Politecnico di Milano

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE
Corso di Laurea Magistrale in Ingegneria dell'Automazione

Tesi di Laurea Magistrale

Progettazione e realizzazione di uno strumento di identificazione bave tramite scansione laser

Candidato:
Tommaso Maffei
Matricola 755303

Relatore:
Prof. Luca Bascetta

Correlatore:
Prof. Gianni Ferretti

Anno Accademico 2015–2016

Sommario

Nella seguente tesi si descrive il lavoro che ha portato alla realizzazione di uno strumento di misurazione di superfici 3D con una risoluzione inferiore a 300 μm. Con lo strumento proposto, grazie alla sua elevata capacità di discriminare le coordinate dei punti misurati sulla superficie dell'oggetto in osservazione, è possibile identificare la presenza di bave di piccole e medie dimensioni, presenti sul profilo di oggetti di forma non nota a priori. Si propone, inoltre, l'utilizzo di un manipolatore antropomorfo a 6 gradi di libertà su cui è stato installato un sistema di misurazione a triangolazione con linea laser, che sfrutta un controllo visuale ibrido *contour following* per l'esecuzione di scansioni di un intero bordo dell'oggetto da sottoporre a sbavatura. Il sistema è in grado, in un successivo momento, di procedere all'operazione di sbavatura senza bisogno di dover modificare la configurazione della macchina. Lo strumento descritto si propone come un primo passo verso lo sviluppo di un sistema di sbavatura robotizzata di cerchioni in lega metallica, che sfrutti la destrezza e la grande ripetibilità del manipolatore, per svolgere in maniera autonoma un compito di sbavatura su oggetti anche di grandi dimensioni e collocati arbitrariamente nello spazio di lavoro. La validità dello strumento di misurazione è confermata dai dati raccolti durante la fase sperimentale, grazie anche allo sviluppo di un algoritmo di *image analysis* che permette l'identificazione della linea laser sul bordo dell'oggetto. L'algoritmo sviluppato si contraddistingue per essere computazionalmente veloce e presentare una buona adattabilità alle diverse condizioni di luce, oltre che essere in grado di offrire buone caratteristiche di reiezione ai disturbi luminosi.

Abstract

In the following thesis we describe the work that led to the realization of a measuring instrument of 3D surfaces with a resolution of less than 300 μm. Thanks to the high ability to discriminate illuminated points on the surface of piece the proposed instrument can identify the presence of small and medium-size burrs on the profile of an object which is characterized by unknown shape. An anthropomorphic manipulator with 6 degrees of freedom equipped with the measuring system uses a hybrid visual control *contour following* to scan an entire edge of the object which has to be subjected to deburring. The system is then able to perform the deburring operation without the need to change the machine configuration. The proposed instrument is considered as a first step towards the development of a robotic deburring system of alloy car rims, which exploits the dexterity and the great repeatability of the manipulator, to independently carry out a deburring task of objects, even large, and arbitrarily placed in the workspace. The validity of the instrument is confirmed by measurements made during the experimental phase, thanks to the development of an algorithm of *image analysis*, which allows to identify the laser line on the edge of the object, that it is computationally fast, can easily adapt to different light conditions, and has good rejection of the bright disturbances.

Indice

Elenco delle figure

Elenco delle tabelle

Elenco dei grafici

Capitolo 1

Introduzione

Oggigiorno il rapporto fra il mondo del lavoro e l'automazione è molto discusso, in particolare l'attenzione dei mezzi di comunicazione di massa si è concentrata sull'evidenziare i possibili sviluppi futuri di una completa sostituzione dell'uomo da parte di robot, nello svolgimento di determinate professioni.

> *"Se un robot svolge gli stessi compiti, dovrebbe essere tassato allo stesso livello"*
> *"If a robot comes in to do the same thing, you'd think that we'd tax the robot at a similar level"*
> (Bill Gates - 17 Febbraio 2017 - Monaco)

Lo sviluppo tecnologico che ha portato, e porterà, alla sostituzione di alcune mansioni, per altro usuranti, tradizionalmente compiute direttamente da umani ed adesso completamente automatizzabili, come il controllo dei mezzi pubblici sotterranei o la verniciatura di componenti automobilistici, ha un importante risvolto sociale da tenere in considerazione. Infatti anche se teoricamente si può essere in grado di sostituire i lavoratori nello svolgimento di tali compiti, per ora non si è in grado di rendere superfluo il ruolo umano nell'organizzazione e gestione dei suddetti lavori, definendo nuove figure lavorative specializzate nella gestione di tali sistemi automatici [9].

In questo orizzonte tecnologico si inserisce il Progetto FIDELIO (*FIxture-less DEburring of wheeLs by human demonstratIOn*), finanziato dall'Unione Europea nell'ambito del Settimo Programma Quadro (FP7/2007-2013), che si prefigge di studiare la fattibilità, ed i relativi vantaggi, di un innovativo approccio al processo industriale di sbavatura. L'informazione proveniente da una serie di misurazioni ed osservazioni dell'operato dell'uomo nello svolgere l'operazione di sbavatura viene utilizzata per creare una rappresentazione astratta del compito, che permette di poter svolgere tale mansione in maniera autonoma, anche in presenza di un ambiente differente da quello studiato nella fase di apprendimento. I manipolatori robotici moderni sono in grado di realizzare una moltitudine di differenti compiti, ad alta velocità e con

un alto grado di ripetibilità, ma sono ancora troppo poco adattabili per poter svolgere autonomamente compiti differenti. Per esempio nel processo di sbavatura si ha la necessità di rimuovere imperfezioni superficiali, rimuovendo il giusto quantitativo di materiale in eccesso, ottenendo la forma nominale dell'oggetto in lavorazione. Attualmente nella maggior parte dei casi tale processo è svolto a mano, con inevitabili carenze nella precisione e maggiori costi in termini di mano d'opera e tempo impiegato; tutto ciò porta ad una stima del costo annuale dovuto alle operazioni di sbavatura di 500 milioni di euro solo in Germania [2]. Quello che rende particolarmente complesso il problema dell'automatizzazione della procedura di sbavatura è la non perfetta conoscenza delle caratteristiche delle bave da rimuovere e la loro localizzazione, un lavoratore esperto infatti sfrutta il senso della vista e del tatto per controllare se è necessario procedere con un passaggio di sbavatura, oppure l'oggetto presenta una forma soddisfacente, e per effettuare questa operazione non necessita di conoscere l'esatta forma dell'oggetto a priori. Al giorno d'oggi si è in grado di automatizzare il processo di rimozione delle bave con manipolatori robotici, ma la traiettoria dell'end-effector è decisa a priori ed eventualmente con il controllo della forza di contatto.

L'impiego di un robot che compia l'operazione di sbavatura in maniera autonoma e flessibile, cioè adattabile ad oggetti di forma differente, al posto di un umano, aiuterebbe a raggiungere una maggiore produttività, ma soprattutto una migliore qualità del prodotto finito, oltre che rimuovere l'operaio specializzato da un lavoro ripetitivo in un ambiente reso insalubre dalla presenza di residui metallici in sospensione e dal forte rumore, tutelandolo dall'esposizione a rischi causati dalla vicinanza con parti mobili senza la possibilità di frapporvi barriere. Infine, semplificando il processo di programmazione o apprendimento del robot questa fase del lavoro potrebbe essere svolta direttamente dallo stesso operaio specializzato.

Per risolvere le problematiche attuali e poter svolgere in maniera versatile, accurata e veloce il processo di sbavatura di un cerchione di forma arbitraria per automobile, la tesi si pone come obbiettivi:

1. la realizzazione di un sistema di *computer vision* che permetta di rilevare le caratteristiche geometriche principali, ed il profilo 3D dei bordi dell'oggetto da sottoporre a sbavatura, in modo da poter approcciare il pezzo in maniera precisa;

2. l'uso di tecniche di misura non a contatto per localizzare e caratterizzare la bava, solitamente di dimensioni che variano da 2-3 mm a 4-10 µm;

3. fornire un *feedback* durante l'operazione di sbavatura sulla posizione dell'end-effector, sulla profondità della smussatura e sulla qualità della lavorazione; in modo da avere un controllo sulla differenza fra la posizione attuale e quella desiderata dell'utensile.

L'obiettivo che ci si è prefissati è perciò di identificare la dimensione di bave di piccola-media grandezza (0.5-3 mm). La metodologia di misura scelta è la scansione della superficie dell'oggetto tramite sensori ottici attivi. Il laser scanner, in seguito ad una analisi teorica preliminare, è risultato essere lo strumento in grado di fornire una maggiore densità di punti rilevati ed una maggiore reiezione ai disturbi luminosi, e di rispondere alle caratteristiche desiderate:

Risoluzione misura :	$<$0.3 mm
Accuratezza misura :	0.1 mm
Intervallo di distanza in cui operare:	0.1 m – 1.5 m
Velocità di scansione:	$53\,\mathrm{cm}^2/s$ (intero cerchione da $40 \times 40\,\mathrm{cm}$ in $30s$)

L'utilizzo di un manipolatore robotico per l'operazione di sbavatura è una scelta auspicabile per diversi motivi, esso infatti garantisce la giusta destrezza per poter posizionare l'utensile di sbavatura nella posizione più conveniente ad effettuare l'operazione, ed inoltre consente la movimentazione della linea laser utilizzata per la misurazione del profilo, senza la necessità di modificare il set-up della macchina; in questo modo non si hanno vincoli sul volume e sul peso dell'oggetto da misurare, che non necessita di essere mosso durante l'operazione di identificazione del profilo e di sbavatura.

I principali problemi riscontrati nel processo di automatizzazione della sbavatura sono:

- l'imprecisione della lavorazione soprattutto su profili spigolosi, l'utensile si distacca dalla superficie e con il solo sensore di forza non è in grado di continuare la lavorazione in maniera ottimale;
- l'assenza di localizzazione della bava e della sua caratterizzazione geometrica;
- l'inutile lavorazione dell'intero profilo del pezzo ed eventualmente di un pezzo esente da bave rilevanti;
- l'eccessiva smussatura del pezzo che può portare alla diminuzione della qualità dello stesso;
- l'approccio iniziale al pezzo estremamente critico che necessita una precisa conoscenza a priori della sua posizione e della sua forma geometrica.

Nella presente tesi si sono presi in considerazione tutti gli aspetti di criticità presentati, concentrandosi però sugli aspetti legati alla *computer vision* ed al rilevamento automatico del profilo di sbavatura dell'oggetto di forma non nota a priori. La sperimentazione è stata svolta presso il Laboratorio di

Meccatronica e Robotica per l'Innovazione (*MERLIN*) nella sede del Dipartimento di Elettronica, Informazione e Bioingegneria (*DEIB*) del Politecnico di Milano, grazie all'utilizzo del robot Smart Six prodotto da COMAU.

1.1 Struttura della tesi

Il lavoro svolto ed i risultati ottenuti sono esposti in 8 capitoli, compreso il presente capitolo introduttivo; in coda ai capitoli sono state aggiunte sezioni di appendice contenenti il codice sorgente degli algoritmi sviluppati. La tesi è sviluppata nel seguente modo:

- **Capitolo 1:** introduzione in cui si riportano le linee guida del progetto di tesi e gli obbiettivi;
- **Capitolo 2:** panoramica riguardante lo stato dell'arte del processo di sbavatura, delle tecniche di misurazione di superfici *3D* e del controllo visuale di un manipolatore robotico; partendo dalla definizione delle caratteristiche delle imperfezioni superficiali di un manufatto metallico, si procede descrivendo le cause che portano alla loro formazione ed in particolare il processo produttivo del cerchione automobilistico in lega metallica, successivamente si elencano i metodi oggigiorno utilizzati nelle industrie manifatturiere per la rimozione di tali imperfezioni, oltre che metodi innovativi introdotti più recentemente od in fase di sperimentazione. In relazione all'approccio di sbavatura scelto si descrivono le varie metodologie di misurazione della superficie tridimensionale di oggetti di forma non nota e una breve introduzione al controllo di un manipolatore robotico che sfrutti informazioni visuali per la pianificazione del moto;
- **Capitolo 3:** studio di fattibilità preliminare eseguito per giungere alla progettazione dello strumento di misura di superfici 3D, si indicano anche alcune possibili configurazioni del sistema e di componenti utilizzabili ed infine si riporta la risoluzione teorica dello strumento.
- **Capitolo 4:** descrizione del setup sperimentale effettivamente realizzato presso il laboratorio di ricerca, si riportano le specifiche tecniche dei componenti utilizzati, oltre che la descrizione dell'architettura software, necessaria allo svolgimento della fase sperimentale ed al rilevamento dei dati; infine viene riportato il funzionamento dell'algoritmo di controllo visuale (*contour following*) del manipolatore per poter eseguire la movimentazione del sistema di misura;
- **Capitolo 5:** si entra nel merito dell'acquisizione dei dati per mezzo di uno strumento di triangolazione laser, si descrivono perciò gli algoritmi di *image analysis* sviluppati per poter identificare con precisione la

posizione della linea laser nell'immagine acquisita tramite videocamera, si riportano le prestazioni degli algoritmi ottenute su misurazioni di oggetti differenti sia durante la movimentazione manuale del manipolatore che durante la movimentazione ottenuta tramite controllo visuale (*contour following*);

- **Capitolo 6:** descrizione del processo di calibrazione dell'intero strumento di misurazione di superfici 3D; il processo di calibrazione è suddiviso in fasi distinte: la calibrazione della videocamera in configurazione *eye-in-hand*; e la calibrazione del sistema a triangolazione composto dal proiettore laser e camera; sono state implementate differenti metodologie di calibrazione della videocamera sfruttando software appositi (CalLab) ed algoritmi sviluppati *ad hoc*, nel capitolo si riportano i risultati ottenuti e si esegue un confronto;

- **Capitolo 7:** risultati sperimentali ottenuti con il sistema di misurazione proposto, sia durante il controllo manuale del manipolatore che durante il controllo visuale, si riportano le misurazioni effettuate del profilo di un manufatto metallico di forma non nota a priori che permettono di identificare la presenza di piccole medie imperfezioni sulla superficie dell'oggetto misurato;

- **Capitolo 8:** conclusioni del lavoro di tesi, descrizione riassuntiva dei risultati ottenuti nei capitoli precedenti e delle problematiche riscontrate durante la fase sperimentale, si riportano anche possibili sviluppi futuri ed ampliamenti delle funzionalità del sistema.

Capitolo 2

Stato dell'arte

2.1 Il processo di sbavatura

Il moderno processo industriale che porta alla creazione di cerchioni in lega metallica si articola in diversi passaggi; un passaggio fondamentale per la realizzazione di un prodotto finito che soddisfi tutte le caratteristiche volute, sia estetiche che di forma e di sicurezza, è il processo di sbavatura (*deburring*). La rimozione di parti metalliche e difetti che si creano durante il processo di realizzazione del pezzo: ad esempio per le lavorazioni di stampaggio e foratura, avviene tramite asportazione meccanica o rimozione del materiale in eccesso, e prende il nome di sbavatura. Ancora oggi nonostante l'elevato grado di automazione della tecnologia di produzione industriale, oggetti di forma complessa come i cerchioni automobilistici, vengono sottoposti ad un processo di sbavatura manuale; questa procedura richiede l'impiego di personale specializzato, in un lavoro ripetitivo ed in un ambiente reso insalubre dalla presenza di residui metallici in sospensione, e dal forte rumore. Negli ultimi anni si è posta molta attenzione da parte di ricercatori ed industrie nel trovare soluzioni tecnologiche innovative che permettessero di rendere automatico tale processo, proprio perché in questo modo si potrebbe velocizzare la produzione e migliorare la qualità del prodotto finito, oltre che limitare l'esposizione a rischi da parte del lavoratore e abbassare i costi di produzione. Uno studio compiuto in Germania eseguendo un confronto fra varie industrie del settore *automotive* e macchine utensili ha riportato che il 15 % della forza lavoro e del tempo di ciclo produttivo sono impegnati nel processo di sbavatura, inoltre il tasso di reiezione del pezzo a causa della permanenza di bave alla conclusione del procedimento di finitura è del 2 % , mentre quello di guasto delle macchine causato sempre dalle bave residue risulta del 4 % ; tutto ciò porta ad una stima del costo annuale dovuto alle operazioni di sbavatura di 500 milioni di euro solo in Germania [2].

2.1.1 Definizione di bava

Le bave possono essere definite come eccedenza di materiale, sporgente dalla superficie del pezzo, mentre quest'ultima é definita dalla geometria desiderata di tale prodotto; la formazione di bave è dovuta al processo di realizzazione del pezzo [2]. La bava può essere rilevata su di un profilo lineare (a) oppure alla congiunzione di due spigoli (b) o su di una superficie (c) ed è caratterizzata da due parametri geometrici: altezza e larghezza; vedi figura (2.1).

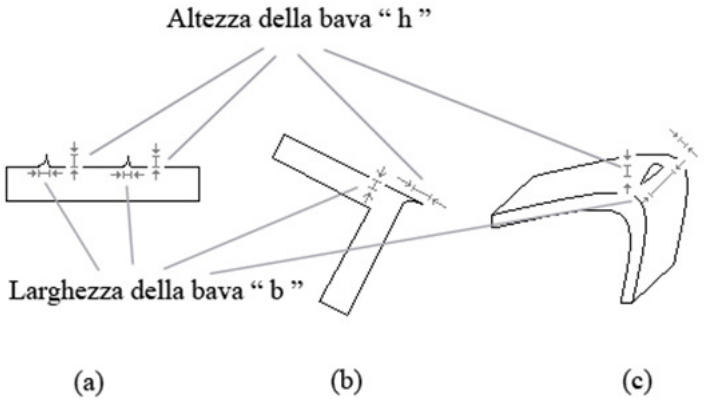


Figura 2.1: Rappresentazione schematica di una bava: (a) profilo lineare (b) congiunzione di due spigoli (c) superficie.

2.1.2 Classificazione delle bave in base al processo di produzione

Le bave sono state classificate in diversi modi, a seconda: della forma geometrica, della lavorazione che le ha prodotte e della posizione relativa al pezzo lavorato. Ciò che ci interessa ai fini del nostro lavoro è capire la tipologia di bava che possiamo riscontrare sui pezzi che andremo a sbavare e la posizione approssimativa in cui ci aspettiamo la presenza di tali imperfezioni, essendo note le lavorazioni precedenti subite dal cerchione o dal pezzo metallico preso in analisi. Per questo motivo considereremo la seguente classificazione per tipologia di bave (tratta da [2]):

- **Bave laterali**- generate dall'accumulo di materiale dovuto alla compressione del pezzo, si formano in direzione ortogonale alla direzione di compressione e presentano un piccolo rapporto altezza-larghezza oltre ad avere una forma generalmente arrotondata.
- **Bave rigirate**- generate da scaglie di materiale che invece di essere tranciato viene ripiegato a lato, si presentano come bave di lunghezza relativamente grande e con forma ricurva, vengono anche dette bave di uscita perché si formano su spigoli del pezzo attraversati per ultimi dall'utensile.
- **Bave di lacerazione**- generate dalla stiratura del materiale del pezzo che si estroflette dalla forma voluta piuttosto che essere tranciato in maniera netta, presentano un rapporto altezza-larghezza abbastanza grande ed un profilo appuntito.
- **Bave di taglio fuori controllo**- generate dal distacco di una parte del pezzo in lavorazione prima che questo venga effettivamente tagliato dalla base dovuto alla forza di gravità, la formazione di questo tipo di bave viene evitata se si afferra oltre alla base del pezzo in lavorazione anche la parte tagliata.
- **Bave di saldatura**- generate durante l'operazione di saldatura, sono formate dalla sovrabbondanza di materiale saldato che fuoriesce dalla superficie del pezzo, hanno un piccolo rapporto altezza-larghezza e presentano in generale forme appiattite e tondeggianti.
- **Bave di stampaggio**- generate durante l'operazione di stampaggio, sono posizionate lungo il profilo dell'oggetto ortogonali alla superficie dello stesso, presentano forme allungate e frastagliate, caratterizzate da un lungo profilo irregolare e larghezza variabile dal decimo di millimetro a qualche millimetro.

Numerosi studi sono stati compiuti per descrivere i processi di formazione delle bave durante le lavorazioni subite dal pezzo semilavorato per giungere

al prodotto finito; l'utilizzo di modelli analitici e metodi FEM ha perciò permesso di determinare quali parametri per ogni lavorazione influenzano la formazione di bave, in quali posizioni sarà probabile riscontrare delle imperfezioni e quali tipologie di bava si possono creare.
Nel processo di **tornitura** si formano due tipologie di bave: le **bave laterali** e le **bave rigirate**, le prime si formano a lato della sezione di taglio se il tagliente dell'utensile si estende oltre il bordo del pezzo; le seconde si formano alla fine della sezione di taglio quando la tornitura viene interrotta (per ragioni geometriche o perché richiesto dalla lavorazione).
Quando il pezzo viene sottoposto a **trapanatura** si forma o una **bava laterale** oppure una **bava rigirata** a seconda delle condizioni di affilatura della punta o della sua velocità di rotazione; se la punta è ben affilata il foro di uscita della punta ha un profilo uniforme e si forma una bava laterale lungo tutta la circonferenza, in genere di altezza ridotta; mentre se la punta è poco affilata o la velocità ridotta, una scaglia di materiale rimane a lato del foro a formare una bava rigirata.
Durante il processo di **fresatura** il semilavorato viene scavato dall'utensile lungo una direzione denominata di alimentazione (*feed*), si individuano così 8 diversi spigoli nel quale si possono formare delle bave: lo spigolo di approccio dell'utensile al pezzo e quello di uscita, dove rispettivamente si formano **bave laterali** e **bave rigirate**; gli spigoli laterali all'incavo prodotto dall'utensile, dove si creano **bave di lacerazione**; ed infine i 4 spigoli laterali all'incavo e posizionati sulle facce di approccio e distacco dal pezzo, dove si formano **bave laterali** o **bave rigirate** a seconda che la velocità periferica dell'utensile sia rivolta verso l'esterno del pezzo o verso l'interno. solitamente le bave poste sulla faccia di approccio al pezzo sono di dimensioni ridotte e vengono denominate **micro-bave** (altezza di bava $< 0{,}1\,\text{mm}$).
Durante il processo di **rettifica** si formano due tipologie di bave: **rigirate** e **laterali**, sullo spigolo di approccio al pezzo si formano delle bave laterali mentre sugli spigoli a lato del pezzo e di distacco da esso si formano bave rigirate; in questa lavorazione sotto condizioni normali, di superficie liscia e direzione di approccio al pezzo parallela alla faccia superiore dell'oggetto, le dimensioni delle bave sono ridotte e queste possono essere considerate micro-bave.

Nella fase sperimentale di questo lavoro (vedi capitolo (7)) si utilizzano due tipologie di oggetti metallici, in modo da poter testare il metodo di sbavatura proposto e la capacità di identificare il profilo di un oggetto con presenza di bave.
Il primo tipo di oggetto è in alluminio e composto da due lastre lisce con bordi sagomati fissate una sopra l'altra tramite quattro viti; le lastre sono state create grazie a macchine a controllo numerico e sono quindi noti i disegni CAD di tali oggetti, un bordo laterale è sagomato a formare archi di circonferenza e profili parabolici con diverse inclinazioni rispetto alla faccia superiore, i

restanti tre lati sono invece ortogonali alla faccia superiore; la struttura a sandwich composta dall'unione delle due lastre permette di posizionare un foglio di laminato metallico e creare bave artificiali. Le lavorazioni subite da tali oggetti sono riconducibili a **fresatura** e **rettifica** quindi possono presentare bave di ridotte dimensioni lungo gli spigoli.
Il secondo tipo di oggetto è un cerchione automobilistico composto da lega di alluminio e magnesio; esso presenta sei razze che collegano la circonferenza esterna alla corona circolare interna, destinata ad ospitare cinque fori per i bulloni di fissaggio. Le lavorazioni necessarie per la realizzazione dell'oggetto sono: il processo di **stampaggio** (*forgiatura*) che verrà descritto nel paragrafo (2.1.3)); la **tornitura**, realizzata con tornio verticale necessaria per la lavorazione di canale e centro ruota (alloggia i mozzi dell'autovettura e l'apparato frenante); ed infine la **fresatura**, nella quale si impiega una fresa a codolo per ottenere la forma delle razze. A causa delle lavorazioni sopracitate sul cerchione si possono riscontrare bave che seguono: il profilo delle razze, il cerchio esterno dei fori per i bulloni di fissaggio e la circonferenza esterna del cerchione.

2.1.3 Il processo produttivo del cerchione

I cerchioni in lega che sono stati utilizzati nella fase sperimentale del nostro lavoro sono realizzati tramite processo di stampaggio. Durante la fase di stampaggio si costringe del materiale metallico a riempire la cavità, detta stampo, che rappresenta in negativo la forma desiderata del cerchione; il metallo viene deformato permanentemente sottoponendolo a forze esterne e comprimendolo fra due semistampi, la deformazione è ottenuta sfruttando la plasticità del materiale ed è in genere facilitata dalle alte temperature. La formazione di bave è una condizione necessaria al processo di stampaggio, per questo il blocco di partenza, denominato *billetta*, ha un volume leggermente maggiore di quello del pezzo finito e viene inoltre predisposta la formazione di una linea delle bave che corrisponde alla linea di giunzione dei due semistampi; la linea delle bave viene realizzata predisponendo sottili canali, i canali *scartabava*, che in generale sono ortogonali alla direzione di compressione e portano ad una serie di vantaggi dal punto di vista produttivo:

- assicurano un tempo di vita degli stampi maggiore: la presenza di bave ha la funzione di ammortizzare l'urto fra i due semistampi o di evitarlo del tutto se si considerano canali scartabava di tipo aperto.
- assicurare un perfetto riempimento delle cavità: le sottili intercapedini in cui si formano le bave, fanno si che il metallo si raffreddi più rapidamente in tali punti ed offra maggiore resistenza radiale, si facilita in questo modo la rimonta assiale del materiale fino ad aderire con precisione a tutta la superficie dello stampo.
- permettono di accogliere il materiale in eccesso.

2.1.4 Metodi industriali di finitura

Nelle industrie sono attualmente in uso varie tecniche per la finitura superficiale dei componenti metallici:

- Sbavatura manuale
- Sbavatura termica
- Finitura di massa
- Sbavatura fluidodinamica
- Sbavatura agli ultrasuoni
- Sbavatura elettrochimica ECD
- Sbavatura robotizzata

ognuna di queste tecniche presenta dei vantaggi e degli svantaggi che andremo a definire.

La **sbavatura manuale** viene realizzata da un operatore specializzato che esercita la propria forza fisica e sensibilità per rimuovere le imperfezioni laddove vengano localizzate. L'operatore si serve della vista e del tatto per identificare la posizione della bava e dell'esperienza per decidere con quale postura avvicinare il pezzo all'utensile e quanta forza esercitare nel contatto. L'utensile di cui ci si serve può essere: una semplice lima, uno scalpello o della carta vetrata per le rifiniture di precisione, da utilizzare con metalli di durezza non troppo elevata; oppure una smerigliatrice o fresa equipaggiata con dischi abrasivi, punte dentate o spazzole, che può essere fissata ad un banco di lavoro oppure impugnata dall'operatore; od infine una cinghia abrasiva che messa in rotazione, permette di asportare il materiale dalla superficie a contatto assecondandone la forma. Esempi di utensili di sbavatura si possono vedere nella figura (2.2) .

Questo è l'approccio più diffuso nelle industrie ed è anche quello largamente praticato nella produzione dei cerchioni; questa tecnica di sbavatura permette di essere molto versatili nella tipologia di pezzi da sbavare e non richiede tempi aggiuntivi di preparazione per passare dalla lavorazione di un lotto ad uno differente, di contro la sbavatura manuale incide in maniera ingente sul costo del prodotto finito e non assicura livelli costanti di accuratezza ed uniformità del processo di finitura.

La **sbavatura termica** si basa sul principio che parti con un grande rapporto superficie-volume si ossidano più facilmente delle altre ad elevate temperature. Questa tecnica permette perciò di ossidare e rimuovere le bave, caratterizzate da lunghezza di bava molto grande rispetto alla larghezza; si

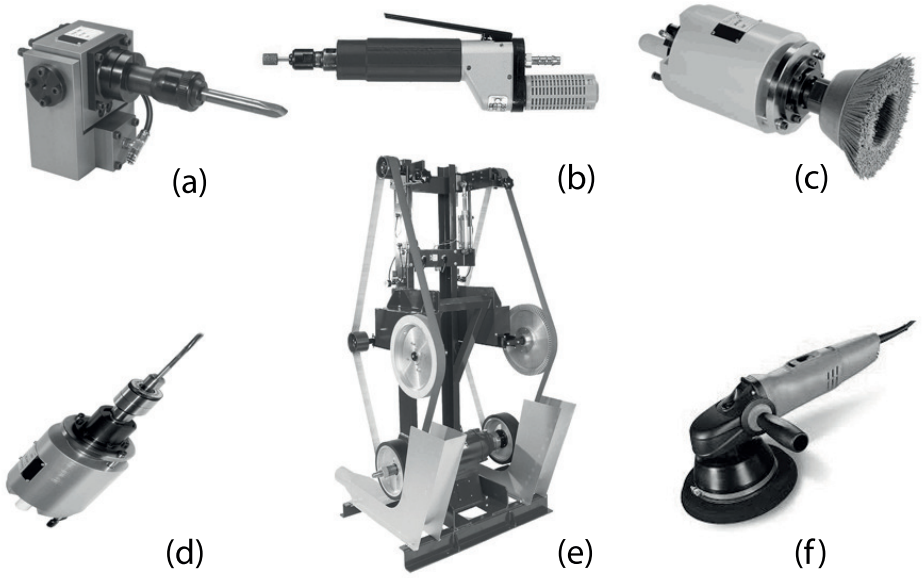


Figura 2.2: (a)Scalpello (b)Smerigliatrice impugnabile con punta in pietra (mola cilindrica) (c)Spazzola (d)Fresa (e)Cinghia abrasiva montata su postazione rotante (f)Smerigliatrice impugnabile con disco abrasivo

posiziona il pezzo in una camera dove viene iniettato il combustibile, una volta innescata la combustione la camera raggiungerà una temperatura di 2500-3000 ℃ provocando l'ossidazione del metallo. Questa tecnica presenta il vantaggio di non essere legata alla conoscenza della forma del pezzo ma risulta inefficace per alcune tipologie di bave, come quelle di fusione (caratterizzate da un basso rapporto superficie volume), e impraticabile per leghe di metalli con un basso punto di fusione, come leghe di alluminio (punto di fusione 660 ℃ circa).

La **finitura di massa** è costituita da diverse lavorazioni: la burattatura, la sabbiatura, l' *abrasive flow machining*; il lavoro di rimozione delle bave viene realizzato dall'usura dovuta allo sfregamento con particelle o grani di materiale abrasivo. Nel processo di burattatura la finitura è svolta a secco, più pezzi vengono posizionati in un contenitore riempito di materiale abrasivo, il contenitore viene poi movimentato inducendo un moto vorticoso del materiale abrasivo, grazie al contatto fra i pezzi e l'abrasivo le bave vengono rimosse e la superficie del pezzo lucidata; un esempio di applicazione industriale di questa tecnica è il *centrifugal barrel finishing*; nel processo di sabbiatura invece, il materiale abrasivo, viene proiettato grazie ad un getto di aria compressa sulla superficie del pezzo, questo fa si che le parti superficiali e più debolmente

attaccate al pezzo vengano rimosse; esiste anche una tecnica di sabbiatura che utilizza particelle di ghiaccio secco in alternativa al materiale abrasivo, il diossido di carbonio a contatto con la superficie del pezzo sublima non lasciando residui e provocando uno shock termico che coinvolge lo strato superficiale del pezzo. Una nuova tecnica di recente sviluppo permette di sbavare e lucidare le superfici interne dei componenti metallici e prende il nome di *abrasive flow deburring* (AFD o AFM), essa consiste nel forzare un fluido particolarmente viscoso, una pasta abrasiva, a passare internamente al pezzo riempiendo le cavità che necessitano di sbavatura; la pasta abrasiva viene spinta da pistoni idraulici in modo da percorrere più volte la superficie del pezzo. Queste tecniche di sbavatura sono facilmente applicabili a qualunque tipologia di pezzo metallico, ed inoltre in alcuni casi, per esempio quando si riscontra la formazione di bave per trapanatura all'intersezione di due fori, la finitura di massa si presenta come una delle più valide tecniche per rimuovere bave in zone altrimenti inaccessibili; d'altronde questi processi non permettono di rimuovere grandi spessori di materiale, perciò bave di grandi dimensioni non saranno rimosse, se non con lunghe e ripetute lavorazioni: inoltre le parti che verranno eliminate sono più localizzate agli angoli o sui bordi del pezzo e meno nelle cavità perciò se si vuole preservare alcuni profili dall'essere smussati bisogna procedere a complesse mascherature, infine la sabbiatura genera una certa rugosità superficiale dovuta all'impatto dei grani sulla superficie del pezzo che può ridurre la qualità del prodotto finale.

La **sbavatura agli ultrasuoni** è eseguita immergendo il pezzo da lavorare in un liquido che costituisce il mezzo di propagazione delle onde di pressione (ad una frequenza di $\sim 20\,\text{kHz}$), il liquido può essere o semplice acqua oppure contenere delle particelle di abrasivo. Le vibrazioni ultrasoniche che si propagano nel mezzo liquido producono una moltitudine di microscopiche bolle che venendo a contatto con la superficie del pezzo generano una forza sufficiente a rimuovere il materiale superficiale. Questa tecnica risulta efficace per la sbavatura di componenti, anche di forma molto complessa e con cavità o fori molto piccoli, difficilmente raggiungibili con altre lavorazioni; la quantità di materiale asportato è ridotta rispetto ad altre tecniche di sbavatura, ed inoltre questa lavorazione presenta il difetto di non poter mantenere gli spigoli vivi del pezzo che subisce il trattamento, siccome l'azione abrasiva non agisce selettivamente sul punto predeterminato ma su tutta la superficie, per ovviare a questo problema spesso si ricorre a mascherature di protezione.

La **sbavatura elettrochimica** ECD si basa sul processo di dissoluzione anodica, l'energia elettrica viene convertita in energia chimica che attiva la reazione di ossidazione del metallo. L'utensile di lavorazione viene collegato al polo negativo di un generatore a corrente continua quindi esso diventa il catodo, mentre il pezzo da lavorare viene collegato al polo positivo divenendo

l'anodo, facendo circolare un liquido adoperato come elettrolita (generalmente una soluzione di cloruro di sodio in acqua) nell'intercapedine fra anodo e catodo, gli ioni negativi che si formano si avvicinano alla superficie del pezzo e cedono elettroni per reazione di ossidazione, mentre il metallo forma degli ioni positivi per dissoluzione che vengono attratti verso il catodo e trasportati dal flusso del liquido lontano dal pezzo. Questa tecnica risulta molto efficace per la sbavatura di precisione di qualunque metallo perché non altera le proprietà fisiche e chimiche della superficie, ed essendo una lavorazione senza contatto non crea stress meccanico o effetti termici, inoltre non è influenzata dalla durezza del metallo da lavorare e comporta bassa usura dell'utensile. La velocità di asportazione del materiale è governata dalle leggi di Faraday sull'elettrolisi[1] ed è proporzionale alla densità di corrente, perciò le estroflessioni di materiale che formano le bave, in principio verranno rimosse maggiormente fino ad appianarsi; il problema di questa tecnica è che porta però ad una asportazione, oltre che del materiale costituente la bava, anche del materiale adiacente, per questo motivo non è adatta alla rimozione di grandi bave[2] [29].

La **sbavatura fluidodinamica** si realizza investendo il pezzo grezzo con un getto di liquido, acqua o olio, ad alta pressione (300-3500 bar). Questo processo viene realizzato in apposite celle robotizzate dove il pezzo viene immerso nel liquido e successivamente investito dal getto in pressione; permette di sbavare agevolmente anche piccole cavità o fori di ridottissime dimensioni, ottenendo inoltre una pulizia superficiale del pezzo. Questa tecnica però non risulta particolarmente efficace su metalli con elevata durezza, ciò comporta una eliminazione solo parziale della bava; per incrementare l'efficacia di questa tecnica viene aggiunto del materiale abrasivo al getto in pressione, in questo modo si riesce a sbavare anche metalli con maggiore durezza, ma si verifica l'inconveniente di dover successivamente rimuovere i residui di materiale abrasivo dal componente.

La **sbavatura robotizzata** viene realizzata in ambiente industriale afferrando il pezzo grezzo e trasportandolo nella posizione convenuta in modo da ricevere il trattamento di sbavatura. Il trasporto del pezzo avviene o con manipolatori multi-assi che posseggono il sufficiente grado di destrezza per compiere la traiettoria voluta oppure tramite nastri trasportatori; una volta che il pezzo è in una posizione nota viene bloccato e il robot fornirà le

[1] Le leggi pubblicate da Michael Faraday dopo aver osservato il fenomeno dell'elettrolisi possono essere riassunte dalla seguente formula: $m = \frac{MQ}{zF}$; dove m è la massa di sostanza rimossa dall'anodo, M è la massa molare della sostanza, $Q = It$ è la quantità di carica trasportata nel caso di corrente costante I nel tempo t, z è il numero di valenza degli ioni della sostanza, $F = 96485\,C/mol$ è la costante di Faraday.

[2] Se una bava presenta grande altezza è necessario più tempo per rimuoverla, di conseguenza il materiale subito adiacente alla radice di bava verrà scavato eccessivamente.

coppie ai giunti necessarie per contrastare le forze di contatto con l'utensile o con il getto in pressione utilizzati per la sbavatura. Vengono sfruttate le caratteristiche di alta ripetibilità e accuratezza nei movimenti del manipolatore robotico per automatizzare le operazioni di sbavatura; per far questo si ricorre ad un attenta pianificazione delle traiettorie di moto dell'organo terminale del robot, al quale viene fissato il pezzo da lavorare oppure l'utensile di sbavatura. Per ottenere la sbavatura automatizzata di un pezzo di forma complessa occorre perciò un tempo più o meno lungo di progettazione del task di movimentazione che dovrà essere rinnovato ogni qualvolta si cambi lotto di produzione.
Il manipolatore può essere equipaggiato con sensori di forza, per poter compiere la lavorazione controllando la forza di contatto esercitata, o anche essere supportato da un sistema ottico per la localizzazione dei pezzi da sbavare; purtroppo ancora oggi nel processo industriale ci si limita ad utilizzare le informazioni date dal sensore ottico: come convalida della presenza o meno del pezzo sul pallet, oppure per sapere con precisione la posizione e l'orientamento su di un piano dell'oggetto da lavorare, lasciando alla programmazione off-line il compito di gestire il task di sbavatura. L'obiettivo di compiere il processo di sbavatura in maniera del tutto automatizzata all'interno di un azienda è raggiunto solo come routine di lavorazione del pezzo, ogni pezzo è considerato aderente al modello predeterminato a priori, e la successione di operazioni da compiere per rimuovere le bave, comprese le traiettorie di moto dell'utensile e del pezzo, è anch'essa prestabilita. L'implementazione di sistemi di controllo avanzati, come il controllo ad impedenza o il controllo attivo di forza o il controllo adattativo, unita allo sviluppo di utensili studiati *ad hoc* con caratteristiche di cedevolezza, permettono di migliorare le performance del processo di sbavatura robotizzata e raggiungere l'obbiettivo di sbavatura di oggetti di forma complessa, ma nota; oppure di forma non nota ma con bave localizzate solo sul contorno del pezzo (effettivamente una sbavatura seguendo il profilo 2D non noto del semilavorato). Questa processo di sbavatura risulta molto efficace per industrie che producono profilati metallici o pezzi con forme semplici, anche perché garantisce risultati costanti su tutti i prodotti del lotto di lavorazione. Nonostante venga implementato nella produzione di cerchioni in lega di alluminio, in tale settore non risulta sufficiente a soddisfare le richieste di qualità del prodotto finito, inoltre i tempi di programmazione off-line non sono compatibili ed economicamente sopportabili nel caso di produzione di lotti di piccole quantità; il che richiede un ulteriore processo di sbavatura manuale per alcune tipologie di cerchioni.

2.1.5 Metodi innovativi di finitura

Negli ultimi vent'anni le sfide tecnologiche che necessariamente si devono affrontare per automatizzare il processo di sbavatura, e per poter ottimizzare tale lavorazione ai fini di migliorarne l'affidabilità e la qualità dei risultati

hanno attratto numerosi ricercatori; la ricerca si è sviluppata in diverse direzioni, sviluppando nuove metodologie di sbavatura, come la sbavatura termica mediante raggio laser o mediante l'uso di particelle abrasive ferromagnetiche; oppure ponendo attenzione alla rimozione di "micro-bave" soddisfacendo così le richieste produttive moderne che esigono una sempre migliore finitura superficiale con tolleranze dell'ordine di pochi micron.
Un approccio innovativo di sbavatura robotizzata è proposto in [37], viene utilizzato un manipolatore SCARA in grado di eseguire la sbavatura, seguendo un profilo planare, senza che sia nota la forma del pezzo, e riducendo il set-point di velocità di avanzamento se lungo il profilo vengono rilevate bave. Il robot utilizza la *legge di controllo ibrida forza/velocità* ed è equipaggiato

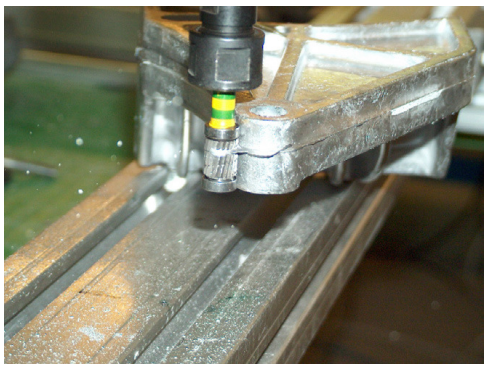


Figura 2.3: Sbavatura robotizzata con fresa a denti elicoidali e cuscinetti a sfera progettata ad hoc [37]

con un sensore di forza **ATI 65/5** montato sull'end-effector ed una fresa a denti elicoidali modificata opportunamente; alla fresa sono stati aggiunti degli anelli con cuscinetti a sfera alla base ed alla testa della punta, in modo da poter esercitare grande forza di contatto nella sola direzione normale al profilo senza penetrare con la fresa nel pezzo (vedi figura (2.3)). Per realizzare il task richiesto è necessario controllare la velocità, tangenziale al profilo del pezzo, con cui avanza la punta della fresa e la forza normale esercitata dalla fresa sul pezzo; per scomporre velocità e forza nelle componenti tangenziale e normale è necessario conoscere l'*angolo di contatto*, cioè l'angolo che si forma fra la direzione x in cui viene misurata la forza dal sensore e la normale al profilo nel punto di contatto; questo angolo viene calcolato utilizzando l'informazione di coppia lungo l'asse perpendicolare al piano x-y, cioè la coppia esercitata dalla fresa riferita al centro del sensore di forza, inoltre per determinare l'offset fra il centro del sensore e il centro della fresa viene proposta una procedura di calibrazione. I risultati sperimentali proposti dimostrano la capacità del sistema di rimuovere bave di stampaggio da un componente metallico di forma non nota ad una velocità di avanzamento di 6 mm/s in assenza di bave e 0.8 mm/s in presenza di esse, il problema di modellazione della forza

di taglio della fresa, dipendente da diversi fattori fra cui le caratteristiche di bava, viene risolto definendo un set-point molto alto di forza normale di contatto e diminuendo la velocità di avanzamento tangenziale quando la coppia lungo l'asse z aumenta oltre una certa soglia.
Un'altra direzione intrapresa dai ricercatori può essere identificata dall'applicazione delle cosiddette tecniche di *Soft Computing*; in questa categoria vengono raggruppati algoritmi e metodologie come: **Logiche Fuzzy**, **Reti Neurali**, **Algoritmi Genetici**, che risolvono, generalmente in maniera sub-ottima, problemi di modellizzazione di sistemi complessi o problemi di controllo.
Nel 2000 è stato proposto da alcuni ricercatori dell'università di Taiwan [17] un sistema di sbavatura robotizzata adattativo che implementa tecniche di modellazione tramite *logiche fuzzy*, unitamente ad un *controllo ibrido di forza/posizione* in grado di regolare la profondità di taglio e la forza di contatto senza l'utilizzo di utensili studiati ad hoc ma grazie solamente ad una fresa montata sul manipolatore robotico ed un sensore di forza. I maggiori ostacoli dal punto di vista controllistico nella realizzazione del task di sbavatura sono rappresentati dalle incertezze del modello dovute alla difficile identificazione della precisa dinamica del manipolatore, che comprende anche la dinamica dei giunti e le forze di contatto, ed inoltre dalle forti vibrazioni introdotte dalla fresa e misurate dal sensore di forza; queste incertezze di misura ma soprattutto di modello possono rendere il sistema instabile oppure condurlo ad un ciclo limite od infine causare delle forti oscillazioni dell'organo terminale al momento del contatto. Partendo dal classico modello di dinamica del robot antropomorfo[3] e utilizzando una relazione generale che definisca le forze di contatto tangenziali e normali[4] si potrebbe realizzare un controllo robusto idealmente in grado di regolare il sistema, ma esso è fortemente dipendente dai parametri di guadagno scelti e quindi di difficile realizzazione pratica; invece realizzando un controllore *fuzzy* le variabili che identificano posizione ed orientamento della terna utensile ($x \in \mathbb{R}^6$), e le variabili che indicano l'errore e la variazione dell'errore di posizione ($s = \dot{x}_d - \dot{x} + \lambda(x_d - x)$), vengono interpretate come *variabili fuzzy* ed attraverso il processo di *fuzzyficazione*, usando come funzioni di appartenenza B-spline di grado zero o superiore, si ottiene come output il valore di coppia da fornire ai giunti del robot. La

[3] Modello dinamico del manipolatore nello spazio operativo: $M(x)\ddot{x} + C(x,\dot{x})\dot{x} + G(x) + D(\dot{x}) = f_\tau + f$; con f_τ vettore che esprime i contributi di coppia ai giunti in termini di forze equivalenti all'organo terminale, $f = [f_n^T + f_t^T, 0, 0, 0]^T$ esprime il contributo di forze all'organo terminale dovute al contatto con l'ambiente.

[4] Modello forze di contatto: $\|f_t\| = g \cdot |\dot{x}_t| + h \cdot |\dot{x}_n| + P_{tool}$ forza tangente alla superficie di contatto del pezzo e rivolta all'opposto del verso di avanzamento, P_{tool} è la soglia costante che rappresenta la forza dell'utensile; $\|f_n\| = \begin{cases} c_2\dot{\delta} + c_1\delta + c_0, & quando \quad \delta \geqslant 0 \\ 0, & altrimenti \end{cases}$ forza normale alla superficie di contatto e rivolta verso l'utensile, δ è la profondità di taglio, c_i coefficienti positivi dipendenti dalle dimensioni della bava, dalla profondità di taglio e dalla velocità dell'utensile.

legge di controllo, $f_\tau = K \cdot s - f + u_{f_k}(\|x\|, s)$ con $u_{f_k}(\|x\|, s) = {\theta_k}^T \cdot v_k$ necessita di tarare i parametri ${\theta_k}^T$ che definiscono i guadagni da assegnare agli output delle funzioni di appartenenza (v_k), ma data la conformazione del sistema, tali parametri si prestano bene ad essere modificati on-line grazie ad una legge adattativa $\dot{\theta_k} = r \cdot v_k \cdot s$ con $r > 0$ costante, risolvendo in tal modo le incertezze di modellazione. Questo controllo permette di ottenere buoni risultati anche a confronto con il lavoro di Kazerooni [16], uno dei ricercatori che per primi hanno gettato le basi per la realizzazione della sbavatura robotizzata; inoltre il controllore è in grado di mantenere regolata la profondità di taglio δ pari a 0.5 mm ad un feedrate medio pari a 2.4 mm/s con un controllo morbido garantito dalle funzioni di appartenenza B-spline superiori al primo ordine, purtroppo dato il grande numero di *regole fuzzy* necessarie, la regolazione è stata limitata al solo piano x-y e richiede comunque che la forma geometrica del pezzo in lavorazione e le sue grandezze siano note a priori, inoltre non è stato considerato il fenomeno di attrito fra utensile e pezzo, infine un particolare non trascurabile è la necessità di avere una profondità di taglio maggiore di zero per poter completare la lavorazione anche in assenza di bave.
Nel 2004 è stato presentato un diverso approccio per la realizzazione di un controllo che permetta al manipolatore robotico di svolgere il compito di sbavatura [33]; viene utilizzata una strategia di *controllo ad impedenza*[5], dove i parametri M_d, B_d, K_d, x_d definiscono la dinamica che si vuole venga assunta dall'organo terminale in risposta alle forze esterne, essi possono anche variare nel tempo in modo da migliorare l'esecuzione della lavorazione; nel controllore proposto sono state usate *regole fuzzy*[6] del tipo **TSK**, esse infatti hanno il vantaggio di non richiedere un ulteriore passaggio di *defuzzyficazione* per ottenere il valore netto dei parametri, come funzioni di appartenenza sono state scelte quelle triangolari, sufficienti, data la struttura del regolatore, a garantire un controllo morbido; siccome le regole *fuzzy* TSK sono determinate dai valori θ_i (con $i = 1 \ldots 34$) che definiscono l'incremento del parametro durante il processo, è importante che questi valori vengano tarati con precisione, per fare ciò nel lavoro proposto vengono utilizzati *algoritmi genetici*, essi selezionano il *cromosoma* che meglio realizza con i sui 34 *alleli*, uno per ogni valore θ_i, il task di sbavatura richiesto. Un altro lavoro degno di nota, per quanto riguarda l'approccio al controllo del processo di sbavatura robotizzata di tipo soft-computing, è stato proposto da Kiguchi-Fukuda [19] nel 2000; i ricercatori hanno sviluppato un controllore neuro-fuzzy adattativo che è in grado di regolare il sistema seguendo il riferimento di forza voluto anche

[5]Definizione della dinamica dell'organo terminale causata dalle forze esterne nello spazio di stato imposta dal progettista: $M_d \cdot \ddot{x} + B_d \cdot \dot{x} + K_d(x - x_d) = f$, con f forze esterne applicate all'organo terminale.

[6]Regole *fuzzy* implementate: $R_i : se((x_1 - x_{1\,d}) \in A_{p_i})\, allora\, \Delta P = \theta_i$,
oppure $R_i : se((\dot{x}_2 - \dot{x}_{2_d}) \in V_{p_i})\, allora\, \Delta P = \theta_i$ con ΔP incremento da assegnare al parametro P definito dalla regola R_i .

in presenza di disturbi dovuti alle vibrazioni causate dall'utensile e anche di ricercare la direzione perpendicolare alla superficie dell'oggetto di forma sconosciuta per poter imporre i vincoli di forza necessari. la problematica maggiore da affrontare consiste nel come applicare la forza di contatto nella direzione giusta, tale forza deve essere inizialmente piccola perchè potrebbe essere applicata nella direzione sbagliata, in questo caso però le vibrazioni della fresa possono dominare il segnale del sensore di forza e portare ad una errata individuazione della direzione del controllo di forza; per superare questo problema viene introdotto il *fuzzy vector method* con cui è possibile correggere ad ogni istante di campionamento la matrice di trasformazione E che definisce la posizione della terna vincolo rispetto alla terna di riferimento del sistema[7], la forza da applicare per il controllo di forza è supposta essere perpendicolare alla superficie del pezzo mentre la forza di reazione è influenzata dalla forza di attrito, non nota, e si trova nel *cono di attrito*, la direzione della forza di controllo viene aggiornata ad ogni istante di campionamento; se la direzione attuale è molto diversa dall'ultima calcolata allora essa è cambiata dell'angolo individuato, ma se il modulo di tale forza è piccolo e c'è il rischio che venga sovrastato dal rumore e dai contributi di forza dell'utensile erroneamente misurati dal sensore, allora la direzione è aggiornata proporzionalmente al modulo della forza ed alla differenza di direzione fra la forza attuale e quella appena calcolata[8]; per il controllo del processo vengono implementati diversi controllori ognuno con uno specifico compito e realizzati da reti neurali che implementano logiche fuzzy di tipo TSK.

2.2 Tecniche di misurazione di superfici 3D

Esistono svariate tecniche per il rilevamento e la misurazione di superfici 3D, osservando quelle elencate in figura (2.4), ci soffermiamo brevemente a descriverne alcuni esempi; le tecniche illustrate in questa sezione sono state scelte per la presenza in letteratura di applicazioni nel campo della sbavatura, oppure perché con caratteristiche congeniali alla nostra applicazione. Questo

[7] Legge di controllo del controllore ibrido forza/posizione:
$\tau = M(q)J^{-1}\{E^{-1}[(I-S)u_p - \dot{E}J\dot{q}] - J\dot{q}\} + h(q,\dot{q}) + F_c sgn(\dot{q}) + J^T E^T f + J^T E^T(S)\, u_f$,
dove τ è la coppia da fornire ai giunti, $M(q)$ è la matrice di inerzia,u_p e u_f sono i vettori di comando per il controllo di posizione e di forza, F_c è l'attrito coulombiano, S è una matrice di selezione che definisce la direzione di applicazione per la forza del controllo di posizione come tangente alla superficie del pezzo, e quella per il controllo di forza come ortogonale alla superficie.

[8] Legge di aggiornamento dell'angolo che determina la direzione di applicazione della forza di controllo: $\Delta\alpha_f = (1-\mu_\alpha)(\alpha_n - \alpha_f)$, dove α_f è l'angolo che identifica l'attuale direzione della forza di controllo, α_n è l'angolo che identifica la direzione normale alla superficie calcolata, μ_α invece è l'output della funzione di appartenenza triangolare, centrata nell'angolo misurato e con ampiezza inversamente proporzionale al modulo della forza misurata, avente come input la direzione della forza attuale misurata.

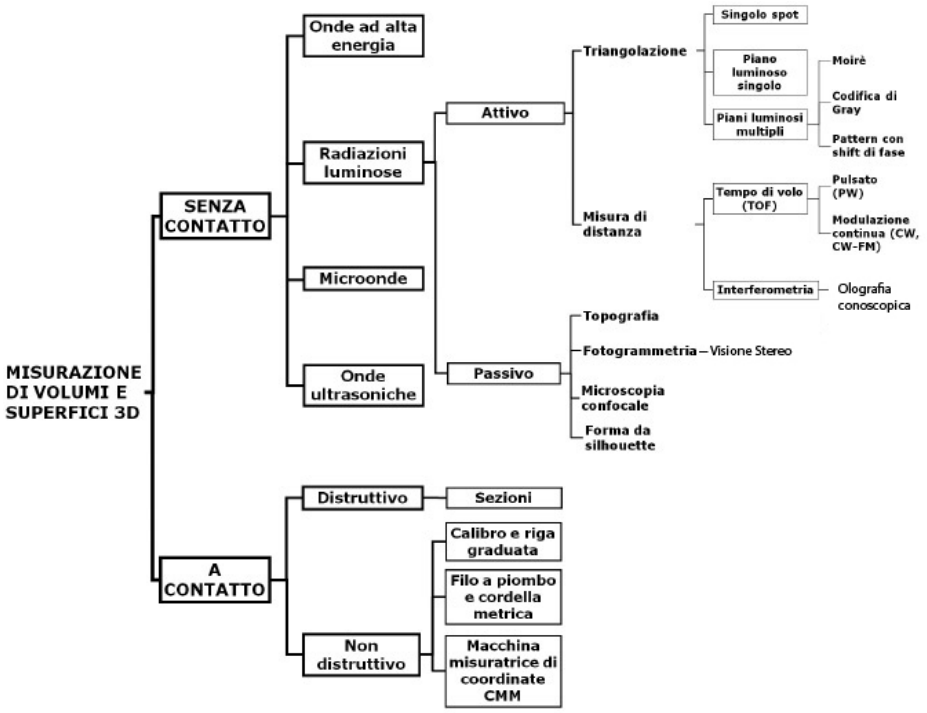


Figura 2.4: Schema rappresentante le attuali tecniche per la misurazione di superfici 3D [28]

paragrafo permette di comprendere meglio lo scenario che ha portato alla scelta della tecnica di triangolazione con linea laser per la nostra applicazione.

Una tecnica molto usata per i rilievi archeologici e per poter mappare oggetti con geometrie complesse è la fotogrammetria[12]; di particolare interesse è il metodo dei fasci proiettivi o *bundle restitution*: questa tecnica consiste nel realizzare diverse fotografie dell'oggetto (minimo 8) da diverse angolazioni (possibilmente $\pm 90°$) e nell'identificare diversi punti caratterizzanti sulla superficie dell'oggetto; in ciascuna immagine questi punti andranno ricercati ed associati in modo da ricavare una nuvola di punti in 3D definita dalle roto-traslazioni necessarie per far coincidere i punti (algoritmo ICP [3], LS3D [13], questa operazione è svolta da software commerciali come RapidForm di INUS Technology, oppure da un software sviluppato ad hoc dall'Istituto di Geodesia e Fotogrammetria dell'ETH di Zurigo [26]), infine a partire dalla nuvola di punti si realizza la mesh della superficie. Questa metodologia di misurazione presenta il pregio di poter essere realizzata con mezzi di basso costo come una reflex digitale (EOS350D) e ciononostante, da dati sperimentali presenta risultati, leggermente inferiori rispetto alla scansione laser, ma con alta precisione e una risoluzione attorno a 1 mm. Come variante delle

tecniche fotogrammetriche possiamo considerare la visione stereo, realizzata con due o più videocamere collegate rigidamente; grazie alle due videocamere è possibile, rintracciando i punti caratterizzanti l'oggetto in entrambe le immagini simultaneamente, ricavare la misura di profondità costruendo una mappa delle disparità causate dal moto apparente di tali punti nelle immagini acquisite da una differente posizione.

I sensori a tempo di volo o ToF forniscono la misura di distanza del punto sulla superficie da rilevare, grazie al tempo che impiega un segnale noto emesso dal sensore a viaggiare verso la superficie, venire riflesso, e tornare al sensore. Il segnale ricevuto infatti è sfasato rispetto al segnale emesso e questa differenza di fase è funzione della distanza percorsa dal segnale stesso, in un altro modello di sensore ToF il segnale invece è pulsato e viene misurato effettivamente l'intervallo di tempo necessario per ricevere il segnale di ritorno. Lo strumento di misura ToF di interesse per la nostra applicazione è la telecamera ToF, essa genera un segnale infrarosso sinusoidale, emesso dal sensore, che investe l'ambiente circostante, il segnale viene poi ricevuto da un sensore simile ad un CCD, ma in grado non solo di registrare l'intensità luminosa per ogni pixel ma anche la misura di distanza derivante dall'informazione del segnale IR riflesso [10]. Due esempi di questa tecnologia sono le camere PMD CamCube 3.0 e Fotonic C70.

Sistemi di misurazione Basati sull'emissione di radiazione luminosa strutturata (LSB *light-structured-based* sono largamente diffusi grazie alla loro accuratezza e flessibilità di utilizzo. Questa tipologia di sensore è oggigiorno utilizzata nell'industria automobilistica ed aeronautica, nonché nella produzione di componenti industriale che necessitano di un elevato livello di controllo qualità. I sistemi LSB sono in grado di rilevare le coordinate 3D di un oggetto proiettando sulla sua superficie una luce con caratteristiche note, la velocità di acquisizione è di sicuro un grande pregio per questo tipo di sistemi. I sistemi LSB si possono suddividere in diverse categorie, a seconda della tipologia di sorgente luminosa ed alla tecnica utilizzata per ricavare l'informazione di distanza: Tecniche a triangolazione e Tecniche interferometriche. Le prime si basano su principi trigonometrici ed utilizzano differenti sorgenti luminose: strisce alternate di diversa intensità luminosa e larghezza (Moiré [4]), oppure ombre di forma nota, o pattern luminosi tali da non creare ambiguità nel riconoscimento dei punti caratterizzanti una volta proiettati sulla superficie; od ancora sorgenti laser: a singolo punto(*single spot*), a linea(slit scanner) o a pattern(Xbox Kinect). Le tecniche interferometriche invece si basano sul principio fisico di interferenza di onde: un segnale sinusoidale e coerente (tipicamente la luce di un proiettore laser), con una lunghezza d'onda nota, a causa di rifrazioni o sfasamenti, quando più raggi si sovrappongono, danno origine a frange, ossia variazioni luminose alternate, rilevate da un sensore, misurando queste frange si ricava la misura di distanza desiderata. Questi sistemi sono caratterizzati da un ottima risoluzione e vengono impiegati per misurazioni di rugosità superficiale di un materiale o anche per definire con

maggior precisione l'unità di misura di lunghezza del sistema internazionale. Per i dispositivi a proiezione di pattern il principio di funzionamento è quello basato sulla triangolazione, alcuni pattern luminosi vengono proiettati sulla superficie e rilevati dal sensore: in una prima categoria, quella degli scanner con pattern a strisce, o anche scanner a proiezione di frange, strisce ad alto contrasto vengono proiettate sulla superficie da misurare ed il disallineamento delle strisce viene registrato con una videocamera; Per ovviare alle ambiguità nell'identificazione della profondità [27], è necessario acquisire una sequenza di più immagini con strisce che variano in larghezza e spostamento per essere in grado di ricostruire l'intera nuvola di punti 3D che rappresentano l'oggetto misurato; durante l'acquisizione delle immagini, tipicamente alcuni secondi, ovviamente l'ambiente misurato deve rimanere statico. Un esempio di questa tipologia di sensore è il HDI Advance R2 di 3D3 solutions. Un'altra categoria di strumenti invece utilizza pattern statici per acquisire tutta l'informazione necessaria per determinare le coordinate 3D dell'oggetto, per ottenere questo risultato è necessario che il pattern proiettato, vedi figura (2.5), sia tale da non condurre ad ambiguità di localizzazione dopo la riflessione sulla superficie dell'oggetto, a condizione di operare nell'intervallo di misura dello strumento.

Figura 2.5: Esempio di pattern statico proiettato da strumenti a triangolazione con luce strutturata[14].

Esempi di strumenti largamente diffusi che utilizzano questa tecnica sono: Asus Xtion PRO 3D e Xbox Kinect, questi strumenti hanno il vantaggio di poter ricostruire la nuvola di punti che definisce la scena con una sola immagine stereo, il che li rende estremamente veloci e capaci di mappare l'ambiente osservato alla stessa frequenza con cui le immagini vengono catturate dalla videocamera, registrando anche rapidi movimenti nell'ambiente misurato, ovviamente il processamento delle immagini deve essere abbastanza veloce da seguire il flusso di dati dalla camera. Un altro esempio di strumento che sfrutta questa tecnica è SCAPE Grid Scanner1 di Scape Technologies.

Altra famiglia di strumenti di misura a luce strutturata sono i sistemi interferometrici ad Olografia conoscopica [34], in seguito si illustra brevemente

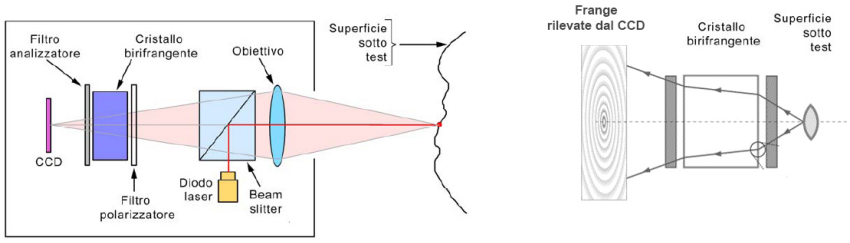


Figura 2.6: Schema di funzionamento della tecnica interferometrica a olografia conoscopica[34].

il principio di funzionamento, vedi figura (2.6). Quando un raggio di luce puntiforme incide su di un cristallo birifrangente (uniassiale), all'interno di questo il raggio viene diviso in due: raggio ordinario e raggio straordinario. Il raggio ordinario si propaga nel cristallo a velocità diversa dal raggio straordinario a causa dell'angolo di incidenza, un secondo polarizzatore sovrappone nuovamente i due raggi che, a causa dello sfasamento, danno origine a frange di interferenza. Le frange vengono registrate da un sensore CCD e dalla misura del loro diametro si ricava la rugosità della superficie e l'altezza delle bave eventualmente presenti. I dati acquisiti dalla scansione sono elaborati rimuovendo i valori con basso rapporto segnale rumore, che rappresentano buchi nella superficie, e sostituendo questi con un valore di default; Inoltre eventuali errori di rilevazione, punti che si discostano molto dai punti adiacenti, vengono cancellati sostituendo tali punti con l'interpolazione dei punti adiacenti. Una successiva elaborazione separa la componente ad alta frequenza (rugosità e bave) da quella a bassa frequenza (ondulazione del profilo), permettendo di identificare il profilo nominale del pezzo. L'operazione di filtraggio è particolarmente critica in presenza di fori, perciò si utilizza un particolare filtro RGR (robust gaussian regression filter[6]). Un esempio di strumento che sfrutta questa tecnologia è ConoProbe di Optimet.

I sensori a triangolazione laser (LTS *laser-triangulation-sensor*) sono oggigiorno i più diffusi per applicazioni di misurazione di superfici 3D e controllo qualità in ambito industriale[4]; nella maggior parte dei casi il sensore viene movimentato grazie a manipolatori robotici o in alcuni casi con l'ausilio di stepper angolari o lineari, in questo modo si garantisce una buona flessibilità di utilizzo e velocità di acquisizione dei dati oltre che l'acquisizione dell'intero oggetto analizzato.

Tali sensori si distinguono in base alla forma della sorgente laser proiettata: singolo punto, linea o multilinea. L'utilizzo di una sorgente luminosa a linea

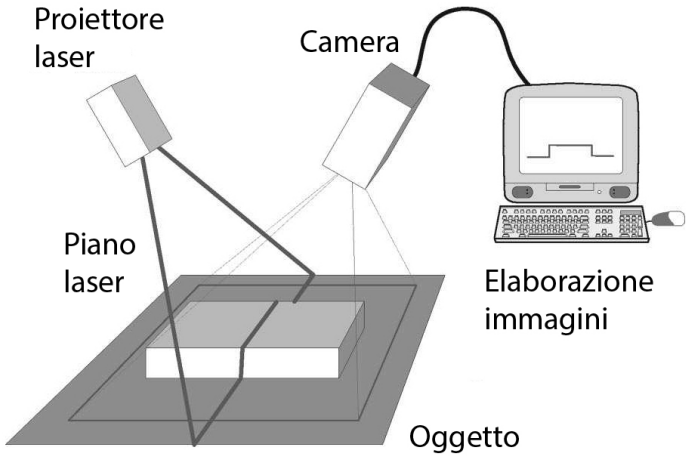


Figura 2.7: Esempio funzionamento Slit scanner o triangolatori a linea laser [4].

laser(*slit scanner*), ben si presta, grazie ad una semplice movimentazione della linea laser, ad acquisire i punti necessari per mappare tramite triangolazione l'intero profilo dell'oggetto con una notevole accuratezza e densità di punti. Come si può notare dalla figura (2.7), il disallineamento dell'immagine della linea laser acquisito dalla videocamera, permette di determinare la distanza fra il centro ottico della camera ed i punti illuminati, in questo modo un singolo profilo 3D dell'oggetto scansionato può essere ricavato per ogni singola immagine, ne consegue che per poter analizzare l'intero oggetto si debba: traslare o ruotare il solo proiettore laser o l'intero sistema LTS,

Campo visivo (FOV):	30°
Accuratezza posizione punto:	+/- 0.04 mm @ 0.36 m(+/- 1 mm @10m, risoluzione 1024p/profilo)
Accuratezza della distanza del punto:	+/- 0.04 mm
Range di misura:	0.11- 0.36 m
Lunghezza d'onda:	655 nm
Potenza del laser:	15 mW
Frequenza di scansione:	4 kHz(765cm^2/s)

Tabella 2.1: Micro-Epsilon scanCONTROL 2800-100 - Esempio specifiche di uno slit scanner.

impiegando qualche secondo per l'intera mappatura; è comunque presente un compromesso fra il tempo di scansione e le dimensioni dell'oggetto da rilevare. Esistono diversi produttori di slit scanner (SICK, Leutze, ShapeGrabber, LMI Technologies, scanCONTROL) oltre che strumenti sviluppati da centri di ricerca in tutto il mondo. Un esempio di strumento commercializzato che sfrutta questa tecnologia è SICK Ranger E50-55, mentre un esempio di dispositivo sviluppato in ambito accademico è il dispositivo BIRIS, equipaggiato con la Twinline Vitana range camera (sviluppata dal gruppo Visual Information Technology del Consiglio Nazionale delle Ricerche del Canada), esso permette di misurare l'informazione 3D con una precisione di circa $50\,\mu$m per distanze che vanno dai 30 ai 50 cm. Infine uno strumento che ben si presta all'analisi delle superfici metalliche per l'identificazione di bave è Micro-Epsilon scanCONTROL 2800-100 / 2810-100, esso infatti rispetta i requisiti di accuratezza nella localizzazione del punto in 3D, ma non quello riguardante il range di misura; lo strumento infatti può effettuare misurazioni solo fino a 0.36 m di distanza e a questa distanza ha un profilo misurabile di 0.14 m,non sufficiente a mappare l'intero profilo del cerchione (0.4x0.4m). Da notare però la velocità di scansione molto elevata, vedi tabella (2.1), che permetterebbe di completare la scansione completa del cerchione nei tempi richiesti.

Si conclude il paragrafo descrivendo le principali problematiche delle varie tecnologie descritte ed i rispettivi punti di forza; è stato dimostrato che la tecnica più affidabile fra quelle basate su sensori ottici per l'identificazione di imperfezioni sulla superficie è la Olografia conoscopica - Conoscopic Holografy [20]. Questa tecnica è infatti in grado di rilevare bave dell'ordine delle decine di micron ($200\,\mu\text{m} - 6\,\mu\text{m}$) con estrema precisione[34], inoltre grazie alla possibilità di proiettare il raggio luminoso e di rilevarlo sul medesimo asse ottico, il sistema permette di analissare profili di forma complessa ed anche la superficie interna dei fori. Presenta però alcuni difetti: non rileva bave molto larghe; fallisce nella misurazione di superfici con elevata curvatura; richiede molto tempo di computazione e scansione ($3.5\,\text{mm}^2/6\,\text{min}$), per queste motivazioni è stato scartato l'utilizzo di un sistema con questa tecnologia. La qualità dei dati 3D dalle camere ToF, invece, dipende fortemente dalla qualità della calibrazione, e gli errori RMS sulle misure, sono comunque significativamente più alti rispetto ad altre tipologie di strumenti. Le camere ToF inoltre, soffrono di un deterioramento del rapporto segnale rumore nel caso in cui le superfici rilevate siano scure, il che si traduce in maggior errore RMS oltre che essere affette da altri errori non sistematici [23]. Per questo motivo anche un sistema ToF non è adatto per la nostra applicazione, difatti la precisione della misura risulta uno degli obbiettivi principali, ed il materiale del cerchione può presentare aree scure. Gli strumenti di misura a proiezione di pattern statici, solitamente hanno la limitazione di poter essere usati solamente in un determinato intervallo di misura, infatti a differenza di altre tipologie di sensore essi non possono essere adattati a funzionare ad

una qualsiasi distanza di lavoro. Inoltre generalmente offrono una mappatura meno densa di punti, cioè determinano per ogni immagine acquisita una nuvola di coordinate 3D meno numerosa rispetto a quelle che si possono ottenere, con più immagini, da sistemi LTS o a proiezione di strisce. Per quest'ultimo motivo uno strumento di questo tipo è stato scartato dalla scelta per la nostra applicazione. Il misuratore a triangolazione con proiezione di frange luminose in sequenza, invece, presenta una forte sensibilità alla luminosità ambientale, questo a causa della limitata potenza luminosa dei proiettori standard, d'altra parte ha il vantaggio di essere composto da una struttura molto semplice e con pochi componenti, oltre al fatto di non avere necessità di parti mobili. La limitata luminosità degli attuali proiettori portatili utilizzabili per l'applicazione rendono questa soluzione inadatta alla nostra applicazione, soprattutto se utilizzata in combinazione con il controllo visuale ad inseguimento del profilo, vedi capitolo(**??**), percui è necessario che il bordo del pezzo in lavorazione sia illuminato.

Infine anche gli strumenti LTS presentano alcune problematiche, ma risultano i più adatti alla nostra applicazione proprio perché permettono di ottenere risoluzioni di misura inferiori a 0.3 mm, con una velocità di acquisizione dei dati elevata, ed anche con condizioni di luce ambientale difficili. Ciononostante presentano anch'essi delle problematiche, infatti sono abbastanza sensibili alle riflessioni speculari; queste possono essere sia doppie riflessioni, in cui la linea laser viene riflessa specularmente su di un altro punto dell'oggetto che a sua volta riflette la luce verso il sensore della camera, sia riflessioni speculari di luce incoerente con la sorgente laser, che può generare punti intensamente illuminati e quindi essere erroneamente identificata come linea laser dall'algoritmo di processamento dell'immagine. Un'altra problematica che si verifica con l'utilizzo di questa tipologia di sensori è causata dalle occlusioni: in alcuni punti del profilo la linea laser proiettata può essere non visibile perché coperta da sporgenze del pezzo in questo caso oltre ad avere un'interruzione nella continuità dei dati, c'è anche il rischio di identificare un riflesso speculare come l'immagine della linea laser desiderata, generando dei valori anomali nelle coordinate 3D.

Come si può dedurre da queste pagine la progettazione di un sistema di misurazione di superfici 3D con luce strutturata, nonostante siano presenti in commercio molti modelli di strumenti differenti, resta ancora un importante campo di studio, le tecniche implementate sono ancora in corso di ottimizzazione, e soprattutto non esiste uno strumento unico adatto ad ogni tipologia di misurazione, è per questo necessario svolgere uno studio focalizzato sulla specifica applicazione, nel nostro caso la sbavatura robotizzata, tenendo in considerazione diversi fattori come l'accuratezza di misura, il volume dell'area da misurare, l'affidabilità della misura, la risoluzione ottenibile, la velocità di scansione ed anche il costo dello strumento stesso o della sua implementazione; Infine per poter risolvere i limiti dovuti al posizionamento ed all'orientamento dello strumento per l'acquisizione della superficie completa dell'oggetto, è

possibile utilizzare un manipolatore robotico, caratterizzato proprio dalla elevata ripetibilità del posizionamento e dalla velocità di movimento.

2.3 Controllo Visuale di un manipolatore robotico

L'utilizzo di un manipolatore robotico per l'operazione di sbavatura è una scelta auspicabile per diversi motivi, vedi (1); in effetti anche la movimentazione della linea laser lungo profilo del pezzo in lavorazione, può essere realizzata direttamente dal manipolatore garantendo sia una grande versatilità del sistema, non ho vincoli sul volume dell'oggetto da misurare ne sul suo peso, sia una elevata velocità di rilevamento delle misure anche di oggetti con forme molto diverse, non avendo necessità di afferrare l'oggetto per muoverlo. Una volta fissato lo strumento di misura sull'end effector del manipolatore, la destrezza del robot permette di seguire anche profili di forma complessa, per questo motivo, si è deciso di sfruttare il manipolatore per posizionarlo nel punto più favorevole al rilevamento delle bave, lungo il bordo del pezzo in lavorazione, e di sviluppare una legge di controllo che muovesse la linea laser lungo tale profilo. Benchè lo sviluppo di un controllo visuale, per realizzare quest'ultimo obiettivo non sia strettamente necessario ai fini della mappatura del profilo 3D, la movimentazione della linea laser può infatti essere realizzata anche come semplice traslazione laterale a velocità costante, l'implementazione di tale controllo migliora le prestazioni del sistema complessivo, perché permette di andare ad analizzare solo le aree con maggiore probabilità di presenza di bave e non l'intero profilo del pezzo, ed inoltre riduce il rischio di occlusioni della linea laser o di errori di misura legati alle diverse condizioni ai bordi dell'immagine. In seguito si descrivono brevemente le caratteristiche che definiscono il controllo visuale di un manipolatore. Gli approcci utilizzati finora nel controllo visuale (*Visual servoing*) possono essere classificati in due grandi categorie: *nello spazio operativo* e *nello spazio delle immagini*. Nel primo approccio, una serie di immagini sono usate assieme ad un modello noto di camera per estrarre la posa di un riferimento nello spazio 3D; Successivamente l'inseguimento del riferimento è eseguito calcolando l'errore fra il riferimento attuale e quello desiderato nello spazio 3D. Nell'altro caso, con approccio del controllo visuale nello spazio delle immagini, l'inseguimento del riferimento è eseguito calcolando l'errore fra la posizione del riferimento e quella desiderata sul piano immagine e riducendolo asintoticamente a zero. Purtroppo a causa della dinamica del manipolatore e delle imprecisioni nella modellazione del sistema, entrambi gli approcci non sono in grado di stabilire il contatto, e mantenerlo, con una superficie i cui precisi posizione ed orientamento non sono noti. Modelli di controllo per sistemi con camera eye-in-hand sono stati introdotti, da Papanikolopoulos e P. K. Khosla [25], in tale schema un controllo adattativo sfrutta la misura di profondità di singole caratteristiche dell'immagine analizzate in tempo

reale durante l'esecuzione; Un altro esempio di controllo visuale avanzato è quello nominato *Visual Compliance*, un sistema nello spazio delle immagini sviluppato da Hutchinson e colleghi [7] in cui il controllo è definito ibrido perché sfrutta una combinazione del controllo nello spazio operativo e nello spazio delle immagini; l'errore su cui si basa tale controllo è definito per alcune componenti, per esempio x e y parallele al piano immagine, nel piano immagine, mentre per altre componenti, per esempio z perpendicolare al piano immagine, nello spazio operativo grazie alle informazioni ottenute per mezzo degli encoder nei giunti del manipolatore. Per concludere Chaumette e Hutchinson [8] forniscono un approccio passo passo alle diverse tecniche di controllo visuale fornendone i risultati teorici e le caratteristiche di stabilità.

Capitolo 3

Studio di fattibilità

Nel seguente capitolo si riporta il lavoro preliminare di ricerca svolto per poter discriminare i componenti necessari alla realizzazione dello strumento di misura, e per poter indirizzare le scelte progettuali grazie ai risultati teorici ottenuti. Dopo aver optato per la misurazione della superficie 3D tramite la realizzazione di un dispositivo costruito appositamente che sfrutti la triangolazione a linea laser, si è proceduto alla scelta dei componenti che costituiscono tale sistema, in particolare:

- la sorgente luminosa: un proiettore laser;
- il sensore di acquisizione delle immagini: una videocamera con la relativa ottica;
- il setup di misura: la disposizione reciproca dei componenti e la distanza di lavoro nominale.

3.1 Scelta della sorgente laser

La risoluzione della misura di posizione del punto nello spazio 3D dipende da diversi fattori, fra questi sicuramente vi è la dimensione del fascio laser (beam diameter), che nella nostra applicazione equivale allo spessore della linea laser nel punto in cui si riflette sulla superficie. Il fascio laser può essere modellato come un fascio gaussiano, perciò per questa particolare categoria la dimensione del diametro del fascio varia a seconda della lunghezza d'onda principale della luce emessa e di quanto si è lontani dal punto di messa a fuoco, quest'ultimo parametro è molto importante perché condiziona la risoluzione del sistema in funzione dell'intervallo in cui si vogliono ottenere misure di distanza, per questo motivo i risultati migliori si ottengono ad una precisa distanza di lavoro Z_{work} a cui il laser viene messo a fuoco.

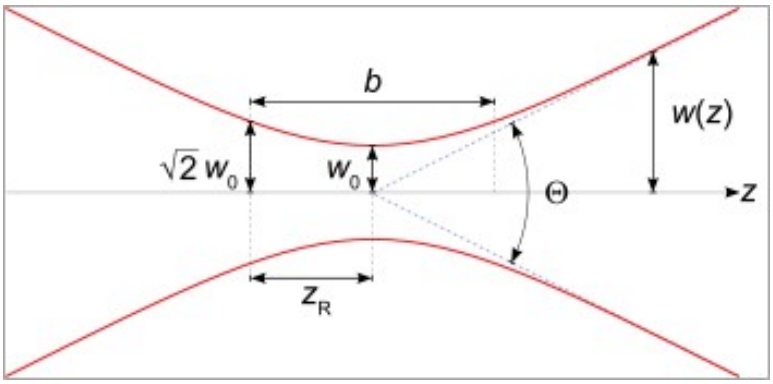


Figura 3.1: Caratteristiche di un fascio laser

$$W(Z) = W_0\sqrt{1+\left(\frac{Z}{Z_R}\right)^2} \qquad Z_R = \frac{\pi {W_0}^2}{\rho} \qquad \zeta \simeq \frac{\rho}{\pi W_0} \tag{3.1}$$

Utilizzando le formule (3.1), ed osservando la figura (3.1) è possibile calcolare la dimensione del punto/linea di luce (*spot size*, *beam width*) alla distanza dal punto di messa a fuoco voluta. Essendo: $W(Z)$ semi-diametro del fascio, W_0 semi-diametro del fascio nel suo punto più stretto ovvero il fuoco, Z distanza dal fuoco lungo la direzione di propagazione del fascio, ρ lunghezza d'onda principale, ζ divergenza del fascio luminoso.

Nelle tabelle (3.2) e (3.3) si presentano vari modelli di sorgenti laser presenti sul mercato. Sono state selezionate solo le sorgenti in grado di realizzare linee laser nello spettro del visibile (in particolare di colore rosso e verde); affianco al nome del modello ed al codice identificativo si trovano dati tecnici rilevanti per la nostra applicazione e forniti dalle schede tecniche dei produttori. L'obiettivo che ci si è prefissati è di identificare la dimensione di bave di piccola-media grandezza (0.4-3 mm) [37]. Perciò lo *slit scanner* dovrà avere determinate caratteristiche, vedi tabella (3.1).

Risoluzione misura :	$<$0.3 mm
Accuratezza misura :	0.1 mm
Intervallo di distanza in cui operare:	0.1 m – 1.5 m
Velocità di scansione:	$53\,\mathrm{cm}^2/s$ (intero cerchione da $40 \times 40\,\mathrm{cm}$ in $30s$)

Tabella 3.1: Specifiche desiderate dello strumento di misura.

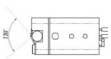
	Modello	Potenza ottica	Lung. d'onda	Diametro del fascio a fuoco	Diametro del fascio @35cm	Diverg. del fascio
	VLM 2 Laser Focusable , Line generator – 80° + Power supply	1.4 mW	635 nm	1 mm	1 mm	
	High-Performance Laser (NT66-375) + Single Line - 60° (NT54-186) + Power supply	5 mW	660 nm	0.025 mm		0.45 x 0.95 mrad
	Micro-Focus Line - 15°, Modulated + Power supply (NT58-100)	<5 mW	670 nm	0.013 mm Distanza di lavoro 65mm	(interpolando linearmente <0.7 mm)	
	Micro VLM Laser Diode, Line - 85° (NT52-266) + Power supply	2.5 mW	670 nm	1mm		<1 mrad Trasversalmente alla linea
	VLM-635-37 LPT Line – 120° + Power supply necessary	<1mW	635 nm		(2 ±1mm @5m)	
	CW532-X-833L Line 120° + Power supply necessary	<5 mW	532 nm		(1.5 mm @ 10m)	<1.2 mrad

Tabella 3.2: Caratteristiche delle sorgenti laser

	Modello	Potenza ottica	Lung. d'onda	Diametro del fascio a fuoco	Diametro del fascio @35cm	Diverg. del fascio
	RLCA532-5-3 Cross Line – + Power supply necessary	5 mW	532 nm			0.1-0.2 mrad
	RLLH635-4-3 Line – 110° + Power supply necessary	4 mW	635 nm		(<0.5 mm @ 0.5m) (<1 mm @ 3m)	1 mrad
	RLLH650-16-3 Line – 100° + Power supply necessary	16 mW	650 nm		(1.5 mm @ 6m)	0.6 mrad
	FVLM2 Crosshair (0222-209-00) focusable Line – 24° + Power supply necessary	1.7 mW	635 nm		(<0.1mm @ 0.1m) (0.4 mm @1m)	1 mrad Adjustable
	Lasiris™ SNF Laser Line Generator (SNF-501L-635-5-90°) Line – 90° + Power supply	5 mW	635 nm	(0.075 mm @ 0.25m)	(in un range di 3 cm la linea resta di dimensioni <0.106mm)	
	Lasiris™ SNF Laser Line Generator (SNF – 701L-635-3-110°) Line – 110° + Power supply	3 mW	635 nm	(0.160 mm @ 0.15m)	(in un range di 4.5 cm la linea resta di dimensioni <0.226mm)	

Tabella 3.3: Caratteristiche delle sorgenti laser

Dalle informazioni raccolte in tabella (3.2) e (3.3), e tenendo conto dell'aspetto importante (anche se non critico) della potenza ottica, si è giunti ad una scelta del modello di laser più appropriato: una linea laser con una potenza nominale di 3.5 mW è ben visibile in una stanza a 9m di distanza, mentre un laser di potenza 6mW, se posto ad una distanza di lavoro di 0.35m, rischia di saturare l'immagine per la troppa luminosità [21]. Questo spiega perché anche la scelta della potenza ottica è da tenere ben in considerazione poiché può portare ad un deterioramento nell'accuratezza della misura. Considerando infine che maggiore è l'angolo della linea laser (Fan Angle) minore è la distanza di lavoro necessaria per coprire l'intero profilo, si ritiene che fra quelli proposti il laser più adatto alla nostra applicazione sia: Lasiris™ SNF Laser Line Generator (SNF-501L-635-5-90°), vedi tabella (3.3); considerando anche che lo spessore della linea laser in un intervallo di 20 cm rimane sotto la risoluzione desiderata.

3.2 Scelta della videocamera e dell'ottica

Esistono diverse videocamere in commercio che presentano caratteristiche assai differenti; principalmente possiamo distinguerle in base alla tipologia di sensore, il *frame rate* massimo a cui può operare ed il gruppo ottico cioè le lenti che possono esservi associate. Per quanto riguarda il sensore principalmente si utilizzano due tecnologie: il CCD (*Charged Coupled Device* o dispositivo a carica accoppiata) ed il CMOS (*Complementary Metal Oxide Semiconductor* o semiconduttore a metallo ossido complementare); il CCD è raccomandato per applicazioni che richiedono una qualità di immagine superiore, specialmente in condizione di bassa luminosità oppure alta velocità di movimentazione della camera, tale caratteristica è dovuta al fatto che i fotodiodi che convertono la luce in carica elettrica, nella modalità *frame transfer* convertono la carica elettrica in tensione elettrica accumulandola in un buffer e trasmettendo un segnale analogico alla scheda della videocamera. Nella tecnologia CMOS, invece, la conversione da carica elettrica a tensione avviene a bordo del singolo pixel ed il segnale trasmesso alla scheda della camera è direttamente digitale, questo porta ad una minore uniformità del segnale di uscita. Ulteriori caratteristiche importanti che distinguono i sensori sono la capacità di produrre immagini a colori o in scala di grigi, le dimensioni del sensore e le dimensioni del pixel o la distanza fra di essi. Nella nostra applicazione il *frame rate* massimo di acquisizione della videocamera risulta anch'esso un parametro importante, perché condiziona la velocità con cui è possibile movimentare il laser e di conseguenza la velocità di scansione dell'intero profilo dell'oggetto. Per esempio con un *frame rate* di 45 fotogrammi al secondo è possibile eseguire la scansione di una striscia pari a 400 mm con risoluzione di 0.3 mm in 30 secondi. Infine, un aspetto

fondamentale da tenere in considerazione nella scelta della videocamera sono le lenti: esse possono essere:

- telecentriche, che limitano l'effetto prospettico che si riscontra con le lenti convenzionali;
- a focale fissa o variabile, a seconda che sia prevista la possibilità di variale la lunghezza focale della lente;
- zoom, se quando la lunghezza focale della camera viene modificata la messa a fuoco viene mantenuta;
- polarizzate, lenti appositamente realizzate per filtrare alcuni raggi luminosi.

Un esempio di camera adatta per l'applicazione in esame è quella in figura (3.2), le cui caratteristiche sono riportate nella tabella (3.4).

Figura 3.2: Camera pike F-100

Tecnologia sensore	CCD
Risoluzione	1000×1000
Risoluzione di profondità bit	$8 - 16$ bit
Dimensione sensore	$2/3''$
Spazio fra un pixel e quello affianco	$7.4 \times 7.4\,\mu$m
Frame rate	60 fps

Tabella 3.4: AVT Pike F-100 specifiche tecniche della camera.

Le linee guida per la scelta del sensore per l'applicazione di triangolazione sono fornite dalla formula che lega il campo visivo della telecamera alla lunghezza focale ed alla dimensione del sensore.

$$\varphi = 2\,\tan^{-1}\left(\frac{Dim}{2\,f}\right) \tag{3.2}$$

Osservando la formula (3.2) che specifica il campo visivo totale del sensore φ in funzione della lunghezza focale f e della dimensione del sensore Dim, si comprende che per aumentare il campo visivo dello strumento, a parità di lunghezza focale sia necessario aumentare le dimensioni del sensore, in modo da comprendere una maggiore superficie durante una singola scansione,

inoltre è ovviamente opportuno ridurre il più possibile la dimensione fra un pixel ed il pixel adiacente (*pixel-pitch*).

La risoluzione del sistema è anch'essa legata alle caratteristiche del sensore e delle lenti. Come dimostrato in (3.14), la capacità di risolvere piccole misure di distanza è determinata in prima approssimazione dalla seguente formula, che definisce la misura di distanza z nel caso particolare in cui il piano laser sia ortogonale al piano dell'oggetto:

$$z = \frac{d \cdot f}{d_{o_c}} \tag{3.3}$$

dove d è la distanza lungo l'asse x della terna camera, del piano laser dal centro ottico (vedi figura 3.6), mentre d_{o_c} è la distanza sul sensore fra il punto luminoso e la proiezione del centro ottico (vedi figura 3.5). La minima variazione $\delta d_{o_c} = \sigma$ equivalente ad 1 pixel rilevabile dal sensore è legata alla minima variazione δz della misura di distanza dalla seguente relazione

$$\frac{\delta z}{\sigma} = \frac{\partial}{\partial d_{o_c}} \frac{d \cdot f}{d_{o_c}} = \frac{-d \cdot f}{{d_{o_c}}^2} \tag{3.4}$$

sostituendo (3.3) nella (3.4) si ottiene

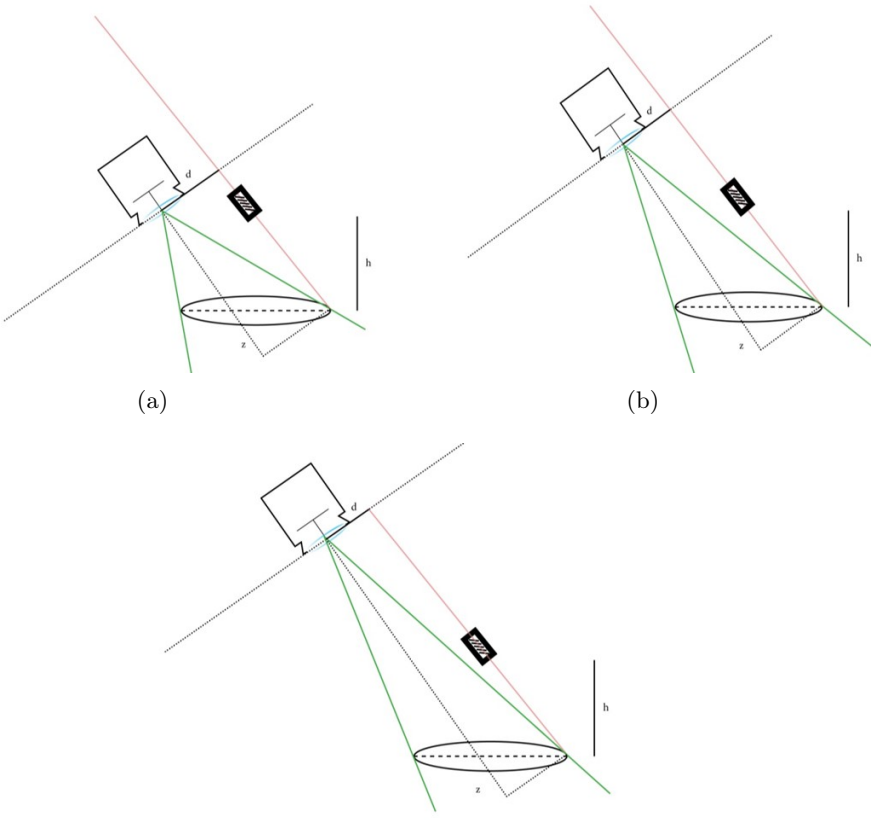
$$\frac{\delta z}{\sigma} = \frac{-z^2}{d \cdot f} \tag{3.5}$$

ora considerando la lunghezza focale f fissa e sostituendo quest'ultima ricavata da (3.2) nell'equazione si ottiene

$$\begin{aligned} \frac{\delta z}{\sigma} &= \frac{-z^2}{d \cdot \dfrac{Dim}{2\,tan^{-1}\left(\frac{\varphi}{2}\right)}} \\ \delta z &= \frac{-z^2}{d \cdot \dfrac{Dim}{2\,tan^{-1}\left(\frac{\varphi}{2}\right)}} \cdot \sigma \end{aligned} \tag{3.6}$$

Ne consegue che volendo ridurre δz, si dovrà scegliere un sensore grande, la dimensione del pixel (σ) piccola, ma soprattutto misurare ad una distanza ravvicinata. Osservando le equazioni (3.5) e (3.2) sembrerebbe di essere di fronte ad un compromesso fra scegliere una focale lunga (f grande), per ridurre il valore di $\dfrac{\delta z}{\sigma}$, oppure scegliere una focale corta che garantisca un campo visivo più ampio, ma scegliendo quest'ultima opzione abbiamo la possibilità di avere un campo visivo più ampio e quindi di poter rilevare la stessa quantità di punti da una distanza più ravvicinata, il che permette di avere z più piccola, che, come si può notare nella equazione (3.6) influenza in modo quadratico il valore di $\dfrac{\delta z}{\sigma}$.

A riprova di queste affermazioni sono state provate alcune configurazioni del sistema a triangolazione variando la lunghezza focale dell'obiettivo (8 mm a sinistra 12 in centro e 16 a destra) e posizionando la macchina ad un angolo di 35° rispetto alla normale del profilo del cerchione. In tabella (3.5) si può notare come la scelta di un ottica con focale fissa da 8 mm garantisca una risoluzione migliore.



(a) (b)

(c)

Figura 3.3: Schemi posizionamento camera in funzione dell'ottica scelta: (a) ottica con lunghezza focale $f = 8$mm, (b) $f = 12$mm, (c) $f = 16$mm

f lunghezza focale (mm)	φ Angolo del campo visivo della camera	d (con laser in bordo destro del cerchione)	z (con laser in bordo destro del cerchione)	δz Risoluzione della misura di distanza
8	49.6°	186 mm	475 mm	1.122 mm
12	34.3°	155 mm	650 mm	1.681 mm
16	26°	136 mm	819 mm	2.281 mm

Tabella 3.5: Risoluzione della misura di profondità per diverse ottiche considerando la configurazione proposta in Figura (3.3).

La camera Pike F-100 si presenta come una buona scelta per l'applicazione in esame, essa è equipaggiata con un sensore a tecnologia CCD da 2/3", difatti si tratta di un sensore di grandi dimensioni e dall'alto rapporto segnale/rumore, inoltre presenta dei pixel di dimensione ridotta ed un *frame rate* abbastanza elevato. La possibilità di poter rilevare immagini a colori non è una caratteristica di primaria importanza per il presente progetto ma può garantire, soprattutto per la scelta di un laser di colore verde, una migliore definizione della linea laser.

3.3 Setup di misura

Come accennato in precedenza, il sistema a triangolazione laser ha bisogno di essere movimentato per poter rilevare l'intera superficie dell'oggetto, per l'applicazione di sbavatura l'utilizzo di un manipolatore robotico antropomorfo garantisce la giusta destrezza per poter posizionare la linea laser nella posizione più conveniente ad effettuare la misurazione del profilo, e successivamente, senza la necessità di modificare il set-up della macchina, di procedere con la rimozione automatica delle eventuali bave rilevate. Siccome lo strumento sarà montato sull'*end-effector* del robot è importante tenere in considerazione anche il peso complessivo dello strumento e soprattutto l'ingombro, in particolare d, la distanza lungo l'asse x della terna camera fra il centro ottico della videocamera ed il piano laser, non può essere troppo grande per non compromettere la maneggevolezza dello strumento.

Il sistema a triangolazione può essere realizzato con differenti configurazioni, come si può notare in figura (3.4). Nella configurazione *ordinaria* la videocamera è posizionata direttamente sopra l'oggetto, parallela al piano di appoggio dell'oggetto, mentre il laser illumina il profilo da misurare lateralmente; ha il vantaggio di permettere il posizionamento della fotocamera direttamente parallela al piano dell'oggetto il che favorisce un controllo visuale nello spazio delle immagini del robot.
Nella configurazione *ordinaria inversa*, invece, il laser e la videocamera sono

scambiati di posizione rispetto alla configurazione ordinaria, in questo caso il laser è posizionato sopra l'oggetto e perpendicolare ad esso; problemi di riflessioni speculari e occlusioni sono limitati grazie a questa configurazione rispetto a quella ordinaria.
Nella configurazione *speculare* invece il proiettore laser e la videocamera sono posizionati ai due lati opposti della normale al piano dell'oggetto, tale configurazione può risultare vantaggiosa per la misurazione di oggetti con una colorazione scura e con una finitura opaca. Infine nella configurazione a *sguardo laterale*, il proiettore e la videocamera sono posizionati dallo stesso lato rispetto alla normale al piano dell'oggetto, in questo caso la risoluzione di misura della distanza è inferiore alle altre configurazioni presentate ma può risultare utile per ridurre il problema delle riflessioni speculari. Inoltre si può aggiungere che per migliorare ulteriormente l'accuratezza della misura e rilevare valori anomali in tempo reale è possibile utilizzare la configurazione *ordinaria inversa* con l'aggiunta di una seconda videocamera speculare alla prima, le misure ridondanti permettono di migliorare l'accuratezza di un fattore $\sqrt{2}$ e di aiutare con il rilevamento dei bordi dell'oggetto e di eventuali riflessioni speculari.

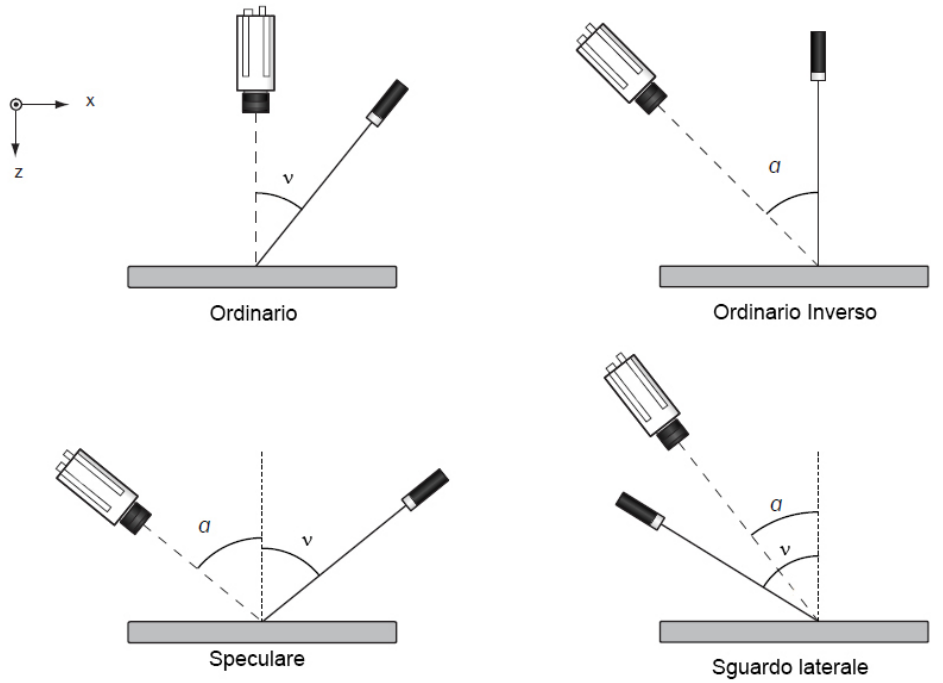


Figura 3.4: Esempi di configurazioni possibili per il sistema a triangolazione laser.

In base alle considerazioni fatte nel presente paragrafo, per lo studio dei

risultati teorici ottenibili dal sistema di misura si è preso in considerazione il setup *ordinario inverso*, con il proiettore laser posizionato in centro al cerchione da misurare ad una distanza di 250 mm da esso; tale distanza garantisce di coprire agevolmente l'intero profilo del cerchione, compreso nel perimetro di una circonferenza di 400 mm di diametro, proiettando una linea laser con un angolo di apertura di 90°. Inoltre si è preso in considerazione la possibilità di variare la posizione della linea laser grazie ad uno stepper angolare permettendo di regolare con alta precisione l'inclinazione del laser α, ovviando in questo modo agli errori introdotti dalla dinamica del robot.

3.4 Risultati teorici ottenibili

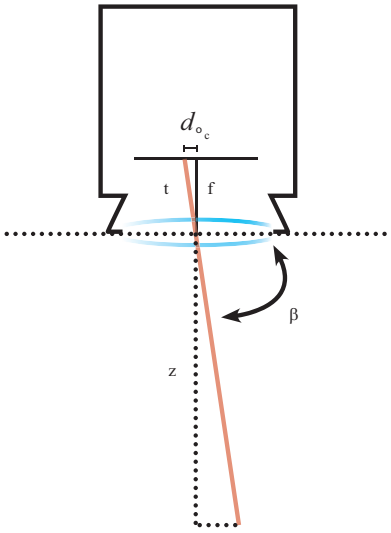


Figura 3.5: Schema modello camera pin-hole.

La formula trigonometria che mette in relazione i parametri della videocamera e la geometria del sistema di misurazione con la misura di distanza z in terna camera, è la seguente [4]

$$z = \frac{f \cdot d}{f \cdot \tan(\alpha) + d_{o_c}} \tag{3.7}$$

dove d è la distanza, lungo l'asse x della terna camera, del piano laser dal centro ottico, f è la lunghezza focale dell'ottica della videocamera, mentre d_{o_c} è la distanza sul sensore fra il punto luminoso e la proiezione del centro ottico (vedi figura 3.5), ed infine α è l'angolo fra il piano z-y della terna camera ed il piano laser (vedi figura 3.6).
Per ottenere tale formula si eseguono alcuni passaggi algebrici partendo dalla nota formula detta *legge dei seni*

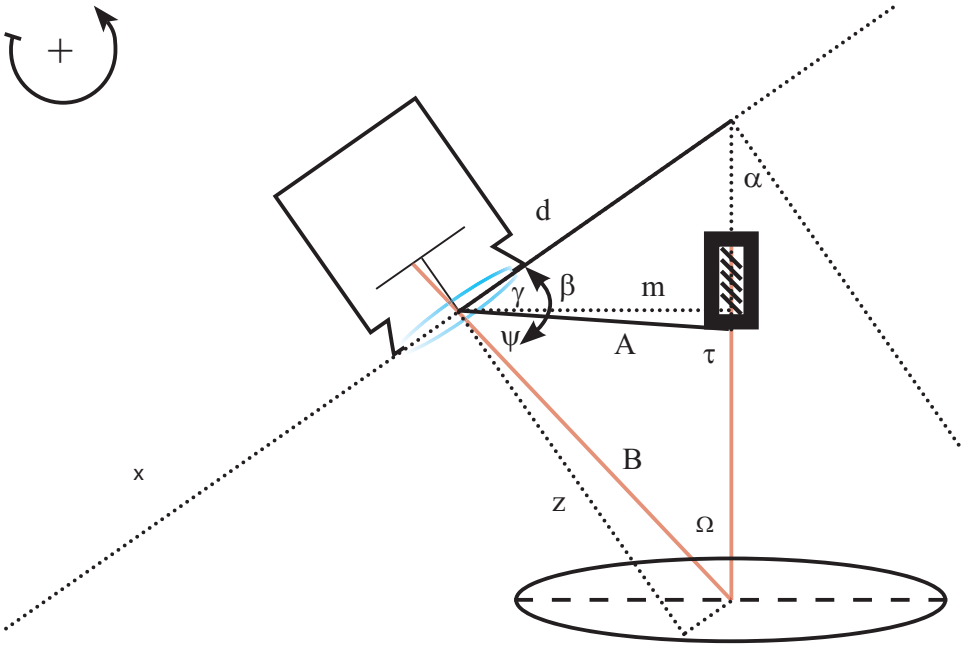


Figura 3.6: Schema sistema teorico di triangolazione laser.

$$\begin{aligned} \frac{B}{sin(\tau)} &= \frac{A}{sin(\Omega)} \\ B &= A \cdot \frac{sin(\tau)}{sin(\tau + \psi)} \end{aligned} \tag{3.8}$$

considerando il modello di camera centrale pin hole (figura 3.5)

$$\begin{aligned} &\begin{cases} t \cdot sin(90° - \beta) &= d_{o_c} \\ t \cdot cos(90° - \beta) &= f \end{cases} \qquad \begin{cases} t \cdot cos(\beta) &= d_{o_c} \\ t \cdot sin(\beta) &= f \end{cases} \\ &\begin{cases} f &= d_{o_c} \cdot tan(\beta) \end{cases} \qquad \begin{cases} \beta &= tan^{-1}\left(\frac{f}{d_{o_c}}\right) \end{cases} \end{aligned} \tag{3.9}$$

dalla figura 3.6 si può notare come con semplici formule trigonometriche si può definire β:

$$\begin{cases} \gamma + (90° - \alpha) + (180° - \tau) = 180° \\ \beta = \gamma + \psi \end{cases} \qquad \begin{cases} \gamma = \alpha + \tau - 90° \\ \beta = \alpha + \tau + \psi - 90° \end{cases} \tag{3.10}$$

sempre osservando lo schema in figura (3.6) si ricava:

$$\begin{aligned} B \cdot cos(90° - \beta) &= z \\ B \cdot sin(\beta) &= z \\ B &= \frac{z}{sin(\beta)} \end{aligned} \tag{3.11}$$

$$\left\{ \begin{array}{c} d \cdot sin(90° - \alpha) = m \\ A \cdot sin(\tau) = m \end{array} \right. \quad \left\{ \begin{array}{c} d \cdot cos(\alpha) = m \\ A \cdot sin(\tau) = d \cdot cos(\alpha) \end{array} \right. \quad \left\{ A = \frac{d \cdot cos(\alpha)}{sin(\tau)} \right. \tag{3.12}$$

si può ora ottenere la formula di triangolazione (3.7) comunemente utilizzata negli algoritmi di *range finding*, partendo dalla formula (3.8) e sfruttando i risultati ottenuti nelle equazioni precedenti (3.10), (3.11), (3.12):

$$\begin{aligned} z &= \frac{A \cdot sin(\tau) \cdot sin(\beta)}{sin(\tau + \psi)} \\ z &= \frac{d \cdot cos(\alpha) \cdot sin(\beta) \cdot sin(\tau)}{sin(\tau + \psi) \cdot sin(\tau)} \\ z &= \frac{d \cdot cos(\alpha) \cdot sin(\beta)}{sin(\tau + \psi)} \\ z &= \frac{d \cdot cos(\alpha) \cdot sin(\beta)}{sin(\beta - \alpha + 90°)} \\ z &= \frac{d \cdot cos(\alpha) \cdot sin(\beta)}{cos(\alpha - \beta)} \\ z &= \frac{d \cdot cos(\alpha) \cdot sin(\beta)}{cos(\alpha) \cdot cos(\beta) + sin(\alpha) \cdot sin(\beta)} \end{aligned} \tag{3.13}$$

$$\begin{aligned} z &= \left\{ \begin{array}{l} d \cdot tan(\beta) \\ \frac{d}{tan(\alpha)} \\ \frac{d \cdot cos(\alpha) \cdot sin(\beta)}{cos(\alpha) \cdot cos(\beta) + sin(\alpha) \cdot sin(\beta)} \end{array} \right. \\ &= \left\{ \begin{array}{lll} \frac{d \cdot f}{d_{o_c}} & se & \alpha = k \cdot \pi, \quad k \in \mathbb{N} \\ \frac{d}{tan(\beta)} & se & \beta = k \cdot \pi + \frac{\pi}{2}, \quad k \in \mathbb{N} \\ \frac{d \cdot cos(\alpha) \cdot sin(\beta)}{cos(\alpha) \cdot cos(\beta) + sin(\alpha) \cdot sin(\beta)} & altrimenti & \end{array} \right. \end{aligned} \tag{3.14}$$

considerando solo l'ultimo caso si ottiene la formula seguente:

$$\begin{aligned} z &= \frac{\frac{d \cdot cos(\alpha) \cdot sin()\beta}{sin(\alpha) \cdot cos(\beta)}}{\frac{cos(\alpha) \cdot cos(\beta)}{sin(\alpha) \cdot cos(\beta)} + \frac{sin(\alpha) \cdot sin(\beta)}{sin(\alpha) \cdot cos(\beta)}} \\ z &= \frac{d \cdot cot(\alpha) \cdot tan(\beta)}{cot(\alpha) + tan(\beta)} \end{aligned} \tag{3.15}$$

$$
\begin{aligned}
z &= \frac{d \cdot cot(\alpha) \cdot \frac{f}{d_{o_c}}}{cot(\alpha) + \frac{f}{d_{o_c}}} \\
z &= \frac{d \cdot \frac{f}{d_{o_c}}}{1 + tan(\alpha)\frac{f}{d_{o_c}}} \\
z &= \frac{d \cdot f}{d_{o_c} + tan(\alpha) \cdot f}
\end{aligned}
\tag{3.16}
$$

che è esattamente equivalente alla formula (3.7) utilizzata in [4]. Inoltre è possibile definire ulteriormente il termine d_{o_c} introducendo le caratteristiche reali del sensore

$$d_{o_c} = -\sigma(\xi_x - c_x)$$

infatti le coordinate di un generico punto P rispetto alla terna camera $({}^c x, {}^c y, {}^c z)$ sono in relazione con le coordinate 2D dell'immagine di P sul piano immagine (ξ_x, ξ_y); ed in prima approssimazione, le equazioni fondamentali della camera prospettica (*modello pin-hole*) affermano che:

$$
\begin{aligned}
\xi_x &= f\frac{{}^c x}{{}^c z} \\
\xi_y &= f\frac{{}^c y}{{}^c z}
\end{aligned}
\tag{3.17}
$$

A queste relazioni bisogna aggiungere l'effetto dovuto alle caratteristiche del sensore di misura ottenendo il modello di camera che utilizzeremo in questa trattazione:

$$
\begin{aligned}
\xi_x &= f_x\frac{{}^c x}{{}^c z} + c_x \\
\xi_y &= f_y\frac{{}^c y}{{}^c z} + c_y
\end{aligned}
\tag{3.18}
$$

dove f_x, f_y sono le lunghezze focali in pixel (incluse le distanze dal centro di un pixel a quello successivo, $f_x = -\frac{f}{S_x}$, $f_y = -\frac{f}{S_y}$, con $S_x = S_y = \sigma$ dimensione del singolo pixel in mm scegliendo un sensore pixel quadrati); infine c_x, c_y sono le coordinate immagine del centro ottico o punto principale (intersezione dell'asse ottico c_z con il piano immagine) questi quattro parametri sono noti come parametri intrinseci della camera.

Questi parametri ci permettono di descrivere direttamente la relazione fra il sistema di riferimento della camera e quello del piano immagine, ovvero la relazione tra le coordinate sul medesimo piano immagine in pixel e in unità di misura SI (mm).

In seguito si simula la misurazione con l'ottica da f=8 mm con la configurazione *ordinaria inversa*, puntando il laser sul punto centrale del cerchione, sul bordo destro più lontano dalla videocamera e sul bordo sinistro più vicino ad essa.

f lunghezza focale (mm)	Parte misurata	φ Angolo del campo visivo della camera	d	z	δz Risoluzione della misura di distanza
	Bordo destro		186 mm	475 mm	1.122 mm
8	centro	49.6°	307 mm	366 mm	0.404 mm
	Bordo sinistro		735 mm	250 mm	0.079 mm

Tabella 3.6: Risoluzione δz per diverse parti del cerchione mantenendo il laser nel centro del cerchione e perpendicolare alla base d'appoggio, con $\alpha = 35°$.

Come si può notare dai dati in tabella, la risoluzione dettata dalle specifiche, con questa configurazione del sistema, viene raggiunta solo nel caso più favorevole, quando effettuiamo misure in prossimità del bordo sinistro del cerchione. Il guadagno, che è legato all'inverso del quadrato della distanza, si riduce molto a causa della distanza (z) crescente che si riscontra allontanandosi dal bordo sinistro.

3.5 Conclusioni

Grazie al sistema a triangolazione laser proposto è possibile, nota la collocazione geometrica del laser (α) e della camera (d) e la lunghezza focale della macchina (f), usando teoremi geometrici, ricostruire la distanza dell'oggetto partendo dalla misura d_{o_c} ottenuta.

Provando a variare l'angolo dell'asse ottico della telecamera α, e portandolo da 35° a 45° (rispetto alla normale del piano del cerchione) si ottengono i risultati in tabella 3.7. Come si può notare dal confronto con i dati in tabella 3.6, si ottengono dei valori migliori di risoluzione dovuti al valore più alto di d e più basso di z.

f lunghezza focale (mm)	Parte misurata	φ Angolo del campo visivo della camera	d	z	δz Risoluzione della misura di distanza
	Bordo destro		256 mm	455 mm	0.748 mm
8	centro	49.6°	377 mm	315 mm	0.243 mm
	Bordo sinistro		1495 mm	174 mm	0.019mm

Tabella 3.7: Risoluzione della misura di profondità con $\alpha = 45°$ considerando la configurazione proposta in Figura (3.3).

A seguito di questa analisi teorica si è dimostrato che: il sistema è in grado di raggiungere e superare le specifiche richieste riguardo alla risoluzione della misura di distanza ma solo per la zona del bordo vicino alla videocamera e con angolazione favorevole. La scansione realizzata tramite la rotazione del laser porta ad avere diverse risoluzioni a seconda della zona illuminata; mentre una scansione lineare che trasli solidalmente telecamera e laser lungo un piano parallelo al cerchione la manterrebbe costante. Le principali problematiche riscontrate nella fase progettuale sono legate alla difficoltà di ottenere misure della superficie 3D con la risoluzione richiesta e nei tempi richiesti, difatti la possibilità di mappare l'intera superficie del cerchione in soli 30 s con una risoluzione di misura inferiore a 0.2mm non è possibile con i componenti e la configurazione proposta. Dai dati riportati in tabella si può notare che aumentando l'angolo α è possibile ottenere una migliore risoluzione della misura di distanza, aumentando però anche la probabilità di occlusioni: bisogna considerare i possibili ostacoli alla linea laser dovuti alla superficie complessa e frastagliata del cerchione; in aggiunta si fanno presenti anche le problematiche legate alla profondità di campo: se il profilo misurato è lontano dal punto di messa a fuoco della videocamera la linea laser risulta meno definita. Infine la risoluzione dei sistemi basati sulla triangolazione δz,come si deduce dalla formula (3.5), è direttamente proporzionale al quadrato della distanza (z), perciò per migliorare le prestazioni bisogna diminuire la distanza dall'oggetto (z_{work}), utilizzando ottiche particolarmente corte, che possono però introdurre rilevanti distorsioni, oppure rinunciando ad osservare l'intero profilo del cerchione e di conseguenza aumentando il tempo necessario per acquisire l'intero modello 3D del cerchione. Per realizzare le specifiche richieste occorre scansionare in più sessioni il profilo: conclusa una scansione parziale, muovere il braccio robotico verso il punto successivo lungo il cerchione e realizzare la nuova scansione.

Capitolo 4

Realizzazione del setup sperimentale

Nel seguente capitolo si descrive lo strumento effettivamente realizzato presso il Laboratorio di Meccatronica e Robotica per l'Innovazione (*MERLIN*) nella sede del Dipartimento di Elettronica, Informazione e Bioingegneria (*DEIB*) del Politecnico di Milano. La ricerca che ha portato alla realizzazione del prototipo è stata finanziata dalla Comunità Europea grazie al programma FP7/2007-2013 nell'ambito del progetto FIDELIO (*FIxtureless DEburring of wheeLs by human demonstratIOn*).

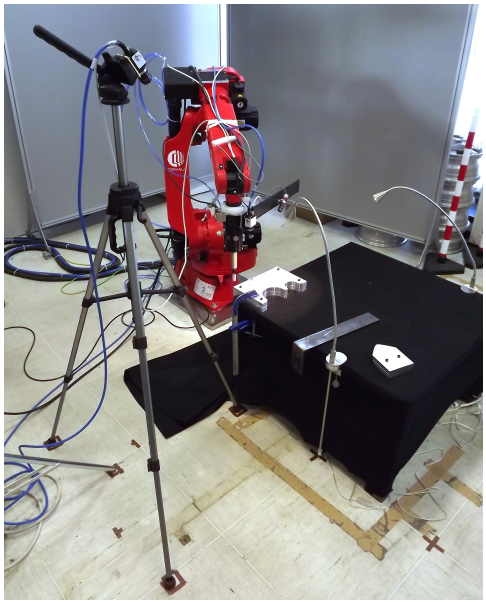


Figura 4.1: Setup sperimentale FIDELIO [22]

Lo scopo della ricerca, come descritto in precedenza, è di definire il

contorno dell'oggetto, in particolare un cerchione automobilistico di forma non nota, da sottoporre a sbavatura, e di rilevare eventuali bave identificandole nello spazio operativo in modo da poter procedere alla loro rimozione. I componenti che sono stati utilizzati per realizzare il setup sperimentale sono i seguenti:

- robot Smart Six a 6 gradi di libertà, 6 kg di payload, manipolatore industriale prodotto da COMAU, equipaggiato con la versione aperta del controllore COMAU C4G;
- PC real-time, basato sull'estensione RTAI Linux real-time, che esegue il software di controllo ed è collegato al controllore C4G per mezzo di una connessione Ethernet che implementa il protocollo RTnet;
- PC non real-time, con sistema operativo Linux, implementa il software di *image processing* ed è connesso al PC real-time per mezzo di socket Ethernet standard;
- utensile di sbavatura ATI Flexdeburr FDB-340, con cedevolezza radiale per manipolatore robotico, montato sulla flangia del sensore di forza/coppia collocato sull'end effector del manipolatore;
- il sistema a triangolazione vero e proprio, composto da una sbarra rigida fissata all'end-effector del robot su cui è posizionato il proiettore a linea laser STOCKERYALE/COHERENT LASIRIS SNF 660nm 10mW, con un angolo di proiezione di 10°, ed una videocamera digitale GigE uEye UI-5240SE, con ottica COSTAR da 25mm di lunghezza focale nominale;
- una seconda videocamera digitale, GigE uEye UI-5240SE, con ottica COSTAR da 16mm di lunghezza focale nominale, montata su un treppiede lontano dallo spazio operativo del robot.

L'utilizzo di un manipolatore robotico antropomorfo a sei gradi di libertà consente di posizionare l'utensile di sbavatura con la giusta angolatura a ridosso del profilo del pezzo e di eseguire l'operazione di sbavatura. Una particolare attenzione è stata spesa nella selezione dei componenti del sistema a triangolazione laser. Il proiettore laser e le lenti delle camere sono state scelte con lo scopo di assicurare la risoluzione di misura richiesta (meno di 0.2 mm), e permettendo all'intero sistema di operare nell'intorno della distanza desiderata z_{work} pari a 195 mm, dalla superficie del pezzo di lavorazione, in modo da corrispondere con le caratteristiche dell'utensile di sbavatura.
La configurazione scelta per lo strumento di misura di distanza è quella *ordinaria*, con la camera fissata all'end-effector in posizione centrale rispetto all'oggetto da misurare e posizionata il più possibile parallela al piano di appoggio dell'oggetto, il proiettore laser invece è posizionato lateralmente alla camera formando un angolo α di circa 48.5° con il piano y-z della terna

camera, e fissato ad un apposito sostegno che collega rigidamente quest'ultimo, la videocamera e l'end-effector del manipolatore (vedi figura 4.2). Si è scelto di utilizzare questa configurazione per ridurre gli effetti prospettici[1]

Tecnologia sensore	CMOS
Risoluzione	1280×1024
Risoluzione di profondità bit	10 bit
Rimensione sensore	$1/1.8''$
Dimensione dell'ottica	6.784×5.427 mm
Spazio fra un pixel e quello affianco	$5.3 \times 5.3\,\mu$m
Frame rate	50 fps

Tabella 4.1: GigE eUeye UI-5240SE specifiche tecniche della camera.

Prima di poter realizzare le misurazioni volute, è necessario svolgere alcune operazioni di calibrazione che permettono di approssimare al meglio le relazioni che intercorrono fra le coordinate dei punti rilevati, sul piano immagine (ξ_x, ξ_y), con quelle degli stessi punti nella terna di riferimento del robot (terna mondo x, y, z). Le procedure ed i metodi utilizzati per la calibrazione della camera *eye-to-hand* posizionata sul treppiede, della camera *eye-in-hand* fissata all'end-effector e del sistema di triangolazione con linea laser verranno spiegati in dettaglio nei successivi capitoli (vedi capitolo 6). Dopo aver svolto le procedure di calibrazione, determinando i parametri geometrici che caratterizzano il piano luminoso generato dal proiettore laser nei confronti della terna camera, la linea laser può essere sfruttata per recuperare l'informazione sulla terza dimensione che, inevitabilmente, è stata persa nelle immagini della camera a causa della proiezione prospettica. Osservando la figura (4.2) che rappresenta il modello di camera prospettica e la configurazione del sistema sperimentale, si può applicare la seguente formula di triangolazione:

$$z = \frac{d \cdot f}{d_{o_c} + tan(\alpha) \cdot f} \tag{4.1}$$

dove d e α rappresentano la posizione e l'orientamento del piano laser rispetto alla terna camera, rispettivamente, f la lunghezza focale nominale delle lenti della videocamera, e d_{o_c} la distanza sul piano immagine fra il punto luminoso e la proiezione del centro ottico. La misura di distanza d_{o_c} può essere definita

[1]Come si può notare la scelta della configurazione è diversa da quella presentata nel paragrafo (3.3), dopo aver determinato da risultati teorici che le richieste in termini di risoluzione e velocità di scansione non erano raggiungibili con i componenti reperibili, vedi pararagrafo (3.5), allora si è scelto di abbandonare l'idea di mappare interamente il profilo senza movimentare il manipolatore, ma anzi di sfruttarlo implementando un controllo ad inseguimento del contorno per agevolare la misurazione con il sistema a triangolazione; di conseguenza il componente di movimentazione del laser (stepper angolare) non è stato più necessario, ed inoltre la configurazione ordinaria rende più semplice il disaccoppiamento delle componenti x y z sfruttate dal sistema di controllo visuale.

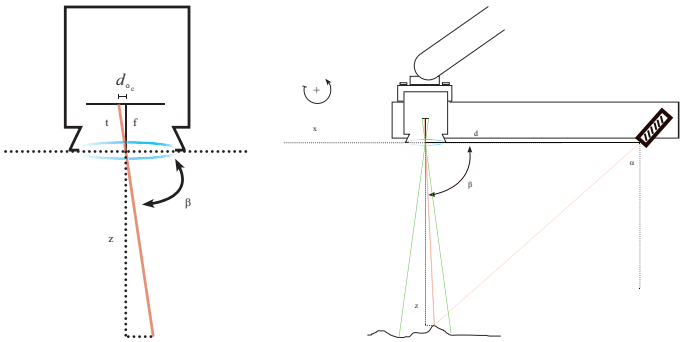


Figura 4.2: Schema del modello di camera *pin-hole* (immagine a sinistra) e disegno della configurazione adottata nello strumento sperimentale (immagine a destra).

con la seguente formula $d_{o_c} = -\sigma(\xi_x - c_x)$ ricavata dal modello di camera prospettica con la conversione della misura in pixel (ξ_x). In seguito assumiamo che il laser, una volta calibrato proietti una linea luminosa che sia parallela all'asse y della terna camera. Mantenendo questa assunzione, la distanza d_{o_c} può essere calcolata come una distanza dalla linea luminosa alla colonna corrispondente al centro ottico. Sfruttando la formula di triangolazione è possibile ricavare i valori teorici del campo visivo della videocamera e la risoluzione del sistema a triangolazione. Assumendo che la distanza di lavoro z_{work} sia pari a 195 mm, il valore di d sia 220 mm, ed utilizzando il valore nominale della focale f pari a 25 mm, mentre il valore di σ, pari ad 1 pixel, si ricava dalle caratteristiche della camera, ed è pari a 5.3 μm, si ottiene che l'intervallo di misura di distanza sia pari a 47.6 mm, precisamente da z_{min} = 174 mm a z_{max} = 221.6 mm misurato in terna camera (vedi figura 4.3). Il campo visivo può essere invece calcolato grazie alla formula (3.2).

$$\varphi = 2\ \tan^{-1}\left(\frac{6.784\,\text{mm}}{2 \cdot 25\,\text{mm}}\right) = 15.45° \tag{4.2}$$

Infine la risoluzione della misura di distanza varia in funzione della distanza a cui effettuiamo la misura, per mezzo della formula (3.5), che tralasciando il segno diventa

$$\delta z = \frac{\sigma \cdot z^2}{d \cdot f} \tag{4.3}$$

ottenendo $\delta z_{min} = 29\,\mu$m e δz_{max} 47 μm. Per quanto riguarda la risoluzione δx e δy questa può essere calcolata come segue, ipotizzando di misurare punti su di un piano parallelo al piano x-y della terna camera:

$$\delta y = \frac{z \cdot \sigma}{f} = \frac{z_{work} \cdot \sigma}{f} = 41\mu m \tag{4.4}$$

$$max\left(\delta x = \frac{d_{work} \cdot \sigma}{f} = 41\mu m, \delta x = vel_x / F_{frame\_rate}\right) \tag{4.5}$$

Riassumendo il sistema a triangolazione qui descritto permette di soddisfare le specifiche richieste, la risoluzione risulta inferiore a 50 μm nel campo di operatività dello strumento, utilizzando una videocamera un proiettore laser ed il manipolatore robotico, sfruttato per la movimentazione della linea laser. Sia per la fase di calibrazione che per la fase sperimentale di misurazione e sbavatura, sono stati realizzati oggetti di test; per questa seconda fase è stato utilizzato un manufatto in alluminio a forma di parallelepipedo scavato con profili curvilinei di misura nota, a questo oggetto sono state applicate bave artificiali di forma non nota a priori. Una seconda camera fissa permette di monitorare il processo dall'esterno, consente di effettuare la fase di approccio al pezzo che può essere fissato in una qualunque posizione del piano di lavoro, il sistema di misurazione viene così posizionato in un intorno della distanza desiderata ed è possibile avviare gli algoritmi di *image analysis* e *contour following*.

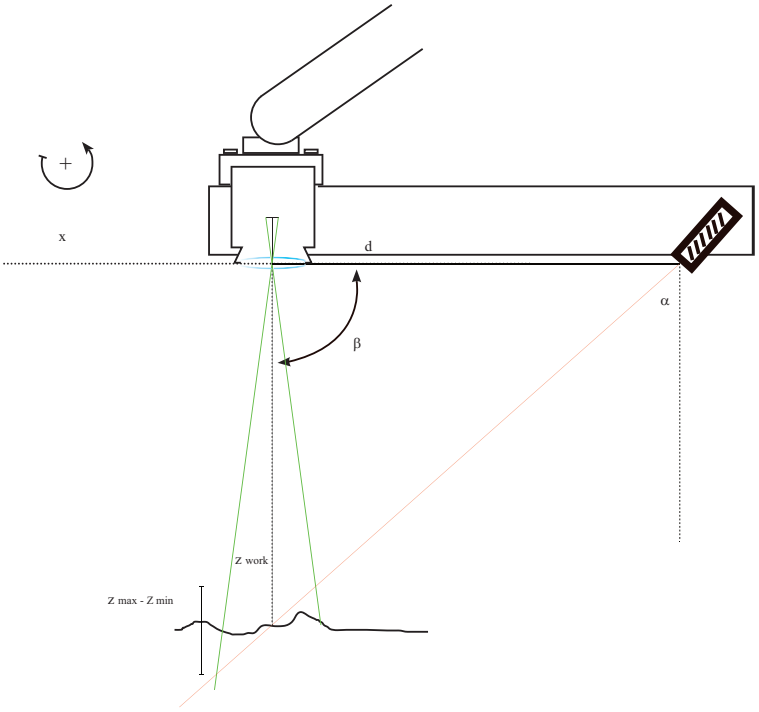


Figura 4.3: Schema intervallo di misura strumento a triangolazione.

4.1 Realizzazione dei supporti per videocamera e laser

Per poter fissare la videocamera ed il proiettore all'end-effector sono stati realizzati dapprima dei supporti provvisori (vedi figura 4.4), in questo modo è stato possibile sperimentare l'effettivo campo visivo dello strumento e l'effettivo ingombro. Un aspetto da tenere in considerazione è la rigidezza della struttura, difatti il supporto che collega la videocamera ed il proiettore laser fra di loro, ed al manipolatore, deve essere sufficientemente rigido per non influenzare la bontà delle misure, bisogna aggiungere però che con le velocità ed accelerazioni con cui è stato sperimentato il sistema di misura non si sono verificate problematiche legate ad oscillazioni della struttura.

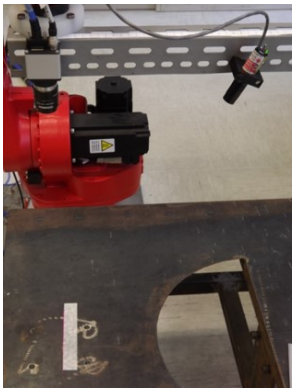


Figura 4.4: Prototipo sistema di misura: dettaglio camera eye-in-hand

Dopo aver terminato la fase prototipale è stato realizzato un supporto *ad hoc* per la videocamera e per il laser che potesse essere rimosso ed installato unitamente all'utensile di sbavatura, vedi figura (4.5)

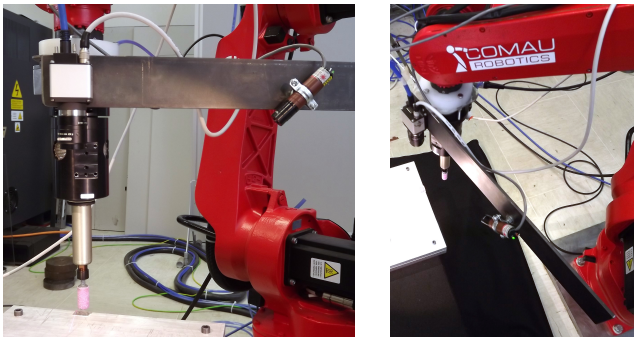


Figura 4.5: Supporto del sistema di triangolazione ed utensile di sbavatura [22].

4.2 Librerie software utilizzate

Il linguaggio di programmazione principalmente utilizzato è stato il C/C++, gli algoritmi di *image analysis* e di interfaccia con le videocamere sono stati sviluppati in questo linguaggio, ed in ambiente Linux su un primo computer, mentre gli algoritmi di controllo del manipolatore robotico, sono stati implementati sempre in questo linguaggio a bordo di un secondo computer real-time, inoltre è stato implementato anche il codice necessario per la comunicazione fra i due computer nel medesimo linguaggio.

Delle librerie largamente utilizzate dalla comunità di sviluppatori, e continuamente aggiornate da essi, sono state diffuse su internet per realizzare compiti nel campo della *computer vision* e della manipolazione delle immagini, una delle più importanti fra queste è sicuramente OpenCV [24]. Questa libreria oltre ad avere svariato materiale informativo e manuali divulgativi appositi è completamente *open source*. Dopo aver testato ed implementato i primi algoritmi sulla libreria commerciale Halcon si è deciso di non investire ulteriormente tempo e denaro su tale libreria e di implementare tutto il codice sfruttando la libreria OpenCV. Tale libreria, anche se meno intuitiva di quella commerciale, offre il vantaggio di poter modificare *ad hoc* tutte le funzioni già implementate e di controllare passo passo gli algoritmi implementati. Per queste ragioni, ai fini accademici e pratici sopra citati, si è scelto di riprogrammare gli algoritmi già sviluppati sulla nuova libreria.

Altre librerie sfruttate per l'implementazione del sistema, oltre ad OpenCV sono state: highgui.h (per l'interfaccia grafica) ed ueye.h (per l'acquisizione di immagini dalle fotocamere ed il settaggio delle stesse).

L'implementazione degli algoritmi di *image processing*, è necessaria per l'identificazione della posizione della linea laser e del bordo dell'oggetto sul piano immagine, in modo da poter trasferire l'informazione sulle coordinate dei punti in terna camera al computer *real-time* che elabora le variabili di controllo da fornire al controllore del manipolatore robotico; vengono inoltre registrate le immagini acquisite dalla videocamera ed i valori di posizione ed orientamento dell'end effector, per una successiva analisi *off-line* dei dati raccolti. Per l'analisi di questi dati ed il calcolo numerico infine è stato utilizzato il software MATLAB di MathWorks, combinato con il pacchetto software di computer vision *Camera Calibration Toolbox* di Jean-Yves Bouguet. Le funzionalità di quest'ultimo *Toolbox* sono state sfruttate sia durante la procedura di calibrazione che per determinare i parametri di distorsione delle lenti delle videocamere.

Infine sono stati utilizzati i software CalLab e CalDe per confrontare differenti metodi di calibrazione delle videocamere; questa *suite* di programmi è stata sviluppata espressamente per la calibrazione di videocamere, presso l'istituto di robotica e meccatronica del centro aerospaziale tedesco (DLR).

4.3 Posizionamento laser e movimentazione

Il posizionamento del laser ad una distanza d lungo l'asse x della terna camera, influenza la risoluzione delle misure di distanza effettuate dallo strumento a triangolazione, per questo motivo esso è fissato in un alloggiamento sul supporto rigido appositamente costruito, posto lateralmente alla videocamera *eye-in-hand*, e dopo essere stato orientato nella fase di calibrazione viene fissato rigidamente con delle viti. La distanza dal centro ottico della videocamera è calcolata per soddisfare le richieste di risoluzione pur tenendo in considerazione che un angolo α troppo grande comporterebbe maggiori problematiche nel rilevamento della linea laser, dato che la superficie complessa e frastagliata del cerchione automobilistico genererebbe delle occlusioni. L'angolo α è scelto maggiore di 45° in accordo con le considerazioni fatte in precedenza (vedi tabella 3.7).

Per poter misurare le coordinate 3D del profilo di sbavatura del pezzo, la linea laser deve scorrere lungo l'intera superficie di interesse in modo da illuminare tutti i punti di cui vogliamo sapere la posizione. Per far questo la movimentazione si realizza direttamente tramite il manipolatore robotico, effettuando delle traslazioni comandate dal sistema di controllo lungo la direzione di avanzamento della scansione, in questo caso data la configurazione del sistema lungo la direzione definita dall'asse x della terna camera.

In alcuni esperimenti si sono effettuate delle misurazioni movimentando il manipolatore manualmente con velocità costante e senza cambiare l'orientamento dell'end-effector, ma per poter facilitare l'acquisizione dell'intero profilo di sbavatura mantenendo la telecamera nella corretta posizione di acquisizione è stato implementato un controllo visuale ad inseguimento del contorno. Gli algoritmi di controllo del manipolatore sono stati sviluppati principalmente a scopo strumentale del sistema di scansione laser, in modo da evitare che l'utente debba operare manualmente, ma soprattutto per mantenere la zona di interesse, il contorno dell'oggetto, al centro del campo visivo della videocamera ed alla giusta distanza da essa; i motivi per mantenere la videocamera posizionata il più possibile sul bordo dell'oggetto sono principalmente legati al fatto che, nella zona centrale dell'immagine, gli effetti di distorsione delle lenti sono minori, ed è inoltre più immediata la misurazione delle caratteristiche delle bave rilevate; mentre il motivo per cui è importante controllare la distanza di lavoro a cui si effettuano le misurazioni è che, utilizzando una lente a focale fissa, la definizione dell'immagine peggiora allontanandosi dal punto di messa a fuoco e l'immagine della linea laser e dei bordi dell'oggetto risulterebbero più sfuocati comportando delle misure meno precise.

Deve essere detto che lo scopo del sistema di analisi delle immagini è quello di misurare, con una elevata risoluzione, il profilo di sbavatura, che

Figura 4.6: Immagine acquisita durante la misurazione del profilo di sbavatura.

può essere di forma non nota a priori, di un oggetto di dimensioni anche maggiori a 0.5 m. Difatti la videocamera fissa posta sul treppiede permette di visualizzare per intero l'oggetto in lavorazione e di identificarne la posa nella spazio operativo, questo consente al sistema di triangolazione di approcciarsi al pezzo senza avere limitazioni di spazio operativo se non quelle dettate dalla cinematica del robot. Nel seguente paragrafo si descrive il funzionamento del controllo visuale implementato.

4.3.1 Contour Following: controllo visuale ad inseguimento del profilo

Il controllo visuale nello spazio delle immagini permette di mantenere il centro dell'immagine il più possibile posizionato sul bordo del profilo del pezzo. Per essere più precisi, assumendo che il profilo giaccia su di un piano parallelo al piano immagine[2]. Gli obbiettivi di tale controllo sono duplici:

- mantenere il piano immagine ad una distanza (costante) desiderata dal piano del profilo scansionato, da misurarsi lungo lasse z della terna camera;
- regolare la posizione dell'end-effector, e dei 6 giunti del manipolatore in modo tale che l'asse ottico della camera montata sull'end-effector intersechi sempre il profilo e che uno degli assi della terna camera sia tangente al profilo.

Il problema descritto può essere facilmente riformulato nel contesto di controllo partizionato visuale [8]. In questo modo si può facilmente disaccoppiare il

[2]Se il piano immagine ed il profilo da sbavare non dovessero essere paralleli al piano immagine, il controllo visuale qui presentato può essere esteso di modo che l'orientamento dell'end-effector venga controllato in base alla normale al profilo di sbavatura.

controllo della posizione x-y sul piano immagine dal controllo lungo l'asse ottico e dal controllo dell'orientamento della camera. Ne risulta, perciò, una corrispondente partizione delle variabili di controllo: la velocità lineare della camera nel piano immagine (v_{c_x} e v_{c_y}) è usata per controllare la posizione nel piano x-y, la velocità angolare attorno all'asse ottico (ω_{c_z}) per regolare l'orientamento della camera, e la velocità della camera lungo l'asse ottico (v_{c_z}) per mantenere la distanza desiderata dal profilo. Seguendo [8] possiamo

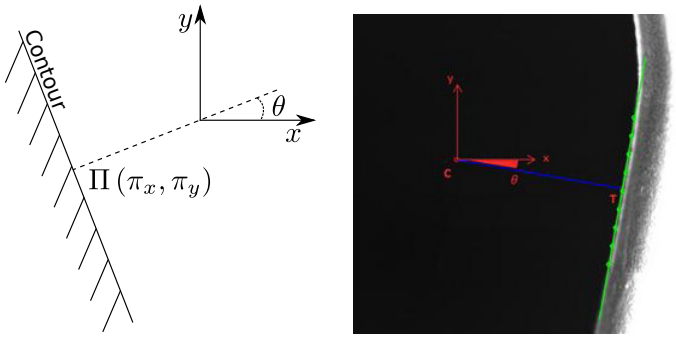


Figura 4.7: Caratteristiche immagine usate durante il controllo visuale ad inseguimento del contorno *contour following* [22].

introdurre la dinamica delle features

$$\dot{s_\pi} = L\left(s_\pi\right) v_c \tag{4.6}$$

dove $s_\pi = \left[s_{\pi_x}, s_{\pi_y}\right]^T$ sono le coordinate in pixel di un punto Π appartenente al profilo di sbavatura scansionato (figura 4.7), $L \in \mathbb{R}^{2\times 6}$ è la matrice di interazione in relazione con s_π e v_c è la velocità della terna camera

$$v_c = \begin{bmatrix} v_{c_x} \\ v_{c_y} \\ v_{c_z} \\ \omega_{c_x} \\ \omega_{c_y} \\ \omega_{c_z} \end{bmatrix}$$

Considerando ora i due obbiettivi affermati in precedenza per il controllore visuale, la matrice di interazione sarà partizionata nel modo seguente

$$L = \left[\ L_{v_{xy}} \ \middle|\ L_{v_z} \ \middle|\ L_{\omega_{xy}} \ \middle|\ L_{\omega_z}\ \right]$$

e l'equazione (4.6) diventa

$$\dot{s}_\pi = L_{v_{xy}} \begin{bmatrix} v_{c_x} \\ v_{c_y} \end{bmatrix} + L_{v_z} v_{c_z} + L_{\omega_{xy}} \begin{bmatrix} \omega_{c_x} \\ \omega_{c_y} \end{bmatrix} + L_{\omega_z} \omega_{c_z} \tag{4.7}$$

introducendo ora l'errore

$$e(t) = s_\pi(t) - s_\pi^* \tag{4.8}$$

dove s_π^* rappresenta il valore desiderato del vettore delle features, si può facilmente derivare la relazione fra la velocità della camera e la variazione nel tempo dell'errore

$$\dot{e} = \dot{s}_\pi = L(s_\pi)\, v_c$$

Infine, imponendo una decrescita esponenziale dell'errore delle features ($\dot{e} = -\lambda e$), ed inserendo l'equazione (4.7) nella (4.8), l'espressione delle due variabili di controllo diventa

$$\begin{bmatrix} v_{c_x} \\ v_{c_y} \end{bmatrix} = -L_{v_{xy}}^{-1} \left(\lambda e + L_{v_z} v_{c_z} + L_{\omega_z} \omega_{c_z}\right)$$

dove il primo termine è un contributo dovuto all'errore delle features, l'altro termine é una compensazione introdotta per ridurre, il più possibile, gli effetti degli altri anelli di controllo (posizione lungo l'asse ottico e orientamento della camera) sull'errore della feature. Seguendo adesso l'assunzione iniziale, che il profilo scansionato giaccia su un piano parallelo al piano immagine, la velocità angolare lungo l'asse x e y, ω_{c_x} e ω_{c_y}, possono essere trascurate. Invece, ω_{c_z} e v_{c_z} sono calcolate mediante leggi di controllo proporzionale basate sull'errore dell'orientamento[3], $e_\theta(t) = \theta^* - \theta(t)$ e l'errore di distanza $e_z(t) = z^* - z(t)$ (θ^* e z^* essendo i valori desiderati di $\theta(t)$ e $z(t)$, rispettivamente).
Riassumendo, il controllo visuale è basato sulle seguenti equazioni

$$\begin{bmatrix} v_{c_x} \\ v_{c_y} \end{bmatrix} = -L_{v_{xy}}^{-1} \left[\lambda (s_\pi - s_\pi^*) + L_{v_z} v_{c_z} + L_{\omega_z} \omega_{c_z}\right] \tag{4.9}$$

$$v_{c_z} = \lambda_{v_z} (z^* - z(t)) \tag{4.10}$$

$$\omega_{c_z} = \lambda_{\omega_z} (\theta^* - \theta(t)) \tag{4.11}$$

dove λ_{v_z} e λ_{ω_z} sono due guadagni proporzionali.

Allo scopo di implementare questa legge di controllo, deve essere determinata l'espressione analitica della matrice di interazione.
Differenziando il modello di camera Pin-hole rispetto al tempo, si ottiene la seguente relazione fra le velocità nello spazio 3D e le velocità nel piano immagine

$$\begin{bmatrix} \dot{\xi}_x \\ \dot{\xi}_y \end{bmatrix} = \begin{bmatrix} \frac{f_x}{{}^c z} & 0 & -\frac{\xi_x}{{}^c z} \\ 0 & \frac{f_y}{{}^c z} & -\frac{\xi_y}{{}^c z} \end{bmatrix} \begin{bmatrix} {}^c\dot{x} \\ {}^c\dot{y} \\ {}^c\dot{z} \end{bmatrix} \tag{4.12}$$

Inoltre, introducendo la seguente relazione cinematica

$${}^o p = {}^o R_c \, {}^c p + {}^o T_c$$

[3]Qui θ è l'angolo fra la linea che collega il punto Π ed il centro ottico, e l'asse x della terna immagine (figura 4.7).

dove ${}^{o}p$ e ${}^{c}p$ sono le posizioni del punto 3D in terna mondo e rispetto alla terna camera, ${}^{o}T_c$ e ${}^{o}R_c$ sono la posizione e l'orientamento della terna camera rispetto alla terna mondo. La derivata nel tempo della relazione precedente invece è la seguente

$$ {}^{o}\dot{p} = {}^{o}R_c\, {}^{c}\dot{p} + {}^{o}\omega_c \times {}^{o}R_c\, {}^{c}p + {}^{o}\dot{T}_c $$

Si può riscrivere l'equazione (4.12) facendo riferimento alla velocità del punto rispetto alla terna mondo.
Considerando ora che l'oggetto sia stazionario rispetto alla terna mondo (${}^{o}\dot{p} = 0$), denotando con v_c e ω_c le velocità della camera lineare e angolare rispetto alla terna camera, la velocità del punto 3D rispetto alla terna camera è data da

$$ {}^{c}\dot{p} = -v_c - \omega_c \times {}^{c}p $$

O equivalentemente

$$ \begin{aligned} {}^{c}\dot{x} &= -v_{c_x} - \omega_{c_y}\, {}^{c}z + \omega_{c_z}\, {}^{c}y \\ {}^{c}\dot{y} &= -v_{c_y} - \omega_{c_z}\, {}^{c}x + \omega_{c_x}\, {}^{c}z \\ {}^{c}\dot{z} &= -v_{c_z} - \omega_{c_x}\, {}^{c}y + \omega_{c_y}\, {}^{c}x \end{aligned} \tag{4.13} $$

Infine, inserendo l'equazione (4.13) nell'equazione (4.12), e tenendo conto del modello della camera (3.18), si ottiene l'espressione della matrice di interazione per il punto feature

$$ \begin{bmatrix} \dot{\xi}_x \\ \dot{\xi}_y \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{f_x}{{}^{c}z} & 0 & \frac{\xi_x}{{}^{c}z} & \frac{\xi_x \xi_y}{f_y} & -\frac{\xi_x^2 + f_x^2}{f_x} & \xi_y \frac{f_x}{f_y} \\ 0 & -\frac{f_y}{{}^{c}z} & \frac{\xi_y}{{}^{c}z} & \frac{\xi_y^2 + f_y^2}{f_y} & -\frac{\xi_x \xi_y}{f_x} & -\xi_x \frac{f_y}{f_x} \end{bmatrix}}_{L} \begin{bmatrix} v_{c_x} \\ v_{c_y} \\ v_{c_z} \\ \omega_{c_x} \\ \omega_{c_y} \\ \omega_{c_z} \end{bmatrix} $$

Sulla base di questa espressione, le equazioni del controllo visuale in (4.9) possono essere riscritte come segue

$$ \begin{aligned} v_{c_x} &= \frac{{}^{c}z\lambda}{f_x} e_x + \frac{{}^{c}z \xi_y}{f_y} \omega_{c_z} + \frac{\xi_x}{f_x} v_{c_z} \\ v_{c_y} &= \frac{{}^{c}z\lambda}{f_y} e_y - \frac{{}^{c}z \xi_x}{f_x} \omega_{c_z} + \frac{\xi_y}{f_y} v_{c_z} \\ v_{c_z} &= \lambda_{v_z} \left(z^* - z\left(t\right) \right) \\ \omega_{c_z} &= \lambda_{\omega_z} \left(\theta^* - \theta\left(t\right) \right) \end{aligned} $$

dove $e_x\left(t\right) = s_{\pi_x}\left(t\right) - s^*_{\pi_x}$ e $e_y\left(t\right) = s_{\pi_y}\left(t\right) - s^*_{\pi_y}$.

Inoltre, per poter implementare questo sistema, sul manipolatore real-time le velocità lineari e angolari dell'end-effector devono essere determinate, usando le seguenti relazioni

$$ {}^{o}v_e = {}^{o}R_e\, {}^{e}R_c\, {}^{c}v - {}^{o}\omega_e \times {}^{o}R_e\, {}^{e}T_c \qquad {}^{o}\omega_e = {}^{o}R_e\, {}^{e}R_c\, {}^{c}\omega $$

dove ${}^{o}R_e$ e ${}^{o}\omega_e$ sono l'orientamento e la velocità angolare assoluta dell'end-effector, rispettivamente; mentre ${}^{e}T_c$ e ${}^{e}R_c$ sono posizione ed orientamento della terna camera rispetto alla terna dell'end-effector.

La velocità di avanzamento dello strumento, imposta lungo l'asse x risulta invece essere pari a δv_{c_x} con opportuno segno, ed è una piccola quantità che viene aggiunta ai segnali di controllo nel momento in cui il segnare di errore rimane sotto una certa soglia; la caratteristica distintiva di tutti questi algoritmi è che sono in grado di garantire una accuratezza nella caratterizzazione del profilo, con un errore inferiore a 0.5 mm.

Per concludere la descrizione del controllo ad inseguimento del contorno e del sistema a triangolazione, la figura 4.8, mostra l'architettura del controllore e chiarifica l'interazione fra i due sottosistemi. Infatti, come può essere visto dall'immagine, la parte del controllo visuale che mantiene la profondità ${}^{c}z$ costante è alimentata dalle misurazioni di profondità fornite dal sistema di triangolazione, mentre le coordinate x e y sono regolate direttamente nello spazio delle immagini.

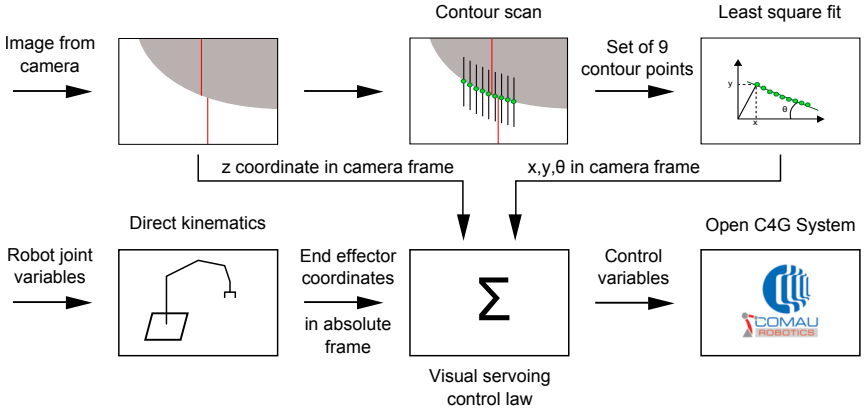


Figura 4.8: Descrizione schematica del sistema di controllo visuale e stima del profilo del pezzo [22].

Capitolo 5

Algoritmi di image analysis: LASER LINE DETECTION

Il seguente capitolo descrive gli algoritmi sviluppati per analizzare le immagini acquisite dal sistema a triangolazione. Si descrivono i programmi sviluppati in linguaggio C ed utilizzati durante le fasi di sperimentazione del sistema di triangolazione laser, il primo obiettivo è stato quello di localizzare la linea laser proiettata sul pezzo in lavorazione, sia esso un manufatto metallico appositamente preparato per la fase sperimentale sia un cerchione automobilistico in lega metallica; il rilevamento della linea laser deve essere affidabile ed il più possibile insensibile alle condizioni di luce ambientali, oltre che non condizionabile dalla colorazione e dalla finitura superficiale del pezzo da analizzare.
Un secondo obiettivo è stato imposto dalla sperimentazione effettuata sfruttando il controllo *contour following* (vedi paragrafo 4.3.1) è stato quello di identificare il punto della linea laser corrispondente al bordo del pezzo, in modo da poter ricostruire il profilo di sbavatura, individuando proprio tale bordo come quello maggiormente soggetto alla formazione di bave. Per questa specifica applicazione la velocità dell'elaborazione dell'immagine e la rapidità computazionale sono state caratteristiche importanti da tenere in considerazione nella scelta dell'algoritmo da utilizzare. Per la realizzazione degli obbiettivi sono stati sviluppati diversi algoritmi con l'ausilio di tecniche di *image processing*, ottenute sfruttando alcune funzioni della libreria OpenCV.

Fin dalle prime misurazioni, effettuate su di una superficie planare con immagini in scala di grigio, si è notato come l'illuminazione dell'ambiente influisse sulla rumorosità dell'immagine acquisita dalla videocamera; la presenza di luce pulsata generata da lampade a soffitto condiziona fortemente la precisione della misura, mentre la presenza di una forte illuminazione solare può portare alla saturazione dell'immagine e ad errori di misura, per questo motivo si sono sfruttati alcuni accorgimenti al fine di ridurre l'instabilità luminosa e poter controllare meglio l'ambiente di misura. Il metodo più immediato

per ridurre la rumorosità dell'immagine è quello di filtrare l'immagine con un processo di *smoothing*, l'immagine originale viene mediata con un filtro a media mobile, che genera una immagine più sfuocata di quella originale, ma con variazione dei livelli di luminosità meno accentuata; il filtro a media mobile maggiormente utilizzato è il filtro gaussiano, esso opera sostituendo al valore di luminosità di ogni singolo pixel quello ottenuto dalla media dei valori dei pixel vicini, la media effettuata è pesata con valori che simulano l'andamento di una curva gaussiana tridimensionale; i metodi di *smoothing* rendono l'immagine della linea laser poco definita e sono di difficile utilizzo pratico nel rilevamento della linea laser affetta da disturbi di tipo *speckle*[1] o riflessioni speculari, mentre negli algoritmi sviluppati per il rilevamento del bordo dell'oggetto, sfruttati per il *contour following* sono stati utilizzati efficacemente. Per migliorare la misurazione della linea laser, si è adottato un *frame rate* di 50fps, questa scelta infatti permette di acquisire le immagini alla stessa frequenza nominale della rete elettrica e di ridurre l'effetto di disturbo luminoso generato dalle lampade al neon. La condizione ottimale per la misurazione tramite laser scanner si verifica con l'assenza di illuminazione, ad esclusione della luce laser, ma questa condizione non è attuabile nella fase di sperimentazione con *contour following*, perciò per poter eseguire le misurazioni con il setup proposto, è opportuno schermare la cella di lavoro del manipolatore dalla luce solare e predisporre una adeguata illuminazione a led a bordo del piano di lavoro, disposte con accortezza al fine di evitare ombre.

5.1 Algoritmo di base

Un primo algoritmo, denominato "Algoritmo di base", individua il valore di massima luminosità di una striscia orizzontale in cui viene partizionata l'immagine. Dopo aver svolto la procedura di calibrazione, si ipotizza che la linea laser sia rilevabile come una linea luminosa verticale se la superficie dell'oggetto è parallela al piano x-y della terna posizionata sull'end-effector, inoltre la distanza della linea dalla colonna centrale indica quanto l'oggetto sia distante dalla distanza di lavoro z_{work}. Per questo motivo la ricerca della linea luminosa viene fatta analizzando una riga orizzontale di pixel per volta, l'area analizzata durante la ricerca della linea laser è detta ROI, regione di interesse, e la sua dimensione può essere ridotta per non dover analizzare l'intera riga di pixel dell'immagine e rendere più rapido l'algoritmo. Il funzionamento dell'algoritmo è semplice: procedendo dal bordo sinistro della ROI si analizza il valore di intensità luminosa di ogni pixel (gray values,

[1] l'effetto di *speckle* si verifica quando una superficie ruvida (non speculare), viene illuminata con una sorgente di luce coerente (sorgente laser monocromatica). La superficie illuminata appare granulosa, presenta tante macchie illuminate accostate a macchie non illuminate, questo fenomeno è dovuto all'interferenza della luce diffusa dall'oggetto[1]

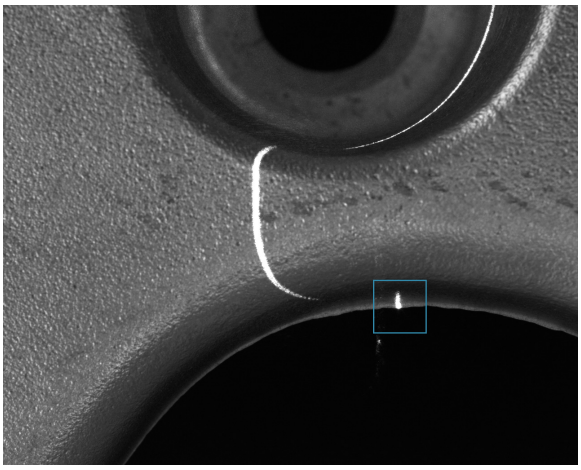


Figura 5.1: Esempio di misurazione del profilo di un cerchione in lega tramite linea laser

0 significa assenza di luce, 255 massima luminosità), i valori con il valore massimo individuati vengono registrati in un array e una volta completata l'operazione sull'intera riga della ROI, si seleziona il valore mediano fra quelli registrati nell'array, in questo modo la posizione della linea laser si identifica con il punto mediano fra quelli più luminosi senza essere condizionati da isolati punti luminosi.

5.2 Algoritmo secondario

Un secondo algoritmo, denominato "Algoritmo secondario", si basa sull'individuazione delle soglie massime, positive e negative, di variazione luminosa, individuando il punto di interesse a metà di esse. Il secondo algoritmo sviluppato richiede l'utilizzo di elaborazioni grafiche effettuate sull'immagine di partenza, con opportuni filtraggi si ottengono le immagini che rappresentano la variazione luminosa lungo una direzione imposta e si ottiene anche una minore rumorosità delle stesse. Questo algoritmo sfrutta sia le tecniche di ricerca della massima variazione luminosa analizzando l'immagine ottenuta con l'operatore differenziale filtro di Sobel (cvSobel), sia le tecniche di identificazione dell'attraversamento dello zero, analizzando la divergenza del gradiente dell'immagine ottenuto grazie all'operatore filtro di Laplace (cvLaplace). Il funzionamento dell'algoritmo è il seguente: inizialmente si esegue il filtraggio dell'immagine per ridurne la rumorosità, si esegue con un filtro passa basso di *smoothing* (cvSmooth), in questo modo vengono rimossi piccoli dettagli e variazioni repentine di luminosità (l'immagine ottenuta viene denominata **median**); successivamente si ottiene l'immagine che approssima il gradiente di

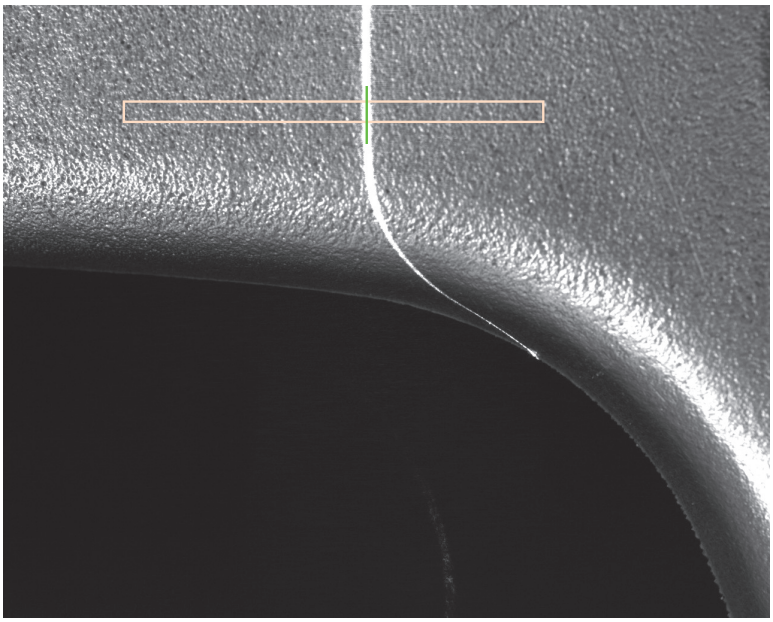


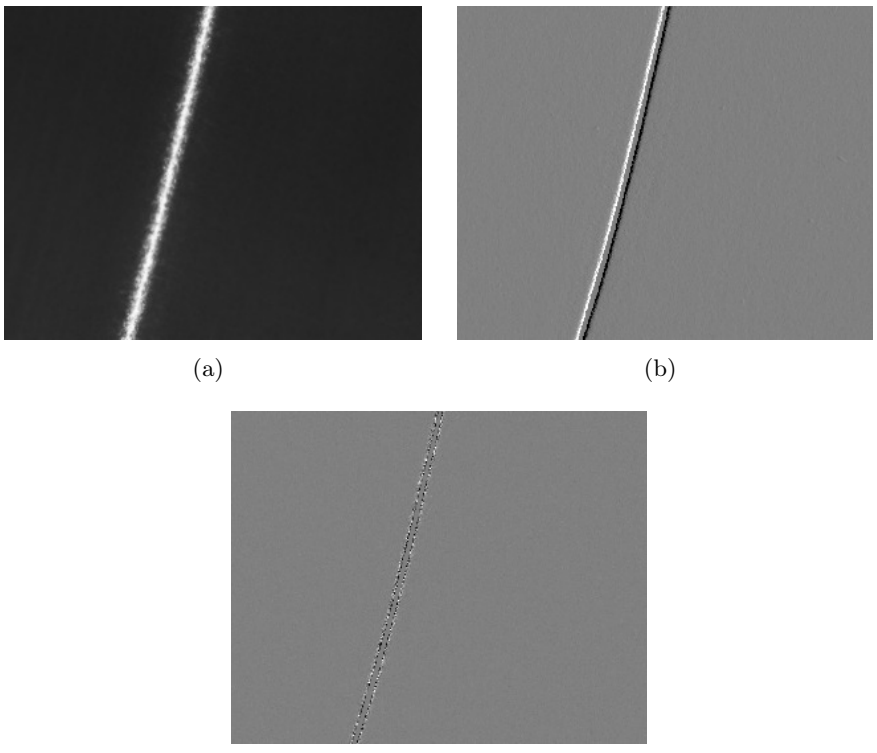
Figura 5.2: Esempio di funzionamento dell'*algoritmo di base*, la regione di interesse (ROI) è evidenziata dal rettangolo mentre il valore ξ_x individuato per la posizione della linea laser è marcato con la linea verde.

luminosità effettuando il filtraggio di Sobel imponendo la direzione orizzontale, in questo modo si evidenziano maggiormente le variazioni luminose lungo la direzione in cui ci si aspetta di rilevare la linea laser. Dall'immagine ottenuta (**FD**) si possono già ricavare la soglia di massima variazione luminosa positiva (area bianca) e quella di massima variazione luminosa negativa (area nera) (vedi figura 5.3)[2]; concentrandosi per il momento sulla soglia di massima variazione positiva si analizza l'intera ROI di **FD** registrando i valori massimi nel vettore denominato **Peaks[**$\cdot, \cdot, \cdots$**]** con una tolleranza del 30%. Successivamente si esegue l'operatore laplaciano (cvLaplace)[3] sempre sull'immagine **median** ottenendo un'approssimazione discreta della seconda derivata della funzione polinomiale che interpola i valori di intensità luminosa; si osservano ora i corrispettivi dei pixel registrati in **Peaks[**$\cdot, \cdot, \cdots$**]** nell'immagine **SD**, si esegue una ricerca dei valori di grigio equivalenti a **ZERO_SHIFT** (valore associato all'assenza di variazione luminosa, corrispondente al valore dei pixel

[2]Osservando la figura 5.4 si comprende che la dimensione del kernel influisce sulla resa dell'immagine le aree di massima e minima variazione sono più definite ma la rumorosità è incrementata, per questo motivo si è preferito implementare un ulteriore passaggio dell'algoritmo e mantenere un immagine meno romurosa.

[3]Per l'esattezza l'operatore matematico in coordinate cartesiane nello spazio a 2 dimensioni $\bigtriangledown^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$ viene approssimato con il suo analogo discreto ed implementato sfruttando la funzione *cvSobel*.

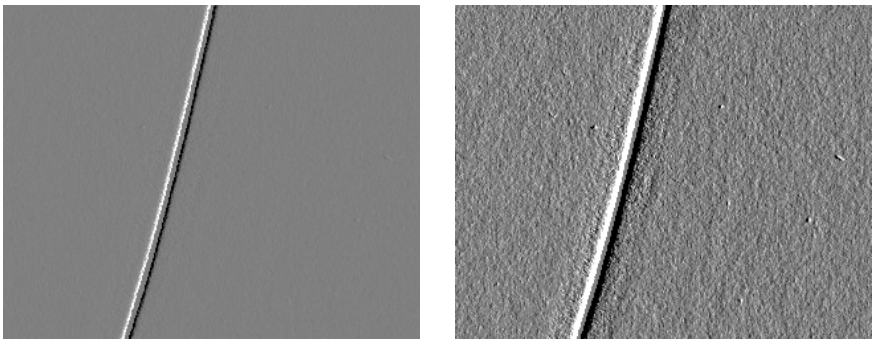
grigi sullo sfondo nell'immagine 5.3 (b) e (c)), vengono anche considerati i punti di attraversamento dello zero in cui un pixel ed il suo successivo presentano valori separati della soglia **ZERO_SHIFT** (in questo caso un pixel presenta un valore di seconda derivata positiva, mentre il suo successivo un valore di seconda derivata negativa; per questi punti viene registrato il valore medio), fra i punti identificati si seleziona quello a cui corrisponde un valore più grande nell'immagine **FD**, se esistono più massimi si seleziona quello più a destra, la coordinata ξ_{x_left} corrispondente a questo punto viene salvata nella variabile **posl**. Si ripete un procedimento analogo per identificare la coordinata ξ_{x_right} associata al minimo di **FD**, se fra i valori selezionati dopo la ricerca dell'attraversamento della soglia zero in **SD** ce ne sono diversi per cui **FD** raggiunge il valore massimo allora si sceglie il valore più a sinistra (il punto viene registrato nella variabile **posl2**); il risultato dell'algoritmo è la coordinata ξ_{x_line}, ottenuta come media fra le due soglie individuate $\xi_{x\_\text{line}} = (\xi_{x\_\text{left}} + \xi_{x\_\text{right}})\frac{1}{2}$ (il valore viene registrato nella variabile **posl3**).



(a) (b)

(c)

Figura 5.3: Esempio immagini analizzate con *algoritmo secondario*: immagine originale (a), elaborazione con filtro Sobel ampiezza con kernel 3 (**FD**) (b), elaborazione con funzione cvLaplace con ampiezza kernel 3 (**SD**) (c).



(a) (b)

Figura 5.4: Esempio immagini filtrate con *cvSobel*: dimensione *kernel* 3 direzione x (a), dimensione *kernel* 5 direzione x (a).

5.3 Confronto fra algoritmi

Per poter verificare l'efficacia degli algoritmi implementati e poterne confrontare i risultati sono state effettuate delle serie di cento immagini dalle quale sono state estrapolate misure dei medesimi punti:

- 1^a serie : linea laser osservata sul piano di appoggio del cerchione;
- 2^a serie : linea laser osservata su di un oggetto di calibrazione formato da gradini regolari, la scala è composta da gradini quadrati posti a distanza di 1 cm fra di loro;
- 3^a serie : linea laser osservata sul cerchione, profilo illuminato di una razza con bava.

I dati ottenuti con i diversi algoritmi, sono stati analizzati e verificate la deviazione standard e la media delle misure ottenute.

Nella 1^a serie la linea laser è stata posizionata verticalmente e passante per il centro immagine; sono state effettuate 9 misure di punti per ognuna delle immagini acquisite a distanze regolari partendo dal bordo superiore e raggiungendo il margine inferiore dell'immagine. Come si può notare nella tabella 5.1 il massimo valore di deviazione standard della misura lo si ottiene con l'*Algoritmo di Base* ed è pari a 105 μm.

Nella 2^a serie la linea laser già centrata è stata proiettata sull'oggetto di calibrazione a scala noto (vedi tabella 5.2); sono state misurate le 9 misure di distanza di ogni gradino. I risultati delle misure sono riportati in tabella; si nota che il massimo valore di deviazione standard della misura lo si ottiene con l'*Algoritmo secondario* ed è pari a 235 μm.

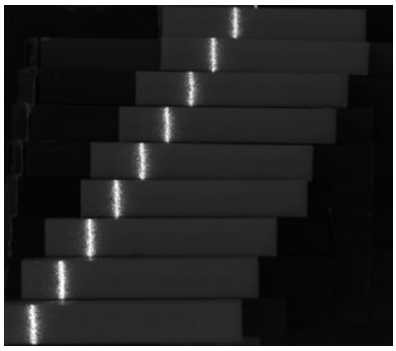
Nella 3^a serie la linea laser è stata proiettata sul profilo di un cerchione automobilistico affetto da bave, sono state rilevate le misure di distanza di

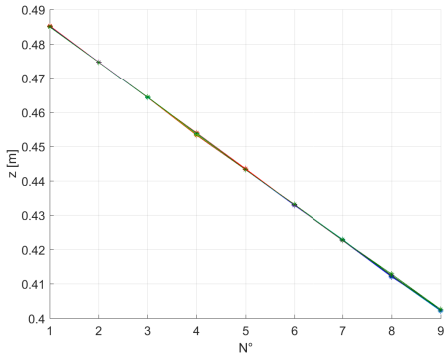
Algoritmo Base		Algoritmo Secondario		Media algoritmi	
Medie	Dev. standard	Medie	Dev. standard	Medie	Dev. standard
495.4	0.000	495.5	0.047	495.4	0.022
495.4	0.000	495.4	0.000	495.4	0.000
495.4	0.000	495.4	0.000	495.4	0.000
495.4	0.000	495.3	0.000	495.4	0.000
495.4	0.000	495.3	0.000	495.4	0.000
495.2	0.041	495.3	0.000	495.3	0.021
495.3	0.105	495.3	0.000	495.3	0.053
495.2	0.093	495.3	0.000	495.3	0.047
495.2	0.000	495.2	0.000	495.2	0.000

Tabella 5.1: Misurazione di un piano parallelo al piano immagine - Campione di 100 misure di distanza effettuate su 9 punti illuminati dalla linea laser [mm].

9 punti ad intervalli regolari dal margine superiore dell'immagine fino al margine inferiore. Come si può notare analizzando i dati riportati in tabella 5.3 i valori di deviazione standard massimi sono pari a 92 μm per l'*algoritmo secondario* e 82 μm per l'*algoritmo di base*.

Dal confronto dei dati riportati in queste serie di misure si ricavano importanti informazioni sul funzionamento degli algoritmi e sulle problematiche legate alla identificazione dell'esatta posizione della linea laser sul piano immagine, ovviamente tali misure forniscono indicazioni sul comportamento degli algoritmi in una condizione di utilizzo statico (manipolatore fermo), immagini affette da rumore e variazioni di luce ambientale. Il massimo valore di deviazione standard riscontrato (vedi tabella 5.2) è ottenuto con l'*algoritmo secondario*, la causa di una maggior variabilità della misura è dovuta al fenomeno di *speckle*, più evidente in queste immagini anche a causa dell'ampio intervallo di misurazione in cui sono rilevate le distanze; Il fatto che anche l'*algoritmo di base* presenti una grande deviazione standard in quel punto indica che questa è la maggior problematica da affrontare durante la fase sperimentale. Un altro aspetto da notare è che i valori di deviazione standard massima delle 3 serie di misurazioni, realizzate su superfici diverse ed in condizioni differenti, si verificano per misurazioni di punti diversi per il primo algoritmo rispetto al secondo (punto 1 tabella 5.1 per l'*algoritmo secondario* e punto 7 per l'*algoritmo di base*; punto 2 tabella 5.3 per l'*algoritmo di base* e punto 9 per l'*algorirtmo secondario*), questo aspetto è da imputare alle differenti tecniche di estrazione della posizione della linea luminosa dei due algoritmi ma dimostra che l'utilizzo combinato degli stessi, come si può notare nella colonna "media algoritmo" riportata nelle tabelle (vedi 5.1,5.2,5.3), porta a ridurre la varianza della misura, ottenendo uno strumento di rilevamento della linea laser più versatile, che offre buoni risultati di misura in condizioni





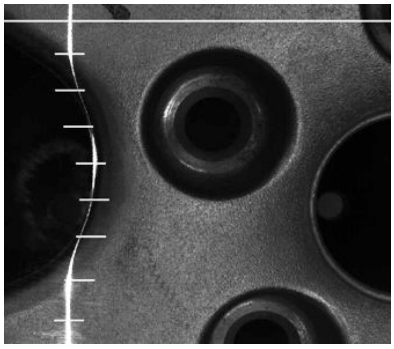
Algoritmo Base		Algoritmo Secondario		Media algoritmi	
Medie	Dev. standard	Medie	Dev. standard	Medie	Dev. standard
485.1	0.000	485.1	0.125	485.1	0.062
474.6	0.000	474.7	0.046	474.6	0.023
464.2	0.000	464.2	0.049	464.2	0.024
453.7	0.081	453.7	0.235	453.7	0.134
443.2	0.000	443.3	0.159	443.3	0.080
433.1	0.038	432.9	0.021	433.0	0.024
422.9	0.000	422.7	0.099	422.8	0.049
412.3	0.021	412.4	0.233	412.3	0.116
402.2	0.000	402.3	0.150	402.2	0.056

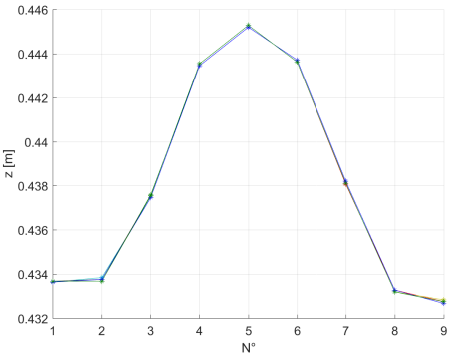
Tabella 5.2: Misurazione di una scala a gradini regolari - Campione di 100 misure di distanza effettuate su 9 punti illuminati dalla linea laser, misure in [mm].

anche molto differenti.

5.4 Applicazione al problema CONTOUR FOLLOWING

Per poter svolgere la fase sperimentale di misurazione del profilo di sbavatura, è stato implementato uno algoritmo di *computer vision* per il rilevamento della linea laser: l'algoritmo è implementato in linguaggio C, con l'ausilio delle librerie grafiche OpenCV, è in grado di identificare la linea laser e di registrarne la posizione finale che corrisponde ad un bordo del pezzo. Ipotizzando di osservare l'oggetto dall'alto e mantenendo la visuale centrata sul bordo inferiore, la linea laser illumina la superficie del pezzo, seguendo l'andamento di questa dall'alto verso il basso si giunge al punto finale in corrispondenza del profilo del pezzo, denominato profilo di sbavatura data la maggior probabilità di rilevare bave su di esso. Acquisendo immagini della linea laser durante la traslazione dello strumento lungo il profilo dell'oggetto, è possibile registrando





Algoritmo Base		Algoritmo Secondario		Media algoritmi	
Medie	**Dev. standard**	**Medie**	**Dev. standard**	**Medie**	**Dev. standard**
433.8	0.000	433.5	0.024	4337	0.012
434.0	0.082	433.5	0.024	433.7	0.044
437.5	0.025	437.5	0.026	437.5	0.025
443.5	0.027	443.5	0.027	443.5	0.027
445.2	0.027	445.3	0.027	445.3	0.027
443.6	0.027	443.6	0.027	443.6	0.027
438.1	0.026	438.1	0.030	438.1	0.027
433.4	0.028	433.0	0.028	433.2	0.028
433.1	0.028	432.4	0.092	432.8	0.055

Tabella 5.3: Misurazione di un profilo del cerchione - Campione di 100 misure di distanza effettuate su 9 punti illuminati dalla linea laser, misure in [mm].

il valore di posizione del punto del profilo (ξ_{x_burr}, ξ_{y_burr}) illuminato in ogni immagine, ricostruire le coordinate (x,y,z) del profilo. Nell'algoritmo sono state implementate soluzioni innovative, verificate sperimentalmente, per risolvere le maggiori problematiche riscontrate nell'identificazione della linea laser:

- la presenza di zone fortemente illuminate, che possono essere interpretate come illuminate dalla linea laser;
- le riflessioni speculari della linea laser causate dalla superficie metallica di forma complessa, sono difficilmente discriminabili dalla linea laser principale, possono causare ambiguità date dall'identificazione di più punti luminosi anche distanti nella stessa ROI (vedi figura 5.6);
- la presenza di disturbi di tipo *speckle*, causati dalla rugosità superficiale del materiale e dalla finitura, generano *blob* luminosi ai margini della linea laser, si possono creare anche punti non illuminati che interrompono la traccia della linea luminosa;

- l'elaborazione dell'immagine computazionalmente pesante, può rallentare l'*image processing* e causare dei problemi di controllo del manipolatore.

In questa particolare applicazione il tempo di elaborazione è un parametro stringente, perché i dati forniti come risultato dell'algoritmo, che identificano la coordinata z del profilo, devono essere utilizzati nell'anello di controllo del manipolatore, per questa ragione si è scelto di implementare l'algoritmo con tempo di elaborazione minore, in questo caso fra i due algoritmi presentati: l'*algoritmo di base*; dopo aver fatto alcune prove su immagini del profilo del pezzo, si è calcolato il tempo medio di ciclo di elaborazione di ogni algoritmo e quello dell'implementazione di entrambi gli algoritmi per usare il risultato medio, il tempo medio di ciclo di quest'ultimo era decisamente superiore ai 100 ms ed anche il tempo computazionale dell'*algoritmo secondario* è risultato maggiore.

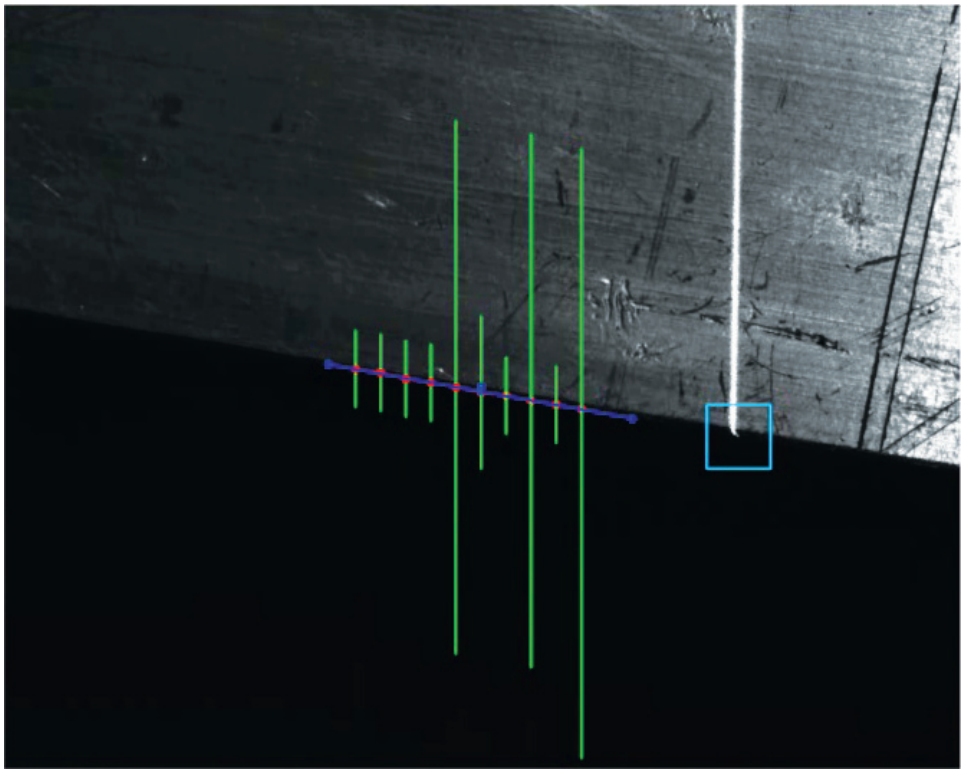


Figura 5.5: Esecuzione dell'algoritmo di *contour following* e misurazione del profilo. Immagine acquisita dal sistema a triangolazione con l'aggiunta di elementi grafici: punto identificato per rilevazione della coordinata z (centro del quadrato azzurro), ROI identificazione bordo algoritmo *contour following* (linee verdi), interpolazione lineare del profilo (linea blu)[22].

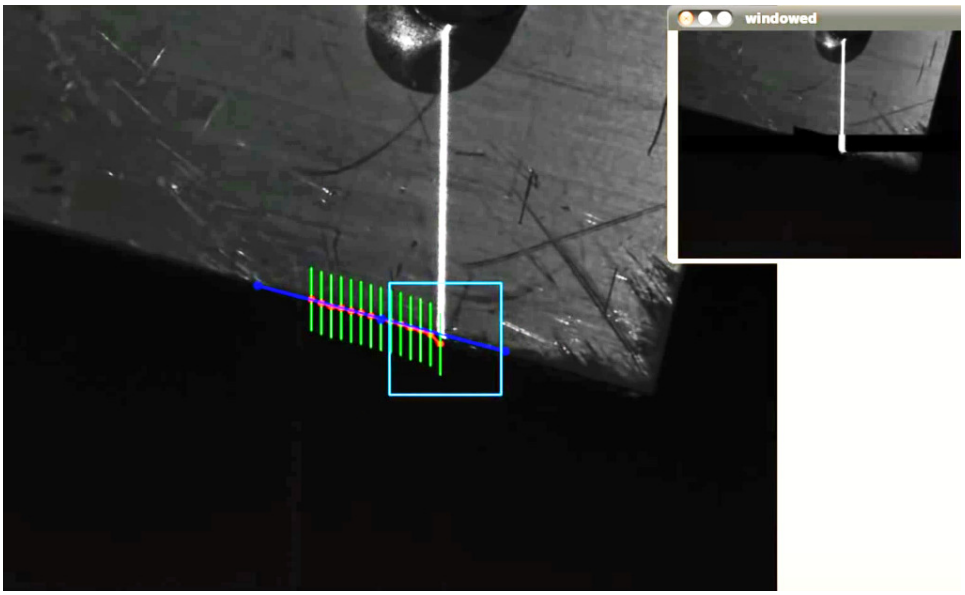


Figura 5.6: Esempio di immagine analizzata durante il processo di *contour following*; vicino al bordo superiore si può notare un effetto di riflessione speculare della linea laser.

L'elaborazione dell'immagine è limitata ad una ristretta area di interesse dell'immagine (**ROI**), e consiste in un aumento di luminosità nell'intorno della posizione della linea laser precedentemente identificata, unita al decremento della luminosità nelle aree più distanti da essa, ma sempre restando all'interno della ROI, i risultati di questa elaborazione possono essere osservati nella schermata denominata **windowed** (vedi figura 5.7), questo permette facilmente di mantenere l'algoritmo computazionalmente veloce anche con immagini di grandi dimensioni.

Per rendere versatile l'algoritmo e renderlo efficace anche in condizioni luminose differenti e superfici di forma complessa, nella fase di inizializzazione è possibile impostare 6 parametri (**Threshold Low Brightness**, **Increment Brightness**, **Decrement Brightness**, **Line Dimension**, **Line Dimension**, **Threshold High Distance**), questi parametri influenzano l'elaborazione dell'immagine ed il risultato dell'algoritmo, quindi una volta impostati correttamente si possono eseguire tutte le misurazioni necessarie sull'oggetto voluto, a parità di condizioni luminose. Il significato fisico di tali parametri è il seguente:

- **Threshold Low Brightness** è utilizzato per identificare quando la riga analizzata non ha più punti sufficientemente luminosi, se il valore di grigio del pixel identificato come il centro della linea laser è inferiore

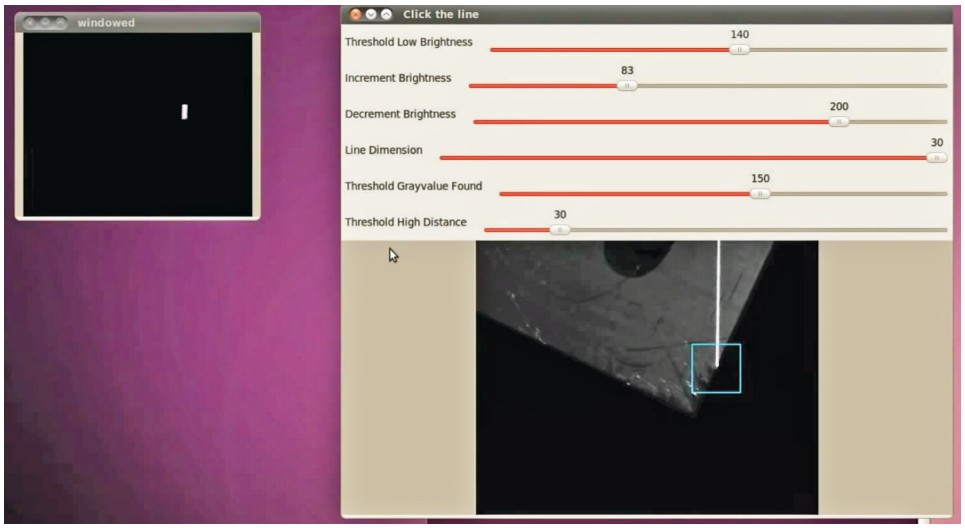


Figura 5.7: Schermata algoritmo di *image analysis*, procedura iniziale di settaggio parametri.

a tale soglia allora o si tratta di un interruzione nella linea laser oppure il bordo del pezzo è stato superato;

- **Increment Brightness** è utilizzato per incrementare la luminosità della zona attorno a cui si è trovata la linea laser precedentemente, il valore di grigio dei pixel in quest'area viene incrementato di questo valore; in questo modo si riesce ad evidenziare la linea anche se in alcuni punti è scura o deteriorata da effetti di *speckle*;
- **Decrement Brightness** è utilizzato per diminuire la luminosità dei pixel della zona lontana dal punto attorno a cui si è trovata la linea laser precedentemente, il valore di grigio dei pixel in quest'area viene diminuito del valore indicato; in questo modo si riescono ad eliminare zone luminose distanti dalla linea dovute ad una eccessiva illuminazione ambientale o a riflessioni del laser;
- **Line Dimension** è utilizzato per impostare la dimensione della linea laser in pixel, se esiste un pixel con valore di luminosità pari al massimo rilevato, più lontano dalla posizione calcolata della linea laser di metà di questo valore, allora vuol dire che l'area luminosa rilevata è troppo grossa e non è la linea laser, essa può essere un'altra area luminosa oppure la linea laser corrotta da riflessi anomali;
- **Threshold Grayvalue Found** è utilizzato per impostare una soglia minima di luminosità oltre la quale si è sicuri di aver rilevato la linea laser, in questo modo si evita che si verifichino le condizioni di uscita

dell'algoritmo prima ancora che venga trovata la linea laser fornendo coordinate di posizione del bordo del pezzo sbagliate;

- **Threshold High Distance** è utilizzato per identificare quando una misura è troppo diversa da quella precedente, questa soglia esprime la distanza in pixel oltre la quale si accende un campanello di allarme ed è opportuno verificare di non aver rilevato una zona luminosa sbagliata che non corrisponde alla linea laser.

Per impostare correttamente questi parametri è stata seguita una semplice procedura, dapprima si è impostato **Line Dimension** scegliendo anche una dimensione di qualche pixel più grande rispetto allo spessore della linea laser osservato, dopo di ciò si è selezionata un'area lontana dalla bava in cui la linea laser è continua, quindi si è regolato **Increment Brightness** iniziando con un valore di **Threshold Low Brightness** alto in modo da verificare che eventuali aree scure del pezzo o graffi non causino mai il verificarsi della condizione di uscita dell'algoritmo; successivamente si è selezionata l'area intorno alla bava e si è regolato **Threshold Low Brightness** in modo che l'algoritmo termini effettivamente in concomitanza del bordo del pezzo e non prosegua lungo il profilo illuminato del pezzo; se non è possibile ovviare in questo modo a quest'ultima evenienza, aumentare Decrement Brightness e/o diminuire Increment Brightness e Line Dimension se possibile. Dopo aver ultimato l'impostazione dei parametri si richiede all'utente di inizializzare l'algoritmo selezionando la posizione della linea in un intorno del bordo del pezzo, in seguito il programma partendo dall'alto analizza riga per riga l'immagine ricercando i punti di massima luminosità e salvando come valore di ξ_{x_line} il punto mediano fra essi; l'algoritmo si interrompe e restituisce il risultato voluto: la posizione del bordo del pezzo (ξ_{x_burr}, ξ_{y_burr}, misurate in pixel sul piano immagine con origine nell'angolo in alto a sinistra; e ${}^{c}z$ distanza del punto in terna camera), solo a seguito del verificarsi di alcune condizioni che identificano l'interruzione della linea laser a causa dell'effettivo raggiungimento del bordo del pezzo. In figura 5.8 si descrive il funzionamento dell'algoritmo, tale schema viene eseguito ad ogni acquisizione di immagini dalla camera.


INIZIO CICLO
Acquisisci immagine da telecamera: src_img
Crea immagine filtrata: flt_img
Operazione di incremento luminosità se nell'intorno della coordinata x-precedente altrimenti decremento di luminosità
Registra elaborazione in: flt_img
Ricerca punto di massima luminosità riga
Il punto è più luminoso di Threshold Grayvalue Round
NO
SI
riga successiva: y+1

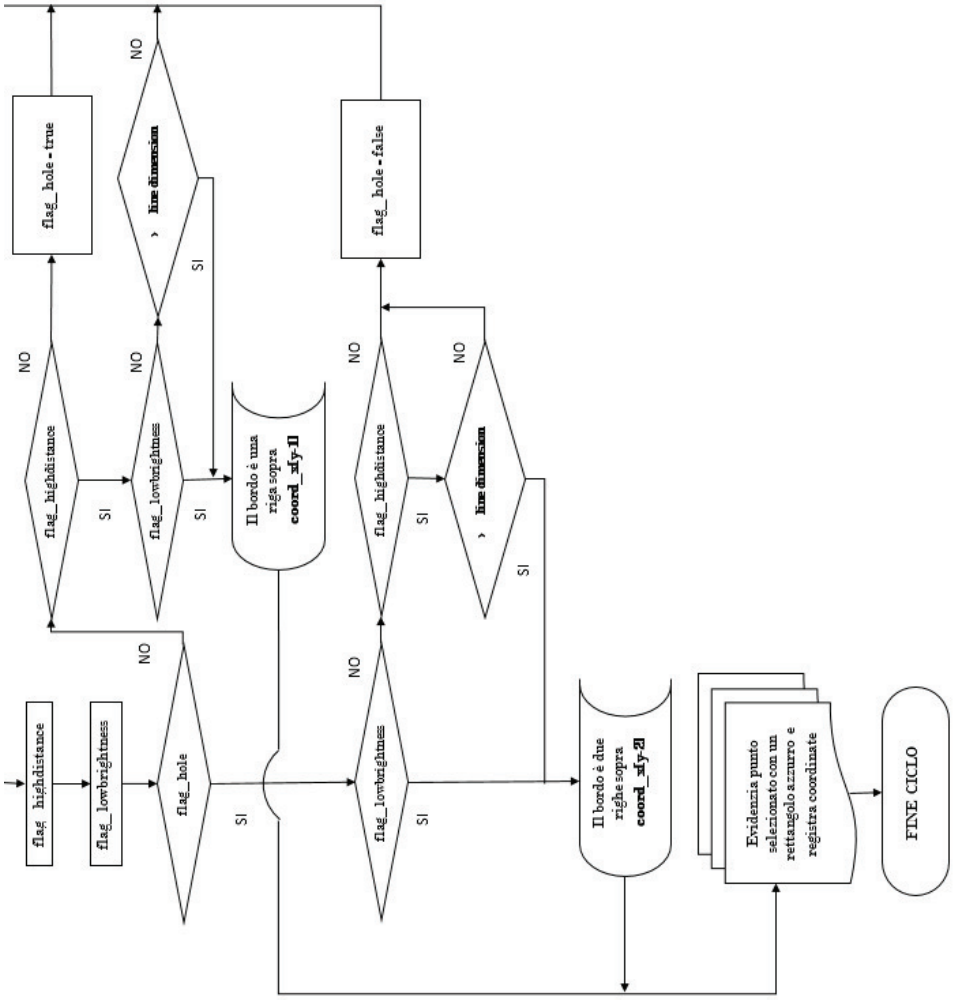


Figura 5.8: Diagramma di flusso algoritmo di ricerca bordo illuminato da linea laser, l'algoritmo rappresenta il ciclo di *computer vision* eseguito per ogni immagine

5.5 Applicazione al problema HIGH COMPLEXITY 3D PROFILE

L'utilizzo dell'algoritmo descritto (vedi paragrafo 5.4) su superfici particolarmente complesse, come la superficie di un cerchione automobilistico affetto da bave, è possibile e permette di ottenere buoni risultati, anche in presenza di effetto di *speckle* ed aree fortemente illuminate al di fuori della linea laser principale; per questa applicazione l'algoritmo registra tutti i

valori di coordinate (ξ_x, ξ_y) in cui è stata rilevata la linea, con l'obbiettivo di ricostruire in seguito l'intera superficie 3D del pezzo dalla nuvola di punti acquisita; risulta comunque utile l'informazione di rilevamento del bordo inferiore del pezzo fornita dall'algoritmo, in questo modo è possibile limitare l'analisi dell'immagine alle sole aree di interesse, riducendo i tempi di calcolo e potendo utilizzare tale informazione per la movimentazione del manipolatore robotico.

L'utilizzo di un metodo molto diffuso per il rilevamento di bordi presenti in un immagine: l'algoritmo *Canny* (implementato nella libreria OpenCV come cvCanny) è stato sfruttato per il rilevamento dei bordi del pezzo dall'algoritmo di *contour following*, ma è risultato meno efficace dell'algoritmo proposto per il rilevamento della linea laser. Le particolari condizioni luminose e la necessità di mantenere la linea laser il più sottile possibile, rendono particolarmente difficile l'identificazione dei dordi della linea laser con l'algoritmo Canny. L'effetto di *speckle* prodotto sulla superficie dei manufatti metallici, che si presenta ruvida e con incisioni, genera per ogni ROI una moltitudine di valori classificati come bordi dall'algoritmo e fra questi, avendo perso le informazioni dei livelli di grigio dell'intera immagine, risulta difficile distinguere l'effettivo bordo della linea da quello di una area a forte variazione luminosa differente; inoltre l'algoritmo Canny rileva bordi presenti nell'immagine in ogni direzione, ma questo non è strettamente necessario per l'applicazione in esame sapendo che dopo l'operazione di calibrazione la linea laser non è rilevabile in posizione orizzontale, e nella maggior parte dei casi (quando la zona illuminata si trova ad una distanza costante) la linea si trova orientata verticalmente.

La problematica delle occlusioni, profondamente legata alla tecnica di scansione 3D con *slit scanner* ed alla configurazione scelta per lo strumento, è riscontrabile in questa applicazione; il profilo del cerchione presenta avvallamenti della superficie anche repentini ed a breve distanza fra loro, perciò la luce proiettata su un area posta ad un livello superiore genera una "zona d'ombra" sull'area posta ad un livello inferiore subito adiacente (per la nostra configurazione se un area a distanza inferiore si trova subito a sinistra di un'area a distanza consistentemente maggiore), di questa "zona d'ombra" non è possibile effettuare misurazioni, se non in taluni casi modificando l'orientamento dello strumento. Per aggirare questo problema è possibile eseguire più scansioni della stessa superficie con orientamenti differenti ed una volta raccolti i dati delle diverse scansioni, unire le nuvole di punti sfruttando le informazioni di posa ed orientamento della camera ottenute in fase di calibrazione e le relazioni cinematiche del manipolatore; oppure senza sfruttare le informazioni di posa ed orientamento della videocamera utilizzare un algoritmo di *matching* di punti caratteristici dell'oggetto (*features*, angoli, spigoli o forme di dimensioni note) per allineare le nuvole di punti con il giusto orientamento (vedi LS3D nel paragrafo 2.2).

Capitolo 6

Calibrazione

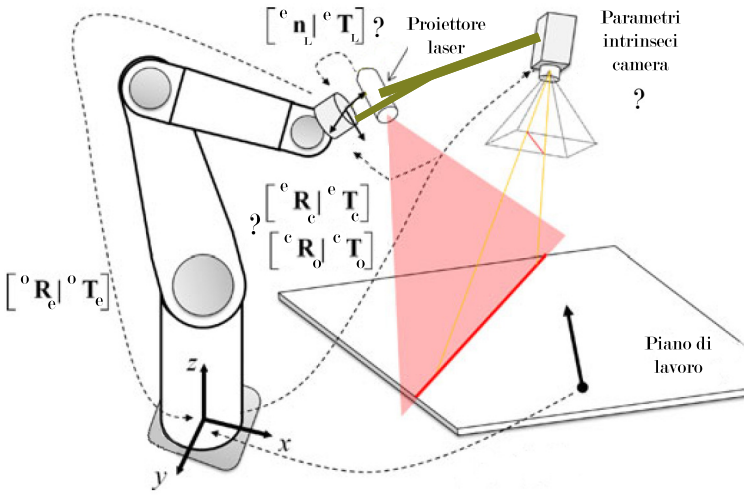


Figura 6.1: Schema del sistema eye-in-hand con linea laser.

Nel seguente capitolo si presenta il lavoro svolto per realizzare la calibrazione del sistema di misurazione a triangolazione laser.

Considerando le equazioni fondamentali di proiezione prospettica, le coordinate di un generico punto P rispetto alla terna camera $({}^c x, {}^c y, {}^c z)$ sono in relazione con le coordinate 2D della proiezione di P sul piano immagine (ξ_x, ξ_y) grazie alle equazioni (3.18), qui riportate:

$$\begin{aligned} \xi_x &= f_x \frac{{}^c x}{{}^c z} + c_x \\ \xi_y &= f_y \frac{{}^c y}{{}^c z} + c_y \end{aligned} \tag{6.1}$$

dove f_x, f_y sono le lunghezze focali in pixel (incluse le distanze dal centro di un pixel a quello successivo), e c_x, c_y sono le coordinate immagine del centro ottico (intersezione dell'asse ottico ${}^{c}z$ con il piano immagine), questi quattro parametri sono noti come parametri intrinseci della camera *pin-hole*. Le relazioni che permettono di definire il medesimo punto, espresso in terna mondo (coordinate ${}^{o}x$, ${}^{o}y$, ${}^{o}z$), nella terna della camera sono definite dalla rotazione ${}^{c}R_o$ e dalla traslazione ${}^{c}T_o$. Si ottengono le seguenti equazioni:

$$\begin{aligned} {}^{c}x &= r_{11}{}^{o}x + r_{12}{}^{o}y + r_{13}{}^{o}z + {}^{c}T_{o_x} \\ {}^{c}y &= r_{21}{}^{o}x + r_{22}{}^{o}y + r_{23}{}^{o}z + {}^{c}T_{o_y} \\ {}^{c}z &= r_{31}{}^{o}x + r_{32}{}^{o}y + r_{33}{}^{o}z + {}^{c}T_{o_z} \end{aligned}$$

dove r_{ij} e ${}^{c}T_{o_a}$ con $a = x, y, z$, sono gli elementi della matrice omogenea $[{}^{c}R_o|{}^{c}T_o]$ che definisce i parametri estrinseci della camera; perciò le trasformazioni che portano un generico punto P in terna mondo al corrispondente punto proiettato sul piano immagine possono essere riscritte in forma più compatta come:

$${}^{c}z \begin{bmatrix} \xi_x \\ \xi_y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [{}^{c}R_o|{}^{c}T_o] \begin{bmatrix} {}^{o}x \\ {}^{o}y \\ {}^{o}z \\ 1 \end{bmatrix} = \mathbf{K} [{}^{c}R_o|{}^{c}T_o] \begin{bmatrix} {}^{o}x \\ {}^{o}y \\ {}^{o}z \\ 1 \end{bmatrix}$$

Questa formulazione rappresenta una prima approssimazione delle trasformazioni che coinvolgono l'acquisizione di punti nello spazio 3D con una videocamera reale; per molte applicazioni tale approssimazione è più che sufficiente ma volendo raffinare il modello è possibile introdurre ulteriori parametri intrinseci (k_1 k_2) che rappresentano la distorsione radiale delle lenti e (p_1 p_2) parametri che rappresentano la distorsione tangenziale dovuta al disallineamento dei centri di curvatura delle lenti (arrivando a 8 parametri intrinseci) o anche il coefficiente di *skew* (s) che permette di definire nel modello la non ortogonalità degli assi dell'immagine [15].

Il procedimento per ottenere i parametri incogniti che riguardano:

- la trasformazione delle coordinate dei punti dalla terna camera al piano immagine (parametri intrinseci: trasformazione prospettica e distorsione introdotta dalle lenti);
- la posa della videocamera rispetto alla terna end effector (parametri estrinseci: rotazione definita dalla matrice ${}^{e}R_c$ e traslazione definita dal vettore ${}^{e}T_c$);
- la relazione trigonometrica tra i punti illuminati dalla linea laser sul piano immagine e la coordinata ${}^{c}z$;

può essere svolto in diverse fasi:

1. scelta dell'oggetto di calibrazione, solitamente una scacchiera a quadrati bianchi e neri alternati, tracciata su di un piano (qualunque griglia di punti planari con distanze note fra loro può essere utilizzata);

2. acquisizione di immagini dell'oggetto di calibrazione da differenti angolazioni; a seconda della tecnica scelta è possibile effettuare spostamenti della camera rispetto all'oggetto, mantenendo lo stesso orientamento (tecnica efficace per una calibrazione veloce sfruttando una risoluzione lineare del problema (vedi paragrafo 6.3); od imponendo rototraslazioni fra un'acquisizione e l'altra (tecnica che richiede la risoluzione di un problema non lineare di ottimizzazione, che permette di identificare più precisamente i parametri intrinseci della camera (vedi paragrafi 6.1 e 6.1);

3. ricostruzione parametri estrinseci della videocamera, utilizzo della matrice di rotazione e del vettore di traslazione così ottenuto come valori di inizializzazione per la risoluzione del problema di ottimizzazione non lineare, che permette di identificare anche i parametri intrinseci della camera;

4. posizionamento del proiettore laser, durante la fase di calibrazione della camera il proiettore laser non è stato sfruttato;

5. acquisizione di immagini di un oggetto di calibrazione illuminato dalla linea laser; benché il procedimento di calibrazione che identifica la posizione del piano laser rispetto alla terna dell'end effector può essere svolto anche utilizzando una scacchiera come oggetto di calibrazione (vedi [18]), si è preferito utilizzare un oggetto 3D di forma nota, per poter sfruttare le proprietà di parallelismo di alcuni piani posti a distanze diverse presenti sull'oggetto;

6. determinazione dei parametri di triangolazione che consentono di associare un punto illuminato dalla linea laser al corrispettivo punto nella terna di riferimento.

in seguito si descrivono brevemente i diversi approcci applicati ed i risultati ottenuti con tre diverse metodologie di calibrazione della videocamera, sfruttando il software dedicato CalLab e CalDe (vedi paragrafo 6.1), oppure il Toolbox per Matlab sviluppato da Jean-Yves Bouguet (vedi 6.2), o ancora una procedura più veloce implementata *ad hoc* per la fase sperimentale del progetto (vedi paragrafo 6.3).

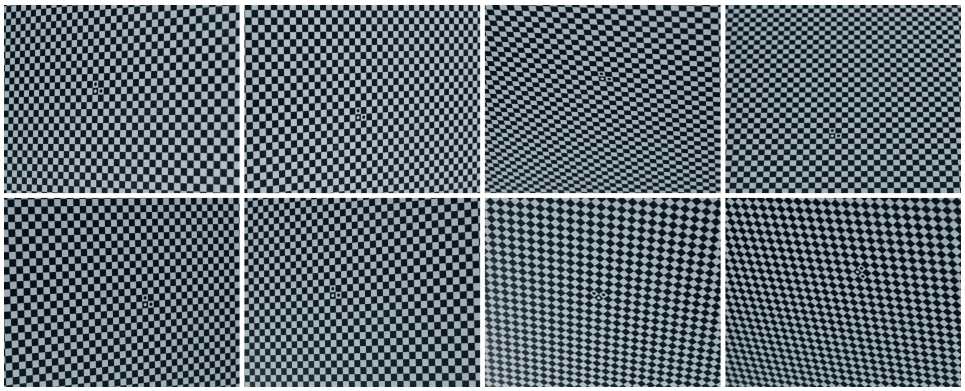


Figura 6.2: Immagini di calibrazione videocamera; 8 immagini da angolazioni differenti di una scacchiera a quadrati regolari di lato 2 mm.

6.1 Calibrazione con utilizzo di software dedicato CalLab e CalDe

Il centro di ricerca dell'Istituto di Robotica e Meccatronica del centro aerospaziale tedesco (DLR) sotto la supervisione di Sepp, Fuchs e Arbter, ha sviluppato un software apposito, DLR CalDe, per la calibrazione delle videocamere da utilizzare in applicazioni di *computer vision*, questo software assieme all'applicativo DLR CalLab, sviluppato da Strobl e Paredes, presso il medesimo Istituto, fornisce in un programma *stand-alone* tutti gli strumenti per ottenere i parametri intrinseci di una videocamera ed i relativi parametri estrinseci di posizione ed orientamento della stessa rispetto ad un oggetto fisso (configurazione *eye-in-hand*). Il pacchetto di programmi suddivide i compiti necessari alla risoluzione del problema di calibrazione fra due differenti software; DLR CalDe (*camera Calibration Detection tool*) ha lo scopo di identificare e misurare la posizione sul piano immagine dei punti salienti identificati sull'oggetto di calibrazione, quest'ultimo può essere sia 2D la classica scacchiera, che 3D con caratteristiche specifiche definite dal realizzatore; successivamente all'identificazione dei punti (*features*) nelle diverse immagini di calibrazione, il software DLR CalLab (*camera Calibration Lab*) processa i punti misurati risolvendo il problema di minimizzazione che porta all'identificazione dei parametri intrinseci della camera, nonché all'individuazione dei parametri estrinseci che definiscono la posa della videocamera rispetto alla terna del manipolatore robotico; la documentazione relativa ai metodi risolutivi sfruttati in questo pacchetto è fornita in articoli pubblicati dagli stessi autori [32] [30] [31] [35]. Il modello utilizzato per la trasformazione prospettica è quello non lineare con la modellazione della distorsione radiale e tangenziale oltre che al fattore di *skew*, ottenendo un modello a 9 parametri intrinseci (f_x, f_y, c_x, c_y, k_1, k_2, p_1, p_2, s), a cui vengono aggiunti i parametri

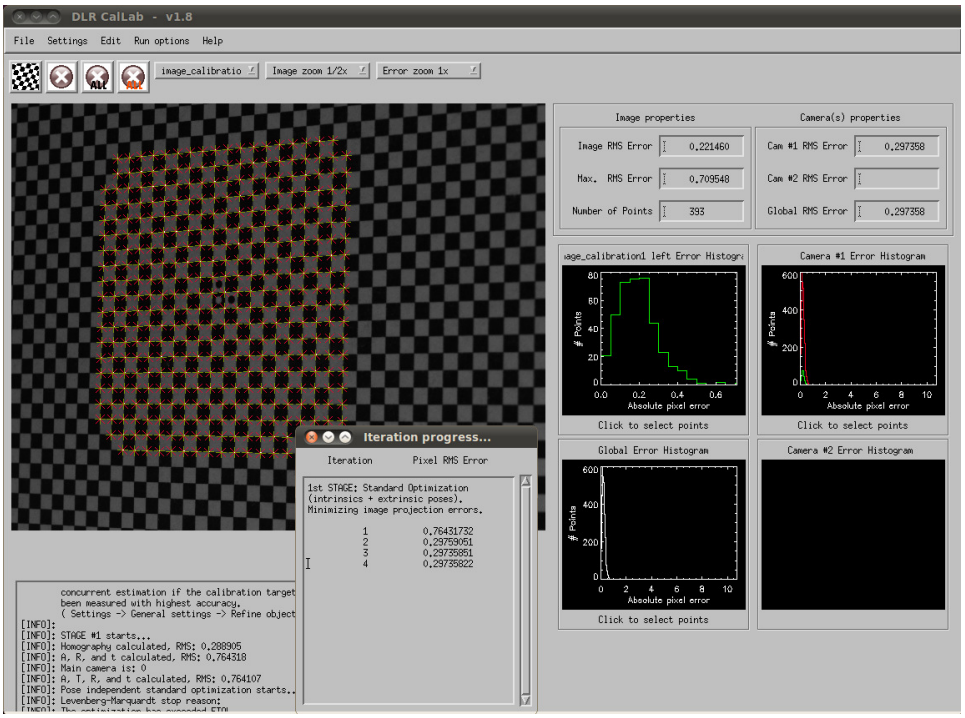


Figura 6.3: Finestra di interfaccia utente del software CalLab utilizzato per la calibrazione della videocamera.

estrinseci di $[{}^cR_e|{}^cT_e]$:

$$\begin{aligned} \xi_{xd} =& \xi_{xu} + (k_1 r^2 + k_2 r^4)\xi_{xu} + 2p_1\xi_{xu}\xi_{yu} + p_2(r^2 + 2 * {\xi_{xu}}^2) \\ \xi_{yd} =& \xi_{yu} + (k_1 r^2 + k_2 r^4)\xi_{yu} + 2p_2\xi_{xu}\xi_{yu} + p_1(r^2 + 2 * {\xi_{yu}}^2) \end{aligned} \tag{6.2}$$

con $r^2 = {\xi_{xu}}^2 + {\xi_{yu}}^2$; dove ξ_{xd} = coordinata con distorsione lenti e ξ_{xu} = coordinata senza distorsione lenti;

$${}^cz \begin{bmatrix} \xi_x \\ \xi_y \\ 1 \end{bmatrix}_{undistorted} = \mathbf{K}\,[{}^cR_e|{}^cT_e] \begin{bmatrix} {}^ex \\ {}^ey \\ {}^ez \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [{}^cR_e|{}^cT_e] \begin{bmatrix} {}^ex \\ {}^ey \\ {}^ez \\ 1 \end{bmatrix} \tag{6.3}$$

I risultati ottenuti dall'algoritmo di minimizzazione (Levenberg-Marquardt) della funzione di costo non lineare generata dagli errori di riproiezione dei punti osservati con i parametri stimati sono:

$$\mathbf{K} = \begin{bmatrix} 5138.70 & -22.97 & 640.28 \\ 0 & 5129.89 & 566.12 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.4}$$

e la corrispondente matrice omogenea che mette in relazione i punti della terna camera con quelli della terna end effector:

$$[{}^{e}R_{c}|{}^{e}T_{c}] = \begin{bmatrix} 0.6948 & 0.7192 & 0.0002 & -49.9107 \\ -0.7191 & 0.6946 & 0.0183 & -53.5406 \\ 0.0131 & -0.0129 & 0.9998 & 101.7711 \end{bmatrix} \tag{6.5}$$

con ${}^{e}T_{c}$ espresso in millimetri, k_1 = 0.0374497 [$1/pixel^2$], k_2 = -2.87514 [$1/pixel^2$], $p_1 = 0$ e $p_2 = 0$; il valore fornito come errore globale RMS sulle proprietà della camera è: 0.2973 [pixel].

6.2 Calibrazione con Camera Calibration Toolbox for Matlab

Un secondo pacchetto di funzioni implementato su software Matlab è il *Camera Calibration Toolbox* di Jean-Yves Bouguet[5]. anchesso permette di realizzare una *self-calibration* di una camera in configurazione *eye-in-hand* ricevendo come input immagini dell'oggetto di calibrazione e posa dell'end effector a cui è ancorata la videocamera durante l'acquisizione delle immagini. Il metodo di risoluzione del problema di ottimizzazione per la ricerca dei parametri incogniti deriva da numerosi articoli pubblicati (vedi [36] [15]), in particolare il modello presenta 8 parametri intrinseci (s è imposto a 0) f_x, f_y, c_x, c_y, k_1, k_2, p_1 e p_2. Fornendo gli stessi input della calibrazione eseguita con CalLab i risultati ottenuti sono i seguenti:

$$\mathbf{K} = \begin{bmatrix} 5128.29 & 0 & 635.14 \\ 0 & 5121.32 & 564.96 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.6}$$

e la corrispondente matrice omogenea che mette in relazione i punti della terna camera con quelli della terna end effector:

$$[{}^{e}R_{c}|{}^{e}T_{c}] = \begin{bmatrix} 0.6915 & 0.7223 & 0.0064 & -52.0189 \\ -0.7223 & 0.6914 & 0.0150 & -54.1912 \\ 0.0064 & -0.0150 & 0.9999 & 96.8719 \end{bmatrix} \tag{6.7}$$

k_1 =-0.00847 [$1/pixel^2$], k_2 = 0.15365 [$1/pixel^2$], $p_1 = 0$ e $p_2 = 0$; il valore fornito come incertezza sui valori è: ± 12 su f_x, f_y, ± 10 su c_x, c_y, ± 0.07 su k_1 e ± 7 su k_2 rendendo di fatto non attendibili i valori di distorsione delle lenti ottenuti.

6.3 Calibrazione utilizzata durante il processo di sperimentazione

Come si può notare confrontando i risultati ottenuti con le medesime immagini il processo di identificazione dei parametri di distorsione introdotti

dalle caratteristiche ottiche produce dei risultati ambigui si è scelto di implementare un ulteriore metodo di calibrazione trascurando tali effetti per poterne confrontare i risultati.

I parametri estrinseci della camera, definiscono la relazione, in termini di posizione ${}^{o}T_c$ ed orientamento ${}^{o}R_c$, tra la terna camera e la terna mondo stabilita. Considerando un punto 3D P, le cui coordinate sono date da ${}^{o}p$ rispetto alla terna mondo e ${}^{c}p$ rispetto alla camera, vale la seguente relazione:

$$ {}^{o}p = {}^{o}R_e\,(q)\,({}^{e}R_c\,{}^{c}p + {}^{e}T_c) + {}^{o}T_e\,(q) \tag{6.8} $$

dove ${}^{o}T_e\,(q)$ e ${}^{o}R_e\,(q)$ sono rispettivamente la posizione e l'orientamento dell'end effector nella terna mondo, essendo $q \in \mathbb{R}^6$ il vettore delle posizioni in coordinate giunti, e ${}^{e}T_c$ e ${}^{e}R_c$ sono, rispettivamente, la posizione e l'orientamento della terna camera rispetto alla terna dell'end effector. Il metodo ispirato alle tecniche di *self-calibration*, sfrutta le misurazioni accurate della posizione dell'end effector ottenute dalla cinematica diretta del manipolatore. Consideriamo ancora un punto P in 3D, le cui coordinate rispetto alla terna mondo sono date da ${}^{o}p$, e due differenti configurazioni del manipolatore, rappresentate dai vettori giunti q_1 e q_2. Vale la seguente equazione:

$$ {}^{o}p = {}^{o}R_e\,(q_1)\,({}^{e}R_c\,{}^{c}p_1 + {}^{e}T_c) + {}^{o}T_e\,(q_1) = {}^{o}R_e\,(q_2)\,({}^{e}R_c\,{}^{c}p_2 + {}^{e}T_c) + {}^{o}T_e\,(q_2) \tag{6.9} $$

dove ${}^{c}p_1$ e ${}^{c}p_2$ sono le coordinate di P rispetto alla terna camera, rispettivamente, il braccio robotico è in configurazione q_1 e q_2. Inoltre se l'orientamento dell'end effector è lo stesso in entrambe le configurazioni, ${}^{o}R_e\,(q_1) = {}^{o}R_e\,(q_2) = {}^{o}R_e$, l'equazione (6.9) può essere riscritta come:

$$ {}^{e}R_c\,({}^{c}p_1 - {}^{c}p_2) = {}^{o}R_e^{-1}\,({}^{o}T_e\,(q_2) - {}^{o}T_e\,(q_1)) \tag{6.10} $$

Dall'analisi della relazione precedente, segue che la parte a destra dell'uguale può essere calcolata per mezzo della cinematica diretta del manipolatore, la parte sinistra, invece, è il prodotto della matrice di orientamento della camera ${}^{e}R_c$, e delle coordinate di P rispetto alla terna camera. Un'espressione che mette in relazione quest'ultimo termine con le coordinate sul piano immagine del punto P, può essere ricavata dal modello pin-hole della camera (6.1)

$$ {}^{c}p_1 - {}^{c}p_2 = \begin{bmatrix} {}^{c}x_1 - {}^{c}x_2 \\ {}^{c}y_1 - {}^{c}y_2 \\ {}^{c}z_1 - \alpha\,{}^{c}z_1 \end{bmatrix} = {}^{c}z_1 \begin{bmatrix} \dfrac{\xi_{x_1} - \alpha\xi_{x_2}}{f_x} \\ \dfrac{\xi_{y_1} - \alpha\xi_{y_2}}{f_y} \\ 1 - \alpha \end{bmatrix} = {}^{c}z_1 \begin{bmatrix} \varphi_{x_1} - \alpha\varphi_{x_2} \\ \varphi_{y_1} - \alpha\varphi_{y_2} \\ 1 - \alpha \end{bmatrix} \tag{6.11} $$

where $\alpha = \dfrac{{}^{c}z_2}{{}^{c}z_1}$ and

$$ \varphi_{x_i} = \frac{\xi_{x_i}}{f_x} \quad \text{and} \quad \varphi_{y_i} = \frac{\xi_{y_i}}{f_y} \qquad i = 1, 2 $$

Raggruppando adesso tutte le quantità che possono essere calcolate utilizzando la cinematica diretta del manipolatore in un unico vettore:

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix} = {}^{o}R_{e}^{-1} \left({}^{o}T_{e}\left(q_2\right) - {}^{o}T_{e}\left(q_1\right) \right) \tag{6.12}$$

e introducendo le relazioni (6.11) e (6.12) nell'equazione (6.10) otteniamo

$${}^{e}R_{c}^{-1} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix} = {}^{c}z_1 \begin{bmatrix} \varphi_{x_1} - \alpha\varphi_{x_2} \\ \varphi_{y_1} - \alpha\varphi_{y_2} \\ 1 - \alpha \end{bmatrix}$$

O in modo equivalente

$$f_1\left(r\right) = r_{11}\tilde{x} + r_{12}\tilde{y} + r_{13}\tilde{z} = {}^{c}z_1\left(\varphi_{x_1} - \alpha\varphi_{x_2}\right) \tag{6.13a}$$

$$f_2\left(r\right) = r_{21}\tilde{x} + r_{22}\tilde{y} + r_{23}\tilde{z} = {}^{c}z_1\left(\varphi_{y_1} - \alpha\varphi_{y_2}\right) \tag{6.13b}$$

$$f_3\left(r\right) = r_{31}\tilde{x} + r_{32}\tilde{y} + r_{33}\tilde{z} = {}^{c}z_1\left(1 - \alpha\right) \tag{6.13c}$$

Dove r_{ij} sono i componenti della matrice ${}^{e}R_{c}^{-1}$. notando che questo insieme di equazioni ha 11 incognite: i 9 componenti della matrice di orientamento della camera ${}^{e}R_{c}^{-1}$, la profondità del primo punto ${}^{c}z_1$ e il rapporto α tra la profondità dei due punti.
Dalle equazioni (6.13a) e (6.13c), o (6.13b) e (6.13c), due espressioni per la profondità ${}^{c}z_1$ possono essere ricavate, come segue

$${}^{c}z_1 = \frac{f_1 - \varphi_{x_2} f_3}{\varphi_{x_1} - \varphi_{x_2}} = \frac{f_2 - \varphi_{y_2} f_3}{\varphi_{y_1} - \varphi_{y_2}} \tag{6.14}$$

Inoltre dall'equazione (6.13c) segue

$$\alpha = 1 - \frac{f_3}{{}^{c}z_1}$$

E sostituendo ${}^{c}z_1$ nella relazione (6.14) si ottiene

$$f_1\left(\varphi_{y_1} - \varphi_{y_2}\right) + f_2\left(\varphi_{x_2} - \varphi_{x_1}\right) + f_3\left(\varphi_{x_1}\varphi_{y_2} - \varphi_{x_2}\varphi_{y_1}\right) = 0$$

Infine sfruttando l'uguaglianza nell'espressione (6.14), si ottiene un equazione dove le incognite α e ${}^{c}z_1$ non appaiono

$$f_1\left(\varphi_{y_1} - \varphi_{y_2}\right) + f_2\left(\varphi_{x_2} - \varphi_{x_1}\right) + f_3\left(\varphi_{x_1}\varphi_{y_2} - \varphi_{x_2}\varphi_{y_1}\right) = 0$$

Considerando una serie di n punti, con $n \geq 9$, la precedente equazione può essere riscritta come un sistema omogeneo lineare rispetto alle 9 incognite r_{ij} della matrice di rotazione

$$\begin{bmatrix} a_1\tilde{x}_1 & a_1\tilde{y}_1 & a_1\tilde{z}_1 & b_1\tilde{x}_1 & b_1\tilde{y}_1 & b_1\tilde{z}_1 & c_1\tilde{x}_1 & c_1\tilde{y}_1 & c_1\tilde{z}_1 \\ a_2\tilde{x}_2 & a_2\tilde{y}_2 & a_2\tilde{z}_2 & b_2\tilde{x}_2 & b_2\tilde{y}_2 & b_2\tilde{z}_2 & c_2\tilde{x}_2 & c_2\tilde{y}_2 & c_2\tilde{z}_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_n\tilde{x}_n & a_n\tilde{y}_n & a_n\tilde{z}_n & b_n\tilde{x}_n & b_n\tilde{y}_n & b_n\tilde{z}_n & c_n\tilde{x}_n & c_n\tilde{y}_n & c_n\tilde{z}_n \end{bmatrix} \begin{bmatrix} r_{11} \\ r_{12} \\ \vdots \\ r_{33} \end{bmatrix} = 0$$

Dove

$$a_i = \left(\varphi^i_{y1} - \varphi^i_{y2}\right) \quad b_i = \left(\varphi^i_{x2} - \varphi^i_{x1}\right) \quad c_i = \left(\varphi^i_{x1}\varphi^i_{y2} - \varphi^i_{x2}\varphi^i_{y1}\right) \quad i = 1, \ldots, n$$

Essendo r_{ij} gli elementi della matrice di rotazione essi devono rispettare il vincolo di ortonormalità e perciò. Per rendere più semplice la computazione di ${}^eR_c$, assicurando l'ortonormalità il problema si può riformulare come un ottimizzazione non lineare, le cui incognite sono gli angoli di Eulero che rappresentano l'orientamento della camera rispetto alla terna dell'end effector. Per determinare la posizione della camera ${}^eT_c$, l'equazione (6.8) può essere sfruttata. Assumendo che una serie di punti complanari, che condividono la stessa coordinata ${}^oz$, siano disponibili. allora l'equazione (6.8) può essere semplificata usando un particolare orientamento del manipolatore:

$${}^oR_e = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

In questo modo l'ultima riga dell'equazione (6.8) può essere riscritta come

$${}^oz = -\begin{bmatrix} r_{13} & r_{23} & r_{33} \end{bmatrix} {}^cp - {}^eT_{c_z} + {}^oT_{e_z}$$

Dove l'unica incognita è ${}^eT_{c_z}$. Con una procedura simile, scegliendo opportunamente la matrice di orientamento ${}^oR_e$, possono essere determinate facilmente ${}^oT_{e_x}$ e ${}^oT_{e_y}$. La procedura di calibrazione appena descritta è molto conveniente per la videocamera *eye-in-hand* perché sfrutta la cinematica diretta del manipolatore per rendere il processo relativamente veloce. I risultati ottenuti sono i seguenti:

$$\mathbf{K} = \begin{bmatrix} 5149.47 & 0 & 640 \\ 0 & 5160.94 & 512 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.15}$$

e la corrispondente matrice omogenea che mette in relazione i punti della terna camera con quelli della terna end effector:

$$[{}^eR_c|{}^eT_c] = \begin{bmatrix} 0.6948 & 0.7191 & 0.0129 & -54.4274 \\ -0.7191 & 0.6949 & -0.0016 & -50.7282 \\ -0.0101 & -0.0082 & 0.9999 & 104.2197 \end{bmatrix} \tag{6.16}$$

Si può notare come i risultati dei vari metodi siano considerabili simili entro un certo margine di errore, e che chiaramente l'accuratezza richiesta nel posizionamento del profilo nello spazio 3D non sia raggiungibile con questi sistemi di calibrazione basti osservare le coordinate del vettore ${}^eT_c$ per comprendere che la non accuratezza nel determinare l'origine dell'asse ${}^cz$ causa un errore maggiore della precisione richiesta per il sistema; Alla luce di queste considerazioni è maggiormente consigliabile l'approccio scelto di utilizzare un

controllo visuale nello spazio immagine per il posizionamento dell'utensile di sbavatura in modo da compensare eventuali errori di posizionamento legati alla calibrazione; difatti la risoluzione dello strumento resta molto alta ed è in grado di fornire in tempo reale dati al controllore per correggere con precisione la posizione.

6.4 Calibrazione laser

Considerando il presupposto che i maggiori errori introdotti nella fase di calibrazione derivano dalla imprecisione dell'oggetto di calibrazione a scacchiera, che può essere non perfettamente planare e con punti a distanza non perfettamente nota, oltre che in secondo luogo alla difficoltà di identificare la soluzione ottima del problema di minimizzazione della funzione di costo (utilizzata per identificare i parametri intrinseci ed estrinseci della camera). Si è deciso di sfruttare il più possibile le caratteristiche geometriche della misurazione basandoci sull'alta precisione di posizionamento del manipolatore robotico; l'oggetto di calibrazione è molto semplice e consiste in una scala a gradini regolari di lato 10 mm ciascuno (vedi figura 6.5). La particolarità di questo oggetto è la possibilità di osservare in un unica immagine diversi piani paralleli, la distanza è nota con precisione, in questo modo osservando in quale maniera la linea laser si riflette sui diversi piani, è possibile ottenere con precisione, compatibilmente all'accuratezza di misura dei gradini, la posizione della linea laser riflessa a distanze diverse, proiettata sul piano immagine. Si deduce che, a meno di un offset sul posizionamento relativo fra il primo gradino e la posizione dell'end effector, la distanza relativa dei vari piani paralleli è nota con grande precisione, ed è possibile definire una relazione fra questa distanza e le coordinate della linea laser rispetto al centro ottico della camera (o dell'origine della terna dell'end effector). Durante la fase preliminare si proietta la linea laser su di un piano parallelo al piano x-y della terna base, posto alla distanza di lavoro desiderata z_{work} dal centro della terna camera, utilizzando i parametri estrinseci calcolati (vedi (6.16)) si impone anche che il piano x-y della terna camera sia parallelo al piano, si regola la messa a fuoco del laser affinché, alla distanza di lavoro desiderata, lo spessore della linea sia il minore possibile, migliorando in questo modo la risoluzione della misura. Questa fase viene detta di "centratura del laser" perché, osservando l'immagine della linea laser acquisita con la videocamera (vedi figura 6.4), si procede ad allineare il fascio luminoso con la colonna centrale[1] dell'immagine ruotando il proiettore laser. Terminata questa operazione sappiamo che, trascurando effetti di distorsione delle lenti, la linea laser osservata sul piano immagine si presenta verticale nei tratti in cui si

[1]La colonna centrale dell'immagine pari a 640 [pixel] è la posizione nominale della proiezione del centro ottico della camera sul piano immagine c_x, tale scelta è supportata dai risultati di calibrazione ottenuti in (6.4).

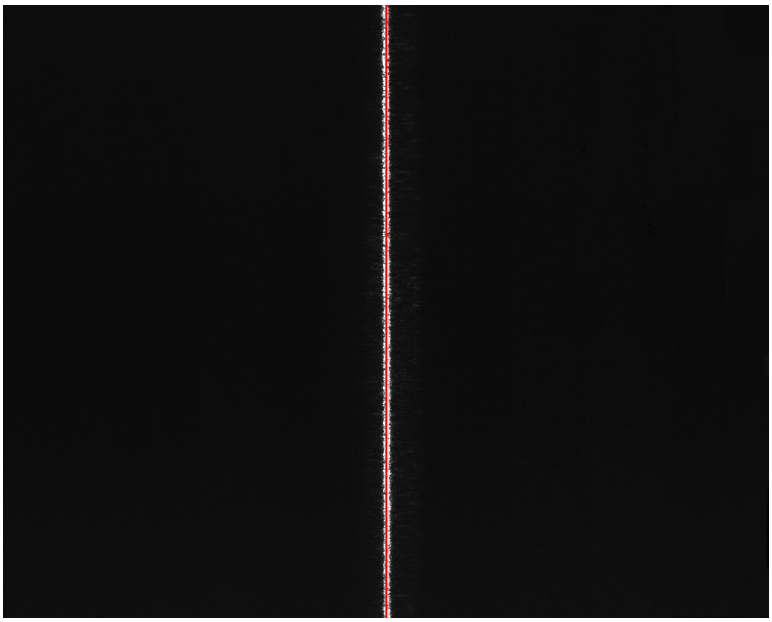


Figura 6.4: Immagine acquisita durante il processo di centratura del laser

riflette su una superficie parallela al piano x-y della terna base. In seguito si posiziona l'oggetto di calibrazione sul medesimo piano in modo che sia nota la posizione del piano zero della scala di calibrazione rispetto alla terna base del robot (terna mondo), e si posiziona il manipolatore in modo da garantire che il piano x-y della terna camera sia parallelo al piano x-y della terna base, ponendoci ad una distanza desiderata z_{work} nota dal gradino posizionato al centro dell'immagine. Dopo che il sistema è stato posizionato alla distanza di lavoro desiderata rispetto all'oggetto di calibrazione, un algoritmo di *image processing* (vedi capitolo 5), misura le coordinate immagine dei segmenti luminosi determinati dalla proiezione della linea laser su ogni gradino[2] (la figura 6.5 mostra un esempio di immagine di calibrazione).
Una volta che le coordinate immagine (ξ_x, ξ_y) di ogni segmento sono state determinate, la calibrazione è eseguita seguendo l'equazione (3.7). In particolare assumendo che l'immagine della linea sia parallela all'asse y, d_{o_c} rappresenta una distanza dalla colonna centrale dell'immagine,

$$z = \frac{f \cdot d}{f \cdot \tan(\alpha) + d_{o_c}}$$

$$d_{o_c} = -\sigma \left(\xi_x - c_x\right)$$

dove σ é la dimensione del pixel mentre c_x la coordinata x e del centro ottico sul piano immagine. l'equazione (3.7) può essere ridotta ad un equazione di

[2]la misura viene ripetuta su 100 immagini ed i dati utilizzati sono quelli mediati ottenuti.

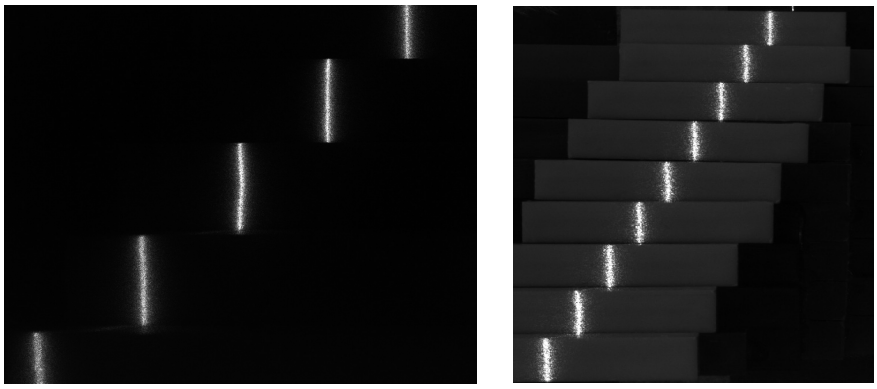


Figura 6.5: Un esempio di immagine di calibrazione del laser (immagine a sinistra) e oggetto di calibrazione (immagine a destra).

due soli parametri incogniti

$$z = \frac{K_1}{K_2 - \xi_x}$$

con

$$K_1 = \frac{d \cdot f}{\sigma} \qquad K_2 = \frac{f \tan(\alpha)}{\sigma} + c_x$$

è possibile quindi determinare K_1 e K_2 minimizzando la seguente funzione di costo

$$J(K_1, K_2) = \sum_{i=1}^{n_s} \left(z_i - \frac{K_1}{K_2 - \xi_{x_i}} \right)^2$$

dove n_s è il numero di passi misurati (vedi grafico (6.1)); i risultati così ottenuti permettono di ottenere i seguenti valori per i restanti parametri incogniti:

$$K_1 = 751320[pixel \cdot mm]$$
$$K_2 = 4275[pixel]$$

con un errore nella riproiezione dei punti inferiore a 0.015 mm.

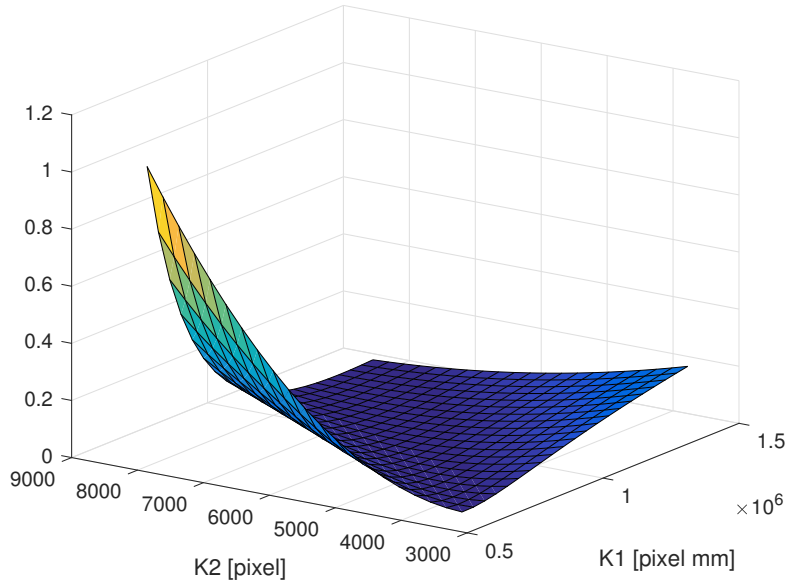


Grafico 6.1: Funzione di costo $J(K_1, K_2)$ normalizzata per la ricerca dei parametri K_1, K_2.

Capitolo 7

Risultati Sperimentali

Nel seguente capitolo si descrivono i risultati ottenuti durante l'utilizzo sperimentale dello strumento di misura. Le prove di misurazione sono state effettuate sia con una movimentazione manuale del manipolatore robotico, sia con l'ausilio del controllo visuale *contour following*; tale controllo, descritto nei capitoli precedenti (vedi paragrafo 4.3.1), svolge la funzione di mantenere la linea laser, riflessa sul bordo dell'oggetto da misurare, in posizione centrale rispetto all'immagine acquisita dalla videocamera, in questo modo è possibile rendere autonomo il processo di misurazione dell'oggetto anche se non si conosce a priori la sua esatta forma. Lo strumento a triangolazione vero e proprio è composto da una videocamera ed un proiettore a linea laser, entrambi fissati rigidamente all'end effector del manipolatore, ottenendo una configurazione nota come *eye-in-hand*; grazie alla destrezza del robot antropomorfo lo strumento di misura può essere orientato agevolmente per poter seguire il profilo del pezzo in misurazione, inquadrandone il bordo con l'angolazione desiderata.

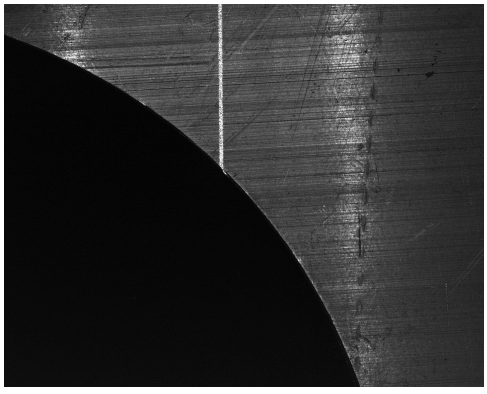


Figura 7.1: Immagine acquisita dal sistema a triangolazione durante la misurazione di un oggetto metallico con profilo curvilineo

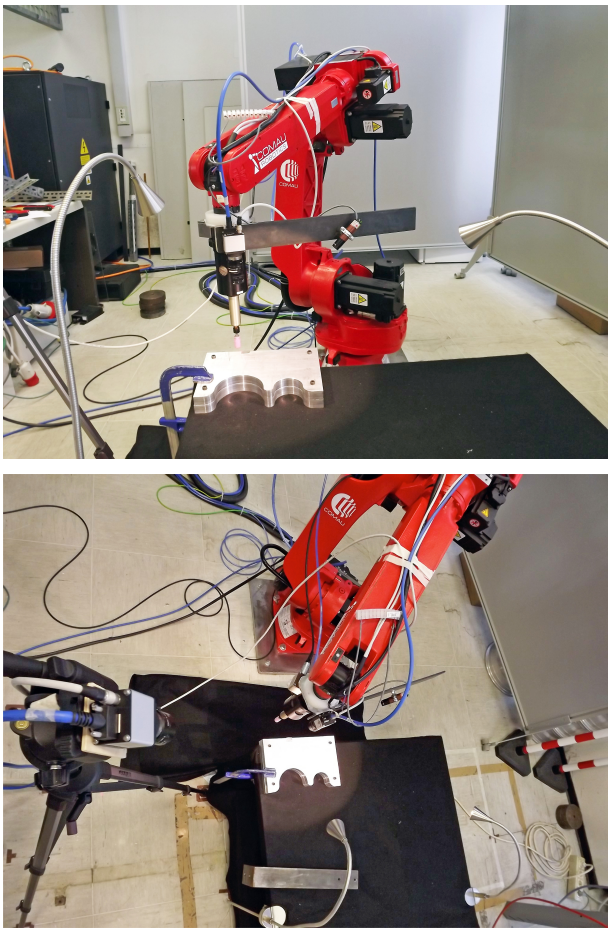


Figura 7.2: Stumento di misurazione utilizzato per l'acquisizione dei dati sperimentali [22].

7.1 Analisi dei dati di scansione: Burr Profile Analysis

Lo strumento ha una risoluzione inferiore a 300 μm ed è in grado di rilevare bave superficiali di piccole e medie dimensioni[1], i risultati sperimentali riportati nei grafici seguenti confermano i risultati teorici (vedi grafico 7.8, 7.10, 7.13, 7.16), si evidenzia infatti come lo strumento consenta di ottenere una precisa riproduzione della forma di un manufatto metallico: sia con profilo lineare (vedi grafico 7.9) che con profilo curvilineo (vedi grafico 7.1).

[1] per il calcolo della risoluzione teorica dello strumento si rimanda a 4.3.

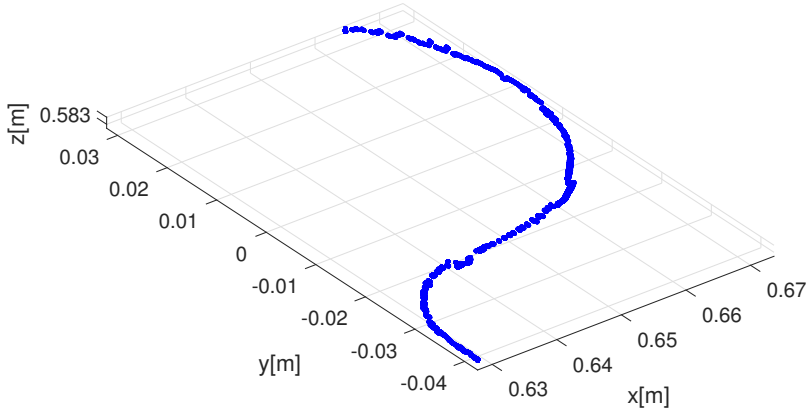


Grafico 7.1: Profilo misurato di un manufatto metallico in 3D.

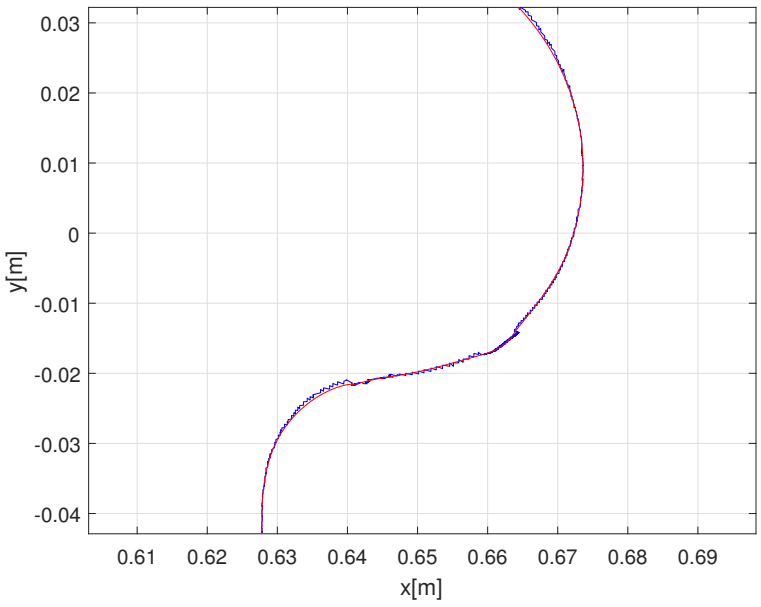


Grafico 7.2: Profilo misurato di un manufatto metallico con presenza di piccole bave.

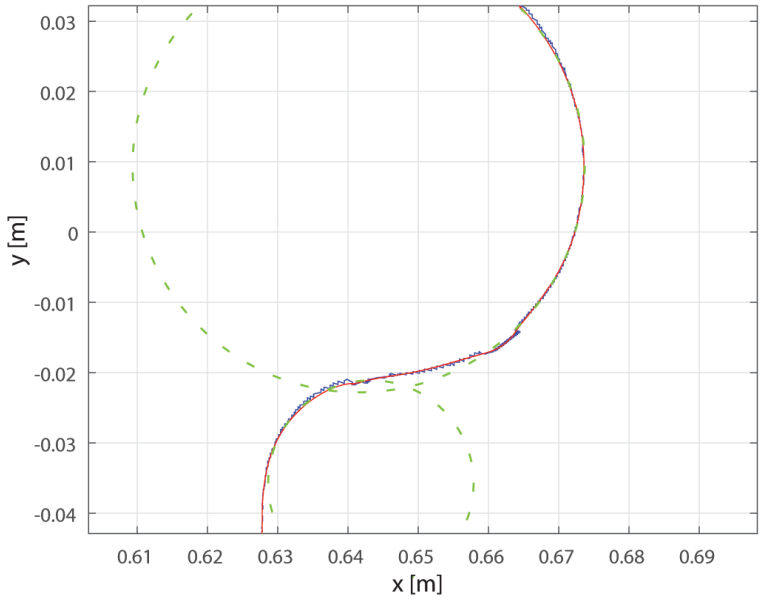


Grafico 7.3: Profilo misurato di un manufatto metallico raggio di circonferenza evidenziato.

Osservando il grafico che riporta la misurazione del profilo curvilineo del manufatto metallico (vedi grafico 7.3), si possono fare considerazioni sull'effettiva bontà della riproduzione della forma del profilo, infatti il tratto circolare che caratterizza un segmento del pezzo misurato, come si evidenzia con la linea tratteggiata di colore verde riportata sul grafico, corrisponde in maniera evidente alla misurazione effettuata. Inoltre, sempre dall'osservazione di questi dati è possibile notare come il sistema sia in grado di discriminare la presenza di bave di piccole dimensioni (inferiori ai 700 μm) presenti su di un profilo di forma curvilinea non nota a priori, (vedi grafico 7.8 e figura 7.3).

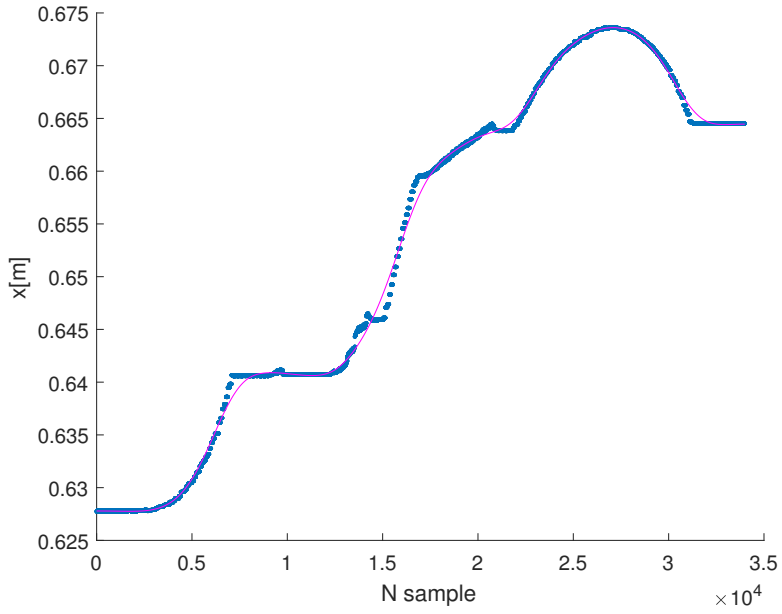


Grafico 7.4: Misure della coordinata x in funzione del sample di acquisizione (misurazione di un profilo curvilineo di manufatto metallico)(dati in blu)(filtro spline robusto in rosso).

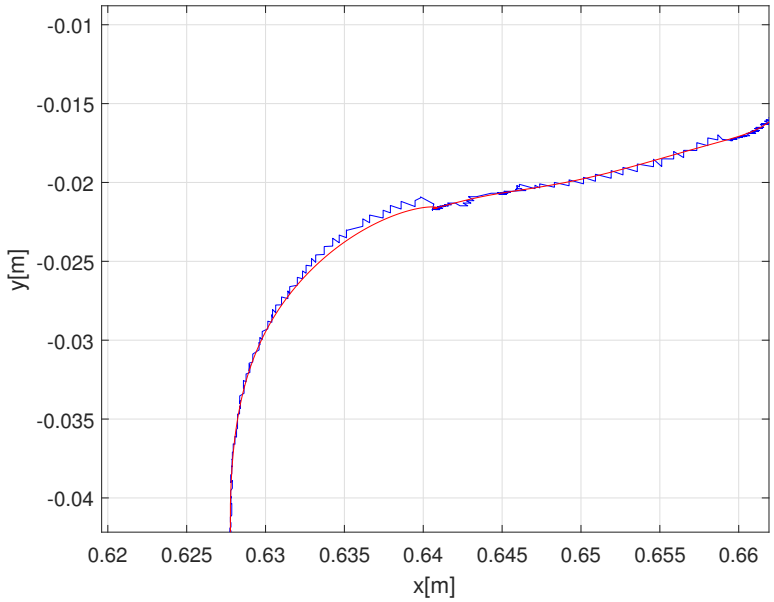


Grafico 7.7: Dettaglio bava di piccole dimensioni rilevata sul profilo curvilineo: dati filtrati con spline robusto (linea rossa), dati direttamente acquisiti (blu)

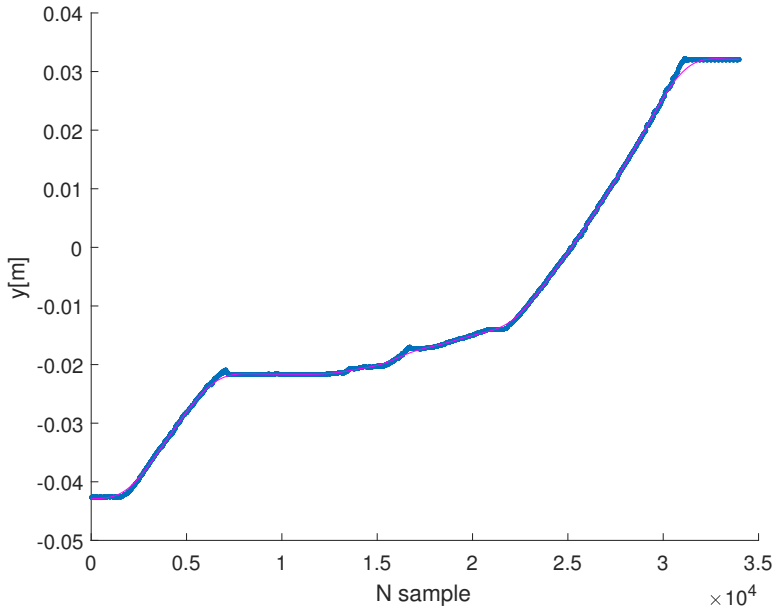


Grafico 7.5: Misure della coordinata y in funzione del sample di acquisizione (misurazione di un profilo curvilineo di manufatto metallico)(dati in blu)(filtro spline robusto in rosso).

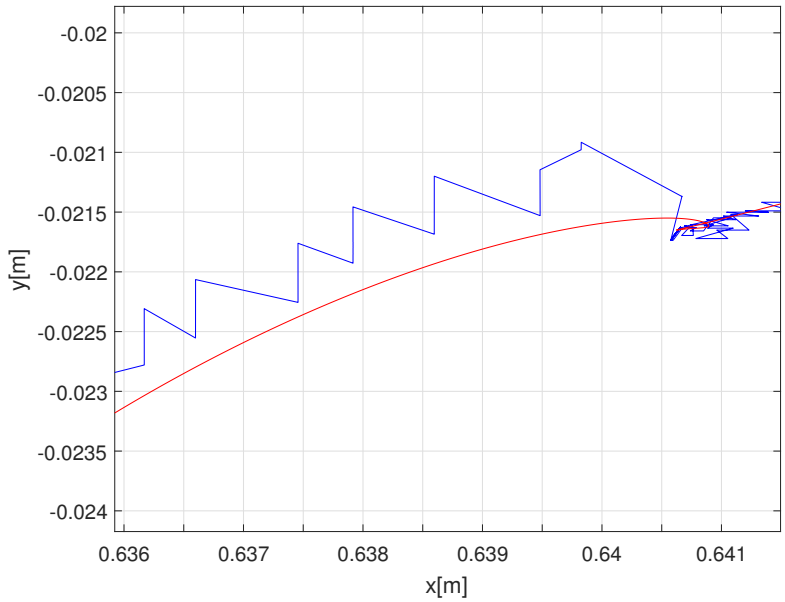


Grafico 7.8: Dettaglio misura dell'altezza di bava: dati filtrati con spline robusto (linea rossa), dati direttamente acquisiti (blu)

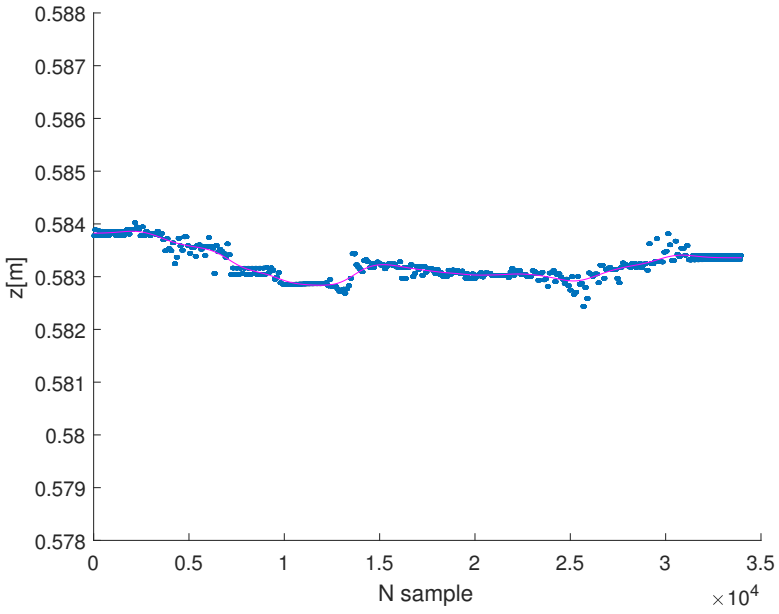


Grafico 7.6: Misure della coordinata z in funzione del sample di acquisizione (misurazione di un profilo curvilineo di manufatto metallico)(dati in blu)(filtro spline robusto in rosso).

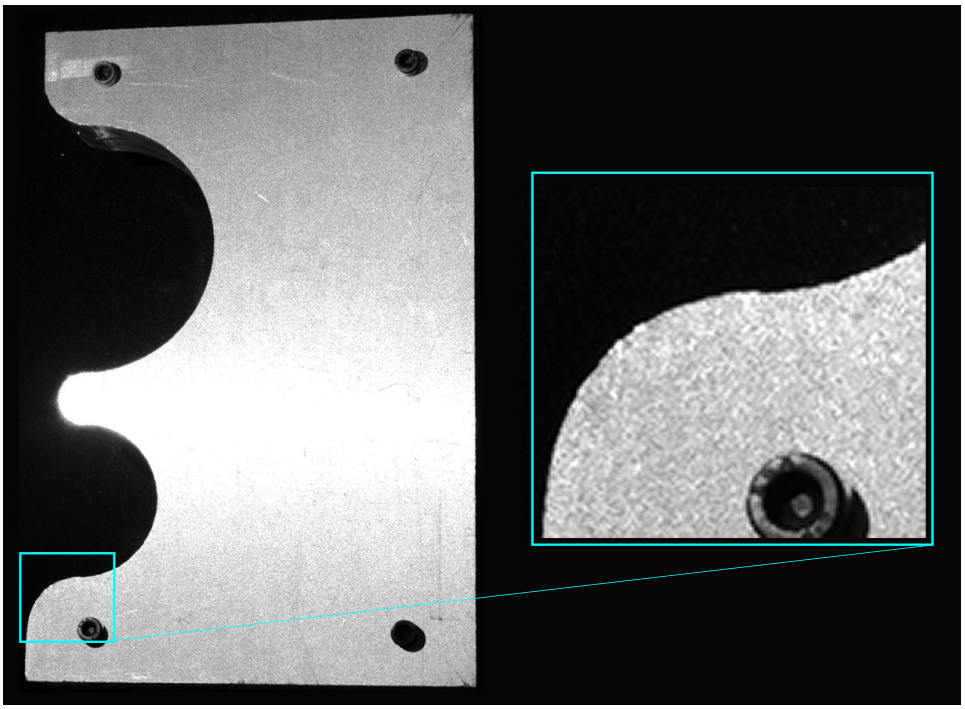


Figura 7.3: Immagine del manufatto con profilo curvilineo misurato, dettaglio della bava di piccole dimensioni rilevata.

La linea rossa presente nei grafici delle misurazioni effettuate sul profilo curvilineo, è ottenuta filtrando i dati di misura di partenza delle singole coordinate con un filtro *spline robusto* [11]; utilizzando un opportuno parametro, determinato sperimentalmente, si può determinare l'intensità dell'effetto "lisciante", tale filtro è detto robusto perché permette di eliminare eventuali valori anomali presenti nella sequenza di dati raccolti, in modo che il profilo determinato non sia condizionato da singoli valori molto diversi da quelli della restante misurazione. L'utilizzo che viene fatto in questo caso dell'operazione di filtraggio è leggermente differente, usualmente si utilizza per effettuare la rimozione del rumore ad alta frequenza presente nei dati di misurazione, ma in questa applicazione lo si utilizza per ottenere una stima del profilo nominale dell'oggetto misurato, la caratteristica di robustezza permette alla stima di non essere condizionata da singoli valori anomali, quindi ipotizzando che il profilo abbia una forma caratterizzata da tratti lineari o circolari od ancora curvilinei ma con un andamento regolare, si può sfruttare l'informazione di distanza lungo la normale al profilo filtrato per identificare punti che si discostano per un certo tratto da quest'ultimo, i tratti così rilevati possono essere classificati come imperfezioni superficiali ed essere misurati, in questo modo si ottiene una prima approssimazione della misura dell'altezza di bava.

Un'ulteriore caratteristica dello strumento è la ripetibilità di misura, osservando i dati riportati (vedi grafico 7.10) si osservano misurazioni successive del medesimo profilo lineare che presenta una bava artificiale di media grandezza, essa oltre ad essere facilmente discriminabile dal profilo nominale dell'oggetto, viene caratterizzata con precisione, determinandone la distanza dal profilo nominale sia lungo la coordinata x che lungo quella z.

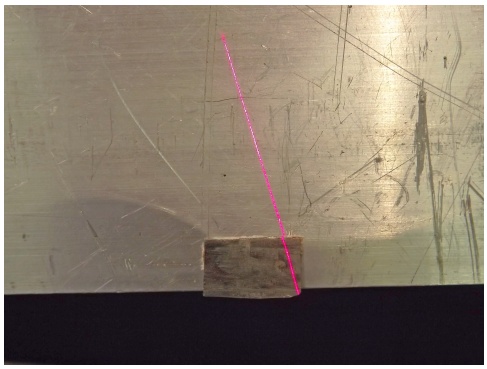


Figura 7.4: Immagine del bordo lineare con bava artificiale di medie dimensioni, (vedi grafico 7.10).

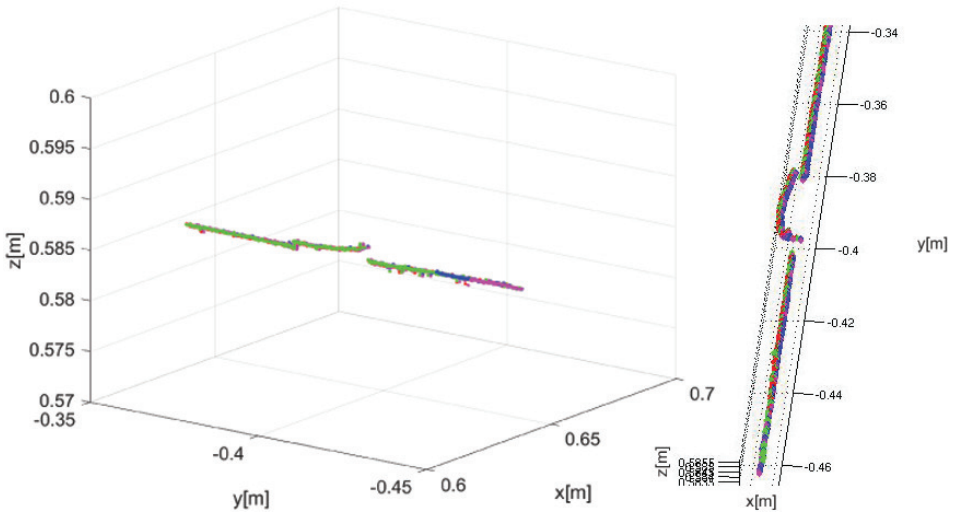


Grafico 7.9: Profilo misurato di un bordo lineare con bava artificiale di medie dimensioni in 3D; i diversi colori indicano misure effettuate in scansioni differenti.

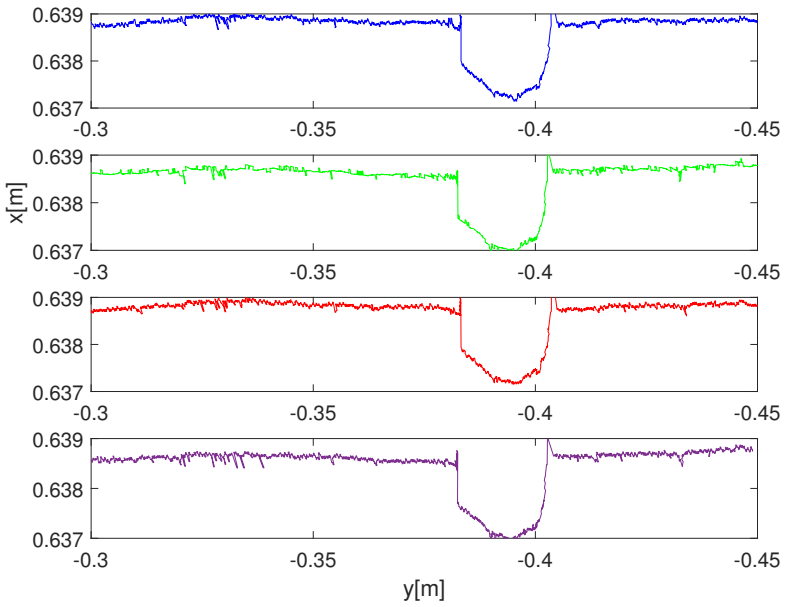


Grafico 7.10: Scansioni ripetute del profilo di sbavatura.

Della medesima bava artificiale si riportano le misurazioni ottenute prima e dopo l'operazione di sbavatura. Inizialmente è caratterizzata da un'altezza di bava di circa 2 mm protesa al di fuori del bordo del pezzo (vedi 7.4), e presenta un rilievo dal piano superiore del manufatto di circa 1 mm, per una larghezza di circa 2 cm; confrontando le misurazioni effettuate prima della sbavatura (vedi grafico 7.9 e 7.10), con quelle effettuate dopo il processo di sbavatura (vedi grafico 7.12 e 7.13), si osserva che lo strumento è in grado di evidenziare ancora la presenza di un'imperfezione sul bordo del pezzo, distinguibile in maniera netta dai dati forniti, ed identificabile come un gradino nella direzione dell'asse x di circa $500\,\mu$m, a conferma della bontà della misura si possono confrontare i grafici che riportano le misure della coordinata z in funzione della coordinata y della medesima scansione (vedi grafico 7.15 e 7.16), i dati mostrano che il rilievo del gradino rilevato sormonta il piano superiore dell'oggetto di circa 1 mm.

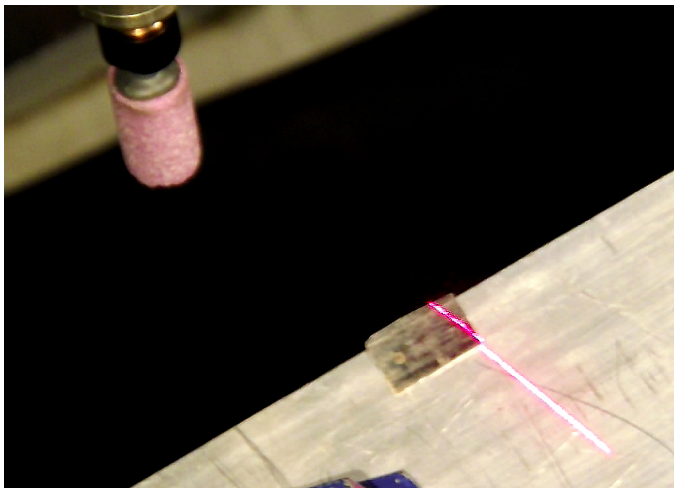


Figura 7.5: Scansione profilo di un bordo lineare con bava artificiale di medie dimensioni.

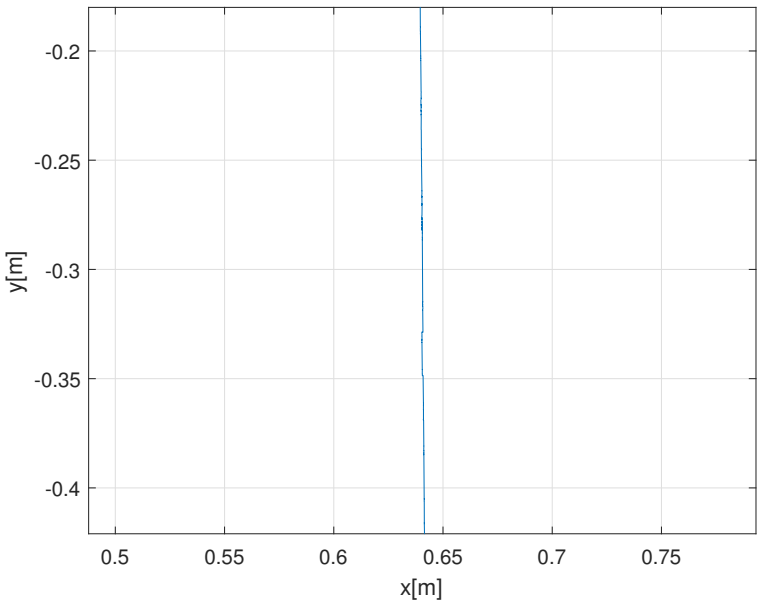


Grafico 7.11: Profilo misurato dopo aver eseguito la procedura di sbavatura (vista dall'alto).

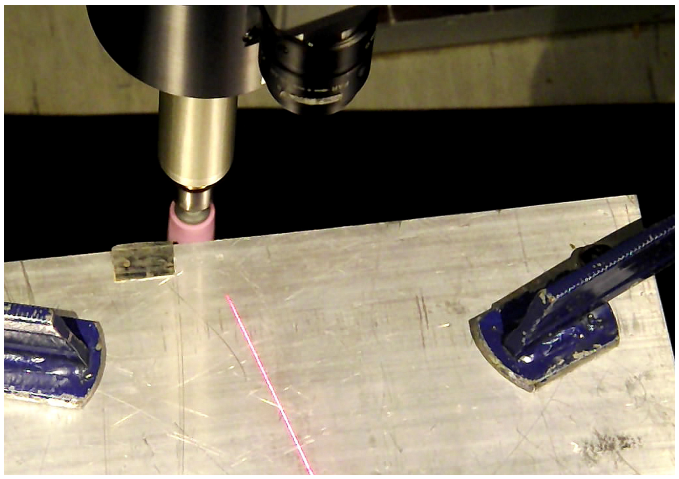


Figura 7.6: Sbavatura profilo di un bordo lineare con bava artificiale di medie dimensioni.

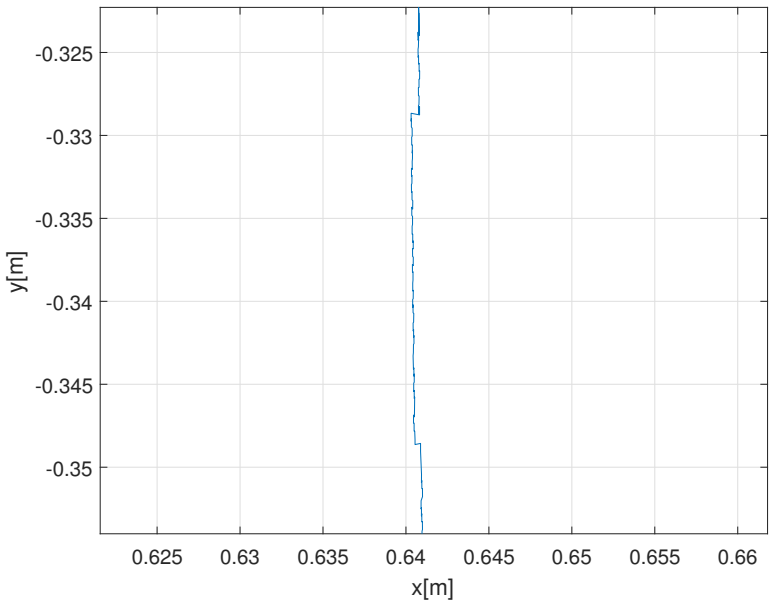


Grafico 7.12: Dettaglio del profilo dopo aver eseguito la sbavatura (vista dall'alto).

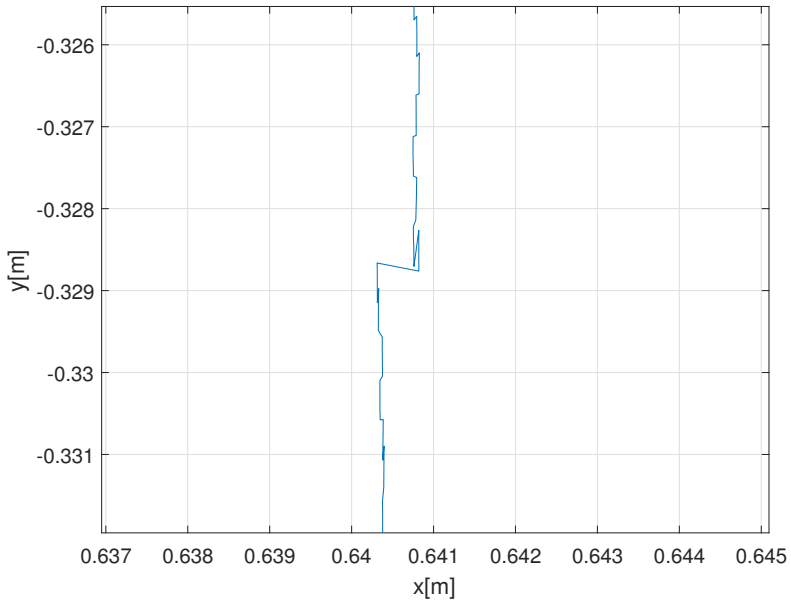


Grafico 7.13: Dettaglio risoluzione misura dell'imperfezione superficiale (vista dall'alto).

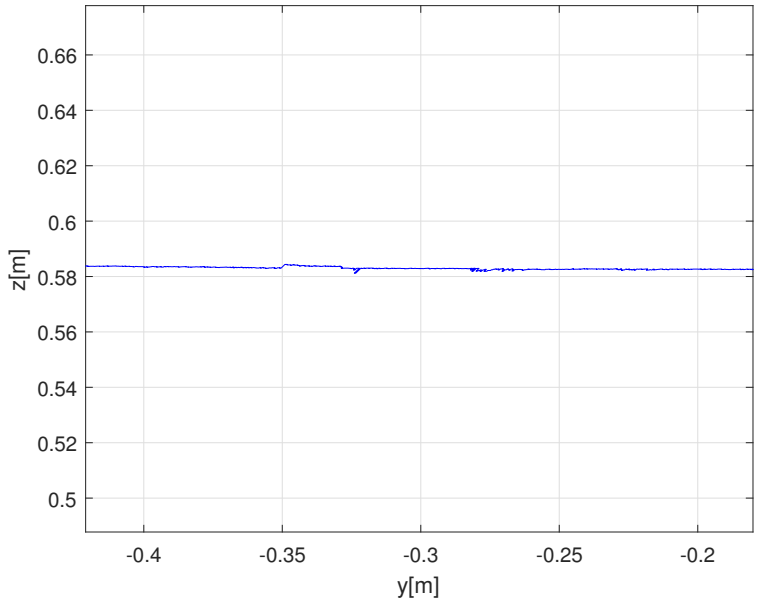


Grafico 7.14: Profilo misurato dopo aver eseguito la procedura di sbavatura (vista laterale).

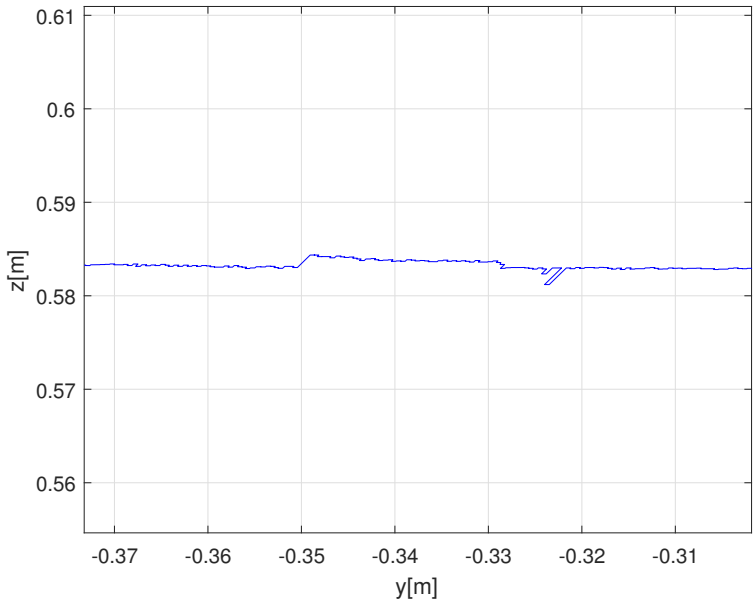


Grafico 7.15: Dettaglio del profilo dopo aver eseguito la sbavatura (vista laterale).

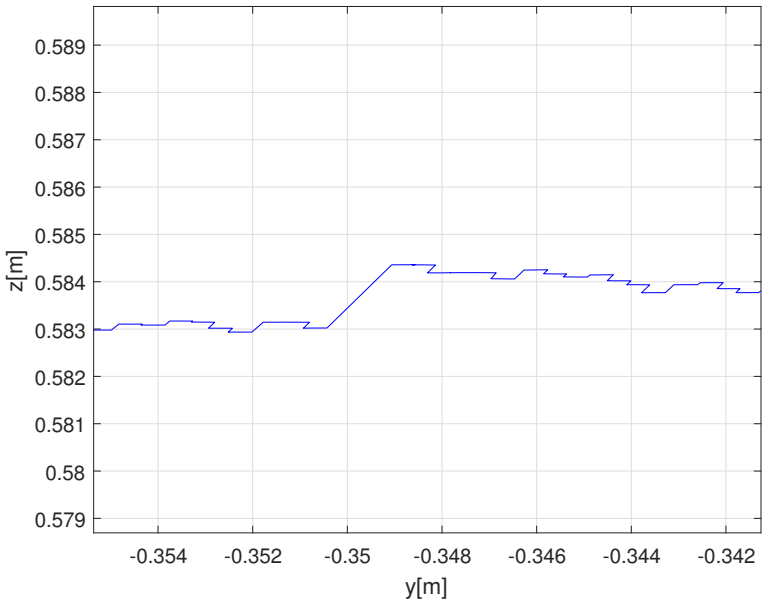


Grafico 7.16: Dettaglio risoluzione misura dell'imperfezione superficiale (vista laterale).

7.2 Analisi del profilo di sbavatura eseguito con Contour Following

A conclusione della presentazione dei risultati sperimentali ottenuti tramite il sistema progettato e realizzato presso il Laboratorio di Meccatronica e Robotica per l'Innovazione (*MERLIN*) nella sede del Dipartimento di Elettronica, Informazione e Bioingegneria (*DEIB*) del Politecnico di Milano, si riporta la misurazione del medesimo tratto curvilineo, in precedenza misurato con il controllo manuale del manipolatore (vedi 7.2), ora misurato autonomamente grazie al controllo visuale ibrido nello spazio delle immagini e nello spazio operativo ad inseguimento del contorno (vedi grafico 7.17); come si può notare il risultato della misure presenta una maggiore rumorosità dei dati, intesa come oscillazione nell'intorno del profilo nominale, ma ciononostante descrive in maniera precisa e soddisfacente il contorno del manufatto misurato.

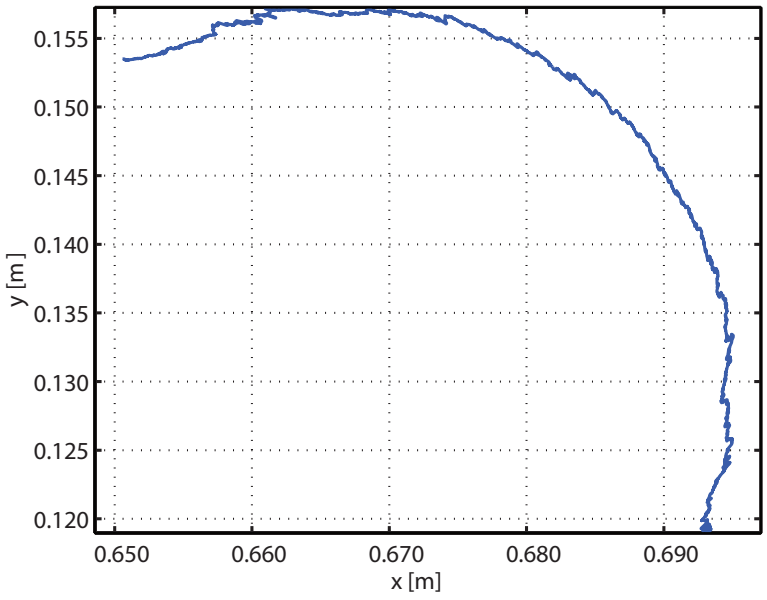


Grafico 7.17: Profilo misurato di un manufatto metallico (vista dall'alto), misurazione ottenuta tramite controllo *contour following*.

Capitolo 8

Conclusioni

Il lavoro svolto in questo progetto di tesi ha portato alla realizzazione di uno strumento di misurazione di superfici 3D con una risoluzione inferiore a $300\,\mu$m. Il raggiungimento di tale obiettivo, imposto dalle specifiche iniziali, è dimostrato dai risultati sperimentali riportati nella presente tesi, inoltre l'elevata capacità di discriminare punti illuminati sulla superficie del pezzo ha permesso di identificare la presenza di bave di piccole e medie dimensioni sul bordo degli oggetti misurati. L'utilizzo di un controllo visuale ibrido *contour following* è stato sperimentato con successo per l'applicazione in esame, è infatti stato ritenuto un valido approccio al controllo di un manipolatore antropomorfo, al fine di eseguire un *task* complesso in autonomia. L'avvicinamento ad un oggetto di forma non nota ed il successivo contatto controllato con la superficie dello stesso, sono compiti che il manipolatore deve essere in grado di compiere per poter svolgere l'operazione di sbavatura automatizzata, lo svolgimento ottimale di questi compiti però ha richiesto una conoscenza precisa dello spazio di lavoro e delle caratteristiche dell'oggetto da sottoporre a sbavatura, ed è per questo motivo che è stato implementato un sistema di misurazione senza contatto a triangolazione laser che permettesse di ottenere una buona stima della posizione del profilo di sbavatura, senza dover entrare in contatto con l'oggetto durante la fase preliminare di acquisizione dei dati. In seguito, i dati raccolti sono stati sfruttati per un successivo processo di sbavatura automatizzato, non trattato in questa tesi. L'obbiettivo importante realizzato nello svolgimento del presente lavoro è stato il perfezionamento di una procedura di *image analisys* sviluppata e validata sperimentalmente, che permettesse l'identificazione dell'esatta posizione della linea laser riflessa dal profilo di sbavatura, e rilevata dai sensori della videocamera, in modo da poter sfruttare il sistema di triangolazione anche in ambienti con presenza di disturbi luminosi e con oggetti di forma complessa come i cerchioni automobilistici. La problematica della calibrazione della videocamera *eye-in-hand* è stata affrontata sfruttando i più recenti metodi di calibrazione disponibili, ed evidenziandone i limiti nell'accuratezza dell'identificazione dei parametri

intrinseci ed estrinseci, ci si riserva perciò l'intento futuro di sfruttare la caratteristica di alta risoluzione dello strumento per lo sviluppo di soluzioni innovative che riducano i possibili errori sistematici introdotti. L'algoritmo di *image analysis* sviluppato è stato validato sperimentalmente ed ha permesso di ottenere le misure dei profili di sbavatura di oggetti di forma complessa, senza che fosse riscontrata la presenza di un numero rilevante di misure non conformi alla forma dell'oggetto. Tale algoritmo è stato impiegato movimentando il manipolatore robotico sia manualmente che con l'ausilio del controllo ad inseguimento del contorno. Si può aggiungere che la videocamera esterna in configurazione *eye-to-hand* può essere sfruttata per la localizzazione iniziale della posa dell'oggetto da misurare, riferita alla terna base del manipolatore, in questo modo è possibile rendere automatico anche il processo di avvicinamento all'oggetto da misurare. Lo strumento realizzato, grazie alle sue caratteristiche di flessibilità e precisione, può essere impiegato in campi anche molto diversi da quelli legati al processo di sbavatura; per esempio il sistema proposto può essere sfruttato nel processo di *Reverse Engineering*, che richiede la creazione di modelli 3D che riproducano fedelmente oggetti di forma complessa e non nota a priori.

Appendice A

Visual servoing

Si riporta in seguito una sezione del codice sorgente, eseguito su PC *real-time* adibito al controllo del manipolatore Comau Smart Six, per poter implementare il controllo *contour following*; per poter sviluppare la legge di controllo che interagisce ad un livello superiore con il controllo dell'architettura aperta C4GOpen, basata sull'unità di governo Comau C4G, è necessario entrare in una particolare modalità denominata MODALITY 5

```
// Executing MODALITY 5...
if (isOpenControllerActive)
{
    //
        ////////////////////////////////////////////////////////////////////////////////

    // Update variables
    //
        ////////////////////////////////////////////////////////////////////////////////

    t = (double)rt_get_time_ns()/1.0e9-t0;

    // Get a set of features from the shared memory
    double s_features_old[VISSERV_MSG_SIZE];
    memcpy(s_features_old, s_features, VISSERV_MSG_SIZE*sizeof(double));

    rt_sem_wait(shrdmem_mutex);
    memcpy(s_features, shrdmem_ptr, VISSERV_MSG_SIZE*sizeof(double));
    rt_sem_signal(shrdmem_mutex);

    if (fabs(s_features[0]-s_features_old[0])>=VISSERV_MAXDELTA_X)
    {
        s_features[0] = s_features_old[0];
        printf("x_feature_increment_too_large.\n");
    }
    if (fabs(s_features[1]-s_features_old[1])>=VISSERV_MAXDELTA_Y)
    {
        s_features[1] = s_features_old[1];
        printf("y_feature_increment_too_large.\n");
    }
    if (fabs(s_features[2]-s_features_old[2])>=VISSERV_MAXDELTA_THETA)
    {
        s_features[2] = s_features_old[2];
        printf("theta_feature_increment_too_large.\n");
    }
```

```
    if (fabs(s_features[3]-s_features_old[3])>=VISSERV_MAXDELTA_Z)
33  {
        s_features[3] = s_features_old[3];
35      printf("z_feature_increment_too_large.\n");
    }
37
    // Retrieve joint/axis/current data from C4Gopen
39  for (axOpenIdx = 0; axOpenIdx < sx_sync.nAxOpen; axOpenIdx ++)
    {
41      qComauAxis_ref[axOpenIdx] = sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D1;
        dqComauAxis_ref[axOpenIdx] = sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D2;
43      qComauAxis_act[axOpenIdx] = sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D3;
        dqComauAxis_act[axOpenIdx] = sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D4;
45      motorCurrent_ref[axOpenIdx] = sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D5;
    }
47  axis_to_joint_pos( qComauJoint_ref, qComauAxis_ref);
    axis_to_joint_vel(dqComauJoint_ref, dqComauAxis_ref);
49  axis_to_joint_pos( qComauJoint_act, qComauAxis_act);
    axis_to_joint_vel(dqComauJoint_act, dqComauAxis_act);
51
    // Compute the tool frame orientation wrt the base frame
53  pose_matrix tool_mpose;
    pose_vector tool_vpose;
55  forward_kinematics(qComauJoint_act, tool_mpose);
    pose_mat2vect_eulZYZ(&tool_vpose, tool_mpose);
57  //
        ////////////////////////////////////////////////////////////////////////////////


59  //
        ////////////////////////////////////////////////////////////////////////////////

    // Compute the visual servoing control
61  //
        ////////////////////////////////////////////////////////////////////////////////

    translation_vector camera_T = CAMERA_WRIST_T;
63  orientation_matrix camera_R = CAMERA_WRIST_R;

65  // Set points
    const double s_x_ref = 0.0;
67  const double s_y_ref = 0.0;
    const double theta_ref = 10*M_PI/180.0;
69  const double z_ref = 194.5e-3;

71  // Controlled variables
    double s_x = s_features[0];
73  double s_y = s_features[1];
    double theta = s_features[2];
75  double z = s_features[3];

77  // Control variables in camera frame
    double vcz, wcz, vcx, vcy;
79  if ((fabs(0.8429-tool_vpose.pos_z)>0.025) && approaching_working_zone)
    {
81      vcz = VISSERV_KP_VCZ*(tool_vpose.pos_z-0.843);
        vcx = vcy = wcz = 0.0;
83  }
    else
85  {
        approaching_working_zone = false;
87
```

```
        vcz = VISSERV_KP_VCZ*(z-z_ref);
89      wcz = VISSERV_KP_WCZ*(theta-theta_ref);
        vcx = (z*VISSERV_KP_S*(s_x-s_x_ref) + z*s_y*wcz + s_x*vcz)/
              CAMERA_WRIST_FX;
91      vcy = (z*VISSERV_KP_S*(s_y-s_y_ref) - z*s_x*wcz + s_y*vcz)/
              CAMERA_WRIST_FY;
    }
93
    // Update the setpoint to generate motion
95  if ((fabs(theta_ref-theta)<=VISSERV_ERR_ANGLE_TH) && (fabs(z_ref-z)<=
        VISSERV_ERR_Z_TH))
    {
97      vcx += VISSERV_SP_INC*cos(theta);
        vcy += VISSERV_SP_INC*sin(theta);
99  }

101 double v_cameraframe[3] = {vcx, vcy, vcz};
    double w_cameraframe[3] = {0.0, 0.0, wcz};
103
    // Control variables in absolute frame
105 translation_vector tool_T;
    orientation_matrix tool_R;
107 pose_mat_getRT(tool_mpose, tool_R, tool_T);

109 double v_absframe[3], v_eeframe[3];
    matrix_vect_mult(&(camera_R[0][0]), 3, 3, &(v_cameraframe[0]), &(v_eeframe[0]));
111 matrix_vect_mult(&(tool_R[0][0]), 3, 3, &(v_eeframe[0]), &(v_absframe[0]));

113 double w_absframe[3], w_eeframe[3];
    matrix_vect_mult(&(camera_R[0][0]), 3, 3, &(w_cameraframe[0]), &(w_eeframe[0]));
115 matrix_vect_mult(&(tool_R[0][0]), 3, 3, &(w_eeframe[0]), &(w_absframe[0]));

117 // Compensate for rotation w
    double camera_T_absframe[3], w_correction_absframe[3];
119 matrix_vect_mult(&(tool_R[0][0]), 3, 3, &(camera_T[0]), &(camera_T_absframe[0]));

121 vect_vect_cross(&(w_absframe[0]), &(camera_T_absframe[0]), &(w_correction_absframe
        [0]));
    v_absframe[0] -= w_correction_absframe[0];
123 v_absframe[1] -= w_correction_absframe[1];
    v_absframe[2] -= w_correction_absframe[2];
125
    // From velocity to delta_pose
127 cart_linearVel linVel = {v_absframe[0]*C4GOPEN_TSAMP, v_absframe[1]*
        C4GOPEN_TSAMP,
                              v_absframe[2]*C4GOPEN_TSAMP
129                          };
    cart_angularVel angVel = {w_absframe[0]*C4GOPEN_TSAMP, w_absframe[1]*
        C4GOPEN_TSAMP,
131                           w_absframe[2]*C4GOPEN_TSAMP
                             };
133
    // From delta_pose to joint displacement
135 joint_vector dqComauJoint;
    inverse_diffkinematics(qComauJoint_act, dqComauJoint, linVel, angVel);
137
    // Convert x,y laser data in world frame
139 double laser_cameraframe[3] = {s_features[4], s_features[5], s_features[3]};

141 double laser_absframe[3];
    if ((laser_cameraframe[0]==(double)NAN) || (laser_cameraframe[1]==(double)NAN))
143 {
```

```
            laser_absframe[0] = (double)NAN;
            laser_absframe[1] = (double)NAN;
            laser_absframe[2] = (double)NAN;
        }
        else
            cameraframe_to_worldframe(&(laser_absframe[0]), &(laser_cameraframe[0]),
                                      camera_T, camera_R, tool_T, tool_R);
        //
            ////////////////////////////////////////////////////////////////////////////////


        joint_vector dqComauAxis;
        if (abnormal_behaviour)
            memset(dqComauAxis, 0, 6*sizeof(double));
        else
            joint_to_axis_vel(dqComauJoint, dqComauAxis);

        if ((mod5_cnt % 500)==0)
        {
            printf("time: %.3e\n", t);
            printf("features: x_cam [px] %.3e, y_cam [px] %.3e, theta_cam [deg] %.3e\n",
                s_features[0], s_features[1], s_features[2]*180.0/M_PI);
            printf("features: z_laser %.3e, x_laser %.3e, y_laser %.3e\n", s_features[3],
                s_features[4], s_features[5]);
            printf("feature errors: x %.3e, y %.3e, angle %.3e, z %.3e\n", s_x-s_x_ref, s_y-
                s_y_ref, theta-theta_ref, z-z_ref);
            printf("v_c_camera: %.3e, %.3e, %.3e\n", v_cameraframe[0], v_cameraframe[1],
                v_cameraframe[2]);
            printf("v_c_ee: %.3e, %.3e, %.3e\n", v_eeframe[0], v_eeframe[1], v_eeframe[2]);
            printf("v_c_abs: %.3e, %.3e, %.3e\n", v_absframe[0], v_absframe[1], v_absframe[2]);
            printf("w_c_camera: %.3e, %.3e, %.3e\n", w_cameraframe[0], w_cameraframe[1],
                w_cameraframe[2]);
            printf("w_c_abs: %.3e, %.3e, %.3e\n", w_absframe[0], w_absframe[1], w_absframe
                [2]);
            printf("dq_joint: %.3e %.3e %.3e %.3e %.3e %.3e\n", dqComauJoint[0],
                dqComauJoint[1],
                  dqComauJoint[2], dqComauJoint[3], dqComauJoint[4], dqComauJoint[5]);
            printf("dq_axis: %.3e %.3e %.3e %.3e %.3e %.3e\n", dqComauAxis[0], dqComauAxis
                [1],
                  dqComauAxis[2], dqComauAxis[3], dqComauAxis[4], dqComauAxis[5]);
            printf("\n");
        }


        for (axOpenIdx = 0; axOpenIdx < sx_sync.nAxOpen; axOpenIdx++)
        {
            sx_C4GOpen_Packet_Tx.AX[axOpenIdx].SM = C4G_MOD_5;

            // Update open axis positions and velocities
            sx_C4GOpen_Packet_Tx.AX[axOpenIdx].D1 = qComauAxis_ref[axOpenIdx] +
                dqComauAxis[axOpenIdx];
            sx_C4GOpen_Packet_Tx.AX[axOpenIdx].D2 = dqComauAxis[axOpenIdx];

            // Update the following error and check weather it is within the specified bounds
            af_PosErr[axOpenIdx] = (sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D1 -
                sx_C4GOpen_Packet_Rx.AX[axOpenIdx].D3);

            if (af_PosErr[axOpenIdx] < 0)
            {
                if (af_PosErr[axOpenIdx] < (-af_PosErrMax[axOpenIdx]))
                    sx_C4GOpen_Packet_Tx.AX[axOpenIdx].SM =
                        C4G_FOLLOWING_ERROR;
```

```
193         }
            else
195         {
                if (af_PosErr[axOpenIdx] > ( af_PosErrMax[axOpenIdx]))
197                 sx_C4GOpen_Packet_Tx.AX[axOpenIdx].SM =
                        C4G_FOLLOWING_ERROR;
            }
199     }

201     // Send data to datalogger
        // [1] t, [2-7] qComauJoint_ref, [8-13] qComauJoint_act, [14-19] qComauAxis_ref,
203     // [20-25] qComauAxis_act, [26-31] dqComauJoint_ref, [32-37] dqComauJoint_act,
        // [38-43] dqComauAxis_ref, [44-49] dqComauAxis_act, [50-55] motorCurrent_ref,
205     // [56] tool_pos_x, [57] tool_pos_y, [58] tool_pos_z, [59] tool_phi, [60] tool_theta, [61]
            tool_psi,
        // [62-65] s_features_ref, [66-69] s_features, [70-72] laser_absframe, [73-75] v_camera,
207     // [76-78] w_camera
        datalog_msg msg;
209     msg.cmd = DATALOG_NONE;
        msg.numVar = 78;
211
        msg.varVector[0] = t;
213
        int k;
215     for (k=0; k<6; k++)
        {
217         msg.varVector[ 1+k] = qComauJoint_ref[k];
            msg.varVector[ 7+k] = qComauJoint_act[k];
219         msg.varVector[13+k] = qComauAxis_ref[k];
            msg.varVector[19+k] = qComauAxis_act[k];
221         msg.varVector[25+k] = dqComauJoint_ref[k];
            msg.varVector[31+k] = dqComauJoint_act[k];
223         msg.varVector[37+k] = dqComauAxis_ref[k];
            msg.varVector[43+k] = dqComauAxis_act[k];
225         msg.varVector[49+k] = motorCurrent_ref[k];
        }
227
        msg.varVector[55] = tool_vpose.pos_x;
229     msg.varVector[56] = tool_vpose.pos_y;
        msg.varVector[57] = tool_vpose.pos_z;
231     msg.varVector[58] = tool_vpose.phi;
        msg.varVector[59] = tool_vpose.theta;
233     msg.varVector[60] = tool_vpose.psi;
        msg.varVector[61] = s_x_ref;
235     msg.varVector[62] = s_y_ref;
        msg.varVector[63] = theta_ref;
237     msg.varVector[64] = z_ref;

239     for (k=0; k<4; k++)
            msg.varVector[65+k] = s_features[k];
241     for (k=0; k<3; k++)
        {
243         msg.varVector[69+k] = laser_absframe[k];
            msg.varVector[72+k] = v_cameraframe[k];
245         msg.varVector[75+k] = w_cameraframe[k];
        }
247
        if (rt_mbx_send_if(mbx, &msg, sizeof(msg)) < 0)
249         printf("Error sending message to datalogger.\n");

251     // Update modality 5 counter
        mod5_cnt++;
```

```
253 }

255 // Exiting MODALITY 5...
    if (stop_mod5 || abnormal_behaviour)
257 {
        isOpenControllerActive = false;
259
        // Send the EXIT FROM OPEN command
261     sx_C4GOpen_Packet_Tx.INFPAR = C4G_EXIT_FROM_OPEN;
        for (axOpenIdx = 0; axOpenIdx < sx_sync.nAxOpen; axOpenIdx++)
263     {
            sx_C4GOpen_Packet_Tx.AX[axOpenIdx].SM = C4G_MOD_5;
265         sx_C4GOpen_Packet_Tx.AX[axOpenIdx].D1 = sx_C4GOpen_Packet_Rx.AX[
                axOpenIdx].D1;
            sx_C4GOpen_Packet_Tx.AX[axOpenIdx].D2 = sx_C4GOpen_Packet_Rx.AX[
                axOpenIdx].D2;
267     }
    }
269 break;

271 default:
            printf("\tModality␣%d␣not␣available...␣\n", (int)sx_C4GOpen_Packet_Rx.AX[0].SM)
                ;
273         break;
```

Appendice B

Codice sorgente algoritmi sviluppati

B.1 Linguaggio C

B.1.1 Algoritmo di image analysis principale (find_profile.c)

Questo algoritmo viene eseguito sul PC non real-time, e consente l'acquisizione di un'immagine dalla videocamera del sistema a triangolazione laser, ne esegue l'elaborazione grafica e ne ricava le coordinate dei punti di interesse in terna camera, inviando tramite socket TCP/IP le informazioni di posizione del punto di interesse per il controllo *contour following* sul piano immagine e la posizione del punto illuminato dalla linea laser sul profilo di sbavatura in terna camera (per i dettagli delle funzioni vedi (B.1.2)).

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include <cv.h>
#include <highgui.h>

#include "../header/const.h"
#include "ids_camera.h"
#include "image_processing.h"


int main()
{
    // Initialize the camera
    camera_param cam_param;
    if (idsCameraInit(&cam_param, CAMERA_WRIST_ID, IS_CM_MONO8,
          IS_SET_TRIGGER_SOFTWARE, CAMERA_WRIST_CONFIGFILE)<0)
    {
        printf("Error initialising the camera.\n");
        exit(0);
    }
    else
        printf("Camera initialised.\n");

    // Acquire the first frame to tune parameters
```

```
    if (idsCameraAcquire(&cam_param)<0)
28  {
        printf("Error acquiring a frame.\n");
30
        idsCameraClose(&cam_param);
32      exit(0);
    }
34  IplImage* img = cvCreateImageHeader(cvSize(cam_param.img_size_x,cam_param.
        img_size_y), 8, 1);
    img->imageData = (unsigned char*) cam_param.img_buf;
36
    // Initialise and tune edge detection
38  edgedetect_param edg_param;
    edgedetect_init(&edg_param, NUM_SCAN_LINE, ROI_X, ROI_Y, ROI_WIDTH,
        ROI_HEIGHT, 0, ROI_Y_DEFAULT, 0, ROI_HEIGHT_DEFAULT);
40  edgedetect_manualSetThreshold(&edg_param, img);

42  // Initialise and tune triangulation
    mouse_param mparam;
44  mparam.flag=0;
    triangulation_param triang_param;
46  triangulation_RoiInit(&triang_param);
    triangulation_MouseInitialization(img,&cam_param,&mparam,&triang_param);
48
    // Initialize TCP/IP connection
50  int sock = -1;
    if ((sock=CreateSocketClientTCP(IMGPROCPC_IPADDR,IMGPROCPC_TCPPORT))
        ==-1)
52  {
        printf("CreateSocketClientTCP: Failed to create socket.\n");
54
        cvReleaseImage(&img);
56      idsCameraClose(&cam_param);
        exit(0);
58  }

60  // Create the output image and output window
    cvNamedWindow("Monitor", CV_GUI_EXPANDED);
62  IplImage *out_img = cvCreateImage(cvGetSize(img), IPL_DEPTH_8U, 3);

64  // Contour detection loop
    bool first_time = true;
66  double x_prev, y_prev, angle_prev, z_prev;

68  contour_lineparam lineparam;
    while ((char)cvWaitKey(4) != 27)
70  {
        // Acquire a frame
72      if (idsCameraAcquire(&cam_param)<0)
        {
74          printf("Error acquiring a frame.\n");

76          cvDestroyWindow("Monitor");
            cvReleaseImage(&img);
78          cvReleaseImage(&out_img);
            idsCameraClose(&cam_param);
80          exit(0);
        }
82
        // Execute edge detection
84      int n_edge = edgedetect_findEdge(&edg_param, img);
```

```
86          // Compute the line
            contour_lineInterp(&lineparam, edg_param.edgedetect_point, edg_param.
                n_edgedetect_line, cam_param.img_size_x, cam_param.img_size_y);
88
            // Laser triangulation
90          triangulation_BaseFilter(img,coord_x,&triang_param);

92          // Draw results
            edgedetect_drawEdgePoint(&edg_param, img, out_img);
94          contour_lineDraw(&lineparam, out_img);
            triangulation_drawMargin(img, out_img, &triang_param);
96          cvShowImage("Monitor",out_img);

98          // Check the image processing results before sending to the control PC
            if (first_time)
100         {
                first_time = false;
102
                x_prev = lineparam.x;
104             y_prev = lineparam.y;
                angle_prev = lineparam.angle;
106             z_prev = triang_param.z;
            }
108         else
            {
110             if (fabs(lineparam.x-x_prev)>VISSERV_MAXDELTA_X)
                {
112                 lineparam.x = x_prev;
                    printf("X out of bounds.\n");
114             }
                else
116                 x_prev = lineparam.x;

118             if (fabs(lineparam.y-y_prev)>VISSERV_MAXDELTA_Y)
                {
120                 lineparam.y = y_prev;
                    printf("Y out of bounds.\n");
122             }
                else
124                 y_prev = lineparam.y;

126             if (fabs(lineparam.angle-angle_prev)>VISSERV_MAXDELTA_THETA)
                {
128                 printf("%.7f ", fabs(lineparam.angle-angle_prev));
                    lineparam.angle = angle_prev;
130                 printf("Angle out of bounds.\n");
                }
132             else
                    angle_prev = lineparam.angle;
134
                if ((fabs(triang_param.z-z_prev)>VISSERV_MAXDELTA_Z) || (!triang_param.
                    line_found))
136             {
                    triang_param.z = z_prev;
138                 printf("Z out of bounds.\n");
                }
140             else
                    z_prev = triang_param.z;
142         }

144         // Send data to the control PC
            triangulation_send(sock, &lineparam, &triang_param);
```

```
146     }

148     // Close socket
        CloseSocketTCP(sock);
150
        // Release the window and the images
152     cvDestroyWindow("Monitor");
        cvReleaseImage(&img);
154     cvReleaseImage(&out_img);

156     // Close the camera
        idsCameraClose(&cam_param);
158     printf("Camera closed.\n");

160     return 0;
    }
```

B.1.2 Algoritmo *image_processing.c*

Contiene le funzioni utilizzate in (B.1.1) e sviluppate per poter realizzare il controllo visuale e gli obbiettivi richiesti di misurazione del profilo

```
1  #include <stdio.h>
   #include <stdbool.h>
3  #include <tpl.h>

5  #include "image_processing.h"
   #include "../utils/matfun.h"
7
   void mouseHandler(int event, int x, int y, int flags, void* param);
9
   ////////////////////////////////////////////////////////////////////////
11 // Internal functions for GUI management                              //
   ////////////////////////////////////////////////////////////////////////
13
   // Global variables
15 int lowSliderPosition = 9;
   int highSliderPosition = 34;
17 int maxHighThreshold = 500;
   int maxLowThreshold = 500;
19
   int TriangLowSliderPosition = 100;
21 int TriangHighSliderPosition = 34;

23 // Callback function to adjust the low threshold on slider movement
   void onLowThresholdSlide(int theSliderValue)
25 {
       lowSliderPosition = theSliderValue;
27 }

29 // Callback function to adjust the high threshold on slider movement
   void onHighThresholdSlide(int theSliderValue)
31 {
       highSliderPosition = theSliderValue;
33 }

35 ////////////////////////////////////////////////////////////////////////

37 // Callback function to adjust the low threshold on slider movement (Laser thresholding)
   void onTriangLowThresholdSlide(int theSliderValue)
```

```
39 {
       TriangLowSliderPosition = theSliderValue;
41 }

43 // Callback function to adjust the high threshold on slider movement (Laser thresholding)
   void onTriangHighThresholdSlide(int theSliderValue)
45 {
       TriangHighSliderPosition = theSliderValue;
47 }
   ////////////////////////////////////////////////////////////////////////////
49

51 void edgedetect_init(edgedetect_param* param, int num_line, int x, int y, int width, int
       height, int x_default, int y_default, int width_default, int height_default)
   {
53     // Initialise the ROI position and dimension
       int i;
55     for (i=0;i<num_line;i++)
       {
57         param->edgedetect_roi[i].x = x + (i*width);
           param->edgedetect_roi[i].x_default = x_default;
59         param->edgedetect_roi[i].y = y;
           param->edgedetect_roi[i].y_default = y_default;
61         param->edgedetect_roi[i].width = width;
           param->edgedetect_roi[i].width_default = width_default;
63         param->edgedetect_roi[i].height = height;
           param->edgedetect_roi[i].height_default = height_default;
65         param->edgedetect_roi[i].flag = 0;
       }
67
       param->n_edgedetect_line = num_line;
69 }

71 void edgedetect_manualSetThreshold(edgedetect_param* param, IplImage* image)
   {
73     // Create monitor window
       cvNamedWindow("Set_image_threshold", CV_GUI_EXPANDED);
75
       // Clone the input image before applying the threasholding
77     IplImage* imgThd = cvCreateImage(cvGetSize(image), IPL_DEPTH_8U, 1);

79     // Create the threshold slider
       cvCreateTrackbar("Low_Threshold","Set_image_threshold", &lowSliderPosition,
           maxLowThreshold, onLowThresholdSlide);
81     cvCreateTrackbar("High_Threshold", "Set_image_threshold", &highSliderPosition,
           maxHighThreshold, onHighThresholdSlide);

83     while ((char)cvWaitKey(2) != 27) // Exit if user press ESC
       {
85         // Apply the thresholding
           cvThreshold(image,imgThd, lowSliderPosition, highSliderPosition,
               CV_THRESH_BINARY);
87         cvCanny(imgThd, imgThd, 80, 200, 3);

89         // Show the image
           cvShowImage("Set_image_threshold",imgThd);
91     }

93     // Release the image and destroy the window
       cvReleaseImage(&imgThd);
95     cvDestroyWindow("Set_image_threshold");
```

```
97      // Update values
        param->low_th = lowSliderPosition;
99      param->high_th = highSliderPosition;
    }
101
    int edgedetect_findEdge(edgedetect_param* param, IplImage* image)
103 {
        int sum = 0;
105
        // Create a copy of the image
107     CvSize image_size = cvGetSize(image);
        IplImage *img_cpy = cvCreateImage(image_size, IPL_DEPTH_8U, 1);
109     img_cpy = cvCloneImage(image);

111     int n;
        for (n=0;n<param->n_edgedetect_line;n++)
113     {

115         // Set ROI
            cvSetImageROI(img_cpy,cvRect(param->edgedetect_roi[n].x,
117                                          param->edgedetect_roi[n].y,
                                             param->edgedetect_roi[n].width,
119                                          param->edgedetect_roi[n].height));

121         // Apply edge detection to ROI
            cvThreshold(img_cpy, img_cpy, param->low_th, param->high_th,
                  CV_THRESH_BINARY);
123         cvCanny(img_cpy, img_cpy, 80, 200, 3);

125         // Find a white pixel in the centre of the ROI
            int i;
127
            param->edgedetect_roi[n].flag = 0;
129         for (i=param->edgedetect_roi[n].y; i<(param->edgedetect_roi[n].y + param->
                 edgedetect_roi[n].height); i++)
            {
131             int col = (int)(((uchar *) (img_cpy->imageData + img_cpy->widthStep*(i) )) [
                      param->edgedetect_roi[n].x+param->edgedetect_roi[n].width/2]);
                if (col==255)
133             {
                    param->edgedetect_point[param->n_edgedetect_line-n-1] = cvPoint(
                          param->edgedetect_roi[n].x + (param->edgedetect_roi[n].width/2),i);
                          //from left to right
135                 param->edgedetect_roi[n].flag = 1;

137                 break;
                }
139         }

141         // Reset ROI
            cvResetImageROI(img_cpy);
143         sum = sum + param->edgedetect_roi[n].flag;
        }
145
        // Set the new position/dimension of the ROI
147     int i;
        for (i=0;i<param->n_edgedetect_line;i++)
149     {
            // If an edge has been found
151         // If ROI_height > 100p reduce the ROI_height and center it on the edge
            if (param->edgedetect_roi[i].flag == 1)
153         {
```

```
            if (param->edgedetect_roi[i].height>100)
155             param->edgedetect_roi[i].height=param->edgedetect_roi[i].height - 100;

157         param->edgedetect_roi[i].y=param->edgedetect_point[param->
                n_edgedetect_line-1-i].y - param->edgedetect_roi[i].height/2;
            if (param->edgedetect_roi[i].y<0)
159             param->edgedetect_roi[i].y = 0;
            if ((param->edgedetect_roi[i].y+param->edgedetect_roi[i].height)>image_size.
                height)
161             param->edgedetect_roi[i].y = image_size.height-param->edgedetect_roi[i].
                    height;
        }
163     // Otherwise set the default ROI parameters
        else
165     {
            param->edgedetect_roi[i].y = param->edgedetect_roi[i].y_default;
167         param->edgedetect_roi[i].height = param->edgedetect_roi[i].height_default;
        }
169 }

171 // Release the copy of the image
    cvReleaseImage(&img_cpy);
173
    return sum;
175 }

177 void edgedetect_drawEdgePoint(edgedetect_param* param, IplImage* src_image, IplImage*
      dest_image)
    {
179 // Create a copy of the image in dest_image
    cvCvtColor(src_image,dest_image,CV_GRAY2BGR);
181
    // Draw edge points
183 int n;
    for (n=0;n<param->n_edgedetect_line;n++)
185 {
        // Draw the scan line
187     cvLine(dest_image,cvPoint(param->edgedetect_roi[n].x+param->edgedetect_roi[n].
            width/2,param->edgedetect_roi[n].y),
              cvPoint(param->edgedetect_roi[n].x+param->edgedetect_roi[n].width/2,
                  param->edgedetect_roi[n].y+param->edgedetect_roi[n].height),
189           CV_RGB(0,255,0),2,8,0);

191     // Draw a point on the detected edge
        cvCircle(dest_image,param->edgedetect_point[n],4,CV_RGB(255,0,0),2,8,0);
193
        // Draw the segment between two points
195     if (n<param->n_edgedetect_line-1)
            cvLine(dest_image,param->edgedetect_point[n],param->edgedetect_point[n+1],
                CV_RGB(255,0,0),2,8,0);
197 }
    }
199
    void edgedetect_drawEdgePoint_2(edgedetect_param* param, contour_lineparam* lineparam,
      IplImage* src_image, IplImage* dest_image)
201 {
    int n;
203 int s1 = 250;
    int s2 = -250;
205
    cvCvtColor (src_image,dest_image,CV_GRAY2BGR);
207 CvPoint COI = cvPoint( 640, 512);
```

```
        CvPoint T = cvPoint( 640+lineparam->x,512+lineparam->y);
209     CvPoint I = cvPoint( 640+lineparam->x + s1, 512+lineparam->y + lineparam->c1*s1);
        CvPoint F = cvPoint( 640+lineparam->x + s2, 512+lineparam->y + lineparam->c1*s2)
            ;
211
        //--points
213     cvCircle(dest_image,COI,4,CV_RGB(250,0,0),2,8,0);
        cvCircle(dest_image,T,4,CV_RGB(250,0,0),2,8,0);
215     //--lines
        cvLine(dest_image,COI,T,CV_RGB(0,0,250),2,8,0);
217     cvLine(dest_image,I,F,CV_RGB(0,250,0),2,8,0);

219     //--contour points
        for (n=0;n<9;n++)
221     {
            cvCircle(dest_image,param->edgedetect_point[n],4,CV_RGB(0,250,0),2,8,0);
223     }
        CvFont font;
225     double hScale = 1.0;
        double vScale = 1.0;
227     int lineWidth = 4;
        cvInitFont( &font, CV_FONT_HERSHEY_SIMPLEX | CV_FONT_ITALIC,
229                 hScale, vScale, 0, 1,lineWidth );
        char text_x[100];
231     char text_th[100];
        char text_y[100];
233     sprintf(text_x,"x_T:␣%.3f",lineparam->x);
        sprintf(text_y,"y_T:␣%.3f",-lineparam->y);
235     sprintf(text_th,"theta:␣%.3f",lineparam->angle);
        cvPutText( dest_image, text_x, cvPoint( 550, 800 ), &font,
237                 cvScalar( 255, 255, 255 ,0) );
        cvPutText( dest_image, text_y, cvPoint( 550, 850 ), &font,
239                 cvScalar( 255, 255, 255 ,0) );
        cvPutText( dest_image, text_th, cvPoint( 550, 900 ), &font,
241                 cvScalar( 255, 255, 255 ,0) );
    }
243

245 void contour_lineInterp(contour_lineparam* lineparam, CvPoint point[], int n_point, int
        img_width, int img_height)
    {
247     double Xp[n_point],Yp[n_point];

249     int i;
        for (i=0;i<n_point;i++)
251     {
            Xp[i]=(double)point[i].x - ((double)img_width) / 2.0; // (Xp,Yp)=(0,0) if the point
                is in center of the image
253         Yp[i]=(double)point[i].y - ((double)img_height) / 2.0;
        }
255
        // Find the least square line that interpolates the points
257     double line_param[3];
        lineFit_RANSAC(Xp, Yp, n_point, 22, 5.0, line_param);
259     lineparam->x = -line_param[0]*line_param[2]/(line_param[0]*line_param[0]+
            line_param[1]*line_param[1]);
        lineparam->y = -line_param[1]*line_param[2]/(line_param[0]*line_param[0]+
            line_param[1]*line_param[1]);
261
        // Old parameters (for drawing function only)
263     if (fabs(line_param[1])<1e-14)
        {
```

```
265         lineparam->c0 = NAN; // Nan is better!! but this parameter is unuseful
            lineparam->c1 = NAN; //slope infinite!!
267         lineparam->angle = M_PI/2.0;
        }
269     else
        {
271         lineparam->c0 = -line_param[3]/line_param[1];
            lineparam->c1 = -line_param[0]/line_param[1];
273         lineparam->angle = atan(lineparam->c1);
        }
275

277 }

279

281 void contour_lineDraw(contour_lineparam* lineparam, IplImage* color_image)
    {
283     CvSize image_size = cvGetSize(color_image);

285     // Image center
        CvPoint coi = cvPoint(image_size.width/2, image_size.height/2);
287     cvCircle(color_image,coi,4,CV_RGB(0,0,255),2,8,0);

289     // Point to draw the perpendicular to the line
        CvPoint T = cvPoint(image_size.width/2 + (int)round(lineparam->x), image_size.height
            /2 + (int)round(lineparam->y));
291     cvCircle(color_image,T,4,CV_RGB(0,0,255),2,8,0);

293     // First point to draw the line

295     if(lineparam->c1!=NAN)CvPoint I = cvPoint(image_size.width/2 + (int)round(
            lineparam->x) + 200,
                            image_size.height/2 + (int)round(lineparam->y + lineparam->c1
                                *200));
297     else CvPoint I = cvPoint(image_size.width/2 + (int)round(lineparam->x),
                            image_size.height/2 + (int)round(lineparam->y + 200));
299     cvCircle(color_image,I,4,CV_RGB(0,0,255),2,8,0);

301     // Last point to draw the line
        if(lineparam->c1!=NAN) CvPoint F = cvPoint(image_size.width/2 + (int)round(
            lineparam->x) - 200, image_size.height/2 + (int)round(lineparam->y - lineparam
            ->c1*200));
303     else CvPoint F = cvPoint(image_size.width/2 + (int)round(lineparam->x),
                            image_size.height/2 + (int)round(lineparam->y - 200));
305     cvCircle(color_image,F,4,CV_RGB(0,0,255),2,8,0);

307     cvLine(color_image,coi,T,CV_RGB(0,0,255),2,8,0);
        cvLine(color_image,I,F,CV_RGB(0,0,255),2,8,0);
309 }

311
    void triangulation_BaseFilter(IplImage* src_img, triangulation_param* triang_param)
313 {
        int yy,xx,tmp,cont,maxval,posl4 ;
315     int flag_highdistance=0,flag_lowbrightness=0,flag_hole=0, flag_start=0;
        float max_buff[triang_param->roi.width],distance;
317     float coord_x[triang_param->roi.height];

319     posl4=triang_param->x_start;

321     triang_param->roi.y=triang_param->y_start;
```

```
323     IplImage *flt_img = cvCreateImage(cvGetSize(src_img), IPL_DEPTH_8U , 1);

325     cvNamedWindow("windowed", CV_GUI_EXPANDED);

327     triang_param->line_found=true;
        for (yy=triang_param->roi.y; yy<triang_param->roi.y+triang_param->roi.height; yy
            ++)
329     {

331         cont=0;
            maxval=0;
333
            for (xx=triang_param->roi.x; xx<triang_param->roi.x+triang_param->roi.width;
                xx++)
335         {
                if (abs(xx-posl4)>triang_param->win_size)
337             {
                    if ((uchar)((src_img->imageData + yy*src_img->widthStep)[xx] ) > (uchar)
                        (triang_param->decr_bright))
339                     ((flt_img->imageData + yy*flt_img->widthStep))[xx] = (uchar)(((
                            src_img->imageData + yy*src_img->widthStep)[xx]) -
                            triang_param->decr_bright);
                    else
341                     ((flt_img->imageData + yy*flt_img->widthStep))[xx] = (uchar)(0);
                }
343             else
                {
345                 if ((uchar)((src_img->imageData + yy*src_img->widthStep)[xx] ) < (uchar)
                        (255-triang_param->incr_bright))
                        ((flt_img->imageData + yy*flt_img->widthStep))[xx] = (uchar)(((
                            src_img->imageData + yy*src_img->widthStep)[xx]) +
                            triang_param->incr_bright);
347                 else
                        ((flt_img->imageData + yy*flt_img->widthStep))[xx] = (uchar)(255);
349             }

351
                if ((tmp =(int) (((uchar *)(flt_img->imageData + yy*flt_img->widthStep))[xx])
                    )>maxval)
353                 maxval=tmp;
            } //end windowing & max
355
            for (xx=triang_param->roi.x;xx<triang_param->roi.x+triang_param->roi.width;
                xx++)
357         {
                if (maxval==(int) (((uchar *)(flt_img->imageData + yy*flt_img->widthStep))[
                    xx]))
359             {
                    max_buff[cont] = xx;
361                 cont++;
                }
363         }

365         if ((cont%2)==0)
                coord_x[yy-triang_param->roi.y] = (max_buff[(cont/2)-1]+max_buff[cont/2])
                    /2;
367         else
                coord_x[yy-triang_param->roi.y] = max_buff[cont/2];
369
            posl4=floor(coord_x[yy-triang_param->roi.y]);
371
```

```
        if (((uchar)(flt_img->imageData + yy*flt_img->widthStep)[posl4]>=(uchar)(
              triang_param->gray_line_found)) && (flag_start==0))
373     {
            flag_start=1;
375         triang_param->x_start=posl4;
        }
377
        if ((yy>0) && (flag_start))
379     {
            flag_highdistance=(fabs(coord_x[yy-triang_param->roi.y]-coord_x[yy-1-
                  triang_param->roi.y])> triang_param->thr_high_dist);
381         flag_lowbrightness=((uchar)((flt_img->imageData + yy*flt_img->widthStep)[
                  posl4] ) < (uchar)(triang_param->thr_low_bright));

383         if (flag_hole)
            {
385             if (flag_lowbrightness)
                {
387                 triang_param->x=coord_x[yy-2-triang_param->roi.y];
                    triang_param->y=yy-2;
389                 triang_param->z=COEFF_K1/(COEFF_K2-triang_param->x);
                    if((triang_param->y_start=triang_param->y-triang_param->roi.
                          height/2)<0)triang_param->y_start=0;
391                 else{
                        if(triang_param->y_start+triang_param->roi.height>
                              CAMERA_WRIST_FRAME_SIZEY)triang_param->y_start
                              =CAMERA_WRIST_FRAME_SIZEY-triang_param->roi.
                              height;
393                 }

395                 cvShowImage("windowed", flt_img);
                                cvWaitKey(4);
397                             cvReleaseImage(&flt_img);

399                 return;
                }
401             else
                {
403                 if (flag_highdistance)
                    {
405                     if (posl4-max_buff[0]>triang_param->line_dim/2)
                        {
407                         triang_param->x=coord_x[yy-2-triang_param->roi.y];
                            triang_param->y=yy-2;
409                         triang_param->z=COEFF_K1/(COEFF_K2-triang_param->
                                  x);
                            if((triang_param->y_start=triang_param->y-triang_param
                                  ->roi.height/2)<0)triang_param->y_start=0;
411                         else{
                                if(triang_param->y_start+triang_param->roi.height>
                                      CAMERA_WRIST_FRAME_SIZEY)triang_param->
                                      y_start=CAMERA_WRIST_FRAME_SIZEY-
                                      triang_param->roi.height;
413                         }

415                         cvShowImage("windowed", flt_img);
                                                cvWaitKey(4);
417                                             cvReleaseImage(&flt_img);

419                         return;
                        }
421                     else flag_hole=0;
```

```
                    }
423                 else flag_hole=0;
                }
425         }
            else
427         {
                if ((flag_highdistance)&&(flag_lowbrightness))
429             {
                    triang_param->x=coord_x[yy-1-triang_param->roi.y];
431                 triang_param->y=yy-1;
                    triang_param->z=COEFF_K1/(COEFF_K2-triang_param->x);
433                 if((triang_param->y_start=triang_param->y-triang_param->roi.
                        height/2)<0)triang_param->y_start=0;
                    else{
435                     if(triang_param->y_start+triang_param->roi.height>
                            CAMERA_WRIST_FRAME_SIZEY)triang_param->y_start
                            =CAMERA_WRIST_FRAME_SIZEY-triang_param->roi.
                            height;
                    }
437
                    cvShowImage("windowed", flt_img);
439                             cvWaitKey(4);
                            cvReleaseImage(&flt_img);
441
                    return;
443             }
                else
445             {
                    if (flag_highdistance)
447                 {
                        if (posl4-max_buff[0]>triang_param->line_dim/2)
449                     {
                            triang_param->x=coord_x[yy-1-triang_param->roi.y];
451                         triang_param->y=yy-1;
                            triang_param->z=COEFF_K1/(COEFF_K2-triang_param->
                                x);
453                         if((triang_param->y_start=triang_param->y-triang_param
                                ->roi.height/2)<0)triang_param->y_start=0;
                            else{
455                             if(triang_param->y_start+triang_param->roi.height>
                                    CAMERA_WRIST_FRAME_SIZEY)triang_param->
                                    y_start=CAMERA_WRIST_FRAME_SIZEY-
                                    triang_param->roi.height;
                            }
457
                            cvShowImage("windowed", flt_img);
459                                         cvWaitKey(4);
                                        cvReleaseImage(&flt_img);
461
                            return;
463                     }
                    }
465                 else
                    {
467                     flag_hole=1;
                    }
469             }
            }
471     }
    }
473 triang_param->x=(double)NAN;
    triang_param->y=(double)NAN;
```

```
475     triang_param->z=(double)NAN;
        triang_param->line_found=false;
477
        cvShowImage("windowed", flt_img);
479     cvWaitKey(4);
        cvReleaseImage(&flt_img);
481
    }
483
    void mouseHandler(int event, int x, int y, int flags, void* param)
485 {
        mouse_param* mparam=(mouse_param*)param;
487
        switch (event)
489     {
            /* left button down */
491     case CV_EVENT_LBUTTONDOWN:
            mparam->x=x;
493         mparam->y=y;

495         printf("x: %d , y: %d\n", mparam->x, mparam->y);
            mparam->flag=1;
497         break;

499         /* mouse move */
        case CV_EVENT_MOUSEMOVE:
501         /* draw a rectangle */
            mparam->x=x;
503         mparam->y=y;
            break;
505     }
    }
507
    void triangulation_MouseInitialization(IplImage* img_in,camera_param *cam_param,
          mouse_param* mparam, triangulation_param* triang_param,)
509 {
        IplImage* img_col;
511     img_col = cvCreateImage(cvSize(CAMERA_WRIST_FRAME_SIZEX,
              CAMERA_WRIST_FRAME_SIZEY),8,3);
        char c;
513     cvNamedWindow("Click the line", CV_GUI_EXPANDED);
        ResizeWindow("Click the line", 800, 800*4/5);
515     cvSetMouseCallback("Click the line", mouseHandler, mparam);

517     //Callback function to adjust the parameters of triangulation_BaseFilter
        void on_thr_low_bright(int theSliderValue)
519     {
            triang_param->thr_low_bright = theSliderValue;
521     }
        void on_incr_bright(int theSliderValue)
523     {
            triang_param->incr_bright = theSliderValue;
525     }
        void on_decr_bright(int theSliderValue)
527     {
            triang_param->decr_bright = theSliderValue;
529     }
        void on_line_dim(int theSliderValue)
531     {
            triang_param->line_dim = theSliderValue;
533     }
        void on_gray_line_found(int theSliderValue)
```

```
535     {
            triang_param->gray_line_found = theSliderValue;
537     }
        void on_thr_high_dist(int theSliderValue)
539     {
            triang_param->thr_high_dist = theSliderValue;
541     }

543     while(c!=27)
        {
545       if(c==127)mparam->flag=0;

547     while (!mparam->flag)
        {
549         if (idsCameraAcquire(cam_param)<0)
            {
551             printf("Error acquiring a frame in triangulation MouseInitialization.\n");

553             cvDestroyWindow("Click the line");
                cvReleaseImage(&img_col);
555             idsCameraClose(cam_param);
                exit(0);
557         }
            cvCvtColor(img_in,img_col,CV_GRAY2BGR);
559         cvRectangle(img_col,cvPoint(triang_param->roi.x, mparam->y-triang_param->
                roi.height/2),
                        cvPoint(triang_param->roi.x + triang_param->roi.width, mparam->y
                            +triang_param->roi.height/2), cvScalar(204,204,255, 0), 2, 8, 0);
561         cvRectangle(img_col,cvPoint(mparam->x - triang_param->line_dim/2, mparam->
                y-triang_param->roi.height/2),
                        cvPoint(mparam->x + triang_param->line_dim/2, mparam->y-
                            triang_param->roi.height/2 + triang_param->line_dim/2*8),
                            cvScalar(0, 255, 0, 0), 2, 8, 0);
563         cvShowImage("Click the line", img_col);
            cvWaitKey(2);
565     }

567     triang_param->x_start=mparam->x;
        triang_param->y_start=mparam->y-triang_param->roi.height/2;
569
        cvCreateTrackbar("Threshold Low Brightness","Click the line", triang_param->
            thr_low_bright, 255,on_thr_low_bright);
571     cvCreateTrackbar("Increment Brightness","Click the line", triang_param->incr_bright,
            255,on_incr_bright);
        cvCreateTrackbar("Decrement Brightness","Click the line",triang_param->decr_bright,
            255,on_decr_bright);
573     cvCreateTrackbar("Line Dimension","Click the line",triang_param->line_dim, 20,
            on_line_dim);
        cvCreateTrackbar("Threshold Grayvalue Found","Click the line",triang_param->
            gray_line_found, 255,on_gray_line_found);
575     cvCreateTrackbar("Threshold High Distance","Click the line",triang_param->
            thr_high_dist, 200,on_thr_high_dist);
        while (c != 27 && c!= 127) // Exit if user press ESC , come back if user press DEL
577     {
            if (idsCameraAcquire(cam_param)<0)
579         {
                printf("Error acquiring a frame in triangulation MouseInitialization.\n");
581
                cvDestroyWindow("Click the line");
583             cvReleaseImage(&img_col);
                idsCameraClose(cam_param);
585             exit(0);
```

```
        }
587     cvCvtColor(img_in,img_col,CV_GRAY2BGR);
        triangulation_BaseFilter(img_in, triang_param);
589     triangulation_drawMargin(img_in, img_col, triang_param);
        cvShowImage("Click the line", img_col);
591     c = (char)cvWaitKey(2);
    }
593 }
    triang_param->win_size=triang_param->line_dim/2+1;
595 cvReleaseImage(&img_col);
    cvDestroyWindow("Click the line");
597 }

599 void triangulation_drawMargin(IplImage* src_image, IplImage* dest_image,
      triangulation_param *triang_param)
    {
601
    if (triang_param->x!=NAN) cvRectangle(dest_image,cvPoint(triang_param->x -
        triang_param->line_dim*3, triang_param->y - triang_param->line_dim*3),
603             cvPoint(triang_param->x + triang_param->line_dim*3, triang_param
                ->y + triang_param->line_dim*3), cvScalar(255,205,42, 0), 2, 8, 0)
                ;
    }
605
    void triangulation_RoiInit(triangulation_param* triang_param)
607 {
    triang_param->roi.x=LASER_ROI_X;
609 triang_param->roi.y=LASER_ROI_Y;
    triang_param->roi.width=LASER_ROI_WIDTH;
611 triang_param->roi.height=LASER_ROI_HEIGHT;
    triang_param->x=TRIANG_INIT_VALUE;
613 triang_param->y=TRIANG_INIT_VALUE;
    triang_param->z=TRIANG_INIT_VALUE;
615 triang_param->x_start=-1;
    triang_param->y_start=-1;
617 triang_param->thr_low_bright=THR_LOWBRIGHTNESS;
    triang_param->incr_bright=INCREMENT_BRIGHTNESS;
619 triang_param->decr_bright=DECREMENT_BRIGHTNESS;
    triang_param->line_dim=LINE_DIMENSION;
621 triang_param->gray_line_found=GRAYVALUE_LINE_FOUND;
    triang_param->thr_high_dist=THRESHOLD_HIGH_DISTANCE;
623 triang_param->win_size=WINDOWING_SIZE;
    }
625
    int triangulation_send(int sock,contour_lineparam *cont_param,triangulation_param *
      triang_param)
627 {
    double s_values[6];
629 s_values[0]=cont_param->x;
    s_values[1]=cont_param->y;
631 s_values[2]=cont_param->angle;
    s_values[3]=triang_param->z;
633 s_values[4]=triang_param->z/CAMERA_WRIST_FX*triang_param->x;
    s_values[5]=triang_param->z/CAMERA_WRIST_FY*triang_param->y;
635
    if (!triang_param->line_found)
637 {
        s_values[4]=(double)NAN;
639     s_values[5]=(double)NAN;
    }
641
    tpl_node *tn = NULL;
```

```
643     if (!(tn = tpl_map( "f#", s_values, 6)))
        {
645         printf("NULL tpl_map pointer.\n");

647         return -1;
        }
649     if (tpl_pack(tn,0)<0)
        {
651         printf("Error packing tpl struct.\n");

653         return -1;
        }
655
        void *buf_addr;
657     size_t buf_len;
        if (tpl_dump(tn, TPL_MEM, &buf_addr, &buf_len)<0)
659     {
            printf("Error dumping tpl struct.\n");
661
            return -1;
663     }
        tpl_free(tn);
665
        int byte_sent;
667     if ((byte_sent=send(sock, buf_addr, buf_len, 0)) != buf_len)
        {
669         printf("Mismatch in number of sent bytes (sent %d bytes instead of %d).\n",
                byte_sent, (int)buf_len);

671         return -1;
        }
673
        return 0;
675 }
```

B.2 Linguaggio MATLAB

Il seguente algoritmo viene utilizzato durante la calibrazione del sistema a triangolazione, per identificare i parametri K_1 e K_2

```
1   close all;
    clear all;
3   clc;

5   % questo algoritmo calcola la cifra di costo J(K1, K2) ed
    % inserisce in un grafico il suo valore al variare dei parametri incogniti
7
    % start= distanza del piano oggetto dalla terna di camera
9
    % i valori nominali dei parametri sono:
11  % f = lunghezza focale
    % d = distanza del punto di incontro del fascio luminoso
13  % e del piano dove giacciono l'asse x e y della terna di camera
    % con il centro stesso della terna
15  % alpha = angolo di rotazione del fascio luminoso (in gradi), positivo in senso orario
    % rispetto alla normale del piano dove giacciono l'asse x e y della
17  % terna di camera
    % Sx = parametro che indica la dimensione del singolo pixel in orizzotale,
19  % ( larghezza sensore / tot. num pixel )
```

```
% Cx = misura del centro ottico su piano del sensore espressa in pixel
%
% n = misura in pixel del punto dove si rileva il fascio luminoso


totmisure= 100;
nm = 5;
dimx = 1280;
dimy = 1024;
f = 25;
Sx = 6.784/dimx;
Cx = dimx/2;
alpha = 48.5;
d = 220;



% cifra di merito:
% J1 = sommatoria degli scarti quadratici medi
% delle distanze misurate (z) da quelle effettive (y)
J1=0;
% formula utilizzata per il calcolo della distanza fra terna camera
% e punto dell'oggetto illuminato
%
% f * d
% z = --------------------------
% f * tan ( alpha ) + P
%
% con P = -[( n - Cx ) * Sx ]
%
%
% K1
% z = --------------------------
% K2 - n
%
%con K1 = f*d/Sx ; K2 = f*tan(alpha)/Sx+Cx
%
%


for jjj = 1:totmisure
st = sprintf('C:/misure%d.txt',jjj);
npixel(:,jjj) = importdata(st);
end

for i = 1:nm
    media(i) = mean(npixel(i,:));
    devstandard(i) = std(npixel(i,:));
    varianza(i) = var(npixel(i,:));
end



% --------------------------test
    --------------------------------

n= media;



for ciclo=0:100
    close all;
```

```
start = 175.+0.2*ciclo; %per verificare anche la possibilita' di un offset sulla posizione
    iniziale si varia in un intorno
% y = il vettore dei termini noti: una serie di 5 valori di distanza
% crescenti, partendo dal valore start e mantenendo una distanza
% di 1 cm da uno all'altro
y = [start:10:start+40]';

for i=1:nm
    z(i) = (f*d)/(f*tan(alpha/180*pi)-(( n(i) - Cx ) * Sx));
end
zprima=z';
K1 = f*d/Sx;
K2 = f*tan(alpha/180*pi)/Sx+Cx;
p=0.4;
K1_vet = [K1*(1-p):(K1*(1+p)-K1*(1-p))/200:K1*(1+p)]';
K2_vet = [K2*(1-p):(K2*(1+p)-K2*(1-p))/200:K2*(1+p)]';
J1_mat = zeros(length(K1_vet),length(K2_vet));
J1=0;

% calcola i valori della cifra di costo per le serie di misure effettuate
% e per le variazioni di parametri volute

for ii=1:length(K1_vet)
    K1=K1_vet(ii);
    for jj=1:length(K2_vet)
        K2=K2_vet(jj);

            % ------------------------test
                ----------------------

            for i=1:nm
                z(i) = K1/(K2-n(i));
            end

            for i=1:nm
                J1 = J1+ (y(i)-z(i))^2;
            end
            J1_mat(ii,jj) = J1;
            J1=0;
    end
end
%normalizzo la funzione di costo
A=reshape(J1_mat,1,length(K1_vet)*length(K2_vet));
minJ1 = min(A);
maxJ1 = max(A);
J1_mat_norm=J1_mat./(maxJ1-minJ1);

figure();
surf(K1_vet,K2_vet,J1_mat_norm(:,:));
xlabel(['K1 [pixel mm]'])
ylabel(['K2 [pixel]'])
title('funzione di costo J normalizzata, scarti dal valore nominale')

m=1;
for ii=1:length(K1_vet)
    for jj=1:length(K2_vet)

        if J1_mat(ii,jj)== minJ1
            vet_minJ1(m,:)= [ii,jj,J1_mat(ii,jj)];
            m=m+1;
        end

```

```
        end
    end
    disp( 'i parametri che minimizzano J1 sono:')
    K1=K1_vet(vet_minJ1(1,1))
    K2=K2_vet(vet_minJ1(1,2))

    %
        ----------------------------------------------------------------------


    for i=1:nm
                    z(i) = K1/(K2-n(i));
    end

    zminJ1 = z'
    diff_valori_nominali = zminJ1-y;
    somma_diff_resid(ciclo+1)=abs(sum(diff_valori_nominali))
end
figure();
plot(somma_diff_resid)
```

Bibliografia

[1] Augusto Ajovalasit. «Analisi sperimentale delle tensioni con la fotomeccanica». In: Aracne. 2006.

[2] Jan C Aurich et al. «Burrs—analysis, control and removal». In: *CIRP Annals-Manufacturing Technology* 58.2 (2009), pp. 519–542.

[3] Paul J Besl e Neil D McKay. «Method for registration of 3-D shapes». In: *Robotics-DL tentative*. International Society for Optics e Photonics. 1992, pp. 586–606.

[4] F. Blais. «Review of 20 years of range sensor development». In: *Journal of Electionic Imaging* 13.1 (2004), pp. 231–243.

[5] J.-Y. Bouguet. *Camera Calibration Toolbox for Matlab*. `http://www.vision.caltech.edu/bouguetj/calib_doc`.

[6] S Brinkmann. «Development of a robust Gaussian regression filter for three-dimentional surface analysis». In: *10. International Colloquim on Surfaces 31.1./1.2. 2000, Chemenitz, Germany* (2000).

[7] Andrés Castaño e Seth Hutchinson. «Visual compliance: Task-directed visual servo control». In: *IEEE transactions on Robotics and Automation* 10.3 (1994), pp. 334–342.

[8] F. Chaumette e S. Hutchinson. «Visual servo control part I: basic approaches». In: *IEEE Robotics and Automation Magazine* 13.4 (2006), pp. 82–90.

[9] Barbara D'Amico. *Automazione e lavori a rischio*. `http://nuvola.corriere.it/2017/01/22/automazione-e-lavori-a-rischio/`.

[10] Stefan Fuchs e Gerd Hirzinger. «Extrinsic and depth calibration of ToF-cameras». In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–6.

[11] Damien Garcia. «Robust smoothing of gridded data in one and higher dimensions with missing values». In: *Computational statistics & data analysis* 54.4 (2010), pp. 1167–1178.

[12] Valentina Alena Girelli. «Tecniche digitali per il rilievo, la modellazione tridimensionale e la rappresentazione nel campo dei beni culturali». Tesi di dott. alma, 2007.

[13] Armin Gruen e Devrim Akca. «Least squares 3D surface and curve matching». In: *ISPRS Journal of Photogrammetry and Remote Sensing* 59.3 (2005), pp. 151–174.

[14] Jacques Harvent et al. «Multi-view dense 3D modelling of untextured objects from a moving projector-cameras system». In: *Machine vision and applications* 24.8 (2013), pp. 1645–1659.

[15] Janne Heikkila e Olli Silven. «A four-step camera calibration procedure with implicit image correction». In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE. 1997, pp. 1106–1112.

[16] MG Her e H Kazerooni. «Automated robotic deburring of parts using compliance control». In: *Journal of dynamic systems, measurement, and control* 113.1 (1991), pp. 60–66.

[17] Feng-Yi Hsu e Li-Chen Fu. «Intelligent robot deburring using adaptive fuzzy hybrid position/force control». In: *Robotics and Automation, IEEE Transactions on* 16.4 (2000), pp. 325–335.

[18] Jwu-Sheng Hu e Yung-Jung Chang. «Automatic calibration of hand–eye–workspace and camera using hand-mounted line laser». In: *IEEE/ASME Transactions on Mechatronics* 18.6 (2013), pp. 1778–1786.

[19] Kazuo Kiguchi e Toshio Fukuda. «Position/force control of robot manipulators for geometrically unknown objects using fuzzy neural networks». In: *Industrial Electronics, IEEE Transactions on* 47.3 (2000), pp. 641–649.

[20] S L Ko e S W Park. «Development of an Effective Measurement System for Burr Geometry». In: *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 220.4 (2006), pp. 507–512.

[21] S Lang et al. «Characterization and testing of the BIRIS range sensor». In: *Instrumentation and Measurement Technology Conference, 1993. IMTC/93. Conference Record., IEEE*. IEEE. 1993, pp. 459–464.

[22] Amir Ghalamzan Gianantonio Magnani Matteo Pirotta Marcello Restelli Paolo Rocco Luca Bascetta Gianni Ferretti. «Learning a human skill». In: *Deliverable N° 2 - FP7 Grant Agreement Number: 231143 - Experiment: FIDELIO*. ECHORD. 2012.

[23] Stefan May et al. «3D pose estimation and mapping with time-of-flight cameras». In: *International Conference on Intelligent Robots and Systems (IROS), 3D Mapping workshop, Nice, France*. 2008.

[24] *OpenCV – Open Source Computer Vision*. `http://opencv.willowgarage.com`.

[25] Nikolaos P Papanikolopoulos, Pradeep K Khosla e Takeo Kanade. «Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision». In: *IEEE transactions on robotics and automation* 9.1 (1993), pp. 14–35.

[26] Fabio Remondino e Li Zhang. «Surface reconstruction algorithms for detailed close-range object modeling». In: *Proceedings of ISPRS Commission III Symposium*. 2006, pp. 117–123.

[27] CMPPC Rocchini et al. «A low cost 3 D scanner based on structured light». In: *Computer Graphics Forum*. Vol. 20. 3. 2001, pp. 299–308.

[28] Michele Russo, Fabio Remondino e Gabriele Guidi. «Principali tecniche e strumenti per il rilievo tridimensionale in ambito archeologico». In: *Archeologia e calcolatori* 22 (2011), pp. 169–198.

[29] S Sarkar, S Mitra e B Bhattacharyya. «Mathematical modeling for controlled electrochemical deburring (ECD)». In: *Journal of materials processing technology* 147.2 (2004), pp. 241–246.

[30] Klaus H Strobl e Gerd Hirzinger. «More accurate camera and hand-eye calibrations with unknown grid pattern dimensions». In: *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE. 2008, pp. 1398–1405.

[31] Klaus H Strobl e Gerd Hirzinger. «More accurate pinhole camera calibration with imperfect planar target». In: *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE. 2011, pp. 1068–1075.

[32] Klaus H Strobl e Gerd Hirzinger. «Optimal hand-eye calibration». In: *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE. 2006, pp. 4647–4653.

[33] Shun-Feng Su, Tar-Jyh Horng e Kuu-Young Young. «Evolutionary-based virtual training in extracting fuzzy knowledge for deburring tasks». In: *Journal of the Chinese Institute of Engineers* 27.2 (2004), pp. 193–202.

[34] Andrey Toropov. «An Effective Visualization and Analysis Method for Edge Measurement». In: *Computational Science and Its Applications – ICCSA 2007: International Conference, Kuala Lumpur, Malaysia, August 26-29, 2007. Proceedings, Part II*. A cura di Osvaldo Gervasi e Marina L. Gavrilova. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 941–950. ISBN: 978-3-540-74477-1.

[35] Juyang Weng, Paul Cohen, Marc Herniou et al. «Camera calibration with distortion models and accuracy evaluation». In: *IEEE Transactions on pattern analysis and machine intelligence* 14.10 (1992), pp. 965–980.

[36] Zhengyou Zhang. «Flexible camera calibration by viewing a plane from unknown orientations». In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 1. Ieee. 1999, pp. 666–673.

[37] G Ziliani, A Visioli e G Legnani. «A mechatronic approach for robotic deburring». In: *Mechatronics* 17.8 (2007), pp. 431–441.