UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Dipartimento di Ingegneria Industriale e Scienze Matematiche

Corso di Laurea Magistrale in Ingegneria Meccanica Termomeccanica

SVILUPPO DI UN SISTEMA PER L'ANALISI TRIDIMENSIONALE DEL MOVIMENTO ATLETICO SPORTIVO BASATO SU ANALISI DI IMMAGINE MEDIANTE INTELLIGENZA ARTIFICIALE

DEVELOPMENT OF A SYSTEM FOR THE THREE-DIMENSIONAL ANALYSIS OF ATHLETIC SPORTS MOVEMENT BASED ON IMAGE ANALYSIS USING ARTIFICIAL INTELLIGENCE

Relatore:

Prof. Paolo Castellini

Laureando:

Giovanni Salerno

Anno accademico: 2021/2022

INDICE

1. INTRODUZIONE

1.1 Visualizzazione e analisi movimento senza contatto

Le camere sono sempre più utilizzate a livello scientifico perché hanno una serie di comodità, quali ad esempio l'assenza di contatto e di sensori fisici, che altri sistemi di rilevamento non hanno e poiché ci sono molteplici possibili evoluzioni con potenzialità notevoli. In questa tesi viene mostrato come sia possibile analizzare il movimento umano partendo da video, questa tecnica sarà applicata ai movimenti atletico/sportivi in particolare al nuoto. Le telecamere hanno un già larghissimo uso nei test ingegneristici ma in questo caso troviamo una sorta di upgrade perché non verrà utilizzata una singola camera ma una serie di camere. Impiegandone due in contemporanea si può osservare una scena in tre dimensioni, in pratica imitando il funzionamento dell'occhio umano, se il numero di camere è superiore a due vale lo stesso discorso, anzi ciò permette di osservare la scena da più angolazioni e poiché il soggetto osservato in questo studio è l'uomo questo porta a notevoli vantaggi, come ad esempio l'assenza di dati persi, perché guardandolo da più punti di vista si ha una ridondanza di informazioni che minimizza l'eventuale singolo errore. Anche se l'applicazione di questa tecnica sarà riguardante il nuoto, le prove sono state effettuate per comodità in laboratorio analizzando movimenti casuali o cercando di simulare un nuotatore. Nella *figura 1* si può vedere un esempio di disposizione delle camere.

Figura 1: esempio disposizione camere

Il software utilizzato per il riconoscimento del corpo umano è Mediapipe, questo programma consente grazie ad un'intelligenza artificiale di identificare da un'immagine il corpo umano suddiviso in 33 punti (*figura 2*) e fornisce le coordinate di ognuno di questi con unità di misura in pixel. Il software lavora con video singoli, quindi, anche se nelle prove il soggetto è stato ripreso da più camere, su Mediapipe verrà esaminato un video per volta.

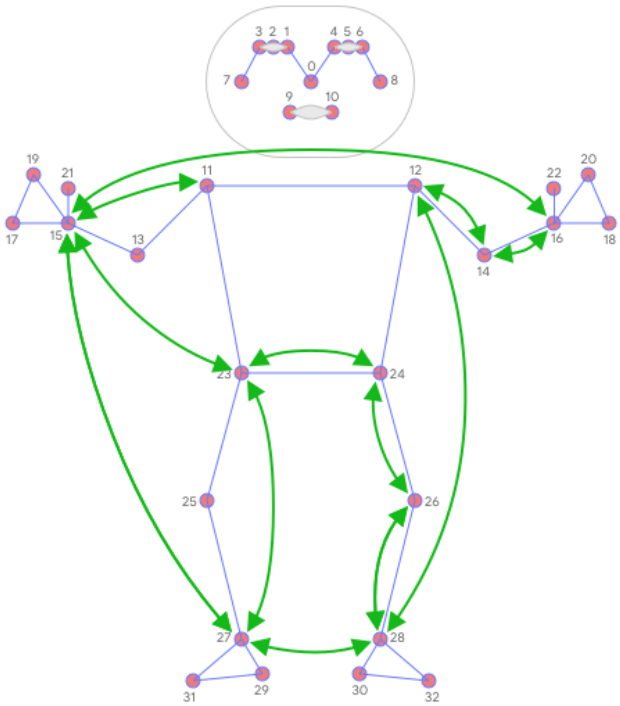

Figura 2: punti del corpo cercati da Mediapipe

Le coordinate vengono analizzate con il software Matlab che sfrutta le calibrazioni delle camere per capire i loro parametri e come sono disposte in termini di posizione e angolazione, unisce questi dati a quelli ottenuti da Mediapipe per ogni camera per triangolare e ottenere le coordinate degli stessi punti in 3D, come si può notare in *figura 3*, in questo caso la scena è in verticale perché c'è una differenza tra il sistema di riferimento delle camere e tra quello standard delle figure di Matlab.

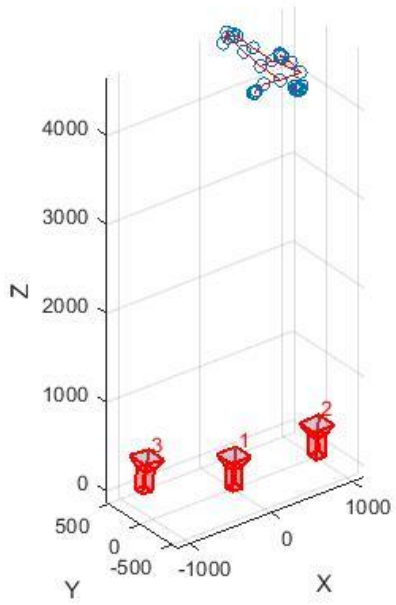

Figura 3: visualizzazione soggetto con camere (Matlab)

1.2 Stato dell'arte:

1.2.1 Kinect:

Tra i sistemi esistenti per il riconoscimento del movimento di sicuro il più importante è il Kinect, un dispositivo sviluppato da Microsoft per le proprie console di videogiochi. Ovviamente l'ambito è totalmente diverso rispetto al presente studio però il funzionamento di base non è così differente.

Il Kinect è provvisto dei seguenti hardware [1]:

- Fotocamera RGB che fornisce le componenti di colore delle immagini;
- sensore di profondità 3D che grazie ad un laser e ad una telecamera riesce a dare informazioni in tre dimensioni, con delle limitazioni sulla portata;
- motore per l'inclinazione dei sensori;
- array di quattro microfoni che fornisce informazioni sulla distanza della sorgente del suono e riesce a distinguere la voce umana dal resto dei rumori.

Per quanto riguarda la parte software, questa è capace di individuare il corpo e separarlo dal resto della scena, inoltre conosce una serie di movimenti tipici e può, quindi, cercare di anticipare il possibile movimento del soggetto [2]. Ci sono diverse tipologie di applicazione, tra cui si possono citare giochi basati sul ballo oppure sulla simulazione di alcuni sport, ma anche programmi per il fitness che si distaccano dai classici videogiochi per ragazzi.

1.2.2 Smart Baduanjin:

Un altro esempio è un'app per dispositivi mobili sviluppata in Cina: "*Smart Baduanjin*". Il Baduanjin è un insieme di otto esercizi fisici praticato principalmente in Cina da persone anziane. Quest'app dimostra la giusta tecnica con cui svolgere gli esercizi e grazie alla camera del dispositivo riesce a riconoscere il corpo dell'utente e a dare un feedback evidenziando eventuali errori.

Il principio di funzionamento è semplificato, sia perché è sviluppato con una singola camera, quindi, il video è in due dimensioni, sia perché essendo utilizzata da cellulari non può risultare troppo pesante a livello computazionale.

2. MATERIAL AND METHOD

Strumenti di misura come le camere vengono impiegati molto spesso nell'ambito delle misurazioni a maggior ragione se è richiesta una certa precisione e se non è possibile utilizzare tecniche a contatto. Sono state effettuate inizialmente prove con due camere per semplicità, successivamente sono state inserite sempre più camere fino a un massimo di cinque. Per migliorare la stabilità sono state fissate su dei cavalletti ed è stato utilizzato un metodo di azionamento a distanza, con un "telecomandino" collegato con un filo, per fare in modo che non si muovesse premendo il pulsante della ripresa (*figura 4)*. Inoltre, il telecomandino aveva un altro vantaggio che era quello di sincronizzare le camere, poiché con un solo pulsante si attivavano tutte quelle connesse.

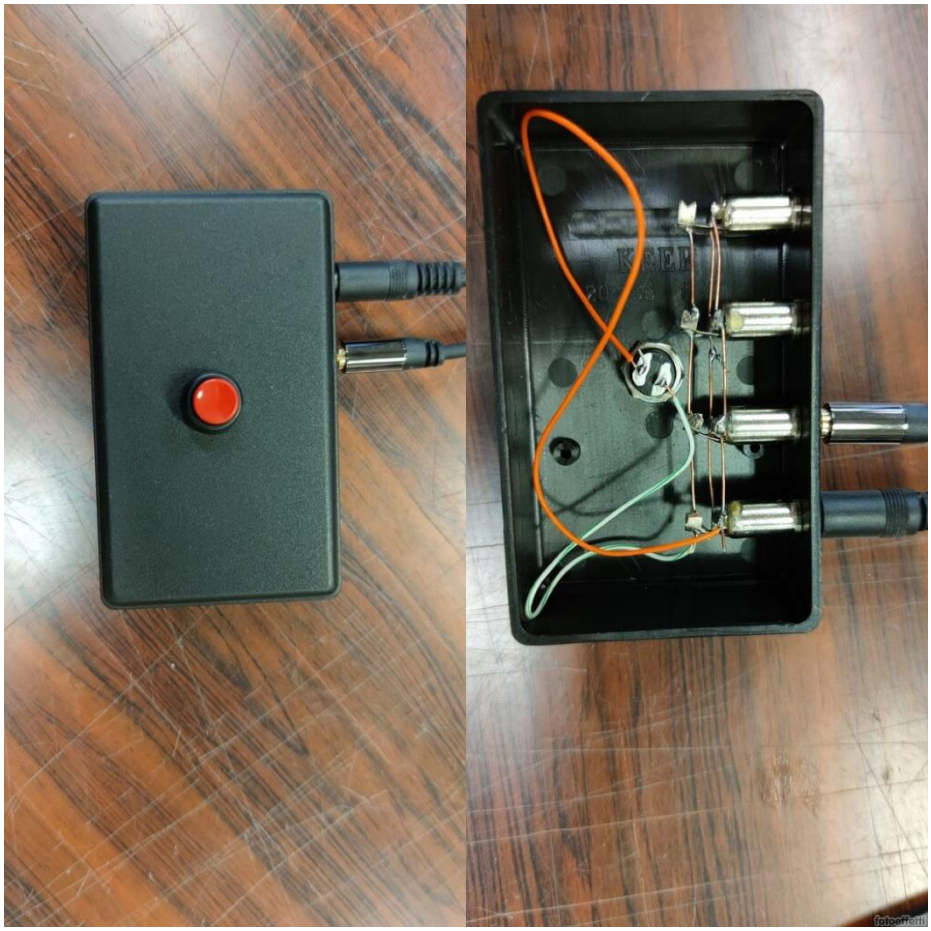

Figura 4: telecomandino chiuso e aperto

In questa sezione verrà descritto inizialmente il software adoperato per il riconoscimento del corpo umano poiché la sua creazione non è stata oggetto dello studio, ma è stato utilizzato come una scatola chiusa; successivamente verranno descritte le operazioni e i software utilizzati in maniera più approfondita.

2.1 Mediapipe

Mediapipe utilizza un'intelligenza artificiale per fornire le coordinate di 33 punti del corpo umano rilevati nell'inquadratura in un sistema di riferimento standard per le immagini ossia con origine in alto a sinistra, asse x orizzontale, asse y verticale e unità di misura in pixel; in realtà il programma fornisce anche la terza dimensione, cioè la profondità, ma come viene specificato anche sul sito [3] questa è ancora in fase di studio e quindi non affidabile. Mediapipe lavora con video singoli, quindi, anche se nelle prove il soggetto è stato ripreso da più camere, il programma analizzerà un video per volta. Considerando quindi, il software come una scatola chiusa, l'input è il video di una camera e l'output è un file testo composto da gruppi di 3 colonne (coordinate x, y e z) e 33 righe (punti del corpo riconosciuti), il numero di gruppi coincide con il numero di frame. Non è detto che il software riconosca sempre tutti i punti, poiché una parte del corpo potrebbe essere nascosta, succede quindi, che per alcuni frame ci possano essere dei dati persi, in tal caso il software assegna degli 'zeri' come coordinate; l'utilizzo di più camere che osservano il soggetto da ogni direzione risolve questo problema. Quando è inserito un video, questo viene inizialmente scomposto nei singoli frame, che però non vengono analizzati del tutto singolarmente perché il software è dotato di un filtro di Kalman [4] cioè uno stimatore ottimo che ipotizza la posizione dei 33 punti in base alla loro posizione nei frame precedenti; ciò può essere allo stesso tempo sia un vantaggio che uno svantaggio, poiché nel caso in cui il corpo umano venga riconosciuto già al primo frame è molto probabile che verrà riconosciuto anche in tutti quelli successivi, viceversa se nei primi frame per un qualunque motivo l'analisi del software risulta sbagliata, è difficile che in seguito verrà corretta e prosegua bene. Grazie alle prove svolte si può affermare che ci sono dei casi in cui il funzionamento di Mediapipe è agevolato, ossia quando il soggetto si trova al centro dell'inquadratura e quando risulta maggiormente visibile, ad esempio se lo sfondo è di colore omogeneo oppure se il colore del vestiario è in contrasto con lo sfondo stesso, inoltre è fondamentale il

riconoscimento del viso quindi importante non avere accessori come la mascherina. In *figura 5* si può vedere un esempio dei punti rilevati dal software in una delle prove.

Figura 5

2.2 Parametri telecamera:

Sono state utilizzate per le prove delle camere reflex (*figura 6*), sulle quali prima di partire con le riprese vengono settati alcuni parametri che non dovranno essere modificati nell'arco di tutta la prova, tra questi si possono definire:

- diaframma: determina la quantità di luce che arriva al sensore, la sua apertura si misura in "F" seguita da un numero che se piccolo indica un'apertura grande (più luce), se grande indica un'apertura piccola, inoltre è legato alla profondità di campo, infatti, un diaframma molto aperto diminuisce la profondità di campo e dà risalto a soggetti vicini e viceversa;

- tempo di esposizione: indica il tempo di apertura dell'otturatore, è consigliato ad esempio un tempo molto basso per soggetti in movimento, inoltre maggiore è il tempo di esposizione e maggiore sarà la luce in ingresso al sensore;
- ISO: anche questo parametro fornisce un modo per aumentare la luce, un ISO molto basso viene utilizzato se la luce naturale è elevata, viceversa per foto scattate ad esempio all'esterno di notte è consigliato un ISO più alto;
- distanza focale: distanza tra l'obiettivo (ridotto a un punto) e il piano focale, ossia il piano su cui viene messo a fuoco il soggetto osservato;
- punto principale: punto centrale del piano focale per cui passa la perpendicolare al piano passante per il centro dell'obiettivo, questi ultimi due parametri sono funzione della messa a fuoco.

Figura 6

2.3 Calibrazione camera

Un passaggio fondamentale per la determinazione dei parametri delle camere è la calibrazione, in genere bisogna calibrare lo strumento di misura ogni volta che viene effettuata una sessione di prove per essere certi che nel mentre non siano stati modificati

i parametri, inoltre bisogna accertarsi che non vengano modificati proprio durante la sessione. Per la calibrazione delle camere è stato utilizzato il software Matlab e in particolare i toolbox di calibrazione mono e stereo.

2.3.1 Calibrazione mono:

Al software vengono fornite una serie di immagini di un checkboard (scacchiera, *figura 7*) che vengono analizzate per ottenere come output i parametri fondamentali. Per semplicità sono stati effettuati dei video in cui un soggetto muoveva la scacchiera in tutta la zona di misura e sono stati estratti i frame dai video che corrispondono all'input del programma di calibrazione (*figura 8*, esempio di un frame). Per ricavare risultati più precisi si cerca di avere uno spazio calibrato molto maggiore dello spazio di lavoro, inoltre siccome il software a parità di immagini dà risultati migliori se queste sono molto diverse tra loro, il soggetto deve cercare non solo di muovere la scacchiera in tutta la zona ma anche di darle più angolazioni possibili. Frame vicini sono molto simili, quindi, non verranno utilizzati tutti i frame di un video ma solo uno ogni 10 o 20 circa. Il consiglio, dato da Matlab stesso, è di fornire almeno una decina di immagini per la calibrazione ma in questo caso in cui la zona di lavoro è abbastanza ampia (qualche metro di larghezza e lunghezza) conviene fornire un numero di immagini maggiore, ad esempio un centinaio. L'altro input è la lunghezza del lato dei quadratini della scacchiera che il software utilizza per fare i calcoli.

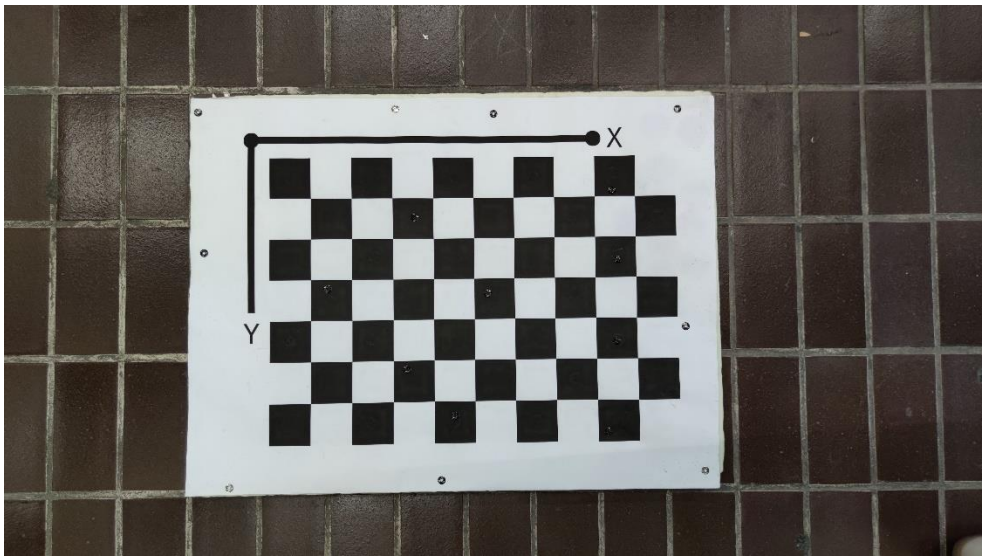

Figura 7: checkboard

Figura 8: esempio immagine per la calibrazione

L'algoritmo utilizzato da 'Camera calibration toolbox' riconosce tutti gli angoli presenti sulla scacchiera e, avendo appreso la relativa distanza (lunghezza quadratini), calcola le distanze reali (quindi in metri non in pixel) in tre dimensioni tra le varie immagini. Estratti gli angoli c'è la fase di inizializzazione in cui viene calcolata una soluzione in forma chiusa per i parametri intrinseci ed estrinseci della camera, seguita dalla fase di ottimizzazione in cui viene introdotta la distorsione (trascurata nella fase precedente) e un errore non lineare da ridurre con il metodo dei minimi quadrati. L'algoritmo calcola in modo automatico il numero di quadrati presenti sulla scacchiera e il coefficiente di

distorsione, anche se entrambi possono essere modificati manualmente; in ogni caso terminata la calibrazione si visualizza per ogni immagine l'analisi fatta dal software e l'errore calcolato in pixel, nel caso si può decidere di scartare alcune immagini oppure, più semplicemente si può scegliere un errore massimo e scartare tutte quelle con uno maggiore (*figura 9*), a questo punto la calibrazione riparte con le sole immagini restanti e chiaramente l'errore sarà inferiore.

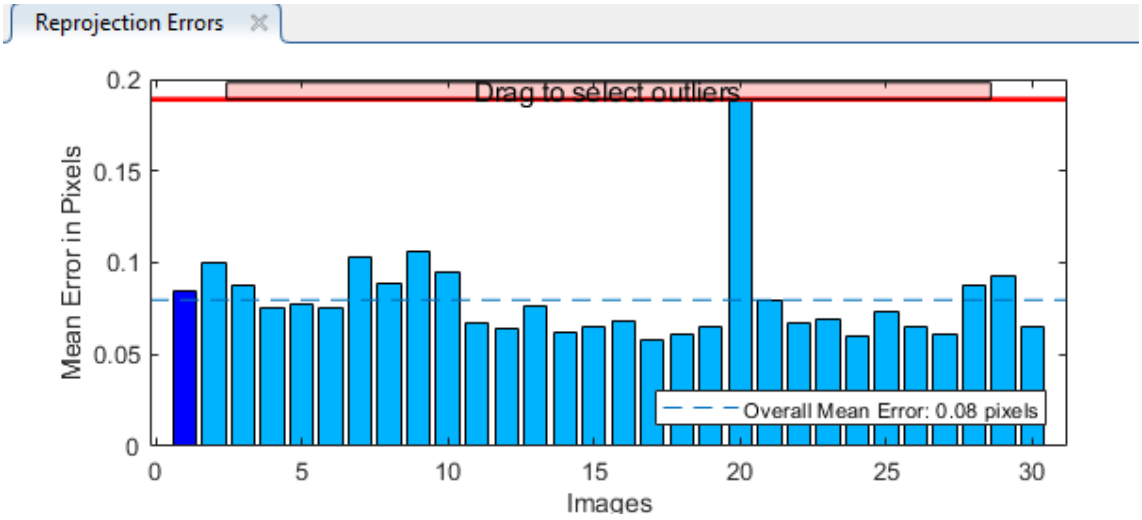

Figura 9: calibrazione mono: analisi errore

L'output è un file che viene chiamato genericamente '*cameraParams'* in cui sono presenti tutte le informazioni ottenute, ossia i parametri intrinseci ed estrinseci.

I parametri intrinseci sono:

- la lunghezza focale;
- il punto principale;
- il coefficiente di distorsione (diviso in coefficiente radiale e tangenziale);
- la dimensione in pixel delle immagini.

I parametri estrinseci invece descrivono lo spazio di lavoro tridimensionale, viene definito un sistema di riferimento centrato nella camera e si può visualizzare come sono disposte tutte le diverse scacchiere (diverse immagini fornite) rispetto alla camera (*figura 10)*.

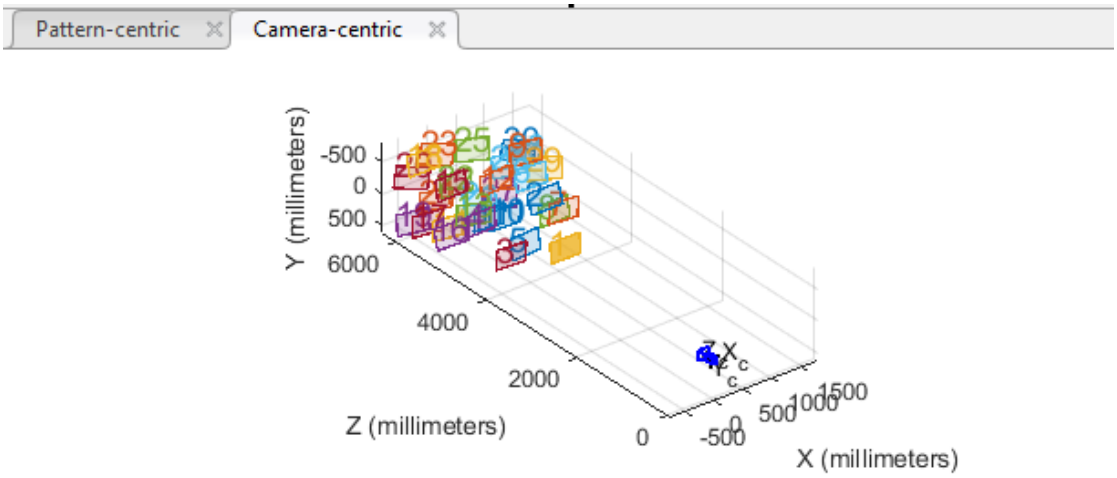

Figura 10: visualizzazione checkboard

Un'attenzione particolare per avere ottimi risultati è muovere lentamente la scacchiera nella fase di ripresa in modo da non avere frame mossi; si potrebbe decidere anche di scattare una serie di fotografie invece del video, ma la scelta è stata fatta per il video per avere gli stessi parametri che si avranno anche nelle riprese delle prove, poiché di solito le camere riducono la qualità durante un video rispetto alle foto. [5]

2.3.2 Algoritmo di calcolo:

L'algoritmo utilizzato dal toolbox di Matlab per estrarre i parametri intrinseci ed estrinseci è l'algoritmo di Zhang, uno dei metodi di calcolo più importanti per eseguire la calibrazione delle camere. L'obiettivo è quello di correlare le coordinate 2D di un punto nel sistema di riferimento dell'immagine con le coordinate 3D dello stesso punto ma rispetto al sistema di riferimento reale. Vengono definiti genericamente $\boldsymbol{m} = [u, v]^T$ il punto in coordinate bidimensionali e $\boldsymbol{M} = [X, Y, Z]^T$ il corrispondente tridimensionale, in questo caso $\boldsymbol{m}$ è la proiezione di $\boldsymbol{M}$ nel piano dell'immagine e la relazione che li lega è la seguente:

$$s\tilde{\boldsymbol{m}} = \boldsymbol{A}\,[\boldsymbol{R}\ \boldsymbol{t}]\tilde{\boldsymbol{M}} \qquad \boldsymbol{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{A}\,[\boldsymbol{r_1} \quad \boldsymbol{r_2} \quad \boldsymbol{r_3} \quad \boldsymbol{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2}$$

Dove *s* è un fattore di scala, ***R*** e ***t*** sono la matrice di rotazione e il vettore di traslazione, ***A*** è la matrice degli intrinseci, con u_0 e v_0 le coordinate del punto principale, α e β i fattori di scala degli assi u e v, e γ che descrive l'inclinazione degli assi dell'immagine. Con $\tilde{\boldsymbol{m}}$ e $\tilde{\boldsymbol{M}}$ si indicano i vettori "aumentati": $\tilde{\boldsymbol{m}} = [u, v, 1]^T$, $\tilde{\boldsymbol{M}} = [X, Y, Z, 1]^T$. Si può introdurre la matrice ***H***:

$$\boldsymbol{H} = \boldsymbol{A}\,[\boldsymbol{r_1} \quad \boldsymbol{r_2} \quad \boldsymbol{r_3} \quad \boldsymbol{t}] \qquad \boldsymbol{H} = [\boldsymbol{h_1} \quad \boldsymbol{h_2} \quad \boldsymbol{h_3}] = \boldsymbol{\lambda}\,\boldsymbol{A}\,[\boldsymbol{r_1} \quad \boldsymbol{r_2} \quad \boldsymbol{r_3} \quad \boldsymbol{t}] \tag{3}$$

$$s\tilde{\boldsymbol{m}} = \boldsymbol{H}\tilde{\boldsymbol{M}} \tag{4}$$

La matrice ***H*** è definita a meno di una costante, $\boldsymbol{\lambda}$ è una costante arbitraria.

Senza commettere errore si può considerare un modello piano in cui si ha: *Z=0*, di conseguenza:

$$[\boldsymbol{h_1} \quad \boldsymbol{h_2} \quad \boldsymbol{h_3}] = \boldsymbol{\lambda}\,\boldsymbol{A}\,[\boldsymbol{r_1} \quad \boldsymbol{r_2} \quad \boldsymbol{t}] \tag{5}$$

E poiché $\boldsymbol{r_1}$ e $\boldsymbol{r_2}$ sono ortonormali:

$$\boldsymbol{h}_1^T(\boldsymbol{A}^{-1})^T\boldsymbol{A}^{-1}\boldsymbol{h}_2 = 0 \tag{6}$$

$$\boldsymbol{h}_1^T(\boldsymbol{A}^{-1})^T\boldsymbol{A}^{-1}\boldsymbol{h}_1 = \boldsymbol{h}_2^T(\boldsymbol{A}^{-1})^T\boldsymbol{A}^{-1}\boldsymbol{h}_2 \tag{7}$$

Vengono definite la matrice ***B*** e il vettore ***b***:

$$\boldsymbol{B} = (\boldsymbol{A}^{-1})^T \boldsymbol{A}^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} \tag{8}$$

$$\boldsymbol{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T \tag{9}$$

Riprendendo la matrice $\boldsymbol{H}$, l'i-esima colonna è: $\boldsymbol{h}_i = [\boldsymbol{h}_{i1}, \boldsymbol{h}_{i2}, \boldsymbol{h}_{i3}]^T$, quindi si ha:

$$\boldsymbol{h}_i^T \boldsymbol{B} \boldsymbol{h}_j = \boldsymbol{v}_{ij}^T \boldsymbol{b} \tag{10}$$

dove:

$$\boldsymbol{v}_{ij} = [\boldsymbol{h}_{i1}\boldsymbol{h}_{j1}, \boldsymbol{h}_{i1}\boldsymbol{h}_{j2} + \boldsymbol{h}_{i2}\boldsymbol{h}_{j1}, \boldsymbol{h}_{i2}\boldsymbol{h}_{j2}, \boldsymbol{h}_{i1}\boldsymbol{h}_{j3} + \boldsymbol{h}_{i3}\boldsymbol{h}_{j1}, \boldsymbol{h}_{i3}\boldsymbol{h}_{j2} + \boldsymbol{h}_{i2}\boldsymbol{h}_{j3}, \boldsymbol{h}_{i3}\boldsymbol{h}_{j3}]^T \tag{11}$$

Rispettando le equazioni precedenti (6, 7) si può riscrivere:

$$\begin{bmatrix} \boldsymbol{v}_{12}^T \\ (\boldsymbol{v}_{11} - v_{22})^T \end{bmatrix} \boldsymbol{b} = 0 \qquad \boldsymbol{V}\boldsymbol{b} = 0 \tag{12}$$

Se si hanno *n* immagini si avranno *n* equazioni, quindi, $\boldsymbol{V}$ è una matrice di dimensione *2n×6*. Una volta determinato $\boldsymbol{b}$ si può passare al calcolo dei parametri intrinseci ed estrinseci:

$$v_0 = (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2)$$

$$\lambda = B_{33} - \frac{[B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]}{B_{11}}$$

$$\alpha = \sqrt{\frac{\lambda}{B_{11}}}$$

$$\beta = \sqrt{\frac{\lambda B_{11}}{B_{11}B_{22} - B_{12}^2}}$$

$$\gamma = -B_{12}\alpha^2\beta/\lambda$$

$$u_0 = \lambda v_0/\alpha - B_{13}\alpha^2/\lambda$$

$$\boldsymbol{r}_1 = \gamma \boldsymbol{A}^{-1}\boldsymbol{h}_1; \qquad \boldsymbol{r}_2 = \gamma \boldsymbol{A}^{-1}\boldsymbol{h}_2; \qquad \boldsymbol{r}_3 = \boldsymbol{r}_1 \times \boldsymbol{r}_2; \qquad \boldsymbol{t} = \gamma \boldsymbol{A}^{-1}\boldsymbol{h}_3. \tag{13}$$

Nella fase successiva viene minimizzato l'errore, nel caso in cui si hanno *n* immagini e *m* punti sul modello planare bisogna minimizzare la seguente funzione:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|m_{ij} - \hat{m}(A, R_i, t_i, M_j)\right\|^2 \tag{14}$$

In cui $\hat{m}$ è la proiezione di M_j nell'immagine i-esima.

Infine, come ultimo passaggio si introduce la distorsione dell'obiettivo, a volte se è molto piccola può essere trascurata commettendo un errore accettabile, ma per essere più precisi si può determinare con le seguenti relazioni:

$$\breve{x} = x + x[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2$$

$$\breve{y} = y + y[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2$$

$$\breve{u} = u + (u - u_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2$$

$$\breve{v} = v + (v - v_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2$$

Dove k_1 e k_2 sono i coefficienti di distorsione radiale, (u, v) sono le coordinate dell'immagine in pixel ideale, $(\breve{u}, \breve{v})$ le corrispondenti reali, ossia considerando la distorsione, analogamente (x, y) sono le coordinate delle immagini normalizzate ideali e $(\breve{x}, \breve{y})$ reali.

Poiché la calibrazione è stata utilizzata come una scatola chiusa grazie al toolbox di Matlab, l'algoritmo di calcolo non è stato esaminato dettagliatamente, per una trattazione più approfondita si può far riferimento a [6], [7].

2.3.3 Calibrazione stereo:

Come già dichiarato sono state utilizzate più camere per avere dei risultati in tre dimensioni, in questo caso è necessario effettuare anche la stereo calibrazione, in cui vengono calibrate le camere in coppia per ottenere sia i parametri intrinseci di entrambe che i parametri estrinseci relativi.

L'input da dare al toolbox 'Stereo Camera Calibration' è l'insieme delle coppie di immagini scattate nello stesso istante, quindi, sono stati effettuati video in contemporanea in cui entrambe le camere osservano il checkboard e da questi sono stati estratti i rispettivi frame; la procedura è quindi la stessa della calibrazione mono con l'accortezza che le camere siano sincronizzate, che le immagini coincidano in termini di pixel, quindi con la stessa risoluzione e che le immagini corrispondenti di una camera e dell'altra siano rinominate allo stesso modo.

L'algoritmo di calcolo cerca una corrispondenza tra le coppie di immagini con l'errore di valutazione che, allo stesso modo della calibrazione mono, deve essere minimizzato. In fase di calibrazione il software riconosce le corrispondenze osservando il checkboard e di conseguenza riesce a passare alle coordinate nel sistema di riferimento reale grazie alla conoscenza delle dimensioni del checkboard e a determinare i parametri estrinseci relativi. Completati i calcoli si avrà, come nel caso precedente, una visualizzazione delle disposizioni delle scacchiere rispetto alle due camere e una degli errori in pixel, (*figura 11*) che genericamente sono un po' più elevati rispetto alla mono calibrazione.

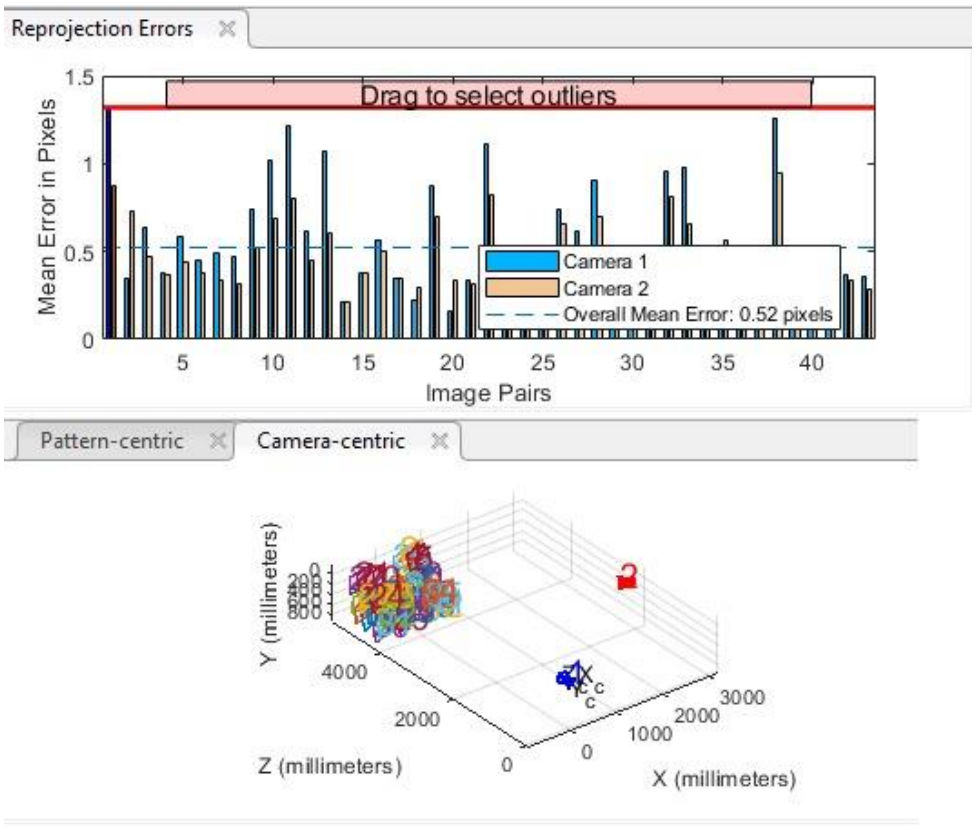

Figura 11: visualizzazione errori pixel e posizioni checkboard

L'output del processo è un file (‘*stereoParams’)* contenente sia informazioni sulle singole camere che informazioni relative, infatti, vengono estratti anche in questo caso i parametri intrinseci delle due che però sono meno precisi rispetto a quelli ottenuti dalla mono calibrazione; per quanto riguarda i parametri estrinseci bisogna specificare che la prima camera viene presa come riferimento, quindi, anche il centro dello stesso sistema di riferimento tridimensionale corrisponde al centro di questa. Si ottengono due informazioni fondamentali ossia il vettore spostamento chiamato genericamente ‘*Traslation of camera 2’* che descrive la posizione della seconda camera rispetto alla prima, e la matrice di rotazione chiamata ‘*Rotation of camera 2’* che ne descrive l'orientamento.

Quando il numero di camere è maggiore di due bisogna effettuare più calibrazioni poiché non esiste una in contemporanea di tre o più camere, verranno calibrate quindi due alla volta e i risultati saranno combinati. Le informazioni richieste dal programma sono la posizione e l'orientamento di tutte le altre camere rispetto alla prima, quindi, sarebbe utile calibrare ognuna singolarmente con la prima, ma se lo spazio di lavoro è

molto ampio o le camere hanno inclinazioni molto differenti non è detto che l'ennesima camera e la prima riescano ad osservare contemporaneamente il checkboard; dunque, vengono calibrate di volta in volta le due camere vicine. Infine, per ottenere le informazioni delle altre camere rispetto alla prima è sufficiente svolgere delle operazioni matriciali, anche questi calcoli sono stati effettuati con Matlab.

Si definisce matrice di trasformazione tra due sistemi di riferimento, quindi tra due camere, quella matrice 4x4 formata dalla matrice di rotazione del secondo sistema rispetto al primo nei primi 3x3 posti (R), il vettore posizione dell'origine del secondo rispetto al primo nell'ultima colonna (P) e tre '0' e un '1' nell'ultima riga (*figura 12*).

$$T = \left[\begin{array}{ccc|c} & R & & P \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Figura 12: matrice di trasformazione

Moltiplicando le matrici di trasformazione si ottengono le trasformazioni successive:

$$^{1}_{3}T = {}^{1}_{2}T\ {}^{2}_{3}T$$

$$^{1}_{n}T = {}^{1}_{2}T\ {}^{2}_{3}T \ldots {}^{n-1}_{n}T$$

Grazie a queste operazioni, in un setup formato da n camere, si possono acquisire tutte le matrici di trasformazione effettuando *n-1* stereo calibrazioni, dalle matrici si estraggono i dati sul posizionamento e sull'orientamento relativi alla prima camera. (*figura 13*).

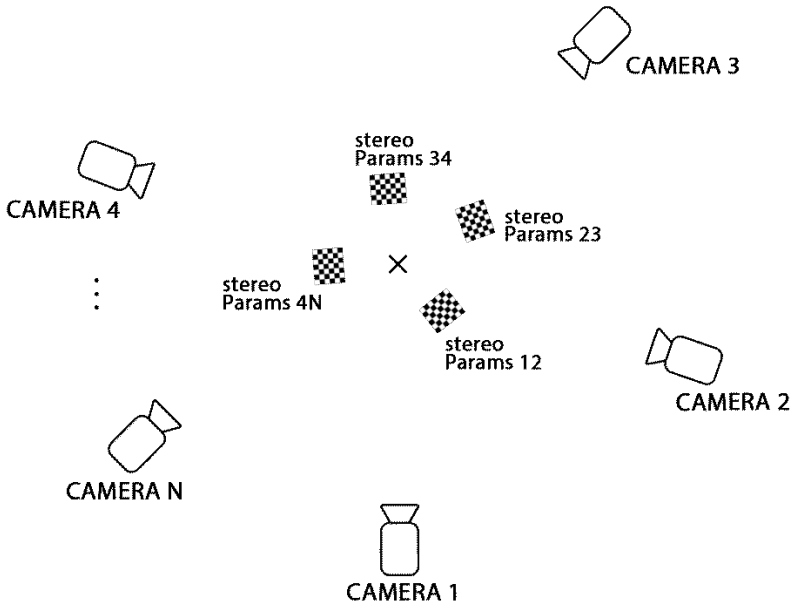

Figura 13: setup n camere

Si ottiene infine una matrice chiamata *'Location'* formata da tutti i vettori posizione delle camere rispetto alla prima, e una matrice 4D chiamata *'Orientation'* formata dalle varie matrici di rotazione riferite sempre alla prima camera. [8], [9].

2.4 Triangolazione:

Avendo acquisito i dati delle calibrazioni (mono e stereo) e i dati delle coordinate dei punti da Mediapipe si può passare alla fase di triangolazione, dalla quale si potranno ottenere le coordinate reali in 3D con unità di misura in metri (o millimetri) dei 33 punti del corpo per ogni frame. In realtà delle calibrazioni non sono necessari tutti i dati ma solo alcuni, in particolare dalle mono si estraggono i parametri intrinseci che vengono racchiusi in un vettore con un numero di componenti pari al numero di camere chiamato '*Intrinsics*', dalle stereo calibrazioni invece vengono estratte le matrici *'Location'* e *'Orientation'*.

La funzione utilizzata per questo calcolo è '*triangulateMultiview*', una funzione nata per un altro scopo, cioè riconoscere punti in comune e dare informazioni in tre dimensioni su di un soggetto fermo osservato da una camera che di volta in volta viene spostata in posizioni diverse. Sebbene sembri un'applicazione diversa in realtà è molto simile: avere un soggetto fermo o in movimento è irrilevante, poiché in entrambi i casi l'analisi

è svolta istante per istante, o meglio frame per frame, infatti nel caso di soggetto in movimento la funzione è sfruttata con un ciclo for eseguito per il numero di frame (*figura 14*).

```
pairsIdx=[(1:33)' (1:33)'];
for jj=1:frames
    vSet = imageviewset;
    points=SURFPoints(squeeze(poses_multi(:,:,jj,1)));
    vSet = addView(vSet,1,rigid3d(orientation(:,:,1),location(1,:)),'Points',points);

    for i = 2:n_cams
        points=SURFPoints(squeeze(poses_multi(:,:,jj,i)));
        vSet = addView(vSet,i,rigid3d(orientation(:,:,i), location(i,:)),'Points',points);
        vSet = addConnection(vSet,i-1,i,'Matches',pairsIdx);
    end

    tracks = findTracks(vSet);
    cameraPoses = poses(vSet);
    xyzPoints(:,:,jj)=triangulateMultiview(tracks,cameraPoses,intrinsics);
    clear vSet
end
```

Figura 14: script Matlab triangolazione

Volendo riassumere ogni passaggio dello script:

- '*pairsIdx'* crea una matrice con 33 righe contenenti i numeri da 1 a 33 e due colonne identiche, serve per numerare i punti che avranno una corrispondenza tra le viste di camere diverse;
- *'imageviewset'* crea un set vuoto in cui inserire le immagini e le informazioni sulle relative camere;
- '*SURFPoints'* è una funzione basata su un rilevatore di punti di interesse, ad esempio angoli o macchie, chiamato SURF (Speeded-Up Robust Featues) [10], in questo caso il suo utilizzo sarà estremamente semplificato poiché sono già presenti dati sulle coordinate dei punti di interesse che corrispondono ai 33 punti del corpo umano riconosciuti da Mediapipe;
- '*addView'* riempie '*vset'* aggiungendo le informazioni sulla camera e sulle coordinate dei punti;

Una volta completati questi passaggi per la prima camera, si passa al secondo ciclo for in cui verrà svolta la stessa analisi ma sulle altre camere:

- '*addConnection'* correla i punti corrispondenti delle relative camere;
- '*tracks'* estrapola le informazioni necessarie dei punti: le coordinate, il nome/numero del punto (da 1 a 33) e le viste in cui è presente (numero delle camere che lo vedono);
- '*poses'* riporta le posizioni e gli orientamenti delle camere relativamente alla prima;
- '*triangulateMultiview'* unisce tutte le informazioni ricavate per ottenere il file finale composto dalle coordinate in tre dimensioni; tutti questi passaggi sono ripetuti per ogni frame.

Il file ottenuto, chiamato nello script '*xyzPoints'*, contiene le coordinate dei 33 punti del corpo in ogni frame in tre dimensioni, riferite al sistema centrato nella prima camera. Partendo da questi dati si possono eseguire una serie di studi dipendenti dalla tipologia di applicazione. Bisogna precisare che per triangolare i dati è necessario che almeno due camere vedano il soggetto, in questo caso con "il soggetto" si indica semplicemente uno dei 33 punti, questo a volte può essere un problema principalmente per le prove con due camere poiché entrambe devono riconoscere ogni punto. Inoltre, la funzione '*triangulateMultiview'* si aspetta di avere come input numeri positivi, ciò vuol dire che non funziona quando ci sono degli '*NaN'* (not a number), quindi quando Mediapipe non riconosce una parte del corpo, in questi casi, come vedremo, sarà fondamentale la fase del filtraggio.

3. RISULTATI

Come è già stato spiegato, nonostante questa tecnica sarà applicata allo studio del nuoto, le prove sono state effettuate in laboratorio, i movimenti del soggetto sono stati a volte casuali a volte invece si è tentato di simulare il movimento di un nuotatore; il motivo principale è per la comodità di poter svolgere tentativi quasi giornalieri in modo da perfezionare l'analisi e da capire i punti deboli dei programmi utilizzati. Per le stesse motivazioni i primi test sono stati svolti con sole due camere, per poi aumentare il numero delle camere utilizzate fino a cinque. Inizialmente lo studio si è soffermato al riconoscimento e alla verifica del movimento, in seguito è stata svolta un'analisi cinematica e infine è stato dimostrato che camere identiche, anche se di qualità inferiore, danno risultati migliori, tramite una prova in cui veniva identificato lo spostamento di una sfera.

Caricati i dati precedentemente descritti, cioè quelli della calibrazione e di Mediapipe, prima di effettuare la triangolazione verrà svolto un passaggio intermedio fondamentale, ossia il filtraggio.

3.1 Filtraggio dei dati

Il filtraggio è una fase fondamentale quando si effettuano prove con probabile presenza di rumore. In questo caso sono presenti di due tipi di rumore: il primo è un semplice errore del software Mediapipe che potrebbe riconoscere in modo errato il corpo o una parte del corpo, il secondo invece riguarda la vera e propria assenza di dati, è probabile che Mediapipe non rilevi una parte del corpo semplicemente perché nascosta alla vista delle camere e questo avviene soprattutto per le prove in cui le camere sono solo due perché si limita molto la visuale generale, a differenza di prove con più camere in cui si può osservare il soggetto da tutte le angolazioni. Questa operazione ha un duplice scopo: modificare i dati presenti basandosi su quelli precedenti e successivi, cioè esegue una sorta di media, e ricostruisce i dati mancanti, anche in questo caso grazie alla conoscenza dei precedenti e successivi.

Esistono varie tipologie di filtraggio che utilizzano metodi diversi, per ognuna di queste l'input è costituito, oltre che dai dati da filtrare, dal passo, che stabilisce la quantità di valori su cui eseguire l'operazione ('*window',* finestra); un modo utile per decidere il valore numerico del passo è confrontarlo con il tempo del movimento, un passo troppo breve rischia di non ricostruire bene tutti i dati perché potrebbero mancarne molti consecutivi, invece un passo troppo ampio rischia di modificarli troppo e quindi di distorcere l'analisi.

```
passo_t=1;
framerate=30;
dt=1/framerate;
passo=passo_t/dt;
for j=1:n_cams
    for i=1:size(pos,1)
        a=squeeze(pos(i,1:2,:,j))';
        aa=smoothdata(a,'lowess',passo)';
        poses_multi(i,1:2,:,j)=aa;
    end
end
```

Figura 15: script Matlab filtraggio

L'esempio in *figura 15* è relativo a una prova in cui il soggetto ha imitato i movimenti di un nuotatore e il passo è stato quindi scelto pari a 1 secondo che è all'incirca il tempo del movimento della bracciata, come input il passo deve essere fornito in termini di numero di valori su cui eseguire le medie, quindi, lo si divide per '*dt'* che corrisponde al tempo equivalente di ogni frame e si ottiene il numero di frame contenuti nel tempo del passo.

Per la tipologia di filtro è stato scelto lo '*smooth lowess'* che calcola una regressione lineare sulla finestra definita, è un metodo leggermente dispendioso a livello computazionale ma implica meno discontinuità [11], in *figura 16* si può notare la differenza tra i dati filtrati e non. Tra gli altri metodi di *smooth*, ad esempio, uno dei più utilizzati è il '*gaussian*' che basa il filtraggio sulla funzione gaussiana.

Un'altra motivazione legata all'utilizzo dello *smooth* è il calcolo della velocità e dell'accelerazione: come si può vedere in seguito, in alcune prove è stata effettuata

l'analisi cinematica, in questo caso è necessaria un'operazione di filtraggio prima di svolgere le derivate per il calcolo delle velocità e successivamente delle accelerazioni.

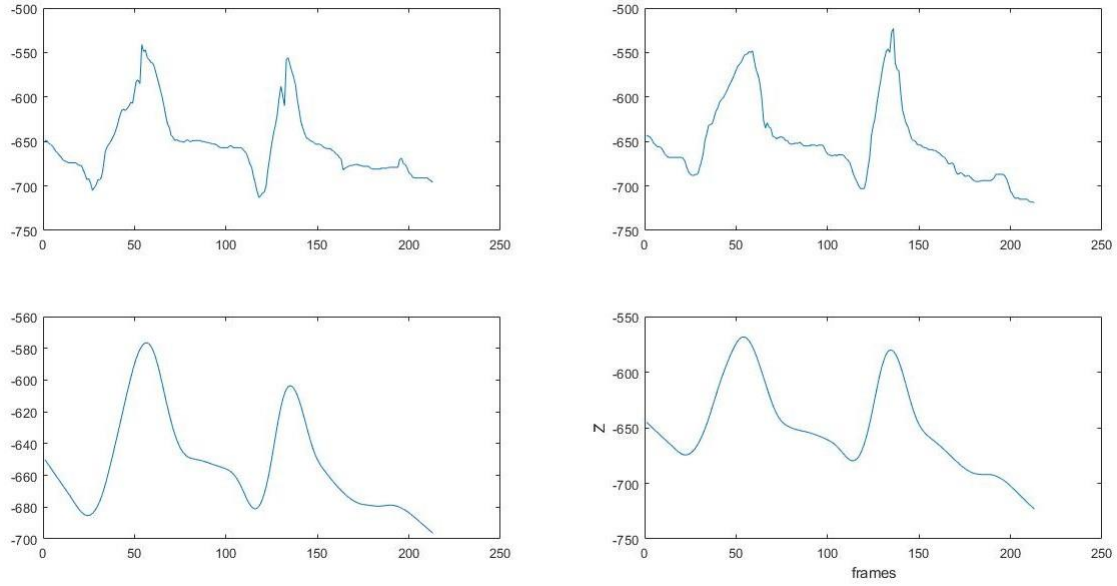

Figura 16: visualizzazione gomito destro, in alto i dati delle camere sinistra e destra, in basso gli stessi dati dopo il filtraggio

Come è stato accennato, la funzione Matlab per la triangolazione non funziona in presenza di '*NaN'* e questo è un grave problema, poiché in questa tipologia di prove è molto probabile che si perda almeno un dato, di conseguenza, il filtro assume un'importanza ancora maggiore perché ricostruisce le informazioni mancanti basandosi sulle precedenti e sulle successive; e in questo caso vale lo stesso per la scelta del passo, che definendo il numero di valori su cui effettuare i calcoli potrebbe non trovare abbastanza valori nel caso in cui una parte del corpo non viene rilevata per molto tempo. Bisogna porre molta attenzione sull'utilizzo del filtraggio e si dovrebbe cercare un compromesso tra un passo più piccolo possibile per non modificare in maniera esagerata i dati, ma grande abbastanza per ricreare tutti i dati assenti.

3.2 Prove con due camere

3.2.1 Osservazione movimento

Come è stato accennato i primi test sono stati svolti con sole due camere per approfondire il metodo e capire le cause e le possibili soluzioni dei vari problemi. Partendo dalla disposizione delle camere si è potuto prendere nota che se si trovano in una posizione troppo ravvicinata la stereo visione può dare problemi, ciò è causato dal fatto che le immagini delle due camere date in pasto al programma per la calibrazione sono troppo simili e di conseguenza si rileva molta fatica da parte del software che spesso non riesce a definire correttamente i parametri estrinseci (posizione e orientamento relativi). Tra le altre motivazioni legate al setup iniziale si trova anche la sincronia delle camere che non è banale, in queste prove è stato usato il telecomandino mostrato in *figura 4*, ma, nonostante ciò, poiché le camere sono diverse tra loro non è stato sempre garantito che l'inizio e la fine della ripresa siano stati nello stesso istante. La differenza era di pochissimi frame (5 o 6 di solito) che equivalgono a decimi di secondo ma in movimenti veloci questo può causare errori; pertanto, si provvedeva a modificare il file con numero di frame maggiore rimuovendo quelli in eccesso, assicurandosi che fossero all'inizio o alla fine del video. Altre difficoltà legate al software Mediapipe sono state descritte al *paragrafo 2.1*, tra queste la più rilevante è la centralità del soggetto nella scena, può succedere che Mediapipe non riconosca la persona se è situata nelle zone laterali dell'immagine e per questo motivo le camere sono stati disposte in modo da avere un'angolazione tale da centrare il soggetto rispetto all'inquadratura. L'ultima, ma più importante problematica, riguarda la diversità delle camere utilizzate; come è stato accennato, verrà dimostrato che camere identiche danno risultati più precisi (*paragrafo 3.4*), o meglio, che utilizzando camere differenti ci saranno piccole imprecisioni nella stereo calibrazione che, sebbene restino poco rilevanti, assumono un'importanza maggiore quando le camere sono più lontane. Nonostante la conoscenza di questo ostacolo, sono state utilizzate diverse tra loro perché si è preferito perfezionare il metodo prima di acquistare nuovi hardware.

Tra le prove svolte con due camere la più interessante riguarda il tentativo di imitazione del movimento dello stile libero di un nuotatore, in cui il soggetto era poggiato su una

sedia con le rotelle spostata tramite una fune, nella *figura 17* si può osservare la sequenza della bracciata. Le camere sono state posizionate come in *figura 18*, in cui si possono notare anche i limiti dello spazio osservato da entrambe.

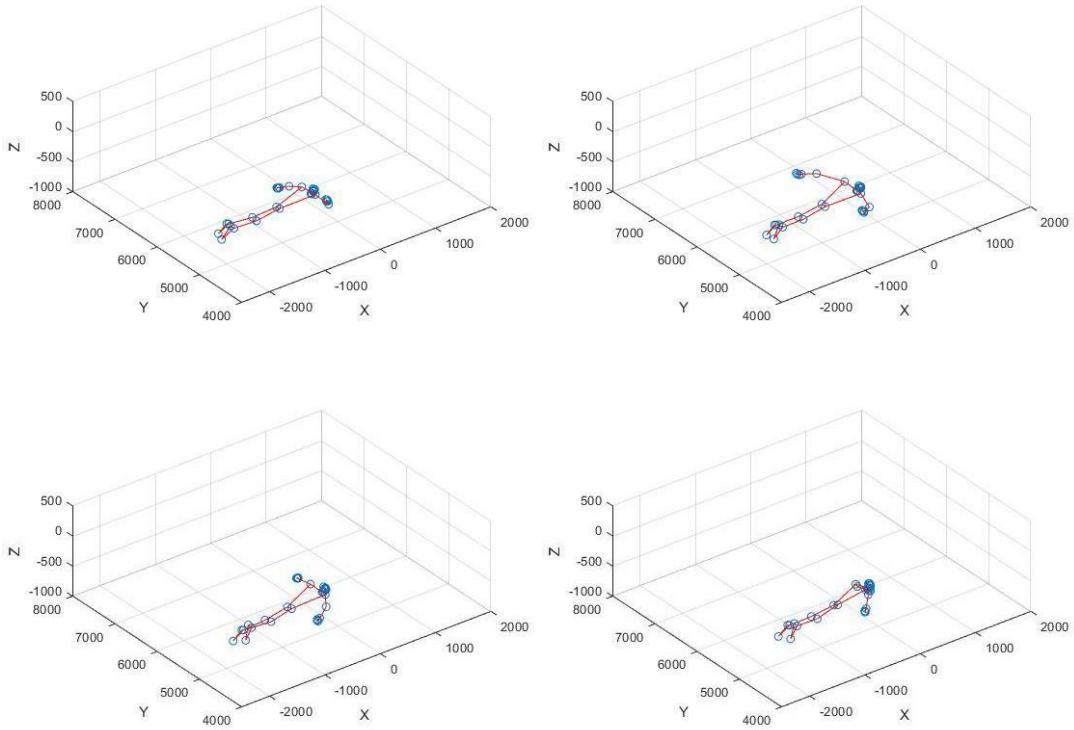

Figura 17: sequenza bracciata

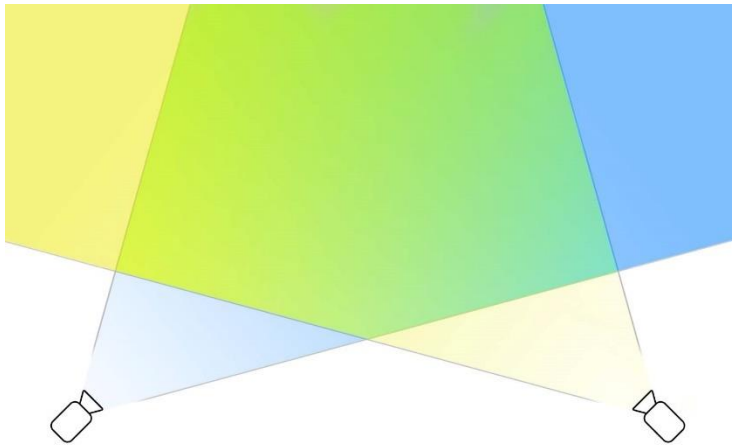

Figura 18: disposizione camere

Purtroppo, la visualizzazione non è sempre stata buona come questa, a causa della vicinanza delle camere; bisogna ricordare, infatti, che per la stereo calibrazione le due camere devono poter osservare contemporaneamente il checkboard, risulta, quindi, limitata la posizione relativa; volendo descrivere le posizioni paragonandole a una circonferenza centrata sul soggetto in movimento si può affermare che le camere al massimo possono distanziarsi di un angolo poco superiore ai 90°. A causa di questa problematica sono molti i dati persi e non sempre ricostruiti in modo corretto dal filtro, infatti, a volte il software ha confuso le due gambe, registrando in movimento la sinistra al posto della destra; un altro errore è legato al braccio sinistro nascosto, durante la bracciata può avvenire che non viene visualizzato da Mediapipe e il filtro non riesce a ricostruire completamente la bracciata, perché sostanzialmente ha perso dati sulla metà della bracciata stessa, come si può constatare dalla *figura 19* in cui il braccio sinistro è molto più lungo del normale.

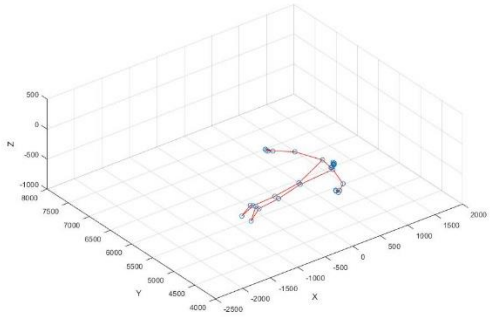

Figura 19: errore visualizzazione braccio sinistro

In una prova analoga si possono notare errori di visualizzazione simili, legati alla difficoltà nell'osservare la parte nascosta (sinistra) del corpo, come si evidenzia in *figura 20* in cui da una parte si osserva un errore grossolano nel riconoscimento della gamba, mentre dall'altra la confusione tra la parte destra e sinistra del corpo, vengono rilevati in movimento entrambi gli arti sinistri, quando in realtà in movimento era la gamba sinistra e il braccio destro.

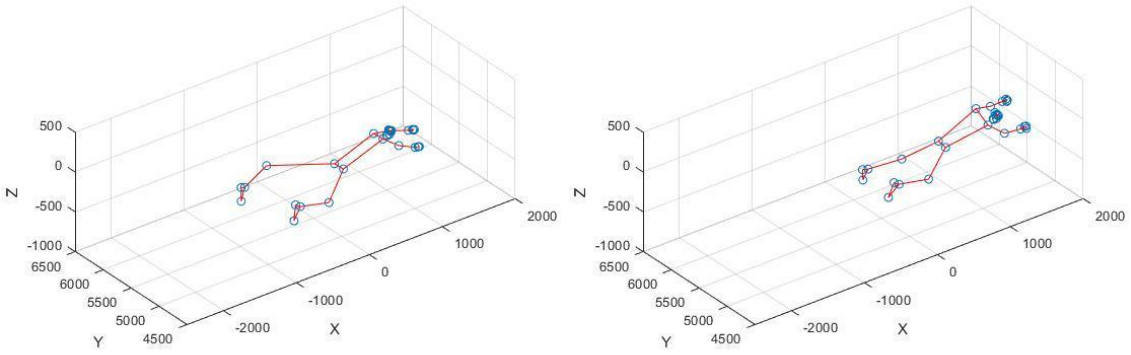

Figura 20: esempi di errori di rilevamento

3.2.2 Analisi cinematica

Per le prove appena mostrate è stata eseguita anche un'analisi cinematica: sono stati approfonditi alcuni aspetti legati a movimenti specifici di alcune parti del corpo ed è stato realizzato il calcolo di velocità e accelerazione. Rifacendosi alla prova di *figura 17* è stato analizzato in particolar modo il braccio destro poiché è la parte del corpo osservata più adeguatamente dalle camere. In *figura 21* è rappresentato lo spostamento lungo ognuno degli assi con unità di misura in millimetri: l'asse x è quello orizzontale, parallelo al movimento in avanti del soggetto, l'asse y indica la profondità, mentre il può rappresentativo in questo caso è l'asse z verticale, si intuisce l'andamento quasi sinusoidale tipico di una bracciata. L'analisi si ferma in questo caso a 160 frame circa che corrispondono al tempo di due bracciate.

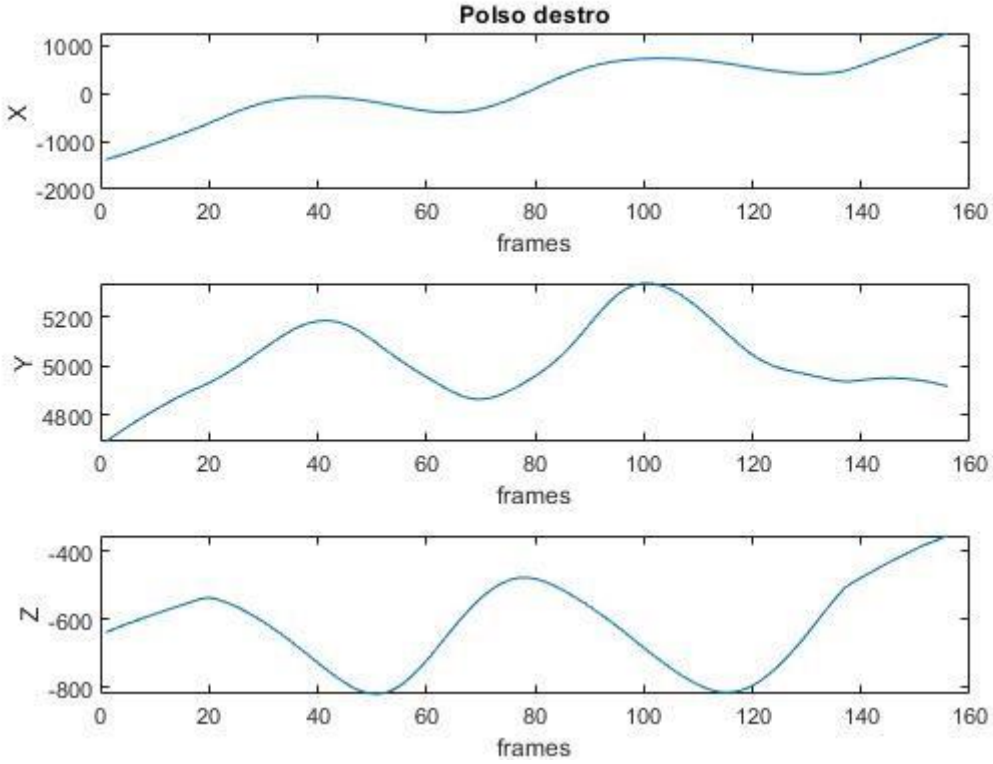

Figura 21: spostamento polso destro

Anche in *figura 22* è rappresentato il movimento del polso, in un grafico in due dimensioni in cui è pienamente visibile la bracciata, si può osservare che le due bracciate sono molto vicine rispetto all'andamento in avanti, questo si ha poiché la prova è una simulazione di un nuotatore su una sedia con le ruote, quindi, semplicemente le braccia sono state mosse più velocemente dell'andamento in avanti.

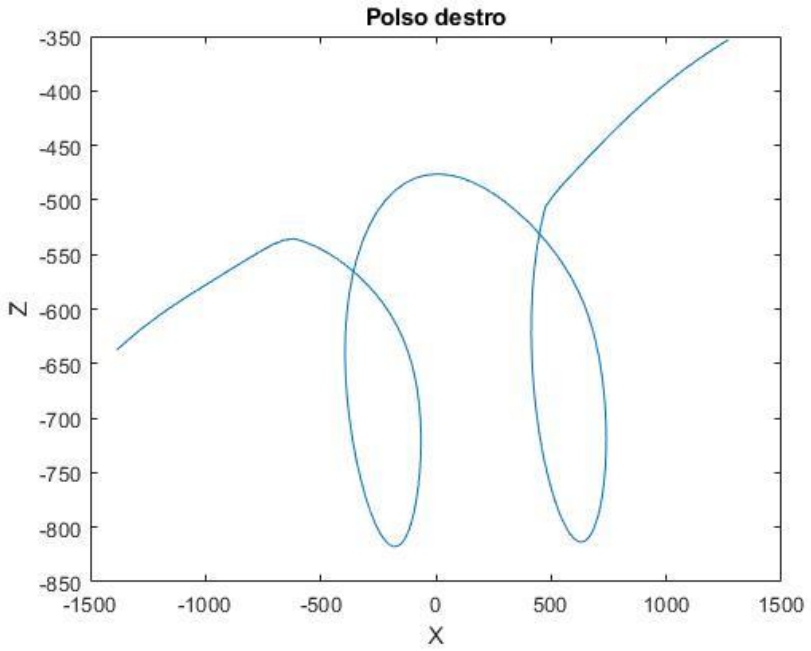

Figura 22: spostamento polso 2D

Una ricerca molto affascinante che si può effettuare con questa prova è controllare la differenza tra due cicli consecutivi, quindi, tra due bracciate consecutive. Ovviamente essendo una simulazione eseguita da un soggetto comune non ci si può aspettare che vengano identiche, tuttavia, il risultato è notevole, in *figura 23* sono mostrate le due bracciate consecutive in termini di posizione z (altezza) in funzione del numero di frame (tempo) e si può evidenziare che effettivamente i cicli sono molto simili, infatti, nella parte bassa dell'immagine è presente la differenza tra i due ed è al massimo di 10 cm nella parte iniziale.

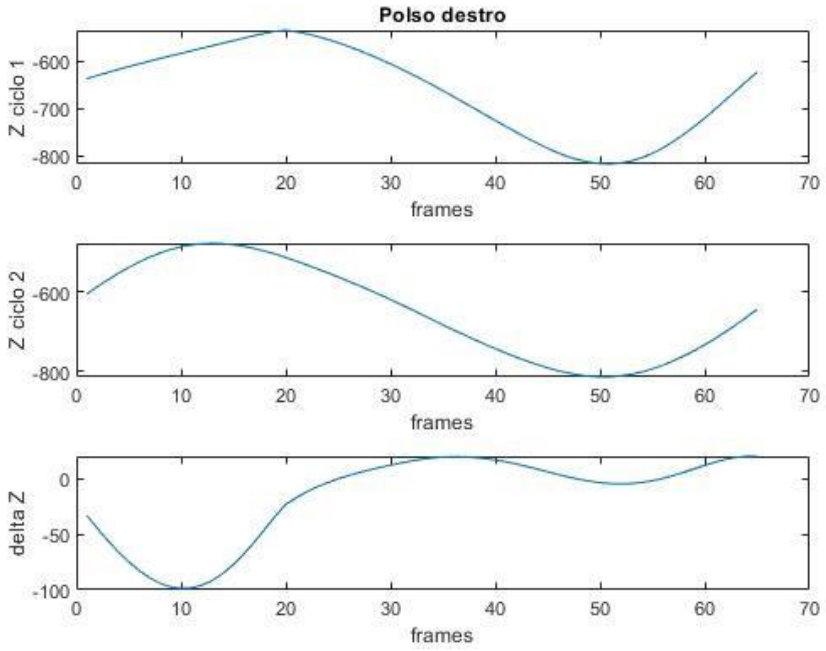

Figura 23: differenza cicli consecutivi

Infine, vengono presentati in *figura 24* grafici rappresentanti la velocità e l'accelerazione in relazione alla posizione. Per questi calcoli è stato fondamentale il filtraggio che ha reso la curva della posizione una effettiva curva eliminando il "zig-zag" tipico causato dall'acquisizione. Si possono notare i picchi di velocità durante la fase centrale dell'alzata e della discesa del braccio, mentre i picchi di accelerazione durante il cambio tra le due fasi. Si è scelto di rappresentare solo le componenti lungo l'asse verticale poiché è il più rappresentativo per il movimento della bracciata.

Grafici di questo tipo possono essere molto interessanti dati in mano a un allenatore di nuoto o ad un esperto, si potrebbe capire facilmente dove c'è stato un eventuale errore, soprattutto collegati alla velocità lungo l'asse x, che non è stata rappresentata in questo caso poiché corrisponde alla velocità con cui è stata tirata la fune, ma in una prova effettuata in piscina sarebbe un'informazione importantissima.

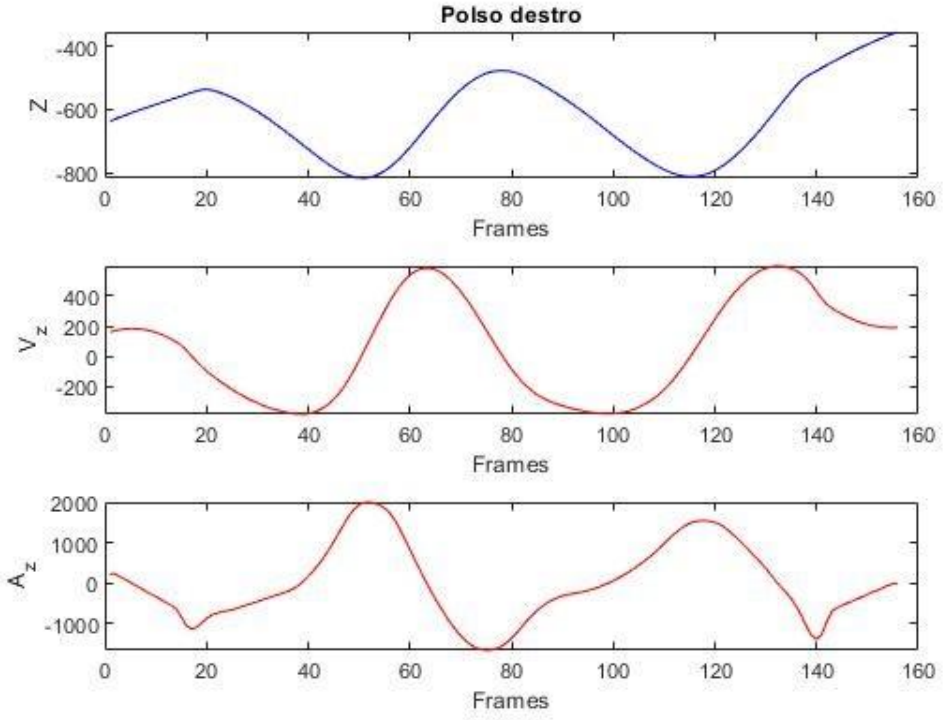

Figura 24: posizione velocità accelerazione

3.3 Prove multicamera

Con l'aumento del numero delle camere si hanno dei vantaggi molto interessanti, in primis la diminuzione, o alcuni casi anche l'assenza, dei dati persi, potendo circondare il soggetto con le camere lo si può osservare da ogni angolazione e non ci saranno parti del corpo nascoste a tutte le camere. Inoltre, aumentando il numero di camere ci si assicura la presenza di una grande ridondanza di dati, si ricordi che il numero minimo di camere per avere informazioni in 3D è due, tutte quelle in più sono in ridondanza. Avere informazioni in ridondanza è molto positivo, poiché i dati in ingresso su cui effettuare i calcoli sono molteplici, quindi, un eventuale errore ha un peso molto minore, che sia un errore di valutazione di Mediapipe o di ricostruzione sbagliata del filtraggio. Basti pensare che in una prova con sole due camere, un errore di valutazione da parte di una delle due compromette i risultati che di sicuro non saranno accettabili; più aumenta il numero delle camere e meno importante sarà la singola imprecisione, è chiaro però, che aggiungere una camera che non dia un risultato corretto sarebbe un errore ancora peggiore, infatti, a questo punto assume grande influenza il setup iniziale.

Di contro, ci sono anche degli svantaggi, in particolare a livello computazionale: essendoci una quantità di dati notevolmente superiore il software è rallentato, la funzione di triangolazione è più complessa, bisogna aggiungere la parte del calcolo matriciale che con sole due camere non era presente e per di più ne risulta peggiorato il problema dell'eventuale presenza degli '*Nan'*, poiché se le camere si trovano intorno al soggetto è molto probabile che per una (o più) di queste ci siano parti del corpo nascoste e quindi dati assenti che verranno ricostruiti grazie al filtro, è un passaggio inevitabile ma se ci sono dati persi e ricostruiti per molte camere si potrebbero ottenere dei risultati finali alterati.

3.3.1 Tre camere

Sono state effettuate una serie di prove con tre camere in cui si cercava di variare maggiormente il setup iniziale rispetto ai movimenti del soggetto, che sono stati spesso standardizzati per confrontare prove con setup diversi e capire quali possano essere i posizionamenti che facilitano il software e quali viceversa che danno problemi.

Il movimento tipico effettuato per capire se l'analisi fosse qualitativamente buona era costituito da alcuni passi fatti per formare un quadrato immaginario, in modo da verificare semplicemente l'accettabilità della prova, nel mentre venivano svolti dei movimenti con le braccia, anch'essi da analizzare.

In *figura 25* si può osservare il setup di una prova con buoni risultati, in cui si hanno le camere poste alla stessa altezza, due delle quali su di una barra metallica, la terza su di un cavalletto, orientate verso il soggetto in modo da averlo al centro di ognuna delle tre inquadrature.

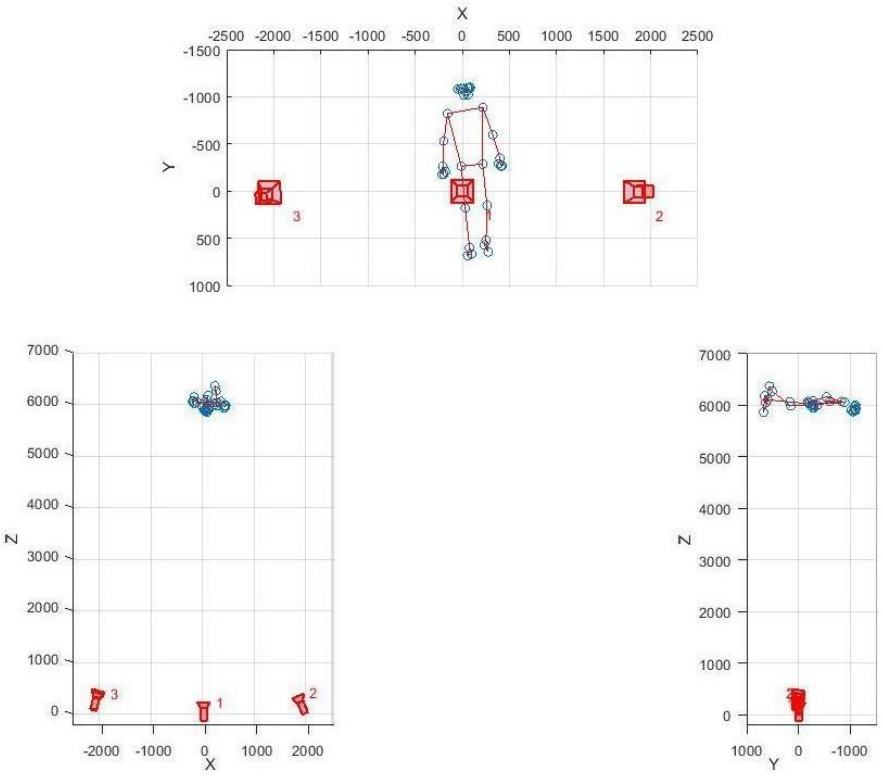

Figura 25: punti di vista setup iniziale

Sono stati eseguiti, come accennato, dei passi per formare un quadrato ed è stato esaminato il movimento di uno dei punti della testa, in particolare il naso, per accertarsi della veridicità dell'analisi. In *figura 26* è presentato il movimento del naso durante tutta la prova, in una visualizzazione in 2D con punto di vista dall'alto.

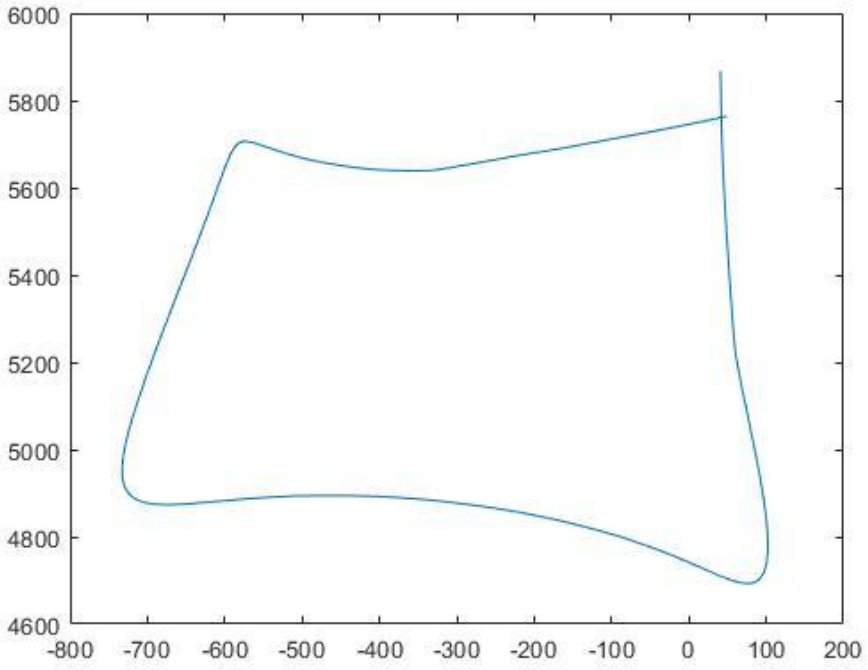

Figura 26: analisi movimento del naso

In questo grafico l'unità di misura è in millimetri, infatti si può osservare che il passo è di circa 80 cm che è molto verosimile. Lo spostamento del naso non raffigura esattamente un quadrato, però, considerando che a livello fisico è difficile non muovere la testa al di fuori del quadrato immaginario che si vuole seguire, e che la differenza con un quadrato perfetto è dell'ordine di una decina di centimetri si può valutare la prova con successo.

Nella parte iniziale della prova è stato compiuto anche un gesto con il braccio, si è alzato in segno di saluto e riabbassato, un movimento all'apparenza banale ma conveniente per appurare il sistema di rilevamento. In *figura 27* si osserva la sequenza del braccio in 2D, a sinistra è illustrato l'innalzamento del braccio mentre a destra la discesa. Sono stati estratti 10 frame dai primi 50, essendo state effettuate le riprese con 30 frame al secondo, si ha un tempo totale impiegato per questo movimento di poco superiore al secondo, quindi, considerando anche la velocità del movimento si può affermare che la prova sia stata molto positiva. In *figura 28* invece si può notare la stessa sequenza ma in 3D con tutto il corpo.

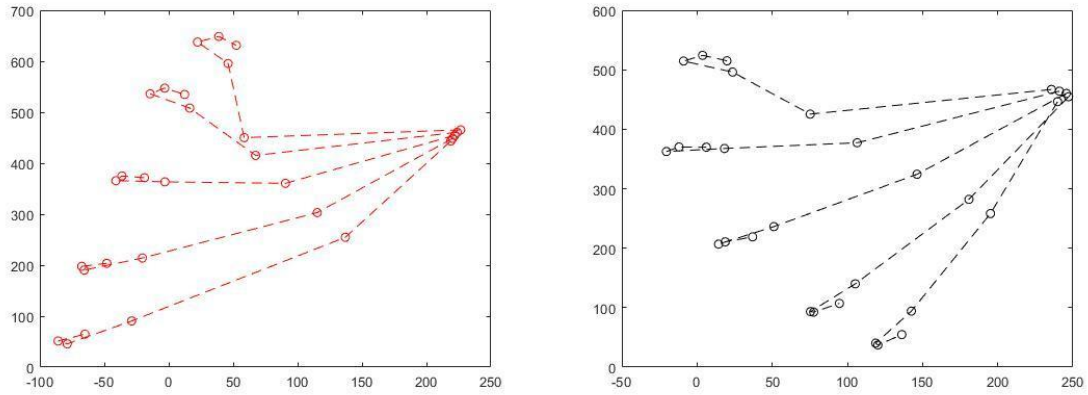

Figura 27: visualizzazione movimento braccio 2D

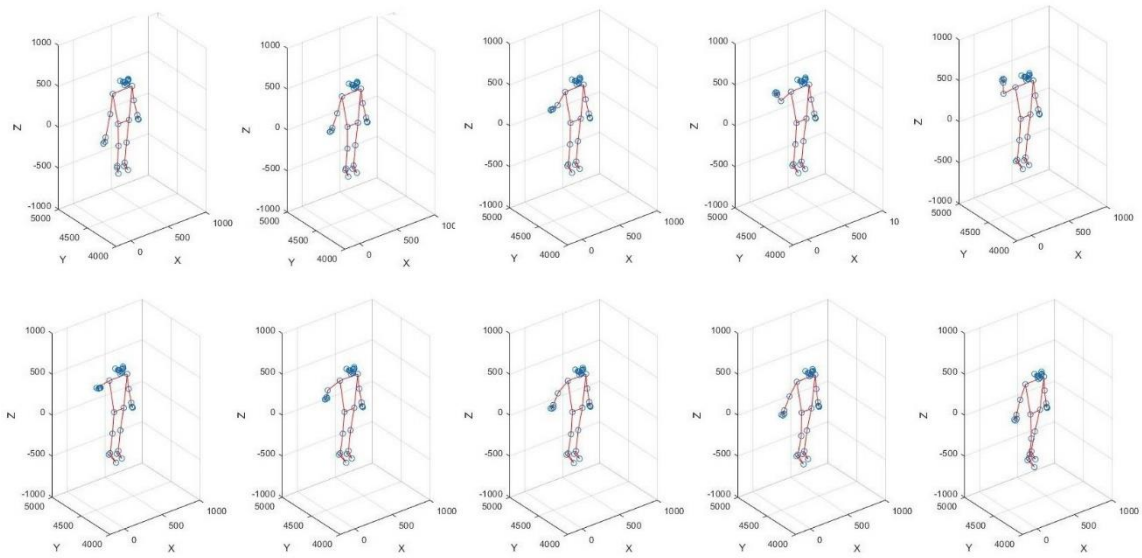

Figura 28: visualizzazione movimento braccio 3D

Bisogna però sottolineare che in questa prova è stato utilizzato un setup delle camere molto conveniente, inoltre, con il soggetto rivolto verso di essere è facilitato il lavoro dei software. Successivamente si è tentata una prova con movimenti non del tutto diversi ma con setup differente, come si può notare in *figura 29*. Le camere sono state disposte con un'angolazione di 90° l'una con l'altra per avere una visione quasi su tutta la circonferenza del soggetto.

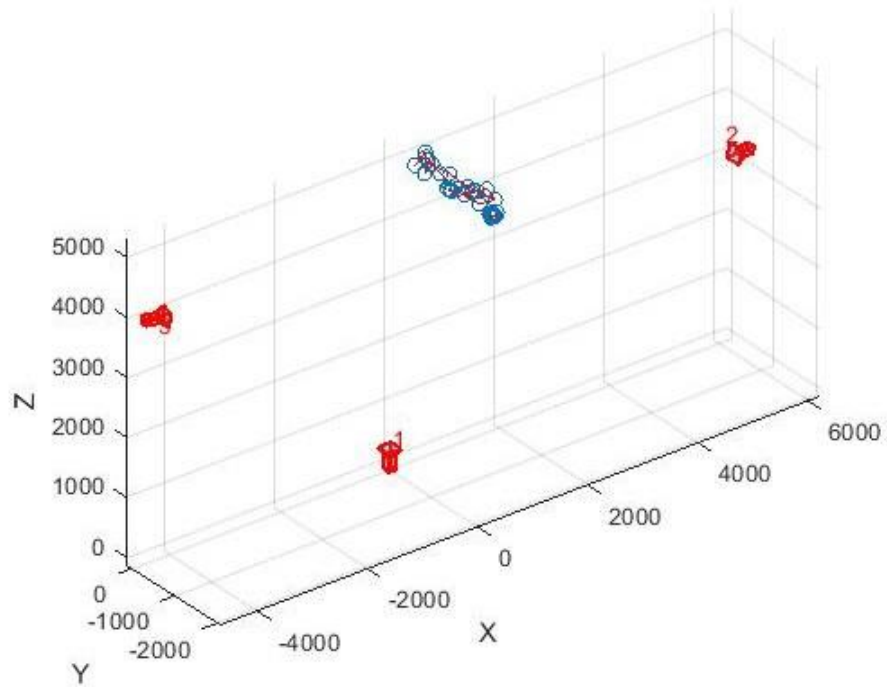

Figura 29: setup iniziale

Questo posizionamento non ha prodotto i risultati sperati, le analisi non sono completamente errate ma in confronto alla precedente non paragonabili; sono state effettuate cinque prove con questo setup con movimenti simili ma con piccole differenze, in due delle quali il soggetto si è rivolto verso le camere numero 2 e 3 ma i risultati sono analoghi: i movimenti registrati non sono lineari ma leggermente obliqui, in *figura 30* (punto di vista dall'alto) è mostrato l'andamento del naso in due prove diverse, in cui il soggetto si è spostato rispettivamente in verticale e in orizzontale, quindi, l'andamento del naso sarebbe dovuto essere idealmente una linea retta che andava e tornava da un estremo all'altro.

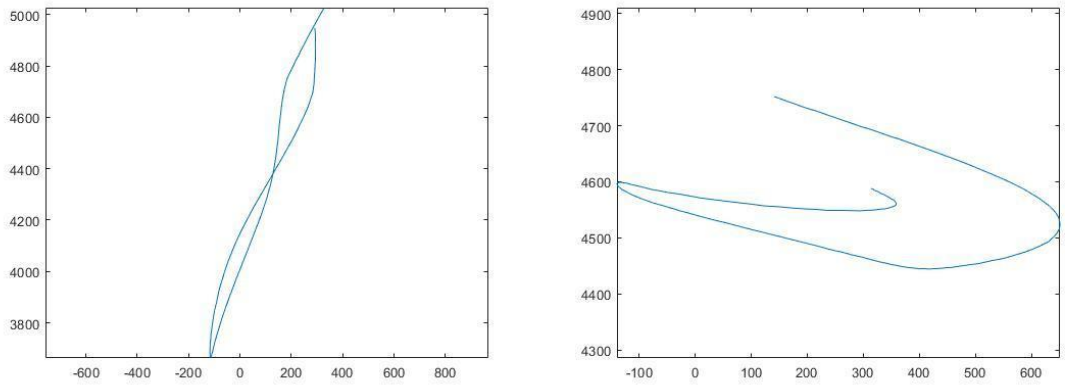

Figura 30: visualizzazione errata movimento naso

Questi risultati dimostrano quanto può influire il setup iniziale, inoltre, è necessario specificare che il sistema di riferimento, oltre ad essere fissato al centro della prima camera, è anche orientato come quest'ultima, quindi, se è stata posizionata leggermente inclinata il sistema di riferimento sarà allo stesso modo inclinato e i risultati anche se qualitativamente sono ottimi, non sono poi facilmente analizzabili poiché non coerenti con gli assi. Una piccola inclinazione può sembrare impercettibile e ininfluente se si osserva direttamente la camera, ma in un'inquadratura di più metri quella piccola inclinazione risulta notevole. L'altra causa di queste imprecisioni è la diversità delle camere che in queste prove ha più rilevanza data la distanza tra di esse.

3.3.2 Cinque camere

Sono state effettuate due prove con cinque camere con lo stesso setup iniziale, purtroppo in queste prove ci sono stati enormi limiti legati all'hardware che hanno causato pessimi risultati. Le camere a disposizione avevano parametri molto differenti tra loro, tra cui il più importante è il numero di frame al secondo per la registrazione dei video, quindi, per arrivare ad avere un numero di dispositivi pari a cinque sono stati utilizzati tre cellulari con camera in modalità pro. Si ricordi che è necessario per effettuare la stereo calibrazione e la triangolazione che ci sia lo stesso numero di frame e la stessa dimensione dell'immagine, quindi stessa risoluzione. Anche se i parametri di ognuna delle camere sono stati settati e non modificati durante le prove, non essendo identici sono sopraggiunti problemi nella fase della stereo calibrazione, in particolare il software non riesce a registrare correttamente le posizioni e le angolazioni relative.

Nella *figura 31* si può osservare il setup delle prove, le camere sono state posizionate all'incirca alla stessa altezza da terra, invece, nell'immagine si nota come le distanze reciproche sono di gran lunga sbagliate, si arriva addirittura ad una differenza di altezza tra la prima e la quinta camera (piano Y-Z) superiore ai 4 metri.

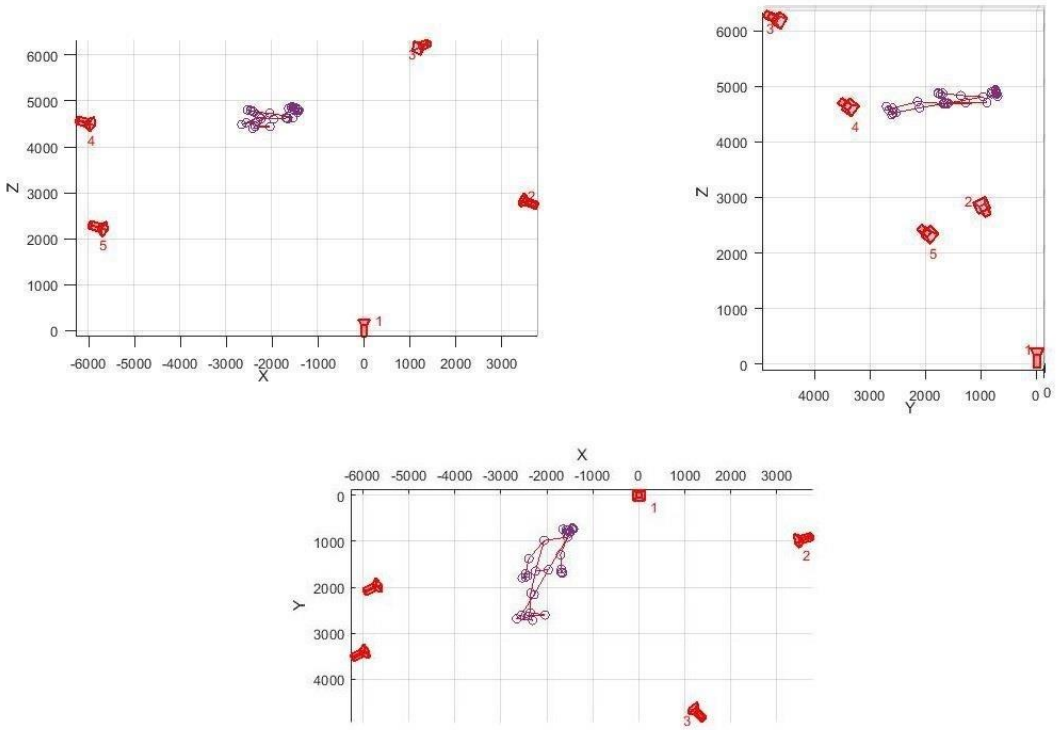

Figura 31: setup 5 camere

Nelle prove presentate con tre camere questa problematica era molto meno influente per due motivi: sia per il posizionamento delle camere che si erano trovate molto più ravvicinate, ad esempio un errore relativamente piccolo su un metro di distanza diventa rilevante quando la distanza è di 5 o anche più metri ed è esattamente quello che è successo; e sia poiché in questa prova sono state utilizzate camere tutte totalmente diverse tra di loro, invece nelle prove con tre camere due di esse erano molto simili mentre la terza leggermente diversa, aggregando queste difficoltà viene fuori un risultato purtroppo non accettabile.

3.4 Verifica spostamento sfera

Come si è potuto constatare la difficoltà più rilevante nei test illustrati è stata la diversità delle camere, sono state effettuate due prove per dimostrarlo in cui si è misurato lo

spostamento di una sfera, nella prima con un setup classico simile alle prove precedenti e con camere diverse, nella seconda usando altre quattro camere identiche tra loro ma di bassa qualità montate su un braccio meccanico, quindi anche con posizione relativa fissa (*figura 32*).

Figura 32: camere su braccio meccanico

La scelta è ricaduta sulla sfera come soggetto poiché si è sfruttato un pacchetto del software Labview capace di estrapolare le coordinate del centro di una sfera e il suo raggio, in pixel, in un'immagine (2D), basandosi sul contrasto di colori rispetto allo sfondo, infatti, nella seconda prova si è utilizzata una sfera di materiale riflettente per migliorare la visualizzazione. È un metodo di misura molto accurato, inoltre, si può visualizzare qual è la circonferenza trovata dal software ed eventualmente modificare, quindi, il livello di precisione diviene notevolmente elevato. Per entrambe le prove sono state utilizzate quattro camere (appositamente calibrate) che hanno ripreso la sfera posizionata in più punti con spostamenti misurati manualmente, successivamente le immagini sono state analizzate con Labview per acquisire le coordinate dei centri della sfera e infine utilizzando Matlab è avvenuta la triangolazione per ottenere le

informazioni in tre dimensioni e confrontarle con gli spostamenti misurati precedentemente.

Per quanto riguarda la prima prova è stata utilizzata la sfera in *figura 33*, con un diametro di circa 25 cm e con un vestito nero dietro per migliorare il contrasto e quindi la precisione di misura.

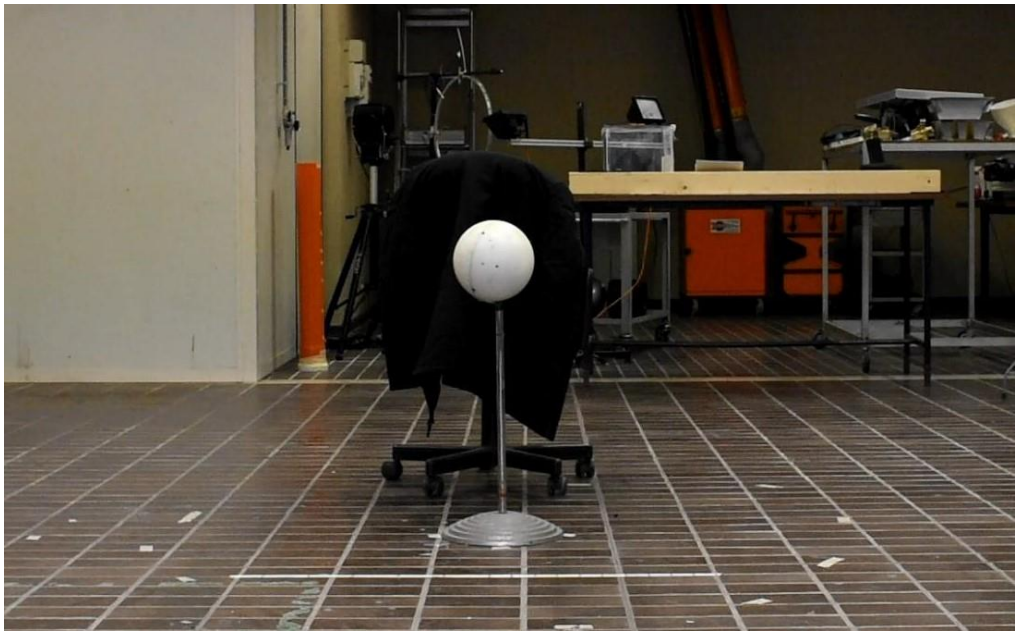

Figura 33: sfera bianca con sfondo scuro

La sfera è stata posizionata in dieci punti diversi e sono stati misurati i nove spostamenti, in *figura 34* si può osservare la visualizzazione di tutte le posizioni della sfera e delle quattro camere (si ricordi che l'immagine è ruotata rispetto alla realtà a causa del sistema di riferimento delle figure di Matlab). Per ottenere le informazioni in 3D è stata utilizzata la stessa funzione Matlab per triangolare ('*triangulate*') e da queste si è calcolato lo spostamento della sfera in immagini successive; in *figura 35* è mostrata la tabella con lo spostamento misurato dal software, quello reale, la differenza e il rapporto in percentuale tra la differenza e quello reale. Gli spostamenti sono stati effettuati in direzioni diverse, anche se in tabella non è specificata quale; per lo spostamento rilevato dal software si è scelto di considerare non la singola componente ma la somma vettoriale delle componenti per aggirare l'eventuale errore dato dall'inclinazione diversa degli assi del sistema di riferimento Matlab da quello reale. Si può notare che l'errore percentuale si aggira intorno al 5-10% che è rilevante

considerando che, trattandosi di spostamenti di un metro circa, difficilmente su una misura del genere l'errore umano può essere di una decina di centimetri, non sarebbe verosimile; e infatti andando a verificare l'attendibilità della stereo calibrazione misurando manualmente le distanze tra le camere si hanno dei valori leggermente diversi da quelli ottenuti dal software, questa è la causa principale dell'errore percentuale così elevato. Inoltre, poiché le distanze tra le camere sono notevoli, bisogna precisare che le stereo calibrazioni sono state effettuate con due camere per volta e di conseguenza un piccolo errore in ogni stereo calibrazione si riflette nella successiva amplificando l'imprecisione.

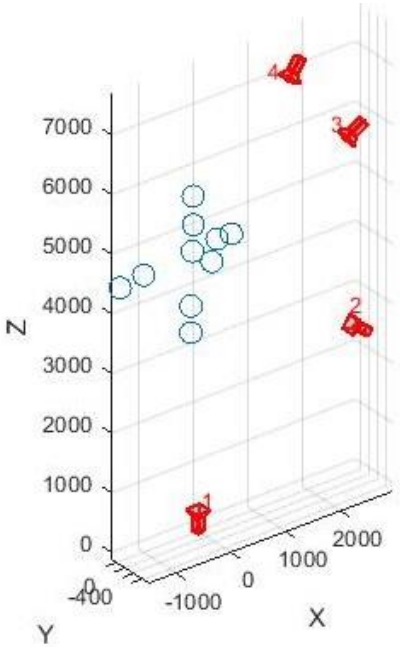

Figura 34: posizione sfera più camere

spostamento misurato [mm]	spostamento reale [mm]	delta spostamento [mm]	rapporto percentuale [%]
899,68	1000,00	100,32	10,03
940,89	1000,00	59,11	5,91
1339,00	1500,00	160,99	10,73
1404,90	1500,00	95,11	6,34
718,17	800,00	81,83	10,23
452,62	500,00	47,38	9,48
959,01	1000,00	40,99	4,10
461,32	500,00	38,68	7,74
405,77	450,00	44,23	9,83

Figura 35

Con la seconda prova è stata dimostrata questa tesi: sono state utilizzate quattro camere identiche tra loro (di qualità inferiore), con la differenza che in questo caso le distanze erano molto inferiori e anche la sfera ovviamente molto più piccola (circa 1 cm di diametro, *figura 36*). Date le dimensioni, l'errore umano per lo spostamento della sfera, trattandosi di movimenti di qualche centimetro, avrebbe potuto essere di gran lunga maggiore in percentuale, ma, nonostante ciò, è molto minore. La sfera è stata poggiata su una guida per migliorare la precisione dei movimenti e le distanze sono state misurate con un calibro. Grazie alla tabella in *figura 37*, analogamente alla precedente, si possono osservare i valori. L'errore percentuale arriva al massimo a poco più del 2%, molto meno rispetto alla prova precedente, questo risultato è importantissimo poiché dà la prova che il motivo principale delle varie imprecisioni in tutte le prove precedenti è la diversità delle camere.

Figura 36: sfera seconda prova

Spostamento misurato [mm]	Spostamento reale [mm]	delta spostamento[mm]	rapporto percentuale [%]
47,43	47,20	0,23	0,49
119,76	120,60	0,84	0,70
64,42	63,90	0,52	0,81
111,74	110,80	0,94	0,85
25,32	25,00	0,32	1,28
51,19	50,20	0,99	1,97
68,36	70,00	1,64	2,34
82,34	83,00	0,66	0,80

Figura 37

4. CONCLUSIONI

La tecnica illustrata non è ancora utilizzabile a pieno poiché è in fase di studio e deve essere affinata, ma le prospettive future sono radiose. Tra i primi aspetti da migliorare rispetto alla presente tesi c'è sicuramente la parte hardware; invece, per quanto riguarda la parte software da un lato si possono snellire e alleggerire i programmi, dall'altro si possono potenziare (e quindi sicuramente appesantire) cercando di modificare le parti in cui si trovano difficoltà, ad esempio il peso diverso che si potrebbe dare ad alcuni dati rispetto ad altri in fase di triangolazione, si potrebbe dare meno importanza ai dati ricostruiti dal filtro, oppure non considerare una camera se non vede alcuni parti del corpo (se ha *NaN*). Bisogna sempre ricordare che i video andranno svolti in una piscina e, quindi, aggiungere le problematiche del caso. C'è molto lavoro ancora da fare ma si potrebbe arrivare ad un punto in cui questa tipologia di analisi verrà aggiunta al classico allenamento fornendo una grande quantità di informazioni ad esempio su velocità, accelerazioni, angolazioni di braccia o polso, differenza tra bracciata destra e sinistra, posizionamenti delle gambe, insomma, una serie di dati che potranno essere fondamentali per migliorare le prestazioni in futuro. Inoltre, questo metodo potrebbe ovviamente essere applicato in altri ambiti, restando nel settore sportivo, studi di questo tipo possono essere utili in ogni sport, in particolare a livello atletico. Concludendo ci si può definire soddisfatti del lavoro svolto e curiosi dell'avvenire.

BIBLIOGRAFIA

[1] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1318–1334, Oct. 2013, doi: 10.1109/TCYB.2013.2265378.

[2] X. Chang, Z. Ma, M. Lin, Y. Yang, and A. G. Hauptmann, "Feature Interaction Augmented Sparse Learning for Fast Kinect Motion Detection," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3911–3920, Aug. 2017, doi: 10.1109/TIP.2017.2708506.

[3] "Mediapipe." https://google.github.io/mediapipe/solutions/pose

[4] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," *In Pract.*, vol. 7, no. 1, pp. 1–16, 2006, doi: 10.1.1.117.6808.

[5] A. Fetić, D. Jurić, and D. Osmanković, "The procedure of a camera calibration using Camera Calibration Toolbox for MATLAB," 2012.

[6] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proceedings of the IEEE International Conference on Computer Vision*, 1999, vol. 1, pp. 666–673. doi: 10.1109/iccv.1999.791289.

[7] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000, doi: 10.1109/34.888718.

[8] J. Heikkila and O. Silvcn, "A Four-step Camera Calibration Procedure with Implicit Image Correction."

[9] C. Gui and L. Tu, "A stereo camera calibration based on robotic vision," in *Proceedings of the 10th IEEE International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2011*, 2011, pp. 318–323. doi:

10.1109/COGINF.2011.6016159.

[10] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-Up Robust Features (SURF),” *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008, doi: 10.1016/j.cviu.2007.09.014.

[11] “smoothdata”: https://it.mathworks.com/help/matlab/ref/smoothdata.html?s_tid=doc_ta

RINGRAZIAMENTI