Vision-based multi-sensor fusion for robust unmanned aerial vehicles autonomous navigation

Yuan, Shenghai

2019

Yuan, S. (2019). Vision-based multi-sensor fusion for robust unmanned aerial vehicles autonomous navigation. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/85185>

<https://doi.org/10.32657/10356/85185>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

**Vision-Based Multi-Sensor Fusion For
Robust Unmanned Aerial Vehicles
Autonomous Navigation**

YUAN SHENGHAI

SCHOOL OF ELECTRICAL & ELECTRONIC ENGINEERING

2019

Vision-Based Multi-Sensor Fusion For Robust Unmanned Aerial Vehicles Autonomous Navigation

YUAN SHENGHAI

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

11/1/2019

Yuan Shenghai

.....

.....

Date

YUAN SHENGHAI

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

11/1/2019

.....

Date

.....

Assoc Prof Wang Han

Authorship Attribution Statement

This thesis contains material from a patent filed to Intellectual Property Office of Singapore where I was the corresponding author.

Chapter 3 is filed as **S. Yuan, H. Wang, K. Wu, M. Esfahani**, “Stereo Depth Map Post-Processing Using Multiple Domain Analysis for Autonomous Navigation” *NTU ref:2018-281, Singapore Provisional Patent No: 10201810715U*.

The contributions of the co-authors are as follows:

- A/Prof Wang provided the initial project direction and edited the manuscript drafts.
- I prepared the manuscript drafts. A/Prof Wang revised the manuscript
- Data collection was done by me. K. Wu came out with the problem during an experiment. I analyzed the data and proposed the analytical solution.
- M. Esfahani assisted in surveying learning based semantic segmentation technique.

11/1/2019

Yuan Shenghai

.....

.....

Date

YUAN SHENGHAI

Acknowledgements

Firstly, I would like to express my gratitude to my supervisor Associate Professor Wang Han who invites me to this research group and gives me opportunities to work on some of the most exciting competitions and projects. Moreover, Prof. Wang guides me when I am frustrated and motivates my research constantly.

Besides, I would like to thanks my parents for believing me that I can study well while far away from my family. That decision makes me the man I am today.

Moreover, I would like to give my appreciation to Professor Xie Lihua, Professor Wang Danwei, Mr. Sunarto Quek and Mr. Hui Tingfat for the opportunities, projects, and competitions they provided. They helped me grow my knowledge and capabilities. Same gratitude goes to those few people whose names and belonging parties cannot be disclosed. They offer me with various pieces of equipment, advice and opportunities.

Last but not least, I would like to express my gratitude to work with Wu Keyu and Mahdi Abolfazli Esfahani on drone project and finish the research work in a short period. I want to give my thanks to Joshua Chao, Marcus from Avetics and Royce Nai from Singapore Airlines for helping me with the professional mechanical and pilot work. I want to thank Lim Kart-Leong, Zhang Handuo, Mou Wei and Zhou Lubing for their kind advice and after work fun times.

Its been my honor to get to know so many wonderful people.

Contents

Statement of Originality	iii
Supervisor Declaration Statement	iv
Authorship Attribution Statement	v
Acknowledgements	vi
Summary	xiii
List of Figures	xiii
List of Tables	xxiii
Symbols and Acronyms	xxv
1 Introduction	1
1.1 Background and Objectives	1
1.2 Motivation	6
1.3 Major Contributions	7
1.4 Outline of the Thesis	8
2 Literature Review	9
2.1 Notations and Mathematical Concept	10
2.1.0.1 Lie Algebra, Lie Group and Quaternions	10
Quaternions representation	12
2.1.0.2 Sensor Model and Calibrations	13
camera model and calibration	13
IMU model and calibration	17
Multi-sensors calibration	19
2.2 The State Estimation	22
2.2.1 A Quick Introduction	22
2.2.2 High Impact Breakthroughs	25
Earlier Works	25

	Recent Visual-Inertial Approaches	27
	Current Multi-Sensor Approaches	30
2.2.3	State Estimation Methodology Classification	33
2.2.3.1	Classify by Feature Type	33
	3D to 3D motion estimation	34
	2D to 2D motion estimation	36
	3D to 2D motion estimation	39
2.2.3.2	Classify by Tracking Method	43
	Indirect method	43
	Semi direct method	44
	Direct method	45
	Discussion on Tracking Method and Feature Den- sity	46
2.2.3.3	Classify by Solvers	47
	Filter based method	47
	Structure from motion based approach	51
	Biological inspired approach	56
2.2.3.4	Classify by Outputs	57
2.2.4	Current Trend - The Neural Network Based Approach . . .	58
2.2.4.1	Basic Building Elements and Operations	59
2.2.4.2	Current popular pipeline	61
2.2.4.3	Outstanding problem and challenges	62
2.3	Obstacles Detection	64
2.3.1	Introduction to obstacle detection problem	64
2.3.2	Obstacle detection physics	66
2.3.3	Radio spectrum	66
2.3.4	Inferred spectrum	68
2.3.5	Visible light spectrum	69
2.3.5.1	Monocular case	70
2.3.5.2	Stereo case	72
3	Multiple Domain Analysis based Stereo Depth Map Post-Processing for Autonomous Real-Time UAV Navigation	77
3.1	Introduction	78
3.2	The State of The Art	81
3.2.1	Feature Cost	82
3.2.1.1	Their Contributions	82
3.2.1.2	Their Limitation and Proposed Solution	83
3.2.2	Cost Aggregation	84
3.2.2.1	Their Contributions	85
3.2.2.2	Their Limitations	85
3.2.3	Stereo Matching	86
3.2.3.1	Their Contribution	87

3.2.3.2	Their Limitation and Solution	87
3.2.4	Post-Processing	87
3.3	Proposed Method Technical Description	90
3.3.1	Stereo Depth Map Post Processing using Multi Domain Analysis	90
3.3.2	Image Domain Analysis	92
	Understanding What Goes Wrong	92
	Observation Summary	94
	Formulate A Solution	94
3.3.3	Semantic Domain Analysis	99
3.3.4	Depth Domain Analysis	103
3.3.5	Corrected Depth Map Enhancement	109
3.4	Implementation and Experiment	113
	RGB Camera	113
	Greyscale Camera	114
3.5	Additional Result and Analysis	116
3.5.1	Qualitative Analysis	116
	Detailed Comparison of Objects	125
3.5.2	Quantitative Analysis	127
3.5.3	Result with the Real World Data	128
3.6	Limitations and future works	130
3.7	Conclusion	132
4	A Generalized Multi Exteroception Sensor Extrinsic Calibration Technique Using Orthogonal Scene	133
4.1	Introduction	134
4.2	The State of The Art	136
4.2.1	Structured Pattern Based Approach	137
	Earlier Formulation Using Chess Board Pattern	137
	Problems in earlier methods	138
	Improved Formulation Using Chess Board Pattern	138
	Outstanding issue	138
	Trihedral-based pattern	138
	Trihedral-based pattern problems	140
4.2.2	Ground Truth Based Approach	140
	The Advantage and Disadvantage	140
4.2.3	Advanced Tape Measured Approach	141
	Advantage and Disadvantage	142
4.2.4	Honorable Mentions	142
4.3	The Proposed Approach	143
4.3.1	Data Preparation	144
4.3.2	Geometric Constraints	146
4.3.3	Formulation	147

4.4	Implementation details	148
4.5	Experiment and Results	149
4.6	Limitations and future work	150
	Lack of Ground Truth	150
	Degenerative case	150
	Far Away Initial Guess	151
4.7	Conclusion	151
5	Real-Time UAV State Estimation using Multi-Sensor Fusion	153
5.1	Introduction and Motivation	154
5.2	The State of the Art	157
5.2.1	Visual-Inertial Fusion	158
5.2.1.1	Tightly Coupled models	159
	Advantage and Disadvantage with Tightly-Coupled models	162
5.2.1.2	Loosely Coupled models	163
	Advantage and disadvantage with Loosely Cou- pled models	163
5.2.2	Fusion-Based State Estimation	164
	Problem with State of the Art	165
5.3	Overview of Proposed Method	166
5.4	Visual-Inertial Based State Estimation Subsystem	168
5.4.0.1	IMU Modeling, Riemannian geometry and Newton- ian laws of Motion	168
	Newtonian Laws of Motion with Riemannian Ge- ometry	169
5.4.0.2	Visual-Inertial State Estimation	171
5.4.1	Analysis - the Problem, the Observation and the Solution . .	175
	Problem	175
	Possible Solution	176
5.4.2	Attempts to Solve the Problem	177
5.4.2.1	Failure Attempts - Hierarchical Graph Model . . .	177
5.4.2.2	Success Attempts - Variance Generation for Vision- based State Estimation	178
5.5	Dual LIDAR-Inertial-Based State Estimation Subsystem	181
5.5.1	Proposed Subsystem Description and Analysis	181
5.5.1.1	Line based Iterative Closest Point matching	182
5.5.1.2	Altitude Estimation	183
	One-D	183
	Two-D	184
	Three-D	184
5.5.1.3	Linear Kalman Fusion	187
5.5.2	Experiments	189
5.5.3	Qualitative and Quantitative Result	192

5.5.4	Existing Problem and Solution	196
	Variance Generation for LIDAR-based State Es- timation	197
5.6	Fusion-Based State Estimation Subsystem	198
5.6.1	Input State Analysis	198
	Inner Sensor Analysis	198
	Inter Sensor Analysis	199
5.6.2	Extended Kalman Filter based Fusion	201
5.6.3	Fusion Experiment Setup	204
5.7	Fusion-Based State Estimation Qualitative Results Analysis	206
5.7.0.1	Qualitative Results With Normal Lighting Condition	206
5.7.0.2	Qualitative Results With Tough Lighting Condition	210
5.8	The Fusion Results Quantitative Discussion For Complete Set Of Sensors	213
5.9	The Fusion Result Analysis For Partial Set Of Sensors	216
5.9.1	Performance In Normal Lighting Condition	217
5.9.1.1	IMU + Compass	217
5.9.1.2	Vision+IMU+Compass	218
5.9.1.3	LIDAR + IMU + Compass	219
5.9.2	Performance In Tough Lighting Condition	220
5.9.2.1	IMU + compass	220
5.9.2.2	Vision + IMU + Compass	221
5.9.2.3	LIDAR + IMU + Compass	221
5.10	The Fusion Result Analysis With Landmark Reference	222
	Normal lighting condition, Vision-based state es- timator and Mapping	223
	Normal lighting condition, LIDAR-based state es- timator and Mapping	223
	Normal lighting condition, Fusion-based state es- timator and Mapping	224
5.11	Limitation and Future Works	226
5.12	Conclusion	226
6	Conclusion and Future Work	229
6.1	Conclusion	229
6.2	Future Work	231
	List of Author's Awards, Patents, and Publications	233
	Bibliography	235
	Appendix A	269
	Jacobian and covariance for coplanar and orthog- onal constraint	270

Summary

The primary objective of this dissertation is to propose a robust multi-sensor fusion-based navigation system for lightweight Unmanned Aerial Vehicles(UAV) to navigate and perform the complex tasks in a challenging environment. In recent years, the demand for autonomous drone has grown exponentially. It is already a widespread practice to use the stereo camera in every drone for its low cost. However, stereo vision currently has significant problems such as accuracy and robustness issues. In this dissertation, a stereo post-processing method is proposed to handle the real world tough predicaments. The proposed stereo post-processing algorithm can fly with any robust state estimation system for autonomous flying. Although stereo can handle some flight case, there is always a top limit for vision system. The vision-based system will not work well with tough lighting condition such as overexposure in lunchtime and underexposure after dark. There is a need to add more active sensing sensors for a robust navigation system. Therefore, the later chapters introduce a multi-sensor fusion based system with Vision, LIDAR, IMU, and Compass. To use a multi-sensor system, there is a need to find the physical relationship between sensors. An exteroception sensors calibration technique is proposed to find those parameters with minimal calibration structure required. Each sensor state estimator works differently with a different process model, and they all have potential problems. The subprocess models have to be analyzed to come up with a quantitative measure of source variance and source error correlation. Then based on all previous observations and conclusions, a state only EKF fusion technique is proposed to merge all sub-state estimation observations. The result can achieve the best performance regarding robustness and accuracy when compared to other state of the art methods. All proposed works in this dissertation have been demonstrated with real-world data to verify the performance. Moreover, all the research have both qualitative and quantitative comparisons with the relevant studies.

List of Figures

1.1	Generalized autonomous UAV model.	2
1.2	Singapore legal flight area map. Any area with color means illegal fly zone[1]	3
1.3	Newly proposed Singapore UAV control law amendments. Operator permit cost around SGD400(2nd drone type)-SGD600(first drone). Activity permit cost SGD60(type II)-SGD75(type I) per flight activities. Image from CAAS proposal documents.	4
2.1	An Illustration of Pinhole Camera Model	14
2.2	An illustration of typical stereo camera calibration process. Image is from Zhang's work[2].	15
2.3	Wide angle stereo and its calibration process using very large chess-board. Image taken in open sea and building near lon/Lat: 1.315862/104.016163	16
2.4	Camera IMU calibration process using Kalibra. Image from Kalibra package[3]	18
2.5	An illustration of 3D LIDAR to 3D camera calibration. Image from Dhall et al. [4] work	21
2.6	An illustration of improvement in feature based method. MONOSLAM(left), PTAM(middle) and ORBSLAM2 (right). Image from the corresponding papers[5][6][7]	25
2.7	An illustration of new methods. DTAM(top), LSDSLAM(middle) and SVO2 (bottom). Image from corresponding papers[8][9][10]	26
2.8	Various visual inertial state estimation performance comparison[11]	28
2.9	IMU impact factor on VIO[12]. All the result is by using cheaper IMU. The left image is without loop closure where as the middle and right has loop closure. Image from in-house test result.	29
2.10	IMU based position estimation error for a static placed IMU. The error is due to drift and bias in the IMU modeling section. Data provided in the spec sheet by VectorNav - An IMU manufacturing company	30
2.11	Visual LIDAR Inertial Odometry[13]. Most complete solution when given enough weight budget and risk free environment	32
2.12	Iterative Closet Point matching. Image source[14]	34
2.13	An illustration of epipolar geometry.	36
2.14	P3P problem setup and simplified model.	40

2.15	An illustration of different tracking methods.	43
2.16	An illustration of direct tracking.	45
2.17	A classification of real time visual odometry based on number of feature and tracking method. Green denotes pure vision based method, blue denotes mult sensor fusion based method	46
2.18	Jacobian is sparse and only related to reference landmark i and j . .	54
2.19	The pipeline of deepVO[15].	61
2.20	VInet, illustration figure from[16]	62
2.21	Test on Training sequences and test on testing sequence using DeepVO with pre-trained weight on Kitti sequence id 01 02 05 08.	63
2.22	An illustration of drone obstacle detection. Image provided by DJI	64
2.23	An illustration of wrongly detect and classified object due to noise. Such noise often present in camera as shown in Figure 2.24. Image from Internet Source	65
2.24	An illustration of wrongly tracked pose due to lighting change. Image taken in 5th floor at longitude 1.291328 latitude 103.786790 facing north direction. Both shutter speed is set to be 15ms for both case. Note higher shutter speed leads to motion blur, top limit by this dissertation is 15ms	65
2.25	An illustration of frequency spectrum. Image from Wikipedia . . .	66
2.26	An illustration of Small size Synthetic aperture radar depth with top down view. Image from research work[17]	67
2.27	An illustration of stereo generated depth with top down view. Lesser PNSR noise when compared with SAR output. Image captured in-house	67
2.28	Typical 1D 2D 3D lidar from left to right and their applications. Sensor image taken from Google	68
2.29	Typical outdoor performance by IR based RGBD camera. Green denotes missing value. Image from research work[18]	69
2.30	Comparison between VOXBLOX(left) and OCTOMAP(right) map representation. Image taken from corresponding paper[19][20] . . .	72
2.31	First row(left to right): Input, Input, Second row(left to right): Real time stereo output, DepthComp[21]	75
3.1	Left: Uber L5 Navigation fatal accident live footage[22], Right: Tesla L3 Navigation fatal accident analysis[23].	78
3.2	The problem of current stereo based drone navigation	79
3.3	Proposed method compared with most recent work[21]. Left is the input census transform stereo raw depth[24].	80
3.4	Biological model turns into Mathematical model	81
3.5	Stereo vision problem in a nutshell. Image from dataset[25]	81
3.6	The evolution of finding raw cost. Image source[26][27]	82
3.7	The State of The Art Learning Based Stereo. Image source[28] . .	83
3.8	Current State of The Art Cost Aggregation Basis. Image source[2]	84
3.9	State of The Art Cross-Scale Cost Aggregation Basis. Image source[29]	85

3.10 Detailed Method for Semi Global Stereo Matching	86
3.11 One of the Common Approach For Stereo Post Processing Using Left Right Depth Consistency Check	88
3.12 Result of Left Right Consistency Check Fails with System Level Error. The error is still present after the check	89
3.13 One of the Current State of the Art of in Stereo Post Processing. Image source[21]	89
3.14 Result form the benchmark[21]. Fails with strong error pattern . .	89
3.15 An overview of enhanced depth map post processing system flow . .	90
3.16 Overview of PPMDA. Image is from the KITTI dataset 2011 09 26 drive 0035 sync sequence	91
3.17 An illustration of stereo error contributed by saturated intensity. . .	92
3.18 An illustration of stereo error contributed by lack of gradient. The green denotes the feature rich area with good depth. The redbox denotes the featureless area which leads to error in depth	93
3.19 An illustration of stereo error contributed by lack of horizontal gra- dient and total bank area. The green denotes the feature rich area with good depth. The redbox denotes the high horizontal gradient reponse area which leads to error in depth	93
3.20 Equation association in each function block.	94
3.21 An illustration of using only E_d term to find areas that are prone to depth error.	95
3.22 An illustration of using only E_d term to find areas that are prone to depth error on a less contrasted image. Depth is from modified census transform pattern with binary SGM	96
3.23 An illustration of using only E_g to find long horizontal gradient patches that are prone to errors.	97
3.24 Comparison of only using E_d and only use E_g	98
3.25 γ value range vs Environment. In the right image denotes the empty scene at lon/lat:1.338485/103.772564 where γ is typically set to be zero , the left scene found at lon/lat:1.291278/103.786854 where γ is best to be selected at one	99
3.26 ICNet Model [30]	100
3.27 ICNet Model representation	100
3.28 Semantic Segmentation based output	101
3.29 Input depth from GPU census transform(top), Optional step of road surface gradient filling based on minimum fit RECT from semantic label(bottom).	102
3.30 An illustration of iterative wrong depth map removal at iteration 0.	103
3.31 An illustration of iterative wrong depth map removal at iteration 1.	104
3.32 An illustration of iterative wrong depth map removal at iteration 2.	105
3.33 An illustration of iterative wrong depth map removal at iteration 3.	106
3.34 Cleared depth map over iterations	108
3.35 Reference patch area and target area generation	109

3.36	Finding the optimal the depth in each iteration	111
3.37	Input image(top), Critical error corrected depth(middle left),minor error corrected depth(middle right), Depth image enhancement on only critical error corrected depth(bottom left), Depth image enhancement on only all error corrected depth(bottom right)	112
3.38	First row: Input left, Input right, Second row:Convolutional neural network feature with SGM depth left, Convolutional neural network feature with SGM depth right, Third row: ICNet output semantic label, Left right consistency based method[28], Last row: DepthComp[21], Proposed method.	112
3.39	The first row is input image left and right, Second row is processed depth and semantic label. Last row DepthComp [21], and proposed solution. Testing image is from KITTI dataset 2011 09 26 drive 0035 sync sequence	115
3.40	Top and side view of projected map	115
3.41	Additional result 1 for enhanced post processing	117
3.42	Additional result 2 for enhanced post processing	118
3.43	Additional result 3 for enhanced post processing	119
3.44	Additional result 4 for enhanced post processing	120
3.45	Additional result 5 for enhanced post processing	121
3.46	Additional result 6 for enhanced post processing	122
3.47	Additional result 7 for enhanced post processing	123
3.48	Additional result 8 for enhanced post processing	124
3.49	Additional object comparison 1. Image intensity increased by 50%	125
3.50	Additional object comparison 2. This small streaking effect ranges from 0 to 255. May cause falsely generated obstacle.	125
3.51	Additional object comparison 3. The horizontal streaking stripe are commonly seen in older 2D scan-line based optimization method.	126
3.52	Additional object comparison 4. The streaking stripe effect will create sudden close obstacle. The pole size is remained in the proposed method where previous work enlarge the obstacle size	126
3.53	Lack of ground truth data from Kitti Dataset[31]. The area 1 to 6 suppose to be compared but lack quantitative data for evaluation.	127
3.54	Quantitative analysis (error in pixels) of error in object level in sub-pixel accuracy. The actual error for proposed should be lower when consider the FN to be occluded area.	127
3.55	Stereo Post Processing for a stable flight at roof. Stereo baseline is about 40cm. The flow chart show case the effectiveness of iterative error removal.	128
3.56	Stereo output in autonomous car test. Proposed method outperform current ST-K autonomous car benchmark. Image captured from Singapore Autonomous car testing facility.	129

3.57	Vision based navigation fail during night. Image taken in 5th floor at latitude 1.291328 longitude 103.786790 facing north direction. Both shutter speed is set to be 15ms for both case.	130
3.58	Vision based navigation fail at dark and the featureless area. Image taken in controlled civilian private airport area at latitude 1.405525 longitude 103.865645 facing East-South direction. Both shutter speed is set to be 15ms for both case.	131
3.59	Vision based navigation fail at 0600 to 1030 and 1530 to 1800.	131
4.1	Animal Sensing Model, Red arrow denotes where the observation is taken. Illustration images come from [32] [33] [31]	134
4.2	Chess board based calibration by [34] (left), and Vasconcelos et al. [35] (middle) and Zhou and Deng [36] (right). Images obtained from each paper	137
4.3	Trihedrons based calibration by [34] (2015). Image from their papers	139
4.4	Failure Scenes for Trihedrons based method. The first assumption for Trihedrons based calibration is stable tracking of corner scene. The edges in the failure case cannot be tracked robustly due to multiple detection and featureless corners.	139
4.5	Ground truth based approach by [37] (2015). Images obtained from their paper and PCL library for illustrative purpose.	140
4.6	Tape Measured Approach by [38] (2017)	141
4.7	Other approaches by [39] (2014), [40] (2018) and [41] (2017)	142
4.8	Assumption and Objective of proposed method. Image source[42]	143
4.9	The overview of the proposed idea to calibrate multiple type of sensors.	144
4.10	Mapping with proposed methods	149
5.1	Various reports of crashes during Autopilot after more than 20 years of autonomous car research. Image Source[43]	154
5.2	Various application for drones, and technology between civilian and military are interchangeable. Image source[44]	155
5.3	Biologically inspired navigation methods. Various Illustrative figures obtained from [32] [33] [31]	156
5.4	The autonomous drone based the solution similar to the one from ETH research group[45]. Image Source[46]	157
5.5	Different Paradigms of visual inertial integration. Note, Implementation cost referring to both implementing the current algorithm with given visual inertial sensors and additional implementation cost for adding more sensors. Robustness problem means the divergence problem and degenerative(singular) case which leads to a runtime exception. Valid variance from each source is needed to loosely coupled method or system diverge fast. Green denotes pros and Red denotes Cons	158
5.6	Tightly Coupled models Problem illustration	159

5.7	The process flow for Tightly coupled visual-inertial system. Image from [12]	159
5.8	Tightly Coupled models top level problem formulation	160
5.9	The sensor models formulations and derivatives forms the second layer of the pyramid model	161
5.10	Various implementation details forms the bottom layer of the pyramid model	162
5.11	Loosely coupled model example: PTAM + IMU with EKF[47] (left, Image from paper[47]). In-house flight tests(right).	163
5.12	The state of the art Visual Inertial LIDAR Fusion-Based State Estimation drone. The drone crashes with sudden controllless drift to one direction. The Pilot force the drone down to avoid hurting staffs. Base on similar understanding and flight experience, the sudden drift only happens with hanging state estimator. The mathematical formula follows the one defined in the review section. The drone image from [48]	164
5.13	Challenging environment in Singapore. Top images from their paper and their testing facility in Guardian Centers training facility in Perry, Georgia, US. Bottom images from Singapore free fly zones and Murai Urban Training Facility. All images are obtained legally by different means	165
5.14	Overview of the proposed Visual LIDAR Inertial Compass fusion system.	166
5.15	Chapter Content Breakdown.	167
5.16	IMU based position estimation error for a static placed IMU. The error is due to drift and bias in the IMU modeling section. Data provided by VectorNav - A IMU manufacturing company	169
5.17	VINSMONO model breakdown. (1)Feature detection and tracking (2)IMU preintegration (3)Calibration and Initialization (4) Runtime nonlinear Optimization (5) Loop Closure	172
5.18	Problematic scenes with vision-based state estimator	175
5.19	Possible new improvement	176
5.20	Robust tracking from Event Camera. Image from ETH group. . . .	176
5.21	Failure attempt using Hierarchical Graph Model based approach. .	177
5.22	When lighting condition changes, Vision-based state estimator completely collapses.	178
5.23	Error correlation between a good result and bad result are similar which means more iteration will not reduce the error and will not improve the result.	179
5.24	A better indication for good and bad Visual-Inertial state estimator performance.	180
5.25	The overview of proposed Dual LIDAR-Inertial-based Navigation Subsystem	182

5.26	Proposed LIDAR based autonomous drone system with pilot hands free. The image was taken in controlled civilian private airport area at longitude 1.405525 latitude 103.865645 facing East-South direction.	182
5.27	Testing ICP with and without line information and compared with vision based state estimation under relative good lighting condition	183
5.28	The raw ground line model detected is shown in red	184
5.29	The drone used in this work which mounted with 3D sensor pointing downwards and on-board processing unit. Image taken at Tengah Airbase on February 27, 2018 by a journalist. Declassified for publication use	185
5.30	The ground model being detected. The colored point cloud is ground model. White denotes outliers. Bottom left and right are measurement from thermal and ZED image. Image taken at 1.291366, 103.786834 heading North East	185
5.31	Three generation of development for LIDAR based drone navigation	190
5.32	Autonomous UAV control and navigation model	191
5.33	Dense point cloud reconstruction after flight inside ship	194
5.34	LIDAR generally performs better when compare with Visual-Inertial based state estimation.	195
5.35	Featureless scene VO lost. Red represent the Vision-based state estimator and purple denotes LIDAR-based state estimator	196
5.36	Dual LIDAR-Inertial based state estimation variance grows when dealing with large planer scene.	197
5.37	Possible wrong path cases.	199
5.38	Improved result by remove error propagation.	200
5.39	Fusion process overview	202
5.40	Normal lighting condition variance plot. Variance in red means vision-based state estimation variance, Variance in blue means LIDAR -based state estimation variance. The first peak happens at the wall, the Second peak happens in the empty carpark. The variance is raw value without normalization	209
5.41	Tough lighting condition variance plot before normalization. Variance in red denotes visual-inertial state estimation variance, variance in blue means LIDAR state estimation variance	212
5.42	Performance in the normal lighting condition. Both Visual Inertial-based state estimation and LIDAR Inertial-based state estimation is still acceptable. The fusion-based state estimation is more accurate. No valid GPS were captured during this sequence. Ground truth are measured after experiment	213
5.43	When dealing with most tough case, proposed method still can generate accurate output where GPS is drift because of multipath from building and IMU drift	214
5.44	Fusion with different set of sensors in the normal lighting condition	216
5.45	IMU+Compass fusion in normal lighting condition. The drift pattern is similar to lost GPS type of drift	217

5.46	State estimation result from fusing Visual-Inertial and compass only (Black) on normal lighting condition	218
5.47	State estimation result from fusing LIDAR-Inertial and compass (Black) on normal lighting condition. The behavior is similar to animal disoriented movement pattern.	219
5.48	Fusion-based state estimation result from other sensor combinations on tough lighting condition	220
5.49	An illustration of expected detected patterns over x-y plane. The expected obstacles(Green) are drawn by hand	222
5.50	Mapped scene with vision-based state estimator on a good day. . .	223
5.51	Mapped scene with LIDAR-based state estimator on a good day. . .	224
5.52	Mapped scene with Fusion-based state estimator on a good day. . .	225
5.53	Variance for both LIDAR-based state estimator and visual-based state estimator are high. The image sequence is taken with high wind speed, plenty of dynamic moving grass and no LIDAR feature around	226

List of Tables

3.1	PC Task Assignment for the RGB Camera	113
3.2	Object level subpixel accuracy comparison	128
5.1	Performance Comparisons between OKVIS and VINSMONO using custom set of parameter	171
5.2	Altitude variance at a different height when the drone has horizontal movement after the control loop compensation. * denotes data gathered from Gen3 platform where others are measured from Gen2 in Figure 5.31. All data captured from longitude: 1.291376 latitude: 103.786854 with the camera placed on the roof and facing towards the sky. * taken at a later time at the approximately same location	186
5.3	Map created in LIDAR based autonomous drone navigation. Ship GPS position unknown, Panamax Class, Front Left Deck	193
5.4	Performance Comparisons between LIDAR based approach and vision based approach	195
5.5	The feedback state variable configuration. 1 denotes present, 0 denotes not present and * denotes differentiated and then force to zero parameters	205
5.6	Vision-based state estimator(red) vs LIDAR-based state estimator(blue) vs Fusion-based state estimator(Yellow). The image is taken from latitudes: 1.293502 longitude: 103.788308 at 1442 noon. Target equivalent GPS location latitudes 1.2934832484 longitudes 103.787625122. Only initial GPS was used in the calculation. Wind speed is less than 5m/s. Green X denotes the starting and ending location	208
5.7	Vision-based state estimation(red) vs LIDAR-based state estimation(blue) vs Fusion-based state estimation(Yellow). The image is taken from latitude:1.293502 longitude:103.788308 after rain and after 1800. Target equivalent GPS location latitude:1.2934832484 longitude: 103.787625122. Only initial GPS was used in the calculation. Wind speed is less than 5m/s. Green X denotes the starting and ending location	212
5.8	Performance Comparisons Among LIDAR-based, Vision-based and Fusion-based Methods.	215
5.9	Accuracy Comparison Among LIDAR-based, Vision-based and Fusion-based Methods	215

Symbols and Acronyms

Symbols

Unless stated otherwise

$\ \cdot\ $	The squared norm of a vector or matrix in Euclidean space
K	Camera Projection Matrix
R	Rotation Matrix $R \in SO(3)$
t	Translation vector
T	Transformation Matrix $T \in SE(3)$
∇f	the gradient vector
$\mathcal{C}^k$	the function with continuous partial derivatives up to k orders
$\triangle I_N$	the first order differential image in N direction
$x_{i,k}$	the i -th component of a vector x at time k
$x_{k k-1}$	state vector from $k-1$ to k
$\mathcal{N}_i$	the normal distribution
$\hat{C}$	estimated value of C

Acronyms

UAV	Unmanned Aerial Vehicle
SO	Special orthogonal Group
SE	Special Euclidean Group
GPS	Global Positioning System
GNSS	Global Navigation Satellite System
LIDAR	Light Detection and Ranging
SLAM	Simultaneous Localization And Mapping
VO	Visual Odometry

IMU	Inertial Measurement Unit
EKF	Extended Kalman Filter
UKF	Unscented Kalman Filter
CNN	Convolutional Neural Network
GCS	Ground Control Station
CAAS	Civil Aviation Authority of Singapore
GPU	Graphics Processing Unit
VICOM	Indoor 3D positioning sensing system
RANSAC	RANdom SAMple Consensus
LSTM	Long short-term memory
RNN	Recurrent neural network
SAR	Synthetic aperture radar
TSDF	Truncated Signed Distance Fields
ESDF	Euclidean Signed Distance Field
DOF	Degree of freedom
DWA	Dynamic Window Approach
RRT	Rapidly-exploring Random Tree
FCU	Flight Controller Unit

Chapter 1

Introduction

1.1 Background and Objectives

In recent years, Unmanned Aerial Vehicle(UAV) is widely used for tasks in the military, agriculture, industrial and personal recreation. Currently, Majority of UAV are in the form of a remotely piloted aerial system. Such a system is known as the man in the loop system. A pilot is needed to control the drones for tasks such as killing the terrorists, building inspection, worker supervision, load carry. Typically drone is controlled by the Ground Control Station(GCS). GCS can be as simple as a joystick or a smartphone. More advanced GCS can control the drone with advanced autonomous flight modes, obtain live telemetry data and carry specific task like avoiding obstacles and dropping payloads.

With the increasing uses, comes the growing demand for autonomous functions. For large drones, they usually stay at high altitudes and have no risk of hitting trees or buildings. Only small drones can move closer to trees and installations to do daily tasks.but small drones are limited by the amount of payload that they can carry. An autonomous car only has to get people around in 2D environments while UAVs have to deal with a more complex 3D environment to navigate around using a lower amount of computational power and sensory information. The challenge is colossal. Currently, Majority of UAVs come with manual flight mode and free space GPS-based navigation mode. With the swipe of a finger, the drone can fly according to the preset GPS path. If there is something in the way, the UAV will

not hesitate and crash into it. More often in the urban environment, GPS is a luxury service. In between buildings, the GNSS satellite count can be lower than two. Empirically, It usually need at least seven satellites for a stable positioning. Moreover, the inbuilt GPS-inertial system can cause the drone to drift and crash into the building when there is multipath or system signal jamming. Even with

FIGURE 1.1: Generalized autonomous UAV model.

various limitations, the drone market has grown exponentially into a multi-billion dollar industry. In the civilian market, Ardrone, DJI, Yihang and 3DR dominate and they have already begun to develop intelligent flight capability. To allow the drone to run fully autonomously in the urban area, the drone has to be able to detect obstacles and know its flight path as can be seen in Figure 1.1.

To allow the non-expert user to interact with the drone and commercialize the drone, there usually need modules such as global optimization module, Mission planning module, and Payload control module. The green boxes are useful to have but not essential. For semi-intelligent drones which can plan a path, avoid obstacle and go round occupied fields, the Path planning and Mapping modules are needed. Therefore, the purple boxes are commonly seen in most research drones. The drones need to be mounted with various sensors to provide localization and mapping information so that the control can be carried out. Therefore those four red boxes are the most critical components in autonomous drones. In order to achieve stable flights, the drone has to equip large amount of sensors. The more sensor information the drone has, the more accurate/robust decisions the drone can make. The core component sounds like a Simultaneous Localization And Mapping(SLAM) problem, but SLAM alone is not robust enough for the autonomous drone. Most visual SLAMs produce poor 3D maps, and almost all VO or SLAM can drift very quickly in the air without any notification. Such visual SLAM drift pattern is worse than GPS drift pattern. Therefore, robust obstacle detection and

robust state estimation are crucial. Only with robustness, can the drone then maximise its full potential for doing tasks such as delivery or inspection.

Drones come with various types and sizes in the market. There are thousands of commercial drone products and hundreds of military drone products. Many classification schemes have been proposed to categorize the drones. Based on the drone categorization, different governments propose different regulations. However, there are no universal way of classification. Usually, drones are classified by its intended mission, flight range, flight duration, and maximum takeoff weight according to van Blyenburgh [49]. Recently, Dalamagkidis and Piegel [50] correlate maximum take-off weight to a set of naming conventions. Most Micro unmanned Aerial Vehicle (MAV) is referring to this standard Dalamagkidis and Piegel [50]. And most regulations follows Federal Aviation Regulation of the United States.

In this dissertation, the experiment is done using drone from micro UAV to mini UAV weight class. In the Singapore context, the operation of UAVs is heavily regulated. Any violation means \$ 20,000 in fines and one year of jail time. Any drones which are below 7kg takeoff weight, fly below 200ft and not in controlled areas require no permit and pilot license. Anything above those regulation require both flight permit, pilot license and additional approval from the air force to operate. Flight permit is needed for each flight, and they cost money and huge manpower to apply. NTU is entirely covered inside the no-fly zone as can be seen from Figure 1.2.

FIGURE 1.2: Singapore legal flight area map. Any area with color means illegal fly zone[1]

Weight of UA	For recreational/research purpose		For non-recreational/non-research purpose
7kg and below	[Proposed] Successful completion of online training programme, if flying UA of total mass of more than 1.5kg	<ul style="list-style-type: none"> • Class 2 Activity Permit, if flown outdoors: <ul style="list-style-type: none"> – in a restricted or danger area, or – within 5km of an aerodrome/military airbase regardless of operating height, or – above 200 feet beyond 5km of an aerodrome/military airbase 	<ul style="list-style-type: none"> • Operator Permit • Class 1 Activity Permit [Proposed] UA Pilot Licence
Above 7kg, but not more than 25kg	[Proposed] UA Pilot Licence [Removed] Operator Permit
[Proposed] Above 25 kg	<ul style="list-style-type: none"> • Class 1 Activity Permit [Proposed] UA Type Certificate/Certificate of Airworthiness [Proposed] UA Operator Certificate [Proposed] UA Pilot Licence [Removed] Operator Permit 		

Source: CIVIL AVIATION AUTHORITY OF SINGAPORE STRAITS TIMES GRAPHICS

FIGURE 1.3: Newly proposed Singapore UAV control law amendments. Operator permit cost around SGD400(2nd drone type)-SGD600(first drone). Activity permit cost SGD60(type II)-SGD75(type I) per flight activities. Image from CAAS proposal documents.

But those were the old rules in 2015. The new rules as shown in figure 1.3 are more strict. Under these rules, the drone testing is limited to small size drones, flying at low altitude and limited areas. The small size drone also means fewer sensors can be used and high GPS interference when flying low. The drone also has to be configurable and easy to build/buy/replace. All those requirements limit the choice of the drone to either a few custom-made frames or DJI M100/M210. In this thesis, both custom frame and M210 is used. The custom frame can provide various configuration possibilities at the cost of integration manpower whereas M210 offers more in robustness and can be RMA exchanged if crashed by internal control flaw.

In this thesis, the aim is to come up with solutions that allow drones to carry lightweight sensors to fly in urban environments at anytime fully autonomously. It translates into a few requirements and assumptions such as (1) Lightweight, (2) GPSless, (3) No photo consistency assumption and (4) dynamic scenes.

With those requirements and assumptions in mind, this dissertation can be broken down into a few parts. The first part address the issue of accurate mapping and dynamic navigation using stereo vision. Then it addresses the robust navigation problem by fusing Vision, LIDAR, IMU, and Compass into a single robust state estimator. To add more sensors in a system, a calibration algorithm is proposed to find the physical relationship between different types of sensors. For multi-sensor fusion, a series of analysis on the current state of the art is done to come up with covariance measures for fusion. At last, the different subsets of the fusion system is demonstrated to provide additional understanding for the multi-sensor fusion system.

All of the research done here complies with the regulation of Singapore CAAS Unmanned Aircraft Systems Rules. Part of the related work done here is already tested with actual operations in both civilian and military installations.

1.2 Motivation

The autonomous drone navigation system in unknown and GPS weak/denied environment is always an exciting topic for both military and civilian uses. The demand for a lightweight drone system to operate robustly in an autonomous manner is high. The weight and autonomous system capabilities are the conflicting factors as a fully autonomous system needs heavier loads. Heavier loads also lead to higher risks for crashing and losses. The problem of how to make a lightweight drone autonomous depends on what sensors and what algorithm is used. It's all about understanding the environment next to the drone. One of the most common approaches is to use vision to control the drone autonomously as the camera is virtually weightless. However, a camera can fail very quickly to light changes or low light and featureless areas. Infrared(IR) depth camera and LIDAR is a robust way of an active sensing method. Due to the popular demand for outdoor tasks, IR depth camera is rarely used as they do not work well under the sunlight. LIDAR is able to work under the sunshine, but long-range LIDAR is generally much more cumbersome. For 16 line Velodyne, the sensor and cable would weigh 1kg payload with large size requirement. There is also a need to have a special powering unit just for those LIDARs and PC. When considering the local regulations, i.e. flight time and risk of crashes, it is clear that putting a Velodyne LIDAR on the unlicensed lightweight drone is impossible. For a licensed drone, every flight requires a sum of fees. And every year, pilot has to renew the pilot license. For practical use in Singapore, short-range single scan line LIDAR is used for estimating motion and detecting the obstacle. Even with single line LIDAR, things can still go wrong. A single line LIDAR can crash on a table as the table surface will not be detected by 2D LIDAR. The drone thinks it is safe to navigate. To solve that, we use multiple lightweight LIDAR together to achieve better results. However, during the test, another problem reveals itself as lightweight LIDAR is usually short range. For longwall surface, position tracking is increasingly unreliable. Then based on the weaknesses of both vision and LIDAR, a fusion technique is proposed to make the drone perform robustly in the urban area.

1.3 Major Contributions

The main contributions can be stated as follows:

- *First:* A robust stereo post-processing method is proposed to handle real-world vision based stereo detection problem. In the real world, the depth map is full of false positives that prevent autonomous navigation to run properly. In the proposed post-processing framework, possible depth error is found by concatenating the analysis from image domain, depth domain, and knowledge domain. Thousands of stereo depth images from in-house and public datasets are studied to determine the characteristics of the poorly produced depth image. By study the characteristic of the poorly constructed depth image, a reasonable process model is proposed from each domain to resolve the error. Then based on the error-free depth map, enhancement is done to fill the missing part.
- *Second:* Vision-based drone still suffers from low light and featureless problems. To solve that, multiple other types of sensors such as LIDAR has to be added. The first thing to use multi-sensor is to calibrate all the sensor together to figure out their external relationship. Usually, this is done with the chessboard. In this dissertation, a calibration method is proposed to find extrinsic parameters among multiple exteroception sensors such as stereo, RGBD, and LIDAR. The proposed calibration method does not need a special calibration pattern which makes it ideal for the field deployment.
- *Third:* Currently, Currently, a lightweight autonomous UAV suffers from robustness and accuracy issues. . Both vision based method and LIDAR-based method fails from time to time. When lighting condition is not good, vision fails. When dealing with featureless 3D scenes, LIDAR fails. In this dissertation, a multi-sensor fusion based approach is proposed to address robustness and accuracy issues. From the failure case in the optimization function, a covariance or inverse confidence measure for each state estimations can be achieved. Based on the analysis result, a unique lightweight visual-LIDAR-IMU-Compass-based state estimation is proposed by using an EKF based multi-sensor fusion. The result is most accurate when compared with any other fusion based work.

1.4 Outline of the Thesis

This dissertation primarily works on the front-end of a lightweight autonomous UAV navigation problem. The front-end is the three input red boxes in Figure 1.1. The thesis is divided into seven chapters.

Chapter 1 presents the introduction to this thesis and outlines the contribution

Chapter 2 is the literature review chapter which covers primarily on the mathematical and conceptual idea of visual state estimation and obstacle detection.

Chapter 3 presents robust stereo vision based navigation using enhanced stereo depth map post-processing technique from multiple domain Analysis. This section addresses the issue of how to process a depth image so that it can be used directly in autonomous navigation tasks.

Chapter 4 presents a calibration method for calibrating multiple exteroception sensors with minimal external calibration structure.

Chapter 5 presents a novel lightweight Visual-LIDAR-Inertial-Compass fusion method to address the robustness and accuracy factor in UAV state estimation problem.

Chapter 6 provides the conclusion of current work and shows the possible future work direction for the lightweight drone navigation.

Chapter 2

Literature Review

There are about countless related works in this area, but only a few hundred of high impact works or closely related works are included. In this dissertation, LIDAR related studies are considered and reviewed under specialized sparse 3D vision sensing system. There will be no separate section for LIDAR-based studies.

This review section focuses on front-end sensing technique, and only generalized formulation is included. The front-end refers to the input red blocks in Figure 1.1. For any high impact work, there are a series of following studies which include minor modifications on top of the high impact works. It is impossible to include all of them. Therefore, Only the most recent and popular methods are discussed in detail. The detailed discussions are inserted at the state of the art section of each chapter for easier reader understanding and problem correlation.

The path planning, navigation, control, multi-drone collaboration, visual detection/classification, drone design and manufacturing, and drone flight dynamics will not be included. In the proposed method section, those additional works used for autonomous drone flight test will be briefly introduced.

The mapping part is the output from the front-end component results. It usually connected with obstacle detection and path planning. Therefore it will be briefly described with more recent works.

2.1 Notations and Mathematical Concept

Several notations and math backgrounds need to be covered in this section. Those equations, method, and ideas form the backbone of this dissertation.

2.1.0.1 Lie Algebra, Lie Group and Quaternions

A pose in 3D space can be represented as displacement in 3D $[x, y, z]$ and rotation in 3D defined by means of a specific case of Euler angle: yaw(ϕ), pitch(χ) and roll(φ). In most cases, 6 degrees of freedom pose is denoted as ξ and is defined by

$$\xi = [x, y, z, \phi, \chi, \varphi] \quad (2.1)$$

Where $\xi \in se(3)$ which $se(3)$ is the special Euclidean group in the Lie-algebra. In this form, it is clear what value represents but difficult in calculating the concatenation of multiple transformations. Therefore an alternate form was proposed. A 3D rigid body transform $T \in SE(3)$ in corresponding special Euclidean group in Lie-group denotes the rotation and translation in 3D and is defined as

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \quad T \in SE(3) \subset R^{4 \times 4} \quad (2.2)$$

Where $R \in SO(3)$ is a 3X3 rotation matrix and $SO(3)$ denotes the special orthogonal group. And t is 3x1 translational vector $t \in R^3$. The T^{-1} can be found by

$$T^{-1} = \begin{bmatrix} R^T & -R^T t \\ 0 & 1 \end{bmatrix} \quad (2.3)$$

Elements in $se(3)$ and $SE(3)$ can be mapped by exponential mapping.

$$T = exp_{se(3)}(\xi) \quad (2.4)$$

The mapping has a closed form solution. Let $\xi = [u, w]^T = [x, y, z, \phi, \chi, \varphi]^T$ denotes translation and rotation in $se(3)$. The u, w is for simpler formulation. The

exponential mapping can be represent by

$$u, w \in R^3 \quad (2.5)$$

$$\Theta = \sqrt{w^T w} \quad (2.6)$$

$$\exp\left(\begin{bmatrix} u \\ w \end{bmatrix}\right) = \exp\left(\begin{bmatrix} w_x & u \\ 0 & 0 \end{bmatrix}\right) \quad (2.7)$$

$$= \begin{bmatrix} \exp(w_x) & Vu \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = T, T \in SE(3) \quad (2.8)$$

$$\exp(w_x) = V = 1 + \left(\frac{I - \cos\Theta}{\Theta^2}\right)w_x + \left(\frac{\Theta - \sin\Theta}{\Theta^3}\right)w_x^2 \quad (2.9)$$

$$w_x = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \quad (2.10)$$

The inverse of exponential mapping is the logarithm mapping where

$$w = \log(R) \quad (2.11)$$

$$\cos\Theta = \frac{1}{2}(tr(R) - 1) \quad (2.12)$$

$$\ln(R) = \frac{\Theta}{2\sin\Theta}(R - R^T) \quad (2.13)$$

$$u = V^{-1}t \quad (2.14)$$

$$V^{-1} = I - \frac{1}{2}w_x + \frac{1}{\Theta^2}\left(1 - \frac{\Theta\sin\Theta}{2(1 - \cos\Theta)}\right)w_x^2 \quad (2.15)$$

where w can be computed by getting element in the off diagonal element of $\ln(R)$. And $\Theta = \sqrt{w^T w}$. This formulation is commonly known as the Rodrigues rotation formula, named after Olinde Rodrigues. The transformation of pose i to pose j can be written as ξ_{ij} . The transformation of pose j to pose k can be written as ξ_{jk} . Therefore the pose concatenation can be denoted as

$$\xi_{ik} = \xi_{ij} \circ \xi_{jk} = \log_{SE(3)}(\exp_{se(3)}(\xi_{ij})\exp_{se(3)}(\xi_{jk})) \iff T_{ik} = T_{ij}T_{jk} \quad (2.16)$$

In the majority of cases, pose in $SE(3)$ is used for calculation of pose concatenation. The above transformation is valid only when the scale is fixed. But in many monocular state estimation cases, there is no fixed scale and the scale is based on initial translational movement baseline[6][51][52][53]. Therefore a 3D similarity

transformation is defined as $S \in Sim(3)$ denotes 3D transformation with a scale factor which is defined as follows

$$S = \begin{bmatrix} sR & t \\ 0 & 1 \end{bmatrix} \quad (2.17)$$

where the letter s denotes the scale factor. It holds similar form as 3D rigid body transformation, but it is usually used when merging a consistent global map with hundreds of scale-free local map. The corresponding Lie-algebra for the similarity transformation $sim(3)$ has seven degrees of freedom with the introduction of a new scale parameter s . The exponential and logarithmic mapping is discussed in detail at Strasdat et al. [54].

For a practical sample of this representation is to generate odometry information from GPS and Inertial measurement. Let x_{lon} , y_{lat} , z_{alt} be longitude, latitude and altitude measured from GPS. $[\phi, \chi, \varphi]$ denotes robot yaw pitch and roll. t represents time. c and s denotes *sine* and *cos*. $[x, y, z]$ are in GPS coordinate. An odometry source $[x_{odom}, y_{odom}, z_{odom}, 1]^T$ at current time can be found from previous measurement $[x_{lon}^{t-1}, y_{lat}^{t-1}, z_{alt}^{t-1}]^T$ by

$$\begin{bmatrix} x_{odom} \\ y_{odom} \\ z_{odom} \\ 1 \end{bmatrix} = \begin{bmatrix} c\chi c\phi & c\phi s\varphi s\chi - c\varphi s\phi & c\varphi c\phi s\chi + s\chi s\phi & x_{lon}^{t-1} \\ c\chi s\phi & c\varphi c\phi + s\phi s\chi s\varphi & -c\phi s\varphi + c\phi s\chi s\varphi & y_{lat}^{t-1} \\ -s\chi & c\chi s\varphi & c\varphi c\chi & z_{alt}^{t-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_{lon}^t \\ y_{lat}^t \\ z_{alt}^t \\ 1 \end{bmatrix}$$

Quaternions representation In many studies, researchers use the quaternions representation for orientation. It was first invented by Hamilton in 1843 in Dublin. Quaternion is represented by

$$q = xi + yj + zk + w \quad (2.18)$$

where $[x, y, z]$ is defined by $[\sin\frac{\theta}{2}u_x, \sin\frac{\theta}{2}u_y, \sin\frac{\theta}{2}u_z]$. $[u_x, u_y, u_z]$ are the unit rotation axis. The angle of rotation w is defined by $w = \cos\frac{\theta}{2}$. The i, j, k denotes imaginary number where $i^2 = j^2 = k^2 = -1$. the concatenation of quaternion is the multiplication in terms of standard vector operations with the first rotation on

the right hand side($SE(4)$ is left)

$$q_t = q_2 q_1 \quad (2.19)$$

Let v be a 3-dimensional vector and q be unit quaternion. To rotate a vector v by q , it can be represented by

$$v_q = q v q^{-1} \quad (2.20)$$

To rotate the vector by q and then rotate by q' the result can be found by

$$(v_q)q' = q' q v q^{-1} q'^{-1} \quad (2.21)$$

Let $[\varphi, \chi, \phi]$ denotes roll pitch yaw. The Quaternion can be converted to Euler angle format by

$$\begin{bmatrix} \varphi \\ \chi \\ \phi \end{bmatrix} = \begin{bmatrix} \arctan \frac{2(q_0 q_1 + q_2 q_3)}{1 - 2(q_1^2 + q_2^2)} \\ \arcsin(2(q_0 q_2 - q_3 q_1)) \\ \arctan \frac{2(q_0 q_3 + q_1 q_2)}{1 - 2(q_1^2 + q_2^2)} \end{bmatrix} \quad (2.22)$$

The primary reason to use the Quaternions is that Quaternions only need 4 parameters to denotes a rotation where as matrix form need 9 parameters. And a small error in Quaternions concatenation will not cause much error where as rotation matrix will be impossible to convert back when orthogonality is lost by minor rounding error. Last but not least, the use of Quaternions can perfectly avoid the cases such as gimbals lock where aggressive rotation(90 degree) is involved.

2.1.0.2 Sensor Model and Calibrations

camera model and calibration The camera model is referred to a matrix which can project the 3D point onto a 2D image plane. Camera matrix, which is usually denoted as K or π , in homogeneous coordinates is defined as:

$$K = \pi = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (2.23)$$

Where c_x, c_y denotes camera optical center in pixel on image and f_x, f_y denotes the focal length in pixels. f_x and f_y are sometimes not equal because of the aspect

ratio α caused by camera hardware. It is possible to convert back to the same focal length f by $f = f_y = f_x * \alpha$. Give a 3D point $P = [x, y, z]^T$, the corresponding 2D projection $p = [u, v, 1]$ on the image can be found by

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} [R|t] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (2.24)$$

If camera undergoes specific motion denoted by R and t , then the resulting 2D point can be found by transform the point to a new coordinate system by pre-multiplying the transform matrix $[R|t]$.

Under ideal case, 2D point can be found by $u = f_x x + c_x z, v = f_y y + c_y z$. By drop the scale factor it can be written into:

$$u = f_x \left(\frac{x}{z} \right) + c_x, v = f_y \left(\frac{y}{z} \right) + c_y \quad (2.25)$$

FIGURE 2.1: An Illustration of Pinhole Camera Model

The result can be seen in Figure 2.1. Because of the nature of lens and the way it is manufactured, all lens comes with radial and tangential distortion. For the majority of visual odometry and visual slam cases, the lens needs to be super wide angle. The type of the imperfection is mostly radial distortion. The distortion can be modeled as $[k_1, k_2, k_3, k_4, k_5, k_6, t_1, t_2]$. The distortion parameters $[k_1, k_2, k_3, k_4, k_5, k_6, t_1, t_2]$ can be found by camera calibration process using a chessboard pattern following the

Zhang's method[55][2]. In most cases, higher order coefficients are not considered such as 5th order of the radial distortion. Then the actual 2D location of 3D point on image becomes

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x \left(\frac{x(1+k_1r^2+k_2r^4+k_3r^6)}{z(1+k_4r^2+k_5r^4+k_6r^6)} + 2t_1 \frac{xy}{z^2} + t_2(r^2 + 2\frac{x^2}{z^2}) \right) \\ f_y \left(\frac{y(1+k_1r^2+k_2r^4+k_3r^6)}{z(1+k_4r^2+k_5r^4+k_6r^6)} + t_1(r^2 + 2\frac{y^2}{z^2}) + 2t_2 \frac{xy}{z^2} \right) \end{bmatrix} \quad (2.26)$$

where

$$r^2 = \frac{x^2}{z} + \frac{y^2}{z}$$

FIGURE 2.2: An illustration of typical stereo camera calibration process. Image is from Zhang's work[2].

All camera has to be calibrated using Zhang's method[2][55] before can be used for motion and structure analysis. For stereo case, both cameras need to be calibrated so that the 3D triangulation tasks can be reduce to 2D search. The detail will be introduced in later sections. For this task, Zhang's method[55] is used for finding both intrinsic and extrinsic of cameras as can be seen in Figure 2.2. They use a

chessboard pattern for finding set of robust matched 3D features . Based on the prior knowledge of number of corner and size of each corner, the position can be found by finding the scale factor(from chessboard hand measured size) on PNP estimated R and t . The for each training set, the R and t are iteratively refined. The motion estimation details can be found in the review section.

The calibration process should have enough set of images at a different scale. The set can also have different yaw and pitch pattern with respect to the camera. It worth notice that, one should minimize the number of roll cases in order to reduce reprojection error. After that rotation, R and translation t between two cameras can be found from this process. And after that, R_1 , R_2 , t_1 , t_2 of each camera rotation and translation with respect to baseline can be found. After that, stereo rectification can be done to align image pixel on the same baseline. Even for a calibrated camera, it is a good practice to consider the remaining pixel alignment error as the variance for structural and motion estimation.

FIGURE 2.3: Wide angle stereo and its calibration process using very large chessboard. Image taken in open sea and building near lon/Lat: 1.315862/104.016163

For common 0.1m baseline stereo e.g. bumblebee camera, the A4 size(0.2mx0.3m) chessboard pattern is good enough. For some cases, stereo is too wide to be calibrated using Zhang’s method[2][55]. When the basedline is proportionally increased to 3-meter , a proportionally up-sized 6mx9m chessboard pattern is needed. It is done to ensure both can see the object and optimize for whole image. And the chessboard must not warp. So the overall weight can goes to hundreds of kg as can be seen from Figure 2.3. Thus it is impossible to carry such board. New methods form Han Wang and Shin [56] are proposed to solve this issue by aligning the sea

floor horizon in both images. By aligning horizon, the camera pitch and roll can be found. Then based on aligning the feature points at infinity point (any boat or mountain), the yaw angle is found. The rest of the translation vector can be found by hand measures. Then an optimization technique is proposed to find the rotation that can best align the features in the same epipolar lines. Such process can be seen from Figure 2.3. The yellow lines are the epipolar lines and the same feature/object should lie on the same yellow line.

IMU model and calibration The IMU measures the linear acceleration and angular velocity of the IMU body frame. Some IMU type directly provides angular measurement.

Here, w represent IMU angular measurement and a represent IMU acceleration measurement. Let $\tilde{x}(t) = [\tilde{w}(t), \tilde{a}(t)]^T$ denote the estimated state from true state $x(t) = [w(t), a(t)]^T$, time-dependent bias $x_b(t) = [w_b(t), a_b(t)]^T$ and measurement Gaussian white noise $x_n(t) = [w_n(t), a_n(t)]^T$.

$$\tilde{x}(t) = x(t) + x_b(t) + x_n(t) \quad (2.27)$$

The measurement noise is given by $x_n \sim N(0, \sigma)$ and can be characterized by using first and the second moments

$$E[x_n(t)] = 0_{3 \times 1} \quad (2.28)$$

$$E[x_n(t + \tau)x_n^T(t)] = \sigma_{x_n} I_{3 \times 3} \delta(\tau) \quad (2.29)$$

where δ is Dirac function and σ_{x_n} hold different values for angular and acceleration measurements. The measurement noise will change overtime and forms a new bias. This bias changes can be modeled by random walk (aka Wiener process or Brownian motions). It is defined by $x_r(t)$.

$$\dot{x}_n(t) = x_r(t) \quad (2.30)$$

where the $x_r(t)$ is modeled by the Gaussian white noise

$$E[x_r(t)] = 0_{3 \times 1} \quad (2.31)$$

$$E[x_r(t + \tau)x_r^T(t)] = \sigma_{x_r} I_{3 \times 3} \delta(\tau) \quad (2.32)$$

The noise characteristics depends on IMU used and the time interval Δt . For discrete type measurement, variance are discretized by

$$\sigma_d = \sigma \sqrt{\Delta t} \quad (2.33)$$

The units for gyro white noise, accelerometer white noise, gyro random walk noise and accelerometer random walk noises are represent by $\frac{rad}{s\sqrt{Hz}}$, $\frac{m}{s^2\sqrt{Hz}}$, $\frac{rad}{s^2\sqrt{Hz}}$, and $\frac{m}{s^3\sqrt{Hz}}$. The sample error caused by the drifting bias can be seen from Figure 5.16.

When dealing with camera and IMU together, the transformation between them has to be found. A package called Kalibr[3] is widely used for calibration and it represent series of work from Rehder et al. [57] [58] [59] [60] [61].

FIGURE 2.4: Camera IMU calibration process using Kalibra. Image from Kalibra package[3]

In Kalibr, they propose to use a set of the QR code for the features as can be seen from Figure 2.4. The physical size of the QR can be measured by tape. The camera can detects the overall shape of the QR code. Based on the shape, the orientation with respect to the QR code can be found. By combining the orientation and size information, the 3D location of the QR code with respect to the drone camera can be found. Drone rotates, the IMU will records set of measurements and camera can measure a set of accurate 3D locations. By finding the similarity during rotation, the relation between IMU and camera can be obtained. The similarity is measured by minimized a set of local window based objective function. Recently Qin et al. [12] [62] in their series of work proposed a tightly coupled method to

allow calibration without a specific pattern. Because no pattern is required, most of the live outdoor recalibration in this dissertation is done using this method.

Visual inertial calibration formulation is similar to visual inertial state estimation formulation. Basically, they are trying to find the same thing which is self motion and minimize the difference from multiple sources e.g. IMU and camera. The difference is calibration have ground truth on 3D features and state estimation is optimized to find a better estimate of 3D features. Based on the initial feature match and Perspective-n-Point(PNP) algorithm, the relative motion can be found. PNP is the problem of estimating the pose of a calibrated camera given a set of n 3D points in the world and their corresponding 2D projection in the image. The detail will be covered in later part of this review section. The true scale motion can be recovered by integrating gravity vector from IMU[12], QR/chessboard code pattern with tape measured size, VICON 3D motion tracking system and etc.

Multi-sensors calibration Recently, LIDAR becomes an essential depth sensing tool in autonomous car industries. LIDARs offer ground truth depth measurement by actively measure the time taken for the light to reach objects. Compare with the passive camera, LIDAR is more resilient to the lighting conditions and small particle haze/smoke. The problem is their physical location has to be known before they can be integrated. It is a complex topic. Let us take a 3-sensors system for illustration. Sensor A and sensor B can be jointly calibrated $T_{A,B}$. Sensor B and sensor C can be jointly calibrated $T_{B,C}$. Sensor A and sensor C can be jointly calibrated $T_{A,C}$. The typical paradoxical problem is the concatenated transform of $T_{A,B}$ and $T_{B,C}$ is not even close to $T_{A,C}$. That leaves the problem of which one should be added into the system.

There are many multi-sensor fusion based research works[63][64][65][66][67][68] with only few works [67] mentioned its calibration method. With those few who includes their calibration method [67], none can show ground truth quantitative measurements and comparison with other approaches. The difficulty is that the internal orientation and position is difficult to be modeled by the external sensor. This problem is commonly observed when multiple 2D LIDARs which can not see the

same thing at the same time. One can only guess where the laser beam is and what's the relative beam direction. There is just no ground truth measurement.

The mainstream multi-sensor calibration studies[34][35][36] use chessboard pattern. They follow a similar concept as monocular camera calibration by solving the least square motion estimates from list of known 3D points. The additional Laser forms the additional constraint for optimization. Recent works such as Dong and Isler [40] show another possibility of using an angled triangular calibration pattern to directly determine the location between 2D LIDAR and camera. Dong and Isler [40] shows that the point to plane constraints from a single observation of a V-shaped calibration pattern composed of two non-coplanar triangles suffice to constrain the relative pose between two sensors uniquely. Then Dong and Isler [40] come up with their analytical approach to solve the calibration by using the point to plane constraints from multiple observations. Their idea is novel with a drawback of a rigid specially designed calibration pattern. For UAV studies, there are plenty of rough landings and crashes. After each accident, sensors on the drone require recalibration frequently on the spot. Thus this sort of fixed 3D pattern-based method is hard to carry in the field.

The other method such as Krishnan and Saripalli [69] uses the small circular opening to align between the camera and 2D LIDAR. By minimizing an objective function that aligns the LIDAR edge to image edge, the relation can be found. In my opinion, this method will always have a residual error as LIDAR has an angular resolution of 0.25 degree and camera will have a certain diffusion effect on the edge. 0.25 degree error in 10 meters range is a relatively large number. There is always going to have a residual error when aligning with those methods. Therefore, his method is not adopted in other third party research. Other works Gomez et al. [70] uses special Trihedrons pattern for both orthogonal constraint and planer constraint. The idea sounds good, but their assumption of continuous tracking of Trihedrons make the system difficult to use. The physical corners are usually full of noise, decoration patterns and other edges. It makes Trihedrons difficult to detect let alone keep track of.

FIGURE 2.5: An illustration of 3D LIDAR to 3D camera calibration. Image from Dhall et al. [4] work

For the full 3D type of calibration, there are a few works which talk about it. Dhall et al. [4] proposed a QR code based 3D alignment-based method for LIDAR and camera calibration. His work requires a full 3D LIDAR and RGBD cameras to work. It is a good basis for the 3D stereo/RGBD camera to 3D LIDAR calibration and this idea can be seen from Figure 2.5. But 3D LIDAR is generally heavier than the 2D LIDAR. For a 2D LIDAR, it is usually about 100 gram. The 3D LIDAR can easily go more than 1 Kg. It is not easy to put such a heavy load on the UAV. Even someone placed it on, the additional power draw, the effect on aerodynamics and other factors will reduce the robustness in drone performance. Therefore, it is not suitable to include this dissertation.

2.2 The State Estimation

2.2.1 A Quick Introduction

The State Estimation is a wide topic which deals with all kinds of sensory feedback to generate local states such as position, landmarks or even semantic labels. The State estimation was also known as odometry[71] or simultaneous localization and mapping(SLAM)[72]. The recent research such as [12],[73], and etc, classify this process as state estimation. When there is more sensors such as IMU, camera and LIDAR, there are more data to associate, transform and concatenate. Those data include raw depth, raw position estimates from various source, and even semantic label[74]. Most of the deployed State Estimation applications are based on optical encoder, LIDAR, and partially camera in the automotive industries. When the camera is used as the primary source of sensory information, it is called Visual State Estimation[71].

Visual state estimation the process of determining the current pose and map (if possible) of the environment through the help of visual cues. It has been active research for a long time but rarely put into actual extensive industry uses for robustness reasons. Initially, state estimation and mapping are independent studies. After years of exploration, people found out that two tasks depend heavily on each other. To know the current change in the pose, you need to know how does the environment looks like and how observation is changed. To understand the current map, you need to know how much your movement is so that observation can be merged into the SLAM model. The terms visual state estimation, visual odometry, and visual SLAM sometimes interchange with each other. Usually, The term SLAM is considered to have more sub-component, such as loop closure and global optimization.

The Loop closure is the problem of recalling revisited scenes for the visual SLAM systems in the topological map. Since the system is not going to match every single key-frame the camera captures due to the heavy computational burden, there exists localization error as the camera revisits the same place. The scale-drift can also cause such localization error for map generation. Loop closing is not an easy problem to be solved, since uncertainties and noises such as illumination, viewpoint changes, etc, have to be concerned. It is a separate topic which usually implemented

by EKF or bag of word model. No matter which way it is implemented, the current hardware can not handle the additional loads and it causes the onboard processing unit to hangs. Therefore it defeats the purpose of robustness and will not be covered in details.

The global optimization approach is referring to the continuous global 3D landmark and poses optimization. The local optimization based approaches are more commonly used in visual state estimation system. It can recover part of the motion. Due to the highly non-convex nature of the SLAM problem, they are susceptible to local minima. The initial and previous estimate of the state plays a critical role in the next motion estimates. The global optimization based methods are usually too computationally intensive to run in a robust manner. Therefore it will not be discussed in details in this dissertation. The local optimization based approaches will be introduced in details in later part of these sections.

The state estimation, odometry, and SLAM terms become increasingly difficult to differentiate. Some odometry method does include global optimization methods [10] just like SLAM case. For running for a short period, they are the same concept. The terms odometry appears in the 1990s [71] and SLAM term popularized in the 2000s [72]. The state estimation can be tracked much earlier. But many multi-sensor fusion works [58][62][75][12] from 2010 prefer to use the term state estimation. Therefore it is logical to think that people understand the problem better and state estimation should be the most representative name for the three terms in future multi-sensor context.

Visual state estimation is something that animal apply every day but often fails at some point. There exist numerous cases where visual state estimation can fail, such as: (1) Outdoor environment. Dynamic lighting changes which affect the resulting image. Also, rain, snow, and fog could cause algorithms to fail. (2) Dynamic environment. In a moving environment e.g supermarket or highway, it is hard to track position as observation is not consistent. The occlusion and wrongly associated feature could easily make the variance of position goes to infinity. (3) Uniform or the featureless environment. Mapping requires association of certain features. In the featureless environment, associate or even finding feature can be difficult. It is quite common to happen in winter snow or clean white areas. (4) Large-scale environment. In a large scale environment, things such as duplicated buildings, also scale drift and loop closing issues can happen. Right now it is still

quite difficult to generate a consistent model for large-scale environments such as school, city or country level. (5) Relative large motion. When large motion occurs, there also exists a high chance that an image acquired is blurred and the feature cannot be associated or tracked. This is typically limited by the CCD or CMOS sensor itself and capturing hardware. In an ideal case, the camera shutter speed should be in terms of nanosecond and camera also have high fps and resolution. But the ideal case never happens as higher shutter speed will cause degradation of the image as less light is traveling into the CCD or CMOS sensor and higher FPS and resolution also impose a heavy load on the data transfer bus which most time is impossible to overcome.

Since there have been many issues with the visual only state estimation, many studies begin to surge to integrate with other sensor sources such as IMU[58][62][75][12], or 3D sources such as Kinect[76] or Velodyne[77]. But that sometimes bring in more problem than it solves. For example, the Kinect has a problem when working outdoors. The IR dots from the Kinect are easily erased out and fused in background solar radiation. The quality of the IMU can have a huge impact. The cheaper type IMU tends to cause the state estimation to have the irrational drift. It is primarily caused by the full integration model that amplifies the noise in each feedback. The details of this process will be introduced in later sections. The Velodyne makes the whole system oversize and overweight. Moreover, it requires both intrinsic and extrinsic calibration when adding a new sensor. Thus it can create more paradoxical problems when adding more sensor.

When everything is calibrated properly, the multi-sensor system [78] [79] [12] do shows more robustness than pure vision-based system [10] [9]. The most recent attempt [80] is making the visual odometry system from the camera, Lidar, and IMU. Where their concept works, but the weight, power, and size are too much for a lightweight drone to carry. So the cost and risk of this method are so high that it cannot be practical in the Singapore context where the government regulates your drone size and weight.

2.2.2 High Impact Breakthroughs

Earlier Works The term state estimation first been introduced in Smith and Cheeseman [71] as a description of geometric uncertainty and correlation between features and landmarks. This is as far as the state estimation concept can be carbon-dated. In the following years, research began to notice the association of the triangulation problems and localization problems[72]. In the following couple of years, the convergence of the state estimation problem has been proven. In the early stage, the state estimation is mostly done using LIDARs.

The visual only state estimation was introduced and demonstrated much later. The first demonstration of such a system is known as Monocular SLAM(MonoSLAM) from [5][81] using EKF based approach as can be seen from left part of Figure 2.6. It uses indirect tracking method by feature detector, descriptor and matcher pipeline. Monoslam uses a probabilistic feature-based map, which keeps track of both the estimates and the uncertainty of the state of the camera position, velocity and orientation, and of the state of all the tracked features. The probabilistic feature-based map evolves over the time, with new features being added, old features being deleted, and estimates and uncertainty being updated during camera motion and feature observation by the Extended Kalman Filter. This first method that proves monocular state estimation works and sparks a series of research behind. Then a lot of works is done to improve the scalability [82] and reduce the processing cost using an improved sampling method in UKF[83]. The limitation is that the state will goes very large as time and distance passes. Eventually the inverse of the residual covariance matrix will be too large to be found in realtime. Therefore it is only suitable for small sized room as can be seen from Figure 2.6.

FIGURE 2.6: An illustration of improvement in feature based method. MONOSLAM(left), PTAM(middle) and ORBSLAM2 (right). Image from the corresponding papers[5][6][7]

After the age of the MonoSLAM, the next key milestone is the Parallel Tracking and Mapping (PTAM) proposed by Klein and Murray [6] as can be seen from middle part in Figure 2.6. It is the world first keyframe-based state estimation. Later this works was improved by using edge information[51] and applied to large-scale stereo version[53]. There is even a mobile AR version implemented by Klein and Murray [52]. The latest along the line of indirect method is ORBSLAM series by [7][84][85][86][87][88]. The ORBSLAM is relative robust method for nonholonomic type of movement model such as car driving sequences from KITTI dataset[31] as can be seen from right part in Figure 2.6. When ORBSLAM is used with the holonomic type of movement such as drone, it quickly loses track. This lost track effect can be recreated easily when there is on the spot turning or medium increase in throttle movement on the Z axis. The algorithm is also not designed for dealing with the large white featureless cloud cases. The improvement offline of indirect method can be seen from Figure 2.6. For simple scene, the indirect method will works fine. But when there is too much repeated feature and too much feature points, this series of method trends to get stuck, lost track and crash. Therefore, this kind of weakness leads to another kind of study which is direct methods.

The next milestone is the rise of the direct methods. The direct methods mean there is no feature descriptor and matcher pipeline. Each pixel is being tracked at the next frame by the association of raw intensity between frames and previously estimated depth. This process can be seen by the left image of the Figure 2.7. The previous depth in the left part of image are propagated into the new image on the right. This series was first start by Dense Tracking and Mapping (DTAM)

FIGURE 2.7: An illustration of new methods. DTAM(top), LSDSLAM(middle) and SVO2 (bottom). Image from corresponding papers[8][9][10]

[8][89] top image in Figure 2.7. It was popularized by Large-scale Semi Direct SLAM(LSDSLAM) series [9] [90]. DTAM is the first dense textured tracking and mapping algorithm that uses direct method and minimized the variational energy.

DTAM generally requires a feature rich scene to work and can only be used with a strong GPU. The feature-rich scene based tracking is usually more challenging for the indirect method. But the direct method can turn the previous challenging problem into a moderate solvable problem by using GPU based tracking for each individual pixel and then globally optimize for the 3D position. The LSDSLAM series are less particular about the scene complexity as it intelligently uses strong gradient pixel for direct odometry estimation and mapping method. Therefore, LSDSLAM is able to run on a simple i5 PC in real-time. They use variance associated inverse depth for each pose updates and each pose updates helps reduces the variance of the depth over time. The small baseline variance is reduced by tracking over a longer period and distance. The latest along LSDSLAM family tree is the Direct Sparse Odometry(DSO) by Engel et al. [91][92]. DSO is a direct improvement of LSDSLAM by added additional photometric calibration step to cope with certain lighting changes.

It worth mention that in between feature based and direct method, there is a method called semi-direct based method proposed by ETH research group [10][93]. In this method, they did not use typical feature extraction and matcher technique. Instead, they use LKT tracker[94] to improve the speed and avoid the RANdom SAmple Consensus(RANSAC) process. RANSAC is commonly used for finding features that has similar motion before. However, RANSAC is a very computational intensive process. For the feature unreliability in semi direct method, they relax the feature by doing an alignment. The result is fast and relative robust in pure vision category. The improvement of this series of work can be seen in Figure 2.7.

Recent Visual-Inertial Approaches In recent years, a lot of new research has gone to either visual with multi-sensors fusion or pure learning-based approaches. Pure learning based approaches will be explained in detail in the later chapter. Currently, most of the previously mentioned authors such as Froster and the Engel are all switching towards visual-inertial state estimation by adding an IMU component to their previous work. Such case can be found by PTAM+IMU[47] and SVO[10]+MSF[95]. Most of the researcher in the community begin to realize that there is no room for improvement for traditional pure visual state estimation.

When dealing with worst cases such as lighting is bad, the feature is minimal or the system is moving too fast, there is no way that visual only state estimation can function well. The Learning-based solution is another way but suffers from other issues. The learning part will be discussed later in this section. Unable to deal with the worst cases is a critical flaw that prevents vision to be extensively used. By assuming the worst cases only last for a short period, the IMU will be an ideal solution to the problem. In terms of long term tracking, IMU is bad and will drift very far. For short period integration of IMU reading, it is still a robust measurement of self-movement. The short term IMU integration provide good initial guesses to the visual state estimation. Thus it further helps the convergence rate of the visual state estimation system.

The famous work in this category are MSCKF[73], OKVIS[45], ROVIO[96] and VINSMONO[12]. There is a detailed benchmark comparison work[11] which compare all of the popular works on popular hardwares. The result can be seen in Figure 2.8. In this graph, the smaller the vertical(cost) and horizontal(resulting error) measurement the better. For example The bright purple color denotes SVO with EKF based approaches. It cost least of all resources but most inaccurate. The VINSMONO with loop closure can gets most accurate result but at high cost in computation and memory. In our in-house test, VINSMONO with loop closure always causes memory overflow and PC hangs after 5-10 mins. Heuristically speaking, there is no best method. The more memory and CPU a algorithm consumes, the better the outcome is.

FIGURE 2.8: Various visual inertial state estimation performance comparison[11]

MSCKF[73], OKVIS[45], ROVIO[96] and VINS-MONO[12] are tightly coupled solutions where SVO+MSF[10][97] or SVO+GTSAM[10][98] are loosely coupled solutions. Difference between MSF and GTSAM is that one is filtering based such as EKF and UKF and one is graph based. For this trend, there are multiple branch approaches for visual inertial integration.

The loosely coupled model is referring to the fusion by combining both position estimation outputs. The tightly coupled model is referring to the fusion by estimating the state from combined raw feature associations. The loosely coupled method is least computational intensive as EKF can run very efficiently on any of the small PCs at the cost of lower accuracy. The tightly coupled process offers more accuracy but requires a more computational cost. Choosing between loosely and tightly coupled method is like choosing a car. If a sports car is used, they can reach from A to B in a very fast manner but at the risk of a crash and stop forever. If the slow tank is used, it can go from A to B in a safe manner but at a much longer time. MSF, MSCKF and ROVIO are typical filtering based approaches. And OKVIS, VINS-Mono, and GTSAM are optimization based methods. But no matter which method chosen, the robustness is for sure much better than pure vision-based approaches.

Typical formulation of visual-inertial state estimation is either loosely coupled Kalman filter[97] based method or minimize bundle adjustment on multi sensor multi frame re-projection error [12]. The formulation [73][45][96][12][98] varies from case to case, but they all can be considered as variant of pure vision based motion estimation methods.

FIGURE 2.9: IMU impact factor on VIO[12]. All the result is by using cheaper IMU. The left image is without loop closure where as the middle and right has loop closure. Image from in-house test result.

Also for visual-inertial state estimation case, IMU quality is an important impact factor to the accuracy and robustness as can be seen in Figure 2.9. In Figure 2.9 only IMU below SGD100 is used, and those type of IMU usually cant be synced with camera, have offset and low update rate. The cost limits its ability to have external IO, internal calibration function and high update rate. Such IMU can easily be seen from earlier work from Ardupilot and PX4 series. There is a list of comparison for IMU quality in Figure. 2.10.

	Accelerometer Bias Error		Horizontal Position Error [m]			
Grade	[mg]		1s	10s	60s	1hr
Navigation	0.025		0.13 mm	12 mm	0.44 m	1.6 km
Tactical	0.3		1.5 mm	150 mm	5.3 m	19 km
Industrial	3		15 mm	1.5 m	53 m	190 km
Automotive	125		620 mm	60 m	2.2 km	7900 km

FIGURE 2.10: IMU based position estimation error for a static placed IMU. The error is due to drift and bias in the IMU modeling section. Data provided in the spec sheet by VectorNav - An IMU manufacturing company

From the experiment done in this thesis, there were a few conclusions. In order to get robust onboard visual inertial odometry output at IMU above SGD100 should be used. In Chapters 3-5 testing case, VectorNAV VN-100 and DJI A3 are used. IMU above SGD100 often comes at 5-8 degree per hour Gyro in run bias and 0.5 to 2-degree pitch/roll error. If the drone is controlled by VICON, then the less than SGD100 type of IMU is good enough. For those more than SGD10k costs or even higher cost, an EKF filter can run with IMU for 2km with odometry shows less than 10 to 100-meter error. The extreme case is the IMU used on the inter-continental ballistic missile(ICBM). Those type of high precision IMU can have less than 500-meter error over 10,000km traveled distance. But for that payload of 100k ton TNT equivalent, 500 meter doesn't make any difference for the reason that all will be obliterated in that area. For the small sized UAV, it is challenging to mount large IMU for that kind of accuracy

Current Multi-Sensor Approaches In this section, there is no standard approach or pipeline. Before 2014, the pure vision way was considered as the primary area of study in state estimation. People have a belief that vision is all it takes for

a robot to go around as human does it daily. Between 2014 and 2018, there are increasing studies that bind the visual and inertial for robust state estimation. The reason why visual-inertial estimation gets popular is that the researcher began to understand that human also depends on IMU balance system to get around. But the current system can never get to what the human is capable of. The human eye has more than 500Megapixel which is hard to replicated and processed with current technology. Most of state estimation system can run in real-time with at most 1Megapixel camera. Therefore, it is logical to add other non-conventional sensors such as LIDAR or Radar for the state estimation system. So it is reasonable to assume this is the next primary area of study in state estimation system.

The most recent vision based multi-sensor fusion is the series of work from CMU [13][77] as seen in Figure 2.11. They place sensors in series circuit format to fuse the data. The IMU-based state estimate is combined with visual-inertial state estimation for initial fusion. After that, The initial fusion is then fused with 3D LIDAR-based scan matching. This is a typical high-frequency to the low-frequency pipeline. The initial fusion follows the visual-inertial state estimation mentioned in previous paragraphs. The LIDAR to visual estimation fusion concept is something novel. The visual-inertial fusion based method will introduce a set of 3D point clouds from initial/previous state hypothesis. The 3D point clouds generated from the camera are aligned with 3D point clouds from LIDAR. The initial fusion is known to have some drift but it provides good initial guess to other processes. The local LIDAR based scan matching is used to correct the local drifts. The process can be seen in Figure 2.11. By estimating the scan match result from the previous 3D point clouds, a good state estimation result can be obtained.

However, the problem is that the computational cost for his method[13] is way too high. Such high computation cost can also be seen in LOAM by Zhang and Singh [99] or BLAM from the University of California, Berkeley 3D LIDAR based methods[100]. Moreover, both LOAM or BLAM requires heavy multi-line 3D-LIDAR which makes it hard to run in the small size drone prelimited by Civil Aviation Authority of Singapore(CAAS).

FIGURE 2.11: Visual LIDAR Inertial Odometry[13]. Most complete solution when given enough weight budget and risk free environment

For their earlier work[77] which is using only pure 3D LIDAR, it is still able to run in real time. For the latest work[13], it takes 300ms to process and get one updates. In the in-house flight test, Their work[77][13] has a problem when dealing with the vertical climb and featureless area. There is also a robust drone control problem that comes with the processing delay. To cope with the vertical climb and featureless area, Zhang and Singh [99] adopted a rotating Hokuyo device for capturing large local area of features. For this work [99], the rotating mechanism is hard to replicate as the wire will tangle and break. They sell their work[99] at 60k SGD, which comes with expensive IMU. The product they come up with only can run in post-processing mode. For 1 min of sequence, it takes about 20min to process. It is possible to try to run the rotating Hokuyo idea on the drone in real time, but the risk of breaking very expensive equipment is too high. During the

implementation of the proposed method in this thesis, no such high-end IMU is used. The IMU used are embedded in the commercial DJI flight controller unit.

There is a separate pure 2D LIDAR [101][102] [103][104] based approach which do not use camera for drone navigation. [102] was one of the earlier attempts for 2D LIDAR based navigation. Their method works well at the indoor controlled environment. When place at outdoor with plenty of rocks or obstacles, their [102] method will have trouble finding the ground reference. [101] proposed a method that most closely resembles one of the proposed methods in this thesis and publishing at a similar time when this was developed. They opt for publication and the author opt for competition and award. But in [101] there are multiple un-presented issues such as how multi-sensor calibration is done or whats the accuracy of altitude estimation. Besides they did not control the drone with the system and they use it as survey tools. The reason why they did not control it autonomously is that the Phantom series do not have DJI Onboard SDK virtual RC control¹. Other approaches like [103][104] uses GPS for positioning which is less relevant to this thesis.

2.2.3 State Estimation Methodology Classification

All state estimation research work can be classified by some characteristics. Such characteristic can be based on what is the feature used, what is the tracking method used, what is the solver used and what the outcome of the work.

2.2.3.1 Classify by Feature Type

Based on the feature type used in motion estimation, state estimation can be classified into 3D features, 2D and 3D concurrent features and 2D features. The variation is usually caused by the input sensor, but there will always be cases where people intentionally combine multiple methods[13] to achieve a specific goal such as processing speed or quality or robustness. Input sensor for state estimation can be

¹According to <https://developer.dji.com/> only OSDK can have onboard virtual RC control. Only limited DJI drone such as M200 and M600 support OSDK

a stereo camera, monocular camera or RGBD camera. Three input types represent three or more distinct way of motion estimation methods.

3D to 3D motion estimation For pure 3D feature case, the sensor used is typically stereo cameras or RGBD camera which is considered as the easiest method to understand. In LIDAR term, this is generally called Iterative Closest Point matching(ICP) as seen in Figure 2.12. As shown in the Figure, the correlation between scans different multiple time are found by minimizing local 3D alignment error. The motion/transformation that gives the minimal alignment error over multiple frames is consider to be the best result.

FIGURE 2.12: Iterative Closet Point matching. Image source[14]

In this case, the feature is coming in terms of $(x, y, z, R, G, B, 1)$ for a color camera or $(x, y, z, I, 1)$ for a greyscale camera where 1 here denotes the true scale. Since true scale and position are known, the camera motion T_k can be computed by finding the transformation which aligns two or more 3D feature sets. The general formulation to find T_k can be considered as minimizing the distance between two aligned 3D feature sets.

$$\operatorname{argmin}_{T_k} \sum ||X_k^i - T_k X_{k-1}^i|| \quad (2.34)$$

where i is the i -th feature and X are the coordinates of 3D points. The minimal case solution needs three sets of 3D to 3D non-collinear correspondences. For many cases where there exist $N \geq 3$ sets of correspondence, the translation can be computed by differentiating centroid of consecutive 3D observation sets.

$$T_k = \hat{X}_k - R\hat{X}_{k-1} \quad (2.35)$$

The rotation can be computed using Singular Value Decomposition(SVD) as

$$R_k = UV^T \quad (2.36)$$

where $USV^T = SVD((X_{k-1} - \hat{X}_{k-1})(X_k - \hat{X}_k)^T)$. When the measurement noise or uncertainties of the 3D points are known, then it can also be added as weights into the motion estimation as described in work[105].

The typical limitation for this 3D feature type motion estimation is that estimated motion outcome depends heavily on the accuracy of input point clouds.

For general stereo case, it is hard to get a dense and accurate disparity map due to all kinds of error such as calibration error, matching error, lack of feature and optimization error. The stereo range, quality and measurement noise heavily depends on the baseline. The larger baseline can measure longer range but at the cost of higher visual occlusion and higher uncertainty. The uncertainty comes from the fact that each pixel can represent objects that are far away. In such a case, the 3D to 3D based motion estimation will be an inaccurate method due to the large variance of depth values in different temporal frames.

The small baseline stereo can produce an accurate stereo matching result at closer range. Based on in-house experiments in the past, a typical Stereo camera with 12 cm baseline can detect objects fairly accurately up to 5-7 meters. It is also possible for such 12 cm baseline stereo to produce a stereo disparity at 10 meter scene. But the noise is much higher. The major issue for short baseline stereo is that drone can easily lose all 3D feature once in mid-air. For a custom 3m baseline stereo system, depth can reach as far as 300m with fairly low uncertainty as can be seen from [106][107][56][108][109][110]. But there is no way a small size drone can carry such as wide baseline stereo. Therefore, pure 3D to 3D based motion estimation is least preferred when it comes to autonomous drone flying. In reality, the small baseline stereo is used widely in the drone from close range obstacle avoidance only. Therefore the benefit from stereo is not fully explored in current drones.

FIGURE 2.13: An illustration of epipolar geometry.

2D to 2D motion estimation In pure 2D to 2D case, motion can be estimated by decomposing from the essential matrix. The essential matrix is an important concept in 3D computer vision. To explain what is the essential matrix, we have to start from the basic concept of epipolar geometry as seen in Figure 2.13.

The epipolar geometry is a set of constraint that generated from camera motion and feature points. Taking a moving camera for example as seen in Figure 2.13. A world 3D feature P was observed in the time k and time $k - 1$. The image center of camera O_k , the image center of the camera at O_{k-1} , and the 3D point P forms a plane when they are not collinear. The plane can be seen as the triangle among Point and two camera center. It has many nice prosperities. When they are coplanar, the three sides of the triangle scalar triple product is always zero e.g. $(P_{k-1} + T)^T(P_{k-1} \times T) = 0$. In later sections, this principle is used frequently.

It is also mentioned before, Rotation and translation can be used to describe a transformation of a 3D feature from one view at O_{k-1} to another view at O_k .

Given the same 3D point P_k from camera view at time k and P_{k-1} from camera view at time $k - 1$, R is 3 by 3 rotation matrix and T is 3D vector, The essential matrix E can be derived from general point coordinate transformation equation:

$$P_k = R(P_{k-1} + T) \quad (2.37)$$

By taking the vector product with T and the scalar product with P_k , the Equation 2.37 can be rewrite as:

$$P_k^T([t]_{\times} R P_{k-1}) = 0 \quad (2.38)$$

where $[t]_{\times}$ denotes the matrix representation of the cross product of T . By replacing the transformation with essential matrix E in the middle, the Equation 2.39 can be rewrite into:

$$P_k^T E P_{k-1} = 0 \quad (2.39)$$

The essential matrix has rank2 which both left and right of the equation have nullspace. Currently this essential matrix is still only related to 3D coordinate. By using the fact that

$$p_{k-1} = \frac{f}{Z_{k-1}} P_{k-1}, p_k = \frac{f}{Z_k} P_k \quad (2.40)$$

, the Equation 2.39 can be converted into 2D form. Where Z denotes the Z direction of camera view coordinate system. Therefore the final correlation can be written as:

$$p_k^T E p_{k-1} = 0$$

where

$$E = [t]_{\times} R_k \quad (2.41)$$

The p is the homogeneous representations of corresponding image coordinates.

Generally, the Rotation R and translation T can be found by performing a Singular Value Decomposition(SVD) on the Essential Matrix E . There are other methods to find the motion, but SVD is most popular methods.

A singular value decomposition of E can be written as:

$$E = U \Sigma V^T \quad (2.42)$$

where U and V are orthogonal 3×3 matrix and Σ is a 3×3 diagonal matrix with

$$\Sigma = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The diagonal entry s is the singular value of E . By defining W as:

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The rotation and translation can be found by:

$$\begin{aligned} [t]_{\times} &= UW\Sigma U^T \\ R &= UW^{-1}V^T \end{aligned}$$

There are many methods to construct an essential matrix from a set of corresponding points using the epipolar constraint. The standard approaches involve five 2D to 2D feature correspondences which is commonly known as perspective 5 point algorithm (P5P)[111].

Because the essential matrix is a faithful representation of the rotation and translation, it only has five degrees of freedoms(DOF). Therefore, the essential matrix must satisfies the additional constraints:

$$\det(E) = 0 \quad (2.43)$$

$$2EE^TE - \text{tr}(EE^T)E = 0 \quad (2.44)$$

There are nine elements in essential matrix E which requires nine equation to find them. Because of the additional constraint, only two of the equations are algebraically independent. The mathematical proof and derivation can be found in the book "Multiple View Geometry in Computer Vision" by Hartley and Zisserman [112]. Because of the additional constraint, only five equation is needed to find the essential matrix. For the given five points, one can get a null space representation of essential matrix by $E = xE_0 + yE_1 + zE_2 + wE_3$. The $E_i, i = 0, 1, 2, 3$ are the null space bases. By using the fact that E is homogeneous, without loss of generality, let w is equal to One. The equation 2.44 forms a 9 x 20 coefficient matrix corresponding to a monomial vector:

$$[x^3, y^3, x^2y, xy^2, x^2z, x^2, y^2z, y^2, xyz, xy, xz^2, xz, x, yz^2, yzy, z^3, z^2, z, 1] \quad (2.45)$$

By using the Gaussian-Jordan elimination method, the upper triangle form of the matrix representation of equation 2.44 can be found. Since the determinant must vanishes, the determinant forms the tenth-degree polynomial. By solving the 10th-degree univariate polynomial, the ten solutions of z can be obtained. By back-substituting those real roots, the unknown element can be solved one by one. Thus the essential matrix can be recovered.

There are other methods such as Longuet-Higgins equation or commonly known as eight-point algorithm(P8P). However, the result from P8P is often not accurate enough due to inaccurate measurement. The result from the SVD can generate non zero least singular value. Consequently, the solution of the homogeneous linear system of equations may not be sufficiently accurate to be useful. Therefore, P5P is the most efficient way of extracting motion from 2D to 2D feature association. More details of Perspective N Point(PNP) and epipolar geometry can be found on the book "Multiple View Geometry in Computer Vision" by Hartley and Zisserman [112].

It worth mention that, the trajectory generate from 2D image sequences usually will end up with a relative scale. So there are many works utilize all kind of environmental information to get the true scale. Such attempts can be seen from estimating scale factor from robot height and planar road model[113] and estimating scale factor from structure from motion and vanishing point [114]. The most robust and popular way is to use other sensor such as LIDAR or IMU concurrently with 2D to 2D motion estimating. By matching the forward acceleration vector and image motion vector, the scale factor can be found. The details of the multi sensor fusion based method will be introduced later.

3D to 2D motion estimation The 3D to 2D motion estimation usually involves minimization of re-projection error instead of minimizing the Euclidean distance between two observations. The transformation T_k is can be found from the 3D o 2D correspondence P_{k1} and P_k . Here we denote 3D point as a capital letter and 2D point as the small letter. P_{k1} can be triangulated by stereo data in stereo setting or calculated through motion stereo in the monocular setting. The monocular case is more complicated as it requires data from three views. The re-projection error can be formulated as shown below

$$\operatorname{argmin}_{T_k} \sum ||p_k^i - K * T_k P_{k-1}^i|| \quad (2.46)$$

where p_k^i and P_{k-1}^i are the corresponding i-th 2D and 3D points, K is the camera projection matrix contain image center and focal length and T_k is the relative motion we want to estimate. This formation is also considered as a Perspective N Point (PNP) problem. But since the original 3D feature points are known, the

minimal solution requires three sets of 3D to 2D correspondence(P3P) to estimate the motion. The formulation of the P3P problem can be seen in Figure 2.14. The 3D features are formed into the new camera frames follows the same perspective positions. To effectively describe the relationship, we have to simplify the model as seen in Figure 2.14.

FIGURE 2.14: P3P problem setup and simplified model.

Let A,B,C be 3 feature points in 3D and P be the new camera location. A,B,C and P form a trihedron with 4 triangular surfaces. For each surface, the Cosine rules apply to the three sides and form the set of Equations 2.47.

$$\begin{aligned} Y^2 + Z^2 - YZp - b'^2 &= 0 \\ Z^2 + X^2 - XZq - c'^2 &= 0 \\ X^2 + Y^2 - XYr - a'^2 &= 0 \end{aligned} \quad (2.47)$$

where $X = |PA|, Y = |PB|, Z = |PC|, \alpha = \angle BPC, \beta = \angle APC, \gamma = \angle APB, p = 2\cos\alpha, q = 2\cos\beta, r = 2\cos\gamma, a' = |AB|, b' = |BC|, c' = |AC|$. By using the fact that, all trihedron side must be a positive number and all angle must be less than 180 degree, the Equations 2.47 can be further simplified into Equation 2.48.

$$\begin{aligned} y^2 - rxy - 1 + qx &= 0 \\ y^2 - py - x^2 + qx &= 0 \end{aligned} \quad (2.48)$$

where $X = xZ, Y = yZ, |AB| = \sqrt{v}Z, |BC| = \sqrt{av}Z, |AC| = \sqrt{bv}Z$. By using the Wu-Ritts zero decomposition method[115] on Equation 2.48, a set of analytical solutions for P3P problem will be obtained. The resulting solution may have the most four distinct solutions. By using a fourth point measurement, the solution can

be disambiguated. This theory is easy to understand but difficult to use because of the necessity of need an additional point for anchor the result.

Another standard approach to solve 3D to 2D motion estimation is the Efficient PNP (EPNP) where $N \geq 4$. In most of motion estimation case, there are more than 4 corresponding points in the scene. The more point will increase the robustness of motion estimation against the noise. But it will also increase the computation complexity. The EPNP method is based on the notion that each of the points can be expressed as a weighted sum for four non-coplanar virtual control points. The camera pose can be found from these control points. let P denotes the reference point in the world coordinate system and C denotes the control points. Each reference point can be expressed as a weighted sum of the control points

$$P_i = \sum_{j=1}^4 \alpha_{ij} C_j \quad (2.49)$$

where $\sum_{j=1}^4 \alpha_{ij} = 1$. The α denotes the homogeneous barycentric which are uniquely defined and can be found easily. This relation remains the same for both local and world coordinate system.

In theory, all the control point can be chosen at random locations. Empirically, they are choosing at principle direction of the point data for robustness reason. Let K be the camera intrinsic matrix and $p \in 2 \times 1$ be the 2D projection of point $P \in 3 \times 1$ and the following equation can be obtained:

$$w_i \begin{bmatrix} p + i \\ 1 \end{bmatrix} = K P_i^c = K \sum_{j=1}^4 \alpha_{ij} C_j^c \quad (2.50)$$

Where w denotes the scale factor of a point. The Equation 2.50 have two equation for row and column for a single point. By removing the scale factor, a linear system can be generated with two equation for all N reference points:

$$Mx = 0 \quad (2.51)$$

where x denotes the 4 control points of 12×1 vector and M is a $2n \times 12$ matrix which is generated by 3D point and projection intrinsic matrix. Solving for x will give set of control points location. Then based on previously mentioned P3P, a set

of the more accurate motion estimation can be obtained. This is by far the most commonly used method for 3D to 2D based estimation.

Normally there would be noise in the measurement of the corresponding points, therefore a RANSAC is commonly used for removing the noise. The idea of RANSAC is randomly proposed a model and test the input data with the proposed model. After N iteration, the best suited model is selected and outliers in the matching process can be removed.

In the stereo camera case, the 3D point can be measured directly. But it would not help in motion estimation as there is an additional stereo uncertainty involved. Dr Nister [116] and Dr Shen[12] shows that by using only monocular triangulation for motion estimation, the resulting motion estimation is more accurate. In monocular case, it is not trivial to obtain ground truth 3D point as it lacks external measurement information such as baseline in stereo case. In this case, it is common practice to first estimate initial motion from two images after a pure translation view change. Such change can be treated as relative scale or unit scale which gives a measurement between virtual distance and real distance. Once initial motion is obtained, 3D point observation and triangulation are done to generate a 3D point sets which are up to the initial motion stereo baseline scale. This way of finding 3D points are usually studied as structural from motion (SFM) problem. The initial baseline is then compared with the gravity vector of IMU, GPS or VICON to compute the scale factor

The process of motion estimation and 3D point triangulation runs concurrently and provide information for each other at the next observation. The difficulties in this kind of method are to maintain feature tracking over multiple frames, which is commonly a difficult task as feature points will undergo geometric and photometric distortion. 3D to 2D matching is the most challenging and exciting method as shown in works from [9] [90] [10] [93] [12] and etc. But it is the most accurate way to get motion estimates. There are more study in 3D to 2D based estimation than any other motion estimation in the recent decade.

2.2.3.2 Classify by Tracking Method

Based on the tracking method, Visual state estimation can be classified into the Direct[90], semi-direct[10] and indirect method[6]. This is a new way of classifying visual state estimation approach influenced by the new work [10][90]. The main difference is how observation is associated and generated.

FIGURE 2.15: An illustration of different tracking methods.

Indirect method The indirect approach is referred to feature keypoint detector, descriptor and matcher as shown in Figure 2.15 and it can be widely seen in such as PTAM[6] and PTAM inspired work[88]. It was widely used in early visual slam work due to early SLAM work has memory and computational constraint which can only use a few stable points. This approach is simple and robust as all the observations are deemed to be unique and stand out. Once feature uniqueness is ensured, the correspondence is calculated more accurately and therefore an accurate pose can be estimated. However, as the name suggested, finding unique feature point will ignore many lines and plane information which will lead to a sparse feature observation. The sparse feature is more likely to be influenced by a repeated feature such as chessboards and brick wall which leads to inaccurate localization and mapping.

Such a failure mode can be seen easily in the demo[6]. To increase the accuracy of state estimation system, one easier way is to increase the feature count. However,

when feature count increases, the computational cost from feature matching process can grow out of bound. There are methods such as bucketing technique [117] [118] to reduce the feature count. But the computational cost is still consider to be high and need additional step such as RANSAC. It became lesser favorable when other method such as inverse depth inspired series(direct) [90][119][120][121][122][123] or semi direct [10][12] series of works was proposed.

Semi direct method The semi-direct approach is the type of feature association technique as shown in Figure 2.15 where the detected feature is tracked over frames through the help of KLT tracker [94][124] [125]. The derivation of KLT tracker can be found in the other research work[126] and will not be discussed here. This method was first seen in the RGBD(kinect) based work[127]. Researcher from ETH research group[10] shows that by tracking the detected feature to next frame will have much lesser computational cost when compared to extract feature descriptor and match the extracted feature descriptor. And feature detection is usually done by a series of convolution kernels which make is a much faster process than extract feature descriptors and match them in brute force manner.

In [10], authors use a downward facing camera on a UAV. The result was very impressive even with very fast motion and reconstructed map was fairly dense. At the time this method is proposed, it is considered one of the key breakthroughs. But soon, research realized that SVO only could deal with planner scene. The semi-direct feature usually has a problem dealing with perspective changes. Until recent SVO2[10]² and [12], full 3D based method was proposed and demonstrated. this feature can still be very useful when used in conjunction with complex feature association/aggregation method.

²SVO1 and SVO2 are from same paper but with different implementation

FIGURE 2.16: An illustration of direct tracking.

Direct method The direct approach is the latest work [90][9] for feature association. Direct method tracking images and calculate motion as can be seen in Figure 2.16. The idea is to find the motion by global minimized the gradient associated intensity error. The orange and blue area are the residual to be minimized. This method can be consider as a direct byproduct from inverse depth series of work by [119][120][121][122][123].

By finding the best transform to minimized the orange and blue part, the local optimal solution can be found. The direct approach direct search pixel p_i from image I_{k1} on current image I_k along epipolar lines. Such method will create the dense mapping of the environment and therefore require much more computation which limited its application to GPU only [8]. It has brought forward to CPU only, but only compute feature tracking at high gradient parts of the image which are usually a line, edge and corner to form a semi-dense SLAM solution [9].

Direct method is considered to be the superior method in visual SLAM as it can produce a dense map and with good tracking result with the help of a wide angle lens camera. From robustness point of new, direct method suffers from many problem such as robust feature intensity and gradient intensity noise. The tracking from the direct method often fails due to simple lighting changes. The map created is often with high noise for the real world lighting is more dynamic.

FIGURE 2.17: A classification of real time visual odometry based on number of feature and tracking method. Green denotes pure vision based method, blue denotes mult sensor fusion based method

Discussion on Tracking Method and Feature Density When considering from feature and tracking method level, the current famous real-time odometry/S-LAM works can be classified based on Figure 2.17

Besides the one previously already mentioned, the DVIO[128] is the newly proposed method with direct visual inertial odometry to generate pose. And then map the environment from the pose by using SFM on semi-dense features. SDVIOM [129] is one of newer approaches which uses the semi-dense method to track positions and use the onboard stereo to map the environment. The Kinect fusion [76] represent another series of work [130] [131] [132][133][134] in which [76] used direct 3D feature and its surface normal to estimate motion. The idea sounds very good but in reality is very computational intensive even with the state of the art GPU. I think it should be classified as indirect method with dense features as surface normal are additional matching conditions.

There is a separate new series of studies which uses a learning-based approach on frequency domain transformed RGBD sensor. The first one being None-Iterative

SLAM (NISLAM)[135] which uses the idea of Kernelized Correlation Filters(KCF)[136] to match point clouds. This NISLAM can get a comparable result in indoor sequence. However, the NISLAM code was not shown in public and in theory it will not work with stereo and works outdoors. Because this method needs to transform into the frequency domain and perform KCC, therefore it will be classified as indirect methods. Chen et al. [137] uses dense tracked optical flow features to estimate robust visual odometry. The result was better but the computational cost is for sure higher. Figure 2.17 only shows a sparse set of odometry and SLAM. Currently, the lower left corner can be implemented in real time in CPU. But whether those method are robust or not, it is another issue. The other method on top right is either never tested case or need heavy GPU assistance for real-time implementations. Most of the cases, a semi direct sparse method are preferred as can be seen from those recent popular implementations.

2.2.3.3 Classify by Solvers

A typical solver for visual SLAM problem can be categorized into three groups, namely (1) filter based method, (2) structure from motion-based method and (3) biologically inspired method.

Filter based method The filtering based approaches represent early attempt to solve the problem. This filter based method consists of various Kalman filter based method and various particle filter-based methods.

The popular EKF based approaches will be introduced in details as there is related work in this dissertation. Let k denotes a time instant, x_k denotes drone state vector, and u_k denotes optional user input from previous time $k - 1$. The m represent all landmarks and z_k be the current observation at time k . The EKF model can be formulated as

$$x_k = f(x_{k-1}, u_k) + w_k$$

where function f is the drone kinematics and w_k is the error in the kinematics model and possible noise. w_k is assumed to be zero mean Gaussian noise with covariance Q_k . The observation model at time k can be modeled as

$$z_k = h(x_k, m) + v_k$$

where function h denotes observation in current states and v_k models the correlated error in the observation. The typical EKF consist of 2 steps namely prediction and update. In the prediction stage, drone motion is predicted with respect to previous stages and updates. In the update stage, the landmark observation is updated. When both steps update the sets of parameters in robot states and observation locations, a covariance matrix P is calculated to provide a measurement for the covariance between drone states and observation.

$$P_{k|k} = \begin{bmatrix} P_{xx} & P_{xm} \\ P_{xm} & P_{mm} \end{bmatrix}$$

At time k , the prediction is done as shown follows

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k)$$

$$P_{xx,k|k-1} = \nabla f P_{xx,k-1|k-1} \nabla f^T + Q_k$$

where ∇f denotes the Jacobian of f during estimating $\hat{x}_{k-1|k-1}$. The update step is done by

$$\begin{bmatrix} \hat{x}_{k|k} \\ \hat{m}_k \end{bmatrix} = [\hat{x}_{k|k-1} \hat{m}_{k-1}] + W_k [z_k - h(\hat{x}_{k|k-1}, \hat{m}_{k-1})]$$

which the 2nd term measures the difference between observation and projected prediction. And the W_k is the Kalman gain. The covariances are updated as

$$P_{k|k} = P_{k|k-1} - W_k S_k W_k^T$$

where

$$S_k = \nabla h P_{k|k-1} \nabla h^T + R_k$$

$$W_k = P_{k|k-1} \nabla h^T S_k^{-1}$$

where ∇h is calculated at $x_{k|k-1}$ and m_{k-1} . The W_k is the Kalman gain. From a mathematical point of view, $P_{k|k}$ will go out of bound as drone moving forward. It is a key limiting factor.

There is another key filtering based method called particle filter. Most of the Kalman Filter based approaches use parametric approach e.g Gaussian distributions. The particle filter based approaches revisit the filtering based method by non-parametric approach e.g sampling the distribution. A particle set is made of state hypothesis and importance weight. The weighted samples represent the posterior. The other step such as prediction and correction are similar to those of standard Kalman filter. In recent year, most of the research adopt optimization based method. Even for the Visual Inertial state estimation, researchers[73][47][95] more often adopt Kalman filter than particle filter. Therefore, particle filter will not be discussed in details here.

There is a long list of filters which scholars have tested in the area of odometry/slam problem such as extended Kalman filter[138] and its variance, Unscented Kalman filter[139], Extended Information filter and its variance, particle filter and etc. All of the mentioned techniques show promising results to some degree. However, all these approaches have a huge problem when operating in a large natural environment because of the nature of its usage. The very early approach to address slam problem was proposed by Smith and Cheeseman [71]. They introduce the idea of the stochastic map and using EKF to solve slam problem. The original EKF-SLAM formulates the problem by using a state vector containing pose and map element and then being recursively estimated from non-linear models of observations and transitions. However, if any, an association of data is incorrect, it could lead to divergence of the entire filter. And the complexity of EKF is quadratic with respect to a number of observations which lead to an inefficient solution to a large-scale problem.

In the next decades, much research is done on how to reduce the SLAM computational complexity, such as Compressed Extended Kalman Filter[140], Sparse Extended Information Filter[141], Divide and Conquer algorithm[142] and was later improved by Conditional Independent Submap[143]. Another early representative work is FASTSLAM[144] and was improved in the following work[145] to handle unknown association data. The key concepts are maintaining pose distribution as a

set of Rao-Blackwellized particles, where each particle represents a trajectory, the map is maintained by an Extended Kalman Filter, with multiple hypotheses on the association of data and the solution survives with a probability measurement. This approach involves generation of particle and re-sampling process to prevent the particle from degeneration, which leads to problems such as how to determine the correct number of particles for moderate accurate representation and a higher number of particles is more memory and CPU demanding. It is worth mention that early work using filter base approaches assume a constant linear and angular motion constraint which is impossible to see in nature. Such assumption was taken by research such as EKF based visual MonoSLAM [5][81] and UKF based stereo visual odometry [139]. To solve the problem of constant motion assumption, the dynamic motion model was later proposed by other works[146] [147] to make the system more robust.

More recent approaches[148] uses only drone pose as state vector and remove other observation vectors to make the covariance bounded. It is done by directly combining multiple previous work[10][149]. Since the system is bounded, this type of approaches are usually called loosely coupled system and requires minimum computational power. It is one of the popular approaches on ultra lightweight UAVs[148]. When compared with other loosely coupled methods [98], the newer method[149] is numerically bounded and does not have any memory leak issue. There are also a trend of generalized EKF/UKF package[66][149] when the others was using highly integrated EFK models. There are other combinational method with works[149][10] to form a Visual-inertial odometry state estimation. The combinational method is a very fast implementation without complicated computational complexity.[66] was implemented with GPS IMU type data but have the possibility to use other sensors. In this dissertation, part of [66] implementation was included in the proposed fusion system.

To make a short summary, filter based approach represents an early engineering solution towards visual SLAM problem. It suffers from many paradoxical issues such as relations between one or more of the characteristics such as memory, computational, static and dynamic motion model, accuracy, scale. For example, high accuracy and large scale each usually requires an insane amount of memory and computation power. When demand for high accuracy and dense output is high,

the computational requirement can be astronomical for filter based approaches. Therefore in most recent works, Structure from motion are more often adopted for high accuracy result.

Structure from motion based approach The structure from motion (SFM) also known as bundle adjustment based method is the newer approach to visual state estimation problem when compared to filter based approach. The structure from motion-based method usually formulates the state estimation problem as a nonlinear least square problem. The idea originated from Bayesian theorem where

$$P(X|Z) = \frac{P(Z|X)P(X)}{P(Z)} \quad (2.52)$$

where X denotes current states and Z denotes the observations from IMU, 2D feature from image or 3D feature from LIDAR. Now the problem can be formulate as a max a posteriori problem which describe as finding the optimal state estimation given the observations. Let f denotes the state transition function and h denotes observation to state transition function. The visual state estimation problem can be formulate as

$$x_k = f(x_{k-1}) \quad z_i = h(z_{i_k}, l_{i,j}) \quad (2.53)$$

where state vector denotes sensor or baselink states $X = [x_1 \dots x_n]$, $L = [l_1 \dots]$ denotes 3D points and Z denotes the feature positions. The equation 2.52 can be rewritten as

$$\begin{aligned} P(X, L|Z) &\propto P(Z|X, L)P(X, L) \\ &\propto [\prod_{i=1}^M P(z_i|X, L)]P(X) \\ &\propto P(x_0)[\prod_{i=1}^M P(z_i|X, L)][\prod_{k=1}^M P(x_k|x_{k-1})] \end{aligned} \quad (2.54)$$

(2.55)

It is under assumption that both state estimates and landmarks are independent and no prior about L . The measurement model are also assume to be independent so that the Markov process model can be applied. By applying the log operation to both end of the equation 2.54, the expression can be change into

$$\operatorname{argmax}_{X,L} P(X, L, Z) = \operatorname{argmin}_{X,L} -\log(P(X, L, Z)) \quad (2.56)$$

The residual covariance of process noise and measurement noise are normalized. And equation 2.57 can be derived into its squared Mahalanobis distance form 2.66. The Ω denotes the measurement covariance and it is omitted during the derivation. By only considering the observation residuals, the nonlinear optimization function can be simplified to

$$\tilde{\xi} = \underset{\xi}{\operatorname{argmin}} \sum_{i=1}^M \|f_i(\xi) - z_i\|_{\Omega}^2 \quad (2.57)$$

This equation can be solved iteratively by

$$\begin{aligned} \underset{\Delta\xi}{\operatorname{argmin}} \sum_{i=1}^M \|f_i(\xi^s + \Delta\xi) - z_i\|^2 \\ \xi_k = \xi_{k-1} + \Delta\xi \end{aligned} \quad (2.58)$$

By apply the first-order Taylor approximation to equation 2.58, the update equation can be rewritten as

$$\begin{aligned} \Delta\xi &\simeq \underset{\Delta\xi}{\operatorname{argmin}} \sum_{i=1}^M \|f_i(\xi^s) - z_i + J_i + \Delta\xi\|^2 \\ &\simeq \underset{\Delta\xi}{\operatorname{argmin}} \sum_{i=1}^M \|r_i(\xi^s) + J_i + \Delta\xi\|^2 \\ &\simeq -(J^T J)^{-1} J^T r(\xi) \end{aligned} \quad (2.59)$$

(2.60)

where $J = \partial r / \partial x$ denotes the Jacobian matrix $J = [J_1 \dots J_M]^T$ and r denotes the residual of the measurements $r = [r_1 \dots r_M]^T$. This is a typical graph based formulation for state estimation.

Full maximum a posteriori(MAP) based method is a slow method as it need to recalculate the Jacobian for each iteration. And there is a fact that $(J^T J)^{-1}$ may not always invertible. But this slow process also ensures most accurate state estimate.

There exists a short cut method called fix-lag smoothing where it uses the sparsity prosperity of Hessian matrix to efficiently find the Hessian matrix H^{-1} where $H^{-1} = (J^T J)^{-1}$. This is done by using a concept called Marginalization or Schurs complement. Let $\bar{A}$ denotes the Schurs complement of A and $\bar{D}$ denotes the

Schurs complement of D. The Schurs complement is defined by

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \quad (2.61)$$

$$\bar{D} = D - CA^{-1}B$$

$$\bar{A} = A - BDD^{-1}C \quad (2.62)$$

So given any linear system,

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad (2.63)$$

it can be reduced by

$$\begin{bmatrix} 1 & 0 \\ -CA^{-1} & 1 \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -CA^{-1} & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad (2.64)$$

$$\begin{bmatrix} A & B \\ 0 & \bar{D} \end{bmatrix} = \begin{bmatrix} b_1 \\ \bar{b}_2 \end{bmatrix} \quad (2.65)$$

$$\bar{b}_2 = b_2 - CA^{-1}b_1$$

This leads to the efficient solution to the maximum a posteriori(MAP) based least square solution. The only problem is that the marginalization will causes fill in effect which will lead H to lose it sparse structure. The vector $\bar{b}_2$ is commonly called coefficient vector. By rewrite the Equation 2.57 into residual covariance residual form:

$$\tilde{x} = \underset{x}{\operatorname{argmin}} \sum_{ij} r_{ij}^T \Omega_{ij} r_{ij} \quad (2.66)$$

$$r_{ij}(x + \Delta x) \simeq r_{ij}(x) + J_{ij} \Delta x \quad (2.67)$$

$$\text{with } J_{ij} = \frac{\partial r_{ij}(x)}{\partial x}$$

The ij denotes the feature first observed at state i and then tracked over to frame j . The residual error $r_{ij}(x)$ only depends on the two parameter blocks of the Jacobian at state x_i and x_j . And everywhere else is will be zero vector as seen in Figure 2.18.

FIGURE 2.18: Jacobian is sparse and only related to reference landmark i and j

The blocks of the Jacobian A and B are defined as J at location i and j and can be found by

$$A_{ij} = \frac{\partial r(x)}{\partial x_i}, B_{ij} = \frac{\partial r(x)}{\partial x_j}$$

The coefficient vector b can be found by

$$b^T = \sum_{ij} b_{ij}^T = \sum_{ij} r_{ij}^T \Omega_{ij} J_{ij}$$

Since the Jacobian is a sparse vector, therefore they can be found by sum over all states to form the coefficient vector. Same concept can be applied to Hessian matrix H

$$H = \sum_{ij} H_{ij} = \sum_{ij} J_{ij}^T \Omega_{ij} J_{ij}$$

Then for each feature constraint, the residual can be first calculated. By applying Marginalization or Schurs complement, the coefficient vector can be found by

$$\bar{b}_i^T + = \sum_{ij} r_{ij}^T \Omega_{ij} A_{ij}, \bar{b}_j^T + = \sum_{ij} r_{ij}^T \Omega_{ij} B_{ij}$$

$$\text{where } A_{ij} = \frac{\partial r(x)}{\partial x_i}, B_{ij} = \frac{\partial r(x)}{\partial x_j}$$

Then the system matrix can be found by

$$\bar{H}_{ii}+ = \sum_{ij} A_{ij}^T \Omega_{ij} A_{ij} \quad \bar{H}_{ij}+ = \sum_{ij} A_{ij}^T \Omega_{ij} B_{ij}$$

$$\bar{H}_{ji}+ = \sum_{ij} B_{ij}^T \Omega_{ij} A_{ij} \quad \bar{H}_{jj}+ = \sum_{ij} B_{ij}^T \Omega_{ij} B_{ij}$$

Then the nonlinear system can be solved efficiently by linear approximation as can be seen from the algorithm below.

Algorithm 1: Nonlinear optimization for state estimation

Result: x

Initialize all matrix;

while *not converged* **do**

$(H, b) = \text{buildLinearSystem}(x);$
 $\Delta x = \text{solveSparse}(H \Delta x = -b);$
 $x = x + \Delta x;$

end

return x

The equation can be solved in real time by g2o[150] or Ceres solver [151]

The typical residual function r for naive vision based state estimator can be found by:

$$r_c = \sum_{C_i \in [C_{i-N+1}-C_i]} \sum_{P_j \in P_i} (d^2(p_{ij}, K_i p_j)) \quad (2.68)$$

where d denotes distance measure between point P_j in camera C_i projected view and p denotes the projected points. The N denotes the number of windows size/frame. This error can be minimized by Levenberg-Marquardt algorithm or Gauss-Newton or Gradient Descent based approaches which the formulation has minor norm changes from case to case. This is directly corresponding to 3D to the 2D type of feature association which discussed previously in this dissertation. After this step, a locally optimized pose can be obtained. The overall concept is similar to the EKF based approach. But the re-linearization in the optimization process makes the SFM based method more accurate and slower

In the early stage, Hartley and Zisserman [112] deploy bundle adjustment to implement first known visual odometry which means feature from all images or subset of images are used for computation which makes it able to determine camera pose

for each frame in a video and position of the observation in a 3D manner. Visual Odometry or visual state estimation forms the backbone of the visual SLAM. The full SLAM terms has additional global optimization part to reduce the drift over time. In the latter approach, local windowed BA[152] was used to calculate real-time robot pose and 3D environment structure in a large scale environment. The visual odometry allows the drone to keep thousands of features in memory and use probabilistic techniques to compute only a few features. One of the benchmark monocular slam approaches is the PTAM [6]. They separate feature tracking for motion and structure from motion mapping into two different threads which set the example for all following works on visual SLAM and visual odometry. PTAM[6] parallel processing technique become the norm in visual odometry / visual state estimation implementation. But PTAM has many tracking problems such as the easily lost track in either non-static and near repeated pattern such as calibration board or brick wall. And also this approach was fit for small workspace setting and have difficulty in tracking in a larger area. In the latest studies, Strasdat et al. [153] shows that an increasing number of features has better improved accuracy than increasing number of frames in memory. Moreover, he also shows that bundle adjustment based technique is much better and accurate than the old filter based approach which leads filter based SLAM approach to fall out of favor. Recent publication such as VISO2(2011)[147], Semi-Direct Visual Odometry(2014)[10], Semi Dense Visual Odometry[90], Large Scale Direct-SLAM(2014)[9] and ORB-SLAM(2015)[7][84][85][86][87][88] are all directly apply minimization of re-projection error which detail of the recent paper will be discussed in case discussion part. When compared to filter based methods, SFM shows higher accuracy in computed pose and can generate more accurate 3D point clouds.

Biological inspired approach The biologically inspired method is referred to RatSLAM[154]. Milford at all employs the hippocampus of the rat to create a SLAM system which is commonly known as RatSLAM. RatSLAM can be used to generate a consistent map at a complex environment by using a monocular camera as observation source. The RatSLAM concept was later being tested at both indoor and outdoor complex environments at large scale and lasts for hours. The later study of this biologically inspired method was based on bees, ants, primates, and humans

2.2.3.4 Classify by Outputs

Generally speaking, the output of visual odometry and SLAM can be classified into a few categories according to some review studies [155]. Based on the output, they can be further classified into geometric metric odometry/SLAM, learning odometry/SLAM, topological odometry/SLAM, and marker odometry/SLAM.

Among those, Geometric Metric SLAM is most popular methods as it produces a most accurate representation of path and structure. In real life, the drone only uses its odometry generated to navigate and deploy other sensors for structure detection and path planning. For example, both MONOSLAM [5][81] and LSDSLAM [9] [90] can produce point clouds that maps accurately in 3D space. But if drone plans a path on the featureless wall, It can crash the drone easily. On the other hand, DTAM shows great potential in mapping, but map takes a long time to converge to real depth render it unusable for relative fast-moving drones.

Learning based odometry is a new topics in recent year deep learning development and it can be seen in works[155] [156] [157] [158] [159] [160] [16] [161] [162] [163]. Learning based solution is different with respect to other methods. The performance of learning odometry is highly depended on the dataset. For testing on same training Kitti sequence 1-8, some could yield a good result and some are ways away. Currently, all of those methods lack generalization and repeatable performance. In the next section, this new trend will be discussed in details.

Topological odometry/SLAM are the branches of studies that produce a semantic topological map. It can be generally considered as finding where the robot is in the connected graph nodes This topological representation[164] is first proposed by using a hierarchical description of the spatial map. It is quite important to reduce the 3D data into the essential topological map[165] using appearance-based methods. B. Kuipers [166] further show to how to analyze the perceptual map and abstract the perceptual map into a topological map. This type of SLAM is often used as loop closure tools in the geometric metric SLAM. Such trend can be seen in the works [167], [168], [169], [170]. Now the pure topological SLAM has become less popular. When the computer was not so fast, this studies was quite popular

as it has the ability to process in real time. As computational power grows, it becomes less necessary for pure topological approaches.

Marker SLAM is the type of SLAM that works in half known environment. It works partially like detecting loop closure in geometric SLAM. But with the marker itself, the drone can find its position and map(based on prior knowledge). One way to Imaging about this concept about this is based on current street view from Google map, find the maximum likelihoods hood of your current position. It assumes global knowledge and repeatable traveling path which usually wont happens for drone based navigation. Since it is less compatible with this thesis, it will not be discussed in details.

2.2.4 Current Trend - The Neural Network Based Approach

The neural network-based method has been theorized for a long time. There were many groundbreaking works before this centuries, but the idea of the end to end learning based approaches wasn't popular. It is only when the Nvidia allows the usage of the parallel processing library CUDA and AlexNet proof that it can perform significantly better than previous methods, the convolution neural network becomes the most critical approaches. Ever since 2012 ImageNet competition, the convolutional neural network becomes an exciting topic in the research community and paper[26] quickly spread over the research communities. Right now it is a general understanding that visual learned features are generally better than handcrafted features. The accuracy overall can improve over 10% for most of the applications. But for the Visual Odometry case, the improvement is sometimes hard to reach 10% and sometimes more prone to error when dataset changes. Overall speaking this type of research can be classified by inputs, outputs, and models.

The input layer of the convolutional neural network-based Visual odometry consists of one or more information from (1)the current image, (2)the present optical flow,(3) semantic map, (4) current depth, (5) current depth flow, (6) the current point cloud from lidar,(7) the current point cloud flow from lidar,(8) IMU and (9) the sequence of information containing information from one or more source from (1)-(8).

The outputs of the current network can be consists of one or more data from (1) Odometry (Twist) updates, (2) Pose (PoseWithTimestamp) updates (3) semantic label map, (4) optical flow, (5) Current depth and (6) one or more information from (1)-(5).

There are thousands of ways to make it a new network by combining different inputs and outputs. There also exist difference in whether the data to the neural network is synchronous or asynchronous.

The model itself changes time to time. But for a large percentage of cases, the visual feature from the convolutional block are transferred into the new model by a technique called transfer learning. The backpropagation is only trying to optimize the fully connected layer for the output. The reason is that the visual feature learned from a small odometry data set is not representative when compared to the kernel learned from millions of training data in Imagenet.

Therefore for this new type of approaches, there are more possibilities than traditional methods. And there are already dozens of work published using this transfer learning concept.

2.2.4.1 Basic Building Elements and Operations

The basic building elements for neural network-based approaches is called neuron and are like LEGO blocks. A group of similar neurons forms a layer. For each layer, it has properties such as its type, input size/shape, and output size/shape. Layers can be cascaded and parallelized. For link weights to each neuron in a layer can be updated by backpropagation to optimized for training label.

The most fundamental layer is called a fully connected layer or inner product layer. It takes multiple input floats and weights and computing a sum and outputs the summation. The fully connected layer is one of the earlier work which could not directly be implemented to understand an image. The reason is that image is pretty large data which has MXN dimension and read in sequence will lose the spatial information. Therefore for a long time, The neural network-based image application is lesser popular approach and there were fewer published works[171] compared with the traditional method[172].

The most recent breakthrough is the convolutional layer. The theory was considered to be published in the early 1990s, and it was named time delay neural network[173]. It was popularized after 2012. It maintained the spatial information and usually in the form of edge or blob detector. The convolution layers can be implemented spatially or temporally. Generally speaking, the convolutional layer downsamples the data at each path until it reaches into a singular value which means it can not be further downsampled. This conventional layers are usually be considered as high-level features and contain the most of the information. The fully connect layer follows traditionally meant for classification or regression.

The deconvolution layer or Transposed convolution layers [174] upsamples the data. It is the counterpart of convolutional layers. The deconvolution layer maintains the data contents and usually performs better than standard interpolation based upsampling. This idea is widely used in semantic segmentation or depth map estimation.

The other most commonly used layer is called polling layers[175]. It shrinks down the image size to reduces the data size. It is an essential tool to control the over-fitting problem. The most common pooling method is max pooling or average pooling. It finds the max or average value in a specified size and outputs. The total number of data reduced is square of the size parameters.

The last but not least is the recurrent neural networks (RNN) or long short-term memory(LSTM) by the works from. It allows the network to extract temporal relations. It is an ideal tool for tasks such as odometry as small baseline tracking often suffers from more observational error. Concatenate the tracking to form large baseline can produce more accurate results. The most recent studies in this area are echo state networks(ESN). ESN differs from the normal RNN or LSTM by dropping the connection to make the network sparsely connected. It exhibits more similarity to human cognitive function. However, when minimized the test error, the network turns out hard to regularize. It means that the network may not have a unique lowest point.

There are many other layers such as the normalization layer, spatial pyramid pooling layer, etc. The use case of those layers is often case by case. At the last of the

networks, it always follows a loss layer. The loss layer contains the error measure and optimizer. Loss can be first order or second order. The optimizers usually closely coupled with the problem and loss function. The explanation of different optimizers, loss, and the combination of optimizer and loss requires a series of dissertations to explain fully and it will not be explained in detail here. The most commonly used case is the mean squared error and Stochastic gradient descent. The next important concept is Backpropagation. Once the error is computed in the last layer, it has to be passed to the previous layer to update the weights of the connection. Once the weight has been updated, the network is feeding forward again to compute a new set of errors.

2.2.4.2 Current popular pipeline

The current pipeline follows the general guideline as this review mentioned earlier. One of the earlier work[176] can be traced back to 2015 where most people just begin to realize the potential of CNN. They use a batch of 5 frame stereo to extract motion information. And they[176] use the same CNN model separately to extract translation and rotational movement. This work sets the general direction for the future works.

FIGURE 2.19: The pipeline of deepVO[15].

Then Wang et al. [15] uses an LSTM to properly encodes the motion information. In this paper, they demonstrate the problem of overfitting in the network training process. Their work can partially replicated but not with so many good case . The model used can be seen in Figure 2.19. The LSTM or RNN is closely resemble the

multi frame bundle adjustment process. The result was a huge improvement over previous work[176].

The most recent work[160] demonstrate a joint depth and odometry estimation. By using depth to recover absolute scale, the result can be exactly like the ground truth. But the model is not revealed. So there is little we can verify their work.

Another important direction is to fuse multiple sensors such as visual and inertial information. This direction can also be find in traditional methods as most of recently proposed vision based odometry [73][45][96][12][97] [98] also need inertial measurement for robust performance. Ronald etc [16], proposed the Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem(VINet) as seen in Figure 2.20. They demonstrate the possibility of fusing both learned visual odometry and unsynched IMU data by an LSTM based network. But same as other work, we are not able to replicate their work.

FIGURE 2.20: VINet, illustration figure from[16]

2.2.4.3 Outstanding problem and challenges

Right now it is safe to say, none of the learning based approaches can be safely transported to generalized drone navigation tasks as can be seen from Figure 2.21. Usually, the performance is well when the test data is precise as training data as shown in from Figure 2.21. The performance can be as good as traditional methods. Once the trained network is being fed with another sequence which captured using the same hardware. The results fail. When using another different capture device, the result became non-sense.

And the other challenges are the published result are hard to replicate. For a convolutional neural network + LSTM based visual (inertial) odometry, there exist

FIGURE 2.21: Test on Training sequences and test on testing sequence using DeepVO with pre-trained weight on Kitti sequence id 01 02 05 08.

hundreds of hyperparameters which in the paper is usually ignored. None of the original deep visual odometry paper releases the pre-trained network. Therefore its hard to replicate the experiments for a fair comparison.

2.3 Obstacles Detection

2.3.1 Introduction to obstacle detection problem

Obstacle detection is a crucial factor in any of the navigation solutions. Generally speaking, autonomous cars can afford heavy and expensive Lidar without any limit. For drones, weight, power, and size pose a heavy restriction on what can be used as can be seen in figure 2.22. This DJI M210 weights 5kg and can have a maximum takeoff weight of 6kg. It only has 1 kg for its payload, whereas a typical 16 line Velodyne Lidar is about 1kg.

FIGURE 2.22: An illustration of drone obstacle detection. Image provided by DJI

In this section, various way of getting obstacle information will be discussed. In which, the vision-based method will be emphasized in details. Ideally, vision can be used for every single task such as detecting an object as seen in Figure 2.23, finding traffic signal, inspection or even track its own motion as seen in Figure 2.24. However, vision-based tasks can fail very easily and often. With a minor change in noise as seen in Figure 2.23, light in Figure 2.24, weather conditions, the result would be entirely different. Even use the state of the art learning system, it still remains unsolved. Most the cases, there has to be a backup plan for detection and avoid obstacles.

FIGURE 2.23: An illustration of wrongly detect and classified object due to noise. Such noise often present in camera as shown in Figure 2.24. Image from Internet Source

FIGURE 2.24: An illustration of wrongly tracked pose due to lighting change. Image taken in 5th floor at longitude 1.291328 latitude 103.786790 facing north direction. Both shutter speed is set to be 15ms for both case. Note higher shutter speed leads to motion blur, top limit by this dissertation is 15ms

2.3.2 Obstacle detection physics

Most of obstacle detection sensor works on a specific electromagnetic frequency band in the electromagnetic spectrum as seen in Figure 2.25 with exception of ultrasonic. Each frequency device has its merit and weakness.

FIGURE 2.25: An illustration of frequency spectrum. Image from Wikipedia

In radio wave level, there is radar with a various band. Among which, Synthetic aperture radar is a type of device that can generate depth image and it is widely used in the military. Moving closer to Infrared level, there are multiple devices constructed using active sensing Infrared beams such as Light Detection and Ranging (LIDAR) and structured light depth camera. In the next visible light level, almost all device is passive such as the camera. It is one of the most widely used band of spectrum as most of the animals operate in this spectrum as well. The higher energy Xray and gamma-ray spectrum are usually used for penetration testing and diagnosis in the clinic and rarely uses in obstacle detections.

2.3.3 Radio spectrum

When the wavelength at radio wave level, the sensing device is usually called Radar. There are families different band of radar for different purposes. Radar is generally more powerful and can detect long range. Among which Synthetic aperture radar (SAR) is a special form of radar that can generate 3D depths as can be seen.

It still impossible to carry good quality SAR on a small drone. But there do have cases that drone uses satellite generated SAR 3D map to localize and map as can be seen in works[177], [178] , and [179]. This type of work is usually using drone

image for localization and odometry. And based on the odometry and pre-recorded SAR map, it can be used for intercity level navigation. The problem is when there is a room which SAR did not map beforehand, there is a high risk of crashing.

FIGURE 2.26: An illustration of Small size Synthetic aperture radar depth with top down view. Image from research work[17]

Normal radar has massive size and not design for small size drone to carry. Therefore it will not be discussed further. In recent works, there have been attempts to use small size drone to make SAR depth image[17], as can be seen in Figure 2.26. But according to the latest classified test by the anonymous party in Singapore, such a product cannot detect another drone in flight. Only when the target drone carries a 15cm Luneburg lens [180], then it can be detected. Compare the radar version with Vision or LIDAR version, Vision or LIDAR is generally preferred as can be seen in Figure 2.27.

FIGURE 2.27: An illustration of stereo generated depth with top down view. Lesser PNSR noise when compared with SAR output. Image captured in-house

Instead of using radio waves, ultrasonic sensor use sound wave to echo measure distances. It is also widely used in robotics research [181][182]. However, because

it is using the wave pattern, the resolution of sonar is low. Moreover, the sound is affected by various factors such as temperature, pressure, and material. In actual operation, the ultrasonic sensor is used for landing. However, to construct multiple sonars to form an array is sometimes counterproductive as they do interfere with each other.

2.3.4 Inferred spectrum

Moving to shorter wavelengths is the long wave Infrared spectrum. In this spectrum, the solar inference is lesser than near visible light inferred.

FIGURE 2.28: Typical 1D 2D 3D lidar from left to right and their applications. Sensor image taken from Google

At 9k to 10k nm, this is typical LIDAR operating range. It offers the most robust measurement to the depth and therefore is widely used in autonomous cars. There are hundreds of studies using this level of EM spectrum. The most famous work is done by Censi [183]. This work uses iterative cost point matching to find robot position and code is published in ROS. This spectrum can be operated during sunlight. The LIDAR measures time taken to hit an object and bounce back to

measure object distance. There is various type of LIDARs such as 1D, 2D, and 3D type. The more dimensions it can sense, the heavier the Lidar becomes with 1D weight about 10+ gram, 2D weights about 130+ gram and 3D weights about 1000+ gram as can be seen in Figure 2.28. Generally speaking, 1D and 2D type are commonly used for small size UAV. 3D type Lidar is only available to large size drone and usually requires a permit to fly.

The LIDAR is not all mighty without fault. Currently, LIDAR cannot work with fog, rain, snow, water pool on the ground, glasses or event presence of another LIDAR. All of those limitations is what fully autonomous car company trying to solve.

Moving closer to visible light is near infrared spectrum. A lot of structured light depth sensing devices are constructed using this spectrum such as Microsoft Kinect, Asus Xtion Pro, and Intel real sense. In the survey conducted by Schning and Heidemann [18], most of this sort of depth-sensing camera is cheap and less reliable outdoor as can be seen in Figure 2.29. This spectrum is easily affected by solar interference and therefore rarely used in the actual outdoor platform for robust navigation.

FIGURE 2.29: Typical outdoor performance by IR based RGBD camera. Green denotes missing value. Image from research work[18]

2.3.5 Visible light spectrum

The last candidate is the visible light spectrum. This spectrum offers most flexibility than any other spectrum. In this spectrum, data gathered can be used for obstacle detection and even object detection/classification. Typically, A technique called motion parallax is used to obtain depth information. Motion parallax is an effect that when the viewpoint is moving, the closer object appears to move

faster. It is one of the fundamental visual cues to perceive 3D space or depth in short. It is often used in monocular visual odometry/SLAM for triangulating 3D points. There are other monocular cues to find depth information such as infinity line/points of perspective projection or relative object size. But the others form of observation requires certain environment setting whereas the motion parallax requires the only a marginal motion.

When there is one camera, it is called motion stereo. When there is a pair of synchronized stereo to form a constant motion parallax, it is called stereo vision. For monocular camera based depth estimation, it offers a lightweight solution and elegant design at the cost of lower robustness and higher computational cost as can be seen from. The stereo way of depth estimation is more accurate (within 100 times of the baseline) at the cost of the additional mechanical structure. It also worth mention that, pure motion parallax have a problem in detecting a front obstacle with pure forward motion.

2.3.5.1 Monocular case

The monocular way of estimate depth is ideal for lightweight drone as it only requires a camera and a processing unit. But it is lesser popular and robust for obstacle detection due to numerous limitations on motion and environment. The first work can be carbon dated back to 90s by Sandini and Tistarelli [184]. Then soon after that, the new works[185][186] quickly forms the basis of SFM as the world knows today. Saxena et al. [187] proposed the first learning based monocular depth estimation. Although MRF based method[187] takes very long to process and less robust, it still sparks the following learning-based approaches. And in recent couple years, learning based monocular approaches become mainstream studies as it overcomes many limitations in the monocular cue and uses previous knowledge in estimating depth. The impact cannot be understated. When Andrew Ng sparks the learning based approaches, the following works[119][188] come up with another milestone in the SLAM world in the same year. The inverse depth parametrization for point features allows uncertainty of triangulated depth to be managed over time. This is key work before works such as DTAM[8] or LSDSLAM[9] to

be possible. In the next couple years, the theory of inverse depth parametrization has been refined and widespread by works[121] [120][189]. In the same time, new SLAM method[190] refines the previous monocular learning based depth estimation and including stereo cues in estimating depth which improves the result and robustness. The direct estimation method[191] combines initial depth prior, the dark channel prior and multiresolution analysis to generate depth in multi-resolution. The multi-resolution map is fused into a singular depth map. There is another series of work[192][193][194] which estimating depth from focus.. When compare defocus-based approach with motion parallax based cue or learning based approaches, defocus-based approach is less accurate and less robust. This series of work receive a lesser amount of attention the more recent one can only be seen in [195][196]. After 2012, the learning based method become dominant force. Such trend can be seen from series of works[197],[198], [199], [200], [201], [202], [203], [204], [205], [206], [207], [208], [209], [210], [211], [212], [213], [214]. Before 2015, number of monocular depth estimation works is less than 10 piece each year. In 2016 and 2017, this number goes to 20+ which mostly consist of learning based approaches. In 2018, this number goes to 70+ and counting. Where most of the work in the recent year cannot be directly transported to drone uses, works [215] shows on paper much better performance than other methods such as works[216] [9]. They use sparse patch tracking and gradient enhanced depth mapping to create the state of the art performance. The idea is similar to minimize the uncertainty of depth on a strong gradient in LSDSLAM approach. However their code was never released, and there is no one confirms the performance afterward. So far the best performance of open source implementation is still REMODE [216].

When talking about monocular detection and navigation on a drone , lesser works is presented. Earlier work[217] [218] [47], uses PTAM with IMU on a low cost UAV to navigate around. The map created contains sparsely populated obstacle information which is not sufficient for the drone to make the smart decision. A similar approach can be found in the work[122] where they use inverse depth concept on sparse feature points. The improvement work[219] uses a laser to help improve the depth and achieve relative autonomous flight. The more intelligent method can be found from the work by [123] where they use inverse depth concept to generate converged features. Those converged features are mapped into a persistent terrain map for obstacle avoidance. More recent approaches[220] use analysis of

expanding convex hull from mapped feature points to find dense mapping information for obstacle avoidance. Essentially this work is directly added analytical solution[217][218][47] to find the correlation between mapped features, assuming mapped features holds planner constraint and then avoid it.

2.30;

FIGURE 2.30: Comparison between VOXBLOX(left) and OCTOMAP(right) map representation. Image taken from corresponding paper[19][20]

The other popular approach that adopted by works[79][221] is the Truncated Signed Distance Fields(TSDF) method from VOXBLOX[19]. TSDF is used to recreate the environment and then transformed into Euclidean Signed Distance Field (ESDF) for path planning by the grouped raycast method. When compared with previous mapping methods, The OCTOMAP[20] stores the probability of an occupied cell and it is most of the time insufficient for path planning. Most of the time, path planning requires direct distance information towards the obstacle. That's why a lot of time ESDF is preferred as it directly gives distance towards obstacles. VOXBLOX has currently beaten other mapping method such as OCTOMAP[20] by having faster map creation speed. However, VOXBLOX requires the preset size of the environment. VOXBLOX also allows more dense and smooth representation of obstacle surfaces at the cost of takes huge memory spaces and more suitable for GPU. In this thesis, OCTOMAP is primarily used for compatible mapping from both 2D/3D LIDAR and stereo/RGBD camera and free navigation without boundary limits. The details of the difference can be seen from Figure 2.30.

2.3.5.2 Stereo case

Stereo vision is a process that a pair of the image is being captured synchronized at a pair of known 3D position. By measure the difference in pixel location, the depth information can thus be obtained. This technique is widely used in robotics

and has been studied for decades. The stereo vision problem is a kind of problem which is easy to define, but not that straightforward to solve in real life.

According to the taxonomy by Scharstein and Szeliski [25]. Stereo vision can break down stereo vision problem into four parts (1) Obtain raw matching cost, (2) Cost aggregation, (3) Stereo disparity computation and optimization and (4) stereo depth map post-processing.

When machine consider whether a pair of pixel belongs to the same thing in real life, the machine can access to only its intensity difference as proposed in the paper[222]. In an image of millions of pixels, simple intensity difference can result in sizable false match area. Then researchers use more adjacent information such as neighborhood pixels to image blocks[223]. After that, people use the more advanced feature and descriptors to measure and link pixel blocks in a more comprehensive way. Such trend can be seen in the works[224][27][225]. At last, people find the final frontier. Research works[28] uses the deep CNN learned feature to describe everything. CNN feature is the feature to solve all feature problem at the cost of high computational demand. There is another series of work[226], [227], [228], [229], [230], [231], [232], [233] which use local binary pattern or census transform for faster processing and real time performance. Majority of stereo done in this thesis is using the real time method[229]. Census transform is generally speaking faster in finding raw costs. It suits best for light weight and low computational power UAVs.

Cost aggregation is one of the important steps in the taxonomy by Scharstein and Szeliski [25]. Early works[223], [234][235] use a simple fixed windows summation approaches to produce clear and more distinctive cost. In the later attempt, weighted summation methods make uses of color similarity and pixel relative location to get better cost value. Such trend can be seen from series of work[236][237]. The other way is to sum over a block of pixels with similar intensities [238][239]. Hirschmuller [240] uses sum from all direction based approach. The aggregated cost can be found by summing the costs of all 1D minimum cost path which goes through the target pixel. More recent approaches[2] use cross based approach to construct adaptive arbitrarily shaped 2D aggregation windows for optimal performance. The latest cost aggregation adopt a biologically inspired multi-scale approach [29].

The actual Stereo disparity computation and optimization process are more easy to define but hard to make perfect. Stereo matching methods can be classified into the local approach, semi-global approach, and global approaches. The local approach is adopting a winner takes all method by finding the lowest matching cost and it be seen from the work [234][236]. This method is the fastest method but usually with a lot of wrong depth. The global method typically uses a global energy function similar to the variational energy function. These type of functions typically contains a data term and smoothness terms. With this method, certain shape and object in the image can become the constraint for matching. In order to remove occlusion and other problems, new methods[235][241] deploy energy functions with more terms. Other typical uses of extra terms are enforce a left/right or symmetric consistency check between images from the work [237][239][238]. There are also other approaches[242] which designed to actively use image segmentation technique to ensure large models to have perfect fits. These methods can also be seen in the stereo depth post-processing technique The steps for finding global minimum changes from approaches to approaches. For example, early dynamic programming[222] [243] perform optimization in 1D for each baseline. This 1D approaches usually ends up in thing lines of false matched depth. Then tree based dynamic programming[239][244] are introduced to solve thing lines of false matched depth. There are other direct 2D optimization methods such as graph cuts [241] or belief propagation [223][237][242]. But no matter which method, globally optimized stereo usually takes much longer times to process. Generally speaking, the robot will run into the wall first before first depth can reach global optimal depth.

In between local and global approach, there is the latest addition namely semi-global approach (SGM) [240]. SGM uses a fast approximation method by doing multi-direction based optimization. Later work[28] uses a similar concept but lesser direction to achieve a similar result with full 16 line approaches.

Stereo depth post processing is one of the last touches up in getting a reasonably good visual obstacle detection output. Post processing includes but not limited to remove false positive output, fill occluded area, check left and right depth consistency and sub-pixel interpolation. Early work[245] uses analytical methods for both remove false positive and check the depth consistency. More recent solutions[246][28][21][247][248][249][250] uses series filters and consistency checks to remove wrong values. The innovative method[251] uses strong matching points as

supporting anchor to get the more dense result. Hole filling is one of the most commonly used tools to remove the false positives. Works[247][248][249] demonstrated smoothed output when under smooth constraint. However, those approaches usually end up with lots of wrong depth value. The most recent one[21] discuss dozens of conditions in detail to achieve better quality. This method is similar to learning based method. They study various case of failure and put a patch to it. But for the tough case shown in other work[28], the manually patched method Abarghouei and Breckon [21] also cannot handle for all the extremes in the real life, as shown in Figure 2.31. For a ground drone 2 DOF driving, such false value will not cause any crash. For UAV full 6 DOF flying capability, this wrong depth value can kill the drone to have sudden halt and cause violent control behavior which leads to odometry lost. The detailed state of the art introduction and patched method[21] failure is included in section 3.2.

FIGURE 2.31: First row(left to right): Input, Input, Second row(left to right): Real time stereo output, DepthComp[21]

Chapter 3

Multiple Domain Analysis based Stereo Depth Map Post-Processing for Autonomous Real-Time UAV Navigation

3.1 Introduction

Binoculars vision or stereo vision has been widely used in various tasks from Mars rover to little toy such as the DJI Mavic drone. There are several autonomous driving companies such as Tesla and Waymo dedicating resources to developing the vision-based solution because of the low cost. Small sized drones cannot carry heavy weight which makes vision an ideal choice for the drone navigation.

However, there is a problem when the depth map is full of missing or wrong values. Some of the values can be a false reading telling that there is an obstacle right in front of the drone and the drone has to stop for conservative navigation. One can use more aggressive navigation method such as Uber and Tesla. But they crashed and kill people as shown in 3.1. Both of them is running on vision only or vision heavily. For the uber they do have LIDAR and LIDAR detects the Elaine Herzberg[252]. But at obstacle output level, they trust more on multiple stereos. And stereo has false detected regions which they removed it by force and navigate it with aggressive navigation method. It is believed to be the root cause to this accident. The Tesla case was more well known as they failed to detect the truck and decapitated the driver.

FIGURE 3.1: Left: Uber L5 Navigation fatal accident live footage[22], Right: Tesla L3 Navigation fatal accident analysis[23].

Such case can be seen from both dataset and real-world testing samples in Figure 3.2. Real world testing has more challenging cases than the dataset. For a real-time stereo method, it is easy to encounter cases where there are white noises as seen from the top in Figure 3.2. Even for the state of the art post processing learning based stereo, such faults also could happen. When left unprocessed, the noise will

then be fused into a 3D navigation map as seen from the bottom figure in Figure 3.2. In this navigation map, a drone may never find an optimized path to get to the target position. There are also high possibilities that the drone has to circle around to get to the nearby target location as if it is trying to avoid an obstacle. Or the drone has to try its luck and aggressively navigate it through which lead to the a similar crash[22][23].

FIGURE 3.2: The problem of current stereo based drone navigation

More often, the obstacle detection module has to be turned off to fly. This action will easily end up in a crash[22][23].

This problem is widely seen in real drone flight. For KITTI[31][253][254] dataset, there are few people who address this problem. The KITTI dataset focus more on autonomous driving instead of UAV navigation. Therefore their top sections are cropped and without any ground truth measurement. The removed parts are normally more prone to depth error. The resulting resolution in KITTI are 1242X375 (16:5). For an autonomous drone to robustly navigate, more observation is needed from top and bottom visual field to find a proper 3D path. This is the embedded assumption in both odometry and navigation. Most of solutions [90][9][10][80] use a wide angle camera to solve robust tracking problem. That kind of tracking camera is usually in 640X480 (16:12) or 752X480 (16:10) resolution with more than 100degreeX80degree lens. The cost is that the wide-angle camera can cause more false matched area. In such case, it is very likely the drone will never find a free or a short direct path to the target locations.

To solve this problem, researchers have tried many models or ways to improve the result. Much of the recent works is done by getting better cost function/descriptor to better matching accuracy. Even with the state of the art slow deep convolutional neural model(CNN) [28], the faulty matching and false detection problems are still there. The problem is at some extreme lighting condition or uncommon image configuration, there is just no feature to match. The harsh conditioned patches in the image can come with various shape and size which is nearly impossible to address as a simple matching problem. For the KITTI dataset, it's still possible to use onboard LIDAR to form certain constrained pattern for stereo depth. For a drone, it less likely to carry heavy equipment and form such constraint patterns. Therefore this problem has no alternative hardware solution. Some researcher[21] went to new level to get better post-processing result by discussing various failure conditions[21]. It can be seen that improvement is marginal as can be seen from Figure 3.3. And that work can not be used in real navigation tasks. Therefore, we proposed a new multiple domain-based analytical solutions to this problem. The result can be seen from Figure 3.3

FIGURE 3.3: Proposed method compared with most recent work[21]. Left is the input census transform stereo raw depth[24].

3.2 The State of The Art

The obstacle detection is one of the fundamental problems in robotics and computer vision studies. Among all the obstacle detection methods, stereo vision is considered to be a type of problem that is easy to define but difficult to solve as can be seen in Figure 3.4.

FIGURE 3.4: Biological model turns into Mathematical model

The difficulties come from the effective matching of left to right patches and then find a good smoothed depth image. To achieve those two step. The stereo vision problem can be broken down in four sub-steps as shown in Figure 3.5.

FIGURE 3.5: Stereo vision problem in a nutshell. Image from dataset[25]

There are many efforts to get better stereo depth output. Those efforts can be found in each branch of the four stereo sub-problems. Amongst them, the stereo method is most difficult to improve. The feature cost can be improved by using the more extensive and distensive feature at the cost of increasing computational power. Post processing is another new frontier where depth map can be even more improved

3.2.1 Feature Cost

The stereo result can be improved by using more complex raw costs[28] [240][224]. Such a trend can be seen from Figure 3.5. In the early 90s, pixels and blocks are used as raw cost. After 2000, when SIFT got popular, there are many research use SIFT and SIFT variant as raw matching cost. After the 2012 ImageNet competition, the deep learned feature becomes more popular research topic as can be seen in Figure3.6.

FIGURE 3.6: The evolution of finding raw cost. Image source[26][27]

The current state the art is mostly from deep learning based approaches. Among the open modeled publication, [28] is considered the lead in this series by 2018. They use deep learned features as raw matching costs. Then the cross based aggregation is used to further combine and enhance the uniqueness of generated in-depth learned features. A semi-global matching technique is used to find raw depth map. After that, the result is interpolated. At last, a series of post-processing technique is used to enhance the depth map. The enhancement technique includes Left-right depth image consistency check, median filter, and bilateral filter. The overview of their system can be seen from Figure 3.7. The details of each sub blocked will be introduced in later sections of prior work.

3.2.1.1 Their Contributions

A deep learning model is proposed to learn the similarity feature from two patches in stereo image pairs. The proposed similarity measure has a more distinctive feature than a traditional hand-designed feature. It is then combined with other popular stereo sub-methods and demonstrated it can get better performance. The result of their work can be seen from Figure 3.7 which is far more better than any other traditional stereo costs.

The Current State of the Art Learning-Based Stereo

FIGURE 3.7: The State of The Art Learning Based Stereo. Image source[28]

3.2.1.2 Their Limitation and Proposed Solution

Currently, they have three limitations namely (A) Slow processing speed, (B) Lack generalization, and (C) still have the imperfect area in depth map. For (A) and (B) they are typical deep learning problem. To generate a stereo pair, they need 67 secs on the Nvidia Titan X. And it is well known that deep learning based system suffers from generalization problem. Even added a few Gaussian noises in the image can result in total failure. Such Gaussian noise can be commonly found in normal CCD/CMOS cameras. There is little thing, people can do to effectively solve them.

For the false positive and false negative area in depth map, this can be solved by analysis on data and create a detection algorithm to find such false positive and false negative area as can be seen from Figure 3.19 in the proposed section.

3.2.2 Cost Aggregation

Cost Aggregation is a stereo process which sums up the raw cost and makes a particular pixel matching cost more distinct. The idea of cost aggregation over past 20 years can be summarized as

- (1) Increase support block size
- (2) Increase number of block per stereo pixel
- (3) Add weight to each block
- (4) Use the adaptive shape of support region
- (5) Multi-scale Cost Aggregation.

Most of the work can be one or more combination of the strategies as mentioned above.

The current state of the art is the series of work by [2]. By a set of similarity measures, the set of left to right and top to down supporting region can be found as seen in Figure 3.8.

FIGURE 3.8: Current State of The Art Cost Aggregation Basis. Image source[2]

Then supporting area are summed to form argument costs. In their recent work [29], a multi-scale cross based method is proposed by checking the multi-scale argument cost consistency based on their previous cross bases cost aggregation as

shown in Figure 3.9. This method takes local area results at multiple pyramidal level for better matching accuracy.

FIGURE 3.9: State of The Art Cross-Scale Cost Aggregation Basis. Image source[29]

3.2.2.1 Their Contributions

Their contribution is (1) using an adaptive generated local cross pattern to find a more distinctive matching cost. (2) using of a multi-scale case to simulate human-like vision system to have a more robust stereo matching process.

3.2.2.2 Their Limitations

The accuracy is high throughout the testing sequence but the computational cost required is very high. For the Middlebury dataset[25], the proposed method requires 0.11sec to 70 sec to process. This method can not be easily turned to a real-time stereo solution.

3.2.3 Stereo Matching

The state of the art for stereo matching is still considered to be semi-global matching[240], as can be seen in Figure 3.10.

FIGURE 3.10: Detailed Method for Semi Global Stereo Matching

Earlier approaches find the lowest matching cost along the epipolar line. The disparity d of the lowest matching cost will be considered as the depth as seen in top part of Figure 3.10. However such method will create the streaking effect and other uneven surfaces. The semi-global based method consider more than one scan line. For each pixel in the left image, a 2D vector of matching cost to the right image can be found. By finding the difference/cost along the one of the path e.g path 1 , A 2D matrix can be formed. Then a weighted sum(in our implementation) of the lowest item $S(p, d)$ in each column can be done to find the approximate matching cost value. By adding other penalizing term, a smoothed surface along direction selected path can be ensured. The mathematical expression for such a term can

be written as:

$$E(D) = \sum_{q \in N_p} P_{smt1} T[D_p - D_q = 1] + \sum_{q \in N_p} P_{smt2} T[D_p - D_q > 1] + \sum_P C(P, D_P) \quad (3.1)$$

A similar method can then be used for path 2 to path 16. When other directions are introduced the streaking effect will be removed. By finding the minimal disparity number in all directions $S(p, q)$, a semi globally optimized disparity can be found. By using this method, the plane structure are maintained. In recent work, there are different variants of implementations such as 3-way, 4-way, and 8-way scan pattern to ensure local consistency.

3.2.3.1 Their Contribution

In the paper[240], the most important thing they proposed is to add semi-global consistency parameter in the cost aggregation stage. The added part introduces a new smoothness constraint that is typically shown in a global optimization technique. The speed cost is higher than older Scanline Optimization method but much faster than a global method such as The GraphCut

3.2.3.2 Their Limitation and Solution

This method is still regarded to be the best. It ensures smoothness transition in between adjacent pixels. The original work uses too much directional cost which makes it run very slow. However, there exist solutions with lower amount of scanned path to ensure fast performance. The result of lower amount of scanned path is generally less accurate than full path. The other way to speed up is to add GPUs.

3.2.4 Post-Processing

Seungmin Choi [246], Zbontar and LeCun [28], and[240] applied left and right depth image consistency check (LRDCC) as a golden rule. There are some variant such as incorporating with a median filter, hole filler or bilateral filter to ensure a local smoothed result as shown in Figure 3.11

FIGURE 3.11: One of the Common Approach For Stereo Post Processing Using Left Right Depth Consistency Check

Let $d_l(x_l, y_l), d_r(x_r, y_r)$ denote the left and right corresponding image. Where x_l, y_l and x_r, y_r are matched points on left and right image image coordinates. The resulting $d_l(x_l, y_l)$ will be

$$\dot{d}_l(x_l, y_l) = \begin{cases} d_l(x_l, y_l), & \text{if } \|d_l(x_l, y_l) - d_r(x_r, y_r)\| \leq \tau \\ 0, & \text{otherwise} \end{cases} \quad (3.2)$$

where τ denotes the maximum difference allowed in two depth. $\dot{d}$ denotes LRDC checked depth. This is effective against occlusion and many other false matches. The overall direction for this sort of removal is correct. By finding corresponding depth value on the left and right image, a common understanding can be formed. The LRDC method can help find the majority of the stereo occlusion and part of the local false matched area. However, for large saturated area of false disparity, this method cannot clear them as shown in Figure 3.12. The method used in Figure 3.12 deploy the state of the art of feature cost, cost aggregation and matching method. But the error can still exist there.

Therefore, LRDC based method still can't cope with the real-life challenges. In proposed sections, the cause for this error will be discussed in detail as shown in Figures 3.17, 3.18 and 3.19.

In most recent proposed work [21], Abarghouei and Breckon [21] extensively discuss about 12 cases where the stereo goes wrong. The top 8 cases can be seen in Figure 3.13. From the discussions in his work, the idea sounds correct. However, in reality, the environment is much more complicated than what they theorized, as can be seen from the Figure. 3.14

FIGURE 3.12: Result of Left Right Consistency Check Fails with System Level Error. The error is still present after the check

FIGURE 3.13: One of the Current State of the Art of in Stereo Post Processing. Image source[21]

FIGURE 3.14: Result form the benchmark[21]. Fails with strong error pattern

3.3 Proposed Method Technical Description

The proposed method can use together with a real-time stereo algorithm(eg. Census transform with Binary Semi Global Stereo Matching on GPU) to achieve high correctness and real-time capability. For real-time methods, there usually wont have deep learning feature, complicated cost aggregation or 16 path SGM. The method only includes Census transform and four way SGM only. With the help of the proposed method, the resulting 3D map can be directly fused into global map for navigation tasks.

3.3.1 Stereo Depth Map Post Processing using Multi Domain Analysis

FIGURE 3.15: An overview of enhanced depth map post processing system flow

The proposed method can be broken down and paralleled into three cores/process/machine for processing, as shown in Figure 3.15. From Image domain, the variational cost is used to determine a particular pixel that has a probability of causing incorrect depth. The process is very intuitive. If an area is mostly dark or bright and lack of feature, then there is a higher chance stereo cost will not work on this region.

From the depth domain, iterative snow piece prediction is used to find out the most likely area of the incorrect depth. This domain analysis is the most critical part

because snow piece on the depth image will cause the drone to have the capricious behavior of trying to dodge phantom ghost. The second order derivative of the image is found to determine those areas which potentially have an error measurement or occlusion. This part usually won't cause trouble for navigation and drone can always move closer to update. Based on the previous 2 steps, two probabilistic maps are obtained to determine which part is undoubtedly wrong and which part is imperfect. Based on the probabilistic map, different level of missing value filling or inpainting is performed to enhance the depth map.

Even with complete image-level processing, there could still have imperfection and wrong detection. Those detections usually lie on the cloud and sky. To UGV, it won't cause trouble because the sky section is cropped. However, for UAV, it is an enormous problem as part of the path can be blocked because of that false matching. The incorrect matching part could sometimes look like to be ordinary stereo cases with depth gradient, thus making it hard to use traditional image processing techniques to remove reliably. That's where semantic domain comes into play. The semantic label generated will provide enough knowledge clue to eliminate the error depth. The resulting flow from each process can be seen from Figure 3.16.

FIGURE 3.16: Overview of PPMDA. Image is from the KITTI dataset 2011 09 26 drive 0035 sync sequence

3.3.2 Image Domain Analysis

Image domain analysis is to provide the cue to stereo on when a depth on a particular pixel should be trusted and when should this depth be considered with large variance.

Understanding What Goes Wrong To understand when can stereo be used for navigation, one has to understand a series of cases that stereo cannot work well. As can be seen from Figure 3.17, depth error is directly correlated to the intensity value. When intensity saturates, there is a high chance that depth generated will be incorrect. This case is often seen when a camera 's dynamic range is not wide enough, or super bright lighting condition which is commonly seen in a sunny day. In reality, most of the global shutter camera do not have a high dynamic range. Therefore, the intensity saturation effect is very commonly seen in robotics.

FIGURE 3.17: An illustration of stereo error contributed by saturated intensity.

There are failure cases that even the intensity value does not saturate as can be seen in red boxes in Figure 3.18, but the featureless area contributed to the majority of the error. Even there is some feature nearby, it is enough to find a reasonably good depth. Such error is typically seen in a blue sky case as can be seen in Figure 3.18.

There are cases as seen in Figure 3.19 that intensity value does not saturate and there is a feature, but the result is very bad. Such a case is usually caused by pure horizontal direction feature as seen in the red box in the figure. In the stereo matching process, pure long horizontal gradient does not contribute to the matching process. The matching result only has direct correlation with the intensity of

vertical gradient. The higher the vertical gradient response, the easier it get to find true matches. Even with the help of cost aggregation, only depth in the surrounding area can be anchored/improved.

FIGURE 3.18: An illustration of stereo error contributed by lack of gradient. The green denotes the feature rich area with good depth. The redbox denotes the featureless area which leads to error in depth

FIGURE 3.19: An illustration of stereo error contributed by lack of horizontal gradient and total bank area. The green denotes the feature rich area with good depth. The redbox denotes the high horizontal gradient reponse area which leads to error in depth

Observation Summary To sum up the observation

- (A) More saturated intensity region is more likely to have error
- (B) Less gradient feature region is more likely to have error
- (C) More horizontal gradient than the vertical gradient is more likely to have an error because most of stereo disparity are obtained with horizontal baseline.

Formulate A Solution As shown in the Equation 3.3, an energy E_u is defined to indicate the probability of finding a wrong depth value. The higher the E is, the more likely the associated pixels are bad. Such E_u can be defined as combination of three factors

$$E_u = \lambda E_s + (1 - \lambda)(\gamma E_g + (1 - \gamma)E_d) \quad (3.3)$$

where E_s denotes the probability of finding a wrong patch from knowledge domain, E_g represents the probability of finding a wrong patch from image gradient domain and E_d is the probability of finding a wrong patch from image intensity domain. And λ and γ is hyper-parameters which controls the weight distribution between each sub-probabilities. Image domain analysis is refer to finding the probability of E_g and E_d .

FIGURE 3.20: Equation association in each function block.

The E_d is defined as

$$E_d = \int (I(u) - f)^2 du$$

when implemented in the discrete form. It is denoted as

$$E_d = \frac{1}{n} \sum_w |(I(u) - f)^2| du$$

where I denotes image pixel value and f is the middle point between the overexposed and the underexposed intensity. The sum denotes the average value in the neighborhood pixel for more stable performance. Sometimes based on data term alone, all false positive can be removed as can be seen in Figure 3.21.

FIGURE 3.21: An illustration of using only E_d term to find areas that are prone to depth error.

For grey scale camera, this is the only term that can be used to find the area that is prone to error such as sky as can be seen from Figure 3.22. On the original

depth map, the left part of the building has a huge part of depth error. Such an error can be seen easily in the sky section. By detecting the sky using this energy potential, the large wrongly assigned value can be removed. For blue sky, there is nothing much one can do to find it as increasing the threshold to remove the blue part of the sky would result in a large section of missing depth in other areas.

FIGURE 3.22: An illustration of using only E_d term to find areas that are prone to depth error on a less contrasted image. Depth is from modified census transform pattern with binary SGM

The E_g is the gradient aspect of the image domain analysis and it is defined as

$$E_g = \begin{cases} \|G_{I_y}^2 - G_{I_x}^2\|, & \text{if } G_{I_y}^2 \geq G_{I_x}^2 \\ 0, & \text{otherwise} \end{cases} \quad (3.4)$$

where $G_{I_y}^2$ denotes the 2nd order gradient of image I in the y direction, as can be seen from Figure 3.23. This is done to address the observation (B) and (C) in Figures 3.17, 3.18 and 3.19. Zero is used in otherwise when the stereo is used in real life where cases are either too dark or too bright. In theory, it could happen to see a featureless wall which is not too dark and not too bright, this case can be solved by iterative error removal and can also be solved by replacing zero with a smoothness measure. Since this work is done for outdoor flight, the simple zero is used. The zero value is used for the reason that false-negative affect the negation the least. The inflated obstacle range in local path planner will flood the zero with adjacent depth/obstacle value.

FIGURE 3.23: An illustration of using only E_g to find long horizontal gradient patches that are prone to errors.

When large gradient along the horizontal direction meets a small gradient along the vertical direction, the probability of finding an error in depth on that pixel

should be close to zero. When large gradient along the horizontal direction meets a large gradient along the vertical direction, the probability of finding an error in depth on that pixel should be small. When small gradient along the horizontal direction meets a large gradient along the vertical direction, the probability of finding an error in depth on that pixel should be high. When small gradient along the horizontal direction meets small gradient along the vertical direction, the probability of finding an error in depth on that pixel should depend highly on E_d as both of them cant provide accurate measurement according to observation (A) in Figures 3.17.

The E_d term and E_g term have different use case for different purpose as can be seen in Figure 3.24. The E_g terms is more likely to detect those areas which fit the observation (B) and (C) whereas E_d is more meant for the flat featureless area as in observation (A). For large area error, E_d can get a 100% clean result. Where for the horizontal featureless area, E_g can get a better result.

FIGURE 3.24: Comparison of only using E_d and only use E_g .

There has to be a weight γ to fuse them together. From the experiment, it is found that a higher E_d is ideal when dealing with a large empty scene. Typically it is selected with $\gamma \in [0, 0.3]$. Moreover, E_g is ideal when the drone is flying in between buildings. Typically it is selected with $\gamma \in [0.7, 1]$. The two cases can be seen in Figure 3.25. By setting it to a binary value, the algorithm can run faster in a low-end onboard PC.

FIGURE 3.25: γ value range vs Environment. In the right image denotes the empty scene at lon/lat:1.338485/103.772564 where γ is typically set to be zero, the left scene found at lon/lat:1.291278/103.786854 where γ is best to be selected at one

3.3.3 Semantic Domain Analysis

Semantic labeling is one of a hot topic in the recent year. Lots of research are trying to jointly estimate depth and semantic label from a single image [255][256][255][257][258]. However few people notice the possibility to use it to work with existing systems[21] to increase the accuracy and robustness. In this part, ICNet[30] in TensorFlow is used for semantic segmentation as shown in Figure 3.26.

The result is not ideal but it is the fastest model available with pre-trained weights. Base on the semantic cue of labels as shown in Figure 3.27, the large area false positive error over the sky could be removed. This cue closely resembles how human partially do stereo matching. Beside stereo cue, this knowledge cue when fully utilized can be very powerful. However, because of error from semantic segmentation model and model generalization problem, it is still can not be used in large scale. Due to current hardware limitation, ICNet has to be run with half or one forth of

FIGURE 3.26: ICNet Model [30]

the original image size in order to achieve the real time performance. When running multiple task on a single GPU core, there are high chance of memory allocate errors. Therefore, semantic segmentation demands a separate GPU allocation to run robustly.

void	road	sidewalk	building	wall
fence	pole	traffic light	traffic sign	vegetation
terrain	sky	person	rider	car
truck	bus	train	motorcycle	bicycle

FIGURE 3.27: ICNet Model representation

Let the E_s denote the semantic cue. For the sky cases, E_s will be the binary initiator on whether sky needs to be removed. If E_s is 0x8c, then the depth image d at pixel u are assigned with 0. Else the depth image d keeps its original image at u . This method is very simple but quite robust and effective in dataset. When dealing with RGB dataset, this case can directly replace original image analysis for finding sky area and mask over it. Therefore the λ should be set to $\lambda \in [0.8, 1]$ when RGB image is used and additional GPU is provided. Only when grey scale camera is used, λ should be set to 0.

FIGURE 3.28: Semantic Segmentation based output

For the KITTI dataset, a denser depth map can be obtained by filling the missing section on the ground as can be seen in Figure 3.29. The method of filling will be discussed in the later section. For actual UAV flight, the drone is dealing with non-uniform terrain all the time which gradient fill based method may fail at anytime. And to allow the drone to work outdoor, a grey scale camera is needed for shorter exposure time. Because of the grayscale camera, there is no semantic segmentation map. The only way to detect sky is through the help of the Image Domain Analysis. Last but not least for stereo post processing under real time constraint, the road filling step should not be performed because of the slow processing speed. Generally speaking, after removing the noise particle, the drone always can find previously missing obstacle information in the next frames. And it is proven in our flight test. Therefore, this worry is unnecessary.

FIGURE 3.29: Input depth from GPU census transform(top), Optional step of road surface gradient filling based on minimum fit RECT from semantic label(bottom).

3.3.4 Depth Domain Analysis

FIGURE 3.30: An illustration of iterative wrong depth map removal at iteration 0.

For depth domain analysis, hundreds of depth images are gathered for analysis and understanding on what is a good quality image and lousy quality image. The images include processed output from both KITTI and live captured images from UAV in Singapore. From experiments, we notice a few problems. The stereo method we use is the census transform with SGM technique. General speaking it can provide the proper result as long as the scene distance is appropriately fit to the baseline and lighting is good enough. However, it is nearly impossible to control the whole environment. After weeks of testing, we conclude for three types of error. Those errors are small piece snowflakes error, large false positive error, and large false negative error.

FIGURE 3.32: An illustration of iterative wrong depth map removal at iteration 2.

critical errors and filter them out. This is the most important step for a proper depth map for navigation. If it is not done, then the error will be left unchecked or even amplified as shown in Figure 3.38 bottom left.

The depth map is optimized according to Equation 3.5.

Let S, T be the source image and target Image patches. The initial value of S is the full depth map I_d . Let T_r be set of target patches to be removed. Typically the gradient of the depth map should not be high as the close depth should smoothly propagate to the far depth. If suddenly there is a large depth gradient and the corresponding depth is also large, it is a good indication of this area are full of wrong depth and it will affect the navigation. Thus it is a good indication that this area needs to be removed.

FIGURE 3.33: An illustration of iterative wrong depth map removal at iteration 3.

$$T = \operatorname{argmin}_T \int S - T_r \quad (3.5)$$

where

$$T_r = \begin{cases} S(u), & \text{if } \|\Delta I_d(u)\|^2 * I_d(u) \geq \tau \\ 0, & \text{otherwise} \end{cases} \quad (3.6)$$

$\|\Delta I_d(u)\|^2$ and $I_d(u)$ denotes strong depth image gradient and strong depth reading. The high response residual will be marked for removal. τ can be found by finding the theoretical derivative of the closest object and zero but it can be a lower value in practice. The $S(u)$ denotes a boundary pixel on the wrongly detected patches will be assigned to T_r for removal. For each iteration, the S is updated by $S_{k+1} = S_k - T_r$. For each iteration, the residual boundary of the snowflakes,

which have high image gradient and irrational close range data, is removed. After that, we check if the system converges or not. If the image has more change or very small changes, then the part we discovered through iteration will be the bugs which cause the drone to misbehave. The iteration will end if the number of iteration reaches the intended iteration limit. The detailed removal/detection can be seen in Figure 3.34 and the algorithm can be seen below

Algorithm 2: Iterative Depth Error Removal

Result: T

```

S = Id;
while S - Tr ≥ r do
    temp = ||ΔId(u)||2 * Id(u);
    while u in d do
        if temp(u) ≥ τ then
            | Tr = S(u);
        else
            | Tr = 0;
        end
    end
    Sk+1 = Sk - Tr
    if k ≥ n then
        | break;
    else
    end
end
T = Sk+1;

```

where n is the maximum allowable iteration and r means the maximum allowable residual error. Typically the n should be set less than 15 iterations for a 480p image. The removed patches contain the target pixel and pixel within one pixel of surrounding pixels.

At this stage, all crucial errors have been detected. The remaining minor error is less obvious and won't cause an immediate navigational error. Those errors include the missing occlusion or mismatched area. The mismatched area appears to be some distant small harmless objects. For the actual flight test, the map is

FIGURE 3.34: Cleared depth map over iterations

created on this filtered depth map. A further enhancement might lead to overload CPU demand. Then the critical error-free map is then further cleared with the less apparent error with the help of E_u .

3.3.5 Corrected Depth Map Enhancement

Although remaining snowflakes will not cause the navigational error, it is always desired to remove them in the process. It has to be done so that the global path planner can plan a longer path in advance correctly. For this task, a bidirectional similarity base method is used to find the most apparent pixel patches to fix to missing values.

Let's use back the previous concept of finding snowflakes error. Let's denote the area to be refined T and area to be the reference as S . The formulation is very similar. But instead of $S = I_d$, this time $S = (255 - I_d^{ErrorFree})$

$$T = \operatorname{argmin}_T \int S - T_r \quad (3.7)$$

where

$$T_r = \begin{cases} S(u), & \text{if } \|\Delta I_d(u)\|^2 * I_d(u) \geq \tau \\ 0, & \text{otherwise} \end{cases} \quad (3.8)$$

In this case, the mean of most of the parameters is the same. But T means the missing area or hole area to be patched. Take note that, we do not seek to fully optimize the image as that could lead to some global error and super slow performance.

FIGURE 3.35: Reference patch area and target area generation

The occluded area is missing enough information for geometry to patch. Without knowing the assignment of foreground or background, there is no way to fixed occlusion accurately. The result of this equation can be seen in Figure 3.35. The target patches are filled with similar intensity value from the reference patches.

Let us define a customized similarity measure $Sim(S, T)$ where S denotes the source and T denotes the target. The $Sim(S, T)$ can be represented as:

$$Sim(S, T) = \nu \frac{1}{N_S} \sum_{P \in S} \min_{Q \in T} \|P - Q\|^2 + (1 - \nu) \frac{1}{N_T} \sum_{Q \in T} \min_{P \in S} \|Q - P\|^2 \quad (3.9)$$

Where N_s and N_t denotes the number of patches in source and target area. Q, P are the patches in T and S , $Q \in T$, $P \in S$. The $\|\cdot\|^2$ denotes euclidean norm. The patch size is 7. ν is a hyperparameter that determines whether it is source critical or targets critical. Moreover, ν has the effect of speeding up the process by setting it to a binary number. On the dataset test, ν is set to be 0.5. On drone test, this was set to 1 to speed up the process by using if condition to drop the other term. However, it is still too slow for real-time drone flight. The pattern search area for depth is usually along the horizontal line. This is generally true for 2D vehicle in KITTI dataset. For the 6 DOF drone, the search area has to be increased due to the roll angle of the drone. In previous error removal part, patches are removed iteratively to correct the wrongly matched part. In this part, pixels from patches of the similar pattern are added iteratively. Let us define q be a pixel in the target area and $I(q)$ denotes its depth map intensity number. The m denotes number of patches in T that has the pixel q and n denotes the number of patches in S that has the pixel p . By introducing an estimated q , the bidirectional similarity error can be defined as

$$Err(I(q)) = \frac{1}{N_S} \sum_{j=1}^n (I(p_j) - I(q))^2 + \frac{1}{N_T} \sum_{i=1}^m (I(p_i) - I(q))^2$$

The missing depth value in the target region $I(q)$ is updated as

$$I(q) = (\frac{1}{N_S} \sum_{j=1}^n I(p_j) + \frac{1}{N_T} \sum_{i=1}^m I(p_i)) / r \quad (3.10)$$

where

$$r = \frac{n}{N_S} + \frac{m}{N_T}$$

where p is the pixel in the source.

The rules to find the optimal depth are: (1) Based on the target patch around the missing depth pixel. Find the most similar patches in the source region. (2) Based on the average pixel value in the source region(from (1), find the most likely matches in the target region. (3) Then based on the average value in target area, update the estimated depth $I^{k+1}(q)$ intensity according to Equation 3.10. The initial guess of the missing depth pixels is found by averaging the patch nearby. The ending criteria are defined similarly to the error removal part. If both patches agree to each other with a small error less than 10, this iteration stops. If the number of iteration is over n , this process stops. The current estimated $I^k(q)$ will be used for filling the missing depth. All the missing depth in the black area of Figure 3.35 are filled in this way. The process of filling missing depth value can be seen from Figure 3.36.

FIGURE 3.36: Finding the optimal the depth in each iteration

In this substage, the objective to find those areas which have no disparity or small snowflakes which floating in the void, as shown in Figure 3.36. This is the same step to find the critical error but on the inverse depth map which means $\tau > \psi$. The ψ are hyperparameter that is picked empirically. Logically speaking there should not be a small floating object on the road. Therefore we use the weak gradient to find those section and propagate them through no depth area. The propagated area which is the total number of pixel u being changes will be recorded and serve as

a masked weighted map. The masked weighted map is fused with image domain probability to form a potential map. The ratio between depth and image potential map is subject to data and camera. If the probability is high, then it will be masked for hole filling. However not all area are suitable for hole filling, Those are with label SKY will be removed for any processing.

For some people may argue about the necessity of removing minor errors. The result of this step can be seen in Figure 3.37. When we perform the inpaint operation, these areas are falsely created due to distant errors. After removing the minor error, the depth map aligns better with the actual depth.

FIGURE 3.37: Input image(top), Critical error corrected depth(middle left), minor error corrected depth(middle right), Depth image enhancement on only critical error corrected depth(bottom left), Depth image enhancement on only all error corrected depth(bottom right)

FIGURE 3.38: First row: Input left, Input right, Second row: Convolutional neural network feature with SGM depth left, Convolutional neural network feature with SGM depth right, Third row: ICNet output semantic label, Left right consistency based method[28], Last row: DepthComp[21], Proposed method.

3.4 Implementation and Experiment

To run the proposed full algorithm in real time, they have to be run in a parallel manner. The message exchange system running behind is set up by UDP based ROS multi-machine communication.

RGB Camera For an RGB camera, the full algorithm can be deployed. Typically one pc is focused on getting the GPU depth map, and the other is focusing on generating the semantic model. On the stereo processing PC TX2, the CPU resource is allocated to visual odometry, OCTOMAP and modified RRT based navigation packages. On the semantic label PC, the CPU resource is allocated for image inpaint and error removal. The overall assignment can be seen from Table 3.1.

Category	CPU	GPU
HP notebook	Visual Odometry Map Generation Path Planning Flight Control	Stereo Matching
Nvidia Tx2	Critical Error Removal Minor Error Removal Inpaint	Semantic Segmentation

TABLE 3.1: PC Task Assignment for the RGB Camera

For the hole filling process, it is a relatively slower and unpredictable process. Multiple experiments are done to test various implementation of hole filling technique such as simple fast marching(FM) based method, exemplar-based(EX) method and latest Shift-Map(SM) based method. The image shown in this work is by using Shift-Map based method on the reduced size depth map.

When the amount of light exposed to the camera is too extreme, it is often running without enhancement as large empty depth requires a longer time to inpaint.

In this experiment, ZED is used for its low cost and compact solution. However, ZED is a rolling shutter camera and characteristic inside cannot be controlled. Uncontrolled or un-modeled rolling shutter camera can cause the drone to crash. Due to safety reason, the flight is not done with ZED camera.

Greyscale Camera The flight test is done with the UI 1221 M camera. The 1221M is a greyscale camera and 122X series has been widely used in research such as in [9],[90],[12], [80]. When using a greyscale camera, the semantic label part is removed. All error removal part is transferred into a single HP notebook PC. All navigation modules are integrated into one PC.

The experiment is done using multiple platforms from low-end HP gaming laptop to TX2. The main test platform is HP Omen 15-ax213TX i7-7700HQ with i7 quad-core 2.8Ghz and 1050ti GPU.

The image used in the data set is 1242*375. The images used in real life is 720*480 from UI-1221 camera. Compare with one i7 and one TX2 based method [12], this method uses the lesser amount of hardware and able to do a larger scale testing. Using GPU census transform with SGM the speed takes to process one frame changes from 17 to 40ms depends on how many processes and how many visualization programs like RVIZ are jointly running behind.

The fast version of the convolutional neural network based stereo feature takes from 4 to 10 sec to complete depends on how many processes are running concurrently. Semantic segmentation takes around 360ms for 0.5MP image and can be accelerated to 120ms @0.125MP by downsampling the image by half. It can be real time if its downsampled again. Image domain analysis takes less than 2ms@0.5MP image Depth domain analysis takes slightly longer for 10ms@0.5MP image because of the iteration. For inpaint, the speed is inconsistent. Time taken to process can varies from 50ms to 100s. The process time is determined by image size, the number of bad pixels, hole filling template size.

For actual flying case, we are flying without the hole filling and ground filling. Simple detect and remove the false positive is enough for the drone to navigate around. The inpainting inconsistency delay causes the control gain become hard to tune due to obstacle avoidance in the loop design. Therefore for safety reason. We did not use it in the actual drone test. We only use inpaint for the offline analysis.

As can be seen from the Figure 3.38, raw proposed result using the census transform can reach the similar level of the deep convolutional neural network-based model.

The real time performance can be seen from the Figures 3.39 and 3.40

FIGURE 3.39: The first row is input image left and right, Second row is processed depth and semantic label. Last row DepthComp [21], and proposed solution. Testing image is from KITTI dataset 2011 09 26 drive 0035 sync sequence

FIGURE 3.40: Top and side view of projected map

3.5 Additional Result and Analysis

3.5.1 Qualitative Analysis

As shown in Figures 3.41 to 3.48, the benchmark[21] shifted the overall depth value and proposed method keep overall depth value. The proposed method do not shift the overall disparity value.

In Figure 3.41, most of the processing result is similar, the proposed method is designed to minimized false positive to reduce map maintenance effort and benchmark are designed for an average result. But the sky area in benchmark[21] has more false positive when compared with proposed methods.

In Figure 3.42, The sky in benchmark[21] turns into streaking stripe lines. And all the lines are considered as false positive. Previous work only consider the smoothing within the same label but did not consider the meaning of that label. In proposed work, the line type of error is removed.

In Figure 3.43, most of the processing result is similar, the benchmark[21] have streaking effect at left sky section where the proposed method use context meaning to remove them. The streaking effect usually holed the form of binary bars codes. The road lamp and trees also suffers from streaking stripe effects from previous method. In the proposed method, the tree and road lamp are well aligned with the input image.

As shown in Figure 3.44, benchmark[21] has large area of false positive in the sky region. In the near field, the depth was less smooth than it is supposed to be. In proposed work, the depth are better and appears to be more smooth.

In Figure 3.45, the rider in the benchmark[21] suffer from series of streaking stripe effect whereas the rider remains its shape and depth in the final proposed method output. In Figure 3.46, benchmark[21] made the pole on the road looks like KFC chicken leg. The propose method maintains the correct depth.

As shown in Figures 3.47 and 3.48, both the semantic labeling output and stereo output are extremely poor. But both benchmark and proposed method have similar in performance. The benchmark trying to fill part of the wall but introduces additional streaking stripe effect.

FIGURE 3.41: Additional result 1 for enhanced post processing

FIGURE 3.42: Additional result 2 for enhanced post processing

FIGURE 3.43: Additional result 3 for enhanced post processing

FIGURE 3.44: Additional result 4 for enhanced post processing

FIGURE 3.45: Additional result 5 for enhanced post processing

FIGURE 3.46: Additional result 6 for enhanced post processing

FIGURE 3.47: Additional result 7 for enhanced post processing

FIGURE 3.48: Additional result 8 for enhanced post processing

Detailed Comparison of Objects The detailed comparison can be seen from Figures 3.49,3.50,3.51,3.52.

FIGURE 3.49: Additional object comparison 1. Image intensity increased by 50%

It can be seen from the Figure 3.49, all the sky are being removed and form into a clear path for the drone to take.

FIGURE 3.50: Additional object comparison 2. This small streaking effect ranges from 0 to 255. May cause falsely generated obstacle.

As can be seen in Figure 3.50, the sky was removed using our proposed work. When using with previous work, their work shows bar code like streaking pattern which in 3D it is represented as close up obstacles.

FIGURE 3.51: Additional object comparison 3. The horizontal streaking stripe are commonly seen in older 2D scan-line based optimization method.

FIGURE 3.52: Additional object comparison 4. The streaking stripe effect will create sudden close obstacle. The pole size is remained in the proposed method where previous work enlarge the obstacle size

As seen in Figures 3.51 and 3.52, the proposed method can handle small and dynamic object in a more proper way. Multi-domain analysis based method can keep obstacle silhouette into minimal so that a free path can be found. In general, the proposed method can perform better in small details by removing falsely added or associated depth. It is important that no additional obstacles are added to the map. So that there will be a path for the drone to take. In some cases, the proposed method overall error might be higher. But drone navigation solution, the lower false positive means that the mission can be carried out without sudden interruption. The false-positive obstacle can be seen from other viewpoints and the obstacle next to the false positive usually will have inflated area that block the drone to go through. Therefore, having a low FP rate is much more important than the overall accuracy rate.

3.5.2 Quantitative Analysis

The quantitative analysis can be shown in the Table 3.2. For dynamic object and small object, there is no LIDAR detection as shown in 3.53. Also for the obstacles exist in the sky, there is also no LIDAR reading. Therefore the ground truth has to be taken from another source.

FIGURE 3.53: Lack of ground truth data from Kitti Dataset[31]. The area 1 to 6 suppose to be compared but lack quantitative data for evaluation.

Here we use subpixel enhanced image with manual labeling of red depth shifted map. Participants are required to label the set of the area which they think depth is wrong. Then among the wrong depth area, subclassification of false positive and false negative are further labeled. The idea can be seen from Figure 3.54.

FIGURE 3.54: Quantitative analysis (error in pixels) of error in object level in subpixel accuracy. The actual error for proposed should be lower when consider the FN to be occluded area.

The result are listed in the table. From the Table 3.2, it can be seen that the proposed method always keep a low false positive rate.

This is only for pixel shift analysis, for ground truth depth measurement, there are not enough points on those small and dynamic objects. Therefore, these studies cannot be done. For autonomous drone navigation, the sudden jump of the obstacle

Item	Overall Error Rate	False Positive Rate	False Negative Rate
<i>Rider_{proposed}</i>	3.01%	0.18%	2.81%
<i>Rider_{DepthComp}</i>	7.90%	5.75%	2.15%
<i>Van_{proposed}</i>	3.27%	0%	3.27%
<i>Van_{DepthComp}</i>	7.90%	3.16%	8.32%
<i>Pole_{proposed}</i>	4.64%	0%	4.64%
<i>Pole_{DepthComp}</i>	6.99%	5.2%	1.79%
<i>Pedestrian_{proposed}</i>	0.82%	0%	0.82%
<i>Pedestrian_{DepthComp}</i>	0.17%	0.17%	0%
<i>Sky_{proposed}</i>	0.54%	0.16%	0.38%
<i>Sky_{DepthComp}</i>	0.63%	0.32%	0.31%

TABLE 3.2: Object level subpixel accuracy comparison

can mean sudden stop of drone. When subcomponent like visual odometry met with this case, the VO can shut down, and the drone can crash.

3.5.3 Result with the Real World Data

The stereo matching used in this test is GPU based Census transform with Semi-Global Matching on four direction. The effect can be seen from Figure 3.55. The

FIGURE 3.55: Stereo Post Processing for a stable flight at roof. Stereo baseline is about 40cm. The flow chart show case the effectiveness of iterative error removal.

error is typically at the sky when the image is not over saturated. When dealing with the extreme case in the lunchtime, the failure can be all over the map and rules for γ has to be set to 0 with higher removal threshold. This is done so that there is free space to travel instead of the fully occupied map by the wrong depth

estimations. The sample videos can be found in the urls below. Recording the ROS bag will cost high CPU usage to handle. All PC resources are occupied for camera IO, Stereo, Post-Processing, State Estimation, 3D Mapping, 3D Path planning and control. Therefore only low bandwidth desktop video stream is recored using external recorder device which is zero CPU usage.

The flight computer video can be found in the links[259][260].

This work also is tested with Singapore Technologies Kinetics Ltd (ST Kinetics) autonomous car projects. They provide with their left-right image pair and their poor result using three-way SGBM with minimal post-processing. The detailed parameter was not provided as it is considered confidential. The data was provided so that they can seek better solutions. The proposed output was generated by adapting all the method and the parameters from the UAV testing case. The result can be seen in Figure 3.56.

FIGURE 3.56: Stereo output in autonomous car test. Proposed method outperform current ST-K autonomous car benchmark. Image captured from Singapore Autonomous car testing facility.

Heuristically speaking, the human was supposed to be a few meters away from the car, and the car is supposed to be able to drive forward with max steering to the left. However, the ST depth map will have obstacles closely in front of the sensor e.g. within 1 meter, which prevent any forward actions. The inflated obstacle in the navigation module will only able to make the car moving back instead of the drive around it. If there is a stereo at the back of the car detecting the same close range obstacle, the car will not be able to move at all. Thus the autonomous car will be stuck in the testing field.

3.6 Limitations and future works

In the Singapore Context, data from the local government shows that it rains on average 167 days and drone is not waterproof. Out of the remaining 198 days, half of the time is at night and is usually too dark for the vision to work at all as can be seen from Figure 3.58. The Monsoon Surges and occasional "Sumatra Squalls" can cause an on average of 40km/h wind speed which prevents the drone from operating normally in autonomous mode. Such strong windy condition can be seen easily in the early morning every day. Safety measure and risk assessment mandate that if winds speed is over 40km/h, it is recommended not to fly. Beside the windy and raining condition, there is too much direct sun irradiation at lunchtime. Excessive irradiation causes the vision to white out.

FIGURE 3.57: Vision based navigation fail during night. Image taken in 5th floor at latitude 1.291328 longitude 103.786790 facing north direction. Both shutter speed is set to be 15ms for both case.

For the indoor case, there are also high chance that particular space has no feature or minimal feature as can be seen in 3.57.

For outdoor cases in Figure 3.59, the sun glare is the real pain in the ass. During the project testing phase, most of the test happens in mid-day with occasionally test at night. During the delivery demo, validation test happens at early morning which sun is still directly shining into the camera. Such a case causes the drone to stay hover as no suitable path can be found. Glare can come with different angle and shape and cause false detection. Currently, no method can solve this.

Under those cases, vision-based autonomous drone navigation could end in disasters. One way to bypass the problem is to use active sensing device like LIDAR

FIGURE 3.58: Vision based navigation fail at dark and the featureless area. Image taken in controlled civilian private airport area at latitude 1.405525 longitude 103.865645 facing East-South direction. Both shutter speed is set to be 15ms for both case.

FIGURE 3.59: Vision based navigation fail at 0600 to 1030 and 1530 to 1800.

or customized outdoor enabled KINECT which later chapter in this dissertation introduces LIDAR based navigation. Overall speaking, the drone is ready to fly autonomous ops for 20-40 days or 480 hours to 960 hours per year. That means the drone can be operational ready only at 5.5% to 11% of the time. With the help of this work, the drone can partially operate in extreme lighting condition and extends partially to the night(no more than 6 pm). It is estimated by professionals from local drone inspection industry that this work can improve operation-ready rate from 5.5% to 11% towards 10% to 15% for the fully-autonomous or semi-autonomous mission. However, this number does not meet the expectation for mature civilian industry or military use.

3.7 Conclusion

In this Chapter, a new stereo depth post-processing method(PPMDA) is proposed to handle the wrong matches and imperfect depth value. The imperfection usually denies the drone to fly intelligently. It is demonstrated on both public and in-house flight tests. Previous works using either left right depth consistency check or multiple cases for outliers removal suffer from difficult lighting condition and the large area of false matching. In the proposed method, most of the unwanted fault in the depth image are removed to ensure smooth navigation. The result outperforms all other stereo post-processing methods. With the proposed PPMDA, the operation ready rate can be improved for the autonomous UAV navigation. The sample video for avoiding a static building can be found in the links[259][260].

Chapter 4

A Generalized Multi Exteroception Sensor Extrinsic Calibration Technique Using Orthogonal Scene

4.1 Introduction

In the previous chapter, it is shown that vision can fail for multiple reasons such as the sun glare, relatively dark environment, and over exposures environment. Therefore, vision alone can not be used for robust navigation. To increase the robustness, the additional sensors must be added.

The humans navigate around using a series of direct and indirect measurement technique. The direct technique includes motion parallax, stereo cue, step counting, IMU feedback, force feedback, and etc. The indirect technique is more complex such as semantic understanding, infinity lines, kinetic depth effect, perspective, relative size, elevation and a variety of other higher cognitive-based methods as seen in figure 4.1.

FIGURE 4.1: Animal Sensing Model, Red arrow denotes where the observation is taken. Illustration images come from [32] [33] [31]

For indirect measurement type, the researchers barely started to scratch the surfaces within the last decades as indirect measurement requires more resources to understand and compute. The direct methods can be seen easily on the robot through the uses of visual-inertial feedback and wheel encoder feedback systems. There is something more even human are not equipped with such as ultrasonic, RADAR and LIDAR measurements. More source of sensors are needed for robust navigation. Hence calibration for multiple types of sensors are needed.

To use multiple sensors together, one has to figure out the position and orientation of each sensor integrated as shown in Figure 4.1 red arrow. This additional work is everyone in the multi-sensor fusion business or autonomous car industry has to deal with but often ignored. Take the Homosapien for example, children learn the position of the two eyes and the position of the IMU from many years of the baby crawling, falling and collisions. The resilient biological body can endure those minor damages by replacing tissue and auto heal broken bones over time. However, for a drone, such a crash usually means the end of this drone. The sensors and body structure parts of the drone can be fractured easily.

The problem is how to calibrate multiple types of sensors using a minimal amount of features and templates.

The mainstream calibration studies use a rigid chessboard pattern or specially ordered rigid and complex boards to calibrate those sensors. To ensure rigidity, heavy metal reinforcement has to be added. Therefore, those target boards are too heavy and bulky to carry around. For easy use, the calibration pattern must be easily seen and can be picked on the testing facility. To achieve that, we proposed to use a series of wall scene to form an analytical solution to this problem. The wall scene can be seen easily in any of human civilization. By adopting a series of constraints, the closed form solution can be achieved.

4.2 The State of The Art

Most of the multi-sensor fusion work originated from the 2007 DAPRA Urban Challenge. At that time, the majority of the team uses hand measured based method because almost everyone is doing it for the first time. From that period, there are many multi-sensor fusion based research works[63][64][65][66][67][68] with only few work [67] mentioned its calibration method. With those few who includes their calibration method [67], even fewer works can show quantitative measurements and comparison with other approaches. The difficulty is that the internal orientation and position is difficult to be modeled by the external sensor. In other words, there is no ground truth. Because it is difficult to model and hard to produce a comparison result, more researchers decided to tape measure everything and focus on other areas such as odometry, Mapping, and Path planning where the result can be compared. It is reasonable to assume the majority of multi-sensor research works uses hand measured result for fusing the observation data. Currently, there are three main approaches to handle the multi-type sensor calibration problem namely:

- (1)Pattern-based approach. This method is done by using a Chessboard pattern or specifically made model for extrinsic calibration. This is the mainstream multi-sensor calibration.
- (2)Ground truth-based approach. This method is done by using Localization or SLAM result from other sources to help measure sensor positioning.
- (3)Tape Measured-based Approach. This method is done by hand measure some observation and used for nonlinear least square fit optimization. It is a simple and effective solution for multi sensor calibration.

4.2.1 Structured Pattern Based Approach

FIGURE 4.2: Chess board based calibration by [34] (left), and Vasconcelos et al. [35] (middle) and Zhou and Deng [36] (right). Images obtained from each paper

Earlier Formulation Using Chess Board Pattern Zhang and Pless [34] first propose to use constrained depth based approaches on top of the standard camera calibration package. In their work, multiple poses of image data and laser data are captured at the same time in front of the camera. For each calibration pose, both image corner points and horizontal plane are extracted. The set of rotation matrix and translation vector from the observation can be obtained. The assumption for his work is that 2D LIDAR is placed horizontally. So the rotation difference between 2D LIDAR and camera observation can be calculated using an approximate transform matrix. For each iteration of the pose, the rotation is refined and the translation is also obtained by the reprojection function.

The summary of his work is to use 3D to a 2D type of reprojection method to find the best rotation and translation calibration parameters.

$$\sum_i \sum_j \|p_{ij} - p(K, R_i, t_i, P_j)\| \quad (4.1)$$

where $p(K, R_i, t_i, P_j)$ denotes the projection of 2D LIDAR points P in pose j into pose i . K denotes the camera projection matrix, and p denotes the image coordinates. By minimizing the 3D to 2D reprojection error, the most suitable transform/calibration result can be found.

Problems in earlier methods The major problem with this method is that (1) Approximated rotation matrix may not be $R \in SO(3)$ (2) The initial guess has to be very close, or the result can lead to local minimal

Improved Formulation Using Chess Board Pattern Zhang and Pless [34]’s work inspires people such as Vasconcelos et al. [35] to extend the calibration into the general monocular visual odometry problem by reformulating the problem as a standard PnP problem. They adopt a PNP problem strategy to find a minimal closed form solution for the extrinsic calibration problem[35] . The chessboard dimensions are known and therefore can be used to form a set of hypothesis LIDAR observations. Then the RANSAC is introduced to couple set of visual and lidar observation. This solution only requires a minimal of 5 pose observation to generate a valid 3D result achieving considerably improved outcomes.

Outstanding issue However, chessboard based approach calibration does not always ensure a good result. The iterative solution can sometimes be trapped in local minima with an incorrect initial guess. To solve this, Zhou and Deng [36] (2014) further introduces an orthonormal constraint in the computation of rotation. The translation vector in Zhou and Deng [36] can be found by a simple least square fit. The least square fits solution can be used as pose hypothesis for the RANSAC algorithm. This algorithm [36] is so far the state of the art in chessboard based approaches[36]. Still, the problem remains. For example, the large chess board also tends to warp and causes an additional error. The wrapped board will cause problems when finding straight lines from LIDAR observations.

Trihedral-based pattern Gomez et al. [70] proposed to use a structured trihedron based pattern for calibration. Their work is considered as an improved form of Zhang and Pless [34]’s work. The trihedron has nice property that every two adjacent surface have orthogonal constraints, and every surface can impose coplanarity constraint. The coplanarity holds in the form of

$$n^T \bullet Rv = r \quad (4.2)$$

where r is the error residual. The n denotes the normal plane vector, R is the rotation from LIDAR to the camera and v is the direction of scan in 3D. When

FIGURE 4.3: Trihedrons based calibration by [34] (2015). Image from their papers

$r = 0$ the coplanarity constraint holds, The orthogonal constraints hold the form of

$$n_i^T \bullet n_j = 0 \quad (4.3)$$

Where $i! = k$ which states that the normal vector dot product should be zero. By solving the nonlinear least square formula

$$\operatorname{argmin}_x \sum_{i=1}^N r_i^t w_i r_i \quad (4.4)$$

Where w denotes the weight of the corresponding observation residual from the trihedron. This solution is one of the cleanest solutions to multi-sensor calibration study, but it has its weakness.

FIGURE 4.4: Failure Scenes for Trihedrons based method. The first assumption for Trihedrons based calibration is stable tracking of corner scene. The edges in the failure case cannot be tracked robustly due to multiple detection and featureless corners.

Trihedral-based pattern problems When more sensor is used, the additional observation does not contribute to accuracy and error can build up. Also, the perfect trihedron is difficult to find as seen in Figure 4.4, initialize and track as Trihedral comes with various sub edges and arc pattern in reality.

4.2.2 Ground Truth Based Approach

Usually, the calibration is done to get accurate mapping and odometry information. However, when good odometry can be obtained from other sources such as expensive IMU with VICON or wheel encoder, the calibration case can be simplified substantially. Such example can be seen from the work of He et al. [37]. Then

FIGURE 4.5: Ground truth based approach by [37] (2015). Images obtained from their paper and PCL library for illustrative purpose.

this sort of problem can be simplified as minimizing the set of 3D observations by

$$\operatorname{argmin}_T \sum ||X_n - T_{nm}X_m|| \quad (4.5)$$

where X denotes the 3D point cloud, n, m denotes sensor id and T is the transform.

The Advantage and Disadvantage The main advantage of this approach is simple and can be expanded to multiple sensors easily. As shown in Figure 4.5, it can be quickly expanded to both 2D/3D types of camera and LIDAR without changing too much of the formulation.

The main disadvantage of this approach lies in its assumption. This method requires odometry/localization/state estimation prior. The requirement for ground truth level of localization is often contradicting with the original intention of calibration which is to get improved odometry from multiple sources of observations.

Besides that, the dense point cloud alignment can meet in degenerative case such as classic planner scenes. To get proper alignment, the initial guess for each sensor has to be accurate. So there is too much assumption using this method which makes it less favorable in lightweight outdoor drone solution.

4.2.3 Advanced Tape Measured Approach

The idea follows the forward kinematics robot arm link setup. The same base to eye transform can be represented by either AX or ZB which forms the nonlinear least squares fit problem. By minimizing the error from two observations, the final result can be obtained. However, this method requires a tape measure of 2 matrix X and B . The tape measurement can be obtained from measuring tape.

FIGURE 4.6: Tape Measured Approach by [38] (2017)

Based on robot multi path robot forward kinematics

$$AX - ZB = r \quad (4.6)$$

where r denotes the error residual. The optimal A and Z can be found by minimizing the nonlinear least squares problem

$$\operatorname{argmin}_{A,Z} ||r||^2 \quad (4.7)$$

Where B and X are measuring tape measurements. This idea works very well for the ground robot with fixed heights. The test result usually won't be trapped in local minima.

Advantage and Disadvantage This method makes the calibration simpler with one camera and one LIDAR. Moreover, a couple of new research works already applied this method.

This method lacks the ability to calibrate with more sensors. When multiple sensors are added, the formulation will not help to improve the result from added observations. What's more, if the newly added LIDAR are placed vertically, then the robot has to turn sideways for calibration. This method will amplify the error between two LIDARs.

4.2.4 Honorable Mentions

In the series of multi-type sensor calibration methods, many other approaches are using either direct 3D to 3D alignment or specially crafted boards. For general direct 3D to 3D alignment, it usually suffers from accuracy problem. The crafted calibration board can make accurate result at the cost of more troublesome hardware setup. For lightweight UAV calibration, those methods are the worst cases. The technique in Figure 4.7 represent specific efforts in multi-sensor calibration, but the contribution is considered limited. Take direct 3D for 3D alignment for example, the method is directly taking from the finding 3D to 3D motion in the chapter 2.

FIGURE 4.7: Other approaches by [39] (2014), [40] (2018) and [41] (2017)

For general exteroception sensing, 3D to 3D type alignment was previously used. It is not ideal to use that type of association for the generalized model because some sensor type may have significant variance in the observation. Moreover, 3D to the 2D type of association often require very complicated scene setup such as the trihedron or the chessboards. Therefore for this work, orthogonal planes models

are used in finding the best T so that all observation can suit the model of the orthogonal plane. The 3D to 3D, 2D to 3D and 2D to 2D based motion estimation are referring to the method explained in chapter 2.

4.3 The Proposed Approach

FIGURE 4.8: Assumption and Objective of proposed method. Image source[42]

The purpose of this work is to design a method that can find the extrinsic parameters for multiple exteroception sensors with a minimum amount of calibration hardware requirements. The exteroception sensors include but not limited to stereo cameras, 2D LIDARs, 3D LIDAR, and RGBD cameras.

The assumption is that the sensor intrinsics are known and calibrated with a minimal re-projection error. When a sensor detects environment, the observation usually comes with various level of noise and false detection. An orthogonal scene is required for using geometric constraints such as co-planarity constraint and surface

orthogonal constraint which is similar to [34][261]. The idea can be seen from Figure 4.8. Based on multiple constraints, it is relatively straightforward to formulate the problem for finding the most suitable rotation and translations. The overall concept can be seen from Figure 4.9. 3D line measurements from multiple sensors are taken for form the orthogonal and coplanar consensus

FIGURE 4.9: The overview of the proposed idea to calibrate multiple type of sensors.

4.3.1 Data Preparation

The 3D type of sensors such as RGBD camera, stereo, and Velodyne often can produce dense 3D data. To use them effectively and accurately, we extract the middle line with a line center crossing the Image center at (c_x, c_y) . If depth across optical center has large uncertainty which defined in previous chapter, the most closest line with low uncertainty in the depth image will be picked. The method for finding the most likely wrongly matched area based on image intensity map can be found in Chapter 3 Image Domain Analysis Section.

The 2D LIDAR scan produces 1080 points at 40 HZ in polar coordinates $[r_i, \theta_i]$.

The raw scan will be transformed from polar to Cartesian coordinate $[x_i, y_i]$. Based on each scan at a particular pose, line detection is applied to find the number of particular lines in the scan. There are multiple approaches to find the line such as the Split and Merge method, RANSAC and Hough transform. For general implementation, there is no limit on which method to use as long as one can find the relationship between each detected line and raw input points. In this work, RANSAC based line detection is used to find the line models and determine the uncertainty along each line. The test for whether a detection belongs to a line is found by

$$A_n x_i + B_n y_i + C_n < e_{line} \quad (4.8)$$

where A, B, C denotes the line parameters for n th line. The x, y denotes LIDAR output. e_{line} is maximum allowed error for a line controller by user. Let L_n denotes the detected line, $[l_x, l_y]^T$ denotes normalized line direction, and $[c_x, c_y]^T$ denotes the line center. To remove the biased line estimation, a probability based clustering is used to find c and l . Let p_i denotes the laser point in 2D Cartesian coordinate where $p_i \in L_n$, its variance $e_p \simeq \mathcal{N}(0, \sigma_p)$. σ_p for LIDAR is static. The centroid of the detected line can be found as

$$c = \frac{1}{N} \sum_{i=1}^N p_i \quad (4.9)$$

where its covariance matrix is defined by

$$\Sigma_c \simeq \frac{1}{N} \mathcal{N}(0, \sigma_p) \quad (4.10)$$

The actual line direction $[l_x, l_y]^T$ is vector corresponding the the largest Eigen value λ of M

$$M = \sum_{i=1}^N (P_i - c)(P_i - c)^T \quad (4.11)$$

The covariance of line $[l_x, l_y]^T$ can be found by the Moore-Penrose inverse of H

$$H = \frac{1}{\sigma_p} \sum_{i=1}^N \begin{bmatrix} y_i y_i & -y_i x_i \\ -y_i x_i & x_i x_i \end{bmatrix} \quad (4.12)$$

4.3.2 Geometric Constraints

The previous observations are still in 2D LIDAR Cartesian coordinate system. Then they are being ported over into general 3D Cartesian coordinate system by adding 0 into z-direction. After that, the 3D coplanarity constraints can be applied to the resulting 3D observation. Let P be the point in 3D space, the coplanarity constraints P_1, P_2, P_3 and P_4 are on the same plane holds if

$$[(P_2 - P_1) \times (P_4 - P_1)] \cdot (P_3 - P_1) = 0 \quad (4.13)$$

This representation opens up the whole new space for multiple sensor alignment problems. To deal with minimal sensor configuration case, we use two observation each time to forms the coplanarity constraints.

$$(R_k I_k^j \times R_v I_v^j) \cdot (R_k P_k^j + t_k - R_v P_v^j - t_v) = 0 \quad (4.14)$$

where the I be the line with line direction $[l_x, l_y, 0]^T$ from line center $[c_x, c_y, 0]^T$. j is the plane id number, R represent the rotation from source sensor frame to target frame where k, v denotes sensor id and $k \neq v, R \in SO(3)$. The P is a point on a LIDAR scan and it is usually represented by line centers. This equation may look complicated, but it is stating the fact that vector I_k, I_v and point P_k, P_v are from same planner structure. No matter which common coordinate they are transformed into, the coplanar condition should hold.

The plane normal in 3D can be denoted by

$$n_j = R_k I_k^j \times R_v I_v^j \quad (4.15)$$

Because of the two planes are supposed to be perpendicular. Therefore their surface normal vector dot product must be zero

$$n_a \cdot n_b = 0 \quad (4.16)$$

where a, b denotes plane id and $j \in [a, b]$.

The orthogonal constraint only applicable to the rotation. The coplanar constraint corrects both rotation and translation. With these conditions, the problem can be formulated.

4.3.3 Formulation

let r_{co}^j denotes residual error from coplanar constraint for plane j and r_{ort} denotes the residual alignment error from orthogonal constraint

$$r_{ort} = n_a \cdot n_b = (R_k I_k^a \times R_v I_v^a) \cdot (R_k I_k^b \times R_v I_v^b) \quad (4.17)$$

$$r_{co}^j = (R_k I_k^j \times R_v I_v^j) \cdot (R_k P_k^j + t_k - R_v P_v^j - t_v) \quad (4.18)$$

The calibration problem can be formulated by minimizing inverse log of maximum likelihood estimation(MLE) problem:

$$argmin_T \sum_1^N \left(\sum_{cb} \frac{1}{\sigma^2} r_{ort} + \sum_{cb} \frac{1}{\sigma_k \sigma_v} r_{co}^a + \sum_{cb} \frac{1}{\sigma_k' \sigma_v'} r_{co}^b \right) \quad (4.19)$$

where N denotes number of samples. cb denotes number combination of basis vector where $cb < oC2$. C denotes the combination and o denotes number of sensor. For a three sensor system, there could have have maximum of $3C2 = 3$ constraint equations on a plane. For the demonstration in this work, only two equations with relation to base camera is used for each plane.

This problem is iteratively solved by Levenberg-Marquardt as

$$[\delta\xi_k, \delta\xi_v]^T = -(H + \lambda diag(H))^{-1}g \quad (4.20)$$

where $\delta\xi$ represent the increment transform in $se(3)$, λ denotes damping factor and H denotes the Hessian matrix with dimension 12×12 . The g is the gradient of each cost function with dimension 12×1 . The H and g are calculated as

$$H = \sum_{i=1}^N J_i^T \frac{1}{\sigma^2} J_i, g = \sum_{i=1}^N J_i^T \frac{1}{\sigma^2} r_i \quad (4.21)$$

where N is the number of constraints of this optimization from the sample calibration data. The detailed Jacobian and variance can be found in appendix. Then the sensor poses are updated as

$$T_k = T_{k-1} \cdot T(\delta\xi)^{-1} \quad (4.22)$$

4.4 Implementation details

The proposed solution is implemented with Mobile Robot Programming Toolkit (MRPT) and Robot Operating System (ROS) jointly. The MRPT package is heavily used for its general math packages but it suffers from multiple sensor IO delay issue. Therefore ROS is used for concurrent multi-sensor IO and display.

For the experiment, multiple sets of stereos, Kinect v2, Xion pro, HOKUYO 20m model and HOKUYO 30m model and 16 lines Velodyne with both usb-type and ethernet type interface were tested using this principle. The stereo has to be calibrated by Zhang [55]’s method. There were mixed test cases of USB and Ethernet IO which single node MRPT can not handle all stream concurrently. Moreover, only ROS can handle multiple sources at the same time with a different node running. Generally speaking, multiple 2D or 3D LIDAR can be calibrated at the same time. However, it is less likely to open multiple sets of stereo or RGBD at the same time due to USB bandwidth issue limited by the drone computing hardware. Only desktop pc with a dedicated PCI-E card can handle such large data flows. For LIDAR and RGBD cameras, a zero mean Gaussian noise is used to model the sensor measurement with a standard deviation of $\sigma = 0.02m$ is used. For stereo case, the sensor noise is modeled by a zero mean Gaussian noise. The stereo noise standard deviation is proportional to disparity value and local smoothness value.

For an autonomous drone use case, one of the LIDAR is always put with little rotation to the camera. This LIDAR is meant to take over the obstacle detection tasks when lighting is sub-optimal or dealing with funny cases. The other LIDARS are usually placed vertically facing front to sense those areas that are in the blind spot of the visual detection system and used for maintaining altitude.

4.5 Experiment and Results

The overall process can be seen from figure 4.9. At first, each sensor data are recorded from the multi-node system. Then based on the extracted data, lines are extracted. Take note that within one batch of observation, there can not be multiple lines with 90 degrees offset or it will trigger a degenerative case. When sensor rig is moving parallel to both walls, a degenerative case will happen as well. For parallel wall moving case, the solution to translation is non-singular. After that, the observed lines information are passed through the function that is constructed from orthogonal constraint and coplanarity constraint. The resulting sensor observation can form right angle observations.

FIGURE 4.10: Mapping with proposed methods

A small rotation error can cause distant measured points to be far away while a small translation error can only cause fixed marginal error as can be seen from Figure 4.10. Although the result is highly dependent on the initial guess, the translation vector is less dependent and can be tape measured to mm accuracy. It is the rotation that requires most of the attention. The far away roof top shows more error due to rotation error. Therefore, there are more rotation associated constraints in the optimization functions.

Although it is impossible to put a number on calibration results directly. But one can easily compare when multiple scans are fused together. The benchmark work[38] was modified to extend to vertical scan by rotating the rig and do an additional vertical measurement and optimization.

As can be seen from Figure 4.10, the tape measured approaches[38] runs for about 5 sec and the map appears to be distorted. And the proposed methods runs for 10 secs and map still consistent. The quantitative measurement can be seen from Figure 4.10. The smaller the point cloud standard deviation is, the more accurate the mapped result is.

In this test, only the frontal wall section is compared since that's the only place that all observations have a common field of view. In this test, the stereo result is not used due to large variance caused by low lighting and featureless white wall. This is one of the cases that prove the importance of multi-sensor fusion.

4.6 Limitations and future work

Lack of Ground Truth This is the most important problem in any of the calibration works. There is no way to measure the ground truth of the sensor placement. For example, each human being perceives the world differently. Some with chronic angle-closure glaucoma have lesser FOV where the majority of humanity has a normal FOV around 210 degrees. Everyone is aware of this case, but there is no direct way to measure and quantify them. This is exactly the same case for multi-sensor calibration.

Degenerative case For any problem that is solved by the least square way, there is always a chance that the rank of Hessian matrix that corresponding to a particular set of data is not full rank. That means it is not invertible. In this case, the update step cannot be performed. In the actual data testing case, such case happens frequently when there is no unique solution. The degenerative case happens with all the observation observe same 90-degree line patterns over time between two of the sensors. When such cases happen, the only solution is to capture another set of data with sensor tilted towards the walls.

Far Away Initial Guess There is another possible way of failure when the initial guess was too far away from the actual sensor layout. Such a case can frequently happen as rotation direction with respect to each sensor is easily mixed. For LIDAR and camera, the coordinates are entirely different. Both LIDAR and CAMERA operates on XY plane. However, XY plane of the camera is the YZ plane of LIDAR. When dealing with multiple types of sensor, those notations need to be transformed and adjusted accordingly before the actual calibration took place. One way to solve this is to run the sensor drivers in one node and open a ROS QT reconfigure tools to adjust roll pitch and yaw so that they appear to be the incorrect direction. After that, the parameters are copied into the initial guess for better optimization. Currently, there could be dozens of frames reference on a single drone and it can trigger mistake easily. In the near future, there should be a unified frame notation for general sensory input.

4.7 Conclusion

In this chapter, a generalized exteroception sensors calibration technique using orthogonal scene is proposed to address multiple sensor calibration problems for field-deployed drones. For a drone to navigate robustly, more sensors are needed to improve drones situational awareness. By knowing reliable drones 3D position and obstacle around the drone, better decisions can be made. However, to uses more sensors, they have to be calibrated in order to know their physical relationship.

Previous work usually uses the chessboard pattern or complex rigid structure for analysis. In field-deployed missions, drones are rarely to have the luxury of using chessboard. And each time when drone had minor crashes, all sensors have to be calibrated. In proposed method, a wall scene is required to calibrate RGBD camera and 2D LIDARs. A 3D camera is known to have a high noise in the measurement. Therefore only trustworthy part along the wall scene is used for estimating relations. By this concept, a higher accuracy measurement can be obtained.

A series of planner constraint and wall orthogonality constraint can be applied to form an analytical solution to find relative transform between each sensor. The

proposed method can also be used to calibrate 3D LIDAR and other devices like SAR image. The proposed method becomes a generalized exteroception sensors calibration solutions.

Chapter 5

Real-Time UAV State Estimation using Multi-Sensor Fusion

5.1 Introduction and Motivation

To navigate accurately and robustly in an unknown environment is the key challenge in the mobile robotics industry. The term accuracy is referring to how precise the measurement is, and the robustness is measuring how likely the system can withstand adverse conditions. Currently, it is safe to say that most of the autonomous navigation solution is not robust enough for a level 5 autonomous driving[262]. When everything is in an ideal condition, a high accuracy result can be observed as can be seen from work from [10][12][88] and [45]. When part of system is in bad condition, both accuracy and robustness can be lost easily. Once accuracy and robustness are gone, the autonomous operation will result in a crash, as can be seen in Figure 5.1. There is a lot of research works that show great performance in the indoor demo but fails in the outdoor test[10][88]. The autonomous driving solution exists before the 2000s. However, till today, no one can claim that their home-brand autonomous car can achieve 100% accuracy and robustness. This argument is supported by many reports of the autonomous car crash with the Autopilot turned on as can be seen in Figure 5.1. Even the police can not escape the fate of being hit by the rogue autonomous car. Therefore, the netizen renames the A.I car as Artificial Imbecile car due to the notorious lack of robustness problem.

FIGURE 5.1: Various reports of crashes during Autopilot after more than 20 years of autonomous car research. Image Source[43]

The autonomous driving concept has been there for about 20 years where the UAV has just begun to emerge for the past ten years. The drone has many practical uses such as disaster management, search& rescue, inspection& survey or even package

delivery as shown in Figure 5.2. The package can be goods from the Taobao or 5kg yield precision strike missile when it is used for military purpose. The technologies for military and civilian application are usually the same and can be interchangeable. The large drones can carry a variety of sensors. They can fly robustly with accurate GPS measurements and results from SAR-based terrain mapping. Such cases can be seen in daily operation in Afghanistan or Syria. This is under the assumption that drone can carry substantial weight and there is informational superiority. In the cases of Unrestricted Warfare [263] or a simple urban valley, GPS will be unreliable. And SAR devices are too difficult to miniaturize. It is still a difficult task for a small sized drone to carry a lightweight payload and navigate robustly in the urban environment.

FIGURE 5.2: Various application for drones, and technology between civilian and military are interchangeable. Image source [44]

There are several ways to fly the drone autonomously such as by using LIDAR[264][101] or vision[10][12][88] [45]. Both methods suffer from stability and robustness problem. The other extreme approach is to build a massive drone with heavier sensor payloads[80]. In the Singapore context, the Civil Aviation Authority of Singapore (CAAS) restriction over small-median sized drone makes the second approach less favorable. Because flight with bigger sized drone requires a license for each flight and each license cost money and manpower. Moreover, a larger sized drone means higher risk and leads to more concern over stability issue.

Such concern raises the question, which is How to make a lightweight drone to fly robustly in an autonomous manner?

The easiest way is to take reference from nature. Take bats for reference, bats use a series of the sensing device jointly to navigate instead of relying solely on one method as seen in Figure 5.3. So instead of make contribution on either visual odometry(VO), more attention should be paid on the fusion of multiple subsystems.

FIGURE 5.3: Biologically inspired navigation methods. Various Illustrative figures obtained from [32] [33] [31]

5.2 The State of the Art

There are very few works[10] that directly use naive visual state estimator for the UAV navigation. Stereo-based state estimation is less favorable as the stereo is limited by its baseline and degenerated into the monocular case during flight. For reliable autonomous drone navigation, the camera and IMU fusion are the minimal requirements. Stereo and LIDAR are usually used for obstacle observations. This trend can be observed from PTAM+IMU[47](2014), OKVIS[45](2014), ROVIO[79](2017), VINSMONO[12](2017). Meanwhile, there are other fusion-based works such as GTSAM[98](2012) and ETH-MSF[95](2013) being proposed to loosely coupled with naive visual state estimator[10]. There is no perfect methods but the most suitable methods for each task and application.

State of the art related to this section can be seen in various defense-related competitions and projects such as DAPRA Fast Lightweight Autonomy (FLA) program in 2017 and 2018), as seen in Figure 5.4.

FIGURE 5.4: The autonomous drone based the solution similar to the one from ETH research group[45]. Image Source[46]

5.2.1 Visual-Inertial Fusion

The visual-inertial-based fusion is the most popular approach for lightweight drone navigation and can be classified into some subcategories. The division can be from the consideration on whether they are tightly-coupled or loosely-coupled models in the state estimation. Also, it can also be from the statement on whether they are the filtering-based approach or graph-based solution. Usually, the graph-based solution is generally tightly-coupled model where the filtering-based method can be either way. The typical input for each model is shown in Figure 5.5. Both camera and IMU can give position estimation independently. The tightly coupled model uses position estimation and raw feature association whereas the loosely coupled model uses conclusions from both state estimates for refinement.

FIGURE 5.5: Different Paradigms of visual inertial integration. Note, Implementation cost referring to both implementing the current algorithm with given visual inertial sensors and additional implementation cost for adding more sensors. Robustness problem means the divergence problem and degenerative(singular) case which leads to a runtime exception. Valid variance from each source is needed to loosely coupled method or system diverge fast. Green denotes pros and Red denotes Cons

5.2.1.1 Tightly Coupled models

Tightly coupled models are the latest approach to solve the multi-sensor fusion problem. The problem is defined by "Given observations from the camera and IMU, find transforms of pose that can minimize the variance infused observation residual". The earliest work can be traced to [265]. The simplified model can be seen from Figure 5.6.

FIGURE 5.6: Tightly Coupled models Problem illustration

When dealing with multiple sensors over multiple time, formulation becomes more tedious than naive camera-based method as can be seen from Figure 5.7. The formulation for the vision part is similar to those approaches in LSDSLAM[9], DSO[92], SVO[10], ORBSLAM[88], etc. The IMU part is more complicated and will be introduced in detail later.

FIGURE 5.7: The process flow for Tightly coupled visual-inertial system. Image from [12]

There are half dozen works on tightly-coupled models. But they can be explained by three levels of hierarchical structure. The pyramid model is used to visualize this structure as seen in Figures 5.8, 5.9, and 5.10.

The factor graph directly gives the factorization of the joint distribution

$$P(\mathbf{z} | \mathcal{X}) = \prod P(\mathbf{z}_l^{c_j} | \mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_c^b, \lambda_l) \times \prod P(\mathbf{z}_{bi+1}^{b_i} | \mathbf{x}_i, \mathbf{x}_{i+1})$$

$$\log \mathcal{L}(\mathcal{X} | \mathbf{z}) = \log P(\mathbf{z} | \mathcal{X})$$

$$\text{argmax}_{\mathcal{X}} \sum_{(l,j) \in \mathcal{C}} -\frac{1}{2} \|\mathbf{z}_l^{c_j} - \tilde{\mathbf{z}}_l^{c_j}\|_{\Sigma_l^{c_j}}^2 + \sum_{k \in \mathcal{B}} -\frac{1}{2} \|\mathbf{z}_{bi+1}^{b_i} - \tilde{\mathbf{z}}_{bi+1}^{b_i}\|_{\Sigma_{bi+1}^{b_i}}^2$$

Find transforms that maximize negative reprojection error

$$\mathcal{X} = [\mathbf{x}_n, \mathbf{x}_{n+1}, \dots, \mathbf{x}_{n+N-1}, \mathbf{x}_c^b, \lambda_m, \lambda_{m+1}, \dots, \lambda_{m+M-1}]$$

Define states Camera pose, IMU pose and Feature.

$$r(\mathcal{X}) = \mathbf{z} - \tilde{\mathbf{z}} \text{ Tayler Expansion on nonlinear function}$$

$$r(\mathcal{X}) = r(\hat{\mathcal{X}} + \delta\mathcal{X}) \approx r(\hat{\mathcal{X}}) + J\delta\mathcal{X}$$

1. solving the following linear system,

$$\mathbf{A}\delta\mathcal{X} = \mathbf{b}$$

$$J^T \Sigma^{-1} J \delta\mathcal{X} = -J^T \Sigma^{-1} r_{\mathcal{A}}(\hat{\mathcal{X}})$$

2. updating the latest estimation:

$$\hat{\mathcal{X}} \leftarrow \hat{\mathcal{X}} + \delta\mathcal{X}.$$

Solve by Nonlinear Least Square Solver

Predict

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k)$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k \quad \mathbf{F}_k = \frac{\partial f}{\partial \mathbf{x}} \Big|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k}$$

Update

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k \quad \mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}} \Big|_{\hat{\mathbf{x}}_{k|k-1}}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}$$

Solve by Extended Kalman filter

FIGURE 5.8: Tightly Coupled models top level problem formulation

The top-level represents problem definition and formulation. The popular choices are the multi-view Bundle Adjustment(BA) based method and the Extended Kalman Filter based approach. There are other methods such as Maximum-A-Posteriori (MAP) Estimation, Unscented Kalman Filter and etc. MAP and UKF are lesser popular due to slow processing time and performance issues[266]. MAP requires more time for recalculating the Jacobian at each iteration but can find the most accurate result. A closely-coupled Extended Kalman Filter sample([48]2018) will be introduced in this section later. Multiview Bundle adjustment based method is considered the common approach for tightly coupled models. This method is also known as Fix-lag Smoothing based method. It is usually formulated as a MAP problem but solved as minimization of inverse-log likelihood problem as can be seen from Figure 5.8. The inverse-log likelihood problem is solved by the nonlinear least square equation on the first order linear approximation error function from Tayler expansion. This process is also known as linearization on current states. This method will give higher error output when the linearization is done on an

erroneous state estimates. The state vector contains sensors position, orientation, speed, acceleration, feature position in 3D and scaling factor at a different time.

FIGURE 5.9: The sensor models formulations and derivatives forms the second layer of the pyramid model

The middle level of the pyramid is represented by various model selection, as seen in Figure 5.9. Based on the camera model and IMU model, a set of sensor model Jacobian J can be found. Such Jacobian is also known as the state transition model F in EKF formulation. There are multiple ways to define the Jacobian such as find partial derivative with respect to pose[45][265][96] and partial derivative with respect to speed[267][12]. Based on the Jacobian, the coefficient vector b and system matrix H can be found. The system matrix is also known as the Hessian Matrix. For the IMU modeling, the bundle adjustment based method and EKF formulation share the same formula and same state transition matrix. For tightly coupled model, most of the features are in 3D. Therefore, there are multiple ways to measure the residual such as reproject using standard camera model K [45][265][96][267][48] or reprojecting on the tangential plane of unit spherical camera model[12].

At the bottom level of the pyramid, its the small choices and implementations make the difference as seen in Figure 5.10. For example, OKVIS[45] uses a custom feature detector with the BRISK descriptor to find feature and perform loop closure. The loop closure gives the state estimation system the ability to optimize for previous pose when facing a place that has seen before. The VINSMONO[12] uses Harris corner and DBOW for feature and loop closure. The majorities [268][96][79] uses FAST feature detector for input feature. For this method, the feature needs to be tracked using LKT tracker over multiple frames so that the optimization can be done on a set of common 3D observations. There are many other implementation differences which make each work[12][268][96][79][45] unique. The samples of the differences can be seen from Figure 5.10

FIGURE 5.10: Various implementation details forms the bottom layer of the pyramid model

Advantage and Disadvantage with Tightly-Coupled models The problem with Tightly-Coupled models is the requirement for higher CPU usage, higher memory usage, and longer latency. And it is hard to add more sensor as it requires another residual minimization function and its corresponding Jacobian. But the result accuracy is usually higher than those loosely-coupled models.

5.2.1.2 Loosely Coupled models

PTAM+IMU[47] and SVO[10]+MSF[95] are the typical examples of loosely coupled models. They use raw visual state estimator output to directly couple with IMU measurement from EKF filter as can be seen from Figure 5.11.

This type of method uses direct position output from naive visual estimator. Therefore there is no need for a complicated version of the Kalman filter such as Extended Kalman Filter or Unscented Kalman Filter. Each source of position output need to has reasonable covariance measures for the fusion to be stable. The EKF process is introduced in the chapter 2 and will not be illustrated again.

FIGURE 5.11: Loosely coupled model example: PTAM + IMU with EKF[47] (left, Image from paper[47]). In-house flight tests(right).

Advantage and disadvantage with Loosely Coupled models The Loosely couple model is generally speaking much faster in processing speed, lower CPU usage, and low memory usage when the state is not referenced to the landmark features. The framework can be expanded to a larger variety of sensor combinations easily. But the problem is that this type of method usually has limited accuracy and robustness. For indoor dataset, the average error can be 10% to 20% [11]. To increase the accuracy, one way is to use more type of sensor with better estimates of source covariance. The other way is to use tightly coupled EKF model such as MSCKF[73]. But MSCKF is designed to work with camera device and it is a closely coupled model which will introduce more implementation cost when adding sensors. It is one of the reasons why [48](2018) did not include LIDAR for robust state estimation.

5.2.2 Fusion-Based State Estimation

The state of the art in lightweight multi-sensor fusion is considered to be the work from Sun et al. [48](2018). They use a tightly coupled EKF based approaches for autonomous UAV navigation. The overall process can be seen from Figure 5.12.

Their model is pretty straightforward and easy to understand. They primarily use stereo depth observation to form z measurement. Then measurements are updated using MSCKF-based[73] approach. The MSCKF is optimized for visual inertial odometry with a lot more internal state such as camera frame state, IMU frame state and current global pose state whereas standard EKF only uses single global pose state. The covariance update needs to be augmented to comfort for additional states. The rest of the update process remains as normal EKF-like update method. The performance in the feature-rich environment is very good, as can be seen from video[269].

FIGURE 5.12: The state of the art Visual Inertial LIDAR Fusion-Based State Estimation drone. The drone crashes with sudden controlless drift to one direction. The Pilot force the drone down to avoid hurting staffs. Base on similar understanding and flight experience, the sudden drift only happens with hanging state estimator. The mathematical formula follows the one defined in the review section. The drone image from [48]

Problem with State of the Art There are two major problems in this work:

(1) The environment punishes this kind of singular external observation based method heavily. This method[48] assume near perfect stereo detection as the stereo is the primary observation. The LIDAR was not extensively used for adding robustness to state estimator but was used as mapping tools. In Singapore in-house testing environment as seen in Figure 5.18, the lighting condition is usually more challenging which yield lesser robust features. The camera used in the experiment is UI1221 and the same model is used in other research groups such as the ETH[10] and TUM[90]. The shutter speed is set to minimum and mounted with orthogonally placed polarizers. Still, It is not enough for this work. They rely on 3D features that are not trackable in the scene.

(2) The Landmark-based EKF estimation will usually cost more memory and CPU usage. Each observation will add 3 variables in the state. After finding all residual covariance, the covariance inverse will be too large to be solved in real time. Consider a numerical sample of 1000x1000 matrix originated from 300+ feature points, it takes Eigen library 1.5 seconds to find its inverse. Therefore only a lesser set of feature can be used which can lead to unstable performance. When dealing with raw features and motion, there is a high chance they are highly nonlinearly coupled. So, a UKF or Graph-based method will be a better choice.

Their work was never able to be reproduced in the Singapore outdoor environment because the lighting is too strong to provide good depth estimation. For the indoor test, MYNT-EYE was used for testing due to the large baseline of outdoor stereo. Their method[48] still drifts heavily indoor. There are many other people who also reports similar tracking failure in their repository[270].

FIGURE 5.13: Challenging environment in Singapore. Top images from their paper and their testing facility in Guardian Centers training facility in Perry, Georgia, US. Bottom images from Singapore free fly zones and Murai Urban Training Facility. All images are obtained legally by different means

5.3 Overview of Proposed Method

State of the art is not robust enough. From Figure 5.3, it can be easily seen that bats can navigate relatively robustly and accurate from fusing all sensors. Bats fly in a swarm and can pinpoint small insects for food. Those bats who failed to do so died to the natural selection. Therefore nature already demonstrated what's the best way to achieve robustness and accuracy. The method is to use all sensors in the state estimations jointly. The proposed system can be seen from the overview

FIGURE 5.14: Overview of the proposed Visual LIDAR Inertial Compass fusion system.

in Figure 5.14. The red block denotes the Dual LIDAR-Inertial based navigation subsystem. The blue block indicates Visual-Inertial navigation subsystem. The green block denotes the navigation method used in conventional commercial manual flight. By using the Extended Kalman filter to merge observation from multiple sources, a more accurate localization result can be achieved. To use any Kalman fusion based method, a robust measurement covariance has to be found. The cost function previously used for the other works show no significant difference between good state estimate and bad state estimate result. Since the error function shows

no difference, the traditional way of bundle adjustment cannot be used further for improving the outcome.

Therefore a new intrinsic variance measure is proposed to determine the likelihood of a good state estimation result. Similarly to the LIDAR-based approach, an analysis is done between the working case and failure case. A reasonable variance is generated for the LIDAR-based state estimator output. Based on two incorrect or imperfect source, better state estimate can be achieved as shown in Figure 5.43.

Each part of the system will be introduced separately in each section of this chapter. And the cases where part of the sensors fail will be discussed in details in later sections. The episode can be broken down into 5 sequences as shown in Figure 5.15.

FIGURE 5.15: Chapter Content Breakdown.

5.4 Visual-Inertial Based State Estimation Subsystem

Visual-Inertial Based State Estimation Subsystem primary uses camera and IMU to find the pose. By combining corresponding stereo depth for each pose, a global mapping can be achieved. This type of navigation system can be seen from a series of works[45]. It is a popular research direction in lightweight drone state estimation study.

5.4.0.1 IMU Modeling, Riemannian geometry and Newtonian laws of Motion

For both vision and fusion-based state estimations, the same IMU model is used. It is considered the Newtonian laws of Motion with Riemannian geometry embedded. In this part, the general model is provided for the conceptual understanding and actual implementation in each subsystem will have a minor difference. The IMU measures the linear acceleration and angular velocity of the IMU body frame. Some IMU type directly provides angular measurement.

Let the w denote the angular measurements and a denotes the linear measurements. Let $\tilde{x}(t) = [\tilde{w}(t), \tilde{a}(t)]^T$ denote the estimated state from true state $x(t) = [w(t), a(t)]^T$, time-dependent bias $x_b(t) = [w_b(t), a_b(t)]^T$ and measurement Gaussian white noise $x_n(t) = [w_n(t), a_n(t)]^T$.

$$\tilde{x}(t) = x(t) + x_b(t) + x_n(t) \quad (5.1)$$

The measurement noise is given by $x_n \sim N(0, \sigma)$ and can be characterized by using first and the second moments

$$E[x_n(t)] = 0_{3 \times 1} \quad (5.2)$$

$$E[x_n(t + \tau)x_n^T(t)] = \sigma_{x_n} I_{3 \times 3} \delta(\tau) \quad (5.3)$$

where δ is Dirac function and σ_{x_n} hold different value for angular velocity and acceleration. The measurement noise will change overtime and forms a new bias. This bias changes can be modeled by random walk (aka Winener process or Brownian

motions). It is defined by $x_r(t)$.

$$\dot{x}_n(t) = x_r(t) \quad (5.4)$$

where the $x_r(t)$ is measured by the Gaussian white noise

$$E[x_r(t)] = 0_{3 \times 1} \quad (5.5)$$

$$E[x_r(t + \tau)x_r^T(t)] = \sigma_{x_r} I_{3 \times 3} \delta(\tau) \quad (5.6)$$

The noise characteristics are depended on IMU used and the time interval. For discrete type measurement, variance is discretized by

$$\sigma_d = \sigma \sqrt{\Delta t} \quad (5.7)$$

The unit for gyro white noise, accelerometer white noise, gyro random walk noise and accelerometer random walk noises are represented by $\frac{rad}{s\sqrt{Hz}}$, $\frac{m}{s^2\sqrt{Hz}}$, $\frac{rad}{s^2\sqrt{Hz}}$, and $\frac{m}{s^3\sqrt{Hz}}$ respectively. The sample error caused by the drifting bias can be seen from Figure 5.16.

	Accelerometer Bias Error		Horizontal Position Error [m]			
Grade	[mg]		1s	10s	60s	1hr
Navigation	0.025		0.13 mm	12 mm	0.44 m	1.6 km
Tactical	0.3		1.5 mm	150 mm	5.3 m	19 km
Industrial	3		15 mm	1.5 m	53 m	190 km
Automotive	125		620 mm	60 m	2.2 km	7900 km

FIGURE 5.16: IMU based position estimation error for a static placed IMU. The error is due to drift and bias in the IMU modeling section. Data provided by VectorNav - A IMU manufacturing company

Newtonian Laws of Motion with Riemannian Geometry The detailed formulation is slightly more complex because of Riemannian geometry. The linear measurements a needs to be rotated differently at each discrete time update interval in order to trace its true location and path. And there is always a gravity vector g which is applied to the system which needs to be removed. With all the

consideration in mind, the detailed formulation for finding IMU rotation and linear measurement is shown below:

$$\tilde{w}(t) = w(t) + w_b(t) + w_n(t) \quad (5.8)$$

$$\tilde{a}(t) = R_w^T(t)(a(t) - g) + a_b(t) + a_n(t) \quad (5.9)$$

where $R_w^T(t)$ denotes the rotation matrix $R \in SO(3)$ generated from Euler angle vector of estimated $\tilde{w}$ at time t by the Rodrigues' rotation formula $EXP(w) = R_w$. The g denotes the gravity direction $g = [0, 0, g_{earth}]^T$

Then the state vector of next sensor frame can be represented by:

$$p(t + \Delta t) = p(t) + \int_t^{t+\Delta t} v(\tau) d\tau + \int \int_t^{t+\Delta t} a(\tau) d\tau \quad (5.10)$$

$$v(t + \Delta t) = v(t) + \int_t^{t+\Delta t} a(\tau) d\tau \quad (5.11)$$

$$R(t + \Delta t) = R_w(t) Exp\left(\int_t^{t+\Delta t} w(\tau) d\tau\right) \quad (5.12)$$

Where the p denotes the position and v denotes the speed, and R denotes the orientation. The double integral means the distance covered by multiplying acceleration with the squared time interval. And the Exp denotes the exponential mapping as introduced in Chapter 2. The sample result of this equation can be found in Table 5.16. This model will output drone current position and orientation states from IMU report. The rotation matrix can also be replaced by quaternion cross product if quaternion angle representation is used[12]. This model depends on initial estimates and accumulated acceleration output and can be erroneous. When only use Equations 5.10, 5.11 and 5.12 on local reference frame. The accumulated error can be removed and the state only depends on adjacent states. This method is commonly known as IMU pre-integration[271].

5.4.0.2 Visual-Inertial State Estimation

There are many Visual-Inertial-based navigation solution such as [73][45][79][96][77]. Most of the formulation such as the popular OKVIS[45] and VINSMONO[12] are the same. The difference is in model and implementation details as can be seen from Figures 5.8, 5.9, 5.10. In in-house experiment, VINSMONO[12] based method achieve highest accuracy to CPU usage ratio(without loop closure) which is ideal for low computation source lightweight drone as seen from Table 5.1. The measurement here is in CPU core usage. Therefore, work from [12] is used for visual state estimation.

Category	VINSMONO	OKVIS
CPU usage	83 % $\pm$ 15%	130% $\pm$ 20%
Processing delay	70 ms $\pm$ 10ms	30 ms $\pm$ 10 ms

TABLE 5.1: Performance Comparisons between OKVIS and VINSMONO using custom set of parameter

VINSMONO[12] is a relative large work with about 15k lines of codes. It is quite tedious work to cover all their content in details. Their original paper is only 17 pages. However, the proof, derivation and implementation detail originated from dozens of previous works[265][271][45][272][273][112][274][275][75][276][277][94][125][278] and it is not trivial tasks to cover all the details in this Chapter. In chapter 2 of this thesis, there are more details.

In this chapter, only a brief introduction will be provided. VINSMONO can be broken down into five major steps namely (1) Feature detection and tracking, (2) IMU pre-integration, (3) calibration and initialization, (4) runtime optimization and (5) Loop Closure. For some steps like (3) and (4), there are multiple sub-steps involved.

Step (1) Feature detection and tracking on a unit spherical model. The feature detection was done using corner detection[125] on each image. The features were then tracked over next frames using LKT[94] optical flow based method. The bucking method[118] was used with 30pixel radius. All features are projected to optical center with unit spherical projection model. The features were aligned and filtered with RANSAC models. The missing features are resupplied with newly detected

Step (2) IMU-based position estimation. The IMU pre-integration is formulated using standard Newton Laws of motions with rotation embedded [265]. Based on the formulation, the IMU Jacobian is found by finding the derivative of the current state vector with respect to previous state vector. This is one of the most effective methods to remove noise concatenated in the IMU state estimations

Step(3) Calibration and initialization. This step has many sub-steps. The first

sub-step is to find or enhance the initial camera to IMU extrinsic guess by using Robot-Hand-Eye calibration method with IMU pre-integrated pose updates. The second sub-step is finding initial 3D observation and rotation by moving the cameras around. This step is commonly known as SFM based approach. The third sub-step is to use initial 3D observation and rotation to align with IMU measurement to find the IMU bias. The forth sub-step is to align the velocity, gravity, and scale till the gravity reaches between 8.8 to 10.8. Then the next step is to use a linear optimization method to find velocity, gravity and scale model so that the gravity measure is close to 9.8. Then the last sub-step is to optimized the local map based on accurate IMU model, camera model and scale measure to find initial 3D observation and poses.

Step(4) Runtime optimization. The first sub-step is to find whether the current frame is a keyframe. A frame is considered as keyframe if either too many frames has passed or too much distance has been traveled. The default windows size is 10(represent by queue). If the feature has enough displacement for the incoming frame, then the first position in the queue will be discarded and the new frame will be the 10th frame(keyframe). If the features do not have enough displacement for the incoming frame, then the last position in the queue will be discarded and the new frame will be the 10th frame(non-keyframe). No matter which case, the new frame will be the 10th frame in the local windows. This method is also called marginalization. The second sub-step is to add new feature points. A point is good 3D observation if it is tracked over multiple frames and can be triangulated.

The next sub-step is the core of the system which is the minimization of pre-integration residual and reprojection residual. After this step, the pose for the latest frame will be obtained. Meanwhile, Schur Complement is applied to the least squares equation to marginalize out the landmark observation parameters. It is done so that the calculation can be transformed from a general MAP problem into efficiently solvable matrix inverse. At last, based on the first tracked position and final traced position, the optimal inverse depth can be obtained.

Step(5) Loop closure. Loop closure is not used in multi-sensor fusion experiment because it hangs after 5-10mins of running. The idea for their loop closure is to use DBow model on the FAST feature of every three frames to find matching points.

Therefore the memory will overflow eventually.

The VINSMONO is an active update work with different updates every year. This analysis is done based on their July 2017 release. The newer version including many changes such as introducing `relocalize_r` and `relocalize_t` in the loop closure to ensure historical pose accuracy. There are too many details to be covered in this chapter. However, there are still a few points that can be improved which will be introduced in later section.

This work is currently one of the states of the art in the visual-inertial fusion system. VINSMONO requires moderate CPU usage and have good state estimations. The OKVIS requires more CPU usage, but it can get better rotation estimates when dealing with extreme lighting conditions. The ROVIO is the fastest method, but the accuracy is low. It is difficult to say who is best among them. To the drone used in this work, VINSMONO is preferred for its accuracy to CPU ratio.

5.4.1 Analysis - the Problem, the Observation and the Solution

Condition	Sample cases	Reason
Works Well (Minor errors)		Enough Lighting Not too bright Not too dark
Partially Work (erroneous)		12 noon strong lighting condition After 6 with low light condition
Failure		Pitch dark area Featureless area Lighting-inconsistent

FIGURE 5.18: Problematic scenes with vision-based state estimator

Problem For every work, there is always room for improvement. For example, VINS-MONO[12] was a direct improvement over their previous work[267][274] by adding minor accuracy items from a unit spherical projection model and etc. There is still room for improvement for their current method. Here the analysis is done on a boarder spectrum of all visual-inertial-based state estimation. Currently, all visual-inertial-based state estimation works in a limited scene as seen in Table 5.18. When lighting is sub-optimal, the results can fail fast. This is an embedded problem with all vision-based approach.

One of the applications for the drone is surveillance. Is it possible to tell the terrorist with ” oh, sorry, please come and attack again in the morning because our drone is not operational with low light” when the terrorist starts the attack at night? The answer is clearly negative. There must be a way to increase the robustness.

Possible Solution There are always plenty of room for improvement as shown in Figure 5.19. For the assumptions, there are chances of skipping it by using specific hardware and corresponding models such as the event camera. Event cameras are bio-inspired vision sensors that output pixel-level brightness changes instead of standard intensity frames. The event camera¹ can solve the problem easily

FIGURE 5.19: Possible new improvement

with sparse edge based feature model. Event cameras can work at tough lighting conditions and independent of photo-consistency assumption. But they are not available commercially before late 2018. For system level state estimation error, there is a chance of removing them by using a hierarchical graph model.

FIGURE 5.20: Robust tracking from Event Camera. Image from ETH group.

¹Event camera demo <https://www.youtube.com/watch?v=jIvJuWdmemE>

5.4.2 Attempts to Solve the Problem

5.4.2.1 Failure Attempts - Hierarchical Graph Model

The other direction is to use multiple layers of the graph model. A full 3D sensor with zero variance observation is provided for initial simulation. A higher level graph node is created with every node 20 meters apart. Based on prediction from the higher level graph, lower level graph states are changed according to higher prediction. The initial measurement was still acceptable. But as distance increases, the prediction goes off and become unstable. The shape of state estimation goes wrong. The overall direction of this approach might be correct. But the implementation details, parameters, and models selection could be the reason for the errors. But when transfer this idea to vision, there could be a bigger problem with wrong feature association. Such problem can already be seen in the image association part of Figure 5.21. The features are wrongly associated when distance is too far apart from each other.

FIGURE 5.21: Failure attempt using Hierarchical Graph Model based approach.

5.4.2.2 Success Attempts - Variance Generation for Vision-based State Estimation

The [12] deploys a multi sensor bundle adjustment based method as shown in equation below.

$$\min_N \{ \|r_p - H_p \chi\|^2 + \sum_{k \in B} \|r_B(\hat{z}_{b_{k+1}}^{b_k}, \chi)\|_{P_{b_{k+1}}^{b_k}}^2 + \sum_{l,j \in C} \rho \|r_C(\hat{z}_l^{c_j}, \chi)\|_{P_{b_{k+1}}^{b_k}}^2 \} \quad (5.13)$$

where $\|r_p - H_p \chi\|^2$ is the prior information from marginalization to bound the computational complexity. r_b and r_C are the error residual for IMU and visual measurement. $\chi = [v_{b_0} \dots v_{b_n}, g^{co}, s]$ denotes the velocity, gravity vector and scale in the local windows. and ρ denotes the Huber norm. This error is minimized by using Ceres Solver[151].

FIGURE 5.22: When lighting condition changes, Vision-based state estimator completely collapses.

Under the general case where lighting is healthy, this elegant solution will have no trouble navigating around. However, in real work, lighting is not consistent. There are cases of extremely low light and extremely bright light as can be seen from 5.22. Under those cases, its the visual tracking part that shows difference whereas the IMU residual alignment is the same.

The VINS MONO[12] uses a semi-direct based method to perform tracking. The best rotation R and translation t are obtained by aligning a sequence of tracked images. Such a case are generally adequate for normal lighting.

The good case and bad case of error of $\|r_C(\hat{z}_l^{c_j}, \chi)\|$ for adjacent feature when under perfect alignment are shown below in Figure 5.23.

FIGURE 5.23: Error correlation between a good result and bad result are similar which means more iteration will not reduce the error and will not improve the result.

Between good state estimator result and lost results, the error is the same. When using other feature types, the error under perfect alignment is similar. It means, the features are easier to mixed and wrongly tracked over bad lighting condition. To solve that, the intrinsic property of the detected feature are used to determines how good a tracking will be as can be seen from Figure 5.24. Let I denote the input images. the measurement over difference in intensity for displacement (u, v) are shown

$$E(u, v) = \sum_{x,y} w(x, y) \|I(x + u, y + v) - I(x, y)\|^2 \quad (5.14)$$

where w denotes the windows weight function. By applying taylor expansion to E , the simplified solution can be find by

$$E(u, v) = [u, v] M \begin{bmatrix} u \\ v \end{bmatrix} \quad (5.15)$$

where M is defined as

$$M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \quad (5.16)$$

Let eigen image $R(x, y)$ be

$$R = \det(M) - k(\text{trace}(M))^2 \quad (5.17)$$

where $\det M = \lambda_1 \lambda_2$, $(\text{trace}(M)) = \lambda_1 + \lambda_2$ and λ_1, λ_2 are the eigenvalue vector of M .

For semi-direct based method, top N (usually 100-300) strongest selected from normalized eigenimage $\hat{R}$. Moreover, for robustness reason, there should always K (90% of N) feature used for optimization in the bundle adjustment where $K \simeq N$. However, this only grants a minimum robust tracking but does not grant accurateness of state estimator output.

If a higher tracked feature collection with the lower number j feature used where $j \ll N$, there is a high chance the whole state estimator thread can crash. Such case can easily be seen when $j < 10$.

FIGURE 5.24: A better indication for good and bad Visual-Inertial state estimator performance.

By finding the number of repose higher than a threshold t in R (without normalization), a set of points can be found. The number of those points is a better representation of good and bad state estimator result. The cases can be seen from Figure 5.22. Take note that at the end of the bad sequence, the lighting condition is not that bad. So the state estimator still works in the end. For the good sequence, the lighting is too strong for the state estimator at the end of the sequence. That is why there is a cross in the graph. The graph is accurate to measure how well the vision-based state estimator performs. But this variance cannot be used directly in the least squares fit as there is no direct covariance correlation between feature extraction and optimization part. To force the additional variance from tracking into the optimization process will cause an additional error in the optimization function, which the error may not get reduced over iteration. In the end, it will appear as hanging.

5.5 Dual LIDAR-Inertial-Based State Estimation Subsystem

There was a massive amount of work using vision, IMU, and GNSS to navigate the drones. Those type of methods can fail very quickly when dealing with the featureless area or doing at night as can be seen from Figure 5.22. Moreover, night ops cases are very often seen in both civilian and military cases. For example, roller coasters in the Singapore Universal studios often find some screw missing and cause panic in public. During the daytime, there is no window of opportunities to inspect. For inspection works, there are still chance that concert light stick can be placed on the ground for adding features to the method such as [10]. But this type of method requires more manpower to place and cleanup. In those case, only active sensing methods such as LIDAR or Kinect can work. Kinect has a problem running in daytime due to IR pattern interference. Therefore, a drone with LIDAR based solution is the good choice.

5.5.1 Proposed Subsystem Description and Analysis

The overview of the proposed method can be seen from Figure 5.25. This navigation system uses 2D LIDAR with laser scan matcher to find x , y , and yaw of the drone. The roll, pitch, and altitude are estimated from IMU and vertical LIDAR. For vertical LIDAR case, there are multiple sensor types to select from. Each type of sensor can provide different accuracy and weight combination. Different combination offers different application by using an alternative size drone. Because of algorithm limitations, parameters such as altitude can be missing for a few frames. It is quite often to happen when dealing with the complex conditions on the ground. To ensure the robustness of the proposed system, a Linear Kalman based filtering system is proposed for robust position estimation. The drone setup and obstacle view can be seen from Figure 5.26. In both day and night, indoor or outdoor, this drone can perform without any issue.

The extrinsic parameter are found by method from previous chapters. Please refer to previous section for more detailed explanation

FIGURE 5.25: The overview of proposed Dual LIDAR-Inertial-based Navigation Subsystem

(A) LIDAR drone position hold

(B) RVIZ view

FIGURE 5.26: Proposed LIDAR based autonomous drone system with pilot hands free. The image was taken in controlled civilian private airport area at longitude 1.405525 latitude 103.865645 facing East-South direction.

5.5.1.1 Line based Iterative Closest Point matching

The merit of this type of method is that the drone can have higher accuracy in the rotation and faster convergence rate to the true pose update in enclosed rooms. The shape of generated odometry is usually more accurate in a static scene, as seen from Figure 5.27. The line based ICP holds similar shape as visual-inertial odometry in a normal lighting condition.

The limitation with this type of method is that when the drone meets the infinitely long walls beyond the observation range of current LIDAR and there is no feature to match. The LIDAR-based state estimator will drift along the wall and be less reliable in accuracy. This can be seen from Figure 5.27 where there is a section of LIDAR-based state estimator lost track over a period of 5 sec.

FIGURE 5.27: Testing ICP with and without line information and compared with vision based state estimation under relative good lighting condition

5.5.1.2 Altitude Estimation

Traditionally altitude is estimated by barometer or GPS. Concerning accuracy, those methods are least preferred. The altitude estimation is done by using a vertical mounted LIDAR. By measuring the distance toward the ground. For the altitude, there are different ways to measure it. The simplest way is a point to point measure. The more complicated way is measuring the distance towards a line. The most robust and computational intensive way is to estimate the distance to the surface.

One-D Point LIDAR uses the direct measurement from the sensor to determine the height. Due to the control, the drone will have a pitch and yaw effect. The estimated height will have a much larger variance. It is commonly used on lightweight drones in FLA program².

²DAPRA FLA program <https://www.youtube.com/watch?v=vDYy3L9nvLk>

FIGURE 5.28: The raw ground line model detected is shown in red

Two-D 2D LIDAR uses line extraction technique for altitude estimation which can be seen from Figure 5.28. By finding distance parameter over the squared error to the line parametric equation, the vertical orthogonal height can be found. But the assumption is there is no roll movement. When the roll is present, it is as inaccurate as point LIDAR. But when roll control is disabled. The accuracy can be as close as the 3D LIDAR version. There are multiple methods to get line like RANSAC, Hough transforms or split and merge. After a few tests, split and merge is shown to be more robust than other methods.

Three-D Drone with 3D LIDAR as shown in Figure 5.29 uses RANSAC based plane extraction technique for altitude estimation. By finding distance parameter over the squared error to the plane parametric equation, the plane orthogonal distance towards the drone can be found as can be seen from Figure 5.30.

The drone is set to hover at preset altitudes. While drone is at preset altitude, horizontal commands are given to measure how well can drone maintain its height during execution of other commands. The external measurement is done by fixing an Xtion pro depth camera on the ground facing toward the sky at night. Xtion

FIGURE 5.29: The drone used in this work which mounted with 3D sensor pointing downwards and on-board processing unit. Image taken at Tengah Airbase on February 27, 2018 by a journalist. Declassified for publication use

FIGURE 5.30: The ground model being detected. The colored point cloud is ground model. White denotes outliers. Bottom left and right are measurement from thermal and ZED image. Image taken at 1.291366, 103.786834 heading North East

is able to see at night for about 10m maximum range. Let d_i denotes the depth captured at time i . The drone position is measured by mean depth in the depth map $\hat{d}_i$. By measuring the variation during the execution of other commands, a table can be drawn as shown in Figure 5.2. It worth pointing out that this measurement may not be accurate as the drone is suffering from external disturbance like winds. However, this is the most suitable method for the reason that no room with ground truth motion capture systems(VICON) has that height. The GPS and barometer based method has a higher error than LIDAR based methods. Therefore, they can not be used in ground truth reference for analysis.

The testing result shows that the 1D LIDAR based method is most inaccurate. However, only 1D LIDAR can be used on ultra lightweight drones such as Gen 1 drone in Figure 5.31. The 2D version shows a more significant error during doing pilot based navigation with roll movement. When controlling the drone fully autonomously by disabling the roll parameter, the accuracy can be as high as a full 3D LIDAR based method. The 3D lidar based method is so far most accurate measurement, but the weight requirement is the highest. Also, the computation cost for 3D LIDAR is the highest.

Distance	LIDAR 1D	LIDAR 2D $\varphi \neq 0$	LIDAR 2D $\varphi = 0$	LIDAR 3D
1.5m	10.43cm	8.99 cm	8.62cm	7.45cm *
2.5m	13.80cm	12.92cm	10.88cm	10.42cm *
3.5m	22.01cm	19.52cm	11.86cm	10.98cm *
4.5m	28.75cm	23.24cm	11.18cm	11.61cm *
5.5m	43.98cm	37.95cm	14.24cm	12.89cm *
6.5m	72.80cm	58.09cm	16.69 cm	15.18cm *

TABLE 5.2: Altitude variance at a different height when the drone has horizontal movement after the control loop compensation. * denotes data gathered from Gen3 platform where others are measured from Gen2 in Figure 5.31. All data captured from longitude: 1.291376 latitude: 103.786854 with the camera placed on the roof and facing towards the sky. * taken at a later time at the approximately same location

It is worth noting that when extracting lines from 2D Lidar, multiple methods are tested such as Hough transform, 2D RANSAC, and split and merge. All methods fail at certain times, the split and merge method fail the least amount of time. There is still a possibility that the line cannot be detected. Therefore a simple Kalman filter is required to smooth the occasionally happened failure.

The first stage of estimation happens when drone just takes-off. The altitude reference is directly from the ground. When the drone gets higher in the building, there is a high chance that ground detection will not be feasible. Therefore, the second stage roof is used on top of the ground for robust navigation. Such a configuration is enough for most of the indoor case because very few single rooms reach 40 meter high. For the Boeing 777, tail height is about 18 meters. For the inspection tasks in the Boeing 777 hanger, the 20m 2D LIDAR model is enough for finding its position.

5.5.1.3 Linear Kalman Fusion

To solve the missing ground problem, a simple linear kalman filter is proposed. let $x_k = [P_x^k, \dot{P}_x^k, P_y^k, \dot{P}_y^k, P_z^k, \dot{P}_z^k]^T$ be the state vector and $u \in [-1, 1]$ denotes user input. The $\dot{P}$ denotes the speed of robot on a particular axis. The linear Kalman filter model can be formulated as

$$x_k = F_k x_{k-1} + B u_k + w_k \quad (5.18)$$

$$y_k = H_k x_k + v_k \quad (5.19)$$

where F denotes state transition model to the previous state, B is the control-input model and H denotes the observation model. In earlier work in PX4, this control model can be found in open source implementation. In recent drone, u is ignored because FCU like the DJI A3 closes all its control dynamics codes. The w_k and v_k denotes process noise model and observation model.

For the cheap IMU, the acceleration can cause a large drift as seen in Figure 5.16. Therefore, a constant acceleration assumption a_k is applied where a_k is considered as normally distributed. The actual equation of x_k can be represented using Newton's Laws of motion which closely resemble the standard Kalman filter equation.

$$x_k = F x_{k-1} + G a_k = F x_{k-1} + w_k \quad (5.20)$$

where F and G can be represented by

$$F = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$G = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \\ \frac{\Delta t^2}{2} \\ \Delta t \\ \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}$$

The process noise w_k is assumed to be zero mean multivariate normal distribution N , with covariance Q_k : $w_k \sim G \bullet N(0, Q_k)$. At each stage, a floor measure could be missing which forms a noisy measurement. Assume the measurement noise v_k is Gaussian distributed with zero mean and deviation of σ_z

$$Z_k = Hx_k + v_k \quad (5.21)$$

where

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

and observation noise covariance $R = E[v_k v_k^T]$. The initial estimation state is set to be all zero. The initial covariance is set with $0.001 * I_{6 \times 6}$. The rest of covariance update or gain update using standard methods. As long as the covariance matrix of Q does not fall into degenerate distribution cases.

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

The measurement update is given by

$$\hat{y}_k = z_k - H_k x_{k|k-1}$$

and covariance can be found by

$$S_k = R_k + H_k R_{k|k-1} H_k^T$$

The Optimal Kalman gain is updated as

$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$

The state, state covariance and measurement are updated again. The state is updated again by

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \hat{y}_k$$

This kalman filter is very simple but very important step for a stable drone performance.

5.5.2 Experiments

There was three evolutions of LIDAR-based drone systems. However, the concept comes from much older work [279] during authors undergraduate study using center line of Kinect for single LIDAR scanline. The work is for a competition and therefore can not use expensive LIDAR. The Kinect version suffers from a narrow-angle observation robust tracking problem. Therefore, wide-angle LIDAR is used when the budget was sufficient. The first LIDAR version uses 3DR PX4 controller on a custom made Quad Copter platform. The 3DR Ardupilot based method opens its code. And most of the parameters can be modeled. The second generation uses DJI N3 flight controller on a CNC cut custom made OctoCopter with inverted X configuration drone. The latest generation uses DJI mass production M210 model for the task. If M210 crashes due to an internal fault, it can be exchanged by DJI for free. This feature makes it very appealing to software designer.

The Gen one uses point LIDAR for altitude estimation. The Gen two and Gen three can use 2D LIDAR for altitude estimation. The Gen 3 version is able to carry a single 3D Velodyne LIDAR for accurate altitude estimation. For Gen3 drone, it

comes with a default stereo camera which sparks the idea of Visual LIDAR Inertial and Compass fusion method. The drone for each generation can be seen from Figure 5.31.

FIGURE 5.31: Three generation of development for LIDAR based drone navigation

For all generations, tether solutions are coupled for long duration flight and safety control.

Multiple flight test was conducted using a single PC. The subsystem flight test was proven working with HP OMEN notebook, IBM t440 notebook, and Intel i7 NUC single board PC. Data are transmitted from the tethered optical fiber when using with a notebook. The overall configuration for the lidar-based drone flight is shown in Figure 5.32.

Take note that RRT based planning [280] or A* & DWA [281][282] based planning are not triggered at the same time. Moreover, they are being called upon on need to use basis. For inspection, the majority of time velocity control is used. When pilot drops control and u become all 0 vectors, drone assumes hover position by the Kalman based feedback. When u is going to crash in a certain direction with obstacle detected, u from the pilot in that direction are attenuated. Wired velocity control is connected by Futaba trainer cable. u model are mapped over the DJI wireless version. 3D pose is given in from RVIZ(x, y, θ) and reconfigure(z). The extra wired control is meant for metal wall microwave oven effect and EM spectrum jamming. The electromagnetic radiation emitted from the wireless controller can aggregate and fry the wireless controller itself.

And the system can run in real-time. The minor difference in performances is expected by various drone hardware. The latest version on 5.9kg M210 can hold a position in an indoor enclosed environment with 5cm accuracy. The maximum

FIGURE 5.32: Autonomous UAV control and navigation model

take-off weight is about 6.1kg and there is a payload camera on top which is continuously moving. The M210 drone directly inspired the multi-model fusion works in this chapter because there is two stereo cameras embedded within this drone.

5.5.3 Qualitative and Quantitative Result

The result can be seen from Dual LIDAR-Inertial based state estimator and map build from the position feedback as seen in Figure 5.3.

Current view in image and map on better day	Status
<p>Initialization in the bottom of the ship. The part connected with blue section is a pipe.</p>
<p>Set goal to Target Position $[0,0,2]$ Rotation $[0,0,0]$. The map start to build up. The state estimation working well</p>

<p>Set goal to Target Position $[0,0,5]$ Rotation $[0,0,0]$. The map is continuously added.</p>
<p>Set goal to Target Position $[0,0,10]$. Most of the ship hull is mapped</p>

TABLE 5.3: Map created in LIDAR based autonomous drone navigation. Ship GPS position unknown, Panamax Class, Front Left Deck

The final result can be seen in Figure 5.33. Previously sparse OCTOMAP is used for real-time navigation. The goal was given by ROS interface. There are plenty of memory limits for small onboard PC. The dense 3D point cloud can only be plotted after the flight. The LIDAR based state estimation takes less than 50% of a CPU core to process.

FIGURE 5.33: Dense point cloud reconstruction after flight inside ship

The in-ship sequence has no comparable result. Inside the ship, it is one of the most challenging environment for an autonomous robot. No ground truth or other sources of state estimation was recorded due to reasons:

- (1) Vision does not work at all.
- (2) VICON does not work because of no power line inside.
- (3) UWB does not work because the metal surface reflects signals and forms multi-path problem.
- (4) GPS does not work because the hull is enclosed. And GPS cannot penetrate the hull.

For the LIDAR sequences captured from 1.293566, 103.787552, each run lasts about 100 over meters. The GPS from drone drifts extensively. The starting and ending positions of drones are marked down physically. The GPS was mounted on poles of 2 meters high. Each time, pole stationed vertically at the marked position for at least 30 min to find the average value of GPS. The accuracy table with multiple sequenced are compared as shown in Table 5.4.

FIGURE 5.34: LIDAR generally performs better when compare with Visual-Inertial based state estimation.

These sequences all suffer from featureless scene problem. Some sequences also have large dynamic obstacles. But the lighting problem is the major issue for vision-based state estimations. When using the vision-based state estimations in normal lighting condition the error can be below 20%. But when using the vision-based state estimations in some tough lighting condition, the error can go up to 40%. The LIDAR performs relative more consistent with around 10% of error despite the featureless scene.

The quantitative result can be seen from Table 5.4. The qualitative result comparison can be seen from Figure 5.34. The dark sequence was taken after 18:00 and

Sequence No.	LIDAR error	Vision error	Remark
1	9.95 m	15.48m	featureless(LIDAR)
2	9.61 m	51.27m	Dark,dynamic,featureless(both)
3	7.49 m	12.85m	Dynamic scene
4	11.66 m	14.54m	Bright Featureless(Both)
5	5.86 m	10.14m	Dark Featureless(Both)

TABLE 5.4: Performance Comparisons between LIDAR based approach and vision based approach

after rain. The bright means lunchtime super bright lighting condition. The dynamic means dynamic moving object. Under all those real imperfect testing cases,

LIDAR-based approaches show the more accurate result when compared with the visual-inertial based approach as can be seen from Figure 5.34.

5.5.4 Existing Problem and Solution

The proposed LIDAR based approaches can work at both day and night with high accuracy when next to target building. The overall weight is light enough that can be running without requiring multiple permits from CAAS. There are only a few cases where this drone would not work properly such as (1) Featureless scene, (2) in presence of glass, rain, smoke and other LIDAR.

At planer scene, LIDAR-based state estimator is prone to drift as can be seen in Figure 5.35.

FIGURE 5.35: Featureless scene VO lost. Red represent the Vision-based state estimator and purple denotes LIDAR-based state estimator

The red vision-based state estimators still being updated, and the robot is moving according to the image flow. However, the LIDAR did not update at all due to this featureless scene. The LIDAR can sense up to 20m. There are multiple cases where the featureless scene is larger than 40 meters which causes the failure. In general, LIDAR produces robust state estimates. It is only at this featureless scene, vision-based state estimator is a better choice.

Variance Generation for LIDAR-based State Estimation The variance of LIDAR measurement is empirically found by σ_{LIDAR}

$$\sigma_{LIDAR} = \sum_w \frac{k}{n+1} \quad (5.22)$$

where k denotes the number of non-zero and none-infinity points from a current scan s . n denotes the number of detected straight line for point to line based iterative cost point matching. $n+1$ is to avoid cases where no line is detected. When the number of points is large, and there are no lines, LIDAR-based state estimator variance should be high. When the number of lines is high, the LIDAR-based state estimator should have low variance.

Such case can be seen from Figure 5.36 below.

FIGURE 5.36: Dual LIDAR-Inertial based state estimation variance grows when dealing with large planer scene.

5.6 Fusion-Based State Estimation Subsystem

Both visual and LIDAR-based system has flaws when dealing with some part of the environment. The critical error happens when there are faults in the fundamental feature associate stage. Therefore, it is logical to fuse them together based on empirically found covariance. There are multiple ways to do so. SVO[10]+MSF[95] use an absolute position based fusion method for binding visual and inertial state estimation feedback. When dealing with two sources of absolute position measurements, they cannot be put together by just finding its average position. Only the velocity and acceleration model can be directly fused which is similar to the pre-integration concept of [12]. The fusion is done by using a state vector without landmark features on EKF-based model. By now, there are two sets of position measurements and two sets of variance measurements waiting to be fused.

5.6.1 Input State Analysis

Inner Sensor Analysis Inner sensor analysis is done to provide the system with the knowledge that when a particular sensor state estimate should be used and when should not be used. The input from multiple sources at frame k are considered as observation z_k^{sensor} . They are estimated by previous location, marginal increment, and observation errors. The the full position observation model can be denoted as $\tilde{z}_k^{sensor}$:

$$\tilde{z}_k^{sensor} = z_{k-1}^{sensor} + v_{k-1}\Delta t + \frac{1}{2}a_{k-1}\Delta t^2 + E_{k-1}^{sensor} \quad (5.23)$$

The $z_{k-1}^{sensor} + v_{k-1}\Delta t + \frac{1}{2}a_{k-1}\Delta t^2$ denotes the true estimate from previous observation and E^{sensor} denotes the accumulated error or random walk error term. And the error terms is assumed to be accumulation of zero mean normal distributed noise from each sensor $E^{sensor} \simeq E^{sensor} + N(0, \sigma^{sensor})$. In worst case, the observation can be approximated as accumulated random walk error, when the noise variance is large

$$\tilde{z}_k^{sensor} \propto E^{sensor}$$

The idea can be seen from Figure 5.37 as the noise can build up and goes to any direction. When the sensor source state estimation is accurate, then the observation

FIGURE 5.37: Possible wrong path cases.

from sensor source should be close to internal sensor fusion estimation within the time interval Δt . This concept can be represented by measuring the Mahalanobis-distance between state estimates and observations. It can be formulated by

$$d(\tilde{z}_k^{sensor}, H\tilde{x}_k) = \sqrt{(\tilde{z}_k^{sensor} - H\tilde{x}_k)^T P^{-1} (\tilde{z}_k^{sensor} - H\tilde{x}_k)} \quad (5.24)$$

where the d is distance measure with no unit and P denotes the covariance matrix, x denotes the state vector. There are four possible outcomes from this equation (1) If distance error is significant and covariance is high, then the input state should not be trusted or even discarded. (2) If distance error is substantial and covariance is little, then fusion system should give current input state estimate a reasonable consideration for fusion state estimation. (3) If distance error is small and variance is large, then the fusion system should give current input state estimate a reasonable consideration for fusion state estimation. (4) If both factors are small, then this reading is good and should be trusted. This method is to be applied to all of the input sensors state outputs. The value to discard is currently set to a high value because it is a measure for all states. Is it an extremely effective measure for removing fast drift error in some of the visual-inertial state estimations. The worst case is that both input state covariance is high, then the fusion state depends heavily on IMU-based state estimation with constant biased variance. It is noted that this worst case never happens and there will always be a sensor which has feedbacks with low variance.

Inter Sensor Analysis The inter sensors analysis is done to minimize the joint cumulative error. Both LIDAR and Vision carry a error term E_k^{sensor} where $E_k^{sensor} \simeq E_{k-1}^{sensor} + N(0, \sigma_k^{sensor})$. The longer the system is running, the bigger the

E^{sensor} is and it is propagate forward to next estimate. The drift pattern in Figure 5.37 can go out of control as skys get dark and eventually ends up like pure IMU state estimation drift pattern.

In this case, the difference between visual and LIDAR state estimator will be enormous. If mean value is found between two inputs, the mean value will be heavily biased toward the bigger error side. This type of fusion is the worst as it is heavily biased to the wrong estimates. By right, it is desired to move away from a larger E^{sensor} . By taking the derivative term of observation to sensor individual update interval $\frac{dz}{dt}$, the positions vector in $\frac{dz}{dt}$ will be free of previous cumulative error E_{k-1} . The current observation input will only have marginal normally distributed error as the reference frame is taken to be the closet frames. The position part of the observation vector will be

$$\dot{z}_k^{sensor} = v_{k-1} + N(0, \sigma_k^{sensor}) \quad (5.25)$$

where

$$E_{k-1}^{sensor} \gg N(0, \sigma_k^{sensor})$$

The $\dot{z}_k^{sensor}$ will be the relative motion update in between individual observations. The LIDAR and vision do not output state vector for acceleration. Therefore the Equation 5.23 can be further simplified into Equation 5.25. And all accumulated error is gone. The concept can be seen from Figure 5.38.

FIGURE 5.38: Improved result by remove error propagation.

By compensating each $\dot{z}_k^{sensor}$ measurements with calibration result, the more accurate state estimation can be achieved. At last the LIDAR and Vision state

estimator observations can be expressed by shifting the reference from world origin to closet reference frame.

$$z_p(t) = R_{calibration}^{sensor} \dot{z}_p^{sensor}(t-1) - t_{calibration}^{sensor} \quad (5.26)$$

$$z_w(t) = Log(R_{calibration}^{sensor} Exp(\dot{z}_w(t-1)^{sensor}))$$

where z becomes the velocity measurement vector and the Log and Exp denotes the exponential and logarithm mapping function between the euler angle and rotation matrix. The vector are element wise shifted by 6 element as can be seen from Table 5.5. The observation for LIDAR and camera only have speed measurement. At each time, only one sensor is updated into this equation.

5.6.2 Extended Kalman Filter based Fusion

The EKF formulation [283][284][149][285][66] is one of the well-studied topics in robotics communities. Here, the implementation details for vision, LIDAR, IMU and compass fusion is presented. The overall concept including the input state analysis and EKF fusion can be seen from Figure 5.39.

The drone state vector x is denoted by:

$$x_k = [x, y, z, \phi, \chi, \varphi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\chi}, \dot{\varphi}, \ddot{x}, \ddot{y}, \ddot{z}]$$

And this state vector represents x, y, z, roll, pitch, yaw, x velocity, y velocity, z velocity, roll velocity, pitch velocity, yaw velocity, x acceleration, y acceleration, and z acceleration. The EKF can be formulated by:

$$x = f(x_{k-1}) + w_{k-1}, \quad (5.27)$$

where f is the nonlinear state transition model shown in Equations 5.10, 5.11 and 5.12. and w is the process noise and are assumed to be normally distributed. In earlier drone models, control dynamics can be found. For the later version, the DJI drones hide the forward control model, therefore no u_k term is presented. The rotation model is in Euler angle format. The measurement model z_k is denoted by

$$z_k = h(x_k) + v_k, \quad (5.28)$$

FIGURE 5.39: Fusion process overview

where h is the nonlinear sensor model and v_k is the normally distributed measurement noise. At first state, a priori is estimated by

$$\hat{x}_k = f(x_{k-1}) \quad (5.29)$$

and error covariance $\hat{P}_k$ is find by projecting P through F

$$\hat{P}_k = F P_{k-1} F^T + Q, \quad (5.30)$$

where F is the Jacobian of f , $F = \partial x_k / \partial x_{k-1}$ and then added by noise Q . f is a standard 3D kinematic model from Newton law of motion as can be seen form Equations 5.10, 5.11 and 5.12. Then the Kalman gain can be found by

$$K = \hat{P}_k H^T (H \hat{P}_k H^T + R)^{-1}. \quad (5.31)$$

where the H denotes the measurement matrix and R is the covariance matrix of measurement constructed from previously mentioned measurement variance. Then the updated state can be found by

$$x_k = \hat{x}_k + K||z - H\hat{x}_k||. \quad (5.32)$$

where $||$ denote the Mahalanobis distance. The updated covariance can be found by

$$P_k = (I - KH) \hat{P}_k (I - KH)^T + K R K^T. \quad (5.33)$$

The standard EKF models construct H from the Jacobian matrix of the observation model h . To use this model with LIDAR-based state estimator, Vision-based state estimator, IMU and Compass output, the assumption is made that each sensor can produce the measurement of the state variable in the estimation process and each process is independent. So the H is set to be an identity matrix. To allow asynchronous measurement feedback, the EKF system is configured to allow partial updates of the specific state variable. The feedback measurement does not update all state variable at the same time. When measuring only N variables, the H becomes $N * 15$ matrices with rank N . The non zero value exist in the columns of the measured variables. This is done to further speed up the process.

The difficulties in the EKF based method is the estimation of process noise covariance matrix Q .

5.6.3 Fusion Experiment Setup

For the experiment, REP 103³ and REP 105⁴ standards are strictly followed. In some implementations, map or landmark is used for computing relative pose. Those methods usually have short operation time as memory will overflow. And in the previous section, it is shown that error is caused by incorrect feature association. In these experiments, only individual state output and IMU are used for estimating a more accurate position.

The pose feedback is differentiated to time to find its relative velocity vector $\dot{x} = \frac{dx}{dt}$. The differential process is individual sensor process. In theory, this concept can be jointly used with UWB and GNSS based positioning method. However, at the time of testing, no UWB is available. Moreover, in 20 testing sequences, only two sequences have incredibly wrong GPS measurement. Thus, no absolute global pose graph measurement is used. The raw measurement is collected within 20 meters range of buildings. And the buildings are as high as 50 meters. The blockage of GPS satellite is confirmed by observing the number of GPS satellite available. Ground truth at starting and ending position are measured by DGPS on a 2m poles. The drones carry a gimbaled camera on top of the GPS-Compass module. Although camera power cable is off, there could still have marginal compass bias from gimbals magnet. For bigger size drone or custom made drone, it is ideal to remove possible EM interference to both GPS and compass.

The compass reading is fused into the IMU. The compass adjusted IMU reading is then being fused into the EKF filter. Let w_{imu} denotes current IMU orientation reading in local NED coordinates and w_{lw} denotes the IMU orientation in local world representation. The local world orientation of the IMU can be found by

$$w_{local_world} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} w_{imu}^T \quad (5.34)$$

The local IMU coordinate is converted to compass referenced UTM frame coordinate by concatenating the transform $w_{wgs84} = R_{compass} w_{local_world}$. $R_{compass}$ is constructed from compass reading.

³REP 103 www.ros.org/reps/rep-0103.html

⁴REP 105 www.ros.org/reps/rep-0105.html

The initial estimate covariance P_0 are set to be $(1e-9)*I_{15 \times 15}$ where $I_{15 \times 15}$ represent identity matrix of 15X15 size. The process noise covariance Q are found to be

$$Q = \text{Diag}(\sigma_q) \quad (5.35)$$

where

$$\sigma_q = [0.05, 0.05, 0.06, 0.03, 0.03, 0.02, 0.025, 0.025, 0.04, 0.01, 0.01, 0.02, 0.01, 0.01, 0.015]$$

The process noise of v_σ has one to one correlation to input vector x . The value is found empirically after a few attempts. The result can be better when better process noise estimate can be obtained. The feedback configuration is shown as in Table 5.5 below

Category	x	y	z	ϕ	χ	φ	$\dot{x}$	$\dot{y}$	$\dot{z}$	$\dot{\phi}$	$\dot{\chi}$	$\dot{\varphi}$	$\ddot{x}$	$\ddot{y}$	$\ddot{z}$
Visual Inertial	*	*	*	*	*	*	1	1	1	1	1	1	0	0	0
LIDAR Inertial	*	*	*	*	*	*	1	1	1	1	1	1	0	0	0
CompassIMU	0	0	0	1	1	1	0	0	0	1	1	1	1	1	1

TABLE 5.5: The feedback state variable configuration. 1 denotes present, 0 denotes not present and * denotes differentiated and then force to zero parameters

The estimated GPS is generated from fixed known GPS points. The final target GPS points were measured multiple times and then the average number is taken. Because of the GPS multipath issue, the closer drone is to the building the more offset it will suffer. This offset affects both live run and posts GPS data collection. The testing sequence contains 2000-4000 image for each run. Multiple sequences contain the dynamic scene. The worst sequences contain both dynamic scene and very dark environment. The process cost minimal CPU usage with less than 10% of a CPU core and less than 1% of memory usage.

5.7 Fusion-Based State Estimation Qualitative Results Analysis

5.7.0.1 Qualitative Results With Normal Lighting Condition

Current view in image and map	Remark
<p>Run for 20 meters, reaching LIDAR featureless scene, RED-VISION, BLUE-LIDAR, Yellow-Fusion, GREEN-Ground truth starting and ending point</p>
<p>30 meters at featureless scene. LIDAR-based state estimator (blue) error accumulated</p>

<p>40 meters. The x plane distance falls short for LIDAR due to featureless scene. Visual falls short in next section.</p>
<p>60 meters, both LIDAR and Visual state estimate fall short. The fusion output is more closer to the goal</p>

<p>90 meters, minor disturbance cause estimator to drift left for half meter</p>
<p>over 100 meters, Fusion based solution is more close to the pole measured ground truth</p>

TABLE 5.6: Vision-based state estimator(red) vs LIDAR-based state estimator(blue) vs Fusion-based state estimator(Yellow). The image is taken from latitudes: 1.293502 longitude: 103.788308 at 1442 noon. Target equivalent GPS location latitudes 1.2934832484 longitudes 103.787625122. Only initial GPS was used in the calculation. Wind speed is less than 5m/s. Green X denotes the starting and ending location

FIGURE 5.40: Normal lighting condition variance plot. Variance in red means vision-based state estimation variance, Variance in blue means LIDAR -based state estimation variance. The first peak happens at the wall, the Second peak happens in the empty carpark. The variance is raw value without normalization

For normal cases, there were two peaks in variance as seen in Figure 5.40. The first peak is in the wall section where there are over 40 meters of empty area. This planner scene causes the drone LIDAR-based state estimator to drift. This can be observed when the robot travels more than 30 meters. The fused result and vision based result more agree to each other on that planer scene. However, when the robot is moving towards the north wall, a large trunk of the feature is missing causing the VO to be shorter than intended.

5.7.0.2 Qualitative Results With Tough Lighting Condition

Current view in image and map on tough lighting condition	Remark
<p>20 meters. Dark environment, featureless are for LIDAR state estimation. Vision fails from start due to low lighting condition. RED-VISION, BLUE-LIDAR, Yellow-Fusion, GREEN-Ground truth starting and ending point</p>
<p>Minimum car for LIDAR to track for Friday after work. Passing through empty park the LIDAR estimator error builds up. Vision error continuously climbing.</p>

<p>U turn near the featureless wall. For both LIDAR and vision it is featureless. LIDAR start to get corner feature reference after the turn.</p>
<p>Fast moving trolley appear in the scene. Both LIDAR and visual are affected</p>
<p>Stop close to building. Visual state estimator goes into the building. The LIDAR and Fusion are loser to the ground truth points.</p>

TABLE 5.7: Vision-based state estimation(red) vs LIDAR-based state estimation(blue) vs Fusion-based state estimation(Yellow). The image is taken from latitude:1.293502 longitude:103.788308 after rain and after 1800. Target equivalent GPS location latitude:1.2934832484 longitude: 103.787625122. Only initial GPS was used in the calculation. Wind speed is less than 5m/s. Green X denotes the starting and ending location

FIGURE 5.41: Tough lighting condition variance plot before normalization. Variance in red denotes visual-inertial state estimation variance, variance in blue means LIDAR state estimation variance

This sequence is a typical illustration of famous The Morpheus law in the aerospace industry. This law means that whatever can go wrong in testing will go wrong. Start of the sequence there is a car driving out. Though out the sequence, the image is too dark to get a robust feature. The dark image was caused by after cloudy rain condition as well as after 1800 work time. After that, there is no car in the car park. The LIDAR will not get enough feature for robust state estimates. Last but not least, there are dealer pushing trolley near the U-turn.

In the variance plot, the peaks in the LIDAR variance denotes the close by moving trolley. Though out the flight, the visual variance is constantly high. Take note that, this variance is not normalized for visualization purpose. This sequence contains the worst cases for all vision and LIDAR-based state estimation solutions.

For navigation use, only topology map is used. For visualization purpose, Google map is used. All the mapped the result is plotted by webtool⁵. All GPS error is calculated by web GPS distance finder ⁶

⁵Tool to draw: [HTTP://www.copypastemap.com/index.html](http://www.copypastemap.com/index.html)

⁶Tool to find distance: <https://gps-coordinates.org/distance-between-coordinates.php>

5.8 The Fusion Results Quantitative Discussion For Complete Set Of Sensors

The results can be found in Figures 5.43 and 5.42 ,5.7 5.6. The first case is demonstrated in the tough lighting conditions where visual inertial-based state estimation constantly fails and dual LIDAR Inertial-based state estimation is partially failing. The second case is visual inertial-based state estimation is good and dual LIDAR Inertial-based state estimation is partially failing. The other cases were recorded at similar locations without the dynamic lighting conditions.

FIGURE 5.42: Performance in the normal lighting condition. Both Visual Inertial-based state estimation and LIDAR Inertial-based state estimation is still acceptable. The fusion-based state estimation is more accurate. No valid GPS were captured during this sequence. Ground truth are measured after experiment

In both dynamic cases, the fusion-based state estimation reports more accurate results as can be seen from Table 5.8. For the LIDAR sequences captured from 1.293566, 103.787552, each run lasts 100 over meters. The GPS from drone drifts extensively. The starting and ending positions of drones are marked down physically. The GPS was then mounted on poles of 2 meters high.

FIGURE 5.43: When dealing with most tough case, proposed method still can generate accurate output where GPS is drift because of multipath from building and IMU drift

Each time, pole stationed vertically at the marked position for at least 30 min to find the average value of GPS. The quantitative difference between LIDAR based approach, Vision-based approach, and Fusion based approach are shown in the Table 5.8

Sequence No.	LIDAR error	Vision error	Fusion output
1 Normal	9.95 m	15.48m	5.72m
2 Dark dynamic(people)	9.61 m	51.27m	7.68m
3 Dynamic(car)	7.49 m	12.85	4.51m
4 Bright lighting	11.66 m	14.54	2.24m
5 Friday afternoon less car	5.86 m	10.14	4.09m

TABLE 5.8: Performance Comparisons Among LIDAR-based, Vision-based and Fusion-based Methods.

Overall fusion based method will ensure a much more accurate result. The fusion process can remove certain levels of the environment imperfection. And this is done without reference with landmarks. By discarding landmarks, the real-time performance can be ensured and computational cost is bounded. There are about 20 ROS bag sequence recorded in that area. On average, the LIDAR-based state estimation has an error between 5m-12m, Vision-based state estimation has a final error between 10m-50m which highly depends on lighting conditions. The fusion-based state estimation output can get as good as 2-8 meters.

Sequence No.	Vision-based	LIDAR-based	Fusion-based
Average error	9.1% - 45.5%	4.55% - 11.1%	1.81%-7.27%

TABLE 5.9: Accuracy Comparison Among LIDAR-based, Vision-based and Fusion-based Methods .

The average accuracy can be seen from Table 5.9. When dealing with normal outdoor lighting condition with vision-based state estimation, the outdoor best case scenario is similar to the worst-case scenario reported in indoor sequences[11]. The LIDAR-based state estimation has no direct external comparison. But it is somewhere between vision based and fusion based state estimation. The error of the LIDAR-based state estimation is usually bounded. The fusion based approaches can get the best results.

5.9 The Fusion Result Analysis For Partial Set Of Sensors

It is an interesting question that what if some sensors are not working. This section introduces the cases where a lesser amount of sensors are used for state estimation. This kind of case often happens when some sensor is broken down or intentionally sabotaged. It is crucial to analyze what will happen when part of the system is not working. Such analysis can be used to further improve the system robustness and accuracy. As seen in Figure 5.44, The more sensor can produce a more accurate

FIGURE 5.44: Fusion with different set of sensors in the normal lighting condition

result. Some subset configuration might work for some time, but there will always be errors in the output.

5.9.1 Performance In Normal Lighting Condition

The overall performance in normal lighting condition can be seen from Figure 5.44. There is some interesting observation with a certain combination of sensors.

5.9.1.1 IMU + Compass

FIGURE 5.45: IMU+Compass fusion in normal lighting condition. The drift pattern is similar to lost GPS type of drift

Figure 5.45 shows the Fusion-based state estimation with only IMU and compass on the topological map[286]. All other sensors are considered to be broken and do not produce data. This result is compared with vision-based state estimator and Dual LIDAR-based state estimator. It reassembles the similar GPS drifting pattern when the number of GPS counter is lower than four. This method is entirely up to the quality of IMU. For good IMU with five digits or six-digit cost, it could be accurate even after a long time. However, in this experiment only a three digit cost IMU is used. Therefore, this combination drifts heavily.

5.9.1.2 Vision+IMU+Compass

FIGURE 5.46: State estimation result from fusing Visual-Inertial and compass only (Black) on normal lighting condition

Figure 5.46 shows the fusion-based state estimation result with only vision, IMU and compass. The effect is similar to the vision+IMU combination. The overall performance is not bad. The observation is continuous. The continuous observation ensures smooth drone flight performance. If accuracy is not mandated, this method can do a fully autonomous navigation work. It is worth noting that the compass can sometimes help or interfere with the result. When there is an interference pattern at the 1st turn, the traveled distance falls short. Also, when clear of all moving cars and magnetic interference, the 2nd turn appears to be more accurate. By adding another component which is prone to environmental interference, the overall system accuracy can be lower. However, as long as the sensor did not disconnect, the robustness remains.

5.9.1.3 LIDAR + IMU + Compass

FIGURE 5.47: State estimation result from fusing LIDAR-Inertial and compass (Black) on normal lighting condition. The behavior is similar to animal disoriented movement pattern.

Figure 5.47 shows fusion based state estimation result with LIDAR, IMU, and compass. It is compared with vision-based state estimator and LIDAR-based state estimator. The overall performance is abysmal. When LIDAR on the drone faces the planer scene, LIDAR did not update its position, but IMU still tells the filter it is moving. In this case, the drone is confused. Biologically speaking, this is exactly what animals are evolved into. When animal rotates heavy for many rounds and comes to a sudden stop, the brain still predicts the movement based on previous states even with the current observation of a static scene. In this case, the animal usually runs in a hilarious manner similar to the Fusion-based state estimation in Figure 5.47. In this case, both semi-auto position hold and fully autonomous navigation are not recommended as it poses too much risk.

5.9.2 Performance In Tough Lighting Condition

The fusion-based state estimation result from other sensor combinations can be seen from Figure 5.48. The result is similar to the cases in the normal lighting

FIGURE 5.48: Fusion-based state estimation result from other sensor combinations on tough lighting condition

conditions.

5.9.2.1 IMU + compass

Figure 5.48(top left) shows fusion-based state estimation result with IMU and compass on a tough lighting condition. The result is compared with vision-based

state estimator(red) and LIDAR-based state estimator(purple). It also resembles the similar GPS drifting pattern as shown in Figure 5.45.

5.9.2.2 Vision + IMU + Compass

Figure 5.48 (top middle) shows fusion-based state estimation result with vision, IMU and compass. The result is compared with vision-based state estimator(red) and LIDAR-based state estimator(purple). The overall performance is awful. Because of the primary source of observation drift and corrupt heavily. The subsequent state obtained will be wrong and there is no way to correct it.

5.9.2.3 LIDAR + IMU + Compass

Figure 5.48 (top right) shows fusion-based state estimation result with LIDAR, IMU, and compass. The result is compared with vision-based state estimator(red) and LIDAR-based state estimator(purple). The overall performance is relatively bad when compared to the full sensor fusion method. The stop points are still far away from measured goal points. When drone faces the planer scene, LIDAR did not update position, but state estimation still tells the filter to go. In this case, the drone is confused. In this case, both autonomous position hold and fully autonomous navigation are not recommended as it poses too much risk.

5.10 The Fusion Result Analysis With Landmark Reference

By using all three methods together, better state estimation and mapping result can be achieved. In this experiment, the data is the same as the previous chapter. All depth sensor are fused according to the calibrated links. For depth part, the only middle x-y plane data is used for robustness. Based on the depth generated and pose from different reports of state estimation, several maps are drawn. The mapped result from different state estimation source is shown in Figures 5.50 5.51 and 5.52.

Please take note that, the author was walking next to the drone, so there might have obstacle mapped along the state estimator path. As shown in Figure 5.49,

FIGURE 5.49: An illustration of expected detected patterns over x-y plane. The expected obstacles(Green) are drawn by hand

there should be clear map separation between each car and buildings. The green color denotes the expected obstacle location. The small corner is representing a partially detected car. The extended corner or line is referring to building walls.

The red, purple and black are the Visual-Inertial-based state estimation, LIDAR-based state estimation, and Fusion-based state estimation output respectively.

FIGURE 5.50: Mapped scene with vision-based state estimator on a good day.

Normal lighting condition, Vision-based state estimator and Mapping

Figure 5.50 shows the mapped result from vision-based state estimator position feedback. The overall performance is not bad. However, as one can see, the last corner is supposed to align with the building. However, it is short in distance. Otherwise, based on this method, navigation is still possible. It is only the distance, and the target goal location needs to be adjusted. So overall speaking, just a minor accuracy problem and can be used for most of the tasks in the daytime.

Normal lighting condition, LIDAR-based state estimator and Mapping

Figure 5.51 shows the mapped result from LIDAR-based state estimator position feedback. In this scene, it can be observed that there are a large area of repeated mapped section along the way.

This error is caused by LIDAR-based state estimator lost when facing planner

FIGURE 5.51: Mapped scene with LIDAR-based state estimator on a good day.

scene. Initially, when the robot starts, the map created was correct. The red part shows minimum variance in mapping. However as the robot goes, it begins to encounter the classic planer scene which cause LIDAR to associate in ICP process falsely. With each wrong ICP reports, the depth is mapped at a wrong place. This error starts to accumulate until the end. When dealing with this case, there is a high chance no path will be found for fully autonomous navigation. The robot will be trapped inside the imaginary bounding boxes. In this case, the drone can only hover at the position as there is no way out in the global map. There is no way to plan a path out.

Normal lighting condition, Fusion-based state estimator and Mapping

Figure 5.52 shows the mapped result from Fusion-based state estimator position feedback. In this scene, it can be observed that the map is created with minimal variance. The north building corner is almost perfectly aligned on the map. Same can be seen in the north corner of the south building. Along the way to the target point, each car also shows very minimal variance. The obstacle along the path is caused by author walking nearby. The proposed multi-sensor fusion approach is so

FIGURE 5.52: Mapped scene with Fusion-based state estimator on a good day.

far the best-mapped case when compared with other solutions. Using this method, the drone is capable of do not only position hold but also perform complex fully autonomous navigation and obstacle avoidance.

5.11 Limitation and Future Works

Currently, this work is under the assumption that at least one sensor is working at a particular time. When both fails, the state estimation will drift because the state estimation only rely on IMU. There are cases that no LIDAR reference is presented and the whole scene is dynamic or featureless. Such case can be found in the open sea, or in tall grassland with a windy condition as shown in Figure 5.53. One possible solution to this kind of case is to fuse more sensor into the same network further. In this open sea and large grass land case, the GPS should have reasonable reading. The other method such as UWB will most likely work as well. By fusing additional information in to the proposed system, a more robust state estimation can be achieved.

FIGURE 5.53: Variance for both LIDAR-based state estimator and visual-based state estimator are high. The image sequence is taken with high wind speed, plenty of dynamic moving grass and no LIDAR feature around

5.12 Conclusion

In this chapter, a multi-sensor fusion based approach is proposed to deal with multiple sensor failure cases. It is done by finding the variance of each sensor feedbacks. Vision variance is found by its tracked feature intrinsic property. The LIDAR variance is tracked by finding the relation between the detected line and in-range point counts. The resulting performance beat any other methods in the research in terms of robustness and accuracy for lightweight solutions.

In this dissertation, a novel lightweight Visual LIDAR Inertial and Compass fusion-based method are proposed to address the robustness and accuracy factor in UAV navigation problem. Traditionally, drones deploy either vision based or LIDAR

based solution for navigational feedbacks. By visual-based state estimation or LIDAR-based state estimation, a drone can perform a certain degree of the autonomous flight. In reality, both methods fail from time to time. When lighting is not sufficient, vision-based solution fails. When dealing with long planner scenes or the noisy featureless environment, LIDAR-based solution fails. From the cost function for motion estimation point of view, there is no difference in error between failure and success case. When there is no difference in error residual, the estimated motion stuck at local minima and the estimated motion will be inaccurate. When multiple unreliable sources are used together, the estimated motion can be improved. With one sensor fails, the other may still work. Therefore, an EKF multi-sensor fusion based approach is proposed to solve this problem. The proposed EKF only requires final feedback from other source and does not require landmark features. This configuration can ensure a real-time performance and can run for a longer duration. The result is more accurate and robust when compared with any other single source lightweight odometry feedback.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This dissertation shows several new methods for autonomous lightweight UAV navigation.

The main contribution of this dissertation is listed as follows.

This dissertation shows robust stereo vision based navigation using enhanced stereo depth map post-processing technique from multiple domain analysis. The stereo method with real-time performance always has flaws. This section addresses the issue of how to process a depth image so that it can be used directly in autonomous navigation tasks. This algorithm uses a series of parallel image processing technique on both raw and depth image to detect the possible area of defects. Based on the detected field, the algorithm performs a clean up of the snowflakes to ensure proper output map for navigation packages. With the help of the semantic scene segmentation, special consideration is given to minimize large area false detection over the sky. In the end, A hole filling operation is done to get a continuous smooth depth map which is reasonable enough for path planning. The proposed method can do better processing than the previous benchmark. Also, live drone navigation using a single processing unit is demonstrated.

Then this dissertation presented a multi-sensor fusion based approach for robust and accurate state estimation. To use those sensors together, a calibration method is proposed to fuse the stereo camera and 2 LIDARs to form a uniform and consistent observations. Then based on the calibrated result, a fusion work can be

tested. The fusion works consist of three parts. The first part is the analysis of what's wrong with the current solution. Then based on the problem, the logical solution or variance measure can be proposed. Then based on each feedback and its variance, a global fusion can be achieved. So far this method shows the most robust and accurate performance on lightweight drones.

6.2 Future Work

The proposed works allow lightweight and robust UAV navigation system in the urban environment. It enables the drone company to automate part of daily routing with low risk bounded. However, there are still possibilities to explore in this field.

1. Fusion with more sensors

It can be seen from Chapter 5 that with each sensor integrated, the state estimation gets more robust. It is logical thinking to further integrate more sources at different sensing spectrum to achieve global robustness.

2. Use of a hierarchical model to improve the result. As shown in Chapter 5, there are a few attempts to improve the outcome by using a hierarchical graph model. Currently, it is still under investigation on what is the best way to constrain bottom layer wrong prediction with a less accurate and robust global frame.

3. Collaborative fusion

Right now, a single drone is limited by its weight, observation, and power. If there is a swarm of drones carrying advanced sensors, there is a real possibility that sensors across multiple machines can be fused. For example, Collaborative Visual SLAM has already been proposed for multi-drone navigation and mapping. It will not be long till the next multi-sensor over multi-machine fusion based method to appear.

4. Towards the end to end Fully learning based method

Learning based method is the dominant method many computer vision studies such as Image Segmentation, Classification, and detection. For multi-sensor fusion, there still have very little work. In other areas, the learning based approach shows 10-20 % of accuracy improvement for almost studies. Although there is a disadvantage that Learning based approach is slow in nature and requires substantial computational power, this method still represents the future for the computer vision and robotics communities.

List of Author's Awards, Patents, and Publications

Patents

- **S. Yuan, H. Wang, K. Wu, Mahdi Abolfazli Esfahani**, "Stereo Depth Map Post-Processing Using Multiple Domain Analysis for Autonomous Navigation" *NTU ref:2018-281, Singapore Provisional Patent No: 10201810715U.*

List of Publication

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "Depth-based Obstacle Avoidance through Deep Reinforcement Learning", ICMRE2019.

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "AbolDeepIO: A Novel Deep Inertial Odometry Network for Autonomus Vehicles", IEEE transactions on intelligent transportation systems, 2018

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "DeepDSAIR: Deep 6-DOF Camera Relocalization using Deblurred Semantic-Aware Image Representation for Large-Scale Outdoor Environments", IEEE Transactions on Automation Science and Engineering, 2018

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "From Local Understanding to Global Regression in Monocular Visual Odometry", International Journal of Pattern Recognition and Artificial Intelligence, 2018

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "A New Approach to Train Convolutional Neural Networks for Real-Time 6-DOF Camera

Relocalization", 2018 IEEE 14th International Conference on Control and Automation (ICCA)

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, " A New Approach to Train Convolutional Neural Networks for Monocular Visual Odometry", International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI), Montreal, Canada, 2018

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "TDPP-Net: Achieving Three-Dimensional Path Planning via a Deep Neural Network Architecture", Neurocomputing, 2018.

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "OTDPP-Net: Achieving Deep Neural Network based Real-Time 3D Path Planning in Completely Unknown Cluttered Environments", IEEE Trans. Cybernetics, 2018.

Keyu Wu, Mahdi Abolfazli Esfahani, Shenghai Yuan, Han Wang, "Learn to Steer through Deep Reinforcement Learning", Sensors, 2018.

H. Wang, S. Yuan and K. Wu, "Heterogeneous stereo: A human vision inspired method for general robotics sensing," TENCON 2017 - 2017 IEEE Region 10 Conference, Penang, 2017, pp. 793-798.

Han Wang, Wei Mou, Xiaozheng Mou, Shenghai Yuan, Soner Ulun, Shuai Yang, Bok-Suk Shin, An automatic self-calibration approach for wide baseline stereo cameras using sea surface images. ARSO 2015: 1-6

Han Wang, Xiaozheng Mou, Wei Mou, Shenghai Yuan, Soner Ulun, Shuai Yang, Bok-Suk Shin, Vision-based long range object detection and tracking for unmanned surface vehicle . RAM/CIS 2015: RAM:101-105

Shenghai Yuan, Han Wang Autonomous object level segmentation . ICARCV 2014: 33-37

Bibliography

- [1] Legal fly zone reference image. <https://flywhere.sg/>, 2019. xv, 3
- [2] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7):1073–1079, July 2009. ISSN 1051-8215. doi: 10.1109/TCSVT.2009.2020478. xv, xvi, 15, 16, 73, 84
- [3] Kalibr camera imu calibration tools. <https://github.com/ethz-asl/kalibr/wiki/camera-imu-calibration>, 2019. xv, 18
- [4] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna. LiDAR-Camera Calibration using 3D-3D Point correspondences. *ArXiv e-prints*, May 2017. xv, 21
- [5] Davison AJ. Real-time simultaneous localisation and mapping with a single camera in. In *In: Proceedings of International Conference on Computer Vision, 14031410.*, 2003. xv, 25, 50, 57
- [6] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *ISMAR, pp. 225234.*, 2007. xv, 11, 25, 26, 43, 56
- [7] Raul Mur-Artal and Juan D. Tardós. Fast relocalisation and loop closing in keyframe-based SLAM. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 846–853, 2014. doi: 10.1109/ICRA.2014.6906953. URL <https://doi.org/10.1109/ICRA.2014.6906953>. xv, 25, 26, 56
- [8] Richard A. Newcombe, Steven J. Lovegrove, and Andrew J. Davison. Dtam: Dense tracking and mapping in real-time. In *Proceedings of the 2011 International Conference on Computer Vision, ICCV '11*, pages 2320–2327,

- Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4577-1101-5. doi: 10.1109/ICCV.2011.6126513. URL <http://dx.doi.org/10.1109/ICCV.2011.6126513>. xv, 26, 45, 70
- [9] Jakob Engel and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *In ECCV*, 2014. xv, 24, 26, 42, 45, 56, 57, 70, 71, 79, 114, 159
- [10] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, May 2014. doi: 10.1109/ICRA.2014.6906584. xv, 23, 24, 26, 27, 29, 42, 43, 44, 50, 56, 79, 154, 156, 157, 159, 163, 165, 181, 198
- [11] Jeffrey Delmerico and Davide Scaramuzza. A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. 05 2018. doi: 10.1109/ICRA.2018.8460664. xv, 28, 163, 215
- [12] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, Aug 2018. ISSN 1552-3098. doi: 10.1109/TRO.2018.2853729. xv, xx, 18, 19, 22, 23, 24, 28, 29, 42, 44, 62, 114, 154, 156, 157, 159, 161, 162, 170, 171, 175, 178, 198
- [13] Ji Zhang and Sanjiv Singh. Laser-visual-inertial odometry and mapping with high robustness and low drift. 2018. doi: 10.1109/ICRA.2015.7139486. xv, 31, 32, 33
- [14] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011. xv, 34
- [15] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. pages 2043–2050, 05 2017. doi: 10.1109/ICRA.2017.7989236. xvi, 61
- [16] H. Wen A. Markham N R. Clark, S. Wang. Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem. In *AAAI*, 2017. xvi, 57, 62

- [17] C. J. Li and H. Ling. Synthetic aperture radar imaging using a small consumer drone. In *2015 IEEE International Symposium on Antennas and Propagation USNC/URSI National Radio Science Meeting*, pages 685–686, July 2015. doi: 10.1109/APS.2015.7304729. xvi, 67
- [18] Julius Schning and Gunther Heidemann. Taxonomy of 3d sensors - a survey of state-of-the-art consumer 3d-reconstruction sensors and their field of applications. 02 2016. doi: 10.5220/0005784801920197. xvi, 69
- [19] Helen Oleynikova, Zachary Taylor, Marius Fehr, Juan I. Nieto, and Roland Siegwart. Voxblox: Building 3d signed distance fields for planning. *CoRR*, abs/1611.03631, 2016. xvi, 72
- [20] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Auton. Robots*, 34(3):189–206, April 2013. ISSN 0929-5593. doi: 10.1007/s10514-012-9321-0. URL <http://dx.doi.org/10.1007/s10514-012-9321-0>. xvi, 72
- [21] Amir Atapour Abarghouei and Toby P. Breckon. Depthcomp: Real-time depth image completion based on prior semantic scene segmentation. In *BMVC*. BMVA Press, 2017. xvi, xvii, xviii, 74, 75, 80, 88, 89, 99, 112, 115, 116
- [22] uber crash. <https://www.mediapost.com/publications/article/316424/self-driving-vehicle-tests-paused-uber-fatal-cras.html>, 2018. xvi, 78, 79
- [23] Tesla crash. <https://intuitivefred888.blogspot.com/2016/07/decapitated-tesla-driver-was-watching.html>, 2018. xvi, 78, 79
- [24] Robert Spangenberg, Tobias Langner, and Raúl Rojas. Weighted semi-global matching and center-symmetric census transform for robust driver assistance. In *International Conference on Computer Analysis of Images and Patterns*, pages 34–41. Springer, 2013. xvi, 80
- [25] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, Apr 2002. ISSN 1573-1405. doi: 10.1023/A:

1014573219977. URL <https://doi.org/10.1023/A:1014573219977>. xvi, 73, 81, 85
- [26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. URL <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>. xvi, 58, 82
- [27] Haitao Niu, Xunjie Zhao, Li Chengjin, and Xiang Peng. Stereo matching based on sift descriptor with illumination and camera invariance. 7658, 05 2010. doi: 10.1117/12.865811. xvi, 73, 82
- [28] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17:65:1–65:32, 2016. xvi, xviii, 73, 74, 75, 80, 82, 83, 87, 112
- [29] K. Zhang, Y. Fang, D. Min, L. Sun, S. Yang, and S. Yan. Cross-scale cost aggregation for stereo matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(5):965–976, May 2017. ISSN 1051-8215. doi: 10.1109/TCSVT.2015.2513663. xvi, 73, 84, 85
- [30] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnnet for real-time semantic segmentation on high-resolution images. 04 2017. xvii, 99, 100
- [31] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. xviii, xix, 26, 79, 127, 134, 156
- [32] Joseph E. Hawkins. Human ear. <https://www.britannica.com/science/ear>, October 25, 2018. xix, 134, 156
- [33] Christopher K. I. Williams John Winn Andrew Zisserman Mark Everingham, Luc Van Gool. The pascal visual object classes (voc) challenge, September 2009, Vol 88: pp. 303-308. xix, 134, 156

- [34] Qilong Zhang and R. Pless. Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2301–2306 vol.3, Sep. 2004. doi: 10.1109/IROS.2004.1389752. xix, 20, 137, 138, 139, 144
- [35] F. Vasconcelos, J. P. Barreto, and U. Nunes. A minimal solution for the extrinsic calibration of a camera and a laser-rangefinder. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2097–2107, Nov 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.18. xix, 20, 137, 138
- [36] Lipu Zhou and Zhidong Deng. A new algorithm for the extrinsic calibration of a 2d lidar and a camera. *Measurement Science and Technology*, 25:065107, 05 2014. doi: 10.1088/0957-0233/25/6/065107. xix, 20, 137, 138
- [37] M. He, H. Zhao, J. Cui, and H. Zha. Calibration method for multiple 2d lidars system. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3034–3041, May 2014. doi: 10.1109/ICRA.2014.6907296. xix, 140
- [38] Khalil M. Ahmad Yousef, Bassam J. Mohd, Khalid Al-Widyan, and Thaier Hayajneh. Extrinsic calibration of camera and 2d laser sensors without overlap. *Sensors*, 17(10), 2017. ISSN 1424-8220. doi: 10.3390/s17102346. URL <http://www.mdpi.com/1424-8220/17/10/2346>. xix, 141, 150
- [39] Yoonsu Park, Seok Min Yun, Chee Sun Won, Kyungeun Cho, Kyhyun Um, and Sungdae Sim. Calibration between color camera and 3d lidar instruments with a polygonal planar board. In *Sensors*, 2014. xix, 142
- [40] W. Dong and V. Isler. A novel method for the extrinsic calibration of a 2d laser rangefinder and a camera. *IEEE Sensors Journal*, 18(10):4200–4211, May 2018. ISSN 1530-437X. doi: 10.1109/JSEN.2018.2819082. xix, 20, 142
- [41] Ankit Dhall, Kunal Chelani, Vishnu Radhakrishnan, and K. M. Krishna. Lidar-camera calibration using 3d-3d point correspondences. *CoRR*, abs/1705.09785, 2017. xix, 142
- [42] Wall image source. <http://www.textures4photoshop.com/tex/brick-and-wall/bricks-wall-wood-floor-room-interior-scene-texture.aspx>, 2018. xix, 143

- [43] List of tesla crash figure. https://www.google.com/search?biw=1465&bih=648&tbm=isch&sa=1&ei=2DxFXczqFaK1mgfzkqTwBA&q=google+tesla+autopilot+car+crash&oq=google+tesla+autopilot+car+crash&gs_l=img.3...7184.9091..9315...0.0..0.112.503.9j1.....0....1..gws-wiz-img.aKJhqSTvyxc&ved=0ahUKEwjMtrHdm-bjAhWimuYKHXMJCU4Q4dUDCAY&uact=5, 2018. xix, 154
- [44] Building inpection figure. https://www.google.com/search?biw=1465&bih=648&tbm=isch&sa=1&ei=4jxFXeiYK43Tz7sPybm08AU&q=drone+building+inpection&oq=drone+building+inpection&gs_l=img.3...353787.359168..359401...2.0..0.92.1088.26.....0...1..gws-wiz-img.....0..35i39j0j0i67j0i5i30j0i8i30j0i24j0i30.Xob7t8tzE7c&ved=0ahUKEwjokanim-bjAhWN6XMBHcmcA14Q4dUDCAY&uact=5, 2018. xix, 155
- [45] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34, 02 2014. doi: 10.1177/0278364914554813. xix, 28, 29, 62, 154, 156, 157, 161, 162, 168, 171
- [46] Fla introduction video. <https://www.youtube.com/watch?v=vDYy3L9nvLk>, 2018. xix, 157
- [47] Jakob Engel, Jrgen Sturm, and Daniel Cremers. Scale-aware navigation of a low-cost quadrocopter with a monocular camera. *Robotics and Autonomous Systems*, 62(11):1646 – 1656, 2014. ISSN 0921-8890. doi: <https://doi.org/10.1016/j.robot.2014.03.012>. URL <http://www.sciencedirect.com/science/article/pii/S0921889014000566>. Special Issue on Visual Control of Mobile Robots. xx, 27, 49, 71, 72, 157, 163
- [48] Ke Sun, Kartik Mohta, Bernd Pfrommer, Michael Watterson, Sikang Liu, Yash Mulgaonkar, Camillo Jose Taylor, and Vijay Kumar. Robust stereo visual inertial odometry for fast autonomous flight. *IEEE Robotics and Automation Letters*, PP, 11 2017. doi: 10.1109/LRA.2018.2793349. xx, 160, 161, 163, 164, 165
- [49] P. van Blyenburgh. Uav systems : Global review. In *In Conference, Amsterdam, The Netherlands*, 2006. 3

- [50] Valavanis K. Dalamagkidis, K. and L. A. Piegl. On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations. In *volume 54. Springer Science and Business Media.*, 2011. 3
- [51] G. Klein and D. Murray. Improving the agility of keyframe-based slam. In *ECCV, pp.802-815,*, 2008. 11, 26
- [52] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *ISMAR, pp. 83-86,*, 2009. 11, 26
- [53] Taih Pire, Thomas Fischer, Gastn Castro, Pablo DeCristforis, Javier Civera, and Julio JacoboBerlles. S-ptam. *Robot. Auton. Syst.*, 93(C):27–42, July 2017. ISSN 0921-8890. doi: 10.1016/j.robot.2017.03.019. URL <https://doi.org/10.1016/j.robot.2017.03.019>. 11, 26
- [54] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. Scale drift-aware large scale monocular slam. In *In Proceedings of Robotics: Science and Systems*, 2010. 12
- [55] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, November 2000. ISSN 0162-8828. doi: 10.1109/34.888718. URL <http://dx.doi.org/10.1109/34.888718>. 15, 16, 148
- [56] Xiaozheng Mou Shenghai Yuan Soner Ulun Shuai Yang Han Wang, Wei Mou and Bok-Suk Shin. An automatic self-calibration approach for wide baseline stereo cameras using sea surface images,. In *Unmanned Systems, 3(4), pp.277-290,*, 2015. 16, 35
- [57] Joern Rehder, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart. Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. pages 4304–4311, 05 2016. doi: 10.1109/ICRA.2016.7487628. 18
- [58] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1280–1286, Nov 2013. doi: 10.1109/IROS.2013.6696514. 18, 23, 24

- [59] Paul Timothy Furgale, Timothy D. Barfoot, and Gabe Sibley. Continuous-time batch estimation using temporal basis functions. In *ICRA*, pages 2088–2095. IEEE, 2012. ISBN 978-1-4673-1403-9. 18
- [60] J. Maye, P. Furgale, and R. Siegwart. Self-supervised calibration for robotic systems. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 473–480, June 2013. doi: 10.1109/IVS.2013.6629513. 18
- [61] L. Oth, P. Furgale, L. Kneip, and R. Siegwart. Rolling shutter camera calibration. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1360–1367, June 2013. doi: 10.1109/CVPR.2013.179. 18
- [62] Z. Yang, T. Liu, and S. Shen. Self-calibrating multi-camera visual-inertial fusion for autonomous mavs. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4984–4991, Oct 2016. doi: 10.1109/IROS.2016.7759732. 18, 23, 24
- [63] I. Miller, M. Campbell, and D. Huttenlocher. Efficient unbiased tracking of multiple dynamic obstacles under large viewpoint changes. *IEEE Transactions on Robotics*, 27(1):29–46, Feb 2011. ISSN 1552-3098. doi: 10.1109/TRO.2010.2085490. 19, 136
- [64] Anna Petrovskaya and Sebastian Thrun. Model based vehicle detection and tracking for autonomous urban driving. *Auton. Robots*, 26(2-3):123–139, April 2009. ISSN 0929-5593. doi: 10.1007/s10514-009-9115-1. URL <http://dx.doi.org/10.1007/s10514-009-9115-1>. 19, 136
- [65] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William “Red” Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous driving in urban environments: Boss and the urban challenge. *J. Field Robot.*, 25(8):425–466, August 2008. ISSN 1556-4959. doi: 10.1002/rob.v25:8. URL <http://dx.doi.org/10.1002/rob.v25:8>. 19, 136

- [66] T. Moore and D. Stouch. A generalized extended kalman filter implementation for the robot operating system. In *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer, July 2014. 19, 50, 136, 201
- [67] Juan Li, Xiang he, and Jia Li. 2d lidar and camera fusion in 3d modeling of indoor environment. 06 2015. doi: 10.1109/NAECON.2015.7443100. 19, 136
- [68] Mengxiao Chen, Shaowu Yang, Xiaodong Yi, and Dan Wu. Real-time 3d mapping using a 2d laser scanner and imu-aided visual slam. 07 2017. doi: 10.1109/RCAR.2017.8311877. 19, 136
- [69] Aravindhnan K Krishnan and Srikanth Saripalli. Cross-calibration of rgb and thermal cameras with a lidar for rgb-depth-thermal mapping. In *Unmanned Systems Vol. 05, No. 02, pp. 59-78,, 2017*. 20
- [70] Ruben Gomez, Jesus Briales, Eduardo Fernandez-Moral, and Javier Gonzalez-Jimnez. Extrinsic calibration of a 2d laser-range-finder and a camera based on scene corners. volume 2015, pages 3611–3616, 06 2015. doi: 10.1109/ICRA.2015.7139700. 20, 138
- [71] Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. In *published in the International Journal of Robotics Research*, 1986. 22, 23, 25, 49
- [72] D. Rye H. Durrant-Whyte and E. Nebot. Localisation of automatic guided vehicles. In *The 7th International Symposium on Robotics Research*, 1995. 22, 23, 25
- [73] Mingyang Li and Anastasios I. Mourikis. High-precision, consistent ekf-based visual-inertial odometry. *Int. J. Rob. Res.*, 32(6):690–711, May 2013. ISSN 0278-3649. doi: 10.1177/0278364913481251. URL <http://dx.doi.org/10.1177/0278364913481251>. 22, 28, 29, 49, 62, 163, 164, 171
- [74] Javier Civera, Dorian Galvez-Lopez, Luis Riazuelo, Juan Tardos, and J Montiel. Towards semantic slam using a monocular camera. pages 1277–1284, 09 2011. doi: 10.1109/IROS.2011.6048293. 22
- [75] M. Faessler, F. Fontana, C. Forster, and D. Scaramuzza. Automatic re-initialization and failure recovery for aggressive flight with a monocular

- vision-based quadrotor. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1722–1729, May 2015. doi: 10.1109/ICRA.2015.7139420. 23, 24, 171
- [76] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047270. URL <http://doi.acm.org/10.1145/2047196.2047270>. 24, 46
- [77] Ji Zhang and Sanjiv Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. volume 2015, 05 2015. doi: 10.1109/ICRA.2015.7139486. 24, 31, 32, 171
- [78] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34, 02 2014. doi: 10.1177/0278364914554813. 24
- [79] Burri M. Omari S. Hutter M. Bloesch, M. and R Siegwart. Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback. In *The International Journal of Robotics Research*, 36(10), 10531072., 2017. 24, 72, 157, 162, 171
- [80] J. Zhang and S. Singh. Laser-visual-inertial odometry and mapping with high robustness and low drift. In *Journal of Field Robotics*. Springer, July 2018. 24, 79, 114, 156
- [81] Molton ND Stasse O Davison AJ, Reid ID. Monoslam: real-time single camera slam. In *Pattern Anal Mach Intell IEEE Trans 29(6): 10521067*, 2007. 25, 50, 57
- [82] J. Hertzberg H. Surmann A. Nuchter, K. Lingemann. 6d slam for 3d mapping outdoor environments. In *Journal of Field Robotics*, 24 (8-9),, 2007. 25

- [83] S.I. Roumeliotis, G. Huang, A.I. Mourikis. A quadratic-complexity observability-constrained unscented kalman filter for slam. In *IEEE Trans. on Robotics*, 29(5): 1226-1243,, 2013. 25
- [84] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. Orb-slam: a versatile and accurate monocular slam system. *CoRR*, abs/1502.00956, 2015. URL <http://arxiv.org/abs/1502.00956>. 26, 56
- [85] Raul Mur-Artal and Juan D. Tardós. Probabilistic semi-dense mapping from highly accurate feature-based monocular slam. In *Robotics: Science and Systems XI, Sapienza University of Rome, Rome, Italy, July 13-17, 2015*, 2015. doi: 10.15607/RSS.2015.XI.041. URL <http://www.roboticsproceedings.org/rss11/p41.html>. 26, 56
- [86] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Trans. Robotics*, 31(5): 1147–1163, 2015. doi: 10.1109/TRO.2015.2463671. URL <https://doi.org/10.1109/TRO.2015.2463671>. 26, 56
- [87] Raul Mur-Artal and Juan D. Tardós. Orb-slam2: an open-source slam system for monocular, stereo and RGB-D cameras. *CoRR*, abs/1610.06475, 2016. URL <http://arxiv.org/abs/1610.06475>. 26, 56
- [88] Raul Mur-Artal and Juan D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017. doi: 10.1109/TRO.2017.2705103. URL <https://doi.org/10.1109/TRO.2017.2705103>. 26, 43, 56, 154, 156, 159
- [89] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *CVPR*, pages 343–352. IEEE Computer Society, 2015. 26
- [90] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *2013 IEEE International Conference on Computer Vision*, pages 1449–1456, Dec 2013. doi: 10.1109/ICCV.2013.183. 26, 42, 43, 44, 45, 56, 57, 79, 114, 165
- [91] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(3):611–625, 2018. 27

- [92] Rui Wang, Martin Schwörer, and Daniel Cremers. Stereo DSO: large-scale direct sparse visual odometry with stereo cameras. *CoRR*, abs/1708.07878, 2017. 27, 159
- [93] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *Trans. Rob.*, 33(2):249–265, April 2017. ISSN 1552-3098. doi: 10.1109/TRO.2016.2623335. URL <https://doi.org/10.1109/TRO.2016.2623335>. 27, 42
- [94] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI’81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. URL <http://dl.acm.org/citation.cfm?id=1623264.1623280>. 27, 44, 171
- [95] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart. A robust and modular multi-sensor fusion approach applied to mav navigation. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3923–3929, Nov 2013. doi: 10.1109/IROS.2013.6696917. 27, 49, 157, 163, 198
- [96] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct ekf-based approach. ETH-Zrich, 2015. doi: 10.3929/ethz-a-010566547. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); Conference Location: Hamburg, Germany; Conference Date: September 28 - October 2, 2015. 28, 29, 62, 161, 162, 171
- [97] Simon Lynen, Markus W. Achtelik, Stephan Weiss, Margarita Chli, and Roland Y. Siegwart. A robust and modular multi-sensor fusion approach applied to mav navigation. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) : 3-7 November 2013, Tokyo, Japan*, pages 3923 – 3929. IEEE, 2013. doi: 10.3929/ethz-a-010023029. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013); Conference Location: Tokyo, Japan; Conference Date: November 3-7, 2013. 29, 62

- [98] Frank Dellaert. Overview in this document i provide a hands-on introduction to both factor graphs and gtsam., 2012. 29, 50, 62, 157
- [99] Ji Zhang and Sanjiv Singh. LOAM: lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, 2014. 31, 32
- [100] B(erkeley) l(ocalization) a(nd) m(apping)- blam. <https://github.com/erik-nelson/blam>, 2019. 31
- [101] Rekha Patil Seong Sill Park G. Ajay Kumar, Ashok Kumar Patil and Young Ho Chai. *A LiDAR and IMU Integrated Indoor Navigation System for UAVs and Its Application in Real-Time Pipeline Classification*. Sensors (Basel). Jun; 17(6): 1268., 2017. 33, 156
- [102] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf. A flexible and scalable slam system with full 3d motion estimation. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 155–160, Nov 2011. doi: 10.1109/SSRR.2011.6106777. 33
- [103] David Roca, Joaquin Martnez-Snchez, Susana Lagela, and Pedro Arias. Novel aerial 3d mapping system based on uav platforms and 2d laser scanners. *Journal of Sensors*, 2016:1–8, 01 2016. doi: 10.1155/2016/4158370. 33
- [104] * S. Lagelaa L. Daz-Vilariaoa D. Roca a, J. Armestoa. *LIDAR-EQUIPPED UAV FOR BUILDING INFORMATION MODELLING*. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-5, 2014 ISPRS Technical Commission V Symposium, 23–25 June 2014, Riva del Garda, Italy, 2014. 33
- [105] Mark Maimone, Yang Cheng, and Larry Matthies. Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186. doi: 10.1002/rob.20184. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20184>. 35
- [106] Xiaozheng Mou and Han Wang. Wide-baseline stereo based obstacle mapping for unmanned surface vehicles. In *Sensors*, 2018. 35
- [107] Xiaozheng Mou and Han Wang. Image-based maritime obstacle detection using global sparsity potentials. In *Journal of Information and Communication Convergence Engineering*, 14(2), pp.129-135, 2016. 35

- [108] Xiaozheng Mou and Han Wang. Global sparsity potentials for obstacle detection from unmanned surface vehicles. In *in Proceedings of IEEE International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pp.1-6,, 2015. 35
- [109] Wei Mou Shenghai Yuan Soner Ulun-Shuai Yang Han Wang, Xiaozheng Mou and Bok-Suk Shin. Vision based long range object detection and tracking for unmanned surface vehicle. In *in Proceedings of IEEE International Conference on CIS-RAM*), pp.101-105,, 2015. 35
- [110] Wei Mou Shenghai Yuan Soner Ulun-Shuai Yang Han Wang, Xiaozheng Mou and Bok-Suk Shin. An automatic self-calibration approach for wide baseline stereo cameras using sea surface images. In *in Proceedings of IEEE International Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pp.1-6,, 2015. 35
- [111] D. Nister. An efficient solution to the five-point relative pose problem. 38
- [112] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003. ISBN 0521540518. 38, 39, 55, 171
- [113] Bernd Manfred Kitt, Joern Rehder, Andrew D Chambers, Miriam Schonbein, Henning Lategahn, and Sanjiv Singh. Monocular visual odometry using a planar road model to solve scale ambiguity. 39
- [114] J. Grter, T. Schwarze, and M. Lauer. Robust scale estimation for monocular visual odometry using structure from motion and vanishing points. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 475–480, June 2015. doi: 10.1109/IVS.2015.7225730. 39
- [115] Wu W J. Basic principles of mechanical theorem proving in elementary geometries. 40
- [116] David Nistr, Oleg Naroditsky, and James Bergen. Visual odometry. pages 652–659, 2004. 42
- [117] Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV)*, 2011. 44

- [118] Bernd Kitt, Andreas Geiger, and Henning Lategahn. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In *Intelligent Vehicles Symposium (IV)*, 2010. 44, 171
- [119] J. M. M. Montiel, Javier Civera, and Andrew J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Robotics: Science and Systems*. The MIT Press, 2006. 44, 45, 70
- [120] Javier Civera, Andrew J. Davison, and J. M. M. Montiel. Inverse depth parametrization for monocular SLAM. *IEEE Trans. Robotics*, 24(5):932–945, 2008. 44, 45, 71
- [121] Javier Civera, Andrew J. Davison, and J. M. M. Montiel. Inverse depth to depth conversion for monocular SLAM. In *ICRA*, pages 2778–2783. IEEE, 2007. 44, 45, 71
- [122] C. Wang, T. Wang, J. Liang, Y. Chen, Y. Zhang, and C. Wang. Monocular visual slam for small uavs in gps-denied environments. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 896–901, Dec 2012. doi: 10.1109/ROBIO.2012.6491082. 44, 45, 71
- [123] Daniel Magree, John G. Mooney, and Eric N. Johnson. Monocular visual mapping for obstacle avoidance on uavs. *Journal of Intelligent & Robotic Systems*, 74(1):17–26, Apr 2014. ISSN 1573-0409. doi: 10.1007/s10846-013-9967-7. URL <https://doi.org/10.1007/s10846-013-9967-7>. 44, 45, 71
- [124] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical report, International Journal of Computer Vision, 1991. 44
- [125] Jianbo Shi and Tomasi. Good features to track. In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994. doi: 10.1109/CVPR.1994.323794. 44, 171
- [126] Stan Birchfield. Derivation of kanade-lucas-tomasi tracking equation. 09 1999. 44

- [127] Shuhui Bu, Yong Zhao, Gang Wan, Ke Li, Gong Cheng, and Zhenbao Liu. Semi-direct tracking and mapping with rgb-d camera for mav. *Multimedia Tools and Applications*, 76(3):4445–4469, Feb 2017. ISSN 1573-7721. doi: 10.1007/s11042-016-3524-x. URL <https://doi.org/10.1007/s11042-016-3524-x>. 44
- [128] Wenju Xu, Dongkyu Choi, and Guanghui Wang. Direct visual-inertial odometry with semi-dense mapping. *Computers and Electrical Engineering*, 67, 02 2018. doi: 10.1016/j.compeleceng.2018.02.002. 46
- [129] W. Liu, G. Loianno, K. Mohta, K. Daniilidis, and V. Kumar. Semi-dense visual-inertial odometry and mapping for quadrotors with swap constraints. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–6, May 2018. doi: 10.1109/ICRA.2018.8463163. 46
- [130] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger. Elasticfusion. *Int. J. Rob. Res.*, 35(14):1697–1716, December 2016. ISSN 0278-3649. doi: 10.1177/0278364916669237. URL <https://doi.org/10.1177/0278364916669237>. 46
- [131] Thomas Whelan, Michael Kaess, Hordur Johannsson, Maurice Fallon, John J. Leonard, and John McDonald. Real-time large-scale dense rgb-d slam with volumetric fusion. *Int. J. Rob. Res.*, 34(4-5):598–626, April 2015. ISSN 0278-3649. doi: 10.1177/0278364914551008. URL <http://dx.doi.org/10.1177/0278364914551008>. 46
- [132] Thomas Whelan, Michael Kaess, Ross Finman, Maurice F. Fallon, Hordur Johannsson, John J. Leonard, and John McDonald. 3D mapping, localisation and object retrieval using low cost robotic platforms: A robotic search engine for the real-world. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, July 2014. 46
- [133] K Whelan, Michael Kaess, Maurice F. Fallon, Hordur Johannsson, John J. Leonard, and John McDonald. Kintinuous : Spatially extended kinectfusion. 2012. 46

- [134] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald. Robust real-time visual odometry for dense rgb-d mapping. In *2013 IEEE International Conference on Robotics and Automation*, pages 5724–5731, May 2013. doi: 10.1109/ICRA.2013.6631400. 46
- [135] Chen Wang, Minh-Chung Hoang, L. Xie, and Junsong Yuan. Non-iterative rgb-d-inertial odometry. 2017. 47
- [136] R. Caseiro J. F. Henriques and P. Martins. High-speed tracking with kernelized correlation filters. 47
- [137] W. Chen, H. Fu, M. Shi, and Y. Chen. A dense optical flow-based feature matching approach in visual odometry. In *2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, volume 2, pages 343–348, Aug 2017. doi: 10.1109/IHMSC.2017.189. 47
- [138] O. G. Grasa, J. Civera, and J. M. M. Montiel. Ekf monocular slam with relocalization for laparoscopic sequences. In *2011 IEEE International Conference on Robotics and Automation*, pages 4816–4821, May 2011. doi: 10.1109/ICRA.2011.5980059. 49
- [139] B. Kitt, A. Geiger, and H. Lategahn. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In *2010 IEEE Intelligent Vehicles Symposium*, pages 486–492, June 2010. doi: 10.1109/IVS.2010.5548123. 49, 50
- [140] Guivant J. Efficient simultaneous localization and mapping in large environments. In *Dissertation, University of Sydney, Australia*, 2002. 49
- [141] Koller D Ghahramani Z et al Thrun S. Simultaneous mapping and localization with sparse extended information filters: theory and initial results. In *Technical Report CMU-CS-02-11, Carnegie Mellon*, 2002. 49
- [142] L. M. Paz, P. Pinis, J. D. Tards, and J. Neira. Large-scale 6-dof slam with stereo-in-hand. *IEEE Transactions on Robotics*, 24(5):946–957, Oct 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2004637. 49
- [143] P. Pinis and J. D. Tards. Large-scale slam building conditionally independent local maps: Application to monocular vision. *IEEE Transactions on Robotics*, 24(5):1094–1106, Oct 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2004636. 49

- [144] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Eighteenth National Conference on Artificial Intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence. ISBN 0-262-51129-0. URL <http://dl.acm.org/citation.cfm?id=777092.777184>. 49
- [145] Michael Steven Montemerlo. *Fastslam: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. PhD thesis, Pittsburgh, PA, USA, 2003. AAI3102462. 49
- [146] A. P. Gee, D. Chekhlov, A. Calway, and W. Mayol-Cuevas. Discovering higher level structure in visual slam. *IEEE Transactions on Robotics*, 24(5): 980–990, Oct 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2004641. 50
- [147] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968, June 2011. doi: 10.1109/IVS.2011.5940405. 50, 56
- [148] Matthias Faessler, Flavio Fontana, Christian Forster, Elias Mueggler, Matia Pizzoli, and Davide Scaramuzza. Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *J. Field Robotics*, 33:431–450, 2016. 50
- [149] S Lynen, M Achtelik, S Weiss, M Chli, and R Siegwart. A robust and modular multi-sensor fusion approach applied to mav navigation. In *Proc. of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2013. 50, 201
- [150] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613, May 2011. doi: 10.1109/ICRA.2011.5979949. 55
- [151] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. <http://ceres-solver.org>. 55, 178
- [152] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Monocular vision based slam for mobile robots. In *18th International Conference on*

- Pattern Recognition (ICPR'06)*, volume 3, pages 1027–1031, Aug 2006. doi: 10.1109/ICPR.2006.810. 56
- [153] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Real-time monocular slam: Why filter? In *2010 IEEE International Conference on Robotics and Automation*, pages 2657–2664, May 2010. doi: 10.1109/ROBOT.2010.5509636. 56
- [154] David Ball, Scott Heath, Janet Wiles, Gordon Wyeth, Peter Corke, and Michael Milford. Openratslam: An open source brain-based slam system. *Auton. Robots*, 34(3):149–176, April 2013. ISSN 0929-5593. doi: 10.1007/s10514-012-9317-9. URL <http://dx.doi.org/10.1007/s10514-012-9317-9>. 56
- [155] Heping Li Yihong Wu, Fulin Tang. Image based camera localization: an overview. In *Invited Paper by Visual Computing for Industry, Biomedicine and Art*, 2018. 57
- [156] I. Laina N. Navab K. Tateno, F. Tombari. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *CVPR*, 2017. 57
- [157] J. Uhrig N. Mayer E. Ilg-A. Dosovitskiy T. Brox B. Ummenhofer, H. Zhou. C demon: Depth and motion network for learning monocular stereo. In *CVPR*, 2017. 57
- [158] C. Schmid BS. Vijayanarasimhan, S. Ricco. Sfm-net: Learning of structure and motion from video. In *arXiv:1704.07804*, 2017. 57
- [159] N. Snavely D.G. Lowe T. Zhou, M. Brown. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 57
- [160] Z. Long D. Gu. R. Li, S. Wang. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *arXiv: 1709.06841*, 2017. 57, 62
- [161] A. Rabinovich D. Detone, T. Malisiewicz. Toward geometric deep slam. In *arXiv:1707.07410*, 2017. 57
- [162] X. Gao and T. Zhang. Unsupervised learning to detect loops using deepneural networks for visual slam system. In *Autonomous Robotics*, 41:1-18, 2017. 57

- [163] H. Araujo E. Konukoglu M. Sitti M. Turan, Y. Almaliloglu. Deep endovo: A recurrent convolutional neural network based visual odometry approach for endoscopic capsule robots. In *Neurocomputing*, 275: 1861-1870, 2018. 57
- [164] B. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. In *Robotics and Autonomous Systems*, 1991. 57
- [165] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *ICRA*, Vol. 2, pp. 1023-1029, 2000. 57
- [166] P. Beeson M. MacMahon F. Savelli B. Kuipers, J. Modayil. Local metrical and global topological maps in the hybrid spatial semantic hierarchy. In *ICRA*, pp. 4845-4851,, 2004. 57
- [167] O. Hasegawa A. Kawewong, N. Tongprasit. Pirf-nav 2.0: Fast and online incremental appearance-based loop-closure detection in an indoor environment. In *Robotics and Autonomous Systems*, 59(10): 727-739, 2011. 57
- [168] J. Gonzalez-Jimenez J.L. Blanco, J.A. Fernandez-Madrigal. Towards a unified bayesian approach to hybrid metric-topological slam. In *IEEE Transactions on Robotics*, 24(2): 259-270, 2008. 57
- [169] N. Snderhauf and P. Protzel. Switchable constraints for robust pose graph slam. In *IROS*, pp. 1879-1884,, 2012. 57
- [170] J. Neira Y. Latif, C. Cadena. Robust loop closing over time for pose graph slam. In *International Journal of Robotics Research*, 32(14): 1611-1626,, 2013. 57
- [171] Y. Le Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. Advances in neural information processing systems 2. chapter Handwritten Digit Recognition with a Back-propagation Network, pages 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990. ISBN 1-55860-100-7. URL <http://dl.acm.org/citation.cfm?id=109230.109279>. 59
- [172] Paul Viola and Michael J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, May 2004. ISSN 0920-5691. doi:

- 10.1023/B:VISI.0000013087.49260.fb. URL <https://doi.org/10.1023/B:VISI.0000013087.49260.fb>. 59
- [173] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J. Lang. Readings in speech recognition. chapter Phoneme Recognition Using Time-delay Neural Networks, pages 393–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990. ISBN 1-55860-124-4. URL <http://dl.acm.org/citation.cfm?id=108235.108263>. 60
- [174] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Robert Fergus. Deconvolutional networks. 60
- [175] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *Proceedings of the 20th International Conference on Artificial Neural Networks: Part III*, ICANN’10, pages 92–101, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-15824-2, 978-3-642-15824-7. URL <http://dl.acm.org/citation.cfm?id=1886436.1886447>. 60
- [176] Kishore Konda. Learning visual odometry with a convolutional network. 61, 62
- [177] Evan Zaugg, D.L. Hudson, and D.G. Long. The byu mu sar: A small, student-built sar for uav operation. pages 411 – 414, 09 2006. doi: 10.1109/IGARSS.2006.110. 66
- [178] Mario Greco, G Pinelli, Krzysztof Kulpa, Piotr Samczynski, B Query, and Stephane Query. *The study on SAR images exploitation for air platform navigation purposes*, pages 347–352. 01 2011. ISBN 978-1-4577-0138-2. 66
- [179] Chiaradia MT Greco M Pinelli G Nitti DO, Bovenga F. Feasibility of using synthetic aperture radar to aid uav navigation. In *Sensors (Basel). 2015 Jul 28;15(8):18334-59*. doi: 10.3390/s150818334., 2015. 66
- [180] R. K Luneburg. Mathematical theory of optics. In *Providence, Rhode Island: Brown University. pp. 189213..*, 1944. 67
- [181] David Davies, Robert Bolam, Yuriy Vagapov, and Peter Excell. Ultrasonic sensor for uav flight navigation. pages 1–7, 01 2018. doi: 10.1109/IWED.2018.8321389. 67

- [182] Gerard Gibbs, Huamin Jia, and Irfan Madani. Obstacle detection with ultrasonic sensors and signal analysis metrics. *Transportation Research Procedia*, 28:173 – 182, 2017. ISSN 2352-1465. doi: <https://doi.org/10.1016/j.trpro.2017.12.183>. URL <http://www.sciencedirect.com/science/article/pii/S2352146517311031>. INAIR 2017. 67
- [183] A. Censi. An icp variant using a point-to-line metric. In *2008 IEEE International Conference on Robotics and Automation*, pages 19–25, May 2008. doi: 10.1109/ROBOT.2008.4543181. 68
- [184] Giulio Sandini and Massimo Tistarelli. Active tracking strategy for monocular depth inference over multiple frames. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1):13–27, 1990. 70
- [185] Hanqi Zhuang, Raghavan Sudhakar, and Jen-yu Shieh. Depth estimation from a sequence of monocular images with known camera motion. *Robotics and Autonomous Systems*, 13(2):87–95, 1994. 70
- [186] Yuji Fujii, David K. Wehe, and Terry E. Weymouth. Robust monocular depth perception using feature pairs and approximate motion. In *ICRA*, pages 33–39. IEEE, 1992. 70
- [187] Ashutosh Saxena, Sung H. Chung, and Andrew Y. Ng. Learning depth from single monocular images. In *NIPS*, pages 1161–1168, 2005. 70
- [188] Andreas Wedel, Uwe Franke, Jens Klappstein, Thomas Brox, and Daniel Cremers. Realtime depth estimation and obstacle detection from monocular video. In *DAGM-Symposium*, volume 4174 of *Lecture Notes in Computer Science*, pages 475–484. Springer, 2006. 70
- [189] Evren Imre, Marie-Odile Berger, and Nicolas Noury. Improved inverse-depth parameterization for monocular simultaneous localization and mapping. In *ICRA*, pages 381–386. IEEE, 2009. 71
- [190] Ashutosh Saxena, Jamie Schulte, and Andrew Y. Ng. Depth estimation using monocular and stereo cues. In *IJCAI*, pages 2197–2203, 2007. 71
- [191] Tien-Ying Kuo and Yi-Chung Lo. Depth estimation from a monocular view of the outdoors. *IEEE Trans. Consumer Electronics*, 57(2):817–822, 2011. 71

- [192] Gunther Schneider, Bernard Heit, Johannes Hönig, and Jacques Brémont. Monocular depth perception by evaluation of the blur in defocused images. In *ICIP (2)*, pages 116–119. IEEE Computer Society, 1994. 71
- [193] Jean-Vincent Leroy, Thierry Simon, and François Deschênes. Real time monocular depth from defocus. In *ICISP*, volume 5099 of *Lecture Notes in Computer Science*, pages 103–111. Springer, 2008. 71
- [194] Tiago Gaspar and Paulo Jorge Ramalho Oliveira. Monocular depth from focus estimation with complementary filters. In *ICRA*, pages 4986–4991. IEEE, 2011. 71
- [195] Tomoyuki Shiozaki and Gamini Dissanayake. Eliminating scale drift in monocular SLAM using depth from defocus. *IEEE Robotics and Automation Letters*, 3(1):581–587, 2018. 71
- [196] Tomoyuki Shiozaki and Gamini Dissanayake. Monocular 3d metric scale reconstruction using depth from defocus and image velocity. In *IROS*, pages 6723–6728. IEEE, 2017. 71
- [197] Che-Chun Su, Lawrence K. Cormack, and Alan C. Bovik. Depth estimation from monocular color images using natural scene statistics models. In *IVMSP*, pages 1–4. IEEE, 2013. 71
- [198] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *CoRR*, abs/1502.07411, 2015. 71
- [199] Miaomiao Liu, Mathieu Salzmann, and Xuming He. Structured depth prediction in challenging monocular video sequences. *CoRR*, abs/1511.06070, 2015. 71
- [200] Hang Xu, Kan Li, FuYu Lv, and JianMeng Pei. 3d depth perception from single monocular images. In *MMM (1)*, volume 8935 of *Lecture Notes in Computer Science*, pages 510–521. Springer, 2015. 71
- [201] Ziyu Zhang, Alexander G. Schwing, Sanja Fidler, and Raquel Urtasun. Monocular object instance segmentation and depth ordering with cnns. In *ICCV*, pages 2614–2622. IEEE Computer Society, 2015. 71

- [202] Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *CVPR*, pages 1119–1127. IEEE Computer Society, 2015. 71
- [203] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. *CoRR*, abs/1612.02401, 2016. 71
- [204] Michele Mancini, Gabriele Costante, Paolo Valigi, and Thomas A. Ciarfuglia. Fast robust monocular depth estimation for obstacle detection with fully convolutional networks. *CoRR*, abs/1607.06349, 2016. 71
- [205] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *CoRR*, abs/1609.03677, 2016. 71
- [206] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks. *CoRR*, abs/1605.02305, 2016. 71
- [207] Yuanzhouhan Cao, Chunhua Shen, and Heng Tao Shen. Exploiting depth from single monocular images for object detection and semantic segmentation. *CoRR*, abs/1610.01706, 2016. 71
- [208] Babak Rezaeirowshan, Coloma Ballester, and Gloria Haro. Monocular depth ordering using perceptual occlusion cues. In *VISIGRAPP (4: VISAPP)*, pages 431–441. SciTePress, 2016. 71
- [209] Michele Mancini, Gabriele Costante, Paolo Valigi, and Thomas A. Ciarfuglia. Fast robust monocular depth estimation for obstacle detection with fully convolutional networks. In *IROS*, pages 4296–4303. IEEE, 2016. 71
- [210] William A. P. Smith, Ravi Ramamoorthi, and Silvia Tozza. Linear depth estimation from an uncalibrated, monocular polarisation image. In *ECCV (8)*, volume 9912 of *Lecture Notes in Computer Science*, pages 109–125. Springer, 2016. 71
- [211] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *CVPR*, pages 4058–4066. IEEE Computer Society, 2016. 71

- [212] Anirban Roy and Sinisa Todorovic. Monocular depth estimation using neural regression forest. In *CVPR*, pages 5506–5514. IEEE Computer Society, 2016. 71
- [213] Tianteng Bi, Yue Liu, Dongdong Weng, and Yongtian Wang. Monocular depth estimation of outdoor scenes using RGB-D datasets. In *ACCV Workshops (2)*, volume 10117 of *Lecture Notes in Computer Science*, pages 88–99. Springer, 2016. 71
- [214] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(10):2024–2039, 2016. 71
- [215] Xiaoshui Huang, Lixin Fan, Jian Zhang, Qiang Wu, and Chun Yuan. Real time complete dense depth reconstruction for a monocular camera. In *CVPR Workshops*, pages 674–679. IEEE Computer Society, 2016. 71
- [216] M. Pizzoli, C. Forster, and D. Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2609–2616, May 2014. doi: 10.1109/ICRA.2014.6907233. 71
- [217] Jakob Engel, Jrgen Sturm, and Daniel Cremers. Accurate figure flying with a quadrocopter using onboard visual and inertial sensing. *IMU*, 320, 01 2012. 71, 72
- [218] J. Engel, J. Sturm, and D. Cremers. Camera-based navigation of a low-cost quadrocopter. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2815–2821, Oct 2012. doi: 10.1109/IROS.2012.6385458. 71, 72
- [219] Q. Zeng, Y. Wang, J. Liu, R. Chen, and X. Deng. Integrating monocular vision and laser point for indoor uav slam. In *2014 Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS)*, pages 170–179, Nov 2014. doi: 10.1109/UPINLBS.2014.7033725. 71
- [220] Abdulla Al-Kaff, Fernando Garca, David Martn, Arturo De La Escalera, and Jos Armingol. Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for uavs. *Sensors*, 17(5):1061, May

2017. ISSN 1424-8220. doi: 10.3390/s17051061. URL <http://dx.doi.org/10.3390/s17051061>. 71
- [221] Yi Lin, Fei Gao, Tong Qin, Wenliang Gao, Tianbo Liu, William Wu, Zhenfei Yang, and Shaojie Shen. Autonomous aerial navigation using monocular visual-inertial fusion. *J. Field Robotics*, 35(1):23–51, 2018. doi: 10.1002/rob.21732. URL <https://doi.org/10.1002/rob.21732>. 72
- [222] Stanley T. Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision*, 35:269–293, 1998. 73, 74
- [223] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 15–18, Aug 2006. doi: 10.1109/ICPR.2006.1033. 73, 74
- [224] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(5):815–830, May 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.77. URL <https://doi.org/10.1109/TPAMI.2009.77>. 73, 82
- [225] Nengjun Wang, Bin bin Peng, Daqi Zhu, and Pengfei Xu. Stereo matching for binocular underwater images using surf operator and epipolar restriction. In Shiwei Ma, Li Jia, Xin Li, Ling Wang, Huiyu Zhou, and Xin Sun, editors, *Life System Modeling and Simulation*, pages 378–386, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-45283-7. 73
- [226] Christian Küblbeck, Roland Ach, and Andreas Ernst. Stereo image based collision prevention using the census transform and the snow classifier. In *ICINCO*, pages 447–454. INSTICC Press, 2005. 73
- [227] Mario Alberto Ibarra-Manzano, Dora Luz Almanza-Ojeda, Michel Devy, Jean-Louis Boizard, and Jean-Yves Fourniols. Stereo vision algorithm implementation in FPGA using census transform for effective resource optimization. In *DSD*, pages 799–805. IEEE Computer Society, 2009. 73
- [228] Jae-Chang Kwak, Tae Ryong Park, Yong-Seo Koo, and Kwang-Yeob Lee. Implementation of improved census transform stereo matching on a multicore

- processor. In *MUE*, volume 240 of *Lecture Notes in Electrical Engineering*, pages 989–995. Springer, 2013. 73
- [229] Li Ma, Jingjiao Li, Ji Ma, and Hanyue Zhang. A modified census transform based on the neighborhood information for stereo matching algorithm. In *ICIG*, pages 533–538. IEEE Computer Society, 2013. 73
- [230] Wade S. Fife and James K. Archibald. Improved census transforms for resource-optimized stereo vision. *IEEE Trans. Circuits Syst. Video Techn.*, 23(1):60–73, 2013. 73
- [231] Stefania Perri, Pasquale Corsonello, and Giuseppe Cocorullo. Adaptive census transform: A novel hardware-oriented stereovision algorithm. *Computer Vision and Image Understanding*, 117(1):29–41, 2013. 73
- [232] Jaeseung Lim, Yong-Ho Kim, and Sangkeun Lee. A census transform-based robust stereo matching under radiometric changes. In *APSIPA*, pages 1–4. IEEE, 2016. 73
- [233] Jaeryun Ko and Yo-Sung Ho. Stereo matching using census transform of adaptive window sizes with gradient images. In *APSIPA*, pages 1–4. IEEE, 2016. 73
- [234] Heiko Hirschmüller, Peter R. Innocent, and Jon Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision*, 47(1):229–246, Apr 2002. ISSN 1573-1405. doi: 10.1023/A:1014554110407. URL <https://doi.org/10.1023/A:1014554110407>. 73, 74
- [235] Michael Bleyer and Margrit Gelautz. A layered stereo matching algorithm using image segmentation and global visibility constraints. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59(3):128 – 150, 2005. ISSN 0924-2716. doi: <https://doi.org/10.1016/j.isprsjprs.2005.02.008>. URL <http://www.sciencedirect.com/science/article/pii/S0924271605000109>. 73, 74
- [236] Kuk-Jin Yoon and In So Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):650–656, April 2006. ISSN 0162-8828. doi: 10.1109/TPAMI.2006.70. 73, 74

- [237] Q. Yang, L. Wang, R. Yang, H. Stewnius, and D. Nistr. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):492–504, March 2009. ISSN 0162-8828. doi: 10.1109/TPAMI.2008.99. 73, 74
- [238] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 23(3):600–608, August 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015766. URL <http://doi.acm.org/10.1145/1015706.1015766>. 73, 74
- [239] Cheng Lei, J. Selzer, and Yee-Hong Yang. Region-tree based stereo using dynamic programming optimization. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2378–2385, June 2006. doi: 10.1109/CVPR.2006.251. 73, 74
- [240] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, Feb 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1166. 73, 74, 82, 86, 87
- [241] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 508–515 vol.2, July 2001. doi: 10.1109/ICCV.2001.937668. 74
- [242] Jian Sun, Yin Li, S. B. Kang, and Heung-Yeung Shum. Symmetric stereo matching for occlusion handling. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 399–406 vol. 2, June 2005. doi: 10.1109/CVPR.2005.337. 74
- [243] G. Van Meerbergen, M. Vergauwen, M. Pollefeys, and L. Van Gool. A hierarchical symmetric stereo algorithm using dynamic programming. *International Journal of Computer Vision*, 47(1):275–285, Apr 2002. ISSN 1573-1405. doi: 10.1023/A:1014562312225. URL <https://doi.org/10.1023/A:1014562312225>. 74

- [244] O. Veksler. Stereo correspondence by dynamic programming on a tree. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 384–390 vol. 2, June 2005. doi: 10.1109/CVPR.2005.334. 74
- [245] H. Hirschmüller and De Montfort University. *Stereo Vision Based Mapping and Immediate Virtual Walkthroughs*. De Montfort University, 2003. URL <https://books.google.com.sg/books?id=0s3QmgEACAAJ>. 74
- [246] Jiho Chang Hochul Shin Eul-Gyoon Lim Jae Il Cho Daehwan Hwang Seung-min Choi, Jae-Chan Jeong. Implementation of real-time post-processing for high-quality stereo vision. In *ETRI Journal 2015; 37(4): 752-765.*, 2017. 74, 87
- [247] Y Berdnikov and Dmitriy Vatolin. Real-time depth map occlusion filling and scene background restoration for projected-pattern-based depth camera. pages 1–4, 09 2011. 74, 75
- [248] Jianhong Shen and Tony F. Chan. Mathematical models for local nontexture inpaintings. *SIAM Journal of Applied Mathematics*, 62:1019–1043, 2002. 74, 75
- [249] I. Daribo and H. Saito. A novel inpainting-based layered depth video for 3dtv. *IEEE Transactions on Broadcasting*, 57(2):533–541, June 2011. ISSN 0018-9316. doi: 10.1109/TBC.2011.2125110. 74, 75
- [250] A. Hervieu, N. Papadakis, A. Bugeau, P. Gargallo, and V. Caselles. Stereoscopic image inpainting: Distinct depth maps and images inpainting. In *2010 20th International Conference on Pattern Recognition*, pages 4101–4104, Aug 2010. doi: 10.1109/ICPR.2010.997. 74
- [251] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In Ron Kimmel, Reinhard Klette, and Akihiro Sugimoto, editors, *Computer Vision – ACCV 2010*, pages 25–38, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-19315-6. 74
- [252] Death of elaine herzberg. https://en.wikipedia.org/wiki/Death_of_Elaine_Herzberg, 2018. 78

- [253] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 79
- [254] Jannik Fritsch, Tobias Kuehnl, and Andreas Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2013. 79
- [255] A. Mousavian, H. Pirsiavash, and J. Koeck. Joint semantic segmentation and depth estimation with deep convolutional networks. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 611–619, Oct 2016. doi: 10.1109/3DV.2016.69. 99
- [256] Omid Hosseini Jafari, Oliver Groth, Alexander Kirillov, Michael Ying Yang, and Carsten Rother. Analyzing modular cnn architectures for joint depth prediction and semantic segmentation. 05 2017. doi: 10.1109/ICRA.2017.7989537. 99
- [257] Peng Wang, Xiaohui Shen, Zhe Lin, S. Cohen, B. Price, and A. Yuille. Towards unified depth and semantic prediction from a single image. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2800–2809, June 2015. doi: 10.1109/CVPR.2015.7298897. 99
- [258] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, April 2017. ISSN 0162-8828. doi: 10.1109/TPAMI.2016.2572683. URL <https://doi.org/10.1109/TPAMI.2016.2572683>. 99
- [259] Normal lighting condition flight. <https://www.youtube.com/watch?v=9NuLg3ouWmM>, 2018. 129, 132
- [260] Challenging lighting condition flight. <https://www.youtube.com/watch?v=1AbfREny30Q>, 2018. 129, 132
- [261] Dong-Geol Choi, Yunsu Bok, Jun-Sik Kim, and In-So Kweon. Extrinsic calibration of 2d laser sensors. In *ICRA*, pages 3027–3033. IEEE, 2014. 144
- [262] The illustration of five level of autonomous driving. <https://medium.com/iotforall/>

- the-5-autonomous-driving-levels-explained-b92a5e834928, 2018. 154
- [263] Qiao Liang and Wang Xiangsui. Unrestricted warfare. In *Peoples Liberation Army Literature and Arts Publishing House*, 1999. 155
- [264] Ryan A. Chisholm, Jinqiang Cui, Shawn K. Y. Lum, and Ben M. Chen. Uav lidar for below-canopy forest surveys. *Journal of Unmanned Vehicle Systems*, 01(01):61–68, 2013. doi: 10.1139/juvs-2013-0017. URL <https://doi.org/10.1139/juvs-2013-0017>. 156
- [265] T. Lupton and S. Sukkariéh. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics*, 28(1):61–76, Feb 2012. ISSN 1552-3098. doi: 10.1109/TRO.2011.2170332. 159, 161, 171, 172
- [266] J. Lacambre, M. Narozy, and J. Louge. Limitations of the unscented kalman filter for the attitude determination on an inertial navigation system. In *2013 IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*, pages 187–192, Aug 2013. doi: 10.1109/DSP-SPE.2013.6642588. 160
- [267] Tightly-coupled visual-inertial sensor fusion based on imu pre-integration. 2016. 161, 175
- [268] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Robotics: Science and Systems*, 2015. 162
- [269] Msckf demo video. <https://www.youtube.com/watch?v=LaXc-jmN89U>, 2018. 164
- [270] Msckf failure report. https://github.com/KumarRobotics/msckf_vio/issues/22, 2018. 165
- [271] T. Lupton and S. Sukkariéh. Efficient integration of inertial observations into visual slam without initialization. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1547–1552, Oct 2009. doi: 10.1109/IROS.2009.5354267. 170, 171

- [272] T. Qin and S. Shen. Robust initialization of monocular visual-inertial estimation on aerial robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4225–4232, Sep. 2017. doi: 10.1109/IROS.2017.8206284. 171
- [273] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart. Real-time on-board visual-inertial state estimation and self-calibration of mavs in unknown environments. In *2012 IEEE International Conference on Robotics and Automation*, pages 957–964, May 2012. doi: 10.1109/ICRA.2012.6225147. 171
- [274] S. Shen, N. Michael, and V. Kumar. Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft mavs. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5303–5310, May 2015. doi: 10.1109/ICRA.2015.7139939. 171, 175
- [275] D. Galvez-Lpez and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, Oct 2012. ISSN 1552-3098. doi: 10.1109/TRO.2012.2197158. 171
- [276] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. Scale drift-aware large scale monocular SLAM. In *Robotics: Science and Systems*. The MIT Press, 2010. 171
- [277] T. Dong-Si and A. I. Mourikis. Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis. In *2011 IEEE International Conference on Robotics and Automation*, pages 5655–5662, May 2011. doi: 10.1109/ICRA.2011.5980267. 171
- [278] K. Eickenhoff, L. Paull, and G. Huang. Decoupled, consistent node removal and edge sparsification for graph-based slam. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3275–3282, Oct 2016. doi: 10.1109/IROS.2016.7759505. 171
- [279] S Yuan, MJ Er, and N Wang. Development control and navigation of octocopter. In *Control and Automation (ICCA), 2013 10th IEEE International Conference on*, 2013. 189

- [280] H. Umari and S. Mukhopadhyay. Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1396–1402, Sept 2017. doi: 10.1109/IROS.2017.8202319. 190
- [281] Eitan Marder-Eppstein, Eric Berger, Tully Foote, Brian Gerkey, and Kurt Konolige. The office marathon: Robust navigation in an indoor office environment. In *International Conference on Robotics and Automation*, 2010. 190
- [282] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics Automation Magazine*, 4(1):23–33, March 1997. ISSN 1070-9932. doi: 10.1109/100.580977. 190
- [283] Gerald L Smith, Stanley F Schmidt, and Leonard A McGee. *Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle*. National Aeronautics and Space Administration, 1962. 201
- [284] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960. 201
- [285] G Welch and G Bishop. An introduction to the kalman filter. university of north carolina, department of computer science. Technical report, TR 95-041, 1995. 201
- [286] Topological map. <https://www.mapbox.com/>, 2018. 217

Appendix A

Jacobian and covariance for coplanar and orthogonal constraint let r_{co}^j denotes residual error from coplanar constraint for plane j and r_{ort} denotes the residual alignment error from orthogonal constraint

$$r_{ort} = n_a \cdot n_b = (R_k I_k^a \times R_v I_v^a) \cdot (R_k I_k^b \times R_v I_v^b) \quad (1)$$

$$r_{co}^j = (R_k I_k^j \times R_v I_v^j) \cdot (R_k P_k^j + t_k - R_v P_v^j - t_v) \quad (2)$$

The Jacobian of a coplanar constraint with respect to rotations and translations constructed from ξ can be found by

$$J_{co} = \begin{bmatrix} (R_k I_k^j \times (R_v I_v^j) \cdot (R_k P_k^j + t_k - R_v P_v^j - t_v)) + R_k(P_k) \times n_k v \\ n_k v \\ -R_v I_v^j \times (R_k I_k^j \cdot (R_v P_v^j + t_v - R_v P_k^j - t_k)) - R_k(P_k) \times n_k v \\ -n_k v \end{bmatrix} \quad (3)$$

This is done by using chain rules. Similarly the Jacobian of orthogonality can be found by

$$J_{ort} = \begin{bmatrix} R_k I_k^a \times (R_v I_v^a \times (R_k I_k^b \times R_v I_v^b) + R_k I_k^b \times (R_v I_v^b \times (R_k I_k^a \times R_v I_v^a)) \\ -R_k I_k^a \times (R_v I_v^a \times (R_k I_k^b \times R_v I_v^b) - R_k I_k^b \times (R_v I_v^b \times (R_k I_k^a \times R_v I_v^a)) \end{bmatrix} \quad (4)$$

The corresponding variance in both constraint can be found by

$$\sigma_{co} = n_{kv}^T \sigma_{co_t} n_{kv}^T + (R_k P_k^j + t_k - R_v P_v^j - t_v)_{kv}^T \sigma_{co_n} (R_k P_k^j + t_k - R_v P_v^j - t_v)_{kv}^T \quad (5)$$

where

$$\sigma_{co_t} = R_v \sigma_{co_tv} R_v^T + P_v \sigma_{co_tv} P_v^T + R_k \sigma_{co_tk} R_k^T + P_k \sigma_{co_tk} P_k^T \quad (6)$$

$$\sigma_{co_n} = n_v R_k \sigma_{co_nk} R_k^T n_v^T + n_k R_v \sigma_{co_nj} R_v^T n_k^T \quad (7)$$

$$\sigma_{ort_n} = n_{vk}^a R_v^T R_k \sigma_{co_nka} R_k^T R_v^T n_{vk}^{a^T} + n_{vk}^b R_k^T R_v \sigma_{co_nkb} R_v^T R_k^T n_{vk}^{b^T} \quad (8)$$