Thesis

Tom Zurales

July 2025

Abstract

This research details the implementation and analysis of a novel, viewpoint aware method for map point culling for use in keypoint based visual SLAM systems. This method makes use of perspective dependent shells around each map point, allowing for the storage of overall observability metadata using constant additional space. This metadata allows for the overall probability of a point's existence to be continuously calculated using a simple Bayesian update step. This existence probability can be used in a myriad of ways. This research explores its use as a method of culling outdated map points, such as those originally seen on an object which has since moved, and as an extension to the RANSAC algorithm, providing the ability to select more robust map points. We provide the implementation of the perspective aware metadata shell as an open-source library, as well as an ORB_SLAM3 implementation utilizing the library for point culling and RANSAC improvement. Additionally, a set of co-registered visual-inertial-lidar datasets are released, containing scenarios specifically intended to exercise and characterize the performance of the system. Through our analysis, (discuss effects on well known, non-dynamic SLAM datasets, along with my datasets)

Contents

1 Introduction

This section provides an introduction to the limitations of SLAM systems, and discusses how the research prevented in this thesis intends to solve those shortcomings.

1.1 Motivation

1.1.1 Problem Statement

1.1.2 Research Questions

Does a probability model which identifies and culls outdated map points provide significant improvements to relocalization and tracking during map reuse in keypoint-based SLAM? To what extent do dynamic changes in a map affect mapping and relocalization performance? Can this be used as a heuristic to determine when to re-enable mapping on MAVs?

The spacial understanding provided by SLAM is not only useful, but necessary for systems intending to operate in and interact with physical environments. Virtual reality, robotics, and industrial automation all make use of SLAM to generate an internal map of the local and global environments. SLAM does have the distinction of being a ”solved” problem in the ideal case. If an agent is able to perfectly measure the environment, and is guaranteed to make correct data associations, then a perfect map can be generated and the agent's location within that map can be known with certainty. This ideal case makes several assumptions, paramount of which is the existence of ideal sensors, but there is a secondary assumption that the state of the world does not change.

It is obvious that the assumption of a static, unchanging world does not hold in practice. In fact, the inspiration for this research came from attempts to perform localization on the International Space Station (ISS). As of the time of publication, there are three mobile autonomous vehicles (MAVs) on board the ISS known as Astrobees. While used for numerous experiments and product development tasks on the ISS, the Astrobees are prone to failure due to loss of localization. The primary cause of this localization failure is the constant changes occurring on the ISS, including equipment changes, resupply missions, and any other activities which may change the visual features of the ISS.

The situation on the ISS is not unique, and would be experienced by any agent running SLAM in all but the most tightly controlled environments. VR goggles using use SLAM to determine their position in 3D space within a room must contend with new objects which are placed in the room, the changing images shown on the TV, people walking in and out of view, etc. Robots operating in an office environment must be robust to moved desks, the movement of people, and more. Even robotic operations in an unmanned space station (a situation proposed for the Lunar Gateway project) would need to be able to perform despite changing lighting conditions, moved equipment, other MAVs in view, etc. Overall, a requirement that a SLAM agent gets to operate in a pristine, unchanging environment would be an insurmountable barrier for real world use.

The field of research into making SLAM perform over long timeframes has been called lifetime SLAM[], eluding to the fact that SLAM systems with the requirement for an unchanging environment will still be able to operate successfully over short timeframes, but will struggle with missions which take place over multiple days, weeks, months, or years.

This research is specific to a subset of the greater SLAM problem known as keypoint-based, visual SLAM. The distinctions between these will be discussed in detail in the Background section, but a high-level overview is offered here. Keypoint-based visual SLAM operates on images taken over time. The core principal involves constructing a sparse 3D map of image features which are identifiable from multiple perspectives. This is accomplished through photogrammetry methods such as the 5-point method, which allows the depths of 5 matched pairs of points, and the parallax transformation to be determined from two 2d images [FACT CHECK and CITATION].

¡!– Go into a few more details about some of the other fields of research which are used by SLAM –¿

There are numerous keypoint-based visual SLAM implementations seeing use today, but all follow a relatively straightforward pipeline, defined as follows:

1. Determine an initial set of 3D points from two images with sufficient parallax 2. For subsequent images, match features with previously identified 3D points 3. Determine the transformation between the previous image and the new image which maximizes the number of map point alignments

Implementation differences tend to come from optimization steps, pruning of redundant data, anomaly handling, and additional sensor integrations. In order to achieve lifetime SLAM, the system must be able to determine what data is remaining static, what data is changing, and act accordingly. Imagine an art gallery with many paintings on the walls. If you were to visit this gallery on two separate occasions one year apart, the paintings on the walls may change, but you are able to identify that you are in the same gallery. People perform this contextual elimination of data which may change on a subconscious level []. Replicating this contextual awareness in SLAM allows systems to recognize and focus on unchanging data while ignoring dynamic data which could clutter and confuse the agent's internal map.

The benefits of eliminating data which is not helpful for long-term operations are plentiful. Just like culling redundant data, culling dynamic data reduces the overall size of the map. This reduces storage capacity requirements, while providing a smaller pool of data through which processes like Random Sample Consensus (RanSaC) need to search. A keypoint which was seen on an object which is later moved will always be an outlier in subsequent observations. Through culling of these dynamic data points, we can improve the speed and robustness of the several optimization steps, reduce overall system hardware requirements, and increase confidence in the accuracy and validity of the produced maps.

Numerous methods for improving SLAM's performance over long timeframes have been implemented, pushing the field of SLAM to the point where it is now seeing deployment in numerous dynamic environments. A discussion of several of these implementations takes place in the Background section, with a focus on each implementation's strengths, weaknesses, and overall effectiveness. An area that remains lacking is implementations on MAVs with limited compute. Due to their mobile nature, MAVs are inherently compute limited, as any additional weight and power consumption decreases capability and operation time. This prevents the inclusion of many popular models for dynamic data elimination such as image segmentation and semantic identification. Other methods utilizing statistical methods for point existence exist, but fail to fully utilize the vast array of data to update their predictions

1.2 Objectives and Scope

This research intends to build upon the previously developed probability models, in order to distill the update step of each map point's probability of existence into a simple Bayesian update step. The goals for this model are as follows:

1. To utilize constant additional space for each map point 2. To complete the update step in constant time 3. To resist updating confidence levels with redundant data

1.3 Contribution

Through this research, we introduce an incrementally updated directional confidence model for the existence of map points. This model differs from other point removal optimizations in several ways. First, this implementation avoids the use of neural networks, facilitating use on resource constrained hardware without facilities optimized to run them. Second, while other probability based point removal optimizations have been developed, this model introduces the idea of utilizing a continuously updated perspective dependent shell of metadata for each keypoint, which can be used to reduce the problem of point existence to a simple Bayesian update step. This implementation allows perspective of observation to play a role in the point's existence probability update step, and avoids some of the common a priori work such as prior estimation common to other point removal optimization techniques. This shell is implemented in both finite and continuous modalities, utilizing regular convex polyhedral shells in the finite case, and von-meiser fisher distributions on the sphere in the continuous case.

To facilitate future research, this model is released as an open-source library, which is compatible with any keypoint based visual SLAM implementation. Additionally, a collection of co-registered visual-inertial and LIDAR datasets is provided, containing instances of multiple traversals through the same environments with changes to scene contents. Information regarding the locations of these environmental changes is included in the dataset, facilitating the benchmarking of point removal optimization implementations.

1.4 Road Map

Chapter X of this thesis discusses the background of the general SLAM problem, covering the history, use cases, and general pipeline utilized by SLAM systems. This is followed by a deeper dive into keypoint-based visual SLAM, the sensor modality targeted by this research. We provide a brief survey of widely utilized extensions to the core SLAM algorithm which target deficiencies in the core pipeline. Finally, we discuss fields outside the scope of SLAM which provide insight and methodology into this research.

Chapter X discusses works related to this research, specifically focusing on extensions targeting improved performance in dynamic situations, with additional focus given to those methods which utilize point removal optimizations.

2 Background

In this chapter, we provide a high level overview of the definitions, objectives, and history of SLAM, in addition to an overview of the common sensor modalities found in SLAM systems. Following this, we discuss the stages of the SLAM pipeline in the generic case, followed by a more in-depth exploration of keypoint-based visual SLAM, the sensor modality targeted by this research. Next, we look into several of the widely adopted extensions to the base SLAM pipeline which address core issues and enhance performance. A discussion of extensions which have similar goals to this research is held for the chapter on related works.

2.1 SLAM Overview

This research is acts as an extension to keypoint-based visual SLAM; a term which warrants some explanation. But before exploring the specifics of keypoint-based visual SLAM, some background on the general SLAM problem is required. The idea behind SLAM is to simultaneously produce a map of an environment, and determine the position of the observer within the map based on a set of sensor data. The process differs heavily based on the sensor types being utilized. For example, LIDAR provides a direct measurement of 3D distances from the sensor, while an RGB camera must calculate them from correspondences between multiple frames. While implementations differ heavily, a common SLAM pipeline could be described as follows:

2.1.1 The SLAM Pipeline

This stage is responsible for the creation of the initial map.

There have been hundreds of SLAM implementations for a wide variety of sensors, commonly targeting combinations of monocular, stereo or RGBD cameras, IMUs, LIDARs, etc.

Due to it providing the motivation for this project, the Astrobee robots will me mentioned several times throughout this work. The Astrobee project was motivated by the desire to research human/robot interaction, robotic automation and inspection, and to provide a research platform on which companies and researchers could deploy software and hardware for testing in a micro-gravity environment. The Astrobee platform has been used to develop satellite rendezvous control algorithms, grippers to capture tumbling orbital debris, inspection methods to autonomously detect anomalous operation, and many other space habitation focused endeavors.

2.1.2 Keypoint-Based Visual SLAM

The term Keypoint-Based Visual SLAM refers to the SLAM modality which primarily utilizes key points extracted from images as the primary means of mapping and navigating. This is distinct from systems like LIDAR-based SLAM, which utilize direct distance measurements from a LIDAR sensor, or Dense Visual SLAM, which

2.1.3 Extensions to Core SLAM

2.2 Additional Fields of Research

2.2.1 Directional Probability

3 Related Work

Numerous methods have been developed to improve long-term SLAM performance, generalizing the problem to remove the constraint that the environment remains static. https://arxiv.org/pdf/2209.1 - ChangingSlam - Uses a Bayes filter to remove changing map points. Utilizes semantic detection to determine dynamic objects before filter is used, which prevents it from working on deformable objects.

3.1 Point Removal Optimizations

3.1.1 Semantics Based Implementations

3.1.2 Probability Based Optimizations

4 Experimental Analysis

4.1 Dataset Creation

4.1.1 Hardware

4.1.2 Structure

4.1.3 Dataset Overview

4.2 Evaluation Metrics

4.3 SLAM System Configurations

4.3.1 Parameter Tuning

4.4 Results

4.4.1 Quantitative Evaluation

4.4.2 Qualitative Evaluation

4.4.3 Ablation Study

5 Discussion

6 Conclusion and Future Work

Appendices

Algorithm Implementations

Full Results