VISUAL-INERTIAL SENSOR FUSION FOR 3D URBAN MODELING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

SALİM SIRTKAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2013

Approval of the thesis:

VISUAL-INERTIAL SENSOR FUSION FOR 3D URBAN MODELING

submitted by **SALİM SIRTKAYA** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. A. Aydın Alatan
Supervisor, **Electrical and Electronics Engineering Department**

Examining Committee Members:

Prof. Dr. Kemal Leblebicioğlu
Electrical and Electronics Engineering Department, METU

Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering Department, METU

Assoc. Prof. Dr. Afşar Saranlı
Electrical and Electronics Engineering Department, METU

Assoc. Prof. Dr. Uluç Saranlı
Computer Engineering Department, METU

Assoc. Prof. Dr. Selim Aksoy
Computer Engineering Department, Bilkent University

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: SALİM SIRTKAYA

Signature :

ABSTRACT

VISUAL-INERTIAL SENSOR FUSION FOR 3D URBAN MODELING

SIRTKAYA, SALİM
Ph.D., Department of Electrical and Electronics Engineering
Supervisor : Prof. Dr. A. Aydın Alatan

September 2013, 128 pages

In this dissertation, a real-time, autonomous and geo-registered approach is presented to tackle the large scale 3D urban modeling problem using a camera and inertial sensors. The proposed approach exploits the special structures of urban areas and visual-inertial sensor fusion. The buildings in urban areas are assumed to have planar facades that are perpendicular to the local level. A sparse 3D point cloud of the imaged scene is obtained from visual feature matches using camera poses estimates, and planar patches are obtained by an iterative Hough Transform on the 2D projection of the sparse 3D point cloud in the direction of gravity. The result is a compact and dense depth map of the building facades in terms of planar patches. The plane extraction is performed on sequential frames and a complete model is obtained by plane fusion. Inertial sensor integration helps to improve camera pose estimation, 3D reconstruction and planar modeling stages. For camera pose estimation, the visual measurements are integrated with the inertial sensors by means of an indirect feedback Kalman filter. This integration helps to get reliable and geo-referenced camera pose estimates in the absence of GPS. The inertial sensors are also used to filter out spurious visual feature matches in the 3D reconstruction stage, find the direction of gravity in plane search stage, and eliminate out of scope objects from the model using elevation data. The visual-inertial sensor fusion and urban heuristics utilization are shown to outperform the classical approaches for large scale urban modeling in terms of consistency and real-time applicability.

Keywords: 3D Point Clouds, Planar Facade Modeling, Visual-Inertial Sensor Integration, Kalman Filter

ÖZ

GÖRSEL-ATALETSEL DUYAÇ TÜMLEŞTİRME KULLANILARAK ŞEHİRLERDE 3B MODELLEME

SIRTKAYA, SALİM
Doktora, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. A. Aydın Alatan

Eylül 2013 , 128 sayfa

Bu tez çalışmasında, geniş şehirsel alanların kamera ve ataletsel sensörler kullanılarak gerçek zamanlı, otomatik ve coğrafi referanslı 3B modellerinin çıkarılması problemi ele alınmıştır. Önerilen çözüm yöntemi, şehirsel alanların kendilerine has özelliklerini kullanma ve görsel-ataletsel duyaç tümleştirme konuları üzerine kurulmuştur. Bunun için şehirsel alanlardaki bina yüzeylerinin düzlemsel ve yer yüzeyine dik olduğu varsayılmıştır. Görsel işaretler ve kamera konum kestirimleri kullanılarak, görüntülenen sahneden seyrek 3B nokta kümesi elde edilmiş, bu nokta kümesinin yerçekimi yönündeki izdüşümünde gerçekleştirilen tekrarlamalı Hough doğru taraması ile düzlem parçalarına ulaşılmıştır. Bu işlem sonucunda bina yüzeylerinin özlü ve yoğun derinlik haritası düzlem parçaları ile ifade edilebilmektedir. Düzlem bulma işlemi birbirini takip eden sahneler üzerinde tekrarlanarak tüm model düzlem birleştirme yöntemi ile elde edilmiştir. Ataletsel duyaç tümleştirmesi ile, kamera konum kestirimi, 3B geriçatma ve düzlemsel modelleme aşamaları iyileştirilmiştir. Kamera konum kestirimi için, görsel ve ataletsel duyaç ölçümlerinin bir dolaylı-geribeslemeli Kalman süzgeci ile bütünleştirilmesi önerilmiştir. Bu sayede küresel konumlandırma sisteminin olmadığı durumlarda da güvenilir ve coğrafi referanslı kamera konum kestirimleri elde edilmesi sağlanmıştır. Ataletsel sensörler ayrıca, 3B geriçatma işleminde hatalı görsel işaretlerin ayıklanması, düzlem çıkarma işleminde yerçekimi yönünün bulunması, 3B modelleme işleminde yerden yükseklik bilgisi kullanılarak istenmeyen nesnelerin çılarılması adımlarında kullanılmıştır. Görsel-ataletsel duyaç tümleştirme ve şehirsel alanların kendilerine has özelliklerinin kullanılması temellerine dayanan yöntemin, geniş şehirsel alan modellemesi için klasik yöntemlere oranla daha tutarlı ve gerçek zamanda çalışabilecek sonuçlar ürettiği gösterilmiştir.

Anahtar Kelimeler: 3B Nokta Kümeleri, Düzlemsel Bina Modelleme, Görsel-Ataletsel Duyaç Bütünleştirmesi, Kalman Filtresi

Anneme, Babama ve Tuba'ya

ACKNOWLEDGEMENTS

I would like to express my gratitude to the following people for sharing my burden, and any others who were left out in this by no means complete list.

First and foremost, I would like to express my sincere appreciation to my supervisor Prof. Dr. Aydın Alatan for his continued guidance, also patience and support in every phase and aspect of my Ph.D. study. He has set the perfect example of being an inspiring researcher and teacher for me. He also provided counsel, and assistance that greatly enhanced my studies and know-how. He has been very influential in shaping this work and will definitely continue to light my way in my future career.

I am fortunate to have had Afşar Saranlı and Uluç Saranlı on my thesis committee. Their feedback helped me gain new understanding about my own research.

I would like to thank my dear colleagues Burak Seymen, Onur Güner, Alper Öztürk and Volkan Nalbantoğlu for their support from the beginning to the end of my Ph.D. study. I am also grateful to my managers Dr. Murat Eren and M. Naci Orhan for their continued indulgence and patience in this long path. ASELSAN Inc. who supported this work is also greatly acknowledged.

Although it feels strange to formally thank my parents as I take their love and support for granted, I would still like to express my gratitude to them: Ömer and Hava Sırtkaya, I'm dedicating this work to you as a futile attempt to express my indebtedness. My sisters Melike and Esra, my parents-in-law, my sister-in-law Pelin and her sweet family, and all the members of my large family made life more bearable for me during the hard times of this Ph.D. period. I want to thank all of them for their support and encouragement.

Finally, I would like to express my most heartfelt gratitude to my wife, Tuba, who has worked every bit as hard as I have for my education. I could not have done it without her.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGEMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xvi

LIST OF FIGURES xvii

LIST OF ABBREVIATIONS xxi

CHAPTERS

- 1 INTRODUCTION 1
  - 1.1 Scope of the Thesis 3
  - 1.2 Related Work 4
    - 1.2.1 Reconstruction from Ground-Level Views 6
    - 1.2.2 Reconstruction from LIDAR 7
    - 1.2.3 Reconstruction from Aerial and Satellite Views 7
  - 1.3 Contributions 8
  - 1.4 Organization 8

2 SPARSE 3D RECONSTRUCTION AND EXTRACTION OF CAMERA POSE FROM CALIBRATED IMAGES 11

2.1 Image Formation and Camera Geometry 12

2.2 Feature Detection and Matching 15

2.3 Epipolar Geometry Estimation 16

2.4 Reconstruction of Scene Structure 17

2.5 Visual Odometry 17

2.5.1 Visual Odometry Error Characteristics 19

2.6 Conclusion 23

3 3D DENSE RECONSTRUCTION OF URBAN ENVIRONMENTS 25

3.1 Plane-Sweeping Stereo with Multiple Sweeping Directions 26

3.1.1 Identifying Plane Priors 27

3.1.2 Plane Hypotheses Generation by Sampling of Plane Priors 31

3.1.3 Plane Sweeping Stereo 31

3.1.4 Depth Estimation 32

3.1.5 Experimental Results 32

3.2 3D Modeling with Plane Association to Superpixels 37

3.2.1 SLIC Superpixel 39

3.2.2 Plane Association 39

3.2.3 Experiments 40

3.3 Large Scale Urban Modeling with Planes 44

3.3.1 Experimental Results 46

3.4 Conclusion 53

4 VISUALLY AIDED INERTIAL POSE ESTIMATION 55

4.1 Inertial Navigation 57

4.1.1 Kinematic Inertial Navigation Equations 57

4.2 Discrete Kalman Filter 59

4.2.1 Direct vs Indirect Kalman Filter 61

4.2.2 Feedforward vs Feedback Kalman Filter 61

4.3 Loosely Coupled Visual Inertial Integration 63

4.3.1 Total System Error Model 64

4.3.2 Initialization of the Filter 64

4.3.3 Navigation Update 66

4.3.4 Kalman Time Update 66

4.3.5 Visual Odometry Measurement Model 68

4.3.5.1 Accumulated Delta Position Measurement Models 69

4.3.5.2 Attitude Change Measurement Model 70

4.3.5.3 Delayed-State Kalman Filtering 71

4.3.6 Kalman Measurement Update 72

4.3.7 Kalman Error State Feedback 73

4.3.8 Experimental Results 73

4.4 Tightly Coupled Visual Inertial Integration 86

4.4.1 Feature Location Based Methods 86

4.4.2 Epipolar Geometry Based Methods 87

4.4.3 Homography Based Methods 87

4.4.4 Proposed Feature Based Visual-Inertial Integration 89

4.4.4.1 Measurement Model 89

4.4.4.2 Experimental Results 94

4.5 Conclusion 100

5 CONCLUSION 103

5.1 Summary 103

5.2 Conclusions 107

5.3 Future Directions 107

REFERENCES 109

APPENDICES

A FRAME AND COORDINATE SYSTEM DEFINITIONS 115

A.1 Reference Frames 115

A.1.1 Earth Centered Inertial Frame (i-frame) 115

A.1.2 Earth Centered Earth-Fixed Frame (e-frame) 115

A.1.3 Navigation Frame (n-frame) 116

A.1.4 Body Frame (b-frame) 116

A.1.5 Vehicle Frame (v-frame) 116

A.2 Coordinate Systems 116

A.2.1 Earth Centered Inertial Coordinate System 116

A.2.2 Earth Centered Earth Fixed Coordinate System 116

A.2.3 Navigation Coordinate System 117

A.2.4 Body Coordinate System 117

A.2.5 Coordinate Transformation 117

B CAMERA INERTIAL SENSOR CALIBRATION 119

B.1 Coordinate System Definitions 120

B.2 Estimation of Calibration Parameters 121

B.3 Experimental Results 124

B.3.1 Simulated Data 124

CURRICULUM VITAE 127

LIST OF TABLES

TABLES

Table 2.1 Visual Odometry OR Errors for Different Algorithms [1] 21

Table 3.1 The global positions of initial frames for sequences in Figure 3.28 and Figure 3.30. 48

Table 4.1 Visual Odometry OR Errors for different algorithms [1], including the proposed methods. 101

LIST OF FIGURES

FIGURES

Figure 1.1 Input data sources for urban reconstruction, figure courtesy of Musialski [2]. 5

Figure 1.2 Overview of urban reconstruction approaches. The methods are grouped according to their outcome. Note that this is a schematic illustration, and in practice many solutions cannot be strictly classified into a particular bin, figure courtesy of Musialski [2]. 5

Figure 1.3 Data Flow between chapters. 10

Figure 2.1 The pinhole camera imaging model 13

Figure 2.2 The Geometry of a Pinhole Camera 13

Figure 2.3 Epipolar Geometry of Two Views 17

Figure 2.4 A block diagram showing the main components of a VO system, figure courtesy of [3] 18

Figure 2.5 Visual odometry error characteristics variation against velocity and path length, [1]. 22

Figure 3.1 Two images from an urban environment with SIFT descriptors (after epipolar constraint check) 26

Figure 3.2 Pictorial depiction of a classical urban scene, ground and building facade planes and their normals. 26

Figure 3.3 (a) 3D point cloud from SfM, (b) 2D projection of the sparse 3D points in the direction of gravity (c) Minimum entropy rotation of the 2D points based on x-y histogram 28

Figure 3.4 Two images from the EPFL dataset [4] that have multiple facades (castleP30-0025 and 0026) 29

Figure 3.5 (a) 3D point cloud from SfM, (b) 2D projection of the sparse 3D points in the direction of gravity, (c) Identified lines with iterative Hough Transform 30

Figure 3.6 Two images from an urban environment from the EPFL dataset [4] (castleP30-0025 and 0026) 33

Figure 3.7 Plane sweeping stereo with planes of minimum entropy (no angle or depth sampling) and without window filtering for the images in Figure 3.6. 34

Figure 3.8 Plane sweeping with depth sampling and window filtering for the images in Figure 3.6; window size = 15, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments 34

Figure 3.9 Plane sweeping with angle sampling and window filtering for the images in Figure 3.6; window size = 15, number of angle samples around minimum entropy plane = -5 to +5 degrees with 1 degree increments 35

Figure 3.10 Colored depth representation as result of plane sweeping with depth sampling for the images in Figure 3.6. 35

Figure 3.11 Plane sweeping with depth and angle sampling and window filtering for the images in Figure 3.4; window size = 15, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments, number of angle samples around prior plane direction = -5 to +5 degrees with 1 degree increments 36

Figure 3.12 Plane sweeping with depth and angle sampling and window filtering for the images in Figure 3.4; window size = 50, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments, number of angle samples around prior plane direction = -5 to +5 degrees with 1 degree increments 36

Figure 3.13 Plane sweeping with depth and angle sampling and window filtering for the images in Figure 3.4; window size = 100, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments, number of angle samples around prior plane direction = -5 to +5 degrees with 1 degree increments 37

Figure 3.14 Labeled sparse 3d points with plane priors 38

Figure 3.15 Simultaneous Linear Iterative Clustering demonstration [5] 39

Figure 3.16 Simultaneous Linear Iterative Clustering for the urban scene 41

Figure 3.17 Superpixels associated to plane priors with neighborhood filling 41

Figure 3.18 Textured dense depth map obtained from SLIC plane patches 42

Figure 3.19 3D Reconstruction Results of algorithms in sections 3.1 and 3.2 43

Figure 3.20 3D Reconstruction Results of algorithms in sections 3.1 and 3.2 43

Figure 3.21 A typical urban scene from KITTI benchmark [1] 44

Figure 3.22 Urban 3D Reconstruction Scheme 46

Figure 3.23 The starting scene of the city sequence [1] 47

Figure 3.24 The accumulated 3D point cloud of the whole sequence 48

Figure 3.25 Road+Vehicle - Building Discrimination in the 3D point cloud based on height data. 48

Figure 3.26 Vertical projection of the accumulated 3D point cloud that belong to the building facades and extracted lines 49

Figure 3.27 Number of Features throughout the sequence, average number is 250 . . . 49

Figure 3.28 Bounding box modeling of the buildings 50

Figure 3.29 An aerial view of the reconstructed sequence of Figure 3.28. 50

Figure 3.30 3D Reconstruction results for another sequence in the dataset 51

Figure 3.31 An aerial view of the reconstructed sequence of Figure 3.30. The yellow box indicates the area with trees and bushes in one-side of the road as shown in Figure 3.32. 52

Figure 3.32 A scene from the sequence of Figure 3.30. Right side of the road is covered with trees and bushes. 52

Figure 4.1 Block Diagram Representation of the Navigation Frame Mechanization of INS 59

Figure 4.2 A complete picture of the operation of the Discrete Kalman Filter [6] . . . 60

Figure 4.3 Indirect Feedforward Kalman Filter structure 62

Figure 4.4 Indirect Feedback Kalman Filter structure 62

Figure 4.5 Loosely Coupled Indirect Feedback Kalman Filter structure 63

Figure 4.6 2D Trajectory plotted on Google Earth data - Ground Truth (blue) vs Hybrid Stereo-VO/IMU (red). 75

Figure 4.7 (a) 2D Position Comparison between ground truth INS-GPS, Stereo VO, IMU Only and Hybrid Stereo-VO/IMU data, (b) IMU Only Data is removed for better scope 76

Figure 4.8 2D Position Comparison between ground truth INS-GPS, Monocular VO and Hybrid Mono-VO/IMU data. 77

Figure 4.9 2D Position Comparison between ground truth INS-GPS, Monocular VO and Hybrid Mono-VO/IMU data. 78

Figure 4.10 The translational error against distance travelled. 79

Figure 4.11 (a) 2D Speed Comparison between ground truth INS-GPS, Stereo-VO, Monocular-VO, Hybrid Stereo-VO/IMU and Hybrid Mono-VO/IMU algorithms., (b) 2D Speed Errors for Hybrid Stereo-VO/IMU Mono-VO/IMU algorithms. . . . 81

Figure 4.12 Attitude Error in terms of Euler angles (Roll, Pitch, Yaw) 82

Figure 4.13 Kalman Filter Gyroscope Bias Estimates 83

Figure 4.14 Kalman Filter Accelerometer Bias Estimates 84

Figure 4.15 Scale factor estimate for stereo and monocular VO in the integration filter. 85

Figure 4.16 Homography induced by the plane $\boldsymbol{\pi}$. 88

Figure 4.17 Pictorial depiction of the tightly coupled visual-inertial filter measurement model. 90

Figure 4.18 (a)&(b) 3D Feature Reconstruction/2D Projection output for the first frame in the sequence , (c)&(d) 3D Feature Reconstruction/2D Projection output after 300 seconds of free-inertial run. 95

Figure 4.19 RMS re-projection error in pixels, tightly coupled vs free-inertial pose estimation 96

Figure 4.20 2D Position Comparison between ground truth INS-GPS, LC Mono-VO/IMU and TC Visual/IMU data 97

Figure 4.21 2D Position Error Comparison between all proposed methods and visual only pose estimates 98

Figure 4.22 Number of feature updates per Kalman cycle, average number is 90 99

Figure 5.1 Processing timeline of the overall algorithm 106

Figure B.1 Coordinate System Definitions for IMU Camera Calibration 121

Figure B.2 Simulated 3D route of the camera-IMU sensor rig. 126

Figure B.3 Camera to IMU boresight estimates. True values are depicted with red lines. 126

LIST OF ABBREVIATIONS

BA	Bundle Adjustment
DOF	Degree of Freedom
ECEF	Earth Centered Earth Fixed Frame
EKF	Extended Kalman Filter
GIS	Geographical Information System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
INS	Inertial Navigation Unit
LC	Loosely Coupled
LIDAR	Light Detection and Ranging
MEMS	Micro Electro-Mechanical Systems
OR	Offset Ratio
RANSAC	Random Sample Consensus
SAR	Synthetic Aperture Radar
SfM	Structure from Motion
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SLIC	Simple Linear Iterative Clustering
TC	Tightly Coupled
VO	Visual Odometry
ZUPT	Zero Velocity Update

CHAPTER 1

INTRODUCTION

Recovering the 3D structure of a scene from 2D images is an old and fundamental problem which attracted engineers, mathematicians, physicists, photographers, painters and many others from different disciplines for decades [7]. The solution to this problem leads to address more concrete problems such as map building, situational awareness, autonomous navigation, 3D media generation etc. The demand for 3D visualization grows rapidly as the camera/display technology and hardware capacities enhance. For example, introduction of the aerial maps of cities was a milestone in consumer navigation market but it did not take too long for them to lose their popularity to street level visualizations. The request for more detailed, more realistic, easy to access maps is still alive. In addition, the demand for 3D maps of urban environments is growing for many applications including urban planning, historic preservation, virtual tourism, flight simulation, traffic simulation, self-driving vehicles, movie industry and several other areas in commercial, industrial and military domains. The issue is also at the focus of research in image processing, computer vision, photogrammetry and robotics societies. Commercial initiatives such as Google Earth and Microsoft Bing Maps provide 3D models of popular cities around the world. These models were first constructed from manual fusion of aerial scenes with GIS data, therefore they were low detailed and lacked ground level realism as described in [8]. In recent years, ground level visualizations are also used in modeling [9, 10, 4]. Google Street View and Microsoft Streetside provide high resolution panoramic views with limited in-picture navigation capability. All these efforts address the problem from a map provider perspective, and their scope is building more accurate and detailed 3D models. In addition to this perspective, the commitment of the industry and governments to autonomous navigation of vehicles such as planes, helicopters and cars, have necessitated a user perspective to the 3D modeling problem. From the user perspective, 3D modeling becomes a local problem and the attention is on localization and robust navigation throughout the constructed model. Furthermore, real-time applicability becomes more important in this new perspective, since the mapping and pose estimation should be available instantaneously for decision making. It is required to enhance the current 3D modeling algorithms and develop novel algorithms to fulfill the requirements of this perspective.

Actually, the 3D structure estimation and modeling from vision is a well examined and mature research area. Approaches such as stereo reconstruction and structure from motion reached their theoretical limits. However, most of the work done in this area applies to only limited small scales, and only partial reconstruction of the environment is possible as described in [11]. Standard multi-view dense stereo methods, reviewed in [12], may lack robustness or produce inaccurate and incomplete models for many specific scenes as demonstrated on an urban scene [13]. Varying lighting conditions, the scale of the environment, and the complexity of outdoor scenes with obstructions, trees and glass surfaces pose enormous challenges to purely vision-based methods. Also, the drift in pose estimation makes it almost impossible to work with long sequences [10], therefore build a consistent model. As a result, building a model of a large scale urban environment brings its own problems with new solution approaches. This makes the subject still an open and attractive problem.

The solution approaches to 3D urban modeling are generally shaped by the sensors that are used in the reconstruction system. Inertial Sensors, GPS, magnetic attitude sensors, LIDAR Scanning Systems, laser range finders, vehicle odometer and many others are used to extend existing algorithms to meet the robustness and reliability requirements to operate in harsh and large scale urban environments.

At the moment, there is a lot of activity around ground LIDAR based approaches in the research world [14, 15, 16]. The technology has been widely accepted as fast and reliable means of capturing data from commercial point of view. However, LIDAR or other light emitting-detecting based sensors are expensive, require more power, and it may be cumbersome to process the large amount of data captured by these sensors in real-time [17]. Also, the scanning field of view is generally not enough to take measurements from upper levels of tall buildings. As a final remark, especially for military applications, using active sensors emitting any kind of signal may not be feasible in a region of armed conflict.

Another general approach to 3D urban modeling is to use aerial data (stereo camera, monocular camera, LIDAR) which provide wide area coverage. However, the resulting models often lack visual realism when viewed from ground level since they can only give building footprints and roof heights due to bad viewing angles. To achieve high quality ground-level visualization one needs to capture data from the ground.

In conjunction with the advances in map technology, the navigation requirements for mobile systems also gets tighter. It was enough to pass Atlantic with a compass and knots in the era of Columbus. Several miles of position error was not considerable, when the human interaction was in progress at that time. The advances in sensor technology and algorithms has made it possible to have the position accuracy at centimeter level for a modern navigation device. Gyroscopes, accelerometers, satellites, odometers, barometers and several other modern instruments serve for better navigation accuracy. They make it possible to fly airplanes safely over long distances, plow the fields accurately for productive agriculture, land astronomical observers to small meteors that are millions of kilometers away from the earth, sail a yacht through a rocky sea, drive a car autonomously through a city, and advance many other applications that make life easier. However, some of these technologies are not applicable to difficult and complex environments, such as urban and indoor, in which the mobile platforms are increasingly forced to operate. In such environments navigation using modern satellite navigation systems is often not possible. Advances in computer vision have made the camera a plausible sensor for map building and navigation purposes especially in these environments.

The close relation between 3D map building and navigating through this map, specifically for urban environments, is the essence of this dissertation. Camera and inertial sensors are chosen as the sensor bed to solve this bilateral problem. GPS and LIDAR, which are commonly used in this type of large scale reconstruction, are not integrated to the proposed system on purpose to investigate the possibility of successful large scale reconstruction with a passive and easy-to-access sensor bed. The reconstruction of accurate 3D models from long video sequences with emphasis on urban environments at ground level is investigated together with reliable camera pose estimation. To build a geo-registered model and to estimate the pose of the camera reliably, inertial sensors are integrated to the vision based algorithms. The visual-inertial integration also eliminates the need for the computationally costly algorithms such as bundle adjustment, segmentation, moving object detection, and makes it possible to build real-time algorithms.

This problem definition is very close to the well known Simultaneous Localization and Mapping (SLAM) problem of Robotics community. The basic distinction of SLAM is it's sparse nature of map building and the architecture of the stochastic framework for the measurement integrations. Still the SLAM literature is very attractive since it addresses the same problem from a different perspective.

1.1 Scope of the Thesis

The main aim of this thesis is to build a complete system for street-level, autonomous, real-time and geo-registered 3D reconstruction and modeling of large scale urban environments using a camera and inertial measurement unit.

3D point cloud formation, dense reconstruction by planar modeling, reliable camera pose estimation by combined visual-inertial navigation and camera-IMU calibration problems are examined in detail for this purpose. Individual steps such as visual feature extraction and matching, Structure-from-Motion, epipolar geometry estimation, triangulation, Hough transform, Kalman filter are exploited by the proposed algorithms.

1.2 Related Work

Building a 3D model of an urban environment is a multi-disciplinary problem which has solution modalities in computer vision, computer graphics, robotics, photogrammetry and even in navigation societies. The scope of the problem changes tremendously with regards to the requirements. Street level visualization, aerial visualization, texturing, view-point invariant map generation, coarse-to-fine detailed mapping, real-time implementation are among the most demanding requirements. The first attempts to construct a 3D model of an urban environment using vision sensors started with extensions of classical Structure from Motion and similar computer vision algorithms. However, these attempts were limited to controlled and small scale environments which were not the right scheme for urban reconstruction. The demand (from the industry) for 3D models of big cities forced the research community to tackle the problem from a wider perspective. The sensor platforms, solution methodologies, representation tools enhanced thereafter. Therefore, it became slightly complicated to cluster the approaches into rigorous groups.

The approaches can be divided, by considering user intervention, as automatic, semi-automatic or manual. The state of the arts methods that proved to be useful and academically respectful are automatic methods that do not need user intervention. Manual 3D processing of images is time consuming and requires the expertise of highly qualified personnel. They mainly bear importance from an industrial point of view. The user drives the reconstruction by selecting and grouping the important point or line features, and the computer refines their locations based on the images. Semi-automated methods as in [18] [19] that support user intervention at keypoints to meet the challenge of constructing complete and extended urban models from aerial and ground images fall in the first era of urban reconstruction history. The Facade system presented by Debevec et al. [20] is a user-assisted or semiautomatic approach. Users indicate key edges in several images which are then formed into blocks. Edge positions and the 3D structure are refined automatically. The user can also specify relative position and symmetry constraints to further aid the reconstruction. Another semi-automatic system is presented by Xiao et al. [21]. First, the urban scene is fully automatically reconstructed as a collection of rectilinear blocks. Then, several simple tools are provided to allow the user to correct the mistakes made by the system.

The solution approaches are also shaped by the sensors as described in the survey of Hu et al. [8]. Monocular and stereo cameras, LIDAR Scanning Systems, laser range finders, inertial sensors, GPS, magnetic attitude sensors, vehicle odometer and many others are used to extend algorithms for robustness and reliability. Active (LIDAR) and passive (camera) sensors establish the main distinction between the approaches.

In another aspect, the approaches might categorized based on the platform of the data taken for reconstruction:

- Reconstruction from Ground-Level Views

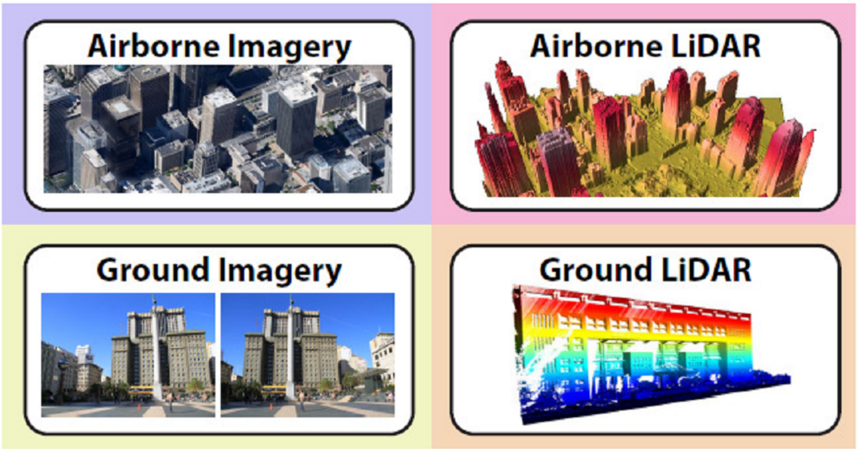


Figure 1.1: Input data sources for urban reconstruction, figure courtesy of Musialski [2].

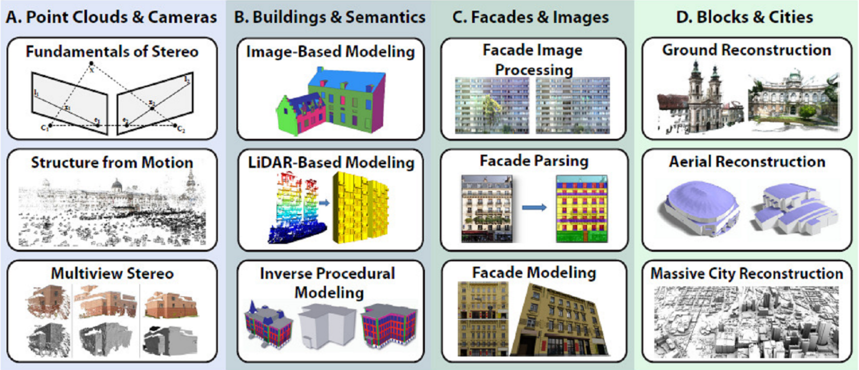


Figure 1.2: Overview of urban reconstruction approaches. The methods are grouped according to their outcome. Note that this is a schematic illustration, and in practice many solutions cannot be strictly classified into a particular bin, figure courtesy of Musialski [2].

- Reconstruction from Aerial Views
- Reconstruction from Satellite Views
- Reconstruction by fusion of Ground Views with Aerial and/or Satellite Views

The input data sources for urban reconstruction are depicted in Figure 1.1. Musialski et al. [2] proposed an output-based ordering of the urban reconstruction approaches. This ordering helps to sequentially explain important concepts of the field, building one on top of another. Overview of urban reconstruction approaches in terms of output types are depicted in Figure 1.2.

A valid clustering might also be on methodology and output of the approach:

- Sparse Reconstruction based on SfM

- Plane Sweeping Stereo [10]
- Height Map Modeling [22]
- Semantic Approaches [9]
- Volumetric Approaches [23]

Any approach might fall into multiple categories, since a complete reconstruction requires integration of different sensors, platforms and modalities.

1.2.1 Reconstruction from Ground-Level Views

The scope of this dissertation is on automatic, street level and vision based modeling of urban environments. There are numerous approaches that may fall into this scope. The body of literature relevant to this dissertation will be reviewed here. MIT City Scanning Project [24] was one of the first complete automatic systems on reconstruction of urban areas. Calibrated hemispherical images of buildings are used to extract planes corresponding to structured surfaces of the buildings, which are then textured and geometrically refined using pattern matching and computer vision methods. Inertial sensors, GPS and optical encoders are used to aid pose estimation.

Position and attitude sensors are natural choices to make the reconstruction system fully automatic, since pose drift is the main problem in large scale environments. Pollefeys et al. [10] developed a fully automatic method that uses GPS and inertial sensors together with a camera rig. They used a loosely coupled INS/GPS integration to geo-reference the model and ease the pose estimation. Smoothed Best Estimated Trajectory (SBET) of the camera is utilized as a post-processing step. Their reconstruction framework is a two-step process. First, sparse reconstruction is achieved from classical from motion techniques. Then, dense reconstruction is held by multi-view stereo techniques. An adapted version of plane sweeping technique is used in this stage. Depth map fusion is utilized at the end for reliable textured surface reconstruction. Their approach is based on ground-level visualizations and they use passive sensors only. Their work is mature and covers all steps needed for a complete, geo-referenced, textured and large scale reconstruction. Variations of their approach, piecewise planar and non-planar reconstruction [25] [17], height-map modeling [22] [26], continue to dominate the research in this area.

It is common to use heuristics about the organized structure of the urban areas. In [27], urban and indoor areas are modeled using a Manhattan world stereo on the assumption that all planes are orthogonal. Micusik and Kosecka [13] proposed another method that uses superpixels as planar patches to model building facades. Superpixels helped to implement more efficient algorithms compared to pixel based methods. In [28] a ground level reconstruction framework is given. They approach the problem from a higher level semantic view. They suggest utilizing the initial knowledge on the environment and claim that 3D reconstruction can become easier and more accurate when the kind of object which is being reconstructed is known. They also suggest combining different algorithms to overcome the weakness of a single algorithm like combination of early processing levels such as stereo with higher semantic levels like object class recognition. This is called cognitive loops. GPS and vehicle odometer is used to compensate for drift of Structure from Motion (SFM) algorithms and for globalizing the coordinate system.

1.2.2 Reconstruction from LIDAR

Light detection and ranging (LIDAR) is an alternative to purely image-based methods. LIDAR measures distance by emitting laser pulses and measuring the time between transmission and detection of the return signal. LIDAR is typically more robust and accurate than stereo. Generally, there are two main types of this class of data: those acquired by ground based devices (terrestrial LIDAR), and those captured from the air (aerial LIDAR). In earlier works, Stamos and Allen [16] present a systematic approach to the problem of photorealistic 3-D model acquisition from the combination of range and image sensing. Their method is based on fitting planar polygons into pre-clustered point-clouds. Christian Fruh and Avideh Zakhor combined a digital camera, a velocimeter, a heading sensor and 2D laser scanners and put them into a truck to model downtown Berkeley [15] [29]. One of the laser scanners is used for relative pose estimation and the other measures the building structures. Other sensors are used to increase the accuracy. For geo-referencing a dead-reckoning method is utilized by successively adding relative position estimates derived from horizontal laser scan matching. Heading sensor and odometer is used for consistency check. Christian Fruh and Avideh Zakhor merge aerial views to their ground based models of in [30]. Using the Digital Surface Model obtained from the airborne laser scans, they localize the acquisition vehicle and register the ground-based detailed structure to the airborne model by means of Monte Carlo Localization method.

Recently, Vanegas et al. [14] proposed an approach for the reconstruction of buildings from 3d point clouds with the assumption of Manhattan World building geometry. Their system detects and classifies features in the data and organizes them into a connected set of clusters from which a volumetric model description is extracted. The Manhattan World assumption has been successfully used by several urban reconstruction approaches.

1.2.3 Reconstruction from Aerial and Satellite Views

City reconstruction from aerial imagery has long been one of the problems studied in photogrammetry. Aerial and satellite data (image, laser scan, range data etc.) provide a rapid and efficient method for the coverage of a wide area. Therefore there exists a variety of approaches to creating 3D models from these views. 3D modeling from satellite images is presented in

[31] and [32]. Different from the aerial views, satellite views have low resolution. Prior knowledge concerning urban structures is utilized in the satellite methods in order to face the difference of data quality.

1.3 Contributions

Main contributions of this thesis to the existing body of literature can be summarized as follows:

Loosely Coupled Visual/Inertial Sensor Fusion:

A novel measurement model is proposed for visual odometry/inertial sensor integration in an indirect feedback Kalman Filter framework in Section 4.3. The proposed solution for visual-inertial integration does not assume a Gaussian noise model for visual odometry error as opposed to current approaches in literature and tackles the non-linear characteristics of this error. The proposed solution is able to model visual odometry error sources that are not addressed in the literature, such as scale factor and lever-arm.

Tightly Coupled Visual/Inertial Sensor Fusion:

The state-of-the-art algorithm in tightly coupled visual/inertial integration is MSCKF (Multi-state Constrained Kalman filter). A different measurement model is proposed for this method, which eliminates the requirement for state cloning for dependence on previous states.

Visual/Inertial Sensor Fusion for Efficient Large Scale 3D Urban Reconstruction:

The main challenge of urban reconstruction is pointed out as scale in the literature. The sequences are long and the captured images are cluttered. Therefore, the proposed solution must be free of pose drift and robust to outliers to achieve a complete and consistent reconstruction. A novel method is proposed for large scale urban reconstruction which exploits the benefits of inertial sensors and their integration to vision algorithms. In the proposed method, inertial sensors are used in pose estimation, feature pre-conditioning, sparse 3D reconstruction, plane extraction, 3D modeling and geo-locational reasoning stages. To our knowledge, this type of tight integration of visual/inertial sensors is pioneering in the literature.

1.4 Organization

The organization of the thesis is as follows:

In Chapter 2, the fundamentals of Structure-from-Motion and camera pose estimation problems are presented. Image formation and camera geometry, feature detection and matching, epipolar geometry estimation, reconstruction of cameras and scene structure topics are discussed in sparse 3D reconstruction framework, mostly following definitions in Hartley and Zisserman [33]. Visual Odometry concept is presented as an alternative form of camera pose estimation.

In Chapter 3, 3D dense reconstruction is build on top of sparse 3D reconstruction. Plane-Sweeping stereo with multiple sweeping directions approach is explained following the work of Pollefeys et al. [10]. Two new approaches are introduced which utilize plane association to superpixels and 3D point cloud itself respectively, in order to model urban environments by planar patches. Inertial sensor integration to individual algorithms that lead to planar modeling is discussed in this chapter. Experimental results and comparative analysis are given at the end of the chapter.

The camera pose estimation problem is discussed In Chapter 4. The solution approaches are based on visual and inertial sensor fusion. Therefore, fundamentals of inertial navigation is introduced first. The fusion of visual and inertial sensors is presented in an Extended Kalman Filter framework. Two methods are introduced for this purpose. In the first method, visual odometry is utilized as a vision based motion sensor for the fusion algorithm. In the second method, the visual feature locations are utilized as the visual measurement and a tight correspondence is constructed between the visual and inertial sensors. Experimental results are presented on a real dataset.

Chapter 5 concludes the dissertation with a summary of the work done, a discussion of the results and pointers for future research.

The data flow between chapter is illustrated in Figure 1.3. It shall be noted that, by looking at the data flow diagram in the figure, the order of Chapter 3 and Chapter 4 might be chosen in the reverse direction. The reason for the current ordering is as follows: 3D Dense Reconstruction chapter (Chapter 3) is constructed such that pose of the camera is assumed to be known. Therefore, it is not required to place this chapter after Visually Aided Inertial Pose Estimation chapter (Chapter 4). In addition, 3D Dense Reconstruction chapter has close connection to Sparse 3D Reconstruction chapter (Chapter 2) which forms a logical order.

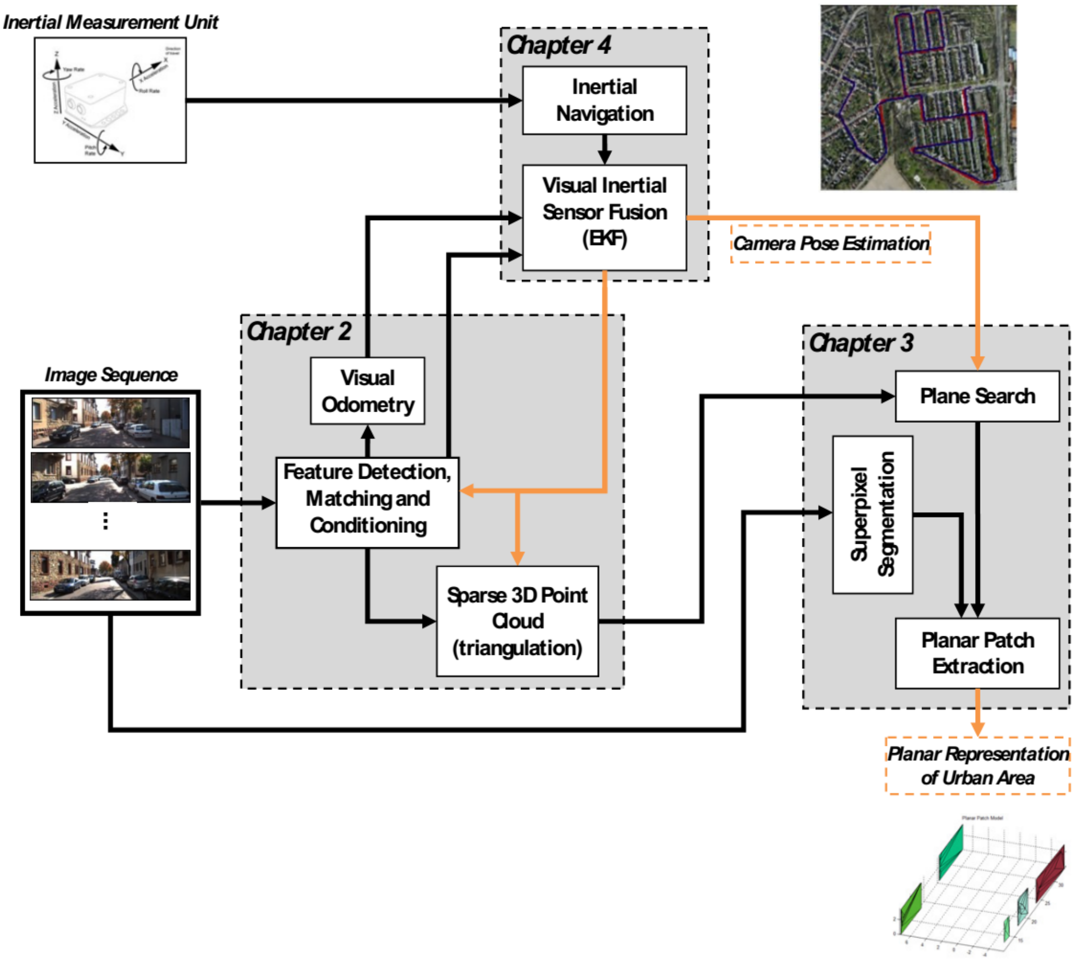

Inertial Measurement Unit
Chapter 4
Inertial Navigation
Visual Inertial Sensor Fusion (EKF)
Camera Pose Estimation
Chapter 2
Visual Odometry
Image Sequence
Feature Detection, Matching and Conditioning
Sparse 3D Point Cloud (triangulation)
Chapter 3
Plane Search
Superpixel Segmentation
Planar Patch Extraction
Planar Representation of Urban Area
Planar Patch Model

Figure 1.3: Data Flow between chapters.

CHAPTER 2

SPARSE 3D RECONSTRUCTION AND EXTRACTION OF CAMERA POSE FROM CALIBRATED IMAGES

In this chapter, the extraction of sparse 3D points from calibrated images in two view reconstruction framework is discussed. Most of the definitions follow the text in [33], therefore the reader should refer to this source for more detail.

In the presented sparse 3D reconstruction framework, the internal calibration parameters of the cameras are assumed to be known. A popular MATLAB toolbox [34] is used for intrinsic camera calibration process. The goal of sparse 3D reconstruction is to retrieve the 3D locations of some sparsely selected points and the camera poses, namely the camera projection matrices.

The normal flow of a general 3D reconstruction algorithm consists of the following steps [35]:

- Feature Detection and Matching
- Epipolar Geometry Estimation
- Reconstruction of the cameras and the structure

There are three problems to be addressed to achieve reconstruction:

- Correspondence Geometry: Given an image point $\boldsymbol{x}$ in the first view , how does it constrain the position of the corresponding point $\boldsymbol{x}'$ in the second view?
- Camera Geometry (motion): Given a set of corresponding image points $\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}'_i$ what are the camera projection matrices P and $\mathrm{P}^{'}$ for the two views such that $\boldsymbol{x} = \mathrm{P}\,\mathrm{X}$?
- Scene Geometry (structure): Given corresponding image points $\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}'_i$ and camera projection matrices P and P′, what is the position of the imaged point X in 3-space?

If the 6-DOF poses of the cameras are known together with the internal calibration parameters, it is not necessary to estimate the epipolar geometry explicitly and the 3D feature locations can directly be estimated by triangulation. The camera poses can be retrieved from external sensors (e.g. IMU, Compass, GPS) and this can lead to a reconstruction in a global reference frame that will be helpful in integrating the constructed model to a geographical information system. Nevertheless, the whole flow of reconstruction, including epipolar geometry estimation, will be discussed for the sake of completeness and for establishing a basis for the visual motion estimation. Since the process of 3D reconstruction yields the camera motion as an intermediate step, it is discussed in the context of this chapter.

2.1 Image Formation and Camera Geometry

Before describing how 3D structure can be reconstructed from images, it is important to understand how images are formed. The images that are used in this thesis are formed by a camera. In mathematical terms, a camera is a mapping between the 3D world (object space) and a 2D image space [33]. The term camera comes from the word camera obscura (Latin for 'dark chamber'), an early mechanism for projecting images. The modern camera evolved from the camera obscura. It was named due to the fact that the device consists of an enclosed hollow with an opening (aperture) that block out all light, except the light passing through it at one end, and a recording surface for capturing the light at the other end.

A mathematical model for the imaging process of the camera is needed in order to construct higher level representations from the images. In mathematical terms, a camera model is a simple transformation that relates the 3D world coordinate system and a 2D image plane. The pinhole camera model is chosen for the mathematical representation of imaging process in the context of this dissertation. The pinhole camera model describes the mathematical relationship between the coordinates of a 3D point and its projection onto the image plane of an ideal pinhole camera, where the camera aperture is described as a point and no lenses are used to focus light. The model does not include, for example, geometric distortions or blurring of unfocused objects caused by lenses and finite sized apertures. It also does not take into account that most practical cameras have only discrete image coordinates. This means that the pinhole camera model can only be used as a first order approximation of the mapping from a 3D scene to a 2D image. Its validity depends on the quality of the camera and, in general, decreases from the center of the image to the edges as lens distortion effects increase.

Some of the effects that the pinhole camera model does not take into account, are compensated for, for example by applying suitable coordinate transformations on the image coordinates, and other effects are assumed to be sufficiently small to be neglected. In summary, the pinhole perspective (also called the central perspective) projection model, first proposed by Brunelleschi at the beginning of the fifteenth century, is mathematically convenient and, despite its simplicity, it often provides an acceptable approximation of the imaging process. Figure 2.1 illustrates how images are formed under the pinhole camera model. Figure 2.2 shows geometrically how a 3D point in object space $\mathbf{X} = (X_1, Y_1, Z_1)^T$ is projected onto a 2D point $\mathbf{x} = (x_1, y_1, z_1)^T$ on the image plane. By similar triangles, this projection can be

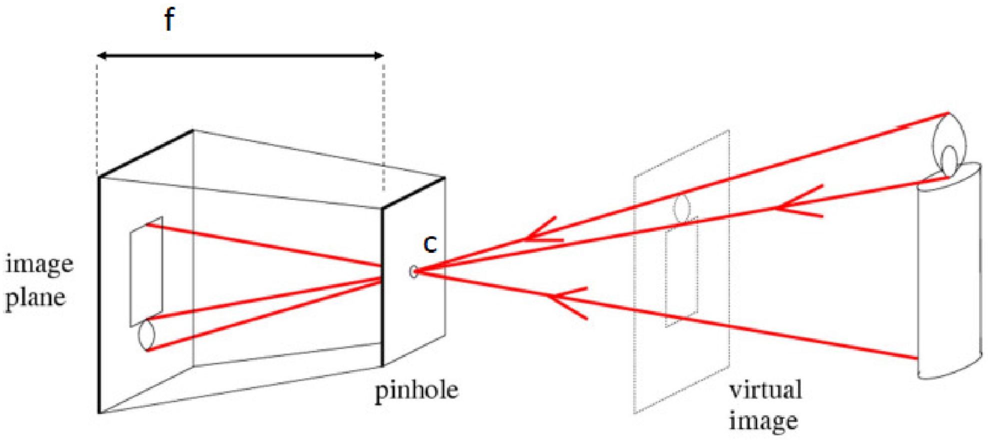


Figure 2.1: The pinhole camera imaging model

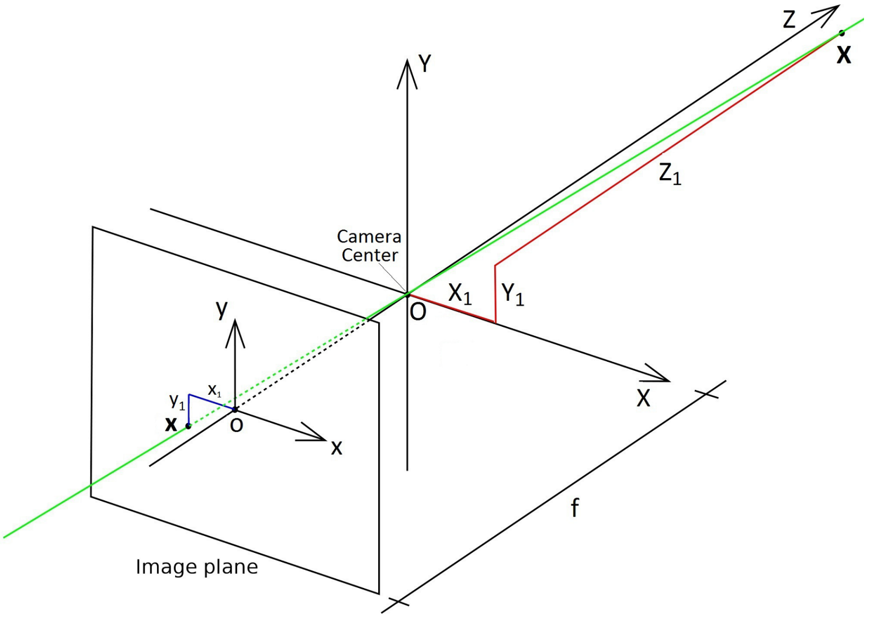


Figure 2.2: The Geometry of a Pinhole Camera

formulated by the following equation:

$$x_1 = f\frac{X_1}{Z_1} \qquad y_1 = f\frac{Y_1}{Z_1} \qquad z_1 = f \tag{2.1}$$

where f is the focal length (distance of the image plane to the pinhole). Since the image plane and the scene are on opposite sides of the pinhole, f and Z will have opposite signs (typically f is negative and Z is positive), resulting in the inverted (flipped and mirrored) image shown in the Figure 2.2. It is simpler to work with the virtual image which is not inverted and where f is positive.

Note that all the points along the viewing direction of $\boldsymbol{X}$ projects to the same point $\boldsymbol{x}$. This can also be inferred from equation 2.1, since the scaling factor cancels in the division operation. This result means that the depth information is lost under perspective projection. This projection can be expressed conveniently in homogeneous coordinates since two points are considered equal in homogeneous coordinates, if the vectors representing them are equal up to a non-zero scale factor. In particular equation 2.1 can be expressed in matrix form as follows:

$$\begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX_1 \\ fY_1 \\ Z_1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix} \tag{2.2}$$

The matrix in this expression is called the camera projection matrix, and denoted as P. The expression for P matrix in equation 2.2 is missing important details, such as principal point offset and camera pose (rotation and translation). The parameters that control how points in world coordinates map to pixel coordinates are called the camera calibration. These parameters can be grouped as intrinsic and extrinsic calibration parameters.

Intrinsic Calibration Parameters:

Intrinsic parameters describe the internal properties of the camera, namely how points on the camera's image plane are mapped to pixel coordinates in the image. In equation 2.2, it is assumed that the origin of coordinates in the image plane is at the principal point. However, points on the image plane are usually expressed in pixel coordinates where the origin is at the top left corner of the image and pixels are spaced one unit apart. Camera skew is also considered as an internal calibration parameter. The calibration matrix, K, is expressed as follows for a CCD camera:

$$\mathrm{K} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

where f_x and f_y are horizontal and vertical focal lengths in pixel coordinates, s is the skew due to the tilt of the pixels, u_0 and v_0 are the principal point offset in pixel coordinates. The intrinsic calibration parameters are obtained by a calibration process as defined in the publicly available Camera Calibration Toolbox [34].

Extrinsic Calibration Parameters:

Extrinsic parameters are defined as the camera position and orientation with respect to a global coordinate system. It is easier to trace the pose of the camera in time with respect to a global coordinate system. Pose of the camera is expressed as a 3x3 rotation matrix R, and a 3x1 translation vector $\boldsymbol{t}$ such that camera center is, $\boldsymbol{c} = \mathrm{R}^{\mathrm{T}} \cdot \boldsymbol{t}$.

Adding up the intrinsic and extrinsic calibration parameters and ignoring the radial distortion, the projection becomes the following:

$$\boldsymbol{x} = \mathrm{P} \cdot \boldsymbol{X} \qquad \mathrm{P} = \mathrm{K} \cdot [\mathrm{R}|\boldsymbol{t}] \tag{2.4}$$

where $\boldsymbol{X}$ is the 3D homogeneous feature location in global coordinates and $\boldsymbol{x}$ is its projection onto the image plane. The modern cameras have lenses to gather and focus the light. The lens of the camera projects the points in the scene nonlinearly, according to their distance from the origin of the image plane; thus, this degradation is called as radial lens distortion. The degree of lens distortion increases as the focal length decreases and this might jeopardize the linearity assumption of the pinhole model. Lens distortion shall be compensated for better representation if the effects are significant.

2.2 Feature Detection and Matching

In general, SfM based 3D reconstruction methods require identification and matching of distinguishable points, features, on images. Feature identification and matching are the most crucial steps in reconstruction, since the reliability of the latter parts depend directly on these steps. If the camera projection matrices are known, the 3D point can be obtained by intersecting the viewing rays of the corresponding points (triangulation) as shown in Figure 2.3.

Feature detection and matching is a comprehensive subject in computer vision. The details of current literature on the issue is beyond the scope of this dissertation. Nevertheless, the selected algorithms will be described briefly and the reasons for these selections will be explained in the subsequent paragraphs.

For the 3D reconstruction phase of two-view urban landscape modeling problem in Chapter 3, the feature points and their matches in and between frames are obtained by using Scale Invariant Feature Transform (SIFT) descriptors [36]. SIFT descriptors are selected for their robustness in finding feature matches between frames that have large angular and translational distance. 3D reconstruction gets more reliable when the angular and translational distance between the frames is large. The computational complexity of SIFT usually discourages its real-time utilization and restricts its usage to off-line applications as [35] emphasized. This drawback might be mitigated by decreasing the rate of SIFT in a suitable manner. For the urban reconstruction problem, reconstruction is not necessarily performed at the image rate, since consequent images contain redundant information. Instead of using every image, it might be logical to use key images that have enough rotational and translational distance [37, 38]. Another way is to construct a more efficient feature detection/matching scheme for the large-scale urban reconstruction and visual-inertial pose estimation problems, since the images are ordered and the rate (10 Hz) is high enough. A combined blob-corner detector is utilized for this purpose as described in [39]. This feature detection/matching scheme is executed on consecutive image pairs as they become available in real-time. In contrast to methods concerned with reconstructions from unordered image collections, a smooth camera trajectory is assumed, superseding computationally intense rotation and scale invariant feature descriptors like SIFT [36] and SURF [40].

2.3 Epipolar Geometry Estimation

The epipolar geometry is the intrinsic projective geometry between two views. It is independent of scene structure, and only depends on camera's internal parameters and relative pose. The fundamental matrix F encapsulates this intrinsic geometry. If a point in 3-space $\boldsymbol{X}$ is imaged as $\boldsymbol{x}$ in the first view, and $\boldsymbol{x}'$ in the second, then the image points satisfy the relation $\boldsymbol{x}'^{\mathrm{T}}\ \mathrm{F}\ \boldsymbol{x} = 0$. The fundamental matrix is independent of scene structure. However, it can be computed from correspondences of imaged scene points alone.

In this dissertation, the fundamental matrix is not estimated from the images (image correspondences). The intrinsic camera calibration parameters are pre-calculated, and the extrinsic camera parameters (rotation and translation) are estimated in the pose estimation stage that is described in Chapter 4. The fundamental matrix is encapsulated in the projection matrix that can be constructed using intrinsic and extrinsic parameters. The fundamental matrices, that are obtained from projection matrices, are used as an epipolar constraint for feature matches to increase the reliability of structure estimation. Outliers are removed by utilizing the epipolar constraint. The projection matrices constraint the location of point $\mathbf{x}'$ in Figure 2.3 if the location of point $\boldsymbol{x}$ is known. The feature matches are checked against this constraint.

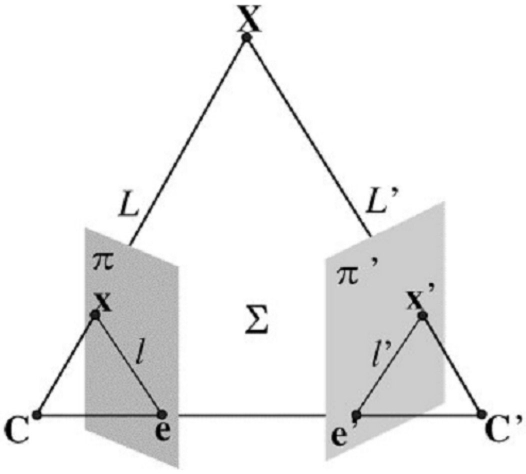


Figure 2.3: Epipolar Geometry of Two Views

2.4 Reconstruction of Scene Structure

The projective reconstruction of scene points are obtained via triangulation, in which the 3D location of a scene point is determined by intersecting two rays, each of which passes through the camera center and the image of the scene point on the image plane. In other words, given projection matrices $\{\mathrm{P}, \mathrm{P}'\}$ and its projections on images $\{\boldsymbol{x}, \boldsymbol{x}'\}$, the 3D location of the point $\boldsymbol{X}$ must be computed such that $\boldsymbol{x} = \mathrm{P}\ \boldsymbol{X}$ and $\boldsymbol{x}' = \mathrm{P}'\ \boldsymbol{X}$.

In the ideal case, for which the image projection locations and projection matrices are noiseless and perfectly available, the two rays representing back-projections intersect at a point in the 3-space; therefore, the location of the scene point can be easily determined. However, this is not the case in practical reconstruction problems, as measurement noise and computation errors are involved. Since the correspondences $\boldsymbol{x}$ and $\boldsymbol{x}'$ do not satisfy the epipolar constraint perfectly, the rays do not meet at a point and the optimal solution for the non-ideal reconstruction problem is searched as defined in [33]. An initial linear triangulation is performed by minimizing the algebraic distance. The linear triangulation is followed by a non-linear step, in which the re-projection error is minimized with Newton iterations. The result is a maximum likelihood estimation of the projective 3D points.

2.5 Visual Odometry

Visual Odometry is typically described as the calculation of position and orientation of a camera throughout a sequence. The visual odometry process is memoryless, meaning that the estimation is conducted on sequential image pairs. The term Visual Odometry was coined by Nister in his seminal paper [41]. The term was selected for its similarity to

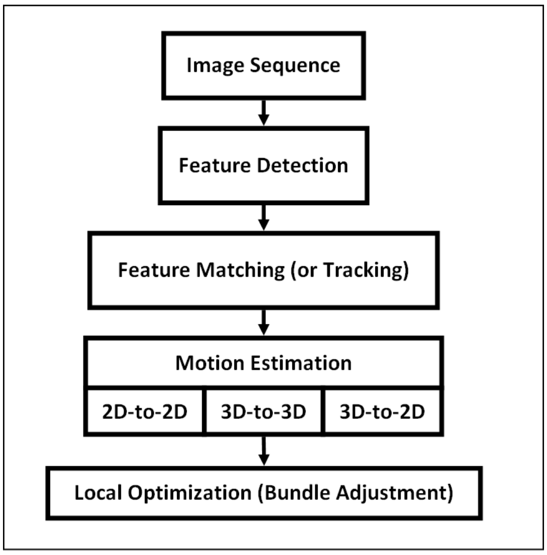


Figure 2.4: A block diagram showing the main components of a VO system, figure courtesy of [3]

wheel odometry, which incrementally estimates the motion of a vehicle by integrating the number of turns of its wheels over time. Visual Odometry is denoted as a particular case of Structure-from-Motion, in which the focus is on estimating the 3-D motion of the camera sequentially as a new frame arrives in real time.

VO can be conducted on stereo image sequences as well as monocular image sequences. In monocular case, only bearing information is available. The disadvantage is that motion can only be recovered up to a scale factor. The absolute scale can then be determined from direct measurements (e.g., measuring the size of an element in the scene, the height of the camera from ground level) or from the integration with other sensors, such as IMU, barometer, and range sensors. Over the years, monocular and stereo VOs have almost progressed as two independent lines of research, however most of the research done in VO has been produced using stereo cameras [3]. 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. It has been demonstrated that VO provides accurate trajectory estimates with relative position error ranging from 0.1 to 2 percent of distance travelled [3] under certain conditions. This capability makes VO an interesting supplement to navigation algorithms. In [3], the visual odometry process is summarized as given by Figure 2.4.

The motion estimation approach depends on whether the feature correspondences are specified in two or three dimensions. For a 2D-to-2D motion estimation, the Essential matrix is estimated first from the epipolar constraint and the rotation and translation of the camera is extracted from this matrix. However, the scale ambiguity is a major problem in this type of estimation. Only relative scale between successive frames can be estimated. The 3D-to-3D motion estimation operates on stereo vision. The solution can be formulated as the transformation between successive camera coordinates that minimizes L2 distance between the two 3D feature set. To compute the transformation, it is also possible to avoid the triangulation of the 3D points in the stereo camera and use trifocal or quadrifocal constraints instead. As pointed out by Nister et al. [41], motion estimation from 3-D-to-2-D correspondences is more accurate than from 3D-to-3D correspondences, since it minimizes the image reprojection error.

In this dissertation, both monocular and stereo VO are utilized to support inertial navigation in a Loosely Coupled Extended Kalman Filter framework. The work of Geiger et al. [42], [39] is followed to get reliable odometry results. Their stereo-based VO scheme proved to be useful in urban environments. They use a Gauss-Newton optimization technique to iteratively minimize a cost function which includes the reprojection error of the previously estimated 3D point cloud onto image planes. The robustness is achieved using RANSAC. This approach can be classified as a 3D-to-2D motion estimation algorithm. The monocular VO algorithm uses the height of the camera from the ground for scale estimation.

2.5.1 Visual Odometry Error Characteristics

The error characteristic of visual odometry should be established well in order to construct a visual odometry based navigation scheme in large scale environments, especially when the VO is subject to be integrated with other navigation systems (e.g. fusion of VO with inertial sensors). In general, visual odometry error is denoted as an offset ratio [41] [3], i.e. the ratio of the final drift value to the traveled distance. The offset ratio is rough and does not represent the heterogeneous character of the error. As a new and promising sensor, visual odometry needs a methodology for systematic and comparative analysis of its drift, in order to quantify the performance of various algorithms. Drift should not increase linearly with the distance traveled. Moreover, running the same algorithms on the same dataset repeatedly should produce quite different ORs. The reason is due to the fact that drift is a random process, and it should not always increase, but sometimes it decreases at some places, as errors in different motion vectors compensate each other to some extent. Thus, using end-point values (the final drift values, and the final traveled distances) is incapable to model the whole random process. In addition, there may be occasions for which the visual odometry process totally fails. These erroneous conditions occur due to unreliable feature matches especially when the camera is directed to the sky or there are independently moving objects in the scene.

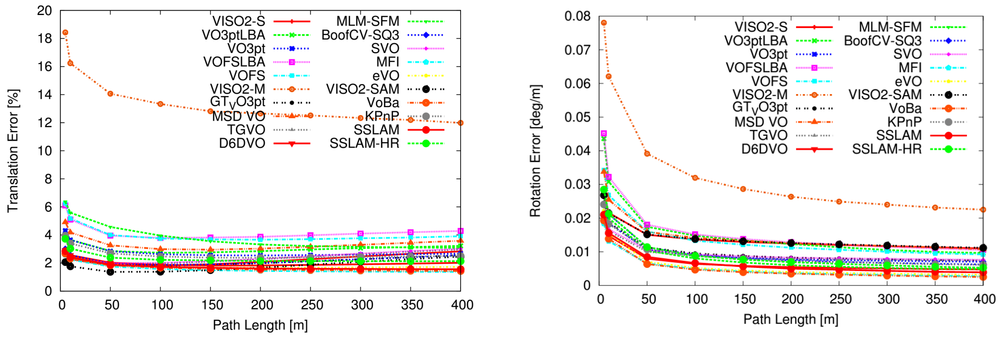
The VO error does not also conform to the traditional model of ’true measurement+Gaussian noise’, which is taken as granted in most of the fusion filters. In addition, unavoidable nonlinear characteristics of visual measurements need to be treated within the fusion filter frameworks. Regardless of the selected algorithm, VO depends on feature matches between the sequential images. Spurious matches (outliers) are the main cause of errors for most of the time. Removing outliers in early steps of VO leads a plausible error behavior at the output. In general, VO algorithms utilize RANSAC [43] as the outlier rejection scheme [41] [39] [44]. Alternative schemes, such as mean-shift based rejection [45], parallax-rigidity based rejection [46] try to improve results over RANSAC. In [47], the error caused by outliers is modeled, instead of rejection, to associate an accurate uncertainty at the output. In their approach, a convex combination model is proposed, in which the inlier error is modeled as a Gaussian distribution and outlier is modeled as a uniform distribution. Jiang et al. [48] modeled the visual odometry error as a composition of an unbounded deterministic part with unknown constant parameters, and a first-order Gauss-Markov process. Being mathematically correct, these attempts lack complete representation of the underlying dynamics that cause the error at the output of VO algorithms. For instance, error characteristic changes, caused by scene conditions, that effect feature inlier quality are not considered in these attempts (in [41], a comparative error analysis for different scene conditions - such as wood, meadow- is given). This spatial correlation creates time correlation in dynamic scenarios. In addition, the error characteristic is correlated with other factors such as speed and individual optimization steps of the selected algorithm.

In Table 2.1, the visual odometry translational and rotational errors are presented as translational and rotational offset ratio for different methods (reader should refer to KITTI Vision Benchmark Suite [1] to examine the details of these algorithms). The methods are tested on all test sequences of the benchmark. Translational and rotational errors are computed for all possible subsequences of length (5,10,50,100,150,...,400 meters). The evaluation table ranks methods according to the average of those values, where errors are measured in percent (for translation) and in degrees per meter (for rotation). A GPS aided inertial navigation system is used in this setup to achieve centimeter level accuracy for the ground truth.

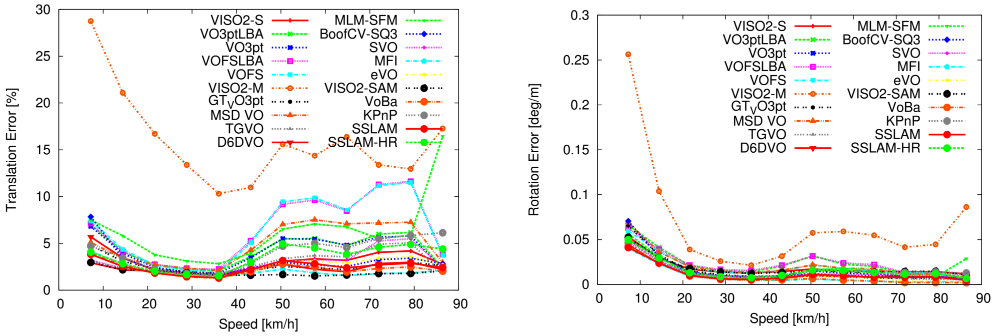
A more detailed comparison for different trajectory lengths and driving speeds can be observed in Figure 2.5. The experimental results support the above discussion on VO error characteristics. From Figure 2.5(c), it is evident that there is some correlation between the speed and VO error. A more detailed comparison for different trajectory lengths and driving speeds can be examined in Figures 2.5(a), 2.5(b), 2.5(c) and 2.5(d). It can be observed that VO error is quantitatively different for different algorithms. In addition, the effect of high speed on VO error becomes significant in some algorithms. These facts should be taken into account when a particular VO algorithm is to be fused with a different pose estimation method.

Table2.1: Visual Odometry OR Errors for Different Algorithms [1]

Rank	**Method**	**Translation**	**Rotation**
1	MFI	1.69 %	0.0066 [deg/m]
2	VoBa	1.77 %	0.0066 [deg/m]
3	VISO2-SAM	1.83 %	0.0152 [deg/m]
4	SSLAM	1.87 %	0.0083 [deg/m]
5	eVO	1.93 %	0.0076 [deg/m]
6	D6DVO	2.10 %	0.0083 [deg/m]
7	GT_VO3pt	2.21 %	0.0117 [deg/m]
8	VISO2-S	2.27 %	0.0152 [deg/m]
9	BoofCV-SQ3	2.27 %	0.0111 [deg/m]
10	TGVO	2.44 %	0.0105 [deg/m]
11	SVO	2.45 %	0.0109 [deg/m]
12	SSLAM-HR	2.45 %	0.0112 [deg/m]
13	KPnP	2.73 %	0.0107 [deg/m]
14	VO3pt	2.93 %	0.0116 [deg/m]
15	VO3ptLBA	3.17 %	0.0180 [deg/m]
16	MSD VO	3.50 %	0.0166 [deg/m]
17	MLM-SFM	4.07 %	0.0104 [deg/m]
18	VOFS	4.21 %	0.0158 [deg/m]
19	VOFSLBA	4.35 %	0.0189 [deg/m]
20	VISO2-M	13.79 %	0.0372 [deg/m]



(a) Translation error against path length (b) Rotation error against path length



(c) Translation error against speed (d) Rotation error against speed

Figure 2.5: Visual odometry error characteristics variation against velocity and path length, [1].

2.6 Conclusion

In this chapter, the fundamentals of visual-only 3D reconstruction and camera motion estimation algorithms are presented in the context of this dissertation. The image formation and projection process, visual feature extraction and matching, epipolar geometry, Structure-from-Motion, 3D Reconstruction, and Visual Odometry topics are visited for this purpose. The error characteristics of Visual Odometry is discussed in detail in order to understand the sources that contribute to this error; and the observations are confirmed from popular benchmark data and results of literature.

In the following chapters, a number of vision algorithms will be proposed based on the fundamental algorithms and observations.

CHAPTER 3

3D DENSE RECONSTRUCTION OF URBAN ENVIRONMENTS

The sparse 3D point cloud, that is described in Chapter 2, can be considered as samples from the surface of the scene observed by the camera. For a general scene, a straightforward method to obtain a dense reconstruction may be to recover the surface via interpolation [7]. Piecewise planar representations such as Delaunay triangulation and rate-distortion are practical ways of interpolating a sparse 3D point cloud to a dense 3D map.

The scope of this dissertation is street-level 3D reconstruction of urban areas from a camera. For this specific problem, there are strong heuristics about the observed scene that are potentially useful during obtaining a dense 3D map from a sparse 3D point cloud: Urban environments are generally composed of buildings that have planar facades, and these facades are placed in the direction of gravity. Having these observations, the piecewise planar representation is emphasized more and the planar patch sizes extend to building facades. These heuristics lead to a more efficient urban modeling scheme in terms of completeness, smoothness, and computational efficiency.

There are different ways to exploit such heuristics. Two novel methods are presented in this section together with a conventional one. These methods are:

- The conventional plane sweeping stereo method with slight modifications and addition of unique steps to plane hypotheses set construction.
- A new method that is based on plane hypothesis matching with superpixels.
- A new method that is based on planar patch fitting to sparse 3D point cloud using facade verticality.

The methods are discussed and examined in a two-view reconstruction framework, and generalized from there. The proposed planar modeling methods are efficient and applicable to large scale 3D urban reconstruction problem. Adaptations of the proposed algorithms are presented for large scale scenes together with some experimental results.

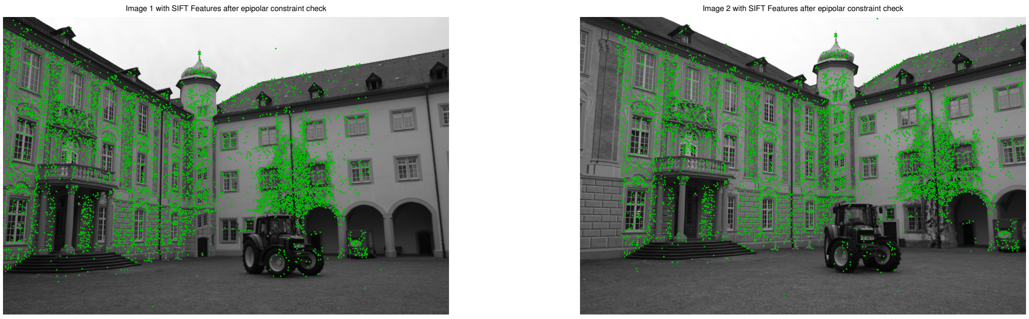


Image 1 with SIFT Features after epipolar constraint check
Image 2 with SIFT Features after epipolar constraint check

(a) Image 1

(b) Image 2

Figure 3.1: Two images from an urban environment with SIFT descriptors (after epipolar constraint check)

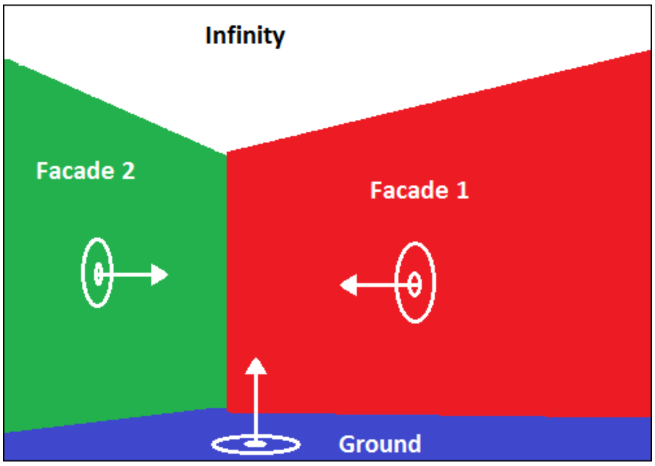


Figure 3.2: Pictorial depiction of a classical urban scene, ground and building facade planes and their normals.

3.1 Plane-Sweeping Stereo with Multiple Sweeping Directions

Urban environments exhibit mostly planar surfaces which are often imaged at oblique angles. A typical image, for example, might contain a ground plane and multiple facade planes intersecting at right angles (see Figure 3.1 and Figure 3.2).

In this section, a multi-view plane-sweep-base stereo algorithm is proposed which can handle multiple slanted surfaces as defined in [10] and [17]. The algorithm consists of (1) identifying the scene's principle plane orientations, (2) constructing a plane hypotheses set from these orientations, (3) estimating depth by performing a plane sweep for each plane hypothesis. Each plane sweep is intended to reconstruct planar surfaces having a particular norm. The plane hypotheses set construction is different from the references. In [10] and [17], the plane hypotheses set is generated to reconstruct the whole depth that leads to a dense reconstruction which contains both the building facades and out of scope objects such as trees and cars. This is an expensive algorithm in terms of computational cost when the goal is to get only building facades, not the whole scene depth. Therefore, this phase is modified such that the plane hypotheses set is generated around the plane priors. Moreover, angle sampling is introduced in addition to depth sampling for plane hypotheses set generation to improve the pixel based plane association.

3.1.1 Identifying Plane Priors

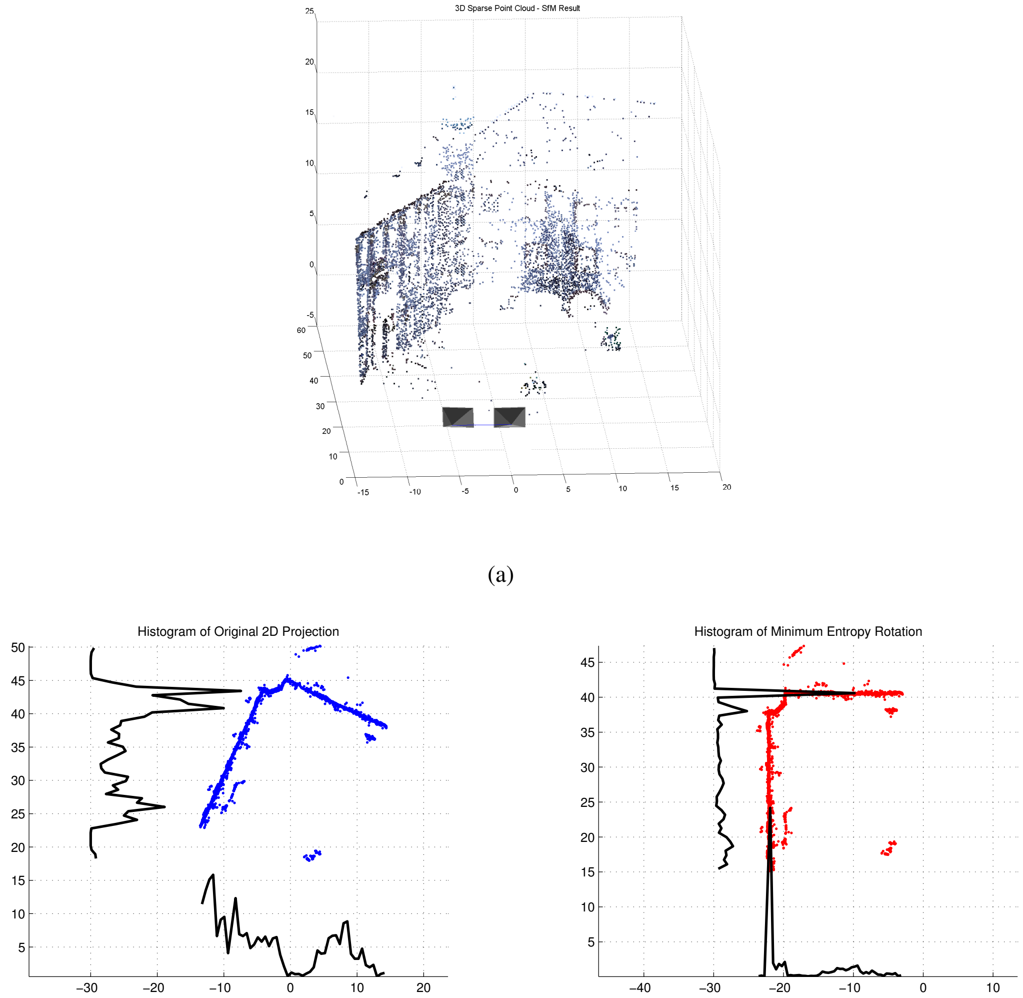
The surface normals of the facade planes can be identified by the analysis of 3D points obtained through sparse structure from motion. The normal vector for the ground can be identified from the direction of the gravity and camera motion to handle slanted ground planes. First the direction of gravity vector is determined either from the IMU or from vanishing points. By assuming the ground plane has zero slope in the direction perpendicular to the computed camera motion, a good estimate can be extracted for the ground plane normal.

$$\pi_{\mathbf{gnd}} = \frac{(\boldsymbol{g} \times \boldsymbol{v}) \times \boldsymbol{v}}{\|(\boldsymbol{g} \times \boldsymbol{v}) \times \boldsymbol{v}\|} \tag{3.1}$$

where $\boldsymbol{g}$ is the gravity vector direction and $\boldsymbol{v}$ is the camera motion direction. Under perpendicular two facade assumption as described in [10], the facade normals are assumed to be perpendicular to the gravity vector and are, therefore, determined by a rotation about the gravity vector. The rotation of the facades can be determined as follows:

```
1. Compute the orthogonal projection of each 3D point in the direction of gravity. Note that 3D points on a common vertical facade will project to a line.
2. Evenly sample the space of in plane rotations from 0 to 90 degrees.
3. For each rotation $R$, rotate the set of 2D points and construct two histograms $H_u$ and $H_v$. Each bin in $H_u$ counts the number of points with a similar $u$ (same for $H_v$).
4. Compute the entropy of each histogram.
5. Select the rotation which has the lowest sum of entropies.
```

As shown in Figure 3.3, entropy is minimized when points are aligned in direction u and v. Note that this algorithm can identify only two perpendicular facades. For a more general scene that have more than two building facades and facades crossing each other at different angles (see Figure 3.4), the entropy based initial sweeping direction identification algorithm is not appropriate. The lines (or line segments) should be extracted carefully from the projected 2D space for proper planar modeling. This problem can be described as fitting instances of a model to data corrupted by noise and outliers. There are numerous algorithms in the literature to tackle this problem, ranging from basic least square estimate to RANSAC. When the num-

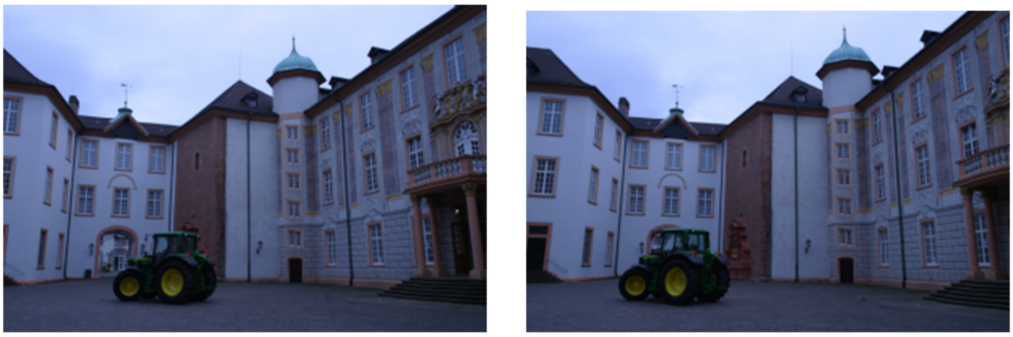


(a)

(b)

(c)

Figure 3.3: (a) 3D point cloud from SfM, (b) 2D projection of the sparse 3D points in the direction of gravity (c) Minimum entropy rotation of the 2D points based on x-y histogram



(a) Image 1

(b) Image 2

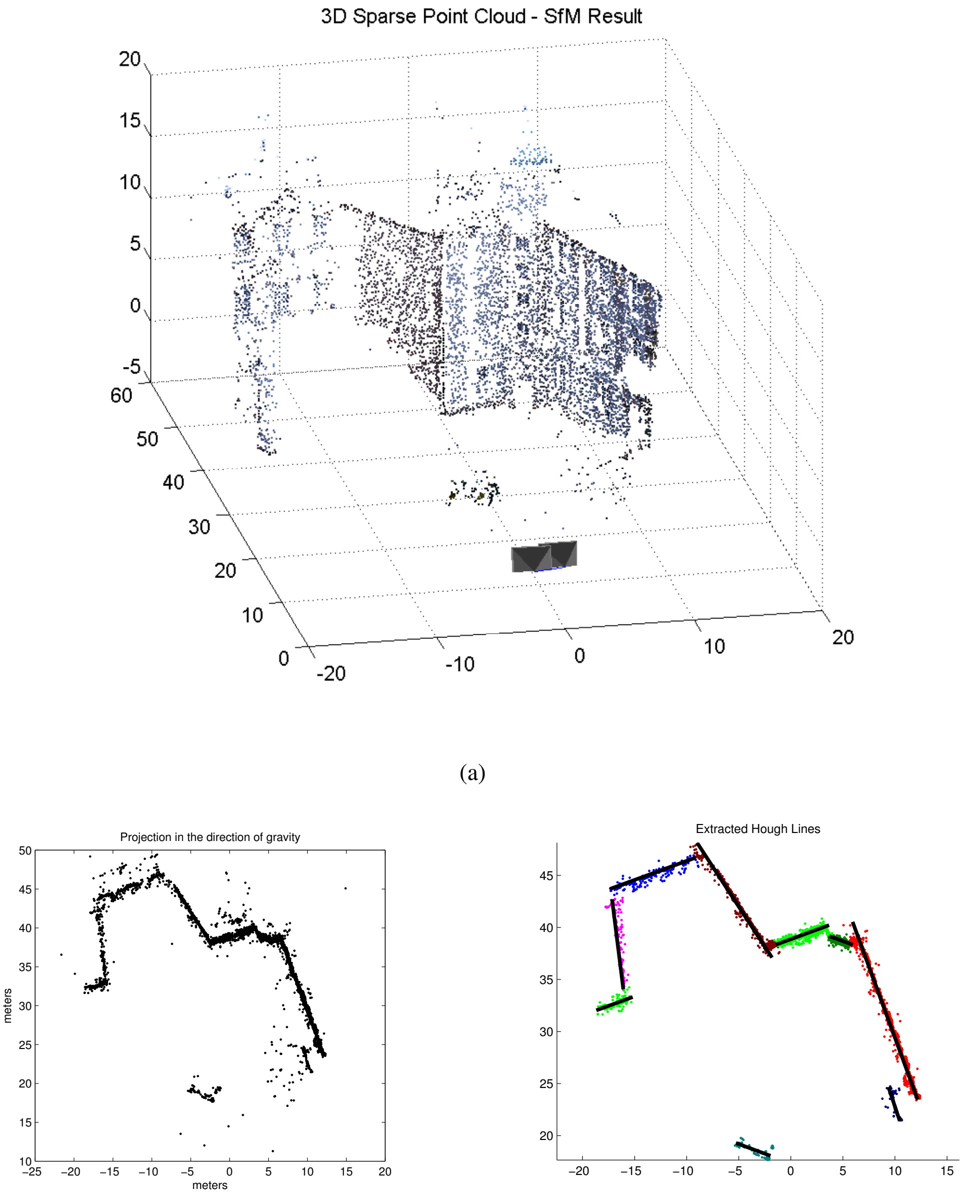
Figure 3.4: Two images from the EPFL dataset [4] that have multiple facades (castleP30-0025 and 0026)

ber of lines are unknown and greater than one, Hough transform [49] and RANSAC based J-linkage [50] are preferred most of the time. The J-linkage approach is time consuming and there are lots of parameters to tune compared to Hough transform.

An iterative Hough transform is proposed for the line fitting problem in the projected 2D point set. Hough transform converts the coordinate space of the points into line parameter (distance from the origin and slope) set and votes for the best match in this set by maximization of accumulation. An iterative Hough transform is required, if there exist more than one line [51]. The Hough Transform itself does not take into account the noisy data explicitly; therefore, a pre-conditioning stage is proposed to filter out the outliers. The iterative Hough transform is applied as follows:

```
1.  Eliminate outliers by checking point distinctiveness.
2.  Apply Hough Transform on remaining points.
3.  Select the maximally voted line, associate close points to this line, remove these points from the set.
4.  Repeat 2-3 until the remaining number of points are below a threshold.
5.  Merge close lines.
6.  Split lines if necessary based on point clustering.
7.  Eliminate line segments represented with small number of points.
```

In Figure 3.5, the output of this algorithm is depicted for the multi-facade urban environment of Figure 3.4. 3D plane hypotheses are obtained by extending these lines in the direction of gravity.



(a)

(b)

(c)

Figure 3.5: (a) 3D point cloud from SfM, (b) 2D projection of the sparse 3D points in the direction of gravity, (c) Identified lines with iterative Hough Transform

3.1.2 Plane Hypotheses Generation by Sampling of Plane Priors

Once the plane priors are computed, a family of planes are generated for each prior. The family of planes are defined by the following formula:

$$\Pi_m = [\mathbf{n}_m^T \;\; -d_m] \tag{3.2}$$

where $\mathbf{n}_m$ is the unit length normal of the plane and d_m is the distance of the plane to the origin, namely center of the reference camera. Therefore, there are two parameters to play during this phase, i.e. depth and angle.

The depth interval is determined by the analysis of sparse 3D points that indicate the location of the facades. They are used as a prior when selecting depth range for the family of planes. The spacing of the planes in the range can be uniform.

There are no heuristics about the angle sweep. A reasonable sweep number and angular increment shall be selected not to overload the processing.

3.1.3 Plane Sweeping Stereo

Plane Sweeping Stereo tests a family of plane hypotheses and records the best plane for each pixel as scored by some dissimilarity measure in a reference view. The inputs are M 3D-planes for the depth tests and $N+1$ images at different camera positions and their camera projection matrices P_k. The camera projection matrices are assumed to be known. The estimation of camera projections are detailed in Chapter 2.4.

In order to test the plane hypotheses Π_m for a given pixel at (x, y) in the reference view I_{ref}, the pixel is projected into the other images $k = 1...N$. The mapping from the image plane of the reference camera P_{ref} to the image plane of the camera P_k is a planar mapping, and can, therefore, be described by a homography $\mathrm{H}(\Pi_m, \mathrm{P}_k)$ induced by the plane Π_m as given in the following equation:

$$\mathrm{H} = \mathrm{K}_1 \left(\mathrm{R}_1 - \frac{\mathbf{t}_1 \, \pi_{\mathbf{n}}^T}{\pi_d} \right) \mathrm{K}_0^{-1} \tag{3.3}$$

where K_0 and K_1 are the calibration matrices, R_1 is the orientation of second camera with respect to the first camera, $\mathbf{t}_1$ is the translation of second camera center with respect to the first camera, $\pi_{\mathbf{n}}$ is the plane normal and π_d is the distance of the plane to the first camera center.

If the plane intersects the surface projected to pixel at (x, y) in the reference view, the colors $I_k\ (x_k, y_k)$ and $\mathrm{I}_{ref}\ (x, y)$ should be similar assuming Lambertian surfaces. The location (x_k, y_k) in image I_k of the mapped pixel (x, y) of the reference view is computed using homogeneous coordinates as given in the following equation:

$$[\tilde{x} \;\; \tilde{y} \;\; \tilde{w}]^T = \mathrm{H}_{\pi_m, P_k}^T \; [x \;\; y \;\; 1]^T \;, \qquad x_k = \tilde{x}/\tilde{w} \qquad y_k = \tilde{y}/\tilde{w} \tag{3.4}$$

The absolute difference of intensities can be taken as the dissimilarity measure. To reduce the sensitivity to noise, several measurements in the neighborhood of the pixel can be used. Once the cost function for all pixels has been computed, the depth map is extracted. The first step is to select the best plane at each pixel in the reference view. This may simply be the plane of minimum cost. The cost function is given in the following equation:

$$C(x, y, \pi_k) = \sum_{k=0}^{N-1} \sum_{(i,j)\in W} |\mathrm{I}_{ref}(x-i, y-j) - \mathrm{I}_k^*(x-i, y-j)| \tag{3.5}$$

where W is the matching window and I_k^* is the image I_k warped by the homography $\mathrm{H}_{\pi_m}^T$. The minimum cost labeling is done in accordance with the following formula:

$$\tilde{\pi}\,(x, y) = \operatorname*{argmin}_{\pi_m} C(x, y, \pi_k) \tag{3.6}$$

3.1.4 Depth Estimation

After identifying the plane association of each pixel, for a given plane Π_m at pixel (x, y) the depth can be computed by finding the intersection of Π_m and the ray through the pixel's center:

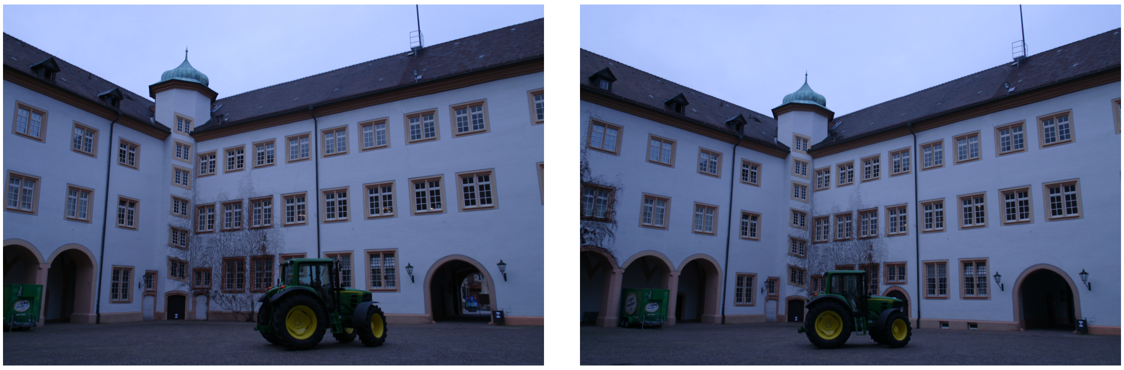
$$\mathbf{z}_m(x, y) = \frac{-d_m}{[\begin{array}{ccc} x & y & 1 \end{array}] \cdot \mathrm{K}_0^{-T} \cdot \mathbf{n}_m} \tag{3.7}$$

3.1.5 Experimental Results

The algorithm is demonstrated on images from EPFL Computer Vision dataset [4] (see Figure 3.6). The projection matrices were available for each scene in the dataset. In Figure 3.7, the result of minimum cost labeling without neighbourhood filtering is depicted. The result is quite noisy, since no neighborhood filtering is present. It is important to note that pixel by pixel minimization of the cost function in equation 3.5 is not sufficient to obtain a smooth result.

In Figures 3.8 and Figure 3.9, the result of minimum cost labeling depth and angle sampling are depicted consecutively. A 15 × 15 window smoothes out the noisy parts in both results. It can be inferred that the angle sampling improves the result. The result of depth estimation is given in Figure 3.10. A neighborhood threshold filter can be applied to the depth map to eliminate the discontinuities.

The reconstruction of ground plane and infinity (sky) with this approach is ill-posed and requires separate handling. The reason is due to the homogenous structure of these areas. The cost minimization for these planes is erroneously affected by other planes, since the homography transformation has the possibility to produce a false plane association, especially for large homogeneous portions of the scene. The proposed superpixel based algorithm in Section 3.2 solves this problem inherently by sticking only to the points that form the planes.



(a) Image 1

(b) Image 2

Figure 3.6: Two images from an urban environment from the EPFL dataset [4] (castleP30-0025 and 0026)

For the scene in Figure 3.4, the reconstruction ambiguities are more observable since the number of building facades is high, there are small facades together with larger ones, and they intersect in non-orthogonal angles. In Figures 3.11, 3.12 and 3.13, the result of minimum cost labeling with depth+angle sampling are depicted. The smoothing filter sizes are 15×15, 50×50 and 100×100, respectively. The smoothing gets better, when the window size is increased; however, this dramatically increases the computational cost and brings the risk of loosing detail.

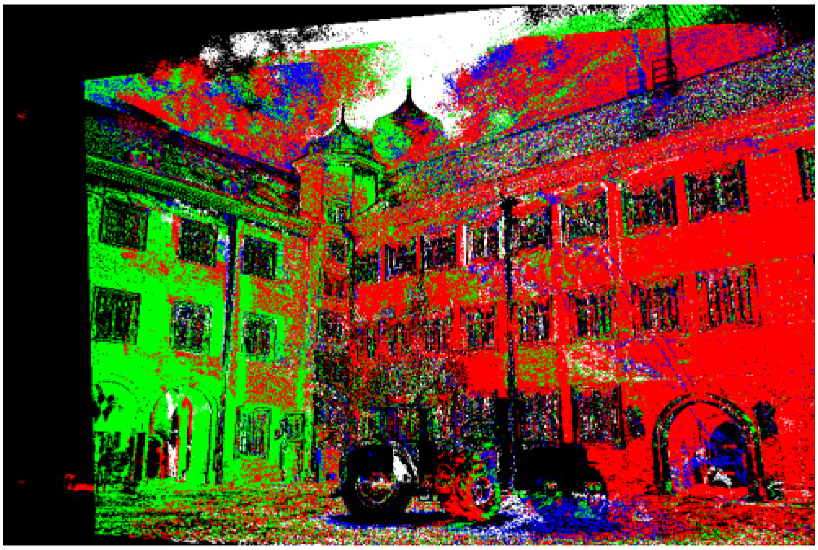


Figure 3.7: Plane sweeping stereo with planes of minimum entropy (no angle or depth sampling) and without window filtering for the images in Figure 3.6.

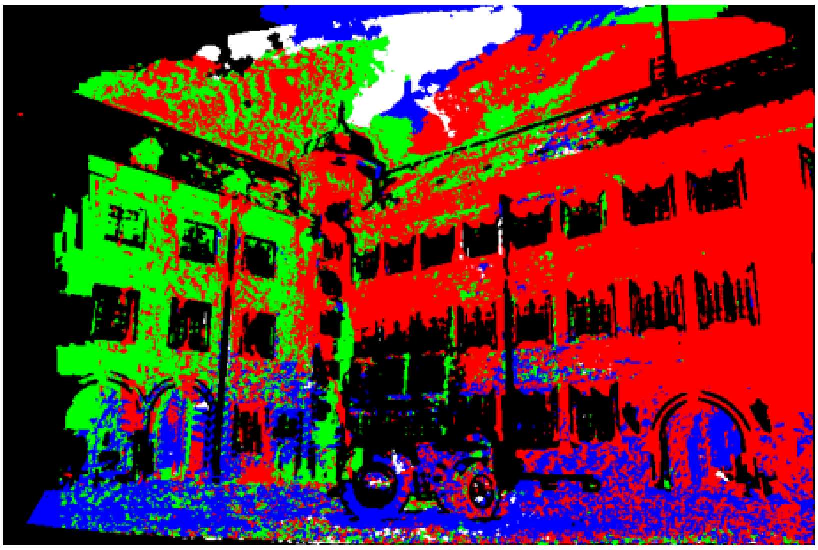


Figure 3.8: Plane sweeping with depth sampling and window filtering for the images in Figure 3.6; window size = 15, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments

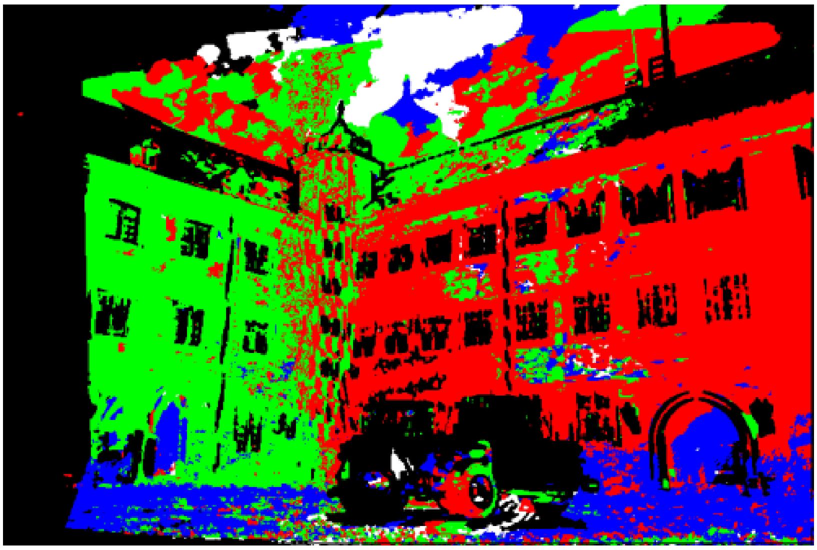


Figure 3.9: Plane sweeping with angle sampling and window filtering for the images in Figure 3.6; window size = 15, number of angle samples around minimum entropy plane = -5 to +5 degrees with 1 degree increments

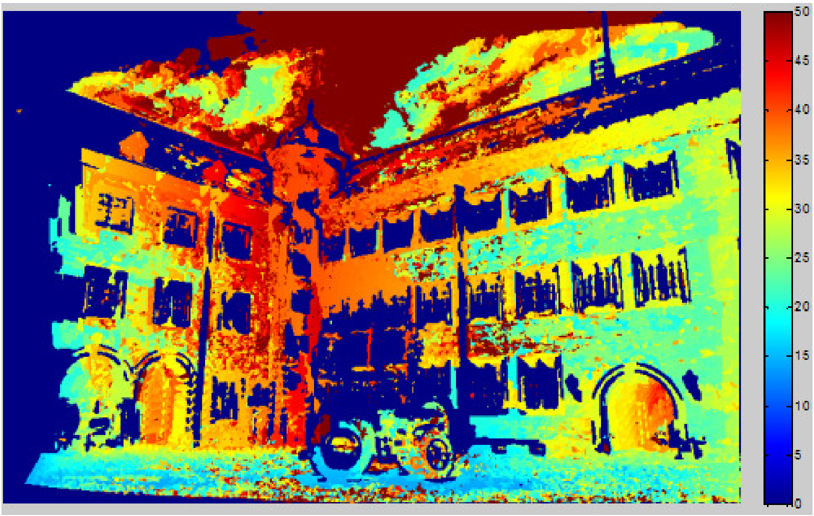


Figure 3.10: Colored depth representation as result of plane sweeping with depth sampling for the images in Figure 3.6.

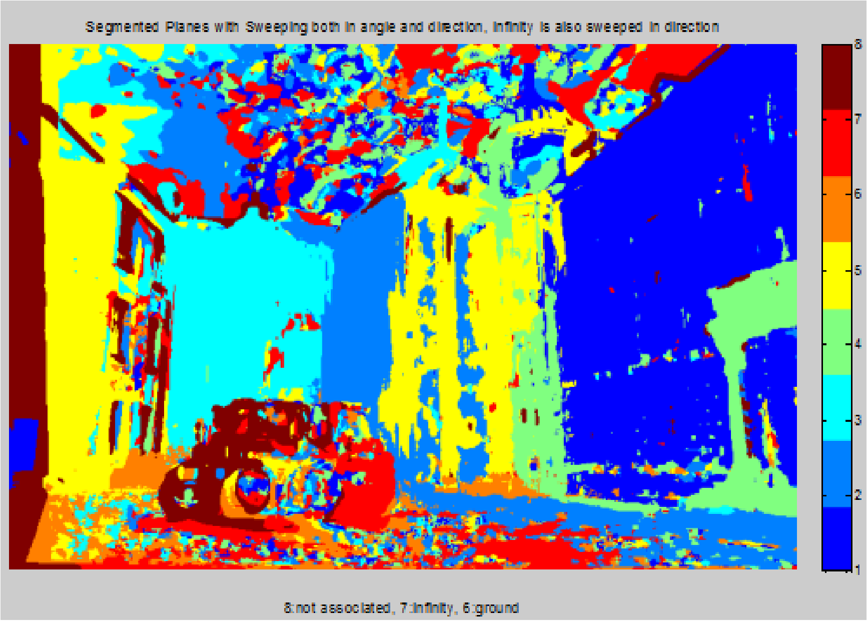


Figure 3.11: Plane sweeping with depth and angle sampling and window filtering for the images in Figure 3.4; window size = 15, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments, number of angle samples around prior plane direction = -5 to +5 degrees with 1 degree increments

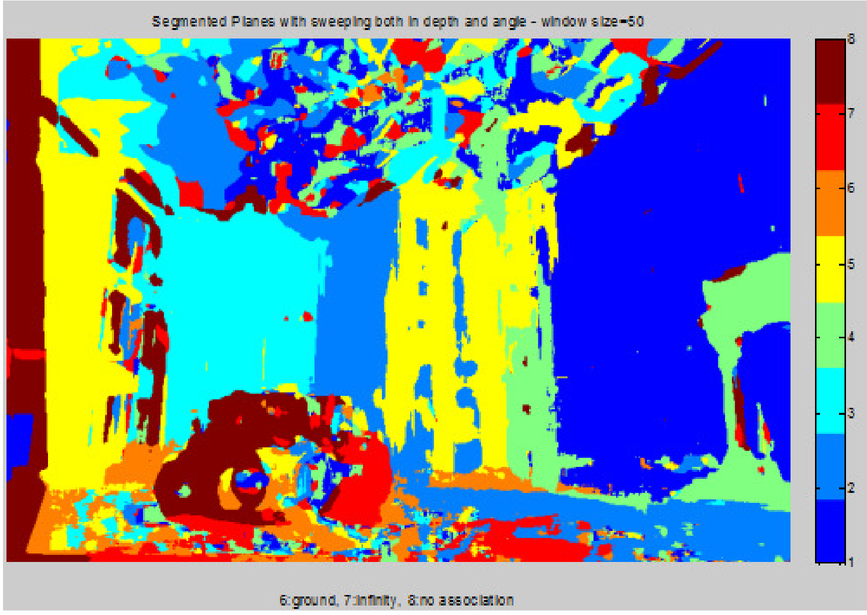


Figure 3.12: Plane sweeping with depth and angle sampling and window filtering for the images in Figure 3.4; window size = 50, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments, number of angle samples around prior plane direction = -5 to +5 degrees with 1 degree increments

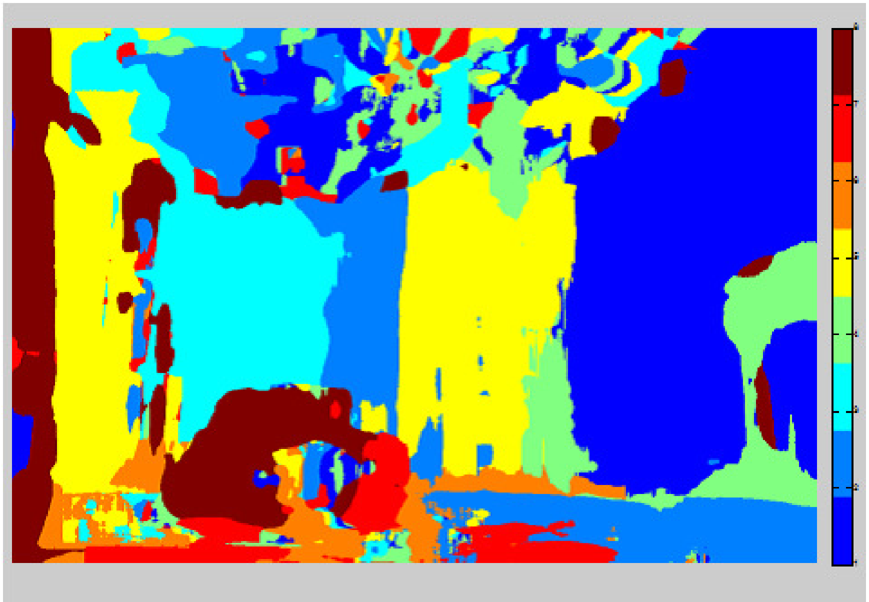


Figure 3.13: Plane sweeping with depth and angle sampling and window filtering for the images in Figure 3.4; window size = 100, number of depth samples around minimum entropy plane = -5m to 5m with 1m increments, number of angle samples around prior plane direction = -5 to +5 degrees with 1 degree increments

3.2 3D Modeling with Plane Association to Superpixels

The algorithm in Section 3.1 exploits the special structure of urban environments in order to find plane priors and uses these plane priors to generate a plane hypotheses set that is used in homography transformation. In one aspect, this approach can be assumed as a loose dependence on the urban structure. Strengthening the dependence on plane priors, that are obtained from the sparse 3D point cloud, could be an option, especially when the scope is reconstructing the building facades, not the whole scene itself. This will be a simpler reconstruction, since it will not contain details that are far from the building facades such as trees, cars etc. However, such a skeletonized reconstruction is beneficial for 3D modeling of the basic structure and especially when the scope is 3D identification of the environment instead of 3D modeling. The proposed algorithm is summarized as follows:

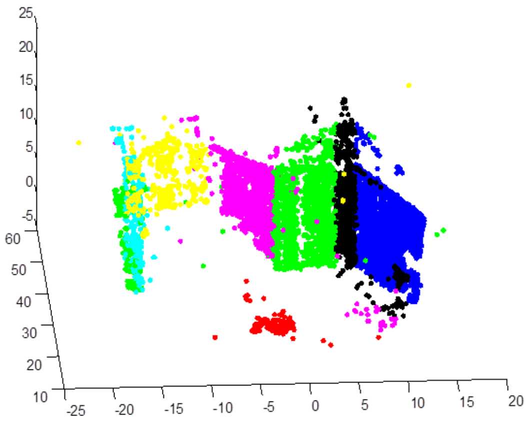


Figure 3.14: Labeled sparse 3d points with plane priors

```
Given two frames whose projection matrices are known
1.  Find SIFT descriptive points.
2.  Find sparse 3D points by SfM.
3.  Project these 3D points in the direction of gravity to local level frame and obtain a 2D point set.
4.  On this 2D point set, perform an iterative Hough transform to obtain lines that correspond to the building facades.
5.  Construct 3D planes that correspond to building facades from the lines and associate the sparse 3D points to these planes.
6.  Divide the first frame into superpixels using SLIC (Simple Linear Iterative Cluster) algorithm.
7.  Associate the superpixels to planes using their 3D point associations.
8.  Associate the unattended superpixels by checking color similarity with neighboring superpixels.
```

For further improving the previous approach, a superpixel based scheme is proposed. Using superpixels exploit another structure of the urban environments, piecewise planarity. Superpixels are utilized to construct building facades as smooth and complete planes.

The proposed algorithm depends on the heuristics that are extracted from the structure of urban environments as in Section 3.1.

Steps 1-4 was detailed in the previous sections. Associating the sparse 3D points to the plane priors is the starting point for the proposed algorithm. This process is shown in Figure 3.14. The 3D point cloud in this figure belongs to the scene of Figure 3.4 and the lines corresponding to the plane priors are given in Figure 3.5. Every plane is represented by a different color.

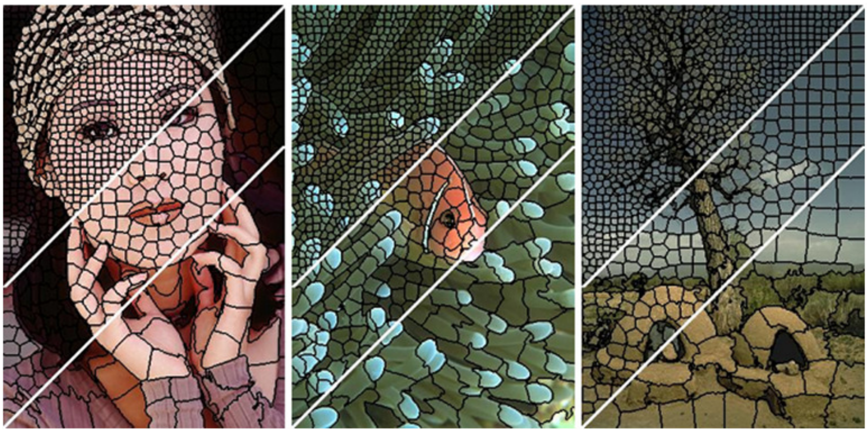


Figure 3.15: Simultaneous Linear Iterative Clustering demonstration [5]

3.2.1 SLIC Superpixel

In order to find a dense depth map, every pixel should be assigned to a plane. This approach yields a huge search problem, especially when the image size and number of planes are large. Superpixels are natural selections for a more efficient implementation, both in processing power and output smoothness. There is a possibility to lose detail in this approach, but this is a minor concern in urban modeling. SLIC (Simultaneous Linear Iterative Clustering) approach is used for superpixel construction [5]. The algorithm adapts a K-means clustering approach to efficiently generate superpixels. It is a gradient-ascent based algorithm. Starting from a rough initial clustering of pixels, the algorithm iteratively refines the clusters until some convergence criterion is met. The clustering is performed on a 5D space (3D color-CIELab, 2D pixel position). Compactness and locality is achieved by introducing 2D pixel position to clustering.

SLIC is selected for its adherence to boundaries, local homogeneous characteristics and cluster number controllability. The result of clustering with SLIC with different number of clusters is depicted in Figure 3.15.

3.2.2 Plane Association

It is required to assign every pixel in the image to a plane in order to achieve a dense reconstruction. This plane association process should be more efficient, if it is conducted on superpixels, instead of individual pixels. There is a possibility to loose details during reconstruction; however, this is a minor issue in urban reconstruction, since the superpixels generally fit onto building facades and follow boundaries. The remaining parts of the image, such as sky, trees, cars and other small objects, will not be assigned to planes in this case, which is feasible if a planar silhouette of the urban structure is required.

A superpixel is assigned to a plane based on the score of the sparse 3D point to plane associations within the superpixel. The sparse 3D points are assigned to planes based on their Euclidian distance to the individual planes. Every 3D point within a superpixel votes for its associated plane and the maximum number of associated plane is selected for that superpixel. In this approach, if a superpixel does not have any projected 3D point within its boundary, it will not be associated to plane. A neighborhood check is proposed to assign these superpixels to the most probable plane. Neighboring superpixels that are assigned to a plane are searched and the one with closest color is selected for merging. If none of the neighbors are associated with a plane, then this superpixel is left unassociated.

3.2.3 Experiments

The first frame is divided into 2000 superpixels in Figure 3.16. The sparse 3D point cloud of this frame couple is given in 3.14. The plane association with neighborhood filling processes is depicted in Figure 3.17. The textured depth map of this superpixel based reconstruction is depicted in Figure 3.18.

In Figure 3.19 and Figure 3.20, the results of plane association and plane sweeping are depicted for comparative analysis. The result of plane association is more clean and exact compared to the plane sweeping stereo. The ground plane and sky modeling ambiguity is inherently solved in the plane association approach. In addition, plane sweeping takes longer time, since it operates on pixels and uses a smoothing post-filter to suppress the noise in pixel based reconstruction.

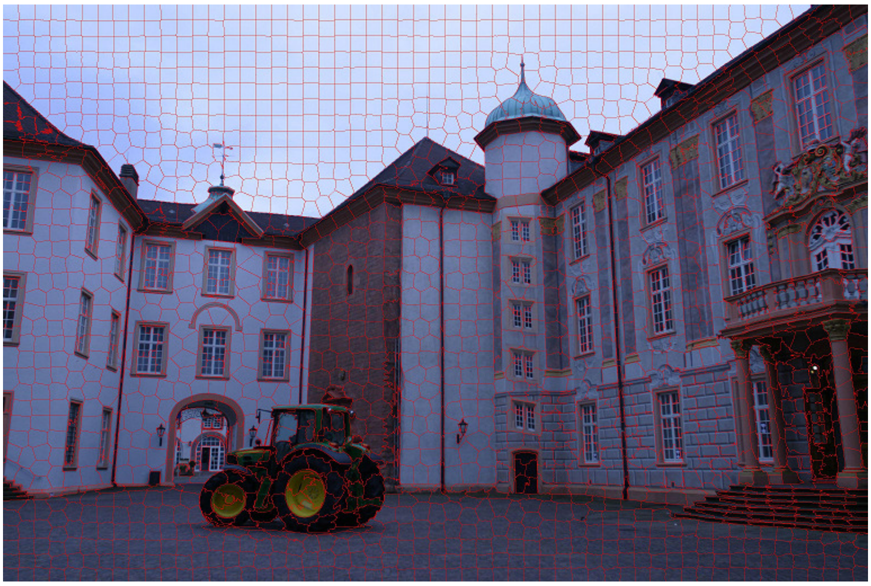


Figure 3.16: Simultaneous Linear Iterative Clustering for the urban scene

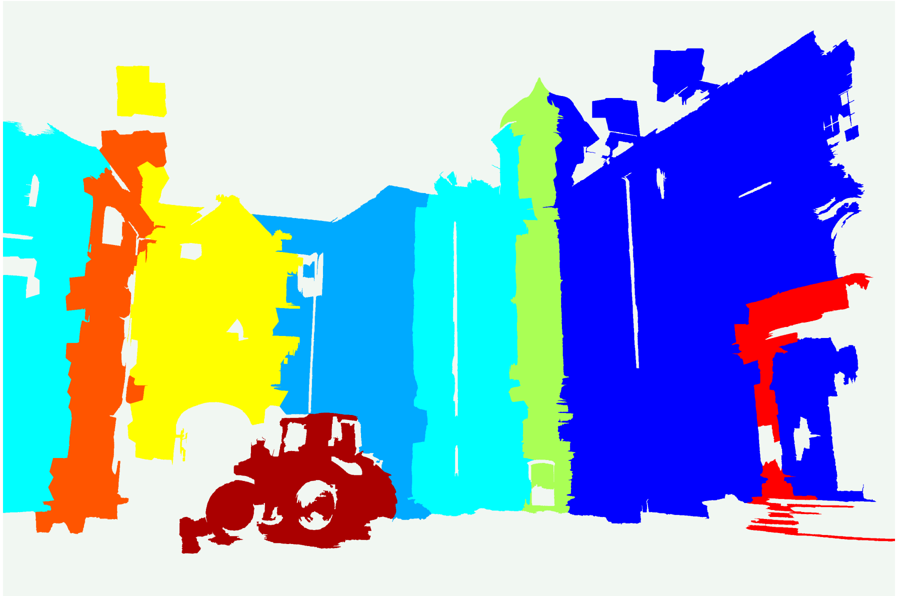
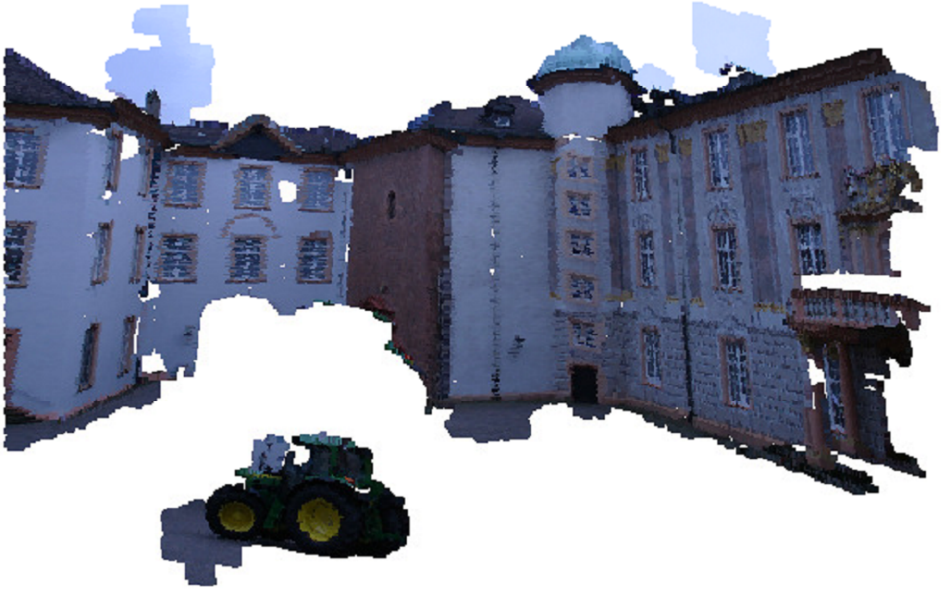
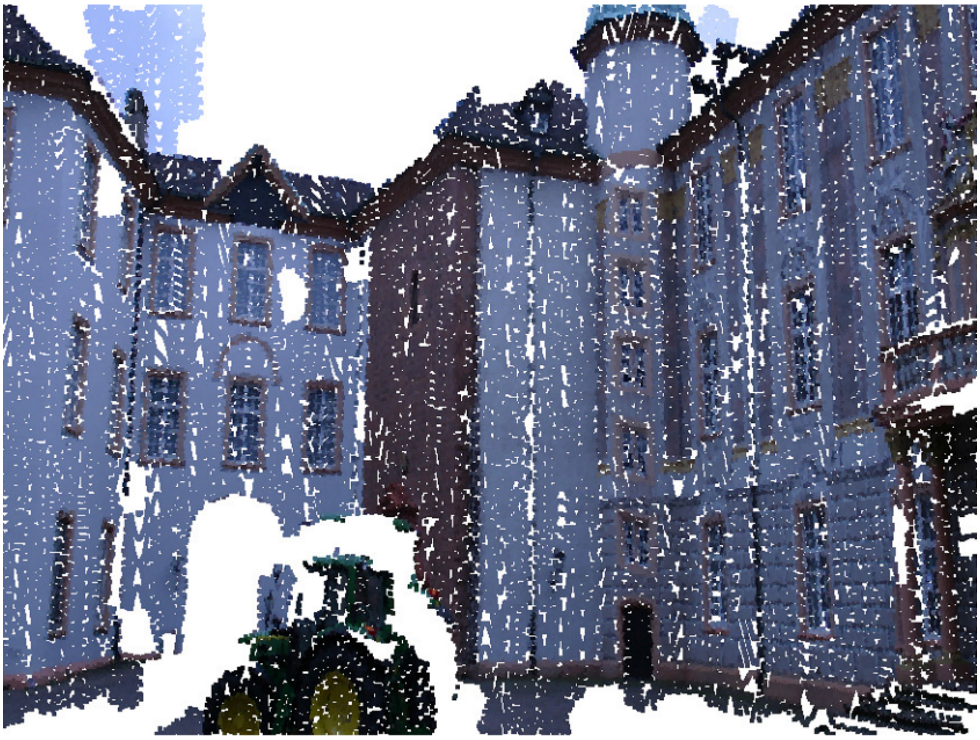


Figure 3.17: Superpixels associated to plane priors with neighborhood filling

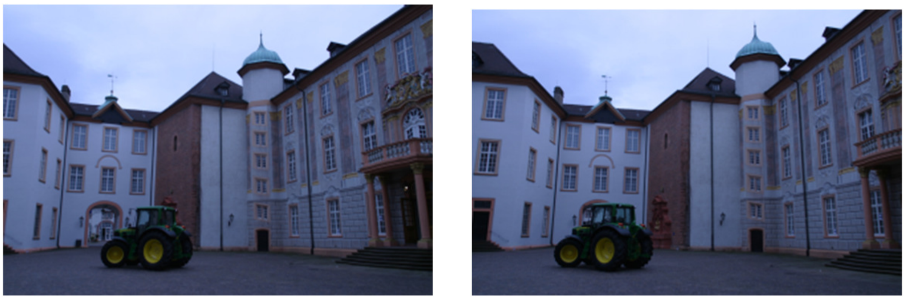


(a) First view from the textured depth map



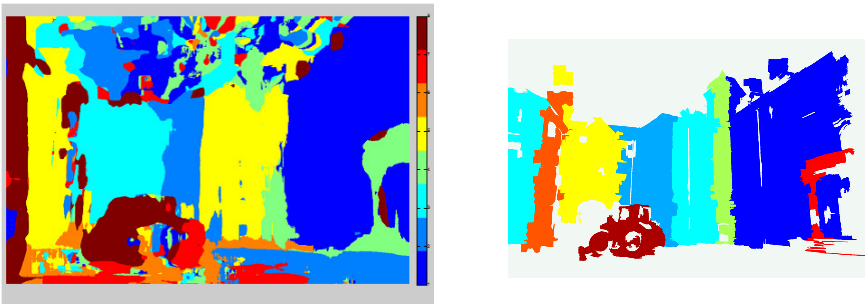
(b) Second view from textured depth map

Figure 3.18: Textured dense depth map obtained from SLIC plane patches



(a) Image 1

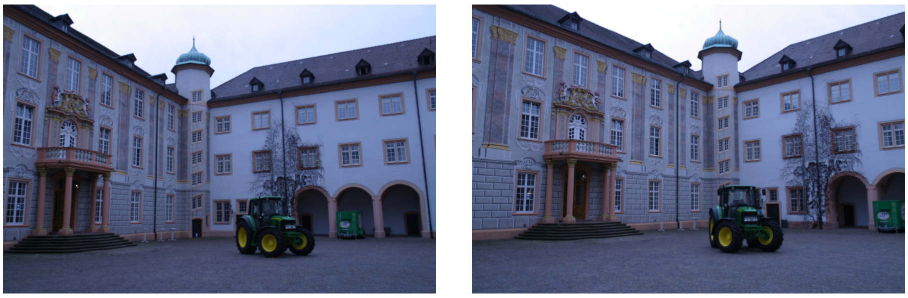
(b) Image 2



(c) Plane-Sweeping Stereo Result

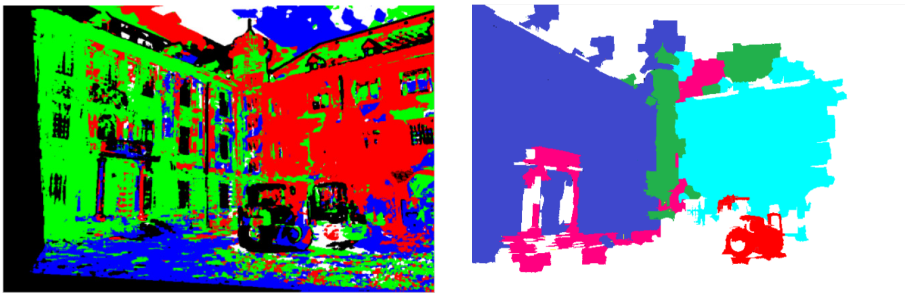
(d) Plane Association with Superpixels Result

Figure 3.19: 3D Reconstruction Results of algorithms in sections 3.1 and 3.2



(a) Image 1

(b) Image 2



(c) Plane-Sweeping Stereo Result

(d) Plane Association with Superpixels Result

Figure 3.20: 3D Reconstruction Results of algorithms in sections 3.1 and 3.2

3.3 Large Scale Urban Modeling with Planes

The 3D reconstruction problem becomes cumbersome for long, large scale and outdoor sequences. The difficulties can be listed as follows:

- Camera pose drift causes map inconsistency as the sequence length increases.
- The image sequences are generally captured from a moving video camera. Motion introduces blur, and the number of spurious feature matches increases under motion.
- The image resolution is kept at a fairly minimum value due to the enormous amount of data storage. This yields a smaller number of features per image.
- Lighting changes and shadows are common in urban scenes and they might cause spurious feature matches.
- Independently moving objects (cars, pedestrians) might cause spurious feature matches.
- The existence of irrelevant static objects (parked vehicles, road signs, trees, bushes etc.) could jeopardize the quality of building models.
- The amount of input (image sequence) and output data (3D map) is enormous.

A typical urban scene is given in Figure 3.21 where pedestrians, parked vehicles, trees and bushes, shadows, and direct sunlight effects are clearly visible. These difficulties might impede the use of a good 2-view reconstruction scheme directly to long urban sequences. In this section, an efficient and reliable method is proposed to tackle all of the above mentioned difficulties.

The camera pose estimation problem is discussed in Chapter 4 and solutions are proposed

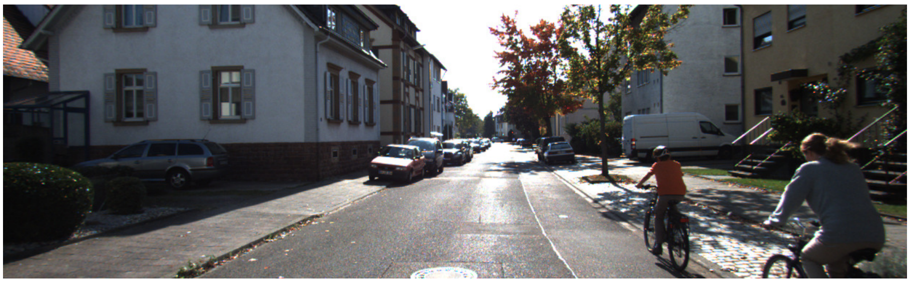


Figure 3.21: A typical urban scene from KITTI benchmark [1]

in a Visual-Inertial integration framework. Therefore, in this section the camera pose is assumed to be known in world coordinates. The reliable camera pose estimates are used as pre-conditioning step for identifying the quality of the feature matches. The feature matches are obtained from SIFT. An epipolar check is applied to feature matches by using the given camera poses only. In addition, a 3D-2D re-projection filter is applied after triangulation based on given camera poses. These filters are very useful in outdoor and uncontrolled environments for removing outliers due to motion blur, independently moving objects, lightning affects or failure of the feature matching algorithm. Another advantage of using a geo-located and metric pose estimation system is to have a metric and geo-located 3D point cloud that can directly be related to real world situations. For example, in an urban area, most of the streets are covered with parked cars, and the 3D point cloud will definitely have points from these vehicles. In an ideal modeling framework, these vehicles should not be present in the model, since they are temporary in the area. These objects can be removed from the model by image segmentation methods that enables to identify building facades, roads, trees, vegetation, cars, pedestrians, sky etc. This kind of a semantic segmentation requires sophisticated algorithms which include a learning stage and a probabilistic labeling stage. Since the metric pose estimation yields a metric 3D point cloud in the proposed scheme, this can be used to remove the objects that are at a certain elevation from the local ground easily. The power of multi-sensor data is used effectively in this case.

The idea of using the planarity and verticality assumption in urban areas, which is described in the previous section, is applied to long sequences as well. The two view-reconstruction scheme that uses these assumptions yields the following:

- A sparse 3D point cloud.
- Line segments extracted from vertically projected 3D point cloud.
- A plane set constructed from vertical lines extensions.
- Plane associations to SLIC superpixels.
- Dense depth map in terms of planar patches.

These outputs can be estimated for each consecutive image pair in the long urban sequence, and fused in a way to obtain the model of the whole urban area. However, extra processing is required for the long urban reconstruction problem. Low image resolution, blurring, lighting change and other disadvantages of large scale reconstruction result in poor and small number of feature matches even with a robust feature tracker, such as SIFT. The sparse 3D point cloud (obtained from SfM) is not able to represent the building surfaces, when the feature number is low. We propose to accumulate the point clouds from previous consecutive reconstructions and apply the planar modeling algorithm afterwards. The accumulation interval can vary depending on the total feature number in the bucket. The 2D projection and iterative Hough transform for line search algorithms are executed on these accumulated 3D point clouds.

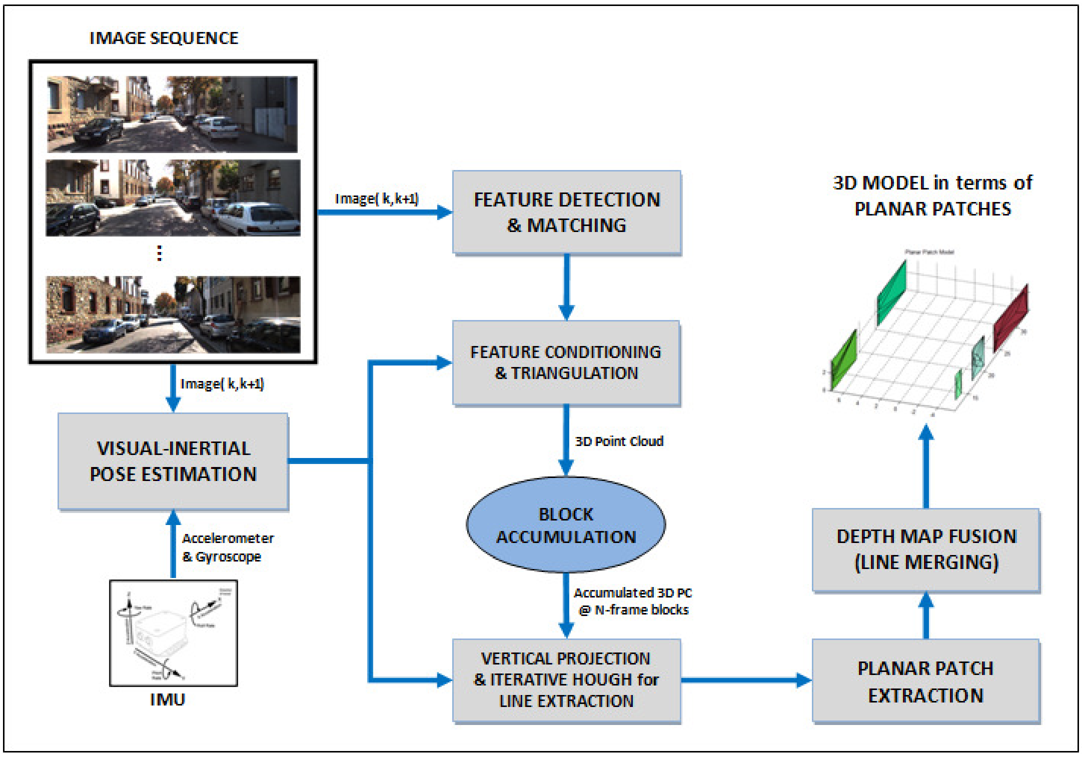


IMAGE SEQUENCE
Image(k,k+1)
FEATURE DETECTION & MATCHING
3D MODEL in terms of PLANAR PATCHES
FEATURE CONDITIONING & TRIANGULATION
Image(k,k+1)
VISUAL-INERTIAL POSE ESTIMATION
3D Point Cloud
BLOCK ACCUMULATION
Accelerometer & Gyroscope
DEPTH MAP FUSION (LINE MERGING)
Accumulated 3D PC @ N-frame blocks
IMU
VERTICAL PROJECTION & ITERATIVE HOUGH for LINE EXTRACTION
PLANAR PATCH EXTRACTION

Figure 3.22: Urban 3D Reconstruction Scheme

In the previous section, it has been shown that SLIC superpixel plane association technique is able to obtain dense depth map of the observed scene and outperforms the pixel-based depth estimation both in computational cost and exactness of the result. However, this new method still has a computational burden for long sequences; it requires image clustering for every image in the sequence, in its primitive form. This requirement can be relaxed by applying clustering only to key-frames by considering visibility and overlap constraints. A simpler way to use the planar verticality assumption is to exploit the 3D point cloud and extracted lines directly to get planar patches without going back to the 2D image domain. In this approach, the scattered 3D points that form a plane segment are projected onto that virtual plane and the convex hull of the planar points are estimated to get the boundaries of the planar patch. The proposed algorithm is depicted in Figure 3.22.

3.3.1 Experimental Results

The KITTI benchmark suite [1] is used for the experimental evaluation of the proposed algorithm. There are several sequences in this data set (road, city, residential etc.), with color and monochrome stereo image sequences, inertial measurement sensor data, and ground truth DGPS aided Inertial Navigation data. We used only the monochrome image sequence of the left camera (monocular) and the inertial measurement sensor data in the experiments. We

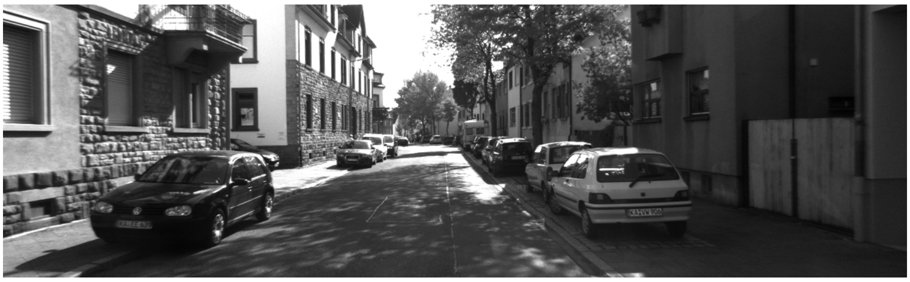


Figure 3.23: The starting scene of the city sequence [1]

have selected a city and a residential sequence for the experiments. The camera internal calibration parameters, transformation between the camera and inertial measurement unit frames, and the time synchronization between the sensors are present in the dataset. The monocular visual odometry aided inertial navigation system, that is described in Chapter 4, is for camera pose estimation.

The two-view reconstruction results are accumulated for periods of non-overlapping 2 seconds (20 reconstructions). Theroad/building discrimination, 2D projection and line detection algorithms are executed on the accumulated 3D point cloud. The extracted lines are checked against previous lines and merged, if necessary. The first image of the city sequence is given in Figure 3.23. This sequence has 268 images collected at 10 Hz. The mean feature correspondence per image is 250. The cumulative 3D point cloud is presented in Figure 3.24. The feature detection algorithm is based on a combined blob-corner detector and feature descriptor as described in [39]. The executable code is downloaded from the KITTI benchmark. The road/building discrimination is achieved by using the metric elevation information of the 3D point cloud as shown in Figure 3.25. The road and building facades are depicted with different colors and markers. The corresponding 3D planar model is depicted in Figure 3.28. In Figure 3.29, an aerial view of the reconstructed are of Figure 3.28 is given.

In Figure 3.30, the 3D reconstruction and planar modeling result is depicted for another sequence in the dataset, together with the camera pose estimates. It should be noted that the algorithm does not extract planes in some areas both in Figure 3.28 and Figure 3.30(b). These areas are mostly composed of trees and bushes (see the yellow boxed area in Figure 3.31 and Figure 3.32) and there are no buildings one side of the road that will form a structured point cloud which can lead to planar patches. The lack of structure in these areas can also be observed from the accumulated 3D point clouds in Figure 3.24 and Figure 3.30(a). In Figure 3.31, an aerial view of the reconstructed are of Figure 3.30 is given.

The global positions of the initial frames for the sequences in Figure 3.28 and Figure 3.30 are given in Table 3.1.

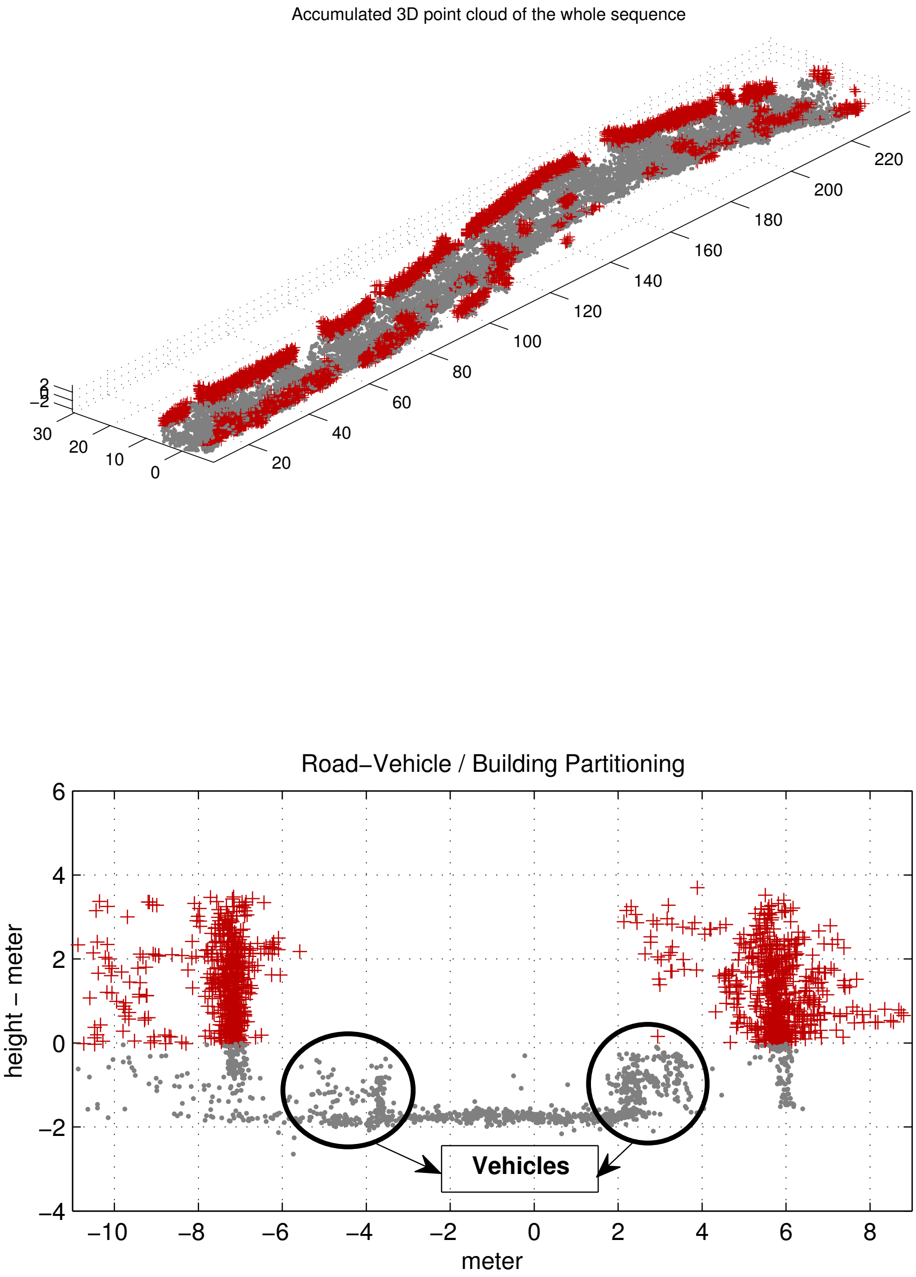
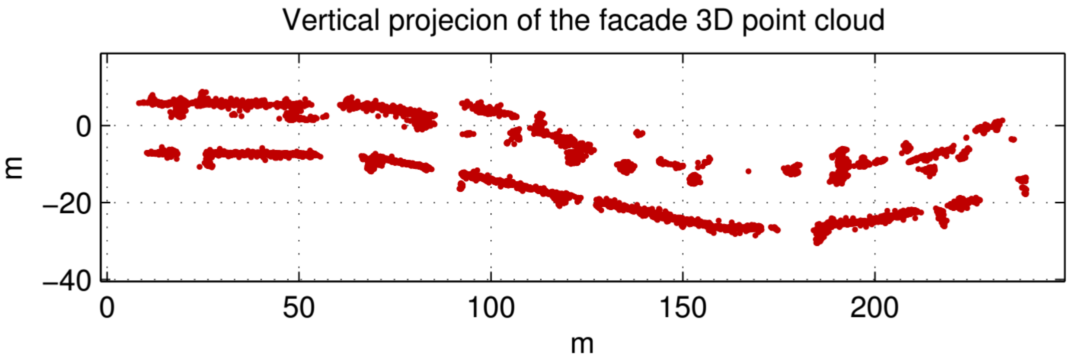


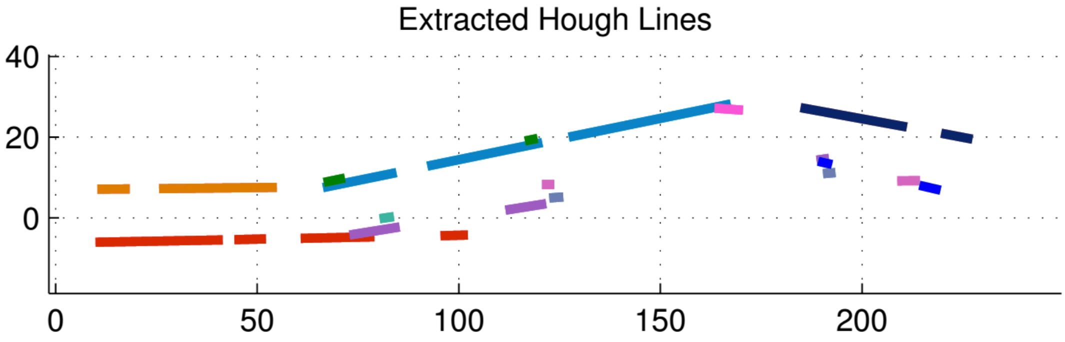
Figure 3.25: Road+Vehicle - Building Discrimination in the 3D point cloud based on height data.

Table3.1: The global positions of initial frames for sequences in Figure 3.28 and Figure 3.30.

Sequence	**Initial Latitude**	**Initial Longitude**
Figure 3.28	48.99627904 deg	8.46957902 deg
Figure 3.30	48.98511761 deg	8.39399876 deg



(a) Vertical projection of the 3D point cloud in the direction of gravity



(b) Detected lines by iterative Hough transform

Figure 3.26: Vertical projection of the accumulated 3D point cloud that belong to the building facades and extracted lines

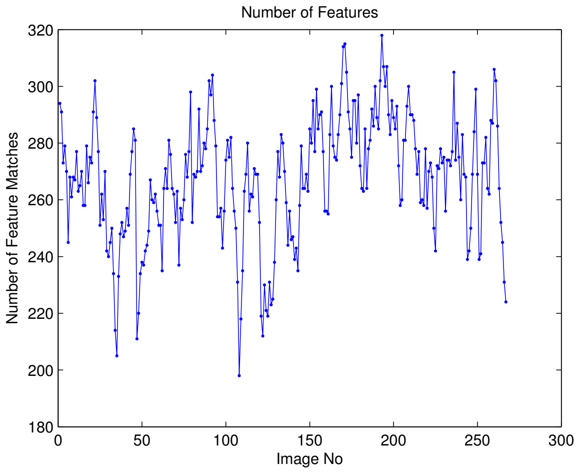


Figure 3.27: Number of Features throughout the sequence, average number is 250

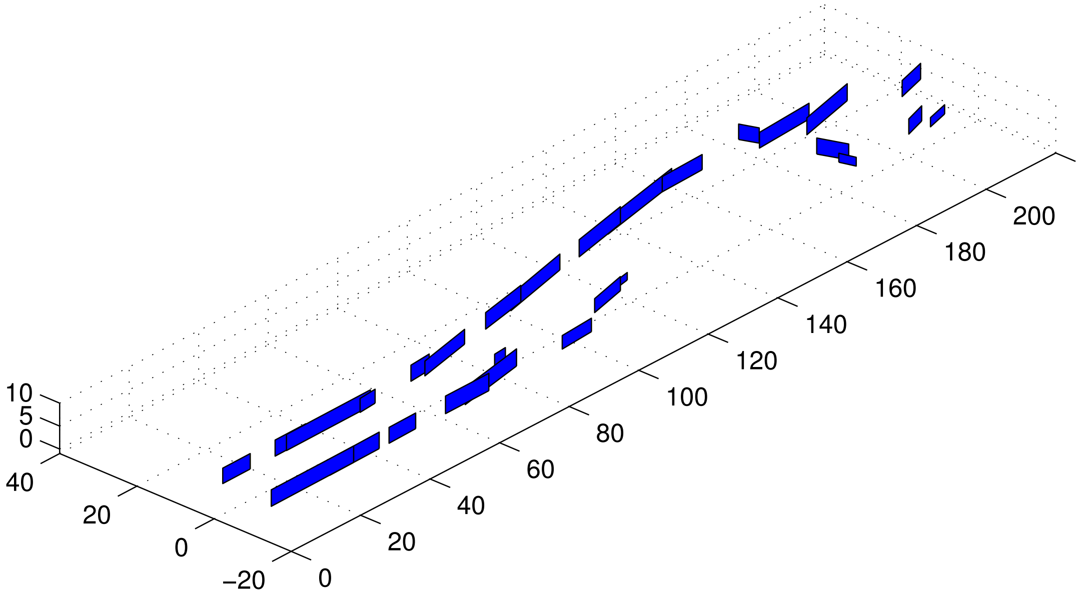


Figure 3.28: Bounding box modeling of the buildings

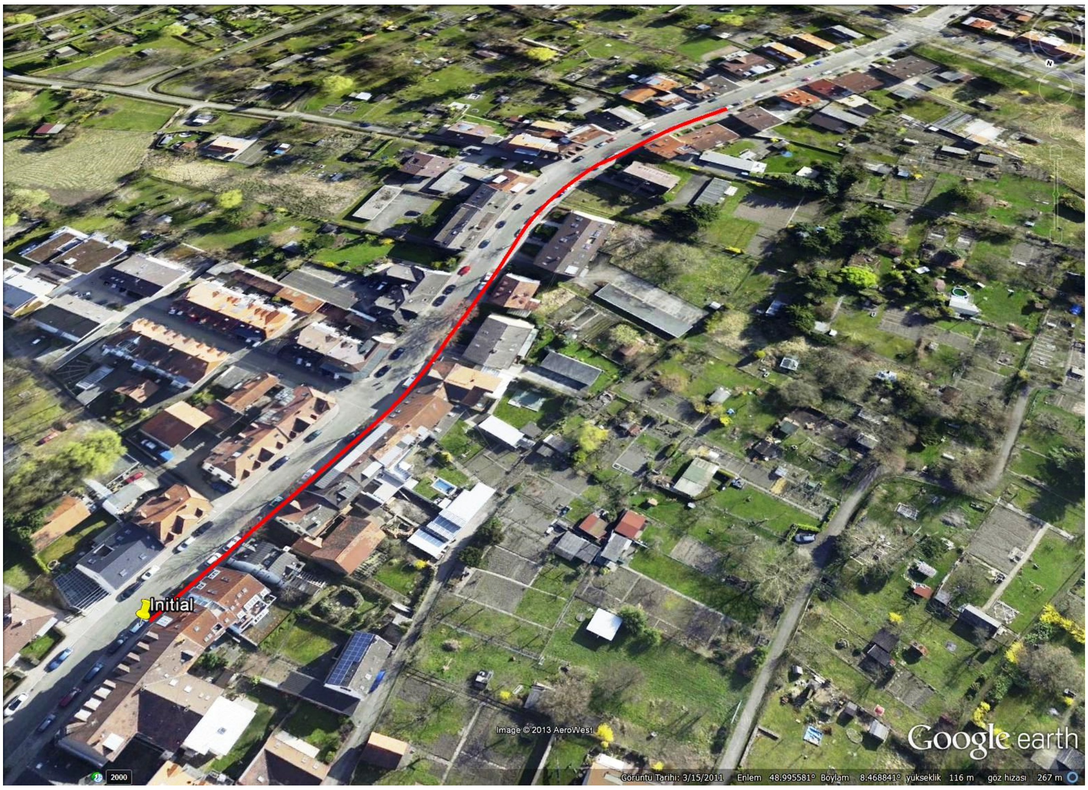
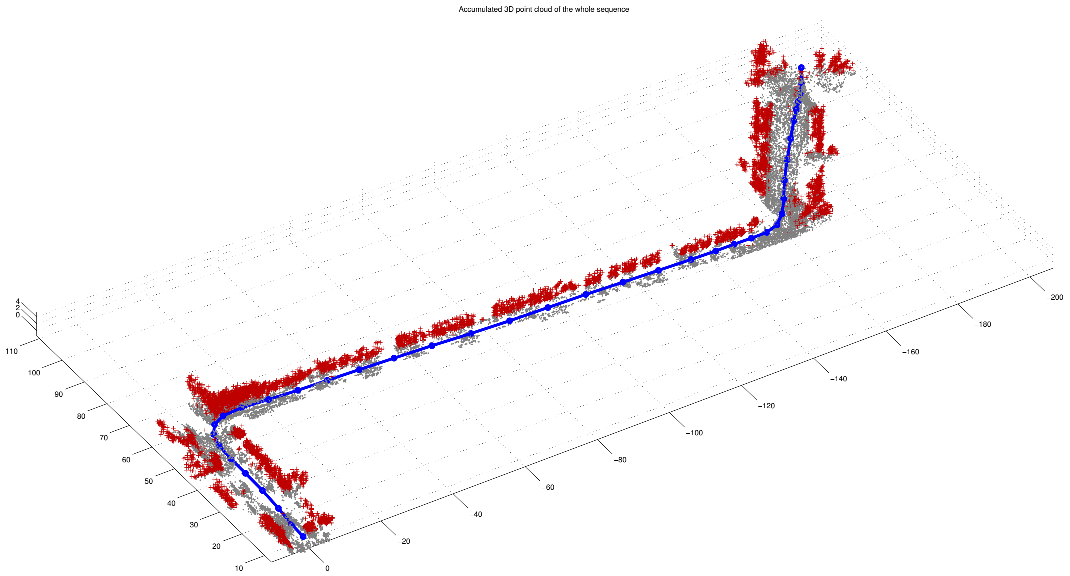
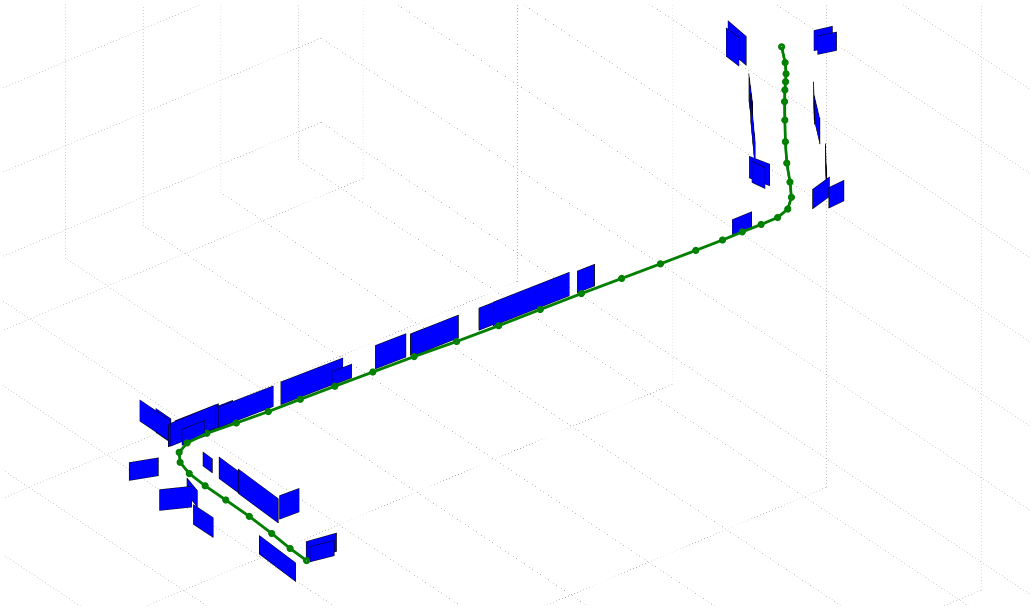

Initial
Image © 2013 AeroWest
Google earth
2000
Görüntü Tarihi: 3/15/2011 Enlem 48.995581° Boylam 8.468841° yükseklik 116 m göz hizası 267 m

Figure 3.29: An aerial view of the reconstructed sequence of Figure 3.28.



(a) Accumulated and segmented 3D point cloud



(b) Planar 3D model

Figure 3.30: 3D Reconstruction results for another sequence in the dataset

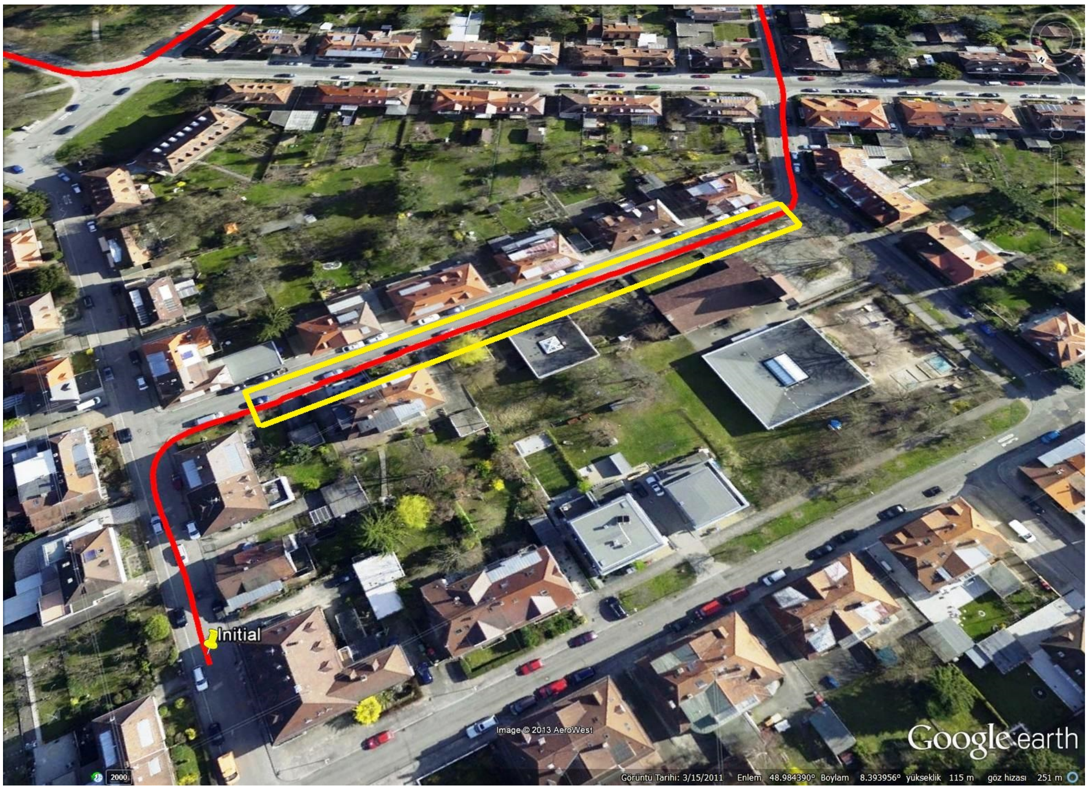

Initial
Image © 2013 AeroWest
Google earth
2000
Görüntü Tarihi: 3/15/2011 Enlem 48.984390° Boylam 8.393956° yükseklik 115 m göz hizası 251 m

Figure 3.31: An aerial view of the reconstructed sequence of Figure 3.30. The yellow box indicates the area with trees and bushes in one-side of the road as shown in Figure 3.32.

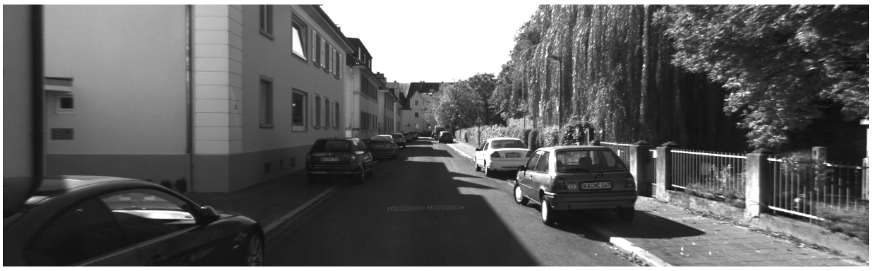


Figure 3.32: A scene from the sequence of Figure 3.30. Right side of the road is covered with trees and bushes.

3.4 Conclusion

In this chapter, a simple and efficient solution approach is proposed for 3D modeling of urban environments by plane fitting to sparse 3D point clouds obtained from SfM. The 3D plane search is down-scaled to 2D line search by means of the verticality assumption. In comparison to widely used conventional plane sweeping stereo algorithm, the proposed approach emphasizes the building heuristics more strongly. Instead of the pixel based cost minimization searches for plane association, a superpixel based direct plane association approach is proposed which is appropriate for the planar-patched structure of the urban environments. The output of the proposed algorithm follows only the building facades given the verticality assumption and is a skeletonized representation. In contrary to the conventional plane sweeping, the proposed algorithm is not able to recover slanted surfaces and objects far from the building facades, such as trees or bushes. This is an intentional preference, which makes it possible to have a computationally efficient algorithm and obtain a compact representation of the explored environment. The simplification of the model is carried one step further and the planar verticality assumption is used to exploit the 3D point cloud and extracted lines directly to get planar patches without going back to the 2D image domain. Such an approach might lead to real-time and on-the-fly implementations. The experimental results are in coherence with the propositions, it is possible to obtain more clean and compact results simpler and faster compared to the conventional plane sweeping stereo algorithm.

The importance of reliable and geo-located camera pose estimation is emphasized, especially for long sequences and large-scale urban reconstruction problem. A combined visual-inertial pose estimation framework is proposed in Chapter 4 for this purpose.

CHAPTER 4

VISUALLY AIDED INERTIAL POSE ESTIMATION

In the previous chapter, dense 3D model of an urban environment is constructed based on the assumption that extrinsic calibration parameters (6-dof poses) of the cameras are known. In Section 2.5, purely vision based 6-dof pose estimation methods are discussed. These methods have inherent ambiguities due to their combined treatment to camera pose and scene geometry estimation. Moreover, visual pose estimation is highly dependent on the feature matches and the solution diverges when the features are unreliable and/or the number of feature matches are small. On top of these, vision only pose estimation suffers from drift.

A solution might be based on embedding a radio-based navigation sensor, such as GPS to the camera for pose estimation. However, GPS requires clear visibility of the satellites, which may not be possible in dense urban environments. Moreover, GPS only provides 3D position and azimuth angle of the platform; it can not provide the remaining two Euler angles, roll and pitch, which are also necessary for reconstruction.

The 6-dof pose estimation problem is solved in marine and aeronautics systems, especially in military, by utilizing inertial navigation sensors (gyroscope and accelerometer). Due to limitations such as complexity, power requirements and price, these systems were not feasible for commercial applications up to the end of the 1990's. Recent advances in the micro-electro mechanical systems (MEMS) have made it possible to use inertial sensors, as the price and power requirements decrease, while the processing power increases. However, high quality MEMS inertial sensors, that will lead to reliable navigation solutions alone, are still not off-the-shelf due to technological limitations. Therefore, the drift problem of the dead reckoning based inertial navigation is severe, when the sensor technology is MEMS. It is fair to state that current MEMS inertial sensor technology is not sufficient for an inertial-only navigation system. Therefore, MEMS based inertial navigation systems must be supported by other sensors that can provide a measure of position, velocity or attitude (GPS, odometer, radar, barometer etc.). GPS is the most common sensor modality for aiding inertial navigation systems [52]. However, difficulties arise, when the GPS signal is blocked by buildings, degraded by multipath effects, or jammed. In these situations, the navigation solution quickly becomes invalid, especially when the inertial sensor quality is low as in MEMS case.

Being the primary source of navigation for most of the living creatures, vision has been adopted as a navigation modality for manmade machines for the last three decades. Indeed, manual methods that make use of visual measurements of fixed objects is an ancient art of navigation. Celestial observations helped the earliest global travelers for long-distance navigation. Observation of stars, measuring angles between horizon and stars, precision timekeeping made it possible to find the latitude and longitude directly [53]. However, autonomous methods that make use of visual measurements as a navigation modality, had have to wait the improvements in sensors and processing units.

Recent advances in vision research have shown great potential for vision sensors to be used as a navigation sensor modality itself, and also be integrated with inertial navigation systems to increase reliability. With a wealth of information available in each captured image, camera-based systems have a large margin for growth as onboard sensors. While vision based aiding may not produce the same kind of absolute positioning information that a GPS solution typically provide, it could still allow the navigation system to operate during periods of GPS outage and reduce the drift significantly. Indeed there are vision based systems which can provide absolute position information based on matching of the observed environment with a GIS map, but that subject is beyond the scope of this thesis.

Integration of inertial sensors with vision sensors for navigation is an active research topic which is formulated as a stochastic estimation filter problem. Despite the various instances of estimation filters, the underlying concepts remain the same. In general, one is interested in combining a known dynamic model of the analyzed system by measurements of the states of that system. A common approach among navigation systems is to use a propagation model for the IMU, instead of an assumed dynamical model [54] in a Kalman Filter framework. The drawback of using dynamic models for state transition is that the state estimate will only be as good as the dynamic model used in the filter. However, high rate IMU should trace the dynamics of the system better.

The integration schemes can be divided as tightly coupled and loosely coupled, based on the interpretation of the visual measurements in the Kalman Filter. In tightly coupled integration, the early steps of visual measurements are embedded to the estimation process. For ultra-tightly integrations, feature matching or tracking steps are also supported by inertial navigation system [53] [55]. In general, feature locations are utilized as visual measurements in tightly coupled approaches. The features need not to be augmented to the state vector as opposed to SLAM based estimation for practical purposes [56]. Epipolar constraints are also enforced in some applications for real-time implementation considerations [57] [58]. On the other hand, in the loosely coupled integration the output of visual motion estimation stage is used as the measurement. Rotational and translational components of visual motion estimates are used separately or together depending on construction of the integration filter. The filter requires only the uncertainty of the visual motion estimate in this case.

In this chapter, a camera is explored as an alternative for inertial navigation aiding in the context of an Extended Kalman Filter. Both loosely coupled and tightly coupled integration schemes are utilized for this purpose. In the loosely coupled scheme, visual odometry is used as an delta-position and attitude change aiding source for the inertial navigation system. In the tightly coupled scheme, 2D feature match locations are utilized as an aiding source for the inertial navigation system. Basics of inertial navigation and discrete Kalman Filter are also discussed for the sake of completeness.

4.1 Inertial Navigation

Inertial sensors, gyroscope and accelerometer, lie at the heart of an inertial navigation system. A gyroscope measures angular rotation with respect to inertial space about its input axis. An accelerometer measures the acceleration relative to free fall along its input axis. (The acceleration relative to free fall, i.e., specific force, is the difference between total acceleration and gravitational acceleration). Multiple gyroscopes and accelerometers can be combined to create various inertial systems. Three mutually orthogonal gyroscopes and three mutually orthogonal accelerometers form an inertial measurement unit (IMU) that is commonly used in most of the inertial navigation systems. In this section, derivation of navigation parameters (linear position, velocity and angular position) from strap-down IMU outputs will be presented. The derivations mainly follow the formulation in [59] and [60]. The basic functions that are executed in a strap-down inertial navigation system, whose operation depends on laws of classical mechanics, are the integration of INS angular rate sensors (gyroscope) data into attitude (denoted as attitude integration), use of attitude data to transform INS accelerometer data from sensor coordinates (the body, b-frame) into navigation coordinates (n-frame), integration of the n-frame acceleration into velocity (denoted as velocity integration), and integration of n-frame velocities into position (denoted as position integration).

The equations in following sections are given in the navigation (north-east-down) frame, n-frame. Other forms of these equations also exist in inertial, ECEF (Earth Centered Earth Fixed Frame), Wander frames. The details of frame and coordinate system definitions are given in Appendix A.

4.1.1 Kinematic Inertial Navigation Equations

Before the kinematic equations for navigation may be derived, some preliminary notes should be stressed:

- The turn rate of the n-frame w.r.t. the earth fixed frame is governed by the motion of the vehicle. This turn rate is called the transport rate and denoted by $\boldsymbol{w}_{EN}$.
- Earth's rotation is denoted by $\boldsymbol{w}_{IE}$ and is approximately 15 degrees/hour.

- The accelerometer measures the specific force in inertial frame which can be expressed as a combination of the second derivative of the position $\boldsymbol{r}$ and gravity $\boldsymbol{g}$. Therefore, the gravitational effect must be compensated before integration for velocity.

$$\boldsymbol{f}^B \;=\; \frac{d^2\boldsymbol{r}}{dt^2}|_i \;-\; \boldsymbol{g} \tag{4.1}$$

- The local gravity vector which includes the effect of the mass attraction of the earth and the centripetal acceleration caused by the earth's rotation is,

$$\boldsymbol{g}_\ell^N \;=\; \boldsymbol{g} \;-\; \boldsymbol{w}_{IE}^N \;\times\; (\, \boldsymbol{w}_{IE}^N \;\times\; \boldsymbol{r} \,) \tag{4.2}$$

- A gyroscope measures the angular rate w.r.t the inertial frame about its input axis.

Attitude Integration:

The attitude is expressed as the C_B^N matrix (directional cosine matrix from body frame to n-frame). The rate of change of the C_B^N matrix is given as,

$$\dot{\mathrm{C}}_B^N \;=\; \mathrm{C}_B^N \;\cdot\; \Omega_{NB}^B \tag{4.3}$$

Ω_{NB}^B is the skew-symmetric form of the angular rate vector $\boldsymbol{w}_{NB}^B$ (rate of body w.r.t. n-frame expressed in body frame). A gyroscope measures not only the body rate but also the earth rate and the transport rate as given,

$$\boldsymbol{w}_{IB}^B \;=\; \boldsymbol{w}_{NB}^B \;+\; C_B^N\, (\, \boldsymbol{w}_{IE}^N \;+\; \boldsymbol{w}_{EN}^N \,) \tag{4.4}$$

Therefore rate of body w.r.t. n-frame expressed in body frame becomes,

$$\boldsymbol{w}_{NB}^B \;=\; \boldsymbol{w}_{IB}^B \;-\; \mathrm{C}_B^N\, (\, \boldsymbol{w}_{IE}^N + \boldsymbol{w}_{EN}^N \,) \tag{4.5}$$

where $\boldsymbol{w}_{IE}^N = [\, \Omega\cos(L) \quad 0 \quad -\Omega\sin(L)\,]^T$ is the earth rate at the current latitude, L, expressed in n-frame, and $\boldsymbol{w}_{EN}^N = [\, \frac{v_E}{R_E} \quad \frac{-v_N}{R_N+h} \quad \frac{-v_E\tan(L)}{R_E+h}\,]^T$ is the transport rate (the turn rate of n-frame w.r.t. e-frame).

Velocity Integration:

The rate of change of velocity expressed in n-frame is:

$$\dot{\boldsymbol{v}}_E^N \;=\; \boldsymbol{f}^N \;+\; \boldsymbol{g}_\ell^N \;-\; (\, 2\, \boldsymbol{w}_{IE}^N \;+\; \boldsymbol{w}_{EN}^N \,) \;\times \boldsymbol{v}_E^N \tag{4.6}$$

where $(\, 2\, \boldsymbol{w}_{IE}^N \times \boldsymbol{v}_E^N \,)$ is the Coriolis acceleration term, $(\, \boldsymbol{w}_{EN}^N \times \boldsymbol{v}_E^N \,)$ is the transport rate term and $\boldsymbol{f}^N \;=\; \mathrm{C}_B^N \cdot \boldsymbol{f}^B$ is the acceleration data expressed in n-frame.

Position Integration:

The rate of change of velocity expressed in n-frame is $\dot{\boldsymbol{r}}^N$:

$$\dot{\boldsymbol{r}}^N \;=\; \mathrm{M} \cdot \boldsymbol{v}^N, \qquad \mathrm{M} \;=\; \begin{pmatrix} \frac{1}{R_N+h} & 0 & 0 \\ 0 & \frac{1}{(R_E+h)\ \cos(L)} & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{4.7}$$

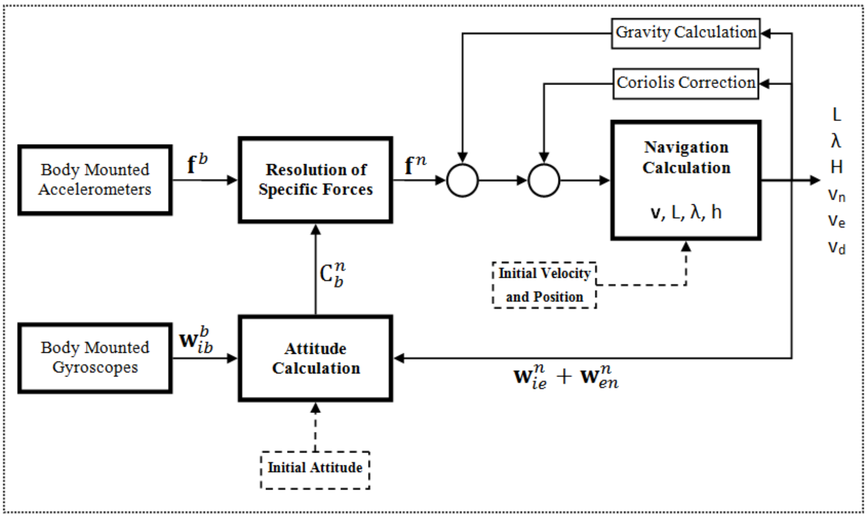


Figure 4.1: Block Diagram Representation of the Navigation Frame Mechanization of INS

where the position vector $\dot{\boldsymbol{r}}^N = [\,L \quad \lambda \quad h\,]$. Here L is latitude, λ is longitude, and h is the height. R_N and R_E are the radii of curvature along lines of constant longitude λ and latitude L. Navigation frame mechanization is summarized as a block diagram in Figure 4.1. Direction cosine matrix is calculated by solving (4.3), (4.4) and (4.5) using the gyroscope outputs, transport and earth rate vectors. Once the accelerometers' output vector is mapped to navigation coordinate system using the calculated direction cosine matrix then it is compensated for both the gravitational and Coriolis forces. The resultant value is integrated to obtain the velocity with respect to Earth frame, which is expressed in navigation coordinate system. Then time rate change of position vector (latitude, longitude and height) is obtained using (4.7). The integration of the resultant value gives the position vector. Due to inevitable inertial measurement and initialization errors, the inertial navigation solution drifts in time.

4.2 Discrete Kalman Filter

Kalman filter provides a method of automatically weighting all measurements based on their statistical worth. The Kalman Filter addresses the general problem of trying to estimate the state of a discrete-time controlled process, $\boldsymbol{x} \in R_n$, which is governed by the linear stochastic difference equation,

$$\begin{aligned} \boldsymbol{x}_k &= \mathrm{A}\,\boldsymbol{x}_{k-1} + \mathrm{B}\,\boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1} \\ z_k &= \mathrm{H}\,\boldsymbol{x}_k + \boldsymbol{v}_k \end{aligned} \tag{4.8}$$

Where the random variables $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ represent the process and measurement noise (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions,

$$\begin{aligned} p(\boldsymbol{w}) &\sim N(0, Q) \\ p(\boldsymbol{v}) &\sim N(0, R) \end{aligned} \tag{4.9}$$

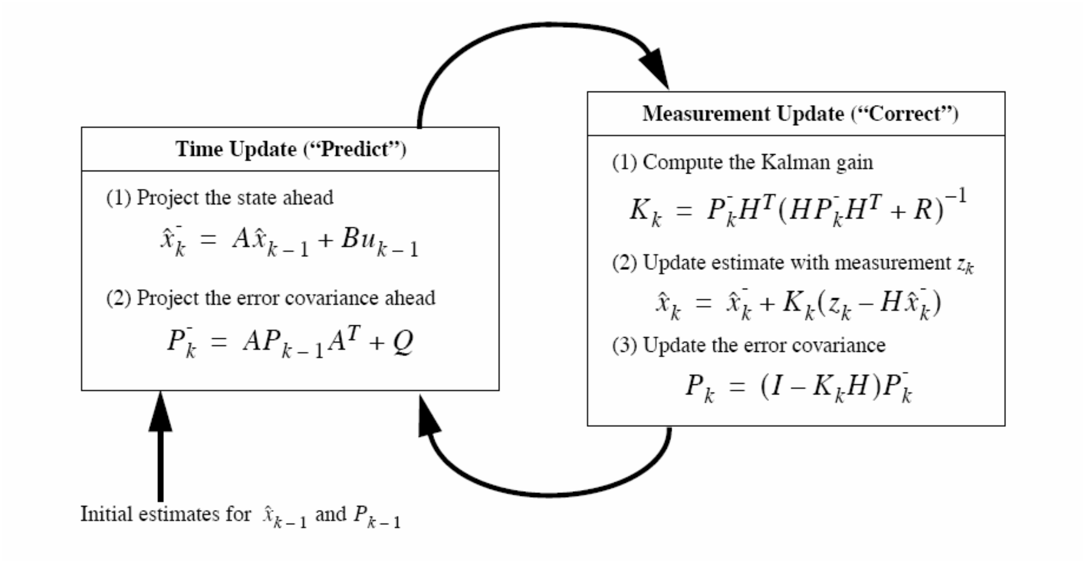

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Figure 4.2: A complete picture of the operation of the Discrete Kalman Filter [6]

The Kalman filter estimates a process by using a form of feedback control: the filter estimates the process state at some time and then obtains feedback in the form of (noisy) measurements. As such, the equations for the Kalman filter fall into two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the a priori estimates for the next time step. The measurement update equations are responsible for the feedback, i.e. for incorporating a new measurement into the a priori estimate to obtain an improved a posteriori estimate.

In general, an EKF may be thought of as a standard Kalman filter, but with the system and measurement models linearized about the current state estimate at each time step. If a nonlinear system's state evolves according to, $\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1})$, its Taylor series expansion with respect to the current state estimate $\hat{\boldsymbol{x}}_{k-1}$ is computed as:

$$\begin{aligned} \boldsymbol{x}_k &= f(\boldsymbol{x}_{k-1}) \\ &= f(\hat{\boldsymbol{x}}_{k-1}) + \frac{\delta f(\boldsymbol{x}_{k-1})}{\delta \boldsymbol{x}} |_{x=\hat{x}_{k-1}} \end{aligned} \tag{4.10}$$

neglecting higher order terms.

4.2.1 Direct vs Indirect Kalman Filter

For the filter in visual inertial integration application, an indirect Kalman Filter structure will be utilized and the states to be estimated will be the errors in the overall navigation states. In the direct Kalman Filter formulation total states such as position and velocity are among the state variables in the filter, and the measurements are IMU outputs and external source signals (odometer, GPS etc.). In the Indirect Kalman Filter formulation, the errors in the INS-indicated position and velocity are among the estimated variables, and each measurement presented to the filter is the difference between INS and external source data [61].

In the direct filter formulation, the Kalman Filter is in the INS loop. The IMU sensor data and external source data both fed into the filter. In the indirect formulation, Kalman Filter estimates the errors in the navigation and attitude information using the difference between INS and external source data. The INS itself follows the high frequency motions of the vehicle and there is no need to model these dynamics explicitly in the filter. Instead, the dynamics upon which the filter is based is the set of inertial system error propagation equations, which are relatively well developed, well behaved, low frequency and very adequately represented as linear. Since the filter is out of the INS loop and is based on low frequency linear dynamics, its sample rate can be much lower than that of a direct filter.

To sum up, such an indirect approach is preferred to use error states instead of total (actual) states in the formulation due to the following reasons:

- Kalman filter might execute in lower rates than the inertial process rate.
- Linearization of the total state equations is cumbersome; whereas linearization of the error states is easier.
- The indirect filter mechanization is more reliable; if the filter should happen to fail in direct mechanization, the entire navigation system fails; the inertial system can not operate without the filter. It is not the case in indirect formulation.

4.2.2 Feedforward vs Feedback Kalman Filter

In a Feedforward Kalman Filter, the estimated errors in the inertial system are fed to the output to maintain optimal estimates of position, velocity and attitude. The inertial system operates as if there were no aiding: i.e. it is unaware of the existence of the filter or the external data. However, acceptable Kalman Filter performance depends upon the adequacy of a linear dynamics model, so it is necessary for the inertial navigation errors to remain in small magnitude. The inertial system in feedforward configuration is free to drift with unbounded errors; thereby, invalidating this basic assumption. The structure of the indirect feedforward Kalman filter is given in Figure 4.3.

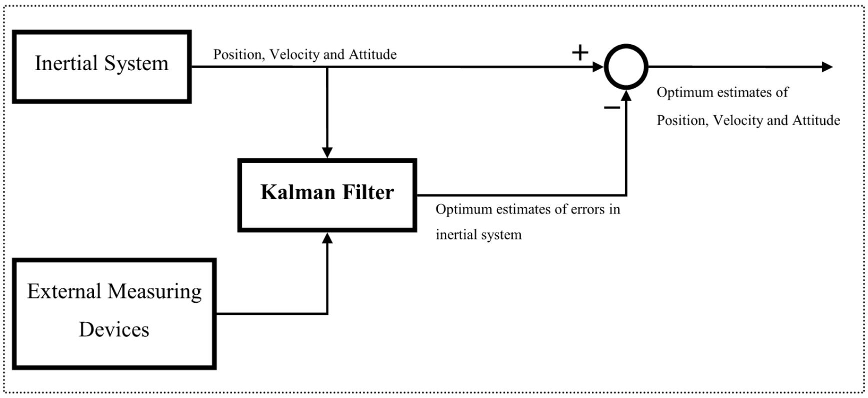

Inertial System
Position, Velocity and Attitude
+
−
Optimum estimates of
Position, Velocity and Attitude
Kalman Filter
Optimum estimates of errors in
inertial system
External Measuring
Devices

Figure 4.3: Indirect Feedforward Kalman Filter structure

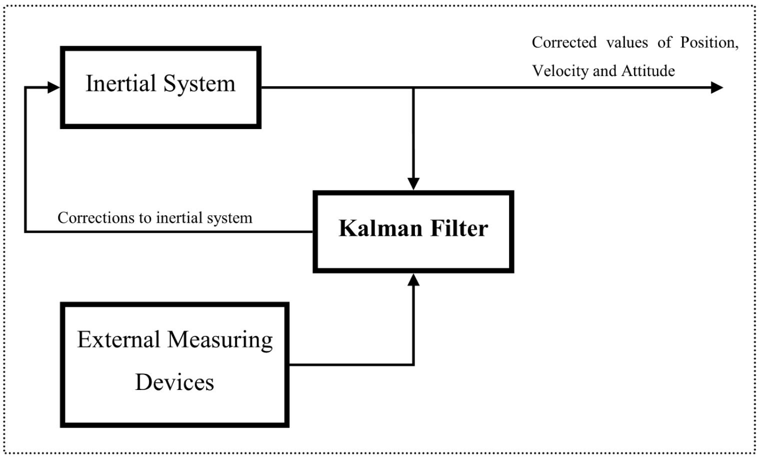


Figure 4.4: Indirect Feedback Kalman Filter structure

To maintain the linearity assumption of the Kalman filter, an indirect feedback configuration is motivated in this dissertation. The Kalman filter generates estimates of the errors in the inertial system, and these errors are fed back into the INS to correct it. In this way, the inertial errors are not allowed to grow unchecked, and the adequacy of a linear model is enhanced. There is a further advantage of this configuration. Since the INS is corrected after each measurement sample, many of the predicted error states at the next sample time will be zero, thus these components need not be computed explicitly. The structure of the indirect feedback Kalman filter is given in Figure 4.4.

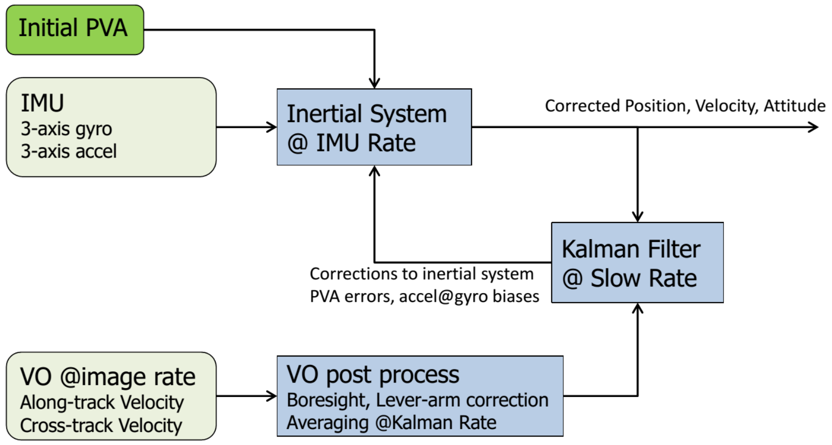


Figure 4.5: Loosely Coupled Indirect Feedback Kalman Filter structure

4.3 Loosely Coupled Visual Inertial Integration

Inertial navigation solution, which is obtained by the integration of the acceleration in a reference frame (navigation frame) maintained by the high rate gyroscopes and accelerometers, supplies high dynamic information of the platform. However, due to the inevitable inertial measurement and initialization errors, the inertial navigation solution suffers from drift and should be compensated with aiding sensors. On the other hand, the visual odometry can generate relatively bounded but noisy delta-position and attitude change information for short time periods as defined in Section 2.5. Instead of the direct fusion of the high rate inertial sensors and relatively low rate VO output which will cause a loss in high dynamic information obtained by inertial sensors (due to low-pass characteristics of Kalman filter), a traditional indirect (error-state) Kalman Filtering method [62] is proposed to utilize the complementary characteristics of inertial sensors and visual odometry. In this method, the errors of the navigation states are estimated by using the error state measurements constructed by the difference of the VO solution and navigation solution rather than directly estimating the navigation states (attitude, velocity, position). In addition to the navigation and inertial sensor errors (bias, scale factor), systematic visual odometry errors such as scale factor and bias are also augmented to the state vector of the indirect Kalman Filter. The estimated errors are then feedback to the inertial navigation integration (instead of feed-forward) in order not to violate the linearity assumption of error dynamics by eliminating the error growths in navigation variables. The structure of the proposed indirect feedback Kalman Filter is depicted in Figure 4.5. The main steps of the INS-Visual Odometry Integration can be summarized as follows:

1. Initialization of the filter

2. Navigation Update (at IMU rate)

3. Kalman Time Update (at image rate)
4. Kalman Measurement Update (at image rate)
5. Kalman Error State Feedback (at image rate)

4.3.1 Total System Error Model

In this section, a generic truth error model is constructed for the loosely coupled filter. The total error sources that may affect the system performance are modeled by a stochastic linear dynamics. The corresponding error state vector is given as,

$$\delta\boldsymbol{x}^T = [\delta\boldsymbol{\phi}^N\ \delta\boldsymbol{v}^N\ \delta\boldsymbol{r}^N\ \delta\boldsymbol{f}^B\ \delta\boldsymbol{\omega}^B\ \delta\boldsymbol{m}_{sf}\ \delta\boldsymbol{m}_b\ \delta\boldsymbol{m}_{bs}\ \delta\boldsymbol{m}_{la}] \tag{4.11}$$

Here $\delta\boldsymbol{\phi}^N$ is the perturbed attitude tilt error, $\delta\boldsymbol{v}^N$ is the perturbed velocity error and $\delta\boldsymbol{r}^N$ is the perturbed position error. $\delta\boldsymbol{f}^B$, $\delta\boldsymbol{w}^B$ are the accelerometer and gyro error vectors respectively. Each of these vectors includes the systematical errors of the inertial sensor such as bias, scale factor, misalignment errors. $\delta\boldsymbol{m}_{sf}$ is the visual odometry scale factor error and $\delta\boldsymbol{m}_b$ is the visual odometry correlated bias error that is induced by the visual odometry algorithms under different scene conditions. $\delta\boldsymbol{m}_{bs}$, $\delta\boldsymbol{m}_{la}$ are the angular (boresight) and translational (lever-arm) displacement between the camera and the IMU, respectively. The INS error state dynamics is well studied subject in inertial navigation community. The perturbation form of the error state dynamics can be expressed in a compact form as [59],

$$\begin{aligned}
\delta\dot{\boldsymbol{\phi}}^N &= -(\times\boldsymbol{\omega}^N_{IN})\delta\boldsymbol{\phi}^N + \delta\boldsymbol{\omega}^N_{IN} - C^N_B\delta\boldsymbol{\omega}^B \\
\delta\dot{\boldsymbol{v}}^N &= -(\times C^N_B\boldsymbol{f}^B)\delta\boldsymbol{\phi}^N - (2\boldsymbol{\omega}^N_{IE} + \boldsymbol{\omega}^N_{EN}) \times \delta\boldsymbol{v}^N - (2\delta\boldsymbol{\omega}^N_{IE} + \boldsymbol{\omega}^N_{EN}) \times \boldsymbol{v}^N + C^N_B\delta\boldsymbol{f}^B + \delta\boldsymbol{g}^N \\
\delta\dot{\boldsymbol{r}}^N &= \delta\boldsymbol{v}^N
\end{aligned} \tag{4.12}$$

Here $\boldsymbol{\omega}^N_{IN}$ is the angular velocity of the navigation frame with respect to inertial frame and $\boldsymbol{\omega}^N_{EN}$ is the angular velocity of the navigation frame with respect to the Earth frame and $\delta\boldsymbol{\omega}^N_{IN}, \delta\boldsymbol{\omega}^N_{EN}$ are the perturbation errors of those quantities. The IMU errors are generally modeled by first-order Gauss-Markov process and/or random walk process. The visual odometry errors scale factor, bias and lever-arm errors can be modeled by random walk processes with a suitable noise density.

4.3.2 Initialization of the Filter

Initial position, velocity and attitude of the vehicle, together with their uncertainties, shall be given to the filter to start estimation. These parameters can be supplied by the user or can be extracted from other sensors (GPS, magnetic compass). Initial position and velocity is easier to access. The system can be started from a known position and while it is stationary (zero initial velocity). However, it is not easy to get the initial attitude. An alignment process is required, if the initial attitude is not supplied externally. Alignment is the process whereby the orientation of the axes of an inertial navigation system is determined with respect to the reference axis system [59]. Accurate alignment is crucial, if precise navigation is to be achieved over long periods of time.

Alignment process consists of finding all three Euler angles: roll, pitch and heading. These angles are defined with respect to the north oriented local level frame. Initial roll and pitch angles can be determined by using the accelerometers by means of a leveling process, if the system is stationary. The horizontal components of the gravity acting in the north and east directions are nominally zero. The system is rotated (mathematically) until the outputs of the north and east accelerometers reach a null, thus leveling the platform. For the heading angle, given the latitude and earth rate, the components of earth rate on gyroscopes are used for estimation.

As described above, the objective of the alignment process is to determine the initial direction cosine matrix, C_B^N, which relates the body and geographic reference frames. The body mounted sensors will measure components of the specific force needed to overcome gravity and components of Earth's rate, denoted by the vector quantities $\boldsymbol{g}^B$ and $\boldsymbol{w}_{IE}^B$, respectively. These vectors are related to the gravity and Earth's rate vectors specified in the local geographic frame, $\boldsymbol{g}^N$ and $\boldsymbol{w}_{IE}^N$, respectively, in accordance with the following equations [59]:

$$\begin{aligned} \boldsymbol{g}^B &= \mathrm{C}_B^N\, \boldsymbol{g}^N && (4.13) \\ \boldsymbol{w}_{IE}^B &= \mathrm{C}_B^N\, \boldsymbol{w}_{IE}^N && (4.14) \end{aligned}$$

where $\boldsymbol{g}^N = [0 \quad 0 \quad -g]^T$ and $\boldsymbol{w}_{IE}^N = [\Omega\cos(L) \quad 0 \quad -\Omega\sin(L)]^T$ in which Ω and L denotes Earth's rate and latitude, respectively. Given knowledge of these quantities, estimates of the elements of the directional cosine matrix may be computed directly from the measurements of $\boldsymbol{g}^B = [g_x \quad g_y \quad g_z]^T$ and $\boldsymbol{w}_{IE}^B = [w_x \quad w_y \quad w_z]^T$ as follows:

$$\begin{aligned} c_{31} &= -\tfrac{g_x}{g} & c_{11} &= \frac{w_x}{\Omega\cos(L)} - \frac{g_x\tan(L)}{g} \\ c_{32} &= -\tfrac{g_y}{g} & c_{12} &= \frac{w_y}{\Omega\cos(L)} - \frac{g_y\tan(L)}{g} \\ c_{33} &= -\tfrac{g_z}{g} & c_{13} &= \frac{w_z}{\Omega\cos(L)} - \frac{g_z\tan(L)}{g} \end{aligned} \qquad (4.15)$$

where $c_{11}, c_{12}, \ldots, c_{33}$ are the elements of the direction cosine matrix C_B^N. The remaining direction cosine elements may be determined by making use of the orthogonality properties of the direction cosine matrix that yield:

$$\begin{aligned} c_{21} &= -c_{12}c_{33} + c_{13}c_{32} \\ c_{22} &= c_{11}c_{33} - c_{31}c_{13} \\ c_{23} &= -c_{11}c_{32} + c_{31}c_{12} \end{aligned} \qquad (4.16)$$

For the MEMS grade IMU, leveling is feasible; however, gyro-compassing is not possible, since the earth rate cannot be discriminated from the gyroscope noise. Therefore, at least the heading angle should be provided from an external source. GPS can be used for this purpose. The GPS device outputs heading angle, while in motion from the resolution of north and east velocities.

In addition to the initial position, velocity and attitude, other filter parameters should also be initialized. For example, an initial gyro bias value should be assigned based on manufacturer's datasheet and the process noise related to its states should also be defined.

4.3.3 Navigation Update

The navigation update consists of attitude integration, velocity integration and position integration stages as defined in Section 4.1. The navigation update runs at IMU rate.

4.3.4 Kalman Time Update

Kalman Time Update might run at image rate. However, visual odometry cannot guarantee that the measurement noise $\boldsymbol{v}$ is normally distributed or zero mean given the typical complexity of such an algorithm. Without these constraints, a Kalman filter is suboptimal at best and potentially divergent. Tardif et al. [63] proposed a method that utilizes central limit theorem to overcome this problem. The visual odometry measurements (along-track and cross-track velocity of the body) are accumulated before incorporated to the filter. In this manner, the total error will tend to satisfy the Kalman Filter assumptions for a sufficiently large accumulation time. This results in a slower rate than the image rate for the Kalman Time Update. The state vector is projected ahead by the following state transition formula:

$$\hat{\boldsymbol{x}}_k^- = \mathrm{A}_k \, \hat{\boldsymbol{x}}_{k-1} \tag{4.17}$$

whereas the state error covariance is projected ahead by the following relation:

$$\mathrm{P}_k^- \; = \; \mathrm{A}_k \, (\, \mathrm{P}_{k-1} \; + \; 0.5 \, Q) \, \mathrm{A}_k^T \; + \; 0.5 \, Q \tag{4.18}$$

Attitude, velocity, position, gyroscope/accelerometer bias, VO scale factor and camera-IMU lever arm error states are modelled in the following formulation. The state vector can be augmented with other variables (such as VO bias, gyroscope/accelerometer scale factor, time delay between camera and IMU etc.) depending on the effect of these state variables on system output performance. State transition matrix and measurement matrix should be modified accordingly when different variables are augmented to the state vector. State transition matrix (continuous time) for the error states is constructed as follows by the perturbation error model [59]:

Attitude to Attitude:

$$\Phi_{1:3\times1:3} = \begin{pmatrix} 0 & -(\Omega \sin L + v_E \frac{\tan(L)}{R}) & \frac{v_N}{R} \\ (\Omega \sin(L) + v_E \frac{\tan(L)}{R}) & 0 & (\Omega \cos(L) + \frac{v_E}{R}) \\ -\frac{v_N}{R} & -(\Omega \cos(L) + \frac{v_E}{R}) & 0 \end{pmatrix} \tag{4.19}$$

Velocity to Attitude:

$$\Phi_{1:3\times4:6} = \begin{pmatrix} 0 & \frac{1}{R} & 0 \\ -\frac{1}{R} & 0 & 0 \\ 0 & -\frac{\tan(L)}{R} & 0 \end{pmatrix} \tag{4.20}$$

Position to Attitude:

$$\Phi_{1:3\times7:9} = \begin{pmatrix} -\Omega\sin(L) & 0 & -\frac{v_E}{R^2} \\ 0 & 0 & \frac{v_N}{R^2} \\ -(\Omega\cos(L) + \frac{v_E}{R\cos^2(L)}) & 0 & \frac{v_E\tan(L)}{R^2} \end{pmatrix} \tag{4.21}$$

Accelerometer Bias to Attitude:

$$\Phi_{1:3\times10:12} = 0_{3\times3} \tag{4.22}$$

Gyroscope Bias to Attitude:

$$\Phi_{1:3\times13:15} = -\mathrm{C}_N^B \tag{4.23}$$

VO Scale Factor to Attitude:

$$\Phi_{1:3\times16} = 0_{3\times1} \tag{4.24}$$

VO Lever Arm to Attitude:

$$\Phi_{1:3\times17:19} = 0_{3\times3} \tag{4.25}$$

Attitude to Velocity:

$$\Phi_{4:6\times1:3} = \begin{pmatrix} 0 & -f_D & f_E \\ f_D & 0 & -f_N \\ -f_E & f_N & 0 \end{pmatrix} \tag{4.26}$$

Velocity to Velocity:

$$\Phi_{4:6\times4:6} = \begin{pmatrix} \frac{v_D}{R} & -2(\Omega\sin(L) + \frac{v_E}{R}\tan(L)) & \frac{v_N}{R} \\ (2\Omega\sin(L) + \frac{v_E}{R}\tan(L)) & \frac{1}{R}(v_N\tan(L) + v_D) & 2\Omega\cos(L) + \frac{v_E}{R} \\ \frac{-2v_N}{R} & -2(\Omega\cos(L) + \frac{v_E}{R}) & 0 \end{pmatrix} \tag{4.27}$$

Position to Velocity:

$$\Phi_{4:6\times7:9} = \begin{pmatrix} -v_E(2\Omega\cos(L) + \frac{v_E}{R\cos^2 L}) & 0 & \frac{1}{R^2}(v_E^2\tan(L) - v_Nv_D) \\ \left(2\Omega(v_N\cos(L) - v_D\sin(L)) + \frac{v_Nv_E}{R\cos^2 L}\right) & 0 & -\frac{v_E}{R^2}(v_N\tan(L) + v_D) \\ 2\Omega v_E\sin(L) & 0 & \frac{1}{R^2}(v_N^2 + v_E^2) \end{pmatrix} \tag{4.28}$$

Accelerometer Bias to Velocity:

$$\Phi_{4:6\times10:12} = \mathrm{C}_N^B \tag{4.29}$$

Gyroscope Bias to Velocity:

$$\Phi_{4:6\times13:15} = 0_{3\times3} \tag{4.30}$$

VO Scale Factor to Velocity:

$$\Phi_{4:6\times16} = 0_{3\times1} \tag{4.31}$$

VO Lever Arm to Velocity:

$$\Phi_{4:6\times17:19} = 0_{3\times3} \tag{4.32}$$

Attitude to Position:

$$\Phi_{7:9\times1:3} = 0_{3\times3} \tag{4.33}$$

Velocity to Position:

$$\Phi_{7:9\times4:6} = \begin{pmatrix} \frac{1}{R} & 0 & 0 \\ 0 & \frac{1}{R\cos(L)} & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{4.34}$$

Position to Position:

$$\Phi_{7:9\times7:9} = \begin{pmatrix} 0 & 0 & \frac{-v_N}{R^2} \\ \frac{v_E\tan(L)}{R\cos(L)} & 0 & -\frac{v_E}{R^2\cos(L)} \\ 0 & 0 & 0 \end{pmatrix} \tag{4.35}$$

Accelerometer Bias to Position:

$$\Phi_{7:9\times10:12} = 0_{3\times3} \tag{4.36}$$

Gyroscope Bias to Position:

$$\Phi_{7:9\times13:15} = 0_{3\times3} \tag{4.37}$$

VO Scale Factor to Position:

$$\Phi_{7:9\times16} = 0_{3\times1} \tag{4.38}$$

VO Lever Arm to Position:

$$\Phi_{7:9\times17:19} = 0_{3\times3} \tag{4.39}$$

All states to Accelerometer Bias:

$$\Phi_{10:12\times1:19} = 0_{3\times19} \tag{4.40}$$

All states to Gyroscope Bias:

$$\Phi_{13:15\times1:19} = 0_{3\times19} \tag{4.41}$$

All states to VO Scale Factor:

$$\Phi_{16\times1:19} = 0_{1\times19} \tag{4.42}$$

All states to VO Lever-Arm:

$$\Phi_{17:19\times3:19} = 0_{3\times19} \tag{4.43}$$

The discretization of the state transition matrix (via Taylor expansion) yields:

$$\mathrm{A}_k = \mathrm{I}_{3\times3} + \Phi_k + 0.5\ \Phi_k^2 \tag{4.44}$$

This concludes the Kalman Time Update process.

4.3.5 Visual Odometry Measurement Model

Visual odometry calculates the delta position vector and attitude change between two consecutive camera frames at image rate. Raw visual odometry measurement cannot guarantee that the measurement noise is normally distributed given the complexity of the algorithm and dynamic scene conditions. Without these guarantees, the Kalman filter is suboptimal at best.

Tardif et al. [63] proposed a method that utilizes central limit theorem to overcome this problem. The visual odometry measurements are accumulated over the Kalman Filter interval before incorporated to the filter. By this type of pre-filtering, the effect of relatively high frequency correlated noises (such as oscillatory disturbances due to vehicle engine) is also eliminated. In this manner, the total error will tend to satisfy the Kalman Filter assumptions for a sufficiently large accumulation time. This results in a slower rate than the image rate for the Kalman measurement update which is computationally efficient especially for large state filters. In addition, the indirect feedback form of the Kalman Filter encourages slow rate measurement updates due to relatively low bandwidth of the error dynamics.

4.3.5.1 Accumulated Delta Position Measurement Models

There can be two different types of accumulation for the delta position vectors. One possible way is the accumulation of the delta position vectors expressed in the instant camera coordinate system or the accumulation of the vectors expressed in the initial or the final time of the accumulation interval. In [63] the accumulation of the vectors in the initial time camera coordinate system is proposed.

Model I:

The accumulated delta position vector of visual odometry, $\Delta\boldsymbol{r}^{\emptyset_k}_{VO,k}$, at Kalman cycle ($k$) defined in the final time camera optical coordinate axis, $\emptyset_k$, for the accumulation interval ($k-1$) can be expressed as,

$$\begin{aligned}\Delta\boldsymbol{r}^{\emptyset_k}_{VO,k} &= \boldsymbol{r}^{\emptyset_k}_{VO,k} - \boldsymbol{r}^{\emptyset_k}_{VO,k-1} \\ &= [C^{\emptyset_k}_{\emptyset_{k-1}}]_{VO} \sum_{m=1}^{N} ([C^{\emptyset_{k-1}}_{\emptyset_{k-1[m-1]}}]_{VO} (M_{SF} \Delta\boldsymbol{r}^{\emptyset_{k-1[m-1]}}_{VO,k-1[m]} + \boldsymbol{m}_{b,k-1[m]} + \boldsymbol{\nu}_{k-1[m]}))\end{aligned} \tag{4.45}$$

where $\Delta\boldsymbol{r}^{\emptyset_{k-1[m-1]}}_{VO,k-1[m]}$ is the delta position vector of visual odometry at the minor interval (m) of the $(k-1)^{th}$ major interval expressed in the previous minor interval $(m-1)$ camera coordinate system and $\boldsymbol{\nu}_{k-1[m]}$ is the measurement noise vector on the minor interval's delta position vector. M_{SF} is the visual odometry scaling diagonal matrix and $\boldsymbol{m}_{b,k-1[m]}$ is the visual odometry correlated bias term. Note that, N corresponds to the number of images in the Kalman interval. $[C^{\emptyset_{k-1}}_{\emptyset_{k-1[m-1]}}]_{VO}$ is calculated by the incremental attitude change measurements recursively as,

$$C^{\emptyset_{k-1}}_{\emptyset_{k-1[m-1]}} = C^{\emptyset_{k-1}}_{\emptyset_{k-1[1]}} \cdot C^{\emptyset_{k-1[1]}}_{\emptyset_{k-1[2]}} \cdots C^{\emptyset_{k-1[m-2]}}_{\emptyset_{k-1[m-1]}} \tag{4.46}$$

On the other hand, the delta position measurement at the camera optical center can be calculated in terms of the INS computed navigation quantities as,

$$\Delta\boldsymbol{r}^{\emptyset_k}_{INS,k} = \tilde{C}^{\emptyset}_{B} (\tilde{C}^{N}_{B_k})^T (\tilde{\boldsymbol{r}}^{N}_{INS,k} - \tilde{\boldsymbol{r}}^{N}_{INS,k-1}) + \tilde{C}^{\emptyset}_{B} (I - \tilde{C}^{B_k}_{B_{k-1}}) \tilde{\boldsymbol{m}}^{B}_{la} \tag{4.47}$$

where $\tilde{C}_B^{\emptyset}$ is the estimated boresight matrix between the camera axis and IMU axis where the true value of it is denoted as $C_B^{\emptyset}$. $\tilde{\boldsymbol{m}}_{la}^B$ is the estimated lever-arm vector emanating from the INS center of navigation to the camera center expressed in body frame. It should be noted that the change of the navigation frame during the Kalman filter cycle is assumed to be negligible for the error modeling. The delta-position error measurement is constructed by the difference of the visual odometry measurement and INS computed delta position quantity as,

$$
\begin{aligned}
\delta \boldsymbol{y}_k = \Delta \boldsymbol{r}_{VO,k}^{\emptyset_k} - \Delta \boldsymbol{r}_{INS,k}^{\emptyset_k} = \\
& [-\tilde{C}_B^{\emptyset}(\tilde{C}_{B_k}^N)^T(\times \Delta \boldsymbol{r}_{INS,k}^N)]\delta \boldsymbol{\phi}_k^N + [C_B^{\emptyset}(\tilde{C}_{B_k}^N)^T](\delta \boldsymbol{r}_k^N - \delta \boldsymbol{r}_{k-1}^N) \\
& + [C_B^{\emptyset}(\tilde{C}_{B_k}^N)^T(\times \Delta \boldsymbol{r}_{INS,k}^N) + (\times \tilde{C}_B^{\emptyset}(I - \tilde{C}_{B_{k-1}}^{B_k})\boldsymbol{m}_{la}^B)]\delta \boldsymbol{m}_{bs,k} \\
& + \tilde{C}_B^{\emptyset}(I - \tilde{C}_{B_{k-1}}^{B_k})\delta \boldsymbol{m}_{la}^B - diag(\Delta \boldsymbol{r}_{VO,k}^{\emptyset_k})\delta \boldsymbol{m}_{sf,k} - \delta \boldsymbol{m}_{b,k} - \boldsymbol{v}_{ACC,k}
\end{aligned}
\tag{4.48}
$$

where $\boldsymbol{v}_{ACC,k}$ is the measurement noise after the accumulation process. In this model, the delta position error is made correlated with the attitude change measurement error due to coordinate transformation process. In the case of using both the delta distance and attitude measurements, the Kalman filter shall be modified for the correlated measurement noise.

Model II:

In this case, the accumulated value of the incremental delta-position vector is obtained as,

$$
\Delta \boldsymbol{r}_{VO,k} = \sum_{m=1}^{N} (\Delta \boldsymbol{r}_{VO,k-1[m]}^{\emptyset_{k-1[m-1]}} + \boldsymbol{v}_{k-1[m]}) \tag{4.49}
$$

The corresponding delta position measurement at the camera center can be calculated in terms of the INS computed navigation quantities by assuming sufficiently small minor interval as,

$$
\Delta \boldsymbol{r}_{INS,k} = \tilde{C}_B^{\emptyset} \int_{t_{k-1}}^{t_k} \left((C_{B(t)}^N)^T \tilde{\boldsymbol{v}}_{INS}^N + \boldsymbol{w}_{EB}^B \times \tilde{\boldsymbol{m}}_{la}^B \right) dt \tag{4.50}
$$

The measurement error vector, $\delta \boldsymbol{y}_k = \Delta \boldsymbol{r}_{VO,k} - \Delta \boldsymbol{r}_{INS,k}$, is constructed as,

$$
\tilde{C}_B^{\emptyset} \int_{t_{k-1}}^{t_k} \left((C_{B(t)}^N)^T [\times \tilde{\boldsymbol{v}}_{INS}^N] \delta \boldsymbol{\phi}^N(t) - (C_{B(t)}^N)^T \delta \boldsymbol{v}^N(t) \right) dt - \tilde{C}_B^{\emptyset} diag(\Delta \boldsymbol{r}_{INS,k}) \delta \boldsymbol{m}_{bs,k} - \boldsymbol{v}_{ACC,k} \tag{4.51}
$$

4.3.5.2 Attitude Change Measurement Model

The attitude change measurement over the Kalman cycle can be obtained by the available incremental attitude change measurements as,

$$
[C_{C_{k-1}}^{C_k}]_{VO} = \prod_{m=1}^{N} [C_{C_{k-1[m]}}^{C_{k-1[m-1]}}]_{VO} \tag{4.52}
$$

Then visual odometry attitude change error measurement can be constructed as follows,

$$
[C_{\emptyset_{k-1}}^{\emptyset_k}]_{VO}[C_{\emptyset_{k-1}}^{\emptyset_k}]_{INS}^T \approx I_{3x3} + \Delta \Psi_{mea}^{\emptyset_k} + \Delta \Psi_{INS}^{\emptyset_k} \tag{4.53}
$$

where $[C^{\emptyset_k}_{\emptyset_{k-1}}]_{INS} = \tilde{C}^{\emptyset}_{B}(\tilde{C}^{N}_{B_k})^T \tilde{C}^{N}_{B_{k-1}}(\tilde{C}^{\emptyset}_{B})^T$ is the INS calculated accumulated attitude change measurement. Here, $\Delta\Psi^{\emptyset_k}_{mea} = [\times\Delta\psi^{\emptyset_k}_{mea}]$ is the VO attitude measurement error matrix. $\Delta\Psi^{C_k}_{INS} = [\times\Delta\psi^{\emptyset_k}_{INS}]$ where,

$$\Delta\Psi^{\emptyset_k}_{INS} = \tilde{C}^{\emptyset}_{B}(\tilde{C}^{N}_{B_k})^T(\delta\boldsymbol{\phi}^N_k - \delta\boldsymbol{\phi}^N_{k-1}) \tag{4.54}$$

4.3.5.3 Delayed-State Kalman Filtering

Note that the derived Model I delta-position and attitude change measurement models include (one sample) delayed elements of the error state vector which is not suitable for the direct implementation of the Kalman filter. Similarly, Model II delta position measurement includes time integral of the error state vector in the measurement interval. In [63] and [64], the delayed-state measurement case is handled by stochastic cloning method where a duplicate of the pose error states (i.e. the attitude and position error states) are used as the traditional states of the Kalman filter. The adaptation of the delayed-state measurements to Kalman filter is a known problem in estimation theory. This type of measurement model is handled by the widely used delayed-state measurement Kalman filter [62]. In this method, the measurement model is represented in the following form,

$$\delta\boldsymbol{y}_k = \Delta\boldsymbol{r}^{\emptyset_k}_{VO,k} - \Delta\boldsymbol{r}^{\emptyset_k}_{INS,k} = H_k\delta\boldsymbol{x}_k + J_k\delta\boldsymbol{x}_{k-1} + \boldsymbol{\nu}_k \tag{4.55}$$

where H_k and J_k are the measurement matrices that relates the measurement errors to the current and delayed states, respectively. By using the linear error dynamics, the delayed error state is expressed in terms of the current error state as follows,

$$\delta\boldsymbol{x}_{k-1} = \Phi^{-1}_{k,k-1}\delta\boldsymbol{x}_k - \Phi^{-1}_{k,k-1}\boldsymbol{w}_k \tag{4.56}$$

Then, the measurement model takes the following form,

$$\delta\boldsymbol{y}_k = (H_k + J_k\Phi^{-1}_{k,k-1})\delta\boldsymbol{x}_k + (-J_k\Phi^{-1}_{k,k-1}\boldsymbol{w}_k + \boldsymbol{\nu}_k) \tag{4.57}$$

Note that the measurement model includes the measurement noise that is correlated with the process noise. Using the Kalman filter equations that are derived for the correlated measurement and process noise, optimal state estimation equations can be derived for the delayed-state measurement [62]. We propose to utilize a more practical method, as suggested in [65] and [66] for INS-GPS integration, using the linear system error dynamics by ignoring the effect of the correlation between the process and measurement noise within the time interval. They used this technique for processing the time-differenced carrier phase measurements of GPS in INS-GPS Kalman filter. In this regard, the delta-position measurement vector for Model I can be re-expressed as,

$$\begin{aligned}
\delta\boldsymbol{y}_k = {} & \\
& [-\tilde{C}^{\emptyset}_{B}(\tilde{C}^{N}_{B_k})^T(\times\Delta\boldsymbol{r}^N_{INS,k})]\delta\boldsymbol{\phi}^N_k + C^{\emptyset}_{B}(\tilde{C}^{N}_{B_k})^T\int_{t_{k-1}}^{t_k}\delta\boldsymbol{v}^N(t)dt \\
& + [\tilde{C}^{\emptyset}_{B}(\tilde{C}^{N}_{B_k})^T(\times\Delta\boldsymbol{r}^N_{INS,k}) + (\times\tilde{C}^{\emptyset}_{B}(I - \tilde{C}^{B_k}_{B_{k-1}})\boldsymbol{m}^B_{la})]\delta\boldsymbol{m}_{bs,k} \\
& + \tilde{C}^{\emptyset}_{B}(I - \tilde{C}^{B_k}_{B_{k-1}})\delta\boldsymbol{m}^B_{la} - diag(\boldsymbol{r}^{\emptyset_{k-1}}_{VO,k})\delta\boldsymbol{m}_{bs,k} - \delta\boldsymbol{m}_{b,k} - \boldsymbol{\nu}_{ACC,k}
\end{aligned} \tag{4.58}$$

Velocity error vector at a given time t in the time interval $[t_{k-1}, t_k]$ can be expressed in terms of the final error state vector as,

$$\delta \boldsymbol{v}^N(t) = H(t)\delta \boldsymbol{x}(t) \approx H(t)\Phi(t, t_{k-1})\Phi^{-1}_{k,k-1}\delta \boldsymbol{x}(t_k) \tag{4.59}$$

where $\Phi(t, t_{k-1})$ is the state transition matrix that maps the state vector at time $k-1$ to the time t. Then the error state vector at the final time can be extracted in the time interval by utilizing the state-transition matrix as defined in [66],

$$\int_{t_{k-1}}^{t_k} \delta \boldsymbol{v}^N(t))dt = \left(\int_{t_{k-1}}^{t_k} H(t)\Phi(t, t_{k-1})dt \right) \Phi^{-1}_{k,k-1}\delta \boldsymbol{x}(t_k) \tag{4.60}$$

where the integral term $\int_{t_{k-1}}^{t_k} H(t)\Phi(t, t_{k-1})dt$ can be calculated during the measurement time interval with the increments Δt recursively as defined in [66],

$$\int_{t_{k-1}}^{t_{k-1}+i\Delta t} H(t)\Phi(t, t_{k-1})dt \cong H(t_{k-1} + i\Delta t)\Phi(t_{k-1} + i\Delta t, t_{k-1})\Delta t + \int_{t_{k-1}}^{t_{k-1}+(i-1)\Delta t} H(t)\Phi(t, t_{k-1})dt \tag{4.61}$$

The same approach can also be applied for Model II delta position and attitude measurements as well.

4.3.6 Kalman Measurement Update

The along-track and cross-track velocities from visual odometry and zero down velocity are used as measurement sources. These three velocities are defined in the vehicle body axis. The inertial velocities (at IMU rate) at the vehicle body axis are accumulated between the updates and the result is compared against the measurements in terms of delta distance. The visual odometry output is compensated with the scale factor and lever arm estimates. The measurement matrix, H, is given as follows:

$$\mathrm{H}_{k,3\times19} = \left[\overline{-\mathrm{C}^{\emptyset}_B \mathrm{C}^B_N \lfloor \boldsymbol{v}^N + 0.5\boldsymbol{a}^N \rfloor_\times} \quad \overline{\mathrm{C}^{\emptyset}_B \mathrm{C}^B_N} \quad 0_{3\times9} \quad \overline{m_{sf,k}\Delta r^{\emptyset_k}_{VO,k}} \quad \overline{\lfloor \boldsymbol{w}^B_{IB} \rfloor_\times} \right] \tag{4.62}$$

where $\overline{-\mathrm{C}^{\emptyset}_B \mathrm{C}^B_N [\boldsymbol{v}^N + 0.5\boldsymbol{a}^N]_\times}$, $\overline{\mathrm{C}^{\emptyset}_B \mathrm{C}^B_N}$ and $\overline{\lfloor \boldsymbol{w}^B_{IB} \rfloor_\times}$ are outputs of navigation process and averaged over the Kalman cycle. $\overline{m_{sf,k}\Delta r^{\emptyset_k}_{VO,k}}$ is the scaled visual odometry output in the forward vehicle direction. The measurement update equations are as follows:

$$\begin{aligned}
\mathrm{I}_k &= \mathrm{H}_k\, \mathrm{P}^-_k\, \mathrm{H}^T_k \;+\; \mathrm{R}_k && (4.63)\\
\boldsymbol{z}_k &= [\overline{VO_{at}} \quad \overline{VO_{ct}} \quad 0]^T - \left(\mathrm{C}^{\emptyset}_B \mathrm{C}^B_N \boldsymbol{v}^N - (\mathrm{C}^{\emptyset}_B \boldsymbol{w}^B_{IB} \times \boldsymbol{m}^B_{la}) \right) && (4.64)\\
\mathrm{K}_k &= \mathrm{P}^-_k\, \mathrm{H}^T_k\, \mathrm{I}^{-1}_k && (4.65)\\
\mathrm{P}_k &= (\mathrm{I}_{19\times19} \;-\; \mathrm{K}_k\, \mathrm{H}_k)\, \mathrm{P}^-_k\, (\mathrm{I}_{19\times19} \;-\; \mathrm{K}_k\, \mathrm{H}_k)^T \;+ \mathrm{K}_k\, \mathrm{R}_k\, \mathrm{K}^T_k && (4.66)\\
\hat{\boldsymbol{x}}_k &= \hat{\boldsymbol{x}}_{k-1} + \mathrm{K}_k(\boldsymbol{z}_k - \mathrm{H}_k \hat{\boldsymbol{x}}^-_k) && (4.67)
\end{aligned}$$

where $\overline{VO_{at}}$ and $\overline{VO_{ct}}$ are averaged (over the Kalman cycle) and scaled (with scale factor estimate) visual odometry along-track/cross-track velocities respectively. $\mathrm{C}^{\emptyset}_B$ defines the IMU to camera rotation matrix and is assumed to be known in advance. The camera frame is assumed to be the vehicle frame in this formulation. $\boldsymbol{m}_{la}^{B}$ defines the translation vector between the IMU and camera origins and is estimated in the Kalman filter.

A 3σ test is proposed in the measurement update stage to increase the robustness of the filter. If the measurement innovation is greater than the estimated 3σ value, then the covariance and state updates are not performed. The 3σ test helps to protect the filter from unexpected erroneous measurements.

4.3.7 Kalman Error State Feedback

The estimated error state vector, $\hat{\boldsymbol{x}}_k$, is used to correct the navigation outputs. The feedback formalization is as follows:

$$
\begin{aligned}
\mathrm{C}_b^n &= (\mathrm{I}_{3\times3} - [\hat{\boldsymbol{x}}_k(1:3)]_\times)\ \mathrm{C}_b^n && (4.68)\\
\boldsymbol{v}^n &= \boldsymbol{v}^n + \hat{\boldsymbol{x}}_k(4:6) && (4.69)\\
\boldsymbol{r}^n &= \boldsymbol{r}^n + \hat{\boldsymbol{x}}_k(7:9) && (4.70)\\
\boldsymbol{b}_a &= \boldsymbol{b}_a + \hat{\boldsymbol{x}}_k(10:12) && (4.71)\\
\boldsymbol{b}_g &= \boldsymbol{b}_g + \hat{\boldsymbol{x}}_k(13:15) && (4.72)\\
m_{sf} &= m_{sf} + \hat{\boldsymbol{x}}_k(16) && (4.73)\\
\boldsymbol{m}_{la}^B &= \boldsymbol{m}_{la}^B + \hat{\boldsymbol{x}}_k(17:19) && (4.74)
\end{aligned}
$$

4.3.8 Experimental Results

Only the 2D translational part of this motion estimate, along-track (forward) velocity and cross-track (leftward) velocity, are used in this integration. Instead of the down velocity estimate of camera, zero down velocity is provided as an aiding to the filter. The down velocity estimate of the camera is noisy and for a land vehicle the down velocity is close to zero at body axes (a wheeled land vehicle is not supposed to fall or jump while in motion). Therefore, it is a logical selection to use zero velocity aiding, instead of the visual odometry aiding in the body down channel. A similar approach can be utilized for the cross-track channel, and it is common in wheel odometry integration, since wheel odometry provides motion estimate only in along-track. A wheeled land vehicle is not supposed to have a cross-track velocity component unless it is turning or there is side slip. However, turns are common, especially in urban environments. If zero cross-track velocity aiding is used, the filter is exposed to erroneous measurements during turns. A general approach is to pause this update, when the vehicle is turning (the turn rate is an output of the navigation process). In visual odometry case, since it is estimated separately, cross-track velocity can be integrated to the filter directly.

The visual odometry-ins integration experiments are performed on KITTI Vision Benchmark Suite [1]. There exists stereo image sequence, raw IMU data, ground truth INS-GPS data, camera-to-IMU rotation and translation parameters in the dataset. the data is captured on a residential area with a consumer car, which is equipped with an IMU and stereo camera. The stereo camera directs to the front of the car. The duration of the sequence is 520 seconds. The image sequence, IMU and ground truth data is time stamped for proper synchronization. The work of Geiger et al. [42], [39] is followed to get both stereo and monocular visual odometry results.

The sequence is captured at 10 Hz and the IMU data is at 100 Hz and they are asynchronous to each other. The navigation calculations run at 100 Hz and the Kalman Filter runs at 1 Hz synchronous with the IMU data. 1 Hz Kalman cycle is counted from the IMU cycle (Kalman Filter is processed at every 100th IMU cycle). At the Kalman cycle, last 10 visual odometry results are accumulated in order to achieve the Gaussian assumption as discussed in Kalman Time Update section. The synchronization of VO data with IMU data is achieved using the time stamps.

The IMU uses three solid-state MEMS angular rate sensors and three servo (force-feedback) accelerometers. The gyroscope and accelerometer bias specifications are denoted as $36\ deg/hr\ 1\sigma$ and $10\ mm/s^2 (1\ milig)\ 1\sigma$, respectively, in the IMU datasheet.

The images are grayscale and image size is 1226x370 pixels.

The 2D plot of the explored trajectory is plotted in Figure 4.6. The ground truth and hybrid Stereo-VO/IMU trajectories are plotted onto the aerial image by using the KML path formatting feature in Google Earth. The ground truth, Stereo VO, IMU only and Hybrid Stereo-VO/IMU 2D position results are displayed in Figure 4.7. The IMU-only position solution drifts and finally looses track. However, the VO aiding compensates for the drift and the hybrid solution traces the true track reliably. The result of same experiment with monocular VO is given in Figure 4.8. The monocular VO is quite erroneous compared to the stereo VO. However, the hybrid Mono-VO/IMU result is not very different from the hybrid Stereo-VO/IMU solution. 3σ test that is discussed in Kalman Measurement Update Section handles erroneous data in visual odometry. Moreover, since the Kalman filter makes use of the delta measurements, rejecting the visual odometry aiding at these time intervals does not effect the overall performance. This result shows that the proposed integration scheme is able to cope with high VO error. The 2D errors of individual algorithms against the ground-truth is depicted in Figure 4.9. Both stereo and monocular VO aiding is able to suppress the drift in inertial navigation. The translational error against travelled distance is given in Figure 4.10. The stereo VO integration is better than the monocular VO integration, since stereo VO produces more plausible results and the scale is inherently solved in the stereo algorithm. It should be noted that the percentage error decreases as the path length increases. It can be concluded that the effect of VO/IMU integration becomes apparent as the path length increases. In Figure 4.11, Stereo-VO, Monocular-VO, Hybrid Stereo-VO/IMU and Hybrid Mono-VO/IMU speed results are compared against the ground-truth INS-GPS speed. Note that there is a jump in ground truth velocities. This jump should be due to loss of GPS satellites in the INS-GPS

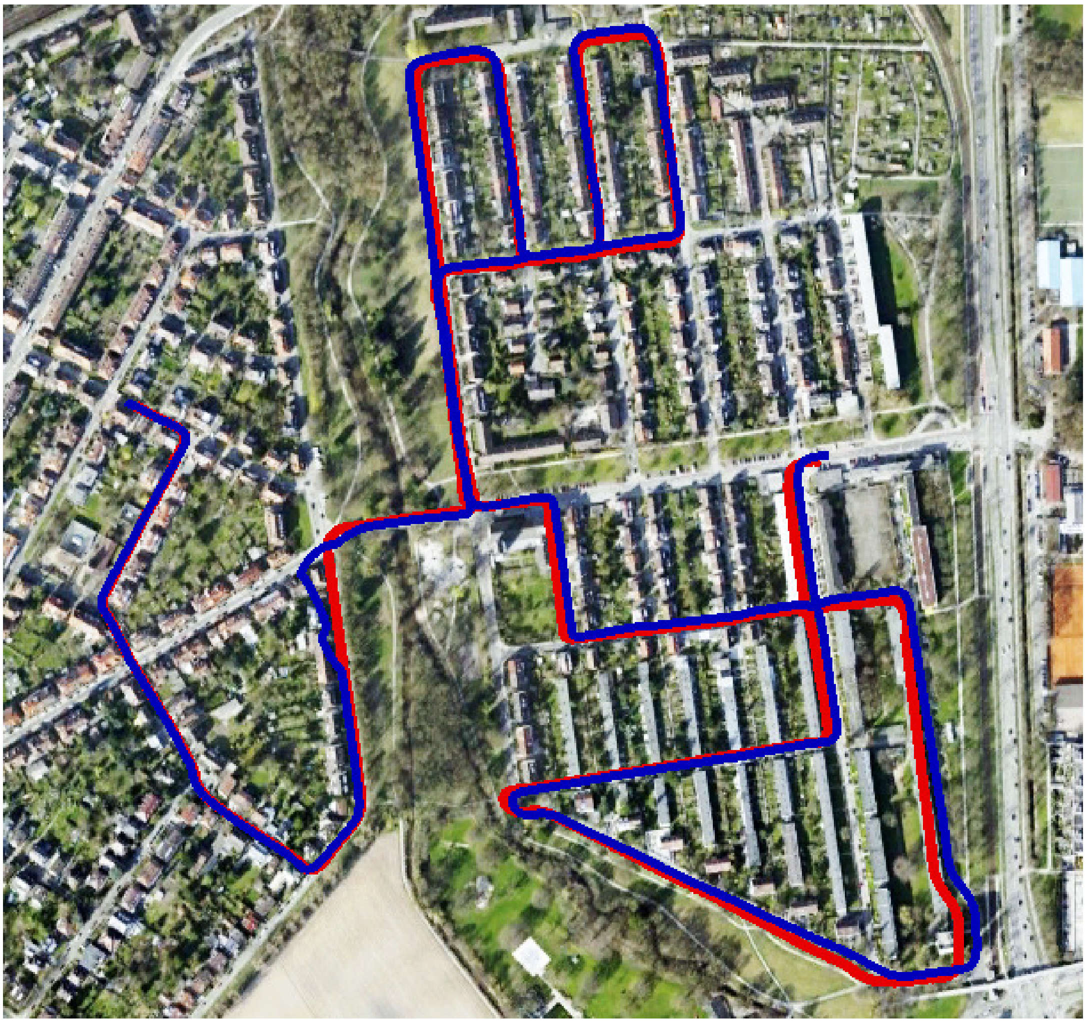
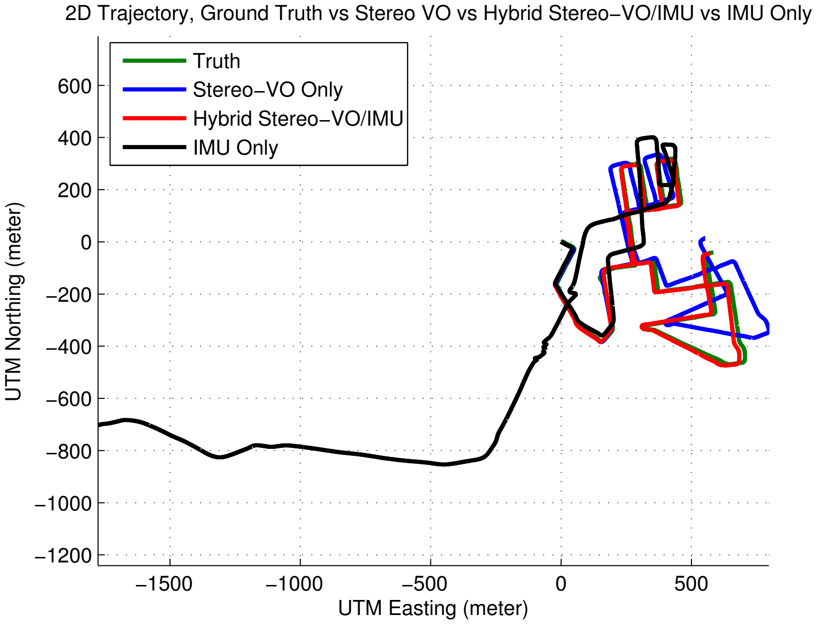
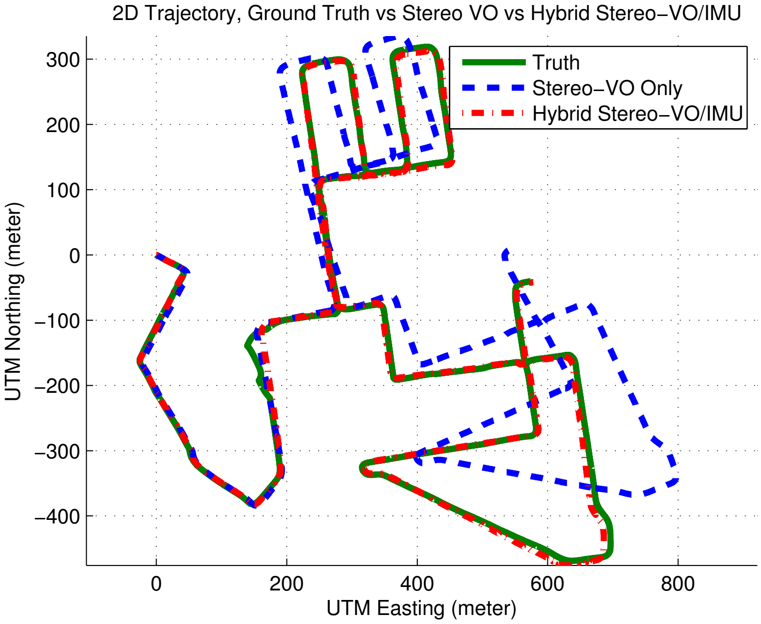


Figure 4.6: 2D Trajectory plotted on Google Earth data - Ground Truth (blue) vs Hybrid Stereo-VO/IMU (red).



(a)



(b)

Figure 4.7: (a) 2D Position Comparison between ground truth INS-GPS, Stereo VO, IMU Only and Hybrid Stereo-VO/IMU data, (b) IMU Only Data is removed for better scope

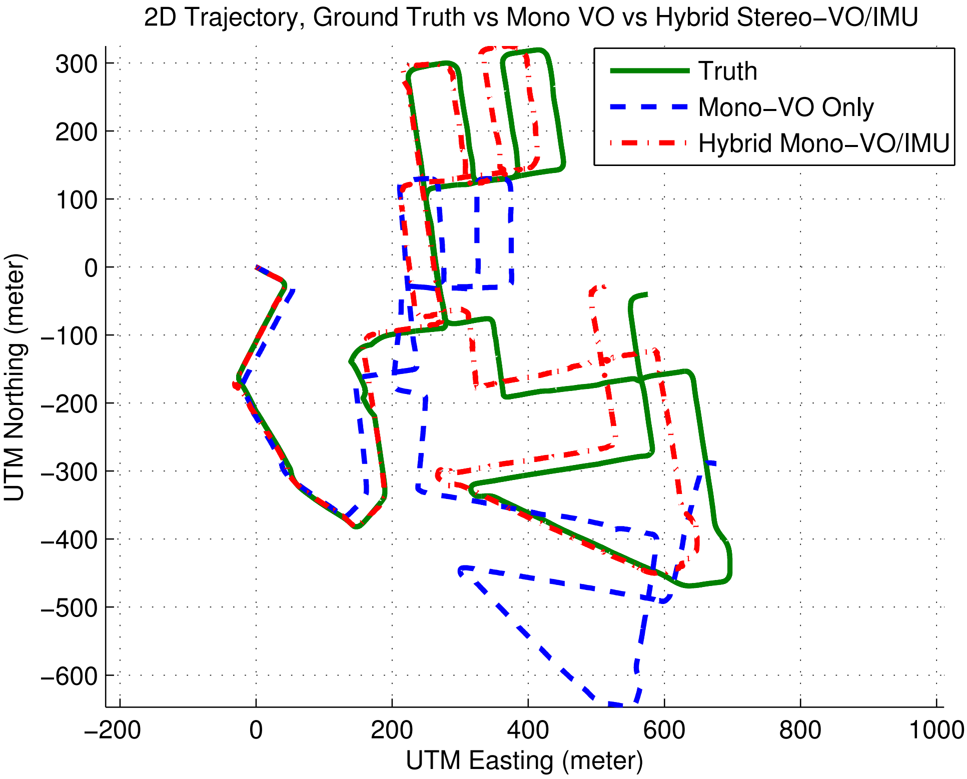


Figure 4.8: 2D Position Comparison between ground truth INS-GPS, Monocular VO and Hybrid Mono-VO/IMU data.

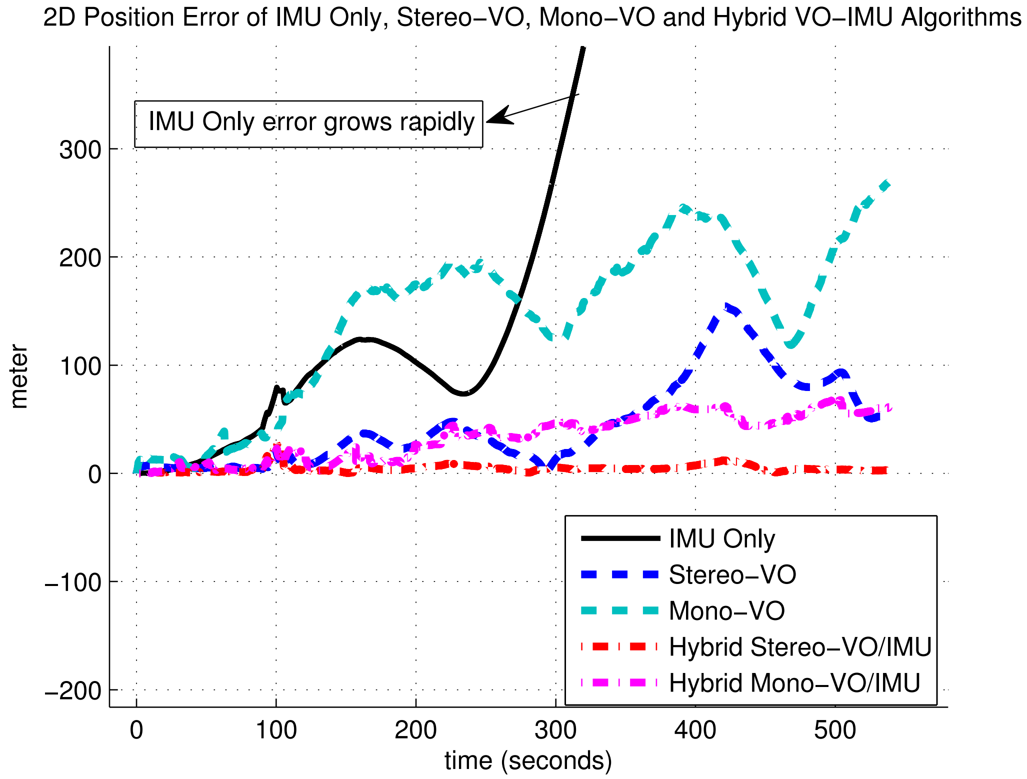


Figure 4.9: 2D Position Comparison between ground truth INS-GPS, Monocular VO and Hybrid Mono-VO/IMU data.

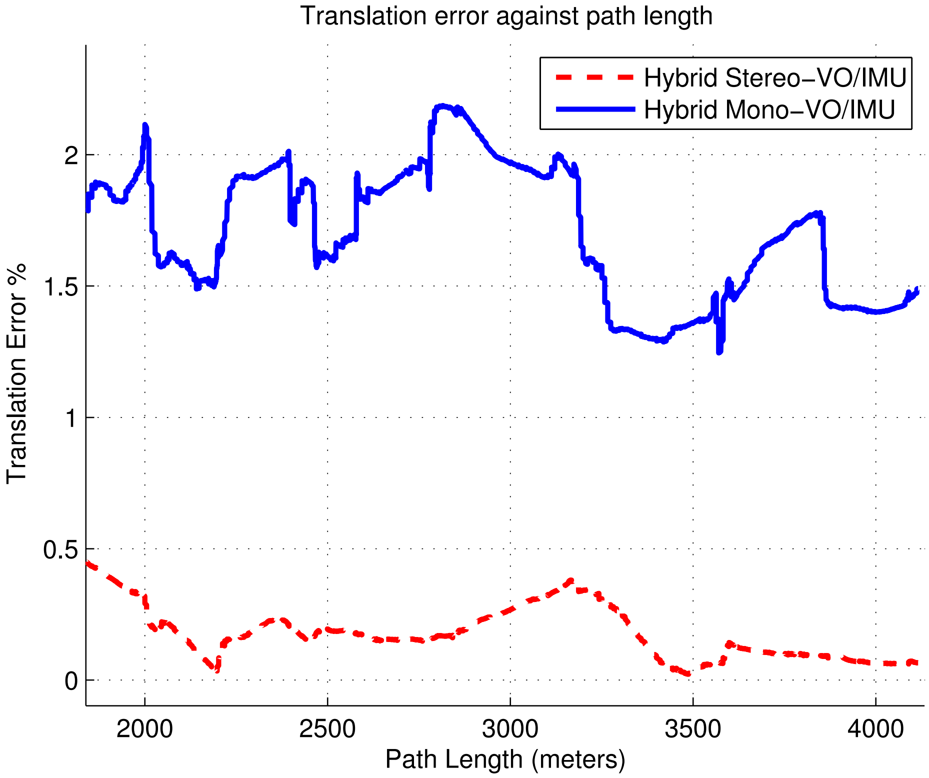
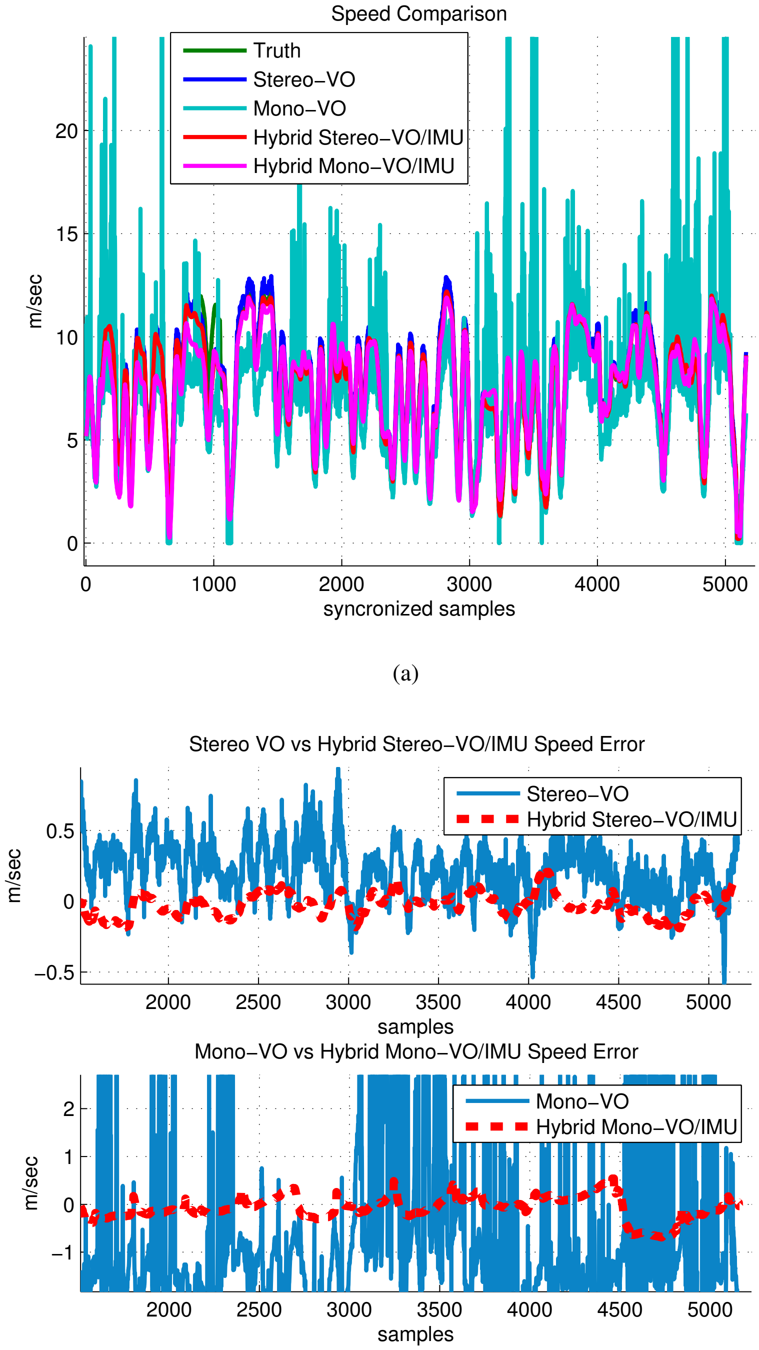


Figure 4.10: The translational error against distance travelled.

data. As expected, Hybrid VO/IMU and IMU only data is not affected from this jump. This coincidence supports the proposition that GPS might not be feasible in cluttered urban environments and visual aiding is a feasible alternative to GPS. It can be observed that VO-IMU solution continues smoothly in this cluttered area. This erroneous area can also be observed in Figure 4.6 as an irregular deviation of ground truth position from the track. The erroneous characteristics of monocular VO is evident in Figure 4.11(b). The VO/IMU integration algorithm is able to suppresses and smoothes this error. The attitude error is given in Figure 4.12 in terms of Euler angles. Note that the roll and pitch angle errors are bounded, but the yaw error grows in time. This is due to the fact that the roll and pitch angles are observable; however, the yaw angle is not observable in the Kalman filter. The state covariances are depicted in the figure to illustrate this phenomena. The success of the filter comes from estimating the unknown parameters of the sensors as well as the state errors. These are gyroscope and accelerometer biases and visual odometry scale factor. In Figure 4.13 and 4.14, Kalman Filter estimates for gyroscope and accelerometer biases for all three axes are depicted, respectively. It can be observed that the bias estimates vary more in the monocular case. This is due to the noisy monocular VO data. The filter tends to charges this error to sensor bias estimates. It might be beneficial to relax the measurement noise setting in the filter to make the bias estimates more smoother. However, this attempt might reduce the usage of VO data. This is a common trade-off in Kalman Filter design, and the filter parameters should be tuned based on the characteristics of the dynamical system and measurements. The bias estimates are still coherent with the specified values in IMU datasheet. The visual odometer scale factor estimates for stereo and monocular experiments are also given in Figure 4.15. The scale factor estimate is more active for the monocular case, since the stereo VO has a smoother scale factor characteristic. It might be beneficial to tune the Kalman Filter parameters (process and measurement noise) in terms of these observations.



(a)

(b)

Figure 4.11: (a) 2D Speed Comparison between ground truth INS-GPS, Stereo-VO, Monocular-VO, Hybrid Stereo-VO/IMU and Hybrid Mono-VO/IMU algorithms., (b) 2D Speed Errors for Hybrid Stereo-VO/IMU Mono-VO/IMU algorithms.

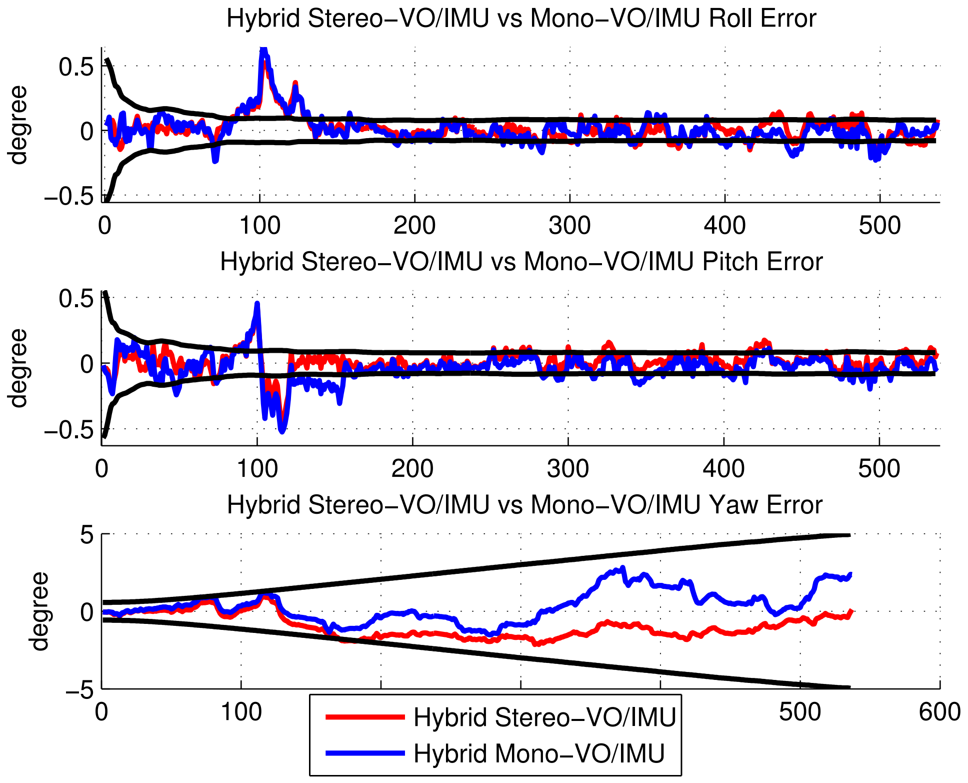


Figure 4.12: Attitude Error in terms of Euler angles (Roll, Pitch, Yaw)

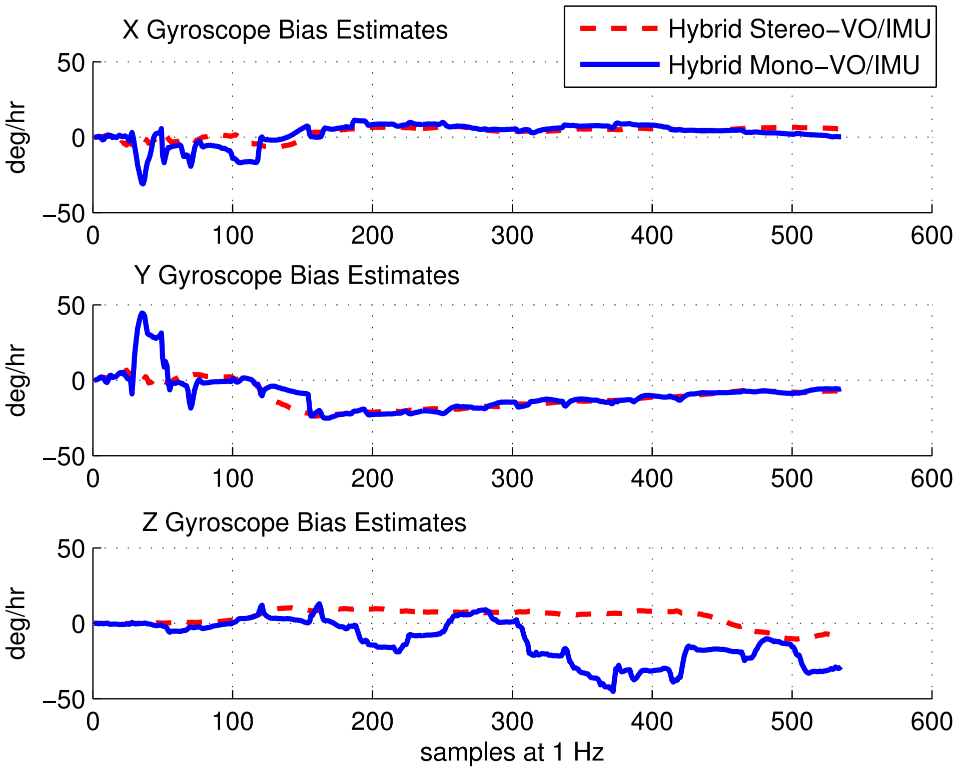


Figure 4.13: Kalman Filter Gyroscope Bias Estimates

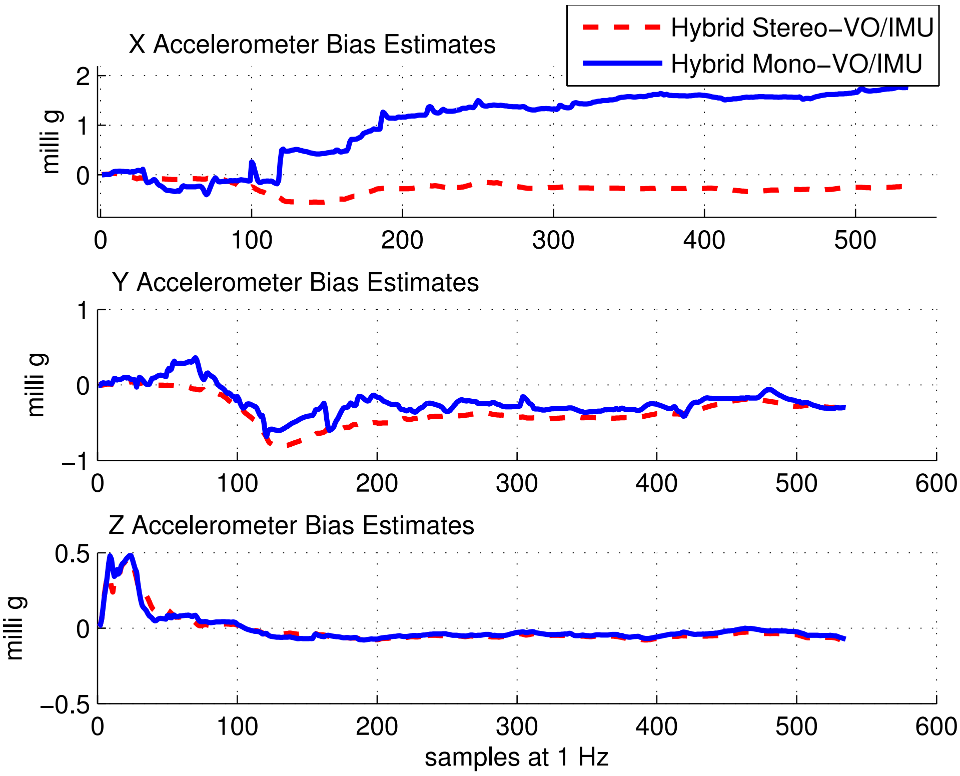


Figure 4.14: Kalman Filter Accelerometer Bias Estimates

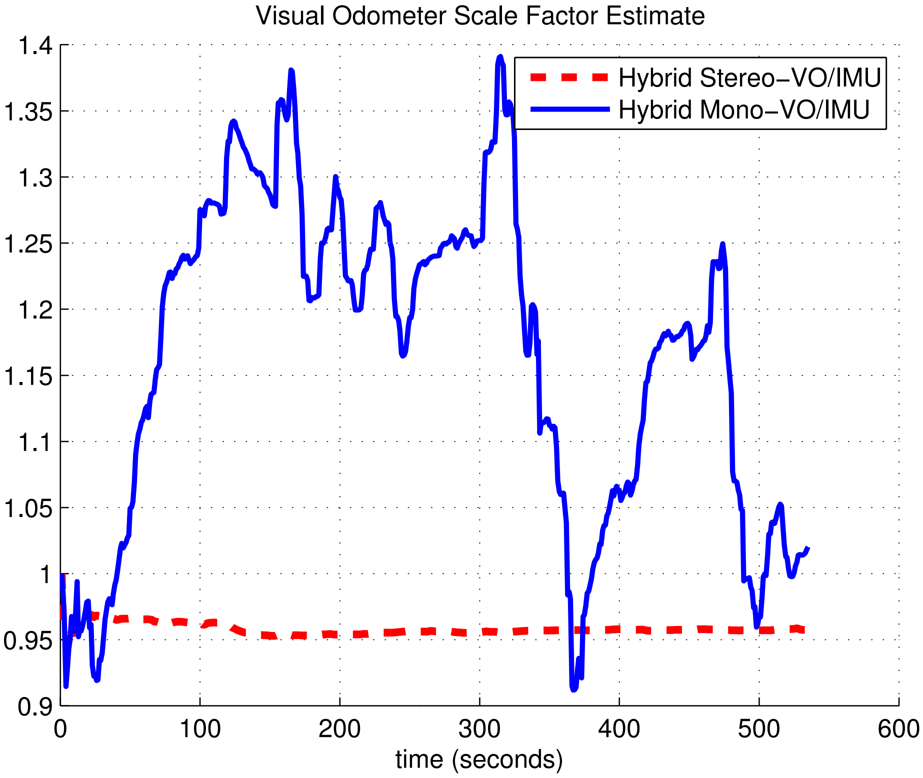


Figure 4.15: Scale factor estimate for stereo and monocular VO in the integration filter.

4.4 Tightly Coupled Visual Inertial Integration

In the loosely coupled visual/inertial sensor fusion, the output of the inertial calculations and visual odometry processes were physically comparable, i.e. position change and attitude change. Therefore it was fairly straight-forward to construct a measurement model in terms of the state vector of the system (equation 4.11). The main drawback of the loosely coupled integration is its dependence on the quality of vision-only motion estimate. If the visual odometry output is implausible at a certain time instant, there is almost no way but to discard the measurement from the filter. However, there might still be valuable implicit information about camera motion, even if the output is invaluable. In order to avoid the dependence of the filter on the quality of visual-only motion estimate, the general approach is to embed the vision based motion estimation stage into the Kalman Filter together with inertial sensor measurements. This type of integration may be labeled as tight coupling, since inertial and visual measurement fusion is achieved at lower levels with raw data (i.e. acceleration, angular velocity, 2D feature locations, epipoles, homography, etc.). The inertial measurements may be used to support the visual algorithms as well. This type of integration, where the visual-inertial aiding is bilateral, may be labeled as deeply coupled or ultra-tightly coupled with analogy to the established GPS-INS integration literature.

There are various methods in the literature that exploit the early stages of visual algorithms as an aiding source for inertial pose estimation. These methods are discussed in the following paragraphs.

4.4.1 Feature Location Based Methods

The feature location based methods rely on the fact that there exist geometric constraints that arise when a static feature is observed from multiple camera poses. These geometric constraints can be expressed in 3D space as well as 2D projective space. It is preferred to express this geometric relation in 2D projective space, since the feature depth ambiguity in 3D space might impact the performance of the algorithm, as Nister pointed out in their visual motion estimation framework [41]. The 2D projections of the features are more reliable than their 3D estimates, and in general the measurement models are based on 2D feature location estimate errors. In a minority of the approaches, the 3D feature locations are assumed to be known in advance. It is possible to obtain absolute position aiding from the known feature locations, which can not be obtained from any type of vision based navigation algorithm; however, a feature identification stage is required in this case, which makes the algorithm cumbersome. Moreover, it is required to have pre-loaded maps for the algorithm to perform. In general, the environment is assumed to be unexplored and 3D feature locations are estimated using stereo or multi-view reconstruction techniques, as they become available and tracked afterwards over frames.

Feature extraction methods typically detect and track hundreds of features in images. These features may or may not be appended in the state vector of the Kalman filter. In SLAM-based estimation methods, the current IMU pose and the 3D position of all features are jointly estimated (e.g., [67], [68]). Therefore the feature locations are appended to the state vector in the filter. The fundamental advantage of SLAM based algorithms is based on the fact that they account for the correlations that exist between the pose of the camera and the 3D positions of the observed features. On the other hand, the main limitation of SLAM is its high computational complexity.

It is also possible to build a measurement model without including the 3D feature locations in the filter state vector, resulting in computational complexity only linear in the number of features [69]. Several algorithms exist, contrary to SLAM, that estimate the pose of the camera (i.e. do not jointly estimate or store the feature positions), with the aim of achieving real-time operation (e.g., [56], [53]).

4.4.2 Epipolar Geometry Based Methods

Constraints between the current and previous images can be defined by using the epipolar geometry as defined in Section 2.3. Indeed, the epipolar geometry is defined only by camera intrinsic and extrinsic (pose) parameters and does not depend on the scene, yet can be estimated from the feature observations. This bilateral characteristic of epipolar geometry makes it a convenient modality for integration with inertial sensors. Epipolar geometry based methods use feature matches between frames as well, with some difference on geometric constraints. In this case, there is no need to estimate the 3D locations of the features explicitly and the features are not augmented to the state vector (e.g., [58], [70]).

4.4.3 Homography Based Methods

A homography is an invertible mapping of points and lines on the projective plane P^2. Hartley and Zisserman [33] provide the specific definition that a homography is an invertible mapping from P^2 to itself such that three points lie on the same line, if and only if their mapped points are also collinear. They also provide the following theorem:

A mapping from $P^2 \rightarrow P^2$ is a projectivity if and only if there exists a non-singular 3x3 matrix $\boldsymbol{H}$ such that for any point in P^2 represented by vector $\boldsymbol{x}$ it is true that its mapped point equals $\boldsymbol{Hx}$.

Homography encapsulates the relative motion between two cameras and can be estimated

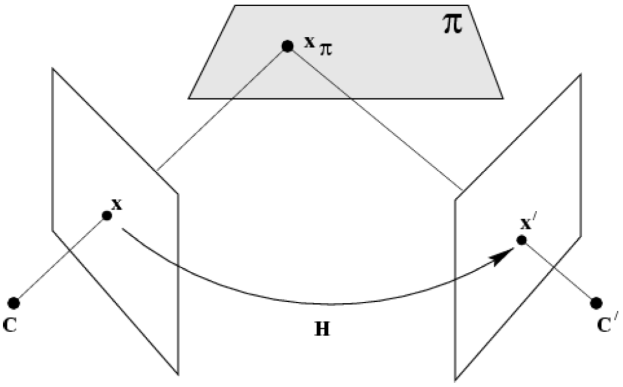


Figure 4.16: Homography induced by the plane $\boldsymbol{\pi}$.

from visual feature matches based on the assumption that they lie on the same plane. The homography induced by a plane is depicted in Figure 4.16. The homography relation can be decomposed as,

$$\begin{aligned} \mathbf{H}_\pi &= \boldsymbol{R} + \tfrac{1}{d}\boldsymbol{t}\,\boldsymbol{n}^T \\ \boldsymbol{x}' &= \mathbf{H}_\pi \boldsymbol{x} \end{aligned} \tag{4.75}$$

where $\boldsymbol{R}$ is the relative rotation between the cameras defined in navigation frame, $\boldsymbol{t}$ is the relative translation between cameras defined in camera frame, d and $\boldsymbol{n}$ are the distance and normal of the plane with respect to the first camera. The normal of the plane contains the roll and pitch angles of the camera.

The homography matrix can be decomposed to get translational and rotational components of the relative motion. There are homography-based and visual-only motion estimation algorithms. If the motion estimate is carried out explicitly, it will be a loosely coupled integration, when fused with the inertial measurements. However, the difference between the homography matrix that is calculated from images and inertial sensors independently might be used as the measurement itself in the Kalman filter. In [71], the homography matrix is reshaped as a 9x1 vector and related to the IMU state vector.

Although there are algorithms which try to utilize multiple homographies for motion estimation [72], the vast majority of the algorithms utilize a single dominant plane (e.g., [71], [73]). Therefore, homography-based vision-aided inertial navigation is generally used in airborne applications, where a dominant ground plane is present most of the time. If the altitude of the air vehicle is sufficiently large, the ground can be assumed to be planar regardless of its structure. Otherwise, planar ground surface requirement is relaxed to piecewise planar patches.

In this section, a feature location based visual-inertial sensor fusion is proposed based on a tightly coupled indirect feedback Kalman Filter. The visual measurements are utilized in terms of 2D visual landmark (feature) location estimates for this purpose.

4.4.4 Proposed Feature Based Visual-Inertial Integration

The total system error model, filter initialization, navigation update, and Kalman time update are similar to the Loosely Coupled integration scheme (Section 4.3); therefore, it will not be repeated in this section. In tightly coupled integration, the main difference is the measurement model and its impact on the filter update rates and state transitions. As a reminder, it should be noted that the error state vector of the Kalman filter is constructed as follows:

$$\delta \boldsymbol{x}^T = [\delta \boldsymbol{\phi}^N \ \delta \boldsymbol{v}^N \ \delta \boldsymbol{r}^N \ \delta \boldsymbol{f}^B \ \delta \boldsymbol{\omega}^B] \tag{4.76}$$

The boresight/leverarm related states are omitted for simplicity, since they have the same formulation with loosely coupled integration. Here $\delta \boldsymbol{\phi}^N$ is the perturbed attitude tilt error, $\delta \boldsymbol{v}^N$ is the perturbed velocity error and $\delta \boldsymbol{r}^N$ is the perturbed position error. $\delta \boldsymbol{f}^B$, $\delta \boldsymbol{w}^B$ are the accelerometer and gyro error vectors, respectively. Each of these vectors might include the systematical errors of the inertial sensor such as bias, scale factor, misalignment errors.

As opposed to SLAM-based approaches, the visual feature locations are not appended to the state vector. The proposed formulation is compatible to the Multi-State Constraint (MSCKF) implementation of Mourikis and Roumeliotis [69], where the last m camera views are appended to the state vector in order to use the additional constraints between multiple poses when a feature is observed in multiple images. In their approach, every time a new image is recorded, a copy of the current camera pose estimate is appended to the state together with its covariance. by directly expressing the geometric constraints between multiple camera poses; hence, the loss of information associated with pairwise displacement estimation is avoided. The augmented state vector becomes,

$$\delta \boldsymbol{x}_{\boldsymbol{k}}^T = [\delta \boldsymbol{x}_{IMU_k} \mid \delta \boldsymbol{\phi}^N_{C_1} \ \delta \boldsymbol{r}^N_{\boldsymbol{C_1}} \ \cdots \ \delta \boldsymbol{\phi}^N_{C_m} \ \delta \boldsymbol{r}^N_{\boldsymbol{C_m}}] \tag{4.77}$$

4.4.4.1 Measurement Model

The proposed measurement model that relates the error state vector to visual measurement errors is the main contribution of the tightly coupled scheme. Since an indirect Kalman filter is used for state estimation, it is sufficient to define a residual that depends linearly on the state errors according to the general form:

$$\delta z_k = z_k - \hat{z}_k \ = \ \mathrm{H} \, \delta \boldsymbol{x}_k + noise \tag{4.78}$$

Here z_k is the visual measurement of the feature location in image plane (from feature matching/tracking algorithm), $\hat{z}_k$ is the estimated feature location in image plane (from projection of 3D feature locations to image plane using inertial pose estimations), H is the measurement Jacobian matrix that relates the state errors to measurement residual, $\boldsymbol{x}_k$ is the error state vector, and the noise term is zero-mean, white and uncorrelated to the state error.

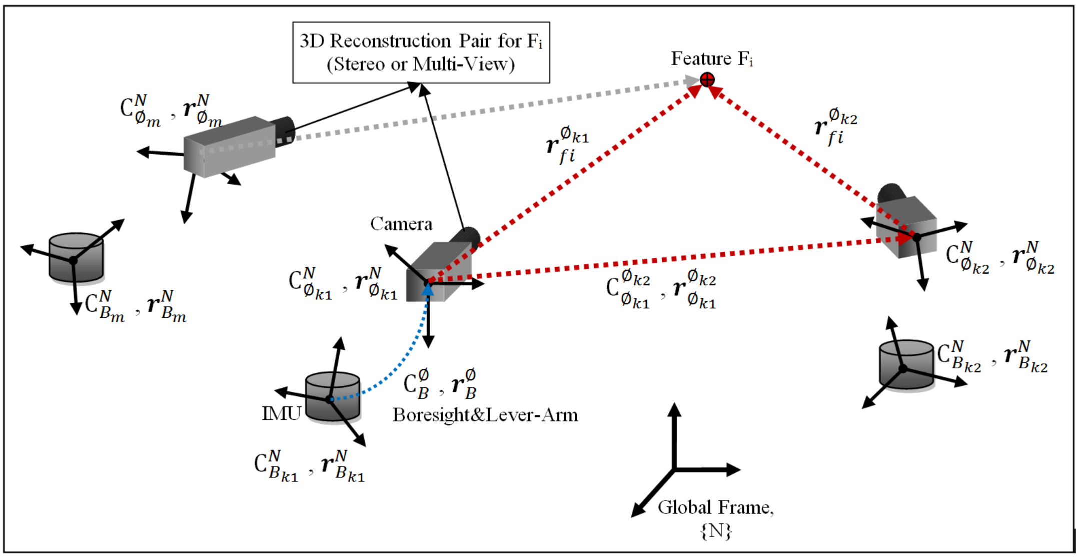

3D Reconstruction Pair for F$_i$
(Stereo or Multi-View)
Feature F$_i$
$C^N_{\emptyset_m}$, $r^N_{\emptyset_m}$
$r^{\emptyset_{k1}}_{fi}$
$r^{\emptyset_{k2}}_{fi}$
Camera
$C^N_{\emptyset_{k1}}$, $r^N_{\emptyset_{k1}}$
$C^{\emptyset_{k2}}_{\emptyset_{k1}}$, $r^{\emptyset_{k2}}_{\emptyset_{k1}}$
$C^N_{\emptyset_{k2}}$, $r^N_{\emptyset_{k2}}$
$C^N_{B_m}$, $r^N_{B_m}$
$C^N_{B_{k2}}$, $r^N_{B_{k2}}$
$C^{\emptyset}_{B}$, $r^{\emptyset}_{B}$
IMU
Boresight&Lever-Arm
$C^N_{B_{k1}}$, $r^N_{B_{k1}}$
Global Frame,
{N}

Figure 4.17: Pictorial depiction of the tightly coupled visual-inertial filter measurement model.

The main motivation behind the measurement model is the fact that viewing a static feature from multiple camera poses results in constraints involving these poses. The measurement model is derived for a single feature, f_i, observed from two camera poses and will be generalized from there. This scenario is depicted in Figure 4.17. The 3D location of the feature can be estimated from a stereo setup or multi-view reconstruction algorithms in the first camera frame. This 3D location can be expressed in the global reference frame with simple matrix multiplications, as well. This frame representations will only change the measurement formulation.

$$\boldsymbol{r}^{\emptyset_{k2}}_{f_i} = \left[\boldsymbol{X}^{\emptyset_{k2}}_{f_i} \quad \boldsymbol{Y}^{\emptyset_{k2}}_{f_i} \quad \boldsymbol{Z}^{\emptyset_{k2}}_{f_i}\right]^T = C^{\emptyset_{k2}}_{\emptyset_{k1}}\left(\boldsymbol{r}^{\emptyset_{k1}}_{f_i} - \boldsymbol{r}^{\emptyset_{k1}}_{\emptyset_{k2}}\right) \tag{4.79}$$

In Equation 4.79, $\boldsymbol{r}^{\emptyset_{k1}}_{f_i}$ and $\boldsymbol{r}^{\emptyset_{k2}}_{f_i}$ are 3D feature locations in the first and second camera frames, respectively, $C^{\emptyset_{k2}}_{\emptyset_{k1}}$ is the rotation and $\boldsymbol{r}^{\emptyset_{k1}}_{\emptyset_{k2}}$ is the translation from camera frame $\emptyset_{k1}$ to $\emptyset_{k2}$. Note that the transformation { $C^{\emptyset_{k2}}_{\emptyset_{k1}}$, $\boldsymbol{r}^{\emptyset_{k1}}_{\emptyset_{k2}}$ } is calculated from the visual-inertial sensor fusion output; therefore, it bears information about the errors of the total IMU states (position, velocity, attitude). The feature location at $\emptyset_{k2}$ is a function of feature location at $\emptyset_{k1}$ and total IMU states at $k1$ and $k2$ by the following formulation:

$$\begin{aligned}
\boldsymbol{r}^{\emptyset_{k2}}_{f_i} &= C^{\emptyset_{k2}}_{\emptyset_{k1}}\left(\boldsymbol{r}^{\emptyset_{k1}}_{f_i} - \boldsymbol{r}^{\emptyset_{k1}}_{\emptyset_{k2}}\right) \\
C^{\emptyset_{k2}}_{\emptyset_{k1}} &= (C^N_{\emptyset_{k2}})^T\, C^N_{\emptyset_{k1}} \\
\boldsymbol{r}^{\emptyset_{k1}}_{\emptyset_{k2}} &= (C^N_{\emptyset_{k1}})^T\left(\boldsymbol{r}^N_{k2} - \boldsymbol{r}^N_{k1}\right) \\
\boldsymbol{r}^{\emptyset_{k2}}_{f_i} &= \ell(\boldsymbol{x}_{k2}, \boldsymbol{x}_{k1}, \boldsymbol{r}^{\emptyset_{k1}}_{f_i})
\end{aligned} \tag{4.80}$$

The feature location at camera $\{\emptyset_{k1}, \boldsymbol{r}_{f_i}^{\emptyset_{k1}}\}$, can be estimated by stereo or multi-view reconstruction algorithms using feature matches/tracks. In the case of stereo reconstruction, the error in camera pose estimates does not have any effect on the feature location estimate. However, in the case of multi-view reconstruction, the error in feature location estimate is correlated with the camera pose estimate errors. This correlation should be handled properly in the filter. In [69], the effect of this correlation is eliminated by modifying the final measurement matrix equation.

The projection of 3D feature $\boldsymbol{r}_{f_i}^{\emptyset_{k2}}$ onto the image plane of camera $\emptyset_{k2}$ is defined with the following formula,

$$z_{k2} = \begin{bmatrix} u_i \\ v_i \end{bmatrix}^{\emptyset_{k2}} = \wp\left(\boldsymbol{r}_{f_i}^{\emptyset_{k2}}\right) = \wp\left(\begin{bmatrix} \boldsymbol{X}_{f_i}^{\emptyset_{k2}} & \boldsymbol{Y}_{f_i}^{\emptyset_{k2}} & \boldsymbol{Z}_{f_i}^{\emptyset_{k2}} \end{bmatrix}^T\right) = \begin{bmatrix} f_x \dfrac{\boldsymbol{X}_{f_i}^{\emptyset_{k2}}}{\boldsymbol{Z}_{f_i}^{\emptyset_{k2}}} + u_0 \\ \\ f_y \dfrac{\boldsymbol{Y}_{f_i}^{\emptyset_{k2}}}{\boldsymbol{Z}_{f_i}^{\emptyset_{k2}}} + v_0 \end{bmatrix} \tag{4.81}$$

Here f_x, f_y, u_0, v_0 are the camera calibration parameters such that the camera calibration matrix is as follows:

$$K = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.82}$$

Using the above formulation, we can relate the current total IMU states in the Kalman filter to the visual feature location measurement in 2D image plane as follows:

$$z_{k2} = \hbar(\boldsymbol{r}_{f_i}^{\emptyset_{k2}}) = \hbar(\boldsymbol{x}_{k2}, \boldsymbol{x}_{k1}, \boldsymbol{r}_{f_i}^{\emptyset_{k1}}) = (\wp \circ \ell)(\boldsymbol{x}_{k2}, \boldsymbol{x}_{k1}, \boldsymbol{r}_{f_i}^{\emptyset_{k1}}) \tag{4.83}$$

ℓ and $\wp$ functions are defined in equations 4.80 and 4.81, respectively. Function ℓ relates the current feature location to previous feature location and IMU states, function $\wp$ performs the 3D-2D projection operation. The current and previous IMU states appear in the non-linear measurement equation; this seems to violate the standard Kalman filter structure, where the measurement should be related only to the current state. It will be shown that in the error state formulation, the difference between current and previous IMU error states can be formulated in the current IMU error states (e.g., the difference in position error is formulated as velocity error of the current state).

In order to derive a linear relation between the current error state of the Kalman filter and measurement error, we need the Jacobians of the non-linear equations ℓ and $\wp$. The Jacobian of $\wp$ is common to almost every visual-inertial integration algorithm, and is given as follows:

$$\boldsymbol{J}_{\wp,k2}^{i} = \frac{1}{\hat{\boldsymbol{Z}}_{f_i}^{\emptyset_{k2}}} \begin{bmatrix} f_x & 0 & -f_x \dfrac{\hat{\boldsymbol{X}}_{f_i}^{\emptyset_{k2}}}{\hat{\boldsymbol{Z}}_{f_i}^{\emptyset_{k2}}} \\ \\ 0 & f_y & -f_y \dfrac{\hat{\boldsymbol{Y}}_{f_i}^{\emptyset_{k2}}}{\hat{\boldsymbol{Z}}_{f_i}^{\emptyset_{k2}}} \end{bmatrix}$$

In order to find a linear relation between the IMU error states and visual measurement error in function ℓ, a perturbation based method is followed. Remembering the measurement residual formulation, $\delta z_k = z_k - \hat{z}_k$, the goal is to represent $\hat{z}_k$ in terms of a "true + error" form. The general perturbation formulas for attitude, position are as follows:

$$\begin{aligned}
\mathrm{C}^N_{\emptyset_{k1}} &= \left(\mathrm{I} - \Psi^N_{\emptyset_{k1}}\right) \cdot \tilde{\mathrm{C}}^N_{\emptyset_{k1}} \\
\mathrm{C}^N_{\emptyset_{k2}} &= \left(\mathrm{I} - \Psi^N_{\emptyset_{k2}}\right) \cdot \tilde{\mathrm{C}}^N_{\emptyset_{k2}} \\
\boldsymbol{r}^N_{\emptyset_{k1}} &= \tilde{\boldsymbol{r}}^N_{\emptyset_{k1}} + \delta \boldsymbol{r}^N_{\emptyset_{k1}} \\
\boldsymbol{r}^N_{\emptyset_{k2}} &= \tilde{\boldsymbol{r}}^N_{\emptyset_{k2}} + \delta \boldsymbol{r}^N_{\emptyset_{k2}} \\
\boldsymbol{r}^{\emptyset_{k1}}_{f_i} &= \tilde{\boldsymbol{r}}^{\emptyset_{k1}}_{f_i} + \delta \boldsymbol{r}^{\emptyset_{k1}}_{f_i}
\end{aligned} \tag{4.84}$$

In the relation above, the variables with tilde $\tilde{()}$ on top are erroneous approximates of the true values, whereas Ψ and δ terms are the perturbations. Specifically, for the ℓ function (Equation 4.80), the perturbation formula can be written as,

$$\begin{aligned}
\boldsymbol{r}^{\emptyset_{k2}}_{f_i} &= \ell(\boldsymbol{x}_{k2}, \boldsymbol{x}_{k1}, \boldsymbol{r}^{\emptyset_{k1}}_{f_i}) \\
&= \mathrm{C}^{\emptyset_{k2}}_{\emptyset_{k1}} \left(\boldsymbol{r}^{\emptyset_{k1}}_{f_i} - \boldsymbol{r}^{\emptyset_{k1}}_{\emptyset_{k2}}\right) \\
&= (\mathrm{C}^N_{\emptyset_{k2}})^T \, \mathrm{C}^N_{\emptyset_{k1}} \left(\boldsymbol{r}^{\emptyset_{k1}}_{f_i} - (\mathrm{C}^N_{\emptyset_{k1}})^T \left(\boldsymbol{r}^N_{k2} - \boldsymbol{r}^N_{k1}\right)\right)
\end{aligned} \tag{4.85}$$

$$\begin{aligned}
\delta \boldsymbol{r}^{\emptyset_{k2}}_{f_i} &= \boldsymbol{r}^{\emptyset_{k2}}_{f_i} - \tilde{\boldsymbol{r}}^{\emptyset_{k2}}_{f_i} \\
&= \left[(\mathrm{C}^N_{\emptyset_{k2}})^T \, \mathrm{C}^N_{\emptyset_{k1}} \left(\boldsymbol{r}^{\emptyset_{k1}}_{f_i} - (C^N_{\emptyset_{k1}})^T \left(\boldsymbol{r}^N_{k2} - \boldsymbol{r}^N_{k1}\right)\right)\right] \\
&- \left[(\tilde{\mathrm{C}}^N_{\emptyset_{k2}})^T \, \tilde{\mathrm{C}}^N_{\emptyset_{k1}} \left(\tilde{\boldsymbol{r}}^{\emptyset_{k1}}_{f_i} - (\tilde{\mathrm{C}}^N_{\emptyset_{k1}})^T \left(\tilde{\boldsymbol{r}}^N_{k2} - \tilde{\boldsymbol{r}}^N_{k1}\right)\right)\right]
\end{aligned} \tag{4.86}$$

Re-writing the first term in Equation 4.86 according to the perturbation form of Equation 4.84,

$$\begin{aligned}
\boldsymbol{r}^{\emptyset_{k2}}_{f_i} = & \overbrace{\left(\left(\mathrm{I} - \Psi^N_{\emptyset_{k2}}\right)\tilde{\mathrm{C}}^N_{\emptyset_{k2}}\right)^T \cdot \left(\left(\mathrm{I} - \Psi^N_{\emptyset_{k1}}\right)\tilde{\mathrm{C}}^N_{\emptyset_{k1}}\right)}^{\mathrm{A}} \cdot \\
& \underbrace{\left[\left(\tilde{\boldsymbol{r}}^{\emptyset_{k1}}_{f_i} + \delta \boldsymbol{r}^{\emptyset_{k1}}_{f_i}\right) - \left(\left(\mathrm{I} - \Psi^N_{\emptyset_{k1}}\right)\tilde{\mathrm{C}}^N_{\emptyset_{k1}}\right)^T \cdot \left(\tilde{\boldsymbol{r}}^N_{\emptyset_{k2}} + \delta \boldsymbol{r}^N_{\emptyset_{k2}} - \tilde{\boldsymbol{r}}^N_{\emptyset_{k1}} - \delta \boldsymbol{r}^N_{\emptyset_{k1}}\right)\right]}_{\mathrm{B}}
\end{aligned} \tag{4.87}$$

$$\mathrm{A} = \overbrace{(\tilde{\mathrm{C}}^N_{\emptyset_{k2}})^T \, \tilde{\mathrm{C}}^N_{\emptyset_{k1}}}^{\tilde{A}} + \overbrace{(\tilde{\mathrm{C}}^N_{\emptyset_{k2}})^T \left(\Psi^N_{\emptyset_{k2}} - \Psi^N_{\emptyset_{k1}}\right)\tilde{\mathrm{C}}^N_{\emptyset_{k1}}}^{\delta A} \tag{4.88}$$

$$\begin{aligned}
\mathrm{B} = & \overbrace{\tilde{\boldsymbol{r}}^{\emptyset_{k1}}_{f_i} - (\tilde{\mathrm{C}}^N_{\emptyset_{k1}})^T \left(\tilde{\boldsymbol{r}}^N_{\emptyset_{k2}} - \tilde{\boldsymbol{r}}^N_{\emptyset_{k1}}\right)}^{\tilde{B}} + \\
& \overbrace{\delta \boldsymbol{r}^{\emptyset_{k1}}_{f_i} - (\tilde{\mathrm{C}}^N_{\emptyset_{k1}})^T \left(\delta \tilde{\boldsymbol{r}}^N_{\emptyset_{k2}} - \delta \tilde{\boldsymbol{r}}^N_{\emptyset_{k1}}\right) - (\tilde{\mathrm{C}}^N_{\emptyset_{k1}})^T \Psi^N_{\emptyset_{k1}} \left(\tilde{\boldsymbol{r}}^N_{\emptyset_{k2}} - \tilde{\boldsymbol{r}}^N_{\emptyset_{k1}}\right)}^{\delta B}
\end{aligned} \tag{4.89}$$

$$\begin{aligned}
\boldsymbol{r}^{\emptyset_{k2}}_{f_i} &= \mathrm{A} \cdot \mathrm{B} \\
&= (\tilde{\mathrm{A}} + \delta \mathrm{A}) \cdot (\tilde{\mathrm{B}} + \delta \mathrm{A}) \\
&= \tilde{\mathrm{A}} \cdot \tilde{\mathrm{B}} + \tilde{\mathrm{A}} \cdot \delta \mathrm{B} + \delta \mathrm{A} \cdot \tilde{\mathrm{B}} + \overbrace{\delta \mathrm{A} \cdot \delta \mathrm{B}}^{\text{Eliminate the } 2^{nd} \text{ order term}}
\end{aligned} \tag{4.90}$$

Remembering that $\tilde{\boldsymbol{r}}_{f_i}^{\emptyset_{k2}} = \tilde{\mathrm{A}}\,\tilde{\mathrm{B}}$, the perturbation $\delta\boldsymbol{r}_{f_i}^{\emptyset_{k2}}$ can be simplified as follows,

$$\begin{aligned}
\delta\boldsymbol{r}_{f_i}^{\emptyset_{k2}} &= \boldsymbol{r}_{f_i}^{\emptyset_{k2}} - \tilde{\boldsymbol{r}}_{f_i}^{\emptyset_{k2}} \\
&= \tilde{\mathrm{A}} \cdot \tilde{\mathrm{B}} + \tilde{\mathrm{A}} \cdot \delta\mathrm{B} + \delta\mathrm{A} \cdot \tilde{\mathrm{B}} - \tilde{\mathrm{A}} \cdot \tilde{\mathrm{B}} \\
&= \tilde{\mathrm{A}} \cdot \delta\mathrm{B} + \delta\mathrm{A} \cdot \tilde{\mathrm{B}} \\
&= (\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\,\tilde{\mathrm{C}}_{\emptyset_{k1}}^{N}\,\delta\boldsymbol{r}_{f_i}^{\emptyset_{k1}} \;-\; (\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\left(\delta\boldsymbol{r}_{\emptyset_{k2}}^{N} - \delta\boldsymbol{r}_{\emptyset_{k1}}^{N}\right) - \\
&(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\,\Psi_{\emptyset_{k1}}^{N}\,\left(\tilde{\boldsymbol{r}}_{\emptyset_{k2}}^{N} - \tilde{\boldsymbol{r}}_{\emptyset_{k1}}^{N}\right) \;+\; (\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\,\left(\Psi_{\emptyset_{k2}}^{N} - \Psi_{\emptyset_{k1}}^{N}\right)\,\mathrm{C}_{\emptyset_{k1}}^{N} \;- \\
&(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\,\left(\Psi_{\emptyset_{k2}}^{N} - \Psi_{\emptyset_{k1}}^{N}\right)\left(\tilde{\boldsymbol{r}}_{\emptyset_{k2}}^{N} - \tilde{\boldsymbol{r}}_{\emptyset_{k1}}^{N}\right)
\end{aligned} \tag{4.91}$$

Rearranging the terms, the relation between the perturbed measurement error and IMU error states becomes,

$$\begin{aligned}
\delta\boldsymbol{r}_{f_i}^{\emptyset_{k2}} = {} & (\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\,\tilde{\mathrm{C}}_{\emptyset_{k1}}^{N}\,\delta\boldsymbol{r}_{f_i}^{\emptyset_{k1}} - (\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\Delta t\,\overline{\delta\boldsymbol{v}_{\emptyset_{k2}}^{N}} \;+\; \left\lfloor(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\left(\tilde{\boldsymbol{r}}_{\emptyset_{k2}}^{N} - \tilde{\boldsymbol{r}}_{\emptyset_{k1}}^{N}\right)\right\rfloor_{\times}\,\overline{\delta\boldsymbol{\phi}_{k2}^{N}} + \\
& \left\lfloor(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\left(\tilde{\boldsymbol{r}}_{f_i}^{\emptyset_{k2}} - \tilde{\boldsymbol{r}}_{\emptyset_{k2}}^{N}\right)\right\rfloor_{\times}\,\overline{\delta\boldsymbol{\omega}_{k2}^{B}}
\end{aligned} \tag{4.92}$$

Using the above formulation, the Jacobian of function ℓ (Equation 4.80) becomes,

$$\boldsymbol{J}_{\ell,k2}^{i} = \left[\left\lfloor(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\left(\tilde{\boldsymbol{r}}_{\emptyset_{k2}}^{N} - \tilde{\boldsymbol{r}}_{\emptyset_{k1}}^{N}\right)\right\rfloor_{\times} \quad -(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\Delta t \quad 0_{3\times3} \quad 0_{3\times3} \quad \left\lfloor(\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\left(\tilde{\boldsymbol{r}}_{f_i}^{\emptyset_{k2}} - \tilde{\boldsymbol{r}}_{\emptyset_{k2}}^{N}\right)\right\rfloor_{\times}\right] \tag{4.93}$$

The Jacobian of the composite function, $(\wp \circ \ell)$ is the product of the Jacobians of the composed functions. Therefore, the final measurement Jacobian of the i^{th} feature at time $k2$ becomes the product Jacobians given in Equations 4.85 and 4.94, respectively.

$$\mathrm{H}_{x,k2}^{i} = \boldsymbol{J}_{\wp,k2}^{i} \cdot \boldsymbol{J}_{\ell,k2}^{i} \tag{4.94}$$

where,

$$\delta z_k \simeq \mathrm{H}_{x,k2}^{i}\,\delta\boldsymbol{x}_{\boldsymbol{k2}} + noise \tag{4.95}$$

In the above measurement formulation, the boresight and lever-arm between the IMU and the camera is omitted (i.e. the sensing element centers are assumed to be coincident). In practice, there is a certain amount of angular misalignment and a translational difference between the sensing centers of these sensors. If the angular misalignment is denoted as $\mathrm{C}_{\emptyset}^{B}$, and the translational difference is denoted as $\boldsymbol{r}_{\emptyset}^{B}$, the following formulation is used to calculate the compensated values,

$$\begin{aligned}
\mathrm{C}_{\emptyset}^{N} &= \mathrm{C}_{B}^{N} \cdot \mathrm{C}_{\emptyset}^{B} \\
\boldsymbol{r}_{\emptyset}^{N} &= \boldsymbol{r}_{B}^{N} + \mathrm{C}_{B}^{N}\boldsymbol{r}_{\emptyset}^{B}
\end{aligned} \tag{4.96}$$

If the previous IMU state estimates are used to compute the feature positions, the error in feature position estimates become apparent in the measurement formulation as follows:

$$\delta z_k \simeq \mathrm{H}_{x,k2}^{i}\,\delta\boldsymbol{x}_{\boldsymbol{k2}} + \mathrm{H}_{f,k2}\,\delta\boldsymbol{r}_{f_i}^{\emptyset_{k1}} + noise \tag{4.97}$$

$$\mathrm{H}_{f,k2} = (\tilde{\mathrm{C}}_{\emptyset_{k2}}^{N})^{T}\,\tilde{\mathrm{C}}_{\emptyset_{k1}}^{N} \tag{4.98}$$

In this formulation, error in $\boldsymbol{r}_{f_i}^{\emptyset_{k1}}$ will be correlated with the IMU error state. To overcome this problem, the authors in [69] suggested to use a modified residual equation where the original residual is projected on the left nullspace of $\mathrm{H}_{f,k2}$ as follows,

$$\delta z_{k2} \simeq \mathrm{V}^T \mathrm{H}_{x,k2}^{i} \, \delta \boldsymbol{x_{k2}} + noise \tag{4.99}$$

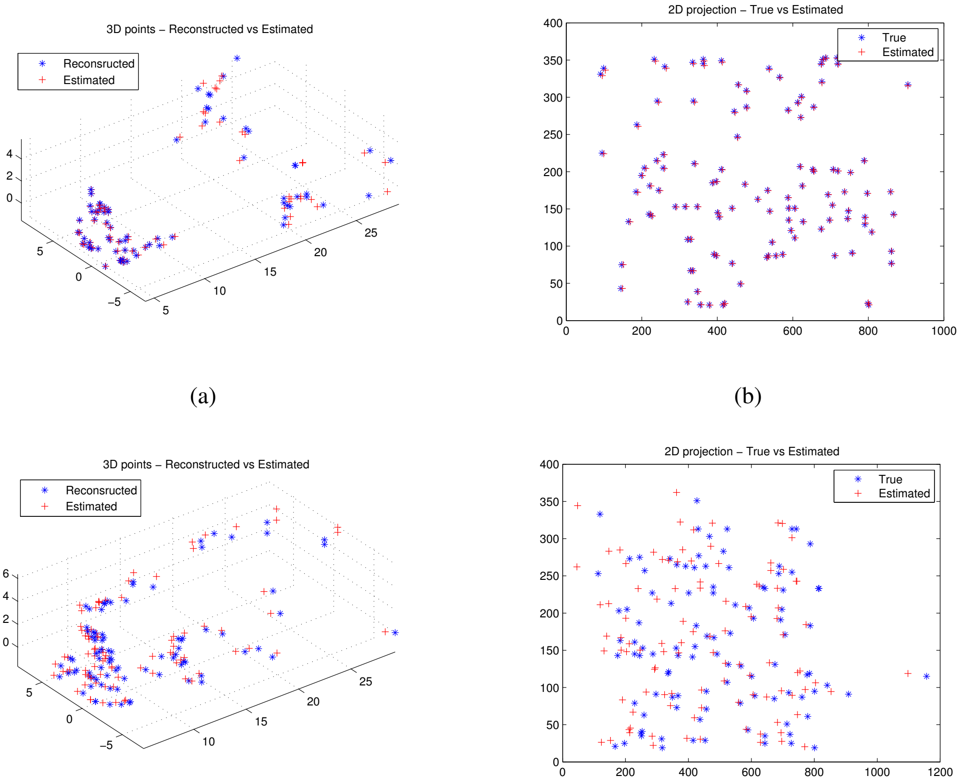
where V denotes the unitary matrix whose columns form the basis of the left nullspace of $\mathrm{H}_{f,k2}$.

4.4.4.2 Experimental Results

The tightly coupled visual-inertial fusion algorithm has been tested extensively with real data. In this section, some representative results are discussed. The experiments are performed on KITTI Vision Benchmark Suite [1], same as Section 4.3. The 3D reconstruction of visual features are performed using multi-view stereo between sequential frames, the 3D-2D projection is performed on the following frame. The Kalman filter runs at the image rate.

In Figure 4.18, a reconstruction-projection cycle is depicted for a free-inertial run in order to emphasize the effect of pose estimation performance on the re-projection error. It is evident that the re-projection error grows as the pose drifts in time. This fact is also observed in Figure 4.19 where the RMS re-projection error is depicted throughout the run. The drift in re-projection error is apparent for the free-inertial pose estimation case.

2D position results for the ground truth, loosely coupled mono-VO/IMU and tightly coupled visual inertial filter result for monocular vision are displayed in Figure 4.20. The number of feature updates per Kalman cycle is given in Figure 4.22. The mean of the number of features is approximately around 90. Li and Mourikis suggest that the more features yield better precision [74]. The number of features range from 25 to 400 in their work, by a 2D position error budget ranging from 25 to 100 meters. The max 2D error is 40 meters in our experiment as it can be observed from Figure 4.21.



(a) (b)

(c) (d)

Figure 4.18: (a)&(b) 3D Feature Reconstruction/2D Projection output for the first frame in the sequence , (c)&(d) 3D Feature Reconstruction/2D Projection output after 300 seconds of free-inertial run.

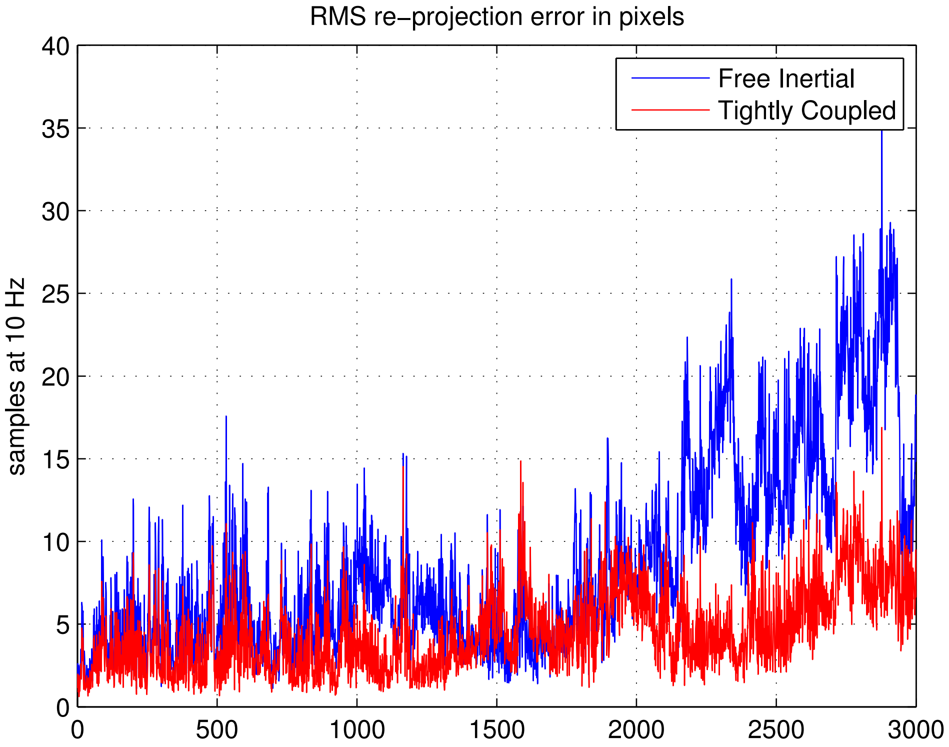


Figure 4.19: RMS re-projection error in pixels, tightly coupled vs free-inertial pose estimation

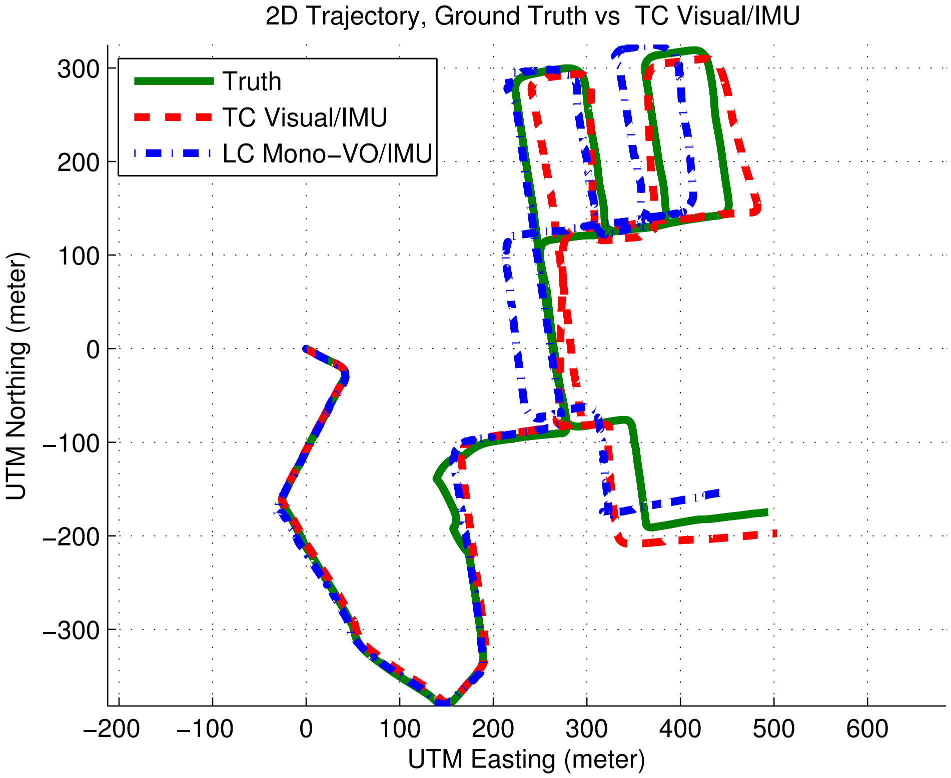


Figure 4.20: 2D Position Comparison between ground truth INS-GPS, LC Mono-VO/IMU and TC Visual/IMU data

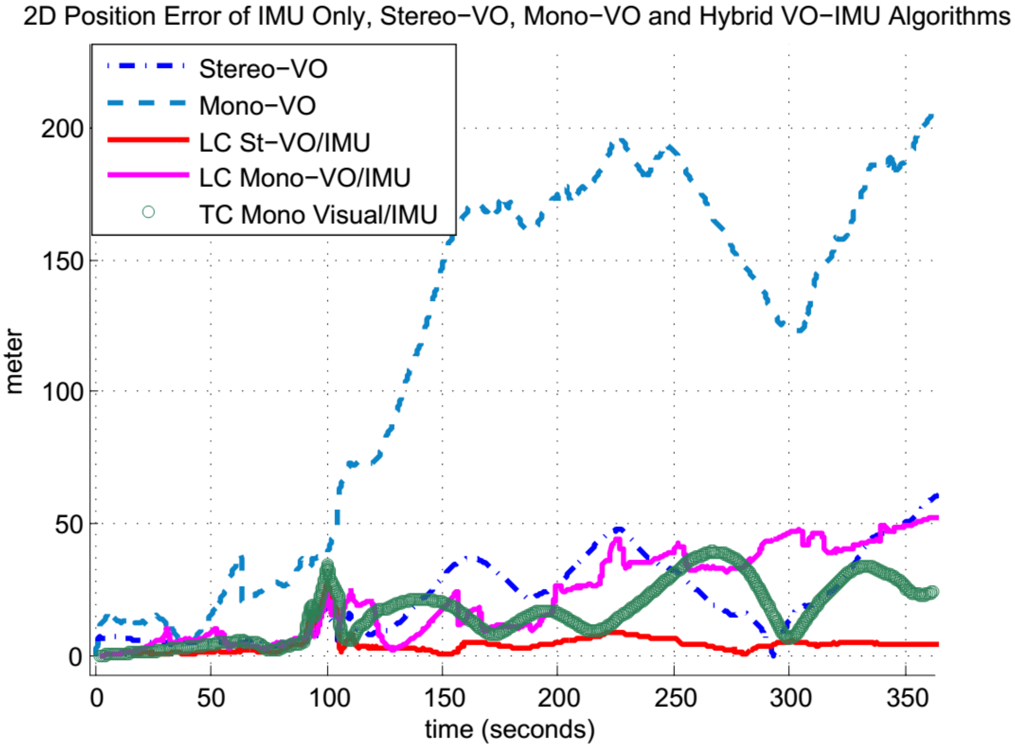


Figure 4.21: 2D Position Error Comparison between all proposed methods and visual only pose estimates

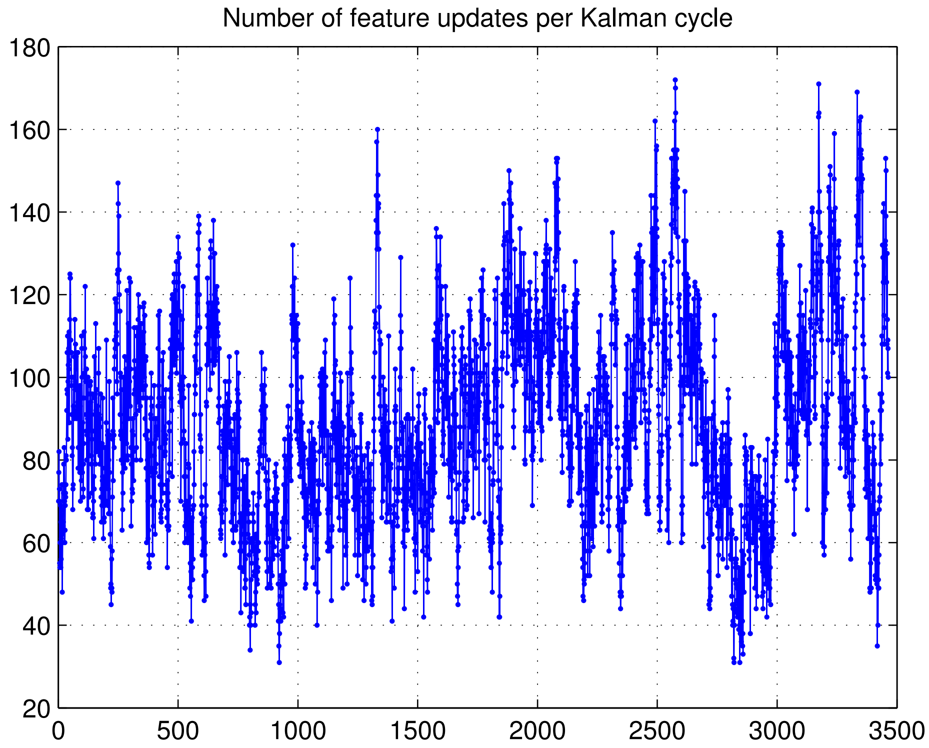


Figure 4.22: Number of feature updates per Kalman cycle, average number is 90

4.5 Conclusion

In this chapter, two solution approaches are proposed for camera pose estimation by combining the inertial sensor data by the visual measurements in an Extended Kalman Filter framework.

In the first approach, a loosely coupled indirect feedback Kalman Filter is presented for integration of visual odometry and inertial navigation system which exploits the complementary characteristics of the visual and inertial sensors. The non-Gaussian, non-stationary and correlated error characteristics of the visual odometry motion estimate is explored and practical methods are proposed to tackle the problems induced by these error characteristics. Visual odometry pre-filtering and measurement sigma-test in the Kalman filter are introduced for this purpose. Two measurement models are derived for the accumulated and incremental visual odometry measurements. A practical measurement model approach is proposed for the delta position and attitude change measurements that inherently includes delayed-state. 6-DoF visual odometry motion estimates (rotational and translational) are used as the measurement.

Experimental results are presented for a real dataset collected with a land vehicle in an urban environment. The proposed integration algorithm is tested on both with stereo and monocular visual odometry systems. The results show that it is possible to get admissible navigation performance even with a low performance inertial sensor by integration of visual odometry to inertial navigation. The best state-of-the-art stereo visual odometry algorithms report an error of %2 OR. The OR value is about %10 for monocular visual odometry algorithms. Our inertial integration scheme is able to reduce the OR error to %0.2 for stereo and %2 for monocular case. The resulting pose estimates are used reliably in Chapter 3 for large scale urban reconstruction.

In the second approach, a tightly coupled indirect feedback Kalman Filter is presented for integration of 2D feature match location estimates and inertial navigation system.

The experimental results are compared against visual only motion estimation techniques based on Kitti benchmark data [1], in Table 4.1. The errors are measured in percent (for translation) and in degrees per meter (for rotation). The angular performance of the proposed algorithm is far better than any algorithm in this table, since an inertial sensor is used in the filter which can provide better angular performance than the camera alone. Based on the performance results of different algorithms in Table 4.1, the best state-of-the-art stereo VO algorithm (MFI) reports an error of %1.5 OR, and monocular VO algorithm (VISO2-M) reports an error of %13. The proposed loosely coupled visual/inertial integration scheme is able to reduce the OR error to %0.2 for stereo (LC-StVO/IMU) and %2 for monocular case (LC-MnVO/IMU). The performance degradation in monocular case is mainly due to scale ambiguity. The tightly coupled monocular visual-inertial integration method (TC-MonoVisual/IMU) is able to reduce the drift slightly as compared to the loosely coupled method.

Table4.1: Visual Odometry OR Errors for different algorithms [1], including the proposed methods.

Rank	**Method**	**Translation**	**Rotation**
1	**Proposed LC-StVO/IMU**	**0.2** %	**0.0005 [deg/m]**
2	**Proposed TC-MonoVisual/IMU**	**1.5** %	**0.0005 [deg/m]**
3	MFI	1.69 %	0.0066 [deg/m]
4	VoBa	1.77 %	0.0066 [deg/m]
5	VISO2-SAM	1.83 %	0.0152 [deg/m]
6	SSLAM	1.87 %	0.0083 [deg/m]
7	eVO	1.93 %	0.0076 [deg/m]
8	**Proposed LC-MonoVO/IMU**	**2** %	**0.0005 [deg/m]**
9	D6DVO	2.10 %	0.0083 [deg/m]
10	GT_VO3pt	2.21 %	0.0117 [deg/m]
11	VISO2-S	2.27 %	0.0152 [deg/m]
12	BoofCV-SQ3	2.27 %	0.0111 [deg/m]
13	TGVO	2.44 %	0.0105 [deg/m]
14	SVO	2.45 %	0.0109 [deg/m]
15	SSLAM-HR	2.45 %	0.0112 [deg/m]
16	KPnP	2.73 %	0.0107 [deg/m]
17	VO3pt	2.93 %	0.0116 [deg/m]
18	VO3ptLBA	3.17 %	0.0180 [deg/m]
19	MSD VO	3.50 %	0.0166 [deg/m]
20	MLM-SFM	4.07 %	0.0104 [deg/m]
21	VOFS	4.21 %	0.0158 [deg/m]
22	VOFSLBA	4.35 %	0.0189 [deg/m]
23	VISO2-M	13.79 %	0.0372 [deg/m]

It should be noted that, vision and inertial sensor fusion algorithms require the precise knowledge of 6-DoF transformation (level-arm/translation and boresight/rotation) between the camera and IMU for a reliable estimation. These unknown constants are modeled in the Kalman filter for on-the-fly estimation. An offline method is also proposed to estimate the transformation for uncalibrated cases in Appendix B.

CHAPTER 5

CONCLUSION

5.1 Summary

This dissertation presents a complete system for real-time, autonomous and geo-registered 3D reconstruction and modeling of urban environments by a camera and inertial sensors. The proposed approach exploits the special structures of urban areas and visual-inertial sensor fusion. The proposed approach is labeled as "complete", since it only requires a time tagged image sequence and inertial sensor readings. All the stages required for a consistent 3D model of the observed urban environment is addressed within the chapters. Below is a brief summary of the research.

3D reconstruction (map building) and pose estimation (localization) problems are tackled in a bilateral solution approach. Moreover, the pose estimation stage is emphasized more than similar works, since it bears importance both for large scale 3D reconstruction and integration of this reconstruction to geographical information systems. The main challenge of urban reconstruction is pointed out as scale. The sequences are long and the captured images are cluttered. Therefore, the proposed solution must be free of pose drift and robust to visual outliers to achieve a complete and consistent reconstruction. The error sources for visual outliers are pointed out as: lightning changes, shadows, glass surfaces, blurring due to motion, and independently moving objects. There are individual visual algorithms that might cope for each error type; however, the computational cost is enormous, if all these algorithms are to be applied.

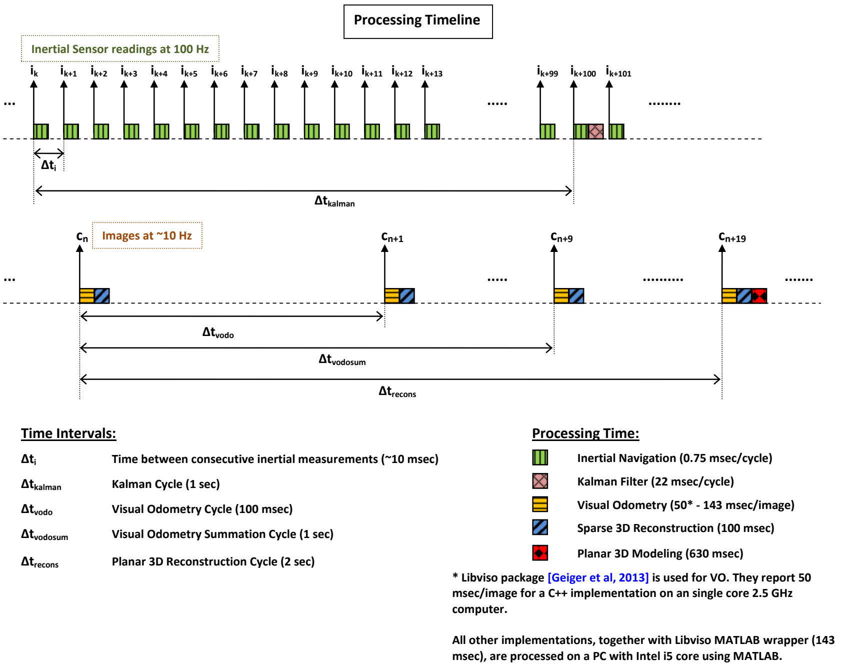
It is shown that visual-inertial sensor fusion might cope for most of these errors with negligible computational cost after eliminating visual outliers by the epipolar constraint and 3D-2D projection tests, since metric camera pose is reliably known. For the 3D modeling stage, the unique properties of urban environments, planarity, orthogonality and verticality, are exploited. The 3D modeling solution is based on sparse 3D point clouds and efficient interpretation of these point clouds by means of unique properties of urban environments. The sparse 3D point clouds are obtained in their geographical coordinates from visual feature matches between consecutive frames by triangulation, since metric camera pose is known. Chapter 3 presents a planar patch modeling for urban reconstruction. A superpixel-based plane association approach is proposed in this stage. The proposed approach is compared against traditional plane sweeping approach. It is shown that obtaining more clean and compact results, as a result of emphasizing the plane priors more is possible by the proposed approach. The superpixel-based method is quite efficient in comparison to pixel-based method in terms of computational complexity. However, superpixel-based methods still have a computational burden for long sequences. Therefore, in addition to superpixel based modeling, a direct plane association approach is presented for real-time applications where the sparse 3D points are projected on virtual plane priors and the convex hull and bounding box planar models are extracted directly from these projected points.

In order to build a metric and precise 3D point cloud, 3D reconstruction algorithms need precise knowledge of the camera poses, as well as reliable visual feature matches. The visual pose estimation, such as SfM and visual odometry, is not reliable for long sequences and suffers from drift in time. A combined visual-inertial pose estimation is proposed in Chapter 4 to overcome the drift problem in the absence of GPS. Two different approaches are proposed for this purpose which utilize visual-inertial sensor fusion: loosely coupled and tightly coupled indirect feedback Kalman filters. In the loosely coupled scheme, visual odometry is used as a delta-position and attitude change aiding source for the inertial navigation system. Since the visual odometry output (delta position and attitude change) is physically comparable to inertial navigation output, it is fairly straightforward to construct a measurement model in terms of the state vector of the system. One of the problems with visual odometry is its non-Gaussian, non-stationary and correlated error characteristics. This behaviour is explored and practical methods are proposed to tackle the problems induced by these error characteristics. Visual odometry pre-filtering and Kalman filter measurement sigma test are introduced for this purpose. Two measurement models are derived for the accumulated and incremental visual odometry measurements. A practical measurement model approach is proposed for the delta position and attitude change measurements that inherently include delayed-states.

There is a major drawback for the loosely coupled integration: its performance depends on the quality of vision only motion estimate. If the visual odometry output is spurious at a certain time, it shall be discarded from the filter. During these visual measurement outages, the system continues with inertial measurements. If these visual outages are instantaneous and does not continue for long durations, the performance is not severely effected, since inertial sensors are adequate to keep the performance within small time intervals. This phenomena is experimentally depicted in monocular visual odometry-inertial sensor integration results. However, if the visual measurement outage duration is longer, the low cost inertial sensors will not be adequate to keep the navigation running within good performance limits, and the drift will might be irreversible, even if the system starts to use visual measurements again. Actually, there might still be valuable information about camera motion somewhere deep in the visual motion estimation stage even if the visual motion output is invaluable. In order to avoid the dependence of the filter on the quality of visual-only motion estimate, a tightly coupled filter is proposed that embeds the vision based motion estimation stage into the Kalman Filter together by the inertial sensor measurements. On the other hand, in the tightly coupled scheme, 2D feature match locations are utilized as an aiding source for the inertial navigation system. The proposed algorithm is tested on real data and it is shown that suppressing the drift is possible with visual-inertial sensor fusion.

As a last remark, vision and inertial sensor fusion algorithms require the precise knowledge of 6-DoF transformation (level-arm/translation and boresight/rotation) between the camera and inertial measurement unit. This problem is formulated and a solution approach is presented in Appendix B. The solution approach is based on an Extended Kalman Filter estimation scheme. The EKF mechanization is similar to the one in Chapter 4. The proposed method does not require any special equipment, except a simple piece of paper, a checkerboard calibration pattern, which is also needed to determine the intrinsic camera parameters.

Real-time applicability is tackled in every step of the proposed solution, and it has been shown that it is possible to get admissible results by integrating inertial sensors to vision algorithms. Inertial sensors are used in pose estimation, feature pre-conditioning, sparse 3D reconstruction, plane extraction, 3D modeling and geo-locational reasoning stages. A detailed process flow is depicted in Figure 5.1 to better illustrate the algorithm flow on a timeline. The real-time applicability can be traced from this figure; the processing time of each stage is explicitly pointed out in the figure. It can be deduced that the proposed solution is near real-time by a MATLAB implementation on an Intel i5 core computer.


Processing Timeline
Inertial Sensor readings at 100 Hz
i_k i_{k+1} i_{k+2} i_{k+3} i_{k+4} i_{k+5} i_{k+6} i_{k+7} i_{k+8} i_{k+9} i_{k+10} i_{k+11} i_{k+12} i_{k+13} i_{k+99} i_{k+100} i_{k+101}
Δt_i
Δt_{kalman}
c_n
Images at ~10 Hz
c_{n+1} c_{n+9} c_{n+19}
Δt_{vodo}
$\Delta t_{vodosum}$
Δt_{recons}

Time Intervals:

Δt_i	Time between consecutive inertial measurements (~10 msec)
Δt_{kalman}	Kalman Cycle (1 sec)
Δt_{vodo}	Visual Odometry Cycle (100 msec)
$\Delta t_{vodosum}$	Visual Odometry Summation Cycle (1 sec)
Δt_{recons}	Planar 3D Reconstruction Cycle (2 sec)

Processing Time:

▥	Inertial Navigation (0.75 msec/cycle)
☒	Kalman Filter (22 msec/cycle)
▤	Visual Odometry (50* - 143 msec/image)
▨	Sparse 3D Reconstruction (100 msec)
🟥	Planar 3D Modeling (630 msec)

* Libviso package [Geiger et al, 2013] is used for VO. They report 50 msec/image for a C++ implementation on an single core 2.5 GHz computer.

All other implementations, together with Libviso MATLAB wrapper (143 msec), are processed on a PC with Intel i5 core using MATLAB.

Figure 5.1: Processing timeline of the overall algorithm

5.2 Conclusions

Visual-Inertial sensor fusion can help to improve visual-only algorithms in an efficient way (such as outlier rejection). It is shown by experiment that visual-inertial sensor fusion can cope with spurious visual features in large scale urban environments, with negligible computational cost by exploiting epipolar constraint and 3D-2D projection tests.

Using heuristics about urban environments leads to efficient, clear and compact modeling results. It is shown by experimentation that, the superpixel-based planar modeling approach is more efficient than the traditional plane sweeping approach in terms of computational cost. It is also shown that obtaining more clean and compact results is possible with the superpixel-based approach by emphasizing the plane priors more. For the large scale urban reconstruction problem, the proposed "direct plane assignment to sparse 3D point cloud by means of vertical facades" method is tested on a large scale urban scene, and near real-time performance is achieved by sacrificing the detail. There is definitely a detail loss in 3D model, as the algorithm makes more assumptions and moves away from pixel-based methods. This is a valid trade-off if the goal is not to build a detailed map, but instead extract a coarse representation of the environment for navigation or other robotic tasks.

For the visual-inertial pose estimation problem, the experimental results are presented for a real dataset collected by a land vehicle which carries a camera and a low cost MEMS inertial sensor onboard in an urban environment. The results showed that it is possible to get admissible navigation performance even with a low performance inertial sensor by visual-inertial sensor fusion, and the loosely coupled visual-inertial fusion results are at least 5 times better than the best visual only algorithm in literature. For the tightly coupled scheme, the feature matches/tracks should be well-conditioned in order to get admissible results. Therefore, it would be best, if the feature matching/tracking algorithm is supported with inertial measurements in an ultra-tightly coupled filter framework. It shall be noted that, stereo setups yield more plausible pose estimation results than monocular setups, since the scale problem is inherently solved in stereo. The experiments also indicate that the effect of visual-inertial integration become more apparent if the path length increases.

5.3 Future Directions

The current content of this dissertation is applicable to solve the pose estimation (navigation) problem in GPS-denied environments for land vehicles by means of visual inertial sensor fusion. This algorithm can be applied to other domains (indoor and airborne) with slight modifications. A large scale 3D urban reconstruction scheme is also proposed which can be used as a starting point for simple city modeling from video.

Below is a non-exhaustive list of pointers for future research efforts:

- Develop a deeply coupled visual-inertial sensor fusion where the feature tracking is supported by the inertial sensors.
- Exploiting visual inertial sensor fusion in the other domains, such as indoor and airborne.
- Visual-inertial sensor fusion for personal navigation problem.
- Proper modeling of low performing inertial sensors in resource constrained mobile systems for reliable and robust estimation frameworks.
- Implementation of the proposed algorithms on resource constrained devices, such as mobile phones, wearable processing systems, embedded flight control processors etc.

REFERENCES

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” *International Journal of Robotics Research (IJRR)*, 2013.

[2] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. van Gool, and W. Purgathofer, “A Survey of Urban Reconstruction,” in *EUROGRAPHICS 2012 State of the Art Reports*, pp. 1–28, Eurographics Association, 2012.

[3] D. Scaramuzza and F. Fraundorfer, “Visual odometry [tutorial],” *Robotics Automation Magazine, IEEE*, vol. 18, pp. 80 –92, dec. 2011.

[4] C. Strecha, W. von Hansen, L. J. V. Gool, P. Fua, and U. Thoennessen, “On benchmarking camera calibration and multi-view stereo for high resolution imagery,” in *CVPR*, 2008.

[5] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Sasstrunk, “Slic superpixels compared to state-of-the-art superpixel methods,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[6] G. Welch and G. Bishop, “An introduction to the kalman filter,” tech. rep., University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995.

[7] E. Imre, *Prioritized 3D Scene Reconstruction and Rate-Distortion Efficient Representation for Video Sequences*. PhD thesis, Middle East Technical University, Turkey, 2007.

[8] J. Hu, S. You, and U. Neumann, “Approaches to large-scale urban modeling,” *IEEE Computer Graphics and Applications*, vol. 23, pp. 62–69, 2003.

[9] N. Cornelis, B. Leibe, K. Cornelis, and L. V. Gool, “Gool. 3d urban scene modeling integrating recognition and reconstruction,” in *IJCV*, 2008.

[10] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles, “Detailed real-time urban 3d reconstruction from video,” *Int. J. Comput. Vision*, vol. 78, pp. 143–167, July 2008.

[11] M. Pollefeys, J.-M. Frahm, F. Fraundorfer, C. Zach, C. Wu, B. Clipp, and D. Gallup, “Challenges in wide-area structure-from-motion,” *Information and Media Technologies*, vol. 6, no. 1, pp. 64–79, 2011.

[12] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, “A comparison and evaluation of multi-view stereo reconstruction algorithms,” in *Proceedings of the 2006*

IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1, CVPR '06, (Washington, DC, USA), pp. 519–528, IEEE Computer Society, 2006.

[13] B. Mičušík and J. Košecká, "Multi-view superpixel stereo in urban environments," *Int. J. Comput. Vision*, vol. 89, pp. 106–119, Aug. 2010.

[14] B. B. Carlos A. Vanegas, Daniel G. Aliaga, "Automatic extraction of manhattan-world building masses from 3d laser range scans," *IEEE Transactions on Visualization and Computer Graphics*, 2012.

[15] C. Früh and A. Zakhor, "Fast 3D model generation in urban environments," in *Multisensor Fusion and Integration for Intelligent Systems, 2001. MFI 2001. International Conference on*, pp. 165–170, IEEE, 2001.

[16] I. Stamos and P. K. Allen, "Geometry and texture recovery of scenes of large scale," *Computer Vision and Image Understanding*, vol. 88, no. 2, pp. 94–118, 2002.

[17] D. Gallup, *Efficient 3d Reconstruction of Large-Scale Urban Environments From Street-Level Video*. PhD thesis, University of North Carolina at Chapel Hill, 2011.

[18] N. Haala and K. heinrich Anders, "Acquisition of 3d urban models by analysis of aerial images, digital surface models and existing 2d building information," in *in 'SPIE Conference on Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision III*, pp. 212–222, 1997.

[19] W. Ribarsky, T. Wasilewski, and N. Faust, "From urban terrain models to visible cities," *IEEE Computer Graphics and Applications*, vol. 22, no. 4, pp. 10–15, 2002.

[20] P. E. Debevec, C. J. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, (New York, NY, USA), pp. 11–20, ACM, 1996.

[21] J. Xiao, T. Fang, P. Tan, P. Zhao, E. Ofek, and L. Quan, "Image-based facade modeling," *ACM Trans. Graph.*, vol. 27, pp. 161:1–161:10, Dec. 2008.

[22] D. Gallup, J. Frahm, and M. Pollefeys, "A heightmap model for efficient 3d reconstruction from street-level video," in *3DPVT10*, pp. xx–yy, 2010.

[23] A. Irschara, C. Zach, and H. Bischof, "Towards wiki-based dense city modeling.," in *ICCV*, pp. 1–8, IEEE, 2007.

[24] S. Teller, "Toward urban model acquisition from geo-located images," in *Proceedings of the 6th Pacific Conference on Computer Graphics and Applications*, PG '98, (Washington, DC, USA), pp. 45–, IEEE Computer Society, 1998.

[25] D. Gallup, J.-M. Frahm, and M. Pollefeys, “Piecewise planar and non-planar stereo for urban scene reconstruction,” *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 0, pp. 1418–1425, 2010.

[26] D. Gallup, M. Pollefeys, and J.-M. Frahm, “3d reconstruction using an n-layer heightmap,” in *Proceedings of the 32nd DAGM conference on Pattern recognition*, (Berlin, Heidelberg), pp. 1–10, Springer-Verlag, 2010.

[27] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski, “Manhattan-world stereo,” *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 0, pp. 1422–1429, 2009.

[28] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool, “3d city modeling using cognitive loops,” in *Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT’06)*, 3DPVT ’06, (Washington, DC, USA), pp. 9–16, IEEE Computer Society, 2006.

[29] C. Früh and A. Zakhor, “An automated method for large-scale, ground-based city model acquisition,” *Int. J. Comput. Vision*, vol. 60, pp. 5–24, Oct. 2004.

[30] C. Früh and A. Zakhor, “Constructing 3D city models by merging aerial and ground views,” *IEEE Computer Graphics and Applications*, vol. 23, no. 6, pp. 52–61, 2003.

[31] F. Lafarge, X. Descombes, J. Zerubia, and M. P. Deseilligny, “3d city modeling based on hidden markov model.,” in *ICIP (2)*, pp. 521–524, IEEE, 2007.

[32] S. Kocaman, L. Zhang, A. Gruen, and D. Poli, “3d city modeling from high-resolution satellite images,” in *ISPRS Workshop on Topographic Mapping from Space*, 2006.

[33] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. New York, NY, USA: Cambridge University Press, 2 ed., 2003.

[34] J. Y. Bouguet, “Camera calibration toolbox for matlab,” 2008.

[35] E. Vural, “Robust extraction of sparse 3d points from image sequences,” Master’s thesis, Middle East Technical University, Turkey, 2008.

[36] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vision*, vol. 60, pp. 91–110, Nov. 2004.

[37] M. Tehrani and M. Seresht, “Pcv 2010 - pose estimation of image sequence captured from urban environment,” in *Photogrammetric Computer Vision and Image Analysis*, p. B:72, 2010.

[38] P. Parsonage, A. Hilton, and J. Starck, “Efficient dense reconstruction from video,” in *Proceedings of the 2011 Conference for Visual Media Production*, CVMP ’11, (Washington, DC, USA), pp. 30–38, IEEE Computer Society, 2011.

[39] A. Geiger, J. Ziegler, and C. Stiller, “Stereoscan: Dense 3d reconstruction in real-time,” in *IEEE Intelligent Vehicles Symposium*, (Baden-Baden, Germany), June 2011.

[40] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *Computer Vision–ECCV 2006*, pp. 404–417, Springer, 2006.

[41] D. Nister, O. Naroditsky, and J. Bergen, “Visual odometry,” *2012 IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 652–659, 2004.

[42] B. Kitt, A. Geiger, and H. Lategahn, “Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme,” in *IEEE Intelligent Vehicles Symposium*, (San Diego, USA), June 2010.

[43] 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, pp. 381–395, June 1981.

[44] K. Konolige, M. Agrawal, and J. Sola, “Large scale visual odometry for rough terrain,” in *In Proc. International Symposium on Robotics Research*, 2007.

[45] R. Subbarao, Y. Genc, and P. Meer, “Nonlinear mean shift for robust pose estimation,” in *Proceedings of the Eighth IEEE Workshop on Applications of Computer Vision*, WACV ’07, (Washington, DC, USA), pp. 6–, IEEE Computer Society, 2007.

[46] E. Tola and A. A. Alatan, “Fast outlier rejection by using parallax-based rigidity constraint for epipolar geometry estimation,” in *Proceedings of the 2006 international conference on Multimedia Content Representation, Classification and Security*, MRCS’06, (Berlin, Heidelberg), pp. 578–585, Springer-Verlag, 2006.

[47] C. N. Taylor, “Improved fusion of visual measurements through explicit modeling of outliers,” *Position Location and Navigation Symposium (PLANS), IEEE/ION*, 2012.

[48] R. Jiang, R. Klette, and S. Wang, “Modeling of unbounded long-range drift in visual odometry,” in *Proceedings of the 2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, PSIVT ’10, (Washington, DC, USA), pp. 121–126, IEEE Computer Society, 2010.

[49] R. O. Duda and P. E. Hart, “Use of the hough transformation to detect lines and curves in pictures,” *Commun. ACM*, vol. 15, pp. 11–15, Jan. 1972.

[50] R. Toldo and A. Fusiello, “Robust multiple structures estimation with j-linkage,” in *In: Proceedings of the European Conference of Computer Vision*, pp. 537–547, 2008.

[51] B. E. Okorn, X. Xiong, B. Akinci, and D. Huber, “Toward automated modeling of floor plans,” in *Proceedings of the Symposium on 3D Data Processing, Visualization and Transmission*, May 2010.

[52] A. Ozturk, “Development, implementation, and testing of a tightly coupled integrated ins/gps system,” Master’s thesis, Middle East Technical University, 2003.

[53] M. Veth and J. Raquet, "Fusion of low-cost imaging and inertial sensors for navigation," tech. rep., Air Force Institute of Technology, 2007.

[54] G. L. Andrews, "Implementation considerations for vision-aided inertial navigation," Master's thesis, Northeastern University, Boston Massachusetts, 2008.

[55] J. Jurado, K. Fisher, and M. Veth, "Inertial and imaging sensor fusion for image-aided navigation with affine distortion prediction," in *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, pp. 518 –526, april 2012.

[56] M. Li and A. I. Mourikis, "Improving the accuracy of ekf-based visual-inertial odometry," in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Minneapolis, MN), pp. 828–835, May 2012.

[57] J.-O. Nilsson, D. Zachariah, M. Jansson, and P. Handel, "Realtime implementation of visual-aided inertial navigation using epipolar constraints," in *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, pp. 711 –718, april 2012.

[58] D. D. Diel, P. DeBitetto, and S. Teller, "Epipolar constraints for vision-aided inertial navigation," in *In EEE Workshop on Motion and Video Computing*, pp. 221–228, 2005.

[59] D. Titterton, J. Weston, and I. of Electrical Engineers, *Strapdown Inertial Navigation Technology*. Iee Radar Series, Institution of Electrical Engineers, 2004.

[60] N. B. Seymen, "Robust set-valued estimation and its application to in-flight alignment of sins," Master's thesis, Middle East Technical University, 2005.

[61] P. S. Maybeck, *Stochastic Models, Estimation, and Control*, vol. 141-1 of *Mathematics in Science and Engineering*. New York: Academic Press, 1979.

[62] R. G. Brown and P. Y. C. Hwang, *Introduction to random signals and applied kalman filtering*. New York, NY: Wiley, 1997.

[63] J.-P. Tardif, M. D. George, M. Laverne, A. Kelly, and A. Stentz, "A new approach to vision-aided inertial navigation.," in *IROS*, pp. 4161–4168, IEEE, 2010.

[64] S. Roumeliotis, A. Johnson, and J. Montgomery, "Augmenting inertial navigation with image-based motion estimation," in *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, vol. 4, pp. 4326 – 4333 vol.4, 2002.

[65] J. Farell, "Carrier phase processing without integers," in *ION 57th Annual Meeting*, 2001.

[66] J. Wendel and G. F. Trommer, "Tightly coupled gps/ins integration for missile applications," *Aerospace Science and Technology*, vol. 8, pp. 627–634, 2004.

[67] D. Strelow, *Motion Estimation from Image and Inertial Measurements*. PhD thesis, School of Computer Science Carnegie Mellon University, 2004.

[68] J. Kim and S. Sukkarieh, “Real-time implementation of airborne inertial-slam,” *Robot. Auton. Syst.*, vol. 55, pp. 62–71, Jan. 2007.

[69] A. I. Mourikis and S. I. Roumeliotis, “A multi-state constraint Kalman filter for vision-aided inertial navigation,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Rome, Italy), pp. 3565–3572, April 10-14 2007.

[70] D. Zachariah and M. Jansson, “Camera-aided inertial navigation using epipolar points,” in *Proceedings of PLANS 2010*, (Palm Springs, California), May 2010.

[71] S. Zhao, F. Lin, K. Peng, B. Chen, and T. Lee, “Homography-based vision-aided inertial navigation of uavs in unknown environments,” in *AIAA GUIDANCE, NAVIGATION, AND CONTROL CONFERENCE*, 2012.

[72] E. Montijano, C. Sagues, E. Montijano, and C. Sagues, “Position-based navigation using multiple homographies,” in *IEEE Int. Conference on Emergent Technologies and Factory Automation (ETFA)*, pp. 994–1101, 2008.

[73] L. C. Cesario Vincenzo Angelino, Vincenzo Rosario Baraniello, “High altitude uav navigation using imu, gps and came,” in *16th International Conference on Information Fusion*, 2013.

[74] M. Li and A. I. Mourikis, “Vision-aided inertial navigation for resource-constrained systems,” in *Proceedings of the IEEE/RSJ International Conference on Robotics and Intelligent Systems (IROS)*, (Vilamoura, Portugal), pp. 1056–1063, October 2012.

[75] F. M. Mirzaei and S. I. Roumeliotis, “A kalman filter-based algorithm for imu-camera calibration: Observability analysis and performance evaluation,” *Trans. Rob.*, vol. 24, pp. 1143–1156, Oct. 2008.

[76] J. Lobo and J. Dias, “Relative pose calibration between visual and inertial sensors.,” *I. J. Robotic Res.*, vol. 26, no. 6, pp. 561–575, 2007.

[77] J. Kelly and G. S. Sukhatme, “Fast relative pose calibration for visual and inertial sensors,” in *Experimental Robotics: The Eleventh International Symposium* (O. Khatib, V. Kumar, and G. J. Pappas, eds.), vol. 54/2009 of *Springer Tracts in Advanced Robotics*, pp. 515–524, Berlin, Germany: Springer, Apr 2009.

APPENDIX A

FRAME AND COORDINATE SYSTEM DEFINITIONS

The definitions in this appendix follow [60].

A.1 Reference Frames

A frame is an unbounded continuous set of points over the Euclidean three dimensional space with invariant distances and which possesses, as a subset, at least three non-collinear points. It is a physical entity consisting of points without relative movement. At least three non-collinear points of a frame must be identified for the frame to occupy the three dimensional space. If three points are picked and connected with a fourth base point, this triad defines the position of the frame completely. The triad can be built from a set of three orthonormal base vectors. The reference frames used in the inertial navigation calculations are defined below.

A.1.1 Earth Centered Inertial Frame (i-frame)

It has its base point at the center of the Earth. Its orientation remains fixed in the ecliptic and is defined by the base vectors i_1, i_2 and i_3 . The first base vector i_1 is along the line from the base point to vernal equinox, the point where sun crosses the equator in the spring, third base vector i_3 is defined as the Earth's axis of rotation and i_2 completes the righthanded triad. Actually, this frame is not an inertial frame due to the Earth's motion around the sun, but for short periods, this can be neglected and it can be considered as an inertial frame.

A.1.2 Earth Centered Earth-Fixed Frame (e-frame)

It has its base point at the centre of the Earth. Its orientation is defined by the base vectors e_1, e_2 and e_3. The first base vector e_1 is along the intersection of the plane of the Greenwich meridian with the Earth's equatorial plane. Third base vector e_3 is along the Earth's axis of rotation and e_2 completes the right-handed triad.

A.1.3 Navigation Frame (n-frame)

It has its base point at the location of navigation systems with its longitude λ and latitude L, The first base vector n_1 points north, n_2 is along east and n_3 is along the local vertical (down).

A.1.4 Body Frame (b-frame)

It has its base point at the location of navigation system. Its orientation is defined by the base vectors b_1, b_2 and b_3. The first base vector b_1 is along the front side of the navigation system panel, b_2 is along the right side of the navigation system, and b_3 is along the down side of the system.

A.1.5 Vehicle Frame (v-frame)

It has its base point at the location of odometer reference point. Its orientation is defined by the base vectors v_1, v_2 and v_3. The first base vector v_1 is along the front side of the vehicle, v_2 is along the right side of the vehicle, and v_3 is along the down side of the vehicle.

A.2 Coordinate Systems

A coordinate system is defined as an abstract entity that establishes a one-to-one correspondence between the elements of the Euclidean three dimensional space and the coordinates. Thus, it is just a mathematical tool. Any coordinate system can be assigned to a frame. The coordinate systems used in the inertial navigation calculations are defined below.

A.2.1 Earth Centered Inertial Coordinate System

The coordinate system is chosen as the preferred coordinate system of inertial frame triad i_1, i_2 and i_3. In other words, x^i axes is along the line from the base point to vernal equinox, z^i axes is defined as the Earth's axis of rotation and y^i completes the right-handed coordinate system.

A.2.2 Earth Centered Earth Fixed Coordinate System

The coordinate system is chosen as the preferred coordinate system of Earth Centered Earth Fixed triad e_1, e_2 and e_3. That is, x^e axis is along the intersection of the plane of the Greenwich meridian with the Earth's equatorial plane, z^e axis is along the spin axis of the Earth and y^e completes the right-handed coordinate system.

A.2.3 Navigation Coordinate System

The coordinate system is chosen as the preferred coordinate system of navigation frame triad n_1, n_2 and n_3. At a specified point on the surface of the Earth with its longitude λ and latitude L, x^n axis points north, the y^n axis points east and z^n points the local vertical (down).

A.2.4 Body Coordinate System

The coordinate system is chosen as the preferred coordinate system of body frame triad b_1, b_2 and b_3. In other words, x^b axes is along the front side of the navigation system panel, y^b axes is along right side of the navigation system and z^b is along the down side of the system.

A.2.5 Coordinate Transformation

Coordinate systems are related by coordinate transformation matrices that relabel the coordinates of a vector in different coordinate systems. Coordinate transformation matrices are orthogonal matrices and determinant of them are equal to one provide that right-handed Cartesian coordinate systems are used. Moreover, taking the transpose of the transformation matrices inverses the sequence of the transformation.

Direction Cosine Matrix:

Direction Cosine matrix (DCM) which is denoted as C_a^b , transforms the coordinates of a vector from coordinate system-a, associated with frame-a to coordinate system-b, associated with frame-b. It is a 3x3 matrix, columns of which consist of frame-a base vectors, expressed in coordinate system-b. On the other hand, the rows are the transposed base vectors of frame-b and expressed in coordinate system-a. The element of the matrix in the i^{th} row and j^{th} column can be interpreted as cosine angle between the i^{th} base vector of the frame-b and j^{th} base vector of the frame-a. A vector quantity defined in coordinate system-a may be expressed in coordinate system-b by pre-multiplying the vector by the direction cosine matrix as follows:

$$\mathbf{v}^b = C_a^b \, \mathbf{v}^a \tag{A.1}$$

Euler Angles:

Angular orientation of one coordinate system with respect to another coordinate system can be expressed as by three successive rotations. The rotation angles are called Euler angles. The sequence of rotations is not unique. In this dissertation, transformation from navigation coordinate system to body coordinate system will be expressed as follows

1. Rotation through an angle ψ (heading) about z_n.

2. Rotation through an angle θ (pitch) about new y axis.

3. Rotation through an angle ϕ (roll) about new x axis.

where the superscript "n" in z_n stands for navigation coordinate system. Transformation matrix from body coordinate system to navigation coordinate system is expressed as:

$$C_b^n = \begin{bmatrix} \cos\theta\ \cos\psi & -\cos\phi\ \sin\psi + \sin\phi\ \sin\theta\ \cos\psi & \sin\phi\ \sin\psi + \cos\phi\ \sin\theta\ \cos\psi \\ \cos\theta\ \sin\psi & \cos\phi\ \cos\psi + \sin\phi\ \sin\theta\ \sin\psi & -\sin\phi\ \cos\psi + \cos\phi\ \sin\theta\ \sin\psi \\ -\sin\theta & \sin\phi\ \cos\theta & \cos\phi\ \cos\theta \end{bmatrix} \tag{A.2}$$

APPENDIX B

CAMERA INERTIAL SENSOR CALIBRATION

Vision and inertial sensor fusion algorithms require the precise knowledge of 6-DoF transformation (level-arm/translation and boresight/rotation) between the camera and inertial measurement unit. This leads to the necessity of a prior or on-the-fly camera-IMU calibration process. Errors in the camera-IMU calibration process cause biases that reduce the estimation accuracy and can even lead to divergence of any estimator processing the measurements from both sensors.

In the industry, the camera-IMU calibration is generally achieved using precise CAD drawings and complex optical methods requiring expensive and delicate equipment. Most of the proposed approaches in the literature solve the problem in statistical framework and use a checkerboard pattern [75], [76], [77].

In this chapter, an Indirect Feedback Kalman Filter based algorithm is proposed for determining the 6-DoF transformation between a single camera and an inertial measurement unit using measurements only from these two sensors. The proposed method does not require any special equipment, except a simple piece of paper, i.e. a checkerboard calibration pattern, which is also needed to determine the intrinsic camera parameters. In addition, the proposed algorithm computes the uncertainty in the estimated quantities in terms of Kalman Filter covariance. The proposed approach is similar to the approach by [75]. The IMU Camera calibration is achieved by a series of processes,

1. Find an initial estimate for the camera pose from the first image.
2. Combine the initial camera pose estimate with an approximate unknown transformation to get an initial IMU pose estimate.
3. Sequentially process measurements from the camera and IMU by means of a Kalman Filter to estimate the unknown transformation together with the position, velocity and attitude of the two sensors.

The Kalman Filter states are augmented by the unknown 6-DoF transformation parameters (3 Euler angles, 3 translational components). In this formulation, the error states are propagated instead of the total states and the estimated errors are fed back to the navigation calculations to correct them. By sequentially processing measurements from the camera and the IMU, the Kalman Filter is able to refine the initial estimate for the unknown transformation while simultaneously tracking the position, velocity and attitude of the two sensors. The main steps of the algorithm can be summarized as follows:

1. Initialization
2. Navigation Update (at IMU rate)
3. Kalman Time Update (at image rate)
4. Kalman Measurement Update (at image rate)
5. Kalman Error State Feedback (at image rate)

The inertial navigation calculations (Navigation Update) will be the same as in Section 4.1. The Kalman Filter mechanization will be similar to the one in Chapter 4.

B.1 Coordinate System Definitions

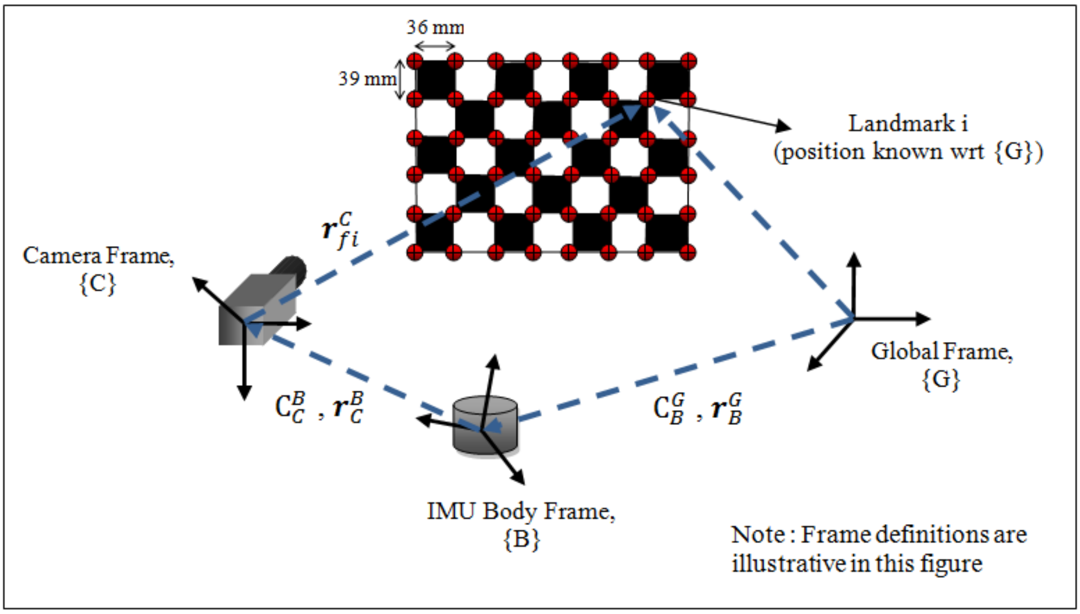
It is necessary to define the coordinate systems for global, camera and IMU clearly before going into details of the proposed algorithm.

$\boldsymbol{r}_{fi}^{G}$: Position of i_{th} landmark in Global Reference Frame, {G}

$\boldsymbol{r}_{fi}^{C}$: Position of i_{th} landmark in Camera Frame, {C}

C_B^G, $\boldsymbol{r}_{B}^{G}$: Orientation and position of IMU Body Frame {B} wrt {G}

C_B^C, $\boldsymbol{r}_{C}^{B}$: Orientation and position of Camera Frame wrt {B}



Note : Frame definitions are illustrative in this figure

Figure B.1: Coordinate System Definitions for IMU Camera Calibration

B.2 Estimation of Calibration Parameters

An indirect Kalman Filter is used to estimate the error states of the navigation variables together with the gyro and accelerometer biases and the unknown transformation between the camera and the IMU.

State Space Model of the System:

The error state vector is defined as:

$$\boldsymbol{x} = \left[\boldsymbol{\psi}^G_{BG}; \quad \delta\boldsymbol{v}^G_B; \quad \delta\boldsymbol{r}^G_B; \quad \boldsymbol{b}^g; \quad \boldsymbol{b}^a; \quad \boldsymbol{\psi}^C_{BC}; \quad \delta\boldsymbol{r}^B_C \quad \right]^T_{1x21}$$

$\boldsymbol{\psi}^G_{BG}$	Attitude error of the body represented in global frame as (small) Euler angles
$\delta\boldsymbol{v}^G_B$	Velocity error of the body represented in global frame
$\delta\boldsymbol{r}^G_B$	Position error of the body represented in global frame
$\boldsymbol{b}^g$	Gyroscope bias
$\boldsymbol{b}^a$	Accelerometer bias
$\boldsymbol{\psi}^C_{BC}$	Camera to IMU boresight (attitude) error in camera frame as (small) Euler angles
$\delta\boldsymbol{r}^B_C$	Position error of the camera to IMU lever arm represented in IMU frame

Filter Initialization:

The initial position, velocity and attitude of the body in G shall be estimated to begin estimation. The body is assumed to be stationary initially. Therefore the initial velocity is zero. The first camera image is processed to compute an initial estimate for the camera position and attitude using the visual features (corners of the squares in the calibration pattern) whose positions are known with respect to global frame G. Then an approximate estimate for the unknown Camera-IMU transformation is used to initialize the position and attitude of the IMU. This estimate can be found manually or using CAD drawings. The initial position and attitude of the IMU is then computed using the following equations:

$$\begin{aligned} \boldsymbol{r}_C^B &= \boldsymbol{r}_C^G - \mathrm{C}_B^G\, \boldsymbol{r}_C^B \\ \mathrm{C}_B^G &= \mathrm{C}_C^G\, \mathrm{C}_B^C \end{aligned} \tag{B.1}$$

Time Update:

The state vector is projected ahead with the following state transition formula:

$$\hat{\boldsymbol{x}}_k^- = \mathrm{A}_k\, \hat{\boldsymbol{x}}_{k-1} \tag{B.2}$$

The state error covariance is projected ahead with the following formula:

$$\mathrm{P}_k^- = \mathrm{A}_k\, (\, \mathrm{P}_{k-1} + 0.5\, Q)\, \mathrm{A}_k^T + Q \tag{B.3}$$

State Transition Matrix is construction as follows:

$$\begin{aligned} \Phi_k &= \begin{pmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -\mathrm{C}_B^G & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & \mathrm{C}_B^G & 0_{3\times3} & 0_{3\times3} \\ \lfloor \boldsymbol{f}^n \times \rfloor & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{pmatrix} \\ \mathrm{A}_k &= I_{3\times3} + \Phi_k + 0.5\, \Phi_k^2 \end{aligned} \tag{B.4}$$

Measurement Update:

Discrete Time Kalman Filter measurement update equations are as follows:

$$\begin{aligned} \mathrm{K}_k &= \mathrm{P}_k^-\, \mathrm{H}_k^T \left(\mathrm{H}_k\, \mathrm{P}_k^-\, \mathrm{H}_k^T + \mathrm{R}\right)^{-1} \\ \hat{\boldsymbol{x}}_k &= \hat{\boldsymbol{x}}_{k-1} + \mathrm{K}_k \left(z_k - \mathrm{H}_k\, \hat{\boldsymbol{x}}_k^-\right) \\ \mathrm{P}_k &= (\mathrm{I} - \mathrm{K}_k\, \mathrm{H}_k)\, \mathrm{P}_k^- \end{aligned} \tag{B.5}$$

The IMU camera moves continuously and records images of a calibration pattern. These are then processed to detect and identify point features whose positions are known with respect to the global frame of reference. Once this process is completed for each image, a list of point features along with their measured image coordinates $(u_i,\ v_i)$ is provided to the Kalman Filter, which uses them to update the state estimates. The projective camera model is employed as,

$$z_i \;=\; \begin{bmatrix} u_i \\ v_i \end{bmatrix} + \boldsymbol{\eta}_i \;=\; \begin{bmatrix} x_i/z_i \\ y_i/zi \end{bmatrix} + \boldsymbol{\eta}_i \;=\; h(\boldsymbol{x}\,,\ \boldsymbol{r}^G_{fi}) \;+\; \boldsymbol{\eta}_i \tag{B.6}$$

$$\boldsymbol{r}^C_{fi} \;=\; \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \;=\; \mathrm{K}\ \cdot \mathrm{C}^C_G\ \cdot \left[\mathrm{I}\,|\ -\boldsymbol{r}^G_C\right]\ \cdot \boldsymbol{r}^G_{fi} \tag{B.7}$$

where, K is the camera calibration matrix. Expressing C^C_G and $\boldsymbol{r}^G_C$ in terms of the total states, the equation becomes,

$$\boldsymbol{r}^C_{fi} \;=\; \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \;=\; \mathrm{K}\ \cdot \mathrm{C}^C_B\ \cdot \mathrm{C}^B_G\ \left[\mathrm{I}\,|\ -\left(\boldsymbol{r}^G_B\ +\ \mathrm{C}^G_B \cdot \boldsymbol{r}^B_C\right)\right]\ \cdot \boldsymbol{r}^G_{fi} \tag{B.8}$$

By re-arranging the terms equation becomes,

$$\boldsymbol{r}^C_{fi} \;=\; \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \;=\; \mathrm{K}\ \cdot\left(\mathrm{C}^C_B\ \cdot \mathrm{C}^B_G\ \cdot\left(\boldsymbol{r}^G_{fi}\ -\ \boldsymbol{r}^G_B\right)\ -\ \mathrm{C}^C_B\ \cdot \boldsymbol{r}^B_C\right) \tag{B.9}$$

The measurement model is expressed in terms of the total states and the measurement. We have to linearize this model in order to construct the H matrix of the discrete time system. We prefer to linearize this equation by means of perturbation. Another approach might be to use the Jacobians. The perturbation equations are,

$$\begin{gathered} \mathrm{C}^C_B \;=\; \left(\mathrm{I}\ -\ \Psi^C_B \cdot \tilde{\mathrm{C}}^C_B\right)\,, \qquad \mathrm{C}^B_G \;=\; \left(\mathrm{I}\ -\ \Psi^B_G \cdot \tilde{\mathrm{C}}^B_G\right) \\ \delta\boldsymbol{r}^B_C \;=\; \boldsymbol{r}^B_C\ -\ \tilde{\boldsymbol{r}}^B_C\,, \qquad \delta\boldsymbol{r}^G_B \;=\; \boldsymbol{r}^G_B\ -\ \tilde{\boldsymbol{r}}^G_B \end{gathered} \tag{B.10}$$

The measurement error equation using the perturbed states is,

$$\delta z_i \;=\; z_i\ -\ \tilde{z}_i\,, \qquad z_i \;=\; z^{true}_i\ +\ \boldsymbol{\eta}_i \tag{B.11}$$

$$\begin{aligned} \delta z_i \;=\;\; & \mathrm{K}\cdot\left(\left(\left(\mathrm{I}\ -\ \Psi^C_B\right)\cdot\tilde{\mathrm{C}}^C_B\right)\cdot\left(\left(\mathrm{I}\ -\ \Psi^B_G\right)\cdot\tilde{\mathrm{C}}^B_G\right)\cdot\ \left(\boldsymbol{r}^G_{fi}\ -\ \left(\boldsymbol{r}^G_B\ +\ \delta\boldsymbol{r}^G_B\right)\right)\ -\right. \\ & \left(\left(\mathrm{I}\ -\ \Psi^C_B\right)\cdot\tilde{\mathrm{C}}^C_B\right)\cdot\left(\boldsymbol{r}^B_C\ +\ \delta\boldsymbol{r}^B_C\right)\Big)\ -\ \mathrm{K}\cdot\left(\tilde{\mathrm{C}}^C_B\cdot\tilde{\mathrm{C}}^B_G\cdot\left(\boldsymbol{r}^G_{fi}\ -\ \boldsymbol{r}^G_B\right)\ -\right. \\ & \left.\tilde{\mathrm{C}}^C_B\cdot\boldsymbol{r}^B_C\right) \end{aligned} \tag{B.12}$$

By eliminating the second order error terms and re-arranging, the equation becomes,

$$\begin{aligned} \delta z_i \;=\;\; & \mathrm{K}\cdot\left(-\tilde{\mathrm{C}}^C_B\cdot\tilde{\mathrm{C}}^B_G\cdot\delta\boldsymbol{r}^G_B\ -\ \tilde{\mathrm{C}}^C_B\cdot\tilde{\mathrm{C}}^B_G\cdot\Psi^B_G\cdot\left(\boldsymbol{r}^G_{fi}\ -\ \tilde{\boldsymbol{r}}^G_B\right)\ -\right. \\ & \left.\Psi^C_B\cdot\tilde{\mathrm{C}}^C_B\cdot\left(\tilde{\mathrm{C}}^B_G\ \cdot\left(\boldsymbol{r}^G_{fi}\ -\ \boldsymbol{r}^G_B\right)\ -\ \boldsymbol{r}^B_C\right)\ -\ \tilde{\mathrm{C}}^C_B\cdot\delta\boldsymbol{r}^B_C\right) \end{aligned} \tag{B.13}$$

Expressing the equation in terms of the error states,

$$\begin{aligned} \delta z_i \;=\;\; & \mathrm{K}\cdot\left(-\tilde{\mathrm{C}}^C_B\cdot\tilde{\mathrm{C}}^B_G\cdot\delta\boldsymbol{r}^G_B\ -\ \tilde{\mathrm{C}}^C_B\cdot\left\lfloor\tilde{\mathrm{C}}^B_G\ \cdot\left(\boldsymbol{r}^G_{fi}\ -\ \boldsymbol{r}^G_B\right)\ \times\right\rfloor\cdot\Psi^B_G\ -\right. \\ & \left.\tilde{\mathrm{C}}^C_B\cdot\left\lfloor\tilde{\mathrm{C}}^B_G\ \cdot\left(\boldsymbol{r}^G_{fi}\ -\ \boldsymbol{r}^G_B\right)\ -\ \boldsymbol{r}^B_C\times\right\rfloor\ -\ \tilde{\mathrm{C}}^C_B\cdot\delta\boldsymbol{r}^B_C\right) \end{aligned} \tag{B.14}$$

The measurement matrix becomes,

$$\mathrm{H}^i = \mathrm{J}^i_{cam} \cdot \left[\mathrm{J}^i_{\Psi_G} \quad 0_{3\times3} \quad \mathrm{J}^i_{P_b} \quad 0_{3\times6} \quad \mathrm{J}^i_{\Psi_C} \quad \mathrm{J}^i_{P_c} \right] \tag{B.15}$$

with,

$$\begin{aligned}
\mathrm{J}^i_{cam} &= \frac{1}{\hat{\boldsymbol{z}}_i^2} \begin{bmatrix} f_x \hat{\boldsymbol{z}}_i & 0 & -f_x \hat{\boldsymbol{x}}_i \\ 0 & f_y \hat{\boldsymbol{z}}_i & -f_y \hat{\boldsymbol{y}}_i \end{bmatrix} \\
\mathrm{J}^i_{\Psi_G} &= -\tilde{\mathrm{C}}^C_B \cdot \left\lfloor \tilde{\mathrm{C}}^B_G \cdot \left(\boldsymbol{r}^G_{fi} - \boldsymbol{r}^G_B \right) \times \right\rfloor \\
\mathrm{J}^i_{P_b} &= -\tilde{\mathrm{C}}^C_B \cdot \tilde{\mathrm{C}}^B_G \\
\mathrm{J}^i_{\Psi_C} &= -\tilde{\mathrm{C}}^C_B \cdot \left\lfloor \tilde{\mathrm{C}}^B_G \cdot \left(\boldsymbol{r}^G_{fi} - \boldsymbol{r}^G_B \right) - \boldsymbol{r}^B_C \times \right\rfloor \\
\mathrm{J}^i_{P_b} &= -\tilde{\mathrm{C}}^C_B
\end{aligned} \tag{B.16}$$

When observations to N features are available concurrently, these observations are stack to one measurement vector, $\boldsymbol{z} = \left[\boldsymbol{z}_1^T \ \boldsymbol{z}_2^T \ ... \ \boldsymbol{z}_N^T \right]^T$, to form a single batch-form update equation. Similarly, the batch measurement matrix, $\mathrm{H} = \left[\mathrm{H}^{1,T} \ \mathrm{H}^{2,T} \ ... \ \mathrm{H}^{N,T} \right]^T$, is formed using each observations measurement matrix as described in [75].

Finally, the measurement residual is computed as,

$$\delta \boldsymbol{z}_i = \boldsymbol{z}_i - \tilde{\boldsymbol{z}}_i \cong \mathrm{H}_i \, \boldsymbol{x} + \boldsymbol{\eta}_i \tag{B.17}$$

where $\boldsymbol{\eta} = \left[\boldsymbol{\eta}_1^T \ \boldsymbol{\eta}_2^T \ ... \ \boldsymbol{\eta}_N^T \right]$ is the measurement noise with covariance $\mathrm{R} = diag(\mathrm{R}_i)$, $i = 1..N$.

B.3 Experimental Results

B.3.1 Simulated Data

The experiment is first conducted on a simulated data set. The simulation data is created in the following steps:

1. Create artificial coordinate systems for Global, Camera and IMU frames. Define the position and orientation of each coordinate system with respect to the others.
2. Create an artificial calibration pattern in the Global Frame
3. Initialize IMU error characteristics
4. Initialize Camera error characteristics
5. Define the trajectory in terms of acceleration, velocity and attitude changes
6. Create the defined trajectory navigation variables (ground truth velocity, position and attitude)

7. Create the simulated IMU data in terms of delta velocities and delta angles (noisy)
8. Create the simulated camera measurements (noisy feature locations in the camera image plane)

Global Frame:

The artificial calibration pattern is assumed to be same as the true pattern. The artificial pattern is placed in the direction of gravity X_G. The upper left corner of the pattern is assumed to be the origin of the Global Frame. The coordinate system definitions are as defined in Figure 2 and Figure 3. In the Global Coordinate System Z_G is assumed to direct true north. Y_G directs west. With these assumptions, the Global Coordinate System becomes a north oriented local level frame.

Camera is assumed to be placed in front of the calibration pattern. The transformation from the camera frame to the global frame is assumed to be the following matrix:

$$Camera\ to\ Global\ Frame\ Transformation = \mathrm{T}_C^G = \begin{bmatrix} 0 & 0 & -1 & 5000 \\ -1 & 0 & 0 & 1000 \\ 0 & 0 & -1 & 10 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

The above transformation corresponds to the following Euler angles that are applied in *yaw* → *pitch* → *roll* order, and translational components defined in global frame:

$$Yaw = 90\,degrees, \quad Pitch = 0\,degrees, \quad Roll = 180\,degrees$$
$$r_{xC}^G = 5000\,mm, \quad r_{yC}^G = 1000\,mm, \quad r_{zC}^G = 10\,mm$$

The position of the origin of IMU body in geographic coordinate system is assumed to be $Latitude = 40.085$, $Longitude = 33.022$, $Altitude = 900$. The transformation from IMU frame to the camera frame is assumed to be the following, assuming that the Euler angles are applied in *yaw* → *pitch* → *roll* order, and translational components defined in global frame:

$$Yaw_{true} = 2.5\,degrees, \quad Pitch_{true} = -0.5\,degrees, \quad Roll_{true} = -90.5\,degrees$$
$$r_{xC}^G = 100\,mm, \quad r_{yC}^G = 50\,mm, \quad r_{zC}^G = -15\,mm$$

The initial estimates for these parameters, that will be supplied to the filter, are as follows:

$$Yaw_{init} = -0.5\,degrees, \quad Pitch_{init} = 3\,degrees, \quad Roll_{init} = -93.5\,degrees$$
$$r_{xC}^G = 150\,mm, \quad r_{yC}^G = 25\,mm, \quad r_{zC}^G = -55\,mm$$

The camera is assumed to make the following motion in front of the calibration pattern: The resulting boresight estimates are given in the following figure:

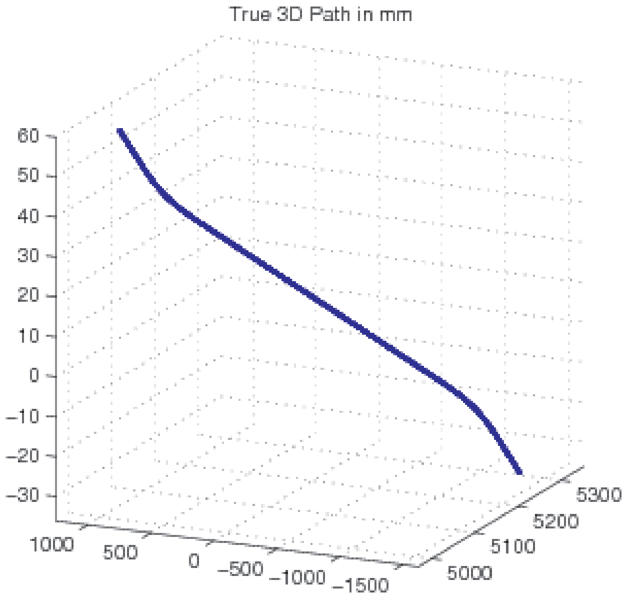


Figure B.2: Simulated 3D route of the camera-IMU sensor rig.

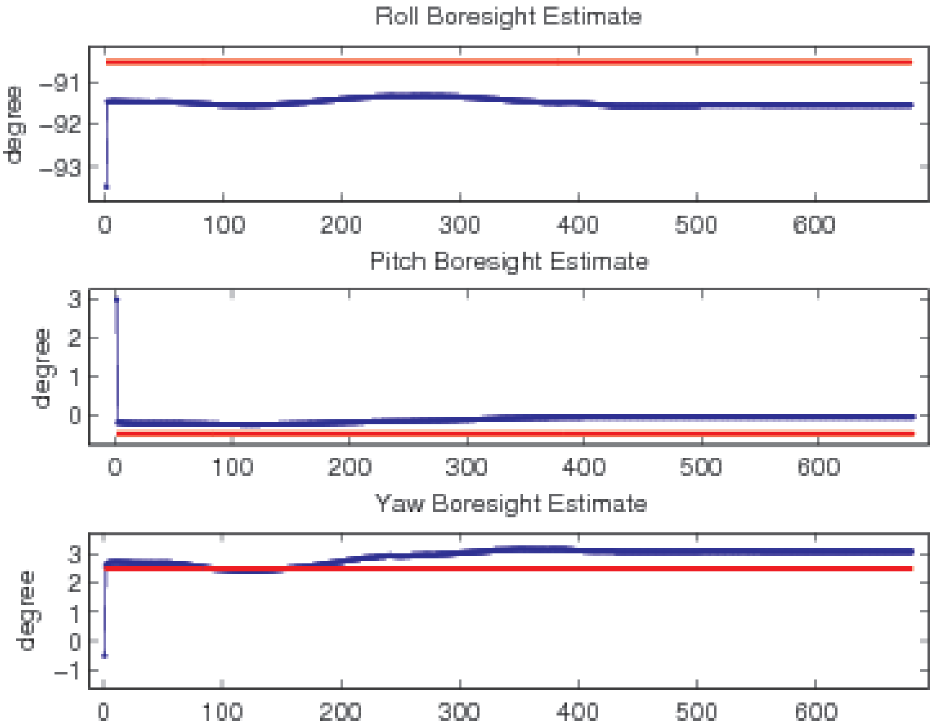


Figure B.3: Camera to IMU boresight estimates. True values are depicted with red lines.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: SIRTKAYA, Salim
Nationality: Turkish (TC)
Date and Place of Birth: 07 October 1979, Trabzon
Marital Status: Married
Phone: +90 (532) 547-6861, +1 (425) 214-6136

EDUCATION

Degree	**Institution**	**Year of Graduation**
M.S.	METU EEE	2001 - 2004
B.S.	METU EEE	1997 - 2001
High School	Trabzon Yomra Fen Lisesi	1997

PROFESSIONAL EXPERIENCE

Year	**Place**	**Enrollment**
(2001 - Present)	ASELSAN	Navigation Systems Design Engineer
(2000 - 2001)	ASELSAN	Intern

PUBLICATIONS

International Conference Publications

1. S. Sırtkaya, A. Alatan, "3D Modeling of Urban Areas Using Plane Hypotheses", presented at the 20th IEEE Conference on Signal Processing and Communication Applications (SIU 2012, Muğla, Turkey, 2012 (in Turkish).

2. S. Sırtkaya, B. Seymen, A. Alatan, "Loosely Coupled Kalman Filtering for Fusion of Visual Odometry and Inertial Navigation", presented at the 16th International Conference on Information Fusion, Istanbul, Turkey, July 2013

3. S. Sırtkaya, B. Seymen, A. Alatan, "Exploiting Visual and Inertial Sensor Fusion for Efficient Large Scale 3D Reconstruction", submitted to the International Journal of Computer Vision