Stereo Visual Simultaneous Localisation and Mapping for an Outdoor Wheeled Robot: A Front-End Study

Presented by:
Ryan Evan Wolf

Supervisor:
Dr M. S. Tšoeu
Dept. of Electrical and Electronics Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town in fulfilment of the academic requirements for a Masters of Science degree in Electrical Engineering.

September 29, 2019

Declaration

---

1. I am presenting this dissertation in FULL/PARTIAL fulfillment of the requirements for my degree.
2. I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own.
3. This thesis/dissertation has been submitted to the Turnitin module (or equivalent similarity and originality checking software) and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.
4. I hereby grant the University of Cape Town free license to reproduce for the purpose of research either the whole or any portion of the contents in any manner whatsoever of the above dissertation.

Signed by candidate

Signature:..... Ryan Evan Wolf

Date:.....10 February 2019.....

Abstract

For many mobile robotic systems, navigating an environment is a crucial step in autonomy and Visual Simultaneous Localisation and Mapping (vSLAM) has seen increased effective usage in this capacity. However, vSLAM is strongly dependent on the context in which it is applied, often using heuristic and special cases to provide efficiency and robustness. It is thus crucial to identify the important parameters and factors regarding a particular context as this heavily influences the necessary algorithms, processes, and hardware required for the best results.

In this body of work, a generic front-end stereo vSLAM pipeline is tested in the context of a small-scale outdoor wheeled robot that occupies less than $1m^3$ of volume. The scale of the vehicle constrained the available processing power, Field Of View (FOV), actuation systems, and image distortions present. A dataset was collected with a custom platform that consisted of a Point Grey Bumblebee (Discontinued) stereo camera and Nvidia Jetson TK1 processor. A stereo front-end feature tracking framework was described and evaluated both in simulation and experimentally where appropriate. It was found that scale adversely affected lighting conditions, FOV, baseline, and processing power available, all crucial factors to improve upon. The stereo constraint was effective for robustness criteria, but ineffective in terms of processing power and metric reconstruction. An overall absolute odometer error of 0.25-3m was produced on the dataset but was unable to run in real-time.

Acknowledgements

It has been a long journey and I am truly indebted and grateful to the myriad of people who have helped me along the way. I want to thank Mohohlo for all the time, effort and guidance afforded to me, it has made me a far better academic than I could have imagined.

To my parents, none of this would have been possible without your support and I am fortunate to have been given such opportunities thanks to your sacrifices. To my family, your kind words and encouragement have been treasured. To Nicole, you were there for me when it was needed most, and I am blessed to have you a part of this journey.

Contents

List of Figures

Chapter 1

Introduction

In this chapter the basic building blocks required of a mapping and localisation pipeline are described for use in autonomous navigation for a small-scale outdoor wheeled robot. For the purposes of this work, a small-scale robot refers to a system that occupies less than $1m^3$ of volume. This will begin with a very broad, abstract overview of Simultaneous Localisation and Mapping (SLAM), Visual SLAM (vSLAM) and indicate where this body of work is placed within the existing literature. The problem statement will then be presented followed by the aims, scope, and approach of the research.

1.1 A Brief Background to Simultaneous Localisation and Mapping

In robotics, a highly desirable outcome of many studies has been to reduce the reliance of robots on human input [2–6]. The autonomy, or intelligence of a robot depends on its ability to interpret sensor information, determine appropriate actions and execute them successfully. In the context of autonomous navigation, Nakhaeinia et al. [7] frame these requirements as follows: "A mobile robot as an intelligent system needs to sense the surroundings, perceive the working environment, plan a trajectory, and execute proper reaction using the information".

Without the ability to perceive the surrounding environment, planning a trajectory and reacting to external stimulus such as obstacles is difficult to achieve. For this reason, interpreting sensor data is considered a critical step in autonomous navigation [4, 8, 9]. In the context of autonomous navigation, a map and robot pose[1] could be one manner of interpreting the surrounding environment in terms of sensor data.

A prominent architecture for interpreting sensor data for navigation is known as SLAM [10–13]. In the context of SLAM, maps often take on either a topological or metric interpretation, where topological maps emphasise the relationship between locations, and metric maps emphasise location relationships in some unit, such as metres or degrees. Localisation refers to the pose within this map, however it is defined. Figure 1.1, illustrates a topological map where each map location is described as a collection of image patches, and a path could be described as a series of discrete map locations. Figure 1.2 illustrates a metric interpretation where a collection of Global Positioning System (GPS) represents a trajectory, and discrete three-dimensional (3D) points encode the surrounding environment.

Mapping and localisation initially started off as separate fields of study, but a key insight into improving accuracy and reliability of both fields was the concurrent approach [11]. Localising, whilst building a map of the environment using proprioceptive and exteroceptive sensors, was shown to reduce inconsistent localisation and increase accuracy of the trajectory estimation over pure odometric methods [12, 16–19]. These concepts have seen use in the context of a variety of autonomous robots and vehicles equipped with Light Detection and Ranging (LIDAR), Inertial Measurement Units (IMU) and camera systems [20–22].

vSLAM research (the use of cameras as sensory input to SLAM) has increased due to the reduction in camera prices, high information content,

[1]The pose of an object refers to the orientation and position of a robot relative to some coordinate frame

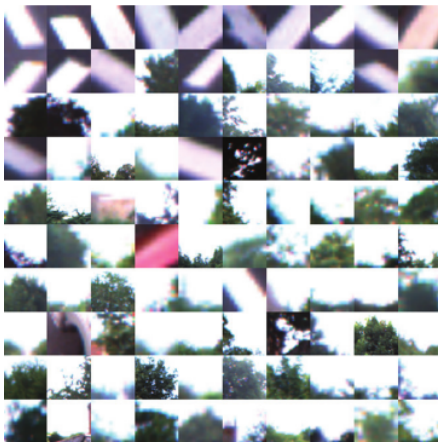


Figure 1.1: A sample collection of features used by the FABMAP [14] system in an urban environment. A single location or frame is represented by a collection of individual image patches known as a feature vocabulary. Two images taken at different points in time with similar vocabulary may indicate they were taken from the same location.

Figure 1.2: A visualisation of the iSAM2 [15] SLAM algorithm on an outdoor trajectory. The lines indicate a path traversed by the ground vehicle, while the dots alongside the path show 3D points relative to the trajectory.

and passive nature of the sensor [9,12,19,23]. vSLAM is easily adapted to fusion type algorithms that use heterogenous sensors [17,24–26], and due to its high information content, is suitable for topological or metric SLAM systems [9,27]. Thus, camera systems are of great interest for robotics due to the low cost and power consumption.

1.2 Visual Simultaneous Localisation and Mapping

A *pipeline* view of vSLAM is adopted in this thesis as described by Ros et al. [23], whereby a generic vSLAM implementation may be broken down into smaller modules as shown in Figure 1.3. While implementations differ from this configuration, most must fulfil the basic functional blocks presented. This includes: Visual Cue Acquisition (VCA) , Parameters Estimation (PE) , information management, optimisation and loop closure detection.

These functional blocks can further be split into two categories, front-end or back-end. The front-end vSLAM system can be described as all the functions required to interpret raw sensor data, and solve what is known

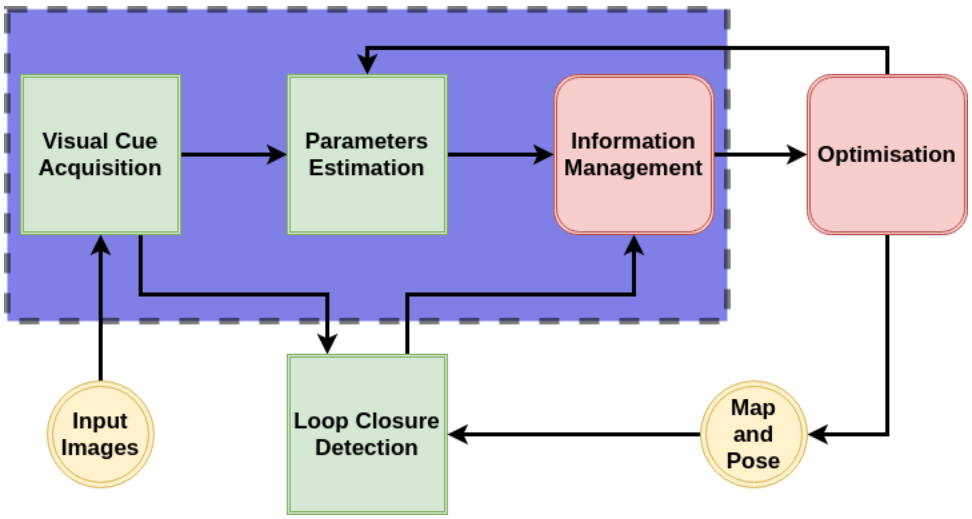


Figure 1.3: A generic visual pipeline as described by Ros et al. [23] is depicted here where each process is described by a block, and information flow by arrows. Green boxes indicate front-end processes, red back-end processes, and yellow input/output data. The blue shaded region indicates the focus of this body of work.

as the data association problem. This entails identifying what type of image patches or "features" to use, how to extract them from an image, and in some algorithms, how to track the same features over time. Because the front-end deals directly with raw images, it is also often the primary source of motion/depth estimates and loop closure events[2].

The front-end system performs visual cue acquisition, parameters initialisation and loop closure detection as indicated by the green boxes in Figure 1.3. The back-end system performs information management and optimisation, as indicated by the red boxes in Figure 1.3. A short synopsis of each subsystem, as described by Ros et al. [23] is presented below.

1.2.1 Visual Cue Acquisition

Visual Cue Acquisition describes the processing of raw stereo image data to correct for image distortion and identify features. Fuentes et al. [12] describe a feature as a visually salient image point or patch. Scaramuzza and Fraundorfer suggest typical features are discrete points and lines [28].

[2]A loop closure event is when two different frames are identified as being the same location, indicating the trajectory of the robot has formed a "loop", returning to a known position.

VCA deals both with finding these image features, as well as the manner in which they are stored or described in computer memory. The output may resemble an array of features, where only the best features are retained and passed on to the rest of the pipeline. A distinction is made between dense and sparse processes, where dense processes extract all available features from an image, while sparse implementations extract fewer, but unique features.

Due to the nature of extracting and matching features, there is often a trade-off between sensitivity and precision based on experimentally defined values [29]. High sensitivity yields many features, decreases matching precision and increases false-positive feature correspondences. Low sensitivity increases precision, but reduces the number of trackable features.

1.2.2 Parameters Estimation

Given a set of features, PE focuses on utilising these features to produce a coarse initial measurement of map parameters, however they are defined. A feature tracking framework is adopted for the purposes of this work, whereby parameters are established by measuring how unique features change over time. Fuentes et al. [12] make the distinction between a feature and a landmark, where a feature is predominantly described in terms of image characteristics, and a landmark is both the associated 3D position and uncertainty of the feature, relative to a coordinate system. A PE module may consist of a set of 3D landmarks relative to a coordinate system, and the associated robot pose or odometry.

1.2.3 Loop Closure Detection

Loop closure detection is a critical aspect of any SLAM system for reducing drift with time [30, 31]. This subsystem must recognise if a location has been revisited or not, and add this as a constraint to the information management process. Without loop closure detection, SLAM degrades into pure odometry, where incremental errors gradually cause the map and pose to become inconsistent with itself, producing conflicting or false trajectories.

1.2.4 Information Management

Information management can be considered a core task for the back-end system, where the set of constraints must be unified into one consistent mathematical framework. This process usually assumes that data-association, loop closure, and odometry have been solved or at least approximated. The task is focused around how to arrange these estimates for further refinement and storage. This may vary from retaining all landmarks and poses as in full SLAM, to keeping only the most current robot pose as in pure odometry. A SLAM system may trade-off between these two extremes, depending on the hardware and algorithms used.

1.2.5 Optimisation

Given an information management system or mathematical framework, the optimisation process attempts to estimate optimal parameters from an initial set with some associated error distribution.

Hartley and Zisserman describe Bundle Adjustment (BA) as the "gold standard" in terms of visual optimisation [32], especially if robustified against false-positives produced by other subsystems [33]. The

optimisation process may take the full set of landmarks, poses and loop closure conditions, and attempt to minimise random measurement errors. Over time, there may be thousands of parameters introduced by the pipeline and optimising all of these parameters may become prohibitively time consuming for real-time applications [34].

1.3 Related Work

We begin the review by discussing relevant contributions from pure Visual Odometry (VO) and fully integrated vSLAM systems. Kitt et al. [35] show robust stereo VO in urban scenes, with 1.3% error of the net distance travelled. They highlight how using geometric constraints between a stereo pair enabled them to reject false-positive correspondences under 20% outlier conditions.

Konolige et al. [36] implemented a visual-inertial odometer for rough terrain producing less than 0.1% error over trajectories as long as 9km at 10Hz. They utilise a feature tracking approach with CenSurE features [37], and Sparse Bundle Adjustment (SBA) [38] as the back-end system to reduce their absolute trajectory error by a factor of 2 to 5. Andrew Howard [39] showed dense stereo VO between 15 and 30 Hz, with less than 0.25% distance travelled on a legged robot Big Dog, and a wheeled vehicle in the outdoors.

Johnson et al. [21] used a feature tracking framework to implement VO for the Mars Exploration Rover Mission. They showed an average error of 2.91m over a 100m trajectory, but critically, identified cases where odometry failed. This included large inter-frame motion of approximately 1m and fewer than 25 tracked landmarks.

Warren et al. [40] performed VO in an urban environment over a 1.6km trajectory with a net displacement error of less than 3.8m. They used a feature tracking framework and sliding window BA to refine the full trajectory [41]. Using a sliding window approximates the effects of full

BA, while minimising the processing time required [42]. Pretto et al. [43] showed visual odometer estimation for a small humanoid robot walking a 10m and 20m trajectory, where standard feature tracking VO failed due to image degradation from motion blurring.

Ultimately pure VO is always limited by the rate of error introduced at each frame, leading to instability over time [44]. Even a low percentage error combined over subsequent frames, leads to "drift" from the true pose of a robot, whereas SLAM retains consistent navigation, even over long trajectories [45]. Some of the related vSLAM contributions are discussed below.

Agrawal et al. showed visual mapping over many kilometres of distance travelled by using a keyframe approach titled FrameSLAM [45]. FrameSLAM reduced computational burden by retaining only frames with high information content related to the robot pose and map, titled keyframes. Agrawal et al. [46] built on their previous odometer system in [47] by including ground plane and obstacle detection, showing up to 5% trajectory error over paths of 150m. They discuss how under certain conditions such as fast lighting changes, turns, and lack of features in the environment, VO failed completely and wheeled odometery had to be used.

Huang et al. [48] used an IMU and Red Green Blue-Depth (RGB-D) camera on an Unmanned Aerial Vehicle (UAV) for mapping and navigation in urban environments in the context of search and rescue. They displayed visually consistent mapping results and UAV control using the internal system alone.

However, higher speeds introduced failure conditions as a result of larger motion between frames, and image blur. Loop closure and the back-end systems were too slow to be run on-board the vehicle itself. A proportional controller was used to vary the front-end sensitivity such that enough features were visible with no tuning required. Schmid et al. [49] utilised a

Field Programmable Gate Array (FPGA) to process dense stereo features and IMU information for a UAV, fusing them together into an obstacle map and trajectory. A mechanical damping mechanism was used to reduce high vibrations from the rotors as they corrupted IMU data and increased motion blur.

ORB-SLAM [50] and ORB-SLAM2 [51] built a sparse feature map of the environment using a visual vocabulary[3] loop closure approach [30] and BA to optimise the estimate. It has been evaluated on multiple datasets such as [52–54], and is described as achieving less than 1% error in most cases. S-PTAM [55] utilises multiple CPU cores and a keyframe approach to build local maps as in [45]. Utilising a Pioneer-3AT mobile robot, they show an error of 0.0086m in translation and 0.18° in orientation over a 320m trajectory.

Fuentes-Pacheco et al. [12] stated that erratic camera movements, and environments with too few/many salient features are two of the common causes of vSLAM failure. These two factors are strongly influenced by the type of environment, robot hardware, vSLAM pipeline, and the context in which these three systems interact. Figure 1.4 gives a highly simplified visualisation of how these different systems may affect each other and what type of factors must be considered. Each factor has an influence on the other two, while the combination of these interactions contributes to the effective solution and performance characteristics.

Interactions between the environment and the robot hardware may include erratic disturbances introduced by rough terrain, particularly in small-scale robots with minimal suspension systems.

These disturbances are a complex function of the terrain, wheels, size, mass, velocity, and acceleration experienced by the robot [56–59]. A small-scale car with minor suspension systems, light mass and travelling at

[3] A set of predefined, learned, or previously seen visual descriptors

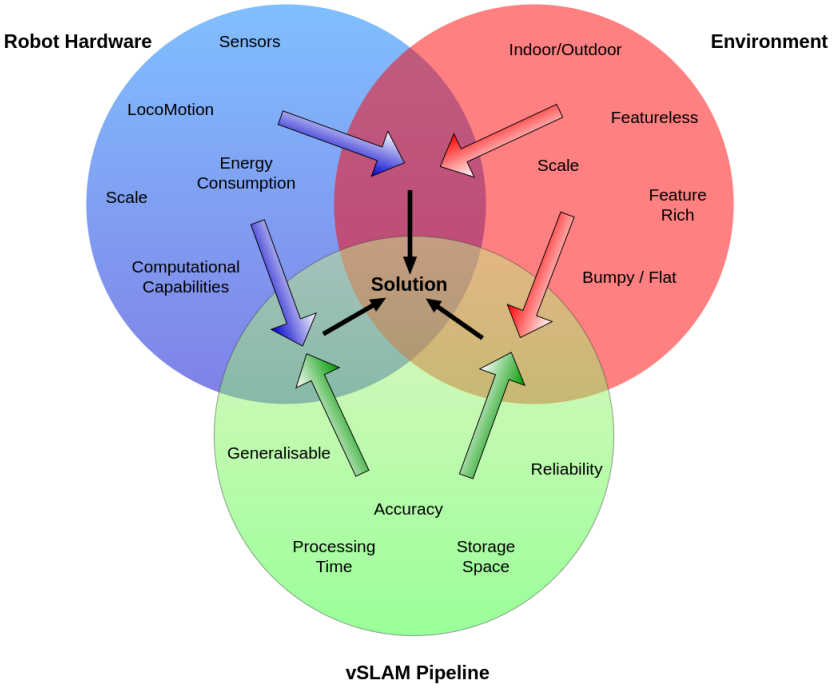


Figure 1.4: A simple visualisation of the different factors affecting a vSLAM pipeline. Robot Hardware refers to the physical hardware used including the sensors, processors and actuation. Environment refers to the environment in which the robot is expected to operate, and pipeline refers to the algorithm employed to solve vSLAM

slower speeds over rough terrain, may experience significantly more relative disturbances than a full scale car optimised for riding comfort on paved roads. These effects can introduce motion blurring and are exacerbated by lower frame rate speeds and fluctuating light conditions, where the camera sensor itself has a trade-off between light, frame rate, motion blur and image noise. Handa et al. [60] simulated some of these conditions from a purely sensor point of view, stating that "using a realistic camera model, there is an optimal frame rate for given lighting levels due to the trade-off between SNR (Signal to Noise Ratio) and motion blur".

The effective robot payload[4] is heavily influenced by the robot scale, and as such, places constraints on the vSLAM pipeline under real-time conditions. Studies such as [48] and [49] required optimised vSLAM algorithms if they were to use the on-board vehicle processor as scale constrained the processing power available. Scale also reduces the size of the effective

[4]Payload in this context means all the hardware a robot is capable of carrying whilst in operation. This includes the sensors, processors, batteries, storage space, and actuators.

baseline[5] between stereo camera pairs, reducing the accuracy to which distant landmarks can be measured [61].

Lastly, the sensor and the effects environment and motion have on the camera sensor, impact pipeline requirements. A camera that undergoes image distortion as a result of low frame rate, high robot velocity, rough terrain, or lighting changes will require robust processes or knowledge about the operating environment in order to be effective.

A camera with a wide field of view[6] (FOV) may be able to track many more features than a narrow FOV, while image texture and lighting in some environments limit front-end sensitivity. Pretto et al. [43] used a deblurring process to help remove image distortion as a result of robot motion, while geometric verification filtered as many poor correspondences as possible from the VCA and PE modules [21, 35, 39]. Kitt et al. [35] made an assumption about planar motion between frames to increase correspondence matching speed, while Warren et al. [40] calibrated the camera sensor to ensure that minimal motion blurring occurs.

Context of operation is thus an important factor to account for in the generic pipeline of Figure 1.3, particularly for small-scale/footprint robots which undergo large relative disturbances, have small FOV, and minimal processing power available.

General pipeline implementations as shown in Figure 1.3 may be a useful tool for understanding the functional requirements of a vSLAM system, but cannot represent the specifics required of a working solution. A vSLAM implementation must balance conflicting demands from the pipeline, hardware, and environment.

However, to find a trade-off appropriate for the context, identifying the

[5]The ideal displacement between two cameras

[6]Refers to the area in front of the camera which can actually be imaged and is dictated by the lens and sensor size.

most influential factors is crucial. Careful selection of the different processes is required from a design perspective to match the context of operation.

Datasets such as [52–54, 62–65] are effective for benchmarking and designing pipelines, as they allow for objective comparison by constraining the hardware and environment to a single type. The pipeline can thus be studied alone, and improvements attributed to the pipeline.

However, this methodology is only effective if the hardware and environmental conditions match that of an intended application. Finding an appropriate dataset for this purpose can be challenging, especially in the case of designs with minimal processing power, FOV, Frames Per Second (FPS), suspension and scale.

Choosing a pipeline methodology has the same challenges, as it is not clear which factors and interactions play a role or will even be present in the system. It is clear from the literature that there are many dozens of variations available for vSLAM, and in this body of work a front-end pipeline is studied in the context of a small-scale wheeled robot operating in an outdoor environment.

1.4 Problem Statement

General pipeline implementations as shown in Figure 1.3 are a useful tool for understanding the functional requirements of a vSLAM system. Under ideal conditions, vSLAM pipelines would be fully transferable from one platform to another with minimal changes. The interaction between the hardware, environment, and pipeline would be generic and tunable for the application. Under these conditions, the trade-offs between accuracy, speed of processing and robustness would be clearly defined.

However, with the role context plays in performance, it is not clear from the presented literature what pipeline is most suitable for the case of a

small-scale, outdoor, wheeled robot. In this regard, the solution space of a vSLAM system is uncertain. It is clear that robustness is of key concern for visual algorithms in outdoor conditions, but it is unclear which factors will be present as a result of environment and hardware, and what processes may be used to mitigate these effects.

1.5 Aims and Research Questions

The aim of this study is to evaluate the performance of a vSLAM front-end system for use on a small-scale robot in an outdoor environment. In particular, this study aims to perform the following functional requirements.

1. Evaluation and study of a front-end feature tracking framework. This framework will be made up of a Visual Cue Acquisition, Parameter Estimation, and simple Information Management module as shown by the blue bounding box of Figure 1.3.
2. Critically identify trade-offs of the proposed front-end
3. Evaluate the effectiveness of the front-end in the context of a small-scale experimental platform

1.6 Scope and Limitations

1. **Local Map Assumption**: A small local map is required. This local map can naturally be included in any submapping approach, or large-scale integration, but it is not addressed in this document
2. **Static World Assumption**: The world environment is completely static and does not change with time.
3. **Salient Feature Assumption**: The operating environment has enough salient features before any geometric distortions as a result of motion or camera limitations. This effectively eliminates harsh environments such as snow or deserts, where unique features are very rare, similar, or have little discernible texture.

4. **Motion Distortion Assumption**: The dominant sources of noise and image deformation are as a result of motion or low frame rate. Thus, it is assumed that there will be no specular objects in the environment, and weather/lighting does not play as large a role as motion blur. This helps constrain the problem to address the challenges associated with the platform and ground interactions.
5. **Camera Dynamics Assumption**: The camera is capable of capturing the dynamics of the motion. Motion is not so fast that features appear for only one frame, or are untrackable.
6. **Singular Sensor:** The camera setup is the only available source of information. GPS ground truth is unavailable.
7. **Odometer:** Loop closure events may be present in the dataset, but no loop closure detection is performed with the features selected.

1.7 Research Approach

The research takes an offline and modular approach to studying the front-end system. A dataset of an outdoor sequence is recorded using a robot. VCA, PE, and information management modules are described and tested with a combination of real and simulated data.

A calibration stage is performed to quantify the sensor characteristics, followed by a simulation benchmarking the PE module. The VCA stage is evaluated on real data, before both stages are tested together on the collected dataset sequence.

1.8 Outline of the Report

Literature Review

In this chapter, implementation details of generic front-end feature tracking frameworks are outlined. This includes the camera model, information management description, relative pose estimation and feature paradigms that are used extensively throughout the pipeline.

Methods

In this chapter, the front-end pipeline is described in detail. Sensor calibration and dataset collection methodology are described together with the intended research platform used in the study to provide context to the work. The formulation of the front-end pipeline is described for the inner workings of the VCA and PE modules. Lastly, the simulation methodology is described.

Results

The results are presented in this chapter for the study, addressing Item 1 of the research questions. This consists of a dataset and sensor calibration, feature tracking simulation, PE and VCA module study. Lastly the combined pipeline results on a dataset sequence are shown in the form of a reconstructed trajectory.

Discussion

In this chapter, the main concerns of the study are addressed. The results are discussed with reference to the problem statement whereby Items 2 and 3 will be answered from the research questions.

Conclusion

A summary of the study is provided here together with the main results, and key points from the discussion. Recommendations and potential future work are discussed.

Chapter 2

Literature Review

In Section 1.2, a generic pipeline view of vSLAM was introduced along with some of the functional requirements that may be expected from each subprocess. In this chapter, the implementation details of these subprocesses are described beginning from the Information Management process and working backwards to the VCA stage.

Coordinate system conventions and rigid body transformations are first described as they are a base building block upon which a pose graph interpretation to vSLAM is described. A common mathematical framework used to model monocular and stereo cameras is reviewed, followed by literature on pose estimation in this framework. This helps contextualise and define the expected output, interpretation and conventions used for vSLAM. The existing literature on processing raw image data is then discussed in the form of feature detection and its interpretation in a pose graph.

2.1 Coordinate System Definitions

For robotic systems where movement in 3D space is concerned, measurements such as robot pose, motion, and landmark position are poorly defined without a clearly defined coordinate frame. A coordinate frame in the context of this work follows a right handed **SE3** Euclidian

space convention which consists of an origin $\mathcal{O}$, and a set of 3×1 orthonormal basis vectors $\mathcal{V}$ or axes. We define all coordinate frames relative to a world coordinate frame $\mathcal{W}$.

Any point within $\mathcal{W}$ can be described in terms of different coordinate frames, so long as the relationship between them is defined. The relationship between between $\mathcal{W}$ and a different coordinate frame is defined by a rigid body transformation. This transformation is made up of a 3×3 rotation matrix $\boldsymbol{R}$, and a 3×1 translation column vector $\boldsymbol{C}$. Figure 2.1 depicts a typical rigid body transform between two coordinate systems.

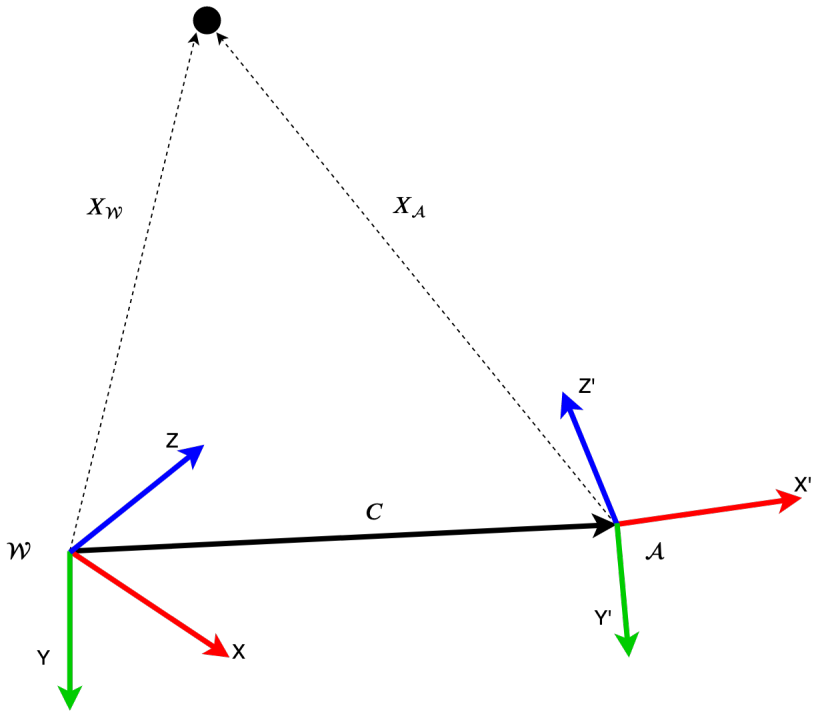


Figure 2.1: A depiction of a rigid body transform between coordinate systems $\mathcal{W}$ and $\mathcal{A}$. The black circle represents a point/landmark visible from both coordinate frame $\mathcal{A}$ and $\mathcal{W}$.

Let a 3×1 vector $X_{\mathcal{W}}$ represent a 3D cartesian coordinate/point relative to frame $\mathcal{W}$. If $X_{\mathcal{A}}$ represents the same point relative to frame $\mathcal{A}$, a 4×4 matrix $\mathcal{T}_{\mathcal{W}}^{\mathcal{A}}$ mapping coordinate systems $\mathcal{W}$ to $\mathcal{A}$ is defined by $\boldsymbol{R}$ and $\boldsymbol{C}$. If each point is rather represented by a 4×1 homogeneous vector denoted by tilde, then this relationship is described by Equation 2.1.

In some texts, the translation component $\boldsymbol{T}=-\boldsymbol{RC}$ is used, whereas $\boldsymbol{C}$ refers to the offset between coordinate frame origins [32]. The inverse of this process is also true such that $X_{\mathcal{W}}$ can be defined in terms of frame $X_{\mathcal{A}}$ where the rigid body transform is $\mathcal{T}_{\mathcal{A}}^{\mathcal{W}} = (\mathcal{T}_{\mathcal{W}}^{\mathcal{A}})^{-1}$.

$$\begin{aligned} X_{\mathcal{A}} &= R(X_{\mathcal{W}} - C) \\ X_{\mathcal{A}} &= RX_{\mathcal{W}} - RC \\ X_{\mathcal{A}} &= RX_{\mathcal{W}} + T \\ \tilde{X}_{\mathcal{A}} &= \mathcal{T}_{\mathcal{W}}^{\mathcal{A}} \tilde{X}_{\mathcal{W}} \\ (\mathcal{T}_{\mathcal{W}}^{\mathcal{A}})^{-1} \tilde{X}_{\mathcal{A}} &= \tilde{X}_{\mathcal{W}} \end{aligned} \tag{2.1}$$

A robot pose, or kinematic chain[1] can be described using these coordinate frame transforms. We define three types of coordinate frames: the world coordinate system or inertial frame $\mathcal{W}$ described earlier, robot centre of mass frame $\mathcal{CM}$, and sensor frame $\mathcal{S}$ aligned with that of a particular camera. Figure 2.2 gives a simple example of these coordinate frames with only a single camera sensor frame defined.

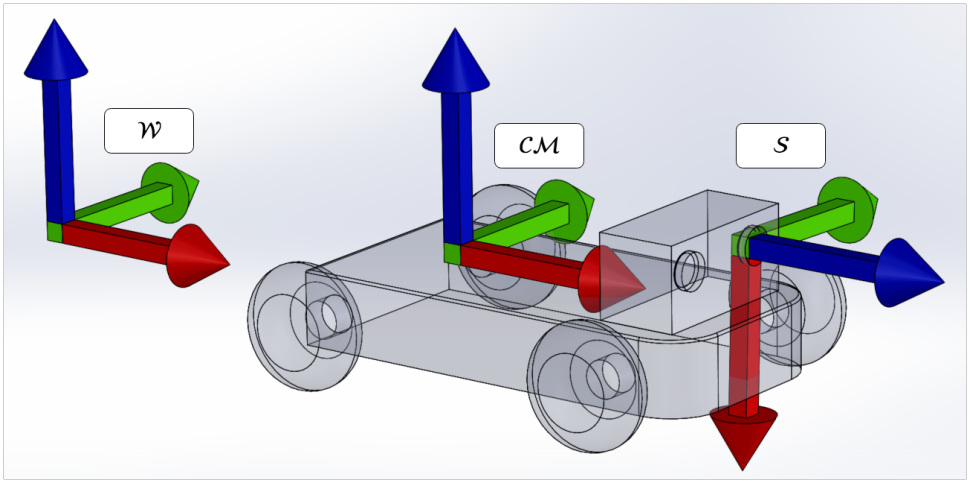


Figure 2.2: An illustration of a simple coordinate system with three frames, the world, centre of mass, and sensor frame. Red arrows indicate x, green indicate y, and blue indicate z axes respectively within each coordinate frame.

[1] A set of sequential coordinate frame transforms. A kinematic chain can be used where multiple joints, sensors, and robots are present

2.2 Graph SLAM

Given a set of measurements about the environment in terms of landmarks and robot poses within an inertial frame, the back-end system is responsible for assembling and maintaining them in one unified mathematical framework. A convenient framework for this process is the graph SLAM interpretation to mapping [66]. Each pose/keyframe and landmark can be modelled as a node on a graph, while sensor measurements are represented by edges between different nodes.

Optimisation of a pose graph interpretation can also be directly interpreted as minimising the error associated with each edge. Packages such as G2o [67], SRBA [68], and iSAM2 [15] are all implementations of graph optimisers that minimise an arbitrary cost function. The graph interpretation to SLAM makes the problem generic and independent of the sensor or pose parameterisation, as any measurement or state can be encoded as an edge or node in this graph so long as the coordinate system conventions are consistently adhered to. Changes in coordinate frame can also be easily incorporated in this SLAM interpretation by adding additional nodes into the graph network. This makes a graph interpretation a useful, generic manner in which an Information Management system can be built, and expanded with multiple heterogenous sensors.

A full vSLAM pose graph is depicted graphically in Figure 2.3, where the circles represent robot poses, triangles represent landmarks, green arrows indicate measurements about poses, and black lines indicate measurements about landmarks. Note that it implicitly assumes that landmark correspondence has been established prior to the graph formulation. A pure odometer system could be described as using only the robot pose nodes and green odometer edges. KF_0 has multiple odometer edges connected and portrays a loop closure event.

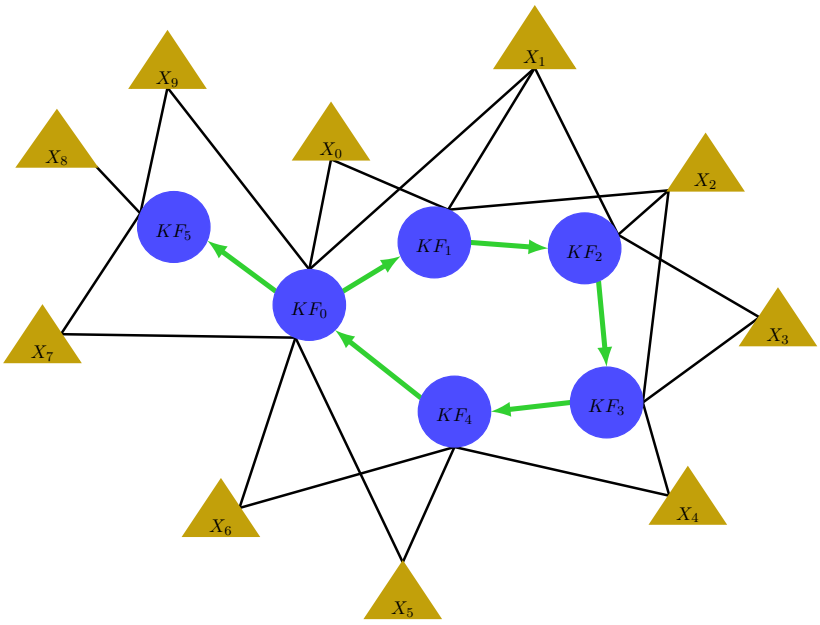


Figure 2.3: A depiction of full vSLAM from a graph perspective. Each node KF_k represents a robot pose, green arrows indicate rigid body transforms between poses, gold triangles depict unique landmarks, and black lines indicate the distance to a landmark as seen from a particular node.

The notation of [68] is used throughout this work, where $KF_{\mathcal{A}}$ is a pose node or keyframe at time $\mathcal{A}$, $K2K^{\mathcal{B}}_{\mathcal{A}}$ is an edge from coordinate system $\mathcal{A}$ to $\mathcal{B}$ encoded as a rigid body transform $\mathcal{T}^{\mathcal{B}}_{\mathcal{A}}$, LM_i represents a landmark node with unique identifier i, and $z^{i,j}_k$ represents an observation edge of a landmark i, first seen in base pose j, at pose k. $X^{i,j}_k$ represents the 3D position of a landmark i, with base pose j, from pose k. For brevity, the base and current pose may be assumed to be the same such that $k = j$ if the superscript j is not explicitly stated.

2.3 Pinhole Camera model

A mathematical model of the ideal camera sensor is used as the basis for many geometric verification schemes as discussed in [36, 43, 69, 70], and depth estimation [61, 71, 72]. An ideal model of a camera system is known as the pinhole camera model [32]. Given a landmark LM_i, the pinhole camera model can be used to predict the location of LM_i on the camera image plane, and can be considered a mapping from the 3D coordinates of $X^i_{\mathcal{S}}$ or $X^i_{\mathcal{W}}$, to pixel location $z^i_{\mathcal{S}}$. The image plane origin is centred around the top left image corner as this naturally transfers to computers where

negative indexes are uncommon.

This is depicted in Figure 2.4 in terms of the $\mathcal{W}$ coordinate frame, and camera centric frame $\mathcal{S}$. Within frame $\mathcal{S}$, the positive z-axis conventionally points forward out the image plane, the positive x-axis points right along the image plane, and the positive y-axis points down along the image plane. The black arrows along the image plane indicate the image origin, while the long arrow from $\mathcal{W}$ to $\mathcal{S}$ depicts the rigid body transform $K2K^{\mathcal{S}}_{\mathcal{W}}$.

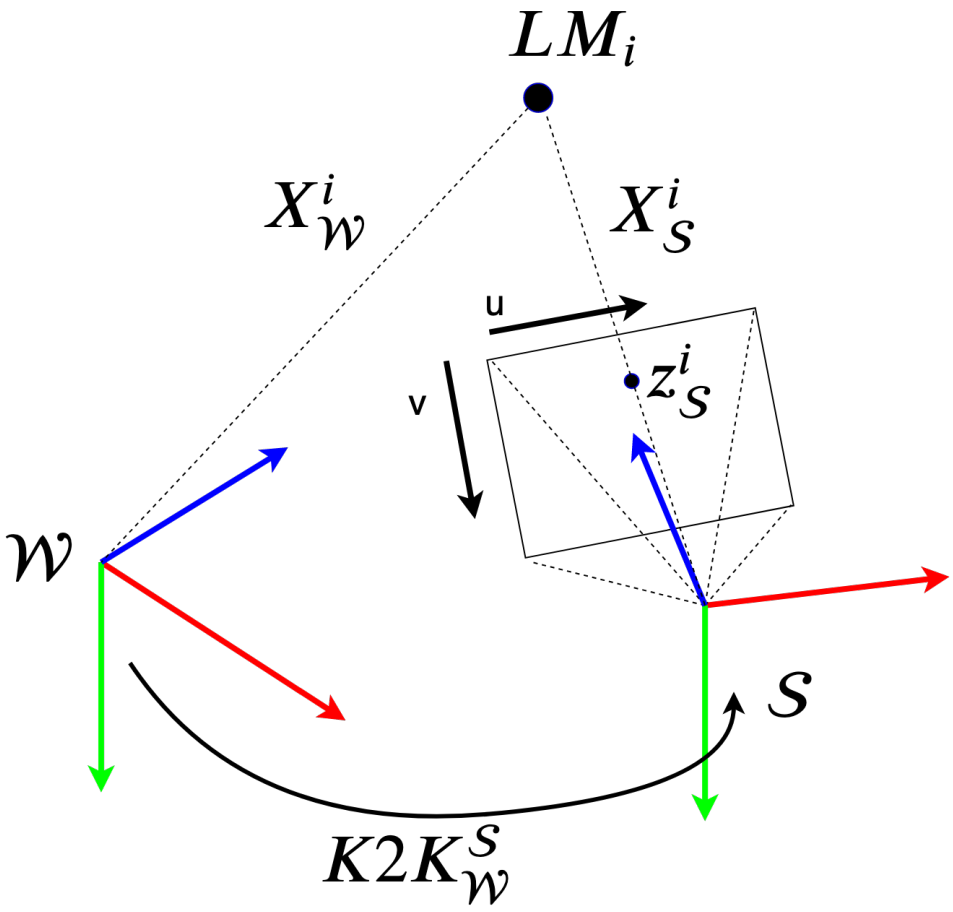


Figure 2.4: A depiction of the pinhole camera model coordinate system. The black dot depicts the landmark LM_i, while $z^i_{\mathcal{S}}$ shows the projection of the landmark onto the image plane. $K2K^{\mathcal{S}}_{\mathcal{W}}$ shows the transform relationship between coordinate frames.

If it is assumed that the 3D location of LM_i is located relative to $\mathcal{S}$, the mapping from LM_i to $z^i_{\mathcal{S}}$ can be conveniently described using homogeneous coordinates. Let $\tilde{\mathbf{X}}^i_{\mathcal{W}}$ be a 4x1 homogeneous vector of LM_i, the predicted pixel position $z^i_{\mathcal{S}}$ is dictated by two properties, the intrinsic and extrinsic camera matrices.

The 3x3 intrinsic matrix $\boldsymbol{K}$ represents physical parameters of a camera such as the effective[2] focal length, centre of projection, and pixel size, whereas the extrinsic matrix represents the edge $K2K^{\mathcal{S}}_{\mathcal{W}}$ from LM_i into the sensor frame $\mathcal{S}$. It is assumed that $\mathcal{S}$ is the centre of projection for the camera.

Under these conditions, the full mapping procedure from $\tilde{\boldsymbol{X}}^{i}_{\mathcal{W}}$ to $z^i_{\mathcal{S}}$ can be conveniently described in terms of a projection function $P(\boldsymbol{K}, K2K^{\mathcal{S}}_{\mathcal{W}}, \tilde{\boldsymbol{X}}^{i}_{\mathcal{W}})$ defined in Equation 2.2. $\boldsymbol{R}$ can be composed with any parameterisation such as Euler or quaternion angles as needed, so long as it is consistently applied throughout. Applications such as drones where full 360° rotation is common must parameterise $\boldsymbol{R}$ such that degenerate configurations can be avoided [73].

$$
\begin{aligned}
\tilde{z}^i_{\mathcal{S}} &= P(\boldsymbol{K}, K2K^{\mathcal{S}}_{\mathcal{W}}, \tilde{\boldsymbol{X}}^{i}_{\mathcal{W}}) \\
\tilde{z}^i_{\mathcal{S}} &= \left[\begin{array}{c|c} \boldsymbol{K} & \mathbf{0} \end{array}\right] K2K^{\mathcal{S}}_{\mathcal{W}} \tilde{\boldsymbol{X}}^{i}_{\mathcal{W}} \\
\begin{bmatrix} u^i_{\mathcal{S}} \\ v^i_{\mathcal{S}} \\ w^i_{\mathcal{S}} \end{bmatrix} &= \left[\begin{array}{ccc|c} f & 0 & cx & 0 \\ 0 & f & cy & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a^i_k \\ b^i_k \\ c^i_k \\ 1 \end{bmatrix} \\
\begin{bmatrix} u^i_{\mathcal{S}}/w^i_{\mathcal{S}} \\ v^i_{\mathcal{S}}/w^i_{\mathcal{S}} \\ 1 \end{bmatrix} &= \left[\begin{array}{ccc|c} f & 0 & cx & 0 \\ 0 & f & cy & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a^i_k \\ b^i_k \\ c^i_k \\ 1 \end{bmatrix}
\end{aligned}
\tag{2.2}
$$

The final expression in Equation 2.2 shows the conversion from a homogeneous representation $\tilde{z}^i_{\mathcal{S}}$ to image plane coordinates $z^i_{\mathcal{S}}$ where the last element containing 1 is removed, going from a 3x1 to a 2x1 matrix. The inverse mapping from $z^i_{\mathcal{S}}$ to $X^i_{\mathcal{S}}$ is possible by inverting the matrices, however, it is correct only up to scale factor η. In the single camera case, without additional information, it is only possible to construct $\eta\tilde{X}^i_{\mathcal{S}}$ as any

[2]We specify "effective" as these are pseudo parameters, measured in terms of pixels as opposed to the true focal length that may specified in mm

of these projected landmarks return the same $z^i_{\mathcal{S}}$ as a result of homogeneous coordinates.

In addition to the intrinsic and extrinsic parameters, non-linearities affect the validity of the pinhole camera model. Common non-linearities include radial and tangential distortion introduced by camera lenses [74, 75]. These are typically corrected for with a secondary mapping function that undistorts pixels. This undistortion or rectification process ensures that straight lines in space such as walls appear straight on the image and unwarped. A vector of four, five, or eight distortion coefficients $\boldsymbol{D}$ can be associated with a camera that describes the remapping function.

The intrinsic, extrinsic, and distortion coefficients are not often provided from camera manufacturers, and as such need to be estimated from calibration techniques such as those discussed in [74, 75]. These exploit known geometry about a checkerboard pattern to infer camera parameters given a set of checkerboard images taken from multiple vantage points. Thus, given the intrinsic, extrinsic, and distortion coefficients estimated from calibration, the projection $z^i_{\mathcal{S}}$ of a landmark LM_i can be calculated by first undistorting the image, and then using Equation 2.2 with the appropriate $K2K^{\mathcal{S}}_{\mathcal{W}}$. Note that if all landmarks can be described within frame $\mathcal{S}$, $K2K^{\mathcal{S}}_{\mathcal{W}}$ can be assumed to be identity, and the projection function depends only on the landmark and intrinsic matrices.

2.4 Stereo Camera Model

Extending the pinhole camera concept to two calibrated cameras, left and right, it is possible to define what is known as an epipolar line. Let $\mathcal{S}_l$ and $\mathcal{S}_r$ denote the left and right sensor coordinate frame respectively with a known edge $K2K^{\mathcal{S}_r}_{\mathcal{S}_l}$. Let the projection functions be $P_l = [\boldsymbol{K}_l|0]$ and $P_r = [\boldsymbol{K}_r|0]$ within their respective camera coordinate frame. If the assumption is made that the left sensor coordinate frame aligns with the world coordinate frame such that $\mathcal{W} = \mathcal{S}_l$, it is possible to rewrite $P_r = [\boldsymbol{K}_r|0]K2K^{\mathcal{S}_r}_{\mathcal{S}_l} = [\boldsymbol{K}_r\boldsymbol{R}|\boldsymbol{T}]$.

If the projection functions P_l and P_r are known with $\mathcal{W} = \mathcal{S}_l$, it is possible to predict the pixel positions $z^i_{\mathcal{S}_l}$ and $z^i_{\mathcal{S}_r}$ from a landmark $X^i_{\mathcal{S}_l}$. Additionally, if an image projection is known such as $z^i_{\mathcal{S}_l}$, it is possible to constrain the location of $z^i_{\mathcal{S}_r}$ to a line $L^i_{z_{\mathcal{S}_l}}$ on the right image and vice versa [32].

$L^i_{z_{\mathcal{S}_l}}$ is known as an epipolar line, and indicates where a corresponding image projection $z^i_{\mathcal{S}_r}$ may be found. Figure 2.5 illustrates this concept in the ideal case where the epipolar lines are drawn in orange and cyan for the left and right images respectively. This constraint can be quantified by Equation 2.3 with a 3x3 matrix E known as the essential matrix dependent only on P_l and P_r. $[T]_\times$ denotes the cross product form of $\boldsymbol{T}$ and $\underline{z}^i_{\mathcal{S}_l} = \boldsymbol{K}_l^{-1} z^i_{\mathcal{S}_l}$, the normalised pixel coordinate.

$$\begin{aligned} (\underline{z}^i_{\mathcal{S}_l})^{\boldsymbol{T}} E(\underline{z}^i_{\mathcal{S}_r}) = 0 \\ (\underline{z}^i_{\mathcal{S}_l})^{\boldsymbol{T}} [T]_\times R(\underline{z}^i_{\mathcal{S}_r}) = 0 \end{aligned} \tag{2.3}$$

The implication of Equation 2.3 is that two tentative correspondences across stereo frames with known extrinsics can define an error score based on geometric properties. Thus, minimising Equation 2.3 is one manner of scoring the legitimacy of a correspondence. Figure 2.5 depicts another error score where the orthogonal distance to an epipolar line could indicate whether the pink features are potential correspondences. Ideal correspondences will fall exactly along the epipolar line, while larger errors indicate a feature is less likely to be a valid correspondence.

However, to define and evaluate this error can be cumbersome for digital images, as the epipolar lines are typically non-horizontal, and do not necessarily span the full image. Depending on $K2K^{\mathcal{S}_r}_{\mathcal{S}_l}$, a feature visible in one camera may not necessarily be visible in the other due to the finite image size.

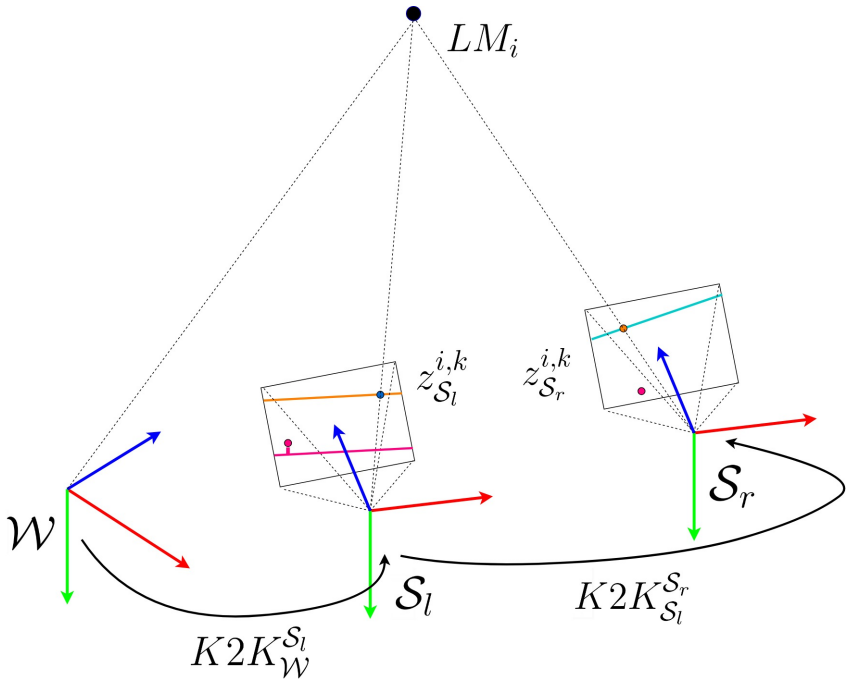


Figure 2.5: An illustration of the pinhole camera model applied to stereo cameras. Three frames are shown, the world frame $\mathcal{W}$, and the left and right camera sensor frames. The orange line on the left image plane depicts the epipolar line for $z^i_{\mathcal{S}_r}$, and the blue for $z^i_{\mathcal{S}_l}$. Note that in the ideal case these features will fall exactly along these lines. In the non-ideal case depicted with pink colours, a feature may not fall exactly on the epipolar line, and if the correspondence is invalid, will fall further away from the epipolar line.

One approach to simplifying this error calculation has been the stereo rectification process [76, 77]. Stereo rectification corrects the intrinsic camera distortion found on each camera, and enforces horizontal epipolar lines between correspondences. This aids in algorithm simplicity, as each epipolar line is horizontal, closely follows the ideal pinhole model in Equation 2.2, and defines valid Regions of Interest (ROI) on each image where the FOV of each camera overlaps sufficiently to establish a correspondence.

Mathematically, this rectification process is equivalent to defining new ideal projective cameras $P_{\mathcal{R}l}$ and $P_{\mathcal{R}r}$ in a rectified coordinate system $\mathcal{R}$. Within this rectified coordinate frame, $P_{\mathcal{R}l}$ and $P_{\mathcal{R}r}$ have parallel image planes, and are separated only in translation. Thus $K2K^{\mathcal{R}_r}_{\mathcal{R}_l}$ has an identity rotation matrix $\mathbf{R}$, and only a horizontal distance component between centres of projection known as the stereo baseline $B = C$.

Projections $z^i_{\mathcal{R}_l}$ and $z^{i2}_{\mathcal{R}_r}$ additionally have the same vertical component and differ only by a disparity ${}^{i2}_{i}\triangle d_{\mathcal{R}_l}$ in the u component. Under these ideal rectified assumptions, an intuitive geometric error score emerges as the vertical discrepancy between two corresponding features. We define this discrepancy between ${}^{i2}_{i}\triangle e_{\mathcal{R}_l} = |v^i_{\mathcal{R}_l} - v^{i2}_{\mathcal{R}_r}|$ as the epipolar error between two correspondences. Another popular geometric error score is known as the reprojective error, whereby the discrepancy between correspondences is measured as the square root distance between a predicted pixel position, and the measured one [32].

Triangulation between corresponding features can be performed with the Direct Linear Transform (DLT) [32]. Alternatively in the case of rectified coordinates, a faster reprojective triangulation scheme can be used based on a reprojection matrix Q, and the stereo projections $z^i_{\mathcal{R}_l}$ and $z^{i2}_{\mathcal{R}_r}$. This is shown in Equation 2.4, where the triangulated coordinate $\tilde{\boldsymbol{X}}^{\,i}_{\mathcal{R}_l}$ can be calculated with a simple matrix calculation.

$$\tilde{\boldsymbol{X}}^{\,i}_{\mathcal{R}_l} = Q \cdot \begin{bmatrix} u^i_{\mathcal{R}_l} \\ v^i_{\mathcal{R}_l} \\ {}^{i2}_{i}\triangle d_{\mathcal{R}_l} \\ 1 \end{bmatrix}$$

$$\tilde{\boldsymbol{X}}^{\,i}_{\mathcal{R}_l} = \begin{bmatrix} 1 & 0 & 0 & -cx_y \\ 0 & 1 & 0 & -cy_y \\ 0 & 0 & 1 & f_l \\ 0 & 0 & \frac{-1}{B} & 0 \end{bmatrix} \cdot \begin{bmatrix} u^i_{\mathcal{R}_l} \\ v^i_{\mathcal{R}_l} \\ {}^{i2}_{i}\triangle d_{\mathcal{R}_l} \\ 1 \end{bmatrix} \tag{2.4}$$

A typical pose chain of a full stereo camera system is depicted in Figure 2.6 which includes the effects of the stereo rectification process, and the robot centre of mass cm. For a control system formulated at the centre of mass KF_{cm} of the robot, the observation measurements would require an additional transform or alternatively constrain the kinematic model to the observations respective coordinate systems.

However, a simplifying assumption can be made whereby the full output is specified in the rectified coordinate frame. Under this assumption, $K2K_w^{\mathcal{R}_l}$ can be set to the identity homography, and the stereo coordinate system can be simplified to only the nodes $KF_w, KF_{\mathcal{R}_l}$ and $KF_{\mathcal{R}_r}$.

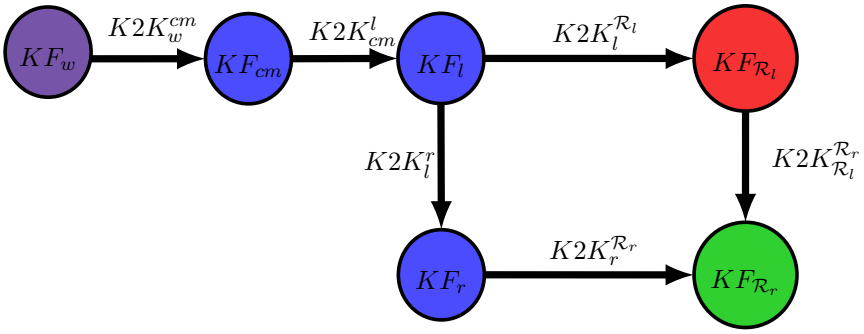


Figure 2.6: A pose graph of a generic stereo camera following rectification relative to an arbitrary world coordinate system. The frames KF_l and KF_r represent the position of cameras in space relative to the centre of mass. $KF_{\mathcal{R}_l}$ (red) and $KF_{\mathcal{R}_r}$ (green) represent the coordinate system in which the ideal projective cameras $P_{\mathcal{R}_l}$ and $P_{\mathcal{R}_r}$ are valid. The edges $K2K_l^{\mathcal{R}_l}$ and $K2K_r^{\mathcal{R}_r}$ are rectification transformations which mathematically rotate each camera such that $K2K_{\mathcal{R}_l}^{\mathcal{R}_r}$ only contains a horizontal translation component B

2.5 Relative Pose Estimation

In the preceding section, a camera sensor model was described using the pinhole model. Through the use of calibration techniques, it is possible to estimate both the intrinsic and extrinsic parameters of a stereo camera. However, in the case of VO and vSLAM, the relative pose edge $K2K_k^{k+1}$ between consecutive frames may not be known beforehand.

If landmark observations are known, it is possible to invert the logic of Equation 2.3 and instead solve for the Essential matrix from known correspondences. Popular solutions from the field of Structure from Motion (SFM) include the Five [78] and Eight-point [79] algorithms.

However, these solutions have been shown to display extreme sensitivity to noisy and false-positive correspondences. Additionally, an important property of the Essential matrix solution is the fact that reconstruction is only to scale, meaning that without additional information, relative pose information is not metric. Visual-inertial systems use supplementary

sensors to estimate this unknown scale, while other algorithms such as those given in [80] solve for scale explicitly with multiple cameras.

Other approaches to estimating relative pose work directly with 3D triangulated points, such as the Iterative Closest Point (ICP) [81], or Singular Value Decomposition (SVD) based algorithm [82]. These algorithms estimate the rigid body transform between 3D points by minimising the geometric distance between corresponding points. Because they work directly with a known unit such as meters, these algorithms produce metric information, but also suffer from the same sensitivity to noise and false correspondences.

Random Sample and Consensus (RANSAC) has been one popular method of robustifying relative pose estimation to false correspondences and is used both in validating correspondences and estimating motion [35, 69, 83, 84].

In the context of relative pose estimation, RANSAC selects a minimum number of samples from a set of potential correspondences, estimates the relative pose according to the selected samples, and tests how accurately it fits the data. This fit can be measured with any appropriate error score such as the reprojective/epipolar error between landmark projections, or absolute distance between 3D points.

However, under large outlier conditions and correspondences, the processing time grows substantially [85]. It can therefore become a potential bottleneck in cases where correspondences are intrinsically hard to establish and the correspondence inlier/outlier ratio is low.

Additional constraints to pose estimation can also be included to increase robustness, processing speed and accuracy. Paz et al. [86] split their landmarks into near and far distances using a threshold value. They describe stereo vision degrading into a bearing sensor for landmarks further than 5m with a 120mm baseline camera. Separate monocular and stereo

edge parameterisations were used to describe this, where monocular edges modelled distant landmarks and stereo edges close landmarks. The split parameterisation improved translation and orientation accuracy by using the most appropriate features for pose estimation.

Cvisic and Petrovic [84] decouple their orientation and translation estimation, first solving for orientation, and using this to aid in accuracy and reliability of the translation estimate. Kitt et al. [35] use a planar motion assumption in their RANSAC scheme to minimise the number of samples required, decreasing processing time, and stereo constraints to reject outlier correspondences.

Thus, given a set of correspondences or triangulated features at two different keyframes, the relative pose can be computed and stored as an edge between keyframes. The accuracy of the estimate is completely dependent on the algorithm, robustness choices, and landmark noise characteristics.

2.6 Feature Detection and Description

In the preceding sections, the mathematical tools used to formulate vSLAM as a pose graph were presented. Odometer and vSLAM methods are built on these fundamental principles to describe the sensor model, and convert image measurements into motion, 3D points and loop closure events.

However, this description exists purely as a mathematical formulation. Without a mechanism to determine pixel measurements and track them through time, nodes and edges cannot be formulated into the graph structure. Additionally, if a collection of nodes and edges defines the state of a vSLAM system, then clearly incorrectly formulated edges will have an influence on the accuracy and reliability of the solution. Therefore, it is important to establish how these edges and vertices are measured from raw image data, and translated into edges and nodes.

In order to fulfil this function, three questions need to be sufficiently answered. How can feature pixel locations be determined that are visually distinctive? How can the image at this location be described, extracted, and retained in computer memory? Lastly, how can we establish correspondence or evaluate how close two features look?

A popular paradigm for answering these three questions has been the interest point detector-descriptor pair approach [87]. The detector-descriptor paradigm answers each of the three questions by defining a detector, extractor, and scoring process. Some of the famous detector-descriptors pairs used in the literature are SIFT [88], SURF [89], HARRIS [90], FAST [91], AKAZE [92], ORB [93], BRISK [94], BRIEF [95] and FREAK [96], ASIFT [97], CENSURE [37], and SUSAN [98]. These detector-descriptor pairs have been well studied in the image processing community with many benchmarks comparing their effectiveness under different image transformations such as scale, rotation, and lighting [87, 99–106].

Typical evaluation criteria include the number of features tracked, precision-recall curves, and processing speed. Aanaes et al. [106] show how change in lighting, viewpoint, and scale affect detector-descriptor pairs differently. They used known viewpoint changes to illustrate a reduction in recall performance for a variety of detector-descriptors as a function of camera distance, angle, and light intensity. Heinly et al. [102] show similar results for binary descriptors, including image transformations such as image compression, blurring, and occlusion. They illustrate reduced performance for detector-descriptor pairs that take account of image transformations not present in the data, recommending pairs that best match the expected properties of the data.

Two benchmarks are performed specifically in the context of indoor vSLAM with predominantly planar motion and minimal in-plane rotation, motion blurring, and lighting changes. Mozos et al. [87] introduced a

"survival ratio", showing how feature redetection rates typically decrease as the platform moves, and vary among detector-descriptor pairs. They showed that landmarks imaged in three or more consecutive frames, may be spuriously added as a new landmark if matching criteria fails. Schmidt et al. [100] showed similar results on the datasets from Aanaes et al. [106] and Sturm et al. [52], a tendency for correct correspondences to decrease as a function of angle and distance.

These studies show how different combinations perform better under different image contexts, but cannot come to a definitive conclusion on which work the best. The dependence on image transformations means that a feature detector-descriptor pair must be selected based on the image qualities.

Minimal processing time detectors such as FAST ensure speedy responses, but do not necessarily promote redetections. A feature detected by FAST in one frame, may be completely ignored in the next. Conversely detectors such as SIFT, SURF, and ORB take longer to process, but are more likely to redetect the same image feature across frames.

Scale, and rotation invariant descriptors such SIFT, SURF and ORB display better resilience to image deformations, but also come with a larger processing and memory overhead. Binary descriptors such as FREAK and BRIEF require minimal space and processing time, but naturally perform worse with larger image deformations and motion.

Once a feature has been detected and described by one of the aforementioned methods, it is matched using a similarity score such as the L1/L2 norm [88, 89], or Hamming distance [94, 95]. In the case of many features, additional heuristic methods are often employed to help refine matches. This includes K Nearest Neighbour (KNN) [104], Octave Pyramids [21], and Lowe Ratio [40, 88] matching schemes. These types of matching schemes are dependent on tuning parameters, and due to

the effects of precision versus recall, image deformations, and repetitive patterns, do not produce perfect correspondences [107]. A single detector-descriptor pair is unlikely to optimise both precision-recall and speed simultaneously.

Using a detector-descriptor pair with an arbitrary matching scheme may produce some correct correspondences, but will most likely be corrupted with outliers as discussed in [21, 35]. Sattler et al. [1] show in Figure 2.7 an example of high matching scores being a misleading measure of correspondence.

Figure 2.7: An illustration of matching limitations for local features described by Sattler et al. [1]. They showed that the coloured dots may be indistinguishable in the right image if compared on an individual basis. By looking in a larger neighbourhood around the point of interest (dotted circle), it may become easier to differentiate features as context helps define them.

The repetitive structure of the window panes is not discernible from a descriptor level. Outdoor environments may contain many areas with grass fields, or sand, making salient feature detection difficult with only local descriptors [47]. Without the whole image, discerning repetitive structures from each other is fundamentally difficult using image patches alone, and can lead to false-positive correspondences. Given these limitations to interest point detectors, what are the methods that have been presented to deal with these shortcomings?

Enforcing strict search grids/blocks/buckets has been one manner to

ensure consistent spatial structure and a sufficient number of features are extracted [21, 35, 51]. An initial set of features is detected for each block, and minimum/maximum number of features are enforced where only features with high detection scores are retained. These methods reduce the likelihood of features grouping together in one part of an image, increasing the accuracy of motion estimation.

Using existing knowledge about the context has been another approach to aid in better correspondence matching. Kitt et al. [108] show that due to difficult visual terrain, plain interest point methods failed to track across frames on tarred roads. Knowing the position of the camera relative to the road, they defined a ROI in consecutive images where a correspondence is likely to exist. Thus they only look for matches where they are likely to be found as opposed to the entire image. Stereo patch schemes such as Sum of Absolute Differences (SAD), Sum of Squared Differences (SSD), and Normalised Cross Correlation (NCC) rely on this concept heavily to constrain the search space using the epipolar geometry between two cameras. Johnson et al. [21] use estimated motion as a means to constrain the search space across frames, estimating a ROI in which to find correspondences. Sanfourche et al. [69] utilise initial motion estimates to guide the search space, doubling the accuracy of visual odometry.

Using multiple features resilient to different types of image deformations has been one approach to aid correspondences. Klein and Murray [109] use both line and interest point features, claiming that while interest point methods failed to track under motion blur, lines were robust. Cvisic and Petrovic [84] utilise both FAST and blob features, retaining the best set of heterogenous features to establish matches. This approach is more resilient, but requires more processing than simple matching, meaning an exhaustive list of different detector types may not be practical for real-time applications.

Ensuring good correspondences from a visual perspective is a critical

step in all the previously mentioned studies. Verifying the correctness of a correspondence is especially challenging from the local interest point perspective due to the potential of false-positive correspondences based on visual description alone. A high matching score indicates a potential correspondence, not necessarily a correct correspondence. With visual description alone, false-positives can be passed into the pipeline. Thus, identifying false-positives is a vital process for robust matching, and visual scores alone are not sufficient under additional sources of noise from viewpoint changes, lighting, and motion blur.

Given a landmark, a set of camera poses and the camera intrinsics, it is possible to estimate the landmark pixel position in each image using the pinhole camera model of Section 2.3. The reprojection of landmarks across frames is a geometric source of verification, whereby each landmark may have an associated Root Mean Square (RMS) pixel error as a result of measurement error, or pixel discretisation. If the pinhole camera model used is accurate, the reprojection error gives an additional set of criteria to aid in verifying the correctness of a correspondence.

One approach to speed up processing includes biasing the solution towards the best set of samples as seen by PROSAC [107] and SCRAMSAC [1]. PROSAC used descriptor matching scores as an indicator as to which correspondences to sample first, while SCRAMSAC used a spatial consistency check. Both increased RANSAC speeds substantially, one experiment of PROSAC reduced the number of iterations required from 106,534 down to 9.

There are a multitude of potential methods to improve robustness of correspondence and motion estimation, these include feature based approaches that work from the descriptor level, geometric methods that use the reprojection error, and heuristic methods that may utilise known information about the context. Determining which of these methods to include however depends entirely on the context or solution space in which

it is applied.

Specifically, the interaction between the hardware, environment and the chosen vSLAM solution may be difficult to evaluate before hand due to the individual variations between implementations. It is thus crucial that the solution space regarding a particular context be known before appropriate corrections can be included. Not all pipelines will benefit from blur and light correction, invariant descriptors, or sped up robustness schemes as these are solutions to specific problems. A design approach must be taken, where the different methods and heuristics can be objectively compared, and ultimately, a trade-off decided upon between speed, robustness, and accuracy.

Chapter 3

Methods

In this chapter, the experimental methodology is presented. We first outline the problem formulation and define the pose graph edges and nodes used in this body of work. The research platform, sensor calibration, and dataset collection methodology is then presented to provide context about the environment and hardware available.

A stereo front-end algorithm is then presented for the VCA and PE modules. A simulation of the PE stage is performed to characterise the performance requirements from VCA, as well as the expected reconstruction performance. Finally, the VCA and PE modules are tested on the collected dataset.

3.1 Problem Formulation

For the purposes of this work, an SRBA [68] approach to SLAM is adopted with the same notation. This is depicted in Figure 3.1 as a pose graph. A rectified stereo pinhole camera model is used as the sensor model with Figure 3.2 illustrating the coordinate frame conventions. The roll-pitch-yaw Euler angles $\beta - \gamma - \alpha$ convention of describing the pose orientation component is used where β is considered an anti-clockwise rotation about the Z-axis, γ an anti-clockwise rotation about the X-axis, and α an anti-clockwise rotation about the Y-axis. We use the notation $\mathbf{R}_\theta$ to indicate

a vector of Euler angles in degrees associated with a rotation matrix such that $\mathbf{R}_\theta = \begin{bmatrix} \beta & \gamma & \alpha \end{bmatrix}$ for convenience purposes.

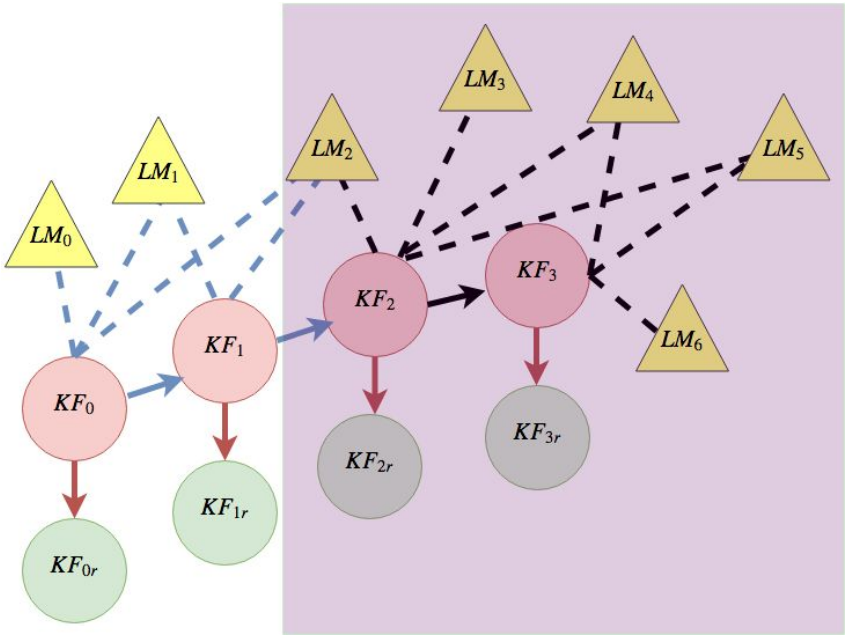


Figure 3.1: A pose graph representation of the SRBA coordinate system adopted in this work for a small problem where t_{kf}=4, t_e=3 and t_{lm}=7. The solid red arrows indicate the known baseline transform $K2K^k_{kr}$ which is constant throughout time, while the other solid arrows depict the keyframe edges $K2K^j_i$. The dotted lines depict landmark observations $\boldsymbol{z}^k_i$, where the purple box indicates an active two frame window.

The problem is broken down into the following components:

1. A set $\boldsymbol{KF}$ of size t_{kf} 6D robot poses or keyframes specified in a rectified sensor centric coordinate system where the simplifying assumption of $KF_k = KF_{cl}$ and $KF_{cr} = KF_{kr}$ is true.

2. A set $\boldsymbol{K2K}$ of size t_e keyframe to keyframe edges $K2K^i_{i-1}$. These edges are parameterised by $\beta - \gamma - \alpha$ in degrees, and the associated translation vector $a = X$, $b = Y$, $c = Z$ in metres. We assume the robot platform will not roll over completely, where both β and γ will be smaller than $\pm 90°$, avoiding the singularity problems associated with the Euler parameterisation.

3. A set $\boldsymbol{LM}$ of size t_{lm} nodes where each node depicts a single landmark LM_i. The base keyframe j is assumed to be the keyframe in which LM_i is first seen.

4. A set $\boldsymbol{z}$ of size t_z landmark edges $\boldsymbol{z}_i^k$ associated with frame k. $\boldsymbol{z}_i^k$ consists of a homogeneous triangulation $\tilde{X}_k^{i,j}$, observations $[z_k^{i,j} z_{kr}^{i,j}]$ in pixels, and pair of descriptors $[D_k^{i,j} D_{kr}^{i,j}]$.

5. A full system state $\boldsymbol{X}$ containing all the edges $[\boldsymbol{LMK2K}]$ contained within an SRBA pose graph

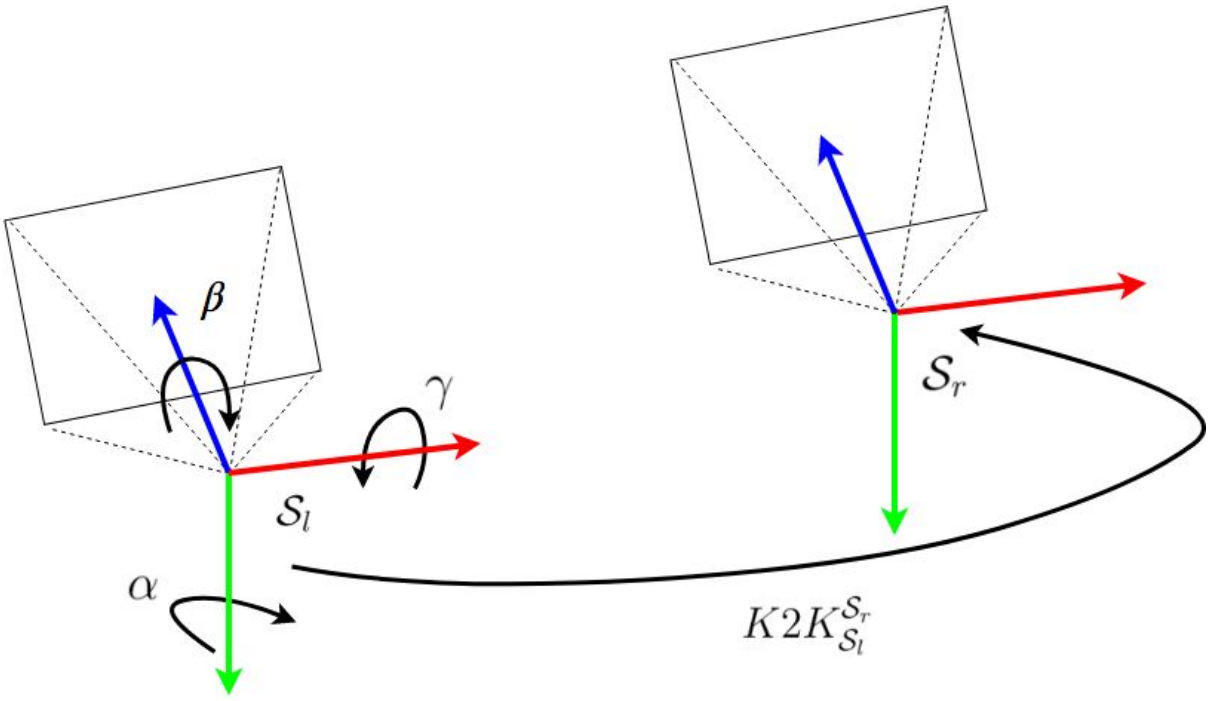


Figure 3.2: A graphical depiction of the stereo camera pose parameterisation where the red, blue, and green arrows indicate the X, Z, and Y coordinate axes.

3.2 Research Platform

A 1/10 scale VRX radio control car was used as the base of the robotic platform with a perspex support structure added to house the electronics as shown in Figure 3.3. A Point Grey BumbleBee (Discontinued) stereo camera was chosen as the primary camera source which has a 120mm baseline between cameras. The BumbleBee can receive 1024x768, 8-bit Bayer pattern stereo images at 7.5 and 15 frames per second (FPS) over a FireWire 400 port.

The main processing unit for the platform was a Nvidia Jetson TK1. This board contains a Cortex A-15 processor, 2 GB RAM, PCI-e mini card

expansion slot, and a 196 CUDA core enabled Graphics Processing Unit (GPU). A PCI-e expansion card was added to the platform to allow for compatibility between the BumbleBee camera and Jetson TK1. Due to the limited on-board memory, an external USB 3.0 stick with 50Mb/s duplex speed was used for additional space.

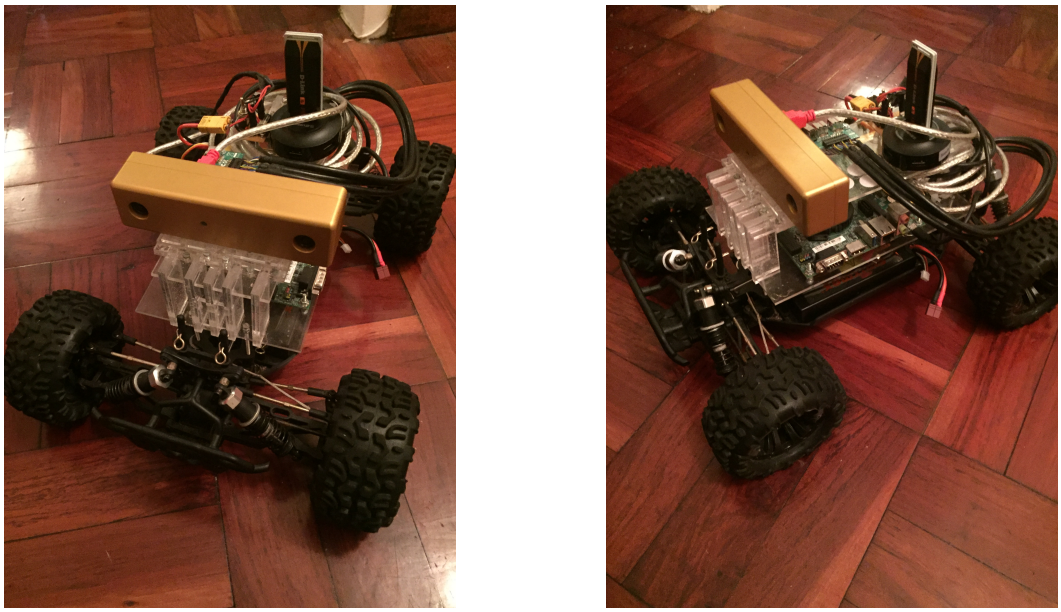


Figure 3.3: The research platform used throughout the study is shown here with the camera, power supply, and processing unit mounted on a perspex housing.

3.3 Dataset Collection and Organisation

For the purposes of this study, a video sequence titled A was recorded with the research platform of Section 3.2 in outdoor terrain. Sequence A was an artificially created semi-structured, planar, outdoor environment. Figure 3.4 shows a top down view of sequence A, where coarse measurements of key locations were taken with a measuring tape indicated by the pink lines.

The artificially placed features in environment A allowed a small degree of control over the depth and expected landmark tracks in the sequence, while still exposing the research platform to the outdoors. In addition, the planar nature of the environment restricted motion to 2D, giving an easy manner to graph and interpret the mapping results.

The green region contained high vegetation content with traffic cones placed close to the intended trajectory. The red region had minimal traffic

cone placements and was predominantly a grass plane. The blue region contained some buildings in the distance while the yellow region contained buildings, hills, and nearby cones.

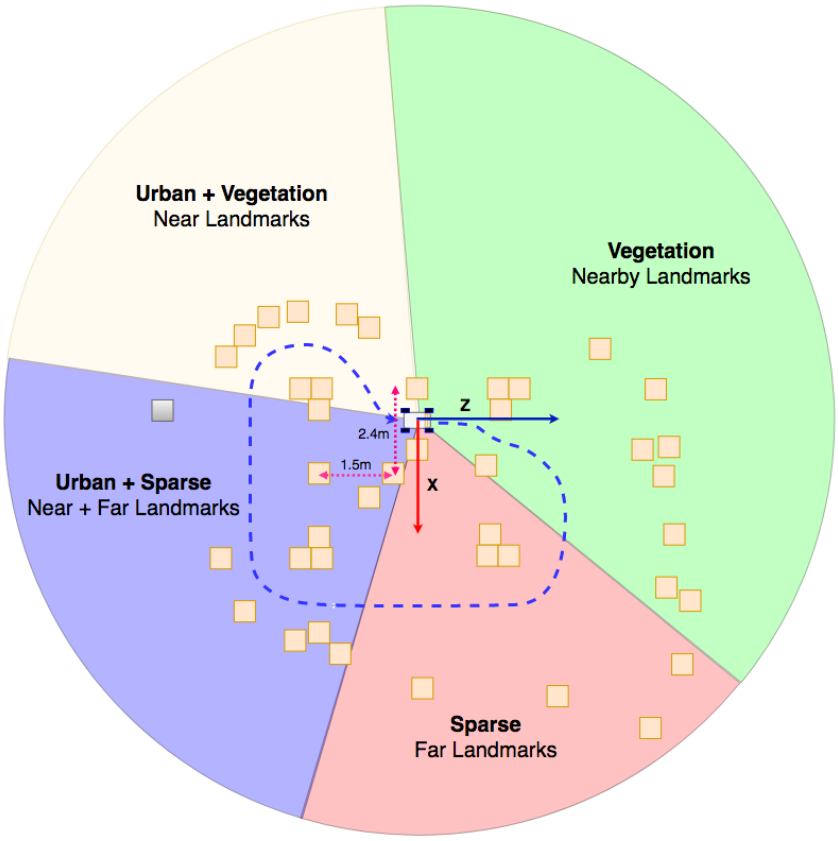


Figure 3.4: A top down view of Sequence A not to scale. Each square represents a traffic cone and the two axes illustrate the map reference frame. The blue line shows an estimate of the expected trajectory with the predominant components being X, Z, and α.

The robot was manually driven over the same path for multiple sequences in order to create a series of trajectories with loop closure conditions at 3 different speeds. These are indicated as A_n where A is the dataset name and n the sequential loop being referred to. Each loop was selected by hand with the intention of starting and stopping with the same pose relative to the world coordinates. Because of the manual control and human error, these start and stop poses will contain an offset translation and orientation error which is assumed to be negligible in the context of the larger loop and reconstruction error.

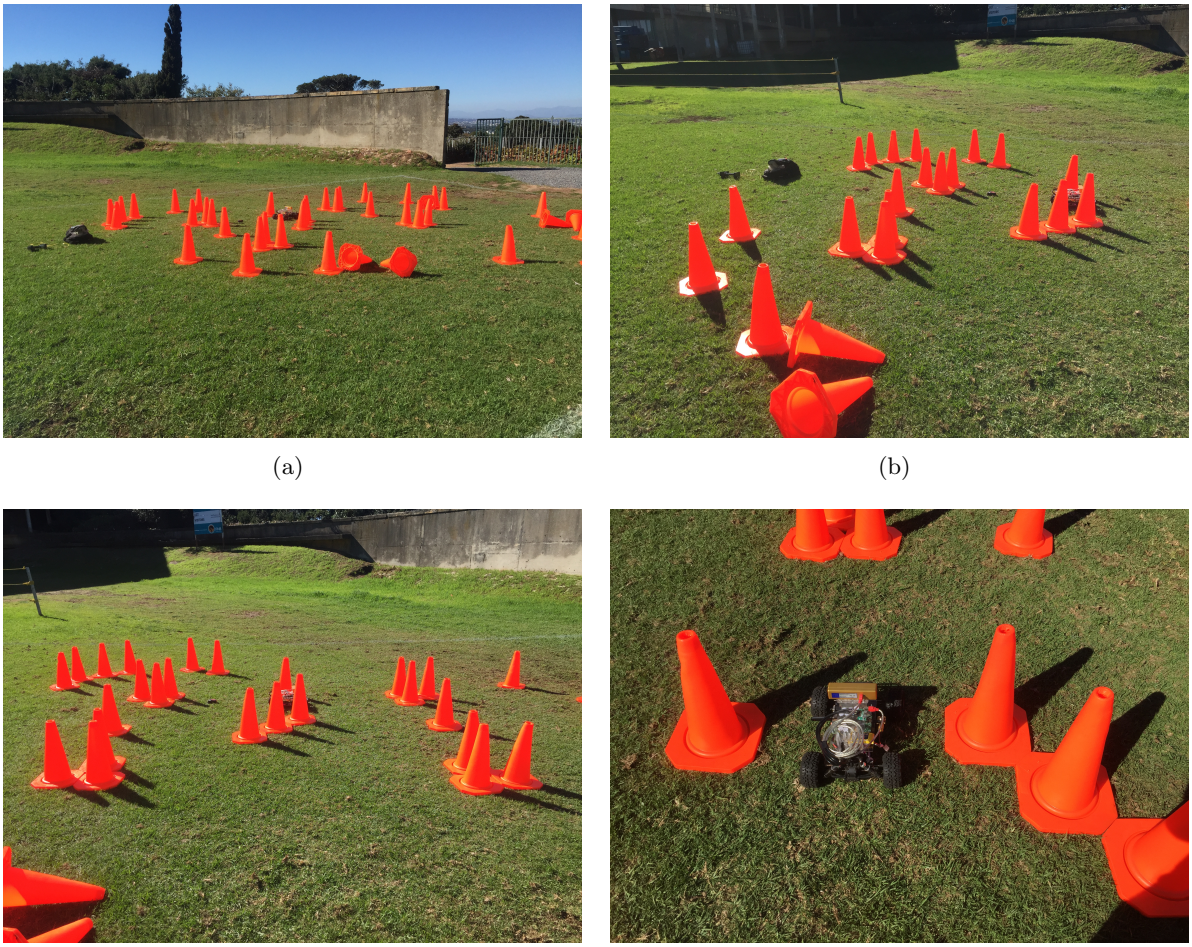
Each loop is additionally assigned a speed label which may either be

"Slow", "Medium", or "Fast", depending on the number of images in each loop N_{A_n}. Under the assumption of constant frame rate and path trajectories, N_{A_n} is a good heuristic of speed and is classified according to Equation 3.1 where T_{Fast} and T_{Medium} are thresholds. Thus an individual loop may be referred to as A_1^{slow} where 1 is the loop ID, and slow indicates which category it was assigned.

$$Speed = \begin{cases} Slow, & N_{A_n} > T_{Medium} \\ Medium, & T_{Medium} <= N_{A_n} <= T_{Fast} \\ Fast, & N_{A_n} < T_{Fast} \end{cases} \tag{3.1}$$

Camera settings were left on automatic to compensate for the variation in lighting levels, but fixed at 15 FPS. Timestamps for each image received were also logged during each sequence to validate the FPS assumption. These are used to calculate the variation about the 15Hz recording time.

The frame rate and camera settings gave each sequence the best chance of containing landmarks to track without saturating image brightness. This would ensure inter-frame motion was purely a function of the change in robot pose, as opposed to inconsistent frame rate. Sample images from sequence A are shown in Figure 3.5.



(a) (b)

(c) (d)

Figure 3.5: A set of images showing sequence A. (a)-(d) show general pictures of the environmental setup. (d) illustrates the starting position of the robot.

3.4 Camera Model and Calibration

A stereo calibrated camera setup was assumed, whereby the intrinsic matrices with a 5-parameter distortion model, as well as the extrinsic stereo parameters were known. The OpenCV library [110] and image coordinate system were used where the top left corner of the image is position $u = v = 0$ and the bottom right corner is position $u = width$, $v = height$. The BumbleBee camera is supplied with calibration from the manufacturer with 0.05 RMS pixel error. However, this is only available through the Software Development Kit (SDK) provided in Windows and uses proprietary calibration algorithms. As such, the camera matrices and the relevant calibration procedure used by Point Grey were unavailable.

A 9x12 checkerboard pattern with 45mm squares was used to calibrate the system using OpenCV calibration functions and a set of 351 checkerboard images with varying orientations. In each image, the checkerboard pattern was searched for with adaptive thresholding and if found, an iterative subpixel refinement process was applied over a 7x7 window with a maximum of 1000 iterations. This produced an initial set of intrinsic parameters for the left and right cameras, total RMS reprojection error for the whole sequence, and a set of average RMS errors for each individual checkerboard image.

A filtering stage was then applied to remove images from the calibration where the average RMS error for an image pair was above a threshold $T_{Calibration}$. This filtering stage removed poor quality images where a checkerboard was found, but may have imaging artifacts affecting the measurement such as motion blurring or reflections. Retaining these images with large RMS errors can lead to calibration parameters that deviate from the true pinhole camera model.

The calibration process was then repeated on the filtered set of images to produce a coarse pair of intrinsic parameters for the left and right camera.

Figure 3.6 shows an example of a matching image pair with the found checkerboard overlaid, and correspondences drawn with green lines.

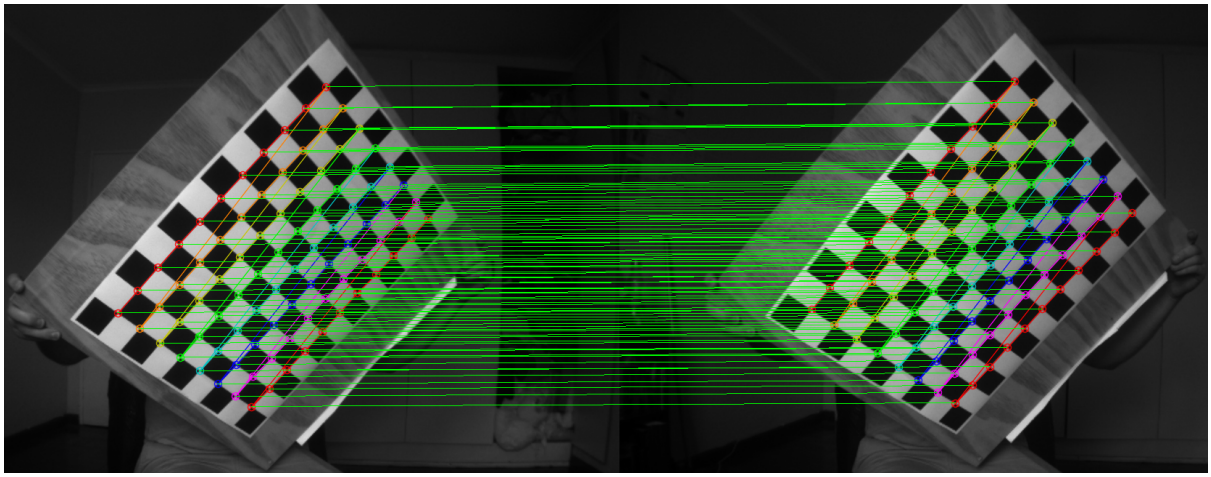


Figure 3.6: An example of a stereo checkerboard image pair where the coloured lines indicate the checkerboard found, and the green lines indicate correspondences. Note how the lines of correspondence are non-horizontal.

The coarse intrinsic parameters were fed into an iterative stereo optimisation algorithm as the initial guess where the reprojective error was minimised. This produced a set of optimised intrinsic and extrinsic parameters from which rectification mapping functions $\boldsymbol{R_k}$ and $\boldsymbol{R_{kr}}$ were derived for the left and right cameras respectively. A ROI is defined for each image, however epipolar lines are only valid if they are defined in both left and right ROI. For this reason, the ROI is further reduced to where both ROI overlap vertically.

Our first measure of calibration performance is defined as the ROI overlap percentage $ROI\% = 100 \times (ROI_{Width} \times ROI_{Height})/(width \times height)$. The second performance measure was the RMS epipolar error in the rectified images. This was measured by first rectifying the image according to mapping functions $\boldsymbol{R_k}$ and $\boldsymbol{R_{kr}}$. The checkerboard pattern was then found using the same methods from the initial calibration stage and the RMS epipolar error calculated for each stereo image pair.

A coverage map of the calibration pattern points used in the calculation

was produced as an additional heuristic method to evaluate rectification performance. This map consists of all the found checkerboard patterns in the rectified images overlaid on a single image, with ideal epipolar horizontal lines as a reference. This helps visualise the calibration results and look for inconsistencies in the rectification map.

Execution times of the debayer and rectification process using the calculated rectification map were recorded on a laptop with an i7-5500U processor, and the Jetson TK1. This calibration process was executed three times with different filtering values $T_{Calibration} = (1.0, 1.1, 1.2)$ titled Cal_A, Cal_B, and Cal_C respectively. The calibration with the best trade-off between RMS epipolar error, $ROI\%$, and speed of implementation was then chosen as the final calibration set.

The ideal calibration set would have the lowest epipolar error, largest $ROI\%$, and fastest speed. However, we prioritise the trade-off as epipolar error first, $ROI\%$ second, and speed third. Without a low epipolar error the reconstruction result would be inaccurate resulting in a poor vSLAM result and as such takes precedence.

While $ROI\%$ is important, it need only be large enough such that features are trackable. So long as the $ROI\%$ does not drop excessively low it can be compromised. Rectification speed takes lowest priority as it is predominantly dictated by hardware. This makes rectification speed less of a decision variable in the study where a single processor is available on the platform.

3.5 Visual Cue Acquisition

In this section, the VCA module is described by three independent, but core algorithms chained together to fulfil the functional requirements expected of the VCA stage. They consist of a feature extraction process, a stereo matching algorithm, and an inter-frame tracking algorithm.

It is assumed that the intrinsic and extrinsic parameters associated with the best calibration set Cal_A, Cal_B, or Cal_C is selected according to the criteria of Section 3.4 and is readily available across the entire processing pipeline.

3.5.1 Feature Extraction

Before feature extraction begins, stereo rectification is performed on a synchronised stereo image pair as described in Section 2.4. The ROI is then applied to the rectified images to ensure only triangulatable features are extracted.

A gridded adaptive FAST algorithm was chosen as the primary detection mechanism for the module and is a hybrid detection scheme from [48] and [35]. Non-Maximum suppression of the results is enforced to increase the spatial distance between detected features and reduce the likelihood of bunched features. The FAST detector was primarily chosen for the low computational requirements, but additionally provides a single tuning threshold T_{FAST}, making for a simple adaptive control law which is implemented as follows.

The ROI is divided into N_r rows and N_c columns with independent threshold values $T_{FAST}^{r,c}$ at row r and column c to aid detection coverage. A minimum threshold value T_{FAST}^{min} is enforced to ensure sensitivity saturation in image regions where there may be no discernible features.

A setpoint of $T_{Setpoint}$ total detected features is defined where each grid has a setpoint $T_{Grid}^{r,c} = T_{Setpoint}/(N_r \cdot N_c)$. An individual grid setpoint error is calculated as $e_g^{r,c} = T_{Grid}^{r,c} - N_{det}^{r,c}$ where $N_{det}^{r,c}$ is the number of detected FAST features at a threshold level $T_{FAST}^{r,c}$. Subpixel refinement is performed on each feature using a 5x5 window, and maximum of 40 iterations.

$N_r = 2$ was selected for the implementation, breaking the view up into a "ground" and "sky" region. Landmarks in the ground region are more likely to be closer to the camera, and thus increase the likelihood of accurate translation estimates. The ground region is biased to have additional features by implementing an unsymmetrical $T_{Grid}^{r,c}$ pattern with $T_{Grid}^{ground,c} = 2 * T_{Grid}^{r,c}$ and $T_{Grid}^{sky,c} = 0.5 * T_{Grid}^{r,c}$.

At the end of each frame, $T_{FAST}^{r,c}$ is updated if $|e_g^{r,c}| > T_{deadband} * T_{Grid}^{r,c}$ where $T_{deadband}$ is on the interval [0,1]. $T_{deadband}$ is an extractor tuning parameter that dictates what percentage error is required on $T_{Grid}^{r,c}$ before an adjustment action is required.

If this condition is met, $T_{FAST}^{r,c}$ is adjusted in the following way. Positive $e_g^{r,c}$ indicates an excessive number of features were detected, and on the next iteration, sensitivity should be reduced by incrementing the detection threshold $T_{FAST}^{r,c}$ += 1. Conversely, a negative value indicates a feature shortage, and the threshold is decremented $T_{FAST}^{r,c}$ −= 1. If $T_{FAST}^{r,c}$ is smaller than the minimum threshold value T_{FAST}^{min}, the threshold is clamped to $T_{FAST}^{r,c} = T_{FAST}^{min}$. Each set of grid detections is combined together into a set of left and right observations $[z_k^{i,j} z_{kr}^{i,j}]$.

Feature detection results are computed for loops A_3^{Slow} and A_{13}^{Fast} to illustrate setpoint tracking for $T_{Setpoint} = [1000, 3000, 5000]$.

3.5.2 Stereo Matching

For each observation found in the left and right stereo frames $[z_k^{i,j} z_{kr}^{i,j}]$, an associated descriptor $[D_k^{i,j} D_{kr}^{i,j}]$ is computed with algorithm $\mathcal{D}$.

SURF, ORB, BRIEF, and FREAK algorithms are benchmarked for $\mathcal{D}$ under different values of $T_{Setpoint}$. A 16 and 64 byte version of BRIEF, extended and standard SURF, standard FREAK, and ORB with 10 and

70 patch size were used.

A brute force matching scheme with cross correspondence is applied between descriptor sets $D_k^{i,j}$ and $D_{kr}^{i,j}$ to produce a matching set M_k. Any matched feature in M_k is considered a false-positive match if its epipolar error is higher than threshold $T_{stereoInlier}$. This is selected based on the average RMS epipolar error from calibration in Section 3.4. The remaining inliers M_k^{stereo}, are treated as good correspondences and the landmark edge structure $\boldsymbol{z}_i^k$ is populated.

The total number of landmark edges added each frame was recorded on loop A_3^{slow} under different descriptor algorithms $\mathcal{D}$ to illustrate the effects $\mathcal{D}$ had on matching. Computational cost was recorded for both an i7 processor implemented in Python, and on the Jetson TK1 implemented in C with bindings to an optimised OpenCV4Tegra library.

3.5.3 Inter-frame Tracking

In the preceding section, a set of landmark edges was created based on epipolar constraints from stereo matches M_k^{stereo}. While this is enough to generate a landmark edge in the pose graph, it is not clear which landmark vertex the edge should be associated with. The inter-frame tracking module either generates a new landmark vertex, or matches it to a previously seen landmark vertex over a two frame window.

Tracking is treated as two separate monocular cases for the left and right camera respectively. Brute force matching with cross correspondence is applied between the left and right landmark edges independently, such that two sets of matches M_k^{k+1} and M_{kr}^{kr+1} are created.

If the same feature is matched in both M_k^{k+1} and M_{kr}^{kr+1}, then it is considered a good feature track and the edge is associated with an already

existing landmark. If there is no associated track in the new frame, it is considered a new landmark and added to the graph as a new vertex and edge. The inlier percentage and total matches were recorded for $Fast$, $Medium$, and $Slow$ loops.

3.6 Parameters Estimation

In the preceding two sections, a stereo algorithm was described that could populate a set of landmark edges and vertices. Given this set of edges, $\tilde{X}_k^{i,j}$ is generated for each edge according to Equation 2.4, and an initial relative pose edge $K2K_k^{k+1}$ estimated. If $\tilde{X}_k^{i,j}$ evaluates to a negative depth (possible under outlier conditions), the track is considered a false-positive and removed from the track set.

The algorithm uses a point cloud registration method to directly solve for metric relative pose edges. The SVD-based solution from [82] is wrapped by a RANSAC layer. A minimum of 3 samples is required to solve for a rigid body transform $\mathcal{T}_k^{k+1}$. Two termination criteria T_{RANSAC}^{max} and T_{RANSAC}^{RMS} are defined for the RANSAC wrapper. T_{RANSAC}^{max} defines the maximum permittable iterations allowed for a single pose estimate. For any parameter set, if the average inlier RMS reprojective error falls below T_{RANSAC}^{RMS}, execution is stopped and the current set is accepted as the best pose solution.

Two tuning parameters are defined for the algorithm. T_{RANSAC}^{inlier} defines the RMS reprojective error required for a landmark to be considered part of the inlier set. A large value for T_{RANSAC}^{inlier} will mean most tracks will be considered inliers and could allow for excessive noise to be introduced. Low values of T_{RANSAC}^{inlier} will mean fewer tracks will be considered inliers, leading to complete motion failure conditions if too few are found.

T_{RANSAC}^{min} defines the minimum number of inlier landmarks for the parameter estimate to be considered as the best fit. If too few inliers are detected over all iterations, relative pose estimation is considered to

have failed and $\mathcal{T}_k^{k+1}$ is set to unity. A motion threshold could also be added to the algorithm, but is not implemented in this work due to the fixed speed assumption on the dataset collection.

3.7 Pose Graph Simulation

In this section we outline a full pose graph simulation used to characterise the expected performance of the PE module. Two categories of motion, $K2K_{Forward}$ and $K2K_{Steering}$ are simulated, where a motion edge is of either dominant forward translation, or constant steering angle.

Dominant forward translation simulates the robot platform moving forward in a straight line with random perturbations in orientation and translation. The constant steering angle includes forward motion and random perturbations, but has an additional α rotation component, modelling the robot driving with a constant steering angle.

The same rectified and calibrated coordinate system is used as described in Section 3.1. Relative pose edge components $\boldsymbol{R}$ and $\boldsymbol{C}$ were modelled as independent Gaussian distributions generated according to Equations 3.2, 3.3 and 3.4. The speed of motion is dictated by μ_{Cmean}^{speed} and μ_{Rmean}^{speed} values in the Z and α component respectively, while the standard deviations σ_{Rnoise} and σ_{Cnoise} model rough terrain. The subscript $clip_{min}^{max}$ indicates a function is limited in the range min to max.

A camera speed of 15FPS is assumed with maximum forward speed of 0.66m/s such that $\mu_{Cmean}^{speed} = 0.044$. The absolute value of Z is taken for both $K2K_{Forward}$ and $K2K_{Steering}$ to ensure motion is always forward. The absolute value of α is taken for $K2K_{Steering}$ to ensure only a single turn direction is present in the simulation where $\mu_{Rmean}^{speed} = 4$.

The complete trajectory is generated by chaining together 150 $K2K_{Forward}$, 30 $K2K_{Steering}$, and 60 $K2K_{Forward}$ pose edges. These 240 edges effectively simulate a 16-second long trajectory, composed of two straight segments

linked by a single cornering segment. Note that the noise components are scaled in the β and γ components to constrain the motion to a predominantly planar trajectory.

$$C_{Forward} = C_{Steering} = \begin{bmatrix} \mathcal{N}(0,\, \sigma_{Cnoise}) \\ \mathcal{N}(0,\, \sigma_{Cnoise}) \\ |\mathcal{N}(\mu^{speed}_{Cmean},\, \sigma_{Cnoise})| \end{bmatrix} \qquad \sigma_{Cnoise} = 0.3 * \mu^{speed}_{Cmean} \tag{3.2}$$

$$R^{T}_{\theta Forward} = \begin{bmatrix} \mathcal{N}(0,\, \sigma_{Rnoise}) \\ 0.2 * \mathcal{N}(0,\, \sigma_{Rnoise}) \\ 0.2 * \mathcal{N}(0,\, \sigma_{Rnoise}) \end{bmatrix} \qquad \sigma_{Rnoise} = 2 \tag{3.3}$$

$$R^{T}_{\theta Steering} = \begin{bmatrix} \mathcal{N}(0,\, \sigma_{Rnoise}) \\ 0.2 * \mathcal{N}(0,\, \sigma_{Rnoise}) \\ 0.2 * |\mathcal{N}(\mu^{speed}_{Rmean},\, \sigma_{Rnoise})|_{clip^{20}_{0}} \end{bmatrix} \tag{3.4}$$

3.7.1 Landmark Edge Model

For motion to be extracted via the module described in Section 3.6, features have to be tracked between keyframes. A total of T_{track} ideal landmarks are generated at KF_0, such that they are visible within the ROI at both KF_1 and KF_0 according to Algorithm 1.

All landmark triangulations $\tilde{X}^{i,j}_{k}$ additionally follow restrictions, such that the Z component must be positive and the Y component has a minimum value of -0.5m . The first constraint ensures that all projections are in front of the camera. The second constraint simulates the effects of the ground as a result of the vehicle height, as features will be unlikely to appear lower than this for planar motion.

The keyframes are then looped through one at a time, and for each keyframe with less than T_{track}, a new landmark is generated according to Algorithm 1. In addition to the ideal projection case, Gaussian noise cases

are simulated by adding $\mathcal{N}(0,\, \sigma_{pixelNoise})$ to the entire set of $\boldsymbol{LM}$ landmark edges in only the u component. This ensures that the rectified projection model is not violated, as any misalignment in the vertical component translates directly into reprojection error according to P_l and P_r. Gaussian noise levels were simulated for $\sigma_{pixelNoise} = [0.05, 0.1, 0.2, 0.3]$ along with different inter-frame tracks $T_{track} = [50, 100, 200, 400]$.

For each T_{track} value, the element-wise difference between R_θ and C is plotted as a 25 bin histogram over the entire sequence. A Gaussian distribution is fitted to the data and overlaid onto the histogram to demonstrate the error distribution. A scatter plot of inlier detections versus inlier RMS reprojective error is also shown to illustrate the effects of poor parameter fitting during pose estimation.

Algorithm 1 Psuedo Landmark Edge Generation

InputData: $[KF_k, KF_{k+1}, ..., KF_{k_{total}}]$
OutputData: A new landmark LM_i with edges $[\boldsymbol{z}^i_k, \boldsymbol{z}^{i,k}_{k+1}, ..., \boldsymbol{z}^i_{k_{total}}]$
while Not Valid Point **do**
    $\tilde{X}^i_k = \begin{bmatrix} \mathcal{N}(\mu_X,\, \sigma^2_X) & \mathcal{N}(\mu_Y,\, \sigma^2_Y) & \mathcal{N}(\mu_Z,\, \sigma^2_Z) & 1 \end{bmatrix}^T$
    $\tilde{z}^i_k = P_l \tilde{X}^i_k$
    $\tilde{z}^i_{kr} = P_r \tilde{X}^i_k$
    $\tilde{X}^{i,k}_{k+1} = \mathcal{T}^k_{k+1} \tilde{X}^i_k$
    $\tilde{z}^i_{k+1} = P_l \tilde{X}^i_{k+1}$
    $\tilde{z}^i_{k+1r} = P_r \tilde{X}^i_{k+1}$
    if $\tilde{z}^i_k$ and $\tilde{z}^i_{kr}$ and $\tilde{z}^i_{k+1}$ and $\tilde{z}^i_{k+1r}$ within ROI **then**
        Valid Point=True
        Populate edge $\boldsymbol{z}^i_k, \boldsymbol{z}^{i,k}_{k+1}$
        while $\tilde{X}^i_k$ visible in future KF **do**
            transform into KF frame $\mathcal{T}^k_{KFfuture}$
            predict pixels $\tilde{z}^i_{KFfuture}, \cdots$
            if $\tilde{z}^i_{KFfuture}, ...$ within ROI **then**
                Populate edge $\boldsymbol{z}^i_{KFfuture}$
            end if
        end while
    end if
end while

Chapter 4

Results

In this chapter, the results of the described system and experiments from Chapter 3 are presented. This includes dataset collection, sensor calibration, VCA, PE, and reconstruction performance of the system. Performance of the VCA module was quantified in terms of total stereo landmarks and processing time. A PE module was evaluated in simulation and experimentally in terms of pose accuracy and inter-frame tracks. Lastly, the combined VCA and PE modules were evaluated in terms of overall odometer error under different levels of speed. This chapter functionally fulfills Item One of the research questions, the evaluation and study of a front-end feature tracking framework.

4.1 Dataset

In Section 3.3, we described sequence A as being segmented into different categories of speed. Figure 4.1 shows the speed categories for each loop using $T_{Fast} = 300$ and $T_{Medium} = 600$. An average of 0.1ms deviation from the quoted 15 FPS was calculated from time stamped stereo images, indicating a consistent recording speed. The start and stop image indices for each loop relative to the whole sequence are attached under Appendix C.

The set of *Slow* loops is selected as A_1^{Slow}, A_2^{Slow} and A_3^{Slow}. Due to the

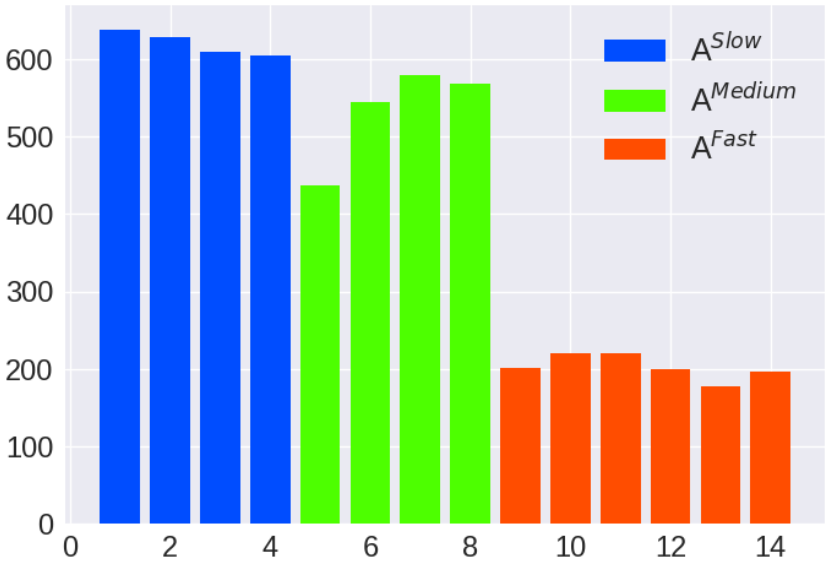


Figure 4.1: A bar graph of speed categories for dataset A where the X-axis displays loop ID and the Y-axis the number of images in the loop

highly similar timings on the medium dataset, only A_5^{Medium} and A_6^{Medium} are used as viable loop samples. Additionally, A_{11}^{Fast} hit an obstacle during the trajectory, making it an invalid loop sample. Only A_{12}^{Fast}, A_{13}^{Fast} and A_{14}^{Fast} were considered for the $Fast$ loop analysis

4.2 Calibration

In this section, the results of stereo camera calibration is presented from 351 image pairs for Cal_A, Cal_B and Cal_C where only 329 image pairs contained valid stereo pairs.

The initial RMS reprojective error before thresholding was 2.239 and 0.860 for the left and right camera respectively. The left camera calibration had a significantly higher average RMS error than the right camera, indicating a poor calibration fit. A histogram of the average RMS error per image pair is shown in Figure 4.2 for 25 bins along with a Gaussian data fit. The combined RMS error displayed a bi-modal distribution as a result of the large left camera error.

The red, green and blue dotted lines on Figure 4.2 illustrate where thresholding was set on the data. There were 144, 155 and 169 images

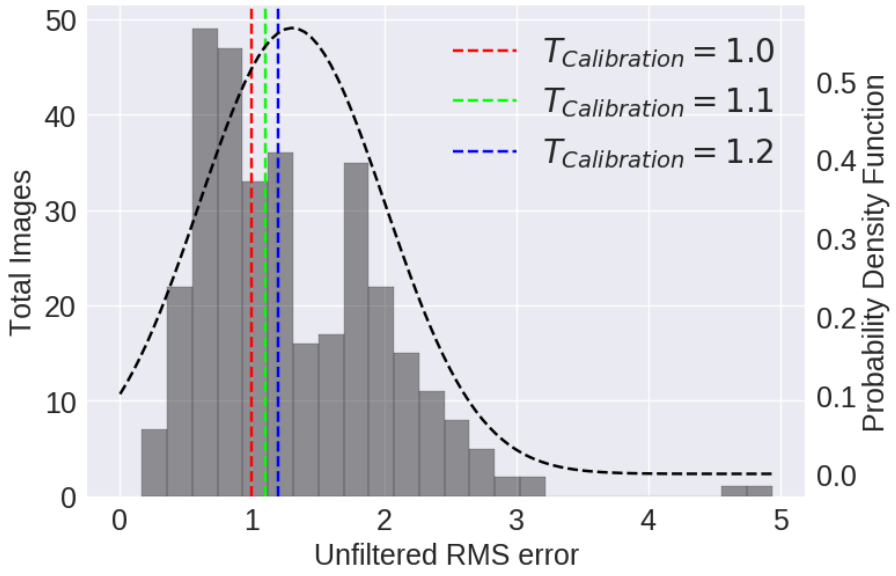


Figure 4.2: A histogram showing raw RMS error from the initial calibration procedure. The black dotted line indicates the estimated probability density function of the fit. The red, green, and blue dotted lines indicate where thresholding took place for the calibration datasets Cal_A, Cal_B, and Cal_C respectively.

remaining after thresholding in Cal_A, Cal_B and Cal_C respectively. Figure 4.3 displays the calibration reprojection error with the filtered set of stereo images.

The RMS reprojective error associated with the left and right camera was $[0.817, 0.821]$, $[0.869, 0.852]$ and $[0.907, 0.885]$ for Cal_A, Cal_B, and Cal_C respectively. All three filtered reprojection errors displayed improved error distributions versus raw calibration, suggesting outlier images skewed the initial calibration.

Using the filtered data, the stereo calibration algorithm gave 0.719, 0.703 and 0.730 RMS reprojection for Cal_A, Cal_B, and Cal_C respectively. A histogram of the RMS epipolar associated with each pair of rectified images is shown in Figure 4.4 where the average RMS epipolar error and $ROI\%$ was $[0.330, 73.95]$, $[0.270, 77.83]$ and $[0.280, 80.52]$ for Cal_A, Cal_B and Cal_C respectively.

The combined debayer and rectification processing time is shown in Figure 4.5 for the i7 and Jetson TK1 platforms where the average time was 16

and 32 ms respectively. Estimated intrinsic and extrinsic parameters are summarised in Appendices A and B respectively. Cal_B had the lowest

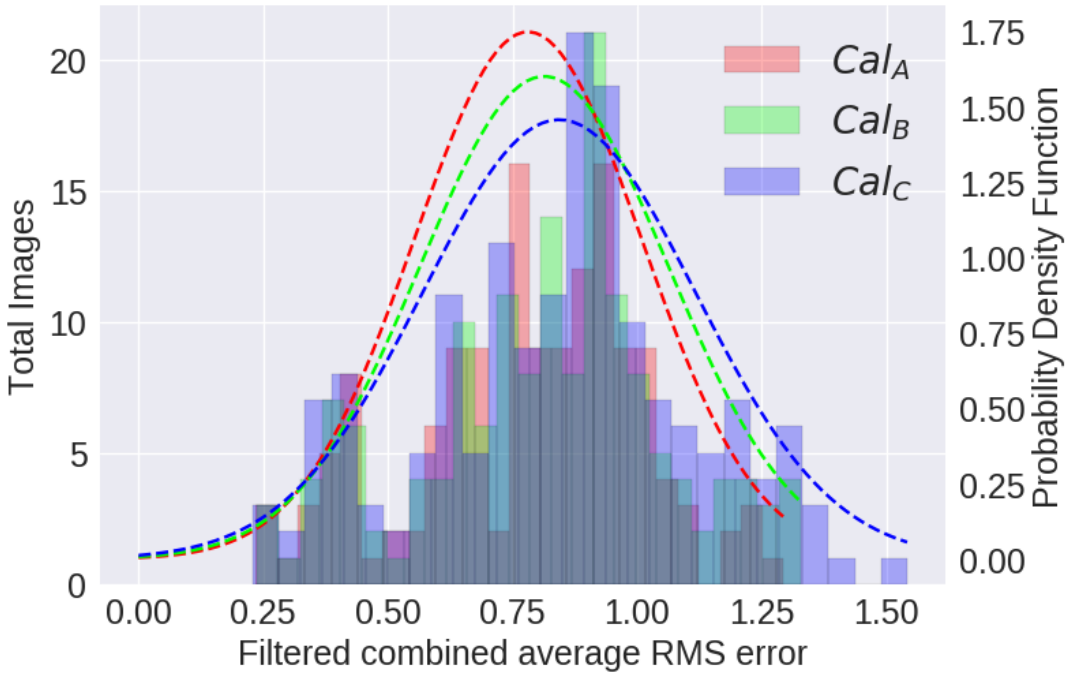


Figure 4.3: The set of superimposed histograms for the average RMS error per stereo image pair before stereo calibration was performed. A curve of best fit to the normal distribution is overlaid for each calibration sequence where each colour represents a different $T_{Calibration}$ used.

RMS error, epipolar error and second largest $ROI\%$, making it the most appropriate calibration set. An example coverage map and rectification output is shown in Figure 4.6 for Cal_B illustrating horizontal epipolar lines. The checkerboard pattern is predominantly located in the centre of the image, meaning images outside this region contained higher reprojection errors.

The quoted manufacturer epipolar error is 0.05 pixels, approximately 5-6 times smaller. This may either be as a result of calibration images not sufficiently covering the edges, or the distortion model not being appropriate near the edges of the images. The 120mm baseline of Cal_B conformed to the quoted manufacturer baseline, suggesting successful calibration but larger error than the proprietary software.

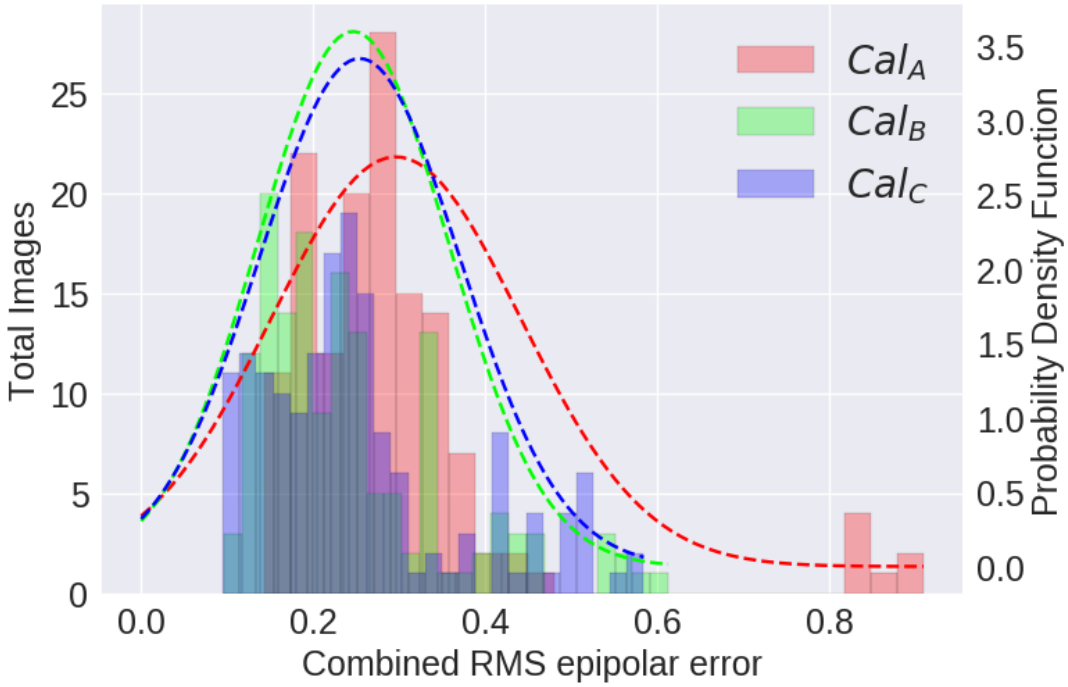


Figure 4.4: A set of superimposed histograms of the combined average RMS epipolar error after stereo rectification. Each figure corresponds with the calibration set and $T_{Calibration}$ from section 4.2

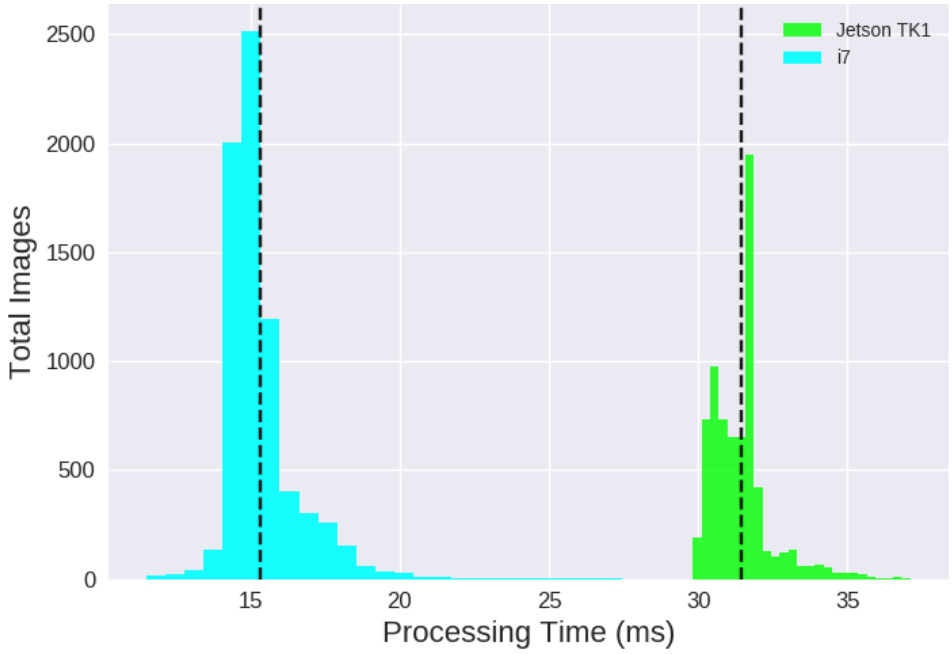
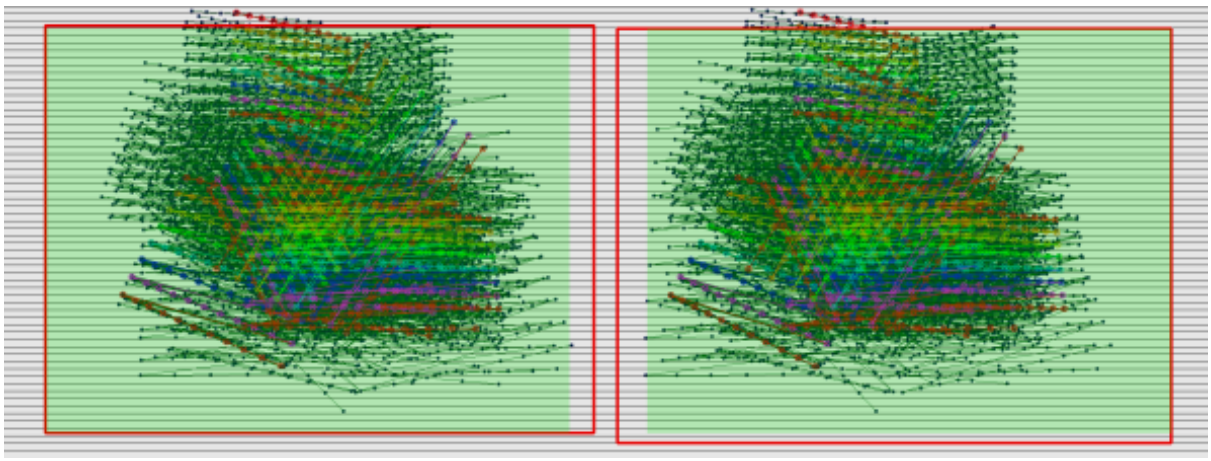
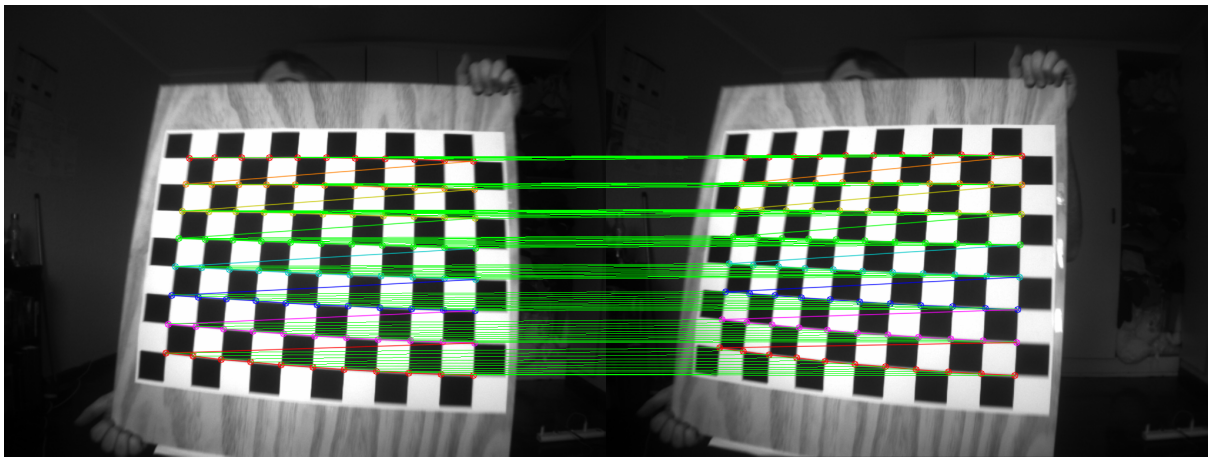


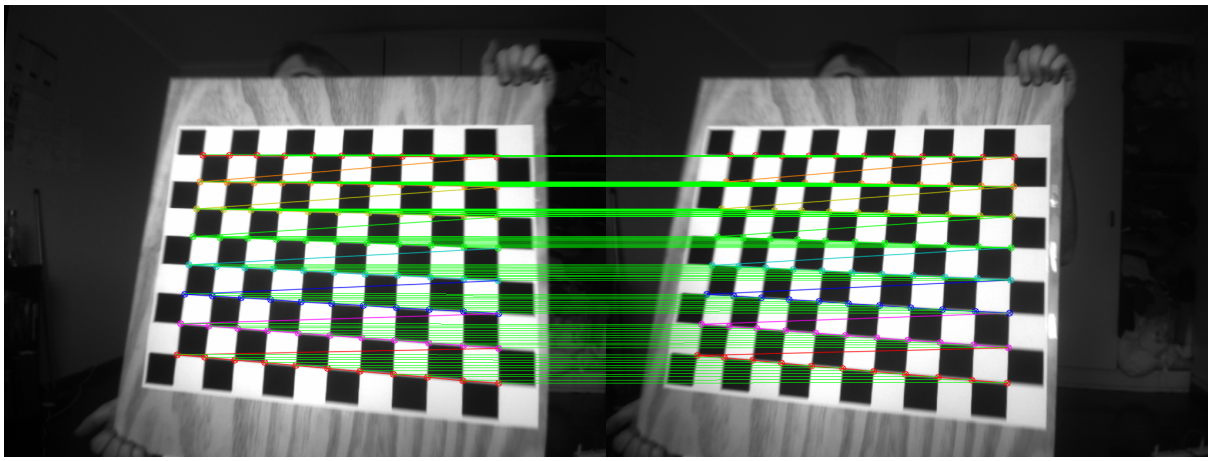
Figure 4.5: A histogram of the processing time required to debayer and rectify an image with an i7 and Jetson TK1. The dotted line shows the average time taken for Cal_B.



(a) Coverage Map



(b) Unrectified

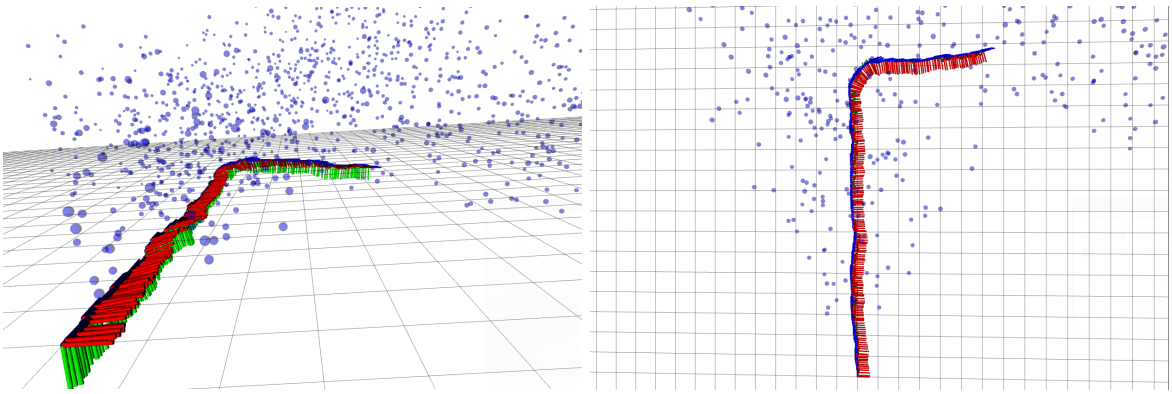


(c) Rectified

Figure 4.6: A set of images showing the coverage map and rectification output for Cal_B. 4.6(a) shows checkerboard patterns used for the final calibration, with a random subset highlighted in colour to make it easier to follow matches with the eye. The black horizontal lines show ideal epipolar lines between left and right stereo images. The red border indicates the individual ROI for each left and right camera while the green shaded area indicates the overlapping ROI. 4.6(b) and 4.6(c) show the unrectified and rectified images respectively where green lines show stereo matches.

4.3 Parameter Estimation Simulation

In this section, the results of the PE simulation are presented. Figure 4.7 shows a visualisation of a single simulation for $T_{track} = 100$ where global landmarks are depicted in blue and the overall trajectory as a set of axes. A summary of the settings and parameters used for the VCA and PE modules can be found in Appendix D.



(a) (b)

Figure 4.7: Visualisations of the simulated path with 100 tracks on a $0.5 \times 0.5m$ grid. (a) shows an arbitrary view of the trajectory and world landmarks in blue. (b) shows the same trajectory from a birds eye view, illustrating the straight trajectory and single turn described in 3.7

Figures 4.8 and 4.9 describe the orientation and translation error associated with each pose under different T_{track}. Pose estimates with lower Gaussian noise had sharper probability density functions (PDF) for the error in both orientation and translation as seen for $\sigma = [0.05, 0.1]$.

Higher noise values with $\sigma = [0.2, 0.3]$ showed performance degradation with flatter PDF outputs. $\sigma = 0.3$ showed significant standard deviation where the PDF was close to horizontal. Increasing the number of landmark tracks narrowed the standard deviation in some areas, but showed no significant improvement for $\sigma > 0.1$.

Translation estimates were far more sensitive to Gaussian noise than orientation. Relative to the simulation parameters used, translation deviation was significant. For $\sigma = 0.3$, the Z standard deviation was 100% larger than the input measurement. Z error showed a larger

standard deviation than X and Y, suggesting that performance degrades as the motion increases. Faster speeds are therefore likely to increase the uncertainty of the measurement and increase error.

Figure 4.10 shows the average RMS reprojection error per landmark as a function of the number of inlier landmarks, showing the objective criteria used to evaluate motion performance in the RANSAC framework. A decreasing trend in RMS error is seen as inlier landmarks increase. $\sigma = [0.05, 0.1]$ showed the lowest reprojection errors and highest inlier count with minimal spread. Conversely, $\sigma = [0.2, 0.3]$ had the highest reprojection errors, and greatest inlier spread. In some cases the reprojected values were 2-3 times larger than their original input noise figures, suggesting even marginal misalignment produced large pose misalignment.

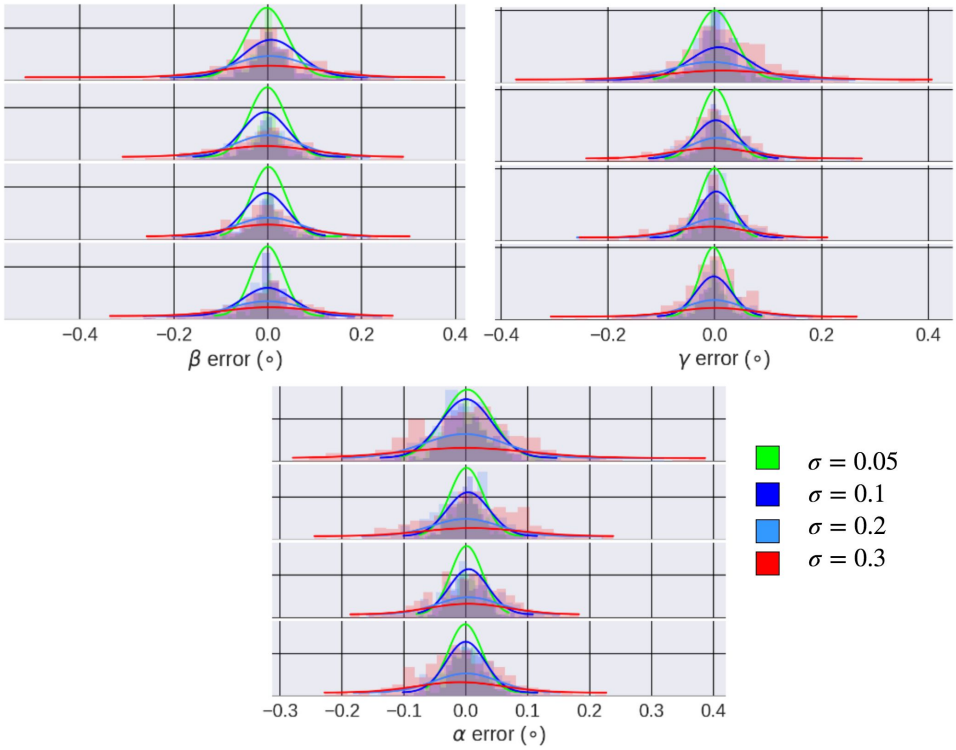


Figure 4.8: A histogram of the orientation error for different T_{track} values. Each motion parameter is stacked in ascending order from $T_{track} = 50$ to $T_{track} = 400$. The top most graph corresponds to $T_{track} = 50$, and the bottom $T_{track} = 400$. The Y-axis corresponds to the estimated PDF for each value of T_{track}

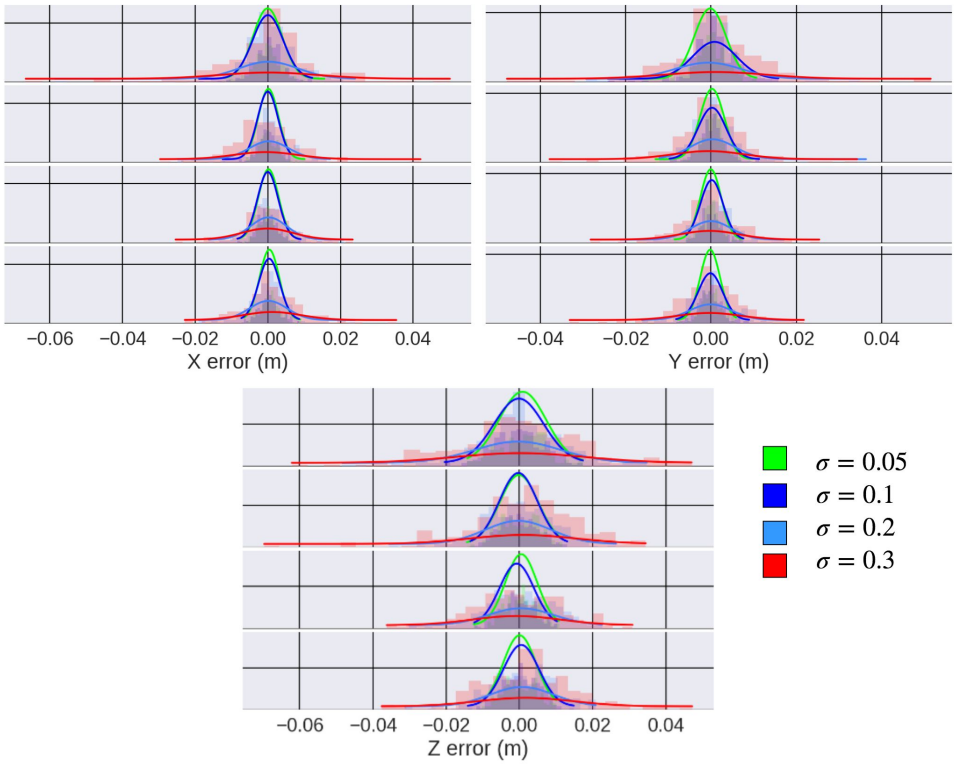


Figure 4.9: A histogram of the translation error for different T_{track} values. Each motion parameter is stacked in ascending order from $T_{track} = 50$ to $T_{track} = 400$. The Y-axis corresponds to the estimated PDF for each value of T_{track}

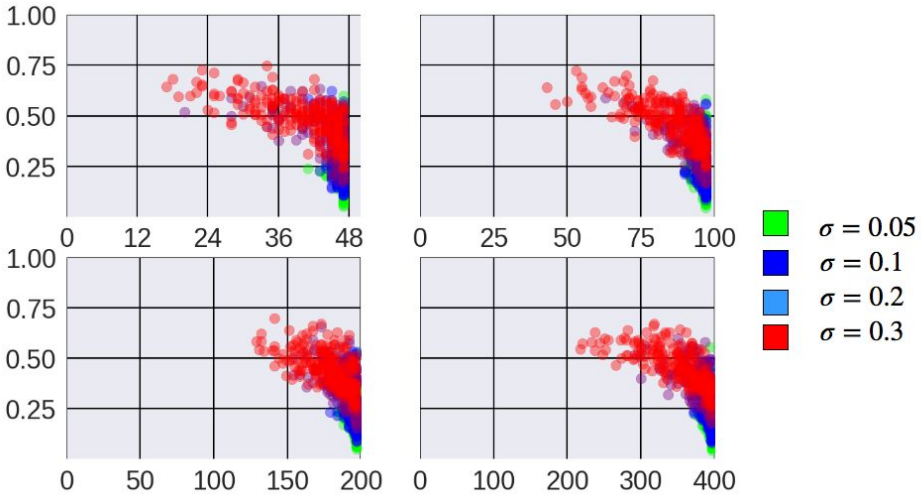


Figure 4.10: Scatter plots of simulated objective criteria for different T_{track} values. The X-axis is the number of inlier landmarks detected by the algorithm while the Y-axis is the average RMS reprojection error associated with the inlier landmarks.

4.4 Feature Extraction

Figure 4.11(a) shows the number of detections per frame for *Fast* and *Slow* loops. The black dotted lines indicate the target setpoints where $T_{Setpoint} =$

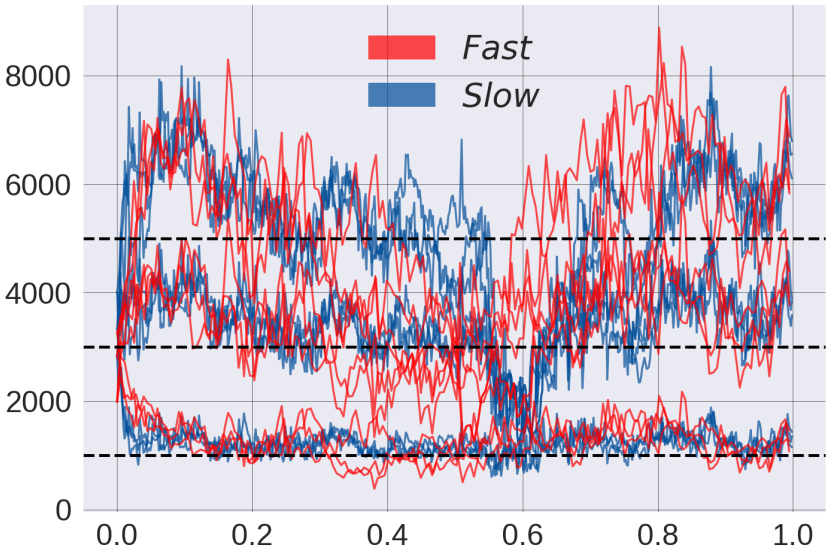
$[1000, 3000, 5000]$. $T_{Setpoint} = 1000$ displayed the most consistent setpoint tracking, whereas high setpoints overshot the setpoint in areas of high texture.

From time 0.5 to 0.8 there is a drastic drop in features extracted. Featureless regions such as the field and sky dominated the image in this portion of the loops. Lowering the saturation threshold T_{FAST}^{min} increased the control action available, but only contributed noise if increased excessively. For $T_{FAST}^{min} = 3$, grids containing plain areas such as the sky correctly saturated and did not introduce detections into the pipeline. T_{FAST}^{min} would have to be experimentally defined for a different environment.

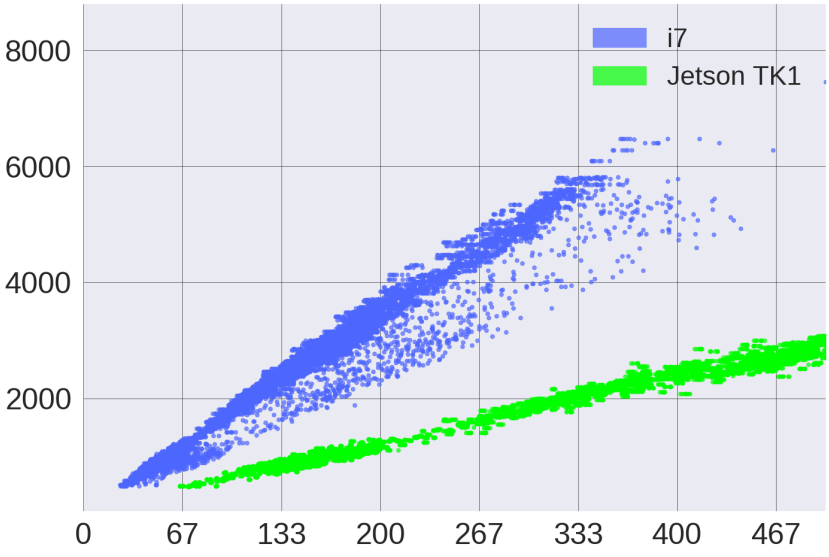
At 0.7, lens glare is present in the image which adversely affected the detector and reduced available features. Glare introduced light and dark regions into the image in such a way as to saturate the image. Saturation in these regions effectively eliminated discernible texture in darker regions, biasing the detector into finding matches predominantly in the light areas.

Fast loops showed much greater variance than their *Slow* counterparts. Larger inter-frame motion contributed to this as features entered and exited grids at a faster rate, meaning T_{FAST} could not react as quickly. A larger threshold step could increase the reaction speed, but also increase sensitivity to noise.

Figure 4.11(b) shows the total landmarks as a function of computational time for the i7 and Jetson TK1. Both versions show highly linear results, with the i7 containing only some outliers. An approximate linear fit shows on average the Jetson TK1 processes approximately 400 features per frame, while the i7 processes 1175.



(a)



(b)

Figure 4.11: A plot of total detections versus normalised time for *Slow* and *Fast* loops is shown in (a). Total detections versus computation time in milliseconds is shown in (b) for an i7 and Jetson TK1

4.5 Stereo Matching

Figure 4.12 shows the matching results of method 3.5.2 on A_3^{Slow}. There was good correlation observed between the detector features and the final stereo matches as they follow similar peaks and dips according to the detection setpoints presented in the previous section.

Figure 4.13(b) and 4.13(a) show the number of landmarks versus

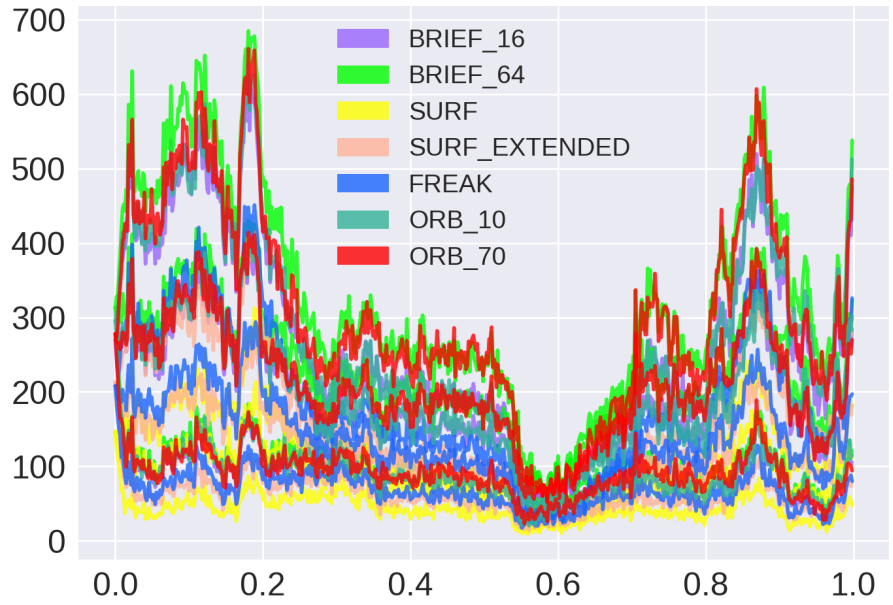


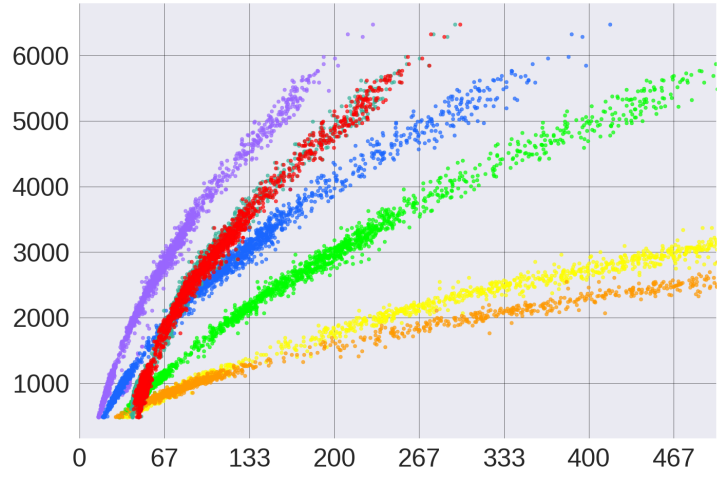
Figure 4.12: Total Stereo matches on A_3^{Slow} versus normalised time under different $\mathcal{D}$ operators and setpoints $T_{setpoint} = [1000, 3000, 5000]$

computation time on the i7 and Jetson TK1 respectively. Binary descriptors ORB, BRIEF and FREAK performed the fastest as expected, outperforming SURF for all but a small number of features.

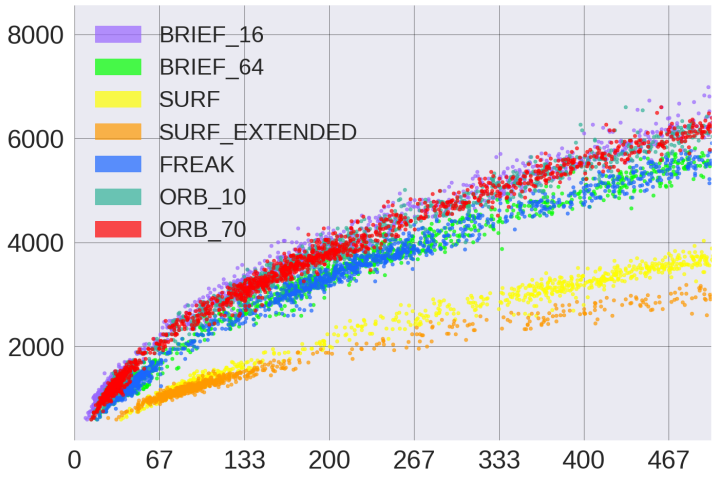
The ORB/BRIEF family of descriptors performed the best, generating the most matches per frame and being computationally efficient. This is inline with the work of [51] where ORB detectors and descriptors were used both for efficiency and effectiveness. The exception to this rule was the 64 byte BRIEF descriptor which took longer to compute than the 16 and 32 byte equivalents. The deviation in the ORB detector as a result of patch size suggests tuning would be necessary in different environmental conditions.

The invariant SURF descriptor performed poorly on all accounts. Using an extended descriptor increased matching performance at the cost of increased computation time, but detected approximately half of the ORB equivalent detections. The FREAK descriptor was a middle range performer, giving comparable results to an untuned ORB descriptor and processing faster than the 64 byte BRIEF descriptor.

Computation and matching time may appear linear at low landmark levels, but took on a logarithmic trend as landmark numbers increased. The logarithmic curve can be accounted for by the brute force matching scheme, steadily getting worse as the number of landmarks increases. The i7 displayed consistent execution whereas the Jetson TK1 showed more variability, illustrating the difference optimisation and processor architecture may have on an algorithm.



(a) Jetson TK1



(b) i7

Figure 4.13: Stereo descriptor and matching results for an i7 and Jetson TK1 showing total stereo landmarks versus processing time for different $\mathcal{D}$

4.6 Experiments

In this section, the results of the combined VCA and PE module are evaluated on experimental data. The relevant results for tracking,

parameter estimation, and the overall loop reconstruction are presented separately. $T_{setpoint} = 5000$ was used to ensure as many features were available for tracking as possible and limit motion failures. $\mathcal{D}$ was chosen as an ORB 70 descriptor as this gave good feature and processing results relative to the other descriptors. It was however unable to run in real-time.

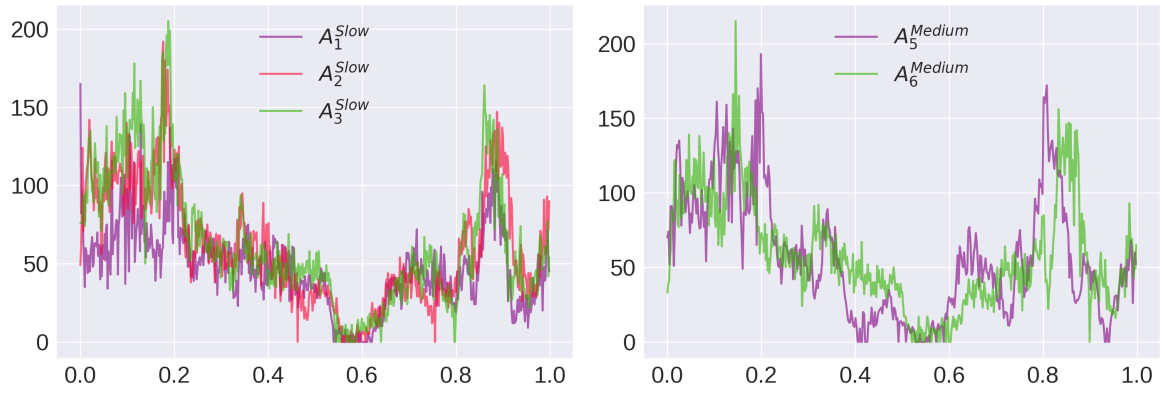
4.6.1 Tracking

Figure 4.14 shows the total inter-frame tracks per frame under different categories of speed. The net feature tracks were consistent over different speeds, peaking at 200 tracks but typically tracking less than 100. A 15-30% matching rate was observed between subsequent stereo frames, dropping to zero in region 0.3-0.6. This corresponds with the featureless region found in Section 4.4 and described in Section 3.3 during data collection, suggesting the low matching rate is a result of minimal stereo detections.

Higher speed increased the tracking failure in the 0.3-0.6 region and marginally affected it in other regions. Increased speed also impacted peaks of the tracking graphs. This is most likely an issue in the dataset collection itself, where the robot may have deviated in path speed or trajectory over short intervals.

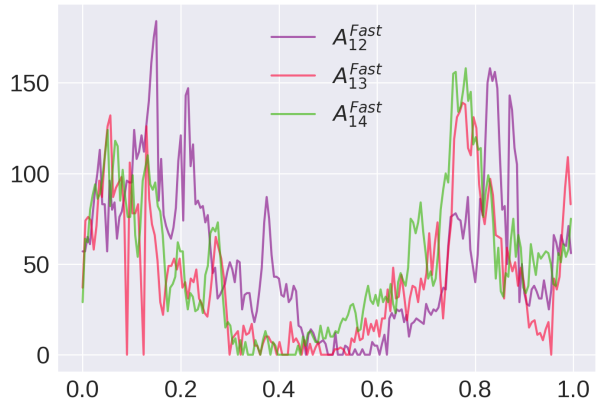
An example stereo feature tracking frame is shown in Figure 4.15(a), illustrating successful epipolar matching per stereo frame. Grid detection was effective at increasing the feature coverage, where landmarks were sufficiently spread over the image. Blatant outlier measurements were correctly removed from the initial matching set, showing the epipolar constraint and independent hypothesis and test scheme for each camera were effective at filtering false correspondences.

A stereoscopic view of the left camera is used to display both tracking,



(a) *Slow*

(b) *Medium*



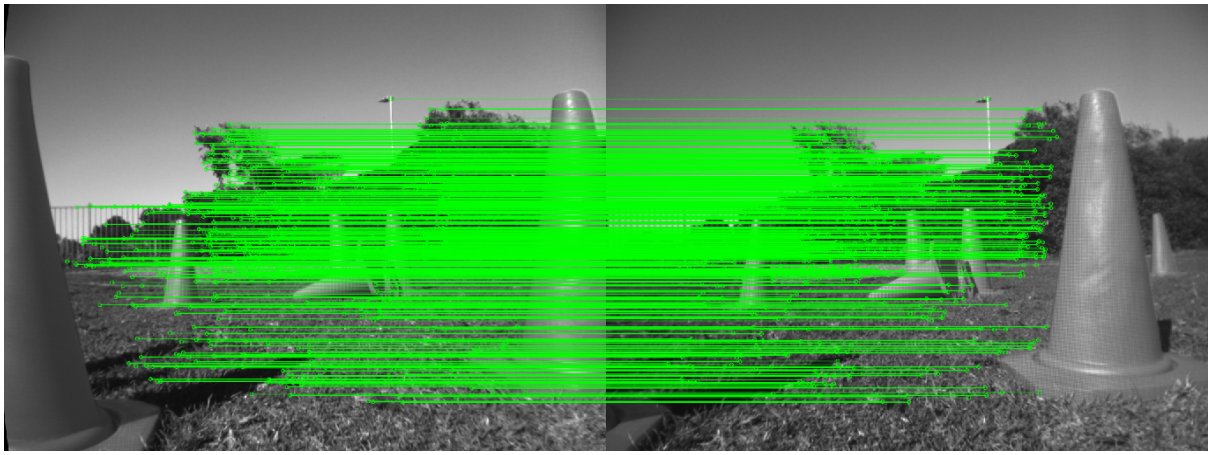
(c) *Fast*

Figure 4.14: A set of graphs displaying the total inter-frame tracks versus normalised loop time for *Slow*, *Medium*, and *Fast* speed categories.

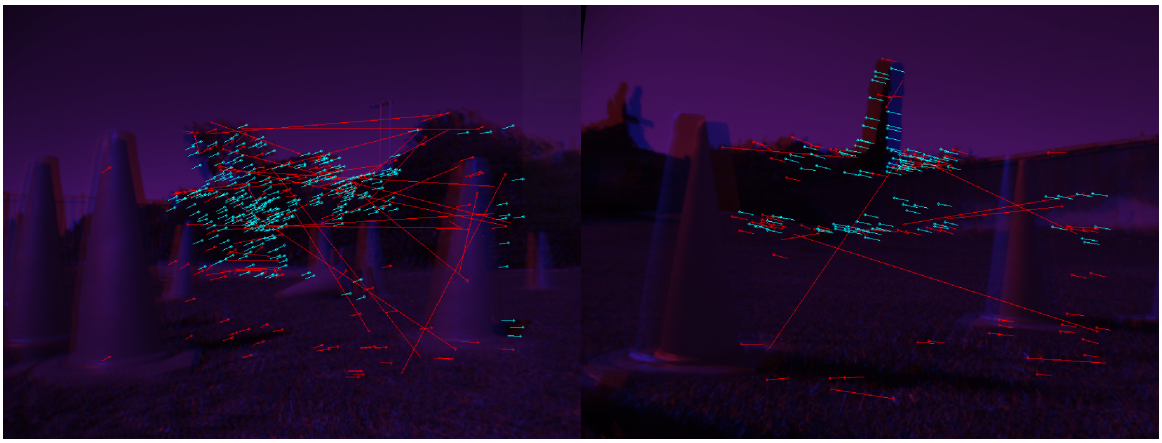
and movement between frames in Figure 4.15(b) at 0.2 and 0.7 for A_3^{Slow}. Red tracks indicate measurements that were flagged as outlier matches by the PE stage while blue lines show inlier tracks.

Inlier feature tracks were strongly biased to the top detection grid. Prominent tracks were found between object outlines and the sky, dense vegetation such as bushes, and structured man-made features. The image contrast in these regions caused strong corner responses and unique descriptors.

Conversely, grass regions contained extremely repetitive texture and features, making for many similar descriptors. The cones presented smooth features with a minimal corner response on their body.



(a) Stereo landmarks



(b) Inter-frame tracks

Figure 4.15: A visualisation of stereo landmark and inter-frame tracks. (a) displays horizontal epipolar lines between matches, and even coverage of the image. (b) Shows red outlier and blue inlier tracks from the PE stage on a stereoscopic image.

Matching across frames behaved differently depending on the grid row they were originally detected in. This is primarily as a result of FOV available to each detector grid. The ground grid predominantly imaged repetitive texture regions, whereas the sky grid imaged stark lines. As a result, tracked landmarks were not evenly covered on the image despite even stereo coverage. In this regard, the simple unsymmetrical detection method was not enough to ensure even coverage of tracks over the whole image.

Uneven track coverage had implications on landmark depth data, as tracks in the top grid were typically triangulated further than 5m away in the far-field, while ground grid tracks were constrained to the near-field. This

effectively limited the stereo point cloud accuracy as far-field landmarks tracked better than their near-field counterparts based on image texture.

4.6.2 Pose Estimation

Figure 4.16 shows the estimated α angle and inlier landmark RMS reprojection error versus forward motion as a scatter plot. The area between the vertical lines on each scatter plot indicates valid motion estimates based on forward motion and maximum speed assumptions from the dataset collection and simulation. Figure 4.17 illustrates the inlier versus outlier motion estimates in terms of the parameter estimation objective criteria.

α estimates correctly corresponded with anticipated corners on each loop. This heuristically indicates orientation estimation was in the correct direction with realistic size. *Slow* loops had better consensus where orientation peaks tightly coincided, whereas $Fast$ loops had greater dispersion. $Fast$ loops also contained larger, longer α steering angles to meet the cornering conditions at higher speeds.

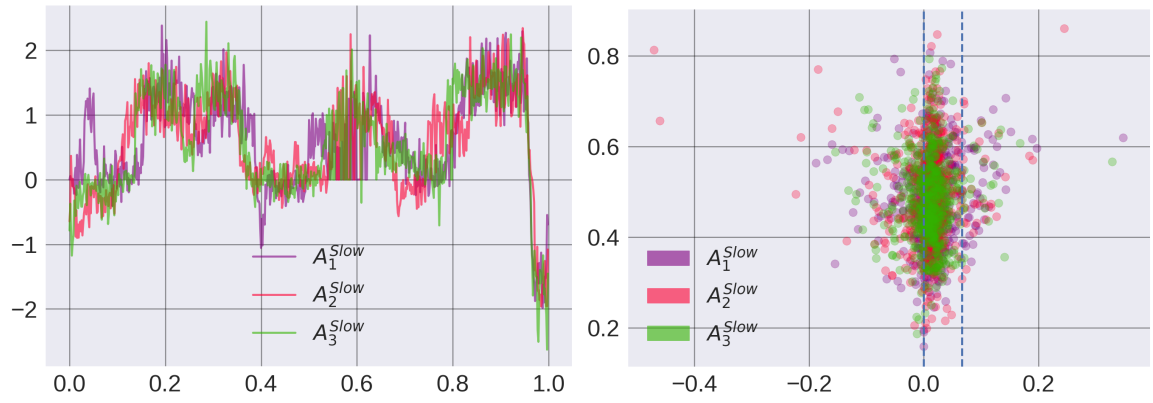
Translation estimation was far less consistent, showing a high level of outlier measurements. *Slow* and $Medium$ loop speeds illustrated similar average speeds indicated by the high number of measurements near the 0.5m/s mark. $Fast$ loops contained far more variance, and contained a high number of datapoints near the 1m/s speed threshold. This suggests that in some portions of the loop, the speed assumption may have been exceeded.

In comparison to the simulated curves in Figure 4.10, Figure 4.17 showed better and less dispersed objective criteria. The simulated curves were well constrained into the bottom right quadrant with $RMS < 0.5$ and inlier ratio > 0.5 under low Gaussian noise. Data on real experiments however

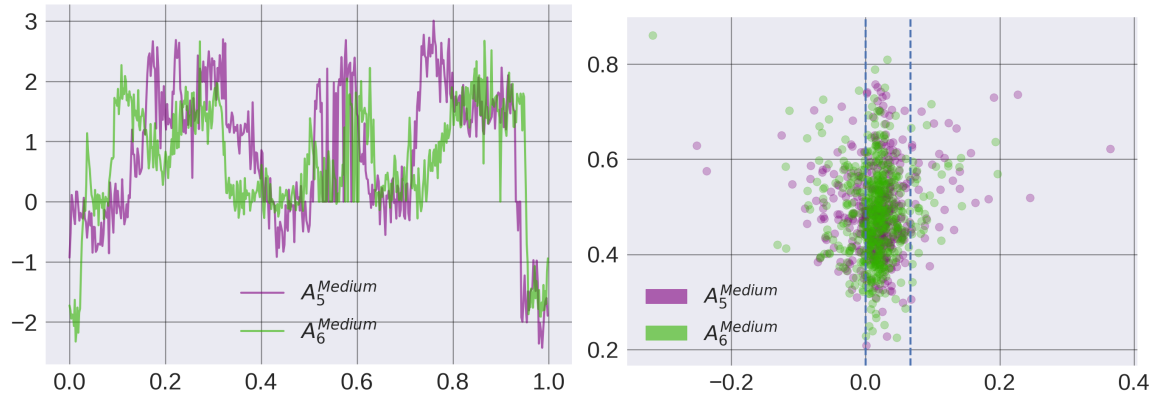
showed samples in all four quadrants, for both inlier and outlier motion estimates.

In this regard, no discernible difference between good and poor measurements could be made under the parameter estimation objective criteria. Simulation data suggested reprojection error and inlier ratio was a good indicator of measurement accuracy, but this is likely only true under low Gaussian noise assumptions. Figure 4.17 indicates that the average Gaussian error associated with the datasets may have exceeded the maximum simulated noise values.

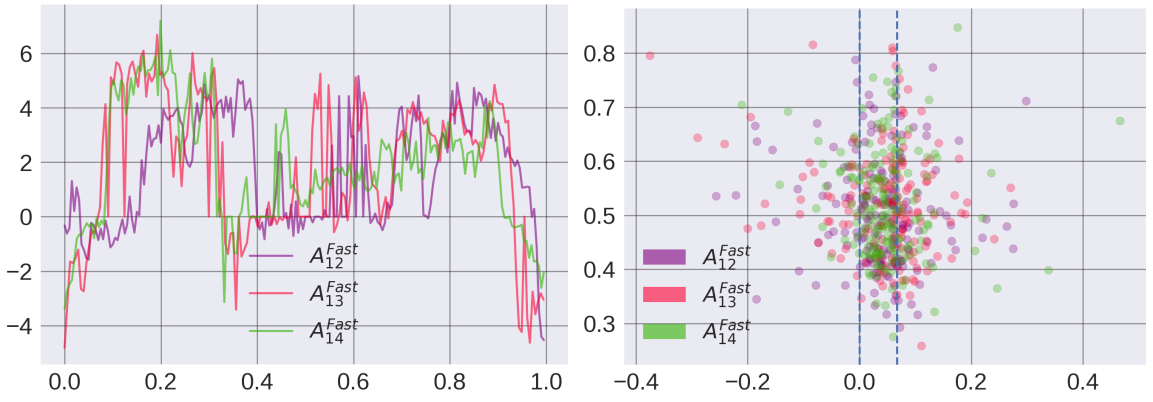
Sensitivity to Gaussian noise was highlighted as a shortcoming of the parameter estimation module, generating the high number of outlier measurements. The large distance to landmark tracks and low average landmarks per frame also contributed to outlier contamination in the pose estimates.



(a) *Slow*

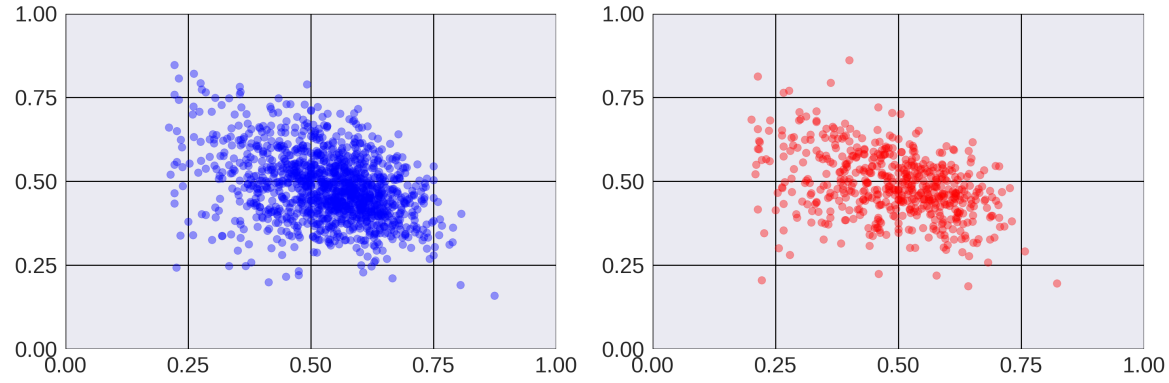


(b) *Medium*

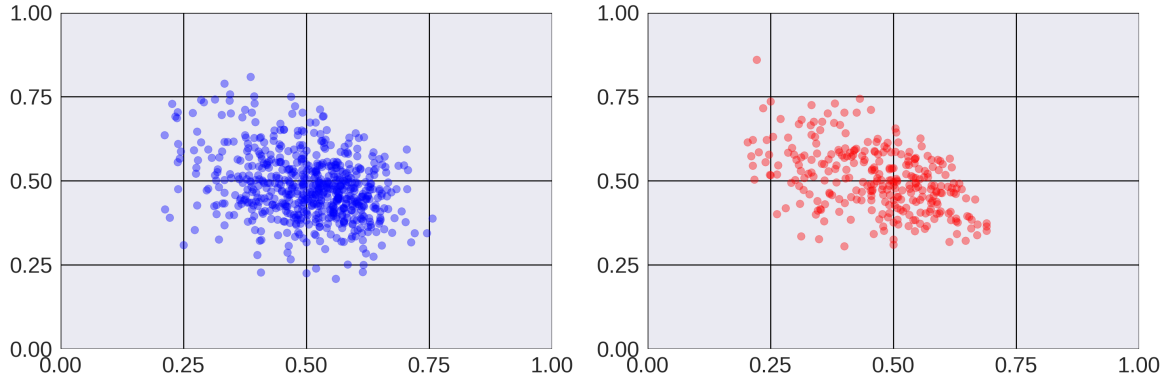


(c) *Fast*

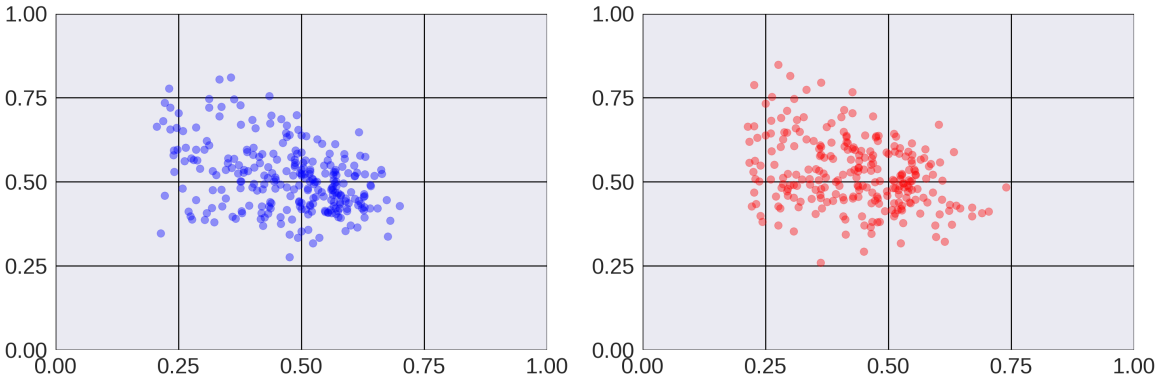
Figure 4.16: A set of α and forward motion estimates. On the left, α in degrees is plotted versus normalised dataset time. On the right, a scatter plot of average inlier reprojection error versus forward motion in metres. The horizontal blue lines indicate a region of inlier motion estimates where the upper forward motion was set as 1m/s.



(a) *Slow*



(b) *Medium*



(c) *Fast*

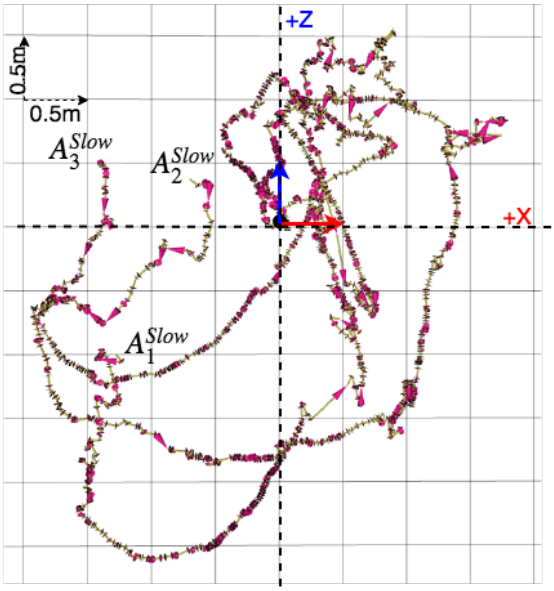
Figure 4.17: Scatter plots of PE objective criteria for inlier and outlier pose estimates in ascending order of speed. Blue and red dots indicate whether it was an inlier or outlier motion estimate respectively. The Y-axis shows average landmark reprojection error versus the inlier ratio on the X-axis.

4.6.3 Reconstruction

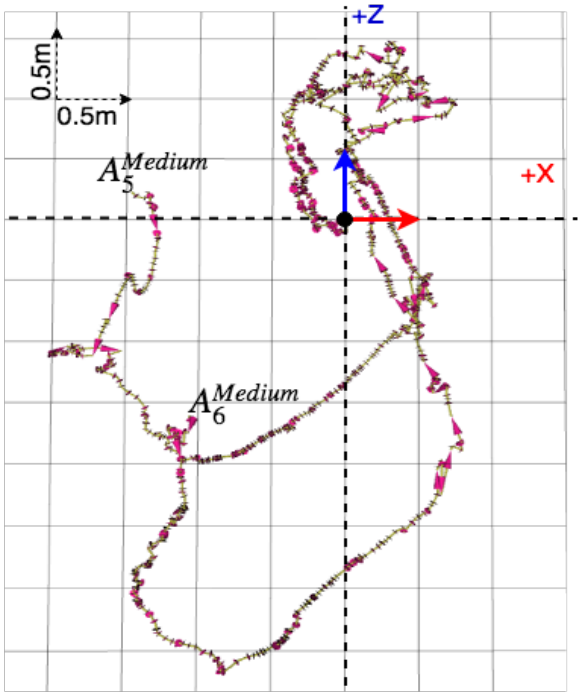
In this section, the output of the combined front-end and pose graph is visualised in terms of the overall pose chain and set of landmarks relative to coordinate frame $\mathcal{W}$. Figure 4.18 displays the set of superimposed pose chains for each category of speed.

Regions 0.2 to 0.6 displayed the most inconsistent results where the effect of outlier translation estimates is visible on the reconstruction accuracy. The effects of negative motion estimates were most clearly seen here where sudden translation jumps frequently occurred, coinciding with the featureless and low track regions. Despite the large outlier values present, the overall trajectory is still clearly visible, especially towards the latter part of the video sequence where smooth local trajectories are visible. The absolute distance between the end pose and origin describes odometer translation error and varied from 0.25m-3m.

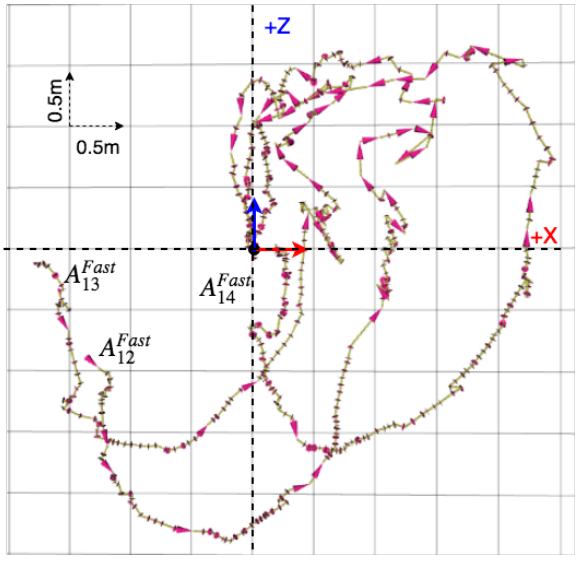
Figure 4.19 displays the set of stereo landmarks for the A_3^{Slow} loop where prominent features of the dataset are visible. Dense foliage at 0.2, rugby posts at 0.7, an embankment and wall at 0.8, and the ground plane were visible as shown by the yellow, green and pink arrows respectively. Local parameter estimation and landmark reconstruction appeared visually successful in populating a pose graph with coarse measurements that resembled the environment.



(a) *Slow*



(b) *Medium*



(c) *Fast*

Figure 4.18: Visualisations of the estimated pose chain for different speeds on a 0.5x0.5m grid. The red and blue arrows indicate the origin, while the pink arrows and yellow lines indicate the relative pose $\mathcal{T}_k^{k+1}$. The robot moved in a clockwise direction and the loop trajectory name is placed next to the final pose estimate. The difference between this final pose and the origin describes the odometer translation error

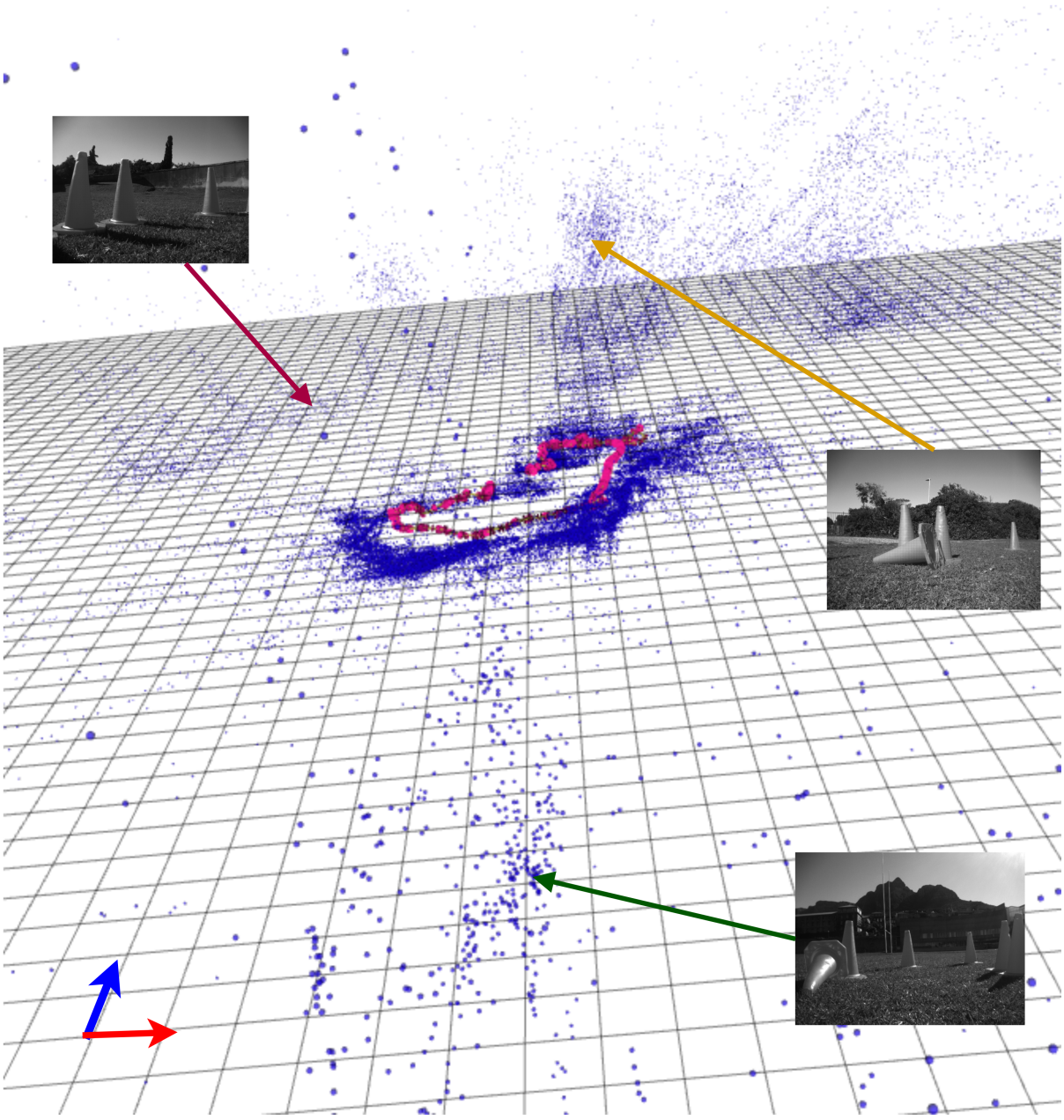


Figure 4.19: Visualisation of the trajectory and stereo landmarks as a point cloud. Some of the features in the dataset can be seen in the point cloud with their relevant camera image placed alongside for reference. The blue and red arrow indicate the direction of the Z and X axis respectively of the starting robot position.

Chapter 5

Discussion

In the preceding section, the results of a front-end feature tracking framework were presented in the context of a small-scale robot. Item One of the research questions, the evaluation and study of a front-end feature tracking framework, was fulfilled by quantifying the VCA, PE, and reconstruction performance of the proposed pipeline. In this chapter, Items Two and Three from the research questions will be addressed.

For Item Two, trade-offs of the proposed front-end were to be identified. A clear trade-off between robustness and processing speed exists in the formulated feature tracking framework. Robustness was proportional to the number of available tracked features, where additional features decreased the likelihood of outlier pose estimates. Setpoint versus processing time thus became a key design choice, where the number of features was limited by the environment, saturation threshold T_{FAST}^{min} and overall processing time required.

For outdoor conditions, a high setpoint was forced to take priority due to the presence of low inlier ratios versus inlier reprojection error and minimal tracked features. A lower setpoint would most likely reduce these tracks to zero, as seen from 0.2 to 0.6 in Figure 4.14, leading to complete motion failure on the majority of the dataset. If the overall matching rate across

frames could be improved, this trade-off would become less prominent. The number of detections required would decrease and less processing power would be wasted on outlier features.

For Item Three, the effectiveness of the front-end system was to be evaluated in the context of a small-scale experimental platform. The problem statement, formulated in Chapter one, asserted that the solution space of a small-scale robot using a feature tracking approach in an outdoor environment did not have a clear solution. A large body of possible interactions existed, but it was unclear how the small-scale constraint would influence the interaction between the hardware, environment and feature tracking algorithm. It was found that five factors were strongly influenced by the scale of the vehicle, and their effects on performance are discussed. Scale affected the camera FOV, suspension system, image distortions, processing power and effective baseline available.

The FOV of the camera had an important implication on tracking results, as the low height and forward facing camera strongly dictated the visible scene. The ground plane and skyline were prominent features on the lower and upper detection grids respectively. As a result, tracking coverage was unevenly distributed across the image, directly influencing the features available for pose estimation.

Motion blurring was a concern for a low (15) FPS camera that underwent motion disturbances as a result of speed or underactuated suspension systems. However, despite these factors being present in the system, motion blurring played a minimal role in pipeline performance. Due to the scale of the vehicle, speed is unlikely to increase dramatically higher than 1m/s without significant investment in hardware. The underactuated suspension, rough terrain and locomotion of the robot acted like a natural speed limitation. In this regard, motion blurring proved to be negligible in this context.

Lighting conditions played a much larger role than the literature indicated, challenging the Motion Distortion Assumption made. Lens glare saturated the image in portions of the loop, reducing the number of matches in these regions. Additionally, driving towards the sun simultaneously introduced bright and dark image regions. This combination proved to be challenging for the camera sensor itself, often limiting detection to the brightest regions and ignoring darker ones. A high dynamic image range would thus be considered more beneficial than a high FPS for small-scale robots in the outdoors.

Scale reduced the effective payload available to the robot, limiting processing power available. Processing speed is a clear factor to improve in feature tracking approaches. Using optimised libraries targeted at the processor architecture, stereo rectification, detection and matching time greatly exceeded the 66ms requirement for real-time operation. With the minimum setpoint in this study of 1000 detections, the total processing time required would be 230ms on the Jetson TK1, taking almost four times longer to compute than the FPS of the system. Two approaches could be used to address the processing speed, namely hardware or software.

From a hardware perspective, dedicated modules could be designed that use smaller, more powerful processors and peripherals such as FPGAs or GPUs. This is the simplest approach to improve performance, however it is not necessarily the most efficient. Adding additional hardware can increase the overall payload/weight, consume more power, and increase development time and cost.

From a software perspective, processing speed could be increased by using a smaller ROI, or utilising the already existing GPU for rectification and detection where applicable. With the limited on-board GPU, careful algorithm selection would be required as memory transfer costs would need to be factored into the effectiveness of the peripheral. Reducing the number of features and ROI would greatly benefit the computation time,

but would clearly impact robustness of the algorithm. A smaller ROI would reduce the FOV, further reducing tracking ability and making the ROI an important tuning parameter of the pipeline.

Vehicle scale affected the available baseline of the system, an important design choice for stereo vSLAM, as this directly impacted point cloud accuracy. Due to the dominant distant feature tracking, most measurements effectively degraded into pure monocular measurements. This meant the tight Gaussian noise requirement necessary for accurate parameter estimation was not valid in this context, reducing robustness and accuracy of the translation estimate.

Stereo point cloud accuracy could be improved with better calibration algorithms for nearby landmarks, but would have minimal effect on distant ones. For this reason minimising calibration error would only marginally improve results. A mixed mode landmark parameterisation as in [86] would seem to be a necessary requirement in this context, as the near and far-field landmarks played a large role in tracking performance. The uneven tracking coverage also suggests that a mixed mode detection scheme would be beneficial, where far and near-field landmarks use a different parameterisation and detection mechanism.

Stereo vision in this context was found to be a simple and effective robustness constraint, but critically suffered in reconstruction performance. Directly using the point cloud for pose estimation meant Gaussian noise reduced accuracy. Reconstruction performance would likely increase post optimisation with BA or sliding window BA, but it is outside the scope of this work. Other relative pose estimators such as those discussed in [35, 41, 108] could be substituted into the pipeline as alternatives. However, metric reconstruction would still be strongly determined by the baseline, tracked features and processing power available, all factors strongly related to the scale of the robot.

The front-end feature tracking framework was successfully implemented in this context, populating a vSLAM pose graph with coarse measurements and reconstructing the robot trajectory. However, the scale of the vehicle introduced constraints on the system that reduced metric reconstruction performance and real-time capabilities. In this regard the system functionally performed the requirements stipulated, but would likely improve with additional hardware. In this context, additional sensors should be a requirement so that multiple motion hypothesis are available. This would have greatly increased reconstruction performance, as any motion failure conditions could be supplemented with odometer readings or IMU measurements for that particular pose edge. Sensor fusion would be strongly recommended where multiple pose estimates over a single frame could be combined and filtered into a single estimate.

Chapter 6

Conclusion

In this body of work, a front-end feature tracking framework was described and evaluated for a front-end vSLAM system. The framework was tested in the context of a small-scale (less than $1m^3$), wheeled robotic platform operating in the outdoors. Three outcomes of the research were the evaluation and study of a front-end feature tracking framework, trade-off identification and the evaluation of the effectiveness of the framework in the context of a small-scale experimental platform. These items will be discussed individually in this chapter, followed by recommendations and possible future studies.

6.1 Evaluation of a Front-End Feature Tracking Framework

In this study, the evaluation of a front-end feature tracking framework was performed on a real dataset and through a simulation. A Visual Cue Acquisition and Parameter Estimation module was outlined and used to populate a vSLAM pose graph system. Performance of the VCA module was quantified in terms of total stereo landmarks and processing time. A PE module was evaluated in simulation and experimentally in terms of pose accuracy and inter-frame tracks. Lastly, the combined VCA and PE modules were evaluated in terms of overall odometer error under different levels of speed.

A calibration procedure was detailed for the Bumblebee (Discontinued) stereo camera with 120mm baseline, where the final stereo RMS calibration error evaluated to 0.27 pixels. The average rectification time took 16 and 32ms respectively for an i7 laptop and Jetson TK1.

A FAST setpoint tracking approach was found to be a simple and effective detection method for generating initial features. The Jetson TK1 could detect approximately 400 features per frame. With 1000 initial detections, the total processing time required would be 230ms on the Jetson TK1, taking almost four times longer to compute than the FPS of the system.

ORB and BRIEF descriptors were found to produce the most stereo features in the least time. The FREAK descriptor was marginally slower than BRIEF and ORB, but produced far less matches. SURF was found to produce the least matches with the longest computational time.

Using an ORB descriptor for the PE module, an average of 100 tracks per frame or 15-30% matching rate was achieved. However, due to the low number of tracks, there were many regions in which motion estimation failed completely. It was also found that tracked features were typically found in the top image region. The repetitive nature of the grass and vegetation in the bottom image caused feature tracking to fail, due to non-unique descriptors.

Motion estimation was evaluated using a simulation-based approach, where it was found that direct use of the stereo point cloud for motion estimation was highly sensitive to Gaussian noise. Trajectory reconstruction yielded approximately 0.25-3m odometer error on the outdoor sequence.

6.2 Trade-off Identification

A strong trade-off was found between the processing speed and robustness in the stereo feature tracking framework. To achieve robust tracking, a

high number of detections was required, meaning processing speed was a limiting factor to the feature tracking approach.

6.3 Front-end Effectiveness and Evaluation on a Small-scale Experimental Platform

The problem statement asserted that context plays an important role in vSLAM performance. It was found in this work that the context did play an important role, particularly in regard to the scale of the vehicle. Figure 1.4 provided an overview of potential factors affecting the solution space of a vSLAM system. Figure 6.1 summarises the key factors found in this study related to the context of a small-scale outdoor wheeled robot, implementing a stereo feature tracking framework.

A small scale affected the FOV of the images, constraining what features were tracked using standard matching algorithms. A small baseline reduced pose estimation accuracy, a shortcoming from directly using the stereo point cloud to estimate motion. Motion blurring was found to be negligible in this context, due to the low speed imposed by the environmental conditions and actuation limitations.

Lighting played a significant role in the pipeline, as image saturation reduced feature tracking effectiveness. A high dynamic range camera would be highly beneficial to a small-scale outdoor robot.

The front-end feature tracking framework was successfully implemented in this context, populating a vSLAM pose graph with coarse measurements and reconstructing the robot trajectory. However, the scale of the vehicle introduced constraints on the system, which reduced metric reconstruction performance and real-time capabilities. A stereo feature tracking framework provided a simple and effective robustness constraint, but critically failed to meet real-time performance. Metric reconstruction was limited by the imposed baseline.

In this regard the system functionally met the requirements stipulated, but would likely improve with additional hardware. In this context, additional sensors should be a requirement so that multiple motion hypotheses are available. This would have greatly increased reconstruction performance, as any motion failure conditions could be supplemented with odometer readings or IMU measurements for that particular pose edge. Sensor fusion would be strongly recommended where multiple pose estimates over a single frame could be combined and filtered into a single estimate.

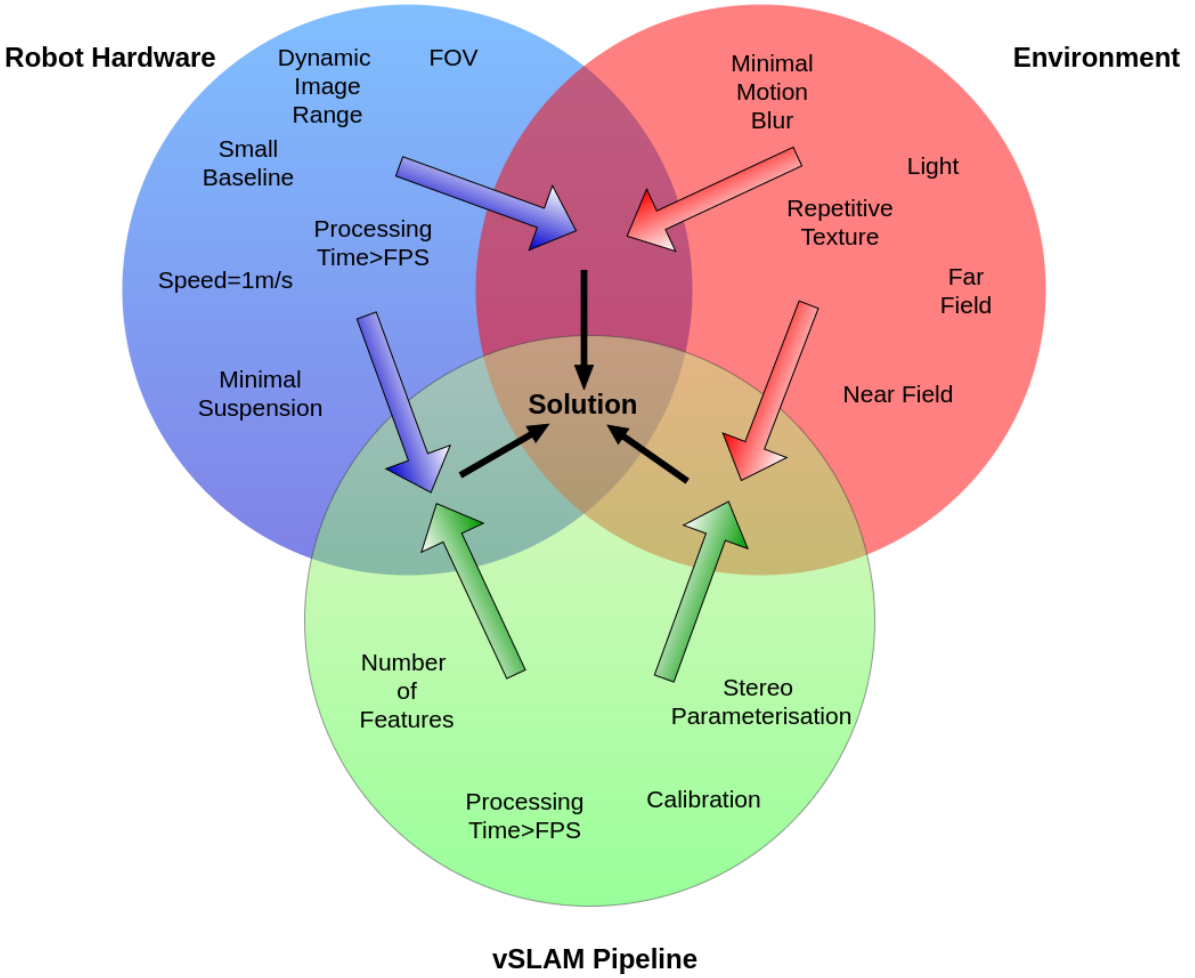


Figure 6.1: An illustration of the solution space where the most relevant factors and constraints found in the study are grouped in their appropriate category

6.4 Recommendations and Future Work

The feature tracking framework was successfully implemented in this context, but would greatly benefit from the following improvements

- **Mixed Mode Parameterisation**: Due to uneven feature tracking coverage and small baseline, many stereo measurements degraded into monocular measurements. Being able to populate the pose graph with both forms of information would improve robustness and allow appropriate pose estimation based on the available measurements.
- **Improved/Dedicated hardware**: Processing speed was an area of concern for the pipeline where real-time performance could not be met. This could likely be improved by introducing dedicated electronics for the VCA module.
- **Mixed Mode Detection and Matching**: While the VCA module was successful in populating stereo constraints, the inter-frame tracking module did not evenly cover the image. A split detection and matching scheme could improve matching in the ground plane, where tracking was less successful due to repetitive texture.
- **Sensor Fusion**: Due to the high possibility of outlier pose estimates, additional sensors would greatly increase the reconstruction accuracy and robustness of pose estimation.
- **High Dynamic Range**: A high dynamic range camera would likely improve results by being able to minimise image saturation.

For future work, the mixed mode parameterisation/detection, combined with sensor fusion, would likely be the best manner to address the problems introduced by scale. Additional sensors would aid the front-end system in harsh visual scenes, while multiple landmark representations would cater for the small baseline and potentially introduce a second source of verification. If these methods successfully improve the accuracy and

robustness, the algorithms can be targeted towards specific processors and hardware to maximise the speed of implementation.

Appendices

Appendix A - Intrinsic Parameters

Table 1: A table of the extracted intrinsic parameters and statistics from the calibration procedures

Name	Cal_A	Cal_B	Cal_C
l_{rms}	0.817	0.869	0.907
r_{rms}	0.821	0.852	0.885
in_{rms}	0.719	0.703	0.730
s_{rms}	0.330	0.270	0.280
L_{ROI}	[71 37 930 686]	[70 30 933 692]	[57 27 958 707]
R_{ROI}	[41 53 898 672]	[19 38 942 704]	[16 35 947 705]
ROI	[71 53 868 670]	[70 38 891 687]	[57 35 906 699]
$ROI_{\%}$	73.95	77.83	80.53
P_l	$\begin{bmatrix} 818.0471 & 0 & 493.8238 & 0 \\ 0 & 818.0471 & 383.4677 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 801.9989 & 0 & 505.3793 & 0 \\ 0 & 801.9989 & 383.3668 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 793.1922 & 0 & 503.715 & 0 \\ 0 & 793.1922 & 383.4072 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
P_r	$\begin{bmatrix} 818.0471 & 0 & 493.8238 & -98.5499 \\ 0 & 818.0471 & 383.467 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 801.9989 & 0 & 505.3793 & -96.353 \\ 0 & 801.9989 & 383.3668 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 793.1922 & 0 & 503.715 & -95.0303 \\ 0 & 793.1922 & 383.4072 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
K_l	$\begin{bmatrix} 925.8863 & 0 & 511.9056 \\ 0 & 922.3966 & 384.4836 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 920.1314 & 0 & 511.9083 \\ 0 & 916.2982 & 384.4698 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 914.3742 & 0 & 512.0757 \\ 0 & 909.6643 & 384.4323 \\ 0 & 0 & 1 \end{bmatrix}$
K_r	$\begin{bmatrix} 925.8863 & 0 & 509.5372 \\ 0 & 922.3966 & 384.6206 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 920.1314 & 0 & 509.8311 \\ 0 & 916.2982 & 384.5295 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 914.3742 & 0 & 510.415 \\ 0 & 909.6643 & 384.3503 \\ 0 & 0 & 1 \end{bmatrix}$
D_l	$\begin{bmatrix} -2.6841e-01 \\ -2.6961e-01 \\ -8.7116e-04 \\ 2.7309e-04 \\ 1.3670e+00 \end{bmatrix}^T$	$\begin{bmatrix} -2.7192e-01 \\ -1.7940e-01 \\ -8.7786e-04 \\ -6.8571e-05 \\ 1.0031e+00 \end{bmatrix}^T$	$\begin{bmatrix} -2.8224e-01 \\ -4.0973e-02 \\ -6.3434e-04 \\ -5.1785e-04 \\ 5.3225e-01 \end{bmatrix}^T$
D_r	$\begin{bmatrix} -2.4935e-01 \\ -5.1255e-01 \\ -2.5266e-03 \\ -9.8753e-04 \\ 2.1620e+00 \end{bmatrix}^T$	$\begin{bmatrix} -0.2656 \\ -0.2137 \\ -0.0023 \\ -0.0012 \\ 0.9171 \end{bmatrix}^T$	$\begin{bmatrix} -0.2665 \\ -0.1703 \\ -0.002 \\ -0.0017 \\ 0.7544 \end{bmatrix}^T$
Q	$\begin{bmatrix} 1 & 0 & 0 & -493.8238 \\ 0 & 1 & 0 & -383.4677 \\ 0 & 0 & 0 & 818.0471 \\ 0 & 0 & 8.3008 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & -505.3793 \\ 0 & 1 & 0 & -383.3668 \\ 0 & 0 & 0 & 801.9989 \\ 0 & 0 & 8.3236 & 0.000 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & -503.715 \\ 0 & 1 & 0 & -383.4072 \\ 0 & 0 & 0 & 793.1922 \\ 0 & 0 & 8.3467 & 0 \end{bmatrix}$

Appendix B - Extrinsic Parameters

Table 2: A table of the extracted extrinsic parameters and statistics from calibration

Name	Cal_A	Cal_B	Cal_C
R	$\begin{bmatrix} 0.9989 & 0.0053 & 0.0469 \\ -0.005 & 1 & -0.0061 \\ -0.0469 & 0.0059 & 0.9989 \end{bmatrix}$	$\begin{bmatrix} 0.9989 & 0.0053 & 0.04606 \\ -0.005 & 1 & -0.0061 \\ -0.466 & 0.0058 & 0.9989 \end{bmatrix}$	$\begin{bmatrix} 0.9989 & 0.0053 & 0.0463 \\ -0.0051 & 1 & -0.0059 \\ -0.0463 & 0.0057 & 0.9989 \end{bmatrix}$
T	$\begin{bmatrix} -0.1205 \\ 0.0011 \\ 0.0009 \end{bmatrix}$	$\begin{bmatrix} -0.1201 \\ 0.001 \\ 0.0022 \end{bmatrix}$	$\begin{bmatrix} -0.1198 \\ 0.001 \\ 0.002 \end{bmatrix}$
E	$\begin{bmatrix} -4.5165e-05 & -8.8210e-04 & 1.0619e-03 \\ -4.7622e-03 & 7.1173e-04 & 1.2037e-01 \\ -4.5435e-04 & -1.2046e-01 & 6.8651e-04 \end{bmatrix}$	$\begin{bmatrix} -3.7814e-05 & -2.1984e-03 & 1.0607e-03 \\ -3.4012e-03 & 7.1216e-04 & 1.2009e-01 \\ -4.4270e-04 & -1.2012e-01 & 6.8056e-04 \end{bmatrix}$	$\begin{bmatrix} -3.6167e-05 & -2.0246e-03 & 1.0133e-03 \\ -3.5226e-03 & 6.9317e-04 & 1.1975e-01 \\ -3.9468e-04 & -1.1979e-01 & 6.6479e-04 \end{bmatrix}$
F	$\begin{bmatrix} 6.4618e-08 & 1.2668e-06 & -1.9269e-03 \\ 6.8392e-06 & -1.0260e-06 & -1.6316e-01 \\ -2.0616e-03 & 1.5993e-01 & 1.0000e+00 \end{bmatrix}$	$\begin{bmatrix} 5.2746e-08 & 3.0794e-06 & -2.5723e-03 \\ 4.7640e-06 & -1.0017e-06 & -1.5682e-01 \\ -1.2906e-03 & 1.5363e-01 & 1.0000e+00 \end{bmatrix}$	$\begin{bmatrix} -3.6167e-05 & -2.0246e-03 & 1.0133e-03 \\ -3.5226e-03 & 6.9317e-04 & 1.1975e-01 \\ -3.9468e-04 & -1.1979e-01 & 6.6479e-04 \end{bmatrix}$
$K2K_l^{\mathcal{R}_l}$	$\begin{bmatrix} 0.9992 & -0.0035 & 0.0396 & 0 \\ 0.0037 & 1 & -0.0028 & 0 \\ -0.0395 & 0.003 & 0.9992 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.9996 & -0.0035 & 0.0283 & 0 \\ 0.0036 & 1 & -0.0028 & 0 \\ -0.0283 & 0.0029 & 0.9996 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.9996 & -0.0031 & 0.0294 & 0 \\ 0.0032 & 1 & -0.0028 & 0 \\ -0.0294 & 0.0029 & 0.9996 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
$K2K_r^{\mathcal{R}_r}$	$\begin{bmatrix} 0.9999 & -0.0088 & -0.0074 & 0 \\ 0.0088 & 1 & 0.0029 & 0 \\ 0.0073 & -0.0029 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.9998 & -0.0087 & -0.0184 & 0 \\ 0.0088 & 1 & 0.0028 & 0 \\ 00183 & -0.003 & 0.9998 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.9998 & -0.0084 & -0.0169 & 0 \\ 0.0084 & 1 & 0.0028 & 0 \\ 0.0169 & -0.0029 & 0.9999 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
$K2K_l^r$	$\begin{bmatrix} 9.9889e-01 & 5.2799e-03 & 4.6869e-02 & -1.2046e-01 \\ -4.9989e-03 & 9.9997e-01 & -6.1105e-03 & 1.0577e-03 \\ -4.6899e-02 & 5.8694e-03 & 9.9888e-01 & 8.8834e-04] \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.9989 & 0.0053 & 0.0466 & -0.1201 \\ -0.005 & 1 & -0.0061 & 0.001 \\ -0.0466 & 0.0058 & 0.9989 & 0.0022] \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0.9989 & 0.0053 & 0.0463 & -0.1198 \\ -0.0051 & 1 & -0.0059 & 0.001 \\ -0.0463 & 0.0057 & 0.9989 & 0.002 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
$K2K_{\mathcal{R}_l}^{\mathcal{R}_r}$	$\begin{bmatrix} 1.0000e+00 & 9.3675e-17 & 8.9512e-16 & -1.2047e-01 \\ -1.0650e-16 & 1.0000e+00 & -1.0322e-16 & -5.7946e-18 \\ -8.9512e-16 & 1.1232e-16 & 1.0000e+00 & 1.4962e-17 \\ 0.0000e+00 & 0.0000e+00 & 0.0000e+00 & 1.0000e+00 \end{bmatrix}$	$\begin{bmatrix} 1.0000e+00 & -1.8133e-17 & -1.8388e-16 & 1.2014e-01 \\ 2.2240e-17 & 1.0000e+00 & 2.0383e-17 & 5.5139e-16 \\ 1.9082e-16 & -2.2551e-17 & 1.0000e+00 & -4.4712e-16 \\ 0.0000e+00 & 0.0000e+00 & 0.0000e+00 & 1.0000e+00 \end{bmatrix}$	$\begin{bmatrix} 1.0000e+00 & 2.4507e-16 & 2.1580e-15 & -1.1981e-01 \\ -2.4213e-16 & 1.0000e+00 & -2.5804e-16 & 2.1318e-16 \\ -2.1649e-15 & 2.5761e-16 & 1.0000e+00 & -2.2898e-16 \\ 0.0000e+00 & 0.0000e+00 & 0.0000e+00 & 1.0000e+00 \end{bmatrix}$

Appendix C - Dataset Loop Definitions

Table 3: Dataset loop closure image indexes for dataset A

Loop Number	**Starting Frame**	**Ending Frame**
1	1097	1735
2	1736	2365
3	2366	2975
4	2976	3581
5	3582	4019
6	4020	4564
7	4565	5144
8	5145	5713
9	5714	5915
10	5916	6136
11	6137	6358
12	6359	6559
13	6560	6737
14	6738	6934

Appendix D - Settings and Parameters

Table 4: A table of settings and parameters used for the simulation and dataset A experiments

Parameter Name	Value
T^{max}_{RANSAC}	400
T^{inlier}_{RANSAC}	1.0
T^{min}_{RANSAC}	0.1
T^{RMS}_{RANSAC}	0.2
T^{min}_{FAST}	3
N_r	2
N_c	3
$T_{deadband}$	0.2
$T^{Initial}_{FAST}$	10
$T_{stereoInlier}$	0.7

Bibliography

[1] T. Sattler, B. Leibe, and L. Kobbelt, "SCRAMSAC: Improving RANSAC's efficiency with a spatial consistency filter," in *2009 IEEE 12th International Conference on Computer Vision*, Sep. 2009, pp. 2090–2097.

[2] A. Saffiotti, "The uses of fuzzy logic in autonomous robot navigation," *Soft Computing*, vol. 1, no. 4, pp. 180–197, Dec. 1997. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/article/10.1007/s005000050020

[3] S. Park and S. Hashimoto, "Autonomous Mobile Robot Navigation Using Passive RFID in Indoor Environment," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 7, pp. 2366–2373, Jul. 2009.

[4] I. J. Cox, "Blanche-an experiment in guidance and navigation of an autonomous robot vehicle," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 2, pp. 193–204, Apr. 1991.

[5] D. Murray and J. J. Little, "Using Real-Time Stereo Vision for Mobile Robot Navigation," *Autonomous Robots*, vol. 8, no. 2, pp. 161–171, Apr. 2000. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/article/10.1023/A:1008987612352

[6] J. Biswas and M. Veloso, "WiFi localization and navigation for autonomous indoor mobile robots," in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 4379–4384.

[7] D. Nakhaeinia, S. H. Tang, S. M. Noor, and O. Motlagh, "A review of control architectures for autonomous navigation of mobile robots," *International Journal of Physical Sciences*, vol. 6, no. 2, pp. 169–174, 2011. [Online]. Available: http://www.academicjournals.org/journal/IJPS/article-abstract/30A542F19531

[8] D. Fox, W. Burgard, and S. Thrun, "Active markov localization for mobile robots," *Robotics and Autonomous Systems*, vol. 25, no. 3-4, pp. 195–207, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889098000499

[9] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *Journal of intelligent and robotic systems*, vol. 53, no. 3, p. 263, 2008. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/article/10.1007/s10846-008-9235-4

[10] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," *IEEE Robotics Automation Magazine*, vol. 13, no. 3, pp. 108–117, Sep. 2006.

[11] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/1638022/

[12] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendn-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, Jan. 2015. [Online]. Available: http://link.springer.com/10.1007/s10462-012-9365-8

[13] G. Lakemeyer and B. Nebel, Eds., *Exploring artificial intelligence in the new millennium*. San Diego, CA: Morgan Kaufmann Publishers, 2003.

[14] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0," *The International Journal of Robotics*

Research, vol. 30, no. 9, pp. 1100–1123, Aug. 2011. [Online]. Available: http://dx.doi.org/10.1177/0278364910385483

[15] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012. [Online]. Available: http://journals.sagepub.com.ezproxy.uct.ac.za/doi/abs/10.1177/0278364911430419

[16] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 155–160. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6106777/

[17] X. Zhang, A. B. Rad, and Y.-K. Wong, "Sensor Fusion of Monocular Cameras and Laser Rangefinders for Line-Based Simultaneous Localization and Mapping (SLAM) Tasks in Autonomous Mobile Robots," *Sensors*, vol. 12, no. 1, pp. 429–452, Jan. 2012. [Online]. Available: http://www.mdpi.com/1424-8220/12/1/429

[18] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visualinertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015. [Online]. Available: http://journals.sagepub.com.ezproxy.uct.ac.za/doi/abs/10.1177/0278364914554813

[19] T. Lemaire, C. Berger, I.-K. Jung, and S. Lacroix, "Vision-based slam: Stereo and monocular approaches," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 343–364, 2007. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/article/10.1007/s11263-007-0042-3

[20] A. Stentz, J. Bares, T. Pilarski, and D. Stager, "The crusher system for autonomous navigation," *AUVSIs Unmanned*

Systems North America, vol. 3, 2007. [Online]. Available: http://www.frc.ri.cmu.edu/users/axs/doc/auvsi07.pdf

[21] A. E. Johnson, S. B. Goldberg, Y. Cheng, and L. H. Matthies, "Robust and efficient stereo feature tracking for visual odometry," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 39–46. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/4543184/

[22] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A system for autonomous flight using onboard computer vision," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 2992–2997.

[23] G. Ros, A. Sappa, D. Ponsa, and A. M. Lopez, "Visual slam for driverless cars: A brief survey," in *Intelligent Vehicles Symposium (IV) Workshops*, vol. 2, 2012. [Online]. Available: http://www.cvc.uab.es/~asappa/publications/C__IEEE_IV_2012_W3.pdf

[24] J. McDonald, M. Kaess, C. Cadena, J. Neira, and J. J. Leonard, "6-DOF Multi-session Visual SLAM using Anchor Nodes," Orebro, Sweden, 2011, pp. 69–76. [Online]. Available: http://eprints.maynoothuniversity.ie/6497/

[25] G. Ntzi, S. Weiss, D. Scaramuzza, and R. Siegwart, "Fusion of IMU and vision for absolute scale estimation in monocular SLAM," *Journal of intelligent & robotic systems*, vol. 61, no. 1, pp. 287–299, 2011. [Online]. Available: http://www.springerlink.com.ezproxy.uct.ac.za/index/D86K080887066017.pdf

[26] D. Magree and E. N. Johnson, "Combined laser and vision-aided inertial navigation for an indoor unmanned aerial vehicle," in *American Control Conference (ACC), 2014*. IEEE, 2014, pp. 1900–1905. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6858995/

[27] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tards, "A comparison of loop closing techniques in monocular SLAM," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1188–1197, Dec. 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889009000876

[28] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, Dec. 2011. [Online]. Available: http://ieeexplore.ieee.org/document/6096039/

[29] T. Tuytelaars and K. Mikolajczyk, "Local Invariant Feature Detectors: A Survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2007. [Online]. Available: http://www.nowpublishers.com/article/Details/CGV-017

[30] T. Botterill, S. Mills, and R. Green, "Speeded-up Bag-of-Words algorithm for robot localisation through scene recognition," in *2008 23rd International Conference Image and Vision Computing New Zealand*, Nov. 2008, pp. 1–6.

[31] E. Eade and T. Drummond, "Unified Loop Closing and Recovery for Real Time Monocular SLAM." British Machine Vision Association, 2008, pp. 6.1–6.10. [Online]. Available: http://www.bmva.org/bmvc/2008/papers/124.html

[32] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003. [Online]. Available: https://books.google.co.za/books?hl=en&lr=&id=si3R3Pfa98QC&oi=fnd&pg=PR11&dq=multiple+view+geometry&ots=aRw-lq578I&sig=dwM93c6dqg9QtJ_eQnB_uCHn0UM

[33] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustmenta modern synthesis," in *International workshop on vision algorithms*. Springer, 1999, pp. 298–372. [Online].

Available: https://link-springer-com.ezproxy.uct.ac.za/chapter/10.1007/3-540-44480-7_21

[34] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz, "Multicore bundle adjustment," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 3057–3064. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5995552

[35] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 486–492. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5548123/

[36] K. Konolige, M. Agrawal, and J. Sola, "Large-scale visual odometry for rough terrain," in *Robotics research*. Springer, 2010, pp. 201–212. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-14743-2_18

[37] M. Agrawal, K. Konolige, and M. R. Blas, "CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching," in *Computer Vision ECCV 2008*, ser. Lecture Notes in Computer Science, D. Forsyth, P. Torr, and A. Zisserman, Eds. Springer Berlin Heidelberg, Oct. 2008, pp. 102–115, dOI: 10.1007/978-3-540-88693-8_8. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-540-88693-8_8

[38] M. I. Lourakis and A. A. Argyros, "SBA: A software package for generic sparse bundle adjustment," *ACM Transactions on Mathematical Software (TOMS)*, vol. 36, no. 1, p. 2, 2009. [Online]. Available: http://dl.acm.org.ezproxy.uct.ac.za/citation.cfm?id=1486527

[39] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *Intelligent Robots and Systems, 2008. IROS*

2008. IEEE/RSJ International Conference on. IEEE, 2008, pp. 3946–3952. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/4651147/

[40] M. Warren, D. McKinnon, H. He, and B. Upcroft, "Unaided stereo vision based pose estimation," 2010. [Online]. Available: http://eprints.qut.edu.au/39881/

[41] Niko Snderhauf, Kurt Konolige, Simon Lacroix, and Peter Protzel, "Visual Odometry Using Sparse Bundle Adjustment on an Autonomous Outdoor Vehicle," 2003.

[42] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual SLAM: why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012. [Online]. Available: http://www.sciencedirect.com.ezproxy.uct.ac.za/science/article/pii/S0262885612000248

[43] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello, "A visual odometry framework robust to motion blur," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on.* IEEE, 2009, pp. 2250–2257. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5152447/

[44] P. Newman and K. Ho, "SLAM-loop closing with visually salient features," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on.* IEEE, 2005, pp. 635–642. [Online]. Available: http://ieeexplore.ieee.org.ezproxy.uct.ac.za/abstract/document/1570189/

[45] K. Konolige and M. Agrawal, "FrameSLAM: From bundle adjustment to real-time visual mapping," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, 2008. [Online]. Available: http://ieeexplore.ieee.org.ezproxy.uct.ac.za/abstract/document/4648456/

[46] M. Agrawal, K. Konolige, and R. C. Bolles, "Localization and mapping for autonomous navigation in outdoor terrains: A stereo

vision approach," in *Applications of Computer Vision, 2007. WACV'07. IEEE Workshop on*. IEEE, 2007, pp. 7–7. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/4118736/

[47] M. Agrawal and K. Konolige, "Real-time Localization in Outdoor Environments using Stereo Vision and Inexpensive GPS," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, 2006, pp. 1063–1068.

[48] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera," in *Robotics Research*, ser. Springer Tracts in Advanced Robotics. Springer, Cham, 2017, pp. 235–252, dOI: 10.1007/978-3-319-29363-9_14. [Online]. Available: https://link-springer-com.ezproxy.uct.ac.za/chapter/10.1007/978-3-319-29363-9_14

[49] K. Schmid, T. Tomic, F. Ruess, H. Hirschmller, and M. Suppa, "Stereo vision based indoor/outdoor navigation for flying robots," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 3955–3962.

[50] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7219438/

[51] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, pp. 1–8, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7946260/

[52] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2012, pp. 573–580.

[53] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013. [Online]. Available: http://dx.doi.org/10.1177/0278364913491297

[54] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The New College Vision and Laser Data Set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, May 2009. [Online]. Available: http://dx.doi.org/10.1177/0278364909103911

[55] T. Pire, T. Fischer, J. Civera, P. D. Cristforis, and J. J. Berlles, "Stereo parallel tracking and mapping for robot localization," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 1373–1378.

[56] A. Pazooki, S. Rakheja, and D. Cao, "Modeling and validation of off-road vehicle ride dynamics," *Mechanical Systems and Signal Processing*, vol. 28, pp. 679–695, Apr. 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0888327011004572

[57] N. Bouton, R. Lenain, B. Thuilot, and J. C. Fauroux, "A rollover indicator based on the prediction of the load transfer in presence of sliding: application to an All Terrain Vehicle," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 1158–1163.

[58] L. Li and C. Sandu, "On the impact of cargo weight, vehicle parameters, and terrain characteristics on the prediction of traction for off-road vehicles," *Journal of Terramechanics*, vol. 44, no. 3, pp. 221–238, Jul. 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022489807000201

[59] G. Venture, P. J. Ripert, W. Khalil, M. Gautier, and P. Bodson, "Modeling and Identification of Passenger Car Dynamics Using Robotics Formalism," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 349–359, Sep. 2006.

[60] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison, "Real-time camera tracking: When is high frame-rate best?" in *European Conference on Computer Vision*. Springer, 2012, pp. 222–235. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/chapter/10.1007/978-3-642-33786-4_17

[61] L. Matthies and S. Shafer, "Error modeling in stereo navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 239–248, 1987. [Online]. Available: http://ieeexplore.ieee.org.ezproxy.uct.ac.za/abstract/document/1087097/

[62] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, Sep. 2016. [Online]. Available: http://dx.doi.org/10.1177/0278364915620033

[63] P. Furgale, P. Carle, J. Enright, and T. D. Barfoot, "The Devon Island rover navigation dataset," *The International Journal of Robotics Research*, vol. 31, no. 6, pp. 707–713, 2012. [Online]. Available: http://journals.sagepub.com.ezproxy.uct.ac.za/doi/abs/10.1177/0278364911433135

[64] C. H. Tong, D. Gingras, K. Larose, T. D. Barfoot, and . Dupuis, "The Canadian planetary emulation terrain 3d mapping dataset," *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 389–395, Apr. 2013. [Online]. Available: http://dx.doi.org/10.1177/0278364913478897

[65] J.-L. Blanco, F.-A. Moreno, and J. Gonzalez, "A collection of outdoor robotic datasets with centimeter-accuracy ground truth," *Autonomous Robots*, vol. 27, no. 4, pp. 327–351, 2009. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/article/10.1007/s10514-009-9138-7

[66] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A Tutorial on Graph-Based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[67] R. Kmmerle, 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*, May 2011, pp. 3607–3613.

[68] J.-L. Blanco, J. Gonzlez-Jimnez, and J.-A. Fernndez-Madrigal, "Sparser relative bundle adjustment (srba): constant-time maintenance and local optimization of arbitrarily large maps," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 70–77.

[69] M. Sanfourche, V. Vittori, and G. L. Besnerais, "eVO: A realtime embedded stereo odometry for MAV applications," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 2107–2114.

[70] J. Hedborg, P.-E. Forssn, and M. Felsberg, "Fast and accurate structure and motion estimation," *Advances in Visual Computing*, pp. 211–222, 2009. [Online]. Available: http://www.springerlink.com/index/7315485122551656.pdf

[71] O. D. Faugeras, "What can be seen in three dimensions with an uncalibrated stereo rig?" in *European conference on computer vision*. Springer, 1992, pp. 563–578. [Online]. Available: http://link.springer.com/chapter/10.1007/3-540-55426-2_61

[72] L. Matthies, "Dynamic stereo vision," Ph.D. dissertation, Carnegie Mellon University, 1989. [Online]. Available: http://reports-archive.adm.cs.cmu.edu/anon/1989/CMU-CS-89-195.pdf

[73] M. H. Ang and V. D. Tourassis, "Singularities of Euler and Roll-Pitch-Yaw Representations," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-23, no. 3, pp. 317–324, May 1987.

[74] J. Weng, P. Cohen, M. Herniou, and others, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 10, pp. 965–980, 1992. [Online]. Available: http://vigir.missouri.edu/~gdesouza/Research/Calibration/Calibration%20-%20Weng.pdf

[75] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 1. Ieee, 1999, pp. 666–673. [Online]. Available: http://ieeexplore.ieee.org.ezproxy.uct.ac.za/xpls/abs_all.jsp?arnumber=791289

[76] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Machine Vision and Applications*, vol. 12, no. 1, pp. 16–22, 2000. [Online]. Available: http://link.springer.com.ezproxy.uct.ac.za/article/10.1007/s001380050120

[77] C. Loop and Z. Zhang, "Computing rectifying homographies for stereo vision," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1. IEEE, 1999. [Online]. Available: http://ieeexplore.ieee.org.ezproxy.uct.ac.za/xpls/abs_all.jsp?arnumber=786928

[78] D. Nistr, "An efficient solution to the five-point relative pose problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 756–770, 2004. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/1288525/

[79] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 6, pp. 580–593, 1997. [Online]. Available: http://ieeexplore.ieee.org.ezproxy.uct.ac.za/abstract/document/601246/

[80] T. Kazik, L. Kneip, J. Nikolic, M. Pollefeys, and R. Siegwart, "Real-time 6d stereo visual odometry with non-overlapping fields of

view," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 1529–1536.

[81] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb 1992.

[82] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 4, pp. 376–380, 1991.

[83] D. Nistr, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. Ieee, 2004, pp. I–I. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/1315094/

[84] I. Cvišić and I. Petrović, "Stereo odometry based on careful feature selection and tracking," in *2015 European Conference on Mobile Robots (ECMR)*, Sep. 2015, pp. 1–6.

[85] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981. [Online]. Available: http://dl.acm.org.ezproxy.uct.ac.za/citation.cfm?id=358692

[86] L. M. Paz, P. Pinis, J. D. Tards, and J. Neira, "Large-scale 6-dof slam with stereo-in-hand," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 946–957, Oct 2008.

[87] . M. Mozos, A. Gil, M. Ballesta, and O. Reinoso, "Interest point detectors for visual slam," in *Conference of the Spanish Association for Artificial Intelligence*. Springer, 2007, pp. 170–179. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-540-75271-4_18

[88] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: http://link.springer.com/article/10.1023/B:VISI.0000029664.99615.94

[89] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in *Computer Vision ECCV 2006*, ser. Lecture Notes in Computer Science, A. Leonardis, H. Bischof, and A. Pinz, Eds. Springer Berlin Heidelberg, May 2006, pp. 404–417, dOI: 10.1007/11744023_32. [Online]. Available: http://link.springer.com/chapter/10.1007/11744023_32

[90] C. Harris and M. Stephens, "A combined corner and edge detector." in *Alvey vision conference*, vol. 15. Manchester, UK, 1988, pp. 10–5244. [Online]. Available: http://courses.daiict.ac.in/pluginfile.php/13002/mod_resource/content/0/References/harris1988.pdf

[91] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*. Springer, 2006, pp. 430–443. [Online]. Available: http://link.springer.com/10.1007/11744023_34

[92] P. F. Alcantarilla and T. Solutions, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *IEEE Trans. Patt. Anal. Mach. Intell*, vol. 34, no. 7, pp. 1281–1298, 2011. [Online]. Available: https://pdfs.semanticscholar.org/6c5c/61bc780b9a696ef72fb8f27873fa7ae33215.pdf

[93] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571.

[94] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Computer Vision (ICCV), 2011 IEEE International Conference on.*

IEEE, 2011, pp. 2548–2555. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6126542/

[95] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-15561-1_56

[96] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*. IEEE, 2012, pp. 510–517. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6247715/

[97] J. Morel and G. Yu, "ASIFT: A New Framework for Fully Affine Invariant Image Comparison," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 438–469, Jan. 2009. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/080732730

[98] S. M. Smith and J. M. Brady, "SUSANA New Approach to Low Level Image Processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, May 1997. [Online]. Available: http://link.springer.com/article/10.1023/A:1007963824710

[99] A. L. Dahl, H. Aan\a es, and K. S. Pedersen, "Finding the best feature detector-descriptor combination," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*. IEEE, 2011, pp. 318–325. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5955377/

[100] A. Schmidt, M. Kraft, M. Fularz, and Z. Domagala, "COMPARATIVE ASSESSMENT OF POINT FEATURE DETECTORS AND DESCRIPTORS IN THE CONTEXT OF ROBOT NAVIGATION." *Journal of Automation, Mobile Robotics & Intelligent Systems*, vol. 7, no. 1, 2013. [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&profile=ehost&scope=site&authtype=crawler&jrnl=18978649&AN=86674188&

h=oY35rOScHAh5Wbg4PJzPRUmK6kcIIGS7zMnfEt9S%
2BAiISsGyO5quFVvcvoVqIAVR4%2BmY5nXLm3ZTZXhNEBHZAw%
3D%3D&crl=c

[101] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[102] J. Heinly, E. Dunn, and J.-M. Frahm, "Comparative evaluation of binary features," in *Computer VisionECCV 2012*. Springer, 2012, pp. 759–773. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-33709-3_54

[103] A. Gil, O. M. Mozos, M. Ballesta, and O. Reinoso, "A comparative evaluation of interest point detectors and local descriptors for visual SLAM," *Machine Vision and Applications*, vol. 21, no. 6, pp. 905–920, 2010. [Online]. Available: http://www.springerlink.com/index/5840554431201255.pdf

[104] O. Miksik and K. Mikolajczyk, "Evaluation of local detectors and descriptors for fast feature matching," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 2681–2684. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6460718/

[105] A. Pieropan, M. a. Bjrkman, N. Bergstrm, and D. Kragic, "Feature Descriptors for Tracking by Detection: a Benchmark," *arXiv preprint arXiv:1607.06178*, 2016. [Online]. Available: https://arxiv.org/abs/1607.06178

[106] H. Aanæs, A. L. Dahl, and K. S. Pedersen, "Interesting interest points," *International Journal of Computer Vision*, vol. 97, no. 1, pp. 18–35, 2012. [Online]. Available: http://link.springer.com/article/10.1007/s11263-011-0473-8

[107] O. Chum and J. Matas, "Matching with PROSAC - progressive sample consensus," in *2005 IEEE Computer Society Conference on*

Computer Vision and Pattern Recognition (CVPR'05), vol. 1, Jun. 2005, pp. 220–226 vol. 1.

[108] B. Kitt, J. Rehder, A. Chambers, M. Schonbein, H. Lategahn, and S. Singh, "Monocular Visual Odometry using a Planar Road Model to Solve Scale Ambiguity," *Proceedings of European Conference on Mobile Robots*, Sep. 2011. [Online]. Available: http://repository.cmu.edu/robotics/941

[109] G. Klein and D. Murray, "Improving the agility of keyframe-based SLAM," in *European Conference on Computer Vision*. Springer, 2008, pp. 802–815. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-540-88688-4_59

[110] "Open computer vision (opencv))," https://opencv.org, 2019, accessed 2019-31-01.