sensors

Article

A Novel LiDAR–IMU–Odometer Coupling Framework for Two-Wheeled Inverted Pendulum (TWIP) Robot Localization and Mapping with Nonholonomic Constraint Factors

Yanwu Zhai and Songyuan Zhang *

State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150001, China; zslhit@126.com
* Correspondence: zhangsy@hit.edu.cn

Abstract: This paper proposes a method to solve the problem of localization and mapping of a two-wheeled inverted pendulum (TWIP) robot on approximately flat ground using a Lidar–IMU–Odometer system. When TWIP is in motion, it is constrained by the ground and suffers from motion disturbances caused by rough terrain or motion shaking. Combining the motion characteristics of TWIP, this paper proposes a framework for localization consisting of a Lidar-IMU-Odometer system. This system formulates a factor graph with five types of factors, thereby coupling relative and absolute measurements from different sensors (including ground constraints) into the system. Moreover, we analyze the constraint dimension of each factor according to the motion characteristics of TWIP and propose a new nonholonomic constraint factor for the odometry pre-integration constraint and ground constraint factor in order to add them naturally to the factor graph with the robot state node on SE(3). Meanwhile, we calculate the uncertainty of each constraint. Utilizing such a nonholonomic constraint factor, a complete lidar–IMU–odometry-based motion estimation system for TWIP is developed via smoothing and mapping. Indoor and outdoor experiments show that our method has better accuracy for two-wheeled inverted pendulum robots.

Keywords: Lidar-IMU-Odometer system; two-wheeled inverted pendulum robot; ground constraints; nonholonomic constraint factor

check for updates

Citation: Zhai, Y.; Zhang, S. A Novel LiDAR–IMU–Odometer Coupling Framework for Two-Wheeled Inverted Pendulum (TWIP) Robot Localization and Mapping with Nonholonomic Constraint Factors. *Sensors* **2022**, *22*, 4778. https://doi.org/10.3390/s22134778

Academic Editor: Hyun Myung

Received: 1 May 2022
Accepted: 18 June 2022
Published: 24 June 2022

1. Introduction

In recent years, Simultaneous localization and mapping (SLAM) for intelligent mobile robots has become a research hotspot [1,2]. Under GPS-denied and unknown environment conditions, mobile robots must be able to utilize onboard sensors to construct an environment map that can be used for navigation [3] and then rely on this environment map for navigation, collision avoidance, and path planning. Many solutions have been proposed based on different sensing methods, mainly divided into vision-based [4–6], lidar-based [7–9], and a combination of the two [10,11]. Meanwhile, in order to cope with complex environments, sensors such as IMUs and encoders that are less affected by the environment can be used to assist with visual [12,13] or lidar [14–18] localization. Although vision-based methods are particularly suitable for position recognition, their sensitivity to initialization, illumination, and range as well as their high depth computation cost makes them unreliable when supporting autonomous navigation systems.

On the other hand, lidar can directly obtain depth information about the surrounding environment, and lidar are immune to illumination changes. Many impressive LiDAR-based localization and mapping algorithms have been proposed. In these algorithms, the IMU, encoder and other high-frequency output sensors are usually used to assist with lidar localization and the deskew point cloud. In [7], the authors present lidar odometry and mapping (LOAM) for low-drift real time state estimation and mapping. In this algorithm, the IMU is only used to de-skew the point cloud and does not participate in lidar localization. Here, for the localization algorithm of the lidar–IMU system, we divided it into filter-based and optimization-based segments according to the coupling method. Filter-based methods typically utilize extended Kalman filtering to couple measurements from lidar and IMU to estimate the state of the robot frame-by-frame. In [14], Lynen et al. proposed a modular method that uses EKF to fuse IMU measurements with relative pose measurements from cameras, lidars, etc. In [15], the authors present a lidar-inertial state estimator called R-LINS, which uses a recursive error-state Kalman filter to estimate the robot's state. In [16], the authors propose a fast, robust and general LiDAR–IMU odometry framework based on an efficient and tightly-coupled iterative Kalman filter for fast, robust, and accurate LiDAR navigation. Optimization-based methods usually use nonlinear optimization to combine a fixed number of lidar keyframes and IMU pre-integration in order to estimate the optimal pose. In the literature [17], a tightly coupled framework lio-mapping of lidar and IMU has been introduced which uses a framework similar to VINS and achieves good accuracy; however, it cannot run in real time. In [18], the authors proposed a framework LIO-SAM for tightly coupling lidar and IMU via smoothing and mapping, achieving highly accurate real-time localization and mapping. Filtering-based methods usually only consider the influence of the previous frame on the current frame, which incurs lower computational costs. The optimization-based method usually adopts the sliding window method to evaluate the constraint relationship between the robot states of the previous n frames (including the current frame). The optimization-based method is more computationally expensive than the filtering-based method, however, it has accuracy. With improved hardware performance, optimization-based methods can run in real time, and should gradually become the mainstream of research. However, in scenarios with sparse structural features, such as long tunnels, wide roads, etc., lidar lacks constraints in certain directions, resulting in decreased localization accuracy. In [19], the authors found that when their mobile robot had constant acceleration or no rotation, the VINS system was unable to distinguish the magnitude of the true body acceleration and the direction of the local gravitational acceleration from that of the accelerometer bias. Similar situations can appear in scenarios where Lidar is degraded.

The localization of mobile ground platforms with unique system architectures has not previously been thoroughly investigated and discussed. Due to ground constraints, many vision-based [20,21] and lidar-based [22,23] SLAM algorithms use the SE(2) pose. However, the ground is rough in the natural environment, and a ground-mobile robot will be disturbed by motion disturbance. In order to better adapt to the natural environment, Ref. [24] proposes a new constraint model, stochastic SE(2) constraints, which usually parameterizes the robot pose on SE(3), allowing for slight disturbance in dimensions other than SE(2). Fan Zheng [25] extended the work of the above article and proposed a complete motion estimation system that utilizes SE(2) constraints and SE(3) pose parameterization.

In this paper, a factor graph is formulated to couple the lidar, IMU, and encoder measurements for the localization and mapping of TWIP robots on an approximately flat road surface. Inspired by the SE(2) constraint–SE(3) pose method used in [24], we propose a new factor with constraints to introduce the encoder's pre-integration and ground constraints, which we achieve by analyzing the motion profile of the robot and its deviation from planar motion (e.g., due to terrain unevenness or motion vibration) and formulating stochastic (i.e., "soft") instead of deterministic (i.e., "hard") constraints. This allows us to properly model the TWIP's almost-planar motion, as shown in Figure 1. Moreover, we use Lie algebra to represent the rotation in the pose imitation process. Lie algebra has no redundant parameters. Compared with Euler angles, Lie algebra does not have gimbal locking problems, and rotation integration based on Lie algebra can be integrated into a closed form, whereas Euler angles are only exact up to the first order [26]. The main novel contributions of this work are as follows:

- A complete LIDAR–IMU–encoder-coupled state estimation and mapping framework is proposed for a two-wheeled inverted pendulum robot. This method extends the

LIDAR–IMU to process odometric measurements and improves the system's localization accuracy.
- A new nonholonomic factor is proposed to introduce encoder pre-integration constraints and ground constraints, which are naturally coupled to the factor graph optimization of the system.
- Indoor and outdoor experiments demonstrate the improved localization accuracy and robustness of the proposed lidar–IMU–odometry coupling framework when the controller is mounted on a two-wheeled inverted pendulum robot navigating on an approximately flat surface.

The rest of the paper is organized as follows: Section 2 provides an overview of the entire methodology; Section 3 reviews the basic knowledge which is be used later in the paper; Section 4 details the five types of factors introduced in our algorithm; and Section 5 describes several experiments we conducted to demonstrate the effectiveness of the proposed method.

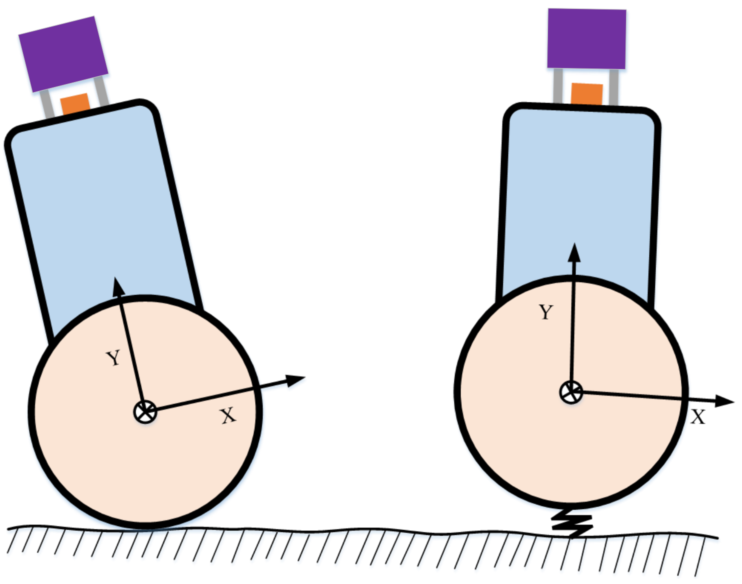

Figure 1. Constraint model: (**left**) deterministic constraints and (**right**) stochastic constraints, considering the motion perturbations addressed in this article.

2. Methodology Overview

This paper aims to estimate the motion state of a two-wheeled inverted pendulum robot (TWIP) via lidar–IMU–Odometry coupling. Compared with a six-degrees-of-freedom robot, TWIP cannot move sideways, and is constrained by the ground. Compared with autonomous vehicles on the ground, TWIP swings back and forth during movement to maintain the balance of movement. If we directly parameterize the robot pose on SE(3) without considering the ground constraints, the state estimation accuracy and robustness is depressed due to fewer constraints in the vertical direction of the lidar. If only ground constraints are considered, that is, the deterministic constraint, the performance of system motion estimation is degraded because the six dimensions of the pose are highly coupled with sensor observations. Therefore, neither accurate pose parameterization considering ground constraints nor 3D pose parameterization can satisfactorily represent the pose of the ground robot, because the former suffers from out-of-constraint motion while the latter does not use plane motion constraints. This prompted us to study the state estimation of the two-wheeled inverted pendulum robot through a lidar–IMU–Odometry system.

This paper proposes a framework for localization using a Lidar–IMU–Odometer system. The flow chart of this framework is shown in Figure 2. When the system receives a lidar scan, it first uses the measurement of the IMU to de-skew the point cloud, then extracts the line and surface feature points. If the pose exceeds a pre-set threshold, the current frame is selected as a keyframe; otherwise, it is discarded. Measurements from the IMU provide the initial pose. Then, the keyframes are matched to the global map to obtain the rough global pose and construct the lidar pose factor. The IMU and odometer measurements between two adjacent lidar keyframes are used to construct constraint factors via pre-integration techniques. Meanwhile, we use the random constraint model to construct the ground constraint factor. Finally, all factors are included in the factor graph and jointly optimized to obtain the optimal robot pose. The factor graph formulated for TWIP, which is shown in Figure 3, incorporates the measurement data of each sensor as a factor in our optimization system. In addition to the above four factors, this factor graph contains closed-loop factors grouped into two categories. The IMU pre-integration factor, the lidar pose factor, and the loop closure factor constrain each dimension of the robot pose, and are named holonomic constraint factors. However, the odometry pre-integration factor and the ground constraint factor only constrain part of the dimensions of the robot's pose, and thus are named non-holonomic constraint factors. For these factors we propose a nonholonomic constraint factor to add them to the system, inspired by the SE(2) constraint SE(3) parameterization method used in [24]. Utilizing such a constraining factor, a complete lidar–IMU–odometry-based motion estimation system for TWIP is developed via smoothing and mapping.

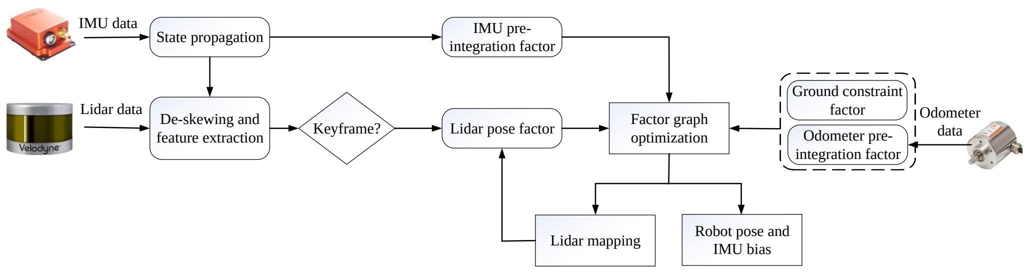

Figure 2. The flowchart of our algorithm.

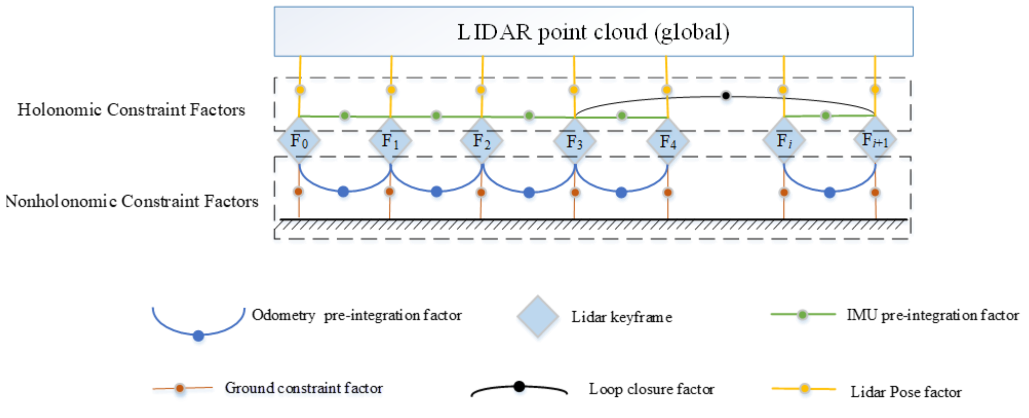
LIDAR point cloud (global)
Holonomic Constraint Factors
Nonholonomic Constraint Factors
F_0
F_1
F_2
F_3
F_4
F_i
F_{i+1}
Odometry pre-integration factor
Lidar keyframe
IMU pre-integration factor
Ground constraint factor
Loop closure factor
Lidar Pose factor

Figure 3. Factor graph optimization.

3. Preliminaries

This paper formulates the state estimation problem based on the Lidar–IMU–Odometry system in terms of a factor graph, that is, a nonlinear optimization problem involving the quantities existing on smooth manifolds. Before delving into details, we review several of the related practical geometric concepts.

3.1. On-Manifold Pose Parameterization

We usually use the Matrix Lie group to describe three-dimensional rotation, which is defined as follows:

$$\mathrm{SO}(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3x3}, \mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I} \right\} \tag{1}$$

However, the rotation combination operation in the SO(3) involves matrix multiplication in the optimisation process, which cannot be directly operated as Euclidean vectors can. Due to Lie groups having the properties of differential manifolds, the tangent space to the manifold (at the identity), called the Lie algebra, can be introduced as a representation of the most diminutive vector form, and is related to Lie groups through matrix exponents and logarithms. Lie algebra is defined as the follows:

$$so(3)=\left\{\phi \in \mathbb{R}^3, \Phi = \phi^\wedge \in \mathbb{R}^{3\times 3}\right\} \tag{2}$$

where ϕ is the rotation vector of $\mathbb{R}^3$, corresponding to a 3 $\times$ 3 skew-symmetric matrix. The hat operator $(\bullet)^\wedge$ is used to find the antisymmetric matrix of a vector, and its form is

$$\Phi = \phi^\wedge = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix} \phi = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} \tag{3}$$

At the same time, we define the inverse mapping of the hat operator $(\bullet)^\vee$ as follows:

$$\phi = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} = \Phi^\vee \tag{4}$$

For any two 3D vectors **r** and **s** and the rotation matrix **R**, the skew-symmetric matrix has the following properties:

$$\mathbf{r}^\wedge \mathbf{s} = -\mathbf{s}^\wedge \mathbf{r} \tag{5}$$

$$(\mathbf{R}\mathbf{s})^\wedge = \mathbf{R}\mathbf{s}^\wedge \mathbf{R}^T \tag{6}$$

$$\left(\mathbf{r}^\wedge\right)^{\mathrm{T}} = -\mathbf{r}^\wedge \tag{7}$$

$$\mathbf{r}^\wedge \mathbf{r}^\wedge = \mathbf{r}\mathbf{r}^{\mathrm{T}} - \mathbf{I}_{3\times 3} \tag{8}$$

Lie algebra can be transformed into a Lie group by exponential mapping (Rodriguez formula):

$$\exp\left(\phi^\wedge\right) = \mathbf{I} + \frac{\sin(\|\phi\|)}{\|\phi\|}\phi^\wedge + \frac{1-\cos\|\phi\|}{\|\phi\|^2}\left(\phi^\wedge\right)^2 \tag{9}$$

A first-order approximation of the exponential map used later on is

$$\exp\left(\phi^\wedge\right) \approx \mathbf{I} + \phi^\wedge \tag{10}$$

In the optimization process, we often use a left-multiplicative perturbation on the manifold through the left Jacobian matrix, approximating a local on-manifold transformation with Lie algebra:

$$\exp\left(\delta\phi^\wedge\right)\exp\left(\phi^\wedge\right) = \exp\left(\left(\mathbf{J}_l^{-1}(\phi)\delta\phi + \phi\right)^\wedge\right) \tag{11}$$

where the left Jacobian matrix is

$$\mathbf{J}_l(\phi) = \frac{\sin\theta}{\theta}\mathbf{I} + \left(1 - \frac{\sin\theta}{\theta}\right)\mathbf{a}\mathbf{a}^{\mathrm{T}} + \frac{1-\cos\theta}{\theta}\mathbf{a}^\wedge \tag{12}$$

where θ and **a** are the norm and direction of the rotation vector $\boldsymbol{\phi}$, $\boldsymbol{\phi}=\theta\boldsymbol{a}$. The inverse of the left Jacobian matrix is

$$\mathbf{J}_l^{-1}(\phi) = \frac{\theta}{2}\cot\frac{\theta}{2}\mathbf{I} + \left(1 - \frac{\theta}{2}\cot\frac{\theta}{2}\right)\mathbf{a}\mathbf{a}^{\mathrm{T}} - \frac{\theta}{2}\mathbf{a}^\wedge \tag{13}$$

Another valuable property of the exponential map is

$$\mathbf{R}\mathrm{Exp}(\phi)\mathbf{R}^T = \exp\left(\mathbf{R}\phi^\wedge\mathbf{R}^T\right) = \mathrm{Exp}(\mathbf{R}\phi) \tag{14}$$

3.2. Factor Graph Optimization

As TWIP moves, lidar measurements are continuously collected to estimate the robot's motion state and model the surrounding environment. This process can be described by a factor graph, where the robot states to be estimated are state nodes. The continuously collected measurements are introduced as factor nodes connected by edges to the involved state nodes (see Figure 3). In general, we assume that these measurements are independent of each other and are affected by Gaussian noise. The robot state estimation problem represented by the factor graph above can be formulated for maximum a posteriori estimation. Therefore, we formulate the above problem as the following nonlinear least-squares problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_i \|\mathbf{r}_i(\mathbf{x})\|^2_{\Sigma_i} \tag{15}$$

where $\mathbf{r}_i$ is the zero-mean residual associated with measurement i, $\mathbf{x}$ is the parameter vector to be optimized, Σ_i is the measurement covariance, and $\|\mathbf{r}_i\|^2_{\Sigma_i} = \mathbf{r}_i^{\mathrm{T}}\Sigma_i^{-1}\mathbf{r}_i$ is the energy norm.

The above least squares problem can be solved iteratively using the Gauss–Newton method. At each GN iteration, we approximate the optimized objective function (the residual function) by first-order Taylor expansion at the current estimated state $\hat{\mathbf{x}}^-$:

$$\begin{aligned} \Delta\hat{\mathbf{x}} &= \arg\min_{\Delta\mathbf{x}} \sum_i \|\mathbf{r}_i(\mathbf{x} \oplus \Delta\mathbf{x})\|^2_{\Sigma_i} \\ &= \arg\min_{\Delta\mathbf{x}} \sum_i \|\mathbf{r}_i(\mathbf{x}) + \mathbf{H}_i\Delta\mathbf{x}\|^2_{\Sigma_i} \end{aligned} \tag{16}$$

where $\mathbf{H}_i = \frac{\partial\mathbf{r}_i}{\partial\mathbf{x}}$ is the Jacobian matrix of the *ith* residual for the robot state and $\Delta\mathbf{x}$ is the increment of the state obtained at each iteration. We define the generalized update operation, $\oplus$, which maps a change in the error state to one in the entire state. After solving the linearized equation, the current state is updated as $\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- \oplus \Delta\mathbf{x}$. This linearization process is then repeated until convergence

4. State Estimation with Accurate Parameterization of TWIP

4.1. Pose Parameterization of TWIP

Compared with a six-degrees-of-freedom robot, TWIP cannot move sideways and is constrained by the ground. Compared with autonomous vehicles on the ground, TWIP swings back and forth during movement to maintain the balance of movement. Therefore, we cannot characterize its motion state completely in SE(2) or SE(3) space. A simplified diagram of this movement is shown in Figure 4.

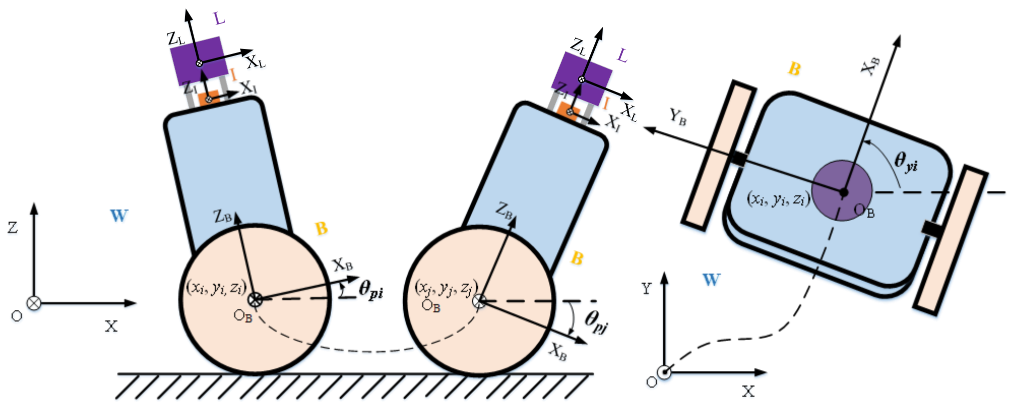

Figure 4. Simplified diagram of TWIP motion.

In order to analyze the movement of TWIP more conveniently, its coordinate system is defined as follows:

Lidar coordinate system (L): The lidar frame originates at the lidar center. The y-axis points to the left, the z-axis points upward, and the x-axis points forward, coinciding with the positive direction of TWIP.

IMU coordinate system (I): The IMU is located below the lidar; its frame originates at the IMU measurement center. The x-, y-, and z-axes are parallel to (L), pointing in the exact directions, and the z-axis points forward, coinciding with the z-axis of the lidar frame.

Robot body coordinate system (B): As shown in Figure 4, the robot body frame is connected to the robot body during the movement process. Its origin is at the midpoint of the line connecting the centers of the two wheels. The x-, y-, and z-axes are parallel to (L), pointing in the exact directions, and the z-axis points forward, coinciding with the z-axis of the lidar frame.

World coordinate system (W): The world frame is the gravity-aligned coordinate system, which is initially coincident with the robot body coordinate system (B).

All of the above frames follow the right-hand rule. We calibrate the transformation between the lidar frame and the robot body frame and between the IMU frame and the robot body frame. The transformation between the robot body coordinate system and the world coordinate system is the motion state of the robot, that is, the target parameters for optimization. Figure 5 shows the posture of TWIP as obtained by the Attitude and Heading Reference System (ARHS) during movement. As seen from the figure, when TWIP moves, there are movements on the two-dimensional plane (yaw) and front and back swings (pitch), as well as movement disturbances in the roll direction. Therefore, we parameterize the robot's pose on SE(3). The motion state of the robot corresponding to the lidar keyframe F_i at time i is

$$\boldsymbol{K}_i = \begin{bmatrix} \mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}} & \mathbf{P}_{\mathrm{B}_i}^{\mathrm{W}} \end{bmatrix} \tag{17}$$

where $\mathbf{P}_{\mathrm{B}_i}^{\mathrm{W}} = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^{\mathrm{T}}$ is the translation and $\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}} \in \mathbb{R}^{3x3}$ is the robot's rotation. Because the rotation matrix is not additive, it is not convenient for iterative optimization; thus, we use Euler angles and Lie algebra to represent the robot's rotation in the following derivation. Euler angles easily distinguish the constraint dimension of each factor, and the pitch does not reach ± 90 degrees during the movement of TWIP, avoiding any problems with gimbal lock. In this paper, $\boldsymbol{\theta} = \begin{bmatrix} \theta_r & \theta_p & \theta_y \end{bmatrix}^{\mathrm{T}}$ represents the Euler angle and θ_r, θ_p, θ_y represents the values of the roll, pitch, and yaw angles, respectively. The rotation integral based on Euler angles is only exact up to the first order in robot state estimation [26]. Therefore, we use Lie algebra, $\phi = \begin{bmatrix} \phi_1 & \phi_2 & \phi_3 \end{bmatrix}^{\mathrm{T}}$, to represent the rotation in the optimization process. Lie algebra can be converted to rotation matrices using Equation (7). According to the motion characteristics of TWIP, $\begin{bmatrix} x & y & \theta_p & \theta_y \end{bmatrix}$ is the result of the robot's motion, which is named the primary parameter, while $\begin{bmatrix} z & \theta_r \end{bmatrix}$ is the motion disturbance caused by uneven ground or the vibration of the robot, and is named the second parameter. The partial parameters in the figure are shown in Table 1.

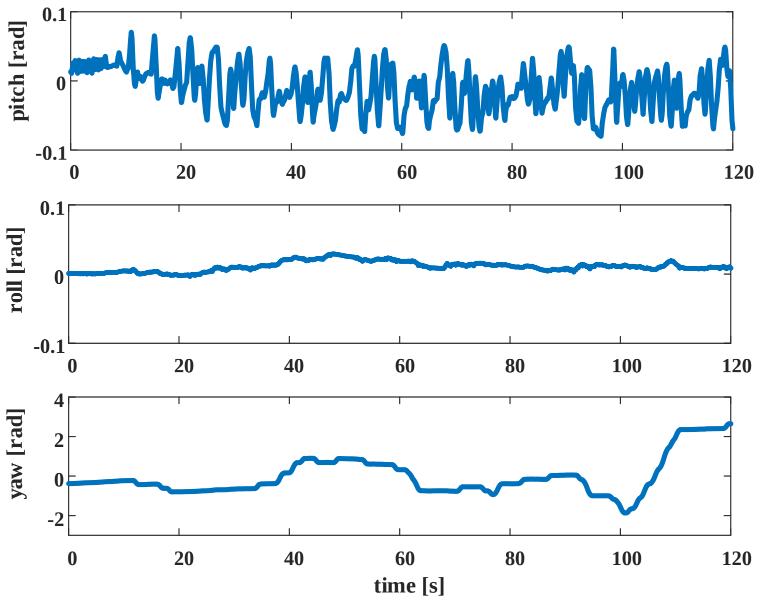

Figure 5. Pose of TWIP during movement (Euler angle).

Table 1. State parameters and frame of TWIP.

Symbols	Descriptions
X, Z, O	The x-axis, z-axis, and origin of the world frame
X_B, Z_B, O_B	The x-axis, z-axis, and origin of the robot body frame
X_L, Z_L, O_L	The x-axis, z-axis, and origin of the lidar frame
$(x_k, y_k, z_k), k = i, j$	The position of the robot in the world frame at time k (in meters)
$\theta_{pk},\ k = i, j$	The pitch angle of the robot relative to the world frame at time k (in radians)
$\theta_{yk},\ k = i, j$	The yaw angle of the robot relative to the world frame at time k (in radians)

4.2. Coupling of Lidar–IMU–Odometer via Factor Graph

In this section, we introduce five types of factors to construct factor graphs in order to estimate the motion state of TWIP. The robot's state at a specific time is attributed to the graph's nodes. According to the constraints of factors on nodes, these factors are divided into holonomic and nonholonomic constraints. The holonomic constraint factors include the lidar pose factor, the IMU pre-integration factor, and the loop closure factor. The nonholonomic constraint factor consists of the odometer pre-integration and ground constraints. A new robot state node is added to the graph when the change in robot pose exceeds a user-defined threshold. The factor graph is optimized upon the insertion of a new node using incremental smoothing and mapping with the Bayes tree (iSAM2) [17]. The process for generating these factors is described in the following sections.

4.2.1. Lidar Pose Factor

Mechanical 3D LiDAR senses its surroundings by rotating a vertically aligned array of laser beams. When a lidar keyframe F_i is selected, we first de-skew the point cloud. The movement of the carrier causes point cloud distortion during the lidar data collection process, and the points in a lidar scan are not collected at the same time; that is, the frames of different lidar points are inconsistent. The bias of the IMU obtained by the factor graph optimization is used to correct the IMU measurement, which can estimate the pose of the carrier. Here, we find the pose corresponding to each lidar point, transform them into the same coordinate system, and obtain the resulting point cloud without distortion. The feature point matching method is used in this paper because it is more robust and efficient than noise-sensitive matching methods, such as Iterative Closest Point (ICP). Similar to [7], this paper uses plane feature points and edge feature points as matching points, which are extracted by smoothness. For any point $\mathbf{p}_k$ in F_i, We find ten straight points of $\mathbf{p}_k$ from the same sweep, of which half are on either side of $\mathbf{p}_k$. The smoothness of $\mathbf{p}_k$ can then be calculated by the following formula:

$$c = \frac{1}{|D| \cdot \|r_k\|} \left\| \sum_{j \in D, j \neq k} \left(r_j - r_k\right) \right\| \tag{18}$$

where D is the point set adjacent to this point we selected and $|D|$ is the number of points in D .

We use $\mathrm{F}_i = \left\{\mathrm{F}_i^{\varepsilon}, \mathrm{F}_i^{\pi}\right\}$ to represent the feature points extracted from the lidar scan at time i, where F_i^{π} represents the planar points and $\mathrm{F}_i^{\varepsilon}$ represents the edge points. Points with slight smoothness are selected as plane points and points with significant smoothness are selected as edge points. A detailed description of the feature extraction process is provided in [7].

After the above processing is complete, we have the feature points of the current lidar frame without distortion. We then need to find the plane and edge line corresponding to the current frame feature point in the target point cloud for the feature-based matching method. Unlike traditional algorithms such as lio_sam and lego_loam, which take lidar scans close to the current frame in the time series as matching point clouds, this paper searches for the matching point cloud of the current frame in Euclidean space, which is more accurate because our optimization process is to minimize the point-to-line and point-to-plane spatial distances. Therefore, in this algorithm the position of the current robot is taken as a priori, and a part of the point cloud is extracted from the global map to align with the feature points without distortion in the current frame in order to estimate the pose of the robot. The IMU provides the initial position. The global map consists of edge feature maps and plane feature maps, which are updated and maintained, respectively. The global map is stored in an incremental 3D tree [18]. Compared with the traditional dynamic data structure, incremental 3D tree has an efficient nearest neighbor search and supports the incremental update of the map. Similar to [18], we convert the current frame to the world frame to obtain ${}^{\mathrm{W}}\mathrm{F}_i = \left\{{}^{\mathrm{W}}\mathrm{F}_i^{\varepsilon}, {}^{\mathrm{W}}\mathrm{F}_i^{\pi}\right\}$, while the IMU provides the initial pose. We find the N nearest edge points in the global edge map for any edge feature point $\mathbf{p}_i^{\varepsilon}$, then calculate the variance of these nearby points. If one of the eigenvalues of the variance matrix is significantly larger than the other two, the points are distributed in a line. The eigenvector $\mathbf{n}_i^{\varepsilon}$, which corresponds to the largest eigenvalue, is the direction vector of the line. The edge point $\mathbf{p}_j^{\varepsilon}$ closest to the feature point is taken as a point on the line. The distance from the feature point to the global line is

$$d_i^{\varepsilon} = \left\| (\mathbf{n}_i^{\varepsilon})^{\wedge} \left(\tilde{\mathbf{T}}_{\mathrm{L}_i}^{\mathrm{W}} \mathbf{p}_i^{\varepsilon} - \mathbf{p}_j^{\varepsilon} \right) \right\|_2 \tag{19}$$

For any plane feature point $\mathbf{p}_i^{\pi}$, we find the M nearest plane points in the global plane map and calculate the variance of these nearby points. If one of the eigenvalues of the variance matrix is significantly smaller than the other two, the points are distributed in a plane. The eigenvector $\mathbf{n}_i^{\pi}$, which corresponds to the smallest eigenvalue, is the normal vector of the plane. We select the closest plane feature point $\mathbf{p}_j^{\pi}$ to the feature point as a point on the plane. Then, the distance from the feature point to the global plane is

$$d_i^{\pi} = \left| (\mathbf{n}_i^{\pi}) \cdot \left(\tilde{\mathbf{T}}_{\mathrm{L}_i}^{\mathrm{W}} \mathbf{p}_i^{\pi} - \mathbf{p}_j^{\pi} \right) \right| \tag{20}$$

We can obtain the optimal pose of the current lidar scan relative to the world frame by minimizing the point-to-line and point-to-face distances:

$$\min_{\tilde{\mathbf{T}}_{\mathrm{L}_i}^{\mathrm{W}}} \sum_{i=0}^{N} \left(d_i^{\varepsilon}\right)^2 + \sum_{i=0}^{M} \left(d_i^{\pi}\right)^2 \tag{21}$$

After optimization, we have the optimal absolute transformation $\tilde{\mathbf{T}}_{\mathrm{L}_i}^{\mathrm{W}}$ from the current keyframe to the world frame, which is used to measure the lidar pose factor.

We obtain the residual of the lidar pose factor using the following formula:

$$\mathbf{r}_{\mathrm{L}_i} = \log\left(\left(\mathbf{T}_{\mathrm{B}_i}^{\mathrm{W}} \mathbf{T}_{\mathrm{L}}^{\mathrm{B}}\right)^{\mathrm{T}} \tilde{\mathbf{T}}_{\mathrm{L}_i}^{\mathrm{W}}\right) \tag{22}$$

where $\mathbf{T}_{\mathrm{L}}^{\mathrm{B}}$ is the transformation between the lidar coordinate system and the robot body coordinate system, calibrated in advance.

The variance of the factor is calculated by the method used in [26]:

$$\Sigma_{\mathrm{L}_i} = \sum_{k=1}^{N} \mathbf{H}_k^T \mathbf{\Lambda}^{-1} \mathbf{H}_k \tag{23}$$

$$\mathbf{H}_k = \begin{bmatrix} \left(\tilde{\mathbf{R}}_{\mathrm{L}_i}^{\mathrm{W}} \mathbf{p}_k + \tilde{\mathbf{P}}_{\mathrm{L}_i}^{\mathrm{W}}\right)^{\wedge} & -\tilde{\mathbf{R}}_{\mathrm{L}_i}^{\mathrm{W}} \end{bmatrix} \tag{24}$$

$$\mathbf{\Lambda} = \begin{bmatrix} 2\sigma_x^2 & 0 & 0 \\ 0 & 2\sigma_y^2 & 0 \\ 0 & 0 & 2\sigma_z^2 \end{bmatrix} \tag{25}$$

where $\mathbf{p}_k$ is the feature point in the current lidar keyframe, and is transformed into the matching point cloud through the optimal transformation, $\tilde{\mathbf{T}}_{\mathrm{L}_i}^{\mathrm{W}} = \begin{bmatrix} \tilde{\mathbf{R}}_{\mathrm{L}_i}^{\mathrm{W}} & \tilde{\mathbf{P}}_{\mathrm{L}_i}^{\mathrm{W}} \\ \mathbf{0} & 1 \end{bmatrix}$, obtained through the above matching. Here, N is the number of features in the current lidar keyframe, $\mathbf{\Lambda}$ is the zero-mean white Gaussian noise of the measurements, and σ_x, σ_y and σ_z are the noise sigma (in meters) on the x, y, and z axes, respectively. The sigmas are multiplied by two, as both the target point and the matching are affected by this noise.

4.2.2. IMU Pre-Integration Factor

The main components of an IMU are a three-axis accelerometer and a three-axis gyroscope that measure the linear acceleration and angular velocity of the carrier, respectively. The pose transformation of the carrier to an inertial frame can be obtained by integral operation. The raw measurements, $\tilde{\mathbf{a}}_t$ and $\tilde{\omega}_t$, at moment t are provided by

$$\begin{aligned} \tilde{\mathbf{a}}_t &= \mathbf{a}_t + \mathbf{b}_{a_t} + \mathbf{R}_{\mathrm{W}}^{\mathrm{I}_t} \mathbf{g}^w + \mathbf{n}_a \\ \tilde{\omega}_t &= \omega_t + \mathbf{b}_{\omega_t} + \mathbf{n}_\omega \end{aligned} \tag{26}$$

These IMU measurements are measured in the IMU frame, and are affected by the additive noise, $\mathbf{n}_a$,$\mathbf{n}_\omega$, the acceleration bias, $\mathbf{b}_{a_t}$, and the gyroscope bias, $\mathbf{b}_{\omega_t}$. Generally, in this paper we model the bias as a random walk with Gaussian derivatives, while the additive noise of the accelerometer and the gyroscope is assumed to be subject to a Gaussian distribution:

$$\mathbf{n}_a \sim \mathcal{N}\left(\mathbf{0}, \sigma_a^2\right), \mathbf{n}_\omega \sim \mathcal{N}\left(\mathbf{0}, \sigma_\omega^2\right) \tag{27}$$

Over time, high-frequency IMU measurement leads to rapid growth in the number of variables in the optimization, increasing the computational cost. We therefore convert the IMU measurements between keyframes into motion constraints via the pre-integration technique, reducing the computational cost and providing a factor, namely, the IMU pre-integration factor. The parameters to be optimized include the robot's pose and the bias of the IMU.

According to the kinematic model in [26], the pose and velocity of the current keyframe in the world frame can be obtained from the measurement of the previous frame. With the pose and velocity of a keyframe i, the pose and velocity of the current keyframe j are:

$$\begin{aligned}\mathbf{P}_{\mathrm{I}_j}^{\mathrm{W}} &=\mathbf{P}_{\mathrm{I}_i}^{\mathrm{W}} + \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}}\Delta t + \iint_{t\in[i,j]}\left(\mathbf{R}_{\mathrm{I}_t}^{\mathrm{W}}(\tilde{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a) - \mathbf{g}^{\mathrm{W}}\right)dt^2 \\ \mathbf{v}_{\mathrm{I}_j}^{\mathrm{W}} &=\mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}} + \int_{t\in[i,j]}\left(\mathbf{R}_{\mathrm{I}_t}^{\mathrm{W}}(\tilde{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a) - \mathbf{g}^{\mathrm{W}}\right)dt \\ \mathbf{R}_{\mathrm{I}_j}^{\mathrm{W}} &=\mathbf{R}_{\mathrm{I}_i}^{\mathrm{W}}\,\mathrm{Exp}\left(\int_{t\in[i,j]}(\tilde{\boldsymbol{\omega}}_t - \mathbf{b}_{\omega_t} - \mathbf{n}_\omega)dt\right)\end{aligned} \tag{28}$$

Using Formula (28), we iterate the IMU integral between times i and j, and obtain:

$$\begin{aligned}\mathbf{I}_{\mathrm{I}_j}^{\mathrm{W}} &= \mathbf{P}_{\mathrm{I}_i}^{\mathrm{W}} + \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}}\Delta t + \sum_{k=i}^{j-1}\left(\mathbf{v}_{\mathrm{I}_k}^{\mathrm{W}}\Delta t - \frac{1}{2}\mathbf{g}^{\mathrm{W}}\Delta t^2 + \frac{1}{2}\mathbf{R}_{\mathrm{I}_k}^{\mathrm{W}}(\tilde{\mathbf{a}}_k - \mathbf{b}_{a_k} - \mathbf{n}_a)\Delta t^2\right) \\ \mathbf{v}_{\mathrm{I}_j}^{\mathrm{W}} &= \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}} + \mathbf{g}^{\mathrm{W}}\Delta t_{ij} + \sum_{k=i}^{j-1}\mathbf{R}_{\mathrm{I}_k}^{\mathrm{W}}(\tilde{\mathbf{a}}_k - \mathbf{b}_{a_k} - \mathbf{n}_a)\Delta t \\ \mathbf{R}_{\mathrm{I}_j}^{\mathrm{W}} &= \mathbf{R}_{\mathrm{I}_i}^{\mathrm{W}}\prod_{k=i}^{j-1}\mathrm{Exp}(\tilde{\omega}_k - \mathbf{b}_{\omega_k} - \mathbf{n}_\omega)\Delta t\end{aligned} \tag{29}$$

where Δt is the interval between two adjacent measurements, $\Delta t_{ij} = \sum_{k=i}^{j-1}\Delta t$. Next, we separate the parts related to IMU measurement from the pose and velocity. The parts related to IMU measurement are as follows:

$$\begin{aligned}\Delta\tilde{\mathbf{P}}_{I_j}^{I_i} &= \sum_{k=i}^{j-1}\left[\Delta\mathbf{V}_k^i\Delta t + \frac{1}{2}\mathbf{R}\left(\gamma_k^i\right)(\tilde{\mathbf{a}}_k - \mathbf{b}_{a_k})\Delta t^2\right] - \delta\mathbf{P}_{ij} \\ \Delta\tilde{\mathbf{V}}_{I_j}^{I_i} &= \sum_{k=i}^{j-1}\mathbf{R}_k^i(\tilde{\mathbf{a}}_k - \mathbf{b}_{a_k})\Delta t - \delta\mathbf{V}_{ij} \\ \Delta\tilde{\mathbf{R}}_{I_j}^{I_i} &= \left(\prod_{k=i}^{j-1}\mathrm{Exp}((\tilde{\omega}_k - \mathbf{b}_{\omega_k})\Delta t)\right)\mathrm{Exp}(-\delta\boldsymbol{\varphi}_{ij})\end{aligned} \tag{30}$$

where $\delta\mathbf{P}_{ij}$, $\delta\mathbf{V}_{ij}$, and $\delta\boldsymbol{\varphi}_{ij}$ are the measurement error of IMU, and are obtained by iterating the initial value at time i. In order to avoid repeated integration due to changes in the bias during the optimization process, we use first-order approximation to linearize the bias terms, as follows:

$$\begin{aligned}\boldsymbol{\Delta}\tilde{\boldsymbol{P}}_{\mathrm{I}_j}^{\mathrm{I}_i} &\approx \boldsymbol{\Delta}\tilde{\boldsymbol{P}}_{\mathrm{I}_j}^{\mathrm{I}_i} + \mathbf{J}_{b_a}^{\boldsymbol{\Delta}\tilde{\boldsymbol{P}}}\delta\mathbf{b}_a + \mathbf{J}_{b_w}^{\boldsymbol{\Delta}\boldsymbol{P}}\delta\mathbf{b}_w \\ \boldsymbol{\Delta}\tilde{\boldsymbol{V}}_{\mathrm{I}_j}^{\mathrm{I}_i} &\approx \boldsymbol{\Delta}\tilde{\boldsymbol{V}}_{\mathrm{I}_j}^{\mathrm{I}_i} + \mathbf{J}_{b_a}^{\boldsymbol{\Delta}\tilde{\boldsymbol{V}}}\delta\mathbf{b}_a + \mathbf{J}_{b_w}^{\boldsymbol{\Delta}\boldsymbol{V}}\delta\mathbf{b}_w \\ \boldsymbol{\Delta}\tilde{\boldsymbol{R}}_{\mathrm{I}_j}^{\mathrm{I}_i} &\approx \boldsymbol{\Delta}\tilde{\boldsymbol{R}}_{\mathrm{I}_j}^{\mathrm{I}_i}\mathrm{Exp}\left(\mathbf{J}_{b_\omega}^{\boldsymbol{\Delta}\tilde{\boldsymbol{R}}}\delta\mathbf{b}_w\right)\end{aligned} \tag{31}$$

The items related to the state at times i and j are as follows:

$$\begin{aligned}\Delta\mathbf{P}_{\mathrm{I}_j}^{\mathrm{I}_i} &= \left(\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\right)^{\mathrm{T}}\left(\mathbf{P}_{\mathrm{I}_j}^{\mathrm{W}} - \mathbf{P}_{\mathrm{I}_i}^{\mathrm{W}} - \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}}\Delta t + \tfrac{1}{2}\mathbf{g}^{\mathrm{W}}\Delta t^2\right) \\ \Delta\mathbf{V}_{\mathrm{I}_j}^{\mathrm{I}_i} &= \left(\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\right)^{\mathrm{T}}\left(\mathbf{v}_{\mathrm{I}_j}^{\mathrm{W}} - \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}} + \mathbf{g}^{\mathrm{W}}\Delta t\right) \\ \Delta\mathbf{R}_{\mathrm{I}_j}^{\mathrm{I}_i} &= \left(\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\right)^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\end{aligned} \tag{32}$$

We construct the motion constraints of the IMU between two keyframes, which naturally provide the IMU pre-integration factor. This factor takes the bias of the IMU as an optimization parameter, and its residual is defined as follows:

$$\begin{aligned}
\mathbf{r}_{\mathbf{P}} &= \Delta\mathbf{P}_{\mathrm{I}_j}^{\mathrm{I}_i} - \Delta\tilde{\mathbf{P}}_{\mathrm{I}_j}^{\mathrm{I}_i} \\
&= \left(\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\right)^{\mathrm{T}}\left(\mathbf{P}_{\mathrm{B}_j}^{\mathrm{W}} - \mathbf{P}_{\mathrm{B}_i}^{\mathrm{W}} - \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}}\Delta t + \frac{1}{2}\mathbf{g}^{\mathrm{W}}\Delta t^2\right) - \sum_{k=i}^{j-1}\left[\Delta\mathbf{V}_k^i\Delta t + \frac{1}{2}\mathbf{R}\left(\gamma_k^i\right)\left(\tilde{\mathbf{a}}_k - \mathbf{b}_{a_k}\right)\Delta t^2\right] \\
\mathbf{r}_{\mathbf{v}} &= \Delta\mathbf{V}_{\mathrm{I}_j}^{\mathrm{I}_i} - \Delta\tilde{\mathbf{V}}_{\mathrm{I}_j}^{\mathrm{I}_i} \\
&= \left(\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\right)^{\mathrm{T}}\left(\mathbf{v}_{\mathrm{I}_j}^{\mathrm{W}} - \mathbf{v}_{\mathrm{I}_i}^{\mathrm{W}} + \mathbf{g}^{\mathrm{W}}\Delta t\right) - \sum_{k=i}^{j-1}\mathbf{R}_k^i\left(\tilde{\mathbf{a}}_k - \mathbf{b}_{a_k}\right)\Delta t \\
\mathbf{r}_{\mathbf{R}} &= \log\left(\left(\Delta\mathbf{R}_{\mathrm{I}_j}^{\mathrm{I}_i}\right)^{\mathrm{T}}\Delta\tilde{\mathbf{R}}_{\mathrm{I}_j}^{\mathrm{I}_i}\right) \\
&= \log\left(\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\right)^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}\prod_{k=i}^{j-1}\mathrm{Exp}\left(\left(\tilde{\omega}_k - \mathbf{b}_{w_{t_i}}\right)\Delta t\right)\right) \\
\mathbf{r}_{\mathbf{b}_a} &= \mathbf{b}_{a_j} - \mathbf{b}_{a_i} \\
\mathbf{r}_{\mathbf{b}_\omega} &= \mathbf{b}_{\omega_j} - \mathbf{b}_{\omega_i}
\end{aligned} \tag{33}$$

where $\mathbf{R}_{\mathrm{I}}^{\mathrm{B}}$ is the rotation between the lidar coordinate system and the robot body coordinate system, calibrated in advance.

The variance of the IMU pre-integration factor can be obtained iteratively from an initial value 0 using the method in [26].

4.2.3. Odometry Pre-Integration Factor

Generally, wheeled robots are equipped with wheel encoders that often provide noisy and only intermittently reliable measurements of the motion of each wheel. In order to reduce the amount of computation and accumulated error, by analogy to IMU we use the pre-integration technique on SE(2) to derive the constraints of the odometer. Unlike general ground mobile robots, TWIP swings back and forth in addition to plane motion, while the encoder only provides two-dimensional plane motion. In order to analyze the constraints of the encoder more conveniently, we add an auxiliary coordinate system, that is, a plane coordinate system. The coordinate system is fixed to the robot; its origin coincides with the origin of the robot body frame, the xy plane coincides with the physical plane, the z-axis is vertically upward, and the y axis coincides with the y axis of the robot body frame, as shown in Figure 6. The encoder measurements between two adjacent keyframes indicate the pose transformation between the plane coordinate systems of the corresponding moments. Therefore, we analyze the constraints provided by the encoder in the planar coordinate system.

The measurement data of the encoder can be converted into linear velocity and angular velocity on a two-dimensional plane:

$$\begin{aligned} v &= \tfrac{r_l\omega_l + r_r\omega_r}{2} \\ \omega &= \tfrac{r_l\omega_l - r_r\omega_r}{l} \end{aligned} \tag{34}$$

where ω_l and ω_r are the rotational velocities of the left and right wheels, respectively, r_l and r_r are their corresponding radii, and l denotes the robot's baseline.

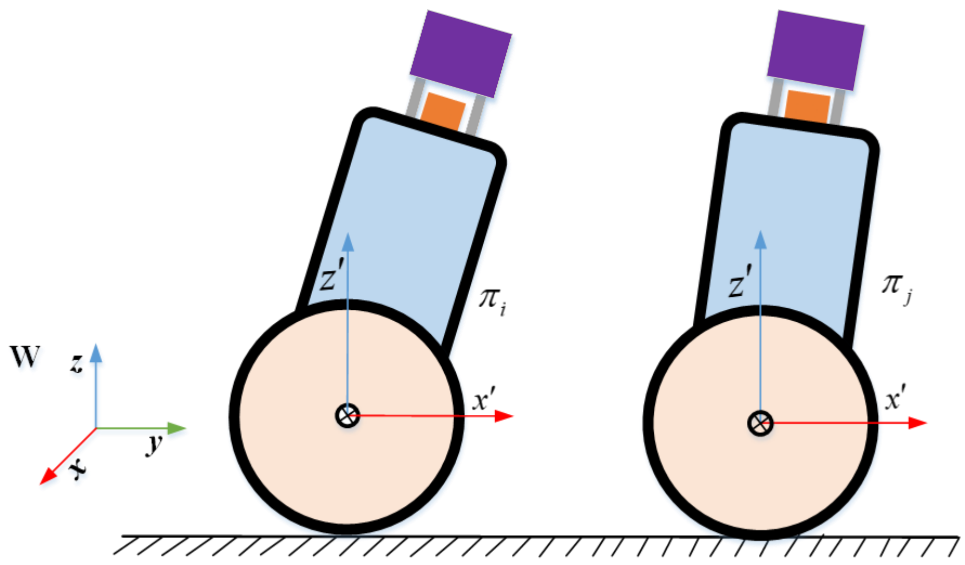

Figure 6. Motion of TWIP in Plane Coordinate System.

We model the encoder's measurement error with a Gaussian model to process the noisy odometer data. We begin by deriving the measurement model, assuming that between successive odometer readings the motion is planar; the encoder provides the following measurements in the planar coordinate system of the current keyframe:

$$\begin{aligned}\tilde{\mathbf{v}}_i &= \mathbf{v}_i + \boldsymbol{\varepsilon}_{vi} \\ \tilde{\omega}_i &= \omega_i + \varepsilon_{\omega i}\end{aligned} \tag{35}$$

where $\mathbf{v}_i = \begin{bmatrix} v_i \\ 0 \end{bmatrix}$, $\boldsymbol{\varepsilon}_{vi} = \begin{bmatrix} \varepsilon_{vi} \\ 0 \end{bmatrix}$; ε_{vi} and $\varepsilon_{\omega i}$ represent zero-mean Gaussian noise.

With two consecutive lidar keyframes, i and j, we obtain the pose of the plane coordinate system corresponding to the current keyframe in the world coordinate system from the pose of the previous keyframe and the measurements of the encoder:

$$\begin{aligned}\phi_{\pi_j}^{\mathrm{W}} &= \phi_{\pi_i}^{\mathrm{W}} + \sum_{k=i}^{j-1} (\tilde{\omega}_k - \varepsilon_{\omega k})\Delta t \\ \mathbf{P}_{\pi_j}^{\mathrm{W}} &= \mathbf{P}_{\pi_i}^{\mathrm{W}} + \sum_{k=i}^{j-1} \mathrm{R}\left(\phi_{\pi_k}^{\mathrm{W}}\right)((\tilde{\mathbf{v}}_k - \boldsymbol{\varepsilon}_{vk})\Delta t)\end{aligned} \tag{36}$$

where $\phi_{\pi_k}^{\mathrm{W}}, k = i, j$ is the yaw angle of the plane coordinate system corresponding to keyframe k relative to the world coordinate system and $\mathbf{P}_{\pi_k}^{\mathrm{W}} = \begin{bmatrix} x_k \\ y_k \end{bmatrix}, k = i, j$ is the 2D position of the plane coordinate system corresponding to keyframe k relative to the world coordinate system:

$$\mathrm{R}(\phi) = \exp\left(\phi^{\wedge}\right) = \exp\left(\begin{bmatrix} 0 & -\phi \\ \phi & 0 \end{bmatrix}\right) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \tag{37}$$

Note that the position propagated here depends on the rotational state. In order to generate a measurement between two keyframes i and j without depending on the robot's state in the world coordinate system, we place the items related to the measurement together and eliminate the items related to the state of *i* and *j*. Equation (36) can be written as:

$$\begin{aligned}\Delta\phi_{ij} &= \phi^{W}_{\pi_j}-\phi^{W}_{\pi_i}\\ &= \sum_{k=i}^{j-1}(\tilde{\omega}_i-\varepsilon_\omega)\Delta t\\ &= \sum_{k=i}^{j-1}\tilde{\omega}_i\Delta t-\sum_{k=i}^{j-1}\varepsilon_{\omega k}\Delta t\\ &= \Delta\tilde{\phi}_{ij}-\delta\phi_{ij}\end{aligned} \tag{38}$$

$$\begin{aligned}\Delta\mathbf{P}_{ij} &= \mathrm{R}\left(-\phi^{W}_{\pi_i}\right)\left(\mathbf{P}^{W}_{\pi_j}-\mathbf{P}^{W}_{\pi_i}\right)\\ &= \sum_{k=i}^{j-1}\mathrm{R}(\phi_{ik})((\tilde{\mathbf{v}}_k-\boldsymbol{\varepsilon}_{vk})\Delta t)\\ &\approx \sum_{k=i}^{j-1}\mathrm{R}(\tilde{\phi}_{ik})\exp(-\delta\phi_{ik})((\tilde{\mathbf{v}}_k-\boldsymbol{\varepsilon}_{vk})\Delta t)\\ &\approx \sum_{k=i}^{j-1}\mathrm{R}(\tilde{\phi}_{ik})\left(\mathbf{I}_{2\times 2}-\delta\phi^{\wedge}_{ik}\right)((\tilde{\mathbf{v}}_k-\boldsymbol{\varepsilon}_{vk})\Delta t)\\ &\approx \sum_{k=i}^{j-1}\mathrm{R}(\tilde{\phi}_{ik})\tilde{\mathbf{v}}_k\Delta t-\sum_{k=i}^{j-1}\mathrm{R}(\tilde{\phi}_{ik})\boldsymbol{\varepsilon}_{vk}\Delta t-\sum_{k=i}^{j-1}\mathrm{R}(\tilde{\phi}_{ik})1^{\wedge}\tilde{\mathbf{v}}_k\Delta t\delta\phi_{ik}\\ &= \Delta\tilde{\mathbf{P}}_{ij}-\delta\mathbf{P}_{ij}\end{aligned} \tag{39}$$

In order to obtain the error transfer model, we derive the integrated noise terms in the form of iterative transfer:

$$\begin{aligned}\delta\phi_{i,k+1} &= \delta\phi_{ik}+\varepsilon_{\omega k}\Delta t\\ \delta\mathbf{p}_{i,k+1} &= \delta\mathbf{p}_{ik}+\mathrm{R}(\tilde{\phi}_{ik})1^{\wedge}\tilde{\mathbf{v}}_k\Delta t\delta\phi_{ik}+\mathrm{R}(\tilde{\phi}_{ik})\Delta t\boldsymbol{\varepsilon}_{vk}\end{aligned} \tag{40}$$

Writing Equation (40) in a compact form, we then have:

$$\begin{bmatrix}\delta\mathbf{p}_{i,k+1}\\ \delta\phi_{i,k+1}\end{bmatrix}_{3\times 1} = \mathbf{A}_k\begin{bmatrix}\delta\mathbf{p}_{ik}\\ \delta\phi_{ik}\end{bmatrix}_{3\times 1}+\mathbf{B}_k\begin{bmatrix}\boldsymbol{\varepsilon}_{vk}\\ \varepsilon_{\omega k}\end{bmatrix}_{3\times 1} \tag{41}$$

where

$$\mathbf{A}_k = \begin{bmatrix}\mathbf{I}_{2\times 2} & \mathrm{R}(\phi_{ik})1^{\wedge}\tilde{\mathbf{v}}_k\Delta t\\ 0 & 1\end{bmatrix} \tag{42}$$

$$\mathbf{B}_k = \begin{bmatrix}\mathrm{R}(\phi_{ik})\Delta t & \mathbf{0}_{2\times 1}\\ \mathbf{0}_{1\times 2} & \Delta t\end{bmatrix} \tag{43}$$

Iterating from the initial value, $\boldsymbol{\Sigma}_{\delta i}=\mathbf{0}_{3\times 3}$, at the time *i*, we can incrementally calculate the pre-integrated terms $\Delta\phi_{ij}$ and $\Delta\mathbf{P}_{ij}$ and their covariance matrix, $\boldsymbol{\Sigma}_{\delta}=\begin{bmatrix}\boldsymbol{\Sigma}_{\delta\mathbf{P}_{ij}} & \mathbf{0}_{2\times 1}\\ \mathbf{0}_{1\times 2} & \Sigma_{\delta\phi_{ij}}\end{bmatrix}$.

Thus far, we have the encoder's pre-integration terms, $\Delta\tilde{\phi}_{ij}$ and $\Delta\tilde{\mathbf{P}}_{ij}$, which provide the measurements of the odometer constraints. Next, we formulate the residual of the odometer pre-integration constraint. Through the definition of the plane coordinate system, we can obtain the rotation of the plane coordinate system corresponding to the keyframes *i* and *j* relative to the world coordinate system, respectively:

$$\mathbf{R}^{W}_{\pi_i}=\begin{bmatrix}-\mathbf{e}_3\times\mathbf{R}^{W}_{B_i}\mathbf{e}_2 & \mathbf{R}^{W}_{B_i}\mathbf{e}_2 & \mathbf{e}_3\end{bmatrix} \tag{44}$$

$$\mathbf{R}^{W}_{\pi_j}=\begin{bmatrix}-\mathbf{e}_3\times\mathbf{R}^{W}_{B_j}\mathbf{e}_2 & \mathbf{R}^{W}_{B_j}\mathbf{e}_2 & \mathbf{e}_3\end{bmatrix} \tag{45}$$

where $\mathbf{e}_i$ is the *ith* column vector of the identity matrix, $\mathbf{I}_{3\times 3}$.

Then, the relative transformation of the plane coordinate system corresponding to keyframes i and j is as follows:

$$\begin{aligned}
\mathbf{R}_{\pi_i}^{\pi_j} &= \left(\mathbf{R}_{\pi_j}^{W}\right)^{T}\mathbf{R}_{\pi_i}^{W} \\
&= \begin{bmatrix} \left(-\mathbf{e}_3 \times \mathbf{R}_{B_j}^{W}\mathbf{e}_2\right)^{T} \\ \left(\mathbf{R}_{B_j}^{W}\mathbf{e}_2\right)^{T} \\ \mathbf{e}_3^{T} \end{bmatrix} \begin{bmatrix} -\mathbf{e}_3 \times \mathbf{R}_{B_i}^{W}\mathbf{e}_2 & \mathbf{R}_{B_i}^{W}\mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix} \\
&= \begin{bmatrix} -\mathbf{e}_2^{T}\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{e}_3^{\wedge}\mathbf{e}_3^{\wedge}\mathbf{R}_{B_i}^{W}\mathbf{e}_2 & \mathbf{e}_2^{T}\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{e}_3^{\wedge}\mathbf{R}_{B_i}^{W}\mathbf{e}_2 & 0 \\ -\mathbf{e}_2^{T}\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{e}_3^{\wedge}\mathbf{R}_{i}^{W}\mathbf{e}_2 & \mathbf{e}_2^{T}\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{R}_{B_i}^{W}\mathbf{e}_2 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{46}$$

As can be seen from Appendixes A and B, the measurement of the rotating part of the encoder $\Delta\tilde{\phi}_{ij}$ provides only one constraint, and the residual of the rotation part is:

$$\mathbf{r}_{\mathbf{R}} = \mathbf{e}_2^{T}\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{R}_{B_i}^{W}\mathbf{e}_2 - \cos\left(\Delta\tilde{\phi}_{ij}\right) \tag{47}$$

We obtain the variance of the rotation residuals using a first-order linear approximation of the trigonometric functions:

$$\cos\phi \approx \cos\left(\Delta\phi_{ij}\right) - \sin\left(\Delta\phi_{ij}\right)\left(\phi - \Delta\phi_{ij}\right) \tag{48}$$

$$\Sigma_{\mathbf{r}_{\mathbf{R}}} = \sin^{2}\left(\Delta\phi_{ij}\right)\Sigma_{\delta\phi_{ij}} \tag{49}$$

Then, its Jacobian matrix relative to the TWIP motion state (the pose of the robot body frame relative to the world frame) is

$$\mathbf{J}_{\phi_i}^{\mathbf{r}_{\mathbf{R}}} = -\mathbf{e}_2^{T}\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{R}_{B_i}^{W}\mathbf{e}_2^{\wedge} \tag{50}$$

$$\mathbf{J}_{\phi_j}^{\mathbf{r}_{\mathbf{R}}} = \mathbf{e}_2^{T}\left(\left(\mathbf{R}_{B_j}^{W}\right)^{T}\mathbf{R}_{B_i}^{W}\mathbf{e}_2\right)^{\wedge} \tag{51}$$

and the residual of the translation part is

$$\begin{aligned}
\mathbf{r}_{\mathbf{P}} &= \left[\left(\mathbf{R}_{\pi_i}^{W}\right)^{T}\left(\mathbf{P}_{B_j}^{W} - \mathbf{P}_{B_i}^{W}\right)\right]_{xy} - \Delta\tilde{\mathbf{P}}_{ij} \\
&= \begin{bmatrix} \left(-\mathbf{e}_3 \times \mathbf{R}_{B_i}^{W}\mathbf{e}_2\right)^{T}\left(\mathbf{P}_{B_j}^{W} - \mathbf{P}_{B_i}^{W}\right) \\ \left(\mathbf{R}_{B_i}^{W}\mathbf{e}_2\right)^{T}\left(\mathbf{P}_{B_j}^{W} - \mathbf{P}_{B_i}^{W}\right) \\ 0 \end{bmatrix}_{xy} - \Delta\tilde{\mathbf{P}}_{ij}
\end{aligned} \tag{52}$$

where $[]_{xy}$ represents the first two rows of the vector, and its covariance is

$$\boldsymbol{\Sigma}_{\mathbf{r}_{\mathbf{P}}} = \boldsymbol{\Sigma}_{\delta\mathbf{P}_{ij}} \tag{53}$$

Then, its Jacobian matrix relative to the motion state of TWIP is

$$\mathbf{J}_{\phi_i}^{\mathbf{r}_{\mathbf{P}}} = \begin{bmatrix} \mathbf{e}_2^{T}\left(\left(\mathbf{R}_{B_i}^{W}\right)^{T}\mathbf{e}_3^{\wedge}\left(\mathbf{P}_{B_j}^{w} - \mathbf{P}_{B_i}^{w}\right)\right)^{\wedge} \\ \mathbf{e}_2^{T}\left(\left(\mathbf{R}_{B_i}^{W}\right)^{T}\left(\mathbf{P}_{B_j}^{w} - \mathbf{P}_{B_i}^{w}\right)\right)^{\wedge} \end{bmatrix} \tag{54}$$

$$\mathbf{J}^{\mathbf{r_P}}_{\mathbf{P}^{W}_{B_i}} = \begin{bmatrix} -\mathbf{e}_2^{T}\left(\mathbf{R}^{W}_{B_i}\right)^{T}\mathbf{e}_3^{\wedge} \\ -\mathbf{e}_2^{T}\left(\mathbf{R}^{W}_{B_i}\right)^{T} \end{bmatrix} \tag{55}$$

$$\mathbf{J}^{\mathbf{r_P}}_{\mathbf{P}^{W}_{B_j}} = \begin{bmatrix} \mathbf{e}_2^{T}\left(\mathbf{R}^{W}_{B_i}\right)^{T}\mathbf{e}_3^{\wedge} \\ \mathbf{e}_2^{T}\left(\mathbf{R}^{W}_{B_i}\right)^{T} \end{bmatrix} \tag{56}$$

4.2.4. Ground Constraint Factor

This paper proposes a new nonholonomic constraint factor in order to add ground constraints. We assume that TWIP moves on an approximately flat surface and that the robot is always constrained by the ground plane during movement. Ideally, the z and pitch angle would remain constant during the planar motion for a two-wheeled inverted pendulum robot. In the natural environment, however, there are motion disturbances in the other dimensions of the robot's pose due to rough terrain and shaking in the robot's motion. In order to reduce the error caused by this motion disturbance, we use a random model to introduce ground constraints and assume that the motion disturbance is a zero-mean Gaussian model. This is more in line with the natural environment, allowing a least squares to be used to model the ground constraints, which naturally adds the ground constraints as a new factor to the factor graph. As shown in Figure 7, when TWIP moves on an ideal plane the roll angle and z of its body frame relative to the world frame should both be zero. Therefore, the residual of the ground constraint is defined as follows:

$$r_{\mathbf{R}} = \mathbf{e}_3^{T}\mathbf{R}^{W}_{B_i}\mathbf{e}_2 \tag{57}$$

$$r_{\mathbf{P}} = \mathbf{e}_3^{T} \cdot \mathbf{P}^{W}_{B_i} \tag{58}$$

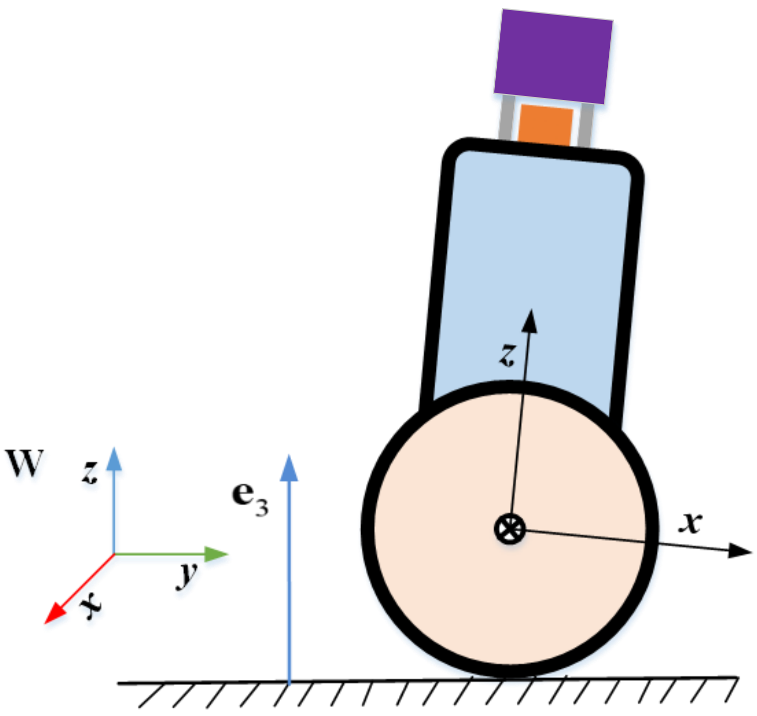

Figure 7. Ground constraints when TWIP is moving on approximately flat ground.

The corresponding covariance matrix is

$$\mathbf{\Sigma}_{\mathbf{r}_p} = diag\left(\ \sigma_r^2 \quad \sigma_z^2\ \right) \tag{59}$$

In this paper, we assume that the motion disturbance of the robot due to rough terrains or motion shaking obeys the Gaussian distribution; σ_z and σ_r are the noise sigma of the z and roll, respectively, which are determined by the conditions of the terrain and the robot's structure.

The Jacobian matrix relative to the state of TWIP is

$$\mathbf{J}_{\phi_i}^{\mathbf{r_R}} = -\mathbf{e}_3^{\mathrm{T}} \mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}} \mathbf{e}_2^{\wedge} \tag{60}$$

$$\mathbf{J}_{\mathbf{P}_{\mathrm{B}_i}^{\mathrm{W}}}^{\mathbf{r_P}} = \mathbf{e}_3^{\mathrm{T}} \tag{61}$$

4.2.5. Loop Closure Factor

When a new lidar keyframe, F_{i+1}, arrives, its corresponding robot state, $\boldsymbol{K}_{i+1}$, is added to the factor graph. We obtain the absolute pose of the current lidar keyframe as in Section 4.2.1. We obtain the absolute pose of the current lidar keyframe as in Section 4.2.1, and use this pose as a prior to find the previous keyframe closest to F_{i+1} in Euclidean space as a candidate frame. For example, the lidar keyframe F_3 in Figure 3 is one of the returned loop closure candidates. Then, taking the position of the lidar keyframe F_3 as the center, we extract the point cloud in the cuboid with a length of 10 m and a height of 5 m in the global map and transform the point cloud into the robot body coordinate system corresponding to the lidar keyframe F_3 to construct a local map M_{i+1}. We try to match F_{i+1} to the local map M_{i+1} using the scan-matching method discussed in Section 4.2.1. In the local map, we select the N nearest feature points for each feature point of the current frame to fit a line or plane. After optimization, we obtain the relative transformation, $\Delta\tilde{\boldsymbol{T}}_3^{i+1}$, and add it to the graph as a loop closure factor. Throughout this paper, we choose to set the loop closure's search distance to 15 m from the new lidar keyframe.

5. Experiment

We conducted experiments on our collected datasets, as we could not find an open dataset to provide calibrated Lidar–IMU–Odometry for a two-wheeled inverted pendulum robot. All experiments were carried out with an Intel CPU i7-8559U (8 cores@2.70 GHz) laptop computer with 8 GB RAM. As shown in Figure 8, the platform was equipped with a Velodyne VLP-16 lidar oprating at 10 HZ, an XSENS MTI-G-710 IMU to provide acceleration and angular velocity at 200 HZ, and an encoder with 1khz, all of which were synchronized and calibrated in advance.

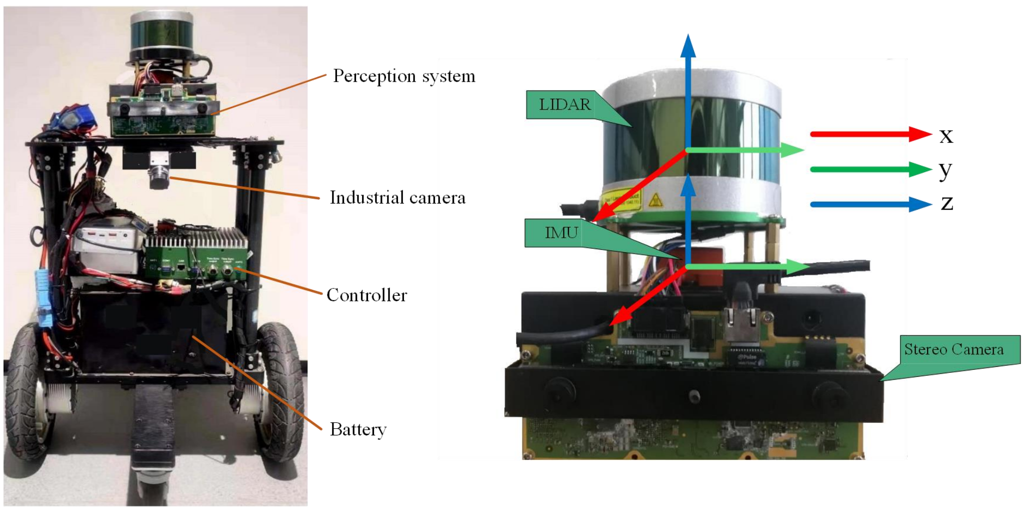

Figure 8. Experimental platform: Two-Wheeled Inverted Pendulum (TWIP) robot (**left**) and environment perception system (**right**).

The dataset comes from two scenes; one is a long straight corridor scene, named Dataset Corridor, and the other is a flat outdoor road, named Dataset Outdoor. The environment of the two datasets is shown in Figure 9. As with the process of data collection, the actual ground truth of the robot is unavailable. Inspired by [25], we equipped TWIP with downward-viewing cameras on the front and rear (their position is shown in Figure 7) and an attitude sensor with a dynamic accuracy of 0.2°. Tens of Aruco code markers were laid out on the floor, and their locations in a predefined world reference were measured with a location error of less than 1 cm. Transformations between industrial cameras and attitude sensors and the robot were all pre-calibrated. The two industrial cameras detected and recorded these marks during the robot's movement. In order not to affect the algorithm in this paper as well as to improve the true accuracy, the trajectory truth was obtained offline. We extracted the marked corner points in each recorded image and approximated the trajectory truth by optimizing the reprojection error of the image and the pose constraints of the pose sensor in an offline batch. In order to verify the algorithm used in this article, we tested the constructed SLAM algorithm and the state-of-the-art lidar slam method (Fast_lio2 [16], Lio_sam [18], Loam [7]) on the same dataset.

Figure 9. Experimental environment: indoor straight corridor (**left**) and outdoor structured environment (**right**).

5.1. Experimental Results and Analysis of Indoor Corridors

Figure 10 shows the results of our algorithm for localization and mapping with Dataset Corridor. Figure 11 compares trajectories in the xy direction recovered by our algorithm, Fast_lio2, Lio_sam, Loam, Odometry, and ground truth on the Long Straight Corridor dataset. The odometer is a trajectory obtained using the less accurate odometer raw data calculated directly from the encoder to emphasize the correction effect of the lidar method on the odometer's accumulated error. Meanwhile, to compare the long-term positioning accuracy of the algorithms, all algorithms did not add a loop detection module, and the pose estimation of all algorithms was performed in the robot body frame. Figures 12 and 13 show the corresponding trajectory errors relative to the ground truth, respectively. It can be seen from Figure 12 that the x-direction is the running direction of TWIP, which has fewer constraints, and these algorithms have more significant errors in the x-direction. Among them, the accuracy of loam in the x-direction is significantly lower than the other algorithms due to the lack of measurements from other sensors. There is a constraint from the wall in the y-direction; all the lidar-based SLAM algorithms have little difference in accuracy, as they all use the same matching method. While the results by our method exhibit much better accuracy, there is a smaller upper limit of errors in the x and y directions. Attention should be paid to the z-axis values in Figure 14, where the z-coordinates estimated by our method are well-bounded around zero (which is consistent with natural indoor environments) while the other algorithms have significantly larger upper bounds. Using a nonholonomic constraint factor and adding planar constraints and encoder pre-integration constraints improves indoor pose estimation accuracy, especially in the z-direction. The numerical results of the estimation errors are presented in Table 2. The root mean square of the errors (RMSE) is used here, including those of the errors in principal parameter directions, namely, the x, y, pitch, and yaw coordinates. In terms of RMSE, the error of our algorithm is lower than the other algorithms in all directions for a two-wheeled inverted pendulum robot.

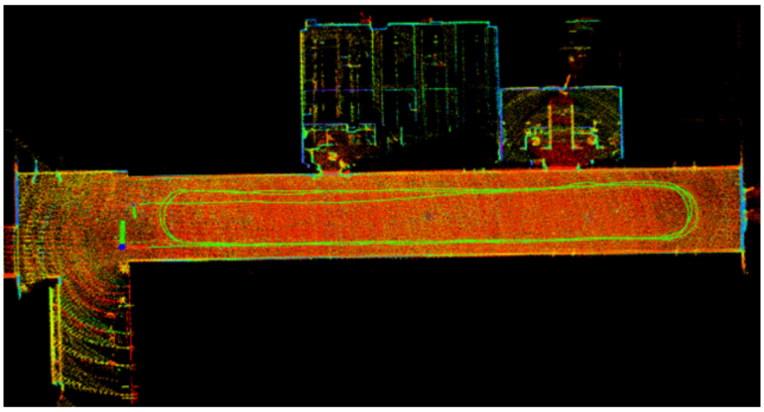

Figure 10. Localization and mapping results of our algorithm on Dataset Corridor.

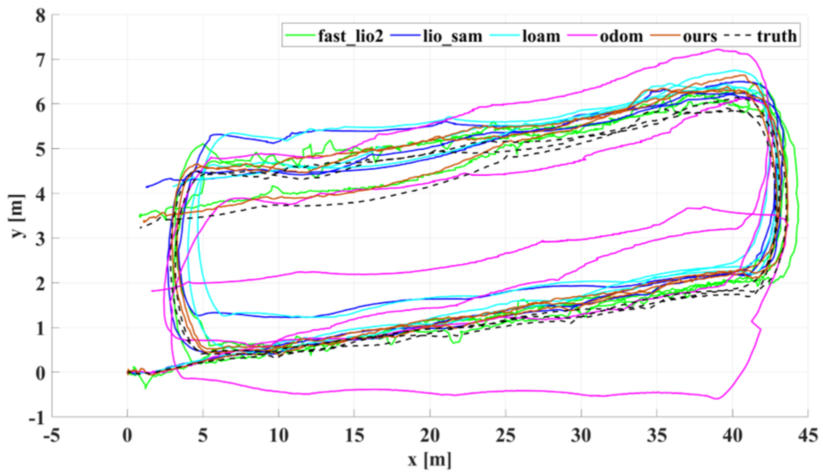

Figure 11. Dataset Corridor evaluation: projections of trajectories estimated by different algorithms on the xy plane.

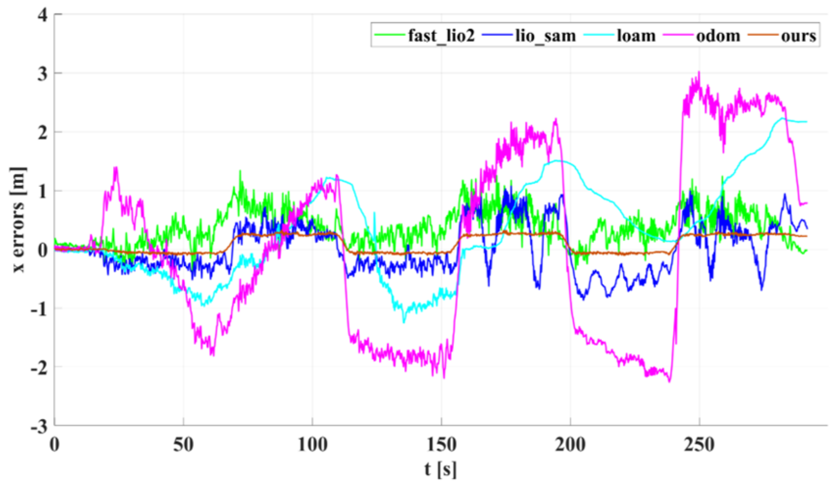

Figure 12. The x-axis estimation errors of different algorithms on Dataset Corridor.

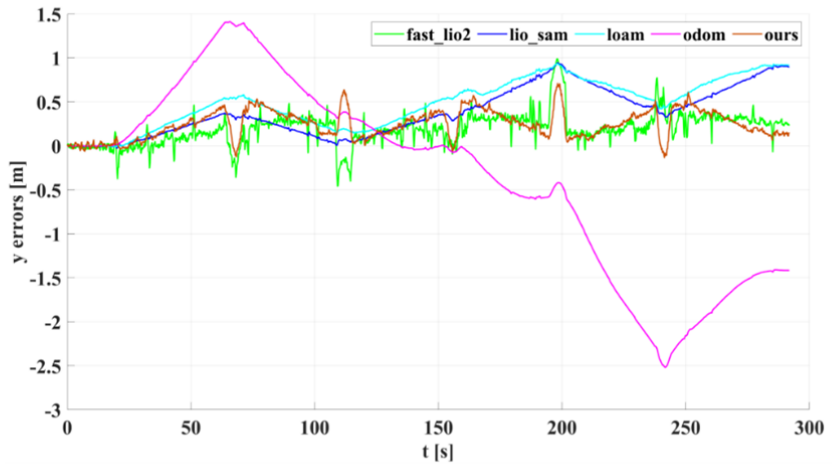

Figure 13. The *y*-axis estimation errors of different algorithms on Dataset Corridor.

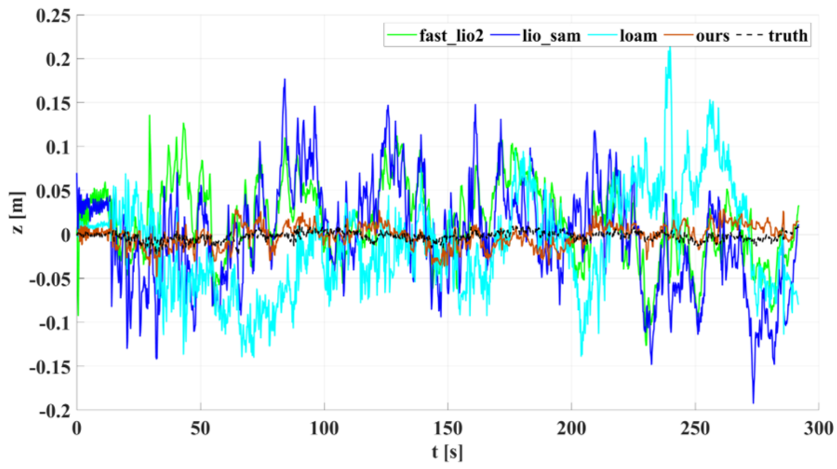

Figure 14. Dataset Corridor evaluation: projections of trajectories estimated by different algorithms on the *z*-axis.

Table 2. The estimation errors (RMSE) of different methods on Dataset Corridor.

Dimension	Fast_lio2	Lio_sam	Loam	Odom	Ours
$x(m)$	0.4680	0.4111	0.9218	1.5468	0.1747
$y(m)$	0.2636	0.4695	0.5550	1.1035	0.2921
$pitch(rad)$	0.0287	0.0420	0.0590	–	0.0176
$yaw(rad)$	0.1740	0.1810	0.3170	0.6010	0.0970

5.2. Experimental Results and Analysis of Outdoor Structured Environment

In our outdoor experiments, we used algorithms such as lio_sam for comparison with our algorithm. Figure 15 shows our algorithm's trajectory and environment map on Dataset Outdoor. Figure 16 compares estimated trajectories in the *x* and *y* directions on Dataset Outdoor, including our algorithm, fast_lio2, lio_sam, loam, odometry, and ground truth. Figures 17 and 18 are the corresponding errors relative to the ground truth, respectively. Figure 19 shows the trajectory in the *z*-direction, which should be constrained around zero. As can be seen from the results, our algorithm continues to outperform the other algorithms in outdoor scenes.

Moreover, the accuracy improvement of our algorithm is more significant in the *z*-axis direction than in the *x*- and *y*-axis directions. There is no degradation scene in the outdoor experiment. The lidar feature points mainly come from the reflection of surrounding objects, which constrains the robot's pose in the horizontal direction. In contrast, lidar provides relatively few feature points in the vertical direction due to the limitation of the vertical field of view. Our algorithm introduces ground constraints, which increase the pose constraints of the robot in the vertical direction and improve the localization accuracy of the system, especially in the z-direction. Meanwhile, we quantitatively analyzed the performance of our algorithm and other algorithms in the direction of the principal parameters on the outdoor dataset in terms of RMSE. The results are shown in Table 3, indicating that the error of our algorithm in the direction of principal parameters is lower than that of the other algorithms.

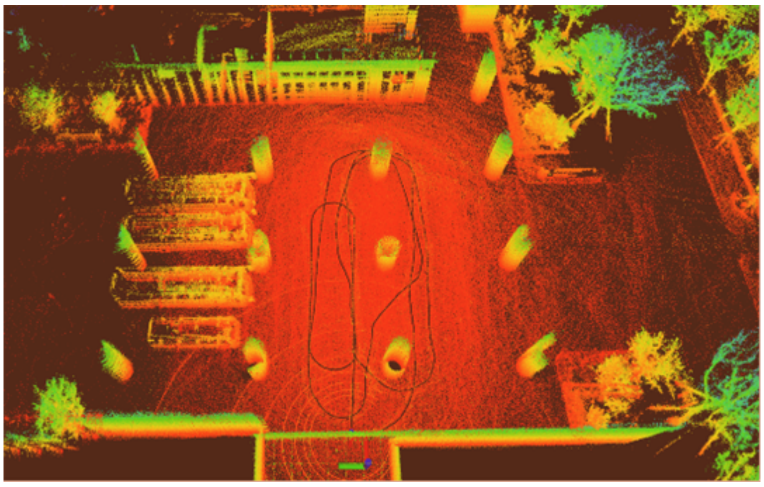

Figure 15. Localization and mapping results of our algorithm on Dataset Outdoor.

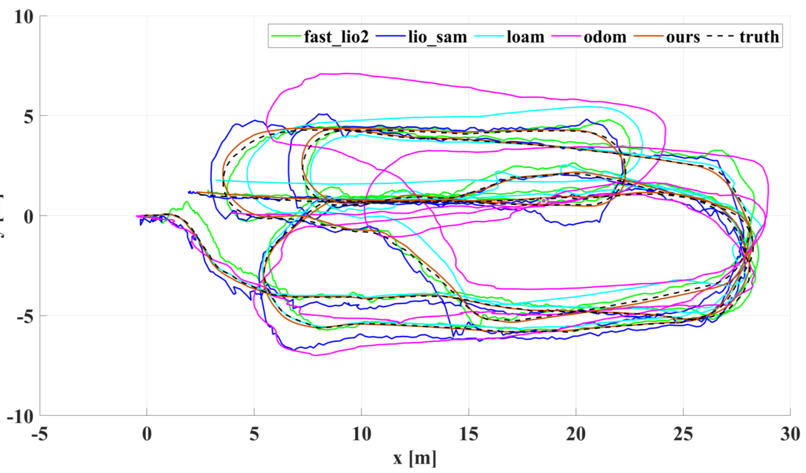

Figure 16. Dataset Outdoor evaluation: projections of trajectories estimated by different algorithms on the xy plane.

Table 3. The estimation errors statistics (RMSE) of different methods on Dataset Outdoor.

Dimension	Fast_lio2	Lio_sam	Loam	Odom	Ours
$x(m)$	0.3113	0.3486	0.4770	1.6786	0.0635
$y(m)$	0.4549	0.4213	0.4999	1.2751	0.1625
$pitch(rad)$	0.0220	0.0281	0.0282	–	0.0172
$yaw(rad)$	0.2990	0.4900	0.6170	0.9368	0.1803

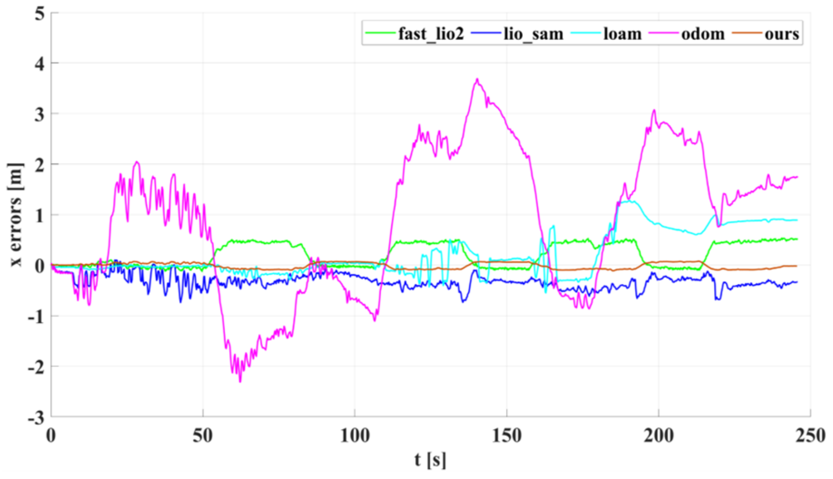

Figure 17. The x-axis estimation errors of different algorithms on Dataset Outdoor.

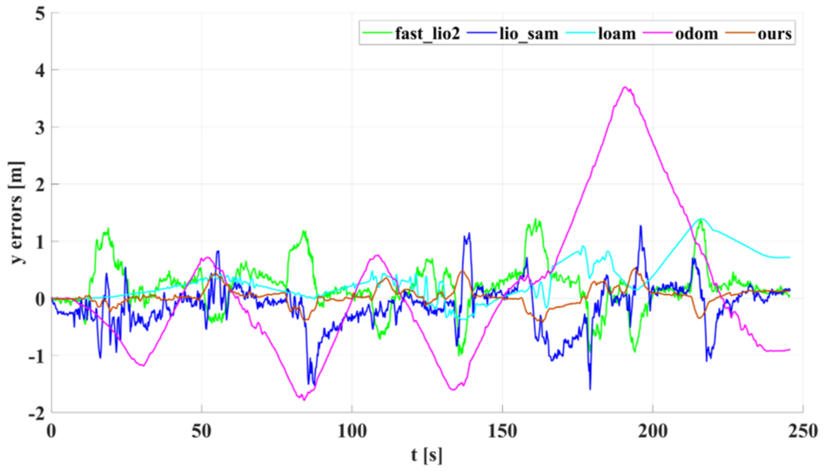

Figure 18. The y-axis estimation errors of different algorithms on Dataset Outdoor.

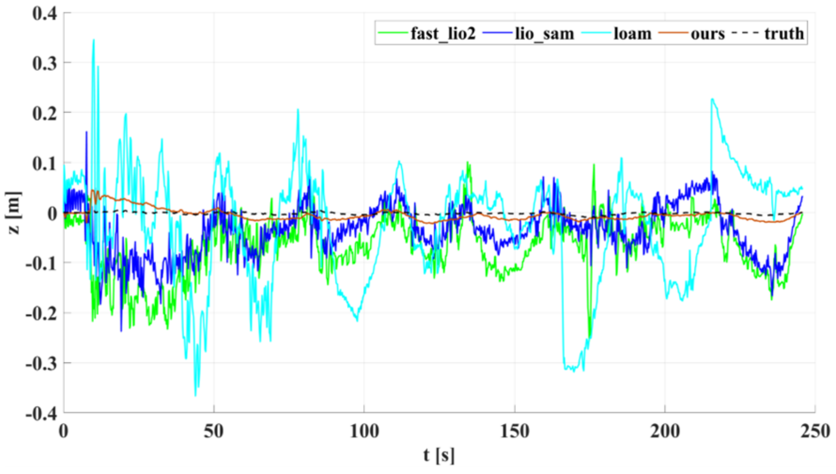

Figure 19. Dataset Outdoor evaluation: projections of trajectories estimated by different algorithms on the z-axis.

Our indoor and outdoor experiments show that the non-full constraint factor proposed by our algorithm naturally introduces the pre-integration constraint of the encoder while adding a ground constraint in the form of the random constraint model, which is more in line with the actual motion of the two-wheeled inverted pendulum robot, resulting in the improved positioning and mapping accuracy of the system.

6. Conclusions

This paper proposes a novel Lidar–IMU–Odometer coupled framework for localization and mapping for a two-wheeled inverted pendulum robot on approximately flat ground. We formulate a factor graph incorporating relative and absolute measurements from different sensors (including ground constraints) into the system as factors. In this coupled framework using a lidar–IMU–encoder system based on a factor graph, we introduce five types of factors and analyze the constraint dimension of each type of factor on the robot state node according to the motion characteristics of TWIP. For the non-fully constrained factors (the encoder pre-integration factor and ground constraint factor), we propose a new factor with constraints in order to add them naturally to the factor graph with the robot state node on SE(3). Experiments in indoor and outdoor environments validate the superior performance of our system compared to other state-of-the-art lidar-based SLAM algorithms.

This proposal paves the way for using constrained models to consider motion perturbations for specially-structured ground robot state estimation with different sensors. In the future, based on the work inof this paper we intend to realize the positioning and mapping of TWIP on undulating roads via the proposed LiDAR–IMU–Encoder coupling framework

Author Contributions: Conceptualization, Y.Z.; Data curation, Y.Z.; Formal analysis, Y.Z.; Funding acquisition, S.Z.; Investigation, Y.Z.; Methodology, Y.Z.; Project administration, S.Z.; Software, Y.Z.; Supervision, S.Z.; Validation, Y.Z.; Writing—original draft, Y.Z.; Writing—review & editing, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of Heilongjiang Province of China (YQ2021F011).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

This section proves the relationship between the item in row 0, column 0,r_{00} and the item in row 1, column 1,r_{11} in the matrix finally obtained in Equation (43). From Equations (7) and (8) we obtain

$$\begin{aligned}
-\mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}\mathbf{e}_3^{\wedge}\mathbf{e}_3^{\wedge}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2 &= -\mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}(\mathbf{e}_3\mathbf{e}_3^{\mathrm{T}}-\mathbf{I}_{3\times3})\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2 \\
&= \mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2-\mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}\mathbf{e}_3\mathbf{e}_3^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2 \\
&= \mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2-\left(\mathbf{e}_3^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{e}_2\right)^{\mathrm{T}}\left(\mathbf{e}_3^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2\right) \\
&= \mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2
\end{aligned} \tag{A1}$$

Therefore, we can see that the item r_{00} and the item r_{11} in the matrix are equal.

Appendix B

This section proves the constraints implicit in the final matrix obtained in Formula (43). When TWIP moves on the ideal ground plane, the x-y plane of the plane coordinate system corresponding to each keyframe is the same plane. Therefore, the y-axis of the plane frame corresponding to keyframe j is in the x-y plane of the plane coordinate system corresponding to keyframe i. As shown in Figure A1, we translate the y-axis y_j' of the plane coordinate system π_j to the plane coordinate system π_i, and all coordinate axes are represented in the world coordinate system. We assume that the angle between y_j' and y_i' is α, that the angle between y_j' and x_i' is β, and that $\alpha+\beta=90^\circ$. Therefore, we have

$$\begin{aligned}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{e}_2\right)^{\mathrm{T}}\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2 &= \mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{e}_2 \cdot \mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2 \\ &= \left\|\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{e}_2\right\|_2\left\|\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2\right\|_2 \cos\alpha \\ &= \cos\alpha\end{aligned} \tag{A2}$$

$$\begin{aligned}-\mathbf{e}_2^{\mathrm{T}}\left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\right)^{\mathrm{T}}\mathbf{e}_3^{\wedge}\mathbf{R}_i^{\mathrm{W}}\mathbf{e}_2 &= \left(\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{e}_2\right)\cdot\left(-\mathbf{e}_3\times\mathbf{R}_i^{\mathrm{W}}\mathbf{e}_2\right) \\ &= \left\|\mathbf{R}_{\mathrm{B}_j}^{\mathrm{W}}\mathbf{e}_2\right\|_2\left\|-\mathbf{e}_3\times\mathbf{R}_i^{\mathrm{W}}\mathbf{e}_2\right\|_2\cos\beta \\ &= \cos\beta \\ &= \sin\alpha\end{aligned} \tag{A3}$$

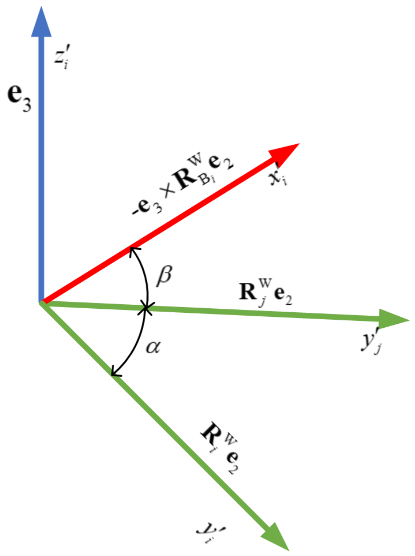
$\mathbf{e}_3$
z_i'
$-\mathbf{e}_3\times\mathbf{R}_{\mathrm{B}_i}^{\mathrm{W}}\mathbf{e}_2$
x_i
β
$\mathbf{R}_j^{\mathrm{W}}\mathbf{e}_2$
y_j'
α
$\mathbf{R}_i^{\mathrm{W}}\mathbf{e}_2$
y_i'

Figure A1. Geometric relationship between two continuous plane coordinate systems π_i and π_j.

References

1. Durrant-Whyte, H.; Bailey, T. Simultaneous localization and mapping: Part I. *IEEE Robot. Autom. Mag.* **2006**, *13*, 99–110. [CrossRef]
2. Bailey, T.; Durrant-Whyte, H. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robot. Autom. Mag.* **2006**, *13*, 108–117. [CrossRef]
3. Bouman, A.; Ginting, M.F.; Alatur, N.; Palieri, M.; Fan, D.D.; Touma, T.; Kim, S.-K.; Otsu, K.; Burdick, J.; Agha-Mohammadi, A.A. Autonomous Spot: Long-Range Autonomous Exploration of Extreme Environments with Legged Locomotion. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 2518–2525. [CrossRef]
4. Mur-Artal, R.; Montiel, M.M.J.; Tardós, D.J. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
5. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
6. Sibley, G.; Matthies, L.; Sukhatme, G. Sliding window filter with application to planetary landing. *J. Field Robot.* **2010**, *27*, 587–608. [CrossRef]
7. Zhang, J.; Singh, S. LOAM: Lidar Odometry and Mapping in Real-time. *Robot. Sci. Syst.* **2014**, *2*, 1–9. [CrossRef]
8. Shan, T.; Englot, B. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4758–4765. [CrossRef]
9. Wang, H.; Wang, C.; Chen, C.-L.; Xie, L. F-LOAM: Fast LiDAR Odometry and Mapping. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 4390–4396. [CrossRef]

10. 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 Robot. Autom. Lett.* **2021**, *6*, 421–428. [CrossRef]
11. Santamaria-navarro, A.; Thakker, R.; Fan, D.D.; Morrell, B.; Agha-mohammadi, A.A. Towards resilient autonomous navigation of drones. In *International Symposium on Robotics Research*; Springer: Cham, Switzerland, 2019.
12. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
13. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
14. Lynen, S.; Achtelik, M.W.; Weiss, S.; Chli, M.; Siegwart, R. A robust and modular multi-sensor fusion approach applied to MAV navigation. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 3923–3929. [CrossRef]
15. Qin, C.; Ye, H.; Pranata, C.E.; Han, J.; Zhang, S.; Liu, M. LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 8899–8906. [CrossRef]
16. Xu, W.; Cai, Y.; He, D.; Lin, J.; Zhang, F. FAST-LIO2: Fast Direct LiDAR-Inertial Odometry. *IEEE Trans. Robot.* **2022**. [CrossRef]
17. Ye, H.; Chen, Y.; Liu, M. Tightly Coupled 3D Lidar Inertial Odometry and Mapping. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 3144–3150. [CrossRef]
18. Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Ratti, C.; Rus, D. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 5135–5142. [CrossRef]
19. Wu, K.J.; Guo, C.X.; Georgiou, G.; Roumeliotis, S.I. VINS on wheels. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 5155–5162. [CrossRef]
20. Lategahn, H.; Geiger, A.; Kitt, B. Visual SLAM for autonomous ground vehicles. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1732–1737. [CrossRef]
21. Ortín,D.; Montiel, J.M.M. Indoor robot motion based on monocular images. *Robotica* **2001**, *19*, 331–342. 3143. [CrossRef]
22. Konolige, K.; Grisetti, G.; Kümmerle, R.; Burgard, W.; Limketkai, B.; Vincent, R. Efficient Sparse Pose Adjustment for 2D mapping. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 22–29. [CrossRef]
23. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-time loop closure in 2D LIDAR SLAM. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1271–1278. [CrossRef]
24. Zheng, F.; Tang, H.; Liu, Y.-H. Odometry-Vision-Based Ground Vehicle Motion Estimation with SE(2)-Constrained SE(3) Poses. *IEEE Trans. Cybern.* **2018**, *49*, 2652–2663. [CrossRef] [PubMed]
25. Zheng, F.; Liu, Y.-H. Visual-Odometric Localization and Mapping for Ground Vehicles Using SE(2)-XYZ Constraints. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; 3556–3562. [CrossRef]
26. Forster, C.; Carlone, L.; Dellaert, F.; Scaramuzza, D. On-Manifold Preintegration for Real-Time Visual–Inertial Odometry. *IEEE Trans. Robot.* **2017**, *33*, 1–21. [CrossRef]