Sensor Fusion for Localization of Automated Guided Vehicles

Onur Sari

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 13.8.2020

Supervisor

Prof. Ville Kyrki

Advisor

MSc Usama Tariq

Aalto University
School of Electrical Engineering

Aalto University
School of Electrical Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author	Onur Sari
Title	Sensor Fusion for Localization of Automated Guided Vehicles
Degree programme	Automation and Electrical Engineering
Major	Control, Robotics and Autonomous Systems
Code of major	ELEC-3025
Supervisor	Prof. Ville Kyrki
Advisor	MSc Usama Tariq
Date	13.8.2020
Number of pages	92
Language	English

Abstract

Automated Guided Vehicles (AGVs) need to localize themselves reliably in order to perform their tasks efficiently. To that end, they rely on noisy sensor measurements that potentially provide erroneous location estimates if they are used directly. To prevent this issue, measurements from different kinds of sensors are generally used together. This thesis presents a Kalman Filter based sensor fusion approach that is able to function with asynchronous measurements from laser scanners, odometry and Inertial Measurement Units (IMUs). The method uses general kinematic equations for state prediction that work with any type of vehicle kinematics, and utilizes state augmentation to estimate gyroscope and accelerometer biases.

The developed algorithm was tested with an open source multisensor navigation dataset and real-time experiments with an AGV. In both sets of experiments, scenarios in which the laser scanner was fully available, partially available or not available were compared. It was found that using sensor fusion resulted in a smaller deviation from the actual trajectory compared to using only a laser scanner. Furthermore, in each experiment, using sensor fusion decreased the localization error in the time periods where the laser was unavailable, although the amount of improvement depended on the duration of unavailability and motion characteristics.

Keywords Sensor fusion, Kalman Filter, Automated Guided Vehicle, Navigation

Preface

This research was conducted for Navitec Systems, and I would like to thank all my colleagues there for their continuous support throughout my work. I am grateful to my supervisor Professor Ville Kyrki and my advisor Usama Tariq who were always ready to help whenever I needed.

I would especially like to thank my parents Yasemin and Omer, my brother Alp and my friends from all around the world for their love and support before, during and after this work. It is thanks to them that I had the strength to overcome challenges along my path, and that I enjoyed the journey as much as the destination.

Otaniemi, 13.8.2020

Onur Sari

Contents

Symbols and abbreviations

Symbols

$\mathbf{a_x}, \mathbf{a_y}, \mathbf{a_z}$	Linear acceleration in three dimensions
c	Speed of light in vacuum $\approx 3 \times 10^8$ [m/s]
$\mathbf{F}$	State transition matrix
$\mathbf{G}$	Input transition matrix
$\mathbf{H}$	Observation matrix
$\mathbf{P}$	State covariance matrix
$\mathbf{p_x}, \mathbf{p_y}, \mathbf{p_z}$	Position in three dimensions
$\mathbf{Q}$	Process noise covariance matrix
$\mathbf{R}$	Measurement noise covariance matrix
$\mathbf{v_x}, \mathbf{v_y}, \mathbf{v_z}$	Linear velocity in three dimensions
$\delta\mathbf{b}$	Measurement bias
θ	Pitch
ϕ	Yaw
ψ	Roll
$\omega_\mathbf{x}, \omega_\mathbf{y}, \omega_\mathbf{z}$	Angular velocity in three dimensions

Abbreviations

AGV	Automated Guided Vehicle
ATE	Absolute Trajectory Error
EKF	Extended Kalman Filter
GPS	Global Positioning System
ICC	Instantaneous Center of Curvature
ICP	Iterative Closest Point
LiDAR	Light Detection and Ranging
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
RPE	Relative Pose Error
SLAM	Simultaneous Localization and Mapping
UKF	Unscented Kalman Filter
ZARU	Zero Angular Rotation Update
ZUPT	Zero Velocity Update

1 Introduction

In recent years, Automated Guided Vehicles (AGVs) have gained widespread usage in many industries due to their ability to efficiently perform various tasks that increase productivity and reduce associated costs. These tasks include material transportation, loading and unloading, as well as product handling for warehouses, factories, paper industry, hospitals and power plants. In order to perform their tasks efficiently, it is important to reliably determine the position and orientation of AGVs in their operational environment. This process is referred to as *localization*, which has been a hot topic of research for many years, leading to improvement of many established and proven techniques such as landmark based localization [1], as well as development of novel ones utilizing neural networks [2], reinforcement learning [3] and computer vision [4].

AGV localization methods make use of different information sources to determine the position of the vehicle. A common method used in most localization solutions is odometry [5][6], also referred to as dead reckoning [7]. Odometry calculates the distance travelled by the vehicle from a known starting position to estimate current location. By using encoders on the wheels and the vehicle kinematic structure, the speed of the vehicle can be calculated based on wheel movements and later integrated to obtain the relative position. While this method can provide accurate information for short time periods, even small errors become integrated and accumulate over time due to the lack of absolute position measurements. This can eventually cause a large drift, which in turn results in greater positioning errors in longer time periods.

As the majority of AGV tasks require robust localization for long time periods, additional sensors are needed to correct the odometry errors. One of the most commonly used sensors for correcting such errors is the *laser range finder*, often called a laser scanner. With a laser scanner, the AGV can create a map of an unknown environment by extracting features, such as walls, corners, trees or tables, and then locate itself in the map by measuring the distance to those features [8][9]. This method provides an absolute position and orientation measurement in the map, which can be used to cancel odometry drift [10].

In addition to odometry and laser scanners, many other types of sensors are used for AGV localization. For example, inertial measurements obtained from an Inertial Measurement Unit (IMU) can be used to enhance odometry for more accurate orientation and position estimates, or Global Positioning System (GPS) measurements can provide global position estimations.

However, most localization techniques need to utilize more than one source of information. While some examples of successful localization have been developed based on a single sensor [11], it is generally beneficial to combine data from multiple sensors for situations when one of the sensors is unavailable or unreliable. Merging the information from two or more sensors is called *sensor fusion* [12]. Various algorithms have been developed to robustly fuse information from multiple sensors, with the most common techniques for sensor fusion being the Kalman Filter and the Particle Filter [13].

Since both Kalman and Particle filters are recursive Bayes estimators [14], they attempt to estimate the state of a system in a certain time step using the state in the previous time step and incoming measurements. The main difference between these approaches is that the Kalman Filter works as an optimal estimator based on the assumption that the system is linear and has Gaussian noise, while the Particle Filter is not bound to such restrictions, though it is computationally expensive [15]. To maintain the advantages of the Kalman Filter in nonlinear systems, several advanced algorithms have been developed, including the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF). Both have been successfully used for sensor fusion in nonlinear systems [16][17]. Of these options, the best method depends on the system conditions, available computational power, sensor types and desired performance metrics.

Although many sensor fusion algorithms have successfully been developed based on both Kalman and Particle Filters, the majority of these are limited to a specific selection of sensors and AGV type. Moreover, most of the algorithms rely on periodic sensor measurements that require readings to arrive at certain time intervals, and few studies have focused on asynchronous measurements that can arrive at any time.

The aim of this thesis is to develop a working prototype of a sensor fusion based localization method for an AGV in order to provide reliable and accurate estimates of position and orientation using asynchronous measurements. To develop this solution, the thesis will utilize a Kalman Filter based approach that is independent from the vehicle kinematic structure and able to work with any combination of laser scanner, odometry and IMU measurements. Furthermore, the non-zero mean bias present in the IMU measurements will be considered and the Kalman Filter will utilize bias estimation to deal with them. The viability of the developed solution will be tested by using an open source multisensor navigation dataset and verifying this solution with real-time experiments on an AGV. Particle Filter based solutions will not be considered in this thesis, since Gaussian noise characteristics are assumed for the system, making the Kalman Filter approach feasible. This assumption would not hold if the initial position of the AGV was unknown. However, it can be reliably determined in a known map using existing methods [18].

This thesis is divided into six chapters. Chapter 2 presents an overview of laser scanners, odometry and IMUs, how they are used in localization, as well as their output and noise characteristics. Chapter 3 explains the Kalman Filter and Extended Kalman Filter methods. The developed localization system is presented in detail in Chapter 4, while Chapter 5 analyzes the performance of the implementation using the results from the experiments. Finally, conclusions drawn from the experiments and possible ideas about further development are discussed in Chapter 6.

2 Sensors

This chapter presents the sensors most commonly used for localization of an AGV, including their output types, noise characteristics and working principles. Sensors provide measurement in their own frame of reference. To use multiple sensors together, measurements need to be transformed to a common reference frame. Next section provides more information about reference frames and transformation between them.

2.1 Reference Frames and Rotation Matrices

Points and vectors in a 2D plane or a 3D space are always defined with respect to a certain coordinate system. This coordinate system is called the *reference frame*, and within the localization scope it is often used to express the reference of a vehicle's position, velocity or orientation. Depending on the application, there are several coordinate frames of interest for localization algorithms [19]:

- **True Inertial Frame:** Sometimes also referred to as *Earth Centered Inertial Frame*, this static frame has its origin at the center of mass of the earth. Although not used directly when expressing position for AGV applications, all inertial sensors will provide readings with respect to this frame.

- **Earth Fixed Frame:** Also called the *World Frame*, this frame also has its origin at the center of mass of the earth, but it rotates together with the earth. In large scale navigation applications spanning kilometers, such as aircraft or marine applications, usually the position of interest is expressed in this reference frame.

- **Navigation Frame:** This frame is defined according to the area the AGV is expected to operate in, when it is smaller than the whole earth frame. This area is most often a map with a known and fixed origin. In most AGV applications, including the one studied in this thesis, this is the frame of interest when expressing the vehicle's position and orientation.

- **Vehicle Frame:** Also referred to as the *Body Frame*, this coordinate frame has its origin at the navigation center of the vehicle, and it moves and rotates with it. Odometry outputs are expressed with respect to this frame.

- **Sensor Frame:** The sensor frames are named after individual sensors (such as laser frame or IMU frame), and have their origin at the exact position of the sensor.

A representation of different reference frames of interest can be observed in Figure 1. In most cases, the sensors are rigidly placed on the vehicle, therefore, they are stationary with respect to the vehicle frame as illustrated in the figure. It should be noted that odometry outputs are generally obtained with respect to vehicle frame, hence there is no separate reference frame for odometry, as described in more detail in Chapter 2.2.

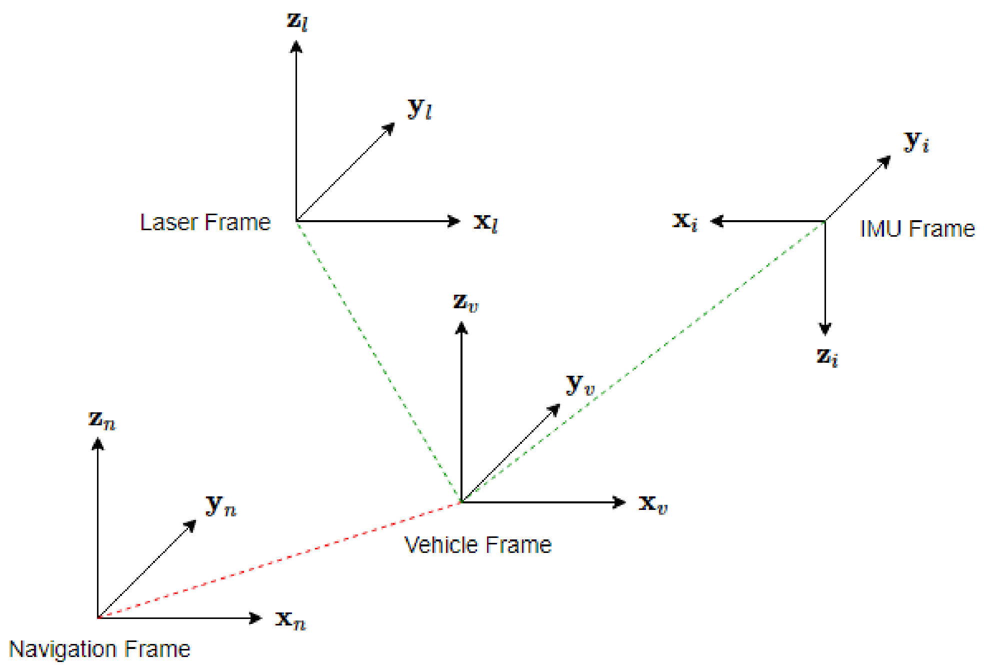


Figure 1: Different reference coordinate frames in AGV navigation, adapted from [20]. As in the original image, green lines represent the frame is stationary with respect to the parent while red line represents a moving frame.

In sensor fusion, velocity and acceleration vectors expressed with respect to one frame frequently have to be converted into other frames of reference, due to sensor outputs being in different coordinate systems from the navigation frame. This is achieved by *rotation matrices* that define how a frame is rotated with respect to another. A vector that is defined according to reference frame A can be converted to reference frame B with the following relation [21][22]:

$$\mathbf{v}_b = \mathbf{R}_a^b \mathbf{v}_a. \tag{1}$$

Here, $\mathbf{v}_a$ and $\mathbf{v}_b$ represent the vector in a and b frames, while $\mathbf{R}_a^b$ is the matrix describing the rotation of frame b with respect to frame a. Rotation matrices follow the relationship

$$\mathbf{R}_b^a = (\mathbf{R}_a^b)^T = (\mathbf{R}_a^b)^{-1} \tag{2}$$

that states inverse of the rotation matrix, which is also equal to its transpose, can be used to describe the reverse rotation between frames, in this case the rotation of frame a with respect to frame b.

A common way to calculate the rotation matrix is using *Euler angles*, that is, to express the rotation in a combination of rotations around x, y and z axes [21]. A rotation around x axis by ψ is given by the matrix

$$\mathbf{R}_x(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\psi) & -sin(\psi) \\ 0 & sin(\psi) & cos(\psi) \end{bmatrix} \tag{3}$$

while a rotation around y axis by θ is

$$\mathbf{R}_y(\theta) = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{4}$$

and a rotation around z axis by ϕ is calculated with

$$\mathbf{R}_z(\phi) = \begin{bmatrix} cos(\phi) & -sin(\phi) & 0 \\ sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

When the rotation angles ψ, θ and ϕ are used to define the orientation of the vehicle frame with respect to navigation frame, they are referred to as *roll*, *pitch* and *yaw* angles, respectively. These angles can be used in defining the rotation between navigation and vehicle frames, hence, can be used to convert measurements obtained from various sensors into quantities in navigation frame. The full rotation matrix can be formed by the multiplication of those three matrices as

$$\mathbf{R}(\phi, \theta, \psi) = \mathbf{R}_z(\phi)\mathbf{R}_y(\theta)\mathbf{R}_x(\psi). \tag{6}$$

Substituting the matrices and denoting cosine and sine operations with $c()$ and $s()$ for brevity of notations, Equation 6 can be written as a single matrix [21] as

$$\mathbf{R}(\phi, \theta, \psi) = \mathbf{R}_{lin} = \begin{bmatrix} c(\phi)c(\theta) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) & c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) \\ s(\phi)c(\theta) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) \\ -s(\theta) & c(\theta)s(\psi) & c(\theta)c(\psi) \end{bmatrix} \tag{7}$$

It should be noted that as matrix multiplication is not commutative, the order of rotations matters in the resulting rotation matrix. In Equation 7, the rotation is first performed on the roll, then pitch and finally in yaw. In navigation of ground vehicles, roll and pitch angles are expected to change much less than the yaw, so first aligning the small differences in roll and pitch to the navigation frame is a commonly used method.

With the matrix in Equation 7, sensor outputs in vector form such as linear velocities or linear accelerations in vehicle frame can be converted to the navigation frame. If the sensor outputs are not directly in vehicle frame but are in the sensor's own frame, those outputs need to be converted first to vehicle frame with a rotation matrix in the same form. However, angular velocities obtained from a gyroscope are converted differently, since each angular velocity also rotates its respective frame which needs to be accounted for in Equation 6. Angular velocities are therefore related to Euler Angle rates of change as follows [23]:

$$\begin{aligned}\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} &= \mathbf{R}(\psi)R(\theta)\begin{bmatrix}0\\0\\ \dot{\phi}\end{bmatrix} + \mathbf{R}(\psi)\begin{bmatrix}0\\ \dot{\theta}\\ 0\end{bmatrix} + \begin{bmatrix}\dot{\psi}\\0\\0\end{bmatrix}\\ &= \begin{bmatrix}1 & 0 & -sin(\theta)\\ 0 & cos(\psi) & sin(\psi)cos(\theta)\\ 0 & -sin(\psi) & cos(\psi)cos(\theta)\end{bmatrix}\begin{bmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix}.\end{aligned} \tag{8}$$

In Equation 8, $\dot{\psi}$, $\dot{\theta}$ and $\dot{\phi}$ represent roll, pitch and yaw rates in the navigation frame while ω_x, ω_y and ω_z are the angular velocities around each axis in the vehicle frame. Taking the inverse results in the angular velocity rotation matrix that can be used to convert gyroscope outputs to Euler Angle rates as follows [24][25][26]:

$$\begin{bmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix} = \begin{bmatrix}1 & tan(\theta)sin(\psi) & tan(\theta)cos(\psi)\\ 0 & cos(\psi) & -sin(\psi)\\ 0 & sin(\psi)/cos(\theta) & cos(\psi)/cos(\theta)\end{bmatrix}\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} = \mathbf{R}_{ang}\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix}. \tag{9}$$

It can be seen from the $cos(\theta)$ terms in the denominators in Equation 9 that there is a singularity at pitch angle of 90 degrees [21]. Therefore, for applications where such a situation is possible including aerial and marine navigation, other representations for rotation such as *quaternions* are preferred. However, for many land applications, the pitch does not approach 90 degrees in normal operational conditions, so the Euler Angles are chosen as adequate representations for rotation within the scope of this thesis. In the reminder of this work, transformation matrices in Equations 7 and 9 are referred to as $\mathbf{R}_{lin}$ and $\mathbf{R}_{ang}$, respectively, to denote they are used to transform linear velocities and accelerations, and angular velocities. It is noted that even though the angular rotation matrix is denoted with the same $\mathbf{R}$ symbol to indicate it can be used for rotations, it is actually not a rotation matrix by itself as its inverse is not equal to its transpose.

2.2 Odometry

For AGV navigation, odometry is defined as using the data from actuators and motion sensors on the vehicle, such as those on wheels or threads, to determine the motion of the vehicle [21, p. 477-479]. It is commonly used in navigation applications, thanks to being able to provide information independent of any external conditions and resources.

The main working principle of this method is to convert velocities of the wheels into linear and angular velocity of the vehicle. Therefore, depending on the locomotion method of the vehicle, the conversion process is different. The following sections introduce these processes for relevant locomotion methods, and finally the output and possible errors in odometry readings are presented.

2.2.1 Differential Drive Odometry

A differential drive vehicle has two wheels mounted on a common axis, that are independently controllable [27]. By controlling the wheels to turn forward or backwards, a rolling motion can be achieved around a point lying on the wheel's common axis, which is referred to as Instantaneous Center of Curvature (ICC) [21]. Differential drive motion can be observed in Figure 2.

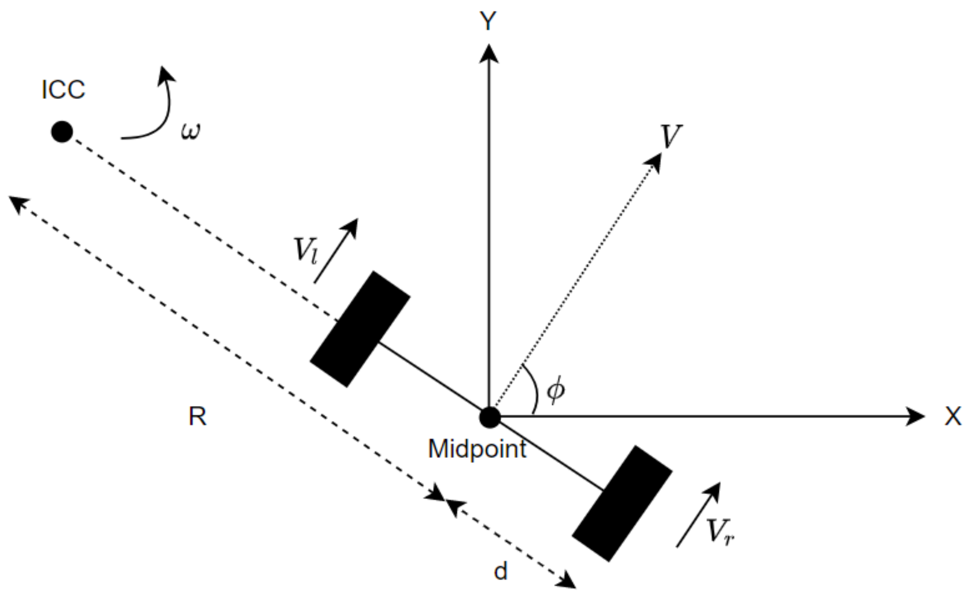


Figure 2: Motion of a differential drive vehicle.

Here, the midpoint of the wheels is referred as the navigation point of the vehicle, and is considered to be the origin of the *vehicle frame* explained in Section 2.1. It is located at a distance of d from each wheel, and R from the ICC. The angular velocity of the vehicle is denoted as ω, and V_l and V_r are the linear velocities of the left and right wheels, respectively. The relationship between the angular velocity and wheel velocities can be written as

$$\omega(R+d) = V_r \tag{10}$$

$$\omega(R-d) = V_l. \tag{11}$$

They can be solved for R and ω as

$$R = \begin{cases} d\,\frac{V_r+V_l}{V_r-V_l} & V_r \neq V_l \\ 0 & otherwise \end{cases} \tag{12}$$

$$\omega = \frac{V_r - V_l}{2d}. \tag{13}$$

From the above equations, it can be deduced that if the wheels have the same velocity in the same direction ($V_r = V_l$), the angular velocity is zero and the vehicle moves without turning. On the other hand, if the velocities are equal in the reverse direction ($V_r = -V_l$), it follows that the ICC is at the midpoint and the vehicle rotates in place [27]. The instantaneous velocity of the midpoint can later be calculated as

$$V = \begin{cases} \omega\, R & V_r \neq V_l \\ V_r = V_l & otherwise \end{cases} . \tag{14}$$

It should be noted that the linear velocity given in Equation 14 is obtained with respect to the vehicle frame, in the *longitudinal* direction, which is the vehicle froward direction. The *latitudinal* direction velocity, that is perpendicular to the vehicle heading, is always zero in the differential drive case as the vehicle cannot move (without an external force) perpendicularly to the wheel base axis. Therefore, the linear velocity vector in the vehicle frame is given by

$$\mathbf{v}_{veh} = \begin{bmatrix} V_{lon} \\ V_{lat} \\ V_z \end{bmatrix} = \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix} . \tag{15}$$

Similarly, the angular velocity ω represents the angular velocity around the z axis in the horizontal 2D plane, resulting in an angular velocity vector in vehicle frame as

$$\omega_{veh} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} . \tag{16}$$

To find the linear rate of change in position and orientation in navigation frame, those velocities are converted by using Equations 7 and 9 as follows:

$$\mathbf{v}_{nav} = \mathbf{R}_{lin}\, \mathbf{v}_{veh} \tag{17}$$

$$\omega_{nav} = \mathbf{R}_{ang}\, \omega_{veh}. \tag{18}$$

It is noted here that if the analysis is constrained to a 2D frame and the tilt of the vehicle is disregarded, Equation 17 reduces to a simple trigonometric multiplication as $v_x = V\, cos(\phi)$ and $v_y = V\, sin(\phi)$ while Equation 18 becomes the identity equation. Since odometry itself does not provide any 3D information, those simplified forms could be used in an analysis considering only odometry, but 3D forms of conversion are used in sensor fusion including IMU.

2.2.2 Omnidirectional Odometry

Similar to the differential drive AGV, an omnidirectional AGV also has independently controllable wheels, but the number of wheels is at least three [5] to gain the ability to follow any trajectory in a given plane. In other words, omnidirectional vehicles can move in both longitudinal and latitudinal directions in the vehicle frame independent of the heading of the vehicle in the navigation frame. Although four or more wheeled configurations also exist [28], this section focuses on the kinematics of three wheeled omnidirectional vehicles.

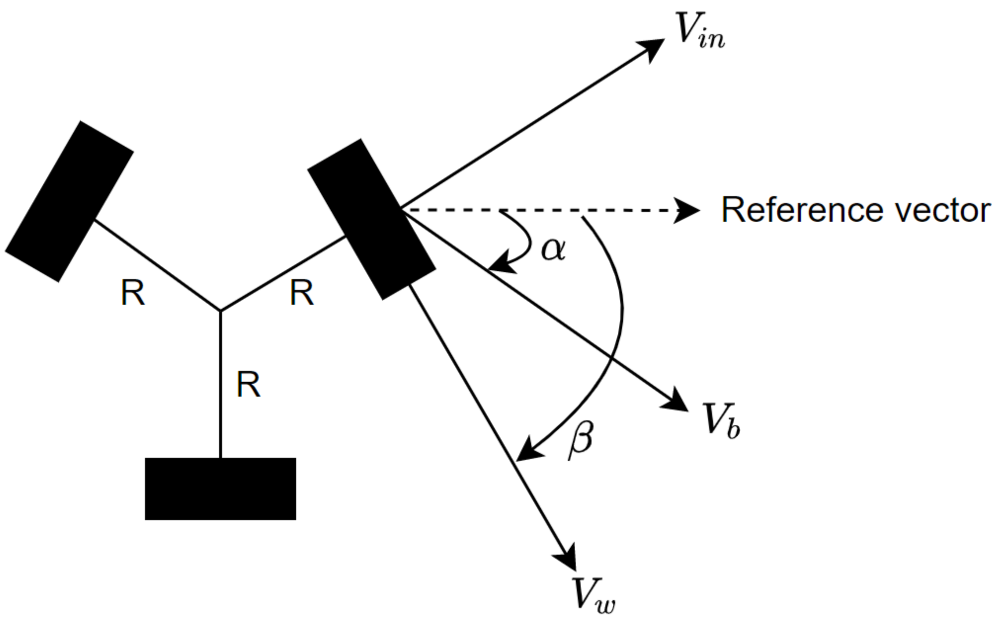


Figure 3: Motion of a omnidirectional drive vehicle, adapted from [5] and [29].

A detailed analysis of omnidirectional drive kinematics is presented by [29] and [5]. The wheel arrangement of a three wheeled omnidirectional drive vehicle can be observed in Figure 3.

Unlike differential drive system, the omni-drive wheels are able to obtain *induced velocity*, which is the velocity induced by the driving force in the other wheels in the sliding direction of the wheel [29]. It is always perpendicular to the wheel velocity, and is illustrated as V_{in} in Figure 3. In the figure, the reference vector denotes the x axis of the vehicle frame, and V_b is the velocity of the vehicle body while V_w is the velocity of the wheel. The vehicle velocity is at an angle of α from the reference vector, while the wheel angle is denoted by β. It is common in a three wheeled omnidirectional vehicle that wheel angles are 0°, 120°and 240°.

In such an arrangement, the relationship between the vehicle body velocity and the wheel velocities are found by [29] as

$$V_w = V_b\,(cos\beta\,cos\alpha + sin\beta\,sin\alpha) + R\omega \tag{19}$$

where ω denotes the angular velocity of the vehicle body around z axis in the vehicle frame. As in the differential drive case, the quantities of interest obtained from the wheel odometry are the longitudinal and latitudinal velocities, that is, the linear velocities in the direction of x and y axes of the vehicle frame. With a similar projection of rotation to 2D plane with the differential drive case, they are given as

$$V_{bx} = V_b\,cos\alpha \tag{20}$$

$$V_{by} = V_b\,sin\alpha. \tag{21}$$

Thus, writing Equation 19 for all three wheels and substituting the longitudinal and latitudinal velocities of the body results in [29]

$$\begin{bmatrix} V_{w1} \\ V_{w2} \\ V_{w3} \end{bmatrix} = \begin{bmatrix} cos(\alpha_1) & sin(\alpha_1) & R \\ cos(\alpha_2) & sin(\alpha_2) & R \\ cos(\alpha_3) & sin(\alpha_3) & R \end{bmatrix} \begin{bmatrix} V_{bx} \\ V_{by} \\ \omega \end{bmatrix}. \tag{22}$$

Finally, by inverting the matrix in Equation 22, the desired velocities can be obtained from the wheel velocities. It should be noted that the described motion dynamics are only valid in the simplified case of three wheels that are not rotating. Using N wheels instead of three would increase the size of the matrix in Equation 22 to $N \times 3$, implying that the wheel velocities needed to obtain the desired motion of V_{bx}, V_{by} and ω in the vehicle is not unique in this case. Furthermore, adding a rotational motion to the wheels would change Equation 19. These other cases of omnidirectional equations are presented in [29], but for the scope of sensor fusion, it is sufficient to note that motion of the vehicle can be obtained from motion of the wheels by using a similar relation to Equation 22.

Once the linear velocities are obtained with respect to vehicle frame, they can be converted to the navigation frame by using the same rotation matrices as in differential drive case, the main difference being that the latitudinal linear velocity can take nonzero values in omnidirectional case.

2.2.3 Odometry Output and Errors

Regardless of the locomotion type of the vehicle, the odometry measurements are in the form of linear and angular velocity of the vehicle. After transformation to navigation frame, these measurements have to be integrated with respect to time to obtain the vehicle's relative position and orientation as follows:

$$x(t) = \int V_x(t)dt \tag{23}$$

$$y(t) = \int V_y(t)dt \tag{24}$$

$$\phi(t) = \int \omega(t)dt. \tag{25}$$

In an ideal system without any kind of noise or errors, the position can be accurately obtained by adding the relative positions to a known initial position. However, that is rarely the case in real-life applications, and the odometry readings include many possible errors that can be grouped as follows [30][31]:

- **Systematic errors:** These errors depend on the AGV itself and are independent of the environment. They include unequal wheel diameters, wheel misalignment, encoder resolution and sampling rate limitations.

- **Non-systematic errors:** These errors depend on the environment the AGV is operating at. They include traveling on uneven floor, unexpected items on the trajectory, slippage of wheels and external forces on the AGV. Unless the operating environment is limited, it is not possible to avoid this category of errors entirely, and they tend to be non-deterministic in general.

While the first category of errors can be avoided by precise modeling and other errors are usually relatively small in each reading, the errors get integrated when determining the position and orientation according to Equations 23 to 25, growing larger with time. This is called the *odometry drift*. While there is a lot of research dedicated to minimize this drift, it is not possible to eliminate it completely, particularly due to non-deterministic error causes in the non-systematic errors. Therefore, in localization, odometry is most often used together with absolute position and orientation measurements to cancel the drift [32], as is the case in its usage for sensor fusion.

To use odometry outputs in a Kalman Filter, as explained later in Chapters 3 and 4, the errors need to be approximated as Gaussian white noise. While approximating the non-systematic errors in such a fashion is possible and yields accurate results [31], systematic errors can induce a scaling factor to the odometry outputs that cannot be modeled as white noise. Therefore, it is assumed in the scope of this thesis that the odometry is properly calibrated and the resulting velocities do not need to be scaled before being used in sensor fusion.

2.3 Inertial Measurement Unit

An Inertial Measurement Unit (IMU) consists of a combination of accelerometers, gyroscopes and magnetometers that measure acceleration, angular velocity and orientation, respectively. They are frequently used in AGV localization applications since they are able to provide accurate measurements independent of external conditions, which makes them a reliable and low cost solution to improve the vehicle odometry together with the data from wheel encoders. Most IMUs have either 6 or 9 degrees of freedom, the former group consisting of only an accelerometer and a gyroscope while the latter group also including a magnetometer. Even though magnetometers provide absolute heading data in contrast to relative measurements of accelerometers and gyroscopes, they are prone to errors if strong external magnetic disturbances are present [33], which is commonly the case in many AGV navigation areas such as factories. Therefore, within the scope of this thesis, 6 degrees of freedom IMUs are considered.

2.3.1 Gyroscope

A gyroscope provides angular velocity measurements in the IMU coordinate frame, however, the measurements are corrupted with a slowly varying bias term and a noise error term [34]. This results in a measurement equation of the following form

$$\omega_m(t) = \omega_i(t) + \mathbf{b}(t) + \epsilon(t). \tag{26}$$

Here, $\omega_m(t)$ denotes the measured angular velocity, $\omega_i(t)$ is the actual angular velocity in the IMU coordinate frame, $\mathbf{b}(t)$ is the bias term and $\epsilon(t)$ is the noise. While $\epsilon(t)$ can be accurately approximated as white noise [34], the bias term has a nonzero mean. It depends on many factors such as start-up currents and temperature, so modeling it as a constant is also not an accurate approximation. Therefore, this bias term has to be estimated and subtracted from the measurements to accurately utilize gyroscope measurements. There is a lot of research dedicated for gyroscope bias estimation [35][36][37] and a method of estimating it within sensor fusion is presented in Chapter 4.

It should be noted that the true values of $\omega_i(t)$ are given in the IMU frame, while the quantities of interest are the rate of changes in vehicle orientation in the navigation frame, hence, Equation 9 is used to convert gyroscope outputs to desired quantities. Unless the IMU and the vehicle frames are aligned, Equation 9 should be preceded by a conversion from IMU to vehicle frame first, but in the implementation described in Chapter 4, those frames are aligned with each other, hence the rotation between them is just a multiplication with identity matrix.

2.3.2 Accelerometer

An accelerometer outputs the acceleration of the sensor by measuring the specific force at each time instant [34]. This force includes the linear acceleration of the sensor and the effect of gravity vector, such that

$$\mathbf{f}_i(t) = \mathbf{a}_i(t) - \mathbf{R}_n^i \mathbf{g}_n. \tag{27}$$

Here, $\mathbf{f}_i(t)$ is the measured force in the IMU frame, $\mathbf{a}_i(t)$ is the linear acceleration in the IMU frame, $\mathbf{R}_n^i$ is the rotation matrix between navigation and the IMU frame and $\mathbf{g}_n$ is the gravity vector in the navigation frame. In most land AGV applications, the navigation frame is considered a perfectly horizontal one according to the earth frame, such that

$$\mathbf{g}_n \sim= \begin{bmatrix} 0 \\ 0 \\ -9.81\, m/s^2 \end{bmatrix}. \tag{28}$$

It should be noted that depending on the convention of navigation frames, such as whether z axis is pointing upwards or downwards, the minus sign in the gravity vector and the subtraction operation in Equation 27 might be reversed. Furthermore, if the IMU is positioned perfectly horizontally in the navigation frame, the effect of gravity on accelerations in x and y axes would be reduced to zero. This is rarely the case in many applications due to uneven terrain, and even small roll and pitch angles can cause great errors in acceleration measurements due to gravity vector having a much larger magnitude than typical accelerations of AGVs [38]. Therefore, it is important that the gravity vector is removed before using the accelerometer measurements.

In addition to the gravity effect, accelerometer measurements also exhibit a slow moving bias and white noise similar to that of gyroscopes [34]. The measurement model is therefore given by

$$\mathbf{a}_m(t) = \mathbf{f}_i(t) + \mathbf{b}(t) + \epsilon(t) \tag{29}$$

where $\mathbf{a}_m(t)$ is the measurement, $\mathbf{b}(t)$ is the bias and $\epsilon(t)$ is the white noise component. Substituting Equation 27 yields

$$\mathbf{a}_m(t) = \mathbf{a}_i(t) - \mathbf{R}_n^i \mathbf{g}_n + \mathbf{b}(t) + \epsilon(t) \tag{30}$$

as the full measurement equation. However, it is important to consider that the measurement processed this way relates the measurements to the accelerations in the IMU frame, while the actual quantities of interest are the accelerations in the navigation frame that are related by the rotation matrix

$$\mathbf{a}_i(t) = \mathbf{R}_n^i \mathbf{a}_n(t). \tag{31}$$

This transforms the measurement model into following form:

$$\mathbf{a}_m(t) = \mathbf{R}_n^i(\mathbf{a}_n(t) - \mathbf{g}_n) + \mathbf{b}(t) + \epsilon(t). \tag{32}$$

It should further be considered that the term $\mathbf{a}_n(t)$, which is the acceleration in the navigation frame, includes an implicit error due to rotation of the earth. However, its magnitude is small enough to be absorbed in gravity and white noise vectors, so that neglecting it does not have an effect on the reliability of the model in Equation 32 [34].

When there are no forces acting on the vehicle except for gravity, an accelerometer can also be used as a tilt estimator [39]. In Equation 32, if the linear acceleration is zero and the gravity vector is known, the unbiased accelerometer measurements can be used to deduce the terms in the rotation matrix as follows:

$$\psi = arctan(\frac{a_y}{a_z}) \tag{33}$$

$$\theta = arctan(\frac{-a_x}{\sqrt{a_y^2 + a_z^2}}). \tag{34}$$

Here, a_x, a_y and a_z are the acceleration in each axis while ψ and θ denote the roll and the pitch of the sensor. As the only other source of roll and pitch are the integrated gyroscope measurements that are subject to drifting, these absolute measurements can be used in sensor fusion to cancel that drift. It should be noted that in situations where the linear acceleration of the vehicle is high, or in applications with more force sources such as legged robot navigation, those measurements may have high noise [40]. This has to be taken into account when using them for sensor fusion, as demonstrated in Chapter 4.

2.4 Laser Scanner

All the sensors introduced until now provide relative measurements in the form of position and orientation difference with the previous timestep. A positioning system only relying on such measurements will inevitably drift away from the correct position in time, therefore, a source of absolute measurements is necessary in localization systems. In outdoor environments they can be provided by a GPS, but the signal is generally too unreliable to use in indoor applications. Instead, due to their ease of usage and ability to provide accurate measurements, laser scanners are extensively used in indoor localization.

A laser scanner, also referred to as a *laser range finder*, measures the distance to nearby objects [41]. It consists of a transmitter that emits laser beams, a receiver that is able to detect reflected beams and a rotating mirror to enable scanning 360° in the environment [42]. A representation of laser scanners can be observed in Figure 4.

Some of the most commonly used methods to determine the distance to an object with laser scanners include *time of flight* and *phase shift* scanners [42]. The time of flight sensors measure the distance D by calculating the time Δt it takes for a light beam with speed of light $c \approx 3 \times 10^8 m/s$ to travel to the object and back, as

$$D = \frac{c\,\Delta t}{2}. \tag{35}$$

As the speed of light is very high, the time Δt in Equation 35 is extremely short and can only be measured by expensive and very precise sensors [42]. Instead, phase shift scanners calculate the more easily measurable phase difference θ between the transmitted and reflected lights. To that end, they emit a light beam with known wavelength λ and frequency f that obeys the equation

$$c = \lambda f. \tag{36}$$

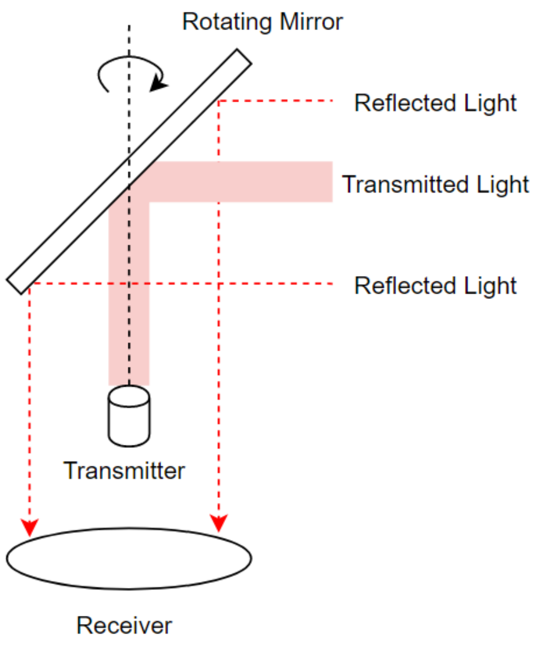


Figure 4: Laser scanner with rotating mirror, adapted from [42].

The measured phase difference can be written as

$$\mu = 2\pi f \Delta t. \tag{37}$$

Substituting Equations 36 and 37 into Equation 35 yields

$$D = \frac{c\,\Delta t}{2} = \frac{\lambda f \mu}{4\pi f} = \frac{\lambda}{4\pi}\mu. \tag{38}$$

With Equation 38, the distance to nearby objects can be calculated by measuring the phase difference instead of time difference [42].

There are many methods to use the distance measurements of a laser scanner for localization of AGVs. Some of them rely on artificial landmarks, such as reflectors, installed in known locations in the operating environment. They measure the distance between the scanner mounted on an AGV and those landmarks then determine the location of the AGV with triangulation of those distances [43][44]. While such methods have been proven to be effective, they have the disadvantage of requiring an installation of these artificial landmarks, which is not always a viable option [43].

Instead, many indoor applications make use of natural features such as corners, walls and objects to create a *map* of the environment with the laser scanner utilizing mapping methods such as Simultaneous Localization and Mapping (SLAM) [45][46]. The resulting map consists of orientations and distances of natural features, and the navigation frame introduced in Section 2.1 is aligned with the map. An example map generated in an office environment can be seen in Figure 5.

Once the map of an environment is available, the problem of localization with a laser reduces to determining the pose of the scanner in the map. One of the most

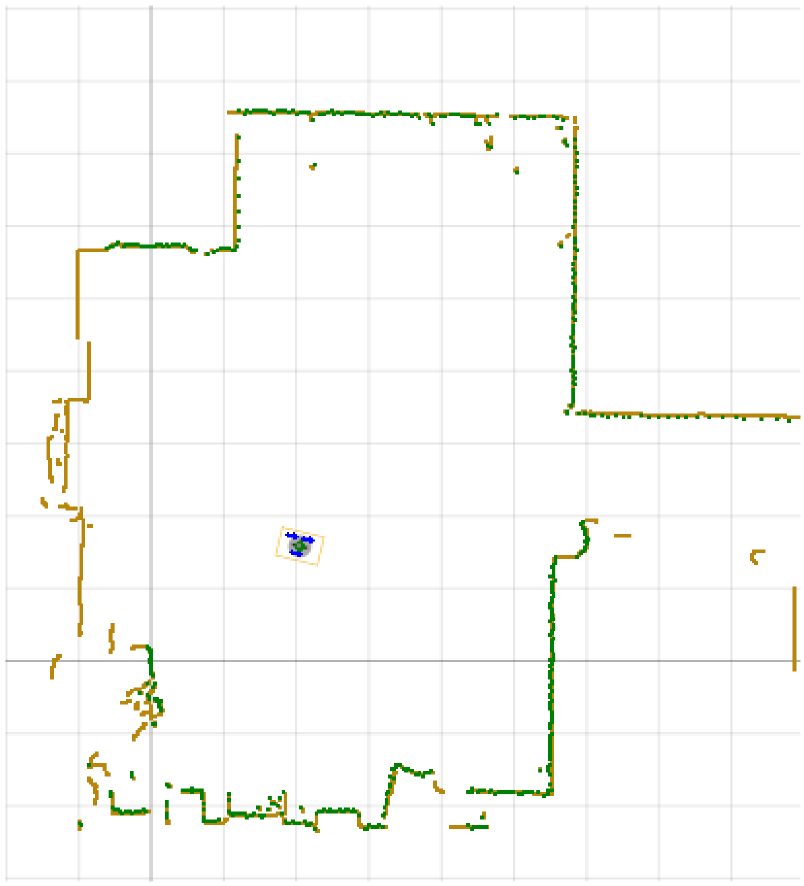


Figure 5: An example map of an office environment, displaying the position of the AGV with the environment features. The scan points can be seen in green, while the walls of the room are displayed with orange lines.

commonly used techniques for that is called *map matching*, which relies on creating a local map denoted as m_{local} and looking for the orientation and position that has the greatest fit between this local map and the global map m [41]. As the local map is expressed in AGV frame while the global map is in the navigation frame, the first step is to align them with a rotation matrix $\mathbf{R}$, which in a 2D case only depends on the yaw angle of the AGV as follows:

$$\mathbf{R}(\phi) = \begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix}. \tag{39}$$

Once the local and global maps are aligned, the correlation between them can be found by the following equation:

$$\rho_{m,m_{local},x_t} = \frac{\sum_{x,y}(m_{x,y} - \bar{m}) \times (m_{x,y,local}(x_t) - \bar{m})}{\sqrt{\sum_{x,y}(m_{x,y} - \bar{m})^2 \sum_{x,y}(m_{x,y,local}(x_t) - \bar{m})^2}}. \tag{40}$$

Here, x_t is the location of the AGV in the navigation frame, while $m_{x,y,local}(x_t)$ and $m_{x,y}(x_t)$ are the grid cells corresponding to $[x\ y]^T$ in local and global maps, respectively. $\bar{m}$ denotes the average map value computed as

$$\bar{m} = \frac{1}{2N} \sum_{x,y} (m_{x,y} + m_{x,y,local}) \tag{41}$$

where N denotes the number of overlapping elements between m_{local} and m. The main goal of map matching method is therefore to find ϕ and x_t such that the probability of the local map conditioned on the global map, calculated as

$$p(m_{local} \,|\, x_t, m) = max(\rho_{m,m_{local},x_t}, 0) \tag{42}$$

is maximized.

In the literature, there are many methods available to be utilized in map matching [43]. One of the most commonly used methods is called *Iterative Closest Point (ICP)*, and it tries to iteratively find a rotation matrix $\mathbf{R}$ and a translation matrix $\mathbf{t}$ such that the Euclidian distance between the global and local maps are minimized [43][47].

Whichever algorithm is used, the output of the map matching method is the robot pose and orientation in the navigation frame, which can be denoted in matrix form as

$$\mathbf{z} = \begin{bmatrix} x_{nav} \\ y_{nav} \\ \phi \end{bmatrix}. \tag{43}$$

Similarly to the other sensors, this output is not free of noise. Possible causes of the measurement noise are listed as follows:

- There is a Gaussian error present in laser scans, resulting from atmospheric conditions and low resolution [41].
- Randomly appearing dynamic objects in the map, such as humans walking around, cause an error term with an exponential distribution [41]. In the literature, some methods for combating this kind of interference can be found [48][49].
- In edges and corners, laser beam can hit more than one surface before being reflected back to the receiver and erroneously output a mean value [50][51].
- Due to shiny and reflective surfaces in the environment, unreliable measurements in the form of outliers can be observed [50]. On the other hand, due to black or non-reflective materials, the scanner may fail to detect an object completely [41].

As explained in Chapters 3 and 4, the usage of measurements in Kalman Filter relies on noise components having a Gaussian distribution. Therefore, for sensor fusion purposes, all other noises are assumed to be filtered out before being used as an input to the algorithm.

3 Kalman Filter

This chapter presents the Kalman Filter, which is the chosen method in this thesis to merge the measurements from different sensors and provide a position estimate.

3.1 Introduction to Kalman Filter

Kalman filter is defined as an *optimal* and *recursive linear estimator* that provides an estimate of a *state* that changes over time, based on periodic *observations* which have a linear relation to the state [52][53]. Due to its success in efficiently providing estimates with noisy observations, as well as its light computational power and memory requirements, Kalman Filter has been extensively used in numerous areas including navigation, economics, computer vision and object tracking [54].

A key aspect of the Kalman Filter is that it estimates the *state*, which is the collection of variables that provide a complete representation of the status of a system [15]. In navigation, the state usually includes the position, linear velocity, acceleration, orientation and angular velocity of the vehicle at a given time, and it can be reduced or expanded with additional variables depending on the application. It is denoted by a $n \times 1$ vector $\mathbf{x}$, each element of which corresponds to a different state variable.

The *recursive* nature of the Kalman Filter enables processing new measurements as they arrive, meaning that the state estimation will be updated at a given time according to the latest received measurement [53]. This makes the Kalman Filter ideal to use in real-time navigation applications, where the navigation state must be updated with each sensor reading. Furthermore, as an *optimal estimator*, Kalman Filter is able to provide an estimate that minimizes the error as long as all noise can be characterized as Gaussian white noise [53][15]. This property is explained further in Section 3.2.

While Kalman Filter can be used with both discrete and continuous time systems, in the scope of sensor fusion for localization, the quantity of interest is the state of the vehicle at the time of sensor measurement, which is a discrete time process. Furthermore, navigation equations can be easily formulated in a discrete time fashion. Therefore, the remainder of this thesis will only focus on discrete time Kalman Filters.

3.2 Linear Discrete Time System and Observation Models

A linear, discrete time system can be characterized with Equation 44.

$$\mathbf{x}(k) = \mathbf{F}(k)\mathbf{x}(k-1) + \mathbf{G}(k)\mathbf{u}(k) + \mathbf{w}(k). \tag{44}$$

Here, $\mathbf{x}(k)$ denotes the system state in sampling time instant k and $\mathbf{u}(k)$ is the known control input. $\mathbf{F}(k)$ and $\mathbf{G}(k)$ are referred to as the *state transition matrix* and *control input model*, respectively. Together, they characterize the linear process of how the state transitions into $\mathbf{x}(k)$, given the previous state $\mathbf{x}(k-1)$ and the control input $\mathbf{u}(k)$. Finally, $\mathbf{w}(k)$ denotes the *process noise*. An important assumption of the Kalman filter is that $\mathbf{w}(k)$ is white noise, meaning it has zero mean. It is characterized as follows [15]:

$$\begin{aligned} \mathbf{w}(k) &\sim N(0, \mathbf{Q}(k)) \\ E\left(\mathbf{w}(k)\right) &= 0 \\ E\left(\mathbf{w}(k)\mathbf{w}(k)^T\right) &= \mathbf{Q}(k). \end{aligned} \tag{45}$$

Here, $\mathbf{Q}(k)$ is referred to as *process noise covariance matrix.* A covariance matrix is a symmetric and positive semi-definite matrix, whose diagonal elements contain the variances of the random variable, in this case of the noise vector $\mathbf{w}(k)$, and off-diagonal elements represents correlations between variables [53]. The $\mathbf{Q}$ matrix should be designed depending on the noise characteristics of the state transition when applying a Kalman Filter algorithm. Its design is presented in Chapter 4 in detail for sensor fusion in navigation applications case.

It is assumed that the states of the system are observed according to a linear equation of the following form [52]:

$$\mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k) + \mathbf{v}(k). \tag{46}$$

Here, $\mathbf{z}(k)$ is referred to as the *observation* or *measurement* at time instant k and $\mathbf{H}(k)$ is the observation matrix that maps the states of the system to the measurements. Similarly to the process noise, $\mathbf{v}(k)$ is called the *measurement noise* and has white noise characteristics as

$$\begin{aligned} \mathbf{v}(k) &\sim N(0, \mathbf{R}(k)) \\ E\left(\mathbf{v}(k)\right) &= 0 \\ E\left(\mathbf{v}(k)\mathbf{v}(k)^T\right) &= \mathbf{R}(k). \end{aligned} \tag{47}$$

In sensor fusion, $\mathbf{R}(k)$ is the measurement noise matrix of each sensor providing measurements to the system. It should be designed when applying the Kalman Filter according to the known noise characteristics of sensor readings.

3.3 Kalman Filter Algorithm

The Kalman Filter estimates the state in time instant i given the measurements up to time instant j such that the Mean Squared Error (MSE) of the estimation is minimized [52], which is represented in the following equation:

$$\hat{\mathbf{x}}(i\,|\,j) = \underset{\hat{\mathbf{x}}(i|j)\in\mathcal{R}^n}{\operatorname{argmin}}\, E\left[(\mathbf{x}(i) - \hat{\mathbf{x}}(i\,|\,j))(\mathbf{x}(i) - \hat{\mathbf{x}}(i\,|\,j))^T\,|\,\mathbf{z}(1), ...\mathbf{z}(j)\right]. \tag{48}$$

In Equation 48, $\hat{\mathbf{x}}$ denotes the estimated state vector of the system. This Kalman Filter estimation of the state is the expected value of the state that is conditioned on the measurements [52][15], such that

$$\hat{\mathbf{x}}(i\,|\,j) = E\left(\mathbf{x}(i)\,|\,\mathbf{z}(1), ...\mathbf{z}(j)\right) \tag{49}$$

and the MSE of the estimate is

$$\mathbf{P}(i\,|\,j) = E\left[(\mathbf{x}(i) - \hat{\mathbf{x}}(i\,|\,j))(\mathbf{x}(i) - \hat{\mathbf{x}}(i\,|\,j))^T\,|\,\mathbf{z}(1),\,\ldots\mathbf{z}(j))\right]. \tag{50}$$

P matrix is called the *state covariance matrix* of the Kalman Filter. Its diagonal elements contain variances of corresponding state variable estimates.

As a recursive estimator, the Kalman Filter assumes an initial estimate of the state with a known covariance matrix is available as a starting point at $k = 0$, denoted as $\hat{\mathbf{x}}(0\,|\,0)$ and $\mathbf{P}(0\,|\,0)$. As measurements become available at sampling times $k = 1, 2....$, the state estimation and the corresponding covariances are updated iteratively with a *prediction* and *measurement update* cycle.

3.3.1 Prediction Step

In the *prediction* step, Kalman Filter provides an *a priori* estimate of the state taking into account the state estimation and measurements up to previous sampling time, such that

$$\hat{\mathbf{x}}(k\,|\,k-1) = E\left(\mathbf{x}(k)\,|\,\mathbf{z}(1),\,\ldots\mathbf{z}(k-1)\right). \tag{51}$$

Substituting Equation 44 and taking into account the expected value of the white noise is zero,

$$\begin{aligned}\hat{\mathbf{x}}(k\,|\,k-1) &= E\left[\mathbf{F}(k)\,\mathbf{x}(k-1) + \mathbf{G}(k)\,\mathbf{u}(k) + \mathbf{w}(k)\,|\,\mathbf{z}(1),\,\ldots\mathbf{z}(k-1)\right]\\ &= \mathbf{F}(k)\,\hat{\mathbf{x}}(k-1\,|\,k-1) + \mathbf{G}(k)\,\mathbf{u}(k).\end{aligned} \tag{52}$$

The covariance of this estimate is given by

$$\mathbf{P}(k\,|\,k-1) = E\left[(\mathbf{x}(k) - \hat{\mathbf{x}}(k\,|\,k-1))(\mathbf{x}(k) - \hat{\mathbf{x}}(k\,|\,k-1))^T\right]. \tag{53}$$

Substituting Equations 44 and 52, it yields

$$\begin{aligned}\mathbf{x}(k) - \hat{\mathbf{x}}(k\,|\,k-1) &= \mathbf{F}(k)\,\mathbf{x}(k-1) + \mathbf{G}(k)\,\mathbf{u}(k) + \mathbf{w}(k)\\ &\quad - \mathbf{F}(k)\,\hat{\mathbf{x}}(k-1\,|\,k-1) - \mathbf{G}(k)\,\mathbf{u}(k)\\ \mathbf{x}(k) - \hat{\mathbf{x}}(k\,|\,k-1) &= \mathbf{F}(k)\left(\mathbf{x}(k-1) - \hat{\mathbf{x}}(k-1\,|\,k-1)\right) + \mathbf{w}(k).\end{aligned} \tag{54}$$

Taking into account that the white noise is uncorrelated to the state, such that $E(\mathbf{x}(k-1)\mathbf{w}(k)) = 0$ and denoting $\hat{\mathbf{x}}(k-1\,|\,k-1)$ as $\hat{\mathbf{x}}(k-1)$ for ease of notation, it follows that

$$\begin{aligned}\mathbf{P}(k\,|\,k-1) = E\,&[\mathbf{F}(k)(\mathbf{x}(k-1) - \hat{\mathbf{x}}(k-1))(\mathbf{x}(k-1) - \hat{\mathbf{x}}(k-1))^T\mathbf{F}^T(k)]\\ &+ E(\mathbf{w}(k)\mathbf{w}^T(k)).\end{aligned} \tag{55}$$

Therefore,

$$\mathbf{P}(k\,|\,k-1) = \mathbf{F}(k)\mathbf{P}(k-1\,|\,k-1)\mathbf{F}^T(k) + \mathbf{Q}(k). \tag{56}$$

Equations 52 and 56 constitutes the Kalman Filter prediction step, as they describe how the state and covariance estimations are updated before taking into account the latest available measurement. It should be noted that for these derivations, the control input $\mathbf{u}(k)$ is assumed to be perfectly known and without inducing noise by itself. This is not a necessary assumption for the Kalman Filter, but a control input noise with a covariance matrix of $\mathbf{C}(k)$ would alter Equation 56 as

$$\mathbf{P}(k\,|\,k-1) = \mathbf{F}(k)\mathbf{P}(k-1\,|\,k-1)\mathbf{F}^T(k) + \mathbf{G}(k)\mathbf{C}(k)\mathbf{G}^T(k) + \mathbf{Q}(k). \tag{57}$$

3.3.2 Measurement Update Step

Once the *a priori* estimate of the state and the covariance matrix are found, the next step of the Kalman Filter is the *measurement update* to obtain *a posteriori* estimate utilizing the measurement $\mathbf{z}(k)$. The remainder of this chapter follows the derivations presented in [52] and [15] with slight notation changes for consistency.

A linear posterior estimation of the state after an observation in the form of Equation 46 can be written as [15]

$$\hat{\mathbf{x}}(k\,|\,k) = \hat{\mathbf{x}}(k\,|\,k-1) + \mathbf{K}(k)\,(\mathbf{z}(k) - \mathbf{H}(k)\,\hat{\mathbf{x}}(k\,|\,k-1)). \tag{58}$$

Here, $\mathbf{K}(k)$ is referred to as the *Kalman Gain*, which needs to be calculated for minimizing the MSE of the estimation. It can be observed that Equation 58 displays a predictor-corrector structure [52], such that the result is a weighted sum of the prior prediction and a correction term by the measurement. The latter term is known as the *innovation*, and sometimes denoted in the literature as

$$\mathbf{y}(k) = \mathbf{z}(k) - \mathbf{H}(k)\,\hat{\mathbf{x}}(k\,|\,k-1). \tag{59}$$

The error term of this estimation is represented as

$$\tilde{\mathbf{x}}(k\,|\,k) = \hat{\mathbf{x}}(k\,|\,k) - \mathbf{x}(k). \tag{60}$$

Substituting Equations 58 and 46, it can be rewritten as:

$$\begin{aligned}\tilde{\mathbf{x}}(k\,|\,k) &= \hat{\mathbf{x}}(k\,|\,k) - \mathbf{x}(k)\\ &= \hat{\mathbf{x}}(k\,|\,k-1)\,(\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k)) + \mathbf{K}(k)\mathbf{z}(k)\\ &= (\hat{\mathbf{x}}(k\,|\,k-1) - \mathbf{x}(k))\,(\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k)) + \mathbf{K}(k)\mathbf{v}(k)\\ &= (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,\tilde{\mathbf{x}}(k\,|\,k-1) + \mathbf{K}(k)\mathbf{v}(k).\end{aligned} \tag{61}$$

The covariance can then be calculated by taking the square of the error term and getting the expected value as follows [52]:

$$\begin{aligned}\mathbf{P}(k\,|\,k) &= E[(\tilde{\mathbf{x}}(k\,|\,k)\tilde{\mathbf{x}}^T(k\,|\,k))\,|\,\mathbf{z}(1), ..., \mathbf{z}(k)]\\ &= (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,E(\tilde{\mathbf{x}}(k\,|\,k-1)\tilde{\mathbf{x}}^T(k\,|\,k-1))\,(\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))^T\\ &+ \mathbf{K}(k)E(\mathbf{v}(k)\mathbf{v}^T(k))\mathbf{K}^T(k)\\ &+ 2(\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,E(\tilde{\mathbf{x}}(k\,|\,k-1)\mathbf{v}^T(k))\mathbf{K}^T(k).\end{aligned} \tag{62}$$

Taking into account the identities of Equations 47 and 53 and the fact that the measurement noise and the state are uncorrelated, the expression for the covariance can be reduced to

$$\mathbf{P}(k\,|\,k) = (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,\mathbf{P}(k\,|\,k-1)\,(\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))^T + \mathbf{K}(k)\mathbf{R}(k)\mathbf{K}^T(k). \tag{63}$$

The optimal estimation of $\mathbf{K}(k)$ should minimize the mean squared estimation error denoted as $J(k)$, which is the mean of squared estimation errors of each element of $\tilde{\mathbf{x}}$ [15], given as

$$\begin{aligned} J(k) &= E(\tilde{\mathbf{x}}_1^2(k\,|\,k) + \ldots + \tilde{\mathbf{x}}_n^2(k\,|\,k)) \\ &= E(\tilde{\mathbf{x}}^T(k\,|\,k)\tilde{\mathbf{x}}(k\,|\,k)) \\ &= trace(E(\tilde{\mathbf{x}}(k\,|\,k)\tilde{\mathbf{x}}(k\,|\,k)^T)) \\ &= trace(\mathbf{P}(k\,|\,k)). \end{aligned} \tag{64}$$

In the minimum, the derivative of the error should be zero, hence,

$$\frac{\partial J(k)}{\partial \mathbf{K}(k)} = \frac{\partial(trace(\mathbf{P}(k\,|\,k)))}{\partial \mathbf{K}(k)} = 0. \tag{65}$$

The derivative of the trace of a matrix, for matrices in the form of $(\mathbf{ABA}^T)$ where $\mathbf{A}$ is any matrix and $\mathbf{B}$ is a symmetric matrix is given as [52]

$$\frac{\partial(trace(\mathbf{ABA}^T))}{\partial \mathbf{A}} = 2\mathbf{AB}. \tag{66}$$

Since the covariance matrices in Equation 63 are symmetric, the above identity can be applied by substituting it to Equation 65, which yields

$$\frac{\partial(trace(\mathbf{P}(k\,|\,k)))}{\partial \mathbf{K}(k)} = -2(\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,\mathbf{P}(k\,|\,k-1)\,\mathbf{H}^T(k) + 2\mathbf{K}(k)\mathbf{R}(k) = 0. \tag{67}$$

Solving for $\mathbf{K}(k)$, it can be obtained by

$$\mathbf{K}(k) = \mathbf{P}(k\,|\,k-1)\,\mathbf{H}^T(k)\,[\mathbf{H}(k)\mathbf{P}(k\,|\,k-1)\mathbf{H}^T(k) + \mathbf{R}(k)]^{-1}. \tag{68}$$

The inverted term in this equation is denoted as $\mathbf{S}(k)$ and referred to as *innovation covariance*. Therefore, the above equation can be written more compactly as

$$\mathbf{S}(k) = \mathbf{H}(k)\mathbf{P}(k\,|\,k-1)\mathbf{H}^T(k) + \mathbf{R}(k) \tag{69}$$

$$\mathbf{K}(k) = \mathbf{P}(k\,|\,k-1)\,\mathbf{H}^T(k)\,\mathbf{S}^{-1}(k). \tag{70}$$

Equation 70 is the optimal solution by the linear estimator minimizing the MSE given in Equation 64. Once the Kalman gain is computed, it can be substituted in Equation 58 to complete the update step of the filter, hence providing a state estimate and corresponding a covariance estimate taking all measurements into account.

After the computation of $\mathbf{K}(k)$, the covariance update given in Equation 63 can be further simplified by substituting $\mathbf{K}(k)$ in as

$$\mathbf{P}(k\,|\,k) = (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,\mathbf{P}(k\,|\,k-1). \tag{71}$$

Equations 58, 70 and 71 constitute the *measurement update* step of the Kalman Filter by describing the calculation of the optimal Kalman gain and updating the state and covariance estimations accordingly.

3.3.3 Summary

A summary of Kalman Filter algorithm is presented in Figure 6. When a new measurement is available, first an *a priori* estimate of the state and covariance are obtained in the prediction step. Then, the estimate is corrected with the received measurement in the measurement update step.

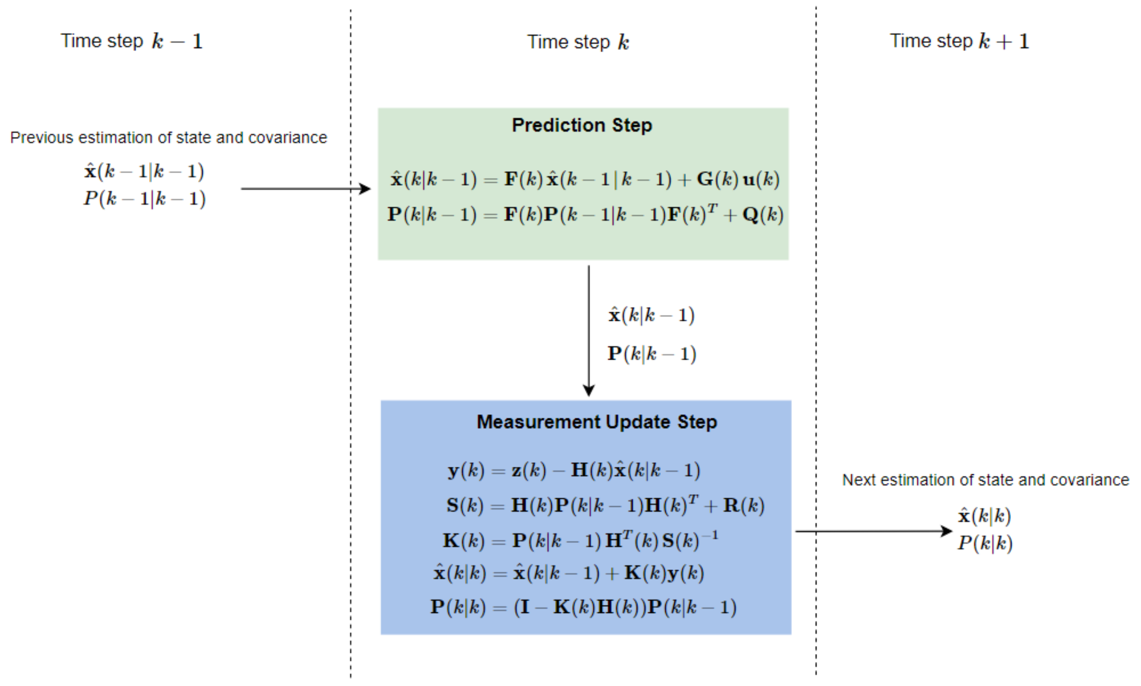

Time step $k-1$
Time step k
Time step $k+1$
Previous estimation of state and covariance
$\hat{\mathbf{x}}(k-1|k-1)$
$P(k-1|k-1)$
Prediction Step
$\hat{\mathbf{x}}(k|k-1) = \mathbf{F}(k)\hat{\mathbf{x}}(k-1\,|\,k-1) + \mathbf{G}(k)\mathbf{u}(k)$
$\mathbf{P}(k|k-1) = \mathbf{F}(k)\mathbf{P}(k-1|k-1)\mathbf{F}(k)^T + \mathbf{Q}(k)$
$\hat{\mathbf{x}}(k|k-1)$
$\mathbf{P}(k|k-1)$
Measurement Update Step
$\mathbf{y}(k) = \mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)$
$\mathbf{S}(k) = \mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k)$
$\mathbf{K}(k) = \mathbf{P}(k|k-1)\,\mathbf{H}^T(k)\,\mathbf{S}(k)^{-1}$
$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)\mathbf{y}(k)$
$\mathbf{P}(k|k) = (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\mathbf{P}(k|k-1)$
Next estimation of state and covariance
$\hat{\mathbf{x}}(k|k)$
$P(k|k)$

Figure 6: Recursive Kalman Filter estimation algorithm.

3.4 Extended Kalman Filter

The Kalman Filter explained so far had the assumptions that the states of the system are linearly related as in Equation 44 and the measurements are linearly related to states described with Equation 46. In many real life applications, those assumptions do not hold, and those relations are described with nonlinear functions $\mathbf{f}$ and $\mathbf{h}$ instead, as

$$\mathbf{x}(k) = \mathbf{f}\left(\mathbf{x}(k-1), \mathbf{u}(k), k\right) + \mathbf{w}(k) \tag{72}$$

$$\mathbf{z}(k) = \mathbf{h}\left(\mathbf{x}(k)\right) + \mathbf{v}(k). \tag{73}$$

As the linearity assumption is no longer valid, the optimal linear estimation techniques cannot be directly used with such a system. Instead, *Extended Kalman Filter (EKF)* attempts to linearize the $\mathbf{f}$ and $\mathbf{h}$ functions at the estimation point, and use the Kalman Filter on the linearized system [55].

As in the linear case, the EKF starts from a known previous estimate $\hat{\mathbf{x}}(k-1\,|k-1)$ and $\mathbf{P}(k-1\,|k-1)$. Equation 72 is then expanded with Taylor Series approximation [52][55] as follows:

$$\mathbf{x}(k) = \mathbf{f}\left(\hat{\mathbf{x}}(k-1\,|k-1), \mathbf{u}(k), k\right) + \nabla\mathbf{f}_{\mathbf{x}}(k)\left(\mathbf{x}(k-1) - \hat{\mathbf{x}}(k-1\,|k-1)\right) + \mathbf{w}(k) + \mathbf{H.O.T}. \tag{74}$$

Here, $\mathbf{H.O.T}$ denotes the higher order terms that are discarded with the first order Taylor Series approximation, and $\nabla\mathbf{f}_{\mathbf{x}}(k)$ is the *Jacobian* matrix of function $\mathbf{f}$ evaluated at $\mathbf{x}(k-1) = \hat{\mathbf{x}}(k-1\,|k-1)$. Taking the expected value, the *a priori* estimate is found as

$$\hat{\mathbf{x}}(k\,|\,k-1) = \mathbf{f}\left(\hat{\mathbf{x}}(k-1\,|\,k-1), \mathbf{u}(k), k\right). \tag{75}$$

The error corresponding to the estimation in Equation 75 is given by

$$\begin{aligned}\tilde{\mathbf{x}}(k\,|\,k-1) &= \mathbf{x}(k) - \hat{\mathbf{x}}(k\,|\,k-1)\\ &= \mathbf{f}\left(\hat{\mathbf{x}}(k-1\,|\,k-1), \mathbf{u}(k), k\right) + \nabla\mathbf{f}_{\mathbf{x}}(k)\left(\mathbf{x}(k-1) - \hat{\mathbf{x}}(k-1\,|k-1)\right)\\ &+ \mathbf{w}(k) - \mathbf{f}\left(\hat{\mathbf{x}}(k-1\,|\,k-1), \mathbf{u}(k), k\right)\\ &= \nabla\mathbf{f}_{\mathbf{x}}(k)\left(\mathbf{x}(k-1) - \hat{\mathbf{x}}(k-1\,|\,k-1)\right) + \mathbf{w}(k). \end{aligned}\tag{76}$$

Similarly to the linear case, the covariance of the estimation is found as

$$\begin{aligned}\mathbf{P}(k\,|\,k-1) &= E(\tilde{\mathbf{x}}(k\,|k-1)\,\tilde{\mathbf{x}}^T(k\,|k-1))\\ &= \nabla\mathbf{f}_{\mathbf{x}}(k)\,\mathbf{P}(k-1\,|k-1)\nabla{\mathbf{f}_{\mathbf{x}}}^T(k) + \mathbf{Q}(k).\end{aligned}\tag{77}$$

As seen, the prediction step of the regular and extended Kalman Filters are quite similar. The only difference is replacing the linear **F** matrix with nonlinear **f** function in the state prediction and substituting it with the gradient of **f** in the covariance prediction.

This is also the case in the measurement update part of the Kalman Filter. The EKF equations can be obtained by similar substitutions, given as

$$\mathbf{y}(k) = \mathbf{z}(k) - \mathbf{h}(\hat{\mathbf{x}}(k\,|k-1)) \tag{78}$$

$$\mathbf{S}(k) = \nabla\mathbf{h}_{\mathbf{x}}(k)\mathbf{P}(k\,|\,k-1)\nabla{\mathbf{h}_{\mathbf{x}}}^T(k) + \mathbf{R}(k) \tag{79}$$

$$\mathbf{K}(k) = \mathbf{P}(k\,|\,k-1)\,\nabla{\mathbf{h}_{\mathbf{x}}}^T(k)\,\mathbf{S}^{-1}(k) \tag{80}$$

$$\hat{\mathbf{x}}(k\,|\,k) = \hat{\mathbf{x}}(k\,|k-1) + \mathbf{K}(k)\mathbf{y}(k) \tag{81}$$

$$\mathbf{P}(k\,|\,k) = \left(\mathbf{I} - \mathbf{K}(k)\nabla\mathbf{h}_{\mathbf{x}}(k)\right)\mathbf{P}(k\,|\,k-1). \tag{82}$$

Even though KF and EKF operate with the same *predict* and *update* cycle with similar equations, it should be noted that unlike KF, EKF is not an optimal estimator. Due to the nonlinearity of the system, EKF estimates do not converge to the optimal MMSE, or may diverge. Therefore, when using Extended Kalman Filter, it should be verified that the first order Taylor series is a reasonable approximation of the system dynamics, and the system and error models should be constructed carefully.

4 Localization System

This chapter describes the localization system developed in this thesis. An Extended Kalman Filter described in Chapter 3 is utilized as the sensor fusion method to make use of different kinds of sensors.

4.1 Prerequisites and Assumptions

This section summarizes the prerequisite conditions for the developed localization system. They include assumptions about sensor characteristics as well as existing methods that complement sensor fusion for AGV localization.

As explained in Section 2.4, laser scanners provide distance measurements that are converted into position and orientation measurements with scan matching methods. For the developed sensor fusion, such a method is assumed to be existing and providing the Kalman Filter with direct position and orientation outputs. Furthermore, the noise characteristic of this output is assumed to be accurately approximated as a Gaussian white noise, such that it does not contain outliers or colored noise components.

Similarly, odometry is assumed to be calibrated such that there is no scaling error in the measurements and only Gaussian white noise is present in its output. However, for the IMU, the developed localization system takes into account that its outputs are corrupted with biases and gravity effects in addition to white noise, so there are no assumptions made about its outputs.

Finally, as the Kalman Filter needs an initial state estimate in the beginning of the algorithm, an initial position and orientation estimate of the AGV is assumed to be available. In addition, the vehicle is considered to be stationary in the start of its motions, such that initializing all velocity and acceleration states to zero is a reasonable approximation.

4.2 Kalman Filter Architecture

The general architecture of the implemented Kalman Filter can be observed in Figure 7. As seen there, each sensor provides measurements in their respective coordinate frames, and those measurements are used in the update step of the Kalman Filter. The prediction-update cycle of the filter is run every time a measurement arrives with an identical prediction step but different update steps specific to each sensor, which enables handling of asynchronous measurements. The prediction step is handled with general kinematic equations which are presented in Section 4.4. It should be noted that in the existing navigation system, laser scanner measurements are converted to absolute position and orientation estimates using scan matching techniques presented in Chapter 2 using odometry and its own outputs as shown with red in Figure 7. Therefore, the output it provides is considered independent of the solution implemented in this thesis, and the sensor fusion algorithm makes use of the outputs of the existing system as laser measurements.

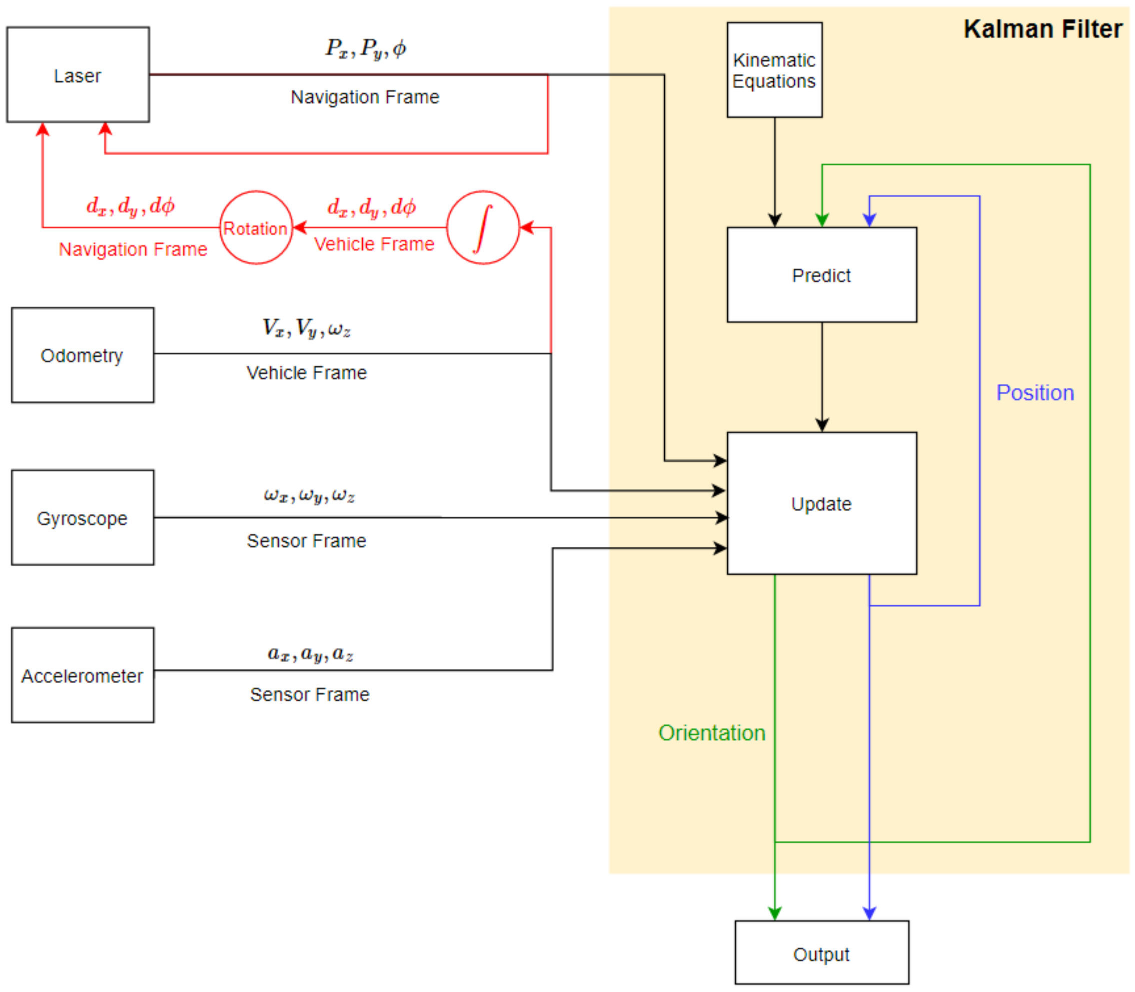


Figure 7: The general architecture of Kalman Filter solution. The red color at the left side indicates the existing scan matching method.

4.3 State Model

The localization system is designed to track the position and heading of the vehicle in a 2D map. Therefore, the minimum state vector of the Kalman filter should include those quantities as

$$\mathbf{x} = \begin{bmatrix} p_x \\ p_y \\ \phi \end{bmatrix} \tag{83}$$

where p_x and p_y denote the vehicle position in the navigation frame and ϕ is the heading. The system includes velocity, angular velocity and acceleration measurements from odometry and IMU, and to be able to utilize them in the update step of the filter without first integrating them, the state vector is extended as follows:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{h} \\ \omega \\ \mathbf{a} \end{bmatrix}. \tag{84}$$

Here, the state vectors are grouped for brevity of notation as $\mathbf{p} = [p_x \; p_y]^T$, $\mathbf{v} = [v_x \; v_y]^T$, $\mathbf{h} = [\psi \; \theta \; \phi]^T$, $\omega = [\omega_x \; \omega_y \; \omega_z]^T$ and $\mathbf{a} = [a_x \; a_y]^T$, which lets the state be a 3D representation of orientation and angular velocities and 2D representation of linear position, velocity and accelerations. The third axis of the latter is ignored as it is assumed the vehicle cannot move in the z axis within the scope of AGV applications considered in this thesis.

In the state vector, the position and orientation states $\mathbf{p}$ and $\mathbf{h}$ are defined in the navigation frame, while the other states are defined in the vehicle frame. This convention allows the usage of matrices in Equations 7 and 9 directly, and results in a nonlinear prediction model but linear measurement models.

Furthermore, as introduced in Section 2.3, IMU sensor measurements have a slowly varying bias which cannot be modelled as white noise as seen in Equations 26 and 29, rewritten here for convenience:

$$\omega_m(t) = \omega_i(t) + \mathbf{b}(t) + \epsilon(t) \tag{85}$$

$$\mathbf{a}_m(t) = \mathbf{f}_i(t) + \mathbf{b}(t) + \epsilon(t). \tag{86}$$

Kalman Filter will not provide optimal estimate if these biased measurements are directly used as input to the filter. A common way of handling this bias is called *state augmentation* that adds the bias terms for each measurement to the state vector, so that the measurements can be expressed in a linear combination of state variables and a white noise component [56][57]. This makes the final state model as

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{h} \\ \omega \\ \mathbf{a} \\ \delta b \end{bmatrix} \tag{87}$$

where the bias vector is $\delta b = [\delta a_x \; \delta a_y \; \delta \omega_x \; \delta \omega_y \; \delta \omega_z]^T$. Therefore, the state vector becomes a 17×1 vector, which is a relatively large vector but not large enough to cause computational speed problems with the available hardware presented in Chapter 5.

It should be noted that the yaw, pitch and roll angles of the state are expressed in Euler angles in radians, which have the equivalence relation $\phi = \phi + 2k\pi$ for all $k \in \mathbb{Z}$. Therefore, in each iteration of the filter, the yaw angle is clamped between π and

$-\pi$ for convenience. While this is also true for roll and pitch angles, practically they never come close to those boundaries in ground applications, so the same clamping is not needed for them.

Once the state vector is chosen, and initial state $\mathbf{x}(0)$ and its respective covariance matrix $\mathbf{P}(0)$ should be made available before the start of Kalman filter algorithm. Their numerical values depend on the initial position of the robot and how much confidence is given to the initialization system. However, one important thing to note while designing $\mathbf{P}(0)$ matrix is that the bias state covariances should have a higher initial value compared to their corresponding navigation states. That way, at the start of the navigation where the AGV is in a stationary position, readings from those biased sensors will change the bias estimations more than the state estimations, resulting in both less bias drift and a more accurate bias estimation. The exact numerical values used in each experimental case is presented later in Chapter 5.

4.4 Prediction Model

In the prediction step of the developed Kalman Filter, the highest order terms are modelled as random walk processes, that is, they propagate to next time step with equations in the form of

$$\begin{aligned} \mathbf{a}(k) &= \mathbf{a}(k-1) + \mathbf{w}_a \\ \omega(k) &= \omega(k-1) + \mathbf{w}_\omega \\ \delta b(k) &= \delta b(k-1) + \mathbf{w}_{\delta b} \end{aligned} \tag{88}$$

where $\mathbf{w}$ is a white noise vector corresponding to each element of the state. For the lower order terms, kinematic equations of the navigation are utilized to predict their state in the next time instant. In some applications in the literature, equations specific to vehicle type are used as they provide a more reliable prediction model [10], but the solution developed in thesis is intended to work regardless of vehicle type, so more general and basic kinematic equations are used, which are presented as follows:

$$\begin{aligned} \mathbf{v}(k) &= \mathbf{v}(k-1) + \mathbf{a}(k-1)\,\Delta t \\ \mathbf{p}(k) &= \mathbf{p}(k-1) + \mathbf{v}(k-1)\,\Delta t + \mathbf{a}(k-1)\,\Delta t^2/2 \\ \mathbf{h}(k) &= \mathbf{h}(k-1) + \omega(k-1)\,\Delta t \end{aligned} \tag{89}$$

where Δt is the time difference between the current and previous timesteps. It should be noted that the above equations hold only in the case of all quantities being defined in the same coordinate frame. However, in the state vector, linear and angular velocities and linear accelerations are defined in the vehicle frame while position and orientation are defined in the navigation frame. The true kinematic equations for orientation in this case becomes

$$\begin{bmatrix} \psi(k) \\ \theta(k) \\ \phi(k) \end{bmatrix} = \begin{bmatrix} \psi(k-1) \\ \theta(k-1) \\ \phi(k-1) \end{bmatrix} + \mathbf{R}_{ang} \begin{bmatrix} \omega_x(k-1) \\ \omega_y(k-1) \\ \omega_z(k-1) \end{bmatrix} \tag{90}$$

where

$$\mathbf{R}_{ang} = \begin{bmatrix} 1 & tan(\theta(k-1))sin(\psi(k-1)) & tan(\theta(k-1))cos(\psi(k-1)) \\ 0 & cos(\psi(k-1)) & -sin(\psi(k-1)) \\ 0 & sin(\psi(k-1))/cos(\theta(k-1)) & cos(\psi(k-1))/cos(\theta(k-1)) \end{bmatrix}. \quad (91)$$

The kinematic equation for the velocity will be unchanged as the acceleration and velocity are defined in the same coordinate frame, but the position update equation becomes

$$\mathbf{p}(k) = \mathbf{p}(k-1) + \Delta t\,(\mathbf{R}_{lin}\,\mathbf{v}(k-1)) + \Delta t^2\,(\mathbf{R}_{lin}\,\mathbf{a}(k-1))/2 \quad (92)$$

where

$$\mathbf{R}_{lin} = \begin{bmatrix} cos(\phi)cos(\theta) & cos(\phi)sin(\theta)sin(\psi) - sin(\phi)cos(\psi) \\ sin(\phi)cos(\theta) & sin(\phi)sin(\theta)sin(\psi) + cos(\phi)cos(\psi) \end{bmatrix}. \quad (93)$$

To sum up all the prediction equations in the form of EKF equation as

$$\mathbf{x}(k) = \mathbf{f}\,(\mathbf{x}(k-1), \mathbf{u}(k), k) + \mathbf{w}(k), \quad (94)$$

in the absence of any control input, the kinematic equations define the nonlinear function $\mathbf{f}$ as follows:

$$\mathbf{x}(k) = \begin{bmatrix} \mathbf{p}(k) \\ \mathbf{v}(k) \\ \mathbf{h}(k) \\ \omega(k) \\ \mathbf{a}(k) \\ \delta b(k) \end{bmatrix} = \begin{bmatrix} \mathbf{p}(k-1) + \Delta t\,(\mathbf{R}_{lin}\,\mathbf{v}(k-1)) + \Delta t^2\,(\mathbf{R}_{lin}\,\mathbf{a}(k-1))/2 \\ \mathbf{v}(k-1) + \Delta t\,\mathbf{a}(k-1) \\ \mathbf{h}(k-1) + \mathbf{R}_{ang}\,\Delta t\,\omega(k-1) \\ \omega(k-1) \\ \mathbf{a}(k-1) \\ \delta b(k-1) \end{bmatrix} + \mathbf{w}(k). \quad (95)$$

With that, the state update of predict step of the EKF equation can be performed as:

$$\hat{\mathbf{x}}(k\,|\,k-1) = \mathbf{f}\,(\hat{\mathbf{x}}(k-1\,|\,k-1), k). \quad (96)$$

For the covariance update, the following equation that is defined in Chapter 3 is used:

$$\mathbf{P}(k\,|\,k-1) = \mathbf{F}(k)\,\mathbf{P}(k-1\,|k-1)\mathbf{F}^T(k) + \mathbf{Q}(k). \quad (97)$$

Here, $\mathbf{F}(k)$ is the 17×17 Jacobian matrix that is obtained by taking the derivatives of each row of $\mathbf{f}$ function with respect to each element of the state vector, as follows:

$$\mathbf{F} = \frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f(1)}{\partial p_x} & \frac{\partial f(1)}{\partial p_y} & \frac{\partial f(1)}{\partial v_x} & \cdots & \frac{\partial f(1)}{\partial \delta\omega_x} & \frac{\partial f(1)}{\partial \delta\omega_y} & \frac{\partial f(1)}{\partial \delta\omega_z} \\ \frac{\partial f(2)}{\partial p_x} & \frac{\partial f(2)}{\partial p_y} & \frac{\partial f(2)}{\partial v_x} & \cdots & \frac{\partial f(2)}{\partial \delta\omega_x} & \frac{\partial f(2)}{\partial \delta\omega_y} & \frac{\partial f(2)}{\partial \delta\omega_z} \\ \frac{\partial f(3)}{\partial p_x} & \frac{\partial f(3)}{\partial p_y} & \frac{\partial f(3)}{\partial v_x} & \cdots & \frac{\partial f(3)}{\partial \delta\omega_x} & \frac{\partial f(3)}{\partial \delta\omega_y} & \frac{\partial f(3)}{\partial \delta\omega_z} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \frac{\partial f(15)}{\partial p_x} & \frac{\partial f(15)}{\partial p_y} & \frac{\partial f(15)}{\partial v_x} & \cdots & \frac{\partial f(15)}{\partial \delta\omega_x} & \frac{\partial f(15)}{\partial \delta\omega_y} & \frac{\partial f(15)}{\partial \delta\omega_z} \\ \frac{\partial f(16)}{\partial p_x} & \frac{\partial f(16)}{\partial p_y} & \frac{\partial f(16)}{\partial v_x} & \cdots & \frac{\partial f(16)}{\partial \delta\omega_x} & \frac{\partial f(16)}{\partial \delta\omega_y} & \frac{\partial f(16)}{\partial \delta\omega_z} \\ \frac{\partial f(17)}{\partial p_x} & \frac{\partial f(17)}{\partial p_y} & \frac{\partial f(17)}{\partial v_x} & \cdots & \frac{\partial f(17)}{\partial \delta\omega_x} & \frac{\partial f(17)}{\partial \delta\omega_y} & \frac{\partial f(17)}{\partial \delta\omega_z} \end{bmatrix} \quad (98)$$

where $f(n)$ denotes the n^{th} row of function $\mathbf{f}$.

To design the $\mathbf{Q}$ matrix, the *piecewise white noise model* presented in [58, p.243-245] is used, where the highest order terms are assumed to stay constant in each time step with an associated Gaussian white process noise, same as in Equation 88, and the process noise of lower order terms have a relation with different orders of time with them. This relation is expressed with with the *noise gain matrix* denoted with Γ, such that

$$\mathbf{Q}(k) = E(\mathbf{\Gamma}(k)\mathbf{w}(k)\mathbf{w}^T(k)\mathbf{\Gamma}^T(k)). \tag{99}$$

To determine Γ, it is useful to break up the state in smaller parts. From Equation 95, position and velocity have the following relation with acceleration:

$$\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{p}(k-1) + \Delta t\,(\mathbf{R}_{lin}\,\mathbf{v}(k-1)) + \Delta t^2\,(\mathbf{R}_{lin}\,\mathbf{a}(k-1))/2 \\ \mathbf{v}(k-1) + \Delta t\,\mathbf{a}(k-1) \\ \mathbf{a}(k-1) \end{bmatrix}. \tag{100}$$

This follows that the noise $\mathbf{w}_a = [w_{ax}\; w_{ay}]^T$ in accelerations will affect these states with

$$\Gamma_1 = \begin{bmatrix} \mathbf{R}_{lin}\,\Delta t^2/2 \\ \Delta t \\ 1 \end{bmatrix}. \tag{101}$$

As the accelerations in different axes are considered independent, the following relation will hold

$$E(\mathbf{w}_a\mathbf{w}_a^T) = \sigma_a^2 = \begin{bmatrix} \sigma_{ax}^2 & 0 \\ 0 & \sigma_{ay}^2 \end{bmatrix} \tag{102}$$

where σ_{ax}^2 and σ_{ay}^2 denote the variance of each noise component. Following Equation 99, and taking into account that $\sigma_a^2 = \sigma_a^{2^T}$, the $\mathbf{Q}$ matrix corresponding to those states are calculated as

$$\mathbf{Q}_1 = E(\mathbf{\Gamma}\mathbf{w}_a\mathbf{w}_a^T\mathbf{\Gamma}^T(k)) = \begin{bmatrix} \mathbf{R}_{lin}\,\sigma_a^2\,\mathbf{R}_{lin}^T\Delta t^4/4 & \mathbf{R}_{lin}\,\sigma_a^2\Delta t^3/2 & \mathbf{R}_{lin}\,\sigma_a^2\Delta t^2/2 \\ \sigma_a^2\,\mathbf{R}_{lin}^T\,\Delta t^3/2 & \sigma_a^2\Delta t^2/2 & \sigma_a^2\Delta t \\ \sigma_a^2\,\mathbf{R}_{lin}^T\,\Delta t^2/2 & \sigma_a^2\Delta t & \sigma_a^2 \end{bmatrix}. \tag{103}$$

Similarly, for the orientation states, assuming the angular velocities have a variance of

$$\sigma_\omega^2 = \begin{bmatrix} \sigma_{\omega x}^2 & 0 & 0 \\ 0 & \sigma_{\omega y}^2 & 0 \\ 0 & 0 & \sigma_{\omega z}^2 \end{bmatrix} \tag{104}$$

the corresponding part of $\mathbf{Q}$ can be found as

$$\mathbf{Q}_2 = \begin{bmatrix} \mathbf{R}_{ang}\,\sigma_\omega^2\,\mathbf{R}_{ang}^T\Delta t^2 & \mathbf{R}_{ang}\,\sigma_\omega^2\Delta t \\ \sigma_\omega^2\,\mathbf{R}_{ang}^T\Delta t & \sigma_\omega^2 \end{bmatrix}. \tag{105}$$

Finally, as the biases are considered independent from other states, their process noise matrix is a diagonal matrix with the variance of each element on the diagonal, such that

$$\mathbf{Q}_3 = \sigma_\delta^2 = \begin{bmatrix} \sigma_{\delta_{ax}}^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{\delta_{ay}}^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\delta_{\omega x}}^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\delta_{\omega y}}^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\delta_{\omega z}}^2 \end{bmatrix}. \tag{106}$$

Combining all 3 matrices results in a $\mathbf{Q}$ matrix of the following form:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0}_{6\times6} & \mathbf{0}_{6\times5} \\ \mathbf{0}_{6\times6} & \mathbf{Q}_2 & \mathbf{0}_{6\times5} \\ \mathbf{0}_{5\times6} & \mathbf{0}_{5\times6} & \mathbf{Q}_3 \end{bmatrix} \tag{107}$$

It should be noted that the $\mathbf{Q}$ matrix defined above holds when the states are ordered as $[\mathbf{p}, \mathbf{v}, \mathbf{a}, \mathbf{h}, \omega, \delta\mathbf{b}]^T$ and elements should be reordered with the state matrix defined in Equation 87.

Furthermore, $\mathbf{Q}$ defined this way is an approximation, as the rotation matrices included are dependent on the states themselves, it implies there is a correlation between the position and orientation as well. However, highly performing implementations have been achieved despite ignoring noise terms in $\mathbf{Q}$ matrix except for highest order terms [58, p.246] and control inputs [26]. Therefore, this approximation is decided to be reasonable within the scope of this thesis.

After all the matrices are designed, the prediction step of the EKF can be performed with Equations 96 and 97.

4.5 Measurement Models

Each sensor has its own measurement model in the Kalman Filter implementation, which are presented throughout this section.

4.5.1 Laser Measurement Model

The laser measurements consist of position and heading in the navigation frame, such that

$$\mathbf{z}_{laser} = \begin{bmatrix} p_x \\ p_y \\ \phi \end{bmatrix} \tag{108}$$

with a corresponding covariance matrix

$$\mathbf{R}_{laser} = \begin{bmatrix} \sigma_{px}^2 & 0 & 0 \\ 0 & \sigma_{py}^2 & 0 \\ 0 & 0 & \sigma_\phi^2 \end{bmatrix}. \tag{109}$$

It is straightforward to derive the linear observation matrix as

$$\mathbf{H}_{laser} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & \mathbf{0}_{3\times 10} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \end{bmatrix} \tag{110}$$

where $\mathbf{0}_{3\times 10}$ is a 3×10 zero matrix. When a laser measurement arrives to the system, update step of the Kalman Filter is performed with the following equations derived in Chapter 3:

$$\begin{aligned} \mathbf{y}(k) &= \mathbf{z}(k) - \mathbf{H}(k)\,\hat{\mathbf{x}}(k \mid k-1) \\ \mathbf{S}(k) &= \mathbf{H}(k)\mathbf{P}(k \mid k-1)\mathbf{H}^T(k) + \mathbf{R}(k) \\ \mathbf{K}(k) &= \mathbf{P}(k \mid k-1)\,\mathbf{H}^T(k)\,\mathbf{S}^{-1}(k) \\ \hat{\mathbf{x}}(k \mid k) &= \hat{\mathbf{x}}(k \mid k-1) + \mathbf{K}(k)\,\mathbf{y}(k) \\ \mathbf{P}(k \mid k) &= (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,\mathbf{P}(k \mid k-1). \end{aligned} \tag{111}$$

In the measurement update of the laser scanner, the update of the heading should be specially noted. As explained in Section 4.3, the state variable for the heading is clamped between $-\pi$ and π radians, or -180 to 180 degrees. Around those borders, the state jumps by 360 degrees. For example, a change from 179 to -179 degrees is actually a 2 degree turn in the positive direction, but is perceived by a 358 degrees turn in the negative direction by Equation 111, which leads to incorrect measurement correction by the filter. This is prevented by adjusting the innovation $\mathbf{y}(k)$ around those borders as follows:

$$y_3(k) > \pi \implies y_3(k) = y_3(k) - 2\pi \tag{112}$$

$$y_3(k) < \pi \implies y_3(k) = y_3(k) + 2\pi. \tag{113}$$

Here, $y_3(k)$ denotes the third element of the innovation vector, which in this case corresponds to heading innovation. $\pm\pi$ is chosen as the limit of correction, so that the innovation will always be calculated as the smallest rotation between two timesteps. As the only sensor providing absolute measurements of heading, this step is only necessary in the laser measurement update.

4.5.2 Odometry Measurement Model

The odometry provides linear velocities in the vehicle frame, and angular velocity of heading in the vehicle frame, such that

$$\mathbf{z}_{odom} = \begin{bmatrix} v_x \\ v_y \\ \omega_z \end{bmatrix} \tag{114}$$

with a corresponding covariance matrix

$$\mathbf{R}_{odom} = \begin{bmatrix} \sigma_{vx}^2 & 0 & 0 \\ 0 & \sigma_{vy}^2 & 0 \\ 0 & 0 & \sigma_{\omega z}^2 \end{bmatrix}. \tag{115}$$

The linear observation matrix is obtained as

$$\mathbf{H}_{odom} = \begin{bmatrix} 0 & 0 & 1 & 0 & & 0 & \\ 0 & 0 & 0 & 1 & \mathbf{0}_{3\times5} & 0 & \mathbf{0}_{3\times7} \\ 0 & 0 & 0 & 0 & & 1 & \end{bmatrix}. \tag{116}$$

The update step when an odometry measurement arrives is then performed with the same equations presented in Equation 111.

4.5.3 Inertial Measurement Unit Measurement Model

The IMU provides linear acceleration and angular velocity measurements in the sensor frame. Within the scope of this thesis, the IMU is considered to be aligned with the vehicle, so that the IMU frame and the vehicle frame are related only with a translation matrix. This lets the IMU measurements in the vehicle frame take the form of

$$\mathbf{z}_{imu} = \begin{bmatrix} a_x \\ a_y \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{117}$$

Here, it should be noted that the 6th measurement a_z is not included in the measurement matrix as the linear movement is considered to be zero in the z axis. In addition, the translation between the vehicle and IMU frames will result in the IMU accelerating due to angular rotation of the vehicle that further affects the output of the accelerometer. This effect is proportional to the distance between two frames, as well as the magnitude of the angular acceleration. It is assumed within the normal operating conditions of AGVs considered in this thesis, this effect will be minimal compared to the white noise, bias and gravity effect of accelerometer, so that it is approximated to zero.

Furthermore, as explained in Section 2.3.2, the accelerometer can also be used to estimate roll and pitch angles by using Equations 33 and 34, repeated below:

$$\psi = arctan(\frac{a_y}{a_z}) \tag{118}$$

$$\theta = arctan(\frac{-a_x}{\sqrt{a_y^2 + a_z^2}}). \tag{119}$$

This leads to the full IMU measurement vector in the form of

$$\mathbf{z}_{imu} = \begin{bmatrix} a_x \\ a_y \\ \omega_x \\ \omega_y \\ \omega_z \\ \psi_{IMU} \\ \theta_{IMU} \end{bmatrix} \tag{120}$$

where ψ_{IMU} and θ_{IMU} are the results of Equations 118 and 119. The IMU has a corresponding covariance matrix

$$\mathbf{R}_{imu} = \begin{bmatrix} \sigma_{ax}^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{ay}^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\omega x}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\omega y}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\omega z}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\psi}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{\theta}^2 \end{bmatrix}. \tag{121}$$

It should be noted here that the last two elements of this matrix σ_ψ^2 and σ_θ^2 correspond to the process noise of roll and pitch measurements, hence, they do not represent a physical sensor noise variance, but the variance of the noise expected of Equations 118 and 119. As they are less reliable in situations with high linear acceleration, they are generally set to comparatively higher values than other elements of the matrix.

As presented in the previous sections, IMU measurements include a bias term in addition to the white noise components, which are also estimated with the Kalman Filter. This leads to an observation matrix as follows:

$$\mathbf{H}_{imu} = \left[\begin{array}{c|ccccccccccccc} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \mathbf{0}_{7\times4} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]. \tag{122}$$

In addition to the white noise and bias, accelerometer measurements are further corrupted with the affects of the gravity vector as presented in Equation 32, repeated below:

$$\mathbf{a}_m(t) = \mathbf{R}_n^i(\mathbf{a}_n(t) - \mathbf{g}_n) + \mathbf{b}(t) + \epsilon(t). \tag{123}$$

Therefore, before being used in the filter, the gravity component should be removed from the measurements with the estimated orientation, such that

$$\mathbf{z}_{acc} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} a_x(measured) \\ a_y(measured) \\ a_z(measured) \end{bmatrix} - \mathbf{R}_{lin}^T \mathbf{g} \tag{124}$$

where $\mathbf{g}$ is the global gravity vector $[0,\ 0,\ \sim 9.81\,m/s^2]^T$ and $\mathbf{R}_{lin}^T$ is the transpose of the linear rotation matrix, which is used to transform the gravity into vehicle frame.

Once the measurement vector and observation matrix is determined, the measurement update with the IMU is performed with the same equations as Equation 111.

4.6 Additional Features

The Kalman Filter model presented so far constitutes the main method of implemented sensor fusion. This section explains additional features that are implemented to be optionally used depending on the operational needs of the AGV localization.

4.6.1 Robust Laser Measurement Model

While Kalman Filter is a powerful algorithm for sensor fusion that can reliably filter Gaussian noise, it is vulnerable to presence of outlier measurements. These outliers are especially likely to occur with laser scanner measurements, due to unaccounted reflective surfaces in the environment as explained in Chapter 2. In the literature, there are several methods of dealing with outliers in the Kalman Filter itself, most of them relying on rejecting the measurements that are classified as outliers [53].

One of those methods is the recently developed *IS-EKF* by Fang et al. [59]. While specifically developed for EKF, the main working principles of their proposed method is also applicable for linear measurements such as laser scanner, which relies on clipping the innovation $\mathbf{y}$ such that the measurement update algorithm becomes

$$\begin{aligned}
\mathbf{y}(k) &= \mathbf{z}(k) - \mathbf{H}(k)\,\hat{\mathbf{x}}(k\,|\,k-1) \\
\mathbf{S}(k) &= \mathbf{H}(k)\mathbf{P}(k\,|\,k-1)\mathbf{H}^T(k) + \mathbf{R}(k) \\
\mathbf{K}(k) &= \mathbf{P}(k\,|\,k-1)\,\mathbf{H}^T(k)\,\mathbf{S}^{-1}(k) \\
\hat{\mathbf{x}}(k\,|\,k) &= \hat{\mathbf{x}}(k\,|\,k-1) + \mathbf{K}(k)\,\mathbf{y}_{sat} \\
\mathbf{P}(k\,|\,k) &= (\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k))\,\mathbf{P}(k\,|\,k-1)
\end{aligned} \tag{125}$$

where $\mathbf{y}_{sat}$ is the saturated innovation vector. Each element of $\mathbf{y}_{sat}$ defined as

$$y_{sat_i} = max\{-\sigma_i(k), min\{\sigma_i(k), y_i\}\}. \tag{126}$$

where $\sigma_i(k)$ is the saturation limit for that measurement. Therefore, the core idea of this robust version of Kalman Filter is that if the laser provides an outlier measurement $\mathbf{z}$ such that for any element of $\mathbf{y}$, $y_i > \sigma_i$ or $y_i < -\sigma_i$, the innovation of that corresponding element is limited to σ_i. This prevents outlier measurements from diverging the Kalman Filter estimates. Furthermore, as explained in detail in [59], the innovation bounds are not static, but they are updated in each time step as

$$\sigma_i(k+1) = \lambda_{1i}\,\sigma_i(k) + \gamma_{1i}\,\epsilon_i(k)\,e^{-\epsilon_i(k)} \tag{127}$$

$$\epsilon_i(k+1) = \lambda_{2i}\,\epsilon_i(k) + \gamma_{2i}\,y_i^2 \tag{128}$$

where $\lambda_1, \lambda_2, \gamma_1, \gamma_2$ are parameters that should be defined for each measurement. ϵ is a variable that affects how much the saturation boundary is changed in each time step, which is updated in every cycle as well. This ensures that the filter adaptively decides whether each measurement is an outlier, so in turn whether to saturate the innovation in measurement update or not.

While this is an effective method to deal with outlier measurements occurring in laser scanners, the parameters $\lambda_1, \lambda_2, \gamma_1, \gamma_2$ for each measurement and the initial values of $\sigma(0)$ and $\epsilon(0)$ have to be defined properly for high performance of the algorithm. With the guidelines presented in [59] and trial-and error in controlled environments this is not a hard task, however, this is not always possible to accurately estimate for an unknown AGV in an unknown working environment. Therefore, this robust version of laser updates are optionally implemented in the sensor fusion algorithm to be used if they can be tuned, or normal laser measurement updates can be used otherwise.

4.6.2 Stationary Measurements

As presented in their relative sections, gyroscope and acceleration measurements are subject to drift during longer time periods in the absence of absolute measurements such as laser scanner. One way to reduce this drift is to use *Zero Velocity Update (ZUPT)* and *Zero Angular Rotation Update (ZARU)* [60]. This method consists of providing linear velocity and angular velocity updates of zero mean and a low variance in time periods where the vehicle is known to be stationary, such that

$$\mathbf{z} = \begin{bmatrix} v_x \\ v_y \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} . \tag{129}$$

This prevents the position and heading estimates of the vehicle from drifting when stationary. However, in the literature, these updates are usually used for foot mounted positioning tracking devices [61][62] rather than AGV localization. The main reason for that is the odometry measurements can already provide updates in the form of Equation 129 for heading and velocity and the accelerometer can prevent the drift of roll and pitch estimates with Equation 33 in stationary time periods.

However, in some practical cases, odometry readings may not be reliable for either linear or angular velocity, and if that is known, the corresponding $\mathbf{R}_{odom}$ matrix is set to larger values for sensor fusion to prevent it from affecting position or orientation estimations negatively. In such a case, even though odometry itself would be providing ZUPT and ZARU updates, they would be disregarded as unreliable. Hence, a separate ZUPT and ZARU update cycle base on Equation 129 is implemented optionally to be used for such cases.

4.6.3 Simplified 2D Model

In AGV applications where the operating surface is horizontal without ramps or cliffs, roll and pitch angles of the AGV is expected to be quite small during normal operation and can be considered zero all the time. This additional information about AGV orientation can be utilized to simplify state estimation to a 2D model which was implemented as an alternative. While having the same working principle, it has the following modifications:

- The state vector is reduced to 12×1 as in the 2D model $\mathbf{h}(k) = \phi(k)$, $\omega(k) = \omega_z(k)$, and $\delta b(k) = [\delta a_x \; \delta a_y \; \delta\omega_z]^T$.
- IMU measurements of roll and pitch rate are not considered, so the reduced measurement vector becomes

$$\mathbf{z}_{IMU} = \begin{bmatrix} a_x \\ a_y \\ \omega_z \end{bmatrix}. \tag{130}$$

- The angular velocity rotation matrix $\mathbf{R}_{ang}$ is reduced to identity equation, as in the absence of roll and pitch $\dot{\phi} = \omega_z$.
- Similarly, linear rotation matrix $\mathbf{R}_{lin}$ can be expressed in 2D as

$$\mathbf{R}_{lin} = \begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix}. \tag{131}$$

This results in a simpler and efficient model that can be used in place of the 3D model in the environments where aforementioned assumptions about AGV orientation are valid. However, in situations where 3D estimation is necessary, for example in a navigation environment with slopes or ramps, it will not be as reliable as the 3D model. Therefore, this 2D model is implemented optionally, together with the 3D sensor fusion model.

4.7 Handling Asynchronous Measurements

Each sensor that is used for localization has different time intervals for providing measurements. Generally, both odometry and IMU can provide readings in regular time intervals, however, IMUs usually perform with a much higher frequency. Even though lasers also scan the environment regularly, there might be situations where the scans cannot provide a location estimate, such as due to lack of sufficient features, that force laser scanner localization to be an irregular process. To be able to handle both regular measurements with different frequencies and irregular measurements, predict and update steps of the Kalman Filter are performed whenever a measurement is available.

This method leads to a varying time interval between timesteps k and $k+1$ in each KF cycle, hence, the Δt variable in Equation 95 has to be recalculated. This leads to $\mathbf{F}$ and $\mathbf{Q}$ being interval dependent matrices that must also be recalculated in each prediction step. Therefore, the developed Kalman Filter is run according to Algorithm 1 presented below.

Algorithm 1 Kalman Filter for Localization

```
$\mathbf{x}(0)$ : Initial state
$\mathbf{P}(0)$ : Initial covariance matrix
$t_{prev} \leftarrow 0$ : Initial timestamp
$k \leftarrow 1$ : Initial step counter
while true do
    if measurement arrived then
        $\Delta t \leftarrow t_{measurement} - t_{prev}$
        $t_{prev} \leftarrow t_{measurement}$
        $\mathbf{F} \leftarrow \textsc{calculate\_F}(\Delta t)$
        $\mathbf{Q} \leftarrow \textsc{calculate\_Q}(\Delta t)$
        $\mathbf{x}(k-1)$, $\mathbf{P}(k-1) \leftarrow \textsc{Predict}(\mathbf{x}(k-1), \mathbf{P}(k-1), \mathbf{F}, \mathbf{Q}, \Delta t)$
        $\mathbf{x}(k)$, $\mathbf{P}(k) \leftarrow \mathrm{SENSOR\_UPDATE}(\mathbf{x}(k-1), \mathbf{P}(k-1), \mathbf{R}_{Sensor})$
        $k \leftarrow k + 1$
    end if
end while
```

5 Experiments and Results

This section describes the experiments conducted to evaluate the performance of the localization solution developed in Chapter 4 and presents the obtained results.

5.1 Validation of Filter Estimation

This set of experiments was conducted to study if the developed algorithm is able to provide correct position and heading estimates. To that end, an open source multisensor dataset with timestamped sensor measurements was used to predict the position of the AGV and compared against the provided ground truth position.

For this experiment, the *Rawseeds Project* multisensor datasets [63][64] were used. Rawseeds project consists of indoor, mixed and outdoor datasets that include measurements from many different sensors with associated timestamps. Furthermore, a reliable ground truth position is provided to be able to compare the localization results against it. From the available datasets, indoor datasets were preferred as the scope of this thesis does not include the GPS measurements generally prominent in outdoor datasets, but instead uses the laser scanners that are more reliable indoors. Ultimately, the indoor dataset *Bicocca-2009-02-25b* was chosen among the available indoor datasets. It includes the necessary IMU, odometry and laser measurements as described below, together with additional sensor data such as sonar and camera readings that are not used in this solution.

This sensor data was gathered during an indoor motion of the AGV in two university buildings that include hallways, libraries and offices. The motion took about 30 minutes from start to finish, and it includes many turns as well as straight segments. As the horizontal nature of the movement satisfies the conditions for the 2D model presented in Section 4.6.3, both the simplified 2D model and full 3D model of the developed localization algorithm were analyzed in each test in order to observe the performance difference of adding a 3D orientation estimate.

For performance analysis in these experiments, visual plots of the trajectories were compared with ground truth. Furthermore, *Absolute Trajectory Error (ATE)* and *Relative Pose Error (RPE)* performance metrics were computed in relevant experiments. ATE measures the linear difference between each point of ground truth with the corresponding point of the estimated trajectory at the same time instant. Mean and standard deviation values of ATE during the motion can indicate success of localization algorithm, with lower values suggesting a better algorithm.

RPE compares the trajectory of the ground truth to the estimated trajectory by calculating relative pose differences over a time period, therefore, a low RPE is an indicator of a lower drift [65]. The RPE at each time step i is computed as

$$\mathbf{E}_i = (\mathbf{Q}_i^{-1}\mathbf{Q}_{i+\Delta})^{-1}(\mathbf{P}_i^{-1}\mathbf{P}_{i+\Delta}) \tag{132}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are poses of the estimated trajectory and the ground truth and Δ is a fixed time interval. The RPE of the motion in this time interval Δ can then be calculated by taking the RMSE of translational and rotational components of these errors as

$$RMSE_{trans}(\mathbf{E}_{1:n}, \Delta) = (\frac{1}{m}\sum_{i=1}^{m} \|trans(\mathbf{E}_i)\|^2)^{1/2} \tag{133}$$

$$RMSE_{rot}(\mathbf{E}_{1:n}, \Delta) = (\frac{1}{m}\sum_{i=1}^{m} \|rot(\mathbf{E}_i)\|^2)^{1/2}. \tag{134}$$

To characterize the motion without being dependent on the chosen time interval Δ, the RPE is finally computed as taking the average of the calculated RMSEs in all possible Δ choices throughout the motion as

$$tRPE = \frac{1}{n}\sum_{\Delta=1}^{n} RMSE_{trans}(\mathbf{E}_{1:n}, \Delta) \tag{135}$$

$$rRPE = \frac{1}{n}\sum_{\Delta=1}^{n} RMSE_{rot}(\mathbf{E}_{1:n}, \Delta) \tag{136}$$

where $tRPE$ and $rRPE$ denote translational and rotational RPE, respectively.

5.1.1 Hardware and Sensor Setup

The Robocom platform [64] illustrated in Figure 8 was used to gather the navigation data in Rawseeds datasets. It is a differential drive AGV that is equipped with ultrasound transducers, camera systems, an Inertial Measurement Unit and laser range finders that provide timestamped data as the vehicle navigates in the environment. The utilized sensors on Robocom platform in this thesis are the following [64]:

- XSense MTi Inertial Measurement Unit
- Two Hokuyo URG-04LX laser range finders
- Sick LMS291 and LMS200 laser range finders
- An odometric system equipped with encoders in the wheel base

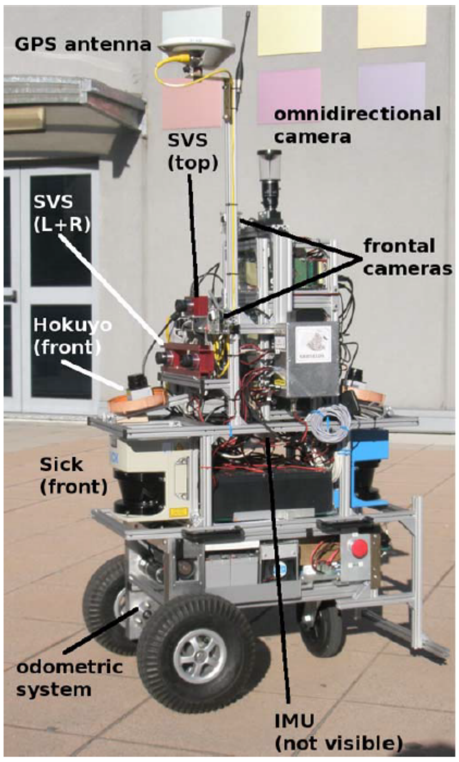


Figure 8: The Robocom platform used in generating the datasets, taken from [64].

5.1.2 Dataset Structure and Data Acquisition

The IMU measurements are provided in the following format:

$$\mathbf{z} = \begin{bmatrix} \text{Timestamp [seconds.microseconds]} \\ \text{Sample counter (modulo } 2^{16} \text{ -1) [unitless]} \\ \text{Acceleration along X } [m/s^2] \\ \text{Acceleration along Y } [m/s^2] \\ \text{Acceleration along Z } [m/s^2] \\ \text{Angular velocity around X [rad/s]} \\ \text{Angular velocity around Y [rad/s]} \\ \text{Angular velocity around Z [rad/s]} \\ \text{Earth's magnetic field along X} \\ \text{Earth's magnetic field along Y} \\ \text{Earth's magnetic field along Z} \\ \text{R1..R9 Orientation matrix, row after row [unitless]} \end{bmatrix}. \qquad (137)$$

From these measurements, only the accelerometer and gyroscope measurements were utilized. It is noted in the dataset that the IMU is aligned with the navigation point of the vehicle such that there is only translation between IMU and vehicle frames. Hence, measurements were used as explained in Section 4.5.3 to obtain the measurement vector in the form of

$$\mathbf{z}_{imu} = \begin{bmatrix} a_x \\ a_y \\ \omega_x \\ \omega_y \\ \omega_z \\ \psi_{IMU} \\ \theta_{IMU} \end{bmatrix}. \tag{138}$$

The odometry measurements are provided in the following format:

$$\mathbf{z} = \begin{bmatrix} \text{Timestamp [seconds.microseconds]} \\ \text{Rolling Counter [signed 16bit integer]} \\ \text{Ticks Right [ticks]} \\ \text{Ticks Left [ticks]} \\ \text{X [m]} \\ \text{Y [m]} \\ \phi \text{ [rad]} \end{bmatrix}. \tag{139}$$

It is noted that the provided X, Y, ϕ are given in the navigation frame, and they are the result of integration and rotation of pure odometry velocities in the vehicle frame. The sensor fusion algorithm needs the odometry measurements as instantaneous velocities in vehicle frame. Therefore, prior to using them in the sensor fusion as inputs, instantaneous velocities in navigation frame were computed from consecutive odometer measurements. Then, the obtained velocities were rotated back to the vehicle frame by using the inverse of the 2D rotation matrix with the odometry heading in each timestep, such that for i^{th} measurement the corresponding odometry measurement vector $\mathbf{z}_{odom}(i)$ was obtained with the following steps:

$$\Delta t(i) = timestamp(i) - timestamp(i-1) \tag{140}$$

$$\begin{bmatrix} v_{xnav}(i) \\ v_{ynav}(i) \\ \omega(i) \end{bmatrix} = \begin{bmatrix} x(i) - x(i-1) \\ y(i) - y(i-1) \\ \phi(i) - \phi(i-1) \end{bmatrix} / \Delta t(i) \tag{141}$$

$$\mathbf{R}_{lin}^{-1} = \mathbf{R}_{lin}^{T} = \begin{bmatrix} cos(\phi(i)) & sin(\phi(i)) \\ -sin(\phi(i)) & cos(\phi(i)) \end{bmatrix} \tag{142}$$

$$\begin{bmatrix} v_{xveh}(i) \\ v_{yveh}(i) \end{bmatrix} = \mathbf{R}_{lin}^{-1} \begin{bmatrix} v_{xnav}(i) \\ v_{ynav}(i) \end{bmatrix} \tag{143}$$

$$\mathbf{z}_{odom}(i) = \begin{bmatrix} v_{xnav}(i) \\ v_{ynav}(i) \\ \omega(i) \end{bmatrix}. \tag{144}$$

The laser measurements are provided in the following format for SICK Scanners

$$\mathbf{z} = \begin{bmatrix} \text{Timestamp [seconds.microseconds]} \\ \text{Number of ranges [unitless]} \\ \text{Angular offset [1/4 degree]} \\ \text{R1..R181 Ranges (zero padded to 181 ranges) [m]} \end{bmatrix} \tag{145}$$

and as follows for Hokuyo Scanners

$$\mathbf{z} = \begin{bmatrix} \text{Timestamp [seconds.microseconds]} \\ \text{R1..R681 Ranges [m]} \end{bmatrix}. \tag{146}$$

These raw measurements cannot be directly used in the developed sensor fusion algorithm. Instead, as explained in Section 2.4, map matching techniques must be applied in order to convert raw laser scanner range measurements into location and orientation in the map. For this purpose, *Benchmark Solutions* provided alongside sensor datasets by Rawseeds were used. Those solutions are results of map matching methods in the form of

$$\mathbf{z} = \begin{bmatrix} \text{Timestamp [seconds.microseconds]} \\ \text{x [m]} \\ \text{y [m]} \\ \phi \text{ [rad]} \end{bmatrix}. \tag{147}$$

They are provided with the intention of providing a benchmark to test other map matching methods against, or to be used in higher level algorithms. For all experiments with *Bicocca-2009-02-25b* dataset described in this chapter, "GraphSLAM" benchmark solution for this dataset was used as laser scanner measurements for sensor fusion. This way, the same architecture presented in Figure 7 was achieved and performance of the fusion can be analyzed.

5.1.3 State Initialization and Parameter Selection

For all tests with the Rawseeds dataset, the initial state $\mathbf{x}(0)$ was determined as follows:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{h} \\ \omega \\ \mathbf{a} \\ \delta b \end{bmatrix} = \begin{bmatrix} \mathbf{p}(0) \\ \mathbf{0}_{2\times1} \\ \mathbf{h}(0) \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_{2\times1} \\ \mathbf{0}_{5\times1} \end{bmatrix} \tag{148}$$

where the initial linear velocities, angular velocities, accelerations and bias estimates are set to zero. The initial position

$$\mathbf{p}(0) = \begin{bmatrix} p_x(0)_{gt} \\ p_y(0)_{gt} \end{bmatrix} \tag{149}$$

corresponded to the first measurement of the ground truth, while the initial tilt is estimated from the first IMU measurement and initial yaw is obtained from the first ground truth as

$$\mathbf{h}(0) = \begin{bmatrix} \psi(0) \\ \theta(0) \\ \phi(0) \end{bmatrix} = \begin{bmatrix} arctan(\frac{a_y(0)}{a_z(0)}) \\ arctan(\frac{-a_x(0)}{\sqrt{a_y^2(0)+a_z^2(0)}}) \\ \phi(0)_{gt} \end{bmatrix}. \tag{150}$$

For the corresponding initial covariance matrix $\mathbf{P}(0)$, the values in Table 1 were used for each state, converted to a diagonal matrix.

Table 1: Initial variances of the state vector

State	Initial Variance
p_x, p_y	0.1
v_x, v_y	0.1
ψ, θ, ϕ	0.1
ω_x, ω_y, ω_z	0.01
a_x, a_y	0.01
δa_x, δa_y	5
$\delta\omega_x$, $\delta\omega_y$, $\delta\omega_z$	5

The initial variances for position, velocity, orientation and acceleration were manually chosen and tuned by trial and error. However, it is important to note that the initial bias variances are comparatively larger than that of their corresponding state variables. This ensures an initially faster moving bias, so that the erroneous and biased measurements at the beginning of the motion will adjust the bias values rather than the actual state values.

For the Q matrix, the variances listed in Table 2 were used for the highest order terms, then Q matrix was constructed as explained in Section 4.4.

Table 2: Process noise variances

Process Noise Variance	Value
σ^2_{ax}, σ^2_{ay}	0.02
$\sigma^2_{\omega x}$, $\sigma^2_{\omega y}$	10^{-6}
$\sigma^2_{\omega z}$	0.175
$\sigma^2_{\delta_{ax}}$, $\sigma^2_{\delta_{ay}}$	10^{-6}
$\sigma^2_{\delta_{\omega x}}$, $\sigma^2_{\delta_{\omega y}}$, $\sigma^2_{\delta_{\omega z}}$	10^{-12}

It can be seen in Table 2 that the process noise variance of bias states were configured much lower compared to those of their corresponding navigation states. This configuration ensured that after the initial step where the biases are estimated at the beginning of the motion, they would be slowly moving. Therefore, when larger measurements arrive, the navigation states moved more rapidly and biases moved slowly.

Furthermore, as shown in Table 2, variance of x and y axis angular velocities were configured much lower compared to z axis, as the vehicle was expected to change its heading a lot, while in a horizontal surface very little tilt was expected. Also, acceleration biases were assigned a larger variance compared to gyroscope biases; this allowed them to move at a faster rate, which was useful to also absorb the effects of gravity in estimation, especially for the 2D model.

For the variances of sensor measurements to be used in R matrix of each sensor, the values in Table 3 were used.

Table 3: Sensor noise variances

Sensor	Sensor Noise Variance	Value
Laser	σ^2_{px}, σ^2_{py}	0.16
Laser	σ^2_{ϕ}	$(0.5 \cdot \pi/180)^2$
IMU	σ^2_{ax}, σ^2_{ay}	0.25
IMU	$\sigma^2_{\omega x}$, $\sigma^2_{\omega y}$	$(0.5 \cdot \pi/180)^2$
IMU	$\sigma^2_{\omega z}$	$(0.5 \cdot \pi/180)^2$
IMU	σ^2_{ψ}, σ^2_{θ}	$(10 \cdot \pi/180)^2$
Odometry	σ^2_{vx}, σ^2_{vy}	0.016
Odometry	$\sigma^2_{\omega z}$	$(\pi/180)^2$

All of the values in Table 3 were tuned with trial and error method, and the values of tilt measurements were kept especially large as those measurements are usually corrupted by the acceleration of the vehicle itself and should be filtered.

5.1.4 Estimation with Fully Available Absolute Positioning

In the first experiment with Rawseeds datasets, all the available measurements from laser, odometry and IMU were used throughout the navigation, and the estimated trajectory was compared with the trajectories obtained from ground truth, only the laser and only the odometry measurements. As the ground truth was available only as the 2D position and heading, tilt estimation of the vehicle was not considered in the analysis. Both the full 3D implementation of the developed localization method and simplified 2D implementation were tested.

In this test, ATE was used as a performance indicator since a successful algorithm should provide close results to the ground truth trajectory when all measurements are always available. However, in this test case, there was a fusion between relative and absolute measurements. In situations where the laser is providing incorrect results but the other sensors are more reliable, the estimation is expected to suddenly jump due to correction, which worsens the RPE of the result, even though those jumps bring the estimate closer to the ground truth than pure laser. Therefore, RPE was not used as a performance metric in this experiment.

The histogram of the ATE of pure laser benchmark solution was provided by Rawseeds, illustrated below in Figure 9, which was later compared to sensor fusion implementations.

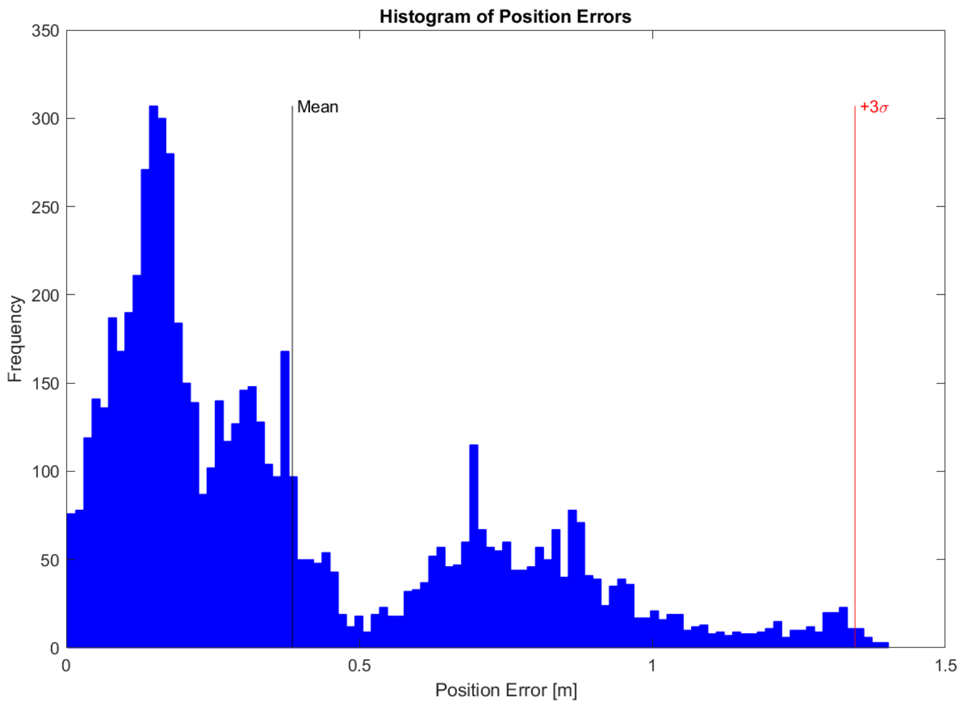


Figure 9: ATE histogram of laser scanner benchmark solution, obtained by functions and results provided by Rawseeds.

The trajectory obtained with 3D sensor fusion implementation can be seen in Figure 10, together with ground truth, pure laser and pure odometry trajectories. As seen, pure odometry drifted away from the ground truth considerably, as expected from a long navigation without absolute measurements, while the other trajectories were very close to each other.

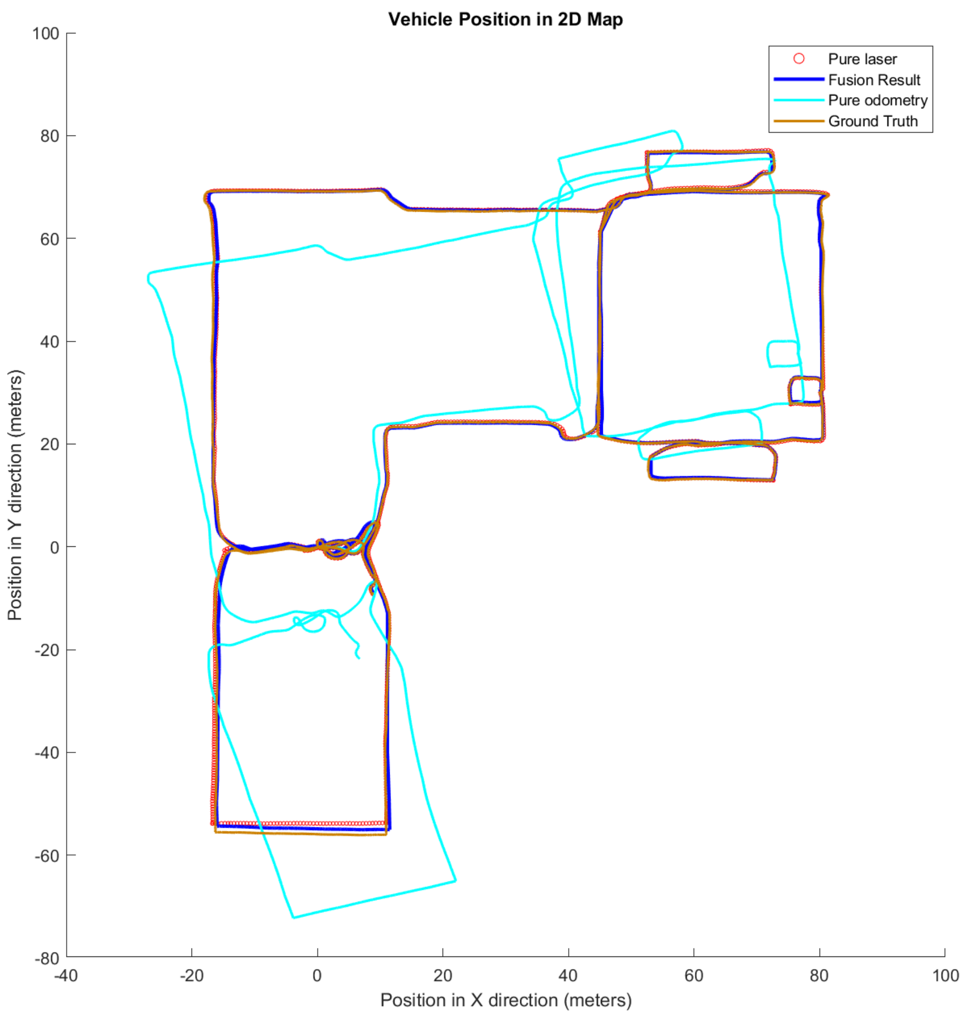


Figure 10: Resulting trajectory of 3D implementation in 2D map.

The corresponding histogram of the ATE of the fusion method is presented in Figure 11. For comparison, the ATE histogram obtained from the simplified 2D algorithm is presented in Figure 12. As the trajectory is almost identical to the one obtained from 3D implementation, the trajectory plot is not presented again to avoid redundancy.

The means and standard deviations of the resulting ATE from both implementations, together with that of the pure laser solution provided by Rawseeds are given in Table 4.

Figure 13 presents the translation and rotation errors with respect to time of both 2D and 3D algorithms, together with the benchmark laser scan matching solution. It is seen in this figure that although the sensor fusion trajectories seem similar to pure laser trajectory with the scale in Figure 10, there are in fact significant differences in their deviation from the ground truth.

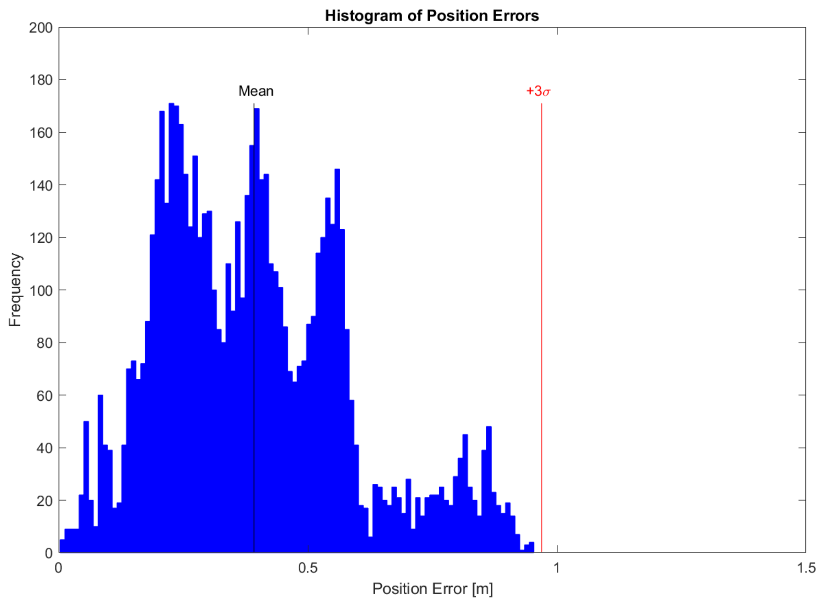


Figure 11: Histogram of position errors of 3D implementation.

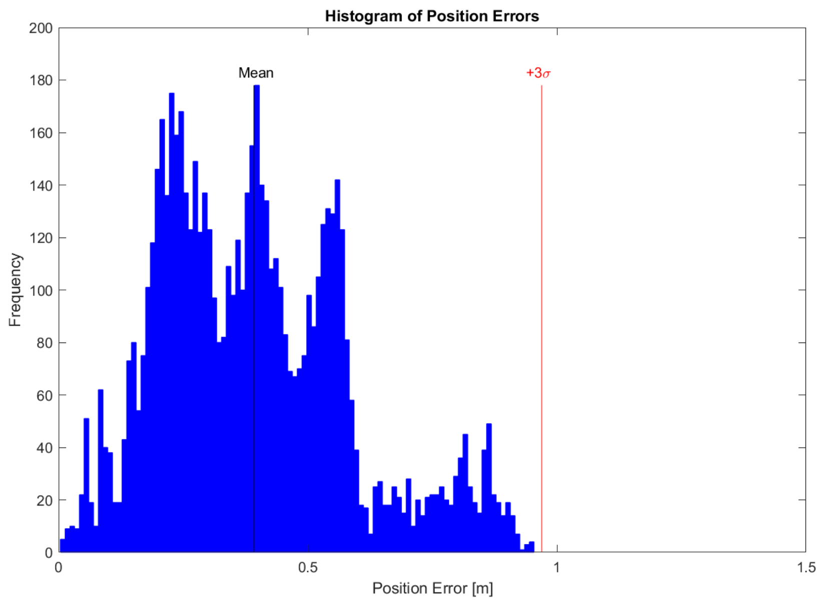


Figure 12: Histogram of position errors of 2D implementation.

Table 4: ATE means and standard deviations

Method	ATE Mean (m)	ATE Standard Deviation (m)
Pure Laser	0.3858	0.3201
2D Sensor Fusion	0.3920	0.1925
3D Sensor Fusion	0.3923	0.1924

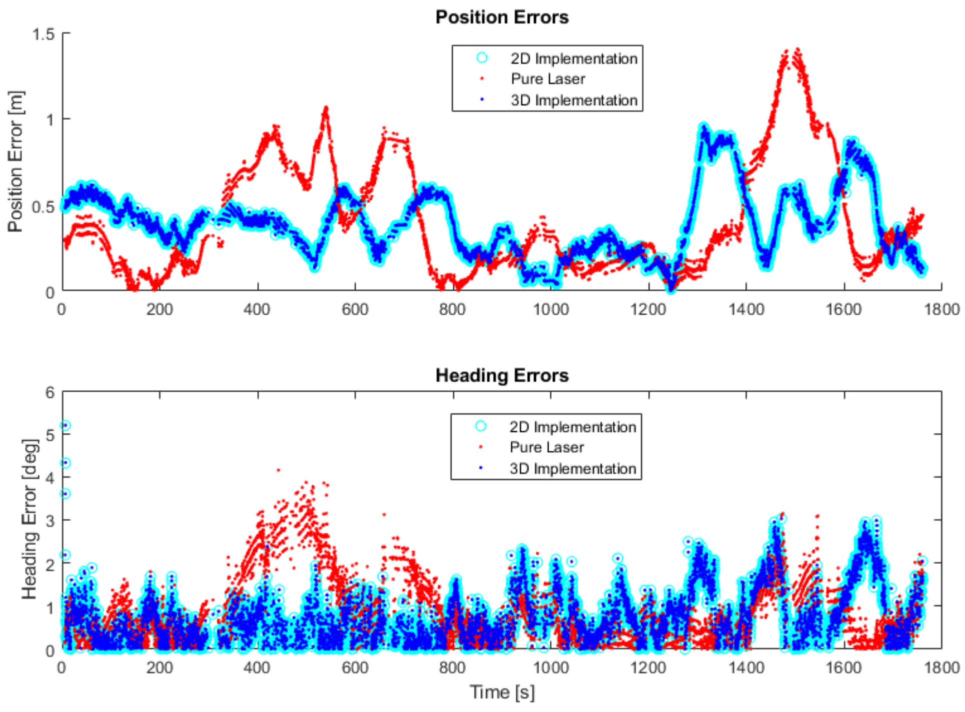


Figure 13: Translation and rotation errors of trajectories obtained with 2D sensor fusion, 3D sensor fusion and only laser scan matching, with respect to time.

As seen by comparing the trajectories and ATE results, for this motion there was no notable difference between a simplified 2D sensor fusion and a full 3D one including tilt estimation. Even though the ATE mean has slightly increased by 6 mm from benchmark solution, there was significant improvement in standard ATE deviation by 13 cm. Comparing the histograms at Figures 9, 11 and 12, and observing the error profile in Figure 13, it is seen that using sensor fusion prevented the highest deviations from the ground truth more than 1 meters that were present in the pure laser navigation, which in turn reduced the standard deviation by a considerable margin.

5.1.5 Estimation with Partially Available Absolute Positioning

The second group of experiments on the Rawseeds dataset was conducted to demonstrate the robustness of fusion algorithm when subject to laser outages; a common scenario for AGV in environments with low amount of natural features or a poor navigation map. This was achieved by ignoring the laser measurements provided in the dataset in certain time intervals, and using all three measurement sources during the rest of the motion. As with the previous experiment case, both 2D and 3D implementations of the developed sensor fusion were tested.

In addition, the effect of IMU in reducing the drift in time periods lacking laser measurements was demonstrated in this experiment. To that end, a third version of the sensor fusion model where all IMU measurements are ignored was also tested as well. It is noted that as IMU was the only information source for 3D orientation, 2D and 3D models are equivalent in this test case.

The simulated laser outage time periods were arbitrarily chosen, but always kept the same through the different tests. Those time periods were between the timestamps listed below in the rows of **L** matrix, their units being in seconds since the start of the motion

$$\mathbf{L} = \begin{bmatrix} 10 & 130 \\ 132 & 250 \\ 252 & 400 \\ 410 & 600 \\ 615 & 700 \\ 750 & 870 \\ 901 & 1000 \\ 1105 & 1300 \\ 1400 & 1550 \\ 1600 & 1782 \end{bmatrix}. \tag{151}$$

This means that if the timestamp of the laser measurement is between the values of any row of **L**, that measurement is ignored. As seen, the laser outage times were chosen such that the laser was used for a few seconds, then it was ignored for a period of time ranging between a minute to three minutes.

To analyse the results, both visual trajectories and ATE for each motion are presented. In absence of absolute positioning, the drift increases over time, therefore, ATE is expected to grow with time. However, as the motion itself and the time periods with and without laser are the same in all 3 cases, ATE is a meaningful performance indicator when used in comparing these cases with each other.

The estimated location with the 3D implementation can be seen in Figure 14 and the corresponding ATE histogram of the motion can be seen in Figure 15. Similarly, ATE histogram and trajectory over time of 2D motion is presented in Figures 16 and 17.

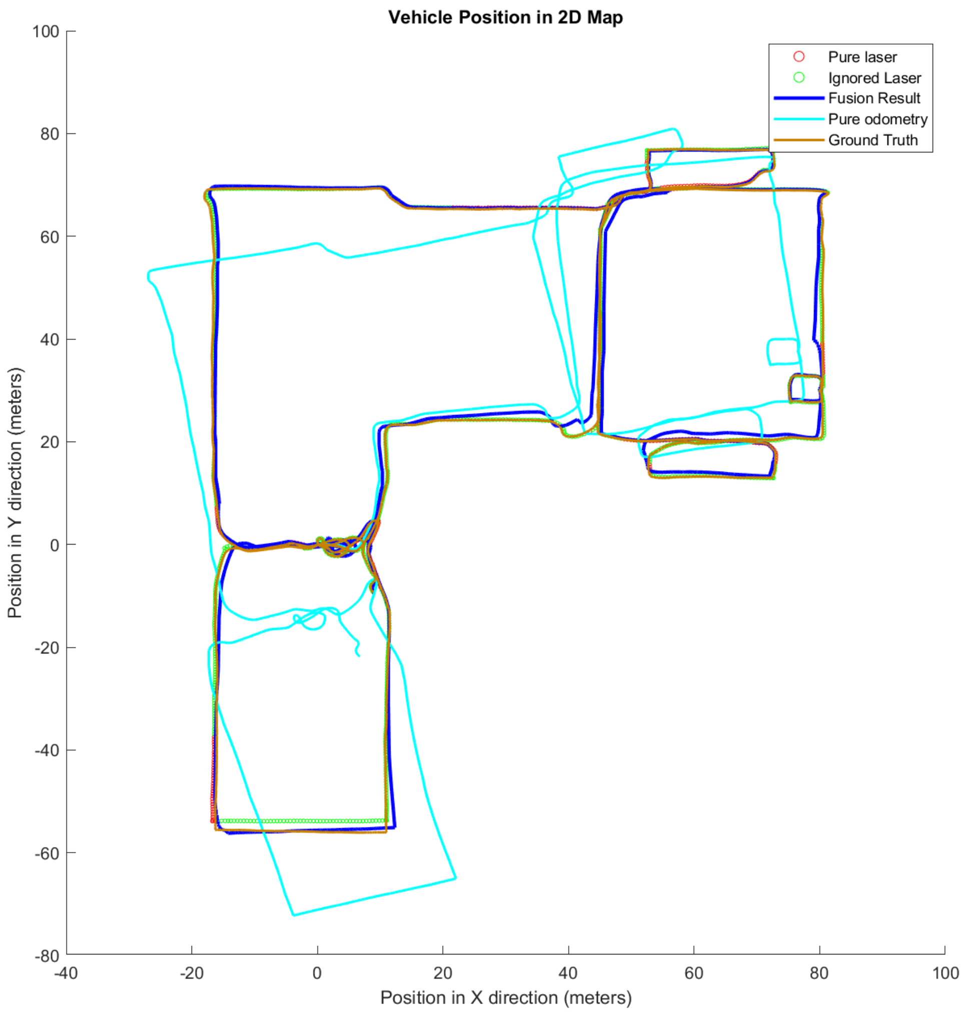


Figure 14: Resulting trajectory of 3D implementation. The periods where the laser is ignored is marked in green.

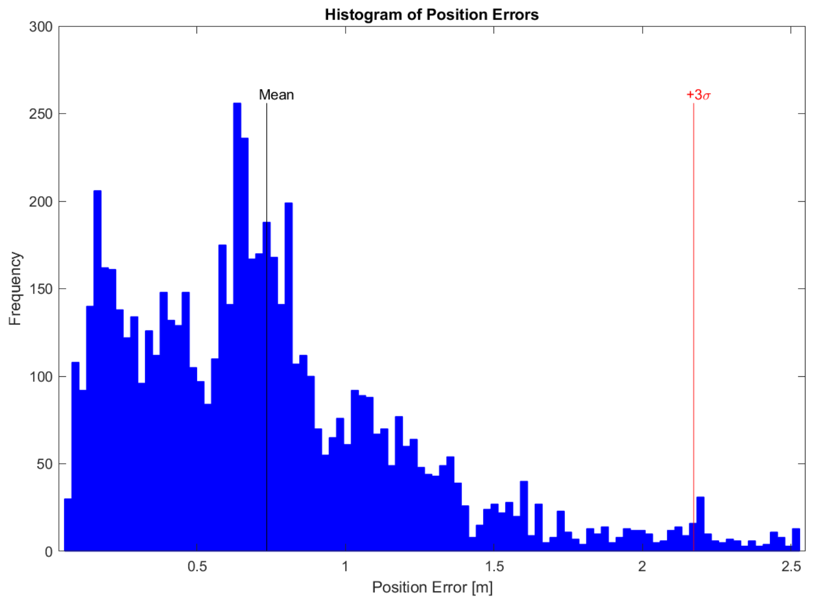


Figure 15: Histogram of position errors of 3D implementation with laser outage.

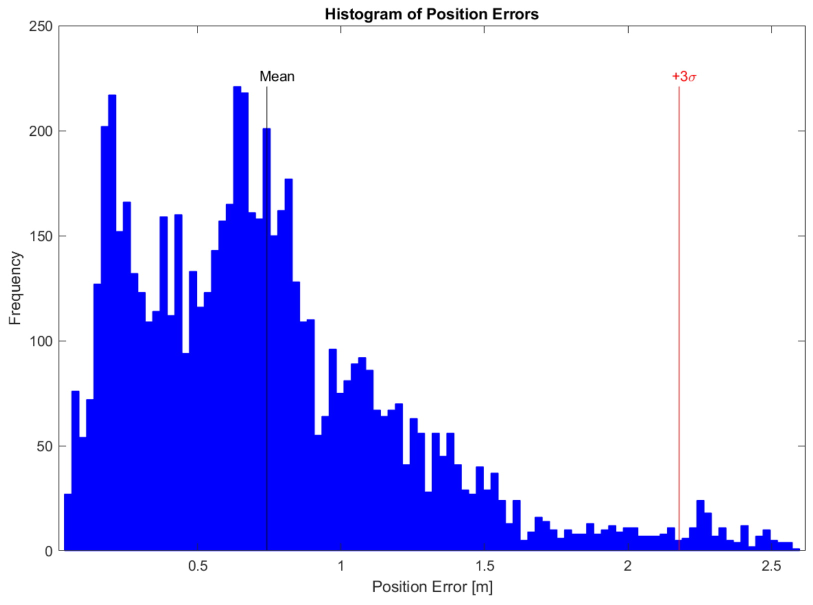


Figure 16: Histogram of position errors of 2D implementation with laser outage.

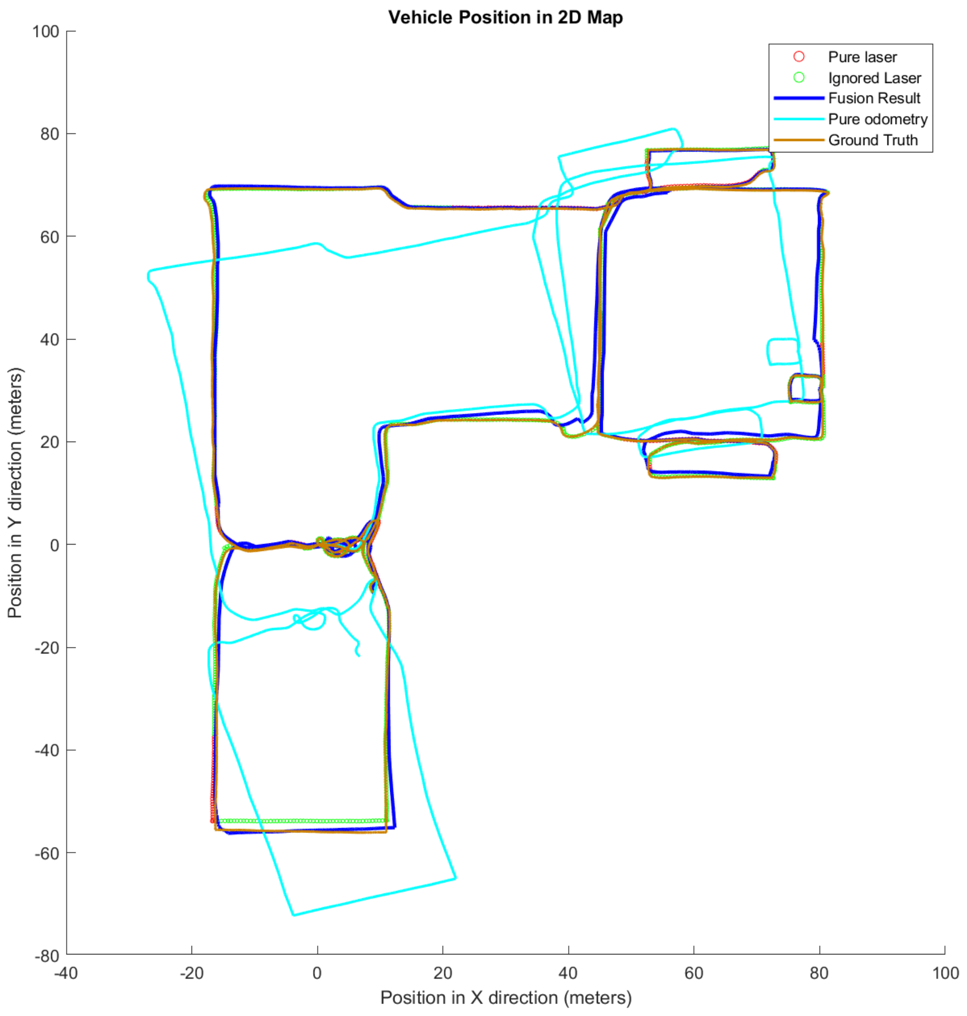


Figure 17: Resulting trajectory of 2D implementation. The periods where the laser is ignored is marked in green.

The resulting trajectory of the implementation without IMU can be seen in Figure 18 and the corresponding ATE histogram in Figure 19.

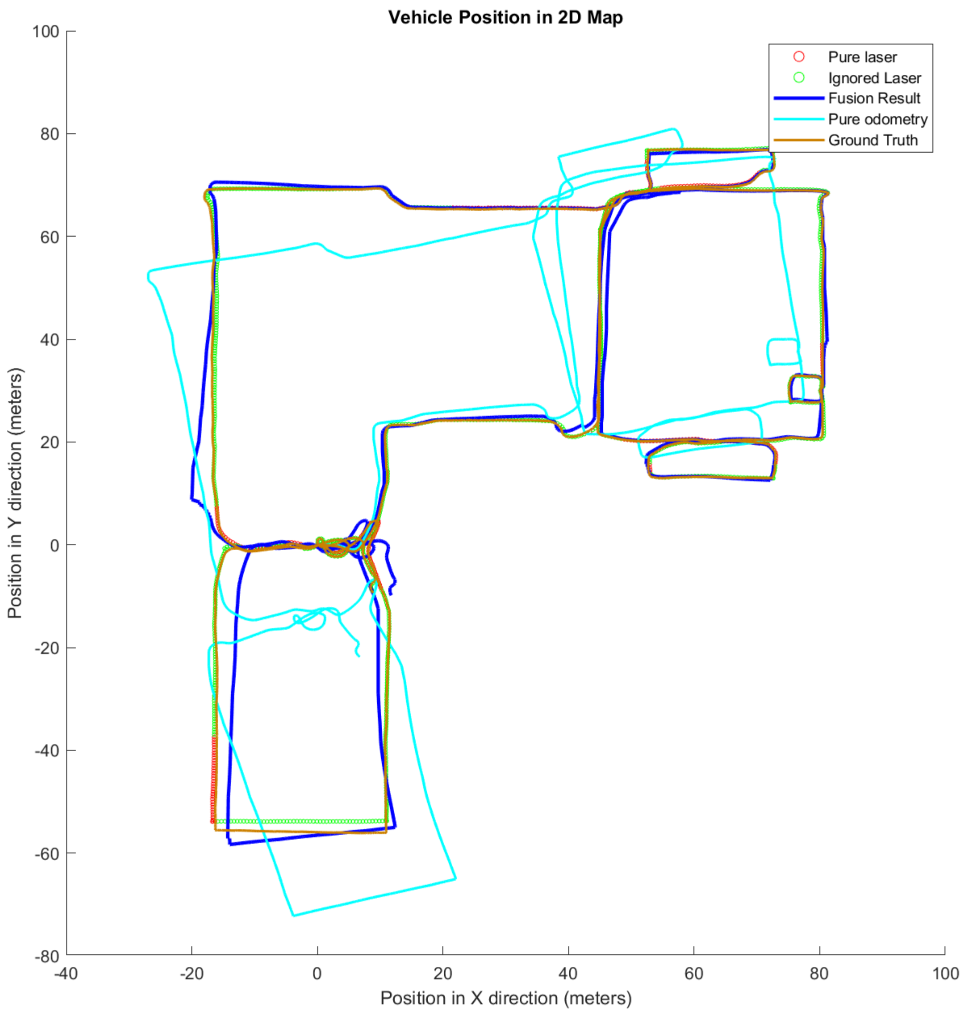


Figure 18: Resulting trajectory of implementation without IMU over time. The periods where the laser is ignored is marked in green.

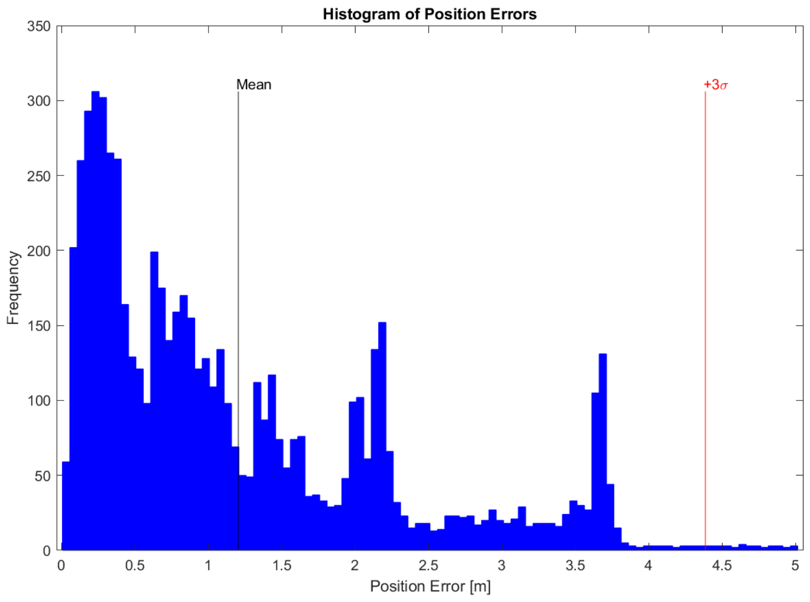


Figure 19: Histogram of position errors of implementation without IMU.

The means and standard deviations of the resulting ATE from all implementations are given in Table 5. In addition, the trajectory errors over time with all three configurations can be seen in Figure 20.

Table 5: ATE means and standard deviations

Method	ATE Mean (m)	ATE Standard Deviation (m)
2D Implementation with IMU	0.7416	0.4789
3D Implementation with IMU	0.7352	0.4791
Implementation without IMU	1.2039	1.0610

Similarly to the case with all sensors, it is seen that there is no meaningful difference between full 3D and simplified 2D implementations of sensor fusion. However, the advantage of both methods over relying on only odometry during the periods without laser scanners is clear with 50 cm improvement in ATE mean and 60 cm improvement in standard deviation with both methods. It is also seen that the amount of drift varies in different laser outage periods in all different configurations. It can be concluded that although in certain outage periods using IMU has resulted in an increasing drift, such as the first outage period in the motion, using IMU is beneficial overall as it has prevented the largest drifts.

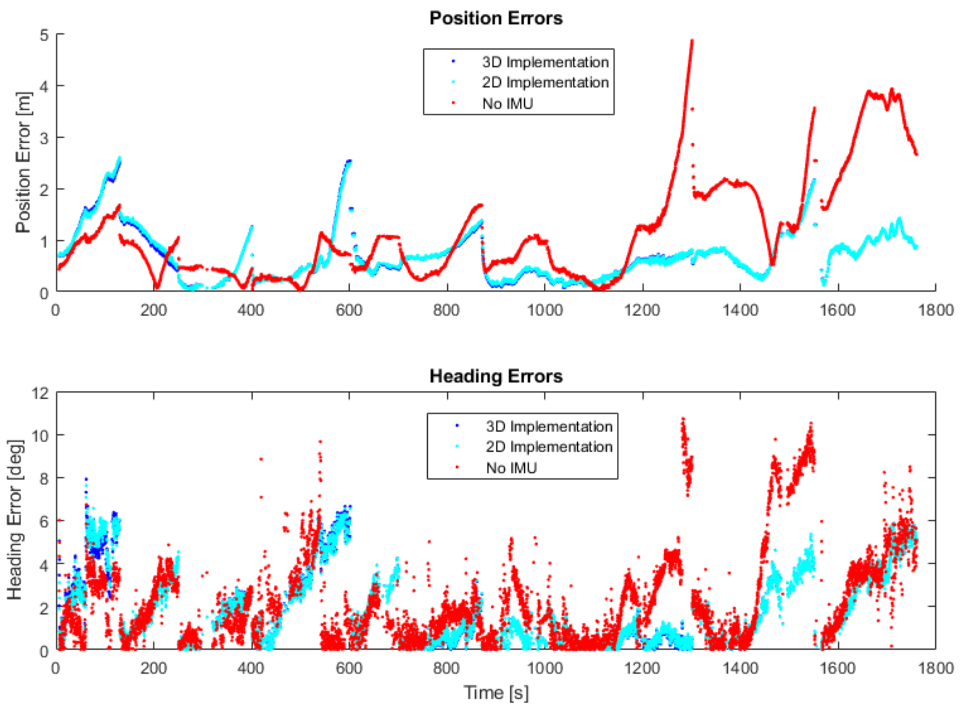


Figure 20: Position and heading errors of all three implementations in laser outage scenario.

5.1.6 Estimation in Absence of Absolute Positioning

The final group of tests with Rawseeds dataset were conducted to demonstrate that the developed algorithm is effective in reducing the drift in absence of absolute positioning sensors throughout the whole motion. To that end, localization was performed without using the laser at all, except for the initialization of the very first position. In this test, sensor fusion of IMU and odometry was compared to the pure odometry solution provided by Rawseeds. Since this test included a long motion without any absolute positioning measurements, a large deviation from the ground truth trajectory is expected in every case, and ATE will not be a meaningful performance indicator. Instead, RPE was used to characterize the amount of drift to assess the success of the algorithm.

The translational and rotational relative pose error in each time step from the pure odometry trajectory can be observed in Figure 21.

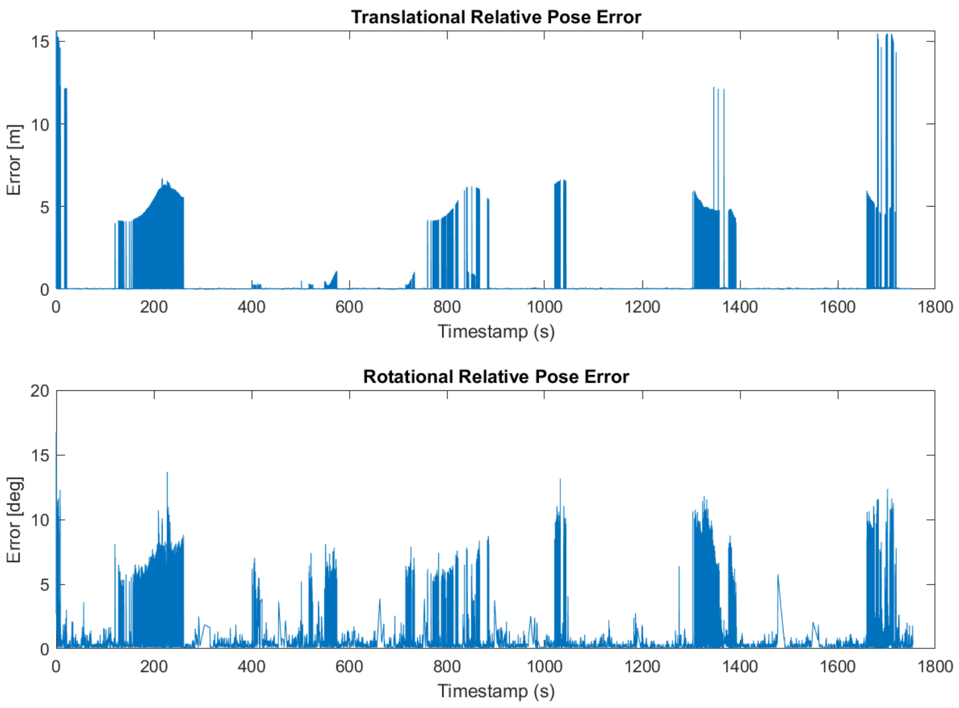


Figure 21: Relative Pose Error of pure odometry trajectory.

For comparison, the same RPE plot for the 3D implementation of sensor fusion between IMU and odometry is shown in Figure 22. The corresponding RMSE values are given in Table 6.

It is seen that sensor fusion of odometry and IMU clearly reduced the drift throughout the motion, resulting in close to 43 cm less tRPE and 0.6 degrees less rRPE. However, it should be noted that while the position estimate has improved, it is still not usable for AGV applications. This can be observed in Figure 23, where the position estimate is more than 20 meters away from the ground truth in some

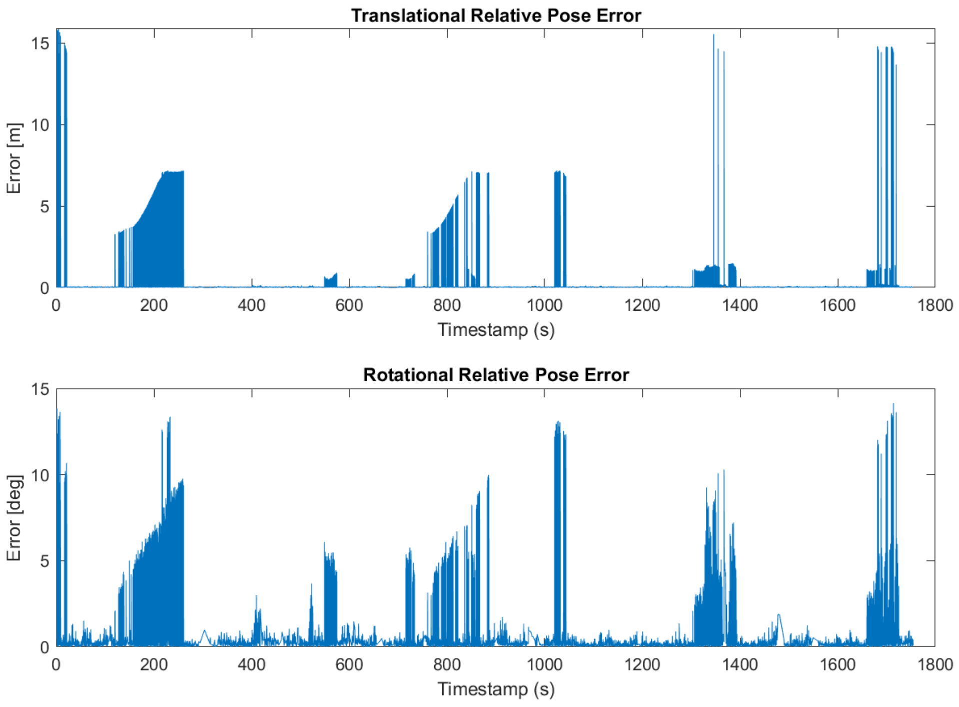


Figure 22: Relative Pose Error of 3D sensor fusion trajectory.

Table 6: Relative Pose Errors

Method	tRPE (m)	rRPE (deg)
Pure Odometry	2.7453	3.3688
3D Implementation with IMU	2.3156	2.7550

places. This is an expected result as with a long motion such as this, the drift will always be very large without using any absolute measurements.

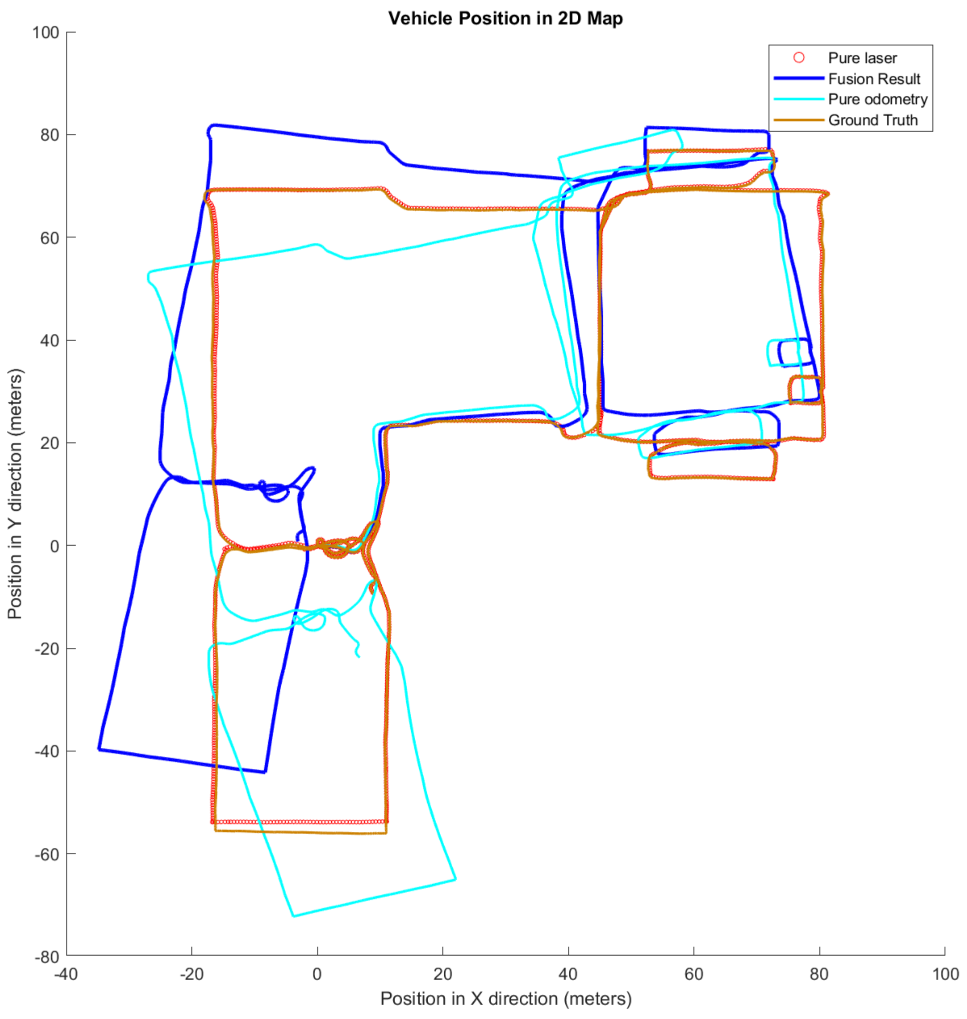


Figure 23: The trajectory obtained without using any laser measurements.

5.2 Real-Time Performance Analysis

In these experiments, the performance of the sensor fusion was analyzed using an AGV while the localization algorithm was run real-time. Unlike the experiments with Rawseeds datasets, there was no universal ground truth available in these experiments. Therefore, instead of quantitative performance indicators like ATE and RPE, the results were analyzed visually. The main purpose of these experiments were to assess the real-time performance of the fusion and to demonstrate whether it could be used as a reliable localization algorithm. Compared to the long motion in Rawseeds experiments, these real-time experiments were focused on motions spanning shorter time periods but with more turns and velocity changes to assess the ability of the algorithm to follow these changes in real-time.

5.2.1 Experiment and Parameter Setup

All of the experiments in real-time were conducted using Navitrol software, which is a navigation solution for AGVs developed by Navitec Systems [66]. The existing software is capable of collecting the relevant data from IMU and odometry in the same way as Rawseeds datasets. Similarly, the laser scanner measurements are used in map matching methods to provide vehicle positions in the form of x, y, ϕ as needed for the Kalman Filter as explained in Section 4.5.1. For the purpose of these experiments, the sensor fusion localization solution was implemented as a parallel method to the existing Navitrol localization in C programming language. This was achieved such that every time an IMU, laser or odometry measurement arrived, Navitrol sent this timestamped measurement to the implemented sensor fusion, and the algorithm produced a localization estimate. This achieved the same architecture presented in Figure 7, such that the map matching was performed based on the output of the existing localization system and not on sensor fusion outputs.

Navitrol software was readily installed and integrated with several demo AGVs with different kinematic configurations. An omnidirectional vehicle was chosen among them to conduct these experiments, to enable maximum freedom in movement. The vehicle was equipped with a Pepperl Fuchs OMD30M-R2000 laser scanner that was used in the existing map matching algorithm together with the odometry measurements to provide the laser location estimates. Futhermore, a low cost MEMS LPMS CU2 IMU provided the necessary accelerometer and gyroscope measurements. For the experiments, all of the noise suppressing methods present in both Navitrol and individual sensors were disabled, such as the low pass filter in gyroscopes and odometry auto-calibration feature of Navitrol, to get raw data from sensors and let the sensor fusion algorithm work with noisy inputs.

The noise variances used in the fusion algorithm for these tests are summarized in Table 7. It is noted that with the notable exception of odometry angular velocity variance and all laser variances, all of the parameters are similar to the ones used for Rawseeds experiments, but they were tuned with further trial and error for higher performance. Most of the smaller changes were done to accommodate the overall slightly more noisy measurements. However, as it was known that the scan matching performed in Navitrol is quite reliable and the scanner installed on the demo AGV is able to provide high quality measurements, all laser variances were set to much lower values compared to the previous experiment.

On the other hand, it was known that this particular hardware produces less accurate odometry measurements for angular velocity compared to IMU measurements. Therefore, the odometry angular velocity measurement was given quite a high value for its variance and was practically ignored compared to the high confidence measurements from the IMU.

Table 7: Sensor noise variances

Variance Type	Navigation State	Value
Initial State Variance	p_x, p_y	0.1
Initial State Variance	v_x, v_y	0.1
Initial State Variance	ψ, θ, ϕ	0.1
Initial State Variance	ω_x, ω_y, ω_z	0.01
Initial State Variance	a_x, a_y	0.01
Initial State Variance	δa_x, δa_y	5
Initial State Variance	$\delta\omega_x$, $\delta\omega_y$, $\delta\omega_z$	5
Process Noise Variance	σ^2_{ax}, σ^2_{ay}	0.04
Process Noise Variance	$\sigma^2_{\omega x}$, $\sigma^2_{\omega y}$	10^{-2}
Process Noise Variance	$\sigma^2_{\omega z}$	0.0175
Process Noise Variance	$\sigma^2_{\delta_{ax}}$, $\sigma^2_{\delta_{ay}}$	10^{-6}
Process Noise Variance	$\sigma^2_{\delta_{\omega x}}$, $\sigma^2_{\delta_{\omega y}}$, $\sigma^2_{\delta_{\omega z}}$	10^{-6}
Laser Noise Variance	σ^2_{px}, σ^2_{py}	0.0004
Laser Noise Variance	σ^2_{ϕ}	$(0.1 \cdot \pi/180)^2$
IMU Noise Variance	σ^2_{ax}, σ^2_{ay}	0.25
IMU Noise Variance	$\sigma^2_{\omega x}$, $\sigma^2_{\omega y}$	$(0.3 \cdot \pi/180)^2$
IMU Noise Variance	$\sigma^2_{\omega z}$	$(0.3 \cdot \pi/180)^2$
IMU Noise Variance	σ^2_{ψ}, σ^2_{θ}	$(10 \cdot \pi/180)^2$
Odometry Noise Variance	σ^2_{vx}, σ^2_{vy}	0.016
Odometry Noise Variance	$\sigma^2_{\omega z}$	$(50 \cdot \pi/180)^2$

5.2.2 Estimation with Fully Available Absolute Positioning

The first test to assess the real-time performance of the algorithm was conducted by using data from all three types of sensors throughout the whole motion. The AGV was navigated in a 3×4 meters indoor space with a horizontal surface following a trajectory with many turns. The estimated trajectory by sensor fusion compared to the trajectory obtained from pure laser measurements can be observed in Figures 24 and 25.

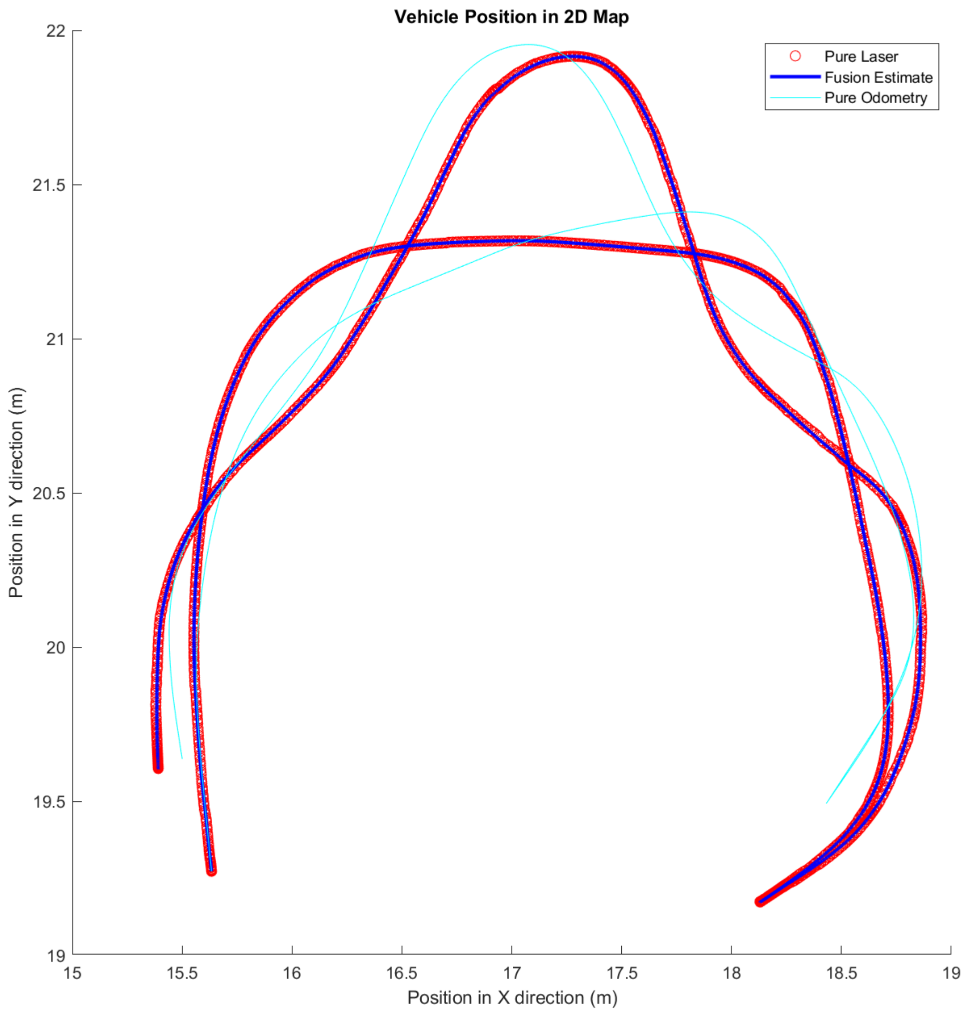


Figure 24: Vehicle position in 2D plane.

It is clearly seen that the fusion result closely follows the laser output, as expected with the low values assigned to laser measurement noise variances, whereas the pure odometry is drifting away. At a glance, it seems like there is no difference between pure laser and fusion trajectories. However, a zoomed in version to the trajectory displayed in Figure 26 demonstrates the advantage of using sensor fusion instead of

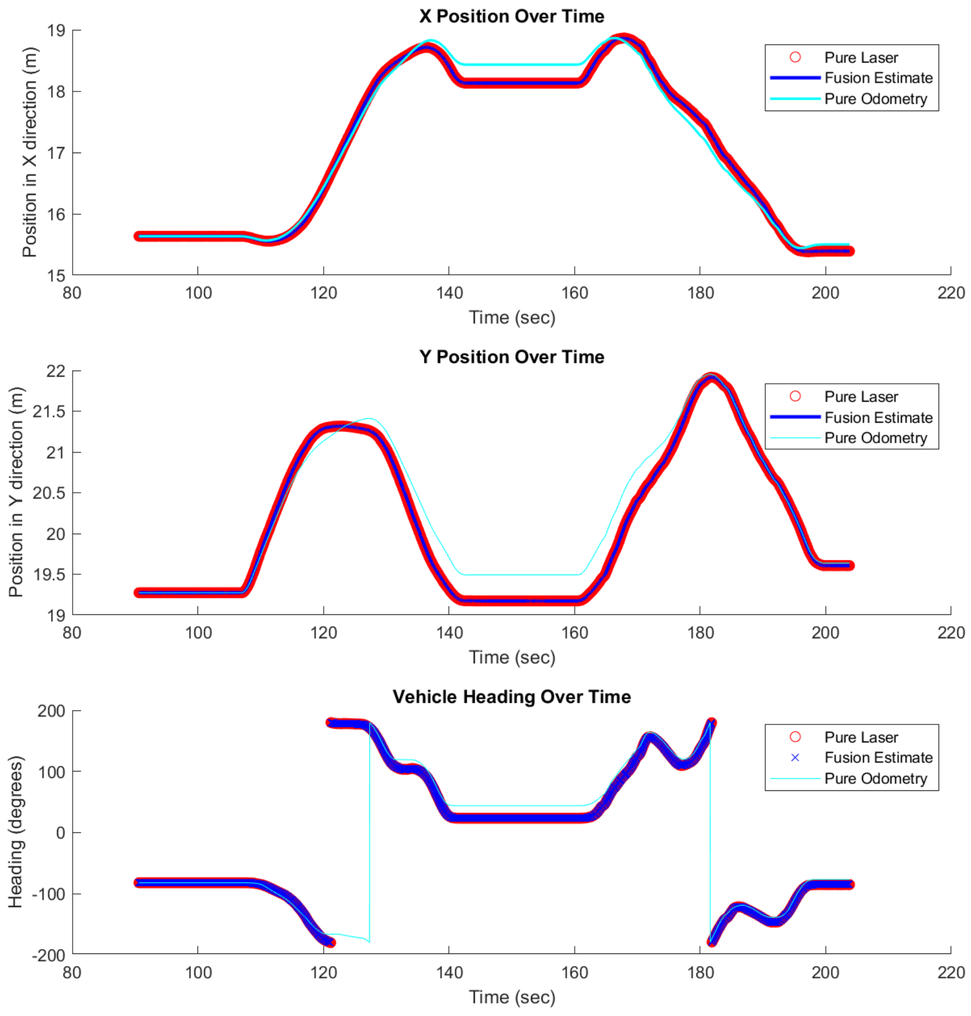


Figure 25: Vehicle position and orientation over time.

pure laser in this setup. It is seen that at the zoomed-in area, the laser was slightly off for a few measurements, which is deduced from the fact that the motion itself was a smooth one as evident from odometry profile. With the addition of measurements from IMU and odometry, the fusion result was able to smoothly follow the motion unlike the purely laser based navigation solution. It is noted that both the scale of the error and the correction is quite small in this case, which are contained within less than 2 cm, but it can still be significant depending on the application requirements.

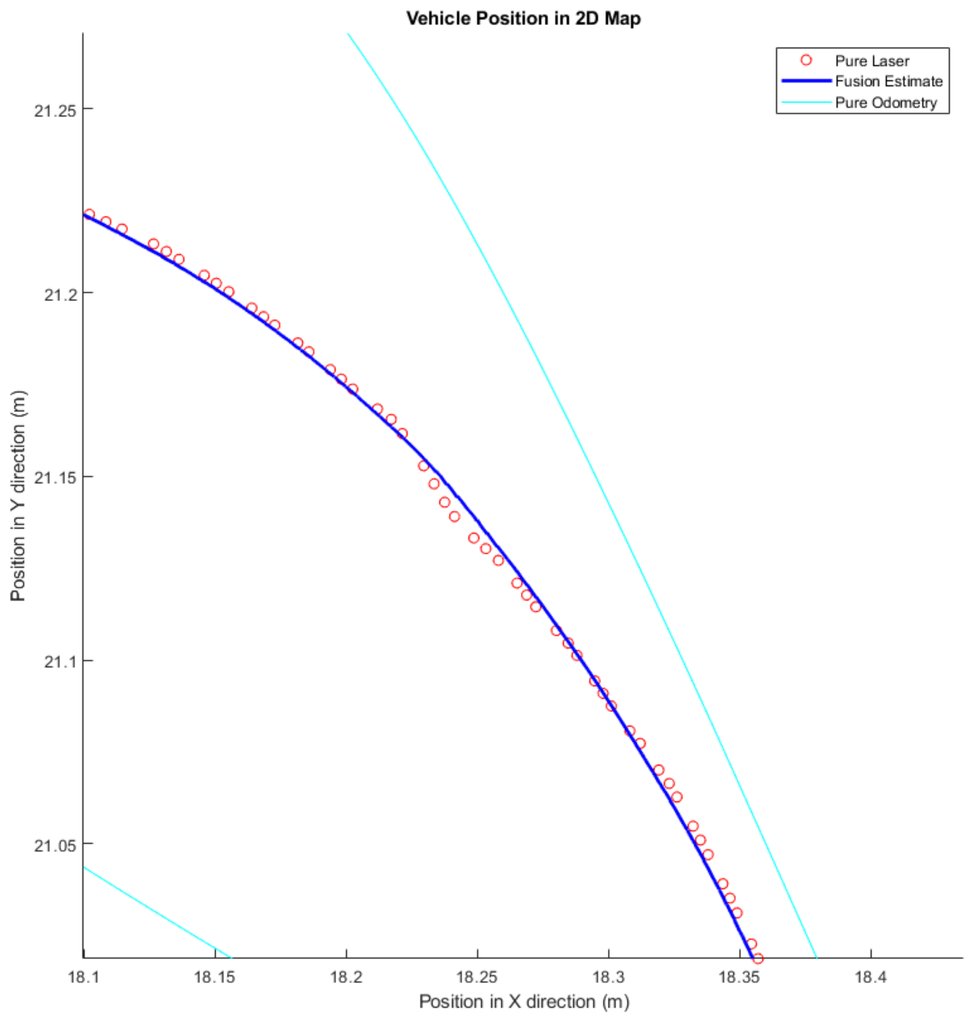


Figure 26: Zoom in on vehicle position in the 2D plane.

5.2.3 Estimation with Partially Available Absolute Positioning

Similar to the Rawseeds experiments, the performance was also tested on real-time for the situations where the laser scanner was unavailable for a certain period of time. The AGV followed a similar motion to the previous experiment, such that it was contained in a 3×4 office room on a horizontal surface. The sensor fusion algorithm was run in this experiment by ignoring the laser measurements in arbitrarily chosen time intervals. Those intervals are listed in the elements of **L** matrix below, such that laser measurements with timestamps between the elements in each row of L matrix were not used. It should be noted that their units are seconds since the time of the first laser measurement in the motion, not the actual timestamp

$$\mathbf{L} = \begin{bmatrix} 20 & 30 \\ 31 & 70 \\ 70.25 & 100 \end{bmatrix}. \tag{152}$$

With the intervals chosen this way, the laser measurements were available for 20 seconds at the beginning of the motion, most of which were spent in the stationary stance. Then, three periods of laser outage about 10, 30 and 40 seconds were simulated with very short periods of laser availability between them. The estimated trajectory can be observed in Figures 27 and 28. It is seen that while there is some drift from the laser trajectory, it is quite minimal. Furthermore, the improvement over the trajectory predicted by pure odometry is clearly seen.

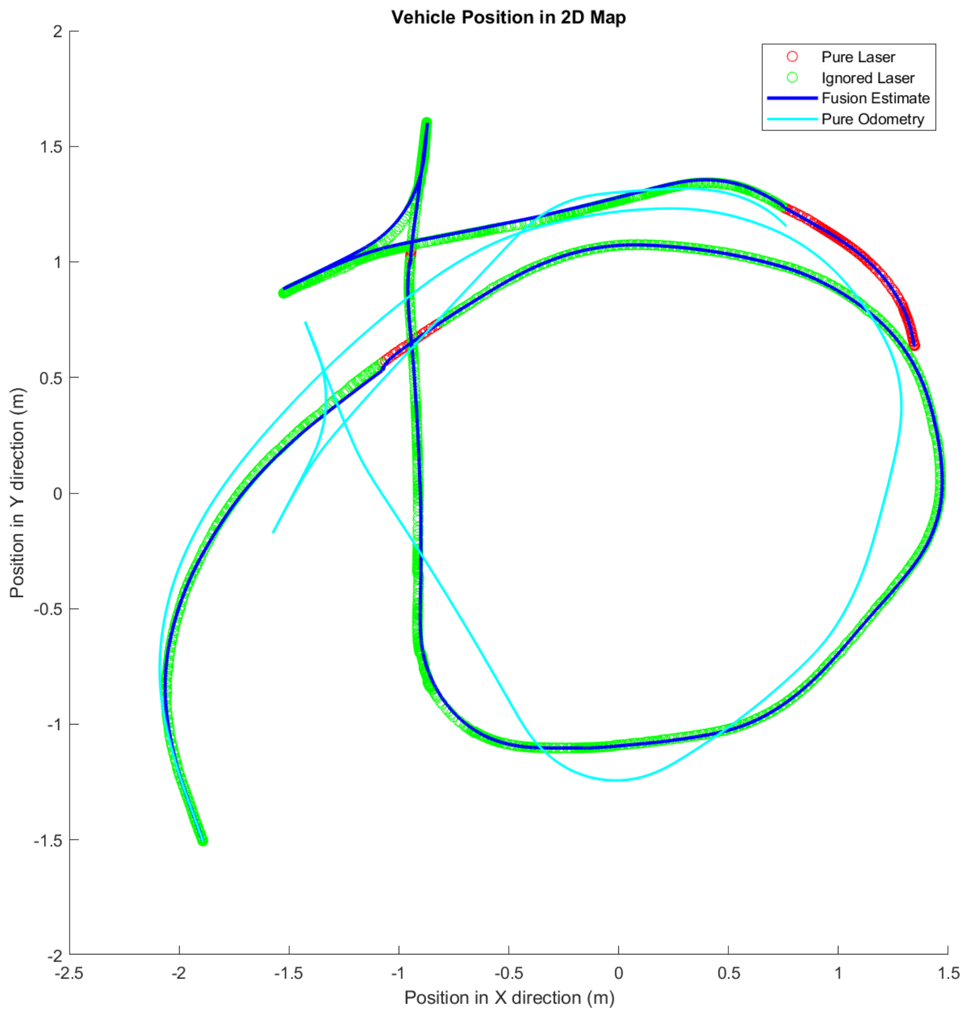


Figure 27: Vehicle position in 2D plane.

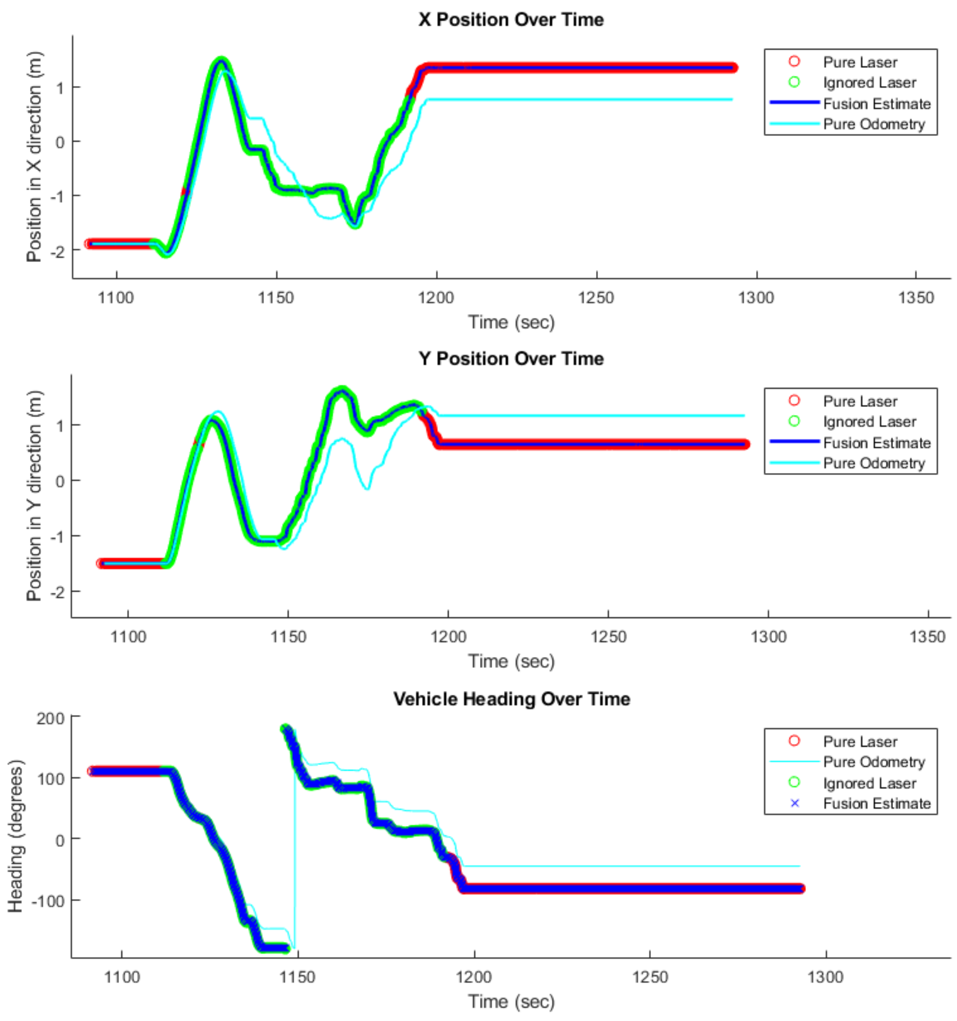


Figure 28: Vehicle position and orientation over time.

Another motion with the same conditions as the previous ones can be seen in Figure 29. It can be observed that the drift from the laser trajectory is a bit higher compared to the previous motion, but still much lower than using only odometry. Therefore, it can be concluded that while using sensor fusion between IMU and odometry in the periods of laser outage is certainly beneficial, the amount of drift during the same time period depends on the motion itself.

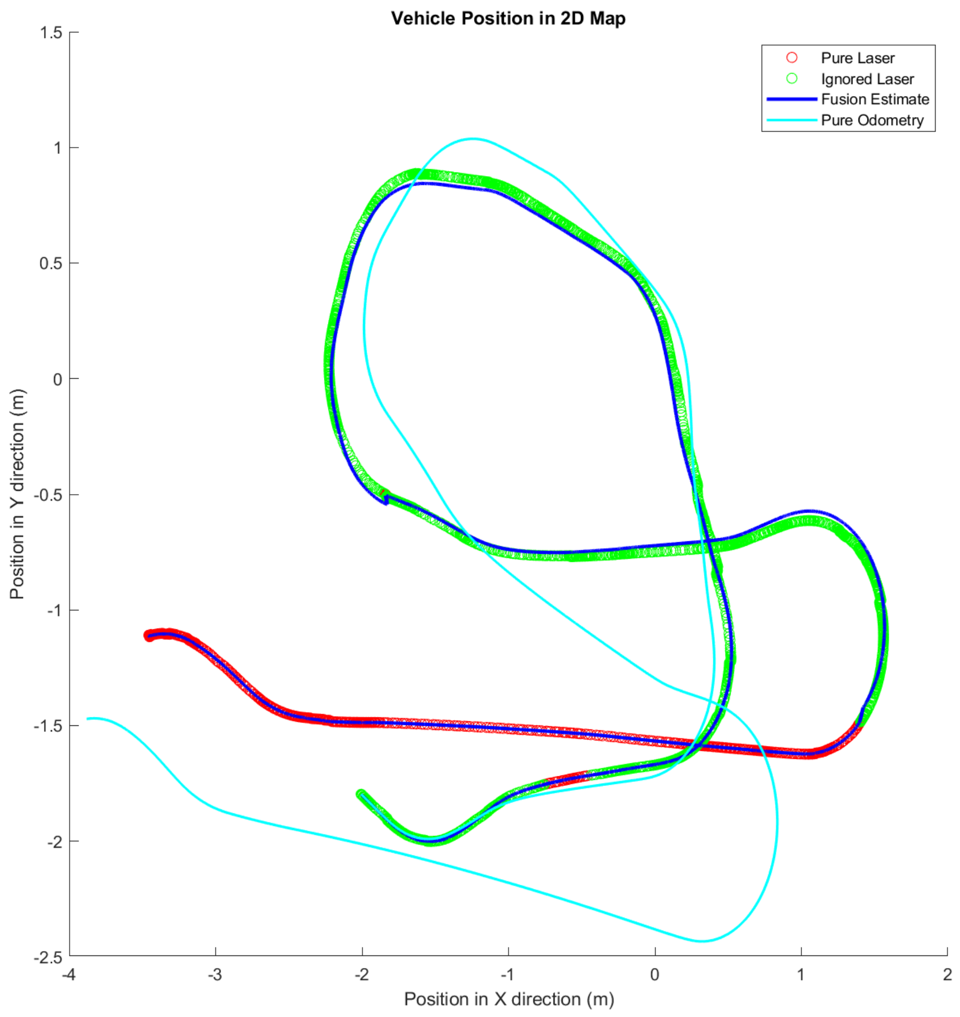


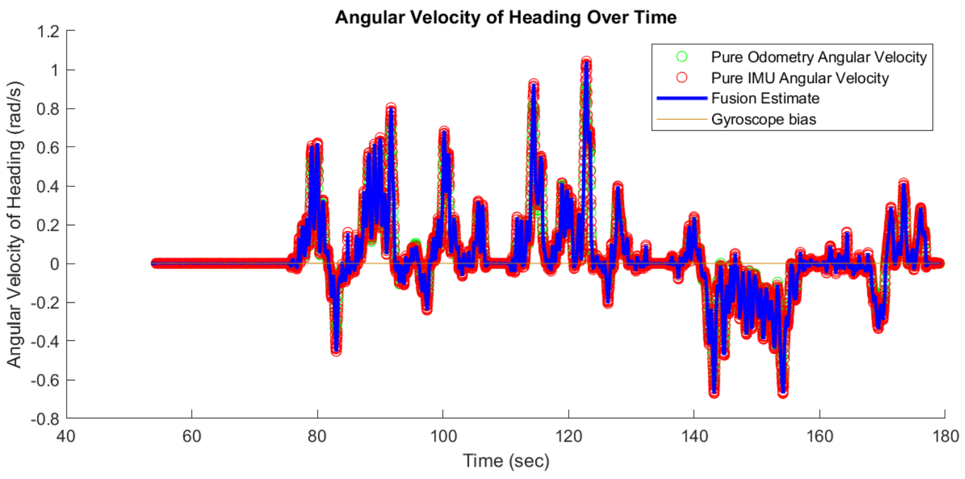
Figure 29: Vehicle position in 2D plane for a different motion.

Figure 30 shows the process of angular velocity and corresponding gyroscope bias estimation. All of IMU, laser and odometry provide measurements about the heading angle. Therefore, angular velocity bias estimation for gyroscope can be best studied for this component. On the top plot of Figure 30, it can be seen that the estimated angular velocity closely followed that of the gyroscope measurements because of the low variance assigned as shown in Table 7. Furthermore, the estimated bias was very small and very slowly moving, which accurately reflects the real case of a gyroscope.

In the bottom figure, the bias can be observed more closely. It is seen that at the start of the motion, it varied quite fast due to high initial assigned variance. Furthermore, in the presence of laser it moved much faster and exhibited sudden jumps when laser is first introduced after an outage period. This is resulting from the fact that the only other available unbiased information about angular velocity was odometry, but due to high variance in odometry measurement of angular velocity, it was essentially ignored. The laser provided a reliable and unbiased information source, so that the filter could differentiate between the actual angular velocity and the bias component from gyroscope, that enabled a much faster moving bias estimation.

It is also seen here that due to bias being modeled as constant in the prediction model presented in Section 4.4 and the low variance configuration, the Kalman Filter expected it to move slowly and not suddenly change. This caused oscillations when the laser was causing the aforementioned jumps in the estimation as can be best observed towards the 160-180 second range. It can be argued that this might cause problems when the laser did not have enough time to settle these oscillations as seen in the two middle sections, but is should be considered that the magnitude of these oscillations were so small that they were essentially invisible compared to the real angular velocity, and it is seen that there was no visible negative effect on the motion itself.

Another motion with the same laser conditions and parameters can be observed in Figure 31. It is seen that the estimation results in absence of laser positioning were much worse than earlier case, even though the time period of simulated laser outage was the same. The difference of this motion to the previous ones was that it included very high speed heading changes up to 57 degrees per second while the vehicle is stationary, which can be seen in the steep changes in the Heading vs. Time graph in Figure 32 around timestamps 5000 and 5040. It stands to reason that in such high speed conditions, the small difference in angular velocity estimate will integrate faster as heading error. This in turn causes a much higher error in the position estimation. Nevertheless, it is seen that sensor fusion was still beneficial over pure odometry estimation, reducing the drift by a considerable margin.



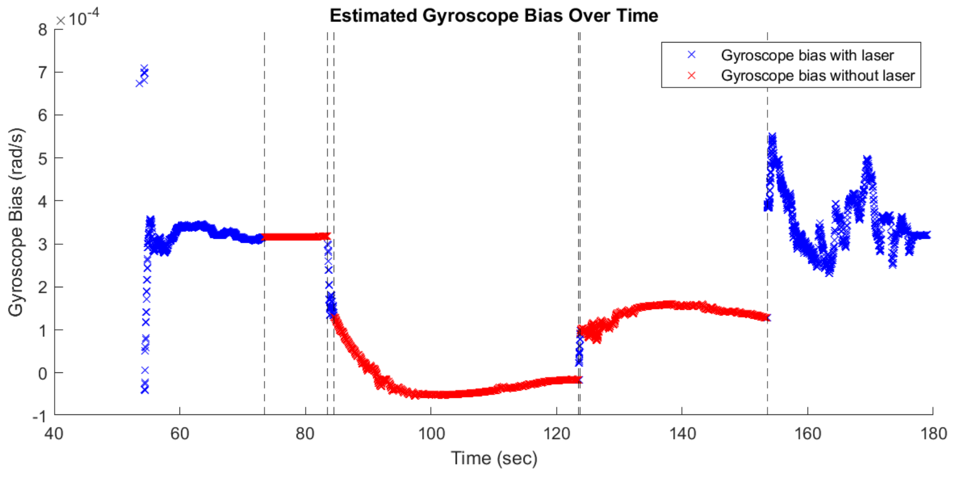


Figure 30: Angular velocity around z axis and a close-up on the bias estimation.

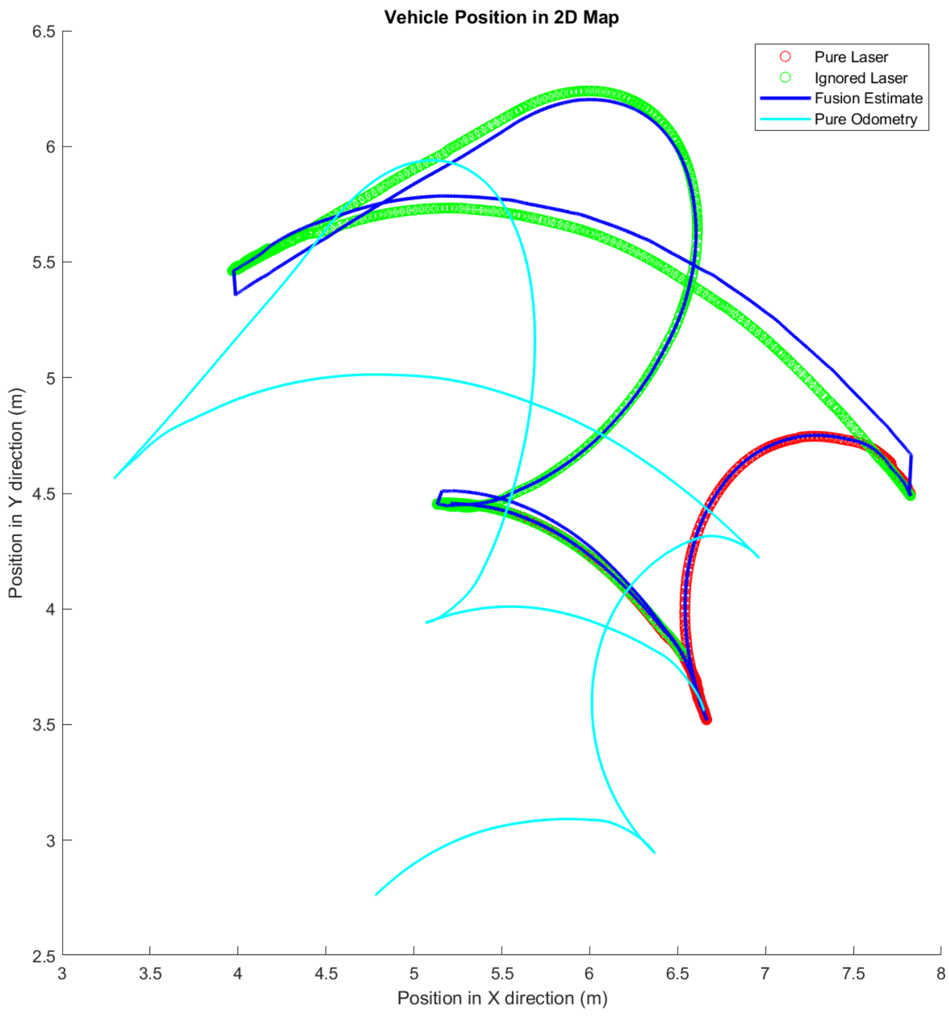


Figure 31: Vehicle position in 2D plane.

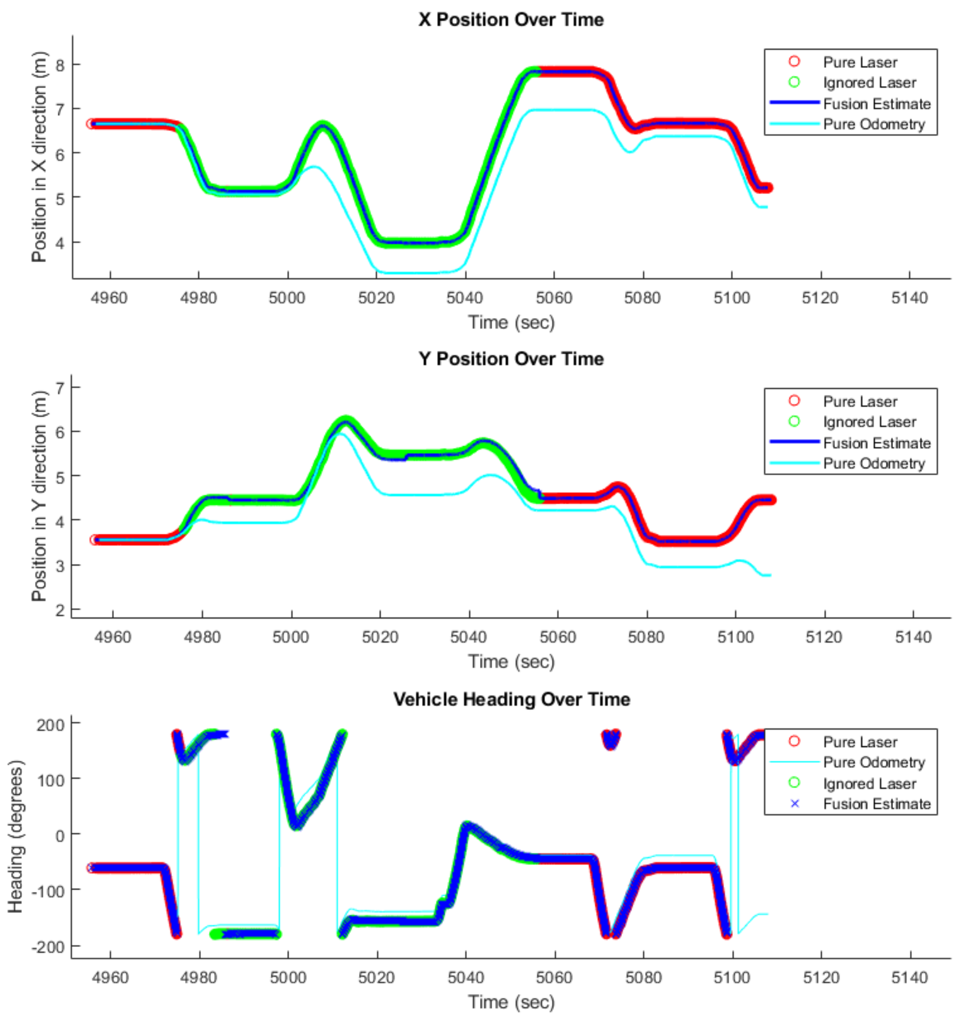


Figure 32: Vehicle position and orientation over time.

5.2.4 Estimation in Absence of Absolute Positioning

The final real-time test was conducted to assess the performance of the sensor fusion by not using the laser scanner at all during the motion, except for state initialization at the start. Unlike the long motion spanning a wide area in Rawseeds tests, the motion in real-time test was constrained to a much smaller timespan and area to assess the short-term performance by comparing the fusion trajectory to laser scanner trajectory. However, this comparison is limited to relative error in absence of a ground truth and RPE was not computed, since the scanner trajectory was not a high enough quality ground truth as can be observed in Figure 26, which might lead to a misleading indication of performance.

Throughout this experiment, the AGV was moved inside a 3 x 4 meters indoor space for about 90 seconds on a horizontal ground. The motion was intentionally made with high acceleration values with abrupt changes in speed and a nonlinear trajectory with many turns in order to make position, heading and tilt estimations more challenging. The estimated trajectory of the fusion without laser positioning compared to pure laser trajectory and pure odometry trajectory can be observed in Figures 33 and 34.

As seen, there was a drift in position that was increasing over time, but the scale of the drift was much smaller than that of odometry position estimate. This clearly demonstrates the advantage of using a fusion between odometry and IMU over relying on just a single source of information in the situations where absolute measurements are not available.

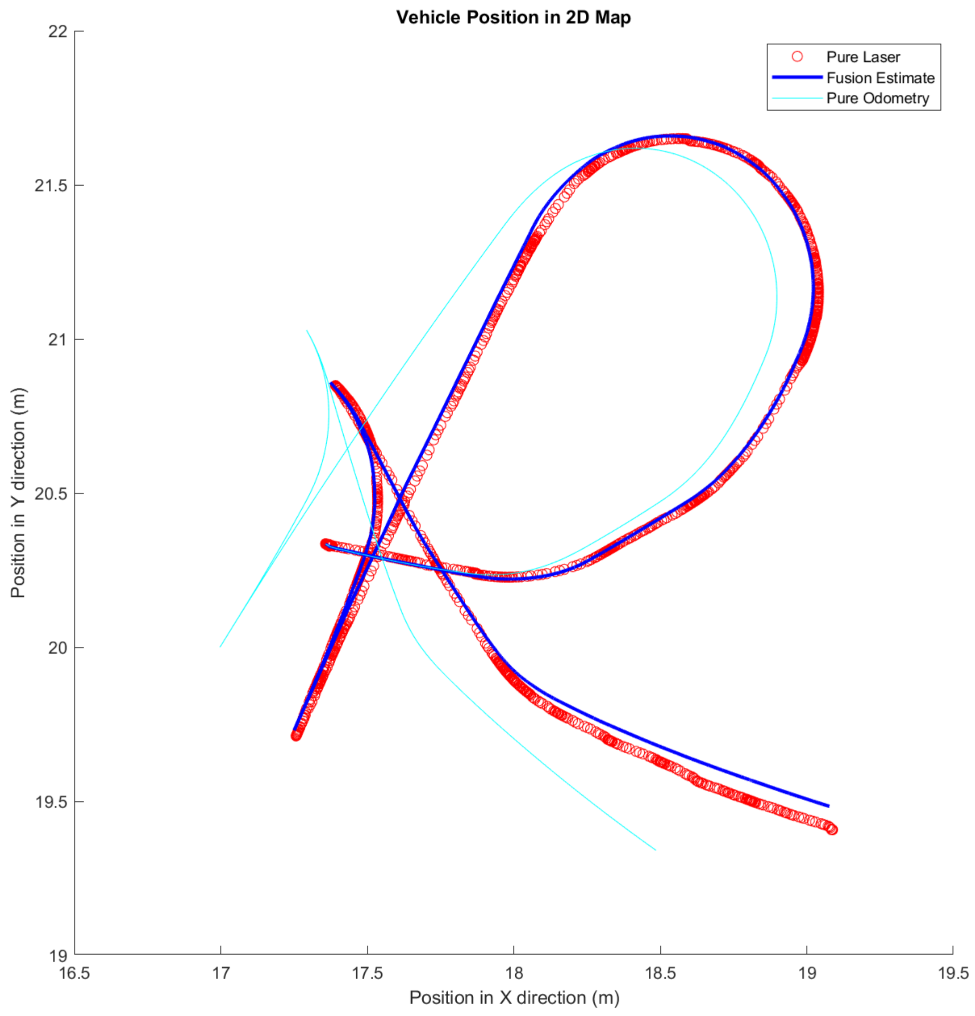


Figure 33: Vehicle position in 2D plane, obtained without using the laser measurements.

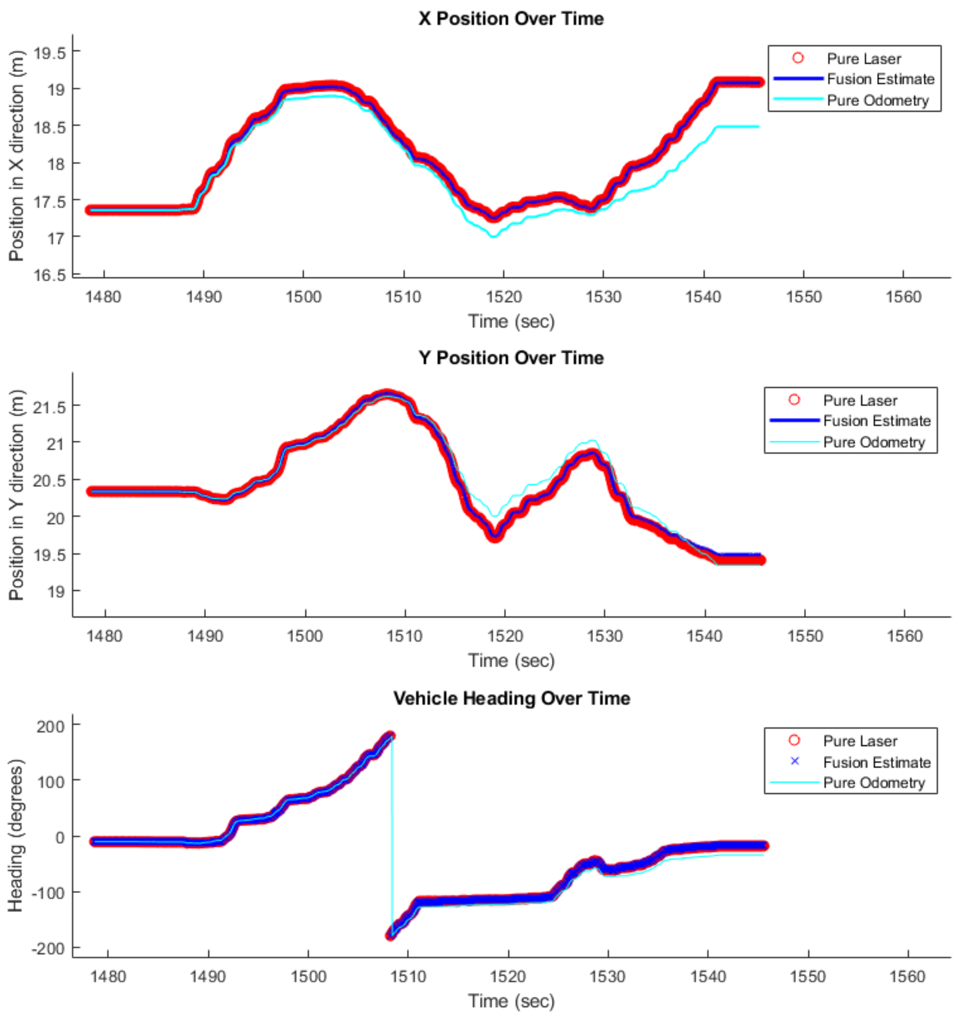


Figure 34: Vehicle position and orientation over time, obtained without using the laser measurements.

In addition to position and heading estimates, the estimated roll and pitch of the AGV during the same motion, which was performed on a horizontal ground, can be observed in Figure 35. As mentioned earlier, this motion included high acceleration values, which made the tilt calculation using Equations 33 and 34 noisy. Even though all of the navigation was performed on a horizontal surface, those equations indicated a tilt angle up to 10 degrees. However, with a correct choice of parameters that include high noise variance for them as shown in Table 7, the tilt estimate was confined to ±1 degrees as expected from a horizontal movement with slight oscillation due to uneven ground.

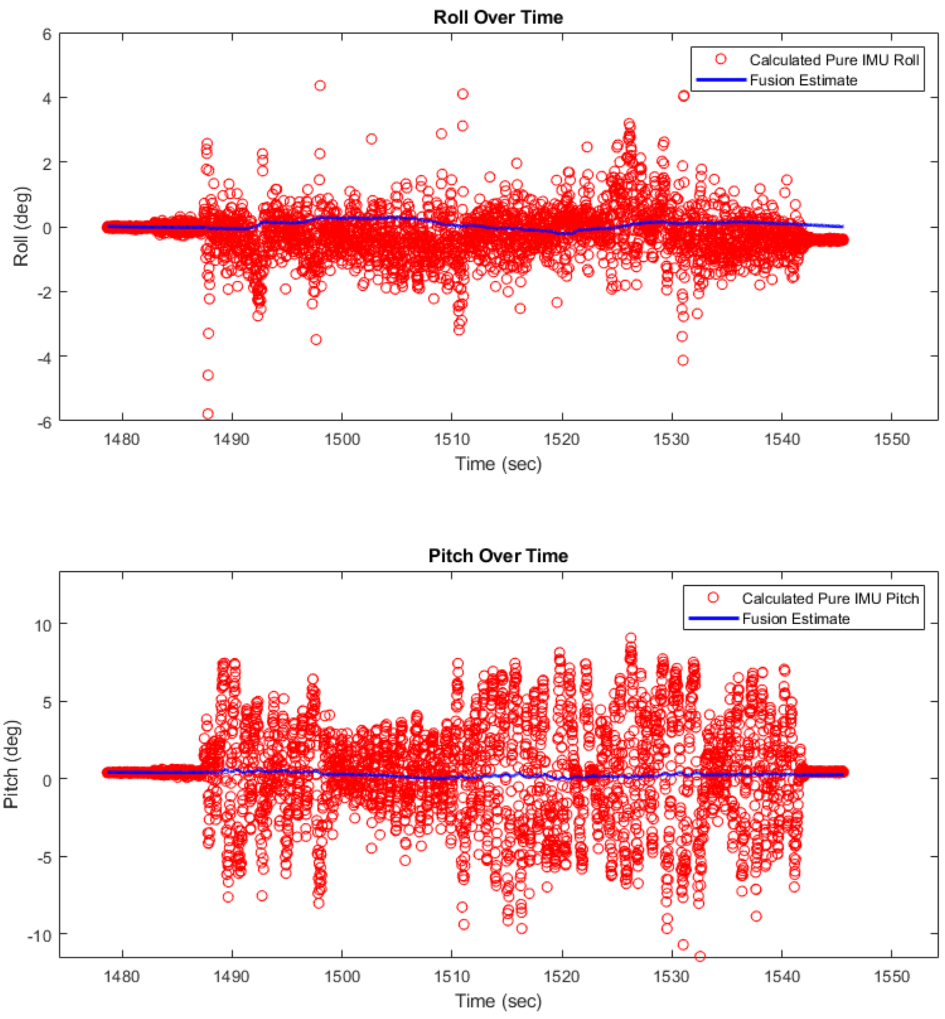


Figure 35: Vehicle roll and pitch estimation over time. The pure roll and pitch displayed in red refer to the results of Equations 33 and 34 obtained from accelerometer measurements.

5.3 Results and Discussion

The experiments consisted of two main groups. The first group utilized an open source multisensor dataset with an associated ground truth, while the second group consisted of real-time experiments performed with an AGV. In both groups of experiments, three scenarios of laser scanner measurements were considered: always available, partially available and not available during the motion.

In the first scenario, it was observed that using sensor fusion instead of a pure scan matching method had no noticeable effect on the mean translational error, but it greatly reduced the translational error variance by eliminating the greatest deviations from the ground truth. Furthermore, sensor fusion was observed to be useful in smoothing the estimated trajectory by eliminating minor deviations in the laser measurements from the true trajectory.

In the second scenario, the laser scanner was partially available during the motion. It was observed that in laser outage periods the results were different depending on motion characteristics. It was observed that while using sensor fusion had variable effects on individual outage periods, it had a better overall performance than relying purely on odometry. Furthermore, in real-time experiments, the amount of drift was reduced to very low values in shorter time periods by sensor fusion such that the estimate could be considered a safe trajectory to follow depending on operational needs.

The final experiment scenario simulated a situation in which the laser scanner was only available for state initialization. It was seen that the relative pose error of a fusion between IMU and odometry was much lower than that of using only odometry. However, the drift from the ground truth trajectory was still significant for both methods with long drive trajectories.

It was observed that in all these scenarios, the results from both experiment groups were consistent with each other. Different experiment cases included long and short motions, differential and omnidirectional vehicle kinematics, high and low speed motions, straight paths, curves and sharp turns. It can therefore be concluded that the experiment results are generalizable to a broad range of motion and vehicle kinematic types.

Both the full 3D algorithm with tilt estimation and a simplified 2D algorithm were tested, and their position and heading estimations were very similar in all experiments. However, it should be noted that in all test cases the AGV moved on a horizontal surface, so a 3D motion type was not tested.

The developed algorithm included a bias estimation method for IMU measurements. While a ground truth value of bias was not available to compare the results, the estimated bias value was in the expected range. In addition, the bias estimate was of slow varying nature, which is consistent with gyroscope bias characteristics. However, the estimation also included small scale jumps and oscillations when the laser measurements were made available after an outage period. While this did not have an adverse affect on the trajectory in the performed experiments, the estimated bias value might be wrong in a situation where the laser was available for a very short time and the oscillations did not have time to settle before another outage period. Even though this is an extremely unlikely situation in common AGV use cases, the potential negative effect from this remains as an open issue in the developed solution.

6 Conclusion and Future Work

In this thesis, a Kalman Filter based sensor fusion algorithm was developed that is capable of working with asynchronous laser scanner, odometry and IMU measurements. Some of these measurements included nonlinearities due to coordinate transformations and error terms with nonzero means due to inherent biases. The state vector in the Kalman Filter was chosen such that the prediction step of the algorithm was nonlinear but all three measurement update steps were linear. An Extended Kalman Filter was used to handle the nonlinearity in state transition while state augmentation was utilized to incorporate non-zero mean noises in the measurements.

The developed method was analyzed with an open source dataset with an associated ground truth, as well as real-time experiments performed with an AGV. It was shown that using sensor fusion was advantageous over relying only on the laser scanner measurements. Furthermore, it was observed that sensor fusion between odometry and IMU resulted in a less drift in time periods where absolute positioning is not available, compared to relying only on odometry. However, it was also seen that the effects of sensor fusion was different in each outage period, and during long time periods without absolute positioning the error between real trajectory and the estimated one was still very large. Therefore, it can be concluded that while using sensor fusion is generally beneficial over relying on a single sensor, but it does not guarantee a reliable position estimate in absence of an absolute positioning sensor.

Both a full 3D sensor fusion with tilt estimation and a simplified 2D sensor fusion with reduced states and simplified calculations were developed and tested, and no meaningful difference was observed between their localization performance. However, it should be noted that in all experiments conducted in this work, navigation took place on a horizontal and smooth surface. If the algorithm is used on a 3D motion, further experimental evaluation is needed to determine the performance of the simplified model.

It was also observed during the experiments that correct choice of parameters were necessary to achieve a high performance sensor fusion. Assigning low noise variance values to low-performance sensor readings or utilizing an unsuitable motion model depending on motion characteristics greatly reduced the localization performance. This makes it very challenging tune the algorithm for high performance with different sensors and vehicles. Utilizing adapting noise parameters that mitigates this issue is left for future work.

Additional future work can include further testing of this solution on navigation on inclined surfaces in order to further assess the tilt estimation. Furthermore, although Euler angles are sufficient to represent orientation in movements taking place on mainly horizontal surfaces, other representations such as quaternions are generally regarded superior in 3D navigation. Therefore, alternative state representations may be beneficial for navigation on inclined surfaces. In addition, different Kalman Filter variants such as Unscented Kalman Filter or Error State Kalman Filter can be tested to analyze whether they can provide better performance.

References

[1] Margrit Betke and Leonid Gurvits. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, 13(2):251–263, 1997.

[2] A. Azenha and A. Carvalho. A neural network approach for agv localization using trilateration. In *2009 35th Annual Conference of IEEE Industrial Electronics*, pages 2699–2702, Nov 2009.

[3] Luıs Garrote, Miguel Torres, Tiago Barros, Joao Perdiz, Cristiano Premebida, and Urbano J Nunes. Mobile robot localization with reinforcement learning map update decision aided by an absolute indoor positioning system. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[4] Leandro B Marinho, Pedro P Reboucas Filho, Jefferson S Almeida, João Wellington M Souza, Amauri H Souza Junior, and Victor Hugo C de Albuquerque. A novel mobile robot localization approach based on classification with rejection option using computer vision. *Computers & Electrical Engineering*, 68:26–43, 2018.

[5] Jiraphan Inthiam and Chirdpong Deelertpaiboon. Self-localization and navigation of holonomic mobile robot using omni-directional wheel odometry. In *TENCON 2014-2014 IEEE Region 10 Conference*, pages 1–5. IEEE, 2014.

[6] Nakju Lett Doh, Howie Choset, and Wan Kyun Chung. Relative localization using path odometry information. *Autonomous Robots*, 21(2):143–154, 2006.

[7] H-J Von Der Hardt, Didier Wolf, and René Husson. The dead reckoning localization system of the wheeled mobile robot romane. In *1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems (Cat. No. 96TH8242)*, pages 603–610. IEEE, 1996.

[8] Patric Jensfelt. *Approaches to Mobile Robot Localization in Indoor Environments.* PhD thesis, KTH, 2001.

[9] Fabjan Kallasi, Dario Lodi Rizzini, and Stefano Caselli. Fast keypoint features from laser scanner for robot localization and mapping. *IEEE Robotics and Automation Letters*, 1(1):176–183, 2016.

[10] Felipe Espinosa, Carlos Santos, Marta Marrón-Romera, Daniel Pizarro, Fernando Valdés, and Javier Dongil. Odometry and laser scanner fusion based on a discrete extended kalman filter for robotic platooning guidance. *Sensors*, 11(9):8339–8357, 2011.

[11] Davide Ronzoni, Roberto Olmi, Cristian Secchi, and Cesare Fantuzzi. Agv global localization using indistinguishable artificial landmarks. In *2011 IEEE International Conference on Robotics and Automation*, pages 287–292. IEEE, 2011.

[12] Fredrik Gustafsson. *Statistical Sensor Fusion.* Studentlitteratur AB, 2010.

[13] Gerasimos G. Rigatos. Sensor fusion-based dynamic positioning of ships using extended kalman and particle filtering. *Robotica*, 31(3):389–403, 2013.

[14] Zhe Chen et al. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, 182(1):1–69, 2003.

[15] Dan Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches.* John Wiley & Sons, 2006.

[16] Jungmin Kim, Yountae Kim, and Sungshin Kim. An accurate localization for mobile robot using extended kalman filter and sensor fusion. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 2928–2933. IEEE, 2008.

[17] Muhammad Latif Anjum, Jaehong Park, Wonsang Hwang, Hyun-il Kwon, Jong-hyeon Kim, Changhun Lee, Kwang-soo Kim, et al. Sensor data fusion using unscented kalman filter for accurate localization of mobile robots. In *International Conference on Control, Automation and Systems 2010*, pages 947–952. IEEE, 2010.

[18] Mohamed M Atia, Shifei Liu, Heba Nematallah, Tashfeen B Karamat, and Aboelmagd Noureldin. Integrated indoor navigation system for ground vehicles with automatic 3-d alignment and position initialization. *IEEE Transactions on Vehicular Technology*, 64(4):1279–1292, 2015.

[19] Aboelmagd Noureldin, Tashfeen B Karamat, and Jacques Georgy. Basic navigational mathematics, reference frames and the earth's geometry. In *Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration*, pages 21–63. Springer, 2013.

[20] Mengxiao Chen, Shaowu Yang, Xiaodong Yi, and Dan Wu. Real-time 3d mapping using a 2d laser scanner and imu-aided visual slam. In *2017 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 297–302. IEEE, 2017.

[21] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics.* Springer, 2016.

[22] Sheila Widnall. Lecture l3-vectors, matrices and coordinate transformations. *Dynamics*, pages 1–15, 01 2009.

[23] Michael S Triantafyllou and Franz S Hover. *Maneuvering and Control of Marine Vehicles.* MIT, Cambridge, MA, 2003.

[24] Thor I Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control.* John Wiley & Sons, 2011.

[25] Anthony Kim and MF Golnaraghi. A quaternion-based orientation estimation algorithm using an inertial measurement unit. In *PLANS 2004. Position Location and Navigation Symposium (IEEE Cat. No. 04CH37556)*, pages 268–272. IEEE, 2004.

[26] Nicholas Carlevaris-Bianco, Arash K Ushani, and Ryan M Eustice. University of michigan north campus long-term vision and lidar dataset. *The International Journal of Robotics Research*, 35(9):1023–1035, 2016.

[27] Gregory Dudek and Michael Jenkin. *Computational principles of mobile robotics*. Cambridge university press, 2010.

[28] Kyung-Seok Byun, Sung-Jae Kim, and Jae-Bok Song. Design of a four-wheeled omnidirectional mobile robot with variable wheel arrangement mechanism. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 1, pages 720–725. IEEE, 2002.

[29] Mark Ashmore and Nick Barnes. Omni-drive robot motion on curved paths: The fastest path between two points is not a straight-line. In *Australian Joint Conference on Artificial Intelligence*, pages 225–236. Springer, 2002.

[30] Thomas Epton. Odometry correction of a mobile robot using a range-finding laser. *All Theses*, 2007.

[31] Kok Seng Chong and Lindsay Kleeman. Accurate odometry and error modelling for a mobile robot. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 2783–2788. IEEE, 1997.

[32] Nuwan Ganganath and Henry Leung. Mobile robot localization using odometry and kinect sensor. In *2012 IEEE International Conference on Emerging Signal Processing Applications*, pages 91–94. IEEE, 2012.

[33] Halil Soken and Shin-Ichiro Sakai. Magnetometer calibration for advanced small satellite missions. In *30th International Symposium on Space Technology and Science*, 2015.

[34] Manon Kok, Jeroen D Hol, and Thomas B Schön. Using inertial sensors for position and orientation estimation. *arXiv Preprint arXiv:1704.06053*, 2017.

[35] Martti Kirkko-Jaakkola, Jussi Collin, and Jarmo Takala. Bias prediction for mems gyroscopes. *IEEE Sensors Journal*, 12(6):2157–2163, 2012.

[36] Zhi-hua Lu, Meng-yao Zhu, Qing-wei Ye, and Yu Zhou. Performance analysis of two em-based measurement bias estimation processes for tracking systems. *Frontiers of Information Technology & Electronic Engineering*, 19(9):1151–1165, 2018.

[37] Inchara Lakshminarayan and Divya Rao. Kalman filter based estimation of constant angular rate bias for mems gyroscope. In *Proceedings of IEEE TechSym 2014 Satellite Conference, VIT University*, 03 2014.

[38] Jonathan R Nistler and Majura F Selekwa. Gravity compensation in accelerometer measurements for robot navigation on inclined surfaces. *Procedia Computer Science*, 6:413–418, 2011.

[39] Mark Pedley. Tilt sensing using a three-axis accelerometer. *Freescale Semiconductor Application Note*, 1:2012–2013, 2013.

[40] Heikki Hyyti and Arto Visala. A dcm based attitude estimation algorithm for low-cost mems imus. *International Journal of Navigation & Observation*, 2015.

[41] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[42] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to Autonomous Mobile Robots*. MIT Press, 2011.

[43] Héber Sobreira, Carlos M Costa, Ivo Sousa, Luis Rocha, José Lima, PCMA Farias, Paulo Costa, and A Paulo Moreira. Map-matching algorithms for robot self-localization: a comparison between perfect match, iterative closest point and normal distributions transform. *Journal of Intelligent & Robotic Systems*, 93(3-4):533–546, 2019.

[44] Jihoon Seong, Jiwoong Kim, and Woojin Chung. Mobile robot localization using indistinguishable artificial landmarks. In *2013 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 222–224. IEEE, 2013.

[45] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006.

[46] Denis F Wolf and Gaurav S Sukhatme. Mobile robot simultaneous localization and mapping in dynamic environments. *Autonomous Robots*, 19(1):53–65, 2005.

[47] Jaromir Konecny, Michal Prauzek, and Jakub Hlavica. Icp algorithm in mobile robot navigation: Analysis of computational demands in embedded solutions. *IFAC-PapersOnLine*, 49(25):396–400, 2016.

[48] Dirk Hahnel, Rudolph Triebel, Wolfram Burgard, and Sebastian Thrun. Map building with mobile robots in dynamic environments. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1557–1563. IEEE, 2003.

[49] Mina Henein, Gerard Kennedy, Viorela Ila, and Robert Mahony. Simultaneous localization and mapping with dynamic rigid objects. *arXiv Preprint arXiv:1805.03800*, 2018.

[50] Andress Nuchter, Hartmut Surmann, Kai Lingemann, Joachim Hertzberg, and Sebastian Thrun. 6d slam with an application in autonomous mine mapping. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 2, pages 1998–2003. IEEE, 2004.

[51] Jin Liang Li and You Xia Sun. Mapping of rescue environment based on ndt scan matching. In *Advanced Materials Research*, volume 760, pages 928–933. Trans Tech Publ, 2013.

[52] Hugh Durrant-Whyte et al. Introduction to estimation and the kalman filter. *Australian Centre for Field Robotics*, 28(3):65–94, 2001.

[53] Lindsay Kleeman. Understanding and applying kalman filtering. In *Proceedings of the Second Workshop on Perceptive Systems, Curtin University of Technology, Perth Western Australia (25-26 January 1996)*, 1996.

[54] François Auger, Mickael Hilairet, Josep M Guerrero, Eric Monmasson, Teresa Orlowska-Kowalska, and Seiichiro Katsura. Industrial applications of the kalman filter: A review. *IEEE Transactions on Industrial Electronics*, 60(12):5458–5471, 2013.

[55] Gabriel A Terejanu. Extended Kalman Filter Tutorial, Department of Computer Science and Engineering, University at Buffalo. http://users.ices.utexas.edu/~terejanu/files/tutorialEKF.pdf, 2008. [Online; accessed 16-July-2020].

[56] Joan Sola. Quaternion kinematics for the error-state kf. *Laboratoire dAnalyse et dArchitecture des Systemes-Centre National de la Recherche Scientifique (LAAS-CNRS), Toulouse, France, Tech. Rep*, 2012.

[57] Mingyang Li and Anastasios I Mourikis. Improving the accuracy of ekf-based visual-inertial odometry. In *2012 IEEE International Conference on Robotics and Automation*, pages 828–835. IEEE, 2012.

[58] Roger Labbe. Kalman and bayesian filters in python. https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python, 2019. [Book Published in Web, accessed 29-Jul-2020].

[59] Huazhen Fang, Mulugeta A Haile, and Yebin Wang. Robust extended kalman filtering for systems with measurement outliers. *arXiv Preprint arXiv:1904.00335*, 2019.

[60] Antonio Ramón Jiménez, Fernando Seco, José Carlos Prieto, and Jorge Guevara. Indoor pedestrian navigation using an ins/ekf framework for yaw drift reduction and a foot-mounted imu. In *7th Workshop on Positioning, Navigation and Communication*, pages 135–143. IEEE, 2010.

[61] Hamza Benzerrouk, Alexander Nebylov, Hassen Salhi, and Pau Closas. Mems imu/zupt based cubature kalman filter applied to pedestrian navigation system. In *Proceedings of International Electronic Conference on Sensors and Applications*, pages 1–7, 2014.

[62] Wenchao Zhang, Xianghong Li, Dongyan Wei, Xinchun Ji, and Hong Yuan. A foot-mounted pdr system based on imu/ekf+ hmm+ zupt+ zaru+ hdr+ compass algorithm. In *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–5. IEEE, 2017.

[63] Andrea Bonarini, Wolfram Burgard, Giulio Fontana, Matteo Matteucci, Domenico Giorgio Sorrenti, and Juan Domingo Tardos. Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets. In *Proceedings of IROS'06 Workshop on Benchmarks in Robotics Research*, 2006.

[64] Simone Ceriani, Giulio Fontana, Alessandro Giusti, Daniele Marzorati, Matteo Matteucci, Davide Migliore, Davide Rizzi, Domenico G Sorrenti, and Pierluigi Taddei. Rawseeds ground truth collection systems for indoor self-localization and mapping. *Autonomous Robots*, 27(4):353–371, 2009.

[65] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580. IEEE, 2012.

[66] Navitec Systems. https://www.navitecsystems.com/. [Online; accessed 25-May-2020].