



UNIVERSITÀ DEGLI STUDI DI FIRENZE
SCUOLA DI INGEGNERIA - DIPARTIMENTO DI INGEGNERIA
DELL'INFORMAZIONE

Tesi di Laurea Triennale in Ingegneria Informatica

**METODI DI OTTIMIZZAZIONE PER PROBLEMI DI
REGISTRAZIONE DI POINT CLOUD**

**OPTIMIZATION METHODS FOR POINT CLOUD
REGISTRATION PROBLEMS**

Candidato
Athos Innocenti

Relatore
Prof. Fabio Tardella

Correlatore
Prof. Marco Sciandrone

Anno Accademico 2020/2021

A Claudia

Indice

Introduzione	i
1 Il problema della registrazione	1
1.1 Tipologie di registrazione	3
1.1.1 Classificazione basata sulle sorgenti	4
1.1.2 Classificazione basata sulla trasformazione	6
1.2 Modello del problema	7
2 L'algoritmo ICP	10
2.1 Versioni di ICP	12
2.1.1 Point to point	12
2.1.2 Point to plane	13
2.1.3 Non linear ICP	15
2.1.4 Generalized ICP	16
3 Analisi delle prestazioni	18
3.1 Modelli e massimo numero di iterazioni	20
3.2 Analisi	23
3.2.1 Sovrapposizione dei baricentri	26
3.2.2 Deformazione	28
3.2.3 Rumore	33

4 Conclusioni	38
4.1 Prospettive future	38
Bibliografia	40
Ringraziamenti	42

Introduzione

Nell'ultimo ventennio, grazie al proficuo sviluppo e alla conseguente immissione sul mercato di tecniche e strumentazioni per l'acquisizione di immagini e video in tempo reale, si è assistito ad una considerevole crescita della visione computazionale che trova applicazione in un numero sempre più grande di ambiti tra i quali, per esempio, la robotica e la guida autonoma.

In questo continuo crescendo, un ruolo cardine nell'evoluzione della visione computazionale è ricoperto dai sensori di acquisizione e dalle strutture dati in cui i dati acquisiti vengono immagazzinati. A tal proposito, se da un lato si riscontra una sempre maggior eterogeneità nelle caratteristiche dei sensori, dall'altro la *point cloud* è diventata il più importante formato per rappresentare il mondo tridimensionale.

Tra la sue varie declinazioni, uno degli aspetti più interessanti della visione computazionale è sicuramente il *problema della registrazione di nuvole di punti* nel quale si vuole determinare la trasformazione spaziale che meglio allinea e sovrappone due o più immagini, generalmente tridimensionali, acquisite da sensori spesso diversi tra loro e in istanti temporali diversi, al fine di combinare insieme tali immagini per ottenere una mappa globale di un ambiente o di un oggetto.

Obiettivo di questa tesi è di analizzare uno dei più importanti algoritmi di registrazione: l'algoritmo *Iterative Closest Point*. In questa ottica è stata

condotta un'analisi delle prestazioni di ICP prendendo come riferimento molteplici possibili applicazioni dello stesso, che implicano proprietà diverse delle immagini fornitegli in ingresso. Partendo da un caso ideale in cui è garantita la perfetta sovrapposizione, si è proseguito avvicinandosi sempre di più a contesti reali, considerando immagini deformate o che presentano rumore e outliers, così da essere in grado di valutare anche l'efficacia di eventuali accorgimenti da affiancare all'algoritmo stesso.

Capitolo 1

Il problema della registrazione

"Allora portavi la tua cenere al monte; oggi vuoi portare il tuo fuoco nelle valli?"

Friedrich Nietzsche

La registrazione di *point cloud* (anche detta nuvola di punti) è uno dei problemi fondamentali della visione computazionale e i metodi di registrazione sono comunemente utilizzati in innumerevoli campi tra cui la robotica, l'astrofotografia e la medicina nonché in altrettante applicazioni tra cui figurano il riconoscimento e la mappatura di oggetti all'interno di un ambiente esteso, la localizzazione spaziale di oggetti e lo scanning tridimensionale. La localizzazione, per esempio, è un aspetto cruciale in un ambito come può essere quello della robotica in cui è fondamentale poter costantemente stimare la posizione di un agente all'interno di un ambiente così da poter determinare, tramite un opportuno processo decisionale, l'interazione che dev'essere compiuta dall'agente autonomo con l'ambiente circostante. Un ulteriore esempio può essere la ricostruzione tridimensionale di una scena, utile sia in ambito di

visione computazionale pura sia in altri ambiti più concreti come la guida autonoma o il monitoraggio in tempo reale di luoghi d'interesse. Inoltre, capita spesso che le nuvole di punti vengano acquisite in istanti temporali diversi utilizzando eventualmente strumentazioni diverse con diverse caratteristiche (e.g. campo visivo, risoluzione, scala metrica) per cui, volendo ricavare la ricostruzione completa di una mappa tridimensionale dell'ambiente in cui si trovano collocati i dispositivi di acquisizione, diventa necessario allinearle tra loro così da poterle combinare insieme arrivando ad ottenere il risultato finale desiderato. Tutto questo processo prende appunto il nome di *registrazione*.

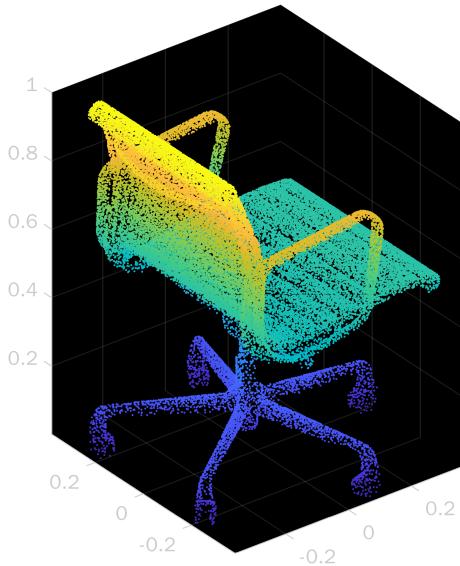


Figura 1.1: Esempio di point cloud

L'obiettivo comune dei metodi di registrazione si può dire essere quello di determinare la posizione di una point cloud rispetto ad un punto di riferimento spaziale e conseguentemente la posizione reciproca tra due diverse nuvole di punti, generalmente tridimensionali, così da poter dare una stima di una trasformazione geometrica ottimale (una matrice di trasformazione) che permetta di mappare le due point clouds e dunque permetta la loro corretta

sovraposizione commettendo il minor errore possibile al netto di eventuali rumori dovuti alla strumentazione.

1.1 Tipologie di registrazione

Le moderne tecnologie hanno permesso un rapido sviluppo di dispositivi per l'acquisizione di immagini tridimensionali che ha portato all'immissione sul mercato di sensori e videocamere ad elevata precisione con caratteristiche anche molto diverse tra loro, ognuno con i propri pregi e difetti, come per esempio i sensori LiDAR e Kinect. Questi ultimi sono in grado di generare una nuvola di punti strutturata e campionata su una griglia regolare, altri sensori invece producono una nuvola di punti priva di una struttura di base. Parallelamente a questo fruttifero sviluppo di nuovi sensori, la point cloud è diventata il formato di riferimento per la rappresentazione di ambienti 3D.

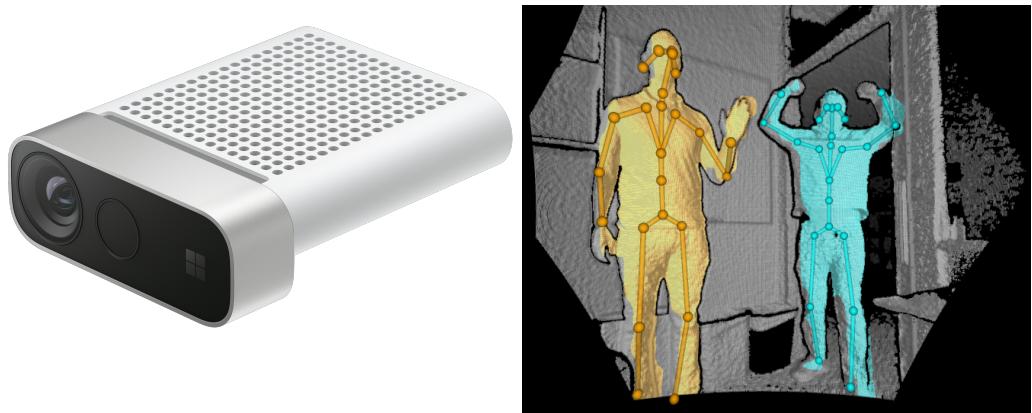


Figura 1.2: Microsoft Azure Kinect ed esempio di body tracking

Poiché i vari dispositivi hanno un campo visivo limitato non sono in grado di scansionare un ambiente o un oggetto nella sua totalità; un algoritmo di

registrazione diventa quindi fondamentale per poter ricavare la scansione di una scena più ampia.

1.1.1 Classificazione basata sulle sorgenti

Oltre alle limitazioni intrinseche dei sensori, e come già accennato nella sezione precedente, l'acquisizione di immagini di un ambiente o di un oggetto può avvenire in tempi diversi ma soprattutto adoperando dispositivi differenti e questo introduce una serie di ulteriori accorgimenti di cui tenere conto durante il processo stesso di registrazione. A tal proposito, data questa possibile eterogeneità nelle sorgenti delle point clouds è possibile suddividere gli algoritmi di registrazione in due classi: quelli in cui le point clouds sono ricavate dallo stesso tipo di sensori (pur eventualmente in tempi e da prospettive diverse) e quelli in cui le nuvole sono acquisite tramite sensori diversi.

Per quanto concerne la prima classe di algoritmi, comunemente indicata col termine *same-source*, i problemi che devono essere gestiti in fase di registrazione sono:

1. **Rumore e outliers** L'ambiente e il rumore legato ai sensori possono variare nel tempo e le diverse nuvole di punti possono presentare diversi rumori e outliers in corrispondenza dei medesimi punti
2. **Sovrapposizione parziale** A causa dei diversi punti spaziali di acquisizione e ad eventuali discontinuità temporali, le nuvole sono solo parzialmente sovrapposte

Nella seconda classe, nota come *cross-source*, l'impiego di sensori eterogenei rende ulteriormente più complessa la registrazione, introducendo problematiche addizionali di cui tener conto:

1. **Rumore e outliers**
2. **Sovrapposizione parziale**
3. **Differenza di densità** Conseguentemente all’impiego di sensori diversi con diverse risoluzioni e diversi meccanismi di acquisizione dell’immagine, le nuove di punti ottenute generalmente differiscono in termini di densità di punti
4. **Variazione di scala** Poiché meccanismi di acquisizione diversi possono adottare metriche diverse, le nuvole di punti possono avere scale diverse

Nonostante la mole di aspetti di cui dover tener conto, molteplici evidenze [10],[4] hanno mostrato che la fusione di nuvole di punti provenienti da sensori diversi può fornire maggiori informazioni e garantire prestazioni migliori in svariate applicazioni pratiche.

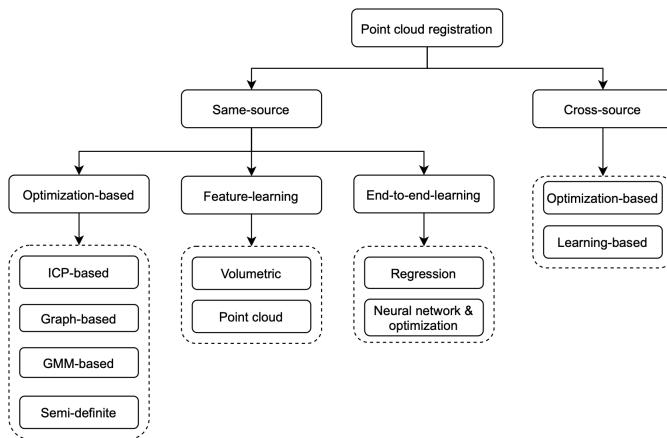


Figura 1.3: Classificazione dei metodi di registrazione

Limitandosi a prendere in analisi i metodi di registrazione di tipo same-source, [3] propone una loro ulteriore suddivisione in tre sottoclassi: *Optimization based*, *Feature learning* ed *End to end learning*. Ai fini di questo lavoro

di tesi è di particolare interesse focalizzarsi sulla prima citata in quanto ne fa parte l'algoritmo ICP che costituisce lo scheletro di questa tesi. I metodi di registrazione basati su ottimizzazione prevedono l'impiego di strategie di ottimizzazione per stimare la matrice di trasformazione. Gran parte di questi metodi prevede una prima fase di ricerca delle corrispondenze (per trovare le coppie di punti corrispondenti tra due diverse point clouds) seguita da una stima della trasformazione che sfrutta le corrispondenze trovate. Durante l'esecuzione iterativa di queste fasi, le corrispondenze dei punti diventano mano a mano più accurate il che implica la definizione di una trasformazione sempre più precisa. I vantaggi di questa classe sono: 1) la teoria matematica che sta alla base di questi algoritmi può garantire la loro convergenza 2) non necessitano di dati di allenamento e gestiscono discretamente scenari sconosciuti. Il principale svantaggio è che richiedono strategie più o meno sofisticate per poter gestire rumore, outliers, variazione di densità e sovrapposizione parziale che aumentano il costo computazionale di questi algoritmi.

1.1.2 Classificazione basata sulla trasformazione

Una seconda possibile distinzione tra gli algoritmi di registrazione può essere fatta considerando la trasformazione prodotta. Da questo punto di vista si distinguono gli algoritmi che permettono di definire una trasformazione rigida e quelli che permettono di definirne una non rigida. I primi stimano una trasformazione omogenea che può essere modellata utilizzando 6 gradi di libertà e che è composta da una matrice di rotazione e da un vettore di traslazione. I secondi invece permettono un maggior numero di gradi di libertà in quanto la trasformazione ottenuta è espressa tramite una funzione non lineare che permette di gestire oggetti più complessi, deformati o che addirittura possono mutare forma nel tempo.

1.2 Modello del problema

La maggior parte degli algoritmi di registrazione sono formulati in modo da andare a minimizzare l'errore commesso tramite due fasi: la prima fase consiste nella ricerca delle corrispondenze tra i punti che compongono le due nuvole, la seconda nella stima della trasformazione. Queste due fasi vengono ripetute fino al raggiungimento dell'errore minimo o di un suo valore accettabile. Note le corrispondenze tra le due nuvole, la stima della trasformazione ammette una soluzione in forma chiusa.

Per poter arrivare a formulare un modello per il problema della registrazione è opportuno introdurre una serie di definizioni preliminari. Sia

$$\mathbb{X} = \{\mathbf{x}_i \in \mathbb{R}^3, i = 1, \dots, M\}$$

l'insieme degli M punti che definisce la prima point cloud (questa sarà la nuvola di punti mobile, quella soggetta alla trasformazione) e sia

$$\mathbb{Y} = \{\mathbf{y}_j \in \mathbb{R}^3, j = 1, \dots, N\}$$

l'insieme degli N punti che definisce la seconda (quella fissa, per generalità si suppone $N \neq M$). Le due nuvole possono essere rappresentate tramite due matrici $X \in \mathbb{R}^{M \times 3}$ e $Y \in \mathbb{R}^{N \times 3}$ delle quali si identificano con \mathbf{x}_i^\top e \mathbf{y}_j^\top i vettori riga corrispondenti alle coordinate cartesiane dell'i-esimo e j-esimo punto all'interno della rispettiva nuvola di punti. Si indica ora con $\mathbb{SO}(3)$ il gruppo delle rotazioni 3D, ovvero il gruppo di tutte le rotazioni rispetto all'origine dello spazio euclideo tridimensionale \mathbb{R}^3 sotto l'operazione di composizione. Le rotazioni sono trasformazioni lineari di \mathbb{R}^3 e questo implica che, scelta una base di tale spazio vettoriale, è possibile rappresentarle tramite matrici. In particolare, se si sceglie una base ortonormale, ogni rotazione può essere rappresentata tramite una matrice quadrata $R \in M(3, \mathbb{R})$ che è una matrice

ortogonale, ovvero è una matrice invertibile la cui inversa coincide con la sua trasposta, il che implica due importanti proprietà:

$$1. \quad RR^\top = I$$

$$2. \quad \det(R) = 1$$

Queste matrici sono note con il nome di *special orthogonal matrices* che spiega la notazione adottata. Tale gruppo viene utilizzato per descrivere le possibili simmetrie rotazionali di un oggetto e le sue possibili orientazioni nello spazio. Sia ora $\phi(., \omega) : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ una funzione di trasformazione dipendente da un vettore di parametri $\omega \in \mathbb{R}^m$ che associa ogni punto $\mathbf{x}_i \in \mathbb{X}$ con uno e un solo punto $\mathbf{y}_j \in \mathbb{Y}$. Questa funzione può rappresentare due diverse tipologie di trasformazione:

Trasformazione rigida la funzione di trasformazione $\phi(., \omega)$ è del tipo:

$$\phi(\mathbf{x}, \omega) = \phi(\mathbf{x}, R, \mathbf{T}) = R\mathbf{x} + \mathbf{T}$$

dove $R \in \mathbb{SO}(3)$ è una matrice di rotazione tridimensionale e $\mathbf{T} \in \mathbb{R}^3$ è un vettore di traslazione

Trasformazione non rigida $\phi(., \omega)$ è una funzione non lineare

Infine, nel caso più generale non sono note a priori le corrispondenze dei punti tra le due point clouds. Di conseguenza, si introducono le variabili binarie δ_{ij} con $i = 1, \dots, M$ e $j = 1, \dots, N$ tali che:

$$\delta_{ij} = \begin{cases} 1 & \text{se } i \rightarrow j \\ 0 & \text{altrimenti} \end{cases}$$

ovvero: data una coppia di punti $\mathbf{x}_i \in \mathbb{X}$ e $\mathbf{y}_j \in \mathbb{Y}$, $\delta_{ij} = 1$ se e solo se il punto \mathbf{x}_i viene mappato nel punto \mathbf{y}_j tramite la funzione di trasformazione.

Date queste premesse, l'obiettivo della registrazione è quello di trovare il vettore di parametri $\boldsymbol{\omega} \in \mathbb{R}^m$ che meglio allinea le due nuvole di punti, ovvero si vuole determinare $\boldsymbol{\omega}$ in modo da minimizzare la somma delle distanze euclidiene tra i punti di \mathbb{X} e quelli di \mathbb{Y} (problema di ottimizzazione di *minimi quadrati*). Si può finalmente formulare il problema della registrazione di point cloud come il seguente problema di *programmazione mista*:

$$\begin{aligned} \min_{\delta, \boldsymbol{\omega}} \quad & \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^N \delta_{ij} \|\boldsymbol{\phi}(\mathbf{x}_i, \boldsymbol{\omega}) - \mathbf{y}_j\|^2 \\ \text{subject to} \quad & \sum_{j=1}^N \delta_{ij} = 1 \quad i = 1, \dots, M \\ & \delta_{ij} \in \{0, 1\} \quad i = 1, \dots, M \quad j = 1, \dots, N \end{aligned}$$

Oppure, in forma del tutto equivalente, come il seguente problema di *programmazione continua*:

$$\begin{aligned} \min_{\delta, \boldsymbol{\omega}} \quad & \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^N \delta_{ij} \|\boldsymbol{\phi}(\mathbf{x}_i, \boldsymbol{\omega}) - \mathbf{y}_j\|^2 \\ \text{subject to} \quad & \sum_{j=1}^N \delta_{ij} = 1 \quad i = 1, \dots, M \\ & \delta_{ij} \geq 0 \quad i = 1, \dots, M \quad j = 1, \dots, N \end{aligned}$$

Nel caso di una trasformazione rigida, detta R la matrice di rotazione e \mathbf{T} il vettore di traslazione, la funzione obiettivo del problema può essere riscritta nella forma:

$$\min_{\delta, R, \mathbf{T}} \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^N \delta_{ij} \|R\mathbf{x}_i + \mathbf{T} - \mathbf{y}_j\|^2$$

Capitolo 2

L'algoritmo ICP

"È impossibile che le cose non siano dove sono; poiché tutto è bene"

Voltaire

Gran parte della applicazioni di questi metodi prevedono l'impiego di una registrazione deterministica basata sulla *Singular Value Decomposition* (SVD) o sulla *Principal Component Analysis* (PCA) o ancora sfruttano schemi iterativi più complessi costruiti sull'algoritmo *Iterative Closest Point*, o ICP, del quale sono state proposte numerose varianti. Sia SVD che PCA sono metodi basati sulle matrici di covarianza e sulla matrice di cross-correlazione delle nuvole di punti, al contrario ICP è un algoritmo che iterativamente minimizza una funzione di costo basata sulla stima delle corrispondenze tra i punti delle due point clouds. Tali corrispondenze determinano la qualità con cui la trasformazione ottenuta sovrapporrà le due nuvole.

La PCA, frequentemente utilizzata in contesti come classificazione e compressione, permette di determinare la direzione di massima varianza corri-

spondente all'autovettore più grande della matrice di covarianza delle nuvole e la sua magnitudine è data dall'autovalore associato. Centrate le due nuvole di modo che le origini delle loro basi ortonormali coincidano, si calcolano le matrici di covarianza e i loro autovettori. La registrazione viene effettuata allineando le direzioni in cui le due point clouds hanno maggior variazione poiché tali direzioni rappresentano l'orientazione delle nuvole. Allineati gli autovettori, il centroide (baricentro) del target viene sommato a ciascuna coordinata trasformata così che i centri delle due nuvole corrispondano. Tuttavia, la PCA non minimizza la distanza euclidea tra le due nuvole di punti, è altamente sensibile agli outliers e funziona bene solo se ciascuna point cloud è approssimativamente distribuita normalmente.

La SVD, applicabile se già si conoscono le corrispondenze tra le due nuvole, costituisce un approccio più robusto poiché risolve un problema di minimi quadrati minimizzando direttamente la somma delle distanze euclidean. Date le corrispondenze, si può calcolare la matrice di cross-correlazione M delle tue point clouds centrate e la si può decomporre come $M = USV^\top$. Le colonne di U e V^\top sono dette *vettori singolari sinistri* e *destri*, gli elementi sulla diagonale della matrice S sono detti *valori singolari* e sono gli unici non nulli, in numero pari al rango di M . La SVD però non è generalmente unica. La soluzione ottimale del problema è definita tramite la matrice di rotazione

$$R_t^s = UV^\top$$

e il vettore di traslazione

$$\tilde{\mathbf{T}} = \tilde{\mathbf{c}_s} - R_t^s \tilde{\mathbf{c}_t}$$

L'algoritmo ICP riceve in ingresso le due point clouds che si desiderano registrare. Queste sono generalmente chiamate *source point cloud* (mobile) e *target point cloud* (fissa). La prima fase dell'algoritmo consiste nella definizione delle corrispondenze tra le due nuvole, questo procedimento il più delle

volte si basa su un approccio di tipo *nearest neighbor* anche se in alcune versioni di ICP vengono utilizzati schemi più complessi che sfruttano proprietà geometriche oppure informazioni associate al colore. A questo punto SVD può essere utilizzato per ottenere una stima iniziale della matrice di trasformazione affine che allinea le due nuvole di punti. Dopo la trasformazione, l'intero processo viene ripetuto rimuovendo gli outliers e ridefinendo le corrispondenze.

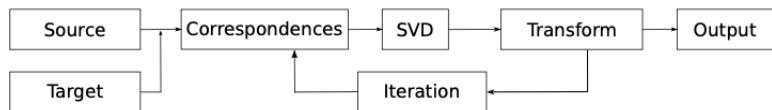


Figura 2.1: Schema di ICP

2.1 Versioni di ICP

Nel corso degli anni sono state proposte molteplici versioni dell'algoritmo ICP [1]. La maggior parte di queste differiscono nell'approccio adoperato per definire le corrispondenze tra i punti delle due diverse point clouds (e dunque nella metrica). Tra le più importanti figurano: *point to point*, *point to plane*, *plane to plane*, *non linear ICP* e *generalized ICP*.

2.1.1 Point to point

La versione point to point può essere considerata come quella più semplice nel senso che definisce le corrispondenze tra i punti, e quindi le coppie di punti più vicini tra loro, utilizzando la distanza tra le coordinate dei punti. Dato un generico punto \mathbf{p}_i nella source cloud identificato tramite il proprio raggio vettore, sfruttando la tecnica del nearest neighbor si cerca quel punto \mathbf{q}_j nella target cloud avente distanza minima da \mathbf{p}_i . In particolare, all'interno della

tecnica del nearest neighbor si utilizza la distanza euclidea tra due punti di modo che, fissato il punto \mathbf{p}_i , il punto \mathbf{q}_j nella target cloud a lui più vicino sarà quel punto avente indice \hat{j} che risolve:

$$\hat{j} = \arg \min_j \|\mathbf{p}_i - \mathbf{q}_j\|^2$$

per $j = 1, \dots, N$ con N il numero di punti nella target point cloud.

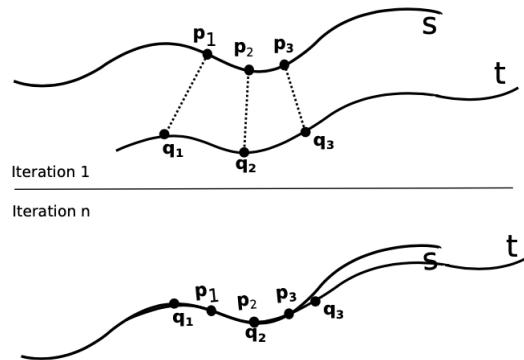


Figura 2.2: Funzionamento di ICP point-to-point

Procedendo in questo modo, la matrice di rotazione $R \in \mathbb{SO}(3)$ e il vettore di traslazione $\mathbf{T} \in \mathbb{R}^3$ sono stimati minimizzando il quadrato delle distanze tra le N coppie di punti corrispondenti ($p_k \in \text{Source}$, $q_k \in \text{Target}$):

$$\hat{R}, \hat{\mathbf{T}} = \arg \min_{R, \mathbf{T}} \sum_{k=1}^N \| (R\mathbf{p}_k + \mathbf{T}) - \mathbf{q}_k \|^2$$

Tuttavia questa versione, nonostante la sua semplicità nel definire le corrispondenze, è sensibile agli outliers. Ne sono state proposte ulteriori versioni con l'intento di migliorare la qualità delle corrispondenze trovate. A solo titolo di esempio se ne citano due: *EfficientVarICP* e *IMLP*.

2.1.2 Point to plane

Nella versione point to plane di ICP invece di determinare direttamente il punto più vicino all'interno della nuvola target si va a prendere quell'insieme

di punti nella target cloud che localmente sono i più vicini al punto candidato \mathbf{q}_j come corrispettivo con il punto \mathbf{p}_i della source cloud in modo da ridurre la sensibilità al rumore da parte dell'algoritmo. Per farlo, questa versione assume che l'insieme locale di punti vicini ad un altro punto sia complanare di modo che questa superficie locale di punti possa essere definita tramite il suo vettore normale \mathbf{n} , ottenuto come il più piccolo autovettore della matrice di covarianza dei punti che circondano il punto candidato \mathbf{q}_j nella target cloud. In questa versione si opera quindi lungo la normale alla superficie.

Il problema può quindi essere scritto nella forma:

$$\hat{R}, \hat{\mathbf{T}} = \arg \min_{R, \mathbf{T}} \sum_{k=1}^N \|[(R\mathbf{p}_k + \mathbf{T}) - \mathbf{q}_k]\mathbf{n}_k\|^2$$

con \mathbf{n}_k la normale alla superficie nel punto candidato \mathbf{q}_k . Dunque, invece di minimizzare direttamente la distanza euclidea tra i punti corrispondenti, si minimizza la proiezione scalare di questa distanza (anche detta distanza ortogonale) su un piano definito tramite il vettore normale \mathbf{n} .

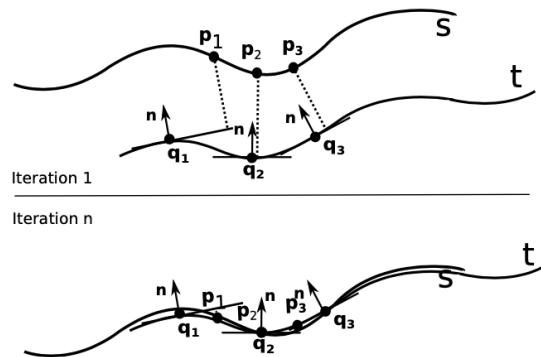


Figura 2.3: Funzionamento di ICP point-to-plane

Utilizzando come punto di partenza la versione point to plane è possibile definire un'ulteriore versione di ICP per stimare le corrispondenze, nota come plane to plane. Nella plane to plane la funzione obiettivo è simile a quella

adoperata per definire la metrica di distanza nella versione point to point ed è data da:

$$\hat{R}, \hat{\mathbf{T}} = \arg \min_{R, \mathbf{T}} \sum_{k=1}^N \| (R\mathbf{np}_k + \mathbf{T}) - \mathbf{nq}_k \|^2$$

dove \mathbf{np} e \mathbf{nq} sono le normali delle due nuvole di punti.

2.1.3 Non linear ICP

Sia la versione point to point che point to plane di ICP hanno una funzione obiettivo quadrata, convessa e differenziabile che comporta la definizione di un problema di ottimizzazione di minimi quadrati, meglio noto con il nome di *ottimizzazione L2*, risolvibile per via numerica tramite la SVD. Tuttavia questa classe di problemi di ottimizzazione è altamente suscettibile gli outliers perché i residui sono elevati al quadrato. Una possibile soluzione a questo problema è la cosiddetta *ottimizzazione L1* in cui viene minimizzata la somma del valore assoluto dei residui piuttosto che il loro quadrato. Purtroppo però la funzione di costo risulta essere non differenziabile nell'origine, il che rende difficile ottenere la soluzione ottima. Come compromesso tra l'ottimizzazione L1 e L2 si può utilizzare la *Huber loss function*, una funzione che per piccoli valori ha un andamento quadratico, come un problema L2, ma che diventa lineare per grandi valori comportandosi dunque come una funzione di costo di tipo L1. La Huber loss function è così definita:

$$e(n) = \begin{cases} \frac{n^2}{2} & \text{se } |n| \leq k \\ k|n| - \frac{k^2}{2} & \text{se } |n| > k \end{cases}$$

con k una soglia definita empiricamente ed n la misura della distanza.

La versione non lineare di ICP sfrutta proprio la Huber loss function al posto della classica funzione di perdita quadratica per ridurre l'influenza degli outliers. Utilizzando dunque questa funzione, l'algoritmo di registrazione

nella versione non lineare può essere scritto come:

$$\hat{R}, \hat{\mathbf{T}} = \arg \min_{R, \mathbf{T}} \sum_{k=1}^N e^2(n_k) = \arg \min_{R, \mathbf{T}} \sum_{k=1}^N e^2(\| (R\mathbf{p}_k + \mathbf{T}) - \mathbf{q}_k \|)$$

Per ottenere le stime delle soluzioni ottime \hat{R} , $\hat{\mathbf{T}}$ si implementa l'algoritmo *Levenberg-Marquardt* (LMA), un metodo iterativo simile ad algoritmi come la discesa del gradiente e i metodi di Gauss-Newton che sono in grado di trovare velocemente il minimo locale di funzioni non lineari.

2.1.4 Generalized ICP

La versione point to point assume che la source cloud appartenga ad una superficie geometrica nota piuttosto che essere ottenuta da misurazioni soggette a rumore e questo, a causa della discretizzazione dell'errore, implica che è altamente difficile ottenere una registrazione perfetta. Conseguentemente, la versione point to plane rilassa questa assunzione permettendo un margine di libertà dei punti nei piani su cui giacciono, al fine di attenuare gli effetti della discretizzazione. Tuttavia, suppone ancora che la source cloud rappresenti un campione discretizzato di una superficie geometrica nota poiché gli offsets lungo la superficie sono consentiti solo nella nuvola target.

Per compensare queste limitazioni, la versione generalizzata di ICP introduce un'interpretazione probabilistica del problema di minimizzazione così che le informazioni strutturali relative alle due nuvole possano essere facilmente incorporate nell'algoritmo di ottimizzazione stesso. Si assume che la source cloud $P = \{\mathbf{p}_k\}$ e la target cloud $Q = \{\mathbf{q}_k\}$ siano formate da campioni casuali estratti da due nuvole di punti ignote $\hat{P} = \{\hat{\mathbf{p}}_k\}$ e $\hat{Q} = \{\hat{\mathbf{q}}_k\}$. Per \hat{P} e \hat{Q} esiste un insieme di corrispondenze perfette, ma lo stesso non vale per P e Q dal momento che ciascun punto \mathbf{p}_k e \mathbf{q}_k si suppone essere campionato da una distribuzione normale tale per cui $\mathbf{p}_k \sim \mathcal{N}(\hat{\mathbf{p}}_k, C_k^P)$ e

$\mathbf{q}_k \sim \mathcal{N}(\hat{\mathbf{q}}_k, C_k^Q)$ dove le matrici di covarianza C_k^P e C_k^Q sono ignote. Se P e Q fossero composte da campioni deterministicici di modelli geometrici noti, entrambe le matrici di covarianza sarebbero nulle così che $P = \hat{P}$ e $Q = \hat{Q}$.

Indicata con T la matrice di trasformazione affine che mappa \hat{P} in \hat{Q} tale che $\hat{\mathbf{q}}_k = T\hat{\mathbf{p}}_k$. Se T fosse nota, la si potrebbe applicare alla source cloud campionata P e definire l'errore da minimizzare come $d_k^T = \mathbf{q}_k - T\mathbf{p}_k$. Ora, poiché si assume che \mathbf{p}_k e \mathbf{q}_k siano campionati da distribuzioni normali indipendenti, anche d_k^T , essendo una loro combinazione lineare, è campionato da una distribuzione normale:

$$\begin{aligned} d_k^T &\sim \mathcal{N}(\hat{\mathbf{q}}_k - T\hat{\mathbf{p}}_k, C_k^Q + TC_k^P T^\top) \\ &= \mathcal{N}(0, C_k^Q + TC_k^P T^\top) \end{aligned}$$

La matrice di trasformazione ottimale \hat{T} è quella trasformazione che minimizza la log-likelihood negativa degli errori osservati d_k :

$$\begin{aligned} \hat{T} &= \arg \min_T \sum_k \log(p(d_k^T)) \\ &= \arg \min_T \sum_k d_k^{T\top} (C_k^Q + TC_k^P T^\top)^{-1} d_k^T \end{aligned}$$

Questa versione generalizzata permette di includere le matrici di covarianza nelle versioni point to point e point to plane che quindi risulteranno essere dei casi particolari di generalized ICP. Infatti, per $C_k^Q = I$ e $C_k^P = 0$ si ottiene la versione point to point, mentre per $C_k^Q = P_k^{-1}$ e $C_k^P = 0$ (con P_k^{-1} la normale alla superficie in \mathbf{p}_k) si ottiene la versione point to plane.

Capitolo 3

Analisi delle prestazioni

*"I'm still standing better than I
ever did looking like a true
survivor, feeling like a little kid"*

Elton John

L'obiettivo di questo lavoro di tesi è valutare le prestazioni dell'algoritmo Iterative Closest Point adottando come punto di partenza una serie di diverse condizioni iniziali al fine di analizzare i risultati prodotti dall'algoritmo coprendo una casistica di possibili situazioni di impiego dell'algoritmo che sia la più ampia possibile. Parallelamente a ciò si vuole anche valutare l'efficacia di eventuali accorgimenti e strategie da applicare ai dati nella fase iniziale precedente l'esecuzione dell'algoritmo. Per condurre questa analisi si è deciso di adottare la versione point to point di ICP.

L'elaborato nella sua totalità è stato sviluppato in linguaggio MATLAB impiegando l'implementazione di ICP fornita dalle librerie di base e definita tramite la funzione PCREGISTERICP. Tale funzione richiede di specificare il valore di alcuni parametri formali tra cui: source point cloud e target point cloud (rispettivamente *moving* e *fixed* nel manuale [8]), la metrica di mini-

mizzazione da adottare nell'esecuzione per stimare la trasformazione rigida, il massimo numero di iterazioni che possono essere eseguite e le tolleranze di traslazione e rotazione tra iterazioni consecutive. Nello specifico, le tolleranze costituiscono due criteri di arresto dell'algoritmo che si aggiungono al massimo numero di iterazioni consentite e rappresentano le differenze assolute della rotazione e traslazione stimate da ICP in iterazioni consecutive. La tolleranza di traslazione misura la distanza euclidea tra due vettori di traslazione mentre la tolleranza di rotazione misura la differenza angolare in gradi; l'algoritmo si arresta quando la differenza media tra le trasformazioni rigide stimate nelle ultime tre iterazioni consecutive è inferiore alle due tolleranze specificate. Volendo condurre un'analisi il più possibile focalizzata sulla versione base dell'algoritmo ad entrambe le tolleranze è stato assegnato un valore di *0.001* in modo che il relativo criterio di arresto non sia predominante nell'esecuzione dell'algoritmo. Infine, come anticipato, la metrica di minimizzazione scelta è la *point to point*. Sono in realtà presenti ulteriori parametri ma ai fini di questo elaborato ne sono stati mantenuti i valori assegnati di default in quanto non influenti nel lavoro intrapreso.

```
% Set parameters values and run ICP
tic;
[~, movingReg, rmse] = pcregistericp(transformedCloud,
    originalCloud, 'Metric', 'pointToPoint', 'MaxIterations',
    maxIter, 'Tolerance', [0.001, 0.001]);
elapsedTime = toc;
```

I valori restituiti dalla funzione sono, rispettivamente, la nuvola di punti ruotata e traslata in base alla trasformazione stimata e l'*rmse*: la radice quadrata dell'errore quadratico medio calcolato come la media delle distanze euclidee tra i punti allineati delle due nuvole. Tramite l'impiego delle funzioni

di supporto TIC e TOC è stato possibile misurare il tempo di esecuzione di ICP (il valore restituito è in millisecondi).

Di fondamentale importanza per il problema della registrazione è la definizione della trasformazione omogenea. Questa, nel caso tridimensionale, è rappresentata tramite una matrice quadrata 4×4 composta da una matrice di rotazione tridimensionale R e da un vettore di traslazione $\mathbf{t} \in \mathbb{R}^3$:

$$T = \left(\begin{array}{c|c} R & \mathbf{t} \\ \hline \mathbf{0} & 1 \end{array} \right) = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Tale matrice di trasformazione viene implementata nella funzione TRANSFORMATION_MATRIX che riceve come parametri in ingresso le tre componenti di traslazione lungo i tre assi cartesiani e i tre angoli di rotazione rispetto agli assi del sistema di riferimento. In particolare, indicati con ψ , θ e ϕ gli angoli di rotazione rispetto agli assi X , Y e Z , la matrice di rotazione è stata definita come (si abbrevia la funzione seno con s e coseno con c):

$$R = \begin{pmatrix} c_\phi c_\theta & -s_\theta c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{pmatrix}$$

3.1 Modelli e massimo numero di iterazioni

L'intera analisi dell'algoritmo ICP è stata sviluppata impiegando quattro esempi tridimensionali: una *sedia*, un *tavolo*, una *tazza da caffè* e un *cappello*, tutti reperibili presso [5],[6]. Per ciascuno dei quattro esempi sono state scelte due immagini tridimensionali dell'oggetto: quelle con maggior numero di punti sono state utilizzate come fixed point clouds, ovvero le nuvole che

durante l'esecuzione di ICP vengono mantenute fisse, mentre quelle con minor numero di punti sono state impiegate come moving point clouds, ovvero le nuvole di punti soggette alla trasformazione rigida stimata.

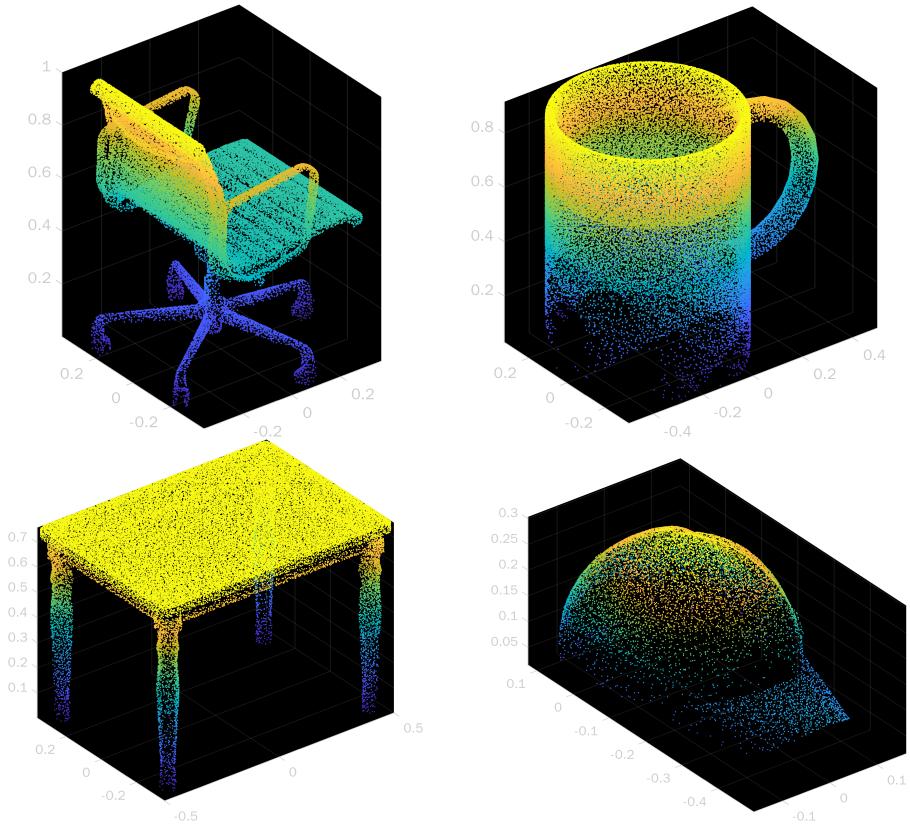


Figura 3.1: Modelli di point clouds usate

Per quanto riguarda il massimo numero di iterazioni si è partiti da un valore di 100 iterazioni ma mediante una prima fase di esperimenti ci si è resi conto che non erano sufficienti per studiare al meglio le prestazioni di ICP e per questo motivo si è deciso di proseguire nell'analisi considerando bene quattro possibili limiti: 50, 100, 200 e 400 iterazioni massime. In questo primo blocco di esperimenti, nella fase preliminare che precede l'esecuzione di ICP, è stata applicata una rotazione alla point cloud mobile definita con-

siderando quattro possibili casi: rotazione rispetto la sola asse X , rotazione rispetto la sola asse Y , rotazione rispetto la sola asse Z e rotazione 3D rispetto ai tre assi con il medesimo angolo. Per il valore dell'angolo si è considerato l'intervallo $[0, 2\pi]$ con passo costante di $\frac{\pi}{6}$ e per ciascun angolo sono state eseguite 100 prove in ciascuna delle quali, insieme alla matrice di rotazione, si è applicato un vettore di traslazione definito casualmente (è stata impostata una traslazione massima MAXTRANSLATION = 1.0).

```

if rotType == 'X'
    % Rotation w.r.t X axis
    initialTransformation = transformation_matrix(radAngle, 0, 0,
        trsX, trsY, trsZ);
elseif rotType == 'Y'
    % Rotation w.r.t Y axis
    initialTransformation = transformation_matrix(0, radAngle, 0,
        trsX, trsY, trsZ);
elseif rotType == 'Z'
    % Rotation w.r.t Z axis
    initialTransformation = transformation_matrix(0, 0, radAngle,
        trsX, trsY, trsZ);
else
    % 3D rotation
    initialTransformation = transformation_matrix(radAngle,
        radAngle, radAngle, trsX, trsY, trsZ);
end

```

Questo procedimento è stato applicato a tutti e quattro gli esempi ponendo inizialmente un massimo numero di iterazioni pari a 100 e successivamente a 200. Dallo studio dei risultati ottenuti si è notato che pur raddoppiando

do il massimo numero iterazioni consentite non ci sono stati miglioramenti particolarmente rilevanti in termini di rmse, ed è proprio per questo motivo che si è deciso di procedere considerando due ulteriori casi in cui il limite di iterazioni è posto a 50 e 400.

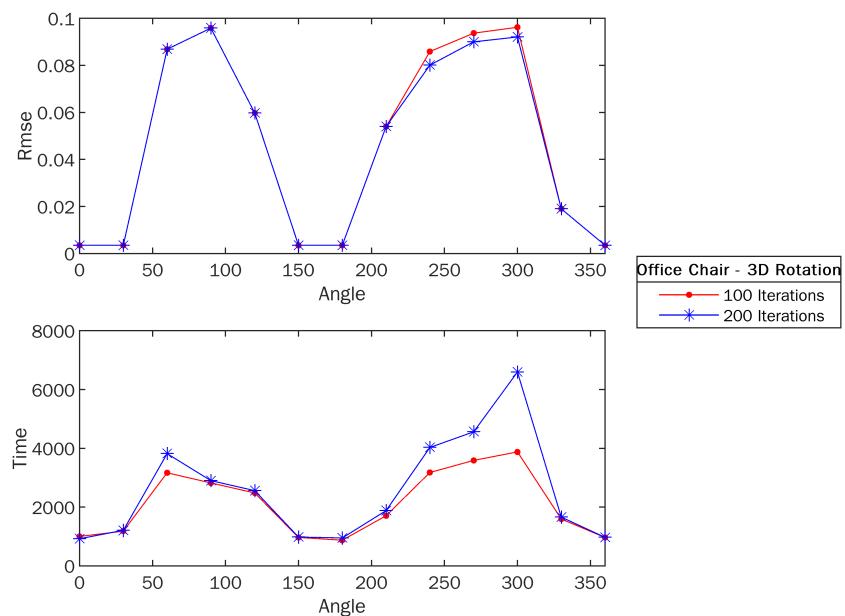


Figura 3.2: Esempio di confronto di rmse [m] e tempo di esecuzione [ms] di ICP

3.2 Analisi

Dopo questa fase di analisi preliminare, scelti i quattro modelli e i quattro limiti nel numero di iterazioni sono state condotte quattro tipologie di prove cercando di coprire più scenari possibili di applicazione di ICP, intendendo come possibili scenari particolari caratteristiche che le due point clouds in input potrebbero avere.

Il primo tipo di prove, implementato nella funzione `ERROR_PER_ITER`, prende in esame l'applicazione più semplice di ICP in cui le due nuvole di

punti, salvo avere un diverso numero di punti, non hanno altre particolari proprietà, per esempio non sono deformate e non presentano alcun tipo di rumore. Per ciascuno dei quattro esempi sono state condotte 100 prove in ciascuna delle quali, prima di eseguire l'algoritmo, è stata applicata una rototraslazione iniziale alla nuvola di punti mobile definita in modo casuale sia per la rotazione che per la traslazione. Come nel caso precedente è stata impostata una traslazione massima MAXTRANSLATION = 1.0 e una rotazione (intera) massima MAXROTATION = 2π .

```
% Define a random 3D rotation from the uniform distribution between
0 and maxRotation
rotX = randi([0 maxRotation], 1, 1);
rotY = randi([0 maxRotation], 1, 1);
rotZ = randi([0 maxRotation], 1, 1);

% Define a random 3D translation vector where each component is
uniformly distributed in the interval [0, maxTranslation]
trsX = 0 + (maxTranslation - 0)*rand(1, 1);
trsY = 0 + (maxTranslation - 0)*rand(1, 1);
trsZ = 0 + (maxTranslation - 0)*rand(1, 1);
initialTransformation = transformation_matrix(deg2rad(rotX),
                                              deg2rad(rotY), deg2rad(rotZ), trsX, trsY, trsZ);
transformedCloud = pctransform(movingCloud, initialTransformation);
```

Inoltre, è stato calcolato l'rmse iniziale prima di eseguire ICP per valutarne l'andamento.

```
[~, pointDistances] = knnsearch(originalCloud.Location,
                                 transformedCloud.Location);
initialRmse = sqrt(sum(pointDistances) / numel(pointDistances));
```

Analizzando i risultati di questo primo tipo di prove si è ottenuto che l'algoritmo ha raggiunto l'*ottimo globale* almeno una volta in tutti e quattro gli esempi, il che significa che è stato in grado di determinare la trasformazione rigida che permette di sovrapporre correttamente le due nuvole.

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	6	6	6	7
	Tavolo	16	25	29	29
	Tazza	3	5	7	9
	Cappello	0	2	7	27

Tabella 3.1: Sovrapposizioni corrette, su 100 prove, dopo aver eseguito ICP

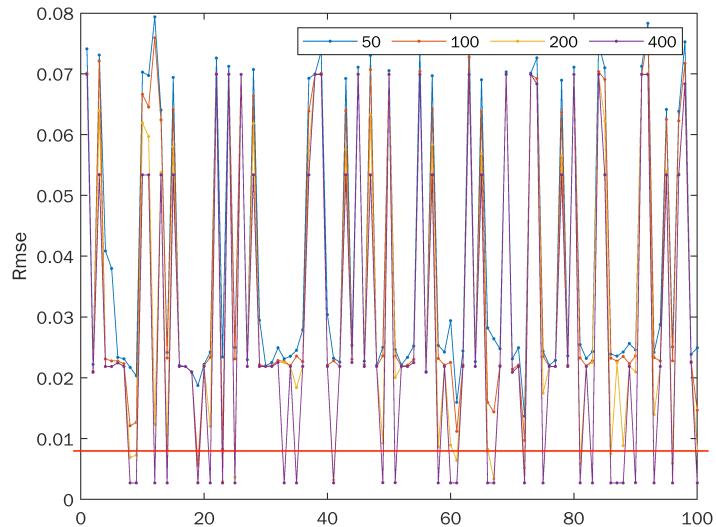


Figura 3.3: Rmse del cappello dopo aver applicato ICP. I punti al di sotto della linea rossa sono le prove in cui si è ottenuta una sovrapposizione corretta

Confrontando i valori di rmse ottenuti prima e dopo aver eseguito ICP si nota che questo è diminuito in media di un ordine di grandezza passando

da 0.1 a 0.01. Inoltre, nelle prove in cui si è ottenuta una sovrapposizione corretta delle due nuvole di punti, l'rmse è diminuito di un ulteriore ordine di grandezza arrivando a 0.001 come si può notare in figura 3.3.

3.2.1 Sovrapposizione dei baricentri

Mantenendo esempi privi di deformazioni e rumore, l'analisi è proseguita introducendo un nuovo passaggio prima di eseguire ICP: definita in modo casuale una matrice di rotazione iniziale come fatto nei precedenti esperimenti, è stato definito un vettore di traslazione da applicare alla point cloud mobile in modo che i baricentri delle due nuvole coincidano. Questo vettore così definito va a sostituire il precedente vettore di traslazione aleatorio.

Per ottenere il vettore di traslazione sono stati da prima calcolati i baricentri delle due nuvole come quei punti che hanno per coordinate la media delle coordinate di tutti i punti che compongono le nuvole e successivamente si è ricavato il vettore definendolo come la differenza tra i due baricentri.

```
% Calculate point clouds centroid
ocCentroid = mean(originalCloud.Location);
tcCentroid = mean(rotatedCloud.Location);

% Define and apply a 3D translation which overlaps the centroids
initialTranslation = tcCentroid - ocCentroid;
centeringTranslation = transformation_matrix(deg2rad(0),
    deg2rad(0), deg2rad(0), -initialTranslation(1),
    -initialTranslation(2), -initialTranslation(3));
transformedCloud = pctransform(rotatedCloud, centeringTranslation);
```

L'introduzione di questo accorgimento preliminare che precede l'esecuzione di ICP ha permesso di aumentare il numero di sovrapposizioni corrette rispetto

ai precedenti esempi in cui non era stata applicata alcun tipo di strategia. Al contrario, l'andamento dell'ordine di grandezza dell'rmse continua a rimanere il medesimo.

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	13	15	16	17
	Tavolo	24	25	24	24
	Tazza	4	5	10	10
	Cappello	19	27	40	44

Tabella 3.2: Sovrapposizioni corrette, su 100 prove, dopo aver sovrapposto i baricentri e aver eseguito ICP

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	+7	+9	+10	+10
	Tavolo	+8	=	-5	-5
	Tazza	+1	=	+3	+1
	Cappello	+19	+25	+33	+17

Tabella 3.3: Variazione del numero di sovrapposizioni corrette con l'aggiunta della sovrapposizione dei baricentri

Esaminando la tabella 3.3 si nota che l'introduzione della sovrapposizione preliminare dei baricentri permette di ottenere un considerevole incremento del numero di sovrapposizioni corrette e in base a questi risultati si può ritenere che sia una tecnica utile nella risoluzione del problema della registrazione. Non a caso, un approccio simile viene utilizzato nella già citata PCA sempre per la medesima tipologia di problemi.

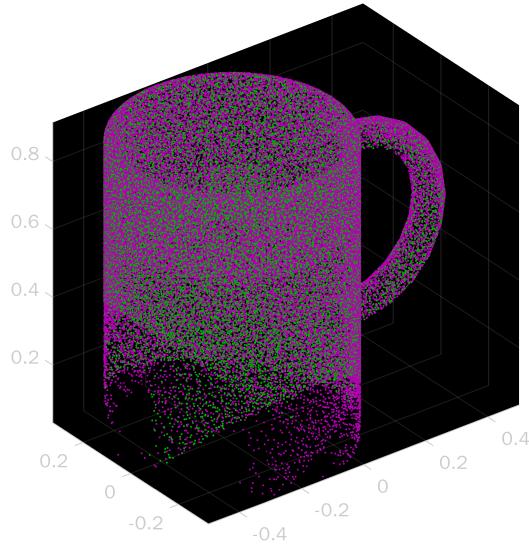


Figura 3.4: Esempio di sovrapposizione corretta delle due point clouds

3.2.2 Deformazione

Con la terza tipologia di prove si sono prese le distanze da situazioni ideali in cui le due nuvole possono essere perfettamente sovrapposte e ci si è allineati con contesti più prossimi alla realtà in cui non si può più garantire la perfetta sovrapposizione perché, per esempio, le immagini sono state acquisite adottando strumenti diversi con diverse caratteristiche e tarature, oppure perché le acquisizioni sono state compromesse da eventuali movimenti indesiderati che hanno deformato il risultato finale o per altri motivi ancora.

Al fine di poter analizzare queste situazioni sono state affiancate alla rotazione e traslazione iniziali ulteriori trasformazioni in grado di deformare la nuvola di punti. Nello specifico, sono state selezionate due trasformazioni:

Scaling operazione tramite la quale si alterano le dimensioni dell'immagine: si scala l'immagine ingrandendola o rimpicciolendola. Il *fattore di scaling* può essere il medesimo per le tre coordinate spaziali oppure può

differire per ciascuna di esse. Nel primo caso l'immagine sarà semplicemente scalata rispetto all'originale ma la sua struttura rimarrà del tutto inalterata. Nel secondo caso l'immagine finale sarà una deformazione di quella originale oltre ad avere dimensioni diverse. Di seguito sono state analizzate entrambe le possibilità

Shearing trasformazione che inclina l'immagine tridimensionale rispetto una o più delle direzioni canoniche X , Y e Z . Inevitabilmente viene alterata la forma dell'oggetto. L'inclinazione può essere controllata mediante un certo numero di parametri che vanno a moltiplicare le coordinate di ciascun punto che compone l'immagine

Dal punto di vista matematico [12],[2] le due trasformazioni sono rappresentate tramite una matrice quadrata 4×4 . Lo *scaling* è rappresentato mediante la matrice diagonale:

$$SC = \begin{pmatrix} SC_X & 0 & 0 & 0 \\ 0 & SC_Y & 0 & 0 \\ 0 & 0 & SC_Z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

dove SC_X , SC_Y e SC_Z sono i tre fattori di scaling che possono essere uguali o diversi tra loro. Lo *shearing* per mezzo della matrice quadrata:

$$SH = \begin{pmatrix} 1 & SH_{Y_x} & SH_{Z_x} & 0 \\ SH_{X_y} & 1 & SH_{Z_y} & 0 \\ SH_{X_z} & SH_{Y_z} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

in cui la prima colonna modella lo shearing lungo l'asse X , la seconda colonna lo shearing lungo Y e la terza colonna quello lungo l'asse Z .

Sono state condotte quattro classi di esperimenti per ognuno dei quattro modelli: scaling uguale, scaling diverso, shearing e infine una combinazione di shearing e scaling nella versione con il medesimo fattore di scala.

```
% Scaling matrix with the same scaling factor for X, Y and Z
scalingFactor = random_value(0.5, 2);
deformationMatrix = [scalingFactor 0 0 0;
                      0 scalingFactor 0 0;
                      0 0 scalingFactor 0;
                      0 0 0 1];

% Scaling matrix with three different scaling factors
deformationMatrix = [random_value(0.5, 2) 0 0 0;
                      0 random_value(0.5, 2) 0 0;
                      0 0 random_value(0.5, 2) 0;
                      0 0 0 1];

% Shearing matrix
shearingX = [random_value(0, 0.5), random_value(0, 0.5)];
shearingY = [random_value(0, 0.5), random_value(0, 0.5)];
shearingZ = [random_value(0, 0.5), random_value(0, 0.5)];
deformationMatrix = [1 shearingX(1) shearingX(2) 0;
                      shearingY(1) 1 shearingY(2) 0;
                      shearingZ(1) shearingZ(2) 1 0;
                      0 0 0 1];

% Scaling + Shearing matrix (same scaling factor for X, Y and Z)
scalingFactor = random_value(0.5, 2);
shearingX = [random_value(0, 0.5), random_value(0, 0.5)];
shearingY = [random_value(0, 0.5), random_value(0, 0.5)];
shearingZ = [random_value(0, 0.5), random_value(0, 0.5)];
deformationMatrix = [scalingFactor shearingX(1) shearingX(2) 0;
```

```

shearingY(1) scalingFactor shearingY(2) 0;
shearingZ(1) shearingZ(2) scalingFactor 0;
0 0 0 1];

```

Il primo passo è stato quello di applicare la trasformazione così definita alla point cloud mobile in modo da deformarla, e successivamente è stata traslata e ruota tramite una matrice di roto-traslazione definita in modo casuale come fatto nelle precedenti fasi dell'analisi. Inoltre, come negli esperimenti precedenti, per tutte e quattro le tipologie di trasformazione e per ciascuno dei quattro modelli, sono state effettuate 100 prove. Si riportano di seguito il numero di sovrapposizioni corrette al variare della trasformazione applicata.

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	6	8	8	8
	Tavolo	14	21	22	26
	Tazza	1	1	1	1
	Cappello	7	12	14	16

Tabella 3.4: Sovrapposizioni corrette, su 100 prove, prodotte da ICP dopo aver effettuato uno scaling

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	5	6	6	6
	Tavolo	8	11	15	15
	Tazza	2	2	2	3
	Cappello	3	3	10	12

Tabella 3.5: Sovrapposizioni corrette, su 100 prove, prodotte da ICP dopo aver effettuato uno scaling con diversi fattori di scala

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	2	2	2	2
	Tavolo	15	17	19	19
	Tazza	1	1	2	2
	Cappello	5	9	9	9

Tabella 3.6: Sovrapposizioni corrette, su 100 prove, prodotte da ICP dopo aver effettuato lo shearing

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	5	6	7	7
	Tavolo	16	23	26	26
	Tazza	3	3	3	4
	Cappello	7	11	11	11

Tabella 3.7: Sovrapposizioni corrette, su 100 prove, prodotte da ICP dopo aver effettuato una combinazione di shearing e scaling

Anche in questo caso per tutti i modelli e per tutti i limiti di iterazioni è stato possibile ottenere almeno una sovrapposizione corretta delle due point clouds, anche se una delle due è stata deformata. I risultati ottenuti in questo esperimento permettono dunque di affermare che ICP è un algoritmo sufficientemente robusto poiché permette di raggiungere l'ottimo globale anche in condizioni iniziali non ottimali come, appunto, nel caso di immagini deformate. In aggiunta si può osservare che il numero di sovrapposizioni corrette nelle quattro trasformazioni ha un andamento analogo tra i quattro esempi: il tavolo e il cappello sono i due esempi con il maggior numero di sovrapposizioni corrette e con il maggior incremento di sovrapposizioni all'aumentare

del numero massimo di iterazioni. Al contrario, la sedia e la tazza sono i due esempi per cui ci sono state meno sovrapposizioni corrette e per cui l'aumento del limite di iterazioni non ha migliorato in modo considerevole tale numero.

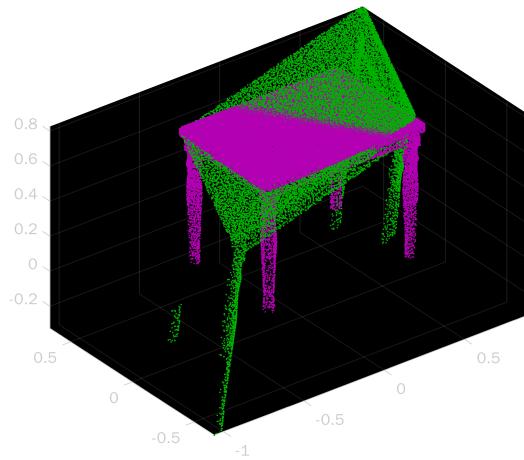


Figura 3.5: Esempio di sovrapposizione corretta nel caso di point cloud soggetta ad una deformazione di tipo scaling + shearing

In conclusione si deve sottolineare il fatto che, avendo introdotto una deformazione della nuvola di punti, in questa tipologia di esperimenti il concetto di sovrapposizione corretta è stato rilassato (come si deduce dalla figura 3.5) intendendo non più una sovrapposizione perfetta delle point clouds ma bensì un buon allineamento delle due nuvole in termini di rotazione e traslazione in quanto ICP non è in grado di sopperire alla deformazione introdotta.

3.2.3 Rumore

Nell'ultima fase dell'analisi di ICP si è perseguito l'intento di studiare le prestazioni dell'algoritmo in situazioni più vicine alla realtà, e per questo motivo ci si è focalizzati nel caso in cui le nuvole di punti presentino al loro interno del rumore che potrebbe ostacolare la loro registrazione. Questa fase

è stata dunque sviluppata su due fronti: da un lato analizzando il comportamento di ICP in presenza di rumore e dall'altro valutando l'efficacia di un semplice algoritmo di rimozione degli *outliers* combinato con ICP.

Per quanto riguarda il primo caso, per ricondursi ad uno scenario in cui è presente del rumore all'interno delle nuvole, sono stati aggiunti in modo casuale 2000 nuovi punti alla point cloud mobile prima di procedere con la sua roto-traslazione iniziale casuale e la conseguente esecuzione di ICP.

```
% Define numOutliers new points as point cloud outliers
noise = zeros(numOutliers, 3);
movingCloud = pcread(transformedModel);
xLimits = movingCloud.XLimits;
yLimits = movingCloud.YLimits;
zLimits = movingCloud.ZLimits;
for p = 1:numOutliers
    noise(p, 1) = random_value(xLimits(1), xLimits(2));
    noise(p, 2) = random_value(yLimits(1), yLimits(2));
    noise(p, 3) = random_value(zLimits(1), zLimits(2));
end
```

Nuovamente si sono mantenute le 100 prove per ciascuno dei quattro modelli.

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	9	10	11	11
	Tavolo	13	19	20	23
	Tazza	5	6	10	14
	Cappello	7	33	35	36

Tabella 3.8: Sovrapposizioni corrette, su 100 prove, prodotte da ICP dopo aver inserito outliers

Dalla tabella 3.8 si può notare che l'aggiunta di outliers non ha impedito di ottenere sovrapposizioni corrette delle due nuvole di punti ma anzi si sono ricavate delle buone percentuali di successo, molto simili a quelle ottenute nello primo blocco di esperimenti che rappresentava un caso più ideale.

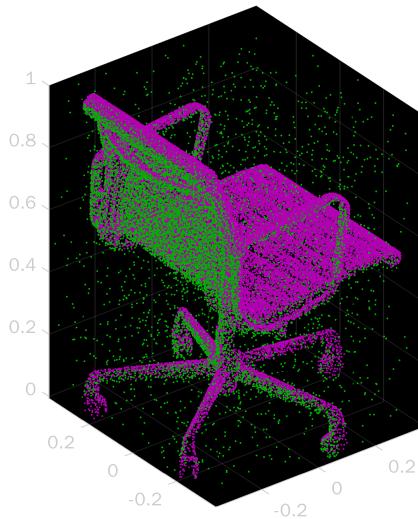


Figura 3.6: Esempio di sovrapposizione corretta nel caso di point cloud con rumore

Concluso lo studio delle prestazioni di ICP in presenza di rumore, si è passati a valutare se l'introduzione di un semplice algoritmo di rimozione del rumore avrebbe migliorato i risultati ottenuti. La rimozione del rumore può essere implementata conducendo un'analisi statistica delle caratteristiche dei punti più vicini a ciascun punto della nuvola. L'algoritmo scelto [7],[11] adotta, in modo analogo ad ICP, un approccio di tipo nearest neighbor: per ciascuno dei punti che compongono la point cloud, calcola i k punti ad esso più vicini e conseguentemente ne calcola le distanze. A questo punto, assumendo che la distribuzione delle distanze sia gaussiana, vengono calcolate la media μ e la deviazione standard σ delle distanze tra ciascun punto ed i suoi k punti più vicini. Fissata una soglia α , tutti i punti le cui distanze me-

die cadono all'esterno dell'intervallo $\mu \pm \alpha\sigma$ definito in funzione della media e della deviazione standard globali sono considerati outliers e conseguentemente vengono rimossi. Questo algoritmo è implementato in MATLAB nella funzione PCDENoise [9] che richiede tre parametri in ingresso: la point cloud sulla quale operare la rimozione del rumore, il numero k di punti più vicini che devono essere trovati per ciascun punto della nuvola (questo valore viene utilizzato per calcolare la media globale delle distanze medie dei vicini di ciascun punto) e la soglia (*threshold*) che verrà moltiplicata per la deviazione standard per stabilire se un punto è o meno un outlier.

Anche in quest'ultima fase dell'analisi sono state condotte 100 prove per ognuno dei quattro modelli. Per ciascuna prova, dopo aver aggiunto in modo casuale 5000 nuovi punti alla point cloud mobile come se fossero del rumore, è stata definita e applicata alla nuvola di punti mobile una roto-traslazione casuale e successivamente, prima di eseguire ICP, è stato eseguito l'algoritmo di rimozione del rumore.

```
% Remove outliers from transformed cloud
denoisedCloud = pcdenoise(transformedCloud, 'NumNeighbors', 5,
    'Threshold', 0.5);
```

Calcolando i 5 punti più vicini per ciascun punto della nuvola e adottando un fattore moltiplicativo della deviazione standard pari a 0.5 si sono ottenuti buoni risultati in termini di rimozione degli outliers.

In termini del numero di sovrapposizioni corrette, l'aggiunta dell'algoritmo di rimozione del rumore non ha introdotto particolari variazioni né positive né negative rispetto al caso precedente. In generale dunque è ancora stato possibile raggiungere l'ottimo globale per tutti e quattro gli esempi e per tutti i limiti del numero di iterazioni e, in media, le percentuali di suc-

cesso sono molto simili a quelle ottenute eseguendo ICP senza aver prima rimosso il rumore.

		Iterazioni massime			
		50	100	200	400
Modello	Sedia	6	6	7	7
	Tavolo	21	25	26	26
	Tazza	3	6	9	13
	Cappello	4	29	41	42

Tabella 3.9: Sovrapposizioni corrette, su 100 prove, prodotte da ICP dopo aver rimosso gli outliers

È dunque possibile affermare che la presenza di outliers non mina in modo considerevole le prestazioni dell'algoritmo che continuano ad essere più che soddisfacenti.

Capitolo 4

Conclusioni

Alla luce dei risultati ottenuti durante le fasi descritte, si evince che l'algoritmo Iterative Closest Point nella sua versione point to point permette di stimare, un soddisfacente numero di volte, una trasformazione rigida in grado di sovrapporre correttamente le due point clouds. Dal punto di vista di eventuali accorgimenti da affiancare ad ICP, si è visto che l'aggiunta della preventiva sovrapposizione dei baricentri consente di aumentare apprezzabilmente la percentuale di successi dell'algoritmo. Infine, anche in presenza di deformazioni o rumore è stato comunque possibile ottenere un buon numero di sovrapposizioni corrette mentre l'aggiunta di un semplice algoritmo di rimozione del rumore non ha apportato variazioni degne di nota.

4.1 Prospettive future

In base alle analisi condotte e alle conclusioni che ne sono state tratte, è possibile affermare che, anche in presenza di rumore, applicando o meno l'algoritmo di rimozione del rumore, si è stati in grado di raggiungere la sovrapposizione perfetta delle due nuvole di punti. L'unico caso in cui

non è stato possibile raggiungere l'ottimo globale è quello in cui sono state applicate delle deformazioni alla moving point cloud che, date le alterazioni introdotte, hanno impedito di ottenere una sovrapposizione perfetta. È proprio questa situazione che può essere sfruttata per poter descrivere una possibile prospettiva futura: poiché nel caso di trasformazione rigida ICP opera definendo solo una rotazione ed una traslazione non tenendo conto di eventuali deformazioni dovute, per esempio, a shearing e scaling si potrebbe introdurre una trasformazione non rigida che vada a compensare tale mancanza in modo da sopperire ad eventuali deformazioni. Nel raggiungimento di tale obiettivo si potrebbe eventualmente considerare l'impiego di un algoritmo di segmentazione che consenta di identificare degli elementi caratteristici nelle due immagini (e.g. lo schienale, i braccioli e le ruote di una sedia) e definire una corrispondenza tra questi elementi per ottimizzare la definizione della trasformazione cercata.

Bibliografia

- [1] Ben Bellekens et al. «A Benchmark Survey of Rigid 3D Point Cloud Registration Algorithms». In: *International Journal on Advances in Intelligent Systems* 8.1-2 (2015). DOI: http://www.iariajournals.org/intelligent_systems/.
- [2] GeeksforGeeks. *Computer Graphics – 3D Shearing Transformation*. 2021. URL: <https://www.geeksforgeeks.org/computer-graphics-3d-shearing-transformation/>.
- [3] Xiaoshui Huang et al. «A comprehensive survey on point cloud registration». In: *arXiv:2103.02690* (2021).
- [4] Xiaoshui Huang et al. «A systematic approach for cross-source point cloud registration by preserving macro and micro structures». In: *IEEE Transactions on Image Processing* (2017), 26(7):3261–3276.
- [5] Kevin Lai, Liefeng Bo e Dieter Fox. *Trimble 3D Warehouse Objects*. 2014. URL: <http://rgbd-dataset.cs.washington.edu/dataset/rgbd-scenes-v2/>.
- [6] Kevin Lai, Liefeng Bo e Dieter Fox. «Unsupervised Feature Learning for 3D Scene Labeling». In: *IEEE International Conference on Robotics and Automation (ICRA)* (2014).

- [7] Point Cloud Library. *Removing outliers using a StatisticalOutlierRemoval filter*. URL: https://pointclouds.org/documentation/tutorials/statistical_outlier.html.
- [8] MathWorks. *Register two point clouds using ICP algorithm*. 2018. URL: <https://it.mathworks.com/help/vision/ref/pcregistericp.html>.
- [9] MathWorks. *Remove noise from 3-D point cloud*. 2015. URL: <https://it.mathworks.com/help/vision/ref/pcdenoise.html>.
- [10] Furong Peng et al. «Street view cross-sourced point cloud matching and registration». In: *IEEE International Conference on Image Processing (ICIP)* (2014), pp. 2026–2030.
- [11] Rusu et al. «Towards 3D Point Cloud Based Object Maps for Household Environments». In: *Robotics and Autonomous Systems Journal* (2008).
- [12] Tutorialspoint. *3D Transformation*. URL: https://www.tutorialspoint.com/computer_graphics/3d_transformation.htm.

Ringraziamenti

Siamo finalmente giunti alle ultime pagine di questa tesi ed è arrivato il momento di scrivere alla prima persona singolare. Questi ultimi 5 anni e mezzo sono stati un vero e proprio pendolo semplice (anche se di semplice c'è stato ben poco) che ha oscillato costantemente tra momenti di felicità assoluta, raggiungendo picchi del calibro di "Nono, non è la mia materia preferita", per poi scendere in picchiata verso buchi neri di insicurezza e anche un po' di rassegnazione. Ma per fortuna, tanto il destino quanto la mia forza di volontà, hanno deciso di tagliare il filo ideale del pendolo nel momento giusto, e di eliminare ogni possibile attrito così che quel corpo, non poi così tanto puntiforme, potesse finalmente spiccare il volo con la giusta velocità di fuga e con tanta energia quanto basta per poter arrivare finalmente a questo tanto ambito traguardo per il quale è stato speso molto lavoro (sulla potenza magari ne riparliamo un'altra volta). E allora eccomi giunto al traguardo, pronto per ringraziare tutte quelle forze esterne che si sono sommate alla mia per riuscire in questa grande impresa.

Ringrazio il Professor Marco Sciandrone per avermi dato fiducia, accettando di lavorare insieme nella realizzazione di questo lavoro rimanendo sempre disponibile nonostante la distanza, e per avermi dato la giusta dose di soddisfazione quando più ne avevo bisogno. Ringrazio anche il Professor

Fabio Tardella per avermi aiutato nella parte finale di questo percorso facendomi da relatore interno.

Ringrazio la Dottoressa Zambuto che negli ultimi mesi ha risposto alla mia richiesta di aiuto ed è stata al mio fianco aiutandomi ad imparare come controllare e disinnescare tutti quei circoli viziosi dentro di me che mi mandavano in cortocircuito e mi ha fatto riscoprire quei punti di riferimento che credevo di non avere.

Durante questo moto armonico ho avuto la fortuna di conoscere tante persone che hanno addolcito il mio percorso, facendomi ridere in tutti i modi possibili a partire dal farmi cantare Gigi d'Alessio e gli Abba. Vorrei quindi ringraziare tutto il gruppo "Prato Merda" e "Dell'arte del ricamo"; grandi compagni d'avventura con cui ho potuto buttar fuori il peggio di me comportandomi nei modi più stupidi possibili senza, quasi, essere giudicato. Un grazie in particolare va a quell'ansia vivente di Bindini, perché nonostante le tante discussioni è sempre stato pronto ad ascoltare i miei audio deliranti e ad immortalare le mie peggiori figuracce. Ringraziamenti a parte per Lo Zio che col suo "fra non sei in grado" c'aveva visto lungo, ma molto molto lungo. Ah, ovviamente grazie anche a Yuri.

Un ringraziamento va anche a Ila, Oliv, Robi, Mugna, Gianlu, Fra e Zarro per avermi accolto nel loro gruppo senza mai farmi sentire di troppo. Grazie per tutte le ore passate a far finta di studiare, per tutti i ciccozzi scrocchiati, per le cene, le uscite, le sagre del vino a Montespertoli e per tutti quegli incomprensibili disegni fatti alla lavagna per cercare di spiegare storie più difficili di Beautiful. State tranquilli... non mi sono dimenticato di nessuno.

no. Non posso non dedicare un ringraziamento particolare a Lau e Mirco che in tutti questi anni mi sono sempre stati accanto nei momenti di felicità e soprattutto nei momenti peggiori in cui credevo che non sarei riuscito in niente. Da un lato la dolcezza e l'empatia di Lau (e Laika), i disagi condivisi e le lunghe passeggiate per far finta che fosse tutto a posto, e dall'altro la razionalità di Mirco e le sue poche parole che bastavano a farmi capire che ci sarebbe sempre stato, mi hanno sempre aiutato a rialzarmi, a credere in me e a superare ogni sfida.

Grazie agli amici di sempre: agli "Scomposti" e al gruppo di "Dublino" per essere sempre stati una di quelle certezze di cui tutti abbiamo bisogno, un punto di riferimento fisso in cui trovare aiuto e un tetto sicuro sotto il quale ho sempre saputo di potermi riparare. In particolare, grazie a Beatrice per essere semplicemente se stessa: una forza della natura nella sua forma più gentile; compagna di innumerevoli momenti passati insieme che non potrò mai dimenticare.

Ma ci sono anche persone che non riuscirò mai a ringraziare abbastanza: Sara e Laura, due delle persone più belle che abbia mai conosciuto. Grazie a Sara perché semplicemente c'è sempre stata, nel bene e nel male, quando non credevo in me stesso ma anche quando l'ego mi dava un po' troppo alla testa. Potrei scrivere miliardi di cose per cui ringraziarti ma le riassumerò tutte in una cosa soltanto: grazie per aver ascoltato ogni singolo, lunghissimo, pesantissimo mio audio durante i miei infiniti deliri. E grazie a Laura perché nell'essere così identici non c'è mai stato bisogno di parlare troppo, per avermi fatto compagnia nei momenti migliori e per avermi rimproverato quando stavo per finire sulla brutta strada.

Ma navigando nel vasto mare degli amici di lunga data è bello imbattersi in nuove isole e quindi eccomi a ringraziare anche Matilde per avermi regalato nuove "dinamiche" in un periodo un po' troppo monotono, ma soprattutto per avermi aiuto ad accendere i motori e prendere in mano la situazione.

Eccomi arrivato ad uno dei ringraziamenti a cui tengo di più in assoluto, persone alle quali voglio talmente tanto bene che voglio nominarle una per una; quindi grazie ad Emilia, Giovanni, Angela, Alice, Caterina e Gilberto. Solo per voi potrei scrivere migliaia di pagine perché migliaia sono i momenti che abbiamo passato insieme da ormai quasi 20 anni. Migliaia sono i biscotti mangiati dopo pranzo e cena, migliaia le volte in cui Giovanni mi ha accolto con: "Athos, ancora qui", migliaia le volte in cui Caterina mi ha bacchettato dicendomi di tornare più spesso a casa "mia", migliaia le volte in cui ho sentito Alice come una sorella maggiore, migliaia le volte in cui Angela mi ha aperto la porta, ha trasformato una mia semplice visita pomeridiana in una cena di famiglia e mi ha fatto capire che al loro tavolo, la sedia alla sinistra di Gilberto sarà sempre la mia. Migliaia le volte in cui Emilia mi ha accolto tra le sue braccia dicendo: "Eccolo! È arrivato il mio Sapientino", ma soprattutto migliaia di migliaia sono le giornate, i momenti, le estati (e le ore aspettate prima di ricevere una risposta) passate con Gilberto. Ormai ognuno di voi per me è casa e il bene che mi volete e che io voglio a voi non potrai mai e poi mai essere eguagliato. Grazie.

Ci tengo a fare un ringraziamento di cuore anche al mio grande amico Simone, a Cristina e a tutta la loro famiglia, a Teresa, a Valeria, Massimo e Tommaso perché tutti loro mi hanno sempre fatto sentire il benvenuto a

casa loro, hanno sempre creduto in me e nelle mie potenzialità, e mi hanno sempre voluto bene come se facessi parte della loro famiglia.

Sono quindi arrivato alla fine, ed proprio alla fine che mi sono lasciato la ciliegina sulla torta, i ringraziamenti più importanti: quelli per mia mamma. Sei sempre stata la mia roccia, il mio faro splendente nel pieno della tempesta, il mio cuscino magico che fa svanire ogni incubo ma soprattutto sei sempre stata la miglior compagna con cui cantare a squarciagola le canzoni dei tuoi anni con cui sono cresciuto. Hai sempre creduto in me, mi hai insegnato a credere in me e se oggi sono quello che sono è solo ed esclusivamente merito tuo e di babbo. Grazie.

P.S. Nonostante tu sia in lizza per vincere il premio "Peggior persona con cui guardare film e serie tv" ti voglio tantissimo bene lo stesso.