Title of project placed here

Name of author placed here

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

A dissertation presented in part fulfilment of the requirements of the Degree of Master of Science at The University of Glasgow

Date of submission placed here

Abstract

abstract goes here

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: _____ Signature: _____

Acknowledgements

acknowledgements go here sdfsdf

Contents

Chapter 1

Introduction

1.1 Sentence

I completed a program which allowed a turtlebot to autonomously map an environment using SLAM as well as create an inventory of items found in the environment.

1.2 Importance/Context/Motivation

Mobile robots are becoming more and more affordable and accessible which has allowed developers to take advantage of their applications in many different ways.

SLAM has it's application in rescue robots which http://www.aaai.org/Pressroom/Releases/release-02-0910.php these robots required realtime control and utilised only video streams to identify and rescue people.

This compares to this which utilises rugged mobile robots to create SLAM maps of mine shafts with minimal supersvision. This can then be applied to areas which are too unsafe/ small for humans to access. https://miningrox.informatik.tu-freiberg.de/en/

More affordable Drones can also be used to increasingly accurate create maps of property as well

Object detection can be used to detect humans via heat sensors etc. Aswell as identifying bombs etc.

1.3 Objectives/Hypothesis Karl Popper/Problem statement

1.4 Description of Objectives

1.5 How I achieved it

1.5.1 System Diagram

1.6 Outline of the dissertation

Chapter 2

Background

Existing/similar applications

Why choosing to talk about these things.

More references.

Include Photos of everything

2.1 Papers

Semantic Slam

2.2 Robot Operating System (ROS)

2.2.1 What is ROS

ROS is a meta-operating system which provides a collection of tools and conventions to aid the writing of robot software. This includes an abstraction of hardware and communications as well as tools including message passing, package management and low level controllers.

2.2.2 Why do people use ROS

ROS features open-source licenses and a large growing collection of packages which contribute to a vibrant ecosystem of developers and researchers working on applications. Some of these packages includes powerful applications such as the Gazebo simulator and the visualisation tool RViz as well as complex algorithms such as SLAM and drivers such as OpenNI.

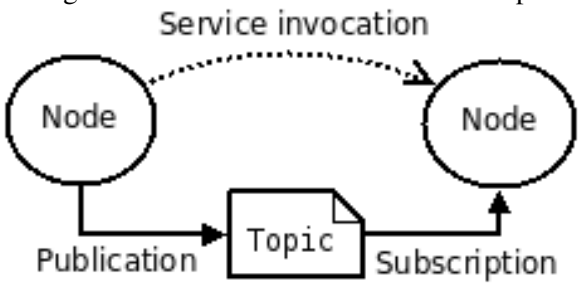
ROS encourages software engineering principles of loose coupling and abstraction which simplifies software integration and reuse as well as meaning that ROS is generally independent of hardware specifics. ROS features standard implementations in popular programming languages such as Python and C++ which increases adoption.

2.2.3 How does ROS works?

ROS consists of some key components within the communications infrastructure.

- Nodes are modular processes performing computation which can communicate with each other using topics or services. A system may consist of many nodes which perform many different tasks.
- Topics are named buses which provide a clear message passing interface between nodes. This is achieved through an anonymous publish/subscribe mechanism. This mechanism allows many-to-many transport.
- Messages are a data structure which are published by nodes to topics. They support strongly typed fields such as primitives or arrays and are either predefined or user-defined.
- The ROS core or Master provides a centralized node which locates and negotiates communications between nodes as well providing naming and registration services.
- Services forgo the many-to-many paradigm and provide a request/reply interaction via servers and clients.

Figure 2.1: Illustration of the ROS concepts



For example a simple robot vehicle system can consist of three nodes. The first node controls the robot's ultrasonic range finder and is publishing a stream of range messages taken from the sensor along the `sensor/sonar` topic.

The second node controls the robot's movement hardware and is publishing the robot's orientation on the `geometry_msgs/pose` topic. It also contains an actionlib server which responses to requests to change the robot's positioning.

The third node controls the robot's movement and is subscribed to the `sensor/sonar` topic and contains a client of the actionlib server. While processing the range messages if the range indicated is very small the node can use the actionlib service to request an adjustment in the orientation of the robot thus meaning the robot will avoid collisions.

Include diagram describing example

RViz

Rviz is a visualisation tool in ROS which provides 3d visualisations of sensor data and robot states, such as cameras, lasers and joint states. Add modular componets which can be customized.

Gazebo

Gazebo provides a simulated environment to build and test applications in ROS. A simulated robot will feature all the same sensors as a real one and a model world can be created. Physics, models, programmable features such as doors orlights.

2.3 Mobile Robots

Mobile robots are robots that are not fixed to one location and are capable of moving around their environments. Movement is achieved generally through legs, wheels or tracks but aerial and nautical robots can use propellors. The robot can be controlled through manual-tele operation involving a human driver, line-following which follow visual cues such as painted lines to navigate or autonomously.

These robots have many applications covering military, agricultural, rescue, transport and domestic usage.

For example: Unmanned Ground Vehicles can include military robots but also self-driving cars and delivery robots. Aerial robots (UAVs) , underwate(AUVS).

Drone missles, resucue robots, self driving cars, roomba, mail robots.

2.3.1 Turtlebot 2

The Turtlebot 2 is a part of a series of low cost, open-source mobile robots by Clearpath Robotics**. These robots are intended for educational and research purposes and provide an 'low-barrier-of-entry' platform for development. The robot consists of a Kobuki base, a computer running ROS and some form of RGB-D or Stereo Camera such as the Xbox Kinect or Asus Xtion.

The Kobuki Base contains IR, bump, cliff and wheel drop sensors as well as different power connectors and ports.

The Kobuki Base requires a computer running ROS to communicate with, in most cases this can simply involve a laptop attached to the top of the base connected by a usb cable. However in some cases it may be more convienent to attach a small netbook or Raspberry Pi to the base and use a seperate computer to communicate with the host.

- Lots of software standard packages support, gazebo models, teleoperation, slam as well as large and growing community of stuff... - Abstracted hardware, no need to understand how it works, just recieve the data,

How it works in ros Odometry

2.4 Cameras

Why do we need them Why discuss both of these? - exemplar

2.4.1 RGB-D

- How it works
- Active camera - range finder
- Kinect specs

2.4.2 Stereo

- Passive camera, works well outdoors
- How depth is calculated
- How images are combined
- About the zed cam - specs

2.5 SLAM

- What is the problem, include references e.g. to correlation
- explain the proble
- solutions to the problem

SLAM is the problem of simultaneously localising (finding the pose and orientation) of a camera within it's surroundings at the same time as mapping the structure of the environment. This requires a robot or camera with the ability to produce odometry readings as well as camera with a range measurement device, either a range finder or found in a stereo camera.

SLAM forms the basis of navigation in most mobile robots, meaning unknown environments can be explored and mapped without the need for technology such as GPS which becomes innaccurate within two metres or in interiors. It has applications in a range of manned and autonomous robots. This includes UAV's, underwater robots and domestic robots such automatic lawnmowers. SLAM is a key component in the development of self-driving cars These cars which are driven along routes while performing SLAM capturing location, feature and obstacle data. Once the map is completed it is processed and the cars are driven autonomously along these routes updating the map as necessary.

This is considered to be a solved problem in Computer Science and their are many different approaches for finding a solution. [1]

During the development of the SLAM problem researchers established the correlation between estimates of a robots location and the landmarks which consituate the map

Building maps once can work, but it can be distrupted by changes in the environment, e.g lighting

We know the position of the landmarks Localisation requires landmarks, estimate the location of these and estimate it's own pose relative to it. This can be inaccurate due to odometry or inaccurate readings, rely on each. 95GEt estimate of where the robot probably is, get it's location relative to landmarks

Cannot fully decouple localisation and mapping as map is needed for localisation and a pose estimate is needed for mapping

Areas which are missing. Passive.

Given: Robot's controls, Observations(laser scans) Wanted: map of environment Path of the robot.

Probablistic techniques, uncertainity of robots motions and observations... represent uncertnity typically in a Gaussian distribution, error can accumalate over time. Take the uncertanity into account.

Slam is difficult because both path and map are unknown, but they are correlated and depend on each other

On a high level SLAM is solved by using the environment to update the pose of the robot. Using odometry as the sole measurement of localisation has an element of uncertainity due to extraneous factors such as wheels slipping on different environments meaning that a stated distance given by an odometry reading may be over or under estimated. Therefore laser scans or other forms of depth readings are used to correct the robot's position by extracting features from the surrounding environment. These are called landmarks and can be extracted by various methods such as Random Sampling Consensus (RANSAC) and provide a growing map of the enviroment.

The robot estimates the location of it's own pose relative to these different landmarks which are all correlated together and increase with successive observations. [**?**]

The combined localisation and mapping estimates as a single estimation problem are convergent

Extended Kalman Filter is used to update where the robot presumes it is based on these landmarks and keeps track of the uncertainity of the robots position and the landmarks seen.

When robot moves the uncertainity of the new position is updated. Landmarks are extracted and reobserved ones are used to update the robots position. New landmarks are added to the EKF.

***Diagram. of sequence of reaffirmation.

Uncertanity is kept track of.

2.5.1 RtabMap

- intro RTAB-Map is a RGB-D graph based SLAM approach. What it is

- Loop closure technique. How it works

Online slam, seeks to recover only the most recent pose, not the entire path algorithm

2.5.2 Others

Compare rtabmap and other papers Cartographer

2.6 Frontier Exploration

2.7 Object Detection

At low levels of the evolutionary scale the eye acts as a type of goal detector rather than a camera.

The information provided by past experience have a greater say on how a scene has been interpreted than immediate information provided by external organs. The eye the brain the computer p208

However in contrast to simple organisms whose focus is on detection resulting in a sensitive and broad field of vision the programme must have the ability to classify which requires precise resolution when required.

Everything is seen for the first time. Our sensory systems does not keep telling us things we already know as most important environmental information stays constant, this process is called adaption and happens not only at a retinal level but at higher levels in the brain.

What different types of neural structure are used to extract information from a visual signal

1) areas of high level of features 2) depth/closeness 3) shapes/edge detection - particularly light and dark

Not feasible to search for objects at all locations in a visual field Partioning or perceptual organisation.

Law of proximity = Stimulus that are close together are percieved to be a group.

Geometric, Photometric modeling scene degmentation, naming+labeling,

Image matching: correlation approach

feature matching approach eges remain across two images of the same scene. relational matching approach

From page 281 –¿ talk about image labeling

Neurophysiologic studies have shown that, in humans, these two factors are the main responsible ones to drive attention. Bottom-up factors emanate from the scene and focus attention on regions whose features are sufficiently discriminative with respect to the features of their surroundings. On the other hand, top-down factors are derived from cognitive issues, such as knowledge about the current task. http://www.prip.tuwien.ac.at/people/krw/more/papers/2012/Antunez2012a.pdf

2.7.1 opencv

2.7.2 Feature Detection

2.7.3 Deep Learning

Tensorflow

Haar Cascades

Google Vision API

3D Detection

2.7.4 Summary and Discussion

What currently exists for completing the objectives

What are the Gaps, What do I have to do to fill the gaps

What methods are not appropriate to fill the gap

Chapter 3

Approach/Implementation/System Implementation

3.1 Anaylsis the approaches

What problems I encounted etc.

3.2 Mapping

3.2.1 Transforming data

3.2.2 Calibration

3.3 Frontier Exploration

3.4 Object Detection and Recognition

3.4.1 Blob Detection

3.4.2 Detecting Clusters - different methods

3.4.3 Creating Boxes

3.4.4 Tracking Boxes

3.4.5 Positioning Boxes in Map/Loop Closure

3.4.6 Recognising Objects

3.4.7 Publishing to Rviz/Rtabmap

3.5 The whole package - how to utilise

Chapter 4

Evaluation

4.1 Testing

Chapter 5

Conclusion

5.0.1 Future work

Appendix A

First appendix

A.1 Section of first appendix

Appendix B

Second appendix

Bibliography

[1] Hugh Durrant-Whyte, Fellow, IEEE, and Tim Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms, 2006.