UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI TECNICA E GESTIONE DEI SISTEMI INDUSTRIALI
CORSO DI LAUREA MAGISTRALE IN INGEGNERIA MECCATRONICA

TESI DI LAUREA MAGISTRALE

STUDIO ED IMPLEMENTAZIONE DI UNA CELLA ROBOTIZZATA PER BIN PICKING COLLABORATIVO

Relatore: Chiar.mo Prof. Giovanni Boschetti

Laureando: Teresa Sinico
2004643

ANNO ACCADEMICO: 2021-22

SOMMARIO

L'obbiettivo di questa tesi è quello di studiare e successivamente implementare una cella per bin picking collaborativo, composta da un sistema di visione ed un robot antropomorfo. In particolare, in un primo momento è stato necessario procedere all'installazione ed alla calibrazione ottimale di un sistema di visione 3D basato su luce strutturata. In un secondo momento, è stato scritto un programma che eseguisse correttamente il bin picking, ponendo particolare attenzione alla corretta presa degli oggetti, alla pianificazione della traiettoria priva di ostacoli ed alla minimizzazione del tempo ciclo. Successivamente sono stati sviluppati una serie di algoritmi per il riconoscimento degli operatori all'interno dell'area di lavoro del robot, basati sulla visione 2D. Questi algoritmi, facenti parte di un programma multitasking, consentono di rilevare la presenza di un operatore, di monitorarne la posizione e quindi di far reagire il robot di conseguenza. Grazie ai test sperimentali effettuati, si è potuta verificare la bontà della soluzione proposta.

I libri non servono per sapere ma per pensare, e pensare significa sottrarsi all'adesione acritica per aprirsi alla domanda, significa interrogare le cose al di là del loro significato abituale reso stabile dalla pigrizia dell'abitudine; è evitare che i testi diventino testi sacri per coscienze beate che, rinunciando al rischio dell'interrogazione, confondono la sincerità dell'adesione con la profondità del sonno.

Umberto Galimberti

RINGRAZIAMENTI

Mi è doveroso dedicare questo spazio del mio elaborato alle persone che hanno contribuito, con il loro instancabile supporto, alla realizzazione dello stesso. In primo luogo, un ringraziamento speciale al mio relatore, il professor Giovanni Boschetti per la sua costante disponibilità ed i suoi preziosi consigli durante la scrittura di questa tesi. Grazie alla mia famiglia, per avermi trasmesso l'importanza della dedizione al proprio lavoro, della tenacia e della perseveranza. Grazie a mio fratello Giacomo, che da sempre è il faro che illumina la mia rotta. Grazie a Filippo, che dalla terza elementare, vicini o lontani, è sempre stato dalla mia parte. Grazie ai miei compagni di corso Luca e Dario, grazie a Paolo, Alberto, Elena ed a tutti coloro che hanno incrociato la loro vita con la mia lasciandomi qualcosa di buono.

INDICE

ELENCO DELLE FIGURE

ELENCO DELLE TABELLE

1 INTRODUZIONE

1.1 IL BIN PICKING

Dall'introduzione dei robot nella produzione industriale c'è stato un desiderio sempre crescente di automatizzare completamente il maggior numero possibile di processi produttivi. Uno dei task che è stato oggetto di numerose ricerche scientifiche negli ultimi decenni [1] [2] è quello di riuscire ad afferrare, tramite un manipolatore industriale, oggetti disposti in maniera casuale e potenzialmente incastrati tra di loro all'interno di un contenitore: questo task in letteratura prende il nome di bin picking. In generale, soprattutto nei sistemi di alimentazione dei componenti, è necessario eseguire la singolarizzazione, l'orientamento ed a volte anche il controllo qualità dei pezzi prima che essi procedano verso le lavorazioni successive. A tale scopo, nei processi produttivi che ormai sono diventati obsoleti, venivano utilizzati i vibroalimentatori orbitali (che prendono spesso il nome di tazze rotanti): questi sistemi sono tuttavia molto costosi e assolutamente non flessibili. È infatti necessario prevedere una tazza rotante per ogni singolo pezzo del quale è necessario eseguire la singolarizzazione. Applicazioni più recenti prevedono che gli oggetti contenuti in un recipiente e provenienti da una lavorazione a monte vengano poi svuotati su lunghi nastri trasportatori al fine di separare tra loro i vari pezzi e renderne possibile la successiva identificazione e presa. Questi sistemi sono ormai sistemi collaudati, ma richiedono grandi spazi e tempi per la singolarizzazione dei pezzi e la loro successiva presa. Proprio per questo motivo, negli ultimi anni si è intensificata la ricerca nel campo del bin picking al fine di riuscire a prelevare le parti direttamente dal contenitore nel quale si trovano disposti alla rinfusa, per mezzo di un robot industriale ed un opportuno sistema di visione. Tramite efficienti sistemi di bin picking è quindi possibile contenere i tempi ed evitare l'impiego di grandi spazi per la disposizione degli oggetti, nonché aumentare la produttività del processo industriale ed eliminare il collo di bottiglia costituito dal sistema necessario per la singolarizzazione dei pezzi. La completa automatizzazione di questo tipo di task è ambiziosa in quanto implica la risoluzione di una serie di problematiche:

- Devono essere stimati correttamente posizione ed orientamento nello spazio delle parti da afferrare. Per fare ciò è necessario l'impiego di opportuni sistemi di visione, più sofisticati rispetto ai classici sistemi di visione 2D oppure ai sensori (tipicamente

encoder) che si utilizzano per afferrare gli oggetti al volo da un nastro trasportatore in movimento;

- Servono degli algoritmi sofisticati per il riconoscimento degli oggetti. Infatti, nei sistemi tradizionali che impiegano i nastri trasportatori, i pezzi sono separati tra di loro e pertanto il loro riconoscimento è relativamente semplice. Nel bin picking, invece, gli oggetti sono sovrapposti tra di loro e sono quindi necessari degli algoritmi avanzati per il riconoscimento dei singoli pezzi;

- È necessario che il sistema di visione sia in grado di riconoscere se gli oggetti sono incastrati tra di loro all'interno del contenitore ed è necessario prevedere delle opportune strategie nel caso questo accada (ad esempio richiedere l'intervento di un operatore o muovere i pezzi in modo da riuscire a separarli);

- È necessaria una pianificazione del moto che eviti le collisioni tra l'organo terminale del robot ed altri pezzi, nonché le collisioni tra l'organo terminale del robot e le pareti del contenitore;

- In generale, è necessario prevedere degli algoritmi che siano in grado di determinare se un oggetto è oppure non è afferrabile: un oggetto può essere non afferrabile perché si trova incastrato con altri oggetti oppure perché la sua presa comporterebbe la collisione del robot con altri oggetti o con le pareti del contenitore;

- Particolare attenzione va posta alla scelta non solo del sistema di visione, ma anche dell'organo terminale: gli oggetti si trovano infatti all'interno del contenitore disposti alla rinfusa, spesso non presentano spazi tra di loro che consentano il passaggio di tradizionali griffe, così come può non essere banale riuscire a prendere un solo pezzo senza afferrare anche pezzi adiacenti. Inoltre, le collisioni che possono verificarsi dipendono fortemente dalla geometria dell'organo terminale;

- Va notato infine che tipicamente il bin picking viene eseguito su pezzi tutti uguali tra di loro. Il task può però complicarsi ulteriormente se i pezzi da prelevare all'interno del contenitore sono diversi tra di loro: questo richiede un'ancora più attenta scelta dell'organo terminale e può richiedere l'impiego di organi terminali multi utensile.

Per tutta questa serie di aspetti, il prelievo automatico di oggetti alla rinfusa è un problema solo parzialmente risolto e molte delle soluzioni proposte non sono abbastanza robuste per applicazioni industriali.

Figura 1.1: Esempio di una cella robotizzata per il bin picking (courtesy Yaskawa)

1.1.1 *Esempio di tipica cella per bin picking*

A titolo di esempio, in figura 1.1 è riportata una possibile configurazione di una cella robotizzata per il bin picking. In figura si notano in particolare un robot antropomorfo ed un sistema di visione 3D. Per eseguire il bin picking si utilizzano infatti manipolatori antropomorfi a 6 gradi di libertà, poiché è necessaria la completa flessibilità in termini di movimenti ed i manipolatori a 4 gdl (quali i manipolatori Scara) sono insufficienti per portare a termine questo tipo di task. In figura 1.1 si nota altresì la presenza di barriere di sicurezza: queste sono necessarie in quanto nell'esempio considerato il robot non è collaborativo e non è dunque adatto ad essere utilizzato a contatto con gli operatori.

1.1.2 *Flowchart del processo di bin picking*

In generale, un processo di bin picking segue il flowchart in figura 1.2. Per prima cosa viene acquisita la mappa 3D tramite il sistema di visione: si parla di mappa 3D in quanto, come illustrato di seguito, nel bin picking è necessario conoscere anche la quota z e l'orientazione nello spazio degli oggetti da prelevare. Successivamente, tramite opportuni algoritmi, vengono riconosciuti gli oggetti e viene selezionato l'oggetto da prendere attraverso una serie di criteri (tipicamente, quello con la quota z maggiore e quello che è effettivamente afferrabile, ovvero non si trova incastrato con altri oggetti e la cui presa non prevede collisioni). Si procede poi con la stima della posizione e dell'orientamento, con la pianificazione della traiettoria priva di collisioni (sia con il contenitore che con altri oggetti) e con il pick and place dell'oggetto. La frequenza con la quale viene acquisita la mappa 3D dipende dalla specifica applicazione: molti sistemi prevedono

Figura 1.2: Flowchart del processo di bin picking

che ogni volta che viene prelevato un pezzo poi venga eseguita nuovamente la scansione 3D. Una scelta di questo tipo è motivata dal fatto che prelevando un oggetto è possibile che se ne spostino di altri, quindi è possibile che altri pezzi precedentemente identificati nel frattempo cambino di posizione. Altri sistemi prevedono invece che venga eseguita una scansione 3D, vengano rilevati posizione ed orientamento di tutti i pezzi afferrabili e poi si proceda al pick and place di ciascuno di essi, senza eseguire ulteriori scansioni 3D nel mezzo. Solo quando tutti i pezzi afferrabili sono stati prelevati, allora si esegue un altra scansione 3D, in quanto nel frattempo saranno diventati afferrabili nuovi pezzi che prima non lo erano. Questa scelta è motivata dal fatto che le scansioni 3D richiedono un tempo non trascurabile, tipicamente nell'ordine di qualche secondo, e possono costituire un elemento critico per ridurre il tempo ciclo, soprattutto se si utilizzano manipolatori performanti che vengono fatti lavorare a velocità sostenute. Proprio per il considerevole tempo richiesto da una scansione 3D, è necessario che essa venga acquisita mentre il robot è in movimento ma si trova al di fuori dell'area inquadrata dal sistema di visione (ad esempio, mentre viene eseguito il place dell'oggetto appena preso).

1.2 I SISTEMI DI VISIONE NEL BIN PICKING

Il sistema di visione è un elemento chiave nel successo degli algoritmi di bin picking. Per eseguire l'attività di bin picking non sono sufficienti i tradizionali sistemi di visione 2D: essi forniscono le informazioni di posizione relative al piano, quindi la posizione in (x, y) e l'orientazione dell'oggetto sotto forma di un angolo θ, ma non la sua altezza rispetto al piano e nemmeno la sua orientazione nello spazio. Essi sono sufficienti per prelevare oggetti disposti ordinatamente su piano o su un nastro trasportatore, del quale si conosce a priori la quota z. Inoltre, per eseguire la presa al volo su un nastro trasportatore spesso non è nemmeno necessario un sistema di visione, ma è sufficiente l'impiego di un encoder. Per il bin picking è invece necessario l'utilizzo di sistemi di visione 3D, tipicamente composti da un proiettore ed una o più telecamere, che sono in grado di fornire le sei coordinate nello spazio tridimensionale, ovvero la posizione (x, y, z) e l'orientamento (w, p, r). Nelle celle robotizzate impiegate per attività di bin picking il sistema di visione può essere fisso, quindi montato su una propria struttura, oppure può essere montato a bordo robot, ovvero su pinza o testa dedicata. Questa seconda opzione consente di eseguire scansioni da diversi punti di vista. In questa tematica particolarmente attuale, i principali brand che si occupano di robotica industriale e di sistemi di visione stanno lanciando nuovi prodotti per rendere l'operazione di riconoscimento degli oggetti sempre più efficiente in termini di tempo e di accuratezza. Tra i sistemi di visione 3D, la luce strutturata codificata è considerata una delle tecniche più affidabili per ricostruire la superficie degli oggetti. Una descrizione più approfondita del funzionamento dei sistemi di visione a luce strutturata è riportata in sezione 2.5.1. In generale, questa tecnica si basa sul proiettare una serie di pattern di luce ed acquisire delle immagini della scena da uno o più punti di vista, che rilevano quindi la deformazione che il pattern di luce subisce quando incontra l'oggetto lungo il suo percorso. In base a tale deformazione, tramite triangolazione geometrica, è possibile risalire alle coordinate degli oggetti nello spazio tridimensionale. Infatti, calibrando il sistema di visione, il modello 3D degli oggetti è ricostruito triangolando la sorgente di luce, la telecamera ed il punto della superficie dell'oggetto colpito dal pattern di luce. I sensori di visione che utilizzano questa tecnologia hanno il pregio di essere velocissimi nel processo di scansione in quanto vengono acquisite contemporaneamente tutte le parti dell'oggetto o degli oggetti che sono colpiti dal pattern di luce. In particolare, i sensori a luce strutturata sono estremamente precisi ed offrono elevata risoluzione. La sorgente di luce è in grado di proiettare griglie di luce strutturata ben definite, come linee parallele, mix di punti in sequenza ordinata o casuale, pixel in sequenze codificate di riquadri e linee. I sensori di visione 3D richiedono tuttavia una calibrazione

accurata ed onerosa, poiché dalla loro corretta calibrazione dipende l'accuratezza in z dei risultati ottenuti. Oltre alla calibrazione, i sensori di visione 3D che sono montati su una struttura fissa e non a bordo robot richiedono anche uno studio del loro posizionamento.

1.3 ORGANI TERMINALI PER BIN PICKING

Come precedentemente accennato, la scelta dell'organo terminale è fondamentale nelle operazioni di bin picking. Un'analisi approfondita del design ottimale degli organi terminali nelle attività di bin picking è riportata in [3]. È bene sottolineare che la scelta dell'organo terminale migliore dipende non solo dalla geometria degli oggetti da afferrare ma anche dallo stato del contenitore, ovvero da quanto pieno esso è. Inizialmente, il contenitore è pieno ed i vari pezzi si trovano molto vicini tra di loro, senza fessure tra di essi che permettano il passaggio delle griffe di una pinza tradizionale. Con il contenitore pieno può essere difficile utilizzare delle pinze tradizionali che realizzano un contatto bilaterale e può essere opportuno utilizzare organi terminali di tipo pneumatico (*vacuum* o *suction grippers*) come suggerito da [4]. A mano a mano che il contenitore si svuota, gli oggetti che rimangono sono distanti tra di loro, rimangono i più difficili da prendere (nel caso ci siano diversi oggetti) e gli oggetti si trovano vicino alle pareti del contenitore. In letteratura si trovano numerose opzioni, tra le quali anche l'utilizzo di due gripper come in [5]: questa scelta aumenta tuttavia il numero di collisioni possibili, quindi richiede uno studio ancora più attento delle traiettorie.

1.4 UTILIZZO DI ROBOT COLLABORATIVI NEL BIN PICKING

La presa di oggetti idealmente disposti alla rinfusa e potenzialmente incastrati tra di loro all'interno di un contenitore tramite un manipolatore industriale è un task che in molti casi non ha ancora raggiunto un *success rate* del 100%, soprattutto nel caso di oggetti dalla geometria complessa. Può infatti capitare che il sistema di visione non sia in grado di identificare se un pezzo è afferrabile oppure no, così come ci sono degli oggetti che sono particolarmente difficili da riconoscere, quali ad esempio oggetti con superfici riflettenti o oggetti trasparenti. Può inoltre succedere che che vi siano oggetti incastrati tra di loro che il robot non riesce a separare in autonomia. In questi casi è tipicamente necessario interrompere l'esecuzione del programma e far intervenire un operatore che ad esempio rimuova l'oggetto che il sistema di visione non riesce a riconoscere oppure che separi oggetti che sono incastrati tra di loro. Poiché l'utilizzo di manipolatori industriali tradizionali richiede che gli stessi siano all'interno di barriere di sicurezza, queste operazioni richiedono il fermo del robot, l'entrata dell'operatore all'interno della cella robotizzata, la risoluzio-

ne della problematica ed il successivo riavvio del robot. Per questo motivo, nell'eseguire il task di bin picking, diventa particolarmente interessante l'utilizzo di robot collaborativi (che prendono spesso il nome di cobot): i robot collaborativi sono infatti particolari robot dotati di sensori che fermano automaticamente il robot se rilevano un contatto con l'operatore e questo ne permette l'utilizzo senza l'impiego di barriere di protezione. Proprio per questo motivo, l'utilizzo di un robot collaborativo per svolgere l'attività di bin picking presenta il vantaggio di rendere le operazioni che richiedono il fermo del robot e l'intervento dell'operatore più veloci, in quanto non è necessario superare barriere di protezione. Inoltre, l'attività di bin picking non richiede generalmente eccessive velocità o elevato payload, il che rientra nel campo di utilizzo dei robot collaborativi. Grazie all'utilizzo di un robot collaborativo, è possibile far in modo che se il robot non è in grado di riconoscere o prelevare un oggetto, questo si fermi ed aspetti che l'operatore provveda alla sua rimozione o alla sua ricollocazione e ad esempio alla pressione di un pulsante che consenta al robot di ripartire: il tutto può avvenire a distanze molto ravvicinate, senza il superamento di barriere nel mezzo. Un esempio di utilizzo di un robot collaborativo in una cella robotizzata per bin picking è riportato in [6].

1.5 IL BIN PICKING COLLABORATIVO

Uno spunto di ricerca interessante, oggetto di questo lavoro di tesi, è quello di investigare la possibilità di rendere collaborativo il task di bin picking. Per far collaborare il robot e l'operatore è innanzitutto necessario riconoscere la presenza dell'operatore all'interno della zona di lavoro del robot e monitorarne la posizione: in genere, questo può essere fatto attraverso l'utilizzo di una telecamera esterna. L'idea innovativa sviluppata durante il lavoro di tesi è quella di utilizzare il sistema di visione 3D di cui è tipicamente composta una cella robotizzata per bin picking non solo per la stima della posizione e dell'orientamento degli oggetti, ma anche per il riconoscimento degli operatori all'interno della zona di lavoro del robot inquadrata dal sistema di visione. A tal proposito, è necessario sottolineare che:

- Se si utilizza il medesimo sistema di visione sia per il riconoscimento dei pezzi dei quali poi eseguire il bin picking sia per il riconoscimento degli operatori, è necessario che il programma con cui viene programmato il robot sia un programma multitasking, quindi che si occupi contemporaneamente sia del pick and place degli oggetti rilevati tramite le mappe 3D, sia del riconoscimento degli operatori. Per questo motivo, è necessario sincronizzare opportunamente i due processi di visione. Se invece si utilizzasse un sistema di visione esterno, a seconda dell'appli-

cazione, potrebbe anche non essere necessario un programma multitasking;

- L'algoritmo che si occupa del riconoscimento degli operatori deve essere robusto ed il riconoscimento deve avvenire in tempi compatibili con la sicurezza degli operatori (e con le velocità di un robot collaborativo, al di sotto dei 250 mm/s);

- Anche se si utilizza un sistema di visione 3D, il riconoscimento degli operatori deve basarsi sulla visione 2D. Il motivo di ciò è che le scansioni 3D richiedono un tempo considerevole (nell'ordine di qualche secondo) per essere acquisite, nonché poi per essere processate. Non è quindi pensabile continuare ad acquisire mappe 3D per il rilevamento degli operatori. Se il sistema di visione 3D è un sistema di visione a luce strutturata, risulta invece fattibile utilizzare una delle due telecamere e quindi sfruttare la visione 2D. Va comunque evidenziato che mentre si esegue una scansione 3D non sarà possibile rilevare anche la presenza degli operatori: a seconda delle applicazioni e dei rischi andrà valutato se questo aspetto è accettabile oppure no.

1.5.1 *Possibili applicazioni di bin picking collaborativo*

Essere in grado di riconoscere la presenza degli operatori all'interno dell'area di lavoro ed essere in grado di monitorarne la posizione rende possibili una serie di applicazioni innovative, in cui robot ed operatore possono interagire in diversi modi. Tra quelli analizzati e presentati di seguito, se ne riportano alcuni:

- L'operatore può entrare nella zona di lavoro del robot per sistemare oggetti incastrati tra di loro o non afferrabili, oppure per rimuovere un oggetto: in questi casi, è possibile far rallentare il robot quando si accorge della presenza dell'operatore all'interno del suo spazio di lavoro per rendere dunque queste operazioni più sicure. Alternativamente, se l'area di lavoro è divisa in due o più zone, è possibile far rallentare il robot solo qualora robot e operatore si trovino a lavorare nella medesima zona;

- Come precedentemente illustrato, i robot collaborativi sono dotati di sensori che permettono il fermo del robot nel caso in cui avvenga un contatto con l'operatore e viene garantito che tale contatto non sia dannoso per le persone. Tuttavia, è comunque desiderabile prevedere tale contatto e fare in modo che lo stesso non avvenga: monitorando continuamente la posizione dell'operatore e del robot stesso, calcolando la distanza tra i due, è possibile far fermare il robot oppure fargli cambiare traiettoria nel caso in cui stia per avvenire una collisione. In questo modo, le collisioni vengono evitate a monte;

- Si può prevedere che l'operatore entri nell'area di lavoro del robot per depositare ulteriori oggetti da prelevare tramite bin picking: in questo caso il robot, una volta riconosciuto l'operatore, dovrà spostarsi dalla zona di lavoro, aspettare che l'operatore esca anch'esso dalla zona di lavoro, eseguire nuovamente una scansione 3D e riprendere la propria attività;

- Molto spesso nelle applicazioni in cui robot ed operatori si trovano a collaborare, il sincronismo tra le attività di uno e dell'altro viene ottenuto grazie alla pressione di pulsanti. Ad esempio, se il robot deve assemblare un oggetto insieme all'operatore, può essere che depositi il pezzo e poi attenda che l'operatore ci svolga un'operazione. Quando l'operatore ha eseguito il suo task, preme un pulsante ed il robot proseguirà con l'operazione successiva. Tramite il riconoscimento dell'operatore per mezzo del sistema di visione, è possibile che la collaborazione tra operatore e manipolatore avvenga senza la pressione di tasti: una volta che l'operatore entra nell'area di lavoro il robot sa che l'operatore ha iniziato ad eseguire il suo task ed attende fino a che l'operatore non esce dalla zona di lavoro.

2 DESCRIZIONE DELLA CELLA ROBOTIZZATA

2.1 INTRODUZIONE

La cella robotizzata presa in considerazione durante la stesura di questa tesi è costituita da un robot antropomorfo e da un sensore di visione 3D: essa è raffigurata in figura 2.1. In particolare, la cella è costituita da un robot antropomorfo collaborativo Fanuc CR-15iA e dal relativo controllore Fanuc R-30iB Plus. Sulla flangia del robot sono montati in serie il sensore di forza FS-15iA e la pinza elettrica collaborativa Schunk Co-act EGP-C. Il sensore di visione 3D utilizzato è il sensore Fanuc 3D Area Sensor, montato in maniera permanente su un'apposita struttura di supporto.

2.2 ROBOT ANTROPOMORFO FANUC CR-15IA

Il robot preso in esame è un robot collaborativo a 6 gradi di libertà, le cui dimensioni ed il cui spazio di lavoro sono riportati in figura 2.2. L'impiego di un robot antropomorfo è necessario per le operazioni di bin picking, in quanto è necessaria la massima flessibilità nelle possibilità di manipolazione degli oggetti. Com'è noto, i robot collaborativi sono robot pensati per interagire con gli operatori e non necessitano di barriere di sicurezza, poiché dotati di particolari sensori che garantiscono che le eventuali collisioni con gli operatori non si traducano in danni agli stessi. Nel caso del robot Fanuc CR-15iA, i sensori che garantiscono le funzionalità collaborative sono posizionati alla sua base. Va sottolineato tuttavia che durante la stesura di questa tesi tutte le funzionalità collaborative non erano utilizzabili, in quanto il robot doveva ancora essere fissato a terra e quindi installato in una postazione in maniera permanente. Tutti i risultati ottenuti sono comunque estendibili al caso in cui il robot sia effettivamente collaborativo, quindi fissato in maniera permanente. Tra le prestazioni del robot Fanuc CR-15iA, meritano una particolare menzione il carico sostenibile al polso (payload) di 15 kg e la velocità massima (dell'organo terminale) di 800 mm/s (si può arrivare a 1500 mm/s con l'aggiunta di un sensore ulteriore). Le prestazioni di questo robot sono superiori alla maggior parte dei robot collaborativi in commercio, che invece presentano dimensioni, payload e velocità massime piuttosto contenute.

Figura 2.1: Cella robotizzata implementata

2.2.1 *Input ed output del robot*

Al robot possono essere collegati una serie di input ed output, sia analogici che digitali. Tra gli input ed output digitali, alcuni di essi sono propri del robot ed in ambiente Fanuc sono identificati dalle sigle RI (robot input) ed RO (robot output): vale la pena nominarli in quanto compaiono nei programmi illustrati di seguito. In particolare, sul secondo braccio del robot sono posizionati due led: uno bianco (RO[1]) ed uno rosso (RO[2]). Alla base del robot è posto un pulsante retroilluminato che costituisce un input come pulsante (RI[1], diventa ON se si tiene premuto, diventa OFF appena viene rilasciato) ed un output come led (RO[3]). Questi led possono essere sfruttati per dare delle informazioni sullo stato di funzionamento del robot. I robot input ed output menzionati sono riportati in figura 2.3.

2.3 SENSORE DI FORZA FS-15IA

In figura 2.4 è riportato un ingrandimento della flangia del robot, sulla quale si vedono essere installati in serie il sensore di forza e la pinza elettrica collaborativa. Diversamente da quanto riportato in figura, il robot preso in esame presenta anche una flangia tra senso-

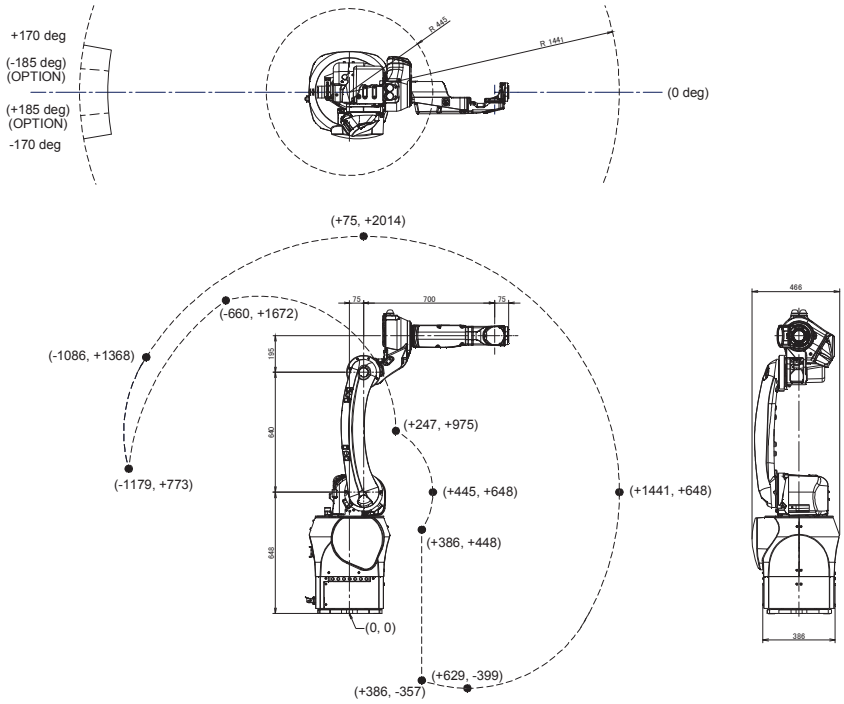


Figura 2.2: Workspace del robot CR-15iA (courtesy Fanuc)

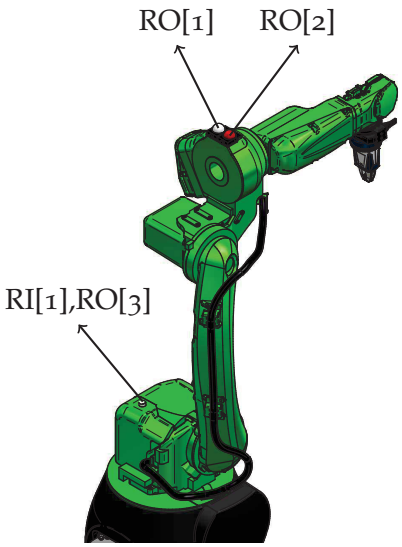


Figura 2.3: Robot input ed output nel robot CR-15iA

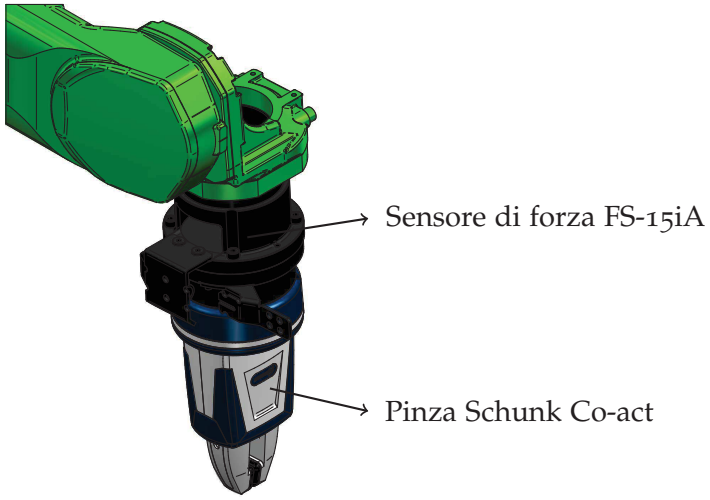


Figura 2.4: Sensore di forza e pinza collegati alla flangia del robot

re di forza e pinza, necessaria per il passaggio dei cavi della pinza. La flangia è stata stampata in materiale polimerico mediante stampa additiva. Il sensore di forza FS-15iA garantisce un controllo di forza ad esempio per applicazioni di assemblaggio, pesatura, sbavatura e lucidatura. Questo sensore è stato opportunamente calibrato, ma il suo funzionamento non è stato testato. Ciò potrebbe essere pertanto oggetto di studi futuri.

2.4 PINZA SCHUNK CO-ACT EGP-C

La pinza che è stata installata sul robot è la pinza Schunk Co-act EGP-C, una pinza elettrica a due griffe parallele, certificata per il funzionamento collaborativo. Questa pinza è pensata per la presa e la movimentazione di componenti di piccole e medie dimensioni: l'apertura delle griffe varia infatti da un minimo di 25 mm ad un massimo di 40 mm. La forza di presa è regolabile a scatti: essa può essere impostata al 25%, 50%, 75% o 100% della forza di presa massima, pari a 140 N. La forza di presa è stata impostata in un primo momento al 50%: questo valore risulta un ottimo compromesso per oggetti rigidi anche diversi tra di loro. Una forza di serraggio maggiore potrebbe essere necessaria per oggetti che presentano un elevata deformabilità. Le griffe installate sulla pinza sono le griffe vendute insieme alla pinza: per la loro forma, esse non sono in realtà adatte alla presa di oggetti molto vicini tra di loro e quindi non risultano griffe ideali per le operazioni di bin picking. Infatti, esse richiedono un certo spazio tra gli oggetti per l'inserimento delle griffe e quindi la presa degli oggetti. Lavori futuri potrebbero esplorare l'utilizzo di griffe auto prodotte mediante stampa additiva che siano maggiormente indicate per la presa di oggetti per bin picking, anche in relazione alla

Tabella 2.1: Colore del led della pinza in funzione degli input digitali

Colore led	RO[7]	RO[8]
Led pinza spento	0	0
Led pinza verde	0	1
Led pinza giallo	1	0
Led pinza rosso	1	1

geometria specifica degli oggetti da afferrare. Vale comunque la pena notare che tra i molti lavori presenti in letteratura riguardo al bin picking, la maggior parte di essi fa utilizzo di organi terminali di tipo pneumatico.

2.4.1 *Input pinza*

Particolare attenzione va posta a come sono collegati input ed output digitali della pinza collaborativa. In figura 2.3 sono stati illustrati gli input e gli output propri del robot CR-15iA. Una volta che la pinza viene collegata, a questi si aggiungono anche gli input e gli output della pinza stessa. In particolare, gli input digitali della pinza DI1 e DI2, sono stati collegati ai robot output RO[7] ed RO[8]. Tramite RO[7] ed RO[8] è possibile modificare il colore del led posizionato sulla pinza, secondo la logica illustrata in tabella 2.1. Inoltre, altri due input della pinza sono stati collegati agli output del robot RO[5] ed RO[6]: essi servono rispettivamente per aprire e chiudere le griffe della pinza. Per ulteriori dettagli su come gestire questi robot output per l'apertura e chiusura delle griffe per le operazioni di pick and place si rimanda in appendice A.9.

2.4.2 *Outut pinza*

La pinza è dotata di due sensori induttivi che permettono di capire se le griffe sono chiuse oppure aperte. Ai robot input RI[2] e RI[3] sono stati collegati i due output della pinza corrispondenti ai due sensori induttivi. Se la pinza è aperta, è alto il robot input RI[2] (RI[2]=ON). Se invece la pinza è chiusa fino alla posizione di finecorsa, è alto il robot input RI[3] (RI[3]=ON). Va sottolineato che la posizione di finecorsa della pinza può essere modificata manualmente, ma questa possibilità non è stata esplorata. Pertanto, quando la pinza afferra degli oggetti di dimensioni maggiori di 25 mm, l'input RI[3] risulta OFF, anche se c'è un oggetto in presa.

2.5 SENSORE DI VISIONE 3D AREA SENSOR

Per eseguire correttamente le operazioni di bin picking, non sono sufficienti delle telecamere tradizionali: esse infatti forniscono le informazioni di posizione solamente relative al piano, quindi le coordinate (x, y) di un punto ed un certo angolo θ che ne rappresenta l'orientazione. Per eseguire il bin picking, è necessario impiegare dei sensori di visione 3D, che siano in grado di fornire le informazioni di posizione (x, y, z) ed orientamento (w, p, r) nello spazio tridimensionale: il sensore di visione è dunque il componente fondamentale delle applicazioni di bin picking. Il sensore di visione 3D utilizzato nella cella robotizzata di figura 2.1 è il sensore Fanuc 3DA (3D Area Sensor): un sensore di visione a luce strutturata costituito da un proiettore e due telecamere con risoluzione 1024×1280 pixel. La particolarità di questo sensore è che ha un campo visivo elevato, pari a $1300 \times 1000 \times 1000$ mm: queste sono le dimensioni standard di un pallet, il che significa che il sensore 3DA è adatto ad eseguire il bin picking da un contenitore che abbia la grandezza di un pallet oppure è adatto a operazioni di depallettizzazione. Esistono in commercio numerosi sensori di visione 3D, alcuni di essi molto più compatti rispetto al sensore 3DA. Vale la pena sottolineare che le due telecamere facenti parte del sensore di visione Fanuc 3DA sono due telecamere in bianco e nero. In [7] è tuttavia riportato che attualmente, la maggior parte di sistemi per bin picking, adotta sistemi RGB-D (*Red Green Blue - Depth*), ovvero sensori 3D costituite da due telecamere a colori. L'utilizzo di telecamere a colori fornisce delle informazioni aggiuntive utili per il riconoscimento degli oggetti. Un esempio delle potenzialità dell'utilizzo di un sensore di questo tipo è riportato in [8].

2.5.1 *Principio di funzionamento dei sensori a luce strutturata*

L'obbiettivo di un sensore 3D è quello di riuscire a determinare la profondità, ovvero la quota z, di ciascun punto acquisito tramite la fotocamera. Per comprendere il principio di funzionamento dei sensori a luce strutturata è necessario prima capire come si può ricavare la quota z di un punto tramite un singolo raggio di luce e poi comprendere come tale concetto può essere esteso per poi ricavare la quota z di ognuno dei punti catturati tramite la fotocamera. Ulteriori e più approfonditi dettagli sulla matematica alla base dei sensori di visione 3D, nonché una guida su come poter costruire un sensore di visione 3D a partire da componenti standard che si trovano in commercio si trova in [9]. In generale, un sensore di visione a luce strutturata si basa sulla proiezione sugli oggetti di pattern di luce da parte di un proiettore e sulla successiva rilevazione della distorsione di tali pattern una volta che essi colpiscono gli oggetti da parte di una o più tele-

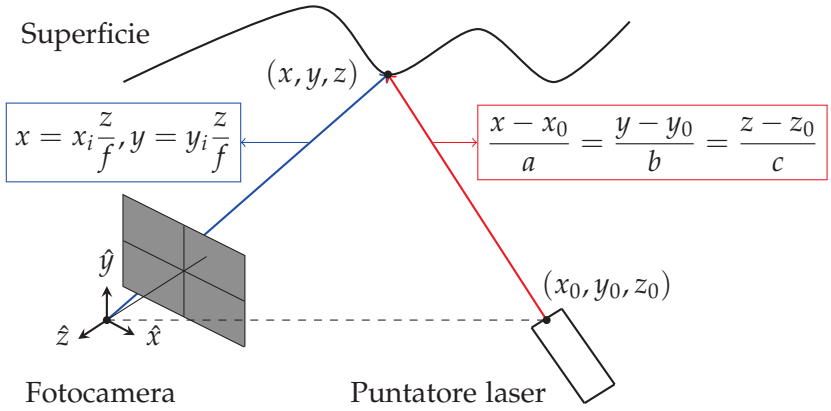


Figura 2.5: Point based range finding

camere. L'analisi della distorsione dei pattern, tramite triangolazione geometrica, consente di risalire alla quota z degli oggetti.

2.5.1.1 *Point based range finding*

Il modo più semplice per ottenere la profondità z di un certo punto acquisito da una fotocamera prende il nome di *point based range finding*. Il principio di funzionamento di questo metodo è illustrato in figura 2.5: si impiegano una fotocamera ed una sorgente di luce (quale ad esempio un puntatore laser). Il puntatore laser produce un singolo raggio di luce, che colpisce la scena in 3D e quindi la superficie degli oggetti. Si assume che la posizione della fotocamera e del puntatore laser siano note rispetto ad un certo sistema di riferimento, in figura indicato dalla terna $(\hat{x}, \hat{y}, \hat{z})$ e posizionato sull'obbiettivo della fotocamera. Queste informazioni di posizione vengono acquisite durante il processo di calibrazione dei sensori. Il singolo raggio di luce prodotto dal puntatore laser colpisce la superficie degli oggetti, e produce un punto luminoso che viene catturato dalla fotocamera in posizione (x_i, y_i). Conoscendo queste coordinate è possibile determinare l'equazione della retta che prende il nome di *camera ray* (in figura indicata con il colore blu), che parte dall'origine del sistema di riferimento e passa per il punto di coordinate (x_i, y_i). La retta rossa prende invece il nome di *light ray* ed è nota una volta posizionato il puntatore laser. Di conseguenza, le coordinate (x, y, z) si trovano come intersezione tra la retta blu e quella rossa. Nella pratica, per applicare questo metodo ciò che si fa è catturare un'immagine della scena senza il puntatore laser, catturare una seconda immagine della scena con il puntatore laser e fare una sottrazione tra le due per evidenziare il punto luminoso e quindi determinare le coordinate (x_i, y_i), da cui si ricavano poi le coordinate del punto fisico. Questa tecnica prende il nome di *background subtraction*. Questo semplice metodo permette di determinare la profondità z di un singolo punto catturato: risulta evidente che per costruire una mappa 3D completa servirebbe un numero troppo

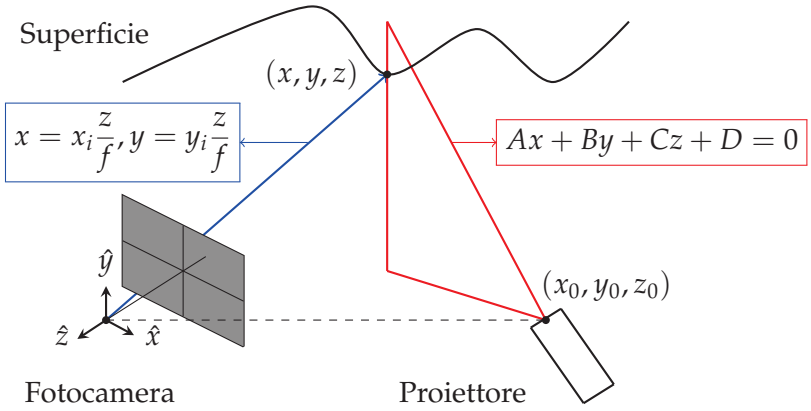


Figura 2.6: Light stripe based range finding

elevato di immagini nonché un tempo di rilevazione esagerato. Ad esempio, se l'immagine è 640×480 pixel, bisognerebbe acquisire un immagine per ogni pixel, quindi oltre 300000 immagini.

2.5.1.2 *Light stripe based range finding*

L'estensione del metodo precedente consiste nell'utilizzare un proiettore che emetta un fascio di luce (visto dal punto di vista del proiettore, questo fascio di luce è di fatto una riga) e non più un singolo raggio di luce. Questo metodo prende il nome di *light stripe based range finding* ed il suo principio di funzionamento è riportato in figura 2.6. Il fascio di luce interseca la superficie degli oggetti formando una curva (le riga di luce, una volta che colpisce gli oggetti, viene distorta). Per ogni punto sulla curva, si vuole determinarne la quota z. Si consideri un singolo punto luminoso (x_i, y_i) rilevato dalla fotocamera, tra quelli sulla curva prodotta dalla deformazione del fascio di luce. Anche in questo caso è possibile determinare la quota z di quel punto in quanto è nota sia l'equazione blu (*camera ray*) che l'equazione del fascio di luce (matematicamente è l'equazione di un piano, prende il nome di *light plane*). Il punto fisico corrispondente al punto (x_i, y_i) che si vede dalla telecamera deve essere all'intersezione tra la retta *camera ray* ed il piano *light plane*. La quota z si trova quindi come:

$$z = \frac{-Df}{Ax_i + By_i + Cf} \tag{2.1}$$

Una volta ottenuta la quota z, si ottengono i valori x e y dall'equazione della retta blu. Utilizzando questo metodo è necessario prendere un immagine per colonna: se l'immagine è 640×480 pixel, è necessario quindi acquisire 640 immagini. Questo è sicuramente un numero di immagini inferiore al precedente, ma non è comunque implementabile nella realtà.

Tabella 2.2: Luce strutturata codificata tramite codice binario

Numero di riga	1	2	3	4	5	6	7
Codifica in binario	001	010	011	100	101	110	111
Bit 1	0	0	0	1	1	1	1
Bit 2	0	1	1	0	0	1	1
Bit 3	1	0	1	0	1	0	1

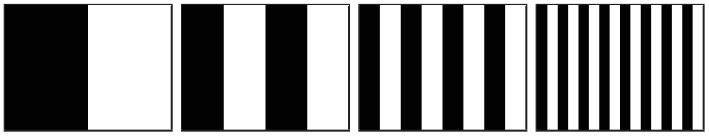
2.5.1.3 *Coded structured light*

Il passo successivo è quindi non proiettare una sola riga di luce alla volta, bensì proiettare più righe contemporaneamente, in modo da determinare la quota z di più punti contemporaneamente. Questo passaggio non è tuttavia banale, in quanto a seconda della geometria dell'oggetto e della scena di cui si vuole ricostruire la profondità, non sempre si è in grado di determinare in maniera univoca quale tra le righe proiettate abbia prodotto un punto luminoso in una certa posizione dell'immagine acquisita. Non sapendo quale riga ha prodotto il punto luminoso, non si è a conoscenza dell'equazione del piano *light plane* e quindi non si può risalire alla coordinate (x, y, x) del punto fisico. In generale, se si proiettano un certo numero di righe, il punto luminoso (x_i, y_i) potrebbe essere stato prodotto da una qualsiasi di esse: questo porta al concetto di luce strutturata codificata. L'idea di base è quella di avere a disposizione un certo numero di righe da proiettare e di acquisire un certo numero di immagini, in ognuna delle quali una ben specifica combinazione delle righe viene proiettata. Un certo punto (x_i, y_i) risulterà illuminato solo in alcune immagini: sapendo in quale immagini il punto è illuminato ed in quali non lo è, e sapendo quale combinazione delle righe è stata proiettata per ciascuna immagine, è possibile risalire in maniera univoca alla riga di luce che ha prodotto il punto luminoso, quindi poi alle coordinate del punto fisico corrispondente. Si supponga di utilizzare 7 righe luminose (chiaramente per una rilevazione efficiente è necessario un numero maggiore di righe), numerate da 1 a 7. Com'è noto, per esprimere i numeri da 1 a 7 in codifica binaria servono 3 bit. Ciò che si fa è prendere 3 immagini, con tre combinazioni di righe di righe accese (quindi proiettate) diverse. Secondo la codifica indicata in tabella 2.2, la prima immagine va acquisita con le prime tre righe luminose spente e le seconde quattro accese, mentre la seconda e la terza vanno acquisite secondo quanto indicato alle ultime due righe della tabella. In questo modo, utilizzando 7 righe luminose e acquisendo 3 immagini con 3 combinazioni di righe accese o spente diverse, è possibile determinare in maniera univoca da quale riga ciascun punto luminoso (x_i, y_i) sia stato prodotto e quindi avendo l'equazione del *light plane* determinare anche la quota z. Si supponga appunto di aver acquisito tre

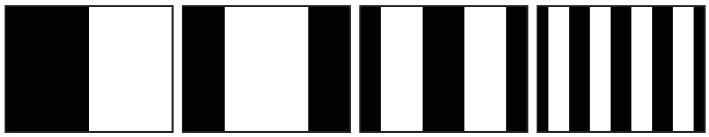
Tabella 2.3: Luce strutturata codificata tramite codice Gray

Numero di riga	1	2	3	4	5	6	7
Codifica in binario	001	011	010	110	111	101	100
Bit 1	0	0	0	1	1	1	1
Bit 2	0	1	1	1	1	0	0
Bit 3	1	1	0	0	1	1	0

immagini secondo la logica appena illustrata: si osserva lo stato di un punto di medesime coordinate (x_i, y_i) nelle tre immagini. A titolo di esempio, se tale punto appare luminoso nella prima immagine, non illuminato nella seconda ed illuminato nella terza questo corrisponde ad un codice in binario 101, quindi significa che è stato prodotto dalla quinta riga luminosa. Conoscendo il suo *light plane* è possibile dunque risalire alle coordinate (x, y, z) del punto fisico corrispondente. In generale, tramite questo metodo, si utilizzano $2^n - 1$ righe luminose e si acquisiscono n immagini (dove il -1 deriva dal fatto che la codifica con tutti zeri corrisponde a tutte le righe luminose spente, quindi ad un'assenza di informazioni). Acquisendo 8 immagini, si utilizzano 255 righe luminose, che è già un numero ragionevole per ottenere una risoluzione discreta. Le immagini in cui alcune righe luminose sono on ed altre off prendono il nome di pattern. Vale la pena sottolineare che il concetto che sta alla base di questo metodo è quello di essere in grado di determinare se ogni pixel acquisito è in stato di on (illuminato dal proiettore) oppure off (non illuminato dal proiettore): nella realtà può esserci una transizione tra i due stati, quindi possono esserci degli errori nel determinare lo stato di on oppure di off di un pixel. Utilizzando la codifica binaria in tabella 2.2 si nota che le transizioni sono in totale 10: per ridurre questo fenomeno, che prende il nome di *light bleeding*, si può utilizzare una codifica Gray, riportata in tabella 2.3. La codifica Gray prevede che si passi da un intero al successivo modificando un solo bit: l'utilizzo di questa codifica riduce a 6 il numero di transizioni di stato, quindi riduce ma non elimina il problema. In figura 2.7 sono riportati degli esempi di pattern a seconda che si utilizzi la codifica binaria oppure la codifica Gray. Quando il sensore 3D è in funzionamento, ciò che si vede è appunto il proiettore che proietta queste immagini velocemente in sequenza. In figura 2.8 è invece riportata la sequenza di luce strutturata con codifica binaria o con codifica Gray nel caso in cui si acquisiscano 8 immagini, quindi si utilizzino 255 righe proiettate. Esistono poi altre codifiche, che si estendono a più livelli e non solo ai due livelli della codifica binaria (livello on oppure off, stato 1 o 0). A titolo di esempio, si possono utilizzare sistemi con base 3, quindi con tre livelli: tra queste possibilità si distingue la codifica (R,G,B). Se si proiettano dei colori,

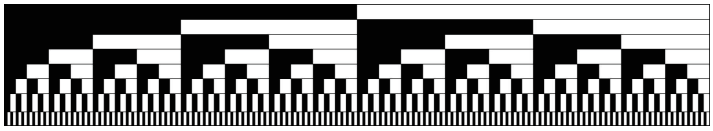


(a) Sequenze di luce strutturata con codifica binaria

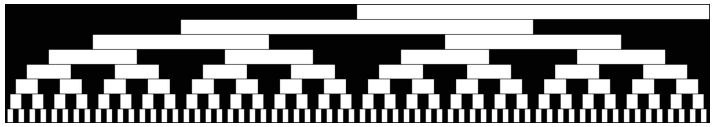


(b) Sequenze di luce strutturata con codifica Gray

Figura 2.7: Esempio di pattern di luce strutturata



(a) Sequenze di luce strutturata con codifica binaria



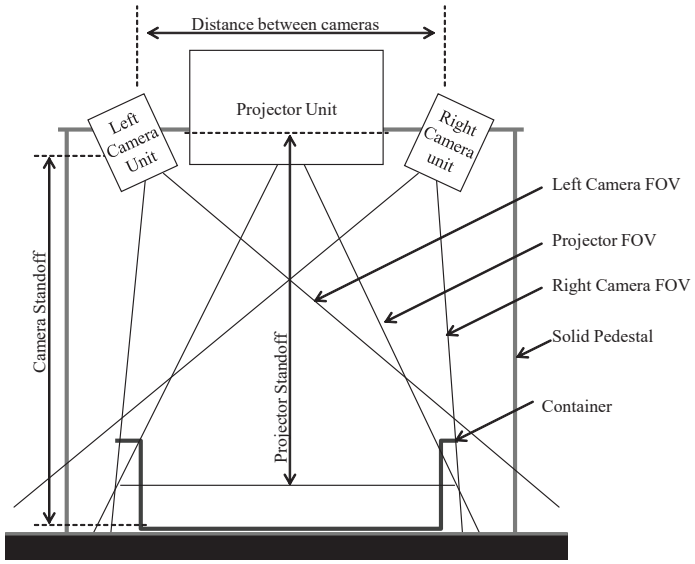
(b) Sequenze di luce strutturata con codifica Gray

Figura 2.8: Confronto tra le sequenze di luce strutturata con codifica binaria e strutturata

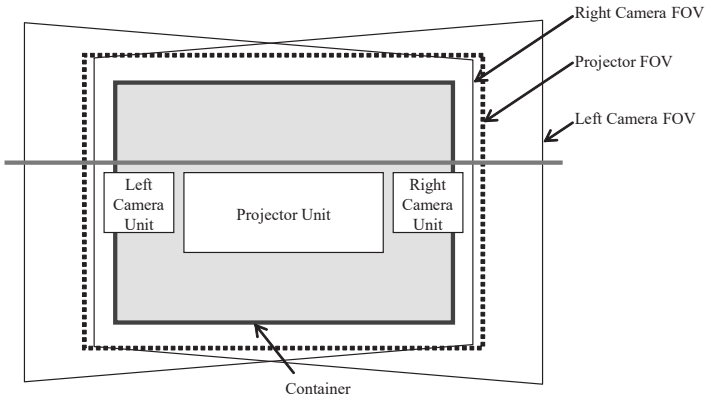
bisogna porre particolare attenzione al fatto che gli oggetti, in base al loro colore, riflettono alcune lunghezze d'onda più di altre, mentre alcune non le riflettono proprio: è il caso dei colori blu e rosso, che si trovano agli opposti dello spettro, il blu non riflette il rosso e viceversa. L'ideale è che la scena sia grigia, in quanto il grigio è in grado di riflettere tutti i colori. Il pattern di luce proiettato dai sensori a luce strutturata è solitamente composto da una serie di strisce, ma può anche essere composto da una matrice di punti o da altre forme. In [10] sono riportati una serie di pattern e di strategie di codifica che possono essere utilizzati nei sensori a luce strutturata.

2.5.2 *Layout del sensore 3DA*

Il sensore di visione 3DA è montato su una struttura fissa, secondo il layout illustrato in figura 2.9. Essendo montato su una struttura fissa, il sensore di visione vede sempre la stessa immagine inquadrata. Un vantaggio di installare il sensore di visione su un'apposita struttura è che il tempo ciclo può essere ridotto in quanto l'immagine può essere acquisita e processata mentre il robot è in movimento e sta



(a) Vista frontale



(b) Vista dall'alto

Figura 2.9: Layout standard del sensore 3DA (courtesy Fanuc)

portando a termine un altro task. Tra le soluzioni utilizzate per il bin picking, molte prevedono invece che il sensore di visione sia montato direttamente sul polso del robot (prende il nome di installazione a bordo robot). In questo caso, il robot calcola la posizione dell'oggetto individuato tenendo in considerazione la posizione della telecamera stessa. Il vantaggio di questa soluzione è che possono essere eseguite diverse acquisizioni in più posizioni e quindi può essere aumentata l'area scansionabile. A titolo di esempio, il sensore 3DV della famiglia Fanuc, è più compatto rispetto al 3DA (in quanto il sensore 3DV in un unico case contiene sia il proiettore che le due telecamere) e viene spesso montato direttamente a bordo robot, su pinza o su testa dedicata.

2.5.3 *Installazione del sistema di visione*

Nell'installare il sistema di visione nell'apposita struttura, le due grandezze alle quali porre maggiore attenzione sono l'altezza alla quale viene posizionato il sistema e la distanza alla quale devono essere posizionate le due telecamere. In tabella 2.4 è riportata l'altezza a cui posizionare il sistema di visione, così come la distanza tra le telecamere in funzione dell'area di misura desiderata, ovvero delle dimensioni del contenitore dal quale prelevare i pezzi per eseguire il bin picking. Il sensore di visione è stato in un primo momento posizionato in modo da prelevare i pezzi presenti sopra ad un tavolo. Se lavori futuri richiedessero la presa da cassone posto direttamente sul pavimento potrebbe essere necessario riposizionare le due telecamere così come modificare l'altezza alla quale esse ed il proiettore sono posizionate. Una volta posizionato proiettore e telecamere all'altezza H e le due telecamere ad una distanza d, è necessario posizionare sistema di visione ed il contenitore dal quale prelevare i pezzi coerentemente tra d loro. Per prima cosa va acceso il proiettore: per accendere il proiettore è necessario andare sulla schermata di modifica del 3D Area Sensor Data. Su *Test Projector Pattern* è necessario selezionare *Frame* sul menù a tendina ed attivare il proiettore cliccando su PRJ ON. Viene quindi proiettato il pattern in figura 2.10. Alternativamente, si può anche proiettare il pattern a righe, ma questo non è utile in fase di regolazione del sistema di visione perché non consente di vedere il centro del pattern. La posizione del contenitore e del proiettore va regolata fino a che il pattern non viene proiettato su tutto il contenitore ed il centro del pattern corrisponde con il centro del contenitore. Va poi sottolineato che risoluzione della mappa 3D e campo visivo del sistema di visione sono strettamente correlati tra di loro. Infatti, il sensore 3DA calcola un certo numero di punti 3D all'interno del campo visivo del proiettore, ad esempio 239×192 punti se la densità è impostata su normale. Di conseguenza, una volta impostata la densità della mappa 3D, la risoluzione dei punti 3D dipende dal cam-

Tabella 2.4: Altezza del sistema di visione e distanza tra le telecamere in funzione dell'area di misura desiderata (dimensioni contenitore)

Area di misura			Distanza tra	Altezza
Lato lungo (mm)	Lato corto (mm)	Altezza (mm)	le telecamere (mm)	telecamere (mm)
1340	1000	1000	1340	2438
1200	900	896	1200	2200
1100	825	821	1100	2030
1000	750	746	1000	1860
900	675	672	900	1691
800	600	597	800	1521
700	525	522	700	1351
600	450	448	600	1181
500	375	373	500	1011

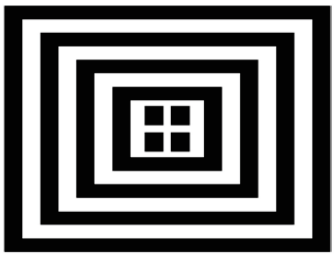


Figura 2.10: Pattern di tipo *frame* proiettato dal proiettore (courtesy Fanuc)

po visivo del proiettore. Più grande il campo visivo del proiettore, maggiore la distanza spaziale tra i punti 3D calcolati. In generale, il campo visivo del proiettore dovrebbe essere il più piccolo possibile per ottenere quindi una risoluzione maggiore. Inoltre, le telecamere devono essere nella stessa linea del proiettore (viste dall'alto) ma non devono trovarsi al di fuori del contenitore altrimenti le pareti del contenitore causeranno dei punti dove c'è assenza di dati. Ciò significa che, in linea d'aria e viste dall'alto, le due telecamere non devono trovarsi fuori dagli estremi del contenitore. Anche l'inclinazione delle telecamere va regolata: tale regolazione va eseguita in modo che tutto il pattern sia visibile da ciascuna delle due telecamere, il centro del pattern deve coincidere con il centro dell'immagine acquisita da ciascuna delle due telecamere. È infine possibile regolare la messa a fuoco sia del proiettore che delle due telecamere, in

2.5.3.1 *Accuratezza in z*

L'accuratezza in z che si ottiene dipende anche dal layout del sistema di visione. Come si nota in figura 2.9b, il proiettore e le telecamere devono essere posizionati secondo una linea retta. Una volta fissata l'altezza alla quale telecamere e proiettore sono posizionati, si può regolare la distanza tra le due telecamere. In particolare, se si aumenta la distanza tra le due telecamere, aumenta l'accuratezza in z. L'accuratezza teorica in z può essere stimata dalla seguente formula:

$$\Delta z = \pm \frac{L \times H}{n \times d} \tag{2.2}$$

dove L è la lunghezza del lato lungo del campo visivo della telecamera, H è l'altezza alla quale sono poste telecamere e proiettore, n è il numero di pixel del lato lungo dell'immagine e d è la distanza tra le due telecamere. Sostituendo nella (2.2) i valori corrispondenti al sistema preso in esame si ottiene:

$$\Delta z = \pm \frac{1450 \times 1550}{1280 \times 900} = 1{,}95\,\text{mm} \tag{2.3}$$

Va sottolineato che quello appena calcolato è un valore teorico. Il valore reale dipende dal pattern impostato nel proiettore, dalla messa a fuoco, dalla luce ambientale, dall'accuratezza della calibrazione, ecc. È anche possibile che proiettore e telecamere non si trovino alla stessa quota z, mentre le due telecamere devono obbligatoriamente trovarsi alla stessa altezza rispetto al pavimento.

2.5.4 *Calibrazione ottimale del sistema di visione*

L'informazione di posizione rilevata dalla telecamera deve essere convertita dal sistema di coordinate della telecamera al sistema di coordinate in utilizzo (tipicamente *user frame* o *tool frame*). Per permettere tale conversione, è necessario avere delle informazioni che descrivano dove si trova la telecamera rispetto a dove si trova il robot. Quest'operazione prende appunto il nome di calibrazione. In particolare, viene calcolato in automatico il rapporto di ingrandimento, che è la relazione che sussiste tra un pixel del sensore e i millimetri del campo inquadrato. Nei sistemi di visione Fanuc esistono due tipi di calibrazione: *grid pattern calibration* ed *robot generated grid calibration*. Nel primo caso, la calibrazione viene eseguita utilizzando una griglia di calibrazione dove si trova un pattern di punti predefinito. Nel secondo caso, durante la calibrazione, sulla flangia del robot viene montato un target con una certa geometria e poi il robot si muove seguendo delle traiettorie predefinite al fine di acquisire immagini del target da varie angolazioni e posizioni. Questo secondo metodo per eseguire la calibrazione, illustrato in figura 2.11 è il metodo raccomandato dalla

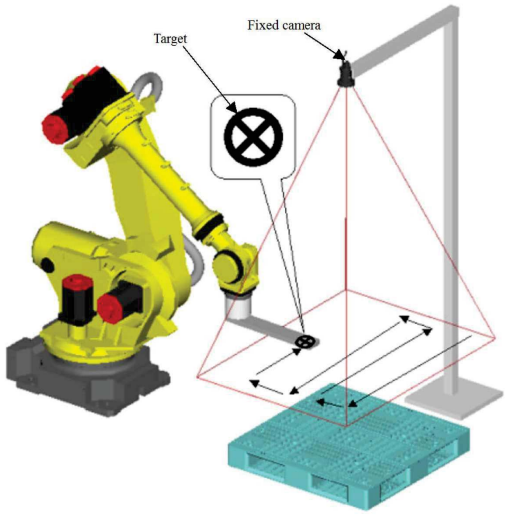


Figura 2.11: Esempio di layout di calibrazione eseguita con metodo *robot generated grid calibration* (courtesy Fanuc)

manualistica Fanuc, in quanto permette di ottenere risultati più accurati. Per la calibrazione deve essere quindi montato sul robot un target, che viene poi riconosciuto dal sistema di visione. Il riconoscimento del target viene impostato tramite il tool *GPM Locator Tool*, che consente di riconoscere una certa forma. Bisogna assicurarsi che il target abbia una dimensione, una volta catturato come immagine, tra 80 e 100 pixel sia in direzione verticale che orizzontale. Ad esempio, se il campo visivo della telecamera è di 900 mm (lenti 8 mm, distanza tra il target e le telecamere 2000 mm), il diametro del target deve essere compreso tra 120 e 160 mm. È inoltre fondamentale che durante la calibrazione è sia impostato l'utensile corretto, ovvero che il robot conosca la distanza rispetto alla flangia alla quale il target è posto. Se durante la calibrazione rimane impostato come utensile corrente l'utensile che è stato rimosso per eseguire la calibrazione, tutta la calibrazione risulterà scorretta e pertanto da rifare.

2.5.5 *Parametri da impostare per l'acquisizione di una mappa 3D*

Una volta terminata la calibrazione, compare una schermata dove è possibile impostare una serie di parametri al fine di ottenere un mappa 3D che sia il più soddisfacente possibile. I tre parametri più importanti da impostare sono:

- *Exposure time:* questo valore va aumentato se le immagini acquisite sono troppo scure, va invece diminuito se le immagini acquisite sono troppo chiare. In entrambi i casi, è importante che il pattern a righe proiettato dal proiettore sia ben visibile e non ci siano zone tutte bianche o tutte nere.

- *Intensity:* l'intensità del proiettore va regolata solo dopo aver regolato il tempo di esposizione. È importante che il valore dell'intensità sia il più alto possibile: un valore elevato di intensità riduce l'influenza della luce ambientale. Se ci sono punti 3D che non possono essere acquisiti per la troppa luminosità, l'intensità va diminuita. Al contrario, se ci sono punti 3D che non possono essere acquisiti per la scarsità di luce, l'intensità va aumentata.
- *3D map density:* questo valore è impostabile scegliendo tra tre livelli: *coarse*, *normal* e *fine*. Maggiore questo valore, più definita e precisa sarà la mappa 3D e maggiore sarà anche il tempo di acquisizione della stessa. Sui manuali Fanuc è consigliato di lasciare questo valore su *normal* e le prove eseguite con questo valore di densità della mappa 3D hanno sempre dato risultati soddisfacenti.

È comunque opportuno sottolineare che la luce ambientale può influire sulla robustezza dei dati rilevati dal sensore 3DA. Più la luce ambientale è forte, meno stabili saranno i dati rilevati dal sensore di visione.

2.5.6 *Mappe 3D*

Una volta installato, calibrato e regolato opportunamente il sistema di visione, è possibile acquisire delle mappe 3D. Un esempio di mappa 3D acquisita mediante il sensore Fanuc 3D Area Sensor è riportato in figura 2.12: in questo caso è la mappa 3D di un contenitore all'interno del quale si trovano degli oggetti cubici sovrapposti tra di loro. Come si nota dalla figura, le mappe 3D vengono visualizzati a colori. In particolare:

- Il colore rosso indica i punti che si trovano più vicini al sensore di visione, quindi quelli con una quota z più elevata;
- Il colore blu indica i punti che si trovano più lontani dal sensore di visione, quindi quelli con una quota z inferiore. Deve essere blu il fondo del contenitore;
- Il colore nero indica invece i punti nei quali c'è un'assenza di dati, quali ad esempio i bordi vicino ad un contenitore se il contenitore ha le pareti molto alte. Assenza di dati può esserci anche tra gli oggetti stessi se i valori di intensità e tempo di esposizione non vengono impostati correttamente.

2.5.7 *Processi di visione*

Una volta che il sensore è stato configurato, è necessario predisporre almeno un processo di visione. Si possono impostare processi di visione 2D, che quindi sfruttano una delle due telecamere presenti sul

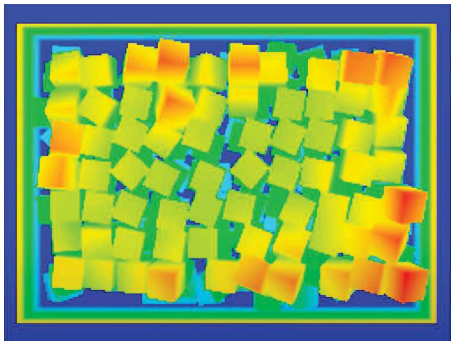


Figura 2.12: Esempio di mappa 3D acquisita mediante il sensore Fanuc 3D Area Sensor

sistema di visione, oppure processi di visione 3D, che sfruttano il sistema di visione 3DA, quindi telecamere e proiettore. Ogni processo di visione sia esso 2D o 3D contiene degli strumenti, che in ambiente Fanuc prendono il nome di *tool*, che concorrono alla generazione dell'immagine o della mappa dei punti 3D. I vari tool vengono impostati secondo un diagramma ad albero, ovvero ci sono alcuni strumenti che possono essere figli di altri, ovvero lavorare partendo dai risultati dello strumento padre. Oltre a generare l'immagine o la mappa, questi strumenti consentono anche di personalizzare e ottimizzare i tempi di esecuzione e la qualità del rilevamento. In generale, tramite PC viene eseguita la calibrazione e vengono impostati i processi di visione tramite interfaccia grafica iRVision. Nei processi di visione si possono inserire diversi tool per il corretto riconoscimento degli oggetti da afferrare. Una volta che si va poi a scrivere il programma per la movimentazione del robot, tramite teaching pendant o tramite iRProgrammer, è necessario richiamare il processo di visione tramite delle opportune istruzioni, che sono sempre precedute dal prefisso VISION (le più utilizzate sono VISION RUN_FIND, per avviare il processo di visione e VISION GET_OFFSET per ottenere le coordinate del pezzo da prelevare). I dati acquisiti dal processo di visione (quali coordinate dell'oggetto da prelevare, il numero di oggetti riconosciuti dalla scansione 3D, la larghezza o la lunghezza degli oggetti, ecc) vengono messi in opportuni registri, ai quali il programma può poi accedere. Una rappresentazione grafica del funzionamento reciproco tra sistema di visione, robot, controllore e PC è riportata in figura 2.13.

2.5.7.1 *Tool più utilizzati*

Di seguito sono riportati gli strumenti più utilizzati nei processi di visione Fanuc, sia 2D che 3D. Tali strumenti saranno utilizzati nei capitoli successivi sia per il riconoscimento degli operatori sia per il riconoscimento degli oggetti dei quali eseguire poi il bin picking.

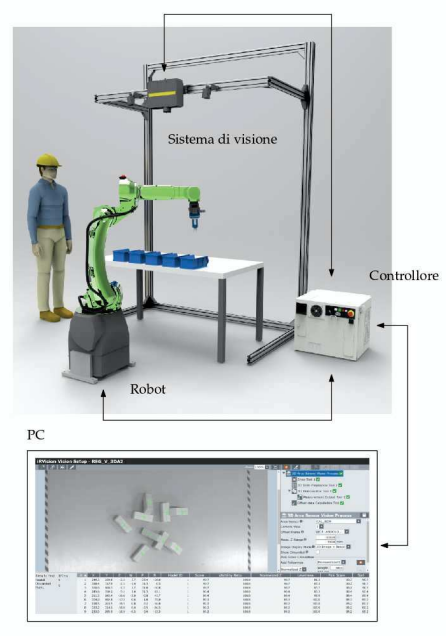


Figura 2.13: Rappresentazione grafica del funzionamento reciproco tra sistema di visione, robot, controllore e PC

- *Snap Tool:* questo strumento deve sempre essere presente in ogni processo di visione, sia esso 2D o 3D. Esso si occupa di catturare l'immagine della scena e possono essere impostati una serie di parametri, quali ad esempio l'area della quale acquisire l'immagine, il tempo di esposizione e la riduzione della risoluzione. Ridurre l'area della quale acquisire l'immagine o ridurre la risoluzione può essere particolarmente strategico in quanto avere un minor numero di dati da processare risulta in un minor tempo necessario per il processo di visione, quindi un minor tempo di visione, che è sempre una caratteristica desiderabile.

- *GPM Locator Tool:* questo strumento si occupa di riconoscere un certo pattern geometrico all'interno dell'immagine acquisita. In questo caso, deve essere fornita un'immagine input che contenga il pattern da riconoscere e successivamente il tool confronta le immagini acquisite con quella di input. Possono poi essere impostati una serie di parametri che stabiliscono quanto il pattern riconosciuto può differire dall'immagine input.

- *3D Peak Locator Tool:* questo strumento permette di trovare i massimi locali della mappa di punti 3D. In particolare, è possibile selezionare una certa area all'interno della quale ricercare i massimi locali. Di fatto, questo strumento restituisce i punti più alti degli oggetti riconosciuti come più in alto (ovvero quelli che saranno i primi ad essere prelevati).

- *3D Blob Locator Tool:* questo strumento si occupa di rilevare dei

punti o delle regioni che differiscono dalle adiacenti per proprietà come luminosità o colore comparati con l'ambiente. In questo modo si riescono a rilevare delle regioni dove ci sono alcune proprietà che rimangono costanti. Per rilevare oggetti che hanno superficie liscia o piana, questo strumento risulta particolarmente utile. Si possono anche in questo caso impostare una serie di parametri, quali il numero minimo o massimo di punti (che quindi determina la grandezza minima e massima dell'area da riconoscere).

- *CSM Locator Tool:* la sigla CSM sta per *Curved Surface Locator* e questo strumento permette di rilevare delle superfici curve, quali dei cilindri. Questo strumento, a differenza di quello al punto seguente, si basa solo sulla visione 2D.
- *3D Cylinder Locator Tool:* questo strumento prima riconosce un blob e successivamente riconosce una forma cilindrica. Questo strumento è più potente rispetto al precedente ed è sempre consigliabile utilizzarlo.
- *3D Gripper Finger Locator Tool:* questo strumento fornisce in output le posizioni nelle quali l'oggetto è afferrabile dalla pinza. In questo tool vanno specificate le dimensioni della pinza.
- *Line Locator Tool:* questo strumento si occupa di trovare un segmento rettilineo, di una lunghezza predefinita, all'interno dell'immagine acquisita. Questo strumento risulta molto utile nel riconoscimento di particolari oggetti, quali ad esempio una chiave esagonale.
- *3D Obstruction Measurement Tool:* questo strumento è utile per capire quanto sono sovrapposti tra di loro due o più oggetti. In particolare, se il sistema di visione riconosce due oggetti, in cui il primo è parzialmente sovrapposto al secondo, questo tool fornisce il numero di punti 3D che si trovano sovrapposti al pezzo che sta sotto.
- *Multi-Locator Tool:* questo strumento è composto di uno o più GPM Locato Tool figli e si occupa di riconoscere, nello stesso processo di visione, più di un oggetto. In particolare, viene acquisita l'immagine ed in tale immagine vengono ricercati i pattern precedentemente impostati come immagini input.
- *Measurement Output Tool:* questo strumento consente di memorizzare alcuni dei valori di output forniti da uno dei tool elencati precedentemente. Ad esempio, consente di memorizzare la larghezza o la lunghezza di un oggetto che viene riconosciuto, così come il numero di punti blob o il numero di punti sovrapposti al pezzo che sta sotto. In generale, una volta che viene richiamato il processo di visione tramite il comando RUN_FIND e

poi tramite il comando GET_OFFSET, viene anche indicato un registro di visione VR[] nel quale inserire i dati acquisiti. Questo registro contiene non solo una certa posizione (ovvero la posizione del primo pezzo da prelevare), ma anche una serie di campi VR[].MES[] all'interno dei quali, tramite lo strumento in questione, si possono inserire alcune delle misure acquisite dal sistema di visione.

- *Conditional Execution Tool:* questo strumento consente di eseguire delle scelte in base al verificarsi o meno di certe condizioni. Ad esempio, si può confrontare una delle misure (lunghezza dell'oggetto riconosciuto) con un valore predefinito e decidere che in caso si verifichi una condizione di maggiore o uguale, il risultato venga invalidato, ovvero l'oggetto venga rimosso dalla lista degli oggetti da prelevare.

2.6 STRUTTURA DI SUPPORTO PER IL SENSORE DI VISIONE

La struttura di supporto per il sensore di visione 3D è stata costruita mediante profilati in alluminio. Le quote della struttura realizzata sono riportate in figura 2.14. Grazie alla struttura di profilati, l'altezza alle quali si trova il proiettore e le due telecamere è regolabile in funzione dell'altezza alla quale devono essere prelevati i pezzi. È inoltre facilmente regolabile anche la distanza tra due telecamere.

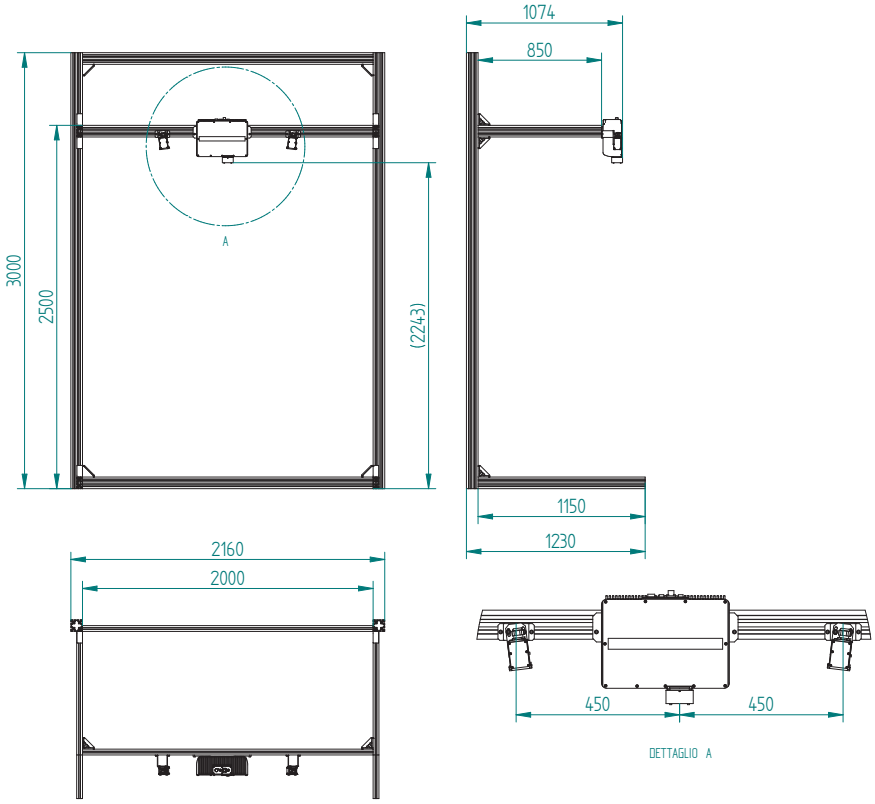


Figura 2.14: Struttura di supporto per il sensore di visione

3

ALGORITMO PER BIN PICKING

3.1 INTRODUZIONE

Una volta provveduto all'installazione della cella robotizzata e alla calibrazione ottimale del sistema di visione, si è proceduto con la stesura di un programma che eseguisse correttamente il bin picking, quindi permettesse di prelevare degli oggetti disposti in maniera casuale all'interno di un contenitore. Gli oggetti da prelevare utilizzati in un primo momento sono dei parallelepipedi di dimensioni nominali $30 \times 40 \times 80$ mm, ottenuti mediante stampa additiva. Come già accennato al capitolo 1, una geometria semplice dei pezzi da prelevare semplifica notevolmente il problema del bin picking, nonché permette l'utilizzo di un organo terminale quale la pinza Schunk a disposizione (purché gli oggetti siano sufficientemente distanti tra di loro, ovvero purché tra di essi vi sia lo spazio sufficiente per l'inserimento delle griffe).

3.2 IMPOSTAZIONE DEL PROCESSO DI VISIONE

3.2.1 *3D Blob Locator Tool*

I processi di visione in ambiente Fanuc, che prendono il nome di *3D Area Sensor Vision Process*, possono comporsi di diversi strumenti (tool) per il corretto riconoscimento degli oggetti. Oltre ai tool di default che compaiono quando si crea un processo di visione, va infatti impostato un tool specifico per il riconoscimento di una certa geo-

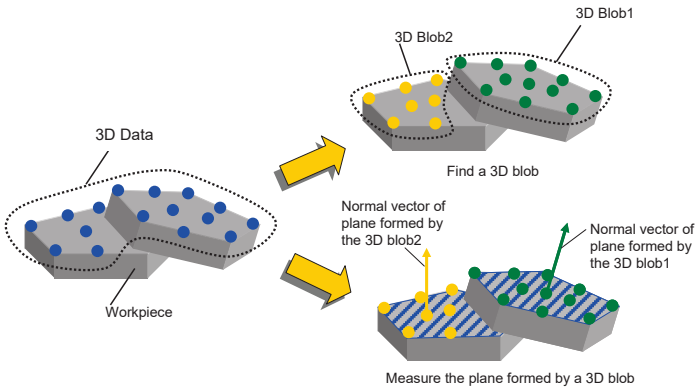


Figura 3.1: Principio di funzionamento del *3D Blob Locator Tool* (courtesy Fanuc)

metria degli oggetti da prelevare: per i parallelepipedi esaminati una buona scelta è quella di utilizzare il *3D Blob Locator Tool*, il cui principio di funzionamento è riportato in figura 3.1. Nella visione artificiale, il rilevamento blob (*blob analysis* o *blob detection*) è una tecnica che ha come obbiettivo il rilevamento di punti e/o regioni che differiscono in proprietà come luminosità o colore comparata con l'ambiente. Di fatto il rilevamento blob consente di rilevare una regione dell'immagine in cui alcune proprietà sono costanti: tutti i punti di un blob sono da considerarsi in qualche modo simili tra di loro. In letteratura esistono due classici rilevatori di blob: metodi differenziali basati su espressioni derivative e metodi basati sugli estremi locali [11]. I rilevatori blob si utilizzano tipicamente per fornire informazioni complementari sulle regioni, informazioni che non è possibile ottenere dal riconoscimento dei contorni o dal riconoscimento degli angoli. Tramite il *3D Blob Locator Tool* non è quindi necessario utilizzare dei modelli 3D dei pezzi da prelevare. Inoltre, nello stesso processo di visione, si possono identificare oggetti anche diversi tra di loro (ad esempio parallelepipedi più grandi di altri), cosa che invece non sarebbe possibile utilizzando altri strumenti (che invece si basano sulla forma o sulle dimensioni dell'oggetto da identificare). Dalla manualistica Fanuc emerge che è utile utilizzare il *3D Blob Locator Tool* per oggetti che hanno una superficie liscia e piana, appunto come i parallelepipedi presi in considerazione.

3.2.2 *Parametri da impostare nel processo di visione*

Di seguito sono riportati i parametri più importanti da impostare nel processo di visione al fine di un corretto riconoscimento dei pezzi tramite il *3D Blob Locator Tool*.

- Di default, non vengono forniti né il piano né l'orientazione dell'oggetto identificato. Queste due informazioni sono invece necessarie per la riuscita del bin picking. A tal fine, il *3D Blob Locator Tool* va messo in *advanced mode* e bisogna spuntare nel menù a tendina *calculate plane* e *calculate angle*. Il motivo per il quale di default non vengono forniti né il piano né l'orientazione dell'oggetto identificato è che ogni informazione aggiuntiva che deve essere fornita di fatto aggiunge del tempo di elaborazione delle immagini, e quindi rallenta il processo di visione. Per questo motivo, è bene richiedere al processo di visione solo le informazioni strettamente necessarie alla riuscita del programma, evitando di richiedere informazioni che poi non vengono sfruttate.

- È possibile impostare un numero massimo ed un numero minimo di *blob points*: in questo modo, i valori sopra o sotto ad una certa soglia non vengono riconosciuti come oggetti da prendere

e vengono rimossi dai risultati. Di fatto, l'impostazione di questi due parametri permette di stabilire un volume minimo o un volume massimo degli oggetti da riconoscere. In particolare, impostare un limite massimo su questo parametro è utile affinché pezzi estranei (troppo grandi) non vengano riconosciuti come oggetti da prendere. L'informazione sul numero di *blob points* può infatti essere efficacemente sfruttata per il riconoscimento di oggetti troppo grandi da prelevare, come illustrato successivamente in sezione 4.7. Va anche sottolineato che il numero di *blob points* dipenda dal valore impostato in *3D map density* durante la calibrazione del sensore di visione. Maggiore la densità della mappa 3D, maggiore il numero di blob points, a parità delle dimensioni degli oggetti.

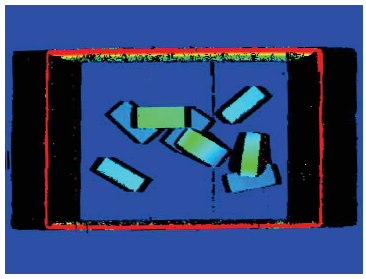
- È possibile impostare solo una certa finestra (*window*) in cui cercare gli oggetti, quindi ridurre l'area in cui l'algoritmo ricerca oggetti. Questo può portare ad un beneficio in termini di tempo ma è efficace solo se il contenitore si trova sempre nel medesimo punto e non viene mai spostato, altrimenti è consigliato lasciare l'area in cui vengono ricercati gli oggetti pari a tutta l'area inquadrata dal sistema di visione.

- Nel campo *Offset Data Calculation Tool* bisogna indicare al programma rispetto a quale sistema di riferimento fornire le coordinate degli oggetti identificati. Bisogna poi prestare particolare attenzione al valore impostato in questo campo una volta che si va a scrivere il programma per la movimentazione del robot.

3.2.3 *Acquisizione dati dal processo di visione*

Una volta impostati tutti i parametri nel processo di visione e quindi nel *3D Blob Locator Tool*, si è provveduto all'acquisizione dei dati dal medesimo processo di visione, per assicurarsi che in diverse condizioni il sistema sia sempre in grado di riconoscere i pezzi. In figura 3.2 è riportato un esempio di acquisizione dati dal sistema di visione. In figura 3.2a si nota che il sistema riconosce correttamente tutti i pezzi presenti all'interno della scatola e li numera secondo un certo criterio. All'interno del processo di visione è infatti possibile impostare un criterio con il quale il sistema di visione numera gli oggetti, che corrisponde all'ordine con il quale verranno poi prelevati. In questo caso il criterio impostato è l'altezza: vengono prelevati per primi i pezzi con la coordinata z maggiore. È anche possibile impostare una combinazione di più criteri, quindi scegliere il criterio con il quale vengono prelevati i pezzi che si trovano alla medesima quota z. In figura 3.2a si nota altresì che il parallelepipedo che si trova sotto agli altri e non è afferrabile non viene riconosciuto come oggetto che può essere preso: esso sarà riconosciuto solo nelle scansioni successive, eseguite dopo

(a) Vista dalla telecamera 1 con *3D Blob Locator*



(b) Ricostruzione della mappa 3D

Figura 3.2: Acquisizione dati dal sistema di visione

il pick and place dei primi pezzi. In figura 3.2b è invece riportata la ricostruzione della mappa 3D degli oggetti: come illustrato in sezione 2.5.6, le aree in rosso corrispondono ai punti più vicini alle telecamere ovvero quelli con la quota z maggiore (quindi i bordi della scatola), mentre quelle in blu corrispondono a quelle più distanti, ovvero quelle con la z minore (quindi il fondo della scatola e la superficie del piano sul quale è appoggiata la scatola stessa).

3.2.4 *Accuratezza dei risultati*

Per verificare l'accuratezza delle misure acquisite dal sistema di visione tramite il *3D Blob Locator Tool* sono state eseguite 85 scansioni dello stesso parallelepipedo posizionato in diversi punti del tavolo e con diverse orientazioni. In particolare, è stato analizzato l'andamento di *3D Blob Width* e *3D Blob Lenght*. Per come è stato posizionato il pezzo durante le prove, la lunghezza nominale è di 80 mm mentre la larghezza nominale è di 30 mm. Misurandole con il calibro, queste due misure sono risultate essere rispettivamente 80,02 mm e 30,03 mm. L'istogramma che rappresenta le frequenze delle misure di *3D Blob Width* è riportato in figura 3.3. Dall'analisi dei dati la larghezza del parallelepipedo risulta di $28,5 \pm 1,5$ mm. L'istogramma che rappresenta le frequenze delle misure di *3D Blob Lenght* è invece riportato in figura 3.4. Dall'analisi dei dati la lunghezza del parallelepipedo risulta di $77 \pm 1,5$ mm. In entrambi i casi, si nota che le misure non seguono una gaussiana, ma si concentrano su valori più elevati, vicino a quelli reali, e presentano una certa dispersione su valori inferiori a quelli reali. Grazie anche all'ausilio grafico, si può notare che il sistema di visione, quando sbaglia ad identificare correttamente una tra lunghezza e larghezza (ad esempio a causa di un illuminazione non ottimale), sbaglia in negativo, ovvero fornisce dei valori inferiori a quelli reali, mentre non fornisce praticamente mai valori superiori a quelli reali. È bene sottolineare che nel programma per il bin picking illustrato di seguito non si utilizzano direttamente

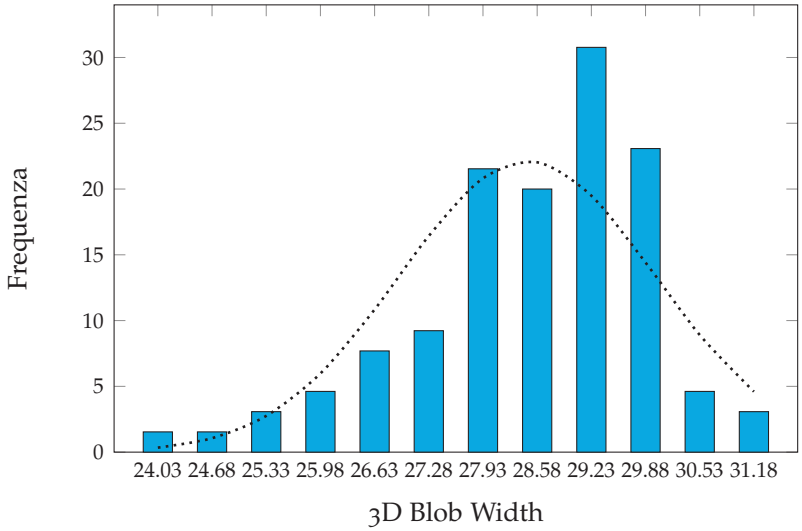


Figura 3.3: 3D Blob width (85 campioni)

i valori di larghezza e lunghezza del blob: i valori di larghezza e lunghezza sono utilizzati per calcolare il centro di massa. Ciò significa che non necessariamente un errore nel calcolo della lunghezza o larghezza del blob si traduce in un errore sul calcolo del centro di massa e di conseguenza una scorretta presa del pezzo. Ad ogni modo, il *3D Blob Locator Tool* è risultato sempre soddisfacente in quanto ha sempre consentito la corretta presa dei pezzi, che presentano comunque una geometria piuttosto semplice.

3.3 PROGRAMMA PER IL BIN PICKING

Di seguito è riportato il codice implementato per l'esecuzione del bin piking di oggetti posizionati in maniera random e sovrapposti tra di loro, posizionati sopra ad un tavolo oppure all'interno di una scatola. Il programma è composto di un programma principale, che si occupa delle operazioni di pick and place, e di un programma secondario, che si occupa di eseguire la scansione della mappa 3D. Questa soluzione è stata adottata in quanto è possibile notare che l'acquisizione della mappa 3D richiede un tempo considerevole, nell'ordine di qualche secondo (dai datasheet Fanuc, da 2 a 3 secondi a seconda di come è impostato il processo di visione). Pertanto, per minimizzare il tempo ciclo, ha senso che essa venga acquisita mentre il robot è in movimento, in particolare mentre sta eseguendo il place dei pezzi e si trova fuori dall'area inquadrata dal sistema di visione. Per fare ciò è necessario scrivere un programma multitasking mediante l'utilizzo della funzione RUN. Per approfondimenti riguardo alle istruzioni multitasking nel linguaggio di programmazione Fanuc si rimanda all'appendice A.10.

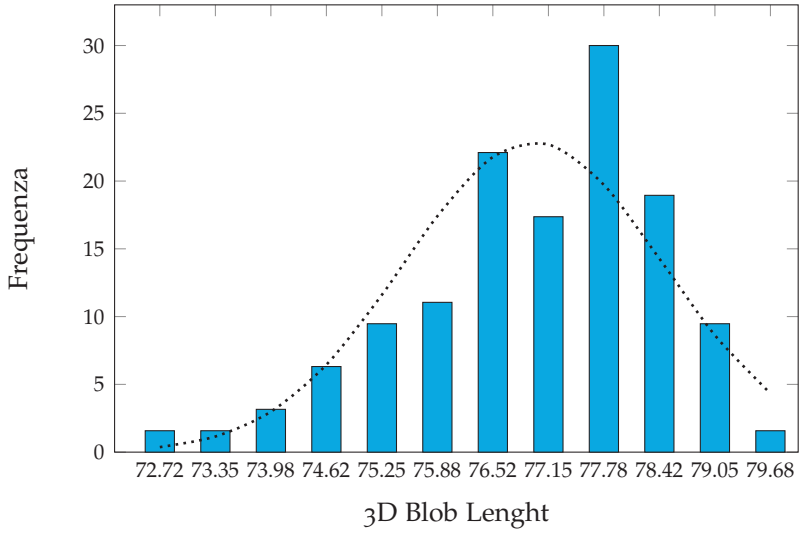


Figura 3.4: 3D Blob Lenght (85 campioni)

3.3.1 *Programma principale*

Il codice del programma principale per l'esecuzione del bin picking è riportato nelle righe che seguono. L'obbiettivo di questo programma è che il robot esegua una prima scansione dell'area inquadrata dal sistema di visione: se non sono presenti pezzi, il programma termina. Se è presente almeno un pezzo, il robot ne esegue il pick dal tavolo o dalla scatola e ne esegue il place dentro ad una seconda scatola. Mentre il robot esegue il place, esso si sposta dall'area inquadrata dal sistema di visione e, mentre esegue il movimento, esegue anche la scansione della mappa 3D. Se rileva almeno un altro oggetto da prelevare, il programma prosegue con la sua esecuzione. Se invece non rileva nessun oggetto da prelevare, il programma, dopo aver portato a termine il place del pezzo, termina.

```
! Programma per bin picking

! Definizione user frame e tool frame
UFRAME_NUM=1
UTOOL_NUM=3

! Il led della pinza viene impostato al colore verde
RO[7]=OFF
RO[8]=ON
! Si apre la pinza
CALL GRIP_OPEN

! Nel registro PR[100] viene salvata la z dell'approach al pezzo
PR[100]=LPOS-LPOS
PR[100,3]=(-30)

! Nel registro PR[99] viene salvato l'offset in z del pezzo
PR[99]=LPOS-LPOS
PR[99,3]=25

! Posizione di home
J P[3] 100% FINE
```

```
23
24 ! RUN del processo di visione
25 R[7]=0
26 RUN PRG_VISION
27 WAIT R[7]=1
28
29 ! Se non ci sono pezzi, il programma salta a LBL[3]
30 IF R[10]=0,JMP LBL[3]
31
32 LBL[2]
33
34 ! Posizione sopra la scatola
35 J P[1] 100% FINE
36
37 ! Si ottiene l'offset del pezzo, ovvero le sue coordinate
38 VISION GET_OFFSET 'REG_V_3DA2' VR[1] JMP LBL[3]
39
40 ! Approach posizione pick
41 L P[2] 100mm/sec CNT 100 Tool_Offset,PR[100] VOFFSET,VR[1]
42 ! Posizione di pick
43 L P[2] 100mm/sec FINE Tool_Offset,PR[99] VOFFSET,VR[1]
44 ! Chiusura pinze
45 CALL GRIP_CLOSE
46 ! Depart posizione pick
47 L P[2] 100mm/sec CNT100 Tool_Offset,PR[100] VOFFSET,VR[1]
48
49 ! Posizioni intermedie
50 J P[5] 100% CNT100
51 J P[5] 100% CNT100
52
53 R[7]=0
54 RUN PRG_VISION
55
56 ! Approach posizione place
57 L PR[4] 350mm/sec CNT100 Tool_Offset,PR[100]
58 ! Posizione di place
59 L PR[4] 350mm/sec FINE Tool_Offset,PR[99]
60 ! Apertura pinze
61 CALL GRIP_OPEN
62 ! Depart posizione place
63 L PR[4] 350mm/sec CNT100 Tool_Offset,PR[100]
64
65 ! Aspetta che il processo di visione abbia terminato la sua esecuzione
66 WAIT R[7]=1
67 JMP LBL[2]
68
69 ! Fine programma
70 LBL[3]
```

Sono necessarie alcune osservazioni:

- Il comando VISION GET_OFFSET fornisce l'offset del pezzo riconosciuto rispetto a quanto impostato sul campo *Offset Data Calculation Tool* nel processo di visione. Una soluzione che funziona è impostare $(0, 0, 0)$ sia in *Reference XYZ* che in *Reference WPR*. In questo modo il comando VISION GET_OFFSET fornisce direttamente le coordinate del pezzo da prendere rispetto all'user frame impostato (nel caso in esame l'user frame è una terna posizionata sull'angolo del tavolo). Di conseguenza, in questo caso, il dato posizione P[2] dovrà contenere $(0, 0, 0)$ in XYZ e $(180, 0, 0)$ in WPR.

- Tramite il comando alla riga 41, il robot esegue l'approach al pezzo, rimanendo sopra alla superficie dello stesso di una quantità pari alla somma tra il valore contenuto all'interno del registro PR[100] (impostabile ad inizio programma) ed il tool offset (che dipende appunto dall'utensile impostato come utensile corrente).

- Tramite invece il comando alla riga 43, il robot va a posizionarsi nella posizione di pick del pezzo. Riguardo alla posizione di pick, nel processo di visione 3D è stato impostato che venga fornita la posizione del *gravity center* e la quota z che viene fornita è quindi quella della superficie superiore del pezzo. Se l'organo terminale fosse una ventosa, non ci sarebbe bisogno di ulteriori offset. Poiché l'organo terminale è una pinza, è necessario definire un ulteriore offset, in questo caso contenuto in PR[99], che indichi la quota in z di cui scendere rispetto alla superficie del pezzo per poter realizzare la condizione di presa. È possibile modificare tale quantità all'inizio del programma.

3.3.2 *Programma per l'acquisizione della mappa 3D*

Di seguito è riportato il programma per l'acquisizione della mappa 3D, richiamato dal programma principale sia all'inizio del programma sia ogni volta che si esegue il place dell'oggetto. Questo programma esegue la scansione 3D secondo i parametri impostati, esegue il processo di visione ed inserisce nel registro R[10] il numero di pezzi trovati, in modo che questa informazione possa essere letta anche dal programma principale ed in modo che lo stesso sia in grado di capire se ci sono ancora pezzi da prelevare oppure no.

```
1  ! Programma per l'acquisizione della mappa 3D
2
3  ! Acquisizione della mappa 3D
4  CALL BINPICK_ACQUIRE3DMAP("3D Area Sensor"='CAL_NEW')
5
6  ! Chiamata al processo di visione
7  VISION RUN_FIND 'REG_V_3DA2'
8
9  ! Nel registro R[10] viene messo il numero di pezzi trovati
10 VISION GET_NFOUND 'REG_V_3DA2' R[10]
11
12 R[7]= 1
```

Tra i comandi utilizzati:

- BINPICK_ACQUIRE3DMAP è un sottoprogramma predefinito che si occupa di eseguire l'acquisizione della mappa 3D. L'argomento di questa chiamata a sottoprogramma deve essere il nome che è stato dato alla calibrazione del sensore di visione. In questo caso, la calibrazione utilizzata aveva il nome CAL_NEW.

- Il comando VISION RUN_FIND serve per eseguire il processo di visione e deve essere seguito dal nome dello stesso: in questo caso il processo di visione era stato chiamato REG_V_3DA2.
- Il comando VISION GET_NFOUND fornisce il numero di oggetto trovati dal processo di visione specificato e mette il risultato in un registro che viene indicato dopo al comando, in questo caso nel registro R[10].

3.4 CONSIDERAZIONI E LIMITI DELLA SOLUZIONE PROPOSTA

Di seguito sono evidenziati una serie di limiti della soluzione proposta, nonché considerazioni e spunti per i lavori futuri.

- La pinza Schunk Co-act utilizzata presenta diversi limiti e non è ottimale per il bin picking in quanto, ad esempio, non permette di afferrare oggetti molto vicini tra di loro (per i quali sarebbe più efficace l'utilizzo di una ventosa). Dai numerosi riferimenti in letteratura [4] e [12], è evidente che la riuscita ottimale di un processo di bin picking, soprattutto di oggetti di forma irregolare, dipende tanto dal corretto riconoscimento degli oggetti da parte del sistema di visione 3D quanto da un'attenta scelta dell'end-effector. In questo caso, è stato necessario utilizzare la pinza Schunk Co-act presente in laboratorio. Lavori futuri potranno approfondire l'utilizzo di griffe diverse, magari grazie all'utilizzo della stampa additiva oppure l'utilizzo di organi terminali di tipo pneumatico quali ventose. La pinza utilizzata può andar bene nel caso il contenitore sia semi vuoto, gli oggetti siano distanti tra di loro e le pareti del contenitore siano non troppo alte oppure nel caso di *table picking*, ovvero nel caso in cui gli oggetti siano posizionati alla rinfusa sopra un tavolo (questo elimina la problematica delle collisioni con le pareti del contenitore).
- Gli oggetti su cui è stato fatto il riconoscimento e la successiva presa sono dei parallelepipedi, per i quali è stato sufficiente l'utilizzo del *3D Blob Locator Tool*. Sarebbe interessante estendere la soluzione proposta anche a geometrie più complesse, per le quali l'utilizzo del solo *3D Blob Locator Tool* probabilmente non sarebbe sufficiente.
- È opportuno sottolineare che molto spesso il bin picking viene fatto su parti tutte uguali tra di loro (o comunque molto simili) in modo che si possa utilizzare lo stesso organo terminale. Volendo realizzare il bin picking con oggetti diversi tra di loro è necessario utilizzare un organo terminale multi utensile, come ad esempio riportato in [6]. Sarebbe interessante approfondire anche questa tematica. Tuttavia, va evidenziato che un organo

terminale più sofisticato risulta anche in un numero maggiore di possibili singolarità che vanno tenute in considerazione per una pianificazione ottimale della traiettoria.

- Non è stata approfondita l'influenza dell'illuminazione ambientale sulla buona riuscita del riconoscimento degli oggetti, così come non è stato approfondito l'utilizzo di un colore diverso della superficie del tavolo, in quanto la soluzione proposta (tavolo bianco, illuminazione tramite led) sembrava funzionare bene ed essere robusta.

4

ALGORITMO PER BIN PICKING COLLABORATIVO

4.1 INTRODUZIONE

Una volta collaudato un algoritmo per il bin picking tradizionale, si è proceduto alla ricerca di un algoritmo per realizzare il bin picking collaborativo. L'obbiettivo di questo lavoro è quello di rendere collaborativo il task di bin picking, quindi di riuscire a far collaborare insieme robot ed operatori. A tal fine, è necessario sia identificare la presenza dell'operatore che monitorarne la posizione. Per riconoscere l'operatore è possibile utilizzare una telecamera esterna. Tuttavia, l'idea innovativa sviluppata durante questo lavoro di tesi è stata di quello di utilizzare il sistema di visione 3D sia per il riconoscimento degli oggetti del quale eseguire il bin picking, sia per il riconoscimento degli operatori. Essendo il sensore di visione 3D analizzato un sensore di visione a luce strutturata, esso è composto di un proiettore e due telecamere: si può sfruttare una delle due telecamere per il riconoscimento degli operatori, che sarà quindi basato sulla visione 2D. Questa soluzione presenta il vantaggio di non richiedere un ulteriore telecamere ma presenta anche l'ovvia limitazione che mentre viene eseguita una scansione 3D viene interrotto l'utilizzo della telecamera 2D. Chiaramente, utilizzando lo stesso sistema di visione per due scopi differenti è necessario sincronizzare la visione 3D e quella 2D.

4.2 PRINCIPIO DI FUNZIONAMENTO

Negli ultimi anni c'è stata una ricerca crescente in merito agli algoritmi per il riconoscimento di operatori tramite delle telecamere, nonché per il riconoscimento dei gesti degli esseri umani. L'ideale sarebbe che questi algoritmi non richiedano l'utilizzo di dispositivi hardware da parte delle persone, ovvero che siano in grado di identificare le persone ed i loro movimenti senza che le stesse debbano indossare o tenere in mano particolari dispositivi. Tra le varie possibilità, è stata investigata la possibilità di fare indossare all'operatore un bracciale rigido sul quale è incollato un target caratterizzato da una geometria particolare, che viene poi riconosciuto dal sistema di visione. In questo modo, si è in grado sia di riconoscere la presenza dell'operatore sia di monitorare in maniera robusta la sua posizione. L'utilizzo di un bracciale potrebbe essere inaccettabile per algoritmi che si occupano del riconoscimento delle persone in altri ambiti (ad esempio per monitorare il numero di clienti all'interno di un negozio), ma risulta accettabile nel caso si tratti di operatori in un luogo di lavoro. Inoltre,

un bracciale rigido è un dispositivo decisamente poco invasivo, che non disturba in alcun modo l'operatore.

4.3 PROCESSO DI VISIONE 2D

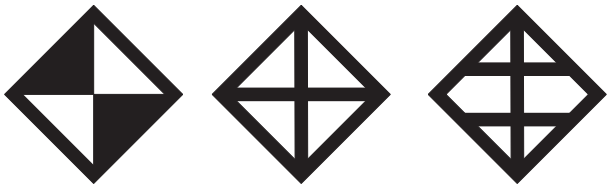
Come già precedentemente esposto, il sistema Fanuc iRVision a disposizione si basa sull'impostazione di processi di visione tramite PC e poi sul richiamare tali processi nei programmi scritti da teaching pendant o iRProgrammer. Per permettere il continuo riconoscimento di un operatore è quindi necessario impostare un secondo processo di visione, basato su una sola telecamera e sulla visione 2D, diverso da quello precedentemente impostato per il riconoscimento dei pezzi tramite *3D Blob Locator Tool*. Questo processo di visione è un processo di visione 2D (che in ambiente Fanuc prende il nome di *2D Single View Vision Process*) e sfrutta una delle due telecamere: in questo caso, è stata impiegata la telecamera 1 (per la quale deve essere effettuata una propria calibrazione). Il processo di visione 2D viene avviato dal programma principale e deve continuare ad iterare per monitorare l'eventuale presenza di operatori, a differenza invece del processo di visione 3D, che viene richiamato tramite l'opportuna istruzione ogni volta che si vuole eseguire il pick di un nuovo pezzo. Il processo di visione 2D termina solamente quando il robot si ferma perché non ha più pezzi da prelevare. Per il riconoscimento del target, per il quale sono state prese in considerazione una serie di geometrie diverse come illustrato in sezione 4.3.1, uno strumento (tool) efficace da utilizzare è il *GPM Locator Tool*. La sigla *GPM* sta per *Geometric Pattern Matching*: questo tool confronta le caratteristiche geometriche dell'immagine rilevata e quelle del target precedentemente impostato nel processo di visione. Questo metodo di riconoscimento è un sistema robusto che riesce a trovare il target nonostante le variazioni di forma, angolo ed illuminazione. In generale, la sua buona riuscita dipende molto dall'illuminazione e dalla condizione dei materiali (ad esempio se sono riflettenti o meno).

4.3.1 *Scelta del target ottimale*

Sono state eseguite una serie di prove al fine di determinare quale target sul bracciale dell'operatore fosse in grado di garantire il minor numero di falsi positivi (ovvero di oggetti riconosciuti come target che in realtà non sono tali) possibili e che allo stesso modo fosse in grado di rilevare in maniera robusta la presenza dell'operatore. I target analizzati sono riportati in figura 4.1 ed in figura 4.2. Per ognuno dei target, è stato creato il relativo processo di visione con *GPM Locator Tool* ed è stata eseguita una presa di 10 pezzi (tutti orientati casualmente e sovrapposti) senza che l'operatore entri nell'area di lavoro del robot. I risultati ottenuti sono i seguenti:

(a) (b) (c)

Figura 4.1: Target circolari

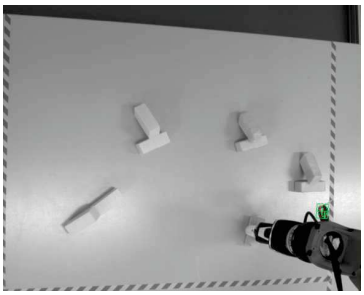


(a) (b) (c)

Figura 4.2: Target a forma di rombo

- Per il target riportato in figura 4.1a, durante la presa dei 10 pezzi si sono verificati due falsi positivi. Il primo, riportato in figura 4.3a (al posto di riconoscere il target il sistema di visione riconosce i cavi presenti sulla pinza del robot), risolto aumentando la soglia sullo score minimo del *GMP Locator Tool*. Il secondo, riportato in figura 4.3b, si può risolvere facilmente coprendo il segnale di sicurezza presente sul braccio del robot, che viene erroneamente riconosciuto al posto del target.
- Per i target circolari riportati in figura 4.1b e 4.1c, dopo gli accorgimenti adottati al punto precedente per evitare i falsi positivi dovuti ai cavi nel polso del robot ed ai simboli di sicurezza, non si sono verificati falsi positivi.
- Per il target a forma di rombo riportati in figura 4.2a, 4.2b e 4.2c non si sono verificati falsi positivi ed, a parità di parametri impostati nel *GMP Locator Tool*, il sistema sembra riconoscere più rapidamente il target.

Una volta adottati gli accorgimenti per risolvere le problematiche riscontrate con il target 4.1a, tutte le prove hanno dato risultati soddisfacenti e pertanto tutti i target risultano utilizzabili come target da posizionare nel bracciale rigido da far indossare all'operatore. Dalle prove eseguite ciò che emerge è che generalmente i target circolari presentano più problematiche, in quanto è più facile che vengano riconosciuti dei falsi positivi ad esempio per della segnaletica di sicurezza presente nei pressi dell'area di lavoro. Il target selezionato per le prove successive è stato il target 4.1a.



(a) Cavo presente nel polso del robot riconosciuto come falso positivo

(b) Simbolo di sicurezza nel braccio del robot riconosciuto come falso positivo

Figura 4.3: Esempi di falsi positivi ottenuti con target circolari

4.3.2 *Parametri da impostare nel processo di riconoscimento*

Ci sono alcuni parametri da impostare nel processo di visione che sono degni di nota. I due tool che vengono utilizzati nel semplice processo di visione impostato sono lo *Snap Tool* ed il *GPM Locator Tool*.

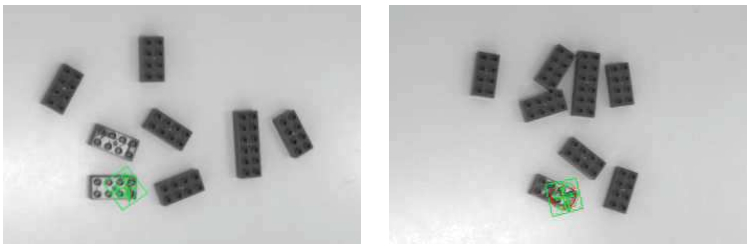
4.3.2.1 *Snap tool*

Per prima cosa, lo strumento presente in ogni processo di visione e che si occupa di acquisire un immagine è lo *Snap Tool*. Tra i vari parametri impostabili, si riconosce il campo *Resolution Reduction*, ovvero il campo in cui è possibile impostare una diminuzione della risoluzione della telecamera. Di default, la risoluzione delle telecamere è 1280×1024 pixel. La riduzione della risoluzione è stata impostata a 2x, quindi la risoluzione è stata diminuita a 640×512 pixel. In questo modo, il riconoscimento richiede un tempo minore in quanto i dati da processare sono minori: questo è particolarmente utile in quanto il processo che riconosce l'operatore deve continuare ad iterare ed è importante che il riconoscimento avvenga nel minor tempo possibile per garantire la sicurezza degli operatori.

4.3.2.2 *GPM Locator Tool*

Il *GPM Locator Tool*, come precedentemente illustrato, si occupa di riconoscere un certo pattern geometrico all'interno dell'immagine acquisita. Esso va impostato in *advanced mode* ed al fine di eseguire un corretto riconoscimento del target i due parametri a cui porre maggiore attenzione sono:

- *Score threshold*: questo parametro stabilisce il valore minimo di *score* che deve avere l'oggetto o la forma vista dal sistema di visione per essere riconosciuta come target. Lo *score* è di fatto un punteggio attribuito agli oggetti riconosciuti che indica



(a) Elasticità 4.5 pix, target a forma di rombo

(b) Elasticità 3.5 pix, target circolare

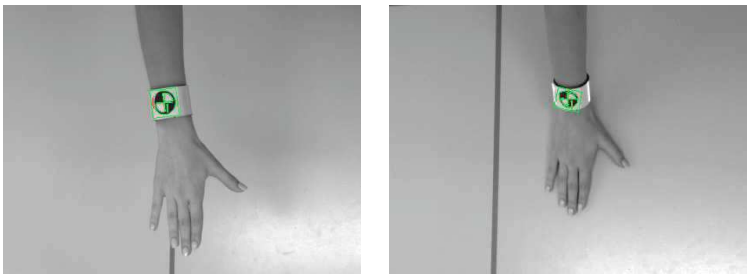
Figura 4.4: Esempi di falsi positivi ottenuti a causa di un valore di elasticità troppo elevato

quanto l'oggetto riconosciuto si avvicini all'oggetto precedentemente insegnato come oggetto target (ovvero come pattern geometrico da individuare). Impostare un valore troppo basso del parametro *score threshold* (ad esempio 50%) può portare al riconoscimento in figura 4.3a, ovvero a falsi positivi. Un valore troppo alto di questo parametro (ad esempio superiore al 90%) non permette di identificare il target in posizioni o orientamenti leggermente diverse rispetto a quella insegnata. Poiché la mano dell'operatore può trovarsi ad un'altezza diversa, è bene sottolineare che maggiore la quota z, più grande sarà visto il target e quindi è necessario prevedere che il target possa avere una dimensione in termini di pixel leggermente variabile.

- *Elasticity*: questo parametro, impostabile da 0.1 a 5 pixel, stabilisce l'elasticità del target, ovvero quanto esso si possa deformare dalla geometria che è stata precedentemente insegnata. Nella manualistica Fanuc, è raccomandato che questo valore venga lasciato al valore di default di 1.5 pix. Infatti, aumentando questo parametro a valori superiori, ad esempio a 3 pix, si nota che vengono riconosciuti oggetti che non dovrebbero essere riconosciuti, sia utilizzando target a forma di rombo (figura 4.4a) sia utilizzando target circolari (figura 4.4b). In questi due casi, oltre al valore troppo elevato di elasticità, sono complici anche il riflesso dell'illuminazione ambientale e gli oggetti dalla superficie irregolare.

4.3.3 *Numero di GPM Locator Tool da impostare*

In ambiente Fanuc iRVision, un processo di visione sia esso 2D o 3D, può essere composto di più strumenti (tool) che concorrono alla generazione del riconoscimento degli oggetti. Per il riconoscimento del target posizionato sul bracciale da far indossare all'operatore, può essere necessario l'utilizzo di uno o più *GPM Locator Tool*, a seconda di qual è la posizione reciproca tra sistema di visione, piano di lavoro ed



(a) Mano distesa sopra al piano di lavoro (target ben visibile)

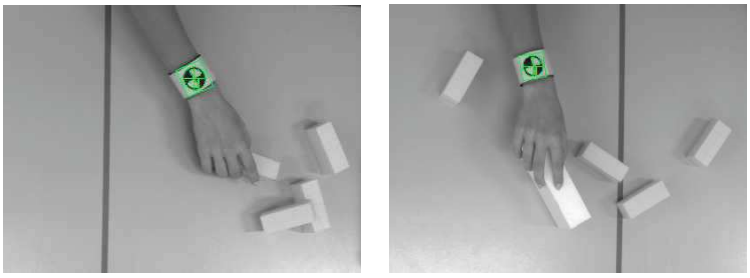
(b) Mano appoggiata sul piano si lavoro (target distorto)

Figura 4.5: Vista del target sulla mano tramite visione 2D nel caso di operatore in piedi

operatore. In particolare, in base ai test sperimentali effettuati è possibile affermare che è necessario distinguere il caso in cui l'operatore si trovi a lavorare in piedi oppure il caso in cui l'operatore si trovi a lavorare seduto.

4.3.3.1 *Operatore che lavora in piedi*

Se è previsto che l'operatore lavori in piedi sopra al tavolo di lavoro, il riconoscimento della mano può risultare complicato: la mano infatti può trovarsi distesa sopra al piano di lavoro, come in figura 4.5a, oppure appoggiata sul piano di lavoro ad esempio per la presa di un oggetto, come in figura 4.5b. Nel primo caso il target è ben visibile e si trova ad una quota z elevata, mentre nel secondo caso il target appare distorto e si trova ad una z quasi nulla, poiché il braccio dell'operatore è quasi in verticale. I test sperimentali condotti evidenziano che nel caso di operatore che lavora in piedi la soluzione ottimale, tenendo anche conto dei possibili falsi positivi, è impostare due *GPM Locator Tool* separati: uno dei due che abbia come target di riferimento un'immagine della mano distesa, quindi del target non distorto, e l'altro che abbia come target di riferimento un'immagine della mano appoggiata sul piano di lavoro, ovvero del target distorto. Poiché questo processo serve per il riconoscimento di un operatore nella zona di lavoro del robot, impostare due *GPM Locator Tool* aumenta i tempi di riconoscimento del target e quindi dell'operatore, il che non è ovviamente desiderabile. Va anche sottolineato che potrebbe essere utilizzato un solo *GPM Locator Tool* con riferimento un'immagine della mano distesa aumentando a 4 o 4.5 pixel il valore dell'elasticità. In questo modo il target viene effettivamente sempre riconosciuto, ma vengono anche riconosciuti falsi positivi e pertanto non è la soluzione migliore.



(a) Operatore che afferra oggetti nel lato destro del tavolo

(b) Operatore che afferra oggetti nel lato sinistro del tavolo

Figura 4.6: Vista del target sulla mano tramite visione 2D nel caso di operatore seduto

4.3.4 *Operatore che lavora seduto*

Se è invece previsto che l'operatore lavori seduto, il riconoscimento della mano risulta più semplice: il target infatti si troverà quasi sempre orizzontale rispetto al sistema di visione e si trova ad una quota z quasi sempre nulla. In figura 4.6 sono riportati due esempi di ciò che vede il sistema di visione 2D in caso di operatore seduto. Nelle figure 4.6a e 4.6b, scattate in due zone diverse del piano di lavoro (quindi con possibili distorsioni dovute all'inclinazione della telecamera 1), il target appare sempre ben visibile. In questo caso, poiché il target non appare ma eccessivamente distorto, è sufficiente impostare un solo *GMP Locator Tool*,che abbia come target di riferimento una qualsiasi delle immagini acquisite, simili a quelle riportate in figura 4.6. L'impiego di un solo *GMP Locator Tool* ha quindi il vantaggio di richiedere un tempo minore per il riconoscimento del target, aspetto sempre positivo dato che si tratta di riconoscere un operatore presente all'interno dell'area di lavoro e quindi è desiderabile che il sistema sia il più reattivo possibile.

4.3.5 *Processi di visione separati*

È inoltre stata investigata la possibilità di impiegare due processi di visione separati, utilizzando entrambe le due telecamere 2D. Poiché le telecamere 2D sono istallate inclinate di un certo angolo come in figura 2.9, se il polso dell'operatore è eccessivamente inclinato può essere che una sola telecamera non sia adatta al riconoscimento del target. Sono stati quindi impostati due processi di visione 2D separati, uno che utilizzi la telecamera 1 ed uno che utilizzi la telecamera 2, entrambi impostati con le medesime logiche e con gli stessi tool. Tuttavia, l'utilizzo di due processi di visione separati non comporta miglioramenti apprezzabili nelle prestazioni del robot rispetto alla soluzione che utilizza una sola telecamera. Al contrario, l'utilizzo di due processi di visione separati può dare luogo ad ulteriori problematiche.

Infatti, a causa dell'inclinazione delle telecamere e del loro posizionamento, esse vedranno in delle coordinate (x, y) diverse il medesimo target e questo può complicare le applicazioni di bin picking collaborativo. Ad esempio, se l'applicazione collaborativa prevede che la zona di lavoro sia divisa in due zone, a causa della distorsione delle telecamere, può essere che in posizione a cavallo tra le due zone una telecamera veda l'operatore in zona 1 e l'altra telecamera veda l'operatore in zona 2. Dai test eseguiti è possibile concludere che l'utilizzo di due processi di visione separati introduce una complessità maggiore non giustificata da un aumento delle prestazioni.

4.4 STRUTTURA GENERALE DEL PROGRAMMA

Ciò che è emerso dargli esperimenti eseguiti è che per la corretta riuscita di un programma di bin picking collaborativo, è necessaria una corretta sincronizzazione tra l'esecuzione del bin picking ed il riconoscimento degli operatori. Questa sincronizzazione è necessaria a maggior ragione se si utilizza il medesimo sistema di visione per il riconoscimento sia degli oggetti che degli operatori. In linea di principio, è necessario predisporre un programma che si occupi della scansione 3D e del pick and place degli oggetti, simile a quello illustrato al capitolo precedente. È poi necessario predisporre un ulteriore programma che continui ad iterare e che si occupi della scansione 2D e quindi del riconoscimento di un operatore. Trattandosi di una serie di programmi multitasking, è necessario porre particolare attenzione a quale programma faccia partire gli altri tramite istruzioni RUN e CALL. Dopo alcune prove, si è concluso che la struttura migliore del programma è quella illustrata di seguito:

- È necessario predisporre un programma principale (un programma *main*) che mandi le istruzioni di RUN a due programmi separati, uno che si occupi della visione 2D ed uno che si occupi del bin picking. In questo modo, nessuno dei due programmi ha la precedenza sull'altro e non si rischia che uno dei due programmi continui ad essere eseguito mentre l'altro attenda (fenomeno che in informatica prende il nome di *starvation*). Ciò che è stato osservato tramite test sperimentali è invece che se il programma che esegue il bin picking richiama per prima cosa il programma che si occupa del riconoscimento degli operatori, sembra che venga data la precedenza al programma che si occupa del bin picking e che contiene istruzioni di movimento, mentre il programma che si occupa del riconoscimento degli operatori soffre appunto di *starvation*.

- Il programma che si occupa della visione 2D e quindi del riconoscimento degli operatori deve essere composto da un ciclo che continua ad iterare fino a che non sono esauriti i pezzi da

prelevare ovvero fino a quando l'operazione di bin picking non è terminata. In ordine, il programma che si occupa della visione 2D e del riconoscimento degli operatori deve:

  - Controllare che il programma che si occupa del bin picking non abbia terminato per assenza di pezzi da prelevare: in caso affermativo, anche il programma che si occupa della visione 2D deve terminare. Questo check può essere fatto facilmente verificando il contenuto di un registro predefinito.
  - Se il programma che si occupa di bin picking è ancora in esecuzione (quindi ci sono ancora pezzi da prelevare), il programma di visione 2D chiama mediante un'istruzione CALL un sottoprogramma che si occupa della scansione 2D, quindi del riconoscimento della presenza dell'operatore. Nel caso sia presente un operatore, il sottoprogramma si può occupare anche di determinare in quale zona dell'area di lavoro l'operatore si trova (nel caso di area di lavoro suddivisa in due zone), di determinare a quale distanza l'operatore si trovi rispetto al robot o eseguire altre elaborazioni, a seconda dell'applicazione.
  - Il programma di visione 2D si occupa poi di eseguire delle scelte decisionali in base alla presenza dell'operatore ed a seconda della zona in cui eventualmente lo stesso si trova. Ad esempio, può occuparsi di impostare correttamente i led RO[1] e RO[2] nonché la velocità del robot.

- Il programma che si occupa del bin picking e della movimentazione del manipolatore è costituito da una serie di istruzioni di JMP LBL (istruzioni di salto condizionato) che di fatto consentono al programma di iterare fino a che non ci sono più pezzi nell'area di lavoro. Questo programma si occupa in ordine di:
  - Impostare di una serie di parametri iniziali quali *user frame* e *tool frame* e valori corretti nei registri dati e di posizione.
  - Chiamare tramite l'istruzione RUN il sottoprogramma che si occupa di eseguire la scansione 3D e di mettere in un registro il numero di oggetti trovati. Se non vengono trovati pezzi da prelevare (ovvero il registro predefinito contiene uno zero), il programma termina, altrimenti procede.
  - Ottenere le informazioni sul primo pezzo da prelevare. A seconda del programma specifico, può essere necessario riconoscere in quale zona si trova il pezzo, oppure fare un controllo sulle sue dimensioni per distinguere pezzi di forma e geometria diversa. Ad esempio, se un pezzo è troppo grande per essere prelevato, è possibile eseguire una chiamata tramite istruzione CALL ad un sotto programma che

si occupi della gestione di pezzi troppo grandi (ad esempio si attende la pressione del pulsante RI[1]). A seconda della dimensione dei pezzi o della loro zona, è anche possibile specificare una diversa posizione di place.

– Una volta ottenute le informazioni necessarie ed eventualmente aggiornati valori di registri dati e di posizione, si procede con il pick e successivamente con il place degli oggetti. Mentre viene eseguito il place, ed il robot si trova fuori dall'area ripresa dal sensore di visione, viene eseguita anche una nuova scansione 3D.

In figura 4.7 è riportato il flowchart di un generico programma per bin picking collaborativo.

4.5 POSSIBILI APPLICAZIONI DI BIN PICKING COLLABORATIVO

Una volta collaudato l'algoritmo per il riconoscimento dell'operatore e verificato che l'operatore venga sempre riconosciuto in tempi ragionevoli, si possono prevedere diverse applicazioni di bin picking collaborativo, come già precedentemente illustrato in 1.5. Essere in grado di riconoscere l'operatore e monitorarne la posizione presenta infatti un'enorme potenzialità che può essere sfruttata in diversi modi. In particolare, è possibile regolare la velocità del manipolatore in funzione alla distanza tra esso e l'operatore. In alternativa, è possibile prevedere che l'area di lavoro del robot sia divisa in due o più zone ed il robot rallenti solo se operatore e robot si trovano nella stessa zona dell'area di lavoro. È inoltre possibile fare in modo che il robot si fermi del tutto oppure cambi traiettoria quando vede che operatore e robot sono troppo vicini e si rischia la collisione. Si possono poi prevedere applicazioni nelle quali robot ed operatore devono lavorare allo stesso task insieme, quindi far fare al robot una certa operazione, successivamente attendere che l'operatore entri nella zona ed esegua il suo task, quindi esca dalla zona in modo che il robot può riprendere con l'operazione successiva. È anche possibile prevedere che il compito dell'operatore sia semplicemente quello di scaricare nuovi oggetti da prelevare: in questo caso, quando il sistema di visione si accorge della presenza di un operatore il robot si sposta in una posizione al di fuori del campo visivo del sistema di visione, aspetta che l'operatore si sposti ed esegue nuovamente una scansione.

4.6 ESEMPIO DI PROGRAMMA PER BIN PICKING COLLABORATIVO

Tra le varie applicazioni di bin picking collaborativo che è possibile prevedere, di seguito ne è riportata una. L'applicazione della quale in seguito è riportato il codice prevede le seguenti caratteristiche:

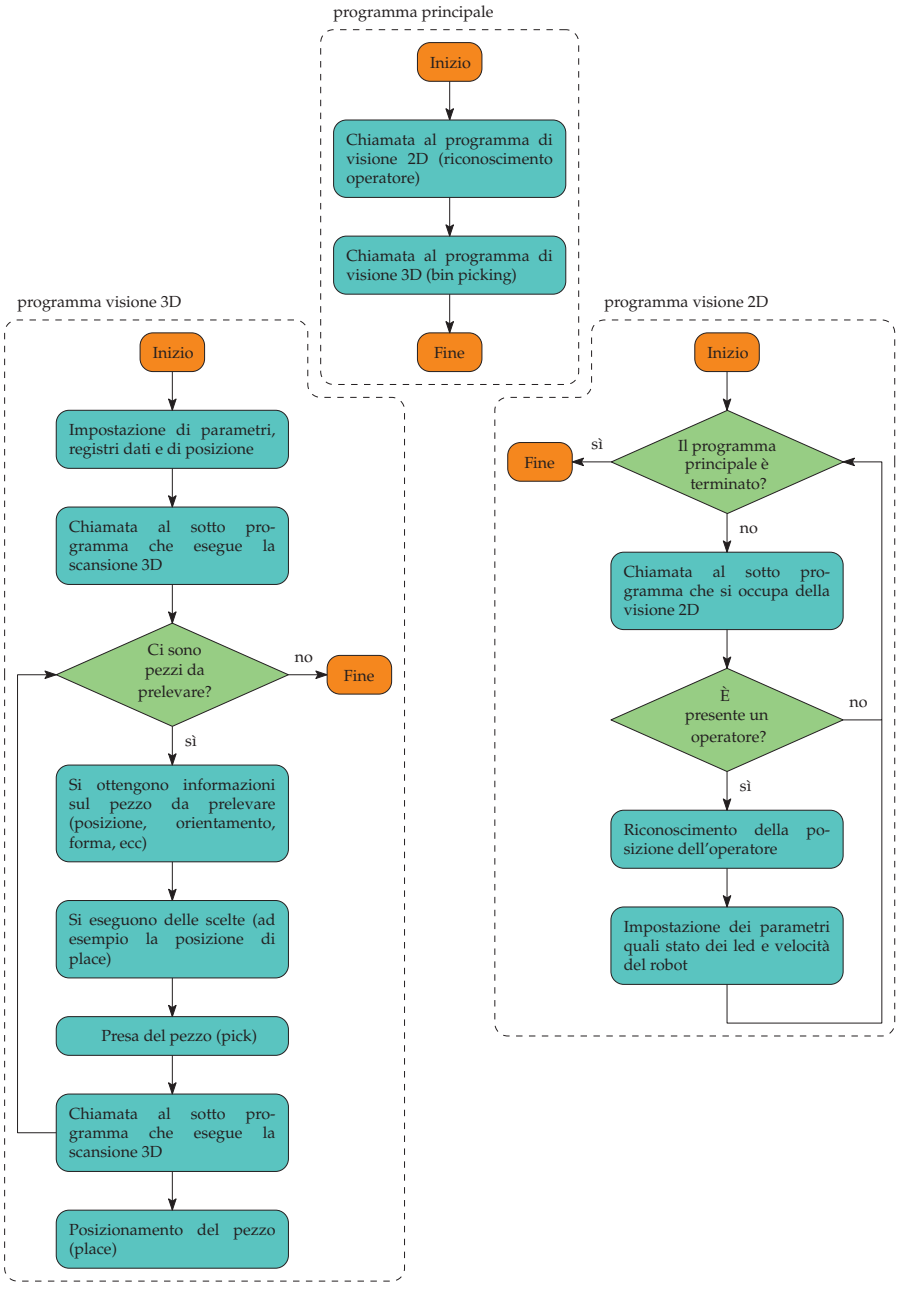


Figura 4.7: Flowchart di un programma per bin picking collaborativo

- L'area di lavoro (un tavolo, simile a quello in figura 2.1) è divisa in due zone rettangolari: la zona 1, nella parte sinistra del tavolo, e la zona 2, nella parte destra del tavolo.
- Se l'oggetto riconosciuto si trova in zona 1, il place deve essere eseguito in zona 1 e gli oggetti vengono disposti in fila a partire da una posizione iniziale. Se l'oggetto riconosciuto si trova in zona 2, il place deve essere eseguito in zona 2 e gli oggetti vengono anche in questo caso disposti in fila.
- Se il sistema di visione riconosce che l'oggetto è troppo grande per essere riconosciuto, il robot si sposta in una posizione al di fuori dell'area illuminata dal proiettore ed attende che l'operatore rimuova il pezzo e proceda alla successiva pressione di un pulsante. Una volta che viene premuto il pulsante, il sistema di visione esegue nuovamente la scansione 3D e l'operazione di bin picking riprende.
- Se l'operatore si trova in zona 1 ed il robot si accinge a prelevare un pezzo nella medesima zona, il robot rallenta, portandosi ad una velocità pari al 30% della sua velocità nominale. Se invece il robot deve prelevare un oggetto nella zona 2, la sua velocità non viene modificata. Analogamente, se l'operatore si trova in zona 2, il robot viene fatto rallentare solo se esso deve prelevare un pezzo in zona 2, altrimenti prosegue senza che la sua velocità venga diminuita.

Di seguito è riportato il codice dei sotto programmi che compongono il programma per la corretta esecuzione dell'applicazione di bin picking collaborativo appena illustrata.

4.6.1 *Programma principale*

Coerentemente con quanto illustrato nel flowchart in figura 4.7, il programma principale (main), riportato di seguito, si occupa di chiamare in ordine prima il programma di visione 2D che gestisce il riconoscimento degli operatori e successivamente il programma di visione 3D che gestisce le operazioni di bin picking. Prima di richiamare i due programmi tramite istruzione RUN, viene anche azzerato il registro R[20], che viene utilizzato per sincronizzare il programma di visione 2D con quello di visione 3D. In particolare, questo registro viene utilizzato per fare in modo che il programma di visione 2D che riconosce la presenza degli operatori termini quando termina il programma di bin picking, ovvero quando non ci sono più pezzi da prelevare (altrimenti continuerebbe ad iterare all'infinito).

```
1 ! Main program
2
3 R[20]=0
4 RUN TS_VISION
```

```
RUN TS_BIN_PICKING
```

4.6.2 *Programma di visione 2D*

Di seguito è riportato il codice del programma di visione 2D che si occupa del riconoscimento degli operatori, che per questo programma è stato chiamato TS_VISION. Questo programma è costituito da un loop infinito che termina solo quando termina il programma principale (perché non ci sono più pezzi da prelevare). Per prima cosa viene richiamato il programma che esegue la scansione 2D e che riconosce la zona in cui si trova l'operatore, dopodiché vengono impostati il colore del led della pinza (verde se non è presente l'operatore, rosso se è presente l'operatore) e la velocità del robot (l'override[1] viene diminuito al 30% se operatore e robot si trovano nella stessa zona dell'area di lavoro).

```
LBL[1]

! Se il programma principale ha terminato, termina anche questo programma
IF R[20]=1,JMP LBL[3]

! Chiamata al programma che esegue la scansione 2D
CALL TS_VISION_OP

! Se non ci sono operatori
IF (R[31]=0) THEN

! Il led della pinza viene impostato sul verde e l'override al 100%
RO[7]=OFF
OVERRIDE=100%
WAIT .01(sec)

ELSE

! Il led della pinza viene impostato sul rosso
RO[7]=ON

! L'override viene diminuito solo se operatore e il pezzo in lavorazione si
    trovano nella stessa zona dell'area di lavoro
IF ((R[15]=1 AND R[13]=1) OR (R[16]=1 AND R[14]=1)) THEN
OVERRIDE=30%
ELSE
OVERRIDE=100%
ENDIF

WAIT .01(sec)
ENDIF

JMP LBL[1]

LBL[3]
```

1 L'override è un valore in percentuale che indica la velocità percentuale rispetto alla nominale. Ad esempio, se in un comando di movimento la velocità impostata è 100 mm/s e l'override è impostato al 30%, la velocità con la quale viene eseguito il movimento è 30 mm/s.

4.6.2.1 *Programma che esegue la scansione 2D*

Come illustrato, il programma TS_VISION chiama a sua volta il programma TS_VISION_OP, che esegue la scansione 2D da parte della telecamera 1 e si occupa di determinare in quale zona dello spazio di lavoro si trovi l'operatore, aggiornando conseguentemente il contenuto di opportuni registri. Questo programma è stato concepito per poi essere eventualmente duplicato anche per la seconda telecamera, e quindi per far lavorare contemporaneamente tutte e due le telecamere. Tuttavia, come esposto in 4.3.5, questa soluzione non sembra portare particolari benefici rispetto alla soluzione con una sola telecamera ma anzi porta solamente ulteriori problematiche.

```
! Si azzera il registro che contiene il numero di operatori riconosciuti
R[11]=0

! Y che discrimina tra le zone dello spazio di lavoro
R[17]=25

! Scansione 2D
VISION RUN_FIND 'REC2D4'

! In R[11] viene messo il numero di operatori riconosciuti
VISION GET_NFOUND 'REC2D4' R[11]

IF (R[11]=0) THEN
! Non ci sono operatori
R[31]=0

ELSE

! Ci sono operatori
R[31]=1
! Si ottiene la posizione dell'operatore
VISION GET_OFFSET 'REC2D4' VR[2] JMP LBL[5]

! Nel registro di posizione PR[10] viene messa la posizione dell'operatore
PR[10]=VR[2].FOUND_POS[1]
! Quota Y della posizione dell'operatore
R[18]=PR[10,2]

IF (R[18]<=R[17]) THEN
!L'operatore si trova in zona 1
R[15]=1
R[16]=0

ELSE

! L'operatore si trova in zona 2
R[15]=0
R[16]=1

ENDIF
ENDIF
```

4.6.3 *Programma di visione 3D*

Il programma che si occupa della visione 3D e quindi del bin picking si basa sul programma precedentemente illustrato in 3.3.1. Questo

programma esegue la scansione 3D, riconosce la zona in cui si trova l'oggetto[2] e riconosce la grandezza dell'oggetto da prelevare. Se l'oggetto da prelevare è troppo grande, viene richiamato un programma apposito e si attende l'intervento di un operatore. Se invece l'oggetto è effettivamente prelevabile, in base alla zona in cui esso si trova, viene anche impostata la posizione di place, che poi viene incrementata ad ogni posizionamento in modo da disporre in fila gli oggetti.

```
! Programma per bin picking

! Definizione user frame e tool frame
UFRAME_NUM=1
UTOOL_NUM=3

! Il led della pinza viene impostato al colore verde
RO[7]=OFF
RO[8]=ON
! Vengono spenti i due led presenti sul robot
RO[1]=OFF
RO[2]=OFF

! Si azzera il numero di oggetti identificati
R[24]=0

! Nel registro PR[100] viene salvata la z dell'approach al pezzo
PR[100]=LPOS-LPOS
PR[100,3]=(-30)

! Nel registro PR[99] viene salvato l'offset in z del pezzo
PR[99]=LPOS-LPOS
PR[99,3]=25

! Posizioni di place a seconda della zona
PR[8]=PR[6]
PR[9]=PR[7]

! Quota z che discrimina tra le due zone dello spazio di lavoro
R[12]=290

! Open gripper
CALL GRIP_OPEN

! Home position
J P[3] 100% FINE

R[7]=0
RUN PRG_VISION
WAIT R[7]=1

! Non ci sono pezzi identificati
IF R[24]=0,JMP LBL[3]

LBL[2]
! Si ottiene la posizione del primo oggetto identificato
VISION GET_OFFSET 'REG_V_3DA3' VR[1] JMP LBL[3]

! Posizione sopra al piano di lavoro
J P[1] 100% FINE
```

2 Osservando il programma, si può notare che si sfruttano due quote diverse per determinare in quale zona si trova l'operatore ed in quale zona si trova il pezzo da prelevare, sebbene l'area di lavoro sia la stessa. Il motivo di ciò è che per il processo di visione 2D e per il processo di visione 3D vengono utilizzate due calibrazioni diverse, riferite a due *user frame* diversi.

```
51
52 ! Si salva in R[19] la quota X dell'oggetto identificato
53 R[19]=VR[1].MES[2]
54
55 ! Riconoscimento della zona in cui si trova l'oggetto
56 IF (R[19]>=R[12]) THEN
57
58 ! Il pezzo si trova in zona 1
59 R[13]=1
60 R[14]=0
61 ! Si accende il led bianco del robot
62 RO[1]=ON
63 RO[2]=OFF
64
65 ELSE
66
67 ! Il pezzo   in zona 2
68 R[13]=0
69 R[14]=1
70 ! Si accende il led rosso del robot
71 RO[1]=OFF
72 RO[2]=ON
73
74 ENDIF
75
76 ! Riconoscimento se l'oggetto ha dimensioni troppo grandi per essere prelevato
77 R[21]=VR[1].MES[3]
78
79 IF (R[21]>300) THEN
80 CALL OBG_BIG
81 J P[3] 100% FINE
82 R[7]=0
83 RUN PRG_VISION
84 WAIT R[7]=1
85 IF R[24]=0,JMP LBL[3]
86 JMP LBL[2]
87 ENDIF
88
89 ! Approach posizione di pick
90 L P[2] 100mm/sec CNT100 Tool_Offset,PR[100] VOFFSET,VR[1]
91 ! Posizione pick
92 L P[2] 100mm/sec FINE Tool_Offset,PR[99] VOFFSET,VR[1]
93 ! Chiusura pinze
94 CALL GRIP_CLOSE
95 ! Depart posizione di pick
96 L P[2] 100mm/sec CNT100 Tool_Offset,PR[100] VOFFSET,VR[1]
97
98 ! Posizione intermedia
99 J P[4] 100% CNT100
100
101 R[7]=0
102 RUN PRG_VISION
103
104 IF (R[13]=1) THEN
105
106 ! Approach posizione place
107 J PR[8] 100% CNT100 Tool_Offset,PR[100]
108 ! Posizione di place
109 L PR[8] 350mm/sec FINE Tool_Offset,PR[99]
110 ! Apertura pinze
111 CALL GRIP_OPEN
112 ! Depart posizione place
113 L PR[8] 350mm/sec CNT100 Tool_Offset,PR[100]
114 ! Incremento X della posizione di place
115 PR[8,1]=PR[8,1]-60
116
```

```
ELSE

! Approach posizione place
J PR[9] 100% CNT100 Tool_Offset,PR[100]
! Posizione di place
L PR[9] 350mm/sec FINE Tool_Offset,PR[99]
! Apertura pinze
CALL GRIP_OPEN
! Depart posizione place
L PR[9] 350mm/sec CNT100 Tool_Offset,PR[100]
! Incremento X della posiione di place
PR[9,1]=PR[9,1]-60

ENDIF

! Si aspetta che la scansione sia stata eseguita
WAIT R[7]=1
J P[3] 100% CNT100
JMP LBL[2]

! Fine programma
LBL[3]
R[20]=1
J P[3] 100% FINE
```

4.6.3.1 *Programma per l'acquisizione della mappa 3D*

Di seguito è riportato il programma per l'acquisizione della mappa 3D, richiamato appunto ogni volta che si vuole acquisire la mappa 3D. Questo programma è del tutto identico a quello precedentemente illustrato.

```
! Programma per visione multitask

! Acquisizione mappa 3D
CALL BINPICK_ACQUIRE3DMAP("3D Area Sensor"='CAL_NEW')

! Chiamata al processo di visione
VISION RUN_FIND 'REG_V_3DA3'

! Nel registro R[24] viene messo il numero di pezzi trovati
VISION GET_NFOUND 'REG_V_3DA3' R[24]

R[7]=1
```

4.6.3.2 *Programma che viene richiamato se l'oggetto è troppo grande*

Il riconoscimento della grandezza di un oggetto viene eseguito secondo la logica esposta in sezione 4.7. Questo programma si occupa semplicemente di illuminare il pulsante retroilluminato, attendere la pressione del pulsante da parte dell'operatore e poi spegnere il pulsante retroilluminato. Come già evidenziato, questa operazione è resa più rapida dall'utilizzo di un robot collaborativo in quanto non è necessario il superamento di barriere.

```
! Programma per la gestione di un oggetto troppo grande

! Si accende il pulsante retroilluminato
RO[3]=ON
! Si attende la pressione del pulsante
```

```
6 WAIT (RI[1]=ON)
7 ! Si spegne il pulsante retroilluminato
8 RO[3]=OFF
```

4.6.4 *Tabella con il contenuto dei registri*

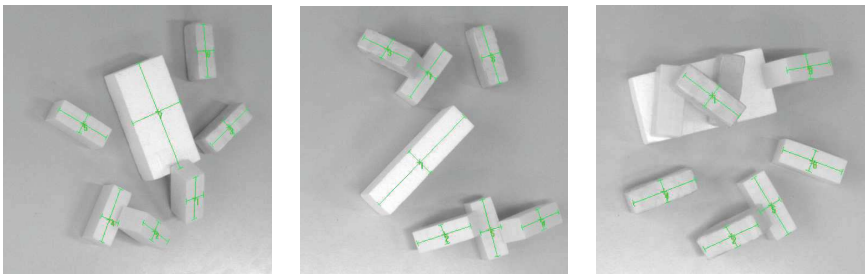
Com'è noto, il linguaggio di programmazione Fanuc non prevede l'utilizzo di variabili, bensì il solo utilizzo di registri: questo può rende difficile la lettura dei programmi. Per facilitarne la comprensione, in tabella 4.1 è riportato l'elenco dei registri utilizzati ed il relativo contenuto. Quelli indicati con la lettera R sono i registri dati, quelli indicati con la lettera P sono i dati di posizione, quelli indicati con PR sono i registri di posizione ed infine quelli indicati con VR sono i registri di visione.

4.7 RICONOSCIMENTO DI OGGETTI CHE IL ROBOT NON RIESCE A PRENDERE

Nel programma illustrato precedentemente, che è solo un esempio di applicazione di bin picking collaborativo, è stata prevista la possibilità che il sistema di visione riconosca che un pezzo è troppo grande per essere prelevato e pertanto richieda l'intervento di un operatore. Questo potrebbe accadere ad esempio perché dalle lavorazioni a valle arrivano dei pezzi destinati ad un altra cella. Come già precedentemente esposto, la pinza Co-act montata sul robot permette di afferrare oggetti di dimensioni relativamente ridotte. Tra gli oggetti presenti in laboratorio, è stato scelto un parallelepipedo di dimensioni maggiori rispetto ai parallelepipedi utilizzati nelle prove sul bin picking illustrate al capitolo 3. Esempi della presenza di un parallelepipedo di dimensioni maggiori sono riportati in figura 4.8. Ciò che si nota in tali figure è che comunque il sistema di visione è in grado di riconoscere tali parallelepipedi al pari di quelli più piccoli. Se il parallelepipedo di dimensioni maggiori si trova sotto ad altri oggetti, esso sarà riconosciuto solo dopo che gli oggetti soprastanti vengono rimossi. In generale, ci sono diversi modi con i quali distinguere un pezzo di dimensioni maggiori dagli altri di dimensioni minori. Come visto precedentemente, per il riconoscimento dei pezzi tramite la visione 3D si utilizza il *3D Blob Locator Tool*, a cui è collegato il *Measurement Output Tool*, il quale consente di trasferire nei registri di visione le informazioni selezionate dall'utente. Il parallelepipedo di dimensioni maggiori può tipicamente trovarsi orientato come in figura 4.8a e 4.8b. Un modo semplice per distinguere gli eventuali pezzi di dimensione maggiore è andare a guardare il numero di *blob points*, oppure i valori di *3D blob lenght* o *3D blob width*. In questo caso è stato scelto di andare a guardare il numero di *blob points*, che per i parallelepipedi di dimensioni minori oscilla da un valore massimo

Tabella 4.1: Contenuto dei registri

R[7]	Registro utilizzato per sincronizzare la scansione 3D e l'avanzamento del programma di bin picking
R[11]	Numero di operatori riconosciuti dalla telecamera 1
R[12]	Quota x che discrimina se il pezzo, riconosciuto dal processo di visione 3D, si trova nella zona 1 oppure nella zona 2 dello spazio di lavoro
R[13]	Se R[13]=1 il pezzo da prelevare si trova in zona 1
R[14]	Se R[14]=1 il pezzo da prelevare si trova in zona 2
R[15]	Se R[15]=1 l'operatore si trova in zona 1
R[16]	Se R[16]=1 l'operatore si trova in zona 2
R[17]	Quota y che discrimina se l'operatore, riconosciuto dalla telecamera 1, si trova nella zona 1 oppure nella zona 2 dello spazio di lavoro
R[18]	Quota y alla quale viene riconosciuto l'operatore
R[19]	Quota x alla quale viene riconosciuto l'oggetto da prelevare
R[20]	Sincronizzare il programma di visione 2D con quello di visione 3D: quando non ci sono più pezzi da prelevare viene posto R[20]=1 ed il programma di visione 2D termina
R[21]	Numero di *blob points* dell'oggetto che è stato riconosciuto
R[24]	Numero di oggetti identificati dal processo di visione 3D
R[31]	Numero di operatori riconosciuti dal processo di visione 2D
P[1]	Posizione sopra al piano di lavoro
P[2]	Contiene $(0, 0, 0)$ in XYZ e $(180, 0, 0)$ in WPR
P[3]	Posizione di home
P[4]	Posizione intermedia (fuori dal campo inquadrato durante la scansione 3D)
PR[6]	Posizione iniziale di place per la zona 1
PR[7]	Posizione iniziale di place per la zona 2
PR[8]	Posizione di place per la zona 1 (viene incrementata con l'avanzare del programma)
PR[9]	Posizione di place per la zona 2 (viene incrementata con l'avanzare del programma)
PR[10]	Posizione dell'operatore
VR[1]	Contiene le informazioni relative all'oggetto riconosciuto dal processo di visione 3D
VR[2]	Contiene le informazioni relative all'oggetto riconosciuto dal processo di visione 2D



(a) (b) (c)

Figura 4.8: Esempi di un parallelepipedo di dimensioni maggiori rispetto gli altri

di 178 ad un valore minimo di 144, mentre per il parallelepipedo di dimensioni maggiori è sempre maggiore di 300. Se il robot si accorge che uno dei pezzi è troppo grande, lo si segnala tramite il led RO[3] e aspetta che il pulsante RI[1] venga premuto. L'operatore a questo punto dovrà provvedere a togliere il pezzo dall'area di lavoro del robot, eventualmente depositare nuovi oggetti, e successivamente premere il pulsante RI[1] presente alla base del robot. Il robot procede a eseguire una nuova scansione 3D e avanza nel programma.

4.8 CALCOLO DELLA DISTANZA TRA ROBOT ED OPERATORE

La presenza di un target geometrico nel bracciale indossato dal robot permette di monitorarne in maniera accurata la posizione. Come accennato precedentemente, è possibile monitorare continuamente la posizione reciproca tra operatore e robot al fine di evitare possibili collisioni e quindi eventuali danni e fermi macchina, nonostante il robot sia collaborativo. Il fatto che il manipolatore preso in considerazione sia collaborativo garantisce che se dovesse avvenire una collisione, questa non si traduce in danni all'operatore: è comunque desiderabile che tali collisioni non avvengano affatto, e l'utilizzo dell'algoritmo sviluppato può essere molto utile in questo senso. La posizione dell'operatore si monitora come illustrato in precedenza, osservando la posizione del target. In questo caso è importante avere ben chiaro a quale sistema di riferimento sono riferite le quote (x, y). La posizione del robot si può invece ricavare tramite le variabili di sistema: in particolare, le variabili di sistema $MCH_POS_X, $MCH_POS_Y, $MCH_POS_Z, $MCH_POS_W, $MCH_POS_P, $MCH_POS_R[3] contengono, istante per istante, posizione ed orientamento del robot. Per

3 Queste variabili di sistema di default sono disattivate: per attivarle è necessario andare su MENU, NEXT, SYSTEM e poi ancora VARIABLES: compare un menù dove si trovano tutte le variabili di sistema del robot. Tra queste, è necessario scorrere finché alla riga 727 non si trova la variabile di sistema $SCR_GRP: premendo su questa variabile di sistema ne compaiono delle altre. È necessario trovare $M_POS_ENB (machine position enable) e metterla a TRUE.

monitorare la posizione del robot si può creare un programma come quello che segue.

```
! Monitoraggio della posizione del robot
R[1]=($SCR_GRP[1].$MCH_POS_X)
R[2]=($SCR_GRP[1].$MCH_POS_Y)
R[3]=($SCR_GRP[1].$MCH_POS_Z)
R[4]=($SCR_GRP[1].$MCH_POS_W)
R[5]=($SCR_GRP[1].$MCH_POS_P)
R[6]=($SCR_GRP[1].$MCH_POS_R)
```

Questo programma può funzionare in logica background (ovvero continua ad iterare in un loop infinito) e quindi può aggiornare continuamente il contenuto dei registri, ogni 0,8 ms (è il tempo ciclo dei programmi Fanuc). La posizione del robot, che deve essere salvata in opportuni registri, è riferita alla terna *world frame* (posizionata sulla base del robot), mentre la posizione dell'operatore può essere fornita rispetto alla terna *world frame* o rispetto a qualsiasi terna *user frame* precedentemente definita a seconda di come viene eseguita la calibrazione. Per confrontare le quote relative al robot e le quote relative all'operatore può quindi essere necessario eseguire una calibrazione apposita oppure eseguire una trasformazione di coordinate mediante delle matrici di trasformazione. Va anche tenuto conto che il target si trova nel polso dell'operatore, mentre la collisione avviene tipicamente con le dita: andrà quindi tenuto conto di un certo offset tra la posizione del target e quella delle dita delle mani dell'operatore. Si può poi quindi scrivere una serie di righe di codice, successive a quelle riportate precedentemente, che calcolino la distanza tra robot ed operatore e va prevista una logica da implementare nel caso in cui operatore e robot si avvicinino troppo. Una logica che si potrebbe implementare, suggerita da [6], è la seguente:

- Se il sistema si accorge della presenza di un operatore a meno di un metro dall'organo terminale del robot, il robot rallenta ad una velocità di 250 mm/s, in accordo con le regole EU. Nel caso della cella robotizzata analizzata, se l'operatore rientra nel campo visivo inquadrato dal sistema di visione 3D, esso si trova già ad una distanza inferiore ad un metro dell'organo terminale del robot.
- Se il sistema si accorge della presenza di un operatore a meno di mezzo metro dall'organo terminale del robot, il robot rallenta ad una velocità pari al 50% rispetto alla velocità nominale.
- Se il sistema si accorge della presenza di un operatore a meno di 20 cm, il robot si ferma. In alternativa, potrebbe anche essere possibile far cambiare traiettoria al robot, ovvero farlo andare in direzione opposta rispetto a dove si trova e sta lavorando l'operatore.

Questa soluzione non è tuttavia stata implementata in quanto l'aggiornamento delle variabili di sistema quali $MCH_POS_X non av-

viene quando il robot si trova in modalità T1 o T2, ma solo quando il robot si trova in modalità AUTO. Durante la stesura di questa tesi, è stato necessario lavorare in modalità T1 o T2 in quanto non essendo il robot ancora installato e fissato al pavimento erano disattivati i sensori di sicurezza alla base del manipolatore che permettono di lavorare in modalità AUTO. Questo potrebbe essere tuttavia uno spunto per i lavori futuri.

4.9 CONSIDERAZIONI E LIMITI DELLA SOLUZIONE PROPOSTA

Grazie allo studio ed ai test sperimentali condotti è stato possibile concludere che si riesce a sfruttare efficacemente una delle due telecamere facenti parte di un sistema di visione 3D a luce strutturata per il riconoscimento di operatori all'interno dell'area di lavoro del robot. In particolare, è possibile affermare che:

- Il riconoscimento degli operatori tramite la telecamera facente parte del sistema di visione 3D avviene in tempi brevi, compatibili con la sicurezza degli operatori e con le tipiche velocità alle quali lavorano i robot collaborativi (inferiori a 250 mm/s). Potrebbe comunque essere interessante confrontare le prestazioni ottenute sfruttando la telecamera del sensore di visione 3D con quelle che si otterrebbero con l'utilizzo di una telecamera esterna. Intuitivamente, ci si aspetta che l'utilizzo di una telecamera esterna velocizzi ulteriormente il processo di riconoscimento dell'operatore e quindi lo renda ancora più efficiente;
- Il task di bin picking può essere reso collaborativo e gli algoritmi sviluppati possono essere sfruttati in applicazioni diverse, che permettono di superare alcuni dei limiti del bin picking tradizionale: in particolare il bin picking collaborativo permette di velocizzare tutte le attività che richiedono l'intervento di un operatore;
- La soluzione proposta ed implementata durante questo lavoro di tesi si basa sull'utilizzo da parte dell'operatore di un bracciale con target, come illustrato nelle sezioni precedenti. In lavori futuri sarebbe opportuno investigare la possibilità di riconoscere l'operatore anche senza che lo stesso indossi un bracciale o un target che lo contraddistingua.

CONCLUSIONI

Il bin picking costituisce un filone di ricerca relativamente nuovo e per questo, oltre che per la complessità stessa del task, la presa da cassone è un problema solo parzialmente risolto. Durante questo lavoro di tesi è stata implementata una cella per bin picking collaborativo, composta da un robot antropomorfo collaborativo e da un sistema di visione 3D a luce strutturata. La cella robotizzata è stata collaudata e si sono potute verificare le potenzialità ed il limiti della stessa, in particolare in relazione al tipo di oggetti che il sistema di visione è in grado di riconoscere e successivamente di prelevare tramite la pinza elettrica collaborativa a disposizione. Ciò che è emerso dalle prove sperimentali è che il sistema di visione è effettivamente molto robusto ed accurato, in grado di riconoscere efficacemente anche pezzi dalle geometrie diverse, mentre il limite dell'applicazione proposta è costituito dalla pinza ed in particolare dalle griffe. Le griffe utilizzate permettono la presa solo di un numero ridotto di oggetti e non sono ideali per la presa di oggetti alla rinfusa da un contenitore. Per quest'applicazione sarebbero invece maggiormente indicate delle ventose. Grazie alle analisi ed ai test sperimentali condotti, è possibile anche concludere che effettivamente si riesce a sfruttare efficacemente una delle due telecamere facenti parte del sistema di visione 3D per il riconoscimento degli operatori all'interno dell'area di lavoro del robot. Il riconoscimento degli operatori avviene in tempi brevi, compatibili con la sicurezza degli operatori e con le tipiche velocità alle quali funzionano i robot collaborativi. Gli algoritmi di bin picking collaborativo possono dunque essere sfruttati in applicazioni diverse e permettono di superare alcuni dei limiti del bin picking tradizionale. Il bin picking collaborativo è quindi una soluzione realizzabile ed ha caratteristiche innovative.

APPENDIX

A

LINGUAGGIO DI PROGRAMMAZIONE FANUC LS

Di seguito sono riportate alcune nozioni di base per la programmazione di robot Fanuc tramite linguaggio di programmazione Fanuc LS. Questo linguaggio è il linguaggio di programmazione che si utilizza quando si programmano i manipolatori Fanuc tramite teaching pendant oppure quando si programmano da PC tramite l'interfaccia iRProgrammer. Per ulteriori dettagli, si rimanda alla manualistica Fanuc.

A.1 INFORMAZIONI DI DETTAGLIO DEL PROGRAMMA

In generale, i dati di dettaglio di un programma ne contengono il nome e le proprietà. Quando viene creato un programma, o si selezionano i dettagli di un programma esistente (ai quali si può accedere tramite il tasto F2 DETAIL), si possono distinguere i seguenti campi:

- *Program name:* è il nome del programma.La lunghezza massima del nome del programma è di 36 caratteri. Il nome del programma non può iniziare con un numero. È ammesso il simbolo di sottolineatura, ma non i simboli chiocciola (@) ed asterisco (*). Nei manuali Fanuc è inoltre possibile visionare i nomi non utilizzabili per un programma.

- *Sub type:* indica il sottotipo del programma (e può anche non essere specificato). Sono disponibili i seguenti sottotipi: job (JB), process (PR), macro (MR), condizioni e collection (CO). Degno di nota è il sottotipo macro, che indica un tipo di programma che può essere eseguito anche a seguito di pressione di tasti, passaggio a ON di ingressi digitali o selezione da menù delle funzioni manuali. Per l'apertura e la chiusura della pinza si definiscono tipicamente degli appositi programmi, il cui sottotipo è macro. Si rimanda alla sezione A.9 per le macro di apertura e chiusura della pinza.

- *Comment:* ogni programma può avere un commento, formato da un massimo di 16 caratteri.

- *Group mask:* è la maschera di selezione del gruppo assi, che consente di associare uno o più gruppi assi (se presenti nella configurazione del robot) ad un programma. Un gruppo di assi è l'insieme di motori che azionano il braccio del robot, eventuali tavole, posizionatori o altri assi ausiliari. Se il sistema è

composto da un solo gruppo, il numero di gruppo di assi preimpostato è 1 (1,*,*,*,*,*,*,*). Per un programma senza gruppi di movimento (ovvero un programma che non comporta movimenti del robot), questo elemento può essere specificato come (*,*,*,*,*,*,*,*).Quando si eseguono programmi multitasking, bisogna porre particolare attenzione al contenuto di questo campo. Per ulteriori dettagli sulla maschera di selezione del gruppo assi e le operazioni multitasking, si rimanda alla sezione A.10.

- *Write protect:* è la protezione da scrittura. Essa specifica se il programma può essere modificato oppure no. Quando è abilitata la protezione da scrittura (nel campo *write protect* è scritto ON), non è possibile modificare il programma, né il suo nome, il commento o il sottotipo. Se invece nel campo *write protect* è scritto OFF, allora il programma può essere modificato. La protezione è preimpostata ad OFF al momento della creazione del programma.

- *Ignore pause:* quando questo campo risulta ON, l'esecuzione del programma continua anche in occasione di un errore, di un arresto in emergenza o di un HOLD. L'*ignore pause* è effettivo solo nei programmi che non hanno gruppi di movimento.

- *Stack size:* definisce la quantità di memoria da utilizzare quando viene eseguita una chiamata ad un programma.

- *Collection:* se il sottotipo del programma è collection, viene visualizzata la schemata *collection editor* per la registrazione dei programmi figli di questo programma. Se invece il sottotipo non è collection, viene visualizzata la lista delle collezioni per registrare il programma come figlio di una collezione.

- *Enable singularity avoidance:* quando questo campo è impostato a TRUE permette di evitare la singolarità di polso durante i movimenti lieari.

A.2 COMANDI DI MOVIMENTO

A.2.1 *Formato del movimento*

Con il formato del movimento si specifica quale tipo di interpolazione sarà utilizzata dal robot per arrivare in posizione. Nel linguaggio di programmazione Fanuc sono disponibili quattro opzioni: joint, lineare, circolare e circle arc. La prima opzione non prevede controllo sulla traiettoria e sull'orientamento dell'utensile, mentre le altre tre prevendono il controllo su traiettoria e orientamento dell'utensile. Le due interpolazioni più utilizzate sono l'interpolazione di tipo joint e di tipo lineare, esposte in seguito.

A.2.1.1 *Interpolazione joint (J)*

L'interpolazione joint è il modo base per far muovere il robot in una posizione specifica. Il robot accelera su tutti gli assi, si sposta ad una certa velocità, decelera e si ferma con tutti i motori allo stesso momento. La traiettoria descritta dall'utensile in questo caso non è lineare, la velocità è espressa in percentuale rispetto a quella massima calcolata per ciascun asse e l'orientamento dell'utensile non è controllato. Di seguito è riportato un esempio di movimenti eseguiti con interpolazione joint.

```
! Movimenti con interpolazione joint
J P[1] 100% FINE
J P[2] 100% FINE
```

Nel linguaggio di programmazione Fanuc, per specificare l'interpolazione joint, va indicata la J prima dell'istruzione di movimento.

A.2.1.2 *Interpolazione lineare (L)*

Un movimento lineare consente il controllo della traiettoria che il punto di centro utensile (TCP) descrive per passare dal punto iniziale al punto finale: il punto di centro utensile si muove ungo una linea retta. In questo caso, la velocità è specificata in mm/sec, cm/min, inch/min. I movimenti lineari sono utili per i movimenti di avvicinamento ed allontanamento dal punto di pick o di place dell'oggetto, così come per avere la certezza che non si vadano a colpure degli ostacoli durante il movimento. Di seguito è riportato un esempio di un movimento painificato nello spazio dei giunti, seguito da un movimento lineare:

```
! Movimenti con interpolazione joint ed interpolazione lineare
J P[1] 100% FINE
L P[2] 500mm/sec FINE
```

Nel linguaggio di programmazione Fanuc, per specificare l'interpolazione lineare, va indicata la L prima dell'istruzione di movimento. I movimenti specificati con la lettera L consentono anche di eseguire un'operazione di rotazione: in questo caso le due posizioni specificate devono avere stesse coordinate ma diverso orientamento e la velocità di avanzamento si specifica in gradi/min. Di seguito è riportato un esempio di operazione di rotazione:

```
! Movimento rotatorio
J P[1] 100% FINE
L P[2] 30deg/sec FINE
```

A.2.2 *Dati di posizione*

I dati di posizione comprendono le posizioni e l'orientamento del robot. Essi vengono scritti nel programma quando si registra una posizione. I dati di posizione possono essere rappresentati in coordinate joint oppure in coordinate cartesiane.

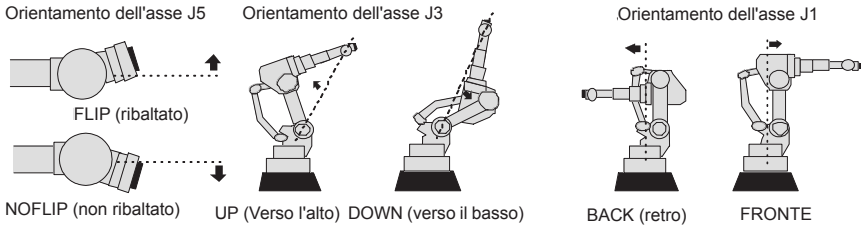


Figura A.1: Possibili configurazioni degli assi J5, J3 e J1

- Se i dati di posizione sono espressi in coordinate joint essi sono definiti come scostamenti angolari rispetto al sistema di coordinate joint in corrispondenza di ciascuna articolazione (J1, J2, J3, J4, J5, J6);
- Se i dati di posizione sono espressi in coordinate cartesiane sono definiti da quattro elementi:
  - *UF*: numero del sistema di coordinate utente (*user frame*). Il sistema di coordinate utente corrisponde generalmente all'angolo del piano sul quale si lavora.
  - *UT*: numero del sistema di coordinate utensile (dipende appunto dall'utensile che è montato sul sul robot).
  - Posizione (x, y, z) ed orientamento (w, p, r). La posizione (x, y, z) rappresenta il punto nello spazio tridimensionale per il TCP nel sistema di coordinate cartesiane. L'orientamento (w, p, r) rappresenta gli angoli di rotazione attorno agli assi X, Y e Z del sistema di coordinate cartensiane.
  - La configurazione, che rappresenta l'orientamento nello spazio del robot. Sono possibili svariate configurazioni che possono soddisfare la condizione descritta dalle coordinate cartesiane. La configurazione è indicata tramite tre lettere, che indicano la configurazione rispettivamente degli assi J5, J3 e J1, come illustrato in figura A.1. Ad esempio, se la configurazione è NUT significa NOFLIP, UP, TOP (fronte).

I dati di posizione possono trovarsi in variabili di posizione, indicate con la lettera P, oppure in registri di posizione, indicati con la lettera PR. La differenza tra le variabili di posizione P ed i registri di posizione PR è che nel caso dei registri si può accedere anche ad uno solo dei campi di posizione, ad esempio solo il valore della x. Inoltre, se si copia ed incolla un programma, le variabili di posizione perdono il loro contenuto, vale a dire che hanno valenza solo all'interno del programma che le ha create.

A.2.3 *Velocità*

La velocità del movimento può essere specificata in maniera diretta oppure in maniera indiretta, inserendo il valore della velocità in un registro. A seconda dell'unità che si utilizza, esistono dei valori limite da rispettare, che si ritrovano nei manuali Fanuc.

A.2.4 *Tipo di posizionamento*

il tipo di posizionamento definisce il metodo utilizzato per terminare un'istruzione di movimento. Si può scegliere tra due tipi di posizionamento: FINE e CNT. Quando si specifica un posizionamento FINE, il robot si arresta alla posizione di destinazione prima di proseguire verso il punto successivo. Quando invece si specifica un posizionamento CNT, il robot si avvicina alla posizione di destinazione, senza fermarsi, e prosegue verso il punto successivo. Inserendo un valore tra 0 e 100 è possibile regolare l'avvicinamento al punto di destinazione: se si immette il valore 0, il robot raggiunge il punto più vicino alla posizione memorizzata senza fermarsi e poi prosegue per il nodo successivo. Se invece si immette il valore 100 il robot non decelera e passa relativamente lontano dalla posizione di destinazione, per poi proseguire immediatamente a quella successiva. A questo aspetto bisogna porre particolare attenzione nel caso siano presenti degli ostacoli nella traiettoria. Di seguito sono riportati due esempi di posizionamento FINE e CNT.

```
1 ! Posizionamento FINE e CNT
2 J P[1] 50% FINE
3 J P[2] 50% CNT50
```

A.3 REGISTRI

Il linguaggio di programmazione Fanuc non prevede l'utilizzo di variabili, bensì prevede solamente l'utilizzo di registri, il che può risultare scomodo. I tre tipi principali di registri che vengono utilizzati sono registri numerici R, registri di posizione PR e registri di visione VR. Esistono anche i registri di pallettizzazione PL ed i registri stringa SR.

A.3.1 *Registri numerici R*

Un registro R è una variabile numerica che può contenere un valore intero o reale (frazione decimale). La quantità standard di registri numerici presenti nel sistema è di 200, ma può essere aumentata fino a 999. Sui registri di posizione possono essere eseguite una serie di operazioni quali somme, sottrazioni, divisioni, modulo, ecc.

```
1 ! Esempio di operazioni sui registri
2 R[1]=RI[3]
```

```
R[R[4]]=AI[R[1]]
R[2]=DI[4]
R[3]=R[4] MOD R[5]
```

A.3.2 *Registri posizione PR*

Un registro di posizione PR è una variabile che può contenere dati di tipo posizione (x, y, z, w, p, r). La quantità standard di registri posizione presenti nel sistema è 100, ma può essere aumentata fino a 200. I registri di posizione sono più potenti rispetto alle variabili di posizione P in quanto tramite i registri di posizione è possibile anche accedere ad uno solo dei campi di posizione. Si consideri l'esempio che segue:

```
! Registri di posizione PR
J P[1] 30% FINE
L PR[1] 300m/sec FINE
R[1]=PR[1,1]
R[2]=PR[1.2]
```

Tramite le ultime due righe, nel registro R[1] viene inserito il valore della x delle coordinate della posizione memorizzata nel registro di posizione PR[1], mentre nel registro R[2] viene inserito il valore della z delle coordinate della posizione memorizzata nel registro di posizione PR[1]. Di default i registri di posizione sono sbloccati, quindi sono modificabili dall'utente. Tuttavia, quando si utilizzano registri PR in istruzioni di movimento è bene che siano bloccati poiché quando si eseguono istruzioni su registri posizione sbloccati potrebbero verificarsi rallentamenti nel ciclo automatico. Di seguito è riportato un esempio di utilizzo delle istruzioni di blocco/sblocco dei registri posizione PR.

```
! Istruzioni di blocco/sblocco dei registri di posizione PR
J P[1] 30% FINE
PR[1]=PR[2]
PR[2]=PR[3]
LOCK PREG
L P[2] 100mm/sec CNT100
L P[3] 100mm/sec CNT100
L PR[1] 100mm/sec CNT100
L P[4] 100mm/sec CNT100 OFFSET, PR[2]
L P[5] 100mm/sec FINE
UNLOCK PREG
```

Quando viene eseguita la linea 5 di questo programma, i registri di posizione PR sono bloccati. Vengono sbloccati nel momento in cui è eseguita la linea 11. Grazie a ciò, le istruzioni di movimento nelle linee di programma 8 e 9, eseguite con i PR bloccati, saranno soggette a precalcolo (il robot legge in anticipo le posizioni che dovrà raggiungere, questo velocizza il ciclo automatico).

A.3.3 *Registri di visione VR*

I registri di visione sono speciali registri utilizzati per memorizzare i dati provenineti dal sistema di visione Fanuc iRVision. Un registro di visione contiene la posizione e l'offset dell'oggetto individuato (posizione ed offset dipendono dai sistemi di riferimento impostati). Inoltre, ogni registro contiene anche delle misurazioni, alle quali si accede tramite VR[].MES[], il cui contenuto dipende da quanto impostato nel *Measurement Output Tool*. Un esempio di utilizzo dei registri di visione è riportato di seguito.

```
! Istruzioni sui registri di visione
VISION GET_OFFSET 'PRG_VISION' VR[1] JMP LBL[3]
R[1]=VR[1].MES[1]
R[2]=VR[1].MES[2]
R[3]=VR[1].MES[3]
L P[1] 100mm/sec CNT100 Tool_Offset, PR[100] VOFFSET, VR[1]
PR[1]=VR[1].FOUND_POS[1]
```

A.4 ISTRUZIONI DI DIRAMAZIONE

Un'istruzione condizionale consente di effettuare una diramazione dell'esecuzione di un programma da una linea ad un'altra. Il linguaggio di programmazione Fanuc LS suppporta quattro tipi di istruzioni di questo genere: istruzione etichetta (LBL), istruzione di fine programma (END), istruzione di salto non condizionato (JMP LBL) ed istruzioni di salto condizionato (IF/SELECT).

A.4.1 *Istruzione etichetta (LBL)*

L'istruzione LBL[] è utilizzata per definire la destinazione di una diramazione dell'esecuzione all'interno di un programma. Di fatto, un'etichetta identifica un punto del programma al quale si vorrà eseguire un salto al verificarsi di determinate condizioni.

A.4.2 *Istruzione di fine programma (END)*

L'istruzione END indica la fine dell'esecuzione di un programma. Questa istruzione appare sempre in automatico alla fine di ogni programma.

A.4.3 *Istruzioni di salto non condizionato*

Un'istruzione di salto incondizionato dirama il flusso di esecuzione del programma ad un punto identificato da una certa etichetta. Questa istruzione è generalmente poco utilizzata in quanto non ha senso eseguire sempre un salto, se non in fase di debug del programma.

A.4.4 *Istruzioni di salto condizionato (IF/SELECT)*

Molto più utili sono le istruzioni di salto condizionato, che prevedono di diramare il flusso di esecuzione all'interno di un programma solo se vengono verificate alcune condizioni. Quando si scrivono istruzioni di salto condizionato, particolare attenzione va riservata alla sintassi delle istruzioni. Una serie di esempi di istruzioni di salto condizionato con istruzione IF sono riportati di seguito.

```
! Istruzioni di salto condizionato (IF)
IF R[1]>10, JMP LBL[1]
IF AO[2] <= 5, JMP LBL[5]
IF DI[1]=ON, JMP LBL[2]
IF DI[1]=ON AND DI[2]=OFF, JMP LBL[4]
```

Oltre all'istruzione IF, è anche possibile utilizzare l'istruzione SELECT, che prevede una serie di confronti e di salti in cui siano verificate le condizioni. Un esempio di utilizzo dell'istruzione SELECT è riportato di seguito.

```
! Istruzioni di salto condizionato (SLECT)
SELECT R[1]=1, JMP LBL[1]
       =2, JMP LBL[2]
       =3, JMP LBL[3]
       =4, JMP LBL[4]
       ELSE, JMP LBL[5]
```

Può essere che la condizione sia verificata per più righe: in questo caso viene eseguito solo il primo tra i possibili salti.

A.5 ISTRUZIONE IF THEN/ELSE/ENDIF

Le istruzioni IF THEN/ELSE/ENDIF sono istruzioni di diramazione condizionale: quando la condizione viene soddisfatta, vengono eseguite le linee del programma comprese tra IF e THEN mentre quando la condizione non viene soddisfatta, vengono eseguire le linee del programma comprese tra ELSE ed ENDIF. Un esempio di utilizzo dell'impiego IF THEN/ELSE/ENDIF è riportato di seguito.

```
! Istruzione IF THEN/ELSE/ENDIF
IF (DI[1]=ON AND DI[3]=ON) THEN
L P[1] 100mm/sec CNT100
L P[2] 100mm/sec CNT100
L P[3] 100mm/sec FINE
ELSE
L P[4] 100mm/sec CNT100
L P[5] 100mm/sec CNT100
L P[6] 100mm/sec FINE
ENDIF
```

A.6 ISTRUZIONI DI ATTESA

L'istruzione WAIT è utilizzabile per interrompere l'esecuzione di un programma per un certo tempo oppure finché non si verifica una determinata condizione. Nel caso in cui si attenda il verificarsi di una

determinata condizione è possibile inserire un'istruzione di timeout, ovvero se la condizione non si verifica nonostante l'attesa di un certo tempo prestabilito, viene eseguito il salto ad una certa etichetta. Questo serve per far sì che il programma non si blocchi in attesa del verificarsi di una condizione. Di seguito sono riportati degli esempi di impiego dell'istruzione di attesa.

```
! Esempi di utilizzo dell'istruzione WAIT
WAIT 10.5sec
WAIT R[1]>1
WAIT R[2]<>1, TIMEOUT LBL[1]
WAIT DI[2]<>OFF, TIMEOUT LBL[2]
```

A.7 ISTRUZIONI FOR/ENDFOR

L'istruzione FOR/ENDFOR serve per ripetere un certo numero di volte programmato, l'esecuzione delle linee di programma comprese tra FOR ed ENDFOR. Per contare il numero di ripetizioni, si utilizza un registro R e viene specificato il suo valore iniziale ed il suo valore finale. Il valore del registro può essere sia incrementato che decrementato, a seconda di come viene impostato il ciclo. È bene sottolineare che quando è in esecuzione un ciclo FOR/ENDFOR il sistema attiva un ritardo interno e non è quindi necessario inserire un'istruzione WAIT tra le istruzioni FOR e ENDFOR. È inoltre possibile creare dei cicli FOR annidati ed il numero massimo di livelli di annidamento è 10. In seguito è riportato un esempio di istruzioni FOR/ENDFOR.

```
! Esempi di utilizzo dell'istruzione FOR/ENDFOR
FOR R[1]= TO 5
L P[1] 100mm/sec CNT100
L P[2] 100mm/sec FINE
L P[3] 100mm/sec CNT100
ENDFOR
FOR R[2]=5 DOWNTO 1
L P[4] 100mm/sec FINE
L P[5] 100mm/sec CNT100
ENDFOR
```

A.8 ISTRUZIONI I/O

Le istruzione I/O (segnali di input e output) sono utilizzate per cambiare lo stato di un segnale digitale in uscita o leggere lo stato di un segnale in ingresso al controllore del robot. In particolare, possono essere effettuate istruzioni su I/O digitali di sistema, I/O digitali del robot, I/O analogici o istruzioni su gruppo di ingresso o di uscita.

A.8.1 *Istruzioni su I/O digitali*

I segnali digitali di sistema di ingresso e di uscita sono indicati rispettivamente dalle sigle DI e DO. Sugli I/O digitali possono essere eseguite una serie di istruzioni: può essere copiato il valore di un

DI o DO (che sarà ON oppure OFF) all'interno di un registro, può essere assegnato il valore ON e OFF ad un output, così come, tramite l'istruzione PULSE, è possibile invertire lo stato del segnale di uscita digitale per un tempo indicato. Di seguito sono riportate delle istruzioni su I/O digitali di sistema.

```
! Istruzioni su I/O digitali di sistema
R[1]=DI[1]
DO[1]=ON
DO[2]=PULSE, 0.2sec
DO[3]=R[2]
```

Esistono anche I/O digitali del robot, identificati dalle sigle RI ed RO. Su di essi si possono eseguire le medesime istuzioni che si eseguono sugli I/O digitali di sistema.

A.9 MACRO PER L'APERTURA E LA CHIUSURA DELLA PINZA

A.9.1 *Macro per l'apertura della pinza*

Di seguito è riportato il programma macro GRIP_OPEN impiegato durante tutte le prove effettuate. Per prima cosa, si controlla che la pinza non sia già aperta, sfruttando il sensore presente nella pinza stessa: se tale condizione è verificata, il programma termina. Se invece la pinza non è già aperta, essa viene aperta, tenendo conto che l'output RO[5] e l'ouput RO[6] non possono mai essere entrambi contemporaneamente ON oppure contemporaneamente OFF, pena il danneggiamento della pinza stessa. È quindi necessario dare il comando per l'apertura della pinza (RO[5]=ON), aspettare il tempo necessario affinché la pinza si apra, e successivamente togliere il comando per l'apertura della pinza (RO[5]=OFF). Nelle schede tecniche della pinza Schunk Co-act EGP il tempo di apertura, così come quello di chiusura, è riportato essere di 0,2 secondi. Si nota tuttavia che un'attesa pari a 0,2 secondi non è sufficiente per la presa di alcuni pezzi. Di conseguenza, è stata impostata un'attesa di 0,25 secondi. Successivamente, si controlla che la pinza si sia effettivamente aperta; in caso contrario, se la pinza non si è aperta dopo un tempo di timeout, si provvede all'error handling e ad eventuali allarmi (in questo caso non impostati perché l'argomento allarmi non è stato approfondito, ma approfondibile in seguito).

```
! GRIP_OPEN: apertura pinza

! Se la pinza risulta aperta, il programma termina
IF (RI[2]),JMP LBL[1]

! Apertura pinza
RO[5]=ON
WAIT    .25(sec)
RO[5]=OFF

WAIT (RI[2]) TIMEOUT,LBL[2]
JMP LBL[1]
```

```
13
14 ! Error handling
15 LBL[2]
16
17 LBL[1]
```

A.9.2 *Macro per la chiusura della pinza*

Di seguito è riportato il programma macro GRIP_CLOSE impiegato durante le prove effettuate. Questo programma è del tutto identico al precedente, con la differenza che si comanda l'output RO[6].

```
1  ! GRIP_CLOSE: chiusura pinza
2
3  ! Se la pinza risulta chiusa, il programma termina
4  IF (!RI[2]),JMP LBL[999]
5
6  ! Chiusura pinza
7  RO[6]=ON
8  WAIT    .25(sec)
9  RO[6]=OFF
10
11 WAIT (!RI[2]) TIMEOUT,LBL[2]
12 JMP LBL[1]
13
14 !Error handling
15 LBL[2]
16
17 LBL[1]
```

A.10 ISTRUZIONI CALL E RUN

Le istruzioni CALL e RUN servono per richiamare l'esecuzione di un secondo programma. La differenza tra le due è che in un caso il programma principale attende l'esecuzione del sotto programma, mentre nel secondo caso il programma principale continua nella sua esezuine senza interrompersi.

A.10.1 *Istruzione di chiamata al sottoprogramma CALL*

L'istruzione CALL sposta il flusso di esecuzione alla prima linea di un altro programma (sottoprogramma). Quando nel sottoprogramma è eseguita l'istruzione END, il controllo torna nel programma chiamante (programma principale) alla linea successiva a quella che contiene l'istruzione CALL. Di seguito è riportata l'istruzione per chiamare la marco che si occupa di aprire la pinza.

```
1 ! Chiamata alla macro per l'apertura delle pinze
2 CALL GRIP_OPEN
```

Va sottolineato che ad un sottoprogramma possono anche essere passati degli argomenti. Si consideri l'esempio che segue:

```
1 ! Chiamata al sottoprogramma con passaggio di argomenti
2 CALL SUB_PRG(1,R[6])
```

In questo caso, il programma principale chiama il sottoprogramma PRG_1. Il sottoprograma avrà il valore 1 nel primo registro argomento (AR[1]=1) ed il valore contenuto nel registro R[6] nel secondo registro argomento (AR[2]=R[6]).

A.10.2 *Istruzione RUN*

Durante l'esecuzione di un programma, l'istruzione RUN avvia l'esecuzione di un altro programma. La differenza rispetto all'istruzione CALL sta nel fatto che il programma chiamante, invece di sospendere la propria esecuzione fino a quando il programma chiamato non termina, prosegue con le istruzioni successive senza interrompersi. Nell'utilizzo dell'istruzione RUN è necessario prestare attenzione ad alcuni aspetti:

- Non è possibile avviare l'esecuzione di un programma che utilizzi lo stesso gruppo di assi del programma chiamante. Perché possano essere eseguiti contemporaneamente, due programmi devono utilizzare due gruppi assi diversi. In alternativa, uno dei due programmi non deve utilizzare alcun gruppo assi (ad esempio un programma che si occupa solo della visione o della gestione input/output).
- Per sincronizzare programmi in esecuzione nello stesso momento, utilizzare le istruzioni per l'elaborazione di valori contenuti in registri numerici e di attesa di condizioni.
- Nel caso in cui il programma richiamato tramite l'istruzione CALL avvii un ciclo, è necessario prevedere un'uscita dal ciclo quando il programma principale termina. Ad esempio, si può prevedere che il programma principale scriva in un registro prima dell'istruzione END e che il programma richiamato con l'istruzione CALL controlli periodicamente il contenuto di quel registro.

Di seguito è riportato l'esempio di un programma SUB_PRG, che viene richiamato tramite l'istruzione RUN dal programma principale. Il programma SUB_PRG assegna due output ed aspetta che diventi ON un input, dopodiché scrive nel registro R[1].

```
1 ! Programma per la gestione I/O
2 DO[1]=ON
3 DO[2]=OFF
4 WAIT DI[1]=ON
5 R[1]=1
```

In seguito è riportato invece il programma principale che richiama il sottoprogramma per la gestione I/O. Prima di richiamare il programma, si annulla il valore nel registro utilizzato per la sincronizzazione. Una volta chiamato il programma per la gestione I/O, il programma

principale esegue una movimentazione e successivamente aspetta, se necessario, che il programma SUB_PRG abbia concluso.

```
1 ! Programma principale
2 R[1]=0
3 RUN SUB_PRG
4 J P[1] 100% FINE
5 J P[2] 100% FINE
6 WAIT R[1]=1
```

Può accadere che se il programma viene interrotto, ad esempio perché si rilascia il pulsante uomo morto, e poi si vuole ripartire dalla prima riga del programma principale, molto spesso questo non è possibile in quanto il programma secondario sta ancora andando (soprattutto nel caso di programma secondario che esegue un ciclo che termina solo quando il programma principale ha terminato). In questo caso compare un messaggio di errore che informa che il programma che si vuole richiamare tramite l'istruzione RUN è già in esecuzione. Per risolvere questo problema, quando si eseguono i programmi dalla teaching pendant e non da iRProgrammer da PC, nella schermata dei programmi si preme su MONITOR (F4), dove è possibile vedere i programmi in esecuzione. Si seleziona quindi il programma in esecuzione e si preme su ABORT. In questo modo sarà possibile far ripartire correttamente il programma principale.

A.10.2.1 *Istruzioni sui registri di posizione in programmi senza gruppo di movimento*

Un problema che si può riscontrare durante l'esecuzione di programmi multitasking è che, se non diversamente specificato, non è possibile eseguire istruzioni sui registri di posizione. Ciò significa che un programma senza gruppo di movimento, che si occupa ad esempio di estrarre delle informazioni dal sistema di visione, non può eseguire istruzioni sui registri di posizione. Non è quindi possibile eseguire l'istruzione indicata di seguito.

```
1 ! Istruzioni sui registri di posizione
2 VISION RUN_FIND 'PRG_VISION'
3 VISION GET_OFFSET 'REC2D4' VR[1] JMP LBL[1]
4 PR[10]=VR[2].FOUND_POS[1]
5 R[18]=PR[10,2]
```

In particolare, nell'eseguire la riga 5, compare l'errore "*INTP-214: specified group not locked*". Per fare in modo che le istruzioni sui registri di posizione possano essere eseguite in un programma senza gruppi di movimento è necessario spuntare "*55: No motion PR operate mode*" sulla schermata di configurazione del sistema. Ciò permette di eseguire operazioni sui registri di posizione in parallelo con l'esecuzione del programma principale, quindi senza fermare i movimenti del robot eseguiti nel programma del task principale. Inoltre, quando si eseguono operazioni sui registri di posizione in multitasking, è necessario utilizzare l'istruzione LOCK PREG per evitare la sovrascrittura del registro di posizione utilizzato. Infatti, la sovrascrittura di un registro di

posizione può avere effetti molto gravi come il movimento del robot su una posizione inaspettata.

BIBLIOGRAFIA

[1] Martin Berger, Gernot Bachler e Stefan Scherer. «Vision Guided Bin Picking and Mounting in a Flexible Assembly Cell». In: *Intelligent Problem Solving. Methodologies and Approaches*. A cura di Rasiah Logananthara, Günther Palm e Moonis Ali. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 109–117. ISBN: 978-3-540-45049-8.

[2] Y. Yanagihara e T. Kita. «Parts picking in disordered environment». In: *Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*. 1991, 517–522 vol.2. DOI: 10.1109/IROS.1991.174524.

[3] Marcos Alonso, Alberto Izaguirre e Manuel Graña. «Current Research Trends in Robot Grasping and Bin Picking». In: *International Joint Conference SOCO'18-CISIS'18-ICEUTE'18*. A cura di Manuel Graña, José Manuel López-Guede, Oier Etxaniz, Álvaro Herrero, José Antonio Sáez, Héctor Quintián e Emilio Corchado. Cham: Springer International Publishing, 2019, pp. 367–376. ISBN: 978-3-319-94120-2.

[4] Masahiro Fujita et al. «Bin-picking Robot using a Multi-gripper Switching Strategy based on Object Sparseness». In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. 2019, pp. 1540–1547. DOI: 10.1109/COASE.2019.8842977.

[5] Carlos Martinez, Heping Chen e Remus Boca. «Automated 3D vision guided bin picking process for randomly located industrial parts». In: *2015 IEEE International Conference on Industrial Technology (ICIT)*. 2015, pp. 3172–3177. DOI: 10.1109/ICIT.2015.7125566.

[6] A. S. Olesen, B. B. Gergaly, E. A. Ryberg, M. R. Thomsen e D. Chrysostomou. «A collaborative robot cell for random bin-picking based on deep learning policies and a multi-gripper switching strategy». English. In: *Procedia Manufacturing*. Vol. 51. Cited By :4. 2020, pp. 3–10. URL: www.scopus.com.

[7] In.

[8] Peichen Wu, Wenbo Chen, Hongrui Liu, Yifan Duan, Nan Lin e Xiaoping Chen. «Predicting Grasping Order in Clutter Environment by Using Both Color Image and Points Cloud». In: *2019 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*. 2019, pp. 197–202. DOI: 10.1109/WRC-SARA.2019.8931929.

[9] Douglas Lanman e Gabriel Taubin. «Build Your Own 3D Scanner: 3D Photograhy for Beginners». In: *SIGGRAPH '09: ACM SIGGRAPH 2009 courses*. New Orleans, LA USA: ACM, 2009, pp. 1–87.

[10] Joaquim Salvi, Jordi Pagès e Joan Batlle. «Pattern codification strategies in structured light systems». In: *Pattern Recognition* 37.4 (2004). Agent Based Computer Vision, pp. 827–849. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2003.10.002. URL: https://www.sciencedirect.com/science/article/pii/S0031320303003303.

[11] Liang Wang e Hehua Ju. «A Robust Blob Detection and Delineation Method». In: *2008 International Workshop on Education Technology and Training 2008 International Workshop on Geoscience and Remote Sensing*. Vol. 1. 2008, pp. 827–830. DOI: 10.1109/ETTandGRS.2008.294.

[12] Ales Pochyly, Tomas Kubela, Vladislav Singule e Petr Cihak. «3D vision systems for industrial bin-picking applications». In: *Proceedings of 15th International Conference MECHATRONIKA*. 2012, pp. 1–6.