file mapping per ottenere l'indirizzo iniziale dell'area mappata.

Nel codice sviluppato è stata definita ed implementata una opportuna classe atta alla gestione della shared memory le cui funzioni membro permettono di:

- Allocare uno spazio di RAM e di associarlo ad un identificativo univoco che tutti i processi interessati potranno utilizzare per raggiungere i contenuti dell'area in condivisione.
- Accedere all'area in modalità di *scrittura* per imporvi dati
- Accedere all'area in modalità di *lettura* per recuperane i dati
- Chiudere l'area al termine del suo utilizzo

Queste funzionalità di base potranno essere riscritte in qualsiasi altra piattaforma senza dover ritoccare il codice che è stato scritto durante

l'implementazione dei processi dedicati alle funzionalità di base dell'applicazione.

Lo scambio dati, avviene, per questi motivi, utilizzando un'area di memoria condivisa, come riportato nella figura sotto:

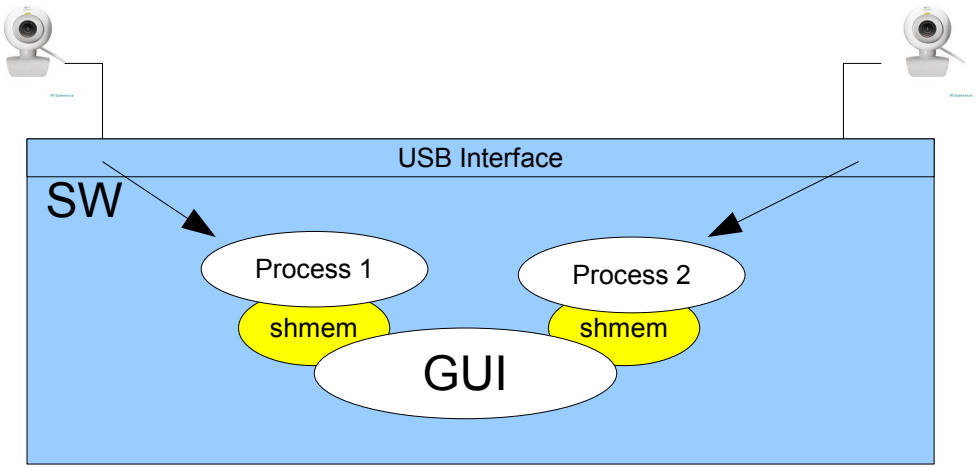


Illustrazione 24: Schema dell'interazione Processo - Shared Memory

Ogni differente processo è caratterizzato da una propria struttura dati attraverso la quale può interscambiare dati con la GUI.

Si è scelto, per evitare problemi di sincronizzazione, che sia la stessa GUI, al momento opportuno, a generare l'area di interscambio e che solo dopo questa operazione sia attivato il processo. Essendo la proprietaria dell'area, sarà ancora la GUI a terminare il processo imponendo nella shared memory uno stato opportuno, ad attendere alcuni istanti e a distruggere definitivamente l'area di memoria in condivisione. Il processo, riconosciuto il segnale di richiesta di terminazione, esce dal proprio loop principale e termina la propria esecuzione.

5.3 Strutture di base

A titolo di esempio si riportano le strutture ed i dati[2] utilizzati dai processi di acquisizione e calibrazione. La struttura di base è stata definita attraverso i seguenti due tipi di dato strutturato:

- `ScacchieraType`
- `CalibrazioneType`

Questi vengono sotto riportati per questioni di praticità:

```
typedef struct _scacchiera
{
          CvSize DimScacchiera;
          float FormatoQuadrato;
} ScacchieraType;


typedef struct _calibrazione
{
          int tipo;
          char baseOutputFile[512];
          unsigned char sincro;
          ScacchieraType sc;
} CalibrazioneType;
```

`ScacchieraType` contiene:

- *`DimScacchiera`*, di tipo CVSize (definito all'interno della libreria OpenCV), costituito da due campi interi che definiscono, in quadrati, le dimensioni della scacchiera che verrà utilizzata.
- *`FormatoQuadrato`*, un float che definisce la dimensione del singolo quadretto della scacchiera nell'unità di misura scelta dall'utente.

CalibrazioneType contiene, invece, tutti i dati di controllo che verranno passati dalla GUI al processo di calibrazione. In particolare:

- *`tipo`*, un intero che definisce il tipo di calibrazione che il processo dovrà

2 Presenti nel file commons/commons.h

applicare, valorizzato a:

1 - nel caso l'operatore richieda una acquisizione live delle immagini.

2 - nel caso in cui l'operatore richieda di effettuare una calibrazione stereo da immagini acquisite

3 - nel caso in cui l'operatore richieda una calibrazione da video[3]

- *baseOutputFile*, una stringa di 512 byte all'interno della quale viene passato il path completo del file xml di output
- *sincro,* un byte che viene utilizzato nel solo caso di acquisizione live per:
  - richiedere la chiusura del programma se valorizzato a `FF`
  - richiedere lo scatto di una fotografia se valorizzato a `01`
  - porre in stato di attesa la GUI se valorizzato a `00`
- `sc`, dato di tipo `ScacchieraType,` attraverso il quale vengono passate le caratteristiche che l'operatore impone a livello di interfaccia grafica.

5.4 Acquisizione delle immagini

Si introdurranno ora gli aspetti implementativi riguardanti l'acquisizione delle immagini, dando spazio alla logica dietro la comunicazione tra i processi e all'algoritmo impiegato. In questa sezione saranno soltanto accennate le operazioni che l'utente dovrà effettuare per eseguire una calibrazione, le quali saranno approfondite nei capitoli successivi.

Per questo genere di operazioni le librerie OpenCV mettono a disposizione la classe `VideoCapture`. L'associazione di un oggetto di questo tipo ad un dispositivo per l'acquisizione video permette di effettuare su di esso un gran numero di operazioni come aprirlo e prelevarne i fotogrammi.

3 La funzionalità “calibrazione da video” non è ancora operativa allo stato attuale, ma è stata predisposta per eventuali sviluppi futuri

La Graphical User Interface consente all'utente di attivare (o disattivare) l'apertura di una o entrambe le telecamere (si veda a tal proposito il paragrafo 7.1.3) in modo da poter decidere, quanti processi saranno eseguiti in parallelo, purché le telecamere siano effettivamente collegate al calcolatore. Una volta imposta tale scelta, una maniera per avviare un processo di acquisizione è quello di accedere al dialogo predisposto alla realizzazione di una nuova calibrazione. L'operatore, nel primo pannello dedicato all'acquisizione live delle immagini, con caratteristiche simili a quelle riportate nell'illustrazione 25, impone l'esecuzione dei processi.

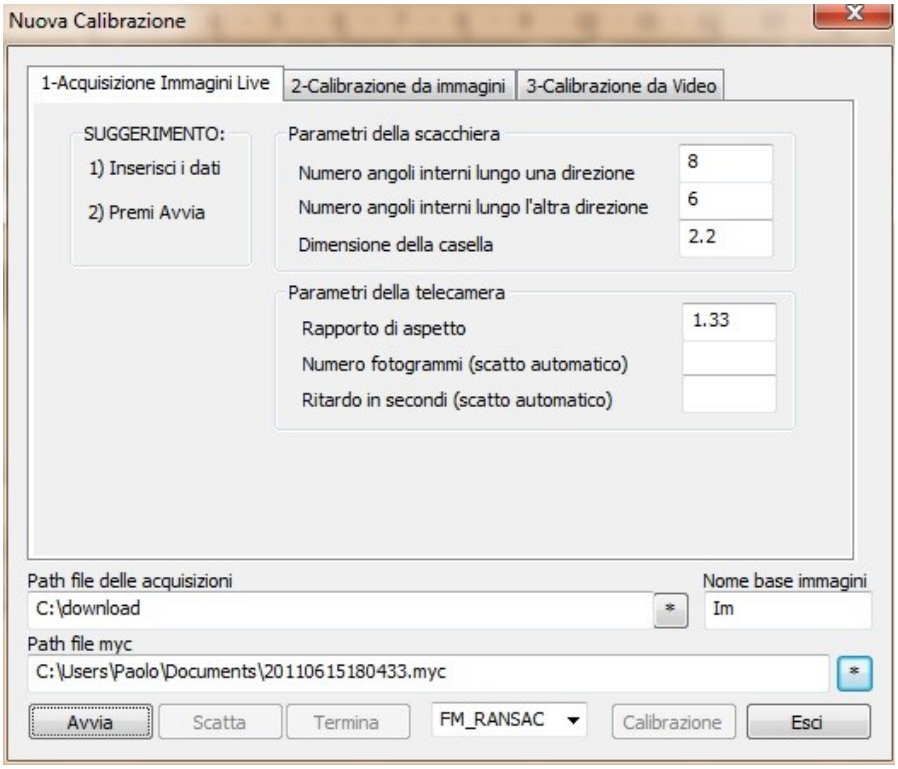


Illustrazione 25: Pannello per l'acquisizione delle immagini

5.4.1 Algoritmo relativo al tasto "*Avvia*"

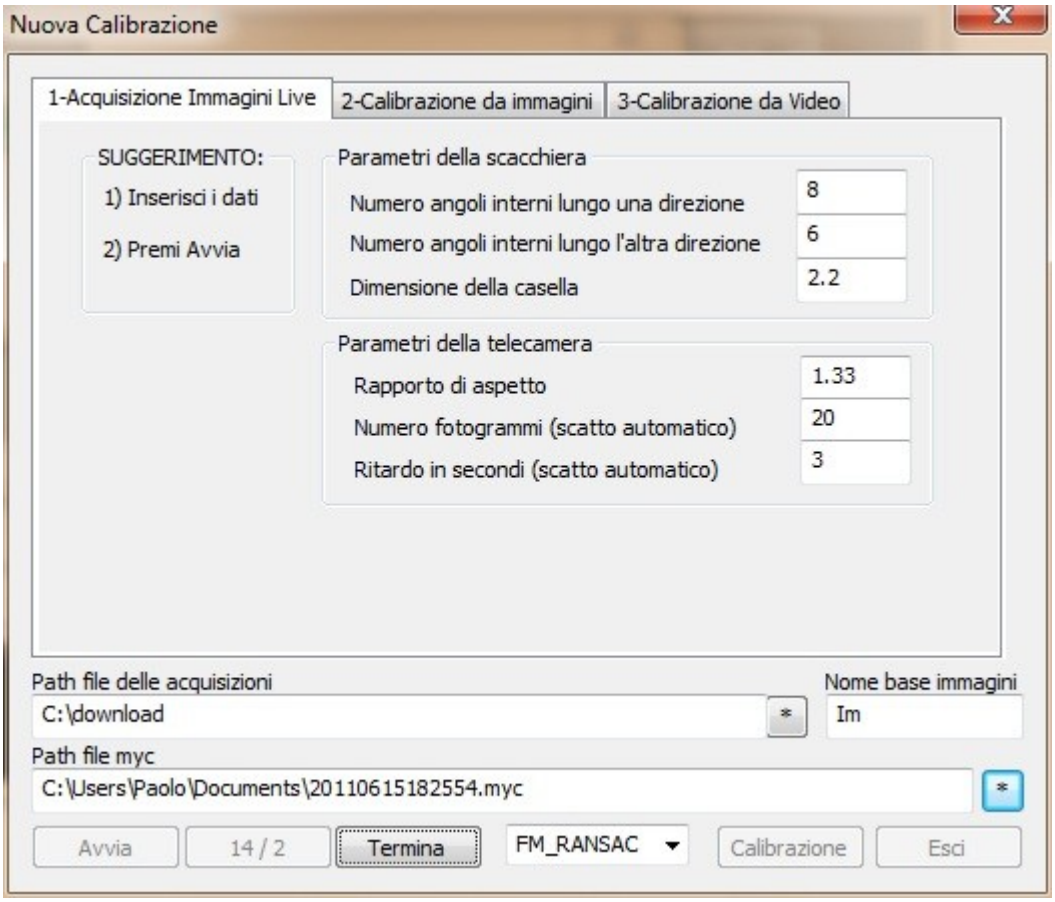


Illustrazione 26: Inserimento dei dati e attivazione dello scatto automatico

Alla pressione del tasto *Avvia* il codice associato al dialogo:

1) Recupera i parametri inseriti.
2) Verifica la validità dei dati.
3) Recupera il path completo del file xml in cui memorizzare i risultati e verifica la possibilità di aprire il file in questione.
4) Recupera il path della directory in cui scaricare le immagini
5) A seguito di tali verifiche, impone, i suddetti parametri, in una variabile

locale del programma GUI, di tipo `CalibrazioneType`.

6) Disabilita il pulsante Avvia così da non essere cliccato un'altra volta
7) Impone, al parametro *`sincro`* di tale area, il valore esadecimale `00` , ad indicare che non vi è alcuna richiesta di interruzione o di scatto da parte dell'operatore.
8) Genera tante aree di shared memory di dimensione 256 byte quante sono le telecamere attivabili. Il numero di telecamere e quali (destra o sinistra) dipende dalle impostazioni del programma stesso.
9) Verifica che le aree siano state correttamente allocate e, per ognuna di esse, impone il dato presente nella variabile di tipo `CalibrazioneType` (la cui dimensione complessiva è minore dei 256 byte predisposti)
10) Attiva i processi di acquisizione per tutte le telecamere definite nel proprio file di configurazione.
11) I processi associati hanno il tempo per predisporre e riconoscere la propria telecamera, acquisendo il dato di input dalla periferica USB a cui la stessa risulta collegata, e ad attivarla raggiungendo lo stato di acquisizione, vale a dire un ciclo while infinito terminabile soltanto da comando proveniente dalla GUI, nel quale si limitano a visualizzare le immagini che stanno scorrendo davanti all'obiettivo. In questo stato permangono fino a quando l'operatore non esegue una scelta a livello di interfaccia grafica.
12) Ultima operazione in questa parte di codice è l'attivazione di un timer (con codice identificativo 12345) che impone, nella coda dei segnali associati alla GUI, un evento ad ogni secondo.
13) Il codice della funzione `OnTimer()` della GUI, tratta questo evento accedendo alla shared memory condivisa con i/il processo di acquisizione attivi/o e sulla base dello stato riportato in *`sincro`*. Se *`sincro`* vale `00` in tutte le aree interessate dal processo, allora nessun processo di acquisizione risulta impegnato ed è quindi possibile "scattare una foto*";* quindi la GUI

abilita il tasto "*Scatta"*. In caso contrario si ha, in qualche area utilizzata, almeno un valore di `sincro` diverso da `00`, quindi esiste almeno un processo che sta impegnando la CPU; di conseguenza la GUI inibisce il tasto Scatta.

In altre parole, ad ogni secondo, la GUI verifica lo stato di sincronizzazione di tutte le telecamere attivate solo per abilitare o disattivare la possibilità di premere il tasto di scatto.
Al primo passaggio, ovvero subito dopo l'attivazione dei processi di acquisizione, la GUI troverà tutti i byte di sincronizzazione a 00 (avendoli impostati lei stessa come descritto nel precedente punto 7) e renderà immediatamente disponibile i tasti "*Scatta*" e "*Termina*".

5.4.2 Algoritmo associato al tasto "*Scatta*"

Alla pressione, da parte dell'operatore, del pulsante di scatto:

a) La GUI accede alla shared memory di ogni processo di calibrazione attivo ed impone il valore esadecimale `01`.
b) disabilita, immediatamente, la possibilità di premere una seconda volta il tasto Scatta e riprende il suo *ciclo di polling*.
c) A questo punto i compiti sono demandati ai processi di acquisizione (descritti in seguito) che, trovando il byte a `01` verificano lo stato di ritorno della funzione di rilevazione angoli. Al termine di tale ciclo il byte di sincronizzazione ritornerà a `00`.

5.4.3 Algoritmo relativo al pulsante "*Termina*"

Alla pressione di questo tasto:

1. La GUI esegue la chiusura del timer, allocato a seguito della pressione del tasto Avvia, attraverso la funzione `KillTimer(12345)` ed impone il valore `FF` a tutti i `sincro` dell'area condivisa.
2. Ogni processo di acquisizione, ricevendo questo "segnale", terminerà qualsiasi interazione con la shared memory ed uscirà dal proprio ciclo infinito. Attenderà, quindi, alcuni secondi, prima di terminare la propria esecuzione.
3. La GUI, trascorsi alcuni millesimi di secondo, chiuderà, definitivamente, le aree di memoria allocate per la pressione del tasto Avvia.

Capitolo 6
Sincronizzazione dei processi

I due processi di acquisizione e la GUI interagiscono fra di loro secondo un ordine ben preciso che evita conflitti durante la fase di elaborazione.

Si consideri "buona" una immagine risultante da una acquisizione che permetta di riconoscere i punti di intersezione della griglia utilizzata e "non buona" una immagine che non consente la rilevazione di tali punti, questo senza tener conto delle condizioni ambientali, in cui avviene l'acquisizione, che influenzano tale rilevazione.

Una immagine, in considerazione dello scopo del suo salvataggio, vale a dire il suo utilizzo per una calibrazione stereo, non deve essere considerata come una entità indipendente, ma come una metà delle due immagini che costituiscono una visione binoculare.

Una coppia di immagini è utilizzabile solo se entrambe sono "buone" e in tal caso vanno registrate in una directory prescelta e passate in una fase successiva al processo di calibrazione stereo.

Al contrario nel caso in cui entrambe le telecamere acquisiscano risultati "non buoni" entrambe le immagini sono da scartare.

Quando soltanto una sola delle due immagini acquisite risulta accettabile, come nell'illustrazione 27, si deve considerare il *problema della sincronizzazione*.

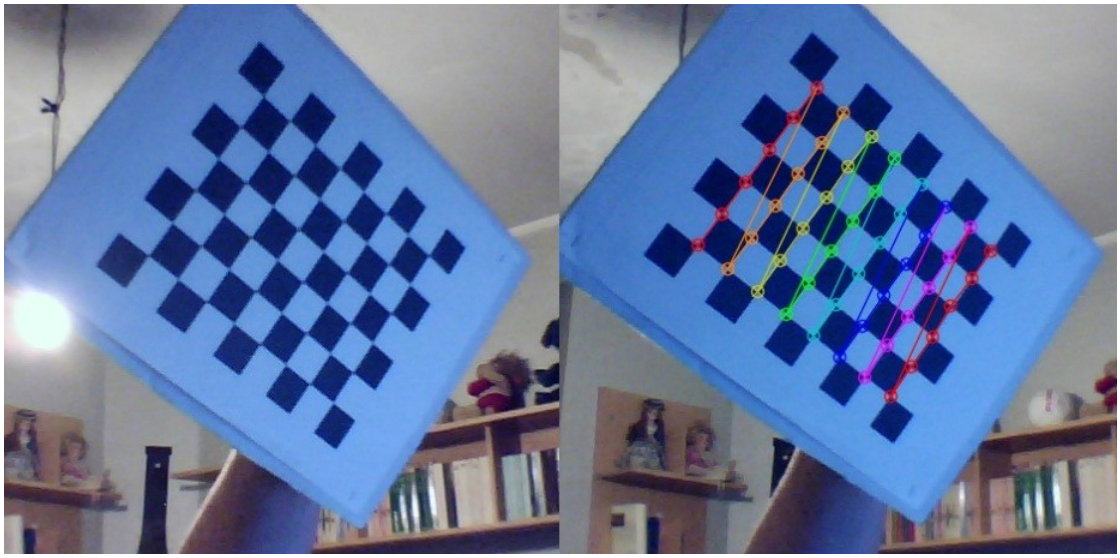


Illustrazione 27: Coppia di immagini in cui soltanto quella di destra permette la rilevazione degli angoli

L'eliminazione di una soltanto delle immagini porta ad una cattiva sincronizzazione che produce un disallineamento negli indici delle immagini acquisite. In questo modo la corrispondenza fra immagine destra ed immagine sinistra verrebbe perduta e non potrebbe essere utilizzata da automatismi che siano in grado di riconoscerla a meno di interventi manuali sui filename prodotti dai processi. Un altro aspetto riguarda la corretta sincronizzazione fra la macchina ed il suo utilizzatore per impedire l'introduzione di possibili problemi a livello di coerenza di dati acquisiti. Questo porta ad una semplice considerazione: uno scatto non può essere effettuato fino a quando non si è completato, da parte delle due telecamere, l'elaborazione associata allo scatto precedente. Per semplificare il problema si è ipotizzato che, indipendentemente dal fatto che una delle due telecamere abbia acquisito una immagine *buona,* se l'altra non ha raggiunto lo stesso obiettivo, la coppia di immagini deve essere scartata. In tal caso

cioè nessuna delle due immagini dovrà essere salvata sul disco del PC in utilizzo. Questo vincolo permette, fra l'altro, di attribuire sequenze numeriche complete nelle quali non si formano salti come nella tabella seguente:

Imma_sx_001	Imma_sx_002	Imma_sx_005	...
Imma_dx_002	Imma_dx_003	Imma_dx_005	...

Tabella 2: Sequenze di filename con salti indesiderati

ma si hanno sequenze:

Imma_sx_<i>

Imma_dx_<i>

nelle quali gli indici <i> sono associati ad acquisizioni avvenute nello stesso istante.

6.1 Funzionamento del processo di acquisizione

Il generico processo di acquisizione associato ad una delle due telecamere funziona secondo il seguente schema:

1. Start e recupero dei parametri fondamentali per il funzionamento dall'area di memoria che condivide con il programma GUI
2. Ingresso in un loop infinito nel quale recupera, dalla struttura passata nella shared memory dalla GUI, un byte (valori compresi fra 00 e FF) utilizzato per la sincronizzazione dei processi,
3. In base al valore recuperato:
    - `FF` : Termina la propria esecuzione

- `00` : Procede nella propria esecuzione recuperando una immagine e visualizzandola
- `01` : Verifica che il recupero dei *corner* nell'immagine correntemente acquisita sia andato a buon fine

Nei primi due casi non ci sono operazioni da compiere mentre nel terzo il programma deve verificare lo stato di ritorno del processo di rilevazione angoli:

- se negativo: l'immagine non deve essere salvata
- se positivo: l'immagine deve essere salvata se anche il processo associato all'altra telecamera, ha dato lo stesso esito

6.2 Funzionamento della GUI

La GUI impone, sotto richiesta dell'utente o tramite timer, lo scatto di una immagine nelle aree di memoria che vengono condivise con i processi associati alle due telecamere e si pone in attesa dei risultati di questi ultimi.

Se il valore assunto dal byte di sincronizzazione, interno alle aree di memoria condivise con i processi dedicati, comunica che i risultati sono entrambi positivi, la GUI informa l'utente di una acquisizione di coppia corretta. Al contrario, se almeno uno dei due valori è negativo, informa l'utente dell'errore e su quale delle due telecamere. Al termine di questa verifica, la GUI riporta il valore del byte di sincronizzazione a `00` attendendo lo scoccare del prossimo timer o della prossima richiesta di scatto da parte dell'utente.

6.3 Diagramma di funzionamento

6.3.1 Processo di acquisizione

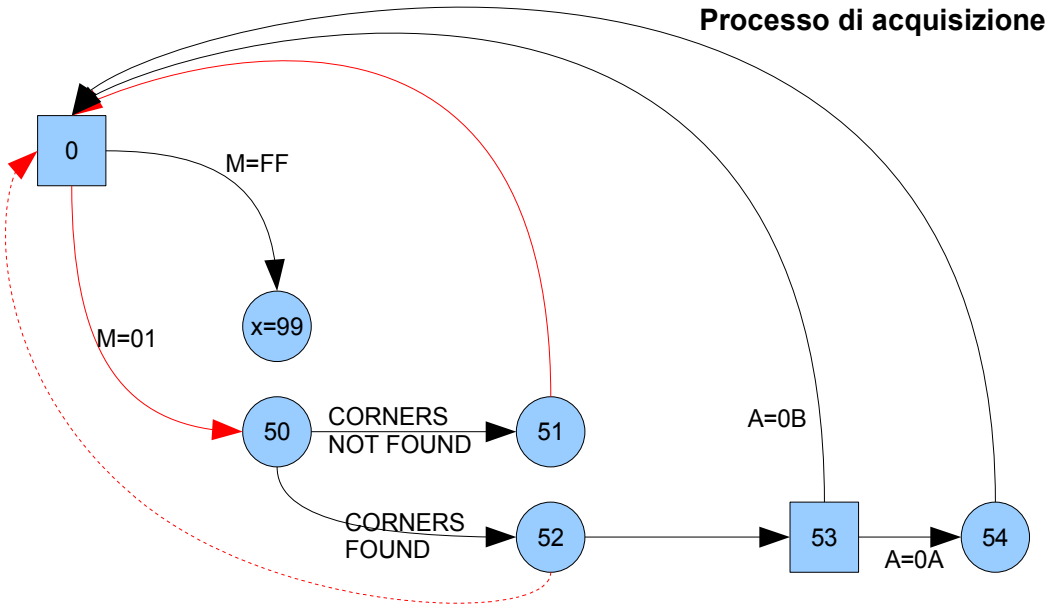


Illustrazione 28: Diagramma di funzionamento dei processi di acquisizione

La figura precedente esplicita il diagramma a stati del processo di acquisizione e mostra il valore dei vari stati all'interno di figure geometriche aventi il seguente significato:

- *Cerchio*: transizione gestita direttamente dal programma. La transizione verso il prossimo stato, è regolata da un evento che lo stato corrente può determinare autonomamente[4].
- *Quadrato*: transizione verso uno stato successivo regolata anche da

4 E' il caso dello stato 50: il passaggio agli stati 51 o 52 è vincolato dal risultato assunto da una funzione invocata internamente al processo.

eventi esterni, ad esempio recuperati dalle aree di shared memory che il processo condivide con altri processi[5].

Le tipologie di transizione fra stati sono indicate con delle frecce dai seguenti tratti:

- *Transizione di colore nero*: indica un percorso che viene eseguito nel caso in cui siano in funzione entrambe le telecamere
- *Transizione di colore rosso:* indica un percorso che il processo intraprende sia nel caso sia in funzione una sola telecamera che nel caso in cui siano in funzione entrambe
- *Transizione di colore rosso tratteggiata*: indica che il processo intraprende quella strada nel solo caso in cui sia in funzione una sola telecamera

6.3.1.1 Stato 0

1. In questo stato il processo attende richieste provenienti dalla GUI controllando il valore del byte di sincronizzazione.
2. Se arriva un segnale di acquisizione da parte della GUI passa allo *stato 50*.
3. Se non arriva alcun segnale (`sincro==00`) si limita ad acquisire

5 Ad esempio, considerando lo stato 0, il processo permane in questo stato fino a quando dalla memoria condivisa con la GUI non viene un valore differente da FF o da 01. La lettera M (mio) indica che l'informazione è letta dall'area di memoria che la GUI utilizza per comunicare con il processo medesimo.

una nuova immagine e a visualizzarla nella propria interfaccia di output (questa interazione non è esplicitata in figura).

4. Se arriva una richiesta di chiusura da parte della GUI (`sincro==FF`) termina la propria esecuzione (x)

6.3.1.2 Stato 50

In questo stato, raggiungibile sia che il processo venga eseguito per una o per due telecamere, viene effettuato un controllo al valore booleano di ritorno della funzione `cvFindChessboardCorners()`. In relazione al valore restituito si ha:

- FALSE: il processo passa allo *stato 51*.
- TRUE: il processo, salva temporaneamente l'immagine su disco e passa allo *stato 52*.

6.3.1.3 Stato 51

A questo stato si giunge per esecuzioni sia con coppie di telecamere che con singola telecamera. Si tratta di uno stato di passaggio, a cui si arriva nel caso di fallimento nella rilevazione degli angoli, all'interno del quale il processo impone, nell'area di memoria che condivide con la GUI, il valore 0A a `sincro` per segnalare che non sono stati trovati angoli nell'elaborazione precedente.

6.3.1.4 Stato 52

In questo stato, a cui si giunge se ha dato esito positivo la rilevazione di angoli, il processo d'acquisizione impone a `sincro` un valore pari a `0B`.

L'immagine, temporaneamente salvata nello stato 50, ha due alternative:

- nel caso di esecuzione singola viene direttamente confermata
- nel caso di esecuzione doppia può essere definitivamente convalidata se anche l'altro processo abbia raggiunto lo stesso risultato

Nel primo caso non sono necessarie altre operazioni ed il controllo può tornare alla GUI. Il processo continua la propria esecuzione acquisendo e visualizzando nuove immagini nell'intervallo prefissato.

Nel secondo caso deve avvenire la verifica dei risultati otteniti dall'altro processo e vanno intraprese le conseguenti azioni, per fare ciò si passa allo *stato 53*.

6.3.1.5 Stato 53

A questo stato si giunge solo per esecuzioni con coppia di telecamere.

Qui il processo controlla sia il byte di sincronizzazione nell'area di memoria che condivide con la GUI sia il valore dello stesso byte del processo gemello.

Se non giungono segnali di interruzione da parte della GUI si hanno queste alternative:

- Se il processo gemello segnala una acquisizione positiva (`sincro==0B`) il processo corrente si sblocca e ritorna allo stato di partenza.
- Al contrario, se il processo gemello da un segnale negativo (`sincro==0A`) passa allo *stato 54*, dove eseguirà la pulizia delle aree temporaneamente impegnate.

6.3.1.6 Stato 54

A questo stato si giunge solo per esecuzioni con coppia di telecamere e solo nel caso in cui l'esito della `cvFindChessboardCorners()` sia:

a) Positivo per la telecamera corrente

b) Negativo per l'altra.

Rappresenta, quindi, solo uno stato di transizione all'interno del quale:

1. Viene cancellata l'immagine temporaneamente salvata.
2. Il contatore di immagini salvate viene resettato al valore precedente.
3. Viene dichiarato il fallimento anche per questa acquisizione. Questo punto permette alla GUI di determinare che l'acquisizione corrente non è andata a buon fine per entrambe le telecamere.
4. Si torna allo *stato 0*.

6.3.2 GUI

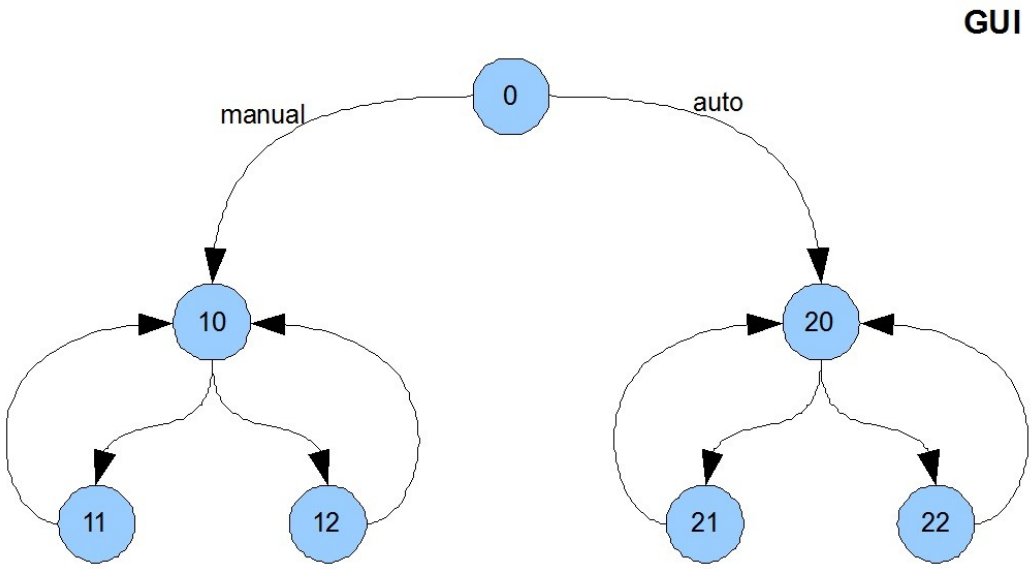


Illustrazione 29: Diagramma di funzionamento della GUI

Alla GUI sono affidate tutte le operazioni di interazione diretta con l'utente. Dall'illustrazione 26 si può vedere che l'interfaccia relativa ad una nuova acquisizione è divisa in schede attraverso un TabControl. La GUI è in grado di distinguere in quale scheda è avvenuta la pressione di un determinato tasto. In funzione delle scelte dell'operatore, la GUI si limita ad ordinare, se l'utente si trova nel primo di questi pannelli, l'acquisizione di una coppia di immagini dalle due telecamere e ad analizzarne i risultati finali proponendo un messaggio di esito sull'acquisizione corrente allo stesso operatore. Essa ha due modalità di funzionamento: quella automatica e quella manuale. Nel primo caso gli scatti sono gestiti da un evento timer, nel secondo da una scelta dello stesso utente. Anche se gli stati proposti nello schema precedente sono molto simili nelle operazioni

compiute è conveniente distinguerli in base al funzionamento: questo evita l'utilizzo di troppi statement if che compromettono la facile lettura del codice e danno adito a cattive interpretazioni. La prima operazione associata alla pressione del pulsante Avvia è il recupero dei parametri inseriti che impone alla GUI il passaggio per lo *stato 0* prossimo descritto.

6.3.2.1 Stato 0

In questo stato la GUI determina, in relazione a quali parametri l'utente ha inserito, se la modalità di esecuzione debba essere manuale o automatica. Sulla base di questo impone uno tra i seguenti stati:

- Stato 10: modalità di esecuzione manuale.
- Stato 20: modalità di esecuzione automatica.

A questo punto la GUI, come descritto nel paragrafo 5.4, sulla base del fatto che siano collegate una o due telecamere, genera le aree di memoria necessarie per il corretto funzionamento del programma e attiva il/i processi di acquisizione. Attiva, quindi il timer per il controllo dei processi e delle richieste utente

6.3.2.2 Stato 10

In questo stato il programma verifica che vi sia un comando utente attraverso la pressione dei tasti *Scatta* o *Termina* e, in base al fatto che vi sia una o più telecamere, interroga ciclicamente (*polling*) il contenuto delle

shared memory alla ricerca di stati di processo comunicati dai programmi di acquisizione. Permane nello stato 10 fino a quando non rileva la richiesta da parte dell'utente di terminare i processi di acquisizione mediante la pressione del tasto *Termina* oppure siano rilevati:

- i valori `0A` o `0B` nel caso di una singola telecamera,
- la coppia di valori (`0A`, `0A`) o (`0B`,`0B`) nel caso di doppie telecamere.

I valori riportati comunicano se si ha avuto un insuccesso o meno dell'operazione richiesta. Si hanno i seguenti casi:

- fallimento: passa allo *stato 11*,
- successo: passa allo *stato 12*.

6.3.2.3 Stato 11

Visualizza un messaggio sull'interfaccia grafica nel quale esplicita l'errore sulla tentata acquisizione e riattiva il tasto scatta

6.3.2.4 Stato 12

Visualizza un messaggio sull'interfaccia grafica nel quale esplicita la riuscita dell'operazione e riattiva il tasto scatta.

Capitolo 7
Interfaccia grafica e tutorial

Questo capitolo descriverà i dialoghi che costituisco la Graphical User Interface e allo stesso tempo mostrerà tutte le funzionalità del programma allo scopo di guidare passo passo anche l'utente più inesperto.

Illustrazione 30: Interfaccia grafica all'avvio del programma

7.1 Approccio iniziale

7.1.1 Esecuzione del programma

Il programma oggetto della presente tesi ha effetto soltanto su S.O. Microsoft. Per la sua esecuzione sono necessarie le Dynamic-link library:

- *opencv_calib3d220.dll*
- *opencv_core220.dll*
- *opencv_highgui220.dll*
- *opencv_imgproc220.dll*

Nonché i file eseguibili:

- *AcqImg.exe*
- *calibrazione.exe*
- *GUI.exe*

Il lancio del programma, che deve avvenire mediante l'ultimo eseguibile, provoca la visualizzazione della finestra principale dotata di barra degli strumenti e del menù mostrata nell'illustrazione 30.

È da tenere presente durante la lettura dei documenti visualizzati dalla GUI che il tratto rosso con cui sono presentati alcuni valori o comunicazioni ha un significato negativo. In altre parole quel dato potrebbe indicare l'impossibilita di attivare una determinata operazione oppure che verosimilmente quel valore si discosta dalla grandezza attesa.

7.1.2 IDD_ABOUTBOX

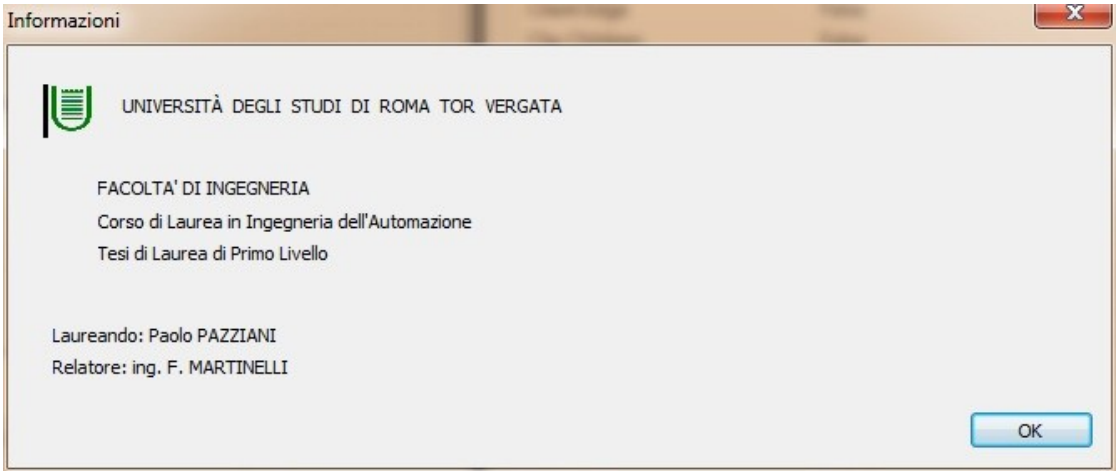


Illustrazione 31: Dialogo per la visualizzazione di informazioni

Il dialogo che visualizza le informazioni relative all'autore del programma

non permette interazioni dirette fra utente e sistema se non la sua chiusura (attraverso la pressione del tasto OK). La classe di controllo `CaboutDlg` è definita ed implementata direttamente nel file GUI.cpp

7.1.3 Impostazione dei processi di acquisizione

È possibile notare la tabella "Modalità di esecuzione impostate" che, in corrispondenza della telecamera riportata nella prima colonna, indica nella seconda colonna:

- Non Attivabile se non è possibile aprire la corrispondente telecamera
- Attivabile in caso contrario

E nella terza colonna:

- Visualizzazione OFF se è disattivata la cattura dei fotogrammi
- Visualizzazione ON in caso contrario

È possibile modificare tali parametri selezionando "Impostazioni" ➪ "Processi Acquisizione" in modo da aprire il dialogo "Controllo dei processi di acquisizione" visibile nell'illustrazione 32.

Questo dialogo ha come identificatore `IDD_IMPOSTAZIONE_PROCESSI` e la sua classe di controllo (`CimpostazioneProcessi`) è implementata nel file ImpostazioneProcessi.cpp e definita in ImpostazioneProcessi.h. Esso, attraverso i check box, controlla il numero di processi associati a qualsiasi fase di acquisizione imponendo nella shared memory utilizzata i comandi di controllo del processo e, se non ancora attivi, permette di impostarli per la prossima esecuzione provocando l'aggiornamento della tabella

nell'interfaccia iniziale. Il tasto "*Reimposta On Line*", se la telecamera relativa è stata attivata in precedenza, forza temporaneamente i nuovi valori nella shared memory ad essa associata.

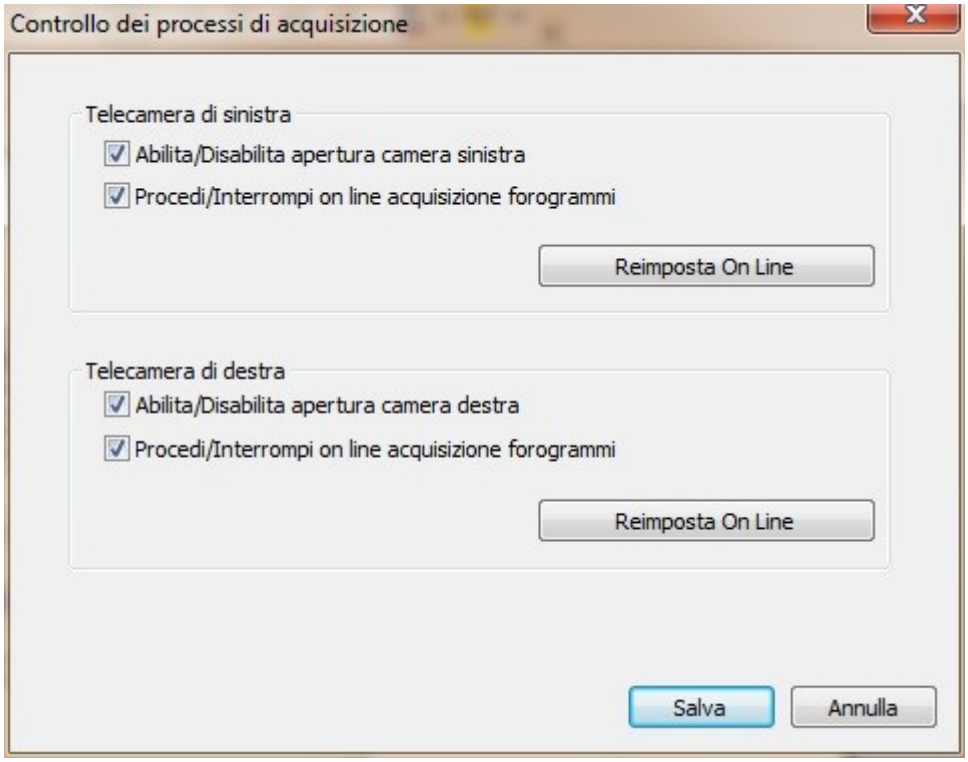


Illustrazione 32: Dialogo per l'impostazione dei processi di acquisizione

7.1.4 Avviare i processi di acquisizione

La pressione della prima icona della toolbar o, in modo equivalente, la selezione di "Controllo" ➪ "Avviare processi acquisizione" permette l'esecuzione dei processi di cattura di flussi video precedentemente impostati. Ogni abilitazione dell'apertura di una telecamera provoca l'esecuzione del processo associato, la visualizzazione di una finestra per la comunicazione delle operazioni eseguite e un dialogo, fornito dalla classe

`VideoCapture` e rappresentato nell'illustrazione 33, che permette la selezione del dispositivo desiderato. Se non si riscontra l'avvio di alcun processo video, nonostante si sia verificato che le telecamere siano state completamente attivate, significa che l'applicazione non riesce a rilevare nessun dispositivo; questo può avvenire per incompatibilità o perché effettivamente nessuna telecamera è collegata correttamente al PC.

Questa prima modalità di esecuzione dei processi di acquisizione ha lo scopo di testare la capacità dell'hardware e del software di supportare questo tipo di applicazione. Nel caso in cui la visualizzazione delle immagini vada a buon fine, senza rallentamenti da parte del sistema nel suo complesso, si ha una indicazione positiva sull'efficacia dell'applicazione in quel particolare contesto.

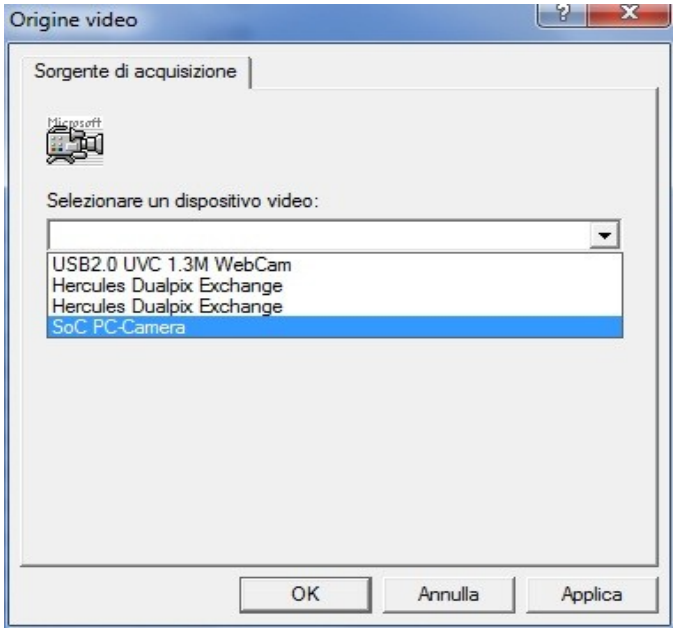


Illustrazione 33: Scelta del dispositivo da cui acquisire il video

Un altro utilizzo può essere quello di ausilio alla sistemazione delle telecamere. Il progetto presuppone l'utilizzo di un sistema di visione stereoscopico, perciò le due telecamere devono essere parallele. Se lo studente o il ricercatore non è in possesso di una telecamera già predisposta per la stereoscopia, le sue telecamere non saranno rigide fra loro e saranno facilmente rotabili, così inquadrando un oggetto con entrambe le telecamere si potrà aumentare il parallelismo tra gli assi ottici cercando di far sì che l'oggetto si trovi sulle stesse *scanline* in entrambe le inquadrature. Allo stesso modo il flusso video permette di sistemare la messa a fuoco manuale.

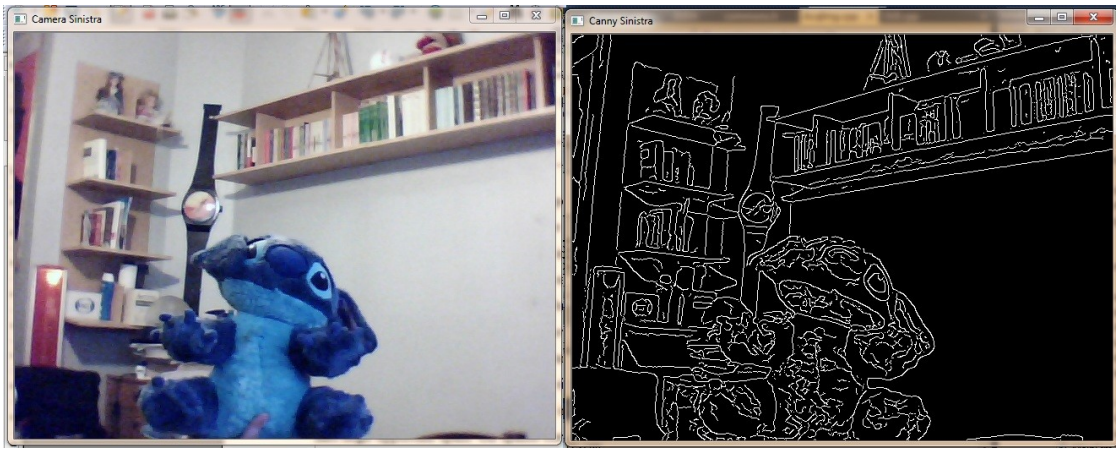


Illustrazione 34: Immagine originale e sua trasformata di Canny

Quando una telecamera è selezionata e confermata si avvia un flusso video per quello specifico dispositivo in una finestra di nome "Camera Sinistra" o "Camera Destra" e associata ad ognuna di queste immagini se ne apre un'altra finestra che visualizza la relativa trasformata di Canny[6]. In questo

6 Eseguita con una specifica funzione OpenCV da un esempio in http://opencv.willowgarage.com/documentation/cpp/reading_and_writing_images_and_video.html

contesto la trasformata di Canny non ha un impatto diretto sui risultati dell'applicazione, sebbene questo tipo di trasformazione abbia importanti applicazioni nell'Image Processing, ma ha il duplice scopo di sollecitare l'utilizzo di memoria volatile (per mettere alla prova l'hardware e la gestione della RAM da parte del sistema operativo) e mostrare come sia possibile riutilizzare il programma per sviluppi futuri che prevedano questo genere di elaborazioni.

7.2 Avviare le operazioni inerenti una calibrazione

Una volta sistemate le telecamere è possibile procedere con la calibrazione cliccando sull'icona a scacchiera o, in modo equivalente, su “Controllo” ➪ “Calibrazione”.

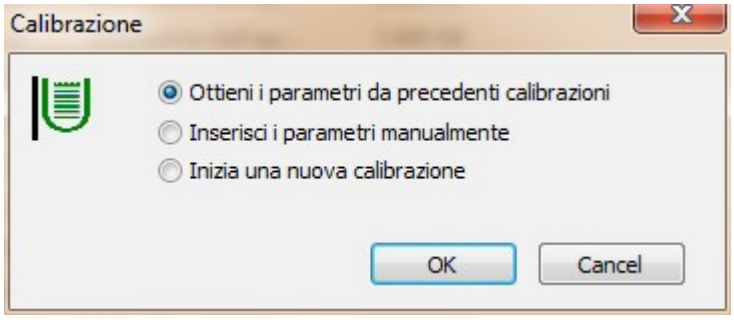


Illustrazione 35: Finestra di dialogo con radio-button

La classe Calibrazione, definita nel file Calibrazione.h ed implementata nel file Calibrazione.cpp, controlla il dialogo IDD_CALIBRAZIONE il quale presenta tre radio-button (illustrazione 35) di cui i primi due permettono la scelta dei parametri di calibrazione, caricando insiemi di dati precedentemente definiti, che verranno applicati durante le fasi successive,

mentre la terza opzione consente di acquisirne di nuovi.
Se è la prima volta che si esegue una calibrazione è necessario selezionare l'ultimo radio-button il quale, una volta premuto il tasto di conferma, avvia il dialogo IDD_NUOVA_CALIBRAZIONE nell'illustrazione 3.

7.3 Implementazione del tab control

La struttura di base del controllo a schede, implementata mediante la libreria Microsoft Foundation Classes, è presentata nell'illustrazione 36. All'apertura del dialogo vengono inizializzati i campi (Tab 1, Tab 2, ...), mentre alla pressione del singolo elemento la tab si ridimensiona opportunamente e consente la visualizzazione al suo interno di un sotto-dialogo.

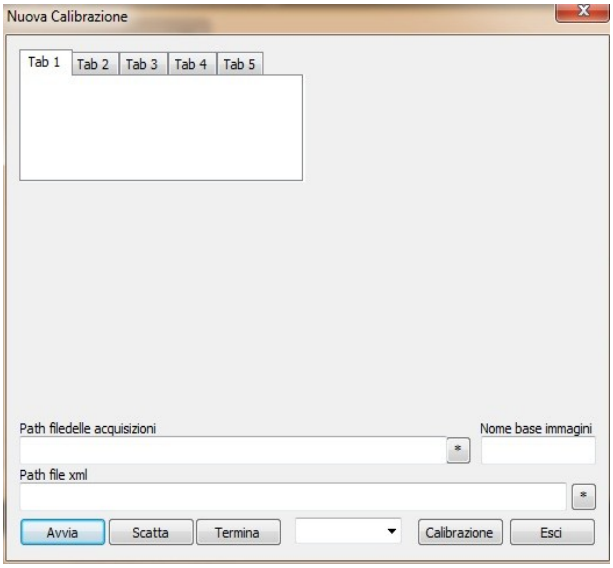


Illustrazione 36: Struttura del Tab control inserito in IDD_NUOVA_CALIBRAZIONE

I sotto-dialoghi permettono di scegliere tra le seguenti operazioni:

1. `IDD_CALIBRAZIONE_LIVE`: Acquisizione e salvataggio delle immagini per la calibrazione
2. `IDD_CALIBRAZIONE_LISTA_IMMAGINI:` Calibrazione dalle immagini salvate
3. `IDD_CALIBRAZIONE_VIDEO`

Ad ogni sotto-dialogo è associata una classe inglobata all'interno della classe CmyTabCtrl, implementata e definita in CmyTabCtrl.cpp e CmyTabCtrl.h, che deriva dalla CTabCtrl di MFC ereditandone tutti i metodi e sovrascrivendo quelli per il controllo dei Tab in modo tale che vengano visualizzati i sotto-dialoghi alla pressione del tab corrispondente.

7.4 Primo sotto-dialogo e raccolta delle immagini

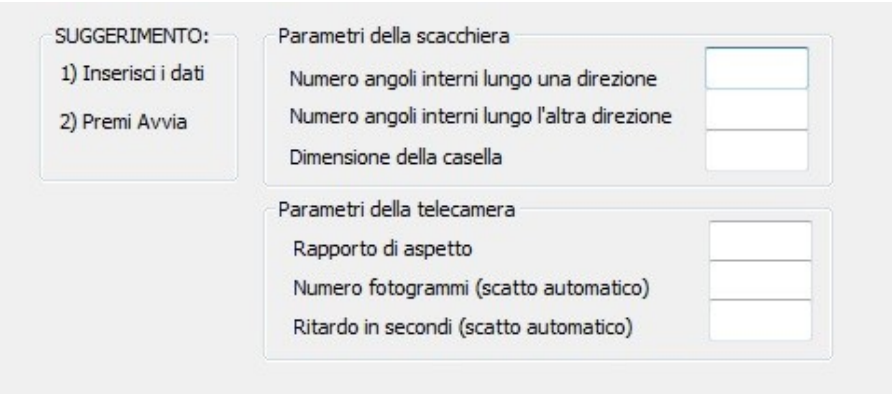


Illustrazione 37: Sotto-dialogo IDD_CALIBRAZIONE_LIVE

La classe di controllo SubDlgCalibrazioneLive, relativa al primo sotto-dialogo, implementata e definita, rispettivamente, nei file SubDlgCalibrazoneLive.cpp e SubDlgCalibrazioneLive.h, è presentata

nell'illustrazione 37 e permette l'inserimento dei parametri di base in modo che la calibrazione possa essere correttamente eseguita.

7.4.1 Inserimento dei dati

Nei primi due spazi di `IDD_CALIBRAZIONE_LIVE`, partendo dall'alto, vanno inseriti gli angoli interni (in orizzontale e in verticale) presenti nella scacchiera; ad esempio per la scacchiera dell'illustrazione 27 va inserito 6 e 8 in qualsiasi ordine. Tali valori ovviamente devono essere maggiori di 2 altrimenti nella relativa direzione si hanno meno di 2 quadrati.
È consigliabile imporre il numero massimo di angoli rilevabili in ogni direzione in quanto valori inferiori, nella maggior parte dei casi, ne impediscono l'estrazione o, ancor peggio, possono portare alla rilevazione di corner differenti nelle due immagini, come può essere notato osservando gli angoli illuminati nella prossima figura.

Illustrazione 38: Rilevazione errata dovuta ad un inserimento non corretto dei parametri

A fianco di "dimensione della casella" va inserita la lunghezza di un lato di una casa degli scacchi in qualsiasi unità di misura, tuttavia l'immissione di

dati in centimetri porta a risultati finali di più facile lettura. In questo caso il controllo sul dato impone valori maggiori di zero.

In “Rapporto di aspetto” si deve inserire il rapporto tra lunghezza e altezza dell'immagine che assume valori compresi tra 1 e 2; se ad esempio si hanno immagini 640x480 il valore dovrà essere 1.33, vale a dire che si hanno immagini in 4/3.

Come si può costatare dalle illustrazioni 25 e 26, nella parte inferiore del dialogo principale sono presenti altri campi di input:

a) Path file delle acquisizioni
b) Path file myc
c) Nome base delle immagini

Le prime due aree accolgono il percorso, rispettivamente, delle directory in cui andranno salvate le immagini e quello del file myc[7] contenente i risultati finali della calibrazione.

Questi due campi possono essere inseriti in due modi differenti:

- manualmente, nel qual caso è preferibile verificare la correttezza del dato.
- attraverso la pressione del tasto contrassegnato da “*” che apre una finestra di dialogo per la selezione della directory.

Dalle finestra per la selezione della directory è possibile navigare su differenti dischi presenti sul PC in utilizzo o navigare all'interno di un singolo disco. La directory per la raccolta di immagini, per default, all'apertura del dialogo viene presentata come quella di esecuzione della

7 Il nome dell'estensione myc è stato scelto in quanto abbreviazione di “my calibration”

GUI ed il relativo disco in cui la stessa è stata definita. Il file myc al contrario viene impostato nella cartella documenti e il suo nome di default deriva da data e orario in cui è stato premuto il tasto "*". Per cambiare directory è necessario imporre il disco, selezionare la directory e alla pressione del tasto imposta, preferibilmente dopo aver verificato il campo "Directory correntemente selezionata", il nome della directory prescelta verrà sovrascritto nel campo associato al tasto premuto.
Per l'area inerente a "Nome Base Immagine" va inserita la radice dei nomi dei file, con estensione jpg, da salvare.
Questi parametri sono sufficienti per effettuare una raccolta di immagini nella modalità con scatto manuale (illustrazione 25), mentre i prossimi due sono necessari se si vuole effettuare una calibrazione con scatto automatico (illustrazione 26).
Lo spazio relativo a "Numero fotogrammi" permette di stabilire il numero di scatti da effettuare, mentre "Ritardo in secondi" imposta il tempo che intercorre tra uno scatto e il successivo.

7.4.2 Scatto manuale o automatico

Lo scatto manuale conferisce una maggiore consapevolezza a priori delle immagini che saranno salvate e successivamente utilizzate per la calibrazione. È possibile infatti posizionare la scacchiera nel modo ritenuto più opportuno e cliccare sul pulsante scatta. In questo caso però l'utente sarà costretto a cliccare sul pulsante "Scatta" per ogni immagine da

catturare e, allo stesso tempo, muovere in maniera idonea la scacchiera.
Nel caso di scatto automatico, dovrà preoccuparsi solo del posizionamento della scacchiera perdendo però il controllo del momento esatto dello scatto. L'indubbia comodità di questa modalità si riflette quindi in una serie di immagini collocate in posizioni non predeterminate e, in taluni casi, se l'operatore non è sufficientemente tempestivo nel sistemare la scacchiera in zone di inquadratura comune tra le due telecamere, si rischia di diminuire il numero di fotogrammi preventivati in fase di inserimento dati. Se tali problemi dovessero risultare considerevoli, un modo per ridurre questi rischi è quello di aumentare il "ritardo in secondi" effettuando una dilazione tra gli scatti e avere così più tempo per collocare opportunamente il pattern di calibrazione.
Una volta catturate le immagini queste vengono salvate nella directory scelta al cui interno, al termine, saranno presenti tre sequenze di immagini i cui filename, supponendo che il nome di base sia "Im", sono mostrati nella seguente tabella:

ImDestra0001.jpg	*ImDestra0002.jpg*	*ImDestra0003.jpg*	...
ImDestra0001_ CORNERS.jpg	*ImDestra0002_ CORNERS.jpg*	*ImDestra0003_ CORNERS.jpg*	...
ImSinistra0001.jpg	*ImSinistra0002.jpg*	*ImSinistra0003.jpg*	...
ImSinistra0001_ CORNERS.jpg	*ImSinistra0002_ CORNERS.jpg*	*ImSinistra0003_ CORNERS.jpg*	...

Tabella 3: Esempio di sequenza di filename salvata su disco

Le immagini che presentano il suffisso "CORNERS" evidenziano i punti salienti della scacchiera.

Nella modalità automatica il programma si arresterà al termine del numero di fotogrammi inseriti, mentre nel caso manuale sarà necessario premere "Termina". A questo punto è possibile passare alla seconda sessione per la calibrazione vera e propria selezionando la seconda scheda del TAB control.

7.4.3 Benefici apportati all'applicazione dal sistema di memorizzazione dei fotogrammi

Non tutte le altre applicazioni per la calibrazione di telecamere prevedono l'acquisizione e il salvataggio delle immagini. Alcune di queste, sfruttando temporaneamente le immagini acquisite in tempo reale (senza il loro salvataggio) eseguono calibrazioni che forniscono dati non ripetibili essendo impossibile acquisire immagini identiche nelle esecuzioni successive. Altri software prevedono l'utilizzo di immagini salvate su disco senza prevedere al loro interno un modo per la memorizzazione obbligando l'utente a ricorrere ad applicazioni esterne per la cattura di sequenze video da telecamere. Le applicazioni che supportano il salvataggio delle immagini, solitamente, non prevedono una preselezione di queste in base alla capacità dei fotogrammi di permettere la rilevazione dei corner. Fotogrammi che presentano un contrasto esageratamente accentuato, o catturati in ambienti dalla illuminazione scarsa oppure in cui sono presenti altri difetti (elevata sfocatura, rumore, etc.) che compromettono la *corner detection,* in genere vengono persi in fase di

calibrazione riducendo il numero delle immagini utilizzabili.
In altre parole il presente progetto sfrutta la funzione `cvFindChessboardCorners()` per salvare su disco soltanto fotogrammi in cui gli angoli della scacchiera vengono effettivamente rilevati spostando a monte dell'algoritmo di calibrazione lo scarto delle immagini non buone.
Un ulteriore vantaggio è quello di poter revisionare ogni immagine seguita dal suffisso “CORNERS” allo scopo di determinare quali immagini diano risultati poco accurati. In fase di test si è riscontrato che alcune sequenze di immagini che davano luogo a errori elevati, a valle di una discriminazione in base all'accuratezza degli angoli evidenziati, portavano a risultati accettabili.

7.5 Secondo sotto-dialogo

Il sotto-dialogo `IDD_CALIBRAZIONE_LISTA_IMMAGINI,` incluso alla pressione del tab “2-Calibrazione da Immagini” all'interno del dialogo `IDD_NUOVA_CALIBRAZIONE`, può essere concettualmente suddiviso in tre parti.
Nella parte superiore compaiono filename e path da cui sono prelevate le immagini; nella parte centrale è presente un “list view control" che visualizza un elenco di elementi con icone, mentre nella parte inferiore figurano due check box.
La classe di riferimento `SubDlgCalibrazioneListaImmagini,` subito prima di visualizzare la dialog box, carica due icone consistenti in due

piccoli cerchi di colore verde e rosso e produce i campi della lista di elementi che si possono osservare nell'illustrazione 39.

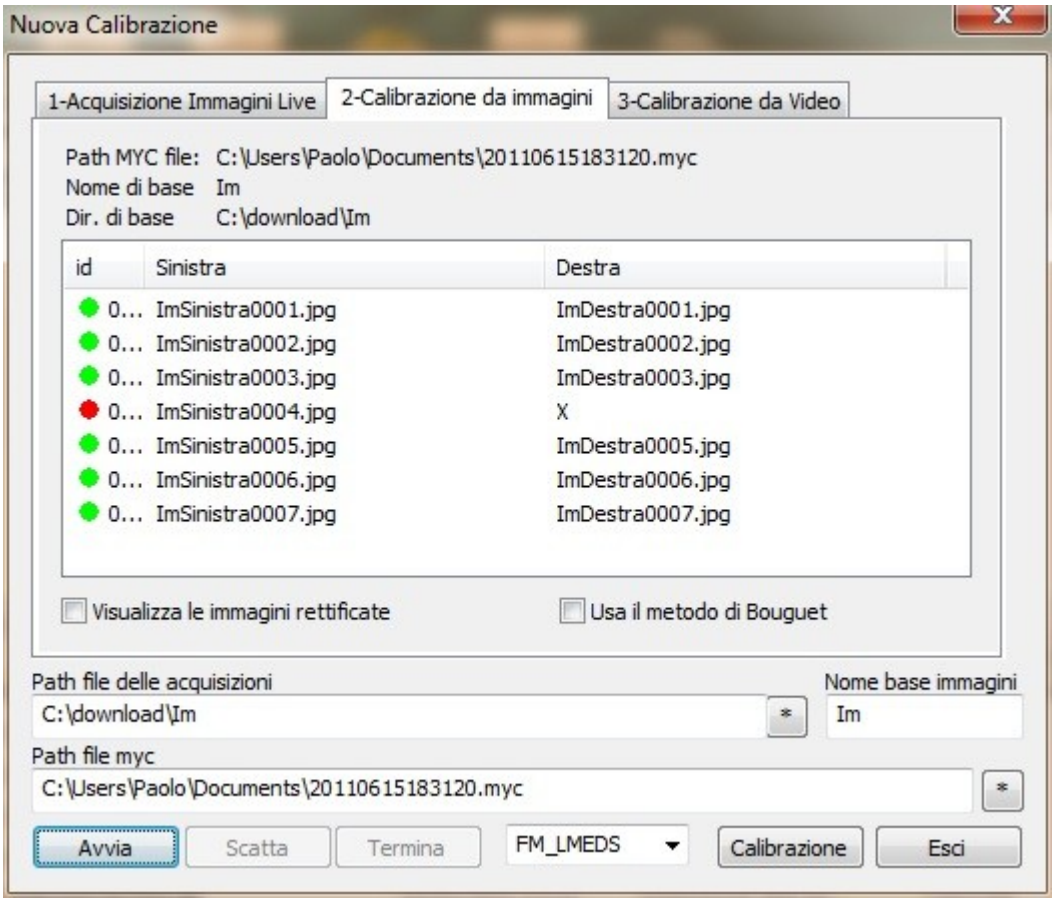


Illustrazione 39: Seconda scheda del tab control

7.5.1 Visualizzazione della lista di coppie di filename

Se l'operatore accede al secondo pannello subito dopo aver acquisito le immagini ha già inserito i dati necessari, al contrario deve inserire i parametri nel caso in cui voglia effettuare una calibrazione da immagini salvate non recentemente.

Alla pressione del tasto “Avvia” viene abilitato il pulsante “Calibrazione”,

sono poi visualizzati nomi e percorsi dei file nella parte superiore prima di chiamare la funzione `ResetContentDialog()`. Questa funzione elimina i vecchi elementi della lista e cerca i file nella directory di base specificata che rispettano i criteri imposti, riempie i campi predisposti e fornisce tre possibili risultati:

1. Le coppie corrispondono perfettamente, quindi viene restituito il numero di coppie individuate.
2. Le immagini di sinistra e quelle di destra sono differenti
3. Non sono presenti immagini nella directory specificata che rispettano i criteri immessi.

Nei primi due casi apparirà una lista in cui figurano i nomi delle immagini disposte in coppie. Ogni coppia corrisponde ad immagini catturate contemporaneamente a cui è associato uno stesso numero più la parola "Sinistra" o "Destra" per specificare la telecamera con la quale è stata catturata. Al lato di ogni coppia è presente il numero nella lista con una icona precedentemente caricata, vale a dire un pallino colorato che può essere:

- Verde se esistono sia l'immagine sinistra che destra
- Rossa se l'immagine destra non è presente ed è quindi etichettata da una "X".

7.5.2 Conferma delle singole coppie di immagini

Facendo un doppio click sul pallino si apre un nuovo dialogo per la visualizzazione della relativa coppia di immagini in un formato compresso

come nell'illustrazione 40. Non ci si deve preoccupare se la compressione riduce considerevolmente la qualità delle immagini poiché nella directory in cui verranno prelevate permangono nella loro forma originaria. Questo dialogo rende l'utente consapevole di qualche errore nel salvataggio delle immagini dovuto ad una memorizzazione esterna al programma[8] oppure da parte del software in fase di acquisizione, permettendo di cancellare mediante la pressione del tasto "Elimina coppia dalla calibrazione" tutte le coppie associate ad un pallino rosso.

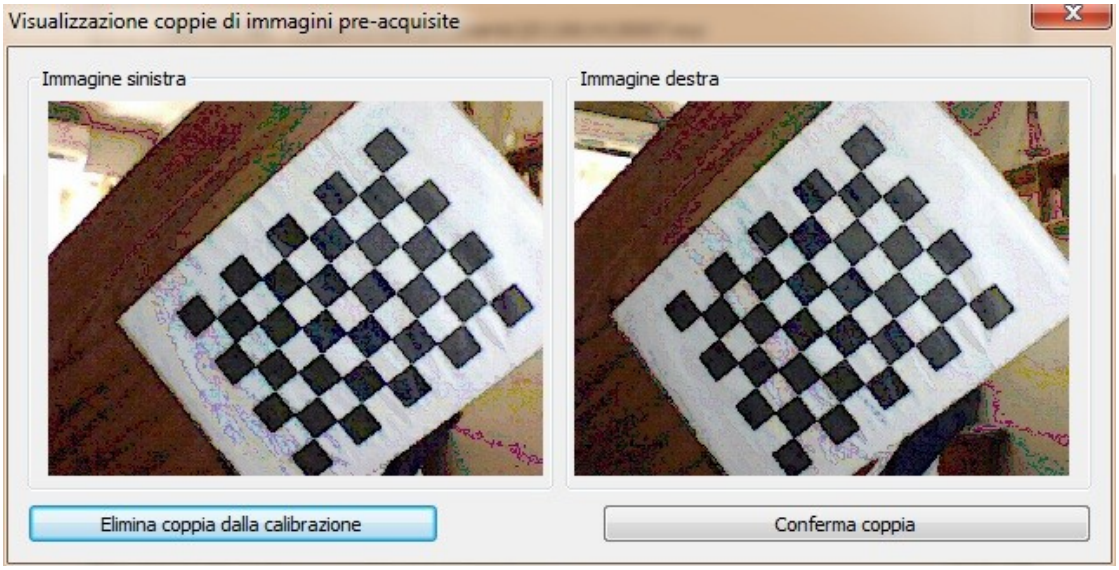


Illustrazione 40: Dialogo con identificativo IDD_VIS_IMG

Risultati poco soddisfacenti di alcune calibrazioni possono essere dovuti ad immagini di scarsa qualità che evidenziano gli angoli della scacchiera nonostante tale deficienza. In certi casi, anche se la cattura delle immagini è un'operazione abbastanza rapida, l'utente può preferire eliminare quelle che non ritiene accettabili senza doverne acquisire di ulteriori.

8 Ad esempio attraverso un copia-incolla di immagini precedentemente acquisite e spostate da un'altra directory.

Sebbene in questo dialogo potrà rimuovere le coppie non ritenute accettabili, nel caso in cui, a fronte di un esito poco soddisfacente della calibrazione, non sia sufficiente il dialogo `IDD_VIS_IMG` per riconoscere quali immagini siano di scarsa qualità può aprire la directory in cui sono state salvate le immagini ed esaminare quelle a cui corrisponde il suffisso "CORNERS" in cui sono distinguibili gli angoli colorati. Qualora l'utente voglia scartare i file in cui siano evidenziati angoli che non corrispondono perfettamente a quelli presenti sulla scacchiera, oppure nei quali vi sia una illuminazione non idonea, dovrà eliminare tutti i file che presentano numero di immagine e nome di base medesimi. Bisogna però tenere in considerazione il numero dei fotogrammi utilizzabili per la calibrazione al termine di questa operazione perché calibrazioni effettuate su poche immagini portano a risultati poco accurati.

7.5.3 Selezione delle opzioni

Una volta cancellate o confermate le immagini l'utente può scegliere se selezionare i due check-box nella parte inferiore del secondo sotto-dialogo. Il primo consente di visualizzare, una volta ultimata la calibrazione, la trasformazione di rettificazione applicata sulle immagini immesse, in modo da verificare preliminarmente la qualità della calibrazione.

La selezione del secondo check-box permette di adottare il metodo di Bouguet[9] per la rettificazione, altrimenti per default viene utilizzato il metodo di Hartley.

9 Vedi capitolo 3

L'ultima opzione concessa all'utente è il metodo con cui effettuare il calcolo della matrice fondamentale, ma è importante rendere noto che tale opzione non ha alcun effetto se è selezionato il metodo di Bouguet in quanto soltanto il metodo di Hartley sfrutta la matrice fondamentale per il calcolo della trasformazione di rettificazione.
I metodi che si possono scegliere sono[10]:

- Metodo dei sette punti indicato con "FM_7POINT".
- Metodo degli otto punti indicato con "FM_8POINT"
- Metodo RANSAC indicato con "FM_RANSAC" e impostato per default
- Metodo LmedS e indicato con "FM_LMEDS"

7.6 Stereo-calibrazione e visualizzazione dei risultati

7.6.1 Pressione del tasto "Calibrazione"

Le prime operazioni alla pressione del tasto per l'avvio della calibrazione vera e propria concernono la verifica dei parametri inseriti e delle coppie presenti nella directory, dopodiché viene creato un vettore di stringhe con i path delle immagini e vengono inizializzate le variabili booleane associate ai check-box descritti nel paragrafo precedente.
A questo punto vengono passati i parametri, il vettore di pathfile e le variabili booleane alla funzione StereoCalib() che è stata prelevata da uno

10 Nel capitolo 3 è presente una più accurata descrizione delle differenze fra i vari metodi

degli esempi campione[11] di OpenCV a cui sono state apportate delle modifiche per adattarla al software qui presentato.

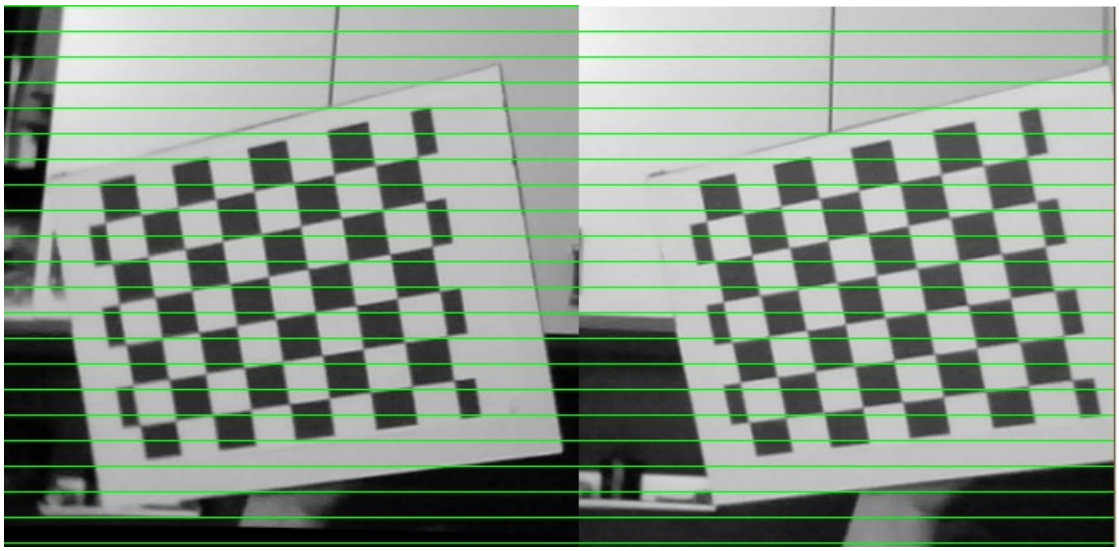


Illustrazione 41: Coppia di immagini rettificate

Nel caso in cui sia stata spuntata la voce per la visualizzazione delle immagini rettificate appariranno, come nell'illustrazione 41, coppie di immagini distorte rispetto alle originali per il processo di rettificazione, il quale fa in modo che i punti corrispondenti possano giacere su una stessa *scanline*.

Una verifica preliminare della bontà della calibrazione può essere effettuata notando che i punti corrispondenti giacciono sulla stessa riga orizzontale di pixel contrassegnata in verde. Una seconda verifica può essere applicata alla distorsione delle immagini; una leggera se pur evidente compensazione solitamente è indice di una buona calibrazione, mentre immagini eccessivamente distorte dovrebbero suggerire di catturare

11 Si tratta in particolare di stereo_calib.cpp

nuove immagini e effettuare nuovamente la calibrazione.

7.6.2 Conferma dei Risultati

Per visualizzare i risultati numerici della calibrazione si devono chiudere le finestre di dialogo sino a tornare alla finestra con i radio button "Calibrazione".

Selezionando "Ottieni i parametri da precedenti calibrazioni" è possibile aprire il dialogo "Recupero dei parametri di precedenti calibrazioni" con identificativo `IDD_SELEZIONE_CALIBRAZIONE` che si riferisce alla classe `SelezioneCalibrazione` implementata e definita negli omonimi file .cpp e .h.

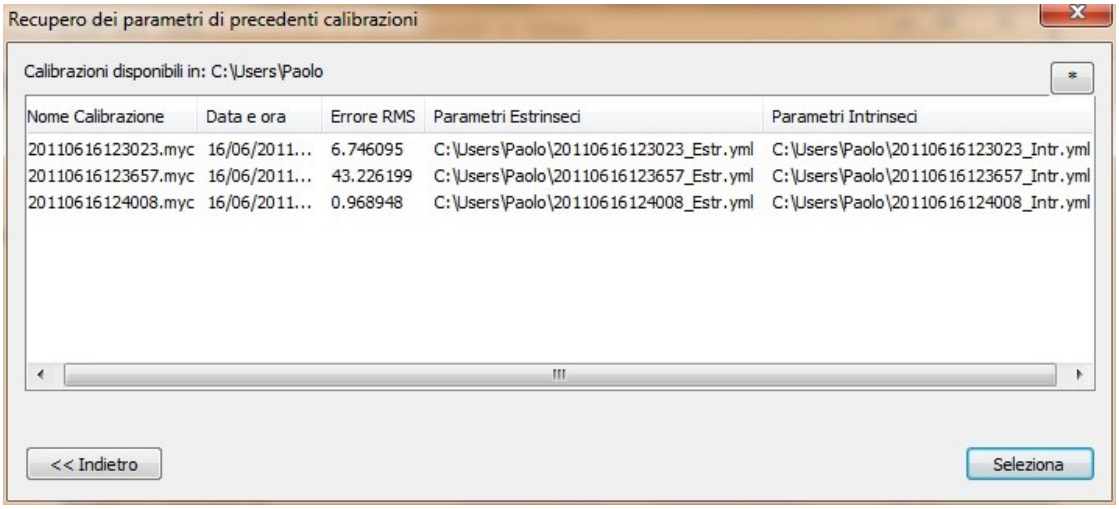


Illustrazione 42: Dialogo per la scelta dei risultati da visualizzare

Questo dialogo accede alla cartella in cui è stato effettuato l'ultimo salvataggio del file myc in cui sono riportati i risultati numerici e visualizza la lista di tutti i file di questo tipo presenti al suo interno

corredata dalle relative informazioni poste di seguito:

- Data e ora del salvataggio
- Radice dell'errore quadratico medio
- Path del file yml dei parametri intrinseci
- Path del file yml dei parametri estrinseci

I file di testo con estensione yml sono utilizzati per la raccolta dei risultati negli esempi campione OpenCV; tra questi quelli che verranno selezionati saranno poi rielaborati per essere visualizzati nella finestra di dialogo IDD_CONFERMA_CALIBRAZIONE, mostrata nell'illustrazione 43, dal titolo "Richiesta conferma scelta parametri di calibrazione" la quale fornisce una distinzione tra i vari elementi delle matrici.

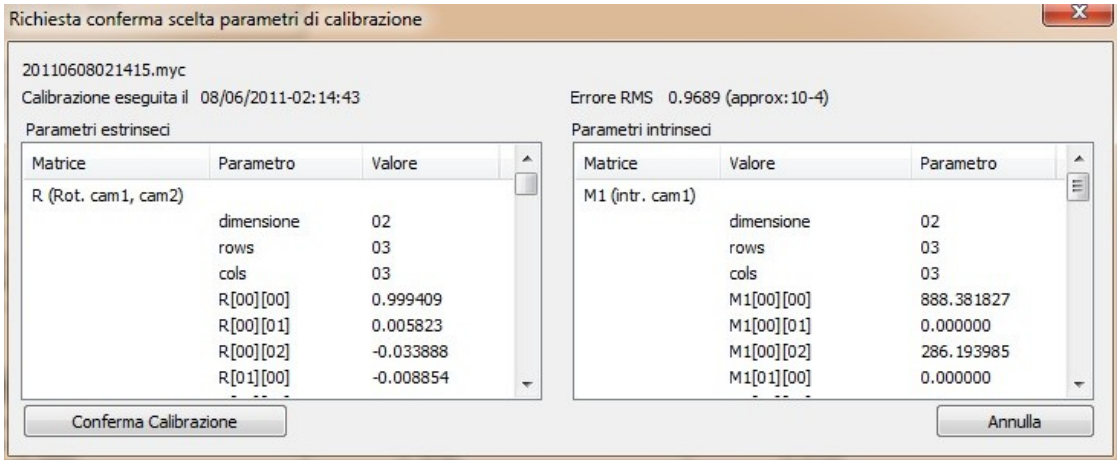


Illustrazione 43: Finestra di dialogo per la visualizzazione e l'approvazione dei parametri

La classe ConfermaCalibrazione gestisce questa finestra di dialogo e inoltre memorizza i dati selezionati prima di mostrarli.

Alla pressione del tasto "annulla" si torna al dialogo precedente, mentre alla conferma si aprirà il documento dell'illustrazione 44 caratterizzato da

una formattazione di più facile lettura, grazie al quale l'utente può visualizzare i dati.

Parametri generali	
Data calibrazione	17/06/2011-18:26:11
Basata sul file	20110617182542.myc
Errore RMS	1.441069
Larghezza immagini	640
Altezza immagini	480

Matrice intrinseci M1		
908.408	0.000	281.260
0.000	908.408	238.781
0.000	0.000	1.000

Matrice intrinseci M2		
908.408	0.000	284.309
0.000	908.408	226.566
0.000	0.000	1.000

Vettore coeff. distorsione D1							
-0.212	-1.201	0.000	0.000	0.000	0.000	0.000	-6.045

Vettore coeff. distorsione D2							
-0.057	-2.638	0.000	0.000	0.000	0.000	0.000	-10.014

Matrice di rotazone R cam1-cam2		
0.997	-0.013	0.076
0.012	1.000	0.006
-0.076	-0.006	0.997

Tr. c1. c2.
-5.282
-0.110
0.090

Angoli RPY
-0.317
4.367
0.708

Rett. cam1 (R1)		
0.998	0.008	0.059
-0.008	1.000	0.002
-0.059	-0.003	0.998

Rett. cam2 (R2)		
1.000	0.021	-0.017
-0.021	1.000	-0.002
0.017	0.003	1.000

Matr. proiez. cam1 (P1)			
749.879	0.000	262.654	0.000
0.000	749.879	233.375	0.000
0.000	0.000	1.000	0.000

Matr. proiez. cam2 (P2)			
749.879	0.000	262.654	-3962.209
0.000	749.879	233.375	0.000
0.000	0.000	1.000	0.000

Matr. disparita-profondita (Q)			
1.000	0.000	0.000	-262.654
0.000	1.000	0.000	-233.375
0.000	0.000	0.000	749.879
0.000	0.000	-0.189	0.000

Illustrazione 44: Risultati numerici relativi alla calibrazione selezionata

Le matrici M1 e M2 sono le matrici dei parametri intrinseci delle due telecamere nelle quali lungo la diagonale principale appare il valore della lunghezza focale, mentre nell'ultima colonna sono riportate le coordinate del punto principale rispetto all'origine dei pixel. Il primo elemento di questo vettore colonna è la coordinata orizzontale del punto principale, invece il secondo indica la coordinata verticale; questi due valori dovrebbero risultare prossimi alla metà dell'ampiezza e altezza di una immagine.

I vettori dei coefficienti di distorsione chiudono la parte proposta relativa ai parametri intrinseci al di sotto dei quali con la matrice di rotazione R tra le due telecamere iniziano quelli estrinseci. Questa matrice, come qualsiasi matrice di rotazione, fornisce un'informazione ridondante circa l'orientamento delle due terne in quanto i nove elementi sono legati tra loro dai sei vincoli di ortogonalità. Una rappresentazione minima per la descrizione di un orientamento è costituita da tre parametri che possono essere, come imposto nel software, i tre angoli di Eulero rollio, beccheggio e imbardata. L'utente può verificare la corrispondenza tra i tre angoli e la rotazione reale tra le due telecamere nonché l'equivalenza tra la distanza tra le stesse e il vettore di traslazione T. Subito sotto vengono mostrate le matrici di rettificazione R1 e R2, le matrici di proiezione P1 e P2 risultanti da tale trasformazione e la matrice di riproiezione Q che, permette di ottenere i punti 3D se si conoscono le coordinate sullo schermo dei corrispondenti punti bidimensionali.

Capitolo 8
Sviluppo di un'applicazione per il tracciamento di un oggetto sferico nello spazio

Una volta ultimata l'implementazione del software per la calibrazione si è deciso di testare i parametri ottenuti mediante la rilevazione di un oggetto sferico nello spazio.

8.1 Parametri della Circle Hough Transform

La tecnica utilizzata per l'estrazione delle feature è stata la trasformata di Hough con il metodo del gradiente, che è stata descritta nel capitolo 2. Questo metodo è molto utilizzato per la sua efficacia e per la complessità computazionale relativamente bassa. I parametri da scegliere con la CHT sono:

- Risoluzione dell'accumulatore
- Distanza minima tra i cerchi per essere considerati differenti
- Parametri dell'operatore di Canny
- Numero minimo di voti per un cerchio candidato
- Raggio massimo e minimo

La risoluzione dell'accumulatore è una sorta di fattore di scala rispetto alla risoluzione dell'immagine. Impostata a 1 si ha la stessa risoluzione

dell'immagine in ingresso, per valori maggiori si hanno risoluzioni per l'accumulatore inversamente proporzionali a quella dell'immagine. Una scelta di tale parametro pari a due si è dimostrata efficace, anche se non l'unica possibile per avere buoni risultati. In definitiva è stata imposta altezza e larghezza dell'accumulatore due volte più piccola rispetto all'immagine acquisita dalle telecamere.

La distanza minima tra i cerchi consente di decidere con quale "densità" vogliamo che siano scelti. In altre parole un valore troppo basso provoca il riconoscimento di cerchi multipli in un piccolo intorno di pixel, mentre un valore elevato può far perdere la rilevazione di due cerchi distinti se questi si trovano ad una distanza inferiore, come mostrato nell'illustrazione 45.

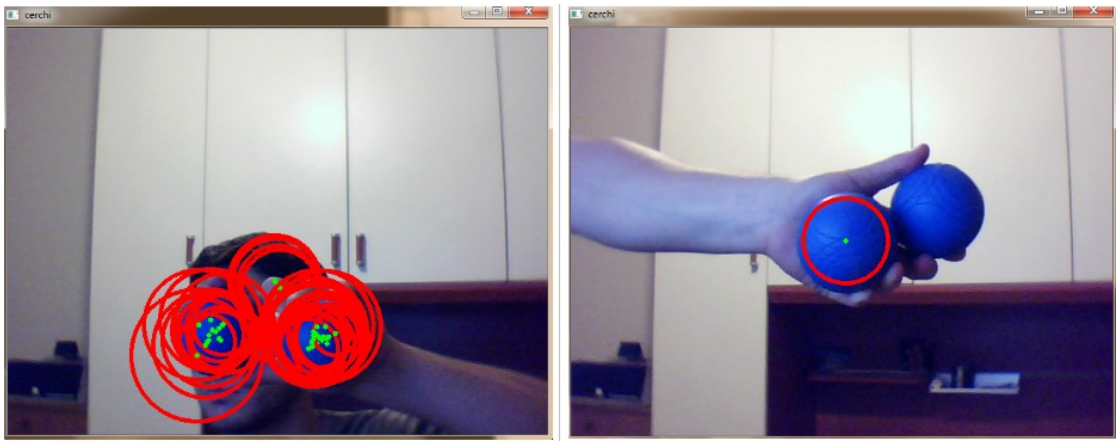


Illustrazione 45: Scelta della distanza minima tra cerchi distinti: 1) Un parametro basso porta a cerchi multipli indesiderati 2) Un parametro elevato provoca la perdita di possibili rilevazioni

Nel caso in esame si vuole rilevare una singola sfera per cui valori che in molte circostanze potrebbero risultare troppo elevati, risultano idonei in questo contesto, mentre è necessario evitare valori bassi che possono portare risultati insoddisfacenti. Si è riscontrato che valori superiori ad un

quinto dell'altezza in pixel dell'immagine danno risultati accettabili.
Un altro parametro rilevante è quello relativo alla soglia dell'accumulatore che elimina i centri candidati i cui voti non la superano. Anche in questo caso un valore troppo basso potrebbe dar luogo a false rilevazioni, ma queste possono considerarsi considerevolmente ridotte dalle restrizioni sul precedente parametro. Una soglia troppo alta può invece eliminare tutti i candidati non rilevando alcun cerchio, infatti sperimentalmente è risultato che, posti i precedenti parametri, per valori superiori a 100 si riducono considerevolmente il numero delle rilevazioni fino ad annullarsi quasi completamente oltre i 200.
Gli ultimi due parametri riguardano l'intervallo in cui facciamo cadere le possibili dimensioni del raggio. Se si considerano immagini con risoluzione 640×480 la proiezione di una sfera completamente all'interno del piano immagine non può avere un raggio superiore a 240. Valori prossimi a metà della risoluzione in altezza si verificano soltanto se l'oggetto è estremamente vicino alla telecamera, perciò si è scelto di escludere circonferenze con raggio superiore a 160. Il limite inferiore è stato posto a 20 perché valori inferiori danno luogo a false rilevazioni.

8.2 Oggetto in movimento in una scena reale

È piuttosto semplice effettuare la rilevazione in una scena a sfondo uniforme in cui sia presente un singolo oggetto statico circolare, se si ha a disposizione una funzione che utilizzi la CHT. In genere in un ambiente comune sono però presenti oggetti e forme di vario tipo, pertanto in una

scena reale raramente l'unico oggetto tondeggiante sarà il nostro target. Se la sfera è statica certamente il suo contorno costituirà un cerchio candidato con un numero di voti tra i più alti, ma non possiamo avere la stessa certezza che sia quello selezionato. La probabilità di essere scelto, se l'oggetto è in movimento, diminuisce all'aumentare della velocità. In funzione della sua dinamicità l'oggetto si presenta esteso e sfocato lungo la direzione del movimento, con una evidente diminuzione della nitidezza. La scelta del range del raggio tra 20 e 160 comporta una distanza del soggetto inferiore ai due metri; la vicinanza al centro ottico comporta maggiori velocità angolari e quindi un maggiore effetto di mosso. Nell'illustrazione 46 è evidente la perdita di nitidezza della sfera in movimento; come si può notare, la rilevazione è avvenuta comunque con successo grazie all'impiego di filtri che saranno descritti in seguito.

8.3 Utilizzo di filtri di tonalità

Garantire la rilevazione dello stesso oggetto per entrambe le immagini in un sistema stereoscopico non è un problema banale. Per tale motivo si è deciso di utilizzare oggetti (sfere) facilmente rilevabili con semplici feature detector (CHT). Per agevolare l'object detection anche nel caso di soggetto in movimento sono state utilizzare delle sfere di colore uniforme che permettono l'utilizzo di appositi filtri [Chen-2010].

Nelle immagini in scala di grigi l'unico attributo necessario a caratterizzare in ogni punto l'immagine è l'intensità, che si traduce nel valore di grigio del corrispondente pixel, mentre la rappresentazione delle immagini a

colori prevede l'impiego di tre matrici di intensità, una di rosso, una di verde, una di blu. In altre parole l'intensità da sola non è sufficiente a caratterizzare l'immagine, in quanto occorre anche esprimere la crominanza costituita dalla tonalità (o tinta, hue in inglese) e dalla saturazione. Per poter discriminare oggetti di differente colore, bisogna agire su questi due attributi dei quali il primo fornisce un indice della purezza del colore mentre il secondo può essere associato alla lunghezza d'onda.

In questa applicazione sono state utilizzate sfere di colore rosso, verde , blu e giallo cercando di sviluppare dei filtri che consentissero di rilevare soltanto la palla del colore scelto. Poiché i quattro colori sono caratterizzati da saturazioni abbastanza elevate, si è scelto di applicare delle soglie unicamente sulla tonalità.

Nel modello RGB ogni colore è definito mediante le intensità dei tre primari che lo compongono, ma la modifica indipendente delle tre componenti RGB cambia la crominanza di ogni pixel, introducendo una distorsione cromatica. Nella classificazione basata sul colore risulta quindi più efficace utilizzare un modello del colore in cui la luminanza è separata dalla crominanza.

Il modello utilizzato è quello HSV (tonalità-saturazione-valore), mostrato nell'illustrazione 46 mediante la sua rappresentazione cilindrica. Tale modello in OpenCV è caratterizzato da una componente di tonalità memorizzata come un valore intero a 8 bit tra 0 e 179, con una certa perdita di risoluzione sul colore. La saturazione e il valore dell'intensità

sono invece compresi tra 0 e 255. In particolare mentre in altri software una saturazione massima equivale a puro bianco, in OpenCV corrisponde sempre a colori brillanti [Emami-2010].

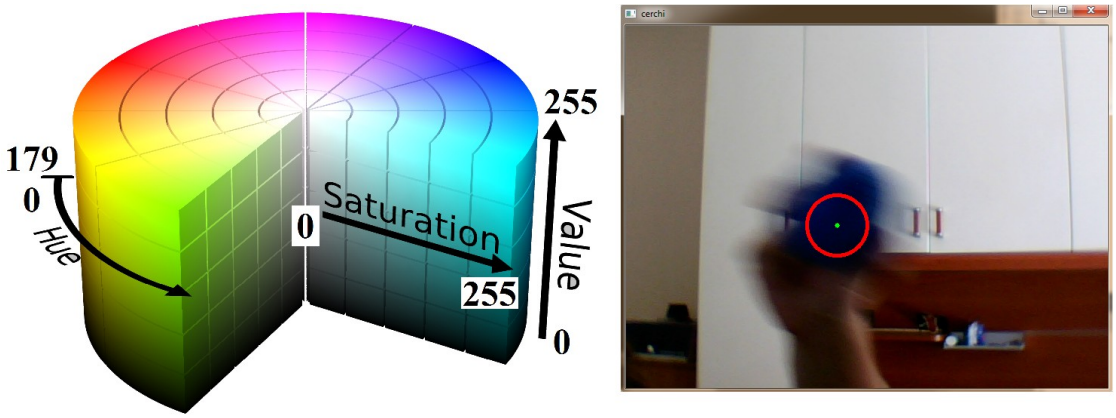


Illustrazione 46: Rappresentazione cilindrica del modello HSV (a sinistra); Rilevazione del target a dispetto dell'evidente effetto di mosso (a destra).

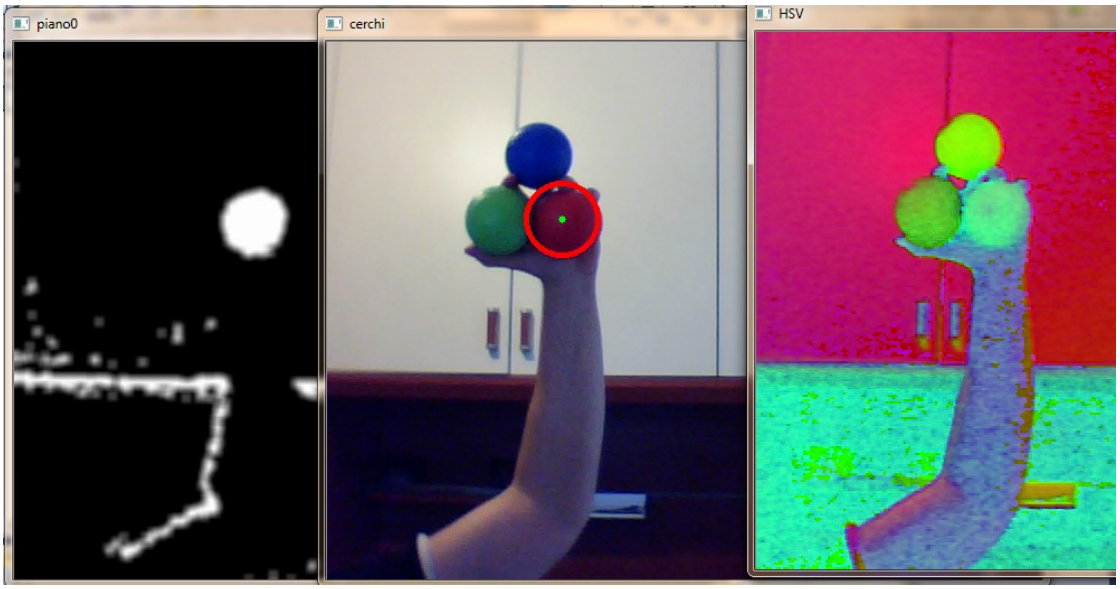


Illustrazione 47: La finestra centrale è quella originale. A destra la conversione nel modello HSV. A sinistra il risultato dopo le operazioni di sogliatura, erosione, dilazione e smoothing Gaussiano.

Nell'immagine di destra dell'illustrazione 47 si può vedere come appare una immagine convertita in HSV a partire dal modello RGB.

Per selezionare la singola tonalità è stata apportata una scissione nelle tre componenti e successivamente sono state applicate una soglia minima e una soglia massima sulla singola componente H; ad esempio per isolare il colore blu la soglia minima è stata scelta pari a 150 e la massima pari a 200 in modo che $150 < H < 200$.

Se il valore della componente Hue per lo sfondo originale non si trova nel range selezionato, si avrà uno sfondo nero nell'immagine in output, mentre i pixel aventi tinta in quell'intervallo rimarranno in scala di grigi, perciò solitamente il filtro sulla tonalità non isola completamente l'oggetto desiderato, ma tutti i pixel della medesima tinta.

A questo punto la singola componente viene convertita in scala di grigi e poi passata alla trasformata di Hough per i cerchi, la quale, se non ci sono macchie di quel colore o altri oggetti sferici della tonalità desiderata, dovrebbe catturare il target, tuttavia non è infrequente riscontrare rilevazioni di pixel nell'intervallo isolato anche nella posizione di oggetti di altra tonalità. In altre parole in corrispondenza di un oggetto a tinta uniforme differente dal colore scelto possono comparire pixel di quella tonalità a causa del rumore e delle condizioni di luminosità.

La sogliatura produce quindi delle zone simili a macchie che possono facilmente essere identificate come cerchi di piccolo raggio.

Per eliminare questi gruppi di pixel è stato applicato all'immagine uno smoothing Gaussiano e una operazione di apertura, vale a dire una erosione seguita da una dilazione [Gonzalez-2002].

Queste operazioni oltre a diminuire i pixel indesiderati rendono anche di

forma più tondeggiante l'immagine mossa, quando l'oggetto sferico è in movimento. L'immagine di sinistra nell'illustrazione 47 mostra l'effetto di questa procedura effettuata sulla tonalità del rosso, mentre al centro è presentata l'immagine originale su cui è stato applicato il cerchio rosso risultante dalla CHT.

8.4 Estrazione delle coordinate 3D

Il centro del cerchio rilevato con la CHT ci fornisce le coordinate della proiezione dell'oggetto sferico sui due piani immagine. La conoscenza di questi due valori unita a quella dei parametri ricavati dalla calibrazione ci permette di estrarre le coordinate del target nello spazio.

Siano ad esempio $(x_{1,} y_1)$ e $(x_{2,} y_2)$ le due coordinate estratte dalla CHT, il primo passo da effettuare è quello di portarle in coordinate omogenee:

$$p_1 = \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} \quad \text{e} \quad p_2 = \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix}$$

È possibile normalizzare questi vettori rispetto alle matrici dei parametri intrinseci calcolate con la calibrazione:

$$p_{1\text{norm}} = M_1^{-1} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} \quad \text{e} \quad p_{2\text{norm}} = M_2^{-1} \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix}$$

Per le coordinate ottenute vanno compensati gli effetti della distorsione (2.13) ottenendo i vettori $p_{1\text{undist}}$ e $p_{2\text{undist}}$. Applicando le rotazioni inverse rispetto ad R_1 ed R_2 si rendono complanari i due piani immagine ed infine si moltiplicano le nuove matrici dei parametri intrinseci che si

possono recuperare dalle prime tre colonne delle matrici P_1 e P_2 per i vettori appena ricavati. Dopo tali operazioni abbiamo ottenuto le coordinate 2D dei centri nelle immagini rettificate che quindi avranno stessa ordinata h:

$$p_{1\text{rect}} = \begin{pmatrix} x'_1 \\ h \\ 1 \end{pmatrix} \quad \text{e} \quad p_{2\text{rect}} = \begin{pmatrix} x'_2 \\ h \\ 1 \end{pmatrix}$$

Sfruttando ora la matrice di riproiezione

$$Q = \begin{bmatrix} 1 & 0 & 0 & -\hat{X}_{01} \\ 0 & 1 & 0 & -\hat{Y}_0 \\ 0 & 0 & 0 & f_{new} \\ 0 & 0 & -1/\hat{T} & \dfrac{\hat{X}_{01} - \hat{X}_{02}}{\hat{T}} \end{bmatrix}$$

nella quale sono presenti i parametri proposti nella (3.47) e nella (3.48), è possibile ricavare le coordinate 3D (X, Y, Z) dell'oggetto dal risultato della seguente espressione

$$\begin{bmatrix} \lambda X \\ \lambda Y \\ \lambda Z \\ \lambda \end{bmatrix} = Q \begin{bmatrix} x'_1 \\ h \\ x'_1 - x'_2 \\ 1 \end{bmatrix}$$

normalizzando per il fattore di scala λ .

8.5 Verifica dell'accuratezza dei risultati

Le otto matrici e i quattro vettori ottenuti con la calibrazione hanno lo scopo di mettere in relazione l'ambiente tridimensionale con quello bidimensionale delle immagini. Verificare empiricamente la veridicità

delle stime dello spazio 3D, effettuate nell'applicazione, assicura indirettamente l'attendibilità dei valori forniti dalla calibrazione.
Se si conosce la grandezza del raggio della circonferenza la stima della profondità risulta calcolabile anche con una singola telecamera applicando delle proporzioni tra le dimensioni del raggio in due e in tre dimensioni. Se non si conosce la misura del raggio il discorso cambia completamente, infatti soltanto nel caso di visione stereoscopica non c'è bisogno di questa informazione preventiva per stimare la profondità; inoltre si può addirittura calcolare la misura del raggio nello spazio a meno di un piccolo margine di errore.
Una volta appreso l'algoritmo per la triangolazione il metodo per estrarre la dimensione reale del raggio è piuttosto banale. La conoscenza sulle due immagini delle dimensioni del cerchio e del suo centro permette il calcolo di un secondo punto che arbitrariamente poniamo a destra lungo la scanline a una distanza pari al raggio. Con un procedimento analogo a quello dei centri calcoliamo la posizione nello spazio di questi due nuovi punti e calcoliamo la distanza euclidea con le coordinate 3D dei centri. La verifica manuale, con uno strumento di misura per lunghezze delle dimensioni reali della sfera, consente di quantificare l'errore.
Sebbene il sistema permetta la rilevazione della sfera anche a velocità elevate, una posizione statica facilita l'accertamento della fedeltà delle coordinate spaziali non essendo necessario conoscere l'istante in cui una specifica posizione è stata occupata. Ponendo l'oggetto in una posizione conosciuta si effettuano tutti i passi descritti fino alla triangolazione e poi

si constata se le coordinate estratte sono prossime a quelle attese. In realtà è da far presente che seppure l'oggetto è fermo ogni fotogramma non sarà esattamente identico al precedente, soprattutto a causa del rumore e delle condizioni di illuminazione che non sono costanti. Questo aspetto determina un cambiamento della sua posizione ad ogni istante, seppure di pochi pixel, come si può vedere dall'illustrazione 48.

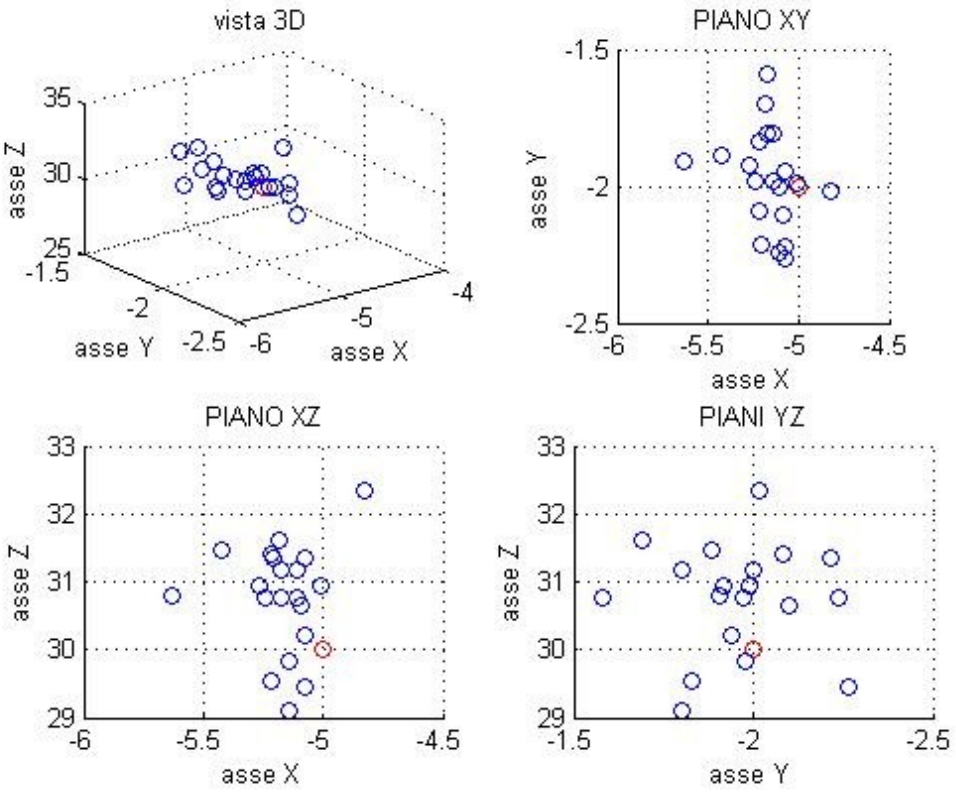


Illustrazione 48: Confronto tra posizione misurata (in rosso) e posizioni stimate (in blu)

Questo movimento apparente genera un errore anche qualora la calibrazione fosse estremamente accurata. Se l'oggetto è fermo la posizione stimata permane comunque entro un certo intervallo, perciò si può aggirare il problema facendo la media delle posizioni occupate.

Nell'esempio in figura la posizione conosciuta in cm è X=-5, Y=-2 e Z=30 in modo che la sfera si trovi rispetto all'asse X al centro tra le due telecamere distanti tra loro circa 10 cm. Il raggio calcolato vale 3.52cm approssimativamente pari al valore reale, mentre la radice dell'errore quadratico medio calcolata sui tre assi è (0.23, 0.18, 1.11) cm.

I vari test hanno mostrato una proporzionalità tra la profondità e il RMSE calcolato su di essa, ad esempio ad una profondità di 44cm l'RMSE è risultato pari a 2.477, mentre è arrivato a 5.91 ad una profondità di 85 cm. L'algoritmo riesce a rilevare la sfera con successo fino a distanze di circa un metro, al di là del quale il raggio della sfera arriva al di sotto della soglia fissata con la trasformata Hough. Una soglia più bassa aumenterebbe in maniera non trascurabile la probabilità di false rilevazioni.

Capitolo 9
Conclusioni e sviluppi futuri

9.1 Conclusioni

In questa tesi è stato descritto il progetto e la realizzazione di un sistema di visione stereoscopica. In particolare è stato sviluppato un software di calibrazione che prevede un'interfaccia grafica orientata all'utente con l'esecuzione automatica della maggioranza dei passi da effettuare.

È stata inoltre implementata una applicazione per il riconoscimento automatico di oggetti sferici in movimento. L'object detection permette di estrarre le coordinate bidimensionali dell'oggetto e in seguito calcolarne la posizione nello spazio mediante rettificazione e triangolazione.

La tesi fornisce inoltre un'accurata descrizione degli algoritmi di image processing, di calibrazione e di ricostruzione della geometria epipolare per la rettificazione.

Il primo obiettivo raggiunto in questo progetto è stato l'ideazione e l'implementazione di un algoritmo che permettesse la memorizzazione automatica di fotogrammi acquisiti simultaneamente mediante l'utilizzo della programmazione parallela.

Inoltre questo algoritmo è stato completato con un sistema che consente la memorizzazione automatica di coppie di immagini che rispettano il requisito di permettere l'estrazione, anch'essa automatica, degli angoli nel pattern di calibrazione.

Il software di calibrazione sviluppato facilita l'utente non solo nella misura dei parametri intrinseci ed estrinseci, ma anche nella sistemazione del sistema stereoscopico, sottolineando quali parametri non sono adatti ad un impianto funzionale.
Una volta conclusa la parte relativa alla calibrazione si è rivolto l'interesse al riconoscimento di un oggetto sferico nello spazio di dimensioni ignote e nell'estrazione delle sue coordinate spaziali, nonché del suo raggio.

9.2 Sviluppi futuri

Il software di calibrazione potrebbe essere utilizzato in tutte quelle applicazioni che ne utilizzano i parametri per la ricostruzione dell'ambiente, soprattutto in quelle applicazioni in cui è desiderabile modificare la posizione relativa tra le telecamere.
L'intero apparato potrebbe essere montato su un robot mobile per l'inseguimento di oggetti oppure potrebbe essere ulteriormente sviluppato per portare un manipolatore in una opportuna configurazione di presa. La traiettoria pianificata potrebbe essere eseguita utilizzando le misure visuali a ciclo aperto utilizzando un approccio look-and-move oppure in retroazione mediante un controllo visuale, calcolando il vettore di errore tra le pose dell'organo terminale e quello dell'oggetto. Per come è stato progettato il sistema, quest'ultimo approccio potrebbe essere più facilmente eseguito nella modalità eye-in-hand in modo che non sia necessario identificare anche l'organo terminale del manipolatore. Inoltre, facendo muovere il sistema stereoscopico in maniera solidale all'organo terminale,

si avrebbe un notevole incremento nell'accuratezza della misura in prossimità del target.

Le lacune del sistema si possono trovare soprattutto nel sistema di riconoscimento degli oggetti che permette di rilevare soltanto oggetti sferici e che è molto influenzato dal rumore e dai cambiamenti di luminosità e per questo presenta problemi di falsi accoppiamenti.

Bibliografia

[Ballard-79] D. H. Ballard, **Generalizing the Hough transform to detect arbitrary shapes**. Computer Science department, University of Rochester, Ottobre 1979

[Bouguet] J. Y. Bouguet, **Camera Calibration Toolbox for Matlab**, http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.

[Bradski-2008] G. Bradski, A. Kaehler. **Learning OpenCV: Computer Vision with the OpenCV Library**, O'Reilly Media, 2008.

[Brown-66] D. C. Brown. **Decentering Distortion of Lenses**. Photometric Engineering 32(3) : 444–462, 1966.

[Brown-71] D. C. Brown. **Close-range camera calibration.** Photogrammetric Engineering, 37(8):855–866, 1971.

[Canny-83] J. F. Canny. **Finding edges and lines in images.** Master's thesis, MIT. AI Lab. TR-720, 1983.

[Caprile-90] B. Caprile and V. Torre. **Using vanishing points for camera calibration**. International Journal of Computer Vision, 4:127--140, 1990.

[Chen-2010] Lior Chen. **Fast object tracking using the OpenCV library**. http://www.lirtex.com/robotics/fast-object-tracking-robot-computer-vision

[Cyganek-09] B. Cyganek, P. Siebert. **An Introduction to 3D Computer Vision Techniques and Algorithms**. Wiley 2009

[Davies-2005] Davies, E. Roy (2005). **Machine Vision: Theory, Algorithms, Practicalities** (3rd ed.)

[Di Stefano] L. Di Stefano, http://didattica.arces.unibo.it/file.php/59/Elaborazione_dellImmagine_L-S/Manuali/tecnologie_3.pdf, Università di Bologna, Didattica Arces, Corso di Elaborazione dell'Immagine L-S, materiale didattico

[Duda-72] R. O. Duda and P. E. Hart. **Use of the Hough transformation to detect lines and curves in pictures.** Communications of the Association for Computing Machinery 15 (1972): 11–15.

[Emami-2010] Shervin Emami.
http://www.shervinemami.co.cc/colorConversion.html

[Euclide] Euclide, **Elementi**, IV - III secolo a.C.

[Faugeras-01] O. Faugeras, Q. Luong. **The Geometry of Multiple Images**. MIT Press (Marzo 2001).

[Faugeras-93] Olivier Faugeras. Three-dimensional computer vision: a geometric viewpoint. MIT Press, Cambridge, 1993.

[Fusiello-2010] A. Fusiello, **Visione Computazionale**. Appunti delle lezioni, 2010.

[Gonzalez-2002] R. C. Gonzalez, R. E. Woods, **Digital Image Processing**, Prentice Hall, 2002.

[Hartley-97] R. I. Hartley. **In defense of the eight-point algorithm**. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(6): 580-593, Ottobre 1997.

[Hartley-98] R. I. Hartley. **Theory and practice of projective rectification**. International Journal of Computer Vision, to appear, 1998.

[Hartley-2003] R. Hartley and A. Zisserman. **Multiple View Geomerty in Computer Vision**. Cambridge University Press, second edition, 2003.

[Hough-59] P. V. C. Hough. **Machine analysis of bubble chamber pictures**. Proceedings of the International Conference on High Energy Accelerators and Instrumentation. (pp. 554–556), 1959.

[Isgrò-99] F. Isgro, E. Trucco. **Projective rectification without epipolar geometry**. In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp.94-99, 1999.

[Koch-Tononi-2008] C. Koch, G. Tononi, **Can Machines Be Conscious? Yes - and a new Turing test might prove it**. IEEE Spectrum, Vol.
, n. 6, pp. 54-59, giugno 2008.

[Lei-99] B. J. Lei, E. A. Hendriks, M. J. T. Reinders. **On Feature Extraction from images.** Technical report on "inventory properties" for MCCWS, 1999.

[LonguetHiggins-81] H. C. Longuet-Higgins. **A computer algorithm for reconstructing a scene from two projections.** Nature 293: 133-135 (10 Settembre 1981)

[Lowe-2004] David G. Lowe. **Distinctive Image Features**

from Scale-Invariant Keypoints. International Journal of Computer Vision, Gennaio 2004.

[Moravec-79] Moravec, H. (1979). **Visual mapping by a robot rover**. In Proceedings of the 6th International Joint Conference on Articial Intelligence, pp. 599-601, Agosto 1979.

[Otha-2007] **Smart CMOS image sensors and applications**, CRC Press, Taylor & Francis Group, New York 2007

[Rousseeuw-87] P. J. Rousseeuw. **Robust Regression and Outlier Detection. Wiley**, New York, 1987.

[Scaramuzza-2003] D. Scaramuzza. **Progetto e realizzazione di un sistema di visione stereoscopica per la robotica, con applicazione all'inseguimento di corpi in moto e all'autolocalizzazione.** Tesi di laurea in ingegneria elettronica presso l'università degli studi di Perugia. A.A. 2003 2004

[Siciliano-2008] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo. **Robotica, Modellistica, pianificazione e controllo**. McGraw-Hill Companies. 2008

[K. R. Spring] K. R. Spring, T. J. Fellers, M. W. Davidson. **Introduction to Charge-Coupled Devices (CCDs)**. http://www.microscopyu.com/articles/digitalimaging/ccdintro.html

[Trucco-98] E. Trucco, A. Verri, **Introductory Techniques To 3d Computer Vision**, Prentice Hall, 1998

[Tsai-87] Roger Y. Tsai. **A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision**

Metrology Using Off-the-shelf TV Cameras and Lenses. IEEE journal of robotics and automation, vol. ra-3, no. 4, Agosto 1987

[UTET-91] **Grande Dizionario Enciclopedico**, Unione Tipografico Editrice Torinese, 1991

[Xie-2003] Ming Xie. **Fundamentals of Robotics, Linking Perception to Action**. Series in Machine Perception and Artificial Intelligence - Vol. 54. Singapore-MIT Alliance & Nanyang Technological University, Singapore 2003

[Zhang-96] Z. Zhang. **Determining the Epipolar Geometry and its Uncertainty: A Review**. Institut national de recherche en informatique et en automatique. N° 2927. Luglio 1996

[Zhang-98] Z. Zhang. **A Flexible New Technique for Camera Calibration**. Technical Report, MSR-TR-98-71, Microsoft Research, December 2, 1998.

ALTRI SITI WEB:

- http://opencv.willowgarage.com
- http://sourceforge.net/projects/opencvlibrary/
- http://en.wikipedia.org
- http://opencvlibrary.sourceforge.net/
- http://tech.groups.yahoo.com/group/OpenCV/
- http://pr.willowgarage.com/wiki/OpenCV