UNIVERSITÀ DEGLI STUDI DI PADOVA

Facoltà di Ingegneria

Corso di Laurea in Ingegneria Informatica Magistrale

Tesi di Laurea Magistrale

Algoritmi per l'elaborazione e l'analisi di immagini tridimensionali per la guida automatica di robot industriali

Relatore Prof. Enrico Pagello
Correlatore Alberto Pretto
Correlatore Matteo Peluso

Studente	*Matricola*
Francesco Tomietto	601980

ANNO ACCADEMICO 2010-2011

Ringraziamenti

Desidero innanzitutto ringraziare il Professor Enrico Pagello per aver accettato l'incarico di relatore per la mia tesi. Inoltre, ringrazio sentitamente il Dr Alberto Pretto correlatore dell'università, per il tempo dedicatomi e per la disponibilità dimostrata nel redimere i miei dubbi durante la stesura del lavoro. Intendo poi ringraziare l'azienda EuclidLabs, in particolare Roberto Polesel e Matteo Peluso, per l'occasione offerta e per avermi seguito costantemente durante il progetto. Infine, ho desiderio di ringraziare con affetto i miei genitori ed i miei amici per il sostegno dato e per essermi stati vicino ogni momento durante tutto il percorso universitario.

Sommario

Questa tesi tratta un aspetto particolare della robotica industriale ovvero il *bin-picking*. Questa attività è molto diffusa in tutti gli ambiti dell'industria e riguarda il problema di isolare un singolo oggetto da un insieme disordinato. Una volta separato e prelevato, esso può essere avviato alla successiva fase di lavorazione (assemblaggio, packaging, trasporto, palettizzazione, ecc...). Questa operazione risulta essere un problema molto complesso ma al tempo stesso ampiamente studiato. Molte aziende al giorno d'oggi eseguono questa attività in molti impianti industriali ma in questo progetto ciò avviene in modo diverso dal solito. Inoltre va precisato che la maggior parte delle aziende esegue tale attività manualmente e non in modo automatico. Questa tesi punta ad introdurre, all'interno di un ambiente simulato, la possibilità di eseguire una scansione tridimensionale di un oggetto grazie ad un sistema Laser più Camera. Successivamente all'acquisizione dei dati, il sistema identifica l'oggetto in esame, la sua posizione ed il suo orientamento rispetto gli assi cartesiani. Tutto ciò è indispensabile per fornire le informazioni al robot per effettuare l'operazione di *bin-picking* completa, dalla scansione al prelievo dell'oggetto. Questo ovviamente avviene sempre all'interno del simulatore tridimensionale.

Questo progetto è l'evoluzione di un simulatore robot 3D fornito dall'azienda Euclid Labs. Alla fine del lavoro sarà in grado di poter eseguire le simulazioni complete del *bin-picking* e potrà giovare di altri numerosi vantaggi portati da tale attività. Uno di essi consiste nella possibilità di effettuare la scansione di un qualunque oggetto importabile nel simulatore ottenendo un risultato paragonabile all'utilizzo del sensore utilizzato nell'applicazione reale (Ruler Sick). Ciò permette all'azienda di non doversi recare dal cliente, di non dover fermare la macchina e di poter eseguire la scansione di qualunque oggetto.

I contributi innovativi di questa tesi sono numerosi in quanto la situazione di partenza presentava solo il simulatore robot. Da qui sono stati ideati e sviluppati diversi algoritmi e strategie, partendo dalla simulazione della scansione 3D, fino ad arrivare all'estrazione delle features. Tutte le tecniche proposte, che spaziano dall'utilizzo della Hough Transform per l'identificazione di rette e di cerchi nello spazio 3D, all'utilizzo del concetto di icosaedro per creare una sfera isotropa da usare in una procedura di voto di versori, hanno fornito risultati sperimentalmente soddisfacenti.

Introduzione

Questa tesi intitolata *" Algoritmi per l'elaborazione e l'analisi di immagini tridimensionali per la guida automatica di robot industriali "* nasce dall'idea di portare un'innovazione nel mondo della programmazione della robotica industriale. Negli ultimi anni la robotica ed il CAD/CAM hanno visto una continua crescita ed evoluzione, introducendo nuove tecnologie e coprendo sempre più svariate applicazioni.

Il campo in cui si andrà ad agire è il *bin-picking*. Tale tecnica consiste nell'individuazione degli oggetti disposti alla rinfusa all'interno di un contenitore. L'oggetto così individuato potrà quindi essere prelevato da un robot, per poi essere soggetto al processo industriale designato. Un aspetto fondamentale di questa attività consiste nell'identificazione dell'oggetto e della sua posizione. Questa attività al giorno d'oggi viene effettuata con diverse tecniche. Quella adottata durante questa tesi consiste nell'utilizzo di un sistema Laser più Camera per la rilevazione dell'immagine tridimensionale. Un'alternativa possibile consiste nella visione stereoscopica. Questa utilizza 2 camere poste ad una certa angolazione tra di esse. Grazie a questo principio si può ricostruire l'immagine tridimensionale utilizzando lo stesso principio dell'occhio umano. Queste sono le tecniche più conosciute ma non le uniche.

Lo scopo ultimo della tesi è lo sviluppo di un nuovo metodo per il riconoscimento di oggetti, della loro posizione e della loro orientazione rispetto gli assi cartesiani. Ciò avviene partendo dai dati ottenuti dalla scansione effettuata dal sistema Laser e Camera.

L'intero progetto su cui è stata redatta questa tesi è stato eseguito in collaborazione con l'azienda Euclid Labs. Questa è un'azienda leader nel settore della robotica, non solo a livello nazionale ma anche europeo. Euclid Labs opera in diversi campi della virtualizzazione e del CAD/CAM. Per quanto riguarda la sezione inerente al progetto, l'azienda ha sviluppato un proprio sistema di virtualizzazione degli impianti robot. Questo rappresenta uno dei punti di forza dell'azienda in quanto le ha permesso di giovare di numerosi vantaggi portati da tale novità. Più precisamente questa applicazione le permette di programmare il sistema robot off-line e di effettuare tutti i test desiderati all'interno del simulatore senza dover ricorrere al fermo macchina. Inoltre tutto ciò non necessita di recarsi dal cliente con ingenti risparmi

di tempo e di costi.

L'azienda allo stato attuale installa già sistemi robot che eseguono il *bin-picking*. Per effettuare tale attività si utilizza un insieme di procedure che sfruttano la scansione tridimensionale grazie al sensore Ruler Sick e l'immagine ottica standard. Solitamente tale procedura può essere riassunta in questi passi : si esegue una prima scansione tridimensionale del pallet contenente gli oggetti da prelevare, si identifica l'oggetto nella posizione ottimale e si danno al robot le coordinare necessarie per poterlo prelevare correttamente. Effettuata tale operazione il robot posiziona l'oggetto su un piano ed esegue una seconda acquisizione ottica. Questa rileva con maggiore qualità le caratteristiche dell'oggetto permettendo di afferrarlo con maggiore precisione.

Obiettivo di questa tesi è affrontare alcuni aspetti molto importanti che il sistema di simulazione 3D attualmente non copre.

Il primo di questi è la possibilità di poter visualizzare all'interno del simulatore le immagini acquisite dal Ruler Sick. Fino ad ora queste venivano visualizzate con dei software esterni "Sick" ma permettevano solo una visione bidimensionale dell'immagine. Con questa attività vi sarà la possibilità di esplorare l'immagine acquisita in modo tridimensionale con tutte le potenzialità del simulatore.

Il secondo è l'introduzione del sistema di scansione tridimensionale nel simulatore. Ovvero fino a questo momento l'azienda poteva simulare qualunque funzionalità del robot all'interno del virtualizzatore ma non aveva sviluppato alcuna funzionalità riguardante la visione tridimensionale. Perciò alla fine del progetto si punta ad ottenere una nuova funzionalità che permetterà di eseguire una scansione simulata all'interno del mondo 3D, come se fosse stata eseguita dal Ruler Sick reale. Ciò prevede la creazione di una base di lavoro dove poter appoggiare l'oggetto in esame, la presenza di un sistema Laser Camera e la possibilità di poter variare parametri di configurazione del sistema. Questi parametri sono la dimensione del piano di lavoro, la risoluzione dell'immagine acquisita che può variare indipendentemente lungo l'asse x ed y, la distanza che intercorre tra il piano di lavoro ed il sistema Laser Camera ed infine la distanza tra la Camera ed il Laser. Tutto ciò permette di poter personalizzare al meglio l'ambiente di simulazione rendendolo il più veritiero possibile rispetto a quello reale.

Il terzo è l'estrazione delle features dall'immagine acquisita dal simulatore. Questa fase sarà molto complessa in quanto impone l'utilizzo di molte tecniche innovative. In particolare si partirà dai dati acquisiti nella fase precedente, cioè da una nuvola di punti nel sistema tridimensionale che rappresentano gli oggetti in esame. Da questi si desidera ricavare le caratteristiche fondamentali degli oggetti che compongono l'immagine. Le features fondamentali per l'identificazione sono i bordi lineari, gli

spigoli ed i fori negli oggetti. Grazie all'identificazione di queste caratteristiche sarà possibile individuare quale degli oggetti risulta essere nella posizione ottimale per il *bin-picking*.

Quelli elencati sono gli obiettivi finali del progetto. Per l'ideazione di tale lavoro sono stati analizzati numerosi testi e paper in modo da conoscere al meglio ciò che è già stato sviluppato da altre aziende e cosa esiste in letteratura. Come si è già detto in precedenza, molte ditte effettuano già il *bin-picking* sfruttando la scansione tridimensionale ma in questo progetto l'aspetto innovativo consiste nell'effettuare tale attività nel simulatore 3D creato dall'azienda Euclid Labs. Inoltre la maggior parte delle aziende che usano la scansione tridimensionale in realtà non eseguono una vera acquisizione 3D ma per esempio sfruttano una sequenza di immagini 2D per poi ricreare l'ambiente 3D. Quanto fatto in questo progetto esegue il tutto partendo proprio dalla nuvola di punti nello spazio senza passaggi intermedi al 2D. Va precisato che sono rare le aziende che utilizzano il *bin-picking* in modo automatico, mentre nella maggior parte viene effettuato manualmente da un operatore.

L'esecuzione e la descrizione del progetto è stata divisa in fasi successive. Man mano che sono stati testati e confermati i risultati parziali si è proseguito allo step successivo.

La prima fase, che corrisponde al primo capitolo della tesi, è dedicata alla presentazione dell'azienda ospitante Euclid Labs e dei suoi principali concorrenti. Inoltre sono stati descritti degli aspetti fondamentali della grafica 3D, dell'elaborazione dei dati 3D e di alcuni metodi di acquisizione 3D cioè Stereovisione, Lidar e sistema Laser Camera Ruler Sick.

Il secondo capitolo è destinato alla spiegazione dell'importazione dei dati creati dal Ruler Sick, la loro elaborazione per la creazione della mesh relativa e la sua importazione nel simulatore creato da Euclid Labs.

Il terzo capitolo è rivolto alla spiegazione di com'è stata ideata e sviluppata l'intera sezione del progetto inerente la simulazione della scansione 3D del sistema Laser più Camera. Come descritto in precedenza tale attività è stata eseguita inserendo tutte le possibilità di parametrizzazione desiderate.

Il quarto capitolo è indirizzato all'estrazione delle features. Identificata la metodologia ideale da utilizzare, vengono descritte l'estrazione dei bordi e dei fori dalla collezione di punti nell'ambiente tridimensionale.

Il quinto ed ultimo capitolo consiste in una sequenza di test per verificare il corretto funzionamento di tre parti fondamentali del progetto. Queste sono la simulazione della scansione 3D con la relativa visualizzazione della mesh nel simulatore, l'iden-

tificazione dei bordi e dei fori dell'oggetto in esame. Va precisato che per le ultime due fasi verranno utilizzati differenti oggetti per verificare il corretto funzionamento del sistema in diverse situazioni di lavoro.

Si anticipa già da ora che l'intero progetto ha avuto successo, gli obiettivi finali sono stati raggiunti e le sperimentazioni eseguite hanno dato ragione alle scelte effettuate.

Indice

Elenco delle figure

Elenco dei listati

1

Ambiente di lavoro

Obiettivo: Illustrazione della situazione di partenza da cui inizia l'intero progetto.

1.1 Presentazione Euclid Labs

Figura 1.1: Logo aziendale Euclid Labs

In figura 1.1 é riportato il logo dell'azienda presso cui è stato effettuato il seguente progetto. Euclid Labs sviluppa soluzioni informatiche per la robotica e l'automazione industriale. In particolare : sistemi di programmazione automatica ed off-line, CAD/CAM, simulazione tridimensionale, visione artificiale e supervisione. Oltre a questi prodotti crea soluzioni personalizzate per problemi di produzione robotizzata, anche in sistemi di robot cooperanti, seguendo ogni parte dello sviluppo, dalla progettazione al collaudo. Per ricavare le seguenti informazioni è stato consultato il sito aziendale [**?**].

Un aspetto fondamentale dei software Euclid Labs è la massima semplificazione del processo di riprogrammazione dell'impianto, riducendo i tempi di fermomacchina.

La mission è l'introduzione della robotica in nuovi mercati (produzioni di piccoli lotti, esecuzione di compiti complessi) attraverso l'uso di moderni strumenti software e l'accurata analisi del processo produttivo del Cliente.

Con l'esperienza maturata e i numerosi successi, l'azienda si sta ampliando sempre più in diverse nazioni fornendo sempre nuove soluzioni per ogni settore.

Analizzando più nello specifico le potenzialità di Euclid Labs possono essere riassunte nei seguneti punti cardine.

Dal disegno al prodotto

Euclid Labs fornisce soluzioni in grado di ottenere il prodotto finito partendo dal modello CAD. L'azienda ha sviluppato un proprio ambiente di simulazione tridimensionale in grado di replicare fedelmente l'ambiente di lavoro e, in modo particolare, l'eventuale manipolatore in esso presente. In tale modo è possibile generare e verificare i percorsi necessari alla realizzazione del prodotto in modo semplice ed accurato.

Soluzioni CAD/CAM

Euclid Labs ha sviluppato dei moduli appositi da inserire nel proprio simulatore, in grado di risolvere efficacemente problemi specifici. In altre parole, il simulatore realizzato dall'azienda consiste nel cuore delle applicazioni mentre a seconda degli scopi da raggiungere nei singoli progetti vengono applicati al simulatore i pacchetti aggiuntivi necessari. Tali moduli sono caratterizzati dalla possibilità di generare il programma robot in modo completamente automatico, in programmazione offline, al fine di limitare i fermi macchina dovuti all'installazione di nuovi programmi. Alcuni tra i moduli disponibili sono : lucidatura stampi, riempimento stampi, rilevamento difetti superficiali, etc...

Sistemi di carico/scarico

Euclid Labs propone soluzioni avanzate per il carico scarico, in grado di soddisfare le più stringenti richieste in fatto di velocità o di potenza. Si possono infatti fornire impianti capaci di movimentare 1000 kg con un singolo robot oppure molto di più con sistemi di cooperazione multi-robot. Tali sistemi possono essere montati su slitta per aumentare la flessibilità dell'impianto. I dati per lo scarico possono essere ricavati da sensori, sistemi di visione o da altro software esterno (nesting).

Software per Macchinari Speciali

Euclid Labs è in grado di seguire la progettazione e lo sviluppo di software per macchine speciali per l'automazione, curando in particolar modo gli aspetti relativi a :

1. gestione completa del processo,
2. controllo del movimento,
3. sistemi di controllo e verifica,
4. sistemi di visione standard e tridimensionali,
5. supervisione della macchina,

6. connessione ad altre macchine o sistemi gestionali.

L'esperienza dell'azienda ha permesso di fornire soluzioni complete in diversi settori tra cui :

1. Gestione del processo del taglio bidimensionale con macchine a lama ed a getto d'acqua,
2. Molatura lenti per occhiali,
3. Taglio e sbavatura lamiere piane
4. Incisione su percorsi tridimensionali, a laser e a getto d'acqua,
5. Taglio tubi, anche di grandi dimensioni (superiori a 1500mm).

Prelievo automatico 3D
La visione per la guida robot ha raggiunto una nuova dimensione con il pacchetto software Euclid Labs Pick3D. Il software permette di recuperare pezzi disposti a caso in un volume, calcolandone non solo la posizione ma anche le orientazioni nello spazio.

L'applicazione si rileva particolarmente utile per riprendere e lavorare componenti che non possono essere vibrati per fattori di forma o per non comprometterne la qualità, componenti che vengono disposti in modo disordinato subito dopo la produzione, oggetti che arrivano su nastri anche sovrapposti tra loro.

Il software sviluppato da Euclid Labs si occupa di ricostruire l'immagine tridimensionale, identificare la posizione di un oggetto che possa essere prelevato, verificare se il robot può raggiungerlo senza collisioni eseguendo una simulazione della manovra nella cella.

Il robot una volta prelevato il pezzo può alimentare presse o macchine utensili, oppure eseguire lavorazioni o assemblaggi.

Quello appena descritto è l'ambito aziendale in cui sarà centrata questa tesi. In particolare si vuole effettuare questa operazione di Pick3D all'interno del simulatore aziendale.

Servizi per costruttori di macchine ed impianti
Euclid Labs mette la propria esperienza a disposizione dei costruttori di macchine ed impianti speciali, aggiungendo al loro know-how le potenzialità della robotica e dei moderni sistemi di programmazione off-line, anche di sistemi multi-robot. I servizi spaziano dalla completa integrazione del robot con una macchina, alla programmazione di un sistema CAD/CAM specifico per l'applicazione. Le tecnologie dell'azienda si applicano ai settori più vari dal metallo al legno, dal tessile al calzaturiero, dall'oreficeria alle lenti per occhiali.

1.2 Concorrenti Euclid Labs

Qui di seguito verranno introdotti alcuni dei principali concorrenti di Euclid Labs. Questi coprono il settore della visione 3D oppure CAM, mentre Euclid Labs si sta specializzando sempre più in ambe due i settori. Le ditte che verranno citate sono Scape tecnoligies, 3ideo e almacam.

1.2.1 Scape tecnologies

Figura 1.2: Logo aziendale Scape

La prima azienda che verrà descritta è Scape tecnologies [5].Tale azienda con sede in Kochsgade 31C, 3. sal DK-5000 Odense C DENMARK, dichiara come propria mission di sviluppare, vendere e implementare bin picking all'avanguardia per la produzione e l'industria manifatturiera.
Scape Technologies sviluppa e vende sistemi bin picking altamente flessibili ed automatizzati. Come è ben noto i robot Bin-picking sono applicabili in molte aree di produzione dall'automobilismo a molto altro. Scape Technologies lavora a stretto contatto con le aziende per ottenere ottimi risultati aumentando la produttività e diminuendo il lavoro manuale del cliente. L'azienda offre un'ottima soluzione bin picking con bracci robotizzati per individuare e scegliere i singoli pezzi originalmente messi in modo casuale accatastandoli generalmente in un pallet. Per effettuare tali operazioni sfrutta le nuove tecnologie di visione 3D in tempi sufficientemente veloci, accurati ed economicamente competitivi.

Riguardo l'aspetto a noi piu interessante, per la visione 3D, utilizza una camera ed un laser rosso sfruttando software proprietario.

Le soluzioni proposte dall'azienda sono le seguenti :

1. Loading of Disc Shaped Parts,
2. Loading of Pipe Shaped Parts,
3. Feeding of Semi-Structured Parts,
4. Assembly of Multiple Parts,
5. Reloading Parts to Next Operation After Press,
6. Precision Loading of Non-Symmetrical Parts,
7. Loading of Cube Shaped Parts for CNC Machining.

1.2.2 3ideo

Figura 1.3: Logo aziendale 3ideo

Quest'altra azienda con sede in 31, rue Lamartine 62000 ARRAS - FRANCE (Web: www.3ideo.com) [6]. 3ideo è un fornitore di prodotti software standard per la robotica, machine vision e inline inspection. La mission azienda consiste nello sviluppo di tecnologie innovative per l'industria sfruttando il CAD 3D. Sfruttando i nuovi sistemi di visione ed i componenti meccanici standard sono stati realizzati numerosi impianti di bin picking ed altre applicazioni.

L'azienda fornisce le seguenti soluzioni : Pick3D , Handling3D, Guidance3D, Identification3D, InGauge3D.

1. Pick3D : Far diventare il robot intelligente per le operazioni di bin-picking,
2. Handling3D : Accurare le operazioni necessarie per le applicazioni di assemblaggio,
3. Guidance3D : Pre attività accurate per migliorare il movimento,
4. Identification3D : Rendere il processo flessibile,
5. Inguage3D : Rendere il controllo di qualità 3D sistematico.

Mentre le **principali attività aziendali** sono :

1. 3Dmachine vision,
2. Automotive,
3. Aerospace,
4. Foundry and Stamping,
5. General mechanical,
6. Medical / Biomechanics,
7. Robotics,
8. Special machines,
9. System integators.

Nel campo del Pick3D l'azienda offre i seguenti moduli :

Automatic bin pikcing module :
- tutte le parti nel piano di lavoro vengono identificate con precisione,
- vengono rilevati gli oggetti con zone di sovrapposizione,
- il movimento della pinza viene simulato in modo da rilevare potenziali collisioni.

Real-time visualization module :
- visualizzazione della scena CAD virtuale integrando i dati della scansione,
- Visualizzazione dello stato, dei warning e degli errori del Pick3D.

Open communication module :
- Comunicazione Scan - head cia TCP-IP, CameraLink,...
- Comunicazione robot via TCP - IP, porta seriale.

Approfondendo l'aspetto più di nostro interesse, ovvero il bin picking, 3ideo punta a dare un servizio sicuro, affidabile, collaudato, flessibile, universale e rapido. 3ideo con le proprie proposte garantisce di raccogliere oggetti distribuiti in modo casuale all'interno di un pallet garantendo di evitare alcuna collisione. Ed il tutto in tempo reale.

Per quanto riguarda le tecnologie utilizzate 3ideo come Euclid Labs può utilizzare svariati robot tra cui ABB, Fanuc, Kuka, Staubli, etc... ed i sensori utilizzati sono sempre SICK IVP Ranger e Ruler.

1.2.3 Alma

alma

Figura 1.4: Logo aziendale

L'ultima azienda citata è Alma che possiede numerose sedi in tutto il mondo (Web : www.almacam.com) [7]. Alma è conosciuta da più di 30 anni come lo specialista del software automatico applicato al taglio di materiali piani. Grazie al suo Dipartimento Industrial software, è attualmente radicata in tre paesi europei, in Cina, in Brasile e negli Stati Uniti, è uno dei leader nella pubblicazione di software CAD-CAM per la lavorazione della lamiera, taglio e robotica.

Nel 1994, Alma ha esteso la sua gamma di software applicativi per la programmazione off-line di robot nel campo della saldatura ad arco e taglio 3D. In conseguenza di questo, Alma offre la gamma più completa di software CAD-CAM per la lamiera e saldatura meccanizzata. I prodotti software applicativi della gamma act coprono tutte le tecnologie usate in queste lavorazioni: punzonatura, taglio 2D, fresatura, tranciatura, piegatura, taglio 3D, taglio di tubi, saldatura ad arco ...

Questi prodotti sono distribuiti dalla rete di distribuzione di Alma. L'azienda commercializza i suoi algoritmi, sotto forma di componenti software, e fornisce la preziosa esperienza tecnica per lo sviluppo del software applicativo.

Uno degli ambiti principali di Alma è act/cut 3D. Da una precisa rappresentazione 3D della macchina (cinematica, limiti degli assi ...) e il suo ambiente, act / cut 3d permette una definizione molto rapida dei profili di taglio su pezzi importati dal CAD. Inoltre, il software consente di modellare automaticamente le attrezzature in base al modello 3D della parte. Poi, il software genera le traiettorie con un potente algoritmo che ottimizza i percorsi con prevenzione delle collisioni. Mentre la creazione e la convalida del programma di taglio sono facilmente realizzati grazie alle funzioni di simulazione realistica e le funzioni di controllo automatico combinato con visualizzazione delle anomalie.

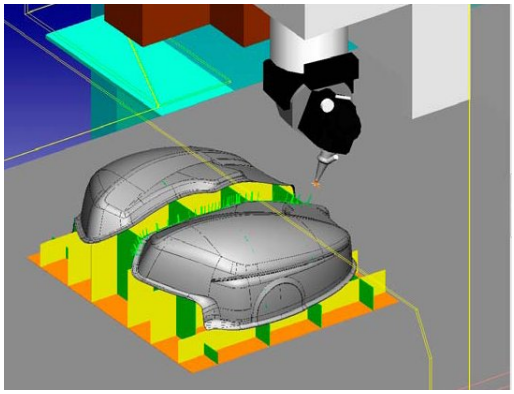


Figura 1.5: Esempio di cut 3D

Per quanto riguarda il settore software dell'azienda si divide in 3 sezioni :

1. Automatic Nesting

2. 1D / 2D / 3D nesting optimization

3. Robotic Applications development

Figura 1.6: Automatic Nesting

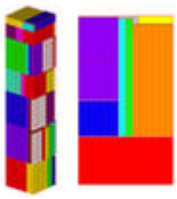


Figura 1.7: 1D / 2D / 3D nesting optimization

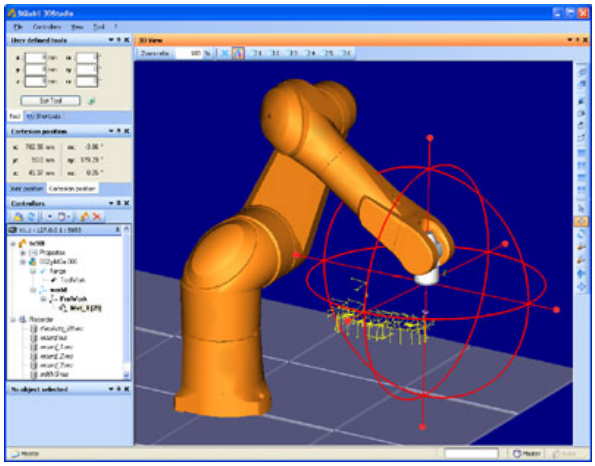


Figura 1.8: Robotic Applications development

1.3 3D Vision

Prima di entrare negli aspetti più tecnici del progetto è bene definire il contesto in qui è realizzato. Al giorno d'oggi ci sono numerose novità riguardo il mondo 3D ma questo ambito realmente è molto ampio. Si riferisce ad una varietà di differenti tecnologie che forniscono una reale prospettiva delle immagini in tre dimensioni. Ovvero dando anche la sensazione della profondità. Le immagini 3D possono essere visualizzate in una stampa, in un computer, al cinema oppure anche in TV. Le nuove tecnologie 3D sono utilizzate in svariati campi tra cui la grafica 3D, computer-

aided design (CAD) ed animazione 3D. Tali immagini posso essere visualizzate dallo spettatore con la sensazione di profondità.

La grafica 3D racchiude tutto ciò che riguarda la creazione, la visualizzazione e la manipolazione di oggetti 3D nel computer. 3D CAD ed i programmi di grafica tridimensionale permettono di creare oggetti con i seguenti parametri, x y z, cioè larghezza, altezza e profondità. Come entità 3D possono essere ruotati e visualizzati da tutte le angolazioni così come essere scalati più o meno grandi. Inoltre permettono l'illuminazione durante la fare di rendering.

Alcuni dei principali ambiti del 3D sono i seguenti: Virtual Reality (Realtà Virtuale), Creating 3D - Stereoscopic Images (Creazione 3D - Immagine stereoscopica), 3D Stills, Cinema 3D, 3D nei Computers e nelle TV.

Realtà Virtuale
La realtà virtuale è un tipo di visualizzazione 3D che viene utilizzato in simulatori di volo così come nei giochi e di intrattenimento. Indossando gli occhiali 3D si ha l'illusione della realtà avendo la sensazione di essere immersi in un ambiente a 360 gradi. Tali sistemi possono impiegare il tradizionale rendering 3D o possono utilizzare metodi di visualizzazione 3D come descritto di seguito.

Creazione 3D - Immagini stereoscopiche
La creazione 3D si ottiene catturando la scena osservandola da due angoli diversi, cioè corrispondenti alla distanza tra gli occhi di una persona (circa 65 mm). Quando osserviamo l'immagine 3D, le immagini sono dirette nuovamente all'occhio sinistro e destro ed il cervello percepisce l'illusione della profondità. Le immagini sono esposte insieme, ma separate dai colori, dalla polarizzazione o alternarsi rapidamente dei fotogrammi. Gli occhiali 3D utilizzati devono essere appositi per il metodo scelto, in modo che le immagini per l'occhio sinistro e destro siano filtrate correttamente.

Stills 3D
Queste sono immagini 3D fisse create da un particolare binocolo costruito nel 16° secolo. Negli anni questa tecnica subì numerose evoluzioni e nel 1800 si sviluppò lo stereoscopio. Mentre al giorno d'oggi ci sono le fotocamere 3D o una lente 3D su una normale fotocamera.

Cinema 3D
Nel 1950, vi fu il primo film 3D ma utilizzava delle tecnologie arretrate mentre al giorno d'oggi la visione dei film 3D sfrutta soluzioni molto più potenti ed ha raggiunto un'espansione in larga scala. Questo è avvenuto a tal punto da renderla una routine giornaliera al cinema ed un po' alla volta entrerà nelle case di tutti.

3D su computer e televisori
Nel 2009, NVIDIA ha introdotto il suo sistema 3D Vision con la quale ha portato la propria esperienza nel mondo dei giochi 3D, ovviamente l'evento è stato accolto

con entusiasmo dagli appassionati dei giochi. Nel corso degli ultimi anni questa tecnologia è stata impiegata in moltissime apparecchiature in modo da fornire sempre più scelta ai potenziali clienti, dai videogiochi 3D, ai film 3D alle TV da casa con tecnologie 3D integrate.

1.4 3D Reconstruction

Nella Computer Vision e nella Computer Graphics, la 3D reconstruction è il processo attraverso cui si cattura la forma e l'aspetto degli oggetti reali. Questo processo può essere effettuato sia con metodi attivi che passivi. Inoltre possiamo scegliere di lasciar cambiare la forma del modello durante il tempo, in questo caso si parla di modellazioni non-rigide oppure spazio-temporali.

Con i metodi attivi si interferisce con l'oggetto meccanicamente. L'esempio più semplice del metodo meccanico può essere l'utilizzo di un profondimetro per misurare lo spessore di un oggetto.

Mentre i metodi passivi di ricostruzione 3D non interferiscono con l'oggetto ricostruito, usano solo un sensore per misurare la radiazione riflessa o emessa dalla superficie dell'oggetto al fine di dedurre la struttura 3D. Tipicamente, il sensore è un sensore di immagine in una macchina fotografica sensibile alla luce visibile e l'ingresso al metodo è un insieme di immagini digitali (uno, due o più) o video.

Un altro metodo per la 3D reconstruction è partendo dai dati che si conoscono dell'oggetto. Ovvero conoscendo tutte le sue caratteristiche fisiche di misure e composizione è possibile ricostruire tale oggetto ottenendo un modello 3D.

1.5 StereoVision, Lidar e Ruler

Ci sono diverse tecniche e apparecchiature per ottenere la 3D Vision. Ognuna delle quali presenta dei pregi e dei difetti. Qui di seguito sono rappresentate tre di queste, dandone i concetti fondamentali per comprenderle.

1.5.1 StereoVision

La prima è la StereoVision basata su telecamere tradizionali in quanto potenzialmente più economiche rispetto ad altri sensori basati su tecnologie di tipo laser, etc. Tra le diverse tecniche di Computer Vision note in letteratura e create per la ricostruzione della struttura tridimensionale di una scena osservata da una o più telecamere (shape from motion, shape from shading, shape from texture) la visione stereoscopica è quella che ha riscosso la maggiore attenzione. Il principio alla base

della visione stereoscopica, noto sin dal rinascimento, consiste in una triangolazione mirata a mettere in relazione la proiezione di un punto della scena sui due (o più) piani immagine delle telecamere (tali punti sono denominati punti omologhi) che compongono il sistema di visione stereoscopico. L'individuazione dei punti omologhi consente di ottenere una grandezza denominata disparità mediante la quale, conoscendo opportuni parametri del sistema stereoscopico, è possibile risalire alla posizione 3D del punto considerato.
La raccolta di informazione riguardanti queste tematiche è stata effettuata da queste fonti : [28], [29], [30].

Geometria di un sistema stereoscopico
La trasformazione prospettica che mappa un punto dello spazio nel piano immagine di una telecamera implica la perdita dell'informazione relativa alla distanza. Questo può essere facilmente rilevato osservando la figura seguente nella quale due distinti punti P e Q nello spazio, intersecano lo stesso raggio che parte dal centro ottico O_1 della Camera e passa per i due punti. Inoltre, questi punti corrispondono allo stesso punto p=q nel piano immagine.

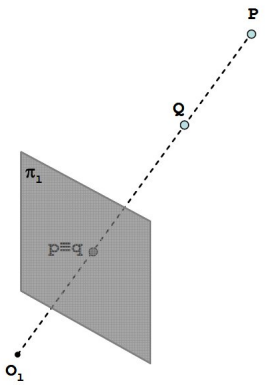


Figura 1.9: Proiezione di 2 punti distinti nel piano immagine

Un metodo per poter risalire a quale punto dello spazio corrisponda la proiezione di un punto sul piano immagine di una telecamera consiste nell'utilizzo di due o più telecamere. Infatti, come mostrato nella figura seguente nel caso di un sistema composto da due telecamere, tra tutti punti nello spazio che giacciono sul raggio che passa per il centro ottico O_l e il punto q, proiezione di Q sul piano immagine πl, al più un solo punto, punto omologo, viene proiettato q anche sul piano immagine πr. La determinazione dei punti omologhi consente di mettere in relazione le proiezioni dello stesso punto sui due piani immagini e di risalire, mediante una procedure denominata triangolazione, alle coordinate dei punti dello spazio rispetto ad un sistema di riferimento opportuno. Sia dato un punto q su un piano immagine πl, proiezione del punto Q appartenente allo spazio 3D. Per ottenere, attraverso la triangolazione, le coordinate 3D del punto nello spazio è necessario determinare

il punto omologo q nel piano immagine πl. Tale problema, dato il punto q nel piano immagine πl, richiederebbe una ricerca bidimensionale del punto omologo q allinterno del piano immagine πr.

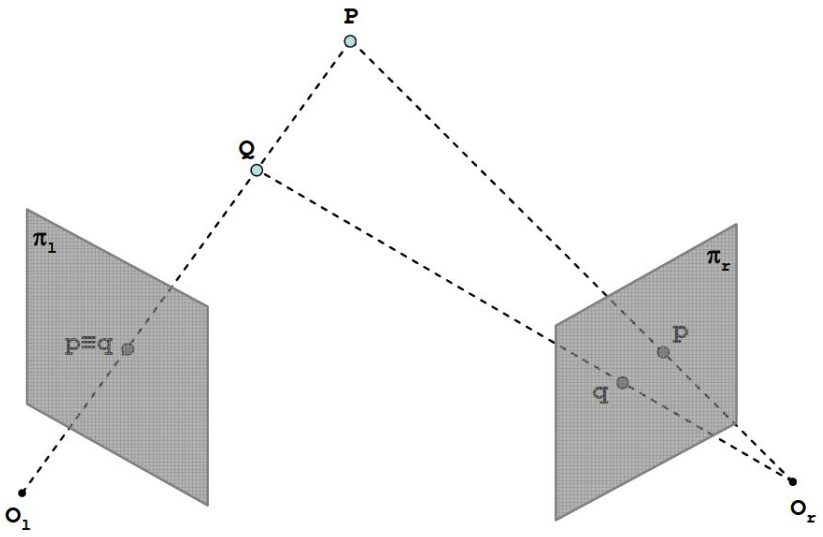


Figura 1.10: Principio alla base di un sistema stereoscopico

In realtà, sfruttando una particolare caratteristica della geometria del sistema stereoscopico, è possibile effettuare la ricerca del punto omologo in uno spazio monodimensionale. Infatti, come mostrato nella figura seguente, gli omologhi di tutti i punti dello spazio che potrebbero risultare proiezione nello stesso punto q del piano immagine πl, punto p proiezione di P o lo stesso punto q proiezione di Q giacciono sulla retta generata dall'intersezione tra il piano immagine πl e il piano denominato piano epipolare che passa per la retta O_lQ_r e i due centri ottici O_l e O_r. Tale vincolo, denominato vincolo epipolare, consente di limitare lo spazio di ricerca dei punti omologhi ad un segmento di retta semplificando considerevolmente il problema delle corrispondenze sia da un punto di vista della complessità algoritmica sia per quanto concerne la correttezza della soluzione. Si fa notare che il problema delle corrispondenze non necessariamente ha soluzione: infatti a causa della diversa posizione delle telecamere che compongono un sistema di visione stereoscopico nello spazio è possibile che un punto non risulti proiettato su tutti i piani immagini delle telecamere. In tal caso il problema delle corrispondenze non ha soluzione e non è possibile determinare la distanza del punto esaminato dalle telecamere (occlusioni). Un sistema di visione stereoscopico è completamente caratterizzato mediante i parametri intrinseci ed estrinseci. I primi consentono di definire la trasformazione che mappa un punto dello spazio 3D nelle coordinate del piano immagine di ogni telecamera e risultano le coordinate relative al piano immagine del principal point (punto di intersezione tra il piano immagine e la retta ortogonale al piano immagine stesso passante per

il centro ottico), la distanza focale, ed eventualmente altri parametri che descrivono altre caratteristiche del sensore (distorsione delle lenti, forma dei pixels, etc). I parametri estrinseci rappresentano le posizioni di ogni telecamera rispetto ad una sistema di riferimento noto.

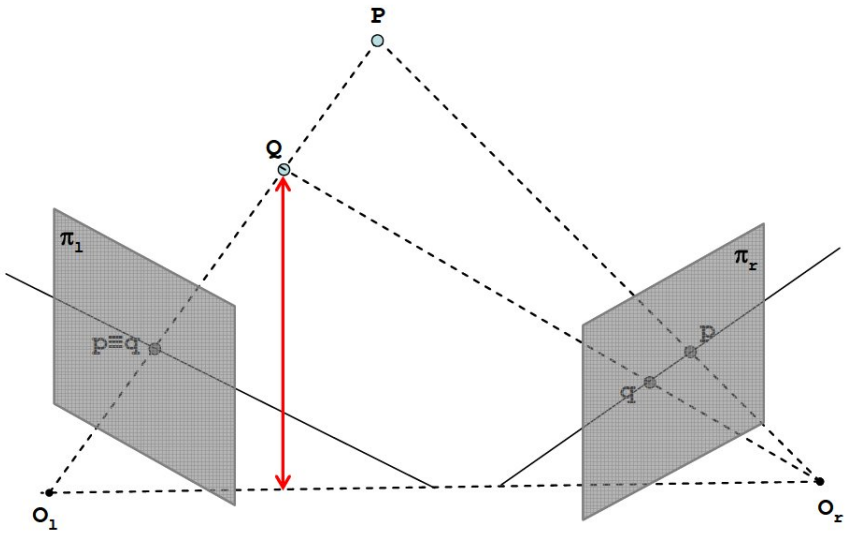


Figura 1.11: Vincolo Epipolare

La determinazione dei parametri intrinseci ed estrinseci, ottenuta mediante la procedura di calibrazione, consente quindi di descrivere completamente il sistema stereoscopico ed in particolare di inferire informazioni relative alle coordinate dei punti nello spazio mediante la triangolazione di punti omologhi.

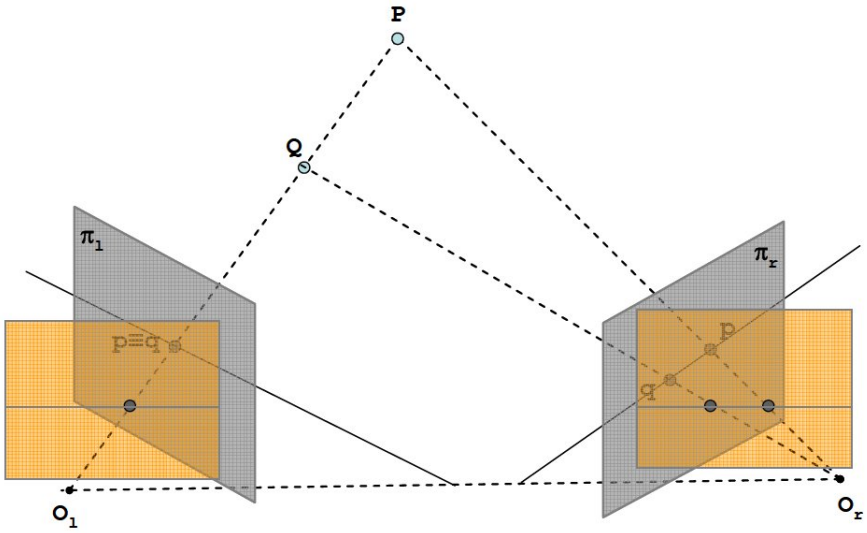


Figura 1.12: Immagine in forma standard

La conoscenza dei parametri intrinseci ed estrinseci consente anche di trasformare le immagini acquisite dal sistema stereoscopico al fine di produrre un sistema virtuale nel quale i piani immagine delle telecamere giacciono sullo stesso piano e nel quale la ricerca dei punti omologhi avviene esaminando le medesime righe nei diversi piani immagine. Tale configurazione del sistema stereoscopico ottenuta mediante una procedura denominata rettificazione è mostrata in figura 1.12 Immagini in forma standard. Osservando la figura si può notare come il sistema risultante dopo la rettificazione risulti composto da due piani immagine virtuali l e r giacenti sullo stesso piano. Le immagini stereoscopiche ottenute da una sistema rettificato sono denominate immagini in forma standard. Si può osservare infine che nelle immagini in forma standard i piani xy dei sistemi di riferimento centrati nei centri ottici delle due telecamere sono complanari.

I passi per risalire alle coordinate dei punti dello spazio mediante un sistema di visione sono calibrazione, acquisizione, rettificazione, matching stereo, trinagolarizzazione.

Per concludere la panoramica sulla stereovisione una della possibili applicazioni di questa tecnica consiste nel 3D people counting and tracking. Di seguito c'è un'immagine d'esempio.

Figura 1.13: Esempio di 3D peolple counting and tracking

1.5.2 Lidar

LIDAR acronimo di Light Detection and Ranging o Laser Imaging Detection and Ranging è una tecnica di telerilevamento che permette di determinare la distanza di un oggetto o di una superficie utilizzando un impulso laser. Come per il radar, che al posto della luce utilizza onde radio, la distanza dell'oggetto è determinata misurando il tempo trascorso fra l'emissione dell'impulso e la ricezione del segnale retrodiffuso. La sorgente di un sistema LIDAR è un laser, ovvero un fascio coerente di luce ad una ben precisa lunghezza d'onda, che viene inviato verso il sistema da osservare. La tecnologia Lidar ha applicazioni in geologia, sismologia, rilevamento remoto e fisica dell'atmosfera.

Il lidar usa lunghezze d'onda ultraviolette, nel visibile o nel vicino infrarosso; questo rende possibile localizzare e ricavare immagini e informazioni su oggetti molto piccoli, di dimensioni pari alla lunghezza d'onda usata. Perciò grazie a queste caratteristiche il lidar è molto sensibile agli aerosol e al particolato in sospensione nelle nuvole ed è molto usato in meteorologia e in fisica dell'atmosfera. Affinché un oggetto rifletta un'onda elettromagnetica, deve produrre una discontinuità dielettrica; alle frequenze del radar un oggetto metallico produce una buona eco, ma gli oggetti non-metallici come pioggia e rocce producono riflessioni molto più deboli, e alcuni materiali possono non produrne affatto, risultando a tutti gli effetti invisibili ai radar. Questo vale soprattutto per oggetti molto piccoli come polveri, molecole e aerosol. I laser forniscono una soluzione a questi problemi: la coerenza e densità del fascio laser è ottima, e la lunghezza d'onda è molto più breve dei sistemi radio, e può andare dai 10 micron a circa 250 nm. Onde di questa lunghezza d'onda sono riflesse ottimamente dai piccoli oggetti, con un comportamento detto retrodiffusione; il tipo esatto di retrodiffusione sfruttato può variare: in genere si sfruttano la diffusione Rayleigh, la diffusione Mie e la diffusione Raman, oltre che la fluorescenza. Le lunghezza d'onda dei laser sono ideali per misurare fumi e particelle in sospensione aerea, nuvole e molecole nell'atmosfera. Un laser ha in genere un fascio molto stretto, che permette la mappatura di caratteristiche fisiche con risoluzione molto

alta, paragonata a quella del radar. Inoltre molti composti chimici interagiscono più attivamente con le lunghezze d'onda del visibile che non con le microonde, permettendo una definizione anche migliore: con adatte combinazioni di laser permettono la mappatura remota della composizione dell'atmosfera rilevando le variazioni dell'intensità del segnale di ritorno in funzione della lunghezza d'onda.

In genere ci sono due tipi di sistemi lidar: lidar a microimpulsi e lidar ad alta energia. I sistemi a microimpulsi sono stati sviluppati recentemente, come risultato della sempre crescente potenza di calcolo disponibile e dei progressi nella tecnologia dei laser. Questi nuovi sistemi usano potenze molto basse, dell'ordine di un watt, e sono spesso completamente sicuri. I lidar ad alta energia invece sono comuni nello studio dell'atmosfera, dove sono impiegati per il rilevamento di molti parametri atmosferici come altezza, stratificazione e densità delle nubi e proprietà del particolato che contengono temperatura, pressione, umidità, venti, concentrazioni di gas traccia.

Un lidar è composto dai seguenti sistemi: Laser, Scanner e ottica, Ricevitore ed elettronica, Sistemi di localizzazione e navigazione.

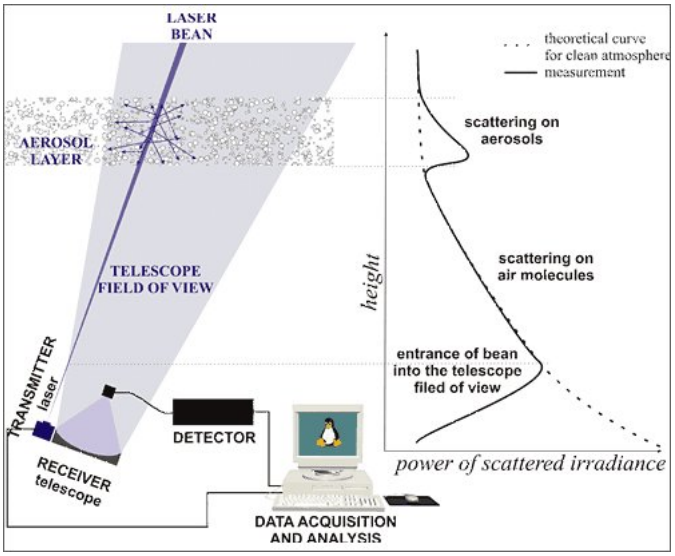


Figura 1.14: Schematizzazione Lidar

1.5.3 Ruler

Questa sezione è dedicata alla spiegazione del metodo e delle apparecchiature utilizzate durante tutto il progetto per la scansione delle immagini 3D. L'apparecchio utilizzato è fabbricato dall'azienda SICK IVP con sede in Wallenbergs gata 4 SE-583 35 Linköping Sweden. Le informazioni utilizzate per scrivere questa sezione sono state ricavate dal sito aziendale Sick e da [2].La Sick produce moltissimi sensori a

seconda degli scopi desiderati. Al nostro fine interessano i sensori di visione o più precisamente le 3DCamera. Sick offre 2 scelte principali ovvero Ranger e Ruler.

Figura 1.15: SICK Ranger e Ruler Cameras

Le due scelte fondamentalmente utilizzano lo stesso principio e matematica ma come si può vedere nella figura seguente Ranger ed il Laser sono due oggetti fisicamente separati mentre per il Ruler sono prodotti già dalla fabbrica in un pezzo unico.

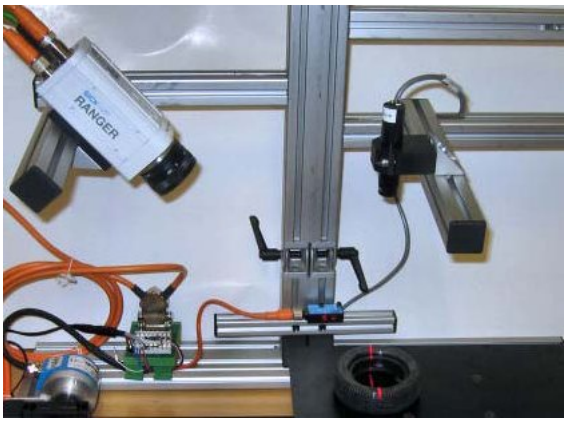
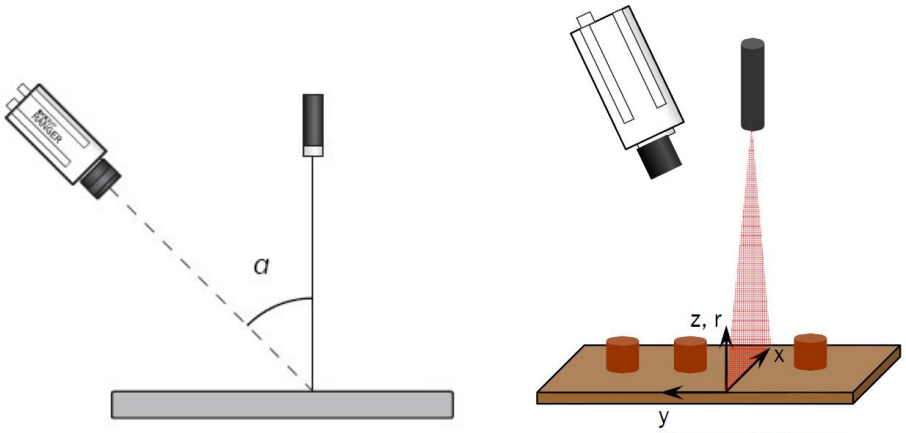


Figura 1.16: Ruler Cameras

Data la possibilità per il Ranger di poter scegliere la configurazione Camera-Laser, Sick offre queste possibili configurazioni a seconda delle necessità o preferenze.

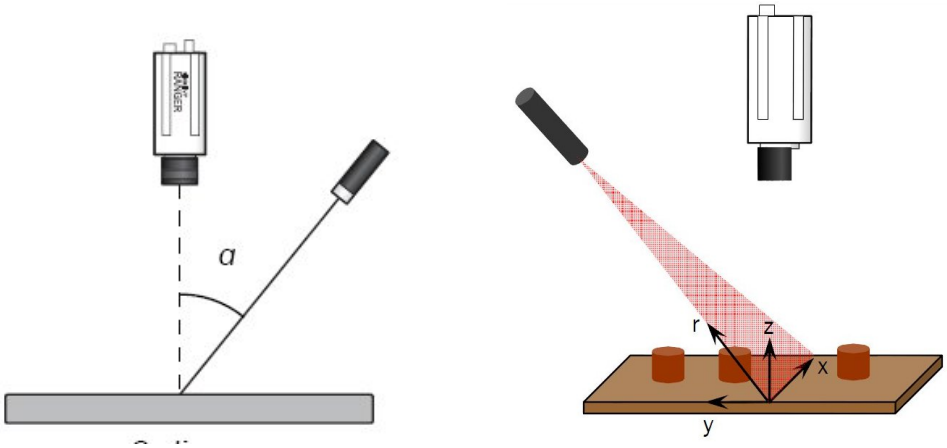
Reverse Odinary



(a) Reverse Ordinary a. (b) Reverse Ordinary b.

Figura 1.17

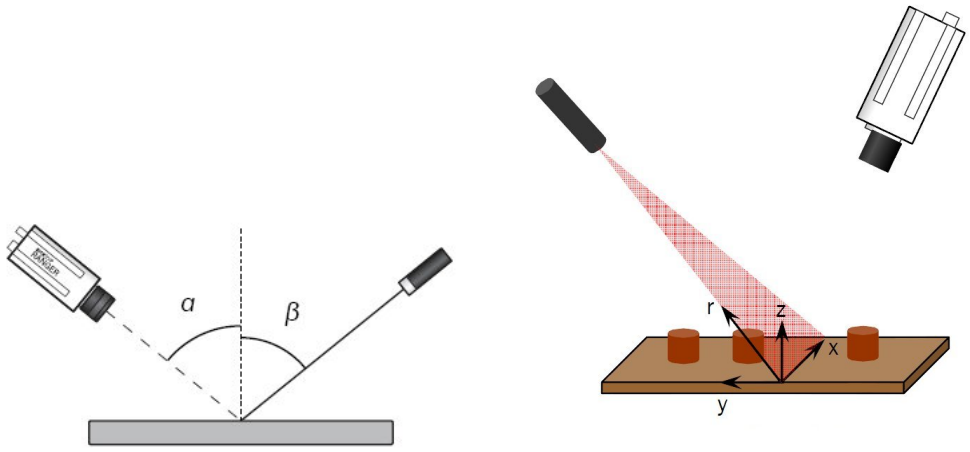
Ordinary Geometry



(a) Ordinary Geometry a. (b) Ordinary Geometry b.

Figura 1.18

Specular Geometry



(a) Specular Geometry a.

(b) Specular Geometry b.

Figura 1.19

Look-away

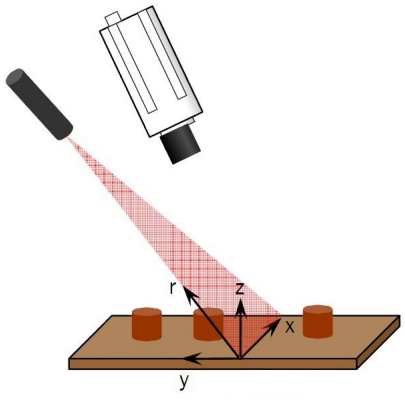


Figura 1.20: Ruler Cameras

A seconda della configurazione geometrica scelta ovviamente cambierà la geometria e la risoluzione. In effetti per esempio la risoluzione lungo l'asse Z cambierà a seconda delle seguenti formule.

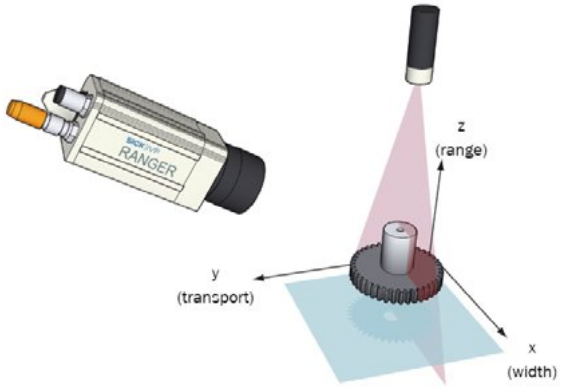


Figura 1.21: Assi

Geometry	Approximate range resolution
Ordinary	$\Delta Z \approx \Delta X / \tan(\alpha)$
Reversed ordinary	$\Delta Z \approx \Delta X / \sin(\alpha)$
Specular	$\Delta Z \approx \Delta X \cdot \cos(\beta) / \sin(\alpha+\beta)$ If $\alpha = \beta$: $\Delta Z \approx \Delta X / 2 \cdot \sin(\alpha)$

Figura 1.22: Geometry

Principi di base

Data una configurazione di base come detto in precedenza il sistema è composto da una particolare camera 3D e da un laser. Il laser proietta semplicemente un fascio nell'oggetto da analizzare. Questo ha l'unica particolarità di dover essere perpendicolare al piano per rendere più semplici i calcoli geometrici. La camera invece ha al suo interno un sensore che le permette di trovare il fascio laser. Questa operazione viene eseguita in modo particolare. Ovvero, ogni immagine rilevata dalla Camera viene divisa in colonne verticali ed analizzate in parallelo. Il sistema interno della camera rileva il punto del laser in ogni singola colonne con diverse possibili tecniche che verranno illustrate in seguito. La linea laser produce un picco distinto di luce su un certo numero di pixel lungo una colonna del sensore e il centro di questo picco sarà definito come il punto d'impatto.

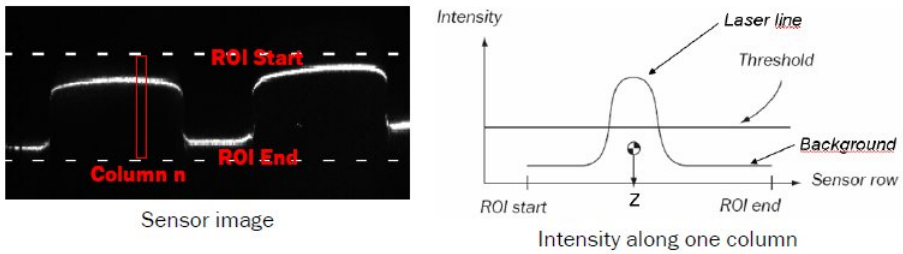


Figura 1.23: Laser

Esistono 3 tipologie di algoritmo per l'identificazione del laser:

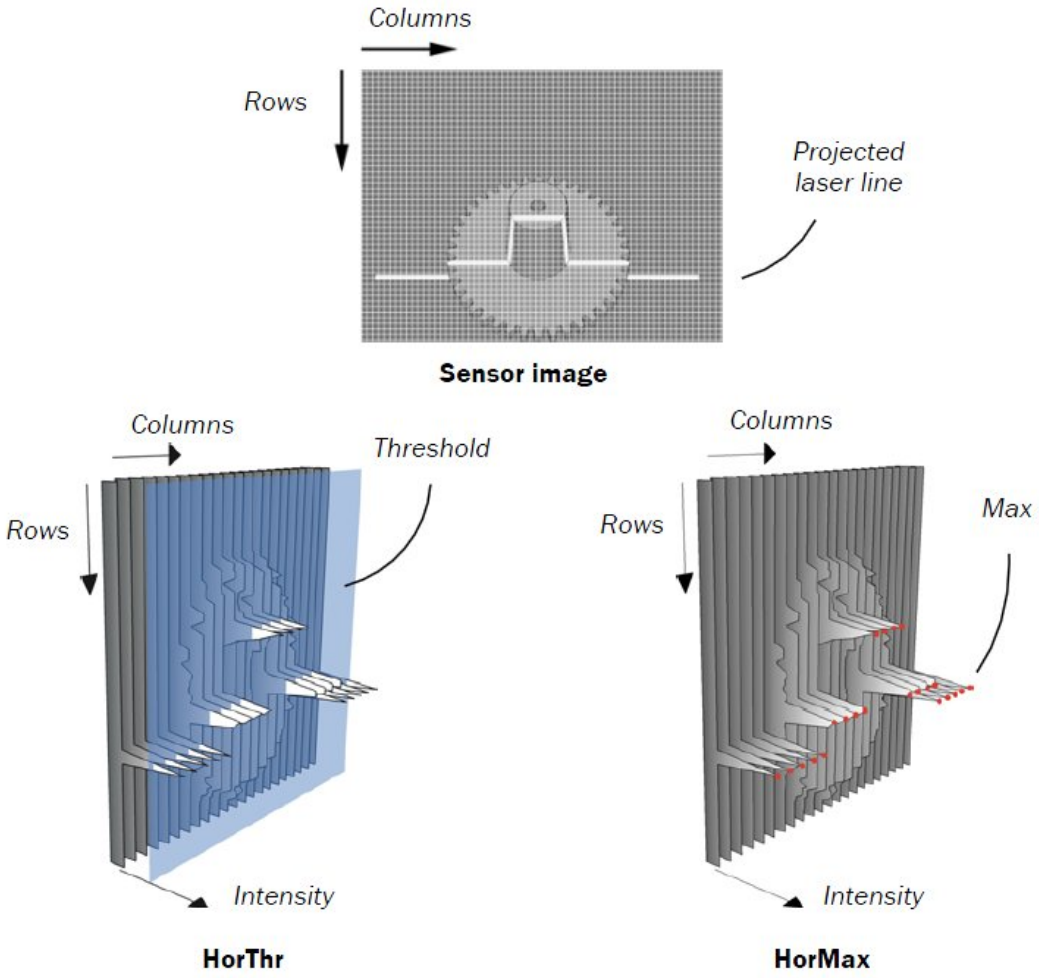


Figura 1.24: Rappresentazione dei metodi

Hi3D

Hi3D è un algoritmo a medio/alta risoluzione (fino a 4 kHz). Utilizza una soglia per eliminare il rumore di fondo prima di passare alla conversione AD. L'altezza di risoluzione è 1/16 di pixel.
Questa è la tecnica maggiormente utilizzata nei progetti Euclid Labs dato che garantisce una maggiore precisione.

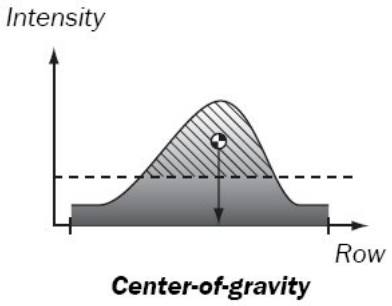


Figura 1.25: Hi3D

HorThr

L'algoritmo HorThr è il più veloce algoritmo 3D (fino a 35 kHz). Per ogni colonna l'algoritmo cerca il primo punto al di sopra di una certa soglia e successivamente anche quando torna al di sotto. Questi 2 punti sono considerati l'inizio e la fine della linea laser. L'algoritmo essere impostato anche per utilizzare due soglie in cui vengono estratte quattro posizioni.

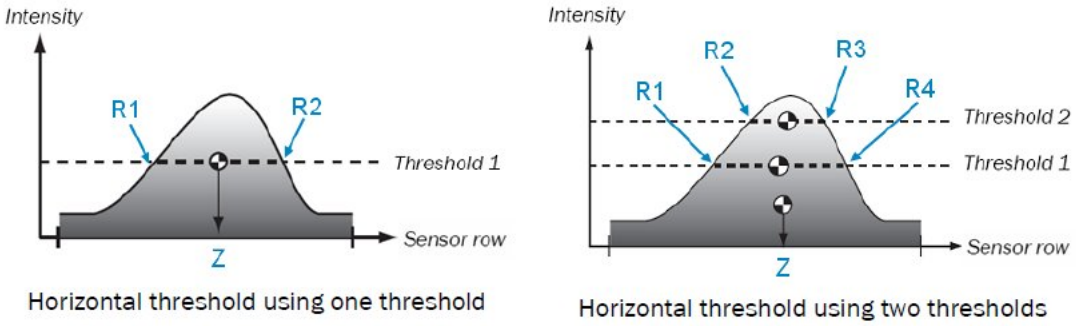


Figura 1.26: Hor

HorMax

L'algoritmo HorMax lavora ad un velocità media di max 5 kHz. Per ogni colonna trova il valore di massima intensità. Anche in questo caso può essere inserita una soglia per eliminare il rumore di fondo. L'altezza della risoluzione è di 1 o $\frac{1}{2}$ pixel.

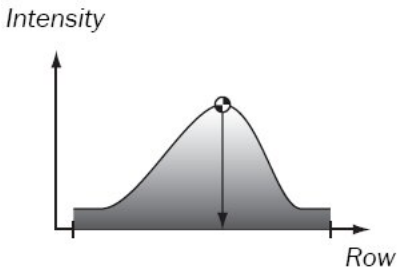


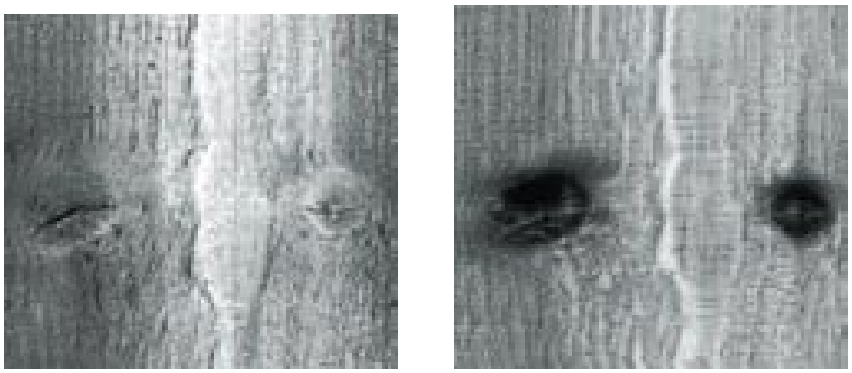
Figura 1.27: Max

Component	Name	Measures	Note
HorThr	Horizontal threshold	Range	Fast – up to 10 000 profiles/s. Resolution ½ or ¼ pixel.
HorMax	Horizontal max	Range, intensity	Resolution 1 or ½ pixel.
HorThrMax	Horizontal max and threshold	Range, intensity	Resolution ½ pixel.
Hi3D	Hi-resolution 3D	Range, intensity, scatter[2]	High range resolution – $^{1}/_{16}^{th}$ pixel.

Figura 1.28: Geometry

Oltre alle precedenti tecniche c'è un'altro algoritmo utilizzato per scopi particolari. Ovvero lo Scatter.

Scatter
Lo Scatter rileva il comportamento del laser appena sotto la superficie dell'oggetto. Lo Scatter è composto da 2 subcompents, laser diretta (restituisce un cosiddetto bilancio 2D) e dispersione (banda laterale). Il principio di base sta nel rilevare come viene riflessa la luce del laser per capire le particolarità della superficie. Tale tecnica ci permette di analizzare com'è fatta la superficie dell'oggetto. Una della applicazioni fondamentali è l'analisi del legno per rilevare i punti in cui sono presenti dei nodi. La potenzialità di questa tecnologia si può apprezzare nelle seguenti illustrazioni. Nell'immagine di sinistra si ha una normale scansione ottica di una tavola in legno mentre in quella di destra quella con l'utilizzo dello scatter. Si vede subito come i nodi del legno vengono evidenziati con delle tonalità molto scure mentre le venature molto chiare, questo è dovuto alla differente riflessione della luce.



(a) Scatter 1. (b) Scatter 2.

Figura 1.29

Informazioni generali

A questo punto si sono viste le metodologie attraverso cui il sistema rileva la posizione del laser. Come già detto questo viene effettuato per ogni colonna lungo la larghezza del fascio laser. Rilevati tutti i punti del laser nelle colonne si ottiene un profilo dell'immagine. Questa procedura di analisi delle colonne viene effettuata per ogni profilo. In questo modo a seconda dei parametri iniziali di configurazione, tra cui la scelta del passo con cui rilevare un nuovo profilo, alla fine di una scansione si ottengono un certo numero di profili che descrivono l'oggetto scansionato.

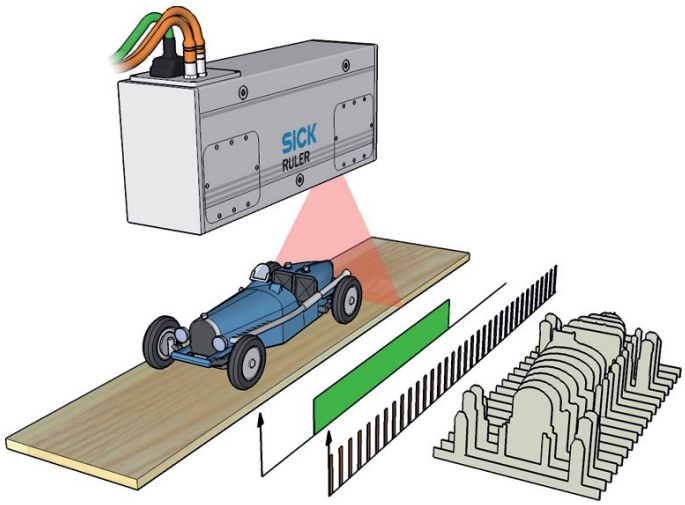


Figura 1.30: Profili rilevati

Per quanto riguarda il Ruler utilizzato da Euclid Labs sono commercializzate solo 3 dimensioni dell'apparecchio ovvero Ruler E 150, Ruler E 600 e Ruler E 1200. Questi utilizzano la stessa tecnologia ma hanno una campo di lavoro differente.

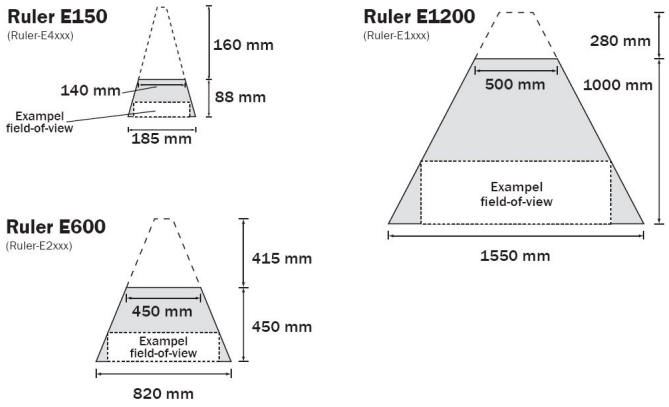


Figura 1.31: Tipi di Ruler

	Ruler E150	Ruler E600	Ruler E1200
Example FOV[1)]	50 x 150 mm	250 x 600 mm	250 x 1200 mm
Total height range	88 mm	450 mm	1000 mm
Stand-off	160 mm	415 mm	280 mm
Max. distance	248 mm	865 mm	1280 mm
Width at stand-off level	140 mm	450 mm	500 mm
Width at max. distance	185 mm	820 mm	1550 mm
Height resolution[2)]	0.05 mm	0.2 mm	0.4 mm
Max. profile rate[3)]	10 000 profiles/s	10 000 profiles/s	10 000 profiles/s

Figura 1.32: Tipi di Ruler

Infine si possono riassumere i benefici dell'utilizzo del Ruler Sick con questo elenco :

1. dati 3D ad alta velocità,
2. calibrazione in fabbrica,
3. facilità di installazione ed integrazione,
4. possibilità di combinazione tra dati di diversi Ruler,
5. libera scelta tra le routine di analisi delle immagini,
6. interfaccia standard, Gigabit Ethernet,
7. custodia robusta,
8. funzionamento garantito anche a basse temperature,
9. buon compromesso qualità/prezzo.

1.6 Euclid Labs

Figura 1.33: Euclid Labs

Allora stato attuale Euclid Labs lavora già con le apparecchiature Sick Ruler ed effettua già il bin-picking di oggetti messi in modo casuale in un pallet. Ma il lavoro effettuato in questo progetto, come detto in precedenza, lavora sul sistema di virtualizzazione creato da Euclid Labs. Questo mondo virtuale rispecchia esattamente la realtà e viene utilizzato per effettuare la simulazione dell'operato del robot. I vantaggi di questa strategia sono molteplici.

1. Possibilità di effettuare qualsiasi test senza dover arrestare la macchina,
2. Possibilità di effettuare qualsiasi test senza doversi recare presso il cliente,
3. Possibilità di testare collisioni tra robot ed ambiente esterno senza rischi.

Fase 1

La prima attività del progetto consiste nell'importare il file creato dal Ruler Sick nel mondo 3D Euclid Labs. Questa operazione non era mai stata effettuata dal personale dell'azienda ma ha portato nuovi vantaggi e risparmi di tempo. Sostanzialmente il principale vantaggio da questa operazione consiste nel poter visualizzare, all'interno del simulatore, l'oggetto desiderato senza averlo fisicamente. Questo aspetto può essere utile in tutti i progetti in quanto per effettuare i test dell'applicazione c'è bisogno di avere l'immagine che ha rilevato il Ruler. Inoltre poter analizzare in modo 3D l'immagine scansionata rende più facile la comprensione di essa ed ancor più la rilevazione di anomalie nell'acquisizione.

Fase 2

La fase successiva consiste nel simulare la scansione dell'oggetto direttamente all'interno del mondo 3D. Ovvero la creazione di un piano di lavoro dove si può collocare

un qualsiasi oggetto importabile dal simulatore ed una guida dove scorrono il laser e la telecamera. Lo scopo di tale algoritmo è di ottenere in uscita di esso un file contenente le stesse informazioni che si sarebbe ottenute effettuano la scansione dell'oggetto vero con il Ruler vero.
Anche questa attività ha numerosi vantaggi in quanto il cliente che commissiona il robot sicuramente è in possesso dei disegni tridimensionali degli oggetti con cui andremo a lavorare. Questi potranno essere in diversi formati tra cui CAD3D, stl, vrml, etc ma sicuramente non avranno la scansione del Ruler. Alla luce di ciò, fino a prima dei risultati di questo progetto, Euclid Labs era costretta a farsi spedire dei pezzi campione per fare delle scansioni con il Ruler oppure, se gli oggetti sono intrasportabili, recarsi direttamente dal cliente per acquisire l'immagine. Mentre, come già detto, alla conclusione del progetto è possibile farsi spedire dal cliente il modello tridimensionale dell'oggetto in uno dei svariati formati acquisibili dal simulatore. Posizionare l'oggetto importato nel piano di lavoro simulato del Ruler, lanciare la scansione ed in uscita otterremo lo stesso file che avremo ottenuto recandoci fisicamente dal cilente. Inoltre con tale approccio è possibile effettuare qualsiasi tipo di test, variando la posizione dell'oggetto o altre preferenze in qualsiasi momento.

Riassumendo i vantaggi che si suppone di avere alla fine del progetto:

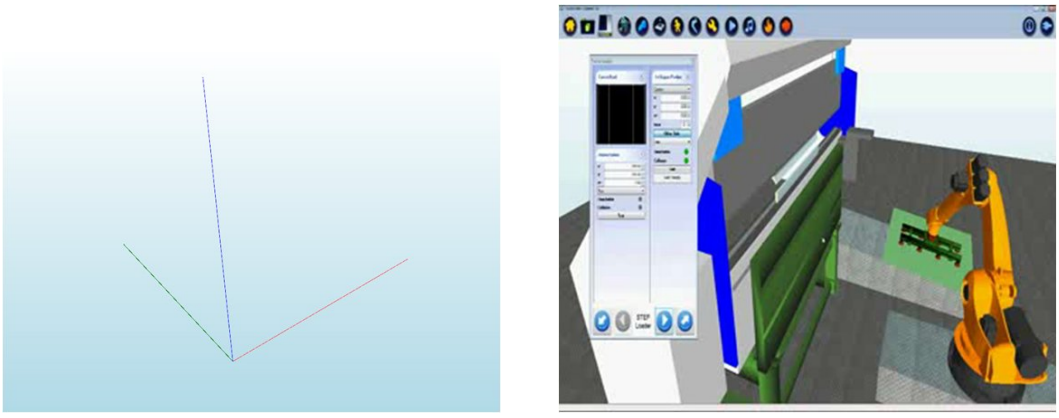
1. **Possibilità di visualizzare all'interno del simulatore l'immagine creata dal Ruler,**
2. **Avendo l'immagine all'interno del simulatore si possono identificare eventuali anomalie della scansione in modo molto agevole,**
3. **Possibilità di effettuare la scansione di un oggetto che non abbiamo fisicamente,**
4. **Evitare di doversi recare dal cliente o di farsi spedire l'oggetto desiderato,**
5. **Possibilità di fare un numero di test a piacere, variando specifiche dell'oggetto, a costo zero ed in qualsiasi momento,**
6. **Possibilità di effettuare qualunque test senza dover arrestare la macchina.**

2

Importazione del file Ruler

Obiettivo: importare nel simulatore 3D creato da Euclid Labs il file dato in uscita dal Ruler Sick

In questo capitolo si procederà con la spiegazione dell'importazione del file dato in uscita dal Ruler Sick nel mondo 3D di simulazione creato da Euclid Labs. Uno dei punti di forza dell'azienda consiste nell'aver sviluppato proprio questo software di simulazione 3D con il quale si possono eseguire i test di funzionamento dell'impianto robot desiderato.



(a) Mondo3D situazione base.

(b) Mondo3D con applicazione robot.

Figura 2.1

Per l'ideazione delle scelte effettuate lungo il capitolo sono stati letti numerosi paper tra cui : [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19]. In seguito saranno precisati i paper da cui sono stati evidenziati particolari interessanti.
Al giorno d'oggi sempre più impianti robot vengono ampliati con un sistema di visione 3D. Euclid Labs principalmente installa la camera 3D Ruler Sick.Infatti si è resa molto utile la possibilità di poter importare nel simulatore 3D l'immagine generata

dal Ruler.
Per poter effettuare questa operazione è stato fondamentale capire come è strutturato il file dato dal Ruler e come si deve costruire l'immagine nel mondo 3D. Per quanto riguarda il file di ingresso è costituito da numerose informazioni ma al fine del progetto è sufficiente la lista di punti descritti dalle coordinate (x, y, z), il numero di punti in ogni riga ed in ogni colonna. Mentre in uscita si dovrà creare una mesh con la struttura richiesta dal sistema ed importarla nel simulatore.

Figura 2.2: Passi da effettuare per l'Import

In particolare il costruttore della mesh utilizzato è il seguente :

$public\ Mesh(int\ numFaces,\ int\ numVertices,\ MeshFlags\ options,\ VertexFormats\ vertexFormat,\ Devicedevice)$;

Per la costruzione di tale struttura partendo dalla lista dei punti è richiesto un lavoro approfondito e suddiviso in più parti. Inoltre tale metodo deve essere progettato in modo da poter funzionare con qualsiasi dimensione dell'immagine, ovvero deve essere progettato in modo totalmente generale.

Nell'algoritmo, la chiamata al metodo è effettuata nel modo seguente.

$m\ =\ new\ Mesh(totfaces,\ totfaces\ *\ 3,\ mf,$
$CustomVertex.PositionNormalTextured.Format,\ device)$;

Come si è può vedere nella chiamata del metodo Mesh il formato dei vertici è del tipo CustomVertex.PositionNormalTextured.Format. Questo particolare formato, come si vede in seguito, richiede numerosi parametri :

$PositionNormalTextured(float\ xvalue,\ float\ yvalue,\ float\ zvalue,\ float\ nxvalue,$
$float\ nyvalue,\ float\ nzvalue,\ float\ u,\ float\ v)$.

1. xvalue : Posizione lungo l'asse x del punto
2. yvalue : Posizione lungo l'asse y del punto

3. zvalue : Posizione lungo l'asse z del punto
4. nxvalue : Componente x della normale del punto
5. nyvalue : Componente y della normale del punto
6. nzvalue : Componente z della normale del punto
7. u : Componente u della texture
8. z : Componente z della texture

2.1 Estrazione dati dal file 3db

Per arrivare al fine richiesto ci sono numerosi passi da effettuare. Il primo di tutti consiste nell'estrazione dal file 3db delle informazioni desiderate. Durante la progettazione del sistema è stata introdotta anche la possibilità di scegliere a priori se si desidera importare solo una parte dell'immagine. Invece di importarla per intero e poi visualizzare solo la parte di interesse, in questo caso, si è scelto di caricare solo la parte desiderata ottenendo un sistema più efficiente. I parametri che si possono scegliere sono i seguenti :

1. full image = false o true : con questo parametro si scegliere se si desidera importare l'immagine intera oppure no
2. Row_Limited : il numero di righe che si vogliono prendere in esame se non si usa il file intero
3. rowStart : indice della riga iniziale da cui acquisire
4. Col_Limited : il numero di colonne che si vogliono prendere in esame se non si usa il file intero
5. colStart : indice della colonna iniziale da cui acquisire

I dati estratti vengono salvati in una matrice data[], a cui ogni riga corrisponde un punto dell'immagine e nelle colonne abbiamo le componenti x, y e z che dal file vengono estratte con i rispettivi comandi $Read3db.x[i]$, $Read3db.id[rigaI]$, $Read3db.range[i]$. I valori della x e della z vengono chiamati con l'indice del punto mentre per la y rimane uguale per un'intera riga di lettura.
Essendoci la possibilità di poter scegliere il tipo di incremento del parametro riga I, si può anche effettuare un'acquisizione non ad ogni riga ma ad un determinato multiplo desiderato.
Mentre a seconda delle restrizione desiderate per il numero di colonne e di righe il sistema imposta la variabile i corretta. In uscita avremo una matrice contenente le coordinate di tutti i punti da visualizzare rispettanti i canoni desiderati.

2.2 Eliminazione dei punti buchi

Dall'analisi dei dati estratti si è avuto conferma che il Ruler Sick non acquisisce tutti i punti correttamente ma se riscontra qualche anomalia setta tale punto a $(0, 0, 0)$. A questo riguardo è stata inserita la possibilità di eliminare o non eliminare i buchi impostando una variabile *deletehole* a true oppure a false. La procedura di eliminazione è composta dai seguenti passi:

1. eliminare i buchi nella prima riga
2. eliminare i buchi nell'ultima colonna
3. eliminare i buchi nei punti rimanenti

Questa sequenza è indispensabile per com'è stato ideato il sistema di eliminazione. Se la prima riga e l'ultima colonna non sono prive di buchi l'algoritmo non porterà il risultato finale desiderato.
In particolare per l'eliminazione dei buchi nel corpo centrale si segue questa logica. Ovvero si sovrascrivono i punti buchi con la media dei valori dei punti vicini.

1. Scorrere i punti dal primo verso l'ultimo fino a quando si trova un buco,
2. Conteggiare se anche i punti successivi sono dei buchi incrementando nFind,
3. Quando si trova il primo punto corretto si modificano i valori dei buchi trovati in questo modo:
4. a) preX = Valore della x del punto precedente al buco,
5. b) postX = Valore della x del punto precedente al buco,
6. c) incrX = (postX - preX) / (nFind + 1),
7. d) preZ = Valore della z del punto precedente al buco,
8. e) postZ = Valore della z del punto precedente al buco,
9. f) incrZ = (postZ - preZ) / (nFind + 1),
10. g) per ogni punto dove è stato trovato il buco cioè for (int k = (i - nFind); k ¡ i; k++),
11. g1) data[k, 0] = data[k - 1, 0] + incr,
12. g2) data[k, 2] = data[k - 1, 2] + incrZ..

Eliminati i buchi presenti nei dati originali si è in possesso di una lista di punti, tutti significativi, perciò si procede con la creazione della mesh.

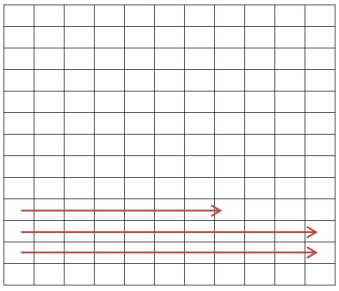


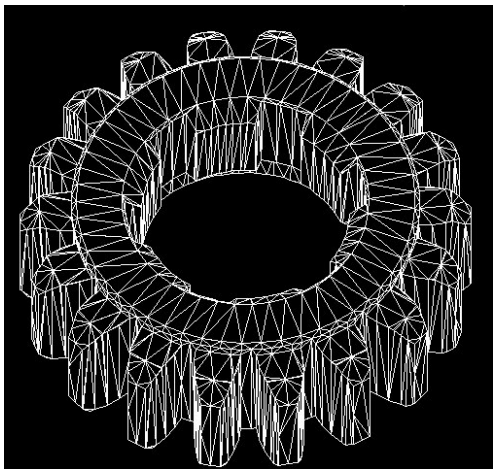
Figura 2.3: Scorrimento griglia

2.3 Creazione Mesh

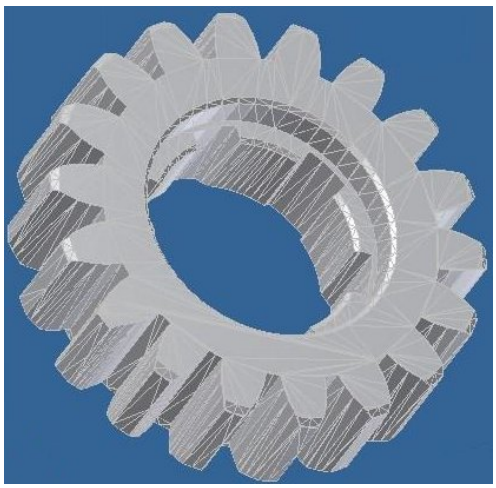
Ora si procede con la creazione della Mesh utilizzando la matrice dei punti estratti $data[]$ ed il numero di colonne Col_Mesh impostato a seconda delle preferenze scelte in precedenza. La chiamata che ritorna la Mesh è la seguente :

$Mesh\ m = CreateMeshFromData3db(Col\_Mesh, data, device, scale, out\ adjacency, out\ materials\_def, out\ materials, out\ textures)$;

All'interno di questo metodo viene creata la struttura interna della Mesh. Il primo passo effettuato è la creazione del $CustomVertex.PositionNormalTextured[]$ $out\_vertexbuffer$. Per effettuare questa operazione bisogna capire a pieno com'è strutturato l'immagine. Ovvero si è in possesso della lista di punti che identificano l'immagine ma in realtà questa può essere visualizzata come una sequenza di tantissime facce affiancate, nel nostro caso dei triangoli.



(a) Struttura STL.



(b) Visualizzazione STL.

Figura 2.4

Nel nostro caso si è agevolati dato che il numero di punti per riga e per colonna sono fissi, perciò si lavora con una sorta di griglia. La prima cosa da effettuare è l'indicizzazione generica di tutti i punti, ed effettuato ciò si può passare alle facce. Ragionando con facce triangolari, va considerato che ogni tre punti vicini vi sarà un triangolo. A noi in particolare interessa la normale di queste facce. Perciò è stato creato un metodo che dato in ingresso la lista dei vertici *data*[], ed il numero di righe e colonne, indicizza tutti i punti in modo da formare i triangoli e ne calcola la relativa normale.

Per popolare la Mesh non sono necessarie le normali delle facce triangolari ma le normali dei vertici. La normale di un vertice può essere ottenuta semplicemente grazie alla normale delle facce che hanno come uno dei tre vertici quello in esame. Anche questa operazione va effettuato con attenzione in quanto vanno indicizzati tutti i vertici e le facce in modo generale. Questo va effettuato in quanto non siamo a conoscenza delle dimensioni dell'immagine con cui andremo a lavorare. In particolare tale operazione va divisa in base alla posizione dei vertici perché oltre a variare l'indice delle facce vicine da prendere in considerazione varia anche il numero delle facce.

Di seguito sono elencate le generalizzazioni scoperte durante la progettazione.

1. Il primo vertice in basso a sinistra ha la normale della prima faccia,
2. I vertici centrali della prima riga usano le 3 facce di cui fanno parte,
3. L'ultimo vertice della prima riga usa le ultime due facce della prima riga,
4. I vertici della prima colonna usano le due facce di cui fanno parte della riga sotto e una faccia della riga sopra,
5. I vertici dell'ultima colonna usano una faccia della riga sotto e due di quella sopra,
6. I vertici centrali al corpo usano le 6 facce di cui fanno parte,
7. Il primo elemento dell'ultima riga usa le prime due facce dell'ultima riga,
8. I vertici centrali dell'ultima riga usano le tre facce di cui fanno parte,
9. L'ultimo vertice in alto a destra ha la normale dell'ultima faccia.

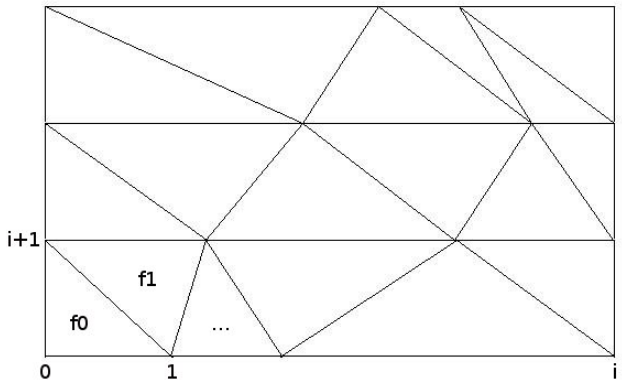


Figura 2.5: Indicizzazione Facce

Come si è visto i casi e le particolarità sono numerose ma è essenziale che ciò sia stato effettuato in questo modo per ottenere la lista dei vertici con le relative normali. Ovviamente sono stati creati dei metodi per il calcolo della normale di un vertice partendo dalla normale di 2 facce, 3 facce oppure 6 facce vicine, ovvero i casi che si sono riscontrati necessari. Ora si può procedere con l'inserimento dei vertici nel $Vertex\_Buffer$ nel modo seguente :

$vertexbuffer.Add(new\ CustomVertex.PositionNormalTextured((float)vertices[i].X * scale,\ (float)vertices[i].Y * scale,\ (float)vertices[i].Z * scale,\ (float)normal\_vertex[i].X,\ (float)normal\_vertex[i].Y,\ (float)normal\_vertex[i].Z,\ 0,\ 0));$

Popolato il $Vertex.Buffer$ va creato l'$I'intex\ Buffer$. L'$I'intex\ Buffer$ consiste nell'indicizzazione di tutti i vertici a tre a tre per formare i triangoli che visualizzaranno l'immagine finale. In particolare vanno salvati gli indici in questo modo :

1. out_indexbuffer[0] : indice 1 della faccia 1
2. out_indexbuffer[1] : indice 2 della faccia 1
3. out_indexbuffer[2] : indice 3 della faccia 1
4. out_indexbuffer[3] : indice 1 della faccia 2

5. out_indexbuffer[4] : indice 2 della faccia 2
6. out_indexbuffer[5] : indice 3 della faccia 2
7. out_indexbuffer[...] : ...
8. out_indexbuffer[(nRow * nFaceRow * 3)-1] : indice 3 della faccia nFaceTot

Una cosa a cui porre attenzione, come appena visto, è di salvare gli indici delle facce il ordine antiorario, in questo modo il sistema interpreta la normale verso l'altro e di conseguenza che è una faccia esterna dell'oggetto.

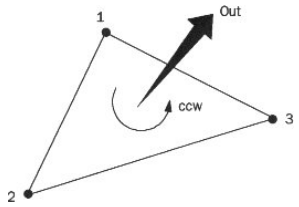


Figura 2.6: Orientazione Facce

Anche in questo caso, come per la creazione del $Vertex\_Buffer$ è stato creato tutto in modo generico e suddiviso per posizione delle facce. Non verranno precisati i dettagli dato che il concetto è simile alla creazione del $Vertex\_Buffer$. Creato il $Vertex\_Buffer$ ed l'$Index_Buffer$ rimane da costruire l' $out\_materialbuffer$ ma nel nostro caso non è di particolare interesse perciò lo lasciamo generico.

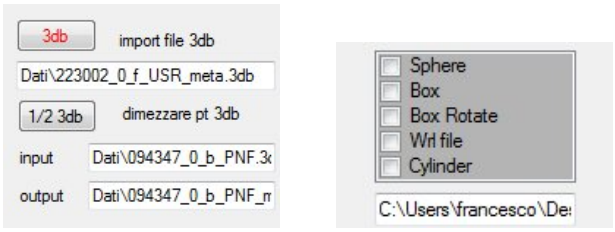
Qui di seguito è stata inserita inserita l'ultima parte di codice per creare e popolare la Mesh dopo aver elaborato tutto quello che è stato descritto in precedenza.

Listato 2.1: Caricamento Mesh

```
%=============================================================================
AttributeRange[] attributerange = CreateAttributeRange(indexbuffer);
MeshFlags mf;
mf = MeshFlags.Managed | MeshFlags.Use32Bit;
int totfaces = indexbuffer.Length / 3;
m = new Mesh(totfaces, totfaces * 3, mf, CustomVertex.PositionNormalTextured.
    Format, device);
m.SetVertexBufferData(vertexbuffer, LockFlags.None);
m.SetIndexBufferData(indexbuffer, LockFlags.None);
m.SetAttributeTable(attributerange);
%=============================================================================
```

2.4 Applicazione

Tutte le funzionalità descritte fino ad ora sono state inglobate in un pulsante nell'applicazione finale, pulsante 3db. Oltre a questa funzionalità principale sono state inserite nel pannello dei comandi anche ulteriori funzionalità.



(a) comandi 1. (b) comandi 2.

Figura 2.7

Come si può notare c'è il pulsante 3db che importa il file 3db corrispondente all'indirizzo indicato nello spazio sottostante. Mentre il pulsante 1/2 3db ci permette di dimezzare il numero di punti presenti in un file indicato nella riga input che viene salvato nel file 3db indicato nella riga output.

Mentre la seconda parte dei comandi ci permette di importare alcune figure di base, sfera, cubo, cubo ruotato,cilindro oppure è stata inserita la possibilità di importare un altro tipo di file, vrml, inserendo l'indirizzo nello spazio sottostante.

2.5 Esempi

Con ciò si può ritenere sufficiente la spiegazione delle parti cruciali del caricamento di un file 3db, della successiva elaborazione di esso per creare la Mesh e la visualizzazione dell'immagine nel mondo 3D creato da Euclid Labs. Qui di seguito sono stai inseriti degli esempi per attestare la buona riuscita di quanto detto fino ad ora.

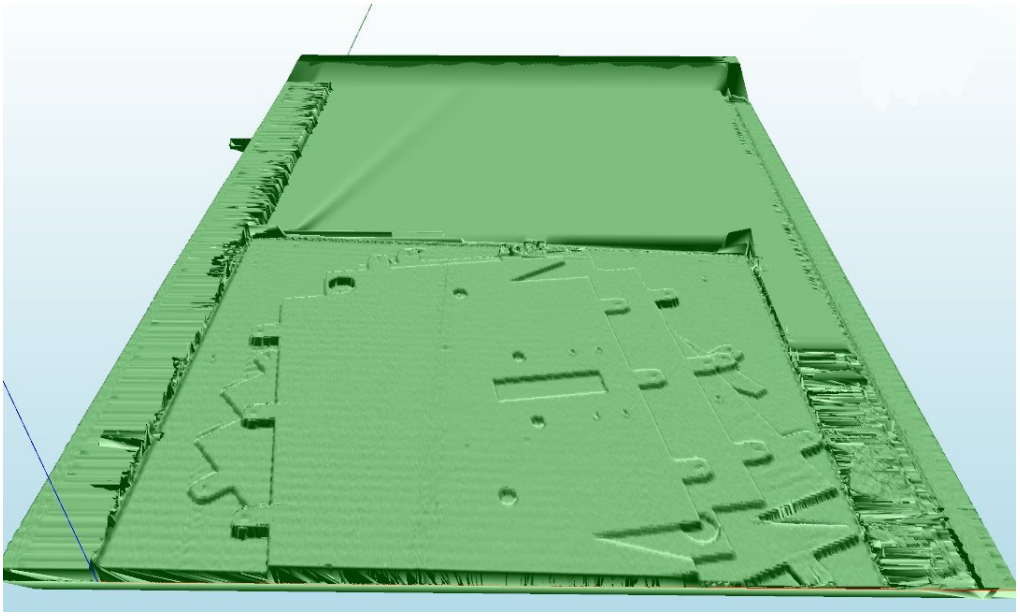


Figura 2.8: Esempio 1, Import totale di un file

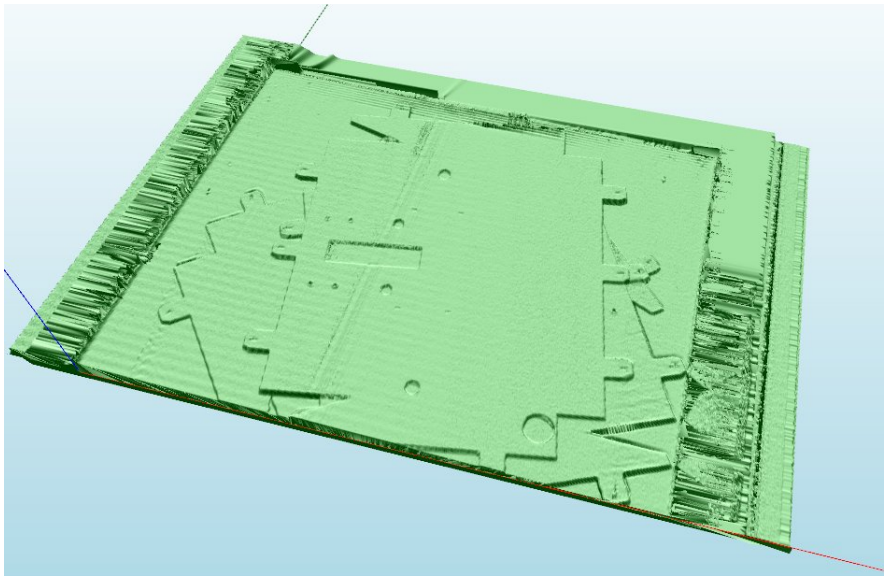


Figura 2.9: Esempio 1, Import limitato nelle righe

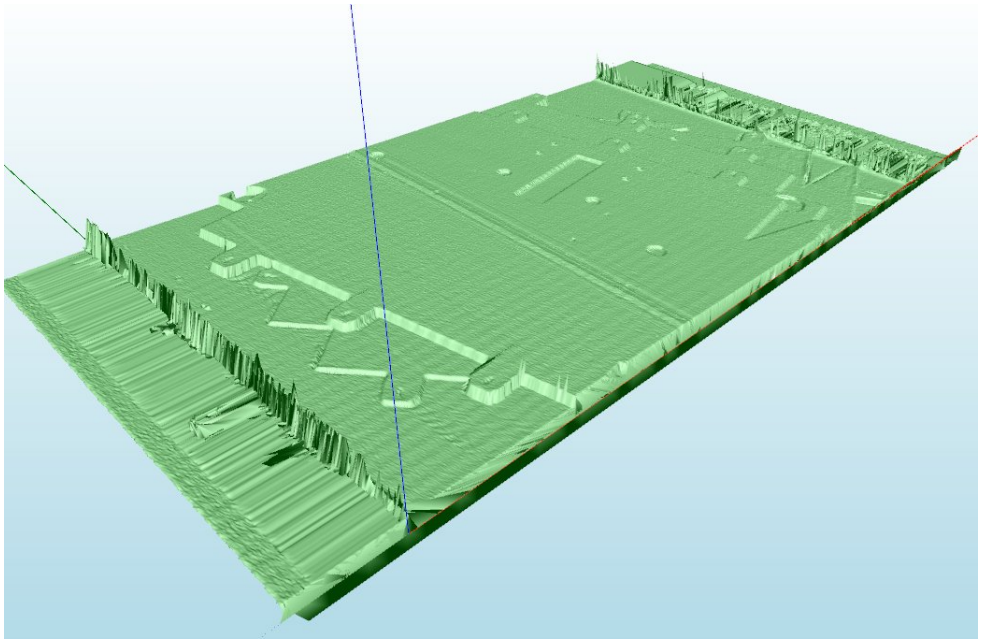


Figura 2.10: Esempio 1, Import limitato nelle righe

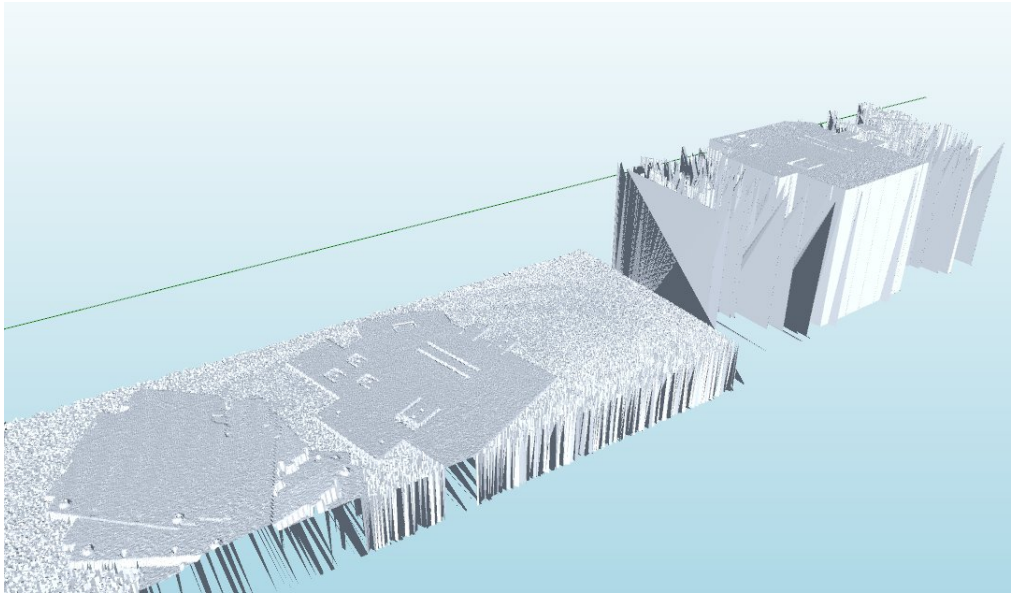


Figura 2.11: Esempio 2, Import limitato nelle colonne

Questa nuova possibilità di visualizzare in modo 3D le immagini scannerizzate dal Ruler Sick ci può permettere anche di rilevare con maggiore semplicità eventuali anomalie. Ovvero si potrebbero visualizzare delle particolarità che permettono di

identificare la causa di un problema. Per esempio, come visualizzato in seguito, si possono vedere delle fasce non regolari simili ad un effetto fontana lungo il bordo del piano dove sono appoggiati gli oggetti. Questo è dovuto al fondo del piano fatto di un materiale riflettente. Di conseguenza la camera rileva i punti lungo il bordo in modo anomalo. Mentre come si può apprezzare dall'immagine l'oggetto in esame è stato comunque rilevato in modo eccellente.

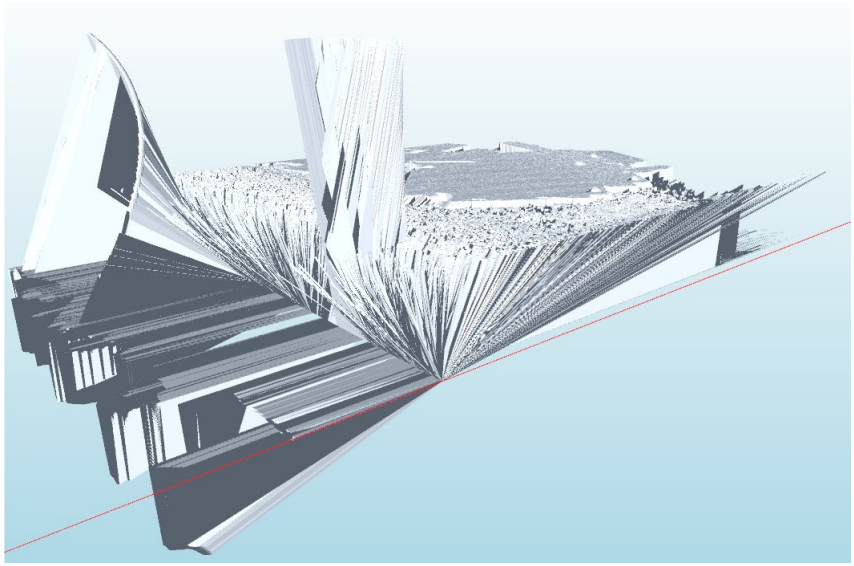


Figura 2.12: Esempio 3, Import con problema base riflettente

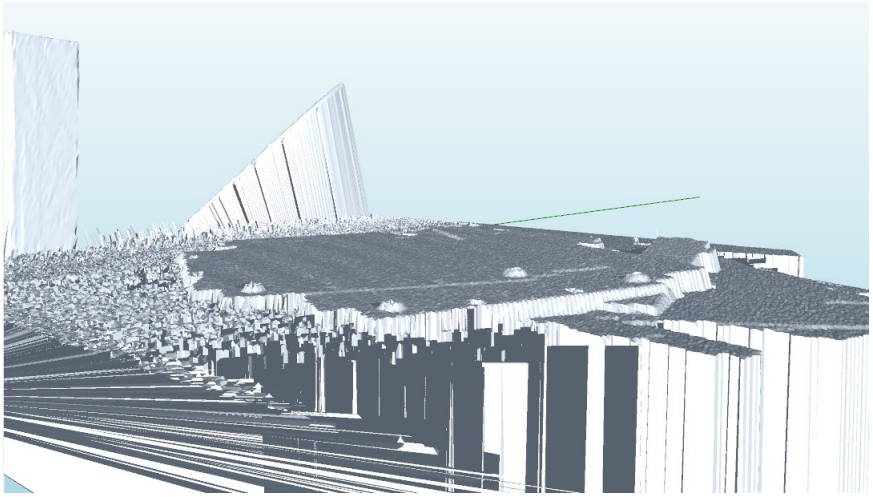


Figura 2.13: Esempio 3, Import oggetto rilevato correttamente

3

Simulazione Ruler Sick

Obiettivo: simulare il comportamento del Ruler Sick.

Riassumendo lo stato attuale, a questo punto, grazie al lavoro eseguito fino ad ora si è in grado di importare nel mondo 3D un file di tipo 3db costruendo l'immagine con una mesh. Il passo successivo del progetto consiste nel simulare il comportamento del Ruler all'interno del mondo 3D. Cioè di creare un'applicazione che permetta di effettuare una scansione simulata dell'oggetto ottenendo gli stessi risultati che avremo ottenuto con l'utilizzo del vero Ruler e del vero oggetto da analizzare.

3.1 Ambiente di simulazione Ruler

Come appena detto il fine di questa parte del progetto è quella di simulare il comportamento del Ruler. Perciò la prima cosa da fare è di creare un ambiente di lavoro che sia simile a quello reale. Ciò è stato effettuato creando diversi oggetti all'interno del simulatore e dando la possibilità di personalizzare alcuni parametri. Per quanto riguarda gli aspetti visibili nell'ambiente :

1. E' stato creato un piano di lavoro dove viene situato l'oggetto in esame, questo è il livello entro il quale successivamente il laser e la camera effettueranno la scansione,
2. E' stata creata una guida dove scorreranno il laser e la camera,
3. Sono stati creati degli oggetti che rappresentano la camera ed il laser,
4. E' stata creata una linea rossa lungo il piano di lavoro che identifica la posizione in cui il laser sta proiettando il fascio e la camera sta eseguendo la scansione,
5. E' stato creato un effetto di animazione di scorrimento del laser, camera e linea laser man mano che vengono eseguiti la scansione ed i calcoli,

6. Lungo la guida di scorrimento vengono stampati dei marcatori gialli nel punto esatto in cui avviene la scansione.

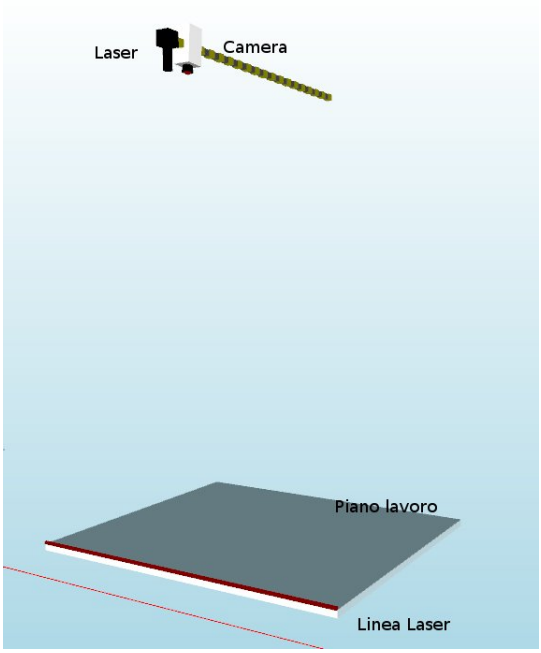
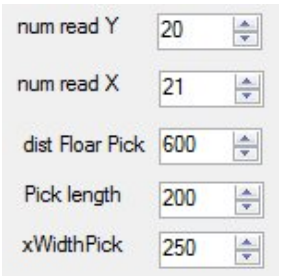


Figura 3.1: Situazione base

Mentre per quanto riguarda i parametri che si possono scegliere a seconda delle esigenze del caso sono state create le seguenti personalizzazioni :



(a) Parametri 1.

(b) Parametri 2.

Figura 3.2

1. num read Y : numero di letture lungo l'asse Y, cioè la direzione dello scorrimento,

2. num read X : numero di letture lungo l'asse X, cioè il ventaglio del laser,

3. dist Floar Pick : distanza tra il piano di lavoro e la guida dove lavorano laser e camera,
4. Pick length : lunghezza del piano di lavoro,
5. xWidthPick : larghezza del piano di lavoro,
6. dist Laser Camera : distanza che intercorre tra la posizione della Camera e del Laser.

Riguardo la distanza tra il laser e la camera è preferibile dare qualche precisazione in quanto presenta degli aspetti particolari. Variando tale parametro si varia la geometria dell'acquisizione ovvero dato che il laser proietta un fascio e la camera rileva dove questo colpisce l'oggetto, a seconda della posizione varia l'angolo tra i due. Il variare di quest'angolo principalmente porta alla variazione del numero di punti oscuri dell'oggetto. Ciò dipende anche dalla conformazione dell'oggetto, ma nella gran parte dei casi, maggiore è l'angolo tra i due, maggiore sarà la quantità di punti nascosti.

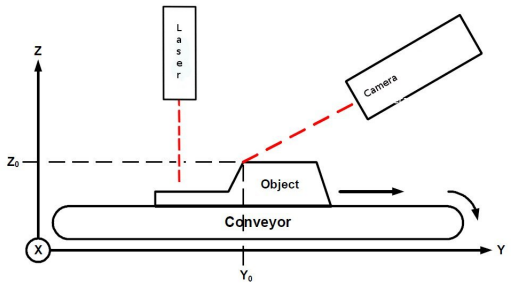


Figura 3.3: Con punto nascosto

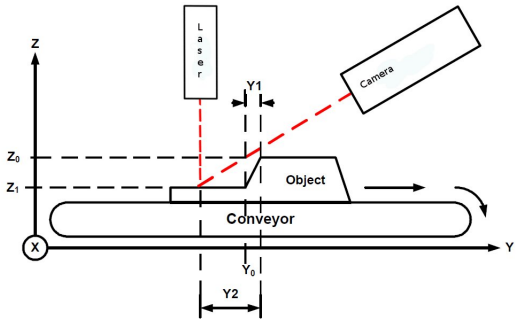


Figura 3.4: Punto visibile

Progettato tutto ciò che riguarda l'ambiente di lavoro e le varie possibilità di personalizzazione di questo si passa alla vera e propria spiegazione della scansione dell'oggetto.

3.2 Pick

Essendo a conoscenza di tutti i parametri della scansione si è a conoscenza della dimensione di una riga e di quante letture vanno effettute lungo ad essa. Perciò si può ricavare la distanza tra una letture e l'altra. Allo stesso modo si può ragionare anche per le colonne, cioè siamo a conoscenza di quante letture andremo ad effetture lungo la scansione e la lunghezza del piano perciò possiamo ricavare la distanza tra un profilo ed il successivo.

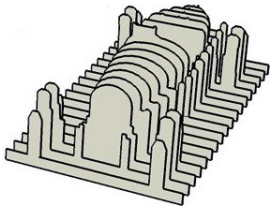


Figura 3.5: Esempio scansione profili

Avendo queste informazioni abbiamo una griglia composta da punti in una precisa posizione. Ogni punto corrisponde alla direzione verso cui verrà effettuata la singola lettura. Infatti per ognuno di questi punti viene effettuata la seguente chiamata a metodo :

$distPick = doPick\_optimized(''oggetto\ in\ esame'', xPickStart, yPosPick, zPickStart, xDirPick, yDirPick, zDirPick)$;

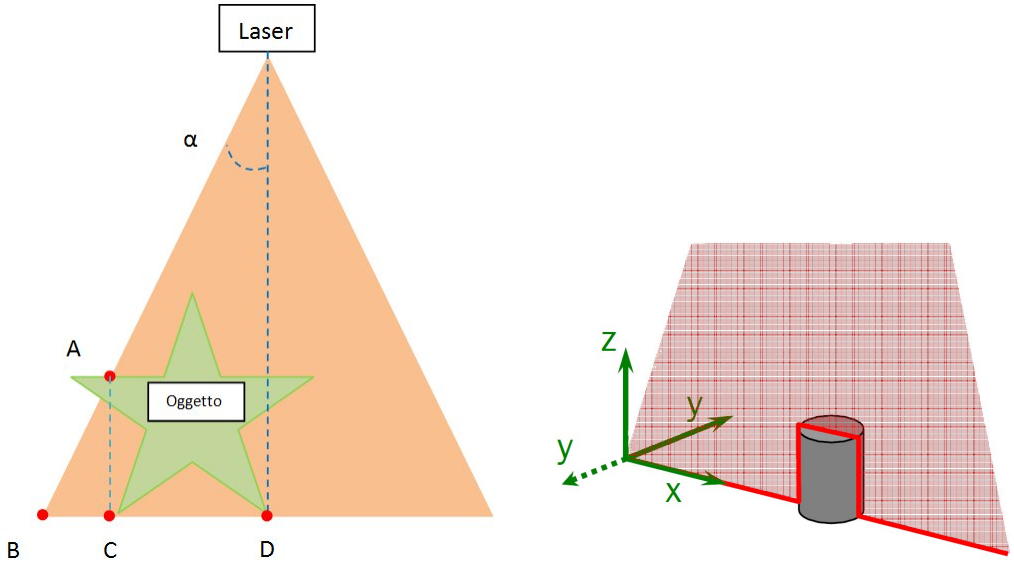
Il metodo richiede di conoscere : l'oggetto in esame nel formato $EuclidLabs.World.el3DMeshObjec$, le tre coordiante che rappresentano la posizione del laser e tre per la direzione del laser. A questo punto chiamando il metodo :

$s.Pick(rayPos, rayDir, outdist)$;

Con rayPos e rayDir costituiti da due dati di tipo Vector3D restituisce la distanza che intercorre tra il laser e il punto in cui il raggio del laser interseca l'oggetto in esame. Mentre se il raggio dovesse arrivare al livello del piano senza intersecare l'oggetto il metodo restituisce un valore di distanza -1.

A questo punto si è in possesso della distanza tra laser ed oggetto ma non le coordinate del punto, perciò entra in gioco un po' di trigonometria. Osservando la successiva immagine si comprende subito la motivazione di questo problema. Ciò si verifica in quanto è stata data una posizione ed una direzione per eseguire il pick ma l'intersezione con l'oggetto è avvenuta prima di arrivare al livello del piano di lavoro. Perciò vanno calcolate coordinate x e z dell'intersezione mentre la coordinate y non

ha subito variazioni dato che è costante lungo tutto il profilo. Per comprendere al meglio la problematica osservare la seguente illustrazione può essere di aiuto.



(a) Pick Laser 1. (b) Pick Laser 2.

Figura 3.6

Come già detto in precedenza, analizzando il caso generale del pick, esso viene effettuato con la direzione che parte dal Laser e va verso il punto B. In questo modo La distanza tra il punto B ed il punto D rispecchia la suddivisione equidistante lungo la larghezza del profilo. Ma dato che il Pick ha intersecato l'oggetto nel punto A va calcolata la reale coordinata x del punto di intersezione. Si consideri che la distanza rilevata tra il Laser e l'intersezione con l'oggetto cioè il punto A è stata nominata con distPick e ci viene fornita come risultato della chiamata del metodo Pick. Mentre la realX è la lunghezza del segmento $\overline{CD}$.

$$realX = distPick * sen(\alpha) \tag{3.1}$$

Ora si può calcolare la reale coordinata z.

$$zPick = zPickStart - (\sqrt{(distPick^2 - realX^2)}); \tag{3.2}$$

Ottenuti tali risultati si possono salvare le coordinate reali del punto trovato.

Listato 3.1: Salvataggio coordinate punto

```
MatPick[r_MatPick, 0] = xPickStart + realXPick;
MatPick[r_MatPick, 1] = yPosPick;
MatPick[r_MatPick, 2] = zPick;
r_MatPick++;
```

In questo modo si sono ottenute le coordinate di tutti i punti in cui si è rilevata l'intersezione del laser con l'oggetto ed i punti in cui è giunti fino al piano di lavoro.

Durante la fase di sviluppo è stata ideata ed inserita la possibilità di stampare tutti i punti trovati. Ma se questo non è desiderato il sistema procede con la fase successiva.

3.3 Punti visibili dalla camera

In questa nuova fase si effettua il test dei punti che sono realmente visibili dalla camera e quelli che sono nascosti. Il concetto è stato chiarito in precedenza ma ora si procederà ad illustrare la strategia adottata per raggiungere questo obiettivo.

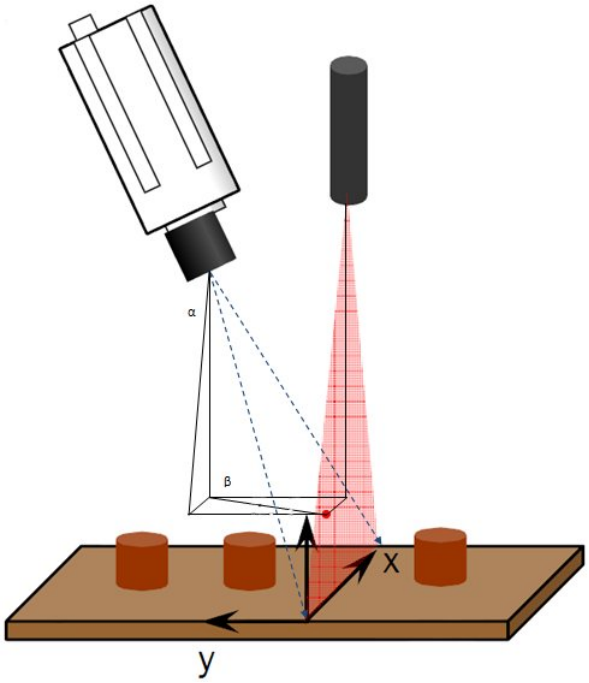


Figura 3.7: Punti visibili

Essendo a conoscenza delle coordinate x,y,z del Laser, della Camera e del Punto in esame si possono calcolare gli angoli α e β. In base a ciò si può procedere con il calcolo delle direzioni del segmento che congiunge la Camera con il punto in esame. Calcolate $xDirPick$, $yDirPick$, $zDirPick$ si hanno i parametri necessari per effettuare il calcolo successivo, ovvero si esegue un'altra operazione di Pick dell'oggetto ma con questa chiamata.

$distPick = doPick\_optimized(''oggetto\ in\ esame'', (int)Camera.X, (int)Camera.Y, (int)Camera.Z, xDirPick, yDirPick, zDirPick);$

Grazie a tale istruzione si ottiene la distanza che intercorre tra la camera ed il punto trovato. Nominiamo questo valore come $Distanza\ effettiva\ tra\ punto\ e\ camera$.

Quanto trovato fino ad ora non è sufficiente per raggiungere lo scopo desiderato. E' necessario calcolare anche la distanza teorica tra la Camera ed il Punto utilizzando i valori trovati precedentemente.

1. Pt_Cam_X : distanza tra il Punto e la Camera lungo l'asse x, MatPick[i,0] - Camera.X

2. Pt_Cam_Y : distanza tra il Punto e la Camera lungo l'asse y, MatPick[i,1]- Camera.Y

3. Pt_Cam_Z : distanza tra il Punto e la Camera lungo l'asse z, Camera.Z - MatPick[i, 2]

4. tolerance : impostata dall'utente

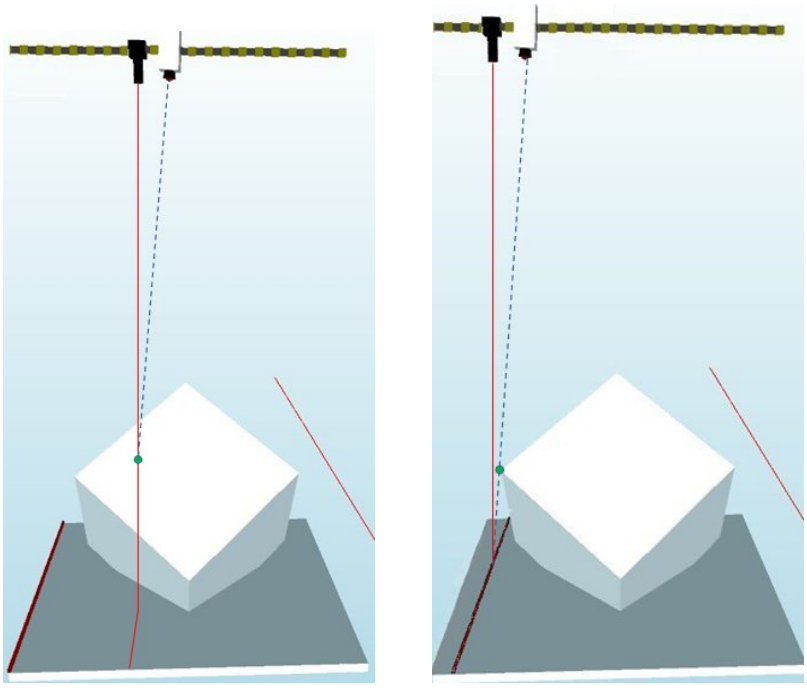
Fissati tali parametri si può procedere con la distanza teorica Punto / Camera.

$$dist\_Teo = \sqrt{(Pt\_Cam\_X^2 + Pt\_Cam\_Y^2 + Pt\_Cam\_Z^2)} \tag{3.3}$$

Tutto ciò viene effettuato perché se la $dist\_Teo$ e la $Distanza\ effettiva\ tra\ punto\ e\ camera$ sono equivalenti salvo una certa tolerance fissata, il punto è visibile mentre se non lo sono il punto è oscurato.

La motivazione di ciò si può spiegare nel seguente modo. Come detto in precedenza quando si calcola la distanza tra laser ed oggetto si richiama la funzione Pick che restituisce la prima intersezione che riscontra con l'oggetto. La stessa cosa viene effettuata anche con la Camera perciò se la camera trova come prima intersezione la stessa che aveva trovato il Laser allora il punto è visibile dalla camera. Nel caso contrario significa che la funzione Pick della Camera trova un'intersezione con l'oggetto prima di arrivare al punto desiderato. Perciò il punto identificato dal Laser e dalla Camera non coincidono.

Come si può apprezzare nella seguente coppia di figure si nota che in quella a sinistra il punto evidenziato dal Laser e dalla Camera coincidono di conseguenza il punto è visibile. Mentre nella figura di destra il Laser va dritto fino al piano mentre la Camera interseca l'oggetto, questa è la situazione in cui il punto risulta essere oscurato.



(a) Pick punto visibile. (b) Pick punto nascosto.

Figura 3.8

Questa procedura di controllo viene effettuata per tutti i punti identificati dal Pick del Laser. Giunti a questo punto si è in possesso di tutti i punti visibili dalla camera con le relative coordinate (x_i,y_i,z_i). Queste informazioni sono salvate in una matrice matrix_visible[numero punti, 3].

Effettuando un'osservazione, si constata che tali punti rappresentano lungo gli assi x ed y una griglia completa. Cioè non vi è un incremento costante nell'asse x ma comunque rimane costante il numero di punti in ambe due le direzioni. Mentre nella direzione z i valori dipendono dalla conformazione dell'oggetto analizzato. Pertanto tale struttura è costruita nello stesso modo del file 3db che si aveva in ingresso nel capitolo precedente. Dato che la procedura per l'import del 3db è stata creata in modo generico in modo da poter visualizzare correttamente le immagini di qualunque dimensione, cioè numero di punti lungo l'asse x ed y generici. E' stata inserita la possibilità di visualizzare all'interno del simulatore la collezione dei punti visibili trovati sfruttando le funzionalità dell'Import3db creato in precedenza.

Per attestare il corretto funzionamento, di seguito è stato inserito un esempio di scansione con le relative annotazioni. Inizialmente, nella situazione di base, erano presenti il piano di lavoro, la camera, il laser e la guida di scorrimento mentre ora sono visibili anche i seguenti elementi.

1. Sfera rossa che rappresenta un qualunque oggetto che può essere posto sul piano di lavoro
2. La nuvola di punti visibili ed appartenenti alla sfera
3. L'oggetto ricostruito grazie alle funzionalità dell'Import3db

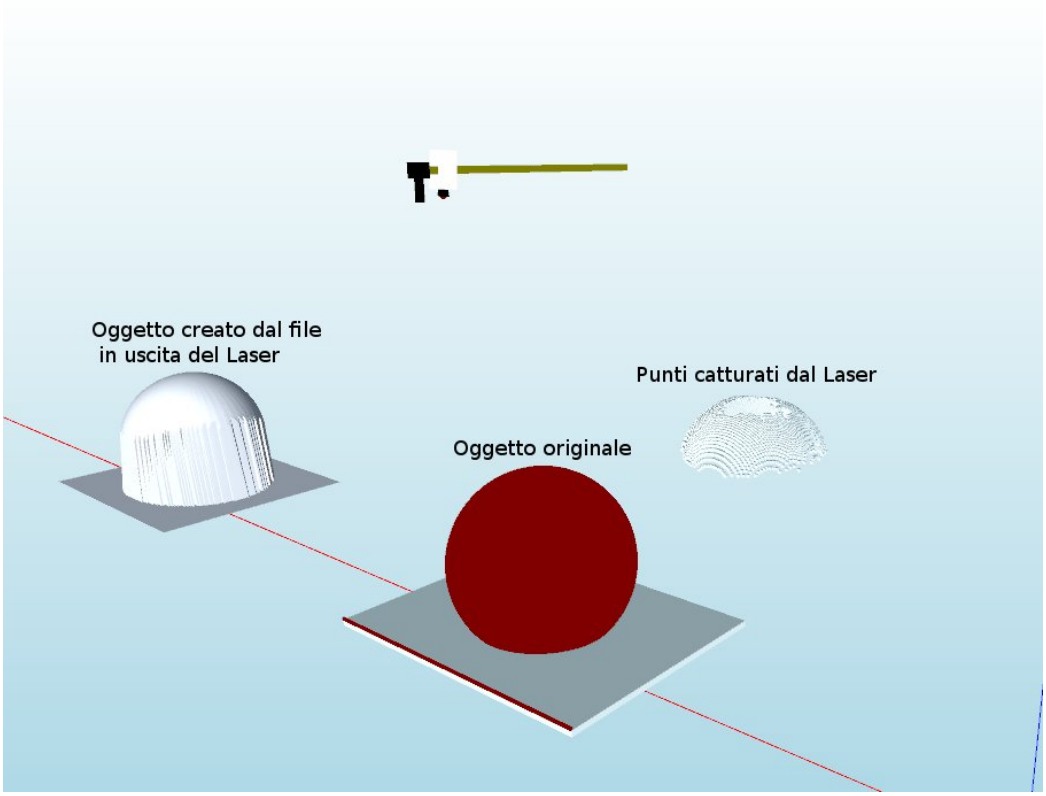


Figura 3.9: Esempio esecuzione scansione simulata

4

Estrazione Features

Obiettivo: estrazione delle features dalla cloud of points.

4.1 Identificazione dei bordi

Il lavoro svolto nei capitoli precedenti ci permette di ottenere una nuvola di punti che descrive la superficie visibile di un oggetto, osservata da una camera che scorre lungo un supporto. Tali punti sono solo quelli realmente visibili da essa e sono identificati dalle 3 coordinate (x_i,y_i,z_i).

Quello che verrà descritto di seguito consiste in uno dei punti fondamentali e complessi del progetto. Per effettuare tali scelte sono stati consultati differenti paper tra cui i seguenti : [35], [36], [37], [38]. In seguito verranno precisati eventuali punti cruciali che sono stati ricavati dai paper ma va precisato già da ora che non sono stati utilizzati metodi o algoritmi già esistenti ma ognuno dei paper ha dato un leggero contributo per creare la nuova idea finale. Oltre ai paper appena citati riguardanti tecniche di line detection sono stati analizzati anche molti altri riguardo svariate tecniche di fitting : [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54]

Partendo da queste informazioni si passa alla fase successiva cioè all'identificazione dei punti vicini ai bordi dell'oggetto. Per ricavare tale informazione è stato pensato di analizzare tutti i punti visibili ma più in particolare si è studiata la normale di ogni punto. L'idea di fondo di questo algoritmo è focalizzata in un unico concetto. I punti appartenenti ad un faccia mantengono la medesima normale mentre i punti vicino ad un bordo variano sempre più tale valore fino ad arrivare a quello della normale della faccia successiva. In altre parole i punti nell'intorno dei bordi sono quelli in cui si verifica un cambiamento di normale.

Per effettuare tale test in modo efficiente ed ottenendo risultati soddisfacenti si sono provate differenti tecniche. Ma la migliore che è stata trovata è la seguente.

Si scorre la lista di tutti i punti visibili e per ognuno di essi si esegue il successivo test della normale. Questo test, come i metodi precedentemente creati, è stato studiato per funzionare con qualunque dimensione di immagine. La sua prima funzione è quella di identificare gli 8 punti più vicini di quello in esame. Identificati questi calcola la differenza tra la normale del punto in esame e quella di ognuno di essi.

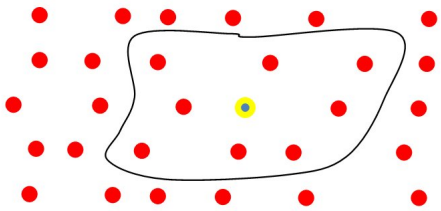
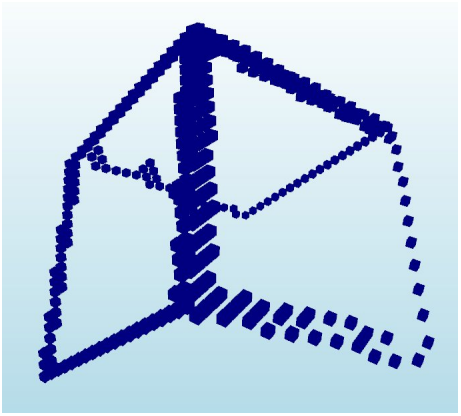


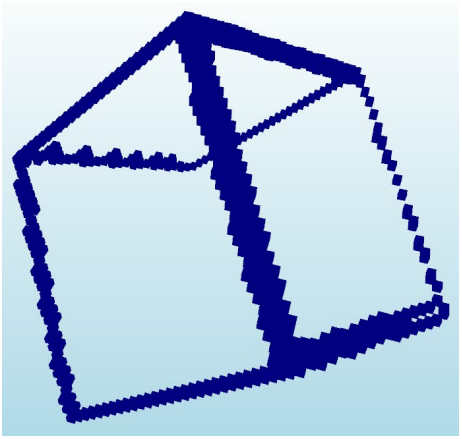
Figura 4.1: Otto punti in esame

Mentre viene calcolata la differenza ne viene anche identificata quella maggiore e viene restituita al metodo chiamante.
Terminata questa procedura si ottiene la lista dei punti con un'informazione in più, quale degli otto punti vicini ha la maggiore differenza di normale ed il suo valore. A questo punto è stata inserita una variabile di soglia personalizzabile che viene utilizzata come filtro. I punti che hanno un vicino con una normale la cui differenza è superiore della soglia sono identificati come punti vicino al bordo in caso contrario sono considerati punti appartenenti alla faccia.

Dai test effettuati si è notato che tale metodo funziona ma la precisione di esso varia a seconda della risoluzione dell'immagine. Ora si motiverà tale affermazione. Come descritto in precedenza questo metodo analizza la normale degli 8 punti vicini perciò se l'immagine ha un numero di scansioni per profilo piccolo i punti che verranno identificati come vicini al bordo copriranno un'area maggiore perciò il risultato sarà meno preciso. Mentre più si eleva in numero di scansioni più si rimpicciolisce l'area identificata vicino al bordo, ottenendo un risultato più accurato.

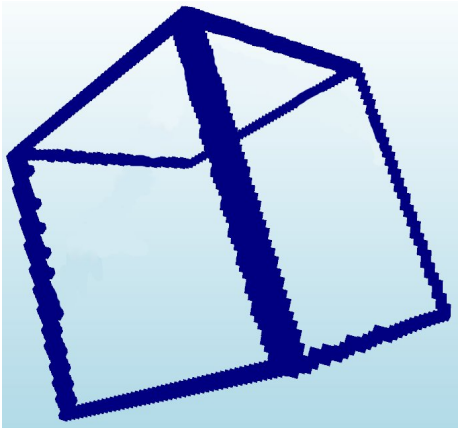


(a) scansione 80x80.

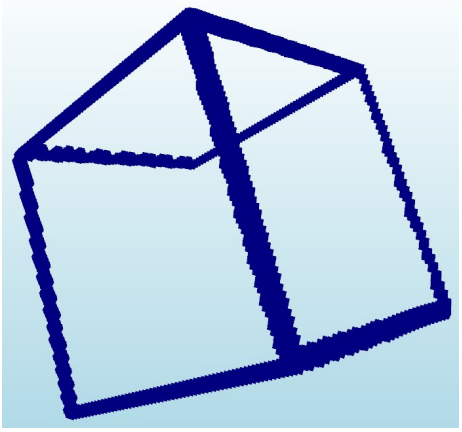


(b) scansione 90x90.

Figura 4.2



(a) scansione 150x150.



(b) scansione 180x180.

Figura 4.3

Riassumendo lo stato attuale, a questo punto, grazie al lavoro eseguito fino ad ora si è ottenuta una struttura dati con un contenuto molto particolare. Il contenuto di questa struttura consiste nei punti nell'immediato intorno dei bordi dell'oggetto in esame. Ognuno di questi punti è rappresentato da 3 dati che corrispondono alle sue coordinate tridimensionali nel piano cartesiano. La creazione di questi dati è stata illustrata nelle sezioni precedenti, mentre ora inizia una parte fondamentale dell'intero progetto. Come scritto nel titolo consiste nell'estrazione di features dalla nuvola di punti. In particolare l'obiettivo è di individuare gli spigoli, i vertici ed i cerchi. Per raggiungere tale obiettivo sono stati studiati molti lavori effettuati da altri ricercatori, ma alla fine di tale attività si è giunti alla conclusione che non ci

sono risultati concreti per questo lavoro. In particolare nel passato sono stati studiati molteplici casi in 2 dimensioni e con diverse tecniche. Ma per quando riguarda il 3D o ci si riconduce alla situazione del 2D oppure si utilizzano tecniche in cui l'idea principale è di muovere l'oggetto da cercare all'interno dell'immagine fino a quando si trova la posizione ideale. Ma questa non è la strada corretta al fine di questo progetto.

4.2 Scelta della strada corretta

In seguito alle ricerche effettuale si sono aperte diverse strada per la soluzione del problema. Ma come si sà, quando si esplorano nuovi terreni non è per niente semplice individuare quale sia la via che permette una soluzione corretta e nel tempo minore. Successivamente a numerosi studi e ragionamenti è stata scelta la seguente strategia. E' stato scelto di creare un sistema simile alla conosciutissima Hough Transform.

La trasformata di Hough è una tecnica utilizzata nell'elaborazione digitale delle immagini. I suoi creatori furono Richard Duda e Peter Hart nel 1972 che con tale scoperta permisero il riconoscimento delle linee di un'immagine 2D. Negli anni successivi Dana H. Ballard riprese in mano tale scoperta che ebbe un nuovo successo grazie alla sua pubblicazione dell'articolo Generalizzazione della trasformata di Hough per il riconoscimento di forme arbitrariamente definite.
Per lo studio e l'approfondimento della Hough Transform sono stati analizzati molti paper tra cui : [20], [21], [22], [23], [24], [25], [26], [27]. Ora vediamo in cosa consiste tale tecnica.

Partendo da quello che dobbiamo trovare, cioè l'equazione generale di una retta :

$$y = mx + c \tag{4.1}$$

Lo scopo della Hough Transform è quello di trovare i parametri (m,c) che identificano la retta in esame. Considerando il punto $(x_i + y_i)$ appartenente alla retta otteniamo queste formule :

$$y_i = mx_i + c \tag{4.2}$$

oppure

$$c = -x_i m + y_i \tag{4.3}$$

Con quest'immagine si riassume la situazione attuale della retta generica nello spazio 2D con un punto generico che vi appartiene.

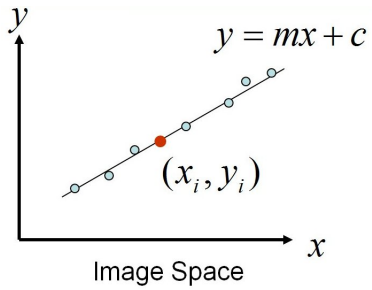


Figura 4.4: Retta generica in 2D

A questo punto è necessario introdurre l'Hough Space concetto fondamentale per capire la Hough Transform. Mentre nella rappresentazione classica della retta negli assi cartesiani si hanno le coordinate x ed y nell'Hough Space si devono mettere 2 variabili che permettono l'identificazione univoca della retta. Più precisamente in questo caso negli assi dell'Hough Space si hanno m e c.

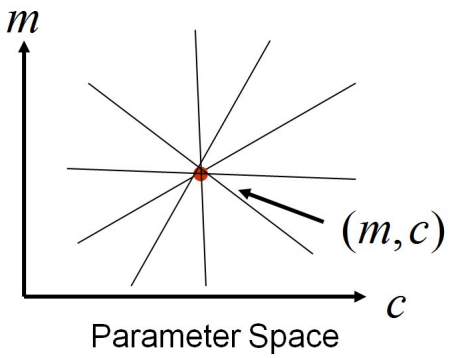


Figura 4.5: Hough Space

Dalla figura 4.5 si nota che l'intersezione di tutte le rette, nello Hough Space, rappresenta proprio i parametri m e c della retta desiderata.
Per entrare più nel particolare l'algoritmo per l'Hough Transform consiste in questi passi principali.

1. Quantizzazione dei parametri dello Space (m,c),
2. Creazione Array di accumulazione A(m,c),
3. Impostazione A(m,c) = 0 ∀m,c ,
4. ∀ m,c incrementare A(m,c) = A(m,c) + 1 se (m,c) si trova nella retta $c = -x_i m + y_i$,
5. Cercare il massimo in A(m,c).

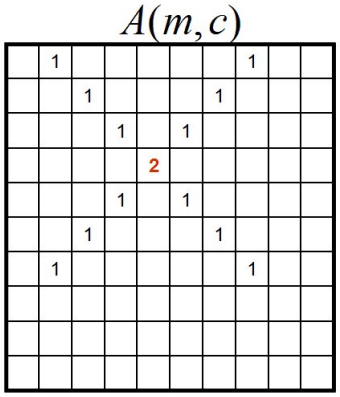


Figura 4.6: Hough Accumulazione

Come si può notare nella figura 4.6 man mano che l'algoritmo itera si accumula il valore massimo nella cella con indici m e c dell'equazione in esame. Il Passo successivo consiste nel considerare la Normal Parametrization di una linea, in particolare una retta generica può essere rappresentata con la seguente equazione, considerando u il vettore normale della retta l:

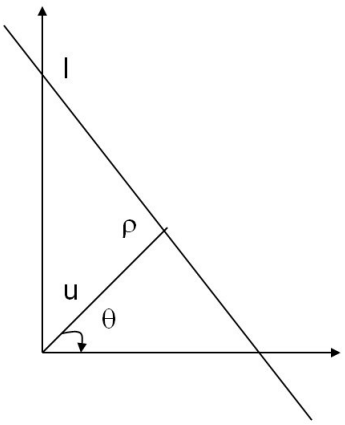


Figura 4.7: Normal Parametrization

$$x * cos\theta + y * sen\theta = \rho \tag{4.4}$$

Ragionando sulle 2 possibilità di esprimere una retta si può ragionare sul fatto che - $\infty \leq m \leq \infty$ mentre per $0 \leq \theta \leq 2\pi$ e $0 \leq \rho \leq \rho_{max}$.

Dati i punti (x_i,y_i) si devono cercare (ρ,θ) .

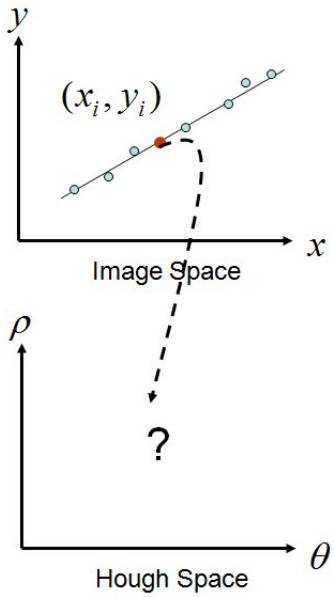


Figura 4.8: Hough con Normal Parametrization

Ora seguono degli esempi inerenti a questa tecnica. Il primo privo di rumore mentre il secondo con una presenza leggera di disturbo.

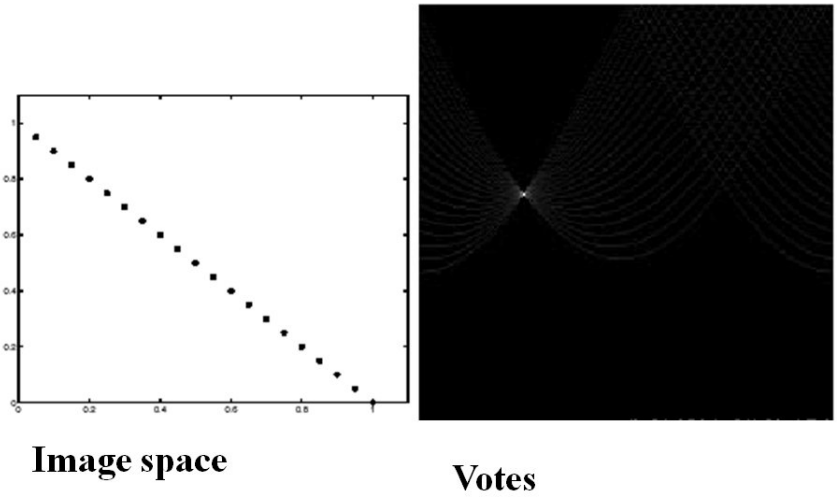


Figura 4.9: Hough Esempio 1

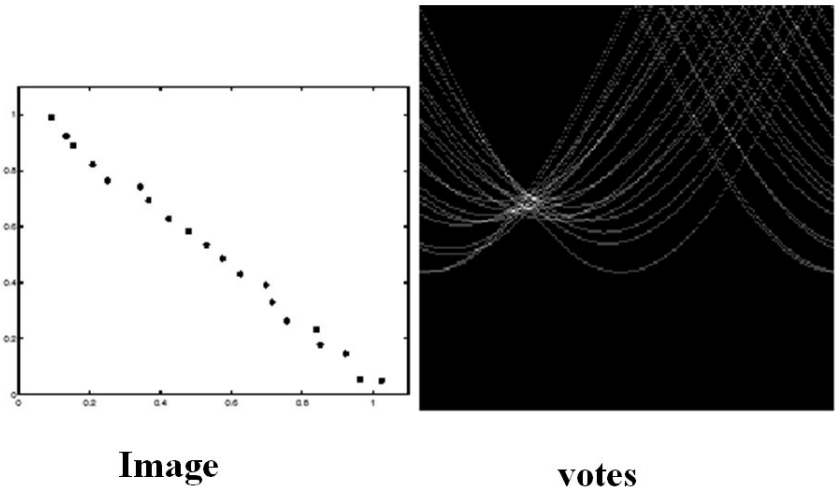


Figura 4.10: Hough Esempio 2

4.3 Hough Transform per 3D

Illustrata la Hough Transform generale si deve pensare cosa c'è di utile per il fine del progetto e cose invece deve essere rivoluzionato. Per prima cosa quanto detto fino a questo punto vale per quanto riguarda le 2 dimensioni e non le 3 dimensioni. Questo aspetto è fondamentale e porta a diverse variazioni. Il ragionamento principale va fatto per fare in modo che sia possibile identificare univocamente una retta nello spazio 3D. Per ideare tale strategia è stato preso spunto dal paper [21], più in particolare è stato pensato quanto segue.

Una retta nello spazio può essere descritta come intersezione di 2 piani ovvero:

$$ax + by + cz + d = 0 \ , \quad ex + fy + gz + h = 0 \tag{4.5}$$

oppure si può rappresentare in forma parametrica in questo modo:

$$retta = \begin{cases} x = lt + x_1 \\ y = mt + y_1 \\ z = nt + z_1 \end{cases} \tag{4.6}$$

la retta passa dal punto (x_1, y_1, z_1) ed è parallela al vettore (l, m, n).

Oltre a ciò, per poter identificare in modo univoco, ogni punto appartenente alla stessa retta è necessario qualcosa in più. Ovvero ogni retta verrà identificata con il versore ad essa associato grazie alla terna (Vx, Vy, Vz) ed alle coordinate del punto appartenente alla retta più vicino all'origine. Questo particolare punto può essere

visto anche come il punto della retta da cui parte la rette normale ad essa e passante per l'origine. Questo particolare punto viene indicato con la terna (O_x, O_y, O_z).
A questo punto si ha la possibilità di identificare in modo univoco una retta nello spazio 3D utilizzando 6 parametri $(V_x, V_y, V_z, O_x, O_y, O_z)$.

4.3.1 Idea base dell'algoritmo

Definito come si identifica una retta nello spazio 3D si procede con l'idea di base dell'algoritmo. Come visto in precedenza per l'Hough Transform del caso 2D anche qui se effettuare una procedura di voto per ogni punto. In particolare preso in esame il punto generico P (x_p, y_p, z_p) si dovranno scandire tutti i versori possibili con il livello di discretizzazione desiderato. Per effettuare tale attività verranno illustrati in seguito più vie possibili. In ogni caso, identificato il versore (V_x, V_y, V_z) si deve procedere con il calcolo delle coordinate del punto più vicino all'origine (O_x, O_y, O_z).

Partendo dalle formule della rappresentazione parametrica e definendo la distanza della retta dall'origine con *dist* otteniamo:

$$(x_p + t * V_x)^2 + (y_p + t * V_y)^2 + (z_p + t * V_z)^2 = dist^2 \tag{4.7}$$

Sviluppando i termini si ottiene:

$$x_p^2 + 2 * x_p * V_x * t + (t * V_x)^2 + y_p^2 + 2 * y_p * V_y * t + (t * V_y)^2 + z_p^2 + 2 * z_p * V_z * t + (t * V_z)^2 = dist^2 \tag{4.8}$$

Per trovare il punto più vicino all'origine ci interessa il minimo di questa funzione ovvero per procedere dobbiamo trovare la derivata prima in t e la imponiamo = 0 per trovare il minimo:

$$2 * x_p * Vx + 2 * t * V_x^2 + 2 * y_p * Vy + 2 * t * V_y^2 + 2 * z_p * Vz + 2 * t * V_z^2 = 0 \tag{4.9}$$

A questo punto isolando la t si ottiene:

$$t_{min} = -\frac{x_p * Vx + y_p * V_y + z_0 * V_z}{V_x^2 + V_y^2 + Vz^2} \tag{4.10}$$

Ottenuto tale risultato si sostituisce t_{min} alla rappresentazione parametrica ottenendo le coordinate dei punti (O_x, O_y, O_z).

$$retta = \begin{cases} O_x = V_x * t_{min} + x_p \\ O_y = V_y * t_{min} + y_p \\ O_z = V_z * t_{min} + z_p \end{cases} \tag{4.11}$$

A seguito di tutto questo procedimento si ottiene una sestupla di valori $(V_x, V_y, V_z, O_x, O_y, O_z)$, che identificano una precisa retta.
L'aspetto fondamentale dell'algoritmo consiste nel fatto che a questo punto tale retta

viene votata. Tale procedimento viene effettuato per i vari versori scansionati e per ogni punto presente nell'immagine. Al fine di tutto le rette che approssimano nel modo migliore i punti dell'immagine sono esattamente quelle che hanno ottenuto il maggior numero di voti. In altre parole è necessario trovare i massimi all'interno della struttura che memorizza i voti.

4.3.2 Scansione dei versori

Ora si entra più nel particolare dell'algoritmo in quanto per prima cosa si deve definire come si desidera effettuare la scansione dei possibili versori. Nel corso degli studi sono stati esaminati e confrontati diversi tipi di approccio nella ricerca della strada migliore. La scansione ideale dovrebbe coprire diversi requisiti tra cui:

1. Permettere di scandire tutti i versori a seconda di un livello di discretizzazione desiderato.

2. Non considerare un versore ed il suo opposto
   (es: $V_x = 1, V_y = 0, V_z = 0$ e $V_x = -1, V_y = 0, V_z = 0$).

3. garantire una equa distribuzione dei versori in tutte le direzioni in modo che i successivi voti abbiano pari peso.

Superficie sfera

La prima scansione presa in esame è la scansione che utilizza le formule della superficie della sfera.
Per prima cosa però è bene definire il sistema sferico per lo spazio. Questo è definito da tre coordinate: ρ, θ e φ. Prendendo in considerazione un generico punto P e la sua proiezione sul piano XY chiamata Q. Con ρ questa volta si indica la distanza di P dall'origine e θ è l'angolo che $\vec{\rho}$ forma con l'asse Z. Indichiamo invece con $\vec{\rho}\,'$ il vettore che collega l'origine con il punto Q, mentre φ rappresenta l'angolo che quest'ultimo vettore forma con l'asse X.

Per passare da un sistema sferico ad uno rettangolare si usano le seguenti uguaglianze:

$$\begin{cases} x = \rho \ * \ \sin\theta \ * \ \cos\phi \\ y = \rho \ * \ \sin\theta \ * \ \sin\phi \\ z = \rho \ * \ \cos\theta \end{cases} \tag{4.12}$$

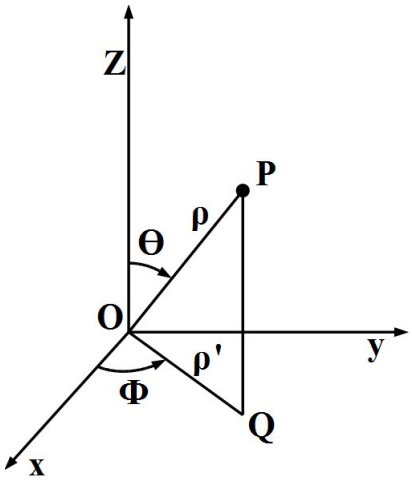


Figura 4.11: Sistema Sferico

Definito il sistema sferico si può passare alla superficie della sfera. I punti della superficie sferica possono essere parametrizzati in coordinate sferiche nel modo seguente:

$$\begin{cases} x = x_0 + r \ * \ \sin\theta \ * \ \cos\phi \\ y = y_0 + r \ * \ \sin\theta \ * \ \sin\phi \\ z = z_0 + r \ * \ \cos\theta \end{cases} \tag{4.13}$$

Come in precedenza θ e ϕ rappresentano la latitudine e la longitudine del punto, variando negli intervalli $0 \leq \theta \leq \pi$, $-\pi \leq \phi < \pi$. Ogni punto della superficie sferica è descritto da una sola coppia (θ, ϕ), tranne i poli che vengono rappresentati dalla coppia (0,ϕ) per il polo nord, e (π, ϕ) per il polo sud per qualsiasi valore di ϕ.

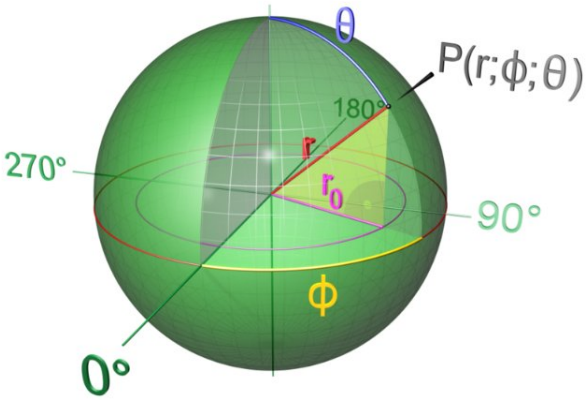


Figura 4.12: Sfera

Tale sistema viene adattato alle nostre necessità effettuando queste variazioni:

1. Interessandoci solo i versori possiamo tenere il raggio r = 1.

2. Il centro della sfera sarà rappresentato dal punto in esame perciò avremo che:

   (a) $x_0 = x_p$

   (b) $y_0 = y_p$

   (c) $z_0 = z_p$

3. Le coordinate trovata rappresenteranno quelle del versore:

   (a) $x = V_x$

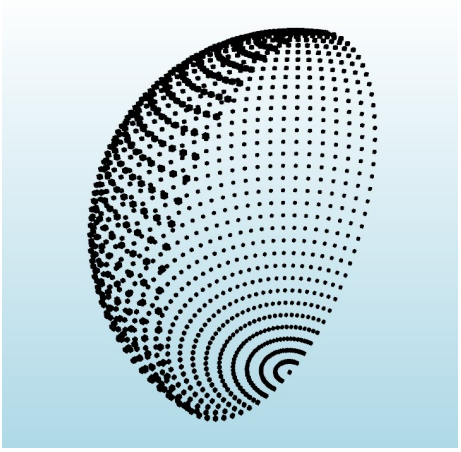
   (b) $y = V_y$

   (c) $z = V_z$

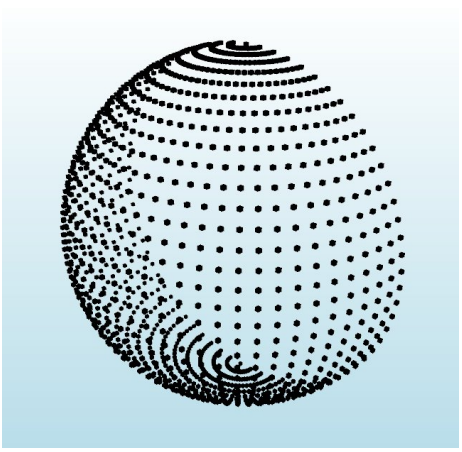
Ottenendo il seguente sistema che ci permette di calcolare le coordinate del versore partendo dagli angoli θ e ϕ:

$$\begin{cases} V_x = x_p + \ \sin\theta \ * \ \cos\phi \\ V_y = y_p + \sin\theta \ * \ \sin\phi \\ V_z = x_z + \cos\theta \end{cases} \tag{4.14}$$

In questo modo si possono avere i punti della superficie della sfera che verranno utilizzati come coordinate del versore ma effettuando un'attenta osservazione se venisse presa l'intera sfera andremo a considerare due volte tutte le possibili rette, Questo è dovuto dalla presenza di un generico versore e del suo opposto. Per ovviare a ciò è molto semplice in quanto è sufficiente non utilizzare l'intera sfera ma solo la parte che ci permette di non avere rette duplicate. Questo consiste nel lavorare con una cupola pari a metà sfera meno uno spicchio. Successivamente verrà spiegato il motivo.
Oltre a ciò nell'esecuzione dell'algoritmo va decisa la discretizzazione dei 2 angoli. Ovvero quanto incrementiamo l'angolo ad ogni scansione. Questo parametro sarà identificato come $inc\theta$ e $inc\phi$. Tornando al concetto di cupola ridotta ora possiamo precisare che $0 \leq \theta \leq \pi$ mentre $-\frac{\pi}{2} \leq \phi < \frac{\pi}{2} - inc\phi$.
Dall'esecuzione di tale procedura otteniamo la seguente cupola:



(a) Cupola 1a.



(b) Cupola 1b.

Figura 4.13

In prima apparenza il risultato sembra soddisfare i requisiti necessari. Ma successivamente ad alcuni test ed all'analisi delle immagini sopra riportate ci si è resi conto dei seguenti problemi.

Problemi

1. Ai poli ho punti sovrapposti

2. Man mano che mi avvicino ai poli ho una maggiore concentrazione dei punti, con una conseguenti non equicità dei pesi dei voti.

Superficie sfera alternativa

Nella sezione precedente è stata illustrata una tecnica sfruttando le equazioni di base della superficie della sfera. Un'altra possibile strada, sempre correlata alla sfera, consiste nell'utilizzare l'equazione cartesiana della superficie sferica:

$$x^2 + y^2 + z^2 + ax + by + cz + d = 0 \tag{4.15}$$

Anche questa volta tale equazione viene modificata in quando il centro della sfera viene posto nell'origine perciò si ottiene l'equazione semplificata:

$$x^2 + y^2 + z^2 = 1 \tag{4.16}$$

Grazie a tale condizione faremo variare x in questo range $0 \leq x \leq 1$, y in $0 \leq y \leq 1 - \sqrt{x}$ e z in $0 \leq z \leq 1 - \sqrt{x} - \sqrt{y}$. In questo modo otteniamo la seguente cupola :

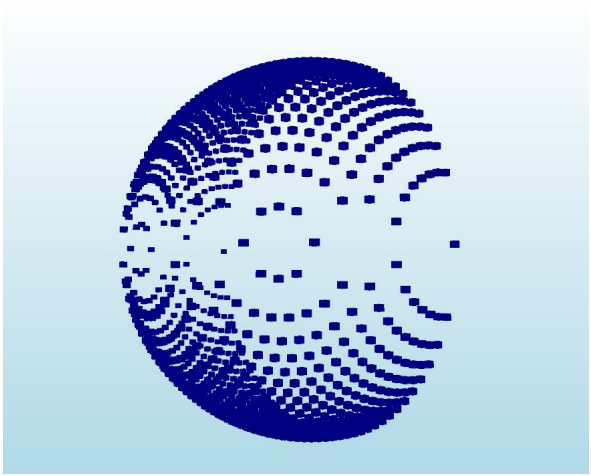


Figura 4.14: Cupola 3

Anche in questo caso il sistema definisce correttamente una sfera ma ai fini del progetto si sono riscontrati dei problemi.

Problemi

1. Non ho un'equa distribuzione dei punti

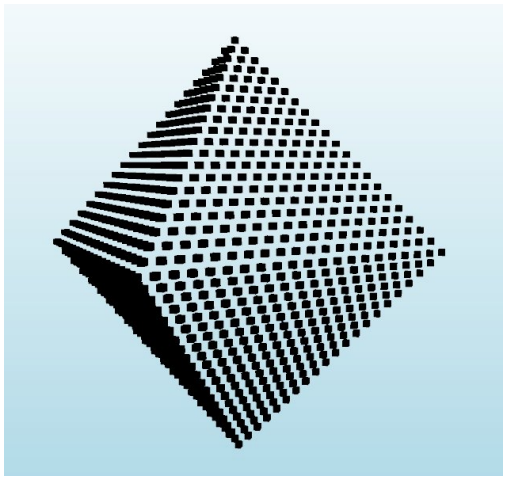
2. Man mano che mi avvicino ai poli ho una maggiore concentrazione dei punti

Rombo

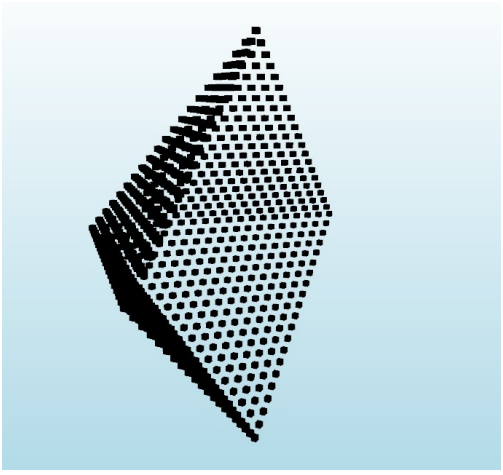
Un'altra possibile tipologia di scansione testata è stata la seguente. Questa tecnica non sfrutta la superficie della sfera come le precedenti ma un concetto molto più semplice cioè sfrutto le possibili soluzioni dell'equazione:

$$x + y + z = 0 \tag{4.17}$$

Facendo variare x, y e z rispettando l'equazione sopra citata otteniamo questa distribuzione di punti.



(a) Distribuzione versori 3 a.



(b) Distribuzione versori 3 b.

Figura 4.15

Con questa scelta non otteniamo più una sfera ma un'altra figura. Con questa distribuzione dei versori è stato più difficile riscontrare se rispecchiava o meno i canoni richiesti ma successivamente a delle analisi si è giunti alla conclusione che anche con questa distribuzione si rilevano gli stessi problemi dei precedenti tentativi.

Problemi

1. Non ho un'equa distribuzione dei punti

2. Man mano che mi avvicino agli assi ho una maggiore concentrazione dei punti

Icosaedro

In geometria l'icosaedro è un qualsiasi poliedro con venti facce. Con il termine icosaedro si intende però generalmente l'icosaedro regolare: nell'icosaedro regolare, le facce sono triangoli equilateri. La struttura base dell'icosaedro è raffigurata nella seguente immagine.

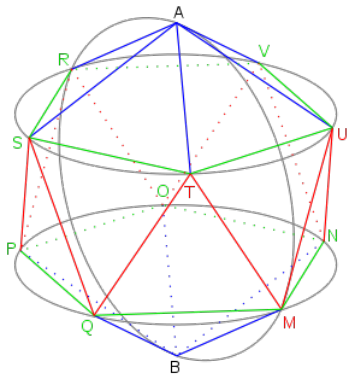


Figura 4.16: Struttura base dell'icosaedro

I 12 vertici principali dell'isocaedro sono definiti nel seguente modo sfruttando la costante $tao = 1.61803399$.

Punti

1. $p_1 = 1, tao, 0$
2. $p_2 = -1, tao, 0$
3. $p_3 = 1, -tao, 0$
4. $p_4 = -1, -tao, 0$
5. $p_5 = 0, 1, tao$
6. $p_6 = 0, -1, tao$
7. $p_7 = 0, 1, -tao$
8. $p_8 = 0, -1, -tao$
9. $p_9 = tao, 0, 1$
10. $p_10 = -tao, 0, 1$
11. $p_11 = tao, 0, -1$
12. $p_12 = -tao, 0, -1$

Da questi 12 punti si ottengono esattamente 20 facce a forma di triangolo. Queste facce compongono l'immagine 4.16 ma al nostro fine questo non è sufficiente. La fase successiva consiste nel dividere ogni singolo spigolo delle facce in n segmenti distinti e da questi ottenere la nuova suddivisione delle facce. Effettuando tale operazione si ottiene questo risultato.

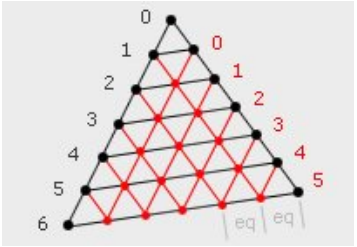


Figura 4.17: Suddivisione facce

A questo punto si ottiene un suddivisione molto più fitta delle facce ed i relativi punti di intersezione tra di essere. A questo punto si devono portare tali punti sulla superficie della sfera sfruttando la normale della faccia a cui appartengono. Terminata questa operazione abbiamo ottenuto il seguente risultato finale.

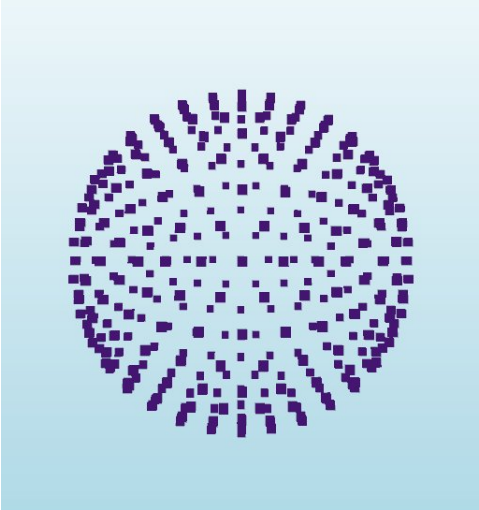


Figura 4.18: Sfera Icosaedro

A questo punto possiamo affermare di aver costruito un sistema che ha risolto le problematiche delle strade precedentemente analizzate e che ci permette di avere una distribuzione equidistante dei punti lungo la superficie della sfera. Questa struttura verrà utilizzata per il sistema di voto delle rette.

4.3.3 Voto delle rette

Precisato il metodo migliore per effettuare la scansione dei versori si può procedere con la fase successiva dell'algoritmo. Seguendo l'idea di base dell'algoritmo per ogni punto dobbiamo lanciare la procedura di voto. Questa testa tutti i versori creati dalla sfera isotropica e dà un voto a tale retta passante per il punto in esame. Effettuata la procedura di voto per tutti i punti ci troveremo ad avere una lista di

rette ed i corrispondenti voti.

Ottenuta tale lista si esegue una ricerca delle istanze, rette, con il numero maggiore di voti. Queste corrispondono alle rette che passano per il numero maggiore di punti.

Effettuando la scansione di un cubo e stampando le rette con maggiori voti otteniamo questo risultato finale. Si può notare che per ogni bordo non si ottiene un'unica retta ma molteplici. Questo fenomeno varia a seconda della risoluzione dell'immagine e delle tolleranze impostate.

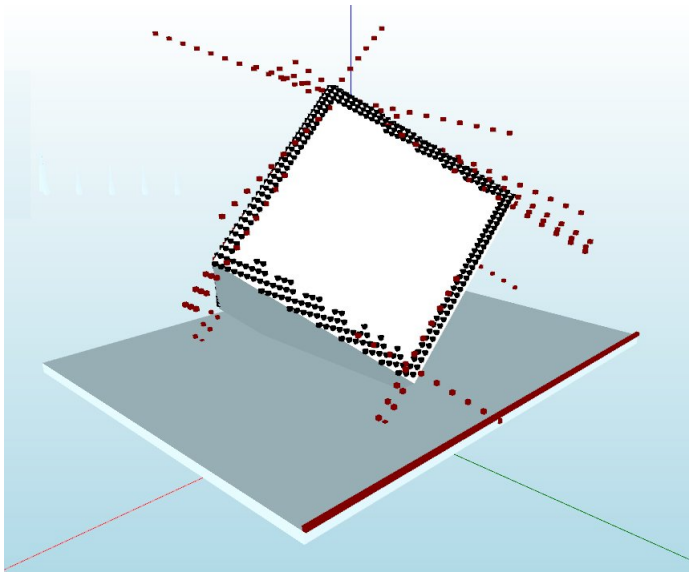


Figura 4.19: Rette rilevate dal cubo

4.3.4 Estrazione feature

Ricordando che lo scopo finale del progetto è di riconoscere un oggetto ed in che posizione è nel piano di lavoro, rilevare le rette che identificano nel modo migliore i bordi non è sufficiente. Il passo successivo consiste nel estrarre delle features significative dalle informazioni di cui siamo in possesso.

Con l'esempio del cubo potrebbe essere identificato perché vengono rilevati tutti i segmenti che rappresentano gli spigoli di pari lunghezza. Mentre dovendo rilevare una classica chiave inglese potrebbe essere necessario identificare 2 segmenti ad una certa distante. Questo al fine di riconoscere il corpo centrale della chiave.

Durante la fase di progettazione di questa sezione è stato deciso di estrarre le seguenti features.

Features

1. Rilevamento di rette parallele entro un certa tolleranza di angolo,
2. Rilevamento della distanza tra le rette identificate come parallele,
3. Identificazione del punto di distanza minima tra 2 rette sghembe nello spazio,
4. Identificazione dei segmenti con relativo punto iniziale e finale,
5. Identificazione dei punti di intersezione tra 2 segmenti (lati),
6. Identificazione dei punti di intersezione tra 3 segmenti (lati).

Rilevamento di rette parallele entro un certa tolleranza di angolo
La prima features rilevata è il Rilevamento di rette parallele entro un certa tolleranza di angolo. Per effettuare ciò sono state analizzate tutte le possibili combinazioni di coppie di rette. Per ognuna di essere è stato analizzato se le componenti dei versori delle 2 rette sono uguali entro una certa soglia.

1. $V_{x1} < V_{x2} + soglia$,
2. $V_{x1} > V_{x2} - soglia$,
3. $V_{y1} < V_{y2} + soglia$,
4. $V_{y1} > V_{y2} - soglia$,
5. $V_{z1} < V_{z2} + soglia$,
6. $V_{z1} > V_{z2} - soglia$.

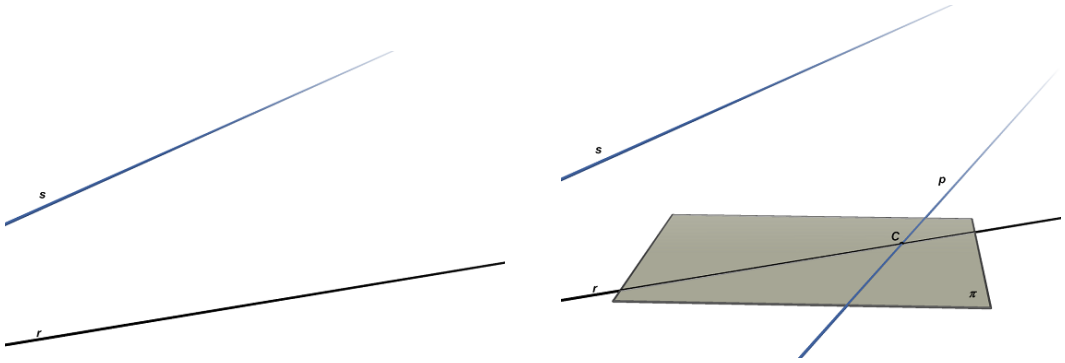
Rilevamento della distanza tra le rette identificate come parallele
Per estrarre questa feature il lavoro è molto semplice considerando un aspetto fondamentale. Se le due rette in esame sono state identificate come parallele e ricordando che ogni retta è rappresentata dalla sestupla $O_x, O_y, O_z, V_x, V_y, V_z$, avranno gli stessi valori del versore mentre valori differenti di O. Identificando come O_1 ed O_2 i due punti delle 2 rette, questi identificano un particolare segmento tra di essere $\overline{O_1O_2}$. Essendo le due rette parallele ed i punti O_1 ed O_2 i punti più vicini all'origine delle rispettive rette, il segmento $\overline{O_1O_2}$ ha la particolarità di essere perpendicolare sia alla retta1 che alla retta2. Questo ci porta alla rapida conclusione che per conoscere la distanza tra le due rette è sufficiente calcolare la lunghezza del segmento $\overline{O_1O_2}$.

Ovviamente tale procedura viene effettuata per ogni coppia di rette parallele identificate.

Identificazione del punto di distanza minima tra 2 rette sghembe nello spazio

Quello che si andrà a descrivere consiste nel metodo più importante e complesso delle estrazioni delle features sviluppare nel progetto. Prima di passare alla descrizione è utile capire la motivazione della sua creazione. Come descritto nelle sezioni precedenti otteniamo un insieme di rette che approssimano i bordi dell'oggetto in esame. Queste rette sono create grazie alla scansione dei versori con l'isocaedro. Ovviamente tale scansione è stata eseguita in modo accurato ma è sempre una forma di discretizzazione perciò non è detto che avremo l'esatta retta, ma un'approssimazione molto vicina. Questo aspetto è presente in tutte le rette identificare. Idealmente due rette si intersecano in un punto preciso dello spazio ma se come appena descritto queste sono approssimate comporta che le rette non si intersecheranno in un punto ma passeranno vicine ad un certa distanze. Perciò il rilevamento della distanza minima tra due rette sghembe nello spazio si rivela essenziale per il nostro scopo, in quanto i punti delle rette a minima distanza verranno successivamente utilizzati per identificare il punto di intersezione fittizio tra le due.

Spiegata la motivazione ed il fine di tale metodo si può passare all'esposizione della parte tecnica. Quanto segue descrive l'idea di base dell'algoritmo sviluppato. Come punto di partenza abbiamo due generiche rette r ed s nello spazio 3D. La prima operazione da effettuare consiste nel creare la retta p parallela alla retta s ma che intersechi la retta r. Si denomina il punto di intersezione tra p ed r con la lettera C, e si crea il piano π passante per r e p.



(a) Distribuzione versori 3 a.

(b) Distribuzione versori 3 b.

Figura 4.20

Passando alla parte matematica si ricorda che le rette in esame vengono fornite attraverso la sestupla (O_x, O_y, O_z, V_x, V_y, V_z). Perciò all'invocazione del metodo si hanno due sestuple una per identificare la retta s ed una per la retta r.

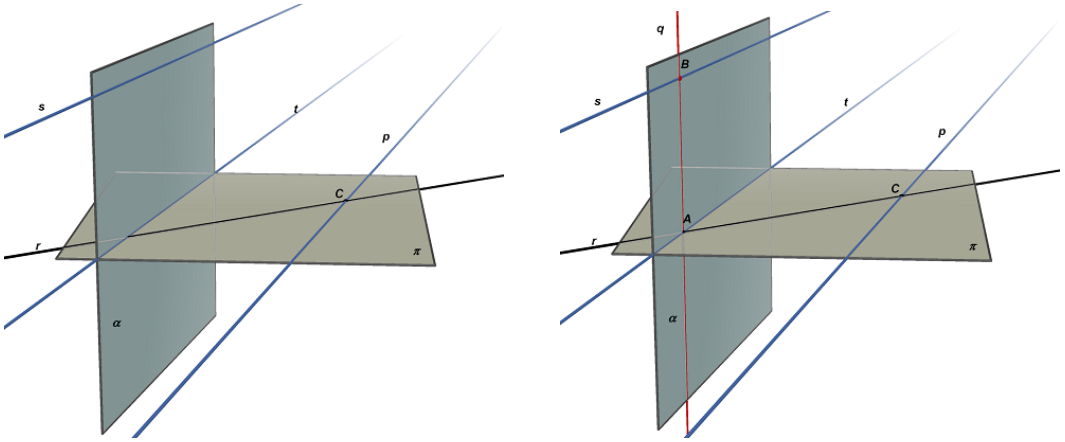
$$rettas = O_{xs}, O_{ys}, O_{zs}, V_{xs}, V_{ys}, V_{zs} \tag{4.18}$$

$$rettar = O_{xr}, O_{yr}, O_{zr}, V_{xr}, V_{yr}, V_{zr} \tag{4.19}$$

Questo è inerente alle rette mentre per quanto riguarda il piano π è rappresentato da questo sistema di equazioni:

$$\begin{cases} X\pi = V_{xp} * s + V_{xr} * t + O_{xr} \\ Y\pi = V_{yp} * s + V_{yr} * t + O_{yr} \\ Z\pi = V_{zp} * s + V_{zr} * t + O_{zr} \end{cases} \tag{4.20}$$

A questo punto si crea il piano α perpendicolare al piano π e passante per la retta iniziale s. Ora si identifica la retta t creata dall'intersezione tra il piano π ed il piano α. Questa retta ha la particolarità di essere parallela alla retta s. Successivamente troviamo il punto di intersezione tra la nuova retta t e la retta e la retta d'origine r e lo nominiamo con la lettera A. Ora si crea la retta q passante per il punto A, appartenente al piano α e perpendicolare alla retta s. Ed infine si trova il punto B intersezione tra la retta s e q.



(a) Distribuzione versori 3 a. (b) Distribuzione versori 3 b.

Figura 4.21

Passando alla parte matematica la prima cosa da calcolare è la normale del piano π che servirà in seguito.

$$\begin{cases} normle\pi.X = V_{yp} * V_{zr} - V_{zp} * V_{yr} \\ normle\pi.Y = V_{zp} * V_{xr} - V_{xp} * V_{zr} \\ normle\pi.Z = V_{xp} * V_{yr} - V_{yp} * V_{xr} \end{cases} \tag{4.21}$$

Il piano α è descritto dal seguente sistema:

$$\begin{cases} X\alpha = s * normle\pi.X + t * V_{xs} + O_{xs} \\ Y\alpha = s * normle\pi.Y + t * V_{ys} + O_{ys} \\ Z\alpha = s * normle\pi.Z + t * V_{zs} + O_{zs} \end{cases} \tag{4.22}$$

A questo punto bisogna calcolare l'intersezione dei piani π e α ma prima di fare ciò passo dalla rappresentazione parametrica a quella cartesiana.

Passaggio da rappresentazione parametrica a cartesiana

Partendo dalla classica rappresentazione parametrica del piano :

$$\begin{cases} X = V_{x1} * s + V_{x2} * t + O_{x2} \\ Y = V_{y1} * s + V_{y2} * t + O_{y2} \\ Z = V_{z1} * s + V_{z2} * t + O_{z2} \end{cases} \tag{4.23}$$

Calcoliamo un punto per ognuna delle 2 rette che formano il piano, per esempio impostando t = 100.

$$\begin{cases} P_{X1} = V_{x1} * t + O_{x2} = V_{x1} * 100 + O_{x2} \\ P_{Y1} = V_{y1} * t + O_{y2} = V_{y1} * 100 + O_{y2} \\ P_{Z1} = V_{z1} * t + O_{z2} = V_{z1} * 100 + O_{z2} \end{cases} \tag{4.24}$$

$$\begin{cases} P_{X2} = V_{x2} * t + O_{x2} = V_{x2} * 100 + O_{x2} \\ P_{Y2} = V_{y2} * t + O_{y2} = V_{y2} * 100 + O_{y2} \\ P_{Z2} = V_{z2} * t + O_{z2} = V_{z2} * 100 + O_{z2} \end{cases} \tag{4.25}$$

$$a = V_{y1} * V_{z2} - V_{z1} * V_{y2} \tag{4.26}$$

$$b = V_{z1} * V_{x2} - V_{x1} * V_{z2} \tag{4.27}$$

$$c = V_{x1} * V_{y2} - V_{y1} * V_{x2} \tag{4.28}$$

$$d = -(a * P_{X1}) - (b * P_{Y1}) - (c * P_{Z1}) \tag{4.29}$$

Avendo così ottenuto la forma cartesiana dell'equazione del piano cioè:

$$a * x + b * y + c * z + d = 0 \tag{4.30}$$

Si può procedere con il calcolo dell'intersezione tra il piano π ed il piano α. Questo calcola si effettua tenendo in considerazione i seguenti assegnamenti delle variabili.

$$Piano\pi : a_\pi * x + b_\pi * y + c_\pi * z + d_\pi = 0 \tag{4.31}$$

$$Pianoo\alpha : a_\alpha * x + b_\alpha * y + c_\alpha * z + d_\alpha = 0 \tag{4.32}$$

Le variabili detta retta t calcolato sono:

$$t_1 = (c_\pi * a_\alpha - c_\alpha * a_\pi)/(((-b_\pi) * a_\alpha) + b_\alpha * a_\pi) \tag{4.33}$$
$$t_2 = (d_\pi * a_\alpha - d_\alpha * a_\pi)/(((-b_\pi) * a_\alpha) + b_\alpha * a_\pi) \tag{4.34}$$
$$t_3 = (((-t1) * (b_\pi/a_\pi)) - (c_\pi/a_\pi)) \tag{4.35}$$
$$t_4 = (((-t2) * (b_\pi/a_\pi)) - (c_\pi/a_\pi)) \tag{4.36}$$

Ottenendo :

$$\begin{cases} X_t = s * t_3 + t_4 \\ Y_t = s * t_1 + t_2 \\ Z_t = s * 1 + 0 \end{cases} \tag{4.37}$$

Ora va calcolata l'intersezione tra la retta t e la retta r nel seguente modo :

$$j = ((t_3 * O_{yr}) - (O_{xr} * t_1) + (t_3 * t_1) - (t_1 * t_3))/((V_{xr} * t_1) - (V_{yr} * t_3)) \tag{4.38}$$
$$i = ((j * V_{xr}) + O_{xr} - t_4)/(t_3); \tag{4.39}$$

$$\begin{cases} X_A = i * t_3 + t_4 \\ Y_A = i * t_1 + t_2 \\ Z_A = j * V_{zr} + 0 \end{cases} \tag{4.40}$$

Attenuto il punto A si determina la retta q:

$$\begin{cases} X_A = t * normle\pi.X + X_A \\ Y_A = t * normle\pi.Y + Y_A \\ Z_A = t * normle\pi.Z + Z_A \end{cases} \tag{4.41}$$

E successivamente si calcola l'intersezione tra la retta q appena trovata e la retta iniziale s ottenendo il punto B di coordinate :

$$\begin{cases} X_B = i' * normle\pi.X + X_A \\ Y_B = i' * normle\pi.Y + Y_A \\ Z_B = j' * V_{zs} + O_{z2} \end{cases} \tag{4.42}$$

L'ultimo passo consiste nel calcolo della distanza 3D dal punto A al punto B :

$$dist\overline{\mathrm{AB}} = \sqrt{((X_B - X_A)^2 + (Y_B - Y_A)^2 + (Z_B - Z_A)^2)} \tag{4.43}$$

La retta q è perpendicolare alla retta s e al piano π e quindi anche alla retta r. E' pertanto la perpendicolare ad entrambe le rette sghembe r ed s. A questo punto punto dimostro che il segmento $\overline{\text{AB}}$ costituisce la distanza minima tra le due rette. Preso un punto D qualsiasi su s e un punto E qualsiasi su r, tracciamo da D la retta v perpendicolare alla retta t e, indicato con F il punto di intersezione tra la retta v e la retta t, tracciamo la retta u passante per F e per E. Le rette v e u sono perpendicolari e pertanto il triangolo DFE è retto in F. L'ipotenusa $\overline{\text{DE}}$ è maggiore del cateto $\overline{\text{DF}}$. Poiché $\overline{\text{DF}}$ è congruente al segmento $\overline{\text{AB}}$ poiché sono i lati opposti di un rettangolo, si deduce che il segmento $\overline{\text{AB}}$ è la minima distanza delle due rette sghembe.

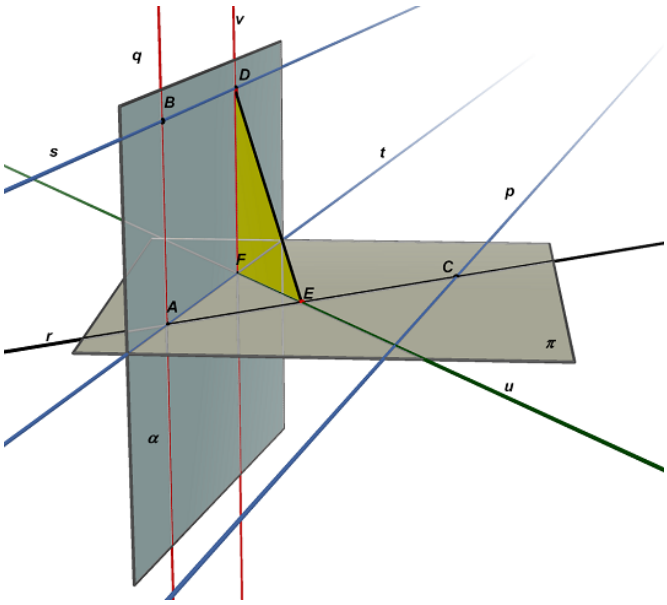


Figura 4.22: Rette rilevate dal cubo

Identificazione dei punti di intersezione tra 2 rette (lati)
Grazie alla procedura appena descritta siamo a conoscenza delle distanze minime tra due generiche rette sghembe nello spazio 3D. A questo punto è stata inserita una soglia per la distanza al di sotto della quale si considera che le due rette si intersecano. Il punto esatto dell'intersezione è identificato come il punto medio del segmento che rileva la distanza minima.

Identificazione dei punti di intersezione tra 3 rette (lati)
Per identificare l'intersezione di 3 rette ci sono più strade possibili ma è stato scelto di analizzare i risultati dell'intersezione tra 2 rette. In particolare con tale procedura siamo a conoscenza del punto di intersezione e di quali rette vi sono coinvolte. Alla luce di ciò per rilevare l'intersezione tre 3 rette bisogna effettuare una scansione

dell'intersezione tra 2 rette e testare quanto segue. Per prima cosa bisogna cercare i punti di intersezione che hanno delle coordinate che differiscono entro una certa soglia d'errore. Identificato tale punto si rilevano le rette che intersecavano in esso.

Identificazione dei segmenti con relativo punto iniziale e finale
L'esecuzione di questa procedura sfrutta i risultati del metodo che identifica il punto di intersezione tra 2 rette. Grazie ad esso siamo a conoscenza di tutti i punti di intersezione tra le varie coppie di rette. A questo punto si esegue una scansione di tutti questi punti di intersezione e si identificano le rette che ne hanno due. Questi 2 punti rappresentato i due estremi di un segmento. Alla fine della scansione otteremo tutti i segmenti rilevati dal sistema.

4.4 Hough Transform 3D per il rilevamento di cerchi

Gli oggetti con cui verrà utilizzato il risultato finale saranno principalmente piani e con dei fori presenti nella superficie. Con quanto descritto fino a questo punto il sistema è in grado di rilevare tutte le rette, i segmenti e gli spigoli presenti nell'oggetto. Ma sia per riconoscere la posizione dell'oggetto sia per eventuali punti in cui prendere l'oggetto è necessario rilevare la posizione dei cerchi (fori) e la normale di essi.

La base di partenza da cui si parte è la medesima della ricerca delle rette descritta fino ad ora. Cioè si è in possesso dei punti nell'intorno degli spigoli, ma gli spigoli possono essere sia quelli rettilinei di un cubo sia quelli che si formano su un foro di un oggetto.

Per lo sviluppo di tale metodo si parte dal presupposto che si sia a conoscenza del raggio del foro da ricercare.

La tecnica scelta per effettuare tale applicazione si basa sullo stesso concetto di quella per rilevare le rette ovvero un sistema di voto come Hough Transform. In questo caso però il procedimento, ovviamente, è differente essendo un cerchio l'oggetto in esame. L'idea di base dell'algoritmo è la seguente:

1. Si crea la sfera del diametro desiderato con il concetto dell'isocaedro,
2. $\forall$ punto (x_i,y_i,z_i) dei bordi si esegue quanto segue:
   (a) Si posiziona il baricentro della sfera nel punto (x_i,y_i,z_i) e si calcolano tutti i punti della superficie con il raggio prefissato,
   (b) $\forall$ punto calcolato si da un Voto,

3. Al termine di tutti i voti si cerca il punto che ha ottenuto il maggior numero di voti, questo è il centro del cerchio,
4. Ottenuto il centro cerco la normale del cerchio.

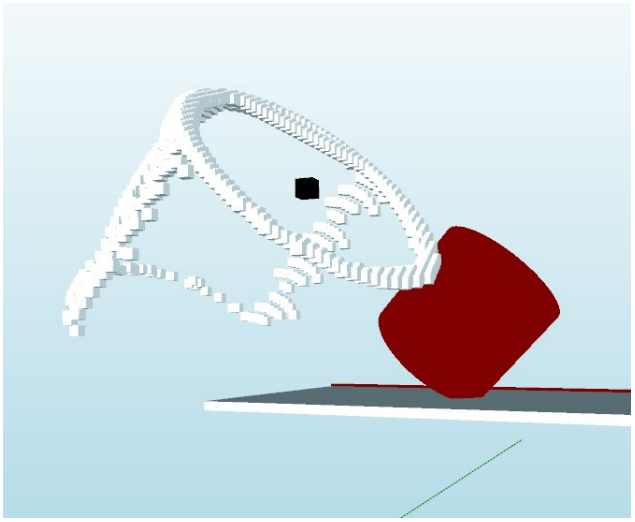


Figura 4.23: Centro del cerchio

Il punto 4, calcolo della normale del cerchio, merita di essere descritto essendo stato creato in modo particolare. Il punto di partenza di questo metodo è la conoscenza del centro del cerchio e di tutti i punti candidati ad essere bordi dell'oggetto. La prima cosa che è stato opportuno fare è di selezionare solo i punti all'interno della sfera con centro il centro del cerchio e del raggio ricercato. In questo modo si prendono in considerazione solamente i punti che fanno parte del cerchio. A questo punto si hanno i dati necessari per procedere con l'algoritmo di Ransac per il rilevamento del piano che approssima nel modo migliore una certa nuvola di punti.

Oltre alla lista dei punti sono necessari altri 3 parametri :

1. tolerance threshold : la soglia di tolleranza della distanza tra il piano ed i punti,
2. forseeable-support : il numero massimo probabile di punti che appartengono al piano,
3. probabilità α : probabilità minima di successi in N prove, che può essere impostata tra 0.9 e 0.99.

Per lo studio e l'analisi dell'algoritmo di Ransac sono stati utilizzati differenti paper tra cui i seguenti : [31], [32], [33], [34]. In modo particolare al fine dell'utilizzo vero e proprio dell'algoritmo è stato preso spunto dal paper [32].

Algoritmo di Ransac per l'identificazione di piani :

1. best Support = 0; bestPlane(3,1) = [0,0,0];
2. bestStd = ∞; i = 0;
3. ϵ = 1 - forseeable-support/length(point-list)
4. N = round(log(1 - α)/log(1 - (1 - ϵ)^3))
5. while i $\leq$ N do
6. j = pick 3 punti in modo random dalla lista di punti
7. pl = pts2plane(j) // dermino il piano passante per i 3 punti selezionati
8. dis = dist2plane(pl, point-list) // determino al distanza tra il piano ed i punti
9. s = find(abs(dis)$\leq$t) // seleziono tutti i punti che sono al un distanza inferiore alla soglia
10. st = Standard-deviation(s) // calcolo la deviazione standard di s
11. if(length(s) $\geq$ bestSupport) or (length(s) = bestSupport and st $\leq$ bestStd) then
12. bestSupport = length(s) // salvo il nuovo miglior piano identificato fino ad ora
13. bestPlane = pl; bestStd = st
14. end if
15. i = i + 1
16. end while

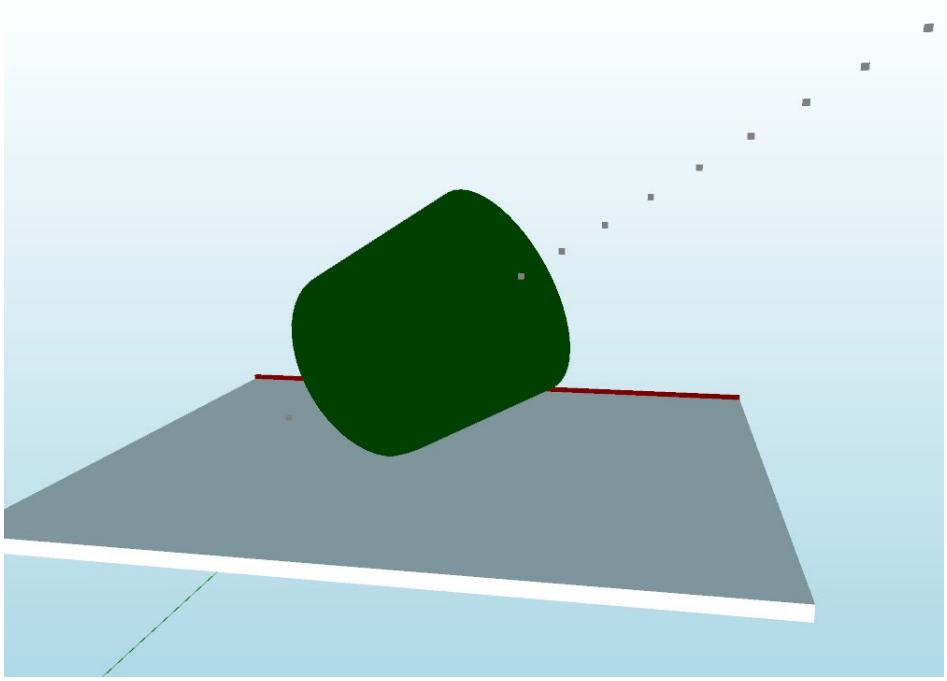


Figura 4.24: Esempio identificazione normale del cerchio

5

Test

Obiettivo: in questo capitolo sono descritti ed analizzati i vari test che sono stati effettuati per attestare il corretto funzionamento del prodotto finale.

5.1 Modalità di test

Lo scopo del progetto, come descritto nei capitoli precedenti, consiste nell'identificare degli oggetti conosciuti a priori nell'immagine scansionata dal sistema laser camera simulato nel mondo 3D. Tale operazione usufruisce di due funzionalità cruciali del progetto. Queste sono la possibilità di identificare i bordi di un oggetto e l'identificazione di cerchi, ovvero fori, con la corrispondente normale.

Per verificare il corretto funzionamento di queste 2 sezioni sono state testate singolarmente con differenti figure e successivamente si è proseguito con l'analisi dei risultati ottenuti.

Dunque, nelle successive sezioni saranno analizzati i test effettuati prima per la ricerca dei bordi e successivamente per la ricerca dei fori.

Ma ancor prima di effettuare i test riguardanti l'estrazione delle feature sono state eseguite delle verifiche riguardanti la costruzione delle mesh partendo dalla nuvola di punti.

5.2 Test Mesh

I primi test effettuati riguardano il controllo della costruzione delle mesh che rappresentano l'oggetto partendo dalla nuvola di punti. Il funzionamento di questa procedura è stato ampiamente descritto in precedenza e sono state inserite anche delle immagini esempio per attestare il corretto funzionamento.

In questa sezione di test è stato scelto di utilizzare i punti generati dalla scansione simulata del laser per creare la mesh dell'oggetto e la relativa visualizzazione nel simulatore 3D.

Il primo test è stato effettuato con un oggetto piastra metallica con 5 fori, 4 di una dimensione ed uno di un'altra. La prima esecuzione è stata effettuata con una risoluzione 100 x 100 mentre la seconda con 200 x 200.

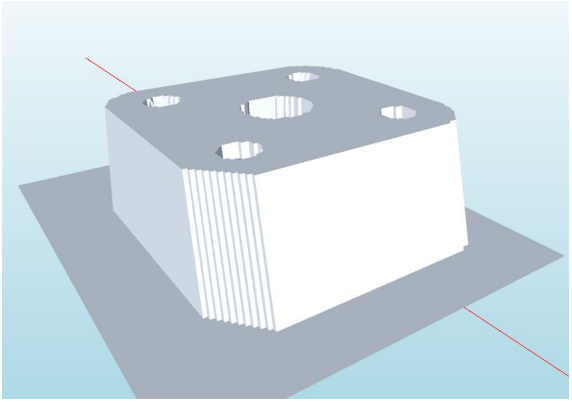


Figura 5.1: Mesh piastra 1, 100 x 100

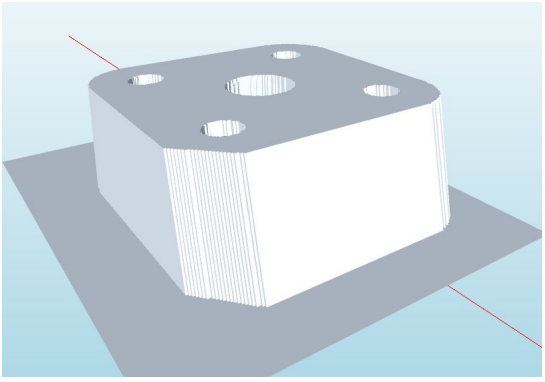


Figura 5.2: Mesh piastra 2, 200 x 200

Il secondo test è stato effettuato con lo stesso oggetto ma ruotandolo rispetto l'asse x ed y. La prima esecuzione è stata effettuata con una risoluzione 100 x 100 mentre la seconda con 200 x 200.

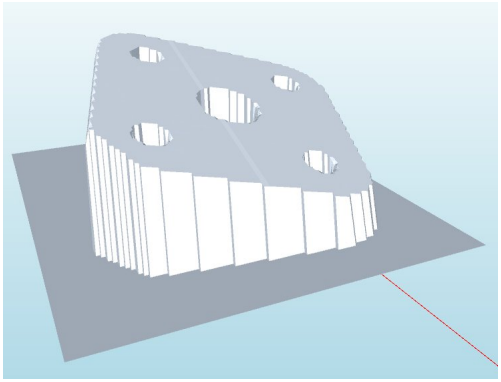


Figura 5.3: Mesh piastra ruotata 1, 100 x 100

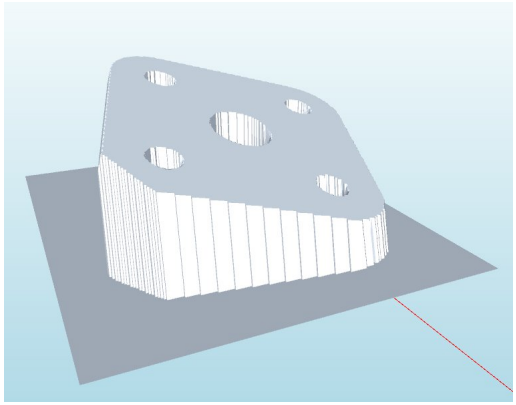


Figura 5.4: Mesh piastra ruotata 2, 200 x 200

Il secondo test è stato effettuato con la scansione di un pezzo robot. Nella prima immagini c'è l'oggetto originale mentre nella seconda l'oggetto creato dalla mesh. In questo caso la scansione è stata effettuata con risoluzione 400 x 400.

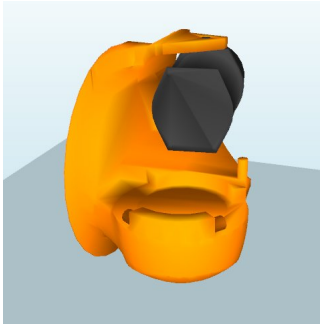


Figura 5.5: Pezzo robot originale

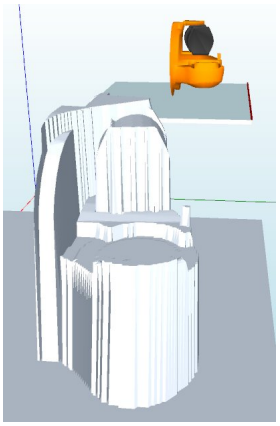


Figura 5.6: Pezzo robot, 400 x 400

5.3 Test bordi

Per i test relativi al rilevamento dei bordi è stato utilizzato un semplice oggetto che rappresenta una piastra metallica quadrata. Le informazioni utili su questo oggetto ai fini del progetto sono:

1. La piastra è quadrata e di lato 30 mm,
2. lo spessore è di 5 mm,
3. è stata posta al centro del piano di lavoro creato,
4. è stata ruotata di 45° rispetto l'asse z.

La risultante situazione di partenza è la seguente.

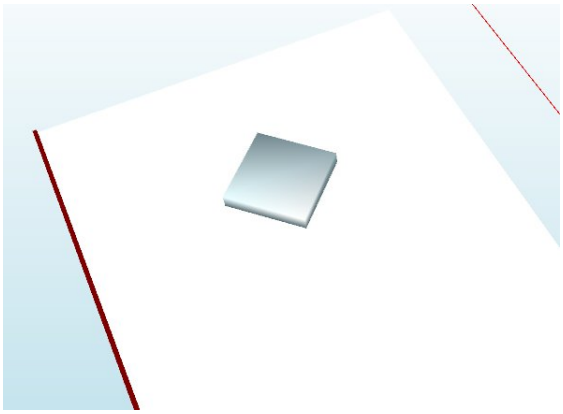


Figura 5.7: Test piano, situazione base

Lo scopo di questo algoritmo è di trovare le rette che approssimano nel modo migliore i bordi dell'oggetto, ma dato che è stato creato ed inserito per i test si è a conoscenza a priori delle equazioni delle rette. In particolare qui di seguito sono inserite le 4 rette con la rappresentazione a sestuple utilizzata lungo tutto il progetto.

1. retta 1 : 336 Ox, 252 Oy, 304 Oz, 0,7 Vx, -0,7 Vy, 0 Vz,

2. retta 2 : 360 Ox, 272 Oy, 304 Oz, 0,7 Vx, -0,7 Vy, 0 Vz,

3. retta 3 : -16 Ox, 16 Oy, 304 Oz, 0,7 Vx, 0,7 Vy, 0 Vz,

4. retta 4: 6 Ox, -6 Oy, 304 Oz, 0,7 Vx, 0,7 Vy, 0 Vz.

Durante i vari test sono state variate la risoluzione dell'immagine e la soglia di accettazione delle rette all'interno dell'algoritmo. Il tutto verrà chiarito, in seguito, in modo più approfondito.

Effettuata la ricerca dei bordi, questi verranno visualizzati nel mondo 3D nel modo seguente.

1. I cubi grigi rappresentano la retta identificata come bordo stampandola all'interno di un range prefissato lungo l'asse x,

2. I cubi neri rappresentano la retta ma il punto di partenza consiste nelle coordinate Ox, Oy, Oz,

3. Mentre i cubi oro sono i punti di intersezione tra le varie rette.

5.3.1 Test piastra 150 x 150, con soglia 210

Il primo test effettuato è con una risoluzione bassa ovvero rilevando 150 x 150 punti nel piano di lavoro. Inoltre è stata impostata una soglia di accettazione delle rette pari a 210. Solo le rette che accumulano un numero di voti al di sopra della soglia superano in test di accettazione e sono candidate come buone rette.

Ox	Oy	Oz	Vx	Vy	Vz	Voti
364	236	304	0,55	-0,85	0	266
316	316	304	0,7	-0,7	0	213
364	232	312	0,5	-0,85	0,05	282
316	316	308	0,65	-0,75	0,1	234
356	268	300	0,65	-0,75	-0,1	228

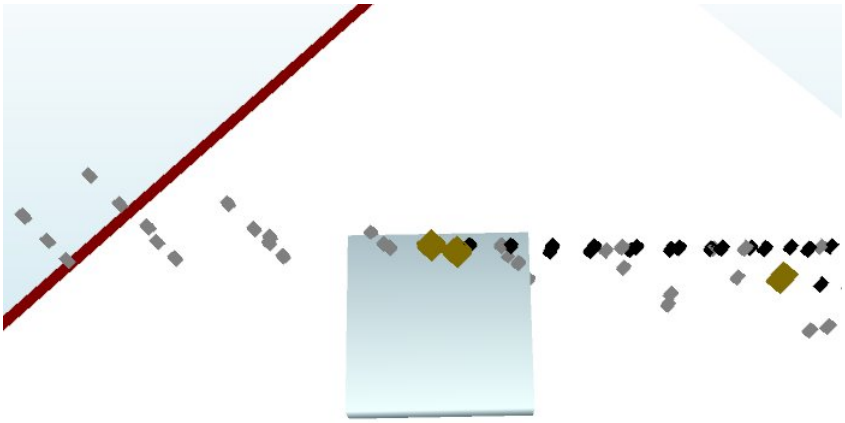


Figura 5.8: Test piano 150 x 150, soglia 210

5.3.2 Test piastra 150 x 150, con soglia 160

Per poter avere un'idea più chiara della scansione con 150 x 150 è stato rieseguito il test con una soglia più bassa, pari a 160 voti.

Ox	Oy	Oz	Vx	Vy	Vz	Voti
364	236	304	0,55	-0,85	0	266
308	328	304	0,65	-0,75	0,15	184
316	316	304	0,7	-0,7	0	213
336	292	304	0,65	-0,75	0	162
368	236	304	0,55	-0,85	0	182
356	268	300	0,65	-0,75	-0,1	228
348	224	296	0,6	-0,8	-0,1	164
340	220	304	0,55	-0,85	0	210
96	-72	284	0,5	0,85	0,05	162
316	316	308	0,65	-0,75	0,1	234
364	232	312	0,5	-0,85	0,05	282

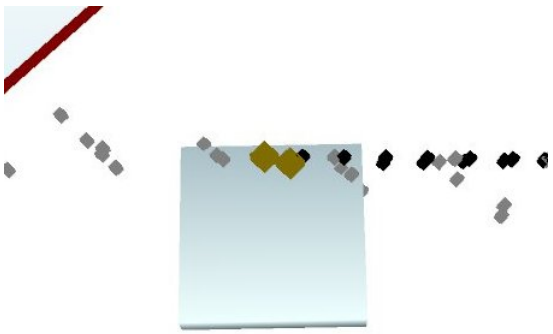


Figura 5.9: Test piano 150 x 150, soglia 180

Come si può notare il risultato con questa risoluzione d'immagine non è sufficiente. Con risoluzione 150 x 150 si ha una bassa accuratezza dei punti nell'intorno dei bordi dell'oggetto in esame. Qui di seguito è inserita la selezione dei punti che è stata identificata con questa risoluzione.

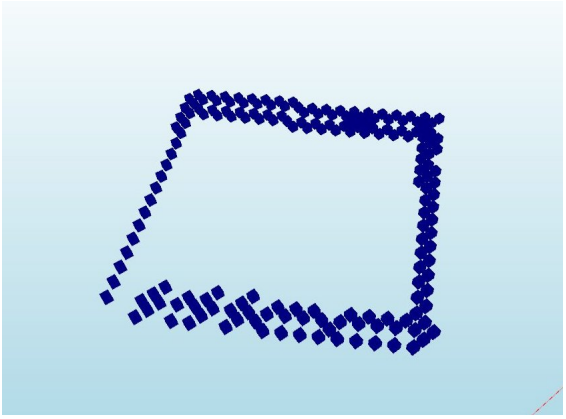


Figura 5.10: Test piano 150 x 150

Data questa bassa qualità, l'algoritmo non può identificare in modo preciso il bordo della figura perciò si procede con l'aumento della risoluzione della scansione.

5.3.3 Test piastra 200 x 200, con soglia 200

Con questo test è stata aumentata la risoluzione a 200 x 200 ed è stata utilizzata una soglia di 200 voti.

Ox	Oy	Oz	Vx	Vy	Vz	Voti
364	236	304	0,55	-0,85	0	273
308	328	304	0,65	-0,75	0,15	300
356	268	300	0,65	-0,75	-0,1	246
368	236	304	0,55	-0,85	0	217
296	340	304	0,7	-0,7	0,1	207
268	356	304	0,8	-0,6	0	201
340	220	304	0,55	-0,85	0	245
316	316	308	0,65	-0,75	0,1	294
364	232	312	0,5	-0,85	0,05	342
316	316	304	0,7	-0,7	0	252
368	240	304	0,55	-0,85	0	217
96	-72	284	0,5	0,85	0,05	222

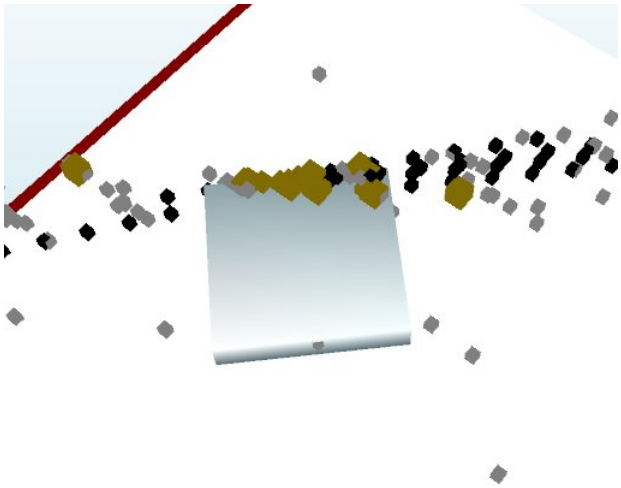


Figura 5.11: Test piano 200 x 200, soglia 200

Anche in questo caso il risultato non è soddisfacente e la selezione dei punti nell'intorno del bordo è la seguente.

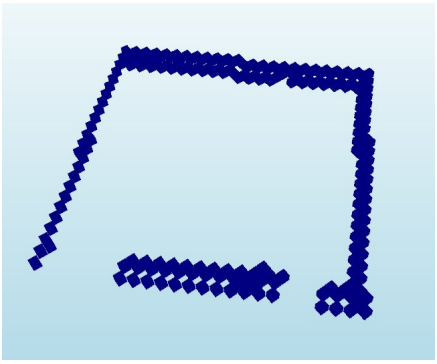


Figura 5.12: Test piano 200 x 200

5.3.4 Test piastra 250 x 250, con soglia 300

Il test successivo è stato effettuato con risoluzione 250 x 250, con una soglia di accettazione di 300.

La selezione dei punti nell'intorno del bordo è la seguente.

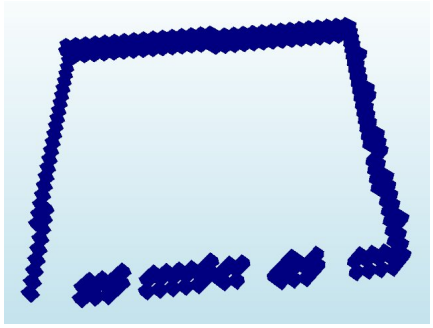


Figura 5.13: Test piano 250 x 250

Ox	Oy	Oz	Vx	Vy	Vz	Voti
296	340	304	0,7	-0,7	0,1	342
332	248	300	0,65	-0,75	-0,1	324
308	328	304	0,65	-0,75	0,15	384
316	316	304	0,7	-0,7	0	318
340	220	304	0,55	-0,85	0	336
364	236	304	0,55	-0,85	0	301
316	316	308	0,65	-0,75	0,1	366
364	232	312	0,5	-0,85	0,05	438
368	240	304	0,55	-0,85	0	315

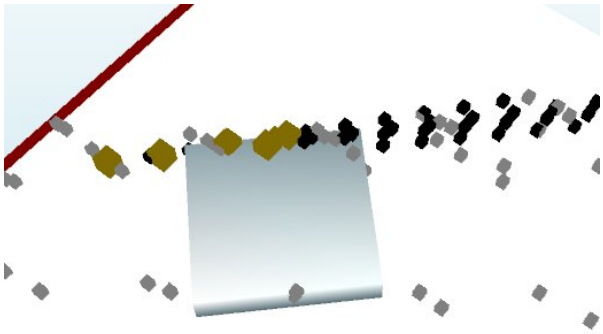


Figura 5.14: Test piano 250 x 250, soglia 300

Come si può notare dai valori nella tabella e dalla rappresentazione grafica delle rette il sistema non è riuscito ad identificare tutte e quattro le rette ma solo due di esse.

5.3.5 Test piastra 250 x 250, con soglia 270

Il test successivo è stato effettuato con risoluzione 250 x 250 e con una soglia di accettazione di 270.

Ox	Oy	Oz	Vx	Vy	Vz	Voti
364	236	304	0,55	-0,85	0	301
276	360	296	0,7	-0,7	0,2	284
308	328	304	0,65	-0,75	0,15	384
316	316	304	0,7	-0,7	0	318
360	276	292	0,7	-0,7	-0,2	280
368	236	304	0,55	-0,85	0	294
356	268	300	0,65	-0,75	-0,1	282
268	356	304	0,8	-0,6	0	279
340	220	304	0,55	-0,85	0	336
332	248	300	0,65	-0,75	-0,1	324
360	272	300	0,65	-0,75	-0,1	300
316	316	308	0,65	-0,75	0,1	366
296	340	304	0,7	-0,7	0,1	342
364	232	312	0,5	-0,85	0,05	438
368	240	304	0,55	-0,85	0	315

Anche in questo caso il sistema non è riuscito ad identificare tutte e quattro le rette ma solo due di esse.

5.3.6 Test piastra 300 x 300, con soglia 360

Il test successivo è stato effettuato con risoluzione 300 x 300 ed una soglia di accettazione di 360.

La selezione dei punti nell'intorno del bordo è la seguente.

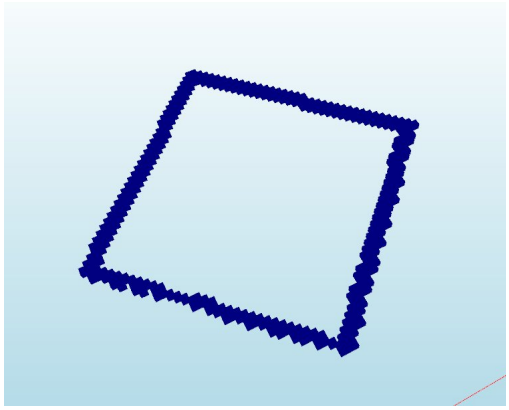


Figura 5.15: Test piano 300 x 300

I valori ottenuti sono i seguenti.

Ox	Oy	Oz	Vx	Vy	Vz	Voti
112	-48	328	0,5	0,85	-0,05	366
332	248	300	0,65	-0,75	-0,1	402
308	328	304	0,65	-0,75	0,15	368
32	-28	304	0,65	0,75	0	372
8	-8	304	0,7	0,7	0	375
56	-36	304	0,55	0,85	0	385
340	220	304	0,55	-0,85	0	455
-12	-24	260	0,65	0,75	0,1	432
96	-72	284	0,5	0,85	0,05	390

5.3.7 Test piastra 350 x 350, con soglia 420

Il test successivo è stato effettuato con risoluzione 350 x 350 ed una soglia di accettazione di 420.

La selezione dei punti nell'intorno del bordo è la seguente.

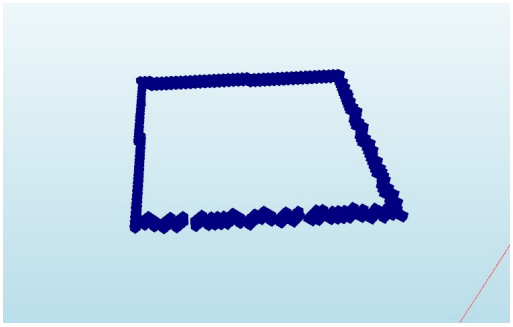


Figura 5.16: Test piano 350 x 350

Ox	Oy	Oz	Vx	Vy	Vz	Voti
296	340	304	0,7	-0,7	0,1	480
332	248	300	0,65	-0,75	-0,1	600
308	328	304	0,65	-0,75	0,15	556
268	356	304	0,8	-0,6	0	429
340	220	304	0,55	-0,85	0	595
316	316	308	0,65	-0,75	0,1	504
360	272	300	0,65	-0,75	-0,1	534
272	360	300	0,75	-0,65	0,1	423
364	232	312	0,5	-0,85	0,05	606
368	240	304	0,55	-0,85	0	483

5.3.8 Test piastra 350 x 350, con soglia 390

Il test successivo è stato effettuato con risoluzione 350 x 350 ed una soglia di accettazione di 390.
Da questo test si sono ottenuti i seguenti valori.

Ox	Oy	Oz	Vx	Vy	Vz	Voti
296	340	304	0,7	-0,7	0,1	480
368	240	304	0,55	-0,85	0	483
360	276	292	0,7	-0,7	-0,2	400
368	236	304	0,55	-0,85	0	392
272	360	300	0,75	-0,65	0,1	423
268	356	304	0,8	-0,6	0	429
364	236	304	0,55	-0,85	0	413
364	232	312	0,5	-0,85	0,05	606
276	360	296	0,7	-0,7	0,2	404
316	316	308	0,65	-0,75	0,1	504
332	248	300	0,65	-0,75	-0,1	600
292	296	308	0,65	-0,75	0,1	414
316	316	304	0,7	-0,7	0	393
308	328	304	0,65	-0,75	0,15	556
96	-72	284	0,5	0,85	0,05	414
340	220	304	0,55	-0,85	0	595
360	272	300	0,65	-0,75	-0,1	534

Con tale valori otteniamo le seguenti rette.

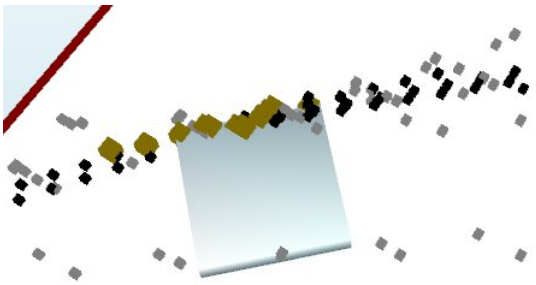


Figura 5.17: Test piano 350 x 350, soglia 420

5.3.9 Test piastra 350 x 350, con soglia 370

Il test successivo è stato effettuato con risoluzione 350 x 350 ed una soglia di accettazione di 370, ottenendo i seguenti valori.

id	Ox	Oy	Oz	Vx	Vy	Vz	Voti
10	296	340	304	0,7	-0,7	0,1	480
12	340	220	304	0,55	-0,85	0	595
6	368	240	304	0,55	-0,85	0	483
0	372	244	296	0,6	-0,8	-0,1	384
2	336	216	312	0,5	-0,85	0,05	390
24	32	-28	304	0,65	0,75	0	372
17	304	328	308	0,8	-0,6	-0,15	360
20	368	236	304	0,55	-0,85	0	392
9	272	360	300	0,75	-0,65	0,1	423
22	268	356	304	0,8	-0,6	0	429
11	360	276	292	0,7	-0,7	-0,2	400
1	364	236	304	0,55	-0,85	0	413
7	364	232	312	0,5	-0,85	0,05	606
18	276	360	296	0,7	-0,7	0,2	404
23	316	316	308	0,65	-0,75	0,1	504
3	112	-48	328	0,5	0,85	-0,05	384
25	332	248	300	0,65	-0,75	-0,1	600
5	292	296	308	0,65	-0,75	0,1	414
21	328	304	308	0,65	-0,75	0,05	352
4	316	316	304	0,7	-0,7	0	393
15	368	236	312	0,5	-0,85	0,05	384
19	308	328	304	0,65	-0,75	0,15	556
8	368	260	292	0,65	-0,75	-0,15	364
13	8	-8	304	0,7	0,7	0	381
16	96	-72	284	0,5	0,85	0,05	414
14	92	-72	284	0,5	0,85	0,05	360
26	360	272	300	0,65	-0,75	-0,1	534

La rappresentazione grafica di queste rette è la seguente.

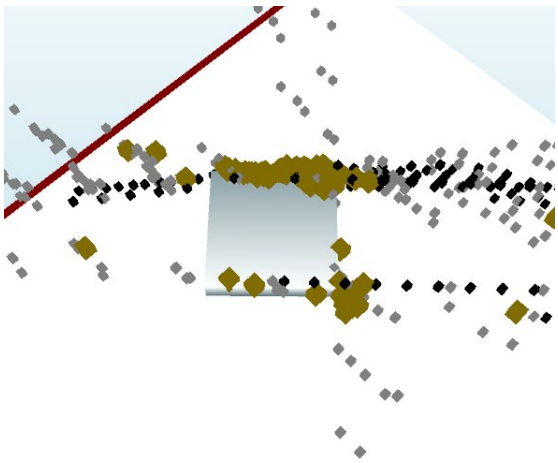


Figura 5.18: Test piano 350 x 350

Dato che i risultati iniziano ad essere di buona qualità si introduce un'altra informazione ovvero quali coppie di rette vengono rilevate come parallele ed a quale distanza.

id retta 1	id retta 2	distanza	id retta 1	id retta 2	distanza
20	23	32,25	22	24	38,78
22	26	36,87	10	21	48,44
11	21	16,97	24	26	1,60
0	24	16,97	2	23	9,79
18	19	45,95	9	21	1,73
7	17	5,65	17	23	33,22
1	7	8,90	2	7	32,24
17	20	8,00	6	7	12,00
0	26	30,72	5	21	36,87
24	25	1,78	1	6	5,65
2	6	4,00	10	11	31,49
1	17	8,90	2	17	37,73
10	15	31,49	5	11	31,24
6	17	8,90	10	19	16,97
1	23	28,80	11	19	14,96
12	14	0,17	13	16	31,24
6	23	34,40	7	23	1,17
5	19	1,50	1	2	35,32
1	20	0,17	2	20	38,50
6	20	0,17	7	20	9,97
5	10	4,40	0	22	4,03

5.3.10 Test piastra 400 x 400, con soglia 430

Il test successivo è stato effettuato con risoluzione 400 x 400 ed una soglia di accettazione di 430.
I vertici identificati nell'intorno del bordo sono i seguenti. Si vede subito che l'accuratezza di tale risultato ci permetterà di arrivare allo scopo finale.

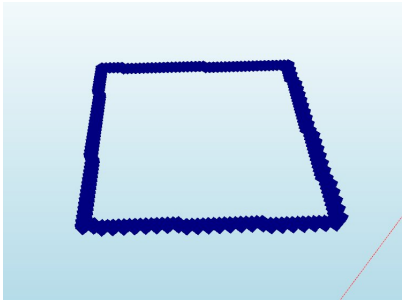


Figura 5.19: Test piano 400 x 400

Con questo test si sono ottenute le seguenti rette.

id	Ox	Oy	Oz	Vx	Vy	Vz	Voti
1	340	220	304	0,55	-0,85	0	707
2	368	240	304	0,55	-0,85	0	455
3	-12	16	304	0,75	0,65	0	468
4	336	216	312	0,5	-0,85	0,05	510
5	-12	-24	260	0,65	0,75	0,1	588
6	-56	24	260	0,75	0,65	0,1	465
7	-16	16	304	0,7	0,7	0	501
8	272	360	300	0,75	-0,65	0,1	459
9	32	-28	304	0,65	0,75	0	537
10	364	232	312	0,5	-0,85	0,05	576
11	364	236	304	0,55	-0,85	0	434
12	112	-48	328	0,5	0,85	-0,05	528
13	332	248	300	0,65	-0,75	-0,1	624
14	56	-36	304	0,55	0,85	0	525
15	292	296	308	0,65	-0,75	0,1	432
16	16	-44	260	0,6	0,8	0,1	492
17	-36	40	304	0,75	0,65	0	459
18	308	328	304	0,65	-0,7	0,15	500
19	68	-56	284	0,5	0,85	0,05	552
20	8	-8	304	0,7	0,7	0	546
21	96	-72	284	0,5	0,85	0,05	552
22	92	-72	284	0,5	0,85	0,05	456
23	360	272	300	0,65	-0,75	-0,1	636

5.3.11 Test piastra 600 x 600, con soglia 640

Il test successivo è stato effettuato con risoluzione 600 x 600 ed una soglia di accettazione di 640.
I vertici identificati nell'intorno del bordo sono i seguenti. Si vede subito che l'accuratezza di tale risultato ci permetterà di arrivare allo scopo finale.

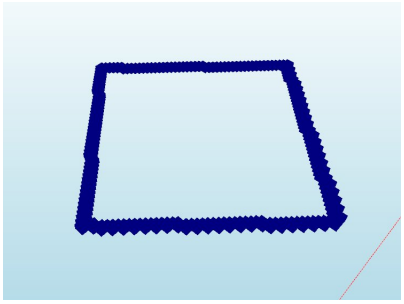


Figura 5.20: Test piano 600 x 600

id	Ox	Oy	Oz	Vx	Vy	Vz	Voti
1	296	340	304	0,7	-0,7	0,1	648
9	340	220	304	0,55	-0,85	0	1106
19	368	240	304	0,55	-0,85	0	665
11	-12	16	304	0,75	0,65	0	777
4	336	216	312	0,5	-0,85	0,05	780
14	-12	-24	260	0,65	0,75	0,1	896
20	-56	24	260	0,75	0,65	0,1	708
23	16	-44	260	0,6	0,8	0,1	716
2	-16	16	304	0,7	0,7	0	765
21	272	360	300	0,75	-0,65	0,1	690
8	84	-56	304	0,55	0,85	0	670
22	32	-28	304	0,65	0,75	0	849
17	364	232	312	0,5	-0,85	0,05	864
24	112	-48	328	0,5	0,85	-0,05	804
10	336	252	304	0,7	-0,7	0	996
5	56	-36	304	0,55	0,85	0	835
18	16	-48	260	0,65	0,75	0,1	748
3	-36	40	304	0,75	0,65	0	702
7	368	236	312	0,5	-0,85	0,05	684
0	308	328	304	0,65	-0,75	0,15	780
16	68	-56	284	0,5	0,85	0,05	840
13	6	-6	304	0,7	0,7	0	864
15	96	-72	284	0,5	0,85	0,05	900
12	92	-72	284	0,5	0,85	0,05	678
6	360	272	304	0,7	-0,7	0	1044

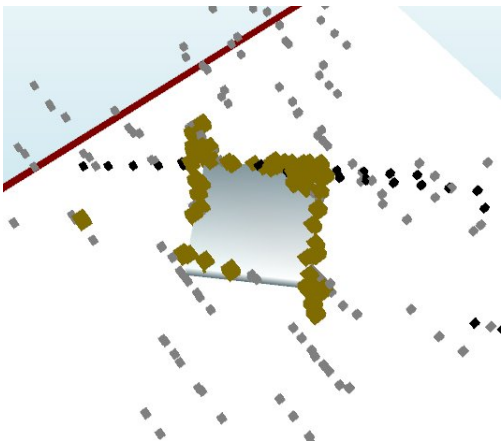


Figura 5.21: Test piano 600 x 600, 2

La prima cosa da mettere in risalto è la presenza delle quattro rette che identificano in modo esatto i bordi dell'oggetto:

1. retta 1 : id 10, 336 Ox, 252 Oy, 304 Oz, 0,7 Vx, -0,7 Vy, 0 Vz ,

2. retta 2 : id 6, 360 Ox, 272 Oy, 304 Oz, 0,7 Vx, -0,7 Vy, 0 Vz ,

3. retta 3 : id 2, -16 Ox, 16 Oy, 304 Oz, 0,7 Vx, 0,7 Vy, 0 Vz,

4. retta 4: id 13, 6 Ox, -6 Oy, 304 Oz, 0,7 Vx, 0,7 Vy, 0 Vz.

Inoltre si può passare all'analisi dei dati ottenuti, anche per quanto riguarda le rette parallele e la loro distanza.

id retta 1	id retta 2	distanza	id retta 1	id retta 2	distanza
18	21	4	4	19	40,79
5	12	54,7	12	23	28,84
2	3	31,24	3	24	33,94
14	21	34,4	13	24	31,24
15	23	32,25	5	15	57,41
14	18	36,88	4	7	37,73
9	20	28	8	12	26,83
2	24	4	8	23	25,61
2	17	65,11	4	9	9,79
7	9	33,22	12	15	4
13	17	31,24	5	23	30,72
5	8	34,41	7	19	8,94
2	13	31,11	3	13	65,11
4	20	32,25	7	20	5,65
8	15	28,28	19	20	12
6	10	31,24	8	22	37,74
1	16	31,49	0	1	16,97
9	19	34,4	5	22	62,09

Questo elenco rappresenta tutte le rette parallele entro una certa tolleranza e la relativa distanza tra di esse. Ai fini del progetto è indispensabile analizzare le distanze tra le rette identificate come i bordi. Queste sono le seguenti.

1. distanza tra 2 e 13 : 31,11,

2. distanza tra 6 e 10 : 31,24.

5.3.12 Conclusioni Test Rette

I test effettuati sono stati numerosi ed ognuno di essi con impostazioni differenti. Riassumendo i parametri sono i seguenti.

id test	risoluzione	soglia di accettazione
1	150 x 150	210
2	150 x 150	180
3	200 x 200	200
4	250 x 250	300
5	250 x 250	270
6	300 x 300	360
7	350 x 350	420
8	350 x 350	390
9	350 x 350	370
10	400 x 400	430
11	600 x 600	640

Alle fine di questi test si può affermare con certezza che l'algoritmo e la sua implementazione sono corrette e funzionanti. In particolare come da obiettivo il sistema identifica le rette che approssimano nel modo migliore i bordi dell'oggetto in esame.

Inoltre dall'analisi effettuata si è constatato che per il corretto funzionamento è necessario prestare particolare attenzione a due aspetti fondamentali.
Il primo, ed il più importante, consiste nella risoluzione dell'immagine. Questa deve essere sufficientemente elevata per permettere una buona approssimazione dei punti nell'intorno dei bordi. Più questa selezione dei punti è precisa migliore sarà il risultato finale. Contrariamente a ciò, se la risoluzione è troppo bassa il sistema identifica comunque delle rette, cioè le rette che totalizzano il numero maggiore di voti, ma queste saranno molto differenti rispetto al risultato obiettivo.
Il secondo punto consiste nel setting della soglia di accettazione. Questo parametro, come descritto lungo il progetto, filtra i risultati al di sotto di una certa soglia impostata. Questa però non può assumere lo stesso valore per tutti gli oggetti e la risoluzione va variata a seconda dei casi. La motivazione di questo aspetto è molto intuitiva in quanto se il numero di punti che identificano un bordo con una determinata risoluzione ammontano ad una certa quantità, se si prende la stessa immagine ma con una risoluzione doppia il numero di punti dello stesso bordo raddoppierà pure lui. Di conseguenza pure il numero di voti per tale retta aumenterà.
Va precisato che l'algoritmo funziona comunque e che questo è solo un filtro in modo da estrarre le rette che hanno totalizzato il numero maggiore di voti. Un'altra strada alternativa potrebbe essere di ordinare in modo decrescente in base al numero di voti tutte le rette e di selezionare solo le prime n desiderate.

5.4 Test fori

Per i test relativi al rilevamento dei fori sono stati utilizzati differenti oggetti ciascuno caratterizzato da alcune particolarità.

Gli oggetti scelti per questi test sono elencati qui di seguito. Le caratteristiche precise di ognuno di essi verranno precisate man mano che verranno utilizzati.

1. Piastra con 4 fori uguali,
2. Piastra con 4 fori uguali ed uno di dimensione differente,
3. Cubo con foro,
4. Oggetto assemblato da 3 tubi.

Per quanto riguarda i risultati dei test verranno inseriti su delle tabelle e visualizzati graficamente con le schermate delle immagini esportate dal mondo 3D. In particolare nelle immagini verranno visualizzati.

1. I centri dei fori con un cubo viola,
2. le normali dei fori con una sequenza di cubi neri.

5.4.1 Piastra con 4 fori uguali

I primi test sono stati effettuati con un oggetto rappresentante una piastra metallica con 4 fori di pari diametro. Le caratteristiche fisiche dell'oggetto sono riassunte nei seguenti punti.

1. Dimensioni dell'oggetto 50 x 50,
2. numero fori 4,
3. diametro fori 4,
4. posizione foro 1 : 287, 320, 304,
5. posizione foro 2 : 287, 300, 304,
6. posizione foro 3 : 311, 300, 304,
7. posizione foro 4 : 311, 320, 304,
8. normale foro 1 : 0, 0, 0,
9. normale foro 2 : 0, 0, 0,
10. normale foro 3 : 0, 0, 0,

11. normale foro 4 : 0, 0, 0,

La risultante situazione di partenza è la seguente.

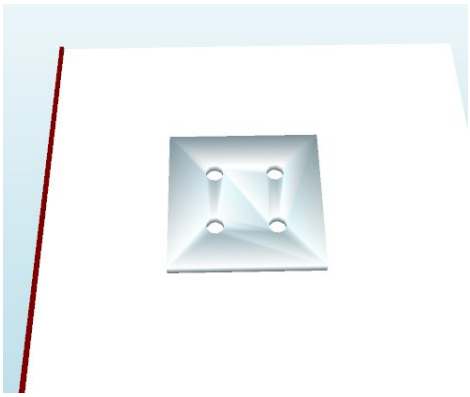


Figura 5.22: Test piastra 4 fori, situazione base

Ora che si è precisata la situazione di partenza si può proseguire con la descrizione del test. L'oggetto in esame è quello appena descritto e l'obiettivo alla fine dell'esecuzione è di ottenere le posizioni e le normali dei fori precise.

5.4.2 Piastra con 4 fori uguali, 60 x 60

La prima scansione è stata effettuata ad una bassissima risoluzione ovvero con 60 x 60 punti. Con questi parametri il sistema ha dato in uscita i seguenti valori.

Centri dei fori.

id foro	X	Y	Z	voti
1	325	308,5	304	85
2	325	322	304	85
3	325	291,5	304	85
4	325	325,5	304	81

Mentre per le normali.

id normale	X	Y	Z
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0

Tali risultati sono visualizzati nelle seguenti immagini.

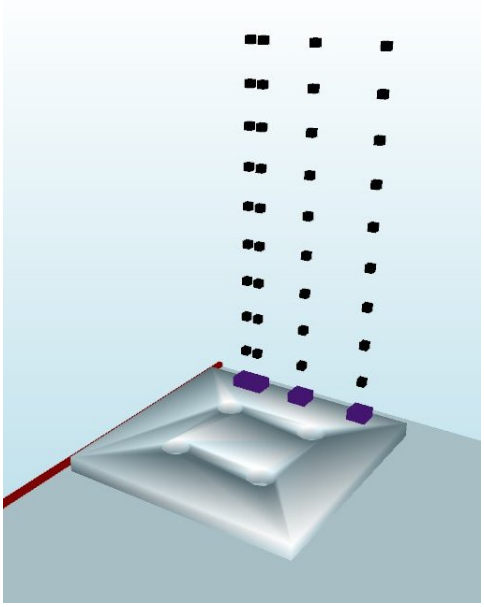


Figura 5.23: Test 1 piastra con 4 fori

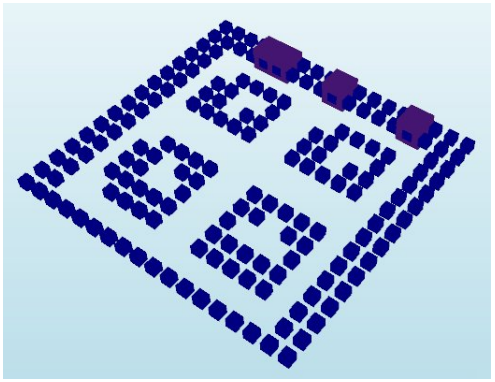


Figura 5.24: Test 2 piastra con 4 fori

Con il primo test si è constatato che la normale è stata identificata correttamente mentre riguardo i centri dei fori il sistema non ha dato il risultato sperato. Si prosegue con il test successivo aumentando la risoluzione dell'immagine.

5.4.3 Piastra con 4 fori uguali, 180 x 180

Nella seconda scansione è stata raddoppiata la risoluzione a 180 x 180. Questo cambiamento sicuramente porta ad un aumento di precisione dei risultati. Con questi parametri il sistema ha dato in uscita i seguenti valori.

Centri dei fori.

id foro	X	Y	Z	voti
1	287	321	304	164
2	287	299	304	141
3	311	298	304	140
4	310,5	321	304	139

Mentre per le normali.

id normale	X	Y	Z
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0

Tali risultati sono visualizzati nelle seguenti immagini.

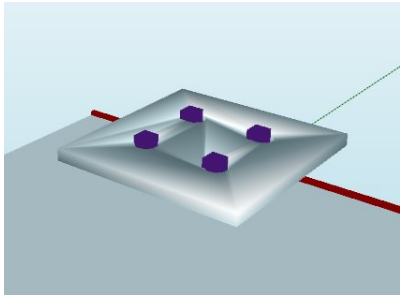


Figura 5.25: Test 3 piastra con 4 fori

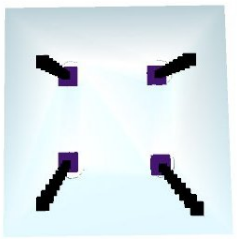


Figura 5.26: Test 4 piastra con 4 fori

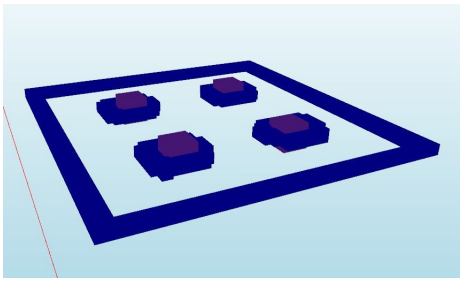


Figura 5.27: Test 5 piastra con 4 fori

Sia dai valori nelle tabelle sia dalle immagini si nota subito che il sistema ha lavorato correttamente ed ha portato i risultati sperati. Infatti dalla prima e dalla terza immagine si può notare che i centri sono stati trovati correttamente, mentre dalla terza si vede che pure le normali dei fori sono precise.

Per quanto riguarda l'analisi dei valori nelle tabelle, che danno una stima molto più precisa del successo del sistema, si nota qualche leggero scostamento dai valori ottimi. Questo puo essere dovuto dalla risoluzione ancora insufficiente dell'immagine oppure anche dalle discretizzazioni inserite nel sistema.

5.4.4 Piastra con 4 fori uguali, 150 x 150, ruotata

Dato che nei test precedenti l'oggetto era parallelo al piano e di conseguenza le normali pure, è stato deciso di ruotale tale oggetto di $\pi/12$ rispetto l'asse x e $\pi/12$ rispetto l'asse y.
Il primo test è stato effettuato con risoluzione 150 x 150 ed ha ottenuto i seguenti risultati.

Centri dei fori.

id foro	X	Y	Z	voti
1	287,5	298	303	142
2	287,5	321	297	138
3	305,5	304	310	125
4	311	300	308,5	125

Mentre per le normali.

id normale	X	Y	Z
1	-21,092	11,235	29,573
2	-4,335	8,823	7,976
3	-0,322	7,748	-11,3
4	-4,077	3,938	14,697

Il secondo test è stato effettuato con risoluzione 180 x 180 ed ha ottenuto i seguenti risultati.

Centri dei fori.

id foro	X	Y	Z	voti
1	287,5	321	296,5	181
2	287,5	298,5	303	157
3	310,5	322,5	303	153
4	287,5	298	303	157

Mentre per le normali.

id normale	X	Y	Z
1	-7,5	7,2	27,1
2	-7,4	11,8	26,6
3	0,7	-0,7	-2,6
4	-2,2	2,1	8

Il terzo test è stato effettuato con risoluzione 220 x 220 ed ha ottenuto i seguenti risultati.

Centri dei fori.

id foro	X	Y	Z	voti
1	287	298,5	303,5	252
2	310	322,5	303,5	246
3	310	299,5	309,5	244
4	287	321,5	298	238

Mentre per le normali.

id normale	X	Y	Z
1	-4,452	4,3	16,049
2	1,817	-1,768	-3,507
3	2,862	-2,764	-10,314
4	-2,007	1,939	7,236

Rappresentando i risultati dell'ultimo test graficamente si ha la seguente immagine.

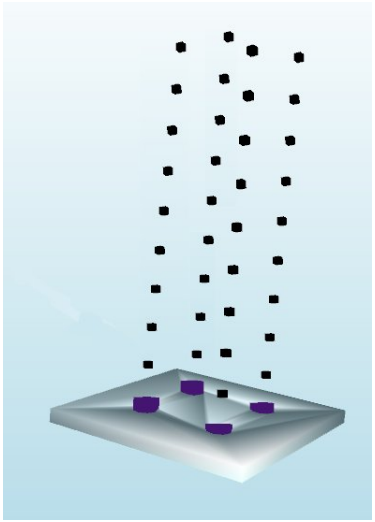


Figura 5.28: Test 6 piastra con 4 fori

Come si puo notare anche in questo caso il sistema ha identificato correttamente le posizioni dei centri dei fori e la loro relativa normale.

5.4.5 Piastra con 4 fori uguali ed uno di dimensione differente

I test precedenti sono stati effettuati grazie ad un oggetto con 4 fori uguali. Mentre il seguente test vuole mettere alla prova l'algoritmo rendendo un po più difficile l'esecuzione. Ovvero invece di esserci solo i 4 fori uguali ne viene inserito un altro di dimensione differente. Ma lo scopo è comunque quello di cercare i 4 fori di diametro 9. L'oggetto in esame è il seguente.

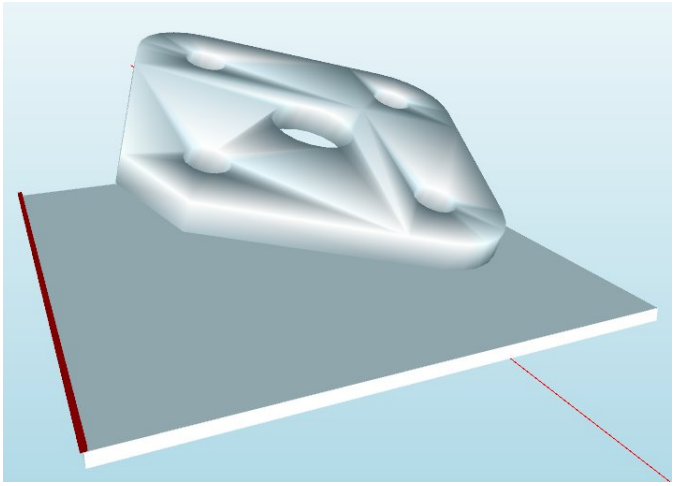


Figura 5.29: Test 1 piastra con 5 fori

I valori obiettivo di questo test sono i seguenti:

1. Centro foro 1 : x = 226,5, y = 293,5, z = 267,
2. Centro foro 2 : x = 284,5, y = 235,5, z = 300,5,
3. Centro foro 3 : x = 333, y = 302,5, z = 332,5,
4. Centro foro 4 : x = 275,5, y = 360,5, z = 298,
5. Normale foro 1 : x = 44,77, y = 1,77, z = -73,4,
6. Normale foro 2 : x = 44,77, y = 1,77, z = -73,4,
7. Normale foro 3 : x = 44,77, y = 1,77, z = -73,4,
8. Normale foro 4 : x = 44,77, y = 1,77, z = -73,4,

Il primo test effettuato è stato eseguito con risoluzione 100 x 100. Inoltre sono stati utilizzati i seguenti parametri di Ransac, double threshold = 5, double forseeable_support = 15, double alfa = 0.95, double bestsupport = 8. Con questa prima esecuzione si sono ottenuti i seguenti valori.

Centri dei fori.

id foro	X	Y	Z	voti
1	226,5	293,5	267	34
2	333	302,5	332	33
3	285	235,5	300	32
4	276	360,5	297	30

Mentre per le normali.

id normale	X	Y	Z
1	-27,89	5,32	40
2	-52,66	16,39	94,96
3	-65,59	-2,59	107,54
4	0	0	0

Il secondo test è stato effettuato con risoluzione 150 x 150 e con gli stessi parametri di Ransac dell'esecuzione precedente. Con questa esecuzione si sono ottenuti i seguenti valori.

Centri dei fori.

id foro	X	Y	Z	voti
1	226,5	293,5	267	76
2	284,5	235,5	300,5	73
3	333	302,5	332,5	73
4	275,5	360,5	298	67

Mentre per le normali.

id normale	X	Y	Z
1	41,76	-13,67	-35,19
2	-45,67	0	3,17
3	27,81	-56,08	-41,1
4	-63,22	-1,62	102,28

Il terzo test è stato eseguito ancora con risoluzione 150 x 150 ma sono stati variati i parametri di Ransac dato che non risultavano idonei al caso in esame. Perciò si è scelto di passare da questi.

1. double threshold = 5,
2. double forseeable_support = 15,
3. double alfa = 0.95,

4. double bestsupport = 8.

Ai seguenti.

1. double threshold = 5,
2. double forseeable_support = 12,
3. double alfa = 0.95,
4. double bestsupport = 15.

Grazie a questa variazione dei parametri si sono ottenuti i seguenti nuovi valori.

Centri dei fori.

id foro	X	Y	Z	voti
1	226,5	293,5	267	76
2	284,5	235,5	300,5	73
3	333	302,5	332,5	73
4	275,5	360,5	298	67

Mentre per le normali.

id normale	X	Y	Z
1	45,58	1,65	-71,34
2	42,64	1,8	-74,63
3	44,77	1,77	-73,4
4	44,77	1,77	-73,4

Visualizzando graficamente l'ultima sequenza di valori si può apprezzare la loro correttezza.

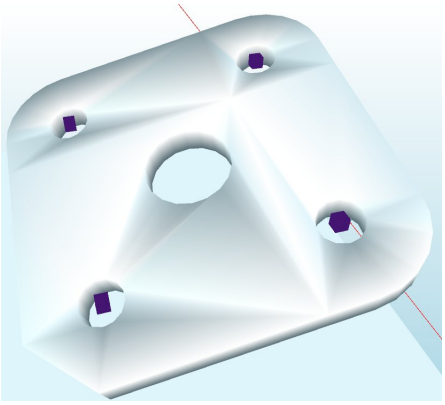


Figura 5.30: Test 2 piastra con 5 fori

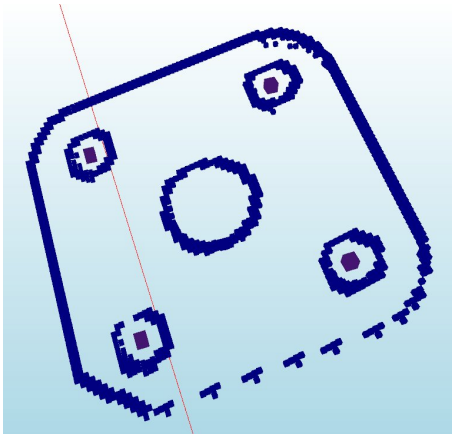


Figura 5.31: Test 3 piastra con 5 fori

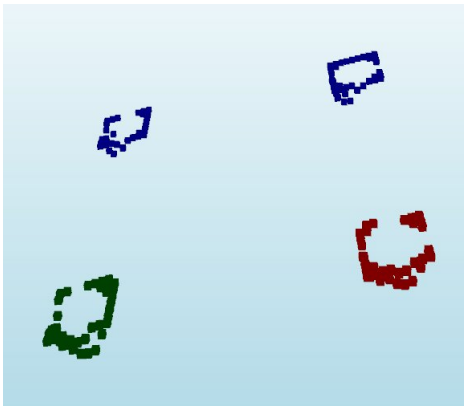


Figura 5.32: Test 4 piastra con 5 fori

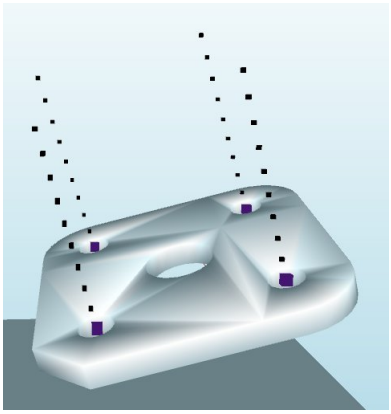


Figura 5.33: Test 5 piastra con 5 fori

5.4.6 Cubo con foro

Questa nuova serie di test è stata eseguita con un oggetto costituito da un cubo con un foro passante di diametro 12mm. Come sempre lo scopo è di ricercare il centro del foro e la relativa normale.

I risultati obiettivo di questi test sono.

1. Centro del foro : x = 283,5, y = 295,5, z = 354,

2. normale del foro : x = 2,96, y = 0,12, z = -4,85.

Anche con questo oggetto sono stati eseguiti numerosi test partendo da una risoluzione bassa ed aumentandola man mano fino ad ottenere dei risultati soddisfacenti.

Il primo test è stato eseguito con una risoluzione 60 x 60 ed ha ottenuto come centro del foro i seguenti valori : x = 280,00, y = 325,5, z = 354.

Il secondo test è stato eseguito con una risoluzione 100 x 100 ed ha ottenuto come centro del foro i seguenti valori : x = 284, y = 296,00, z = 352.

Il terzo test è stato eseguito con una risoluzione 200 x 200 ed ha ottenuto come centro del foro i seguenti valori : x = 283,5, y = 295,5, z = 354.

Il quarto test è stato eseguito con una risoluzione 400 x 400 ed ha ottenuto come centro del foro i seguenti valori x = 283,5, y = 295,5, z = 354 e come normale del foro Ox = 283,5, Oy = 295,5, Oz = 354, Vx = 2,957, Vy = 0,118, Vz = -4,848.

Visualizzando alcuni di questi risultati graficamente si mettono in evidenza quelli del test con risoluzione 400 x 400.

La prima immagine raffigura i punti identificati come nell'intorno dei bordi con dei cubi blu mentre il cubo viola il centro del foro. Come si può notare la selezione dei punti del bordo è molto precisa e ciò ci porta ad ottenere un'identificazione anche del centro ottima.

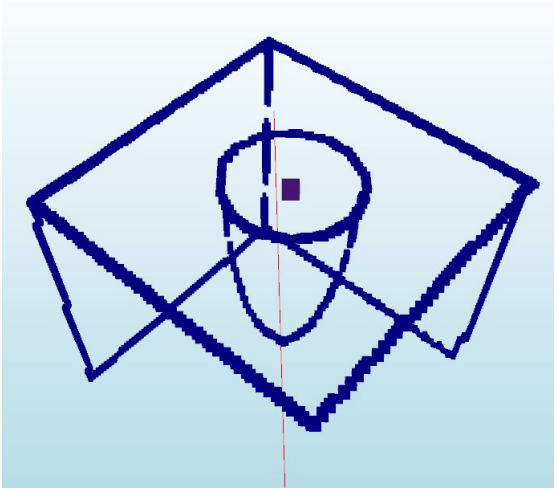


Figura 5.34: Test 1 cubo con foro

Nella seconda immagine si possono vedere sia il centro del foro sia la normale identificate correttamente.

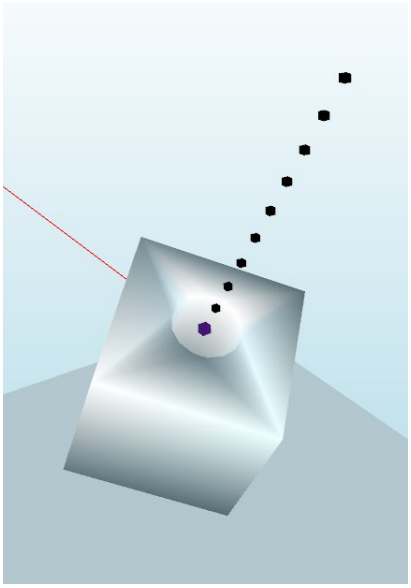


Figura 5.35: Test 2 cubo con foro

Nella seguente immagine si è voluto mettere in risalto un aspetto molto particolare. Ognuno dei cubi raffigurati rappresenta un punto di quelli selezionati per il calcolo della normale. Ovviamente più l'immagine ha una buona risoluzione, più i bordi dell'oggetto vengono identificati correttamente, più preciso avviene il voto per il centro del foro e più precisa avviene questa fase di selezione dei punti che poi vengono utilizzati per trovare la normale utilizzando Ransac. Infatti nell'ultimo test

con risoluzione 400 x 400 si è ottenuto un buon risultato anche in questa fase in quanto i punti esterni al cerchio del foro sono poco numerosi ed in questo modo non influenzano considerevolmente il corretto calcolo della normale.

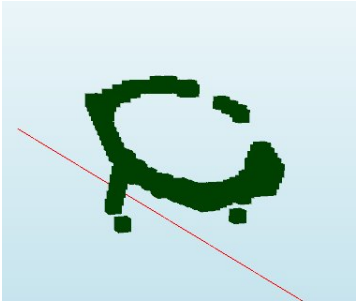


Figura 5.36: Test 3 cubo con foro

5.4.7 Quattro cubi con foro

Nella reale applicazione di questo algoritmo molto raramente ci si troverà ad analizzare immagini con un singolo oggetto ma invece si analizzeranno immagini composte da più oggetti. Per effettuare un test più complesso è stato deciso di inserire 4 cubi con foro utilizzati nel test precedente. I dati obiettivo di questo test sono riassunti nel seguente elenco.

1. numero fori 4,
2. diametro fori 12,
3. centro foro 1 : x = 363,5, y = 285,5, z = 355,5,
4. centro foro 2 : x = 312,00, y = 215, z = 313,
5. centro foro 3 : x = 264,50, y = 299, z = 297,
6. centro foro 4 : x = 322,00, y = 355,5, z = 334,5,
7. normale foro 1 : x = 8,932, y = -45,326, z = 91,827,
8. normale foro 2 : x = -58,437, y = -17.546, z = 16.321,
9. normale foro 3 : x = -238,274, y = -135,478, z = -184,882,
10. normale foro 4 : x = 189,174, y = 66,168, z = -158,738.

Il primo test è stato effettuato con una risoluzione di 100 x 100 ed ha portato i seguenti risultati.

Centri dei fori.

id foro	X	Y	Z	voti
1	322,50	355,5	334	30
2	363,50	285	356,5	29
3	312,00	215,5	313	29
4	266,50	300	300	26

Mentre per le normali.

id normale	X	Y	Z
1	-18.175	-6.869	34.647
2	-0.001	43.978	-164.130
3	-58,437	-17.546	16.321
4	-162.733	-87.208	-281.859

Il secondo test è stato effettuato con una risoluzione maggiore, pari a 140 x 140, ed ha portato ai seguenti risultati.

Centri dei fori.

id foro	X	Y	Z	voti
1	363,5	285,5	355,5	54
2	312,0	215	313	51
3	264,5	299	297	46
4	322,0	355,5	334,5	43

Mentre per le normali.

id normale	X	Y	Z
1	8,9	-45,6	91,8
2	-58,4	-17.6	16.3
3	-238,3	-135,5	-184,9
4	189,2	66,2	-158,7

Come si può notare i valori dei centri e delle normali risultano sufficientemente precisi. Ora si può passare alla rappresentazione grafica di tali valori per rendere meglio l'idea della correttezza del sistema.

La prima immagine rappresenta la selezione dei punti di bordo con risoluzione 100 x 100 mentre la seconda rappresenta quelli con la risoluzione 140 x 140.

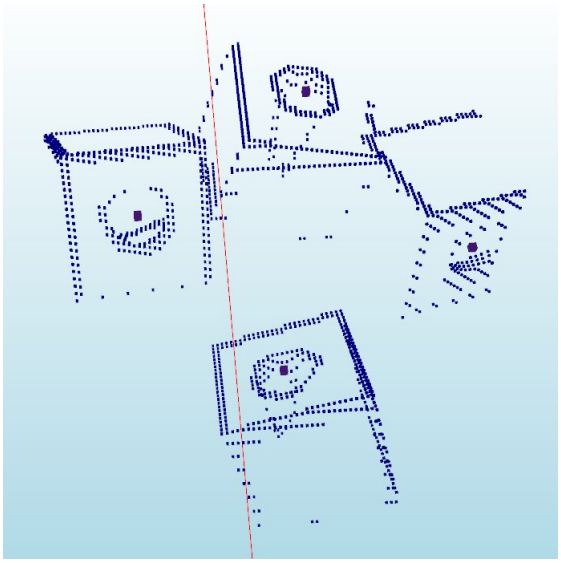


Figura 5.37: Test 4 cubo con foro, 100 x 100

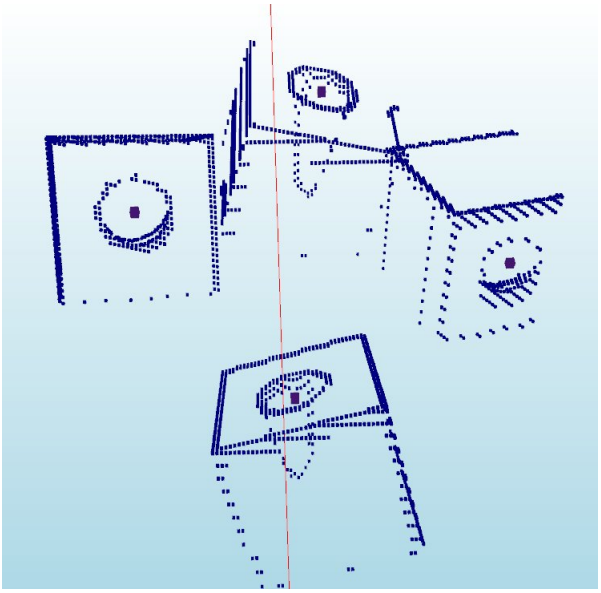


Figura 5.38: Test 5 cubo con foro, 140 x 140

Come si può notare i bordi sono ben definiti ed i centri dei fori sono identificati in modo corretto. Passando alle successive immagini, rappresentano l'assieme degli oggetti con le relative normali.

Come detto in precedenza, in diverse occasioni, l'algoritmo funziona correttamente se si ha una risoluzione dell'immagine sufficientemente accurata ed un'impostazione

dei parametri corretta. I Parametri possono variare a seconda dell'oggetto e della risoluzione dell'immagine. Infatti come si può notare nella prima immagine, i centri dei fori sono stati identificati correttamente mentre per quanto riguarda le normali, una di esse risulta essere errata.

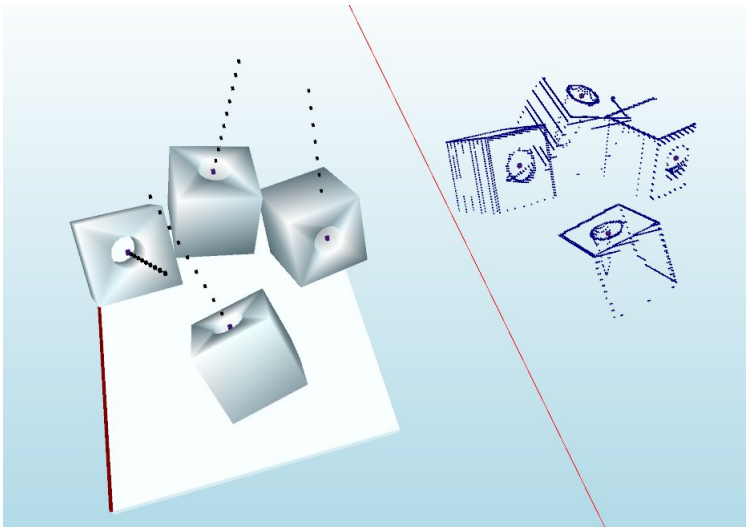


Figura 5.39: Test 6 cubo con foro

In questo caso il problema è dovuto al fatto che il foro in esame è in una posizione molto verticale. Questo problema è stato risolto grazie ad una accurata variazione dei parametri di Ransac. Effettuata tale operazione si sono ottenuti i seguenti risultati con tutte e 4 le normali corrette.

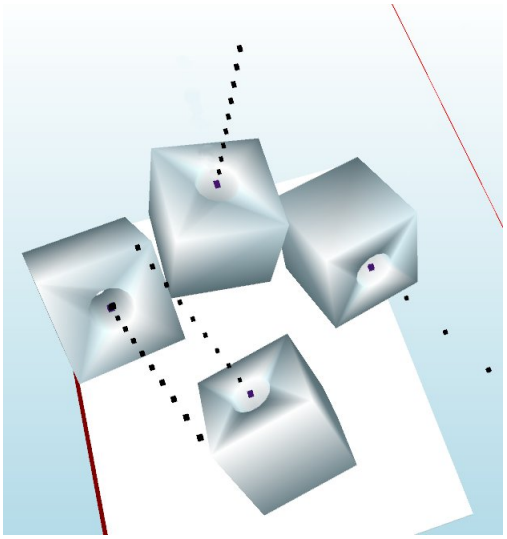


Figura 5.40: Test 7 cubo con foro

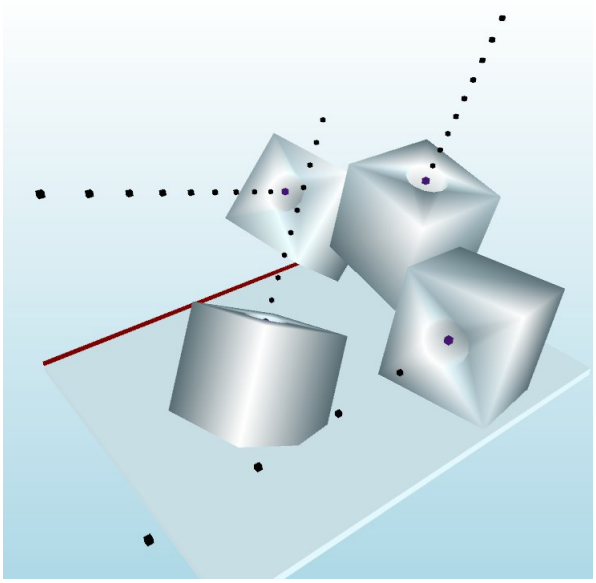


Figura 5.41: Test 8 cubo con foro

5.4.8 Oggetto assemblato da 3 tubi

Un altro oggetto utilizzato in questi test consiste in un raccordo tubolare formato da un assemblaggio di tubi di diametro 4 mm. Questo presenta esattamente 5 estremità del tubo che possono essere identificate sempre con l'algoritmo dei fori. Qui di seguito è raffigurato l'oggetto in esame.

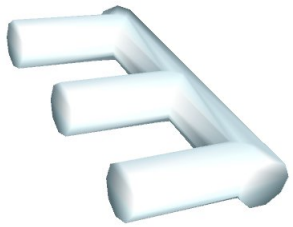


Figura 5.42: Test 1 tubi

Per questo test è stato deciso di partire direttamente con risoluzioni elevate dato che l'oggetto è abbastanza di piccole dimensioni e complesso. Dunque si è deciso di partire direttamente con una risoluzione di 600 x 600. Con tale test si sono ottenuti i seguenti valori.

Centri dei fori.

id foro	X	Y	Z	voti
1	311,5	329	308,5	865
2	298,5	311	300	851
3	286,5	292,5	292	842
4	300	346,5	391,5	785

Mentre per le normali.

id normale	X	Y	Z
1	-5,00	5,43	-2,65
2	-4,9	5,39	-2,62
3	-5,1	5,5	-2,68
4	-14,66	-20,61	-10,86

Visualizzando graficamente tali risultati si hanno le seguenti immagini.

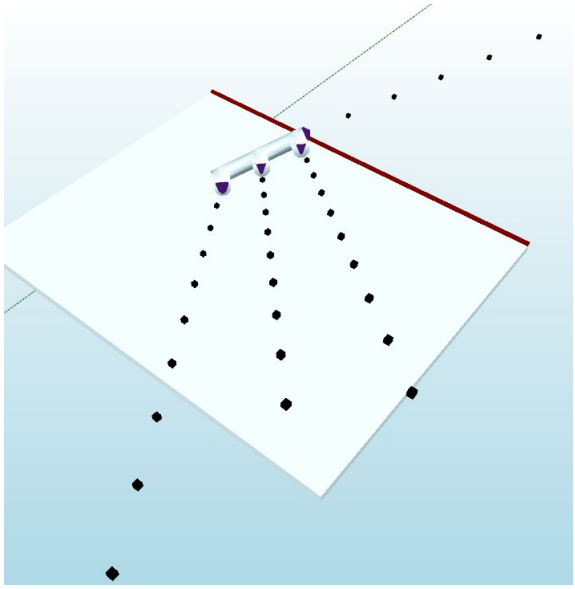


Figura 5.43: Test 2 tubi

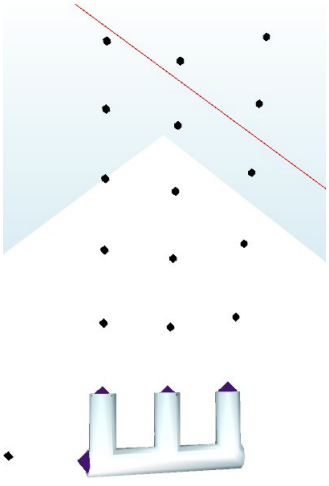


Figura 5.44: Test 3 tubi

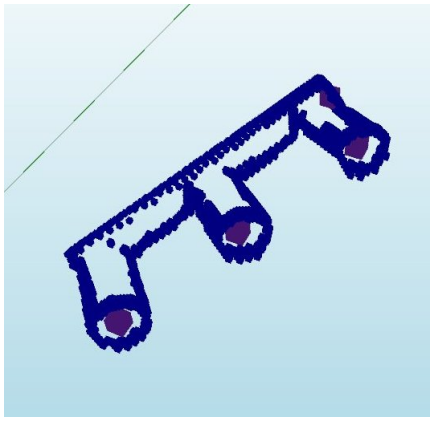


Figura 5.45: Test 4 tubi

Anche in questo caso si sono eseguite delle variazioni nei parametri dell'algoritmo per ottenere i valori prima elencati. Come si può notare dalle immagini questi rispecchiano pienamente gli obiettivi desiderati e confermano nuovamente la buona riuscita dell'algoritmo.

6

Conclusioni

Tutti gli obiettivi prefissati per questo lavoro di tesi sono stati raggiunti in modo ottimale. Lungo il percorso sono state effettuate delle scelte fondamentali per il progetto che dopo analisi e test si sono rivelate molto promettenti. Il sistema sviluppato permette di eseguire la scansione simulata di qualunque oggetto importabile come se fosse stata eseguita dal Ruler Sick. Inoltre tale scansione può essere personalizzata a piacere variando le parametrizzazioni che sono state richieste all'inizio del progetto. Passando alla fase successiva il sistema identifica correttamente le features desiderate per il riconoscimento dell'oggetto. Va precisato che tale attività è stata ampiamente testata ed è stato confermato che l'algoritmo dà risultati ottimi solo se l'immagine ha una risoluzione sufficientemente elevata e vi è una corretta configurazione dei parametri dell'algoritmo. Queste caratteristiche vanno configurate a seconda della dimensione e della conformazione dell'oggetto in esame. Una volta settate, il sistema funziona correttamente con qualunque numero e posizione di oggetti presi in esame. Perciò tale operazione di settaggio va effettuata solo nella fase iniziale di installazione.

Alla luce di ciò, si può affermare che le sperimentazioni eseguite hanno dato ragione alle scelte effettuate.

Il successo di tale progetto porta a numerosi vantaggi tra cui :

1. Possibilità di effettuare qualsiasi test senza dover arrestare la macchina,
2. Possibilità di effettuare qualsiasi test senza doversi recare presso il cliente o di farsi spedire l'oggetto fisico desiderato,
3. Possibilità di testare collisioni tra robot ed ambiente esterno senza rischi,
4. Possibilità di visualizzare all'interno del simulatore l'immagine creata dal Ruler,
5. Avendo l'immagine all'interno del simulatore si possono identificare eventuali anomalie della scansione in modo molto agevole,

6. Possibilità di effettuare la scansione di un oggetto che non si possiede fisicamente,

7. Possibilità di fare un numero di test a piacere, variando specifiche dell'oggetto, a costo zero ed in qualsiasi momento,

Sviluppi futuri

Avendo a disposizione altro tempo da poter dedicare a questo progetto si potrebbero migliorare i seguenti aspetti :

1. Il primo ed il più importante consiste nella rilevazione delle occlusioni. Ovvero il sistema deve essere in grado di rilevare se più oggetti possono avere delle parti sovrapposte. Ipoteticamente si potrebbero rilevare correttamente tutte le features ricercate su più oggetti, ma questi potrebbero essere parzialmente accavallati. La conseguenza potrebbe essere la collisione tra oggetti ed il robot.

2. Creare una metodologia con la quale il sistema decida autonomamente quali siano le features cruciali da identificare per il riconoscimento di un generico oggetto. Questa attività risulta essere molto complessa ma porterebbe il sistema ad essere totalmente generico. Va sottolineato che al fine dell'applicazione per cui è stato ideato lo sviluppo di questa attività non è essenziale in quanto solitamente il numero di oggetti con cui il robot va a lavorare è limitato, massimo 10. Perciò non ha senso investire risorse in questo campo ma se vi fosse la necessità o se fosse ideata una nuova applicazione dell'intero sistema questo può essere uno dei punti di sviluppo.

3. Come in tutti gli ambiti software un altro punto su cui si può lavorare è la velocità del sistema, infatti nell'intero progetto non è mai stata analizzata con precisione perché lo scopo principale consisteva nel verificare che il tutto funzionasse correttamente. Ma quando verrà usata in un'applicazione con dei tempi da rispettare probabilmente dovranno essere introdotti dei miglioramenti computazionali.

4. Se richiesto potrebbe essere modificato anche il sistema di acquisizione. Attualmente si è puntato a simulare il comportamento del Ruler Sick ma il sistema potrebbe essere ampliato introducendo la possibilità di effettuare la scansione con altri sistemi, ad esempio l'introduzione di un altro Laser, di un'altra Camera o di altre modifiche desiderate. Si potrebbe voler utilizzare altri sistemi per la scansione per evitare eventuali problematiche, per esempio con l'utilizzo di 2 telecamere si evita il sotto-squadro oppure si migliorerebbe l'affidabilità se il sistema stesse fermo.

Bibliografia

[1] Boguslaw Cyganek, J. Paul Siebert, *An Introduction to 3D Computer Vision Techniques and Algorithms*, edizione 2009, Wiley

[2] Sick IVP, *Ranger and Ruler Technical Workbook*

[3] Carlo Federico Zoffoli, *Corso di DirectX con C#*

Siti internet consultati

[4] Sito aziendale di Euclid Labs, *www.euclidlabs.it*

[5] Sito aziendale di Scape technologie, *www.scapetechnologies.com*

[6] Sito aziendale di 3ideo, *www.3ideo.com*

[7] Sito aziendale di alma, *www.almacam.com*

[8] Wikipedia, *www.wikipedia.com*

Rette e piani nel 3D

[9] A84CEED5d01.pdf, *Geometria in R3*

[10] Equations of Lines and Planes, *http://www.math.oregonstate.edu/home /programs/undergrad/CalculusQuestStudyGuides/vcalc/lineplane/lineplane.html*, 1996 Department of Mathematics, Oregon State University.

[11] Mario Boella, *http://areeweb.polito.it/didattica/polymath/htmlS/argoment/ CABRI/Cabri_Giu08/MinDistSghembe.htm*, Politecnico di Torino.

[12] ALeXio, Rette e piani nello spazio *http://al3xios.altervista.org/ReTTe_e_PiaNi %203D.htm*

[13] Coordinate curvilinee, cilindriche, sferiche *http://www.elettra2000.it/vdegliesposti /Dispense%20Propagazione/Coordinate%20curvilinee.pdf*

[14] Le cupole geodetiche *http://www.mat.uniroma3.it/users/magrone/2008_09/ magistrale/cupolegeodetiche.pdf*

[15] *Equazioni parametriche e cartesiane di sottospazi affini di Rn*, 2006, Trapani

[16] Garroni, Rette e piani nello spazio *http://www.mat.uniroma1.it/people/garroni/pdf/Lezione7.pdf*, uniroma

[17] Garroni, Prodotto vettoriale e applicazioni *http://www.mat.uniroma1.it/people/garroni/pdf/Lezione4.pdf*, uniroma

[18] Fabrizio Pucci, Matteo Salleolini, *Tesi di laurea : Coperture di grandi luce in legno lamellare: progetto strutturale con gli eurocodici della copertura di un palazzetto dello sport polivalente*, 2007/2008 università di firenze

[19] Marco Debernardi, *http://digilander.libero.it/marcodebe79/Geom1A/Spazio.pdf*

Hough

[20] Ghassan Hamarneh, Karin Althoff, Rafeef Abu-Gharbieh, *Automatic Line Detection*, 1999, Image Analysis Group Department of Signals and Systems Chalmers University of Technology.

[21] Kourosh Khoshelham, *Extending generalized Hough Transform to Detect 3D objects in laser range data*, 2007, University of Technology, Finland.

[22] Michael Miska Kazhdan, *http://www.cs.jhu.edu/ misha/Fall04/GHT1.pdf*, Baltimore.

[23] Ballard, *Generalizing the Hough transform to detect arbitrary shapes*, 1981.

[24] Duda, R.O. and Hart, *Use of the Hough transformation to detect lines and curves in pictures.*, 1972.

[25] Titus Zaharia e Françoise Prêteux, *Hough transform-based 3D mesh retrieval*,Institut National des Télécommunications, Evry Cedex.

[26] Haiyuan WU, Genki YOSHIKAWA, Tadayoshi SHIOYAMA, Shihong LAO and Masato KAWADE, *Glasses Frame Detection with 3D Hough Transform*,Dept. of Mechanical and System Engineering, Kyoto Institute of Technology, Matsugasaki, Sakyo-ku, Kyoto, Japan.

[27] S. Arcelli, *Metodi di riconoscimento in fisica subnucleare*, 2010

Stereovisione

[28] 6.869 Advances in Computer Vision, *http://people.csail.mit.edu/torralba/courses/6.869/6.869.computervision.htm*, 2010

[29] 6.869 Advances in Computer Vision, *Lecture 16 3D*, 2010

[30] Stefano Mattoccia, *Introduzione alla Visione Stereo*, Università di Bologna

Ransac

[31] Marco Zuliani, *RANSAC for Dummies*, agosto 2011

[32] Michael Ying Yang, Wolfgang Forstner, *Plane Detection in Point Cloud Data*, Institute of Geodesy and Geoinformation University of Bonn

[33] V´aclav Hlav´a, *RANSAC*, Czech Technical University, Faculty of Electrical Engineering Department of Cybernetics, Center for Machine Perception Praha, Karlovo, Czech Republic

[34] Konstantinos G. Derpanis, *Overview of the RANSAC Algorithm*, 2010

Line Detection

[35] Christian Bahnisch, Peer Stelldinger, and Ullrich Kothe, *Fast and Accurate 3D Edge Detection for Surface Reconstruction*, Hamburg, Germany.

[36] Canny, J, *A computational approach to edge detection*, 1986, TPAMI.

[37] Monga, O., Deriche, R., Rocchisani, J, *3d edge detection using recursive filtering*, 1991, application to scanner images. CVGIP: Image Underst.

[38] Dr. George Bebis, Line Detection*http://www.cse.unr.edu/ bebis/CS791E/Notes/LineDetection.pdf*

Bin Picking

[39] Nitesh Shroff, Yuichi Taguchi, Oncel Tuzel, Ashok Veeraraghavan, Srikumar Ramalingam, and Haruhisa Okuda *Finding a Needle in a Specular Haystack*, University of Maryland, Mitsubishi Electric Research Labs, Mitsubishi Electric Corporation

[40] A. Agrawal, Y. Sun, J. Barnwell, and R. Raskar *Visionguided Robot System for Picking Objects by Casting Shadows*, The International Journal of Robotics Research, 2010.

[41] R. Bolles and P. Horaud *A three-dimensional part orientation system*, The International Journal of Robotics Research, 1986.

[42] http://en.wikipedia.org/wiki/3dimensional_matching.

Fitting Algorithm

[43] Formal Statement of the FHT Plane Fitting Algorithm *http://gandalf-library.sourceforge.net/tutorial/report/node142.html*

[44] Michael A. Burr, Alan C. Cheng, Ryan G. Coleman, Diane L. Souvaine *Transformations and Algorithms for Least Sum of Squares Hypersphere Fitting*, 16th Canadian Conference on Computational Geometry, 2004.

[45] Jan Klein, University of Paderborn, Germany e Gabriel Zachmann University of Bonn, Germany *Interpolation Search for Point Cloud Intersection*

[46] Craig M. Shakarji, *Least-Squares Fitting Algorithms of the NIST Algorithm Testing System*, 1998, Journal of Research of the National Institute of Standards and Technology.

[47] Radu Bogdan Rusu and Steve Cousins, *3D is here: Point Cloud Library (PCL)*,Willow Garage USA.

[48] *http://www.udel.edu/HNES/HESC427/Sphere%20Fitting/LeastSquares.pdf*

[49] Jared Glover Computer Science and Artificial Intelligence Laboratory Massachusetts Institute of Technology Cambridge, Gary Bradski and Radu Bogdan Rusu Willow Garage Menlo Park *Monte Carlo Pose Estimation with Quaternion Kernels and the Bingham Distribution.*

[50] Radu Bogdan Rusu, Nico Blodow, Michael Beetz, *Fast Point Feature Histograms (FPFH) for 3D Registration*, Intelligent Autonomous Systems, Technische Universitat Munchen.

[51] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, Michael Beetz, *Aligning Point Cloud Views using Persistent Feature Histograms*, Intelligent Autonomous Systems, Technische Universitat Munchen.

[52] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, Michael Beetz, *Towards 3D Point Cloud Based Object Maps for Household Environments*, Technische Universitat Munchen, Computer Science Department, Intelligent Autonomous Systems group.

[53] Brian JONES, Michel AOUN, *Learning 3D Point Cloud Histograms*, December 11, 2009.

[54] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow and Michael Beetz, *Persistent Point Feature Histograms for 3D Point Clouds*, Intelligent Autonomous Systems, Technische Universität München.