2021-01-20

Wheel Odometry Aided Visual-Inertial Odometry in Winter Urban Environments

Huang, Cheng

UNIVERSITY OF CALGARY

Wheel Odometry Aided Visual-Inertial Odometry in Winter Urban Environments

by

Cheng Huang

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN GEOMATICS ENGINEERING

CALGARY, ALBERTA

JANUARY, 2021

Abstract

Over the last decade or so, the world has witnessed the rapid changes in the way people drive. How to ensure the navigation performance in challenging environments such as complex urban canyon environments or winter road environment with a relatively low-cost navigation system has become a popular research topic. Global Navigation Satellite System (GNSS) positioning is commonly used for land vehicle navigation. However, the accuracy of GNSS positioning is reduced in such challenging environments due to obstructions and multipath effects. Thus, the development of an alternative, accurate, inexpensive, and self-contained land vehicle navigation systems to bridge the GNSS gaps is significant for land vehicle navigation systems. Visual-inertial odometry (VIO) is an accurate, inexpensive, and complementary approach for land vehicle navigation in GNSS signal-denied environments. VIO is subject to scale drift because it estimates forward direction translation using distant feature points that are generally located only in the forward direction. Wheel odometer measurements can be obtained from the CANBUS interface of most modern passenger vehicles and these provide reliable estimates of the forward wheel speed. In this thesis, an innovative approach to incorporate wheel odometry (WO) and non-holonomic constraints (NHC) together with tightly-coupled monocular visual-inertial odometry using the Multi-State Constraint Kalman Filter (MSCKF) is proposed and implemented. The algorithm is first validated using the public KITTI Dataset [1] with simulated wheel odometer data. Then, the KAIST Complex Urban Dataset [2] is used to test the performance of IMU+Vision+WO integration system in urban canyon environments. Winter driving data is collected in Calgary and used to evaluate the influence of winter road conditions on the proposed algorithm. The results demonstrate that WO and NHC are able to control the scale drift, and as a result are able to control both scale and orientation over longer periods than

IMU+Vision alone. IMU+Vision+WO achieved 1.814 m horizontal position error in a 1-minute drive in an urban canyon environment in the KAIST Complex Urban Dataset and 19.649 m and 3.456 m horizontal position errors in two 1-minute drives in our Calgary winter urban environment. The results demonstrate that IMU+Vision+WO is a very promising method to bridge the GNSS outages and performs very well in some challenging environments.

Acknowledgments

Being able to study at PLAN group and Geomatics department of University of Calgary will always be an unforgettable and important experience of my life. First of all, I would like to express my deepest gratitude to my supervisor, Dr. Kyle O’Keefe for sharing his insights and knowledge, and always be so supportive for me. It has been a very rewarding journey to work with you.

I would also like to thank all the professors for broadening my horizons to the most cutting-edge geomatics research: Dr. Kyle O’Keefe, Dr. Yang Gao, Dr. Naser El-Sheimy and Dr. Mozhdeh Shahbazi. Your amazing lectures really laid a solid foundation for my future research and career. In addition, I would like to extend my gratitude to Dr. Yang Gao, Dr. Naser El-Sheimy and Dr. Derek Lichti for agreeing to be on my defense committee. Also, I would like to thank Dr. Alex Bruton and Dr. Ivan Detchev for letting me to be your teaching assistant.

Special thanks go to my friend Yang (River) Jiang. Thank you for your generosity for providing your insights and help during the data collection of my project. I would also like to thank all the PLAN Group members: Chandra, Asal, Paul V. G., Dongyu, Paul G., Rene, Andreas and Maliha for all the fun discussions we had. I would like to thank my friends, Tian Jin, Eric Wang and Changlin Yang for all the good time we spent together.

My deepest thanks go to my parents and my beloved grandmother. Thank you for always being at my side no matter what.

2020 was a very challenging year for me, and probably, for most people. Finally, I would like to express my sincere gratitude to all the health care workers around the globe. You are the true heroes.

Dedication

To:

my beloved mother Wenjie Zhang

my father Keqing Huang

and my grandmother Yuxiu Zhu.

Table of Contents

List of Tables

List of Figures and Illustrations

List of Symbols

Abbreviations	**Definitions**
b	bias
f	focal length
H	design matrix
J	Jacobian matrix
K	intrinsic calibration matrix
n	Process noise
P	projection matrix
${}^{b}p_{a}$	position of a described in frame b
${}_{b}^{a}q$	quaternion vector that represents the rotation from frame b to frame a
R	rotation matrix
X	3×1 coordinate vector
x	2×1 coordinate vector
α	yaw angle
β	pitch angle
γ	roll angle
λ	scale factor
δ	error
Φ	transition matrix

List of Abbreviations

Abbreviations	**Definitions**
ABS	Antilock Breaking System
ADAS	Advanced Driver Assistance Systems
AOA	Angle of Arrival
BA	Bundle adjustment
BLE	Bluetooth Low Energy
CNNs	Convolutional Neural Networks
CUPT	Coordinate Updates
DLT	Direct Linear Transformation
DoF	Degree of Freedom
DR	Dead-Reckoning
DSO	Direct Sparse Odometry
ECEF	Earth-Centered Earth Fixed Frame
ECI	Earth-Centered Inertial Frame
EKF	Extended Kalman Filter
EOP	Extrinsic Orientation Parameters
FLANN	Fast approximate nearest neighbor
FoV	Field of View
GNSS	Global Navigation Satellite System
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System

IOP	Intrinsic Orientation Parameters
KF	Kalman Filter
LLF	Local Level-Frame
MEMS	Microelectromechanical system
MSAC	M-estimator SAC
MSCKF	Multi-State Constrained Kalman Filter
NFC	Near-Field Communication
NHC	Non-holonomic Constraints
PDR	Pedestrian Dead Reckoning
PnP	Perspective-n-Point
PPP	Precise Point Positioning
PRN	Pseudorandom Noise
PTAM	Parallel Tracking and Mapping
RANSAC	Random sample consensus
RFID	Radio Frequency Identification
RSSI	Received Signal Strength Indicator
RTK	Real-Time Kinematic
SFM	Structure from motion
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SVO	Semi-direct Visual Odometry
SWF	Sliding window filter
TDOA	Time Difference of Arrival

TOA	Time of Arrival
UKF	Unscented Kalman Filter
UWB	Ultrawideband
VIO	Visual-Inertial Odometry
VO	Visual Odometry
V-SLAM	Visual simultaneous localization and mapping
WLAN	Wireless Local Area Network
WLS	Weighted least squares
WO	Wheel Odometry
ZUPT	Zero-velocity Updates

Chapter 1 INTRODUCTION

Over the last decade or so, the world has witnessed rapid changes in the way people drive. Advanced Driver Assistance Systems (ADAS) as well as self-driving cars have received a tremendous amount of attention from both academia and industry due to some fundamental advancements in Simultaneous Localization and Mapping (SLAM), satellite navigation and sensor integration technology. With the help of all these sensors, cars are equipped with “eyes” to perceive the environment and become more intelligent like never before. Among all auxiliary sensors, due to the low cost and rich information, cameras have been the most promising and fundamental sensors for cognitive navigation. From a localization and navigation perspective, Visual Odometry (VO) or Visual SLAM is the answer to perform Dead-Reckoning (DR) in an unknown environment employing cameras. As for land vehicle motion sensors, Wheel Odometry (WO) and Inertial Measurement Units (IMU) are the most common ones mounted on cars. Ensuring land vehicle navigation systems function properly in challenging environments, such as urban canyon environments and winter road conditions, has become a very popular research topic in both academia and industry. This thesis investigates the integration of cameras and wheel odometry with inertial navigation in such challenging environments.

This chapter provides an overview of the development of the land vehicle navigation methodologies and the limitations of the respective methods. The motivations, objectives, and contributions of this research are then presented.

1.1 Land Vehicle Navigation

In this section, the existing and most commonly used technologies for land vehicle navigation systems will be briefly reviewed.

For land vehicle navigation, Global Navigation Satellite Systems (GNSS) are always the first to be mentioned due to the advancements of the related satellite navigation technologies. However, GNSS receivers have many vulnerabilities and monitoring the integrity of their positioning solutions is both theoretically and practically difficult, thus, external support from additional information sources is required to obtain the desired accuracy, integrity, availability and continuity for land vehicle navigation systems [3]. Generally speaking, the most commonly used information sources for land vehicle navigation can be categorized as: (1) GNSS/RF-based Positioning; (2) Vehicle motion sensors; (3) Road maps; (4) Visual sensors and (5) Vehicle motion models [3].

In addition, many land vehicles in professional services nowadays, such as ambulances, buses, fire trucks and etc., are equipped with navigation systems that not only show the current location but constantly communicate the vehicle location to a monitoring center as well [3]. Moreover, further development of intelligent transport system (ITS) applications, such as advanced driver assistance systems (ADAS), traffic control, automatic positioning of accidents, electronic toll collection, goods tracking, etc., requires not only navigation systems with higher accuracy but also better reliability and integrity [4], i.e., redundant information sources are needed.

The ultimate goal of ITS is to endow land vehicles with intelligence to achieve autonomous driving. For the past ten years, autonomous vehicles (AVs) have drawn significant amount of attention in both academia and industry. AV technologies have promised to decrease transportation costs, increase safety, and some have suggested they might increase accessibility to low-income households and persons with mobility issues [5].

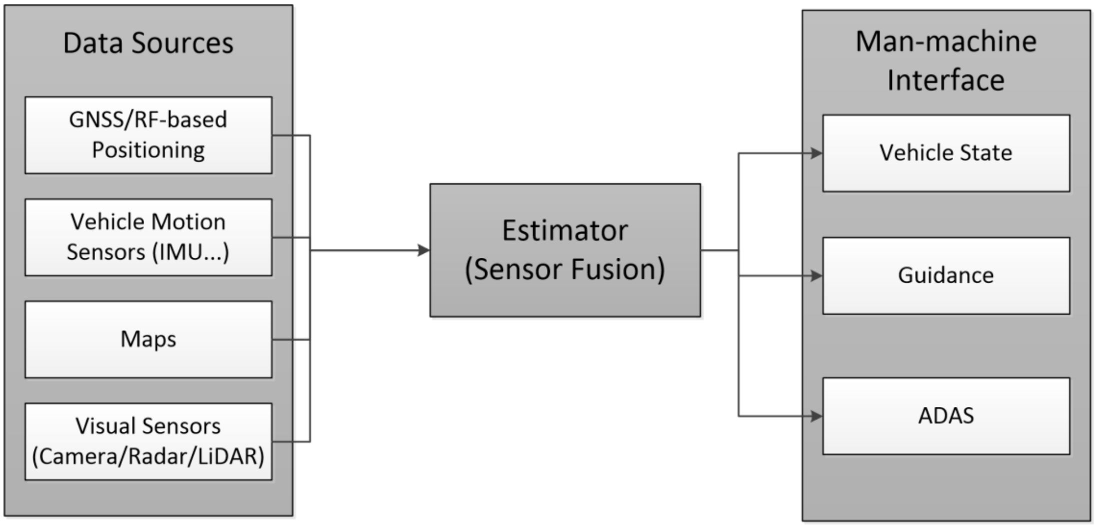

Data Sources
GNSS/RF-based Positioning
Vehicle Motion Sensors (IMU…)
Maps
Visual Sensors (Camera/Radar/LiDAR)
Estimator (Sensor Fusion)
Man-machine Interface
Vehicle State
Guidance
ADAS

Figure 1-1: General data sources and Man-machine interface for land vehicle navigation systems after [3].

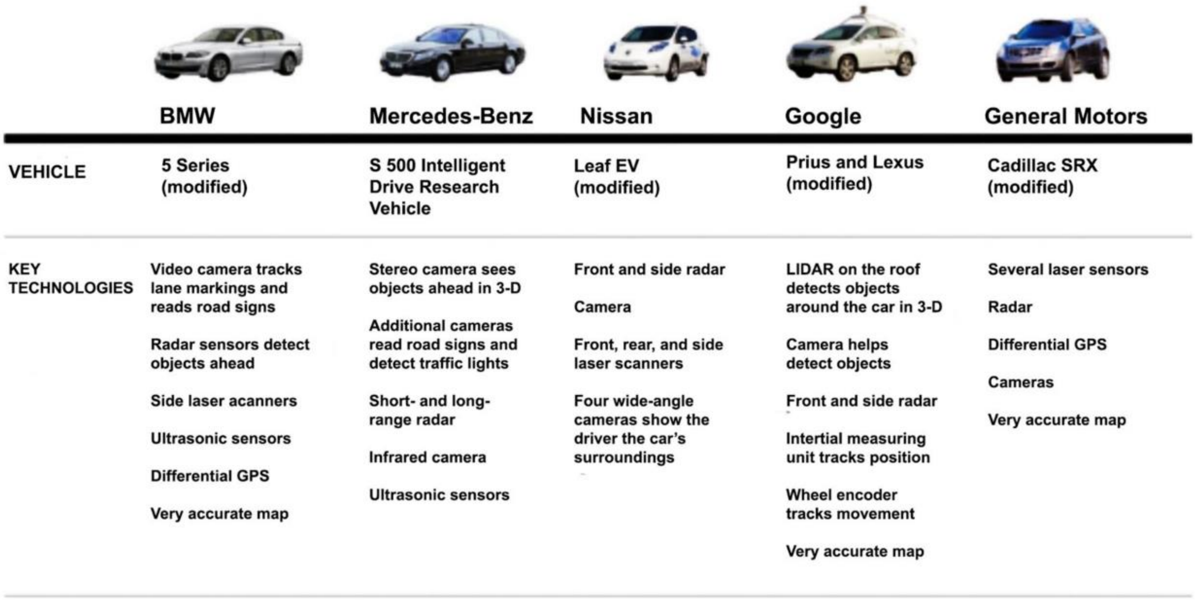
In addition to a large body of publicly available academic research and work going on privately in industry, there have been a number of research projects and competitions funded to expand the potential of AV. The concept dates back to as far as early 1920s [6]. The first modern AV competition was DARPA Grand Challenge [7]. In April 2017, Waymo (a subsidiary of Google) started a limited trial of a self-driving taxi service in Phoenix, Arizona. On December 5, 2018, the service launched a commercial self-driving car service called "Waymo One". Users in the Phoenix metropolitan area use an app to request a pick-up. In May 2020, Waymo announced its first outside funding round of $2.25 billion. On June 25, 2020, Waymo announced a partnership with Volvo to integrate Waymo’s self-driving technology into their vehicles [8].

The automobile industry has defined five levels of automation for AVs [5] that are described as follows:

- Level 0: At all times, the driver has complete and sole command and control of the vehicle with respect to steering, braking, throttle and motive power.

- Level 1: Some specific control function(s) such as electronic stability control or pre-charged brakes is(are) automated.
- Level 2: At least two main control functions such as adaptive cruise control in combination with lane centering are automated.
- Level 3: Under certain traffic or environmental conditions, the driver cedes full control of all safety–critical functions and relies heavily on the vehicle to watch for any changes in conditions requiring transition to driver control. The driver will be required to resume control of the vehicle, but with sufficient transition time.
- Level 4: The vehicle is intelligently designed to monitor roadway conditions and act solo, performing all safety–critical driving functions for an entire trip (a fully driverless level).

Currently, all the available AVs in the market are at Level 2. In 2019, Audi released Audi A8 and claimed it has reached Level 3. Even though academia and industry have invested significant effort to achieve full automation of AV, according to MIT Technology Review [9], due to the restrictions of technology and policy, there is a long way to go before level 3 or 4 vehicle are common.



	BMW	Mercedes-Benz	Nissan	Google	General Motors
VEHICLE	5 Series (modified)	S 500 Intelligent Drive Research Vehicle	Leaf EV (modified)	Prius and Lexus (modified)	Cadillac SRX (modified)
KEY TECHNOLOGIES	Video camera tracks lane markings and reads road signs Radar sensors detect objects ahead Side laser acanners Ultrasonic sensors Differential GPS Very accurate map	Stereo camera sees objects ahead in 3-D Additional cameras read road signs and detect traffic lights Short- and long-range radar Infrared camera Ultrasonic sensors	Front and side radar Camera Front, rear, and side laser scanners Four wide-angle cameras show the driver the car's surroundings	LIDAR on the roof detects objects around the car in 3-D Camera helps detect objects Front and side radar Intertial measuring unit tracks position Wheel encoder tracks movement Very accurate map	Several laser sensors Radar Differential GPS Cameras Very accurate map

Figure 1-2: Expected specification of AV by the year 2020 from [5]

Before reviewing the various technologies, starting with GNSS/RF-based positioning technologies, a basic question must be addressed: what is navigation? According to The Concise Oxford Dictionary, navigation is "any of several methods of determining or planning a ship's or aircraft's position and course by geometry, astronomy, radio signals, etc.". This concept can be interpreted in two ways. The first is to determine the position, velocity and attitude of a moving object with respect to a known reference. The second is the path planning of the object to navigate it from one location to another [10]. In modern navigation engineering, these tasks can be divided into two parts: perception and path planning. In this thesis, perception will be a major focus.

Most of the current navigation techniques can be categorized as position fixing or dead reckoning (DR). Position fixing is accomplished by measuring range and/or bearing to known objects. Bearing and elevation measurements can be obtained via theodolite, magnetic compass, camera and etc. Ranging measurements can be obtained by using many sensors including radio signals, lasers, radar, lidar or ultrasound. In geomatics applications, 3D position fixing can be achieved by trilateration (multilateration) and intersection methods.

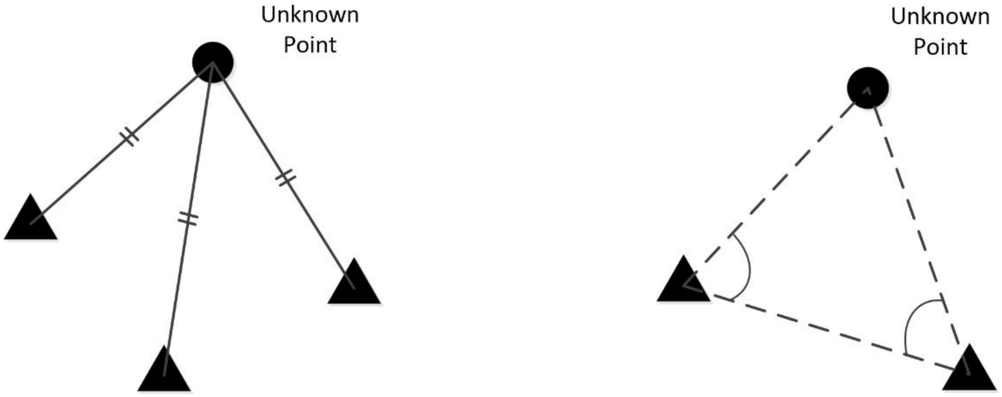


Figure 1-3: Concepts of Trilateration and Intersection

Dead Reckoning either measures the change in position or measures velocity and attitude change and integrates these over time [10]. The speed or distance traveled is measured in the body coordinate frame. Thus, orientation of the object has to be measured to calculate the rotation matrix to the reference frame. For pedestrians, distance and velocity measurements were traditionally obtained by counting paces. Nowadays, for pedestrian dead reckoning (PDR), accelerometers are used to determine step length. For land vehicle DR, an odometer can be used to count the rotation of a wheel to calculate the velocity and traveled distance. Contemporary velocity measurement methods include Doppler radar and integrating accelerometer measurements within an inertial navigation system.

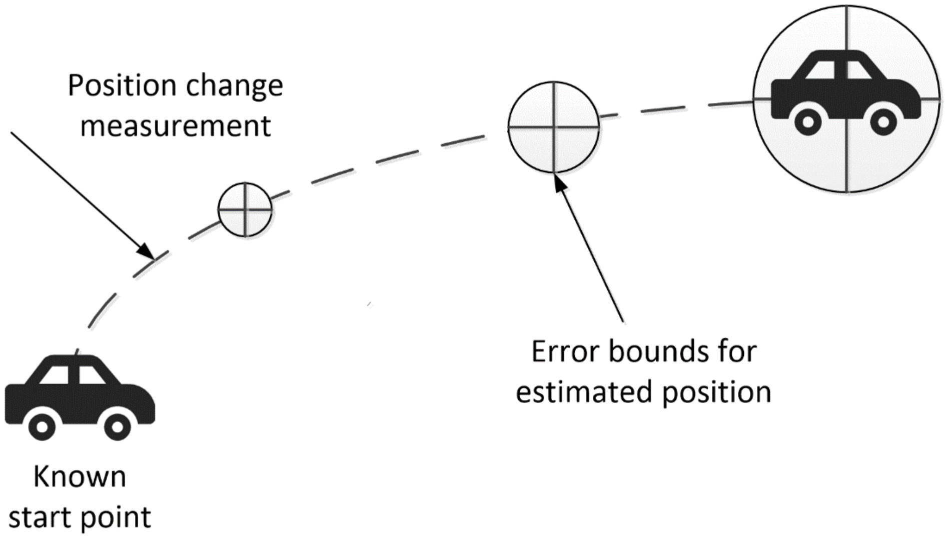

Position change measurement
Error bounds for estimated position
Known start point

Figure 1-4: Concept of DR

1.1.1 GNSS/RF-based positioning methods

The GNSS/RF-based positioning are the most widely-applied position fixing methods in the world nowadays. For land vehicle applications, the most common of modern in-car navigation systems match the position information from a GNSS receiver with digital map to estimate the vehicle position on the road. Generally speaking, there are three types of measurements in the RF-based positioning systems: Angle of Arrival (AOA), Time of Arrival (TOA), Time Difference of Arrival (TDOA). A fourth measurement, Received Signal Strength Indicator (RSSI), can be treated as another method of TOA.

A satellite navigation system is a system that uses satellites to provide autonomous geo-spatial positioning and timing services [11]. The foundation of satellite navigation system was the US TRANSIT system. TRANSIT used Doppler positioning, which provided only one independent two-dimensional position fix per satellite pass [12]. To date, there are three fully operational Global Navigation Satellite Systems (GNSSs): the United States' Global Positioning System (GPS), Russia's Global Navigation Satellite System (GLONASS) and China's BeiDou Navigation Satellite System (BDS). The European Union's Galileo scheduled to be fully operational by the end of 2020. Besides, there are two regional satellite navigation systems, Japan's Quasi-Zenith Satellite System (QZSS) and the Indian Regional Navigation Satellite System (IRNSS), designed to enhance GPS’s accuracy. For improved error modelling in the navigation receiver, geostationary satellites as part of Satellite Based Augmentation Systems (SBASs) send local correction data. The importance and applications of GNSS technology is profound that many countries are putting it on a national strategic level. Over the last few decades, the Global Navigation Satellite Systems have made some exciting and profound advancements. The GNSS measurements are comprised of Pseudorandom Noise (PRN) codes and carrier wave for the GNSS signal. Measurements based on PRN modulation are unambiguous, the positioning accuracy of using PRN codes is from meters to sub-meter level. However, for some specific applications, such as surveying, earthquake monitoring, and vehicle navigation in challenging environments, higher positioning accuracy is required [13]. The carrier wave for the GNSS signal is a sine wave with a period of less than 1 meter (19 cm for L1), allowing for more precise measurements. Real-Time Kinematic (RTK) is a differential GNSS technique that uses carrier-based ranging to calculate position that is more precise than code-based positioning. The common RTK GNSS positioning accuracy is up to 1 cm + 1 ppm. On the other hand, Precise Point Positioning (PPP) does not require base stations to remove atmospheric and clock errors, instead, it depends on GNSS satellite clock and orbit correction from a network of global reference stations (such as IGS). Generally speaking, the PPP GNSS positioning accuracy is up to 3 cm. A typical PPP solution requires a period of time to converge to decimeter accuracy in order to resolve any local biases such as the atmospheric conditions, multipath environment and satellite geometry. The actual accuracy achieved and the convergence time required is dependent on the quality of the corrections and how they are applied in the receiver [14].

Despite the fact that the current GNSS technologies have already achieved very high accuracy positioning solutions in many scenarios, there are still challenges needed to be tackled. First of all, a GNSS receiver needs a clear line-of-sight to the satellites it is tracking. If the signal is blocked by objects like buildings or trees, the receiver cannot function normally, especially in urban canyon environments. Also, if the signal from a satellite arrives at the GNSS receiver via multiple paths due to reflection and diffraction, the nondirect-path signal will distort the signal and degrade the GNSS receiver performance. This effect is called multipath effect [15]. Second, due to the long distance (around 20200 km) between the GNSS satellites and the receiver on earth, the received GNSS signals are so weak that they can be easily squelched by natural or man-made interference [15].

Other common RF-based positioning technologies include Wi-Fi, Bluetooth, ZigBee, Radio Frequency Identification (RFID) and Ultrawideband (UWB).

Wireless Local Area Network (WLAN), also referred as Wi-Fi, transmits and receives data using electromagnetic waves, providing wireless connectivity within a coverage area [16]. For Wi-Fi positioning system, there are usually three ways to determine a user’s location: 1) the propagation model of a known antenna, 2) multiliteration method and 3) fingerprinting. For most commercially available Wi-Fi positioning systems employ some form of fingerprinting with an accuracy of tens of meters [16]. Bluetooth is a wireless communication technology that uses digitally embedded information on radio frequency signals. Bluetooth technology has been considered for indoor position systems as a competitor to Wi-Fi, in particular since the widespread adoption of Bluetooth Low Energy (BLE), due to its availability (it is supported by most modern smartphones), low cost, and very low power consumption, which allows fixed emitters to run on batteries for several months or even years [17]. Apple proposed iBeacons to apply BLE for localization purposes. The accuracy is affected by signal attenuation and reflection due to obstacles, and range from sub-meter level to several meters. ZigBee is a wireless communication standard developed by the ZigBee Alliance [18]. Most of ZigBee positioning systems use RSSI values to estimate the position, just like Wi-Fi and Bluetooth. RFID is a technology that uses radio waves to make a specialized circuit produce a response containing a unique identifier. A famous application of RFID is NFC (Near-Field Communication), which has been applied on most modern smart phones for mobile payment purposes [16]. LANDMARC (Location Identification based on Dynamic Active RFID Calibration) is a pioneering RFID system, the authors claimed that the accuracy can reach 1 meter [19]. UWB is based on the transmission of electromagnetic wave forms formed by a sequence of very short pulses using a very large bandwidth [19]. The advantage of using UWB for localization is the high precision of time-of-flight measurement and the multipath immunity. TOA and TDOA can be used in UWB positioning systems. The accuracy can reach centimeter level. It remains an open question if and which of these sensors and technologies will ultimately be included in autonomous vehicles and if and how each might be integrated with such a system.

1.1.2 Vehicle motion sensors

For land vehicle navigation, vehicle motion sensors are very useful to provide information about a vehicle’s state that can be used to estimate position, velocity and attitude. The commonly used vehicle motion sensors include: steering angle encoder, wheel odometer, wheel velocity encoder, electronic compass, accelerometer and gyroscope [3].

The steering encoders measure the angle of the steering wheel. By combining with the front wheel speed, the steering angle can be used to calculate the heading rate of the vehicle [3]. Wheel odometer can measure the number of full and fractional rotations of the wheels to output the traveled distance of the vehicle. In the mean time, the wheel velocity encoder can observe the rotation rate of the wheels to provide measurements of vehicle's velocity. If two separate encoders are mounted on the left and right wheels of either the front or rear wheel pair, the heading change of the vehicle can be estimated by calculating the difference between the wheel speeds [20]. Wheel encoder information is often available through the sensors of an Antilock Breaking System (ABS), which must compare each wheel's speed to detect if and when wheels slip or lock, and can be accessed through CANBUS port from modern vehicles. However, the resolution using standard CANBUS messages is usually 1 km/h, which is too low for many applications. Therefore, additional wheel encoders, or custom CANBUS messages, might be necessary to provide reliable wheel speed information. The premise of using wheel rotation to estimate traveled distance, velocity and heading rate is that wheel revolutions can be directly translated into linear displacements relative to the ground [3]. However, in real-world driving scenarios, this assumption might not hold up. The reasons are as follows [21]:

1. Wheel diameter change due to temperature, pressure and tire wear;
2. Wheel slippage and skidding;
3. Uneven road surface;
4. Unequal wheel diameters;
5. Limited resolution and sample rate of the wheel odometer.

An electronic compass is constructed from magnetometers to provide heading information relative to the earth's magnetic north by observing the directions of the earth local magnetic field

[22]. However, since the electronic compass is based on sensing the magnetic field, the power lines and metal structures alongside the road which affect the magnetic field will cause unpredictable errors. A method to account for magnetic declination is also required if absolute orientation is sought.

An Inertial Navigation System (INS) is a complete three-dimensional DR navigation system comprised of a set of inertial sensors (accelerometers and gyroscopes, also known as Inertial Measurement Unit (IMU)) and a processor. The IMU usually consists of three orthogonal accelerometers and three gyroscopes to output 6D poses [10]. The accelerometer measures the acceleration caused by specific force (all forces except for gravity). The gyroscope measures the angular rotation rate of the object relative to the inertial frame of reference. By mounting an IMU on vehicle, the acceleration and angular rate measurements can be mapped into estimates of vehicle’s velocity, position and attitude [3]. IMU sensors have a wide range of types and applications. High-end IMU sensors are usually used in ships, aircraft, missiles and surveying. Nowadays, with the advancements in microelectromechanical system (MEMS) sensor technology, the low-cost MEMS IMUs have gained more and more attention. Unlike the vehicle motion sensors mentioned above, IMU sensors are fully self-contained and are able to output 6D pose information at a very high frequency. However they suffer from unbounded errors that drift rapidly as a function of time and making low-cost INS alone unsuitable as a DR technology, unless the errors can be properly modelled and corrected using other data sources. Generally speaking, the IMU sensor errors include: bias, scale factor, nonlinearity, scale factor sign asymmetry, dead zone, quantization error and cross coupling error [23]. As a result, IMUs must be calibrated and some errors states must be estimated with the navigation solution.

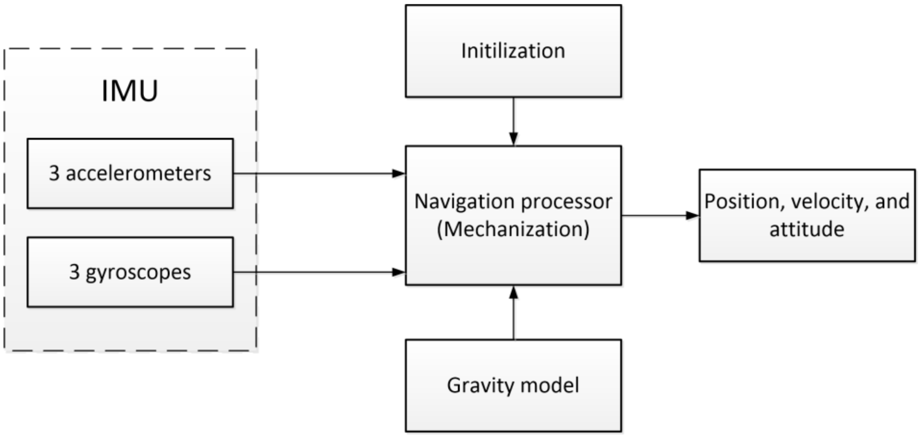

Initilization
IMU
3 accelerometers
3 gyroscopes
Navigation processor (Mechanization)
Position, velocity, and attitude
Gravity model

Figure 1-5: Inertial Navigation System working flow

1.1.3 Road maps

To achieve high-level autonomy of land vehicle navigation, an accurate digital map is necessary. Speaking from navigation standpoint, a digital map can be used to impose constraints on the navigation solution of a land vehicle navigation system. This technique is also referred as map matching [24]. The digital maps are built up as databases of topological (connectivity properties of features) and metrical (coordinates) information, together with attributes such as road class, street name, driving speed limit, and turn restriction. In addition, the road network is generally represented by a planar model on digital maps, where the street system is represented by a set of arcs [3]. Generally speaking, there are three steps of map matching process: 1. select a set of candidate arcs or segments, 2. calculate the likelihood of the selected arc or segments using the topological and geometric information as well as the coordinate relation between the vehicle trajectory and the candidate paths in the digital map, 3. determine the most likely road segments [3].

1.1.4 Visual sensors

Visual sensors, due to the rich information included in images, are believed to potentially cover all relevant information needed for driving [25]. With the fast development of machine vision technology, some companies, for example Tesla, hold the strong belief that LiDAR is too expensive and bulky for daily driving, and cameras represent the future of autonomous vehicle perception technology [26]. To be more specific, visual sensors are able to provide information about: lane geometry, traffic signs and lights, objects in view, drivable road segments, vehicle poses and etc. With the fast advancements of artificial intelligence, visual sensors are also being deployed to understand diverse driving scenes. Compared with the range or velocity measurements from LiDAR, Radar and ultrasonic sensors, the measurements of visual sensors are the brightness intensity of images which requires extra computationally and algorithmically demanding processing procedures to extract information. In this section, some of the aspects of the applications of visual sensors in land vehicle navigation systems based on single-, dual-, and multi-image methods are briefly reviewed [25].

(1) Single Frame

For all the digital image processing algorithms, the first step will always be the camera intrinsic calibration. By using a well-measured target, such as a checkboard, an Aprilgrid or a Circlegrid [27], the camera's principal points, focal distance, radial and tangential distortion coefficients can be estimated. This relates the image frame (2-D) with the world frame (3-D). For a single image in land vehicle navigation, the most common applications are traffic sign recognition and vanishing potin estimation [25]. The classic traffic sign detection is based on brightness and color gradients [28] [29] to classify circular or triangular edges. In some more advanced works, the neural networks are utilized to handle some real-world driving challenges, such as perspective distortion, lighting changes, partial occlusions and shadows [30], thus making the traffic sign detection process more robust. In addition, the intersection of straight lines in a single image can also be applied to estimate the vanishing point. When a camera is properly calibrated, each vanishing point defines a direction vector originating from the focal point of the camera. In [31], the authors utilized this effect to constrain the visual odometry attitude error drift by fusing with the vanishing directions.

(2) Dual Frame

Observation of a scene either from different viewpoints (stereo vision) or from a moving camera (optical flow/VO) at different times yields images bearing significantly more information than single ones [25]. Both stereo vision and optical flow share the same objective: finding the matches between two overlapping images. According to how many pairs of corresponding pixel coordinates are being matched, the general image matching techniques can be divided into dense matching and sparse matching. For dense matching methods, the search for corresponding points is done at nearly all the pixels. However, for sparse matching methods, the search for corresponding points is done only at salient points (feature points) of the images [25].

The sparse matching methods, similar to those used in photogrammetry, were developed first. The key of sparse image matching methods is the feature point detection. The features of an image contain the 2-D coordinates of the keypoints on the image and a descriptor that describes the keypoint’s shape, color, orientation, texture and etc. A keypoint should be different than its surrounding pixels, which means the image pattern around a keypoint distinguishes from its neighborhood. Good features should have the following properties [32]:

- Repeatability: a good feature should be able to be detected in both images taken from different time/view.
- Distinctiveness: the detected feature should be distinctive from its neighborhood.
- Quantity: the number of detected features should be sufficient for two overlapping image to match.

- Accuracy: a good detected feature should be localized accurately.
- Efficiency: for real-time applications, the feature extraction process should be conducted efficiently.
- Robustness: a good feature detection algorithm should be robust against image noise, discretization effects, compression artifacts, blur, etc.

In order to find the best match for given overlapping images, the common approach is to minimize the Hamming distance for binary descriptors [25]. Generally speaking, feature detection methods can be divided into three categories: edge detection, corner point detection, blob detection [33]. For localization and mapping purposes, corner point detection and blob detection are more important as their position in the image can be measured accurately [34]. A corner point is defined as a point at the intersection of two or more edges. A blob is an image pattern that distinguish from its immediate neighborhood in terms of intensity, color, and texture [34]. Corner detection methods include: (1) gradient based: Harris detector [35], KLT [36], Shi-Tomasi detector [37]; (2) template based: FAST [38], FAST-ER [39]; (3) contour based: DoG-curve [40], ACJ [41]. Blob detection methods include: (1) PDE based: SIFT [42], SURF [43], Rank-SIFT [44], KAZE [45], WADE [46]; (2) template based: ORB [47], BRISK [48], FREAK [49]; (3) segmentation based: MSER [50], FLOG [51, p.], BPLR [52]. To sum up, corner detectors are fast but less distinctive while blob detectors are more distinctive but take longer to compute. For VO/SLAM applications, ORB feature is usually deployed for some real-time scenarios [53] while SIFT and SURF are better with post processing for higher accuracy.

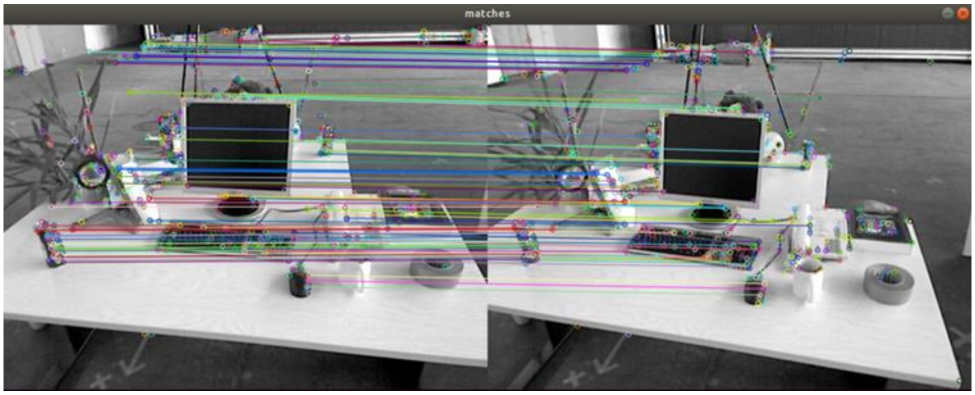


Figure 1-6: An example of feature matching

A major advantage of using stereo vision over monocular vision is that the fixed baseline between the two cameras helps solving ambiguities [25]. Furthermore, two cameras means a larger FoV which makes some data-intensive dense matching methods feasible. According to the amount of pixels during the optimization process, the stereo matching methods can be divided into local methods and global methods. Local methods only optimize the dissimilarity of small image patches, thus susceptible to ambiguities [54]. Correspondingly, global methods optimize the energy term over the entire images [55], [56].

Apart from stereo vision, two images taken by the same camera at different time and location can also have overlapped patterns. Optical flow represents the velocity and direction of image motion. The motion of pixels can be caused either by the motion of camera or the motion of object, this leads to the two different fields of applications: environment change identification and camera motion estimation. In [57], the authors exploited optical flow for human-body motion analysis. In [58], the authors deployed optical flow for measuring fluid depth and velocity. By exploiting epipolar geometry, the camera's ego-motion can also be estimated. More detailed review will be given in Chapter 2.

(3) Multi-Frame

Multi-frame methods evaluate a quadruple of images from two cameras at two points in time. The application of which is scene flow estimation, a motion field representing the 3-D velocities of reconstructed 3-D points [25]. In [59], the authors proposed the concept of 6-D vision, which additionally estimates and compensates ego-motion to offer results with respect to a world coordinate frame rather than the cameras', which is very useful for moving object detection and tracking. Sparse scene flow estimation can perform a more robust consistency check than described above by requiring a closed loop of pair-wise matches from current-left over current-right, previous-right, previous-left back to current-left image [60]. Dense methods vary in the extent of coupling of the four images: Independently estimated initial disparities and flow vectors can be merged on the level of rigid moving objects [61].

1.1.5 Vehicle motion models

Due to some intrinsic characteristics of land vehicles, certain vehicle models can be applied to constrain the navigation solution. Most common used methods include: Non-holonomic Constraints (NHC), Zero-velocity Updates (ZUPT), Coordinate Updates (CUPT) and height constraint.

For the ideal ground vehicle navigation, the vehicle moves in a planar road condition with no wheel slippage and movement in the direction perpendicular to the road surface. Thus, in the vehicle body frame, the upward and sideway velocities should be zero. The most used vehicle models in land vehicle navigation is the Non-Holonomic Constraints (NHC), which constrain the lateral and vertical velocities to zero [62]. For ground vehicle applications, NHC is usually combined with the odometer output to form an auxiliary velocity update to constrain the navigation solution [63] [3]. In [64], the authors used wheel odometer velocity and NHC as observations to integrate with MEMS IMU. The results showed significant improvements compared with pure INS integration results during GNSS outages. [65] developed an INS/Wheel Odometer/NHC DR algorithm which incorporates the calibration process of the IMU body frame and the vehicle frame. The author mathematically proved that the navigation accuracy is affected by the calibration accuracy, and the results showed that by incorporating the calibration process, the accumulated errors drops significantly. In [66], the authors exploited NHC to form motion hypotheses instead of the traditional RANSAC algorithm to improve the outlier reject in VO. The results showed that this proposed outlier rejection scheme can yield more inliers compared with the traditional 5-point RANSAC algorithm.

Apart from NHC, other constraints that take advantage of the vehicle dynamics include: constant height constraint [67] [68] [69], constant LLH position [67] and constant slope [67]. Among which, the constant height constraint is the most commonly used. Normally, when a land vehicle travels in an urban environment, the vehicle height may be assumed to be constant for short time intervals [67]. For pedestrian and land vehicle navigation in urban environments, a constant height or the relative height by using differential barometry with respect to the initial height can be maintained for the navigation solution [68] [65].

Another type of constraints is utilizing the fact that a vehicle can be stationary (ZUPT) and coordinate information can be available from other sources (CUPT). ZUPT is a commonly used technique to constrain the INS drifting errors for both vehicular and pedestrian navigation systems. The idea of applying ZUPT in INS navigation systems can date back to 1960s [70]. The key of applying ZUPT properly is to detect the stopping time interval [71]. The commonly used ZUPT methods include: curve fitting method, maximum likelihood estimation method and Kalman filtering method [72]. Nowadays, some more sophisticated and robust ZUPT methods are developed. In [73], the authors proposed an adaptive threshold of the ZUPT detector in pedestrian inertial navigation systems, which enables the detector to adjust to gait patterns with different speeds. In [74], the authors proposed an machine learning model for ZUPT-aided foot-mounted inertial navigation system. The results showed that by applying the machine learning algorithm, the positioning accuracy can achieve 55 cm over a 1.8 km indoor/outdoor path. In [75], the authors discussed the impact that ZUPT brought to the GPS/INS systems, The results showed that the implementation of ZUPTs increased the quality and reliability of the GPS/INS positioning, by lowering the rate of error growth during the GPS loss of lock. Similar to ZUPT, CUPT is also a very useful method for long-term navigation systems. In some applications the coordinate update can be provided continuously or at some CUPT stations depending on the availability of GPS coordinate update measurements. At each CUPT station, the coordinate update measurements provided by the aiding source (GPS) is compared to the INS position output to constrain the solution [76].

1.2 Challenges of Navigating in Winter and Urban Environments

For driving at northern latitudes, such as Canada, it is difficult to stay safe on the road. According to [77], almost 30% of car accidents in Canada happen on snowy or icy roads. Five percent of those accidents happen during snowfall and more than 50,000 accidents that occur each year are due to precipitation. Canadian winter weather conditions play a huge factor in the number of accidents each year. In 2010, over 1,400 accidents cited weather conditions among factors. With the snowiest months between November and April in Canada, the average snowfall is around 6 cm. Extreme snow depths can reach 25 cm. Even though Canadians are not strangers to driving in the snow, the winter weather continues to play a chief factor in the amount of collisions that happen each and every year. During 2010, over 30% of accidents' major contributing factor was the environmental condition and almost five percent of all accidents resulting in death cited the weather. In more than 26% of 2010 accidents, packed snow or ice was present and during 1,500 accidents, heavy snowfall was occurring. Also, according to Waymo [78], there was 1.35 million deaths worldwide due to land vehicle crashes in 2016, and 2.4 million injuries in 2015 due to vehicle crashes. How to increase the safety level of driving especially in urban areas is of vital importance for public safety.

As has been introduced in the previous section, due to the different characteristics of land vehicle sensors, driving in winter road conditions and complex urban environments is very challenging if only one sensor is deployed. Each sensor has its own pros and cons with different characteristics. Thus, a sensor fusion framework is vital for land vehicle navigation systems in such challenging environments. In addition, due to the limitations of GNSS, the development of an alternative self-contained navigation system in winter urban environment is crucial for various application scenarios.

Table 1-1: Selected Sensors' Performances in Different Challenging Environments

	icy road	snow/fog/mist	urban canyon	dynamic environment
INS	✓	✓	✓	✓
Camera	✓	×	✓	×
Wheel Odometer	×	✓	✓	✓
Digital Maps	✓	✓	✓	✓
GNSS	✓	✓	×	✓

To sum up:

1. For land vehicle navigation systems, GNSS is the major data source for absolute positioning. However, the limited satellite visibility and the multipath effect will degrade the GNSS performance in the urban canyon environment. Also, the GNSS signal is easy

to spoof which makes it not reliable for some safety-critical applications (such as self-driving car).

2. The snowy and/or icy road condition magnify slippage of wheels, which reduces the accuracy of wheel odometer/encoder.
3. The performance of visual sensor is challenged by the complexed scenarios in urban environments (illumination, dynamic environments, shadow and etc.). Also, in the snow-covered environment during winter, the visual sensor suffers from the low-texture scenes of environment.

In recent years, Visual-Inertial Odometry (VIO) has gained more and more attention from both academia and industry [79]. This technology is a self-contained, inexpensive and accurate way to bridge the GNSS outages, and has been successfully applied in land/aerial vehicle navigation, AR and etc. [80]. In addition, wheel odometer data is relatively easy to obtain for modern land vehicle systems. Combining VIO and wheel data is an inexpensive and complementary approach to bridge the GNSS gaps in challenging environments. In [81]–[83], the authors already discussed the feasibility of integrating wheel data with VIO. However, in these works, the potential of applying VIO+WO in winter urban environment has not been fully evaluated.

1.3 Motivations and Objectives

As discussed in the previous section, the motivation of this research can be summed up as:

1. Driving in the winter road conditions and the complex urban environments is very challenging. The accuracy of GNSS positioning is reduced in such environments. The development of an alternative, accurate, inexpensive and self-contained land vehicle

navigation systems to bridge the GNSS gaps is a popular research topic from both academia and industry.

2. VIO is an accurate and inexpensive approach for land vehicle navigation system with a lot of potential to explore. Meanwhile, wheel odometers are present on all modern land vehicles. Both camera and wheel odometer are affected by the winter urban environment. However, existing works do not investigate the performance of the integration system of VIO+WO+NHC.

The main objective of this thesis is to design and implement a wheel odometry aided VIO algorithm for land vehicle navigation and test its performance in winter urban environments. The main objective is fulfilled with two sub-objectives: (1) investigating the influence of winter road conditions and urban environments on the camera and wheel odometer, and (2) exploring the feasibility of applying NHC + wheel speed as external constraints on the VIO velocity drift. In this thesis, an VIO framework named Multi-State Constraint Kalman Filter (MSCKF) will be deployed. The algorithm is first validated on the KITTI dataset [1], and then tested on the KAIST complex urban dataset [84] and a winter driving dataset collected in Calgary, Canada.

The main contributions of this thesis are:

1. Investigation and demonstration of integrating wheel odometer, NHC and VIO as a promising alternative to the tradition GNSS positioning.
2. Evaluation of the influence of winter and urban environments on the camera and wheel odometer sensors.
3. Publishing the Calgary Winter Driving Dataset which contains RGB images, an automotive-grade IMU data, wheel odometer data and GNSS/INS ground truth data.

1.4 Thesis Outline

The remainder of the thesis is organized as follows:

- Chapter 2 provides a comprehensive background and literature review on the cutting-edge VO, VIO, motion constraint aided inertial navigation and sensor fusion techniques.
- Chapter 3 presents the system design and the fundamentals of the WO aided MSCKF.
- Chapter 4 focuses on the data collection platform setup and the algorithm verification.
- Chapter 5 shows the results of the KAIST Complex Urban dataset and our Calgary winter driving dataset. The detailed analysis of the results is also presented.
- Chapter 6 provides a summary of the results and recommendations for future work.

1.5 Publication

- Cheng Huang, Yang Jiang, and Kyle O’Keefe, " Wheel Odometry aided Visual-Inertial Odometry for Land Vehicle Navigation in Winter Urban Environments," Proceedings of the 33nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020), September 2020.

Chapter 2 BACKGROUND

This chapter provides the background and literature review on the fundamentals and recent developments of VO, VIO, motion constraint aided inertial navigation and sensor fusion frameworks. The chapter begins with the review of VO and V-SLAM from monocular, stereo, RGB-D and event camera point of view. Then, in Section 2.2, the general sensor fusion framework is reviewed. In Section 2.3, VIO algorithms are reviewed with emphasis on MSCKF.

2.1 Visual Odometry and V-SLAM

The term visual odometry (VO) was coined by Nister in his landmark paper about solving the five-point relative pose problem [85]. Similar to wheel odometry (WO), VO incrementally estimates the vehicle poses through examination of the changes that motion induces on the images of its onboard camera [34]. The reconstruction of camera poses and 3-D scene structure was first studied by the photogrammetry community and then redeveloped by the computer science community as structure from motion (SFM) [86], [87]. However, SFM focuses on 3-D reconstruction of both the scene structure and camera poses from sequential image sets, and the final structure and camera poses are usually offline optimized through bundle adjustment [34]. VO, on the other hand, only focuses on the online estimation of the 3-D motion of the camera. In this case, bundle adjustment can be used (optionally) to optimize the local trajectory. The first known real-time VO implementation is presented by Moraveck's PhD thesis [88]. In Maraveck's work, he proposed the first image motion-estimation pipeline as well as the earliest corner detector (Moraveck corner detector). He tested his work on a planetary rover equipped with a slide stereo camera: that is a single camera sliding on a rail. The planetary rover operated in a stop-and-go fashion. At each stop, the camera slid horizontally taking nine pictures and matched these with the pictures taken at the next stop. The corner points were estimated by triangulation at two consecutive robot positions, and the motion was computed as the rigid body transformation to align the triangulated points. The system equations were solved via a weighted least squares (WLS) adjustment, and the weights were computed as inversely proportional to the distance from the triangulated points. Although Moraveck’s work only involved with a single camera, it still belongs to the stereo VO for using a triangulation method to determine the 3-D feature position. The main motivation of early VO research is to equip planetary rovers with the ability to measure 6 DoF motion in the presence of wheel slippage in uneven and rough terrain in which scenario that wheel odometer cannot function normally [34]. With modern developments in camera sensing technology, VO-based applications have been widely broadened such as land/aerial/oceanic vehicle navigation [89]–[91], AR/mixed reality applications [92], visual surveillance systems[93] , medicine [94] and etc. Generally speaking, VO is an inexpensive and alternative odometry technique that is more accurate than conventional techniques, with a relative positioning error ranging from 0.1 to 2% of distance travelled [34].

The advantages of VO are manifold:

1. Straightforward, accurate and inexpensive;
2. Able to be used in GPS-denied environment, such as indoor, underwater and outer space environment;
3. Lightweight;
4. Do not emit any detectable energy into the environment, do not suffer from the interferences often encountered when active ultrasonic/laser or RF-based sensors are used;
5. Easy to integrate with other sensors.

Besides all the advantages, there are still some limitations when using imaging sensors especially in outdoor large-scale environments. The main challenges in VO systems are mainly related to computational cost (especially optimization methods), light conditions (direct sunlight, shadows and image blur) and dynamic environment [95]–[97]. In addition, in some low-texture environments, such as white walls or snow-covered environments, the feature-based VO methods will also suffer from insufficient feature points. For standalone monocular VO systems, there is the additional challenge of the scale uncertainty [98]. The absolute scale can be alternatively solved or estimated from other measurements, such as direct measurements (measuring the size of an element in the scene), motion constraints, IMU, air pressure, ranging sensors or a pre-determined camera baseline (stereo camera). Also, due to the fact that the distance of the camera above the road surface is nearly constant for land vehicles, [99] proposed a planar road model to solve the scale ambiguity. However, there are two assumptions to be made: (1) streets are assumed to be approximately planar in the vicinity of the vehicle; (2) the roll and pitch movement of the vehicle assumed to be negligible.

Although this section mainly focuses on the review of VO, the relation between the visual simultaneous localization and mapping (V-SLAM) and VO still has to be mentioned. SLAM is a technique for obtaining the sensor motion and the environment map (3-D structure) in an unknown environment simultaneously. This technique was originally proposed for autonomous robotic control [100], and has gained more and more attentions over the years. V-SLAM means that the camera is the only exteroceptive sensor. The classic basic V-SLAM modules contain three parts: initialization, tracking and mapping [101]. Initialization is an essential step to define the global coordinate system and also provide or estimate the initial absolute coordinate for the map. After the initialization, tracking and mapping are performed simultaneously to continuously estimate the camera motion. The tracking module can be interpreted as VO. In the mapping module, the map is expanded by computing the 3-D structure when the camera observes unknown areas where the mapping has not been performed before. Besides the basic modules, two additional advanced modules are also included in the V-SLAM algorithms for more accurate and robust localization purposes: relocalization and global map optimization. Relocalization is of vital importance for long-run V-SLAM operations when the tracking module fails due to fast camera motion or disturbances. In such scenarios, the camera pose needs to be re-initialized again with respect to the map. Global optimization is a very important module to reduce the accumulative errors. When a starting region is captured again and being recognized successfully, the reference information represents the accumulative error from the beginning to the present can be computed. In the meantime, a loop constraint can be used to reduce the error in the global optimization [102].

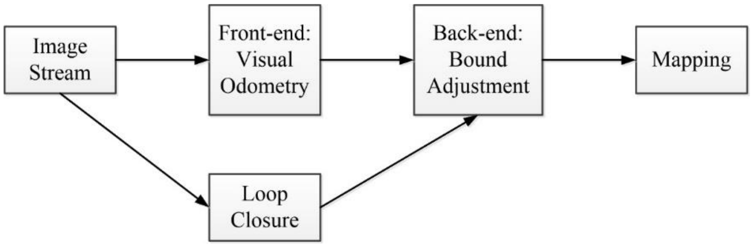


Figure 2-1: Classic V-SLAM Pipeline

For modern development of VO systems (Figure 2-1), more types of camera are being used, such as monocular camera, stereo camera, omnidirectional camera, RGB-D camera, event camera and etc. Due to the strong connection between VO and V-SLAM, the following VO introductions will also cover some the advancements from V-SLAM research.

(1) Stereo VO

Due to the scale factor uncertainty in monocular VO, most classic early-stage work focused on stereo VO. There are three main approaches to estimate the motion by leveraging stereo pair of images. The first and most intuitive one is to triangulate every stereo pair to obtain the 3D points, and the relative motion is solved as a 3-D to 3-D point registration (alignment) problem. In 2003, [85] proposed the first real-time long-run implementation with a robust outlier rejection scheme. First, contrary to all previous works, they did not track features among frames but detected features (Harris corners) independently in all frames and only allowed matches between features. This has the benefit of avoiding feature drift during cross-correlation-based tracking. Second, they did not compute the relative motion as a 3-D to 3-D registration problem but as a 3-D to 2-D camera pose estimation problem. Finally, they incorporated RANdom SAmple Consensus (RANSAC) outlier rejection into the motion estimation step. Finally, a motion estimation scheme was introduced by [103]. Instead of using 3-D to 3-D point registration or 3-D to 2-D camera pose estimation techniques, they relied on the quadrifocal tensor, which allows motion to be computed from 2-D to 2-D image matches without having to triangulate 3-D points in any of the stereo pairs. The benefit of using directly raw 2-D points in lieu of triangulated 3-D points lies in a more accurate motion computation [34].

(2) Monocular VO

The interest in monocular methods is due to the observation that stereo VO can degenerate to the monocular case when the distance to the scene is much larger than the stereo baseline (i.e., the distance between the two cameras). A monocular camera, by its nature, only measures the bearing information. Thus, the motion can only be recovered up to a scale factor. The main difference between stereo VO and monocular VO is that monocular VO has to compute both relative motion and 3-D structure from 2-D bearing data. Since the absolute scale is unknown, the distance between the first two camera poses is usually set to one. As new images arrive, the relative scale and camera pose with respect to the first two frames are determined using either the knowledge of 3-D structure or the trifocal tensor [34].

Related works on monocular VO can be divided into three categories: feature-based methods, direct methods, hybrid methods and deep learning methods.

Feature-based methods are based on salient and repeatable features that are tracked over frames. There are two types of feature-based methods: filtering-based and BA-based. In this type, Nister et al. proposed the innovative five-point minimal solver to calculate the motion hypotheses in RANSAC [85]. His ground-breaking paper made five-point RANSAC very popular [104], [105]. Lhuillier and Mouragnon et al. presented an approach based on local windowed-bundle adjustment to recover both the motion and the 3-D map [105]. In Tardif et al. work [106], they decoupled the rotation and translation estimation. The rotation was estimated by using points at infinity and the translation from the recovered 3-D map. Erroneous correspondences were removed with five-point RANSAC. MonoSLAM is the first real-time monocular V-SLAM system, it also represents the filtering-based VO methods [107]. In MonoSLAM, the camera pose and 3-D environment structure are simultaneously estimated via EKF. Both 6 DoF camera motion and 3-D feature positions are included in the state vector of EKF. Depending on the camera motion, new feature points are added to the state vector. The initialization is done by observing a known object which global coordinate system is defined. However, there are several limitations of MonoSLAM. First, the computational cost increases in proportion to the size of feature points. For land vehicle applications, the large-scale environment will result in a large size of state vector, which means it is difficult to achieve real-time computation. Second, the sparse features in the map are easy to lose track. Parallel Tracking and Mapping (PTAM), on the other hand, is the first method which deploys BA rather than filtering in real-time V-SLAM algorithms [108]. PTAM split the feature tracking and mapping into two parallel threads on CPU, which improves the computational efficiency. In addition, a keyframe mechanism is proposed in PTAM, that is, instead of carefully processing each image, several key images are stringed together to optimize its trajectory and map. The disadvantage of this method is that the scene is small and tracking is easy to lose. In [109], the authors compared the differences between EKF-based estimation in MonoSLAM and BA-based estimation in PTAM. The authors conducted a series of Monte Carlo experiments to investigate the accuracy and cost of V-SLAM. As a result, they proved that it is important to increase the number of feature points for higher accuracy. And from this point of view, the BA-based method is better at handling large number of feature points than the EKF-based methods. ORB-SLAM is considered to be the most complete feature-based V-SLAM method. ORB-SLAM [110] was first proposed in 2015, it deployed and calculated ORB features, including ORB dictionary for visual odometry and loop detection. ORB feature calculation efficiency is higher than SIFT (Scale-Invariant Feature Transform) or SURF (Speed-Up Robust Features), and it has good rotation and scaling invariance. ORB-SLAM innovatively uses three parallel threads to increase the computational efficiency. The three threads are: real-time feature tracking thread, local bundle adjustment thread and global pose graph optimization thread. The disadvantage of this method: it is very time-consuming to calculate the ORB feature once for each image, and the three-thread structure brings a heavy burden to the CPU. The sparse feature point map can only meet the positioning needs, and cannot provide navigation, obstacle avoidance and other functions. In 2017, the authors proposed ORB-SLAM2, which includes loop detection, re-localization and map reuse [53]. ORB-SLAM2 is the first open source SLAM system that supports monocular, stereo and

RGB-D cameras. In 2020, ORB-SLAM3 was released [111]. Compared with ORB-SLAM2, ORB-SLAM3 also includes visual-inertial odometry system.

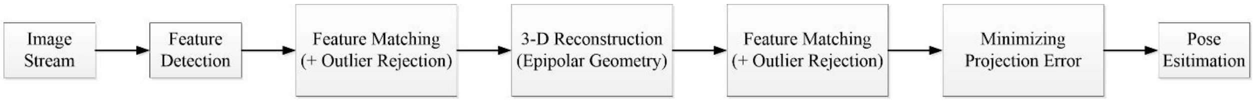


Image Stream
Feature Detection
Feature Matching (+ Outlier Rejection)
3-D Reconstruction (Epipolar Geometry)
Feature Matching (+ Outlier Rejection)
Minimizing Projection Error
Pose Esitimation

Figure 2-2: Feature-based VO pipeline

Compared with feature-based methods (Figure 2-2), direct methods directly use pixel intensity of input image without computing feature descriptors and detectors. Thus, the direct methods are also called feature-less methods. In general, photometric consistency is used as an error measurement in direct methods whereas geometric consistency such as positions of feature points in an image is used in feature-based methods. Inspired by PTAM, DTAM is the first direct dense method V-SLAM system [112]. It calculates key frames to build a dense depth map by minimizing the global spatial norm energy function, while the camera pose is calculated by direct image matching. This method is robust to low-texture environment and motion blur. The disadvantage of this method is that the amount of calculation is very large, and GPU parallel computing is required. DTAM assumes a constant luminosity and is not robust enough for global illumination processing. LSD-SLAM is another representative direct V-SLAM methods. The idea of LSD-SLAM came from semi-dense VO [113], which the reconstruction targets are limited to areas which have intensity gradient compared to DTAM which reconstructs full areas. This means that it ignores textureless areas because it is difficult to estimate accurate depth information from images [101]. LSD-SLAM [114] built a large-scale direct monocular SLAM framework, and proposed an image matching algorithm to directly estimate the similarity transformation between key frames and scale perception, and realize the reconstruction of semi-dense scenes on the CPU. The disadvantage of this method is that it is sensitive to the camera's internal parameters and exposure, and it is easy to lose when the camera moves quickly, and it still needs feature points for loop detection. SVO [115] (Semi-direct Visual Odometry) is a semi-direct visual odometer, which is a mixture of feature-based methods and direct methods: some corner points are tracked, and then like the direct method, based on the information around the key points to estimate camera movement and position. Since there is no need to calculate a large number of descriptors, the speed is extremely fast, reaching 300 frames per second on consumer laptops and 55 frames per second on drones. The shortcomings of this method are: abandoning the back-end optimization and loop detection, there is a cumulative error in the pose estimation, and it is difficult to relocate after loss. Direct Sparse Odometry (DSO) [116] is a visual odometry method based on highly accurate sparse direct structure and motion formula. Without considering the geometric prior information, the photometric error can be directly optimized. And considering the photometric calibration model, its optimization range is not all frames, but a sliding window formed by the latest frame and its previous frames, and this window has 7 key frames. In addition to perfecting the error model of direct method pose estimation, DSO also adds affine brightness transformation, photometric calibration, and depth optimization. There is no loopback detection in this method.

Due to the limitations of the traditional feature-based and direct methods in environment adaptability, there has been recent research focusing on applying deep learning technology into the VO/V-SLAM field. CNN-SLAM [117] deployed Convolutional Neural Networks (CNNs) to predict depth maps for the goal of accurate and dense monocular reconstruction. The authors demonstrated in the paper that by fusing the predicted depth information with the depth measurement obtained from direct monocular SLAM, the absolute scale can be estimated more accurately. The PoseNet [118] trained a convolutional neural network to regress the 6-DoF camera pose from a single RGB image in an end-to-end manner with no need of additional engineering or graph optimisation. The authors demonstrated that the PoseNet localizes from high level features and is robust to difficult lighting, motion blur and different camera intrinsics where point based SIFT registration fails. Furthermore we show how the pose feature that is produced generalizes to other scenes allowing us to regress pose with only a few dozen training examples. UnDeepVO [119] is able to estimate the 6-DoF pose of a monocular camera and the depth of its view by using deep neural networks. The features of UnDeepVo are twofold: one is the unsupervised deep learning scheme, and the other is the absolute scale recovery.

(3) RGB-D VO

In the past years, novel camera systems like the Microsoft Kinect or the Asus Xtion sensor that provide both color and dense depth images became readily available. The applications of such RGB-D cameras in VO/V-SLAM algorithms are gaining more and more attentions. By using RGB-D cameras, 3D structure of the environment with its texture information can be obtained directly. In addition, in contrast to monocular V-SLAM algorithms, the scale of the coordinate system is known because 3D structure can be acquired in the metric space. KinectFusion was proposed in 2011 [120]. The camera motion is estimated by the ICP algorithm using an estimated 3-D structure and the input depth map. KinectFusion is implemented on GPU to achieve real-time processing. SLAM++ uses RGB-D camera to register several 3-D objects in database in advance [121]. In this system, the real-time 3D object recognition and tracking provides 6DoF camera-object constraints are conducted which feed into an explicit graph of objects, continually refined by efficient pose-graph optimisation.

(4) Event Camera VO

Event cameras are bio-inspired sensors that differ from conventional frame cameras: Instead of capturing images at a fixed rate, they asynchronously measure per-pixel brightness changes, and output a stream of events that encode the time, location and sign of the brightness changes. Event cameras offer attractive properties compared to traditional cameras: high temporal resolution, very high dynamic range, low power consumption, and high pixel bandwidth, resulting in reduced motion blur. Hence, event cameras have a large potential for robotics and computer vision in challenging scenarios for traditional cameras, such as low-latency, high speed, and high dynamic range. However, novel methods are required to process the unconventional output of these sensors in order to unlock their potential [122]. In [123], the authors proposed a novel and effective solution to 3D reconstruction using a pair of temporally-synchronized event cameras in stereo configuration. It outperforms state-of-the-art stereo methods using the same spatio-temporal image representation of the event stream. However, even with the great potential of event camera, this field still needs a lot of explorations.

2.2 Sensor Fusion Frameworks

Sensor fusion can be distinguished into two categories: centralized and decentralized (or distributed) approaches [124]. The centralized sensor fusion framework usually offers a higher degree of accuracy as all information is available during the state estimation. In addition, centralized sensor fusion framework is under the consistent model assumptions that contains all relevant modeling knowledge. Furthermore, it does neither double count information nor is uncertain whether all available information has been processed [124]. The decentralized sensor fusion framework distribute the computational load over multiple hardware units which results in a higher computational efficiency. [125] proposed distributed sensor fusion with an EKF for navigation proposes. Particle filters have been used for sensor fusion applied to distributed surveillance [126]. In this thesis, centralized sensor fusion framework will be employed because it requires fewer assumptions about the sensors to be made.

Sensor fusion techniques are a trend, but it is also a compromise. Due to the fact that a single sensor cannot adapt to all scenarios, only by combining multiple sensors with complementary characteristics can we achieve the goal of an ideal localization and navigation system.

2.3 Visual-Inertial Odometry

Visual-Inertial Odometry (VIO) can be interpreted as the integration of VO and inertial measurements, and has been applied broadly in mobile robotics, AR, self-driving cars, UAV, underwater vehicles and etc. [80]. The reasons why VO and inertial sensors are complementary are threefold:

1. VO is sensitive to motion blur induced by rapid motions. Due to the high frequency output, IMU result can be still be relied upon in high-speed scenarios;
2. IMU measurements can drift away even the agent is still, VO can constrain the drifts;
3. VO is not robust in low-texture environments (snow, mist, dark …) and dynamic environments (with moving objects). IMU measurements are not affected by those factors.

There is a considerable amount of work covering a wide range of estimation techniques to do this integration. From the integration point of view, these techniques can be characterized as either loosely coupled or tightly coupled. The loosely coupled system simply fuses the pose estimation results from both IMU and camera, which means that VO here can be treated as a black box. The detected feature points will not be included in the state vector. Stephen Weiss proposed and implemented two outstanding open-source loose integration frameworks: SSF and MSF [125].

Due to the fact that the IMU and the VO are two independent modules in the loosely coupled VIO systems, the update frequencies of the two modules are different and VO is simply used to correct the IMU drift errors. The advantages of loosely coupled VIO are its simplicity and lower computational cost. However, the disadvantages are in that the system cannot correct the drift intrinsic to VO. On the other hand, the tightly coupled method fuses the state of the camera and the IMU together into set of a motion and observation equations, and then perform the state estimation. In this case, the detected feature points will be included into the state vector. Since the IMU accumulated errors between image frames is relatively small, IMU data can be used to predict the inter-frame motion. The advantage of tightly coupled VIO systems is that the IMU scale metric information can be used to aid in the estimation of the VO scale, thus having a higher positioning accuracy. The disadvantages are high computational cost and the addition of 3D feature points in the state that need special handling. Due to the correlations with the SLAM research, the tightly coupled VIO systems have evolved into many different and successful algorithms which can be generally divided into three types: filtering-based, optimization-based and deep-learning approaches [80].

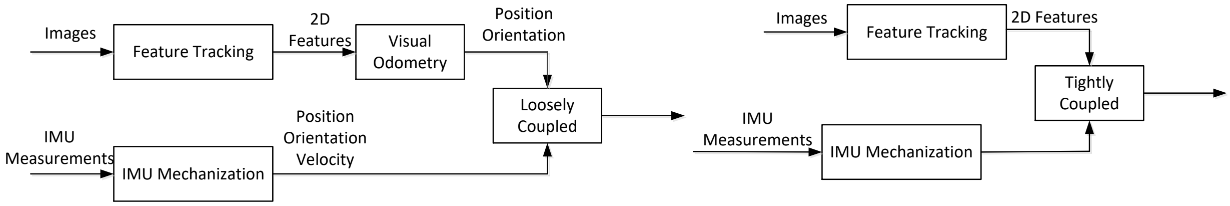


Figure 2-3: Comparison of traditional loosely coupled VIO and tightly coupled VIO

(1) Filtering-based VIO

The filtering-based algorithms only infers and updates the most recent state, thus can perform efficient estimation. The early and traditional filtering-based VIO algorithms include: EKF-based

VIO [127]–[131], Unscented Kalman Filter (UKF) VIO [132], and batch or incremental smoother VIO [133]. Among these, MSCKF [128] stands out due to its simplicity and high efficiency. MSCKF was applied to the application of spacecraft descent and landing [130] and fast UAV autonomous flight [134] This approach uses the quaternion-based inertial dynamics for state propagation tightly coupled with an efficient EKF update. As is shown in Figure 2-4, rather than adding features detected and tracked over the camera images to the state vector, their visual bearing measurements are projected onto the null space of the feature Jacobian matrix, thereby retaining motion constraints that only relate to the stochastically cloned camera poses in the state vector [135]. While reducing the computational cost by removing the need to co-estimate potentially hundreds and thousands of point features, this operation prevents the relinearization of the features' nonlinear measurements at later times, yielding approximations deteriorating its performance.

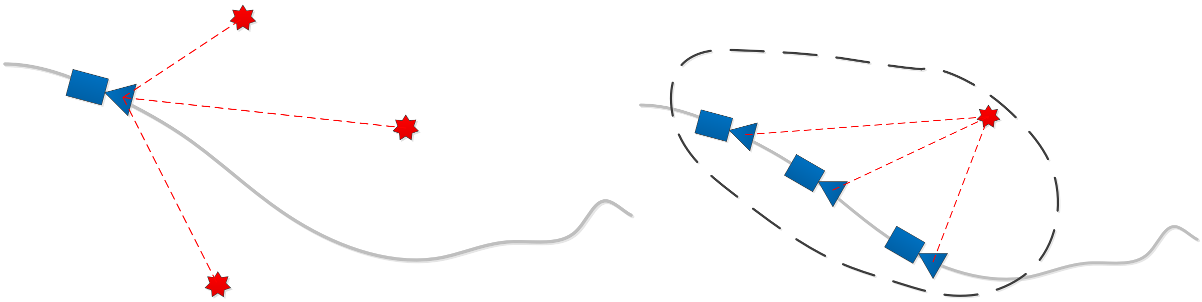


Figure 2-4: EKF-based VIO vs. MSCKF-based VIO

To sum up, the traditional EKF-based VIO includes the feature position estimates in the state vector, and only the most recent pose is estimated, old states are deserted. The algorithm complexity is cubic in number of features. The MSCKF incorporates the sliding window scheme and maintains a sliding window of camera poses in the state vector. Each feature position is used as a constraint of a series of poses. The filter updates only when feature is out of view or reach the maximum of tracking frame number. The algorithm complexity is linear in number of features, thus having a better computational efficiency. In [136], the authors proved that the standard method of computing Jacobian matrixes in filters inevitably resulted in inconsistencies and a loss of accuracy through simulation tests, which showed that the yaw errors of the MSCKF lay outside the $\pm 3\sigma$ bounds indicating inconsistencies. Thus they proposed modifications to the MSCKF to ensure the correct observability properties without incurring additional computational costs. [137] compared MSCKF and the sliding window filter (SWF). Its results showed the SWF to be more accurate and less sensitive to tuning parameters than the MSCKF. However, the MSCKF is computationally cheaper, has good consistency properties, and improves in accuracy as more features are tracked. S-MSCKF [134] can be considered a stereo version of MSCKF. The software takes synchronized stereo images and IMU messages and generates a real-time 6DOF camera pose estimation. It uses the FAST corner to increase the speed and tracked features with KLT optical flow. In addition, circular matching can be used to remove outliers generated during feature tracking and stereo matching.

(2) Optimization-based VIO

Theoretically, filtering-based VIO system suffer from one limitation: non-linear measurements must have a one-time linearization before processing, thus possibly introducing large linearization errors into the estimator and degrading performance. Batch optimization methods, by contrast, solve a nonlinear least-squares (bundle adjustment or BA [138]) problem over a set of measurements, allowing for the reduction of error through relinearization but with high computational cost. [139] introduced a keyframe-based optimization approach (i.e., OKVIS), where a set of non-sequential past camera poses and a series of recent inertial states, connected with inertial measurements, was used in nonlinear optimization for accurate trajectory estimation. OKVIS [139] is called Open Keyframe-based Visual-Inertial SLAM. This solution uses a keyframe-based sliding window (that is, a fixed lag smoother). The cost function is based on the weighted reprojection error of the visual landmark It is combined with the weighted inertial navigation error term and uses Google Ceres Solver for nonlinear optimization. The front end uses multi-scale Harris corner detection to find feature points, and completes the data association between the two frames based on the BRISK descriptor. Older key frames in the sliding window will be deleted and will no longer be estimated. It should be noted that OKVIS is not optimized for monocular VIO. In [139] a solution with a binocular configuration is given, which shows certain superior performance. VINS-Mono [140] is a tightly coupled sliding window estimator based on nonlinear optimization, and the feature points are GFTT. VINS-Mono introduces several new features for this category of estimation framework. First, the author proposes a loosely coupled sensor fusion initialization method, using SFM to estimate the pose and 3D point inverse depth of all frames in the sliding window purely, and finally align with IMU pre-integration to solve the initialization parameters. It performs pre-integration when obtaining new IMU measurement data, and after obtaining IMU constraints, and performs nonlinear optimization together with visual constraints and closed-loop constraints to solve attitude and offset. In addition, VINS-Mono performs loopback optimization based on the 4-DOF pose graph.

(3) Deep Learning VIO

Recently, due to the huge advancements in deep learning research, researchers have already demonstrated that it is possible to train a deep neural network to regress the interframe pose between two images acquired from a moving robot directly from the original image pair [141] effectively replacing the standard geometry of visual odometry. Likewise, it is possible to localize the 6-DoF of a camera using a regression forest [66] and with deep convolutional neural network [118], and to estimate the depth of a scene (in effect, the map) from a single view solely as a function of the input image [142].

Chapter 3 METHODOLOGY

In this chapter, the methodologies employed to develop visual inertial odometry and wheel odometry are provided. First, in Section 3.1 and 3.2, the notations and concepts of descripting rotation and coordinates in this thesis are introduced. Then, the pipeline of feature-based VO based on epipolar geometry and PnP are presented in detail. This chapter finishes by presenting the proposed wheel odometer aided MSCKF algorithm with details of implementation.

3.1 Coordinate System

The relevant coordinate system throughout this thesis are described in this section. All of them are defined as right handed coordinate systems.

3.1.1 Earth-Centered Inertial Frame (i-frame)

An inertial coordinate frame is one that does not accelerate or rotate with respect to the rest of the universe. In navigation systems, a more specific inertial frame, known as Earth-Centered Inertial Frame (ECI), is centered at the Earth's center of mass and oriented with the z-axis parallel to the Earth's spin axis and the x-axis oriented to the vernal equinox. ECI is a sufficiently accurate approximation to an inertial frame for navigation purposes [10]. ECI is of vital importance because inertial sensors measure motion with respect to a generic inertial frame, which greatly simplifies the navigation equations.

3.1.2 Earth-Centered Earth Fixed (ECEF) Frame (e-frame)

The Earth-Centered Earth Fixed (ECEF) Frame, is similar to ECI frame, except that all three axes are fixed with respect to the Earth. The origin of ECEF is at the center of the ellipsoid model of the Earth surface. The z-axis points at the Earth's axis of rotation from the center to the true North (North Pole). The x-axis points from the center to the intersection of the equator with the IERS reference meridian (IRM) or conventional zero meridian (CZM), which defines 0 degree longitude. The y-axis completes the right-handed orthogonal set, pointing from the center to the intersection of the equator with the 90-degree east meridian [10].

3.1.3 Local Level Frame (l-frame)

The Local Level Frame is important for navigation because the user needs to know their position relative to the east and north and up direction. The $x - y$ plane of l-frame is locally horizontal. The origin of the l-frame changes as the user's location changes. In this thesis, the ENU definition of l-frame is adopted. In the ENU frame, the x, y, and z axes are pointing in the direction of east, north and up.

3.1.4 Navigation Frame (n-frame)

Navigation frame can be assigned to be one of the frames defined above depending on the applications. Here we use local level frame as the navigation frame for near-Earth navigation in non-polar areas. Usually, position is expressed in curvilinear coordinates: geodetic latitude, longitude, and geodetic height.

3.1.5 IMU Body Frame (b-frame)

The origin and axes are fixed with respect to the IMU body. Generally, body axes are aligned with the vehicle's lateral, longitudinal, and vertical direction, for better describing the vehicle's orientation. In this thesis, the x-axis points forward, the y-axis points left and the z-axis points up. The origin is the measurement origin of the IMU. IMU sensors measure the motion of the IMU body frame with respect to the inertial frame. The commonly used attitude Euler angles, pitch, roll, and heading, are defined between b-frame and n-frame.

3.1.6 Camera Frame (c-frame)

The origin of the Camera Frame is the perspective center of the camera. The z-axis aligns with optical axis, pointing towards the scene. The x-axis points to the right, and the y-axis points to downward.

3.1.7 Vehicle Motion Frame (m-frame)

The Vehicle Motion Frame is where the wheel odometer and NHC are measured. The origin of the Vehicle Motion Frame is the ground projection of the center point of the vehicle's rear wheel axle. The x-axis is pointing forward, the y-axis is pointing towards the left side of the vehicle and the z-axis is perpendicular pointing up. The Vehicle Motion Frame usually has a near-constant displacement and orientation relationship with the IMU Body Frame, known as the lever-arm and misalignment between the two frames [65].

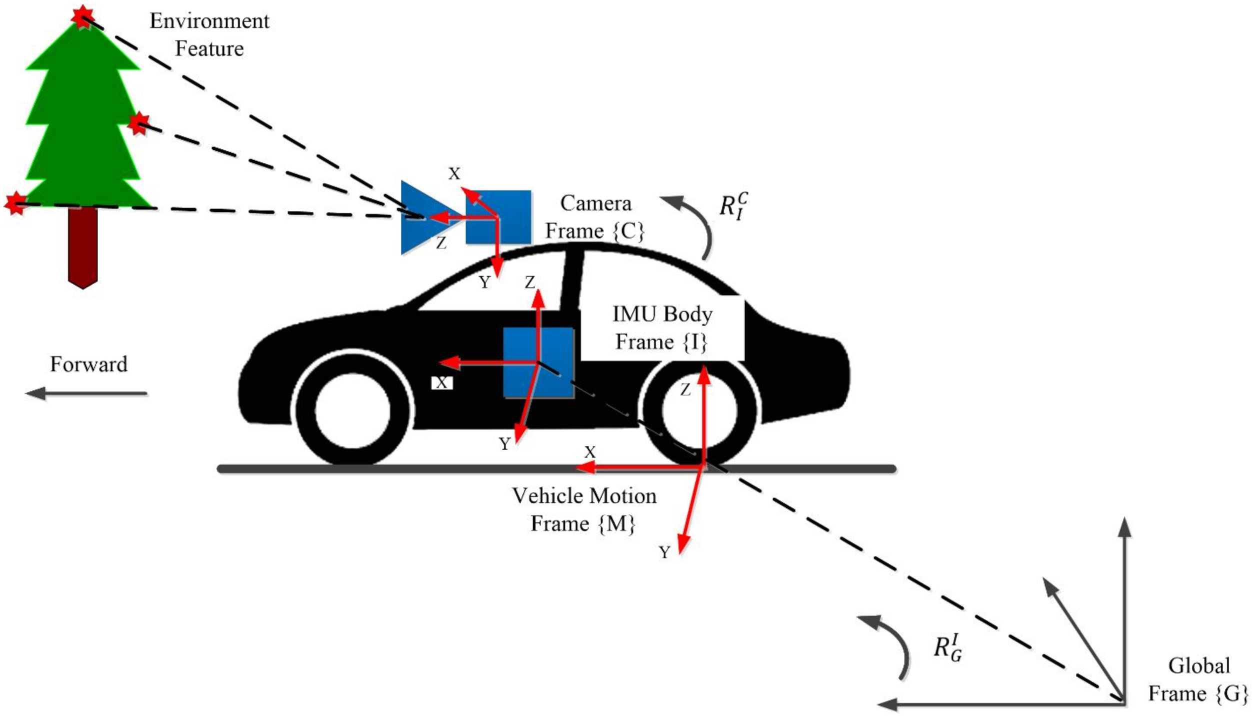


Figure 3-1: Coordinate System Illustration

3.2 Vehicle Motion

As has been reviewed in Chapters 1 & 2, due to the complementary characteristics of wheel odometer, camera and inertial sensors, the integration of WO/VO/INS is a sensible choice for land vehicle navigation in order to bridge GNSS gaps in winter urban environments. For a wheeled ground vehicle, the six DoF (three translations and three rotations about the x-axis, y-axis and z-axis) can be simplified as a rigid body motion when suspension characteristics are not taken into account [143]. Nowadays, some land vehicles are four-wheel steered, however in general, most land vehicles are front-wheel steered. Ackermann steering geometry is used for the vehicular model in this thesis. The Ackermann steering geometry was first designed to minimize tire scrub during cornering, thus the wheels need to roll without tire-relative lateral sliding [143].

This geometry is used in combination with the assumption for the choice of the vehicle coordinate system when NHC is applied. This vehicle model and assumption are only valid when there is no wheel slippage. In addition, the model is only applicable if the effects of the vehicle suspension can be neglected.

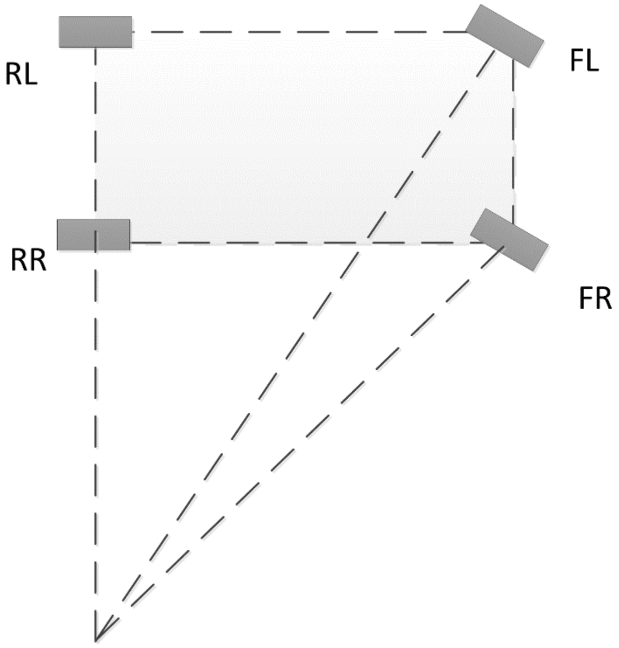


Figure 3-2: Ackermann Steering Geometry

For land vehicle navigation, rigid body motion is utilized to represent the vehicle's pose. For rigid bodies, the distance between any two points remains unchanged during the course of motion of the body. As rigid bodies are viewed as collections of points, it is sufficient to describe the rigid body motion by the rotation and translation of one single point. Considering an arbitrary moving point a, the rigid body transformation can be written in homogeneous form as:

$$\begin{bmatrix} a' \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ O^T & 1 \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix} \triangleq T \begin{bmatrix} a \\ 1 \end{bmatrix} \tag{3.1}$$

where R and t represent the rotation and translation, T is the transformation matrix.

The inverse of the motion is defined as:

$$T^{-1} = \begin{bmatrix} R^T & -R^T t \\ 0 & 1 \end{bmatrix} \tag{3.2}$$

The body frame of the vehicle is defined as the z-axis point up, the x-axis forward, and the y-axis to the left completing a right handed system. The rotations in this thesis are represented as either Euler form or quaternion form. According to the Euler's Theorem, a 3D body can be rotated about three orthogonal axes, as shown in Figure 3-2, these rotations are referred to as yaw, pitch, and roll. In this thesis, the following convention is used:

- A yaw angle is a counter clockwise rotation of α about the z-axis. The rotation matrix is given by:

$$R_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

- A pitch angle is a counter clockwise rotation of β about the y-axis. The rotation matrix is given by:

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{3.4}$$

- A roll angle is a counter clockwise rotation of γ about the x-axis. The rotation matrix is given by:

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{3.5}$$

The yaw, pitch and roll rotations can represent a 3D body in any orientation. A single rotation matrix can be formed by multiplying the three individual rotation matrices:

$$\begin{gathered} R(\alpha,\beta,\gamma) = R_z(\alpha)R_y(\beta)R_x(\gamma) \\ = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix} \end{gathered} \tag{3.6}$$

It should be noted that in this convention, the rigid body performs roll first, then the pitch, and finally the yaw.

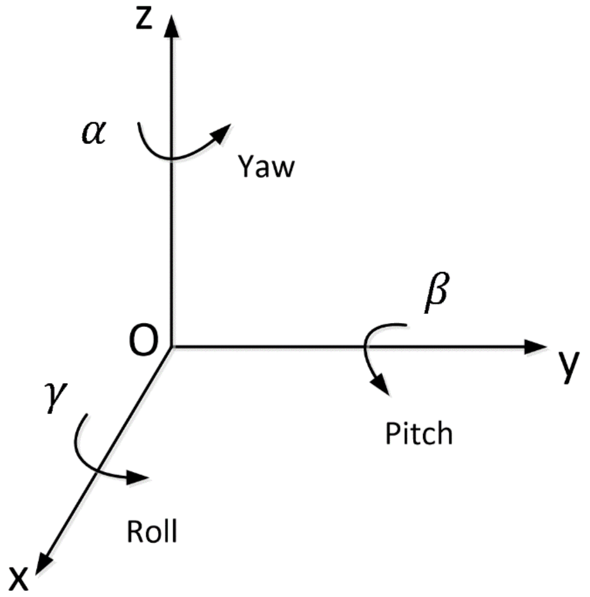


Figure 3-3: Any three-dimensional rotation can be described as a sequence of yaw, pitch, and roll rotations

In order to avoid the gimbal lock problem and to improve the computational efficiency, the derivations of the inertial navigation and VO use quaternion parameters for the parameterization of the rotation matrix.

Quaternions can be represented as hyper complex numbers with three imaginary parts:

$$q = iq_1 + jq_2 + kq_3 + q_4 \tag{3.7}$$

with constraints:

$$i^2 = j^2 = k^2 = ijk = -1 \tag{3.8}$$

The vector representation of the quaternion can be written as:

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \left(\frac{\gamma}{\Theta}\right) \sin \frac{\Theta}{2} \\ \left(\frac{\beta}{\Theta}\right) \sin \frac{\Theta}{2} \\ \left(\frac{\alpha}{\Theta}\right) \sin \frac{\Theta}{2} \\ \cos \frac{\Theta}{2} \end{bmatrix} \tag{3.9}$$

where $\Theta = \sqrt{\alpha^2 + \beta^2 + \gamma^2}$ is the rotation angle, $\frac{\gamma}{\Theta}, \frac{\beta}{\Theta}, \frac{\alpha}{\Theta}$ are the three direction cosines of the rotation axis with respect to the original coordinate system.

The rotation matrix can be recovered from the quaternion by:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1 q_2 - q_3 q_4) & 2(q_1 q_3 + q_2 q_4) \\ 2(q_1 q_2 + q_3 q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2 q_3 - q_1 q_4) \\ 2(q_1 q_3 - q_2 q_4) & 2(q_2 q_3 + q_1 q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \tag{3.11}$$

3.3 Feature-based Visual Odometry

As has been reviewed in Chapter 2, VO is the process of estimating vehicle poses through examination of the changes that motion induces on the images of its onboard camera. Compared to the wheel odometry, VO can output more information. Based on the estimation technique employed, VO can be classified into feature-based methods, direct methods and deep learning methods. The EKF feature-based VO is a classic, efficient and accurate approach, but sensitive to the lighting conditions and dynamic objects. In this section, the pipeline of the EKF VO is presented in detail.

3.3.1 Pinhole Camera Model

The pinhole camera model is the simplest model to describe the imaging process by a camera recognized by a flat image plane and perspective center. The premise of the pinhole camera model is that light travels in form of straight lines in homogenous materials and the optical path is reversible.

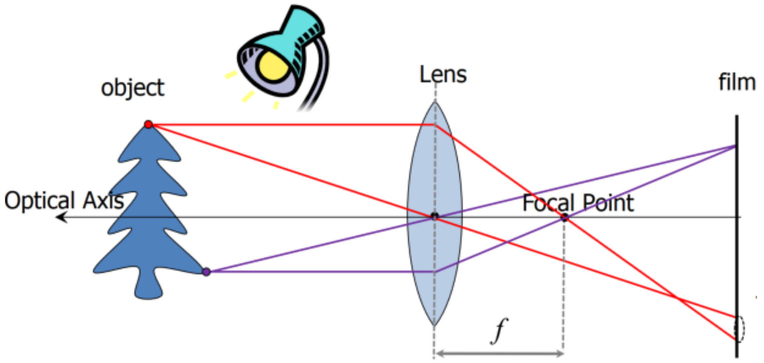


Figure 3-4: Pinhole Camera Model from [34]

A point $A = [X_A \quad Y_A \quad Z_A]$ in the camera frame can be projected onto the image plane using the pinhole camera model.

$$p_a = \begin{bmatrix} x_a \\ y_a \end{bmatrix} = \frac{f}{Z_a} \begin{bmatrix} X_A \\ Y_A \end{bmatrix} \tag{3.12}$$

To describe the parameters corresponding to the project, the Intrinsic Orientation Parameters (IOP) and lens distortions need to be considered. The image measurements are measured in the image frame. The IOP are the parameters that define the image coordinate system, which contain the offsets of principal point (x_p, y_p) and the focal length f.

The image measurement of point A can be described in the image frame as:

$$p_a = \begin{bmatrix} x_a \\ y_a \\ 0 \end{bmatrix} \tag{3.13}$$

And the center of the camera frame O (principal point) can be described as:

$$o = \begin{bmatrix} x_p \\ y_p \\ f \end{bmatrix} \tag{3.14}$$

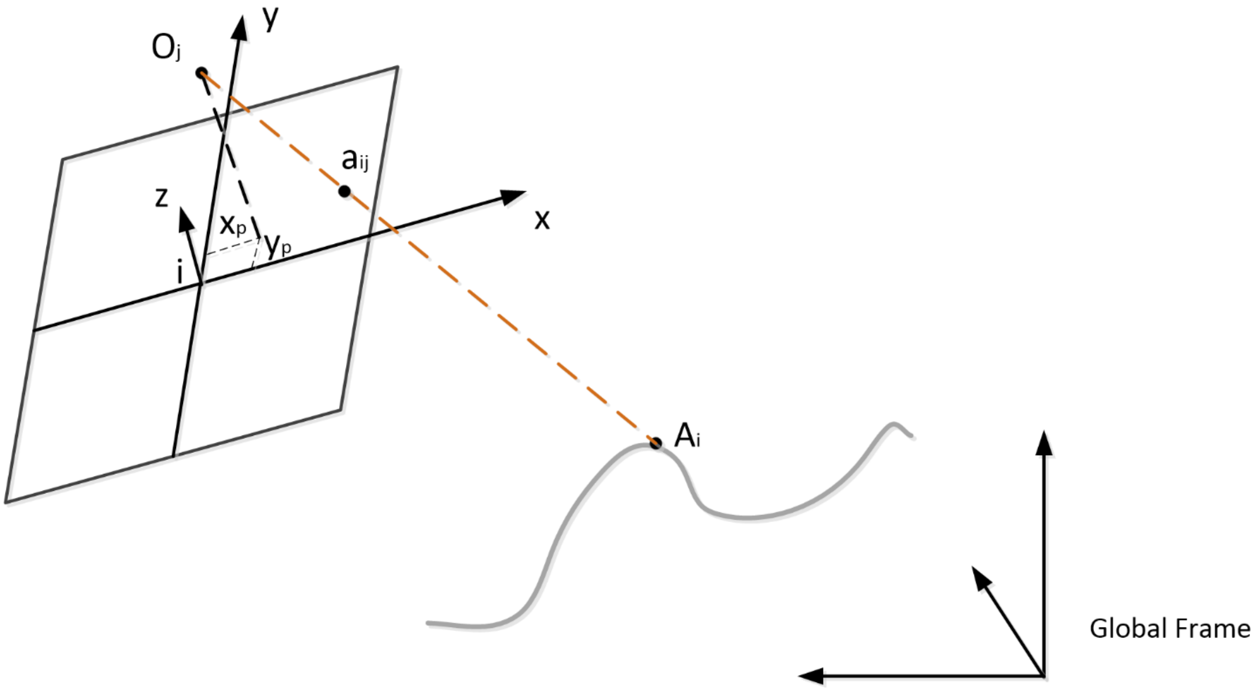


Figure 3-5: Perspective Projection

Once the IOP are obtained, the geometric relation between the 3D object point and the image measurement 2D point can be built based on the perspective projection. The pinhole camera can be ideally modelled as the perspective projection. It means the object point is transformed by a perspective projection matrix to yield the image point. The basis of the perspective projection is the collinearity equations. Collinearity equations define the mathematical relation between the camera, image point and object point. From the collinearity condition, this relation in Figure 3-5 can be expressed as:

$$\lambda_{ij} \left[\begin{bmatrix} x_{ij} \\ y_{ij} \\ 0 \end{bmatrix} - \begin{bmatrix} x_p \\ y_p \\ f \end{bmatrix} \right] = R_g^i\left(\omega_j, \varphi_j, \kappa_j\right) \begin{bmatrix} X_i - X_{O\,j} \\ Y_i - Y_{O\,j} \\ Z_i - Z_{O\,j} \end{bmatrix} \tag{3.15}$$

$$\lambda_{ij}\begin{bmatrix}1 & 0 & -x_p\\0 & 1 & -y_p\\0 & 0 & -f\end{bmatrix}\begin{bmatrix}x_{ij}\\y_{ij}\\1\end{bmatrix} = \begin{bmatrix}R_g^i & -R_g^i C_j\end{bmatrix}\begin{bmatrix}X_i\\Y_i\\Z_i\\1\end{bmatrix}, where\ C_j = \begin{bmatrix}X_{O_j}\\Y_{O_j}\\Z_{O_j}\end{bmatrix} \tag{3.16}$$

When the coordinates are expressed in homogeneous form, the perspective project can be written as:

$$\lambda_{ij}\tilde{x}_{ij} = KD_j\tilde{X}_i = P_j\tilde{X}_i \tag{3.17}$$

where $K = \begin{bmatrix}1 & 0 & -x_p\\0 & 1 & -y_p\\0 & 0 & -f\end{bmatrix}^{-1}$ is the intrinsic calibration matrix that relates the image and camera observations and P_j is called the projection matrix. Due to the fact that the depth of the object point A_i is usually unknown, the positive scale factor λ_{ij} is added to form the perspective projection equation.

3.3.2 Camera Calibration

The purpose of camera calibration is to determine the IOP, EOP and the lens distortion parameters. Specifically, camera calibration solves the intrinsic camera matrix K and the extrinsic parameters D_j from equation (3.6).

First, the convention of the intrinsic calibration matrix need to be clarified. If the image observations (x_{ij}, y_{ij}) are in the principal image coordinate system with its units (e.g. mm), then the intrinsic camera matrix can be written as:

$$K = \begin{bmatrix}1 & 0 & -x_p\\0 & 1 & -y_p\\0 & 0 & -f\end{bmatrix}^{-1} = \begin{bmatrix}1 & 0 & -\frac{x_p}{f}\\0 & 1 & -\frac{y_p}{f}\\0 & 0 & -\frac{1}{f}\end{bmatrix} \tag{3.18}$$

If the image observations are measured in the sensor coordinate frame with its units (e.g. pixel), then the intrinsic camera matrix can be written as:

$$K = \begin{bmatrix} s & 0 & \left(-\frac{width}{2}\right)s - x_p \\ 0 & -s & -\left(-\frac{height}{2}\right)s - y_p \\ 0 & 0 & -f \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{s} & 0 & \frac{c_x}{sf} \\ 0 & -\frac{1}{s} & -\frac{c_y}{sf} \\ 0 & 0 & -\frac{1}{f} \end{bmatrix} = \begin{bmatrix} -\frac{f}{s} & 0 & -\frac{c_x}{s} \\ 0 & \frac{f}{s} & \frac{c_y}{sf} \\ 0 & 0 & 1 \end{bmatrix} \tag{3.19}$$

where $\frac{f}{s}$ is the focal length in pixel units, $width$ and $height$ represent the image's width and height in pixel units, $c_x = \left(-\frac{width}{2}\right)s - x_p$, $c_y = -\left(-\frac{height}{2}\right)s - y_p$.

Due to the different types of distortions, such as lens distortions, image plane distortions, atmospheric refraction and Earth curvature, the theoretical straight line from the object point might not end up at the same position at the observed image point. In general applications, we only consider the calibration of lens distortions.

In this thesis, the MATLAB camera calibration toolbox is used [144] [145]. There are two main lens distortions considered in this toolbox: the radial and the tangential distortions. The radial distortion coefficients can be specified either a two- or three-element vector. This type of distortion is caused by the shape of the lens. Given a camera pixel observations (x_{ij}, y_{ij}), the radial location of the corresponding point can be rescaled on the undistorted output image as:

$$x_{distorted} = x \times (1 + k_1 \times r^2 + k_2 \times r^4 + k_3 \times r^6) \tag{3.20}$$

$$y_{distorted} = y \times (1 + k_1 \times r^2 + k_2 \times r^4 + k_3 \times r^6)$$

where x, y represent the undistorted pixel locations, k_1, k_2, k_3 represent the radial distortion coefficients of the lens, and $r^2 = x^2 + y^2$. Typically, two coefficients are sufficient. For severe distortion, k_3 can be included. The undistorted pixel locations appear in normalized image coordinates, with the origin at the optical center. The coordinates are expressed in world units.

Tangential distortion coefficients are usually given as a two-element vector. This type of distortion occurs when the lens and the image plane are not parallel. Regarding the tangential distortion, the rectified image can be computed as:

$$x_{distorted} = x + [2 \times p_1 \times x \times y + p_2 \times (r^2 + 2 \times x^2)] \tag{3.21}$$

$$y_{distorted} = y + [p_1 \times (r^2 + 2 \times y^2) + 2 \times p_2 \times x \times y]$$

The general camera calibration process using the MATLAB camera calibration toolbox can be summarized as [144]:

1. Prepare images, camera, and calibration pattern (usually checker board);
2. Import images and select the corresponding camera model;
3. Calibrate the camera;
4. Evaluate the calibration accuracy;
5. Adjust parameters to improve the accuracy;
6. Export the calibrated matrices.

The detailed camera calibration results will be shown in Chapter 4.

3.3.3 Feature Point Extraction and Matching

As was reviewed in Chapter 2, feature detection and matching are the second part of the feature-based VO pipeline. There are a number of methods with pros and cons in extracting the feature positions. ORB feature detection is usually deployed for some real-time scenarios while SIFT and SURF are better with post processing for higher accuracy. In this section, the SIFT method will be introduced and deployed in the later experiments.

The Scale-Invariant Feature Transform (SIFT) is a classic and accurate feature extraction method which is invariant to scale and rotation. This method is robust to affine distortions and linear illumination changes, but correspondingly requires a larger amount of calculation [42]. The SIFT is comprised of two parts: extracting SIFT keypoints and calculating SIFT descriptors.

The procedure of detecting SIFT keypoints can be briefly summarized as [42]:

1. Creating image pyramid (octaves at the spatial domain) and Gaussian smoothing (scale domain).
2. Differentiating Gaussians and finding extrema (initial keypoint) at all octaves.
3. Precise localization of keypoints.
4. Removing low-contrast keypoints.
5. Removing edge response.

After precisely detecting the location of keypoints, the corresponding descriptors are calculated as [42]:

1. Assign an orientation to each keypoint.
2. Computing the gradient orientation histograms for each keypoint.
3. Stack the histograms together into a vector and normalize the vector.

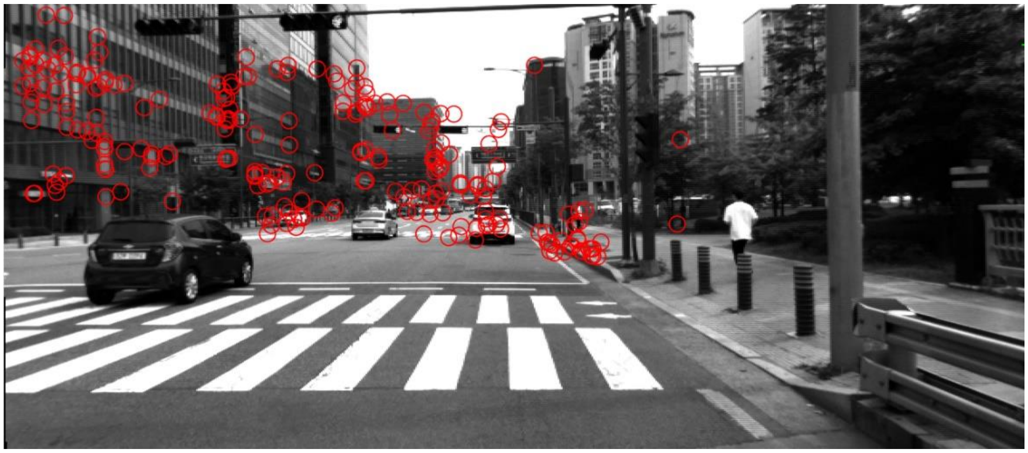


Figure 3-6: An example of detected SIFT feature using the KAIST Complex Urban Dataset (trajectory 39)

Feature matching in an essential step for VO. More specifically, feature matching solves the "data association" problem in VO, which is to determine the relations between the feature in current image and previous one. By accurately matching the descriptors between image and image, or between image and map, the pose estimation and optimization process can be proceeded. However, due to the local characteristics of image features, mismatching remains a problem and limits the long-term operations for VO system. The most straightforward feature matching method is the Brute-Force Matcher. Given keypoint i from the first image, search for the keypoint j from the second image whose feature vector has the shortest Euclidean distance from the feature vector of keypoint i:

$$arg \min_{j}(s_{i,j} = \left\| d_i = d_j \right\|) \tag{3.22}$$

For binary descriptors (such as BRIEF), Hamming distance is often used instead of Euclidean distance. However, when the number of feature points is large, the computational complexity of the brute force matching method will become too large, especially when we want to match a frame and a map. To improve the computational efficiency, the fast approximate nearest neighbor (FLANN) algorithm (which is included in the OpenCV library) is more suitable for the situation with a large number of matching points [146].

3.3.4 Epipolar Geometry (2D-2D)

After feature matching, the corresponding relations between features are obtained. Next step is to estimate the pose information by exploiting the geometry, which can summarized as:

- Epipolar Geometry: 2D-2D geometry between two monocular images (For monocular VO initialization).
- Perspective-n-Point Pose Problem (PnP): 3D-2D matching between image frame to map.

- Iterative Closest Point (ICP): 3D-3D matching between frame to map (point cloud).

In this thesis, a monocular image stream is utilized, thus the Epipolar Geometry and PnP algorithm are presented in this chapter respectively.

Considering two overlapped images I_1 and I_2 (Figure3-7), the projections of the 3D point A_i on the two image planes are $a_{i,1}$ and $a_{i,2}$ respectively. The $A_iO_1O_2$ plane is called the Epipolar Plane. The intersection of the epipolar plane with the two image planes creates two lines $a_{i,1}e_1$ and $a_{i,2}e_2$, these lines are known as the corresponding Epipolar Lines, and e_1, e_2 are called Epipoles. All the epipolar lines intersect at the epipoles. Epipolar Constraint means that given a point $a_{i,1}$ in the first image, its corresponding point in the second image is constrained to lie on the epipolar line. When in practice, $a_{i,1}$ and $a_{i,2}$ can be obtained by feature matching, A_i, e_1 and e_2 are unknown. By utilizing epipolar geometry, the transformation matrix T_{12} can be obtained as a result.

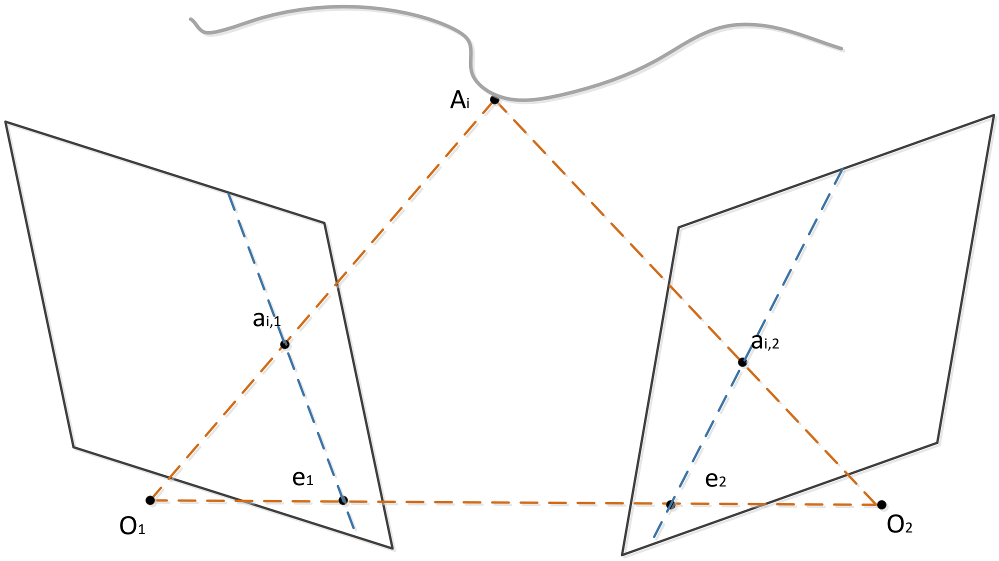


Figure 3-7: Epipolar Geometry

According to Section 3.3.1, the following perspective projection equations can be formed:

$$\tilde{x}_{i,1} = K_1 \begin{bmatrix} R_g^1 & -R_g^1 C_1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} = K_1 R_g^1 G_i - K_1 R_g^1 C_1 \tag{3.23}$$

$$\tilde{x}_{i,2} = K_2 \begin{bmatrix} R_g^2 & -R_g^2 C_2 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} = K_2 R_g^2 G_i - K_1 R_g^2 C_2$$

Combining the two above projection equations:

$$\left(K_1 R_g^1\right)^{-1} \tilde{x}_{i,1} + C_1 = \left(K_2 R_g^2\right)^{-1} \tilde{x}_{i,2} + C_2 \tag{3.24}$$

And the relations between the $a_{i,1}$ and $a_{i,2}$ can be formulated as:

$$R_1^g (K_1)^{-1} \tilde{x}_{i,1} = (C_2 - C_1) + R_2^g (K_2)^{-1} \tilde{x}_{i,2} \tag{3.25}$$

Applying cross product with $(C_2 - C_1)$ to both sides of (3.25):

$$[C_2 - C_1]_\times \left(R_1^g (K_1)^{-1} \tilde{x}_{i,1}\right) = [C_2 - C_1]_\times R_2^g (K_2)^{-1} \tilde{x}_{i,2} \tag{3.26}$$

$$\left(R_1^g (K_1)^{-1} \tilde{x}_{i,1}\right)^T [C_2 - C_1]_\times \left(R_1^g (K_1)^{-1} \tilde{x}_{i,1}\right) = \left(R_1^g (K_1)^{-1} \tilde{x}_{i,1}\right)^T [C_2 - C_1]_\times R_2^g (K_2)^{-1} \tilde{x}_{i,2}$$

Finally, the Epipolar equation can be written as:

$$\begin{aligned} 0 = \tilde{x}_{i,1}^T \left((K_1^{-T}) R_g^1 [C_2 - C_1]_\times R_2^g (K_2)^{-1} \right) \tilde{x}_{i,2} &= \tilde{x}_{i,1}^T \left((K_1^{-T}) [T_{12}]_\times R_2^1 (K_2)^{-1} \right) \tilde{x}_{i,2} \\ &= \tilde{x}_{i,1}^T F \tilde{x}_{i,2} \end{aligned} \tag{3.27}$$

where $F = \left((K_1^{-T}) [T_{12}]_\times R_2^1 (K_2)^{-1} \right)$ is called Fundamental Matrix. The DoF of the fundamental matrix is 7.

If the image observations $\tilde{x}_{i,1}$ and $\tilde{x}_{i,2}$ are already calibrated, then the Epipolar equation can be simplified as:

$$0 = \left(\tilde{x}_{i,1}^c\right)^T ([T_{12}]_\times R_2^1) \tilde{x}_{i,2}^c = \left(\tilde{x}_{i,1}^c\right)^T E \tilde{x}_{i,2}^c \tag{3.28}$$

where $E = ([T_{12}]_\times R_2^1)$ is called the Essential Matrix. The DoF of the essential matrix is 5.

Recall the fact that two images have total 12 EOPs, however, the essential matrix only has 5 DoF, which means that there can only be 5 independent parameters to describe the relative orientation between two calibrated images.

In order to determine the relative orientation parameters directly, there are several methods to achieve that [147]:

- Limited 8-point method (for either E or F)
- Minimal 7-point method (for F)
- Minimal 5-point method (for E)
- Minimal 2-point method (for E with known rotation)
- Minimal 2-point method (for E given an object symmetric with respect to a 3D plane)\

In this thesis, we apply the vision algorithm in large scale environment, which will usually have enough matched point pairs to conduct RO. Thus, the 8-point method is applied here.

Simplifying the epipolar equation as follows:

$$x_{i,1}x_{i,2}f_{00} + x_{i,1}y_{i,2}f_{01} + x_{i,1}f_{02} + x_{i,2}y_{i,1}f_{10}$$

$$+y_{i,1}y_{i,2}f_{11} + y_{i,1}f_{12} + x_{i,2}f_{20} + y_{i,2}f_{21} + f_{22} = 0 \tag{3.29}$$

$$A_{n\times 9}f_{9\times 1} = 0 \tag{3.30}$$

where $A = [x_{i,1}x_{i,2} \quad x_{i,1}y_{i,2} \quad x_{i,1} \quad x_{i,2}y_{i,1} \quad y_{i,1}y_{i,2} \quad y_{i,1} \quad x_{i,2} \quad y_{i,2} \quad 1\,]$, and $f =$ $[f_{00} \quad f_{01} \quad f_{02} \quad f_{10} \quad f_{11} \quad f_{12} \quad f_{20} \quad f_{21} \quad f_{22}]^T$.

Since the fundamental matrix F is defined up to an arbitrary scale, by constraining $\|f\| = 1$, the whole problem becomes: $A_{n\times 9}f_{9\times 1} = 0$ subject to $\|f\| = 1$.

Step of solving $A_{n\times 9}$ can be summarized as [34]:

1. Decompose A as $A = USV^T$;
2. Check the diagonal elements of S are sorted non-increasingly;

3. Extract the last column of V and reshape to the 3×3 fundamental matrix $\hat{F}$;

4. Decompose $\hat{F}$ as $\hat{F} = USV^T$;

5. Check the diagonal elements of S are sorted non-increasingly;

6. Impose the rank-2 constraint: $S_{new} = \frac{1}{\sqrt{s_{0,0}^2+s_{1,1}^2}} diag([s_{0,0} \quad s_{1,1} \quad 0])$;

7. Compute the new fundamental matrix with rank-2: $\hat{F} = US_{new}V^T$;

8. Given the IOP, recover the essential matrix from the fundamental matrix: $\hat{E} = (K_1)^T \hat{F} K_2$;

9. Decompose $\hat{E}$ as $\hat{E} = USV^T$;

10. Check the diagonal elements of S are sorted non-increasingly;

11. Impose the constraint: $S_{new} = diag([1 \quad 1 \quad 0])$;

12. Compute the new essential matrix: $\hat{E} = US_{new}V^T$;

13. Recover R and t from $\hat{E}/\hat{F}$ (t does not include scale information).

It should be noted that if the points lie on the same 3D plane or are very close to a 3D plane, the direct 8-point method is unusable/unstable. In addition, if the camera does not move or only rotates, the 8-point method cannot yield a solution. And if there are more than 8 points, least-square method will usually be utilized rather than the direct method.

3.3.5 Triangulation and Depth Estimation

After the rotation and translation of the camera are determined, the scale, or the depth of the corresponding feature remains unknown. The common approach is to form a simple least squares to estimate the depth by triangulation.

Considering two scale factors s_1 and s_2 relates to the unknown feature:

$$s_1 \tilde{x}_{i,1}^c = s_2 R \tilde{x}_{i,2}^c + t \tag{3.31}$$

$$[-R\tilde{x}_{i,2}^{c} \quad \tilde{x}_{i,1}^{c}]\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = t \tag{3.32}$$

$$Ax = b \tag{3.33}$$

where, $A = [-R\tilde{x}_{i,2}^{c} \quad \tilde{x}_{i,1}^{c}],\ x = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}, t = b.$

In this least squares problem:

$$x = (A^T A)^{-1} A^T b \tag{3.34}$$

The premise of triangulation is translation. If the camera does not move or only rotates, epipolar constraints will always be satisfied. Therefore, to improve the accuracy of triangulation, one way is to improve the accuracy of feature point detection. Another way is to increase the amount of translation. However, an increase in the amount of translation will cause obvious changes in the appearance of the image. The appearance changes can make feature extraction and matching difficult.

3.3.6 PnP (3D-2D)

Perspective-n-Point problem refers the process of estimating the transformation from 3D point to a 2D point. It describes how to estimate the pose of a camera when n 3D feature points and their perspective projections are known. As mentioned in Section 3.3.4, the 2D-2D epipolar geometry method requires eight or more point pairs (take the 8-point method as an example), and there are problems with the initialization, pure rotation, and scale. However, if the 3D position of one of the features in the two images is known, then at least point pairs (at least one additional point verification result is required) can be used to estimate the camera motion. The 3D feature position can be determined by triangulation or the depth map if using an RGB-D camera. Therefore, in stereo or RGB-D VO, PnP can be directly used to estimate camera motion. In the monocular VO, the 3D feature position must be initialized before using PnP. The 3D-2D method does not require epipolar geometry constraints, and can obtain better motion estimation with fewer matching points.

The PnP problem has many solutions, such as Direct Linear Transformation (DLT), P3P [148], EPnP [149], UPnP [150], and so on. Besides, PnP can also be solved by Bundle Adjustment (BA) methods.

First and foremost, the DLT method is a straightforward algebraic solution. DLT of the perspective projection directly relates the inhomogeneous coordinates of the object points with the coordinate of image points of a straight line preserving perspective camera:

$$x_{ij} = \frac{p_{00}X_i + p_{01}Y_i + p_{02}Z_i + p_{03}}{p_{20}X_i + p_{21}Y_i + p_{22}Z_i + p_{23}} \tag{3.35}$$

$$y_{ij} = \frac{p_{10}X_i + p_{11}Y_i + p_{12}Z_i + p_{13}}{p_{20}X_i + p_{21}Y_i + p_{22}Z_i + p_{23}}$$

where: $P_j = K\left[R_g^j \quad -R_g^j C_j\right] = \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{02} \\ p_{10} & p_{11} & p_{12} & p_{13} \\ p_{20} & p_{21} & p_{22} & p_{23} \end{bmatrix}$, with $d.o.f = 11$.

Given n known 3D object points and their corresponding image observations, there will be $2n \times 12$ linear constraints to the problem:

$$A_{2n\times 12}P_{12\times 1} = 0 \tag{3.36}$$

where: $A = \begin{bmatrix} X_i & Y_i & Z_i & 1 & -x_{ij}X_i & -x_{ij}Y_i & -x_{ij}Z_i & x_{ij} \\ X_i & Y_i & Z_i & 1 & -y_{ij}X_i & y_{ij}Y_i & -y_{ij}Z_i & -y_{ij} \end{bmatrix}$, and $P =$

$[p_{00} \quad p_{10} \quad p_{20} \quad p_{01} \quad p_{11} \quad p_{21} \quad p_{02} \quad p_{12} \quad p_{22} \quad p_{03} \quad p_{13} \quad p_{23}]^T$.

Since the perspective projection matrix is defined up to an arbitrary scale, we can impose an arbitrary constraint on the norm of P, and in particular we can set: $\|P\| = 1$. Decompose A as $A = USV^T$, the solution $\hat{P}$ will be the last column of V.

The DLT method treats R, t as independent unkowns, and it requires at least 6 pairs of feature points, thus it is also been called P6P. In this thesis, a more efficient method P3P is utilized [148].

Given three non-collinear points and their corresponding calibrated image coordinates (Figure 3-8).

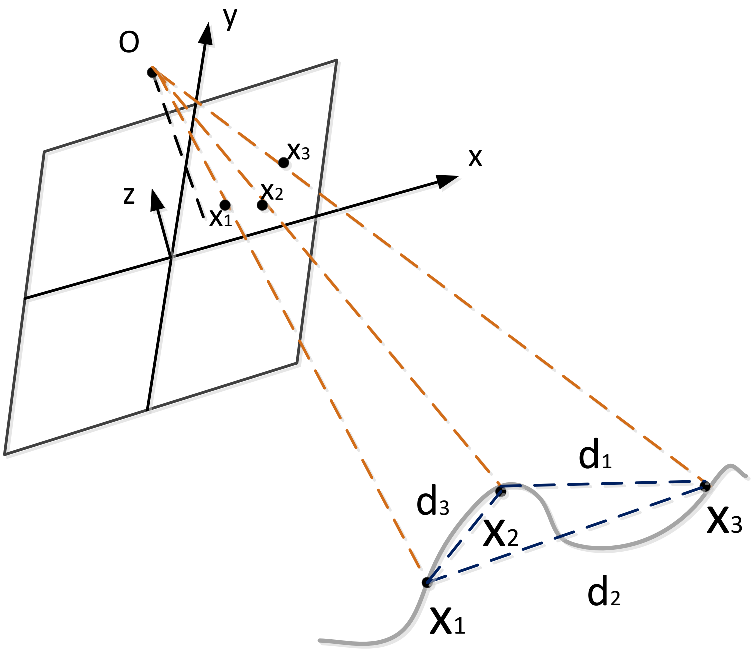


Figure 3-8: P3P

The image observations can be written in the camera frame as:

$$x_1^c = \begin{bmatrix} x_1 - x_{pp} \\ y_1 - y_{pp} \\ -f \end{bmatrix} \tag{3.37}$$

$$x_2^c = \begin{bmatrix} x_2 - x_{pp} \\ y_2 - y_{pp} \\ -f \end{bmatrix}$$

$$x_3^c = \begin{bmatrix} x_3 - x_{pp} \\ y_3 - y_{pp} \\ -f \end{bmatrix}$$

where (x_{pp}, y_{pp}) represent the principle point.

From Figure-3, it can be noted that $O - X_1X_2X_3$ forms a triangular pyramid shape, thus:

$$\begin{cases} \theta_3 = \angle(x_1^c, O, x_2^c) = \operatorname{acos}\left(\dfrac{x_1^c \cdot x_2^c}{|x_1^c||x_2^c|}\right) \\ \theta_2 = \angle(x_1^c, O, x_3^c) = \operatorname{acos}\left(\dfrac{x_1^c \cdot x_3^c}{|x_1^c||x_3^c|}\right) \\ \theta_1 = \angle(x_2^c, O, x_3^c) = \operatorname{acos}\left(\dfrac{x_2^c \cdot x_3^c}{|x_2^c||x_3^c|}\right) \end{cases} \tag{3.38}$$

$$\begin{cases} d_3 = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2 + (Z_2 - Z_1)^2} \\ d_2 = \sqrt{(X_3 - X_1)^2 + (Y_3 - Y_1)^2 + (Z_3 - Z_1)^2} \\ d_1 = \sqrt{(X_2 - X_3)^2 + (Y_2 - Y_3)^2 + (Z_2 - Z_3)^2} \end{cases} \tag{3.39}$$

$$\begin{cases} r_1 = \sqrt{(X_O - X_1)^2 + (Y_O - Y_1)^2 + (Z_O - Z_1)^2} \\ r_2 = \sqrt{(X_O - X_2)^2 + (Y_O - Y_2)^2 + (Z_O - Z_2)^2} \\ r_3 = \sqrt{(X_O - X_3)^2 + (Y_O - Y_3)^2 + (Z_O - Z_3)^2} \end{cases} \tag{3.40}$$

According to the law of cosines for the triangles:

$$\begin{cases} d_1^2 = r_2^2 + r_3^2 - 2r_2r_3 \cos\theta_1 \\ d_2^2 = r_1^2 + r_3^2 - 2r_1r_3 \cos\theta_2 \\ d_3^2 = r_2^2 + r_1^2 - 2r_2r_1 \cos\theta_3 \end{cases} \tag{3.41}$$

Solving the equations for r_1:

$$r_1^2 = \frac{d_1^2}{u^2 + v^2 - 2uv\cos\theta_1} = \frac{d_2^2}{1 + v^2 - 2v\cos\theta_2} = \frac{d_3^2}{1 + u^2 - 2u\cos\theta_3} \tag{3.42}$$

where: $u = r_2/r_1$, $v = r_3/r_1$.

Substitute u in terms of v:

$$A_4v^4 + A_3v^3 + A_2v^2 + A_1v^1 + A_0 = 0 \tag{3.43}$$

The coefficients of the above equation depend on the known values of $d_1, d_2, d_3, \theta_1, \theta_2, \theta_3$.

Solving the polynomial equation results up to 4 solutions for v, u, r_1, r_2 and r_3. Therefore, a 4th point is needed to validate the EOP solution.

Now from collinearity equations, for each point $(i = 1,2,3)$:

$$r_i x_i^c = R(X_i - C) \tag{3.45}$$

Eventually, the 3D rigid body transformation with unknown EOP R, C that transform X_i to $r_i x_i^c$ are obtained. The drawback of this method is that you always need another point to validate which of the four possible solution is correct. On the other hand, the advantage is that it requires less features to estimate the camera motion, which is very useful in some homogenously textured scenes (such as snow environment). In addition, for outlier detection algorithms like RANSAC, the least the number of points required for model estimation, the better.

3.3.7 Outlier Detection

Outlier detection is an very crucial part in order to improve the robustness of VO. More specifically, in VO systems, outlier detection algorithms are used to reject outliers caused by moving objects in dynamic environments or wrongly matched feature points. The classic approach of outlier detection is Random Sample Consensus (RANSAC). RANSAC was first proposed by Fisher and Bolles in 1981 [151] to interpret/smooth data containing a significant percentage of gross errors. There are basically two steps of RANSAC algorithm: (1) generating a hypothesis from random sample; (2) verifying the hypothesis to the data and finding inliers [151]. RANSAC is an iterative algorithm but not require complex optimization since the basic idea is random sampling. The workflow of the traditional RANSAC algorithm is shown in Figure 3-9.

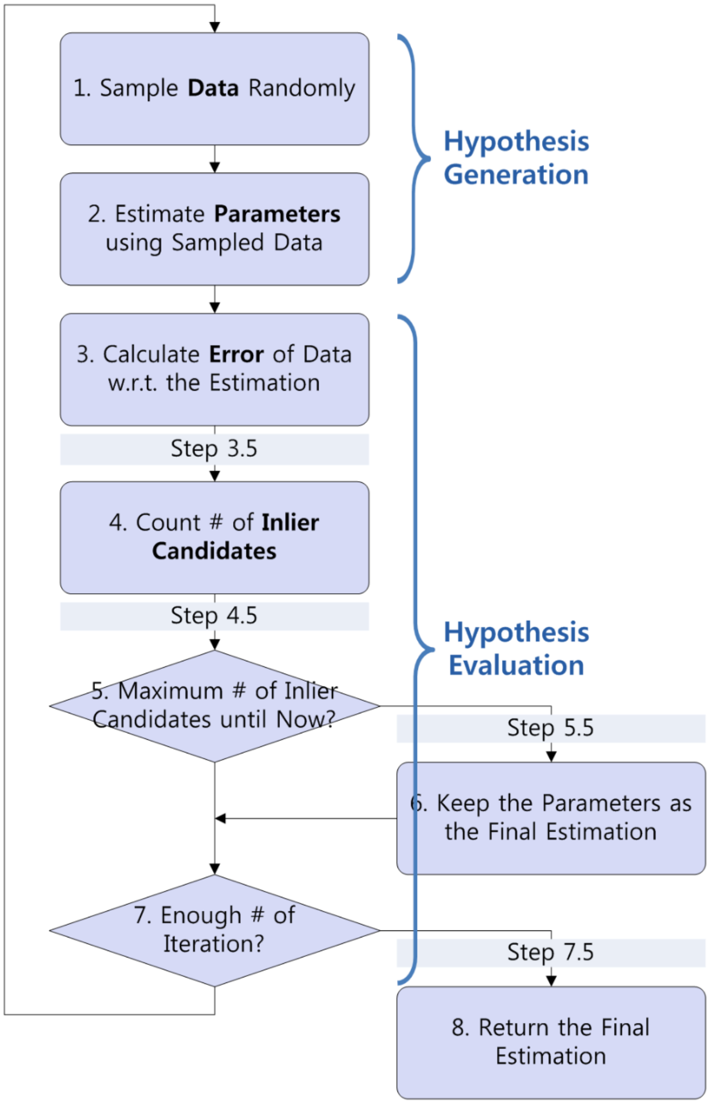

1. Sample Data Randomly
2. Estimate Parameters using Sampled Data
3. Calculate Error of Data w.r.t. the Estimation
Step 3.5
4. Count # of Inlier Candidates
Step 4.5
5. Maximum # of Inlier Candidates until Now?
Step 5.5
6. Keep the Parameters as the Final Estimation
7. Enough # of Iteration?
Step 7.5
8. Return the Final Estimation
Hypothesis Generation
Hypothesis Evaluation

Figure 3-9: Flowchart of RANSAC from [152]

For VO systems, RANSAC is especially valuable in epipolar geometry estimation and camera motion estimation. In this section, RANSAC will be introduced with an example in 8-point algorithm outlier detection.

The first step is to randomly choose a minimal number (subset) of 8 corresponding feature points, then estimate the fundamental/essential matrix from this chosen subset using the 8-point algorithm. Then check the consistency of the estimated solution with other corresponding points. For any point i that is not in the subset of 8 points, the residual can be calculated by:

$$\hat{v}_i = \left|\left(\tilde{x}_{i,1}\right)^T \hat{F} \tilde{x}_{i,2}\right| \tag{3.46}$$

There is a threshold (in this case, 0.01) set to determine the number of points that are consistent with the estimated solution. The number of selecting subsets as the maximum number of attempts to find a consensus set is defined by [151]:

$$k = \frac{\log(1 - z)}{\log(1 - (1 - b)^n)} \tag{3.47}$$

where: n denotes the minimal number, z represents the probability of the data which are outlier-free, b is the assumed inlier ratio. If the maximum iteration is met, then the procedure terminates. The tentative solution with the maximum support is the correct solution. The observations that support this solution are inlier. Otherwise, continue to randomly select points and do the consistency check.

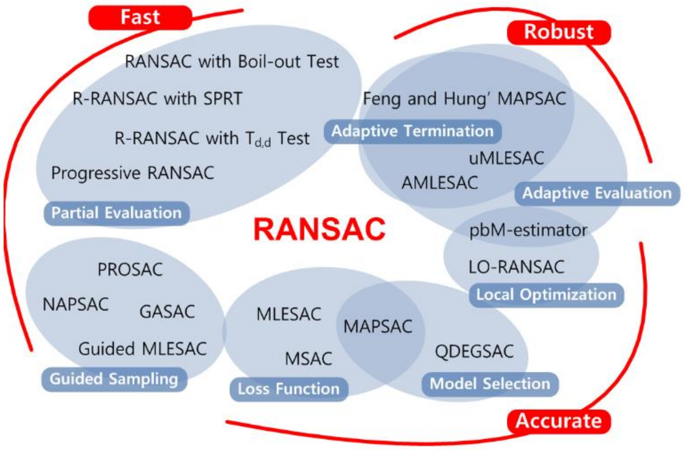


Figure 3-10: RANSAC Family from [152]

To date, there are many variations or improved versions of RANSAC (Figure 3-10). The consistency check step in the traditional RANSAC deploys a loss function as:

$$Loss(e) = \begin{cases} o & |e| < c \\ const & otherwise \end{cases} \qquad (3.48)$$

where c is the threshold. RANSAC has a constant loss at large error [152].

In this thesis, MSAC (M-estimator SAC) is employed. The only difference between MSAC and RANSAC is that MSAC adopts M-estimator to bound the loss function as:

$$Loss(e) = \begin{cases} e^2 & |e| < c \\ c^2 & otherwise \end{cases} \qquad (3.49)$$

3.4 Wheel Odometer Aided Multi-State Constrained Kalman Filter

After introduced the workflow of VO algorithm, in this section, a novel approach that incorporates wheel odometry and NHC together with tightly-coupled monocular visual-inertial odometry using the MSCKF will be proposed and discussed in detail.

As has been mentioned in Chapter 2, the usual approach to EKF-based VIO involves augmenting an inertial navigation filter with additional states for feature points tracked over multiple images As an alternative, the MSKCF developed by [128] augments an inertial navigation filter with additional states for the camera poses corresponding to a series of images that contain common features. Each feature in each frame then provides a constraint relating the camera pose states and the inertial navigation solution.

The workflow of MSCKF can be summarized as [153]:

1. Propagate the whole state and covariance matrix using IMU measurements.
2. When a new image arrivals:
    a. State augmentation: augment the state vector and the covariance matrix with the current IMU state.

b. Feature detection and matching.

c. For each feature which completes tracking, compute the residual term and design matrix. Run the outlier detection algorithm, use all the salient features to perform the EKF update.

d. Remove the corresponding features from the state vector.

The MSCKF has several advantages. First, it does not require accurate initial depth information and covariance to maintain a consistent solution since feature point positions are no longer included in the state vector. Second, by using each feature to constrain multiple states, the camera pose estimation is improved. Finally, the complexity of the algorithm is linear in the number of features, rather than quadratic as is the case in traditional EKF-based approaches. However, even with the help of inertial measurements, VIO is still subject to scale drift because it estimates forward direction translation using distant feature points that are generally located only in the forward direction. This leads to drift in the velocity solution which will then degrade the position estimate.

The proposed algorithm augments the monocular MSCKF method with wheel odometry (WO) and non-holonomic constraints (NHC) to bound the cumulative velocity error. Wheel speed can be easily obtained from the CANBUS port on most modern vehicles. The forward vehicle speed can then be combined with NHC pseudo-measurements of zero across-track and vertical velocity components to update the MSCKF. However, both visual and wheel odometry techniques for ground vehicles can be particularly challenging in winter conditions since imaging sensors suffer from the low-texture environment and potential harsh weather (snow, fog, mist) while wheel slippage will be magnified in ice and snow. Providing a continuous and robust navigation solution in urban winter road environments remains an open problem.

In [137], the authors compare MSCKF with a sliding window filter using the KITTI Dataset [1]. To avoid the impact of the velocity drift, the authors removed the velocity from the state vector and fed the gravity-corrected linear ground truth velocity, as measurements, to the IMU mechanization. The simplified $(12 + 6N)$ state MSCKF was shown to perform well with the KITTI data, however the availability of high-precision velocity is not realistic in most land vehicle navigation scenarios. In this thesis, publicly available code from [137] to employ a $(15 + 6N)$ state MSCKF filter as proposed in [128]. In addition, the SIFT feature detection [42] method is added to replace the SURF method used in [137] has been augmented. Wheel odometer measurements and NHC are incorporated into the MSCKF as additional measurements updating the $(15 + 6N)$ states.

3.4.1 System Model

The MSCKF takes full advantage of the constraints that a set of environment feature points provide, however, a monocular camera identifying corresponding feature points mainly in the forward direction provides a poor constraint in terms of forward translation and speed. Thus, the system still suffers from the scale drift in forward velocity. By integrating wheel odometer measurements and NHC, the scale drift issue in monocular MSKCF system should be improved significantly.

Figure 3-11 shows a general overview of the 15-state MSKCF augmented with WO and NHC. The state vector x is updated using the 6-DoF INS output u_{INS}, wheel odometer measurements z_{odo} and camera measurements z_{cam}.

According to the Bayes' theorem, the problem can be stated as:

$$p(x_k|z_k, u_k) = p(z_k|x_k)p(x_k|x_{1:k-1}, u_k)$$

$$= p\left(z_{odo,k} \middle| x_k\right) p\left(z_{cam,k} \middle| x_k\right) p\left(x_k \middle| x_{1:k-1}, u_{INS,k}\right) \qquad (3.50)$$

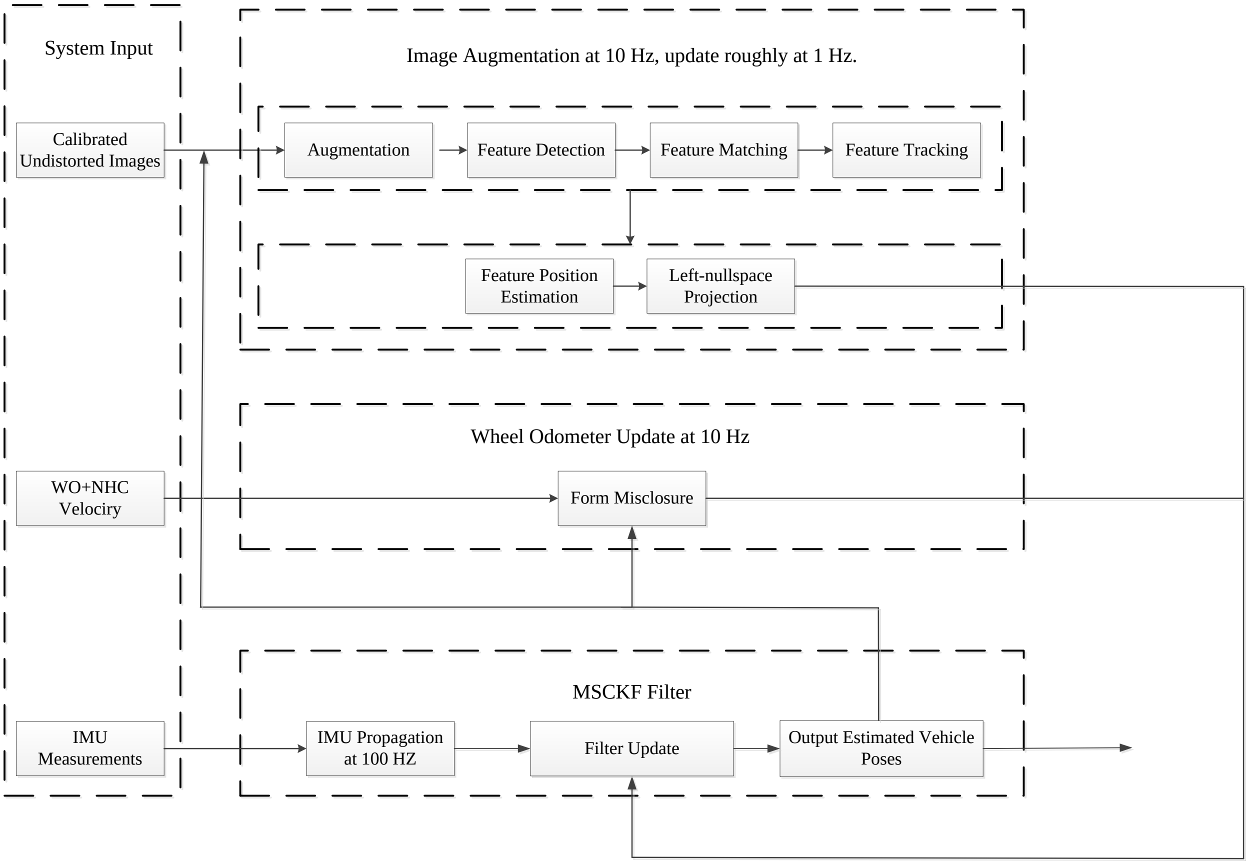

System Input
Calibrated Undistorted Images
WO+NHC Velociry
IMU Measurements
Image Augmentation at 10 Hz, update roughly at 1 Hz.
Augmentation
Feature Detection
Feature Matching
Feature Tracking
Feature Position Estimation
Left-nullspace Projection
Wheel Odometer Update at 10 Hz
Form Misclosure
MSCKF Filter
IMU Propagation at 100 HZ
Filter Update
Output Estimated Vehicle Poses

Figure 3-11: The Workflow of Wheel Odometry aided MSCKF.

Thus, our system can be described as one dynamics model $f(\cdot)$ that propagates the whole state using INS output and two measurement models, $h_{odo}(\cdot)$, $h_{cam}(\cdot)$ that can be linearized to design matrices that project the state vector into the two measurement spaces:

$$x_k = f_{INS}\left(x_{k-1}, u_{INS,k}\right) + n_{INS}$$

$$\hat{z}_{odo,k} = h_{odo}(x_k) + n_{odo}$$

$$\hat{z}_{cam,k} = h_{cam}(x_k) + n_{cam} \qquad (3.51)$$

where, n_{INS} is the process noise, n_{odo} and n_{cam} are the measurement noises.

3.4.2 Strapdown IMU Mechanization

Strapdown inertial system means that the inertial sensors are rigidly mounted on the vehicle. Compared to the gimbaled systems, strapdown systems are more popular in many applications due to their low cost and smaller size. In this thesis, strapdown inertial navigation system is used, which means that INS represents strapdown INS.

As has been mentioned in Chapter 2, an IMU usually includes 3 orthogonal accelerometers and 3 orthogonal gyroscopes that measure the specific force and angular velocity. The INS mechanization is essentially a time integration process using the input information given the initial navigation state [63]. The mechanization process is showed in Figure 3-12. For the land vehicle navigation systems, the navigation frame is chosen as the local-level frame.

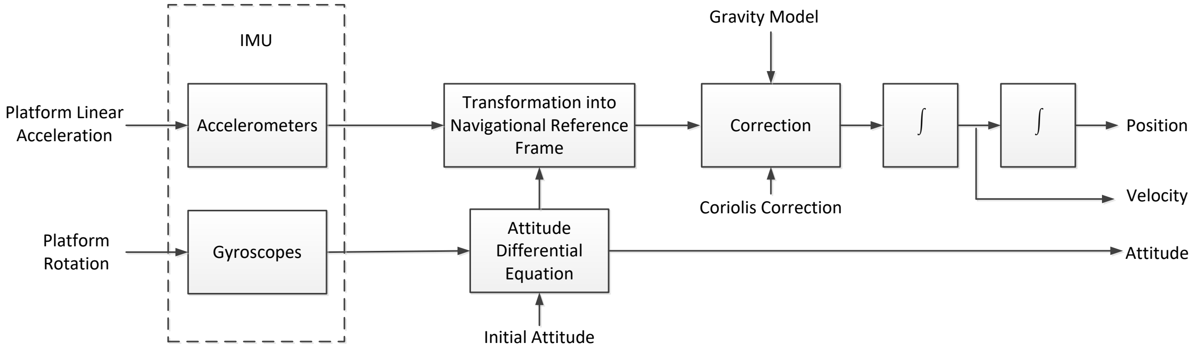


Figure 3-12: Strapdown INS Mechanization Workflow after [65]

In order to incorporate the wheel odometer measurements into MSCKF, the INS state vector is defined as,

$$x_{INS} = \left({}_{G}^{I}q^{T} \quad b_{g}^{T} \quad {}^{G}v_{I}^{T} \quad b_{a}^{T} \quad {}^{G}p_{I}^{T} \right)^{T} \tag{3.52}$$

where ${}_{G}^{I}q$ is the 4×1 unit quaternion vector that represents the rotation from the Global frame $\{G\}$ to the IMU body frame $\{I\}$. $b_g, b_a \in \mathbb{R}^3$ are the biases of the measured gyroscope and accelerometer readings from the IMU, respectively. ${}^{G}v_I \in \mathbb{R}^3$ and ${}^{G}p_I \in \mathbb{R}^3$ describes the IMU velocity and position in the Global frame $\{G\}$.

Because this is an extended Kalman filter, the error state $\delta x_{INS} \in \mathbb{R}^{15}$ given by:

$$\tilde{x}_{INS} = \begin{pmatrix} {}_{G}^{I}\tilde{\theta}^T & \tilde{b}_g^T & {}^{G}\tilde{v}_I^T & \tilde{b}_a^T & {}^{G}\tilde{p}_I^T \end{pmatrix}^T \tag{3.53}$$

is used, where, ${}_{G}^{I}\tilde{\theta} \in \mathbb{R}^3$ represents the perturbation of the INS attitude in the body frame. In the quaternion form, the error is defined as: ($\otimes$ denotes quaternion multiplication)

$$\delta q = q \otimes \hat{q}^{-1} \approx \begin{pmatrix} \frac{1}{2}\, {}_{G}^{I}\tilde{\theta}^T & 1 \end{pmatrix}^T \tag{3.54}$$

In this form, the attitude errors are reduced to its minimal representation which corresponds to 3 DoF.

The INS continuous-time kinematics model is given as:

$$\begin{gathered} {}_{G}^{I}\dot{\hat{q}} = \frac{1}{2} \times {}_{G}^{I}\hat{q} \otimes \widehat{\omega} = \frac{1}{2} \times \Omega(\widehat{\omega}) \times {}_{G}^{I}\hat{q} \\ \dot{\hat{b}}_g = 0_{3\times 1} \\ {}^{G}\dot{\hat{v}} = C({}_{G}^{I}\hat{q}\ )^T \hat{a} + {}^{G}g, \\ \dot{\hat{b}}_a = 0_{3\times 1} \\ {}^{G}\dot{\hat{p}}_I = {}^{G}\hat{v} \end{gathered} \tag{3.55}$$

where $\widehat{\omega}$ and $\hat{a}$ are obtained subtracting the biases from the measurements.

$\widehat{\omega} = \omega_{meas} - \hat{b}_g, \hat{a} = a_{meas} - \hat{b}_a.$

$\Omega(\hat{\omega}) = \begin{pmatrix} -\lfloor \hat{\omega}_\times \rfloor & \omega \\ -\omega^T & 0 \end{pmatrix}$; $\lfloor \hat{\omega}_\times \rfloor = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$ is the skew-symmetric matrix; and $C(\cdot)$ denotes the function converting quaternion to the corresponding rotation matrix.

The linearized continuous INS system model is given as:

$$\dot{\tilde{x}}_I = F\tilde{x}_I + Gn_I \tag{3.56}$$

In which n_I is the Gaussian noise $n_I = \begin{pmatrix} n_g^T & n_{wg}^T & n_a^T & n_{ag}^T \end{pmatrix}^T$, n_g^T and n_a^T represents the gyroscope and accelerometer noises, n_{wg}^T and n_{ag}^T represents the random walk rate of the gyroscope and accelerometer measurement biases.

With Jacobian matrices F and G given as:

$$F = \begin{bmatrix} -\lfloor \hat{\omega}_\times \rfloor & -I_3 & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ -C({}_G^I\hat{q})^T \lfloor \hat{a}_\times \rfloor & 0_{3\times3} & 0_{3\times3} & -C({}_G^I\hat{q})^T & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 & 0_{3\times3} & 0_{3\times3} \end{bmatrix}_{15\times15} \tag{3.57}$$

$$G = \begin{bmatrix} -I_3 & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_3 & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & -C({}_G^I\hat{q})^T & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_3 \end{bmatrix}_{15\times12} \tag{3.58}$$

For the discrete-time implementation, the IMU error state transition matrix Φ_k comes from integration equations $\dot{\Phi}_k = F(t)\Phi_k$, where $F(t)$ is the Jacobian of the continuous-time system model for the IMU motion [154]. More specifically, in the GPS/INS community, one step approximation $\Phi = \mathrm{I} + \mathrm{F}\Delta\mathrm{t}$ is commonly used to calculate the transition matrix. In [125], the authors used a closed-form discretized propagation to calculate the transition matrix. In this thesis, a 4-th order Runge-Kutta numerical integration [155] of the IMU continuous-time kinematics model is applied to propagate the estimated INS state.

The 4-th order Runge-Kutta (sometimes also referred as RK4) is a classic method to approximate the ordinary differential equations. As for INS integration, the general solving process for position, velocity and attitude can be given by [155]:

$$\dot{x} = f(t, x), x(t_0) = x_0 \tag{3.59}$$

$$x(t + \Delta t) = x(t) + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{3.60}$$

where:

$$k_1 = f(t, x)$$

$$k_2 = f\left(t + \frac{\Delta t}{2}, x + k_1 * \frac{\Delta t}{2}\right)$$

$$k_3 = f(t + \frac{\Delta t}{2}, x + k_2 * \frac{\Delta t}{2})$$

$$k_4 = f(t + \Delta t, x + \Delta \mathrm{t} * k_3)$$

To propagate the uncertainty of the state, the discrete time state transition matrix of the linearized continuous dynamic model for error IMU state and discrete time noise covariance matrix need to be computed first:

$$\Phi_k = \Phi(t_{k+1}, t_k) = \exp\left(\int_{t_k}^{t_{k+1}} F(\tau)\, d\tau\right) \tag{3.61}$$

$$Q_k = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau)\, GQG\Phi(t_{k+1}, \tau)^T d\tau \tag{3.62}$$

When Δt is small, the transition matrix can be expanded as:

$$\Phi_k = \Phi(t_{k+1}, t_k) = \exp\left(\int_{t_k}^{t_{k+1}} F(\tau)\, d\tau\right) = \exp(F\Delta t) = I + F\Delta t + \frac{1}{2!}(F\Delta t)^2 + \frac{1}{3!}(F\Delta t)^3 + \cdots \quad (3.63)$$

The propagated IMU covariance matrix is:

$$P_{II_{k+1|k}} = \Phi_k P_{II_{k|k}} \Phi_k^T + Q_k \quad (3.64)$$

3.4.3 Full System Model

To include camera poses into state vector, MSCKF creates a sliding window of the $6N$ camera states and in total forms the $(15 + 6N)$ error states:

$$\tilde{x} = \begin{pmatrix} \tilde{x}_{INS}^T & \tilde{x}_{C_1}^T & \ldots & \tilde{x}_{C_N}^T \end{pmatrix}^T \quad (3.65)$$

where each camera error state is given as,

$$\tilde{x}_{C_i} = \begin{pmatrix} {}^{C_i}_{G}\tilde{\theta}^T & {}^{G}\tilde{p}_{C_i}^T \end{pmatrix}^T. \quad (3.66)$$

Meanwhile, the full covariance is formed as:

$$P_{k|k} = \begin{pmatrix} P_{II_{k|k}} & P_{IC_{k|k}} \\ P_{IC_{k|k}}^T & P_{CC_{k|k}} \end{pmatrix} \quad (3.67)$$

where P_{II} is the inertial state covariance matrix, P_{CC} is the camera state covariance matrix, P_{IC} is the correlation between the Initial state and camera state. The full noise propagation can be written as:

$$P_{k+1|k} = \begin{pmatrix} P_{II_{k+1|k}} & \Phi_k P_{IC_{k|k}} \\ P_{IC_{k|k}}^T \Phi_k^T & P_{CC_{k|k}} \end{pmatrix} \quad (3.68)$$

The filter is updated with two sources of information: vision and odometer measurements. When a new image arrives, six states are added to the filter for the image pose and are initialized to the current IMU pose:

$$
\begin{aligned}
{}^{C}_{G}\hat{q} &= {}^{C}_{I}\hat{q} \otimes {}^{I}_{G}\hat{q} \\
{}^{G}\hat{p}_{C} &= {}^{G}\hat{p}_{I} + C({}^{G}_{I}\hat{q})^{T}\ {}^{I}\hat{p}_{C}
\end{aligned}
\tag{3.69}
$$

And the covariance matrix is augmented as:

$$
P_{k|k} = \begin{pmatrix} I_{15+6(N+1)} \\ J \end{pmatrix} P_{k|k} \begin{pmatrix} I_{15+6(N+1)} & J^{T} \end{pmatrix}
\tag{3.70}
$$

With the Jacobian matrix J derived as:

$$
J = \begin{bmatrix} C({}^{C}_{I}\hat{q}) & 0_{3\times 9} & 0_{3\times 3} & 0_{3\times 6N} \\ -C({}^{I}_{G}\hat{q})^{T} \lfloor {}^{I}\hat{p}_{C} \times \rfloor & 0_{3\times 9} & I_{3} & 0_{3\times 6N} \end{bmatrix}
\tag{3.71}
$$

3.4.4 Camera Measurement Model

The development of the camera measurement model in [128] is reproduced in this section. The MSCKF proposes a novel approach to use 3D feature position to constrain all of the camera poses at which the measurements of that feature occurred. This is achieved without including the feature position in the filter state vector [128]. Considering a single feature f_i, that has been observed from a set of M_j camera poses $\left(q_{G}^{C_i}, p_{C_i}^{G}\right), i \in S_j$, each of the M_j observations of the feature are described by the model [128]:

$$
z_{i}^{(j)} = \frac{1}{Z_{j}^{C_i}} \begin{bmatrix} X_{j}^{C_i} \\ Y_{j}^{C_i} \end{bmatrix} + n_{i}^{(j)}
\tag{3.72}
$$

where $n_i^{(j)}$ is the image noise vector, with covariance $R_i^{(j)} = \sigma_{im}^2 I_2$. The feature position expressed in the camera frame is given by:

$$p_{f_j}^{C_i} = \begin{bmatrix} X_j^{C_i} \\ Y_j^{C_i} \\ Z_j^{C_i} \end{bmatrix} = C_G^{C_i}\left(p_{f_i}^G - p_{C_i}^G\right) \tag{3.73}$$

where $p_{f_i}^G$ is the 3D feature position in the global frame. If C_n is the camera frame in which the feature was observed for the first time, then the feature coordinates with respect to the camera at the i-th time instant are:

$$p_{f_j}^{C_i} = C_{C_n}^{C_i} p_{f_j}^{C_n} + p_{C_n}^{C_i} \tag{3.74}$$

In the above equation, the $C_{C_n}^{C_i}$ and $p_{C_n}^{C_i}$ are the rotation and translation between the camera frames at time instant n and i, respectively. The above equation can be rewritten as:

$$p_{f_j}^{C_i} = Z_j^{C_n}\left(C_{C_n}^{C_i} \begin{bmatrix} \frac{X_j^{C_n}}{Z_j^{C_n}} \\ \frac{Y_j^{C_n}}{Z_j^{C_n}} \\ 1 \end{bmatrix} + \frac{1}{Z_j^{C_n}} p_{C_n}^{C_i} \right) = Z_j^{C_n}\left(C_{C_n}^{C_i} \begin{bmatrix} \alpha_j \\ \beta_j \\ 1 \end{bmatrix} + \rho_j p_{C_n}^{C_i} \right) = Z_j^{C_n} \begin{bmatrix} h_{i1}(\alpha_j, \beta_j, \rho_j) \\ h_{i2}(\alpha_j, \beta_j, \rho_j) \\ h_{i3}(\alpha_j, \beta_j, \rho_j) \end{bmatrix} \tag{3.75}$$

with $\alpha_j = \frac{X_j^{C_n}}{Z_j^{C_n}}$, $\beta_j = \frac{Y_j^{C_n}}{Z_j^{C_n}}$ and $\rho_j = \frac{1}{Z_j^{C_n}}$.

Substituting the above equation into the measurement model gives:

$$z_i^{(j)} = \frac{1}{h_{i3}(\alpha_j, \beta_j, \rho_j)} \begin{bmatrix} h_{i1}(\alpha_j, \beta_j, \rho_j) \\ h_{i2}(\alpha_j, \beta_j, \rho_j) \end{bmatrix} + n_i^{(j)} \tag{3.76}$$

Then, the global feature position is computed by:

$$\hat{p}_{f_j}^G = \frac{1}{\hat{\rho}_j} {C_G^{C_n}}^T \begin{bmatrix} \hat{\alpha}_j \\ \hat{\beta}_j \\ 1 \end{bmatrix} + \hat{p}_{C_n}^G \tag{3.77}$$

Once the estimate of the feature position is obtained, the measurement residual can be computed as:

$$r_i^{(j)} = z_i^{(j)} - \hat{z}_i^{(j)} \qquad (3.78)$$

where $\hat{z}_i^{(j)} = \frac{1}{\hat{Z}_j^{C_i}} \begin{bmatrix} \hat{X}_j^{C_i} \\ \hat{Y}_j^{C_i} \end{bmatrix}, \begin{bmatrix} \hat{X}_j^{C_i} \\ \hat{Y}_j^{C_i} \\ \hat{Z}_j^{C_i} \end{bmatrix} = \hat{C}_G^{C_i}\left(\hat{p}_{f_i}^G - \hat{p}_{C_i}^G\right)$.

Linearizing the above measurement equation, the residual can be approximated as:

$$r_i^{(j)} \cong H_{X_i}^{(j)}\tilde{X} + H_{f_i}^{(j)}\tilde{p}_{f_i}^G + n_i^{(j)} \qquad (3.79)$$

$H_X^{(j)}$ and $H_{f_i}^{(j)}$ are the Jacobians of the measurements $z_i^{(j)}$ with respect to the state and the feature position, respectively. $\tilde{p}_{f_i}^G$ is the error in the position estimate of f_i.

By stacking the residuals of all M_j measurements of this feature, the residual vector is given:

$$r^{(j)} \cong H_X^{(j)}\tilde{X} + H_f^{(j)}\tilde{p}_{f_i}^G + n^{(j)} \qquad (3.80)$$

where the covariance matrix of $n^{(j)}$ is $R^{(j)} = \sigma_{im}^2 I_{2M_j}$.

Note that since the state estimate, X, is used to compute the feature position estimate, the error $\tilde{p}_{f_i}^G$ in is correlated with the errors $\tilde{X}$. Thus, the residual $r^{(j)}$ is not in the form of the above equation and cannot be directly applied for measurement updates in the EKF. To overcome this problem, [128] define a residual $r_0^{(j)}$, by projecting $r^{(j)}$ on the left null space of the matrix $H_f^{(j)}$. Specifically, if A denotes the unitary matrix whose columns form the basis of the left null space of H_f, then:

$$r_0^{(j)} = A^T\left(z^{(j)} - \hat{z}^{(j)}\right) \cong A^T H_X^{(j)}\tilde{X} + A^T n^{(j)} = H_0^{(j)}\tilde{X} + n_0^{(j)} \qquad (3.81)$$

Since the $2\mathrm{M_j} \times 3$ matrix $H_f^{(j)}$ has full column rank, its left null space is of dimension $2\mathrm{M_j} - 3$. Therefore, $r_0^{(j)}$ is a $(2\mathrm{M_j} - 3) \times 1$ vector. This residual is *independent* of the errors in the feature coordinates, and thus filter updates can be performed based on it. A *linearized* constraint between all the camera poses from which the feature f_j was observed. This expresses all the available information that the measurements $z_i^{(j)}$ provide for the M_j states, and thus the resulting EKF update is optimal, except for the inaccuracies caused by linearization.

3.4.5 Wheel Odometer Measurement Model

The wheel odometer observations and Non-Holonomic Constraints are defined in the vehicle motion frame $\{M\}$. The origin of $\{M\}$ is at the ground projection of the center point of the vehicle's rear wheel axle. The x axis is pointing forward, the y axis is pointing towards the left side of the vehicle and the z axis is perpendicular pointing up.

For wheel encoders, the forward speed can be calculated by:

$$\begin{cases} v_l = \dfrac{c_l}{r_{rate} * \Delta t} * \pi * d_l \\ v_r = \dfrac{c_r}{r_{rate} * \Delta t} * \pi * d_r \end{cases} \tag{3.82}$$

where, v_l and v_r are the velocities of left and right wheels, c_l and c_r are the left and right wheel encoder counts, r_{rate} is the encoder resolution, d_l and d_r are the diameters of the left and right wheels. The final forward velocity can be calculated as:

$$v = \frac{v_l + v_r}{2}. \tag{3.83}$$

Combing the wheel odometer forward velocity with the NHC pseudo-measurements, a 3×1 velocity measurement vector can be formed as:

$$
{}^{M}v_{veh} = {}^{M}\hat{v}_{veh} + {}^{M}\tilde{v}_{veh} = \begin{pmatrix} {}^{M}v_{odo} \\ 0 \\ 0 \end{pmatrix} \tag{3.84}
$$

The relationship between the IMU body frame velocity and vehicle motion frame velocity can be expressed as:

$$
{}^{I}v_{INS} = C_M^I \, {}^{M}v_{veh} - [\omega \times] r^I \tag{3.85}
$$

where the r^I is the lever-arm between the IMU body frame and vehicle motion frame, and C_M^I is the rotation matrix from the vehicle motion frame to the IMU body frame. We construct the misclosure vector:

$$
r_{odo} = {}^{M}v_{veh} - {}^{M}\hat{v}_{veh} = -C_G^M \delta \, {}^{G}v_I + C_G^M \left[{}^{G}v_I \times \right] {}_{G}^{I}\tilde{\theta} + n_{odo} \tag{3.86}
$$

The design matrix is written as:

$$
H_{odo} = \left[C_G^M \left[{}^{G}v_I \times \right] \quad 0_{3\times 3} \quad -C_G^M \quad 0_{3\times 3} \quad 0_{3\times 3} \quad 0_{3\times 3} \quad 0_{3\times 3} \right]. \tag{3.87}
$$

Chapter 4 SYSTEM VERIFICATION AND EXPERIMENTAL SETUP

With the theories of the wheel odometer aided MSCKF being covered in Chapter 3, this chapter will cover the experiment setup and system verification with the classic KITTI dataset. First, in Section 4.1, the datasets used in this thesis will be introduced in detail. Then, in Section 4.2, the related sensor calibration results will be presented. At last, in Section 4.3, the verification results will be presented.

4.1 Datasets Introduction

In this thesis, there will be three datasets used to test the proposed 15-state MSCKF implementation augmented with WO and NHC. First, the classic KITTI dataset [1] with simulated wheel velocity will be used to examine the correctness of the implementation. Then, the algorithm will be evaluated using an urban canyon dataset (KAIST Complex Urban Dataset [84]) and winter driving data collected in Calgary.

4.1.1 KITTI Dataset

The KITTI Dataset is one of the most popular existing public autonomous driving datasets. The KITTI dataset was recorded from a moving platform mounted on car while driving in and around Karlsruhe, Germany [1]. The duration of each sequence varies from 10 seconds to several minutes, and the driving scenarios contain city, residential, road, campus [1].

The sensor layout of the recording platform is shown in Figure 4-1. The platform includes camera images, laser scaners, high-precision GPS measurements and IMU measurements from an integrated GPS/IMU system. The corresponding sensor specifications are listed in Table 4-1. The dataset can be downloaded from [156].

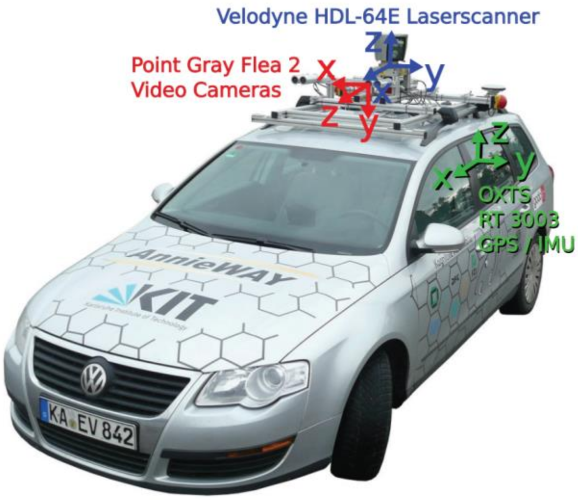


Figure 4-1: Recording Platform of KITTI Dataset from [1]

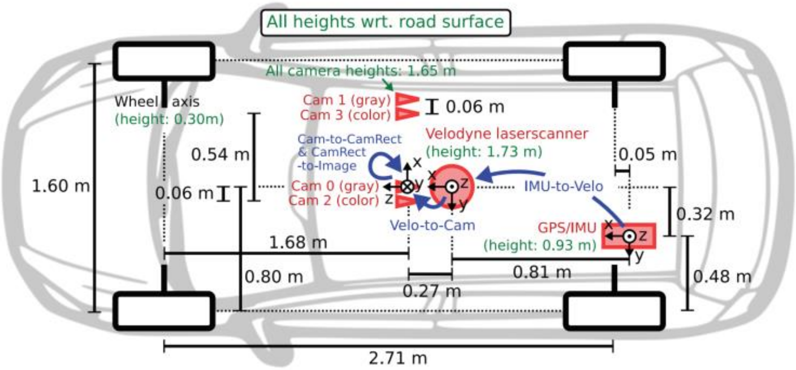


Figure 4-2: Sensor Setup of KITTI Dataset from [1]

Table 4-1: Sensor Specifications of KITTI Dataset after [1]

Sensor	Manufacturer	Model	Description	Number	Hz	Accuracy	Range
Stereo grayscale cameras	PointGray	FL2-14S3MC	1.4 Megapixels, 1/2” Sony ICX267 CCD, global shutter	2	10		
Stereo RGB cameras	PointGray	FL2-14S3C-C	1.4 Megapixels, 1/2” Sony ICX267 CCD, global shutter	2	10		
3D laser scanner	Velodyne	HDL-64E	64 beams, 0.09 degree angular resolution	1	10	2 cm	120 m
GPS/INS system	OXTS	RT3003	L1/L2 RTK	1	100	0.02 m + 0.1 degree	

4.1.2 KAIST Complex Urban Dataset

The KAIST Complex Urban Dataset is a dataset focused on driving environment perception and localization in challenging complex urban environments [84]. The dataset was collected in Korea with a vehicle equipped with stereo camera pair, 2d SICK LiDARs, 3d Velodyne LiDAR, Xsens IMU, fiber optic gyro (FoG), wheel encoders, and RTK GPS. The recording platform and sensor setup are shown in Figure 4-3, and the related sensor specifications are shown in Table 4-2. The dataset can be downloaded from [156]. In order to access the data, in this thesis, a data parser was developed. Due to the fact that the KAIST Complex Urban Dataset was collected in highly dynamic environment, a very challenging open research question is being able to handle dynamic objects seen from the cameras. In this thesis, the MSAC (M-Estimator RANSAC) is utilized to remove the effect brought by moving objects.

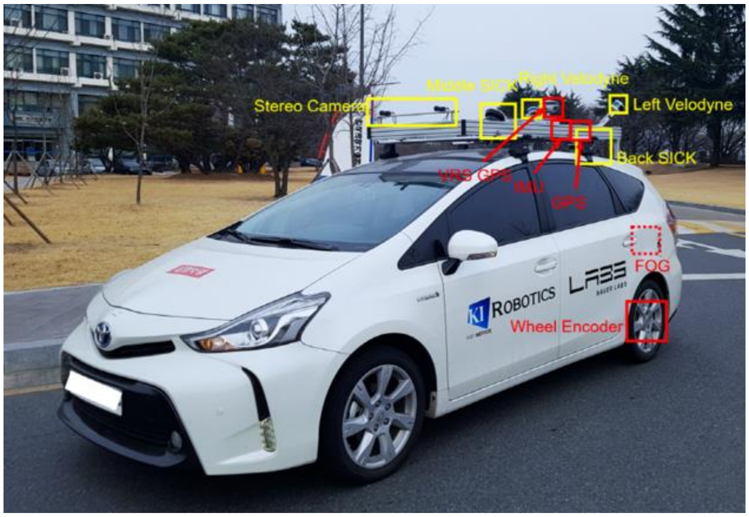


Figure 4-3: Recording Platform from [84]

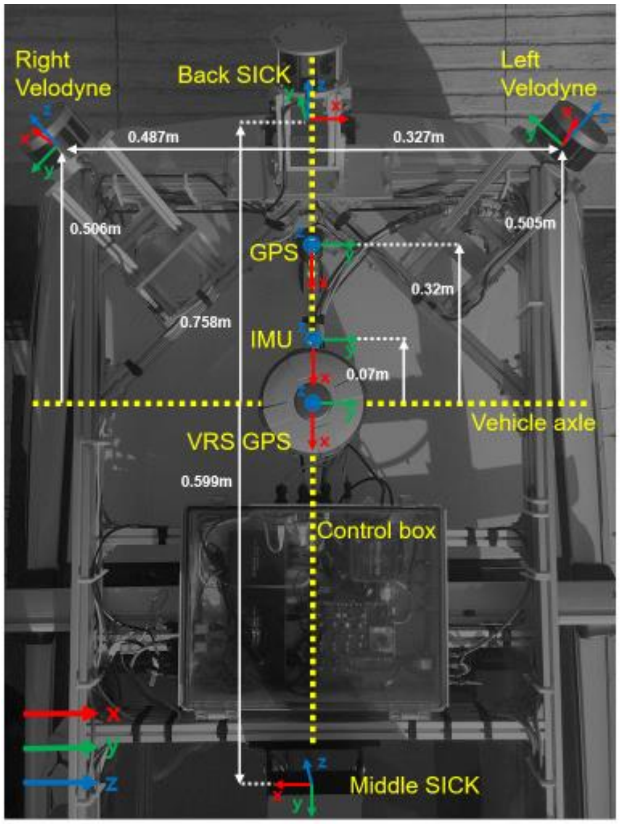


Figure 4-4: Sensor Setup of KAIST Complex Urban Dataset from [84]

Table 4-2: Sensor Specifications of KAIST Complex Urban Dataset after [84]

Sensor	Manufacturer	Model	Description	Number	Hz	Accuracy	Range
Stereo RGB cameras	PointGray	Flea3	1600x1200 color, 59 FPS	2	10		
IMU	Xsens	MTi-300	Enhanced AHRS Gyro	1	100	10°/h	
3-axis FOG	KVH	DSP-1760	Fiber Optics Gyro (3 axis)	1	1000	0.05°/h	
Wheel Encoder	RLS	LM13	Magnetic rotary encoder	2	100	4096 (resolution)	
GPS	U-Blox	EVK-7P	Consumer level GPS	1	10	2.5 m	
Altimeter	Withrobot	myPressure	Altimeter	1	10		
3D LiDAR	Velodyne	VLP-16	16 channel LiDAR, 360° FOV	2	10		100 m
2D LiDAR	SICK	LMS-511	1 channel LiDAR, 190° FOV	2	100		80 m

4.1.3 Calgary Winter Driving Dataset

To test the wheel odometer aided MSKCF's performance in winter driving environment, a data collection platform was built to collect driving data in winter Calgary, Canada. The data collection platform consists of stereo cameras, an automotive grade IMU, a vehicle odometer data logger, and a high-end GNSS/IMU system.

It should be noted that the wheel odometer data logger used in this system is not a wheel encoder as in KAIST Complex Urban Dataset. Most modern land vehicles are equipped with CAN-BUS, the resolution of the CAN-BUS output wheel speed is 1 km/h. Compared with the high-precision wheel encoder data in the KAIST Complex Urban dataset, the CAN-BUS wheel speed data is easier to access at the expense of a large quantization error.

The vision system used was originally developed by Bernhard Aumayer [143] and is the same hardware used in [157]. The system consists of two RGB cameras and a u-blox 6 receiver. The PPS signal output from the GPS receiver is used to ensure the shutter synchronization between the two cameras and provide time-tagged images.

A reference trajectory is obtained use a Novatel SPAN-LCI tightly-coupled RTK GNSS/INS solution generated by Inertial Explorer software. The base station for the RTK solution was set up at the rooftop of the Calgary Center for Innovation Technology (CCIT) building. The ground truth solution processing consists of a tightly-coupled RTK GNSS/INS forward and backward differential carrier phase post-processing. The overall accuracy is centimeter level in open-sky conditions and the several cm level during short travels in urban canyons.

Winter driving data was collected on 15 March 2020 for just over one hour in a mix of urban and suburban areas near the University of Calgary, Canada after a significant snowfall. Most of the route is suburban but there are some segments on the University of Calgary campus with significant urban canyon effects. A second data set, in summer driving conditions, was collected on 18 August 2020. The running trajectory of the Calgary Winter Driving Dataset is shown in Figure 4-6.

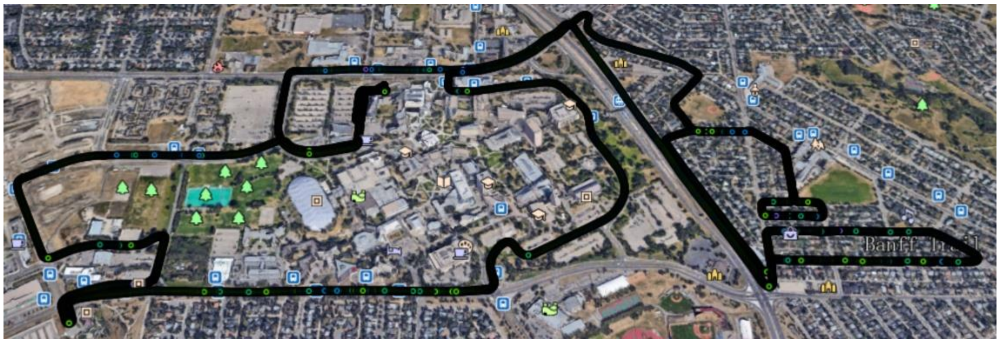


Figure 4-5: Running trajectory of Calgary Winter Driving Dataset (2020-03-15)

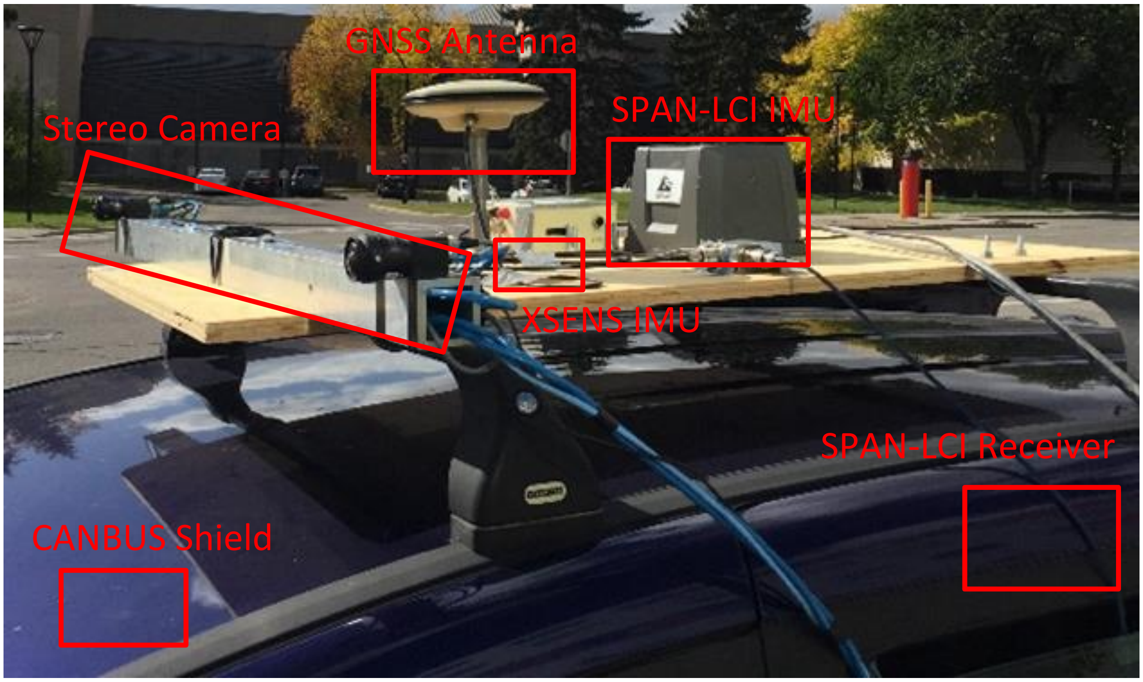


Figure 4-6: Recording Platform of the Calgary Winter Driving Dataset

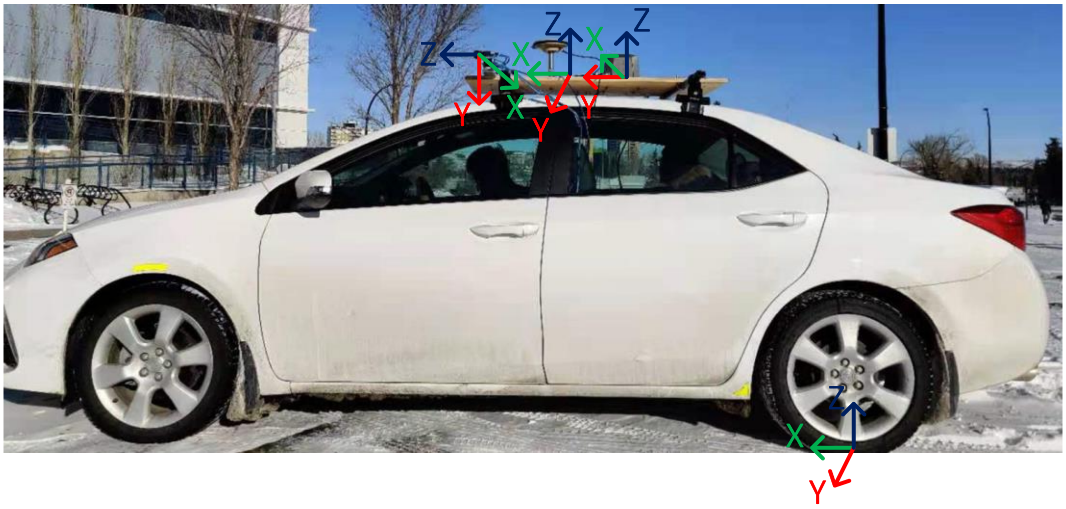


Figure 4-7: Sensor Setup of the Calgary Winter Driving Dataset

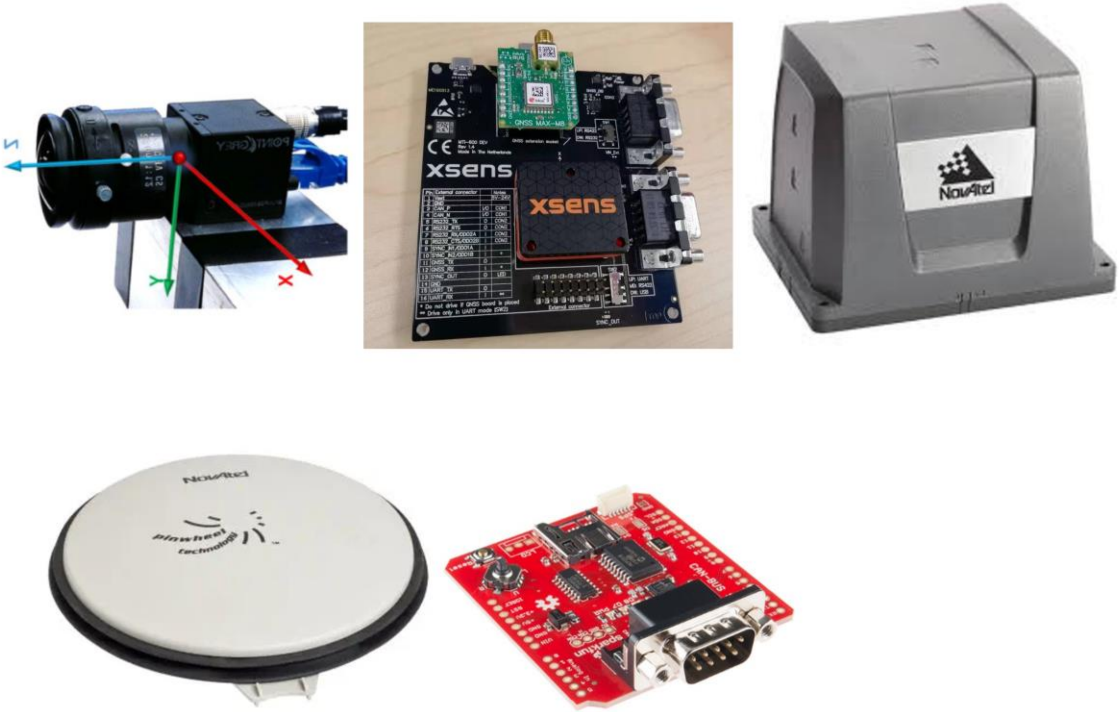


Figure 4-8: A Close-up look of the sensors: PointGray Camera, XSENS MTi-600 IMU, Novatel SPAN-LCI IMU, Novatel 702-gg Antenna, Sparkfun CANBUS Shield

Table 4-3: Sensor Specifications of the Data Collection Platform

Sensor	Manufacturer	Model	Description	Number	Hz	Accuracy
Stereo RGB cameras	PointGray	Blackfly-S GigE	1288 x 728 color, Global shutter	2	10	
IMU	Xsens	MTi-600	Enhanced AHRS Gyro	1	100	12°/h
GPS/INS	Novatel	SPAN-LCI	FOG Gyros + MEMS Accelerometers	1	100	0.06m + <1°/h
Wheel Odometer	Sparkfun	CAN-BUS Shield	Vehicle wheel odometer data logger	1	10	1 km/h (resolution)

After introducing the datasets used in this thesis, in the next section, relevant sensor and algorithm verifications are described.

4.2 Sensor Calibration

As has been described in the previous chapters, sensor calibration in of vital importance for the multi-sensor integration systems. In this section, the methodologies and results of the related camera intrinsic calibration, IMU calibration, wheel odometer scale factor calibration and camera-IMU calibration discussed in detail.

4.2.1 Camera Intrinsic Calibration

The first step of any vision-based navigation system is always to determine the camera intrinsic parameters. The Intrinsic Orientation Parameters (IOPs) will be used to rectify the image measurements. In this thesis, the camera calibration process was conducted using the “Camera Calibration” toolbox in MATLAB. The calibration was done in outdoor environment before the data collection. The checkerboard used for calibration is a 11×11 checkerboard with cell size of 50 mm. During the calibration, images containing the full body of the checkerboard were collected at different orientations and locations with respect to the camera (as shown in Figure 4-11).

The images used for calibration should:

1. Does not have motion blur effect;

2. Calibration board can be viewed in all areas of the image;

3. Camera is in focus;

4. Calibration board can be seen from different orientations, distances and locations.

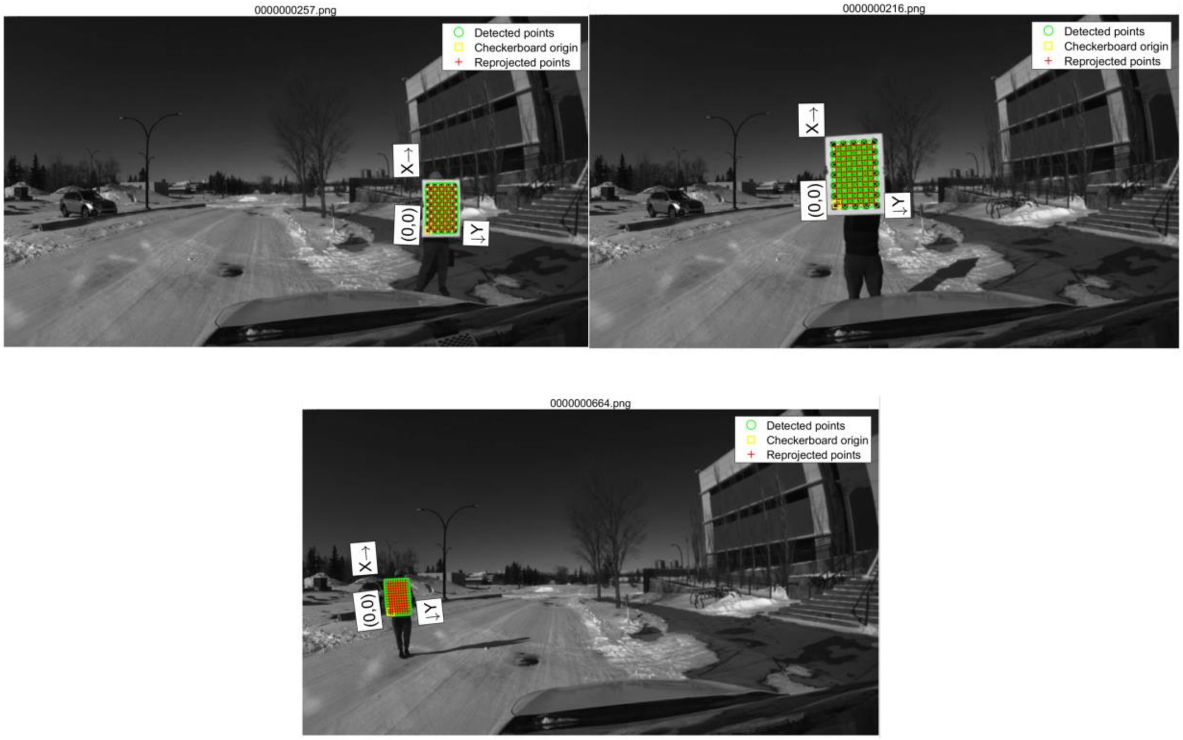


Figure 4-9: Camera Calibration Process

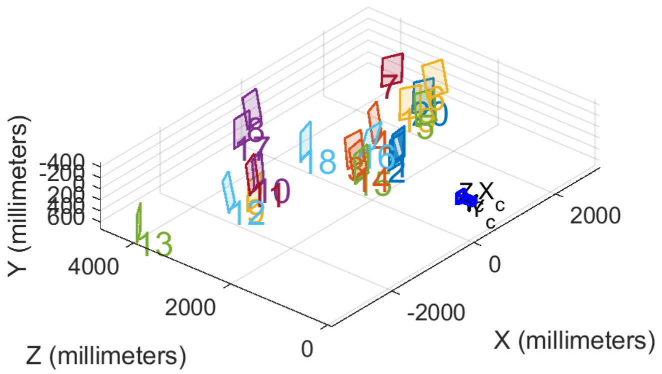


Figure 4-10: Checkerboard Locations with respect to Camera for the Camera Calibration

In this case, 20 images were recorded, the final calibration results can be seen in Table 4-4, and the corresponding reprojection errors are shown in Figure 4-12. A reprojection error is the distance between a pattern keypoint detected in a calibration image, and a corresponding world point projected into the same image [158]. If the overall mean reprojection error is too high, consider excluding the images with the highest error and recalibrating. Generally speaking, the overall mean error should be less than pixel to be accepted.

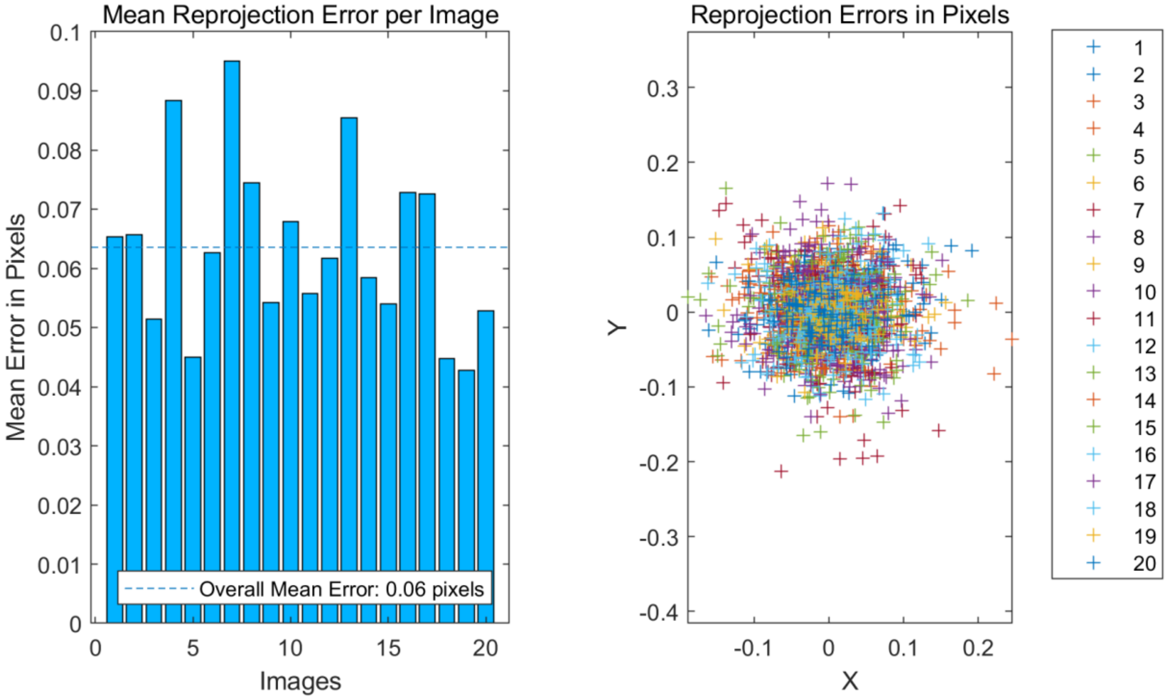


Figure 4-11: Reprojection Error in Camera Calibration

Table 4-4: Camera Intrinsic Parameters

Camera Intrinsic Parameters	Value (pixel)
Focal Length $(f_x \quad f_y)$	$(543.0091 \quad 543.0796)$
Principal Point $(c_x \quad c_y)$	$(593.8106 \quad 369.3547)$
Image Size	$(728 \quad 1288)$
Radial Distortions	$(-0.2866 \quad 0.0915)$
Tangential Distortions	$(1.7771 * 10^{-4} \quad -5.1903 * 10^{-4})$

Another very important step is to use the radial distortion parameters and tangential distortion parameters to undistort the image before feature extraction and matching.

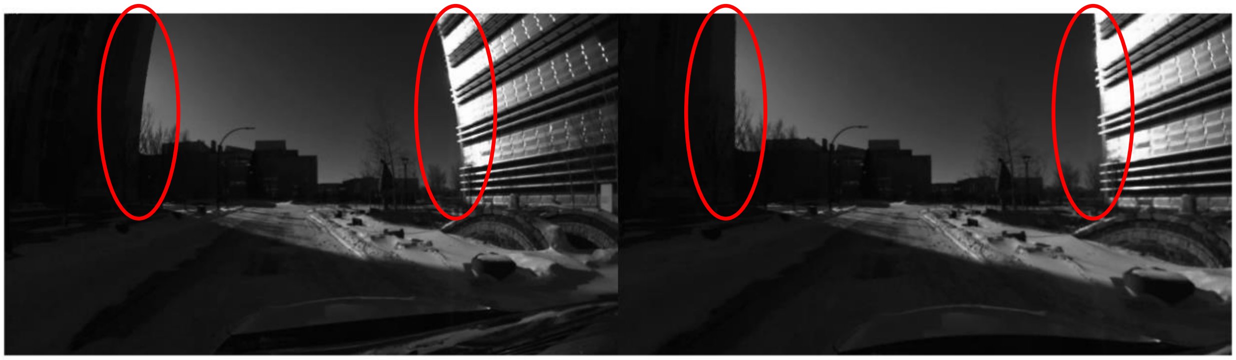


Figure 4-12: An Example of Calgary Winter Driving Dataset. Left: the original image. Right: after rectification

From Figure 4-13, the left image shows that the due to the distortions, the contours of the building seem to be “bended”. However, after the rectification, the contours of the buildings come back the “straight lines”.

4.2.2 IMU Intrinsic Calibration

As has been reviewed in Chapter 2 & 3, the IMU measurements from gyroscopes and accelerometers contain errors from instrument bias, scale factor, non-orthogonality (misalignment), and most importantly, sensor noise. The IMU intrinsic calibration can be defined as a process of comparing the output with the known reference information to determine these parameters. The common calibration methods include: Local Level-Frame (LLF) calibration, six-position static test and angular rate test [159]. The LLF calibration and angular rate test require multi-axis turntable, thus, in this thesis, the six-position test is utilized to determine the initial bias for the Xsens MTi-600 IMU and SPAN-LCI IMU. For the record, the KITTI Dataset and the KAIST Complex Urban Dataset do not provide IMU intrinsic parameters, so a “trial and error” approach is used to determine the best-fit initial biases for the IMUs.

The basic idea of the six-position static test is to mount the IMU on a level table with each sensitive axis pointing alternately up and down (six positions). The advantage of this method is its simplicity, and the disadvantage is that non-orthogonality cannot be determined [159]. With each axis being pointing up and down, the corresponding biases can be calculated as (take z-axis for instance):

$$\begin{cases} b_{a,z} = \dfrac{x_{up} + x_{down}}{2} \\ b_{w,z} = \dfrac{\omega_x(up) + \omega_x(down)}{2} \end{cases} \tag{4.1}$$

The measurements taken for each position is 5 minutes. Due to the fact that the Xsens MTi-600 IMU cannot be placed upside down horizontally, the six-position method's result does not yield the correct bias. The SPAN-LCI IMU initial biases are determined as:

$$\begin{cases} b_{a,initial} = [0.0083 \quad 0.0153 \quad 0.0119]^T \left(\dfrac{m}{s^2}\right) \\ b_{g,initial} = [0.0012 \quad -0.0016 \quad -0.0010]^T \left(\dfrac{deg}{s}\right) \end{cases} \tag{4.2}$$

Another very important aspect of IMU intrinsic calibration is to determine the measurement noise parameters. More specifically, to determine $[\sigma_g \quad \sigma_a \quad \sigma_{b_g} \quad \sigma_{b_a}]$. The values for σ_{b_g} and σ_{b_a} are normally included in the IMU datasheet as either "angular random walk" or "velocity random walk". In [160], the authors maintain an open source project which estimates IMU noise parameters by computing Allan variance from static IMU observations. Unfortunately, due to the time limit of this project, the calibration of IMU intrinsic parameters are not done properly, for the following results and comparison, the values from trial and error in Table 4-5 are used.

Table 4-5: IMU Noise Parameters of The SPAN-LCI IMU

Noise Parameter	Value (for SPAN-LCI IMU)	Unit
Gyroscope “white noise” σ_g	0.001	$\frac{rad}{s}\frac{1}{\sqrt{Hz}}$
Accelerometer “white noise” σ_a	0.01	$\frac{m}{s^2}\frac{1}{\sqrt{Hz}}$
Gyroscope “random walk” σ_{b_g}	0.001	$\frac{rad}{s^2}\frac{1}{\sqrt{Hz}}$
Accelerometer “random walk” σ_{b_a}	0.0005	$\frac{m}{s^3}\frac{1}{\sqrt{Hz}}$

4.2.3 Wheel Odometer Calibration

Similar to the IMU measurements, the wheel odometer measurement also contains: scale factor, bias, misalignment and measurement noise. For the simplicity of the overall measurement model, most research only consider the scale factor when utilizing the wheel odometer measurements [82]. In this thesis, for the KITTI Dataset, the simulated wheel odometer measurements are used to verify the implementation, so the calibration process is not considered. For the KAIST Complex Urban Dataset, the dataset provides calibrated wheel encoder parameters using high-precision GPS and FOG sensors [2]. The wheel odometer parameters are given by:

$$w = (d_l \quad d_r \quad w_b) \tag{4.3}$$

where: d_l and d_r represent the left and right rear wheel diameters, and w_b means the wheel base between the two rear wheels. To construct the relative measurement of the vehicle using the GPS and FOG, the 2D pose consisting location and orientation is given by [2]:

$$\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}), \boldsymbol{x_i} = (x_i, y_i, \theta_i)) \tag{4.4}$$

The wheel encoder parameters are estimated through the forward motion kinematics:

$$k_i(w) = \begin{bmatrix} \Delta \mathrm{x_i} \\ \Delta y_i \\ \Delta \theta_i \end{bmatrix} = \begin{bmatrix} l_a \cos(\Delta\theta_i) \\ l_a \sin(\Delta\theta_i) \\ \frac{l_{diff}}{w_b} \end{bmatrix} \tag{4.5}$$

$$l_a = \frac{\frac{c_l}{4096}\pi d_l + \frac{c_r}{4096}\pi d_r}{2} \tag{4.6}$$

$$l_{diff} = \frac{c_l}{4096}\pi d_l + \frac{c_r}{4096}\pi d_r \tag{4.7}$$

where: l_a is the average distance, l_{diff} is the difference distance between the left and right rear wheels, 4096 is the wheel encoder resolution, c_l and c_r are the wheel encoder counts for each wheel.

And the objective function of this optimization problem is described as [2]:

$$w^* = \arg\min_w \sum_i \|z_i \ominus k_i(w)\|_{\Omega_\mathrm{i}} \tag{4.8}$$

where $\ominus$ stands for the inverse motion operator, Ω_i represents the uncertainty of the GPS and FOG.

For our Calgary Winter Driving Dataset, the wheel odometer speed with the SPAN-LCI output forward speed are plotted in Figure 4-14. From Figure 4-14, it can be noted that the CAN-BUS wheel speed aligns with the ground truth speed. This will also be discussed in Chapter 5. Thus, for the simplicity of our estimator, the scale factor of the wheel speed is not taken into consideration.

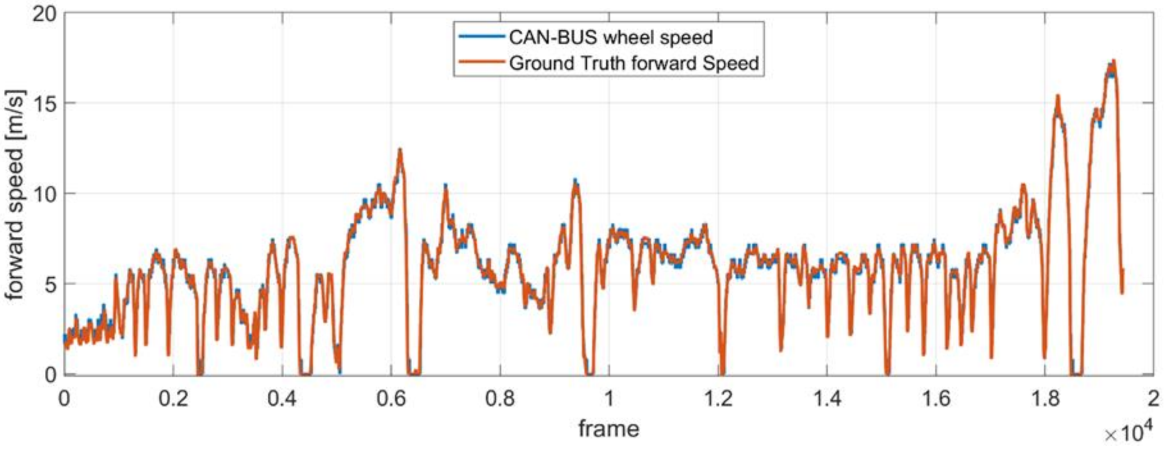


Figure 4-13: CAN-BUS wheel speed vs. Ground truth forward speed from the Calgary Winter Driving Dataset

In addition, due to the fact that wheel odometer and NHC measurement are both recorded in the vehicle frame, the lever-arm offset and misalignment between the vehicle frame and the IMU body frame might also degenerate the overall accuracy of the integration system. In [65], the author first proved that the lever-arm offset and misalignment errors will degenerate the performance of INS/WO/NHC integration results. The author proposed an online calibration method which include an additional 6×1 lever-arm offset (3), boresight errors (2), and scale factor into the state vector to constrain the corresponding errors. In this thesis, due to the time limit of the project, the main objective is to prove the feasibility of integrating INS, VO and WO, thus this calibration method is not implemented. However, this is very important for the optimal performance, which is why it is included in Chapter 6 future works recommendations.

4.2.4 Camera-IMU Calibration

One of the biggest challenges of utilizing visual-inertial sensing system is how to accurately calibrate the transform between IMU and camera. Both the KITTI Dataset and the KAIST

Complex Urban Dataset use highly accurate 3D LiDAR systems as the medium to bridge the IMU and the camera. More specifically, the KITTI Dataset register the 3D LiDAR with respect to the reference camera coordinate system (the left camera of the stereo camera) by initializing the rigid body transformation [1]. Then, the error function calculated from the Euclidean distance of 50 manually selected correspondence is built for optimization [1]. As for the KAIST Complex Urban Dataset, the extrinsic parameters are calculated by projecting the global point clouds that were reconstructed through the vehicle path onto each image. Similar to the approach of the KITTI Dataset, the extrinsic parameters are also estimated by minimize a optimization problem [2].

For our Calgary Winter Driving Dataset, due to the absence of the accurate 3D LiDAR data, the transformation between the IMU and camera has to be directly estimated.

A simple and intuitive calibration method was first conducted for camera-IMU calibration:

- The car is parked at a fixed point with a fixed orientation, and the checkerboard is placed at multiple fixed known positions.
- Use the images focusing on the checkerboards to calibrate the intrinsic parameters of the camera.
- Since the relative position of the board to the vehicle body is known, the position of the camera to the vehicle body can be inferred.
- At the same time, the transformation between the vehicle and the IMU can be obtained during the installation of the IMU, so the position of the camera to the IMU can also be obtained.

The disadvantage of this method is that the transformation between the vehicle body frame to the IMU body frame still needs to be tape-measured. Thus, the overall calibration accuracy degenerates.

To improve the camera-IMU calibration accuracy, the Kalibr calibration toolbox is utilized here. The Kalibr calibration toolbox [161] is an open source project that aims to solve the following calibration problems:

- Multiple camera calibration: intrinsic and extrinsic calibration of a camera-systems with non-globally shared overlapping fields of view.
- Camera-IMU calibration: spatial and temporal calibration of an IMU w.r.t a camera-system.

This toolbox must be used under ubuntu system with support of ROS (Robot Operating System). In this thesis, Kalibr calibration toolbox is used to calibrate the camera-IMU transformation matrix. To use this toolbox, it is important to minimize the motion blur in the camera while also ensuring that you excite all axes of the IMU. One needs to have at least one translational motion along with two degrees of orientation change for these calibration parameters to be observable [161].

The calibration data was collected in an indoor lab environment (Figure 4-15). According to the recommendation of the project page, the calibration board was an Aprilgrid 6x6 0.8x0.8 m (A0 page). In total, 2855 images and about 4 minutes 44 seconds IMU data were collected. It should be noted that in order to use the Kalibr calibration toolbox, the data that to be converted to the ROS bag format. The timestamp in the ROS bag format is the 19-digits nanosecond [161]. After collected the data, the ROS bag can be created with the Kalibr function by running the command:*“kalibr_bagcreater --folder dataset-dir --output-bag x.bag”*.

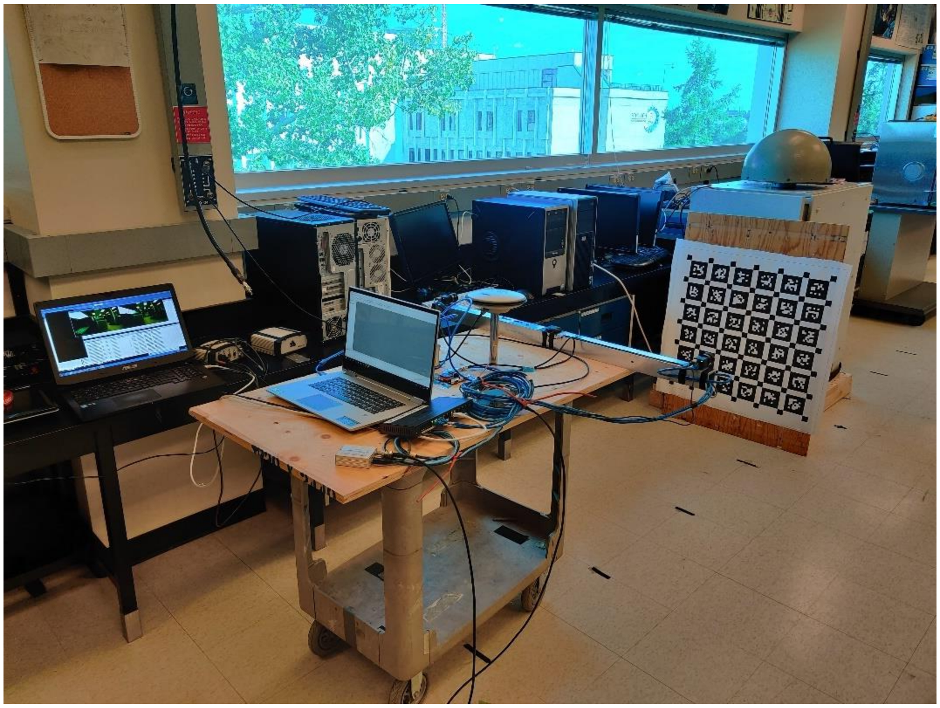


Figure 4-14: Camera-IMU Calibration Experiment Setup

After collected the calibration data, the calibration can be executed by:

- Use the function “*kalibr_calibrate_imu_camera*”;
- Input the static calibration file which will have the camera topics in it;
- Make an “*imu.yaml*” file with the corresponding noise parameters;
- Execute the calibration.

The final results are stored in the “*camchain-imucam.yaml*” file:

```
cam0:
  T_cam_imu:
  - [-0.12877581419264783, -0.968321101668008, -
0.2139416596726299, 0.17908882006069335]
  - [-0.5182924714776103, 0.24964659780319412, -0.8179544548536455, -
0.6218344539905737]
  - [0.8454523663037764, 0.005551600647467514, -0.534021980859964, -
3.468015324857509]
  - [0.0, 0.0, 0.0, 1.0]
  cam_overlaps: []
  camera_model: pinhole
  distortion_coeffs: [-0.3506651907097312, 0.111827012633611, -0.00393521731047083,
    0.03103925759556292]
  distortion_model: radtan
  intrinsics: [578.7038030040761, 581.3393979849152, 511.73540354347483, 407.33762931478185]
  resolution: [1288, 728]
  rostopic: /cam0/image_raw
```

Together with the transformation matrix T_{imu}^{cam} being estimated, a report containing the relevant statistics are also generated.

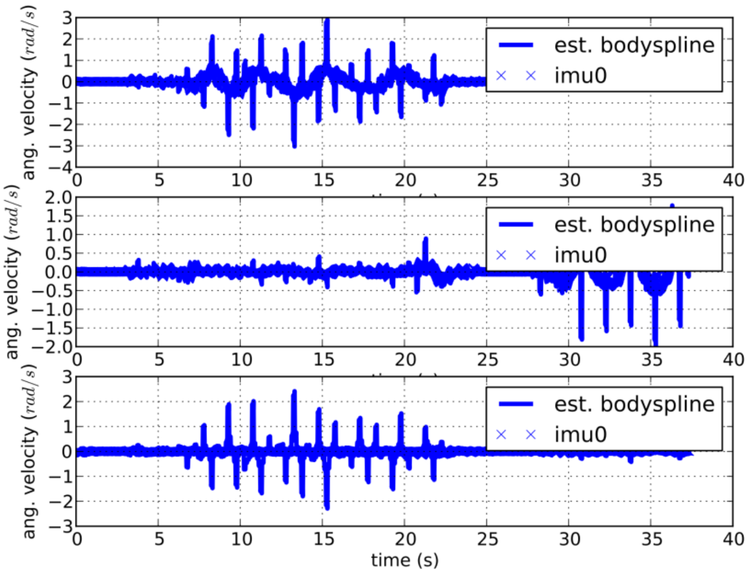


Figure 4-15: Comparison of Predicted and Measured Angular Velocities (body frame)

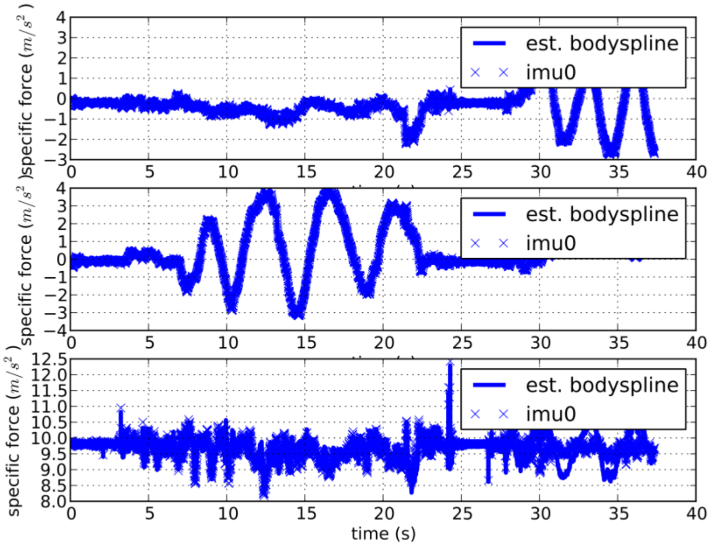


Figure 4-16:Comparison of Predicted and measured Specific Force (IMU frame)

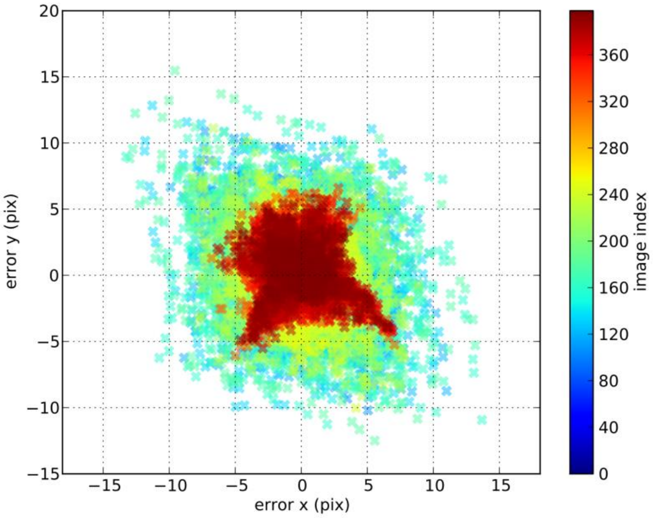


Figure 4-17: Camera Reprojection Error of the Camera-IMU Calibration

From Figure 4-16 and Figure 4-17, it can noted that the predicted acceleration and angular velocities fit the IMU measurements pretty well. However, from Figure 4-18, the camera reprojection error seems to be too large (> 5 pixels). The reasons behind this could be: (1) Inaccurate IMU noise parameters; (2) The board with all the equipment mounted is very heavy, which makes it very hard to control to make slow and smooth movement to excite all axes of the IMU.

In addition, the transformation between camera and IMU can also be included in the state vector to be estimated online. The relevant work can be found at [153].

4.3 Verifications Results with the KITTI Dataset

After calibrating the sensors, the popular KITTI Dataset is used to numerically examine the correctness of the implementation. Due to the fact that the KITTI Dataset does not provide wheel odometer measurements, we use IMU body frame velocity from the OXTS output as wheel odometer measurements. In order to imitate the real-world driving velocity, we assign large measurement noise to the derived IMU body frame velocity. In this, and each subsequent test, the filter is initialized with the position, velocity, and orientation of the corresponding reference trajectory to simulate a transition from complete GNSS availability to complete outage.

In this section, the KITTI Dataset sequence 0095 (city scene, 27 seconds) and sequence 0117 (city scene, 66 seconds) are used.

Figure 4-18: Sample Image from KITTI Dataset 0095

Figure 4-19:Sample Image from KITTI Dataset 0117

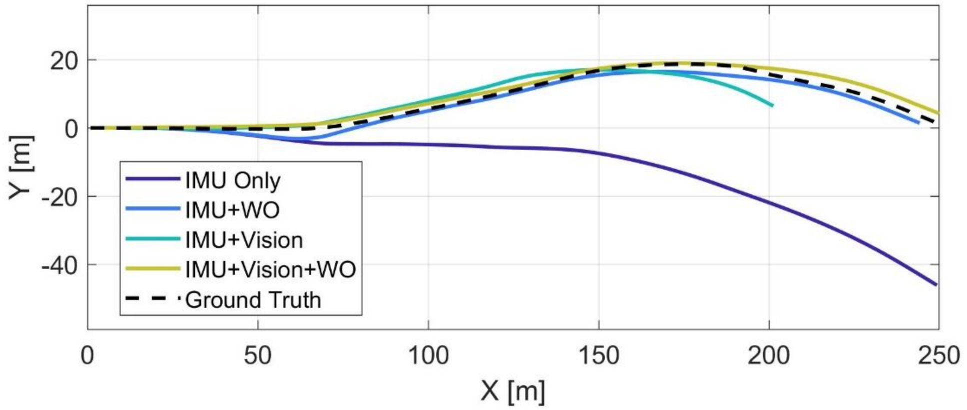


Figure 4-20: Trajectories of using IMU propagation only, IMU+WO, IMU+Vision, IMU+Vision+WO on the KITTI 0095

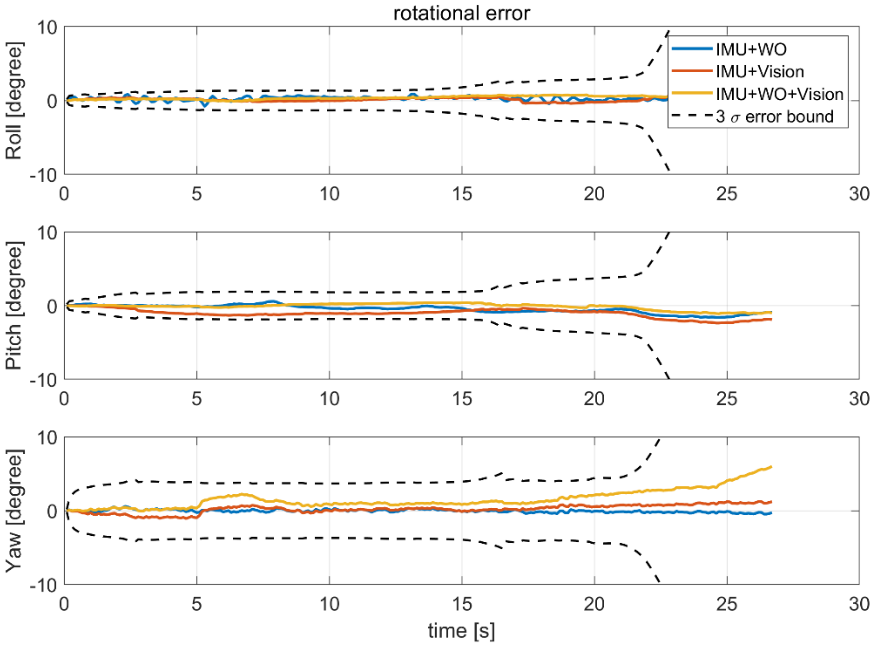


Figure 4-21: The Rotational Errors (with 3 sigma error bound) of using IMU+WO, IMU+Vision, IMU+Vision+WO on the KITTI 0095

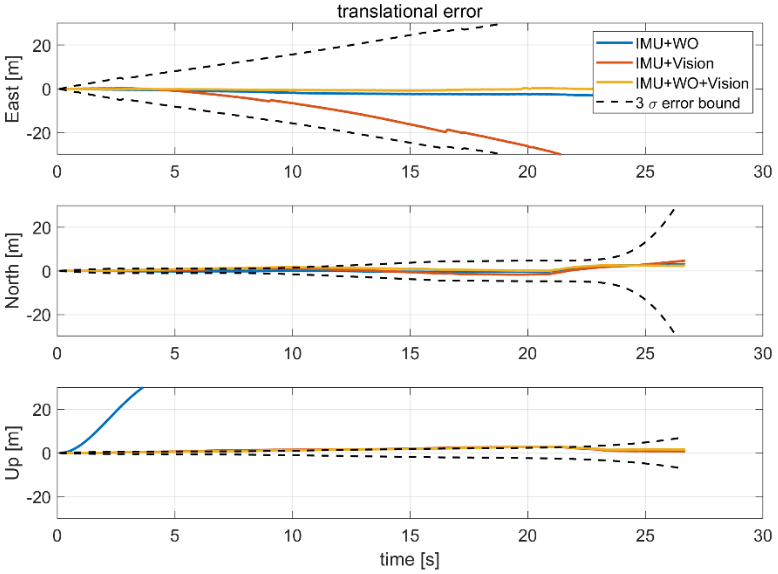


Figure 4-22: The Translational Errors (with 3 sigma error bound) of using IMU+WO, IMU+Vision, IMU+Vision+WO on the KITTI 0095

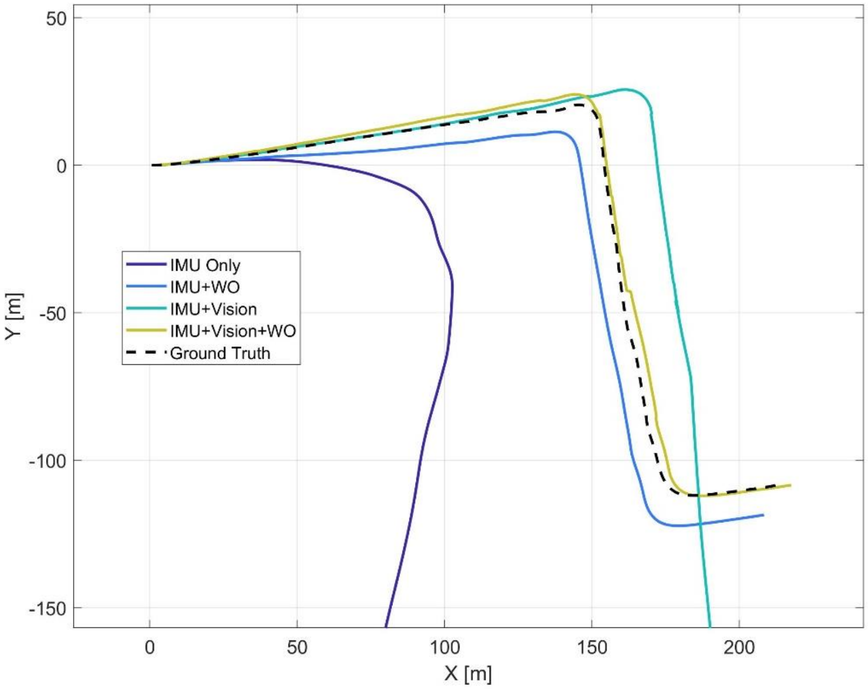


Figure 4-23: Trajectories of using IMU propagation only, IMU+WO, IMU+Vision, IMU+Vision+WO on the KITTI 0117

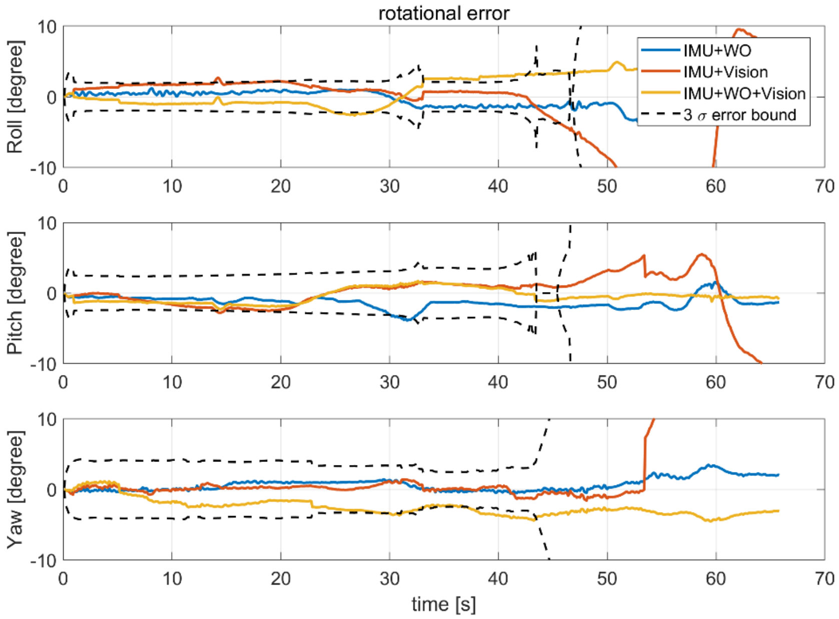


Figure 4-24: The Rotational Errors (with 3 sigma error bound) of using IMU+WO, IMU+Vision, IMU+Vision+WO on the KITTI 0117

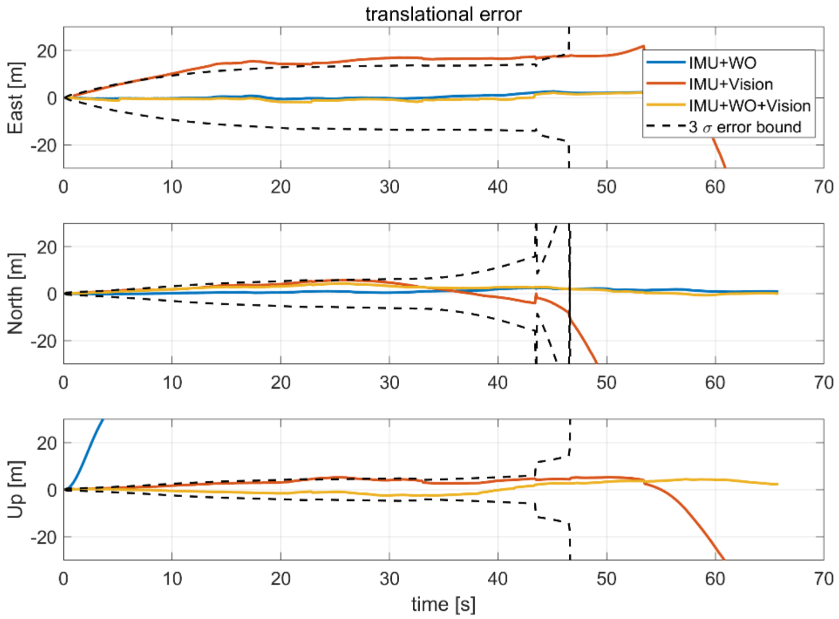


Figure 4-25: The Translational Errors (with 3 sigma error bound) of using IMU+WO, IMU+Vision, IMU+Vision+WO on the KITTI 0117

Figure 4-21 and Figure 4-24 plot the trajectories obtained for the KITTI 0095 and 0017, respectively. Figure 2-22 and 2-23 show the rotational and translational errors with 3σ error bound for the results of the KITTI 0095, respectively. Figure 2-25 and 2-26 show the rotational and translational errors with 3σ error bound for the results of the KITTI 0117, respectively. From the results from KITTI 0095 and 0117, it is clearly shown that when there is no source of update, the IMU integration results quickly diverge. However, when the wheel odometer measurement and NHC are integrated into the system, the velocity and orientation drifts are controlled. When IMU is combined with monocular vision, we can see from the second trajectory (Figure 4-24) that the heading angle and position drift are controlled before the first turn. However, when the IMU position drift becomes too large, the monocular vision cannot provide enough constraints to contain the drift. This is due to the fact that a single camera does not observe the absolute scale, and the filter must rely on the IMU for scale information. When the drift of IMU integration results becomes too large to ignore, the bearing correction provided by the monocular vision cannot control the forward motion degeneracy. When IMU, wheel odometer and monocular vision are integrated together, the wheel odometer can provide the correct scale information and monocular vision can control the orientation. The detailed RMSE results for all data sets are listed in the Table 4-6.

Table 4-6: Average Root Mean Square Error (ARMSE) of IMU Only, IMU+WO, IMU+Vision and IMU+WO+Vision on KITTI Dataset traverses 0095 and 0117

Dataset		KITTI 0095	KITTI 0117
Duration		27 (s)	66 (s)
IMU Only	Horizontal ARMSE (m)	13.475	158.784
	Rot. ARMSE (deg)	0.417	2.319
	Final Horizontal Pos Error (m)	47.392	731.258
IMU+WO	Horizontal ARMSE (m)	3.194	6.491
	Rot. ARMSE (deg)	0.417	2.321
	Final Horizontal Pos Error (m)	5.468	11.507
IMU+Vision	Horizontal ARMSE (m)	11.401	50.892

	Rot. ARMSE (deg)	0.669	2.042
	Final Horizontal Pos Error (m)	48.854	518.086
IMU+WO+Vision	Horizontal ARMSE (m)	0.844	1.868
	Rot. ARMSE (deg)	0.970	2.084
	Final Horizontal Pos Error (m)	2.476	4.195

From the above discussion, the numerical correctness of the implementation is verified by the KITTI Dataset with simulated wheel odometer data. In the following chapter, the algorithm is going to be tested on the complex urban environments and winter driving environments.

Chapter 5 RESUTLS AND ANALYSES

After validating the implementation using the KITTI Dataset, in Section 5.1, the performances of the wheel odometer aided VIO will be first evaluated in urban canyon environments (Seoul City) using the KAIST Complex Urban Dataset. In Section 5.2, the performances of the algorithm will be tested on the Calgary Winter Driving Dataset to reveal the effect of winter driving environment on different sensors. For the results, “WO” represents “WO+NHC”.

5.1 Performances in Urban Canyon Environments

As has been reviewed in Chapter 2, performing localization and navigation tasks in the urban canyon environment can be very challenging. On the one hand, the limited satellite visibility and the multipath effect will degrade the GNSS performance in the urban canyon environment. On the other hand, due to the complex and dynamic scenes (moving objects) in the urban environment, the feature extraction and tracking process can be challenging. In this section, “trajectory urban 39” of the KAIST Complex Urban Dataset is utilized to evaluate the performance of the proposed system and whether it can serve as an alternative information source for land vehicle navigation in complex urban canyon environment.

This section of the data is a very good representation of the daily driving environment in urban canyons, as shown in Figure 5-1, these scenes include moving vehicles and moving pedestrians.

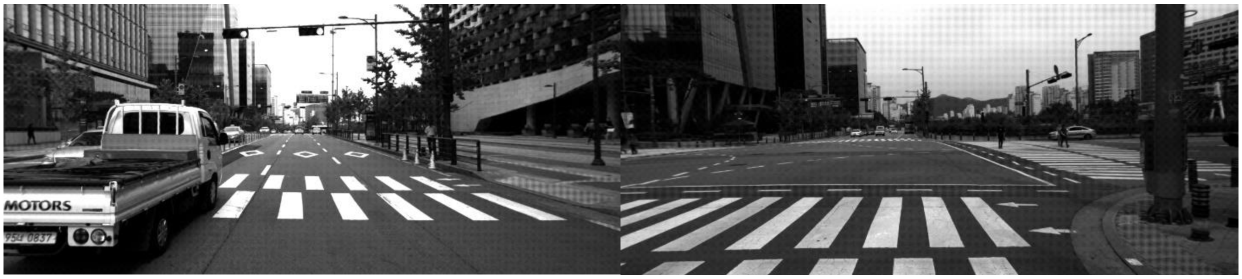


Figure 5-1: Sample Images from the “trajectory urban 39” of the KAIST Complex Urban Dataset

The trajectory is shown in Figure 5-2. The detailed RMSE results are listed in Table 5-1. Similarly, the IMU integration results start to drift away first, the monocular vision helps control the heading angle, but due to the lack of correct scale information, the trajectory still has a large drift after the turn. By incorporating the wheel encoder measurements, both IMU+WO and IMU+WO+Vision keep the correct scale. In addition, due to the fact that the wheel speed data comes from the high-resolution wheel encoders in the KAIST Complex Urban Dataset, the IMU+WO outperforms the IMU+Vision solution. This proves wheel odometer can be a very reliable information source in the normal driving environments for land vehicle navigation.

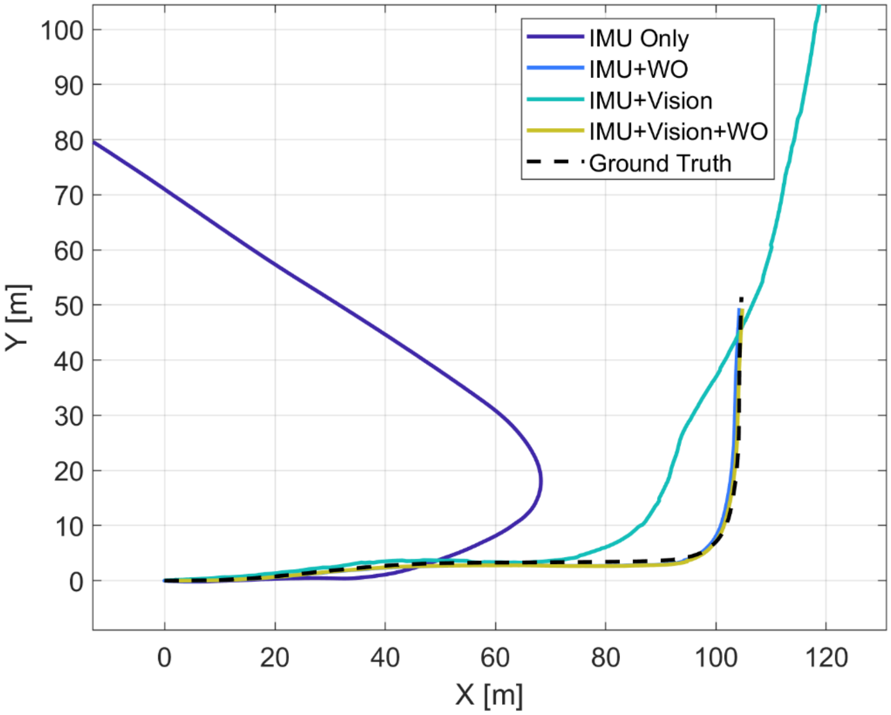


Figure 5-2: Trajectories of using IMU propagation only, IMU+WO, IMU+Vision, IMU+Vision+WO on KAIST Complex Urban Dataset (trajectory urban 39), respectively.

However, compared to the KITTI 0117 results, the IMU+Vision results do not maintain the along-track accuracy and starts to drift before the turn. By plotting the orientation and position estimate, we can see that before the turn happens, there is a wrong update from the monocular vision that makes the orientation estimate jump from the correct value to a clearly incorrect value (shown in Figure 5-3).

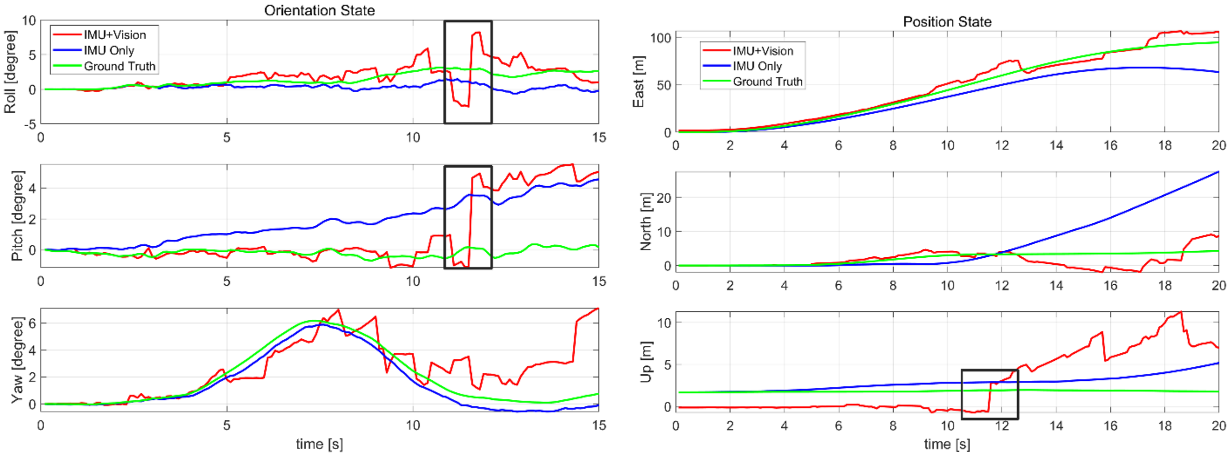


Figure 5-3: Orientation and Position State of the Filter Before the Turn in KAIST Complex Urban Dataset (trajectory urban 39)

To make sense of the reason behind this wrong update, all the estimated feature-to-vehicle distances are plotted in Figure 5-4, where we can clearly see that some of the estimated feature distances are larger than 1000 meters. Generally speaking, the camera perception range is around 200~300 meters. Some of the detected features are also obviously points on other moving vehicles. Feature-based VO algorithms are based on the premise that the observed environment is static. For the implementation of feature tracking in this thesis, the outliers are excluded by the MSAC algorithm, which can be considered as an robust version of the regular RANSAC algorithm. During the feature tracking results of this section of trajectory, it can be observed that most of the features located on the moving objects are excluded by the MSAC algorithm. However, it is shown in the Figure 5-5 that by only relying on RANSAC-like algorithms are not enough to cope with the high dynamic scenes in urban environments. When a feature is moving while also being tracked in view, the relative motion might be constant or even increasing which is not accounted for in the model. That is a main reason why navigating using vision in highly dynamic environments still remains a extremly challenging problem for land vechicle navigation. The possible and promising solution to this problem is to make the algorithm understand the surrondings to fully eliminate the influence brought by dynamic objects. Currently, relevant research on how to leverage the semantic segmentation techniques to improve the performances of visual localization systems has become a very popular research topic [162].

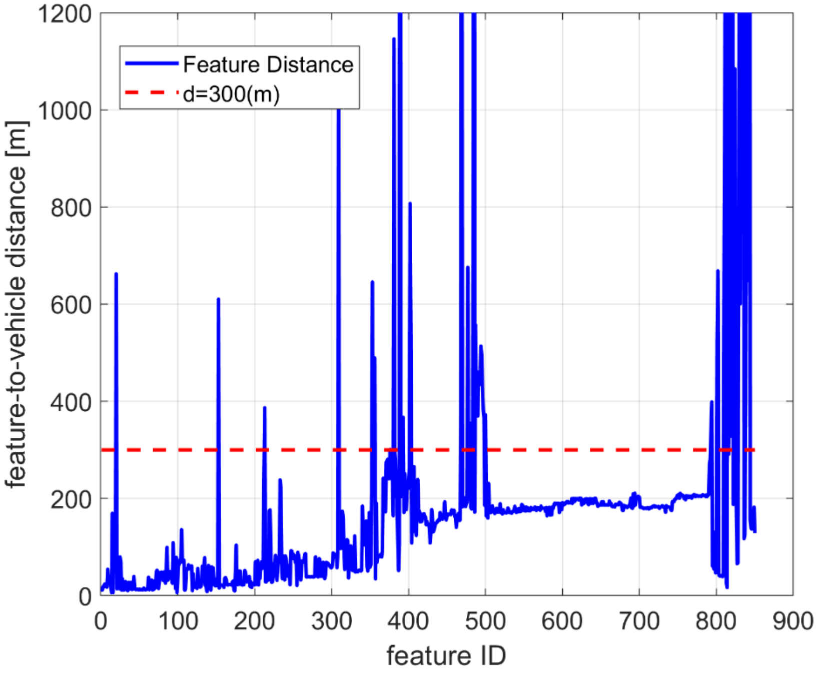


Figure 5-4: The Estimated feature-to-vehicle Distance

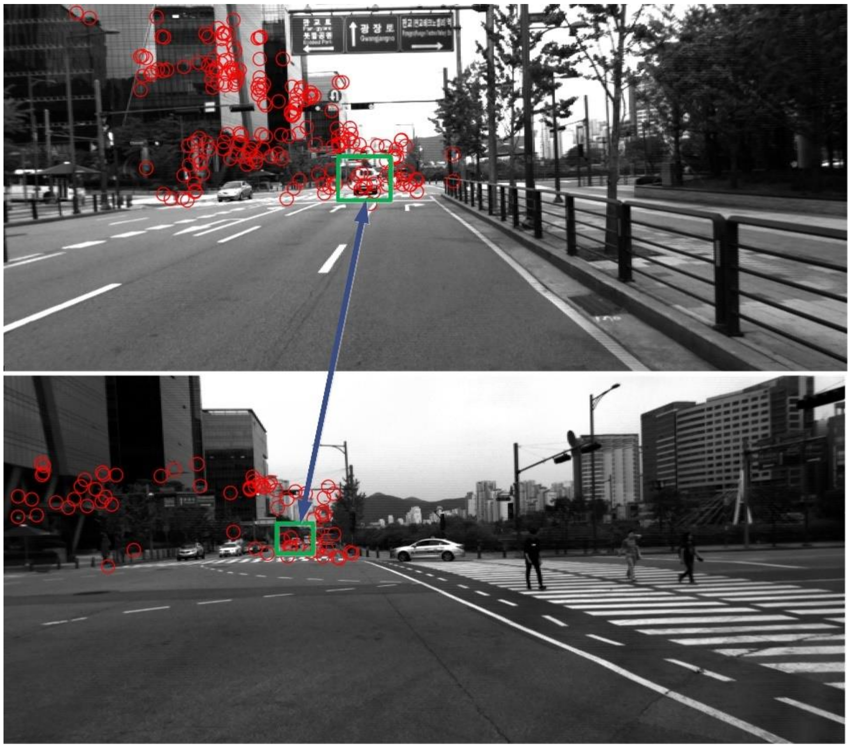


Figure 5-5: Feature Tracking of a Moving Vehicle Before and During the Turn

Table 5-1: Average Root Mean Square Error (ARMSE) of IMU Only, IMU+WO, IMU+Vision and IMU+WO+Vision of KAIST Complex Urban Dataset (trajectory urban 39)

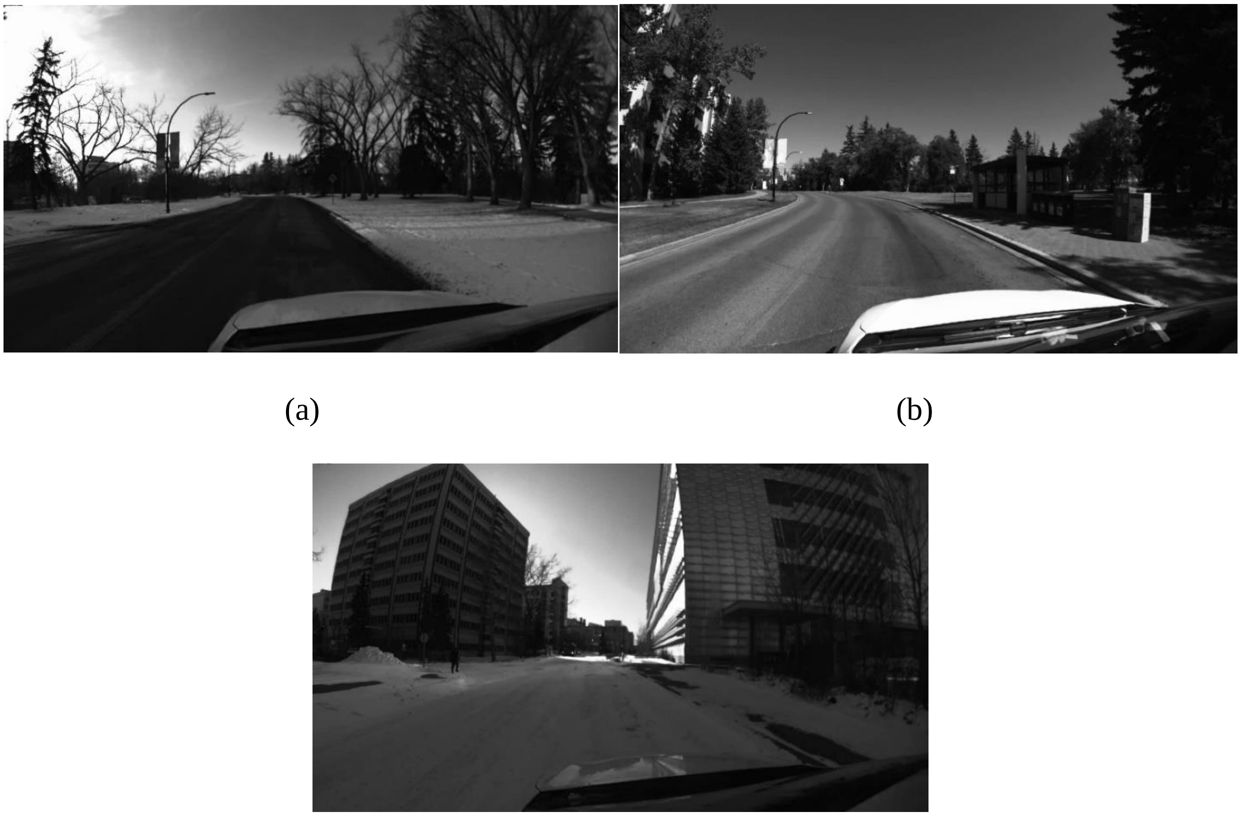
Dataset		KAIST urban 39
Duration		55 (s)
IMU Only	Horizontal ARMSE (m)	46.577
	Rot. ARMSE (deg)	3.253
	Final Horizontal Pos Error (m)	198.047
IMU+WO	Horizontal ARMSE (m)	1.809
	Rot. ARMSE (deg)	3.253
	Final Horizontal Pos Error (m)	2.054
IMU+Vision	Horizontal ARMSE (m)	28.024
	Rot. ARMSE (deg)	4.565
	Final Horizontal Pos Error (m)	150.672
IMU+WO+Vision	Horizontal ARMSE (m)	0.845
	Rot. ARMSE (deg)	3.752
	Final Horizontal Pos Error (m)	1.814

5.2 Performances in Winter Driving Environments

After investigating the proposed IMU+WO+Vision algorithm's performance in complex urban canyon environment, the important question to be asked is how will winter driving conditions affect the accuracy of the solution. To evaluate the proposed algorithm's performance in winter driving environment, two segments of the Calgary Winter Driving Dataset are chosen for comparison. The first winter-1 (66 s) consists of driving data collected in mixed suburban and urban areas where the roads was half covered by snow. The second winter-2 (62 s) involves driving data collected in urban areas where roads was fully covered by snow. In addition, summer-1 (149 s) represents the driving data collected in clean road condition with the similar running trajectory to winter-1 in order to compare winter and summer conditions. Sample images of the driving data are shown in Figure 5-6.

Due to the fact that the camera-IMU extrinsic calibration was not conducted with high accuracy in this thesis, the SPAN-LCI IMU will be used to evaluate the proposed algorithm's performance in Section 5.2.1 in order to verify whether camera and wheel odometer can be served as reliable information source in winter urban environments.

The trajectories of winter-1, summer-1 and winter-2 are shown in Figure 5-7, Figure 5-8 and Figure 5-9. The detailed RMSE results are listed in Table 5-2. Sample images from winter-1, winter-2 and summer-1 are listed in Figure 5-1, it can be noted that Figure 5-1 (a) and (b) captured the same scene in winter and summer.



(a)

(b)

(c)

Figure 5-6: Sample Images from Calgary Driving Dataset, winter-1 (a), summer-1 (b) and winter-2 (c)

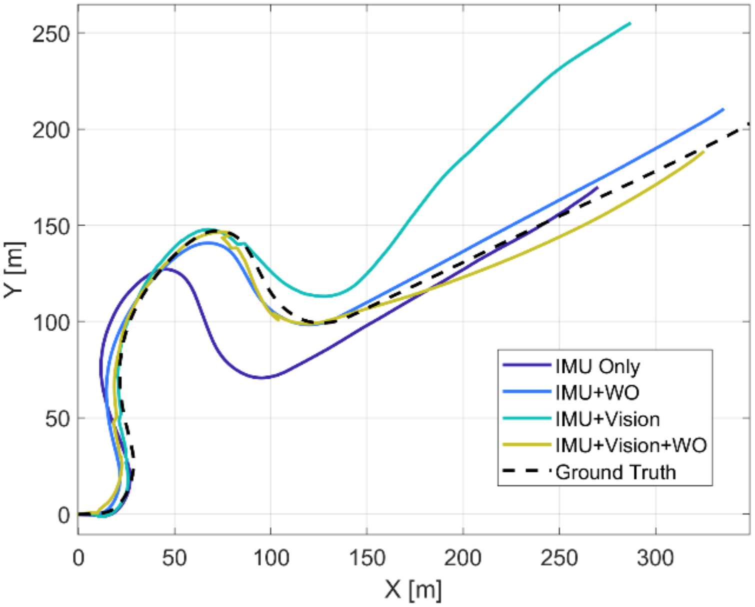


Figure 5-7: Trajectories of using IMU propagation only, IMU+WO, IMU+Vision, IMU+Vision+WO on Calgary Winter Driving Dataset (winter-1)

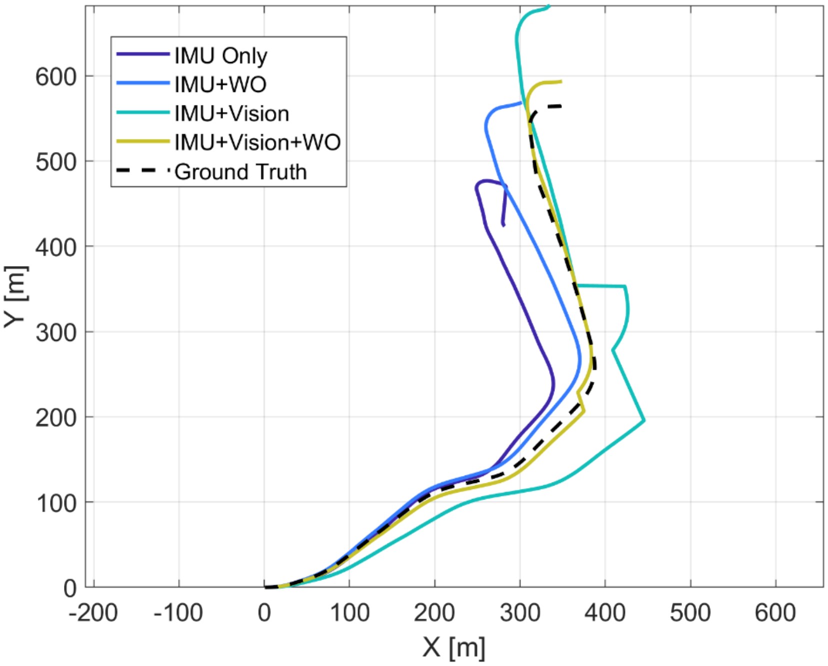


Figure 5-8: Trajectories of using IMU propagation only, IMU+WO, IMU+Vision, IMU+Vision+WO on Calgary Winter Driving Dataset (summer-1)

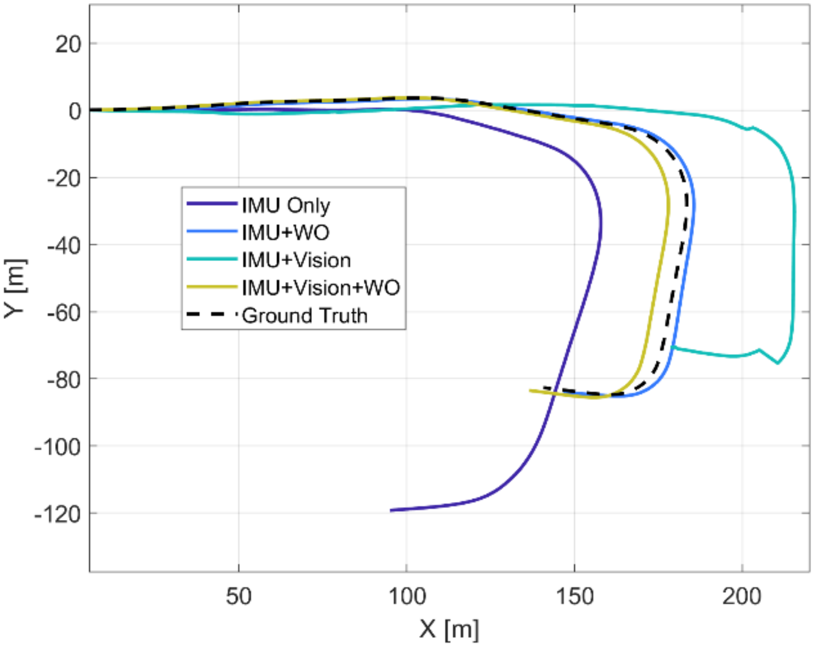


Figure 5-9: Trajectories of using IMU propagation only, IMU+WO, IMU+Vision, IMU+Vision+WO on Calgary Winter Driving Dataset (winter-2)

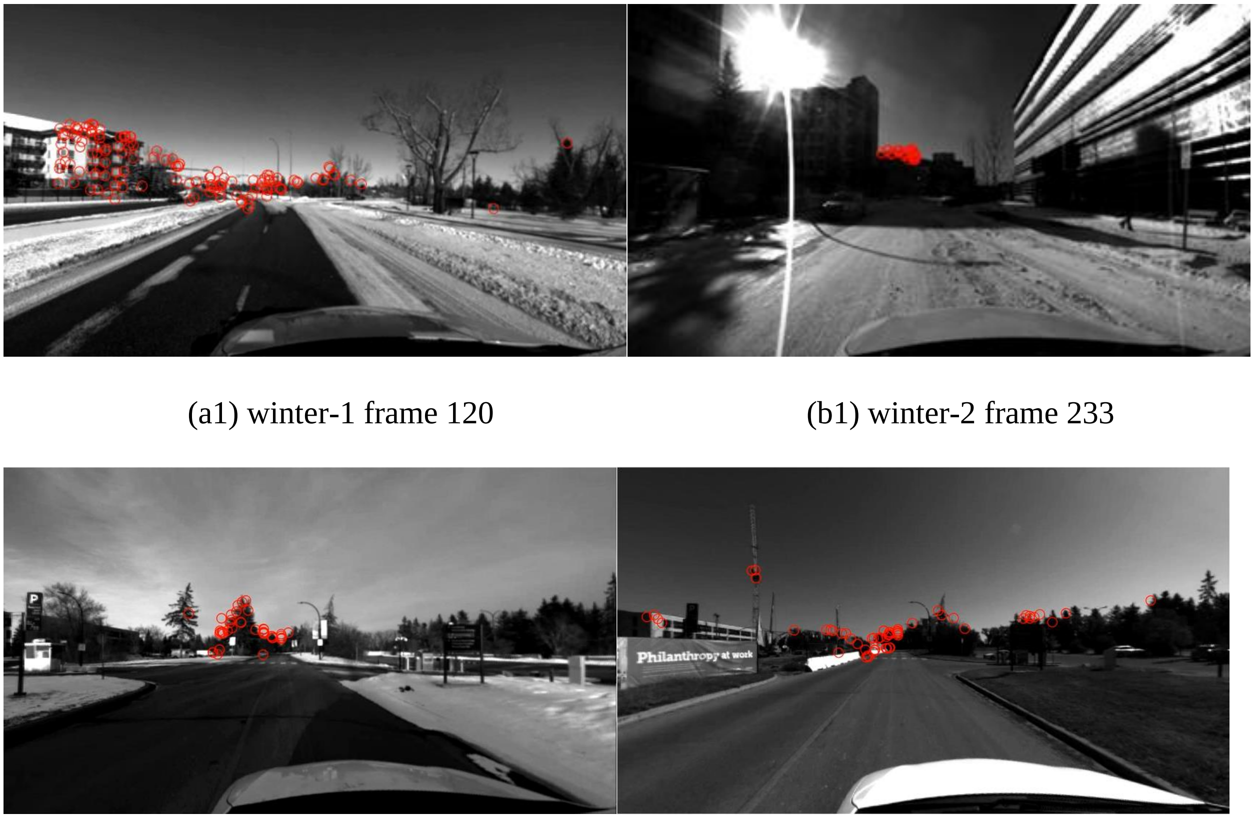
From Figure 5-7, Figure 5-8 and Figure 5-9, it can be noted that the high-end IMU reduces the scale drift significantly. However, due to the fact that the accurate extrinsic parameters between IMU body frame and camera frame is absent, thus, the rotational error of IMU+Vision and IMU+Vision+WO accumulate faster than expected. From the results from winter-1, we can see that by incorporating the vision information, the IMU+Vision and IMU+Vision+WO results align with the ground truth trajectory closely. However, after the turn, the IMU+Vision result starts to diverge. Including the CANBUS wheel speed information controls the scale very well. Compared with the IMU results, the horizontal accuracy improved 74.78% and 89.90% for IMU+WO and IMU+Vision+WO, respectively. IMU+Vision+WO achieved 19.649 m and 3.456 m horizontal position errors in two 1-minute drives in our Calgary winter urban environment. The results demonstrates that by incorporating WO, NHC and VO, the wheel odometer aided VIO is a complementary dead-reckoning approach that is able to function in winter driving environment.

We expected the winter driving conditions to result in a lack of salient features for the vision system and increased slippage and skidding of vehicle affecting the wheel speed. Given these two concerns, the camera and wheel odometer performances will be discussed in detail.

(1) Camera

In Figure 5-11, multiple scenes from winter-1. summer-1 and winter-2 are compared side by side, the red circles represent the SIFT features that are extracted and being tracked in the corresponding frames. Generally, the imaging sensor performance can be affected by moving objects, lighting condition, shadow and the texture of the environment [79]. In the snow-covered environment, the feature extraction might be affected by the low-texture environment. In Figure 5-11 (a1), it can be noted that most of the features are located on buildings, trees and road marks.

However, in Figure 5-11 (b1), there are no features on the snow-covered road. Also, the shadows in Figure 5-11 (b1) obscure many potential features. This shows that by using camera as the only navigation sensor in winter environment might not be a feasible solution. In Figure 5-11 (a2-4 ,b2-4), the corresponding scenes from winter-1 and summer-1 are compared side by side. From these comparisons, it shows that in most cases, there are more salient feature points in summer than in winter. However, an environment half-covered by snow will not lead to a sharp decrease in the number of feature points.

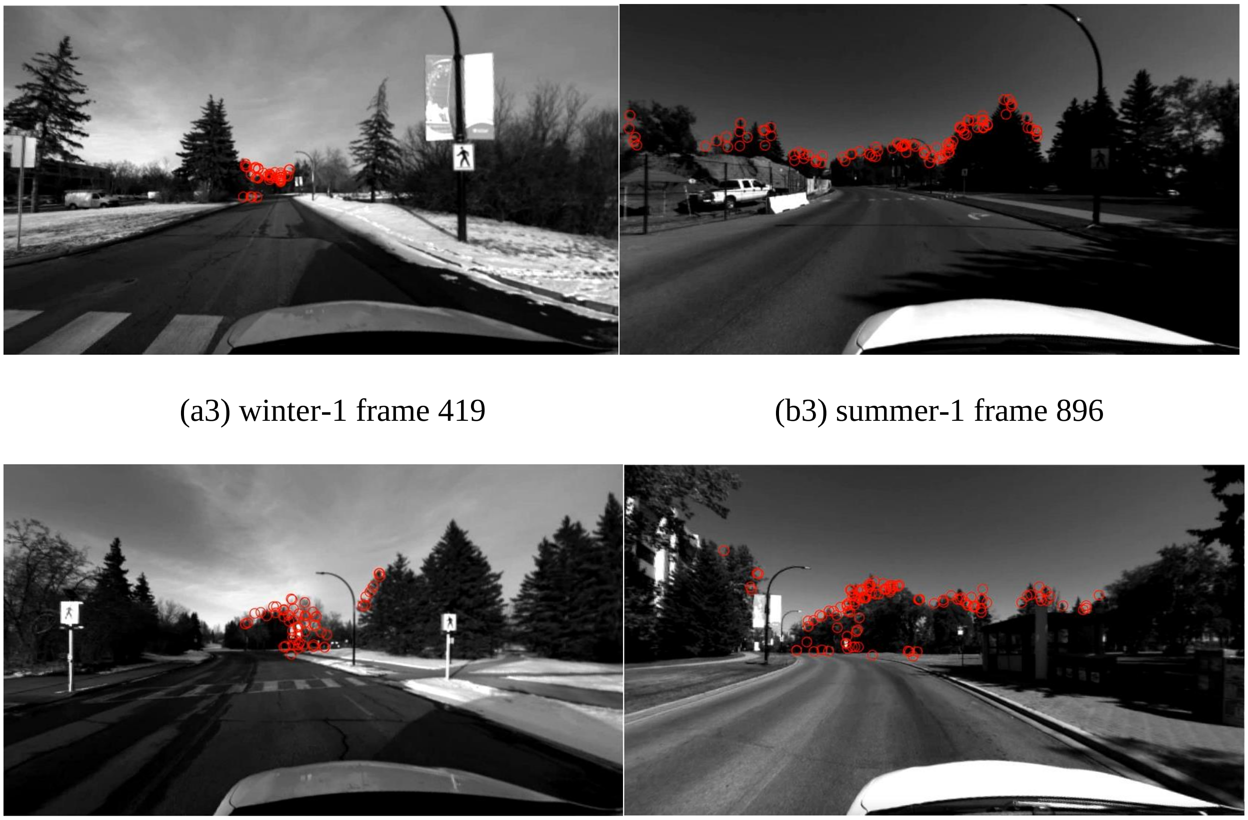


(a1) winter-1 frame 120

(b1) winter-2 frame 233

(a2) winter-1 frame 210

(b2) summer-1 frame 453



(a3) winter-1 frame 419

(b3) summer-1 frame 896

(a4) winter-1 frame 710

(b4) summer-1 frame 1491

Figure 5-10: Scenes from Calgary Winter Driving Dataset (winter-1, summer-1 and winter-2)

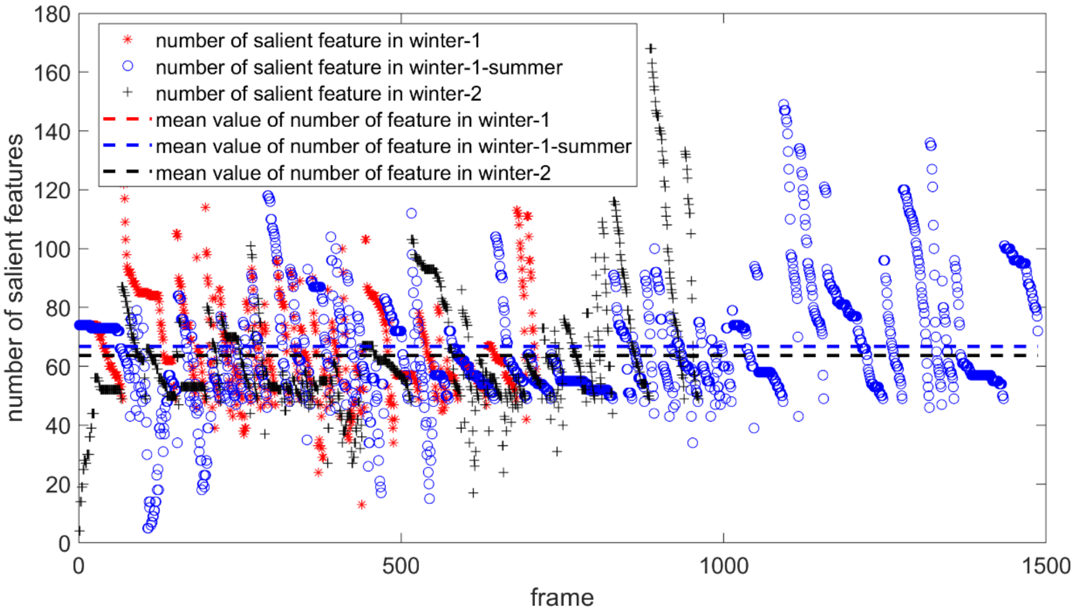


Figure 5-11: Number of Salient Features Per Frame in winter-1, summer-1 and winter-2

In Figure 5-10, the number of tracked salient features per frame for each dataset are plotted. From this figure, considering winter-1 and summer-1 have similar running trajectories in winter and summer road conditions, the number of detected features do not show a significant increase. In addition, note that both test segments (winter-1 and winter-2) have sufficient of features (> 50) for VIO purposes and these two cases demonstrate the feasibility of using a camera as a navigation sensor in a winter urban and suburban environments.

(2) Wheel odometer

To evaluate the impact of snowy road condition on the wheel odometer performance, we compare the wheel odometer speed with the SPAN-LCI output forward speed. The differences between the two velocities are calculated.

$$\Delta v = abs\left(v_{wo} - v_{forward}\right) \tag{5.1}$$

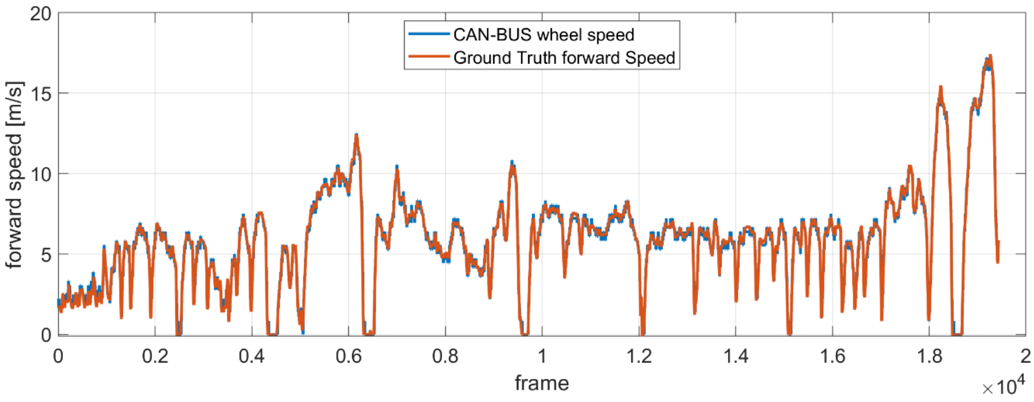


Figure 5-12: CAN-BUS Wheel Speed vs. Ground Truth Forward Speed

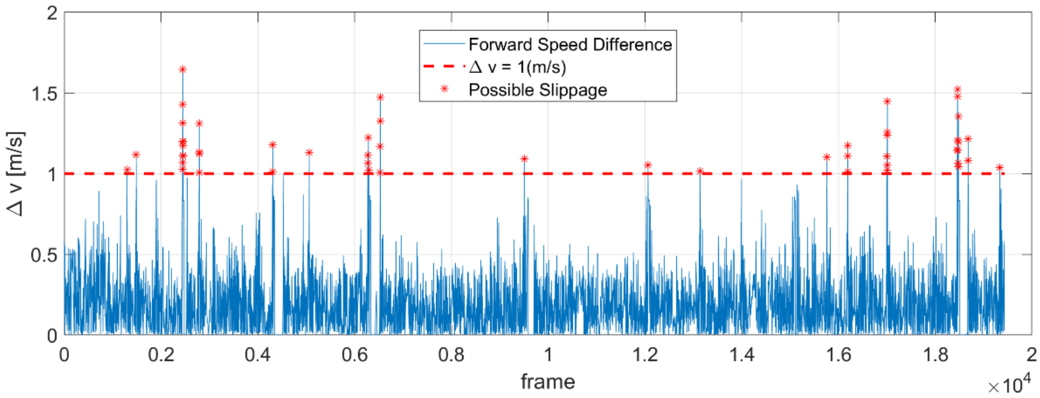


Figure 5-13: Difference Between the Wheel Odometer Output and Ground Truth Forward Velocity

The resolution of CAN-BUS wheel speed is 1 km/h (≈ 0.28 m/s), considering the measurement noise of the sensor, we choose 1 m/s as the threshold to filter out the possible slippages. The total count of possible slippage time is 54 times (out of ~1h driving). Considering the vehicle used for Calgary Winter Driving Dataset was equipped with winter tires, the error brought by the wheel slippage can be neglected. From Table 5-2, it can be noted that all the IMU+WO solutions result in the second best accuracy. This demonstrates that wheel odometer or wheel encoder sensors can be a great alternative and reliable sensor to provide motion constraints for land vehicle navigation in winter time.

Comparing Figure 5-7 (winter-1) and Figure 5-8 (summer-1), the clear road conditions do not show a significant improvement in results as expected. The reasons behind this might be concluded as:

- Given the duration of the dataset, summer-1 is 149 (s) and winter-1 is 66 (s). Both monocular vision and IMU will result in larger drifts as time increase without using absolute information (such as GNSS) to contain the accumulated errors.
- VO and wheel odometers can function normally and be trusted in the conventional winter urban environments.

Table 5-2: Average Root Mean Square Error (ARMSE) of IMU Only, IMU+WO, IMU+Vision and IMU+WO+Vision of Calgary Winter Driving Dataset (winter-1, summer-1 and winter-2)

Dataset		Calgary winter-1 (snow)	Calgary summer-1 (summer)	Calgary winter-2 (snow)
Duration		66 (s)	149 (s)	62 (s)
IMU Only	Horizontal ARMSE (m)	19.724	43.782	14.066
	Rot. ARMSE (deg)	2.023	5.691	1.758
	Final Horizontal Pos Error (m)	60.086	212.201	59.409
IMU+WO	Horizontal ARMSE (m)	6.049	18.538	2.164
	Rot. ARMSE (deg)	2.137	5.654	1.758
	Final Horizontal Pos Error (m)	15.152	54.015	6.002
IMU+Vision	Horizontal ARMSE (m)	12.491	49.891	15.753
	Rot. ARMSE (deg)	5.891	8.881	7.818
	Final Horizontal Pos Error (m)	48.092	251.081	41.088
IMU+WO+Vision	Horizontal ARMSE (m)	7.083	12.084	4.455
	Rot. ARMSE (deg)	3.366	7.054	2.754
	Final Horizontal Pos Error (m)	19.649	39.042	3.456

Chapter 6 CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

In this thesis, a new method to tightly integrate IMU+Vision+WO+NHC based on modification of the MSCKF algorithm proposed by [128] is introduced. The primary objective of this thesis is to investigate and evaluate the proposed algorithm's performances in winter urban environments in order to constrain the navigation errors during the short-term GNSS outages. The implementation of the proposed algorithm is first validated using the KITTI Dataset sequence 0095 and 0117 with the simulated wheel odometer data. Then, in Chapter 5, it is tested with the KAIST Complex Urban Dataset (trajectory 39) and the Calgary Winter Driving Dataset (winter-1. summer-1 and winter-2).

The conclusions and contributions of this thesis are summarized as follows.

1. WO and NHC were able to control the scale drift brought by monocular vision and IMU, and as a result were able to control both scale and orientation over longer periods than IMU+Vision alone.
2. Dynamic scenes in complex urban canyon environments would severely degrade VO's performance. More specially, dynamic objects results in wrong estimates of the feature depth information, which will cause the motion degeneration. The conventional RANSAC or the MSAC outlier rejection algorithms cannot remove all the features tracked on dynamic objects in complex real-world scenarios. By testing on real-world driving data in urban canyon environment, the proposed IMU+Vision+WO algorithm achieved 1.814 m horizontal position error in a 1-minute drive in an urban canyon environment in the KAIST Complex Urban Dataset. This proves that by integrating with

IMU, WO together with NHC, the motion drift brought by dynamic objects can be potentially controlled. The proposed IMU+Vision+WO system could serve as an alternative solution to bridge the GNSS gaps in the challenging urban canyon environments over short periods. However, how to bring out the full potential of VO and provide a robust navigation solution in complex urban scenes still remains a challenging question.

3. In this thesis, a new real-world driving dataset “Calgary Winter Dataset” containing images, IMU, wheel odometer and ground truth were created and soon published online.
4. The proposed IMU+Vision+WO algorithm achieved 19.649 m and 3.456 m horizontal position errors in two 1-minute drives in our Calgary winter urban environment. Compared with the IMU results, the horizontal accuracy improved 74.78% and 89.90% for IMU+WO and IMU+Vision+WO, respectively.
5. It was proven in this thesis that in normal winter urban environments, the salient features per frame in winter driving scenes were above 50, and when winter tires were equipped, the influence of wheel slippage on the sensor integration system was not obvious.

6.2 Recommendations for Future Works

The recommendations for future works are listed as follows.

1. How to properly calibrate the sensors is the key to bring out the best performance of multi-sensor integration systems. In this thesis, the extrinsic parameters between the IMU body frame to the camera frame, as well as the IMU body frame to the vehicle frame are not calibrated with high precision. For the camera-IMU extrinsic calibration, the best way

is still by utilizing 3D LiDAR as medium to calibrate the transformation matrix. However, the Kalibr Toolbox is also very useful tool for this matter. To properly use the Kalibr toolbox, the IMU intrinsic parameters need to be determined accurately beforehand. Also, the platform motion during the calibration should be slow and steady to fully excite all the three axes of the IMU and camera coordinate systems. As for the vehicle-IMU extrinsic parameters calibration, apart from measuring the vehicle lengths accurately, online calibration method can also be given a try.

2. Due the fact that the main objective of this thesis is mainly to investigate the feasibility of using IMU, WO and camera as alternative information sources to bridge the GNSS outages, the VO implementation part is far from optimal. For better accuracies, the BA-based methods together with the loop-closure detection algorithm can enhance the VO performance significantly.
3. Issues such as moving-objects and wheel slippages have their patterns. Deep learning based algorithm should be of great help when dealing with such problems for land vehicle navigation systems.

References

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013, doi: 10.1177/0278364913491297.

[2] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, “Complex urban dataset with multi-level sensors from highly diverse urban environments,” *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 642–657, May 2019, doi: 10.1177/0278364919843996.

[3] I. Skog and P. Handel, “In-Car Positioning and Navigation Technologies—A Survey,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 4–21, Mar. 2009, doi: 10.1109/TITS.2008.2011712.

[4] M. Schlingelhof, D. Bétaille, P. Bonnifait, and K. Demaseure, “Advanced positioning technologies for co-operative systems,” *IET Intell. Transp. Syst.*, vol. 2, no. 2, p. 81, 2008, doi: 10.1049/iet-its:20070046.

[5] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, “Autonomous vehicles: challenges, opportunities, and future implications for transportation policies,” *Journal of Modern Transportation*, vol. 24, no. 4, pp. 284–303, Dec. 2016, doi: 10.1007/s40534-016-0117-3.

[6] “Where to? A History of Autonomous Vehicles,” *CHM*, May 08, 2014. https://computerhistory.org/blog/where-to-a-history-of-autonomous-vehicles/ (accessed Aug. 04, 2020).

[7] G. Seetharaman, A. Lakhotia, and E. P. Blasch, “Unmanned vehicles come of age: The DARPA grand challenge,” *Computer*, vol. 39, no. 12, pp. 26–29, Dec. 2006, doi: 10.1109/MC.2006.447.

[8] “Waymo,” *Wikipedia*. Aug. 03, 2020, Accessed: Aug. 04, 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Waymo&oldid=970963313.

[9] “Driverless Cars Are Further Away Than You Think,” *MIT Technology Review*. https://www.technologyreview.com/2013/10/22/175716/driverless-cars-are-further-away-than-you-think/ (accessed Aug. 04, 2020).

[10] P. D. Groves, *Principles of GNSS, inertial, and multisensor integrated navigation systems*. .

[11] “Satellite navigation,” *Wikipedia*. Aug. 31, 2020, Accessed: Sep. 14, 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Satellite_navigation&oldid=976032172.

[12] R. B. Kershner and R. R. Newton, “The Transit System,” *The Journal of Navigation*, vol. 15, no. 2, pp. 129–144, Apr. 1962, doi: 10.1017/S0373463300035943.

[13] “GNSS Measurements.” https://novatel.com/an-introduction-to-gnss/chapter-5-resolving-errors/gnss-measurements (accessed Sep. 16, 2020).

[14] “Precise Point Positioning (PPP).” https://novatel.com/an-introduction-to-gnss/chapter-5-resolving-errors/precise-point-positioning-ppp (accessed Sep. 18, 2020).

[15] P. J. G. Teunissen and O. Montenbruck, Eds., *Springer Handbook of Global Navigation Satellite Systems*. Cham: Springer International Publishing, 2017.

[16] R. F. Brena, J. P. García-Vázquez, C. E. Galván-Tejada, D. Muñoz-Rodriguez, C. Vargas-Rosales, and J. Fangmeyer, “Evolution of Indoor Positioning Technologies: A Survey,” *Journal of Sensors*, Mar. 29, 2017. https://www.hindawi.com/journals/js/2017/2630413/ (accessed Sep. 14, 2020).

[17] R. Faragher and R. Harle, “Location Fingerprinting With Bluetooth Low Energy Beacons,” *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, Nov. 2015, doi: 10.1109/JSAC.2015.2430281.

[18] S. Farahani, *ZigBee Wireless Networks and Transceivers*. Newnes, 2011.

[19] L. M. Ni, Yunhao Liu, Yiu Cho Lau, and A. P. Patil, “LANDMARC: indoor location sensing using active RFID,” in *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003).*, Mar. 2003, pp. 407–415, doi: 10.1109/PERCOM.2003.1192765.

[20] C. R. Carlson, J. C. Gerdes, and J. D. Powell, “Practical Position and Yaw Rate Estimation with GPS and Differential Wheelspeeds,” p. 8.

[21] J. Borenstein and Liqiang Feng, “Measurement and correction of systematic odometry errors in mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 6, pp. 869–880, Dec. 1996, doi: 10.1109/70.544770.

[22] E. Abbott and D. Powell, “Land-vehicle navigation using GPS,” *Proceedings of the IEEE*, vol. 87, no. 1, pp. 145–162, Jan. 1999, doi: 10.1109/5.736347.

[23] A. Morrison, V. Renaudin, J. B. Bancroft, and G. Lachapelle, “Design and Testing of a Multi-Sensor Pedestrian Location and Navigation Platform,” *Sensors (Basel)*, vol. 12, no. 3, pp. 3720–3738, Mar. 2012, doi: 10.3390/s120303720.

[24] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, “Integrity of map-matching algorithms,” *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 4, pp. 283–302, Aug. 2006, doi: 10.1016/j.trc.2006.08.004.

[25] B. Ranft and C. Stiller, “The Role of Machine Vision for Intelligent Vehicles,” *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 8–19, Mar. 2016, doi: 10.1109/TIV.2016.2551553.

[26] A. J. Hawkins, “Elon Musk still doesn’t think LIDAR is necessary for fully driverless cars,” *The Verge*, Feb. 07, 2018. https://www.theverge.com/2018/2/7/16988628/elon-musk-lidar-self-driving-car-tesla (accessed Jun. 13, 2019).

[27] L. Song, W. Wu, J. Guo, and X. Li, “Survey on Camera Calibration Technique,” in *2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*, Aug. 2013, vol. 2, pp. 389–392, doi: 10.1109/IHMSC.2013.240.

[28] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.

[29] R. O. Duda and P. E. Hart, “Use of the Hough transformation to detect lines and curves in pictures,” *Commun. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972, doi: 10.1145/361237.361242.

[30] A. de la Escalera, J. M. Armingol, and M. Mata, “Traffic sign recognition and analysis for intelligent vehicles,” *Image and Vision Computing*, vol. 21, no. 3, pp. 247–258, Mar. 2003, doi: 10.1016/S0262-8856(02)00156-7.

[31] T. Schwarze and M. Lauer, *Minimizing Odometry Drift by Vanishing Direction References*. .

[32] T. Tuytelaars and K. Mikolajczyk, “Local Invariant Feature Detectors: A Survey,” *CGV*, vol. 3, no. 3, pp. 177–280, Jun. 2008, doi: 10.1561/0600000017.

[33] Y. Li, S. Wang, Q. Tian, and X. Ding, “A survey of recent advances in visual feature detection,” *Neurocomputing*, vol. 149, pp. 736–751, Feb. 2015, doi: 10.1016/j.neucom.2014.08.003.

[34] D. Scaramuzza and F. Fraundorfer, “Visual Odometry [Tutorial],” *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, Dec. 2011, doi: 10.1109/MRA.2011.943233.

[35] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” in *Procedings of the Alvey Vision Conference 1988*, Manchester, 1988, p. 23.1-23.6, doi: 10.5244/C.2.23.

[36] C. Tomasi and T. Kanade, “Detection and Tracking of Point Features,” p. 22.

[37] Jianbo Shi and Tomasi, “Good features to track,” in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 1994, pp. 593–600, doi: 10.1109/CVPR.1994.323794.

[38] E. Rosten and T. Drummond, “Machine Learning for High-Speed Corner Detection,” in *Computer Vision – ECCV 2006*, Berlin, Heidelberg, 2006, pp. 430–443, doi: 10.1007/11744023_34.

[39] E. Rosten, R. Porter, and T. Drummond, “Faster and Better: A Machine Learning Approach to Corner Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, Jan. 2010, doi: 10.1109/TPAMI.2008.275.

[40] X. Zhang, H. Wang, A. W. B. Smith, X. Ling, B. C. Lovell, and D. Yang, “Corner detection based on gradient correlation matrices of planar curves,” *Pattern Recognition*, vol. 43, no. 4, pp. 1207–1223, Apr. 2010, doi: 10.1016/j.patcog.2009.10.017.

[41] G.-S. Xia, J. Delon, and Y. Gousseau, “Accurate Junction Detection and Characterization in Natural Images,” *International Journal of Computer Vision*, vol. 1, no. 106, pp. 31–56, 2014, doi: 10.1007/s11263-013-0640-1.

[42] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Sep. 1999, vol. 2, pp. 1150–1157 vol.2, doi: 10.1109/ICCV.1999.790410.

[43] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded Up Robust Features,” in *Computer Vision – ECCV 2006*, Berlin, Heidelberg, 2006, pp. 404–417, doi: 10.1007/11744023_32.

[44] B. Li, R. Xiao, Z. Li, R. Cai, B.-L. Lu, and L. Zhang, “Rank-SIFT: Learning to rank repeatable local interest points,” in *CVPR 2011*, Jun. 2011, pp. 1737–1744, doi: 10.1109/CVPR.2011.5995461.

[45] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, “KAZE Features,” in *Computer Vision – ECCV 2012*, Berlin, Heidelberg, 2012, pp. 214–227, doi: 10.1007/978-3-642-33783-3_16.

[46] S. Salti, A. Lanza, and L. Di Stefano, “Keypoints from Symmetries by Wave Propagation,” 2013, pp. 2898–2905, Accessed: Oct. 07, 2020. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2013/html/Salti_Keypoints_from_Symmetries_2013_CVPR_paper.html.

[47] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571, doi: 10.1109/ICCV.2011.6126544.

[48] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary Robust invariant scalable keypoints,” in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2548–2555, doi: 10.1109/ICCV.2011.6126542.

[49] A. Alahi, R. Ortiz, and P. Vandergheynst, “FREAK: Fast Retina Keypoint,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 510–517, doi: 10.1109/CVPR.2012.6247715.

[50] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, Sep. 2004, doi: 10.1016/j.imavis.2004.02.006.

[51] C. Cui and K. N. Ngan, “Scale- and Affine-Invariant Fan Feature,” *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1627–1640, Jun. 2011, doi: 10.1109/TIP.2010.2103948.

[52] J. Kim and K. Grauman, “Boundary Preserving Dense Local Regions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 5, pp. 931–943, May 2015, doi: 10.1109/TPAMI.2014.2360689.

[53] R. Mur-Artal and J. D. Tardós, “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017, doi: 10.1109/TRO.2017.2705103.

[54] A. Geiger, M. Roser, and R. Urtasun, “Efficient Large-Scale Stereo Matching,” in *Computer Vision – ACCV 2010*, Berlin, Heidelberg, 2011, pp. 25–38, doi: 10.1007/978-3-642-19315-6_3.

[55] Y. Boykov, O. Veksler, and R. Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001, doi: 10.1109/34.969114.

[56] F. Guney and A. Geiger, “Displets: Resolving Stereo Ambiguities Using Object Knowledge,” 2015, pp. 4165–4175, Accessed: Oct. 07, 2020. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Guney_Displets_Resolving_Stereo_2015_CVPR_paper.html.

[57] K. Simonyan and A. Zisserman, “Two-Stream Convolutional Networks for Action Recognition in Videos,” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 568–576.

[58] T. Xue, M. Rubinstein, N. Wadhwa, A. Levin, F. Durand, and W. T. Freeman, “Refraction Wiggles for Measuring Fluid Depth and Velocity from Video,” in *Computer Vision – ECCV 2014*, Cham, 2014, pp. 767–782, doi: 10.1007/978-3-319-10578-9_50.

[59] C. Rabe, T. Müller, A. Wedel, and U. Franke, “Dense, Robust, and Accurate Motion Field Estimation from Stereo Image Sequences in Real-Time,” in *Computer Vision – ECCV 2010*, Berlin, Heidelberg, 2010, pp. 582–595, doi: 10.1007/978-3-642-15561-1_42.

[60] A. Geiger, J. Ziegler, and C. Stiller, “StereoScan: Dense 3d reconstruction in real-time,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2011, pp. 963–968, doi: 10.1109/IVS.2011.5940405.

[61] M. Menze and A. Geiger, “Object Scene Flow for Autonomous Vehicles,” 2015, pp. 3061–3070, Accessed: Oct. 07, 2020. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2015/html/Menze_Object_Scene_Flow_2015_CVPR_paper.html.

[62] G. Dissanayake, S. Sukkarieh, E. Nebot, and H. Durrant-Whyte, “The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 5, pp. 731–747, Oct. 2001, doi: 10.1109/70.964672.

[63] X. Niu, S. Nassar, and N. El-Sheimy, “An Accurate Land-Vehicle MEMS IMU/GPS Navigation System Using 3D Auxiliary Velocity Updates,” *NAVIGATION*, vol. 54, no. 3, pp. 177–188, 2007, doi: 10.1002/j.2161-4296.2007.tb00403.x.

[64] X. Niu, S. Nassar, and N. El-Sheimy, “An Accurate Land-Vehicle MEMS IMU/GPS Navigation System Using 3D Auxiliary Velocity Updates,” *Navigation*, vol. 54, no. 3, pp. 177–188, 2007, doi: 10.1002/j.2161-4296.2007.tb00403.x.

[65] Z. Liu, “Vision Sensor Aided Navigation for Ground Vehicle Applications,” University of Calgary, Calgary, AB, 2019.

[66] D. Scaramuzza, A. Censi, and K. Daniilidis, “Exploiting Motion Priors in Visual Odometry for Vehicle-Mounted Cameras with Non-holonomic Constraints,” p. 8.

[67] I. Klein, S. Filin, and T. Toledo, “Pseudo-Measurements as Aiding to INS during GPS Outages,” *NAVIGATION*, vol. 57, no. 1, pp. 25–34, 2010, doi: 10.1002/j.2161-4296.2010.tb01765.x.

[68] S. Godha and M. E. Cannon, “GPS/MEMS INS integrated system for navigation in urban areas,” *GPS Solut*, vol. 11, no. 3, pp. 193–203, Jul. 2007, doi: 10.1007/s10291-006-0050-8.

[69] O. Mezentsev, “Pedestrian and Vehicular Navigation Under Signal Masking Using Integrated HSGPS and Self Contained Sensor Technologies,” p. 29, 2003.

[70] T. O. Tindall, “Evaluation of the Position and Azimuth Determining System’s Potential for Higher Accuracy Survey,” ARMY ENGINEER TOPOGRAPHIC LABS FORT BELVOIR VA, Mar. 1982. Accessed: Oct. 06, 2020. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA109006.

[71] L. Xiaofang, M. Yuliang, X. Ling, C. Jiabin, and S. Chunlei, “Applications of zero-velocity detector and Kalman filter in zero velocity update for inertial navigation system,” in *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*, Aug. 2014, pp. 1760–1763, doi: 10.1109/CGNCC.2014.7007449.

[72] G. A. O. Zhongyu, “Kalman Filter Design of Inertial Positioning System [J],” *Journal of Chinese Inertial Technology*, vol. 4, 2000.

[73] Y. Wang and A. M. Shkel, “Adaptive Threshold for Zero-Velocity Detector in ZUPT-Aided Pedestrian Inertial Navigation,” *IEEE Sensors Letters*, vol. 3, no. 11, pp. 1–4, Nov. 2019, doi: 10.1109/LSENS.2019.2946129.

[74] Y. Kone, N. Zhu, V. Renaudin, and M. Ortiz, “Machine Learning-Based Zero-Velocity Detection for Inertial Pedestrian Navigation,” *IEEE Sensors Journal*, vol. 20, no. 20, pp. 12343–12353, Oct. 2020, doi: 10.1109/JSEN.2020.2999863.

[75] D. A. Grejner-Brzezinska, C. K. Toth, and Y. Yi, “Bridging GPS Gaps in Urban Canyons: Can ZUPT Really Help?,” Jun. 2002, pp. 231–240, Accessed: Sep. 30, 2020. [Online]. Available: http://www.ion.org/publications/abstract.cfm?jp=p&articleID=956.

[76] E.-H. Shin and N. El-Sheimy, “Accuracy Improvement of Low Cost INS/GPS for Land Applications,” Jan. 2002, pp. 146–157, Accessed: Oct. 06, 2020. [Online]. Available: http://www.ion.org/publications/abstract.cfm?jp=p&articleID=204.

[77] “Canadian Collision Statistics in the Winter - Winter Driving Preperation,” *Waterdown Collision*, Dec. 09, 2016. https://www.waterdowncollision.com/blog/auto-repair/winter-collision-statistics-canada/ (accessed Oct. 14, 2019).

[78] “Waymo,” *Waymo*. https://waymo.com/ (accessed Oct. 09, 2020).

[79] G. Huang, “Visual-Inertial Navigation: A Concise Review,” *arXiv:1906.02650 [cs]*, Jun. 2019, Accessed: Jul. 18, 2019. [Online]. Available: http://arxiv.org/abs/1906.02650.

[80] C. Chen, H. Zhu, M. Li, and S. You, "A Review of Visual-Inertial Simultaneous Localization and Mapping from Filtering-Based and Optimization-Based Perspectives," *Robotics*, vol. 7, no. 3, p. 45, Sep. 2018, doi: 10.3390/robotics7030045.

[81] C.-R. Lee and K.-J. Yoon, "Fusion of Camera, IMU, and Speedometer for Localization of Autonomous Vehicles," p. 3.

[82] D. Tian, X. He, L. Zhang, J. Lian, and X. Hu, "A Design of Odometer-Aided Visual Inertial Integrated Navigation Algorithm Based on Multiple View Geometry Constraints," in *2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Aug. 2017, vol. 1, pp. 161–166, doi: 10.1109/IHMSC.2017.43.

[83] M. Quan, S. Piao, M. Tan, and S. Huang, "Tightly-Coupled Monocular Visual-Odometric SLAM Using Wheels and a MEMS Gyroscope," *IEEE Access*, vol. 7, pp. 97374–97389, 2019, doi: 10.1109/ACCESS.2019.2930201.

[84] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex Urban LiDAR Data Set," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, May 2018, pp. 6344–6351, doi: 10.1109/ICRA.2018.8460834.

[85] D. Nister, "An efficient solution to the five-point relative pose problem," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, Jun. 2003, vol. 2, p. II–195, doi: 10.1109/CVPR.2003.1211470.

[86] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, pp. 133–135, 1981.

[87] C. G. Harris and J. M. Pike, "3D positional integration from image sequences," *Image and Vision Computing*, vol. 6, no. 2, pp. 87–90, 1988.

[88] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover.," Stanford Univ CA Dept of Computer Science, 1980.

[89] C. F. Olson, L. H. Matthies, J. R. Wright, R. Li, and K. Di, "Visual terrain mapping for Mars exploration," *Computer Vision and Image Understanding*, vol. 105, no. 1, pp. 73–85, Jan. 2007, doi: 10.1016/j.cviu.2006.08.005.

[90] J. Artieda *et al.*, "Visual 3-D SLAM from UAVs," *J Intell Robot Syst*, vol. 55, no. 4, p. 299, Jan. 2009, doi: 10.1007/s10846-008-9304-8.

[91] M. Johnson-Roberson, O. Pizarro, S. B. Williams, and I. Mahon, "Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys," *Journal of Field Robotics*, vol. 27, no. 1, pp. 21–51, 2010, doi: 10.1002/rob.20324.

[92] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov. 2007, pp. 225–234, doi: 10.1109/ISMAR.2007.4538852.

[93] C. Mei, E. Sommerlade, G. Sibley, P. Newman, and I. Reid, "Hidden view synthesis using real-time visual SLAM for simplifying video surveillance analysis," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 4240–4245, doi: 10.1109/ICRA.2011.5980093.

[94] O. G. Grasa, J. Civera, and J. M. M. Montiel, "EKF monocular SLAM with relocalization for laparoscopic sequences," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 4816–4821, doi: 10.1109/ICRA.2011.5980059.

[95] R. Gonzalez, F. Rodriguez, J. L. Guzman, C. Pradalier, and R. Siegwart, "Combined visual odometry and visual compass for off-road mobile robots localization," *Robotica*, vol. 30, no. 6, pp. 865–878, Oct. 2012, doi: 10.1017/S026357471100110X.

[96] M. O. A. Aqel, M. H. Marhaban, M. I. Saripan, and N. Bt. Ismail, “Review of visual odometry: types, approaches, challenges, and applications,” *SpringerPlus*, vol. 5, no. 1, p. 1897, Oct. 2016, doi: 10.1186/s40064-016-3573-7.

[97] T. Takahashi, “2D localization of outdoor mobile robots using 3D laser range data,” Carnegie Mellon University, 2007.

[98] C. Cadena *et al.*, “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016, doi: 10.1109/TRO.2016.2624754.

[99] B. M. Kitt, J. Rehder, A. D. Chambers, M. Schonbein, H. Lategahn, and S. Singh, “Monocular Visual Odometry using a Planar Road Model to Solve Scale Ambiguity,” Sep. 2011, doi: 10.1184/R1/6555626.v1.

[100] R. Chatila and J. Laumond, “Position referencing and consistent world modeling for mobile robots,” in *1985 IEEE International Conference on Robotics and Automation Proceedings*, Mar. 1985, vol. 2, pp. 138–145, doi: 10.1109/ROBOT.1985.1087373.

[101] T. Taketomi, H. Uchiyama, and S. Ikeda, “Visual SLAM algorithms: a survey from 2010 to 2016,” *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, p. 16, Jun. 2017, doi: 10.1186/s41074-017-0027-2.

[102] M. Labbé and F. Michaud, “Appearance-Based Loop Closure Detection for Online Large-Scale and Long-Term Operation,” *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 734–745, Jun. 2013, doi: 10.1109/TRO.2013.2242375.

[103] A. I. Comport, E. Malis, and P. Rives, “Accurate Quadrifocal Tracking for Robust 3D Visual Odometry,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 40–45, doi: 10.1109/ROBOT.2007.363762.

[104] D. Nistér, O. Naroditsky, and J. Bergen, “Visual odometry for ground vehicle applications,” *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, 2006, doi: 10.1002/rob.20103.

[105] M. Lhuillier, “Automatic Structure and Motion using a Catadioptric Camera,” United States, 2005, Accessed: Jul. 04, 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00091156.

[106] J. Tardif, Y. Pavlidis, and K. Daniilidis, “Monocular visual odometry in urban environments using an omnidirectional camera,” in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2008, pp. 2531–2538, doi: 10.1109/IROS.2008.4651205.

[107] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, “MonoSLAM: Real-Time Single Camera SLAM,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007, doi: 10.1109/TPAMI.2007.1049.

[108] G. Klein and D. Murray, “Parallel Tracking and Mapping for Small AR Workspaces,” in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov. 2007, pp. 225–234, doi: 10.1109/ISMAR.2007.4538852.

[109] H. Strasdat, J. M. M. Montiel, and A. J. Davison, “Visual SLAM: Why filter?,” *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, Feb. 2012, doi: 10.1016/j.imavis.2012.02.009.

[110] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, “ORB-SLAM: A Versatile and Accurate Monocular SLAM System,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015, doi: 10.1109/TRO.2015.2463671.

[111] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, “ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM,” *arXiv:2007.11898 [cs]*, Jul. 2020, Accessed: Oct. 24, 2020. [Online]. Available: http://arxiv.org/abs/2007.11898.

[112] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, “DTAM: Dense tracking and mapping in real-time,” in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2320–2327, doi: 10.1109/ICCV.2011.6126513.

[113] J. Engel, J. Sturm, and D. Cremers, “Semi-dense Visual Odometry for a Monocular Camera,” 2013, pp. 1449–1456, Accessed: Oct. 26, 2020. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2013/html/Engel_Semi-dense_Visual_Odometry_2013_ICCV_paper.html.

[114] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: Large-Scale Direct Monocular SLAM,” in *Computer Vision – ECCV 2014*, Cham, 2014, pp. 834–849, doi: 10.1007/978-3-319-10605-2_54.

[115] C. Forster, M. Pizzoli, and D. Scaramuzza, “SVO: Fast semi-direct monocular visual odometry,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 15–22, doi: 10.1109/ICRA.2014.6906584.

[116] J. Engel, V. Koltun, and D. Cremers, “Direct Sparse Odometry,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, Mar. 2018, doi: 10.1109/TPAMI.2017.2658577.

[117] K. Tateno, F. Tombari, I. Laina, and N. Navab, “CNN-SLAM: Real-Time Dense Monocular SLAM with Learned Depth Prediction,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, Jul. 2017, pp. 6565–6574, doi: 10.1109/CVPR.2017.695.

[118] A. Kendall, M. Grimes, and R. Cipolla, “PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization,” 2015, pp. 2938–2946, Accessed: Jul. 22, 2019. [Online]. Available: https://www.cv-foundation.org/openaccess/content_iccv_2015/html/Kendall_PoseNet_A_Convolutional_ICCV_2015_paper.html.

[119] R. Li, S. Wang, Z. Long, and D. Gu, “UnDeepVO: Monocular Visual Odometry Through Unsupervised Deep Learning,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 7286–7291, doi: 10.1109/ICRA.2018.8461251.

[120] S. Izadi, “KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera,” *In Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp. 559–568.

[121] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison, “SLAM++: Simultaneous Localisation and Mapping at the Level of Objects,” 2013, pp. 1352–1359, Accessed: Oct. 26, 2020. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2013/html/Salas-Moreno_SLAM_Simultaneous_Localisation_2013_CVPR_paper.html.

[122] G. Gallego *et al.*, “Event-based Vision: A Survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2020, doi: 10.1109/TPAMI.2020.3008413.

[123] Y. Zhou, G. Gallego, H. Rebecq, L. Kneip, H. Li, and D. Scaramuzza, “Semi-Dense 3D Reconstruction with a Stereo Event Camera,” 2018, pp. 235–251, Accessed: Oct. 29, 2020. [Online]. Available:

https://openaccess.thecvf.com/content_ECCV_2018/html/Yi_Zhou_Semi-Dense_3D_Reconstruction_ECCV_2018_paper.html.

[124] C. Merfels, "Sensor fusion for localization of automated vehicles."

[125] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 957–964, doi: 10.1109/ICRA.2012.6225147.

[126] M. Rosencrantz, G. Gordon, and S. Thrun, "Decentralized Sensor Fusion With Distributed Particle Filters," *arXiv:1212.2493 [cs]*, Oct. 2012, Accessed: Oct. 28, 2020. [Online]. Available: http://arxiv.org/abs/1212.2493.

[127] J. Kim and S. Sukkarieh, "Real-time implementation of airborne inertial-SLAM," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 62–71, Jan. 2007, doi: 10.1016/j.robot.2006.06.006.

[128] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Rome, Italy, Apr. 2007, pp. 3565–3572, doi: 10.1109/ROBOT.2007.364024.

[129] M. Bryson and S. Sukkarieh, "Observability analysis and active control for airborne SLAM," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 1, pp. 261–280, Jan. 2008, doi: 10.1109/TAES.2008.4517003.

[130] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, A. E. Johnson, A. Ansar, and L. Matthies, "Vision-Aided Inertial Navigation for Spacecraft Entry, Descent, and Landing," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 264–280, Apr. 2009, doi: 10.1109/TRO.2009.2012342.

[131] Z. Huai and G. Huang, "Robocentric Visual-Inertial Odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 6319–6326, doi: 10.1109/IROS.2018.8593643.

[132] S. Ebcin and M. Veth, "Tightly-Coupled Image-Aided Inertial Navigation Using the Unscented Kalman Filter," AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH, Jan. 2007. Accessed: Oct. 28, 2020. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA473019.

[133] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing," *Robotics and Autonomous Systems*, vol. 61, no. 8, pp. 721–738, Aug. 2013, doi: 10.1016/j.robot.2013.05.001.

[134] K. Sun *et al.*, "Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 965–972, Apr. 2018, doi: 10.1109/LRA.2018.2793349.

[135] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: a generalized framework for processing relative state measurements," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, May 2002, vol. 2, pp. 1788–1795 vol.2, doi: 10.1109/ROBOT.2002.1014801.

[136] M. Li and A. I. Mourikis, "Improving the accuracy of EKF-based visual-inertial odometry," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 828–835, doi: 10.1109/ICRA.2012.6225229.

[137] L. E. Clement, V. Peretroukhin, J. Lambert, and J. Kelly, "The Battle for Filter Supremacy: A Comparative Study of the Multi-State Constraint Kalman Filter and the

Sliding Window Filter," in *2015 12th Conference on Computer and Robot Vision*, Halifax, NS, Canada, Jun. 2015, pp. 23–30, doi: 10.1109/CRV.2015.11.

[138] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle Adjustment — A Modern Synthesis," in *Vision Algorithms: Theory and Practice*, 2000, pp. 298–372.

[139] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-Based Visual-Inertial SLAM using Nonlinear Optimization," presented at the Robotics: Science and Systems 2013, Jun. 2013, doi: 10.15607/RSS.2013.IX.037.

[140] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018, doi: 10.1109/TRO.2018.2853729.

[141] G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia, "Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 18–25, Jan. 2016, doi: 10.1109/LRA.2015.2505717.

[142] D. Eigen and R. Fergus, "Predicting Depth, Surface Normals and Semantic Labels With a Common Multi-Scale Convolutional Architecture," 2015, pp. 2650–2658, Accessed: Jul. 22, 2019. [Online]. Available: https://www.cv-foundation.org/openaccess/content_iccv_2015/html/Eigen_Predicting_Depth_Surface_ICCV_2015_paper.html.

[143] B. M. Aumayer, "Ultra-tightly Coupled Vision/GNSS for Automotive Applications," 2017.

[144] "Object for storing camera parameters - MATLAB." https://www.mathworks.com/help/vision/ref/cameraparameters.html (accessed Nov. 18, 2020).

[145] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000, doi: 10.1109/34.888718.

[146] D. Arthur and S. Vassilvitskii, "k-means++: The Advantages of Careful Seeding," Jun. 2006, Accessed: Nov. 30, 2020. [Online]. Available: http://ilpubs.stanford.edu:8090/778/.

[147] M. Shahbazi, G. Sohn, and J. Théau, "Evolutionary Optimization for Robust Epipolar-Geometry Estimation and Outlier Detection," *Algorithms*, vol. 10, no. 3, Art. no. 3, Sep. 2017, doi: 10.3390/a10030087.

[148] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 930–943, Aug. 2003, doi: 10.1109/TPAMI.2003.1217599.

[149] V. LEPETIT, F. MORENO-NOGUER, and P. FUA, "EPnP : An Accurate O(n) Solution to the PnP Problem," *Int. j. comput. vis*, vol. 81, no. 2, pp. 155–166, 2009.

[150] A. Penate-Sanchez, J. Andrade-Cetto, and F. Moreno-Noguer, "Exhaustive Linearization for Robust Camera Pose and Focal Length Estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 10, pp. 2387–2400, Oct. 2013, doi: 10.1109/TPAMI.2013.36.

[151] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981, doi: 10.1145/358669.358692.

[152] S. Choi, T. Kim, and W. Yu, “Performance Evaluation of RANSAC Family,” in *Procedings of the British Machine Vision Conference 2009*, London, 2009, p. 81.1-81.12, doi: 10.5244/C.23.81.

[153] M. Li, “Visual-Inertial Odometry on Resource-Constrained Systems,” 2014.

[154] C. Forster, “Visual Inertial Odometry and Active Dense Reconstruction for Mobile Robots,” Dissertation, University of Zurich, 2016.

[155] Yanling Hao, Zhilan Xiong, Wei Gao, and Lijuan Li, “Study of strapdown inertial navigation integration algorithms,” in *2004 International Conference on Intelligent Mechatronics and Automation, 2004. Proceedings.*, Aug. 2004, pp. 751–754, doi: 10.1109/ICIMA.2004.1384296.

[156] “KAIST Urban Data Set.” http://irap.kaist.ac.kr/dataset (accessed Dec. 08, 2020).

[157] Paul Verlaine Gakne, “Improving the Accuracy of GNSS Receivers in Urban Canyons using an Upward-Facing Camera,” 2018.

[158] “Evaluating the Accuracy of Single Camera Calibration - MATLAB & Simulink.” https://www.mathworks.com/help/vision/ug/evaluating-the-accuracy-of-single-camera-calibration.html (accessed Dec. 10, 2020).

[159] N. El-Sheimy, “Inertial techniques and INS/DGPS integration,” *Engo 623-Course Notes*, pp. 170–182, 2006.

[160] *rpng/kalibr_allan*. Robot Perception & Navigation Group (RPNG), 2020.

[161] *ethz-asl/kalibr*. ETHZ ASL, 2020.

[162] C. Yu *et al.*, “DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments,” *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1168–1174, Oct. 2018, doi: 10.1109/IROS.2018.8593691.

Appendix A: Copyright Materials

(1) Figure 1-2

https://s100.copyright.com/AppDispatchServlet?title=Autonomous%20vehicles%3A%20challenges%2C%20opportunities%2C%20and%20future%20implications%20for%20transportation%20policies&author=Saeed%20Asadi%20Bagloee%20et%20al&contentID=10.1007%2Fs40534-016-0117-3&copyright=The%20Author%28s%29&publication=2095-087X&publicationDate=2016-08-29&publisherName=SpringerNature&orderBeanReset=true&oa=CC%20BY

(2) Figure 3-4

https://s100.copyright.com/AppDispatchServlet#formTop

(3) Figure 3-9 and Figure 3-10

https://dblp.org/db/about/copyright.html

(4) Figure 4-1 and Figure 4-2

https://s100.copyright.com/AppDispatchServlet#formTop

(5) Figure 4-3 and Figure 4-4

https://s100.copyright.com/AppDispatchServlet#formTop

Appendix B: Noise Parameters and Filter Initialization

(1) KITTI Dataset

- IMU noise parameters

Noise Parameter	Value (for SPAN-LCI IMU)	Unit
Gyroscope “white noise” σ_g	0.001	$\frac{rad}{s}\frac{1}{\sqrt{Hz}}$
Accelerometer “white noise” σ_a	0.01	$\frac{m}{s^2}\frac{1}{\sqrt{Hz}}$
Gyroscope “random walk” σ_{b_g}	0.001	$\frac{rad}{s^2}\frac{1}{\sqrt{Hz}}$
Accelerometer “random walk” σ_{b_a}	0.0005	$\frac{m}{s^3}\frac{1}{\sqrt{Hz}}$

- Image pixel measurement noise parameters

Noise Parameter	Value
Pixel coordinate variance in u direction	2.3242e-04
Pixel coordinate variance in v direction	2.3242e-04

- Wheel odometer measurement noise parameters

Noise Parameter	Value	Unit
Wheel odometer measurement noise	0.5	m/s

- Initial covariance value

Noise Parameter	Initial Value
Orientation q_{var_intial}	1e-6 * ones(1,3)
Position $p_{var\_initial}$	1e-6 * ones(1,3)
Velocity $v_{var\_initial}$	0.5 * ones(1,3)
Gyro bias $b_{g_{var\_initial}}$	1e-4 * ones(1,3)
Accelerometer bias $b_{a_{var\_initial}}$	1e-1 * ones(1,3)

(2) KAIST Complex Urban Dataset

- IMU noise parameters

Noise Parameter	Value (for SPAN-LCI IMU)	Unit
Gyroscope “white noise” σ_g	0.01	$\frac{rad}{s}\frac{1}{\sqrt{Hz}}$
Accelerometer “white noise” σ_a	0.02	$\frac{m}{s^2}\frac{1}{\sqrt{Hz}}$
Gyroscope “random walk” σ_{b_g}	0.01	$\frac{rad}{s^2}\frac{1}{\sqrt{Hz}}$
Accelerometer “random walk” σ_{b_a}	0.005	$\frac{m}{s^3}\frac{1}{\sqrt{Hz}}$

- Image pixel measurement noise parameters

Noise Parameter	Value
Pixel coordinate variance in u direction	1.8154e-04
Pixel coordinate variance in v direction	1.8111e-04

- Wheel odometer measurement noise parameters

Noise Parameter	Value	Unit
Wheel odometer measurement noise	0.1	m/s

- Initial covariance value

Noise Parameter	Initial Value
Orientation q_{var_intial}	0 * ones(1,3)
Position $p_{var\_initial}$	0 * ones(1,3)
Velocity $v_{var\_initial}$	0.7 * ones(1,3)
Gyro bias $b_{g_{var\_initial}}$	1e-2 * ones(1,3)
Accelerometer bias $b_{a_{var\_initial}}$	1e-1 * ones(1,3)

(3) Calgary Winter Driving Dataset

- IMU noise parameters

Noise Parameter	Value (for SPAN-LCI IMU)	Unit
Gyroscope “white noise” σ_g	0.01	$\frac{rad}{s}\frac{1}{\sqrt{Hz}}$
Accelerometer “white noise” σ_a	0.02	$\frac{m}{s^2}\frac{1}{\sqrt{Hz}}$
Gyroscope “random walk” σ_{b_g}	0.01	$\frac{rad}{s^2}\frac{1}{\sqrt{Hz}}$
Accelerometer “random walk” σ_{b_a}	0.005	$\frac{m}{s^3}\frac{1}{\sqrt{Hz}}$

- Image pixel measurement noise parameters

Noise Parameter	Value
Pixel coordinate variance in u direction	1.0000e-03
Pixel coordinate variance in v direction	1.0000e-03

- Wheel odometer measurement noise parameters

Noise Parameter	Value	Unit
Wheel odometer measurement noise	0.3	m/s

- Initial covariance value

Noise Parameter	Initial Value
Orientation q_{var_intial}	0 * ones(1,3)
Position $p_{var\_initial}$	0 * ones(1,3)
Velocity $v_{var\_initial}$	0.7 * ones(1,3)
Gyro bias $b_{g_{var\_initial}}$	1e-2 * ones(1,3)
Accelerometer bias $b_{a_{var\_initial}}$	1e-1 * ones(1,3)