Università degli Studi di Padova

Dipartimento di Ingegneria Industriale

Tesi di Laurea Magistrale in Ingegneria Aerospaziale

Analisi sperimentale di un metodo di navigazione basata su lidar 3D

Relatore:
Prof. Marco Pertile

Candidato:
Stefano Giaretta

Anno Accademico 2022/2023

Abstract

La guida autonoma di un veicolo, o robot, continua a essere sempre più oggetto di studio e ricerca da parte di università e imprese in tutto il mondo. I vantaggi sono notevoli quali, per esempio, una maggiore sicurezza nell'ambito dei trasporti e una riduzione del rischio in tutte quelle operazioni che vedono protagonista l'intervento umano. Questa tesi si pone come obiettivo l'analisi sperimentale di un metodo di navigazione basata su lidar 3D. In particolare, vengono ricostruite le traiettorie del rover Morpheus ottenute attraverso l'utilizzo del lidar e, successivamente, confrontate con i dati acquisiti con il gps a frequenza singola. Viene poi proposto un ulteriore confronto tra due lidar, caratterizzati da un FoV orizzontale diverso. In conclusione, vengono messi in evidenza i relativi punti di forza del metodo scelto e avanzate possibili soluzioni per migliorare i risultati ottenuti.

Indice

Capitolo 1

Metodo

Una delle possibilità per ricostruire il percorso compiuto da un corpo e la mappa dell'ambiente circostante, usando un sistema Lidar 3D, è allineare le scansioni lidar ottenute dal veicolo in movimento, applicando l'algoritmo NDT (Normal Distributions Transform), e combinarle in un'unica grande nuvola di punti. Vediamo in dettaglio il metodo scelto [6] rappresentato in figura 1.1.

1.1 Raccolta Point Cloud e Preprocess

La raccolta della Point Cloud, in base al lidar utilizzato, può risultare differente. In particolare, il funzionamento dei lidar utilizzati è esposto nel capitolo 3. La nuvola di punti viene, successivamente, divisa tra Point Cloud fissa, che rappresenta l'ambiente circostante il sensore, e Point Cloud in movimento che individua, invece, tutti quei punti che rappresentano oggetti in moto rispetto alla Point Cloud fissa. Inizia la fase di "preprocess". Entrambe le Point Clouds vengono quindi analizzate come illustrato da Igor Bogoslavskyi & Cyrill Stachniss, Bonn [2], P. H. S. Torr and A. Zisserman [9] e Marius Muja, David G. Lowe [5]. Vengono, dunque, individuati e rimossi il "ground plane" e l'"ego vehicle".

1.2 Algoritmo NDT

L'algoritmo NDT, ampiamente argomentato nel capitolo 2, individua la trasformazione tra due Point Clouds. Si articola nel modo seguente:

1. calcola le distribuzioni normali per la Point Cloud fissa dividendo l'area coperta dalla scansione della Point Cloud in "voxel", riquadri 3D di dimensione costante contenenti punti della scansione. Per i punti in ciascun voxel, viene calcolata la media e la matrice di covarianza;

2. allinea la Point Cloud in movimento alla Point Cloud fissa;

3. massimizza lo score di probabilità sulle distribuzioni normali della Point Cloud in movimento rispetto alla Point Cloud fissa;

4. ripete l'allineamento della Point Cloud in movimento con la Point Cloud fissa usando la nuova trasformazione dallo step precedente. Infine ripete l'ottimizzazione;

5. controlla che i parametri di tolleranza e il massimo numero di iterazioni non vengano superate.

1.3 Allineamento e unione

In quest'ultima fase, la Point Cloud viene allineata, attraverso una trasformazione rigida, a quella precedente (allineamento) e, infine, unite (unione). Per ottenere la mappa, con la traiettoria ricostruita dal sensore, è necessario applicare quanto visto finora con l'intera sequenza di Point Clouds raccolte. Infine, è importante una buona stima della trasformazione iniziale in quanto necessaria per inizializzare la registrazione.

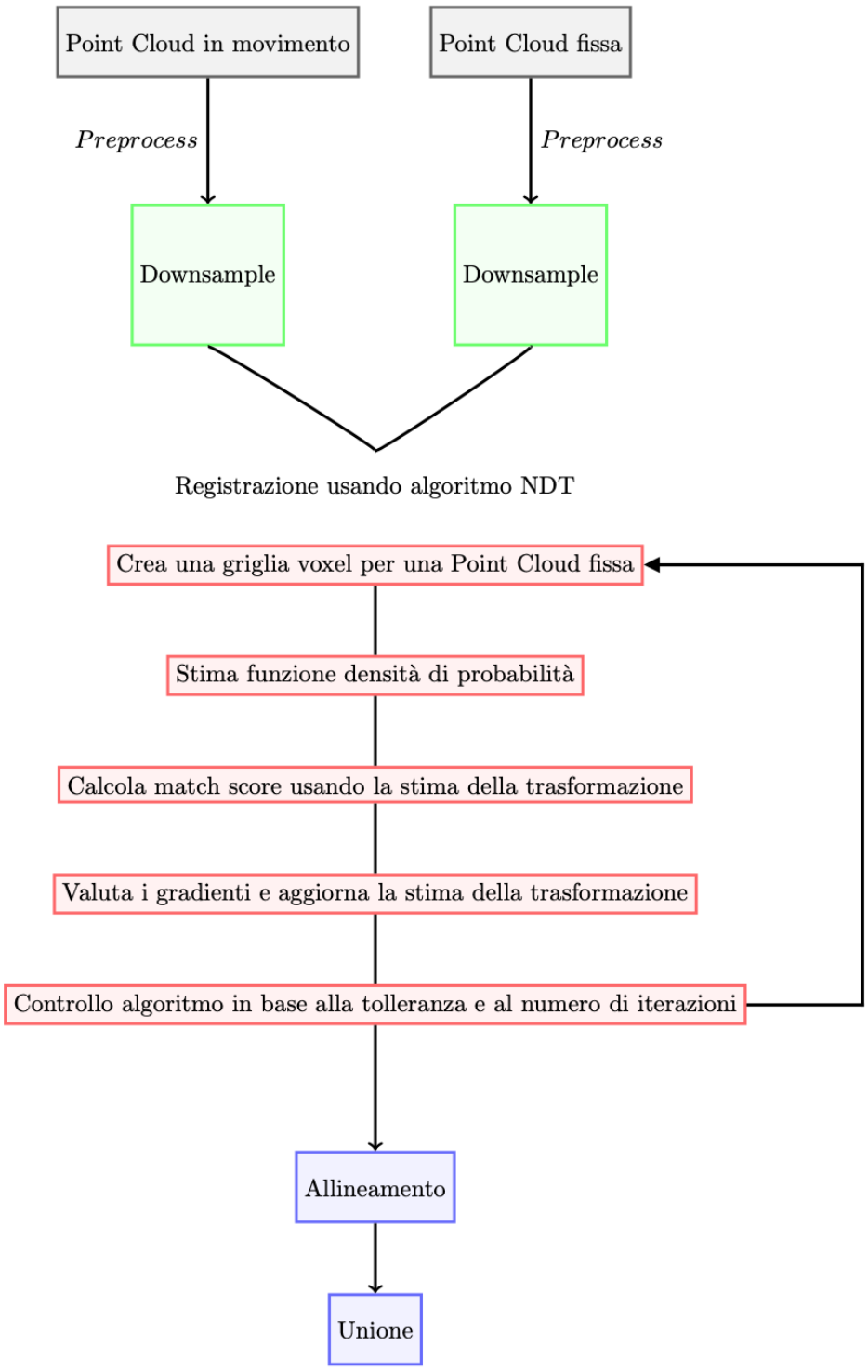

Figura 1.1: Metodo utilizzato

Capitolo 2

ALGORITMO NDT

2.1 Introduzione a NDT

La trasformazione delle distribuzioni normali rappresenta un metodo per descrivere in modo compatto una superficie. Questa viene delineata mediante una nuvola di punti e descritta come un insieme di funzioni di densità di probabilità locale (PDF). L'algoritmo NDT [4] suddivide lo spazio della scansione in una griglia di celle (quadrati nel caso 2D, o cubi nello spazio 3D). Per ogni cella viene calcolato un PDF e, successivamente, determinata la posizione dei punti $\vec{x}$ sulla superficie attraverso un processo casuale D-dimensionale. Quindi la probabilità risulta essere:

$$p\left(\vec{x}\right) = \frac{1}{\left(2\pi\right)^{\frac{D}{2}}\sqrt{|\sum|}} exp\left(-\frac{\left(\vec{x}-\vec{q}\right)^{T}\sum^{-1}\left(\vec{x}-\vec{q}\right)}{2}\right) \tag{2.1}$$

dove con $\vec{q}$ e $\sum$ vengono indicati il vettore medio e la matrice di covarianza dei punti della superficie di scansione di riferimento all'interno della cella in cui si trova $\vec{x}$. In particolare, il fattore di scala $\left(\left(2\pi\right)^{\frac{D}{2}}\sqrt{|\sum|}\right)^{-1}$ può essere scritto utilizzando una costante c_0. La media si calcola come

$$\vec{q} = \frac{1}{n}\sum_{i=1}^{n}\vec{x}_i \tag{2.2}$$

mentre la covarianza

$$\sum = \frac{1}{n-1}\sum_{i=1}^{n}\left(\vec{x}_i-\vec{q}\right)\left(\vec{x}_i-\vec{q}\right)^{T} \tag{2.3}$$

dove $\vec{x}_{i=1,\ldots,n}$ sono le posizioni dei punti della scansione di riferimento contenuti

nella cella. Le distribuzioni normali, attraverso derivate continue, danno una rappresentazione liscia a tratti della Point Cloud. La PDF offre un'approssimazione della superficie locale, descrivendone la relativa posizione e l'orientamento. Nel caso unidimensionale, partendo dall'equazione (2.1) e sostituendo $D = 1$, una variabile aleatoria x normalmente distribuita è caratterizzata da un valore atteso q e da una relativa incertezza espressa con una varianza σ

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\frac{(x-q)^2}{2\sigma^2}\right) \tag{2.4}$$

Nel caso multidimensionale, come vedremo in seguito, media e varianza sono invece descritte dal vettore $\vec{q}$ e dalla matrice di covarianza $\sum$. Gli elementi diagonali della matrice di covarianza denotano la varianza di ciascuna variabile, mentre gli elementi fuori dalla diagonale indicano la covarianza delle variabili. Gli autovettori e gli autovalori della matrice di covarianza descrivono, inoltre, l'orientamento della superficie. Nei casi 2D e 3D, l'orientamento e la levigatezza della superficie possono essere valutati dagli autovettori e dagli autovalori della matrice di covarianza. In particolare, gli autovettori descrivono le componenti principali della distribuzione rappresentando, quindi, un insieme di vettori ortogonali corrispondenti alle direzioni dominanti della covarianza delle variabili.

2.1.1 Registrare una scansione usando NDT

Per ogni scansione registrata, utilizzando l'algoritmo NDT, si vuole ottenere la pose che massimizza la probabilità che i punti si trovino sulla superficie della scansione di riferimento. Introduciamo, quindi, un vettore $\vec{p}$ contenente i parametri da ottimizzare, rotazione e traslazione. Definiamo, inoltre, una Point Cloud $X = \{\vec{x}_1, ..., \vec{x}_n\}$ che rappresenta la scansione corrente. Si può, allora, individuare una funzione di trasformazione spaziale $T(\vec{p}, \vec{x})$ che descrive lo spostamento nello spazio di un punto $\vec{x}$ in base alla pose $\vec{p}$. Per esempio, data l'equazione (2.1), la pose migliore risulta essere quella che massimizza la funzione di probabilità:

$$\Psi = \prod_{k=1}^{n} p\left(T\left(\vec{p}, \vec{x}_k\right)\right) \tag{2.5}$$

oppure, minimizza la probabilità logaritmica negativa di Ψ:

$$-log\ \Psi = -\sum_{k=1}^{n} log\left(p\left(T\left(\vec{p}, \vec{x}_k\right)\right)\right) \tag{2.6}$$

Utilizziamo, ora, una funzione combinazione di una distribuzione normale e una

distribuzione uniforme:

$$\bar{p}\left(\vec{x}\right) = c_1 exp\left(-\frac{\left(\vec{x}-\vec{q}\right)^T \sum^{-1}\left(\vec{x}-\vec{q}\right)}{2}\right) + c_2 p_0 \tag{2.7}$$

dove indichiamo con p_0 il rapporto atteso di valori anomali. Le costanti c_1 e c_2 possono essere determinate imponendo la funzione di probabilità $\bar{p}\left(\vec{x}\right)$ uguale a 1 all'interno dello spazio occupato dalla cella. La funzione logaritmica di probabilità

$$\bar{p}\left(x\right) = -log\left(c_1 exp\left(\frac{-x^2}{(2\sigma^2)}\right) + c_2\right) \tag{2.8}$$

può essere approssimata dalla gaussiana

$$\widetilde{p}\left(x\right) = d_1 exp\left(\frac{-d_2 x^2}{(2\sigma^2)}\right) + d_3 \tag{2.9}$$

scrivendo d_i in modo che $\widetilde{p}\left(x\right)$ abbia lo stesso andamento di $p\left(x\right)$ per $x = 0$, $x = \sigma$ e $x = \infty$. Otteniamo, dunque:

$$d_1 = -log\left(c_1 + c_2\right) - d_3 \tag{2.10}$$

$$d_2 = -2log\left(\frac{\left(-log\left(c_1 exp\left(-\frac{1}{2}\right) + c_2\right) - d_3\right)}{d_1}\right) \tag{2.11}$$

$$d_3 = -log\left(c_2\right) \tag{2.12}$$

L'influenza di un punto dalla scansione corrente sulla funzione score NDT risulta:

$$\widetilde{p}\left(\vec{x}_k\right) = -d_1 exp\left(-\frac{d_2}{2}\left(\vec{x}_k - \vec{q}_k\right)^T \sum_k^{-1}\left(\vec{x}_k - \vec{q}_k\right)\right) \tag{2.13}$$

dove $\vec{q}_k$ indica la media e $\sum_k$ la matrice di covarianza della cella NDT in cui si trova $\vec{x}_k$. Scriviamo ora la funzione NDT score $s\left(\vec{p}\right)$ che definisce la probabilità che i punti $\vec{x}_k$ stiano sulla superficie della scansione di riferimento, trasformata da $\vec{p}$:

$$s\left(\vec{p}\right) = -\sum_{k=1}^{n} \widetilde{p}\left(T\left(\vec{p}, \vec{x}_k\right)\right) \tag{2.14}$$

Un punto critico della funzione di probabilità è rappresentato dalla necessità di calcolare l'inversa della matrice di covarianza, $\sum^{-1}$. Quest'ultima, infatti, nel caso di punti complanari o collineari, è singolare e non può essere invertita. Nel caso 3D, dato che la matrice di covarianza calcolata con 3 punti o meno risulterà sempre singolare, la PDF viene calcolata solo per le celle contenenti almeno 5 punti.

2.1.2 Algoritmo di Newton

È possibile utilizzare l'algoritmo di Newton per trovare i parametri $\vec{p}$ che ottimizzano $s(\vec{p})$. Data la matrice hessiana H e il vettore gradiente $\vec{g}$ di $s(\vec{p})$, risolviamo iterativamente l'equazione:

$$H\Delta\vec{p} = -\vec{g} \tag{2.15}$$

Per ogni iterazione, l'incremento $\Delta\vec{p}$ viene aggiunto alla stima della pose corrente, in modo che $\vec{p} \leftarrow \vec{p} + \Delta\vec{p}$. Definiamo ora con $\vec{x}'_k$ il punto $\vec{x}_k$ trasformato dai parametri della pose corrente, relativa al centro della PDF della cella a cui appartiene. In formula:

$$\vec{x}'_k \equiv T(\vec{p}, \vec{x}_k) - \vec{q}_k \tag{2.16}$$

Gli elementi g_i del vettore gradiente $\vec{g}$ sono:

$$g_i = \frac{\delta s}{\delta p_i} = \sum_{k=1}^{n} d_1 d_2 \vec{x}'^{\,T}_k \Sigma_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_i} exp\left(\frac{-d_2}{2} \vec{x}'^{\,T}_k \Sigma_k^{-1} \vec{x}'_k\right) \tag{2.17}$$

Gli elementi H_{ij}, invece, della matrice hessiana H:

$$\begin{aligned} H_{ij} &= \frac{\delta^2 s}{\delta p_i \delta p_j} \\ &= \sum_{k=1}^{n} d_1 d_2 exp\left(-\frac{d_2}{2} \vec{x}'^{\,T}_k \Sigma_k^{-1} \vec{x}'_k\right) \\ &* \left(-d_2 \left(\vec{x}'^{\,T}_k \Sigma_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_i}\right)\left(\vec{x}'^{\,T}_k \Sigma_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_j}\right) + \vec{x}'^{\,T}_k \Sigma_k^{-1} \frac{\delta^2 \vec{x}'_k}{\delta p_i \delta p_j} + \frac{\delta \vec{x}'^{\,T}_k}{\delta p_j} \Sigma_k^{-1} \frac{\delta \vec{x}'_k}{\delta p_i}\right) \end{aligned} \tag{2.18}$$

2.2 Introduzione a NDT 2D

Ripercorriamo ora quanto descritto in precedenza per il caso bidimensionale. L'algoritmo NDT [7] trasforma l'insieme discreto di punti 2D, ricostruiti da una singola scansione, in una densità di probabilità continua a tratti e differenziabile definita sul piano 2D. Questa densità di probabilità consiste in un insieme di distribuzioni normali che possono essere facilmente calcolate. Successivamente, per massimizzare la somma, viene abbinata all'NDT una seconda scansione. L'NDT, dunque, modella la distribuzione di tutti i punti 2D ricostruiti da una scansione laser mediante una raccolta di distribuzioni normali locali. Lo spazio 2D attorno, quindi, viene suddiviso regolarmente in celle di dimensioni costanti. Per ogni cella, che contiene almeno tre punti, si eseguono le seguenti operazioni:

1. raccogliere tutti i punti 2D $x_i = 1 \dots n$ contenuti nella cella
2. calcolare la media $q = \frac{1}{n} \sum_i x_i$
3. calcolare la matrice di covarianza $\sum = \frac{1}{n} \sum_i (x_i - q)(x_i - q)^t$

La probabilità di misurare un campione nel punto "x" contenuto in questa cella è ora definita dalla distribuzione normale $N(q, \sum)$:

$$p(x) \sim exp\left(-\frac{(x-q)^t \sum^{-1} (x-q)}{2}\right) \tag{2.19}$$

L'NDT, dunque, stabilisce una suddivisione regolare del piano. In particolare, rappresenta la probabilità di misurare un campione per ogni posizione all'interno della cella. Vediamo ora come allineare 2 scansioni laser.

2.2.1 Allineamento delle scansioni

La mappatura spaziale T tra due frames di coordinate è data da

$$T : \begin{pmatrix} x' \\ y\prime \end{pmatrix} = \begin{pmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{2.20}$$

dove $(t_x, t_y)^t$ descrive la traslazione, mentre con ϕ la rotazione tra due frames. L'obiettivo dell'allineamento della scansione è recuperare questi parametri utilizzando le scansioni laser eseguite in due posizioni. Il procedimento è il seguente:

1. costruire l'NDT della prima scansione
2. inizializzare la stima dei parametri (da zero o usando i dati di odometria)

3. per ogni campione della seconda scansione: mappare il punto 2D ricostruito nelle coordinate del frame della prima scansione secondo i parametri
4. determinare le corrispondenti distribuzioni normali per ogni punto mappato
5. lo score per i parametri viene determinato valutando la distribuzione per ciascun punto mappato e sommando il risultato
6. calcolare una nuova stima del parametro cercando di ottimizzare il risultato ottenuto al punto precedente attraverso l'algoritmo di Newton
7. tornare al punto 3 fino a raggiungere convergenza.

Definiamo quindi:

- il vettore dei parametri da stimare $p = (p_i)^t_{i=1..3} = (t_x, t_y, \phi)^t$
- x_i: il punto 2D ricostruito del campione della scansione laser "i" della seconda scansione nelle coordinate del frame della seconda scansione
- x'_i: il punto "x_i" mappato nel frame di coordinate della prima scansione in base al vettore dei parametri p , in modo che $x'_i = T(x_i, p)$
- $\sum_i, q_i$: la matrice di covarianza e la media della corrispondente distribuzione normale per il punto x'_i

La mappatura, in base al vettore p, è considerata ottimale se la somma che valuta le distribuzioni normali di tutti i punti x'_i con i parametri $\sum_i$ e q_i risulta essere un massimo. In particolare, definiamo questa somma come score di p:

$$score(p) = \sum_i \exp\left(\frac{-(x'_i - q_i)^t \sum_i^{-1} (x'_i - q_i)}{2}\right) \tag{2.21}$$

2.2.2 Ottimizzazione usando l'algoritmo di Newton

In generale un problema di ottimizzazione viene spesso descritto come problema di minimizzazione. In questo caso, vogliamo ottimizzare la funzione $-score$. Per mezzo dell'algoritmo di Newton, ricaviamo iterativamente i parametri $p = (p_i)^t$ che minimizzano una funzione f. Ogni iterazione risolve la seguente equazione:

$$H\Delta p = -g \tag{2.22}$$

dove g è il gradiente trasposto di f con entrate

$$g_i = \frac{\partial f}{\partial p_i} \tag{2.23}$$

e con H viene indicata l'hessiana di f con entrate

$$H_{ij} = \frac{\partial f}{\partial p_i \partial p_j} \tag{2.24}$$

La soluzione di questo sistema lineare risulta essere un incremento Δp, in modo che:

$$p \leftarrow p + \Delta p \tag{2.25}$$

se H è definita positiva, $f(p)$ diminuirà inizialmente nella direzione di Δp. In caso contrario, H viene sostituita da $H' = H + \lambda I$, con λ scelto in maniera opportuna al fine che H' sia definita positiva. Applichiamo ora la procedura appena vista alla funzione $-score$. Il gradiente e l'hessiana sono costruiti raccogliendo le derivate parziali di tutte le somme dell'equazione 2.21. Per semplificare la notazione, eliminiamo l'indice i per il numero di campione. Scriviamo, inoltre:

$$q = x'_i - q_i \tag{2.26}$$

Le derivate parziali di q rispetto a p sono uguali alle derivate parziali di x'_i. Quindi:

$$s = -exp\frac{-q^t \sum^{-1} q}{2} \tag{2.27}$$

Applichiamo la regola della catena e otteniamo le seguenti entrate del gradiente

$$\widetilde{g_i} = -\frac{\partial s}{\partial p_i} = -\frac{\partial s}{\partial q}\frac{\partial q}{\partial p_i} = q^t \Sigma^{-1} \frac{\partial q}{\partial p_i} exp\frac{-q^t \sum^{-1} q}{2} \tag{2.28}$$

Le derivate parziali di q rispetto a p_i sono date dalla matrice jacobiana J_T di T

$$J_T = \begin{pmatrix} 1 & 0 & -xsin\phi - ycos\phi \\ 0 & 1 & xcos\phi - ysin\phi \end{pmatrix} \tag{2.29}$$

Le entrate della matrice hessiana sono date da:

$$\begin{aligned} \widetilde{H_{ij}} &= -\frac{\partial s}{\partial p_i \partial p_j} \\ &= -exp\frac{-q^t \sum^{-1} q}{2} \left(\left(-q^t \Sigma^{-1} \frac{\partial q}{\partial p_i} \right) \left(-q^t \Sigma^{-1} \frac{\partial q}{\partial p_j} \right) + \left(-q^t \Sigma^{-1} \frac{\partial^2 q}{\partial p_i \partial p_j} \right) \right. \\ &+ \left. \left(-\frac{\partial q^t}{\partial p_j} \Sigma^{-1} \frac{\partial q}{\partial p_i} \right) \right. \end{aligned} \tag{2.30}$$

Le derivate seconde di q sono:

$$\frac{\partial^2 q}{\partial p_i \partial p_j} = \begin{cases} \begin{pmatrix} -xcos\phi + ysin\phi \\ -xsin\phi - ycos\phi \end{pmatrix} & i = j = 3 \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & altrimenti \end{cases} \tag{2.31}$$

In particolare, le funzioni trigonometriche dipendono solo da ϕ.

2.2.3 Tracciamento della posizione

Vediamo ora come, usando l'algoritmo appena descritto, possiamo tracciare la posizione corrente a partire da un tempo dato $t = t_{start}$. Il frame delle coordinate di riferimento globale è definito dal frame delle coordinate locale. Il tracciamento viene eseguito rispetto al fotogramma chiave ovvero la rispettiva scansione laser. Al tempo t , l'algoritmo svolge i seguenti passaggi:

1. sia δ la stima dello spostamento tra il tempo t_{k-1} e t_k
2. mappa la stima della posizione al tempo t_{k-1} in accordo con δ
3. esegue l'algoritmo di ottimizzazione usando la scansione corrente, l'NDT del fotogramma chiave e la stima della nuova posizione
4. controlla se il fotogramma chiave è abbastanza vicino alla scansione corrente.

In particolare, la scelta se una scansione è abbastanza vicina o meno si basa su un criterio empirico che coinvolge la traslazione e la distanza angolare tra il fotogramma chiave e il fotogramma corrente.

2.2.4 Implementazione in SLAM

Possiamo definire una mappa come una raccolta di fotogrammi chiave insieme alle loro posizioni globali. Per ogni scansione "i" nella mappa, vengono associati un angolo ϕ_i (o matrice di rotazione R_i) e un vettore di traslazione $(t_x, t_y)_i^t = T_i$. Essi descrivono la posizione della scansione "i" nel frame delle coordinate globali. In particolare, la posizione attuale del sistema è indicata da una matrice di rotazione R e da un vettore di traslazione T. La mappatura T' dal frame delle coordinate del sistema al frame delle coordinate della scansione "i" risulta essere:

$$T' : \begin{pmatrix} x' \\ y' \end{pmatrix} = R_i^t \left(R \begin{pmatrix} x \\ y \end{pmatrix} + T - T_i \right) \tag{2.32}$$

La mappatura di un punto 2D di una scansione è calcolato usando T'. Inoltre possiamo osservare un aumento di complessità riguardante la matrice jacobiana e le derivate parziali seconde di T'. Dunque la jacobiana e la mappatura sono date da:

$$J_{T'} = R_i^t J_T \tag{2.33}$$

mentre le derivate parziali seconde di T':

$$\frac{\partial^2 x'}{\partial p_i \partial p_j} = \begin{cases} R_i^t \begin{pmatrix} -xcos\phi + ysin\phi \\ -xsin\phi - ycos\phi \end{pmatrix} & i = j = 3 \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & altrimenti \end{cases} \tag{2.34}$$

Per l'algoritmo di ottimizzazione, il gradiente e l'hessiana possono essere costruiti sommando tutte le scansioni sovrapposte. Ad ogni passo, la mappa consiste in un insieme di fotogrammi chiave con le loro "poses" (posizioni e orientamento) nel fotogramma delle coordinate globali. Se la sovrapposizione della scansione corrente con la mappa risulta essere troppo piccola, la mappa viene estesa dall'ultima scansione abbinata con successo. Quindi ogni scansione sovrapposta viene abbinata separatamente al nuovo fotogramma chiave, ottenendo la pose relativa tra le due scansioni. Con l'aggiunta di un nuovo fotogramma chiave, la mappa viene perfezionata ottimizzando una funzione di errore sulla base dei parametri di tutti i fotogrammi chiave. I risultati della registrazione a coppie vengono utilizzati per definire un modello di errore quadratico per ciascuna coppia abbinata. I parametri globali di due scansioni definiscono anche la loro pose relativa. Sia Δp la differenza tra la pose relativa definita dai parametri globali e la pose relativa definita dal risultato della corrispondenza delle coppie. Utilizzando il modello quadratico, derivato da un'espansione di Taylor dello "score" intorno a $\Delta p = 0$, definiamo lo "score" di queste due scansioni in funzione di Δp:

$$score'(\Delta p) = score + \frac{1}{2}(\Delta p)^t H(\Delta p) \tag{2.35}$$

Quando l'abbinamento a coppie converge e quindi si ottiene l'hessiana, otteniamo lo "score" finale. In particolare, manca il termine lineare perché ci siamo espansi su un punto di estremità. Infine, se il numero di fotogrammi chiave "N" aumenta, questa minimizzazione non può più essere eseguita in tempo reale in quanto il numero di parametri liberi è $3N - 3$.

2.3 Implementazione al caso 3D

Per applicare l'algoritmo NDT al caso 3D, è necessario modificare la funzione di trasformazione spaziale $T(\vec{p}, \vec{x})$ e le relative derivate parziali. Utilizzando, quindi, gli angoli di Eulero 3D si ottengono 6 parametri di trasformazione da ottimizzare: 3 traslazioni e 3 rotazioni. La pose, dunque, è individuata dal seguente vettore:

$$\vec{p}_6 = [t_x, t_y, t_z, \phi_x, \phi_y, \phi_z]^T$$

Si può scrivere ora la funzione di trasformazione spaziale T come segue, usando la sequenza di Eulero $z - y - x$

$$T_E\left(\vec{p}_6, \vec{x}\right) = R_x R_y R_z \vec{x} + \vec{t} = \begin{bmatrix} c_y c_z & -c_y s_z & s_y \\ c_x s_z + s_x s_y c_z & c_x c_z - s_x s_y s_z & -s_x c_y \\ s_x s_z - c_x s_y c_z & c_x s_y s_z + s_x c_z & c_x c_y \end{bmatrix} \vec{x} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

dove $c_i = \cos\phi_i$ e $s_i = \sin\phi_i$.

La derivata prima $\left(\frac{\delta}{\delta p_i}\right) T_E\left(\vec{p}_6, \vec{x}\right)$ corrisponde alla colonna i della matrice jacobiana

$$J_E = \begin{bmatrix} 1 & 0 & 0 & 0 & c & f \\ 0 & 1 & 0 & a & d & g \\ 0 & 0 & 1 & b & e & h \end{bmatrix}$$

dove

$a = x_1\left(-s_x s_z + c_x s_y c_z\right) + x_2\left(-s_x c_z - c_x s_y s_z\right) + x_3\left(-c_x c_y\right),$

$b = x_1\left(c_x s_z + s_x s_y c_z\right) + x_2\left(-s_x s_y s_z + c_x c_z\right) + x_3\left(-s_x c_y\right),$

$c = x_1\left(-s_y c_z\right) + x_2\left(s_y s_z\right) + x_3\left(c_y\right),$

$d = x_1\left(s_x c_y c_z\right) + x_2\left(-s_x c_y s_z\right) + x_3\left(s_x s_y\right),$

$e = x_1\left(-c_x c_y c_z\right) + x_2\left(c_x c_y s_z\right) + x_3\left(-c_x s_y\right),$

$f = x_1\left(-c_y s_z\right) + x_2\left(-c_y c_z\right),$

$g = x_1\left(c_x c_z - s_x s_y s_z\right) + x_2\left(-c_x s_z - s_x s_y c_z\right),$

$h = x_1\left(s_x c_z + c_x s_y s_z\right) + x_2\left(c_x s_y c_z - s_x s_z\right),$

Infine, la derivata seconda $\left(\left(\frac{\delta^2}{\delta p_i \delta p_j}\right)\right) T_E\left(\vec{p}_6, \vec{x}\right)$ corrisponde all'elemento $\vec{H}_{ij}$ della matrice a blocchi simmetrica

$$H_E = \begin{bmatrix} \vec{H}_{11} & \cdots & \vec{H}_{16} \\ \vdots & \ddots & \vdots \\ \vec{H}_{61} & \cdots & \vec{H}_{66} \end{bmatrix} = \begin{bmatrix} \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{0} & \vec{0} & \vec{a} & \vec{b} & \vec{c} \\ \vec{0} & \vec{0} & \vec{0} & \vec{b} & \vec{d} & \vec{e} \\ \vec{0} & \vec{0} & \vec{0} & \vec{c} & \vec{e} & \vec{f} \end{bmatrix}$$

$$\vec{a} = \begin{bmatrix} 0 \\ x_1\left(-c_x s_z - s_x s_y c_z\right) + x_2\left(-c_x c_z + s_x s_y s_z\right) + x_3\left(s_x c_y\right) \\ x_1\left(-s_x s_z + c_x s_y c_z\right) + x_2\left(-c_x s_y s_z - s_x c_z\right) + x_3\left(-c_x c_y\right) \end{bmatrix}$$

$$\vec{b} = \begin{bmatrix} 0 \\ x_1\left(c_x c_y c_z\right) + x_2\left(-c_x c_y s_z\right) + x_3\left(c_x s_y\right) \\ x_1\left(s_x c_y c_z\right) + x_2\left(-s_x c_y s_z\right) + x_3\left(s_x s_y\right) \end{bmatrix}$$

$$\vec{c} = \begin{bmatrix} 0 \\ x_1\left(-s_x c_z - c_x s_y s_z\right) + x_2\left(-s_x s_z - c_x s_y c_z\right) \\ x_1\left(c_x c_z - s_x s_y s_z\right) + x_2\left(-s_x s_y c_z - c_x s_z\right) \end{bmatrix}$$

$$\vec{d} = \begin{bmatrix} x_1\left(-c_y c_z\right) + x_2\left(c_y s_z\right) + x_3\left(-s_y\right) \\ x_1\left(-s_x s_y c_z\right) + x_2\left(s_x s_y s_z\right) + x_3\left(s_x c_y\right) \\ x_1\left(c_x s_y c_z\right) + x_2\left(-c_x s_y s_z\right) + x_3\left(-c_x c_y\right) \end{bmatrix}$$

$$\vec{e} = \begin{bmatrix} x_1\left(s_y s_z\right) + x_2\left(s_y c_z\right) \\ x_1\left(-s_x c_y s_z\right) + x_2\left(-s_x c_y c_z\right) \\ x_1\left(c_x c_y s_z\right) + x_2\left(c_x c_y c_z\right) \end{bmatrix}$$

$$\vec{f} = \begin{bmatrix} x_1\left(-c_y c_z\right) + x_2\left(c_y s_z\right) \\ x_1\left(-c_x s_z - s_x s_y c_z\right) + x_2\left(-c_x c_z + s_x s_y s_z\right) \\ x_1\left(-s_x s_z + c_x s_y c_z\right) + x_2\left(-c_x s_y s_z - s_x c_z\right) \end{bmatrix}$$

Inoltre, per angoli inferiori a 10°, è possibile scrivere le seguenti approssimazioni:

$$\sin\phi \approx \phi$$

$$\cos\phi \approx 1$$

$$\phi^2 \approx 0$$

Capitolo 3

Strumentazione utilizzata

3.1 Lidar

3.1.1 Caratteristiche generali

I lidar, dall'inglese "light detecting and ranging", sono dispositivi capaci di rilevare la distanza da un oggetto utilizzando un fascio laser. In particolare hanno la peculiarità, data la lunghezza d'onda corta, di emettere raggi laser altamente focalizzati senza lobi laterali. La distanza può essere misurata nei seguenti modi:

- per triangolazione
- con il tempo di volo del fascio emesso
- utilizzando il phase shift

Nel primo caso, come si può vedere in figura 3.1, identifichiamo con r_2 la distanza dello spot illuminato dal fascio laser visto dal ricevitore. Si determina, dunque, l'angolo θ e quindi r conoscendo ϕ e r_3. Risulta difficile, però, misurare superfici con spigoli vivi dal momento che i punti laser sono identici. Il secondo metodo consiste nel misurare il tempo impiegato da un impulso laser, una volta emesso, a tornare indietro. Conoscendo il mezzo di propagazione, ad esempio l'aria, è possibile determinare la distanza percorsa. Infine, si può utilizzare lo sfasamento del fascio in entrata rispetto a quello in uscita da una superficie, come mostrato in figura 3.2. Indichiamo con ϕ lo sfasamento, ω_m la lunghezza d'onda di modulazione, C la velocità della luce e f_m la modulazione in frequenza. La distanza r è data da:

$$r = \frac{\phi \omega_m}{4\pi} = \frac{\phi C}{4\pi f_m} \tag{3.1}$$

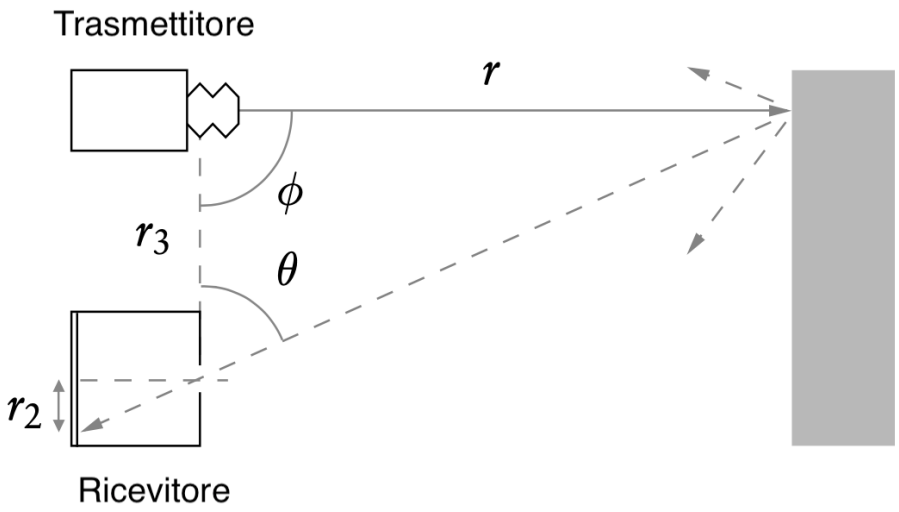

Figura 3.1: Triangolazione

Lo sfasamento può essere misurato elaborando i due segnali e calcolando la media del risultato su diversi cicli di modulazione. Esiste, però, un intervallo massimo dato dalla frequenza di modulazione dopo la quale il segnale "si avvolge".

3.1.2 Lidar Livox Horizon

Il sensore Lidar Livox Horizon [8], di dimensioni 77 x 115 x 84 mm, è composto, come si può vedere in figura 3.3, da:

- una finestra ottica, attraverso cui il fascio laser passa per scannerizzare l'ambiente circostante;
- modulo di auto-dissipazione, necessario a mantenere la temperatura di esercizio del sensore al di sotto di 85° C;
- presa d'aria sul retro, consente all'aria calda di uscire dal modulo di auto-dissipazione.

La massa, inoltre, è pari, includendo i cavi, a 1180 grammi. In particolare, è in grado di rilevare corpi a una lunga distanza (fino a 260 metri con oggetti con riflettività dell'80%), con un campo di vista (FoV) orizzontale di 81.7° e 25 .1° verticale, come mostrato in figura 3.4. L'intervallo di tensione, durante la fase operativa, è pari a 10-15 V. La potenza, invece, in condizioni normali di utilizzo,

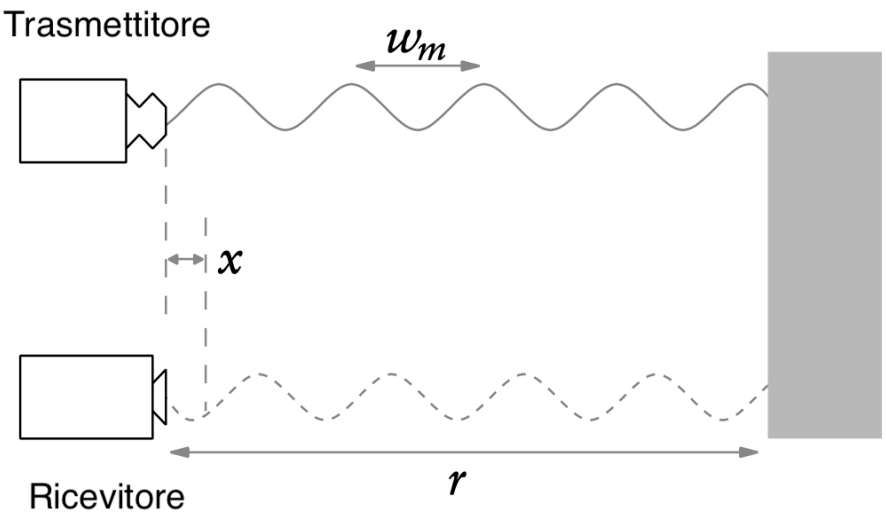

Figura 3.2: Phase Shift

raggiunge i 12 W. Nel caso estremo in cui la temperatura raggiunga valori da -40° a -20° C, il sensore entra in modalità di auto-riscaldamento per una durata complessiva di 3 minuti. I dati sono forniti mediante Point Cloud, raccolta dei punti appartenenti alle superfici degli oggetti individuati all'interno del campo di vista (FoV) del sensore. Contengono informazioni quali:

- riflettività, descritta con un valore appartenente al range 0-255. Da 0 a 150 si assume valido il modello di riflessione lambertiana, mentre per valori da 151 a 255 vengono individuati oggetti con proprietà di retroriflessione;
- coordinate cartesiane (x, y, z) e sferiche $(r,\ \theta,\ \phi)$. In particolare, esiste la relazione descritta in figura 3.5.

Nel caso in cui non venga rilevato alcun oggetto, le coordinate cartesiane assumeranno il valore (0,0,0) mentre quelle sferiche $(0,\ \theta,\ \phi)$. La tecnologia di scansione è illustrata in figura 3.6. L'immagine rappresenta la Point Cloud che rientra nel FoV in $0.1s$. Si può notare come la scansione nella zona centrale risulti essere più densa, con uno spazio tra le linee di circa 0.2°. Le due zone laterali circolari, invece, presentano una densità inferiore, con una distanza tra le linee pari a 0.4°. É possibile calcolare la copertura del FoV con la seguente formula:

$$C = \frac{Area\ totale\ illuminata\ dai\ fasci\ laser}{Area\ totale\ in\ FoV} * \ 100\% \tag{3.2}$$

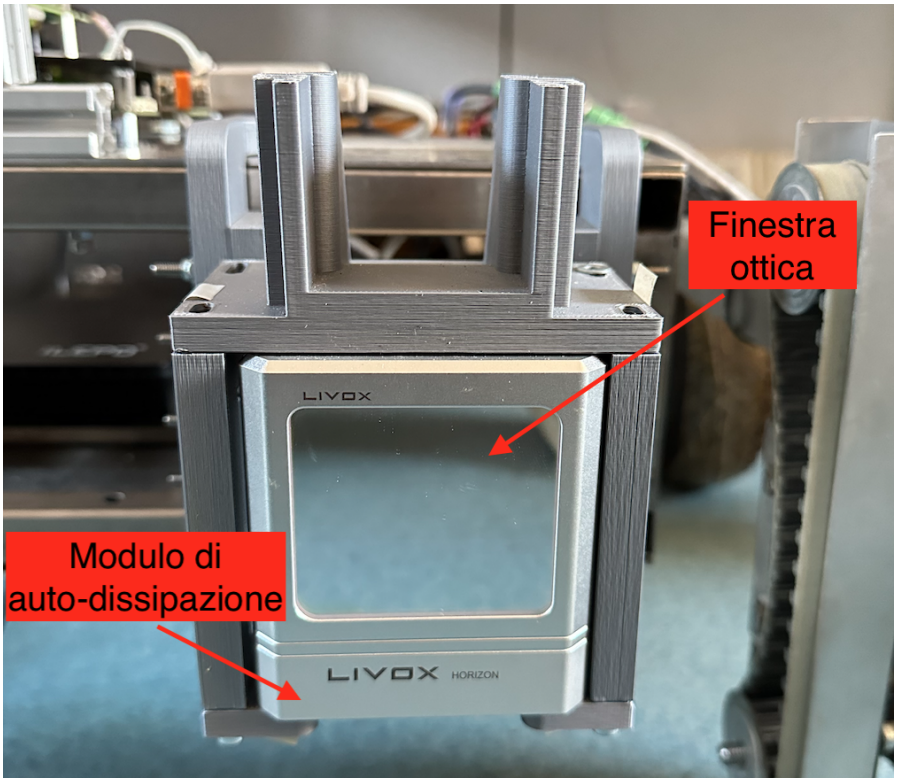

Figura 3.3: Lidar vista frontale

Aumentando il tempo di integrazione, C cresce raggiungendo circa il 100% in 0.5s.

3.1.3 Lidar Ouster

Il lidar Ouster OS1, "Mid-Range High-Resolution Imaging Lidar", è progettato per essere usato in molteplici ambiti quali guida autonoma, mappatura e, in generale, applicazioni nel campo della robotica. É caratterizzato da un range massimo di 200 metri e fino a 90 metri nel caso di target riflettente. Offre fino a 128 canali di risoluzione. La principale differenza del lidar Ouster OS1 [3], rappresentato in figura 3.7, rispetto al Livox Horizon, risulta essere il campo di vista orizzontale che è pari a 360°, mentre quello verticale è di 45° (da +22.5° a -22.5°). La tecnologia utilizzata, per la raccolta dati, si chiama "Multi-Beam Flash" Lidar: "Flash" in quanto ogni pixel del sensore è illuminato dal laser e raccoglie simultaneamente la luce, "Multi-Beam" perchè ogni scena è illuminata con fasci di luce di precisione. Durante l'utilizzo richiede una potenza di 14 W che può arrivare a 20 W a seconda della risoluzione impostata. La tensione nominale può essere di 12 V o 24 V. La

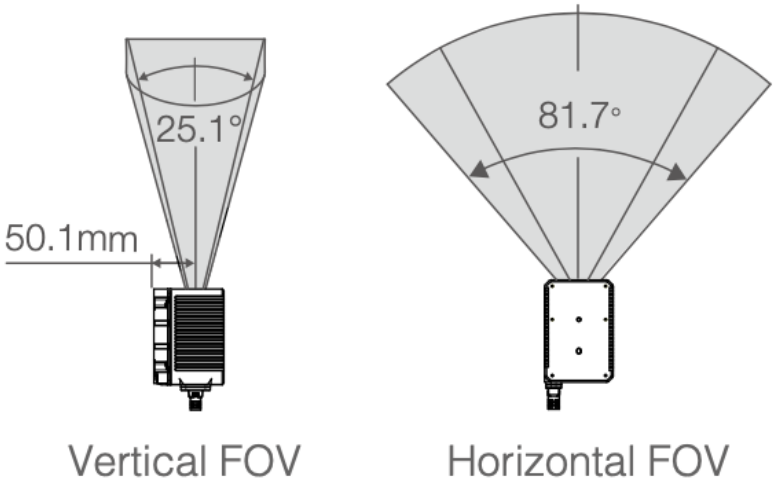

Figura 3.4: FoV lidar Livox Horizon

temperatura di esercizio ricade nell'intervallo da -40°C a +60°C. Il calore viene dissipato attraverso le alette poste sulla parte superiore e inferiore del sensore. La tipologia di scansione, a linee orizzontali, è rappresentata in figura 3.8. La risoluzione angolare verticale è pari a 0.35°.

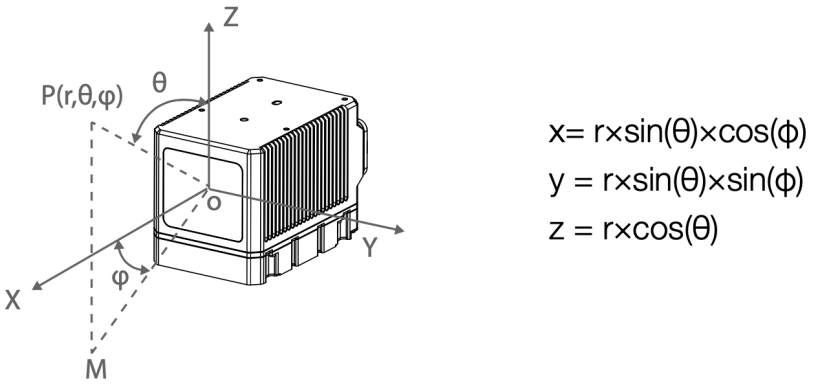

Figura 3.5: Relazione tra coordinate cartesiane e sferiche

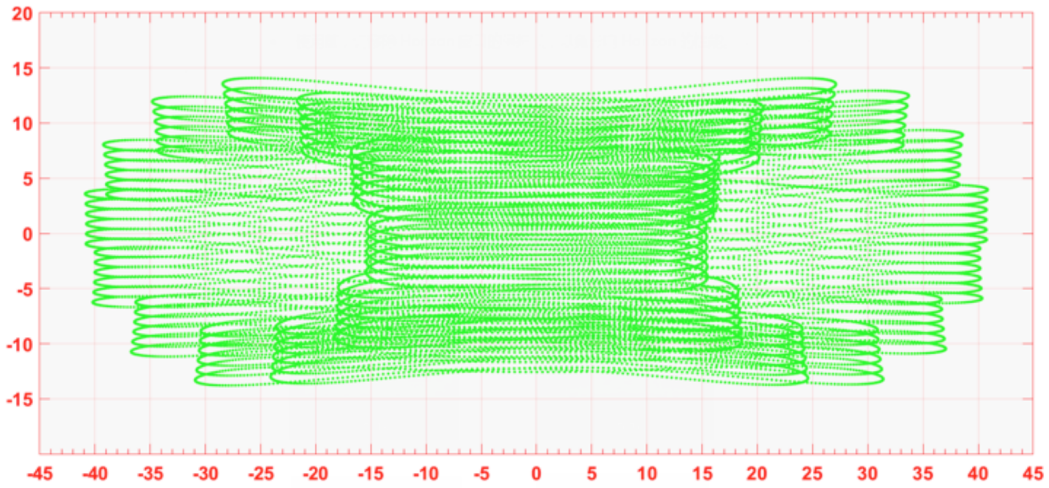

Figura 3.6: Livox Point Cloud in FoV in 0.1s

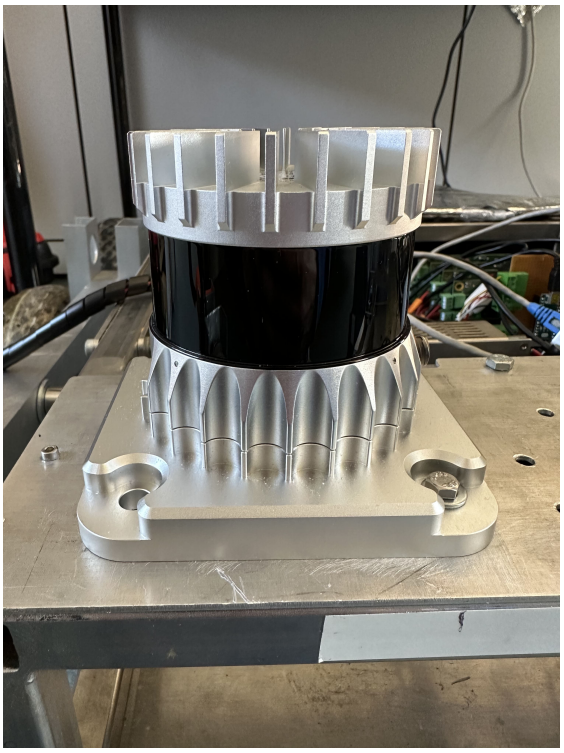

Figura 3.7: Lidar Ouster OS1

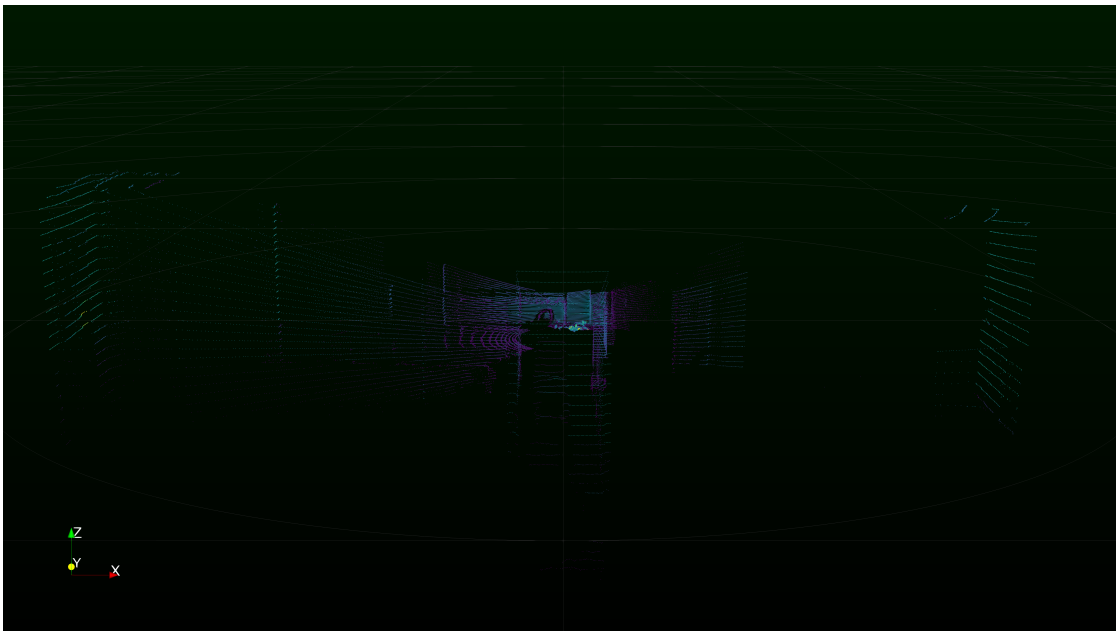

Figura 3.8: Ouster Point Cloud in FoV

3.2 GPS

3.2.1 Segnale

A partire dalla frequenza fondamentale $f_0 = 10.23\ MHz$, il segnale GPS viene modulato sulle portanti L1 e L2, appartenenti alla banda L (1.2 – 1.6 GHz). In particolare:

- $L1 = f0 * 154 = 1575.42\ MHz\ (\lambda = 19.05\ cm)$;
- $L2 = f0 * 120 = 1227.60\ MHz\ (\lambda = 24.45\ cm)$.

3.2.2 Schema di funzionamento

Come si può vedere in figura 3.9 e descritto in "Posizionamento Satellitare e Determinazione Orbitale" [1], la portante del segnale su L1 è data da una componente in fase e da una in quadratura di fase. La componente in fase viene modulata mediante modulazione BPSK (Binary Phase Shift Keying) da un segnale di 50 Hz, messaggio di navigazione, e da un Coarse Acquisition Code (C/A), codice pseudorandom. Quella in quadratura, invece, sempre attraverso modulazione BPSK ma utilizzando il Precision Code (P-Code), codice pseudorandom diverso. La portante L2, utilizzata soprattutto per scopi militari, viene modulata per mezzo del messaggio di navigazione e dal P-code.

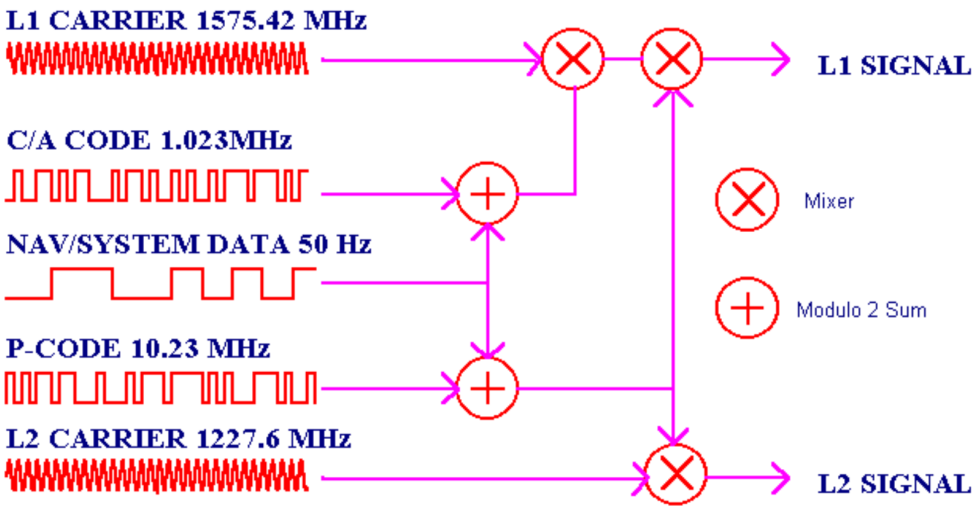

Figura 3.9: Modulazione portanti L1 e L2

Capitolo 4

Analisi sperimentali

4.1 Introduzione

Tutte le analisi sono state effettuate usando il rover del progetto studentesco Morpheus, mostrato in figura 4.1. In particolare, si è deciso di confrontare la traietto-

Figura 4.1: Rover Morpheus

ria percorsa dal rover, ricostruita con l'utilizzo del lidar Livox Horizon, con il gps operante su una singola banda di frequenza (L1). Infine viene proposto un confronto tra il lidar Livox Horizon e il lidar Ouster. Inizialmente sono stati presi in considerazione gli spazi esterni al Dipartimento di Ingegneria Industriale, per poi

successivamente percorrere la strada lungo l'argine del Piovego fino a raggiungere il ponte pedonale Parco Europa. In rosso è rappresentata la traiettoria descritta dal lidar, mentre in blu quella dal gps. Di seguito i risultati ottenuti.

4.2 Risultati utilizzando solo il Lidar Livox Horizon

4.2.1 Percorso a "C"

La prima analisi è stata condotta nel giardino davanti al Dipartimento di Ingegneria Industriale. In particolare, per questo caso, ripercorriamo le fasi del metodo, che saranno analoghe nelle analisi successive. Il rover ha compiuto un percorso a forma di "C". Il preprocess della Point Cloud in movimento e quella fissa porta al risultato in figura 4.2. L'ego vehicle, dunque il rover, e il ground plane, evidenziati

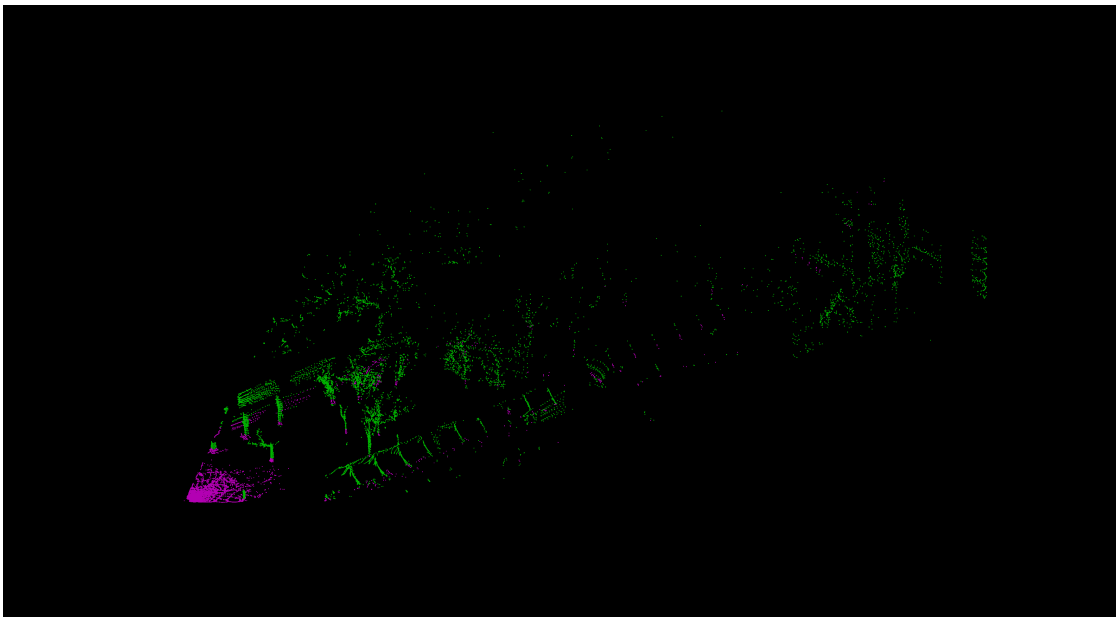

Figura 4.2: Preprocess percorso a "C"

col colore magenta, vengono rimossi. Si può notare che l'ambiente all'interno del FoV del lidar viene correttamente riconosciuto, come ad esempio il parcheggio per le biciclette sulla destra e gli alberi davanti al rover. Viene successivamente eseguito l'allineamento delle 2 Point Clouds, come mostrato in figura 4.3. Infine, in figura 4.4 vengono unite. La traiettoria è mostrata in figura 4.5. Proiettandola in Google Earth, si ottiene il risultato mostrato in figura 4.6. Il percorso descritto

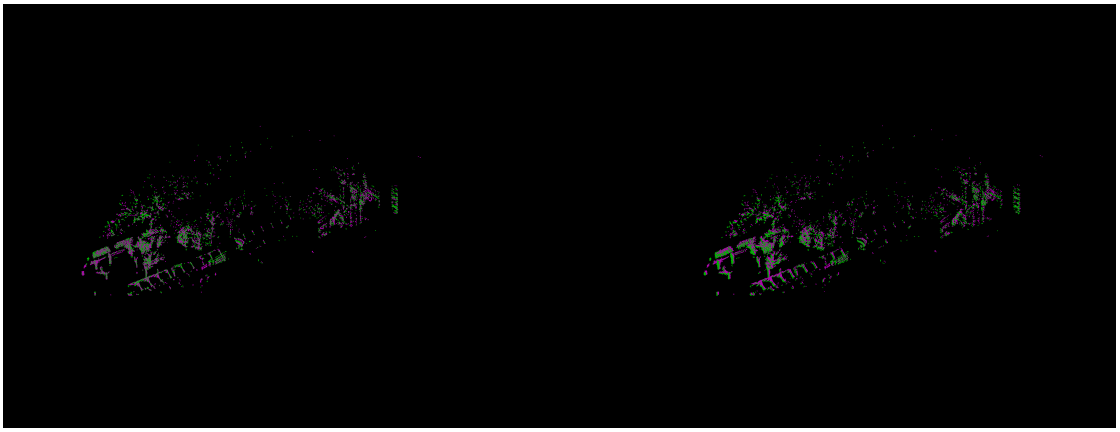

Figura 4.3: Allineamento Point Clouds: a sinistra prima della registrazione, a destra dopo

rientra completamente all'interno dell'area verde, evidenziando dunque bassi valori di drift.

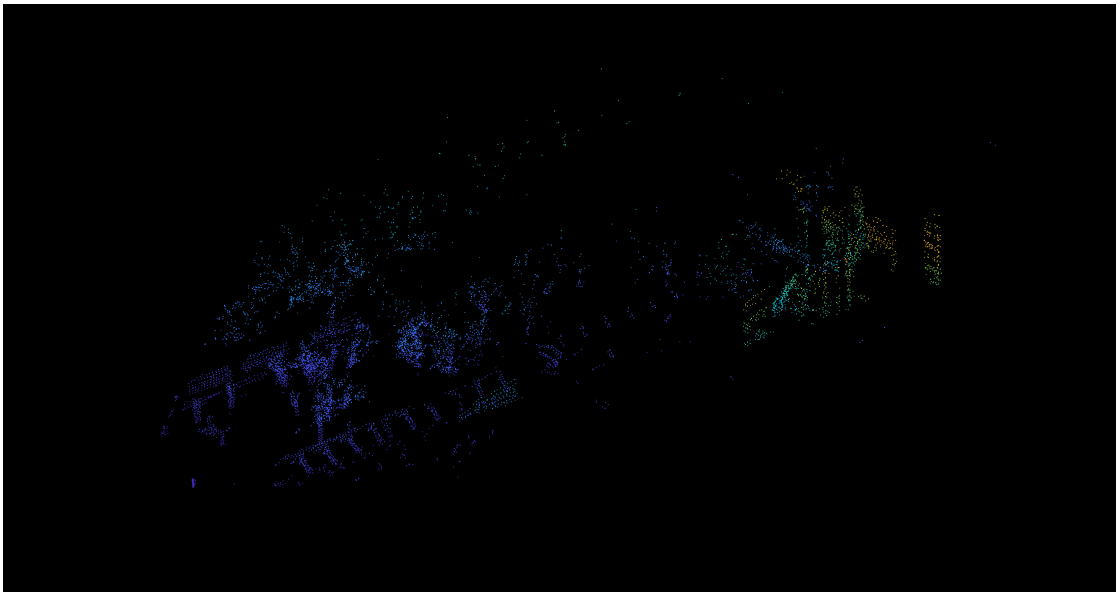

Figura 4.4: Unione delle 2 Point Clouds

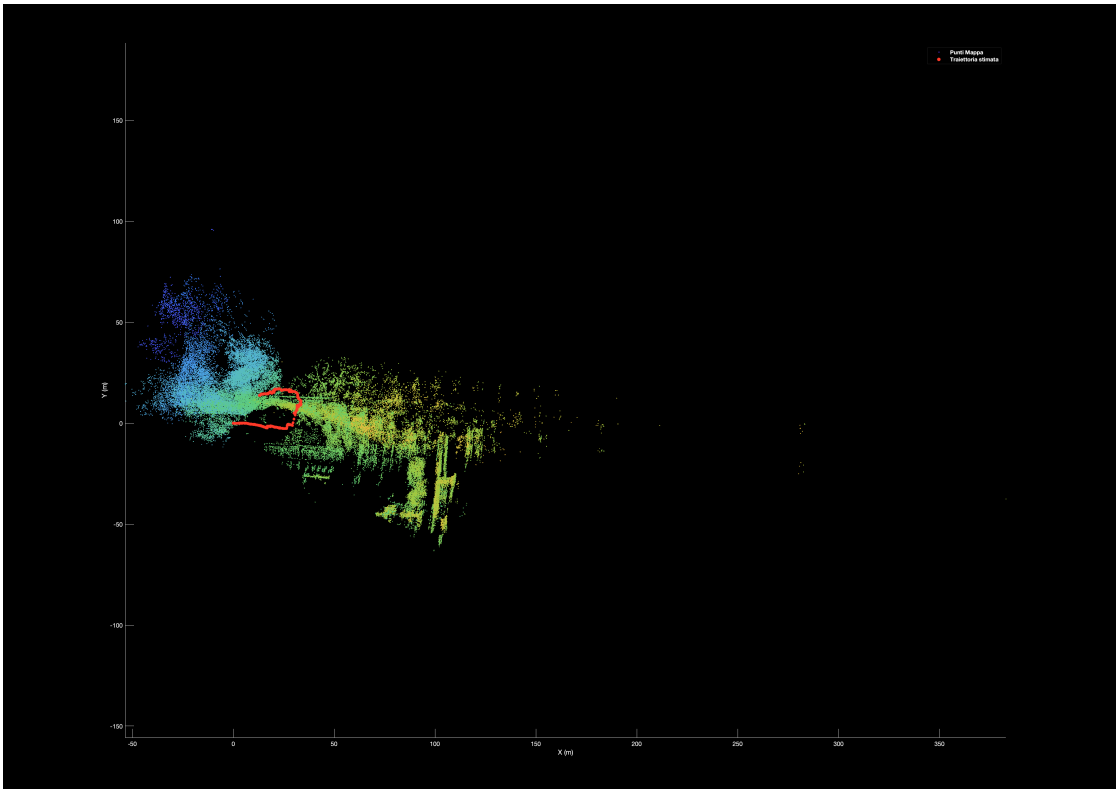

Figura 4.5: Traiettoria percorso a "C"

Figura 4.6: Proiezione percorso a "C" in Google Earth

4.3 Risultati ottenuti utilizzando il Lidar Livox Horizon e il gps

4.3.1 Uscita dal dipartimento di ingegneria industriale

L'analisi successiva offre un primo confronto tra i due strumenti utilizzati. Vediamo, quindi, i risultati ottenuti in figura 4.7 e 4.8. La traiettoria descritta dal

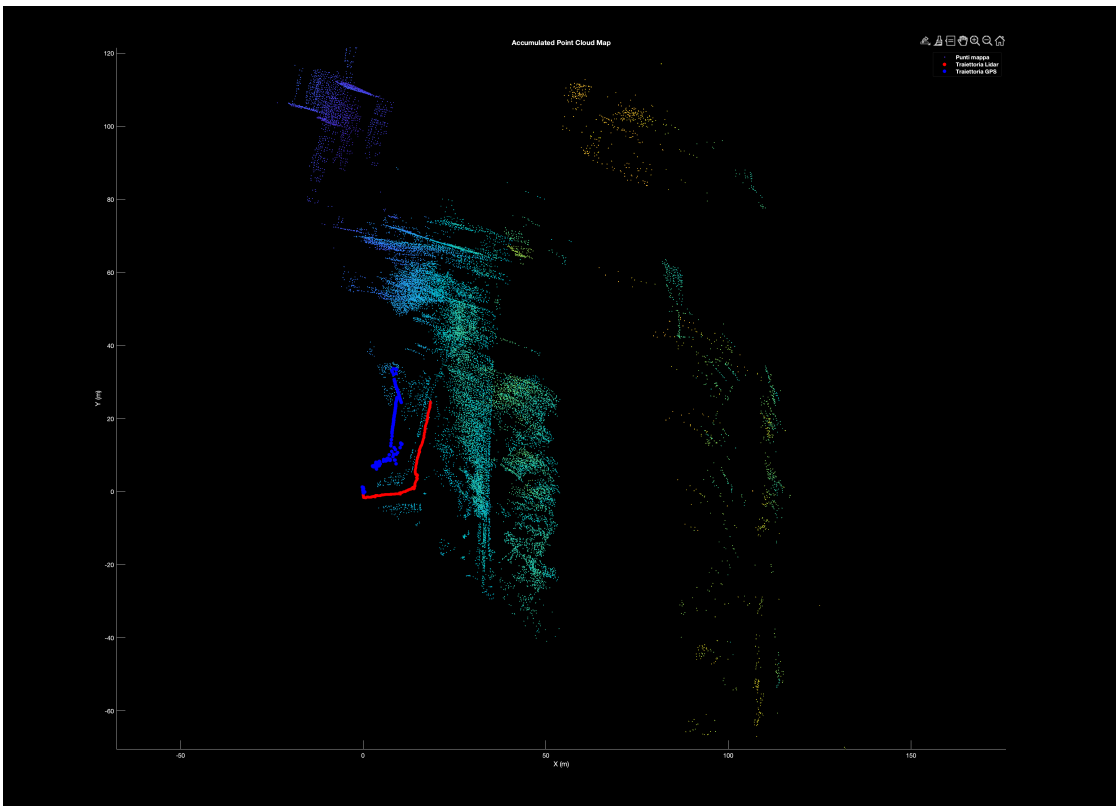

Figura 4.7: Traiettorie "Uscita dal Dipartimento di Ingegneria Industriale"

lidar ripercorre bene quella seguita. Il gps, invece, dimostra alcune difficoltà dovute principalmente alle limitazioni di natura tecnologica e alla presenza di alberi che ostacolano la ricezione del segnale.

4.3.2 Argine del Piovego

Passiamo ora a un'altra analisi condotta lungo l'argine del Piovego. In questo caso la traiettoria prevista è rettilinea. I risultati sono visibili in figura 4.9 e 4.10. Le traiettorie quasi coincidono nella prima parte per poi, verso la fine,

Figura 4.8: Proiezione traiettorie "Uscita dal Dipartimento di Ingegneria Industriale" in Google Earth

allontanarsi. Nell'ultimo tratto, infatti, il lidar è affetto da drift. Le distanze, effettivamente percorse, trovano riscontro con quelle ricavate sperimentalmente, evidenziando come nei tratti rettilinei l'algoritmo utilizzato per processare i dati lidar, e dunque le Point Clouds, restituisca buoni risultati.

4.3.3 Ponte pedonale Parco Europa

Il percorso è rappresentato in figura 4.11. Proiettando le traiettorie su Google Earth, otteniamo i risultati di figura 4.12. Nel ripercorrere il ponte, la traiettoria ricostruita con il lidar risulta essere più lineare, mentre quella descritta dal gps appare segmentata, evidenziando leggere differenze. Considerando plausibile un errore di precisione del gps di 1-2 metri, i risultati sono considerati buoni. Per completezza, viene proposta anche la traiettoria compiuta al ritorno. In figura 4.13 e 4.14 i risultati. Come mostrato in precedenza, la traiettoria del lidar risulta essere più accurata; il gps mostra, anche in questo caso, errori che non permettono di tracciare un percorso lineare. Complessivamente, i risultati sono comunque da considerare accettabili.

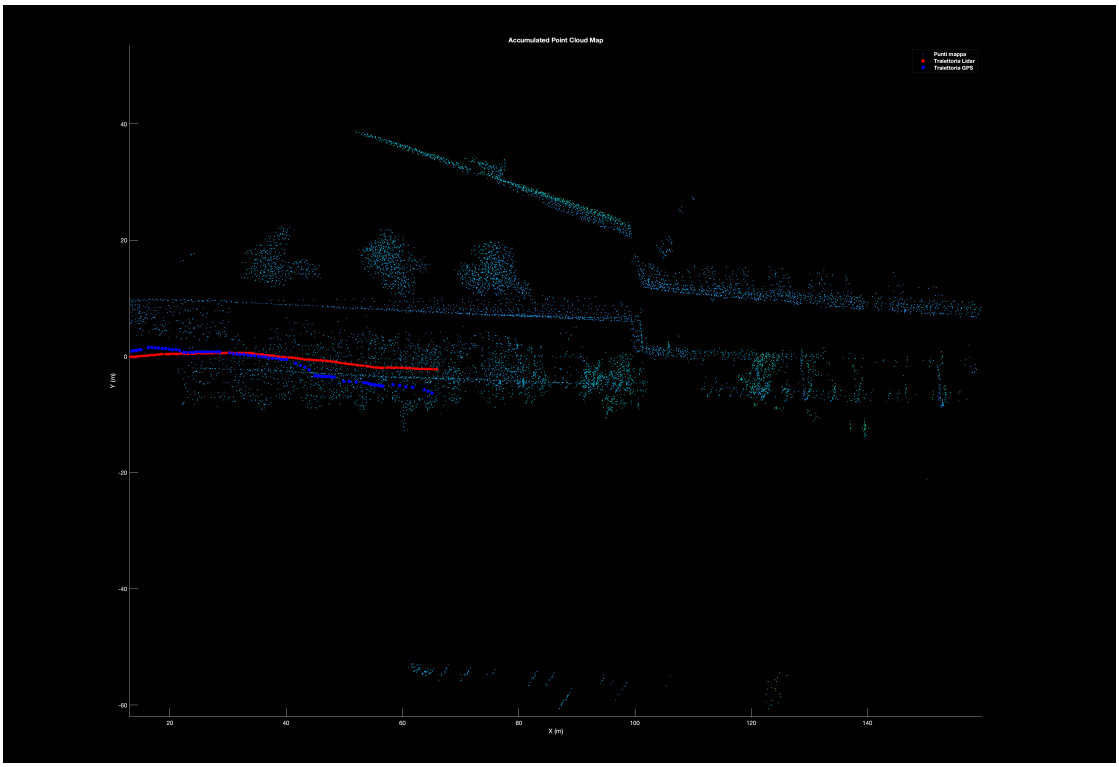

Figura 4.9: Traiettorie "Argine del Piovego"

Figura 4.10: Proiezione traiettorie "Argine del Piovego" in Google Earth

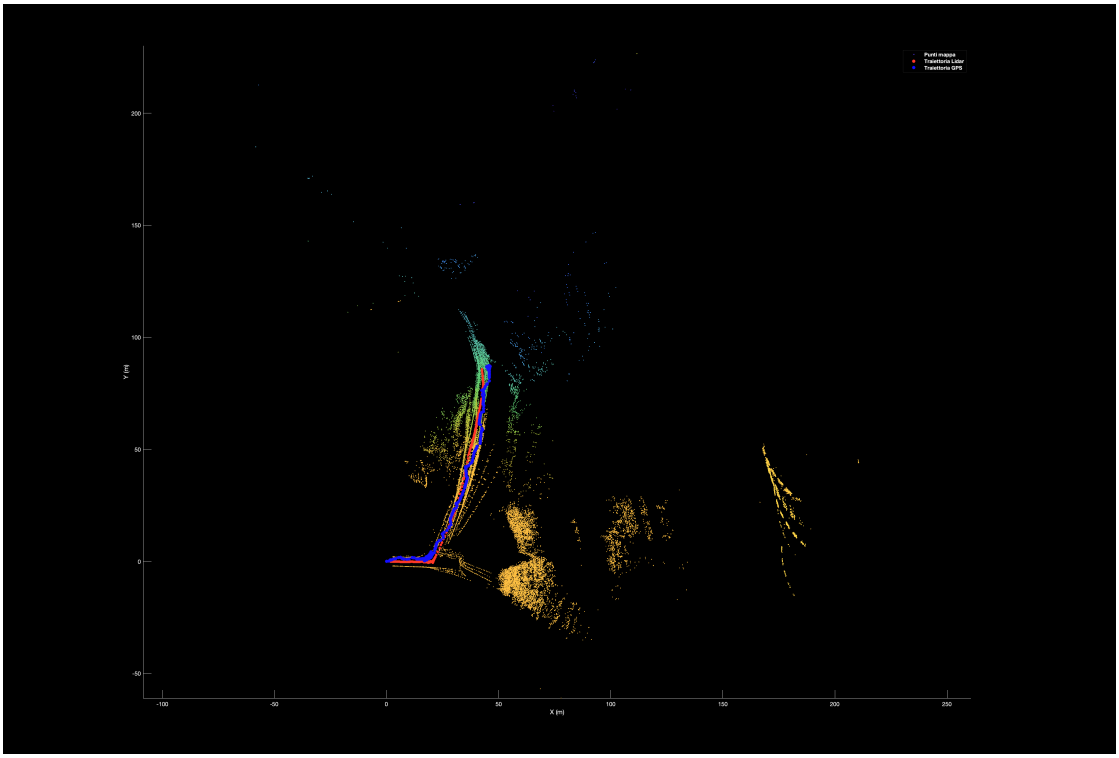

Figura 4.11: Traiettorie "Ponte andata"

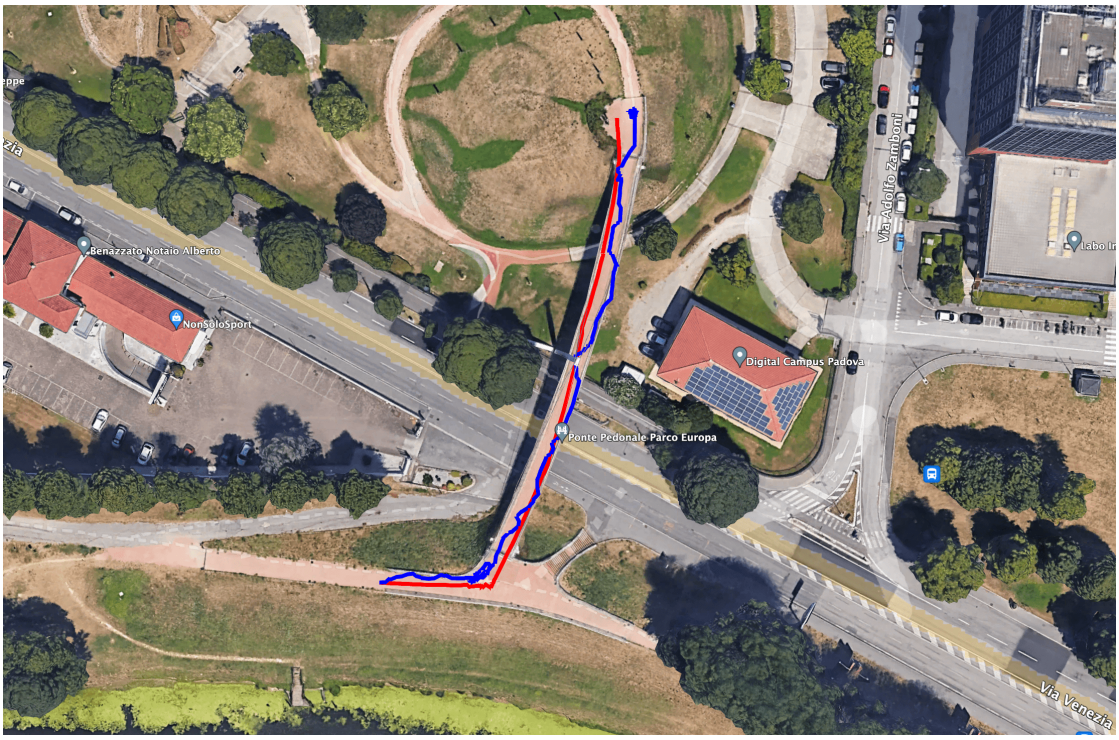

Figura 4.12: Proiezione traiettorie "Ponte andata" in Google Earth

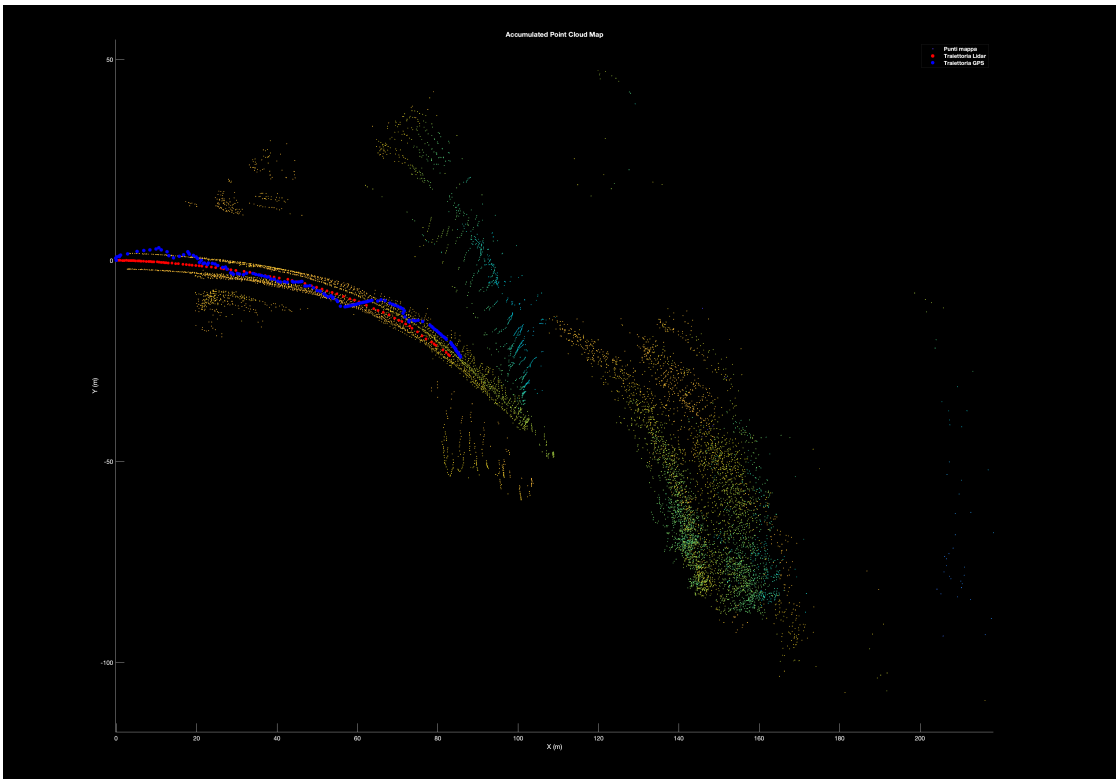

Figura 4.13: Traiettorie "Ponte ritorno"

Figura 4.14: Proiezione traiettorie "Ponte ritorno" in Google Earth

4.4 Confronto lidar Livox Horizon con lidar Ouster

L'ultima analisi proposta è un confronto tra il lidar Livox Horizon, usato nelle prove precedenti, e il lidar Ouster OS1. I dati sono stati raccolti all'interno del Dipartimento di Ingegneria Meccanica, in particolare muovendo il rover lungo i corridoi del terzo piano. Il FoV del lidar Ouster è stato ridotto di circa 20° per evitare di comparire nelle scansioni. Per completezza, per la prima acquisizione vengono riportati i passaggi del metodo scelto. In rosso è illustrata la traiettoria descritta dal lidar Livox Horizon, in verde quella dell'Ouster OS1.

4.4.1 Corridoio principale

Il rover ha percorso circa l'intera lunghezza del corridoio. L'analisi dei preprocess è raffigurata in figura 4.15. Come si può vedere, il FoV diverso dei due lidar viene

Figura 4.15: A sinistra: preprocess frame Livox Horizon, a destra preprocess frame Ouster

correttamente riconosciuto. Dunque l'allineamento delle due Point Clouds in figura 4.16. Le traiettorie sono riportate in figura 4.17. Entrambi i lidar percorrono una traiettoria rettilinea. Come si poteva immaginare, il Livox, dato il ristretto FoV, non riesce a costruire una mappa dei corridoi laterali, ricostruiti, invece, dall'Ouster. Infine, considerando le distanze sull'asse x e y, queste trovano riscontro, in maniera approssimata, con quelle del corridoio.

4.4.2 Curva a sinistra in fondo al corridoio

Nell'analisi successiva, il rover ha compiuto una curva verso sinistra alla fine del corridoio principale. In figura 4.18, le traiettorie. Anche in questo caso, il lidar

Ouster riesce, come in precedenza, a descrivere in maniera più esaustiva l'ambiente circostante rispetto al Livox. I riflessi delle finestre hanno creato alcuni problemi nel determinare la corretta orientazione del corridoio al lidar Livox, mentre il problema si è rivelato più contenuto per l'Ouster.

4.4.3 Curva a destra e ritorno in laboratorio

Infine, il rover ha percorso l'ultimo tratto del secondo corridoio, per poi immettersi in quello principale e raggiungere il laboratorio Morpheus. In figura 4.19, le traiettorie. Il lidar Livox riesce a ricostruire abbastanza bene i corridoi, mostrando però un po' di deriva nella parte iniziale. Dalla ricostruzione del lidar Ouster, riuscendo a determinare gli spazi circostanti, è possibile notare che il rover si ferma in corrispondenza del laboratorio.

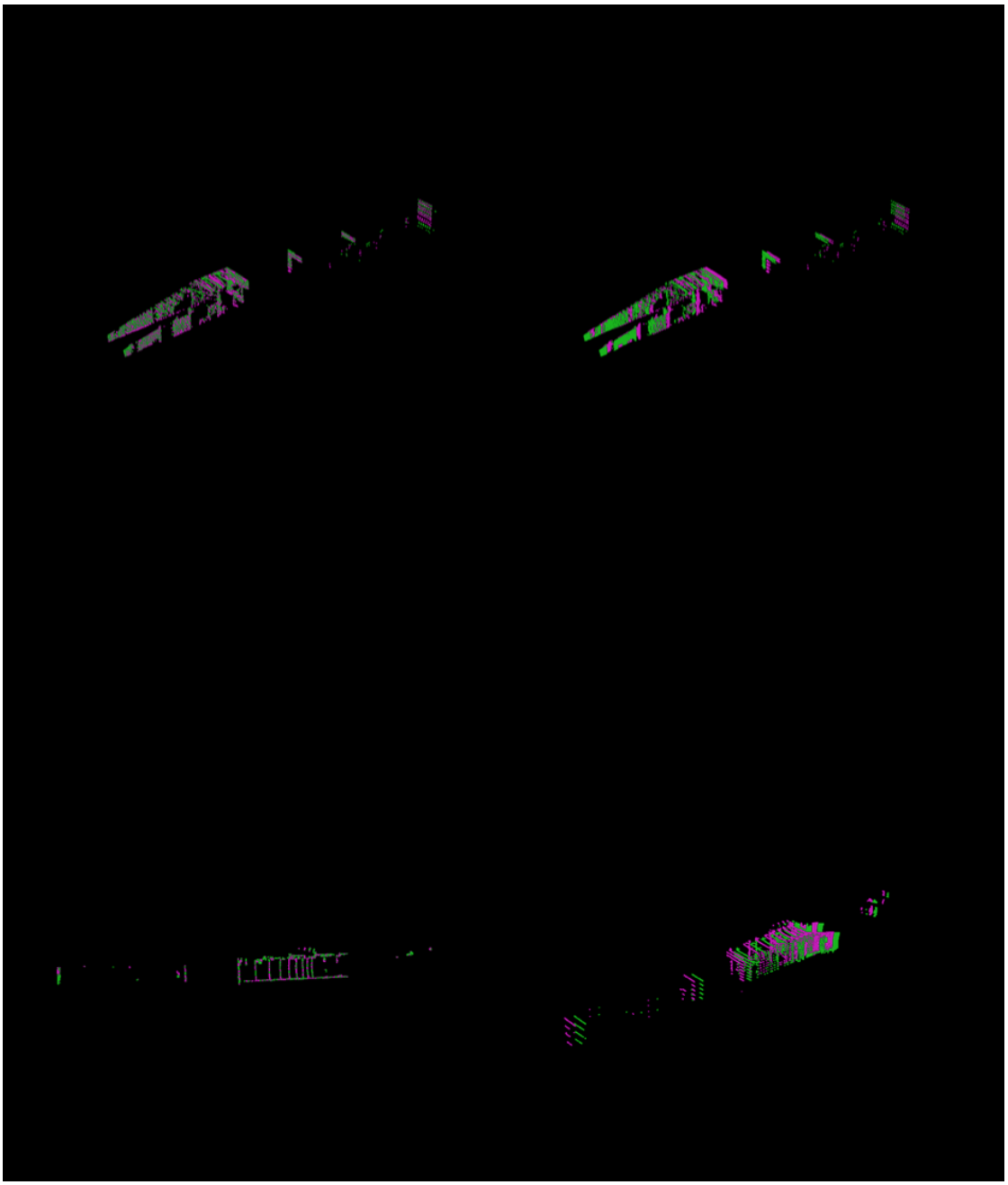

Figura 4.16: In alto: allineamento frame Livox Horizon, in basso allineamento frame Ouster

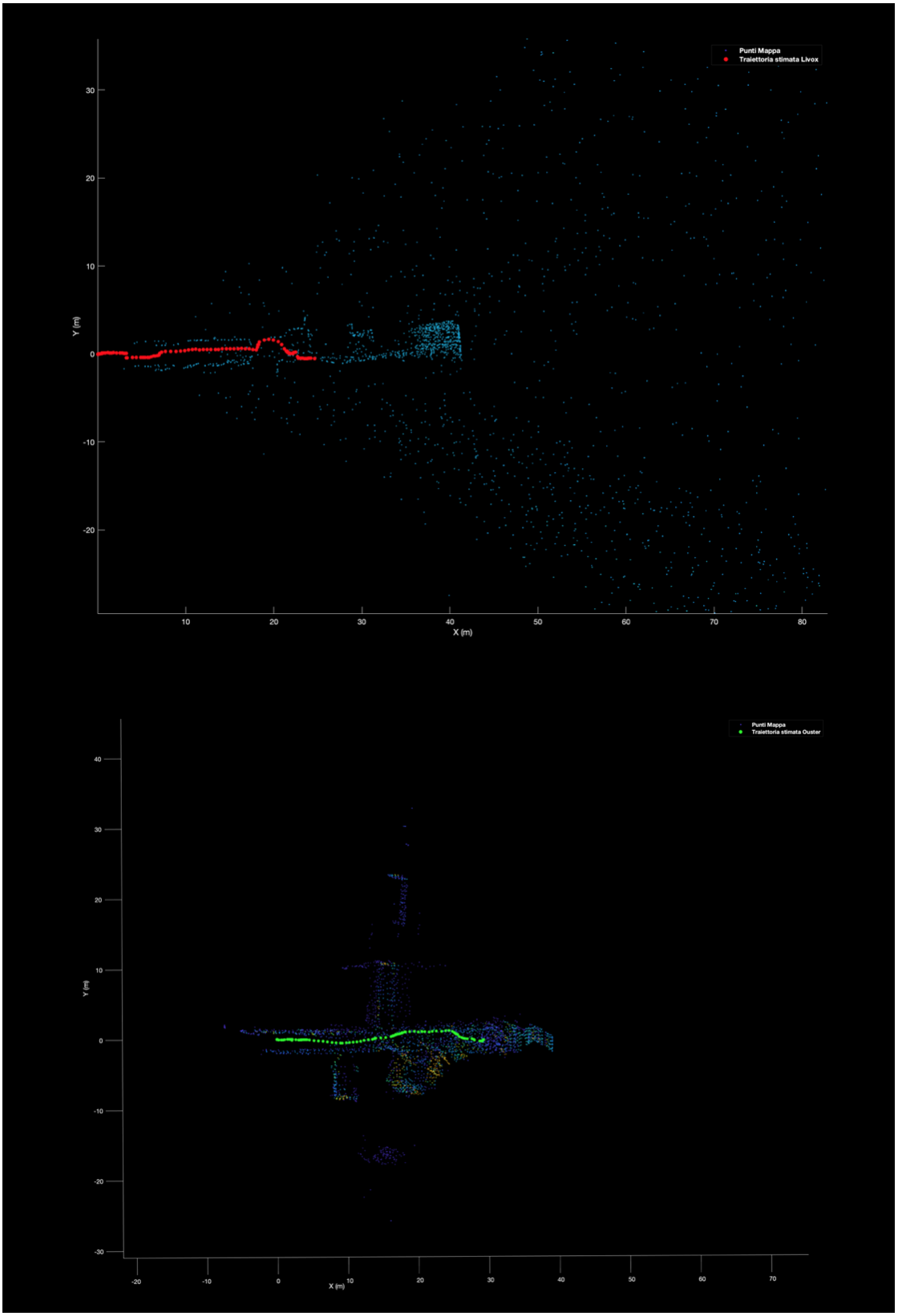

Figura 4.17: In alto: traiettoria lidar Livox Horizon, in basso traiettoria lidar Ouster

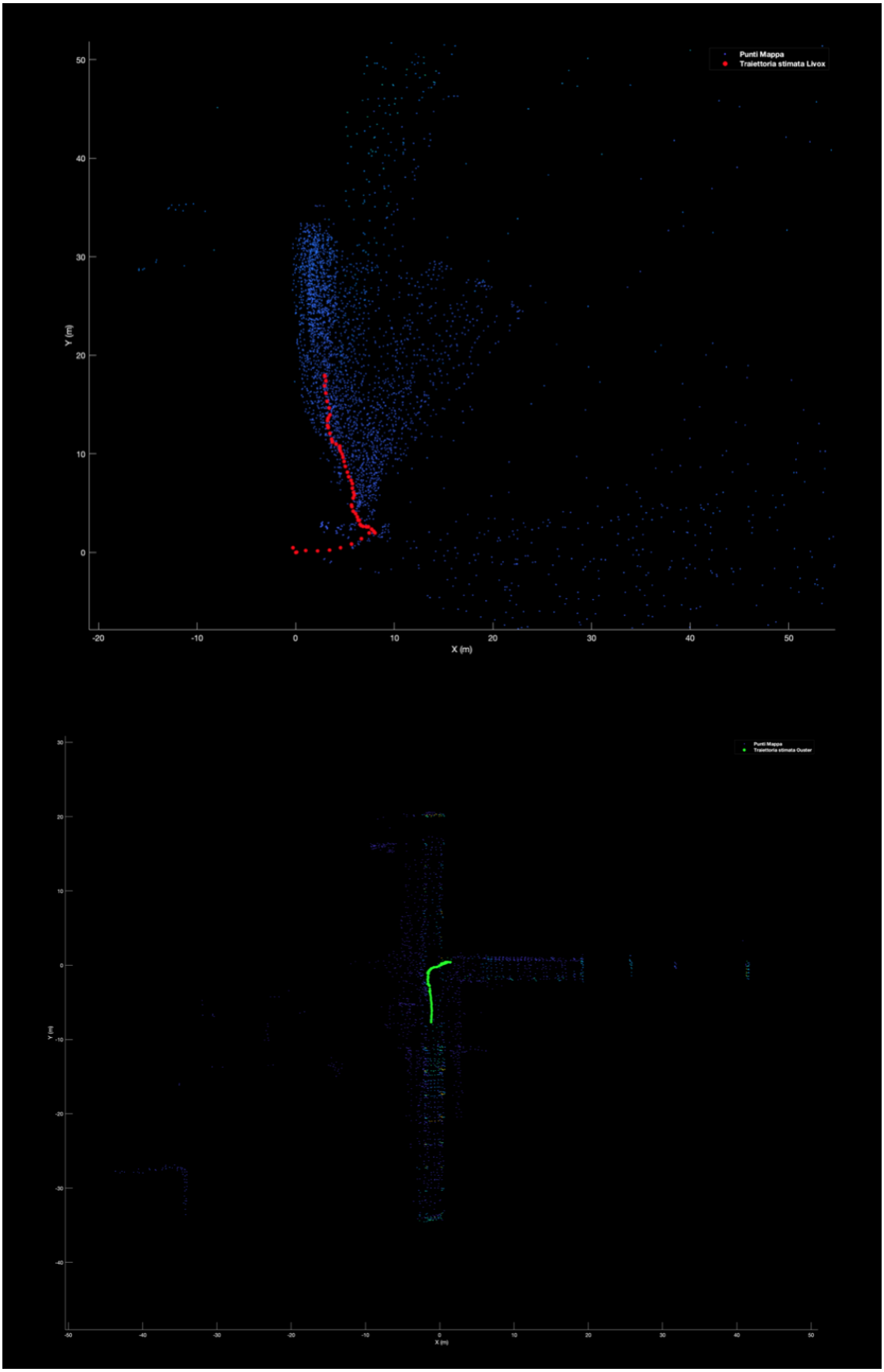

Figura 4.18: In alto: traiettoria lidar Livox Horizon, in basso traiettoria lidar Ouster

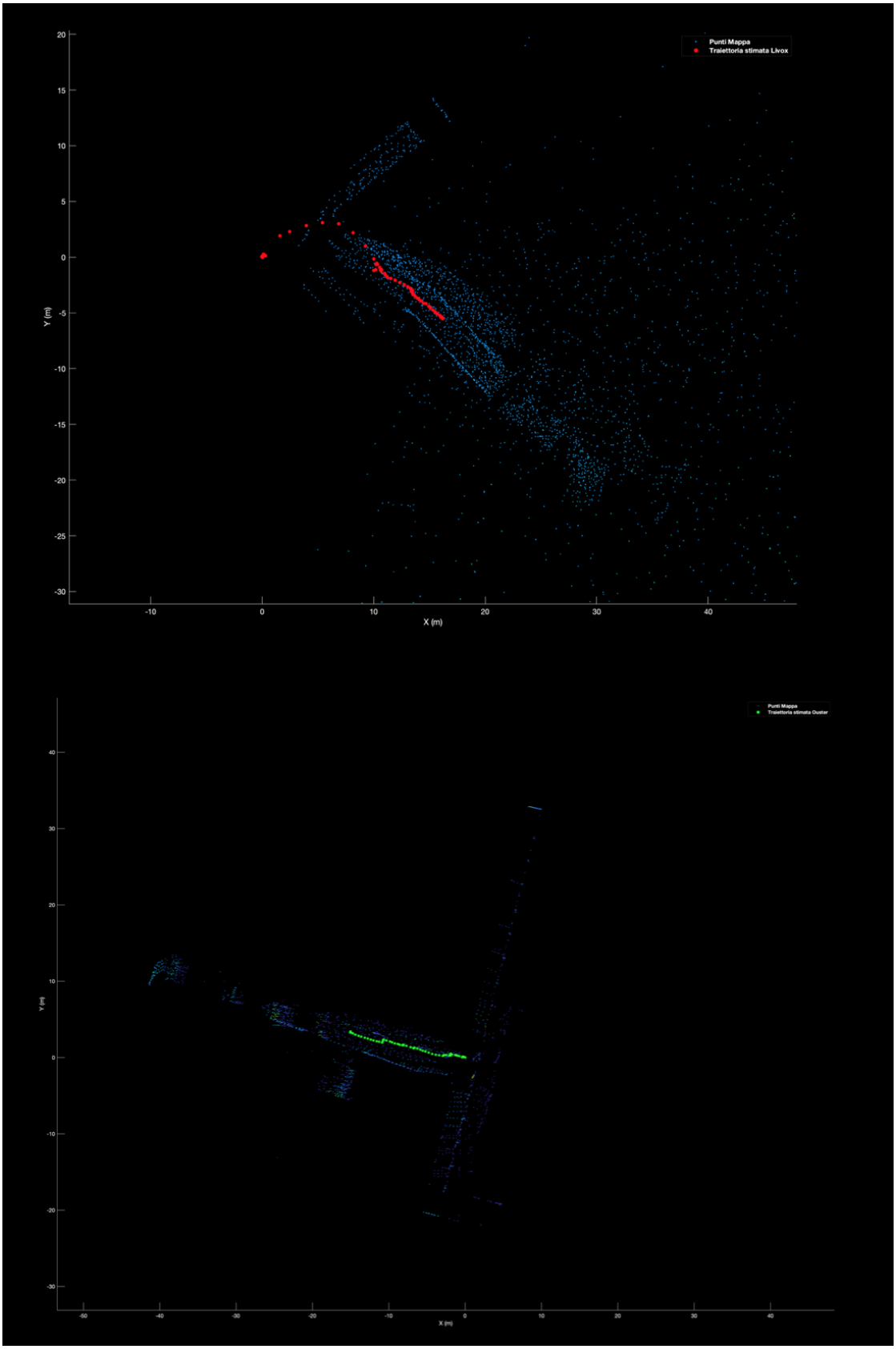

Figura 4.19: In alto: traiettoria lidar Livox Horizon, in basso traiettoria lidar Ouster

Capitolo 5

Conclusioni

L'utilizzo del metodo scelto per le analisi condotte, usando il lidar Livox Horizon e il lidar Ouster sul rover del progetto studentesco "Morpheus", ha messo in luce aspetti positivi e negativi. Le traiettorie all'aperto, con e senza il supporto del gps, sono state ricostruite in maniera abbastanza fedele alla realtà. Il confronto con la traiettoria ricostruita grazie ai dati gps ha dato modo di analizzare ulteriormente i percorsi compiuti, mettendo in risalto come, nei tratti ombrosi e vicino a edifici, la traiettoria ottenuta con il lidar sia risultata più attendibile. L'elaborazione dei dati raccolti in corridoio ha evidenziato come il metodo scelto funzioni meglio con un lidar con FoV maggiore, capace di restituire non solo una visione dell'ambiente più completa, ma anche una traiettoria nettamente migliore. Di contro, il metodo scelto, utilizzando il lidar Livox Horizon, non è stato in grado di ricostruire percorsi attendibili in tutte le situazioni. I casi più critici, soprattutto per il Lidar Livox, si sono riscontrati nel momento in cui il rover doveva percorrere curve strette, ruotava velocemente o percorreva zone con molte finestre che davano origine a riflessi. Alcune soluzioni, quindi, per migliorare i risultati ottenuti potrebbero essere l'utilizzo di dati IMU al fine di correggere il drift che si genera nell'utilizzo del lidar, l'impiego di un gps differenziale per avere un riscontro molto più preciso della traiettoria percorsa e, infine, adoperare il lidar Ouster OS1 anche per acquisire dati in ambiente esterno.

Appendice

Codici MATLAB

```
%Codice MATLAB per importare i dati da file .bag (Custom)

clear all
close all

fileName = 'esempio.bag';

%Parametri
dt = 0;

rosbag('info', fileName);

bag = rosbag(fileName);

t0 = bag.StartTime;
tF = t0+10;

bagSel = select(bag, 'Time', [t0, tF]);

msgType = 'livox_ros_driver/CustomMsg';

n = select(bagSel, 'MessageType', msgType).NumMessages;

bagSelPCL = select(bagSel, 'MessageType', msgType);

pcMsgs = readMessages(bagSelPCL,'DataFormat','struct');

for i = 1 : n

    xyz(:,1) = [pcMsgs{i}.Points.X];
```

```
    xyz(:,2) = [pcMsgs{i}.Points.Y];
    xyz(:,3) = [pcMsgs{i}.Points.Z];
    reflectivity = [[pcMsgs{i}.Points.Reflectivity]
        [pcMsgs{i}.Points.Reflectivity] [pcMsgs{i}.Points.Reflectivity]];

    reflectivity(xyz(:,1)==0, :)=[];
    xyz(xyz(:,1)==0,:)=[];


    ptCloud{i} = pointCloud(double(xyz),'Color',reflectivity);

    figure(1)
    pcshow(ptCloud{i})

    clear xyz
    clear reflectivity
end

%% Accesso ai dati

PointCloud = [ptCloud{1,:}];

PointCloud = transpose(PointCloud);
```

```
%Codice MATLAB per importare i dati da file .bag (PointCloud2)

clear all
close all

fileName = '2013-01-01-02-15-43.bag';

%Parameters
dt = 0;

rosbag('info', fileName);

bag = rosbag(fileName);

t0 = bag.StartTime;
tF = t0+434;

bagSel = select(bag, 'Time', [t0, tF]);

msgType2 = 'sensor_msgs/PointCloud2';

n2 = select(bagSel, 'MessageType',
    msgType2,'Topic','/livox/lidar').NumMessages;

bagSelPCL2 = select(bagSel, 'MessageType',
    msgType2,'Topic','/livox/lidar');

pcMsgs2 = readMessages(bagSelPCL2,'DataFormat','struct');


%% Estrae i PointCLouds

for i=1:n2
    xyz2=rosReadXYZ(pcMsgs2{i});
    ptCloud{i}=pointCloud(double(xyz2));
end

%% Accesso ai dati

PointCloud = [ptCloud{1,:}];

PointCloud = transpose(PointCloud);
```

```
% Script per leggere files da lidar OUSTER

%% Per vedere la registrazione
% Richiamo file di calibrazione
ousterReader = ousterFileReader("scansione_1.pcap","calibrazione.json");
%%

% Definisco limiti player

xlimits = [-4 4];
ylimits = [-4 4];
zlimits = [-2 3];

% Chiamo il player

player = pcplayer(xlimits,ylimits,zlimits);
xlabel(player.Axes,"X (m)");
ylabel(player.Axes,"Y (m)");
zlabel(player.Axes,"Z (m)");
ousterReader.CurrentTime = ousterReader.StartTime + seconds(0.0);
while(hasFrame(ousterReader) && player.isOpen())
    ptCloud = readFrame(ousterReader);
    view(player,ptCloud);
end

%% Estrae PointCloud

%inserire il numero di frames che interessa analizzare

Nuvole_punti=[];
for i=1:1:n; % n = numero punti
Nuvole_punti{i} = readFrame(ousterReader,i);
end

%% Accesso ai dati

PointCloud = [Nuvole_punti{1,:}];

PointCloud = transpose(PointCloud);
```

```
%% Script per analizzare i dati lidar %%
%
% Script adattato e basato su un metodo proposto
% nella sezione Computer Vision Toolbox (MathWorks)
%
clc;
%% IMPORTA I DATI

%% Visualizzazione dati

% Limiti
xlimits = [-5 45]; % metri
ylimits = [-45 45];
zlimits = [-10 20];

% PCPplayer
lidarPlayer = pcplayer(xlimits, ylimits, zlimits);

% Assi
xlabel(lidarPlayer.Axes, 'X (m)')
ylabel(lidarPlayer.Axes, 'Y (m)')
zlabel(lidarPlayer.Axes, 'Z (m)')

title(lidarPlayer.Axes, 'Dati sensore Lidar')


    for l = 1 : height(PointCloud)

        % Estrazione Point Cloud
        ptCloud = PointCloud(l);

        % Visualizzazione
        view(lidarPlayer, ptCloud);
    end

%% Costruzione mappa

frameNum = 1;

ptCloud = PointCloud(frameNum);
```

```
skipFrames = 5;

frameNum  = 100;

fixed  = PointCloud(frameNum);

moving = PointCloud(frameNum + skipFrames);

fixedProcessed = helperProcessPointCloud(fixed);

movingProcessed = helperProcessPointCloud(moving);

hFigFixed = figure;
pcshowpair(fixed, fixedProcessed)
view(2);

helperMakeFigurePublishFriendly(hFigFixed);

% Downsample Point Clouds prima della registrazione.

downsamplePercent = 0.1; %[0-1]
fixedDownsampled = pcdownsample(fixedProcessed, 'random',
    downsamplePercent);
movingDownsampled = pcdownsample(movingProcessed, 'random',
    downsamplePercent);

regGridStep = 5; %[>5]
tform = pcregisterndt(movingDownsampled, fixedDownsampled, regGridStep);

movingReg = pctransform(movingProcessed, tform);

% Visualizza allineamento

hFigAlign = figure;

subplot(121)
pcshowpair(movingProcessed, fixedProcessed)
title('Prima della registrazione')
view(2)

subplot(122)
pcshowpair(movingReg, fixedProcessed)
title('Dopo la registrazione')
```

```
view(2)

helperMakeFigurePublishFriendly(hFigAlign);

mergeGridStep = 0.5;
ptCloudAccum = pcmerge(fixedProcessed, movingReg, mergeGridStep);

hFigAccum = figure;
pcshow(ptCloudAccum)
title('Point Cloud accumulata')
view(2)

helperMakeFigurePublishFriendly(hFigAccum);

%% Traiettoria

mapBuilder = helperLidarMapBuilder('DownsamplePercent',
    downsamplePercent);

rng(0);

closeDisplay = false;
numFrames   = height(PointCloud);
tform = rigidtform3d;

for n = 1 : skipFrames : numFrames-skipFrames

    % Prende ennesimo Point Cloud
    ptCloud = PointCloud(n);

    % Stima trasformazione rigida
    initTform = tform;

    % Aggiorna mappa usando Point Cloud
    tform = updateMap(mapBuilder, ptCloud, initTform);

    updateDisplay(mapBuilder, closeDisplay);
end

helperAddLegend(mapBuilder.Axes, ...
    {'Punti mappa', 'Traiettoria Lidar'});

%% GPS
```

```
% Origine GPS coincide con Lidar

origin = [gpsSequence.Latitude(1), gpsSequence.Longitude(1),
    gpsSequence.Altitude(1)];

% Conversione dati GPS nel sistema locale East-North-Up

[xEast, yNorth, zUp] = latlon2local(gpsSequence.Latitude,
    gpsSequence.Longitude, ...
    gpsSequence.Altitude, origin);

% Rotazione
theta = median(atan2d(yNorth(1:15), xEast(1:15)));

R = [ cosd(90-theta) sind(90-theta) 0;
     -sind(90-theta) cosd(90-theta) 0;
     0              0              1];

groundTruthTrajectory = [xEast, yNorth, zUp]%* R;

% Aggiungi a mappa

hold(mapBuilder.Axes, 'on')
scatter(mapBuilder.Axes, groundTruthTrajectory(:,1),
    groundTruthTrajectory(:,2), ...
    'blue','filled');

helperAddLegend(mapBuilder.Axes, ...
    {'Punti mappa', 'Traiettoria Lidar', 'Traiettoria GPS'});

%% Rotazione Lidar
% Seleziona punto di riferimento per traiettoria lidar
[gpsSequence.Latitude(1), gpsSequence.Longitude(1),
    gpsSequence.Altitude(1)];
% Estrae vettori coordinate lidar

[xEst] = [mapBuilder.RelativePositions(:,1)];
[yNord] = [ mapBuilder.RelativePositions(:,2)];
[zsu] = [ mapBuilder.RelativePositions(:,3)];

% Rotazione
theta = median(atan2d(yNord(1:15), xEst(1:15)));
```

```
R = [ cosd(-90-theta) sind(-90-theta) 0;
    -sind(-90-theta) cosd(-90-theta) 0;
    0              0             1];

Percorso_Lidar = [xEst, yNord, zsu] * R;

% Aggiungi a mappa
hold(mapBuilder.Axes, 'on')
scatter(mapBuilder.Axes, Percorso_Lidar(:,1), Percorso_Lidar(:,2), ...
    'green','filled');

helperAddLegend(mapBuilder.Axes, ...
    {'Punti mappa', 'Traiettoria Lidar', 'Traiettoria GPS','Traiettoria
       Lidar modificata'});

%% Cambio coordinate per Google Earth Lidar
origin = [45.4103944, 11.8990916, 11.0]; %valori esempio
[lat_lidar,lon_lidar] = local2latlon(x_Lidar,y_Lidar,z_Lidar,origin);

%% Cambio coordinate per Google Earth gps
origin = [45.4103944, 11.8990916, 11.0]; %valori esempio
[lat_gps,lon_gps] = local2latlon(x_gps,y_gps,z_gps,origin);

%% salvataggio traiettoria lidar
namefile1=strcat('Lidar.kml');
kmlwriteline(namefile1,lat_lidar,lon_lidar,'Color','red','Linewidth',4)

%% salvataggio traiettoria gps
namefile1=strcat('gps.kml');
kmlwriteline(namefile1,lat_gps,lon_gps,'Color','blue','Linewidth',4)
```

Bibliografia

[1] Alessandro Caporali. *Posizionamento Satellitare e Determinazione Orbitale*. 2018.

[2] Bonn Igor Bogoslavskyi & Cyrill Stachniss. «Efficient Online Segmentation for Sparse 3D Laser Scans». In: (2016).

[3] Ouster Inc. *OS1, Mid-Range High-Resolution Imaging Lidar*. 2023.

[4] Martin Magnusson. *The Three-Dimensional Normal-Distributions Transform – an EfficienMt RaerptrienseMntatgionnufsosroRnegistration, Surface Analysis, and Loop Detection*. 2009.

[5] David G. Lowe Marius Muja. «Fast Approximate Nearest Neighbors With Automatic Algorithm Configuration». In: (2009).

[6] MathWorks. «Help Computer Vision Toolbox». In: (2022).

[7] Wolfgang Straßer Peter Biber. «The Normal Distributions Transform: A New Approach to Laser Scan Matching». In: (2003).

[8] Livox Tech. *Manuale Livox Horizon 1.0*. 2019.

[9] P. H. S. Torr e A. Zisserman. «MLESAC: A new robust estimator with application to estimating image geometry». In: (1996).

Ringraziamenti

Grazie a tutti coloro che mi hanno supportato in questi anni di università. Un ringraziamento speciale al mio relatore, il professore Marco Pertile, e al professore Sebastiano Chiodini, responsabile del progetto studentesco "Morpheus", che mi hanno dato la possibilità di fare questa esperienza, aiutandomi nell'acquisizione dei dati e rendendosi sempre disponibili. Ringrazio la mia famiglia con cui ho condiviso e superato momenti difficili, gli amici che ho incontrato lungo questo cammino e che hanno alleggerito le giornate di studio regalandomi momenti felici.