Capitolo 5

Applicazioni sviluppate

È possibile, dopo aver ottenuto nei capitoli 2 e 4 una conoscenza più specifica dei componenti hardware e software, ricavare uno schema generale più accurato dell'intero sistema di asservimento visuale (rispetto ai precedenti schemi delle figure 1.1 e 2.1).

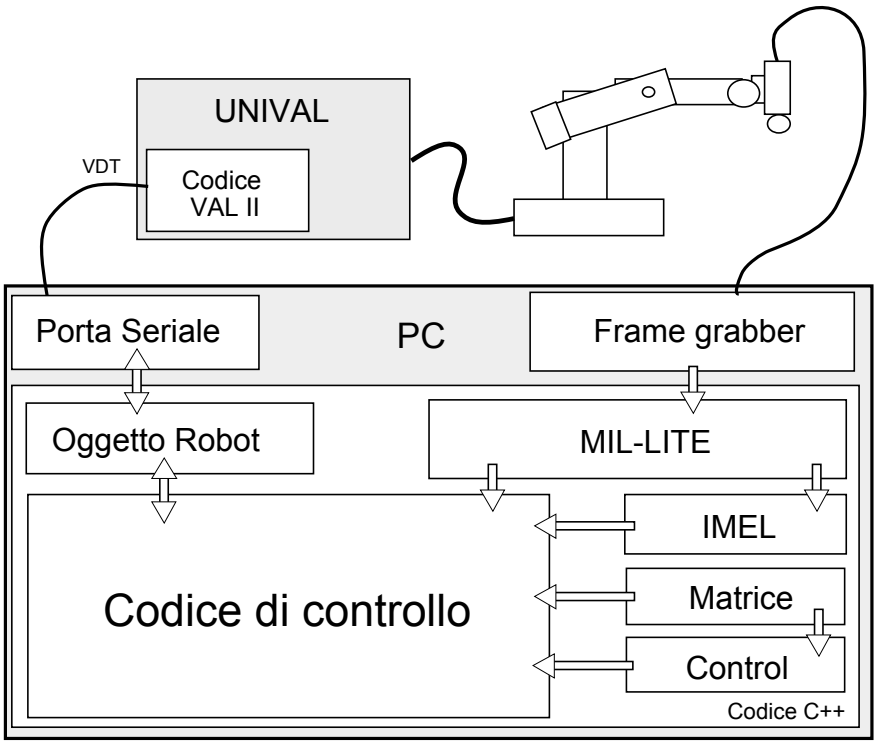


Figura 5.1: Schema funzionale dell'intero sistema di visual servoing.

In figura 5.1 vengono evidenziati tutti i blocchi (logici e fisici) che costitu-

iscono il sistema, le frecce tra i vari blocchi indicano la direzione del flusso di informazioni, ad esempio, essendo un organo sensoriale passivo, la telecamera invierà solo dati verso il PC senza riceverne e questo viene indicato da una freccia unidirezionale ($\Rightarrow$), al contrario il controllore Unival oltre che ricevere i set point di posizione (attraverso la seriale) dal PC invierà a quest'ultimo un prompt di attesa comandi e questo viene indicato da una freccia bidirezionale ($\Leftrightarrow$).

L'alta modularità dell'intero sistema permette di mantenere in entrambe le applicazioni realizzate il medesimo schema funzionale e solo i codici dei blocchi *"Codice di controllo"* e *"Codice Val II"* cambiano in funzione dell'applicazione implementata (vedere appendici F e D).

Di seguito verranno presentate le due applicazioni sviluppate descrivendo brevemente gli obiettivi prefissi ed evidenziando per entrambi i casi il diagramma di flusso del *codice di Controllo*.

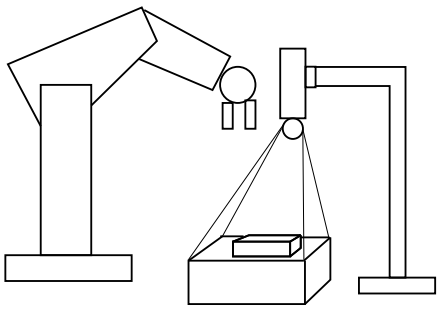


Figura 5.2: Applicazione ad anello aperto.

5.1 Applicazione ad anello aperto "pick and place"

Questo tipo di applicazione[1] utilizza l'architettura di tipo *"look and move"* (vedere 1.3). La telecamera è fissa e posta ad una distanza (imprecisata) ma in una posizione tale da riprendere il piano di lavoro, in queste condizioni la calibrazione della telecamera avviene attraverso la procedura indicata in 3.4.1, basandosi su una immagine di tre riferimenti (R_1, R_2, R_3 espressi in coordinate WORD) posti sul piano di lavoro; tali riferimenti, per determinare una buona calibrazione, dovranno formare possibilmente un triangolo rettangolo con i cateti di lunghezze diverse.

[1] Per il codice vedere l'appendice F.1.2.

5.1.1 Procedura di calibrazione

Per capire quale punto dell'immagine associare ad un dato riferimento è stato implementato un algoritmo[2] che calcola le distanze reciproche tra i 3 punti (r_1, r_2, r_3) presenti nell'immagine ripresa, per cui si ottengono 2 distanze $(d_1(r_i), d_2(r_i))$ per ognuna delle 3 immagini (r_i con $i = 1..3$) per un totale di 6 misure che vengono poi normalizzate rispetto alla più piccola ($min(d_j(r_k))$ con $j = 1..2$ e $k = 1..3$).

Applicando lo stesso procedimento alle distanze assolute (espresse nello spazio del robot e note a priori) si ottengono altre 2 misure $(D_1(R_i), D_2(R_i))$ per ognuno dei 3 riferimenti R_1, R_2, R_3, tali misure vengono normalizzate rispetto al $min(D_j(R_k))$ con $j = 1..2$ e $k = 1..3$.

A questo punto una immagine r_j è associata ad un certo riferimento R_k se ha le sue stesse distanze normalizzate[3] cioè se

$$\begin{cases} d_1(r_j) &= D_1(R_k) \\ d_2(r_j) &= D_2(R_k) \end{cases}$$

Questo algoritmo è quindi in grado di rilevare se gli oggetti inquadrati dalla telecamera sono effetivamente i riferimenti presenti sul piano di lavoro ed in più, se opportunamente applicato, può riuscire ad estrarre da un immagine con più di 3 oggetti quali di questi sono associati ai riferimenti[4] e quali sono da considerarsi *"disturbi"* sul processo di calibrazione.

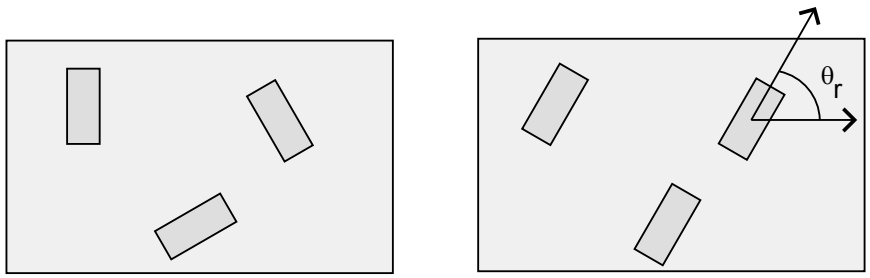


Figura 5.3: Esempio di allineamento di 3 oggetti.

[2] Vedere procedura *precalib()* appendice F.4.

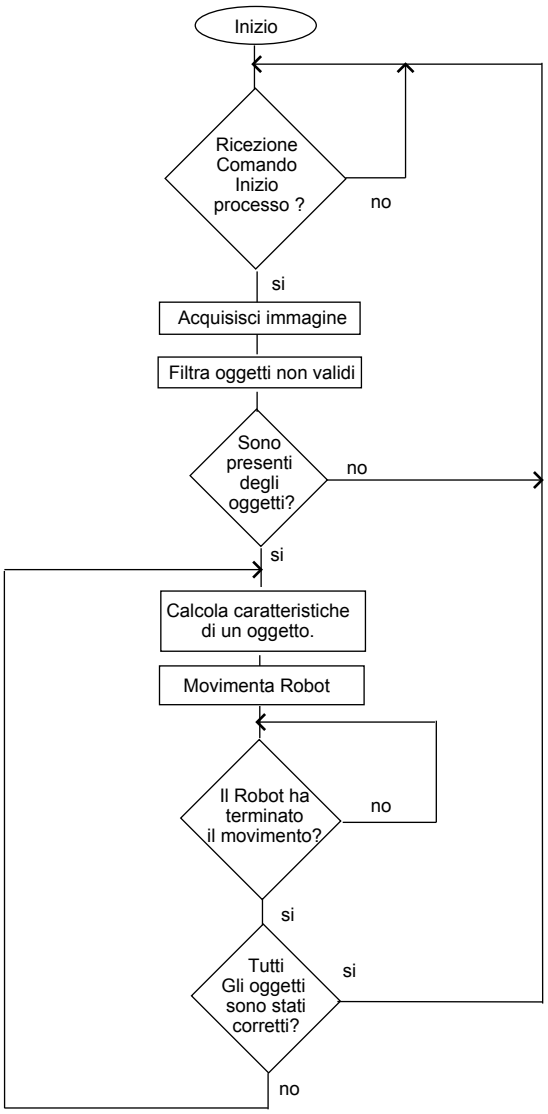
[3] Questo discorso è valido nell'ipotesi in cui la telecamera non introduca una grossa distorsione tra asse x ed y, la Jai CV-m10 ha una distorsione trascurabile.

[4] Questo funzionalità non è però stata implementata.

5.1.2 Obiettivo dell'applicazione

Una volta calibrata la telecamera e posizionati n oggetti sul piano di lavoro il sistema elabora **una sola immagine** che giunge dalla telecamera e movimenta il robot affinche faccia assumere a tutti gli n oggetti la medesima inclinazione θ_r[5], come mostrato dall'esempio di figura 5.3.

Terminato il compito il sistema è in grado di rielaborare un'altra immagine ed eseguire con altri m oggetti la medesima azione. Questa sequenza di operazioni viene realizzata eseguendo i passi indicati dal successivo diagramma di flusso



[5]Questi oggetti potrebbero ad esempio essere introdotti da un rullo sincronizzato opportunamente con il processo.

Il diagramma è stato implementato in C++ nel file `OPENLOOP.CPP` che rappresenta il *Codice di controllo* di figura 5.1.

5.2 Applicazione ad anello chiuso "Eye in hand"

Anche questo tipo di applicazione[6] utilizza l'architettura di tipo *"look and move"* di cui si è trattato nel paragrafo 1.3. La telecamera è solidale con l'organo terminale del robot, per cui, differentemente dall'applicazione precedente, se ne conosce la posizione precisa nello spazio del robot.

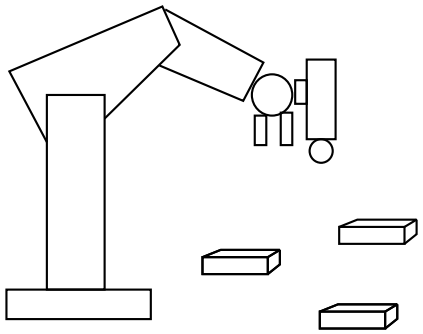


Figura 5.4: Applicazione ad anello chiuso.

5.2.1 Calibrazione telecamera

Effettuando comunque una calibrazione della telecamera attraverso il procedimento descritto nel paragrafo 3.4.2 si rende il sistema indipendente dall'orientamento della telecamera montata sull'end-effector[7].

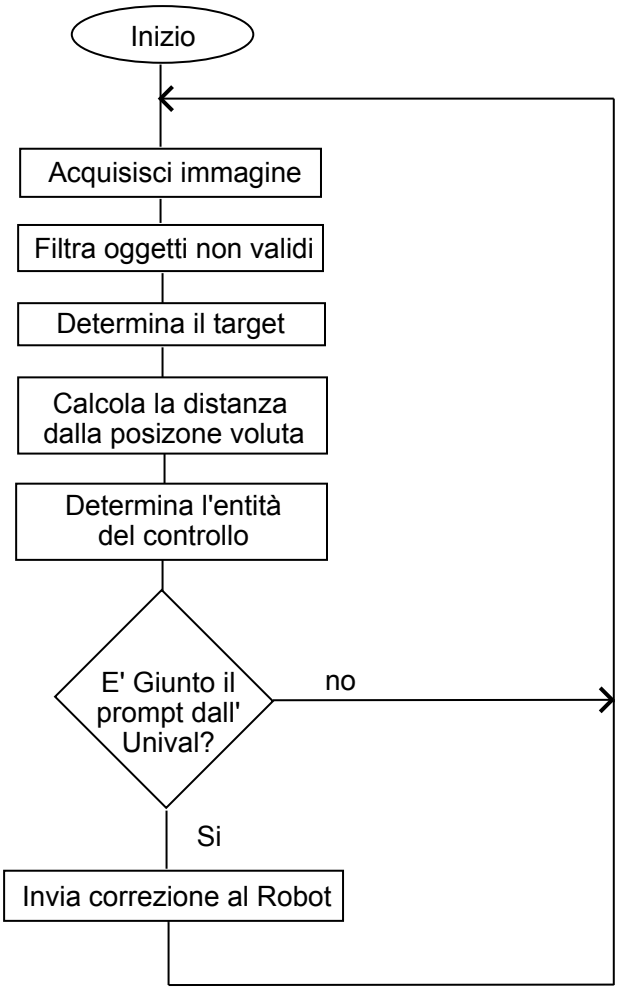
Si sottolinea che in questa applicazione, avendo una visione soggettiva (dal robot) dell'ambiente in cui si muove il braccio, si è deciso di utilizzare il sistema di riferimento TOOL del Puma 260. Questo approccio permette, una volta definita una qualsiasi *locazione* dell'end-effector, di poter intercettare un target che si muove su un qualsiasi piano di lavoro parallelo al piano visivo del robot.

[6] Per il codice vedere appendice F.1.1.

[7] Questo procedimento è stato implementato nella procedura *calibrazione()* presente in appendice F.1.1.

5.2.2 Obiettivi dell'applicazione

Una volta calibrata la telecamera, il sistema seleziona un *target* tra tutti gli oggetti inquadrati[8]. Ad ogni nuovo fotogramma viene aggiornata la posizione assunta dal target ed il sistema calcola una correzione da impartire al robot per porre l'end-effector sulla verticale dell'oggetto agganciato[9]. Di seguito si propone il diagramma di flusso del codice C++ del file `EYEHAND.CPP` che rappresenta il *Codice di controllo* di figura 5.1.



A differenza della applicazione ad anello aperto questo sistema opera in re-

[8] Per decidere all'inizializzazione del sistema quale sia il target si è utilizzato in questa tesi un *"imprinting"*, ovvero il sistema segue il primo oggetto che rileva, nulla vieta però di utilizzare una strategia diversa.

[9] Questo comportamento è necessario per poter implementare un eventuale cattura del target.

al time calcolando la posizione del target ad ogni immagine che giunge dalla telecamera (30 fps) e quindi in linea di principio è in grado di *intercettare* sia oggetti fermi che in movimento (notare che quest'ultima operazione non è realizzabile dall'applicazione pick and place come anticipato nel paragrafo 1.2). La dinamica e le prestazioni dell'inseguimento dipendono dalle strategie di controllo implementate. Una trattazione di tali strategie è presentata nei capitoli 6 e 7.

5.3 Ulteriori commenti

I metodi di *"stima della posizione"* del (dei) target[10] in entrambe le applicazioni sono del tutto generalizzati e permettono quindi di operare su un piano di lavoro che è posizionato arbitrariamente nello spazio come indicato in figura 5.5

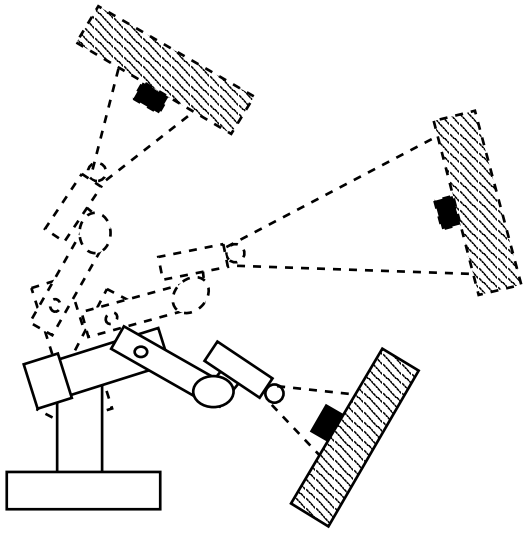


Figura 5.5: Esempi di diverse condizioni operative.

Affinchè ciò avvenga correttamente è necessario che le procedure di calibrazione siano eseguite meticolosamente, una particolare attenzione bisogna porre nella calibrazione del sistema ad anello aperto infatti la conoscenza delle posizioni assolute dei riferimenti in coordinate WORD indica che deve essere stata effettuata una loro precedente misura. Questa operazione di misura viene realizzata attraverso un posizionamento manuale del robot (utilizzando il *teach pendant*, vedi il paragrafo 2.4) nella posizione (x, y, z) dei riferimenti.

[10]Per la stima della posizione vedi il paragrafo 3.4.

Per il sistema ad anello chiuso invece bisogna solo assicurarsi che durante i movimenti generati per avere più visioni del medesimo riferimento[11], quest'ultimo non esca mai dallo campo visivo del robot.

Si sottolinea infine come attraverso la relazione 3.51 anche l'applicazione ad anello chiuso, una volta realizzata la calibrazione, possa ottenere la distanza della telecamera dal target e quindi ricavare la posizione relativa (x, y, z) di quest'ultimo rispetto al sistema di riferimento Tool dell'end-effector.

[11] Vedere il paragrafo 3.4.2.