

Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

Fusion-SLAM by Combining RGB-D SLAM and Rat SLAM

A thesis presented in partial fulfilment of the requirements for the degree of

Master of Engineering

In

Mechatronics

At

Massey University,

Albany, New Zealand.

Robert Tsunemichi Tubman

2016

Abstract

Robotic Simultaneous Localization and Mapping (SLAM) is the problem of solving how to create a map of the environment while localizing the robot in the map being created. This presents a causality dilemma where the map needs to be created in order to localize the robot, but the robot also needs to be localized in order to create the map. In past research there have been many solutions to this problem ranging from Extended Kalman Filter (EKF) to Graph SLAM systems. There has also been extensive research in bioinspired methods, like ratSLAM implemented in aerial and land-based robots. The different research setups use sensors such as Time of Flight (ToF) e.g. laser scanners and passive devices e.g. cameras. Over the past few years a new type of combined apparatus has been developed by Microsoft called the Kinect. It combines active and passive sensing elements and aligns the data in a way which allows for efficient implementation in robotic systems. This has led to the Kinect being implemented in new research and many studies, mostly around RGB-D SLAM. However these methods generally require a continuous stream of images and become inaccurate when exposed to ambiguous environments.

This thesis presents the design and implementation of a fusion algorithm to solve the robotic SLAM problem. The study starts by analysing existing methods to determine what research has been done. It then proceeds to introduce the components used in this study and the Fusion Algorithm. The algorithm incorporates the colour and depth data extraction and manipulation methods used in the RGB-D SLAM system while also implementing a mapping step similar to the grid cell and firing field functions found in the ratSLAM. This method improves upon the RGB-D SLAM's weakness of requiring a continuous stream and ambiguous images. An experiment is then conducted on the developed system to determine the extent to which it has solved the SLAM problem. Moreover, the success rate for finding a node in a cell and matching its pose is also investigated.

In conclusion, this research presents a novel algorithm for successfully solving the robotic SLAM problem. The proposed algorithm also helps improve the system's efficiency in navigation, odometry error correction, and scan matching vulnerabilities in feature sparse views.

Acknowledgments

I would like to express my deepest gratitude and sincerest appreciation to Dr Khalid Arif for his supervision, enthusiastic guidance and continuous encouragement throughout the course of this study.

I would also like to express my appreciation to my co-supervisor A/Professor Johan Potgieter, and the University staff for continuous support and encouragement throughout the course of this study.

I am also thankful to my family and friends for continuous support, encouragement and emotional support throughout all of my achievements in life.

Table of Contents

Abstract	i
Acknowledgments	ii
Table of Contents	iii
List of Figures	v
List of Abbreviations.....	x
Chapter 1 - Introduction	1
1.1 - Problem Statement	1
1.2 - Proposed Solution and Scope.....	2
1.3 - Thesis Layout	2
Chapter 2 - Literature Review	3
2.1 - EKF-SLAM Systems	3
2.2 - RatSLAM Systems.....	5
2.3 - RGB-D SLAM Systems.....	9
2.3.1 - 2D Mapping.....	9
2.3.2 - 3D Mapping.....	11
2.4 - Analysis and Summary of Findings	15
Chapter 3 - Robotic System Setup	17
3.1 - Hardware Components.....	17
3.1.1 - P3DX Mobile Robot Base	19
3.1.2 - Microsoft Kinect Sensor.....	20
3.1.3 - Custom Neck Platform	21
3.1.4 - Arduino Neck control.....	21
3.2 – Software Components	23
3.2.1 – Robot Operating System.....	23
3.2.2 - ARIA	24
3.2.3 - Standard C++, Qt and Boost.....	24
3.2.4 - Feature Detection and Matching Implemented	24
Chapter 4 - Fusion SLAM System.....	28
4.1 – RGB-D SLAM Elements.....	28
4.2 - RatSLAM Elements	30
4.3 - System Architecture	31
4.3.1 - Image Call-back Loop	35
4.3.2 - Odometry Call-back Loop.....	37
4.3.3 - Neck Angle Call-back Loop.....	42

4.3.4 - Robot Process Loop.....	42
4.3.5 - Robot Navigation.....	43
Chapter 5 - Experiment and Results	45
5.1 - Experimental Setup	45
5.1.1 - Testing the Robotic Fusion SLAM Algorithm.....	45
5.1.2 – Testing the Image Search Algorithm	47
5.1.3 – Testing the Image Node Transform Algorithm	47
5.2 - Experimental Results	49
5.2.1 – Results of the Robotic Fusion SLAM Algorithm	49
5.2.2 – Results of the Image Search Algorithm	57
5.2.3 – Results of the Image Node Transform Algorithm	60
Chapter 6 - Conclusion and Further Research	63
6.1 – Conclusion.....	63
6.2 - Further Research	66
References	67
Appendix A	70
Appendix B	72
Appendix C	74
Appendix D	76

List of Figures

<i>Fig 2.1: Plane data on the centre of gravity of the point segments.</i>	.5
<i>Fig 2.2: The temporal map of the ratSLAM navigation.</i>	.8
<i>Fig 2.3: Visualization of different edge detections.</i>	.14
<i>Fig 3.1: Robot assembled with hardware components.</i>	.18
<i>Fig 3.2: P3DX robot base dimensions.</i>	.19
<i>Fig 3.3: Top-down view of P3DX's front sonar arrangement.</i>	.19
<i>Fig 3.4: Bumper arrangement on P3DX robot base.</i>	.20
<i>Fig 3.5: Speckle image produced by the Kinect.</i>	.20
<i>Fig 3.6: Kinect Sensor components</i>	.21
<i>Fig 3.7: The neck platform height and the angle range.</i>	.22
<i>Fig 3.8: The ROS system architecture.</i>	.24
<i>Fig 3.9: SIFT matching algorithm</i>	.25
<i>Fig 3.10: SURF matching algorithm</i>	.26
<i>Fig 4.1: RGB-D SLAM system layout.</i>	.29
<i>Fig 4.2: ratSLAM relation between local view cells and pose cells.</i>	.30
<i>Fig 4.3: RGB-D SLAM and ratSLAM Fusion system layout.</i>	.32
<i>Fig 4.4: Topics layout in the robot system.</i>	.33
<i>Fig 4.5: Main Processing Loop.</i>	.34
<i>Fig 4.6: The grid cell neighbour calculation from the centre of the current cell.</i>	.40
<i>Fig 4.7: The bottom red cell is adjacent to both the blue cells and the green cell but is not adjacent to the top red cell. The bottom red cell also contains no other adjacency information regarding the other cells nearby even if the blue cell directly above to the left is mapped for example.</i>	.41

<i>Fig 4.8: The robot navigation planner. The starting point is the light blue cell and the goal point is the dark blue cell. The robot will keep moving to the lighter cells until the goal cell is reached.</i>	.44
<i>Fig 5.1: Layout of the experiment environment. The red boxes are the target positions and the blue objects are obstacles in the environment. The orange line is the path of the robot when creating the initial map..</i>	.45
<i>Fig 5.2: The moderately cluttered desk view.</i>	.47
<i>Fig 5.3: Cluttered cabinet view.</i>	.48
<i>Fig 5.4: Uncluttered blank wall.</i>	.48
<i>Fig 5.5: The robot's internal odometry vs. actual odometry in the environment using the SIFT algorithm for image nodes.</i>	.50
<i>Fig 5.6: The robot's internal odometry vs. actual odometry in the environment using the SURF algorithm for image nodes.</i>	.50
<i>Fig 5.7: The robot's internal odometry vs. actual odometry in the environment using the ORB algorithm for image nodes.</i>	.51
<i>Fig 5.8: The robot's internal odometry vs. actual odometry in the environment with all of the algorithms data combined.</i>	.51
<i>Fig 5.9: The robot's actual odometry at the goal before navigating vs. actual odometry after navigating the environment using the SIFT algorithm for image nodes.</i>	.52
<i>Fig 5.10: The robot's actual odometry at the goal before navigating vs. actual odometry after navigating the environment using the SURF algorithm for image nodes.</i>	.52
<i>Fig 5.11: The robot's actual odometry at the goal before navigating vs. actual odometry after navigating the environment using the ORB algorithm for image nodes.</i>	.53
<i>Fig 5.12: The robot's actual odometry at the goal before navigating vs. actual odometry after navigating the environment with the entire algorithm data combined.</i>	.53
<i>Fig 5.13: The total number of grid cells created.</i>	.54
<i>Fig 5.14: The average position of the goal cell.</i>	.55
<i>Fig 5.15: The total number of image cells created.</i>	.55

<i>Fig 5.16: The portion of image nodes that are part of the goal cell.</i>56
<i>Fig 5.17: The total number of grid cells created in the map.</i>56
<i>Fig 5.18: The total number of image nodes created in the map.</i>57
<i>Fig 5.19: Transform estimates obtained by actively searching for an image node specified while using the SIFT algorithm.</i>58
<i>Fig 5.20: Transform estimates obtained by actively searching for an image node specified while using the SURF algorithm.</i>59
<i>Fig 5.21: Transform estimates obtained by actively searching for an image node specified while using the ORB algorithm.</i>59
<i>Fig 5.22: Distance transforms calculated by the stationary robot viewing the moderately cluttered desk.</i>61
<i>Fig 5.23: Distance transforms calculated by the stationary robot viewing the cluttered cabinet</i>61
<i>Fig 5.24: Distance transforms calculated by the stationary robot viewing the blank wall.</i>		.62

List of Tables

<i>Table 3.1: Hardware and software components used in the project.</i>	.17
<i>Table 4.1: Pseudocode of the image callback process</i>	.36
<i>Table 4.2: Pseudocode of the odometry callback process</i>	.37
<i>Table 4.3: Formula used to calculate the grid cell neighbours where O is the current cell centre pose and d is the grid cell radius. The current cell centre x pose is O_x and y pose is O_y.</i>	.39
<i>Table 5.1: Steps taken to test the robot in two stages.</i>	.46
<i>Table 5.2: The average distance covered, error accumulated and the standard deviation for internal vs. measured odometry results.</i>	.49
<i>Table 5.3: The average distance covered, error accumulated and the standard deviation for odometry results before and after navigation.</i>	.49
<i>Table 5.4: The total average distance covered, error accumulated and the standard deviation for odometry results.</i>	.49
<i>Table 5.5: The average transform estimates obtained by the different algorithms observing a stationary, desk, cabinet, and blank wall.</i>	.60
<i>Table A.1: Measured odometry at the goal position before and after navigation using the SIFT algorithm for the image nodes.</i>	.70
<i>Table A.2: Internal and measured odometry at the goal position after navigation using the SIFT algorithm for the image nodes.</i>	.70
<i>Table A.3: Measured odometry at the goal position before and after navigation using the SURF algorithm for the image nodes.</i>	.70
<i>Table A.4: Internal and measured odometry at the goal position after navigation using the SURF algorithm for the image nodes.</i>	.71
<i>Table A.5: Measured odometry at the goal position before and after navigation using the ORB algorithm for the image nodes.</i>	.71
<i>Table A.6: Internal and measured odometry at the goal position after navigation using the ORB algorithm for the image nodes.</i>	.71

<i>Table B.1: Map data with the SIFT algorithm for image nodes.</i>72
<i>Table B.2: Map data with the SURF algorithm for image nodes.</i>72
<i>Table B.3: Map Data with the ORB algorithm for image nodes.</i>73
<i>Table C.1: The transform estimate of the difference in the robot's pose when viewing the goal image while using SIFT.</i>74
<i>Table C.2: The transform estimate of the difference in the robot's pose when viewing the goal image while using SURF.</i>74
<i>Table C.3: The transform estimate of the difference in the robot's pose when viewing the goal image while using ORB.</i>75
<i>Table D.1: The transforms obtain while scan matching when observing the desk.</i>		.76
<i>Table D.2: The transforms obtain while scan matching when observing the cabinet.</i>		.78
<i>Table D.3: The transforms obtain while scan matching when observing the white wall.</i>		.80

List of Abbreviations

BRIEF – Binary Robust Independent Elementary Features

CMOS – Complimentary Metal-Oxide Semi-conductor

DoF – Degrees of Freedom

EKF - Extended Kalman Filter

FAST – Features from Accelerated Segment Test

GTSAM – Georgia Tech Smoothing and Mapping

ICP – Iterative Closest Point

IR – Infra Red

MAV - Micro Aerial Vehicle

ORB – Oriented FAST and Rotated BRIEF

RANSAC – Random Sample Consensus

RFID – Radio Frequency Identification Device

RGB-D – Red Green Blue Depth, Colour and Depth data

ROS - Robot Operating System

SIFT – Scale Invariant Feature Transform

SLAM - Simultaneous Localisation and Mapping

SPmodel - Symmetries and Perturbations model

SURF – Speeded Up Robust Features

ToF – Time of Flight

TSDF - Truncated Signed Distance Function