# Augmentation of Visual Odometry using Radar

by

David Murdoch

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Masters of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2021

## Abstract

As Unmanned Aerial Vehicles (UAVs) become viable for more applications, pose estimation continues to be critical. All UAVs need to know where they are at all times, in order to avoid disaster. However, in the event that UAVs are deployed in an area with poor visual conditions, such as in many disaster scenarios, many localization algorithms have difficulties working.

This thesis presents Visual Inertial Loop-closing Direct Sparse Odometry (VIL-DSO) a visual odometry method as a pose estimation solution, combining several different algorithms in order to improve pose estimation and provide metric scale. This thesis also presents a method for automatically determining an accurate physical transform between radar and camera data, and in doing so, allow for the projection of radar information into the image plane. Finally, this thesis presents Extended Visual-Inertial Loop Closing Direct Sparse Odometry (EVIL-DSO), a method for localization that fuses visual-inertial odometry with radar information. The proposed EVIL-DSO algorithm uses radar information projected into the image plane in order to create a depth map for odometry to directly observe depth of features, which can then be used as part of the odometry algorithm to remove the need to perform costly depth estimations.

Trajectory analysis of the proposed algorithm on outdoor data, compared to differential GPS data, shows that the proposed algorithm is more accurate in terms of root-mean-square error, as well as having a lower percentage of scale error. Runtime analysis shows that the proposed algorithm updates more frequently than other, similar, algorithms.

**Acknowledgements**

I would like to thank all the people who made this thesis possible. Dr. Melek and Dr. Shaker, my advisors, gave me a lot of good advice advice and helped direct my occasionally directionless energy. Dylan Gareau gave a lot of help in the implementation of VIL-DSO and obtaining of results, as well as mentorship and advice. Assylbek Dakibay helped with the practical issues of drone flights and hardware synchronization, as well as various ROS problems. Barry Gilhuly and Olzhas Adiyatov gave some very helpful technical advice, and help with troubleshooting the stickiest problems I ran into.

## Dedication

This is dedicated to Amber, without whose support it never would have been finished.

# Table of Contents

# List of Tables

# List of Figures

# List of Abbreviations

**BoW** Bag of Words 24

**CFAR** Constant False Alarm Rate x, 58

**CUT** Cell Under Test 58

**DATMO** Detection and Tracking of Moving Objects 12, 13

**DoA** Direction of Arrival 11

**DSO** Direct Sparse Odometry 9, 19

**DVSO** Deep Virtual Stereo Odometry 9, 10, 33, 39

**EKF** Extended Kalman Filter 12

**EVIL-DSO** Extended Visual-Inertial Loop Closing Direct Sparse Odometry iii, 32, 33, 38–42

**FFT** Fast Fourier Transform 57

**FMCW** Frequency Modulated Continuous Wave x, 11, 12, 56, 57

**FoV** Field of View 18, 20, 32

**ICP** Iterative Closest Point 12

**IF** Intermediate Frequency 57

**IMU** Inertial Measurement Unit 1–3, 15, 17, 19–21, 30

| | |
|---|---|
| **LDSO** | Loop Closing Direct Sparse Odometry 9, 15, 24 |
| **LiDAR** | Light Detection and Ranging 1, 10, 12 |
| **RMSE** | Root Mean Square Error 38 |
| **Rx** | Receiving 59 |
| **SLAM** | Simultaneous Localization and Mapping 1, 4, 10–12, 15, 33, 38 |
| **Tx** | Transmit 59 |
| **UAV** | Unmanned Aerial Vehicle 1, 2, 10, 12, 13, 24, 33, 38, 40, 42, 57 |
| **UAVs** | Unmanned Aerial Vehicles iii, 1, 2, 4, 11, 13, 41, 42 |
| **VI-DSO** | Visual Inertial Direct Sparse Odometry 9, 33, 39 |
| **VIL-DSO** | Visual Inertial Loop-closing Direct Sparse Odometry iii, 15, 25, 32, 33, 39 |
| **VO** | Visual Odometry 1, 3, 4, 8, 10, 11, 13, 15, 25, 38–40 |

# Chapter 1

# Introduction

Unmanned Aerial Vehicles (UAVs) and other robots were once solely ideas in science fiction. In recent years, however, they have become relatively commonplace due to wider availability and lower cost in acquiring both the robots themselves, and the sensors that are crucial for them to be useful. This has led to UAVs being used with a wide variety of sensors, in a wide variety of applications. Accurate estimation of the pose of the Unmanned Aerial Vehicle (UAV) is required across almost all applications, which has also contributed to developments in odometry and Simultaneous Localization and Mapping (SLAM) algorithms to accomplish this task.

A wide variety of sensors have been used in odometry and Simultaneous Localization and Mapping (SLAM) algorithms. Some have used laser range finders [82], others have used Light Detection and Ranging (LiDAR) [30], or Inertial Measurement Unit (IMU) and GPS based approaches [24]. Camera based approaches have also had great success, particularly with UAVs, due to the relatively low cost, size, and weight of the sensor. Cameras are also frequently used as part of the sensor loadout for a wide variety of UAV applications other than localization, such as surveying, aerial filming and photography, or as a navigational aid for other vehicles. This incentivizes the use of cameras for localization, instead of mounting an extra sensor. For these reasons, cameras are frequently used in algorithms for pose estimation, such as in Visual Odometry (VO). A single camera on its own, however, cannot be used for certain aspects, such as direct depth estimation, and can have difficulty in areas with low textured areas and fast maneuvers. With a single camera, it is also impossible to estimate a real world scale between features. For these reasons, it is frequently combined with one or more other sensors.

Frequently, a camera is combined with an IMU in order to provide accurate short term motion estimation, as well as to add an inertial component to optimization, which allows for scale estimation. Other sensors can also be used to augment this approach, which allows for each sensor to add its unique strengths to the method. Radar can be used for direct measurement of depth information of points, as well as for more accurate information in lower textured areas.

This thesis presents a method for using radar, IMU, and camera information in tandem, in order to produce an accurate estimation of the current robot pose, and an accurate map of the surrounding area. In particular, it focuses on the estimation of pose for UAVs, using sensors that are commonly used in existing UAV applications.

## 1.1 Motivation

In general, pose estimation and mapping for robotics is extremely important. Accurate mapping, in fact, is tied closely to accurate pose estimation; a robot can't map the area around it accurately if it doesn't know where it is and what its orientation is within that environment. These techniques can be can be used in a wide variety of applications, such as in remote infrastructure inspection, search and rescue, and defense.

In remote infrastructure inspection, precise estimation of a robot's pose is important, as they frequently work in GPS-deprived environments, and have to localise solely based on their surroundings. Similarly, mapping and inspection is useful for getting useful data from drones that are performing inspections. Infrastructure inspection can be performed in a variety of conditions, but places a premium on accurate information gathering, as missed fault or crack can cost millions of dollars to fix if it is left for too long. In extreme cases, as with dams and bridges, infrastructure failure can cost lives, and so it is imperative that the data available be as accurate as possible. It is for this reason that we turn to fusions of multiple sensors, such as radar and camera, in order to provide the robust and informative data needed for inspection, and reduce uncertainty inherent in each individual sensor.

In defense applications, mapping can also be used for a wide variety of tasks. Mapping of terrain before an operation can be extremely useful to inform personnel about what sorts of hazards they could face over the course of their work, and establish what sorts of dangers are present. Live mapping during an operation can also allow for localisation of threats to personnel quickly and accurately, potentially saving their lives.

Pose estimation and mapping are especially important for time sensitive applications, such as search and rescue. Natural disasters can completely change the environments that they occur in, rendering any existing maps and information about the environment useless, while also creating situations in which timely operation is essential in order to save lives. A matter of minutes can make a significant difference, as statistics show that 93 percent of avalanche victims can survive if found and dug out within 15 minutes, while the survival rate drops to 30 percent at 35 minutes[17]. It is also crucial to have information about areas affected by natural disaster, such as the types of terrain, or a reconstructed surface, in order to inform ground teams, ground robots, and rescue efforts in general, to aid in both damage mitigation and human rescue efforts as necessary.

Many existing algorithms using purely visual methods have significant sources of error that can accumulate throughout the process. Error can accumulate via simple calculation error, or through drift from the ground truth. In visually difficult areas, there can also be a failure to detect and track points, resulting in a loss of tracking and a total failure of the method. Methods that include IMU can add robustness to lower texture areas, but also add their own error source in the form of IMU drift. Purely visual methods also have difficulty calculating depth, as point tracking and depth calculations are difficult, and stereo vision is impossible on a standard drone at high altitudes, as the cameras would need to be several meters apart to enable stereo vision.

Natural disaster and defense scenes also frequently include adverse visual conditions. These can include low texture variation, fog, shifting rubble, low light, rain, or even flooded water, that prevent purely visual point selection can have great difficulties in selecting and tracking points. mmWave radar, on the other hand, is unaffected by a adverse visual conditions, and can allow for a greater range of operational conditions than purely visual odometry can. In addition, it allows for additional sensory opportunities, such as for the detection of humans through objects, and the measurement of heart rate and breathing[6].

Radar is also more accurate at depth estimation, and cheaper as well. The lower price should allow for more accurate pose estimation and greater robustness in the system for all applications, while still being at lower cost than other, completely camera based systems.

This research is therefore motivated towards creating pose estimation methods that are more robust to changing visual conditions, as well as more accurate depth estimation. The purpose of this is to support operations in difficult, changing conditions, such as disaster scenarios and defense applications.

## 1.2 Problem Statement

VO performs poorly in visually difficult areas on the whole. Radar ignores most conditions that cause difficulty for visual sensors, but it is rarely used alongside visual sensors. Calibration between camera and radar in both azimuth and elevation is difficult, and there is no currently accepted technique for doing so. Radar also has difficulty processing terrain in many signal processing methods, which can lead to difficulty in using it as a standalone processing method.

This thesis will hence focus on developing solutions to address the problem of VO using UAVs in visually difficult areas, specifically with an aim at reducing error due to depth estimation, and improving estimation of scale. This thesis also aims to minimise the cost of VO, and allow for accurate estimation of pose at longer distance and higher altitudes from the ground.

## 1.3 Contributions

This thesis presents a method for calibrating radar and camera information, by using nonlinear optimization methods to determine a best fit transform between the two sensors. This thesis also presents an extension to existing methods by calibrating for a radar in both azimuth and elevation.

Current literature presents methods for visual-inertial odometry and loop closing visual odometry, but no methods for combining the two together short of a full SLAM solution. This thesis provides a method for combining visual inertial and loop closure together, in order to create more accurate visual odometry, and includes results on both datasets and acquired data.

Finally, this thesis presents a method for augmenting visual odometry information with the use of radar. Firstly, it uses the calibrated radar information to augment the depth estimation of points, thereby increasing the accuracy of the scale estimation in the visual inertial odometry, and therefore increasing the accuracy of the pose. Furthermore, it can be used to select points and perform assocations even under difficult visual conditions, such as in low light conditions or under adverse weather, such as fog. This method is then verified through simulation and experimental results.

## 1.4 Outline

This thesis will present state-of-the-art techniques and relevant background information in Chapter 2. Chapter 3 describes the visual odometry method to be augmented using radar. Chapter 4 describes the method used for radar-camera calibration, and results for the accuracy in calibration. Chapter 5 described the process of augmenting visual odometry with radar. Chapter 6 presents the conclusions of this thesis, as well as recommendations for future work.

# Chapter 2

# Review and Background

In this chapter, we will go over necessary background information for the presentation of the proposed visual odometry methods, such as factor graph optimization and radar signal processing. We also discuss previous research efforts in visual odometry, mapping and localization with radar, and methods for fusing the two sensors that have been previously researched.

## 2.1 Factor graph optimization

Many dynamic systems can be modelled using Markov chains, in which possible states are entirely dependent on previous states in the system. In pose estimation for robotics, the states to be observed are $x \in R^6$. In a a hidden Markov model, the states are assumed to be not directly observable, and therefore they must be inferred from the noisy measurements $z \in R^n$, which can be directly observed. The set of states $x_t$ from all timesteps is noted as $X$, and the set of all observations is represented by $Z$. The observations $Z$ are obviously dependent on the hidden states $X$, which leads to the conditional probability formulation $P(Z|X)$. As an example of what such a model can look like, Figure 2.1 shows a hidden Markov Model represented as a Bayesian network, where the task is to determine the hidden state sequence $(x_1, x_2, x_3)$ for measurements $z_1, z_2, z_3$ that minimises the posterior probability $P(z_1, z_2, z_3|x_1, x_2, x_3)$.

This system can also be represented using a factor graph. A factor graph is a bipartite graph with vertices representing factors and variables. Variables represent the unknown states to be estimated, while factors represent probabilistic information on those variables

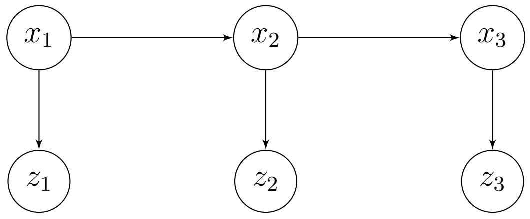


Figure 2.1: A Hidden Markov Model modeled as a Bayesian network

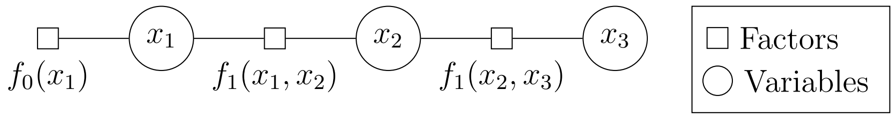


Figure 2.2: A Hidden Markov Model modelled as a factor graph

that is gained from the observations. Edges can only exist between factors and variables, to connect each factor to the variables that the factor provides information on. The same bayesian network as before is illustrated in Figure 2.2.

The value of the graph is defined as

$$f(X) = \prod_i f_i(x_i). \tag{2.1}$$

The goal is to estimate the subset of state variables $x_i \in X$ that maximizes the value of the graph, and by extension, maximizes the probability of receiving the given measurements $Z$. This yields the maximum a posteriori probability state as

$$X^{\text{MAP}} = \arg\max_X \prod_i f_i(x_i). \tag{2.2}$$

The state estimates required for this are generated through optimization methods. The residual function $r(x_i, z_i)$ of the factor graph can be used to determine the cost function for optimization. For example, a simple residual function is

$$r(x_i, z_i) = h(x_i) - z_i, \tag{2.3}$$

where $h(x)$ is the sensor model that maps states $X$ to measurements $Z$. The goal of the estimation is to find the values of $X$ such that the likelihoods of observations $Z$ are maximized, as in

$$p(Z|X) \propto exp\left(-\frac{1}{2}||h_i(x_i) - z_i||^2_{\Sigma_i}\right). \tag{2.4}$$

where $\Sigma_i$ is the known covariance matrix and $||\cdot||_{\Sigma_i}$ is the Mahalanobis norm. Since the logarithm of Equation 2.4 is monotonic, maximizing the probability is equivalent to minimizing the negative log-likelihood function and dropping the $\frac{1}{2}$ factor, which results in

$$X^{\text{MAP}} = \arg\min_X \sum_i ||h_i(x_i) - z_i||^2_{\Sigma_i}. \tag{2.5}$$

This is equivalent to minimizing the sum of nonlinear least squares. The cost function is then defined as

$$g(X) := \sum_i ||h_i(x_i) - z_i||^2_{\Sigma_i}. \tag{2.6}$$

Optimizing this cost function yields an estimate for the entire sequence of states $X$, which tends to yield a smoother trajectory, as opposed to methods which only estimate the current state. This problem is commonly solved using Gauss-Newton optimization or Levenberg-Marquardy optimization.

## 2.2 Camera

### 2.2.1 Visual odometry methods

In general, VO methods can be seperated into 4 different categories, defined by 2 different qualities. In all formulations of the VO problem, a probabilistic model that takes measurements $Z$ as input is used, which then computes an estimate of the actual value $X$, typically using a maximum likelihood approach of some kind to maximise the probability of obtaining the actual measurements. Where the types of models differ is how much of the available sensor data they use, and how exactly they use it. This leads to a separation of algorithms based on whether they are dense or sparse, and direct or indirect.

#### Direct vs. Indirect

Indirect methods proceed in 2 steps. The first step is pre-processing the raw camera data in some form. This typically takes the set of extracting a set of keypoints, in the form of features of some kind, which can then be used as noisy measurements in a model to determine camera position. It can also take the form of established correspondences using dense optical flow, or methods that extract and match geometric primitives, such as lines or corners. Indirect methods optimize a geometric error, as the quantities observed are geometric in nature.

Direct methods, on the other hand, work directly using the raw sensor data provided from the sensor itself. They optimise a photometric error, as the camera sensor itself is a photometric sensor.

#### Dense vs. Sparse

Dense methods use all pixels available in the two dimensional image plane, whereas sparse methods use a selected set of independent points(traditionally corners.) Semi-dense methods exist as well, which use a large subset of largely connected, well constrained points of the image, but do not use the complete surface.

Dense and sparse methods also differ in their use of geometric priors. In sparse methods, points are regarded as conditionally independent from each other, and no geometric priors are used in optimisation. In dense and semidense methods, the connectedness of the region of points used is exploited to form a geometric prior, favoring smoothness. Geometric priors are actually required in order to make a dense world model observable [74] [67] [88]

All 4 possible combinations of these methods exist in the literature. There are also hybrid methods, which do not fit neatly into any of these categories, such as [21]. Relevant methods are described in sections subsection 2.2.2 and subsection 2.2.3.

### 2.2.2 Existing Visual odometry methods

Most relevant to this thesis are direct, sparse algorithms, such as the Direct Sparse Odometry (DSO) family of algorithms. These methods are keyframe based, sparse methods that include dynamic marginalization to improve the accuracy of estimation[14]. Later additions have also added inertial sensors, to add real world scale to the estimation [92], known as Visual Inertial Direct Sparse Odometry (VI-DSO), as well as loop closure, to reduce the drift error inherent in estimation from purely visual and inertial measurements [22], known as Loop Closing Direct Sparse Odometry (LDSO). Another version of DSO added stereo capabilities [94]; however, stereo vision is unusable from higher altitudes, requiring cameras to be several meters apart at 100m of altitude. This is thus disregarded as a possible solution for direct depth estimation. DSO has also been augmented with deep learning techniques to perform virtual stereo depth estimation, known as Deep Virtual Stereo Odometry (DVSO), and shown in [98]. Other methods have used fisheye cameras to create an omnidirectional view [60].

Many other visual odometry methods have been developed, however. Among the most notable is [21], which uses a semi-direct formulation for optimization, but otherwise uses a keyframe based approach. The originator of the keyframe based approach used by many of these methods was [52], who originally used the nonlinear optimisation model for odometry, as opposed to the filter based approaches of the past, and it was further improved by [65].

Depth estimation is a problem of great interest in visual odometry. Some methods combine visual methods with LiDAR for improved depth estimation, such as [26] [25] [103], which has the disadavantage of requiring LiDAR. LiDAR can be heavy, power intensive, and often less accurate and robust than radar. Similarly, others have attempted the use of RGB-D camera systems, which can directly measure depth, with algorithms such as [45] [58] [42] [108] [43]. RGB-D systems can be flooded out easily in outdoor applications due to ambient infrared light, and are both expensive and power hungry. They also lose accuracy at longer distances. For these reasons, we disregard RGB-D cameras as a suitable sensor for use in this method.

Other methods use deep learning techniques to attempt to estimate depth directly from images. DVSO [98] is one example of this, but there are many others, including [57], which augments [21] with depth prediction, as well as [59] [76] [99] [93] [53] [102], which all use some form of depth estimation to augment their visual odometry systems. These systems, however, add significant extra computational cost to the odometry system, which may not always be feasible if the computation is to be performed on the UAV, as opposed to on a base station. Additionally, these methods often have difficulties if datasets are unavailable for the type of terrain the UAV will be flying over. It is both computationally simpler and more reliable to simpler measure the depth directly.

### 2.2.3 Existing Visual SLAM methods

Visual SLAM methods are frequently used in autonomous vehicles as a whole, but are less frequently used in UAV applications, due to their more demanding computational requirements. While VO methods generally only optimise pose in a local window, SLAM solutions optimise both the trajectory and the measured points points globally, simultaneously optimally mapping the space and using that map to localize the robot. Visual odometry measures points, but does not consider them as variables to be jointly optimised with the poses. This leads to a more accurate, but much more computationally expensive estimation in SLAM methods.

Visual SLAM methods can also be categorized as direct, indirect, sparse, or dense, similar to VO methods. ORB-SLAM [63] and its successor, ORB-SLAM2 [64], are sparse, indirect methods, which are the most commonly used types of SLAM system, along with systems like monoSLAM [10] and PTAM [46]. Dense, indirect methods, like [77], or [90], using geometric priors and geometric error to optimise. Dense, direct methods also exist, with methods like DTAM [67] and the predecessor to it [88] using a combination of geometric priors and photometric error. Direct, sparse SLAM methods, however, are few in number, with most direct, sparse methods being VO methods instead of SLAM.

## 2.3 Radar

### 2.3.1 Signal processing methods

For this project, a FMCW radar operating in the 77-81 GHz band is used. High frequency FMCW has several benefits, including simple transmitters, excellent range resolution, and resistance to interference, all of which are crucial to operation with UAVs. With FMCW radar, signal processing begins with a Fourier Transform being performed on the returns detected from each receiving antenna on the radar, in order to determine absolute ranges from each antenna. The results of the Fourier Transforms are then run through a Direction of Arrival (DoA) algorithm, in order to determine an accurate estimate of the angle from which obstacles are detected. This can then be used, in combination with the ranges from the Fourier Transforms, to determine Cartesian coordinates for targets.

The full mathematical breakdown of the signal processing methods used and FMCW radar can be found in A.

### 2.3.2 Existing Radar Mapping Methods

Historically, existing radar based mapping methods are satellite based methods, which use the principal of interferometry. Interferometry is a technique that combines waves with the same frequency, using the principle of superposition, in order to intentionally cause interference, and extract meaningful data from the resulting interference. This allows for useful information to be obtained because when two waves of the same frequency are combined using superposition, the resulting wave is a function of the phase difference between the two waves, which can be used to calculate data like time of flight or distance travelled, and can give a highly accurate reading. This technique has been in use for several decades [27] [101] [80] for use in topographical measurements. However, in more recent years, other techniques have become more common, as satellite mapping is frequently used for other applications, such as mapping ice cover, flooding, forests, and other environmental concerns.

### 2.3.3 Existing Radar SLAM methods

Radar based SLAM frequently uses different algorithms than modern camera based systems. Many systems use an Extended Kalman Filter (EKF) based approach, with varying signal processing approaches, such as [50] which used doppler radar to detect points and then used an EKF SLAM method on landmarks extracted from the doppler. A group from Korea used EKF methods as well, but used W-band radar instead of doppler radar [48]. A group from LASMEA, France, used FMCW radar, which is used in this thesis, and is described in detail in Appendix A, in order to perform SLAM using an EKF, and also usesdthe Fourier Mellin transform in order to ignore data association and do efficient matching [23]. Similar results can be seen with [51], who used simulated FMCW radar as well.

Not all radar SLAM methods are EKF based methods, however. There are still some methods that use graph based optimisation methods. A group from Daimler AG used landmarks determined from radar points in a graph based optimisation method and achieves good results in a parking lot setting [81]. Another group used radar scans directly and matched them using an Iterative Closest Point (ICP) method, then put those matches into a factor graph to achieve results better than more advanced radars with more dense point clouds [31].

Recent methods appear to focus primarily on EKF based methods, with some graph based methods. However, a few other methods have been used. One group used a novel framework with an augmented state vector that used the radar cross sections of the detected objects in order to perform SLAM [37]. On the other hand, [61] uses a normal distribution transform to perform SLAM and also compares it to data from a LiDAR using the same algorithm, finding the radar to be slightly less accurate, though noting that the original radar sensor was less accurate as well. Notably, however, none of these methods or experimental results have been tested on a UAV.

## 2.4 Radar-camera fusion

Radar camera fusion has been performed in some applications before, but always using ground based vehicles. A frequent use of radar-camera calibration is in Detection and

Tracking of Moving Objects (DATMO), in which the ability of a radar to directly measure velocity can be of great use. Frequently, such algorithms use radar as a method to direct the attention of a camera to a given area, or extract features that can be used to augment visual tasks, such as in [33], [33], [39], or [96], where features detected by the radar are used to augment object classification via a neural net, or in [44] [87] [5] [71], where radar is used to augment camera based tracking of objects, including pedestrians and vehicles. Other approaches have more recently begun to incorporate neural nets, in order to perform the classification. CenterFusion [66] is an approach that detects center points using camera data with a neural net first, then associating radar tracks with them using a frustrum based method. Another group used neural networks that take both camera and radar data at the same time in order to detect obstacles in a two dimensional plane, and used novel neural network training techniques to improve the accuracy of their fusion [68]. Occasionally radar and camera are also fused in different ways, such as using detection of movement with radar to direct security cameras to areas that need extra attention[41]. All of these methods, however, are focused on object detection, classification, and tracking, which leads them to use a lot of similar fusion methods. None of them focus on localization of a robot, which by necessity, requires a different approach. More data is needed for accurate localization than is needed for accurate object detection, which leads our approach to other techniques detailed later.

Besides DATMO, the most common use of radar and camera calibration is for textured surface reconstruction. Our review of the available literature has come up with several methods by which this may be performed. A group in France [12] uses an approach based on sensor geometry and measurement of three dimensional features, which is extended in [13]. Meanwhile, [47] uses a mmWave multiview scanner in order to take measurements, and uses structure from motion techniques in order to create a textured representation of the environment, as well as estimate the pose of the scanner. However, both of these techniques are solely for ground based surface estimation; neither of them have been tested on a UAV.

## 2.5 Summary of previous methods

In summary, none of the existing methods solve the problem stated in Section 1.2. Several methods of monocular VO exist, but few of them work for UAVs. Of those that do, the only algorithms that work with UAVs have their own problems, including lack of depth perception, lack of scale estimation, and lack of any good way to estimate depth. They also are not robust to any difficult visual conditions. Stereo algorithms are insufficient for the task, as the distance between cameras needed for depth perception quickly becomes too great to be feasibly mounted on a drone. Radar algorithms on their own do not allow for the same level of accuracy as purely visual methods, due to difficulty in associating points in radar data. Theoretically, radar-camera fusion should be more accurate than either radar or camera localization on their own. It should also allow for more robust odometry. However, no efforts have been made to perform this localization. This is what this thesis aims to do, in order to solve the problem outlined in Section 1.2.

# Chapter 3

# Visual Inertial Odometry method

## 3.1 Overview

In order to augment visual odometry with radar, as this thesis proposes, we first require a base visual odometry method to augment. Visual odometry is frequently used for robotics applications that don't have the computational resources to use a full SLAM system. In order to make our augmented version as accurate as possible, it is helpful to have a solution that is reasonably accurate even without radar augmentation.

In this section of the thesis, we present the VO method to be augmented, known as VIL-DSO. It uses a keyframe based, sliding window optimisation framework, similar to the framework used in [52], [64], and [22], among others. VIL-DSO is built from LDSO[22] and includes inertial measurements from an IMU, which are tied to the keyframes of the system, as well as loop closure for reducing drift error. It is composed of three primary threads: the coarse tracker, the local optimizer, and the global optimiser. These threads, as well as the required initialization, are summarised below, as are the results of VIL-DSO, which are comparable or better than state-of-the-art VO and SLAM algorithms.

## 3.2 Initialization and Pre-integration of the IMU

Before any optimisation can occur, a number of key parameters related to the IMU need to be estimated. These include the gyroscope and accelerometer biases, as well as the magnitude of gravity and scale, which is defined as the ratio between distances in the

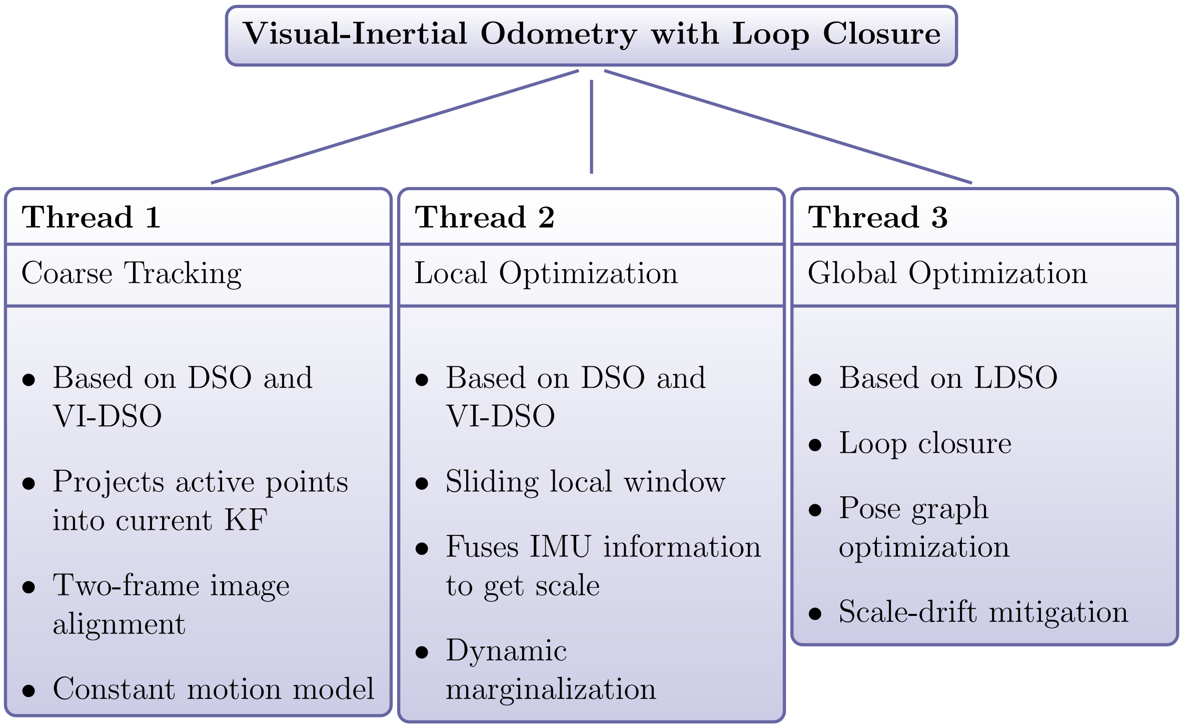


Figure 3.1: Main threads in VIL-DSO

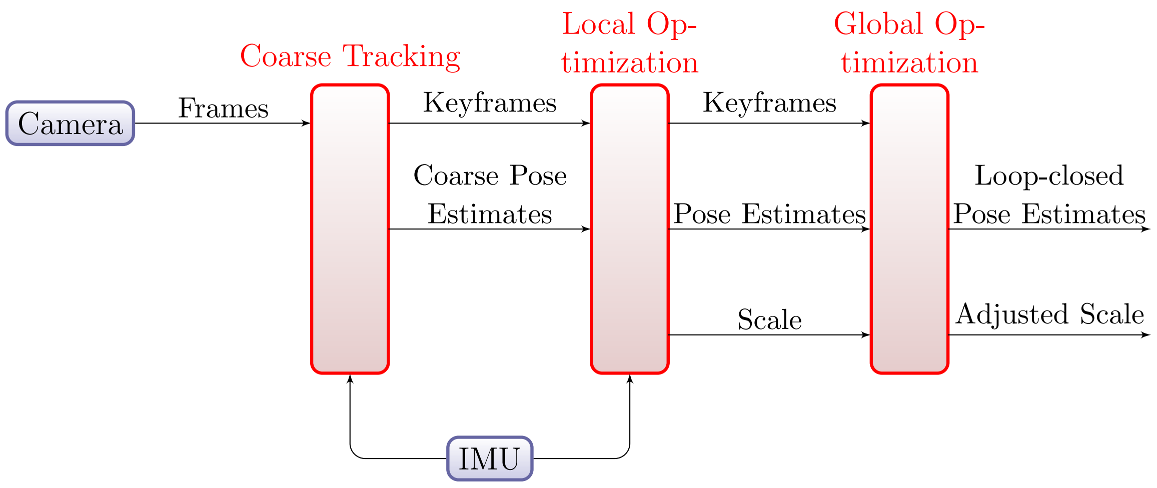


Figure 3.2: Structure of VIL-DSO

odometry reference frame and the world reference frame. The initial estimate for gravity is determined by using the first 30 measurements from the IMU, before any visual information is used. The scale is initially assumed to be 1.0, while the initial estimates for biases, denoted as the vector $\mathbf{b_0}$, are set to each be 0.0. These are only initial estimates, and will be included in joint optimisation later on.

IMU data is almost always generated at a much higher rate than camera data. In order to generate individual transforms that can be associated with keyframes, the IMU data is preintegrated using a constant acceleration model, using the manifold method developed in [20]. These individual transforms can then be used to represent all IMU readings taken between consecutive frames or keyframes, depending on where the information is being used.

## 3.3 Key frame and Point Management: The Coarse tracker

After initialization of the IMU, each frame input into the odometry program passes through a number of steps to determine if it will be used as a keyframe, and added to the sliding window used for local optimisation.

### 3.3.1 Coarse Tracking

Each frame is first tracked using the coarse tracker relative to the latest keyframe. This allows for tracking in between keyframes, where it isn't efficient to perform a full local optimisation. It also provides an initial estimate for the local window optimization to start at.

The coarse tracker first projects all active points into the current keyframe and dilates them. It then uses conventional direct image alignment between the latest frame and the latest keyframe, during which the scale is fixed. This alignment uses multi-scale pyramid representation and a constant motion model. Inertial residuals are calculated from IMU preintegration and included between subsequent frames. After each completion of the local optimization, a new keyframe is generated, and the coarse tracker is reinitialized using the scale, gravity, biases, and velocities as references. Resulting poses from the coarse tracker are used as initial estimates for local optimisation.

### 3.3.2 Key Frame Generation

Keyframe generation is performed with new frames after coarse tracking is complete. If a weighted summation of the following criteria exceeds a predefined threshold, then a new keyframe is generated:

1. The Field of View (FoV) changes significantly, as measured by the mean square optical flow from the last keyframe to the latest frame,

$$f := \left( \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{p} - \mathbf{p}'||^2 \right)^{\frac{1}{2}}, \tag{3.1}$$

   where $n$ is the number of frames sinces the last keyframe, $\mathbf{p}$ is the vector of estimated points in the image frame, and $\mathbf{p}'$ is the vector of projections of those points using the current inverse depth estimate. . This projection is performed using

$$\mathbf{p}' = P_c(\mathbf{R}P_c^{-1}(\mathbf{p}, d_p) + \mathbf{t}), \tag{3.2}$$

   where $P_c$ is the projection from $\mathbf{R^3}$ into the image plane using pinhole camera parameters $c$, $P_c^{-1}$ is the back projection from the image plane into $\mathbf{R^3}$ , $d_p$ is the current inverse depth estimate, and $\mathbf{R}$ and $\mathbf{t}$ are the estimated rotation and translation matrices between the last keyframe and latest frame.

2. The translation of the camera causes occlusion of the points, which is indicated by the mean flow without rotation. This quantity is defined as $f_t$, the mean optical flow with only translation, and calculate it as

$$f_t := \left( \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{p} - \mathbf{p}_t'||^2 \right)^{\frac{1}{2}}, \tag{3.3}$$

   where $\mathbf{p}_t'$ is the warped point position calculated using Equation 3.2 with $\mathbf{R} = \mathbf{I}_{3x3}$, determining the mean optical flow without accounting for rotation.

3. There is significant change in brightness, as indicated by a relative brightness factor between the keyframe and the current frame,

$$a := |log(e^{a_j - a_i} t_j t_i^{-1})|, \tag{3.4}$$

   where $t_i$ and $t_j$ are the exposure times of the last keyframe and current frame respectively, and $a_j$ and $a_i$ are the current estimates of our scalar affine brightness parameters, which are jointly optimized as part of the local optimization thread.

The weighted summation of the three above criteria is defined as

$$b := w_f f + w_{f_t} f_t + w_a a \tag{3.5}$$

where $w_f$, $w_{f_t}$, and $w_a$ are the relative weights, with the only constraint $w_f, w_{f_t}, w_a > 0$. If $b$ exceeds a preset threshold, a new keyframe is generated. The threshold is defined as $T_{kf} = 1$.

After the above weighted criteria check, an additional check against IMU timestamps is calculated. If the time between the current frame IMU timestamp and the latest keyframe IMU timestamp exceeds an arbitrary threshold, in this case set to 0.45 seconds, and $b$ exceeds a second threshold $T_{imu}$, then a new keyframe is taken. $T_{imu} = 0.45$ and is also arbitrarily defined.

This section includes several different parameters that can be tuned and are which all can effect the performance of the algorithm. However, [14] performed an analysis of the effects of changes in each of these parameters. In order to tune these parameters for the experiments described in Chapter 5, the parameters there were used as a starting point, and a simple gradient descent was used to optimise them for our specific experiments.

### 3.3.3 Point selection

A limited number of points are used for estimation, in order to reduce redundancy and make computation faster. In each keyframe, a number of point candidates are proposed by uniform random selection, arbitrarily set to be 1500. Points are then selected from the point candidates by use of 2 different methods. The first method is the method originally proposed by DSO [14], which uses a gradient filter to detect points with high gradients, while the second method uses the easy to compute Shi-Tomasi score[83] to detect corners. This method uses a mix of the two methods in order to allow higher accuracy when corners can be detected using the Shi-Tomasi score, while also still allowing for robustness in lower texture areas with the DSO point selection strategy.

The points are then tracked through subsequent keyframes, with the different poses of the frames being used to form an estimate of the depth, as well as determine the variance of that estimate. The process of forming the estimate involves minimising the error in the depth estimation, based on the matching of the points in previous keyframes. Our replacement to this depth estimation is described in Section 5.1.

### 3.3.4 Marginalisation

Marginalisation is used to remove old states, in order to reduce the complexity of calculations while retaining some prior information about previous states. The criteria to decide if certain data is marginalized or not is based on frame and pose data. Once the data from a frame is marginalised, associated factors are removed from the factor graph, and a new factor is formed to act as a prior for the window. Marginalisation is completed using the Schur complement.

The criteria for marginalization, as detailed in [14], are listed below, where $I_1$ is the current keyframe:

1. Keep the latest two keyframes, $I_1$ and $I_2$.

2. Marginalize a keyframe if it shares less than 5% of its points shared with $I_1$.

3. Marginalize a keyframe if the number of keyframes exceeds the set threshold, $N_I$. A distance score between keyframes is calculated to select the frame to remove,

$$s(I_i) = \sqrt{d(i,1)} \sum_{j \in [3,n] \setminus \{i\}} (d(i,j) + \epsilon)^{-1}, \tag{3.6}$$

   where $d(i,j)$ is defined as the Euclidean distance in $\mathbf{R}^3$ between the estimated poses of keyframes $I_i$ and $I_j$, defined as $||\mathbf{t}_i - \mathbf{t}_j||_2$ where $\mathbf{t}_i$ and $\mathbf{t}_j$ are the $\mathbf{R}^3$ estimated positions of each keyframe, and $\epsilon$ is a small constant.

4. Marginalize all points that are no longer in the FoV, across all frames.

5. Marginalize all points for which the host frame is marginalized.

These marginalisation criteria work for all variables. Marginalized information is retained as priors for future optimization for all variables in the current window that are connected to older states, with the sole exception of scale. In order to handle this, [92] introduces the idea of dynamic marginalization, which maintains multiple marginalization factors that are calculated and maintained, and are reset when the scale estimate deviates significantly. The three factors used are:

1. $M_{visual}$, which contains only scale independent information generated from visual information, and includes no IMU data.

2. $M_{curr}$, which contains all information since the last frame was marginalized.

3. $M_{half}$, which contains only recent states with similar scale estimates to the most recent estimate.

When the current scale deviates significantly from the scale estimate at the latest marginalized prior, the priors are reset by setting $M_{curr}$ equal to $M_{half}$ and $M_{half}$ equal to $M_{visual}$. This results in the optimization always retaining some information from prior states regarding the scale estimate.

## 3.4 Optimization in the Local Window: Pose Estimation

In the local window thread, we create a factor graph to minimise photometric error between keyframes, by optimizing energy residuals as in [14]. This is calculated in a sliding window to allow for easier computation. Frames and points that leave the field of view are marginalized. The remaining keyframes form a window, local to the current keyframe, where the associated poses are optimised.

The factor graph shown in Figure 3.3 shows how error terms used in the optimization are created. Error terms are generated between the host keyframe of a point and every keyframe that shares that point. As an example, $E_{\mathbf{p}1^3}$ is the error term generated by Point 1($\mathbf{P}1$), which is hosted by keyframe 1 and overlapped with keyframe 3. Further definition of these error terms can be found in [14]. Blue edges are between points and their host keyframes, while red edges are between points and other overlapping keyframes.

The sum of the energy residuals represents the photometric error for all the keyframes and points represented in the factor graph. This is defined [14] as

$$E_{photo} := \sum_{i \in F} \sum_{\mathbf{p} \in P_i} \sum_{j \in obs(\mathbf{p})} E_{\mathbf{p}j}, \tag{3.7}$$

where $i$ iterates over all keyframes $F$ in the local window, $\mathbf{p}$ iterates over all points P in keyframe $i$, and $j$ iterates over all keyframes in which the point $\mathbf{p}$ is visible. Optimization of the graph is performed using initial estimates from the coarse tracking thread, as described in section Section 3.3, and optimised using the Gauss-Newton algorithm.

To add metric scale to the optimization, however, more terms must be added to the factor graph. As in [92], the factor graph must also include pose, IMU biases, velocity, gravity direction, and scale, in addition to visual information. Factors are generated using

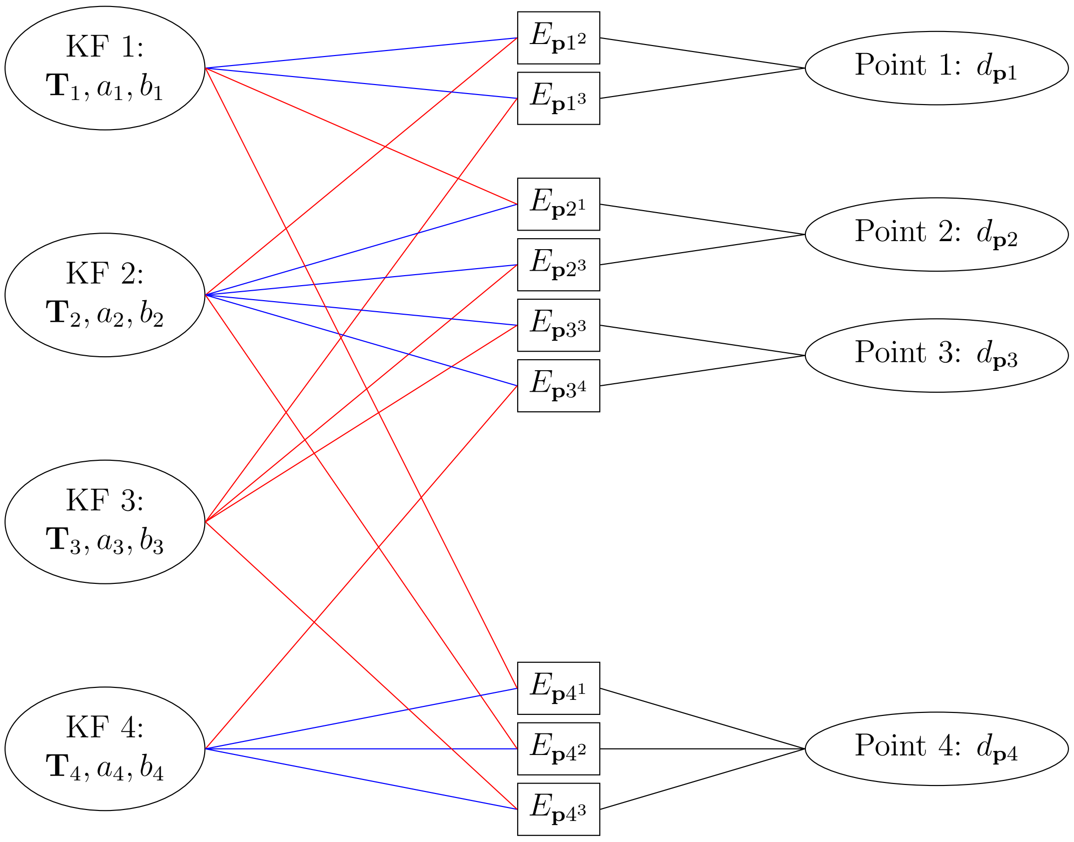

KF 1: $\mathbf{T}_1, a_1, b_1$
KF 2: $\mathbf{T}_2, a_2, b_2$
KF 3: $\mathbf{T}_3, a_3, b_3$
KF 4: $\mathbf{T}_4, a_4, b_4$
$E_{\mathbf{p}1^2}$
$E_{\mathbf{p}1^3}$
$E_{\mathbf{p}2^1}$
$E_{\mathbf{p}2^3}$
$E_{\mathbf{p}3^3}$
$E_{\mathbf{p}3^4}$
$E_{\mathbf{p}4^1}$
$E_{\mathbf{p}4^2}$
$E_{\mathbf{p}4^3}$
Point 1: $d_{\mathbf{p}1}$
Point 2: $d_{\mathbf{p}2}$
Point 3: $d_{\mathbf{p}3}$
Point 4: $d_{\mathbf{p}4}$

Figure 3.3: Structure of the bundle adjustment factor graph from DSO, from [14].

the inertial and visual measurements are used to generate factors relating the unknown variables, using the known probabilistic information. The resulting factor graph is shown in Figure 3.4.

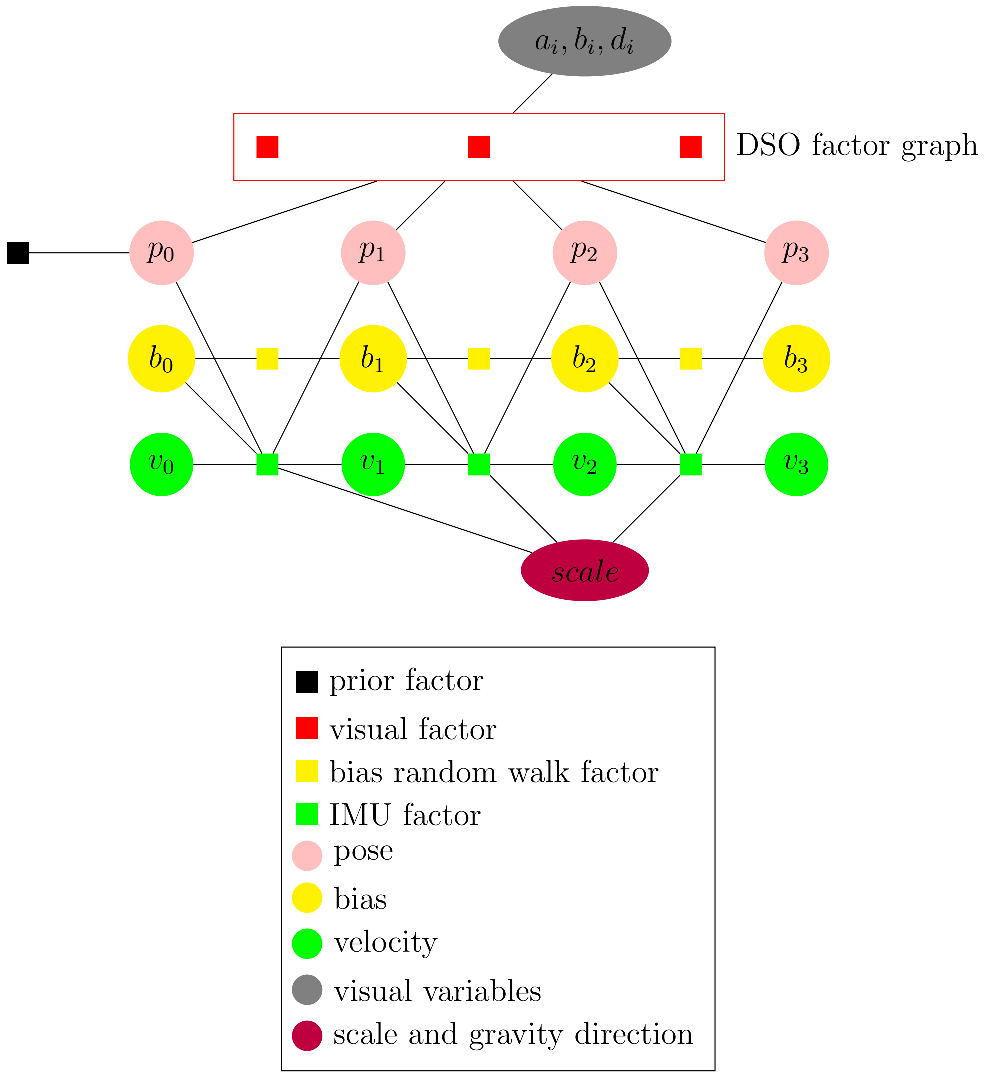


Figure 3.4: Factor graph for VI-DSO [92].

## 3.5 Optimization in the Global Window: Loop Closure

Each of the estimated poses for the keyframes are calculated in a local window. However, this doesn't take into account for when the UAV revisits previous locations. When the UAV revisits other areas, the information can be used to reduce drift error by utilizing loop closure. Instead of using the the local sliding window, loop closure uses all available keyframes, in a global window. The factors between keyframes in the global pose graph are calculated from relative poses that were determined from the pose estimates in the local sliding window.

In order to first detect loop closure, the ORB [79] descriptors are calculated from each keyframe. These should be repeatable due to to how the LDSO point selection uses the Shi-Tomasi score to ensure that points are corners. These descriptors are then packed into a Bag of Words (BoW), which helps to store the data compactly. Whenever a new keyframe is generated, matching point candidates are proposed by referencing the BoW. When there are enough matches, a loop is detected and a transformation between the matched frames is calculated. This transformation is used to generate an edge in the global pose graph, as demonstrated in Figure 3.5.

The edge consists of a transform between the reference frame and the current keyframe being matched. First, let $P = \{\mathbf{p}_i\}$ be the reconstructed features in the reference keyframe, taken from the BoW and with inverse depth $\mathbf{d}_{\mathbf{p}_i}$, and let the matched features in the current keyframe be $Q = \mathbf{q}_i$ with $\mathbf{D}$ being the sparse inverse depth map created by projecting points in the current sliding window into the current keyframe being matched. Because $\mathbf{D}$ is a sparse map, some features have depth associated with them, while others have only pixel positions. Let $Q_1 \subseteq Q$ be the features without depth estimates, and $Q_2 = Q \setminus Q_1$ be the features with depth estimates. Using all of these, a pose estimate between the reference and current keyframe $S_{cr}$ can now be computed by minimising the energy function

$$\begin{aligned} E_{loop} := \sum_{q_i \in Q_1} & w_1 ||S_{cr} P_c^{-1}(\mathbf{p}_i, \mathbf{d}_{\mathbf{p}_i}) - P_c^{-1}(\mathbf{q}_i, \mathbf{d}_{\mathbf{q}_i})||_2 + \\ \sum_{q_j \in Q_2} & w_2 ||P_c(S_{cr} P_c^{-1}(\mathbf{p}_j, \mathbf{d}_{\mathbf{p}_j})) - \mathbf{q}_j||_2, \end{aligned} \tag{3.8}$$

where $P_c^{-1}$ and $P_c$ are defined as before, and $w_1$ and $w_2$ are weights to balance out different measurement units.

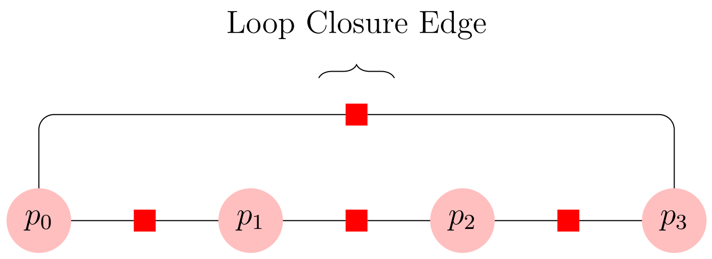


Figure 3.5: A representation of loop closure in a factor graph.

## 3.6 Summary of contribution

In this chapter, we presented a VO algorithm that includes visual and inertial information, to be augmented by radar information in future steps. VIL-DSO gives accurate localization in ideal circumstances, and also uses easily associated, repeatable features. However, it has no means of accurate depth estimation, as it is only a monocular system, and has significant error built into estimations of scale. This makes it ideal to augment with radar information. The method for augmenting it will be presented in the following two chapters.

# Chapter 4

# Radar-Camera Calibration

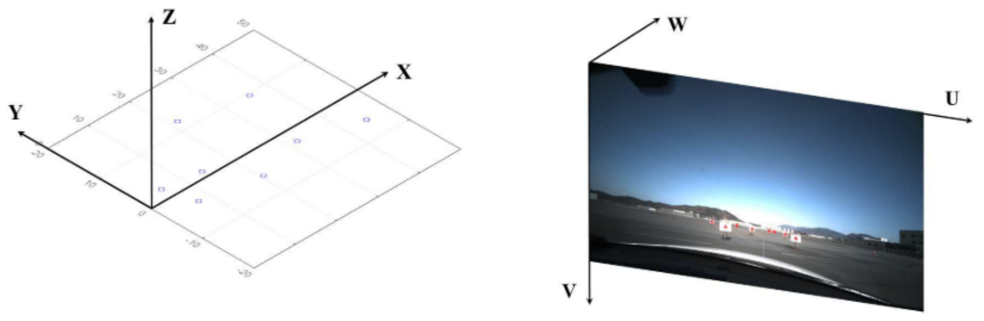
This chapter deals with the problem of determining the rigid transform between a radar sensor and a camera sensor, in order to directly detect depth and augment the visual odometry presented in Chapter 3. Most existing camera-radar calibration methods only use 2 dimensions, and ignore any information along the z-axis. The method presented here also presents a novel technique for use of points along the z-axis. This section also goes into detail about how the calibrated points are turned into a depth map, which can then be used to augment visual odometry.

## 4.1 Calibration Methodology

The method used, broadly speaking, determines a transformation to map a three-dimensional point in space to a point on the image plane of the camera. It is similar to extrinsic calibration methods for camera pose estimation [70]. For this method, the coordinate systems of the respective sensors are defined as in Figure 4.1. The calibration can then be modelled with the following equation:

$$\tilde{\mathbf{q}} = \mathbf{K}[\mathbf{R}\,\mathbf{t}]\tilde{\mathbf{p}}, \tag{4.1}$$

where $\tilde{\mathbf{p}} = [x, y, z, 1]^t$ and $\tilde{\mathbf{q}} = [u, v, 1]^t$ are the homogeneous coordinates of $\mathbf{p}$ and $\mathbf{q}$, respectively. Normally, with a 2 dimensional radar, $z = 0$, but as we are considering the three-dimensional radar case, we also consider height in this equation. In Equation 4.1, $\mathbf{K}$



(a) Radar (b) Camera

Figure 4.1: The coordinate systems used in calibration [70].

is the camera intrinsic matrix, obtained via calibration, and defined as

$$\begin{bmatrix} -k_u f_x & s & u_0 \\ 0 & k_v f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

As in [13], we consider $s$, the skew parameter, to be negligible, and therefore $s = 0$. The matrix $[\mathbf{R}\,\mathbf{t}]$ is known as the extrinsic matrix, and defines the rigid spatial transform between the 2 sensors. It consists of the rotation matrix as

$$\mathbf{R} = \mathbf{R}_z \mathbf{R}_y \mathbf{R}_x \tag{4.3}$$

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & \sin\theta_x \\ 0 & -\sin\theta_x & \cos\theta_x \end{bmatrix}, \mathbf{R}_y = \begin{bmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{bmatrix}, \tag{4.4}$$

$$\mathbf{R}_z = \begin{bmatrix} \cos\theta_z & \sin\theta_z & 0 \\ -\sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.5}$$

and the translation vector as

$$\mathbf{t} = [t_x, t_y, t_z]^T \tag{4.6}$$

The transformation is clearly defined by the six variables

$$\mathbf{w} = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z], \tag{4.7}$$

which can then be used as the optimization variables in a nonlinear optimization problem with the following objective function:

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \|\tilde{\mathbf{q}}_i - \mathbf{K}[\mathbf{R}\,\mathbf{t}]\tilde{\mathbf{p}}_i\|_2^2 \tag{4.8}$$

This optimisation problem is aimed at minimising the pairwise error between radar points and camera points, by minimising the error in the transform. $\mathbf{p}_i$ and $\mathbf{q}_i$ are paired points, describing a three dimensional point and the corresponding point on the image plane. The points are typically spread at an variety of distances and angles from the sensors being calibrated. After obtaining a resulting $\mathbf{w}$, an arbitrary radar point target can then be mapped onto an image, or by inverting the transformation to get

$$[\mathbf{R}\,\mathbf{t}]^{-1}\mathbf{K}^{-1}\tilde{\mathbf{q}} = \tilde{\mathbf{p}}, \tag{4.9}$$

an image point can be mapped to a radar point, or a point on a generated surface in the radar coordinate frame.

## 4.2 Calibration Results

In order to determine the quality of calibration, a simple experiment is devised. Camera calibration was conducted via OpenCV and a checkerboard, in order to determine the intrinsic matrix $\mathbf{K}$. In order to obtain data for the calibration, a radar and camera were each affixed to a board to create a rigid transform. A series of steel targets were set up and covered with paper, in order to make them easy to detect using conventional computer vision methods. The targets were placed at a variety of heights and distances from the radar and camera, to allow for many points to improve optimizations. They were set up in 4 different configurations, with one configuration being used for calibration, and the remaining three being used for evaluation.

In order to evaluate the quality of calibration, an equation must be defined for calibration error. The calibration error for $N$ pairs of points is defined as

$$\frac{1}{N}\sum_{i=1}^{N} d(\hat{\mathbf{q}}_i, \mathbf{q}_i), \tag{4.10}$$

where $d$ is the Euclidean distance on the image plane between 2 points, defined as $||\hat{\mathbf{q}} - \mathbf{q}_i||_2$, and $\hat{\mathbf{q}}_i$ is an image coordinate mapped from $\mathbf{p}_i$ by the estimated image. This compares

Table 4.1: Radar Camera Calibration methods compared between 2 and 3 dimensions

| Method | N_r=4 | N_r=5 | N_r=6 | N_r=7 | N_r=8 | N_r=16 |
|---|---|---|---|---|---|---|
| 2 Dimensions: | - | - | 3.370 | 3.263 | 3.194 | 2.942 |
| 3 Dimensions: | 3.523 | 3.421 | 3.351 | 3.312 | 3.205 | 2.997 |

the distance between an actual image coordinate of a target, and the projected image coordinate from a radar point, in order to evaluate the quality of the calibration performed.

The calibration was carried out, and then compared to results from [70], which calibrate for radar points in 2 dimensions. These results are used to show that there is no significant loss in calibration accuracy in the transition from 2 dimensions to 3. As can be seen from the results in 4.1, the results are basically indistinguishable in accuracy from the 2 dimensional case, meaning this method is an accurate and acceptable method for radar-camera calibration.

## 4.3 Depth map creation

After calibration has occurred, radar points can be projected into the camera space. However, this alone is not sufficient to augment the visual odometry method proposed. The radar point cloud is sufficiently sparse that features in any given image are unlikely to exactly match a radar point, and doing point to pixel association at runtime is costly and difficult to implement well. In order to avoid this problem, the radar information will be formed into a depth map. An example depth map is shown in Figure 4.2.

In order to determine the depth of individual pixels, each radar point must first be input into Equation 4.9, to determine where in the image the point is. Each of these points was mapped onto an identically sized image to the input image, which is the base for the depth map. Next, piecewise linear interpolation was performed between all adjacent points in order to fill out the rest of the depth map. This allows for pixelwise association between a depth and a feature in the original image.

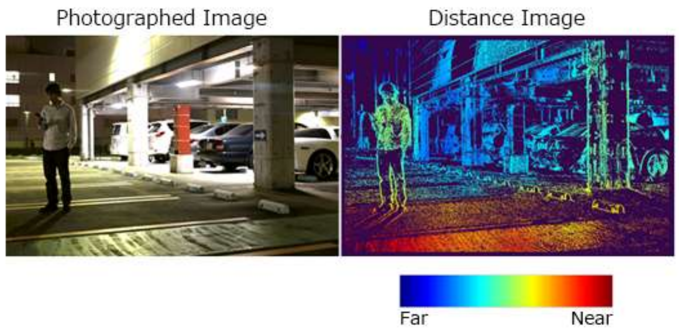


Figure 4.2: A depth map, displayed next to an actual picture [2]

### 4.3.1 Optimisation of depth map creation

Interpolating between each point, for each frame of data, is obviously an expensive operation. However, it also isn't necessary to interpolate between all points at every frame. The movement of the drone is approximately known at each time step, as a result of the preintegration of the IMU that is performed regularly. As a result, instead of fully recalculating at each new image, the approximate transform generated by pre-integrated IMU data is performed on the point cloud. New points are then projected onto the depth map, and in any area in which the depth map is not defined as a result of the transformation, the piecewise linear interpolation is performed again.

Additionally, occasionally during testing, there were issues with the point cloud becoming too sparse, and the resulting depth map being inacccurate. In order to combat this, during the piecewise linear interpolation, a check was added to calculate distance between adjacent points. If points are too far apart, the region between them is considered to be unknown, and is set to have a depth of zero. This lets the odometry know that the depth for that pixel is invalid, and any point candidates that use the depth in that area are considered to be invalid, and are not kept.

## 4.4 Summary

In this chapter, we present a method for calibrating radar information with camera images, by projecting radar information onto the plane of an image. This allows for the creation of a sparse depth map for each image. This can, in turn, be used to associate pixels and features with an absolute depth. This can be used in visual odometry methods in order to improve accuracy, as is demonstrated in Chapter 5.

# Chapter 5

# Visual Odometry with Radar Augmentation

The VIL-DSO method has been introduced, as has the method of radar-camera calibration, which allows the projecting of depth information onto images from the camera. Together, this information allows for the augmentation of the visual odometry method presented in Chapter 3. In this chapter, we present a method for augmenting the data used for VIL-DSO with radar depth information to create Extended Visual-Inertial Loop Closing Direct Sparse Odometry (EVIL-DSO). We also show that the augmentation of VIL-DSO with radar information both improves the accuracy of the visual odometry algorithm, and allows for more frequent updates of the estimated pose.

## 5.1 Modified point selection strategy and coarse tracker

In order to augment the data efficiently, the point selection stategy used for VIL-DSO must be adjusted. The basic point selection method selects from all parts of the image, selecting a certain number of points from individual regions in the image, but using the entirety of the image. The modified point selection strategy restricts the regions used for point candidate selection to the regions of the image where the FoV of the radar and camera overlap and depth information is available, which can be determined by only taking into account pixels for which a depth value has been added. This allows for an initial estimate of depth to be added to the point as soon as possible.

Once the tracking of points begins, however, point candidates that have already been generated do not necessarily require further radar information to improve the depth estimation. As a result, when coarse tracking is establishing correspondences between points, the entire image is used for tracking points, in order to keep tracking as robust as possible. If depth information is available for the point after tracking in subsequent frames, then the new depth information is included in the estimate using an extended Kalman Filter. Otherwise, depth information is left unchanged, as opposed to using the Levenberg Marquardt optimization on each point in order to maximize energy residuals and better estimate the depth that is performed in VIL-DSO.

## 5.2 Experiment design

To test the proposed algorithm, we use a set of 3 data sequences, with a total flight time of 12 minutes. The radar and camera are attached to the bottom of the UAV used, with a fixed translation between them. The UAV used had a D-GPS unit, the data from which is used as ground truth for the purposes of this experiment. Data collected is then used to compare EVIL-DSO against other visual odometry and SLAM systems, including ORB-SLAM2, VIL-DSO, VI-DSO, and DVSO. All comparisons were done offline, not in real time.

Sequence 1 is a flight at approximately 1.5 m/s, moving in squares over the terrain at a constant altitude. Figure 5.1 shows a general overhead view of the path taken by the UAV during the sequence. At each corner in the sequence, the UAV paused for about 1 second before turning and continuing the path. Sequence 1 was designed to allow maximum opportunity for loop closure.

Sequence 2 is a flight at approximately 2 m/s, in which the altitude remains constant after takeoff at 30m. Unlike in sequence 1, the UAV does not stop at any point. Figure 5.3 shows an overview of the path taken. Sequence 2 was created as a medium difficulty sequence, with minimal opportunity for loop closure. Sequence 2 was also useful for testing robustness of the tracking algorithm.

Sequence 3 is a flight at approximately 4m/s, where the altitude varies between 20m and 45m above the ground. Sequence 3 was created specifically to test the depth estimation of EVIL-DSO against other algorithms, and examine what effects that may have on accuracy. Sequence 3 also intentionally includes almost no opportunity for loop closure, and yaws randomly throughout the flight, up to a full 360 degree rotation of the camera. It does this in an attempt to make accurate point matching and depth estimation as difficult as possible. A bird's eye view of the trajectory in the x-y plane is shown in Figure 5.4.

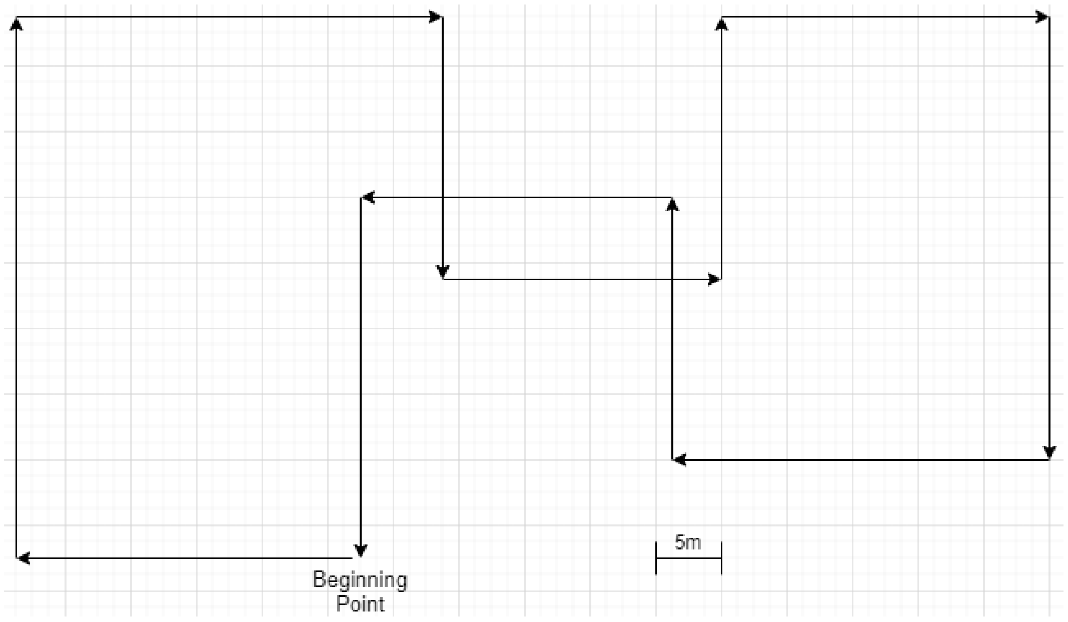


Figure 5.1: An overview of the path taken in sequence 1.

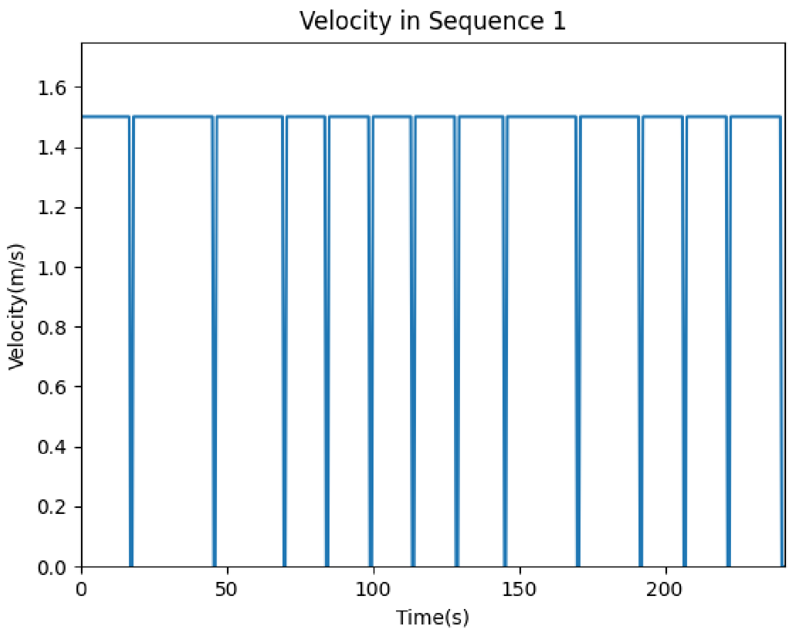


Figure 5.2: A plot of the velocity in sequence 1.

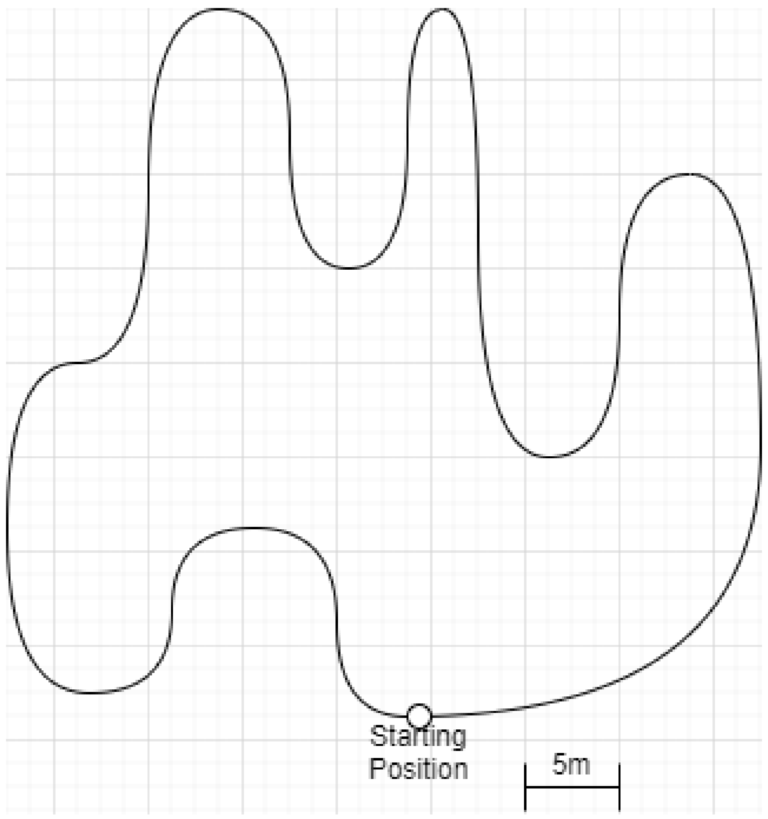


Figure 5.3: An overview of the path taken in sequence 2.

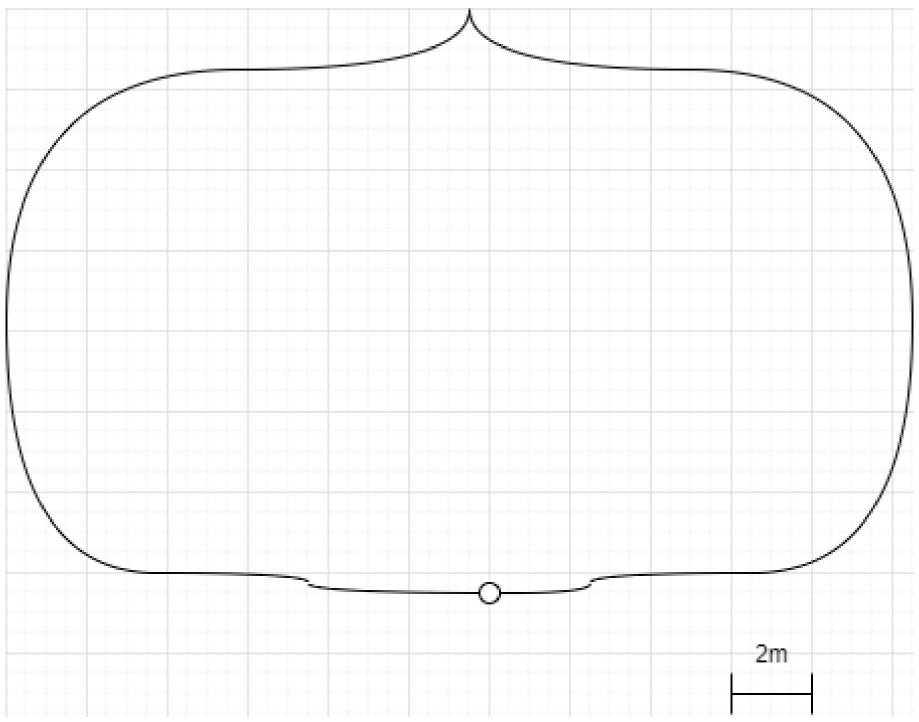


Figure 5.4: An overview of the path taken in sequence 3.

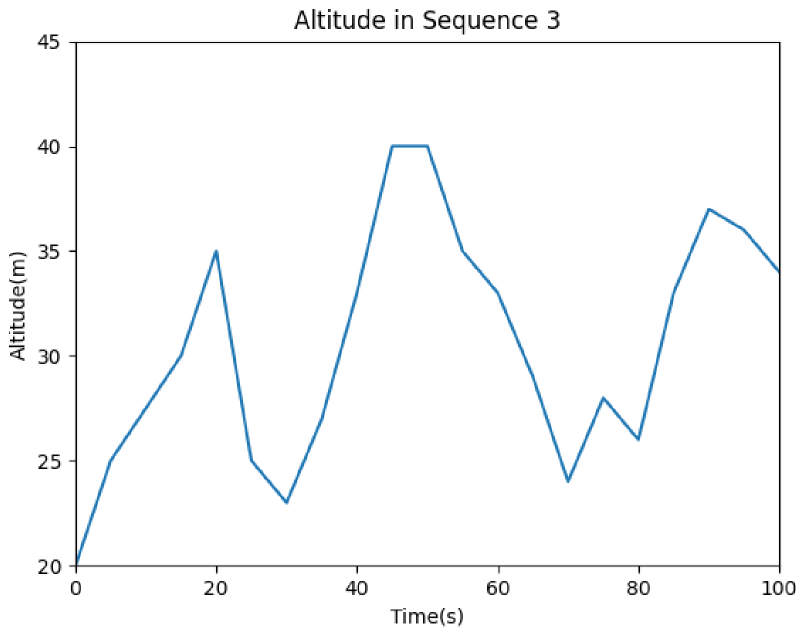


Figure 5.5: A plot of the altitude over time in sequence 3.

## 5.3 Experimental Results

All measurements in this section are calculated as the mean of 10 runs of each sequence, unless otherwise noted.

### 5.3.1 Accuracy

In this section, we compare multiple state of the art VO and SLAM algorithms to the performance of EVIL-DSO, using trajectory alignment and analysis techniques from [106]. All algorithms are compared using the data provided from the D-GPS data from the UAV as ground truth to calculate three different metrics. The first is the translation Root Mean Square Error (RMSE), defined as the root mean square error between the ground truth trajectory and the calculated trajectory from EVIL-DSO. The second is the percentage scale error, which is defined as the RMSE between the estimated scale at each pose and the actual scale. The final metric is the RMSE between ground truth and calculated trajectory after the calculated trajectory is aligned to the ground truth and scaled using the ground truth scale estimate. This is referred to as the ground truth scaled error, or gt-scaled. In order to give a more accurate comparison, global optimization is disabled for ORB-SLAM. Note that DVSO was trained with data from the 2019 IEEE GRSS Data Fusion contest[7]. The results are summarised in Table 5.1.

EVIL-DSO has noticeably lower scale error than other options, as a result of the use of radar in point selection. With augmented point selection, scale error becomes negligible, as rather than point depth being measured at an arbitrary scale, where a conversion scale must then be estimated, the point scale is almost exactly the real scale. This allows for a more accurate estimation, as can be seen from the table.

### 5.3.2 Runtime efficiency

In order to measure runtime efficiency, 2 different metrics are considered. The first is the time it takes to perform initial tracking and depth estimation. After initial tracking of an input frame, a rough estimate of a pose can be calculated based off of the last optimized pose, so the rough estimate can be used The second is the frequency of optimized updates from the local window optimization common to all of these algorithms, which gives the most update to date, optimized current pose. In order to measure these estimates, each of the three sequences was run 10 times, and the measured times for each frame and optimized pose update were averaged together.

Table 5.1: Visual odometry and SLAM methods compared

| Method | Sequence | $S_1$ | $S_2$ | $S_3$ |
|---|---|---|---|---|
| VI-DSO | Translation RMSE (m) | 0.075 | 0.145 | 0.517 |
| | Translation gt-scaled (m) | 0.066 | 0.057 | 0.448 |
| | Scale RMSE (%) | 1.1 | 2.0 | 5.4 |
| VIL-DSO | Translation RMSE (m) | 0.067 | 0.097 | 0.111 |
| | Translation gt-scaled (m) | 0.047 | 0.062 | 0.073 |
| | Scale RMSE (%) | 1.2 | 1.8 | 3.8 |
| DVSO | Translation RMSE (m) | 0.075 | 0.084 | 0.102 |
| | Translation gt-scaled (m) | 0.072 | 0.070 | 0.082 |
| | Scale RMSE (%) | 0.5 | 0.8 | 3.2 |
| EVIL-DSO | Translation RMSE (m) | **0.065** | **0.064** | **0.074** |
| | Translation gt-scaled (m) | 0.068 | 0.063 | 0.069 |
| | Scale RMSE (%) | 0.3 | 0.2 | 0.5 |
| VI-ORB-SLAM | Translation RMSE (m) | 0.075 | 0.084 | 0.087 |
| | Translation gt-scaled (m) | 0.049 | 0.062 | 0.067 |
| | Scale RMSE (%) | 0.5 | 0.8 | 1.5 |

All runtime analysis is performed using an Intel NUC 8 computer, using an i5-8295U processor with 8Gb of ram and running Ubuntu version 16.04, in order to test how EVIL-DSO can run on a computer that could actually work on a drone. Signal processing for the radar is performed using the onboard digital signal processing of the radar. This does result in slightly higher latency between triggering of the sensors and input of that data to the VO algorithm, but the throughput of the algorithm itself remains unaffected. The results for both point selection and local window optimization are presented in Table 5.2.

EVIL-DSO has a noticeably faster point intialization rate than any of the other options. This is because VIL-DSO and VI-DSO both perform individual Levenberg Marquardt optimizations on each point in order to update the depth information, DVSO has to perform inference on the image to extract depth information, and VI-ORB-SLAM has to extract features and then perform tracking and depth estimation. EVIL-DSO performs none of those steps, instead directly observing the depth and allowing for faster updating of the depth information.

The optimized pose update rate of EVIL-DSO is sligtly faster than the rates for VI-DSO and VIL-DSO, as the optimization converges slightly faster with the better initial

Table 5.2: Runtime analysis results for various SLAM and VO algorithms.

| Method | Optimization Metric | Update rate(Hz) |
|---|---|---|
| VI-DSO | Point initialization | 14.21 |
| | Optimized pose | 4.54 |
| VIL-DSO | Point initialization | 14.02 |
| | Optimized pose | 4.32 |
| DVSO | Point initialization | 17.03 |
| | Optimized pose | 5.32 |
| EVIL-DSO | Point initialization | 17.18 |
| | Optimized pose | 5.95 |
| VI-ORB-SLAM | Point initialization | 13.06 |
| | Optimized pose | 5.05 |

estimates for points and pose.

## 5.4 Summary

In this section, we presented EVIL-DSO, a method which uses VO techniques combined with radar information in order to perform localization. This method is shown to be more accurate than previous efforts using only visual-inertial information, and has significantly lower scale error, as a result of the fusion of different sensors. This method is also shown to be more efficient than other methods on an example of a computer that could feasibly be mounted on a UAV, in terms of how quickly new information can be processed, and how often optimized poses can be obtained from the system.

# Chapter 6

# Conclusion

In this thesis, a method augmenting visual odometry with radar information is presented, with the resulting algorithm being EVIL-DSO. The presented algorithm combines previous work in visual odometry together, implementing loop closure and visual inertial information, and combining it with radar information to produce a robust algorithm that allows for direct observation of depth.

The experimental results of EVIL-DSO show that the proposed algorithm has lower RMSE translational error than state of the art options for visual odometry. It also consumes fewer computational resources, making it ideal for deployment on UAVs. It also functions without GPS data, making it ideal for either indoor or outdoor scenarios for any application, including search and rescue.

## 6.1 Recommendations

The use of radar in EVIL-DSO should allow for more robust performance even in poor visual conditions. However, due to issues with waterproofing and flight safety with the currently available UAVs, this has not been tested. Given adequate waterproofing and safety measures, EVIL-DSO should be tested in adverse visual conditions such as rain and snow, in order to verify this hypothesis.

Surface reconstruction using visual odometry as a basis has been explored as a topic before, frequently with [74] and [21] as the algorithms used. However, algorithms such as the one proposed in [74] grow more accurate with both more accurate pose estimations and more accurate estimations of depth. Reconstruction of scenes using sensor fusion methods from UAVs has yet to be explored, and EVIL-DSO provides an excellent foundation to explore that space, along with related topics such as aerial semantic segmentation of terrain.

The points generated by the algorithm also form a point cloud when considered all together. This point cloud is highly accurate, and could allow for explorations into topics such as surface modelling of terrain from a UAV, potentially opening the door to ground vehicles navigating based off of surfaces examined and modelled by UAVs.

# References

[1] Constant False Alarm Rate (CFAR) Detection - MATLAB & Simulink.

[2] Imaging technique acquires a color image and depth map from a single monocular camera image.

[3] Indoor radar SLAM A radar application for vision and GPS denied environments - IEEE Conference Publication.

[4] TI Industrial mmWave Sensors Chirp Design Overview. Technical report.

[5] Francisco A.R. Alencar, Luis Alberto Rosero, Carlos Massera Filho, Fernando S. Osorio, and Denis F. Wolf. Fast Metric Tracking by Detection System: Radar Blob and Camera Fusion. In *Proceedings - 12th LARS Latin American Robotics Symposium and 3rd SBR Brazilian Robotics Symposium, LARS-SBR 2015 - Part of the Robotics Conferences 2015*, pages 120–124. Institute of Electrical and Electronics Engineers Inc., 2 2016.

[6] Mostafa Alizadeh, George Shaker, Joao Carlos Martins De Almeida, Plinio Pelegrini Morita, and Safeddin Safavi-Naeini. Remote monitoring of human vital signs using mm-Wave FMCW Radar. *IEEE Access*, 7:54958–54968, 2019.

[7] Marc Bosch, Kevin Foster, Gordon Christie, Sean Wang, Gregory D. Hager, and Myron Brown. Semantic stereo for incidental satellite images. In *Proceedings - 2019 IEEE Winter Conference on Applications of Computer Vision, WACV 2019*, pages 1524–1532. Institute of Electrical and Electronics Engineers Inc., 3 2019.

[8] Jonas Callmer, David Törnqvist, Fredrik Gustafsson, Henrik Svensson, and Pelle Carlbom. Radar SLAM using visual features. *EURASIP Journal on Advances in Signal Processing*, 2011(1):71, 12 2011.

[9] Alejo Concha and Javier Civera. DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2015-Decem, pages 5686–5693. IEEE, 9 2015.

[10] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 6 2007.

[11] Tobias Deissler and Jorn Thielecke. Feature based indoor mapping using a bat-type UWB radar. In *2009 IEEE International Conference on Ultra-Wideband*, pages 475–479. IEEE, 9 2009.

[12] Ghina El Natour, Omar Ait Aider, Raphael Rouveure, Francois Berry, and Patrice Faure. Radar and vision sensors calibration for outdoor 3D reconstruction. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2015-June, pages 2084–2089. Institute of Electrical and Electronics Engineers Inc., 6 2015.

[13] Ghina El Natour, Omar Ait-Aider, Raphael Rouveure, François Berry, and Patrice Faure. Toward 3D reconstruction of outdoor scenes using an MMW radar and a monocular vision sensor. *Sensors (Switzerland)*, 15(10):25937–25967, 2015.

[14] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, 7 2018.

[15] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-Scale Direct monocular SLAM. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8690 LNCS, pages 834–849. Springer Verlag, 2014.

[16] European Microwave Association., IEEE Antennas and Propagation Society., IEEE Aerospace and Electronic Systems Society., and Germany) European Microwave Week (2013 : Nuremberg. *2013 10th European Radar Conference : 9-11 October 2013, Nuremberg, Germany.* EuMA, 2013.

[17] Markus Falk, Hermann Brugger, and Liselotte Adler-Kastner. Avalanche survival chances. *Nature*, 368(6466):21–21, 3 1994.

[18] Paolo Favaro, Hailin Jin, and Stefano Soatto. A semi-direct approach to structure from motion. In *Proceedings - 11th International Conference on Image Analysis and Processing, ICIAP 2001*, pages 250–255, 2001.

[19] Ning Feng, Yong Zhang, Yuan Xu, and Yueyang Li. Robust Laser Radar-Based Robot Localization Using UFIR Filtering. In *2018 International Conference on Information, Cybernetics, and Computational Social Systems, ICCSS 2018*, pages 473–477. Institute of Electrical and Electronics Engineers Inc., 12 2018.

[20] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Robotics: Science and Systems*, volume 11. Georgia Institute of Technology, 2015.

[21] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22. IEEE, 5 2014.

[22] Xiang Gao, Rui Wang, Nikolaus Demmel, and Daniel Cremers. LDSO: Direct Sparse Odometry with Loop Closure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2198–2204. IEEE, 10 2018.

[23] Franck Gérossier, Paul Checchin, Christophe Blanc, Roland Chapuis, and Laurent Trassoudaine. Trajectory-oriented EKF-SLAM using the Fourier-Mellin transform applied to microwave radar images. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pages 4925–4930, 12 2009.

[24] Fabian Girrbach, Jeroen Hol, Giovanni Bellusci, and Moritz Diehl. Optimization-Based Sensor Fusion of GNSS and IMU Using a Moving Horizon Approach. *Sensors*, 17(5):1159, 5 2017.

[25] Riccardo Giubilato, Sebastiano Chiodini, Marco Pertile, and Stefano Debei. Scale Correct Monocular Visual Odometry Using a LiDAR Altimeter. In *IEEE International Conference on Intelligent Robots and Systems*, pages 3694–3700. Institute of Electrical and Electronics Engineers Inc., 12 2018.

[26] Johannes Graeter, Alexander Wilczynski, and Martin Lauer. LIMO: Lidar-Monocular Visual Odometry. In *IEEE International Conference on Intelligent Robots and Systems*, pages 7872–7879. Institute of Electrical and Electronics Engineers Inc., 12 2018.

[27] Leroy C. Graham. Synthetic Interferometer Radar For Topographic Mapping. *Proceedings of the IEEE*, 62(6):763–768, 1974.

[28] Anna Guerra, Francesco Guidi, Lilia Ahtaryieva, Nicolo Decarli, and Davide Dardari. Crowd-based personal radars for indoor mapping using UWB measurements. In *2016*

*IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*, pages 1–4. IEEE, 10 2016.

[29] J. K. Harmon, M. A. Slade, R. A. Vélez, A. Crespo, M. J. Dryer, and J. M. Johnson. Radar mapping of Mercury's polar anomalies. *Nature*, 369(6477):213–215, 5 1994.

[30] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2D LIDAR SLAM. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2016-June, pages 1271–1278. IEEE, 5 2016.

[31] Martin Holder, Sven Hellwig, and Hermann Winner. Real-Time Pose Graph SLAM based on Radar. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1145–1151. IEEE, 6 2019.

[32] Martin Holder, Sven Hellwig, and Hermann Winner. Real-time pose graph SLAM based on radar. In *IEEE Intelligent Vehicles Symposium, Proceedings*, volume 2019-June, pages 1145–1151. Institute of Electrical and Electronics Engineers Inc., 6 2019.

[33] Harimohan Jha, Vaibhav Lodhi, and Debashish Chakravarty. Object Detection and Identification Using Vision and Radar Data Fusion System for Ground-Based Navigation. In *2019 6th International Conference on Signal Processing and Integrated Networks, SPIN 2019*, pages 590–593. Institute of Electrical and Electronics Engineers Inc., 5 2019.

[34] Harimohan Jha, Vaibhav Lodhi, and Debashish Chakravarty. Object Detection and Identification Using Vision and Radar Data Fusion System for Ground-Based Navigation. In *2019 6th International Conference on Signal Processing and Integrated Networks, SPIN 2019*, pages 590–593. Institute of Electrical and Electronics Engineers Inc., 5 2019.

[35] Zhengping Ji and Danil Prokhorov. Radar-vision fusion for object classification. *Proceedings of the 11th International Conference on Information Fusion, FUSION 2008*, 2:265–271, 2008.

[36] E. Jose and M.D. Adams. Millimetre Wave RADAR spectra simulation and interpretation for outdoor SLAM. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, pages 1321–1326. IEEE, 2004.

[37] E. Jose and M.D. Adams. An augmented state SLAM formulation for multiple line-of-sight features with millimetre wave RADAR. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3087–3092. IEEE, 2005.

[38] Christoforos Kanellakis, Sina Sharif Mansouri, Emil Fresk, Dariusz Kominiak, and George Nikolakopoulos. Cooperative UAVs as a Tool for Aerial Inspection of Large Scale Aging Infrastructure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5040–5040. IEEE, 10 2019.

[39] Daejun Kang and Dongsuk Kum. Camera and Radar Sensor Fusion for Robust Vehicle Localization via Vehicle Part Localization. *IEEE Access*, 8:75223–75236, 2020.

[40] Jonathan Kelly, Srikanth Saripalli, and Gaurav S Sukhatme. Combined visual and inertial navigation for an unmanned aerial vehicle. In *Springer Tracts in Advanced Robotics*, volume 42, pages 255–264, 2008.

[41] Deepak Khosla, David J. Huber, and Yang Chen. Automated scheduling of radar-cued camera system for optimizing visual inspection and detection of radar targets. In *2017 IEEE International Symposium on Technologies for Homeland Security, HST 2017*. Institute of Electrical and Electronics Engineers Inc., 6 2017.

[42] Changhyeon Kim, Pyojin Kim, Sangil Lee, and H. Jin Kim. Edge-Based Robust RGB-D Visual Odometry Using 2-D Edge Divergence Minimization. In *IEEE International Conference on Intelligent Robots and Systems*, pages 6887–6894. Institute of Electrical and Electronics Engineers Inc., 12 2018.

[43] Deok Hwa Kim and Jong Hwan Kim. Effective background model-based RGB-D dense visual odometry in a dynamic environment. *IEEE Transactions on Robotics*, 32(6):1565–1573, 12 2016.

[44] Kyeong Eun Kim, Chang Joo Lee, Dong Sung Pae, and Myo Taeg Lim. Sensor fusion for vehicle tracking with camera and radar sensor. In *International Conference on Control, Automation and Systems*, volume 2017-October, pages 1075–1077. IEEE Computer Society, 12 2017.

[45] Pyojin Kim, Hyon Lim, and H. Jin Kim. Robust visual odometry to irregular illumination changes with RGB-D camera. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2015-December, pages 3688–3694. Institute of Electrical and Electronics Engineers Inc., 12 2015.

[46] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*, 2007.

[47] Jaime Laviada, Ana Arboleya-Arboleya, Yuri Álvarez, Borja González-Valdés, and Fernando Las-Heras. Multiview three-dimensional reconstruction by millimetre-wave portable camera. *Scientific Reports*, 7(1), 12 2017.

[48] Hyukjung Lee, Joohwan Chun, and Kyeongjin Jeon. Autonomous back-in parking based on occupancy grid map and EKF SLAM with W-band radar. In *2018 International Conference on Radar (RADAR)*, pages 1–4. IEEE, 8 2018.

[49] Hyukjung Lee, Joohwan Chun, and Kyeongjin Jeon. Autonomous back-in parking based on occupancy grid map and EKF SLAM with W-band radar. In *2018 International Conference on Radar, RADAR 2018*, volume 2018-January. Institute of Electrical and Electronics Engineers Inc., 12 2018.

[50] Hyukjung Lee, Joohwan Chun, and Kyeongjin Jeon. Experimental Results and Posterior Cramér-Rao Bound Analysis of EKF-Based Radar SLAM with Odometer Bias Compensation. *IEEE Transactions on Aerospace and Electronic Systems*, pages 1–1, 8 2020.

[51] Hyukjung Lee, Joohwan Chun, Kyeongjin Jeon, and Heedeok Lee. Efficient EKF-SLAM Algorithm Based on Measurement Clustering and Real Data Simulations. In *IEEE Vehicular Technology Conference*, volume 2018-August. Institute of Electrical and Electronics Engineers Inc., 7 2018.

[52] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[53] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. UnDeepVO: Monocular Visual Odometry Through Unsupervised Deep Learning. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 7286–7291. Institute of Electrical and Electronics Engineers Inc., 9 2018.

[54] Xiong Liu, Dongying Li, and Wenxian Yu. A Radar-Based Simultaneous Localization and Mapping Paradigm for Scattering Map Modeling. In *2018 IEEE Asia-Pacific Conference on Antennas and Propagation (APCAP)*, pages 534–535. IEEE, 8 2018.

[55] Xiong Liu, Dongying Li, and Wenxian Yu. Radar Simultaneous Localization and Mapping (SLAM) for Stochastic Spread Targets. In *2018 Asia-Pacific Microwave Conference (APMC)*, pages 369–371. IEEE, 11 2018.

[56] Xiong Liu, Dongying Li, and Wenxian Yu. Radar simultaneous localization and mapping (SLAM) for stochastic spread targets. In *Asia-Pacific Microwave Conference Proceedings, APMC*, volume 2018-November, pages 369–371. Institute of Electrical and Electronics Engineers Inc., 1 2019.

[57] Shing Yan Loo, Ali Jahani Amiri, Syamsiah Mashohor, Sai Hong Tang, and Hong Zhang. CNN-SVO: Improving the Mapping in Semi-Direct Visual Odometry Using Single-Image Depth Prediction. In *IEEE International Conference on Robotics and Automation*, pages 5218–5223. Institute of Electrical and Electronics Engineers (IEEE), 8 2019.

[58] Yan Lu and Dezhen Song. Robustness to lighting variations: An RGB-D indoor visual odometry using line segments. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2015-December, pages 688–694. Institute of Electrical and Electronics Engineers Inc., 12 2015.

[59] Fangchang Mal and Sertac Karaman. Sparse-to-Dense: Depth Prediction from Sparse Depth Samples and a Single Image. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4796–4803. Institute of Electrical and Electronics Engineers Inc., 9 2018.

[60] Hidenobu Matsuki, Lukas Von Stumberg, Vladyslav Usenko, Jorg Stuckler, and Daniel Cremers. Omnidirectional DSO: Direct Sparse Odometry with Fisheye Cameras. *IEEE Robotics and Automation Letters*, 3(4):3693–3700, 10 2018.

[61] Malcolm Mielle, Martin Magnusson, and Achim J. Lilienthal. A comparative analysis of radar and lidar sensing for localization and mapping. In *2019 European Conference on Mobile Robots, ECMR 2019 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 9 2019.

[62] John Mullane, Martin D. Adams, and Wijerupage Sardha Wijesoma. Evidential versus Bayesian Estimation for Radar Map Building. In *2006 9th International Conference on Control, Automation, Robotics and Vision*, pages 1–8. IEEE, 12 2006.

[63] Raul Mur-Artal, J. M.M. Montiel, and Juan D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 10 2015.

[64] Raul Mur-Artal and Juan D. Tardos. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 10 2017.

[65] Raul Mur-Artal and Juan D. Tardos. Visual-Inertial Monocular SLAM with Map Reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 4 2017.

[66] Ramin Nabati and Hairong Qi. CenterFusion: Center-based Radar and Camera Fusion for 3D Object Detection. 11 2020.

[67] Richard A. Newcombe, Steven J. Lovegrove, and Andrew J. Davison. DTAM: Dense tracking and mapping in real-time. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2320–2327, 2011.

[68] Felix Nobis, Maximilian Geisslinger, Markus Weber, Johannes Betz, and Markus Lienkamp. A Deep Learning-based Radar and Camera Sensor Fusion Architecture for Object Detection. In *2019 Symposium on Sensor Data Fusion: Trends, Solutions, Applications, SDF 2019*. Institute of Electrical and Electronics Engineers Inc., 10 2019.

[69] Sandra Nowok, Simon Kueppers, Harun Cetinkaya, Martin Schroeder, and Reinhold Herschel. Millimeter wave radar for high resolution 3D near field imaging for robotics and security scans. In *2017 18th International Radar Symposium (IRS)*, pages 1–10. IEEE, 6 2017.

[70] Jiyong Oh, Ki-seok Kim, Miryong Park, and Sungho Kim. A Comparative Study on Camera-Radar Calibration Methods. *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1057–1062, 2018.

[71] Shuaib Omar and Simon Winberg. Multisensor data fusion: Target tracking with a doppler radar and an Electro-Optic camera. In *Proceedings - 2011 IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2011*, pages 210–215, 2011.

[72] Stephan Palm, Rainer Sommer, and Uwe Stilla. Mobile Radar Mapping—Subcentimeter SAR Imaging of Roads. *IEEE Transactions on Geoscience and Remote Sensing*, 56(11):6734–6746, 11 2018.

[73] K. C. Peng, L. Feng, Y. C. Hsieh, T. H. Yang, S. H. Hsiung, Y. D. Tsai, and C. Kuo. Unmanned Aerial Vehicle for infrastructure inspection with image processing for quantification of measurement and formation of facade map. In *Proceedings of the 2017 IEEE International Conference on Applied System Innovation: Applied System Innovation for Modern Technology, ICASI 2017*, pages 1969–1972. IEEE, 5 2017.

[74] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. REMODE: Probabilistic, monocular dense reconstruction in real time. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2609–2616, 2014.

[75] Declan D. G. Radford, Matthew J. Cracknell, Michael J. Roach, and Grace V. Cumming. Geological Mapping in Western Tasmania Using Radar and Random Forests. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(9):3075–3087, 9 2018.

[76] Noha Radwan, Abhinav Valada, and Wolfram Burgard. VLocNet++: Deep Multitask Learning for Semantic Visual Localization and Odometry. *IEEE Robotics and Automation Letters*, 3(4):4407–4414, 10 2018.

[77] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense Monocular Depth Estimation in Complex Dynamic Scenes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, pages 4058–4066. IEEE Computer Society, 12 2016.

[78] R. Rouveure, M. O. Monod, and P. Faure. Mapping of the environment with a high resolution ground-based radar imager. In *Proceedings of the Mediterranean Electrotechnical Conference - MELECON*, pages 822–828. IEEE, 5 2008.

[79] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2564–2571, 2011.

[80] David Sandwell, Rob Mellors, Xiaopeng Tong, Matt Wei, and Paul Wessel. Open radar interferometry software for mapping surface Deformation. *Eos, Transactions American Geophysical Union*, 92(28):234–234, 7 2011.

[81] F. Schuster, C. G. Keller, M. Rapp, M. Haueis, and C. Curio. Landmark based radar SLAM using graph optimization. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2559–2564. IEEE, 11 2016.

[82] Sheng Fu, Hui-ying Liu, Lu-fang Gao, and Yu-xian Gai. SLAM for mobile robots using laser range finder and monocular vision. In *2007 14th International Conference on Mechatronics and Machine Vision in Practice*, pages 91–96. IEEE, 12 2007.

[83] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[84] Soji Shimono, Osam Matsubara, Shigeki Toyama, Uichi Nishizawa, Shin Kato, and Hitoshi Arisumi. Development of underwater inspection system for dam inspection. In *OCEANS 2015 - MTS/IEEE Washington*, pages 1–6. IEEE, 10 2016.

[85] Soji Shimono, Shigeki Toyama, and Uichi Nishizawa. Development of underwater inspection system for dam inspection: Results of field tests. In *OCEANS 2016 MTS/IEEE Monterey, OCE 2016*, pages 1–4. IEEE, 9 2016.

[86] Young Sik Shin, Yeong Sang Park, and Ayoung Kim. Direct visual SLAM using sparse depth for Camera-LiDAR system. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5144–5151. Institute of Electrical and Electronics Engineers Inc., 9 2018.

[87] Roman Streubel and Bin Yang. Fusion of stereo camera and MIMO-FMCW radar for pedestrian tracking in indoor environments. In *FUSION 2016 - 19th International Conference on Information Fusion, Proceedings*, pages 565–572, 2016.

[88] Jan Stühmer, Stefan Gumhold, and Daniel Cremers. Real-time dense geometry from a handheld camera. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6376 LNCS, pages 11–20, 2010.

[89] Dave Tahmoush and Jerry Silvious. Time-integrated range-Doppler maps for visualizing and classifying radar data. In *2011 IEEE RadarCon (RADAR)*, pages 372–374. IEEE, 5 2011.

[90] Levi Valgaerts, Andrés Bruhn, Markus Mainberger, Joachim Weickert, A Bruhn, M Mainberger, · J Weickert, and J Weickert. Dense versus Sparse Approaches for Estimating the Fundamental Matrix. *Int J Comput Vis*, 96:212–234, 2012.

[91] Damien Vivet, Paul Checchin, and Roland Chapuis. Localization and mapping using only a rotating FMCW radar sensor. *Sensors (Basel, Switzerland)*, 13(4):4527–52, 4 2013.

[92] Lukas Von Stumberg, Vladyslav Usenko, and Daniel Cremers. Direct Sparse Visual-Inertial Odometry Using Dynamic Marginalization. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2510–2517, 2018.

[93] Chaoyang Wang, Jose Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning Depth from Monocular Videos Using Direct Methods. In *Proceedings of the IEEE*

*Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2022–2030. IEEE Computer Society, 12 2018.

[94] Rui Wang, Martin Schworer, and Daniel Cremers. Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-October, pages 3923–3931. Institute of Electrical and Electronics Engineers Inc., 12 2017.

[95] Weidong Wu, Murad A. Qurishee, Joseph Owino, Ignatius Fomunung, Mbakisya Onyango, and Babatunde Atolagbe. Coupling Deep Learning and UAV for Infrastructure Condition Assessment Automation. In *2018 IEEE International Smart Cities Conference (ISC2)*, pages 1–7. IEEE, 9 2018.

[96] Ritu Yadav, Axel Vierling, and Karsten Berns. Radar + RGB Fusion for Robust Object Detection in Autonomous Vehicle. In *Proceedings - International Conference on Image Processing, ICIP*, volume 2020-October, pages 1986–1990. IEEE Computer Society, 10 2020.

[97] Dong Won Yang, Seok Jae Lee, Tae Ha Kang, Joo Hong Yoon, and Jung Ho Ko. An improved classification method of concealed obstacles using UWB radar and stereo cameras. In *2011 3rd International Asia-Pacific Conference on Synthetic Aperture Radar, APSAR 2011*, pages 125–128, 2011.

[98] Nan Yang, Rui Wang, Jörg Stückler, and Daniel Cremers. Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11212 LNCS, pages 835–852. Springer Verlag, 2018.

[99] Xiaochuan Yin, Xiangwei Wang, Xiaoguo Du, and Qijun Chen. Scale Recovery for Monocular Visual Odometry Using Depth Estimated with Deep Convolutional Neural Fields. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5871–5879, 2017.

[100] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics. *Intelligent Industrial Systems*, 1(4):289–311, 12 2015.

[101] Howard A. Zebker and Richard M. Goldstein. TOPOGRAPHIC MAPPING FROM INTERFEROMETRIC SYNTHETIC APERTURE RADAR OBSERVATIONS. In

*Digest - International Geoscience and Remote Sensing Symposium (IGARSS)*, volume 91, pages 113–117. John Wiley & Sons, Ltd, 4 1985.

[102] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian M. Reid. Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 340–349, 2018.

[103] Ji Zhang and Sanjiv Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2015-June, pages 2174–2181. Institute of Electrical and Electronics Engineers Inc., 6 2015.

[104] Jixian Zhang, Shucheng Yang, Zheng Zhao, and Guoman Huang. SAR mapping technology and its application in difficulty terrain area. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 3608–3611. IEEE, 7 2010.

[105] Yongchao Zhang, Yin Zhang, Yulin Huang, and Jianyu Yang. Superresolution ground mapping for scanning radar with IAA-based inverse filtering. In *2016 CIE International Conference on Radar (RADAR)*, pages 1–4. IEEE, 10 2016.

[106] Zichao Zhang and Davide Scaramuzza. A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry. In *IEEE International Conference on Intelligent Robots and Systems*, pages 7244–7251. Institute of Electrical and Electronics Engineers Inc., 12 2018.

[107] Junping Zhong, Zhigang Liu, Zhiwei Han, Ye Han, and Wenxuan Zhang. A CNN-Based Defect Inspection Method for Catenary Split Pins in High-Speed Railway. *IEEE Transactions on Instrumentation and Measurement*, 68(8):2849–2860, 8 2019.

[108] Yi Zhou, Hongdong Li, and Laurent Kneip. Canny-VO: Visual Odometry with RGB-D Cameras Based on Geometric 3-D-2-D Edge Alignment. *IEEE Transactions on Robotics*, 35(1):184–199, 2 2019.

# APPENDICES

# Appendix A

# Radar Signal Processing

In this section, we present details about the type of radar used, and the radar signal processing methods used in order to create a point cloud approximation.

## A.1 Frequency Modulated Continuous Wave Radar

The type of radar used for this experiment is known as FMCW. It emits a radar pulse of continuous linearly increasing frequency, known as a chirp. Upon receiving a return signal, the difference between the frequency currently being emitted and the frequency of the received signal are compared, and the difference between them can be used to determine the distance of the object that the radar reflected off of, according to the formula

$$R = \frac{c_0|\Delta f|}{2\frac{df}{dt}}. \tag{A.1}$$

The rate of frequency increase over time is known, as is $c_0$, the speed of light. This allows for a precise measurement of range, and is the basic formula on which FMCW radar is based. A single pulse is normally known as a chirp, and chirps are typically organised into frames.

In practice, a FMCW radar is normally implemented using sine waves of increasing frequency, with a circuit schematic similar to the one shown in Figure A.1. The inputs to the mixer are typically two sine waves, defined as

$$x_1 = \sin[w_1 t + \phi_1], x_2 = \sin[w_2 t + \phi_2], \tag{A.2}$$

and with the output of the mixer being

$$x_{out} = \sin[(w_1 - w_2)t + (\phi_1 - \phi_2)]. \tag{A.3}$$

$x_{out}$ is also known as the Intermediate Frequency (IF) signal, If a Fast Fourier Transform (FFT) is performed on the IF, then the resulting function would have peaks at frequencies corresponding to distances of detected objects from the radar as depicted by Equation A.1. This signal tends to have many peaks, however, as a result of noise in both detections and circuitry. This necessitates some form of threshold, to determine which peaks correspond to objects, and which ones are only noise. The method used for this thesis is discussed in Section A.2.

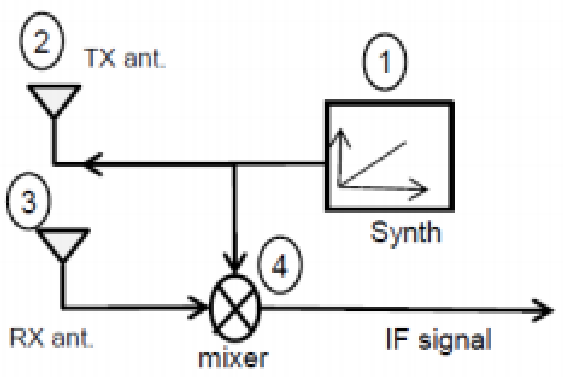


Figure A.1: A single output, single output FMCW radar [4].

FMCW radar has two distinct advantages for UAV applications. Firstly, the use of frequency modulation allows for extremely precise measurements. FMCW radar has a minimal measurable range comparable to the largest wavelength of the transmitted wave. In this thesis, the wavelength used varies from 77 to 81 GHz, so the minimal measurable range is approximately 0.3cm. Second, outside of the actual mixing and receiving at the radar antenna, all the signal processing involved is performed at a significantly lower frequency than the transmitted wavelength, which allows for significantly less complex circuitry and lower power consumption in the signal processing itself.

## A.2 Object Detection using Radar

As mentioned before, a threshold is needed to determine what is noise, and what is an object. The naive implementation is to set a static threshold, but that is clearly not ideal, as the level of noise present can change depending on the environment. A more robust approach is the use of CFAR detection, which uses an adaptive threshold, defined as

$$T = \alpha P_n, \tag{A.4}$$

where $\alpha$ is a scaling factor known as the threshold factor, and $P_n$ is the estimate of noise power. In order to use CFAR detection, the signal must first be divided into cells. These cells are then individually tested to see if they represent a target or not.

The power of a set number of cells around the Cell Under Test (CUT) are averaged in order to determine the noise estimate $P_n$. These cells are known as training cells. However, a few cells on either side of the CUT are not included in this average, in order to keep the estimate from being effected by the CUT. These cells are known as guard cells.

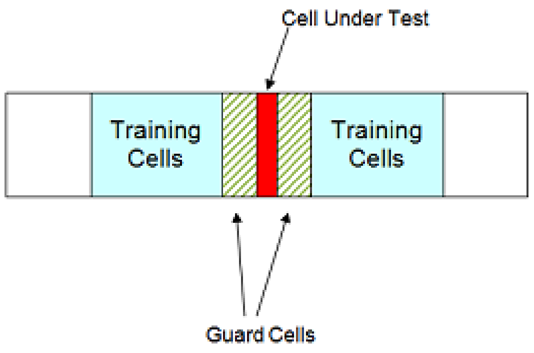


Figure A.2: A diagram illustrating CFAR estimation. [1]

As part of the process of performing CFAR detection, a parameter known in literature as the desired false alarm rate $P_{fa}$ must also be specified. Note that this isn't necessarily exactly the probability that any detection will be a false alarm, but is instead more of a tuning parameter, tuned to determine the sensitivity of the system to noise. We use $P_{fa}$ to calculate

$$\alpha = N(P_{fa}^{\frac{-1}{N}} - 1), \tag{A.5}$$

where $N$ is the number of training cells used, which then can be used with Equation A.4 in order to calculate the threshold for the cell under test.

## A.3 Angle of Arrival Estimation

All of the work shown thus far has been specifically for single transmitter and single receiver models. These are simple, but don't give anything beyond a range. In order to calculate a precise position in three dimensions, more information is needed. Specifically, both azimuth and elevation angles must be determined. For this purpose, the radar used in this thesis has 4 Receiving (Rx) antenna, and 3 Transmit (Tx) antenna. The angle can then be estimated by receiving the reflected signal from the obstacle using multiple receivers, spaced apart with a constant distance $d$. The arriving signal will then be delayed by $dsin(\theta)$ at each antenna, which causes a phase shift of

$$\frac{2\pi d \sin(\theta)}{\lambda}. \tag{A.6}$$

This can then be used to estimate an angle in a single plane. An example is shown in Figure A.3. Using antennae spaced both vertically and horizontally, the process can be repeated to obtain a second estimated angle, to represent an elevation angle. With two angles and a range, we can employ basic trigonometry to determine coordinates for each detected object, and use that information to form a point cloud.

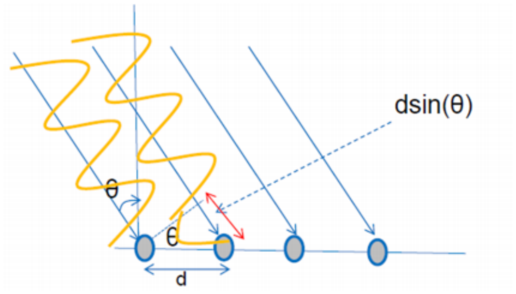


Figure A.3: A diagram illustrating the geometry used for estimating an angle $\theta$. [4]

# Glossary

**OpenCV** Open Computer Vision. A widely used collection of computer vision algorithms consolidated into a single software package. 28