ALMA MATER STUDIORUM UNIVERSITA' DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA Sede di Forlì

Corso di Laurea in INGEGNERIA AEROSPAZIALE

Classe 9234 L-9

ELABORATO FINALE DI LAUREA In:

Satelliti e missioni spaziali:

STUDIO DI FATTIBILITÀ DELLA MISURAZIONE DI ERRORE DI PUNTAMENTO DI UN'ANTENNA TERRESTRE ATTRAVERSO TECNICHE DI STAR TRACKING

Candidato: Vassilios Silaidis

Relatore: Prof. Alfredo Locarini

Correlatore: Ing. Giacomo Curzi

Anno Accademico 2019-2020

A mia Madre…

Sommario

Nomenclatura e Simboli

VHF/UHF	*Very High Frequency/Ultra High Frequency;*
CCD	*Charge-Coupled Device*
CMOS	*Complementary Metal-Oxide Semiconductor;*
AIM	*attitude estimation using image matching*
RF	*Radio Frequency*
$\boldsymbol{q}$	*quaternione;*
$\boldsymbol{b}$	*Generico vettore colonna;*
$A^T A$	*tensore struttura dell'immagine nel punto p;*
$P_1, P_2 \ldots . P_n$	*pixel all'interno della finestra di calcolo;*
$I_x(P_n), I_y(P_n), I_t(P_n)$	*derivate parziali dell'immagine I rispetto alla posizione x, y e tempo t,*
$w(x, y)$	*funzione finestra che esalta gli spostamenti più vicini al pixel centrale.;*
W	*matrice diagonale nxn che contiene i pesi* $W_{ii} = w_i$*;*
(x, y),	*coordinate delle stelle nel piano immagine;*
$(\mathrm{x}_0, \mathrm{y}_0)$	*intersezione tra il piano focale e l'asse ottico;*
(i, j, k)	*vettore unitario della stella osservata;*
FOV	*FIELD OF VIEW della camera (γ).*
$\mathrm{t_x}$	*distanza (pixels) di cui le stelle del database vengono traslate nella direzione x*
$\mathrm{t_y}$	*distanza (pixels) di cui le stelle del database vengono traslate nelle direzione y;*
$\mathrm{q_{dat}}$.	*quaternione d'assetto dell'immagine in database.;*
$\mathrm{u_i}$ *i=1,2,3*	*Componenti vettore u;*
$\mathrm{v_i}$ *i=1,2,3*	*Componenti vettore v;*
$\mathrm{w_i}$ *i=1,2,3*	*Componenti vettore w;*
$\mathrm{v_r(i_r, j_r, k_r)}$	*vettore unitario dopo la rotazione da parte dell'inverso di* qdat*;*
$\mathrm{q_a}$	*quaternione di assetto reale;*
$\mathrm{q_e}$	*quaternione della stima d'assetto dello scatto;*
b *e* l	*valori massimi che le coordinate dei pixel possono avere nel piano focale;*
ϕ	*angolo di rollio (roll angle);*
ψ	*Angolo di imbardata (yaw angle);*
θ	*Angolo di beccheggio (pitch angle);*

Elenco Figure

Introduzione.

Il lavoro condotto in questa tesi si basa sullo studio di fattibilità della misurazione di errore di puntamento di un'antenna terrestre. Per raggiungere lo scopo sono state utilizzate tecniche di star tracking, necessarie alla determinazione d'assetto di una camera (o, in generale, di un satellite), e un sistema di puntamento e riconoscimento di un fascio laser.

Dagli anni 70', con gli sviluppi che interessarono l'osservazione astronomica, sono stati introdotti i primi satelliti dotati di sistemi ottici per il riconoscimento della volta celeste. Ad oggi l'utilizzo di star tracker come metodo di determinazione d'assetto di un satellite in orbita è considerato affidabile, accurato ed è fra i più diffusi [1][2]. I dati rilevati da questi sistemi possono essere comunicati a terra alle stazioni di controllo, consentendo di effettuare il monitoring dell'assetto con cadenza di tempo regolare.

Ground Segment:

Il segmento di terra (GroundSegment) consente la gestione di un veicolo spaziale e la distribuzione di dati scientifici tra le parti a terra. Si divide tipicamente in:

1. Flight Operation Segment, responsabile dello scambio reciproco di informazioni necessarie al funzionamento di un satellite (TT&C: Tracking, Telemetry & Telecommand).
2. Payload Data Segment, per la ricezione e distribuzione dei dati scientifici. [3]

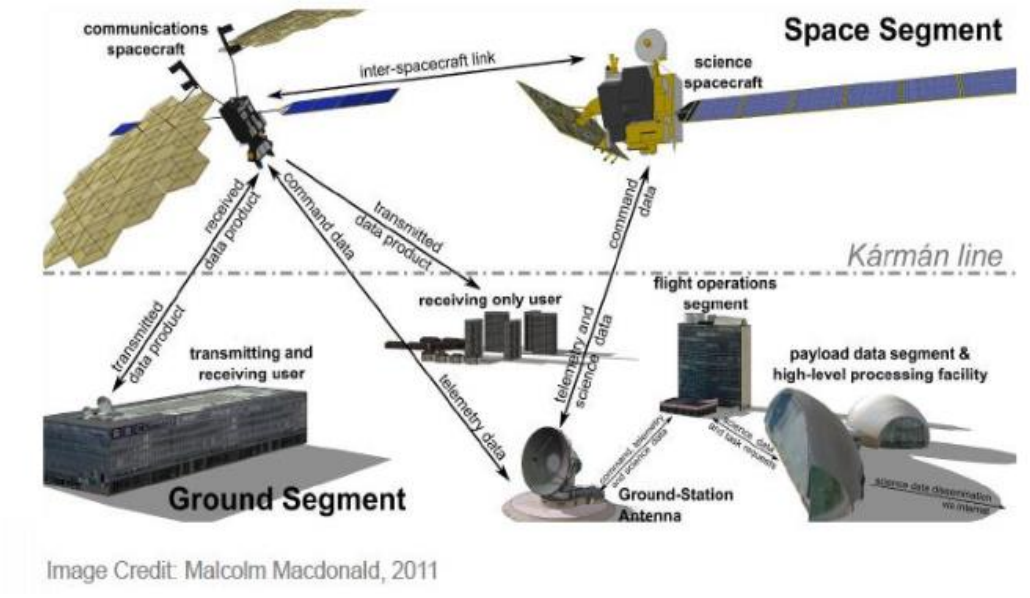


FIGURA 1: ARCHITETTURA DI UN GROUND SEGMENT IN COMUNICAZIONE CON IL FLIGHT OPERATION SEGMENT E GLI USERS.

Gli elementi principali di un Ground Segment sono:

1. Stazioni di terra, che forniscono interfacce radio con veicoli spaziali;
2. Centri di controllo missione, da cui vengono gestiti veicoli spaziali;
3. Link di terra, che collegano gli elementi di terra l'uno all'altro;
4. Terminali remoti, utilizzati dal personale di supporto;
5. Impianti di integrazione e test;
6. Strutture di lancio;

Nelle stazioni di terra, l'antenna e gli equipaggiamenti di trasmissione/ricezione sono gli elementi fondamentali del canale di comunicazione fra terra e gli strumenti in orbita. È importante garantire dei requisiti di puntamento per l'antenna. Infatti, tutte le antenne non omnidirezionali (solitamente utilizzate in questo ambito) necessitano di essere accuratamente "puntate" nella direzione desiderata di trasmissione/ricezione attraverso opportune procedure di puntamento; tanto più è efficiente il puntamento tanto maggiori sono le prestazioni del sistema di radiocomunicazione in termini di potenza elettromagnetica trasmessa/ricevuta e quindi anche sulla qualità del segnale in termini di rapporto segnale/rumore. [4] [6]

I recenti sforzi dell'industria aerospaziale per fornire un accesso a basso costo allo spazio hanno portato allo sviluppo sempre più diffuso di nano-satelliti. Tipicamente le comunicazioni spazio-terra nell'ambito di nano-satelliti sono effettuate in bande VHF/UHF in cui i requisiti hardware (come la precisione di puntamento dell'antenna) non sono critici come per le frequenze più alte. Nelle bande VHF/UHF, una stazione di terra affidabile ed economica può essere integrata utilizzando hardware e software radio amatoriali commerciali. Rispetto ai tipici sistemi radio VHF/UHF sui nano-satelliti, la comunicazione in banda S si sta affermando in questo campo poiché offre miglioramenti alla quantità dei dati scaricabili sfruttando la maggiore frequenza. Quando però si aumenta la frequenza, diventano più stringenti i requisiti di puntamento, per questo motivo è necessaria una più accurata calibrazione del sistema di puntamento.[7] Serve quindi un sistema di misurazione in grado verificare che sia rispettato il requisito misurando l'errore di puntamento, ed è in questo frangente che si inserisce la tesi.

Pointing Budget:

Gli errori di puntamento di un'antenna possono scaturire da molteplici sorgenti. In questa sezione analizziamo quali fra questi sono di interesse specifico per la misurazione in oggetto.

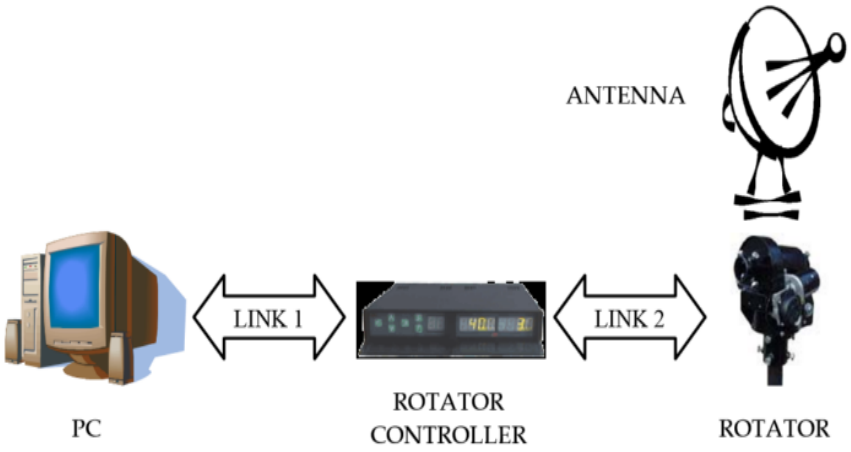


FIGURA 2: ARCHITETTURA DI UN SISTEMA DI PUNTAMENTO PER ANTENNA.

I componenti principali dell'architettura di un sistema di puntamento per antenna sono: un calcolatore, un rotore, un'antenna e il controllore del rotore. A questi andranno poi aggiunti il sistema di puntamento laser e la camera esterna per completare il sistema di misurazione proposto e stimare le prestazioni di puntamento.

Con riferimento alla Figura 2, gli errori durante un tracciamento di satellite possono essere dovuti a molti fattori [5]:

1. PC:

   a. algoritmo matematico per il tracciamento satellitare;
   b. parametri di bassa qualità/ vecchia orbita satellitare (TLE);
   c. bassa qualità del riferimento del tempo del PC;

2. Link 1:

   a. ritardi di trasmissione tra il PC e il controller del rotore.

3. Controllore del rotore:

   a. Algoritmo per il controllo del rotore;
   b. precisione/risoluzione della misurazione della rotazione del rotore
   c. ritardo connesso all'avviamento e all'arresto dei motori per la rotazione (errore dinamico);
   d. risonanza nel controllo dei rotatori.
   e. deviazione dovuta ad errori degli encoder assoluti del rotore (direzione letta dal driver e quella effettiva differiscono);

4. Link 2:

   a. ritardi nella trasmissione tra il rotore e il controller del rotore;
   b. calibrazione della posizione zero, calibrazione dell'angolo di rotazione;

5.Antenna:

   a. vibrazioni dovute al vento;
   b. vibrazioni del rotore;
   c. inerzia del sistema;
   d. deviazione elettromagnetica dovuta a presenza di elementi metallici indesiderati;
   e. deviazione dovuta ad assemblaggio impreciso dell'antenna (esempio RF-Feed non allineato);

Per garantire che il sistema di puntamento orienti l'antenna con una precisione sufficiente da dirigere (o ricevere) il massimo segnale elettromagnetico si effettua una misurazione. Essendo il rotore e il suo driver il cuore del sistema di puntamento, gli errori più importanti che devono essere minimizzati sono quelli riferiti principalmente ai punti 3.c, 3.e, 5.d e 5.e.

Con il sistema di misurazione che verrà proposto si vanno ad identificare esplicitamente gli errori 3.c e 3.e. lasciando i punti 5.d e 5.e ad una calibrazione iniziale. Infatti, gli errori di cui al 5.d e 5.e non variano sensibilmente con il movimento dell'antenna.

Principio di funzionamento della misurazione:

Il laser, solidale all'antenna, crea un raggio luminoso nella sua direzione di puntamento. Tramite una camera (che può essere calibrata per limitare errori di distorsione ottica) si riprende una determinata porzione di cielo, riconoscendovi le costellazioni e il punto in cui termina il raggio laser. Se si dà all'antenna un target da seguire rispetto alle stelle, il raggio laser misura con precisione l'effettiva direzione di puntamento, quindi l'errore. C'è bisogno di una calibrazione iniziale per allineare il raggio laser con la direzione di massima radiazione dell'antenna (puntando verso un punto noto, ad esempio stazione trasmittente), eliminando la deviazione dovuta ad assemblaggio impreciso dell'antenna, che porterebbe in sé un errore costante.
Affinchè possa attuarsi questa procedura sono necessari:

1. Image processing: per l'elaborazione e il riconoscimento del fascio laser nell'immagine (Capitolo 1: Riconoscimento Laser)
2. Algoritmi di Star tracking: per riconoscimento del pattern di stelle (Capitolo 2: Sviluppo Algoritmo AIM)

Star Tracking

Gli Star sensors sono costituiti da una camera (CCD, CMOS) abbinato ad un microcomputer, concepiti per la determinazione d'assetto di un satellite. Possono essere di tipo: Scanners, Trackers, Mappers.

Gli Star Trackers sono strumenti di alta precisione pensati per l'utilizzo durante la fase di missione a bassa velocità angolare. Oltre a questa principale funzione, quelli più moderni possiedono in aggiunta caratteristiche avanzate che consentono di:

• produrre con elevata accuratezza la stima del rateo

• fornire informazioni sulle velocità angolari anche quando la determinazione dell'assetto non è fattibile, ovvero quando la piattaforma non è stabile o ruota; [2]

. Tutti i metodi di star tracking per determinazione d'assetto statico agiscono concettualmente in maniera similare:

1. si esegue una foto di un certo scenario di stelle e si calcolano i *centroidi* di ciascuna stella;
2. si confronta lo scatto con quelli che sono caricati nel database e si procede tramite una conversione di coordinate;
3. infine, si applica un algoritmo che ha il compito di estrarre gli angoli di rotazione.

In questo modo si può determinare l'assetto del satellite con un'accuratezza nel range di pochi arco-secondi Possono essere utilizzati anche in eclissi, e sono molto utilizzati nelle missioni interplanetarie. L'assetto fornito in uscita è di solito sottoforma di quaternione (Quaternioni) .Soffrono però della presenza di emissioni luminose indesiderate da altre sorgenti, come la Terra o il Sole.

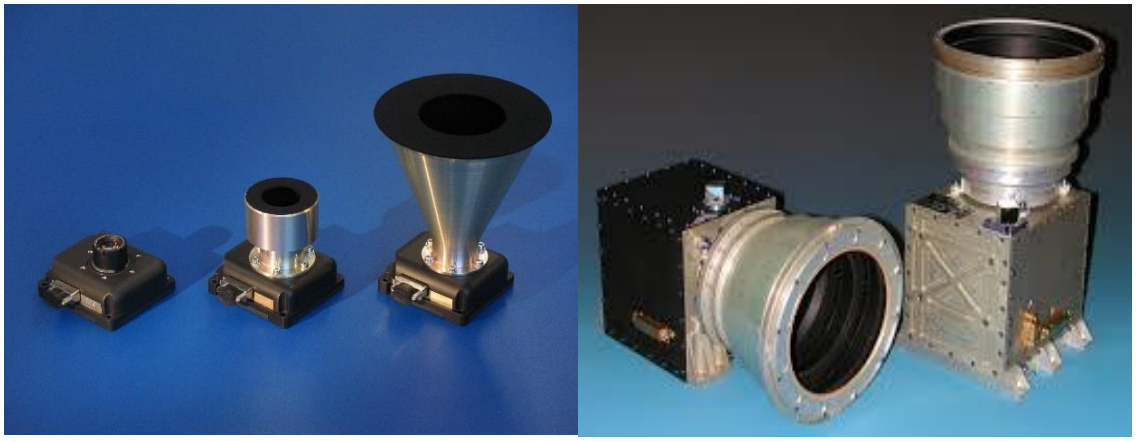


FIGURA 3 E FIGURA 4: ALCUNI ESEMPI DI STAR TRACKERS MODERNI CON SISTEMI OTTICI INTEGRATI.

Nello studio preso in esame, l'algoritmo usato per la conversione di coordinate e per estrarre gli angoli di rotazione è basato su un efficiente approccio di sovrapposizione ottimale delle stelle, in due immagini. In molti casi ha il vantaggio di eliminare una conversione di coordinate computazionalmente molto intensiva, la quale viene eseguita nei normali algoritmi di calcolo dell'assetto [8].

Quaternioni

I quaternioni, indicati con $\mathrm{q} = (\mathrm{q1}, \mathrm{q2}, \mathrm{q3}, \mathrm{q4})^T$ sono entità matematiche utilizzate per la descrizione dell'assetto di un corpo, alternative alla matrice di assetto, agli angoli di Eulero o a un sistema asse/angolo di Eulero. Essi sono definiti come segue:

$$q_1 \equiv e_1 \sin(\Phi/2)$$

$$q_2 \equiv e_2 \sin(\Phi/2)$$

$$q_3 \equiv e_3 \sin(\Phi/2)$$

$$q_4 \equiv \cos(\Phi/2)$$

È facile verificare a partire dalla definizione sopra che | q | = 1, il che suggerisce che i quaternioni hanno solo 3 parametri indipendenti (come deve essere), ma godono di una serie di vantaggi numerici, fra cui il fatto di non soffrire di singolarità.

Oggi, i quaternioni vengono utilizzati ampiamente nel controllo dell'assetto, in fisica e in astrodinamica. La ragione è che la combinazione di molte trasformazioni descritte da quaternioni, oltre ad essere più efficiente, è più stabile numericamente rispetto alla combinazione di molte trasformazioni matriciali.

Il vantaggio dei quaternioni è dato dal modo in cui sono gestite rotazioni successive. Consideriamo per esempio una rotazione tra q e q', e una seconda rotazione tra q' e q'':

$$A(\boldsymbol{q}'') = A(\boldsymbol{q}')A(\boldsymbol{q}) \quad \rightarrow \quad \boldsymbol{q}'' = \begin{bmatrix} q_4' & q_3' & -q_2' & q_1' \\ -q_3' & q_4' & q_1' & q_2' \\ q_2' & -q_1' & q_4' & q_3' \\ -q_1' & -q_2' & -q_3' & q_4' \end{bmatrix} \boldsymbol{q} = \boldsymbol{Q}'\boldsymbol{q}$$

La forma a destra coinvolge solo 16 moltiplicazioni, invece delle 27 di quella a sinistra. [3]

Architettura e progetto della procedura:

La seguente sezione presenta il procedimento realizzato per eseguire lo studio di fattibilità di una tecnica di misurazione per l'errore di puntamento d'antenna. Unendo l'algoritmo di image processing per riconoscere la terminazione del fascio laser con l'algoritmo di star tracking, che consente di orientarsi rispetto alla volta celeste si può ricavare l'effettiva direzione di puntamento del laser (ovvero dell'antenna).

Per questa procedura sono stati utilizzati:

1. Un computer portatile Acer Aspire E1-522, sul quale sono stati sviluppati e testati gli algoritmi principali in ambiente Matlab.
2. Uno smartphone Huawei Y6 II con accelerometro e magnetometro integrato (le cui funzionalità verranno spiegate successivamente) con una camera da 13 megapixel di risoluzione, con il quale sono state simulate le costellazioni per la prova.
3. Un puntatore laser verde SKY5-532mn per lunghe distanze ad uso professionale.
4. Un appoggio stabile costituito da un selfie-stick per smartphone regolabile, che è stato opportunamente fissato per effettuare gli scatti.
5. Le applicazioni StarWalk2 e multiTool Instruments per Android installate sullo smartphone, utilizzate rispettivamente per la creazione del database di stelle e la determinazione dell'assetto relativo dell'asse camera al momento dello scatto (Pitch,Roll,Azimuth).

FIGURA 5 E FIGURA 6: PUNTATORE LASER E COMPUTER ACER UTILIZZATI PER LA PROVA

Specifiche salienti Acer E1-522
SCHERMO: Led 15.6” 16:9 (HD Ready 1366x768).
CPU: AMD A4-5000 (1.50 Ghz, 4 core, 2 MB CACHE L2).
SCHEDA GRAFICA: AMD Radeon HD 8330 memoria condivisa.
HARD DISK: 500 GB SATA 5400 rpm.
RAM: 4 GB DDR3L.

Specifiche salienti Huawei Y6 II
CPU: HiSilicon Kirin 620
GPU: Mali-450MP4
Numero core: octa
Frequenza: processore1.2 GHz
RAM: 2 GB
Memoria interna (ROM): 16 GB
Accelerometro - Bussola digitale -Sensore di prossimità -Sensore di luminosità

Tramite lo smartphone è stato fatto un video in cui viene ripreso il cielo di notte, con l’ausilio di un appoggio stabile, puntando con il laser-pen un raggio verso una costellazione. Nel video si vedono le stelle sullo sfondo e il laser che si muove su di esso in maniera distinta. Questo passaggio serve successivamente per testare l’algoritmo di riconoscimento laser. È importante che lo smartphone sia fermo e che il video mostri sempre la stessa porzione di cielo. Per questo si è utilizzato lo stick-regolabile su un appoggio fisso che limitasse il movimento della camera.

Primo step per star tracking

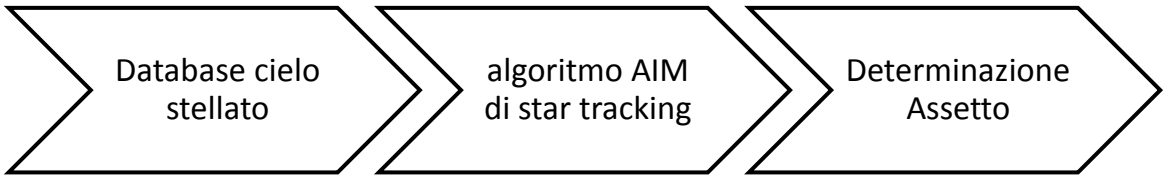
È stato costruito un database di immagini di costellazioni, necessari come riferimento per l’algoritmo di star tracking, dunque la determinazione d’assetto. Per la determinazione degli errori di angolo da parte della camera non è indispensabile l'uso di immagini reali di sfondi stellati: in questo caso sono state utilizzate immagini simulate tramite l’app Skywalk2, ma è possibile procedere diversamente. Ad esempio, per un ampio database di cieli stellati si può ricorrere allo SKY2000 catalog. È importante che si rilevi al momento dello scatto l’assetto della fotocamera in coordinate euleriane.

Per questa determinazione è necessario che la camera (in questo caso lo smartphone) possieda un accelerometro e un magnetometro interno. Utilizzando l'applicazione MultiSensorTool è stato possibile ricavare gli angoli di assetto dell'asse ottico della camera, ogni volta effettuato lo scatto di una costellazione.

Secondo step

Una volta creato un database con un discreto numero di immagini, si trasferiscono i dati significativi su dei file testo .txt (si può usare qualsiasi formato leggibile da Matlab), in modo da non dover caricare ogni volta le immagini .png, bensì sfruttare una semplice funzione che sappia leggere e riportare le informazioni più velocemente dal documento di testoche è stato già elaborato) all'ambiente di calcolo. Un buon file testo per il database deve contenere almeno le coordinate immagine delle stelle calcolate (vedi Centroidi.m) e l'assetto al momento di quello scatto. Si possono aggiungere anche i nomi delle singole stelle che conosciamo e che abbiamo fotografato, ma questo esula dall'obiettivo dell'algoritmo.

A questo punto si è sviluppato l'algoritmo di star tracking e testata la sua funzionalità.



Capitolo 1: Riconoscimento Laser

L'algoritmo di riconoscimento laser fa uso dell'implementazione di diverse strategie di image processing. Per "image processing" intendiamo l'insieme di attività di elaborazione digitale di immagini ed estrarre informazioni utili. Lavorando sui pixel dell'immagine con trasformazioni numeriche, tali algoritmi restituiscono in output ancora delle immagini. [10]

Di Seguito si fa riferimento alle principali elaborazioni che vengono fatte sui frame del video affinchè si possa riconoscere il raggio laser, e successivamente le sue coordinate finali (nel piano immagine).

Hough Transform e operatore Canny:

Sul primo frame individuato dal video viene utilizzata la trasformata di Hough [11]. La trasformazione di Hough è usata per rilevare caratteristiche geometriche come le linee rette nelle immagini digitali [13]. Essa è una delle procedure più utilizzate nella visione artificiale. Nel caso specifico serve a identificare la linea individuata dal raggio laser. Per applicare la trasformazione, è innanzitutto auspicabile una preelaborazione del rilevamento dei bordi. Questo di solito viene effettuato tramite l'algoritmo di Canny [12].

I bordi caratterizzano i confini e sono quindi un problema di fondamentale importanza nell'elaborazione delle immagini. Essi sono aree con forti contrasti di intensità – un salto di intensità da un pixel all'altro. L'algoritmo, lungo queste aree, tiene traccia e sopprime qualsiasi pixel che non è al massimo (soppressione non massima). L'isteresi viene utilizzata per tenere traccia dei pixel rimanenti che non sono stati soppressi.
L'isteresi utilizza due soglie: se la grandezza è al di sotto della prima soglia il pixel viene impostato su zero (configurazione non edge). Se la grandezza è al di sopra della soglia alta, il pixel è mantenuto e viene fatto un bordo (configurazione edge). E se la grandezza è tra le 2 soglie allora viene mantenuta l'ultima classificazione (edge o non-edge).

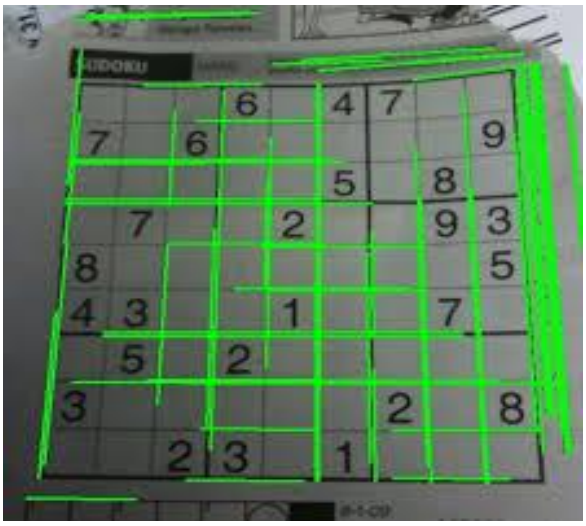


FIGURA 7: DETERMINAZIONE DELLE LINEE PRESENTI NELL'IMMAGINE DI UNA GRIGLIA SUDOKU TRAMITE L'ALGORITMO DI HOUGH.

FIGURA 8: DETERMINAZIONE BORDI DI UN'IMMAGINE PER MEZZO DELL'OPERATORE DI CANNY.

Optical Flow – Lucas Kanade Method

Per i frame successivi si individua il laser tramite l'algoritmo di Lukas-Kanade, un algoritmo di Optical Flow.

La stima del movimento, generalmente noto come optical flow, è una branca molto importante dell'elaborazione di sequenze di immagini. Esso consiste nel processo di determinazione dei vettori di movimento che descrivono la trasformazione da un'immagine a un'altra. In altre parole, si calcola la velocità delle strutture dell'immagine da un fotogramma a quello successivo. La stima del movimento è principalmente classificata in due categorie:

1. stima dei parametri di movimento;
2. segmentazione del movimento. [15]

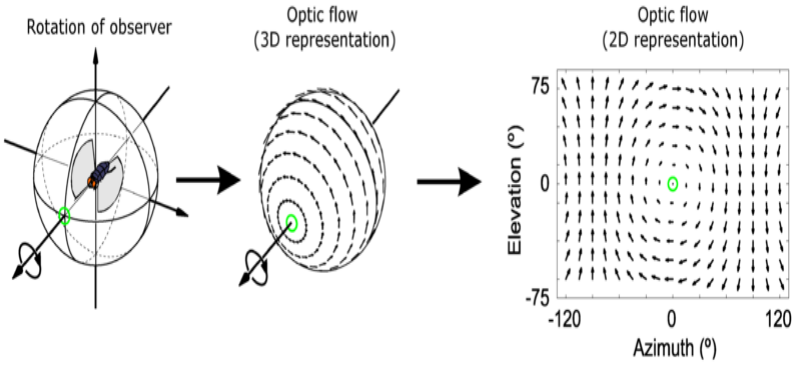


FIGURA 9. IL FLUSSO OTTICO SPERIMENTATO DA UN OSSERVATORE ROTANTE (IN QUESTO CASO UNA MOSCA). LA DIREZIONE E LA GRANDEZZA DEL FLUSSO OTTICO IN OGNI POSIZIONE SONO RAPPRESENTATE DALLA DIREZIONE E DALLA LUNGHEZZA DI OGNI VETTORE. [WIKIPEDIA]

Il metodo di Lucas-Kanade è del tipo *gradient based approach*. È un metodo ampiamente utilizzato nella stima del flusso ottico e nella visione artificiale. Questo metodo risolve le equazioni di base del flusso ottico per tutti i pixel della regione presa in considerazione, in base al criterio dei minimi quadrati. È un metodo puramente locale perché non può fornire informazioni di flusso all'interno di regioni uniformi dell'immagine. Qui si presume che il flusso sia essenzialmente costante in un intorno locale del pixel in esame. Il metodo Lucas Kanade presuppone che lo spostamento del contenuto dell'immagine tra due frame vicini sia piccolo e approssimativamente costante all'interno di un intorno del punto p [16][17].

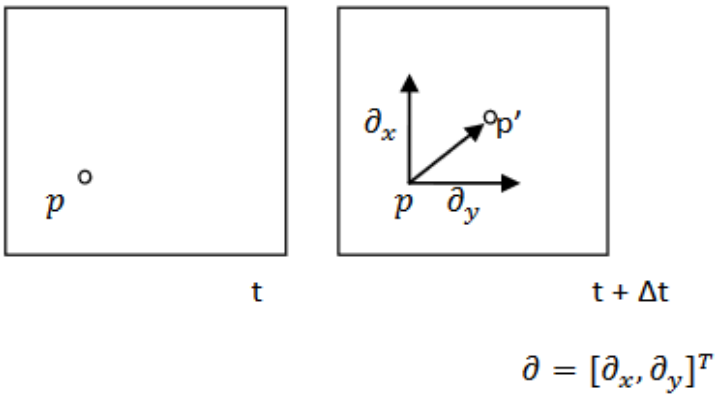


FIGURA 10: IMMAGINE 2-D DELLA POSIZIONE DEL PIXEL

$$I_x(P_1)V_x + I_y(p_1)V_y = -I_t(p_1)$$

$$I_x(P_2)V_x + I_y(p_2)V_y = -I_t(p_2)$$

.
.
.

$$I_x(P_n)V_x + I_y(p_n)V_y = -I_t(p_n)$$

Dove $P_1, P_2 \ldots . P_n$ sono pixel all'interno della finestra, $I_x(P_n), I_y(P_n), I_t(P_n)$ sono le derivate parziali dell'immagine I rispetto alla posizione x, y e time t, valutate nel punto P_ne al tempo corrente. Le equazioni possono essere scritte in forma di matrice:

$$Av = b \text{ , con } A = \begin{bmatrix} I_x(P_1) & I_y(P_1) \\ I_x(P_2) & I_y(P_2) \\ & . \\ & . \\ & . \\ I_x(P_n) & I_y(P_n) \end{bmatrix}, \qquad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix}, \qquad b = \begin{bmatrix} -I_t(p_1) \\ -I_t(p_2) \\ . \\ . \\ . \\ -I_t(p_n) \end{bmatrix}$$

Utilizzando il l'ottimizzazione ai minimi quadrati si determina la soluzione di Lucas kanade. Ora, possiamo scrivere questa equazione in un'altra forma in modo da ottenere:

$$A^T AV = V = (A^T A)^{-1} A^T b$$

La matrice $A^T A$ è spesso chiamata tensore struttura dell'immagine nel punto p.

La soluzione dei minimi quadrati però dà la stessa importanza a tutti gli N pixel $P_n$nella finestra. In pratica è meglio dare più peso ai pixel che sono più vicini al pixel centrale p. Per questo, si utilizza la versione ponderata dell'equazione dei minimi quadrati:

$$A^T WAV = A^T W\, b$$

Dove A^T è la trasposta della matrice A. Possiamo scrivere questa equazione in un'altra forma in modo da ottenere:

$$V = (A^T WA)^{-1} A^T W\, b$$

Dove W è una matrice diagonale n × n che contiene i pesi $W_{ii} = w_i$ da assegnare all'equazione di pixel. Cioè, calcola:

$$V = \begin{bmatrix} \sum_i w_i I_x(pn)^2 & \sum_i w_i I_x(pn) I_y(pn) \\ \sum_i w_i I_x(pn) I_y(pn) & \sum_i w_i I_y(pn)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i w_i I_x(pn) I_t(pn) \\ -\sum_i w_i I_x(pn) I_t(pn) \end{bmatrix}$$

Il peso w_i è di solito impostato su una funzione gaussiana della distanza tra p_n e p. Lucas e Kanade hanno implementato un'ottimizzazione ai minimi quadrati (LS) minimizzando l'espressione:

$$\sum w^2(x,y)\,[\nabla I(x,y,t)v \;+\; It(x,y,t)]^2$$

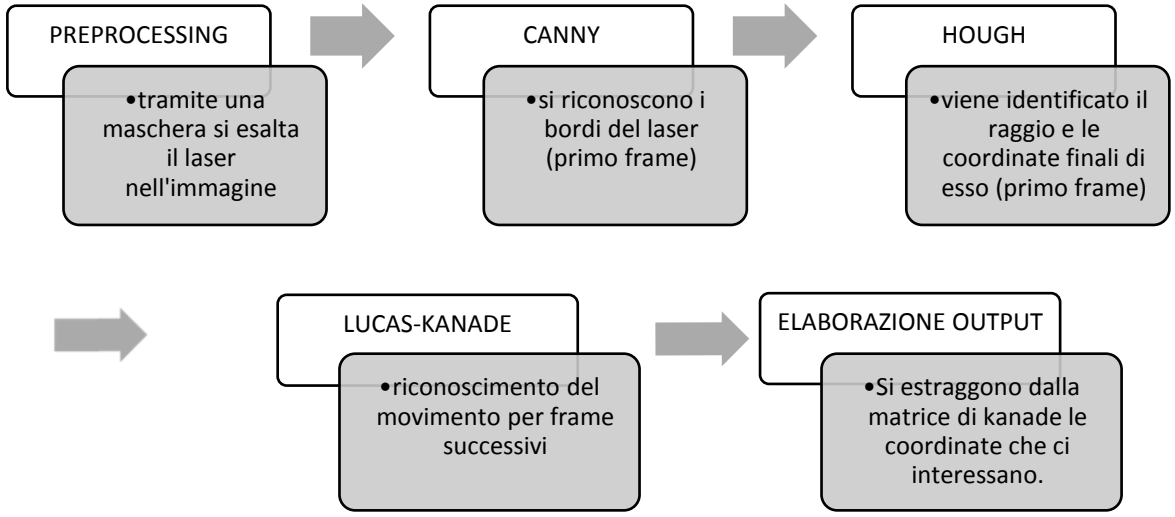
Dove $w(x,y)$ indica una funzione finestra che dà maggiore influenza agli spostamenti più vicini al pixel centrale.

Cause di errori in questa procedura possono essere i seguenti:

1. ATA è facilmente invertibile.
2. La costanza di luminosità non è soddisfatta.
3. Il movimento non è sempre piccolo
4. Problema nello spostare la posizione dei pixel come i suoi pixel vicini
5. La dimensione della finestra è troppo grande (occorre trovare una dimensione ideale) [16]

Implementazione in ambiente MATLAB:

L'algoritmo di image processing richiede come abbiamo visto vari step. Di seguito viene presentato il diagramma a blocchi del codice implementato (vedi allegati):



È importante per una questione di performance dell'algoritmo di Kanade (step 4), che si scelga come immagine di input non tutto il frame del video (360x680), bensì una regione più ristretta attorno al raggio laser. Il numero di pixel adatto per la finestra può variare in base al video a disposizione, in questo caso si è scelta una finestra 81x81.

L'algoritmo di Kanade presenta una soglia di sensibilità al movimento: se nel video il laser si muove troppo velocemente è probabile che per alcuni frame l'errore di determinazione delle coordinate sia maggiore.

La function built-in di Matlab "estimate_flow" può essere utilizzata in maniera preliminare per riconoscere i pixel dei frame che vengono stimati in movimento. Un output della funzione è una matrice 81x81, i cui elementi rappresentano ciascun pixel, con valore posto uguale a zero se stimato fermo, posto a un valore diverso da zero se stimato in movimento.

Rielaborando i dati di questa matrice si possono isolare solo i pixel corrispondenti alle coordinate che a noi interessano, ovvero quelle che si trovano sulla fine del puntamento del laser.

FIGURA 11:SCREEN-SHOT DALL'AMBIENTE DI CALCOLO MATLAB, RAPPRESENTA UN FRAME DEL VIDEO IN CUI SI RICONOSCONO LE COORDINATE DELLA TERMINAZIONE DEL LASER.

Nella figura, si può vedere il risultato a video del codice. Viene riconosciuta la coordinata finale del laser nel piano immagine (marcata con una "X" rossa). In alto a sinistra, in scala di grigio, si può vedere la finestra scelta per fare operare Kanade. Le coordinate del puntamento vengono salvate ad ogni frame in un vettore.

Capitolo 2: Sviluppo Algoritmo AIM

L'obiettivo dell'algoritmo è quello di determinare l'assetto del satellite o più in generale, del corpo su cui è montata la camera, in funzione di un database che possiede preliminarmente. Questo algoritmo si basa sullo studio condotto dal Phd. Delabie [9] e si sviluppa nei seguenti step:

Determinazione centroidi:

Il primo step dell'algoritmo è quello di individuare i centroidi dell'immagine di stelle. Questo corrisponde a individuare sul piano immagine i punti mediamente più illuminati e quindi ci dà un'informazione precisa sulle coordinate delle stelle nel piano (x, y).

Identificazione delle stelle:

Si calcola la similarità tra l'immagine della fotocamera e le immagini di database pre-elaborate utilizzando la trasformazione di distanza più breve. L'immagine del database con la somiglianza più elevata viene mantenuta e le stelle possono essere identificate in questo modo. Questo algoritmo è significativamente più robusto di altri algoritmi di identificazione per stelle. Inoltre, è veloce e dà un valore di prestazioni affidabile, che indica se le stelle sono state identificate correttamente.

Conversione delle coordinate:

Negli attuali algoritmi l'assetto viene determinato utilizzando vettori unitari (i, j, k). Pertanto, il passaggio successivo consiste nel convertire le coordinate del piano focale (x, y) delle stelle osservate in vettori unitari, per poi compararli con quelli presenti nel database. Questo può essere fatto utilizzando un semplice modello come descritto:

$$\begin{cases} i = \dfrac{\frac{x-x_0}{F}}{\sqrt{1+(\frac{x-x_0}{F})^2+(\frac{y-y_0}{F})^2}} \\ j = \dfrac{\frac{y-y_0}{F}}{\sqrt{1+(\frac{x-x_0}{F})^2+(\frac{y-y_0}{F})^2}} \\ i = \dfrac{1}{\sqrt{1+(\frac{x-x_0}{F})^2+(\frac{y-y_0}{F})^2}} \end{cases} \quad (a)$$

In queste equazioni (x, y) sono le coordinate delle stelle nell'immagine; (x_0, y_0) sono le coordinate dell'intersezione tra il piano focale e l'asse ottico; (i, j, k) è il vettore unitario della stella osservata, descritto nel frame di riferimento del sensore; e F è la lunghezza focale del sistema ottico.

Nell'algoritmo di Delabie, invece di convertire le stelle osservate, viene eseguita la conversione delle coordinate del vettore (i, j, k) delle *stelle del database* nelle coordinate $(\hat{x}, \hat{y})$, tramite il seguente sistema

$$\begin{cases} \hat{x} = x_0 + F i_r / k_r \\ \hat{y} = y_0 + F j_r / k_r \end{cases} \qquad (b)$$

Questa conversione è molto meno intensiva dal punto di vista computazionale rispetto alla conversione nel versore delle coordinate 2D delle stelle nell'immagine della camera.

Per convertire i vettori di unità del database in coordinate nel piano focale, i vettori selezionati devono prima essere ruotati in modo che siano centrati attorno all'asse *k* della camera. Per eseguire questa rotazione, viene selezionato il quaternione q_{dat}. Questo quaternione può essere considerato un'approssimazione dell'assetto attuale.

Poiché q_{dat}è un quaternione unitario, l'inverso è uguale al suo coniugato. Per ruotare il vettore di unità del database v (i, j, k) sull'inverso del quaternione q_{dat}:

$$v_r = \bar{q}_{dat}\, v \quad (c)$$

$v_r(i_r, j_r, k_r)$ è il vettore unitario a seguito della rotazione da parte dell'inverso di q_{dat}; $\bar{q}_{dat}$ indica che viene preso il coniugato del quaternione.
Il vettore v_r si può calcolare in maniera esplicita seguendo la seguente espressione:

$$v_r = \begin{bmatrix} v_{r1} \\ v_{r2} \\ v_{r3} \end{bmatrix} = \begin{bmatrix} (1 - 2q_2^2 - 2q_2^3) & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & (1 - 2q_{21} - 2q_{23}) & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & (1 - 2q_1^2 - 2q_2^2) \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Dove $v = \mathbf{i}v_1 + \mathbf{j}v_2 + \mathbf{k}v_3$ e $q = q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3$. Per implementare questo passaggio si può utilizzare la funzione di Matlab "quatrotate", la quale in input accetta un vettore 1×3 (o matrice $n \times 3$) e un quaternione (o matrice $m \times 4$). Il vettore ruotato di output è una matrice $m \times 3$.

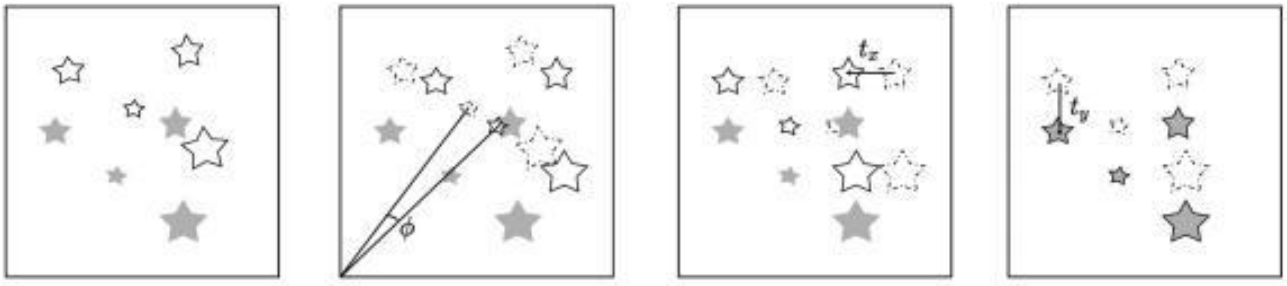
NB: nel database, è possibile usare direttamente dei file testo che contengano i valori dei vettori $v_r(i_r, j_r, k_r)$ per ogni stella identificata, al posto del file con le coordinate (x, y) e l'assetto della camera. In questo modo viene applicato direttamente l'algoritmo AIM, saltando lo step di confronto fra centroidi.

Stima dell'assetto:

Al centro di AIM vi è un'efficiente ottimizzazione per trovare la trasformazione che riporta l'immagine della telecamera a l'immagine delle stelle del database in modo ottimale uno sopra l'altro. Questa trasformazione viene quindi utilizzata per determinare la differenza tra l'assetto (rappresentato dal quaternione q_{diff} in cui è stata scattata l'immagine della stella della telecamera (essendo il vero assetto q_a) e l'assetto in corrispondenza del quale è stata selezionata l'immagine della stella del database q_{dat}. Infine, l'assetto stimato del satellite q_e è calcolato dai risultati precedenti. Per trovare la trasformazione che corrisponde in modo ottimale alle stelle di due immagini una sopra l'altra, viene costruita una funzione di costo che deve essere ridotta al minimo. Questa funzione di costo è la somma delle distanze Euclidee tra ogni coppia di stelle della foto scattata e la corrispondente stella di database trasformata:

$$\Gamma(\phi, t_x, t_y) = \sum_{i=1}^{n_s} w_i \{[x_i - \hat{x_i}\, cos(\phi) + \hat{y_i} sin(\phi) - t_x]^2 + [\, y_i - \, \hat{x_i} sin(\phi) - \hat{y_i} cos(\phi) \, - \, t_y]^2\}$$

In questa funzione, w_i è il peso dato alla stella i-esima ; (x, y) sono le coordinate della stella i-esima nel piano focale; $(\hat{x}, \hat{y})$, sono le coordinate della stella candidata dal database corrispondente alle stelle nel piano focale; ϕ è l'angolo su cui le stelle del database vengono ruotate rispetto all'origine del fotogramma del database; t_x e t_y sono le distanze su cui le stelle del database vengono traslate rispettivamente nelle direzioni x e y; n_s è il numero di stelle utilizzate nell'algoritmo di stima dell'assetto.



a) Camera stars (filled) and database stars **b) Rotation over angle ϕ** **c) Translation over t_x** **d) Translation over t_y**

Per ottenere una corrispondenza ottimale delle stelle di database con le stelle fotocamera-immagine, la distanza totale tra le stelle corrispondenti deve essere ridotta al minimo:

$$min[\Gamma(\phi, t_x, t_y)]$$

Le variabili sconosciute ϕ, t_x e t_y, che minimizzano questa funzione di costo, possono essere trovate calcolando la derivata di questa funzione di costo per ciascuna delle incognite, impostando le tre equazioni ottenute uguali a zero e risolvendo. Queste tre relazioni esplicite sono fornite nelle equazioni seguenti:

$$\phi = arctan2(sx.s\hat{y} - sy.s\hat{x} - sx\hat{y} + sy\hat{x}\,, -sx.s\hat{x} - sy.s\hat{y} + sx\hat{x} + sy\hat{y}) \qquad (d)$$

$$t_x = (sx - \frac{b}{2})cos(\phi) + (sy - \frac{b}{2})sin(\phi) + \frac{b}{2} - s\hat{x} \qquad (e)$$

$$t_y = -(sx - \frac{l}{2})sin(\phi) + (sy - \frac{l}{2})cos(\phi) + \frac{l}{2} - s\hat{y} \qquad (f)$$

Qui, b e l sono i valori massimi che le coordinate dei pixel possono avere nel piano focale; $(\mathrm{b}/2;\mathrm{l}/2)$ è quindi la coordinata del centro del piano focale. Le variabili $sx\,; s\hat{x};\ s\hat{y}; sy\,; sx\hat{x}\,; sy\hat{y}\,; sy\hat{x}$ sono le somme pesate delle coordinate [9].

I tre valori di trasformazione vengono quindi convertiti in tre angoli di Eulero:

$$\boldsymbol{\varphi = \varphi} \qquad \boldsymbol{\theta = arctan(\gamma, \frac{ty}{\gamma})} \qquad \boldsymbol{\psi = arctan(\gamma, \frac{tx}{\gamma})} \qquad \textbf{(g)}$$

in cui gamma è la fotocamera FOV. Poiché la maggior parte dei calcoli nel sistema di controllo e determinazione d'assetto vengono eseguiti utilizzando quaternioni, questi angoli di Eulero vengono poi convertiti in un quaternione q_{diff}. Questo quaternione rappresenta la rotazione stimata necessaria per ruotare il quaternione di database q_{dat} fino ad avere il quaternione q_a di vero assetto. Infine, moltiplicando q_{diff}. per q_{dat}, il quaternione q_e è stimato:

$$q_e = q_{diff} \otimes q_{dat} \qquad (h)$$

q_e è ciò di cui si ha bisogno per lo scopo, rappresenta una stima dell'assetto.

Implementazione in ambiente Matlab:

Sono stati implementati i codici, che seguono fedelmente quanto appena discusso. Il primo step (determinazione centroidi) viene svolto in particolare da uno script indipendente, i tre seguenti sono stati implementati insieme in un'unica funzione. Di seguito vengono presentati sotto forma di flusso logico:

Centroidi.m

carico immagine.txt dal database	•carico l'assetto dell'immagine in coordinate euleriane e le coordinate delle stelle (datapoints) •converto l'assetto in quaternione q
carico foto.png da confrontare	• calcolo i centroidi dell'immagine (imagepoints) • plot dei punti trovati
salvataggio dati per step successivo	• salvo datapoints • salvo imagepoints

Aim.m

carico dati input	• quaternione di assetto database • imagepoints e datapoints
calcolo coordinate (i,j,k) delle stelle del database	• opportuno ciclo for con all'interno sistema di conversione (a)
conversione delle coordinate (i,j,k) in (x^,y^)	• sistema di coordinate conversion (b)
Determinazione parametri phi , psi , theta	•calcolo dei valori tx e ty (e)(f)
determinazione funzione di costo	•conoscendo tutti i valori, si imposta un ciclo che segua la formulazione (Γ)

Obiettivo del codice:

Si vuole testare che: partendo da uno scatto del cielo ricavato con l'applicazione Star-Walk2 (oppure scatto del cielo con camera), l'algoritmo sia in grado di riconoscere la costellazione fotografata per confronto con il database interno. Il database interno è formato da file di testo:

Estratto di Auriga.txt

Nome stella	4.80299539170507	523.367007488479	-75.8 -176.0 139.4
Nome stella	42.7543520309478	705.005802707930	
Nome stella	80.1814650388457	575.857380688124	
Nome stella	183.884227129338	64.5668769716088	
Nome stella	186.828182941904	555.152039555006	
Nome stella	205.988755020080	376.987148594378	
Nome stella	254.977412731006	546.726899383984	
Nome stella	413.242759700324	677.341048865034	
Nome stella	472.269681742044	560.025125628141	
Nome stella	482.973188151270	310.433672872029	
Nome stella	521.083262531861	353.663551401869	

…..

nel file di testo sono presenti 4 tipologie di dati: nella colonna delle "nome stella", è possibile inserire i nomi delle stelle che si conoscono; nelle due colonne successive vi sono i valori delle coordinate X e Y delle stelle nell'immagine; l'ultima colonna presenta un solo dato, ovvero l'assetto in coordinate euleriane dell'immagine (ricavato al momento dello scatto tramite Multi Sensor Tool).

Per estrarre questi dati è stata implementata una funzione specifica "importfile.m".

Il codice Centroidi.m serve a ricavare le coordinate delle stelle dello scatto della costellazione. Esso è in grado di riconoscere le regioni dell'immagine che superano una certa soglia di luminosità e, calcolando il baricentro di queste regioni, salva i valori dei pixel corrispondenti. Le coordinate delle stelle che possediamo in database sono note da un file text. Il codice AIM.m, attuando i calcoli e le opportune conversioni discusse, ci dà l'informazione su quanto le due immagini si discostino fra loro in termini di rotazioni (ϕ, ψ, θ).

Infine, utilizzando la funzione di costo che minimizza le distanze fra le stelle di due immagini, attuando un confronto fra lo scatto in input e tutte le immagini presenti in database, si vuole riconoscere quale fra le nostre immagini in database quale si accosti di più allo scatto (riconoscere quindi la costellazione e infine il quaternione dell'immagine).

FIGURA 12: IMMAGINE IN FORMATO PNG DELLA COSTELLAZIONE DELL'AURIGA.

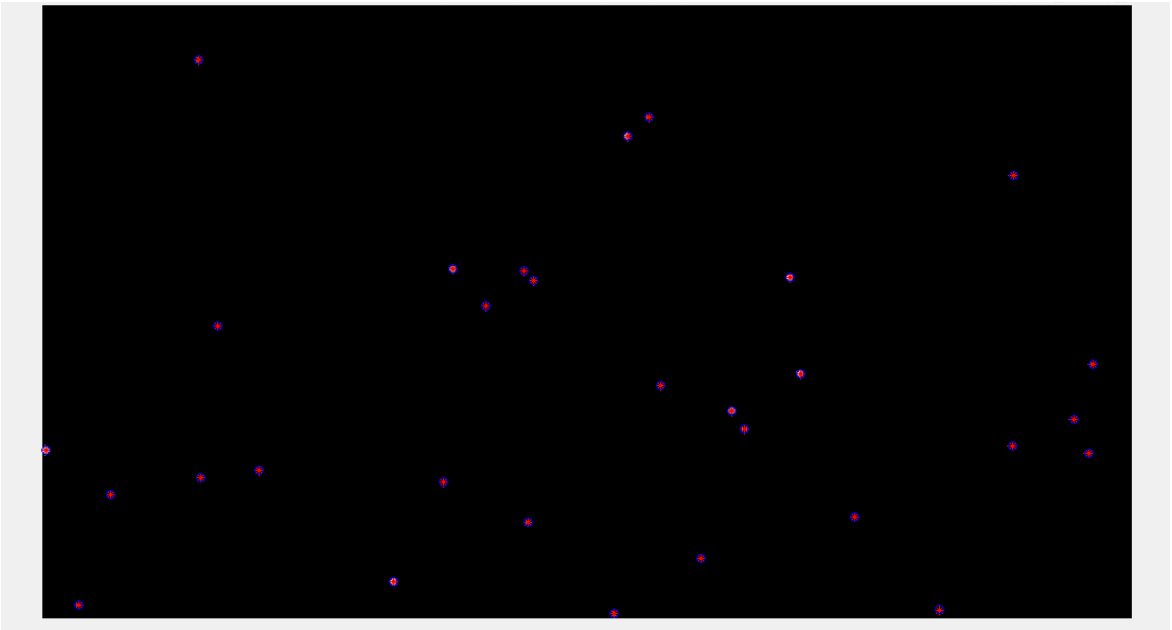


FIGURA 13: PLOT IN AMBIENTE MATLAB DEI CENTROIDI RILEVATI SUL PIANO IMMAGINE

Per testare la funzionalità di questi algoritmi si è proceduto nel seguente modo: sapendo che l'output della sequenza centroidi.m-AIM.m sono gli angoli di Eulero ϕ, ψ, θ di scostamento fra due immagini, si è verificato che per due immagini coincidenti questi angoli avessero valori nulli.

Risultati

Presa l'immagine della costellazione "nome.png" e confrontata con il file "nome.txt" presente nel database che si è precedentemente creato, si verifica che:

Confronto	Φ [deg]	Ψ [deg]	Θ [deg]
auriga.png vs auriga.txt	16.64e-015	5.04e-012	-16.58e-012
orsa_maggiore.png vs orsa_maggiore.txt	3.98e-015	775.91e-015	-3.88e-012
cancro.png vs cancro.txt	-15.72e-015	-4.46e-012	16.88e-012
cefeo.png vs cefeo.txt	4.51e-015	1.29e-012	-5.53e-012

Come mostrato da questa tabella, i valori sono pressoché nulli per tutti e tre gli angoli in ogni caso che si è confrontato. I risultati trovati sono coerenti con quelli che ci si aspettava. Il codice è stato verificato per tutte le 25 immagini di database, portando a risultati analoghi.

A questo punto si è proceduto con lo sviluppo dell'ultimo algoritmo. Si vuole fare in modo che il codice dica fra tutte le immagini del database, quale si avvicina effettivamente di più allo scatto di riferimento, e ci dia in output una stima dell'assetto vero della nostra foto.

Come detto, q_e sarà il quaternione dell'assetto stimato, e Γ la funzione di costo che dà un'informazione su quanto l'immagine si discosti da quelle nel database. Di seguito viene presentata la funzione quat_e_determination.

Quatedetermination.m

Questa funzione è stata creata una volta verificati i codici precedenti, perché ci fornisse direttamente la stima dell'assetto e la funzione di costo dell'immagine analizzata in input, senza passare per due script differenti, velocizzando molto il processo.

Input	• Textfile contenente i vettori V(i j k) delle stelle • textfile contenente le coordinate immagine (x,y) e l'assetto del file di database • scatto di cui si vuole conoscere l'assetto
carico assetto e vettori v(i j k)	• coordinate euleriane dal file • q=eul2quat • qi=quatinv(q)
confronto con la foto calcolo puntiImmagine (coordinate della foto)	• dall'algoritmo di centroidi
calcolo coordinate (i,j,k) delle stelle del database	• opportuno ciclo for con all'interno sistema di conversione [delabie]
conversione delle coordinate (i,j,k) in (x^,y^)	• sistema di conversione di coordinate(a)
Determinazione parametri phi , psi , theta	• calcolo dei valori tx e ty, ovvero di quanto sono traslate le due immagini in assi x e y
determinazione funzione di costo	• conoscendo tutti i valori, si imposta un ciclo che segua la formulazione (Γ)

La funzione è stata testata analizzando i risultati di output per diversi scatti eseguiti. La logica è stata quella di scegliere come scatti di riferimento dal database, quelli che presentavano la funzione di costo più bassa. Di seguito un breve esempio in linguaggio MATLAB

```
clc
clear
foto='cancro.png'; %inserire la foto che si vuole riconoscere
format longEng
[qe1,g1]=quate_determination('cigno.txt','Vcigno.txt',foto);
[qe2,g2]=quate_determination('auriga.txt','Vauriga.txt',foto);
.
.
.
[qe16,g16]=quate_determination('ofiuco.txt','Vofiuco.txt',foto);
```

```
GAMMA=[g1 g2 g3 g4 g5 g6 g7 g8 g9 g10 g11 g12 g13 g14 g15 g16];
choice=find(GAMMA==min(GAMMA))
```

Inserendo una foto, ad esempio la costellazione del cancro, il risultato atteso, è che la funzione di costo corrispondente all'immagine della costellazione del cancro (che è nel database) sia la più piccola rispetto alle funzioni di costo di tutte le altre immagini paragonate. In questo modo l'algoritmo ha riconosciuto che porzione di cielo è stata fotografata, e può restituire il quaternione di assetto corrispondente.

Risultati

Nelle seguenti tabelle sono riportati i risultati delle funzioni di costo ottenute. In input sono state date alla funzione quatedetermination.m le immagini delle costellazioni del cancro (tabella 1), la costellazione dell'auriga (tabella 2), la costellazione del dragone (tabella 3).

Immagine di riferimento nel database	**Funzione di Costo**
Cigno	64.76e+006
Boote	17.38e+006
Cancro	**61.68e-021**
Cassiopea	59.96e+006
Cefeo	1.01e+009
Dragone	314.66e+006
Gemelli	28.62e+006
Giraffa	325.30e+006
Orsa_minore	772.93e+006
Auriga	4.38e+006
Orsa_maggiore	100.91e+006
Ofiuco	430.27e+006

Tabella 1.

Immagine di riferimento nel database	**Funzione di Costo**
Cigno	440.21e+006
Boote	796.53e+006
Cancro	611.15e+006
Cassiopea	307.41e+006
Cefeo	372.93e+006
Dragone	1.08e+009
Gemelli	967.48e+006
Giraffa	514.21e+006
Orsa_minore	1.47e+009
Auriga	**273.34e+005**
Orsa_maggiore	1.25e+009
Ofiuco	518.72e+006

Tabella 2.

Immagine di riferimento nel database	**Funzione di Costo**
Cigno	835.94e+006
Boote	1.10e+009
Cancro	854.45e+006
Cassiopea	1.09e+009
Cefeo	1.09e+009
Dragone	**1.31e+003**
Gemelli	1.35e+009
Giraffa	739.76e+006
Orsa_minore	902.67e+006
Auriga	680.69e+006
Orsa_maggiore	1.26e+009
Ofiuco	467.07e+006

Tabella 3.

Come si può notare dalle tabelle, le funzioni di costo sono minime proprio in corrispondenza delle immagini di database attese. L'algoritmo è in grado di calcolare quindi quale fra le immagini del database sia più vicina a quella di input.

Nel caso della prima tabella, l'immagine di input è la stessa di quella presente in database. Si può notare l'enorme differenza fra la funzione di costo dell'immagine esatta (e-21) e le altre (fra e+06 ed e+09). Nel caso della seconda e terza tabella si valuta anche come si comporta l'algoritmo se si presenta un'immagine traslata e ruotata rispetto a quella in database, ovvero un'immagine che riprende la stessa porzione di cielo ma da punti differenti. Gli ordini di grandezza sono più ridotti ma comunque rilevanti in termini di confronto (c'è sempre una differenza di almeno un ordine di grandezza fra l'immagine esatta e le altre).

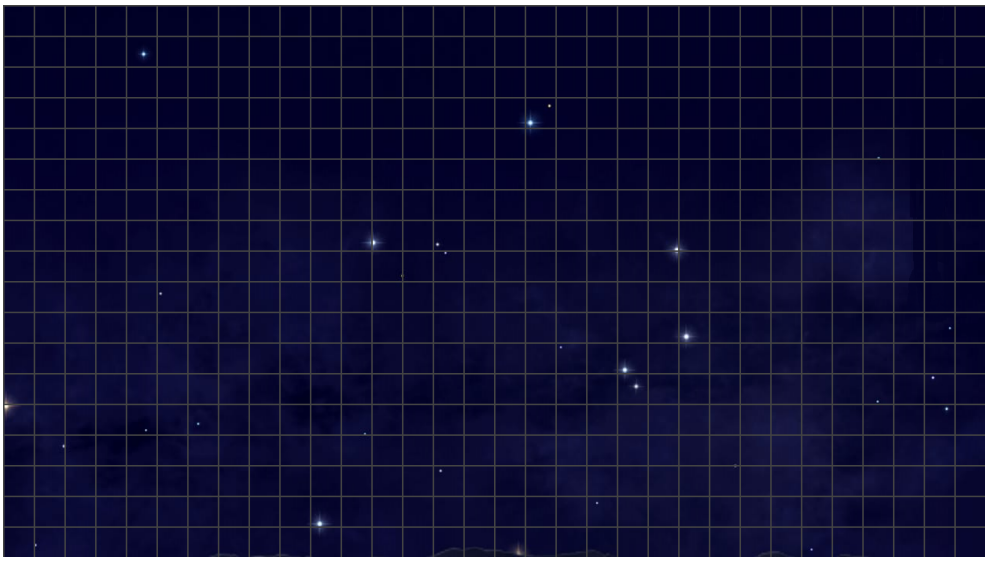


FIGURA 14: LA COSTELLAZIONE DELL'AURIGA PRESENTE NEL DATABASE.

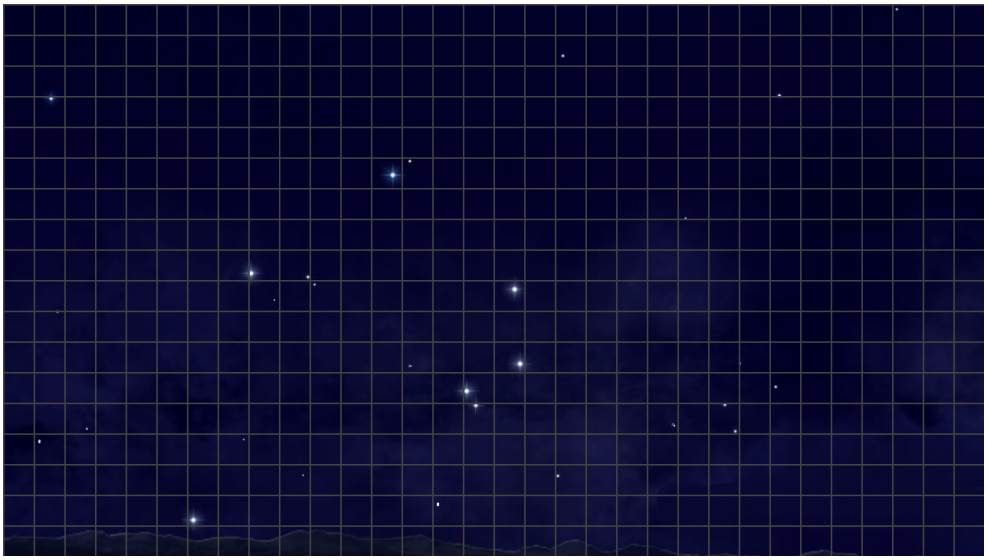


FIGURA 15: LO SCATTO DATO IN INPUT ALLA FUNZIONE.

In termini di errori e possibili miglioramenti, un parametro fondamentale è sicuramente la dimensione del database, ovvero la quantità di immagini di costellazioni catalogate di cui si dispone.

Una volta che si è individuata la costellazione fotografata, è possibile riportare il quaternione corrispondente. Si è ricavato quindi l'assetto.

$$q_{e\_cancro} = [-0.599, \quad -0.459, \quad 0.258, \quad -0.604]$$

$$q_{e\_auriga} = [-0.093.93, \quad 0.957, \quad -0.004, \quad -0.271]$$

$$q_{e\_dragone} = [0.503, \quad -0.580, \quad 0.416, \quad -0.488]$$

Conclusioni:

Sono stati sviluppati e verificati due test in ambiente Matlab in maniera parallela, utili al riconoscimento d'assetto di un corpo su cui si monta una camera solidale e al riconoscimento ottico di un laser.

Queste due prove, che sembrerebbero diversificate in termini di test, in realtà fanno parte dello studio di fattibilità di un unico progetto che prevede il loro utilizzo combinato. In particolare, l'idea è quella di stimare l'errore di puntamento di un'antenna, mettendola solidale a un dato puntatore laser e puntandola verso una stella nota.
Conoscendo le coordinate spaziali e l'assetto della camera, e conoscendo la stella (fissa) grazie all'algoritmo di star tracking, si può misurare l'errore che c'è fra il puntamento del laser (grazie all'algoritmo di riconoscimento si ottiene le coordinate della terminazione) e la posizione da puntare. Da qui infine ottenere un'informazione sull'errore complessivo di puntamento.

Per la prima prova è stato implementato un opportuno algoritmo di star tracking, dopo aver creato un database di riferimento; per la seconda prova, tramite un algoritmo di image processing, sono state tracciate le coordinate della terminazione di un fascio laser, ripreso dalla camera. In entrambi i casi i risultati possono essere migliorati, ad esempio effettuando la calibrazione della fotocamera (per evitare distorsioni intrinseche alle lenti), ampliando e dettagliando il numero di scatti e le informazioni del database; applicando delle tecniche di pre-processing dei frame discusse nel capitolo 1.

Visti i risultati ottenuti, l'apparecchio di misura descritto nell'introduzione risulta fattibile. Sebbene l'effettiva precisione di misura rimanga da verificare come sviluppo futuro del presente lavoro.

Riferimenti:

[1]: *Calibrazione di un banco prova per la determinazione d'assetto di un satellite tramite sensore di stelle* - Rosetti Lorenzo – Alma Mater Studiorum Bologna

[2]: *Implementazione e test di un algoritmo per la determinazione d'assetto di microsatelliti tramite sensore di stelle*. Traini Federico – Alma Mater Studiorum Bologna

[3]: Slide del corso "Satelliti e Missioni Spaziali" del Prof. Locarini

[4]**:** *Tracking Quality Measurements Of Ground Station For Low Earth Orbit Satellite*: Marcin Stolarski-Space Research Centre

[5] *S-band ground station prototype for low-earth orbit nanosatellite missions* -Octavian CRISTEA, Paul DOLEA, Paul Vlăduţ DASCĂL*****

[6] *THE USE AND ADVANCEMENT OF AN AFFORDABLE, ADAPTABLE ANTENNA POINTING MECHANISM*- Mark Ferris- Nigel Phillips

[7] *Control and Pointing Challenges of LargeAntennas and Telescopes* -Wodek Gawronsi

[8] *Measuring method for real-time accurate direction of astronomical telescope*

Matching. Tjorven Delabie, ∗ Joris De Schutter,† and Bart Vandenbussche‡

[9]: *Highly Efficient Attitude-Estimation Algorithm for Star Trackers Using Optimal Image Tjorven Delabie,∗ Joris De Schutter,† and Bart Vandenbussche‡ Katholieke Universiteit Leuven, 3001 Heverlee, Belgium*

[10] *Metodi numerici per la segmentazione di immagini digitali* -Elena Morotti – Alma Mater Studiorum Bologna

[11] *How the Hough Transform Was Invented*- Peter E.Hart

[12] Canny, J, *A computational approach to edge detection* - IEEE Transactions on Pattern Analysis and Machine Intelligence

[13] *Use of the hough trasformation to detect lines and curves in pictures*- April 1971 By: Richard O. Duda Peter E. Hart

[15] *The Computation of Optical Flow* -S. S. Beauchemin And J. L. Barron -Universlty of Western Ontario

[16] *Optical Flow Measurement using Lucas kanade Method* - Dhara Patel- Saurabh Upadhyay,

[17] *Secrets of Optical Flow Estimation and Their Principles* - Deqing Sun Brown University - Stefan Roth TU Darmstadt -Michael J. Black Brown University

Appendice

Matrice di assetto:

Serve a specificare l'orientamento relativo tra due sistemi di riferimento. Sia $\mathcal{F}b$ un sistema di riferimento mobile, identificato dalla triade di vettori unitari **u**, **v** e **w**, il cui orientamento è richiesto rispetto ad un sistema di riferimento «fisso» $\mathcal{F}i$, identificato dalla triade di vettori unitari **c1**, **c2** e **c3**. Il modo più naturale per eseguire questa operazione è specificando le componenti di **u**, **v** e **w** nel sistema di **c1**, **c2**, e **c3**. I nove parametri possono essere organizzati in una matrice: [3]

$$\hat{u} = \left(u_1, u_2, u_3\right)^T$$

$$\hat{v} = \left(v_1, v_2, v_3\right)^T \qquad A = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ w_1 & w_2 & w_3, \end{bmatrix}$$

$$\hat{w} = \left(w_1, w_2, w_3\right)^T$$

La matrice A è detta matrice d'assetto, chiamata in alternativa anche Direction Cosine Matrix, DCM. Dato che gli elementi di A sono componenti di versori mutuamente ortogonali, essi devono rispettare i seguenti 6 vincoli:

$$\hat{u} \cdot \hat{u} = \hat{v} \cdot \hat{v} = \hat{w} \cdot \hat{w} = 1$$

$$\hat{u} \cdot \hat{v} = \hat{u} \cdot \hat{w} = \hat{v} \cdot \hat{w} = 0$$

Questo significa che solo 3 elementi di A sono indipendenti, che suggerisce di cercare altri metodi di rappresentazione d'assetto, che facciano uso di meno di 9 elementi. Dalla definizione di A, può essere verificata la proprietà per cui l'inversa della matrice di assetto è uguale alla sua trasposta.

La matrice di assetto mappa la rappresentazione in colonna di qualsiasi generico vettore b, dal sistema di riferimento inerziale, al sistema di riferimento corpo, come si può verificare dal seguente prodotto matriciale:

$$A\boldsymbol{b}_{Fi} = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ w_1 & w_2 & w_3, \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{u}} \cdot \boldsymbol{b} \\ \hat{\boldsymbol{v}} \cdot \boldsymbol{b} \\ \hat{\boldsymbol{w}} \cdot \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} b_u \\ b_v \\ b_w \end{bmatrix} = \boldsymbol{b}_{Fb}$$

Un'ultima e molto importante proprietà della matrice di assetto è che ha sempre almeno un autovalore unitario. Questo significa, per definizione di autovalore, che esiste sempre un vettore **e** tale che:

$$A\hat{\mathbf{e}} = \hat{\mathbf{e}}$$

$\hat{e}$ è detto asse di Eulero o asse di rotazione.

RINGRAZIAMENTI

Ringrazio il Professore Alfredo Locarini, per la sua gentilezza e grande disponibilità, per avermi consigliato e dato la possibilità di intraprendere questo percorso di tirocinio e tesi sotto la sua guida e dell'Ing. Giacomo Curzi.

Ringrazio il mio supervisore Ing. Giacomo Curzi per la pazienza e i preziosi consigli elargiti, per avermi schiarito lungo questi mesi tutti i dubbi che gli sottoponessi, in maniera elegante e chiara.

Ringrazio la mia famiglia, senza la quale questo lungo percorso non avrebbe avuto inizio, per aver sempre creduto in me.

Ringrazio Luana, per avermi fatto da spalla in questi tre anni intensi, per essere mia complice.