UNIVERSITY OF CALIFORNIA
RIVERSIDE

# A Point Cloud Motion Distortion Correction Method Based on Motion Estimation Using Consecutive Lidar Scans

A thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Electrical Engineering

by

Yipeng Wang

September 2023

Thesis Committee:

Dr. Konstantinos Karydis, Chairperson
Dr. Amit K. Roy-Chowdhury
Dr. Hyoseung Kim

The Thesis of Yipeng Wang is approved:

---

---

---

Committee Chairperson

University of California, Riverside

## Acknowledgments

I would like to begin by thanking my supervisor Prof. Konstantinos Karydis. Without taking his course back in Fall 2021, I wouldn't have chosen the thesis option and started my research. I am also very thankful to Hanzhe Teng, who brought me into the world of LiDAR and provided me with so much help.

I deeply appreciate my committee members, Prof. Amit K. Roy-Chowdhury and Prof. Hyoseung Kim. For my personal reasons, the defense became rushed and unscheduled. Thank you for your patience.

Thanks, also, to all the people at CRIS. Thanks to Keran Ye for showing me the dedication and enthusiasm to research. Thanks to Tomas Olvera Hale for bringing everybody closer. Thanks to everyone for making CRIS such a fun place.

Thanks to my parents for supporting me throughout my life.

# ABSTRACT OF THE THESIS

A Point Cloud Motion Distortion Correction Method Based on Motion Estimation Using Consecutive Lidar Scans

by

Yipeng Wang

Master of Science, Graduate Program in Electrical Engineering
University of California, Riverside, September 2023
Dr. Konstantinos Karydis, Chairperson

This thesis presents a novel motion distortion correction method for point clouds acquired from LiDAR sensors. The proposed method estimates a motion profile from multiple point cloud segments by fitting a second-order polynomial motion model using axis-angle representation and Swing-Twist Decomposition. The proposed algorithm can be integrated into a LiDAR odometry and mapping system, and can enhance the overall performance in the absence or failure of external inertial measurement units. Experimental results demonstrated the effectiveness of the proposed method on a mobile robot operating in structured and unstructured environments.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Many LiDARs can be modeled as a series of laser rangefinders attached to a rotation unit. During its continuous rotation, 3D points are acquired based on the readings and the orientation of the laser rangefinders. When enough points to cover a certain field of view are acquired, they are defined as a "scan" of the LiDAR. Since the acquisition of a scan is carried on during a continuous period of time, motion distortion will take effect as the LiDAR moves. Figure 1.1 shows several point clouds captured by a LiDAR with 360 degrees of field of view (FoV) in the same room environment with different types of motion distortions.

LiDAR odometry and mapping is attracting more and more attentions in recent years [2, 3, 6, 7, 15, 20, 21, 23–28], since LiDAR sensors can often provide more accurate laser-point observations and is robust to environmental variations such as illumination and weather change compared to visual-odometry [9]. Most LiDAR-odometry methods improve their accuracy by utilizing a map. A map contains historical observations, either extracted

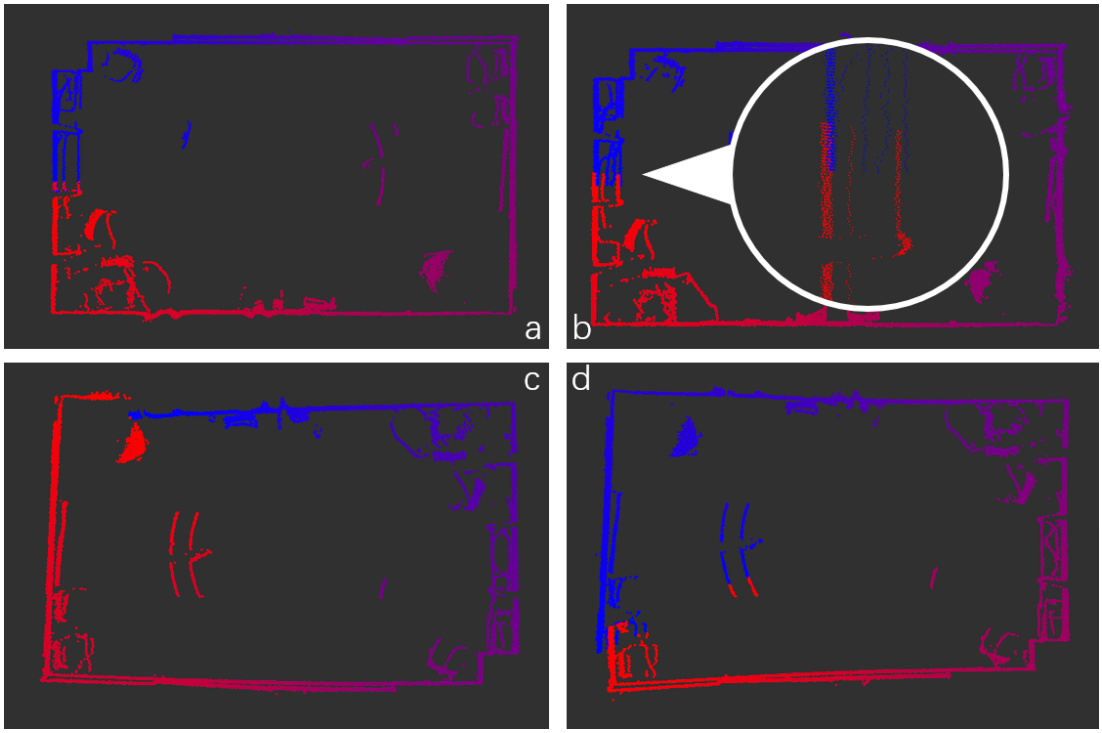


Figure 1.1: Example scans with different types of motion distortion. The color of each point indicates the changes in its timestamp, where blue denotes the oldest and red denotes the newest. (a) No motion distortion. (b) Motion distortion caused by rightward translation. (c) Motion distortion caused by counter-clockwise rotation. (d) Motion distortion caused by clockwise rotation.

features ( [10], [23]) or sub-sample of previous scans ( [22], [16, 18]). By incorporating scan-to-map registration, the accumulated error caused by scan-to-scan registration could be eliminated. However, accurate mapping requires motion distortion correction, as matching a distorted scan to a distortion-free map, or adding a distorted scan to the map, will both impact the result. 1.2 shows the map built by KISS-ICP [10] with and without its motion distortion correction module. Since the distorted points are added to the map, the odometry accuracy is also impacted.

With the help of other sensors, typically IMU, GPS, wheel encoder, etc., the distortion can be corrected by measuring the motion of the LiDAR. [12] provides a distortion-free

visual-LiDAR dataset for autonomous driving scenarios. [20] and [16] both remove rotation distortion of the scan using IMU in their preprocessing step. However, requiring a certain combination of sensors will restrict the application scenarios. Integrating different types of sensors also requires high-quality spatial and temporal calibration. As a result, there still remains a big interest in LiDAR-only odometry approach ( [10], [17], [23] and [22]).

For LiDAR-only odometry methods, various methods have tried to compensate for it by assuming that the LiDAR motion follows certain motion models. [22] compensate for such distortion with a constant velocity assumption based on the trajectory output of previous scans. This method is generally applicable and works well as long as the odometry is accurate and there's no excessive change in velocity. [13] proposed a method that iteratively estimates the velocity of the scan and deskews the scan based on estimated velocity using ICP. [23] proposed a two-stage distortion compensation by simplifying [13] into only two steps. These methods could utilize the output of LiDAR-odometry and thus do not bring any computation overhead.

On the other hand, some research works try to take into account the motion during the scan by estimating a continuous-time trajectory. [5] and [1] both define a continuous trajectory with multiple control poses using linear interpolation or B-splines. However, both methods are not in real-time. [10] proposed a real-time elastic registration method that optimizes two transformations and compensates the point cloud using their interpolation. While also assuming a constant velocity motion, this approach decouples the transformation between scans and the motion during the scan, thus achieving increased

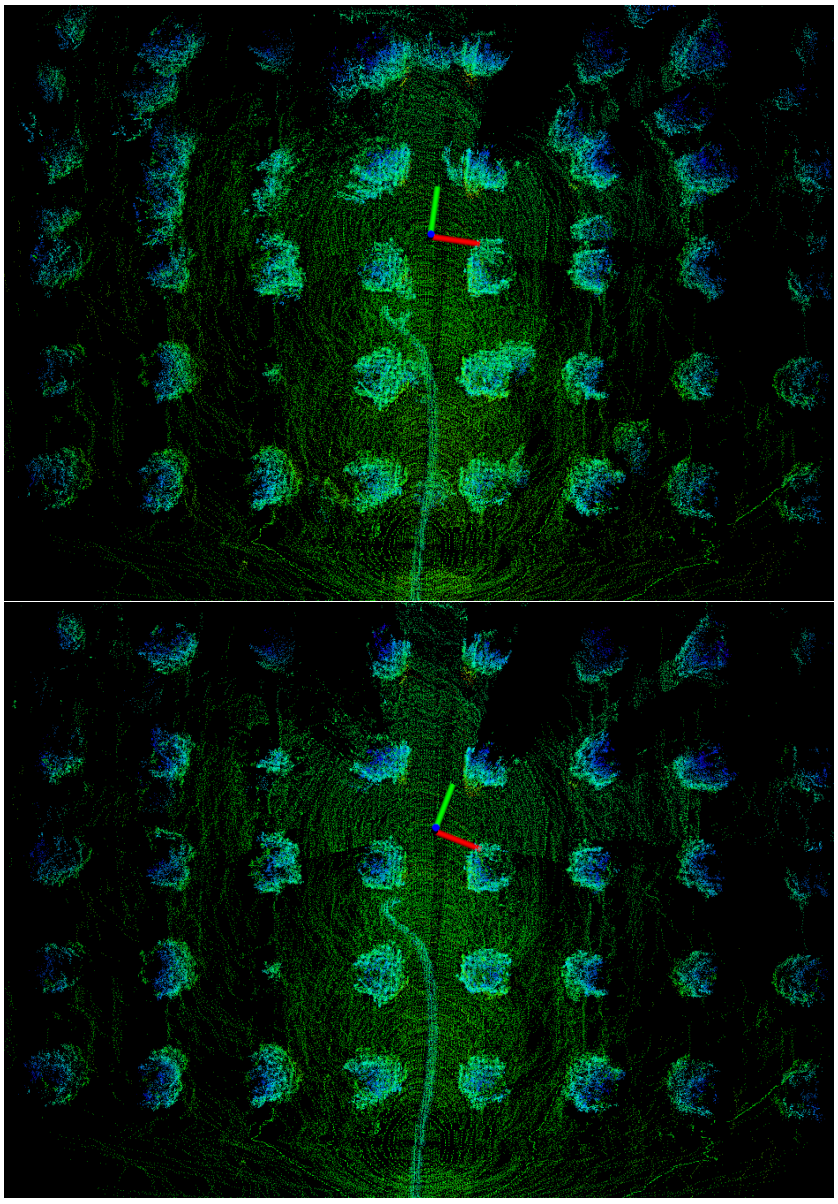

Figure 1.2: The map built by KISS-ICP when the motion distortion correction is disabled (top) v.s. enabled (bottom). The top map suffers severe ghosting and gives inaccurate pose estimation.

robustness to high-frequency motions.

Most existing LiDAR-only motion distortion correction methods have three common restrictions:

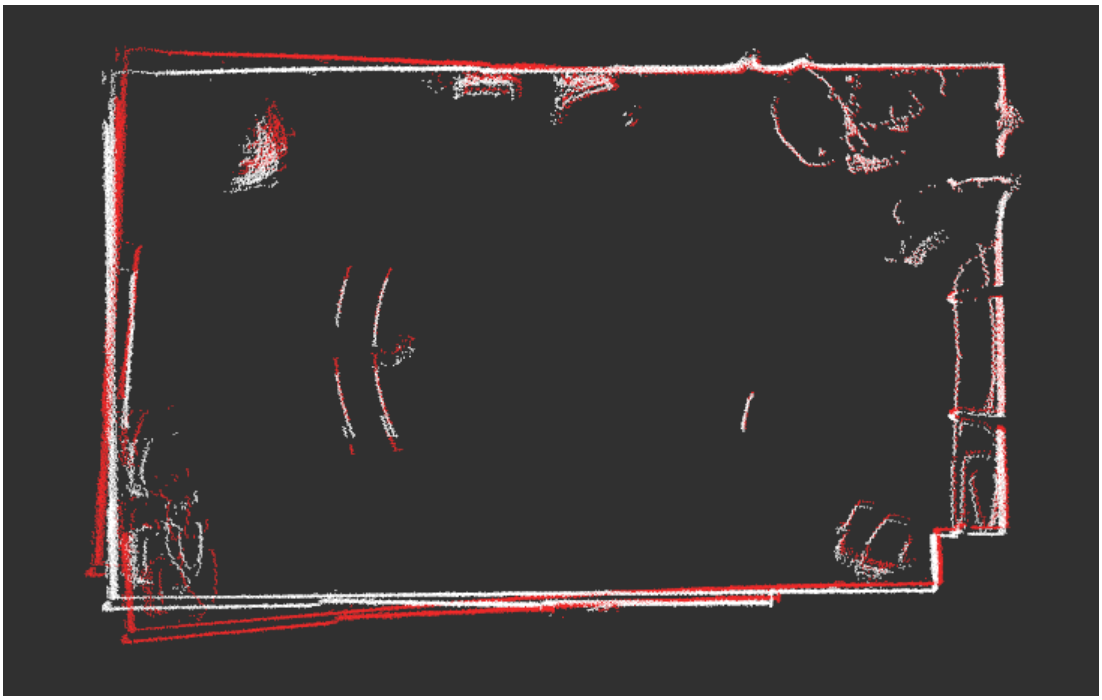

Figure 1.3: A comparison with the point cloud with and without motion distortion. The red points were the original data captured by LiDAR, while the white points are the corrected points using our proposed method.

- They all assume a constant velocity motion, i.e. the LiDAR moves with the same translational and rotational velocity during the scan. This assumption may not hold in real-world robotics applications(e.g. a robot takes an aggressive move to avoid obstacles).

- The constant velocity is acquired with the help of LiDAR-odometry. This may limit the application of certain methods to other systems.

- There is no indication of the validity of the motion distortion correction. The result will always be treated as a distortion-free scan.

In this work, we propose a new method for point cloud motion distortion correction by estimating a constant acceleration motion model during the scan. To the best of our knowledge, the proposed method is the first LiDAR-only motion distortion correction method that can estimate acceleration models from consecutive scans. Unlike existing methods which were coupled to certain odometry algorithms or rely on trajectory estimation, it does not require any input from the odometry and can be directly integrated into any dense LiDAR-odometry systems. The proposed method is intended to correct the LiDAR scan before the scan-to-map registration in our LiDAR-odometry framework, which is to be discussed in Chapter 3. The accuracy of the proposed method in rotation-only scenarios is compared against constant velocity mode, a continuous-time elastic model [10], and the ground truth distortion model provided by an IMU. Figure 1.3 provides a qualitative example of a top-down view of the raw and motion-corrected point cloud.

# Chapter 2

# Related Work

## 2.1 Point Cloud Registration

### 2.1.1 ICP

Originally proposed by [4], Iterative Closet Points (ICP) is an iterative algorithm that can be summarized in two steps:

- Given transform $\mathrm{T} \in \mathrm{SE}(3)$, for each point $a_i$ in point cloud $A$, find corresponding point $b_i$ in point cloud $B$ that is closest to $\mathrm{T} \cdot a_i$.
- Given a set of correspondence $\{a_i, b_i\}$ in point cloud $A$ and $B$, find transform T that minimizes $\sum_i ||b_i - \mathrm{T} \cdot a_i||^2$, i.e. distance between correspondences.

Since in real-world scenarios the full-overlap assumption does not usually hold, a maximum matching threshold $d_{max}$ is usually added to remove outliers. Only correspondences with a distance smaller than $d_{max}$ will be taken into account.

### 2.1.2 Point to Plane ICP

Point-to-Plane ICP ( [8]) improves the performance of ICP by taking surface normal into account. Instead of minimizing $\sum_i ||b_i - \mathrm{T} \cdot a_i||^2$, point-to-plane ICP minimizes

$$\sum_i ||\boldsymbol{n}_i \cdot (b_i - \mathrm{T} \cdot a_i)||^2$$

where $\boldsymbol{n}_i$ is the surface normal of $b_i$. Since the error is projected onto the surface normal, point-to-plane ICP could overcome some ambiguous scenarios, where ICP would usually fail. According to [14], point-to-plane ICP is more resistant to Gaussian noise in terms of accuracy. However, the report also shows that point-to-plane ICP is less resistant to Gaussian noise in terms of validity since the estimation of surface normal will be affected by noise.

### 2.1.3 Generalized-ICP

Segal et al. proposed Generalized-ICP [19] by introducing a probabilistic model into the minimization step. Assuming $a_i$ and $b_i$ are generated according to $a_i \sim \mathcal{N}(\hat{a}_i, C_i^A)$, $b_i \sim \mathcal{N}(\hat{b}_i, C_i^B)$, where $C_i^A$ and $C_i^B$ are covariance matrices associated with the measured points, the minimization term can be written as

$$\sum_i {d_i^{(\mathrm{T})}}^T (C_i^B + \mathrm{T} C_i^A \mathrm{T}^T)^{-1} d_i^{(\mathrm{T})}$$

where $d_i^{(\mathrm{T})} = b_i - \mathrm{T} \cdot a_i$.

Since it's difficult to acquire accurate covariance estimation based on a single scan, GICP is usually implemented as a "plane-to-plane" ICP by assuming every scan point is locally a plane. For each point $p_i$, surface normal $\boldsymbol{n}_i$ is first estimated, the covariance $C_i^P$

is then composed by

$$C_i^P = \begin{pmatrix} \boldsymbol{n} & \boldsymbol{v_1} & \boldsymbol{v_2} \end{pmatrix} \cdot \begin{pmatrix} \epsilon & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{n} & \boldsymbol{v_1} & \boldsymbol{v_2} \end{pmatrix}^T$$

where $\boldsymbol{n}$ is the surface normal of the point, $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ are two orthonormal vectors that are perpendicular to $\boldsymbol{n}$. $\epsilon$ is a small constant representing the covariance along the surface normal. GICP has been proven to be more accurate and robust in scan-to-scan registration problems [19].

## 2.2 Motion Distortion Correction

### 2.2.1 IMU Motion Distortion Correction

Given IMU orientation measurement $\mathrm{R}_1$ and point $p_1$ acquired at time $\tau_1$, IMU orientation measurement $\mathrm{R}_2$ and point $p_2$ acquired at time $\tau_2$, we can remove the rotation distortion by transforming $p_2$ into $p_1$ frame:

$$p_2' = \mathrm{R}_1^T \mathrm{R}_2 \cdot p_2$$

If the frequency of the IMU is high enough, we can assume a constant velocity between two IMU measurements, and all points acquired between $\tau_1$ and $\tau_2$ can be transformed to $p_1$ frame by

$$p_i' = {\mathrm{R}_1}^T \cdot \mathrm{Slerp}(\mathrm{R}_1, \mathrm{R}_2, \tau') \cdot p_i$$

where $\tau' \in [0, 1]$ is the relative time of point $p_i$ between $\tau_1$ and $\tau_2$.

### 2.2.2 Constant Velocity Motion Distortion Correction

Denoting the pose at $k^{th}$ scan is $\mathrm{T}_k$, the relative transform of $(k-2)^{th}$ scan and $(k-1^{)th}$ can be obtained from

$$\xi_{k-2}^{k-1} = \log(\mathrm{T}_{k-2}^{-1}\mathrm{T}_{k-1})$$

where $\xi_{k-2}^{k-1} \in \mathfrak{se}(3)$. Assuming a constant velocity motion during the previous scan and the current scan, i.e. $\xi_{k-1}^{k} = \xi_{k-2}^{k-1}$, the point $p_i$ acquired at relative time $\tau_i$ can be transformed to the beginning of the scan by

$$p_i' = \exp(\tau_i \cdot \xi_{k-2}^{k-1}) \cdot p_i$$

Except for the historical trajectory, the constant velocity can also be extracted from other sources. [13] estimates the velocity through iterative ICP. [23] obtains the final velocity from the odometry output of the current scan. While the undistortion cannot contribute to the scan-to-map registration, the points added to the map will have better accuracy.

### 2.2.3 Elastic Registration

[10] proposed an elastic registration method by optimizing both the start pose $\mathrm{T}_s$ of the scan and the end pose $\mathrm{T}_e$ of the scan with respect to key points in the scan against the local map while assuming the start pose of current scan is different from the end pose of the previous scan and a constant velocity motion during the start and the end pose. While this method was not originally proposed for motion distortion correction, given the estimated $\mathrm{T}_s$ and $\mathrm{T}_e$, a point $p_i$ acquired at relative time $\tau_i$ can be transformed to the beginning of

the scan by

$$p_i' = \log(\mathrm{T}_s^{-1}\mathrm{T}_e)$$

By introducing a discontinuity between scans, i.e. the begin pose $\mathrm{T}_s^k$ of the current scan is not necessarily equal to the end pose $\mathrm{T}_e^{k-1}$ of the previous scan, this method uses two distinct components to describe the $k-th$ frame: a rigid relative transform $\mathrm{T}_r = \left(\mathrm{T}_e^{k-1}\right)^{-1}\mathrm{T}_s$ and a constant velocity transform $\mathrm{T}_c = \mathrm{T}_s^{-1}\mathrm{T}_e$. The elastic registration method will find the "best" constant velocity that minimizes the distortion error, while the constant velocity motion distortion correction method can be seen as a special case of this method where $\mathrm{T}_r = \mathrm{T}_c$.

# Chapter 3

# Algorithm Description

Given the current scan $B$ and the previous scan $A$, the proposed method deskews $B$ by estimating a motion model during the acquisition of $B$ using $B$ and $A$. If there is no failure during the estimation, $B$ is compensated using the model. For convenience, the motion model is divided into rotation and translation.

## 3.1 Rotation Distortion Correction

The rotation of the motion model can be decomposed into 2 parts:

- A constant angular acceleration rotation along a fixed axis, describing the difference in distortion caused by rotation between $B$ and $A$.
- A constant angular velocity rotation along the same axis, describing the distortion of $A$. Here the previous scan $A$ is assumed to obey a constant velocity rotation model.

### 3.1.1 Constant Angular Acceleration Rotation

A scan-to-scan registration is first performed using all points in $B$ and $A$:

$$\mathrm{R}, \mathrm{t} = \textbf{Registration}(B, A) \tag{3.1}$$

where GICP is used for its higher accuracy and robustness in scan-to-scan registration. R and t is the estimated rotation and translation, respectively. The fixed rotation axis $\xi$ is obtained by

$$\xi = \frac{\log(\mathrm{R})}{|\log(\mathrm{R})|} \tag{3.2}$$

Given $\tau_i \in [0, 1]$ is the relative timestamp of $b_i \in B$, the current scan $B$ is then segmented into $k$ pieces based on timestamp of the points, denoted as $B^1, ..., B^k$, where for $b_m \in B^n$, $\tau_m \in [\frac{n-1}{k}, \frac{n}{k}]$. The transformation between $B^j$ and $A$ can be obtained by solving a registration problem:

$$\mathrm{R}^j, \mathrm{t}^j = \textbf{Registration}(B^j, A) \tag{3.3}$$

For $\mathrm{R}^j$, a swing-twist decomposition [11] is performed to obtain the rotation component along the fixed axis $\xi$:

$$\mathrm{R}^j_{twist}, \mathrm{R}^j_{swing} = \textbf{SwingTwistDecomposition}(\mathrm{R}^j, \xi) \tag{3.4}$$

where $\mathrm{R}_{twist}$ is a rotation around axis $\xi$ and $\mathrm{R}_{twist}$ is a rotation around an axis that is perpendicular to $\xi$. $\mathrm{R}^j = \mathrm{R}^j_{swing} \cdot \mathrm{R}^j_{twist}$. The angle of $\mathrm{R}^j_{twist}$ is given by

$$\theta^j = |\log(\mathrm{R}^j_{twist})| \tag{3.5}$$

By assuming $\mathrm{R}^j$ describes the rotation of the middle point in $B^j$ with respect to relative timestamp, we have a series of rotation angle $\theta^j$ around the axis $\xi$ and their

corresponding relative timestamp $\tau_j = \frac{j-0.5}{k}$, a constant angular acceleration model can be obtained by solving the least square problem

$$\boldsymbol{\theta} = \frac{1}{2}\alpha \cdot \boldsymbol{\tau}^{\circ 2} + \omega \cdot \boldsymbol{\tau} + \theta_0 \tag{3.6}$$

where $\boldsymbol{\theta} = (\theta^1, ..., \theta^k)^T$ and $\boldsymbol{\tau} = (\tau_1, ..., \tau_k)^T$. $\alpha$, $\omega$ and $\theta_0$ denote the angular acceleration, initial angular velocity, and initial rotation around the given axis $\xi$, respectively.

The current scan $b_i \in B$ can then be deskewed using the motion model

$$b_i' = \exp\left(\xi \cdot (\alpha\tau_i^2 + \omega\tau_i)\right) \cdot b_i \tag{3.7}$$

Note that $\theta_0$ is ignored above since applying a constant rotation to the whole point cloud will not affect its distortion.

This step compensates for the distortion caused by the difference in angular velocity compared to the previous scan $A$. Since $A$ might also be distorted, this step can be regarded as "distorting the current scan so that it has similar distortion to the previous scan".

### 3.1.2 Constant Angular Velocity Rotation

Although $\theta_0$ is ignored in the previous step, it represents the rotation from $A$ to $B$ at time 0. Since the acquisition of the scans is continuous, it can also be interpreted as the rotation from the first point to the last point of $A$. Assuming there is a constant angular velocity motion during $A$, $A$ can be decomposed using

$$a_j' = \exp(\xi \cdot \theta_0\tau_j) \cdot a_j \tag{3.8}$$

where $\tau_j \in [0, 1]$ is the relative timestamp of $a_j$. Since $B$ is distorted to have a similar distortion to $A$ by (3.7), (3.8) can also be applied to $B$.

Combining (3.7) and (3.8), the full rotation motion correction can be written as

$$b'_i = \exp\left(\xi \cdot (\alpha\tau_i^2 + \omega\tau_i + \theta_0\tau_i)\right) \cdot b_i \tag{3.9}$$

Denoting the rotation in (3.7) $\mathrm{R}_a = \exp\left(\xi \cdot (\alpha\tau_i^2 + \omega\tau_i)\right)$, the rotation in (3.8) $\mathrm{R}_c = \exp(\xi \cdot \theta_0\tau_j)$, the rotation part of the constant velocity motion correction model can also be seen as a special case of the proposed model, where $\mathrm{R} = I$.

Compared to the elastic registration approach, $\mathrm{R}_a$ models the 2nd order motion, while the elastic registration method minimizes the influence of high-frequency motion by introducing $\mathrm{R}_a$.

### 3.1.3 Failure Detection

Since the proposed method assumes a constant angular acceleration model along a fixed axis, it fails and outputs erroneously distorted points. On the other hand, for LiDAR odometry, it's usually acceptable to skip several scans while still maintaining relatively high accuracy thanks to its dense measurement. Based on such fact, the proposed method contains a failure detection module that skips the deskew for failure scans and notifies the odometry system of potential bad input.

In equation (3.4), $\mathrm{R}_{swing}$ is not used by the motion correction pipeline. The rotation angle of $\mathrm{R}_{swing}$ is

$$\theta_{swing} = |\log(\mathrm{R}^j_{swing})| \tag{3.10}$$

If the constant angular acceleration assumption holds, $\theta_{swing}$ should be subtle. A large $\theta_{swing}$ implies either a significant registration error in GICP, or a motion that cannot be described by the constant angular acceleration model.

A threshold $\theta^*_{swing}$ is used to detect such a situation. For segment $B^j$, if $\theta^j_{swing} > \theta^*_{swing}$, this segment will be discarded. Furthermore, among all $k$ segments, if less than $k_{min}$ segments remain, it is considered a failure.

Mean squared error is also used to measure the error of (3.6). If the MSE exceeds $\varepsilon_{MSE}$, the model is considered unable to describe the motion and treated as a failure.

### 3.1.4 Doppler Effect Compensation

Since the range sensors in a LiDAR are spinning around an axis, motion rotating in the same direction will have less density, while motion rotating in the opposite direction will have higher density. To compensate for such an effect, the model obtained from (3.6) is scaled by $\frac{\pi + \theta_c}{\pi}$, while $\theta_c$ is the angle of the component of $\exp(\xi \cdot \theta_0)$ around the spinning axis of LiDAR.

## 3.2 Translation Distortion Correction

Similar to rotation distortion correction, the translation of the motion model can also be decomposed into a constant acceleration motion and a constant velocity motion along the same direction.

### 3.2.1 Constant Acceleration Motion

Given t from (3.1), the fixed translation direction $\boldsymbol{x}$ can be obtained by

$$\boldsymbol{x} = \frac{\mathrm{t}}{|\mathrm{t}|} \tag{3.11}$$

For the translation of each segment $\mathrm{t}^{\mathrm{j}}$ from (3.3), similar to (3.4) and (3.5), we can decompose $t^j$ into $\mathrm{t}^j_{\boldsymbol{x}}$ and $\mathrm{t}^j_r$

$$\mathrm{t}^j_{\boldsymbol{x}} = (\boldsymbol{x} \cdot \mathrm{t}) \tag{3.12}$$

$$\mathrm{t}^j_r = |\mathrm{t} - \mathrm{t}^j_{\boldsymbol{x}}| \tag{3.13}$$

where $\mathrm{t}^j_{\boldsymbol{x}}$ is the translation along $\boldsymbol{x}$ and $\mathrm{t}^j_r$ is the translation along a direction which is perpendicular to $\boldsymbol{x}$. Given relative timestamp $\tau_j = \frac{j-0.5}{k}$, by assuming $\mathrm{t}_j$ describes the translation of the middle point in $B^j$ with respect to relative timestamp, a constant acceleration model can be obtained by solving a least square model similar to (3.6):

$$\boldsymbol{d} = \frac{1}{2} a \cdot \boldsymbol{\tau}^{\circ 2} + v \cdot \boldsymbol{\tau} + \mathrm{t}_0 \tag{3.14}$$

where $\boldsymbol{d}(|\mathrm{t}^1_{\boldsymbol{x}}|, ..., |\mathrm{t}^k_{\boldsymbol{x}}|)$ and $\boldsymbol{\tau} = (\tau_1, ..., \tau_k)^T$. $a$, $v$ and $\mathrm{t}_0$ denotes linear acceleration, initial velocity and initial position along $\boldsymbol{x}$, respectively.

The current scan $b_i \in B$ can then be deskewed using the constant acceleration motion model

$$b'_i = \frac{1}{2} a \cdot \tau_i^2 \cdot \boldsymbol{x} + v \cdot \tau_i \cdot \boldsymbol{x} + b_i \tag{3.15}$$

$\mathrm{t}_0$ is also ignored above since applying a constant translation to the whole point cloud will not affect its distortion.

Similar to (3.7), this step will give $B$ a similar distortion to $A$ with respect to translation.

### 3.2.2 Constant velocity motion

Similar to (3.8), the translation from the first point to the last point of $A$ is estimated as $\mathrm{t}_0$ in (3.14). Assuming a constant velocity motion during the acquisition of $A$, on stack of (3.15), $B$ can be compensated using

$$b_i' = b_i + \tau_i \cdot \boldsymbol{x} \cdot \mathrm{t}_0 \tag{3.16}$$

Combining (3.15) and (3.16), the full translation motion correction can be written as

$$b_i' = \frac{1}{2} a \cdot \tau_i^2 \cdot \boldsymbol{x} + v \cdot \tau_i \cdot \boldsymbol{x} + \mathrm{t}_0 \cdot \boldsymbol{x} \cdot \tau_i + b_i \tag{3.17}$$

According to (3.9) and (3.17), the full motion distortion correction of point $b_i \in B$ can be obtained by

$$b_i' = \frac{1}{2} a \cdot \tau_i^2 \cdot \boldsymbol{x} + v \cdot \tau_i \cdot \boldsymbol{x} + \mathrm{t}_0 \cdot \boldsymbol{x} \cdot \tau_i + \exp\left(\xi \cdot (\alpha \tau_i^2 + \omega \tau_i + \theta_0 \tau_i)\right) \cdot b_i \tag{3.18}$$

### 3.2.3 Failure Detection

Similar to (3.10), $|\mathrm{t}_r^j|$ is used for failure detection. A threshold $\mathrm{t}_r^*$ detects whether the constant acceleration assumption holds, i.e. $|\mathrm{t}_r^j|$ is subtle. The segment will be discarded if $|\mathrm{t}_r^j| > \mathrm{t}_r^*$ and the motion distortion correction will be considered a failure if less than $kmin$ segments remain.

Mean-squared error of (3.14) is also used for failure detection, similar to the rotation part.

# Chapter 4

# Evaluation

## 4.1 Experiment Setup

### 4.1.1 Data Collection

Real-world data based on unmanned vehicle equipment with a Velodyne VLP-16 LiDAR and a MicroStrain 3DM-GX5-AHRS IMU were collected to test the performance of the proposed method against other motion distortion correction methods. The ground truth was obtained by deskew the scam using the IMU.

Since the distortion caused by translation is relatively small in the experiment environment, and to have a fair comparison against ground truth, only the rotation distortion correction is tested. The experiment dataset is thus mostly a rotation-only scenario.

The test dataset contains

- Sequence one in structured environments with mostly smooth but occasionally sharp motion

- Sequence two in agriculture environments with mostly smooth but occasionally sharp motion
- Sequence three in agriculture environments with commonly sharp and occasionally extremely sharp motion

The reason why agricultural environments are chosen is that the rugged terrain provides high-frequency motion, which is a challenge for LiDAR-only motion distortion correction methods.

### 4.1.2 System Overview

The evaluation of the proposed method and the constant velocity motion correction are implemented on our LiDAR-odometry framework, which can be described as following steps:

1. Preprocessing, including a self-filter and down-sampling
2. A scan-to-scan registration using GICP
3. A scan-to-map registration using the result of scan-to-scan registration as initial estimation
4. Map update if the condition is met

The motion distortion correction is applied between step 2) and step 3). The distortion correction is not applied before step 2) because the consecutive scans are naturally similar. The registration of two similarly distorted scans is usually better than the result of one distorted and one distortion-free scan. Although estimation is executed on a down-sampled

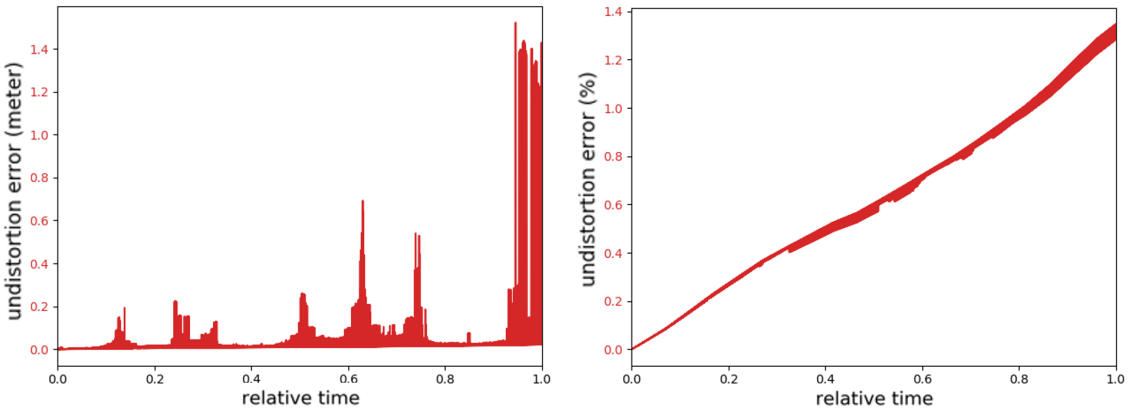


Figure 4.1: Original (left) and normalized (right) distortion error of all points in a scan to its relative timestamp.

scan, the motion model is recorded to perform distortion correction on the full scan for comparison.

The evaluation of [10] is based on their open-source implementation. The beginning pose and the ending pose of every scan are recorded to perform the distortion correction.

The distortion error of a point $p$ is the distance between the motion distortion corrected point $p'$ and the ground truth $p_g$, normalized by the distance of the point to the origin:

$$E_p = \frac{|p' - p_g|}{|p_g|} \tag{4.1}$$

where the ground truth is the point corrected using the IMU measurements. The reason why the normalized error was normalized by the distance of the point is shown in Fig.4.1. The spikes in the left figure are caused by the distance of the surrounding environment, which is a noise for the rotation-only scenario. The right figure shows a smoother plot.

## 4.2 Results and discussions

### 4.2.1 Distortion Error on different sequences

Table 4.1 shows the mean distortion error of the proposed method, constant velocity model and Elastic Registration [10] across all scans in the sequence. It can be observed that in the structured environment with smooth motion, where the constant velocity assumption holds in most cases, the constant velocity distortion correction is overall the most accurate approach. However, in agricultural environments, the constant velocity method is outperformed by our proposed method by a large margin. The reason the elastic registration method performs worse on average could be ascribed to its overfitting to the key points since the start and end pose are estimated using a local map and key points.

Table 4.1: Summary of Mean Distortion Error in Each Sequence.

| Sequence | Proposed Method | Constant Velocity | Elastic Registration |
|---|---|---|---|
| #1 | 0.191% | 0.177% | 0.642% |
| #2 | 0.28% | 0.335% | 0.33% |
| #3 | 0.266% | 0.421% | 0.449% |

A more detailed view is provided by Fig.4.2, Fig.4.3 and Fig.4.4. In Fig.4.2, it's clear that the constant velocity motion distortion method has the lowest overall error. However, it is less resistant to sudden changes in angular velocity, as it assumes no such change in the first place. The elastic registration method and our proposed method are

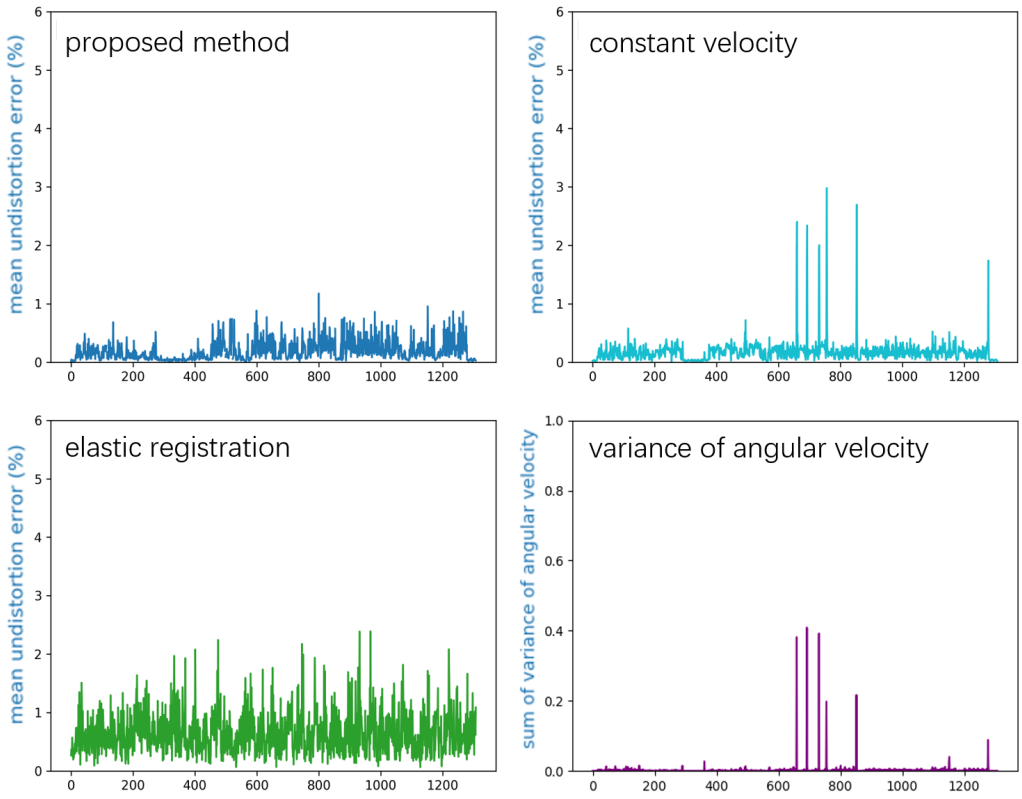


Figure 4.2: The mean distortion error along the scan index of sequence #1. The bottom-right figure shows the sum of the variance of angular velocity in each scan.

both resistant to such change.

When a similar motion profile is executed in agricultural environments, as shown in Fig.4.3, the constant velocity distortion method loses its leadership in mean error due to high-frequency motions. Both our proposed method and the elastic registration method still show good resistance to small angular velocity spikes.

When the robot moves so aggressively that it's too difficult to model the motion during the scan, Fig.4.4 shows that all three methods are not able to compensate for the distortion. However, thanks to the failure detection in our proposed method, the scans with

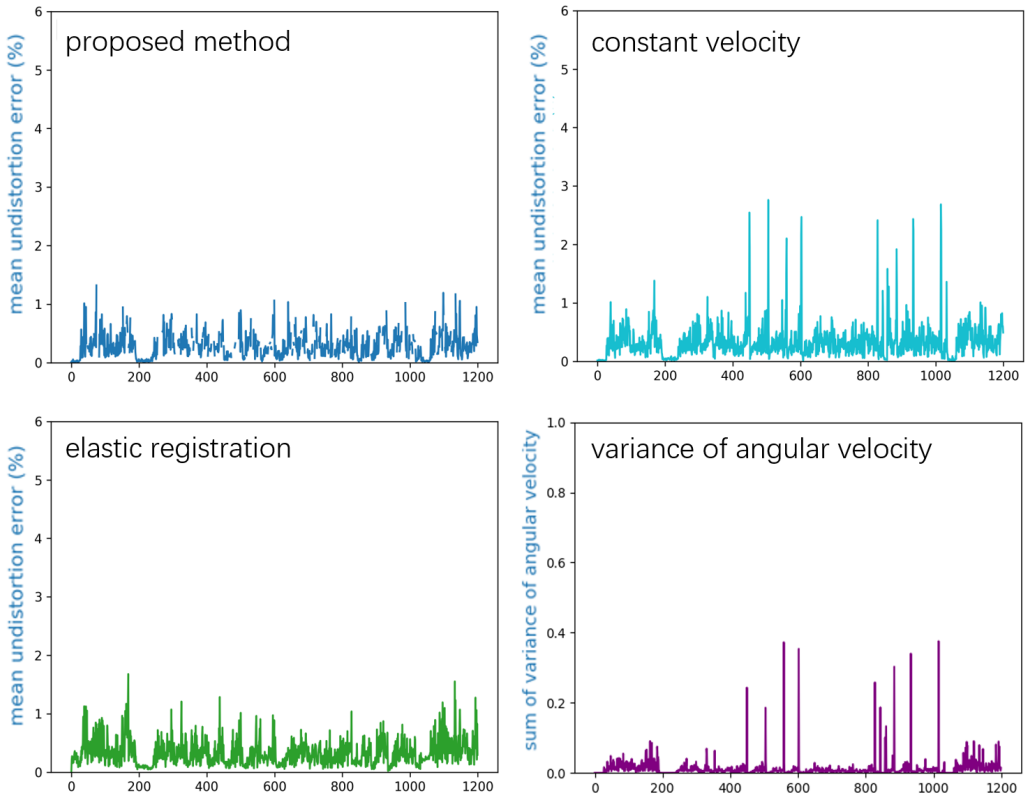


Figure 4.3: The mean distortion error along the scan index of sequence #2. The bottom-right figure shows the sum of the variance of angular velocity in each scan.

the largest distortion error in other methods are skipped by our proposed method, giving the odometry system a chance to avoid potential data pollution.

### 4.2.2 Motion Model Restoration

Fig.4.5 shows the restored motion model during a scan in sequence #2. Although both elastic registration and constant motion model correction methods estimate a constant angular velocity, the former could correctly estimate the beginning and ending pose, while the latter depends on the pose estimation and cannot correctly reflect the change of velocity

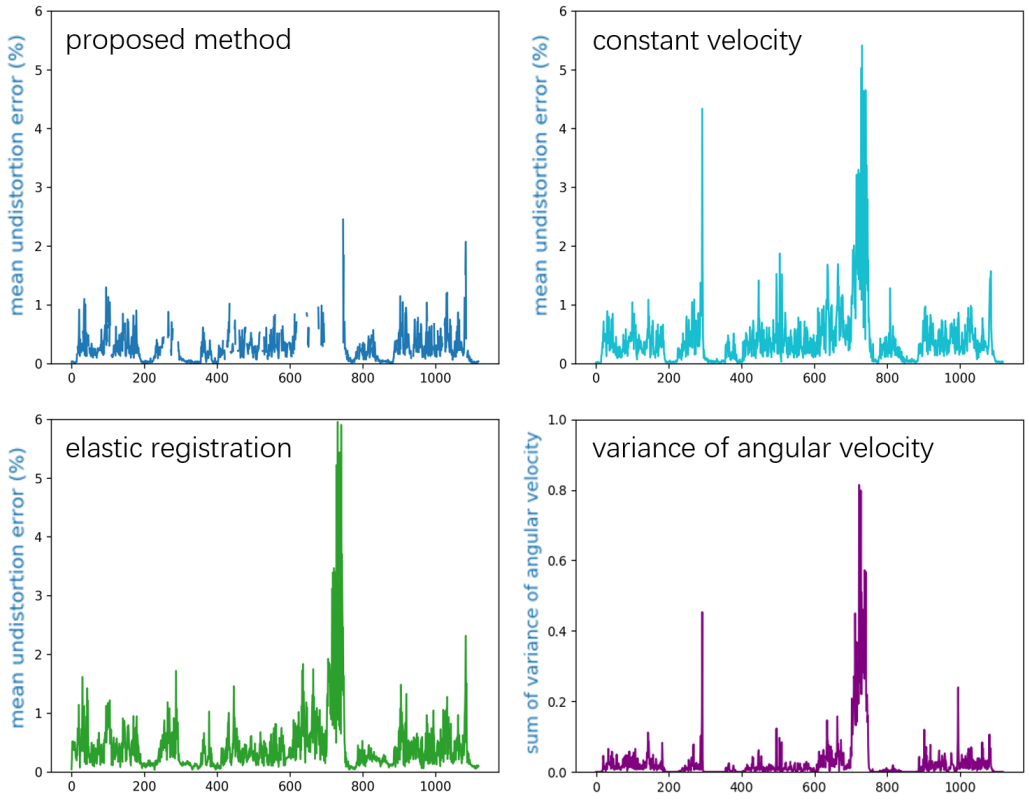


Figure 4.4: The mean distortion error along the scan index of sequence #3. The bottom-right figure shows the sum of the variance of angular velocity in each scan.

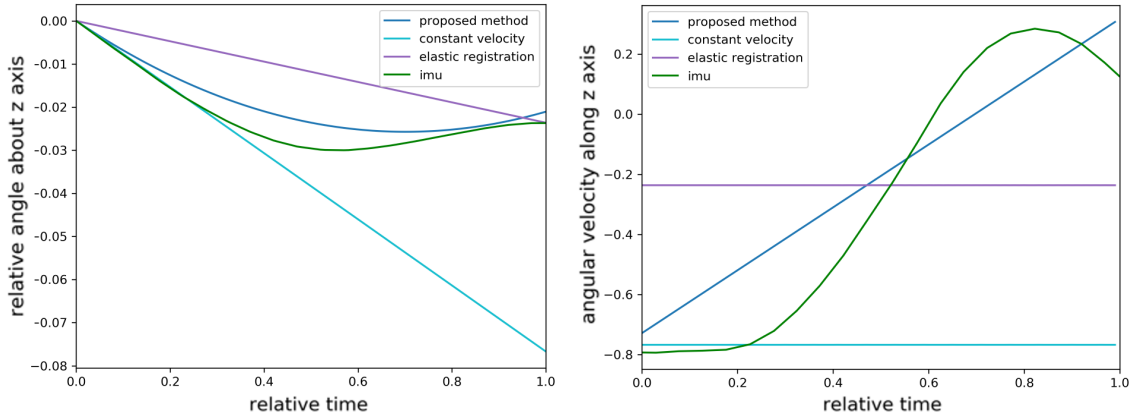


Figure 4.5: Relative rotation angle along $z$-axis (left) and angular velocity along $z$-axis (right).

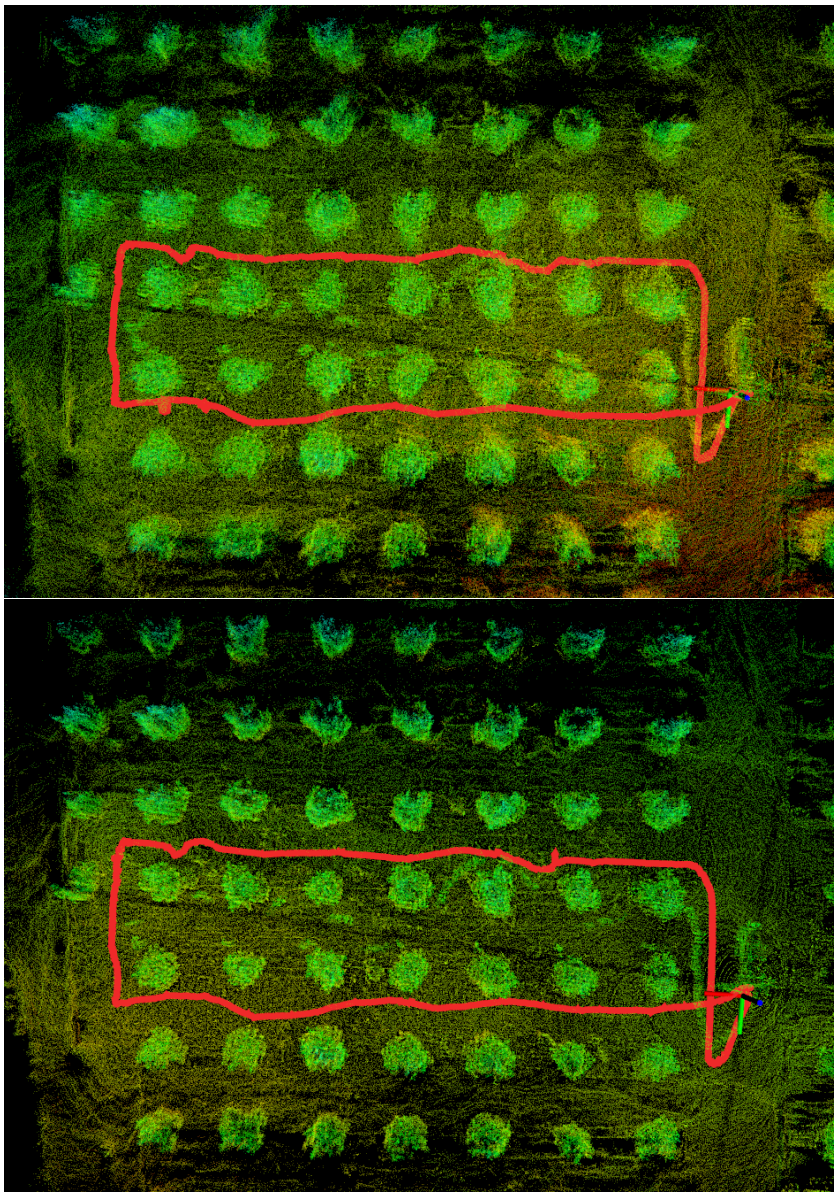

Figure 4.6: The map built by KISS-ICP with its original motion distortion correction method (top) and our proposed method (down). By using the proposed method, the LiDAR-odometry could output a better map with clearer and sharper contours.

during the current scan. The proposed method could model the angular acceleration, thus providing a better approximation in such a scenario.

## 4.3 Integration Test

The proposed method is also integrated into KISS-ICP and replaces its original motion distortion correction module with nearly zero modification. 4.6 shows a comparison of the result of KISS-ICP running in agricultural environments with its original method versus our proposed method. Although both setups can finish the sequence and correctly estimate the trajectory of the robot, with our proposed method, KISS-ICP could generate a better map. In the top figure, the map is blurry, and the contours of the trees are smoothed out, making them indistinguishable from each other. With our proposed method, the map is sharp and clean, and the edges of the trees are also preserved.

# Chapter 5

# Conclusion

## 5.1 Contributions

This dissertation presents a new motion distortion correction method for point clouds acquired from LiDAR sensors by estimating a constant acceleration motion model during the scan. By evaluating the proposed method in different scenarios and comparing it to other methods, our proposed method shows close performance to the constant velocity motion correction method and leading performance in agriculture environments.

## 5.2 Limitations

### 5.2.1 Motion Model

The proposed method assumes the motion model as a constant angular acceleration motion around a fixed axis, and a constant acceleration motion along a fixed direction. This assumption is not generically applicable and may limit the application of the proposed

method, especially for scenarios such as aerial robots and underwater robots.

### 5.2.2 Computation Overhead

For the experiment setup, the proposed method took an average of 36ms to process a scan, which is about 60% of the total execution time of the whole framework. This makes the proposed method less practical on platforms with limited computation resources.

## 5.3 Future Work

The scan-to-segment registration provides essential data for motion model estimation. However, the registration does not take the spatial continuity of the segments into account, forcing us to compromise either accuracy or validity and robustness. A better registration method that could take advantage of both the temporal and spatial continuity of the segments while consuming less computation power would be an interesting topic in the future.

# Bibliography


[1] Alismail, H., Baker, L.D., Browning, B.: Continuous trajectory estimation for 3d slam from actuated lidar. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 6096–6101. IEEE (2014)

[2] Bai, C., Xiao, T., Chen, Y., Wang, H., Zhang, F., Gao, X.: Faster-LIO: Lightweight Tightly Coupled Lidar-Inertial Odometry Using Parallel Sparse Incremental Voxels. IEEE Robotics and Autom. Letters **7**(2), 4861–4868 (2022)

[3] Behley, J., Stachniss, C.: Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In: Robotics: Science and Systems (2018)

[4] Besl, P.J., McKay, N.D.: A Method for Registration of 3-D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14**(2), 239–256 (1992)

[5] Bosse, M., Zlot, R.: Continuous 3d scan-matching with a spinning 2d laser. In: 2009 IEEE International Conference on Robotics and Automation. pp. 4312–4319. IEEE (2009)

[6] Chen, S.W., Nardari, G.V., Lee, E.S., Qu, C., Liu, X., Romero, R.A.F., Kumar, V.: S-LOAM: Semantic Lidar Odometry and Mapping for Forest Inventory. IEEE Robotics and Autom. Letters **5**(2), 612–619 (2020)

[7] Chen, X., Milioto, A., Palazzolo, E., Giguere, P., Behley, J., Stachniss, C.: SuMa++: Efficient LiDAR-based Semantic SLAM. In: IEEE/RSJ Int. Conf. on Intell. Robots and Systems (IROS). pp. 4530–4537 (2019)

[8] Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. Image and Vision Computing **10**(3), 145–155 (1992). https://doi.org/https://doi.org/10.1016/0262-8856(92)90066-C, range Image Understanding

[9] Debeunne, C., Vivet, D.: A review of visual-lidar fusion based simultaneous localization and mapping. Sensors **20**(7), 2068 (2020)

[10] Dellenbach, P., Deschaud, J.E., Jacquet, B., Goulette, F.: CT-ICP: Real-time Elastic LiDAR Odometry with Loop Closure. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 5580–5586 (2022)

[11] Dobrowolski, P.: Swing-twist decomposition in clifford algebra. In: arXiv preprint. pp. 3354–3361 (2015). https://doi.org/arXiv:1506.05481

[12] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3354–3361 (2012). https://doi.org/10.1109/CVPR.2012.6248074

[13] Hong, S., Ko, H., Kim, J.: Vicp: Velocity updating iterative closest point algorithm. In: 2010 IEEE International Conference on Robotics and Automation. pp. 1893–1898 (2010). https://doi.org/10.1109/ROBOT.2010.5509312

[14] Li, P., Wang, R., Wang, Y., Tao, W.: Evaluation of the icp algorithm in 3d point cloud registration. IEEE Access **8**, 68030–68048 (2020). https://doi.org/10.1109/ACCESS.2020.2986470

[15] Lin, J., Zhang, F.: Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV. In: IEEE Int. Conf. on Robotics and Autom. (ICRA). pp. 3126–3131 (2020)

[16] Palieri, M., Morrell, B., Thakur, A., Ebadi, K., Nash, J., Chatterjee, A., Kanellakis, C., Carlone, L., Guaragnella, C., Agha-Mohammadi, A.a.: LOCUS: A Multi-Sensor Lidar-Centric Solution for High-Precision Odometry and 3D Mapping in Real-Time. IEEE Robotics and Automation Letters **6**(2), 421–428 (2020)

[17] Pan, Y., Xiao, P., He, Y., Shao, Z., Li, Z.: MULLS: Versatile LiDAR SLAM via Multi-metric Linear Least Square. In: IEEE Int. Conf. on Robotics and Autom. (ICRA). pp. 11633–11640 (2021)

[18] Reinke, A., Palieri, M., Morrell, B., Chang, Y., Ebadi, K., Carlone, L., Agha-Mohammadi, A.A.: LOCUS 2.0: Robust and Computationally Efficient Lidar Odometry for Real-Time 3D Mapping. IEEE Robotics and Autom. Letters **7**(4), 9043–9050 (2022)

[19] Segal, A., Haehnel, D., Thrun, S.: Generalized-ICP. In: Robotics: science and systems. vol. 2, p. 435 (2009)

[20] Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., Rus, D.: LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In: IEEE/RSJ Int. Conf. on Intell. Robots and Systems (IROS). pp. 5135–5142 (2020)

[21] Tagliabue, A., Tordesillas, J., Cai, X., Santamaria-Navarro, A., How, J.P., Carlone, L., Agha-mohammadi, A.a.: LION: Lidar-Inertial Observability-Aware Navigator for Vision-Denied Environments. In: Int. Symposium on Experimental Robotics (ISER). pp. 380–390 (2020)

[22] Vizzo, I., Guadagnino, T., Mersch, B., Wiesmann, L., Behley, J., Stachniss, C.: KISS-ICP: In Defense of Point-to-Point ICP Simple, Accurate, and Robust Registration If Done the Right Way. IEEE Robotics and Autom. Letters **8**(2), 1029–1036 (2023)

[23] Wang, H., Wang, C., Chen, C.L., Xie, L.: F-LOAM: Fast LiDAR Odometry and Mapping. In: IEEE/RSJ Int. Conf. on Intell. Robots and Systems (IROS). pp. 4390–4396 (2021)

[24] Xu, W., Cai, Y., He, D., Lin, J., Zhang, F.: FAST-LIO2: Fast Direct LiDAR-Inertial Odometry. IEEE Transactions on Robotics **38**(4), 2053–2073 (2022)

[25] Xu, W., Zhang, F.: FAST-LIO: A Fast, Robust LiDAR-Inertial Odometry Package by Tightly-Coupled Iterated Kalman Filter. IEEE Robotics and Autom. Letters **6**(2), 3317–3324 (2021)

[26] Ye, H., Chen, Y., Liu, M.: Tightly Coupled 3D Lidar Inertial Odometry and Mapping. In: IEEE Int. Conf. on Robotics and Autom. (ICRA). pp. 3144–3150 (2019)

[27] Zhang, J., Singh, S.: LOAM: Lidar Odometry and Mapping in Real-time. In: Robotics: Science and Systems. vol. 2, pp. 1–9 (2014)

[28] Zhou, P., Guo, X., Pei, X., Chen, C.: T-LOAM: Truncated Least Squares LiDAR-Only Odometry and Mapping in Real Time. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–13 (2021)